Instituto Tecnológico de Costa Rica

Carrera Ingeniería Mecatrónica

Diseño de entorno robótico como herramienta para el desarrollo de arquitecturas cognitivas

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

Emanuel Fallas Hernández

Cartago, 22 de junio 2021

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

En los casos en que he utilizado bibliografía, he procedido a indicar las fuentes mediante las respectivas citas bibliográficas.

En consecuencia, asumo la responsabilidad total por el trabajo de graduación realizado y por el contenido del correspondiente informe final.

Cartago, 22 de junio 2021

efalles

Emanuel Fallas Hernández
Céd: 305120164

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

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

Estudiante: Emanuel Fallas Hernández

Proyecto: Diseño de entorno robótico como herramienta para el desarrollo de arquitecturas cognitivas.

JUAN CARLOS BRENES TORRES (FIRMA)

Firmado digitalmente por JUAN CARLOS BRENES TORRES (FIRMA)
Fecha: 2021.06.25 12:09:37 -06'00'

MSc. -Ing. Juan Carlos Brenes Torres

Asesor

Cartago, 22 de junio 2021

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

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

Estudiante: Emanuel Fallas Hernández

Proyecto: Diseño de entorno robótico como herramienta para el desarrollo de arquitecturas cognitivas.

Miembros del jurado evaluador

TEC | Tecnológico de Costa Rica
ANA GABRIELA ORTIZ LEON (FIRMA)
2021.06.25 16:04:25 -06'00'

Dra. -Ing. Gabriela Ortiz León

Jurado

ROGER STUART MELENDEZ POLTRONIERI (FIRMA)
Firmado digitalmente por ROGER STUART MELENDEZ POLTRONIERI (FIRMA)
Fecha: 2021.06.25 12:03:15 -06'00'

Ing. Roger Meléndez Poltronieri

Jurado

ANA MARIA MURILLO MORGAN (FIRMA)
Firmado digitalmente por ANA MARIA MURILLO MORGAN (FIRMA)
Fecha: 2021.06.25 11:49:29 -06'00'

Ing. Ana María Murillo Morgan

Jurado

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

Cartago, 22 de junio 2021

Resumen

Se presenta el diseño e implementación de una herramienta para el desarrollo de arquitecturas cognitivas en el Laboratorio de Inteligencia Artificial para las Ciencias Naturales del Instituto Tecnológico de Costa Rica. El problema solucionado fue la falta de capacidad experimental dentro del laboratorio que impedía el establecimiento de una línea de investigación en robótica cognitiva. El producto derivado de este proyecto es un entorno robótico simulado basado en Gazebo y ROS titulado *Open-Source Cognitive Applied Robot* (OSCAR). Además, se diseñó una interfaz que permite ejecutar experimentos de aprendizaje utilizando la memoria a largo plazo de la arquitectura cognitiva *Multilevel Darwinist Brain*. A través de una serie de experimentos se comprobaron las especificaciones más importantes de la herramienta para validar su funcionamiento.

Palabras clave: Aprendizaje de tareas, arquitectura cognitiva, memoria a largo plazo, percepción visual, robótica cognitiva

Abstract

The design and implementation of a framework for the development of cognitive architectures in the Laboratory of Artificial Intelligence for Natural Sciences of the Costa Rican Insitute of Technology is presented. The problem solved was the lack of experimental capacity in the laboratory, that halted the establishment of a research line in cognitive robotics. The product derived from this project is a simulated robotic enviroment based on Gazebo and ROS named *Open-Source Cognitive Applied Robot* (OSCAR). Furthermore, an interface that allows the execution of learning experiments using the long-term memory of the Multilevel Darwinist Brain cognitive architecture was designed. Via a series of experiments, the critical specifications of the framework were validated.

Keywords: Cognitive architecture, cognitive robotics, long-term memory, task learning, visual perception

AGRADECIMIENTOS

Quiero comenzar agradeciendo al Dr. Juan Luis Crespo Mariño por abrirme las puertas del LIANA y permitirme realizar este proyecto. También por la gran cantidad de conocimiento que me transmitió durante los últimos cursos de la carrera.

Al Ing. Ronald Loaiza Baldares por el gran apoyo dado durante todo mi tiempo en LIANA. Por el gran interés mostrado en el éxito de este proyecto, desde que era una idea vaga hasta su finalización. Y especialmente por su amistad.

Al Ing. Juan Carlos Brenes Torres por su gran trabajo asesorando este proyecto. Sus aportes fueron clave para mejorar la calidad del proyecto.

Al Dr. José Antonio Becerra Permuy por facilitarme el acceso al MDB y explicarme su funcionamiento. Su ayuda fue muy importante para el éxito de una de las etapas más críticas del proyecto.

A mis compañeros de laboratorio Frederik Orozco Reyes y Sebastián Rodríguez Alfaro por hacer muy llevadero el limitado tiempo que pude trabajar presencialmente en LIANA.

A todos mis amigos y amigas que me acompañaron durante la carrera. Gracias a ustedes el gran esfuerzo de los últimos 5 años se disfrutó al máximo. Me llevo muchísimos recuerdos con ustedes que atesoraré durante mi vida.

Finalizo agradeciendo a mi familia, que me ha dado todo lo que necesito y más. Sin ustedes no hubiera llegado a este momento.

LISTA DE CONTENIDOS

LISTA DE FIGURAS

LISTA DE TABLAS

LISTA DE ACRÓNIMOS

AC Arquitectura Cognitiva

AMADIS Arquitectura Modular de Atención con Distribución de la Información y Sensorización

ANOVA Análisis de Varianza

API *Application Programming Interface*

DOE Diseño de Experimentos

GII Grupo Integrado de Ingeniería

KDL *Kinematics and Dynamics Library*

LIANA Laboratorio de Inteligencia Artificial para las Ciencias Naturales

LTM *Long-Term Memory* (Memoria a largo plazo)

MDB *Multilevel Darwinist Brain*

ODE *Open Dynamics Engine*

OMPL *Open Motion Planning Library*

OSCAR *Open-Source Cognitve Applied Robot*

ROS *Robotic Operating System*

RTTs *Rapidly-exploring Random Trees*

SRK *Skill-Rule-Knowledge* (Habilidad-Regla-Conocimiento)

CAPÍTULO 1

INTRODUCCIÓN

El proyecto que se presenta a continuación fue desarrollado en el *Laboratorio de Inteligencia Artificial para las Ciencias Naturales* (LIANA) del Área Académica de Ingeniería Mecatrónica, en la sede central del Instituto Tecnológico de Costa Rica. Este laboratorio fue fundado en el año 2016 por el Dr. Juan Luis Crespo Mariño con los objetivos de desarrollar novedosos paradigmas de inteligencia computacional basados en el estudio de fenómenos naturales, así como usar paradigmas de computación inteligente a la resolución de problemas propios de las ciencias naturales y otras disciplinas tecnológicas [1]. Las áreas de trabajo del laboratorio se orientan a la investigación biomédica y al estudio de datos astronómicos.

Este proyecto aborda el interés del laboratorio por establecer una línea de investigación en robótica cognitiva. Esta área consiste en el diseño de capacidades motoras, cognitivas y sociales en robots altamente autónomos [2]. El primer acercamiento a esta línea de investigación se da con el desarrollo de la *Arquitectura Modular de Atención con Distribución de la Información y Sensorización* (AMADIS) en la tesis doctoral del Dr. Crespo Mariño [3]. También, se tiene una colaboración

activa con el *Grupo Integrado de Ingeniería* (GII) de la *Universidade da Coruña* en el desarrollo de la arquitectura cognitiva *Multilevel Darwinist Brain* (MDB). El aporte del LIANA se ha dado en el desarrollo de una herramienta de visualización del aprendizaje en la memoria a largo plazo de la arquitectura MDB [4] y en el estudio sobre delimitación perceptual mediante el uso de redes neuronales.

La principal limitación existente dentro del laboratorio -previo a la realización de este proyecto- era la incapacidad de realizar experimentos avanzados e integrados en robótica cognitiva, lo cual impedía que desde LIANA se pueda realizar un aporte relevante a esta área de investigación. La ausencia de instrumentos y herramientas en el laboratorio era una de las principales aristas de la problemática abarcada. Esto se hacía evidente en la falta de un equipo de naturaleza robótica, que es indispensable para realizar investigación de alto nivel en un tema como robótica cognitiva. Una herramienta apta para el estudio de robótica cognitiva no consiste únicamente en un robot, este debe ser adaptado para ofrecer interfaces de sensorización y actuación en un alto nivel. Además, para realizar experimentos en arquitecturas cognitivas deben establecerse métodos de integración para extraer y procesar los datos generados.

Por lo tanto, el problema abordado fue la falta de capacidad experimental dentro del laboratorio LIANA para el establecimiento de una línea de investigación en robótica cognitiva. Se expondrá el diseño de un entorno robótico simulado que permitirá al LIANA aplicarlo en experimentación sobre mecanismos cognitivos. De esta manera, gracias al desarrollo de este proyecto, el laboratorio cuenta con un mecanismo de integración entre una arquitectura cognitiva (en este caso el MDB del GII) y su propia plataforma de simulación robótica.

El desarrollo de este proyecto promoverá el aumento de la producción científica del LIANA dentro del área de robótica cognitiva. Esto permitirá que el laboratorio participe en convocatorias de investigación nacionales e internacionales que otorguen fondos para aumentar su equipo y personal. Desde un punto de vista científico, la

investigación en robótica cognitiva es un campo altamente estudiado en el cual todavía existe una gran cantidad de preguntas abiertas, como el desarrollo de arquitecturas, robustez en ambientes no estructurados y aceptación social [2]. Este proyecto es un primer paso para que desde LIANA se generen soluciones novedosas a los distintos vacíos de conocimiento existentes. Al tratarse de una implementación simulada, permitirá al LIANA evitar -en esta etapa inicial de la línea de investigación- los costos asociados a la compra y manufactura de una plataforma robótica física. También, el laboratorio podrá aprovechar los beneficios que trae el uso de ambientes simulados, como la capacidad de prototipado flexible, aplicaciones con múltiples agentes y la investigación colaborativa [5].

1.1 Objetivos

1.1.1 Objetivo general

Diseñar un ambiente robótico simulado que permita la integración con diversas arquitecturas cognitivas dentro del LIANA, para la ejecución de tareas de aprendizaje básicas con la arquitectura cognitiva *Multilevel Darwinist Brain (MDB)*.

1.1.2 Objetivos específicos

1. Determinar las especificaciones requeridas por la herramienta de acuerdo con criterios bibliográficos y del LIANA.

2. Implementar dentro de un simulador físico la actuación y sensorización del robot seleccionado, de forma que obtenga datos de su entorno y sea controlable.

3. Desarrollar una redescripción de los datos sensoriales obtenidos del entorno simulado de manera que pueda integrarse con la arquitectura cognitiva *Multilevel Darwinist Brain* (MDB).

4. Validar el funcionamiento de la herramienta desarrollada a partir de una tarea de aprendizaje que utilice la arquitectura MDB.

El principal aporte de ingeniería dado en este proyecto fue el diseño de una herramienta computacional que permite la integración con la arquitectura MDB para la realización de experimentos en robótica cognitiva. La implementación modular de la herramienta otorgará una base sobre la que será posible desarrollar funciones mucho más complejas de cognición y autonomía. Por esta razón, se espera que esta primera implementación pueda guiar el trabajo futuro del LIANA en el área de robótica cognitiva.

1.2 Estructura del documento

A continuación se describe la estructura que se seguirá en los siguientes capítulos de este documento. En el capítulo 2 se presentan los aspectos teóricos y del estado del arte más importantes para la ejecución del proyecto. El capítulo 3 presenta la metodología de diseño seguida en el proyecto. Los capítulos siguientes describen cada uno de los subsistemas que componen el proyecto, de manera que se expone la totalidad del diseño, la implementación desarrollada y se finaliza con los experimentos de validación de cada subsistema. Adicionalmente, cada capítulo tiene una correspondencia directa con un objetivo específico planteado: en el capítulo 4 se presenta el diseño de la plataforma robótica simulada, abordando así el objetivo específico 2; el desarrollo de las capacidades de percepción, acción e integración con el MDB se presenta en el capítulo 5, correspondiente al objetivo específico 3; en el capítulo 6 se presenta la integración de la herramienta junto con la validación experimental correspondiente al objetivo específico 4. El análisis financiero del proyecto se presenta en el capítulo 7. Finalmente, se exponen las principales conclusiones y recomendaciones para el trabajo futuro en el capítulo 8.

CAPÍTULO 2

MARCO TEÓRICO

En este capítulo se presentan los fundamentos teóricos que sustentan el diseño realizado, así como una revisión del estado del arte en la temática que desarrolla este proyecto. Se comenzará tratando el tema de arquitecturas cognitivas en general. Posteriormente, se expondrá la aplicación de las arquitecturas cognitivas en robótica, generando el campo de estudio de la robótica cognitiva. Además, se expondrá el funcionamiento y estructura de la arquitectura MDB. Finalmente, se explican los fundamentos de los mecanismos robóticos, la visión robótica y el diseño de experimentos.

2.1 Arquitecturas cognitivas computacionales

Una *Arquitectura Cognitiva* (AC) es la implementación computacional de un modelo cognitivo. Su objetivo es capturar, explicar y simular una gran variedad de fenómenos psicológicos y cognitivos mediante una estructura unificada [6, p. 5]. Una AC define los componentes de la mente y genera los métodos por los cuales estos componentes interaccionan entre sí para formar un modelo unificado [6, pp. 7-9]. Cada AC está

construida bajo su propio conjunto de premisas, lo que genera una gran diversidad en cómo estas son implementadas. A pesar de la gran diversidad existente, se espera que la mayoría de arquitecturas tengan las siguientes características: comportamiento flexible, operación en tiempo real, racionalidad, base amplia de conocimientos, aprendizaje, habilidades lingüísticas y autoconsciencia [7, p. 3].

Las arquitecturas cognitivas generalmente son clasificadas de acuerdo al tipo de representación de la información que utilizan [7, p. 24]. De acuerdo con este criterio han surgido tres tipos principales de arquitecturas cognitivas: simbólicas, emergentes e híbridas. Las ACs simbólicas representan conceptos mediante símbolos que son manipulados mediante instrucciones predefinidas [7, p. 24]. Las AC emergentes se basan en la auto-organización, y generalmente se basan en el uso de redes neuronales artificiales [8]. En el enfoque híbrido, se combinan características simbólicas y emergentes para contrarrestar las desventajas existentes en los enfoques anteriores [7, p. 24].

En las siguientes secciones se comentarán los elementos principales que componen una AC. Entre estos se comentarán los conceptos de cognición, percepción, atención, memoria y acción.

2.1.1 Cognición

La cognición es el proceso en el cual un ente genera descripciones lógicas del ambiente en el que se desarrolla, las cuales le permitan prosperar en el tiempo de acuerdo con sus características corporales [9, p. 1]. En la figura 2.1 se muestra un diagrama que presenta los componentes fundamentales de la cognición de acuerdo a la teoría presentada por Uexkull [10]. En el diagrama se distinguen dos grupos de información existentes dentro del agente, el modelo del mundo que el agente percibe (*Umwelt*) y los estados internos que caracterizan a la entidad en un periodo específico de tiempo (*Innenwelt*). Las acciones que realiza el agente se ven determinadas tanto por su modelo de mundo como

por sus estados internos, este proceso genera un ciclo de percepción y acción en el cual el agente puede satisfacer las necesidades dictadas por sus estados internos a partir de su influencia en el ambiente.

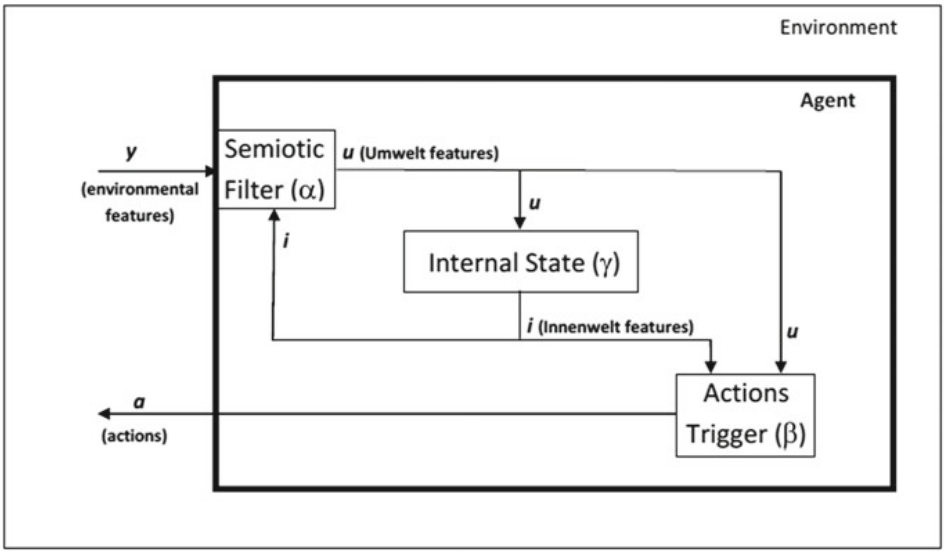


Figura 2.1: Componentes fundamentales de la cognición. Recuperado de: [9]

Muchas arquitecturas cognitivas buscar simular los mecanismos humanos para el procesamiento de la información, de forma que puedan simular la inteligencia humana. Uno de los modelos desarrollados por la psicología experimental para explicar la forma en que los humanos procesan la información es el *Skill-Rule-Knowledge (Habilidad-Regla-Conocimiento)* (SRK) [11]. Este modelo distingue tres niveles de control cognitivo. En el nivel de habilidad (*Skill*) los estímulos sensoriales son procesados para generar señales que pueden utilizarse directamente para generar una acción asociada [12, p. 3]. En el nivel de regla (*Rule*) los datos perceptuales pasan un proceso adicional de reconocimiento que permite asociar el estado a una tarea específica, lo que lleva a ejecutar rutinas aprendidas previamente para esa tarea [12, p. 3]. Cuando se encuentra una situación nueva, se activa el nivel de conocimiento (*Knowledge*). En este nivel se da un procesamiento consciente en el que se reconoce una meta específica y se realiza un planeamiento de las acciones a desarrollar [12, p. 4]. La figura 2.2 presenta un diagrama de bloques del modelo SRK.

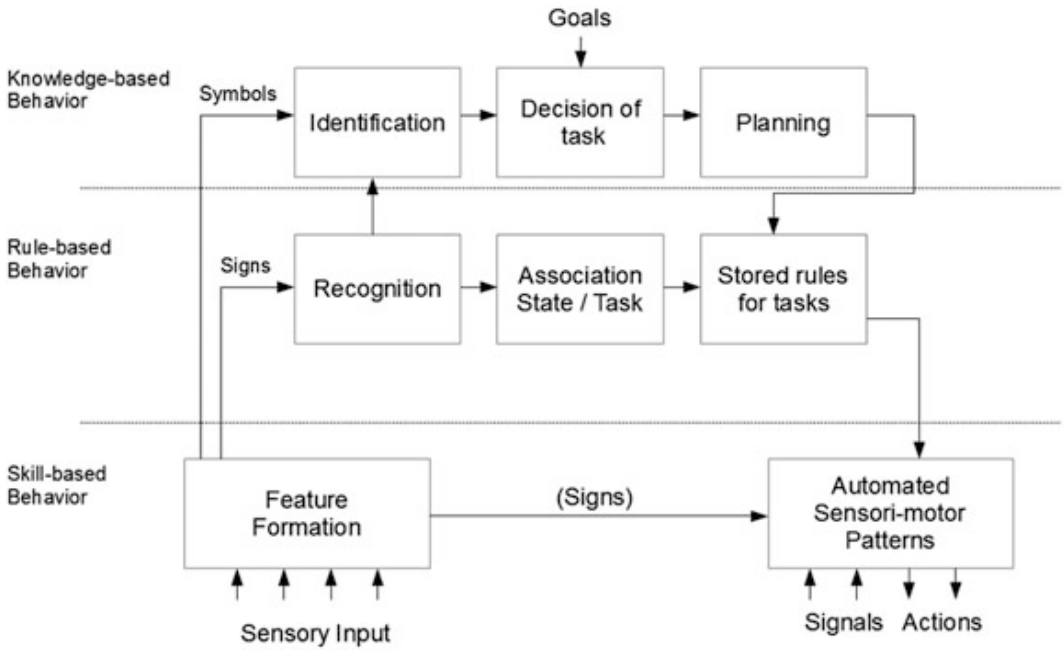


Figura 2.2: Modelo *Skill-Rule-Knowledge*. Recuperado de: [12]

2.1.2 Percepción

La percepción es el proceso de adquisición y procesado de información sensorial de manera que se pueda ver, escuchar, saborear o sentir objetos en el mundo para guiar al organismo a tomar acciones sobre esos objetos [12, p. 6]. Una de las tareas de percepción visual más importantes es la búsqueda de un objeto dentro del campo de visión, en la que sea necesario buscar características específicas del objetivo. Treisman y Gelade [13], presentan la teoría de integración de características para explicar la percepción de objetos. De acuerdo con este modelo, al enfocarse la atención en un área visual se extraen de forma automática características de la información obtenida. En este proceso se crean mapas de localización, color y orientación que son combinados para generar una representación de objetos. Al final del proceso los objetos son procesados por un mecanismo de reconocimiento de alto nivel. En la figura 2.3 se muestra un esquema del modelo de integración de características.

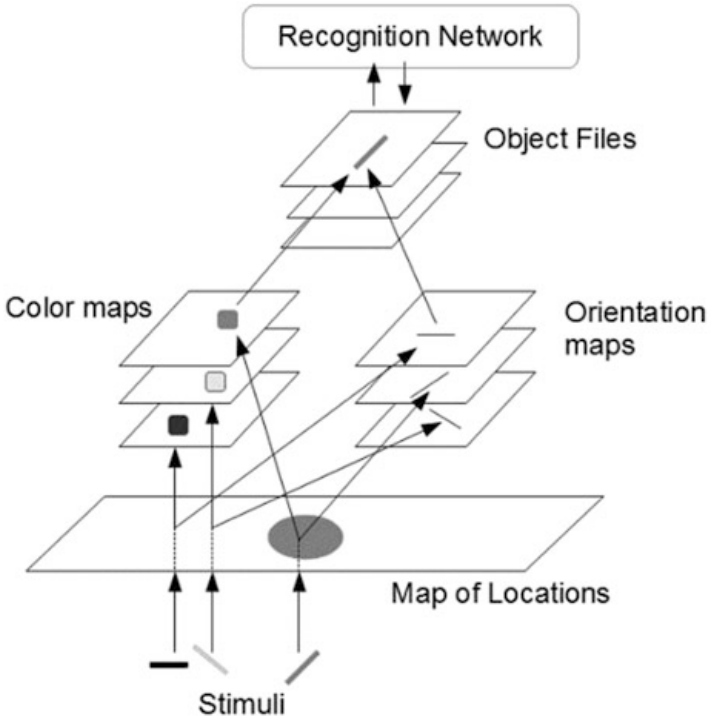


Figura 2.3: Teoría de integración de características. Recuperado de: [12]

2.1.3 Atención

La atención es un mecanismo selectivo que tiene como objetivo la optimización del funcionamiento de un sistema con capacidades de procesado limitadas [3, p. 23]. Tiene la función de seleccionar, a partir de una gran cantidad de información proveniente del ambiente, la información necesaria para realizar la tarea de interés [12, p. 13]. La teoría de recurso unitario propuesta por Kahneman [14] se presenta en la figura 2.4. Esta teoría sostiene que existe una cantidad limitada de recursos atencionales que son asignados a diferentes tareas. Como se muestra en el diagrama, los distintos estímulos psicológicos tienen influencia en la asignación de recursos atencionales a cada una de las tareas que se realizan. El desempeño del agente depende de la cantidad de recursos atencionales que le dedica a cada una de las tareas, por lo que existe un mecanismo de evaluación que determina la cantidad de recursos que requiere cada una de las tareas por realizar. Bajo la perspectiva neurocientífica, Charron y Koechilin [15] observaron que la corteza frontal de los lados izquierdo y derecho procesan de forma separada dos

tareas concurrentes y que la corteza prefrontal se encarga de coordinar ambos procesos. Esa observación sugiere que la atención humana se limita a dos tareas concurrentes [12, p. 14].

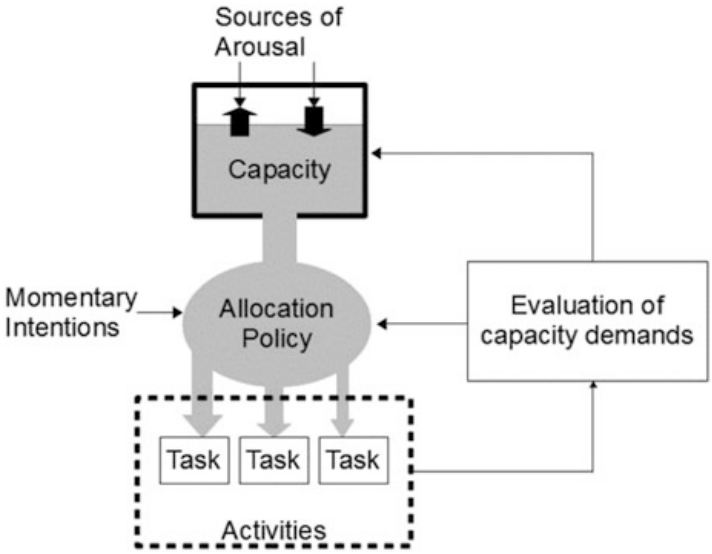


Figura 2.4: Teoría del recurso unitario. Recuperado de: [12]

2.1.4 Memoria

La memoria es un componente clave de todos los procesos cognitivos, ya que en esta se basan los procesos de planeamiento, decisión, atención y procesamiento visual [16]. Generalmente se identifican dos tipos de memoria: la memoria a corto plazo, que mantiene la información consciente en un tiempo corto y la memoria a largo plazo, que se encarga de mantener una gran cantidad de conocimiento por un tiempo largo [16].

La memoria a corto plazo se compone de dos elementos: la memoria sensorial y la memoria de trabajo [16]. En la memoria sensorial se guarda la información recibida por el sistema perceptual. La memoria de trabajo se refiere a la memoria activa en la cual se mantiene la información que será procesada para llevar a cabo una actividad [17, p. 40]. Baddeley presenta un modelo para la memoria de trabajo en el que se identifican cuatro componentes principales: el sistema ejecutivo central, el sistema visual-espacial,

el sistema fonológico y un búfer episódico [18] (Figura 2.5). En este modelo, el sistema central se encarga de dirigir la atención para seleccionar la información visual, espacial y fonológica obtenida de los sistemas respectivos. El búfer episódico se encarga de almacenar la información importante en bloques, de forma que esta pueda mantenerse por mayor tiempo en la memoria de corto plazo [17, p. 41].

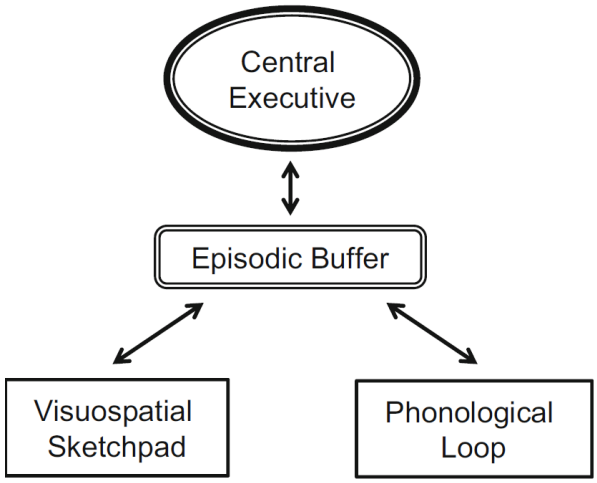


Figura 2.5: Modelo de Baddeley para la memoria de trabajo. Recuperado de: [17]

La memoria a largo plazo es utilizada para guardar información tanto consciente como subconsciente. Entre la información subconsciente se guardan datos como las habilidades motoras. La información consciente almacenada tiene conocimientos generales del mundo (memoria semántica) y eventos pasados experimentados por el agente (memoria episódica) [16]. Fuster presenta una teoría en la cual la memoria a largo plazo es un patrón de conexiones entre neuronas asociadas mediante la experiencia [19]. En la figura 2.6 se ejemplifica el modelo de Fuster, donde se puede observar que al recibir estímulos sensoriales de forma simultánea, se crean asociaciones entre las neuronas. Posteriormente, al recibir un estímulo aislado se activan también las neuronas asociadas al estímulo anterior. Este efecto podría generar, por ejemplo, que al palpar un objeto (Figura 2.6.6) se evoque la imagen visual que fue aprendida anteriormente (Figura 2.6.4).

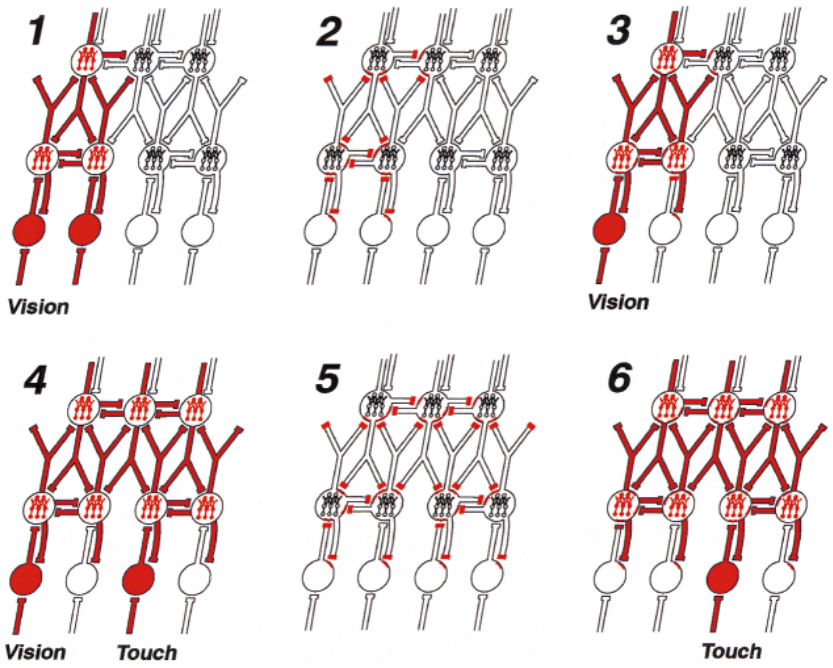


Figura 2.6: Memoria por asociación sensorial. Recuperado de: [20]

2.1.5 Acción

La acción es el proceso en el cual las intenciones del agente, generadas por los mecanismos de alto nivel como percepción, atención y memoria, son convertidas en comandos de bajo nivel que permiten la actuación de su sistema motor [21]. La acción es clave dentro del desarrollo cognitivo debido a que los procesos de autonomía son generados directamente por la interacción del agente con su entorno [22, p. 3].

2.2 Robótica cognitiva

De acuerdo con la *Enciclopedia de las Ciencias del Aprendizaje* la robótica cognitiva se refiere al uso de modelos bioinspirados para el diseño de capacidades sensoriales, motoras, cognitivas y sociales en robots autónomos [2]. El principal objetivo de esta rama de la robótica es crear sistemas de aprendizaje abiertos y autónomos en contraparte con el crear robots que llevan a cabo una única tarea específica [23]. Un robot cognitivo es caracterizado por su capacidad de obtener

conocimiento en una forma adaptativa y autónoma, esto significa que tiene una capacidad de aprendizaje [24]. Además, existe una importante relación con el estudio de la psicología cognitiva, ya que el uso de robótica permite la evaluación de los modelos desarrollados por los investigadores en esta área [25].

2.2.1 Estado del arte

En esta sección se dará una visión general sobre la investigación realizada en robótica cognitiva. Se da especial énfasis a publicaciones que hayan utilizado tareas robóticas de manipulación para evaluar el funcionamiento de diferentes arquitecturas cognitivas. Además, se excluye la investigación realizada en la arquitectura MDB debido a que esta será abarcada en la siguiente sección.

Uno de los ejemplos más importantes de la investigación en robótica cognitiva es iCub, un robot de 53 grados de libertad con el tamaño de un niño de cuatro años [26]. El objetivo de esta plataforma es el desarrollo de habilidades cognitivas que puedan simular el desarrollo de los infantes [22, Ch. 6]. Adicionalmente, iCub tiene una licencia abierta (*GNU General Public Licence*) que permite la modificación libre, esto ha permitido que distintos investigadores utilicen esta plataforma para desarrollar sus estudios. Leitner [27], presentó una arquitectura para dotar de capacidades avanzadas de coordinación mano-ojo al iCub. Entre sus principales logros se destacan obtener capacidades para evitar obstáculos móviles de forma autónoma y la manipulación de objetos complejos a partir de datos visuales. Nguyen et al [28] presentan dentro del desarrollo de la arquitectura cognitiva MACSi, un mecanismo para que el robot iCub pueda obtener información de objetos a partir de manipulación y aprendizaje asistido. En el experimento realizado colocan diversos objetos sobre una mesa, en los cuales el robot decide a partir de procesos creativos autónomos cuál objeto desea explorar.

Otras arquitecturas cognitivas han desarrollado experimentos que involucran capacidades de manipulación robótica. En un experimento presentado por Wilson et

al. [29], se utilizó la arquitectura cognitiva DIARC en un robot PR2 para demostrar el aprendizaje de toma de objetos a partir de un único ejemplo. Se utilizó un simulador físico para emular un proceso de simulación mental que permite mejorar el desempeño en el mundo real, incluso cuando los objetos tienen características distintas a las mostradas en el ejemplo. Dong y Franklin presentaron el desarrollo de un sistema motor para la arquitectura cognitiva LIDA [21]. Utilizaron un youBot simulado en el software Webots para validar el controlador, que permite la sujeción de objetos utilizando un modelo que concuerda con datos de manipulación en humanos. Mohan et al. presentan un experimento robótico utilizando la arquitectura cognitiva SOAR [30]. En el experimento utilizan un robot de escritorio para enseñarle tareas a partir de instrucción verbal, el robot aprende a manipular objetos mediante instrucciones primitivas como "recoge el cilindro azul y colócalo en la despensa". Los investigadores proponen que el mecanismo utilizado permitirá que el robot aprenda a resolver juegos como Torre de Hanoi.

2.3 Multilevel Darwinist Brain (MDB)

El MDB es una arquitectura cognitiva desarrollada por el Grupo Integrado de Ingeniería (GII) que busca lograr robots cognitivos que puedan aprender distintas tareas y objetivos de manera adaptativa, utilizando un enfoque basado en la neuroevolución [23]. La versión actual de la arquitectura, denominada MDB-epistémico busca el aprendizaje "sin límites" a partir de la integración incremental de bloques de conocimiento [31]. Los conceptos básicos utilizados por esta arquitectura son los siguientes [31]:

- **Vector perceptual** ($\mathbf{P}[t]$): Conjunto de valores que recibe la arquitectura a partir de los sensores del robot en un instante de tiempo.
- **Espacio perceptual** ($\mathbf{P}$): Conjunto de todos los posibles vectores perceptuales.

- **Vector de actuación** ($\mathbf{A}[t]$): Valores enviados a los actuadores del robot en un instante de tiempo.
- **Espacio de actuación** ($\mathbf{A}$): Conjunto de todos los posibles vectores de actuación.
- **Episodio** ($\mathbf{E}$): Muestra del estado del robot en un instante. Incluye a los vectores perceptuales y de actuación en el tiempo t y el vector perceptual en el tiempo $t+1$.
- **Modelo de mundo** ($\mathbf{WM}$): Función que permite predecir el estado perceptual futuro del robot a partir del vector perceptual actual y la acción realizada.
- **Utilidad**: Es una medida que muestra que el robot ha aumentado la satisfacción de sus objetivos. Esta se ve determinada por su interacción con el ambiente.
- **Meta**: Estado perceptual ($\mathbf{P}[t]$) en el cual el robot recibe utilidad.
- ***Policy*** (π): Estructura de decisión que indica las acciones a realizar de acuerdo con el estado perceptual.

La estructura básica de la arquitectura se muestra en la figura 2.7. Se distinguen tres elementos principales: el modelador de mundo (MP-Learner), el motor motivacional y la memoria a largo plazo (LTM, por sus siglas en inglés). Dentro del modelador de mundo se da el proceso de aprendizaje de bloques primitivos de conocimiento, como los modelos de mundo y las *policies* [31]. El aprendizaje del modelador de mundo se da utilizando los datos guardados en la memoria episódica de corto plazo. El motor motivacional se encarga de generar modelos que permitan evaluar los estados sensoriales del robot de manera que este pueda no solo cumplir las metas propuestas, sino también explorar el mundo para mejorar sus modelos internos y aumentar su conocimiento [31]. La *Long-Term Memory (Memoria a largo plazo)* es el elemento central de la arquitectura

cognitiva que permite relacionar los distintos bloques de conocimiento generados por las otras estructuras de la arquitectura.

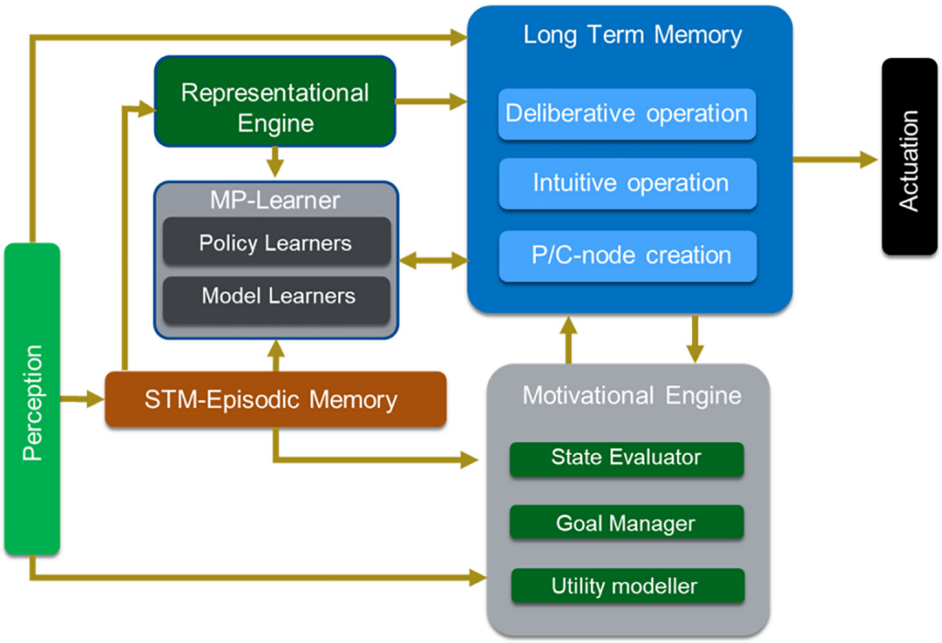


Figura 2.7: Diagrama de bloques de la arquitectura e-MDB. Recuperado de: [31]

2.3.1 Memoria a largo plazo

La LTM en la arquitectura MDB utiliza una estructura conexionista para generar relaciones de asociación entre los distintos elementos de conocimiento, esta asociación depende de regularidades que el robot identifica en el ambiente [32]. En la figura 2.8 se muestra la estructura de la LTM, en esta se asocian las percepciones (PN), los modelos de mundo (WM), las metas (G) y las *policies* (π).

En la LTM las percepciones se guardan utilizando *clases perceptuales* o *P-Nodes* (PN), definidas como áreas del espacio perceptual que comparten características comunes. Los *P-Nodes* representan zonas en las cuales el robot puede responder de la misma manera al estar dentro de un mismo contexto [32].

Las asociaciones generadas en la LTM se generan mediante los nodos de contexto o *C-Nodes* (CN). Un *C-Node* es un bloque de conocimiento que representa un contexto,

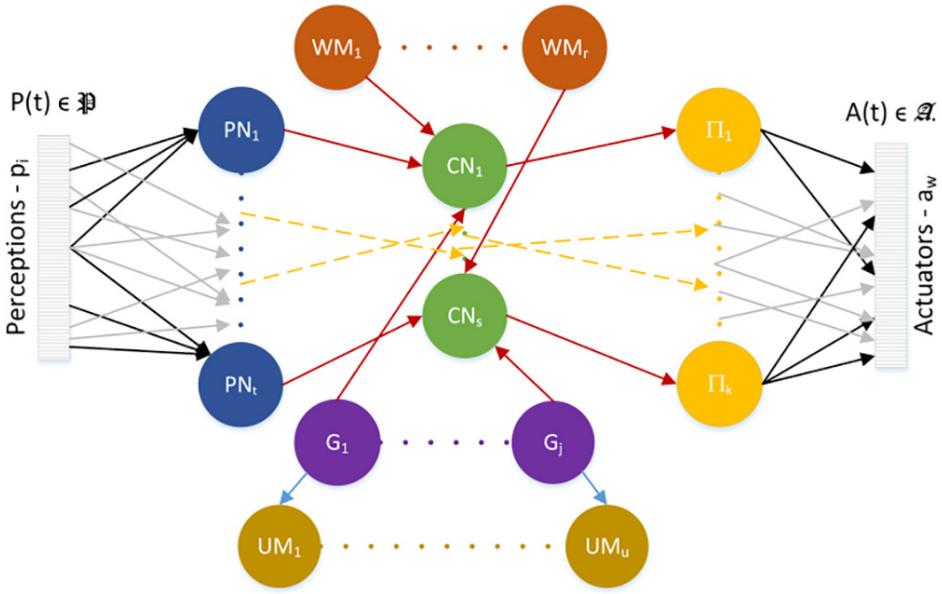


Figura 2.8: Estructura de la memoria a largo plazo de la arquitectura e-MDB. Recuperado de: [31]

el cual se establece mediante el conjunto $\{PN,\ WM,\ G,\ \pi\}$, que permite asociar percepciones, mundos, metas y las acciones que han generado utilidad. Los nodos de contexto permiten que el robot, al conocer su estado perceptual, el mundo actual y la meta activa (conjunto $\{PN,\ WM,\ G\}$), pueda seleccionar las acciones más apropiadas basándose en su experiencia [33, p.7].

2.3.2 Experimentos realizados

Durante el desarrollo del MDB se han presentado múltiples experimentos robóticos, donde destaca el uso de un robot Baxter como plataforma experimental para ejecutar tareas de manipulación. En [34] se le dio a Baxter la tarea de colocar una bola en una canasta, sin embargo debía presionar un botón antes para poder acceder al objeto. Este experimento permitió comprobar la capacidad del motor motivacional para identificar metas secundarias que debe cumplir antes de lograr el objetivo principal. Como demostración de la capacidad del motor motivacional para generar diferentes modelos de utilidad al combinarlos mediante modulación, se colocó

a Baxter en un entorno de cocina en el cual este debía ser capaz de manipular la comida mientras evita fuentes de calor [35]. En [36] se instruyó a Baxter a realizar la limpieza de un espacio de trabajo para demostrar la capacidad del motor motivacional para encontrar funciones de utilidad. Finalmente, en un experimento reciente se integraron las capacidades del motor motivacional para enseñarle a Baxter una tarea compleja de ensamblaje de objetos [37].

Para demostrar el funcionamiento de la LTM se ha utilizado un experimento de manipulación básica para cual se definieron distintos contextos para que Baxter pudiera identificar y aprender. En [32] se muestra la capacidad de generalización del espacio perceptual a partir de *P-Nodes* utilizando tanto métodos probabilísticos como redes neuronales para su definición. Este experimento fue ampliado en [33] al agregar más combinaciones de tareas y mundos, se concluyó que la LTM permite el aprendizaje duradero de tareas, incluso cuando estas cambian en el tiempo. Finalmente, en [31] se muestra la capacidad de integración entre la LTM y las demás estructuras del MDB al realizar un experimento en el que se utilizaron las *policies* aprendidas por la arquitectura para generar un modelo que las relaciona para generar *meta-policies* de mayor nivel.

2.4 Robótica

La robótica se puede definir como el estudio de las máquinas que pueden reemplazar a humanos en la ejecución de diferentes tareas [38]. Los robots pueden ser clasificados en cuatro grupos: manipuladores, vehículos robóticos, sistemas humano-robot y robots bioinspirados [39].

2.4.1 Manipuladores robóticos

Un manipulador robótico es una secuencia de vínculos rígidos (*links*) conectados mediante articulaciones (*joints*) que le permiten ejecutar movimientos. Al final de la

secuencia se encuentra una herramienta, llamada **efector final** que le permite realizar la tarea deseada [38]. Un manipulador es caracterizado por su número de grados de libertad y por su espacio de trabajo.

Es importante dentro de la robótica el poder describir la posición en la que se encuentran los distintos vínculos del manipulador y su efector final. Se definen dos espacios para describir la posición de un manipulador, el **espacio de articulaciones** en el cual se indican las posiciones en las que se encuentra cada una de las articulaciones del manipulador y el **espacio cartesiano** o espacio de trabajo, en el que se define la pose que tiene el efector final con respecto al sistema de coordenadas de referencia del manipulador [40].

La **cinemática directa** es el problema geométrico que consiste en obtener la posición y orientación del efector final, a partir de los valores conocidos en el espacio de articulación [40]. Con respecto a este problema, el número de grados de libertad consiste en el número mínimo de variables de posición que deben establecerse para conocer la posición de todos sus elementos, en el caso de manipuladores de cadena cinemática abierta el número de articulaciones es la cantidad de grados de libertad [40].

La **cinemática inversa** es el problema en el cual, dada la posición y orientación del efector final, se busca obtener el conjunto de las posibles configuraciones en el espacio de articulación que permiten obtener la pose especificada [40]. Todos los puntos en los cuales existe una solución para la cinemática inversa definen el espacio de trabajo del manipulador [40].

2.4.1.1 Métodos numéricos para la solución de la cinemática inversa

Es ampliamente sabido que el problema de la cinemática inversa es complejo de resolver, esto debido a que las ecuaciones generalmente son no lineales y que es posible que existan infinitas, múltiples o ninguna solución para una determinada pose [38,

p. 91]. Por esta razón se han desarrollado múltiples métodos numéricos que permiten encontrar una solución a la cinemática inversa.

Una de las alternativas es el uso de mínimos cuadrados para aproximar linealmente la solución del problema. Se puede encontrar la solución de la cinemática inversa al resolver iterativamente la ecuación 2.1, donde $\vec{\mathbf{e}}$ es el cambio deseado en la pose del manipulador, $\boldsymbol{\Delta\theta}$ es el cambio en la posición de las articulaciones que permite que el efector final alcance la posición deseada y $J(\boldsymbol{\theta})$ es la matriz jacobiana del manipulador. En [41] se presentan diversos métodos para resolver la solución numérica presentada.

$$\vec{\mathbf{e}} = J(\boldsymbol{\theta})\boldsymbol{\Delta\theta} \tag{2.1}$$

Existe una implementación computacional para la cinemática inversa publicada en la *Kinematics and Dynamics Library* (KDL), que utiliza el método de Newton-Raphson y la descomposición en valores singulares para encontrar la solución a la ecuación 2.1. En [42] y [43] se pueden encontrar detalles adicionales sobre la implementación de KDL.

2.4.1.2 Planeamiento de trayectorias

La generación de trayectorias consiste en el cálculo de funciones continuas que permitan un movimiento controlado y coordinado del manipulador cuando se mueve de una pose a otra [40]. Una consideración importante en la generación de trayectorias es que se deben evitar las colisiones y estados inalcanzables, por esta razón se han implementado diversos algoritmos que permiten el planeamiento de rutas que tomen en cuenta estas consideraciones.

La *Open Motion Planning Library* (OMPL) es una biblioteca que contiene una gran cantidad de algoritmos de planeamiento de trayectorias [44]. Uno de los métodos más sencillos disponibles en OMPL es el *Rapidly-exploring Random Trees* (RTTs), en el cual se realiza una búsqueda en el espacio de articulación mediante el crecimiento de un árbol de estados conectado mediante movimientos válidos de las articulaciones

[45]. Este método permite considerar restricciones generadas por obstáculos y por la mecánica del manipulador.

2.4.2 Visión Robótica

El uso de sistemas de visión permite que el sistema robótico obtenga información cualitativa y cuantitativa del ambiente que le rodea [38]. A continuación se presentan los métodos utilizados en sistemas de visión robótica que utilizan una cámara única.

En la figura 2.9 se muestran los distintos sistemas de coordenadas que constituyen un sistema de visión robótico. El objetivo del sistema es poder encontrar, a partir de un punto q encontrado en la imagen, las coordenadas reales del punto Q [39].

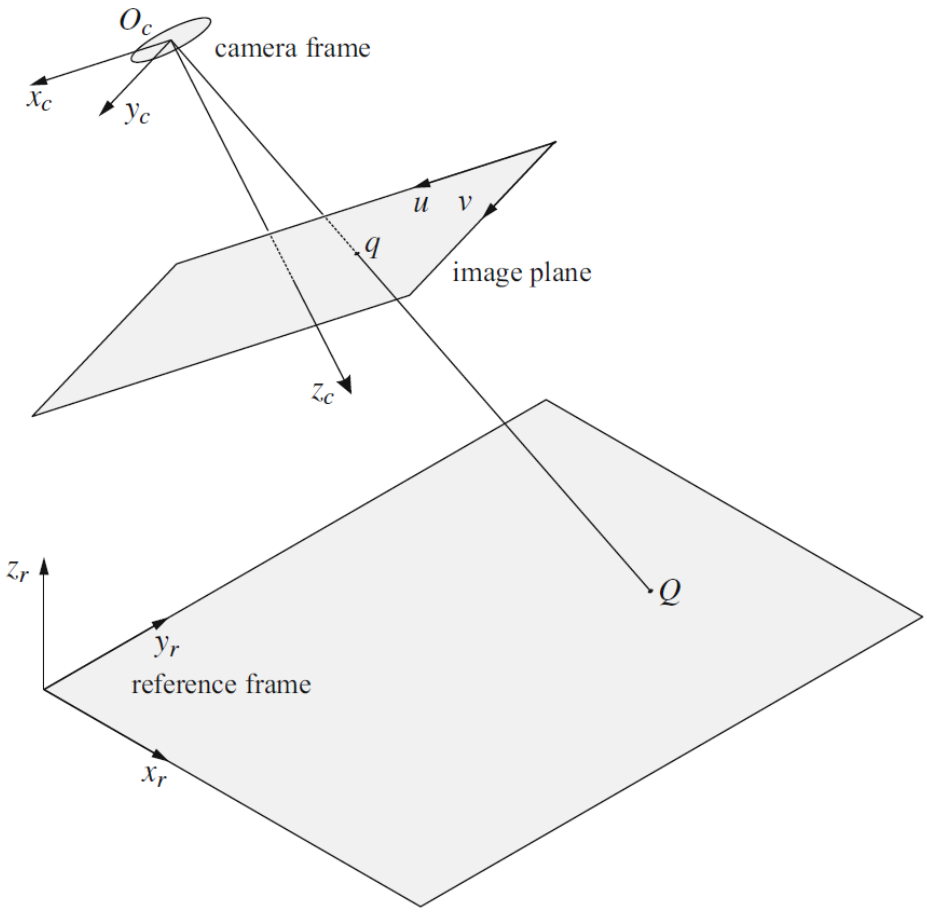


Figura 2.9: Sistemas de coordenadas en un sistema de visión robótico. Recuperado de: [39]

La proyección de un punto real en una imagen está descrito por la ecuación 2.2. En la que se pueden encontrar las coordenadas u y v en la imagen en la que se proyecta un punto determinado (x_c, y_c, z_c) localizado según el sistema de coordenadas relativo a la cámara. La proyección también depende de un factor de escala s que por lo general es desconocido.

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{P} \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} \tag{2.2}$$

La matriz $\mathbf{P}$ contiene los parámetros intrínsecos de la cámara. Los valores de f_x y f_y representan las distancias focales a lo largo de los ejes de la imagen y se definen según la ecuación 2.4, donde f_c es la distancia focal y (D_x, D_y) representan el tamaño del píxel. El centro de la imagen se localiza en la posición (u_0, v_0).

$$\mathbf{P} = \begin{bmatrix} f_x & 0 & u_0 & 0 \\ 0 & f_y & v_0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{2.3}$$

$$\begin{aligned} f_x &= \frac{f_c}{D_x} \\ f_y &= \frac{f_c}{D_y} \end{aligned} \tag{2.4}$$

Mediante la ecuación 2.2 se puede calcular el punto en la imagen en el cual quedará proyectado un punto del espacio específico. Sin embargo, si se desea obtener un punto del espacio a partir de la imagen, la ecuación queda indeterminada por exceso de variables [39, p. 113-115]. Por esta razón, se requiere información adicional sobre las dimensiones del ambiente para poder realizar la proyección inversa.

2.4.3 Middlewares Robóticos

Un *middleware* es una capa de abstracción que actúa entre el sistema operativo y las aplicaciones de *software* en un sistema robótico [46]. Sus principales funciones son: manejar la heterogeneidad del hardware, simplificar el diseño del software y reducir los costos de desarrollo [46]. Además, permite distribuir los diferentes algoritmos de un sistema robótico en módulos que permiten su reutilización e integración.

El *middleware* utilizado durante el proyecto es el *Robotic Operating System* (ROS), que fue diseñado para la ejecución modular operaciones robótica en una red con múltiples equipos y para permitir que sus componentes sean desarrollados en distintos lenguajes de programación [47]. En el uso de ROS se distinguen los siguientes conceptos:

- **Nodo:** Es un programa que utiliza la red de ROS para comunicarse con otros nodos [48, p. 5].
- **Mensaje:** Es una estructura de datos definida que es utilizada por los nodos para enviar información [47].
- ***Topic:*** Es un canal de información en el cual nodos envían mensajes de un tipo específico. Múltiples nodos pueden leer o escribir en un *topic* al mismo tiempo [47].
- **Servicio:** Utilizados para realizar transacciones sincrónicas de información entre nodos. Se definen por un mensaje de solicitud y un mensaje de respuesta [47].
- **Acciones:** Permiten la realización de tareas de larga duración y proveen *topics* para recibir realimentación del proceso o cancelar la tarea [48, pp. 64-65].
- **Paquete:** Grupo de código en el que se envuelven los distintos componentes para realizar una función específica. Múltiples paquetes se ejecutan simultáneamente en un sistema robótico [48, p. 9].

2.4.4 Simulación robótica

La simulación computacional es una técnica que se utiliza para probar de forma eficiente nuevos conceptos, estrategias y algoritmos dentro de la investigación robótica [49]. El uso de simulaciones toma un papel muy importante para acelerar el diseño de robots inteligentes que se utilizan en ambientes no estructurados [50].

Para la implementación del proyecto se utilizó el simulador *Gazebo* [51], que utiliza la *Open Dynamics Engine* (ODE) para la simulación física. ODE es un motor de físicas diseñado para simular la dinámica y cinemática de cuerpos rígidos articulados. Los siguientes conceptos clave se extraen del manual de ODE [52]:

- **Cuerpos Rígidos:** Contienen la información que representa un objeto como su posición, orientación, velocidad, centro de masa e inercia.
- **Restricciones:** Son relaciones entre cuerpos rígidos que fuerzan que estos se orienten en posiciones específicas. Permiten simular articulaciones.
- **Colisiones:** En cada iteración se calculan los puntos de contacto y se calcula la fuerza normal. La fuerza del contacto se calcula según los parámetros k_p y k_d, que permiten simular el contacto como un sistema resorte-amortiguador.
- **Fricción:** Se utiliza una aproximación de tipo pirámide, en la que se calcula la fuerza límite que soporta cada punto de contacto según la fuerza normal y el coeficiente de fricción.

2.5 Diseño de experimentos

El *Diseño de Experimentos* (DOE) de acuerdo con Gutierrez y de la Vara, "Consiste en planear y realizar un conjunto de pruebas con el objetivo de generar datos que, al ser analizados estadísticamente, proporcionen evidencias objetivas que permitan responder las interrogantes planteadas por el experimentador sobre determinada situación." [53,

p. 5]. Con respecto a esta técnica estadística se distinguen los siguientes conceptos clave:

- **Experimento**: Cambio en las condiciones de operación de un sistema para medir el efecto sobre una o varias de sus propiedades [53, p. 7].
- **Variables de respuesta**: Resultados obtenidos de una prueba experimental que miden el desempeño del sistema [53, p. 7].
- **Factores**: Son las variables que describen al sistema. Si existen mecanismos para fijarlos en un valor se denominan *factores controlables* [53, p. 8].
- **Niveles**: Valores que se asignan a cada uno de los factores estudiados en un diseño experimental [53, p. 8].
- **Experimento factorial**: Tienen como objetivo estudiar el efecto que tienen varios factores sobre una o varias respuestas. En un experimento factorial completo se estudian en orden aleatorio todas las combinaciones de niveles de los factores estudiados [53, p. 128].
- ***Análisis de Varianza* (ANOVA)**: Técnica de análisis de datos experimentales en la que se separa la variabilidad observada de acuerdo con la contribución de cada uno de los factores que influye en el experimento [53, p. 65].

CAPÍTULO 3

METODOLOGÍA

A continuación se exponen los aspectos metodológicos que se utilizaron para guiar el desarrollo del proyecto. Se tratarán los aspectos teóricos de la metodología y además se describe cómo esta fue adaptada para la ejecución del proyecto.

3.1 Metodología de diseño

El diseño es la formulación de un plan para satisfacer una necesidad específica o resolver un problema particular [54, p. 4]. Esta definición requiere que se realice un planeamiento estructurado de las etapas que permitan -a partir de la definición de una necesidad específica- llegar a una solución aceptable. Para este proyecto se adoptó la metodología mostrada en la figura 3.1, la que se basa en el proceso de desarrollo del concepto presentado en [55, pp. 16-18].

A continuación se detallan cada una de las etapas mostradas en la figura 3.1.

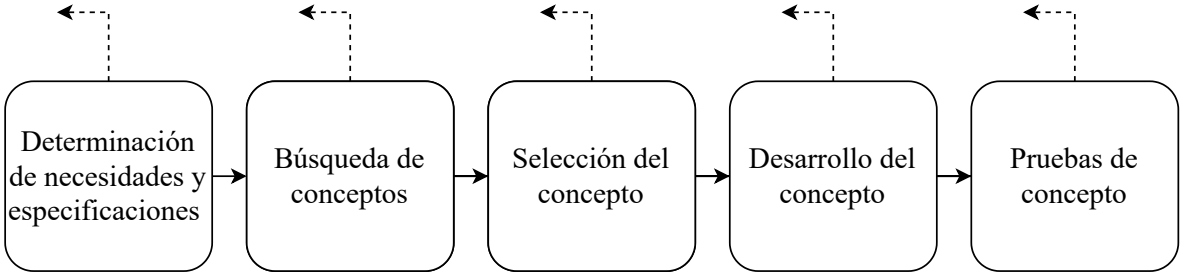


Figura 3.1: Metodología de diseño seguida. Elaboración propia

3.1.1 Determinación de necesidades y especificaciones

El objetivo de esta etapa es entender lo que requiere el cliente y que sea comunicado de forma clara. Las necesidades se sintetizan en una serie de enunciados jerárquicos que establecen las características que la herramienta debe tener. Las especificaciones describen de forma medible lo que la herramienta tiene que hacer, de manera que se satisfagan las necesidades. Cada especificación consiste en una métrica de desempeño y un rango de valores en el cual debe encontrarse esa métrica para que sea aceptable para el cliente. En esta etapa se recurre a entrevistas con el cliente e investigación bibliográfica sobre otras plataformas robóticas.

Cada una de las necesidades y especificaciones lleva asociado un valor de importancia, este se utiliza para priorizar el desarrollo de las características más importantes deseadas por el cliente. La tabla 3.1 muestra la escala de importancia que se utiliza a lo largo del diseño.

Tabla 3.1: Valores de importancia utilizados para las necesidades y especificaciones.

Valor	Descripción
1	Es indeseable
2	No es importante
3	Es deseable
4	Es importante
5	Es imprescindible

3.1.2 Búsqueda de conceptos

Esta etapa busca explorar todos los posibles conceptos de herramienta que permitan satisfacer las necesidades del cliente. Se comenzará con una descomposición del problema a solucionar, para ello se utilizará la descomposición en "caja negra" [55, p. 124]. Como se muestra en la figura 3.2 se comienza representando el sistema o subsistema mediante flujos de *material*, *energía* y *señales*. Las líneas delgadas (energía) representan la transmisión y transformación de energía dentro de un sistema, las líneas gruesas representan movimientos de material y las líneas discontinuas representan señales de datos para el control. Una vez que se ha definido una caja negra, esta se divide en subfunciones que permiten detallar el funcionamiento del producto a diseñar.

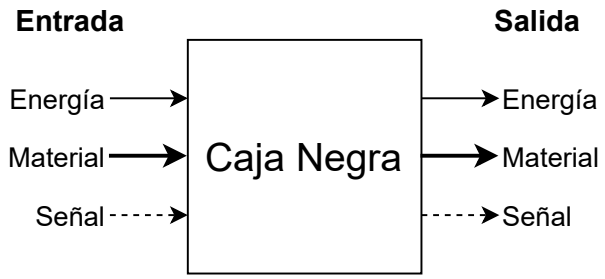


Figura 3.2: Representación de caja negra. Adaptado de: [55, p. 124]

Una vez que se ha realizado una subdivisión suficientemente detallada, se pueden buscar soluciones específicas para cada función, este proceso de búsqueda de conceptos hará uso tanto de fuentes externas como de soluciones encontradas mediante un proceso creativo interno. La búsqueda generará un conjunto de *fragmentos de concepto* que al combinarse sistemáticamente generan los distintos conceptos de solución.

3.1.3 Selección del concepto

En esta etapa se realiza un análisis secuencial de las distintas soluciones encontradas, estas son evaluadas según los criterios establecidos en las necesidades del

cliente junto con otras restricciones como costo y complejidad. Este proceso permite eliminar soluciones hasta converger en el concepto más prometedor.

Se hará uso de un análisis relativo, esto significa que se tomará un concepto "promedio" como referencia y los demás se evaluarán de acuerdo a su desempeño respecto al concepto de referencia. Dos procesos serán utilizados para la selección de conceptos: filtrado y evaluación. El filtrado buscará disminuir rápidamente el número de conceptos considerados, para esto se utilizará una matriz donde se comparará cada una de las características de los conceptos respecto a una referencia y se asigna una puntuación como mejor (+), peor (-) o igual (0) [55, pp. 150-153]. El proceso de evaluación busca una comparación más refinada entre los conceptos, para esto, a cada criterio de selección se le dará un peso que pondera las calificaciones relativas que se le dan a cada concepto. La selección se realiza al sumar la puntuación obtenida por cada concepto, la tabla 3.2 muestra la escala utilizada para la evaluación de conceptos.

Tabla 3.2: Escala de desempeño relativo utilizada para la evaluación de conceptos.

Calificación	**Desempeño Relativo**
1	Mucho peor que la referencia
2	Peor que la referencia
3	Igual que la referencia
4	Mejor que la referencia
5	Mucho mejor que la referencia

3.1.4 Desarrollo del concepto

Esta etapa consiste en la implementación del concepto seleccionado. Se utilizarán las especificaciones planteadas para generar de manera progresiva cada una de las funciones requeridas por la herramienta.

3.1.5 Pruebas de concepto

Consiste en una serie de experimentos que permitirán establecer el grado de cumplimiento de las necesidades y especificaciones establecidas. Los datos obtenidos permitirán además ajustar parámetros de la herramienta para optimizar su funcionamiento. Estas pruebas se detallan en la sección 3.4

3.2 Bitácora del producto

Una bitácora de producto permite documentar el plan de proyecto establecido y los resultados de la fase de desarrollo [55, p. 377]. A solicitud del cliente se utilizará una bitácora de producto para sintetizar la documentación generada en cada una de las etapas de la metodología. Esto permitirá que el cliente tenga un manual de referencia en el cual encontrar el detalle de cada una de las etapas del diseño. Este documento se puede encontrar en el apéndice A.

3.3 Descomposición del proyecto

Debido a la complejidad que representa el sistema a diseñar, el primer paso del desarrollo conceptual fue generar una descomposición del problema. En la figura 3.3 se muestra el diagrama de bloques utilizado para subdividir la herramienta en sus principales subsistemas.

El subsistema de la plataforma robótica representa todas las estructuras correspondientes al robot a utilizar. Recibe como entradas las señales que controlan sus actuadores, además se ve influenciado por la acción del ambiente. Como salidas envía datos sensoriales recogidos y también genera fuerzas de manipulación sobre el ambiente. El subsistema de ambiente es el encargado de generar el mundo donde la plataforma robótica se desempeña. Este recibe como entrada la influencia del robot y

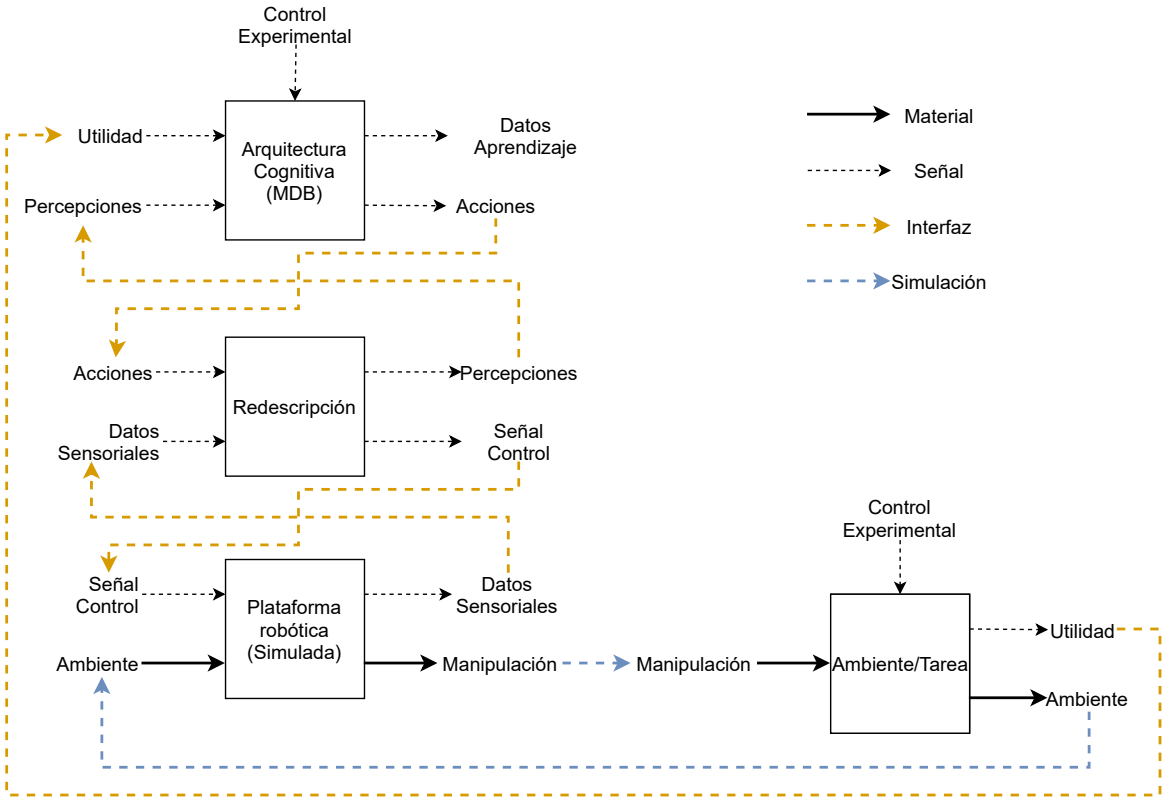


Figura 3.3: Diagrama de bloques identificado para el proyecto. Elaboración propia

además una señal de control experimental que le permite modificar su estado de acuerdo a lo necesario por el experimentador. Como salidas se tiene la representación del mundo y sus objetos, también tiene la capacidad de detectar cuando el robot ha cumplido la tarea deseada y lo comunica mediante la señal de utilidad.

El subsistema de redescripción es utilizado para integrar los datos de bajo nivel correspondientes a la plataforma robótica con las representaciones de alto nivel que utiliza la arquitectura cognitiva. Este bloque recibe datos sensoriales y obtiene información de estos para formar percepciones; por otro lado, toma las acciones propuestas por la arquitectura cognitiva y las transforma en señales de control para accionar el robot.

El subsistema de arquitectura cognitiva se encarga de decidir cuáles acciones debe realizar el robot de acuerdo con los datos de percepción y utilidad. En este subsistema se generan las capacidades de aprendizaje y memoria del robot. Para este proyecto se

hará uso de la arquitectura cognitiva MDB, específicamente su estructura de memoria a largo plazo (LTM).

Para acoplar cada uno de los subsistemas expuestos será necesario crear una interfaz computacional, la cual es representada como la línea discontinua naranja en la figura 3.3. Por otro lado, las interacciones entre subsistemas que se dan mediante simulación física se representan como la línea discontinua azul en la figura 3.3.

El diseño detallado y la implementación de cada uno de los subsistemas mencionados será expuesto los siguientes capítulos.

3.4 Diseño de experimentos

Para evaluar el cumplimiento de las especificaciones críticas de la herramienta se planteó una serie de experimentos. Para validar cada uno de los subsistemas se realizaron los siguientes experimentos:

- **Caracterización de manipuladores:** Tiene como objetivo determinar la precisión de los manipuladores y su espacio de trabajo.
- **Toma de objetos:** Buscará cuantificar la confiabilidad con la que el robot puede tomar objetos en el ambiente mediante rutinas de *pick and place*.
- **Percepción visual:** Permitirá evaluar la precisión con la que el subsistema de redescripción obtiene datos a partir de la información visual del sistema.

Estos experimentos y los resultados obtenidos serán detallados en los capítulos siguientes.

3.4.1 Prueba de aprendizaje

Para validar el funcionamiento de la herramienta como parte de la experimentación en el desarrollo de arquitecturas cognitivas, se realizará un

experimento de aprendizaje que utilice el módulo de memoria a largo plazo del MDB, tomando como base el experimento planteado en [32]. La figura 3.4 muestra un diagrama general del experimento. La tarea consiste en que el robot deposite el objeto dentro de la canasta. Al comienzo del experimento ambos objetos son colocados en posiciones aleatorias en el área de trabajo. El objeto puede colocarse en cualquier parte del área de trabajo, mientras que la canasta siempre estará al alcance de alguno de los manipuladores. Presionar el botón provoca que el objeto se mueva para estar al alcance de alguno de los manipuladores en caso de que este se encuentre fuera del alcance. Al depositar el objeto en la canasta se reinicia el ambiente.

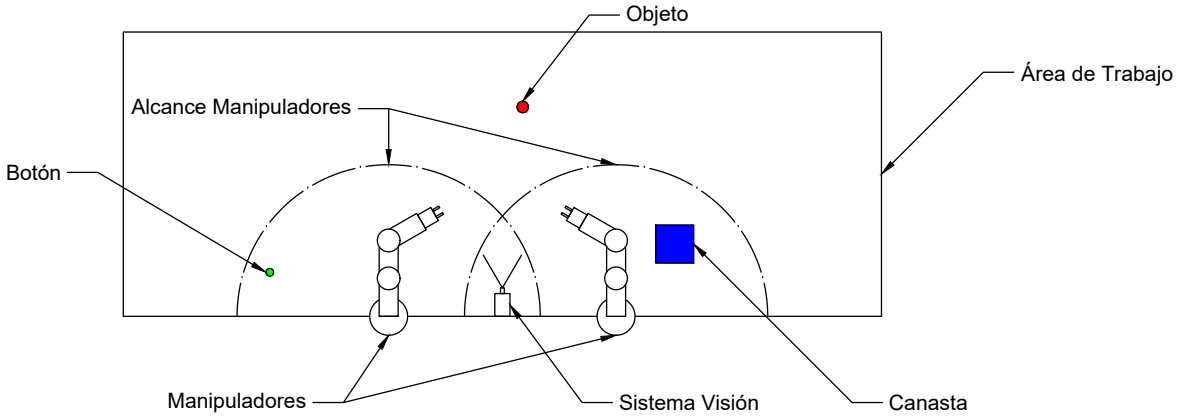


Figura 3.4: Diagrama del experimento de aprendizaje. Elaboración propia: AutoCAD

El vector perceptual será conformado por dos sensores virtuales de posición y dos sensores de tacto. Los sensores de posición indican la distancia y el ángulo en el que se encuentran el objeto y la canasta de la base del robot, mientras que los sensores de tacto indican si el manipulador tiene sujetado un objeto en cada uno de sus brazos. La descripción de las acciones que puede realizar el robot durante el experimento se encuentra en la tabla 3.3.

El experimento analizará el aprendizaje generado por la LTM, por lo tanto, este será el factor a analizar. Se registrará en cada iteración del experimento si la acción seleccionada generó recompensa. Se analizará la proporción de acciones recompensadas y su evolución a través del experimento para determinar si se manifiesta el aprendizaje

de la tarea ejecutada.

Tabla 3.3: *Policies* disponibles para el robot.

Policy	Descripción
grasp_right	Toma el objeto con el brazo derecho
grasp_left	Toma el objeto con el brazo izquierdo
press_button	Presiona el botón de ayuda
place_object_right	Deposita el objeto en la canasta con el brazo derecho
place_object_left	Deposita el objeto en la canasta con el brazo izquierdo
change_hands	Cambia el objeto de un brazo a otro

CAPÍTULO 4

DISEÑO ROBÓTICO

En este capítulo se expone el proceso de diseño seguido para los subsistemas que representan la componente robótica de la herramienta, denominados *plataforma robótica* y *ambiente* en la figura 3.3. Se comenzará detallando los requerimientos indicados por el cliente, que guiaron el diseño expuesto en las secciones posteriores. Seguidamente, se explicarán aspectos del diseño e implementación del robot y el ambiente simulado. Se finalizará presentando los resultados de los experimentos de caracterización de manipuladores y de toma de objetos mencionados en la sección 3.4.

4.1 Identificación de requerimientos

En entrevistas realizadas con el cliente en etapas tempranas del proyecto, se indagó sobre los requisitos que debía cumplir la herramienta y la información obtenida de estas entrevistas se sintetizó en un conjunto de necesidades. En este capítulo interesan las necesidades que tratan aspectos relacionados con la componente robótica de la herramienta, las cuales son presentadas en la tabla 4.1.

Se puede observar que las necesidades de mayor importancia indican que la herramienta debe contar con una plataforma robótica que tenga la capacidad de realizar la manipulación de objetos y que además tenga un conjunto de sensores que le permitan obtener información de su ambiente. Por otro lado, los clientes plantearon la necesidad de que la implementación cuente con una estructura de software modular. Esta arquitectura de software busca que la herramienta implemente sus funciones en componentes que realizan tareas pequeñas, esto facilitará el procesamiento de datos en un computador externo y la interfaz entre sensores y actuadores. Finalmente, es importante para el LIANA que la herramienta sea de desarrollo abierto, esto implica tanto aprovechar hardware y software que sea publicado de forma abierta como facilitar el rediseño y adaptación de la herramienta para el LIANA mediante una documentación adecuada.

Tabla 4.1: Necesidades concernientes al diseño robótico de la herramienta.

No.	Necesidades Interpretadas Cliente	Importancia
1.	**La herramienta incluye la implementación de una plataforma robótica**	
1.1	La plataforma robótica cuenta con capacidades de sensorización	5
1.2	La plataforma robótica tiene herramientas para la manipulación	5
1.3	La plataforma robótica tiene una morfología que le permite la manipulación ambidiestra	3
1.4	La plataforma robótica es de bajo costo	3
1.5	La plataforma robótica tiene capacidad de locomoción	2
2.	**La herramienta permite la obtención de datos perceptuales**	
2.2	El sistema perceptual recibe datos mediante un sistema de visión	4
3.	**La herramienta es capaz de controlar las acciones ejecutadas**	
3.1	El sistema de acciones permite la ejecución de movimientos básicos	5
5.	**La herramienta se implementa en una arquitectura de software modular**	
5.1	La herramienta permite el procesamiento en un computador externo	4
5.2	La herramienta permite la interfaz de sensores y actuadores con un host de procesado	3
6.	**La herramienta cuenta con documentación adecuada**	
6.3	La herramienta es de desarrollo abierto	4

Con el objetivo de tener una referencia medible de cómo abordar las necesidades del cliente, se generó un conjunto de especificaciones. Como se observa en la tabla 4.2, se enlistan todas las características que debe cumplir la plataforma robótica, como grados de libertad, cantidad de brazos, sensores equipados, carga útil, entre otras. También, se especifica el campo de visión y la resolución que debe tener el sistema de visión solicitado por el cliente. De las especificaciones con mayor importancia en la tabla se

destacan las que permiten medir el desempeño de la plataforma robótica, en este caso la precisión en la pose del manipulador y la proporción de aciertos obtenidos a la hora de tomar objetos.

Tabla 4.2: Especificaciones objetivo concernientes al diseño robótico de la herramienta.

No.	Métrica	Unidad	Imp.	Valor marginal	Valor ideal
1	Grados de libertad por brazo	Cantidad	5	>5	>6
2	Número de brazos	Cantidad	3	1	2
3	Alcance manipulador	mm	4	>300	>500
4	Sensores equipados	Lista	5	-Sistema Visión -Sensores Absolutos Joints	-Sistema Visión -Sensores Absolutos Joints -Sensores Presencia Gripper -Sensores de Fuerza y Torque
5	Dimensiones máximas objeto a manipular	mm	4	>25	>50
6	Peso máximo objeto a manipular	g	3	>200	>500
7	Velocidad manipulador	m/s	4	>0.25	>1
8	Costo plataforma robótica	$	3	<8500	<6250
9	Clasificación movilidad	Ver Clasificación.	2	≥1	≥2
10	Precisión en la pose del manipulador	mm	5	<0.1	<0.05
11	Campo de visión	mm	4	>500x650	>1000x1300
12	Resolución sistema visión	mm	4	<1.5	<1
13	Aciertos en la toma de objetos	% aciertos/intentos	5	>95	100
22	Puntuación de apertura componentes Hardware	Ver Puntuación.	4	>4	>6
23	Puntuación de apertura módulos de Software	Puntuación OSS Watch.	4	>60	>75

Los valores para las especificaciones fueron discutidos con el cliente y se basaron en su gran mayoría en un estudio comparativo con diferentes plataformas robóticas existentes en el mercado. Este estudio puede encontrarse en la sección A.3.2 de la bitácora de diseño. Además, una discusión completa sobre el valor establecido a cada una de las especificaciones se muestra en la tabla A.14.

4.2 Selección del software

Debido a la naturaleza computacional de la plataforma robótica que se diseñó, un aspecto clave fue la selección de las herramientas de *software* apropiadas para su implementación. Como se mostró en la descomposición del proyecto de la figura 3.3 se requieren dos tipos principales de *software*: uno que se encargue de simular al robot y el ambiente, mientras que otro debe encargarse de realizar la interfaz entre los distintos módulos de procesamiento y la plataforma robótica.

Para realizar la integración entre los módulos que componen la herramienta se optó por utilizar un *middleware* de robótica, ya que ofrecen la funcionalidad requerida para la comunicación de diferentes módulos de *hardware* -en este caso simulado- y *software*. Se consideraron las opciones de middleware presentadas en la tabla 4.3.

Tabla 4.3: *Middlewares* considerados para la interfaz computacional.

Middleware	**Referencia**
Robotic Operating System (ROS)	[56]
Yet Another Robot Platform (YARP)	[57]
OpenJAUS	[58]
OpenRTM-aist	[59]

Se optó por seleccionar el uso de ROS basándose en los siguientes criterios:

- Permite establecer una red de comunicación basada en nodos que tiene mayores capacidades que las demás opciones consideradas, ya que permite el envío de información de forma sincrónica y asincrónica desde paquetes de software programados en distintos lenguajes (C++, Python, Lisp o Matlab).

- A diferencia de los demás conceptos estudiados, ROS ofrece una gran cantidad de herramientas adicionales útiles en robótica como interfaces de control y planeamiento de trayectorias.

- La arquitectura cognitiva MDB está desarrollada usando ROS, por lo que se facilitará el desarrollo de la integración con la plataforma robótica.

- Cuenta con una mayor disponibilidad de documentación y recursos de ayuda. Entre estos destacan: *ROS Wiki*, donde se alojan descripciones de los distintos paquetes disponibles y *ROS Answers*, donde la comunidad realiza preguntas y obtiene ayuda de otros usuarios.

- Es un software libre, por lo que no se debe incurrir en costos de licencias.

La selección de ROS se convirtió en una guía muy importante para el diseño de la herramienta, ya que toda su arquitectura se basó en el uso de este *middleware*. Por esta razón, se procuró que el resto del diseño utilizara en su mayor parte las herramientas disponibles de forma nativa para ROS.

Para realizar la simulación robótica se consideraron los siguientes paquetes de *software*: *Coppelia Sim* [60], *Gazebo* [51], *Simulink* [61] y *Webots* [62]. Para realizar la selección se utilizó la matriz de evaluación mostrada en la tabla 4.4. Como se observa, el simulador con mejor calificación fue *Gazebo*. El criterio que contribuyó en mayor medida a su selección fue su integración con ROS, ya que este es el simulador utilizado por defecto junto con el *middleware* y permite el uso de la mayoría herramientas de forma eficiente. Sin embargo, *Gazebo* también permite la simulación física de la dinámica robótica y de diversos sensores. Finalmente, *Gazebo* también es un software libre que no implica ningún costo de licenciado.

Tabla 4.4: Selección del programa utilizado para la simulación física. (En negrita se indica el concepto utilizado como referencia para cada criterio.)

		Gazebo		**Webots**		**CoppeliaSim**		**Simulink**	
Criterios de Selección	**Peso (%)**	**Calif.**	**Pond.**	**Calif.**	**Pond.**	**Calif.**	**Pond.**	**Calif.**	**Pond.**
Integración con ROS	40	5	2	4	1,6	**3**	1,2	3	1,2
Documentación disponible	20	5	1	3	0,6	**3**	0,6	2	0,4
Capacidad modelo físico	10	4	0,4	**3**	0,3	4	0,4	5	0,5
Disponibilidad de sensores	10	4	0,4	5	0,5	**3**	0,3	2	0,2
Apertura software	20	5	1	4	0,8	**3**	0,6	1	0,2
	Total	4,8		3,8		3,1		2,5	
	Lugar	1		2		3		4	
	¿Seleccionar?	Sí		No		No		No	

4.3 Definición de la morfología

A continuación se presenta el análisis realizado para seleccionar la plataforma robótica que se adaptara de mejor manera a las necesidades del cliente. Se realiza una selección desde el punto de vista morfológico, ya que se escogen las principales partes

y la forma en la que se distribuyen en el robot. Para este análisis se buscaron distintas soluciones para cada uno de los subproblemas descritos en la descomposición funcional de la figura A.2 de la bitácora de diseño. La búsqueda de soluciones se basó en la matriz de combinación presentada en la figura 4.1, donde se muestran las opciones consideradas para cada uno de los subproblemas.

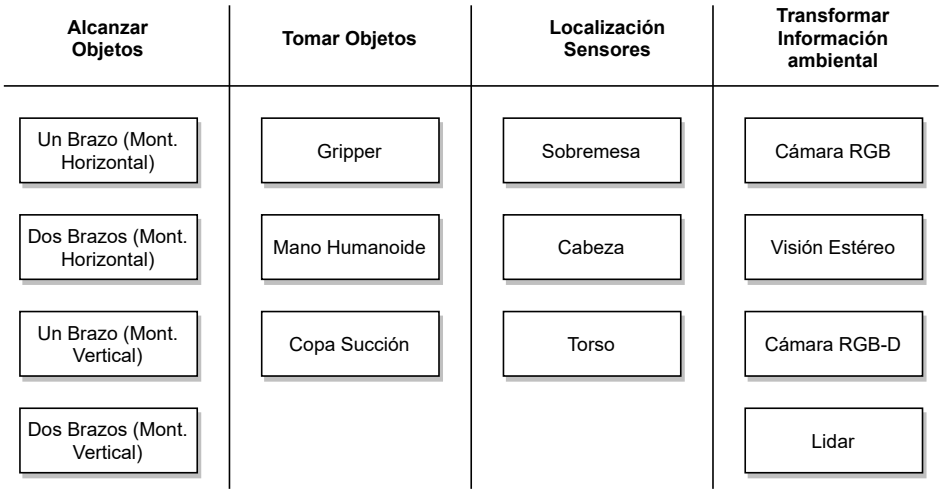


Figura 4.1: Matriz de combinación de fragmentos para la morfología robótica. Elaboración propia

A partir de una combinación sistemática se generaron 10 conceptos distintos, los cuales fueron filtrados, combinados y mejorados para generar los conceptos finalistas que se muestran en la figura 4.2. El proceso completo que llevó a la generación de los conceptos finalistas se puede consultar en la sección A.4.2.2 de la bitácora de diseño. Para seleccionar el concepto a desarrollar se utilizó la matriz de evaluación presentada en la tabla 4.5. De la evaluación no se observó una diferencia amplia entre las opciones, sin embargo se seleccionó el concepto 3+, ya que es una opción que se desempeña bien en la mayoría de criterios.

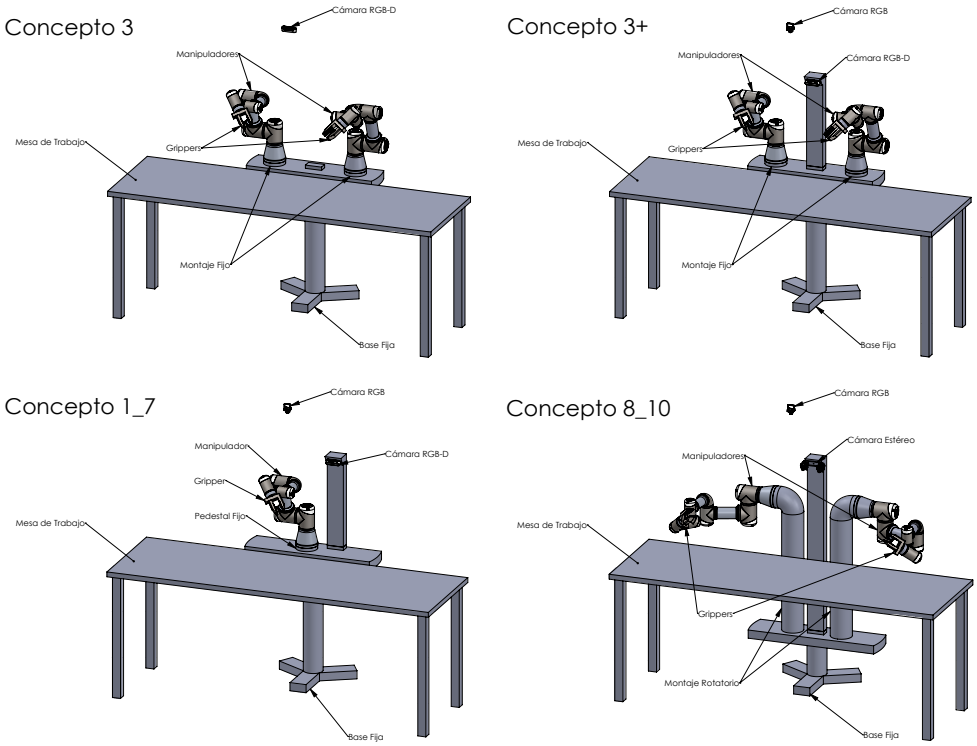


Figura 4.2: Conceptos evaluados para la morfología de la plataforma robótica. Elaboración propia: SolidWorks

4.4 Selección del manipulador

Uno de los componentes más importantes para la plataforma robótica es el manipulador, por esta razón es crítico realizar su diseño de manera apropiada. Sin embargo, debido a la complejidad inherente a un manipulador robótico, se optó por seleccionar un manipulador disponible comercialmente o de *hardware* libre. Por esta razón, se realizó una búsqueda amplia de manipuladores de pequeña escala disponibles en el mercado y se realizó una preselección de aquellos que cumplieran las especificaciones planteadas para la herramienta y que fuesen de costo razonable. Los resultados de esta preselección se muestran en la tabla 4.6, se puede consultar la totalidad de manipuladores considerados y sus documentos de referencia en la bitácora de diseño (tabla A.15).

Para realizar la selección del manipulador a utilizar se realizó la matriz de

Tabla 4.5: Matriz de evaluación de conceptos para la plataforma robótica.

		Concepto 1_7		**Concepto 8_10**		**Concepto 3**		**Concepto 3+**	
Criterio de Selección	**Peso (%)**	**Calif.**	**Pond.**	**Calif.**	**Pond.**	**Calif.**	**Pond.**	**Calif.**	**Pond.**
Morfología Humanoide	15	1	0,15	5	0,75	2	0,3	**3**	0,45
Capacidad de Manipulación	25	2	0,5	**3**	0,75	3	0,75	3	0,75
Obtención de datos	25	4	1	**3**	0,75	2	0,5	4	1
Simplicidad Control	10	4	0,4	2	0,2	**3**	0,3	3	0,3
Simplicidad Procesamiento de Datos	10	3	0,3	2	0,2	4	0,4	**3**	0,3
Costo	15	4	0,6	1	0,15	**3**	0,45	2	0,3
	Total	2,95		2,8		2,7		3,1	
	Lugar	2		3		4		1	
	¿Continuar?	No		No		No		Desarrollar	

Tabla 4.6: Manipuladores preseleccionados para la evaluación.

No.	**Nombre**	**Grados Libertad**	**Alcance (mm)**	**Carga (g)**	**Actuación/ Sensorización**	**Paquete ROS**	**Calif. *Open Hardware***	**Costo ($)**
1	Thor	6	422.5	750	Motores a pasos Finales de Carrera	No	5	420
2	Niryo One	6	440	300	Servomotores Finales de carrera Encoders Absolutos	Sí	5	2000
3	WidowX 6DoF	6	650	250	Servomotores Encoders Absolutos	Sí	2	2895
4	AR3	6	629	1900	Motores a pasos Finales de carrera Encoders Incrementales	No	3	2000

evaluación mostrada en la tabla 4.7. Como criterios de selección se tomaron en cuenta aspectos propios del desempeño de los brazos como destreza, espacio de trabajo, capacidad de carga y sensorización. Sin embargo, el criterio más significativo en la decisión correspondió a la disponibilidad de los modelos y documentación del manipulador, evaluado en el criterio *Apertura hardware y software*. Aunque la implementación a realizar sea simulada, se procuró enfocar la búsqueda a manipuladores de bajo costo que aumenten la viabilidad de una futura implementación física, por esta razón el criterio de costo tiene un peso alto. Por otro

lado, se le dio importancia a que el manipulador seleccionado contara con un paquete para integrarse con ROS de manera que facilitara el desarrollo de la implementación simulada.

Tabla 4.7: Tabla de evaluación de los manipuladores.

		Thor		**Niryo One**		**WidowX 6DoF**		**AR3**	
Criterios de Selección	**Peso (%)**	**Calif.**	**Pond.**	**Calif.**	**Pond.**	**Calif.**	**Pond.**	**Calif.**	**Pond.**
Apertura hardware y software	25	5	1,25	4	1	2	0,5	**3**	0,75
Capacidad de carga	10	4	0,4	**3**	0,3	2	0,2	5	0,5
Compatibilidad ROS	15	2	0,3	**3**	0,45	5	0,75	2	0,3
Costo	20	5	1	**3**	0,6	1	0,2	3	0,6
Destreza y Precisión	10	2	0,2	**3**	0,3	4	0,4	4	0,4
Espacio de trabajo	10	2	0,2	**3**	0,3	5	0,5	4	0,4
Sensorización	10	1	0,1	2	0,2	5	0,5	**3**	0,3
	Total	3,45		3,15		3,05		3,25	
	Lugar	1		3		4		2	
	¿Seleccionar?	Sí		No		No		No	

La evaluación se realizó basándose en los datos resumidos en la tabla 4.6, y de la calificación ponderada se observa que el manipulador mejor calificado fue *Thor*. Este manipulador destacó en los criterios de mayor peso en la selección, como costo y apertura del *hardware*. Esto permitió compensar sus menores prestaciones en cuanto a espacio de trabajo y sensorización. Este manipulador pareció ser una buena opción debido a que se encuentra dentro de los valores establecidos para la mayoría de especificaciones, la mayor desventaja que posee es la falta de sensores para la medición en las articulaciones. Sin embargo, esto no es un problema inmediato para la implementación simulada y al ser de hardware abierto se cuenta con todos los archivos CAD que permitirían una adaptación si llega a ser necesario en la implementación real.

El segundo clasificado es el robot *AR3* el cual cuenta con una buena relación entre costo y prestaciones. No fue seleccionado debido a que los modelos CAD requeridos para realizar la implementación simulada debían ser adquiridos bajo un costo monetario, lo que disminuyó su calificación en el criterio de apertura de hardware. A pesar de esto, es una opción a considerar para una futura implementación física.

4.5 Implementación de la plataforma robótica

La implementación de la plataforma robótica se resume en el diagrama de bloques presentado en la figura 4.3. En el diagrama se pueden observar los principales nodos de ROS que conforman la plataforma robótica y las interfaces de comunicación que utilizan.

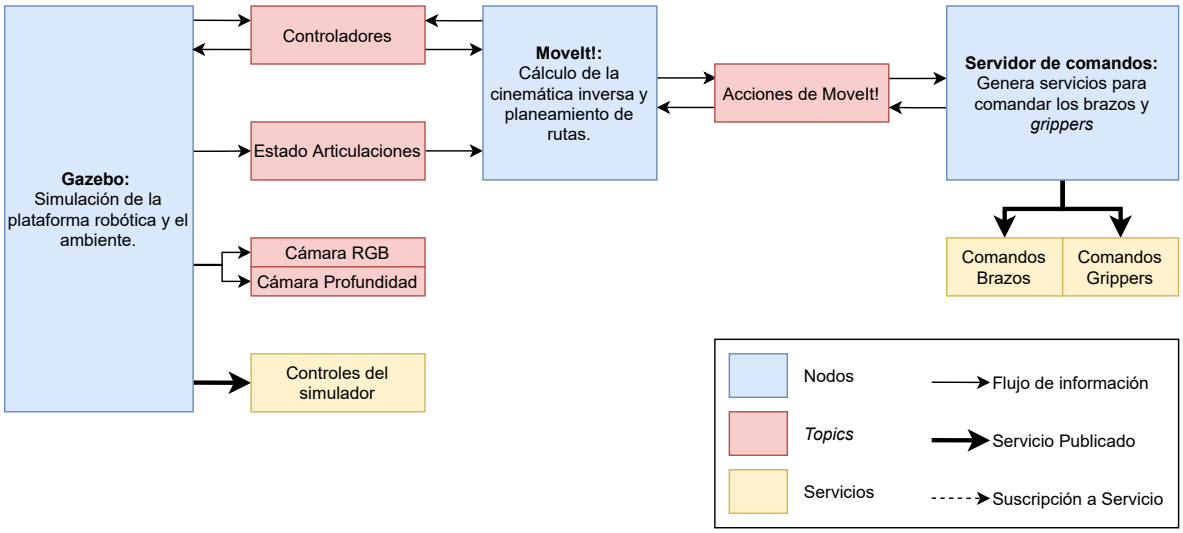


Figura 4.3: Diagrama de bloques de la plataforma robótica. Elaboración propia

Gazebo es el nodo principal de la plataforma, ya que en este se genera la mayoría de información utilizada. Desde el simulador se publican en tiempo real los controladores utilizados, el estado de las articulaciones y las imágenes de las cámaras RGB y RGB-D. Adicionalmente, el simulador publica servicios que permiten controlar la simulación. Estos servicios permiten modificar el estado de cualquier modelo del ambiente y las propiedades de la simulación, como la cantidad de iteraciones por paso, y la velocidad de la simulación respecto al tiempo real.

MoveIt es un paquete de planificación de rutas para manipuladores integrado completamente con ROS [63]. El nodo de *MoveIt* se comunica con *Gazebo* a través de los mensajes de los controladores y el estado de las articulaciones. Con estos datos es capaz de ejecutar las rutas planeadas, las cuales se solicitan a través de un conjunto de acciones publicadas por el nodo. Estas acciones son las que utiliza el nodo del

servidor de comandos para indicarle a *MoveIt* cuáles rutas debe planear y ejecutar. Finalmente, el servidor de comandos para los brazos y *grippers* permite que el usuario del robot controle la plataforma mediante un *Application Programming Interface* (API) sencillo.

Los modelos del manipulador *Thor* se adaptaron para su uso en *Gazebo* utilizando *SolidWorks*, que permite exportar modelos directamente al fomato requerido por el simulador. Un proceso similar se utilizó para generar un *gripper* apropiado para las tareas de manipulación. El *gripper* tiene una apertura máxima de 50 mm, lo que permite cumplir la especificación sobre el tamaño máximo del elemento. Los *links* que componen al manipulador y *gripper* se muestran en las figuras 4.4 y 4.5. El montaje general de la plataforma robótica se muestra en la figura 4.6, donde también se muestra el sistema de coordenadas de referencia utilizado.

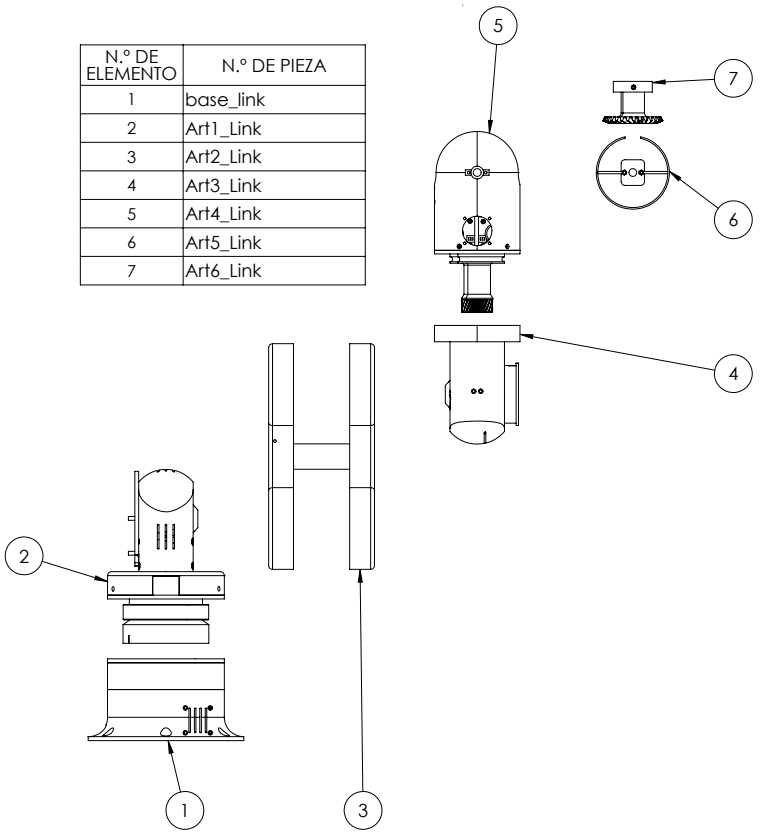


Figura 4.4: *Links* que componen al manipulador simulado. Elaboración propia: SolidWorks

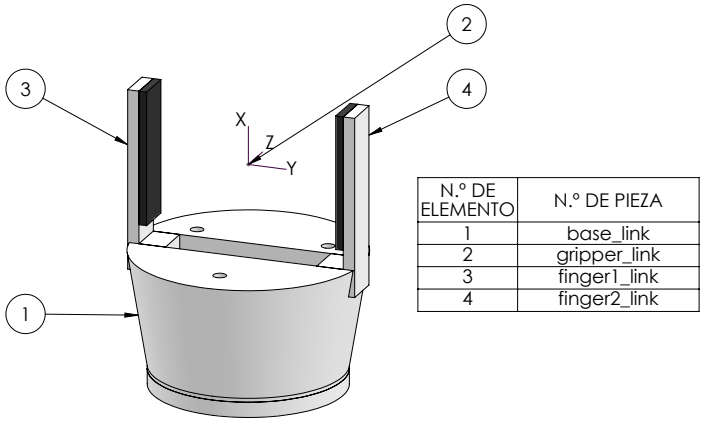


Figura 4.5: *Links* que componen al *gripper*. Elaboración propia: SolidWorks

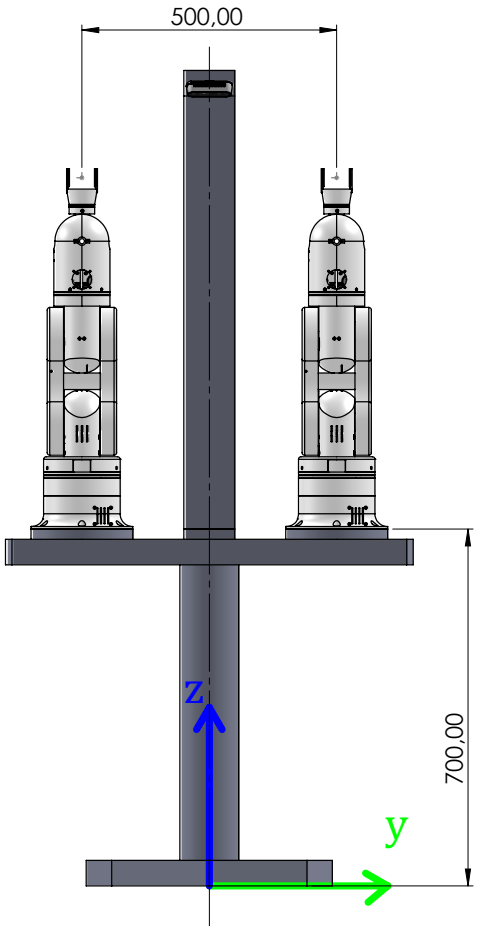


Figura 4.6: Montaje de los manipuladores. Elaboración propia: SolidWorks

4.5.1 Control de los manipuladores

Para permitir el movimiento de los modelos robóticos en el simulador *Gazebo*, se definió una serie de articulaciones que permiten la rotación y desplazamiento. En el caso del manipulador, se utilizaron las articulaciones rotacionales definidas en la tabla 4.8. En el caso del gripper se definieron las articulaciones prismáticas presentadas en la tabla 4.9.

Tabla 4.8: Propiedades definidas para los *joints* del manipulador

No.	Nombre	Padre	Hijo	Límite Inferior (°)	Límite Superior (°)
1	Art1_Yaw	base_link	Art1_link	-90	90
2	Art2_Pitch	Art1_link	Art2_link	-90	90
3	Art3_Pitch	Art2_link	Art3_link	-90	90
4	Art4_Yaw	Art3_link	Art4_link	-120	120
5	Art5_Pitch	Art4_link	Art5_link	-90	90
6	Art6_Yaw	Art5_link	Art6_link	$-\infty$	$+\infty$

Tabla 4.9: Propiedades definidas para los *joints* del gripper.

No.	Nombre	Padre	Hijo	Límite Inferior (mm)	Límite Superior (mm)
1	finger1_prismatic	gripper_link	finger1_link	0	20
2	finger2_prismatic	gripper_link	finger2_link	0	20

Para controlar el brazo robótico se hizo uso del paquete *ros_control* que permite el control de robots de manera simple y genérica mediante la red de comunicación provista por ROS [64]. Se utilizó la estrategia de control independiente para cada articulación del robot, ya que de esta forma se aprovecha la interfaz propuesta por *ros_control*, que permite generar controladores PID independientes para cada articulación.

El uso de controladores PID trae consigo la necesidad de calibrar las ganancias para obtener un funcionamiento adecuado. Esto se llevó a cabo mediante un proceso empírico, de manera que se evitó desarrollar un modelo analítico del manipulador y se agilizó el desarrollo del controlador. Por supuesto, esta decisión disminuyó la capacidad

de realizar un control robusto sobre el manipulador, sin embargo, como no se conoce el grado de correspondencia que tendrá el modelo simulado con el mundo real, no se consideró que el desarrollo de una estrategia de control avanzada estuviera dentro del alcance de este proyecto.

La calibración permitió el ajuste de características adecuadas del comportamiento de las articulaciones. Se buscó obtener un tiempo de estabilización aproximado de 2 segundos, sobreimpulso aproximado a cero y error de estado estacionario aproximado a cero. Del proceso de calibración surgieron dos controladores que se presentan en la tabla 4.10. El controlador 0 fue calibrado con el manipulador sin carga, mientras que el controlador 1 se calibró mientras el brazo sostenía una carga de 500g sujetada de forma rígida al manipulador. Los controladores asociados al gripper se calibraron de forma separada, por esta razón son iguales en ambas configuraciones.

Todos los controladores configurados consideran un término de saturación integral para evitar torques y fuerzas excesivas en caso de colisiones con el ambiente. En el caso de los *grippers* el límite integral permitió definir la fuerza de sujeción máxima. El desempeño de ambos controladores se comparará más adelante en el experimento de caracterización del manipulador.

Tabla 4.10: Valores de las constantes de los controladores PID.

		Configuración 0				**Configuración 1**			
No.	**Articulación**	**P**	**I**	**D**	**Límite I**	**P**	**I**	**D**	**Límite I**
1	Art1_Yaw	100	1	1	100	100	0	5	*
2	Art2_Pitch	300	3.5	1	100	2000	1000	5	7
3	Art3_Pitch	150	1	1	100	100	1000	5	7
4	Art4_Yaw	100	1	0	1	200	10	1	1
5	Art5_Pitch	100	1	0	100	20	10	0	5
6	Art6_Yaw	10	0.1	0	100	100	10	0	10
7	finger1_prismatic	300	10	0.1	0.25	300	10	0.1	0.25
8	finger2_prismatic	300	10	0.1	0.25	300	10	0.1	0.25

4.5.2 Servidor de comandos

Con el objetivo de facilitar el uso de la plataforma robótica de una forma modular, se definió un nodo que permite abstraer los comandos de movimiento. Esto se realizó mediante el uso de servicios de ROS que utilizan el API de *MoveIt*.

Para los comandos de los manipuladores se estableció el mensaje mostrado en el *listing* 4.1, como solicitud se envían las coordenadas cartesianas de la posición que se quiere alcanzar y la velocidad de movimiento. Opcionalmente, se puede enviar un *string* donde se indica el nombre de una pose definida, esto hace que el manipulador se dirija directamente a esa pose. El servicio responde dos valores booleanos, en el campo *plan*, se indica si se encontró una ruta para obtener la posición deseada, el campo *execute* indica si la ejecución del movimiento fue correcta.

Listing 4.1: Mensaje del servicio de control de los brazos.

```
float32 x
float32 y
float32 z
float32 vel
string named_pose
---
bool plan
bool execute
```

El servicio de comando de los brazos tiene la función adicional de calcular automáticamente la orientación 3D que debe tener el *gripper* de acuerdo a la posición solicitada. Para esto se realiza el cálculo de los ángulos de Euler (α, β, γ) que definen la rotación del marco de referencia *gripper_link* (ver figura 4.5) respecto al marco de referencia global que se encuentra en la base del robot. El ángulo α o *roll* se mantiene fijo en cero para que las garras del gripper se mantengan a los lados del objeto a tomar. El ángulo γ o *yaw* calcula según el brazo que se comanda, en caso del brazo izquierdo con la ecuación 4.1 y para el brazo derecho la ecuación 4.2. El ángulo *yaw* se calcula de esa manera para que todo el brazo gire (utilizando la articulación *Art1_Yaw*) para orientarse en dirección a la posición deseada. El ángulo β o *pitch* se define dentro del rango $[0, \pi/2]$. Cuando $\beta = \pi/2$ el gripper se orienta verticalmente

hacia abajo, mientras que si $\beta = 0$ el *gripper* se coloca en posición horizontal. Para escoger el ángulo *pitch* se calculan rutas de forma iterativa comenzando en $\beta = \pi/2$ hasta que se encuentre una ruta válida.

$$\gamma = \arctan \frac{y - 0.25}{x} \tag{4.1}$$

$$\gamma = \arctan \frac{y + 0.25}{x} \tag{4.2}$$

El mensaje de servicio definido para el comando de los grippers es más sencillo, la solicitud es un valor booleano que indica si se debe cerrar el gripper. El servicio responde un booleano llamado *object*, este tiene un valor de *true* cuando se detecta que el *gripper* no pudo cerrarse completamente, lo que implica que hay un objeto sujetado. El archivo del servicio definido puede observarse en el *listing* 4.2.

Listing 4.2: Mensaje del servicio de control del *gripper*.

```
1 bool close
2 ---
3 bool object
```

4.5.3 Ambiente simulado

Para permitir la ejecución del experimento de aprendizaje, se definió un ambiente simulado en Gazebo que genera los objetos necesarios. El ambiente incluye: una mesa de trabajo, el objeto a manipular, la canasta y el botón de ayuda. En la figura 4.7 se muestra el ambiente generado junto con la plataforma robótica.

Como mesa de trabajo, se utiliza un plano inmóvil con dimensiones de 2x0.65 m y 30 mm de espesor, que está configurado para permitir colisiones con el resto de objetos. La canasta tiene dimensiones de 100x100 mm con una altura de 50 mm, la apertura de la canasta es de 94x94 mm. Como objetos a manipular se definieron dos geometrías, un cubo de 25x25x25 mm y un cilindro de 25 mm de diámetro y altura. El botón es

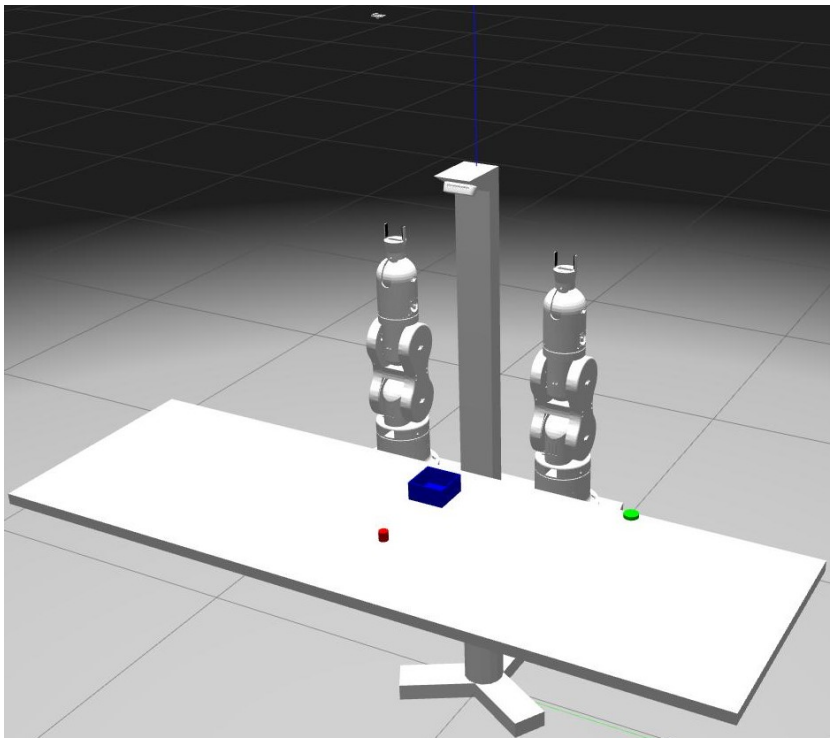


Figura 4.7: Ambiente simulado. Elaboración propia: Gazebo

un cilindro de 40 mm de diámetro y 10 mm de altura, este objeto se coloca de forma fija y no genera colisiones con el resto de objetos. Los colores utilizados en los objetos tienen el objetivo de facilitar la identificación visual de los mismos, por esta razón, se utilizaron los colores verde, azul y rojo que permiten la segmentación mediante los canales RGB de las cámaras utilizadas.

Debido a que la tarea se realizará mediante una simulación física que busca la manipulación realista de objetos, las propiedades físicas de cada uno de los modelos son claves para el correcto funcionamiento de la simulación. Entre las propiedades físicas, los parámetros correspondientes a la simulación de colisiones son críticos. En la tabla 4.11 se muestran las propiedades físicas para los objetos que componen el ambiente.

Los objetos a manipular requirieron una cuidadosa selección de sus parámetros de colisión. Estos fueron calibrados para asegurar un correcto agarre de los grippers, para esto, se utilizó un coeficiente de fricción alto y se tomó un valor del coeficiente de rigidez

Tabla 4.11: Propiedades de los objetos del ambiente.

No.	Nombre	Masa (g)	Coeficiente fricción	Coeficiente k_p	Coeficiente k_d
1	basket	84.7	1	1E+12	1
2	button	No aplica	No aplica	No aplica	No aplica
3	red_box	50	10	5000	1
4	red_cylinder	50	10	5000	1
5	table	No aplica	0.1	1E+12	1

de contacto (k_p) bajo. El coeficiente de amortiguamiento de contacto (k_d) se mantuvo en su valor por defecto. Esta combinación de propiedades permite que los objetos se comporten de forma blanda al ser sujetados por el gripper, lo que ayuda a generar una mayor cantidad de puntos de contacto y que estos sean estables. En cuanto a la mesa y la canasta, los valores establecidos por defecto resultaron en un comportamiento adecuado, sin embargo, en el caso de la mesa se disminuyó el coeficiente de fricción para simular una superficie lisa.

4.5.4 Sistema de visión

El componente más importante del sistema sensorial de robot es su sistema de visión. Como se definió en el estudio conceptual, este cuenta con dos cámaras: una cámara RGB colocada en posición cenital sobre la mesa y una cámara RGB-D que se coloca en la "cabeza" del robot.

4.5.4.1 Visión 2D

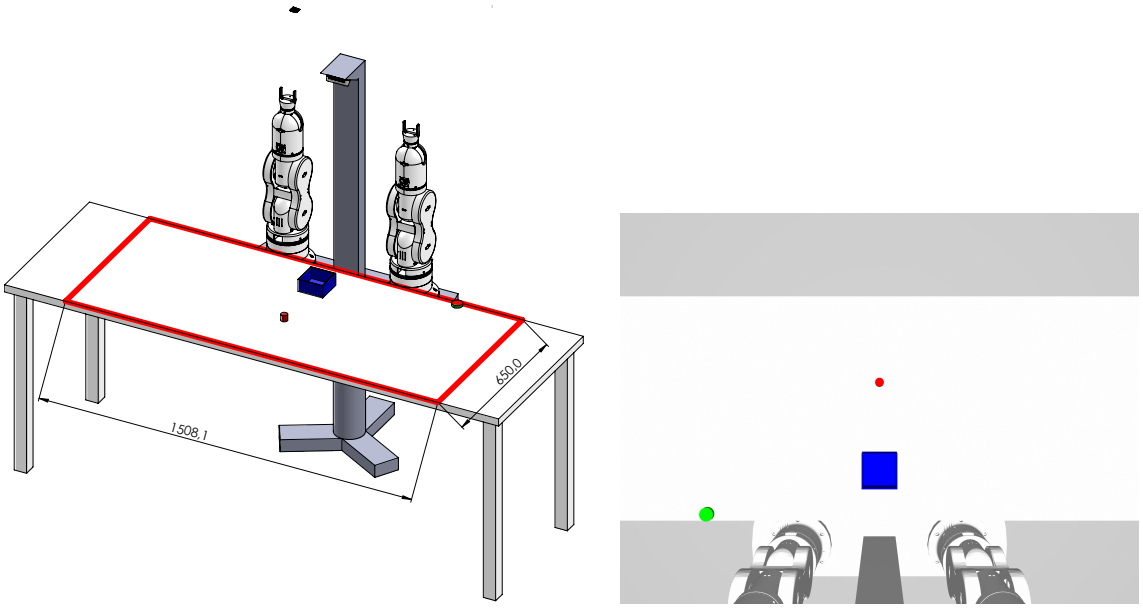
En el caso de la cámara RGB se estudiaron los módulos de visión disponibles para el uso con Raspberry Pi [65], que se muestran en la tabla 4.12. De estos módulos se escogió simular el RPI v2, debido a que es una cámara de alta resolución y bajo costo que podría adaptarse a una implementación física futura. Es importante destacar que existe una variedad muy alta de cámaras y ópticas disponibles, y el objetivo del estudio actual no fue encontrar la cámara más apropiada para la implementación física, sino,

tener una referencia inicial.

Tabla 4.12: Módulos de visión para Raspberry Pi disponibles.

No.	Módulo	Resolución	Distancia Focal	Costo ($)
1	RPI v1	2592 x 1944	3.60 mm	25
2	RPI v2	3280 x 2464	3.04 mm	25
3	RPI HQ	4056 x 3040	Depende del lente	50

La cámara se colocó a una altura de 1.25 m sobre el centro de la mesa de trabajo, esta altura permitió obtener un campo de visión amplio, mostrado en la figura 4.8. Como se observa, el campo de visión no cubre la mesa completa, se seleccionó así para no disminuir excesivamente la resolución, específicamente en el eje Y (de arriba hacia abajo en la imagen).



a) b)

Figura 4.8: Campo de visión de la cámara RGB. a) Dimensiones, b) Foto de la cámara. Elaboración propia: SolidWorks/rqt

Para estimar la resolución del sistema de visión se calcula una relación entre el campo de visión y el número de píxeles. Es importante destacar que en la práctica, efectos de la óptica e iluminación pueden causar que la resolución práctica del sistema de visión sea menor, sin embargo, desde la simulación actual estos efectos no pueden

ser estimados. El cálculo de la resolución para cada uno de los ejes de la imagen se muestra en las ecuaciones 4.3 y 4.4. En este caso, la resolución obtenida supera de manera satisfactoria el valor establecido en las especificaciones de la tabla 4.2.

$$\text{Resolución}_X = \frac{1508,1 \text{ mm}}{3280 \text{ px}} = 0.46 \frac{\text{mm}}{\text{px}} \tag{4.3}$$

$$\text{Resolución}_Y = \frac{650 \text{ mm}}{1415 \text{ px}} = 0.46 \frac{\text{mm}}{\text{px}} \tag{4.4}$$

4.5.4.2 Visión 3D

En el caso de las cámaras RGB-D, se estudiaron las dos opciones más importantes que existen actualmente en el mercado, las cámaras *Realsense* de Intel [66] y el *Kinect* de Microsoft [67]. Las características de algunas de las cámaras de estas dos familias se muestran en la tabla 4.13.

Tabla 4.13: Algunas cámaras RGB-D disponibles en el mercado.

No.	Módulo	Resolución	Rango operación (m)	Ángulo de visión (°)	Costo ($)
1	Realsense D415	1280 x 720	0.5 - 3	65 x 40	159
2	Realsense D435	1280 x 720	0.3 - 3	69 x 42	189
3	Realsense D455	1280 x 720	0.6 - 6	87 x 58	249
4	Azure Kinect DK	1024 x 1024	0.25 - 2.21	120 x 120	869

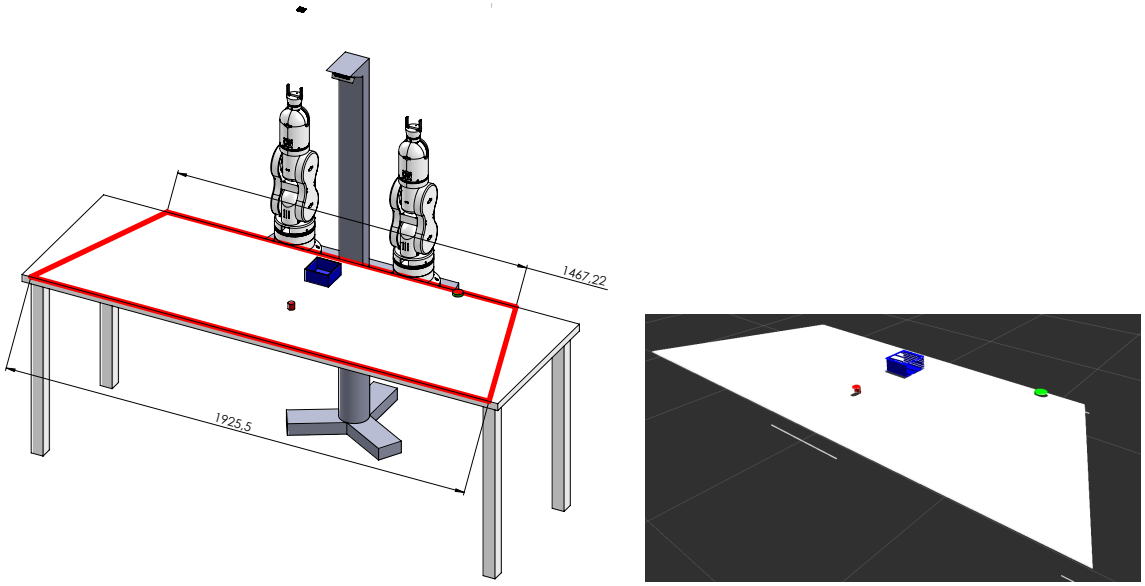
Para la implementación simulada se optó por utilizar la cámara *Realsense* D435, esto debido a que cuenta con rango de operación y ángulo de visión amplios que se adaptan a la distancia de trabajo de los objetos utilizados, además cuenta con un costo moderado. Esta cámara fue colocada a una altura de 770 mm sobre el nivel de la mesa de trabajo y se le dio una inclinación de 61,5 grados para que el borde inferior de la imagen coincidiera con el borde de la mesa de trabajo.

Las dimensiones del campo de visión generado por la cámara de profundidad se muestran en la figura 4.9. Para calcular la resolución del sistema se utilizó el mismo procedimiento que para la cámara RGB, y en las ecuaciones 4.5 y 4.6 se muestra el

resultado obtenido. Como se observa, esta cámara está justo en el mínimo de resolución aceptable de acuerdo con la especificación definida, esto es esperable debido al menor tamaño de imagen que dispone la cámara de profundidad.

$$\text{Resolución}_X = \frac{1925.5 \text{ mm}}{1280 \text{ px}} = 1.50 \frac{\text{mm}}{\text{px}} \tag{4.5}$$

$$\text{Resolución}_Y = \frac{650 \text{ mm}}{502 \text{ px}} = 1.29 \frac{\text{mm}}{\text{px}} \tag{4.6}$$



a) b)

Figura 4.9: Campo de visión de la cámara RGB-D. a) Dimensiones, b) Nube de puntos generada. Elaboración propia: SolidWorks/rviz

4.6 Resultados Experimentales

4.6.1 Caracterización de manipuladores

Este experimento tuvo como objetivo caracterizar los manipuladores de la plataforma robótica. En la figura 4.10 se muestran las condiciones generales del experimento. Para cada uno de los manipuladores se estableció una serie de puntos

que debe alcanzar con el efector final. En cada punto se midió el error de posición como la distancia euclidiana entre la posición deseada y la posición final. Como puntos de prueba se definió un cubo que abarca las posiciones: $x = [0.1, 0.625]$ m, $y = [-0.875, 0.875]$ m, $z = [0.8, 1.325]$. En los ejes x y z se dividió el intervalo en 11 segmentos mientras que en el eje y se dividió en 13 segmentos, por lo tanto, en cada corrida experimental se exploraron 1573 puntos con cada brazo.

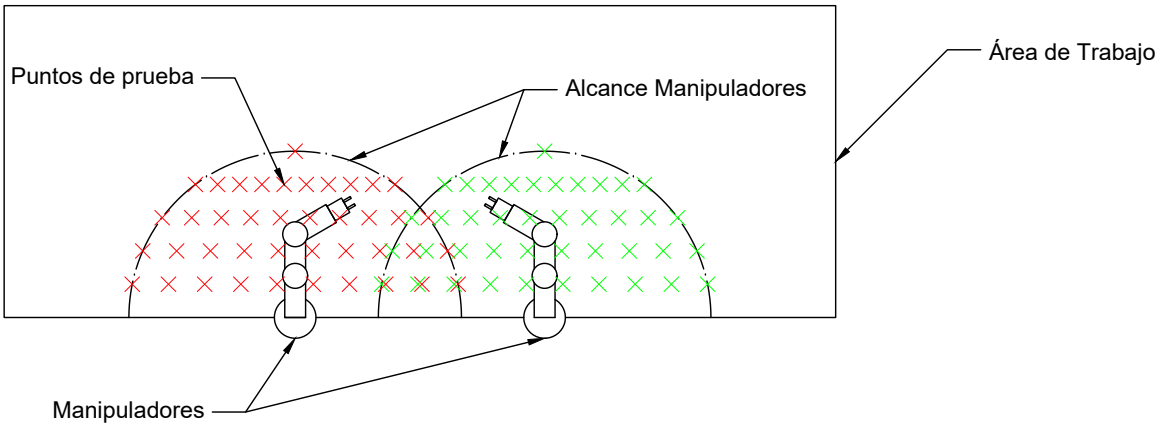


Figura 4.10: Diagrama experimento de caracterización de manipuladores. Elaboración propia: AutoCAD

El experimento fue de tipo factorial completo y se ejecutó una única replica para cada condición experimental. En la tabla 4.14 se muestran los factores y niveles definidos. El diseño experimental generó un total de ocho corridas, una para cada combinación de factores. Las corridas se ejecutaron en orden aleatorio. Las variables de respuesta estudiadas fueron el promedio y la desviación estándar del error de distancia en cada condición experimental.

Tabla 4.14: Factores y niveles del experimento de caracterización de manipuladores.

Factor	Valor Bajo	Valor Alto
Parámetros Controlador	Configuración 0	Configuración 1
Velocidad	50 %	100 %
Carga	50 g	200 g

En la figura 4.11 se muestra un gráfico de cajas con el resultado del error obtenido

para cada una de las corridas experimentales. Del gráfico se puede observar que hay una amplia diferencia entre el error promedio para las distintas configuraciones del controlador. En el diagrama de Pareto de la figura 4.12 obtenido mediante la técnica ANOVA, se confirma que la configuración utilizada en el PID genera el mayor efecto en el error promedio obtenido. De la misma manera, se puede identificar que la carga sostenida por el brazo tiene un efecto significativo en el error.

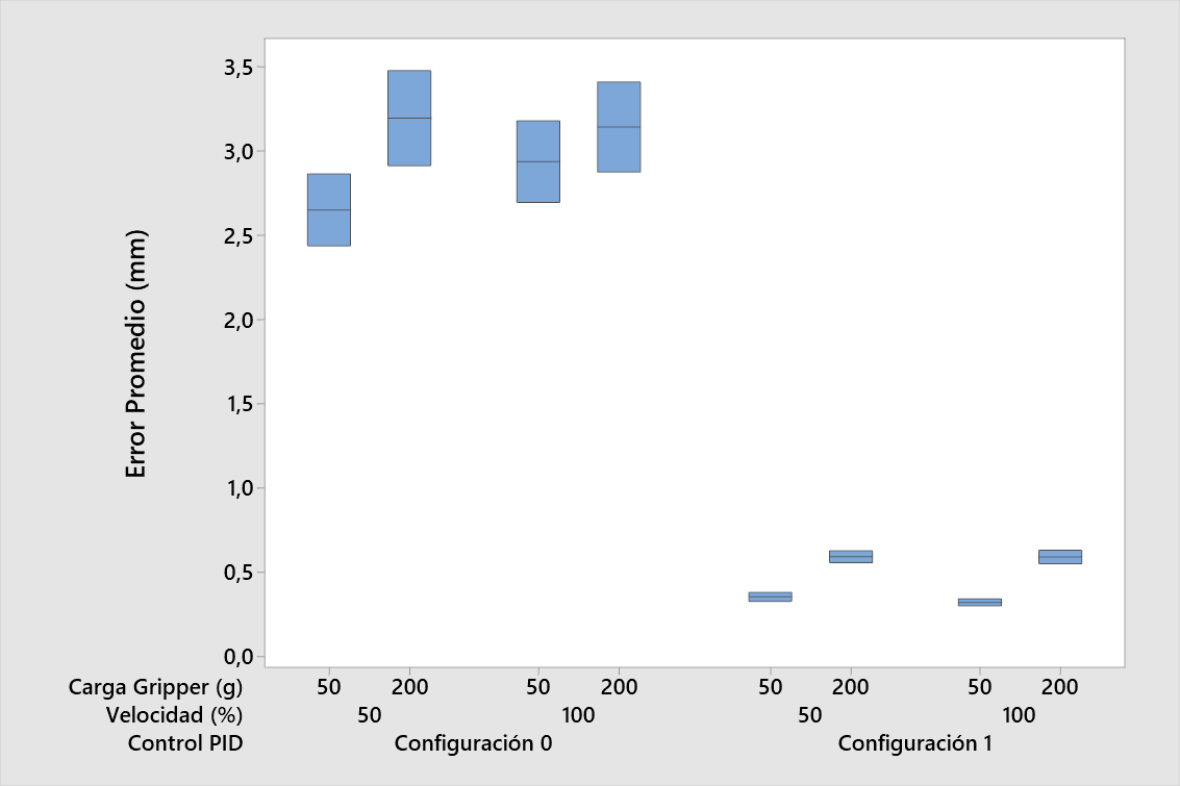


Figura 4.11: Gráfico de cajas del error de posición. Elaboración propia: Minitab

Al utilizar el controlador PID con la configuración 1 de la tabla 4.10, se logra minimizar el error de distancia en el manipulador de manera que no supera el valor de 1 mm. Aunque exista un efecto significativo respecto a la carga del manipulador, se observa que con los valores de carga utilizados, el error se mantiene en un rango aceptable. Se puede afirmar que los manipuladores son robustos con respecto al efecto de las distintas velocidades de operación, esto debido a que no se identificó un efecto significativo de ese factor. En general, el uso del controlador en la configuración 1

permite satisfacer las especificaciones buscadas para el desempeño de los manipuladores.

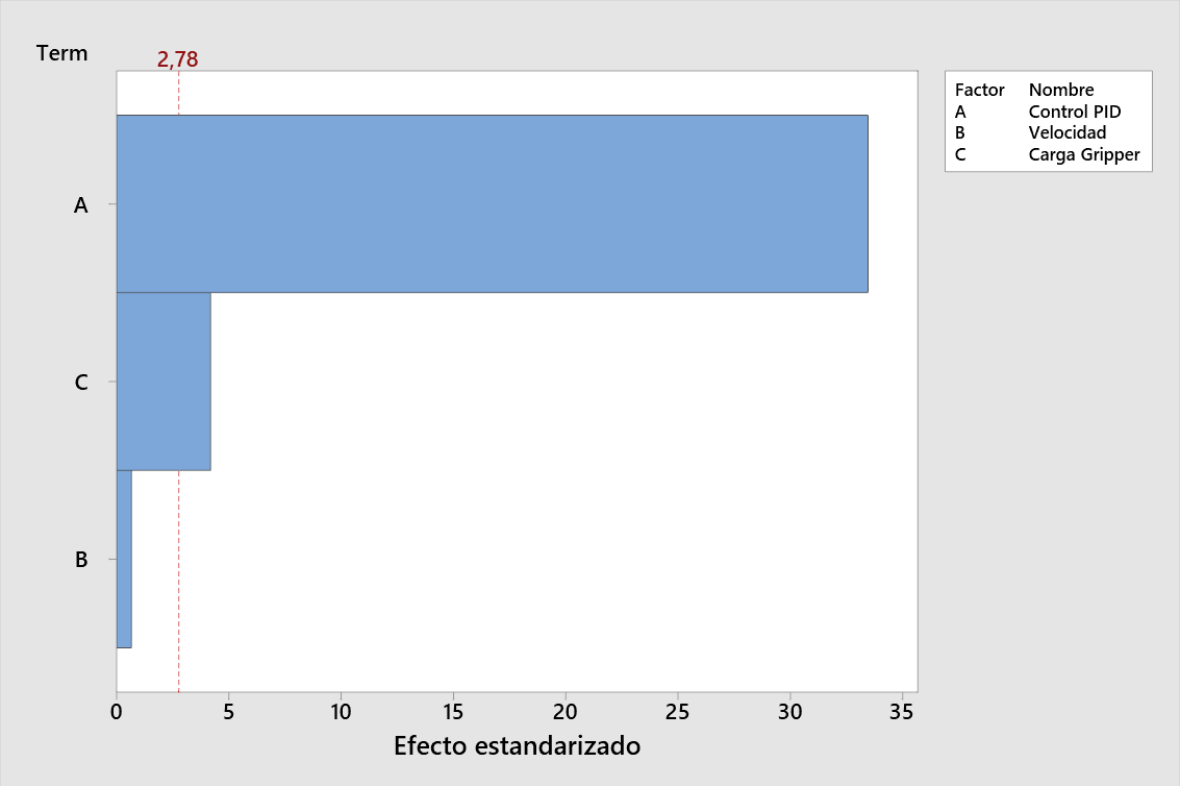


Figura 4.12: Diagrama de Pareto de los efectos principales del experimento de caracterización. Elaboración propia: Minitab

En la figura 4.13 se muestran los puntos alcanzados por cada brazo a una altura de 125 mm sobre la superficie de la mesa. A esta altura el manipulador cuenta con el espacio de trabajo de mayor extensión. Durante todo el experimento se utilizó el servidor de comandos desarrollado para los manipuladores, por lo que en todos los puntos el efector final está orientado hacia la mesa. El alcance máximo de los manipuladores en el eje x es de 467.5 mm, que se alcanza en puntos directamente al frente de cada uno de los brazos. Se observa una marcada simetría entre el espacio de trabajo de cada uno de los brazos y además, existe una "zona compartida" centrada en el eje y donde ambos brazos tienen alcance.

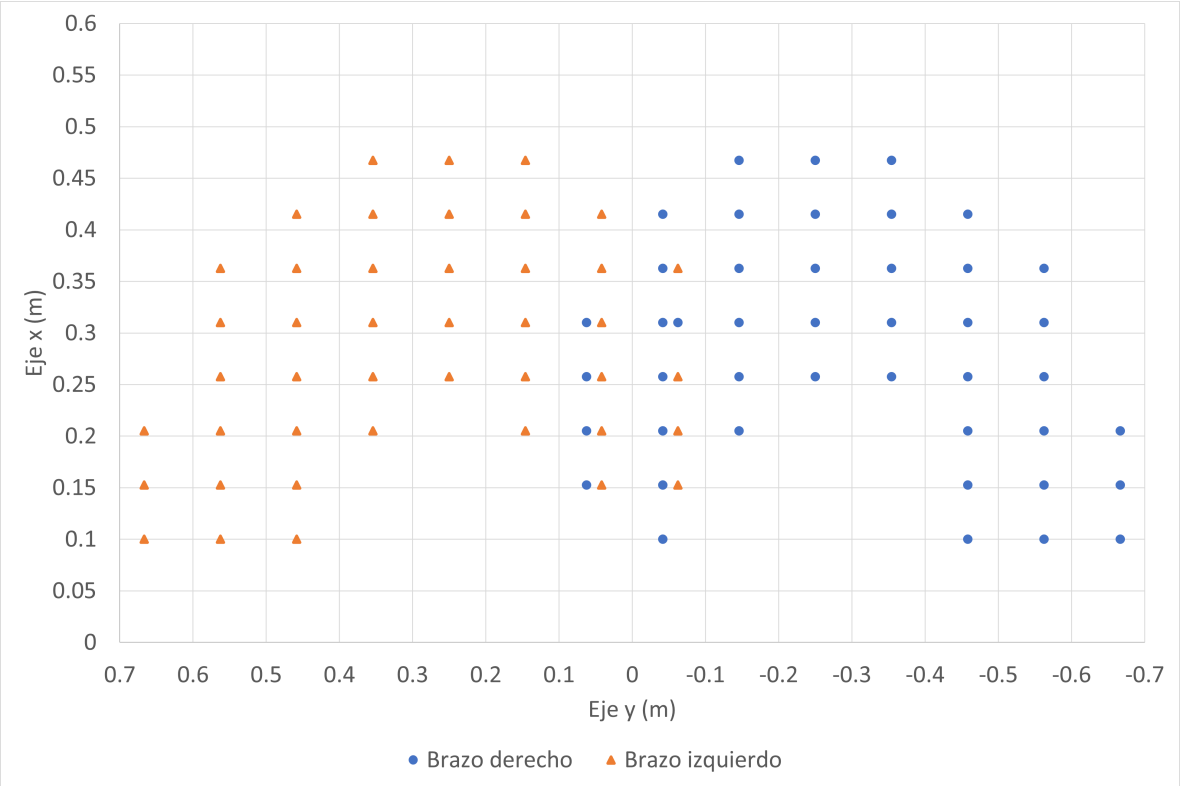


Figura 4.13: Espacio de trabajo del manipulador a $z = 0.905$ m. Elaboración propia: Microsoft Excel

4.6.2 Toma de objetos

El objetivo de este experimento es determinar el nivel de confiabilidad para operaciones básicas de *pick and place*. En la figura 4.14 se muestran las condiciones generales del experimento, en cada corrida experimental se colocó un objeto en varias posiciones del espacio de trabajo de cada manipulador y se comandó al manipulador a tomarlo y depositarlo en una canasta en la parte central del área de trabajo. La operación contó como un acierto si al final de la secuencia el objeto se encuentra dentro de la canasta. En este experimento no se incluyó el sistema perceptual del robot, los comandos dados al manipulador fueron la posición exacta de cada objeto.

Para realizar el experimento se tomaron como posiciones de prueba los puntos encontrados en el espacio de trabajo a una altura de $z = 0.8$ m en el experimento anterior, debido a que a esta altura el gripper puede tomar los objetos. En cada

corrida experimental cada brazo realizó la operación de *pick and place* para 27 puntos distintos de su espacio de trabajo. La canasta se colocó en la posición (0.15, 0, 0.78) m, de manera que ambos brazos pudieran depositar el objeto en la canasta.

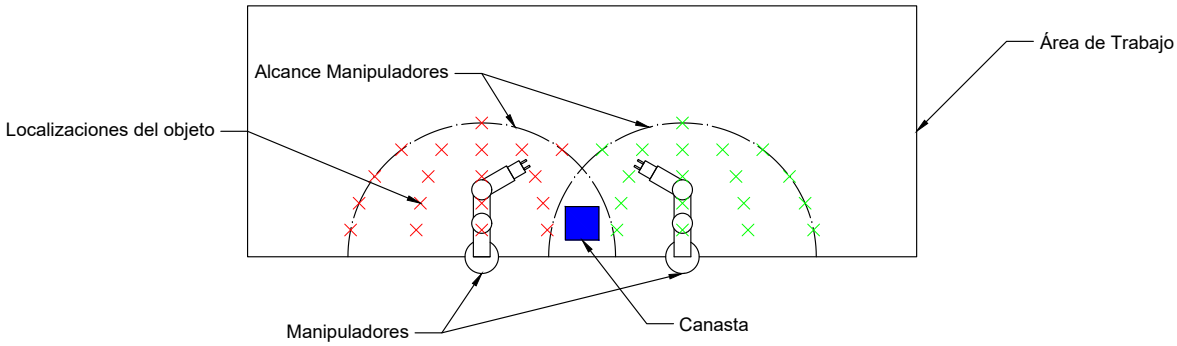


Figura 4.14: Diagrama experimento de toma de objetos. Elaboración propia: AutoCAD

El experimento se diseñó como factorial completo y se ejecutaron 3 réplicas para cada condición experimental. Los factores y niveles considerados se muestran en la tabla 4.15, al ser un experimento 2^3 replicado tres veces, se realizaron un total de 24 corridas. La variable de respuesta es la proporción de aciertos obtenida en cada una de las corridas.

Tabla 4.15: Factores y niveles del experimento de toma de objetos.

Factor	Valor Bajo	Valor Alto
Geometría Objeto	Cubo	Cilindro
Velocidad Brazo	50 %	100 %
Masa Objeto	50 g	200 g

En la figura 4.15 se observa un gráfico de cajas que resume el porcentaje de secuencias de recolección y colocación del objeto exitosas para cada las distintas corridas experimentales. Considerando todas las ejecuciones únicamente cuatro secuencias fallaron. Tomando los resultados de las 24 corridas experimentales se obtuvo un porcentaje de acierto global del 99,69%. El análisis ANOVA encontró que ninguno de los factores tuvo un efecto significativo en los resultados, lo que indica que

los fallos encontrados no tienen una relación estadísticamente significativa con los factores estudiados.

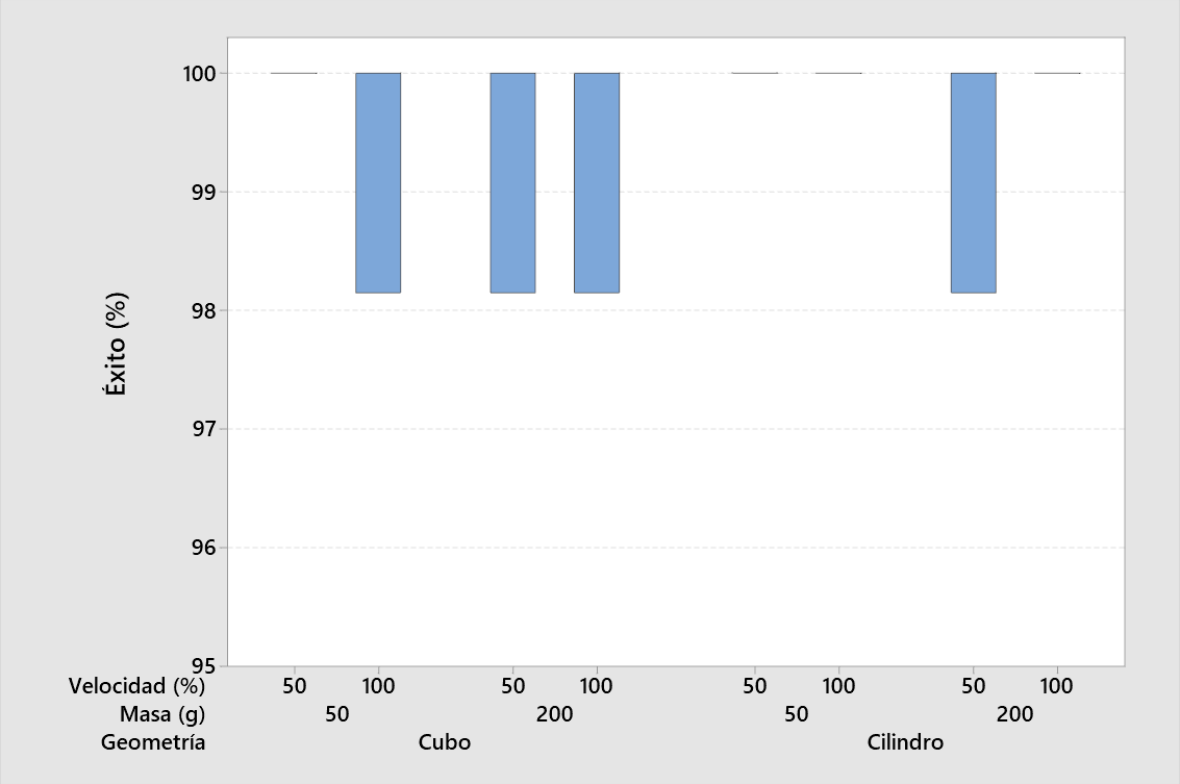


Figura 4.15: Gráfico de cajas de la proporción de aciertos en las secuencias de *pick and place*. Elaboración propia: Minitab

Gracias a la alta tasa de aciertos obtenida en este experimento, se pudo concluir que el sistema es capaz de manipular objetos de forma robusta. Esto implica que tanto los controladores del *gripper* y *manipulador* como los parámetros físicos del ambiente simulado fueron seleccionados de forma apropiada. Debido a que parece que el uso de la geometría tipo cubo tiene una mayor tendencia a provocar fallos en la manipulación, para el experimento de aprendizaje de la sección 6.3 se utilizó el objeto con forma cilíndrica.

CAPÍTULO 5

DISEÑO COGNITIVO

En este capítulo se expondrá el diseño realizado para los subsistemas que representan la componente cognitiva del proyecto, específicamente el subsistema de *redescripción* y su integración con la arquitectura cognitiva MDB. Se comienza definiendo los requerimientos para los subsistemas y la descomposición funcional que guió el diseño. Posteriormente, se detalla la selección del software utilizado y la implementación de los subsistemas. Se concluye con el experimento de validación del sistema de percepción visual de la herramienta.

5.1 Identificación de requerimientos

Al igual que los subsistemas presentados en el capítulo anterior, la componente cognitiva de la herramienta se diseñó tomando en cuenta los requerimientos expresados por el cliente en las entrevistas realizadas. Los requerimientos se organizaron mediante necesidades y especificaciones, que se muestran en las tablas 5.1 y 5.2 respectivamente.

El problema abordado en este capítulo consiste en generar los módulos de software

Tabla 5.1: Necesidades concernientes al diseño cognitivo de la herramienta.

No.	Necesidades Interpretadas Cliente	Importancia
2.	**La herramienta permite la obtención de datos perceptuales**	
2.1	El sistema perceptual extrae características del entorno a partir de datos sensoriales	5
2.2	El sistema perceptual recibe datos mediante un sistema de visión	4
3.	**La herramienta es capaz de controlar las acciones ejecutadas**	
3.1	El sistema de acciones permite la ejecución de movimientos básicos	5
3.2	El sistema de acciones permite la manipulación de objetos	5
4.	**La herramienta permite la ejecución de experimentos de aprendizaje cognitivo**	
4.1	La herramienta permite ejecutar experimentos de manipulación básicos	5
4.2	La herramienta permite integrarse con una arquitectura cognitiva	5
4.3	La herramienta permite el aprendizaje de tareas por refuerzo	3
5.	**La herramienta se implementa en una arquitectura de software modular**	
5.1	La herramienta permite el procesamiento en un computador externo	4
5.3	La herramienta distribuye sus capacidades en módulos	3
6.	**La herramienta cuenta con documentación adecuada**	
6.3	La herramienta es de desarrollo abierto	4

Tabla 5.2: Especificaciones objetivo concernientes al diseño cognitivo de la herramienta.

No.	Métrica	Unidad	Imp.	Valor marginal	Valor ideal
14	Error en la prueba de percepción	mm	5	<3	<2
15	Proporción de acciones recompensadas	% recompensas/acciones	5	>70	>80
16	Estructuras cognitivas integradas	Lista	5	-LTM	-Modelador Mundo -MotivEn -LTM
17	Tiempo procesamiento de percepciones	ms	3	<1000	<500
19	Módulos ejecutados de forma concurrente	Lista	3	-Percepción -Acción	-Percepción -Acción -Aprendizaje y Memoria
23	Puntuación de apertura módulos de Software	Puntuación OSS Watch.	4	>60	>75

que permitan dotar a la plataforma robótica de las capacidades cognitivas requeridas. Esto incluye las capacidades de percepción y acción que le permitan llevar a cabo el experimento de aprendizaje cognitivo presentado en la sección 3.4.1. También se deben seguir los lineamientos del uso de una arquitectura modular y de desarrollo abierto.

5.2 Descomposición funcional

El subsistema denominado *redescripción* es el foco de este capítulo, y como fue presentado en la figura 3.3 es el encargado de generar un puente entre la arquitectura

cognitiva y la plataforma robótica. Como se muestra en la figura 5.1, el subsistema tiene dos funciones principales: extraer la información contenida en los datos sensoriales en forma de percepciones y transformar las acciones o *policies* propuestas por la arquitectura cognitiva en comandos que pueda ejecutar la plataforma robótica.

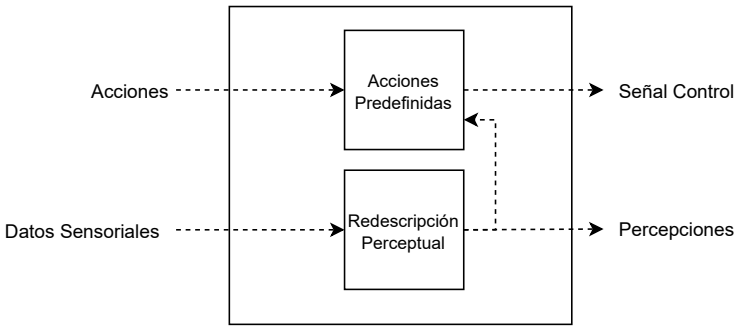


Figura 5.1: Descomposición funcional del subsistema *redescripción*. Elaboración propia

Un aspecto a destacar es que el diseño del sistema de redescripción en este proyecto tiene como objetivo fundamental generar las percepciones y *policies* específicas para el experimento de aprendizaje presentado en la sección 3.4.1. En general, una redescripción es una transformación de datos de una forma a otra, y en el contexto de robótica cognitiva está orientada a permitir una mayor flexibilidad en el aprendizaje de distintas tareas. Actualmente se investigan métodos para que un agente aprenda redescripciones del ambiente de forma automática y flexible [31].

5.3 Selección del software

Para realizar la implementación de las funciones requeridas se debió realizar la selección de las herramientas de *software* para desarrollar el procesamiento de las percepciones y la ejecución de las *policies*.

En el caso de las acciones, se aprovecha el servidor de comandos descrito en la sección 4.5.2 para realizar la implementación. Por esta razón se utilizaron secuencias de comandos basadas en *Python*. Esta selección se realizó debido a la existencia de

un API que permite la ejecución de servicios de ROS de forma programática desde ese lenguaje.

Para el procesamiento de la información visual, se buscaron opciones que permitieran la extracción de la información requerida a partir los datos de las cámaras disponibles en la plataforma robótica. Para el procesamiento de los datos en 2D generados por la cámara RGB de la plataforma robótica se planteó el uso de *OpenCV* [68] o *Scikit-Image* [69] como biblioteca para el procesamiento de imágenes. Adicionalmente, se identificó la biblioteca *CameraTransform* [70] que permite localizar objetos en imágenes mediante transformaciones de perspectiva.

Para el procesamiento de los datos 3D generados por la cámara RGB-D, se consideró el uso de las siguientes herramientas: *FindObject* [71], *Point Cloud Library* [72] y *Object Recognition Kitchen* (ORK) [73]. Las opciones consideradas tienen funcionalidades ligeramente distintas, *FindObject* y *Object Recognition Kitchen* utilizan técnicas de reconocimiento de patrones (ver [74, ch. 14] y [75] para una explicación sobre estas técnicas) para encontrar objetos en la escena y además permiten estimar su posición. Por otro lado, *Point Cloud Library* es una biblioteca que permite el procesamiento general de nubes de puntos donde se pueden realizar operaciones como el segmentado y la identificación de objetos por su geometría.

La operación de localización de los objetos requerida es relativamente sencilla, ya que estos están siempre sobre una mesa de altura conocida y además tienen un color uniforme. Por esta razón se decidió utilizar únicamente los datos de la cámara RGB para obtener las percepciones, de manera que se pudiera disminuir el tiempo de desarrollo de la herramienta. Adicionalmente, las opciones *FindObject* y *Object Recognition Kitchen* no tienen un funcionamiento adecuado con objetos de textura uniforme, por lo que se descartó su uso en este proyecto. Entre las opciones de procesado de imágenes RGB se optó por el uso de Scikit-Image, debido a que proporciona un API de *Python* ampliamente documentado que permite realizar las

operaciones necesarias. Eventualmente, debido a la modularidad del software desarrollado se podrán implementar módulos basados en *OpenCV* si se desea dotar de capacidades adicionales al sistema perceptual. También se decidió utilizar la biblioteca *CameraTransform* por las facilidades que ofrece. En la bitácora de diseño se pueden encontrar las matrices de evaluación que sustentaron las decisiones explicadas (tablas A.22 y A.23).

5.4 Redescripción perceptual

En la figura 5.2 se muestra el diagrama de bloques que representa las operaciones realizadas en el proceso de redescripción perceptual. Este diagrama se diseñó de forma específica a los datos disponibles en la plataforma robótica y a la tarea experimental deseada. En las siguientes secciones se abarcarán los dos tipos de procesamiento que se realizan.

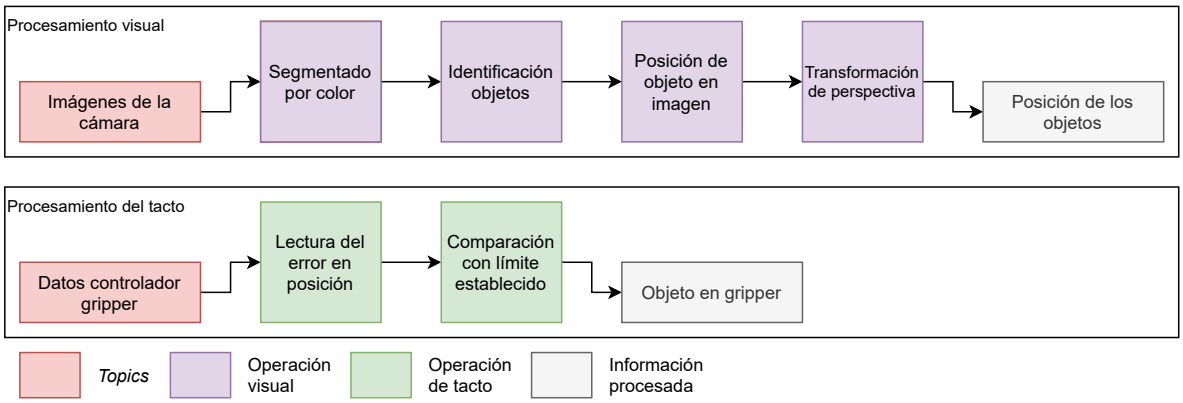


Figura 5.2: Diagrama de bloques del proceso de redescripción perceptual. Elaboración propia

Para integrar las percepciones a la red de comunicación de ROS se utilizó un servicio que permite solicitar la información perceptual. El mensaje del servicio se presenta en el *listing* 5.1. Cada vez que se solicitan las percepciones, se envía tanto la información de posición del objeto y la canasta como un booleano que indica si se encuentra sujetado un objeto en alguno de los *grippers*.

Listing 5.1: Mensaje del servicio de percepción.

```
---
geometry_msgs/Point red_object
geometry_msgs/Point basket
bool obj_in_left_hand
bool obj_in_right_hand
```

5.4.1 Procesamiento visual

Como se presentó en el diagrama de bloques de la figura 5.2, el procesamiento visual comienza con una segmentación por color. Debido a que los objetos del ambiente tienen colores que corresponden a los canales de la imagen RGB, se realizó la segmentación en ese espacio de color. La ecuación 5.1 representa la segmentación realizada, donde $B(i,j)$ representa la imagen binarizada resultante. $R(r,g,b)$ representa el conjunto de valores para cada uno de los canales RGB que generan la segmentación deseada. Para segmentar el objeto rojo se utiliza la región descrita en la ecuación 5.2, mientras que para la canasta se utiliza la región descrita en (5.3).

$$B(i,j) = \begin{cases} 1 & \text{si } I(i,j) \in R(r,g,b) \\ 0 & \text{en otro caso} \end{cases} \tag{5.1}$$

$$R(r,g,b) = \begin{cases} r \in [200,255] \\ g \in [0,5] \\ b \in [0,5] \end{cases} \tag{5.2}$$

$$R(r,g,b) = \begin{cases} r \in [0,5] \\ g \in [0,5] \\ b \in [200,255] \end{cases} \tag{5.3}$$

Una vez que se han segmentado los objetos, estos son identificados al encontrar las regiones de píxeles blancos con mayor área. Una vez que las regiones que corresponden a cada objeto han sido identificadas se puede obtener la localización en el plano de la

imagen. La figura 5.3 muestra un ejemplo del resultado final de este proceso.

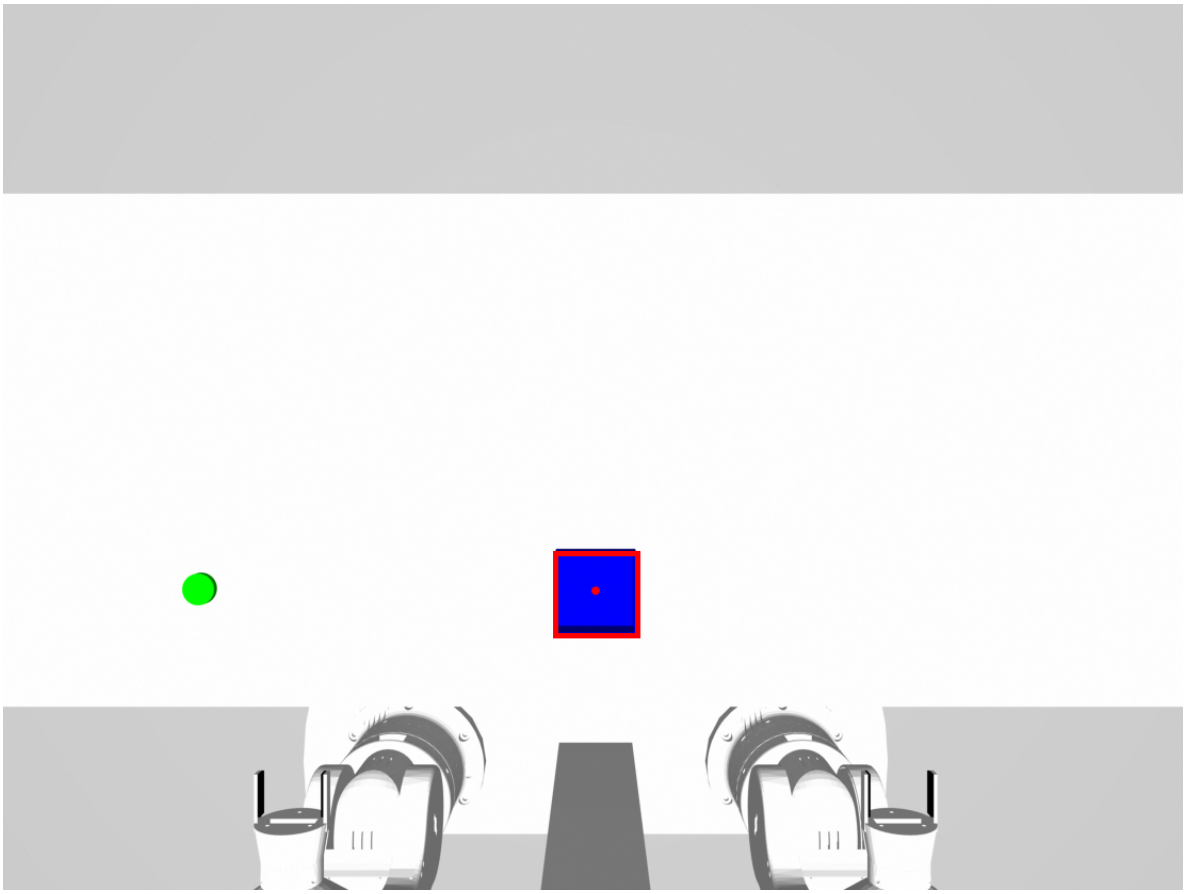


Figura 5.3: Segmentado e identificación de la canasta. Elaboración propia

La última operación del procesamiento visual es una transformación de perspectiva que permite obtener la posición de los objetos respecto a las coordenadas de la plataforma robótica. Esta transformación se implementó utilizando la biblioteca *CameraTransform*, en la cual se pueden definir los parámetros de la cámara para realizar la transformación. Debido a que la cámara es implementada en simulación se utilizaron como parámetros intrínsecos las especificaciones dadas por el fabricante que se presentaron en la tabla 4.12. Es importante destacar que estos valores no deben utilizarse fuera del entorno simulado, ya que no tienen la exactitud necesaria, en un caso práctico deben ser obtenidos mediante un proceso de calibración. La ecuación 5.4 muestra la matriz de parámetros intrínsecos utilizada.

$$\mathbf{P} = \begin{bmatrix} 2709.56 & 0 & 1640 & 0 \\ 0 & 2713.97 & 1232 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{5.4}$$

5.4.2 Procesamiento del tacto

Para detectar si se ha sujetado un objeto en alguna de las manos del robot, se utilizará la medición de error del controlador de los *grippers*. Esto debido a que en esta etapa no se equipó al robot de sensores de presencia/ausencia simulados. En la figura 5.4 se muestra el error de los controladores de cada uno de los dedos del *gripper* al sujetar un objeto y posteriormente soltarlo.

En la gráfica se observa que al tocar el objeto el error comienza a subir linealmente, este fenómeno se da debido a que *MoveIt* da un comando de rampa al controlador de los *grippers* cuando se pasa a la posición cerrada. Una vez que la consigna del controlador llega al valor máximo, el error se estabiliza en un valor cercano a 7.5 mm, que corresponde al radio del objeto sujetado (12.5 mm) menos los 5 mm de apertura que tiene cada uno de los dedos del *gripper*. Este comportamiento es muy apropiado para detectar la presencia de un objeto en el *gripper*.

Debido a que no se puede asegurar que el objeto se encuentre siempre centrado en el *gripper* o que siempre se sujete a través de su diámetro, para decidir si un objeto se encuentra dentro del *gripper* se utilizan los datos del error de ambos dedos del *gripper*. Como se muestra en la ecuación 5.5, si el promedio de los errores de cada uno de los dedos de un *gripper* supera los 5 mm se determina que existe un objeto sujetado en el *gripper*.

$$\text{Objeto Sujetado} = \begin{cases} 1 & \text{si } \frac{e_0+e_1}{2} > 5 \text{ mm} \\ 0 & \text{en otro caso} \end{cases} \tag{5.5}$$

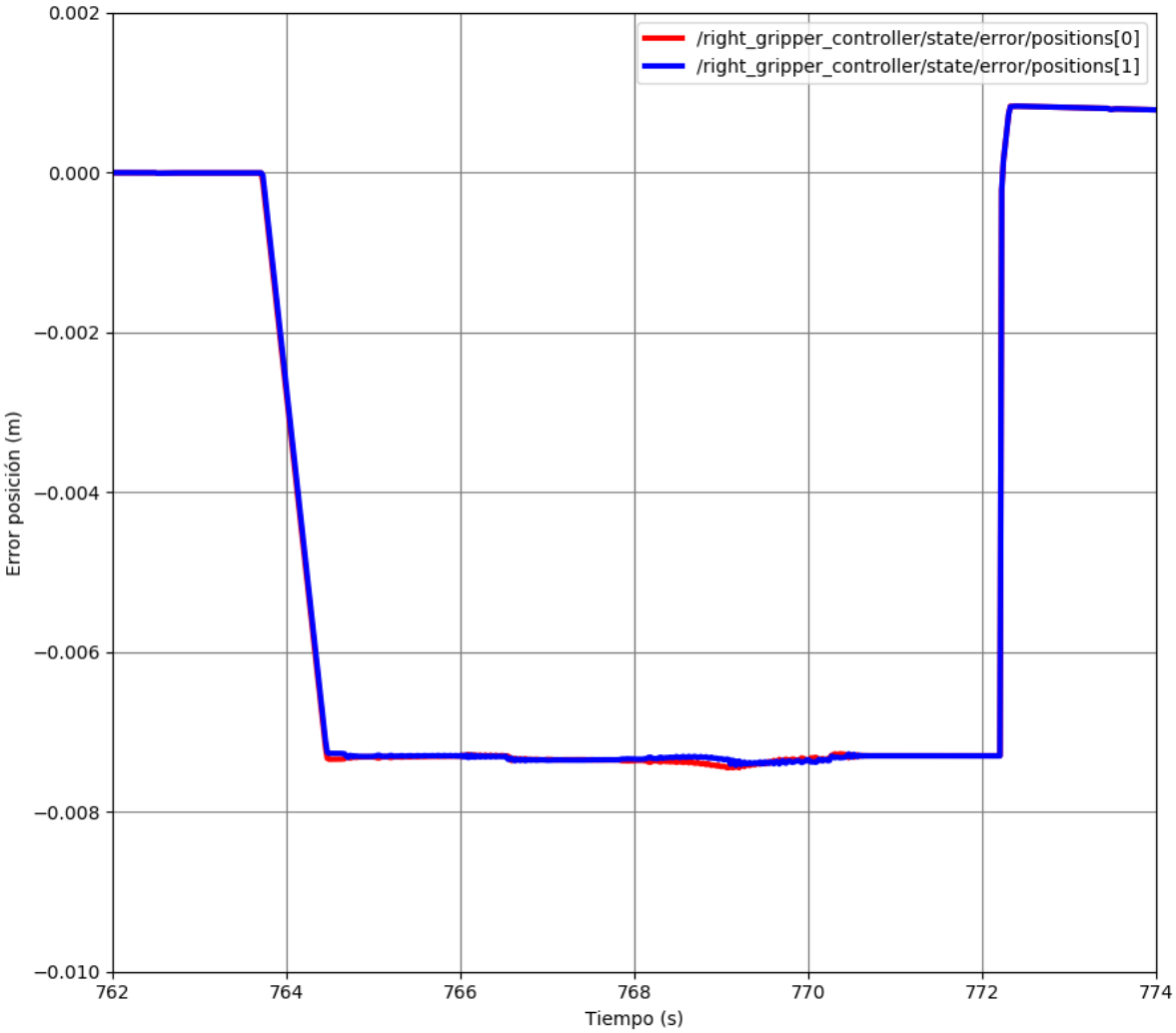


Figura 5.4: Error del controlador PID de los dedos del *gripper*. Elaboración propia: rqt_graph

5.5 Acciones predefinidas

Las acciones predefinidas o *policies* consisten en una secuencia de comandos que se envían a la plataforma robótica para lograr un objetivo específico. Las *policies* son fijas y se definieron junto con el experimento de aprendizaje en la sección 3.4.1. Todas las acciones predefinidas que se implementaron se presentan en la tabla 5.3, en la cual se incluyen enlaces hacia videos que muestran la ejecución de las acciones.

Las policies se implementaron utilizando el servidor de comandos descrito en la sección 4.5.2. Previo a realizar cualquier acción, se revisa que existan condiciones

Tabla 5.3: Demostración de las *policies* implementadas.

Policy	**Descripción**	**Video**
grasp_left	Toma el objeto con el brazo izquierdo	Enlace
grasp_right	Toma el objeto con el brazo derecho	Enlace
press_button	Presiona el botón de ayuda	Enlace
place_object_left	Deposita el objeto en la canasta con el brazo izquierdo	Enlace
place_object_right	Deposita el objeto en la canasta con el brazo derecho	Enlace
change_hands	Cambia el objeto de un brazo a otro	Enlace

adecuadas para realizar la *policy*, en caso de que no se pueda ejecutar correctamente la acción el robot se mantiene inmóvil. En el caso de *grasp_left* y *grasp_right* se revisa que el objeto esté dentro del alcance del manipulador. Para *press_button* se revisa que ninguno de los brazos tenga sujetado al objeto. En el caso de *place_object_left* y *place_object_right* se revisa que el brazo correspondiente tenga el objeto y que la canasta se encuentre al alcance. Finalmente, para *change_hands* se revisa que alguno de los brazos esté sujetando al objeto.

La *policy change_hands* es capaz de acercar el objeto al manipulador. Para asegurar que el manipulador siempre pueda tomar el objeto, se definió una zona en la que el objeto se coloca después de presionar el botón, que está dentro del espacio disponible para el manipulador. El objeto se coloca de forma aleatoria en cualquier punto de la zona de colocación mostrada en la figura 5.5. Sin embargo, existe una restricción respecto a la distancia del objeto y la canasta, ya que siempre se asegura que estos estén a una distancia mínima de 150 mm. Esta restricción existe para evitar colisiones del brazo con la canasta cuando se va a tomar el objeto.

5.6 Interfaz con la arquitectura cognitiva

Ya se han explicado los métodos para obtener percepciones y ejecutar acciones, por lo que ahora se va a presentar la forma en la que estos dos componentes se utilizarán para integrarse con la arquitectura cognitiva MDB, de forma que se pueda ejecutar el

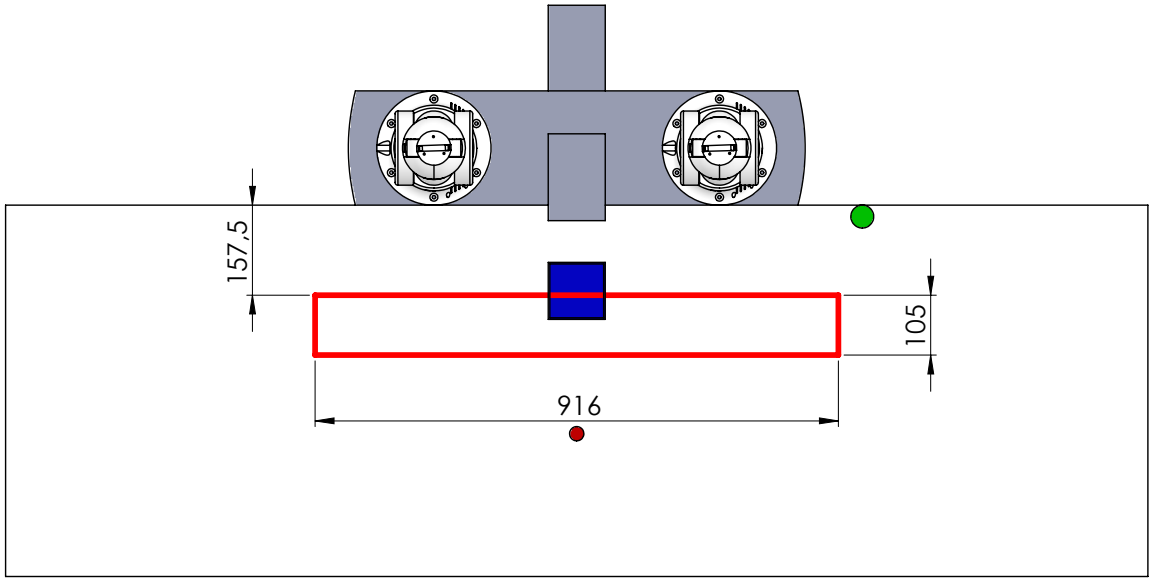


Figura 5.5: Zona de colocación del cilindro después de la *policy press_button*. Elaboración propia: SolidWorks

experimento de aprendizaje. La implementación actual utiliza únicamente la estructura de memoria a largo plazo de la arquitectura, esto debido a que las demás estructuras se encuentran en desarrollo activo y se optó por utilizar la LTM debido a que es la única que cuenta con una versión estable en la que se puede ejecutar el experimento planteado.

Para generar la integración se comenzó diagnosticando el funcionamiento de la arquitectura cognitiva, para esto se hizo uso de un simulador de eventos desarrollado por el GII. Gracias a los experimentos ejecutados con el simulador de eventos se identificó el diagrama de flujo presentado en la figura 5.6. El ciclo de acción-percepción que se genera, permite la ejecución de experimentos de aprendizaje durante un número específico de iteraciones.

Aparte del funcionamiento de la arquitectura, se identificaron los canales de comunicación que se debían utilizar para realizar la integración. En la figura 5.7 se muestra un esquema de los *topics* que se utilizan para integrar la LTM con la plataforma robótica. Se identificaron tres tipos de mensajes, que corresponden a las

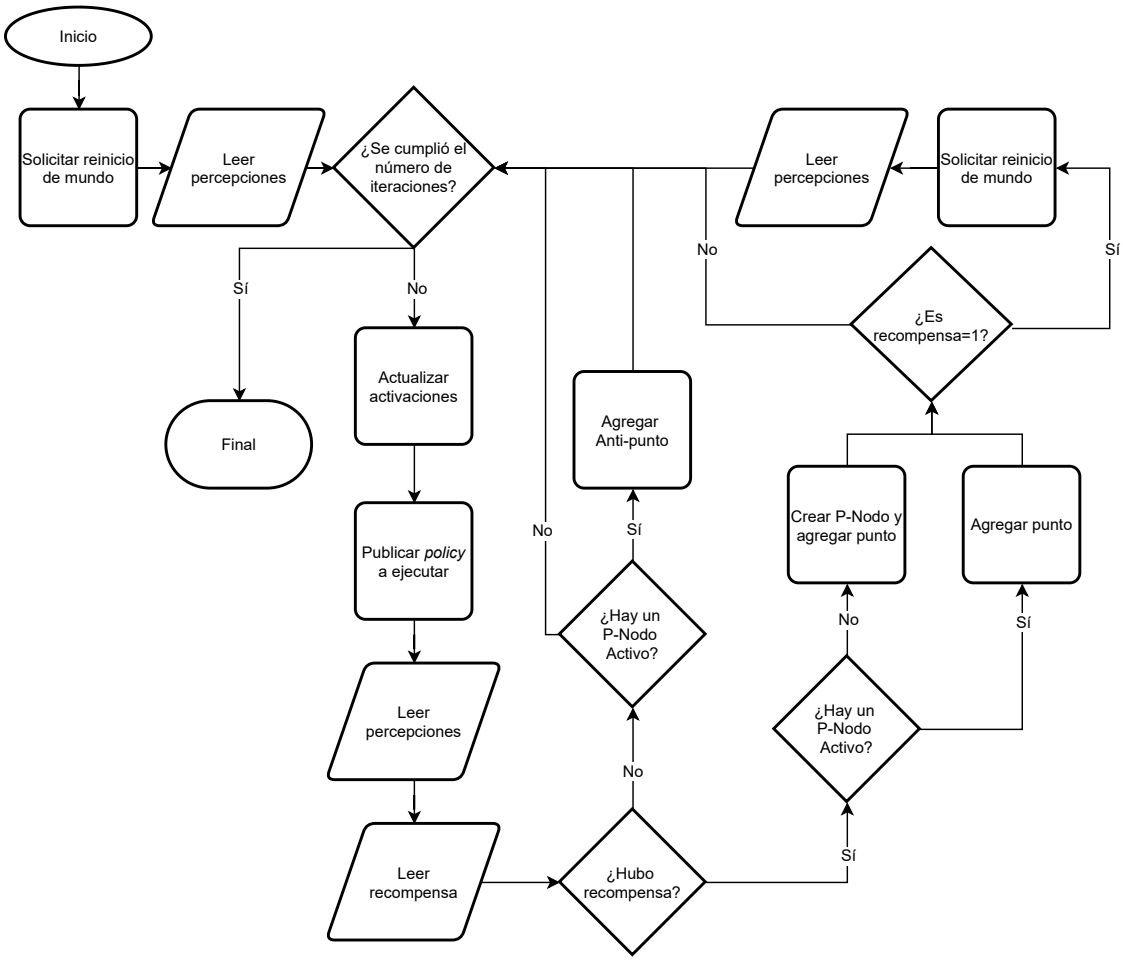


Figura 5.6: Diagrama de flujo del funcionamiento de la memoria a largo plazo del MDB. Elaboración propia

percepciones, acciones y control del experimento.

La integración de la plataforma robótica con la arquitectura cognitiva se realiza mediante el nodo denominado *servidor plataforma robótica*. Tiene la función de permitir que la plataforma robótica realice las operaciones requeridas por la arquitectura cognitiva para ejecutar el experimento de aprendizaje. En la figura 5.8 se muestra el diagrama de flujo del programa que integra la plataforma robótica con el MDB. El programa consiste de dos *callbacks*, es decir, ejecuta alguna de sus dos ramas en el momento que se publica un mensaje en los *topics* donde la arquitectura solicita *policies* o el reinicio del mundo.

El comando de reinicio del mundo coloca tanto el objeto como la canasta en

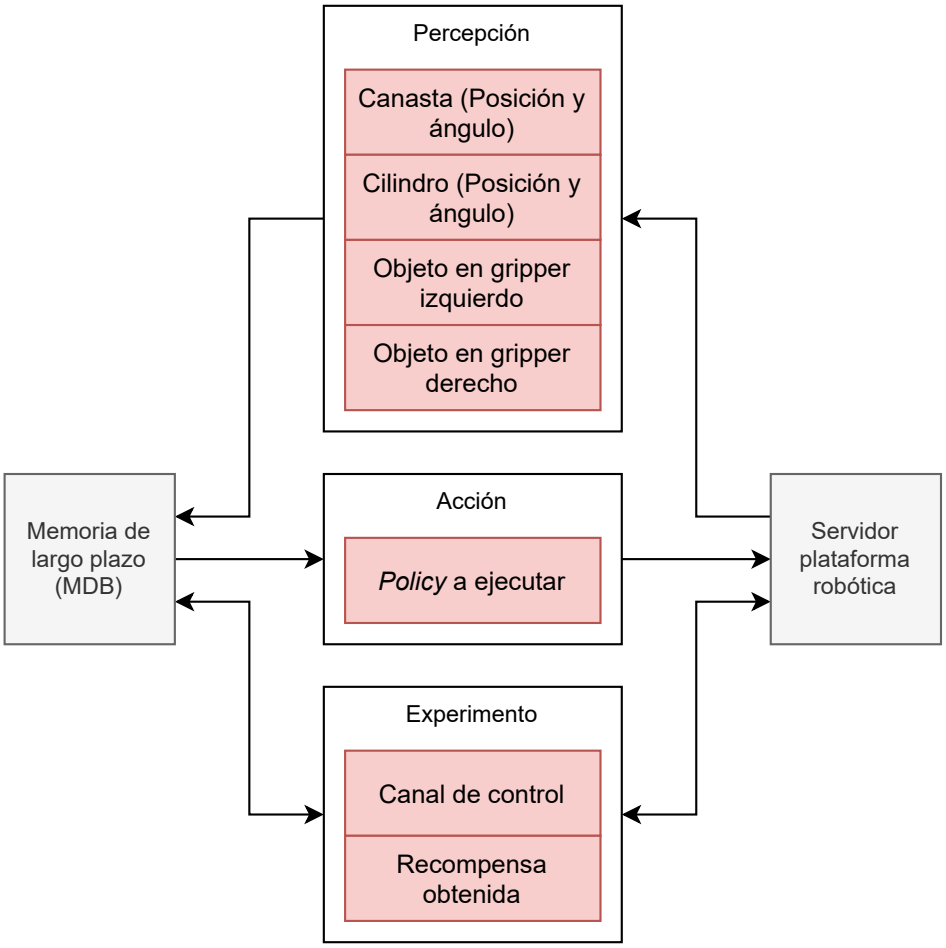


Figura 5.7: Topics utilizados para la integración con el MDB. Elaboración propia

posiciones aleatorias, sin embargo, para evitar condiciones en las que el robot no tenga forma de obtener recompensa, se colocó una restricción en la posición de la canasta. Como se observa en la figura 5.9, las posiciones donde aparece la canasta se encuentran limitadas a un rectángulo que está completamente dentro del espacio disponible por el robot en la altura $z = 0.9$ m. Sin embargo, el objeto puede colocarse en cualquier posición dentro del campo visual del robot.

Las recompensas que puede obtener el robot se resumen en la tabla 5.4. La recompensa más alta se obtiene cuando el objeto se encuentra dentro de la caja, para esto se comprueba que la distancia entre centros de la canasta y el objeto no supere 43 mm. Cuando el brazo que está sujetando el objeto no alcanza a la canasta y se ejecuta la *policy* de cambio de brazos, se otorga 0.75 de recompensa. Cuando se

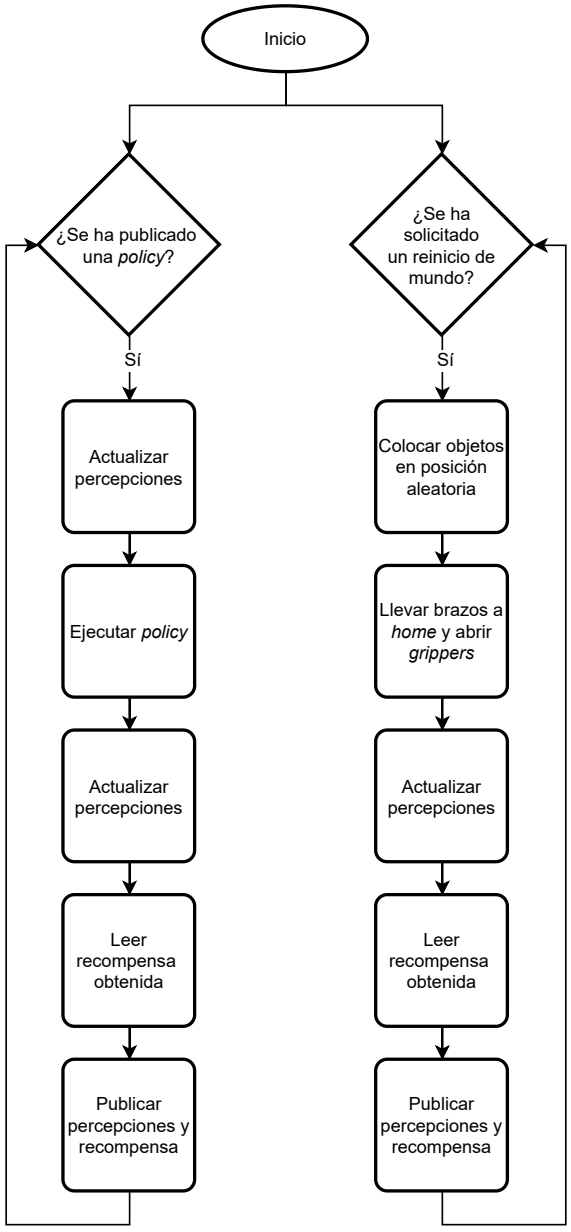


Figura 5.8: Diagrama de flujo del funcionamiento del programa de integración con el MDB. Elaboración propia

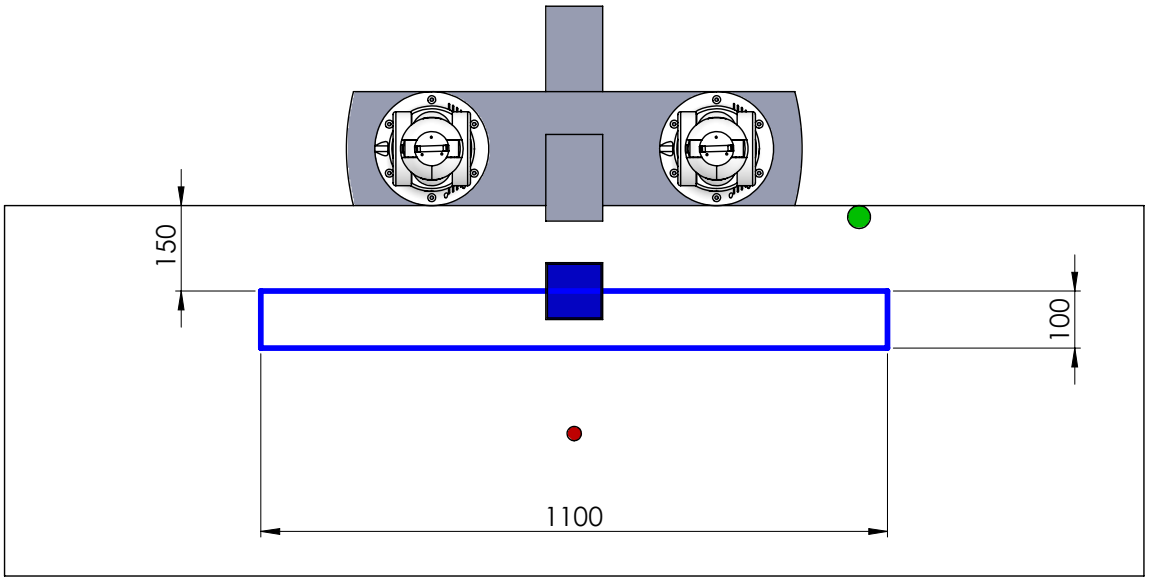


Figura 5.9: Zona de colocación de la canasta. Elaboración propia: SolidWorks

detecta que después de realizar una *policy* alguno de los sensores de objeto en *gripper* se activa, significa que se logró tomar el objeto, por lo que se otorga 0.5 de recompensa. Finalmente, si se realizó la operación de acercar el objeto al robot con ayuda del botón, se asigna una recompensa de 0.25.

Tabla 5.4: Recompensas obtenidas por el robot.

Condición	Recompensa
Se colocó el objeto en la canasta	1
Se cambió de manos el objeto apropiadamente	0.75
Se tomo el objeto con alguno de los brazos	0.5
Se acercó el objeto con ayuda del botón	0.25

5.7 Experimento de percepción visual

El objetivo de este experimento es comprobar la precisión de las medidas generadas por el subsistema de redescripción perceptual, específicamente para el procesamiento visual. La figura 5.10 muestra las condiciones generales del experimento: se colocó un objeto sucesivamente en el área de trabajo y para cada posición el sistema perceptual

calculó la posición del objeto, también se registró la posición real del objeto reportada por el simulador.

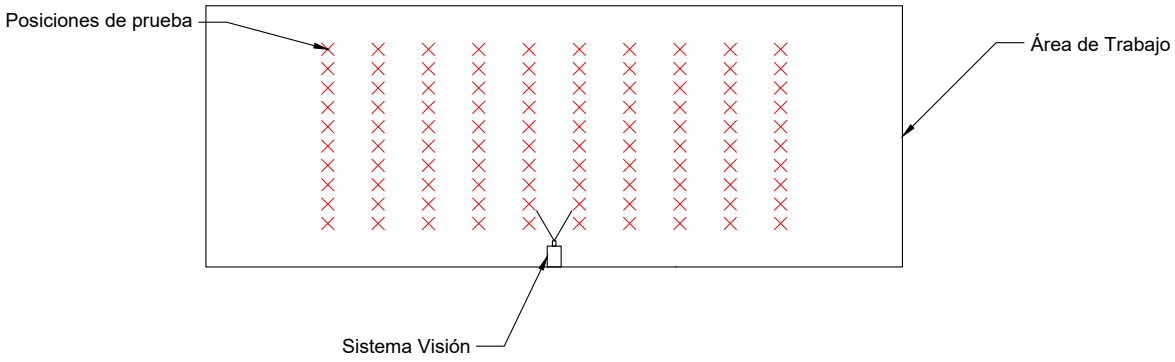


Figura 5.10: Diagrama experimento de percepción. Elaboración propia: AutoCAD

El factor estudiado en este experimento fue el desempeño del sistema visual ante los distintos objetos que constituyen el ambiente, por lo tanto, se colocaron los tres objetos en diferentes posiciones que cubrieron la totalidad del campo visual de la cámara RGB. Para la canasta se exploraron 7700 posiciones, mientras que para el cubo y el cilindro se probaron 9802 puntos. De esta forma se formó una rejilla de aproximadamente 10x10 mm en el campo visual.

En la figura 5.11 se muestra un gráfico de cajas y bigotes que resume los resultados obtenidos en el experimento. Se puede observar que se obtiene una mayor variabilidad en el error de medición de la canasta, esto puede ser causado por la geometría irregular que amplifica el efecto de perspectiva de la cámara dentro de la medición, por lo que se obtiene un desempeño peor conforme la canasta se aleja del centro de la imagen. Los efectos de la perspectiva también afectan la medición del cilindro y el cubo, sin embargo, el efecto en estos objetos es menor, por lo que se observa una mayor precisión.

En general, los resultados demuestran que el sistema perceptual logra obtener mediciones con un error aceptable. En el caso de la canasta el 50% de las medidas tienen un error entre 1 mm y 3 mm, mientras que para el cilindro y el cubo el error se encuentra entre 1 mm y 2 mm. La precisión obtenida cumple con las especificaciones objetivo planteadas y por esto se decidió que los datos del sistema perceptual eran

aptos para utilizarse en la tarea de manipulación.

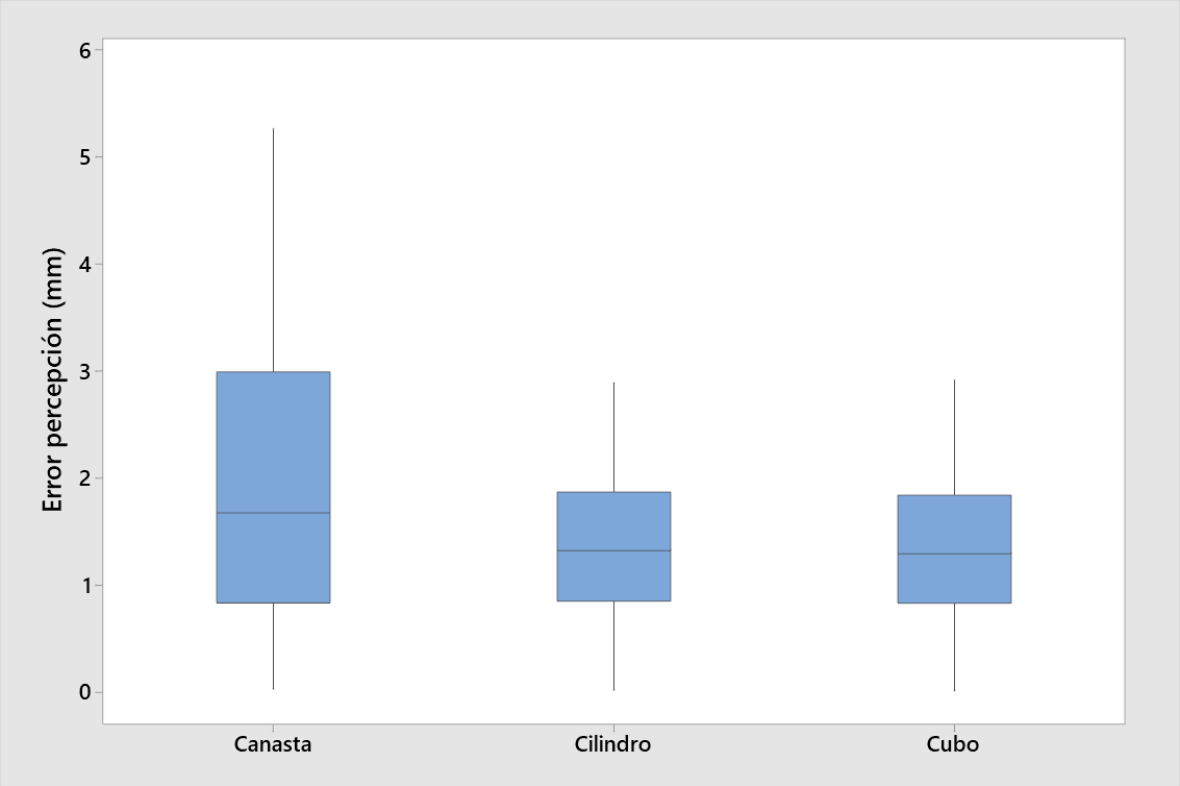


Figura 5.11: Gráfico de cajas del error de distancia en la percepción de distintos objetos. Elaboración propia: Minitab

CAPÍTULO 6

INTEGRACIÓN DEL SISTEMA

En los capítulos anteriores se abarcaron los distintos subsistemas que componen el entorno robótico desarrollado. A continuación, se tratarán los aspectos de la integración de la herramienta como un sistema: la interfaz entre los distintos subsistemas y la organización del repositorio de *software*. Se finalizará con la prueba de aprendizaje utilizada para validar el diseño del sistema. A la herramienta diseñada se le dio el nombre *Open-Source Cognitve Applied Robot* (OSCAR), que permite sintetizar las principales características de su diseño y propósito.

6.1 Intercomunicación entre subsistemas

En la figura 3.3 se presentó una vista general de la descomposición del proyecto desde una perspectiva conceptual. Durante el diseño, esa estructura se siguió para dar lugar a la herramienta diseñada, cuya implementación en ROS se muestra en la figura 6.1.

Como se observa, el diagrama de bloques se basa en el que se presentó en la figura 4.3

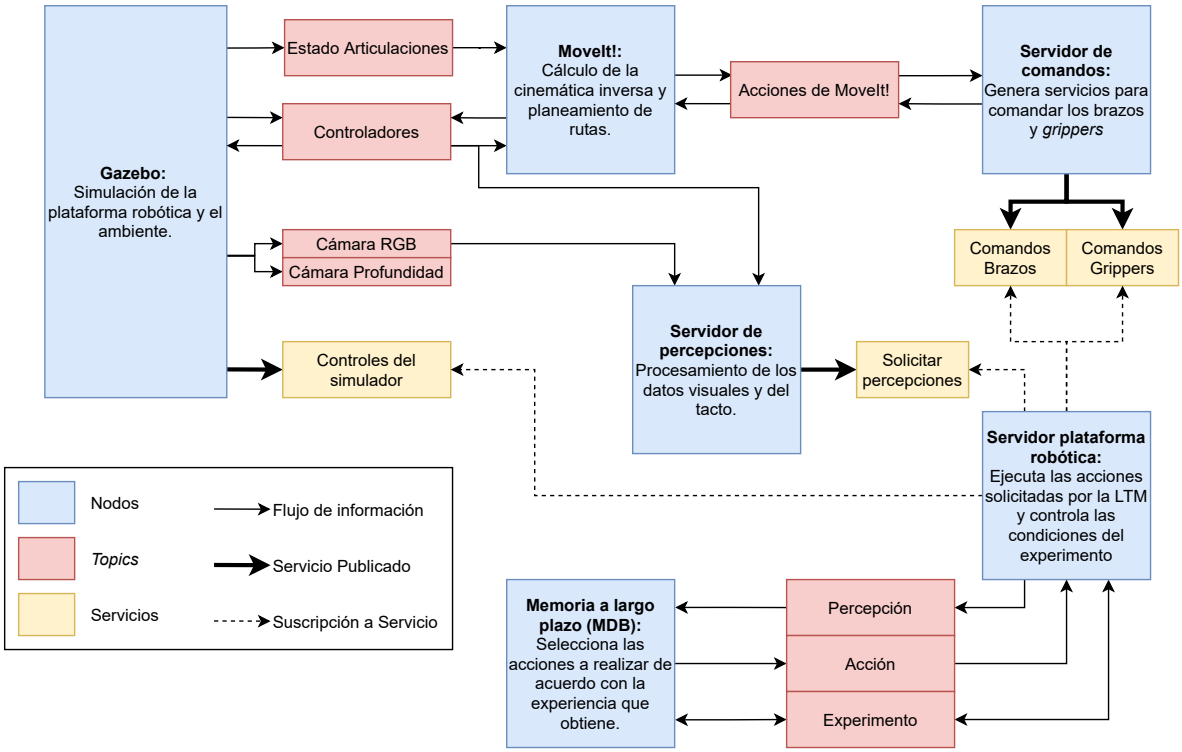


Figura 6.1: Diagrama de bloques de OSCAR. Elaboración propia

para la plataforma robótica. Para finalizar la herramienta, se añadieron tres nodos adicionales: el servidor de percepciones, el servidor de la plataforma robótica y la memoria a largo plazo del MDB. Estos nodos fueron abarcados en el capítulo 5, ya que corresponden a la componente cognitiva de la herramienta.

Una de las principales ventajas que provee la implementación altamente modular de la herramienta es que cada componente ve al resto como una caja negra, es decir, cada nodo recibe un tipo específico de información pero la fuente del que la recibe no es importante. Esta característica le da a la herramienta una alta flexibilidad a la hora de modificar módulos para mejorar sus capacidades ya que, mientras se respeten los canales de comunicación, la implementación específica de cada nodo puede cambiarse sin que el resto del sistema necesite modificación alguna. Esto ayudará al proceso de adaptación de la herramienta a una implementación física de la plataforma robótica.

6.2 Descripción del repositorio de software

La implementación de OSCAR se organizó mediante un grupo de paquetes de ROS. Para facilitar el acceso y publicación del proyecto, los paquetes se alojaron en repositorios remotos. En la figura 6.2 se muestran los paquetes alojados en cada uno de los repositorios. En los siguientes enlaces se puede acceder al código de la herramienta: OSCAR, THOR_SIMULATOR. Dentro de los repositorios se encuentra la documentación de cada paquete implementado. Adicionalmente, se incluyen manuales de instalación y uso de la herramienta. Toda la implementación de la herramienta fue publicada mediante la licencia *GNU GPL*v3 [76].

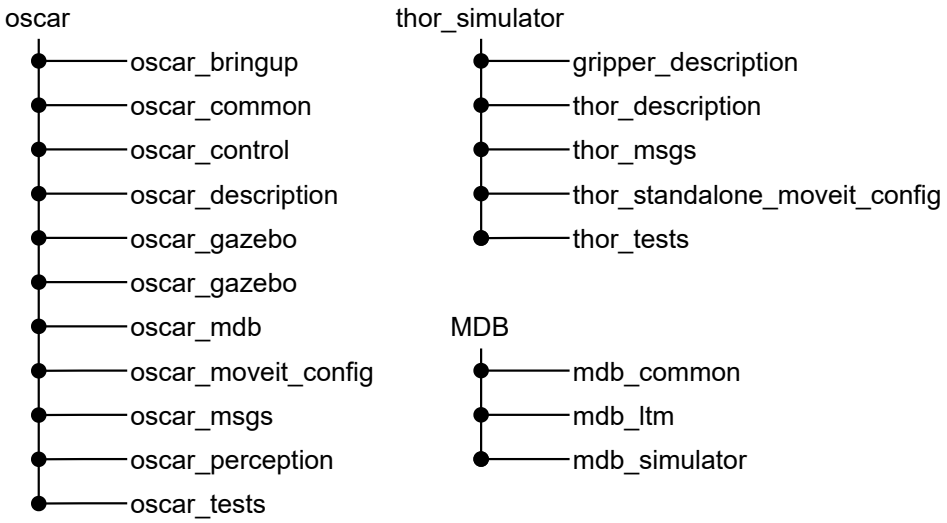


Figura 6.2: Estructura de paquetes de ROS en los repositorios de *software*. Elaboración propia

6.3 Prueba de aprendizaje

Para finalizar la validación de la herramienta, mediante este experimento se demuestra la capacidad de aprendizaje generado mediante la integración de la plataforma robótica y el MDB. En este experimento se ejecutó la memoria a largo plazo por 3000 iteraciones. Se colocó una restricción para que después de 10

iteraciones sin recompensa se realizara un reinicio del mundo, esto para que en caso de algún fallo el robot pudiera continuar el aprendizaje. En el siguiente enlace se puede acceder a un video que muestra el aprendizaje: ENLACE.

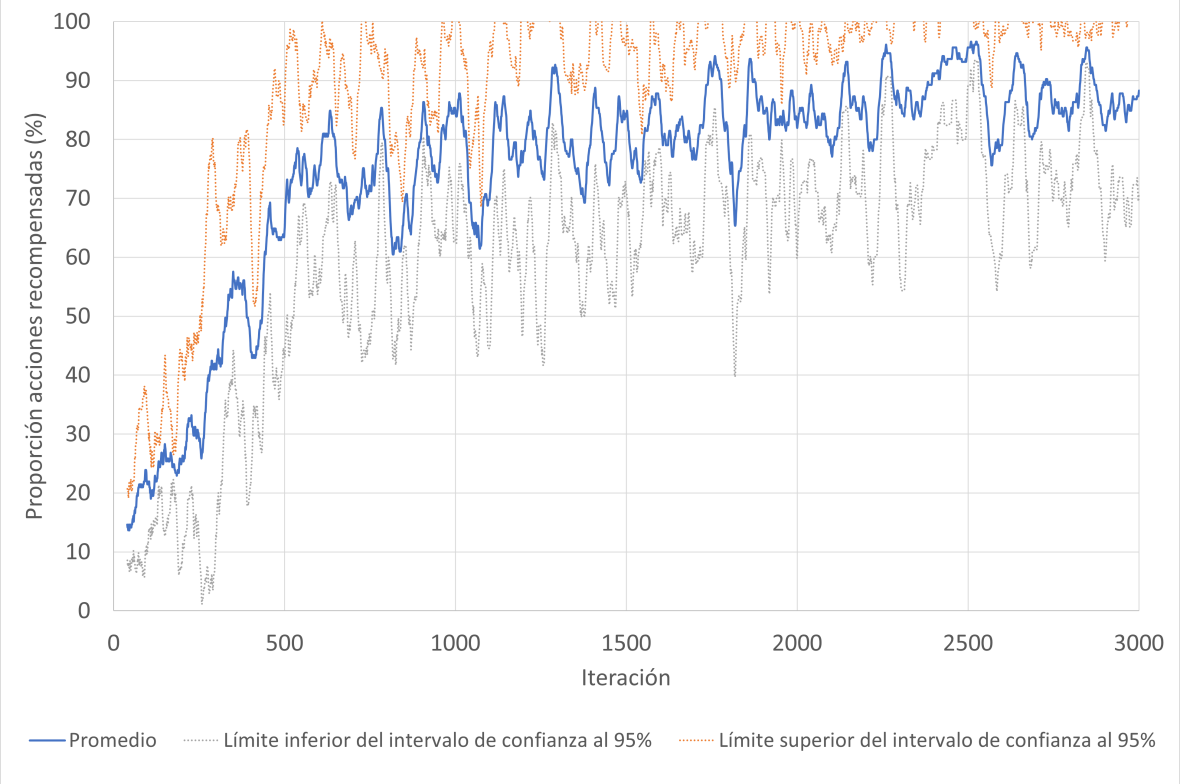


Figura 6.3: Proporción de acciones recompensadas durante la ejecución de 5 corridas experimentales. Elaboración propia: Microsoft Excel

En la figura 6.3 se muestra la proporción de recompensas obtenidas para un promedio de 5 ejecuciones. Es evidente el aprendizaje acelerado que se genera en las primeras 500 iteraciones del experimento, que corresponde al momento en el que la memoria a largo plazo crea sus primeros P-Nodos. Al obtener zonas de certeza, aunque sean burdas, el desempeño en la tarea crece significativamente respecto a la selección aleatoria de *policies* que se realiza cuando no existe ningún P-Nodo activado.

Posterior a las 500 iteraciones, la mejora del desempeño se desacelera y comienza un periodo en el cual el aprendizaje progresa lentamente. Esto se da debido a que los P-Nodos lograron delimitar gran parte del espacio perceptual de manera que la mayoría

del tiempo los objetos del ambiente se disponen en posiciones que el robot ya conoce. A pesar de esto, se observan puntos en los cuales en los que baja significativamente el desempeño del robot, como sucede alrededor de la iteración 1050 y 1800. Esas disminuciones en las recompensas se dan cuando los objetos quedan en una zona límite, por ejemplo, cuando el cilindro queda colocado cerca de los bordes del espacio de trabajo de alguno de los manipuladores. En un caso como ese, es posible que una *policy* incorrecta tenga un alto nivel de activación, lo que provoca repetidos fallos que impactan negativamente las recompensas obtenidas.

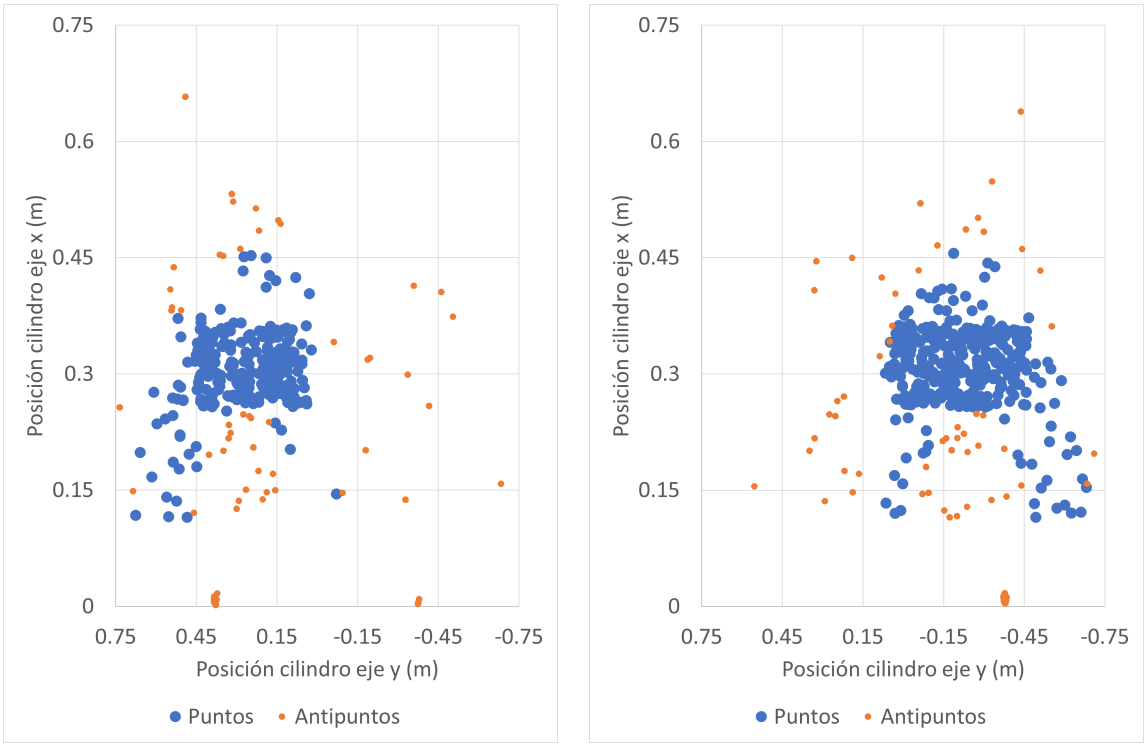


Figura 6.4: Puntos y antipuntos en el P-Node asociado a: a) *grasp_left*, b) *grasp_right*. Elaboración propia: Microsoft Excel

Con respecto a los P-Nodes, estos se comportaron de acuerdo con el resultado esperado. Por ejemplo, en la figura 6.4 se muestran los P-Nodos que se asocian con las acciones de tomar el cilindro. Se puede observar que existe una correspondencia directa con el espacio de trabajo que se determinó en la figura 4.13. Por otro lado,

también se puede observar una acumulación de puntos en el centro del espacio, que corresponde a la zona en la cual la *policy press_button* coloca a los cilindros que se encontraban fuera del alcance (figura 5.5). También se puede observar que la zona compartida por ambos manipuladores fue desarrollada principalmente por el manipulador derecho. Se observó que únicamente uno de los brazos toma los objetos de la zona compartida pero esa dominancia se alternó entre los dos manipuladores para las distintas ejecuciones. Ese fenómeno demuestra la flexibilidad con la que la memoria a largo plazo es capaz de aprender la tarea de manipulación.

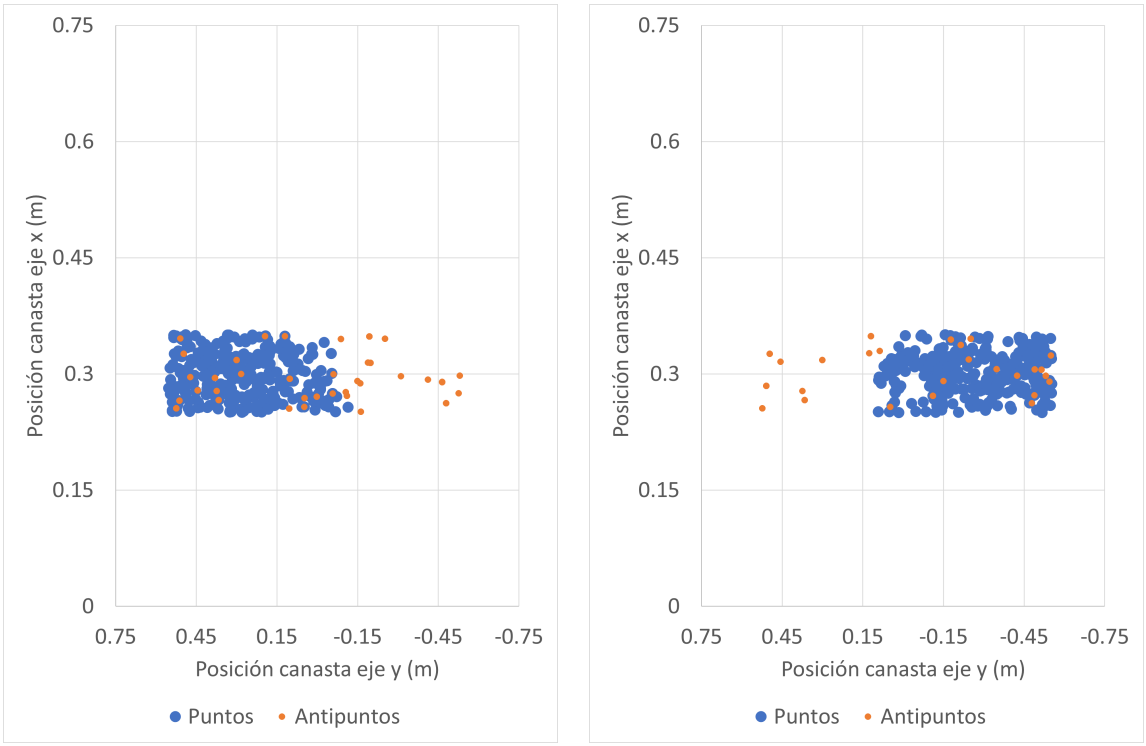


Figura 6.5: Puntos y antipuntos en el P-Node asociado a: a) *place_object_left*, b) *place_object_left*. Elaboración propia: Microsoft Excel

Otro ejemplo que muestra la consistencia de los P-Nodos con las condiciones del experimento se muestra en la figura 6.5, con los P-Nodos asociados a depositar el objeto en la canasta. Se observa que los puntos se dispusieron de forma rectangular, siguiendo exactamente la forma de la zona definida para la colocación de la canasta

(figura 5.9). Dentro de esa zona también se observan algunos antipuntos, pero corresponden a momentos en los que el robot no tenía el objeto sujetado con su gripper.

Con los resultados obtenidos en esta prueba de aprendizaje se puede afirmar con completa certeza que la herramienta diseñada cumple su objetivo. Se demostró que la integración entre la plataforma robótica y la arquitectura cognitiva MDB generan la habilidad de superar la tarea planteada. Sin embargo, el diseño actual no se limita a la tarea presentada en este experimento, sino que se abre la posibilidad de adaptar la herramienta a tareas más complejas que utilicen la interfaz desarrollada con el MDB.

CAPÍTULO 7

ANÁLISIS ECONÓMICO

Para determinar la validez del proyecto presentado debe analizarse su componente económica. En este capítulo se analizan los recursos invertidos durante este proyecto y sus principales beneficios. Debido a que el producto se concibió para aplicarse en investigación, se analizan los beneficios científicos que brinda la herramienta al LIANA. El análisis se complementa con la estimación del costo de una posible implementación física de OSCAR y se realiza una comparación con plataformas robóticas comerciales utilizadas para investigación. Material adicional a este análisis puede encontrarse en la sección A.5 de la bitácora de diseño.

Para el desarrollo del proyecto fue importante el uso de recursos de cómputo: una computadora Mac Pro perteneciente al LIANA y una computadora personal. Para estimar el costo de este recurso se utilizó el cálculo de la depreciación durante el proyecto, de acuerdo con los datos de la tabla 7.1. Durante las 16 semanas del proyecto el equipo Mac Pro se depreció por 44308 colones y la computadora personal por 18831 colones.

El uso de recurso de *software* también fue muy importante durante la ejecución del

Tabla 7.1: Valores de la depreciación del recurso de cómputo utilizado. (En colones)

Activo	Valor de compra	Valor Residual	Vida útil aproximada
Computadora MacPro	1200000	480000	5 años
Computadora Personal	510000	204000	5 años

proyecto. Sin embargo, el costo por licencias fue minimizado al utilizar *software* abierto de forma mayoritaria. La única licencia requerida correspondió al uso de SolidWorks, cuya licencia estudiantil tiene un costo de $99 anuales.

El principal recurso invertido en este proyecto fue el tiempo de desarrollo. Fue necesario dedicar 40 horas semanales para la ejecución del proyecto. En promedio se utilizó una hora semanal para retroalimentación con el cliente, 5 horas para documentar la herramienta y 34 horas para el diseño e implementación. Esto quiere decir que durante las 16 semanas de duración del proyecto se utilizaron aproximadamente 640 horas de trabajo en total. En la tabla 7.2 se resumen los costos incurridos para el desarrollo de OSCAR. El costo del tiempo de desarrollo se estimó según los salarios correspondientes a las asistencias estudiantiles del Tecnológico de Costa Rica.

Tabla 7.2: Costos de desarrollo del proyecto. (En colones)

No.	Detalle	Costo
1	Depreciación equipos	63138
2	Licencias de software	61200
3	Desarrollo	832000
	Total	956338

Como ya se consideraron los recursos utilizados, ahora se tratarán los beneficios obtenidos por el LIANA gracias a la ejecución de este proyecto. Se entregó al laboratorio una herramienta de experimentación en robótica cognitiva, la cual da al LIANA la capacidad de generar publicaciones científicas en el tema. Además, los beneficios generados por la herramienta no se traducen únicamente al corto plazo, gracias a la modularidad del *software*, el LIANA podrá desarrollar activamente nuevas capacidades sobre la herramienta. Los módulos de software utilizados tendrán soporte

activo hasta mayo del 2025, lo que significa que la herramienta se mantendrá vigente por lo menos durante cuatro años antes de requerir migraciones significativas.

Para finalizar el análisis económico, se presenta en la tabla 7.3 la estimación del costo de la implementación física de OSCAR. La información utilizada para la estimación fue la lista de materiales disponibles para el manipulador *Thor* [77] y los módulos de visión simulados, cuyos costos se estimaron de acuerdo con el precio dado por distribuidores locales. Se utilizó el criterio experto para realizar una estimación inicial del costo de las estructuras de montaje y la mano de obra necesaria para la implementación. Se estimó un 20% de gastos imprevistos debido a la temprana etapa del desarrollo en la cual se realiza esta estimación.

Tabla 7.3: Estimación del costo de la implementación física de la plataforma robótica.

No.	Componente	Cantidad	Costo Unitario ($)	Costo Total ($)
1	Manipulador Thor	2	786,75	1573,49
2	Realsense D435	1	236,25	236,25
3	Raspberry Cam V2	1	45,95	45,95
4	Estructuras de montaje	1	150,00	150,00
5	Mano de obra	1	650,00	650,00
6	Imprevistos	1	531,14	531,14
			Total	3186,83

Tabla 7.4: Estudio comparativo de distintos robots comerciales y OSCAR.

No.	Métrica	Unidad	Baxter	PR2	Fetch	OSCAR
1	Grados de libertad por brazo	Cantidad	7	7	7	6
2	Número de brazos	Cantidad	2	2	1	2
3	Alcance manipulador	mm	1210	800	940,5	422,5
4	Sensores equipados	Lista	- 5 x Cámaras RGB - Sensores de Fuerza - Sonar	-2 x Cámaras Stereo -Cámara RGB-D -3 x Cámara RGB -IMU -2 x Escáneres Láser -Sensores táctiles	-Láser 2D -Cámara RGB-D -2 x IMU	-Cámara RGB -Cámara RGB-D
5	Dimensiones máximas objeto a manipular	mm	150	90	100	50
6	Peso máximo objeto a manipular	g	2200	1800	5000	750
7	Costo plataforma robótica	$	25000	400000	100000	3200

En la tabla 7.4 se presentan las características de distintas plataformas robóticas comerciales y OSCAR. Se puede observar que la herramienta diseñada ofrece un costo significativamente menor que el de las plataformas robóticas comerciales estudiadas. El bajo costo permitirá que la implementación física sea viable a un menor plazo, sin embargo, es evidente que se obtienen prestaciones menores. A pesar de esto, se ha demostrado que con las características actuales de la herramienta se cumplen las necesidades del laboratorio a corto y mediano plazo.

CAPÍTULO 8

CONCLUSIONES

Al finalizar el proyecto se puede afirmar que se dotó al LIANA de capacidad experimental en robótica cognitiva. La herramienta desarrollada permitió el aprendizaje de una tarea de manipulación utilizando la arquitectura cognitiva MDB. Las principales conclusiones del proyecto fueron las siguientes:

- Se realizó un estudio de las necesidades del cliente, las cuales resultaron en un conjunto de especificaciones que permitieron guiar el desarrollo de la herramienta.
- Se realizó la implementación de una plataforma robótica en el simulador físico Gazebo que puede ser controlada mediante un API de ROS. Las evaluaciones mostraron un error de distancia promedio de 0.46 mm en las poses alcanzadas y una tasa de éxito del 99.69% en operaciones de *pick and place*.
- Se desarrolló un módulo de procesamiento de los datos sensoriales generados por el robot que permitió la localización de los objetos del ambiente con un error promedio de 1.55 mm.

- Se estimó que la implementación física de la herramienta simulada podría realizarse con una inversión inicial de entre 3.2% a 12.8% del costo de opciones comerciales disponibles.

- Se validó el funcionamiento de la herramienta a través de un experimento de aprendizaje integrado con la arquitectura cognitiva MDB, en el cual se logró obtener una tasa de recompensas superior al 80% a lo largo de cinco ejecuciones.

8.1 Recomendaciones

Este proyecto fue ideado como una implementación inicial de lo que serán las herramientas de experimentación cognitiva del LIANA a largo plazo, por lo que se tienen varios puntos de mejora:

- Adaptar la herramienta para mejorar su compatibilidad con el recurso de cómputo actual basado en el sistema operativo Mac OS.

- Debe expandirse la estrategia del cálculo de la orientación generada por el servidor de comandos de la plataforma robótica, de manera que puedan desarrollarse tareas más complejas.

- Se pueden utilizar los datos de la cámara RGB-D con los algoritmos perceptuales ofrecidos por MoveIt para la detección de colisiones.

- Deben explorarse opciones para aumentar las capacidades perceptuales de la herramienta, de forma que no se esté limitado a objetos simples en una mesa de trabajo fija.

- El manipulador *Thor* tiene un espacio de trabajo y destreza limitadas, para el desarrollo de tareas de mayor complejidad deben considerarse otras opciones de manipulador que ofrezcan mayores prestaciones.

- La implementación física de la herramienta debe diseñarse de manera que pueda reutilizar en el mayor grado los módulos de procesamiento generados en este proyecto. Para esto debe utilizar las interfaces de comunicación de ROS definidas.

REFERENCIAS BIBLIOGRÁFICAS

[1] J. L. Crespo Mariño, "Laboratorio de Inteligencia Artifical para las Ciencias Naturales." `https://www.tec.ac.cr/unidades/laboratorio-inteligencia-artificial-ciencias-naturales-liana`. Último Acceso: 25/11/20.

[2] G. Metta and A. Cangelosi, "Cognitive Robotics," in *Encyclopedia of the Sciences of Learning* (N. M. Seel, ed.), pp. 613–616, Boston, MA: Springer US, 2012.

[3] J. L. Crespo Mariño, *Una arquitectura de atención distribuida para agentes con sensorización multimodal*. PhD thesis, Universidade da Coruña, 2007.

[4] G. E. Barrantes Gómez, "Diseño de un sistema de visualización para la memoria a largo plazo en la arquitectura cognitiva Multilevel Darwinist Brain." Tesis de licenciatura, Instituto Tecnológico de Costa Rica, 2021. URL: `http://hdl.handle.net/2238/12345`.

[5] A. Cangelosi and M. Schlesinger, *Developmental Robotics: From Babies to Robots*. The MIT Press, 2015.

[6] R. Sun, *Anatomy of the Mind*. Oxford University Press, 1 ed., 2016.

[7] I. Kotseruba and J. K. Tsotsos, "40 years of cognitive architectures: core cognitive abilities and practical applications," *Artificial Intelligence Review*, vol. 53, pp. 17–94, 1 2020.

[8] A. Mishra and J. Son, "Cognitive Robotics: A Platform for Innovation," *IEEE Potentials*, vol. 38, pp. 39–42, 5 2019.

[9] M. I. Aldinhas Ferreira, "Cognitive Architectures: The Dialectics of Agent/Environment," in *Cognitive Architectures* (M. I. Aldinhas Ferreira, J. Silva Sequeira, and V. Rodrigo, eds.), pp. 1–12, Springer Nature Switzerland, 2019.

[10] J. von Uexkull, *Foray into the Worlds of Animals and Humans : With A Theory of Meaning*. Minneapolis: University of Minnesota Press, 2010.

[11] J. Rasmussen, "Skills, rules, and knowledge; signals, signs, and symbols, and other distinctions in human performance models," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-13, pp. 257–266, 5 1983.

[12] K. Shinohara, "Perceptual and Cognitive Processes in Human Behavior," in *Cognitive Neuroscience Robotics B* (M. Kasaki, H. Ishiguro, M. Asada, M. Osaka, and T. Fujikado, eds.), ch. 1, pp. 1–22, Tokyo: Springer Japan, 2016.

[13] A. M. Treisman and G. Gelade, "A feature-integration theory of attention," *Cognitive Psychology*, vol. 12, pp. 97–136, 1 1980.

[14] D. Kahneman, *Attention and Effort*. Englewood Cliffs, NJ: Prentice-Hall, 1973.

[15] S. Charron and E. Koechlin, "Divided Representation of Concurrent Goals in the Human Frontal Lobes," *Science*, vol. 328, pp. 360–363, 4 2010.

[16] F. Bellas, P. Caamaño, A. Faiña, and R. J. Duro, "Dynamic learning in cognitive robotics through a procedural long term memory," *Evolving Systems*, vol. 5, no. 1, pp. 49–63, 2014.

[17] M. Osaka, "Working Memory as a Basis of Consciousness," in *Cognitive Neuroscience Robotics B* (M. Kasaki, H. Ishiguro, M. Asada, M. Osaka, and T. Fujikado, eds.), ch. 3, pp. 39–57, Tokyo: Springer Japan, 2016.

[18] A. Baddeley, "Working Memory: Theories, Models, and Controversies," *Annual Review of Psychology*, vol. 63, pp. 1–29, 1 2012.

[19] R. J. Duro, J. A. Becerra, J. Monroy, and P. Caamano, "Considering Memory Networks in the LTM Structure of the Multilevel Darwinist Brain," in *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion*, (New York, NY, USA), pp. 1057–1060, ACM, 7 2016.

[20] J. M. Fuster, "Network memory.," *Trends in neurosciences*, vol. 20, pp. 451–9, 10 1997.

[21] D. Dong and S. Franklin, "A New Action Execution Module for the Learning Intelligent Distribution Agent (LIDA): The Sensory Motor System," *Cognitive Computation*, vol. 7, pp. 552–568, 10 2015.

[22] D. Vernon, C. von Hofsten, and L. Fadiga, *A Roadmap for Cognitive Development in Humanoid Robots*, vol. 11 of *Cognitive Systems Monographs*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011.

[23] F. Bellas, R. J. Duro, A. Faina, and D. Souto, "Multilevel Darwinist Brain (MDB): Artificial Evolution in a Cognitive Architecture for Real Robots," *IEEE Transactions on Autonomous Mental Development*, vol. 2, pp. 340–354, 12 2010.

[24] R. Salgado, F. Bellas, P. Caamaño, B. Santos-Díez, and R. J. Duro, "A procedural Long Term Memory for cognitive robotics: Optimizing adaptive learning in dynamic environments," *2012 IEEE Conference on Evolving and Adaptive Intelligent Systems, EAIS 2012 - Proceedings*, pp. 57–62, 2012.

[25] R. de Kleijn, G. Kachergis, and B. Hommel, "Robotic Action Control: On the Crossroads of Cognitive Psychology and Cognitive Robotics," in *Cognitive Robotics* (H. Samani, ed.), ch. 9, pp. 171–187, Taipei: CRC Press, 2016.

[26] G. Metta, L. Natale, F. Nori, G. Sandini, D. Vernon, L. Fadiga, C. von Hofsten, K. Rosander, M. Lopes, J. Santos-Victor, A. Bernardino, and L. Montesano, "The iCub humanoid robot: An open-systems platform for research in cognitive development," *Neural Networks*, vol. 23, pp. 1125–1134, 10 2010.

[27] J. Leitner, "A Bottom-Up Integration of Vision and Actions To Create Cognitive Humanoids," in *Cognitive Robotics* (H. Samani, ed.), ch. 10, pp. 191–214, Taipei: CRC Press, 2016.

[28] S. M. Nguyen, S. Ivaldi, N. Lyubova, A. Droniou, D. Gerardeaux-Viret, D. Filliat, V. Padois, O. Sigaud, and P.-Y. Oudeyer, "Learning to recognize objects through curiosity-driven manipulation with the iCub humanoid robot," in *2013 IEEE Third Joint International Conference on Development and Learning and Epigenetic Robotics (ICDL)*, pp. 1–8, IEEE, 8 2013.

[29] J. R. Wilson, E. Krause, M. Scheutz, and M. Rivers, "Analogical generalization of actions from single exemplars in a robotic architecture," *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, no. Aamas, pp. 1015–1023, 2016.

[30] S. Mohan, J. Kirk, and J. Laird, "A Computational Model for Situated Task Learning with Interactive Instruction," in *International Conference on Cognitive Modeling, 2013*, 2013.

[31] J. A. Becerra, A. Romero, F. Bellas, and R. J. Duro, "Motivational engine and long-term memory coupling within a cognitive architecture for lifelong open-ended learning," *Neurocomputing*, 11 2020.

[32] J. A. Becerra, R. J. Duro, and J. Monroy, "A Redescriptive Approach to Autonomous Perceptual Classification in Robotic Cognitive Architectures," *Proceedings of the International Joint Conference on Neural Networks*, vol. 2018-July, no. 640891, pp. 1–8, 2018.

[33] R. J. Duro, J. A. Becerra, J. Monroy, and F. Bellas, "Perceptual Generalization and Context in a Network Memory Inspired Long-Term Memory for Artificial Cognition," *International Journal of Neural Systems*, vol. 29, no. 6, pp. 1–26, 2019.

[34] R. Salgado, A. Prieto, F. Bellas, L. Calvo-Varela, and R. J. Duro, "Motivational engine with autonomous sub-goal identification for the Multilevel Darwinist Brain," *Biologically Inspired Cognitive Architectures*, vol. 17, pp. 1–11, 2016.

[35] A. Romero, J. A. Becerra, F. Bellas, and R. J. Duro, "Modulation Based Transfer Learning of Motivational Cues in Developmental Robotics," *Proceedings of the International Joint Conference on Neural Networks*, vol. 2019-July, no. July, pp. 1–8, 2019.

[36] A. Romero, F. Bellas, A. Prieto, and R. J. Duro, "Developmental Learning of Value Functions in a Motivational System for Cognitive Robotics," *Proceedings of the International Joint Conference on Neural Networks*, 2020.

[37] A. Romero, F. Bellas, J. A. Becerra, and R. J. Duro, "Motivation as a tool for designing lifelong learning robots," *Integrated Computer-Aided Engineering*, vol. 27, no. 4, pp. 353–372, 2020.

[38] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics*. Advanced Textbooks in Control and Signal Processing, London: Springer London, 2009.

[39] M. Mihelj, T. Bajd, A. Ude, J. Lenarčič, A. Stanovnik, M. Munih, J. Rejc, and S. Šlajpah, *Robotics*. Cham: Springer International Publishing, 2019.

[40] J. J. Craig, *Robótica*. México D.F.: Pearson Educación, 3 ed., 2006.

[41] S. R. Buss, "Introduction to Inverse Kinematics with Jacobian Transpose, Pseudoinverse and Damped Least Squares methods." Último Acceso: 26/05/21, 2009.

[42] K. Khokar, P. Beeson, and R. Burridge, "Implementation of KDL Inverse Kinematics Routine on the Atlas Humanoid Robot," *Procedia Computer Science*, vol. 46, pp. 1441–1448, 2015.

[43] H. Bruyninckx and P. Soetens, "Open Robot Control Software." `https://www.orocos.org/kdl.html`. Último Acceso: 26/05/21.

[44] I. A. Şucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library," *IEEE Robotics & Automation Magazine*, vol. 19, pp. 72–82, December 2012. `https://ompl.kavrakilab.org`.

[45] J. Kuffner and S. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 2, pp. 995–1001, IEEE, 2000.

[46] A. Elkady and T. Sobh, "Robotics Middleware: A Comprehensive Literature Survey and Attribute-Based Bibliography," *Journal of Robotics*, vol. 2012, pp. 1–15, 2012.

[47] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source Robot Operating System," in *ICRA workshop on open source software*, p. 5, 2009.

[48] W. S. Newman, *A Systematic Approach to Learning Robot Programming with ROS*. CRC Press, 2018.

[49] W. Qian, Z. Xia, J. Xiong, Y. Gan, Y. Guo, S. Weng, H. Deng, Y. Hu, and J. Zhang, "Manipulation task simulation using ROS and Gazebo," in *2014 IEEE International Conference on Robotics and Biomimetics (ROBIO 2014)*, pp. 2594–2598, IEEE, 12 2014.

[50] H. Choi, C. Crump, C. Duriez, A. Elmquist, G. Hager, D. Han, F. Hearl, J. Hodgins, A. Jain, F. Leve, C. Li, F. Meier, D. Negrut, L. Righetti, A. Rodriguez, J. Tan, and J. Trinkle, "On the use of simulation in robotics: Opportunities, challenges, and suggestions for moving forward," *Proceedings of the National Academy of Sciences*, vol. 118, p. e1907856118, 1 2021.

[51] Open Source Robotics Foundation, "Gazebo." `http://http://gazebosim.org/`. Último Acceso: 11/05/21.

[52] R. L. Smith, "Manual - ODE." `http://ode.org/wiki/index.php?title=Manual`, 2019. Último Acceso: 01/06/21.

[53] H. Gutiérrez Pulido and R. de la Vara Salazar, *Análisis y diseño de experimentos*. México D.F.: McGraw-Hill/Interamericana Editores S.A. de C.V., 2 ed., 2008.

[54] R. G. Budynas and J. K. Nisbett, *Diseño en ingeniería mecánica de Shigley*. McGraw-Hill/Interamericana Editores S.A. de C.V., 9 ed., 2012.

[55] K. T. Ulrich and S. D. Eppinger, *Diseño y desarrollo de productos*. México D.F.: McGraw-Hill/Interamericana Editores S.A. de C.V., quinta ed., 2013.

[56] Open Robotics, "ROS.org: Powering the world's robots." `https://www.ros.org/`. Último Acceso: 11/05/21.

[57] G. Metta, P. Fitzpatrick, and L. Natale, "YARP: Yet Another Robot Platform," *International Journal of Advanced Robotic Systems*, vol. 3, p. 8, 3 2006.

[58] OpenJAUS, "OpenJAUS Software Development Kit (SDK)." `https://openjaus.com/sdk/`. Último Acceso: 11/05/21.

[59] National Institute of Advanced Industrial Science and Technology, "OpenRTM-aist." `https://openrtm.org/openrtm/`. Último Acceso: 11/05/21.

[60] Coppelia Robotics, "Robot simulator CoppeliaSim." `https://coppeliarobotics.com/`. Último Acceso: 11/05/21.

[61] MathWorks, "MATLAB and Simulink for Robotics and Autonomous Systems." https://www.mathworks.com/solutions/robotics.html. Último Acceso: 11/05/21.

[62] Cyberbotics Ltd., "Webots: robot simulator." https://cyberbotics.com. Último Acceso: 11/05/21.

[63] PickNik Robotics, "MoveIt Motion Planning Framework." https://moveit.ros.org/. Último Acceso: 13/05/21.

[64] S. Chitta, E. Marder-Eppstein, W. Meeussen, V. Pradeep, A. Rodríguez Tsouroukdissian, J. Bohren, D. Coleman, B. Magyar, G. Raiola, M. Lüdtke, and E. Fernandez Perdomo, "ros_control: A generic and simple control framework for ROS," *The Journal of Open Source Software*, vol. 2, p. 456, 12 2017.

[65] Raspberry Pi Foundation, "Camera Module - Raspberry Pi Documentation." https://www.raspberrypi.org/documentation/hardware/camera/. Último Acceso: 24/05/21.

[66] Intel Corporation, "Intel Realsense Computer Vision - Depth and Tracking Cameras." https://www.intelrealsense.com/. Último Acceso: 24/05/21.

[67] Microsoft, "Azure Kinect DK hardware specifications." https://docs.microsoft.com/en-us/azure/kinect-dk/hardware-specification. Último Acceso: 24/05/21.

[68] OpenCV, "OpenCV modules." https://docs.opencv.org/4.5.2/. Último Acceso: 03/05/21.

[69] scikit-image development team, "scikit-image 0.18.0 docs." https://scikit-image.org/community_guidelines.html. Último Acceso: 03/05/21.

[70] R. C. Gerum, S. Richter, A. Winterl, C. Mark, B. Fabry, C. Le Bohec, and D. P. Zitterbart, "Cameratransform: A python package for perspective corrections and image mapping," *SoftwareX*, vol. 10, p. 100333, 2019.

[71] Labbé, M., "Find-Object." http://introlab.github.io/find-object, 2011. Último Acceso: 15/05/21.

[72] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *IEEE International Conference on Robotics and Automation (ICRA)*, (Shanghai, China), May 9-13 2011.

[73] R. c. Willow Garage, "ORK: Object Recognition Kitchen." https://wg-perception.github.io/object_recognition_core/. Último Acceso: 15/05/21.

[74] E. Davies, *Computer and Machine Vision*. Elsevier, 4 ed., 2012.

[75] D. Tyagi, "Introduction To Feature Detection And Matching." https://medium.com/data-breach/introduction-to-feature-detection-and-matching-65e27179885d, 2019. Último Acceso: 15/05/21.

[76] Free Software Foundation, "GNU General Public License." https://www.gnu.org/licenses/gpl-3.0.html, 2007. Último Acceso: 01/06/21.

[77] AngelLM, "Thor: OpenSource 3D printable Robotic Arm." https://hackaday.io/project/12989-thor. Último Acceso: 01/05/21.

[78] J. Moreno, E. Clotet, R. Lupiañez, M. Tresanchez, D. Martínez, T. Pallejà, J. Casanovas, and J. Palacín, "Design, Implementation and Validation of the Three-Wheel Holonomic Motion System of the Assistant Personal Robot (APR)," *Sensors*, vol. 16, p. 1658, 10 2016.

[79] P. Waugh, "Openness Rating: How Open Is Your Software Project?," 2014.

[80] J. Bonvoisin, R. Mies, J.-F. Boujut, and R. Stark, "What is the "Source" of Open Source Hardware?," *Journal of Open Hardware*, vol. 1, 9 2017.

[81] IEEE Robots, "Baxter." https://robots.ieee.org/robots/baxter/. Último Acceso: 24/02/21.

[82] M. Maggiali, L. Fiorio, U. Pattacini, and A. Derito, "Technical Specifications of the iCub Platform." https://bit.ly/3bFCjYu, 2019. Último Acceso: 24/02/21.

[83] Willow Garage, "PR2: Hardware Specs." http://www.willowgarage.com/pages/pr2/specs. Último Acceso: 24/02/21.

[84] Willow Garage, "KUKA youBot: Research & Application Development in Mobile Robotics." https://www.generationrobots.com/img/Kuka-YouBot-Technical-Specs.pdf. Último Acceso: 24/02/21.

[85] FLIR, "FLIR PackBot." https://www.flir.com/products/packbot/. Último Acceso: 24/02/21.

[86] ABB, "Technical data IRB 14000 YuMi." https://bit.ly/3sqcF0G. Último Acceso: 24/02/21.

[87] Fetch Robotics, "Fetch Mobile Manipulator." https://fetchrobotics.com/robotics-platforms/fetch-mobile-manipulator/. Último Acceso: 24/02/21.

[88] Kawada Robotics, "NEXTAGE." https://www.kawadarobot.co.jp/en/nextage/. Último Acceso: 24/02/21.

[89] PAL Robotics, "TIAGo Technical Specifications." https://pal-robotics.com/wp-content/uploads/2020/05/TIAGo-Datasheet.pdf. Último Acceso: 24/02/21.

[90] Universal Robots, "Universal Robot UR3e." https://www.universal-robots.com/products/ur3-robot/. Último Acceso: 01/05/21.

[91] Yaskawa, "GP8: Six-Axis High-Speed Assembly and Handling Robot." https://www.motoman.com/en-us/products/robots/industrial-robots/assembly-handling/gp-series/gp8. Último Acceso: 01/05/21.

[92] Robotis, "OpenMANIPULATOR-X." https://emanual.robotis.com/docs/en/platform/openmanipulator_x/overview/. Último Acceso: 01/05/21.

[93] Robotis, "OpenMANIPULATOR-P." https://emanual.robotis.com/docs/en/platform/openmanipulator_p/overview/. Último Acceso: 01/05/21.

[94] Mecademic, "Meca500 Six-Axis Industrial Robot Arm." https://www.mecademic.com/en/meca500-robot-arm. Último Acceso: 01/05/21.

[95] Barrett Advanced Robotics, "WAM Arm." https://advanced.barrett.com/wam-arm-1. Último Acceso: 01/05/21.

[96] Trossen Robotics, "PincherX 150 Robot Arm." https://www.trossenrobotics.com/pincherx-150-robot-arm.aspx. Último Acceso: 01/05/21.

[97] Trossen Robotics, "ReactorX 150 Robot Arm." https://www.trossenrobotics.com/reactorx-150-robot-arm.aspx. Último Acceso: 01/05/21.

[98] RT Corporation, "Robot de brazo CRANE-X7." https://rt-net.jp/products/crane-x7/. Último Acceso: 01/05/21.

[99] Svenza Robotics, "Revel Robotic Arm." https://svenzva.com/revel/. Último Acceso: 01/05/21.

[100] Han's Robot, "Cute Educational Robot." https://www.hansrobot.net/products/detail-4.html. Último Acceso: 01/05/21.

[101] Robotis, "Manipulator-H." https://emanual.robotis.com/docs/en/platform/manipulator_h/introduction/. Último Acceso: 01/05/21.

[102] Kinova, "Kinova Jaco Assistive Robotic Arm." https://www.kinovarobotics.com/en/assistive-technologies/column-a1/kinova-assistive-robotic-arm. Último Acceso: 01/05/21.

[103] Robai Corporation, "Cyton Gamma 1500 Arm Specifications." https://datasheet.octopart.com/CYTON-GAMMA-1500-Robai-datasheet-67184669.pdf, 2015. Último Acceso: 01/05/21.

[104] ST Robotics, "Low cost 5 and 6 axis small collaborative robot arm." https://strobotics.com/small-articulated-robot.htm. Último Acceso: 01/05/21.

[105] BCN3D, "BCN3D MOVEO – Un brazo robótico de código abierto impreso en 3D." https://www.bcn3d.com/es/bcn3d-moveo-un-brazo-robotico-de-codigo-abierto-impreso-en-3d/. Último Acceso: 01/05/21.

[106] Niryo, "Niryo One, a 6 axis robot designed for education and research." https://niryo.com/product/niryo-one/. Último Acceso: 01/05/21.

[107] C. Annin, "Annin Robotics - open source 6 axis robots you can build yourself." https://www.anninrobotics.com/. Último Acceso: 01/05/21.

[108] Trossen Robotics, "WidowX 250 Robot Arm 6DOF." https://www.trossenrobotics.com/widowx-250-robot-arm-6dof.aspx. Último Acceso: 03/05/21.

APÉNDICE A: BITÁCORA DE DISEÑO

A.1 Introducción

En este apéndice se recoge toda la documentación realizada en la etapa correspondiente al diseño conceptual de la herramienta. La organización de este documento sigue las secciones de la metodología expuesta en el capítulo 3 del documento principal.

En la sección de necesidades se presentan las necesidades interpretadas a partir de una serie de entrevistas y de investigación bibliográfica. En la sección de especificaciones se presentan las métricas establecidas, plataformas robóticas evaluadas como referencia y finalmente los valores propuestos como especificaciones objetivo. En el estudio de conceptos se comienza presentando las descomposiciones funcionales utilizadas, posteriormente se detallan los conceptos estudiados en cada subproblema. Se finaliza presentando material correspondiente al estudio económico y las especificaciones finales de la herramienta.

A.2 Necesidades

A.2.1 Entrevistas

Los cuadros A.1 y A.2 presentan las necesidades interpretadas a partir de las entrevistas realizadas con el cliente.

Tabla A.1: Tabla de necesidades interpretadas del cliente.

Cliente	Juan Crespo/Ronald Loaiza	
Fecha Entrevista	15 de Octubre, 2020	
Pregunta/Sugerencia	**Enunciado del Cliente**	**Necesidad Interpretada**
Capacidades herramienta	"que sepamos que es una plataforma que puede desempeñarse en un entorno lo más realista posible de acuerdo a experimentos que se hayan hecho en campo, a partir de la determinación de requerimientos."	La herramienta permite realizar experimentos similares a los realizados por otros investigadores
Capacidades herramienta	"me resultaría muy útil que el robot sea móvil .... Un baxter pequenito que se mueva. Me interesa un tema que poca gente se ha tomado en serio que es el de la propiocepción, saber distinguir cuando es uno el que se mueve y no una influencia externa. Y la principal acción donde se da esto es en el movimiento"	La plataforma robótica tiene capacidad de movimiento
Capacidades herramienta	"debe tener capacidades de cambio del entorno elevadas (brazos, articulaciones), pero que al mismo tiempo pueda moverse"	La plataforma robótica tiene capacidades de manipulación
Capacidades herramienta	"sería interesante que el robot pueda conectarse con un host o máquina remota"	La herramienta permite el procesamiento en un computador externo
Capacidades herramienta	"que sea replicable .... hay muchas cosas que no se pueden trabajar porque solo se tiene uno"	La herramienta robótica es de bajo costo
Enfoque de la investigación	"nos interesa trabajar en el tema general de una arquitectura cognitiva y especialmente los temas de plasticidad en los esquemas de memoria a corto y largo plazo, que sean flexibles"	La herramienta permite integrarse con una arquitectura cognitiva
Entregables del proyecto	"si el resultado es una herramienta de experimentación, el manual debe estar contemplado. Incluso yo hablaría de una documentación de la herramienta"	La herramienta tiene un manual de uso
Entregables del proyecto	"nos estamos creando una herramienta, utilísima, desde el momento que tengamos el control completo sobre el desarrollo de la herraminenta; sabemos como funciona, es adecuada para nosotros, si en un futuro hay que cambiarla tenemos todo el proceso intelectual que ha llevado a la misma."	La herramienta tiene una documentación de su funcionamiento
Entregables del proyecto	"nos estamos creando una herramienta, utilísima, desde el momento que tengamos el control completo sobre el desarrollo de la herraminenta; sabemos como funciona, es adecuada para nosotros, si en un futuro hay que cambiarla tenemos todo el proceso intelectual que ha llevado a la misma."	La herramienta es de desarrollo abierto

A.2.2 Síntesis

Para complementar las necesidades identificadas a través del proceso de entrevistas se incluyeron una serie de recomendaciones para el desarrollo de sistemas cognitivos presentes en la literatura [22]. Estos criterios son presentados en el cuadro A.3. Las pautas de diseño se organizan en las categorías de encarnación, percepción, acción y autonomía. En el cuadro A.5 se presentan las necesidades identificadas. Estas fueron agrupadas para generar necesidades primarias y secundarias. El nivel de importancia de las necesidades secundarias se presenta de acuerdo con la escala presentada en el cuadro A.4.

Tabla A.2: Tabla de necesidades interpretadas del cliente.

Cliente	Juan Crespo/Ronald Loaiza	
Fecha Entrevista	22 de Enero, 2021	
Pregunta/Sugerencia	**Enunciado del Cliente**	**Necesidad Interpretada**
Procesamiento del lenguaje y capacidades de nivel alto	...entiendo percepción como la sensorización llevada a un ente de nivel superior...	El sistema perceptual genera características a partir de la sensorización del robot
Procesamiento del lenguaje y capacidades de nivel alto	"Si me parece que más importante la capacidad de actuación y sensorización desde el punto de vista del hardware y la comunicación con un host externo"	La plataforma robótica cuenta con capacidades de sensorización
Procesamiento del lenguaje y capacidades de nivel alto	"Si me parece que más importante la capacidad de actuación y sensorización desde el punto de vista del hardware y la comunicación con un host externo"	La plataforma robótica cuenta con capacidades de sensorización
Modelo Skill-Rule-Knowledge	"Del diagrama ya se llegaría con los feature formation listos... y me parecen importantes el nivel intermedio y superior (Rule y Knowledge) como un primer acercamiento"	El sistema perceptual permite la extracción de características del entorno
Experimentos deseados	"Los experimentos que nos queremos enfrentar como agarres básicos, nos queremos orientar a eso"	La herramienta permite ejecutar experimentos de manipulación básicos
Descripción de la plataforma esperada	"La definición es: un robot que; sin perder de vista la capacidad de hacer otros experimentos; pero que pueda realizar experimentos de manipulación ambidextra de objetos sobre los cuales se pueden realizar acciones y realimentados por medio de visión ...."	La plataforma robótica tiene una morfología que le permite la manipulación ambidiestra
Descripción de la plataforma esperada	"La definición es: un robot que; sin perder de vista la capacidad de hacer otros experimentos; pero que pueda realizar experimentos de manipulación ambidextra de objetos sobre los cuales se pueden realizar acciones y realimentados por medio de visión ...."	El sistema perceptual recibe datos mediante un sistema de visión
Descripción de la plataforma esperada	"... O dicho de otra manera, dos brazos que se van a mover y que tienen un conjunto de mano/brazo que le permiten hacer varias cosas, agarrar, levantar, soltar, tirar ...."	El sistema de acciones permite la ejecución de movimientos básicos
Descripción de la plataforma esperada	"... Y que reciben una realimentación por medio de un sistema de visión, de forma que puedan aprender a hacer acciones por refuerzo del estilo le doy una pelotita y un vaso y veo cuánto tarda en darse cuenta que tiene que colocar la pelotita en el vaso o acciones donde vaya viendo diferentes conjuntos de percepción/accion y una vez que hayamos visto que ha aprendido a hacer 1, 2, 3 y 4 cuando le volvemos a presentar la primera nos damos cuenta que no ha olvidado."	La herramienta permite el aprendizaje de tareas por refuerzo
Descripción de la plataforma esperada	"La capacidad de manipulación, por lo menos de un brazo yo la veo completamente imprescindible"	La herramienta permite la manipulación de objetos
Complejidad del sistema de acción y percepción	"En tanto otras cosas se le puedan añadir. Que de alguna forma sea modulable. Que yo si pueda en un futuro considerar otras percepciones y añadirselas a la misma estructura."	La herramienta distribuye sus capacidades en módulos

Tabla A.3: Colección de recomendaciones seleccionadas para el proyecto.

No.	**Pautas de diseño [22]**
	Encarnación (Agente robótico)
G1	Colección amplia de interfaces motoras y de sensorización
G3	Morfología Humanoide
	Percepción
G15	Discriminación de objetos
	Acción
G10	Movimientos se organizan mediante acciones
G14	Los movimientos se organizan para disminuir los grados de libertad
	Autonomía
G2	Presenta procesos propios de homeóstasis
G30	Tiene un conjunto de comportamientos innatos para la exploración y supervivencia
G43	Se da la operación concurrente de los componentes y subsistemas

Tabla A.4: Escala de importancia.

1	Es indeseable
2	No es importante
3	Es deseable
4	Es importante
5	Es imprescindible

Tabla A.5: Necesidades establecidas para el proyecto.

No.	Necesidades Interpretadas Cliente	Importancia
1.	**La herramienta incluye la implementación de una plataforma robótica**	
1.1	La plataforma robótica cuenta con capacidades de sensorización	5
1.2	La plataforma robótica tiene herramientas para la manipulación	5
1.3	La plataforma robótica tiene una morfología que le permite la manipulación ambidiestra	3
1.4	La plataforma robótica es de bajo costo	3
1.5	La plataforma robótica tiene capacidad de locomoción	2
2.	**La herramienta permite la obtención de datos perceptuales**	
2.1	El sistema perceptual extrae características del entorno a partir de datos sensoriales	5
2.2	El sistema perceptual recibe datos mediante un sistema de visión	4
3.	**La herramienta es capaz de controlar las acciones ejecutadas**	
3.1	El sistema de acciones permite la ejecución de movimientos básicos	5
3.2	El sistema de acciones permite la manipulación de objetos	5
4.	**La herramienta permite la ejecución de experimentos de aprendizaje cognitivo**	
4.1	La herramienta permite ejecutar experimentos de manipulación básicos	5
4.2	La herramienta permite integrarse con una arquitectura cognitiva	5
4.3	La herramienta permite el aprendizaje de tareas por refuerzo	3
5.	**La herramienta se implementa en una arquitectura de software modular**	
5.1	La herramienta permite el procesamiento en un computador externo	4
5.2	La herramienta permite la interfaz de sensores y actuadores con un host de procesado	3
5.3	La herramienta distribuye sus capacidades en módulos	3
6.	**La herramienta cuenta con documentación adecuada**	
6.1	La herramienta tiene un manual de uso	5
6.2	La herramienta tiene documentado su diseño e implementación	4
6.3	La herramienta es de desarrollo abierto	4

A.3 Especificaciones

A.3.1 Métricas

Para medir el grado de cumplimiento de las necesidades se desarrolló un conjunto de métricas presentado en el cuadro A.6. Cada una de estas responde a una necesidad específica y adicionalmente puede hacer referencia a alguna de las pautas presentadas en el cuadro A.3. La escala de importancia asociada es la presentada en el cuadro A.4. Algunas métricas hacen referencia en paréntesis a pruebas planteadas, estas son presentadas en el cuadro A.7 y se detallan la sección 3.4 del documento principal.

La métrica 9 hace referencia al nivel de movilidad de la plataforma robótica. Para realizar esta clasificación se tomó en cuenta el tipo de actuador utilizado para generar locomoción y el tipo de movimiento disponible para la plataforma, de acuerdo con las clasificaciones presentadas en [78, Tab. 1] y [38, p. viii]. En el cuadro A.8 se detallan los niveles de la escala.

Tabla A.6: Métricas establecidas para el proyecto.

No.	No. Necesidad	Pauta	Métrica	Unidad	Imp.
1	1.2/1.3	G1/G3	Grados de libertad por brazo	Cantidad	5
2	1.3	G1/G3	Número de brazos	Cantidad	3
3	1.2		Alcance manipulador	mm	4
4	1.1/2.2	G1	Sensores equipados	Lista	5
5	1.2		Dimensiones máximas objeto a manipular	mm	4
6	1.2		Peso máximo objeto a manipular	g	3
7	1.2		Velocidad manipulador (P1)	m/s	4
8	1.4		Costo plataforma robótica	$	3
9	1.5	G1	Clasificación movilidad	Ver Clasificación.	2
10	1.2/3.1	G14	Precisión en la pose del manipulador (P1)	mm	5
11	2.2	G1	Campo de visión	mm	4
12	2.2	G1	Resolución sistema visión	mm	4
13	3.2/4.1	G10	Aciertos en la toma de objetos (P2)	% aciertos/intentos	5
14	2.1	G15	Error en la prueba de percepción (P3)	mm	5
15	4.1/4.3		Proporción de acciones recompensadas (P4)	% recompensas/acciones	5
16	4.2	G2/G30	Estructuras cognitivas integradas	Lista	5
17	2/5	G15	Tiempo procesamiento de percepciones (P3)	ms	3
18	5.1/5.2		Confiabilidad de la comunicación	% datos recibidos/datos enviados	3
19	5.3	G43	Módulos ejecutados de forma concurrente	Lista	3
20	6.1		Opinión del cliente sobre manual de uso	Subjetivo	5
21	6.2		Aspectos detallados documentación de diseño	Lista	4
22	6.3		Puntuación de apertura componentes Hardware	Ver Puntuación.	4
23	6.3		Puntuación de apertura módulos de Software	Puntuación OSS Watch.	4

Tabla A.7: Pruebas a realizar.

P1	Caracterización de manipuladores
P2	Toma de objetos
P3	Percepción visual
P4	Aprendizaje

Tabla A.8: Escala de movilidad.

Categoría	Descripción
0	Completamente rígido y fijo (0 DoF)
1	Manipulador fijo
2	Manipulador móvil (Ruedas, No Holonómico)
3	Manipulador móvil (Ruedas, Holonómico)
4	Manipulador móvil (Piernas)
5	Manipulador móvil (Aéreo)

Las métricas 22 y 23 hacen referencia al uso de módulos de *software* y *hardware* abiertos. Para cuantificar el grado de aprtura del software utilizado se hará uso de la escala propuesta por *OSS Watch* [79]. Por otro lado, para establecer la apertura de componentes de hardware se utilizará la escala de ocho puntos propuesta por [80]. En el cuadro A.9 se presentan los criterios considerados por la escala, cumplir cada criterio otorga un punto, para una calificación máxima de 8.

Tabla A.9: Criterios para la clasificación de *hardware* abierto.

Criterio	Descripción
a	Archivos CAD disponibles
b	Archivos CAD editables
c	Instrucciones de Ensamblaje Disponibles
d	Instrucciones de Ensamblaje Editables
e	Lista de materiales disponible
f	Lista de materiales editable
g	Guías para la colaboración
h	Uso comercial permitido

A.3.2 Información comparativa

Para obtener referencia sobre los valores de las métricas establecidas se recabó información sobre distintas plataformas robóticas. Estos valores se utilizaron posteriormente para generar las especificaciones objetivo del proyecto. En los cuadros A.10, A.11, A.12 se muestran los valores de las métricas establecidas en distintas plataformas robóticas comerciales.

Tabla A.10: Comparativa de plataformas robóticas.

No.	Métrica	Unidad	Baxter [81]	iCUB [82]	PR2 [83]
1	Grados de libertad por brazo	Cantidad	7	16	7
2	Número de brazos	Cantidad	2	2	2
3	Alcance manipulador	mm	1210		800
4	Sensores equipados	Lista	- 5 Cámaras - Sensores de Fuerza - Sonar	-Cámaras Stereo -IMU -Micrófono -Sensores fuerza y torque -Sensores táctiles -Encoders	-2 pares Cámaras Stereo -Sensor Profundidad -IMU -Cámara 5MP -2 x Escáner Láser -Sensores táctiles
5	Dimensiones máximas objeto a manipular	mm	150		90
6	Peso máximo objeto a manipular	g	2200		1800
7	Velocidad manipulador (P1)	m/s			
8	Costo plataforma robótica	$	25000	300000	400000
9	Clasificación movilidad	Ver Clasificación.	1	4	3
10	Precisión en la pose del manipulador (P1)	mm		1	
22	Puntuación de apertura componentes Hardware	Ver Puntuación.	0	2	0

Tabla A.11: Comparativa de plataformas robóticas (continuación).

No.	Métrica	Unidad	YouBot [84]	Packbot [85]	IRB 14000 [86]
1	Grados de libertad por brazo	Cantidad	5	4	7
2	Número de brazos	Cantidad	1 ó 2	1	2
3	Alcance manipulador	mm	540.5	1870	559
4	Sensores equipados	Lista	-Ninguno	-4 Cámaras -IMU -GPS y brújula -Sensores sonido	- 2x Cognex AE3 In-Sight
5	Dimensiones máximas objeto a manipular	mm	70		50
6	Peso máximo objeto a manipular	g	500	5000	256
7	Velocidad manipulador (P1)	m/s			1.5
8	Costo plataforma robótica	$		150000	40000
9	Clasificación movilidad	Ver Clasificación.	3	2	1
10	Precisión en la pose del manipulador (P1)	mm	0.1		0.02
22	Puntuación de apertura componentes Hardware	Ver Puntuación.	0	0	0

Tabla A.12: Comparativa de plataformas robóticas (continuación).

No.	Métrica	Unidad	Fetch [87]	NEXTAGE [88]	TIAGo++ [89]
1	Grados de libertad por brazo	Cantidad	7	6	7
2	Número de brazos	Cantidad	1	2	2
3	Alcance manipulador	mm	940,5	625	870
4	Sensores equipados	Lista	-Láser 2D -Cámara RGB-D -2x IMU (gripper y base)	-Cámaras Stereo -Cámara Mano	-Cámara RGB-D -Láser y Sonar
5	Dimensiones máximas objeto a manipular	mm	100		
6	Peso máximo objeto a manipular	g	5000	2500	3000
7	Velocidad manipulador (P1)	m/s	1		
8	Costo plataforma robótica	$	100000	70000	60000
9	Clasificación movilidad	Ver Clasificación.	2	1	2
10	Precisión en la pose del manipulador (P1)	mm		0,03	
22	Puntuación de apertura componentes Hardware	Ver Puntuación.	0	0	0

A.3.3 Valores objetivo

En el cuadro A.13 se presentan los valores objetivos planteados para las métricas establecidas. Estos fueron establecidos utilizando la información recabada y diversos criterios, en el cuadro A.14 se comentan los aspectos considerados al establecer el valor de cada métrica.

Tabla A.13: Especificaciones objetivo planteadas.

No.	Métrica	Unidad	Imp.	Valor marginal	Valor ideal
1	Grados de libertad por brazo	Cantidad	5	>5	>6
2	Número de brazos	Cantidad	3	1	2
3	Alcance manipulador	mm	4	>300	>500
4	Sensores equipados	Lista	5	-Sistema Visión -Sensores Absolutos Joints	-Sistema Visión -Sensores Absolutos Joints -Sensores Presencia Gripper -Sensores de Fuerza y Torque
5	Dimensiones máximas objeto a manipular	mm	4	>25	>50
6	Peso máximo objeto a manipular	g	3	>200	>500
7	Velocidad manipulador (P1)	m/s	4	>0.25	>1
8	Costo plataforma robótica	$	3	<8500	<6250
9	Clasificación movilidad	Ver Clasificación.	2	$\geq$1	$\geq$2
10	Precisión en la pose del manipulador (P1)	mm	5	<0.5	<0.1
11	Campo de visión	mm	4	>500x650	>1000x1300
12	Resolución sistema visión	mm	4	<1.5	<1
13	Aciertos en la toma de objetos (P2)	% aciertos/intentos	5	>95	100
14	Error en la prueba de percepción (P3)	mm	5	<3	<2
15	Proporción de acciones recompensadas (P4)	% recompensas/acciones	5	>70	>80
16	Estructuras cognitivas integradas	Lista	5	-LTM	-Modelador Mundo -MotivEn -LTM
17	Tiempo procesamiento de percepciones (P3)	ms	3	<1000	<500
18	Confiabilidad de la comunicación	% datos recibidos/datos enviados	3	>95	100
19	Módulos ejecutados de forma concurrente	Lista	3	-Percepción -Acción	-Percepción -Acción -Aprendizaje y Memoria
20	Opinión del cliente sobre manual de uso	Subjetivo	5	>3	>4
21	Aspectos detallados documentación de diseño	Lista	4	-Misión -Necesidades -Especificaciones -Conceptos -Informe Pruebas -Análisis Económico	-Misión -Necesidades -Especificaciones -Conceptos -Informe Pruebas -Análisis Económico
22	Puntuación de apertura componentes Hardware	Ver Puntuación.	4	>4	>6
23	Puntuación de apertura módulos de Software	Puntuación OSS Watch.	4	>60	>75

Tabla A.14: Comentarios sobre valores establecidos.

No.	Métrica	Comentario
1	Grados de libertad por brazo	Grados de libertad brazo antropomórfico con mínimo una muñeca de 2DoF para destreza [38, p. 8-10].
2	Número de brazos	Preferible la utilización de dos brazos para obtener morfología humanoide.
3	Alcance manipulador	Valores intermedios encontrados para los brazos comerciales y plataformas robóticas de pequeña escala investigados.
4	Sensores equipados	El sistema de visión es necesario para dotar de percepciones al robot, los encoders de los joints para poder realizar control por realimentación del brazo robótico y evitar el drift. La sensorización ideal dotaría a la herramienta de capacidades superiores para desarrollar al largo plazo.
5	Dimensiones máximas objeto a manipular	Valores intermedios encontrados para los brazos comerciales y plataformas de pequeña escala robóticas investigados.
6	Peso máximo objeto a manipular	Valores intermedios encontrados para los brazos comerciales y plataformas de pequeña escala robóticas investigados.
7	Velocidad manipulador (P1)	Valores intermedios encontrados para los brazos comerciales y plataformas de pequeña escala robóticas investigados.
8	Costo plataforma robótica	Se buscará desarrollar una plataforma a un costo entre la tercera y cuarta parte del costo de un Baxter
9	Clasificación Movilidad	Se espera como mínimo un manipulador fijo. Una plataforma con capacidades básicas de locomoción proporcionaría capacidades amplias a largo plazo.
10	Precisión en la pose del manipulador (P1)	Valores intermedios encontrados para los brazos comerciales y plataformas de pequeña escala robóticas investigados.
11	Campo de visión	Basado en campo de visión sensor Kinect en posición vertical a 60cm y 1,2m del plano de trabajo.
12	Resolución sistema visión	Aproximado asumiendo un tamaño de 1280x1024 píxeles. Se hace la relación entre campo de visión y número de píxeles para obtener un estimado.
13	Aciertos en la toma de objetos (P2)	Resultados esperados de la prueba.
14	Error en la prueba de percepción (P3)	La percepción se espera que tenga un error mayor que la resolución del sistema de visión por los errores presentes en el procesado.
15	Proporción de acciones recompensadas (P4)	Aproximado al obtenido en el experimento presentado en [33, Fig. 6a].
16	Estructuras cognitivas integradas	Se tiene acceso directo al codigo del LTM y del MotivEn. Para disminuir la complejidad, y considerando el hecho de que la integración dentro de la propia arquitectura se encuentra en fase experimental [31], es mejor comenzar integrando sólo la estructura del LTM.
17	Tiempo procesamiento de percepciones (P3)	Un tiempo "razonable", en los experimentos del MDB a la sincronización temporal no se le ha dado mucha importancia.
18	Confiabilidad de la comunicación	Se espera una alta confiabilidad en la comunicación entre los módulos
19	Módulos ejecutados de forma concurrente	Los módulos de percepción y acción deben ser ejecutados de forma paralela. Sin embargo, si se dan problemas con la integración con la arquitectura se podría optar por hacer el aprendizaje de forma offline.
20	Opinión del cliente sobre manual de uso	Suponiendo una encuesta de 1 a 5 aplicada a distintos usuarios donde se pide seguir el manual y evaluar aspectos como claridad, profundidad, facilidad, etc.
21	Aspectos detallados documentación de diseño	Basado en la tabla de contenido de una *Bitácora de producto* presentada en [55, p. 377].
22	Puntuación de apertura componentes Hardware	Se espera tener los archivos CAD editables y una lista de materiales que faciliten la implementación futura y la estimación del costo.
23	Puntuación de apertura módulos de Software	Se espera tener acceso de uso libre y sin licencia de los módulos de software que se utilicen. Además estos deben tener una documentación "aceptable" para facilitar su uso.

A.4 Estudio de conceptos

A.4.1 Descomposiciones funcionales

En las siguientes figuras se presentan las descomposiciones funcionales utilizadas. La figura A.1 es la descomposición principal del proyecto en subsistemas. En la figura A.2 se presenta la descomposición funcional del subsistema *plataforma robótica* y la del subsistema *ambiente* se muestra en la figura A.3. La descomposición del sistema de redescripción se presenta en la figura A.4, que fue subdividido como se muestra en las figuras A.5 y A.6.

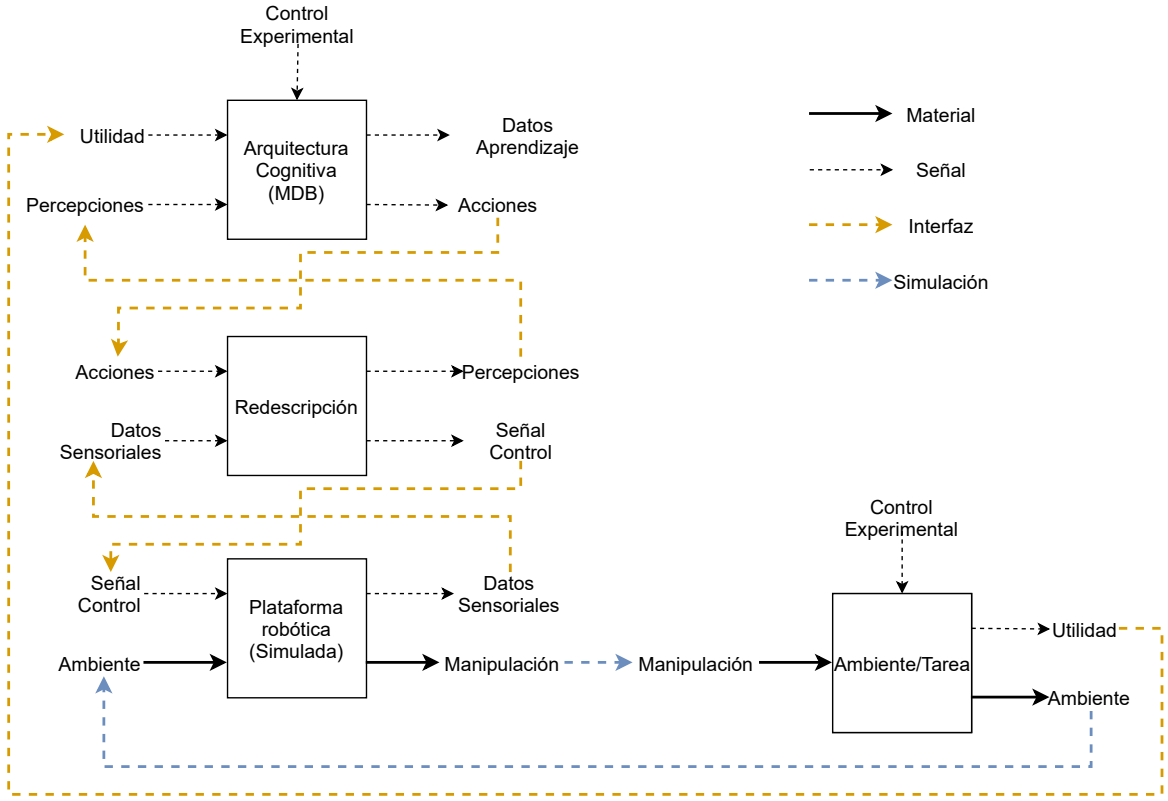


Figura A.1: Diagrama de bloques identificado para el proyecto. (Repetido de figura 3.3) Elaboración propia

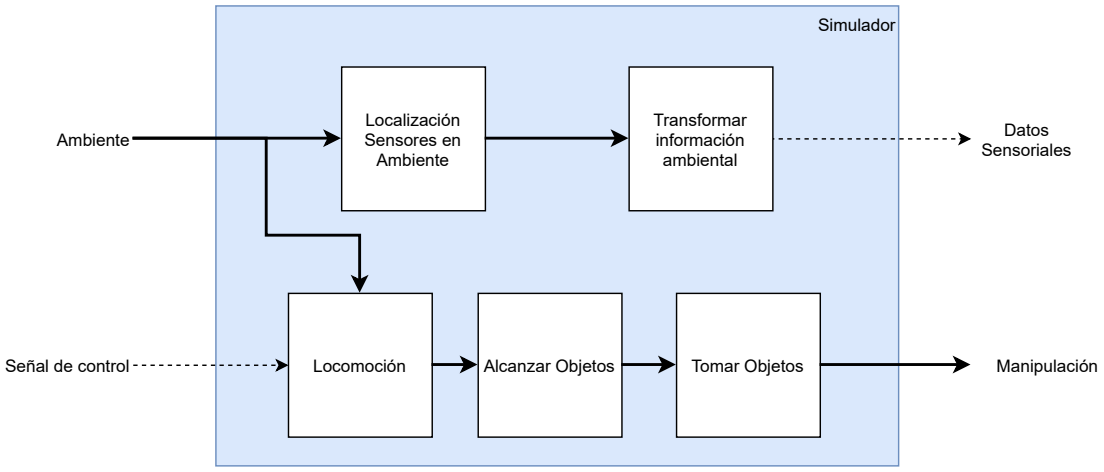


Figura A.2: Descomposición funcional del subsistema *plataforma robótica*. Elaboración propia

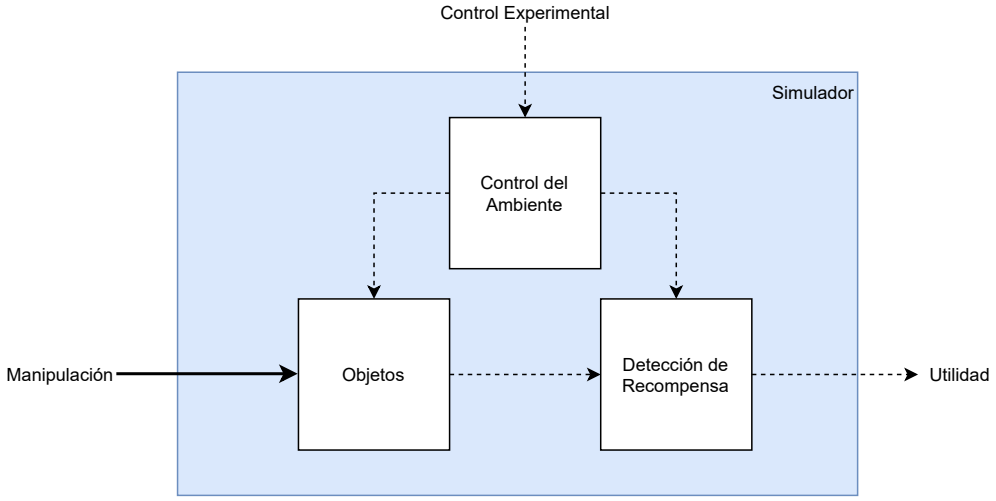


Figura A.3: Descomposición funcional del subsistema *ambiente*. Elaboración propia

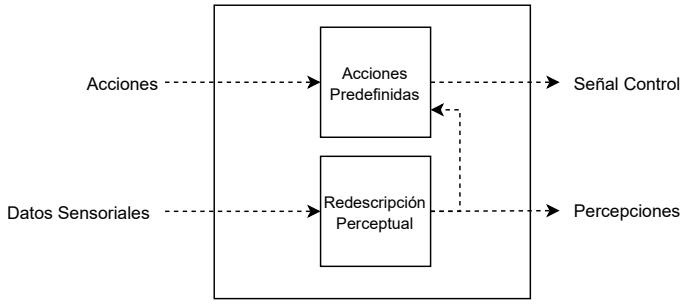


Figura A.4: Descomposición funcional del subsistema *redescripción*. Elaboración propia

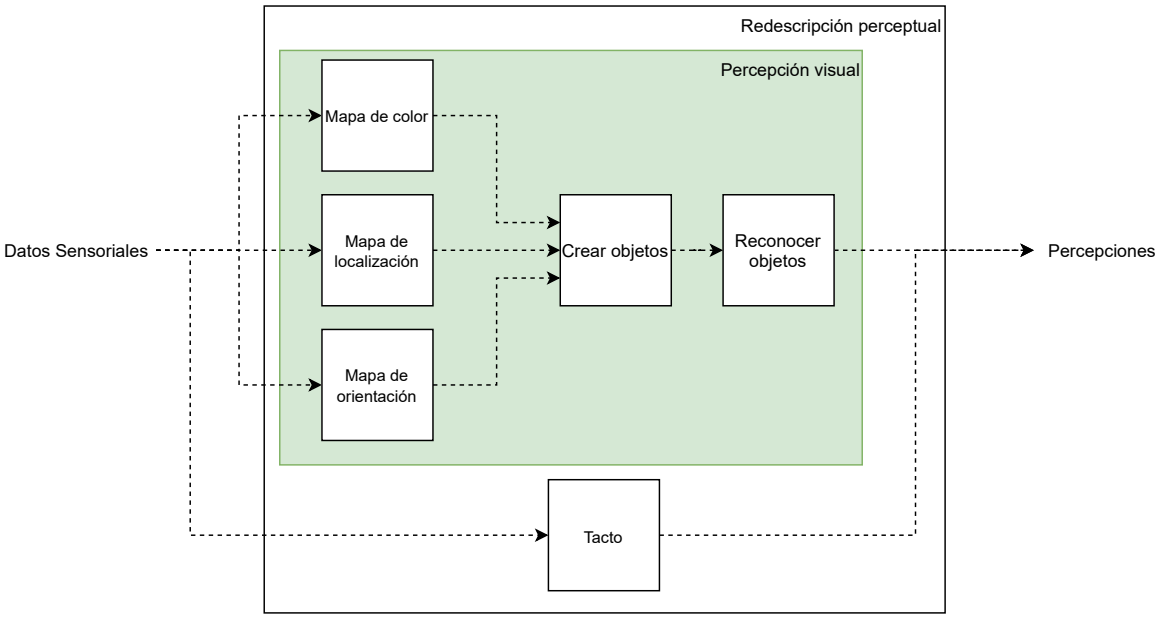


Figura A.5: Descomposición funcional del subsistema perceptual. Elaboración propia

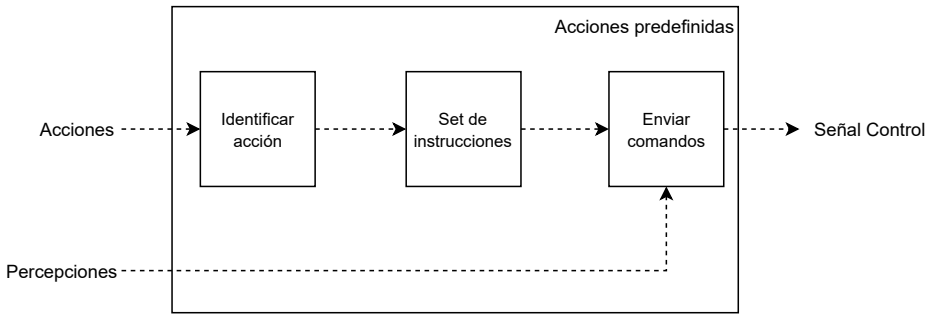


Figura A.6: Descomposición funcional del subsistema de acciones. Elaboración propia

A.4.2 Plataforma robótica

A.4.2.1 Fragmentos de concepto

Para cada subproblema presentado en la descomposición funcional de la figura A.2 se exploraron distintos fragmentos de solución, estos fueron clasificados en los árboles de las figuras A.7-A.10. Los fragmentos marcados en negrita fueron seleccionados para el estudio posterior. En la figura A.11 se muestra la matriz de combinación resultante de los fragmentos de concepto.

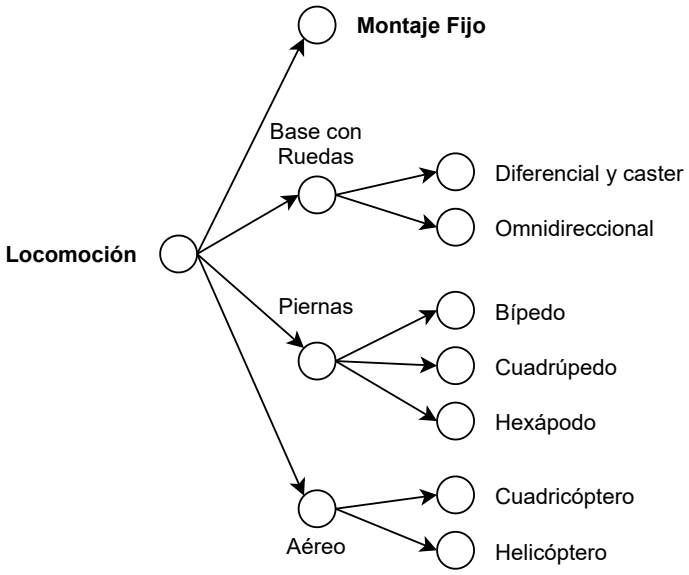


Figura A.7: Árbol de clasificación para el subproblema de *locomoción*. Elaboración propia

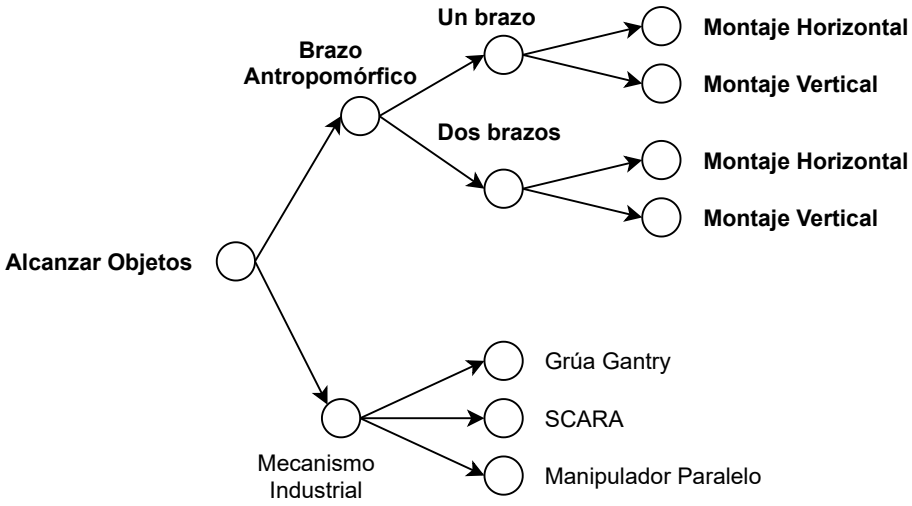


Figura A.8: Árbol de clasificación para el subproblema de *alcanzar objetos*. Elaboración propia

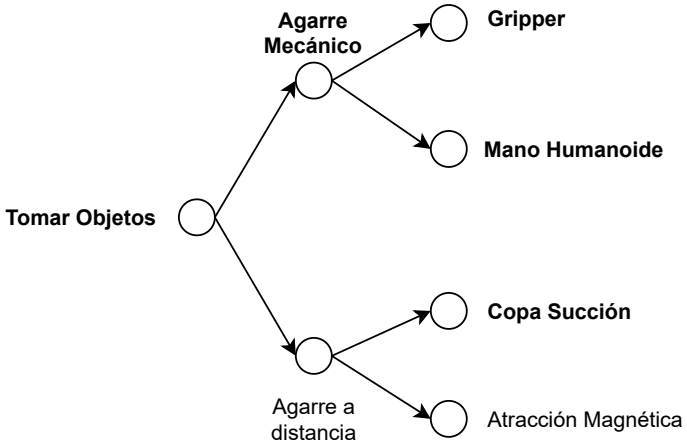


Figura A.9: Árbol de clasificación para el subproblema de *tomar objetos*. Elaboración propia

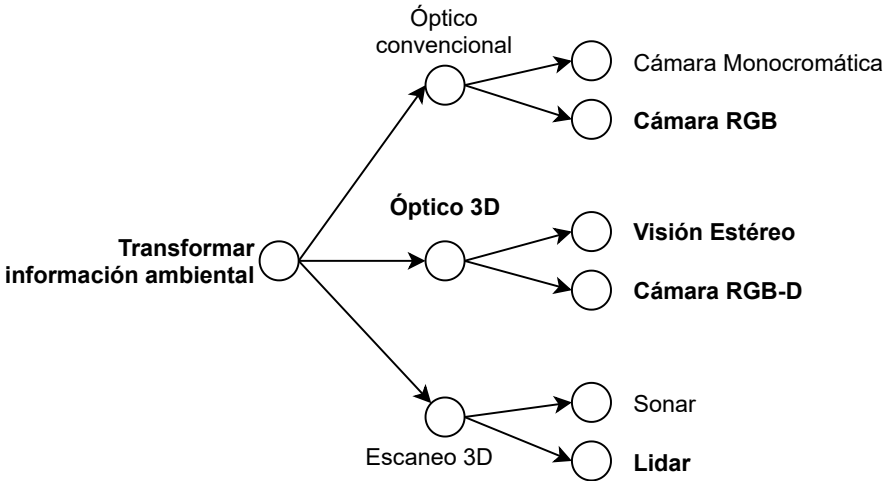


Figura A.10: Árbol de clasificación para el subproblema de *transformar información ambiental*. Elaboración propia

Alcanzar Objetos	Tomar Objetos	Localización Sensores	Transformar Información ambiental
Un Brazo (Mont. Horizontal)	Gripper	Sobremesa	Cámara RGB
Dos Brazos (Mont. Horizontal)	Mano Humanoide	Cabeza	Visión Estéreo
Un Brazo (Mont. Vertical)	Copa Succión	Torso	Cámara RGB-D
Dos Brazos (Mont. Vertical)			Lidar

Figura A.11: Matriz de combinación de fragmentos. Elaboración propia

A.4.2.2 Conceptos

A partir de combinaciones de fragmentos de la matriz de la figura A.11 se generaron los conceptos de las figuras A.12 a A.21. Las distintas opciones de manipuladores comerciales consideradas se resumen en la tabla A.15.

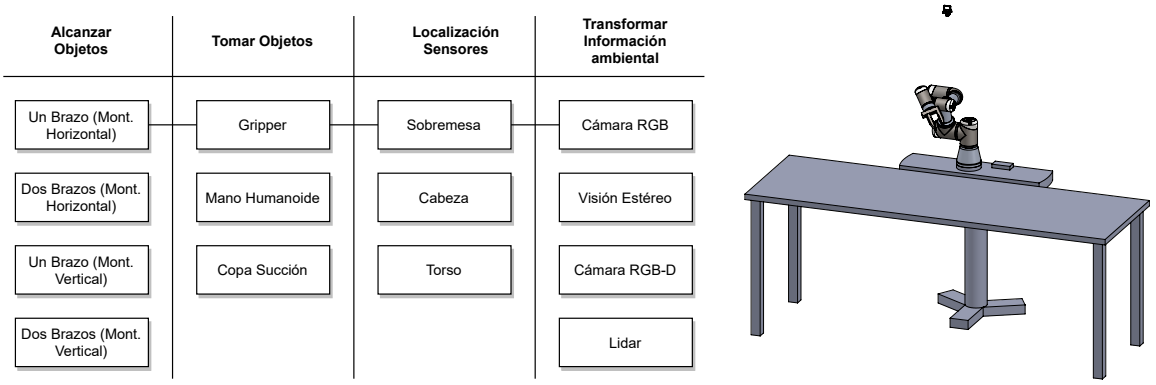


Figura A.12: Tabla de combinación y boceto del concepto 1. Elaboración propia: SolidWorks

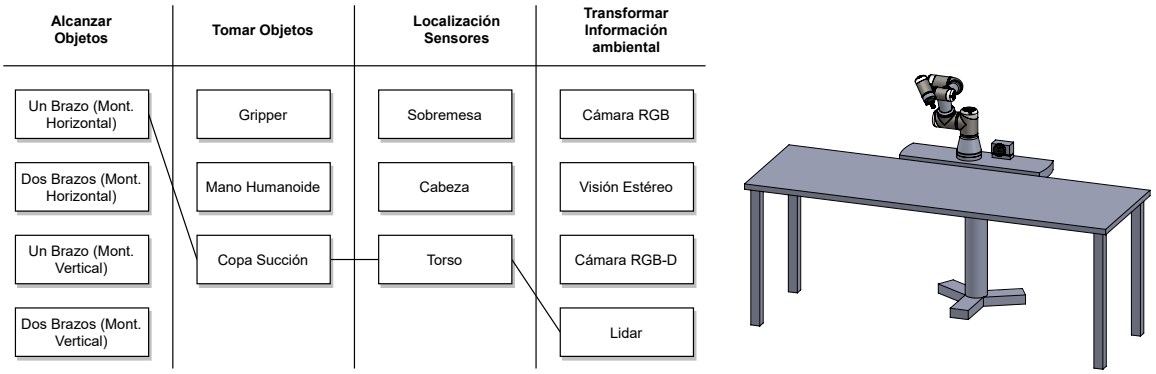


Figura A.13: Tabla de combinación y boceto del concepto 2. Elaboración propia: SolidWorks

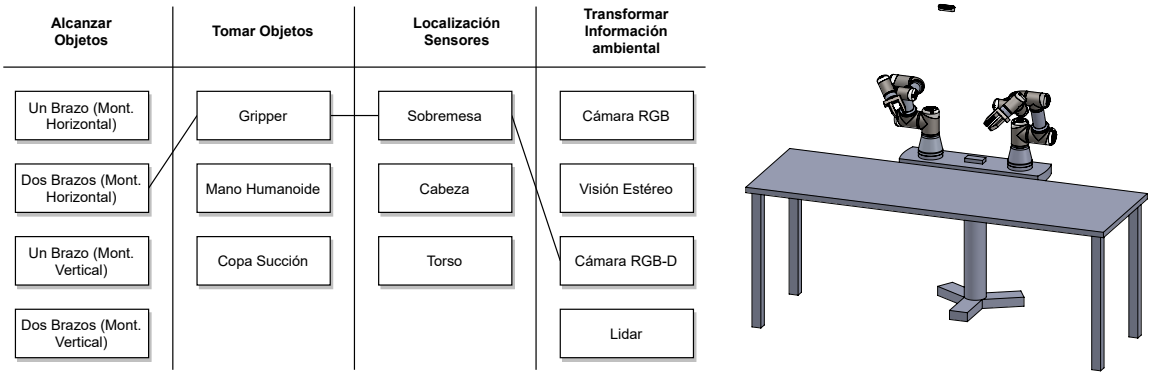


Figura A.14: Tabla de combinación y boceto del concepto 3. Elaboración propia: SolidWorks

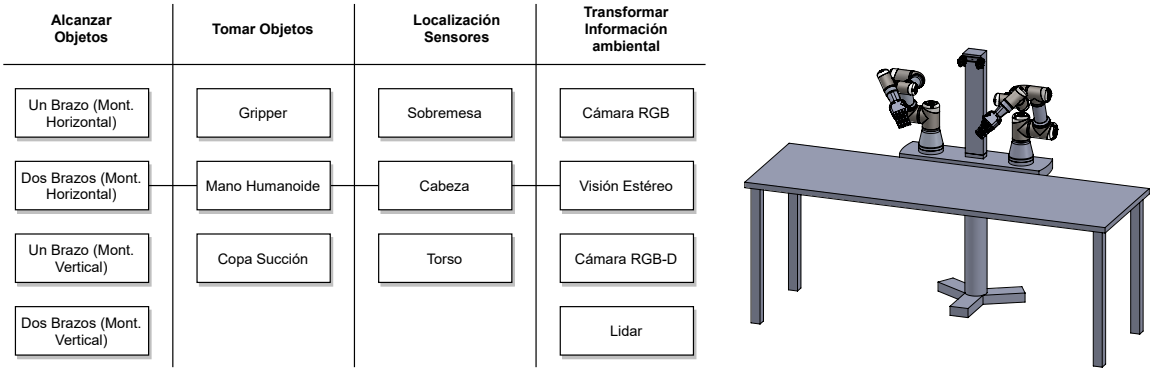


Figura A.15: Tabla de combinación y boceto del concepto 4. Elaboración propia: SolidWorks

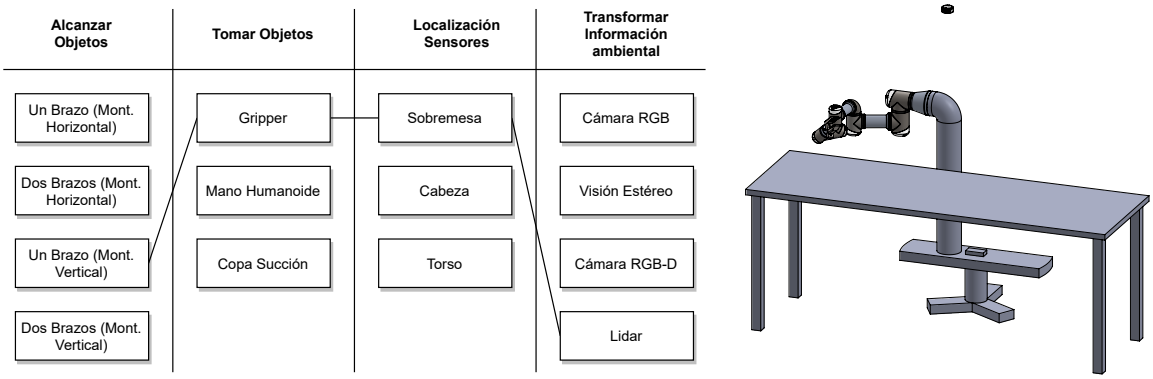


Figura A.16: Tabla de combinación y boceto del concepto 5. Elaboración propia: SolidWorks

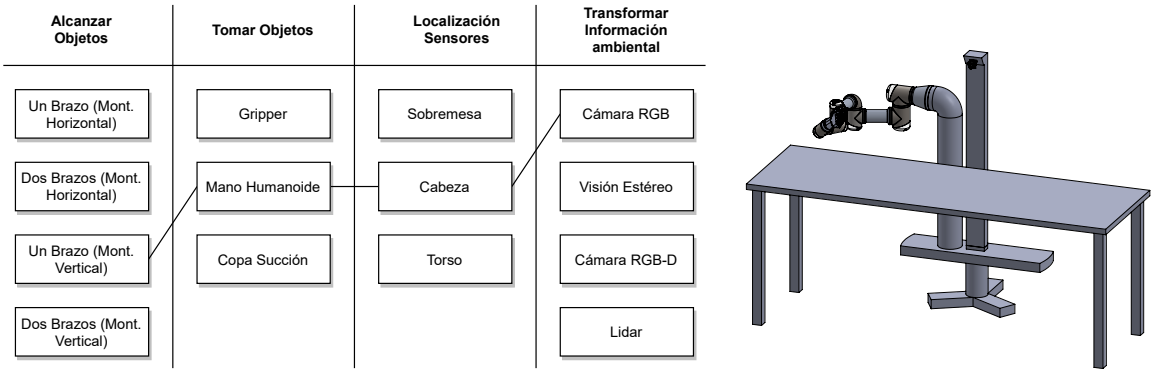


Figura A.17: Tabla de combinación y boceto del concepto 6. Elaboración propia: SolidWorks

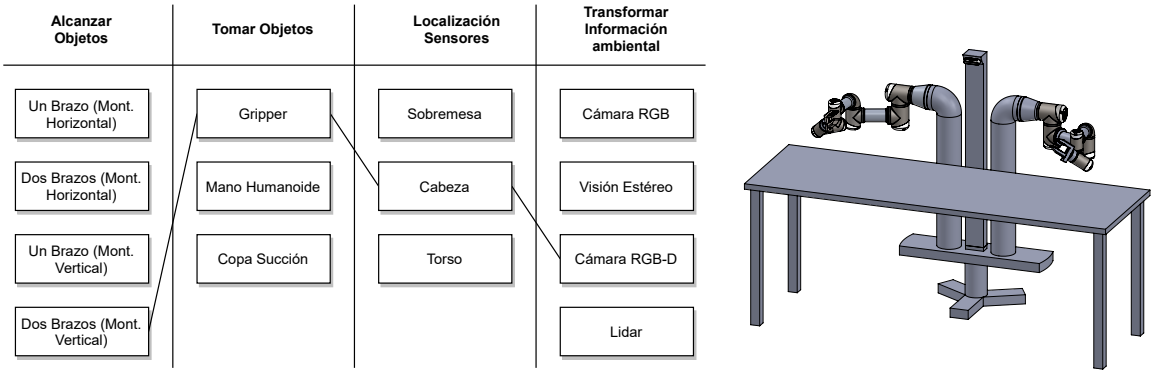


Figura A.18: Tabla de combinación y boceto del concepto 7. Elaboración propia: SolidWorks

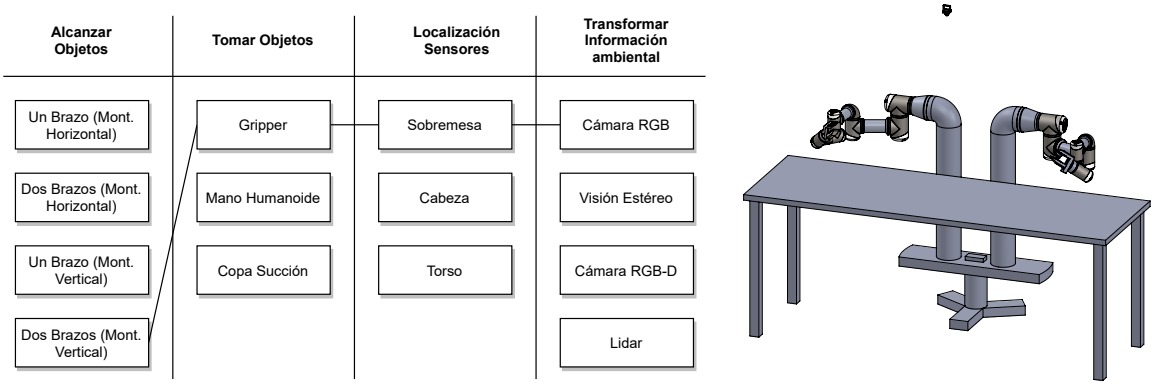


Figura A.19: Tabla de combinación y boceto del concepto 8. Elaboración propia: SolidWorks

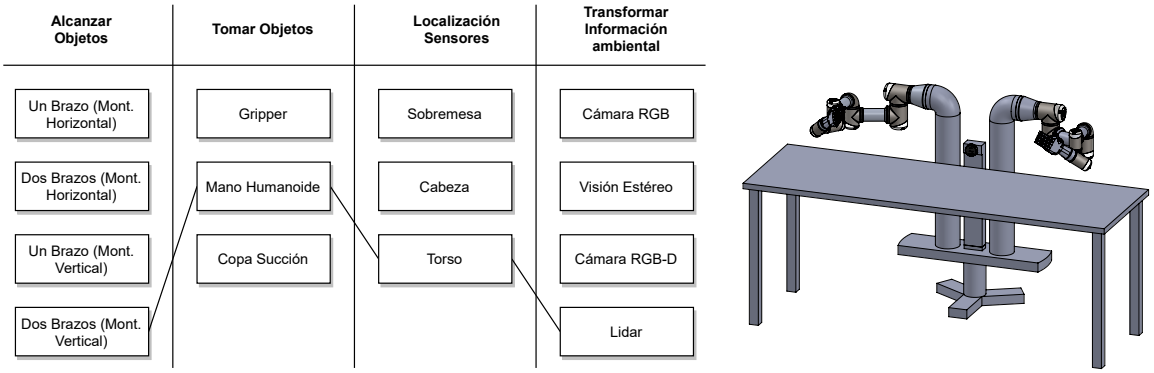


Figura A.20: Tabla de combinación y boceto del concepto 9. Elaboración propia: SolidWorks

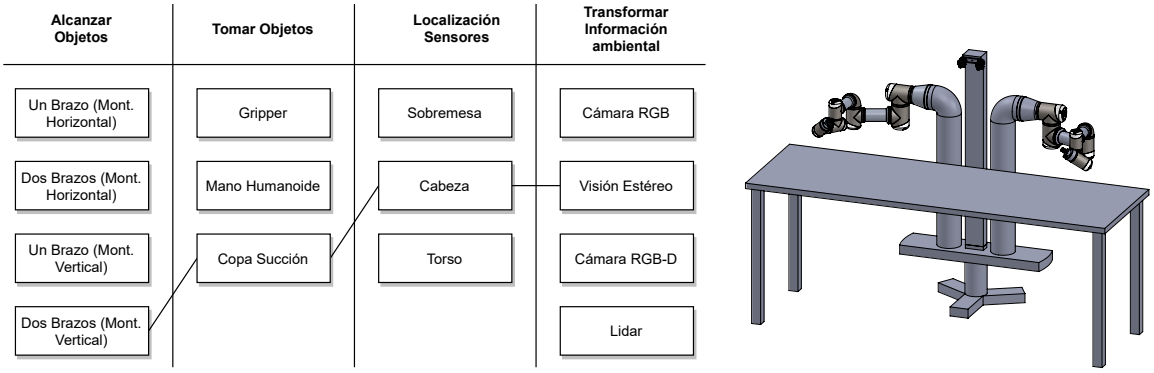


Figura A.21: Tabla de combinación y boceto del concepto 10. Elaboración propia: SolidWorks

Tabla A.15: Manipuladores comerciales disponibles.

No.	Nombre	Grados de Libertad	Alcance (mm)	Carga (g)	Costo ($)	Referencia
1	UR3e	6	500	3000	27000	[90]
2	Motoman Gp8	6	727	8000	20000	[91]
3	OpenManipulator-X	4	380	500	1000	[92]
4	OpenManipulator-P	6	645	3000	17000	[93]
5	Meca500	6	260	500	15000	[94]
6	WAM	7	1000	3000	150000	[95]
7	PincherX 150	5	450	50	950	[96]
8	ReactorX 150	5	450	100	1300	[97]
9	Crane-X7	7	500	500	8500	[98]
10	Revel	6	630	1200	6500	[99]
11	Han's Cute	7	506	300	6000	[100]
12	Manipulator-H	6	645	3000	19000	[101]
13	JACO	6	900	1800	35000	[102]
14	Cython Gamma	7	680	3000	10000	[103]
15	ST r12	5	500	500	9000	[104]
16	Thor	6	422.5	750	420	[77]
17	BNC3D MOVEO	5	525	*	400	[105]
18	Niryo One	6	440	300	2000	[106]
19	AR3	6	629	1900	2000	[107]
20	WidowX	6	650	250	2900	[108]

A.4.2.3 Selección de conceptos

Los conceptos presentados en la sección anterior para la morfología robótica fueron filtrados, como se muestra en la tabla A.16. A partir de esa tabla se obtienen los conceptos finalistas presentados en las figuras A.22-A.25, que fueron evaluados en la tabla A.17 para seleccionar el concepto 3+ para el desarrollo.

En el caso de los manipuladores, se realizó una preselección mostrada en la tabla A.18, debido a su cumplimiento de las especificaciones. Posteriormente, en la tabla A.19 se muestra la evaluación que permitió seleccionar el manipulador Thor.

Tabla A.16: Filtrado de conceptos para la plataforma robótica.

No. Concepto	Morfología Humanoide	Capacidad de Manipulación	Obtención de datos	Simplicidad Control	Simplicidad Procesamiento de Datos	Costo	Calificación Neta	Posición	¿Continuar?
1	-1	-1	0	1	0	1	0	2	Combinar
2	-1	-1	0	1	-1	0	-2	4	No
3	-1	1	1	1	0	0	2	1	Sí y Mejorar
4	1	1	-1	-1	-1	-1	-2	4	No
5	-1	-1	1	0	0	0	-1	3	No
6	0	0	0	-1	-1	0	-2	4	No
7	1	0	1	0	-1	-1	0	2	Combinar
8	**0**	**0**	**0**	**0**	**0**	**0**	0	2	Combinar
9	1	0	0	-1	-1	-1	-2	4	No
10	0	0	1	1	-1	-1	0	2	Combinar

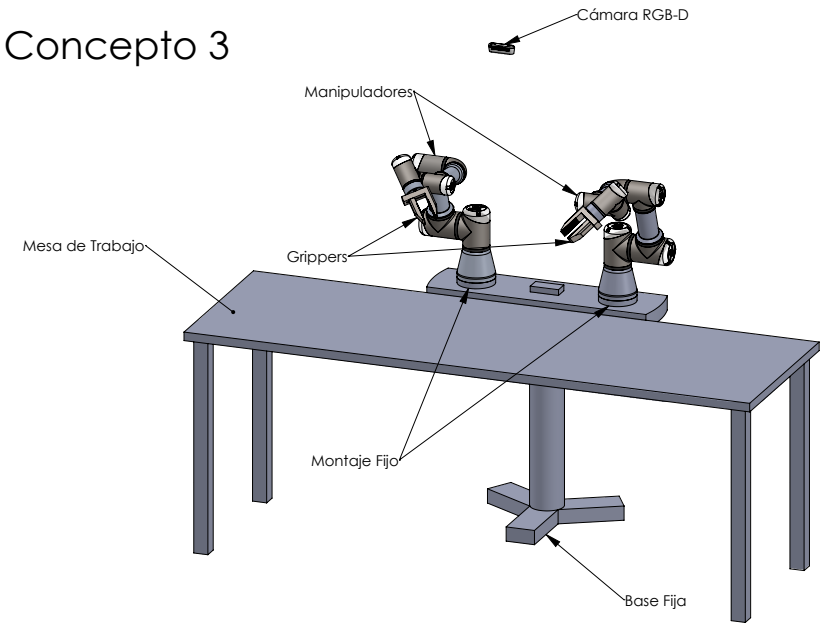


Figura A.22: Concepto 3. Elaboración propia: SolidWorks

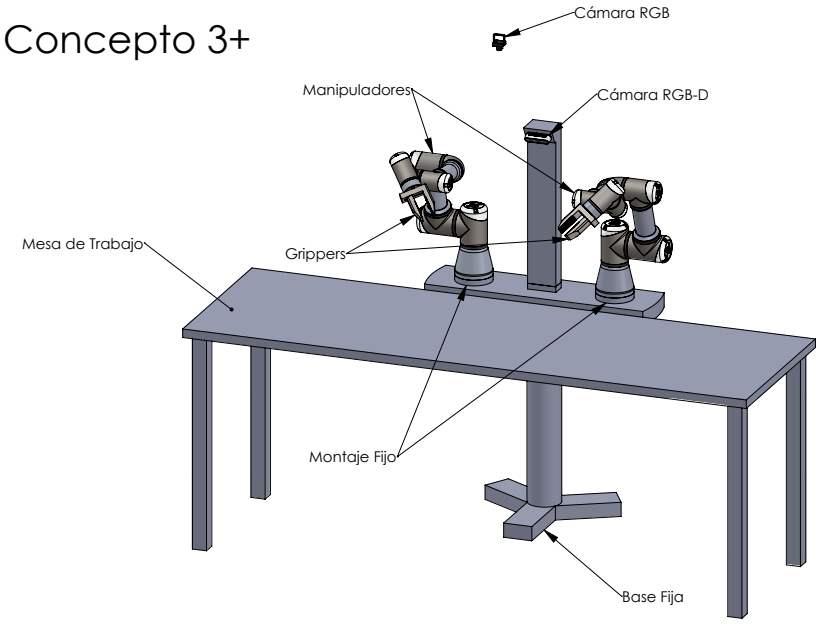


Figura A.23: Concepto 3 mejorado. Elaboración propia: SolidWorks

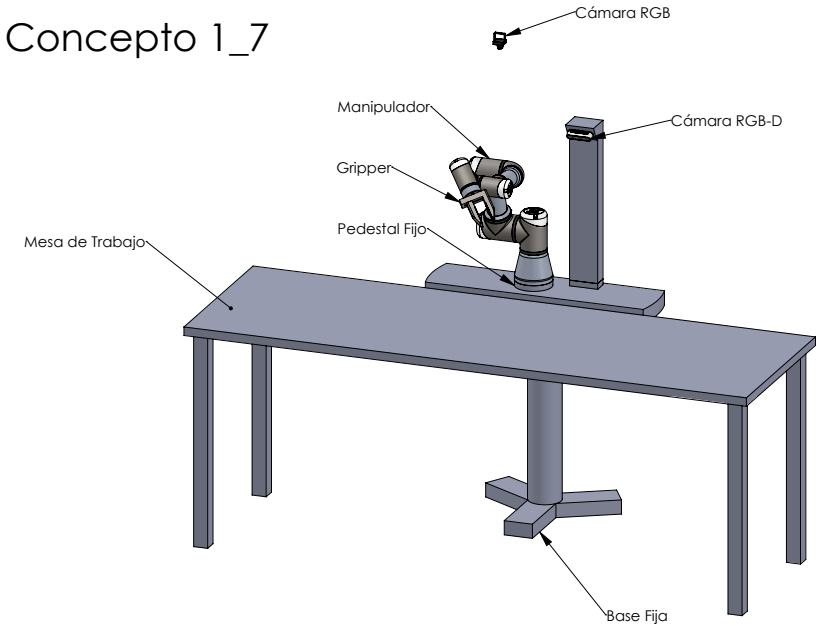


Figura A.24: Combinación de los conceptos 1 y 7. Elaboración propia: SolidWorks

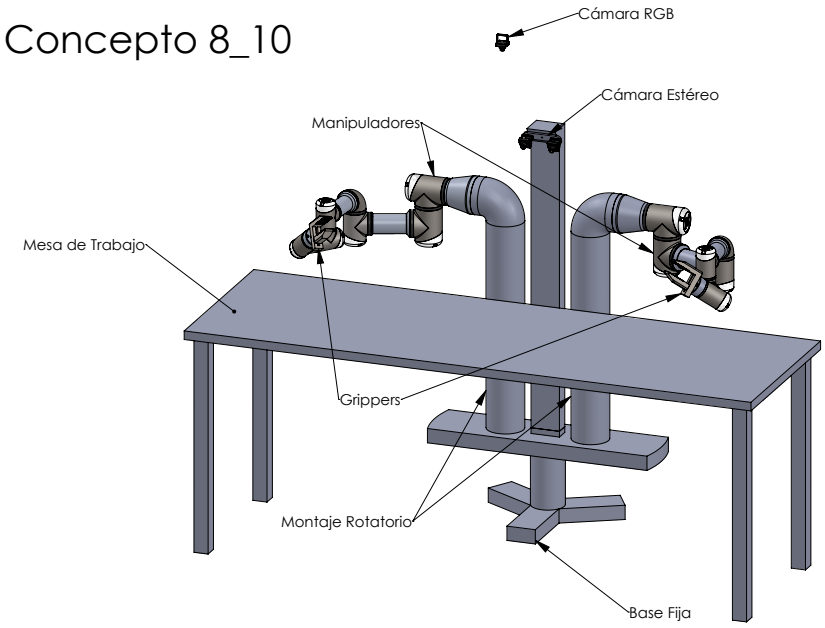


Figura A.25: Combinación de los conceptos 8 y 10. Elaboración propia: SolidWorks

Tabla A.17: Matriz de evaluación de conceptos para la plataforma robótica.

		Concepto 1_7		**Concepto 8_10**		**Concepto 3**		**Concepto 3+**	
Criterio de Selección	**Peso (%)**	**Calif.**	**Pond.**	**Calif.**	**Pond.**	**Calif.**	**Pond.**	**Calif.**	**Pond.**
Morfología Humanoide	15	1	0,15	5	0,75	2	0,3	**3**	0,45
Capacidad de Manipulación	25	2	0,5	**3**	0,75	3	0,75	3	0,75
Obtención de datos	25	4	1	**3**	0,75	2	0,5	4	1
Simplicidad Control	10	4	0,4	2	0,2	**3**	0,3	3	0,3
Simplicidad Procesamiento de Datos	10	3	0,3	2	0,2	4	0,4	**3**	0,3
Costo	15	4	0,6	1	0,15	**3**	0,45	2	0,3
	Total	2,95		2,8		2,7		3,1	
	Lugar	2		3		4		1	
	¿Continuar?	No		No		No		Desarrollar	

Tabla A.18: Manipuladores preseleccionados para la evaluación.

No.	**Nombre**	**Grados Libertad**	**Reach (mm)**	**Payload (g)**	**Actuación/ Sensorización**	**Paquete ROS**	**Calif. Open Hardware**	**Costo ($)**
1	Thor	6	422.5	750	Motores a pasos Finales de Carrera	No	5	420
2	Niryo One	6	440	300	Servomotores Finales de carrera Encoders Absolutos	Sí	5	2000
3	WidowX 6DoF	6	650	250	Servomotores Encoders Absolutos	Sí	2	2895
4	AR3	6	629	1900	Motores a pasos Finales de carrera Encoders Incrementales	No	3	2000

Tabla A.19: Tabla de evaluación de los manipuladores.

		Thor		**Niryo One**		**WidowX 6DoF**		**AR3**	
Criterios de Selección	**Peso (%)**	**Calif.**	**Pond.**	**Calif.**	**Pond.**	**Calif.**	**Pond.**	**Calif.**	**Pond.**
Apertura hardware y software	25	5	1,25	4	1	2	0,5	**3**	0,75
Capacidad de carga	10	4	0,4	**3**	0,3	2	0,2	5	0,5
Compatibilidad ROS	15	2	0,3	**3**	0,45	5	0,75	2	0,3
Costo	20	5	1	**3**	0,6	1	0,2	3	0,6
Destreza y Precisión	10	2	0,2	**3**	0,3	4	0,4	4	0,4
Espacio de trabajo	10	2	0,2	**3**	0,3	5	0,5	4	0,4
Sensorización	10	1	0,1	2	0,2	5	0,5	**3**	0,3
	Total	3,45		3,15		3,05		3,25	
	Lugar	1		3		4		2	
	¿Seleccionar?	Sí		No		No		No	

A.4.3 Redescripción

A.4.3.1 Conceptos estudiados

Para el sistema de redescripción se consideraron varias opciones para el procesado de los datos visuales. Para el procesamiento de imágenes en 2D se estudiaron las herramientas mostradas en la tabla A.20. Para el caso de los datos visuales en 3D se estudiaron las herramientas mostradas en la tabla A.21.

Tabla A.20: Opciones de software consideradas para el procesado de datos visuales 2D.

Software de procesamiento	Referencia
OpenCV	[68]
Scikit-Image	[69]
CameraTransform	[70]

Tabla A.21: Opciones de software consideradas para el procesado de datos visuales 3D.

Software de procesamiento	Referencia
FindObject	[71]
Object Recognition Kitchen	[73]
Point Cloud Library	[72]

A.4.3.2 Selección de conceptos

La selección de la herramienta de *software* de procesado de datos visuales se basó en los resultados de las matrices A.22 y A.23.

Tabla A.22: Selección de software para procesado de datos 2D

		OpenCV		**Scikit Image**		**CameraTransform**	
Criterios de Selección	**Peso (%)**	**Calif.**	**Pond.**	**Calif.**	**Pond.**	**Calif.**	**Pond.**
Integración con ROS	10	4	0,4	**3**	0,3	2	0,2
Robustez del software	25	5	1,25	**3**	0,75	2	0,5
Documentación disponible	30	3	0,9	3	0,9	**2**	0,6
Facilidad de implementación	35	1	0,35	**3**	1,05	5	1,75
	Total	2,9		3		3,05	
	Lugar	3		2		1	
	¿Seleccionar?	No		Utilizar		Utilizar	

Tabla A.23: Selección de software para procesado de datos 3D

		FindObject		Point Cloud Library		Object Recognition Kitchen	
Criterios de Selección	**Peso (%)**	**Calif.**	**Pond.**	**Calif.**	**Pond.**	**Calif.**	**Pond.**
Integración con ROS	10	4	0,4	**3**	0,3	2	0,2
Robustez del software	20	3	0,6	**5**	1	**3**	0,6
Documentación disponible	30	**3**	0,9	4	1,2	**1**	0,3
Facilidad de implementación	35	4	1,4	**2**	0,7	**3**	1,05
	Total	3,3		3,2		2,15	
	Lugar	1		2		3	
	¿Seleccionar?	Considerar		Considerar		No	

A.5 Estudio económico

A continuación se presentan materiales adicionales al análisis realizado en el capítulo 7 del documento principal. En la tabla A.24 se sintetizan los costos de desarrollo del proyecto en términos monetarios. El costo del tiempo de desarrollo se estimó utilizando el esquema de pagos de las asistencias estudiantiles del Instituto Tecnológico de Costa Rica para el total de horas invertidas al proyecto.

También se presenta la estimación de la inversión inicial para la implementación física del proyecto. El costo del manipulador *Thor* se estimó a partir de la tabla de materiales disponible [77], los cuales fueron buscados en distribuidores locales para tener una estimación realista. El costo de los componentes mecánicos se muestra en la tabla A.25 y de los componentes electrónicos en A.26. La estimación completa del costo de la implementación física se muestra en la tabla A.27 (presentada también en el documento principal).

Tabla A.24: Costos de desarrollo del proyecto.

No.	**Detalle**	**Costo**
1	Depreciación equipos	63138
2	Licencias de software	61200
3	Desarrollo	832000
	Total	956338

Tabla A.25: Estimación de costos de los componentes mecánicos del manipulador Thor.

Componente	**Cantidad**	**Costo Unitario ($)**	**Costo total ($)**
Filamento 3D (kg)	4	40,95	163,80
Motor Nema 17 40mm	1	16,95	16,95
Gearmotor Nema 17 34mm	3	33,4	100,20
Motor Nema 17 34mm	3	28,95	86,85
GT2 Closed Belt 208mm	2	4,69	9,38
GT2 Open Belt (m)	1	10,95	10,95
GT2 Pulley 20t	3	2,95	8,85
GT2 Pulley 40	2	6,02	12,04
Bearing 16014zz	1	27	27,00
Bearing 625ZZ	11	11,425	125,68
Bearing MF84ZZ	2	13,04	26,08
Barra 4 mm (28 mm largo)	1	0	0,00
Barra 5 mm (276,5 mm largo)	1	17,25	17,25
Perno M3x6	6	0,1165	0,70
Perno M3x8	46	0,0985	4,53
Perno M3x10	3	0,101875	0,31
Perno M3x12	15	0,116	1,74
Perno M3x16	9	0,12275	1,10
Perno M3x18	4	0,153125	0,61
Perno M3x25	8	0,1785	1,43
Perno M3x28	8	0,89625	7,17
Perno M3x30	8	0,89625	7,17
Perno M3x40	13	0,185	2,41
Perno M3x46	8	0,198	1,58
Perno M5x18	1	0,186375	0,19
Tuerca M3	90	0,014625	1,32
Tuerca Seguridad M5	1	0,062	0,06
		Total	635,34

Tabla A.26: Estimación de costos de los componentes electrónicos del manipulador Thor.

Componente	**Cantidad**	**Costo Unitario ($)**	**Costo total**
Arduino Mega	1	18,95	18,95
Final de carrera	1	1,95	1,95
Ventilador 40x40	9	4,95	44,55
A4988 Stepper Driver	8	7,5	60
Capacitores	8	0,95	7,6
Resistencias	19	0,06	1,14
Leds	3	0,16	0,48
Diodo 1N4004	1	0,15	0,15
Fusible MF-R700	1	1	1
Pines macho	169	0,0275	4,6475
PCB Terminal block	2	0,85	1,7
Female Pins	128	0,0175	2,24
Sensor óptico	4	1,75	7
		Total	151,41

Tabla A.27: Estimación del costo de la implementación física de la plataforma robótica.

No.	**Componente**	**Cantidad**	**Costo Unitario ($)**	**Costo Total ($)**
1	Manipulador Thor	2	786,75	1573,49
2	Realsense D435	1	236,25	236,25
3	Raspberry Cam V2	1	45,95	45,95
4	Estructuras de montaje	1	150,00	150,00
5	Mano de obra	1	650,00	650,00
6	Imprevistos	1	531,14	531,14
			Total	3186,83

A.6 Especificaciones finales

En la tabla A.28 se presentan las especificaciones de la herramienta al finalizar esta etapa del desarrollo. Los valores se basan en los experimentos realizados y en el análisis de la herramienta.

Tabla A.28: Especificaciones finales de la herramienta.

No.	Métrica	Unidad	Imp.	Valor Final
1	Grados de libertad por brazo	Cantidad	5	6
2	Número de brazos	Cantidad	3	2
3	Alcance manipulador	mm	4	467.5
4	Sensores equipados	Lista	5	-Sistema Visión -Finales de carrera
5	Dimensiones máximas objeto a manipular	mm	4	50
6	Peso máximo objeto a manipular	g	3	500
7	Velocidad manipulador (P1)	m/s	4	*
8	Costo plataforma robótica	$	3	3186
9	Clasificación Movilidad	Ver Clasificación.	2	1
10	Precisión en la pose del manipulador (P1)	mm	5	0.46
11	Campo de visión	mm	4	1508 x 650
12	Resolución sistema visión	mm	4	1.5
13	Aciertos en la toma de objetos (P2)	% aciertos/intentos	5	99.69
14	Error en la prueba de percepción (P3)	mm	5	1.55
15	Proporción de acciones recompensadas (P4)	% recompensas/acciones	5	80
16	Estructuras cognitivas integradas	Lista	5	-LTM
17	Tiempo procesamiento de percepciones (P3)	ms	3	*
18	Confiabilidad de la comunicación	% datos recibidos/datos enviados	3	*
19	Módulos ejecutados de forma concurrente	Lista	3	-Percepción -Acción -Aprendizaje y Memoria
20	Opinión del cliente sobre manual de uso	Subjetivo	5	*
21	Aspectos detallados documentación de diseño	Lista	4	-Necesidades -Especificaciones Objetivo -Conceptos -Análisis Económico -Especificaciones Finales
22	Puntuación de apertura componentes Hardware	Ver Puntuación.	4	5
23	Puntuación de apertura módulos de Software	Puntuación OSS Watch.	4	*