UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERÍA
CARRERA DE INGENIERÍA EN TELECOMUNICACIONES

Implementación de un robot de carga autónomo para el transporte de bloques de construcción frescos en la fábrica "BLOQUERA NAULA" aplicando tecnologías IOT Y visión artificial

Trabajo de Titulación para optar al título de
INGENIERO EN TELECOMUNICACIONES

Autor:
Naula Sislema Darwin Alexander

Tutor:
Phd. Leonardo Fabian Rentería Bustamante

Riobamba, Ecuador. ***2026***

DECLARATORIA DE AUTORÍA

Yo, Darwin Alexander Naula Sislema, con cédula de ciudadanía 0606310514, autor del trabajo de investigación titulado: "Evaluación del uso de un robot de carga autónomo para el transporte de bloques de construcción frescos en la fábrica "Bloquera Naula" aplicando tecnologías IoT y visión artificial" , certifico que la producción, ideas, opiniones, criterios, contenidos y conclusiones expuestas son de mí exclusiva responsabilidad.

Asimismo, cedo a la Universidad Nacional de Chimborazo, en forma no exclusiva, los derechos para su uso, comunicación pública, distribución, divulgación y/o reproducción total o parcial, por medio físico o digital; en esta cesión se entiende que el cesionario no podrá obtener beneficios económicos. La posible reclamación de terceros respecto de los derechos de autor (a) de la obra referida, será de mi entera responsabilidad; librando a la Universidad Nacional de Chimborazo de posibles obligaciones.

En Riobamba, el 5 de enero de 2026

Darwin Alexander Naula Sislema

C.I: 0606310514

DICTAMEN FAVORABLE DEL PROFESOR TUTOR

Quien suscribe, Leonardo Fabián Rentería Bustamante catedrático adscrito a la Facultad de Ingeniería, por medio del presente documento certifico haber asesorado y revisado el desarrollo del trabajo de investigación titulado: "Implementación de un robot de carga autónomo para el transporte de bloques de construcción frescos en la fábrica "BLOQUERA NAULA" aplicando tecnologías IOT Y visión artificial" , bajo la autoría de Darwin Alexander Naula Sislema ; por lo que se autoriza ejecutar los trámites legales para su sustentación.

Es todo cuanto informar en honor a la verdad; en Riobamba, a los 16 del mes de diciembre de 2025

PhD. Leonardo Fabián Rentería Bustamante

C.I:1104064132

CERTIFICADO DE LOS MIEMBROS DEL TRIBUNAL

Quienes suscribimos, catedráticos designados Miembros del Tribunal de Grado para la evaluación del trabajo de investigación IMPLEMENTACIÓN DE UN ROBOT DE CARGA AUTÓNOMO PARA EL TRANSPORTE DE BLOQUES DE CONSTRUCCIÓN FRESCOS EN LA FABRICA "BLOQUERA NAULA" APLICANDO TECNOLOGÍAS IOT Y VISIÓN ARTIFICIAL, presentado por DARWIN ALEXANDEE NAULA SISLEMA con cédula de identidad número 0606310514, bajo la tutoría de PhD Leonardo Fabián Rentería Bustamante; certificamos que recomendamos la APROBACIÓN de este con fines de titulación. Previamente se ha evaluado el trabajo de investigación y escuchada la sustentación por parte de su autor; no teniendo más nada que observar.

De conformidad a la normativa aplicable firmamos, en Riobamba 5 de enero de 2026

Carlos Peñafiel, PhD.
PRESIDENTE DEL TRIBUNAL DE GRADO

José Jinez, Mgs.
MIEMBRO DEL TRIBUNAL DE GRADO

Klever Torres, Dr.
MIEMBRO DEL TRIBUNAL DE GRADO

C E R T I F I C A C I Ó N

Que, **NAULA SISLEMA DARWIN ALEXANDER** con CC: **060631051-4**, estudiante de la Carrera **TELECOMUNICACIONES**, Facultad de **INGENIERIA**; ha trabajado bajo mi tutoría el trabajo de investigación titulado " **Implementación de un robot de carga autónomo para el transporte de bloques de construcción frescos en la fábrica "BLOQUERA NAULA" aplicando tecnologías IOT Y visión artificial**", cumple con el 2 % de similitud y 4% de Inteligencia Artificial, de acuerdo al reporte del sistema Anti plagio **Compilatio**, porcentaje aceptado de acuerdo a la reglamentación institucional, por consiguiente autorizo continuar con el proceso.

Riobamba, 18 de diciembre de 2025

Firmado electrónicamente por:
LEONARDO FABIAN
RENTERIA BUSTAMANTE
Validar únicamente con FirmaEC

Phd. Leonardo Rentería
TUTOR(A)

DEDICATORIA

El presente trabajo de investigación está dedicado a mis abuelitos, quienes desde el primer día de mi vida me vieron crecer, quienes con su amor, esfuerzo y consejos han sido el principal pilar de mi vida, este logro es a la vez el de ellos, a mis padres, por su apoyo y acompañamiento en esta travesía de la formación profesional, cada uno a su manera, a mis hermanos, aunque ya no está presente con nosotros, siempre lo estará en mi corazón y en cada hito alcanzado y a mi pequeña hermana, quien ha sido fuente de alegrías y energía que me impulsa a superar cada día en busca de un mañana mejor.

AGRADECIMIENTO

Mis agradecimientos quedarán plasmados hacia mi tutor PhD. Leonardo Rentería, gracias a su paciencia, dedicación y confianza depositada en este trabajo de titulación. Su guía académica y humana han sido un ejemplo a seguir y primordial para la culminación de este proyecto.

A la carrera y a sus docentes que desde el primer día me demostraron su profesionalismo y calidad humana en su trabajo, me enseñaron comó analizar y solucionar problemas reales con las ciencias exactas, quien me enseñó a encender un led y dejó la iniciativa a evolucionar a sistemas más complejos, elevándonos a temas más dedicados a la carrera, me indicaron cómo mínimos detalles como una distancia de un cuarto de longitud de onda, un aumento de temperatura o una máscara de red, marca la diferencia entre el correcto funcionamiento o falla del sistema de comunicación. Mis docentes me han inspirado tanto dentro como fuera del aula, inculcándome la curiosidad de entender el funcionamiento de las cosas, la afirmación de que no es necesario seguir otras carreras para aprender algo nuevo y que uno debe ser el primero en confiar en sí mismo.

A mis compañeros que formamos lazos de hermandad por el camino, Washo y Kevin, quienes han estado presentes en momentos clave, sus palabras y apoyo me permitieron intentarlo una vez más y seguir, su presencia y ayuda en este último proceso han sido invaluables.

A mis queridas amigas, Cristina, quien me dio el empuje necesario para seguir mis grandes aspiraciones, También a quienes no se encuentran presencialmente cerca, me enseñan que una amistad puede resistir tiempo y distancia Dirzze y Allison, no me arrepiento de seguir este camino por haber conocido tan maravillosas personas.

A todos ustedes, gracias por formar parte de este recorrido y por contribuir, de distintas maneras, a que este trabajo sea una realidad.

ÍNDICE GENERAL

ÍNDICE DE TABLAS.

ÍNDICE DE FIGURAS.

RESUMEN

En el presente trabajo se muestra el diseño, construcción e implementación de un robot autónomo móvil empleando visión artificial para el transporte de elementos prefabricados de hormigón de la empresa “Bloquera Naula”. El diseño del sistema es una arquitectura descentralizada, con un nodo central de procesamiento de algoritmos y un nodo a bordo encargado en la navegación, control y registro de datos, el prototipo consta de sistema de locomoción eléctrica controlado mediante el framework ROS 2 y algoritmos de navegación mediante visión artificial, se emplea una cámara RGB-D como sensor principal para la obtención de datos, percepción, localización, generación de mapas y estimación de pose, además de un sensor IMU para registrar aceleraciones en el eje Z, en las pruebas se programó el prototipo para seguir trayectorias predefinidas en el área de pruebas con dos perfiles de navegación donde se modificaron parámetros como aceleraciones y velocidades. A partir de los datos de navegación se obtuvieron tiempos de recorrido, distancia restante a la meta, velocidad media y niveles de vibración sobre la carga, evidenciándose que el perfil Equilibrado logra mayor porcentaje de trayectorias exitosas con errores de posicionamiento cercanos a 0,19 m y menores picos de aceleración, demostrando la viabilidad del prototipo para mejorar la ergonomía y la eficiencia en el transporte interno.

Palabras claves: AMR, Visión artificial, ROS 2, Navegación autónoma, Cámara RGB-D.

ABSTRACT

This work presents the design, construction and implementation of an autonomous mobile robot using computer vision for the transport of precast concrete elements of the company "Bloquera Naula". The system is designed with a decentralized architecture, with a central node for algorithm processing and an onboard node in charge of navigation, control and data logging. The prototype consists of an electric locomotion system controlled by the ROS 2 framework and navigation algorithms based on computer vision. An RGB-D camera is used as the main sensor for data acquisition, perception, localization, map generation and pose estimation, together with an IMU sensor to record accelerations on the Z axis. In the tests, the prototype was programmed to follow predefined trajectories in the test area with two navigation profiles, in which parameters such as accelerations and velocities were modified. From the navigation data, travel times, remaining distance to the goal, average speed and vibration levels on the load were obtained, showing that the "Equilibrado" profile achieves a higher percentage of successful trajectories with positioning errors close to 0.19 m and lower acceleration peaks, demonstrating the feasibility of the prototype to improve ergonomics and efficiency in the internal transport process.

Keywords: AMR, vision artificial, ROS 2, Autonomous navigation, RGB-D camera.

Doris Alexandra
Chuquimarca Once

Time Stamping
Security Data

Reviewed by:
Doris Chuquimarca Once, M.A. in TESOL
ESL PROFESSOR, UNACH
I.C. 060449038-3

CAPÍTULO I. INTRODUCCIÓN.

El sector de la construcción es un pilar fundamental para la economía ecuatoriana que representa aproximadamente el 6.19% de producto interno bruto nacional de 2023 [1], dentro de este sistema el 92% está conformado por micro, pequeñas y medianas empresas (MIPYMES) enfocadas a la fabricación de materiales y suministros de construcción [1]. Sin embargo, dicho sector enfrenta una falta de tecnificación, limitando de esta manera su competitividad frente a empresas ya consolidadas en el mercado.

Dentro de las operaciones realizadas en la industria, una de las menos atendidas tecnológicamente es el transporte interno de bloques recién elaborados, Actualmente esta fase de producción en las empresas MIPYMES se realiza de manera manual utilizando una "carretilla tipo uña". Este método tiene distintas formas de mejorar, ya que cuenta con consecuencias directas en la producción y terminado final, además de que la fatiga física del personal que disminuye el ritmo de trabajo, genere obstrucción en la producción, cuenta con la probabilidad de que el error humano esté presente y resulte en daños en el producto causados por movimientos bruscos y manejo inapropiado, como podría ser micro fisuras, daños estéticos y hasta descarte de la producción [2], además de que se expone a los trabajadores a lesiones ergonómicas [3].
Si bien existen soluciones para el problema de manejo de materiales con la automatización industrial, estas están diseñadas para grandes volúmenes de producción y una enorme inversión económica [4] que no es factible para las pequeñas industrias ecuatorianas, de tal manera que se forma una brecha tecnológica que perpetúa la dependencia de métodos manuales y limitando la producción.

Para abordar esta problemática, la presente tesis propone el diseño, desarrollo y validación de un robot de carga autónomo de bajo costo para el transporte de tableros con bloques recientemente fabricados de construcción. Esta solución se fundamenta en la integración de hardware accesible y software de código abierto, donde se hace uso de sistemas como Raspberry PI 5, Ubuntu en sus versiones a largo plazo como lo es Noble y ROS 2 [5]. Adicional a ello los sistemas de visión artificial compatibles con el hardware Intel RealSense, el cual permitirá descubrir el entorno para posteriormente navegar en el de manera segura, desde distintos puntos evitando obstáculos imprevistos con un manejo controlado y cuidadoso.

En este trabajo se propone el desarrollar una alternativa de automatización viable y accesible que permita a las industrias pequeñas del sector añadir una mejora productiva a sus sistemas, aumentando su eficiencia, reduciendo porcentajes de perdidas, aumentando la calidad del producto y mejorando la calidad de trabajo de sus empleados. De esta manera contribuyendo e innovando en un segmento importante de la economía del país.

1.1 Antecedentes

Los avances en robótica independiente han logrado avances significativos en áreas como la producción, manejo autónomo y entornos controlados. Trabajos de investigación como "Construcción de un prototipo de sistema robótico para el despacho de productos farmacéuticos de bajo peso utilizando visión artificial y comunicación inalámbrica" [6] fortalecen la idea de fiabilidad de prototipos que emplean la visión artificial para actividades de interpretación y selección, sin embargo, las capacidades que se emplean en el área de construcción, en la investigación anterior se ha diseñado un prototipo móvil en un entorno ideal y con una carga máxima de 0,2 kilogramos, siendo una capacidad inferior a las magnitudes empleadas en el área de elementos prefabricados.

Varios proyectos contemporáneos se han enfocado en el desarrollo de algoritmos para la evasión de obstáculos y la optimización de una navegación autónoma [7], [8], [9] de manera general, sin integrarla a algún fin determinado o aplicable directamente a la innovación industrial, con ello limitando la modernización en industrias MIPYMES y permaneciendo una brecha en las industrias por falta de opciones de automatización viables por costos comerciales elevados.

Debido a la aparición de nuevas tecnologías de hardware y software accesibles surge la oportunidad de eliminar estas limitaciones indicadas anteriormente, como nuevas versiones optimizadas del Framework ROS [5], dedicado a áreas como la automatización robótica, accesibilidad a sensores anteriormente de alto costo como lo son las cámaras RGB-D de Intel y librerías optimizadas como REALSENSE. Son las bases esenciales para el desarrollo de este tipo de proyectos, como hardware de fácil acceso, optimización de los algoritmos para la navegación autónoma como lo son el mapeo y localización simultáneo (SLAM) y detección de objetos del entorno.

En consecuencia, este proyecto se fundamenta principalmente en avances previos en visión artificial y robótica móvil para la resolución de una problemática en las desatendidas microindustrias, creando una solución tecnológica, viable y accesible para el movimiento de cargas en un entorno normalizado.

1.2 Planteamiento del problema

En el sector de la construcción ecuatoriana, quien es aportante constante al PIB nacional está compuesto principalmente por empresas MIPYMES [1] (micro, pequeñas y medianas empresas) tiene un limitado acceso a la tecnificación en sus etapas de producción. Una de sus fases desatendidas de forma tecnológica es el transporte interno de productos recientemente elaborados y que no cuentan aún con la resistencia final, en empresas pequeñas este proceso se lo hace de forma manual, con una pseudo "carretilla tipo uña", de esta forma se añaden movimientos que afectan al producto, como puede ser: vibraciones, aceleraciones y maniobras bruscas [2], perjudicando su geometría e integridad del producto,

adicionalmente produce fatiga al operario, complicaciones en su salud y obstrucción de la producción.

Esta problemática es relevante a la hora de la fabricación de bloques para la construcción, existen estándares de calidad de la normativa técnica ecuatoriana NTE que presenta rangos de defectos en el momento de la entrega del material, específicamente la norma NTE INEN 3066, que indica restricciones como elementos despostillados y fisuras menor al 5% [10], por ello la etapa de manipulación y transporte es clave para minimizar daños y cumplir con el estándar de calidad.

Pequeñas empresas como “Bloquera Naula”, existe una demanda constante del producto y el transporte interno de forma manual se ha determinado como una etapa clave por ser propenso a daños, deformaciones y un trabajo extenuante para el operador, comprometiendo así la calidad y cadena de producción. Existen alternativas que ofrecen a la industria destinada a enormes volúmenes y principalmente con inversiones elevadas, de esta manera manteniendo una distancia tecnológica de las empresas MIPYNES y manteniendo la dependencia de este método manual. Al no atenderse esta problemática, se mantiene latente el riesgo de aumentar el porcentaje de bloques defectuosos y el descarte de estos, además de reducir la competitividad frente a compañías que cuentan con los procesos automatizados.

Por ende, se tiene la necesidad de desarrollar una solución que optimice la etapa de transporte, resolviendo interrogantes sobre las tecnologías, diseño y validación de un sistema de bajo costo en comparación con las soluciones disponibles en el mercado. El sistema debe asegurar el transporte seguro de elementos recién fabricados, minimizando daños ocasionados por vibraciones, cumpliendo con la normativa ecuatoriana vigente y ofreciendo una opción viable para el mercado local, con potencial de expansión futura.

1.3 Justificación

La presente propuesta responde a una necesidad real y latente en el sector productivo de la construcción, el mejorar una etapa donde actualmente predominan métodos manuales que pueden generar un daño al producto final, limitaciones en la producción, fatiga y problemas de salud en el personal [3], la propuesta es la evaluación de un robot de carga autónomo como una posible alternativa técnica y accesible para la disminución de estas problemáticas y aumentar la competitividad en la industria.

Desde una perspectiva técnica, las tecnologías como visión artificial, internet de la cosas y control autónomo nos permite abordar las causas principales del problema: detección y evasión de objetos, planificación de trayectorias [11] y cuidado en el manejo de cargas. El uso de nuevas tecnologías como plataformas embebidas para el procesamiento a bordo (Edge computing), nos dotan de más capacidad y nuevas herramientas para la implementación de este prototipo sea funcional en condiciones reales.

En la visión económica, la propuesta tiene la estrategia de ser una solución de bajo costo frente a las soluciones comerciales, con el análisis preliminar de los componentes y presupuestos del proyecto se observa una inversión moderada, accesible a las empresas MIPYMES que forman el mayor porcentaje de actividades relacionadas a "Materiales y Suministros para la Construcción" [1] . Ya que es favorable la recuperación de la inversión en un corto-medio plazo considerando el aumento de producción y disminución de elementos descartables.

Dando un enfoque social y laboral llega a tener el mismo nivel de relevancia, ya que se pretende reducir el esfuerzo físico del operario y llegar a automatizar tareas repetitivas, disminuyendo así lesiones ergonómicas [3] y mejorando la calidad de trabajo para el personal, mejorando así condiciones laborales y aumentando la seguridad laboral. Además de dar un mejor entorno laboral por la reducción de esfuerzo.

La viabilidad de este proyecto radica en el esquema metodológico planeado, que es la recolección de métricas mediante sensores, observaciones y medición de la integridad de los bloques, donde los indicadores principales serán: tiempo de entrega, niveles de aceleración durante el transporte, cantidad de elementos perjudicados en el proceso de transporte y precisión de posicionamiento final. Estas magnitudes nos permitirán demostrar objetivamente la construcción del prototipo a la eficiencia y calidad del proceso.

1.4 Objetivos.

1.4.1 Objetivo General

- Evaluar el uso de un robot de carga autónomo para el transporte de bloques frescos de construcción en la empresa “Bloquera Naula”, aplicando tecnologías IoT y visión artificial, y que cumpla con los requisitos de eficiencia y seguridad.

1.4.2 Objetivos Específicos

- Investigar y seleccionar algoritmos de visión artificial apropiados para la detección y reconocimiento de objetos relevantes en el entorno de operación del vehículo de carga autónoma.
- Diseñar e implementar un prototipo del vehículo autónomo de carga de bloques, utilizando tecnologías IoT y algoritmos de visión artificial.
- Evaluar el funcionamiento del prototipo de robot autónomo dentro del proceso de producción de bloques en la empresa “Bloquera Naula”.

CAPÍTULO II. MARCO TEÓRICO.

2.1 Estado del arte.

En investigaciones contemporáneas se ha observado una evolución de los sistemas de vehículos guiados hacia la capacidad de percibir, planificar y tomar decisiones dependiendo de su entorno, lo que ha permitido aplicarse en varios sectores como: manufacturas, industrias, logística, agricultura, educación, salud y área militar [12]. Particularmente en los últimos años hubo una drástica adopción de robots móviles autónomos (AMRs) debido a la pandemia de COVID-19 para reducir o evitar interacciones entre personas en actividades como transporte y desinfección, y por ende evitar una mayor propagación [13].
Los AMR modernos están formados por varios sensores, algoritmos y frameworks que facilitan las tareas fundamentales de la robótica móvil, como: navegación, localización y mapeo.

En los entornos de construcción sus avances han sido más limitados que en otras áreas, sin embargo, últimamente se ha visto la existencia de AMR para tareas de manejo de materiales de obra con particulares desafíos debido al ambiente donde se encuentran expuestos, como polvo, humedad, temperaturas, interacción con personal e incluso con maquinaria pesada. Esto demuestra que actualmente existen tecnologías que pueden adaptarse a la problemática del transporte de prefabricados de concreto en entornos semiestructurados y teniendo en cuenta los requisitos en seguridad [14].

Dentro de los parámetros que se consideran para determinar el rendimiento están magnitudes como tiempo, distancia, consumo energético, seguridad, tiempo de procesamiento, entre otras. Los métodos para la navegación autónoma se han clasificado en métodos clásicos y aproximaciones heurísticas. Su diferencia principal está en sus diseños, mientras los métodos clásicos fueron creados para entornos estáticos, los métodos heurísticos fueron diseñados para la problemática de moverse en entornos dinámicos. En estos dos métodos existen varias formas de resolver el problema de mapeo, la decisión radica primordialmente en el entorno, el grado de conocimiento del mapa, si es estático o dinámico o coste de procesamiento [12]. En la práctica, se han destacados métodos híbridos, es decir que emplean los tipos de métodos para una solución más robusta y funcional.

Tabla 1. Métodos de navegación destacados.

Método	Ventajas	Limitaciones	Escenario Optimo
VG (Visibility Graph)	Trayectorias óptimas euclidianas en 2D	Sensible al ruido geométrico	Mapas 2D con obstáculos bien definidos
VD / GVG	Separación máxima a los obstáculos	Rutas largas y desviación	seguro en ambientes compartidos
PRM	Escalable, rápida consulta	Reentrenamiento por obstáculos	Rutas repetitivas
Híbridos (Global + Local)	Unifica métodos para optimización	Complejidad de integración	Casos reales

En artículos recientes de comunidades como Visual SLAM y fusión LIDAR-cámara-IMU documentan avances en los problemas de ubicación, mapeo y movimiento de los AMR en ambientes dinámicos y despliegue en tiempo real en condiciones dinámicas y adversas [15], [16], Asimismo, nuevas librerías relacionadas a los sensores capaces de medir distancia mediante la detección de luz (LiDAR), unidades de medida inercial (IMU) y sensores visuales han demostrado ser una opción robusta y precisa para AMR que se encuentran en entornos industriales [17].

Uno de los conceptos dentro del mundo tecnológico se hace presente en estos trabajos, el Internet de las Cosas destinado a la industria (IIoT) permite el acceso en tiempo real a los datos, para analizar procesos y realizar mantenimientos preventivos antes de un error crítico. Con esta idea, se ha desarrollado la tendencia de implementar hardware de cómputo en la maquinaria (Edge computing), siendo su principal objetivo reducir la latencia de comunicación y disminuir la dependencia de la conexión a Internet.
Finalmente, ante distintos avances en áreas relacionadas se llegó al concepto de "Internet of Robotics Things" (IoRT), el cual se encarga en finalizar el bucle de percepción, decisión y acción [18].

Adicionalmente a estos desarrollos, las comunidades de software libre han participado y obtenido gran relevancia con sus aportaciones, como es el caso del framework ROS, donde sus contribuidores dan dirección al proyecto hacia la interoperabilidad y flujos industriales con la implementación de IIoT, añadiendo así conceptos de estructuras universales con arquitecturas unificadas, Protocolos de comunicación como MQTT (ligeros sobre TCP/IP) para la telemetría a través de la nube y móviles , favoreciendo así sus nuevas versiones, ROS-Industrial y ROS 2, donde se evolucionó en conceptos y aumentando su viabilidad [19].

En materia de normativas y estándares de seguridad para la implementación de AMR en entornos compartidos con operarios se establecen funciones de seguridad, verificación y responsabilidades del fabricante/usuario. La Organización Internacional de Normalización ISO define los requisitos necesarios para la operación de vehículos industriales sin conductor

y sus respectivos sistemas e indica la implementación de detección de personas, modos de operación y más. En las últimas ediciones del estándar con respecto a robots industriales, aumentan la rigurosidad de la normativa base para la colaboración humano-robot, indicando límites de velocidad y fuerza para evitar lesiones o golpes con varios umbrales respecto al cuerpo humano [20].

La inspiración de este estudio ha sido en gran medida a los montacargas autónomos, que han consolidado nuevas formas de percepción del entorno, principalmente mediante sensores LiDAR y cámaras de profundidad RGB-D, junto con algoritmos de detección y seguimiento para el tránsito en entornos controlados, además de funciones de seguridad [22]. A pesar de todo ello, su validación y diseño han sido en entornos controlados, que tiene un conocimiento total de su entorno y este mismo se diseña en base a ellos.

En la evaluación e implementación de un dispositivo AMR en el área definida, se pueden tomar varias operaciones y acoplarlas al diseño y variables de estudio, tales como: conjunto de sensores, métodos de planificación de rutas híbridos con respecto a la estabilidad de carga, reglas y zonas de tránsito seguro.

2.2 Fundamentación teórica

2.2.1 Robótica móvil.

Un robot móvil se lo define como una plataforma mecánica que consta con un sistema de locomoción, sensores y actuadores, con la capacidad de desplazarse por el entorno con algún grado de autonomía, con mínima o nula intervención humana. Su autonomía se define por la capacidad de comprender el entorno, planificar trayectorias y tomar acciones de acuerdo con lo obtenido [23].
En el área industrial se aplican normas referentes a este tema como la norma ISO 3691-4, que los reúne con los vehículos industriales sin conductos o sin operador a bordo En este grupo se los ha delimitado en dos conjuntos, los vehículos guiados y robots autónomos [21], como en la figura 1.

(a) (b)

Figura 1. Tipos de Robots *móviles*: (a). vehículo autónomo guiado, (b) Robot autónomo móvil. (a):[24];(b): [25]

2.2.1.1 AGV y AMR: definición y diferencias.

Como se menciona previamente, los robots móviles se han clasificado en dos tipos, diferenciándose en su sistema de navegación, donde uno toma sus decisiones según la información obtenida por sus sensores, mientras el otro caso depende de guías externas para seguir una ruta predefinida.

Vehículo guiado automático (AGV): Necesita de rutas predefinidas e infraestructura, además de supervisión externa para su navegación y cualquier modificación en la trayectoria necesita un cambio en la infraestructura.

Robot autónomo móvil (AMR): Percibe el entorno mediante sensores integrados con los cuales, localiza, planifica y plantea rutas para evitar obstáculos y adaptarse al flujo de trabajo sin necesidad de guías físicas.

Se puede apreciar que un AMR proporciona una mayor flexibilidad y adaptación con respecto a un AGV, sin embargo, este tiene un precio de tener una mayor complejidad en su sistema de percepción y control [12], [21], [22].

Se llega a determinar su nivel de independencia con respecto a la forma en cómo puede interactuar sobre su entorno, procesar esa información a órdenes y aplicarlas de locomoción y lograr movilizarse de hacia una meta.

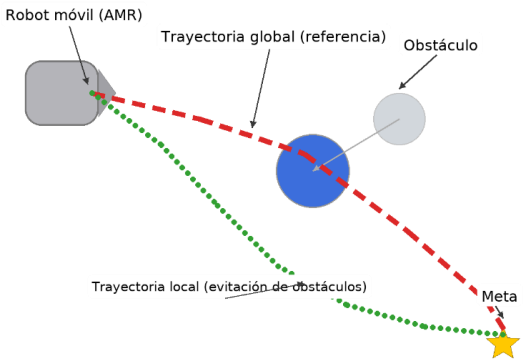
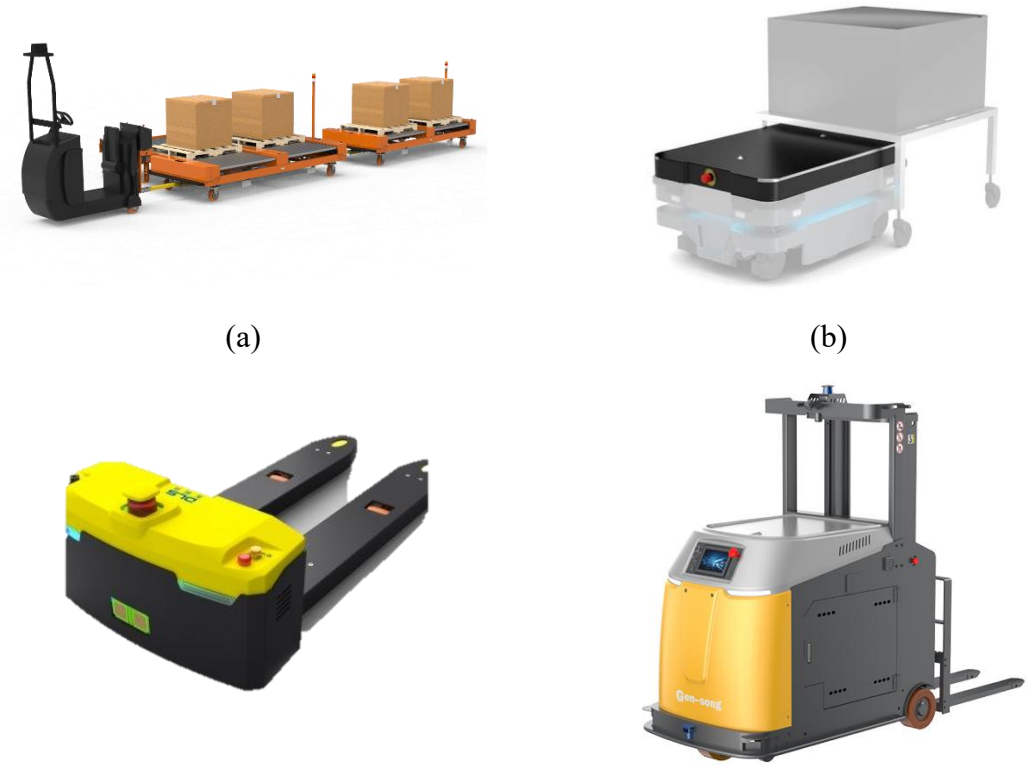


Figura 2. Generación de ruta alternativa por presencia de obstáculo.

2.2.1.2 Tipos de AMR.

Los AMR se han desarrollado para varios tipos de empleo, estos pueden ser diseñados para trabajos como: Arrastre de cargas (tugger), transporte de estanterías (Shelf-carrier), manejo de palés (pallet-AMR) y montacargas autónomos (autonomous forklift). Donde puede diferenciarse por la capacidad de carga, diseño mecánico y nivel de autonomía necesaria para un determinado flujo de trabajo. De esta clasificación el AMR que integra mayores avances y funciones relacionadas con la percepción tridimensional del entorno son los montacargas autónomos, ya que necesitan funciones avanzadas de seguridad para poder operar con las personas de manera cooperativa y elevación de cargas [11], [22]



(a) (b)

(c) (d)

Figura 3. Tipos de AMR. (a) Tugger, (b) Shelf carrier, (c) Pallet, (d) Forklift.(a): [26];(b):[27];(c):[28];(d):[29]

2.2.1.3 Clasificación por sistema de movimiento.

Las configuraciones empleadas en el diseño de los AMR dependen del entorno en el cual se movilizarán y las soluciones disponibles, de ellas han destacado configuraciones como: sistema diferencial (ruedas motorizadas y apoyo), giros por deslizamiento controlado, sistemas holonómicos y tipo orugas. Estos sistemas han sido empleados en varios escenarios con distintos objetivos, dentro de la implementación industrial, los AMR generalmente son no holonómicos, es decir que su movimiento no se encuentra definido por ecuaciones o la incapacidad de generar movimientos laterales instantáneos, por lo cual tiene la necesidad de realizar maniobras que combinan acciones como avanzar, retroceder y girar [6], [7], [8], [11]. Con esto en cuneta se puede determinar que el movimiento de un AMR está conformado por al menos 3 grados de libertad que son:

$$v_x = Velocidad\ lineal\ en\ x$$
$$v_y = Velocidad\ lineal\ en\ y$$
$$\omega = Velocidad\ angular$$

Tabla 2. Configuraciones de locomoción de un AMR

Configuración	Holonómicos	Movimiento instantáneo	Ventajas	Desventajas
Diferencial	No holonómico	v_x, ω	Sencillo, eficiente y robusto	Necesita de maniobras, radio de giro limitado
Deslizamiento Controlado	No holonómico	v_x, ω	Mecánica simple, buena tracción	Deslizamiento lateral, mayor desgaste de las ruedas
Oruga	No holonómico	v_x, ω	Baja presión en el suelo, excelente tracción	Menor Eficiencia, Costo elevado y mayor mantenimiento
Omnidireccional	Holonómico	$v_x v_y, \omega$	Maniobrabilidad en espacios reducidos	Agarre reducido y baja eficiencia

2.2.2 Modelo Cinemático de un AMR de Tracción Diferencial.

Para el estudio del movimiento de un robot móvil se hace uso de la cinemática clásica, quien centra su análisis del movimiento en base a la geometría del mismo, dando como resultado: un modelo matemático útil para el diseño de control, el comportamiento del movimiento que se puede comprobar con simulaciones y ecuaciones con las cuales se puede estimar su posición en x, y, θ [30].

El modelo empleado en el diseño del AMR es de tipo tracción diferencial, con una configuración de dos motores independientes en cada rueda y un caster, denominado también base diferencial con rueda pivotante. Su comportamiento cinemático es similar al de un diferencial clásico, ya que la rueda pivotante no participa en la generación de movimiento.

2.2.2.1 Cinemática de un robot móvil diferencial (2D).

En robótica móvil, la cinemática describe el cambio que presentan la posición y orientación del robot en el tiempo en base a sus velocidades. En un AMR de tracción diferencial clásico (dos ruedas motrices), el movimiento está condicionado por las restricciones de rodadura de las ruedas, lo cual provoca que este sea un sistema no holonómico (limita direcciones instantáneas de movimiento) que no afecta a la posición [31] .

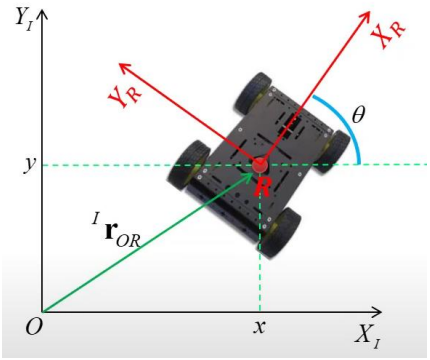


Figura 4. Marcos y pose del robot [31].

La figura 4 muestra dos sistemas de referencia utilizados, sistema de referencia inercial {I}, el cual es global y fijo, además del sistema de referencia del robot {R}, que es representado con punto R, donde este punto se puede representarlo con un vector r.

$$I_{r_{R0}} = \begin{bmatrix} x \\ y \\ 0 \end{bmatrix} \qquad (1)$$

Este vector está formado por coordenadas del sistema inercial $x, y, z = 0$, por encontrarse en el plano XY.
Para la orientación del robot se usa el sistema de referencia de este {R}, donde su punto de referencia es R , además del giro del robot en base a {I}, representado por θ (giro alrededor del eje Z). Para la representación de estos valores de posición y orientación se hace uso de un solo vector, ya que $z = 0$, con ello se llega al vector de posición y orientación.

$$I_{\xi=\begin{bmatrix} x \\ y \\ \theta \end{bmatrix}} \qquad (2)$$

La velocidad inercial será la derivada con respecto al tiempo de la posición y orientación.

$$I_{\dot{\xi}=\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix}} \qquad (3)$$

Estas expresiones se encuentran con respecto al sistema inercial {I}, si se desea cambiar de sistema al de robot {R} se emplea una matriz de rotación.

$$R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \qquad (4)$$

donde

$$\theta_I = \theta_R = \theta \qquad (5)$$

$$\begin{bmatrix}\dot{x}\\ \dot{y}\end{bmatrix}^{I} = R(\theta)\begin{bmatrix}\dot{x}\\ \dot{y}\end{bmatrix}^{R} \quad (6)$$

$$\begin{bmatrix}\dot{x}\\ \dot{y}\end{bmatrix}^{R} = R(\theta)^{T}\begin{bmatrix}\dot{x}\\ \dot{y}\end{bmatrix}^{T} \quad (7)$$

Para pasar del sistema robot al del inercial se debe emplear la ecuación (6) y viceversa (7).

2.2.2.2 Restricciones cinemáticas por el tipo de rueda.

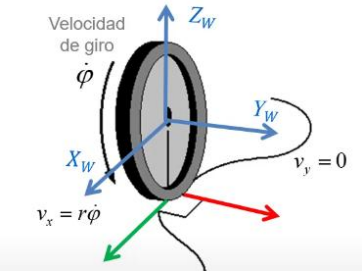


Figura 5. Valores de una rueda simple [32]

Existen restricciones de movimiento que están dadas por el tipo de rueda que se emplea, debido a la geometría de esta. En una rueda fija existen dos valores, siendo de radio r y ϕ la velocidad de giro de la rueda y cuenta con dos restricciones de movimiento, los cuales son:

- Rodadura. – Gira únicamente en el plano que lo contiene (no se desliza) $v_x = r\phi$
- No deslizamiento. – La rueda no se resbala lateralmente $v_y = 0$

Estas condiciones limitan los grados de libertad del movimiento y permiten relacionar de forma consistente v, ω y las velocidades angulares de rueda ω_d(derecha) y ω_i (izquierda). En la Figura 5 se ilustran los marcos y la notación utilizada para aplicar dichas restricciones.

2.2.2.3 Restricciones cinemáticas de un robot diferencial.

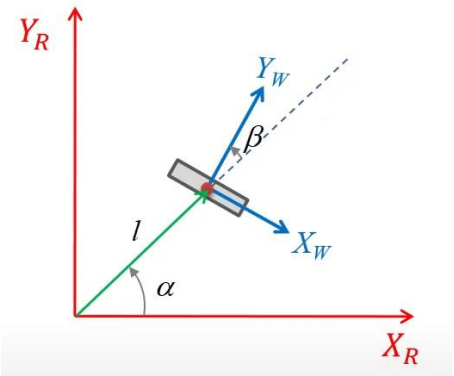


Figura 6. Restricciones de la rueda [32].

Partiendo desde las ecuaciones de restricciones de las ruedas es necesario encontrar los parámetros de α, β, l indicados en la figura 6. Donde las ruedas convencionales son fijas, es decir que β se mantiene constante, además de ser paralelo Y_w y Y_R como muestra la figura 7, el valor de β en la rueda izquierda será de 0 grados y en la rueda derecha debido a la rotación del sistema de referencia será de 180 grados. De igual manera los valores de α serán de 90 y -90 y la distancia b se mantendrá igual en los dos casos.

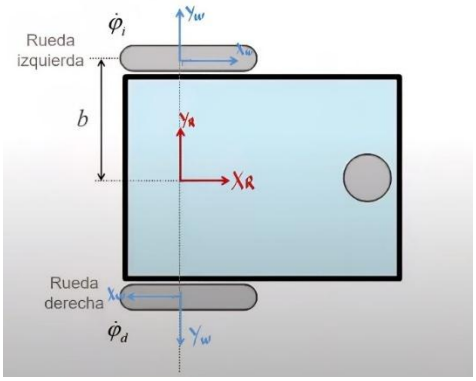


Figura 7. Sistema de referencia del robot(R) y ruedas(W).

Con los valores de α, β, l encontrados se reemplaza en las ecuaciones de las restricciones de las ruedas convencionales.

$$[\sin(\alpha+\beta) - cos(\alpha+\beta) - lcos\beta]\xi = r\phi \quad (8)$$

$$[cos(\alpha+\beta)sin(\alpha+\beta)lsin\beta]\xi = 0 \quad (9)$$

Al reemplazar los valores nos da unos valores que se puede representar de manera matricial, siendo la matriz de restricciones A y B la representación el radio de las llantas y ceros [31].

$$A = \begin{bmatrix} 1 & 0 & b \\ 1 & 0 & -b \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}; B = \begin{bmatrix} r & 0 \\ 0 & r \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \quad (10)$$

De manera simplificada la ecuación de restricción de un sistema diferencial de dos ruedas convencionales fijas será:

$$A_\xi^R = B\phi \quad (11)$$

Finalmente, de la ecuación 11 que es la fórmula general del movimiento cinemático de un robot diferencial se puede obtener la cinemática directa y cinemática inversa despejando sus valores correspondientes ξ para obtener la cinemática directa y ϕ para obtener los valores de velocidades de las ruedas.

$$\xi = A^{\#}B\dot{\phi} \rightarrow \begin{bmatrix} r/2 & r/2 \\ 0 & 0 \\ \frac{r}{2b} & \frac{r}{2b} \end{bmatrix} \begin{bmatrix} \phi_d \\ \phi i \end{bmatrix} \quad (12)$$

$$\dot{x} = v = \frac{r}{2}(\phi d + \phi i) \quad (13)$$

$$\theta = \omega = \frac{r}{2b}\left(\dot{\phi}_d - \dot{\phi}_i\right) \quad (14)$$

En la ecuación 12 es la representación matricial de los valores de velocidad y posición del robot en función de la velocidad angular de las ruedas. De igual manera se parte de la ecuación 11 para obtener la cinemática inversa.

$$\phi = AB^{\#}\xi \rightarrow \begin{bmatrix} \phi_d \\ \phi_i \end{bmatrix} = \begin{bmatrix} 1/r & 0 & b/r \\ 1/r & 0 & b/r \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \theta \end{bmatrix} \quad (15)$$

$$\phi_d = \frac{1}{r}(v + b\omega) \quad (16)$$

$$\phi_i = \frac{1}{r}(v - b\omega) \quad (17)$$

De la misma manera se representa de forma matricial de las velocidades angulares de las ruedas en función de la velocidad y orientación del sistema de referencia del robot. Como se mencionó al principio de esta sección, es necesario utilizar una matriz de rotación para cambiar el sistema de referencia, en este caso del sistema de referencia del robot {R} al sistema inercial{I}.

$$I_{\dot{\xi}} = R(\theta)^R\dot{\xi} \rightarrow \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix}^I = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ 0 \\ \omega \end{bmatrix} = \begin{bmatrix} v\cos\theta \\ v\sin\theta \\ w \end{bmatrix} \quad (18)$$

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix}^I = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}^R \quad (19)$$

2.2.3 Tecnologías de navegación y control.

La navegación autónoma en robots móviles se organiza en distintas tomas de decisiones, que transforma la percepción del entorno en movimiento del robot. Inicia con la localización y mapeo del AMR en el entorno, para este proceso se utilizan algoritmos SLAM (Localización y mapeo simultaneo). Posteriormente a comprender el entorno y su posición es necesario la implementación del método de navegación, quien planifica las trayectorias a aplicarse y efectúa acciones en su sistema de locomoción [33].

Uno de los Frameworks más empleados para el diseño de AMRs es ROS (Robot Operating System), un conjunto de herramientas y librerías que facilitan el desarrollo de aplicaciones enfocadas en la robótica, en su versión reciente ROS 2, cambia el esquema de comunicación interno de ROS, conformada por un nodo maestro y conexiones TCP y UDP a DDS (Data Distribution Service) un middleware para la comunicación publicador-suscriptor, mejorando el despliegue de múltiples dispositivos permitiendo mayor escalabilidad y flexibilidad. [34].

Con la aplicación del sistema modular de ROS 2 y los algoritmos diseñados para la navegación autónoma, puede responder a las incógnitas relacionadas al tema, posición y mapeo con SLAM, planificación de movimiento y controlador con NAV2.

2.2.3.1 Navegación de Robots.

La navegación de los robots se puede dividir en tres fases principales:

- Mapeo del entorno
- localización en el entorno
- Planeamiento de ruta.

El mapeo y localización están estrechamente relacionados y son las principales tareas de las que se encarga SLAM, este algoritmo no trabaja en la parte de planificación de ruta, por lo cual facilita las dos primeras fases de navegación de un robot [35].

2.2.3.2 Mapeo del entorno.

La creación del mapa del entorno es un proceso indispensable para la navegación del robot, debido a ser utilizado en la fase de localización, un mapa es la reconstrucción espacial del entorno del robot, denominada modelo y se obtiene a partir de sensores [35].
Dentro de los sensores, el principalmente utilizado para SLAM son las cámaras RGB-D y complementarse con sensores adicionales como, escáneres LiDAR o ultrasonidos.
Existen 3 maneras en representar los mapas:

- Mapas de cuadricula por ocupación
- Mapas basados en características
- Mapas topológicos

Con los mapas de cuadricula de ocupación representan el entorno con información espacial probabilística. El mapa se divide en cuadriculas y se les asigna un valor en referencia a la

probabilidad que esté ocupada el espacio de esta. Existen 3 estados en los que puede estar una cuadricula: libre, ocupado y desconocido. Se representa el valor de las cuadriculas con espectro monocromático, mientras más oscura la cuadricula la probabilidad de estar ocupada aumenta [35], [36].

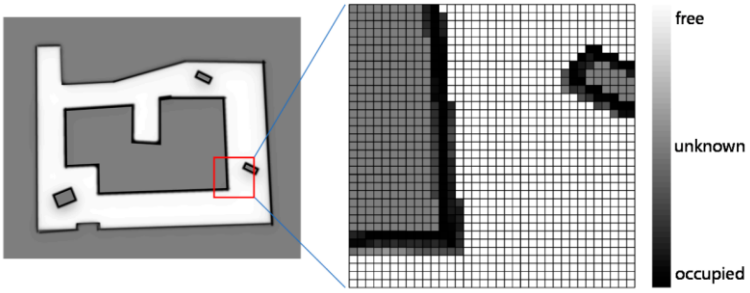


Figura 8. Ejemplo de mapa de cuadricula de ocupación.

Generar este tipo de mapas requiere de una potencia de cálculo y memoria a considerar. Una opción a este son los mapas basados en características, en el cual se extraen características esenciales del entorno como puntos, líneas y planos, reduciendo así la potencia y memoria necesaria para su representación como muestra a continuación [35].

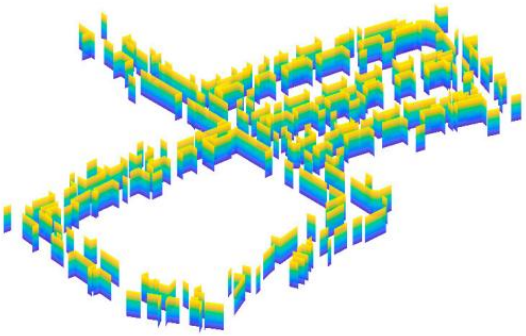


Figura 9. Ejemplo de mapa basado en características [36].

Adicionalmente existe la representación con mapas topológicos, donde se presenta con nodos que indican alguna zona o característica conectado con aristas que señalan una ruta entre los nodos, con información adicional de conectividad y distancia, generalmente son rutas únicas. Este tipo de mapas son empleados por sus atributos que representan ventajas en la localización, navegación y planteo de rutas [36].

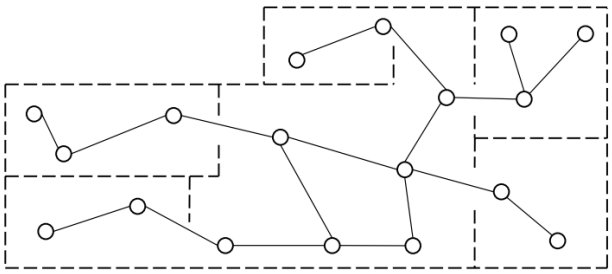


Figura 10. Ejemplo de mapa topológico [36].

Para establecer la posición del robot en el entorno no es suficientemente confiable el uso de la odometría, por ello se hace uso de sensores adicionales, como sensores inerciales (IMU)

y cámaras RGB-D. Con estos valores adicionales que nos proporcionan los sensores, SLAM tiene la capacidad de corregir el error de la odometría.

2.2.3.3 Planteamiento de ruta.

La navegación de un AMR se fundamenta en la planificación de rutas para determinar el recorrido entre el origen y la meta, teniendo en cuenta la optimización de tiempo, distancia, seguridad y energía. Los métodos de navegación se clasifican en dos familias: clásicos (efectivos en entornos conocidos) y heurísticos (Robustos ante incertidumbre y escenarios dinámicos). Actualmente los métodos más usados los enfocados en escenarios dinámicos, sin embargo, varios métodos clásicos siguen presentes en los estudios [12].

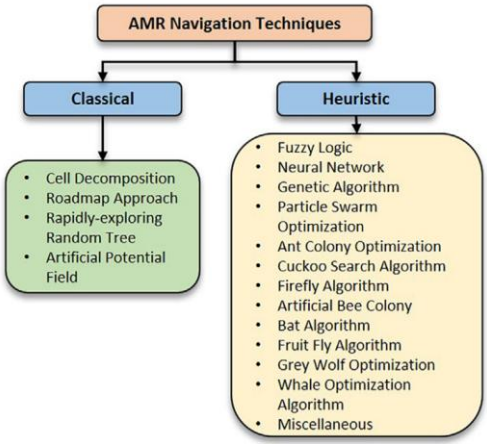
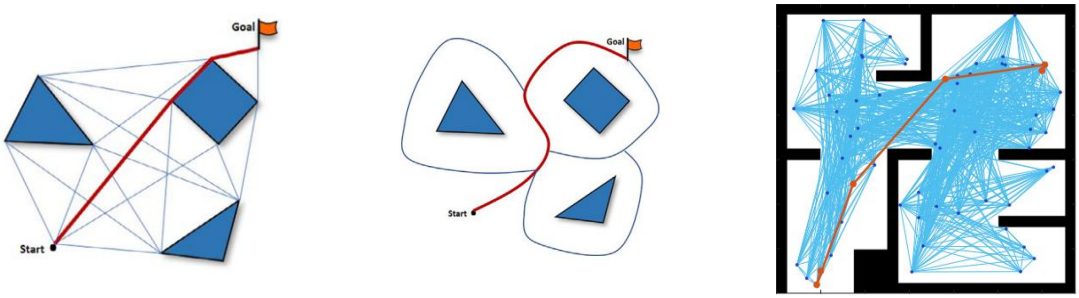


Figura 11. Técnicas de navegación clásicas y heurísticas [12].

En los métodos clásicos destacan los métodos de enfoque de ruta (RA), los cuales construyen su navegación en el espacio con nodos y aristas sin colisión que posteriormente eligen el mejor camino, al ser parte de los métodos clásicos, necesita conocer el escenario en donde se encuentra o actualizándose en tiempo real como en SLAM.



(a) Grafos visuales (b) Diagrama de Voronoi (c) Hoja de ruta probabilística

Figura 12. Ejemplos de distintos métodos de navegación basados en enfoque de rutas [12].

Con los métodos de enfoque de ruta se tiene tres formas de crear los grafos de navegación: Grafos visuales (VG), Diagramas de Voronoi (VD) y Hojas de ruta probabilísticas (PRM).

De estos grafos existen distintas rutas de la cual se elige el mejor camino con buscadores, los más empleados con RA son:
Dijkstra (DA). - Encuentra la distancia mínima desde el inicio al resto de nodos, empleado en grafos pequeños.
A*. - Encuentra un camino admisible de costos, consiguiendo el valor mínimo hasta la meta, estos costos se dan dependiendo: distancia, tiempo, energía, riesgo.
D* Lite. - Replanifica el camino cuando cambian los costos o se descubren obstáculos en el camino. Este método de búsqueda es muy utilizado con SLAM que actualiza el mapa.

2.2.3.4 Control de movimiento y odometría.

La ejecución del movimiento se fundamenta en el modelo cinemático diferencial de 2.2.2. el cual envía un comando (v, ω) al local, se calculan las velocidades de ruedas (ω_i, ω_d) por cinemática inversa (ecs. (15)–(17)) y se envían al controlador de motor de cada rueda. Este bucle superior se ejecuta en ROS 2 a través del servidor de control (Nav2).
La odometría diferencial es susceptible a deslizamientos, desfases y deriva, por ello se emplea la fusión sensorial mediante un Filtro de Kalman Extendido (EKF) para combinar odometría, IMU y SLAM [33].

2.2.3.5 Visión Artificial en Robots Móviles.

La visión artificial cumple 3 funciones principales:

- Estimación del movimiento
- Localización y mapeo
- planificación y evasión

Estas funciones brindan una mejora a las herramientas que se implementan para realizar la navegación autónoma, como sería limitar los errores de odometría tradicional, con un sistema adicional de odometría visual, construir y mantener un mapa acorde al entorno con algoritmos como VSLAM, finalmente con ello realizar la planificación y evasión de obstáculos con Nav2 [37] .

Tabla 3. Funciones principales de la visión artificial en un AMR

Función	Algoritmo	Propósito
Estimación del movimiento	Odometría Visual (OV)	Limitar errores de posición por desfases en la odometría
Localización y mapeo	Visual SLAM (VSLAM)	Construcción del mapa y ser consciente del entorno
Planificación de ruta	Navegation2 (Nav2)	Creación y selección de la ruta óptima desde el punto de partida a la meta.

Para la implementación de visión artificial a un AMR es necesario la existencia de sensores diseñados para dicha función, para ello se consideran principalmente cámaras de profundidad (RGB-D) y cámaras estéreo rectificadas (arreglo de cámaras donde conocen su distancia y calculan la distancia de objetos mediante trigonometría) como fuente de visión principal.

(a) (b)

Figura 13. Cámaras para visión artificial. (a) cámara estéreo, (b) cámara RGB-D.

Las cámaras de profundidad RGB-D constan de infrarrojos que proyectan un patrón determinado de puntos y mide su deformación empleando triangulación para la reconstrucción de la escena como se muestra en la siguiente figura.

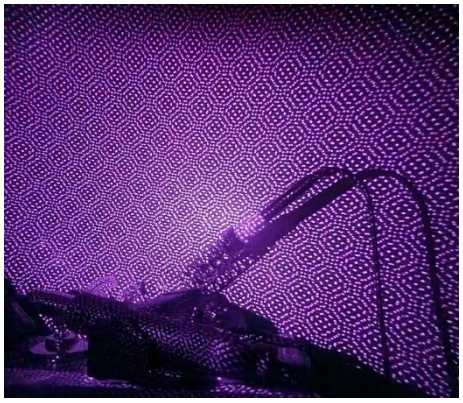


Figura 14. Malla de puntos de cámara Intel REALSENSE D415.

Estos sensores se complementan con unidades de medición inercial (IMU) y encoders como señales auxiliares que estabilizan la estimación de la posición, movimiento y aumentan la certeza del mapa.

Tabla 4. Tabla de señales emitidas por los sensores.

Origen	**Tipo de dato en ROS 2**	**Unidad**	**Uso en el proceso**
Imagen color	`sensor_msgs/Image`	px (8-bit)	VO/VSLAM (features), visualización
Profundidad alineada a color	`sensor_msgs/Image`	m (float32/píxel)	Percepción 3D, proyección a costmaps
Nube de puntos	`sensor_msgs/PointCloud2`	m (XYZ)	Obstáculos 3D
IMU (orientación)	`sensor_msgs/IMU`	rad/s, m/s²	VIO/VO (asistencia), fusión EKF
Encoder ruedas	`std_msgs/Int32` (acum.)	ticks	Odometría de ruedas
Velocidad rueda	`std_msgs/Float32`	rad/s o m/s	Control/diagnóstico
Odometría por ruedas	`nav_msgs/Odometry`	m, m/s	Entrada a EKF (robot_localization)

2.2.3.6 Odometría Visual y Fusión Visual-Inercial.

La odometría visual (VO) logra estimar la posición del robot en base de una serie de imágenes para la determinación de una trayectoria aproximada y velocidades en un marco de referencia en el nodo "/odom". La precisión inicial es alta, sin embargo, mientras aumenta la trayectoria recorrida sufre una deriva acumulativa por falta de mecanismos de corrección global. La adición de un sensor IMU da como resultado la odometría visual-inercial (VIO), con el cual se logra estabilizar el seguimiento ante casos donde se deriva la trayectoria por: desenfoque por movimiento, vibraciones y regiones con poca información [15].
Dependiendo del sensor que se emplea, existen 3 configuraciones comúnmente utilizadas para la aplicación de odometría visual.

Tabla 5. Comparación de Configuraciones para odometría visual.

Configuración	Ventaja	Desventaja
Monocular	Mas económica y ligera	No controla la escala observable
Estéreo	Posibilidad de determinar la escala	Delicada calibración y sensibilidad a la deriva.
RBG-D	Creación de mapa de profundidad por pixeles. Robusto en entornos normalizados.	Costo más alto del tipo de sensor. Mayor costo computacional. Sensible a la luz solar directa.

El flujo de procesamiento en los casos de VIO se basa en la adquisición sincronizada de datos de la cámara e IMU. Por la parte visual, se detectan elementos distintivos, se asocia su posición entre fotogramas y se estima la posición, refinándolo con filtros integrados en los algoritmos. La adición de un IMU logra reducir una cantidad considerable de comparaciones para la estimación de la orientación y velocidad, de igual manera se emplean filtros para la fusión de información y corregir los sesgos creados el giroscopio y acelerómetros [15]. En base a lo analizado se puede determinar una configuración similar de método de visual SLAM a emplear, como se ve en la siguiente figura.

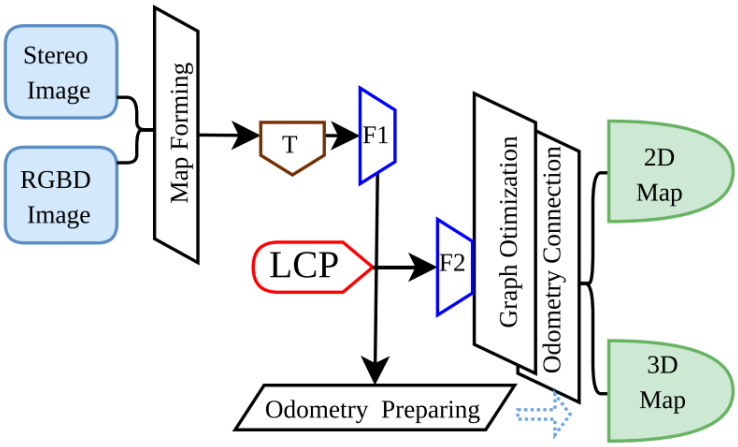


Figura 15. método de visual SLAM simplicada. RTAB-MAP-SLAM [15].

La configuración planeada para el sistema de localización y mapeo es similar al método indicado en la fig. 15, debido a que integra una cámara RGB-D y unidad inercial, donde se calcula la posición en base a la información visual y movimiento. además de tener una ventaja fundamental, su integración nativa con ROS 2, facilitando significativamente la integración de sensores y algoritmos, por lo cual, se obtiene una arquitectura robusta, escalable y confiable para la aplicación en AMRs [37].

2.2.3.7 Visual SLAM.

Visual SLAM es un sistema donde la cámara obtiene una secuencia de imágenes que se procesan para estimar la posición de la cámara y de manera paralela la construcción de un mapa que representa el entorno [37]. En el caso de las cámaras RGB-D e IMU, el sensor de profundidad de la cámara alineada con el color simplifica la reconstrucción geométrica.

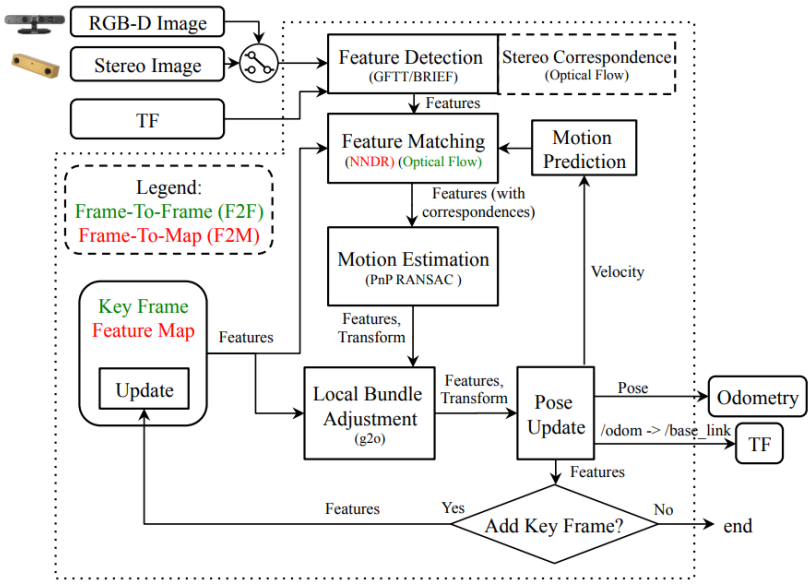


Figura 16. Diagrama general del proceso de estimación visual en sistemas VSLAM (Front-End) [37].

La arquitectura de divide en un front-end que estima la posición relativa entre los fotogramas, al utilizar RGB-D, se inicia la transformación cámara-mundo (calcular la posición de la cámara en el entorno) con el problema Perpective-n-Point (PnP) aplicadas sobre los datos 2D y 3D obtenidos de los fotogramas y mapas de profundidad, al realizar este proceso, el algoritmo genera puntos incorrectos, como bordes repetitivos o ruido, esto lo limitan con el método RANSAC (Random Sample Consensus), que se encarga de:

- Prueba varias combinaciones de puntos
- Descarta resultados incoherentes
- Selecciona la solución más consistente.

Con ello se obtiene una posición aproximada la cámara, posteriormente ajusta simultáneamente la posición de los keyframes en los puntos 3D observados con el proceso "Bundle Ajustment" y obtener una actualización de la posición [37].

Un keyframe es un fotograma representativo que se selecciona por su novedad visual o geométrica, este almacena observaciones que sirven para el seguimiento y la optimización. Elegir buenos keyframes reduce redundancia, mejora la estabilidad numérica y concentra el cálculo en los momentos que más aportan a la consistencia del mapa y la trayectoria [37].

Mientras dentro del back-end se toma las posiciones locales que entrega el front-end y las organiza en un grafo de poses, donde los nodos representan keyframes, restricciones odométricas y de cierre de bucle. Cada restricción lleva su incertidumbre, de modo que la optimización global pondera más las observaciones confiables. Cuando el sistema detecta que el robot ha regresado a un lugar ya visitado, añade una restricción de largo alcance y ejecuta una optimización no lineal que corrige la deriva y produce una trayectoria global

coherente. Con la cámara RGB-D (D415), estas restricciones pueden verificarse geométricamente y proyectarse a un mapa de ocupación 2D o un modelo 3D [35], [37].

2.2.3.8 RTAB-MAP y Nav2.

El método RTAB-MAP se basa en gráficos creados a partir de nodos y enlaces, integrado nativamente en framework ROS y compatible con odometrías externas, sea odometría a partir del modelo cinemático, VO/VIO y LiDAR, entregando a su salida la posición y mapas de ocupación) que son publicados como tópicos de ROS en línea [37].

Figura 17. Mapa de ocupación 2D [37].

Nav2 toma los datos de RTAB-MAP para la localización, planeamiento y evitación de obstáculos, con el siguiente flujo de trabajo:

1. Localización Global. - RTAB-Map entrega la relación entre ‘map’ y ‘odom’. Así, la odometría local del robot (encoders/IMU/VO) queda anclada dentro del mapa global y se corrige cuando hay cierre de bucle.

2. planificación Global. - Nav2 toma el mapa de ocupación publicado por RTAB-Map como mapa estático para calcular rutas en el plano

3. Evasión de obstáculos en tiempo real. – El mapa de costos se actualiza en tiempo real y utilizando una nube de puntos para determinar los obstáculos cerca del AMR

4. Control de movimiento. – Nav2 convierte la ruta establecida a comandos hacia los motores para el movimiento de AMR, al momento de un obstáculo nuevo Nav2 lo replanifica la trayectoria utilizando el mapa global y costo de ruta.

5. Cierre de bucle. - Cuando el AMR regresa a una zona conocida, RTAB-Map detecta el fin del bucle, optimiza el grafo actualizando la relación entre ‘map’ y ‘odom’. Resultando en que el mapa global y la pose quedan sin saltos en el control.

2.2.4 ROS 2: Fundamentos, transporte de datos e integración con tecnologías de navegación.

ROS 2 es un conjunto de herramientas de desarrollo (Framework) pensada en la creación de sistemas robóticos en general, Su forma de comunicación se basa en el tipo publicación-suscripción que son representados con nodos y comunicados por tópicos, servicios y acciones para el intercambio de mensajes. La comunicación interna de ROS 2 se basa en una capa intermedia DDS (Data Distribution Service) la cual facilita la comunicación entre distintos componentes, sean sensores, dispositivos y aplicaciones dentro de una red local [5].

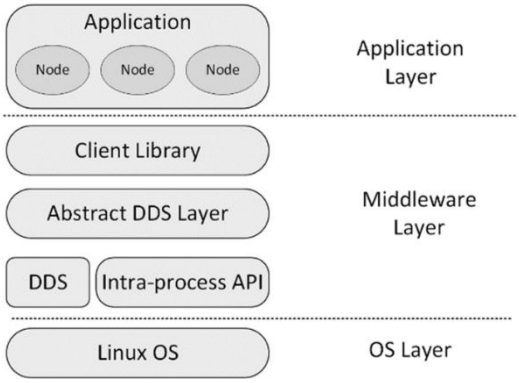


Figura 18. Arquitectura ROS 2 [38].

Para el trasporte y preprocesamiento de imágenes. Los nodos creados desde la cámara publican distintos mensajes como color, profundidad o nubes de puntos para la detección de obstáculos y brindar datos a los algoritmos de VO y SLAM, estos nodos se sincronizan mediante un nodo dedicado a (estampas de tiempo) y se aplican filtros que reducen el ruido y discontinuidades [37].

Para una comunicación más eficiente dentro de la red, los mensajes son comprimidos, sin perder sincronización y optimizando el ancho de banda, posterior a ello se realiza la transformación de los datos para la referencia entre los marcos del robot y la cámara para el posicionamiento de los puntos correctamente con respecto al AMR. [37].

Para la integración de los algoritmos de visión artificial y sensores, se parte desde la odometría local, la cual se obtiene combinando la información de los controladores de los motores e IMU la que es tratada con un filtro de Kalman al nodo de localización del robot (robot_localización) que nos brinda la estimación de la posición del robot apta para un control suave y listo la combinación con un nodo de VO con el fin de una mayor estabilidad. Con los nodos de la cámara de color y profundidad son procesados con RTAB-MAP y construye un mapa de ocupación que contiene información entre la posición del AMR, el marco de referencia global y odometría local. Cuando el AMR regresa a una zona ya visita anteriormente, RTAB-MAP cierra el lazo, optimiza el grafo y actualiza si posición corriendo la deriva sin comprometer el control. [37].

En paralelo al algoritmo RTAB-MAP, Nav2 utiliza los mapas generados para generar el cost map que se actualiza en tiempo real para determinar obstáculos cercanos y planificar automáticamente la ruta que convierte en comandos a los motores del AMR.

En retrospectiva, el uso de visión artificial permite información adicional que aporta una mejor precisión de odometría, aporta a RTAB-MAP datos para asegurar un mejor reconocimiento del entorno global y permite a Nav2 planificar y realizar acciones en el AMR, sincronizado sobre el framework ROS 2.

Tabla 6. Conceptos generales empleados en ROS 2.

Concepto	Definición breve
Nodo	Procesos de ROS 2
Tópico	Canal de comunicación entre nodos
Marcos de referencia	Sistema de coordenadas y transformaciones
Odometría local	Estimación para el control
Mapa de ocupación	Rejilla 2D libre/ocupado/desconocido
Costmap	Mapa de costo para clasificar mejor ruta

2.2.5 Internet of Things y Comunicación de Sistemas Autónomos.

El internet de las cosas (IoT) en entornos industriales varia su concepto debido al entorno, el internet industrial de las cosas (IIoT) que se utiliza principalmente para el control de procesos en tiempo real, seguir eventos de operación y revisión de indicadores [18]. Por ende, en este proyecto IoT se emplea como una forma de supervisión del funcionamiento del AMR.

La capa de IoT se puede utilizarse para el control y observación de los siguientes datos.

- Telemetría de operaciones: estado del AMR y avance del trabajo.
- Datos del transporte: tiempos realizados y distancias recorridas.
- Análisis de rendimiento: porcentajes de éxitos al llegar a la meta y estabilidad en el trayecto.
- Supervisión remota: visualización y alertas fuera de la red de control.

De esta manera divide el tráfico para reducir tráficos que puedan afectar los datos para la navegación de AMR en la red local y mantenerla aislada. Los datos para control y observación se pueden publicar por bróker MQTT lo que facilita el análisis del desempeño sin comprometer la operación del sistema.

CAPÍTULO III. METODOLOGÍA.

3.1 Tipo de Investigación.

La investigación es de enfoque cuantitativo, ya que el desempeño del prototipo se evalúa mediante mediciones de variables como tiempo, estabilidad, velocidad y precisión, las cuales permiten un análisis estadístico. Asimismo, presenta un diseño experimental al posibilitar la manipulación de parámetros de operación, como los algoritmos de navegación, para analizar su efecto en las variables de estudio.

3.2 Diseño de Investigación.

En el estudio adopta un diseño experimental con referencia a las distintas configuraciones de algoritmos de navegación y medición del desempeño del AMR en su flujo de trabajo, debido a su carácter experimental, podemos controlar condiciones, ajustar parámetros y validar el prototipo en un entorno representativo hasta estabilizar su rendimiento.

3.3 Técnicas de recolección de Datos.

Los datos se obtendrán mediante sensores que registran variables como velocidad, aceleración y tiempo del AMR, mientras que las variables no captadas directamente se evaluarán mediante observación y medición específica, como la exactitud de entrega.

Tabla 7. Operacionalización de variables.

Variable	Concepto	Indicadores	Técnicas e Instrumentos
	Independiente		
Tipo de algoritmo	Algoritmos utilizados para el desplazamiento del dispositivo	**Tipo**	**Observación**
	Dependientes		
Tiempo de Entrega	Tiempo en transportar una carga desde el punto de inicio hasta el destino.	**Segundos**	**Cronometraje**
Estabilidad del Dispositivo	Estabilidad del dispositivo, determinada por las aceleraciones en el eje Z .	**m/s2**	**Sensor de Vibraciones IMU**
Velocidad del Dispositivo	Velocidad a la que el dispositivo se desplaza en el entorno.	**m/s**	**Prototipo de manera automática**
Exactitud de la Entrega	La aproximación de la posición de entrega del prototipo respecto a la distancia total recorrida.	**Error**	**Prototipo de manera automática**

3.4 Población de estudio y tamaño de muestra.

La población de estudio estará conformada por los datos de las variables dependientes obtenidas en las pruebas realizadas con éxito, ya que cuentan con una interpretación homogénea

3.5 Hipótesis.

La presente investigación evalúa el efecto del Tipo de algoritmo usado para el desplazamiento (variable independiente) sobre el desempeño del AMR en la "Bloquera Naula". Se formulan las siguientes hipótesis:
H1. El Tipo de algoritmo usado para el desplazamiento influye significativamente en el Tiempo de Entrega, la Estabilidad del Dispositivo, la Velocidad del Dispositivo y la Exactitud de la Entrega del AMR, observándose diferencias entre configuraciones.
H0. No existen diferencias estadísticamente significativas en las variables de desempeño del AMR entre los distintos tipos/configuraciones del algoritmo de desplazamiento evaluados.

3.6 Métodos de análisis y procesamiento de datos.

Para el procedimiento del análisis y procesamiento de datos se plantea una organización de tres fases:
Fase 1: Preparación y Selección de tecnologías.
Fase 2: Desarrollo del prototipo.
Fase 3: Implementación final y evaluación.

En la siguiente figura se muestra el diagrama de trabajo divido en las fases, donde se observan las acciones a realizar dentro de las mismas y el resultado de estas.

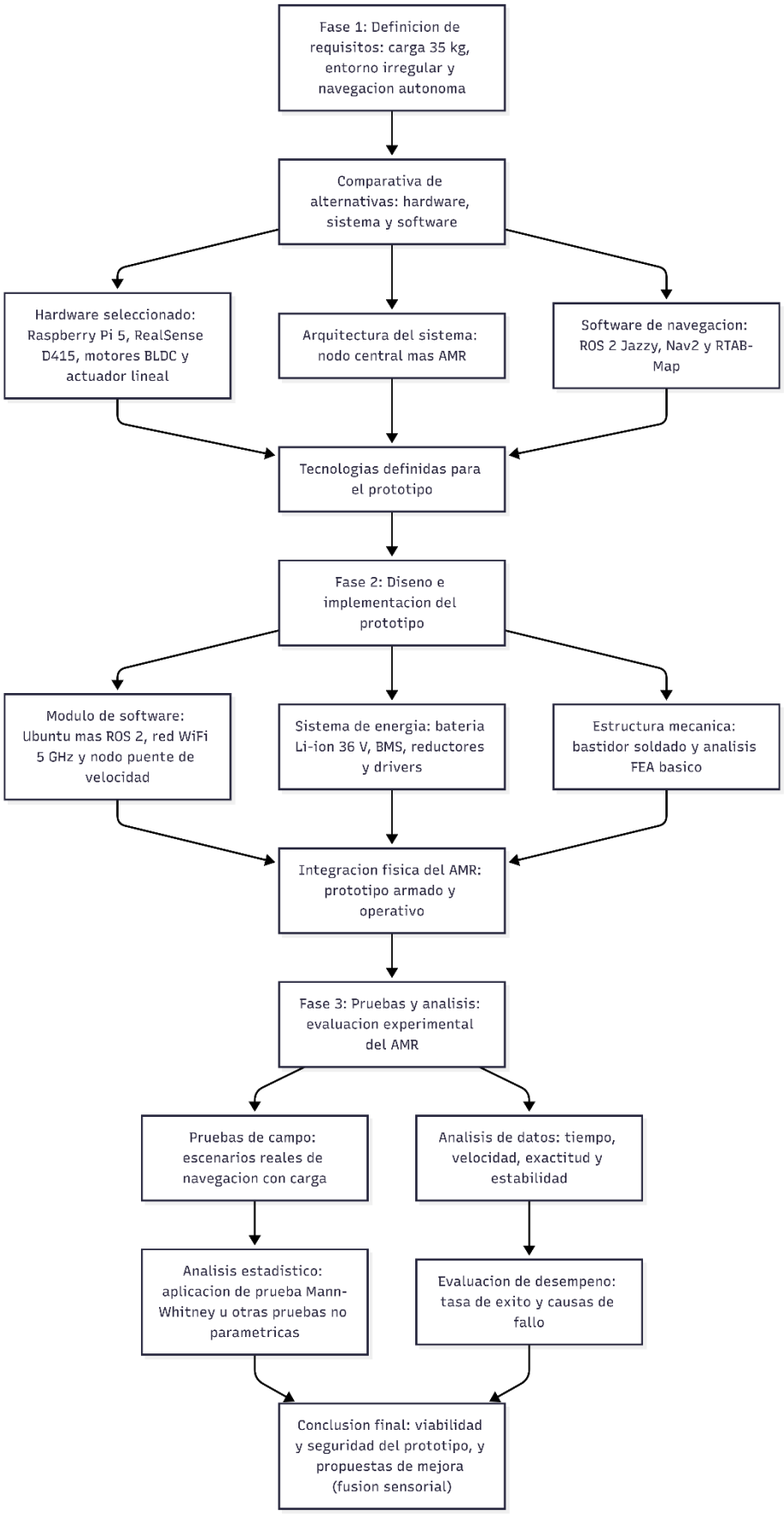


Figura 19. Diagrama de flujo de las fases.

3.7 Diseño e implementación del prototipo.

3.7.1 Fase 1. Preparación y Selección de tecnologías.

Para la preparación del prototipo se da requisitos al AMR para realizar el transporte de unidades recién fabricadas para seleccionar con criterio los elementos de Hardware, Software y sistema de comunicación que se empleará.

3.7.1.1 Requisitos funcionales y entorno.

3.7.1.1.1 Carga y transporte.

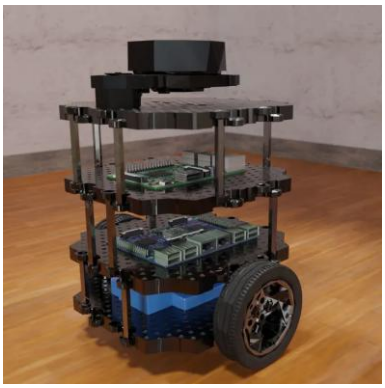
El AMR debe mover una carga con una masa promedio de 35 kg, con un factor de servicio que permita el uso prolongado del chasis, durante el transporte, el objeto no debe tocar el piso.

3.7.1.1.2 Entorno.

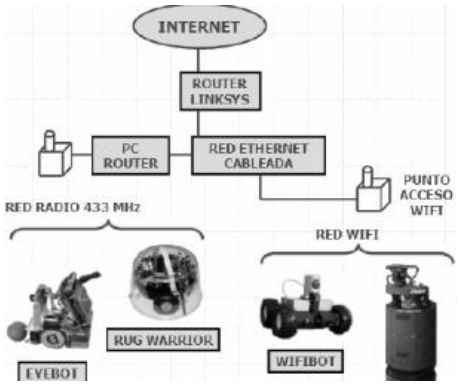
El área de prueba es un entorno abierto plano con algunas irregularidades e incidencia de luz solar, condición que puede afectar a los sensores ópticos.

3.7.1.2 Selección del sistema de arquitectura.

Para la elección del sistema de arquitectura se ha considerado la distribución de procesamiento de datos, donde existen dos tipos de sistemas, Sistema centralizado de cómputo, donde todos los sensores, unidades de procesamiento y actuadores están ubicados en el mismo equipo y sistema distribuido con nodo central, en el cual los procesos se dividen entre un servidor fijo en el que se realiza el procesamiento de datos y un dispositivo a bordo, el cual se encarga en la recepción de datos y control de actuadores, como se representa en la siguiente figura.



(a)



(b)

Figura 20. Arquitectura del sistema: (a). AMR con sistema a bordo, (b). AMR con sistema descentralizado con nodo central.

Para el prototipo se determinó inclinarse debido a una mejor viabilidad por un sistema descentralizado con nodo central, ya que permite aprovechar la capacidad de cómputo de la estación central disponible, reducir el hardware, mejorar el consumo energético del sistema a bordo y facilitar las tareas de desarrollo y mantenimiento, manteniendo al mismo tiempo la posibilidad de escalar el sistema a futuros robots móviles.

Tabla 8 Comparativa de arquitecturas de sistema.

Criterio	**Sistema a bordo**	**Sistema descentralizado con nodo central**
Dependencia de la red	Nula	Alta
Hardware a bordo	mayor potencia que microcontroladores.	Menor costo energético, respecto a un ordenador.
Seguridad del Hardware	Baja	Alta
Consumo de energía	Alto.	Bajo.
Escalabilidad	Coordinación de varios dispositivos.	Mayor escalabilidad.
Mantenimiento/ Reparación	Alto en caso de falla del sistema lógico.	Bajo.
Adaptación al proyecto	Viable, costoso y complejo en el robot.	Adecuado a los recursos disponibles.

3.7.1.3 Selección del Hardware

Los elementos considerados en este prototipo se seleccionaron en base a los criterios anteriormente indicados, a continuación, se analizan cada uno de los elementos empleados.

3.7.1.3.1 Ordenador a bordo.

Por concepto de necesidades del proyecto se determinará entre distintas plataformas de cómputo embebido y de control identificar la mejor relación entre capacidad de procesamiento, consumo de energía e integración con los sensores y actuadores del prototipo. Especialmente que permita la comunicación de la cámara RGB-D, IMU, controladores y actuadores. El dispositivo seleccionado será el encargado de adquirir los datos del entorno, transmitirlos al nodo central y aplicar las órdenes de control sobre el chasis del robot.

Tabla 9. comparación de Ordenadores a bordo.

Criterio	Ordenador a bordo	Jetson Nano	Raspberry Pi 5
Capacidad de procesamiento	Muy alta	Alta (CPU + GPU embebida)	Media–alta
Compatibilidad con sistemas	Multiplataforma	Linux, compatible con ROS 2	Linux y ROS 2
Consumo de energía	Alto	Medio–Bajo	Consumo medio
Integración mecánica a bordo	Voluminoso y pesado	Compacto, exige buena disipación	Compacto y fácil de integrar
Costo y disponibilidad	Costo alto, menor acceso	Costo alto, disponibilidad limitada	Costo intermedio, buena disponibilidad
Adecuación al prototipo	Sobredimensionado para este AMR	Más potente de lo necesario como nodo a bordo	Mejor equilibrio costo/rendimiento

Con respecto a la comparación realizada, aunque un computador industrial a bordo o una tarjeta embebida con GPU permitirían concentrar todo el procesamiento en el AMR, su costo, consumo de energía y complejidad de integración resultan poco favorables para el alcance del prototipo. En cambio, la Raspberry Pi 5 ofrece la capacidad suficiente para ejecutar sistemas operativos y comunicarse con la cámara RGB-D, la IMU y los controladores de los motores, manteniendo un consumo moderado y un costo accesible. Por estas razones se selecciona la Raspberry Pi 5 como ordenador de a bordo del AMR.

Figura 21. Placa de desarrollo Rapsberry pi 5.

3.7.1.3.2 Nodo Central.

Debido a la disponibilidad de los recursos, no existió una comparación en general para la elección del nodo central, sin embargo, se determinó unos requisitos mínimos para que cumpla su función.

Figura 20. Laptop nodo Central.

Equipo de portátil de uso general, empleado en este proyecto como nodo de procesamiento central, debido a sus mayores capacidades de procesamiento y desempeño, configurado para la ejecución de un entorno Linux de manera nativa, simulación y computo intensivo del Hardware móvil.

Tabla 10. Especificaciones del nodo Central.

Característica	**Especificación**	**Descripción**
Procesador	AMD Ryzen 7 5800H	CPU multinúcleo con alta capacidad de procesamiento
Memoria RAM	32 GB	Ejecución de algoritmos de navegación por visión artificial
Tarjeta gráfica	NVIDIA GeForce RTX	Soporte de GPU para cómputo intensivo y visión
Sistema operativo	Ubuntu 24.04 LTS	Sistema Nativo de ROS Jazzy
Rol dentro del sistema	Nodo central de procesamiento	Ejecuta navegación, mapeo y supervisión del AMR

3.7.1.3.3 Sensores.

Para la obtención de los datos necesarios para la navegación se determinaron dos sensores principales.

Para la detección de aceleraciones y giros que permitan la evaluación de la estabilidad del dispositivo se empleó una unidad de medición inercial.

Tabla 11. Características IMU LSM6DS3.

	Descripción
Tipo	Unidad inercial de 6 DoF (acelerómetro + giroscopio).
Magnitudes medidas	Aceleración lineal y angular en los ejes X, Y, Z.
Voltaje de alimentación	3,3 Vcc
Frecuencia de Muestreo	12,5Hz a 6 KHz
Interfaz	Comunicación I2C

Figura 22. Sensor IMU LSM6DS3.

La cámara RGB-D es el sensor principal que permite al prototipo tener la percepción del entorno, ya que nos brinda datos de color y profundidad, los cuales son fundamentales para la creación de mapas, reconocimiento del entorno y demás algoritmos empleados.

Tabla 12. Características cámara Intel Realsense D415.

Ítem	**Descripción**
Datos Obtenidos	Color y Profundidad
Interfaz de comunicación	USB 3.0 tipo C
resolución estándar	1280x720 pixeles a 30 FPS
Angulo de visión	69.4°
Rango de operación	0,3-10m
Software	SDK 2.0 de RealSense

Figura 23. Cámara Realsense D415.

3.7.1.3.4 Actuadores.

Los actuadores seleccionados se obtuvieron mediante la disponibilidad del mercado local, cumpliendo con los requisitos de tracción y manipulación de la carga.

Para el movimiento de las ruedas se decidió emplear motores de corriente continua sin escobillas (BLDC) que pueden ser controlados mediante Drivers y detectar su movimiento mediante sensores capacitivos Hall.

Tabla 13.Características de motores

Ítem	Descripción
Voltaje de alimentación	36 V
Potencia máxima	360 W
Diámetro	0.2 m
Tipo de alimentación	DC
Sensores de Control	Sensores de efecto Hall

Figura 24. Motores BLDC

Es necesario para el prototipo un actuador que permita levantar la carga, por ello se empleará un actuador lineal formado por un motor DC junto con un sistema reductor integrado para aumentar su fuerza y convertirlo a movimiento lineal.

Tabla 14. Características del Actuador Lineal.

Ítem	Descripción
Voltaje de alimentación	24 V
Carga máxima	100 kg
Certificado de protección	IP54
Extensión de cilindro	0.3 m
Sensores de Control	Finales de carrera integrados

Figura 25. Actuador Lineal

3.7.1.3.5 Sistema de energía y control.

El diseño del sistema de energía y control se desarrollaron con la idea de que ser seguros y estables a la potencia requerida, esta alimentará a los motores de las ruedas, actuador lineal y electrónica de control.

Para la base de la batería se emplean celdas de ion litio Samsung INR21700, que destacan por su alta densidad de energía y ser empleados en las baterías de vehículos eléctricos.

Tabla 15. Características de las celdas Samsung INR21700.

Ítem	Descripción
Voltaje nominal	3,6 V
Tipo de Celda	Ion litio
Capacidad	5 Ah
Amperaje máximo	18,5 amperios
Densidad de energía	259 Wh/kg
Formato de celda	21700

Figura 26. Celdas Samsung INR21700-50G

Para el uso seguro de las celdas es necesario un sistema de administración de baterías (BMS) que nos permite descargar y cargar las celdas, brindando protección a las mismas e indicando valores principales.

Tabla 16. Características del BMS.

Ítem	Descripción
Monitoreo	Aplicación
Protecciones	Cortocircuito, Sobrecargas, Descarga profundas, temperatura y sobre corrientes
Comunicación	Bluetooth
Configuración de celdas	2 celdas a 12 Celdas
Suministro constante de Corriente	20 amperios
Picos de corriente suministrado	60 amperios

Figura 27. BMS JBD.

Como existen varios elementos con distintos voltajes de trabajo, es necesario cambiar las tensiones a los valores de trabajo, es donde se emplea un módulo convertidor Buck nos permite variar el voltaje de salida con respecto al de entrada, logrando reducirlo y limitar la corriente que se emplea en la salida.

Tabla 17. Características del reductor de voltaje.

Ítem	Descripción
Modelo	XL4015 E1
Voltaje de entrada	8 – 36 V
Voltaje de salida	1.25 – 32 V
Protección	Cortocircuitos
Corriente de salida máxima	5 amperios

Figura 28.Reductor XL4015 E1

Para el control de los motores BLDC es necesario un circuito capaz de soportar los voltajes y potencias empleados, adicional de controlarlo mediante señales digitales enviadas desde el nodo a bordo del prototipo.

Tabla 18. Características del controlador ZS-X11H.

Ítem	Descripción
Control	PWM
Voltaje de entrada	12 – 60 V
Potencia máxima	500 w
Protección	Arranque suave por control de efecto hall
Corriente de salida máxima	10 amperios

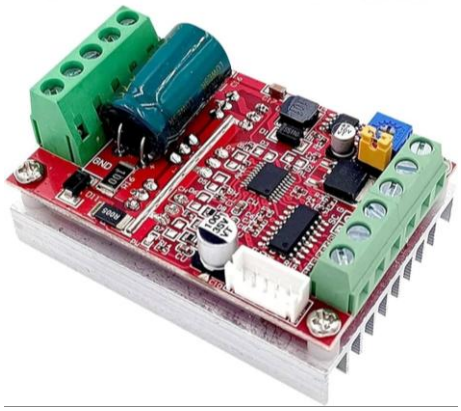


Figura 29. controlador ZS-X11H

El actuador lineal necesita un módulo de control que permita invertir la polaridad, el módulo controlador BTS7960 es capaz de realizar ese trabajo, además de contar con varias protecciones.

Tabla 19. Características del puente H BTS7960.

Ítem	Descripción
Control	Lógico, PWM
Voltaje de entrada	5 – 27 V
Corriente de operación continua máxima	20 amperios
Protección	Sobre voltajes, corrientes y temperatura
Corriente de salida máxima	43 amperios

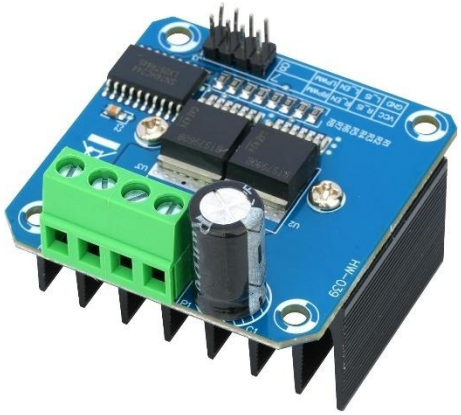


Figura 30. BTS7960

3.7.1.4 Selección del Software del sistema.

Debido a la necesidad de integración del distinto hardware y módulos se vio la solución de trabajar en framework que utilicen una comunicación Publicador-suscriptor, permitiendo así generar distintos nodos y que para su funcionamiento conjunto necesiten únicamente una entrada con un tipo de mensaje compatible, por ello se destacaron los Framework ROS 1 y ROS 2.

Tabla 20. Comparación entre Frameworks.

Criterio	ROS 1	ROS 2
Modo de comunicación	Nodo Central	Protocolo DDS
Algoritmos de navegación compatibles	Navegation Stack	Nav2, Rtab-map
Estado de distribución	Fin de vida útil	Soporte hasta 2029
Diseño de arquitectura	Pensado para un dispositivo	Pensado para desplegar nodos en distintos dispositivos
Compatibilidad con Proyecto	Viable, con distinta arquitectura	Compatible con la arquitectura empleada

Figura 31. ROS 2, Distribución Jazzy jalisco Framework

Finalmente, con los conocimientos adquiridos en el proceso de investigación, se logró determinar los algoritmos más compatibles con el sistema de distribución, hardware y software empleados.

3.7.2 Fase 2. Desarrollo del prototipo.

3.7.2.1 Diseño estructural y sistema de energía.

Para el diseño del prototipo del AMR se utilizó el programa de Autodesk Inventor, en el cual se realizó el modelado del bastidor, chasis, sistema de elevación y bosquejo del sistema eléctrico. Se definió que la estructura sea capaz de soportar sin problemas una masa de 100kg, para tener gran holgura al soportar la carga útil de 35kg, además de que se encuentre el peso cerca del centro de masa, para mantener rigidez y estabilidad adecuadas en el momento del transporte.

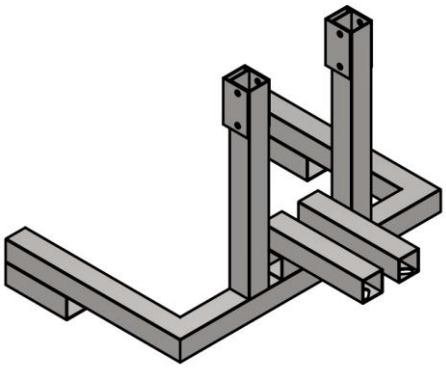


Figura 32. Diseño del bastidor.

El bastidor obtiene una forma inspirada en los elevadores de palets, utilizando tubo estructural cuadrado de 50x50x4mm de espesor, seleccionado por su resistencia mecánica y disponibilidad. Las dimensiones del bastidor son de 680x650x550mm de alto, ancho y altura, don una distancia de 650mm entre los ejes. Para la verificación de resistencia se empleó un análisis por elementos finitos en Inventor, aplicando una carga de 1000N sobre los soportes

con un resultado de desplazamiento máximo de 0.061mm, lo cual demuestra el comportamiento firme de la estructura.

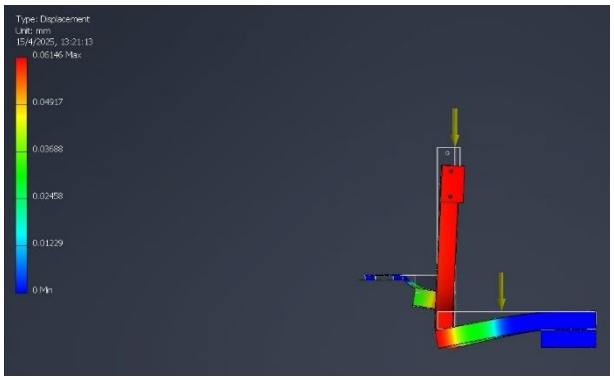


Figura 33. Análisis de elementos finitos en el bastidor.

Validado el bastidor se procede con el diseño y adición de los elementos motrices, siendo un actuador lineal de 1000N montado en la parte frontal del bastidor el cual permite levantar la carga para el momento de trasladarlo y el sistema de tracción en el cual se utilizan las ruedas con motores BLDC incorporados de 200mm de diámetro y 43 mm de ancho, con un recubrimiento de goma dura, en la parte trasera para cubrir el principio de estabilidad de 3 puntos se añadió una rueda caster neumática, con la ventaja de no importa la desigualdad del suelo, los 3 puntos de apoyo siempre harán contacto con la superficie.

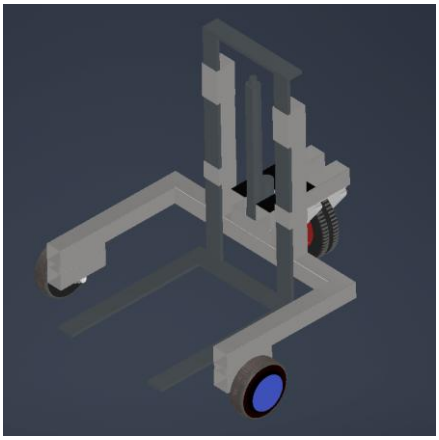


Figura 34. Chasis del AMR.

De esta forma se distribuye el peso para que el centro de gravedad permanezca bajo y cercano al eje del chasis, el conjunto de bastidor, sistema de elevación y ruedas tiene una masa de 26 kg, adicionalmente la fuente de energía le añade 2 kg más, resultando el peso total de 28kg del AMR sin carga.

Para el suministro de energía a los actuadores del AMR se diseñó una batería que emplea celdas de ion-litio 21700, en una configuración de 10 celdas en serie y 2 en paralelo (10S2P), con esto obteniendo un voltaje nominal de 36v, el cual varia del 100% a 42V y 0% a 28V, con una capacidad de 10 Ah y siendo posible suministrar 360w en una hora y hasta 720w continuos en media hora.

Figura 35.Celdas 21700 en configuración 10S2P.

En la creación de la batería se utilizaron porta celdas, tiras de níquel de 0.15mm de grosor y cinta térmica para su aislamiento, para el control de carga y descarga se empleó un BMS JBD, el cual controla y monitorea individualmente la energía en las celdas, temperatura y estado de carga mediante la aplicación, , brindado una potencia suficiente para la operación simultanea de los actuadores.

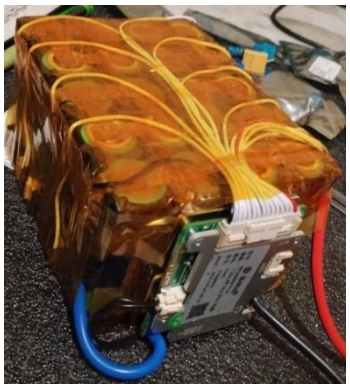


Figura 36. Paquete de celdas y BMS.

Como la batería fue diseñada para el sistema de potencia, su salida principal de 36V se distribuye en dos ramas, una directa a los controladores de las ruedas y la restante a un circuito de reducción de voltaje para el sistema de elevación, donde el actuador lineal utiliza 24V, En la siguiente figura se muestra la disposición de los circuitos empleados.

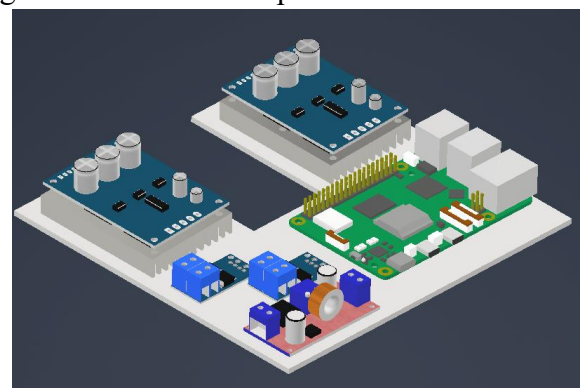


Figura 37. sistema de control montando en base.

Finalmente, con los sistemas mecánicos y eléctricos diseñados, se procede a la incorporación de todos estos en el AMR.

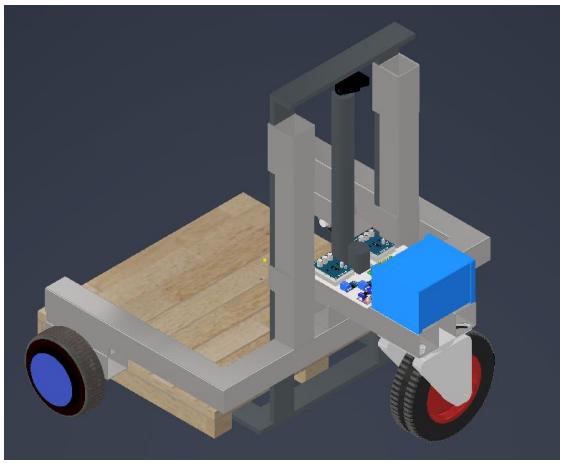


Figura 38. Diseño final de prototipo.

3.7.2.2 Arquitectura de software.

Como se mencionó en la fase anterior, la arquitectura se base en un esquema modular basado en el framework ROS 2, debido a su diseño dedicado a la escalabilidad e interoperabilidad de los subsistemas que conforman un AMR. Este sistema se ha organizado en cuatro fases de operación:

Tabla 21. Fases del Sistema.

Percepción	Encargado del proceso el nodo de RealSense, proporciona tópicos como imagen y profundidad.
Localización y mapeo	Se emplea el algoritmo RTAB-Map que realiza odometría visual y utiliza los datos para la creación de un mapa del entorno.
Planificación y acción	Utiliza el mapa generado para calcular trayectorias y evitar obstáculos.
Control	Nodo personalizado para la traducción de mensajes tipo Twist a pulsos PWM.

Cada fase está formada por distintos nodos que se comunican por el protocolo DDS y utilizando tópicos para el envío de distintos datos de los nodos que se ejecutan, en el siguiente diagrama se puede observar la interacción entre fases y sus nodos:

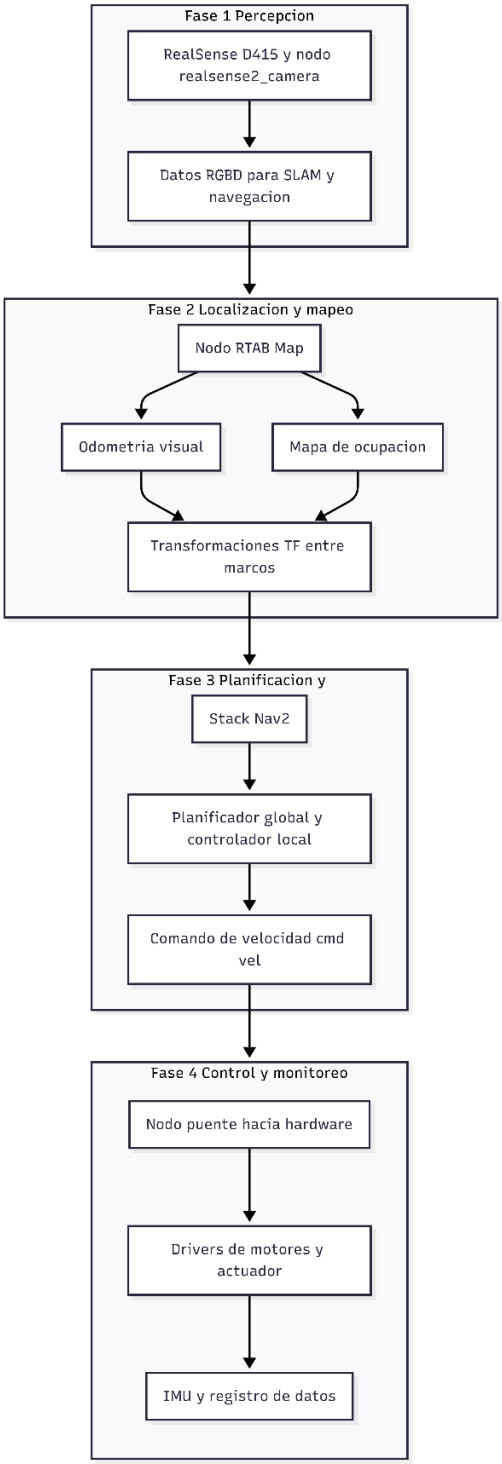


Figura 39. Fases del operación del Sistema del prototipo y flujo de datos.

La lógica empleada para el prototipo de robot autónomo se utilizó la siguiente lógica:

```
Entrada: Objetivo de Navegación (Coordenadas X, Y, Theta)
Salida: Movimiento físico del AMR hacia el destino
```

```
Inicio
    1.  Inicialización de Nodos:
        Iniciar Lanzador de Cámara (realsense2_camera)
        Iniciar SLAM (RTAB-Map) y cargar base de datos (.db)
        Iniciar Pila de Navegación (Nav2)
        Iniciar Nodo de Control (Puente_nodo) con privilegios de root (GPIO)

    2.  Bucle Principal (mientras Sistema Activo):
        a.  Adquisición de Datos:
            Leer fotogramas RGB-D y Nube de Puntos

        b.  Actualización de Estado (SLAM):
            Estimar odometría visual (VIO)
            Actualizar mapa de ocupación local y global

        c.  Planificación (Nav2):
            Si (Existe Nuevo Objetivo):
                Calcular ruta óptima libre de colisiones (A*)
            Si (Obstáculo detectado):
                Replanificar trayectoria local
            Generar velocidades requeridas (v, w) -> Tópico /cmd_vel

        d.  Ejecución de Control (Puente_nodo):
            Leer mensaje Twist (/cmd_vel)
            Aplicar cinemática inversa diferencial:
                Vel_Derecha = (v + (w * L) / 2)
                Vel_Izquierda = (v - (w * L) / 2)
            Mapear velocidades a Ciclo de Trabajo PWM (0-100%)
            Escribir en pines GPIO -> Drivers BTS7960/ZS-X11H


    Fin Bucle
Fin
```

3.7.2.3 Implementación de la Arquitectura de Software y Framework

Para la eliminación de latencias y la interacción directa entre el nodo central y el nodo a bordo, se empleó de manera nativa el sistema operativo de Ubuntu 24.04 LTS, compartiendo el mismo sistema el ordenador y la raspberry pi 5, para una comunicación mediante tópicos y nodos de ROS 2, se realizó la configuración de un dominio de ROS y así obtener comunicación entre los nodos dentro de la red, posterior a ello, se instalan las librerías necesarias para el uso del hardware y se procede a validar resultados los algoritmos, esto se lo realiza visualizando los tópicos de salida del nodo objetivo si se encuentran activo o abriendo el mensaje que este debe publicar.

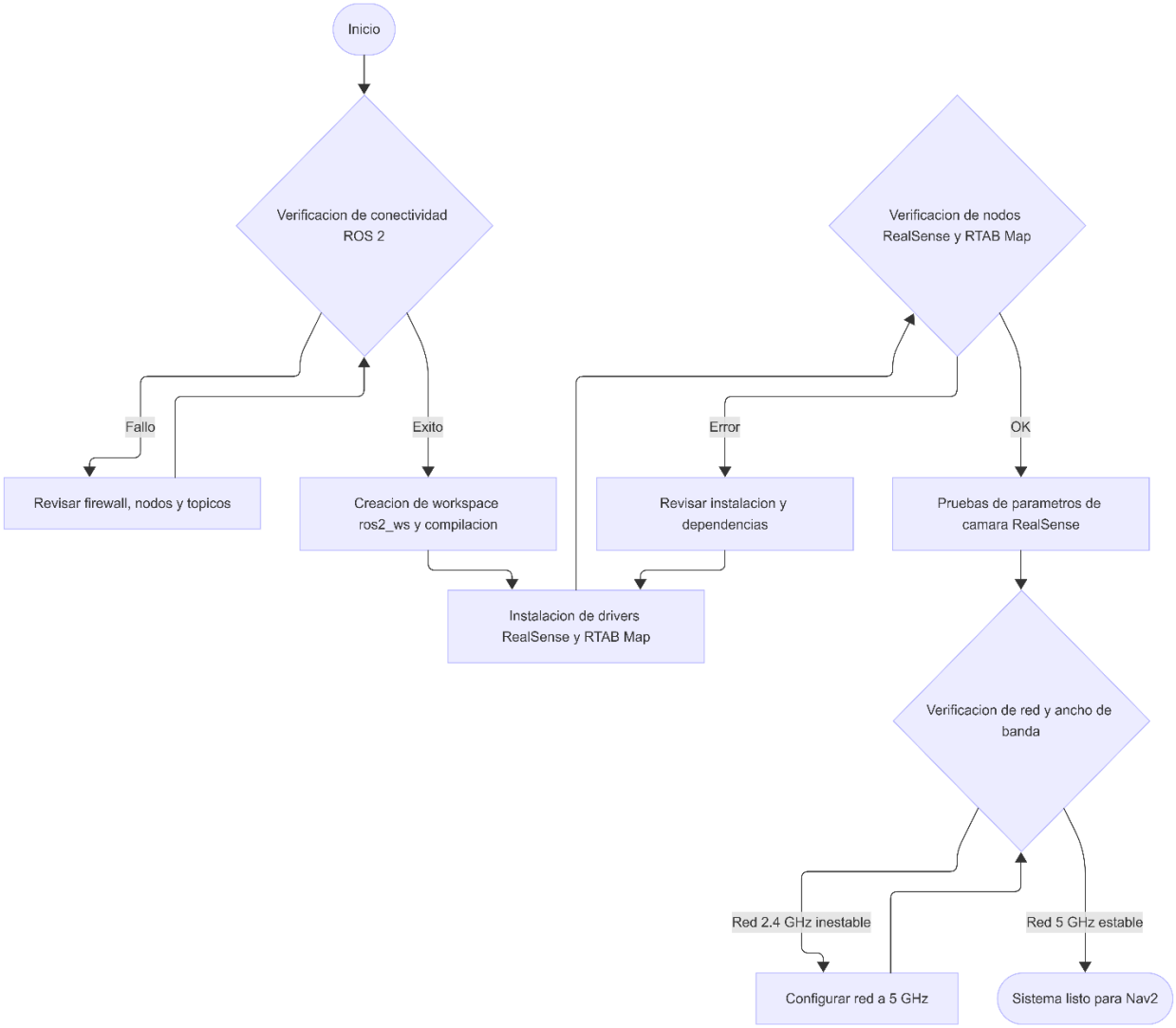


Figura 40. Diagrama de flujo del proceso de puesta en marcha del sistema de control del prototipo.

Los resultados obtenidos en cada fase del software se podrán observar en la siguiente fase del documento.

3.7.3 Fase 3. Implementación y evaluación.

3.7.3.1 Diseño final del prototipo.

Finalizado el proceso de diseño del bastidor, se realizó la integración completa de los sistemas que conforman el prototipo del AMR, donde puede observarse el sistema de elevación y tracción, el sistema de control colocado en el bastidor y la batería Customizada. Este diseño mantiene el centro de masa bajo y cerca de las ruedas motrices, manteniendo una mejor estabilidad.

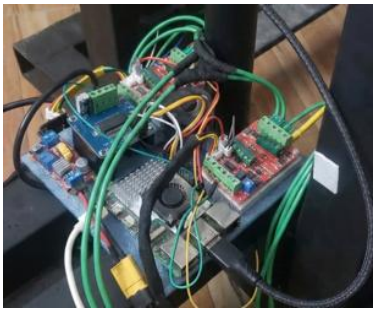


Figura 41. integración de sistemas en el prototipo.

La batería se hace puede apreciar en una caja personalizada realizada con impresión 3d para la protección de las celdas de ion litio y su BMS y siendo capaz de alimentar a los elementos actuadores y módulos de control.

Tabla 22. Especificaciones de la batería del prototipo.

Característica	**Valor**
Configuración	10S2P
Tensión nominal	36 V
Capacidad nominal	10 Ah
Energía aproximada	360 Wh
Tipo de celdas	Celdas cilíndricas 21700 de ion-litio
Sistema de protección	BMS para 10S con protección por tensión y corriente

Figura 42. Batería 10S2P del AMR.

La parte de control fue colocada encima del bastidor, dejándolo en una posición donde sus componentes puedan ser de fácil acceso para la modificación o ajuste necesario.

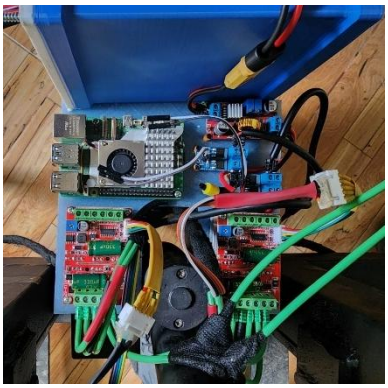


Figura 43. Sistema de control Montado en el Bastidor

Finalmente, el resultado de la implementación del todos los sistemas para la creación del primer prototipo, dio como resultado una estructura resistente, con su sistema de control incorporado y funcional, listo para realizar las pruebas pertinentes en el entorno físico, este prototipo tiene las siguientes especificaciones generales:

Tabla 23. Especificaciones del AMR.

Parámetro	**Valor.**
Carga útil	35kg (hasta 90kg)
Dimensiones generales (L × W × H)	680 × 650 × 550 mm
Material del bastidor	Acero estructural negro, 50 × 50 × 4 mm
Distancia entre ejes	650 mm
Actuador lineal	305 mm de carrera – 1000 N
Desplazamiento máximo (FEA)	0.061 mm
Ruedas motrices	Ø 200 mm × 53 mm
Peso del bastidor y sistema	26 kg
Peso del sistema energético	2 kg
Voltaje nominal del sistema	36 V DC
Capacidad de la batería	10 Ah (360 Wh)
Consumo en operación	136 W
Tiempo de trabajo	158 min
Corriente máxima continua	20 amperios
Pico de corriente	40 amperios

Figura 44. Resultado de la implementación.

3.7.3.2 Resultado de implementación del Software.

Después de realizar las configuraciones indicadas en la parte de implementación se ha llegado a los resultados de las 4 fases indicadas en la arquitectura de software.
En la fase de percepción mediante la implementación de la cámara realsense y los algoritmos utilizados para su funcionamiento, se pudo obtener distintos perfiles de la cámara, donde se modifica su resolución y fotogramas por segundo. Se pudo observar adicionalmente el consumo de banda ancha que necesitan los datos de estos nodos para la transmisión por la red LAN como se indica en la siguiente tabla:

Tabla 24. Comparación perfiles de Cámara RealSense.

Perfil de video	Latencia(s)	Ancho de banda usada (MB)	FPS	Observación
1280x720x30	2	150	3.6	Descartado
640x480x15	0.5	110	9	Estable
640x360x30	0.31-2	130	20	Inestable
640x360x15	0.7	80	15	Alta latencia
424x240x15	0.5	35	15	Excelente
424x240x60	0.3	135	25	Prometedor

Se Determinó que el ancho de banda necesario es alto para el envío de datos, por lo cual se buscó formas de compresión de la información, en el siguiente figura se puede observar distintos nodos disponibles para la toma de datos, donde se empleará los tópicos comprimidos y mejorar la latencia.

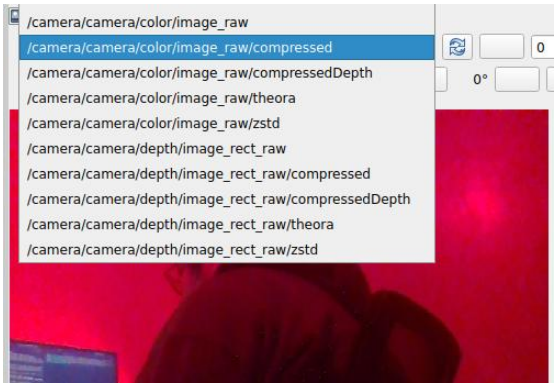


Figura 45. Visualizador rqt y nodos disponibles.

Figura 46. Imagen de profundidad en escala de grises (oscuro más cerca)

Como resultado se llegó a una configuración del algoritmo de Realsense estable en el área de pruebas que se describe en la siguiente tabla.

Tabla 25. Perfil de cámara Realsense determinada.

Parámetro	**Valor**
Resolución	424x240x15
Archivo	Comprimidos
Ancho de banda empleado	35 MB/s
Latencia de información	0,2 s

Para la ejecución del algoritmo RTAB-Map es necesario tomar los datos de imagen, profundidad y nueve de puntos de la fase de percepción, ya que son primordiales para que realice la comparación de keyframes para la realización de odometría visual.

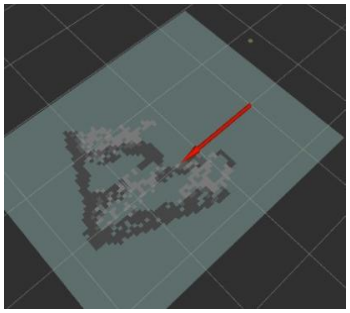


Figura 47. Inicio de RTAB-Map.

Al momento de empezar el algoritmo, busca puntos de referencia para realizar el mapa de ocupación del entorno, posteriormente se puede mover la cámara en el área para el escaneo del entorno y genere el mapa en 3D y 2D, es necesario mover la cámara de manera estable para que logre una superposición de las imágenes y logre identificar puntos comunes entre los frames.

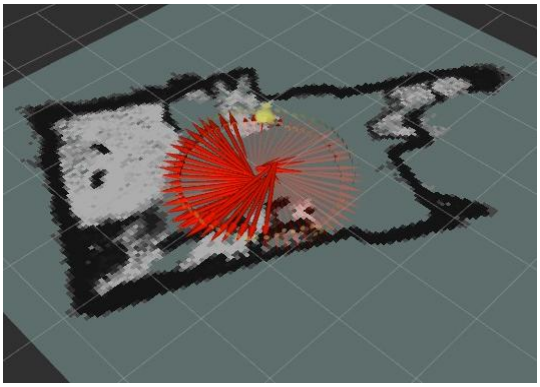


Figura 48. Rtab-Map en el Proceso de escaneo del entorno

Al finalizar el escaneo, se obtiene un archivo en formato base de datos, en el que se puede observar los siguientes resultados:

Tabla 26. Resultados de RTAB-Map.

Resultado	**Uso**
Mapa de ocupación	Limita el área donde puede moverse el prototipo
Nube de puntos	Empleado en el racionamiento del entorno y determinar su posición
Odometría	Brinda posición del prototipo en el área

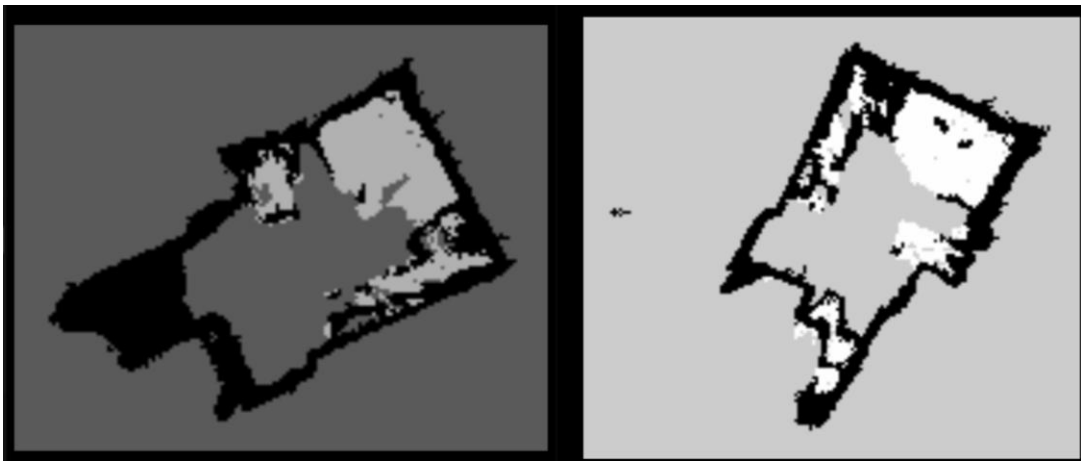


Figura 49. Mapa de ocupación 2D.

Al tener los datos brindados por la fase de perspectiva, localización y mapeo, se tiene los datos necesarios para correr el algoritmo que controla el movimiento del prototipo en el entorno, Nav2 utiliza los datos del mapa de ocupación, odometría y nube de puntos para la creación de un mapa de costos, para determinar el mejor camino a la meta planteada, además de un mapa de costos local que se utiliza para la evasión de obstáculos cercanos. Como resultado Nav2 puede realizar las siguientes funciones.

Tabla 27. Funciones de Nav2

Resultado	**Uso**
Mapa de costos	Mapa realizado en base del mejor camino, más corto, con menos obstáculos, más eficiente
Mapa de costos local	Empleado la evasión de obstáculos cercanos
Planificador	Encargado en crear la trayectoria que se recorrerá
Actuador	Publica datos de velocidad para mover los motores y el prototipo recorra la trayectoria

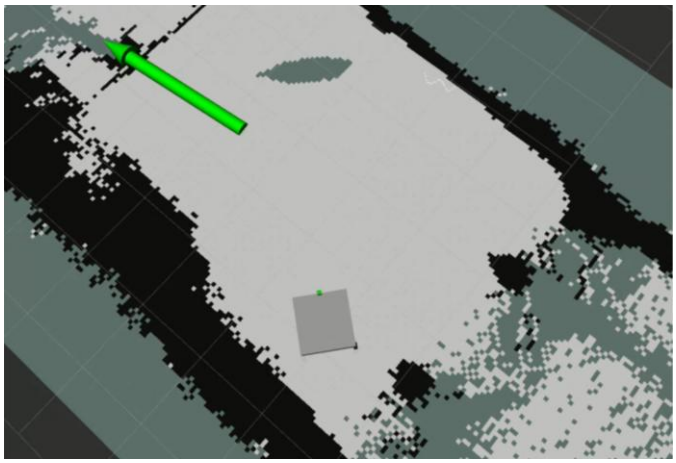


Figura 50. Movimiento del Prototipo en escenario real de pruebas y sincronizado con la representación virtual.

En la fase de control, se realizó la creación de un nodo traductor de datos, donde ingresan valores delo nodo /Cmd_vel y se los transforma a pulsos PWM calculado mediante velocidades de la rueda obtenida en distintos valores de trabajo del pulso y regresión lineal, obteniendo un pseudocódigo como el siguiente:

```
Entrada: Tópico /cmd_vel (Geometry_msgs/Twist)
Salida: Señales PWM y Dirección (GPIO)
Inicio
    1.  Inicialización:
        Cargar parámetros (radio_rueda, ancho_vía, limites_accel, pines_GPIO)
```

```
        Configurar GPIO en modo BCM
        Inicializar PWM a 1000 Hz con ciclo de trabajo 0%
        Iniciar temporizador de control (100 Hz)

    2.  Callback de Suscripción (/cmd_vel):
        Al recibir mensaje Twist:
            Saturar v_cmd entre [-max_linear, max_linear]
            Saturar w_cmd entre [-max_angular, max_angular]
            Actualizar last_cmd_time = tiempo_actual

    3.  Bucle de Control (Ejecución periódica):
        a. Watchdog de Seguridad:
           Si (tiempo_actual - last_cmd_time > 0.35s):
               Detener robot (v_cmd = 0, w_cmd = 0)

        b. Suavizado de Movimiento (Rampas):
           v_set = CalcularRampa(v_actual, v_cmd, accel_limit)
           w_set = CalcularRampa(w_actual, w_cmd, ang_accel_limit)

        c. Cinemática Diferencial Inversa:
           v_izq = v_set - (w_set * ancho_vía / 2)
           v_der = v_set + (w_set * ancho_vía / 2)

        d. Generación de PWM (Función speed_to_pwm):
           Para cada rueda:
               Calcular fracción de velocidad (v_rueda / v_max_rueda)
               Determinar dirección (HIGH/LOW) según signo e inversión
               Mapear magnitud a Duty Cycle [pwm_min_duty, 100%]
               Si (magnitud < deadband): Duty Cycle = 0
        e. Escritura en Hardware:
           Actualizar pines de Dirección y PWM
    4.  Cierre Seguro (Signal Handler):
        Al finalizar: Detener PWM y limpiar configuración GPIO
Fin
```

Con este nodo personalizado, se toma los valores del publicador de velocidad y se los transformar a PWM, con algunos parámetros de seguridad y configuración respecto a las dimensiones de las ruedas, para limitar su velocidad, aceleración y comportamiento en casos de errores.

3.7.3.3 Evaluación.

Las pruebas de navegación autónoma se realizan en el área de prueba que contiene características de la zona donde se implementa en la realidad, piso con rugosidad, presencia de obstáculos cercanos al recorrido y una carga que manipula el operario. Sobre esta área se genera previamente un mapa mediante RTAB-Map, utilizando la cámara RGB-D montada en el AMR.

Figura 51. Escenario de pruebas.

Para el análisis del prototipo se optó por enviar diferentes posiciones objetivas dentro de la zona de pruebas, variando la distancia y la orientación final del robot. Cada posición objetivo se envía como una orden de navegación autónoma.

En cada trayectoria se registran de forma automática la distancia recorrida por el AMR, el tiempo total de navegación, la velocidad media de desplazamiento y las aceleraciones en el eje vertical (z) medidas por la IMU instalada sobre la estructura que soporta la carga, además se registra el resultado de la navegación reportado por Nav2 (SUCCEEDED o ABORTED).

3.7.3.2 Configuraciones del algoritmo de navegación basado en Nav2

La variable independiente definida como tipo de algoritmo en esta fase serán las diferentes configuraciones del algoritmo de navegación basado en Nav2.
Se plantean perfiles de parametrización que modifican principalmente las velocidades máximas lineales y angulares y las aceleraciones máximas (rampas) aplicadas al AMR:

Tabla 28. Perfiles de Nav2 a Analizar

Tipo de algoritmo	Descripción
Perfil conservador (Nav2–C)	Valores de velocidad y aceleración reducidas, con el objetivo de minimizar vibraciones en el eje z y maximizar la integridad de la carga.
Perfil equilibrado (Nav2–E)	Rango de velocidades y rampas de aceleración expandidas, donde se busca la optimización del tiempo de entrega y vibraciones en Z

Estas variaciones se implementan ajustando los parámetros de Nav2 que se ejecuta en la Raspberry Pi. De esta forma, todo el sistema de navegación y control se mantiene basado en Nav2, pero se evalúan distintos niveles de desempeño y exigencia dinámica.

3.7.3.3 Variables e instrumentos de medición

Como se indicó en la sección 3.3, se indicó brevemente la forma en que se recolectaron los datos, en esta sección será más específica de las pruebas que se realizarán y los datos que se evaluarán. En cada recorrido se registran las variables de desempeño definidas en el perfil de tesis, utilizando los sensores e instrumentos disponibles en el prototipo:

Tiempo de entrega [s]:

Periodo desde el momento en que se envía la orden de navegación autónoma hasta la meta, en el que el algoritmo Nav2 indica su finalización del intento. En las pruebas obtenidas que concluyen de manera efectiva se lo determina en un estado de Exitoso, mientras que en pruebas donde no llega a la meta se las marca como Abortadas.

Velocidad de desplazamiento [m/s]:

Velocidad promedio del prototipo en el trayecto, calculada a partir de los datos de odometría publicados en ROS 2. La distancia recorrida se obtiene integrando la odometría a lo largo del trayecto hacia la posición objetivo.
De esta forma determinando la velocidad media real en las trayectorias realizadas por el AMR.

Estabilidad del transporte (aceleraciones en eje Z) [m/s²]:

Mediante un sensor IMU en la estructura que soporta la carga se registran las aceleraciones en el eje Z, relacionadas a vibraciones generadas en el trayecto. Los datos se adquieren a una frecuencia de sesenta veces por segundo y se envían al sistema de registro mediante microcontrolador y comunicación inalámbrica, obteniendo así aceleraciones máximas en z y aceleraciones promedio durante el trayecto.

Exactitud en la posición de entrega [Error]:

Porcentaje de error de la distancia al llegar al destino con respecto a la trayectoria realizada entre la posición inicial y la posición real alcanzada por el AMR al finalizar la maniobra de navegación. Esta magnitud puede estimarse mediante la información de localización proporcionada por RTAB-Map

Todos los datos de sensores (IMU, odometría, tiempos de ROS 2) se almacenan en archivos de registro para su análisis posterior en el Capítulo IV, donde se presentarán tablas, gráficos y comparaciones cuantitativas entre configuraciones.

CAPÍTULO IV. RESULTADOS Y DISCUSIÓN

Para el siguiente análisis de resultados se asignó un valor la variable dependiente, siendo para el algoritmo de Nav2-Equilbrado 0 y el algoritmo Nav2-Conservador 1

4.1 Prueba de normalidad.

Se realizó la prueba de normalidad para concluir si los datos obtenidos en las pruebas realizadas tienden a una formar una distribución estándar para determinar el tipo de modelo estadístico a aplicar.

H_0: Los datos recolectados tienen una distribución normal
H_1: Los datos recolectados no tienen una distribución normal

Como se observa en la tabla de prueba de normalidad, la mayoría de los datos evaluados ratifican el uso de pruebas no paramétricas (p-value < 0.05), se rechaza la hipótesis nula y se acepta la hipótesis alternativa, para nuestro estudio se usará la prueba de Mann-Whitney.

Tabla 29. Valores resultantes de la prueba de normalidad.

	P-value	
	Equilibrado	Conservador
Tiempo de entrega	<0.001	<0.001
Velocidad del dispositivo	0.004	<0.001
Exactitud de la entrega	<0.001	<0.001
Estabilidad del dispositivo	0.039	0.314

4.2 Prueba de Mann-Whitney para muestras de tiempo de entrega

H_0: La distribución del tiempo de entrega es la misma entre los tipos de algoritmo
H_1: La distribución del tiempo de entrega no es la misma entre los tipos de algoritmo

Como se muestra en la tabla 30, dado que el p-values es menor a 0.05 se rechaza la hipótesis nula, por lo que se concluye que la distribución del tiempo de entrega no es la misma entre los tipos de algoritmo.

Tabla 30. Valores resultantes de la prueba Mann-Whitney para tiempos de Entrega

Tiempo de entrega	p-value
Equilibrado	0.004

Conservador

En la Figura 52 se puede observar diagramas de cajas que señalan que, el perfil 0 se desplazada hacia valores de tiempo menores, indicando que este algoritmo completa las trayectorias en menos tiempo que el algoritmo 1. Además, la altura de la caja del perfil 1 es menor, por lo que sus tiempos de entrega son más homogéneos, mientras que el perfil Conservador presenta una caja más alta y un recorrido de bigotes mayor, reflejando una mayor dispersión de los tiempos.

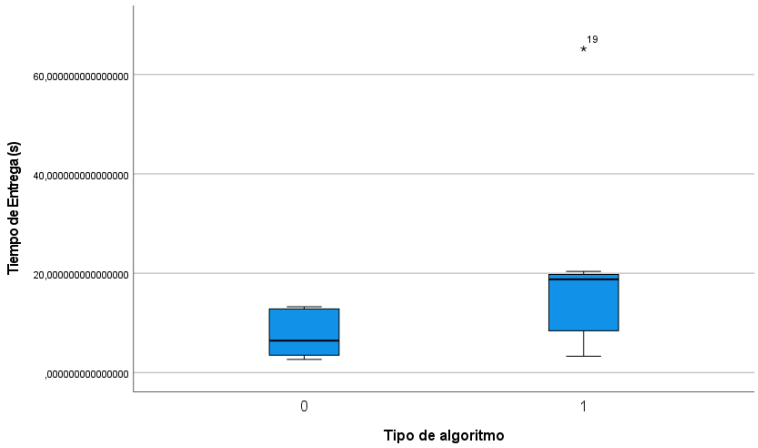


Figura 52. Diagrama de cajas del tiempo de entrega según el tipo de algoritmo.

4.3 Prueba de Mann-Whitney para muestras de velocidad del dispositivo.

H_0: La distribución de la velocidad del dispositivo es la misma entre los tipos de algoritmo
H_1: La distribución de la velocidad del dispositivo no es la misma entre los tipos de algoritmo

La tabla 31 señala que el p-values es menor a 0.05 se rechaza la hipótesis nula, por lo que se concluye que la distribución de la velocidad del dispositivo no es la misma entre los tipos de algoritmo.

Tabla 31. Valores resultantes de la prueba Mann-Whitney para Velocidad del Dispositivo.

Velocidad del dispositivo	p-value
Equilibrado	<0.001
Conservador	

En la Figura 53 se aprecia que el perfil 0 alcanza velocidades medias mayores que el perfil Conservador, ya que su caja se encuentra desplazada hacia valores superiores en el eje

vertical, el rango del perfil 1 se concentra en valores más bajos, indicando velocidades menores, siendo coherente con los resultados de la prueba.

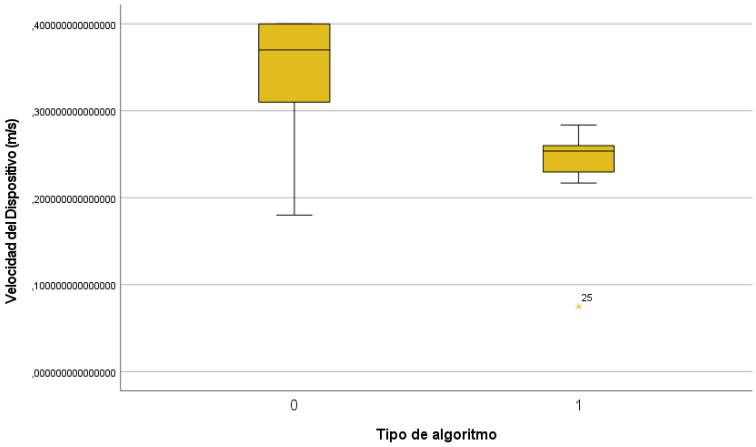


Figura 53. Diagrama de cajas de velocidad del dispositivo según el tipo de algoritmo.

4.4 Prueba de Mann-Whitney para muestras de exactitud de la entrega

H_0: La distribución de la exactitud del dispositivo es la misma entre los tipos de algoritmo
H_1: La distribución de la exactitud del dispositivo no es la misma entre los tipos de algoritmo

Dado que el p-values es mayor a 0.05 se acepta la hipótesis nula, por lo que se concluye que la distribución de la exactitud del dispositivo es la misma entre los tipos de algoritmo, mostrados en la siguiente tabla.

Tabla 32. Valores resultantes de la prueba Mann-Whitney de exactitud de entrega.

Exactitud de la entrega	p-value
Equilibrado	0.252
Conservador	

En la Figura 54 las cajas de los algoritmos 0 y 1 son similares y sus medianas son muy cercanas, mostrando que el error de entrega es similar en ambos casos, no se observa una variación entre algoritmos, por lo que ningún perfil muestra ventaja evidente en la exactitud de la entrega. Coincidiendo con el p-value (0,252) no muestra diferencias significativas entre los algoritmos.

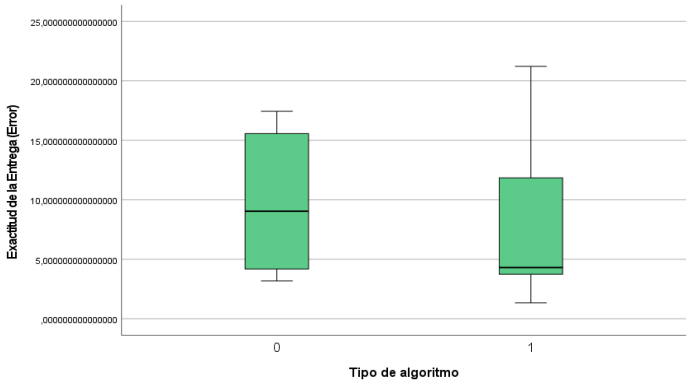


Figura 54. Diagrama de cajas de exactitud de entrega según el tipo de algoritmo.

4.5 Prueba de Mann-Whitney para muestras de estabilidad del dispositivo.

H_0: La distribución de la estabilidad del dispositivo es la misma entre los tipos de algoritmo
H_1: La distribución de la estabilidad del dispositivo no es la misma entre los tipos de algoritmo

Dado que el p-values es mayor a 0.05 se acepta la hipótesis nula, por lo que se concluye que la distribución de la estabilidad del dispositivo es la misma entre los tipos de algoritmo.

Tabla 33. Valores resultantes de la prueba Mann-Whitney de estabilidad del dispositivo.

Estabilidad del dispositivo	p-value
Equilibrado	0.940
Conservador	

En la Figura 55 los diagramas de cajas de estabilidad del dispositivo para los tipo de algoritmos presentan medianas y rangos similares, sugiriendo que el tipo de algoritmo no afecta en la estabilidad del dispositivo. Este resultado es consistente con la prueba de Mann-Whitney, que reporta un p-value de 0,940 y confirma la casi nula existencia de diferencia

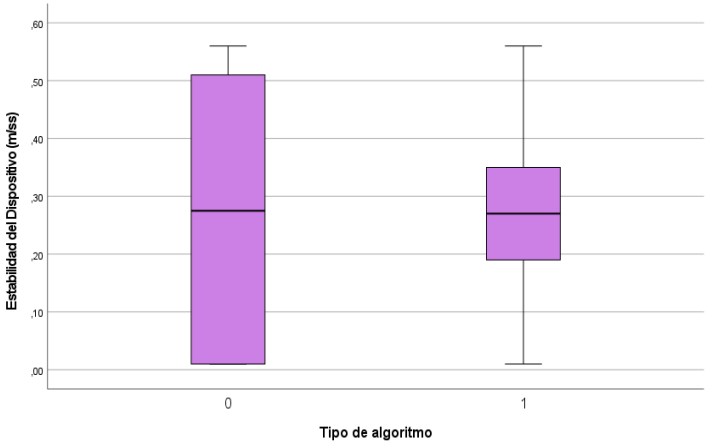


Figura 55. Diagrama de cajas de estabilidad del dispositivo según el tipo de algoritmo.

CAPÍTULO V. CONCLUSIONES y RECOMENDACIONES

5.1 Conclusiones

En base al desarrollo experimental y al análisis de los resultados obtenidos, se presentan las siguientes conclusiones alineadas con los objetivos de la investigación:

- Respecto al objetivo de investigar y seleccionar algoritmos de visión artificial, se determinó que la dependencia exclusiva del sensor RGB-D Intel RealSense D415 y de algoritmos de odometría visual y SLAM (RTAB-Map) presenta restricciones operativas críticas en entornos industriales con iluminación variable o baja textura visual. Aunque el prototipo logró completar alrededor del 71 % de las trayectorias de prueba, el resto registraron fallos de localización por pérdida de referencias visuales. Esto permite concluir que, si bien la cámara es efectiva para la evasión de obstáculos y el mapeo en condiciones controladas, no brinda por sí sola la robustez necesaria para una operación autónoma continua, por lo que se recomienda integrar algoritmos de fusión sensorial con fuentes de información complementarias.

- En relación con el diseño e implementación del prototipo de vehículo autónomo de carga, el desarrollo del nodo de interfaz personalizado y de la arquitectura de control validó la factibilidad de transportar bloques de concreto fresco sin comprometer su integridad física. En la totalidad de las trayectorias exitosas, la implementación de rampas de aceleración por software permitió limitar las fuerzas inerciales, manteniendo la aceleración del chasis por debajo del umbral crítico de 20 m/s^2. Esto demuestra que es posible prescindir de sistemas de amortiguación mecánica adicionales cuando se aplica un control cinemático riguroso que gestione la suavidad de los movimientos de arranque, cambio de velocidad y frenado.

- Al comparar los perfiles de navegación conservador y equilibrado, los resultados de las pruebas muestran que la elección del algoritmo influye directamente en las variables dependientes de velocidad y tiempos de entrega. El perfil equilibrado presenta tiempos de entrega significativamente menores y una mayor velocidad media que el perfil conservador, sin embargo, para los valores dependientes de exactitud de la entrega no se observan diferencias estadísticamente significativas entre ambos algoritmos ($p = 0,252$), lo que indica que la precisión no se afecta por la parte del algoritmo y en valores de aceleraciones en el eje vertical no existe variación.

- En general, la investigación y evaluación del uso de un robot de carga autónomo con tecnologías IoT y visión artificial, validan el potencial del prototipo como una solución de automatización para empresas MIPYMES. La arquitectura de software basada en ROS 2 y la implementación de un esquema IoT con nodo central de supervisión y nodo abordo (Raspberry Pi 5) resultaron funcionales para la gestión y monitoreo de tareas de logística interna, permitiendo registrar datos de operación en

tiempo real. Sin embargo, se concluye que el prototipo se encuentra en una fase de validación tecnológica, por lo que para su despliegue comercial o industrial pleno es imprescindible superar las limitaciones de percepción identificadas mediante la adición de sensores adicionales y estrategias de navegación más robustas a las condiciones cambiantes del entorno.

5.2 Recomendaciones.

Con el fin de incrementar la fiabilidad del sistema y evolucionar el prototipo hacia un producto industrial, se sugieren las siguientes líneas de trabajo futuro:

- Se recomienda la integración de un sensor LiDAR 2D o 3D al esquema de navegación actual. La fusión de los datos geométricos del láser (que ofrecen alta precisión y rango de 360°) con la información visual de la cámara permitiría resolver los problemas de "pérdida de localización" en pasillos monótonos o con luz solar directa, elevando la tasa de éxito de las trayectorias hacia estándares industriales (>95%).
- se recomienda evaluar una arquitectura de localización global asistida. Esto podría implementarse mediante el uso de cámaras cenitales que transmitan la posición absoluta al robot, proporcionando una "posición absoluta" que corrija la deriva acumulada de la odometría.
- Se sugiere la incorporación de encoders de mayor fiabilidad en las ruedas motrices y la implementación de un Filtro de Kalman Extendido (EKF) para mejorar la estimación del desplazamiento físico del robot reduciendo la dependencia de los algoritmos visuales y permitiría al AMR navegar distancias con mayor precisión en caso de fallo temporal de los sensores.

BIBLIOGRAFÍA

[1] Subgerencia de Análisis de Productos y Servicios, «Ficha Sectorial – Construcción». CFN B.P., febrero de 2024. [En línea]. Disponible en: https://www.cfn.fin.ec/wp-content/uploads/2024/05/Ficha-Sectorial-Construccion.pdf

[2] V. Yepes, «Vibrado del hormigón», El blog de Víctor Yepes. [En línea]. Disponible en: https://victoryepes.blogs.upv.es/tag/vibrado-del-hormigon/

[3] «Hazard Analysis - Lifting and Carrying (Manual materials Handling», The Center for Construction Research and Training, Silver Spring, MD, Technical Report, 2023. [En línea]. Disponible en: https://www.cpwrconstructionsolutions.org/masonry/hazard/63/load-unload-and-distribute-construction-materials-lifting-and-carrying-manual-materials-handling.html

[4] MESH Automation, «How Much Does an AGV Cost in 2025? Price by Type & Features». MESH Automation, 2024. [En línea]. Disponible en: https://meshautomationinc.com/agv-cost-2025/

[5] F. M. T. P. de L. Martín y Arturo H. Q. Alonso, Learning ROS 2: A beginner's guide to programming robots, 2nd ed. Birmingham, UK: Packt Publishing, 2023.

[6] P. F. C. Allaico y D. A. R. Naranjo, «CONSTRUCCIÓN DE UN PROTOTIPO DE SISTEMA ROBÓTICO PARA EL DESPACHO DE PRODUCTOS FARMACÉUTICOS DE BAJO PESO UTILIZANDO VISIÓN ARTIFICIAL Y COMUNICACIÓN INALÁMBRICA», ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO, Riobamba, 2022.

[7] A. Marquinez y Z. Guerrero, «DESARROLLO DE ALGORITMOS DE APRENDIZAJE PROFUNDO BASADOS EN VISIÓN ARTIFICIAL PARA VEHÍCULOS AUTÓNOMOS DE INTERIORES», Tesis de Máster, Universidad del Pais Vasco, Bilbao, 2022.

[8] O. GERALDO, «SISTEMA DE NAVEGACIO´ N BASADA EN VISIÓN ARTIFICIAL PARA UN ROBOT MÓVIL», INSTITUTO TECNOLÓGICO DE LA PAZ DIVISI´ON DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN MAESTRÍA EN SISTEMAS COMPUTACIONALES, LA PAZ, 2022.

[9] N. D. B. Vacca y A. J. B. Rangel, «DISEÑO DE UN SISTEMA AUTOMÁTICO DE EVASIÓN DE OBSTÁCULOS BASADO EN VISIÓN ARTIFICIAL PARA EL ROBOT COLABORATIVO UR3», 2022.

[10] NTE INEN 3066: Bloques de hormigón. Requisitos y métodos de ensayo, : NTE INEN 3066, Quito - Ecuador., noviembre de 2016.

[11] E. Jiva, «Desarrollo de la teleoperación de robots industriales y colaborativos mediante técnicas avanzadas de visión artificial», Tesis de Máster, Universitat Politècnica de València, Valencia, 2019. [En línea]. Disponible en: https://riunet.upv.es/handle/10251/130010

[12] A. Loganathan y N. S. Ahmad, «A systematic review on recent advances in autonomous mobile robot navigation», Eng. Sci. Technol. Int. J., vol. 40, p. 101343, abr. 2023, doi: 10.1016/j.jestch.2023.101343.

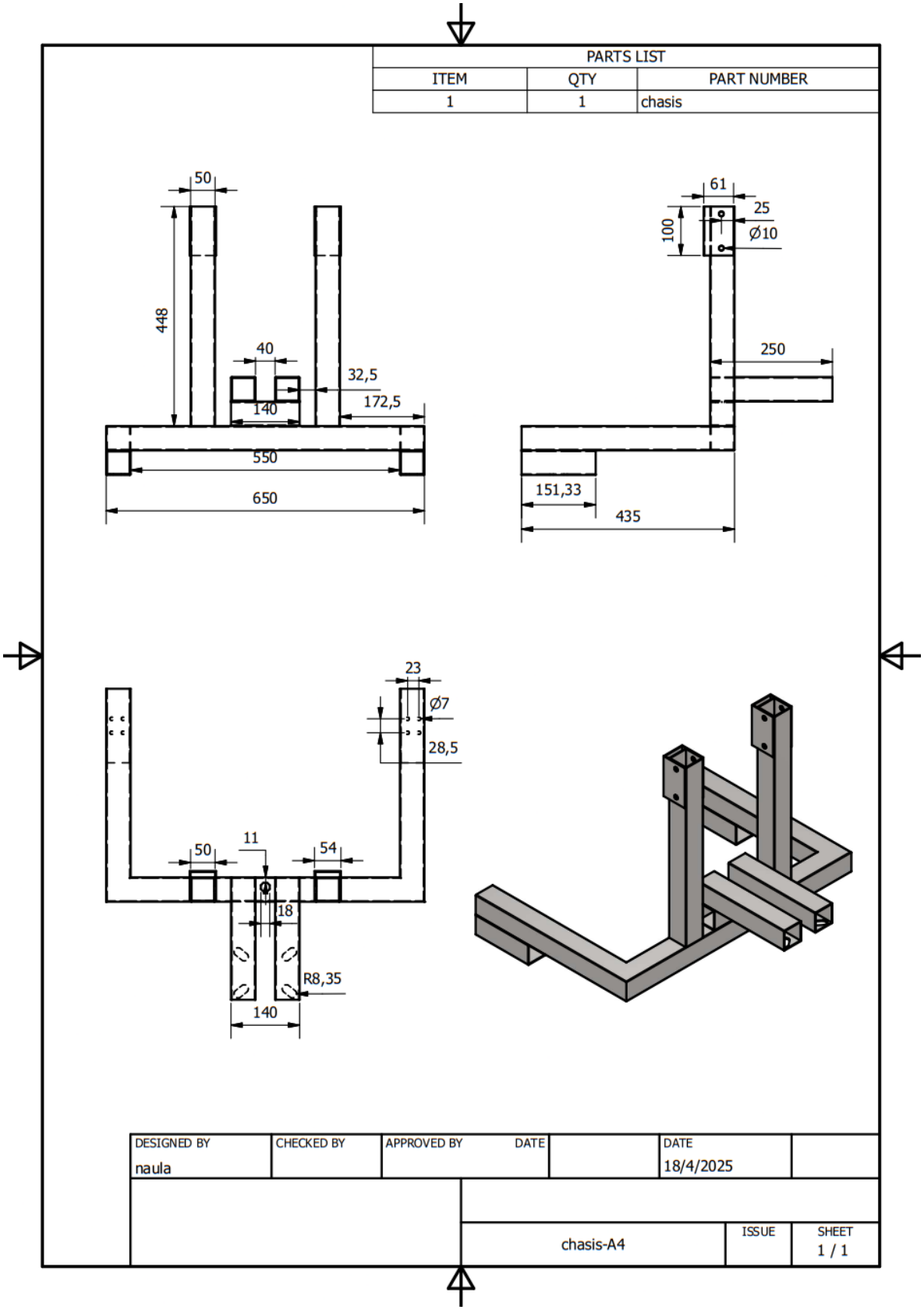
[13] X. V. Wang y L. Wang, «A literature survey of the robotic technologies during the COVID-19 pandemic», J. Manuf. Syst., vol. 60, pp. 823-836, jul. 2021, doi: 10.1016/j.jmsy.2021.02.005.
[14] D. Di Paola, A. Milella, G. Cicirelli, y A. Distante, «An Autonomous Mobile Robotic System for Surveillance of Indoor Environments», Int. J. Adv. Robot. Syst., vol. 7, n.o 1, p. 8, mar. 2010, doi: 10.5772/7254.
[15] B. Al-Tawil, T. Hempel, A. Abdelrahman, y A. Al-Hamadi, «A review of visual SLAM for robotics: evolution, properties, and future applications», Front. Robot. AI, vol. 11, p. 1347985, abr. 2024, doi: 10.3389/frobt.2024.1347985.
[16] Y. Zhang, Y. Wu, K. Tong, H. Chen, y Y. Yuan, «Review of Visual Simultaneous Localization and Mapping Based on Deep Learning», Remote Sens., vol. 15, n.o 11, p. 2740, may 2023, doi: 10.3390/rs15112740.
[17] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, y D. Rus, «LIO-SAM: Tightly-coupled Lidar Inertial Odometry via Smoothing and Mapping», en 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA: IEEE, oct. 2020, pp. 5135-5142. doi: 10.1109/IROS45743.2020.9341176.
[18] X. Wang, H. Hu, Y. Wang, y Z. Wang, «IoT Real-Time Production Monitoring and Automated Process Transformation in Smart Manufacturing»:, J. Organ. End User Comput., vol. 36, n.o 1, pp. 1-25, ene. 2024, doi: 10.4018/JOEUC.336482.
[19] Matthew Robinson, «ROS-Industrial Roadmap Journey and a Path Forward». [En línea]. Disponible en: https://rosindustrial.org/news/2025/4/30/ros-industrial-roadmap-journey-and-a-path-forward
[20] SICK, «More safety for humans and machines: Interview on the new robot standard ISO 10218», SICK Sensor Blog. Accedido: 16 de septiembre de 2025. [En línea]. Disponible en: https://www.sick.com/sk/cs/more-safety-for-humans-and-machines/w/blog-robotic-norm-iso10218
[21] Industrial trucks — Safety requirements and verification — Part 4: Driverless industrial trucks and their systems, 3691-4, Geneva., junio de 2023. [En línea]. Disponible en: https://www.iso.org/standard/83545.htm
[22] M. A. Fraifer, J. Coleman, J. Maguire, P. Trslić, G. Dooly, y D. Toal, «Autonomous Forklifts: State of the Art—Exploring Perception, Scanning Technologies and Functional Systems—A Comprehensive Review», Electronics, vol. 14, n.o 1, p. 153, ene. 2025, doi: 10.3390/electronics14010153.
[23] J. Jimbo y J. Estéban, «Levantamiento de mapas de ambientes para la navegación autónoma de robots móviles».
[24] R.-M. O. Corporation, «Lurking AMR - AMR ROBOT, AMR, AGV | Robotic Metal Tube Machines Manufacturer», YLM Group. Accedido: 24 de septiembre de 2025. [En línea]. Disponible en: https://www.ylm.com.tw/en/product/YLM_autonomous_mobile_robot_lidar_slam_JU.html
[25] «The Difference Between AGVs And Mobile Robots - CrossCo». Accedido: 24 de septiembre de 2025. [En línea]. Disponible en:

https://www.crossco.com/resources/articles/the-difference-between-agvs-and-mobile-robots/
[26] A. W. Staff, «Jingsong launches automated counterbalance forklift», Automated Warehouse. Accedido: 24 de septiembre de 2025. [En línea]. Disponible en: https://www.automatedwarehouseonline.com/jingsong-launches-automated-counterbalance-forklift/
[27] «Autonomous Carts - CarryMatic Automated Cart Systems», www.google.com. Accedido: 24 de septiembre de 2025. [En línea]. Disponible en: https://www.google.com/imgres?q=amr+tugger&imgurl=https://www.jtecindustries.com/wp-content/uploads/2022/09/Autonomous-Transfer-1.271.png&imgrefurl=https://www.jtecindustries.com/autonomous-carts-carrymatic/&docid=6MTj_4_lRqfDeM&tbnid=WMd7QxHHdAdHRM&vet=12ahUKEwiWmJWVjvOPAxUPRDABHaB3LC8QM3oECDMQAA..i&w=1600&h=844&hcb=2&ved=2ahUKEwiWmJWVjvOPAxUPRDABHaB3LC8QM3oECDMQAA&sfr=vfe&source=sh/x/im/can/1
[28] «MiR Shelf Carrier 250 | Mobile industrielle Robotter | AMR | i2r A/S», www.google.com. Accedido: 24 de septiembre de 2025. [En línea]. Disponible en: https://www.google.com/imgres?q=amr+Shelft-carrier&imgurl=https://www.robotbutikken.dk/wp-content/uploads/MIR-Shelf-carrier.jpg&imgrefurl=https://www.robotbutikken.dk/robotbutikken/roeq-tilbehoer-til-mobile-robotter/roeq-top-vogn-reolmoduler/mir-shelf-carrier-250/&docid=bDSxpdIISFtGwM&tbnid=o2IPqaV8NMs8fM&vet=12ahUKEwjM2OyhjvOPAxUbSTABHfA6AgYQM3oECCAQAA..i&w=551&h=550&hcb=2&itg=1&ved=2ahUKEwjM2OyhjvOPAxUbSTABHfA6AgYQM3oECCAQAA&sfr=vfe&source=sh/x/im/can/1
[29] «Old version Mini pallet truck AMR, mini forklift agv - Pallet truck forklift AGV - automated guided vehicle, autonomous mobile robot - okagv». Accedido: 24 de septiembre de 2025. [En línea]. Disponible en: https://www.ok-agv.com/Pallet_truck_forklift_AGV/%7B$title%7D-73-112-1.html
[30] J. Jimbo y J. Estéban, «Levantamiento de mapas de ambientes para la navegación autónoma de robots móviles».
[31] Oscar Ramos, Cinemática de Robots Móviles (parte 1/2), (7 de noviembre de 2021). Accedido: 26 de septiembre de 2025. [En línea Video]. Disponible en: https://www.youtube.com/watch?v=2Asd4RH3Gmw
[32] Oscar Ramos, Cinemática de Robots Móviles (parte 2/2), (7 de noviembre de 2021). Accedido: 27 de septiembre de 2025. [En línea Video]. Disponible en: https://www.youtube.com/watch?v=CAx47sW-xxM
[33] T. Lackner, J. Hermann, C. Kuhn, y D. Palm, «Review of autonomous mobile robots in intralogistics: state-of-the-art, limitations and research gaps», Procedia CIRP, vol. 130, pp. 930-935, 2024, doi: 10.1016/j.procir.2024.10.187.
[34] «ROS 2 Políticas de Calidad de Servicio». Accedido: 28 de septiembre de 2025. [En línea]. Disponible en: https://design.ros2.org/articles/qos.html?utm_source=chatgpt.com

[35] J. A. Ruiz, «Analysis of a RGB-D SLAM system using Real-Time Appearance-Based Mapping on the Kbot robot», Comput. Sci..
[36] S. Yu, C. Fu, A. K. Gostar, y M. Hu, «A Review on Map-Merging Methods for Typical Map Types in Multiple-Ground-Robot SLAM Solutions», Sensors, vol. 20, n.o 23, p. 6988, dic. 2020, doi: 10.3390/s20236988.
[37] M. Labbé y F. Michaud, «RTAB-Map as an Open-Source Lidar and Visual SLAM Library for Large-Scale and Long-Term Online Operation», J. Field Robot., vol. 36, n.o 2, pp. 416-446, mar. 2019, doi: 10.1002/rob.21831.
[38] «Figure 5. ROS2 architecture overview.», ResearchGate. Accedido: 11 de octubre de 2025. [En línea]. Disponible en: https://www.researchgate.net/figure/ROS2-architecture-overview_fig2_372420797
[39] mattwojo, «Instalación de WSL». Accedido: 4 de noviembre de 2025. [En línea]. Disponible en: https://learn.microsoft.com/es-es/windows/wsl/install
[40] «Ubuntu (deb packages) — ROS 2 Documentation: Jazzy documentation». Accedido: 4 de noviembre de 2025. [En línea]. Disponible en: https://docs.ros.org/en/jazzy/Installation/Ubuntu-Install-Debs.html

ANEXOS

Anexo 1. Características de dimensiones del bastidor del prototipo



Anexo 2. Proceso de levantamiento de nodos ROS para la funcionalidad del sistema

Nodo de percepción Realsense.

```
ros2 launch realsense2_camera rs_launch.py \
initial_reset:=true \
rgb_camera.enable:=true depth_module.enable:=true \
rgb_camera.color_profile:=424x240x15 \
depth_module.depth_profile:=424x240x15 \
pointcloud.enable:=false \
align_depth.enable:=true
Nodo Traductor de datos Twist a PWM.
sudo -E bash -c "source /opt/ros/jazzy/setup.bash; \
source /home/darwin/ros2_ws/install/setup.bash; \
ros2 run paquete_amr cmdvel_pwm_bridge"
Nodo Publicador de estado del Prototipo.

ros2 run robot_state_publisher robot_state_publisher \
"$(ros2 pkg prefix
my_amr_description)/share/my_amr_description/urdf/amr_triciclo.urdf"
Nodo RTAB-Map.
ros2 launch rtabmap_launch rtabmap.launch.py \
frame_id:=base_link \
approx_sync:=true rgbd_sync:=true queue_size:=10 \
rgb_topic:=/camera/camera/color/image_raw \
depth_topic:=/camera/camera/aligned_depth_to_color/image_raw \
camera_info_topic:=/camera/camera/color/camera_info \
rtabmap.rgb_image_transport:=compressed \
rtabmap.depth_image_transport:=compressedDepth \
rgbd_sync.image_transport:=compressed \
rgbd_sync.depth_image_transport:=compressedDepth \
rgbd_odometry.image_transport:=compressed \
rgbd_odometry.depth_image_transport:=compressedDepth \
rtabmap_viz.rgb_image_transport:=compressed \
rtabmap_viz.depth_image_transport:=compressedDepth \
rtabmap_args:="--Optimizer/Strategy 1" \
database_path:=$HOME/rtabmap_db/exp_prueba_AMRvel.db
Nodo Nav2
ros2 launch amr_nav2_bringup nav2_amr_online.launch.py \
use_sim_time:=false \
```

Anexo 3. Algoritmo personalizado para la traducción de señales de velocidad de Nav2 a pulsos PWM

```
params_file:=/home/darwin/ros2_ws/nav2_params_amr.yaml
Script Bridge de datos Cmd_vel a PWM
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
import math
import signal
import time
import threading
import RPi.GPIO as GPIO
import rclpy
from rclpy.node import Node
from geometry_msgs.msg import Twist
def clamp(x, lo, hi):
return max(lo, min(hi, x))
class CmdVelBridge(Node):
def __init__(self):
super().__init__('cmd_vel_bridge')
# Parámetros ROS
        self.R = float(self.get_parameter('wheel_radius').value)
        self.L = float(self.get_parameter('track_width').value)
        self.VMAX = float(self.get_parameter('max_linear').value)
        self.WMAX = float(self.get_parameter('max_angular').value)
        self.DEADBAND = float(self.get_parameter('deadband_speed').value)
        self.ACC_LIM = float(self.get_parameter('accel_limit').value)
        self.ACCW_LIM = float(self.get_parameter('ang_accel_limit').value)

        self.PIN_L_PWM = int(self.get_parameter('left_pwm_pin').value)
        self.PIN_L_DIR = int(self.get_parameter('left_dir_pin').value)
        self.PIN_R_PWM = int(self.get_parameter('right_pwm_pin').value)
        self.PIN_R_DIR = int(self.get_parameter('right_dir_pin').value)
        self.PIN_ENA   = int(self.get_parameter('enable_pin').value)

        self.INV_L = bool(self.get_parameter('invert_left').value)
        self.INV_R = bool(self.get_parameter('invert_right').value)

        self.PWMF = int(self.get_parameter('pwm_freq').value)
        self.DUTY_MIN = float(self.get_parameter('pwm_min_duty').value)
        self.DUTY_MAX = float(self.get_parameter('pwm_max_duty').value)
        self.RATE = float(self.get_parameter('control_rate').value)

        self.WATCHDOG = float(self.get_parameter('watchdog_timeout').value)
        self.RAMP_DOWN = bool(self.get_parameter('ramp_down_on_timeout').value)

        # Límite de velocidad de rueda (conservador)
        self.VWHEEL_MAX = max(0.05, self.VMAX + self.WMAX * self.L * 0.5)
```

```
        # Estados
        self._lock = threading.Lock()
        self.v_cmd = 0.0
        self.w_cmd = 0.0
        self.v_set = 0.0
        self.w_set = 0.0
        self.last_cmd_time = time.monotonic()

        # GPIO
        GPIO.setwarnings(False)
        GPIO.setmode(GPIO.BCM)

# Lista de pines a usar
        pins = [self.PIN_L_DIR, self.PIN_R_DIR, self.PIN_L_PWM, self.PIN_R_PWM]
        if self.PIN_ENA >= 0:
            pins.append(self.PIN_ENA)

        # IMPORTANTE: usar setup en lista + initial=GPIO.LOW
        GPIO.setup(pins, GPIO.OUT, initial=GPIO.LOW)

        # Habilitar drivers si se usa pin de ENABLE
        if self.PIN_ENA >= 0:
            GPIO.output(self.PIN_ENA, GPIO.HIGH)

        # PWM
        self.pwmL = GPIO.PWM(self.PIN_L_PWM, self.PWMF)
        self.pwmR = GPIO.PWM(self.PIN_R_PWM, self.PWMF)
        self.pwmL.start(0.0)
        self.pwmR.start(0.0)

        # Suscripción
        self.sub = self.create_subscription(Twist, '/cmd_vel', self._on_cmd_vel,

        # Timer de control
        self.dt = 1.0 / self.RATE
        self.timer = self.create_timer(self.dt, self._control_loop)

        # Señales
        signal.signal(signal.SIGINT, self._sig_handler)
        signal.signal(signal.SIGTERM, self._sig_handler)

        self.get_logger().info('cmd_vel_bridge listo. Esperando /cmd_vel...')

    def _on_cmd_vel(self, msg: Twist):
        with self._lock:
            self.v_cmd = clamp(msg.linear.x, -self.VMAX, self.VMAX)
            self.w_cmd = clamp(msg.angular.z, -self.WMAX, self.WMAX)
            self.last_cmd_time = time.monotonic()
```

```
def _control_loop(self):
    now = time.monotonic()

    # Watchdog
    if (now - self.last_cmd_time) > self.WATCHDOG:
        if self.RAMP_DOWN:
            self.v_cmd = 0.0
            self.w_cmd = 0.0
        else:
            self._apply_pwm(0.0, 0.0)
            return

    # Rampas
    with self._lock:
        self.v_set = self._ramp(self.v_set,self.v_cmd,self.ACC_LIM, self.dt)
        self.w_set =self._ramp(self.w_set,self.w_cmd,self.ACCW_LIM, self.dt)

        # Diferencial
        v_l = self.v_set - (self.w_set * self.L * 0.5)
        v_r = self.v_set + (self.w_set * self.L * 0.5)

    # PWM + DIR
    duty_l, dir_l = self._speed_to_pwm(v_l, self.INV_L)
    duty_r, dir_r = self._speed_to_pwm(v_r, self.INV_R)

    GPIO.output(self.PIN_L_DIR, dir_l)
    GPIO.output(self.PIN_R_DIR, dir_r)
    self.pwmL.ChangeDutyCycle(duty_l)
    self.pwmR.ChangeDutyCycle(duty_r)

def _ramp(self, current, target, accel_limit, dt):
    max_step = accel_limit * dt
    if target > current:
        return min(target, current + max_step)
    else:
        return max(target, current - max_step)

def _speed_to_pwm(self, v_wheel, invert: bool):
    if abs(v_wheel) < self.DEADBAND:
        return 0.0, GPIO.LOW

    frac = clamp(v_wheel / self.VWHEEL_MAX, -1.0, 1.0)
    forward = frac >= 0.0
    dir_bit = GPIO.HIGH if forward else GPIO.LOW
    if invert:
        dir_bit = GPIO.LOW if dir_bit == GPIO.HIGH else GPIO.HIGH

    mag = abs(frac)
    duty = self.DUTY_MIN + mag * (self.DUTY_MAX - self.DUTY_MIN)
    duty = clamp(duty, 0.0, self.DUTY_MAX)
```

```
        return duty, dir_bit

    def _apply_pwm(self, duty_left, duty_right):
        self.pwmL.ChangeDutyCycle(clamp(duty_left, 0.0, self.DUTY_MAX))
        self.pwmR.ChangeDutyCycle(clamp(duty_right, 0.0, self.DUTY_MAX))

    def _sig_handler(self, *_):
        self.get_logger().warn('Parando PWM y limpiando GPIO...')
        try:
            self.pwmL.ChangeDutyCycle(0.0)
            self.pwmR.ChangeDutyCycle(0.0)
            if self.PIN_ENA >= 0:
                GPIO.output(self.PIN_ENA, GPIO.LOW)
        finally:
            GPIO.cleanup()
            rclpy.shutdown()

def main():
    rclpy.init()
    node = CmdVelBridge()
    try:
        rclpy.spin(node)
    except KeyboardInterrupt:
        pass
    finally:
        try:
            node._sig_handler()
        except Exception:
            pass

if __name__ == '__main__':
    main()
```