ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS

INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES

CARRERA DE INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

TRABAJO DE TITULACIÓN, PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

TEMA: DISEÑO E MPLEMENTACIÓN DE UN SISTEMA DE CLASIFICACIÓN DE OBJETOS BASADO EN EL ROBOT HUMANOIDE NAO

AUTOR: ENRÍQUEZ RODRÍGUEZ, KATHERINE ANDREA

DIRECTOR: ING. IBARRA JÁCOME, OSWALDO ALEXANDER MGs.

SANGOLQUÍ

2019

ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS

INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES

CARRERA DE INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

CERTIFICACIÓN

Certifico que el trabajo de titulación, "**DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE CLASIFICACIÓN DE OBJETOS BASADO EN EL ROBOT HUMANOIDE NAO**" fue realizado por la señorita ***Enríquez Rodríguez, Katherine Andrea*** el mismo que ha sido revisado en su totalidad, analizado por la herramienta de verificación de similitud de contenido; por lo tanto cumple con los requisitos teóricos, científicos, técnicos, metodológicos y legales establecidos por la Universidad de Fuerzas Armadas ESPE, razón por la cual me permito acreditar y autorizar para que lo sustente públicamente.

Sangolqui, Julio 05 de 2019

Firma:

.....

Ing. Ibarra Jácome Oswaldo Alexander MGs.

C. C. 171953542-7

UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES

CARRERA DE INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

AUTORÍA DE RESPONSABILIDAD

Yo, ***Enríquez Rodríguez, Katherine Andrea*** declaro que el contenido, ideas y criterios del trabajo de titulación: **DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE CLASIFICACIÓN DE OBJETOS BASADO EN EL ROBOT HUMANOIDE NAO** es de mi autoría y responsabilidad, cumpliendo con los requisitos teóricos, científicos, técnicos, metodológicos y legales establecidos por la Universidad de Fuerzas Armadas ESPE, respetando los derechos intelectuales de terceros y referenciando las citas bibliográficas. Consecuentemente el contenido de la investigación mencionada es veraz.

Sangolquí, Julio 05 de 2019

Firma:

.....

Enríquez Rodríguez Katherine Andrea

C. C. 1727296418

UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES

CARRERA DE INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

AUTORIZACIÓN

Yo, ***Enríquez Rodríguez, Katherine Andrea*** autorizo a la Universidad de las Fuerzas Armadas ESPE publicar el trabajo de titulación: **DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE CLASIFICACIÓN DE OBJETOS BASADO EN EL ROBOT HUMANOIDE NAO** en el Repositorio Institucional, cuyo contenido, ideas y criterios son de mi responsabilidad.

Sangolquí, Julio 05 de 2019

Firma:

.....

Enríquez Rodríguez Katherine Andrea

C. C. 1727296418

DEDICATORIA

A mis padres y a mis abuelos, que han estado anhelando este momento desde mi primer día de universidad. A mi hermana, para que al fin pueda decir que soy ingeniera.

Y en especial, a los que desde el primer momento dijeron que no lo iba a lograr.

AGRADECIMIENTO

En primer lugar agradezco a Dios, al universo y a todas las energías que nos gobiernan, por ayudarme a entender el mundo y permitirme seguir adelante con sabiduría.

A mi tutor de tesis, Ing. Alexander Ibarra, por haberme enseñado y apoyado no solo durante el proceso de tesis, si no desde hace mucho tiempo atrás cuando era mi profesor. Gracias por ser ese gran docente, gran tutor y sobretodo gran amigo.

A mis padres, Adrián y Rocío por ser mis consejeros y por ser un pilar fundamental en cada etapa de mi vida, es por ustedes que siempre me mantuve en pie, sean cuales sean las circunstancias y principalmente es por ustedes que me he convertido en lo que soy ahora. Gracias por tanto!

A mis abuelos, por llenarme de amor incondicional día a día.

A mis amigos de la universidad, gracias por haberme acompañado en este arduo proceso lleno de aprendizajes, por los buenos y malos momentos, porque es de ellos que aprendemos más.

Finalmente, gracias a la vida por permitirme soñar y por dejarme cumplir con todos mis sueños.

INDICE DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

ÍNDICE DE ANEXOS

RESUMEN

La robótica humanoide revolucionó a la sociedad desde su aparición y ha sido objeto de estudio durante varios años, enfocando principalmente hacia el aspecto social y hacia la relación e interacción que podría desarrollarse entre los humanos y los robots. En el presente documento se muestra el desarrollo de un sistema de clasificación de objetos por color basado en el robot humanoide NAO, para el desarrollo de esta aplicación se dividió el proceso en tres etapas: el reconocimiento de objetos, en donde se realiza la búsqueda de objetos de interés en base al color, dichos objetos se encontrarán en un medio aislado con iluminación controlada junto con el robot en un área de 1,35m de ancho y 1,5m de largo; a continuación la etapa de seguimiento, en la que se calcula la distancia y la posición relativa del objeto de interés con respecto al robot para que el humanoide pueda acercarse e interactuar con dicho objeto, en esta fase se desarrollaron dos algoritmos matemáticos para los cálculos correspondientes y finalmente la etapa de manipulación, encargada de tomar el objeto con el actuador del brazo izquierdo del humanoide y proceder con la clasificación y depósito del mismo en el contenedor que corresponda. Este proyecto se realizó en lenguaje de programación Python, utilizando el NAOqi framework y fue implementado mediante el software Choregraphe.

PALABRAS CLAVE:

- **ROBÓTICA SOCIAL**
- **ROBOT NAO**
- **CLASIFICACIÓN DE OBJETOS**
- **MANIPULACIÓN DE OBJETOS**
- **SEGUIMIENTO A OBJETOS**

ABSTRACT

Humanoid robotics revolutionized society since its appearance and has been studied for several years, focusing mainly on the social aspect and the relationship and interaction that could develop between humans and robots. In the present document the development of an object classification system by color based on the humanoid robot NAO is shown, for the development of this application the process was divided into three stages: object recognition, where the search for objects of interest based on color takes place, these objects will be found in an isolated environment with controlled lighting together with the robot in an area of 1.35m wide and 1.5m long; then the tracking stage, which calculates the distance and relative position of the object of interest with respect to the robot so that the humanoid can approach and interact with that object, in this phase two mathematical algorithms were developed for the corresponding calculations, and finally the manipulation stage, in charge of taking the object with the actuator of the humanoid's left arm and proceeding with the classification and deposit of the same in the corresponding container. This project was made in the Python programming language, using the NAOqi framework and was implemented using the Choregraphe software.

KEY WORDS:

- **SOCIAL ROBOTICS**
- **NAO ROBOT**
- **OBJECT CLASSIFICATION**
- **OBJECT ROBOTIC MANIPULATION**
- **OBJECT TRACKING**

CAPÍTULO I

GENERALIDADES

1.1 Antecedentes

En los últimos años, las instituciones de investigación y enseñanza han adoptado e incorporado robots de tipo humanoide para el desarrollo de innovación y apropiación social de la tecnología. (Lopez-Caudana, Quiroz, Rodríguez, Yépez, & Ibarra, 2017)

La robótica humanoide ha sido utilizada en varias áreas de investigación desde aplicaciones médicas, como la asistencia a pacientes y la ejecución de terapias, hasta el aspecto industrial, realizando actividades en ambientes potencialmente peligrosos. Para permitir a los robots la interacción con un ambiente real es importante que estos posean capacidades de percepción de las señales del medio en el cual se encuentran, siendo la visión el sentido más importante en términos de movimiento e interacción. (Hernández , Gómez, Crespo , & Barber , 2016) Es así que las técnicas de visión artificial son claves para el desarrollo de proyectos con androides NAO en los cuales la temática principal es la toma de decisiones.

El problema de la toma de decisiones está presente en muchas áreas de la robótica, desde el control de movimientos de robots hasta la robótica cognitiva, pero su tratamiento casi nunca se aborda de forma integrada. El estudio y, por tanto, las soluciones, suelen particularizarse para situaciones muy concretas. Por ejemplo, el

tratamiento que se da al problema de cómo un robot móvil puede realizar tareas de clasificación. (Salichs, Malfaz, & Gorostiza, 2010)

La clasificación de objetos involucra un conjunto de procesos fundamentales como búsqueda y reconocimiento, desplazamiento hacia un objeto determinado y manipulación.

En el ámbito del reconocimiento de objetos se han realizado estudios como el mostrado en (Hernández , Gómez, Crespo , & Barber , 2016), en el cual se implementó un sistema de visión que permite detectar objetos y reconocer su localización dentro de un medio cotidiano mediante la cámara integrada en un robot móvil. De igual forma se han realizado diversos análisis mediante técnicas de filtrado RGB para la localización de objetos como se detalla en (ROSAS-ARIAS, VALLEJO-MERAZ, PÉREZ-BAILÓN, & ROJAS-CID, 2017).

Entre otros trabajos, en (Monzón, 2014) se detalla un sistema de reconocimiento y movilización de objetos creado con un manipulador y la cámara web de un computador portátil, a través del cual se realizan procesos de visión artificial. Se identifica un trabajo similar en (Zurita Pérez, 2014) realizado en la Universidad de las Fuerzas Armadas – Extensión Latacunga.

Además, en (Lopez-Caudana, Quiroz, Rodríguez, Yépez, & Ibarra, 2017) se realiza la clasificación y traslado de objetos en base a las señales acústicas generadas por las características intrínsecas del material del cual están compuestos, donde este sistema

tiene la capacidad de diferenciar entre aluminio, cartón o plástico de acuerdo a la frecuencia de la onda resultante de golpear dicho elemento repetidas veces.

El Departamento de Eléctrica, Electrónica y Telecomunicaciones de la Universidad de las Fuerzas Armadas ESPE posee actualmente a disposición de los estudiantes y profesores el robot humanoide NAO, ideal para varias aplicaciones en diversas áreas como medicina, educación, e investigación como el tópico más importante. (NAO Los robots del futuro son ya una realidad, s.f.) Dado que los prototipos fueron adquiridos recientemente, existen pocos proyectos de investigación en curso con los mismos, es por este motivo que se plantea desarrollar un sistema de clasificación de objetos utilizando el androide.

1.2 Justificación e importancia

El Plan Nacional de Desarrollo 2017-2021 denominado Toda una Vida es el resultado de la incorporación de las propuestas presentadas en más de 380 mesas de diálogo realizadas a lo largo del país y fue aprobado bajo por el Consejo Nacional de Planificación el 22 de septiembre de 2017. El plan se organiza en tres ejes programáticos y en nueve objetivos nacionales de desarrollo, sobre la base de sustentabilidad ambiental y el desarrollo territorial. (Dirección de Comunicación SENPLADES, 2017)

Enfatizando el tema de la tecnología y la investigación, el eje N° 2: *Economía al Servicio de la Sociedad*, define el Objetivo 5*: Impulsar la productividad y competitividad para el crecimiento económico sostenible de manera redistributiva y solidaria,* el cual plantea como una política de acción: '*Promover la investigación, la formación, la*

capacitación, el desarrollo y la transferencia tecnológica, la innovación y el emprendimiento, la protección de la propiedad intelectual, para impulsar el cambio de la matriz productiva mediante la vinculación entre el sector público, productivo y las universidades.', además establece *'Aumentar el número de publicaciones científicas a 2021'* como una de las metas definidas ((CNP), 2017), por lo cual el tema planteado se sustenta en políticas gubernamentales.

En la actualidad se ha generado un interés en los proyectos de clasificación y movilización de objetos con sistemas automáticos dada la importancia que se observa en las aplicaciones de este concepto que varían desde uso industrial en logística de cargas hasta en medicina. (Castro Díaz, 2016)

Hasta el momento, los trabajos de investigación en cuanto a la clasificación de objetos se han realizado en su mayoría mediante un manipulador robótico y una cámara adicional localizada cerca él. Sin embargo, su principal inconveniente reside en que son sistemas estáticos, lo que significa que los objetos con los cuales se trabaja deben estar localizados dentro de su volumen de trabajo; caso contrario no podrán ser manipulados por el robot.

La importancia del presente proyecto reside en la creación de un sistema autónomo de clasificación de objetos, es decir, que no requiera de hardware adicional y que pueda movilizarse libremente; es por este motivo que los robots humanoides NAO son ideales para ello.

La implementación de un sistema de clasificación de objetos mediante visión artificial utilizando los androides NAO, representa la integración de los estudios previamente realizados en el ámbito del análisis y procesamiento de imágenes, junto con las nuevas tecnologías adquiridas por la universidad. A su vez, al ser un sistema móvil, será la base que permitirá explorar y desarrollar actividades en una mayor cantidad de campos de aplicación a lo largo de la industria y de la sociedad.

Además, el presente proyecto aportará al desarrollo académico de estudiantes de pregrado puesto que en la malla curricular de la carrera de Ingeniería en Electrónica y Automatización consta la asignatura de Robótica Social. Asimismo, fomentará el desarrollo en investigación para estudiantes de postgrado.

1.3 Definición del Proyecto

El presente proyecto de titulación pretende realizar el diseño e implementación de un sistema que permita clasificar objetos de acuerdo a su color mediante el uso de la cámara de un robot humanoide NAO, utilizando visión artificial. Además, se prevé el uso de sensores propios del androide para la navegación en entornos.

El sistema operará de la siguiente manera:

1. Inicialmente el robot se encontrará a la espera de la orden de inicio, como se muestra en la Figura 2.

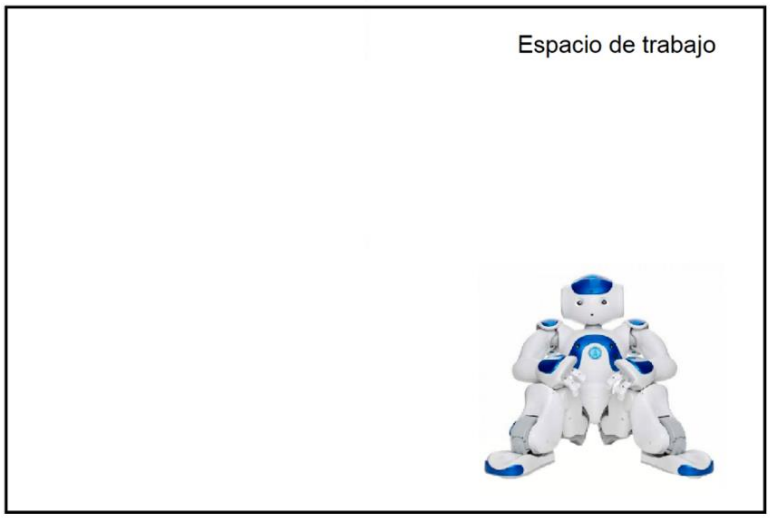

Figura 1. Robot NAO en espera de la orden de inicio

2. Una vez iniciado el sistema, el robot realizará una búsqueda visual a su alrededor hasta encontrar objetos de interés dentro del entorno que lo rodea, reconocerlos y finalmente clasificarlos.

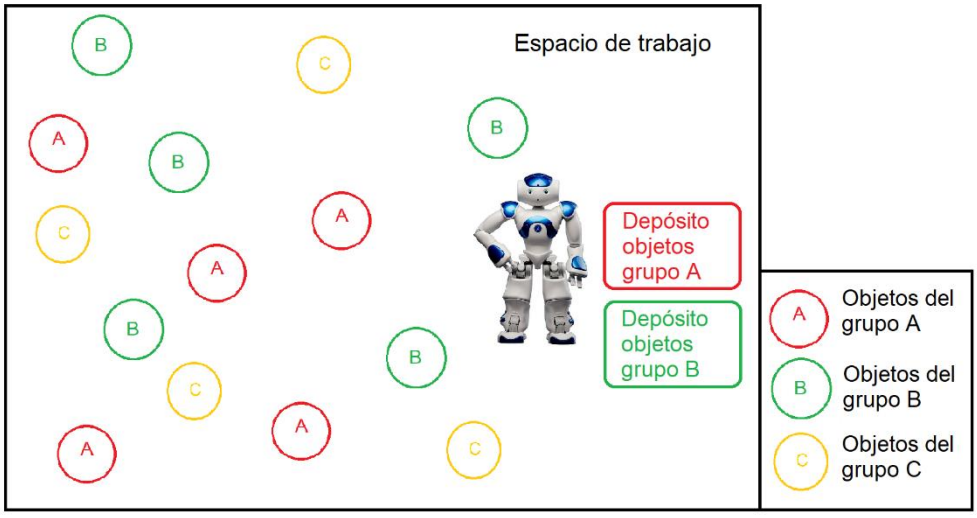

Figura 2. Robot NAO inicia con la búsqueda de objetos de interés

En la Figura 2 se puede observar un ejemplo del androide localizado en un área de trabajo, en la cual se encuentran varios objetos dispuestos en posición aleatoria. Existen

3 clases distintas marcadas como A, B y C representados por círculos de distintos colores y dos contenedores para los objetos de clase A y B.

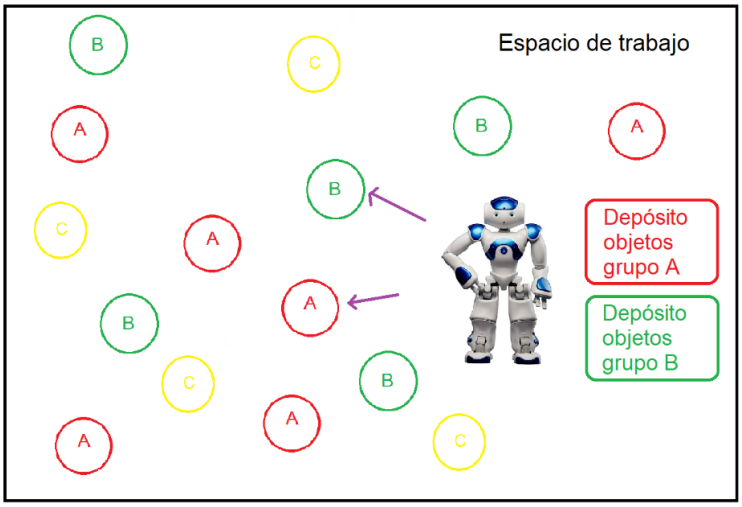

Figura 3. Robot detecta los objetos de interés presentes en el medio

3. Una vez que se ha detectado un objeto de interés (Figura 3), el androide se trasladará hacia él (Figura 4). En caso de existir más de un objeto de interés cerca al androide, se dirigirá hacia el primero que sea detectado.

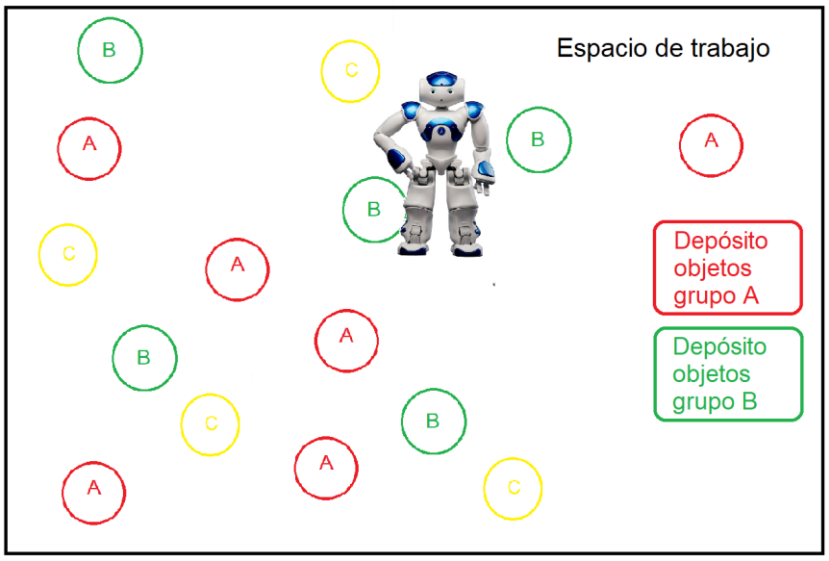

Figura 4. Robot NAO se dirige hacia un objeto de interés

4. Posterior a ello, el androide tomará el objeto y se dirigirá hacia el depósito al cual corresponda (Figura 5).

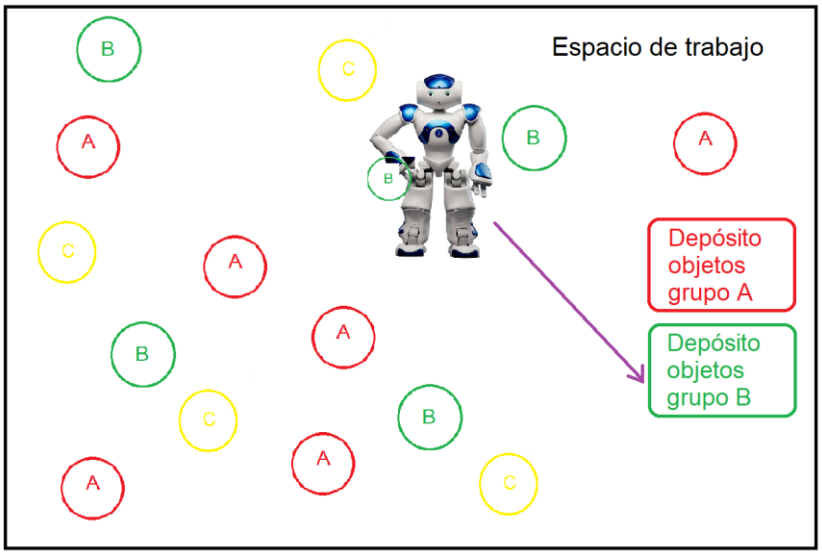

Figura 5. Proceso de análisis para la clasificación del objeto

5. Una vez alcanzado el nuevo destino, el androide procede a depositar el objeto. (Figura 6).

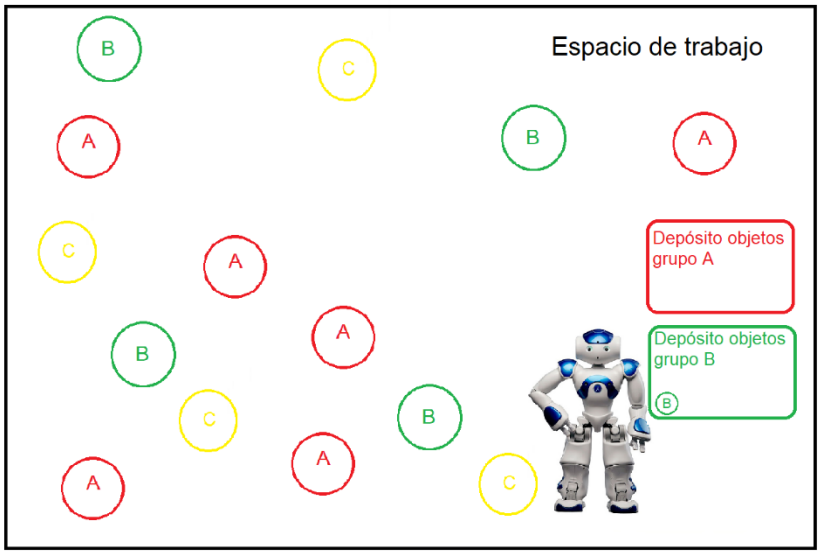

Figura 6. El androide coloca el objeto en el depósito

6. Este proceso se repetirá mientras existan objetos de interés en el medio en el que se encuentra el androide. En caso de no existir más objetos, el androide permanecerá en un estado de espera (Figura 7).

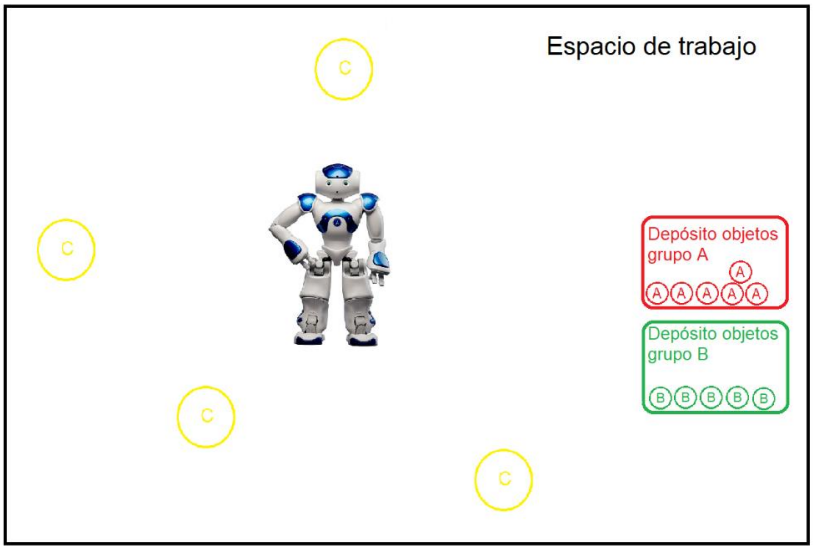

Figura 7. Espera para la identificación de nuevos objetos de interés

7. El usuario del androide podrá finalizar la tarea en el momento que desee.

1.4 Alcance

El presente proyecto constará del desarrollo de una aplicación para el robot humanoide NAO, que permita clasificar distintos tipos de objetos presentes en el medio, utilizando el sistema de visión que posee el androide.

La finalidad es crear un sistema que tenga la capacidad de clasificar un conjunto de objetos de forma autónoma, de acuerdo con lineamientos predefinidos por un usuario; es decir, utilizando únicamente el sistema de visión que posee el robot sin la necesidad de utilizar hardware adicional.

Como se definió anteriormente, la actividad de clasificación de objetos contiene inmersos 3 procesos: búsqueda y reconocimiento de objetos, desplazamiento hacia el objeto, y la manipulación del mismo.

El robot tendrá la capacidad de reconocer mediante una búsqueda visual, los objetos de interés (datos definidos previamente) que se encuentren localizados en su alrededor y dentro del rango de visión del mismo; a continuación procederá a acercarse hacia él (desplazamiento hacia el objeto).

Una vez que el androide ha alcanzado al objeto, continúa con la manipulación del mismo para poder realizar el transporte hacia el depósito correspondiente. Esto se realizará con todos los objetos que estén en la zona antes mencionada.

En caso de haber finalizado la clasificación de los elementos disponibles, el robot esperará a que aparezcan nuevos objetos de interés o que el usuario decida terminar la tarea.

1.5 Objetivos

1.5.1 Objetivo General

Diseñar e implementar un sistema basado en el robot humanoide NAO que permita la clasificación de diversos objetos presentes en el medio.

1.5.2 Objetivos específicos

- Programar el funcionamiento del robot humanoide NAO mediante lenguaje de programación libre para crear módulos que puedan ejecutarse de forma remota y local.

- Desarrollar e implementar un algoritmo de reconocimiento de objetos mediante visión artificial para extraer la información necesaria del medio en el cual se encuentra el androide.
- Implementar un algoritmo de desplazamiento hacia objetos con el robot humanoide NAO.
- Implementar un algoritmo de recolección de objetos con el robot humanoide NAO.

CAPÍTULO II

FUNDAMENTACIÓN TEÓRICA

2.1 Introducción

El año 1921 revolucionó a la sociedad con el aparecimiento del término robot, proveniente de la palabra checa ‘robota’, que significa *trabajo obligatorio*, la cual fue utilizada durante una obra teatral de Karel Capek. Posteriormente en el año 1939 Isaac Asimov se encargó de acuñar el término robótica y planteo las ‘tres leyes de la robótica’ iniciando una nueva tendencia en el mundo tecnológico.

A partir de ese momento empezó el desarrollo de una ciencia que hasta el momento no había sido explorada, dando como resultado robots que facilitaban los procesos de producción en distintos tipos de industrias, cuyos movimientos eran programados a precisión y planeados desde un inicio.

La definición de robot propuesta por el Robot Industries Association (RIA), establece que: “Un robot es un manipulador funcional reprogramable, capaz de mover material, piezas, herramientas o dispositivos especializados mediante movimientos variables programados, con el fin de realizar tareas diversas”; cabe destacar que un robot es multifuncional y ofrece programabilidad, es decir tiene la capacidad de adaptarse a su medio y de modificar la tarea que realiza. (Brady & Paul, 1984)

De acuerdo con su arquitectura, los robots se clasifican en:

- Humanoides, como su nombre lo indica tienen forma humana y tienden a replicar su comportamiento.
- Móviles, se desplazan mediante ruedas.
- Zoomórficos, sistema de locomoción imitando a los animales.
- Poliarticulados, principalmente utilizados en el ámbito industrial.

Figura 8. Tipos de robots por su fisionomía.

En la actualidad las aplicaciones principales de la robótica son:

- Industrial, se encargan de realizar tareas en el ámbito productivo.
- De servicio o social, generalmente robots autónomos utilizados en tareas domésticas, de investigación o exploración.
- Militares, su función principal es asistir a los ejércitos en sus operaciones
- Médicos, se encargan de tareas que varían desde la asistencia a profesionales hasta la terapia de pacientes. (EBS Robótica, 2018)

Como es el objeto del presente proyecto, en las secciones 2.4 y 2.5 se detallará más a fondo los robots humanoides y se realizará un acercamiento a la robótica social.

2.2 La robótica en la industria 3.0

La industria 3.0 llegó con la electrónica, la automatización y generó los primeros pasos de la robótica dentro de los procesos industriales para incrementar la eficiencia y convertirlos en procesos más seguros.

En esta etapa de la industria, el objetivo principal del uso de la robótica en los procesos de fabricación era realizar las labores pesadas y/o peligrosas para los obreros. En la Figura 9 se puede observar un brazo robótico utilizado para el transporte de cargas pesadas en un centro de distribución.

Figura 9. Brazo robótico encargado del movimiento de cargas pesadas.
Fuente: (Merrill, 2018)

2.3 La robótica en la industria 4.0

La industria 4.0 está directamente ligada con tecnologías como el IdC (Internet de las Cosas), *Big Data*, *Cloud Computing,* Realidad virtual y aumentada, Inteligencia artificial y Robótica Avanzada.

En el ámbito de la robótica en la industria 4.0, se espera que los robots colaboren entre ellos y con los operadores e ingenieros durante los procesos que se ejecuten, esto se denomina, robótica colaborativa y es uno de los pilares fundamentales para el desarrollo de la industria 4.0 (Pelegrí, 2019), para que la relación humano-robot y robot-robot sea una realidad se han ido desarrollando diversas técnicas de visión e inteligencia artificial, permitiendo que los robots puedan tomar decisiones en base a la información que recepten del medio en el que se encuentran, por ejemplo, dotarlos de la capacidad de inspeccionar productos defectuosos y tomar acciones de corrección. (Martell Chávez, 2018).

De igual forma, otro enfoque para la industria 4.0 es el *Industrial Internet of Things,* mejor conocido como IIoT, el cual busca dotar de flexibilidad a las empresas para que las fábricas puedan trabajar de forma autónoma reduciendo el número de obreros. En la Figura 10 se puede observar un ejemplo de interacción robot-robot.

Figura 10. Brazos robóticos trabajando en conjunto

En comparación con la etapa anterior, en la 4ta revolución industrial, los robots no serán utilizados únicamente en labores pesadas y peligrosas, sino también en labores del día a día, que hasta ahora han sido realizadas por los operarios y supervisores humanos. (Martin, 2016)

2.4 Robótica Social

La robótica social está introduciéndose a un ritmo moderado dentro de la sociedad humana formando parte de la normalidad del día a día, permitiendo así que el contacto y la interacción humano-robótica se produzcan a mayor escala, incluyendo así poco a poco a los robots en la vida social humana. (Castro-Gonzáles, Martín, Castillo, & Salichis, 2017)

Pero, ¿qué quiere decir que un robot es social? La sociabilidad implica la existencia de relaciones de interacción, es por ello que Diaz & Chaves definen a la robótica social como un campo de la robótica cuyo fin es el de crear robots capaces de interactuar y comunicarse con los seres humanos siguiendo comportamientos, patrones y normas sociales. Más allá de ello, la robótica social persigue en la actualidad el objetivo de entender cómo los robots pueden llegar a ser parte de la vida cotidiana de las personas y cómo pueden colaborar con ellas, dentro de las estructuras sociales y culturales existentes.

Figura 11. Robótica social
Fuente: (Vanderkley, 2015)

Este término engloba todas las plataformas que dan forma a este fin y se encuentra en constante crecimiento aportando resultados positivos. La importante inversión concedida a la robótica social en los últimos años hace que se hayan aumentado las líneas de investigación y se centre en diferentes sectores. (Díaz & Chaves, 2017)

2.4.1 Aplicaciones

Las aplicaciones más comunes de los robots sociales son diversas, una de ellas es el entretenimiento, un claro ejemplo de ello es Aibo de la empresa Sony (Figura 13).

Figura 12. Aibo de Sony

Otro tema ampliamente investigado es la compañía a personas mayores o con discapacidades, en la Figura 13 se puede observar un prototipo de la empresa china Siasun presentado en el año 2017, el cual es un robot diseñado especialmente para este fin, cuyas características principales son el desplazamiento dentro de una vivienda y toma de imágenes que permiten tareas de vigilancia y cuidado.

Figura 13. Robots cuidan adultos mayores.
Fuente: (teleSUR - ap - HR, 2017)

En el ámbito de la educación se han desarrollado plataformas que interactúen tanto con los estudiantes como con los docentes, un ejemplo de ello se evidencia en la investigación “Robots sociales en la escuela” realizada por Díaz, Amara, Casacuberta, & Angulo, que concluyó que los niños presentan satisfacción y un alto interés por sus clases cuando se relacionan con robots. El robot NAO es un robot creado principalmente para aplicaciones educativas (Dsouza, 2016) , véase la Figura 5.

Sin embargo, existen otros escenarios en los que la robótica social está inmiscuida, por ejemplo, en el ámbito comercial y productivo (Abad Sacoto, Sánchez Delgado, Crespo Cedeño, & Alvarado Chang, 2017), en la Figura 15 se puede observar

a Pepper, un robot humanoide encargado de brindar apoyo dentro de un supermercado, ofreciendo información a sus visitantes.

Figura 14. Robot NAO en la enseñanza.
Fuente: (AliveRobotics, s.f.)

Figura 15. Pepper atendiendo en un supermercado.
Fuente:(Chuet-Missé, 2017)

Por otro lado, los robots sociales han sido utilizados como herramientas complementarias en rehabilitación así como en terapias para personas con autismo (Díaz, Amara, Casacuberta, & Angulo, 2009), como se puede apreciar en la Figura 16, Baxter es un robot creado en la escuela de medicina Perelman que permite realizar

rehabilitación, terapia y vigilancia, permitiendo a los doctores analizar y monitorear el progreso de los pacientes de forma remota.

Figura 16. Baxter, para rehabilitación y monitoreo de pacientes.

Además de los mencionados anteriormente, Abad Sacoto, Sánchez Delgado, Crespo Cedeño, & Alvarado Chang, en el artículo Sistemas de reconocimiento en la robótica social, mencionan que los escenarios de conflicto actuales incrementan los incentivos en el ámbito de la investigación y desarrollo militar así como en tareas de rescate o defensa, en la Figura 17 se puede observar un claro ejemplo de este tipo de robots: Amitiel es un robot explorador que ser utilizado en servicios de rescate mediante teleoperación que posee cámaras inalámbrica, puede llevar hasta 20kg de peso internamente, soporta golpes de 40 kg y peso de 50 kg sobre él. (Entrevista a Sebastián Rojas, 2016)

Como se ha podido comprobar la robótica social se encuentra cada vez más presente en nuestro uso diario, proporcionándonos día tras día servicios de atención al usuario. Sin embargo, aunque el progreso siempre es beneficioso, se deben considerar

muchos factores a la hora de implantar una nueva tecnología y es por ello que al igual que la robótica social se encuentra en plena investigación, así mismo se encuentran las leyes que regirán las acciones desarrolladas en un entorno robótico sofisticado. (Castro-Gonzáles, Martín, Castillo, & Salichis, 2017)

Figura 17. Amitiiel, robot de rescate.

Los obstáculos imaginables para el desarrollo de aplicaciones en la robótica social están ligados al ámbito legal más que al ingenieril o tecnológico, debido a que, dejar la decisión a un robot de herir a un ser humano es un tema de alto debate y análisis. (Abad Sacoto, Sánchez Delgado, Crespo Cedeño, & Alvarado Chang, 2017)

2.5 Robots humanoides

En este apartado se tratará acerca de las generalidades de la robótica humanoide, cuáles fueron sus inicios, sus características principales tanto físicas y se realizará un énfasis en los tipos de sensores y actuadores que poseen para su correcto funcionamiento. Posteriormente se realizará un enfoque especial en el humanoide NAO.

2.5.1 Generalidades de los robots humanoides

Durante años ha sido de gran interés para la comunidad científica la construcción y creación de prototipos que se asemejen a los humanos, tanto en comportamiento como en el aspecto físico, es por ello que en el año 1939 se presentó en Nueva York el primer robot humanoide llamado Elektro (Figura 18) creado por Westighouse Electric Corporation. (Senad, 2009)

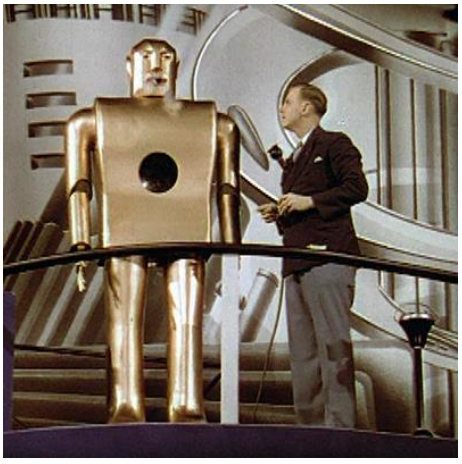

Figura 18. Elektro, Nueva York – 1939
Fuente: (Keller, 2016)

A partir de ese momento, los estudios tomaron lugar y el desarrollo de la robótica humanoide se disparó a través de los años, obteniendo como resultado una serie de ejemplares con capacidad de moverse y realizar tareas bien diseñadas.

Entonces, definimos como robot humanoide a un *tipo de robot que se asemeja en forma a un ser humano y tiene la capacidad de realizar movimientos similares.*

La mayoría de robots humanoides posee un torso con una cabeza, dos brazos y dos piernas, sin embargo existen ciertos ejemplares que poseen variaciones, pueden ser

únicamente partes del cuerpo humano (Figura 20) e incluso también pueden tener cara con ojos y boca, un claro ejemplo de ello es el humanoide Sofía (Figura 19).

Figura 19. Humanoide Sofía
Fuente:(Grundy, 2018)

La ventaja principal, y a su vez la importancia de la utilización de robots humanoides reside en que puede trabajar directamente en el mismo entorno y con los utensilios y herramientas fabricados para los humanos. En la Figura 20 se puede observar a Halodi Eve, un robot humanoide utilizado como auxiliar de cocina.

Figura 20. Robot ayudante de cocina.
Fuente: (robot-man, 2018)

2.5.2 Sensores y actuadores

Los dos componentes más importantes de los robots humanoides son los sensores y los actuadores.

A través del uso de sensores y actuadores se ha demostrado que es posible realizar acciones complejas como lanzar y atrapar cualquier tipo de objetos y manipular distintos tipos de herramientas. (Laschi, Dario, & Chiara)

Los sensores son dispositivos capaces de detectar magnitudes físicas o químicas, como pueden ser: parámetros ambientales (calor, sonido, luz, temperatura, entre otros) o parámetros físicos o fisiológicos (como movimiento u orientación), mientras que los actuadores son dispositivos mecánicos cuya función es transformar energía para generar un efecto.

En el presente documento clasificaremos los sensores en dos grupos: propioceptivos y exteroceptivos.

Tabla 1

Tipos de sensores y ejemplos

Sensores propioceptivos	Permiten obtener información acerca del estado interno del robot	Acelerómetros, sensores de temperatura, giroscopios.
Sensores exteroceptivos	Permiten obtener información acerca del entorno en que se encuentra el robot, para poder detectar localización de objetos, parámetros ambientales, entre otros.	Cámaras de video, micrófonos, sensores ultrasónicos.

En la investigación de la robótica humanoide se han estudiado los diferentes sentidos humanos, pero el enfoque ha sido guiado principalmente hacia las características visuales, auditivas y táctiles en comparación con las olfativas y gustativas (Tabla 2).

Tabla 2

Aplicación de los sensores en la robótica humanoide

Sentido	Descripción	Principales inconvenientes
Visión	Representan el mayor desafío para el campo de la visión por computadora debido a su naturaleza sin restricciones, además es la más costosa computacionalmente hablando.	Es necesario ampliar el campo de visión de las cámaras actuales puesto que los límites son mucho menores a los del ser humano, causando la pérdida información de aquellas áreas no cubiertas. Además, la adaptación a los cambios en las condiciones de luz aún es un proceso complejo en medios dinámic os.
Sensibilidad táctil	Es utilizado principalmente para tareas de manipulación y locomoción.	Detectan información puntual, es por ello que se utilizan en pocos puntos específicos de las plataformas humanoides.
Sonido	Son utilizados para adquirir información de sonido del entorno en una plataforma robótica. Se pueden instalar en diferentes configuraciones para detectar y regular el sonido.	Es muy común que se recepten los sonidos internos del robot, el ruido interno, la fusión de sensores y demás componentes no deseadas, lo que crea la necesidad de suprimir toda la información no necesaria para la aplicación.

Los actuadores se utilizan para funcionar como articulaciones y músculos. En su mayoría, los robots humanoides utilizan actuadores rotadores para lograr el efecto como movimiento humano, principalmente en las actividades de manipulación y locomoción, en la Tabla 3 se puede observar a detalle de que trata cada uno de ellos. Los actuadores pueden ser neumáticos, hidráulicos o eléctricos.

Tabla 3

Aplicación de los actuadores en los robots humanoides

	Descripción	**Principales inconvenientes**
Manipulación	Es una capacidad esencial en la robótica humanoide. En la actualidad las manos de los robots humanoides tienen tres, cuatro y cinco dedos.	El futuro manipulador humanoide debe ser capaz de interactuar como mínimo con los mismos objetos y mostrar propiedades activas y pasivas similares a las de las manos humanas.
Medio de locomoción	Una característica clave que se ha ido desarrollando y perfeccionando en los robots humanoides es la capacidad de caminar como un humano, mejor conocida como locomoción bípeda.	La locomoción bípeda y estable no está resuelta en su totalidad en las aplicaciones de robótica humanoide y se ha demostrado que la complejidad de este problema crece exponencialmente con la altura y peso del humanoide

2.6 Robot humanoide NAO

El robot humanoide NAO es un robot de locomoción bípeda, creado por la empresa SoftBank Robotics, su año de lanzamiento fue en 2006 y actualmente ha pasado por 6

versiones hasta llegar a NAOv6. En la Figura 14 se puede observar dos ejemplares de la versión 5.

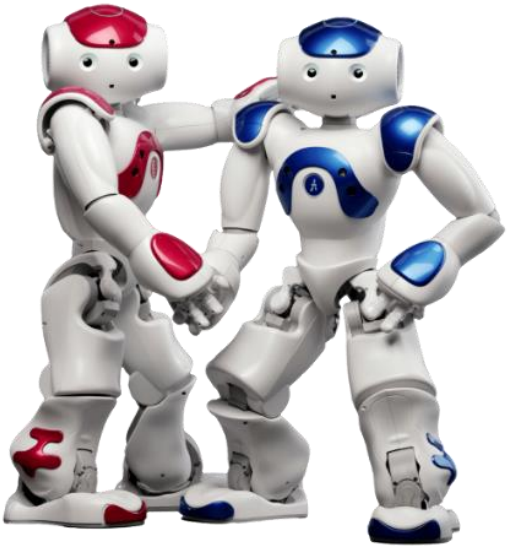

Figura 21. Robot Humanoide NAO

2.6.1 Características técnicas del robot NAO

NAO mide 58 cm y pesa aproximadamente 5,4 kg (el detalle de las dimensiones del robot se muestra en la Figura 22) tiene 25 grados de libertad que le permiten realizar movimientos complejos y fue creado principalmente para educación, investigación y aplicaciones en la medicina. Posee dos procesadores Intel ATOM 1’6 GHz localizados en la cabeza y en el torso.

El robot NAO posee varios sensores y actuadores, los cuales se enlistan en la Tabla 4 . Además, en la Figura 23 se muestra de forma gráfica las partes del humanoide.

Tabla 4

Partes del humanoide NAO

Cantidad	Elemento
2	Cámaras KVGA
4	Micrófonos
2	Altavoces
1	Sintetizador de voz
53	Leds RGB
9	Sensores táctiles
8	Sensores de presión
2	Sensores infrarrojos
1	Acelerómetro
1	Giroscopio
2	Sensores Ultrasónicos

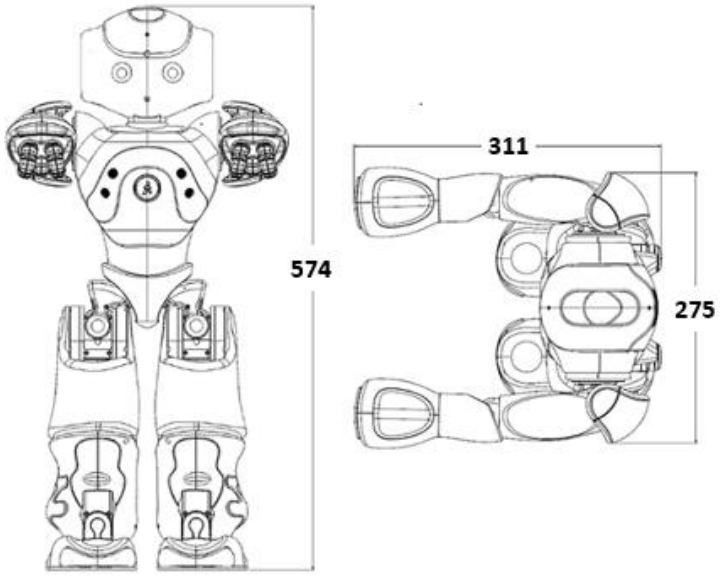

Figura 22. Dimensiones del robot NAO v5.
Fuente: (NAO Construction, s.f.)

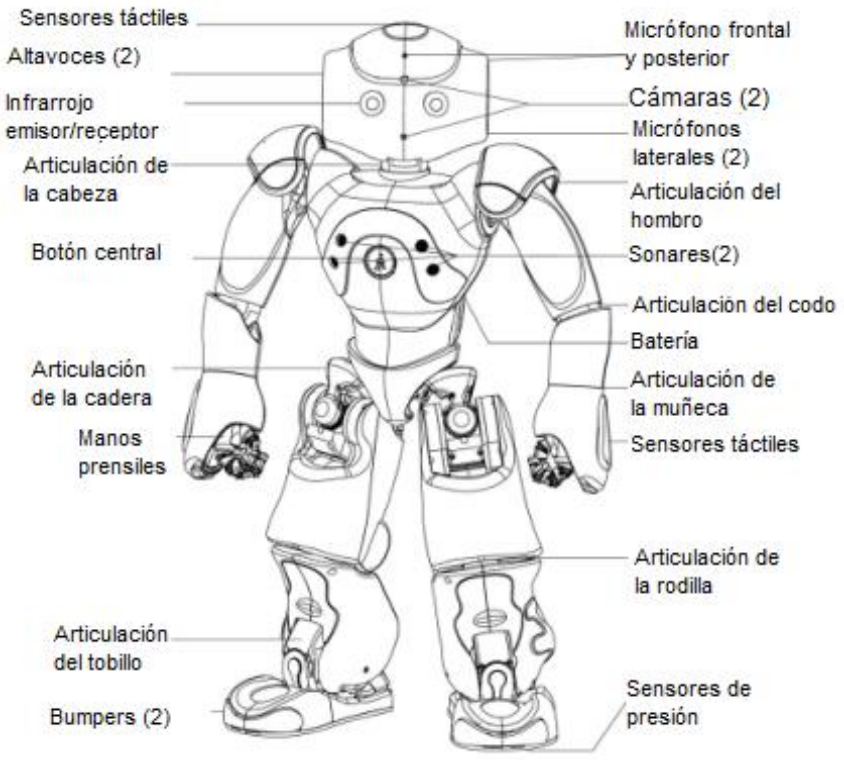

Figura 23. Partes del robot humanoide NAO.
Fuente: (Coltin)

Los rangos de visión cubiertos por las cámaras del robot se muestran en la Figura 24.

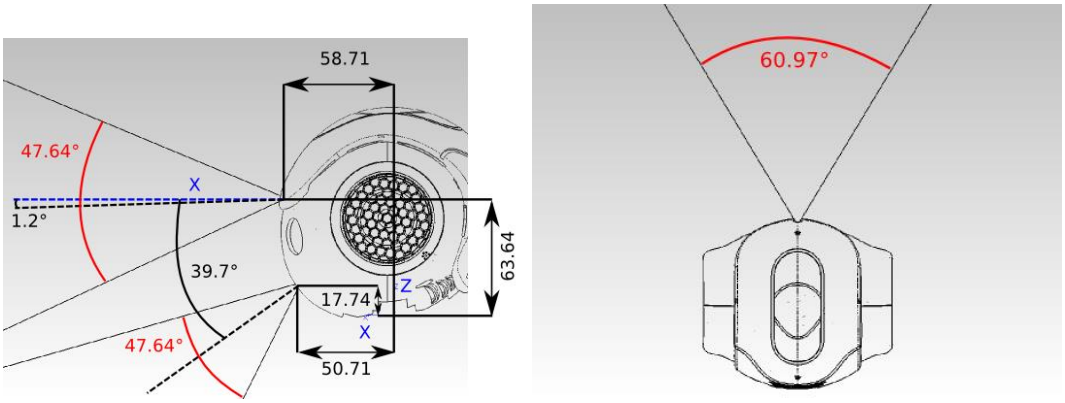

Figura 24. Rango de visión de las cámaras del robot NAO.

En términos de conectividad, el androide puede comunicarse mediante WiFi o Ethernet, con cifrado en WAP o WPE y otra de sus principales características es su capacidad de reconocer 19 idiomas distintos.

2.6.2 NAOqi Framework

Naoqi es el sistema operativo principal que permite el funcionamiento del humanoide NAO, mientras que NAOqi framework es el conjunto de funciones y herramientas que permiten acceder a los módulos del robot para controlar sus actividades.

NAOqi framework permite desarrollar aplicaciones desde distintos sistemas operativos y mediante dos distintos lenguajes de programación: C++ y Python.

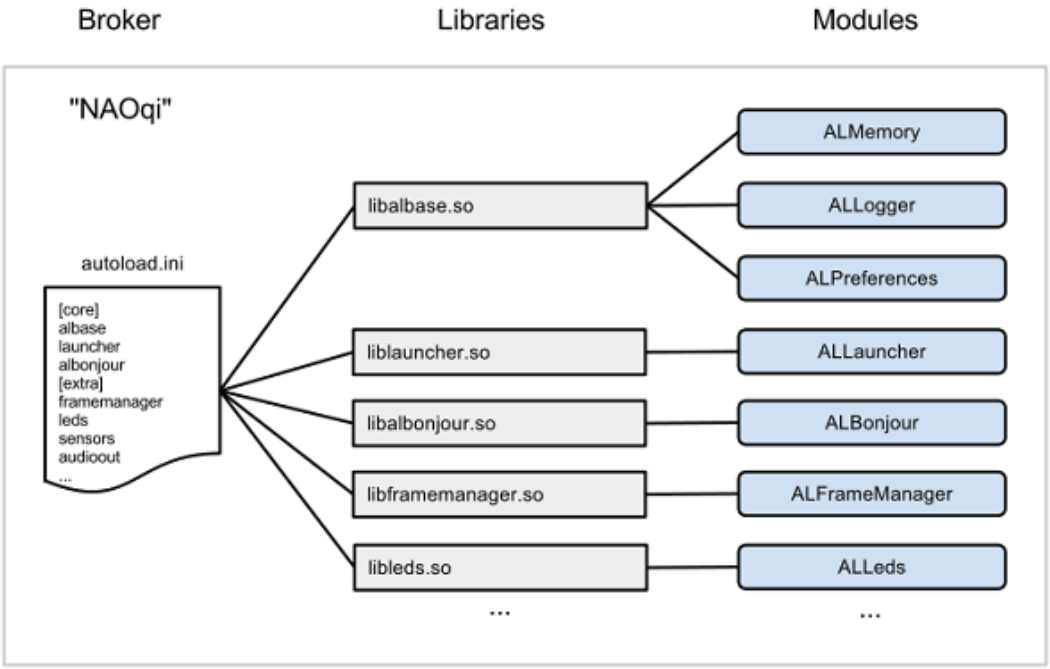

Figura 25. Estructura de NAOqi.
Fuente: (Aldebaran , s.f.)

El archivo ejecutable de NAOqi, es el bróker principal, permite definir las librerías que deben prepararse para la ejecución de aplicaciones, cada una de las librerías poseen módulos que utilizan al bróker principal como un intermediario para anunciar sus métodos, en la Figura 18 se puede observar de forma gráfica la estructura de NAOqi.

Además, el bróker provee servicios de búsqueda que permiten que cualquier módulo del árbol pueda encontrar cualquier método que requiera. Los módulos cargados forman un árbol de métodos adjuntos a módulos y a su vez módulos adjuntos a un bróker principal, como muestra la Figura 26.

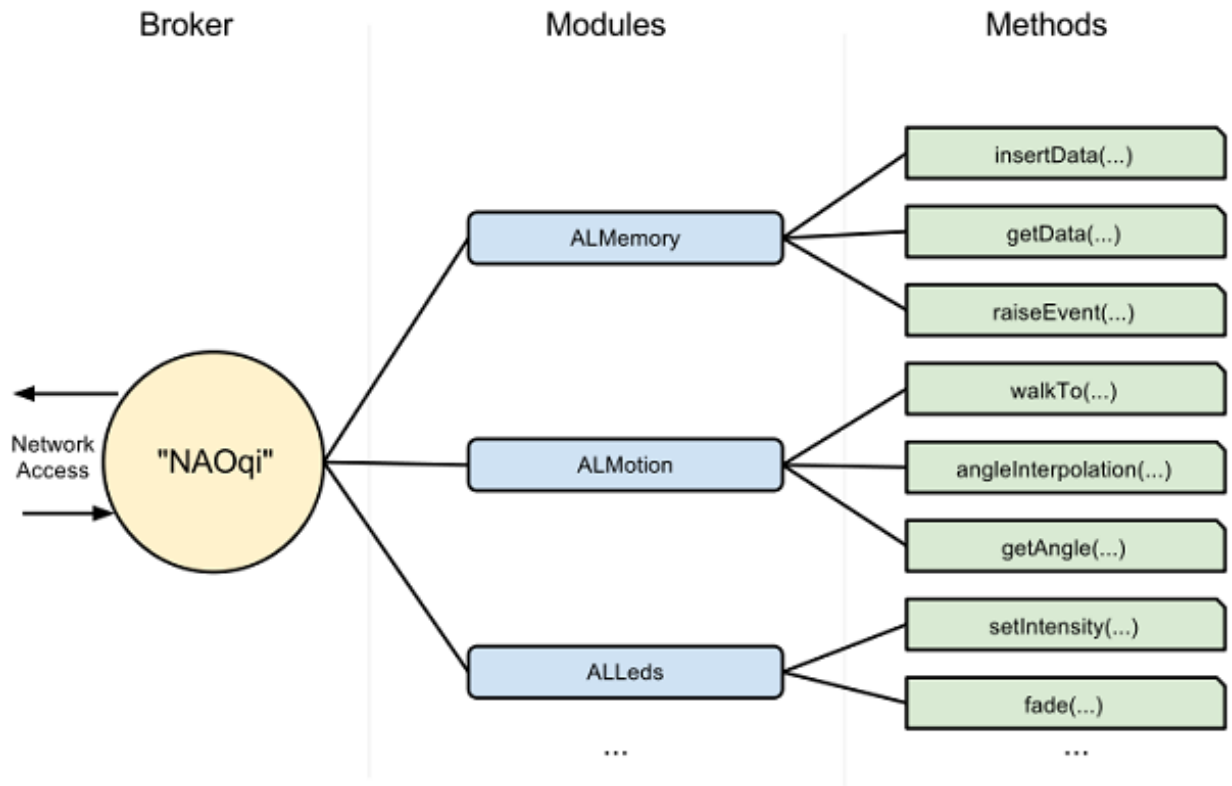

Figura 26. Estructura principal NAOqi.
Fuente: (Aldebaran , s.f.)

Los módulos y funciones utilizadas en el presente proyecto serán especificados en el capítulo 3.

2.6.3 Choregraphe

Choregraphe es una plataforma que permite desarrollar aplicaciones creada específicamente para el humanoide NAO, permite también simular los comportamientos en un prototipo simulado así como monitorear y controlar al robot. En la Figura 20 se puede observar la pantalla de arranque del software.

Figura 27. Pantalla de inicio de Choregraphe

El objetivo principal de esta plataforma, es el permitir a sus usuarios crear aplicaciones complejas sin necesidad de dominar ningún tipo de lenguaje de programación, esto se debe a que posee bloques gráficos que facilitan el uso del robot. Cada uno de estos bloques está compuesto típicamente por líneas de código en lenguaje Python que son las encargadas de realizar el comportamiento del bloque. Además Choregraphe permite modificar el comportamiento interno de los bloques existentes en

el software, e incluso es posible crear nuevos bloques personalizados de acuerdo con los requerimientos del usuario.

Existen 4 tipos de bloques gráficos existentes en el software: *Diagram,* el cual contiene más bloques internamente; *Timeline,* este tipo de bloque es utilizado para realizar movimientos con el robot; *Python,* este bloque posee código Python que puede ser modificado a gusto del usuario y finalmente *Dialog* que permite crear un diálogo entre el robot y un agente externo. En la Figura 28 se puede observar los íconos que representan a cada tipo de bloque en Choregraphe.

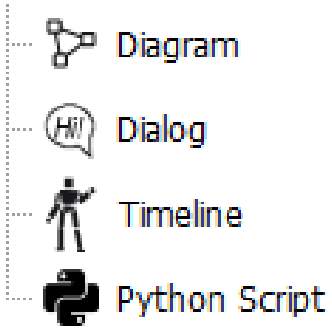

Figura 28. Tipos de bloques gráficos en Choregraphe

2.6.4 ROS

ROS es un sistema meta-operativo de código abierto para robots, realiza abstracción de hardware, el control de los dispositivos de bajo nivel el paso de mensajes e información entre procesos y la administración de paquetes. Ofrece también herramientas y bibliotecas para obtener, crear, escribir y ejecutar código en diversos tipos de plataformas robóticas. (Watkins, 2018).

Figura 29. Logo de ROS

ROS fue creado para fomentar la colaboración en el desarrollo de software de robótica, por ejemplo si un grupo de investigadores logran solucionar un sistema de reconocimiento facial podrían colaborar con otra agrupación de estudiantes al otro lado del mundo que desean implementar una aplicación que utiliza técnicas de acceso biométrico.

ROS está diseñado de forma que todos los procesos se denominan nodos, y cada uno de estos nodos es responsable por realizar una tarea. Los nodos se comunican entre si utilizando mensajes que se transmiten por canales lógicos llamados tópicos. Cada uno de los nodos puede enviar o recibir datos de otro nodo utilizando un modelo de publicar/suscribirse. (Tawil, 2017)

2.6.5 qiBuild

qiBuild es un entorno de trabajo creado para los sistemas que funcionan en base a NAOqi, principalmente se encarga de gestionar las dependencias entre proyectos y soporta la compilación cruzada.

Posee diversas herramientas clave para el desarrollo de aplicaciones, tales como la administración de proyectos contenidos en distintos repositorios, creación de

documentación, permite también traducir proyectos en varios idiomas, tiene la capacidad de crear paquetes binarios, así como adjuntar código en Python o C++ dentro de ellos y finalmente permite utilizar extensiones de Python en programas creados en C/C++ con librerías puras de Python.

En comparación con ROS, qiBuild proporciona una forma limpia de empaquetar y usar dependencias de terceros sin tocar el sistema, que funcionará en cualquier distribución de Linux.

2.7 Estado del arte

En el presente apartado se muestra brevemente el estado del arte de sistemas de clasificación de objetos que utilicen o estén basados en cualquier tipo de robot humanoide.

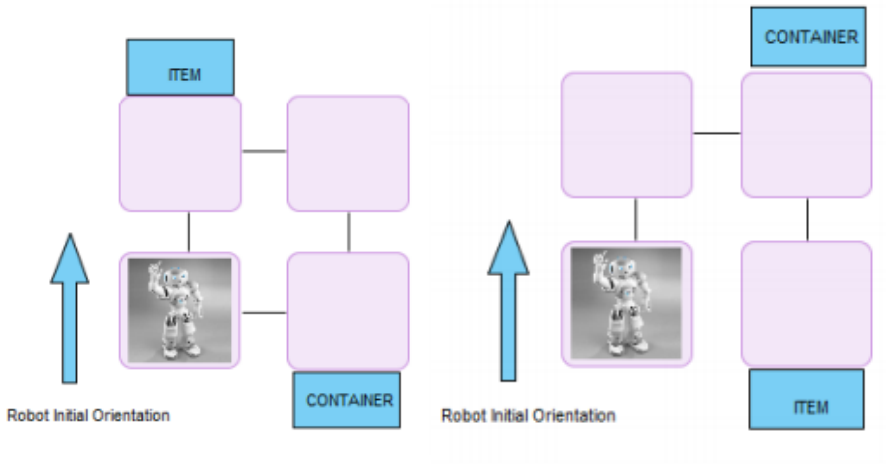

Figura 30. Sistema de control para recolección de objetos con NAO

El autor Zimmermann Mracos en el año 2014 desarrollo un sistema de control para un robot humanoide NAO mediante la utilización de planificación automática y visión artificial, cuyo objetivo era dotar al robot de la capacidad de recoger un objeto declarado al inicio de la aplicación y llevarlo hacia una posición conocida previamente, este sistema tiene la capacidad de almacenar los objetos y la trayectoria realizada para poder ser analizada posteriormente por los humanos que lo operen, para realizarlo se utilizó un ordenador como encargado de la ejecución del sistema de control y la arquitectura de planificación ejecución y aprendizaje. En la Figura 30 se pueden observar los dos esquemas experimentados.

De este estudio se determinó que el robot NAO presenta complicaciones en aspectos de odometría, puesto que en cada giro realizado se presentaban errores que no permitían que siga por las rutas predeterminadas, obligando al autor a realizar correcciones accediendo a los distintos sensores propioceptivos del robot.

Figura 31. Ginoide AILA

Durante la feria de Industria y maquinaria Hanover Messe del año 2010, el centro de investigación alemán para la inteligencia artificial presentó a la ginoide AILA (Figura 31), que tenía la capacidad de detectar objetos y manejarlos de forma individual para clasificarlos de acuerdo a las características cada elemento obtenidas mediante lectura del código RFID de una base de datos de productos comunes. AILA posee una base móvil a base de ruedas para su desplazamiento, brazos antropomórficos, torso con 4 grados de libertad y cabeza rotatoria. (BOXBYTE, 2010)

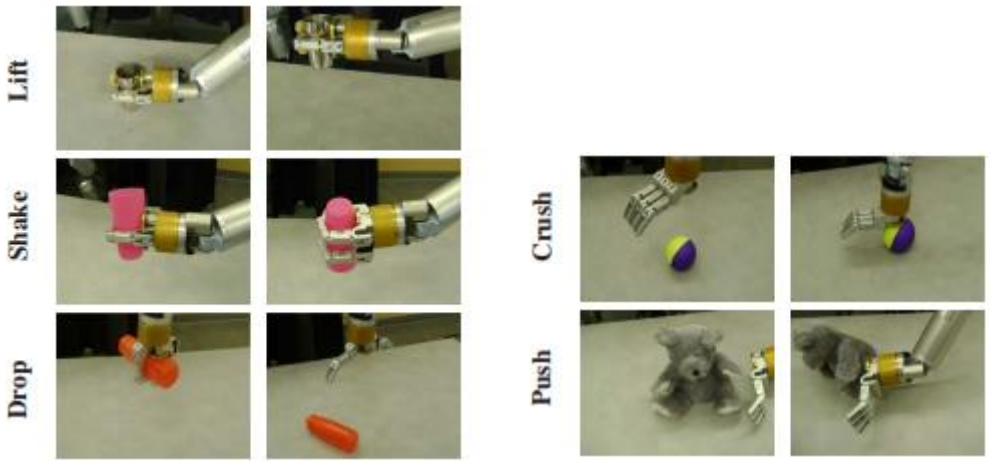

Figura 32. Comportamiento exploratorio para detección de objetos.
Fuente: (Sinapov, 2014)

Por otro lado, Sibapov plantea un sistema de detección de objetos encontrados típicamente en un hogar, mediante un torso robótico en su estudio denominado "Reconocimiento de la categoría de objetos por un robot humanoide utilizando un aprendizaje relacional basado en el comportamiento". El método propuesto consiste en que el robot manipula cada elemento mediante comportamientos exploratorios (levantar, agitar, soltar, aplastar y empujar) para poder tomar una decisión en base a los sonidos

generados en cada actividad. El objetivo de esta investigación era comprobar y remarcar que, además de la percepción visual, se pueden detectar distintos tipos de elementos utilizando información de tipo auditivo.

En el año 2017, se implementó un sistema de clasificación de materiales utilizando la plataforma humanoide NAO, en el cual mediante el manejo de señales acústicas se realizó un estudio que permitía detectar y diferenciar objetos de acuerdo con las características intrínsecas del material del cual están compuestos, este sistema tiene la capacidad de diferenciar entre aluminio, cartón o plástico de acuerdo a la frecuencia de la onda resultante de golpear dicho elemento repetidas veces. Una vez determinado el tipo de objeto, el robot procede a tomarlo y trasladarlo hacia una posición previamente definida (Figura 33).

Figura 33. Clasificación de objetos por señales acústicas con NAO
Fuente: (Lopez-Caudana, Quiroz, Rodríguez, Yépez, & Ibarra, 2017)

Como conclusiones se obtuvo que las señales acústicas contienen información importante y directamente relacionada con las propiedades físicas de cada objeto,

además el procesamiento de señales de audio requiere de menos recursos, tiempo y almacenamiento en comparación con imagen y video. Además se especifica que este sistema de clasificación no está creado para diferenciar materiales cuyos rangos de frecuencia sean similares. (Lopez-Caudana, Quiroz, Rodríguez, Yépez, & Ibarra, 2017)

CAPÍTULO III

DISEÑO E IMPLEMENTACIÓN DEL SISTEMA DE CLASIFICACIÓN

3.1 Descripción del sistema de clasificación de objetos

La aplicación creada es capaz de clasificar dos colores según la selección de usuario simplemente introduciendo lo códigos RGB correspondientes, en el presente trabajo se utilizaron pelotas de color rojo, azul y verde de 4 cm de diámetro, cuyos valores RGB se detallan en la sección 3.2.

El área de clasificación es de $2m^2$ dentro de cual se encuentran ubicados el humanoide, los depósitos y los objetos, tal como se muestra en la Figura 34.

Figura 34. Área utilizada para el sistema de clasificación de objetos

Los depósitos de objetos se encuentran ubicados en la parte posterior del área de trabajo, y su indicador correspondiente deberá estar colocado a 50 cm de altura desde el piso (Figura 35). El humanoide inicia la clasificación en la posición mostrada en la Figura 34 y los objetos a clasificar se encuentran localizados sobre pedestales de madera dispuestos de forma aleatoria dentro del área de clasificación. Se dispuso utilizar los pedestales antes mencionados para evitar el riesgo de que el humanoide pierda el equilibrio al momento de descender hacia el piso para tomar el objeto y caiga, la altura de los pedestales varía entre 42 y 42,5 centímetros facilitando el proceso de manipulación. Los objetos a clasificar pueden ser de distintos colores, sin embargo, la aplicación reconocerá y por ende clasificará únicamente aquellos cuyos colores hayan sido definidos en el programa.

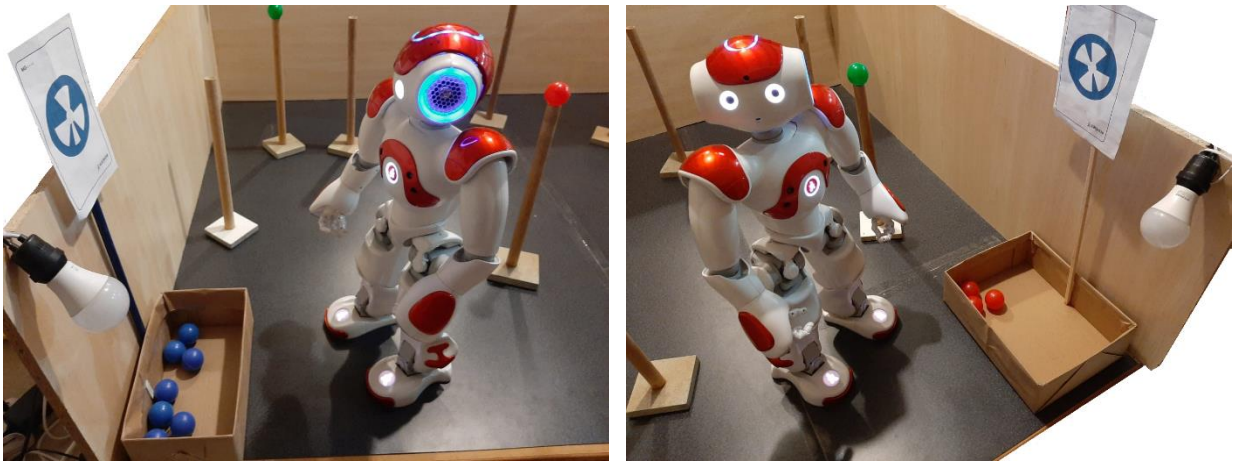

Figura 35. Ubicación de los depósitos de objetos

Es importante mencionar que el presente proyecto no tiene como objetivo la evasión de obstáculos, por lo cual deben ubicarse los pedestales de tal forma que no

interfieran físicamente entre sí, o deben ser retirados del área de trabajo cuando el robot ya haya tomado el objeto sobre ellos. Debido a que el rango de visión horizontal del robot es de 60.97° (Figura 24), la zona de clasificación se divide en 3 secciones para cubrir con toda el área de trabajo.

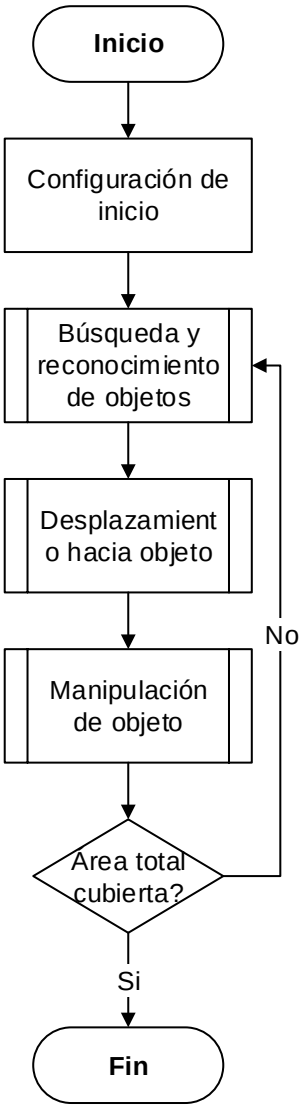

Figura 36. Diagrama de flujo del sistema de clasificación de objetos.

Como se puede observar en la Figura 36 el sistema de clasificación de objetos está conformado de 3 subprocesos fundamentales que son: Búsqueda y Reconocimiento de objetos, Desplazamiento hacia objeto, Manipulación del objeto; las cuales serán detalladas en las secciones 3.2, 3.3 y 3.4 respectivamente.

La implementación del programa principal en el software se puede observar en la Figura 37, en ella se pueden observar 3 diagramas que son: Inicio_App, Clasificacion_Objetos, Finalizar_App.

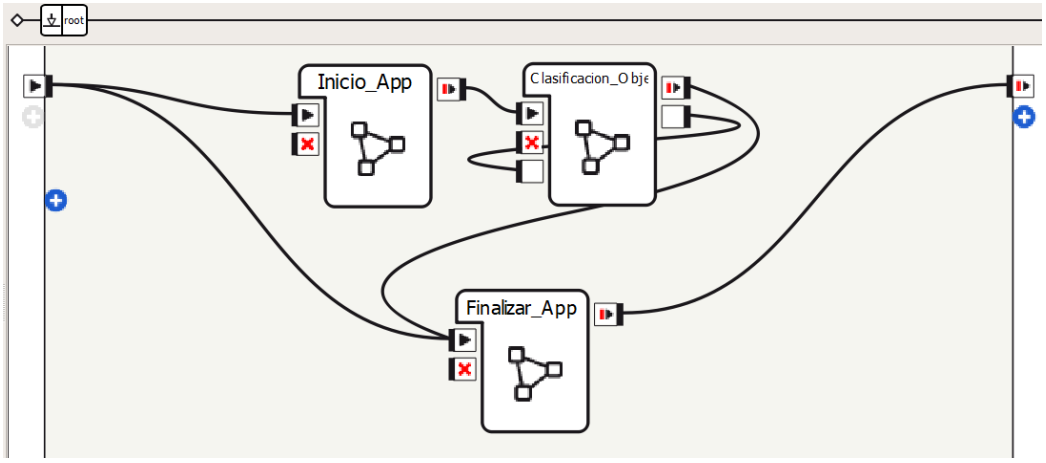

Figura 37. Contenido del programa principal.

- ***Diagrama Inicio_App***

Este diagrama contiene internamente bloques de programación encargados de la configuración inicial y preparación del robot para ejecutar las actividades planteadas para la clasificación de objetos, en la Figura 38 se puede observar el contenido del diagrama.

- **Bloque de código Iniciar**

Recibe una señal proveniente del sensor táctil frontal ubicado en la cabeza del robot, (de acuerdo con la Figura 39, corresponde al sensor A). En caso de ser presionado el sensor, dicha señal toma el valor de 1.0, lo que permite iniciar la aplicación.

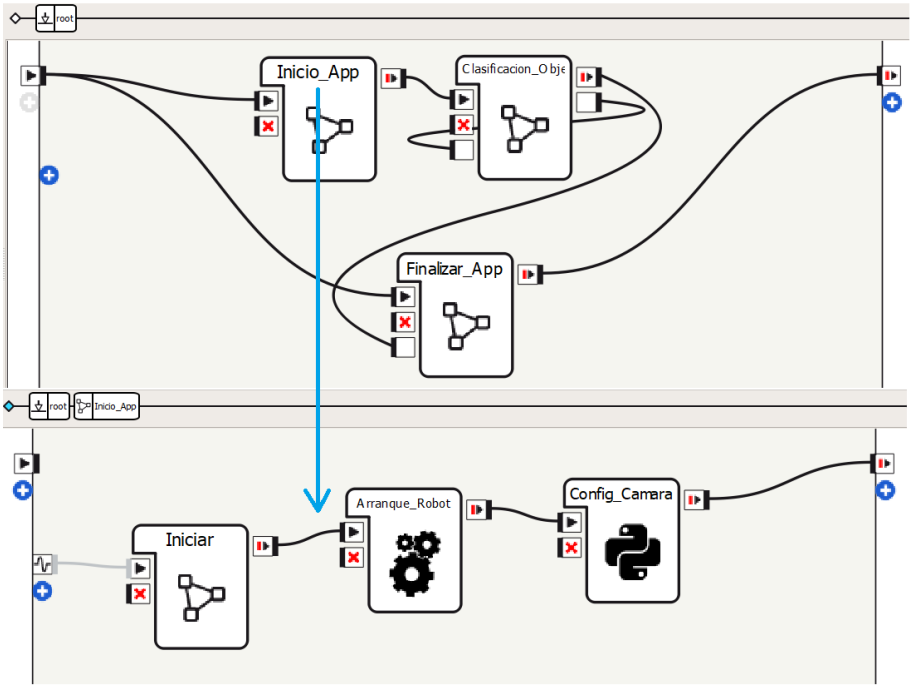

Figura 38. Contenido del diagrama Inicio_App

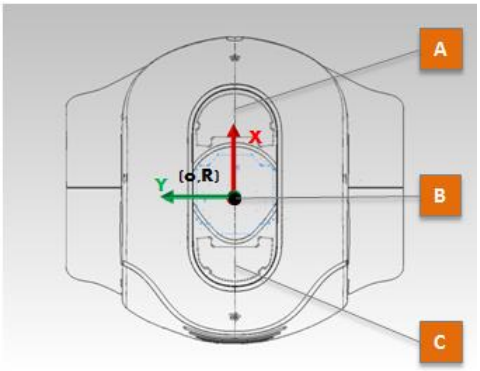

Figura 39. Sensores táctiles de la cabeza del humanoide NAO

- **Bloque de código Arranque_Robot**

Este bloque se encarga de implementar la configuración inicial requerida para ejecutar la aplicación de clasificación de objetos, su diagrama de flujo se muestra en la Figura 40. El código completo se encuentra en el Anexo A1 del presente documento.

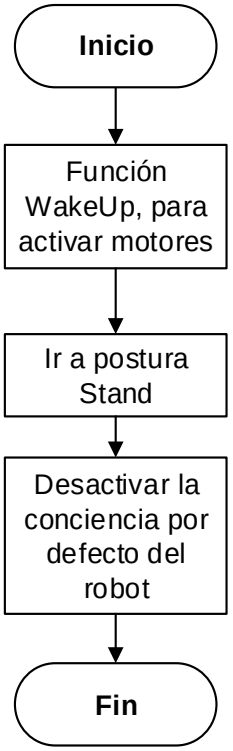

Figura 40. Diagrama de flujo del bloque Arranque_Robot

En este proceso se utilizaron 3 módulos: *ALMotion*, utilizado para gestionar los movimientos del robot en esta ocasión fue utilizado para cambiar al humanoide de modo descanso a modo activado; *ALRobotPosture*, que posee diversas funciones para trabajar con las distintas poses por defecto propias del robot, permitió llevar al robot a la posición predefinida Stand y *ALBasicAwareness*, encargada de administrar las actividades de conciencia implementadas por defecto en el humanoide, las cuales permiten hacer el

seguimiento de rostros, de sonidos y de objetos en movimiento, sin embargo en este caso se requirió desactivar dichas actividades. Para definir y llamar cada módulo se utilizó el siguiente código:

```
self.motion = ALProxy("ALMotion")//Módulo para movimiento
self.posture = ALProxy("ALRobotPosture")//Módulo para poses
self.awareness = ALProxy('ALBasicAwareness')//Módulo de
conciencia
```

Las funciones utilizadas se muestran a continuación.

```
self.motion.wakeUp()//Activación de los motores
self.posture("Stand",0.7)//LLevar al robot a postura Stand a una
velocidad de 0.7
self.awareness.stopAwareness()//Desactivar la conciencia
```

- **Bloque de código Config_Camara**

Se desarrolló este bloque programado en Python para establecer los parámetros de las cámaras superior e inferior, el código completo se puede observar en el anexo A2 del presente documento y el diagrama de flujo correspondiente se muestra en la Figura 41.

Para realizar la configuración de los parámetros se utilizó el módulo *ALVideoDevice*, el cuál debe ser inicializado utilizando la sintaxis mostrada a continuación:

```
self.camera=ALProxy("ALVideoDevice")//Módulo de cámara
```

Para modificar los parámetros deseados de la cámara, se debe utilizar la siguiente sintaxis, en donde ID corresponde al código del parámetro y valor, al nuevo dato que se desea ingresar.

```
self.camera.setParam(ID, valor)//Modificación parámetros de la cámara
```

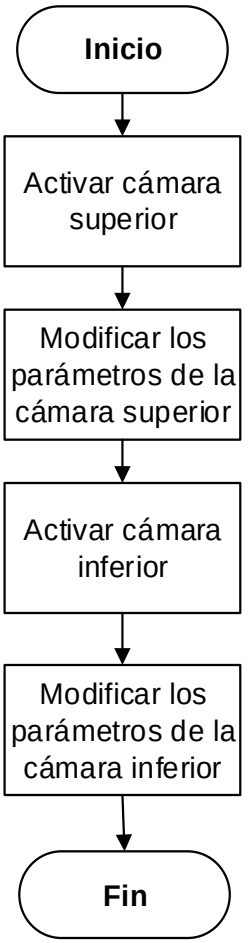

Figura 41. Diagrama de flujo del bloque Config_Camara

Los parámetros modificados y sus valores correspondientes se muestran en la Tabla 5.

Tabla 5

Parámetros de las cámaras del humanoide modificados para el desarrollo del sistema de clasificación de objetos.

Parámetro	ID	Valor	Rango permitido
kCameraAutoExpositionID	11	0	0 o 1
kCameraAutoWhiteBalanceID	12	0	0 o 1
kCameraExposureID	17	352	Entre 1 y 2500
kCameraGainID	0	76	Entre 32 y 255
kCameraBrightnessID	6	55	Entre 0 y 255
kCameraContrastID	1	32	Entre 16 y 64
kCameraSaturationID	2	128	Entre 0 y 255
kCameraWhiteBalanceID	33	-47	Entre -150 y -21

- ***Diagrama Clasificación_Objetos***

Este diagrama contiene los subprocesos contenidos en la aplicación, en la Figura 42 se puede observar el contenido del diagrama Clasificacion_Objetos.

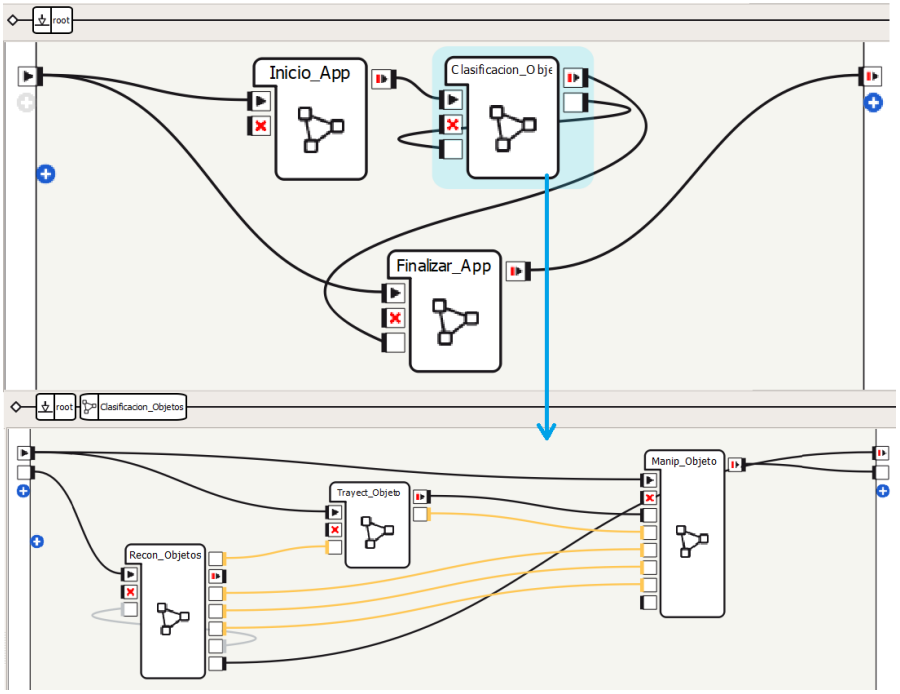

Figura 42. Contenido del diagrama Clasificacion_Objetos

- **Bloque de código Recon_Objetos**

Este bloque se encarga de realizar la búsqueda y reconocimiento de objetos, en la sección 3.2 se detalla su contenido.

- **Bloque de código Trayect_Objeto**

Este bloque se encarga de realizar el cálculo de la distancia y la generación de la trayectoria entre el objeto de interés y el robot, en la sección 3.3 se detalla su contenido.

- **Bloque de código Manip_Objeto**

Este bloque se encarga de realizar la manipulación del objeto, así como del depósito de este, en la sección 3.4 se detalla su contenido

- ***Diagrama Finalizar_App***

Este diagrama se encarga de llevar al robot hacia un estado de reposo una vez terminada la clasificación de los objetos. En la Figura 43 se puede observar el contenido interno del presente diagrama.

Existen dos formas de acceso a esta etapa del programa, la primera sucede cuando el sistema de clasificación de objetos ha finalizado su operación, y la segunda opción es mediante la pulsación del sensor táctil trasero localizado en la cabeza del robot (de acuerdo con la Figura 39, el sensor C).

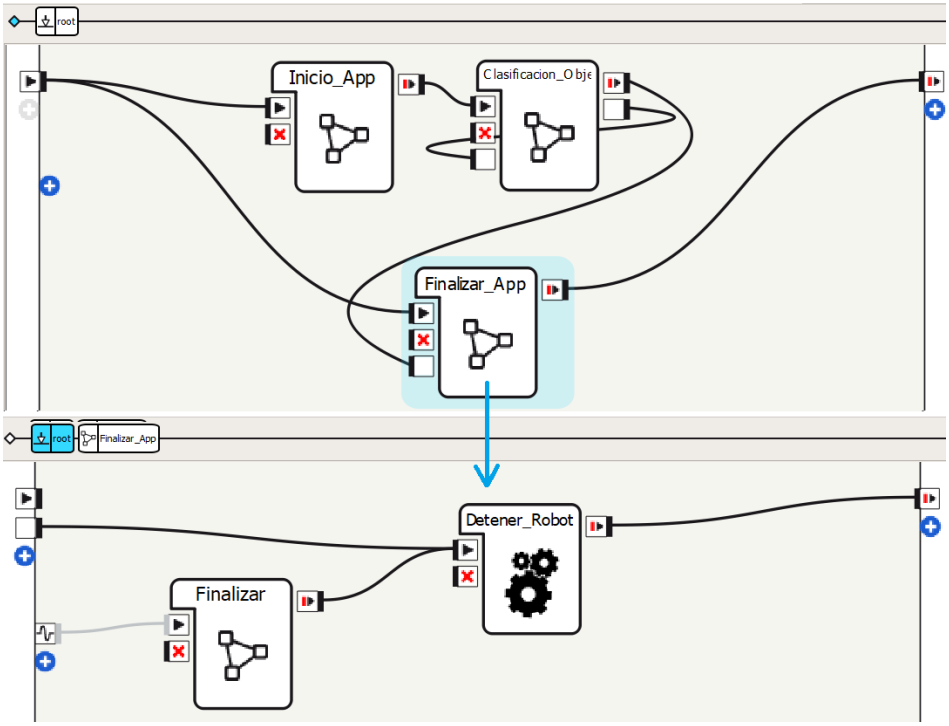

Figura 43. Contenido del diagrama Finalizar_App

- **Bloque de código Finalizar**

Es un proceso de lectura y verificación de que el sensor táctil trasero de la cabeza haya sido presionado.

- **Bloque de código Detener_Robot**

Se desarrolló este bloque programado en Python para enviar al robot a un estado de descanso una vez finalizada la actividad, el código completo se muestra en el anexo A14 del presente documento, el diagrama de flujo correspondiente se muestra en la Figura 44 .

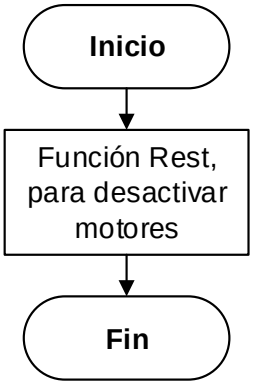

Figura 44. Diagrama de flujo del bloque Detener_Robot

Se utiliza el módulo *ALMotion*, y la función para este caso es la siguiente:

```
self.motion.rest() // Apagar motores, robot descansa
```

3.2 Desarrollo del proceso de búsqueda y reconocimiento de objetos

El sistema de reconocimiento de objetos realiza la búsqueda de elementos de interés que se encuentren cercanos al robot, analiza cual se encuentra más cerca y lo selecciona como objetivo para proceder a la clasificación, para ello extrae las características de su ubicación en 2D dentro del plano de la cámara y las envía hacia la etapa de *trayectoria hacia el objeto* para continuar con el proceso.

La implementación del sistema de reconocimiento de objetos en Choregraphe se muestra en la Figura 45, el cual corresponde al diagrama de flujo de la Figura 46.

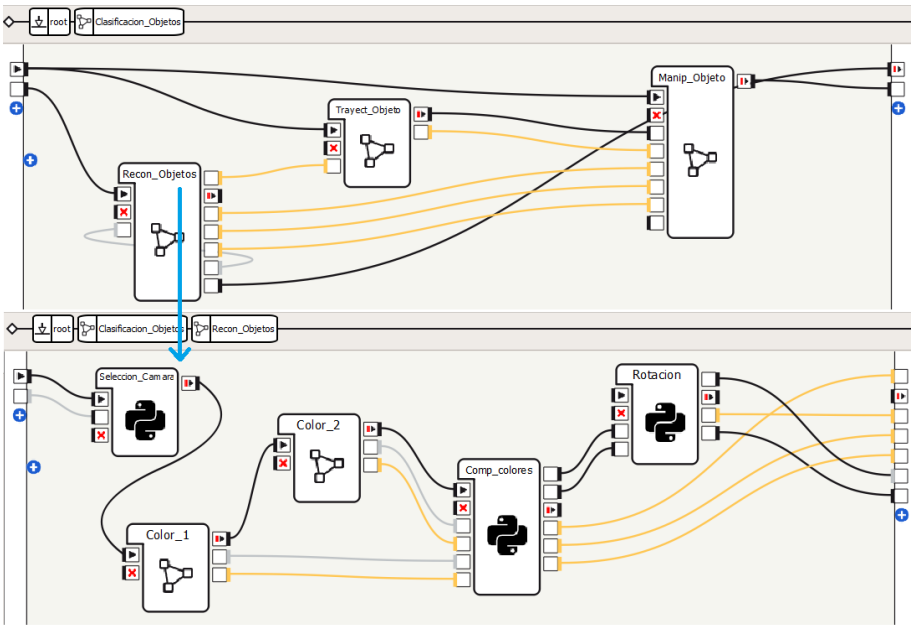

Figura 45. Proceso de búsqueda y reconocimiento de objetos.

El programa de reconocimiento está conformado por 5 bloques de código, los cuales se detallan en la siguiente sección.

Tabla 6

Códigos RGB utilizados

Color	Rojo	Verde	Azul
R	243	42	31
G	81	231	82
B	30	22	224
Threshold	53	45	61

Los elementos de interés se determinan ingresando dos colores distintos mediante código RGB, los cuales pueden ser ingresados por el usuario. En el presente trabajo se

realizó el análisis con tres tipos de pelotas de distintos colores: rojo, verde y azul, cuyos valores RGB se muestran en la Tabla 6.

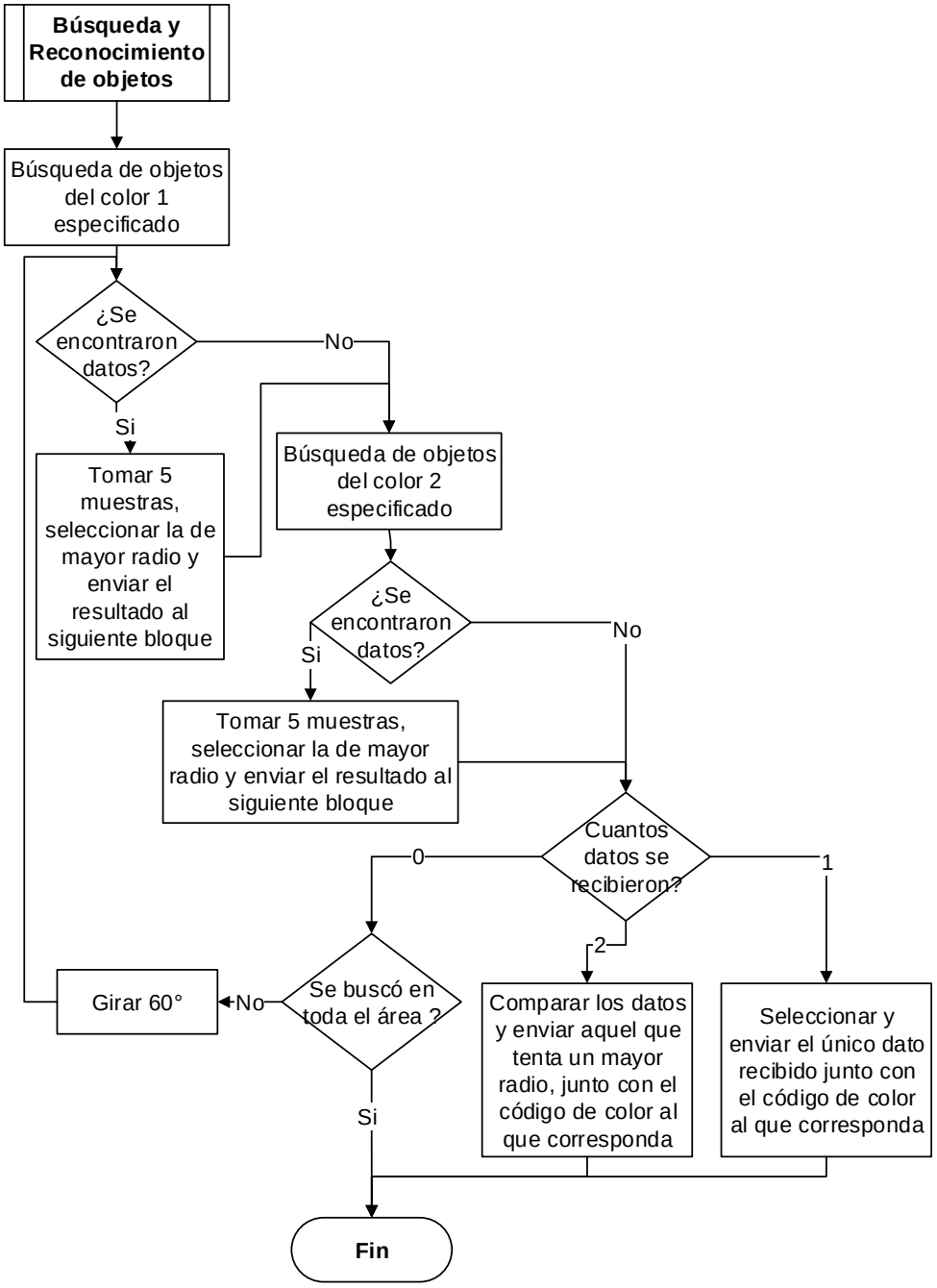

Figura 46. Diagrama de flujo del proceso de Búsqueda y Reconocimiento de objetos

- ***Bloque de código Seleccion_Camara***

Dado que el humanoide posee dos cámaras, la cámara inferior se utiliza para detectar objetos más cercanos al robot y la cámara superior para aquellos objetos localizados más lejos. El presente bloque se encarga de seleccionar y configurar la cámara que se necesite utilizar en cada etapa del programa, al iniciar con la búsqueda y reconocimiento de objetos se utiliza la cámara inferior y posteriormente se modificará esta configuración según sea necesario. El diagrama de flujo se muestra en la Figura 47.

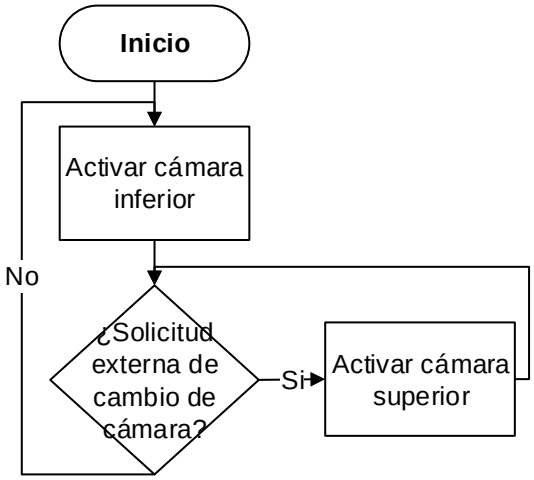

Figura 47. Diagrama de flujo del bloque Seleccion_Camara

Para la configuración de la cámara se utilizó el módulo *ALVideoDevice*, la sintaxis de la función utilizada para el cambio de cámara se muestra a continuación. El parámetro valor puede ser '0', que selecciona la cámara superior, o '1' que activa la cámara inferior.

```
self.video.setActiveCamera(valor)//Determina la cámara a utilizar
```

Este bloque de código fue creado para el desarrollo de la presente aplicación, el código completo se muestra en el Anexo A3 al final del documento.

- ***Diagramas Color_1 y Color_2***

Estos diagramas son idénticos y se encargan del reconocimiento de los objetos de acuerdo al valor RGB ingresado. Inicialmente se realiza un análisis de la presencia de elementos del color correspondiente en el área de búsqueda, para ello se toma la información proveniente del módulo *ALBlobDetection,* Se almacenan los 5 primeros resultados obtenidos en forma de vectores, posteriormente se realiza una comparación entre la información almacenada y se selecciona aquel vector cuyo valor de diámetro sea mayor para enviarlo hacia el siguiente bloque como resultado de la búsqueda de cada color. Es importante mencionar que mientras mayor sea el diámetro del objeto detectado, más cerca estará ubicado.

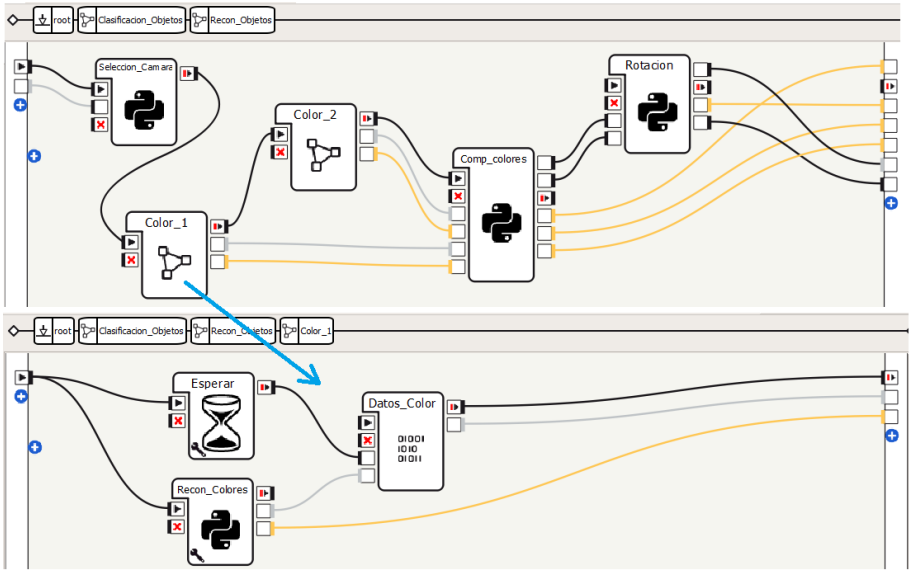

Figura 48. Contenido del diagrama Color_1

En caso de no encontrarse objetos dentro del área de búsqueda, cada diagrama posee un tiempo de espera de 5 segundos, al haber transcurrido este tiempo sin haberse generado ninguna lectura, el bloque envía una matriz de datos vacía indicando que no se han detectado elementos.

Como se puede observar en la Figura 45, los bloques Color_1 y Color_2 son de tipo diagrama, lo que implica que contienen más bloques internos. El contenido de Color_1 se muestra en la Figura 48.

- **Bloque de código Reco_Colores**

Este bloque se encarga de suscribirse al evento que permite el reconocimiento del color y a la extracción de información acerca del objeto detectado, el diagrama de flujo del presente bloque se muestra en la Figura 49. Se utilizó el módulo *ALColorBlobDetection*, el cual permite establecer los códigos de los colores, cuya llamada se realiza como se muestra a continuación.

```
self.colorFinder = ALProxy("ALColorBlobDetection")//Módulo para
la detección de colores
```

Para suscribir un bloque de código a un evento, se debe añadir el módulo *ALMemory* puesto que es el encargado de almacenar las llamadas generadas cuando sucede un evento al que el robot está suscrito, para ello se utilizó la siguiente línea de código.

```
self.memory = ALProxy("ALMemory")// Módulo para acceso a eventos
```

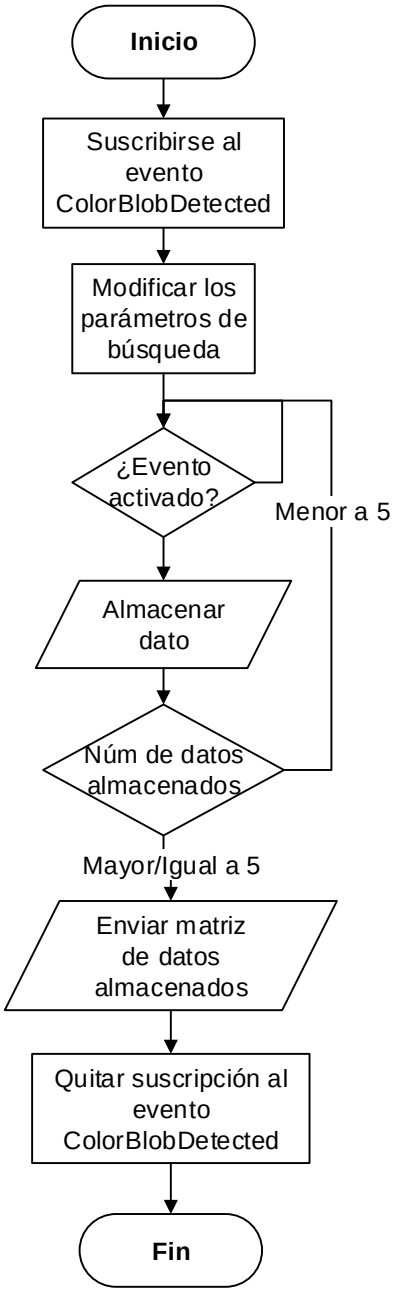

Figura 49. Diagrama de flujo del bloque Reco_Colores

Debido a la estructura de programación del lenguaje Python, cada vez que un evento suceda, este debe direccionarse hacia una nueva función dentro del programa, para ello se debe crear una nueva función interna, tal como se muestra a continuación.

```
self.BIND_PYTHON(self.getName(), "ColorDetected")// Creación de
una nueva función
```

Las funciones que permiten modificar los parámetros de búsqueda son los siguientes:

- setColor, requiere 4 valores que deben oscilar entre 0 a 255, que son: código R, código G y código B y Threshold;
- setOBjectProperties, requiere que se ingresen dos valores: el tamaño mínimo del objeto en pixeles y el diámetro del mismo en centímetros.

```
self.colorfinder.setColor(self.valorR,self.valorG,self.valorB,se
lf.valorT) // Características del color de interés
self.colorfinder.setObjectProperties(10, 0.4)// Características
del objeto de interés
```

Cuando el evento de detección se activa, el programa es redirigido a la función *ColorDetected* creada previamente, en la cual se utiliza la función getCircle, que permite obtener la posición en los planos *x* y *y* del objeto detectado, así como el radio de dicho objeto con respecto al plano 2d de la cámara en pixeles. Para llamar a la función se utiliza la siguiente línea de código, se almacenan 5 vectores de posición de objetos en una matriz para proceder a compararlos y seleccionar el que tenga un diámetro mayor.

```
value=self.colorfinder.getCircle() // Toma de datos del color
detectado
```

Este bloque de código fue creado para el desarrollo de la presente aplicación, el código completo se muestra en el Anexo A4 al final del documento.

- **Bloque de código Datos_Color**

Se encarga de filtrar y seleccionar el vector de datos del objeto detectado que tenga el diámetro mayor para enviarlo a los siguientes módulos. El diagrama de flujo correspondiente se muestra en la Figura 50.

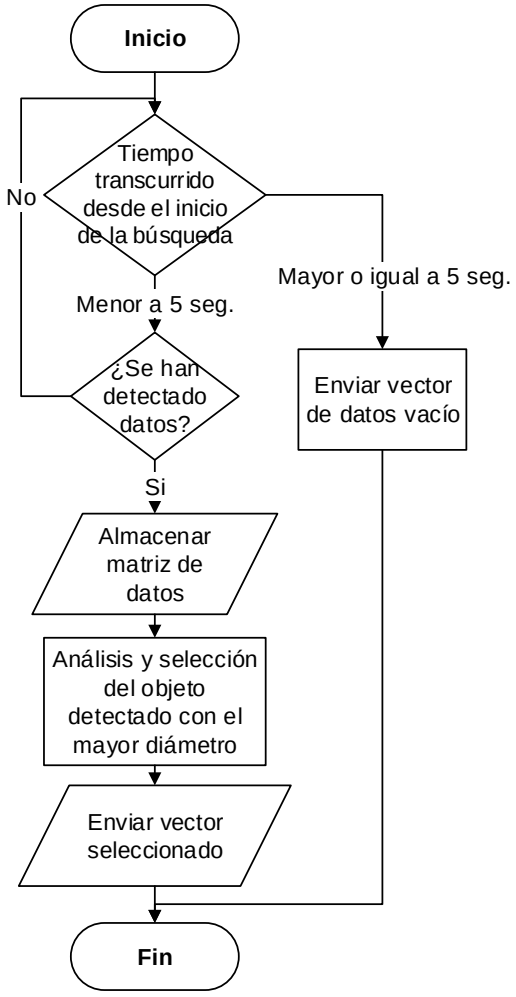

Figura 50. Diagrama de flujo del bloque Datos_Color

Este bloque de código fue creado para el desarrollo de la presente aplicación, el código completo se muestra en el Anexo A5 al final del documento.

- ***Bloque de código Comparacion_Colores***

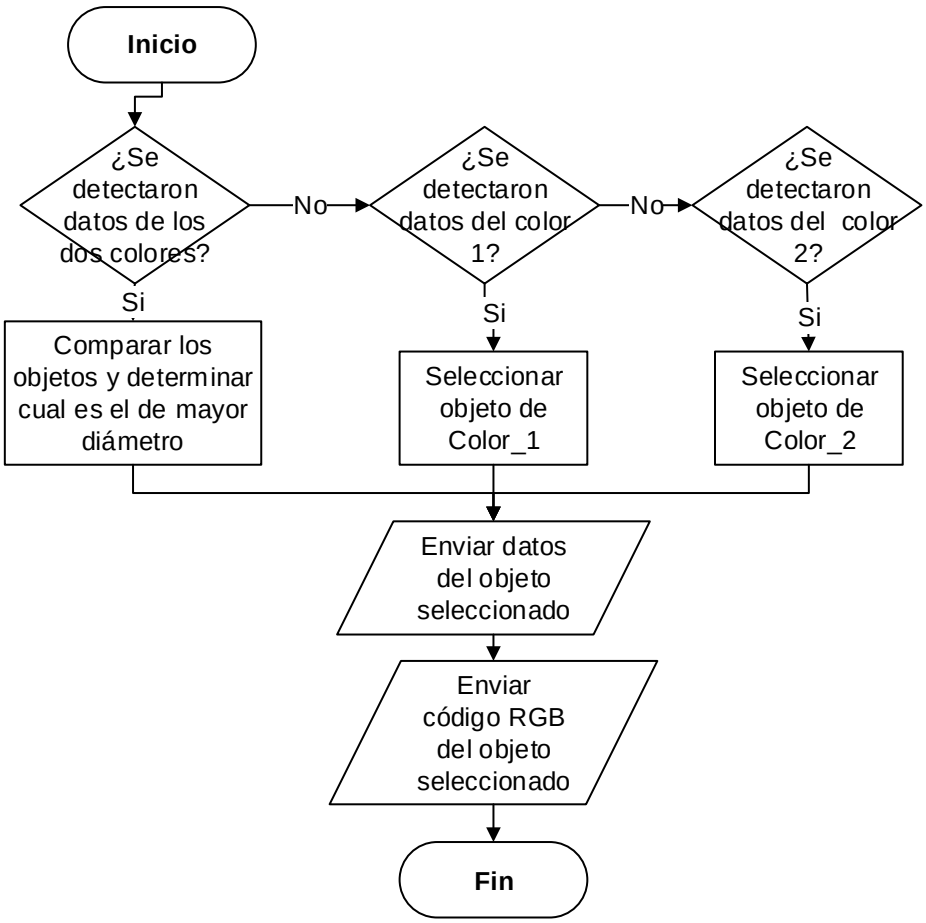

Figura 51. Diagrama de flujo del bloque Comp_Colores

Comparacion_Colores recibe lo resultados de la búsqueda de cada color obtenidas en los bloques anteriores y se encarga de comparar cuál de ellos tiene un diámetro mayor, para seleccionarlo como objetivo de clasificación.

Una vez definido el objeto, este bloque se encarga de enviar la información de la posición, el color seleccionado, y el código RGB correspondiente hacia la siguiente etapa del sistema. En caso de que se haya recibido información únicamente de un bloque, ya sea desde Color_1 o de Color_2, será seleccionada la existente sin necesidad de realizar comparaciones. Por otro lado, si no se han recibido datos de ninguno de los colores, se activa una señal que realiza el cambio de cámara. Si en dos ocasiones consecutivas no se detectan objetos, se envía una señal para rotar al robot y continuar con la búsqueda en una nueva área de búsqueda.

Este bloque de código fue creado para el desarrollo de la presente aplicación, el código completo se muestra en el Anexo A6 al final del documento, su diagrama de flujo se muestra en la Figura 51.

- ***Bloque de código Rotacion***

Este módulo está encargado de realizar la rotación del humanoide cuando no se han detectado objetos con ninguna de las cámaras. Una vez que el humanoide haya rotado 3 veces, se habrá cubierto la zona de trabajo finalizando la clasificación. En este bloque se utilizó el módulo *ALMotion*, para la rotación y movimiento del robot.

La sintaxis de la función utilizada se puede observar a continuación, en donde el parámetro distancia es un dato tipo vector que contiene distancia en x, en y y un ángulo para la rotación del robot. En este caso se envía únicamente el ángulo que se desea que el humanoide rote para que se dirija hacia los depósitos de objetos.

```
self.motion.moveTo(distancia)//  Mover  el  robot  hacia  vector
definido
```

Este bloque de código fue creado para el desarrollo de la presente aplicación, el código completo se muestra en el Anexo A7 al final del documento.

3.3 Desarrollo del proceso de seguimiento de objetos

En el presente apartado se detallará el desarrollo del proceso de desplazamiento hacia objetos, los cálculos realizados para encontrar la posición del objeto con respecto al robot y la implementación en el software.

El proceso de seguimiento a objetos funciona de acuerdo al diagrama mostrado en la Figura 52.

3.3.1. *Cálculo de la distancia hacia el objeto*

Una vez obtenida la localización y el radio en el plano 2D del objeto de interés, se determina trayectoria que debe recorrer el robot para alcanzar dicho objeto.

Inicialmente se calcula la distancia existente entre la cámara y el objeto de interés, para ello se aplica la técnica de similitud triangular (Figura 53), la cual expresa que una

cámara monocular genera una relación uno-uno entre el objeto y la imagen que genera. (Emara, 2018)

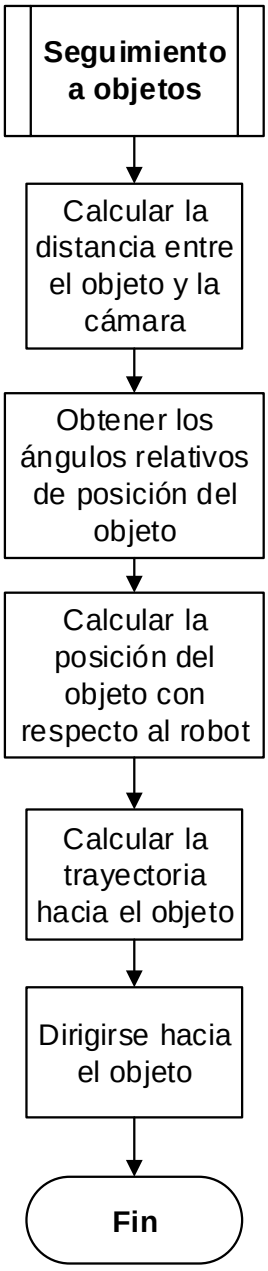

Figura 52. Diagrama de bloques del proceso de seguimiento a objetos

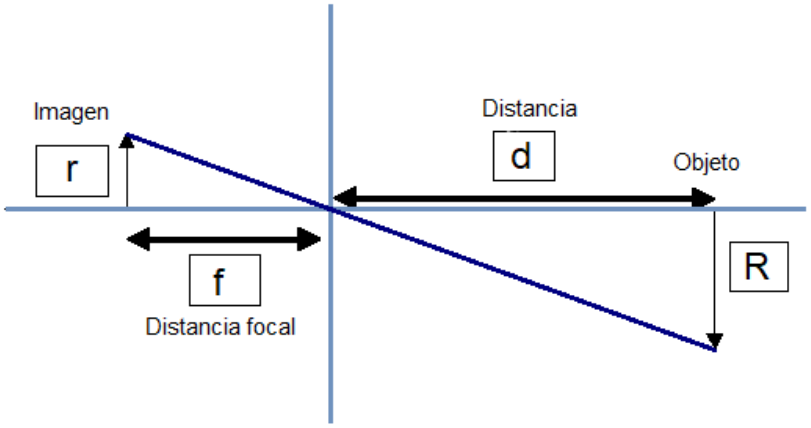

Figura 53. Técnica de similitud triangular.
Fuente:(Emara, 2018).

De la Figura 53 se deduce la siguiente fórmula,

$$\frac{f}{d} = \frac{r}{R} \qquad Ec.1$$

Donde:

- f = distancia focal (se obtiene mediante un proceso de calibración que se mostrará más adelante)
- d = distancia de la cámara al objeto
- r= radio del objeto generado por la cámara
- R = tamaño real del objeto

Para obtener el valor de la distancia focal se debe realizar un proceso de calibración que consiste en colocar un elemento a una distancia conocida, obtener el valor del radio que la cámara define y medir la longitud real del objeto, de esta forma la ecuación Ec.2 queda de la siguiente manera:

$$f = \frac{r \,.\, d}{R} \qquad Ec.2$$

Una vez hallada la distancia focal de la cámara, la ecuación Ec.2 es utilizada como se muestra a continuación, en donde los valores de la distancia focal, el radio de la imagen obtenida y el radio real son datos conocidos, permitiéndonos así obtener la distancia entre un objeto y una cámara.

$$d = \frac{r \,.\, f}{R} \qquad Ec.3$$

Posteriormente se obtienen los ángulos relativos de ubicación con respecto a la cámara, en la Figura 54 se puede observar una representación gráfica de los ángulos obtenidos.

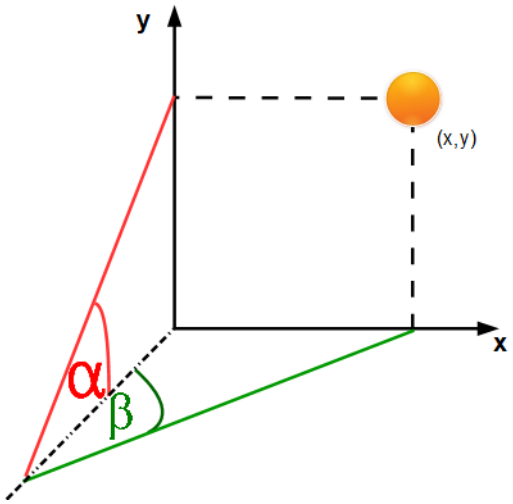

Figura 54. Ángulos relativos con respecto a la cámara

La trayectoria que debe seguir el robot se determina utilizando técnicas de geometría en el espacio, en la Figura 55 se puede observar un esquema del objeto de

interés en el plano espacial con respecto a un punto de observación, en este caso la cámara. En la Figura 56 se puede observar un extracto de los triángulos α y β .

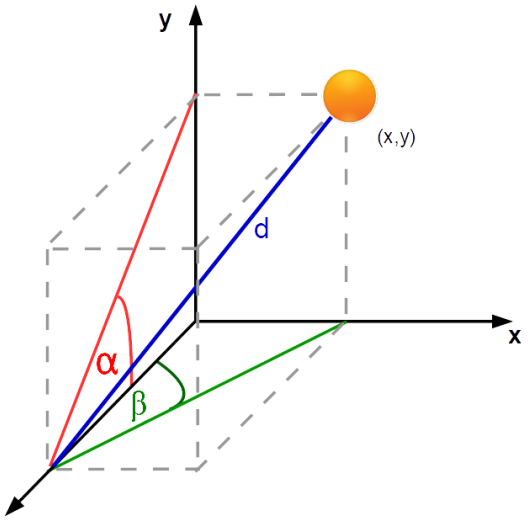

Figura 55. Esquema del objeto localizado.

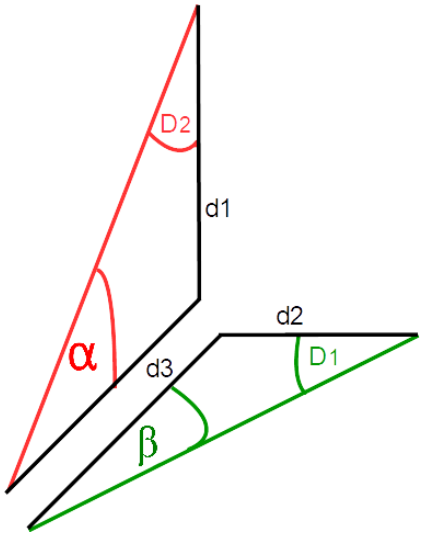

Figura 56. Cálculo de la trayectoria.

De la Figura 56, se pueden extraer las ecuaciones Ec.4 y Ec.5 aplicando ley de senos con respecto a los ángulos α y β.

$$\frac{d_3}{\sin D_1} = \frac{d_1}{\sin \beta} \qquad Ec.4$$

$$\frac{d_3}{\sin D_2} = \frac{d_2}{\sin \alpha} \qquad Ec.5$$

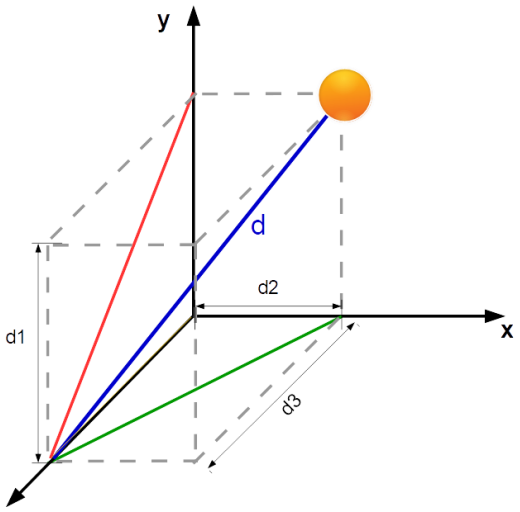

Figura 57. Esquema de datos para el cálculo de la trayectoria.

La ecuación Ec.6 se obtiene mediante la fórmula de Pitágoras en el espacio, la cual permite relacionar las distancias d_1, d_2 y d_3 con la distancia d obtenida previamente (Figura 57).

$$d^2 = d_1^2 + d_2^2 + d_3^2 \qquad Ec.6$$

Despejando las variables d_1, d_2 de las ecuaciones Ec.4 y Ec.5 obtenemos,

$$d_1 = \frac{d_3 \,.\, \sin \beta}{\sin D_1} \qquad Ec.7$$

$$d_2 = \frac{d_3 \,.\, \sin \alpha}{\sin D_2} \qquad Ec.8$$

Reemplazando las ecuaciones Ec.7 y Ec.8 en Ec.3 tenemos,

$$d^2 = \left(\frac{d_3 \,.\, \sin\alpha}{\sin D_2}\right)^2 + \left(\frac{d_3 \,.\, \sin\beta}{\sin D_1}\right)^2$$

Y despejando la variable d_3 se obtiene la siguiente expresión:

$$d_3 = \sqrt{\frac{d^2}{1 + \left(\frac{\sin\alpha}{\sin D_2}\right)^2 + \left(\frac{\sin\beta}{\sin D_1}\right)^2}} \qquad \text{Ec. 9}$$

Donde:

- d_3 = distancia en el eje x desde el robot hacia el objeto
- d_2 = distancia en el eje y desde el robot hacia el objeto

3.3.2. *Implementación del proceso de desplazamiento hacia objetos*

La implementación del programa de seguimiento a objetos se puede observar en la Figura 58, está conformado por dos bloques de código, los cuales se detallan a continuación.

- **Bloque de código Calculo_Distancias**

Como se mencionó en la sección anterior, el primer paso es convertir los datos de ubicación del objeto en el plano 2D hacia ángulos relativos de posición, para ello se realiza una llamada al módulo *ALVideoDevice* utilizando la siguiente línea de código.

```
self.conversion = ALProxy("ALVideoDevice")// Módulo para
conversión de datos
```

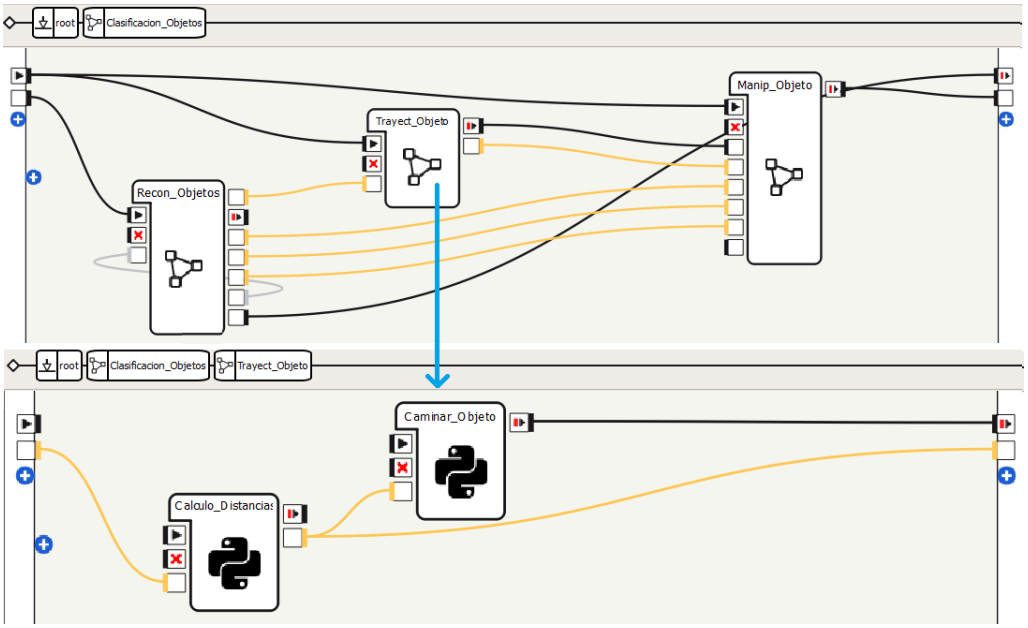

Figura 58. Proceso de seguimiento a objetos

El proceso de conversión se realiza mediante la función mostrada a continuación, la cual recibe dos parámetros que son: el índice de la cámara que puede ser ‘1’ para la cámara inferior o ‘0’ para la cámara superior, y el vector de posición obtenido en la sección 3.2.

```
value= getAngularPositionFromImagePosition(cam_index,
self.posicion) //Conversión de datos, de pixeles a ángulos
```

Una vez hallados los ángulos, se procede a plasmar las ecuaciones halladas en la sección 3.3.1, en el siguiente bloque de código se puede observar la implementación de

las funciones antes mencionadas. Cabe recalcar que para utilizar las funciones *seno* y *coseno*, se debe importar el módulo *almath* al código.

```
[d1,d2] = [1.5708 - math.fabs(beta),1.5708 - math.fabs(alpha)]
// Cálculo de ánguos d1 y d2
[sin_alpha, sin_beta]=[math.sin(alpha), math.sin(beta)]
// Cálculo del seno de los ángulos alpha y beta
[sin_d1, sin_d2]=[math.sin(d1), math.sin(d2)]
// Cálculo del seno de los ángulos d1 y d2
valor_m=math.pow(sin_beta/sin_d1,2)
valor_n=math.pow(sin_alpha/sin_d2,2)
d2=math.sqrt(math.pow(self.d,2)/(valor_n+valor_m+1))
//Cálculo de d2
d3=dist_y*sin_alpha/sin_d2
// Cálculo de d3
```

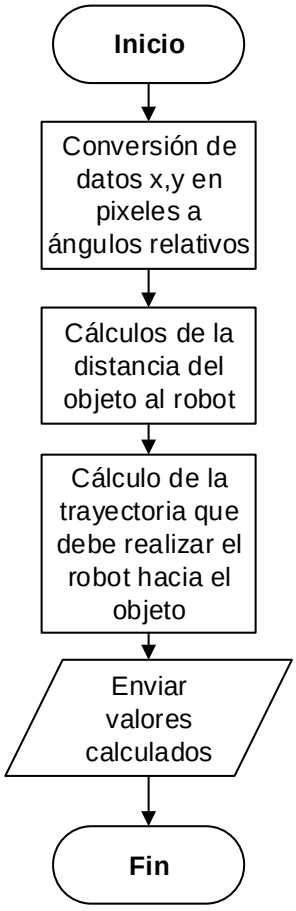

Figura 59. Diagrama de flujo del bloque Calculo_Distancias

Este bloque fue creado para el desarrollo de la presente aplicación, el diagrama de flujo se muestra en la Figura 59 y el código completo se puede observar en el Anexo A8 al final del documento.

- **Bloque de código Caminar_Objeto**

Cuando la posición del objeto con respecto al robot ya se ha determinado, se envía al robot a caminar hacia el elemento, para ello se utiliza la función moveTo perteneciente al módulo *ALMotion.*

```
self.motion.moveTo(self.distancia[0]/100, self.distancia[1]/100,
0)// Mover robot hacia la posición definida
```

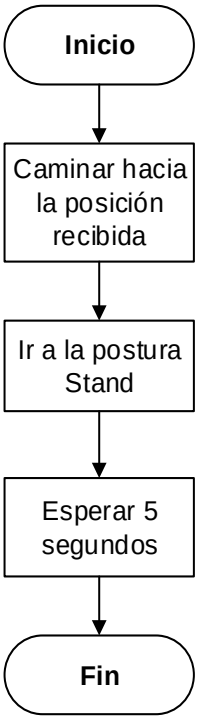

Figura 60. Diagrama de flujo del bloque Caminar_Objeto

Este bloque fue creado para el desarrollo de la presente aplicación, el diagrama de flujo se muestra en la Figura 60 y el código completo se puede observar en el Anexo A9 al final del documento.

3.4 Desarrollo del proceso de manipulación de objetos

El proceso de manipulación de objetos es en encargado de tomar el objeto, una vez que el robot se encuentre a una distancia menor a los 20 cm, y retornar hacia la posición inicial para buscar los depósitos, acercase a ellos y colocar el elemento que ha tomado previamente en el lugar que corresponda. Para localizar los depósitos del color 1 y 2 respectivamente, se utilizaron los *NAOMark,* que son conjunto de imágenes de referencia con patrones que el robot tiene la capacidad de distinguir, además son de rápido reconocimiento que permiten localizar zonas de interés y extraer información relacionada.

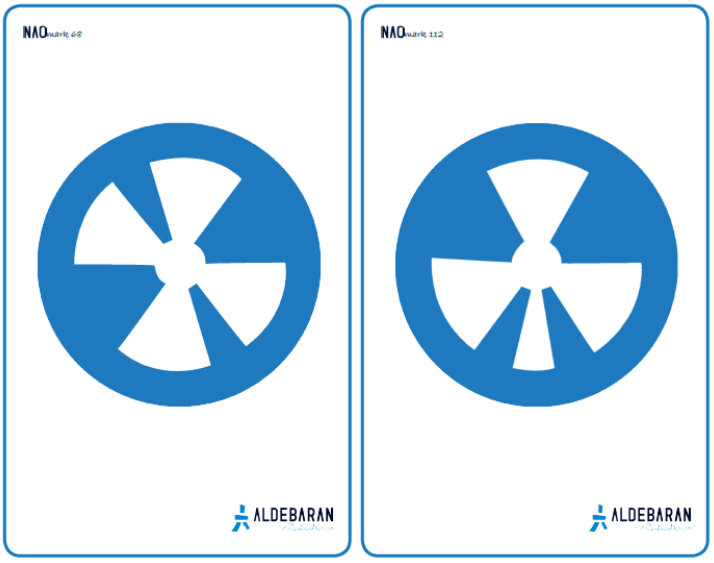

Figura 61. NAOMarks

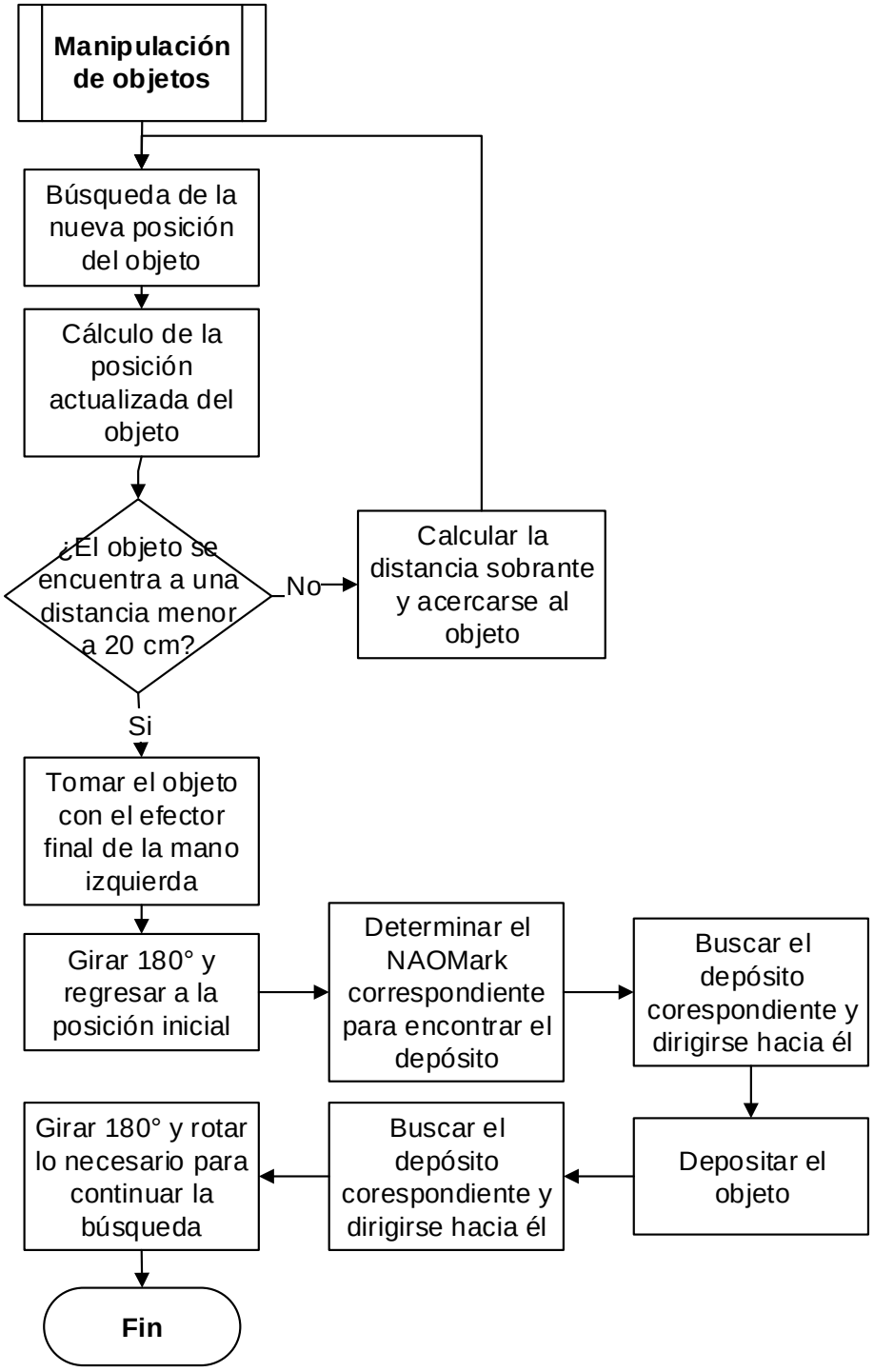

Figura 62. Diagrama de bloques del proceso de manipulación de objetos

Para la presente aplicación se utilizaron los *NAOMark* 68 y 112, en la Figura 61 se puede observar las marcas seleccionadas, las cuales estarán colocadas sobre cada depósito. Por defecto los objetos del color 1 serán depositados en el contenedor colocado bajo la marca 68 y los elementos del color 2 con la marca 112, es posible modificar las marcas que se utilicen de acuerdo con la gama de patrones existentes, así como la relación entre las *NAOMarks* y los colores. En el Anexo B del presente documento, se encuentra la librería completa de NAOMarks provista por Aldebaran Robotics.

El proceso de manipulación de objetos funciona de acuerdo al diagrama de bloques mostrado en la Figura 62 , posteriormente en la Figura 63 se muestra la implementación del proceso de manipulación de objetos en el software.

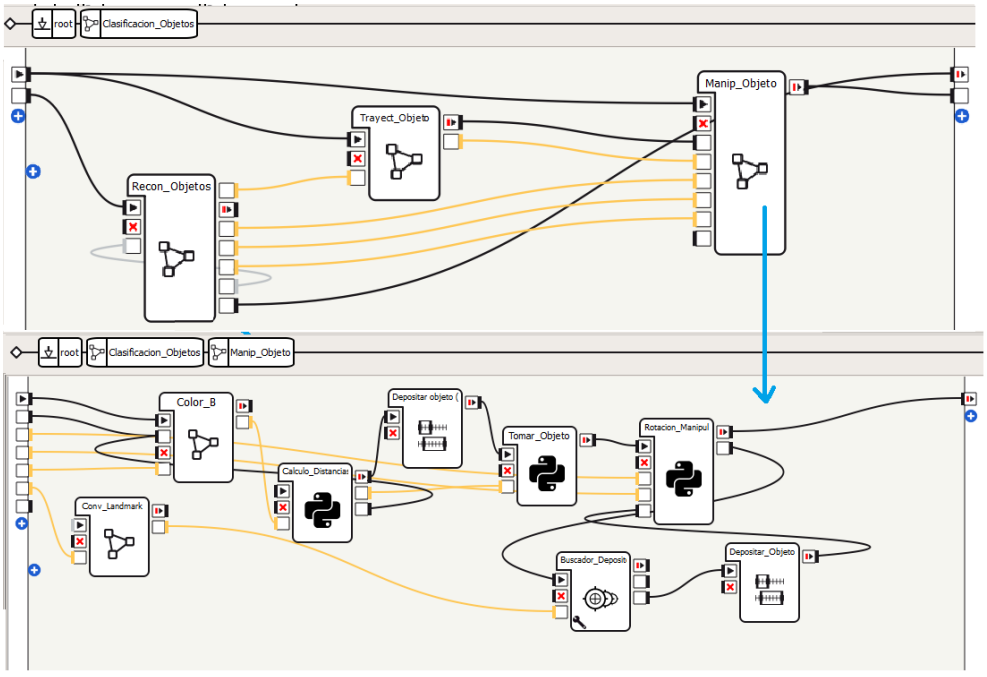

Figura 63. Implementación del proceso de manipulación de objetos.

El programa de manipulación de objetos es sin duda alguna, el más extenso de los presentados previamente y está conformado por 8 bloques de código los cuales serán detallados a continuación.

- **Bloque de código Conv_Landmark**

Este bloque de código se encarga de seleccionar el código del NAOMark correspondiente de acuerdo con el color del objeto seleccionado para la clasificación. El diagrama de flujo correspondiente se puede observar en la Figura 64.

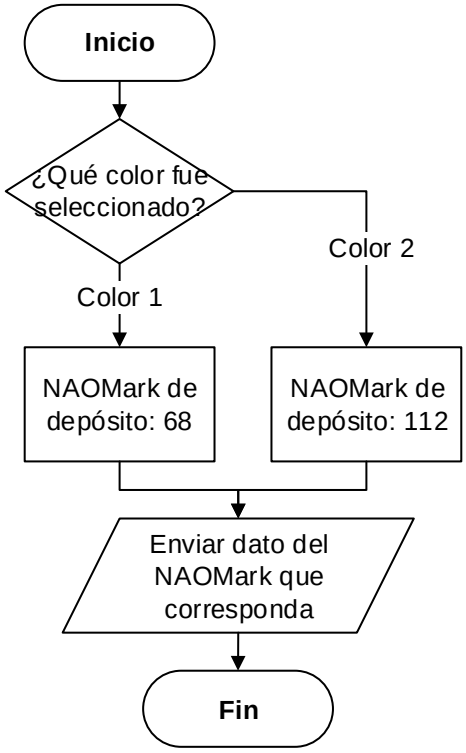

Figura 64. Diagrama de flujo del bloque Conv_Landmark

Como se mencionó anteriormente, por defecto el NAOMark 68 corresponderá al color 1 y el NAOMark 112 al color 2. Esto puede modificarse de acuerdo a las

necesidades, ya sea la relación entre cada color con la marca deseada o el uso de las marcas existentes en la librería provista por el fabricante. (Vèase Anexo B.)

Este bloque de código fue creado para el desarrollo de la presente aplicación, el código completo se muestra en el Anexo A10 del presente documento.

- ***Diagrama Color_B***

El presente diagrama funciona de la misma manera que los bloques Color_1 y Color_2 detallados en la sección 3.2., sin embargo, en este caso el código RGB no es ingresado por el usuario, sino que es enviado desde el proceso de Reconocimiento de objetos de acuerdo con el color seleccionado.

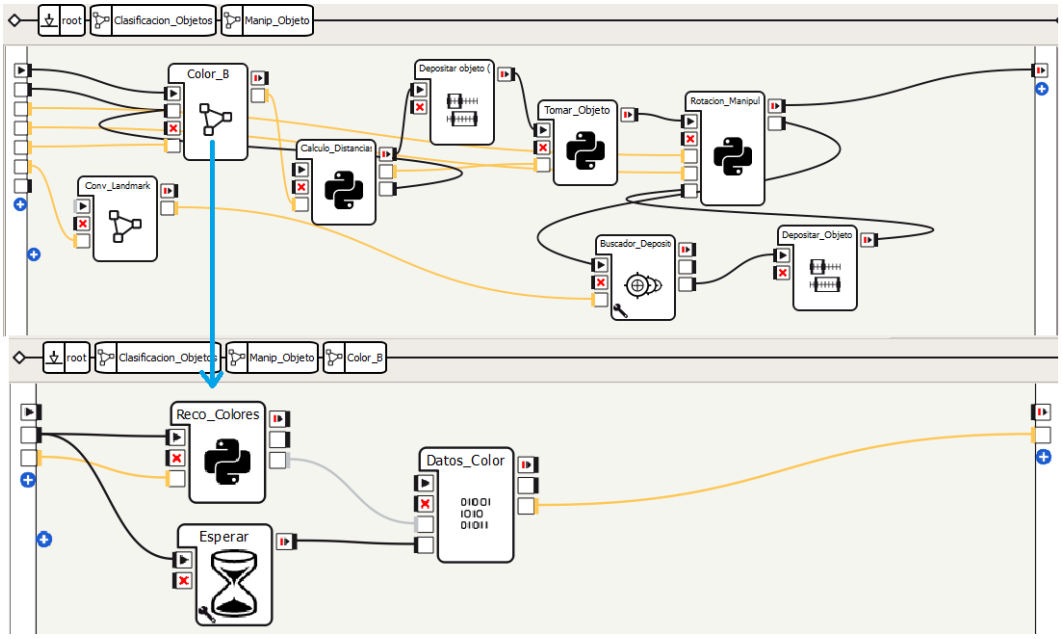

Figura 65. Contenido del diagrama Color_B

El objetivo de esta sección es encontrar los nuevos datos de posición del elemento de interés, debido a que el robot ya no se encuentra en la posición inicial y se deben actualizar los datos para proceder con la manipulación y toma del objeto. En la Figura 65 se puede observar el contenido del diagrama Color_B.

- ***Bloque de código Calculo_Distancias_Manip***

El bloque Distancias contenido en el proceso de manipulación de objetos calcula la nueva posición del objeto de interés, si dicho elemento se encuentra a una distancia menor a 20 cm, se envían los datos obtenidos hacia la siguiente fase para tomar el objeto. En caso de que dicha distancia sea mayor a 20 cm, el robot de desplazará hacia una nueva posición según los cálculos realizados, e iniciará nuevamente el proceso de búsqueda con el bloque Color_B. El diagrama de flujo correspondiente se puede observar en la Figura 66.

La programación y el cálculo de los valores es el mismo con respecto al bloque Distancias del proceso de Desplazamiento hacia objetos. Este bloque fue creado para el desarrollo de la presente aplicación, el código completo se muestra en el Anexo A11 al final del documento.

- ***Bloque de código Tomar_Objeto***

En esta sección se realizan los movimientos del brazo izquierdo del robot para poder tomar el objeto y continuar con el proceso de clasificación. El diagrama de flujo correspondiente se muestra en la Figura 67.

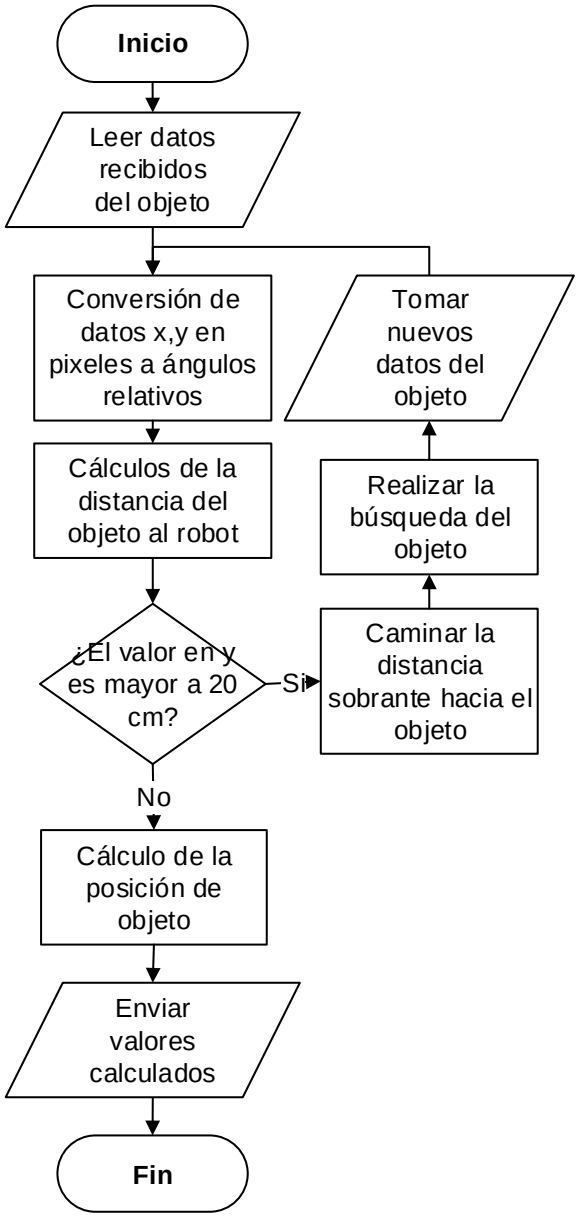

Figura 66. Diagrama de flujo del bloque Calculo_Distancias_Manip

Posteriormente se utiliza la función setPosition para mover el efector final localizado en la mano del humanoide hacia la posición del objeto obtenida previamente, a continuación se puede observar la sintaxis de la función.

```
self.motion.setPosition(cadena, espacio, objetivo, velocidad,
mascara) // Mover efector final hacia la posición deseada
```

En donde, *cadena* es el nombre del conjunto de articulaciones utilizadas para alcanzar el punto deseado, en este caso se coloca "LArm" puesto que se está utilizando el brazo izquierdo para tomar el objeto; *espacio* representa el sistema de referencia sobre el cual está localizado el objeto que se desea alcanzar, *objetivo* es un vector que contiene la posición del objeto , *velocidad* es un valor comprendido entre 0 y 1 que representa la velocidad de movimiento con el cual se realizarán los movimientos y *mascara* es un dato constante de valor ‘7’.

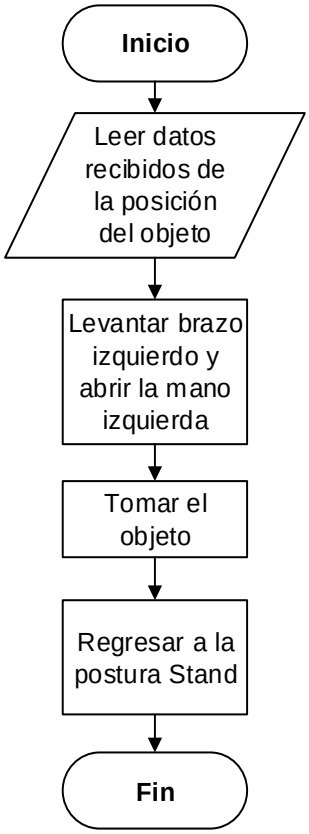

Figura 67. Diagrama de flujo del bloque Tomar_Objeto

El humanoide NAO posee un actuador similar a una mano con 3 dedos, que puede ser abierta o cerrada mediante líneas de código, para ello se utiliza el módulo *ALMotion* y la función mostrada a continuación.

```
self.motion.openHand('LHand')// Abrir mano izquierda
self.motion.closeHand('LHand') //Cerrar mano izquierda
```

Para el primer movimiento realizado con la función setPosition, se utilizan los valores *x* y *y* resultantes del cálculo de la posición, y para el caso *z* o de la altura del objeto, se dirige 5 cm por sobre él, abre la mano, luego de 3 segundos de pausa, desciende el efector hacia la posición real del objeto, cierra la mano y regresa a la postura anterior.

Este bloque fue creado para el desarrollo de la presente aplicación, el código completo se muestra en el Anexo A12 al final del documento.

- ***Bloque de código Rotacion_Manipul***

En esta sección el bloque de rotación se encarga de enviar al robot a la posición inicial en la que se encontraba antes de acercarse al objeto, para ello, una vez tomado el objeto de su pedestal respectivo, gira 180º y realiza la misma trayectoria que realizó previamente para volver al punto inicial de búsqueda. Este bloque fue creado para el desarrollo de la presente aplicación, el código completo se muestra en el Anexo A13 del presente documento, su diagrama de flujo se muestra en la Figura 68.

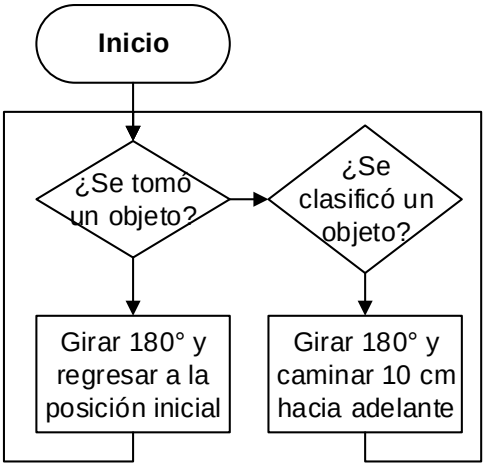

Figura 68. Diagrama de flujo de Rotacion_Manipul

- ***Bloque de código Buscador_Deposito***

Como se mencionó anteriormente, la diferenciación de los depósitos se realizó colocando NAOMarks sobre cada uno de ellos, en este bloque de código se realiza la búsqueda de las marcas y seguimiento hacia su posición. Para acceder a los elementos de memoria que contienen la información de las marcas se debe llamar al módulo ALLandMarkDetection, como se describe a continuación.

```
self.tracker=ALProxy("ALLandMarkDetection”)// Módulo para
NAOMark
```

Para implementar el seguimiento hacia el NAOMark se deben configurar características de la localización del robot con respecto a la marca, para ello se utiliza la siguiente función.

```
self.tracker.setRelativePosition([-distanciaX, distanciaY, anguloWz,
thresholdX, thresholdY, thresholdWz]) // Definiciòn de los parámetros
de búsueda de los NAOMarks
```

En donde *distanciaX y distanciaY* representan la distancia en el eje *x* y *y* a la cuál debe mantenerse el robot cuando tenga cerca la marca, a*nguloWz* se encarga de la rotación del robot con respecto al NAOMark. Los valores restantes son los datos de margen de error permitidos para cada variable.

Una vez iniciada la búsqueda del depósito, la función track se encarga del seguimiento del robot hacia la marca correspondiente.

```
self.tracker.track("Landmark")// Búsqueda y seguimiento
                        del NAOMark
```

- ***Timeline Depositar_Objetos***

Este bloque es de tipo Timeline, es decir, está formado por una serie de cuadros de movimiento que se ejecutan de forma ordenada. Se encarga de ejecutar el depósito de los objetos en sus contenedores correspondientes una vez haber llegado hasta ellos. En la Figura 69 se puede observar los movimientos que se ejecutan en el presente bloque.

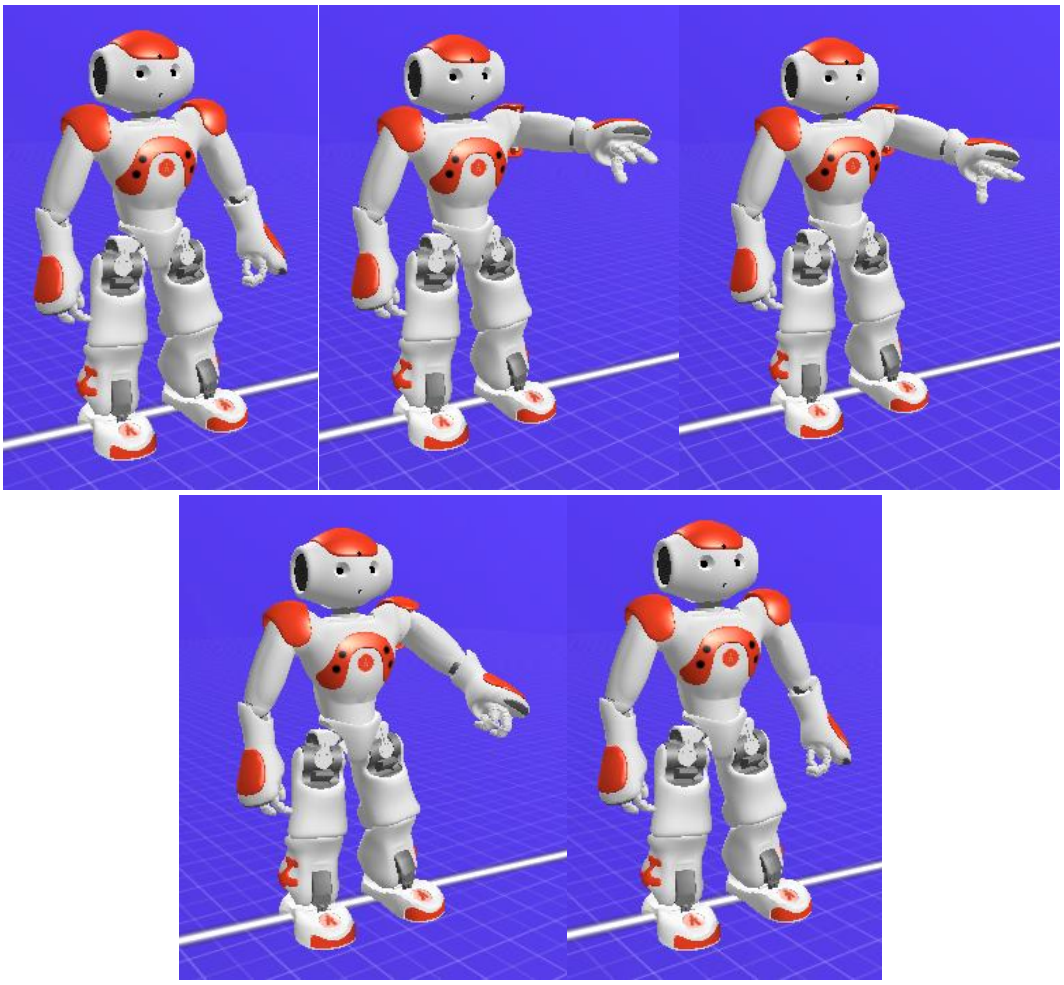

Figura 69 Secuencia de movimientos realizada en la sección Depositar_Objetos.

CAPÍTULO IV

PRUEBAS Y RESULTADOS

4.1 Pruebas

Para la realización de pruebas y análisis de la aplicación creada, se utilizó un modelo de *Diseño de experimentos sencillos,* cuyo objetivo es descubrir si ciertos factores determinados influyen en los resultados obtenidos de una variable de interés. La metodología de este modelo es la experimentación, puesto que, si un experimento se repite, en condiciones indistinguibles, los resultados tienden a presentar variabilidad.

La clave para la implementación de un modelo de esta naturaleza, reside en definir correctamente los factores y sus respectivos niveles, donde un factor es una variable de interés que posiblemente afecte a un resultado o respuesta que se desea analizar, cada factor tiene distintos grados o tipos que pueden ser aplicados al experimento, a esto se denominan los niveles del factor. Los factores pueden ser cualitativos o cuantitativos.

En cada una de las pruebas se definió los factores y sus respectivos niveles aplicados, los cuales se especifican a continuación.

4.1.1 Prueba 1: Error entre la distancia calculada y la distancia real

Consiste en calcular el error existente entre la distancia calculada desde el robot hacia un objeto de interés localizado dentro del área de trabajo. Para realizar esta prueba se colocó el robot en el área de trabajo, frente a él se ubicaron los objetos rojo, azul y

verde, uno a la vez, dispuestos a distancias aleatorias en cada tratamiento y en 3 distintos sectores con respecto al robot: centro, izquierda y derecha.

En este caso se tienen los siguientes factores con sus respectivos niveles:

- Color: Rojo, Azul y Verde
- Ubicación con respecto al robot: Centro, Izquierda, Derecha

En la Figura 70 se muestra el experimento realizado para el color azul, en la Figura 71 se muestra el experimento para el color rojo y en la Figura 72 se muestra el experimento para el color verde.

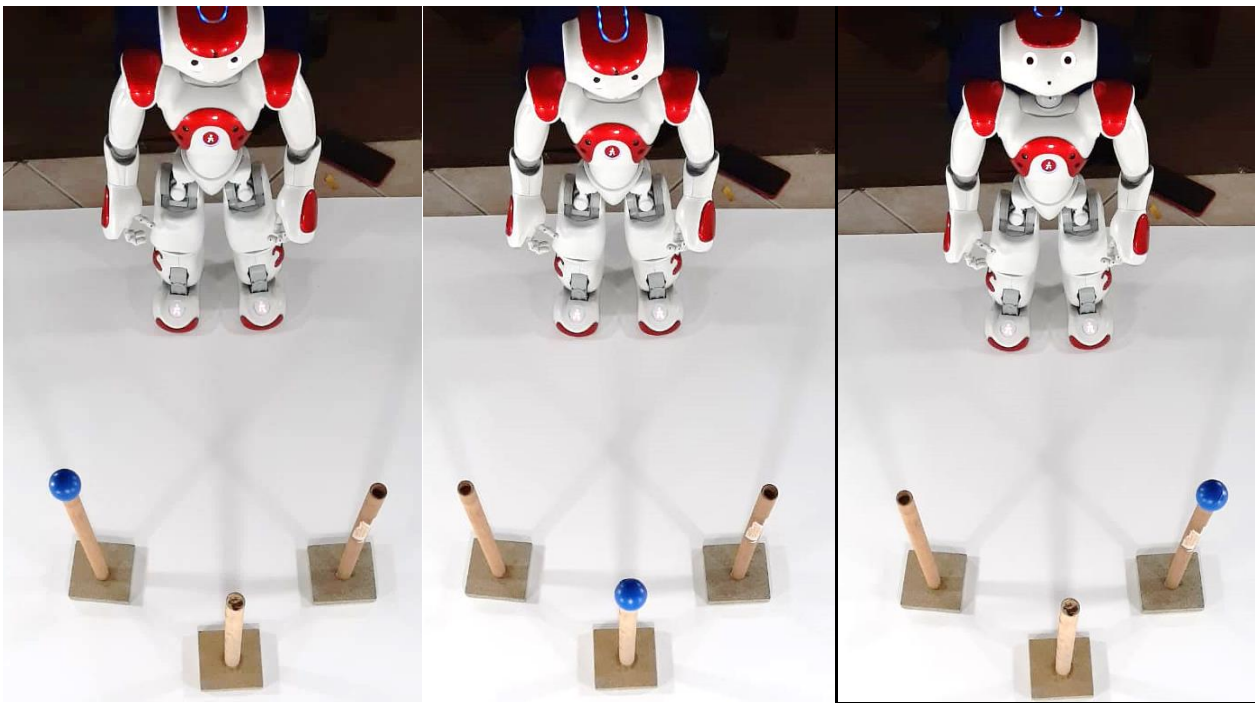

Figura 70. Prueba 1 – Color azul

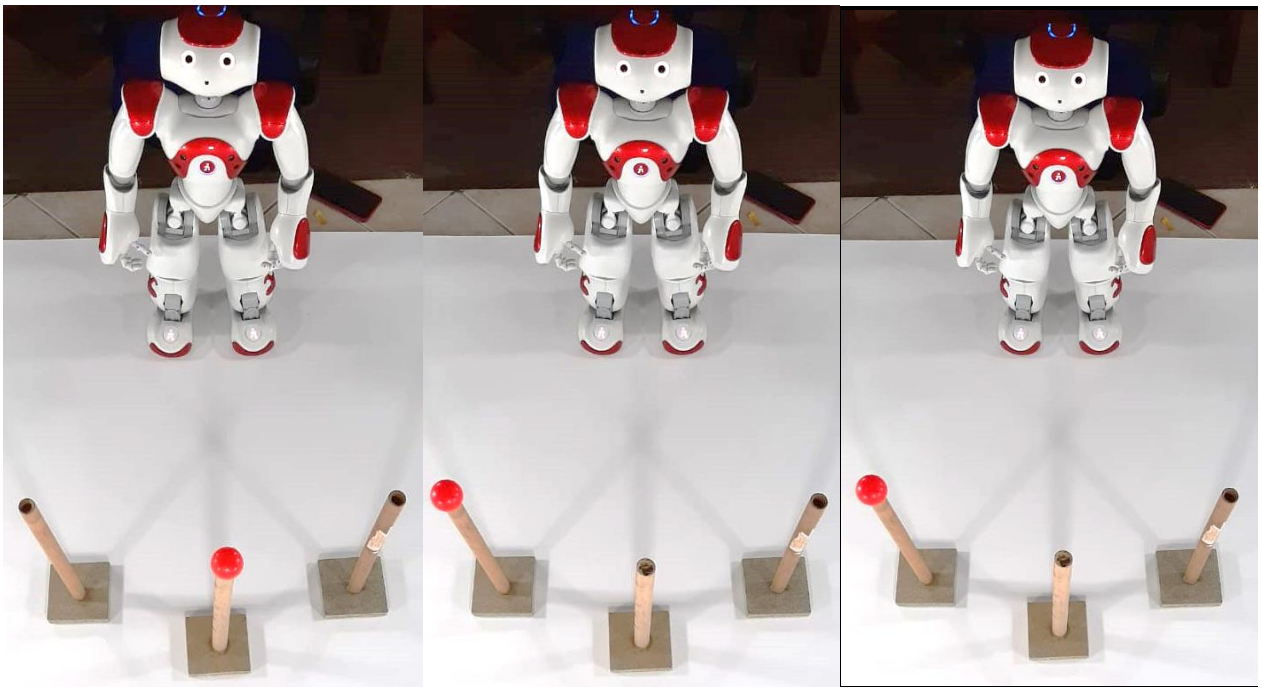

Figura 71. Prueba 1 – Color rojo

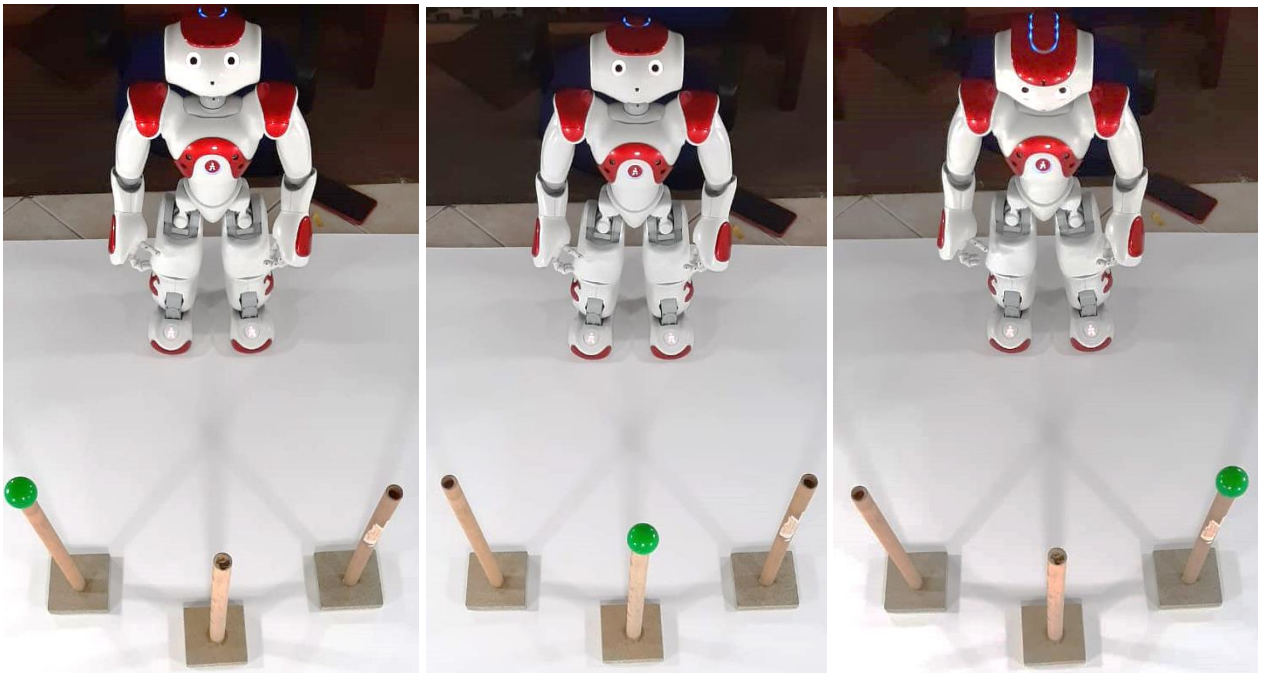

Figura 72. Prueba 1 – Color verde

Se tomaron los datos de la distancia entre en objeto y la cámara calculados en el programa y se compararon con los datos reales medidos para cada tratamiento.

Los resultados obtenidos y su análisis se muestran en la sección 4.2.1.

4.1.2 Prueba 2: Error entre la posición calculada y la posición real

El escenario de esta prueba es el mismo al detallado en la sección 4.1.1, la diferencia reside en que, en este caso se calcula el error entre la posición calculada y la posición real de los objetos con respecto al robot, además se realiza un análisis cualitativo en el cuál se especifica si el robot llegó o no al objeto o a una posición que le permita continuar con el proceso de clasificación.

En este caso se tienen los siguientes factores con sus respectivos niveles:

- Color: Rojo, Azul y Verde
- Ubicación con respecto al robot: Centro, Izquierda, Derecha

El umbral de tolerancia del error se determina para cada eje coordenado y fue calculado de acuerdo al alcance del actuador del brazo izquierdo del humanoide, así como del rango de visión de la cámara inferior. En la Tabla 7 se muestran los valores antes mencionados.

Tabla 7

Tolerancia de error para las coordenadas calculadas.

	Rango admitido	**Rango reajuste**
x	[-3,5 ; +1,5]	N/A
y	[-3 ; +5]	[-10; -3] y [+5 ; +10]

Existen 4 casos que pueden generarse de acuerdo a los resultados obtenidos, los cuales son:

- Admitido: El error generado en el tratamiento es muy bajo y se encuentra dentro del umbral de tolerancia admitido, es posible continuar con el proceso.
- Reajuste: El error generado en el tratamiento es considerable, no se encuentra dentro del rango admitido, pero si es menor al valor máximo para reajuste y se requiere repetir el cálculo de la distancia y trayectoria para acercarse de nuevo al objeto.
- Perdido: La coordenada y calculada tiene un error alto que está fuera del umbral de tolerancia para reajuste, el robot perdió al objeto del plano de visión al desplazarse.
- Crítico: El error está por fuera del umbral de tolerancia y posiblemente se genere una colisión del robot contra el pedestal del objeto. Sucede cuando el valor de la coordenada x supera el máximo admitido.

Los resultados obtenidos y su análisis se muestran en la sección 4.2.2.

4.1.3 Prueba 3: Análisis del sistema de clasificación

Esta prueba consiste en analizar que objetos fueron clasificados correctamente y cuáles fueron las causas para que la población restante no haya formado parte del caso de éxito. A continuación se muestra el procedimiento que seguirá el robot durante esta prueba.

1.- El robot es ubicado en el punto de inicio, se presiona el sensor para iniciar el sistema (Figura 73).

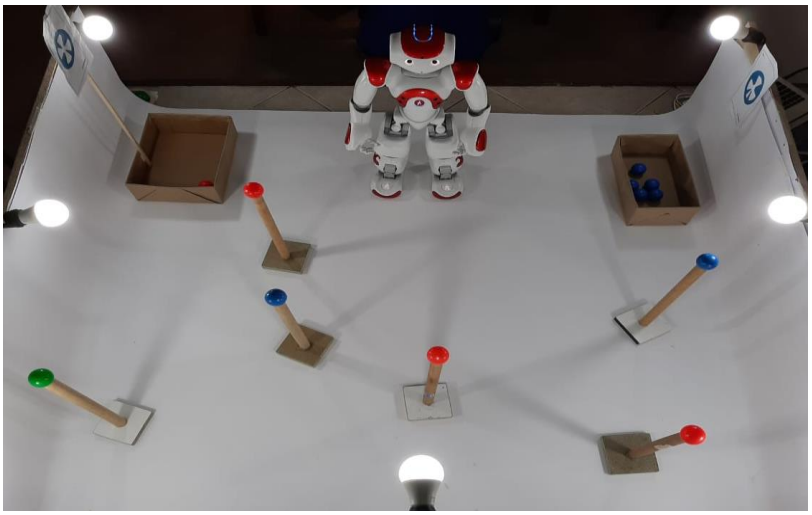

Figura 73. Paso 1 del sistema de clasificación de objetos.

2.- El humanoide gira 60° de su posición inicial para iniciar la búsqueda en la primera sección del área de trabajo (Figura 74).

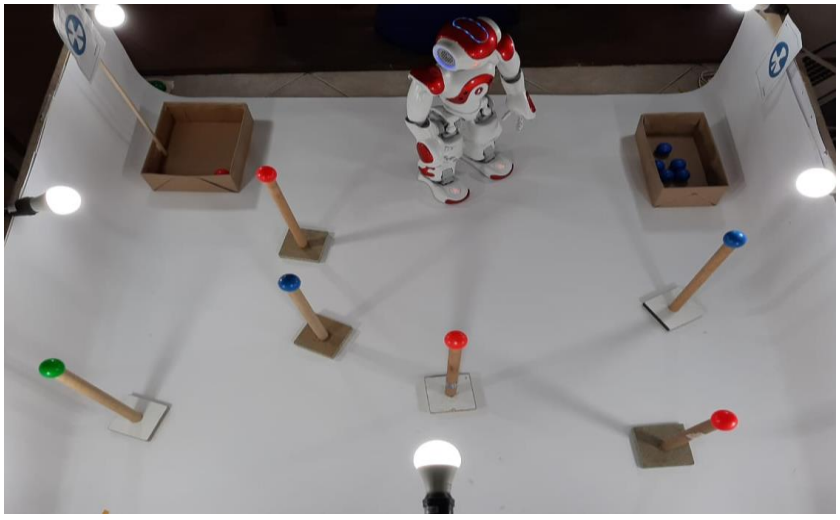

Figura 74. Paso 2 del sistema de clasificación de objetos

3.- Luego de realizar una búsqueda de los objetos presentes en la sección, se selecciona el de mayor diámetro y el robot se acerca hacia él para poder tomarlo con su mano izquierda (Figura 75).

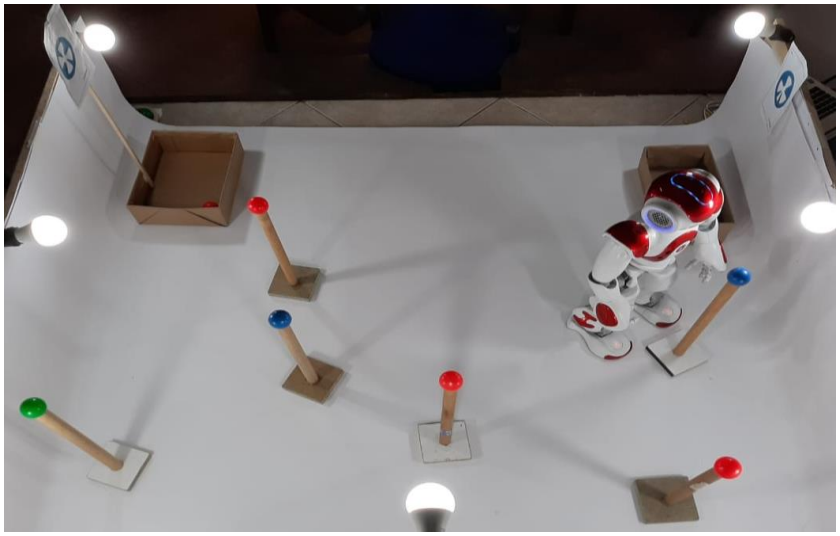

Figura 75. Paso 3 del sistema de clasificación de objetos

4.- Cuando el humanoide sujeta el objeto en su mano, analiza el color del mismo para determinar el depósito al cual pertenece, realiza un giro apuntando hacia la dirección de los depósitos y se acerca al que corresponda.

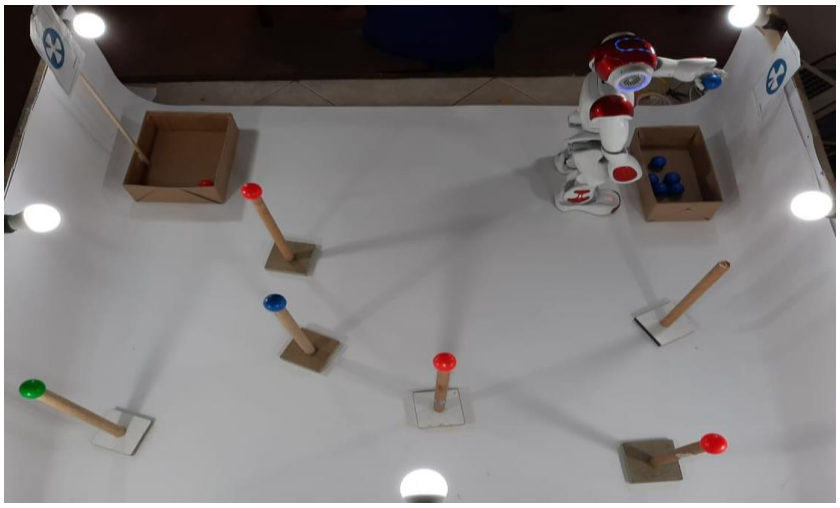

Figura 76. Paso 4 del sistema de clasificación de objetos

5.- Deposita el objeto y regresa a la posición inicial para continuar con la búsqueda en la primera sección (Figura 77).

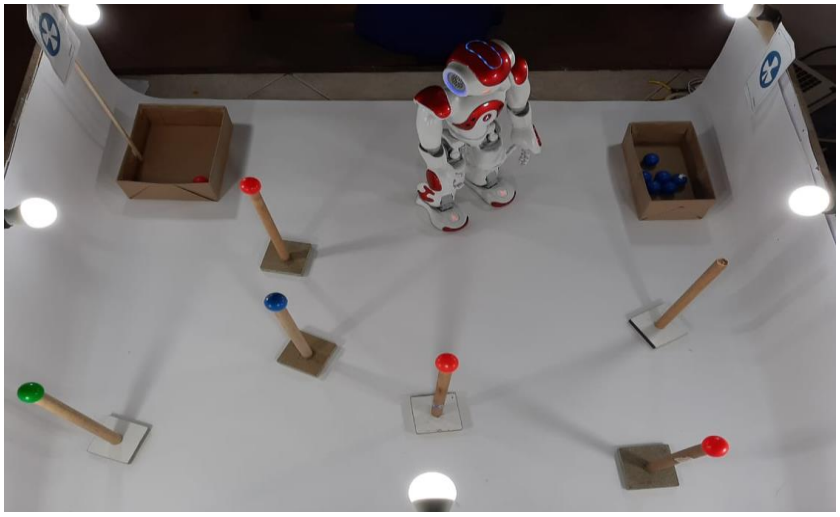

Figura 77. Paso 5 del sistema de clasificación de objetos

6.- En caso de no existir más objetos en la sección actual, el humanoide gira 60° para poder cubrir la nueva sección (Figura 78).

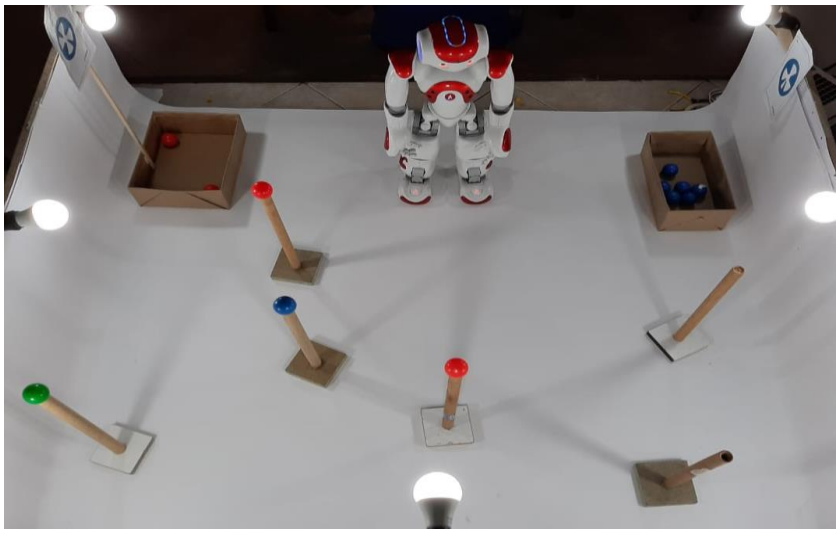

Figura 78. Paso 6 del sistema de clasificación de objetos

7.- A continuación se repiten los pasos 3, 4, 5 y 6 hasta completar la clasificación de objetos en toda la zona de trabajo.

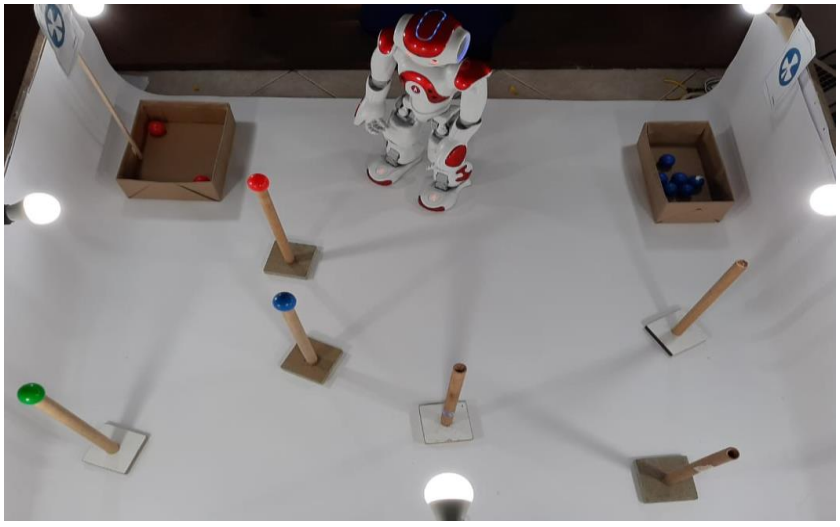

Figura 79. Paso 7 del sistema de clasificación de objetos

8.- Cuando el robot ha detectado que no existen más objetos de interés y que toda el área de trabajo ya ha sido completada, procede a finalizar la aplicación y es llevado al modo de descanso (Figura 80).

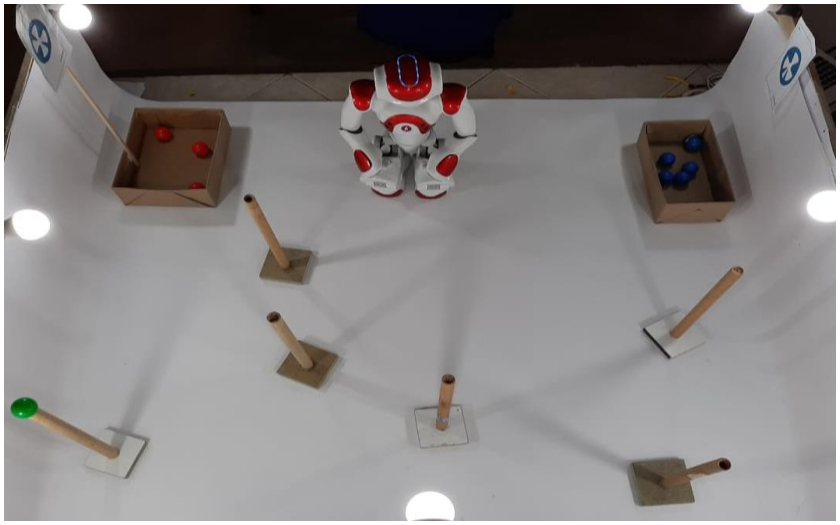

Figura 80. Paso 8 del sistema de clasificación de objetos

En esta prueba se realizaron 3 experimentos distintos, cada uno con 8 objetos a clasificar en el medio en el que se encuentra, de los cuales 7 son objetos clasificados y 1 es un objeto cuyo color no es de interés para la aplicación. Los resultados obtenidos se muestran en la sección 4.2.3.

4.2 Resultados

4.2.1 Prueba 1

Para cada tratamiento se realizaron 8 repeticiones, a continuación se muestran los resultados obtenidos para cada color.

- ***Color Azul***

En la Figura 81 se pueden observar los resultados obtenidos del experimento con color azul y el objeto ubicado en el centro del plano, para la ubicación a la izquierda del plano se muestra en la Figura 82 y finalmente para el lado derecho en la Figura 83.

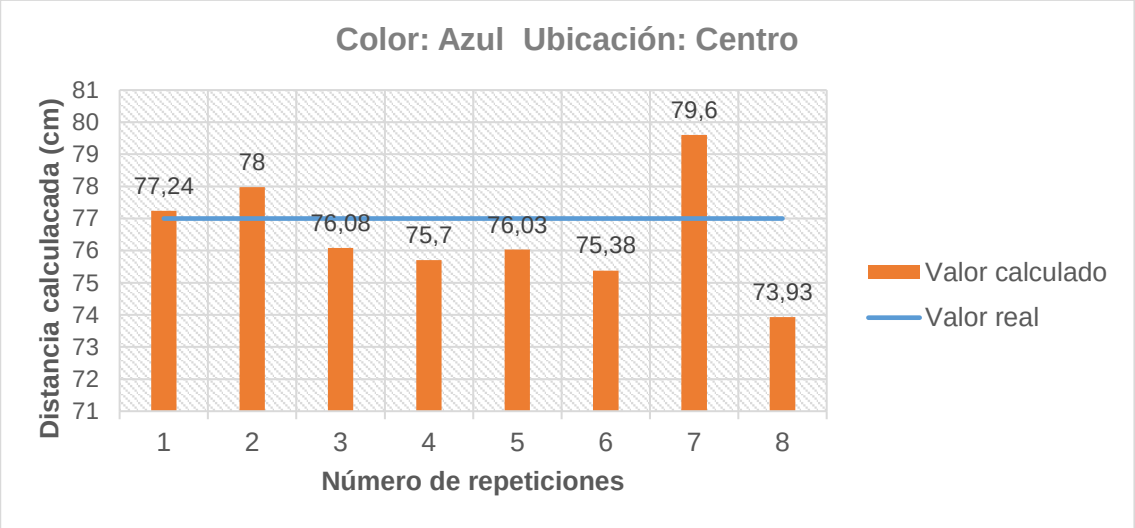

Figura 81. Resultados obtenidos para el color azul, ubicación centro.

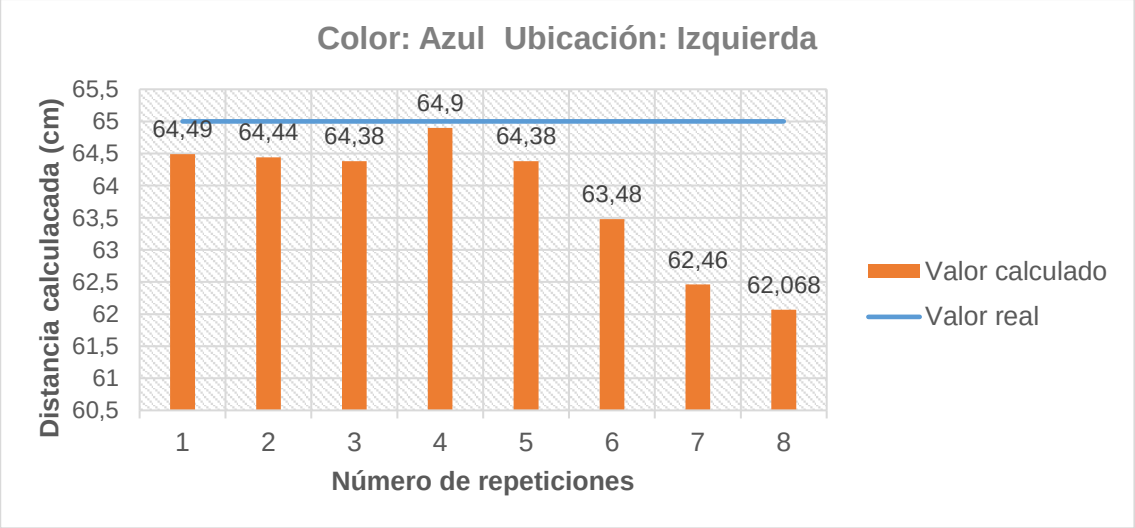

Figura 82. Resultados obtenidos prueba 2: color azul, ubicación izquierda.

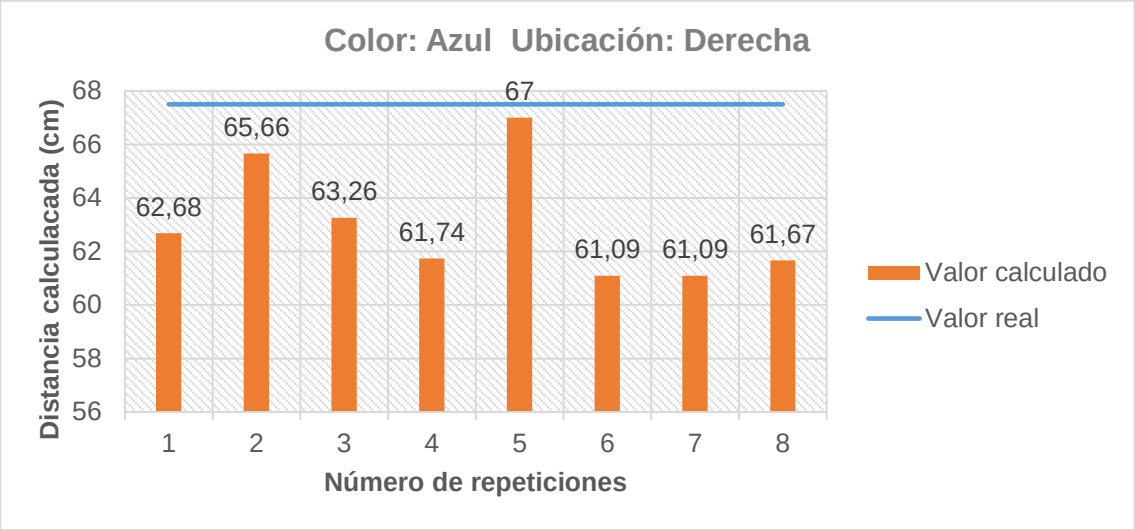

Figura 83. Resultados obtenidos prueba 2: color azul, ubicación derecha.

En la Tabla 8 se puede observar el error calculado para cada uno de los tratamientos realizados para el color azul.

Tabla 8

Error porcentual en cálculo distancia para objetos de color azul.

		Ubicación		
		Centro	**Izquierda**	**Derecha**
Número de experimento	**1**	0,31%	0,78%	7,14%
	2	1,27%	0,86%	2,73%
	3	1,19%	0,95%	6,28%
	4	1,69%	0,15%	8,53%
	5	1,26%	0,95%	6,28%
	6	2,10%	2,34%	9,50%
	7	3,38%	3,91%	9,50%
	8	3,99%	4,51%	8,64%

Como se puede observar en la Tabla 8, los porcentajes de error son bajos cuando los objetos se encuentran ubicados en el centro y en el lado izquierdo, para el lado derecho son valores superiores al 5% y se dan debido a que la iluminación implementada no es uniforme en su totalidad y existen espacios en los cuales se generan sombras que afectan al reconocimiento de objetos y por ende al análisis de resultados.

- ***Color rojo***

En la Figura 84 se pueden observar los resultados obtenidos para los objetos de color rojo ubicados en el centro, para la ubicación en el lado izquierdo se muestra en la Figura 85 y del lado derecho en la Figura 86 .

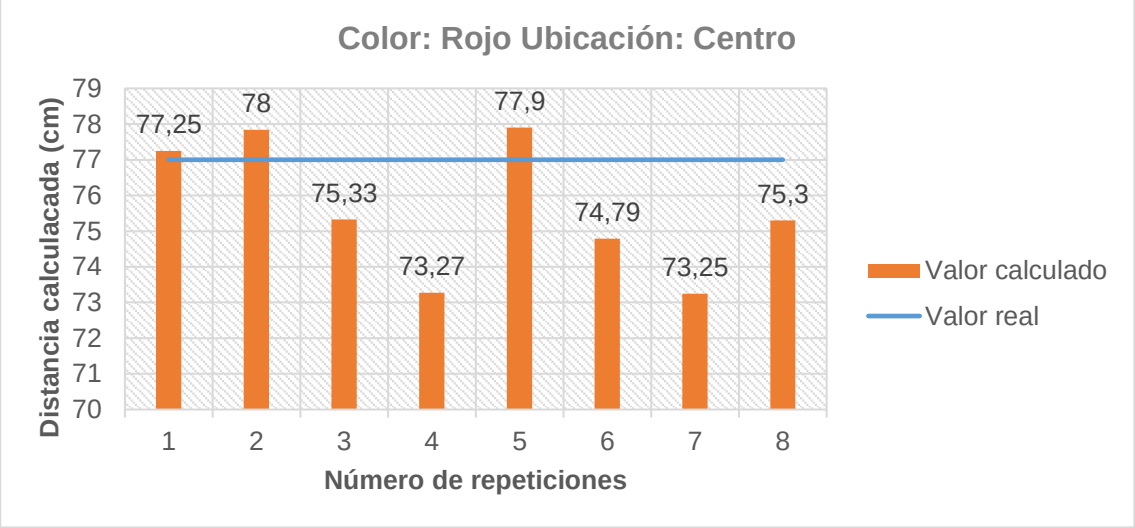

Figura 84. Resultados obtenidos prueba 2: color rojo, ubicación centro.

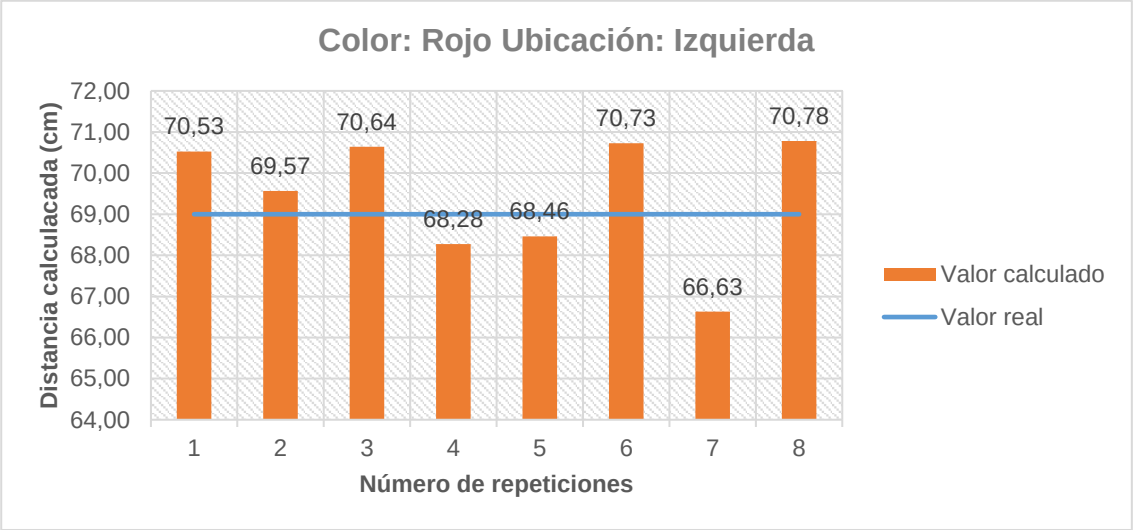

Figura 85. Resultados obtenidos prueba 2: color rojo, ubicación izquierda.

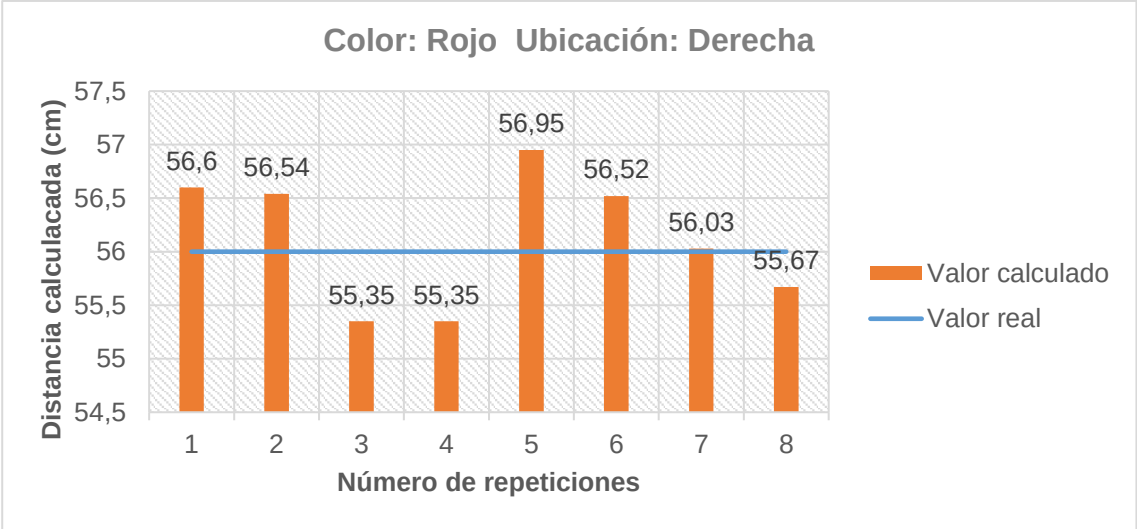

Figura 86. Resultados obtenidos prueba 2: color rojo, ubicación derecha.

A continuación, en la Tabla 9 se puede observar el error calculado para cada uno de las repeticiones realizadas a cada tratamiento del color rojo.

Tabla 9

Error porcentual en cálculo distancia para objetos de color rojo.

		Ubicación		
		Centro	**Izquierda**	**Derecha**
|Número de experimento	**1**	0,32%	2,21%	1,07%
	2	1,09%	0,83%	0,96%
	3	2,17%	2,38%	1,16%
	4	4,84%	1,05%	1,16%
	5	1,17%	0,78%	1,70%
	6	2,87%	2,51%	0,93%
	7	4,87%	3,43%	0,05%
	8	2,21%	2,58%	0,59%

En este caso, los porcentajes de error de todos los experimentos realizados en cada tratamiento son menores al 5%. A diferencia con el caso anterior, la configuración de las características de la cámara beneficiaron al análisis y a la detección del color rojo, mas que con los otros 2 colores utilizados para realizar las pruebas.

- ***Color verde***

En la Figura 87 se pueden observar los resultados obtenidos para los objetos de color rojo ubicados en la parte central del plano, para la ubicación en izquierda se muestra en la Figura 88 y del lado derecho en la Figura 89.

A continuación, la Tabla 10 contiene el error porcentual calculado para cada una de las repeticiones realizadas a cada tratamiento del color verde.

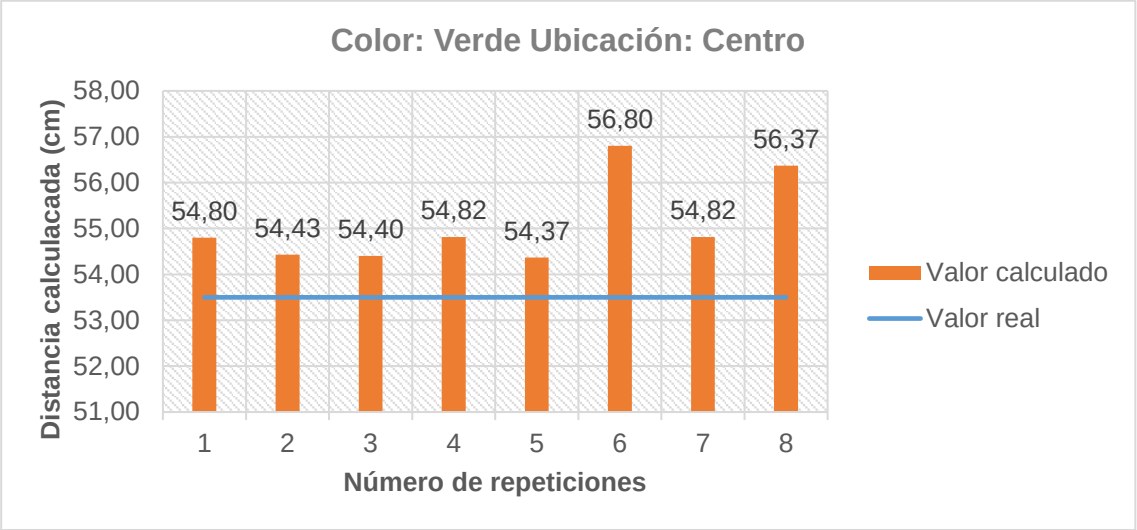

Figura 87. Resultados obtenidos prueba 2: color verde, ubicación centro.

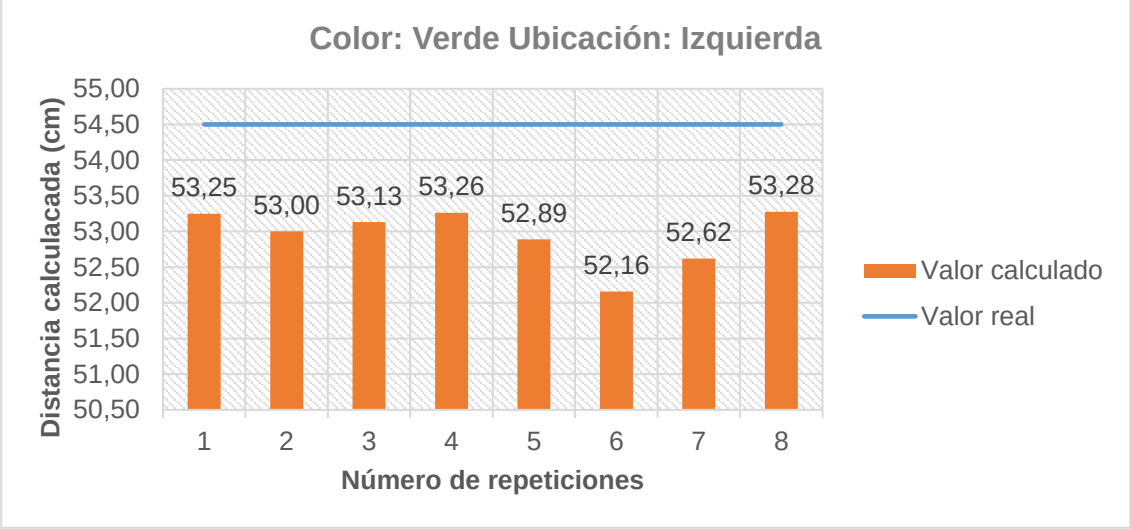

Figura 88. Resultados obtenidos prueba 2: color verde, ubicación izquierda.

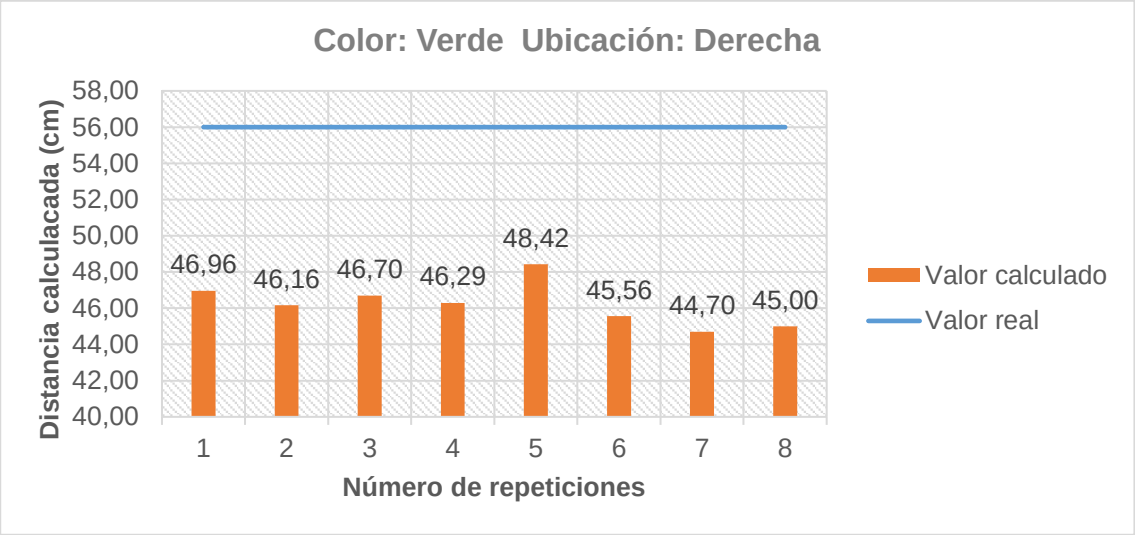

Figura 89. Resultados obtenidos prueba 2: color verde, ubicación derecha.

Tabla 10

Error porcentual en cálculo distancia para objetos de color verde.

		Ubicación		
		Centro	**Izquierda**	**Derecha**
Número de experimento	**1**	2,43%	2,29%	16,14%
	2	1,74%	2,75%	17,57%
	3	1,68%	2,51%	16,61%
	4	2,46%	2,28%	17,34%
	5	1,63%	2,95%	13,54%
	6	6,17%	4,29%	18,64%
	7	2,46%	3,45%	20,18%
	8	5,36%	2,24%	19,64%

Para este caso, los resultados oscilan entre el 1,63% y 20,18%. Los valores altos de error se generan debido a que la iluminación implementada no era completamente uniforme y en sectores del área de trabajo se generaban sombras que afectaban con el análisis de resultados y la lectura de los objetos en el medio. Además, el código RGB

utilizado para el color verde requirió de un umbral de tolerancia alto, lo que permitía que los pixeles alrededor de la imagen de interés también sean considerados como parte del objeto y por ende, si el objeto es mayor, la distancia disminuye aumentando el error en el experimento.

4.2.2 Prueba 2

Para cada tratamiento se realizaron 8 repeticiones, a continuación se muestran los resultados obtenidos para cada color.

- ***Color azul***

En la Tabla 11 se muestran los resultados obtenidos en el cálculo del error relativo entre la posición calculada en el programa y la posición real del objeto de interés con respecto al robot, en la Figura 90 se muestran los resultados del análisis cualitativo.

Tabla 11

Error relativo en cálculo de trayectoria para objetos de color azul.

	Ubicación								
	Centro			**Izquierda**			**Derecha**		
# Exp.	**x**	**y**	**Resultado**	**x**	**y**	**Resultado**	**x**	**y**	**Resultado**
1	-0,92	-0,94	Admitido	-1,72	-0,50	Admitido	-7,18	1,34	Reajuste
2	-0,14	-1,70	Admitido	-1,75	-0,53	Admitido	-3,00	-2,32	Admitido
3	-1,96	-2,03	Admitido	-1,85	-0,40	Admitido	-6,08	-0,04	Reajuste
4	-2,40	-1,07	Admitido	-1,30	-1,10	Admitido	-7,49	-0,60	Reajuste
5	-2,03	-2,72	Admitido	-1,82	-0,82	Admitido	-5,89	-0,02	Reajuste
6	-2,60	-2,10	Admitido	-2,93	3,36	Admitido	-6,56	0,89	Reajuste
7	1,19	-1,50	Admitido	-2,66	-0,20	Admitido	-8,39	0,15	Reajuste
8	-2,00	-0,54	Admitido	-2,81	-0,96	Admitido	-6,13	0,33	Reajuste

Total reajustes para el color azul = 7

Total admitidos para color azul = 17

Total perdidos para el color azul = 0

Total colisiones para el color azul = 0

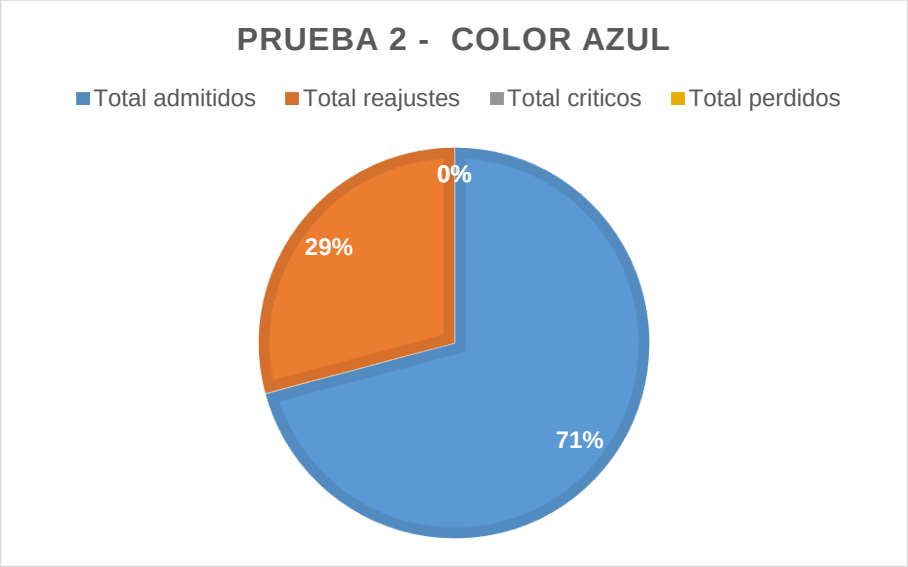

***Figura 90*.** Resultados obtenidos en la prueba 3 - color azul.

El 23% de casos requirieron de un reajuste para poder acercarse al objeto a una distancia que le permita tomarlo del pedestal para proceder con la clasificación, es importante recalcar que un escenario de reajuste no es un error y por lo tanto, el valor alto no implica que los resultados obtenidos no son aceptables.

- **Color Rojo**

En la Tabla 12 se muestran los resultados obtenidos en el cálculo del error relativo entre la posición calculada en el programa y la posición real del objeto de interés con respecto al robot, en la Figura 91 se muestran los resultados del análisis cualitativo.

Tabla 12

Error relativo en cálculo de trayectoria para objetos de color rojo.

	Ubicación								
	Centro			Izquierda			Derecha		
# Exp.	**x**	**y**	**Resultado**	**x**	**y**	**Resultado**	**x**	**y**	**Resultado**
1	-0,9	-0,43	Admitido	0	0,5	Admitido	-1,42	-0,37	Admitido
2	-0,44	0,1	Admitido	-1,02	2,5	Admitido	-1,67	-0,55	Admitido
3	-2,92	-2,88	Admitido	-2,02	2,61	Admitido	-2,67	0,01	Admitido
4	-1,2	-1,05	Admitido	0,36	-1,16	Admitido	-2,67	0,01	Admitido
5	-3	-3,83	Reajuste	0,16	0,34	Admitido	-1,32	-0,7	Admitido
6	-2,1	0,64	Admitido	-2,44	-5,39	Admitido	-1,7	-0,58	Admitido
7	-4,178	-0,21	Reajuste	-1,4	-3,1	Admitido	-2,1	-0,29	Admitido
8	-3,3	-2,19	Reajuste	-1,44	-2,93	Admitido	-2,4	-0,13	Admitido

Total reajustes para el color rojo = 3

Total admitidos para color rojo = 21

Total perdidos para el color rojo = 0

Total colisiones para el color rojo = 0

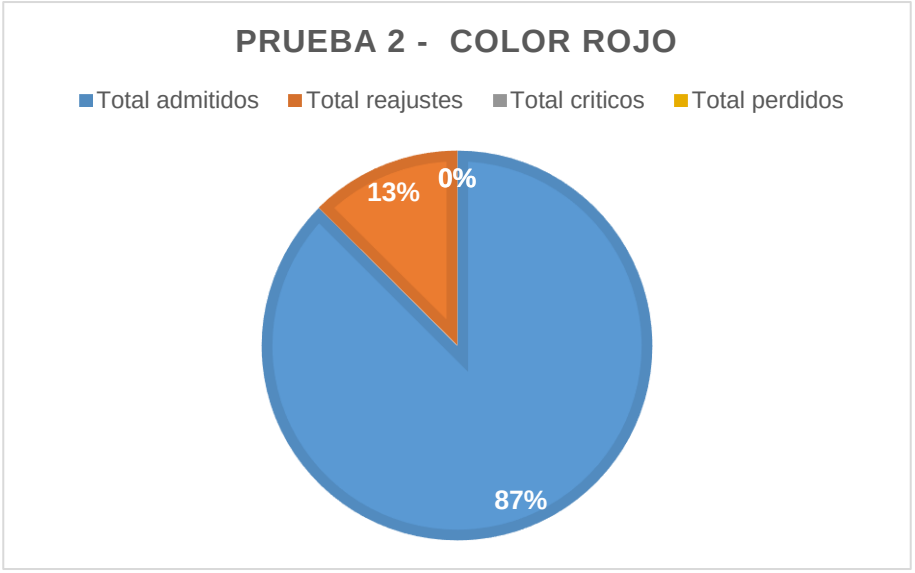

Figura 91. Resultados obtenidos en la prueba 3 - color rojo.

El 13% de casos requirieron de un reajuste para poder acercarse al objeto a una distancia que le permita tomarlo del pedestal para proceder con la clasificación, es importante recalcar que un escenario de reajuste no es un error y por lo tanto, el valor alto no implica que los resultados obtenidos no son aceptables.

- **Color Verde**

En la Tabla 13 se muestran los resultados obtenidos en el cálculo del error relativo entre la posición calculada en el programa y la posición real del objeto de interés con respecto al robot, en la Figura 92 se muestran los resultados del análisis cualitativo.

Tabla 13

Error relativo en cálculo de trayectoria para objetos de color verde.

	Ubicación								
	Centro			Izquierda			Derecha		
# Exp.	**x**	**y**	**Resultado**	**x**	**y**	**Resultado**	**x**	**y**	**Resultado**
1	-0,6	-0,27	Admitido	-1,43	0,11	Admitido	-8,22	7,25	Reajuste
2	-0,96	-0,26	Admitido	-2,21	0,22	Admitido	-9,13	7,79	Reajuste
3	-0,75	-0,3	Admitido	-1,21	0,14	Admitido	-8,65	7,52	Reajuste
4	-0,6	-0,27	Admitido	-1,72	0,11	Admitido	-9,1	7,46	Reajuste
5	-1	-0,21	Admitido	-2,85	0,3	Admitido	-7,05	7,123	Reajuste
6	1,329	-0,28	Admitido	-1,92	0,5	Admitido	-9,734	7,7	Reajuste
7	0,4	-0,27	Admitido	-1,17	0,34	Admitido	-11,84	4,42	Reajuste
8	0,93	-0,32	Admitido	-1,81	0,14	Admitido	-10,31	7,81	Reajuste

Total reajustes para el color verde = 8

Total admitidos para color verde = 16

Total perdidos para el color verde = 0

Total colisiones para el color verde = 0

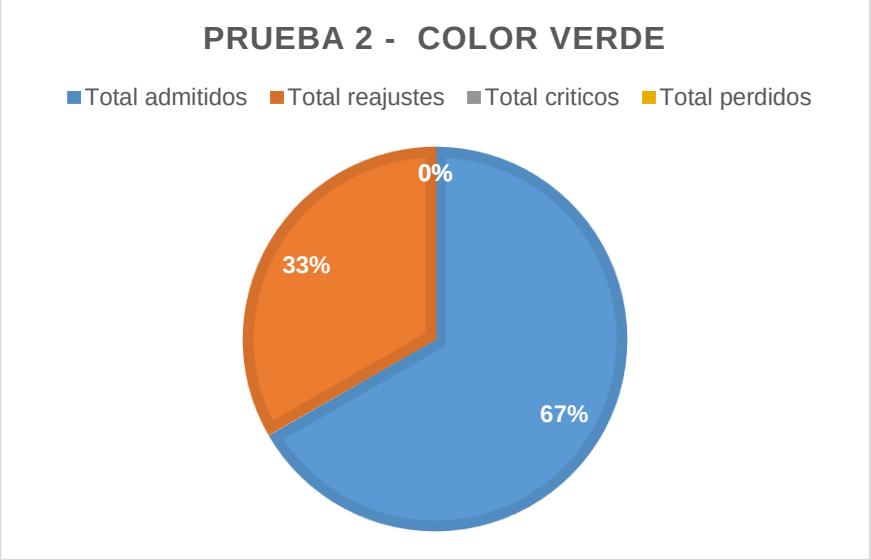

Figura 92. Resultados obtenidos en la prueba 3 - color verde.

El 33% de casos requirieron de un reajuste para poder acercarse al objeto a una distancia que le permita tomarlo del pedestal para proceder con la clasificación, es importante recalcar que un escenario de reajuste no es un error y por lo tanto, el valor alto no implica que los resultados obtenidos no son aceptables.

Para este caso el error en reajuste es alto lo cual concuerda con los resultados obtenidos en la prueba anterior, puesto que si el cálculo en la distancia tiene porcentajes de error altos, el cálculo de la posición relativa del objeto también presentará errores, sin embargo, si el tratamiento requiere de un reajuste, los resultados siguen siendo favorables en términos cualitativos.

4.2.3 Prueba 3

Se realizaron 3 experimentos con el sistema de clasificación de objetos, cada experimento consta de 2 colores de interés, los cuales se especificarán a continuación.

- ***Experimento 1***

La Figura 93 muestra el esquema del experimento 1, los colores de interés definidos son: rojo y azul, lo que significa que los objetos de color verde no deben ser detectados ni tomados en cuenta para el proceso de clasificación. Los resultados obtenidos se pueden observar en la Tabla 14.

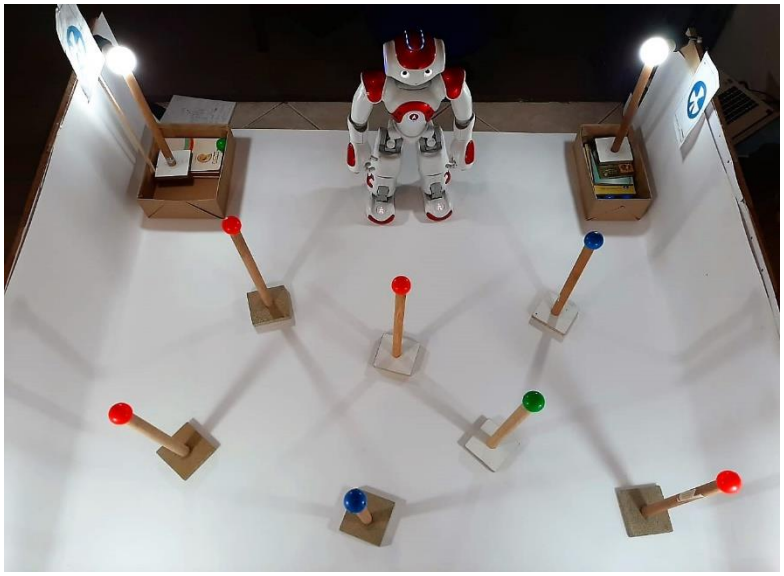

Figura 93. Esquema del experimento 1 - Prueba 3

Tabla 14

Resultados obtenidos del experimento 1 realizado en la prueba 3.

	Color	**Resultado final**	**Observaciones**
Objeto 1	Azul	Clasificado correctamente	-
Objeto 2	Rojo	Clasificado correctamente	Requirió reajuste
Objeto 3	Rojo	Clasificado correctamente	-
Objeto 4	Verde	No detectado	N/A
Objeto 5	Azul	No detectado	
Objeto 6	Rojo	Clasificado correctamente	-
Objeto 7	Rojo	Clasificado correctamente	-

El resumen de los datos obtenidos se muestra en la Figura 94.

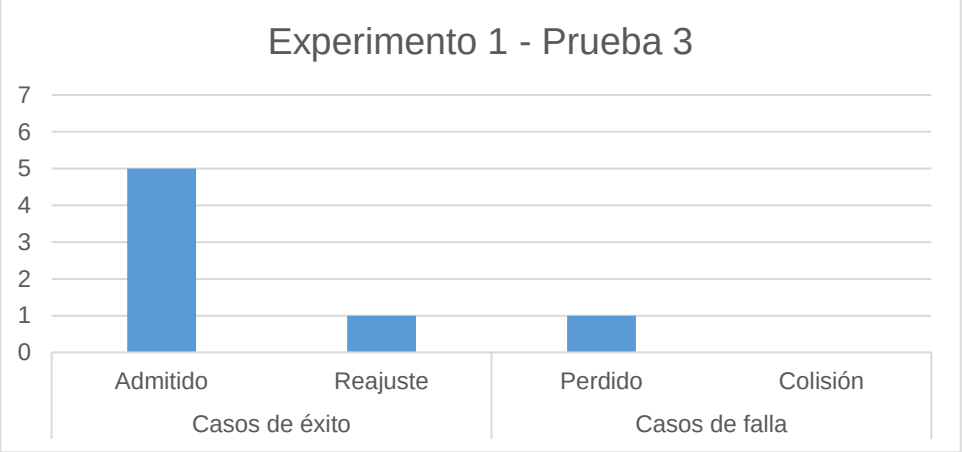

Figura 94. Resultados obtenidos del experimento 1 – Prueba3

- ***Experimento 2***

La Figura 95 muestra el esquema del experimento 2, los colores de interés definidos son: rojo y azul, lo que significa que los objetos de color verde no deben ser detectados ni tomados en cuenta para el proceso de clasificación. Los resultados obtenidos se pueden observar en la Tabla 15

Tabla 15

Resultados obtenidos del experimento 2 realizado en la prueba 3.

	Color	**Resultado final**	**Observaciones**
Objeto 1	Rojo	Clasificado correctamente	Requirió reajuste
Objeto 2	Azul	Clasificado correctamente	Requirió reajuste
Objeto 3	Azul	Clasificado correctamente	Requirió reajuste
Objeto 4	Rojo	Clasificado correctamente	-
Objeto 5	Rojo	Clasificado correctamente	-
Objeto 6	Verde	No detectado	N/A
Objeto 7	Rojo	Clasificado correctamente	Requirió reajuste

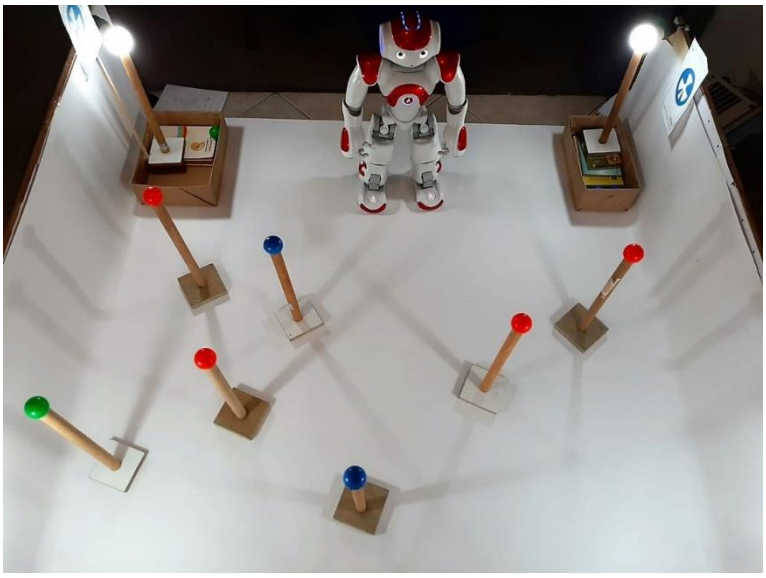

Figura 95. Esquema del experimento 2 – Prueba 3

El resumen de los datos obtenidos se muestra en la Figura 96.

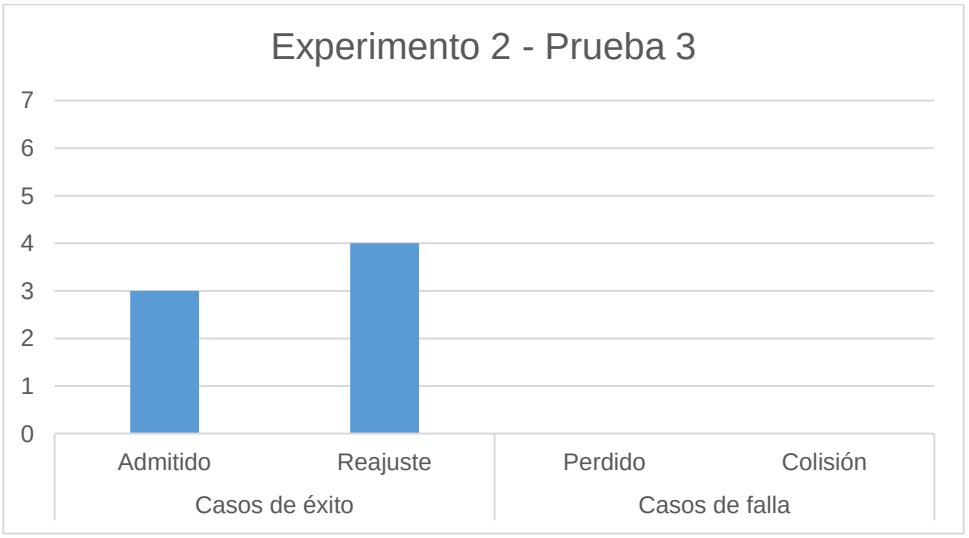

Figura 96. Resultados obtenidos del experimento 2 – Prueba3

- ***Experimento 3***

La Figura 97 muestra el esquema del experimento 2, los colores de interés definidos son: rojo y azul, lo que significa que los objetos de color verde no deben ser detectados ni tomados en cuenta para el proceso de clasificación. Los resultados obtenidos se pueden observar en la Tabla 16.

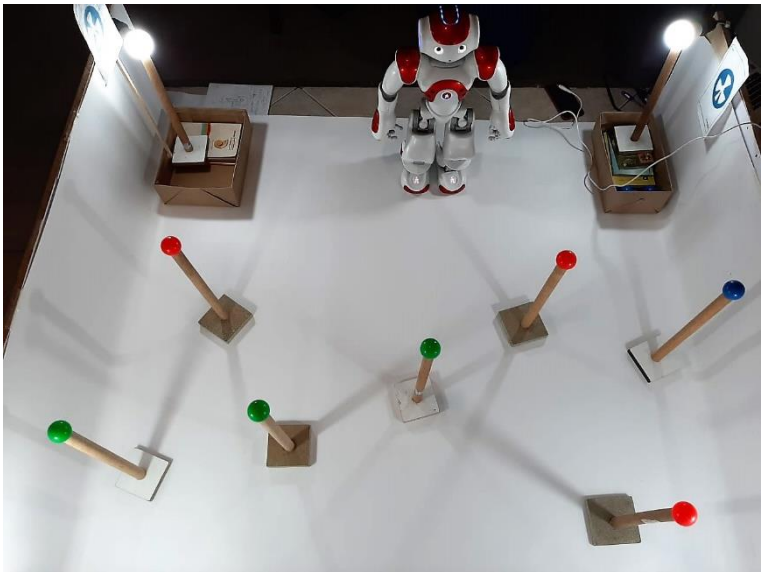

Figura 97. Esquema del experimento 3 – Prueba 3

Tabla 16

Resultados obtenidos del experimento 3 realizado en la prueba 3.

	Color	**Resultado final**	**Observaciones**
Objeto 1	Azul	No detectado	N/A
Objeto 2	Rojo	Clasificado correctamente	Requirió reajuste
Objeto 3	Rojo	Clasificado correctamente	-
Objeto 4	Verde	Clasificado correctamente	Requirió reajuste
Objeto 5	Verde	Clasificado correctamente	Requirió reajuste
Objeto 6	Rojo	Clasificado correctamente	-
Objeto 7	Verde	No detectado	-

El resumen de los datos obtenidos se muestra en la Figura 98.

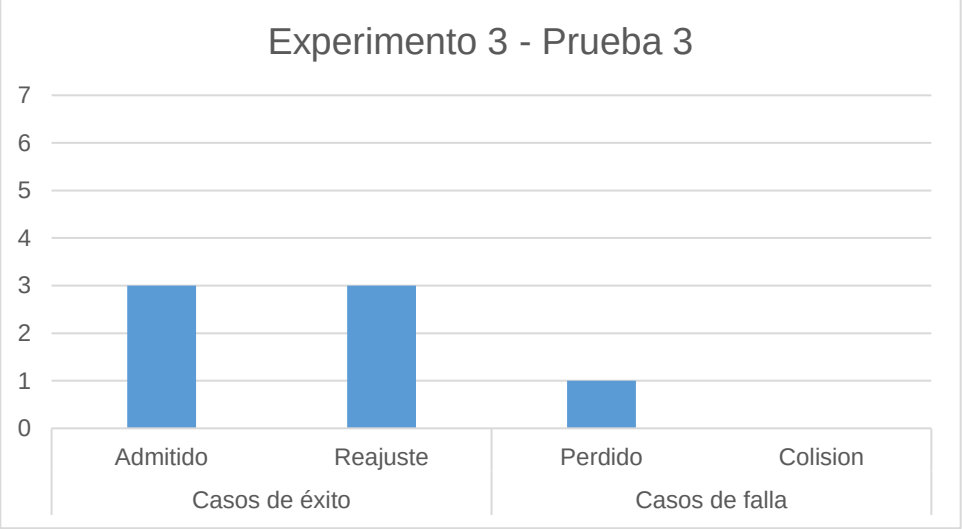

Figura 98. Resultados obtenidos del experimento 3 – Prueba3

CAPÍTULO V

CONCLUSIONES Y RECOMENDACIONES

5.1 Conclusiones

La programación de las etapas del sistema de clasificación de objetos se realizó mediante el lenguaje de programación Python y fue implementado en el software Choregraphe utilizando los recursos del robot NAO accediendo por medio de NAOqi framework.

Para la etapa de reconocimiento de objetos, se crearon 5 bloques de código, los cuales permiten la búsqueda, el reconocimiento de objetos en base a su color y el análisis de los resultados obtenidos para extracción de información clave que permita continuar con el proceso. En esta fase, se extraen los datos de la posición y el diámetro del objeto de interés, el cual se obtuvo en pixeles, todo ello con respecto a un plano en 2 dimensiones.

Se utilizaron técnicas básicas de visión artificial para calcular la distancia entre un objeto y una cámara, la cual fue utilizada durante la segunda etapa del sistema. Posteriormente, se utilizaron funciones propias del robot para obtener los ángulos relativos del objeto con respecto al plano.

Se desarrolló un algoritmo que permite calcular la posición relativa del objeto con respecto al robot, para ello se utilizaron los datos obtenidos en el cálculo de la distancia y los ángulos relativos de cámara.

Para la etapa de seguimiento de objetos, se crearon 2 bloques de código los cuales realizan el cálculo de la distancia existente entre la cámara utilizada y el objeto de interés detectado en la etapa previa. Además, se realiza el cálculo de la posición relativa del objeto de interés y el robot utilizando el algoritmo desarrollado, para proceder con el desplazamiento del robot hacia una posición que le permita continuar con la siguiente fase.

Para la etapa de manipulación de objetos, se crearon 5 bloques de código, en donde se realiza un análisis de la posición actual del objeto de interés, si este no se encuentra dentro del alcance del robot, se realiza un reajuste de datos para que el humanoide se desplace de nuevo hasta que la distancia entre el objeto y el robot este dentro de un umbral de tolerancia definido en base a las características físicas del humanoide, esto para proceder con la toma del objeto con el actuador del brazo izquierdo.

Se realizaron 3 pruebas utilizando el método de diseño de experimentos sencillos, la primera prueba mostró el escenario de error obtenido al calcular la distancia entre el objeto y la cámara del robot. La segunda prueba analizó el error entre el cálculo de la posición relativa del objeto y el dato real, además se realizó un análisis cualitativo acerca de los resultados obtenidos, en donde se definió el número de casos de éxito y de falla. Finalmente en la tercera prueba se implementaron 3 experimentos en un esquema de clasificación completo, cada uno de ellos con 7 objetos ubicados en el área de trabajo localizados de forma aleatoria, los cuales debían ser clasificados , en esta prueba se analizaron el número de casos de éxito y de falla.

De acuerdo con los resultados obtenidos, el promedio de error en el cálculo de la distancia para el caso el color azul, varía entre 0,15 % y 9,50 %, siendo los valores más altos de error los obtenidos cando los objetos se encuentran en la parte derecha del robot. Para el color rojo, el error varía entre 0,32% y 4,87%. Y para el color verde, el error oscila entre el 1,74% y 20,50%. El motivo por el cual se dan valores altos de error es debido a que la limitación de los códigos RGB, y en especial del umbral de tolerancia del color, deben ser minuciosos y excluir posibles pixeles que se generen alrededor del robot debido a las configuraciones de cámara.

Los resultados obtenidos en la prueba 3 muestran que el sistema de clasificación de objetos desarrollado en el presente proyecto, cumple con su objetivo. Es importante recalcar que un escenario de reajuste de datos, no implica un error ni una falla y es por ello que está considerado dentro del grupo de casos de éxito.

5.2 Recomendaciones

Se recomienda utilizar el robot para experimentos con visión artificial en un medio con iluminación uniforme, estable y controlada, debido a que pequeñas variaciones en cuanto a este factor puede generar datos erróneos.

Para aplicaciones en las cuales el robot debe movilizarse, es importante considerar que la rugosidad del piso interferirá directamente en el proceso de caminata del robot y podría generar errores en el resultado final. Además, es importante limpiar la superficie

sobre la cual el robot va a desplazarse, esto para prevenir posibles accidentes o caídas del robot, para evitar errores así como para prevenir el desgaste del humanoide.

El robot no debe ser utilizado durante largos periodos de tiempo puesto que los motores de las articulaciones tienden a sobrecalentarse, causando que los procesos se vean afectados, principalmente de caminata puesto que se altera su funcionamiento y produce errores en el resultado final.

Para determinar los códigos RGB de los colores con los cuales se desee trabajar, es recomendable obtenerlos bajo las mismas condiciones de iluminación en las que funcionará el robot, para evitar futuras incongruencias en cuanto a la detección del color causando que se omitan objetos de interés presentes en el medio.

Se recomienda verificar los recursos que requieren cada módulo del NAOqi y sus funciones correspondientes, debido a que cada recurso no puede ser utilizado por distintos llamados al mismo tiempo, y si esto sucediera podrían enviarse comandos contradictorios al robot y causar accidentes.

La caminata del robot humanoide NAO no es precisa ni exacta, por lo cual se recomienda realizar las correcciones respectivas de acuerdo a las necesidades de las diversas aplicaciones que pueden realizarse.

5.3 Trabajos futuros

La aplicación creada en el presente proyecto de titulación es escalable por lo tanto es posible aumentar las características del sistema de clasificación, así como mejorar los resultados que se muestran en este documento.

De acuerdo con los resultados presentados en este documento, la tasa de objetos que requirieron un reajuste para ser alcanzados por el robot es alta, por lo cual se plantea una mejora en la etapa de seguimiento a objetos, de tal forma que el cálculo de la distancia y de la posición del objeto con respecto al robot se realice de forma continua y se corrijan los errores durante el proceso de caminata del robot.

Por otro lado, se plantea recrear el sistema de clasificación de objetos aumentando el número de colores que pueden clasificarse, y por ende aumentando el área de trabajo del robot.

Finalmente, se plantea la implementación de este proyecto utilizando las distintas resoluciones de cámara que ofrece el robot NAO para realizar una comparación de resultados.

BIBLIOGRAFÍA

(CNP), C. N. (2017). *Plan Nacional de Desarrollo 2017-2021. Toda una Vida.* Quito : Secretaría Nacional de Planifiación y Desarrollo, Senplades.

Abad Sacoto, K., Sánchez Delgado, M., Crespo Cedeño, J., & Alvarado Chang, J. (2017). Sistemas de reconocimiento en la robótica social.

AliveRobotics. (s.f.). *NAO, la mejor plataforma para educación.* Obtenido de ROBOTRONICA: https://aliverobots.com/nao-educacion/

BOXBYTE. (22 de Abril de 2010). *AILA: Una elegante robot que reconoce objetos para su manipulación y clasificación*. Obtenido de FayweWayer: https://www.fayerwayer.com/2010/04/aila-una-robot-que-reconoce-objetos-para-su-manipulacion-y-clasificacion//

Brady, M., & Paul, R. (1984). *Robotics Research: The First International Symposium.* Massachusetts: Cambridge.

Castro Díaz, M. (2016). *Manipulación y posicionamiento de objetos desconocidos por parte de un robot autónomo.* Santiago de Chile: Universidad de Chile.

Castro-Gonzáles, Á., Martín, F., Castillo, J., & Salichis, M. (28 de Noviembre de 2017). *Evolución de la robótica social y nuevas tendencias.* Obtenido de Research gate: https://www.researchgate.net/publication/321332806

Cecilia Laschi, P. D. (s.f.). *Grasping and Manipulation in Humanoid Robotics.* Pisa: Scuola Superiore Sant'Anna.

Chuet-Missé, J. P. (26 de Marzo de 2017). *Los robots echarán a la mitad de los trabajadores*. Obtenido de Economía Digital:

https://www.economiadigital.es/tecnologia-y-tendencias/robots-amenaza-trabajo_402138_102.html

Díaz, I., & Chaves, P. (2017). *Desarrollo de actividades basadas en robótica social para pacientes con TEA.* Madrid, España: Universidad Carlos III de Madrid.

Díaz, M., Amara, A., Casacuberta, J., & Angulo, C. (2009). *Robots sociales en la escuela.* Catalunya, España: Universitat Politècnica de Catalunya.

Dirección de Comunicación SENPLADES. (11 de Noviembre de 2017). *El Plan Nacional de Desarrollo 2017-2021 "Toda una Vida" se presentó en Riobamba*. (Secretaría Nacional de Planificación y Desarrollo) Obtenido de http://www.planificacion.gob.ec/el-plan-nacional-de-desarrollo-2017-2021-toda-una-vida-se-presento-en-riobamba/

Dsouza, R. (2016). Humanoid Robots – Past, Present and the Future. *European Journal of Advances in Engineering and Technology*, 8-15.

EBS Robótica. (3 de Agosto de 2018). *Clasificación de los robots según su función*. Obtenido de Esneca Business School: http://www.esneca.com/blog/clasificacion-de-los-robots-segun-su-funcion/

Egbert van der Wal. (2012). *Object Grasping with the NAO.* University of Groningen.

Emara, T. (3 de Noviembre de 2018). *Real-time Distance Measurement Using Single Image*. Obtenido de EmaraIC: http://emaraic.com/blog/distance-measurement

Entrevista a Sebastián Rojas. (2016). Obtenido de SOLACYT: http://solacyt.org/entrevista-sebastian-rojas/

Fong, T., Nourbakhsh, I., & Kerstin Dautenhahn. (2003). A survey of socially interactive robots. *Robotics and autonomous systems*, 143-166.

Grundy, T. (9 de Marzo de 2018). *Interview: HKFP speaks to Sophia, Hong Kong's own social humanoid A.I. robot*. Obtenido de Hong Kong Free Press: https://www.hongkongfp.com/2018/03/09/interview-hkfp-speaks-sophia-hong-kongs-social-humanoid-i-robot/

Hammad, A., Xiaolan, Y., & Muhammad, M. (2017). Humanoid Robots Object Grasping and Manipulation Learning by demonstration. *3rd Interntional Conference on Control, Automation and Robotics.*

Hernández , A., Gómez, C., Crespo , J., & Barber , R. (2016). Object Classification in Natural Environments for Mobile Robot Navigation. *IEEE International Conference on Autonomous Robot Systems and Competitions.* Madrid.

Keller, J. R. (4 de Octubre de 2016). *Two or three things you may not know about the history of automation*. Obtenido de nesta: https://www.nesta.org.uk/blog/two-or-three-things-you-may-not-know-about-the-history-of-automation/

Laschi, C., Dario, P., & Chiara, M. (s.f.). *Grasping and manipulation in humanoid robotics.* Viareggio, Italia: Centro INAIL RTR.

Lopez-Caudana, E., Quiroz, O., Rodríguez, A., Yépez, L., & Ibarra, D. (2017). Classification of materials by acoustic signal processing in real time for NAO robots. *International Journal of Advanced Robotic Systems*, 1-10.

Martell Chávez, F. (5 de Marzo de 2018). *Industria 4.0: Automatización, Robótica y Visión Artificial*. Obtenido de Visión Industrial: http://www.visionindustrial.com.mx/industria/tecnologia/industria-4-0-automatizacion-robotica-y-vision-artificial

Martin, J. (16 de Noviembre de 2016). *La Industria 4.0 será el final de la industria*. Obtenido de Futurizable: https://futurizable.com/industria/

Merrill, D. (13 de Junio de 2018). *Robots en los CeDis: ¿El empleado perfecto?* Obtenido de inBound Logistics LATAM: http://www.il-latam.com/blog/global-logistics/robots-en-los-cedis-el-empleado-perfecto

Monzón, J. (31 de Marzo de 2014). *Un brazo robot que permite mover y clasificar objetos*. (Argentina Investiga) Obtenido de http://argentinainvestiga.edu.ar/noticia.php?titulo=un_brazo_robot_que_permite_mover_y_clasificar_objetos&id=2057

NAO Construction. (s.f.). Obtenido de Aldebaran Robotics: http://doc.aldebaran.com/2-1/family/robots/dimensions_robot.html

NAO Los robots del futuro son ya una realidad. (s.f.). Obtenido de AliveRobots by Robotrónica: https://aliverobots.com/nao/

Panfir, A. N., Boboc, R. G., & Mogan, G. (2013). NAO Robots collaboration for object manipulation. *Applied Mechanics and Materials, 332*, 218-223.

Pelegrí, J. (10 de Enero de 2019). *La Cuarta Revolución Industrial: Cobots y Automatización*. Obtenido de Universal Robots: https://blog.universal-robots.com/es/revolucion-industrial-cobots

robot-man. (7 de Agosto de 2018). *Halodi Eve Humanoid Robot for the Kitchen*. Obtenido de Robotic Gizmos: http://www.roboticgizmos.com/halodi-eve-humanoid-robot-kitchen/

ROSAS-ARIAS, L., VALLEJO-MERAZ, J. d., PÉREZ-BAILÓN, W., & ROJAS-CID, J. D. (2017). Robot clasificador de objetos de color utilizando técnicas de filtrado RGB. *Revista de Prototipos Tecnológicos, 3*(10), 50-59.

Salichs, M., Malfaz, M., & Gorostiza, J. (2010). Toma de Decisiones en Robótica. *Revista iberoamericana de automática e Informática Industrial, 7*(4), 5-16.

Sánchez, Á. (s.f.). *Sensores y actuadores.* Obtenido de Universidad Pontificia Comillas : https://www.iit.comillas.edu/alvaro/teaching/Clases/Robots/teoria/Sensores%20y%20actuadores.pdf

Senad, C. (19 de Enero de 2009). *Humanoid robots.* Obtenido de Uni-Hamburg: https://tams-www.informatik.uni-hamburg.de/lehre/2008ws/seminar/ra/PDF/humanoid-robots.pdf

Sgorbissa, A., & Verda, D. (21 de Junio de 2013). *Structure-based object representation and classification in mobile robotics through a Microsoft Kinect, Robotics and Autonomous Systems.* Obtenido de http://dx.doi.org/10.1016/j.robot.2013.06.006

Sinapov, J. (29 de Mayo de 2014). *Object Category Recognition by a Humanoid Robot Using Behavior-Grounded Relational Learning.* Obtenido de IEEE International Conference on Robotics and Automation : https://www.researchgate.net/publication/224252794

Sobrado Malpartida, E. (2003). *Sistema de visión artificial para el reconocimiento y manipulación de objetos utilizando un brazo robot .* Lima: Pontificia Universidad Católica del Perú.

Tawil, Y. (26 de Junio de 2017). *An Introduction to Robot Operating System (ROS)*. Obtenido de All About Circuits: https://www.allaboutcircuits.com/technical-articles/an-introduction-to-robot-operating-system-ros/

teleSUR - ap - HR. (27 de Octubre de 2017). *Compañía china lanza robot para atender ancianos*. Obtenido de teleSUR: https://www.telesurtv.net/news/Compania-china-lanza-robot-para-atender-a-ancianos-20171027-0036.html

Vanderkley, R. (21 de Agosto de 2015). *Robots and Communication, with Eleanor Sandry*. Obtenido de RoboHub: https://robohub.org/robots-podcast-robots-and-communication/

Watkins, D. (Marzo de 2018). *Quick Introduction to ROS.* Obtenido de David Watkins - Robotics Ph.D. Student: https://davidwatkinsvalls.com/files/2018_spring_ros_tutorial.pdf

Who is NAO? (s.f.). Obtenido de SoftBank Robotics: https://www.softbankrobotics.com/emea/en/robots/nao

Wohlkinger, W., & Vincze, M. (2010). 3D Object Classification for Mobile Robots in Home-Environments Using Web-Data. *19th International Workshop on Robotics in Alpe-Adria-Danube Region.* Budapest.

Yánez Arancibia, J. M. (2013). *Detección robusta de objetos en robots humanoides.* Santiago de Chile: Universidad de Chile.

Zurita Pérez, J. L. (2014). *Diseño e implementación de una máquina automática clasificadora de objetos según su color detectados mediante un sensor de color y clasificados por un brazo robótico.* Latacunga: ESPE.