UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

Desarrollo de un robot anfitrión

INFORME DE ACTIVIDADES PROFESIONALES

Que para obtener el título de

Ingeniero Mecatrónico

P R E S E N T A

Kevin Vargas Gutiérrez

ASESOR DE INFORME

M.I. Erik Peña Medina

Ciudad Universitaria, Cd. Mx., 2026

PROTESTA UNIVERSITARIA DE INTEGRIDAD Y HONESTIDAD ACADÉMICA Y PROFESIONAL
(Titulación con trabajo escrito)

De conformidad con lo dispuesto en los artículos 87, fracción V, del Estatuto General, 68, primer párrafo, del Reglamento General de Estudios Universitarios y 26, fracción I, y 35 del Reglamento General de Exámenes, me comprometo en todo tiempo a honrar a la institución y a cumplir con los principios establecidos en el Código de Ética de la Universidad Nacional Autónoma de México, especialmente con los de integridad y honestidad académica.

De acuerdo con lo anterior, manifiesto que el trabajo escrito titulado DESARROLLO DE UN ROBOT ANFITRION que presenté para obtener el titulo de INGENIERO MECATRÓNICO es original, de mi autoría y lo realicé con el rigor metodológico exigido por mi Entidad Académica, citando las fuentes de ideas, textos, imágenes, gráficos u otro tipo de obras empleadas para su desarrollo.

En consecuencia, acepto que la falta de cumplimiento de las disposiciones reglamentarias y normativas de la Universidad, en particular las ya referidas en el Código de Ética, llevará a la nulidad de los actos de carácter académico administrativo del proceso de titulación.

KEVIN VARGAS GUTIERREZ
Número de cuenta: 317089042

AGRADECIMIENTOS

Deseo expresar mi más sincero agradecimiento a todas las personas que me brindaron su apoyo, guía y colaboración durante la elaboración de este informe de titulación, ya que sin apoyo este logro no habría sido posible. En primer lugar, agradezco profundamente a mis principales asesores y maestros, el Ing. Felipe Rivas Campos y el M.I. Erik Peña Medina, por su constante orientación, la generosidad con la que compartieron sus conocimientos, sus valiosos consejos y, sobre todo, por creer en mí y estar siempre disponibles cuando más los necesité.

Agradezco a Optimissa y Actinver, principalmente a Jorge Ulises González Medina, Chief Data & AI Officer de Actinver, por darme la oportunidad de desarrollar este proyecto. Su confianza, apoyo y disposición por proporcionar la confianza y los recursos necesarios para que pudiera aplicar mis conocimientos en un área que me apasiona. Extiendo también mi agradecimiento a mi compañero de trabajo, Saúl Duarte González, por su colaboración, por permitirme trabajar conjuntamente durante el desarrollo del proyecto y por confiar plenamente en mis capacidades.

Sobre todo, agradezco a mi familia, que han sido siempre mi principal pilar y motivación. Gracias a su apoyo brindado incondicionalmente, de manera moral y económicamente, gracias a ello, he logrado alcanzar mis metas y llegar hasta este momento tan importante en mi vida. Este logro no habría sido posible sin su constante respaldo y confianza en mí.

Finalmente, quiero agradecer a todas las personas que, de alguna u otra forma, contribuyeron a la elaboración de este informe. A cada uno de ustedes les doy mi más profundo reconocimiento y gratitud. Espero que se sientan parte de este logro, tal como yo lo siento.

ÍNDICE

CONTENIDO

1. INTRODUCCIÓN

En un entorno global cada vez más competitivo, la transformación digital se ha convertido en un eje central para las instituciones financieras que buscan mejorar la eficiencia operativa y ofrecer experiencias personalizadas a sus clientes. Tecnologías de vanguardia como la inteligencia artificial y la robótica, están siendo adoptadas a nivel internacional para revolucionar los procesos y servicios tradicionales.

En este contexto, surge el proyecto de este informe, impulsado por la observación directa de inversionistas y clientes de un banco, que, tras viajar por distintos países, identificaron una clara diferencia entre la tecnología ofrecida en México y aquellas implementados en mercados más desarrollados. Mientras que en otros países ya es común encontrar robots que interactúan con los clientes y ofrecen servicios automatizados de atención, en México es muy poco probable ver a un robot interactuando con los clientes. Tal es el caso de "*Pepper*", el robot de SoftBank desplegado en sucursales de HSBC que como lo menciona Bill Streeter en su artículo *Seriously Successful Results From HSBC Bank's Branch Robot Rollout*, *"Jeremy Balkin, HSBC USA's Head of Innovation, says there has been a 60% increase in new business at the flagship branch compared to the prior comparable year and more than a five-fold increase in foot traffic since Pepper was introduced. In addition, the bank has tallied more than 1 billion social media impressions about the robot, 99% expressing a positive sentiment. Across all of HSBC's Pepper devices there have been almost 25,000 consumer interactions"* [1]. El robot marcó una gran diferencia al crear un gran impacto por implementar este tipo de tecnología dentro de sus sucursales, reportando un aumento del 60 % en nuevos negocios y más de 25,000 interacciones con clientes en su primera fase operativa [1]. O casos como el del restaurante de Viña del Mar con el "michi-robot", que elevó sus ganancias hasta en 20% [2], o el robot de servicio "Relay" creado por la empresa estadounidense Savioke, empleado en cadenas hoteleras para la entrega automática de comida, pasta de dientes, toallas y otros objetos en las habitaciones

de los clientes de hoteles [3]. Todos estos casos demuestran el impacto y la eficiencia alcanzada por la robótica de servicio en distintas áreas alrededor del mundo.

Motivados por esta situación y con el objetivo de colocar al banco a la vanguardia de la innovación nacional en servicios financieros, se desarrolló un robot móvil autónomo capaz de atender a los usuarios en el entorno de los centros financieros.

El robot host financiero es un sistema robótico móvil autónomo diseñado para actuar como anfitrión en los centros financieros del banco, capaz de recibir a los usuarios, interactuar con ellos de forma personalizada según su rango de edad, y guiarlos hacia áreas y servicios clave dentro de la sucursal mediante navegación autónoma. De este modo, no solo se optimiza la experiencia del cliente, sino que también se genera algo distintivo para el banco en comparación con los demás centros financieros en México.

Este informe lo hice como parte de mi proceso de titulación por experiencia profesional, en este documento detallo todas las etapas del desarrollo del proyecto en el que estuve y aún estoy trabajando. En este trabajo trato de reflejar como apliqué mis conocimientos y habilidades que adquirí a lo largo de la ingeniería en un problema real; desde mi perspectiva, este proyecto fue perfecto porque es lo que yo esperaba hacer cuando comencé a estudiar la ingeniería en mecatrónica; en otras palabras, tener conocimiento de distintas ingenierías para poder innovar y dejar huella en el mundo, creando cosas útiles por mí mismo.

1.1. OBJETIVO

El objetivo del proyecto es desarrollar el primer robot anfitrión autónomo de México en un centro financiero, transformando la experiencia del servicio de la atención al cliente en entornos bancarios. Este robot será el host de un centro financiero, por lo que deberá recibir a los clientes y mantener una interacción personalizada con ellos, de tal forma que dependiendo de la edad del usuario con el que este interactuando, el robot modificará la velocidad a la que navega y la forma en la que se comportará con ellos, esto con el objetivo de ofrecer una experiencia más personalizada. Además, tendrá que identificar el destino al que quieren llegar los usuarios y guiarlos mediante navegación autónoma dentro de los centros financieros. Se tiene pensado que el robot trabaje en un solo piso plano dentro del centro financiero, evitando rampas y escaleras para tener un buen desempeño. Todo esto con el objetivo de hacer que cada interacción con el cliente se sienta más humana y personalizada, además de reducir el tiempo en el que los usuarios tardan en ubicar las áreas dentro del centro financiero.

2. ANTECEDENTES

En este apartado hablaremos sobre los antecedentes antes de que trabajara en el proyecto y cómo fue que comencé a relacionarme con él. El proyecto del robot host financiero fue llevado a cabo desde el inicio como un desarrollo iterativo, permitiendo una evolución progresiva en su diseño y funcionalidad. En un principio se fueron delimitando las funciones y las especificaciones que se buscaban para lograr cumplir con el objetivo. En cuanto a su apariencia, se buscó que tuviera un enfoque minimalista, con una apariencia amigable y atractiva para los usuarios. Durante las primeras etapas se exploraron diversas propuestas de diseño; sin embargo, el concepto visual de la Figura 1 fue el punto de partida e inspiración inicial del proyecto, dicho concepto me lo compartió el banco por medio de una documentación para conocer la conceptualización del proyecto.

Figura 1. Diseño con el que se inspiró el proyecto.

Se propuso que el robot contara con una configuración de locomoción diferencial con cuatro llantas (Figura 2, sacada de la documentación del proyecto), pensando en que pudiera soportar el peso total del robot y al mismo tiempo con una buena capacidad de movimiento incluso en espacios cerrados.

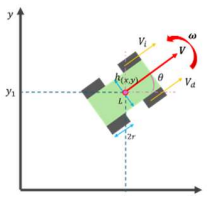


Figura 2. Representación de una configuración diferencial de 4 ruedas.

2.1. PRIMERA VERSIÓN (PROTOTIPO A)

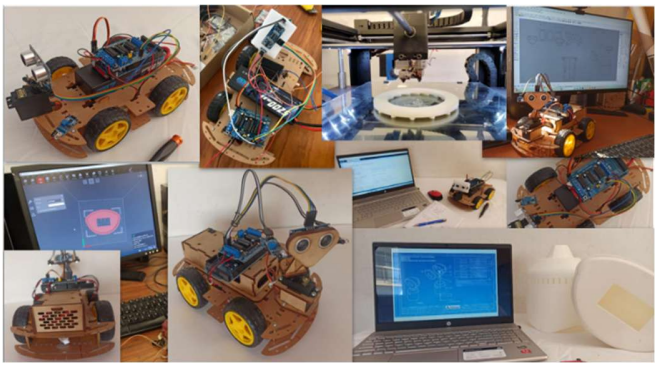


Figura 3. Proceso de diseño e implementación de la primera etapa.

En el Prototipo A, se propuso desarrollar una versión funcional básica del robot enfocada en la integración de posibles sensores y la selección de un microcontrolador robusto y escalable, capaz de ejecutar algoritmos de evasión de obstáculos mediante un comportamiento reactivo. Además, se realizó una primera aproximación al diseño en CAD del chasis del robot, en la Figura 3 se pueden ver algunas imágenes del proceso de diseño e implementación de esta primera etapa.

Como se mencionó anteriormente, este desarrollo siguió un enfoque iterativo. Al finalizar esta primera etapa, el prototipo presentó las características mostradas en la Figura 4.

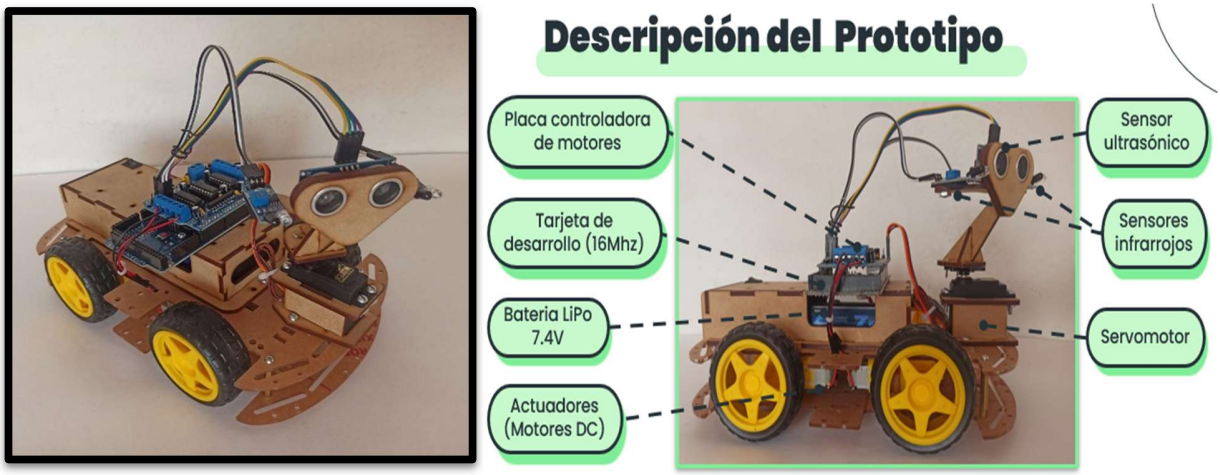


Figura 4. Prototipo A.

Como se puede observar en las imágenes, el primer prototipo cuenta con una carcasa fabricada en MDF (*Medium Density Fiberboard*), cortada con láser, sobre la cual se integraron los sensores y actuadores básicos. El robot fue diseñado para desplazarse con un comportamiento reactivo basado en la lógica de orden cero. Esto significa que el robot

avanza continuamente mientras un servomotor realiza un barrido con un sensor ultrasónico para detectar obstáculos al frente y a los lados (Figura 5, sacada de la documentación del proyecto).

En función de esta detección, el robot toma decisiones simples: si detecta un obstáculo al frente, girará hacia cualquier lado libre; si detecta un obstáculo al frente y a la izquierda, girará a la derecha, y así sucesivamente. De este modo, el robot busca desplazarse hacia la dirección con mayor espacio libre, sin seguir una meta específica.

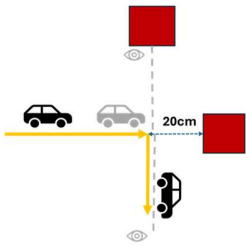


Figura 5. Lógica de orden cero.

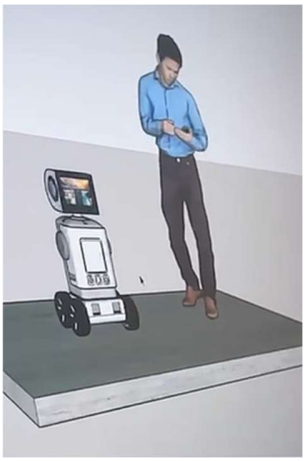
2.2. SEGUNDA VERSIÓN (PROTOTIPO B)

En la segunda versión del prototipo refinó la apariencia y las funcionalidades del robot, esta versión incorporó una mayor cantidad de sensores para mejorar la percepción del entorno, los cuales consisten de un sensor ultrasónico montado sobre un servomotor para realizar un barrido frontal, dos sensores ultrasónicos adicionales orientados hacia los costados, con el objetivo de detectar obstáculos laterales, la implementación de los encoders de los motores de las llantas y un sensor IMU para obtener datos sobre la posición y orientación del robot en tiempo real. El chasis fue mejorado utilizando acrílico, lo que permitió agregar un segundo nivel destinado a servir como base para un sensor LiDAR (aunque en esta etapa aún no está en funcionamiento), dejando el primer piso libre para colocar una Raspberry Pi o un microcontrolador. Asimismo, se optimizaron otros componentes del robot, como las llantas (sustituidas por unas con mayor agarre), la batería (por una con mayor capacidad), los indicadores visuales y un sistema amortiguador de colisiones, entre otros elementos, buscando así una versión más robusta y preparada para hacer pruebas y preparada para futuras mejoras (Figura 6).

Figura 6. Prototipo B.

2.3. DISEÑO DEL CHASIS

Se optó por realizar el diseño del chasis y su fabricación de forma externa. Para ello, se elaboró el diseño utilizando AutoCAD (Figura 7.a), y posteriormente se fabricó la carcasa a base de MDF con las dimensiones reales y características que iba a tener (Figura 7.b). A continuación, se presenta el resultado final, así como el proceso de fabricación, el cual se puede consultar en el siguiente enlace: https://drive.google.com/drive/folders/1SuVRtF243Yd7mBcYPLg6X5PWe9NxElzm?usp=sharing



a)

b)

Figura 7. a) Diseño del chasis y b) Carcasa hecha de MDF.

2.3.1 PLANOS DEL CHASIS

En este apartado se muestran los planos en milímetros con los cuales se elaboró el chasis del proyecto (Figura 8), fue realizado tratando de mantener el concepto original, y que al mismo tiempo cumpla con los requerimientos necesarios para poder lograr sus objetivos.

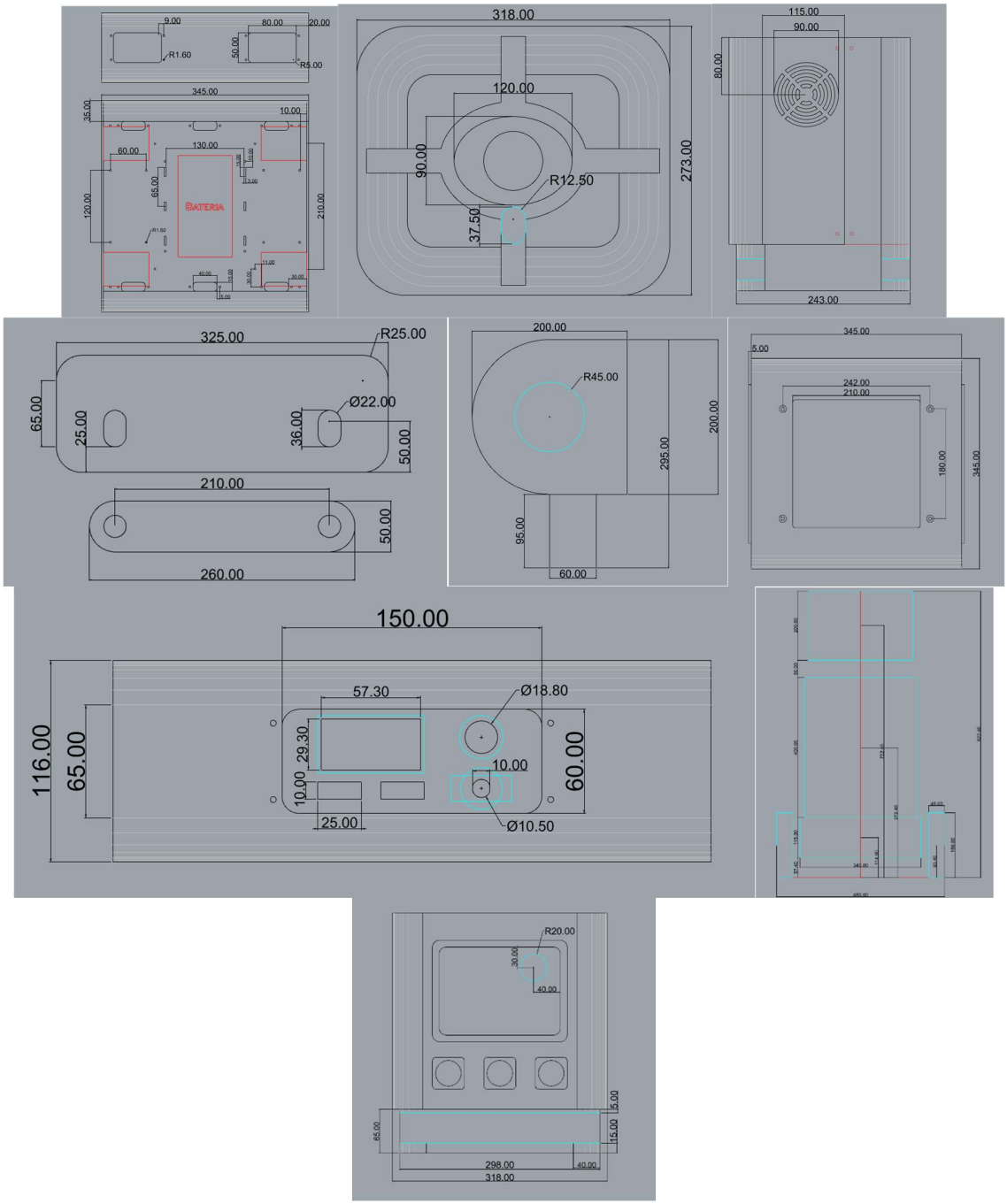


Figura 8. Planos del Chasis.

2.4. PROPUESTA DE DESARROLLO

Para poder cumplir con los objetivos planteados en el proyecto propuse seguir desarrollando el proyecto con ROS 2 (*Robot Operating System 2*), ya que este es un entorno de desarrollo que proporciona un conjunto de herramientas, bibliotecas y repositorios que conforman una infraestructura modular para el desarrollo de sistemas robóticos. Existen dos versiones principales de ROS: ROS1 y ROS 2; sin embargo, decidí desarrollar el proyecto en ROS 2 ya que tiene un enfoque más industrial, tiene una arquitectura moderna modular y actualmente todas las actualizaciones de ROS se están llevando a cabo en ROS 2, por lo que es una opción más robusta y escalable en comparación con ROS 1, que tiene un enfoqué un poco más didáctico y ya no se trabaja en nuevas actualizaciones, por lo que sus herramientas están más limitadas. Entre las distintas distribuciones de ROS 2, decidí utilizar una de las versiones LTS (*Long Term Support*) la distribución Humble Hawksbill, ya que proporciona estabilidad y soporte a largo plazo [4]. En resumen, usar ROS2 para desarrollar el proyecto me permitió trabajar con herramientas profesionales especializadas en la robótica, mejorando la eficiencia y la escalabilidad del proyecto. Algunas de las principales razones por las que elegí ROS2 las detallaré a continuación:

1. Comunicación modular y en tiempo real

ROS 2 está diseñado para soportar aplicaciones que requieren comunicación en tiempo real y lo hace de una forma modular, ya que utiliza el middleware DDS (*Data Distribution Service*) que garantiza una comunicación confiable y eficiente entre los diferentes nodos (módulos o procesos del robot) [5]. Esto lo hace con modelos basados en una comunicación de tipo publicador/suscriptor, tal como se muestra en la Figura 9.

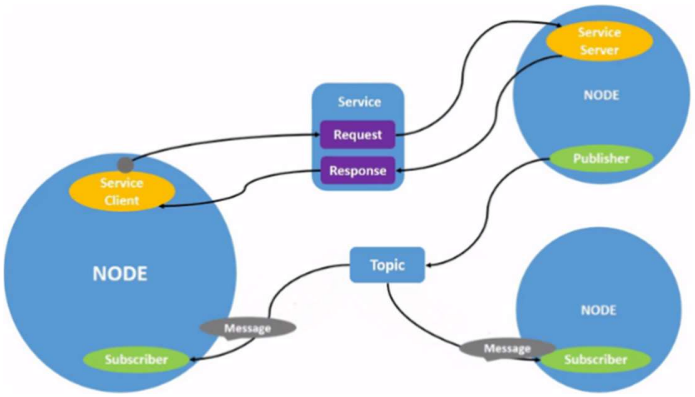


Figura 9. Representación de la comunicación por nodos en ROS 2 [6].

2. Soporte para robótica industrial y entornos críticos

A diferencia de ROS 1, que estaba orientado principalmente a investigación y aplicaciones no críticas, ROS 2 se ha desarrollado pensando en sistemas industriales y en entornos críticos donde la fiabilidad, la seguridad y la implementación en tiempo real son esenciales. Esto lo hace apto para aplicaciones más complejas y profesionales [4].

3. Mejor manejo de plataformas embebidas

ROS 2 está mejor adaptado para sistemas embebidos y recursos limitados, lo que permite usar hardware más económico y con menos capacidad (como microcontroladores) sin perder la funcionalidad del sistema de control robótico [4].

4. Escalabilidad y modularidad

También está diseñado para ser escalable, permitiendo que se utilicen redes distribuidas de robots o que se desplieguen sistemas complejos con múltiples nodos en diferentes computadoras o dispositivos. También mejora el modularidad, permitiendo dividir el sistema robótico en componentes reutilizables [4].

5. Comunidad activa

La comunidad de ROS 2 sigue creciendo rápidamente, lo que facilita encontrar soporte, bibliotecas y paquetes desarrollados por otros, así como colaboración con investigadores y profesionales de todo el mundo [4].

6. Recursos de desarrollo

ROS 2 cuenta con herramientas que nos facilitan el desarrollo de nuestro robot para lograr nuestros objetivos, algunas de las herramientas que utilicé son: Rviz, Gazebo y Navigation2; de las cuales hablaré más adelante [4].

2.4.1. HERRAMIENTAS VIRTUALES

Rviz es una herramienta integrada en ROS para el manejo de datos de diferentes tipos de sensores, como cámaras, LiDAR, sensores ultrasónicos, GPS, y otros. Se pueden ver las imágenes de cámaras, los puntos de nube generados por el LiDAR o los datos de distancias de otros sensores, también muestra el modelo del robot en 3D en un entorno gráfico interactivo, lo que facilita comprender cómo se está moviendo, su orientación y posición en el espacio, y cómo interactúa con el entorno, además, permite visualizar trayectorias de movimiento planeadas y ejecutadas, así como mapas generados por algoritmos de navegación y localización (SLAM), lo que es vital para robots móviles que operan en entornos complejos [4] y [7]. Es extremadamente útil para desarrollar y depurar aplicaciones robóticas, ya que proporciona una representación visual en tiempo real del entorno del robot y de sus percepciones. En la Figura 10 podemos ver el logo de Rviz.

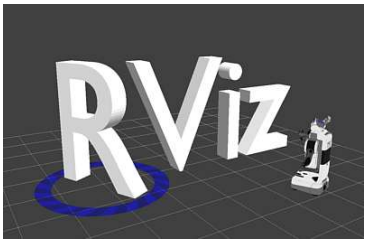


Figura 10. Logo de Rviz [8].

Gazebo es un simulador de robótica con un motor de física de código abierto que permite crear y simular robots en un entorno virtual realista antes de llevarlos al mundo físico (Figura 11). Es utilizado en el desarrollo de la robótica debido a su capacidad para simular sensores y entornos complejos de manera precisa, esto facilita el desarrollo de algoritmos de percepción y localización sin necesidad de implementar el hardware de inmediato. Gazebo se integra con el ROS para el diseño e implementación de robots complejos con comportamiento avanzado altamente personalizable, permitiendo a los usuarios crear y modificar mundos y modelos virtuales según sus necesidades, por lo que podemos generar un espacio virtual (como en un videojuego), que simule con modelos 3D realistas lo que pasaría en el mundo real (generar muros, colisiones, efectos de la gravedad, comportamiento de los sensores ante ciertas situaciones, etc) [9]; además, se pueden repetir las pruebas en Gazebo tantas veces como sea necesario sin el riesgo de dañar componentes físicos. Lo anterior, es especialmente útil cuando se están ajustando parámetros como los controladores de velocidad o algoritmos de control, ya que las simulaciones son predecibles y se pueden controlar completamente.

Figura 11. Logo de Gazebo [9].

Nav2 (abreviatura de *Navigation 2*, su logo se puede ver en la Figura 12), es un sistema de navegación modular desarrollado por Steve Musinski para ROS 2. Es la versión mejorada del paquete de navegación original de ROS 1, diseñado para dotar a los robots móviles moverse de manera autónoma en un entorno. Nav2 es especialmente útil para robots móviles diferenciales (que se mueven controlando por separado las velocidades de sus ruedas). Nav2 es una solución poderosa y flexible para desarrollar robots móviles

diferenciales, ya que proporciona todas las herramientas y algoritmos necesarios para la navegación autónoma y el control de movimiento, dentro de lo que se puede lograr hacer con Nav2 en un robot móvil se encuentra: Localizarse dentro de un mapa conocido o previamente generado (localización), mapear el entorno (SLAM) si el robot no tiene un mapa previo, planificar rutas desde su posición actual hasta un destino, evitar obstáculos en su trayecto y controlar el movimiento de manera segura y eficiente, ejecutando los comandos necesarios para seguir la ruta planificada [10].

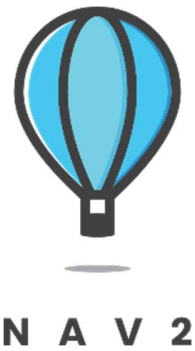


Figura 12. Logo de NAV2 [10].

Fusion 360 (Figura 13), es un software de diseño asistido por computadora (CAD), fabricación asistida por computadora (CAM) e ingeniería asistida por computadora (CAE) desarrollado por Autodesk. Este programa tiene la capacidad de integrar múltiples herramientas de diseño paramétrico que facilita el diseño, la simulación y la fabricación de productos de manera rápido y eficiente [11]. Es conocido por su simplicidad y baja curva de aprendizaje, lo que lo hace accesible tanto para profesionales como para aficionados.

Figura 13. Logo de Fusion 360 [11].

2.5. DIVISIÓN DE TRABAJO

Actualmente, el proyecto robot host financiero ha sido desarrollado por dos personas: el responsable principal del proyecto y yo. Como hemos visto a lo largo de esta sección, me integré al proyecto después de que se completara el segundo prototipo, en el momento en que ya se había mandado a fabricar la estructura del robot. En este contexto, el enfoque principal eran las funciones del robot. Durante el primer mes, mientras se revisaba mi propuesta de desarrollo, comencé a trabajar en el sistema de interfaz, creando imágenes de las animaciones con una técnica de Pixel Art y dándole movimiento utilizando un programa de Python. En paralelo, el encargado del proyecto trabajaba en la implementación de un algoritmo de búsqueda para lograr una navegación deliberativa, utilizando un mapa generado manualmente. No obstante, luego de que se revisara mi propuesta con el líder del departamento, se aceptó mi propuesta, dando lugar a que el proyecto comenzara a desarrollarse en ROS 2, desde ese momento, las responsabilidades se dividieron de la siguiente manera:

- El encargado del proyecto se enfocó en el desarrollo del sistema de chasis y alimentación, incluyendo la creación de soportes para los distintos componentes y el diseño del esquema eléctrico para distribuir adecuadamente la energía entre los sistemas.
- Por mi parte, asumí el desarrollo de los sistemas de instrumentación, interfaz, locomoción y procesamiento, integrando todos estos sistemas dentro del entorno de ROS 2, para cumplir con el objetivo planteado.

En la siguiente sección se profundizará en la descripción de estos sistemas principales y cómo se sintetizan entre sí para lograr el funcionamiento del proyecto de robot host financiero.

2.6 METODOLOGÍA

El desarrollo del proyecto se realizó por medio de un proceso iterativo, buscando siempre mejorarlo aprovechando la escalabilidad del proyecto y la modularidad de sus componentes; además, para su desarrollo se utilizaron todas las herramientas virtuales mencionadas en la sección 2.4.1, teniendo diferentes criterios de validación en cada etapa, analizando de forma independiente cada sistema.

El primer paso fue el diseño del robot, en donde se exploraron muchas posibilidades siempre teniendo como criterio de evaluación las funciones y las especificaciones que se buscaban para lograr cumplir con el objetivo, según la información que me dieron, el diseño es el resultado de una combinación de ideas, bocetos, comparaciones con robots ya existentes y la ayuda de la inteligencia artificial para generar un diseño que no se haya visto antes.

El siguiente paso (el prototipo A), fue el resultado de empezar a materializar el concepto y el diseño que se tenía desde un inicio, en donde se exploraron diferentes posibilidades para poder integrarle diferentes sensores al robot, en este caso el criterio de evaluación fue el desempeño de los sensores al enfrentarse a condiciones reales dentro del espacio de trabajo, una de las herramientas fundamentales en esta etapa fueron los softwares de diseño asistido por computadora, pues al mismo tiempo se buscaba explorar la forma de la estructura del robot, de tal manera que pudiera integrar todos los elementos necesarios.

Para la segunda versión (el prototipo B) ya se contaba con la integración de los sensores seleccionados para probar las funciones y los comportamientos del robot, además de que se reforzó su estructura para poder probar los algoritmos de navegación en este prototipo antes de implementarse en el modelo real, tratando de evitar mayores daños al cuerpo del robot real. En este caso se evaluó la navegación reactiva del robot con comportamientos simples, modificando de forma iterativa ciertos parámetros y condiciones buscando el mejor desempeño del mismo y que funcionara a manera de demostración para explicar los principios básicos de la navegación autónoma.

Para la versión actual del robot, como ya se ha mencionado, se fue desarrollando de forma modular, trabajando de forma paralela los diferentes sistemas con los que cuenta el robot,

cada sistema se evaluó de forma independiente; por ejemplo, en el sistema de la interfaz, se fueron evaluando el desempeño de cada una de las funciones del robot de forma independiente, corrigiendo errores, analizando y probando nuevas posibilidades.
En el caso de los algoritmos de navegación, primero tuve que preparar un espacio de pruebas virtual que me ayudara a comprobar el comportamiento de mis algoritmos a nivel simulación, por lo que me ayude de herramientas de simulación como gazebo y softwares de CAD para poder crear modelos virtuales, simularlos, y ver cómo se comportan ante diferentes situaciones. Esto me permitió identificar posibles errores, limitaciones y sobre todo, ajustar parámetros, cuando lograba un comportamiento aceptable a nivel simulación entonces verificar el desempeño de cada sensor haciendo pequeños *scripts* que me ayudaran a comprobar el funcionamiento de los mismos, para ello utilice algunas herramientas como Rviz o inclusive hasta la misma terminal, de esta manera descartaba posibles errores de medición que me complicaran la detección de fallas al implementarlo en el espacio de trabajo real, posteriormente integraba todos los elementos necesarios en el prototipo B y ponía a prueba su comportamiento en el espacio de trabajo, y finalmente integraba todo en el robot, ajustando algunos parámetros para mejorar su desempeño.
Algunos de los criterios de evaluación que utilicé para la navegación autónoma fue la exactitud con la que llegaba al destino, su comportamiento reactivo con obstáculos estáticos, su comportamiento reactivo con obstáculos dinámicos, la velocidad de navegación (sin perder precisión), la velocidad y precisión de localización, etc.
Cada función y cada algoritmo fue analizado de forma independiente, por lo que tener esta modularidad me ayudo a identificar más rápidamente los errores que surgieron y me permitió tratarlos de forma independiente, llegando más rápido a darles una solución.
Al ser escalable el proyecto cada que se quiere integrar una nueva función aplicamos el mismo proceso iterativo: diseño, simulación, pruebas en el prototipo, evaluación de desempeño en el modelo real, análisis y ajustes, aunque cabe recalcar que en algunos casos no es necesario probar las funciones antes en el prototipo; tal es el caso de las funciones de interacciones con la interfaz que no involucren movimiento, pues este tipo de funciones se pueden trabajar directamente sobre el modelo real ya que no comprometen la integridad del robot.

3. PRINCIPALES SISTEMAS

En esta sección presento los principales sistemas de los cuales está compuesto el proyecto, explicando tanto su función como los componentes que los integran.
De forma general, el robot cuenta con seis sistemas principales: el sistema de instrumentación, el sistema de procesamiento, el sistema de interfaz, el chasis o estructura, el sistema de locomoción y el sistema de alimentación. Estos sistemas trabajan en conjunto de forma sincronizada, permitiendo que el robot cumpla con el objetivo de forma eficiente y autónoma.

Nota: En la sección de anexos se puede encontrar un diagrama general de arquitectura realizado por mí en donde represente las interconexiones entre los sistemas del robot, además, de un diagrama eléctrico y finalmente un diagrama de conexiones realizados por mi compañero de trabajo para fines técnicos y de documentación.

3.1. INSTRUMENTACIÓN

Este sistema de encarga del reconocimiento del entorno. Son todas las entradas que recibe el robot y que determinaran su comportamiento, por lo que está formado por todos sensores como el LiDAR, los encoders y las cámaras, los cuales le permiten al robot obtener información de su espacio de trabajo u operación. Haciendo una analogía este sistema son los sentidos del robot (vista, oído, etc), sin este sistema el robot no tendría forma de saber lo que pasa en tiempo real en el mundo que lo rodea, por lo que no podría tomar decisiones en tiempo real y, por lo tanto, limitaría su autonomía y su funcionalidad.
Sus principales componentes son:

3.1.1. SENSOR LiDAR

El Slamtec RPLIDAR A1M8-R6 (Figura 14), es un sensor LiDAR de 360 grados diseñado para mapear, navegar y detectar obstáculos en tiempo real [12]. Este es uno de los principales sensores utilizados en automóviles autónomos. Este sensor dispara un láser a una gran velocidad en 360 grados (véase la Tabla 1 para ver los detalles técnicos), y calcula el tiempo que tarda en regresar para determinar la distancia a la que estas los objetos a su alrededor, lo que le permite cumplir con su objetivo.

Figura 14. Slamtec rplidar [12].

Tabla 1. Especificaciones del Slamtec rplidar A1 [12].

Especificaciones	Detalles
Rango de medición	0.15 – 12 m
Frecuencia de Escaneo	5.5 – 10 Hz
Campo de visión (FOV)	360°
Precisión	<1% del rango de medición
Resolución Angular	≤ 1°
Conexión	Serial (UART/USB)
Compatibilidad	ROS 2, Rviz
Voltaje de operación	5V
Consumo de Energía	2W (con motor)
Dimensiones	70 mm x 70mm x 41mm
Peso	170 g

3.1.2. CÁMARA DE PROFUNDIDAD

La *Intel®RealSense Depth Camera D435i* es una cámara (Figura 15), con un sensor RGB y sensores de profundidad, lo que permite tomar imágenes a color en 3D [13]. Su función principal está enfocada en el reconocimiento facial, seguimiento de objetos y la navegación autónoma.

La cámara tiene una estética moderna y no pesa mucho (véase la Tabla 2 para las especificaciones), además de que es compatible con ROS 2, por lo que es una buena opción para nuestro proyecto que está desarrollado en ROS 2 Humble; adicionalmente, tiene integrado un sensor IMU que nos podría ser útil para mejorar la localización y navegación del robot.

Figura 15. Cámara Intel RealSense D435i [14].

Tabla 2. Especificaciones Intel RealSense D435i [13].

Especificación	Detalles
Resolución	1280 x 720 (30fps)
Rango de profundidad	0.2 – 10 m
Campo de visión (FOV)	85.2° x 58°
Conexión	USB 3.0
Sensores Integrados	Sensor RGB, Sensores de Profundidad, IMU (6DOF).
Micrófonos	No
Compatibilidad	ROS 2, Rviz
Consumo de Energía	5V, 700 mA
Dimensiones	90 mm x 25mm x 25mm
Peso	70 g

Nota: Este componente también es compatible con sistemas en módulo (SOM) de la familia de Nvidia Jetson, lo que nos ayudaría al momento de integrar la inteligencia artificial.

3.1.3. CÁMARA

Esta cámara fue seleccionada debido a que la cámara de profundidad no era compatible con la placa Raspberry Pi 5; sin embargo, esta no queda descartada del todo, pues para que tenga un mejor desempeño podríamos utilizar más de una cámara, migrando los paquetes a una Jetson nano, por ejemplo. La Raspberry Pi Ai cam (Figura 16), tiene unas características técnicas inferiores a la cámara de profundidad (véase la Tabla3) [15]; sin embargo, es más pequeña y discreta, además de que está diseñada para trabajar con algoritmos de inteligencia artificial en la Raspberry Pi 5, por lo que es una buena opción. La cámara es capaz de procesar muchos datos sin sobrecargar la CPU principal, optimizando el rendimiento del sistema y reduciendo la latencia en el procesamiento de imágenes. Fue desarrollada en colaboración con Sony, por lo que incorpora el sensor Sony IMX500, que combina un sensor de imagen CMOS de 12,3 megapíxeles con capacidades de inferencia de inteligencia artificial integradas. Este módulo está especializado para trabajar con visión por computadora en tiempo real, por lo que podríamos ser capaces de hacer reconocimiento de objetos, seguimiento de movimiento y ayudarnos a mejorar la navegación autónoma [15].

Figura 16. Cámara Raspbererry Pi Ai cam [16].

Tabla 3. especificaciones cámara Raspberry Pi Ai cam [15].

Especificación	Detalles
Sensor	Sony IMX500
Resolución	12.3 megapíxeles (el valor original que da el fabricante es: 12,3)
Velocidad de fotogramas	Hasta 30 fps en modo 2×2 binned a 2028×1520; hasta 10 fps a resolución completa
Tamaño del sensor	7.857 mm (tipo 1/2.3, el valor original que da el fabricante es: 7,857)
Campo de visión (FOV)	Horizontal: 66° ±3°; Vertical: 52.3° ±3° (el valor original que da el fabricante es: 52,3° ±3°)
Apertura	f/1.79
Conexión	Interfaz de cámara en serie (CSI) de 15 pines
Sensores Integrados	Sensor RGB, Sensores de Profundidad, IMU (6DOF).
Compatibilidad	Raspberry Pi4 y 5, ROS 2, Rviz
Dimensiones	25 mm × 24 mm × 11.9 mm (el valor original que da el fabricante es: 25 mm × 24 mm × 11,9 mm
Peso	3-5 g

3.2. PROCESAMIENTO

Este sistema recibe la información de los sensores para procesarla y determinar el comportamiento de los actuadores, haciendo que el robot sea capaz de realizar la tarea que se le asigne. Análogamente, es el “cerebro” del robot. Sin este sistema el robot no tendría forma de interpretar las señales del sistema de instrumentación para completar sus tareas. Los principales componentes que lo integran son:

3.2.1. RASPBERRY PI 5

La Raspberry Pi 5 (Figura 17), ha sido seleccionada para el modelo por su capacidad de procesamiento, tamaño compacto y amplia compatibilidad con diversos sistemas operativos y software de robótica como ROS 2 y otras aplicaciones, sus características se presentan en la Tabla 4. El soporte para salida de video 4K y la amplia capacidad gráfica

permiten aplicaciones de alta resolución y calidad de imagen, mientras que su tamaño compacto y peso ligero facilitan la integración en el diseño del robot [17]. Actualmente, la Raspberry Pi 5 alberga el entorno de desarrollo y ejecución de ROS 2, siendo responsable de administrar los nodos, coordinar la lógica principal de los sistemas, y servir como enlace central entre los distintos módulos del robot, como la instrumentación, locomoción e interfaz con el usuario. Su eficiencia y bajo consumo energético la convierten en una opción ideal para aplicaciones móviles y embebidas como la del presente proyecto.

Figura 17. Raspberry Pi 5 [17].

Tabla 4. Especificaciones Raspberri pi 5 [17].

Especificación	Detalles
Procesador	Quad-core BCM2712, SoC de 64 bits y cuatro núcleos Cortex-A76 (ARM v8) a 2.4 GHz (el valor original que da el fabricante es: Quad-core BCM2712, SoC de 64 bits y cuatro núcleos Cortex-A76 (ARM v8) a 2,4 GHz)
Memoria RAM	8 GB LPDDR4X-4267
Almacenamiento	MicroSD (128 GB), soporte para almacenamiento USB
Conectividad	2 x USB 3.0, 2 x USB 2.0, USB-C, HDMI dual
Red	Gigabit Ethernet, Wi-Fi 802.11ac, Bluetooth 5.0
GPU	VideoCore VII 800 MHz (OpenGL ES 3.1, Vulkan 1.2), soporte 4K
Sistema Operativo	Raspberry Pi OS, compatible con otras distribuciones Linux, Windows 10 IoT, etc.
GPIO	40 pines
Compatibilidad	ROS 2, RViz, Gazebo, Nav2, etc.
Consumo de Energía	5 V / 3 A (15 W máx.)
Dimensiones	85 mm x 56 mm x 17 mm
Peso	46 g

3.2.2. ESP32

El ESP32 (Figura 18), es un microcontrolador desarrollado por la empresa *Espressif Systems*, que integra conectividad Wi-Fi y Bluetooth de modo dual. Este dispositivo puede gestionar el enlace y procesamiento de señales de movimiento de los motores, así como la lectura de señales de sensores como encoders [18]. Su arquitectura versátil y su amplia gama de interfaces periféricas facilitan la integración de múltiples componentes en sistemas robóticos. Las características de este dispositivo se presentan en la Tabla 5.

Figura 18. Microcontrolador ESP32 [19].

Tabla 5. Especificaciones ESP32 [18].

Especificación	Detalles
Procesador	32 bits Xtensa LX6 de dos núcleos, operando a 160 ó 240 MHz
Memoria	SRAM 520 KB, ROM 448 KB
Conexión	USB C
Conectividad	Wi-Fi 802.11 b/g/n y Bluetooth v4.2 BR/EDR y BLE
Protocolos de comunicación	Hasta 4 interfaces SPI, 2 interfaces I^2C, 3 interfaces UART, Ethernet MAC con DMA dedicado, CAN bus 2.0
No. Pines	34 GPIO, 18 canales ADC, 2 canales DAC, 16 PWM
Plataformas y lenguajes	Arduino IDE (C/C++), PlatformIO (C/C++), ESP-IDF (*Espressif IoT Development Framework*) (C/C++), MicroPython (Python).
Consumo de Energía	Activo: 160 mA, Suspensión ligera: ~2.5 mA, Suspensión profunda: ~5 µA
Dimensiones	~25.4mm x 50.8 mm (1 x 2 pulgadas)
Peso	~10g

3.3. INTERFAZ

El sistema de la interfaz de usuario (UI, por sus siglas en inglés) está formado por todos los elementos que hacen posible la interacción con el usuario. El objetivo principal de este sistema es lograr que la interacción con el robot sea más agradable y natural. Sin este sistema el robot no podría interactuar con los usuarios por lo que no se cumpliría el objetivo. Sus principales componentes son:

3.3.1. *DISPLAY*

Se sabe que, en una conversación, la mayor parte de la información se recibe mediante la comunicación no verbal. Por ello el principal componente de la interfaz es la pantalla portátil WIMAXIT 10.1 (Figura 19), un *display* de 10.1 pulgadas que muestra animaciones e interfaces para interactuar con el usuario, este componente fue seleccionado principalmente por su tamaño, ya que quedaba perfecto con el tamaño del robot. Adicionalmente, está diseñada para trabajar con la Raspberry Pi, por lo que no tenemos problemas de compatibilidad. Cuenta con una resolución Full HD de 1024x600 píxeles, lo que nos permite tener una interfaz con imágenes nítidas y detalladas; además, tiene integrada la función táctil, lo que nos permite ampliar las opciones de interacción entre el robot y el usuario, facilitando el control y la programación, en la Tabla 6 pueden verse todas sus especificaciones [20].

Figura 19. Pantalla portátil WIMAXIT 10.1 [20].

Tabla 6. Tabla de especificaciones pantalla portátil WIMAXIT 10.1 [20].

Especificación	Detalles
Tamaño	10.1 in
Resolución	1024 x 600 IPS
Relación	16:9
brillo	400 cd/m²
Ángulo de Visión	178°
Tiempo de Respuesta	5 ms
Panel Táctil	Sí
Conexión	HDMI, USB-C
Compatibilidad	Raspberry Pi, PC, Laptop, Consolas.
Sistemas Operativos	Windows, Linux, macOS, Android
Alimentación	5V/2A (USB-C)
Consumo de Energía	10W (máx.),
Dimensiones	257 mm x 168 mm x 9 mm
Peso	600 g

3.3.2. ALTAVOZ

El usuario también debe de interactuar con el robot escuchando información por ello el robot integra el Dayton Audio PC68-8 (Figura 20), que es un controlador de altavoz de rango medio de 2.5” [21]. Es ideal para aplicaciones de audio en proyectos de robótica y automatización que requieren una alta calidad de sonido en un formato compacto. Este altavoz destaca por su eficiencia y capacidad para manejar altas potencias sin distorsión. Todas sus características se pueden ver en la Tabla 7.

Figura 20. Dayton audio PC68-8 [21].

Tabla 7. Tabla de especificaciones pantalla portátil dayton audio PC68-8 [21].

Especificación	Detalles
Diámetro	2.5 pulgadas (65 mm)
Impedancia	8 ohms
Potencia RMS	20W
Potencia Máxima	40W
Sensibilidad	85.6 dB @ 2.83W/1m
Respuesta de frecuencia	110 Hz - 17000 Hz
Material del Cono	Fibra de vidrio tejido
Iman	Ferrita
Consumo de Energía	2W
Dimensiones	73D mm x 37mm
Peso	200 g

3.3.3. AMPLIFICADOR

El amplificador para el sistema de audio fue seleccionado con base en las especificaciones de los altavoces, es un Makerhawk mini (Figura 21), este amplificador brinda la potencia de salida suficiente en cada canal con un margen de seguridad para evitar dañar los componentes, además, tiene una conectividad versátil y diseño compacto (ver la Tabla 8). Este dispositivo proporciona una amplificación clara y potente para entornos un tanto ruidosos como el lugar de operación. El Bluetooth 5.0 permite una integración sencilla con diversos dispositivos, también cuenta con una entrada Aux-in (3.5 mm) y es compatible con altavoces de 4-8 ohmios. Ofrece poca distorsión en relación con el volumen y se adapta a diferentes fuentes de alimentación [22].

Figura 21. Makerhawk mini amplificador 2x50W [22].

Tabla 8. Tabla de especificaciones pantalla portátil Makerhawk mini amplificador 2x50W [22].

Especificación	Detalles
Potencia de Salida	2 canales, 50W
Impedancia	4-8 ohms
Distorsión Armónica Total (THD)	<0.04%
Relación Señal-Ruido (SNR)	≥ 98 dB
Respuesta de frecuencia	20 Hz - 20,000 Hz
Conectividad	Bluetooth 5.0, Aux-in (3.5mm)
Voltaje de Operación	DC 5V - 24V
Dimensiones	113 mm x 69mm x 49mm
Peso	50 g
Funcionalidad Adicional	Control de tono (graves y agudos), Protección contra sobrecarga

3.4. CHASIS/ESTRUCTURA

El chasis o la estructura, es el cuerpo físico del robot; es lo que le da forma y está diseñado para soportar el peso de todos los componentes, tanto electrónicos como mecánicos, lo que le permite mantener fijos a los componentes durante el movimiento y que se vea estéticamente llamativo, manteniendo a salvo del exterior a los demás sistemas. A lo largo del desarrollo del proyecto, el chasis ha sufrido una gran variedad de cambios, pues las primeras versiones estaban construidas en MDF cortado con láser con un diseño totalmente diferente al diseño final, después se cambió el MDF por placas de acrílico, lo que hizo que fuera más modular y versátil; sin embargo, no fue hasta que se hizo una propuesta de diseño más robusta y profesional, que se logró llegar a el resultado actual.

3.4.1. CARCASA DE FIBRA DE VIDRIO

Para la versión final, se decidió fabricar la carcasa externa en fibra de vidrio (Figura 22), ya que es un material conocido por ser ligero, resistente, impermeable, elegante y

fácilmente moldeable. Esto nos permitió tener una carcasa más duradera y resistente a las condiciones del entorno al momento de hacer las pruebas, además de que le dio una apariencia más profesional al robot. La fabricación fue realizada por un equipo especializado en el manejo de este material, lo que garantizó un acabado de alta calidad adaptado a las dimensiones y necesidades específicas del robot.

Figura 22. Carcasa en fibra de vidrio mandada a hacer.

3.4.2. ESPECIFICACIONES

Peso desglosado (gramos)

Base (completo): 5390 g

- motores, batería, drivers, ESP32, tablero de carga, relevador, soportes, tapas, cableado y tornillería.

Ruedas: 3714 g

- rueda DI: 925 g
- rueda TI: 933 g
- rueda DD: 928 g
- rueda TD: 928 g

Cabeza (completo): 2310 g

- bocinas, pantalla, tapas y caratulas, cables (audio, HDMI, alimentación pantalla).

Cuerpo (completo): 4080 g

- Raspberry Pi 5, regulador, Raspberry Pi AI cam, amplificador, LiDAR, lampara led, cableado, tapas, soportes (cámara, cuello, LiDAR).

Total, robot host financiero: 15494 g

Peso resumido (en bascula corporal)

9.4 Kg base y ruedas
6.4 Kg cabeza y cuerpo
Total:15.6 Kg completo

Dimensiones generales (con ruedas):

- Altura total: 82 cm
- Ancho total: 48 cm
- Profundidad total: 39.6 cm

Base:

- Con ruedas:
  - Altura: 18 cm
  - Ancho: 48 cm
  - Profundidad: 39.6 cm

- Sin ruedas:
  - Altura: 11.5 cm
  - Ancho: 34.5 cm
  - Profundidad: 35 cm

Cuerpo del robot:

- Altura: 40 cm
- Ancho: 32 cm
- Profundidad: 25 cm

Cabeza del robot:

- Altura: 29.5 cm
- Ancho: 30 cm
- Profundidad: 20.5 cm

Parámetros de distribución física:

- Distancia entre ejes (delantero y trasero): 21 cm
- Distancia entre ruedas (izquierda a derecha, de centro a centro): 43 cm
- Diámetro real de rueda: 18.68 cm

Centro de gravedad aproximado:

- Altura desde el suelo: 26.7 cm

Ubicación de componentes:

- Altura de la cámara: 56 cm
- Altura de lectura del sensor LiDAR: 22 cm

Nota: La posición relativa de las ruedas respecto al cuerpo se encuentra centrada en el eje longitudinal del robot, equidistante respecto al centro geométrico de la base.

3.5. LOCOMOCIÓN/PROPULSIÓN

Este sistema incluye los motores, llantas y todos los elementos que permiten el desplazamiento del robot. En el robot host financiero, se optó por una configuración diferencial con cuatro llantas, con el objetivo de soportar todo el peso del robot y al mismo tiempo tener una buena maniobrabilidad de conducción. Sin este sistema el robot sería incapaz de moverse, sus principales componentes son:

3.5.1. MOTORES

El motorreductor JGB37-545 (Figura 23), tiene un alto rendimiento y está diseñado para aplicaciones que requieren una gran potencia y torque en un formato compacto. Este tipo de motorreductor es ideal para robots móviles y otras aplicaciones de automatización que demandan precisión y durabilidad. Cuenta con un engranaje metálico que proporciona una relación de reducción de 131:1, lo que permite un control preciso del movimiento a bajas velocidades. Además, incluye un encoder para monitoreo preciso de la posición y la velocidad del eje [23]. Las especificaciones del mismo junto con las de su encoder integrado pueden verse en la Tabla 9.

Figura 23. Motorreductor con enconder JGB37-545 [23].

Tabla 9. Especificaciones Motorreductor con enconder JGB37-545 [23].

Especificación	Detalles
Relación de reducción	131:1
Velocidad sin carga	95 RPM
Par nominal	20 Kg*cm
Par máximo	35 Kg*cm
Corriente sin carga	<0.4 A
Corriente máxima	5 A (bloqueo no permitido)
Voltaje nominal	24 V DC
Potencia de salida	17W
Rotación	bidireccional
Eje	6 mm de diámetro, 15.5 mm de longitud
Dimensiones	37D mm x 115 mm
Peso	~280 g
Encoder	
Tipo	Incremental, magnético de cuadratura
Resolución	16 CPR (cuentas por revolución), 2 canales

Nota: Las especificaciones pueden variar ligeramente según el fabricante y la configuración específica del motor. Es recomendable consultar la documentación del fabricante.

3.5.2. PUNETE H

El Puente H BTS7960 (Figura 24), ha sido seleccionado por su capacidad para manejar altas corrientes (hasta 43A por canal) y voltajes de 6V a 27V, haciéndolo ideal para

motores de alta potencia como los de 24V con encoder utilizados en este proyecto, sus especificaciones se pueden encontrar en la Tabla 10. Un aspecto clave en su selección es su protección integrada contra sobre corriente, sobrecalentamiento y cortocircuito, que garantiza una operación segura. Además, su control mediante modulación por ancho de pulso (PWM) permite regular de manera precisa de velocidad y dirección [24].

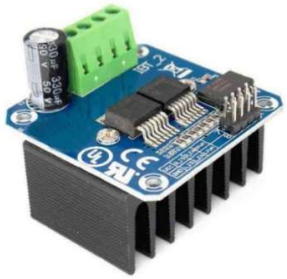


Figura 24. Puente H BTS7960 [22].

Tabla 10. Especificaciones puente H BTS7960 [22].

Especificación	Detalles
Voltaje de Operación	6V - 27V
Corriente Máxima de Salida	43A
Canales	2 (1 motor dos direcciones)
Protección	Sobrecorriente, Sobretensión, Sobrecarga, Cortocircuito
Método de Control	PWM
Dimensiones	50 mm x 50 mm x 30 mm
Peso	70 g

3.5.3. RUEDAS

Estas ruedas de 180 mm de diámetro se seleccionaron para nuestra aplicación debido a su tamaño (Figura 25), que se encuentra dentro del rango propuesto de 160 a 200 mm de diámetro (ver Tabla 11) [25]. Este tamaño es ideal para mantener una adecuada proporción en el diseño dando visibilidad suficiente a estos elementos. Además de cumplir, estas ruedas están diseñadas para soportar cargas mucho mayores a las del robot, proporcionando durabilidad y rendimiento.

Figura 25. Rueda para vehículo no tripulado AGV 180mm [25].

Tabla 11. Especificaciones rueda para vehículo no tripulado AGV 180mm [25].

Especificación	Detalles
Diámetro	180 mm
Material de la llanta	Fundición de aluminio
Ancho de la rueda	45 mm
Capacidad de carga	200 kg
Compatibilidad con eje	6 mm
Tipo de montaje	Montaje en eje
Dimensiones	180D mm x 45 mm
Peso	0.84 Kg

3.6. ALIMENTACIÓN

Proporciona la energía necesaria para todos los sistemas anteriores. Incluye baterías recargables y circuitos de distribución de energía cuidadosamente seleccionados para asegurar autonomía y seguridad operativa.

3.6.1. BATERÍA

La batería es de polímero de litio diseñada para aplicaciones que requieren una gran demanda de energía, como drones, vehículos RC y robots (Figura 26). Su alta densidad energética y tasa de descarga la hacen ideal para proyectos que necesitan un suministro constante y robusto de energía [26]. La capacidad fue determinada de acuerdo con el consumo de energía de los componentes eléctricos y electrónicos para operar durante 2

horas de funcionamiento continuo. Se desea que el peso sea lo menor posible, y como vemos en la Tabla 12 junto con las demás especificaciones, no es tan pesada como otras del mercado.

Figura 26. Batería LIPO 6s 22.2v 16000mah 25c [26].

Tabla 12. Especificaciones batería LIPO 6s 22.2v 16000mah 25c [26].

Especificación	Detalles
Tipo	LiPo (polímero de Litio)
Voltaje	22.8V (6s)
Capacidad	16000 mAh
Tasa de descarga	25C
Conexión	XT90s
Dimensiones	195 mm x 76 mm x 66 mm
Peso	~1.83 Kg
Ciclo de vida	Mas de 150 ciclos

3.6.2. REGULADOR DE VOLTAJE

El regulador de voltaje DROK con *display* (Figura 27), fue seleccionado para la alimentación de los componentes electrónicos del robot, nos brindará un voltaje estable aislando el circuito eléctrico de los controladores y sensores evitando en gran medida la repercusión de los picos de corriente provocados por los motores en el sistema. El regulador tiene la ventaja de poseer un *display* digital que muestra el voltaje de entrada y salida, permitiendo un ajuste preciso y monitoreo en tiempo real sobre el consumo de los componentes. Acepta una entrada de 6V a 55V y ofrece una salida de 0V a 50V con una corriente máxima de 8 (Tabla 13), lo que es adecuado para los requisitos de potencia de la cámara, las tarjetas controladoras y los sensores [27].

Figura 27. Regulador de voltaje DC DROCK [27].

Tabla 13. Especificaciones regulador de voltaje DC DROCK [27].

Especificación	Detalles
Rango de Voltaje de Entrada	6 - 55V
Rango de Voltaje de Salida	0 - 50V
Corriente de Salida Máxima	8A
Potencia de Salida Máxima	400W
Eficiencia	Hasta 90%
Display	Digital, muestra voltaje de entrada/salida y corriente
Protección	Sobretensión, sobrecorriente, sobretemperatura
Dimensiones	79 mm x 43 mm x 45 mm
Peso	130 gramos

3.6.3. MONITOR DE BATERÍA

Debido al constante cambio de voltaje de la batería en cada ciclo de carga y descarga, se requiere de un dispositivo de monitoreo que nos permita supervisarla asegurando un mejor rendimiento y prolongando su vida útil al indicar si la tensión a caído lo suficiente para ocasionar algún daño en las celdas. El monitor de batería SUPERNOVA (Figura 28), está diseñado para proporcionar información precisa sobre el estado de carga y el voltaje de baterías tiene capacidad para medir voltajes desde 8V hasta 100V (Tabla 14). Su pantalla LCD ofrece una visualización clara de los datos de la batería como su voltaje, temperatura y porcentaje, además, cuenta con diversas alarmas de voltaje que pueden ser configuradas según la necesidad [28]. El diseño es compacto con protección a salpicaduras y puede ser fácilmente montado sobre algún tablero.

Figura 28. Monitor de batería digital SUPNOVA [28].

Tabla 14. Especificaciones rueda para vehículo no tripulado AGV 180mm [28].

Especificación	Detalles
Rango de Voltaje Medible	8V - 100V
Compatibilidad de Baterías	12V, 24V, 36V, 48V, 60V, 72V, 84V
Pantalla	LCD
Funciones Adicionales	Alarma sonora (buzzer), medición de temperatura
Precisión de Medición	±1%
Frecuencia de Actualización	2Hz
Consumo de Energía	Corriente de trabajo < 15 mA
Dimensiones	61.3 mm × 33.3 mm × 13.5 mm
Peso	~30g
Protección	IPX7 (resistente al agua)

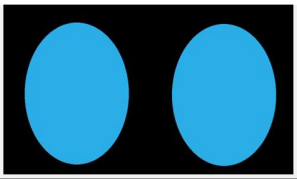
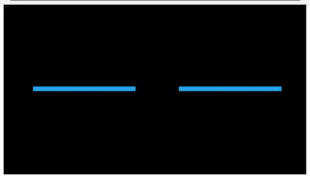
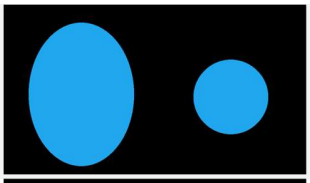
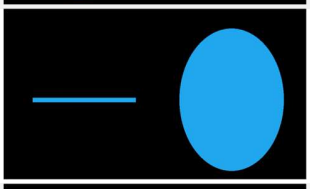
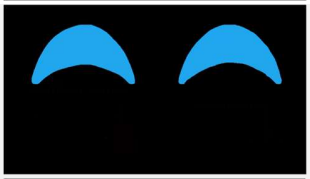
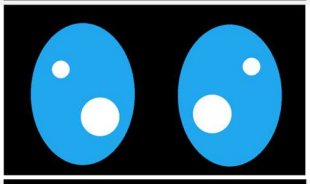
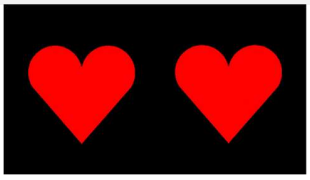
4. FUNCIONES

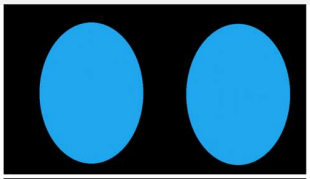
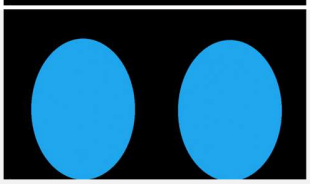
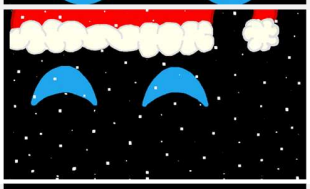
Después de haber hablado sobre los diferentes sistemas que conforman al robot host financiero y los componentes que los integran; sin embargo, ¿Cómo estos sistemas interactúan entre sí para poder cumplir con el objetivo planteado? En esta sección se abordará la integración de los sistemas, enfocándonos en las funciones específicas que con las que cuenta actualmente el robot.

4.1. INTERACCIONES MANUALES

Actualmente, el robot dispone de 54 frases y 13 animaciones que pueden activarse manualmente, ya sea mediante un control remoto o presionando teclas específicas para facilitar la interacción con los usuarios. Estas expresiones visuales y sus transiciones están compuestas por fotogramas creados con una técnica de píxel art, a una resolución de 1024x600 píxeles, que corresponde al *display* del monitor portátil conectado a la Raspberry Pi 5. Al ejecutarse de forma secuencial a una velocidad determinada (60 FPS), los fotogramas generan un efecto de movimiento que da lugar a las animaciones visibles en pantalla. El código, fue desarrollado en Python y está diseñado para crear, gestionar y ejecutar las animaciones del robot de forma remota. El enfoque principal del programa consiste en almacenar los distintos *frames* (imágenes) en listas, y reproducir porciones específicas de esas listas dependiendo de la animación que se quiera mostrar. Cada animación se activa mediante funciones que determinan cual segmento de la lista de imágenes debe reproducirse al detectar una tecla o comando determinado. A continuación, se detallan las animaciones disponibles en la Tabla 15:

Tabla 15. Animaciones del robot.

#	Nombre	Duración [s]	Tecla/comando asignada/o	Cantidad de *frames*	Animación
1	Parpadeo	0.46	p	7	
2	Sleep	1.4	m	21	
3	Confusión	1.73	c	26	
4	Guiño	0.66	g	10	
5	Felicidad	1.29	f	19	
6	Emoción	1.13	e	17	
7	Tristeza	1.33	u	20	
8	Amor	2.93	q	44	

9	Estrella	1.93	space	29	
10	Negación	0.66	d	10	
11	Afirmación	0.53	a	8	
12	Navidad	2.6	x	39	
13	Día de muertos	1.93	w	29	

4.2. INTERACCIÓN PASIVA

Además de poder activar las interacciones manualmente, el robot cuenta con una función de interacción pasiva, esto con el objetivo de darle vida aun cuando no está interactuando con ningún usuario. De tal forma que, cuando el robot no interactúa con algún usuario en 5 segundos, entra en un estado de espera o en el modo "*idle*", en este estado el robot parpadeara cada 5 segundos, como si fuera un ser humano. Adicionalmente, para evitar que esto se vea repetitivo, después de que haya pasado un rango entre veinte segundos y un minuto, el robot ejecutará aleatoriamente alguna de las animaciones predefinidas en una variable. Cada animación tiene una probabilidad predefinida de ser seleccionada, la cual se muestra en la Tabla 16, lo que nos permite controlar que animaciones son más probables a ser seleccionadas sin perder ese toque de espontaneidad.

Tabla 16. Probabilidades del estado idle.

#	Nombre	Probabilidad [%]
1	Parpadeo	10
2	Sleep	10
3	Confusión	7
4	Guiño	7
5	Felicidad	5
6	Emoción	5
7	Tristeza	5
8	Amor	1
9	Estrella	1

4.3. COREOGRAFÍAS INTEGRADAS

El robot también cuenta con algunas coreografías temáticas integradas, estás fueron diseñadas y preprogramadas para hacer que el robot sea más llamativo en sus presentaciones e interacciones con los usuarios. Actualmente, el robot puede ejecutar tres coreografías temáticas distintas, una con temática navideña, otra con temática de día de muertos y una más movida con temática pop.

Cada una de estas coreografías ha sido diseñada para coordinar movimientos físicos, animaciones visuales y reproducción de audio en perfecta sincronía, permitiendo mostrar el potencial del robot host financiero a los espectadores como un robot interactivo, versátil y personalizable.

4.4. CONTROL REMOTO

El robot puede ser controlado a distancia de 3 formas diferentes: la primera es por medio de la computadora, presionando las teclas del teclado, la segunda, a través de una aplicación móvil por conexión Bluetooth, y la tercera es mediante un mando de videojuegos. Si es controlado por medio de la computadora tendremos un control completo

sobre todas las funciones del robot, incluyendo el movimiento, las animaciones, la reproducción de frases, el botón de paro de emergencia, la navegación, etc. Y además tendremos la opción de visualizar en tiempo real todos los procesos que ocurren con el robot. Por otro lado, con el modo Bluetooth, igualmente podremos tener un control total sobre las funciones del robot; sin embargo, no contaremos con la retroalimentación de ver que es lo que está ocurriendo con los nodos en tiempo real. Todas las formas de controlar al robot son útiles en diferentes contextos, ya que una me ofrece más retroalimentación para el análisis, otra más facilidad de uso y otra más portabilidad, esto me facilito tanto el desarrollo como la interacción en presentaciones y entornos controlados.

4.5. CONTROL DIFERENCIAL EN ROS 2

Como ya mencionamos hay 3 formas de controlar el robot a distancia, y todas están sincronizadas con ROS2 utilizando un control diferencial; por ejemplo, en el caso de la computadora el robot puede ser operado mediante un sistema de teleoperación, empleando un *plugin* de control diferencial. Esta función está disponible tanto para el modelo simulado en Gazebo como para el robot físico real, permitiendo controlar las velocidades de los motores a través de comandos de teclado que publican mensajes en el tópico estándar “/cmd_vel”.

Este enfoque es ideal para pruebas de navegación, integración con algoritmos de SLAM y validación de comportamiento autónomo. En la Figura 29, se muestra la interfaz del control por teclado utilizando el nodo “teleop_twist_keyboard”, junto con una breve explicación de sus comandos.

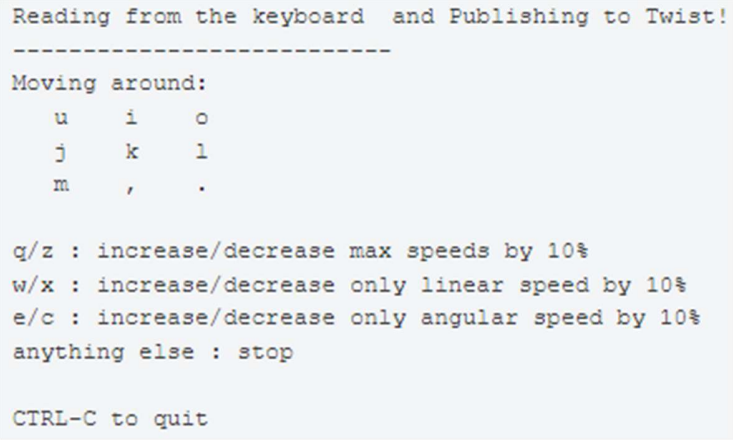


Figura 29. Interfaz de control diferencial de teleop_twist_keyboard.

u	→	Avanzar haciendo un círculo hacía adelante y a la izquierda.
i	→	Avanzar hacia adelante.
o	→	Avanzar haciendo un círculo hacía adelante y a la derecha.
j	→	Girar sobre su propio eje hacía la izquierda.
K	→	Detenerse
l	→	Girar sobre su propio eje hacía la derecha.
m	→	Avanzar haciendo un círculo hacía atrás y a la izquierda.
,	→	Avanzar hacia atrás.
.	→	Avanzar haciendo un círculo hacía atrás y a la derecha.
q	→	Aumentar en 10 % la velocidad máxima.
z	→	Disminuir en 10 % la velocidad máxima.
w	→	Aumentar en 10 % la velocidad lineal.
x	→	Disminuir en 10 % la velocidad lineal.
e	→	Aumentar en 10 % la velocidad angular.
c	→	Disminuir en 10 % la velocidad angular.
CTR + C	→	Salir de la interfaz

Este método de control complementa el control de forma remota, integrando al robot completamente en el ecosistema de ROS 2 para pruebas de simulación, navegación autónoma y validación de comportamiento en tiempo real.

4.6. FUNCIONES DE SEGURIDAD BÁSICAS

Con el objetivo de garantizar un entorno de pruebas seguro y proteger el robot, este incorpora funciones básicas de seguridad tanto a nivel físico como lógico.
Entre estas medidas se encuentra la implementación de un botón físico *on/off*, el cual permite desactivar el robot en caso de emergencia o al finalizar una sesión de pruebas.
Adicionalmente, se han integrado comandos de paro inmediato desde los controles remotos, ya sea mediante interacción por teclado o a través del control por Bluetooth, lo que permite detener rápidamente cualquier acción en curso ante un comportamiento no deseado, asegurando así un mayor control durante la operación del robot.

4.7. INTERFACES DE COMUNICACIÓN Y TELEOPERACIÓN

El robot host financiero cuenta con múltiples interfaces de comunicación que permiten su operación remota, monitoreo y programación desde distintas plataformas, adaptándose a diferentes necesidades de desarrollo y control:

- VNC (*Virtual Network Computing*): Permite el acceso remoto al entorno gráfico de la Raspberry Pi 5 por medio de wifi, facilitando la visualización y manipulación directa de la interfaz visual del sistema operativo y las aplicaciones en ejecución.
- SSH (*Secure Shell*): Esta conexión nos permite acceder a la terminal de la Raspberry Pi 5 de forma segura por medio de wifi, lo que nos permite ejecutar comandos, gestionar archivos y controlar los procesos sin necesidad de una interfaz gráfica.
- Comunicación serial: Utilizada principalmente para la programación directa de microcontroladores y monitoreo de datos en tiempo real durante las pruebas.
- Bluetooth: Habilita el control remoto desde la aplicación móvil o el mando, esto nos permite controlar mejor al robot y movernos físicamente de una forma más fácil.

Estas interfaces nos permiten tener una gran variedad de opciones para controlar al robot facilitando el desarrollo y el control de este en diferentes contextos como ya lo vimos en la sección 4.4 del control remoto.

4.8. VISUALIZACIÓN DE DATOS EN TIEMPO REAL

Un punto muy importante es que podemos ver los datos de entrada del robot y como es que se van ejecutando los procesos en tiempo real, la principal herramienta que nos permite ver los datos de los sensores en tiempo real es *ROS Visualitation (Rviz)*. Esto nos facilita mucho el análisis y la depuración de los datos provenientes de cada sensor, además, de que nos permite ver al mundo tal cual como lo ve el robot.

4.9. SIMULACIÓNES REALISTAS

Es posible modelar al robot en mundos virtuales realistas, que nos permiten someter al robot a condiciones parecidas a las de nuestro entorno físico, pero en una simulación antes de ponerlo en práctica en el entorno de trabajo. Esto permite realizar pruebas, validar funciones y evitar comportamientos no deseados sin comprometer la integridad física del robot. Inicialmente, cree una descripción genérica del robot utilizando solo figuras primitivas, como cubos, cilindros y esferas (ver Figura 30). Cada figura representa de forma genérica, las dimensiones reales del robot físico. Esta descripción del robot la hice en el paquete de descripción del robot, el cual centraliza los archivos URDF y los elementos visuales necesarios para instanciar robot.

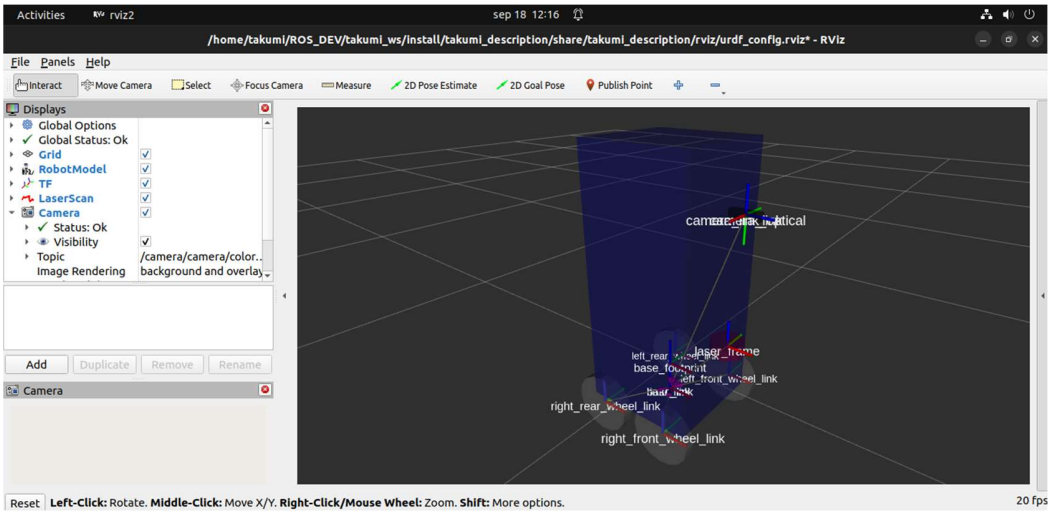


Figura 30. Modelo del robot con figuras primitivas.

Posteriormente, cree un mundo virtual personalizado en el simulador Gazebo (Figura 31). En este mundo coloque diferentes objetos y obstáculos, de tal forma que me ayudaron a analizar y explicar el comportamiento del robot, además, de detectar y explicar las limitaciones del robot en entornos reales, como la altura de los objetos tal es el caso de las mesas o sillas.

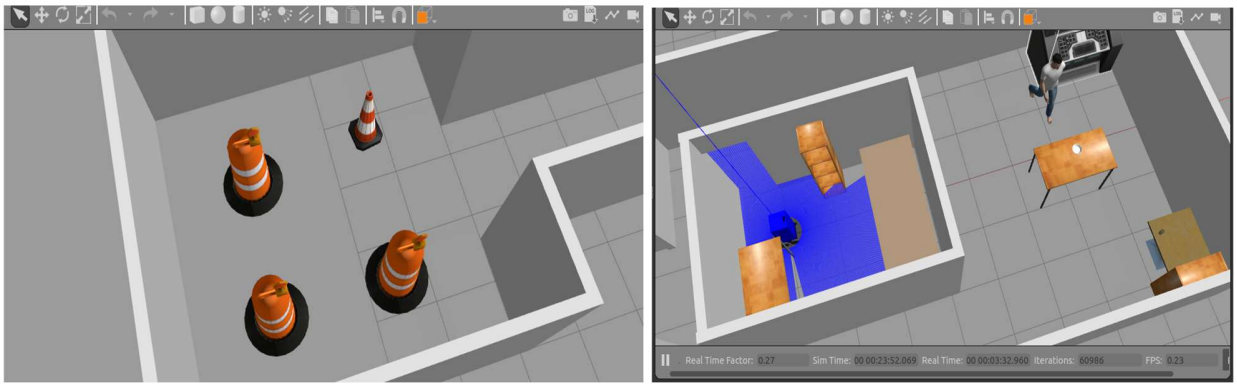


Figura 31. Espacio virtual de Gazebo.

Antes de utilizar los sensores físicos, integré *plugins* de simulación en Gazebo para emular el funcionamiento de dispositivos como el LiDAR, la cámara y la unidad de medición inercial (IMU). Esto me permitió observar la interacción entre los sensores, el rendimiento

del sistema de percepción, y la sincronización de datos en tiempo real. La información generada en la simulación es visualizada a través de herramientas como Rviz, donde pueden observarse los datos de los sensores virtuales, tales como las líneas de escaneo del LiDAR y las imágenes capturadas por la cámara simulada (ver Figura 32).

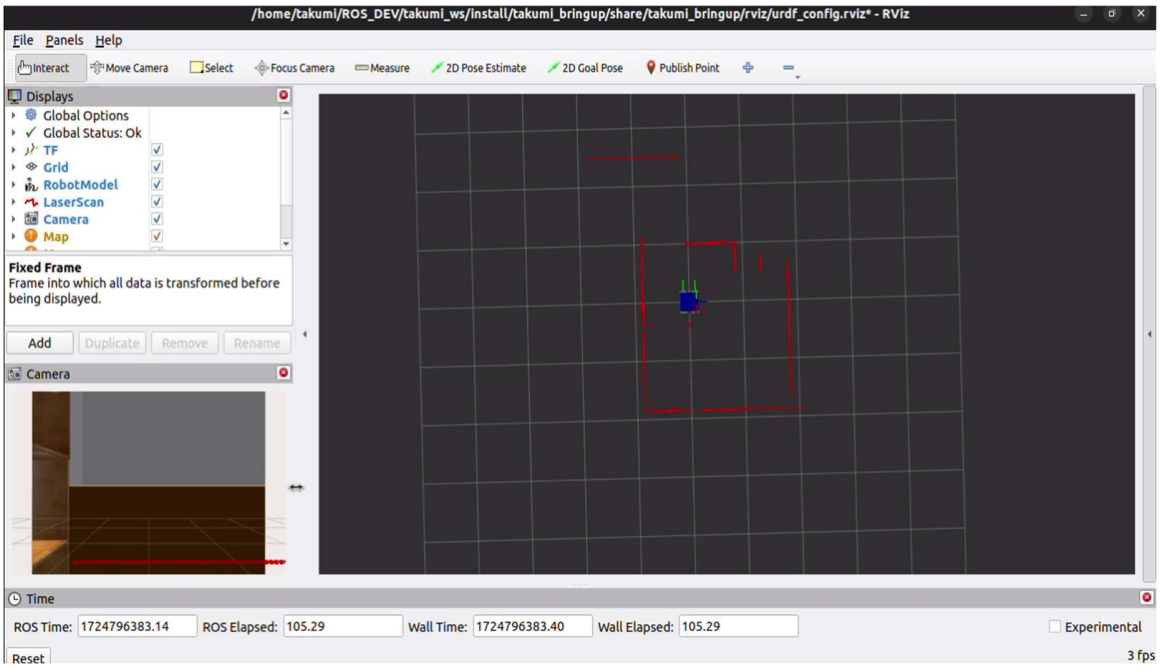


Figura 32. Visualización de los datos recibidos de la simulación en tiempo real

Una vez validado el funcionamiento inicial con figuras primitivas, desarrollé una descripción más precisa del robot basada en el modelo CAD original, utilizando el paquete de descripción del modelo. Esta nueva versión del modelo fue exportada a formato URDF y cuenta con una geometría detallada derivada directamente del diseño 3D realizado en Fusion 360 (Figura 33), lo cual permite simulaciones más realistas en gazebo y una representación visual más exacta en Rviz (Figura 34).

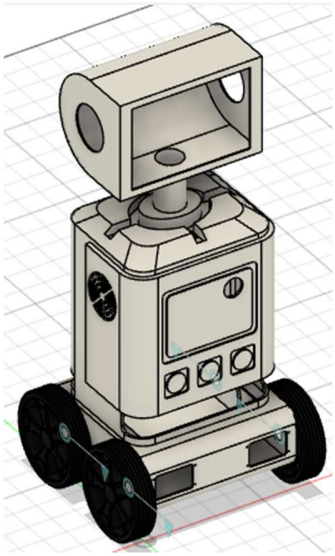


Figura 33. Modelo CAD creado en Fusión.

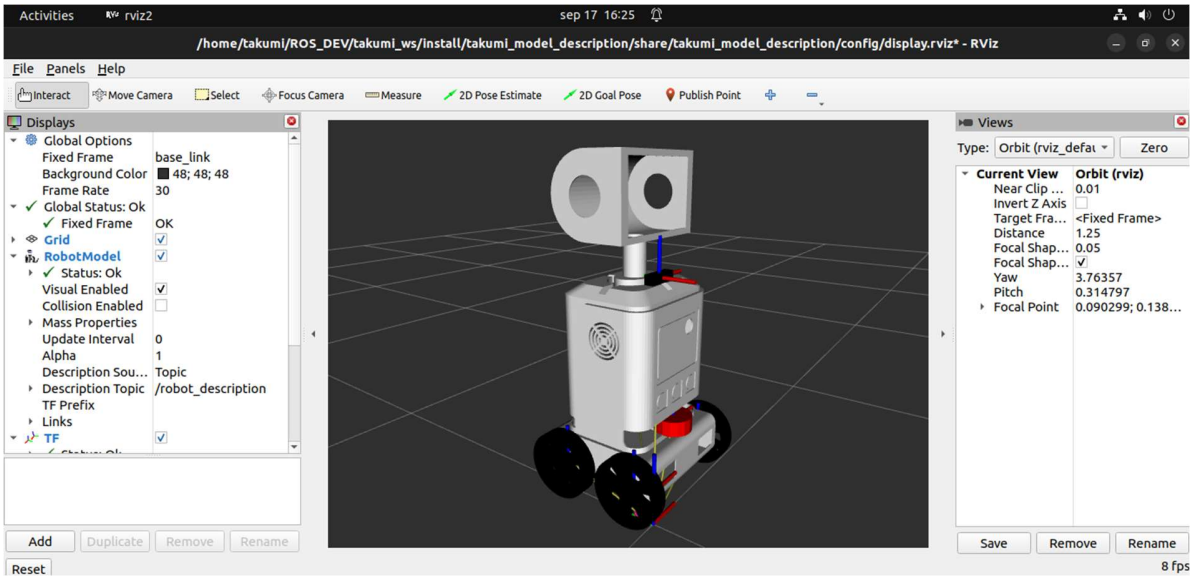


Figura 34. CAD del modelo en Rviz.

La simulación realista del robot no solo facilita la verificación de sus capacidades antes de la implementación física, sino que también agiliza el desarrollo y la depuración del sistema, contribuyendo a un proceso de integración más robusto y seguro. En la Figura 35 podemos ver este modelo interactuando con el espacio virtual previamente generado.

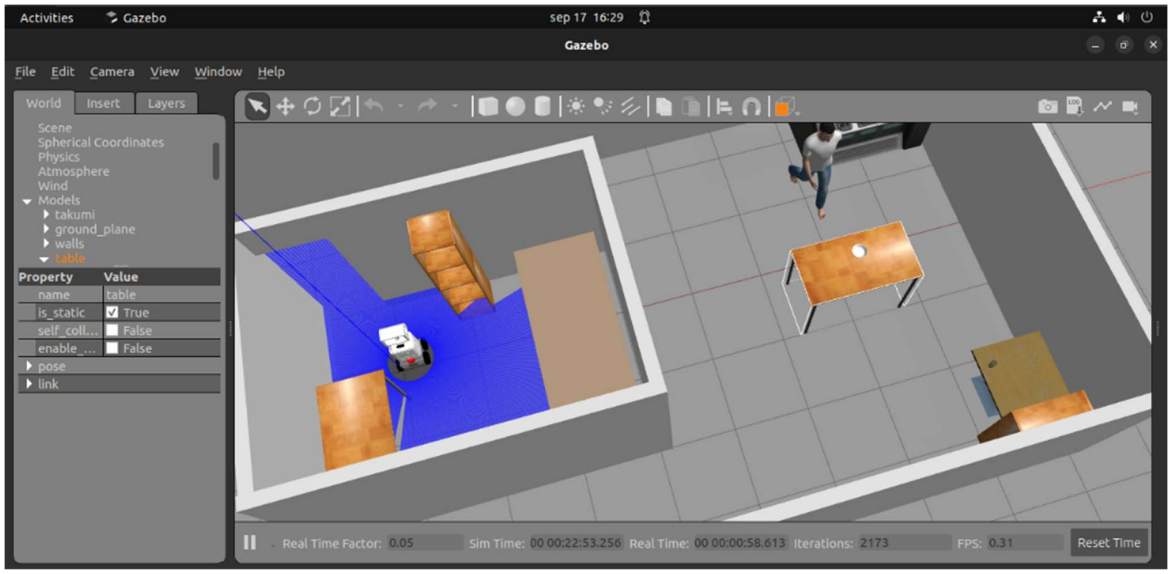


Figura 35. CAD del modelo en la simulación de gazebo.

4.10. GENERACIÓN DE MAPAS

El robot es capaz de generar mapas de entornos desconocidos por medio de algoritmos SLAM, los cuales permiten al robot construir un mapa del entorno al mismo tiempo que determina su posición dentro de él (ver Figura 36 y 37). Este proceso se realiza en tiempo real durante una exploración del espacio, la cual puede llevarse a cabo a través del control remoto. Los algoritmos SLAM combinan la información proveniente de diferentes sensores, como el LiDAR, los encoders y la IMU, para determinar las características del entorno, lo que le permite reconocer su entorno y determinar la distancia a la que están los objetos a su alrededor de forma más eficiente y autónoma. Gracias a esto, el robot puede:

- Reconocer nuevos espacios sin intervención humana directa.
- Adaptarse a cambios en la disposición del entorno.
- Planificar trayectorias más eficientes, reduciendo el consumo energético.
- Mejorar su precisión en la ejecución de tareas.

La integración de algoritmos SLAM en el robot son fundamentales para la autonomía y eficiencia de los robots móviles diferenciales, permitiéndoles operar de manera efectiva en entornos dinámicos y desconocidos.

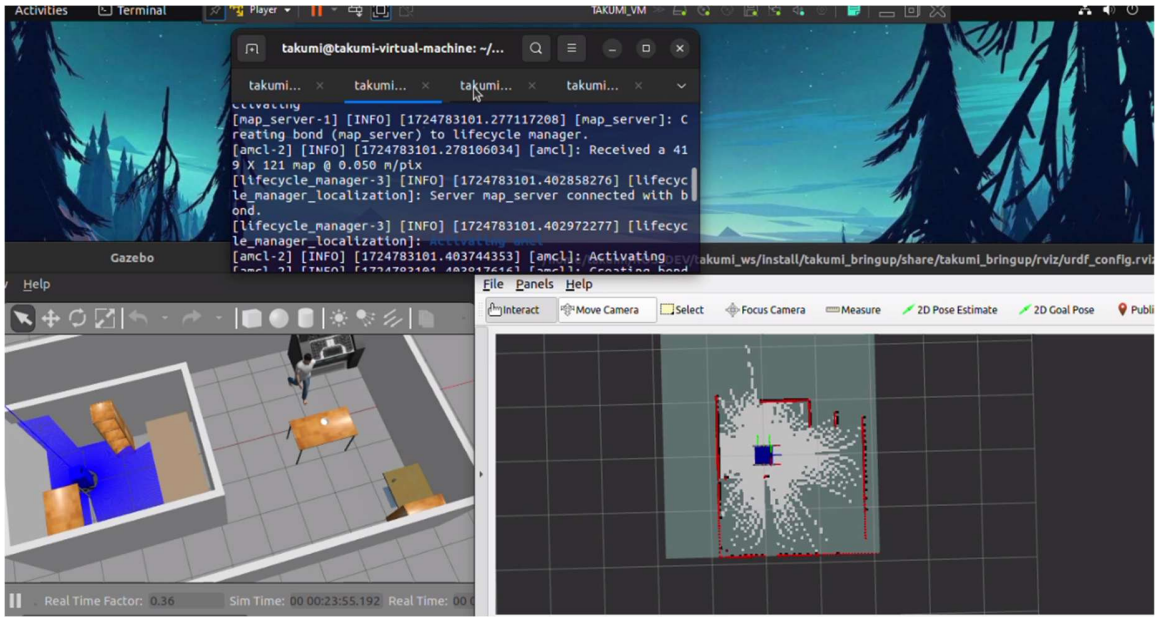


Figura 36. Rviz con algoritmos SLAM activados en la simulación.

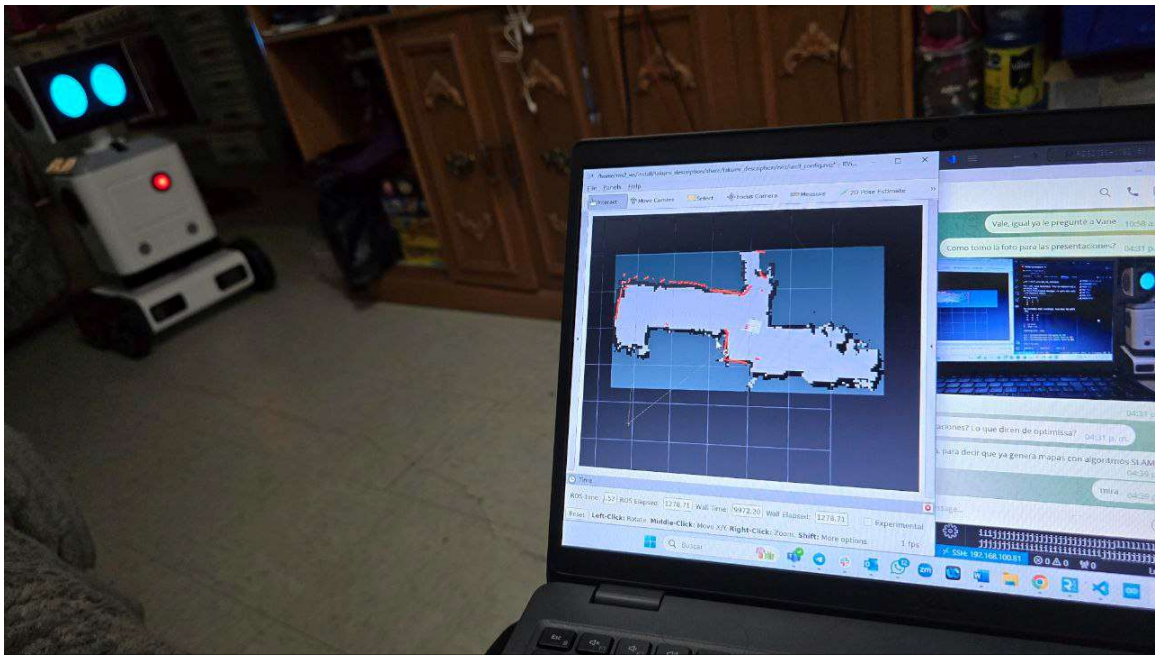


Figura 37. Rviz con algoritmos SLAM activados en Takumi.

En la figura 38, podemos ver el mapa generado por el robot del espacio virtual previamente generado, así es como ve el robot el espacio donde navega; sin embargo, debido a algunas limitaciones del sensor como las formas cónicas o superficies huecas como muebles, mesas y sillas hay que hacer pequeñas correcciones al mapa para que pueda utilizarse un filtro de *keepout* y evitar que el robot entre en zonas donde pueda hacerse daño, como se muestra en la Figura 39, donde se sombrea el área de las mesas, sillas y muebles.

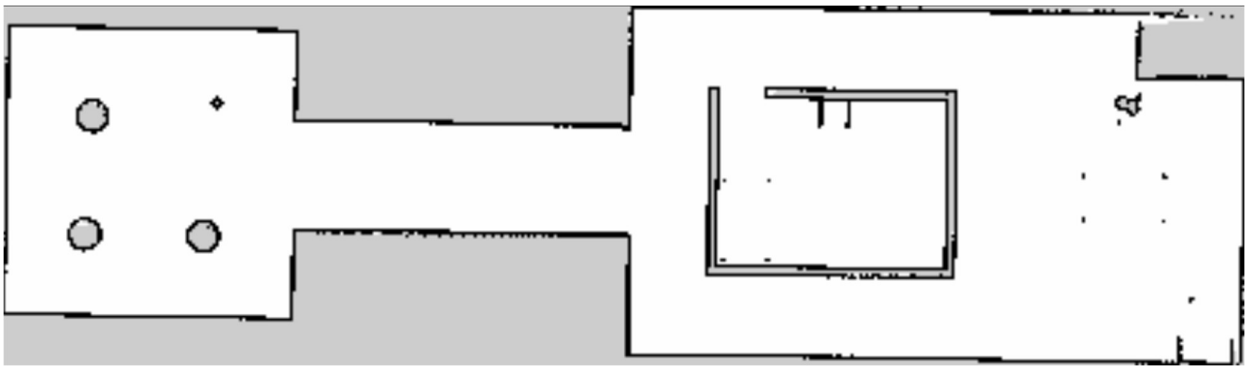


Figura 38. Archivo .pgn generado del espacio virtual.

Figura 39. Mapa del entorno virtual con corrección.

Finalmente, en la Figura 40 podemos ver el mapa generado con algoritmos SLAM del espacio real mostrado en la Figura 41.

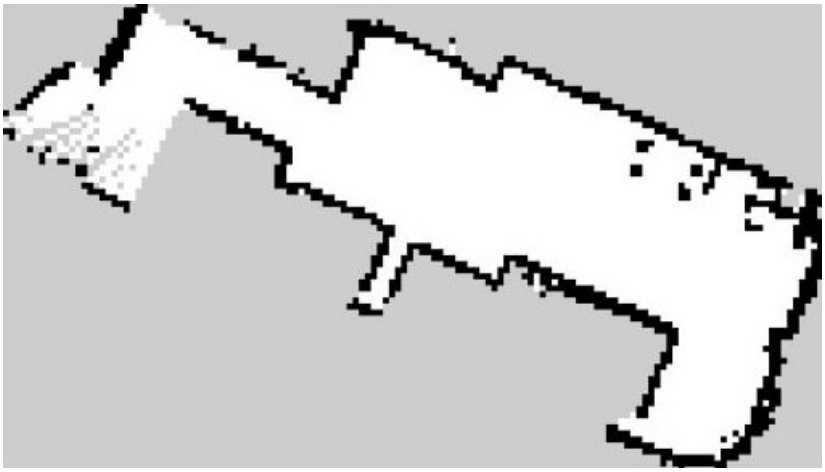


Figura 40. Mapa generado con SLAM de un departamento.

Figura 41. Departamento mapeado.

4.11. LOCALIZACIÓN

El robot cuenta con un sistema de localización que le permite ubicarse con precisión dentro de un mapa previamente generado (Figura 42 y 43). Para ello, implementé el filtro AMCL (*Adaptive Monte Carlo Localization*), este filtro utiliza la información de sensores como el LiDAR y los encoders para determinar constantemente la posición y orientación del robot dentro del entorno.

Esta localización permite que el robot, pueda hacer lo siguiente:

- Sepa en todo momento dónde se encuentra dentro del mapa
- Realice navegación planificada hacia objetivos definidos
- Recupere su ubicación incluso ante ligeras desviaciones o ruidos del entorno

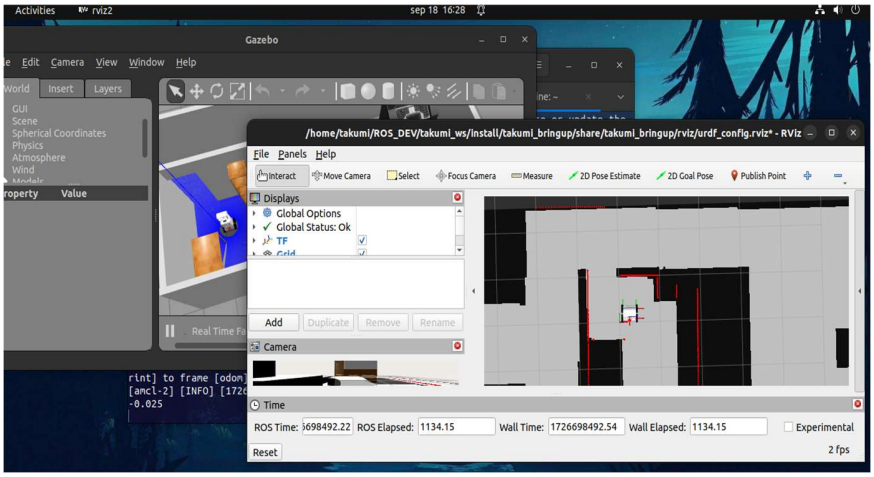


Figura 42. Localización del robot dentro del mapa generado en la simulación.

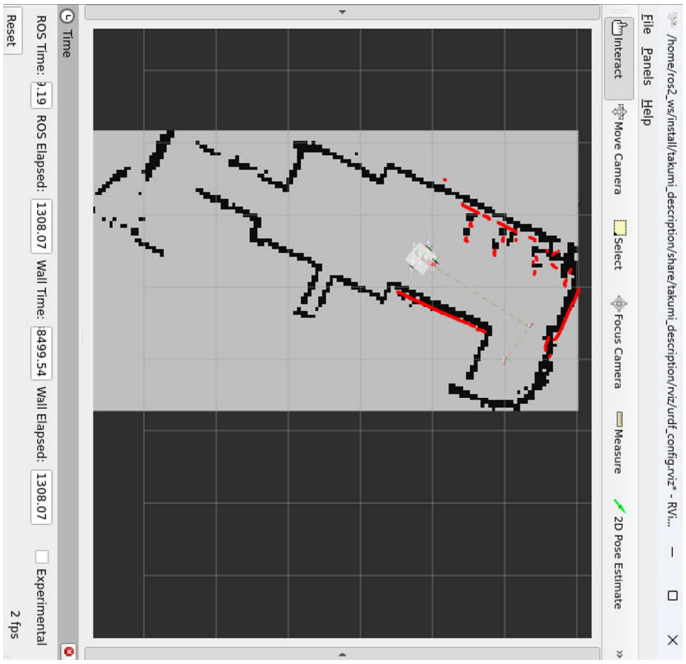


Figura 43. Localización del robot dentro del mapa generado del departamento.

4.12. NAVEGACIÓN AUTÓNOMA

El robot es capaz de desplazarse de manera autónoma en su entorno, sin necesidad de intervención humana directa. Esta capacidad se logra mediante la combinación de sensores, algoritmos de planeación y sistemas de control que le permiten percibir su entorno, generar rutas seguras y adaptarse a situaciones cambiantes en tiempo real. El sistema de navegación autónoma está compuesto por dos enfoques principales: navegación deliberativa y navegación reactiva, que se complementan para poder cumplir de mejor manera el objetivo planteado.

4.12.1. NAVEGACIÓN DELIBERATIVA

La navegación deliberativa permite al robot planificar la ruta más eficiente hacia un destino, utilizando como referencia un mapa previamente generado con algoritmos SLAM. Este comportamiento se basa en un mapa de costos global (Figura 45), el cual asigna valores a cada celda (píxel) del mapa según su cercanía a obstáculos, representados como zonas negras. Cuanto mayor es la cercanía, mayor el costo asignado, visualizado con distintos niveles de opacidad.

Con esta información, el sistema de navegación (basado en el *framework* Nav2 de ROS 2) utiliza algoritmos de búsqueda como A*, Dijkstra o RRT para calcular la ruta óptima hacia el destino, minimizando riesgos y tiempo de recorrido. En la Figura 44 podemos ver cómo se utilizan los mapas de costos para determinar las rutas más optimas:

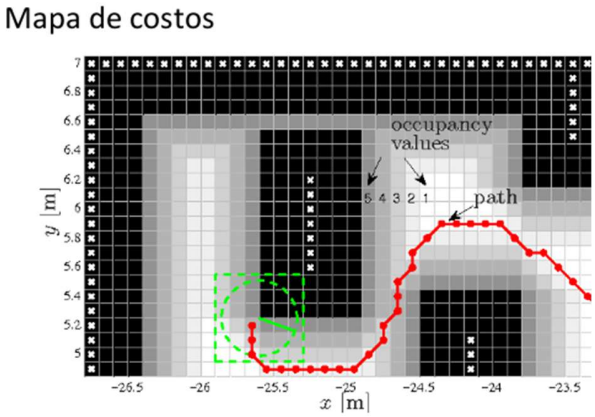


Figura 44. Mapa de costos para la navegación autónoma [29].

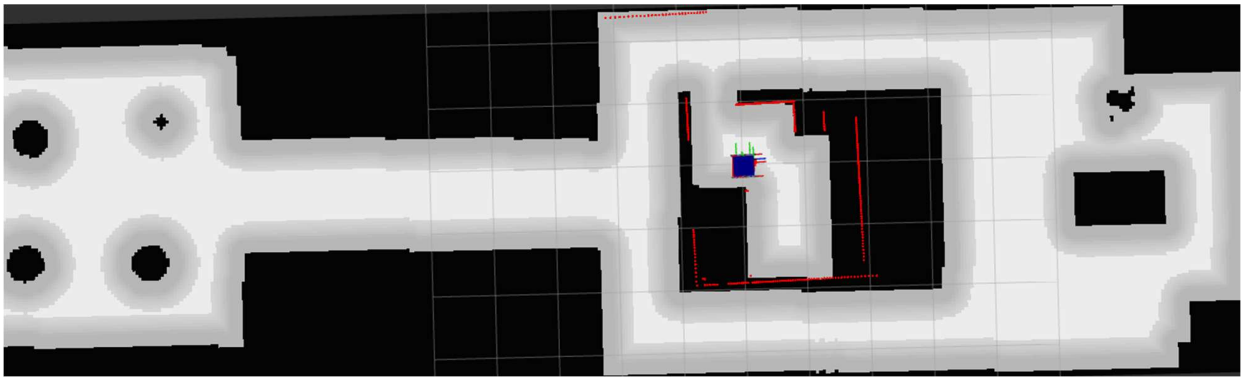


Figura 45. Mapa de costos global de Takumi en el espacio virtual generado.

4.12.2. NAVEGACIÓN REACTIVA

Por otro lado, la navegación reactiva permite que el robot tenga un comportamiento más dinámico, evitando obstáculos inesperados en tiempo real (como personas caminando). A diferencia de la navegación deliberativa, la navegación reactiva no utiliza un mapa previamente mapeado, en su lugar utiliza la información de los sensores en tiempo real (principalmente del sensor LiDAR). El robot determina la distancia de los obstáculos y el asigna un consto alto en su mapa de costos local (Figura 46). Gracias a esto el robot es capaz de ajustar y recalcular su trayectoria sobre la marcha para evitar colisiones. Esta navegación es muy importante en entornos reales ya que difícilmente no son dinámicos.

Figura 46. Mapa de costos local del robot en el espacio virtual generado.

4.12.3. RESULTADOS DE NAVEGACIÓN AUTÓNOMA

La combinación de la navegación deliberativa, para poder calcular la ruta optima del punto inicial al punto deseado, y la navegación reactiva, para poder esquivar obstáculos en tiempo real, hacen posible que el robot pueda navegar de forma autónoma, en la Figura 47 podemos ver como el robot de la simulación tiene una ruta trazada por la navegación deliberativa, que en la imagen se ve como una línea roja, y al mismo tiempo está ocupando la información del sensor LiDAR para generar su mapa de costos local, lo que le permitiría cambiar la ruta si es que se requiere, por ello podemos ver un contraste entre el mapa de costos global que se asigna a todos los obstáculos representados en un color oscuro y que fueron previamente mapeados y su mapa de costos local que tiene colores más brillantes alrededor del robot y que corresponden a lo que ve en ese momento; de forma similar esto se puede ver en el robot real de la Figura 48, donde los puntos rojos representan la nube de puntos generada por el LiDAR, en este caso inclusive podemos ver cómo funcionan los filtros de *keepout*, pues a pesar de que el LiDAR no detecta ningún obstáculo en la zona oscura (filtro *keepout*), el robot traza una ruta para no pasar por esa zona.

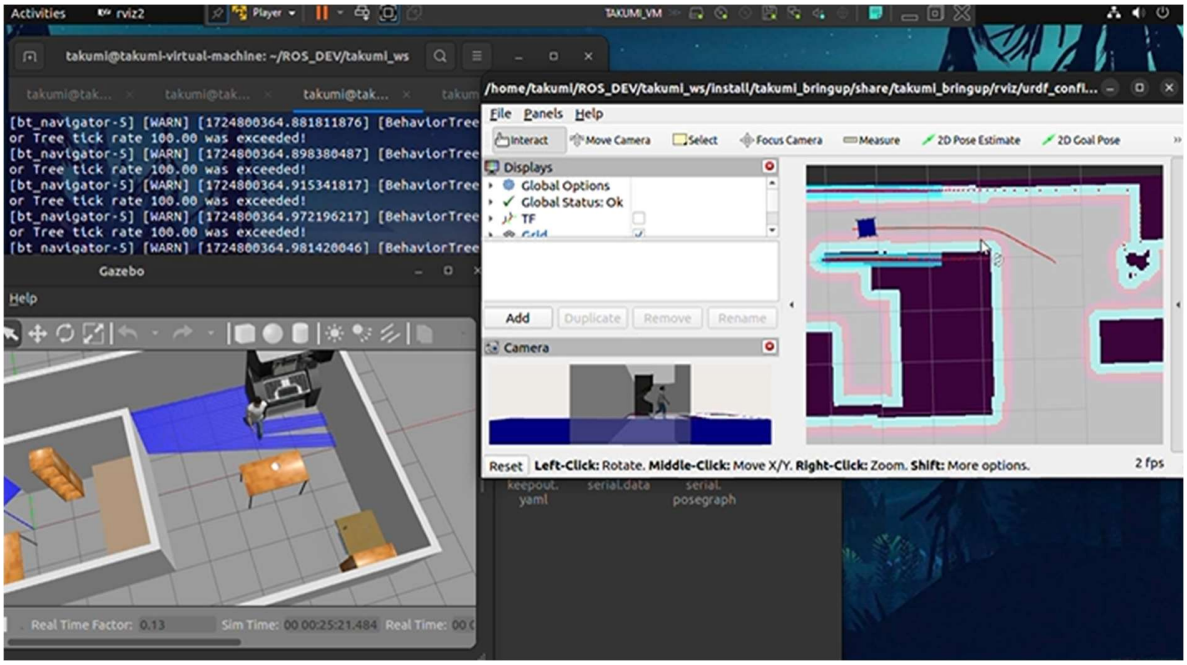


Figura 47. Navegación autónoma del robot en la simulación.

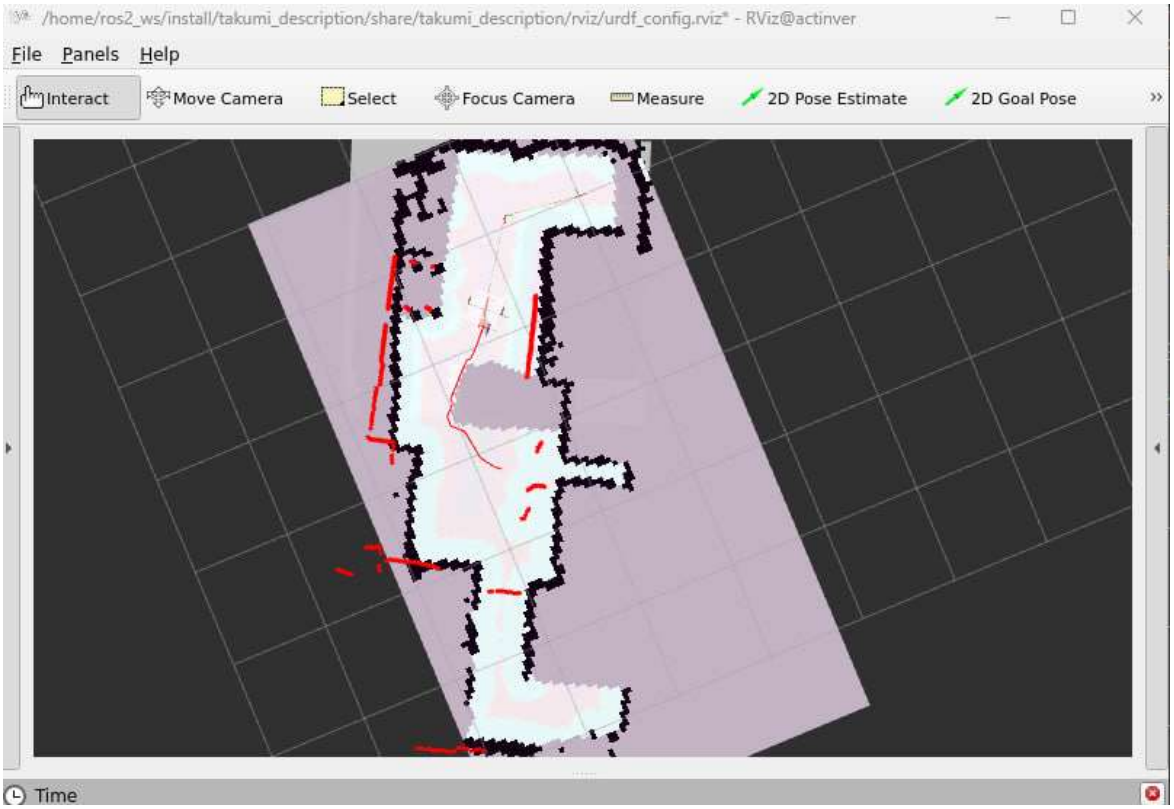


Figura 48. Navegación autónoma del robot en el departamento.

5. ANÁLISIS DE RESULTADOS

El desarrollo del robot me ha permitido validar múltiples aspectos fundamentales para la interacción y la navegación autónoma del mismo en entornos reales. Como se mencionó en un inicio, el desarrollo se fue de manera iterativa, mejorando poco a poco los diferentes aspectos de este, intentando resolver problemas tanto de hardware como de software; sin embargo, los resultados finales han sido satisfactorios tanto físicamente como en la simulación. Uno de los principales logros fue la implementación de la navegación autónoma llevada a cabo en el centro financiero, que permite al robot desplazarse desde su posición inicial hasta un punto de interés del usuario dentro de un mapa previamente generado con algoritmos SLAM. Esta función se validó tanto en simulación (Gazebo) como en el robot físico, lo que demostró el correcto funcionamiento de los paquetes implementados con ROS2. Debido a la naturaleza diferencial de cuatro ruedas del robot, se enfrentaron desafíos al momento de representar y controlar correctamente su dinámica. Inicialmente, modelé como un sistema diferencial doble; es decir, un control diferencial delantero y uno trasero, pero tuve dificultades a la hora de implementarlo de forma física, por eso opté por simplificar el modelo, tratando al robot como un robot diferencial de dos ruedas, de modo que utilicé una misma señal de velocidad para controlar 2 motores del mismo lado. Esta simplificación hizo que redefiniera la descripción de mi robot en los archivos URDF, en la configuración de los controladores y en la lógica de publicación de comandos en el tópico de /cmd_vel. Por otro lado, utilicé las herramientas de TF2 para verificar que la interpretación del espacio y el posicionamiento de mis sensores y actuadores se estuviera llevando a cabo de forma adecuada, viendo principalmente las transformaciones entre los distintos marcos de referencia del robot. Para ello generé un archivo PDF representando el árbol de transformaciones, donde visualicé las relaciones jerárquicas entre los *frames* como base_footprint, base_link, camera_link, imu_link, entre otros (ver Figura 49).

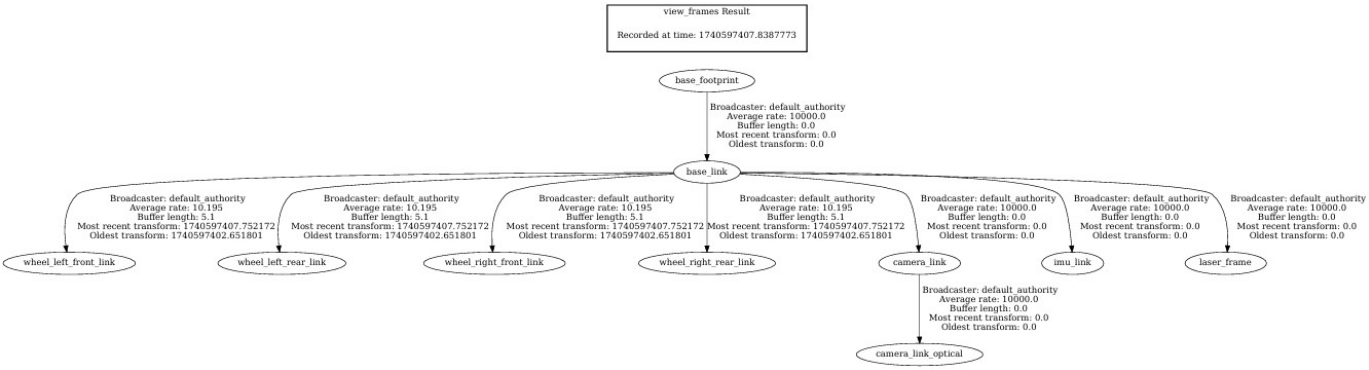


Figura 49. Árbol de relación de los sistemas de referencia en la descripción del robot.

Esta visualización permitió identificar problemas de conexión o definiciones incorrectas entre *frames*, así como como:

- Rate de transmisión promedio
- Duración del buffer de transformaciones
- Transformaciones más recientes y antiguas

Debido a esta sobrecarga de detalles, el diagrama resulta poco distinguible. Por ello, en la Figura 50, se presenta un diagrama simplificado que muestra únicamente las relaciones entre los sistemas de referencia del robot, facilitando así su comprensión.

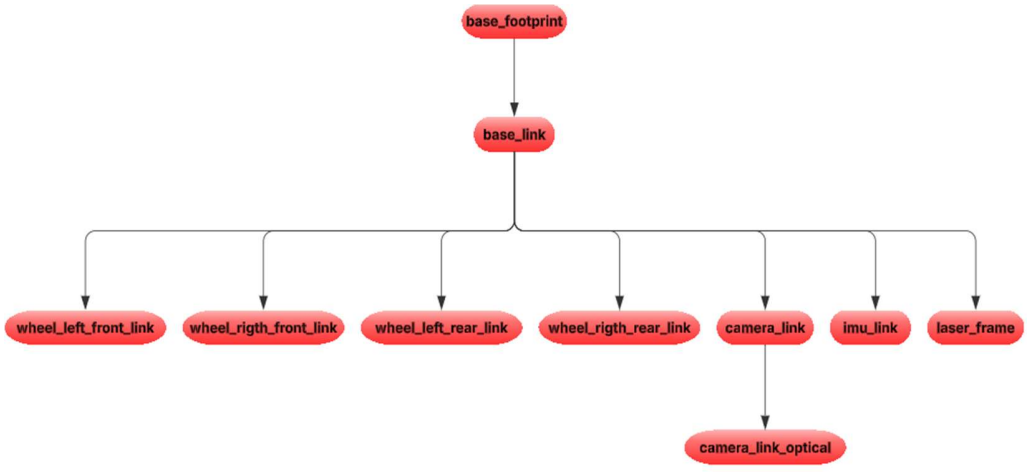


Figura 50. Diagrama simplificado del árbol de relación de los sistemas de referencia en la descripción del robot.

Como podemos ver en la imagen, el nodo raíz es el “base_footprint” que hace referencia a la huella del robot, y de ahí parten todos los sistemas, siendo el “base_link” el nodo principal donde se conectan todos los sensores del robot, con esta estructura bien definida, garantizamos que el sistema de transformaciones del robot sea preciso y funcional, lo que resulta fundamental para su correcta operación en simulación y en el mundo real.

Por otro lado, utilicé la herramienta rqt_graph para analizar dinámicamente el comportamiento del sistema, que permite visualizar la arquitectura activa del sistema en términos de nodos y tópicos. Esta herramienta me ayudo a identificar errores en el flujo de la información y a comprender este flujo de información más a detalle entre los nodos y tópicos.

En la Figura 51 podemos ver detalladamente la conexión entre los nodos y tópicos en la descripción del robot, y la interacción entre los distintos componentes del mismo. Esto también es útil para optimizar los diferentes procesos y garantizar el correcto funcionamiento de todos los nodos.

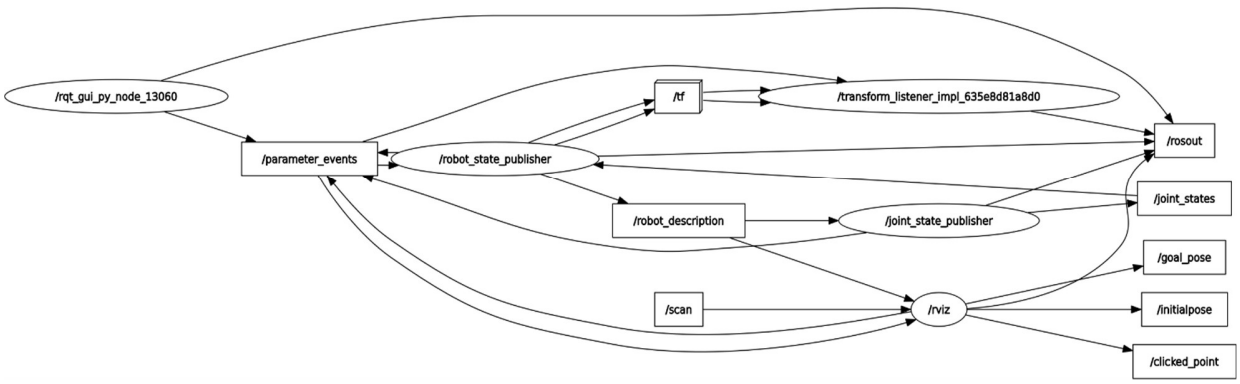


Figura 51. Relación entre nodos y tópicos en la descripción del robot.

Asimismo, observe que todos los nodos críticos (por ejemplo, los de descripción, navegación, control y los de los sensores) estaban correctamente conectados. En la imagen podemos ver de forma más evidente como la comunicación es modular, organizada y escalable. Aprovechando la modularidad, decidí agrupar los nodos principales por medio de archivos *launch* según su función, los cuales son activados

mediante alias dentro del contenedor de Docker. Así es más fácil inicializar el robot, ver si algo salió mal en alguna parte en específico y finalizar al robot. Actualmente, empleo nueve alias: interfaz (inicia el nodo de la interfaz y se suscribe a tópicos como /goal_pose para determinar cuándo inicia la navegación y reproducir sonidos, o como /cmd_interfaz, un tópico que utilizo para sincronizar a la interfaz con los nodos de control), robot (inicializa todos los nodos de control del robot, para poder controlar los motores, obtener las lecturas de los encoders y actualizar el estado de las juntas), LiDAR (activa el sensor LiDAR para obtener la información del entorno alrededor del robot), ver (inicia Rviz con una configuración preestablecida en una ventana de mi computadora), Slam (ejecuta los algoritmos Slam del paquete de Slam_toolbox para poder mapear entornos desconocidos), manejar (permite controlar al robot manualmente desde la terminal utilizando el paquete de teleop_twist_keyboard), Svmap (guarda el mapa previamente generado con algoritmos Slam en una carpeta que esta sincronizada con mis archivos locales, lo que permite tener los mismo dentro y fuera del contenedor y que no se pierdan los mapas al terminar de ejecutar el contenedor), localizar (me permite localizar al robot dentro de un mapa previamente generado con algoritmos Slam) y navegar (me permite utilizar los paquetes de Nav2 para navegar de un punto a otro dentro del mapa previamente generado). Desarrolle tres formas de poder controlar al robot, todas diferentes y útiles en diferentes contextos, facilitándome tanto el desarrollo como el control del robot en demostraciones y pruebas en ambientes reales.

5.1. RETOS ENCONTRADOS

Durante el desarrollo del proyecto, me encontré con diferentes desafíos, esto me exigió aprender más sobre temas en específico y proponer soluciones creativas. A continuación, mostrare a detalle algunos de los retos a los cuales tuve que enfrentarme:

5.1.1. COMPATIBILIDAD

Uno de los principales desafíos a los que me enfrente fue que ROS 2 Humble no es compatible directamente con la Raspberry Pi 5. Esto debido a que la Raspberry Pi 5 solo permite la instalación de sistemas operativos basados en Ubuntu 24.04 o versiones más actualizadas, o en su defecto la instalación de sistemas operativos alternos como el *Raspberry Pi OS Bookworm*, y ROS 2 Humble solo es compatible oficialmente con Ubuntu 22.04, por lo que no fue posible instalar directamente el Ubuntu 22.04 para poder trabajar con ROS 2 Humble. Por otro lado, la mayor parte del desarrollo de los paquetes incluyendo los paquetes de los sensores estaba desarrollado para ROS2 Humble, por lo que, no era conveniente rehacer todos los paquetes, en su lugar decidí crear un contenedor de Docker personalizado, de esta manera pude seguir con el desarrollo del robot en ROS 2 Humble dentro de la Raspberry Pi 5. Esta solución implicó aprender sobre Docker, desde como descargarlo, implementarlo y hasta como crear una imagen y contenedor personalizados. Otro caso de incompatibilidad se presentó con la cámara *Intel RealSense D435i*, la cual no cuenta con soporte adecuado para la Raspberry Pi 5 debido a restricciones en controladores y dependencias del sistema. Como solución, se reemplazó este componente con la cámara de las Raspberry Pi cam, lo que permitió continuar con las pruebas de percepción visual sin afectar el cronograma del proyecto. De forma general, se presentaron múltiples casos de incompatibilidad entre componentes de hardware y software, debido a que se habían adquirido desde antes de requerirse, estos obligaron a ajustar diseños, adquirir nuevos dispositivos o modificar paquetes existentes.

5.1.2. DISEÑO DE ARQUITECTURA DE CONTROL

El robot host financiero está diseñado como un robot diferencial de cuatro ruedas, lo cual generó ciertas complicaciones a nivel de simulación y control físico. Inicialmente, se implementó un modelo con doble control diferencial en el simulador Gazebo. Sin embargo, al momento de trasladar esta configuración al robot físico, surgieron problemas de sincronización y control entre los motores. La solución fue simplificar la arquitectura lógica,

reconfigurando todos los paquetes de ROS 2 para tratar al robot como si fuera un sistema diferencial de dos ruedas, enviando una única señal a cada par de motores del mismo lado. Esto me permitió que pudiera aplicar un control diferencial logrando el resultado deseado.

5.1.3. TRANSICIÓN DE ENTORNOS

También, considero que fue un gran reto poder llevar a la práctica todos los conocimientos adquiridos en clases, ya que la mayoría de mis conocimientos estaban basados en simulaciones y clases teóricas, pero al enfrentarse a un entorno real te encuentras con problemas que no tenías contemplados de forma teórica o en las simulaciones por más realistas que sean. Esta transición me demostró que no todo se aprende en las aulas de clases, y que como profesionales tenemos que seguir con nuestra formación de manera autodidacta, realizando cursos, leyendo artículos, revisando la documentación oficial y foros de la comunidad e incluso investigando contenido específico para resolver los distintos problemas.

5.1.4. COMUNICACIÓN Y PRESENTACIONES EJECUTIVAS

Además, de los problemas técnicos, también fue necesario trabajar en mis habilidades blandas, sobre todo en la parte de mi comunicación oral. Ya que durante el desarrollo muchas veces tuve que se realizar presentaciones del proyecto ante directores y personas interesadas en el proyecto dentro del banco; sin embargo, hay que tener en cuenta que cada público es diferente y el mensaje tienes que hacerlo llegar de distintas maneras. Este tipo de presentaciones no pueden compararse con las exposiciones académicas que realizaba en la facultad, pues, aunque te ayudaban a quitarte el nerviosismo de hablar en público, muchas veces no se hacía énfasis en esta parte de la oratoria de hacer llegar el mismo mensaje, pero con diferentes palabras dependiendo del público al que le estes exponiendo el proyecto, esto requería explicar aspectos técnicos complejos de manera clara, concisa y comprensible para audiencias no técnicas.

5.2. ÁREAS DE OPORTUNIDAD

Eh identificado algunas áreas de mejora para futuras versiones del proyecto, sin perder de vista el objetivo general del mismo. A continuación, mostrare algunas de ellas:

5.2.1. INTERFAZ E INTERACCIÓN

- Automatizar la activación de animaciones en respuesta a eventos o condiciones del entorno con ayuda de la inteligencia artificial.
- Integrar un chatbot básico para mejorar la interacción verbal sin necesidad de conexión a modelos externos.
- Ampliar la interacción mediante la incorporación de un chatbot avanzado basado en chatGPT, que permita conversaciones más naturales y adaptativas.
- Implementar reconocimiento facial básico para inferir características como grupo de edad o emociones predominantes.
- Desarrollar un sistema de reconocimiento facial avanzado que permita identificar clientes específicos.

5.2.2. SEGURIDAD

- Incorporar sensores ultrasónicos para mejorar la detección de obstáculos que no puede detectar el sensor LiDAR como una pared de cristal.
- Añadir sensores de detección de caída para prevenir que el robot se caiga por las escaleras en caso de que falle algo en la navegación.
- Implementar funciones de seguridad en caso de que se pierda la conexión wifi o bluetooth con la Raspberry Pi 5, que haya errores de comunicación entre los nodos o que se desconecta algún sensor o actuador.

5.2.3. CHASIS/ESTRUCTURA

- Rediseñar el chasis interno para que se adapte mejor a la carcasa externa

5.2.4. NAVEGACIÓN

- Habilitar la funcionalidad de navegación bidireccional, permitiendo que el robot se desplace a un destino y regrese de forma autónoma a su punto de origen.
- Configurar zonas específicas dentro del mapa previamente generado para que el robot pueda reconocerlas y operar de acuerdo con su función (por ejemplo, recepción, sala de espera, etc.).
- Desarrollar un sistema de adaptación al tipo de piso, permitiendo al robot modificar su comportamiento o velocidad en función de superficies detectadas (alfombra, baldosa, etc.).
- Integrar algoritmos de aprendizaje reforzado que, a través del reconocimiento visual mediante la cámara, permitan al robot adaptar su comportamiento dinámicamente en función de los objetos detectados en su entorno.

5.2.5. OPTIMIZACIÓN

- Reestructurar el código y los procesos en una arquitectura más modular para facilitar la escalabilidad y la integración de nuevas funciones.

6. CONCLUSIÓN

El desarrollo de este proyecto, el primer robot host financiero en México en una sucursal bancaria representa un paso significativo hacia la transformación de la atención al cliente en entornos bancarios en México, alineándose con la visión de ofrecer experiencias más personalizadas, eficientes y tecnológicamente innovadoras. Actualmente, si bien gran parte de las interacciones con el robot aún se realizan de forma manual, excepto por su comportamiento pasivo (que activa animaciones cada cierto tiempo y parpadea cada 5 segundos), y la navegación autónoma hacia puntos específicos dentro de un mapa generado previamente, se han establecido las bases sólidas del objetivo, que habilitan el desarrollo iterativo hacia una versión más autónoma. La implementación de la navegación autónoma, sus diferentes formas de control a distancia y la interacción con los usuarios en escenarios reales demuestran el éxito del proyecto. El robot ya es capaz de ejecutar tareas clave como desplazarse de forma precisa por superficies planas dentro de un entorno mapeado, lo que constituye un logro fundamental en esta primera etapa del proyecto.

Estoy consciente que el robot aún está en proceso de desarrollo, pero tiene un alto potencial para incorporar inteligencia artificial, reconocimiento facial, comportamiento adaptativo y sistemas de interacción avanzada. Con estas mejoras, el robot podrá evolucionar desde un agente semiautónomo hacia un asistente anfitrión completamente autónomo y proactivo, capaz de revolucionar la experiencia del cliente en los centros financieros del banco.

Mi formación universitaria me proporcionó los fundamentos teóricos necesarios para poder llevar a cabo este proyecto de la mejor forma. El progreso alcanzado hasta ahora es solo el inicio para crear el mejor robot host en una sucursal bancaria, que no solo reduce el tiempo de búsqueda de lugares clave, sino que también revoluciona la forma en que los usuarios interactúan con los servicios en los centros financieros de México.

REFERENCIAS

[1] B. Streeter, «The Financial Brand,» 05 06 2019. [En línea]. Available: https://thefinancialbrand.com/news/banking-branch-transformation/hsbc-banks-branch-robot-pepper-digital-transformation-phygital-84245. [Último acceso: 05 2025].

[2] «LA TERCERA,» 07 02 2023. [En línea]. Available: https://www.latercera.com/piensa-digital/noticia/el-michi-robot-que-revoluciona-a-un-restaurante-de-vina-del-mar/SSJ55OHJUFEZHFYIHWDKHQ64VQ/. [Último acceso: 05 2025].

[3] «LA RAZÓN,» 14 01 2016. [En línea]. Available: https://www.larazon.es/tecnologia/robot-relay-servicio-de-habitaciones-OO11682048/. [Último acceso: 05 2025].

[4] «The ROS Community,» 2021. [En línea]. Available: https://www.ros.org/blog/community/. [Último acceso: 05 2025].

[5] W. Woodall, «ROS en DDS,» 07 2019. [En línea]. Available: https://design-ros2-org.translate.goog/articles/ros_on_dds.html?_x_tr_sl=en&_x_tr_tl=es&_x_tr_hl=es&_x_tr_pto=tc. [Último acceso: 05 2025].

[6] H. Kutluca, «Medium,» 06 12 2020. [En línea]. Available: https://medium.com/software-architecture-foundations/robot-operating-system-2-ros-2-architecture-731ef1867776. [Último acceso: 05 2025].

[7] Gary, «hackster.io,» 19 05 2023. [En línea]. Available: https://www.hackster.io/gary26/learning-slam-and-dynamic-obstacle-avoidance-with-myagv-fd0ceb. [Último acceso: 05 2025].

[8] Open Robotics, «rviz – ROS Index,» Open Robotics, 2025. [En línea]. Available: https://index.ros.org/p/rviz/. [Último acceso: 21 01 2026].

[9] «GAZEBO,» [En línea]. Available: https://gazebosim.org/home. [Último acceso: 05

2025].

[10] S. a. M. F. a. W. R. a. G. C. J. {Macenski, «Nav2,» Open Navigation LLC, 2020. [En línea]. Available: https://docs.nav2.org/. [Último acceso: 05 2025].

[11] «AUTODESK,» [En línea]. Available: https://www.autodesk.com/mx/products/fusion-360/overview?term=1-YEAR&tab=subscription. [Último acceso: 05 2025].

[12] SLAMTEC, «A1 Lidar SLAMTEC,» SLAMTEC, [En línea]. Available: https://www.slamtec.com/en/Lidar/A1. [Último acceso: 21 01 2026].

[13] RealSense AI, «Depth Camera D435i - Realsense AI,» RealSense AI, [En línea]. Available: https://www.realsenseai.com/products/depth-camera-d435i/. [Último acceso: 21 01 2026].

[14] RS Components, «Cámaras de profundidad – RS Components,» RS Components, [En línea]. Available: https://es.rs-online.com/web/p/camaras-de-profundidad/1720981. [Último acceso: 21 01 2026].

[15] Raspberry Pi Ltd, «AI Camera – Raspberry Pi,» Raspberry Pi Ltd, [En línea]. Available: https://www.raspberrypi.com/products/ai-camera/. [Último acceso: 21 01 2026].

[16] 330 Ω, «Cámara AI Oficial Raspberry Pi – 330Ω,» 330 Ω, [En línea]. Available: https://www.330ohms.com/shop/rp-01174-camara-ai-oficial-raspberry-pi-1461. [Último acceso: 21 01 2026].

[17] Raspberry Pi Ltd, «Raspberry Pi 5 – Raspberry Pi,» Raspberry Pi Ltd, [En línea]. Available: https://www.raspberrypi.com/products/raspberry-pi-5/. [Último acceso: 21 01 2026].

[18] https://documentation.espressif.com/esp32_datasheet_en.pdf, «ESP32 Series,» [En línea]. Available: https://documentation.espressif.com/esp32_datasheet_en.pdf. [Último acceso: 21 01 2026].

[19] GOOBE, «Hiletgo ESP-WROOM-32 / ESP32 Placa de Desarrollo – MercadoLibre México,» MercadoLibre México, [En línea]. Available: https://www.mercadolibre.com.mx/hiletgo-esp-wroom-32-esp32-esp-32s-placa-de-

desarrollo-24gh/p/MLM2034440797?pdp_filters=item_id:MLM1997708793. [Último acceso: 21 01 2026].

[20] Amazon, «WIMAXIT Portable Screen for Raspberry Pi – Amazon.com.mx,» Amazon, [En línea]. Available: https://www.amazon.com.mx/WIMAXIT-port%C3%A1til-pantalla-Raspberry-controlador/dp/B09QPWRCB1. [Último acceso: 21 01 2026].

[21] Dayton Audio, «PC68-8 2-1/2" Full-Range Poly Cone Driver – Dayton Audio,» Dayton Audio, [En línea]. Available: https://www.daytonaudio.com/product/1539/pc68-8-2-1-2-full-range-poly-cone-driver. [Último acceso: 21 01 2026].

[22] MakerHawk-MX, «Amplificador Bluetooth de potencia para altavoces – Amazon.com.mx,» Amazon, [En línea]. Available: https://www.amazon.com.mx/amplificador-Bluetooth-potencia-Amplificador-altavoces/dp/B08Z3FBSDT/ref=pd_day0fbt_d_sccl_2/147-9143893-5324527?pd_rd_w=g1Stv&content-id=amzn1.sym.7097b179-c7b2-4b3f-9bd2-0a674e70d880&pf_rd_p=7097b179-c7b2-4b3f-9bd2-0a674e70d880. [Último acceso: 21 01 2026].

[23] Amoq, «Motorreductor 10–1500 RPM con encoder – Amazon.com.mx,» Amazon, [En línea]. Available: https://www.amazon.com.mx/10-1500RPM-Motorreductor-Velocidad-Reducci%C3%B3n-Exc%C3%A9ntrico/dp/B07WXFX5SX/ref=sr_1_1?__mk_es_MX=%C3%85M%C3%85%C5%BD%C3%95%C3%91&crid=3UXF0MANFKEY6&dib=eyJ2IjoiMSJ9.YBnUslKJ8aGcS6ynvCR1Tu6Rb-AXtfrWZvs7u22noRXmBmFb0ZAQPlpq6dR. [Último acceso: 21 01 2026].

[24] Handson Technology Co., Ltd., «BTS7960 Motor Driver Datasheet,» Handson Technology Co., Ltd., [En línea]. Available: https://www.handsontec.com/dataspecs/module/BTS7960%20Motor%20Driver.pdf. [Último acceso: 21 01 2026].

[25] Store, Go to School, «Motorreductor / Producto Electrónico – AliExpress,»

AliExpress, [En línea]. Available: https://es.aliexpress.com/item/1005005012551847.html. [Último acceso: 21 01 2026].

[26] Genstattu, «Tattu G-Tech 6S 10000mAh 30C 22.2V Lipo Battery Pack with EC5 Plug for UAV Drone – Genstattu,» Genstattu, [En línea]. Available: https://genstattu.com/ta-30c-10000-6s1p-ec5.html. [Último acceso: 21 01 2026].

[27] Amazon, «DC Buck Converter 6-55V to 0-50V 8A 400W 6V 12V 24V 36V 48V Constant Voltage and Constant Current Buck Module DC Regulated Power Supply – Droking,» Droking, [En línea]. Available: https://www.droking.com/dc-power-supply/dc-buck/dc-buck-converter-6-55v-to-0-50v-8a-400w-6v-12v-24v-36v-48v-constant-voltage-and-constant-current-buck-module-dc-regulated-power-supply. [Último acceso: 21 01 2026].

[28] Shenzhen SUPERNOVA Electronics, «Monitor de batería de 12 V, 24 V, 36 V, 48 V, 60 V, 72 V, coche, carrito de golf, probador de batería, capacidad de visualización digital, 7 a 100 V, monitor de voltaje, probador de porcentaje de – Amazon.com.mx,» Amazon, [En línea]. Available: https://www.amazon.com.mx/dp/B09JMY44LX?ref=cm_sw_r_cso_wa_apan_dp_1VRF34C037JMAM80CFV5&ref_=cm_sw_r_cso_wa_apan_dp_1VRF34C037JMAM80CFV5&social_share=cm_sw_r_cso_wa_apan_dp_1VRF34C037JMAM80CFV5&starsLeft=1&skipTwisterOG=1&th=1. [Último acceso: 21 01 2026].

[29] I. P. M. Dakulović, «Two-way D* algorithm for path planning and replanning,» *Robotics and Autonomous Systems,* vol. 59, nº 5, p. 329–342, 2011.

ANEXOS

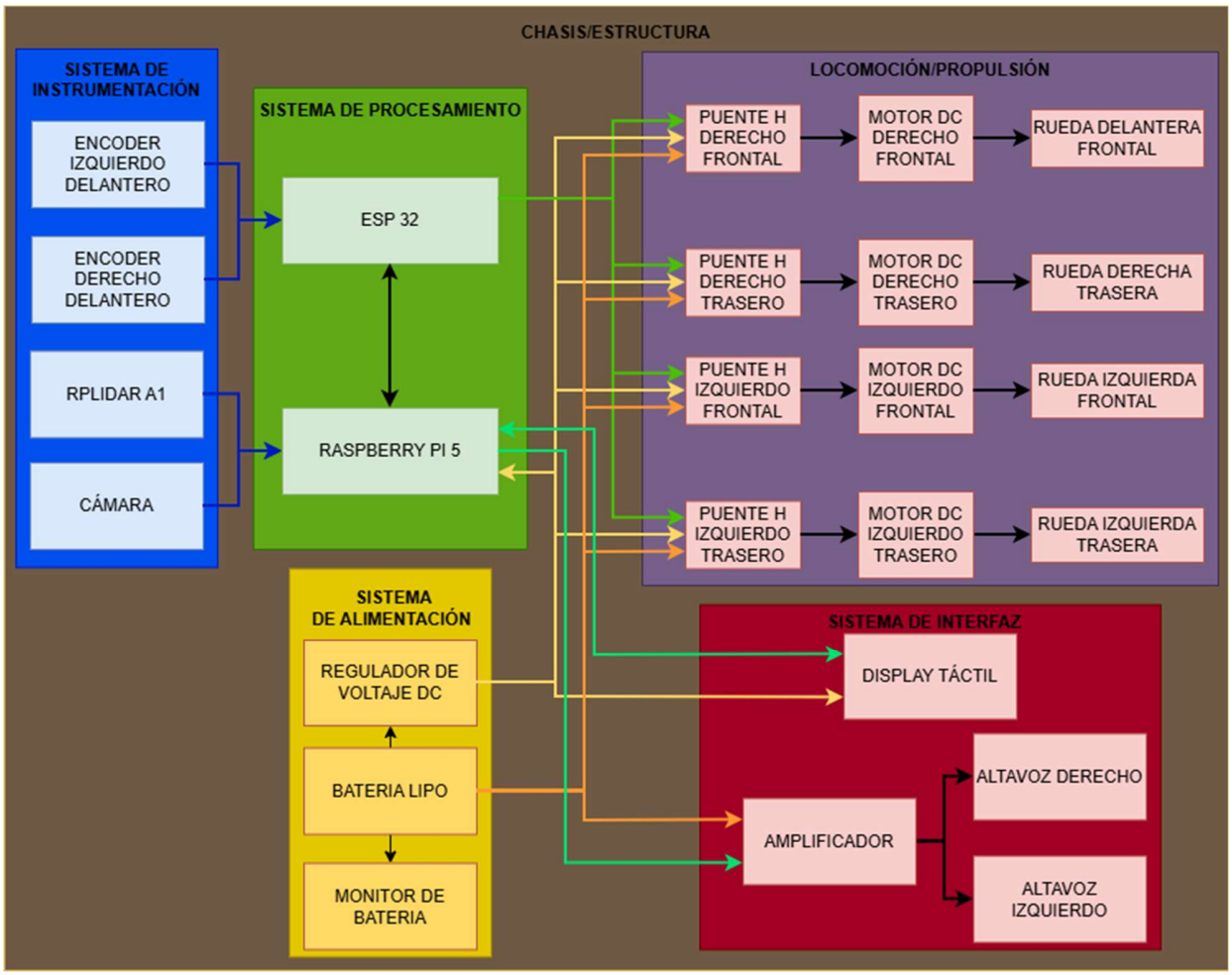
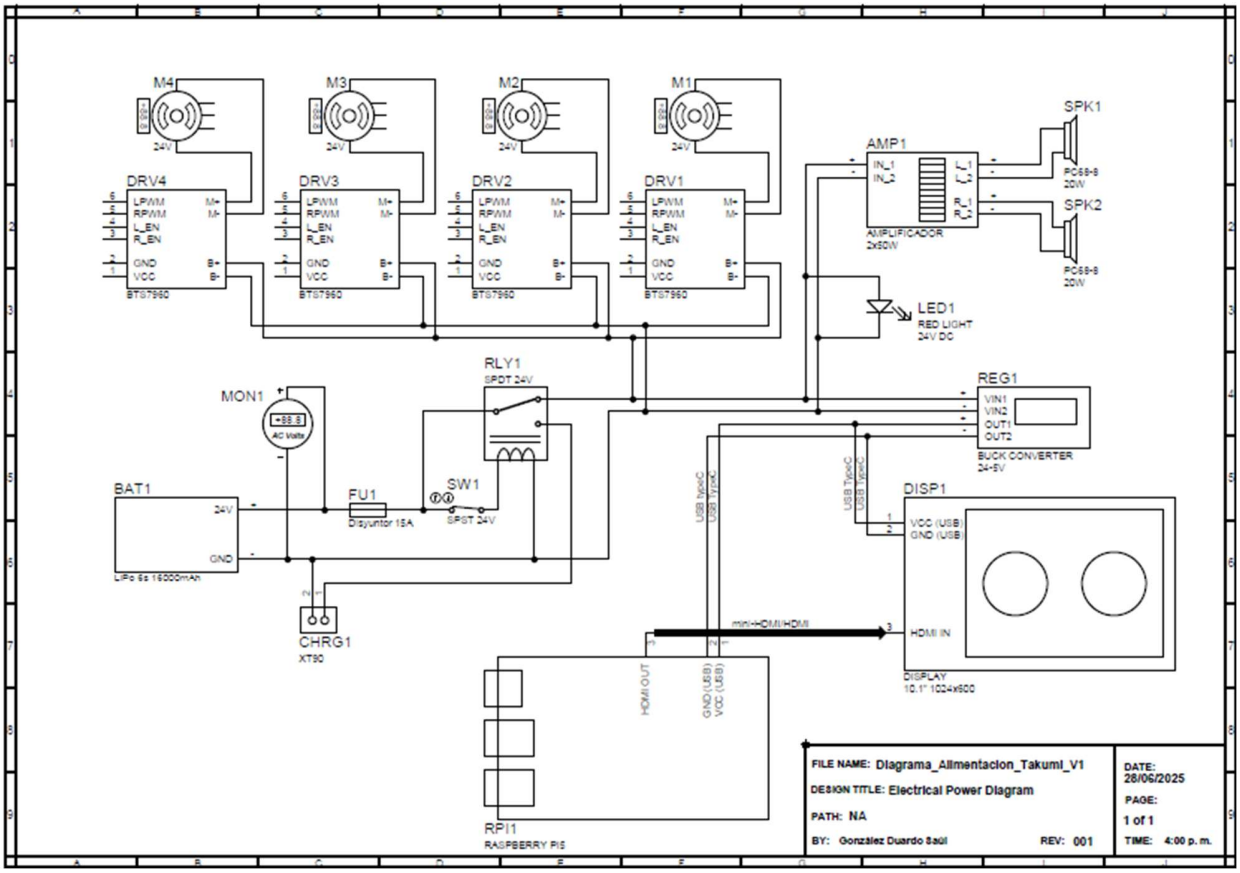
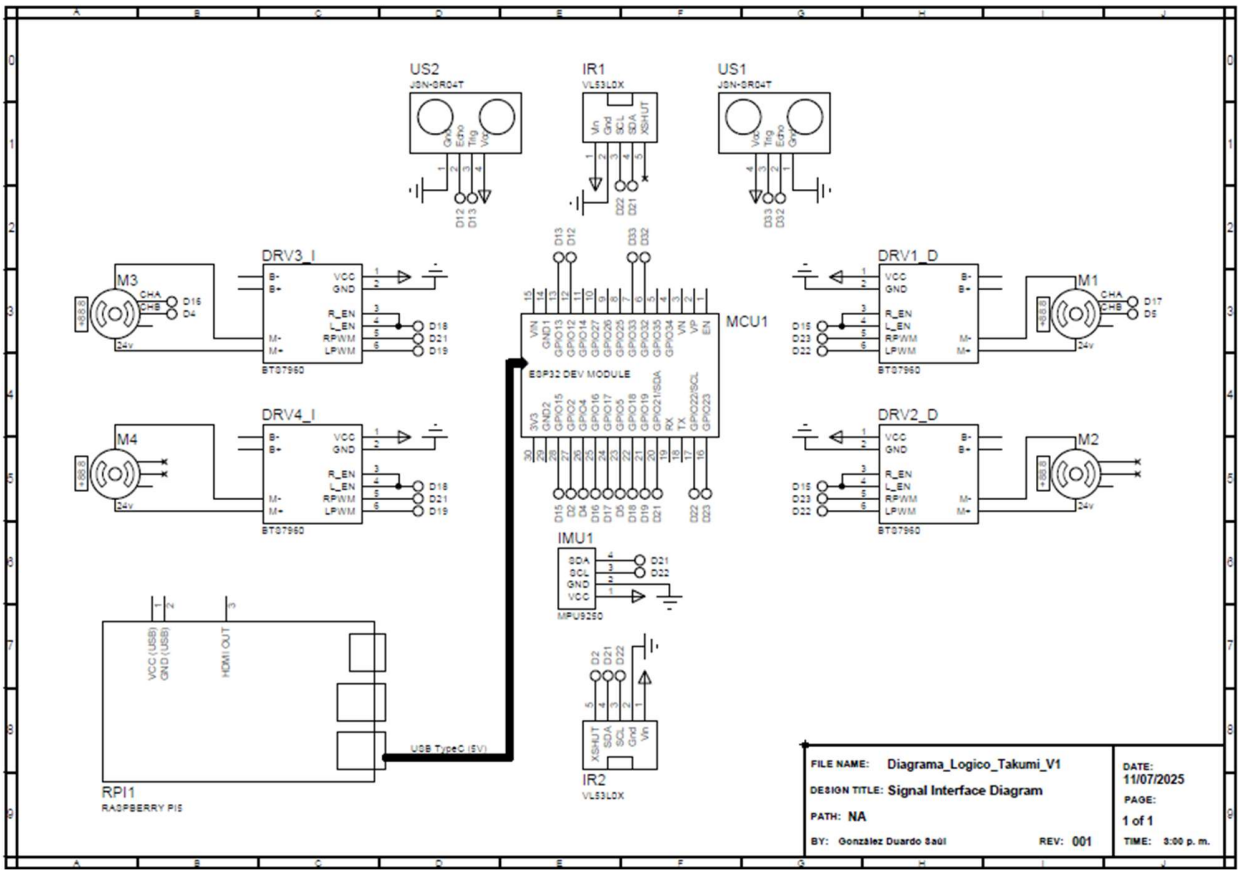


diagrama general de arquitectura.pdf



PDF

Electrical Power Diagram.pdf



Signal Interface Diagram.pdf