UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE GUAYAQUIL

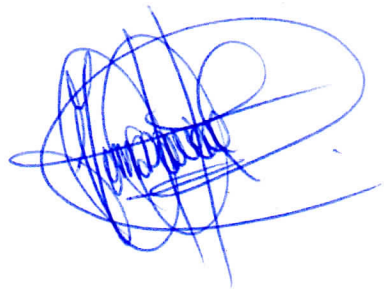
CARRERA DE MECATRÓNICA

DESARROLLO DE UN ROBOT AUTÓNOMO PARA MAPEO DE ENTORNOS USANDO UN SENSOR LIDAR

Trabajo de titulación previo a la obtención del
Título de Ingeniero en Mecatrónica

AUTORES: Julianno Anthony Román Verdesoto
Joan Andrés Vega Solano

TUTOR: Ing. Tomás Santiago Gavilánez Gamboa, MSc.



Guayaquil - Ecuador
2023 - 2024

CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE TITULACIÓN A LA UNIVERSIDAD POLITÉCNICA SALESIANA

Nosotros, **Julianno Anthony Román Verdesoto** con documento de identificación N° **0954280715** y **Joan Andrés Vega Solano** con documento de identificación N° **0930463617**, expresamos nuestra voluntad y por medio del presente documento cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos autores del **Dispositivo Tecnológico: DESARROLLO DE UN ROBOT AUTÓNOMO PARA MAPEO DE ENTORNOS USANDO UN SENSOR LIDAR**, el cual ha sido desarrollado para optar por el título de: Ingeniero en Mecatrónica, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En concordancia con lo manifestado, suscribimos este documento en el momento que hacemos la entrega del trabajo a final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana

Guayaquil, 20 de febrero del año 2024

Atentamente,

Julianno Anthony Román Verdesoto
0954280715

Joan Andrés Vega Solano
0930463617

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE TITULACIÓN

Nosotros, **Julianno Anthony Román Verdesoto** con documento de identificación N° **0954280715** y **Joan Andrés Vega Solano** con documento de identificación N° **0930463617**; manifestamos que:

Somos los autores y responsables del presente trabajo; y, autorizamos a que sin fines de lucro la Universidad Politécnica Salesiana pueda usar, difundir, reproducir o publicar de manera total o parcial el presente trabajo.

Guayaquil, 20 de febrero del año 2024

Atentamente,

Julianno Anthony Román Verdesoto
0954280715

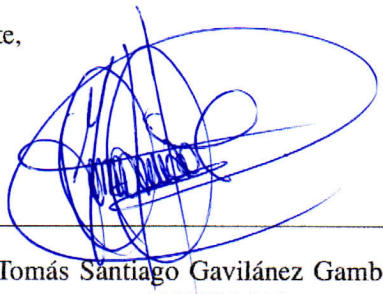
Joan Andrés Vega Solano
0930463617

CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

Yo, **Tomás Santiago Gavilánez Gamboa**, docente de la Universidad Politécnica Salesiana, declaro que bajo mi tutoría fue desarrollado el trabajo de titulación: **DESARROLLO DE UN ROBOT AUTÓNOMO PARA MAPEO DE ENTORNOS USANDO UN SENSOR LIDAR**, realizado por **Julianno Anthony Román Verdesoto** con documento de identificación N° **0954280715** y por **Joan Andrés Vega Solano** con documento de identificación N° **0930463617**, obteniendo como resultado final el trabajo de titulación bajo la opción **Dispositivo Tecnológico** que cumple con todos los requisitos determinados por la Universidad Politécnica Salesiana.

Guayaquil, 20 de febrero 2024

Atentamente,



Ing. Tomás Santiago Gavilánez Gamboa, MSc.
1802792646

DEDICATORIA

Dedico este trabajo a mis padres, por el apoyo constante y sacrificios que hicieron para brindarme la oportunidad de alcanzar mis metas. A mi familia y amigos, por su aliento y comprensión en los momentos difíciles. Agradezco a mis abuelos y tíos, cuyo constante apoyo y motivación fueron fundamentales para perseverar hasta el final. A mis profesores, por su guía, conocimiento y sabias enseñanzas que han marcado mi camino académico. Y a todas las personas que de alguna manera han contribuido a mi formación y crecimiento personal. Agradezco a mi tutor por su paciencia y valiosa asistencia durante todo el proceso de desarrollo de este proyecto.

Julianno Anthony Román Verdesoto

A mi tutor. Gracias Ing. Tomás Santiago Gavilánez Gamboa por todas esas jornadas de paciencia y sapiencia que le dedicó a mi proyecto, ya que sus directrices me ayudaron en los momentos de dudas, pude afianzar y reforzar mis conocimientos. A los jóvenes. A aquellos que vienen detrás de mí, espero que mi trabajo y el de todos mis compañeros sea el espejo donde se refleje ese mismo entusiasmo y ganas que yo tuve por estudiar esta hermosa profesión.

Joan Andrés Vega Solano

AGRADECIMIENTO

Este trabajo de titulación esta dedicado a mi madre, quien gracias a sus ánimos y sus esfuerzos logre terminar mis estudios universitarios. A mi padre, que me enseño sobre no rendirme y trabajar arduamente para conseguir mis objetivos. A mis abuelos y tíos porque gracias a sus ánimos y ayuda incondicional, estoy aquí en el ultimo paso para obtener mi titulo.

Julianno Anthony Román Verdesoto

A mis padres. Por ser los que me han forjado y convertido en la persona que soy en la actualidad. Siempre fueron y seguirán siendo mi apoyo emocional y material en todo sentido. El mejor ejemplo de perseverancia lo recibí de ustedes.

A mi abuelita. A pesar que ya no esté físicamente, estoy seguro que se sentiría muy feliz por este logro, ya que fue también un gran apoyo en mis primeros años, derrochando sobre mí su amor, cuidados y consejos para que siga adelante.

En general, a todos los demás, familiares, amigos y conocidos; que de una manera u otra me ayudaron a culminar mi carrera profesional. Muchas gracias.

Joan Andrés Vega Solano

I. Resumen

El proyecto consistió en la implementación de diversos tipos de algoritmos de mapeo y navegación en un robot autónomo, con el fin de determinar cuál de ellos satisfacía simultáneamente los requisitos de mapeo y navegación. El objetivo era el desarrollo de un robot autónomo destinado al mapeo de entornos utilizando un sensor láser denominado LIDAR. Se llevaron a cabo pruebas y evaluaciones de distintos algoritmos de mapeo y navegación en un entorno controlado. Se emplearon cuatro algoritmos de mapeo: Hector mapping, Gmapping mapping, Karto mapping y Cartographer mapping, así como dos algoritmos de navegación: rrt_exploration y otro destinado a la evasión y detección de obstáculos. Los resultados indicaron que el algoritmo de navegación que mejor evitaba los obstáculos era el de evasión y detección de obstáculos, mientras que el algoritmo de mapeo que proporcionaba un mayor detalle y presentaba menos distorsión de imagen al mapear el entorno fue Karto mapping. Para validar esto, se llevaron a cabo 50 pruebas con los cuatro algoritmos implementados, lo que confirmó su viabilidad.

Palabras claves: Robot autónomo, robot con sistema de mapeo, robot móvil, robot autónomo con sistema de mapeo, sensor lidar.

ABSTRACT

The project consisted of the implementation of various types of mapping and navigation algorithms in an autonomous robot, to determine which of them simultaneously satisfies the mapping and navigation requirements. The objective was the development of an autonomous robot intended for mapping environments using a laser sensor called LIDAR. Testing and evaluation of different mapping and navigation algorithms were carried out in a controlled environment. Four mapping algorithms were used: Hector mapping, Gmapping mapping, Karto mapping, and Cartographer mapping, as well as two navigation algorithms: rrt_exploration and another intended for obstacle avoidance and detection. The results indicated that the navigation algorithm that best-avoided obstacles was obstacle avoidance and detection, while the mapping algorithm that provided greater detail and presented less image distortion when mapping the environment was Karto mapping. To validate this, fifty tests were carried out with each of the algorithms, confirming their feasibility.

Keywords: Robot, autonomous robot, mapping robot, lidar, autonomous mapping robot.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

II. Introducción

En la actualidad, la integración de la vida cotidiana humana con la tecnología ha alcanzado niveles sin precedentes. Uno de los campos donde se destaca más está revolución tecnológica es en el ámbito de la robótica. Los robots normalmente usados en entornos industriales y de investigación, ahora están emergiendo como actores clave en sectores como: la atención médica, agricultura, logística y la educación. Es un fenómeno global que está transformando la relación humano-robot a nivel internacional, países de todo el mundo están invirtiendo recursos significativos en investigación, desarrollo y despliegue de robots y tecnologías relacionadas, con el objetivo de mejorar la eficiencia, la productividad y la calidad de vida de sus ciudadanos.

Ecuador ha experimentado un crecimiento significativo en el uso y la adopción de robots y tecnología en diversos sectores, la robótica puede ser empelada en distintos campos laborales: automatizaciones industriales, medicina, agricultura, educación y servicios públicos. A medida que la tecnología y robótica avanzan, surgen nuevos desafíos y oportunidades dando paso a nuevos espacios laborales y el país debe abordar para aprovechar plenamente el potencial de la revolución tecnológica.

Esta proyecto propone examinar el estado actual del uso de robots y su tecnología interna, analizando tanto los avances logrados como los desafíos que persisten en este campo. Debido a que la robótica no es un campo totalmente desarrollado, se tiene que la problemática en general es el funcionamiento correcto del robot autónomo con sistema de mapeo en un entorno controlado, ya que se tiene factores pueden afectarlo, ya sea en: la movilidad del robot, la búsqueda de rutas libres de colisiones en un entorno dinámico y la limitación de la visión debido a la altura excesiva o insuficiente del robot. El lugar específico es en la Universidad Politécnica Salesiana sede Guayaquil, donde se realizarán pruebas con distintos algoritmos, tanto de mapeo como de navegación para testear cual es el más eficiente a la hora de realizar el trabajo. La hipótesis consiste en desarrollar un algoritmo de navegación y mapeo basándose en la amplia biblioteca de ROS, el cual va a arrojar como resultado una imagen cartográfica del entorno que se escanea, con sus correctas delimitaciones. El objetivo central es desarrollar un robot autónomo para mapeo de entornos, usando un algoritmo de navegación y un sensor LIDAR.

III. Problemas de Estudio

En la actualidad, la tecnología ha logrado una amplia presencia en numerosos aspectos de la vida cotidiana, adentrándose incluso en la creación de robots autónomos. Estos dispositivos han sido empleados para resolver inconvenientes en diversas áreas, como se ha documentado en investigaciones previas. Por ejemplo, se ha constatado su utilización en situaciones de desastres naturales [1], en tareas de búsqueda y rescate [2], en exploraciones de diversa índole [3] en el ámbito de los servicios de entrega [4], en labores de limpieza [5], así como también en el área de agricultura [6], entre otras aplicaciones.

El principal desafío reside no solo en la construcción del robot, sino también en la adquisición de los materiales necesarios, dado que Ecuador no cuenta con la capacidad de fabricarlos internamente. Aunque la transferencia de tecnología de países desarrollados se ha considerado como un enfoque para reducir la brecha de desarrollo entre naciones [7],es importante tener en cuenta que el transporte de dichos materiales también conlleva costos significativos y requiere más tiempo del estimado para iniciar la construcción de los robots. Además, el desarrollo de esta tecnología presenta dificultades debido a la existencia de diversos sistemas que permiten a cada robot autónomo desempeñar acciones específicas para llevar a cabo eficientemente su labor, si bien estos enfoques también pueden resultar costosos [8].

La aplicación de esta tecnología puede ser altamente beneficiosa; no obstante, los robots autónomos se han enfrentado a varios desafíos, uno de estos desafíos reside en su capacidad de respuesta ante situaciones que requieren toma de decisiones rápidas, como la selección de rutas o la búsqueda de alternativas cuando se encuentran en un camino sin salida [3]. Otros puntos de consideración son dificultades en la capacidad de trazar rutas adecuadas para la navegación o en la identificación precisa de los objetos buscados [2]. En la investigación realizada por [9], se menciona otros tipos de problemas que afectan la movilidad del robot, como la búsqueda de rutas libres de colisiones en un entorno dinámico, la prontitud en la adecuada exploración de los terrenos agrícolas con el fin de detectar posibles daños o amenazas, así como la limitación de la visión debido a la altura excesiva o insuficiente del robot, lo que podría afectar su control del área [5].

Aunque existen métodos para abordar dichos problemas, presentan la desventaja de requerir una selección computacional de entornos eficientes para el aprendizaje del robot, lo que implica el aumento de otros costos [3].

En el Ecuador, el desarrollo de robots autónomos y los subsistemas que lo integran como el mapeo son muy limitadas y aún se encuentra en desarrollo, por lo que no se producen a escala comercial, consecuentemente son inasequibles. La falta de investigación en esta área generaría un retraso en la tecnificación de sectores estratégicos para el país como son la agricultura y minería. Además de, crear una dependencia tecnológica de otros países.

IV. JUSTIFICACIÓN

A nivel mundial, la implementación de robots autónomos equipados con sistemas de mapeo brinda la oportunidad de facilitar tareas en diversas áreas de la vida cotidiana. Su aplicabilidad puede ir desde tareas de alto riesgo, mejora a sectores agrícolas [6], e incluso en tareas del hogar [10].

En una investigación realizada por [11] se planteó diseñar un robot autónomo para el sector agrícola, el cual trataba de asumir el movimiento del ser humano al momento de realizar la plantación, este tuvo una serie de pruebas las cuales cumplió al momento de detectar el surco donde se realizaría la plantación.

En un estudio llevado a cabo en una mina al norte de Nevada, se implementó un robot autónomo tipo dron que permitió mapear y realizar una reconstrucción tridimensional (3D) de la mina. Esta estrategia logró aumentar significativamente la eficiencia en la exploración autónoma de minas, proporcionando una mayor comprensión del entorno minero [12]. Asimismo, se ha propuesto otro estudio de un robot autónomo en el área de limpieza, según lo presentado por [5] este robot, equipado con sistema de mapeo, tiene la capacidad de evitar obstáculos durante el proceso de limpieza. Esta característica le permite aprender y optimizar su tiempo de limpieza, mejorando así su eficiencia en esta tarea específica.

A partir de la información recopilada, se evidencia que los robots equipados con sistemas de mapeo tienen el potencial de ser sumamente beneficiosos. Su implementación permitiría una planificación y selección de rutas más eficiente, lo que generaría ventajas significativas en diversos campos, facilitando las tareas y reduciendo el riesgo laboral asociado. En vista de esto, se plantea diseñar un prototipo de robot autónomo móvil con sistema de mapeo, que mediante la aplicación permita tener un entorno controlado como demostración del funcionamiento y las capacidades de esta tecnología.

V. Objetivos

V-A. *Objetivo general*

- Desarrollar un robot autónomo para mapeo de entornos, usando un algoritmo de navegación y un sensor LIDAR.

V-B. *Objetivos específicos*

- Implementar los sistemas mecánicos, electrónicos y de control para el funcionamiento del Robot móvil.
- Implementar un algoritmo de navegación para mapeo de entornos con detección de obstáculos usando un sensor Lidar.
- Validar el sistema mediante pruebas controladas en diferentes entornos.

V-C. Tabla de objetivos

Tabla I
MATRIZ DE OBJETIVOS

OBJETIVO	PLANTEAMIENTO	META	INDICADOR
OE.1	Implementar los sistemas mecánicos, electrónicos y de control para el funcionamiento del Robot móvil.	Navegar dentro de un entorno controlado detectando un 85 % de los obstáculos.	Porcentaje de obstáculos detectados.
OE.2	Implementar un algoritmo de navegación para mapeo de entornos con detección de obstáculos usando un sensor Lidar.	Reducir el tiempo de mapeo en 10 iteraciones del algoritmo.	Número de iteraciones de mapeo.
OE.3	Validar el sistema mediante pruebas controladas en diferentes entornos.	50 pruebas de Mapeo de entorno con un error hasta del 15 %.	Porcentaje de error.

VI. Marco teórico referencial

VI-A. Robots autónomos

La tecnología abarca diversos aspectos de la vida cotidiana y se encuentra estrechamente conectada con los avances en el campo de la robótica. Un ejemplo destacado de relación se encuentra en los robots autónomos, los cuales desempeñan un papel en distintas áreas, tales como la agricultura [11], la minería [12], los servicios de entrega [4] y la limpieza [10], entre otros campos de aplicación.

Dentro de la categoría de los robots autónomos, se distinguen diversos tipos, incluyendo aquellos de tipo aéreo, como los drones [6], así como los terrestres, que pueden contar con sistemas de ruedas [11], capacidad de trepado o escalada [10], entre otras configuraciones. Estos robots han experimentado un gran avance, ejemplificado en los robots de entrega autónomos (ADR, por sus siglas en inglés), los cuales han experimentado un rápido crecimiento en una variedad de aplicaciones en los últimos años. Estos robots encuentran su principal aplicación en la entrega de alimentos y artículos de menor envergadura, tal como lo subraya [4].

En este contexto, se observa una amplia gama de sistemas mecánicos y electrónicos, junto con el uso de diversos programas informáticos para el control y manejo de los robots autónomos. Ejemplo de ello es el enfoque presentado por [10], quien introduce un robot limpiador que emplea redes neuronales convolucionales profundas para detectar escaleras, desencadenando así el funcionamiento de su sistema mecánico de descenso. Adicionalmente, se presenta el caso del robot limpiador propuesto por [13], el cual ha sido desarrollado inicialmente a través de un diseño en CAD para lograr la adecuada distribución de sus componentes. Este sistema electrónicamente equipado incorpora tres sensores infrarrojos estratégicamente ubicados en la parte frontal y lateral del robot, posibilitando la evasión de obstáculos. La gestión de este diseño es dirigida por un controlador Raspberry Pi.

A pesar de la eficacia observada en estos sistemas, diversas metodologías y técnicas han surgido con el propósito de optimizar aún más la autonomía de los robots. En la invetigación de [5] se destaca la importancia crucial de la capacidad de visión en cualquier robot autónomo, ya que esto determina el grado de control del entorno en el que opera. Para abordar este requisito esencial, se han desarrollado técnicas que buscan mejorar la capacidad visual del robot. En consecuencia, [9] propone un método para mejorar la visión el cual está basado en el Aprendizaje Profundo por Reforzamiento (DRL, por sus siglas en inglés) junto con una red de atención gráfica. Esta metodología se enfoca en analizar la distancia mínima promedio y utilizar los datos de frecuencia de incomodidad para fundamentar decisiones de movimiento seguras y eficientes. Así mismo, [14] propone una técnica que implica que el robot sea capaz de identificar y priorizar tareas de mayor relevancia, abordando una de las limitaciones recurrentes en estos robots: la duración de la batería. Este enfoque fomenta la gestión óptima de la energía y la utilización eficiente de los recursos disponibles, lo que contribuye a mejorar el rendimiento global del robot.

Para finalizar, con la información previamente analizada se puede notar la variedad de robots autónomos presentes en el mundo, y a su vez ver las distintas maneras de elaborar uno para las distintas áreas de desarrollo, con esto en mente se podría decir que la mejor elección seria elaborar un robot autónomo móvil de cuatro ruedas debido a su rentabilidad y facilidad de poder maniobrar este tipo de robots, lo que es primordial para la movilidad del robot autónomo, a partir de aquí se procederá a analizar otras partes necesarias y seleccionar las características mas óptimas para desarrollar el proyecto.

VI-B. Mapeo

El mapeo es una de las características mas importantes cuando se habla de localización, debido a que es una herramienta básica que permite realizar cartografía, este método se basa en marcar trazos, recoger información y diagnosticar aspectos de un área terrestre, representando la parte superficial escaneada y convirtiendo la información geoespacial en una representación plana, mediante un algoritmo que descifre dicha representación [15]. Además, se enfoca en calcular las trayectorias mientras se construye un mapa del entorno, los sensores determinan la ubicación y posición del robot, considerando que cualquier error en la odometría puede afectar significativamente el proceso de construcción del mapa. La odometría permite una mayor precisión en la navegación de los robots, combinando con información visual proveniente de cámaras o sensores, proporcionando una localización más precisa [16]. Se debe considerar que la calidad del mapeo está vinculada a la precisión de los sensores y a la eficacia de los algoritmos de post-procesamiento de datos [15]. Las estimaciones cartográficas se generan a través de un algoritmo SLAM, el cual demuestra su utilidad tanto en el control como en la navegación del robot [17].

Como fue previamente mencionado el mapeo es una herramienta que permite recopilar información, esto es de gran utilidad en aplicaciones que abarcan áreas como la posición, navegación, localización y respuesta a emergencias. Lo que facilita la creación de mapas detallados de entornos, en este contexto hay que conocer que el mapeo se clasifica en manual y automático. En el mapeo manual, son datos que se recopilan y registran a través de encuestas y cargas de información manuales. Ejemplos de esto son servicios como Google Maps o OpenStreetMap. Por otro lado, el mapeo automático es más eficiente, ya que detecta estructuras y las reconstruye automáticamente utilizando datos de escaneo en un entorno digital [18], como se muestra en la Figura 1.

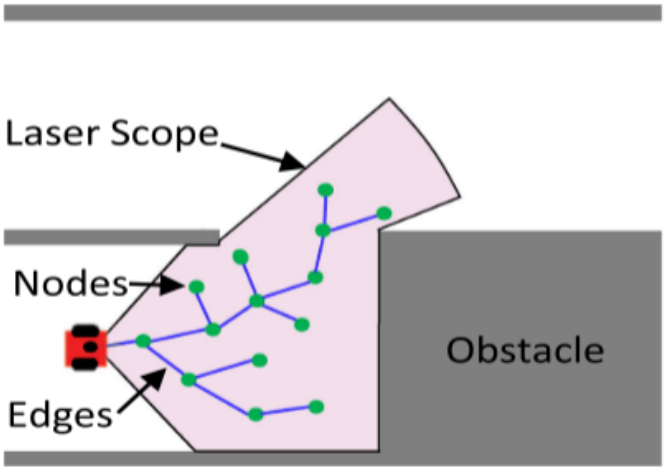


Figura 1. Función del mapeo automático.
Fuente: [3]

En [18] se subraya que los métodos de mapeo automatizados ofrecen eficiencia en la producción y una amplia aplicabilidad en la creación de mapas. En los últimos años, el uso de tecnología de escaneo láser móvil ha ganado relevancia, particularmente en entornos interiores debido a su alta precisión y fácil accesibilidad.
Estos avances tecnológicos han posibilitado el uso del mapeo para evaluar la degradación de materiales y el desgaste de las estructuras en edificaciones, lo que permite predecir daños causados por fenómenos climáticos [19].

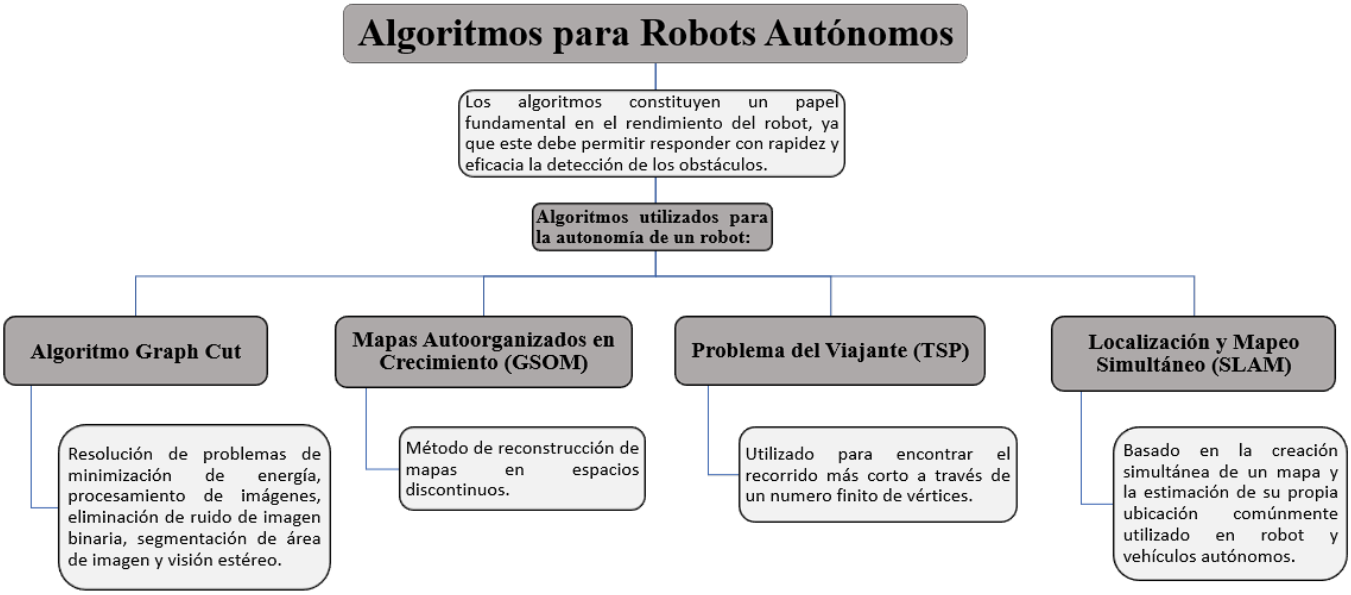


Figura 2. Algoritmos usados en robots autónomos.
Fuente: Autores

Es importante subrayar que el rendimiento del robot no solo está vinculado al sistema implementado, sino también al algoritmo utilizado para la detección de obstáculos en su entorno. En este contexto, se puede observar en la Figura 2, los diferentes algoritmos que contribuyen a obtener escaneos precisos de entornos, como el algoritmo Graph Cut, utilizado para resolver problemas de minimización de energía, procesamiento de imágenes, eliminación de ruido de imagen binaria, segmentación del área de la imagen y visión estéreo [20]. Se mencionan dos técnicas: smoothness energy utilizado para medir la suavidad entre pares de pixeles unidos y data energy usado para medir la diferencia entre pixeles en base a las disparidades asumidas [21]. La imagen se modela como una red de flujo donde varios píxeles se asocian gracias a los nodos y los pesos de los arcos definen la similitud entre los píxeles, lo que se busca es un corte de valor mínimo [21]. Se arma un grafo donde cada nodo representa un pixel de la imagen, en la Figura 3 se representa una red de flujo que contiene dos vértices llamados S (Fuente) y T (Pozo), un conjunto de etiquetas que representan los posibles valores de disparidad, los pesos de las aristas (valores de energía definidos) y las capacidades de cada arco [21].

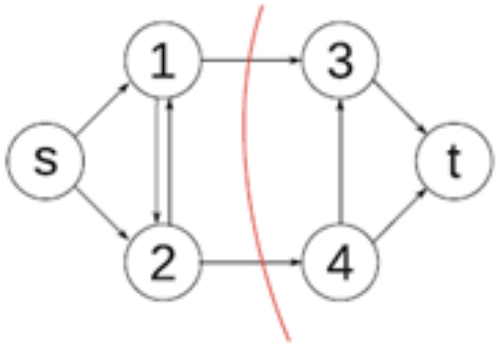


Figura 3. Algoritmo Graph Cut.
Fuente: [20].

El siguiente algoritmo a mencionar, es el de los Mapas Autoorganizados en Crecimiento (GSOM, por sus siglas

en inglés) se presentan como un método de reconstrucción de mapas mediante un conjunto de puntos cuando el espacio es discontinuo [22]. Este método adapta su estructura automáticamente a los datos, para conseguir una representación óptima de los mismos, crecen en tamaño según la distribución de los datos para conseguir una representación correcta [23]. GSOM permite acomodar y representar mejor la distribución de datos en un espacio dimensional. En la Figura 4, se detalla que la capa inferior tiene una representación de datos con mayor detalle que la capa superior.

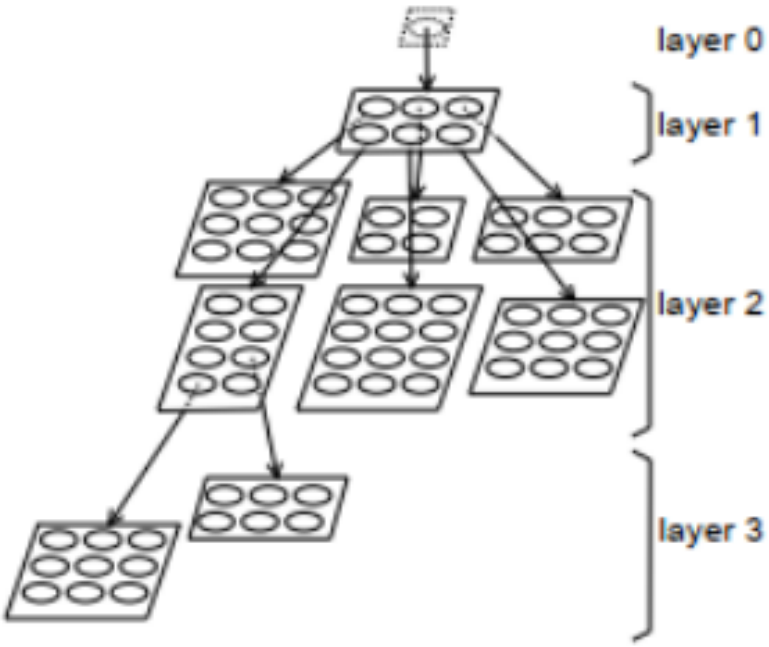


Figura 4. Jerarquía de GSOM de tamaño independiente
Fuente: [23].

Otro de los algoritmos, es el Problema del Viajante (TSP, por sus siglas en inglés) se utiliza para abordar problemas que involucran encontrar el recorrido más corto posible a través de N vértices, visitando cada uno de ellos [24]. Se tiene dos tipos de TSP: TSP simétrico es cuando dos puntos tienen caminos en dirección opuesta, al momento de calcular la distancia entre sí, se forma un grafo no dirigido y esto reduce a la mitad de posible número de soluciones y el TSP asimétrico es cuando no existen caminos en ambas direcciones o distancias diferentes, formando un grafo dirigido. Los accidentes de tráfico, las calles de un solo sentido o los peajes, son ejemplos la asimetría [25]. El método comienza en un vértice específico y termina cuando se visita una vez el resto de los vértices. Si no existe camino entre un par de ciudades, se añade arbitrariamente una arista larga para completar el grafo sin afectar el recorrido como se muestra en la Figura 5.

Por ultimo, el algoritmo de Localización y Mapeo Simultáneo (SLAM, por sus siglas en inglés) se basa en la creación simultánea de un mapa y en la estimación de su propia ubicación en un entorno desconocido [26]. El sensor más usado a la hora de implementar SLAM es el escáner láser debido a su precisión, rapidez y poco procesamiento que necesitan los datos a la salida, estos sensores son conocidos por ser caros y suelen tener inconvenientes con el vidrio y el agua debido a su poco alcance [27].

Como es conocido, también existen algoritmos de código abierto, como el Gmapping SLAM, utilizado para entornos simples. Este algoritmo se basa en construir mapas utilizando únicamente información láser y utiliza

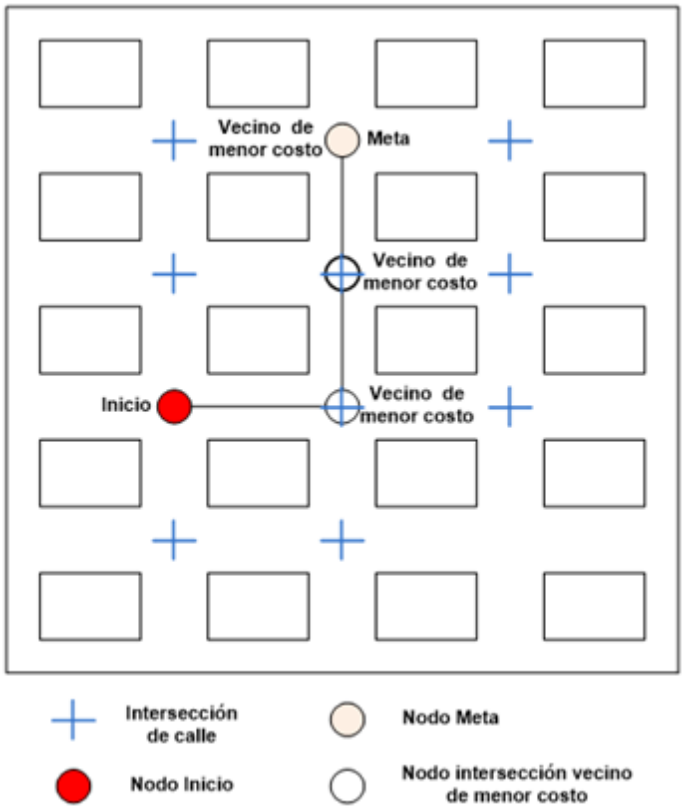


Figura 5. Distancia mínima entre dos puntos
Fuente: [25].

el método de filtros de partículas [28]. Aunque el método mencionado presenta limitaciones notables, como su susceptibilidad al ruido, que puede causar perturbaciones, y su incapacidad para detectar obstáculos como sillas y mesas durante el escaneo, considerándolos como paredes homogéneas [29]. De manera similar, Cartographer es otro algoritmo de código abierto, emplea SLAM láser 2D para mapeo en entornos de trabajo complejos, utilizando datos del odómetro y de la IMU para estimar la trayectoria [28].Según [30] los resultados de Gmapping son precisos respecto a la creación de mapas, ya que se basa en gráficos y cuadrículas de ocupación RBPF, se usa una cantidad pequeña de partículas RBPF en su algoritmo, permitiendo la reducción de recursos computacionales para el muestreo y la generación de mapas.

Para desarrollar el proyecto, se ha optado por la utilización del método SLAM, el cual ha sido empleado por más de 30 años en la navegación de robots móviles autónomos en entornos desconocidos. Durante este tiempo, se han desarrollado diversos algoritmos y filtros para mejorar su eficiencia [28]. El método SLAM posibilita la utilización de sensores como LIDAR para la construcción del mapa del entorno durante la navegación hacia una ubicación específica. Es imperativo emplear sensores y algoritmos de alta precisión en tiempo real para llevar a cabo el mapeo de manera efectiva [29]. Además, este sistema permite la utilización de sensores pequeños y de bajo costo, en la Figura 6 se puede observar el resultado del objeto físico mapeado logrado mediante la aplicación de uno de estos algoritmos [26].

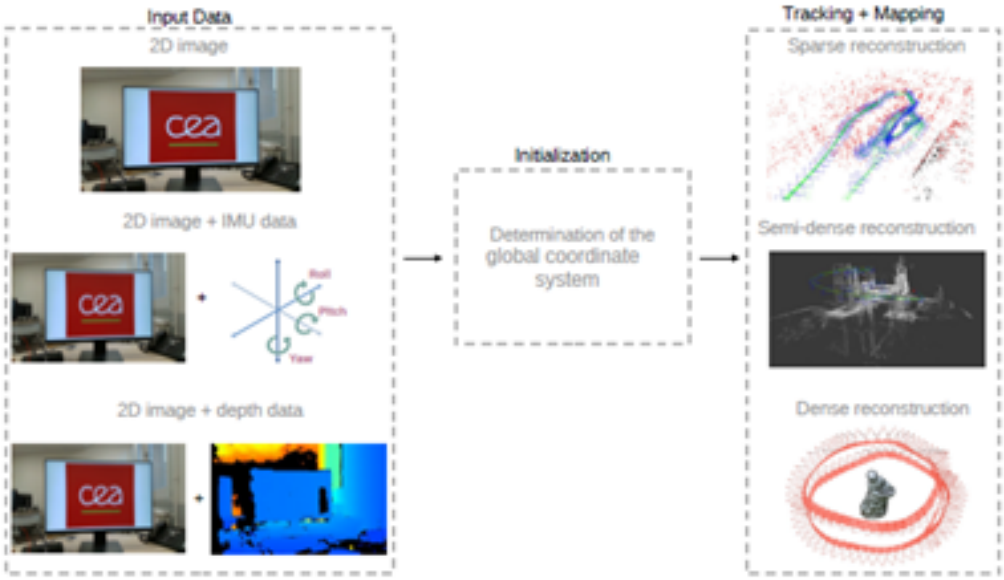


Figura 6. Componentes generales de un SLAM basado en elementos visuales
Fuente: [26].

VI-C. Robots autónomos con sistema de mapeo

Los robots autónomos con sistemas de mapeo representan una innovación notable debido a su amplia aplicación en diversos campos de la ingeniería. Se utilizan en el mapeo de interiores para llevar a cabo inspecciones de tuberías subterráneas, como lo destaca [31], quienes señalan la necesidad de dotar a los robots de autonomía en sus operaciones. Con este fin, se busca emular comportamientos biológicos y promover la intercomunicación entre los robots para lograr una mayor precisión en sus tareas.

Existe una amplia gama de algoritmos que los robots autónomos pueden emplear para su operación, la elección y elaboración de los mismos está condicionada por las especificaciones técnicas y las tareas que se pretendan ejecutar. Por ejemplo, en la investigación realizada por [32], propone un algoritmo basado en Mapas auto-organizados en crecimiento (GSOM, por sus siglas en inglés) que permite a los robots actuar de acuerdo con el conocimiento adquirido a partir de datos previamente recopilados. Además de resaltar la importancia del desarrollo cognitivo en los robots móviles autónomos.

La relevancia de los algoritmos en el adecuado funcionamiento de los robots es innegable, por lo cual [33] propone dos tipos de algoritmos distintos: uno es el algoritmo de Seguimiento y mapeo paralelos (PTAM, por sus siglas en inglés), que ofrece un rendimiento sólido y una capacidad de seguimiento, y el otro es el algoritmo de cono invertido, que proporciona una detección más precisa de obstáculos, aunque demanda una segmentación más precisa de los mismos. El objetivo final es generar diversas rutas mediante la utilización de los datos recopilados, lo que favorece un mejor rendimiento en las tareas desempeñadas por el robot. Otro de los aspectos que permiten a los robots autónomos con sistema de mapeo funcionar de forma eficiente son los métodos que se emplean para que esto mismos pueden aprender y mejorar su desempeño. Por esta razón, [9] en una investigación presenta un método en el cual se utiliza un robot autónomo para evaluar grietas, tanto naturales como producidas por la acción humana. Este robot cuenta con un sistema de brazo robótico con múltiples grados de libertad, cámaras para visión artificial y un sensor ultrasónico para medir el ancho y la profundidad de las grietas detectadas.

Además, en un caso realizado por [34] promueve, el uso de un módulo de agregación de características multiescalar. Este método fusiona la información capturada por las cámaras y se encarga de rellenar los vacíos generados por lecturas defectuosas, mejorando la precisión de las percepciones sensoriales del robot, lo que contribuye a una

mejor toma de decisiones y navegación eficiente. Otro método que es relevante mencionar es la recreación virtual de interiores en 3D, de acuerdo con [35], estos robots se esfuerzan por obtener diversas perspectivas mediante movimientos manuales o el seguimiento de rutas predefinidas. Aprovechando herramientas virtuales, se detectan intensidades, profundidades y máscaras semánticas para generar representaciones realistas de los interiores.

Así mismo, en la investigación realizada por [36], menciona que los robots bípedos muestran un rendimiento superior debido a su posición relativa respecto al suelo, facilitando la recopilación de datos adicionales. Estos robots bípedos se han empleado en la modelización de interiores y en la evaluación de posibles mejoras estructurales mediante el uso del mapeo a través del sensor LIDAR. No obstante, es importante tener presente que estos robots no han sido diseñados específicamente para eludir obstáculos, lo que limita su capacidad para escanear áreas que no estén casi despejadas o en zonas estables. Otro caso presentado donde se puede observar ciertas mejoras son los robots autónomos móviles con brazo robótico acoplado, estos brindan asistencia autónoma a pacientes con movilidad reducida para prevenir caídas. [37] describe que estos robots cuentan con un algoritmo de control cerrado que les permite seleccionar una trayectoria factible, mientras utilizan un dispositivo LIDAR para medir distancias y orientaciones. Además, se destacan los brazos robóticos autónomos que emplean una técnica adaptativa en su visión, la cual mejora su estabilidad y adaptabilidad, acelerando así el tiempo de respuesta aplicado a los servomotores, tal como señala [38].

Una vez conocido las diferentes algoritmos que se han implementado en otros robots autónomos con sistema de mapeo, se puede ver que características permiten a estos robots realizar un buen desempeño, con esto claro se pretende buscar y utilizar un algoritmo que permita al robot tener un mapeo y navegación exitosa. Por último, se propone desarrollar un robot autónomo con un sistema de mapeo. Con la siguiente distribución en el nivel superior se ubica el sensor Lidar para garantizar una visión despejada y facilitar la detección del entorno circundante. En el nivel intermedio, se sitúan la placa de control y el Raspberry Pi, ambos conectados entre sí. Esta disposición protege contra posibles daños que podrían ocurrir al estar expuestos en la parte superior o inferior del robot móvil. Asimismo, simplifica las conexiones con los motores en la parte inferior y el sensor Lidar en la parte superior. En el nivel inferior del robot se disponen exclusivamente los motores, las ruedas y la batería. Esta disposición responde a la necesidad de tener los motores y las ruedas cerca del suelo para facilitar la movilidad del robot. La ubicación de la batería en esta sección se justifica por la disponibilidad de espacio y su proximidad debajo de la placa de control, lo que permite suministrar de energía al robot.
La elección de esta estructura para los componentes se debe a la necesaria versatilidad y distancia requeridas en las conexiones esto permite tener distribuido los componentes para mejor facilidad y accesibilidad, ya que el robot debe ser agradable para el usuario.

VI-D. Interacción humano-robot

La interacción robot-humano es un aspecto importante que la mayoría de los robots descuida en sus funcionalidades. Sin embargo, es fundamental que los robots sean capaces de interactuar de manera efectiva con los usuarios, ya sea a través de estímulos auditivos o visuales. Una investigación en particular ha explorado el impacto emocional de los colores, y en base a ello, [39] ha propuesto la incorporación de luces expresivas en un robot limpiador de suelos. Estas luces permiten generar un nivel de atención al usuario al mostrar información relevante, como el estado de la batería, control de daños o posibles averías del robot.

Por otro lado, cuando se trata de sistemas de mapeo de interiores con robots móviles autónomos, [40] describe que su función principal es la creación de mapas semánticos utilizando un enfoque cliente-servidor. Este enfoque permite un análisis y diseño estructural más preciso, y los mapas resultantes pueden visualizarse digitalmente utilizando programas de renderización como Unity 3D, Processing, entre otros. Esta visualización digital proporciona una representación clara y detallada del entorno mapeado, lo que facilita su análisis y comprensión.

VII. METODOLOGÍA

El propósito de este proyecto es el desarrollo de un robot autónomo con un sistema de mapeo integrado. Con este fin, se realizó una exhaustiva investigación de diversos diseños de robots autónomos y opciones de robots de mapeo disponibles en el mercado. Se examinaron funciones fundamentales, estructuras y algoritmos de estos robots, respaldados por la revisión de artículos científicos, vídeos y tesis relacionados para obtener una comprensión detallada de los principios operativos y enfoques utilizados en este ámbito. La evaluación de varios diseños determinó la configuración adecuada del robot para satisfacer las necesidades del proyecto.

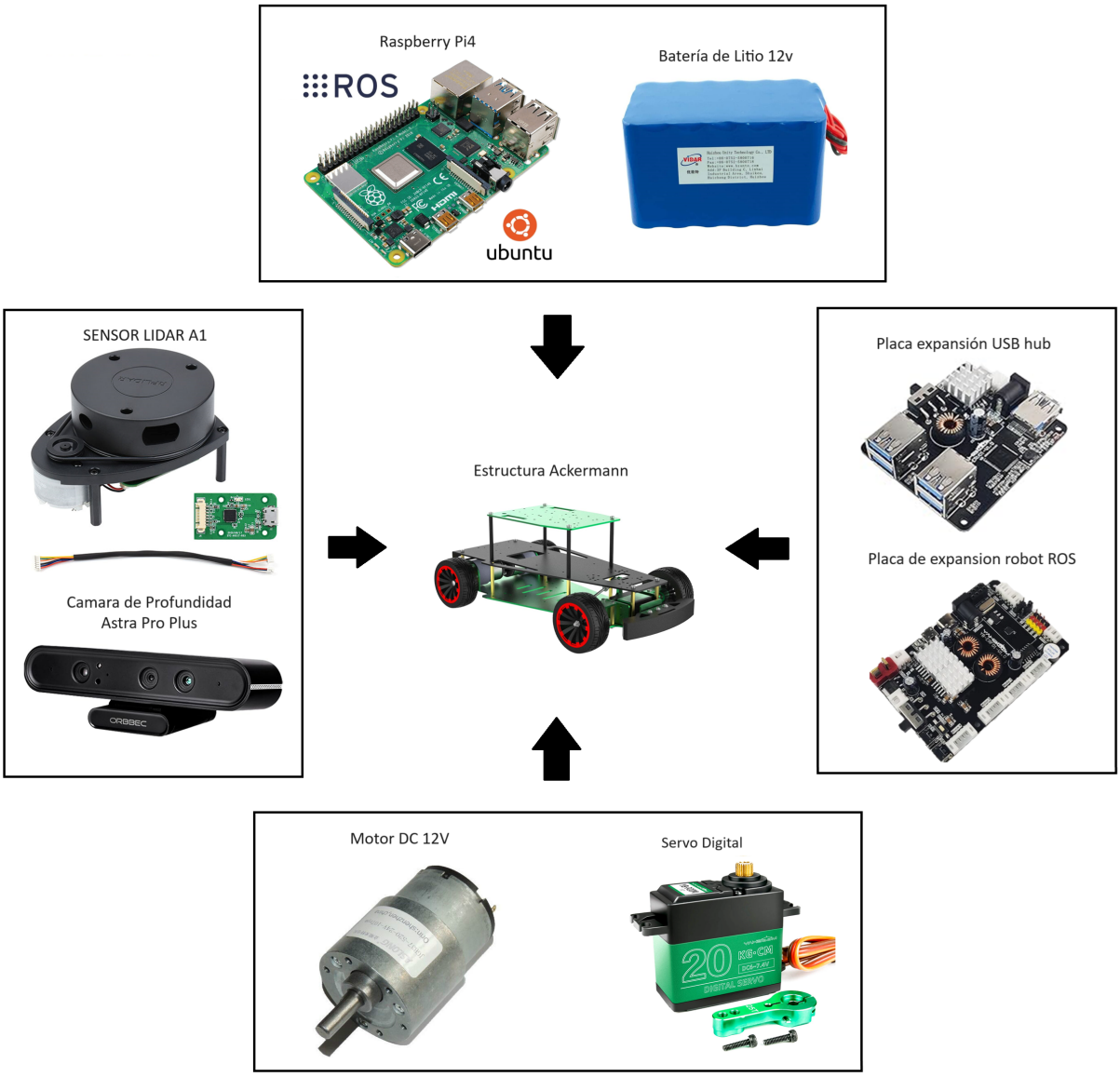


Figura 7. Diagrama de componentes del robot autónomo móvil.
Fuente: Autores.

En la Figura 7, se presenta el diagrama de componentes del sistema del robot autónomo. La conexión entre el sensor LIDAR y la cámara con respecto al Raspberry Pi 4 es crucial, ya que son los componentes responsables de la percepción y lectura del entorno. Esto posibilita que el robot tome decisiones para planificar y ajustar su ruta de manera autónoma. La placa de control se compone de la placa de expansión USB Hub, facilitando la conexión

entre el Raspberry Pi 4 y la placa de expansión ROS. Este enlace posibilita la transmisión de comandos a los motores y al servomotor, controlando así la movilidad del robot. Así mismo, la batería se conecta a la placa de expansión ROS para suministrar energía a todo el sistema.

El funcionamiento del robot implica su capacidad para moverse en el entorno deseado, requiriendo de esta manera el reconocimiento de su ubicación y la detección de los obstáculos a su alrededor. El sistema debe evitar obstáculos previamente detectados para retomar la ruta planificada [41]. Este tipo de robot suele implementarse con el Sistema de operación robótica (ROS, por sus siglas en inglés), un sistema de código abierto que mejora la capacidad de toma de decisiones [42]. Además, la aplicación de estos sistemas se realiza en plataformas que operan bajo el entorno del sistema operativo Linux. Por lo cual, se tomó la decisión de utilizar el programa Ubuntu, dado que posibilita la instalación del Sistema ROS y los paquetes necesarios para el desarrollo del proyecto.

Una vez establecido este marco, se procederá a instalar los paquetes ROS, que permiten simular al robot autónomo y la lectura del sensor LIDAR. Esto permitirá realizar una demostración simulada del funcionamiento del robot autónomo.

VII-A. Diseño Mecánico

La estructura del robot debe cumplir ciertas condiciones importantes que permitan al robot móvil poder desplazarse con rapidez, eficiencia y soportar el peso de sus componentes como el sensor lidar, la cámara, la batería, el Raspberry Pi4 y sus placas de expansión para el control de los motores.
Por esta razón, a continuación se procederá a realizar el análisis estático de la estructura del robot ya que esto garantiza que el diseño sea seguro cuando se lo compara con las propiedades del material. Mediante un software de modelado CAD se comprobara que la estructura sea segura tomando en cuenta las tensiones y deformaciones.
Lo primero que se realizara para proceder con el análisis estático es obtener el peso que soportara la base del robot tomando en cuenta el prototipo con la mayor cantidad de componentes. Se obtuvo que pesa aproximadamente 2722 [g], transformando ese valor a kilogramo da un valor de 2,72 [kg]. Con este dato aproximado se decidió redondear el valor a 3 [kg], con esto se calcula la fuerza que se ejerce sobre la base del robot que equivale a 29,43 [N], como se puede ver en la ecuación 6.

$$Efy = 0 \tag{1}$$

$$N - W = 0 \tag{2}$$

$$N = W \tag{3}$$

$$W = m * g \tag{4}$$

$$W = (3) * (9,81) \tag{5}$$

$$W = 29,43N \tag{6}$$

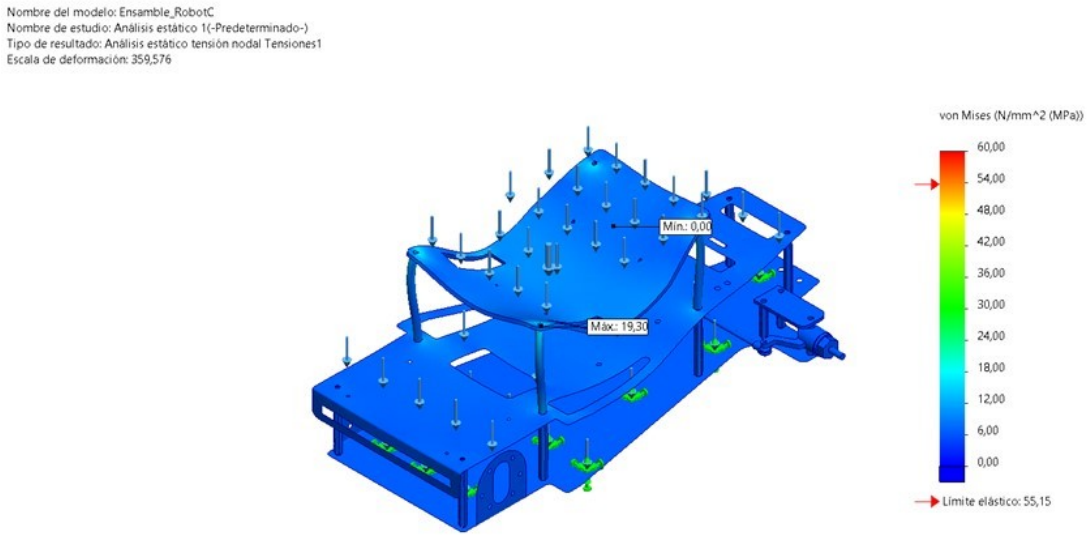


Figura 8. Análisis de Von Mises.
Fuente: Autores.

En la Figura 8 se muestra el valor máximo de Von Mises, el cual ser menor al limite elástico del material para garantizar su seguridad. Como se puede ver en la imagen el análisis muestra el valor máximo de 19,30 $[MPa]$ del esfuerzo de Von Mises que es menor al limite elástico del material de aluminio 6061 correspondiente a un valor de 55,15 $[MPa]$, cumpliendo con la condición de que el esfuerzo de Von Mises es menor que el limite elástico.

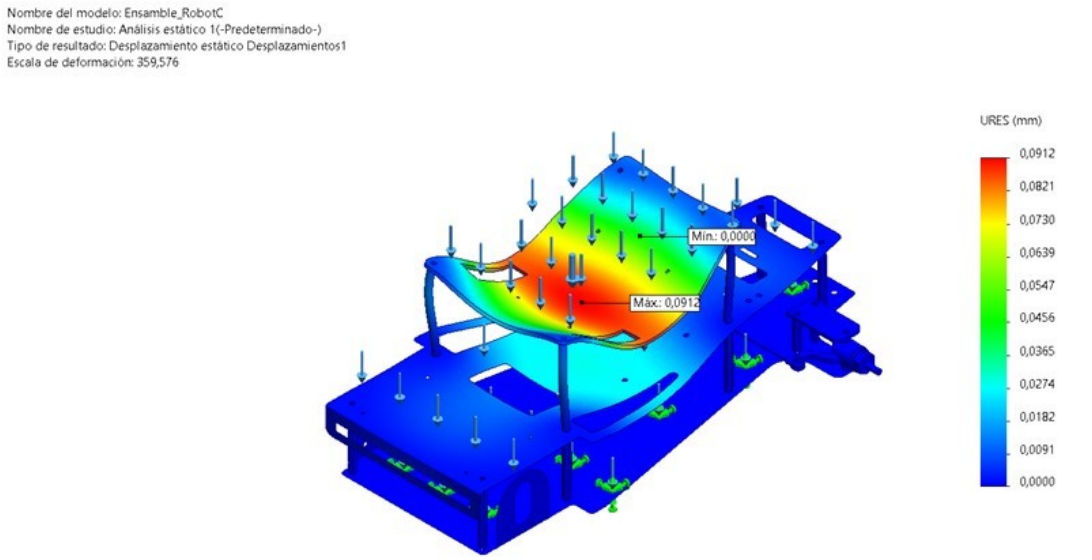


Figura 9. Desplazamiento estático.
Fuente: Autores.

Se puede observar en la Figura 9, el desplazamiento que sufre la estructura del robot, donde la parte más afectada es la base superior. El robot al ser sometido a una fuerza de 29,43 $[N]$, sufre un desplazamiento de 0,0912 $[mm]$, esto indica que sufre un desplazamiento mínimo, lo que no afectaría mucho a la estructura del robot.

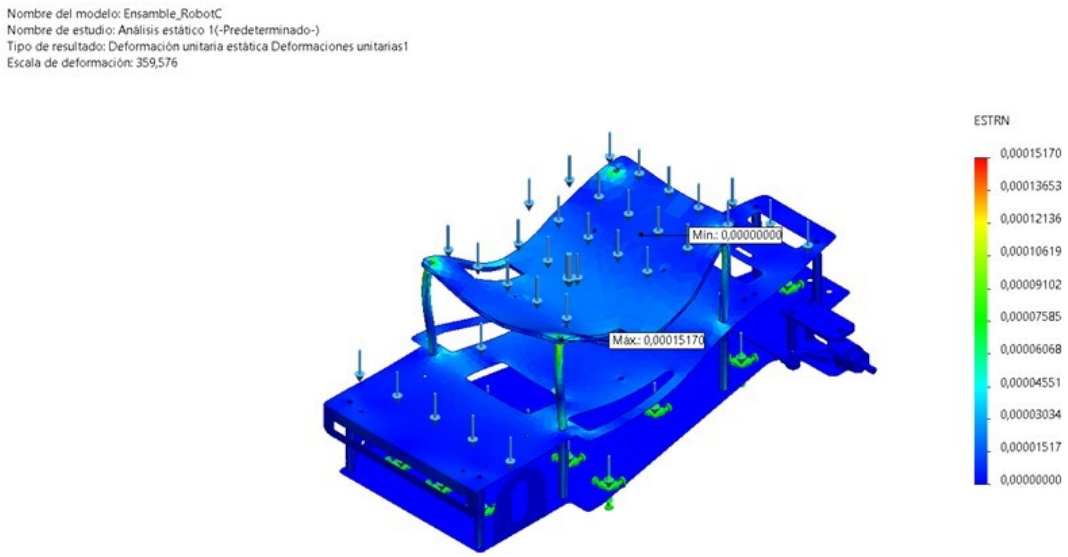


Figura 10. Deformación unitaria estática.
Fuente: Autores.

La deformación que sufre la estructura del robot mostrada en la Figura 10, es deformación imperceptible debido a que el robot esta diseñado para soportar cargas dinámicas. En la Figura 11 se muestra el factor de seguridad de la base, donde se puede observar el valor mínimo es de 2.857 este valor es mayor a 1, lo que indica que es seguro. Este valor se obtuvo de la división del limite elástico de 55,15 $[MPa]$ con el valor máximo del esfuerzo de Von Mises de 19,30 $[MPa]$, cabe recalcar que este es un análisis estático, en el cual no se han tenido en cuenta los posibles impactos o golpes que el robot pueda experimentar durante su navegación.

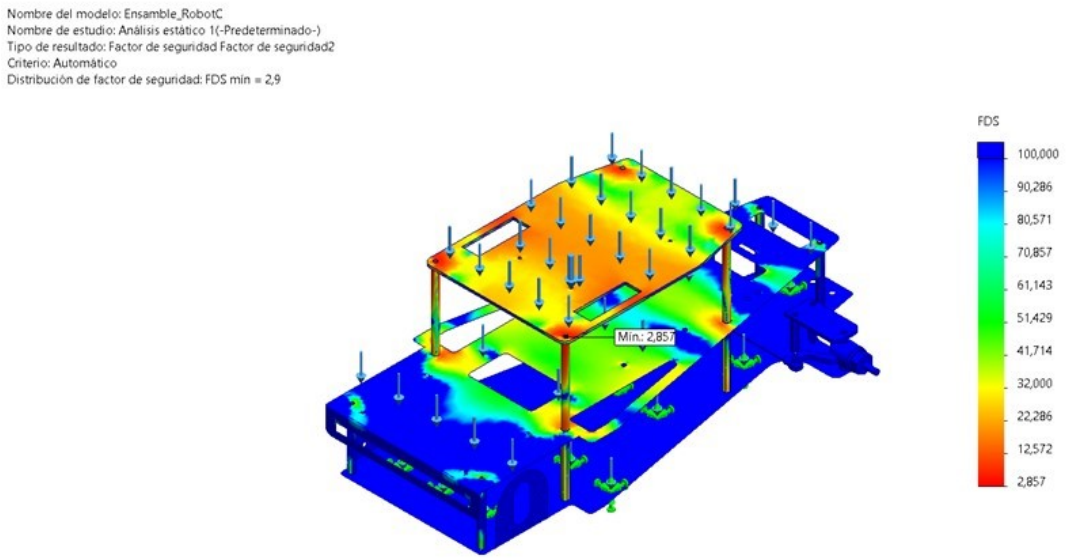


Figura 11. Factor de seguridad.
Fuente: Autores.

VII-A1. Sistema Ackermann: El sistema Ackermann es una configuración de la dirección implementada en vehículos, particularmente en automóviles y algunos robots móviles. Este sistema direccional fue diseñado con el propósito de asegurar que las ruedas delanteras de un vehículo sigan trayectorias precisas durante las maniobras, mejorando así la maniobrabilidad y disminuyendo el desgaste de los neumáticos.

En la Figura 12, se puede observar el sistema Ackermann, la ubicación de las ruedas delanteras se sitúan de manera que los ejes de dirección convergen en un punto imaginario, ubicado en el eje de simetría del vehículo, generalmente detrás del eje trasero. Esta configuración posibilita que las ruedas delanteras sigan trayectorias circulares durante las curvas, aspecto crucial para la ejecución de giros eficientes y suaves.

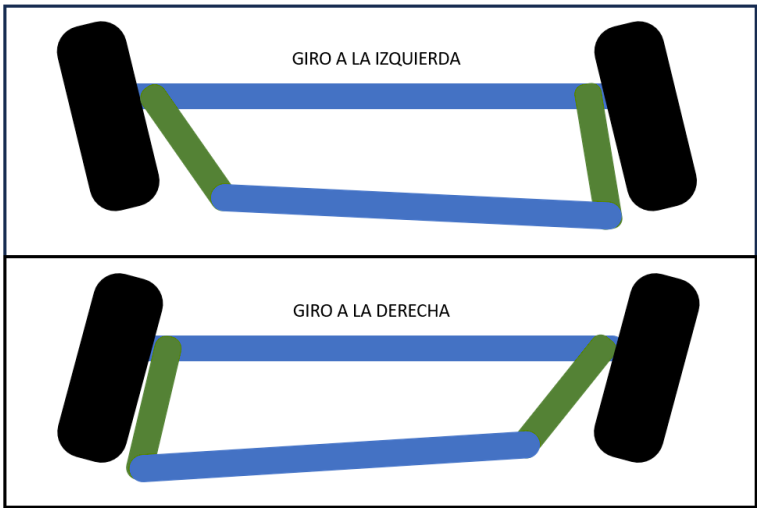


Figura 12. Funcionamiento del mecanismo Ackermann.
Fuente: Autores.

El Robot autónomo móvil implementa un mecanismo de estructura Ackermann que se encuentra conectado a un servomotor, y a las dos ruedas delanteras. Este mecanismo posibilita el control de las direcciones, permitiendo los movimientos de izquierda hacia la derecha y viceversa. Las dos ruedas traseras, desempeñando la función de ruedas motrices, son las encargadas de propulsar el movimiento tanto hacia adelante como hacia atrás del robot móvil. Los principales parámetros asociados con las estructuras Ackermann comprenden la distancia entre los ejes izquierdo y derecho, la separación entre las ruedas delanteras y traseras, así como el radio de giro.

VII-A2. Análisis de mecanismo: En esta sección se realizara el análisis de mecanismos para obtener una compresión detallada del comportamiento y funcionamiento del mecanismo del robot móvil, en este caso se vera la posición y velocidad que requiere el robot para girar a la derecha o a la izquierda según sea necesario. Además, este análisis permite diseñar componentes con la capacidad de interactuar entre si, permitiendo un correcto funcionamiento. Este enfoque contribuye a la mejora de la eficiencia de los sistemas mecánicos que componen el robot.

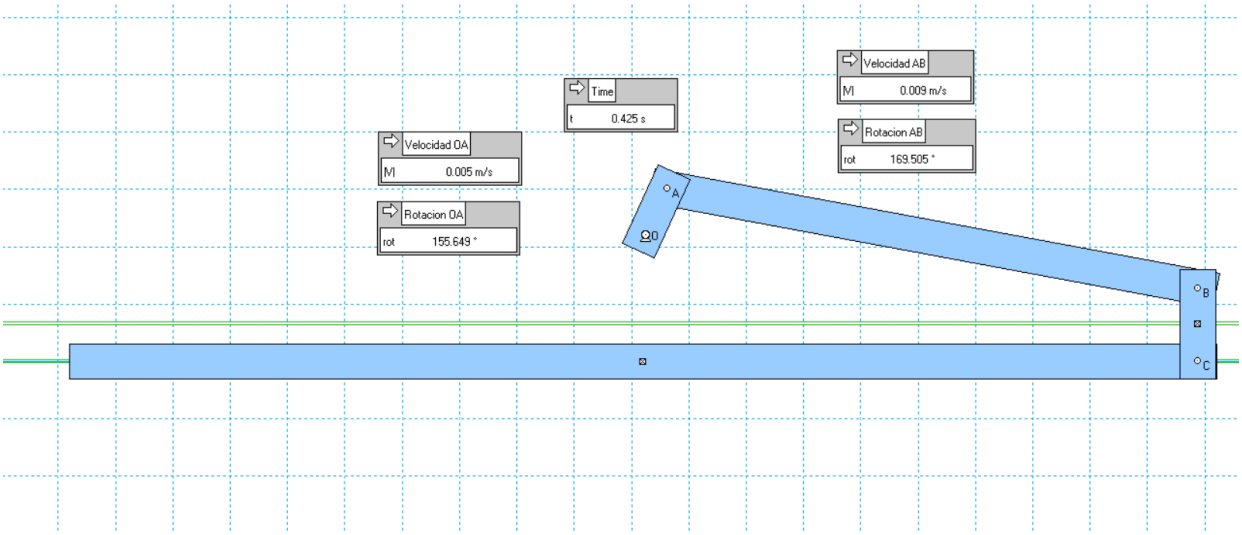


Figura 13. Velocidad y posición del mecanismo.
Fuente: Autores.

Como se observa en la Figura 13, se asignó una velocidad de 0.005 $[m/s]$ al mecanismo Ackermann con el propósito de medir el tiempo requerido para realizar un giro, como se puede ver el mecanismo precisa de un tiempo de 0.425 $[s]$ para girar los eslabones a la derecha, permitiendo que el robot móvil pueda inclinar las ruedas a la izquierda y avanzar en esa dirección.

VII-A3. Radio de giro: El radio de giro era una medida que describe la capacidad de un vehículo específico para realizar giros. Para ello, se procedió a calcular el radio de giro empleando la ecuación 7.

$$D = \frac{L}{sen(\theta)} \tag{7}$$

Donde L es la distancia entre los ejes del vehículo y θ el ángulo de giro de las ruedas delanteras, el robot autónomo tiene una distancia entre ejes de 23,5 $[cm]$, y el ángulo de 25 grados, se procede a reemplazar en la fórmula 8, obteniendo como resultado 55,61 $[cm]$ de radio.

$$D = \frac{23,5}{sen(25)} = 55,61[cm] \tag{8}$$

Cabe mencionar que la fórmula describe situaciones en las cuales el robot ejecuta giros cerrados. Además, tanto la fórmula como los datos asociados varían según la situación y el vehículo en consideración.

VII-B. Sistema Electrónico

En esta sección se analizará los componentes que formaran parte del sistema electrónico y porque son la mejor opción para realizar este proyecto. Se debe seleccionar los componentes que permitan desarrollar el sistema electrónico para la fabricación del robot autónomo móvil, el cual debe ser compatible con los elementos que lo conformaran y permita el robot mapear su entorno de manera eficiente. Una vez definido las necesidades del robot se procede a seleccionar los componentes tomando en cuenta los requerimientos del mismo, para poder cumplir con lo deseado. En la Figura 14 se observa los componentes que conformaran el sistema electrónico.

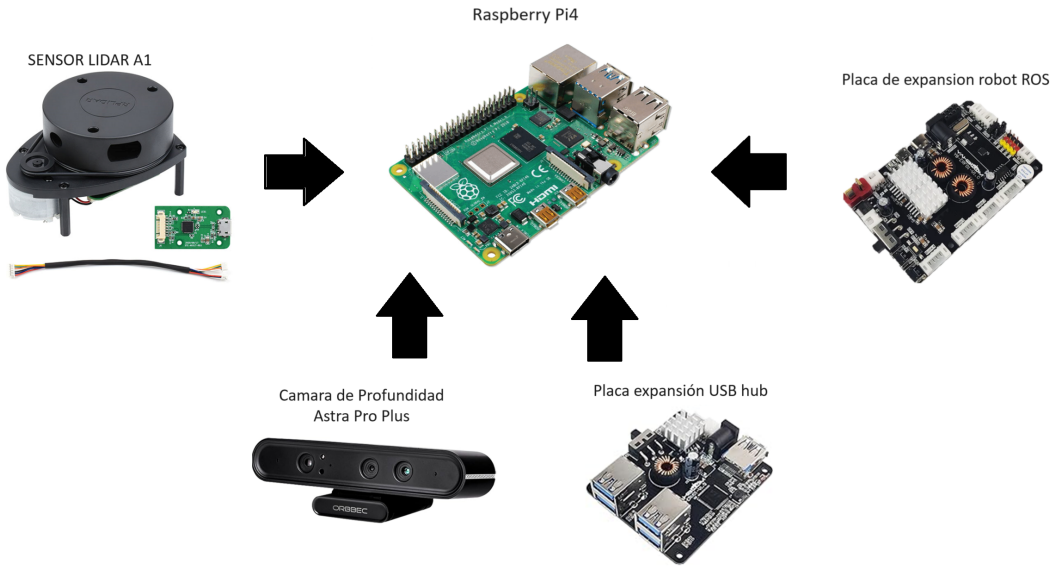


Figura 14. Componentes que conforman el sistema electrónicos para el Robot Autónomo de Mapeo.
Fuente: Autores.

VII-B1. Controladores: Como selección principal del proyecto se tiene el controlador debido a que tiende a ser el cerebro del robot por lo cual debe ser capaz de procesar el programa de manera rápida y eficiente, además de tener las entradas y salidas necesarias para cada componente del robot móvil, por esta razón se selecciono el Raspberry Pi debido a sus potentes funciones, a que es muy utilizado para la elaboración de este tipo de robot con sistema de mapeo, además de que se lo puede utilizar para la medición de desplazamiento en tiempo real basado en la visión. Cuenta con un software de procesamiento de imágenes utilizando el lenguaje C++, el cual es compatible con ROS [43]. Este mismo controlador deberá estar conectado a una tarjeta de expansión ya que se necesitará más pines en los cuales conectar los otros componentes, en este caso se eligió la placa de expansión ROS ya que es compatible con el Sistema de operación robótica (ROS) y permite controlar los dispositivos de hardware simplemente enviando comandos al puerto serie.

En la Tabla II, que se presenta a continuación, se evidencia que el Arduino Uno no resulta viable para el control de este proyecto debido a limitaciones relacionadas con el tamaño de su memoria. Este dispositivo carece de la capacidad de almacenamiento y procesamiento necesaria para cargar y ejecutar los programas requeridos. En cambio, el Raspberry Pi se adecua a esta función de manera más efectiva, dado que se trata de un minicomputador. Es importante destacar que, al comparar la Pi3 y la Pi4, se observa que la Pi3 tiene una capacidad de 1GB RAM, mientras que la Pi4 ofrece ventajas significativas, como una mayor capacidad de 2GB a 8GB lo que facilita la ejecución de los programas necesarios y la instalación de paquetes requeridos, como el sistema operativo Linux, la aplicación Ubuntu y el sistema ROS, para llevar a cabo el mapeo del robot autónomo.

Tabla II
COMPARACIÓN DE PLACAS DE CONTROL

Características	**Raspberry Pi4**	**Raspberry Pi3**	**Arduino uno**
Costo	80	55	20
Microprocesador	Cortex A-72 Quad Core	Cortex A-53 Quad Core	ATmega328
Pines digitales	40	40	14
Pines analógicos	0	0	6
USB	2*USB 2.0 2*USB 3.0	4*USB 2.0	No contiene
Bluetooth	4.2	5.0	No contiene
Memoria	2GB a 8GB ram	512MB a 1GB ram	2KB SRAM 1KB EEPROM
Velocidad	1,5 GHz, 64 Bits	1,4 GHz, 64 Bits	16 MHz
Espacio	Ranura Micro SD de 32 a 64 GB	Ranura Micro SD de 32 a 64 GB	32KB

Con la información previamente vista en la Tabla II, se procederá a hacer una selección del controlador para ver cual es el más apto para este proyecto, como se observa en la Tabla III se realiza la evaluación de los controladores seleccionados para calificar las características que lo componen y ver cual es el adecuado para el proyecto.

Tabla III
SELECCIÓN DE PLACA DE CONTROL

	Raspberry Pi4	**Raspberry Pi3**	**Arduino**
Criterios de evaluación	**Calificación**	**Calificación**	**Calificación**
Costo	3	3	5
Numero de I/O	5	5	2
Tamaño	4	4	5
Programación	4	4	5
Conexión Wifi	5	5	0
Conexiones USB	5	4	0
Frecuencia	5	4	1
Memoria	5	5	2
Total	36	34	20
Aprobación	Si	No	No

VII-B2. Sensores: El sensor lidar es una tecnología que permite medir la distancia que hay desde el láser que emite el sensor hacia un objeto que se encuentre en su entorno. Esta distancia se calcula midiendo en un intervalo de tiempo entre la emisión del pulso del láser y la recepción del mismo.
El sensor Lidar cuenta con una precisión alta y constante a la hora de realizar mediciones, justo lo que se necesita para la elaboración de este prototipo debido a que es excelente trabajando con sistemas de visión, y según su longitud mucho más eficientes para medir a través del polvo, esto indica también que si el mango de escaneo es más largo entonces la intensidad de la señal es más fuerte. Por lo general, Lidar es usado para la creación estimada de mapas en 2D o 3D, el funcionamiento de esto es bastante similar la diferencia es el plano de medición que se muestra. En el Plano 2D para poder obtener las coordenadas del obstáculo escaneado se emplean la distancia desde el sensor hasta el objeto escaneado y el ángulo del haz, es decir, el ángulo en el que se distribuye la luz desde la fuente, como se observa en la Figura 15.

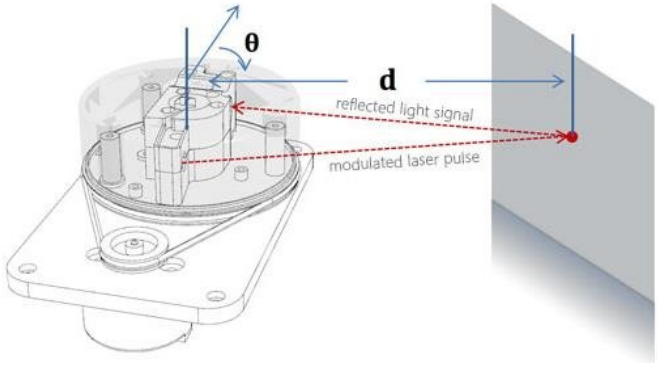


Figura 15. Funcionamiento Sensor Lidar.
Fuente: [44]

Como se mencionó antes el sensor Lidar calcula la distancia del objeto desde que emite el láser hasta que lo recibe esta es representada por la ecuación 9. Donde c es la velocidad de la luz en el medio por el cual viaja el pulso láser (usualmente se toma como $3 \cdot 10^8$ metros por segundo en el aire), y t es el tiempo que tarda el pulso láser en ir y volver desde el objeto.

$$D = \frac{c \cdot t}{2} \tag{9}$$

VII-B3. Fuente de alimentación: La fuente de alimentación es fundamental en el robot debido a que proporciona energía necesaria para su funcionamiento, su función principal es suministrar el voltaje a los diferentes componentes y sistemas que conforman al robot, permitiendo que realice sus tareas de manera efectiva. Por esta razón, este robot estará alimentado por una batería de litio de 12 $[V]$ a 6000 $[mAh]$ debido a que tiene las capacidades necesarias para alimentar al robot, además de que las baterías de litio son las más comunes de utilizar para la elaboración de este tipo de proyectos, el único problema que cabe destacar es la densidad de energía limitada, debido a que no pueden satisfacer una alta demanda de densidad energética, pero es útil para la fabricación de este prototipo [45].

VII-B4. Motor DC: Se ha elegido usar motores de corriente continua con enconders, debido a que con esto se puede controlar la velocidad del motor, además es compatible con los elementos del robot, lo cual es un factor importante en la navegación del robot autónomo. El motor posee las siguientes características 4.8 $[W]$ de potencia nominal, 2.8 $[A]$ de corriente, 550 rpm, y funciona con un voltaje de 3,3 $[V]$ a 5 $[V]$, una vez teniendo estos datos se puede obtener el torque del motor.

$$T = \frac{HP \cdot 5252}{RPM} \tag{10}$$

Con la Ecuación 10 se puede obtener el torque del motor, para esto primero se obtiene los caballos de fuerza (HP), haciendo una conversiones de los 4.8 $[W]$ a $[HP]$, donde 1 $[HP]$ equivale a 745.7 $[W]$.

$$4{,}8[W] \cdot \frac{1[HP]}{745{,}7[W]} = 0{,}00643691[HP] \tag{11}$$

Una vez obtenido el valor de los caballos de fuerza se procede a calcular el torque, dando como resultado 0.0615 $[lb \cdot ft]$.

$$T = \frac{0{,}00643691 \cdot 5252}{550rpm} = 0{,}0615[lb \cdot ft] \tag{12}$$

VII-C. Software

En esta sección se considera los diferentes algoritmos empleados para el mapeo del entorno y las pruebas desarrolladas para la navegación autónoma. La función y aplicación de un software es de gran importancia por su utilidad en las diversas áreas, debido a que permite procesar datos y automatizar diferentes procesos, esto implica la ejecución de instrucciones y algoritmos que permitan al robot desempeñar de mejor manera sus actividades.

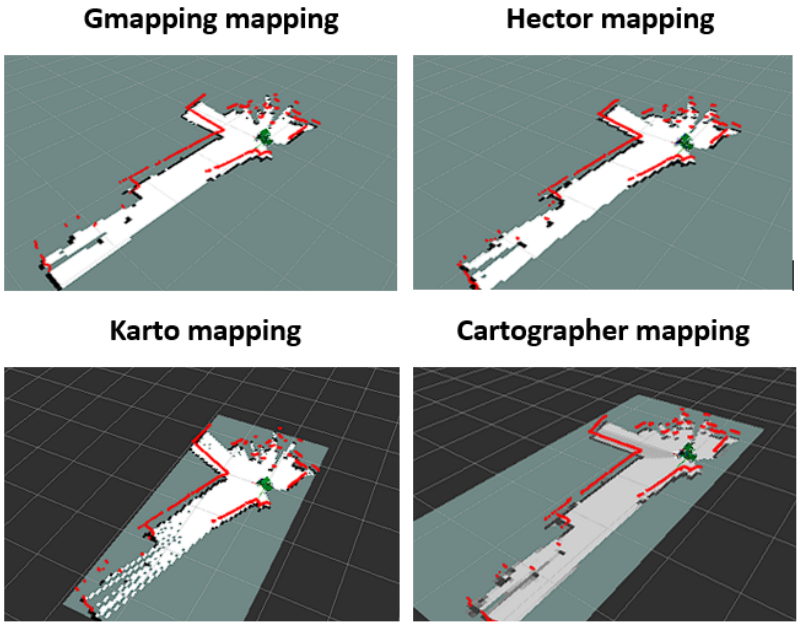


Figura 16. Algoritmos SLAM probados.
Fuente: Autores.

Como se mencionó en la sección V-B, se seleccionó el método SLAM debido a su experiencia en el área del mapeo y a sus diferentes algoritmos que se pueden emplear, en este caso se probó cuatro algoritmos: Gmapping mapping, Hector Mapping, Karto Mapping y Carthographer Mapping.

Como se observa en la Figura 16, estos cuatro algoritmos tienen una reconstrucción del mapa visualmente similar, pero cada uno tiene ciertas particularidades que los diferencian. Para empezar el algoritmo gmapping mapping utiliza un método basado en filtros de partículas y solo construye mapas basados en información láser [28], por lo cual es utilizado en entornos simples, el resultado que presenta este algoritmo son bastantes precisos al crear mapas, ya que se basa en gráficos y cuadrículas de ocupación de filtración de partículas Rao-Blackwellized (RBPF, por sus siglas en inglés), se usa una cantidad pequeña de partículas RBPF en su algoritmo, permitiendo la reducción de recursos computacionales para el muestreo y la generación de mapas, uno de sus principales problemas es susceptible a ruido causado por las perturbaciones. Además, de no detectar obstáculos como sillas o mesas [30].

El algoritmo Hector SLAM usa un método de mapeo en base a celdas y no requiere datos de odómetro, ya que usa el sensor Lidar para estimar su posición, este algoritmo es usado para lugares cerrados y pequeños, ya que no es necesario cerrar bucles demasiado grandes [46], al momento de construir el mapa el problema que presenta es que requiere tener una velocidad de rotación baja para no distorsionar la lectura. Mientras tanto, el algoritmo Karto Mapping usa resultados de su posición y trayectoria, este algoritmo usa los nodos de sus puntos de ubicación guardados para construir el mapa. La base de su trabajo es descomponer las matrices de Cholesky para minimizar el error, arrojando resultados de una postura y trayectoria. Este método construye el mapa usando nodos que guardan puntos de ubicación de la trayectoria del robot y los datos captados por las mediciones del sensor [47]. En cambio, el algoritmo Cartographer Mapping usa datos de odómetro y de la Unidad Inercial de Medida (IMU, por sus siglas en ingles) para estimar la trayectoria, en este caso el problema que presenta es las imágenes distorsionadas o con efectos borrosos, esquinas falsas que no son parte del modelo original. Finalmente, se implementaran cuatro algoritmos de mapeo con el objetivo de analizar su funcionamiento, características visuales. Esto tiene como fin identificar posibles limitaciones que se puedan mejorar para aplicarlos al robot autónomo de mapeo. Además, se realizó pruebas con el algoritmo seleccionado para poder reducir el margen de error asociado al proceso de mapeo. Así mismo, se implementaron dos algoritmos de navegación, el primero que usa el láser del lidar para esquivar los obstáculos y el segundo que es uno de los algoritmos que proporciona ROS llamado rrt_exploration el cual permite seleccionar un área que se desea mapear y proporciona rutas que pueden ejecutarse en esa área seleccionada. En la siguiente sección se describe la estructura necesaria para implementar lo algoritmos dentro de ROS.

VII-C1. Creación de la mesa de trabajo: Como se mencionó anteriormente, la implementación de los algoritmos permitió la visualización y recreación del mapa 2D de los diversos escenarios en los que se someterá a prueba el robot. Sin embargo, antes de poder implementar estos algoritmos, fue necesario crear el espacio de trabajo, también conocido como ”workspace”. En este espacio se colocaron los diferentes paquetes que contenían cada algoritmo de mapeo y navegación requerido para el desarrollo del proyecto. La estructura del espacio de trabajo se puede observar en la Figura 17.

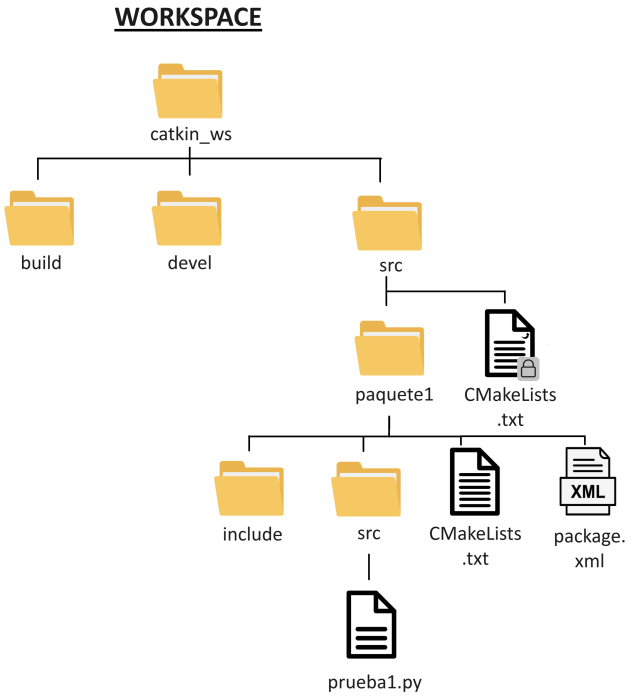


Figura 17. Estructura del Workspace.
Fuente: Autores.

Para poder crear el espacio de trabajo mencionado anteriormente se debe seguir los siguientes pasos:

1) Para crear la carpeta del workspace, primero se pone en la terminal, ”mkdir -p ˜/catkin_ws/src”, donde catkin_ws es el nombre que se le quiere asignar a la carpeta y src es la carpeta de recursos que estará dentro de la carpeta catkin_ws.

2) Se procede a colocar en la terminal ”cd catkin_ws” para entrar en la carpeta.

3) Luego colocamos ”catkin_make” para crear las carpetas necesarias para trabajar en entornos de trabajo catkin.

4) Después de colocar el comando se crean las carpetas build, devel y src, se entra a la carpeta ”src” con el comando ”cd src” y se observa que tendrá un archivo llamado ”CmakeLists.txt”.

5) Dentro de la carpeta ”src” se procede a crear nuestro primer paquete en donde se coloca la programación del robot, en la terminal colocamos lo que se muestra en la Figura 18, donde ”paquete1” es el nombre que se le quiere asignar al paquete y ”rospy, roscpp y std_msgs” son las dependencias que se usaran en el paquete.

```
$ catkin_create_pkg paquete1 rospy roscpp
```

Figura 18. Comando para crear el paquete de ROS
Fuente: Autores.

6) Dentro del paquete que se creo, se observara una carpeta ”include”, ”src” y otro archivo llamado ”Cmake-Lists.txt” en este caso este archivo si es editable, este ayudara más adelante para la creación de nodos.

7) Estando dentro del paquete se puede usar la carpeta ”src” que se creo, para los códigos ”.py” o ”.cpp” según la programación que se emplee, para programar el movimiento del robot, o si se desea se puede crear otra carpeta dentro del paquete para guardar los códigos.

Una vez creado el espacio de trabajo, hay que ir al archivo ”.bashrc” para colocar el nombre de la carpeta de tu espacio de trabajo y así no tener problemas al ejecutar los comando para activar los códigos de los paquetes.

VII-C2. Creación de un nodo: Para crear un nodo en ROS, se debe tener en cuenta que es un nodo, estos son puntos de conexión, los cuales permiten crear, enviar y recibir información. En este caso, los nodos se encuentran en cada paquete que contienen las líneas de instrucciones utilizadas para una tarea específica, además de servir para almacenar y organizar información de manera jerárquica. Es fundamental destacar que cada programa debe tener su propio nodo. La falta de un nodo impediría su reconocimiento por parte de ROS, lo que resultaría en la imposibilidad de utilizar el código desarrollado. En la Figura 19, se puede observar como cada nodo depende del anterior para poder activarse, cabe recalcar que no siempre es así, ya que hay nodos que pueden ser usados de manera independiente siempre y cuando el sistema de ros este activado. En la Figura 19, se observa que la activación inicial de los nodos 3 y 4 requería la activación previa del nodo 1. Del mismo modo, para activar los nodos 4 o 5, era necesario que el nodo 2 estuviera activado primero, ya que su activación permitía la activación de los otros nodos.

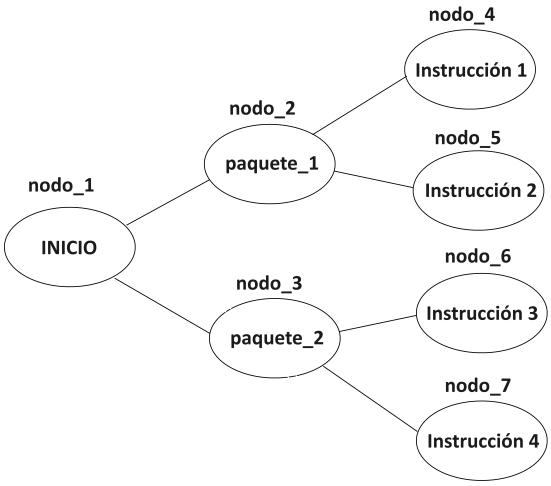


Figura 19. Ejemplo de conexión de los nodos
Fuente: Autores.

Continuando con el tema, para crear el nodo se debe saber el lugar donde se encuentra el código o instrucción al cual se le desea generar un nodo. Luego se entrara en la carpeta ”src” del paquete el cual contenga el código creado por el usuario, se le puede colocar cualquier nombre siempre y cuando al final se coloque el tipo de lenguaje de programación, este podría ser ”prueba1.py”, donde el ”.py” indica que el código esta escrito en python o ”.cpp” lo que incida que el código esta escrito en C++.

Luego salimos de la carpeta ”src” con el comando ”cd ..”, estando ahí entramos al archivo ”CmakeLists.txt”, donde se colocara el nodo del código utilizado. Dentro del ”CmakeLists.txt” al final del archivo se coloca el comando de la Figura 20, con el nombre del código que se creo con anterioridad, para que ROS reconozca el código que se creo y se pueda activar en la terminal.

```
catkin_install_python(PROGRAMS scripts/prueba1.py
  DESTINATION ${CATKIN_PACKAGE_BIN_DESTINATION})
```

Figura 20. Comando para activar nodo en el CmakeLists.txt
Fuente: Autores.

VII-C3. Creación de un launcher: Antes de iniciar con la creación de un launcher, es importante comprender su naturaleza. Un launcher es un archivo, que actúa como herramienta para llamar un nodo en especifico o un conjunto de nodos simultáneamente. Además, el launcher tenía la capacidad de ejecutar un conjunto de instrucciones, buscando y activando los códigos que estuvieran dentro de un paquete de ROS al que se había referido. También se utilizaba para buscar y ejecutar carpetas, rutas de carpetas, otros launchers o paquetes de códigos con sus instrucciones incluidas.

Para crear un archivo launcher inicialmente se debe estar dentro del paquete creado con anterioridad. Posteriormente, se procedió a crear una carpeta llamada ”launch”, donde se colocarían los archivos launcher para activar los códigos con un solo comando. Dentro de la carpeta ”launch”, se colocó un archivo con extensión ”.launch”, denominado ”’prueba1.launch”, este es un ejemplo ya que puede tener cualquier nombre siempre y cuando termine con la extensión ”.launch”

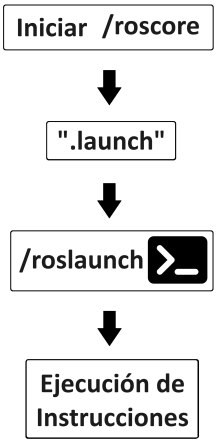


Figura 21. Estructura para iniciar el archivo launch
Fuente: Autores.

En la Figura 21, se presenta una guía sobre cómo ejecutar el archivo launch previamente creado. Para ello, se inició el sistema ROS en la terminal mediante el comando ”roscore”. Luego, se localizó el archivo launch que se iba a ejecutar, así como el paquete en el que se encontraba. Finalmente, en otra ventana de la terminal, se ejecutó el comando ”roslaunch” con la estructura siguiente: ”roslaunch paquete1 prueba.launch”, donde ”paquete1” representa el paquete creado y ”prueba.launch” el archivo launch previamente creado. Este procedimiento permitió la ejecución de las instrucciones contenidas en el archivo launch.

VII-D. Resultados

En esta sección se mostraran las pruebas de validación obtenidas al usar los diferentes sistemas de mapeo y navegación para el robot. Así mismo, se presentaran la ventajas y limitaciones de los algoritmos presentadas durante todas sus pruebas. Finalmente, se realiza un análisis comparativo evaluando el desempeño de los mismos. Se realizó un total de 50 pruebas en tres escenarios diferentes, a continuación se mostrara los resultados obtenidos por los algoritmos de hector mapping, gmapping mapping, karto mapping y cartographer mapping. En cada uno de los escenarios se han implementado mejoras para mitigar los inconvenientes presentados.

VII-E. Pruebas escenario 1

Como se observó en la Tabla IV, las pruebas realizadas en el primer escenario inicialmente detectaron correctamente el entorno, mostrando en blanco los espacios vacíos y en negro los obstáculos. Sin embargo, presentaba la limitación de no retroceder, sino simplemente girar las ruedas, lo que resultaba en colisiones. Además, se notó que a una distancia de 55 $[cm]$, el robot era incapaz de esquivar un obstáculo, lo que lo llevaba a caer en un bucle infinito. La Figura 22, muestra el primer escenario donde se llevaron a cabo las pruebas, en el cual solo se incluyeron 2 obstáculos. También se puede apreciar uno de los principales fallos que tenía, que consistía en escanear zonas inexistentes.

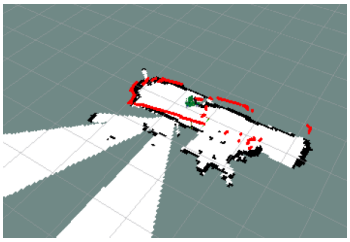


Figura 22. Prueba 1 de mapeo y navegación autónoma
Fuente: Autores.

Figura 23. Modelo de escenario 1
Fuente: Autores.

El primer escenario consistía en un área de aproximadamente 7 $[m^2]$ por lo que únicamente se implementaron los algoritmos de Hector mapping y gmapping mapping. Como se evidencia en la Tabla IV el Hector mapping tuvo un mejor desempeño en el mapeo de esta área con un tiempo promedio 1 minuto. Adicionalmente, en las diferentes iteraciones se implementaron mejoras para mejorar el desempeño. Sin embargo, pese a todos las mejoras implementadas, el robot normalmente colisionaba ligeramente con su entorno dando como resultado un mapeo con varias distorsiones como se muestra en la Figura 22. En esta fase del proyecto no se pudo determinar a ciencia cierta la efectividad de los algoritmos de mapeo debido a problemas en la navegación, para esto luego de implementar algunas consideraciones adicionales, se procedió a realizar pruebas en el escenario 2.

Tabla IV
PRUEBAS REALIZADAS EN EL ESCENARIO 1

N	Algoritmo	Tiempo	Navegación			Mapeo			Detección Obstáculos			Total
			Bueno	Regular	Malo	Bueno	Regular	Malo	Bueno	Regular	Malo	
1	Hector mapping	00:01:01	-	-	Si	-	-	Si	-	Si	-	1
2	Hector mapping	00:00:58	-	Si	-	-	Si	-	-	Si	-	3
3	Hector mapping	00:01:11	-	Si	-	-	Si	-	-	Si	-	3
4	Hector mapping	00:01:07	-	Si	-	-	Si	-	-	Si	-	3
5	Hector mapping	00:01:00	-	Si	-	-	Si	-	Si	-	-	4
6	Hector mapping	00:01:25	-	Si	-	Si	-	-	-	Si	-	4
7	Hector mapping	00:02:02	-	Si	-	Si	-	-	Si	-	-	5
8	Gmapping mapping	00:00:40	-	Si	-	Si	-	-	Si	-	-	5
9	Gmapping mapping	00:02:50	-	Si	-	Si	-	-	Si	-	-	5
10	Cartographer mapping	00:02:18	Si	-	-	-	Si	-	Si	-	-	5

VII-F. Pruebas escenario 2

El segundo escenario corresponde a una área de aproximadamente 14 $[m^2]$, en esta ocasión se implementaron adicionalmente los algoritmos de karto mapping y cartographer mapping. Como se muestra en la prueba 23 el algoritmo de gmapping logro mapear correctamente el área en un tiempo de 1 minuto 12 segundos. La Figura 24 representa el mejor resultado obtenido utilizando el algoritmo de karto mapping, como se puede apreciar aun existen ligeras distorsiones.

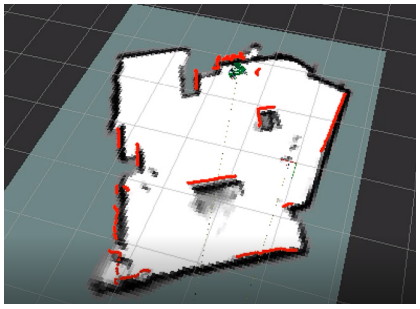


Figura 24. Prueba 20 de mapeo y navegación autónoma
Fuente: Autores.

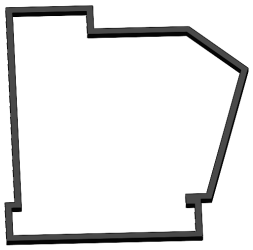


Figura 25. Modelo de escenario 2
Fuente: Autores.

En la Tabla V, se observa los resultados obtenidos en el segundo escenario donde los algoritmos que presentaron mejor resultado fueron karto mappinh y gmapping mapping.

Tabla V
PRUEBAS REALIZADAS EN EL ESCENARIO 2

N	Algoritmo	Tiempo	Navegación			Mapeo			Detección Obstáculos			Total
			Bueno	Regular	Malo	Bueno	Regular	Malo	Bueno	Regular	Malo	
11	Gmapping mapping	00:00:40	-	Si	-	-	Si	-	-	Si	-	3
12	Gmapping mapping	00:00:35	-	Si	-	-	-	Si	-	Si	-	2
13	Karto mapping	00:00:58	-	Si	-	-	Si	-	Si	-	-	4
14	Karto mapping	00:01:27	Si	-	-	Si	-	-	Si	-	-	6
15	Gmapping mapping	00:01:13	-	Si	-	-	Si	-	Si	-	-	4
16	Hector mapping	00:0:41	-	-	Si	Si	-	-	-	-	Si	2
17	Karto mapping	00:00:32	-	-	Si	-	Si	-	-	-	Si	1
18	Gmapping mapping	00:00:33	-	-	Si	-	Si	-	-	-	Si	1
19	Karto mapping	00:01:30	-	Si	-	Si	-	-	-	Si	-	4
20	Karto mapping	00:00:21	-	-	Si	-	Si	-	-	-	Si	1
21	Cartographer mapping	00:01:34	-	Si	-	-	-	Si	-	Si	-	2
22	Catographer mapping	00:02:30	-	Si	-	-	Si	-	-	Si	-	3
23	Gmapping mapping	00:01:12	-	Si	-	Si	-	-	Si	-	-	5
24	Gmapping mapping	00:01:03	Si	-	-	Si	-	-	Si	-	-	6
25	Gmapping mapping	00:01:00	-	Si	-	Si	-	-	-	Si	-	4

VII-G. Pruebas escenario 3

En los ensayos realizados previamente se detectó que la principal razón de distorsión esta relacionada con la perdida de conexión de los cables producto del movimiento. Una vez subsanado este detalle se probó el robot dando como resultado la Figura 26. Como se puede ver en la Figura 26, el mapeo coincide el plano mostrado en la Figura 27. Este resultado fue generado en un tiempo de 1 minuto 22 segundos en donde el robot fue capaz de esquivar 7 obstáculos.

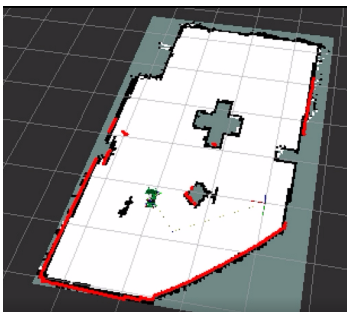


Figura 26. Prueba 50 de mapeo y navegación autónoma
Fuente: Autores.

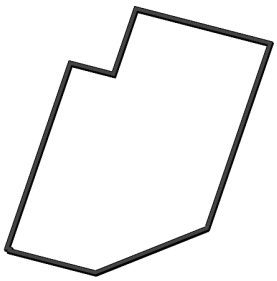


Figura 27. Modelo de escenario 3
Fuente: Autores.

El tercer escenario consistía de un área de aproximadamente 18 $[m^2]$, en la Figura 26 se tiene un referencia del escenario mencionado el cual fue sometido el robot para realizar las ultimas pruebas. Donde se puede observar que el algoritmo implementado tiene mejores resultados a la hora de mapear. Esto debido a los parámetros seleccionados para la navegación del robot y para la reconstrucción del mapa. En la Tabla VI, se muestra las pruebas realizadas en el tercer escenario al principio siguió detectando ciertos errores debido a los colores brillantes que habían en las paredes, pero poco a poco mediante ajustes y mejoras en la estructura se obtuvo un mejor desempeño.

Tabla VI
PRUEBAS REALIZADAS EN EL ESCENARIO 3

N	Algoritmo	Tiempo	Navegación			Mapeo			Detección Obstáculos			Total
			Bueno	Regular	Malo	Bueno	Regular	Malo	Bueno	Regular	Malo	
26	Cartographer mapping	00:00:37	-	Si	-	-	-	Si	Si	-	-	3
27	Cartographer mapping	00:01:14	Si	-	-	Si	-	-	Si	-	-	6
28	Cartographer mapping	00:01:25	Si	-	-	Si	-	-	Si	-	-	6
29	Cartographer mapping	00:01:17	-	-	Si	-	Si	-	-	Si	-	2
30	Gmapping mapping	00:02:28	-	Si	-	Si	-	-	Si	-	-	5
31	Gmapping mapping	00:00:33	-	-	Si	-	-	Si	-	-	Si	0
32	Hector mapping	00:00:48	-	Si	-	-	Si	-	-	-	Si	2
33	Hector mapping	00:01:23	Si	-	-	Si	-	-	Si	-	-	6
34	Karto mapping	00:00:55	Si	-	-	Si	-	-	Si	-	-	6
35	Karto mapping	00:01:04	Si	-	-	Si	-	-	Si	-	-	6
36	Karto mapping	00:00:32	-	-	Si	-	-	Si	-	Si	-	1
37	Karto mapping	00:01:12	Si	-	-	Si	-	-	Si	-	-	6
38	Karto mapping	00:01:15	Si	-	-	Si	-	-	Si	-	-	6
39	Cartographer mapping	00:03:02	-	Si	-	-	Si	-	-	Si	-	3
40	Cartographer mapping	00:04:19	-	Si	-	-	Si	-	-	Si	-	3
41	Cartographer mapping	00:01:58	-	-	Si	-	Si	-	-	Si	-	2
42	Hector mapping	00:00:43	Si	-	-	-	Si	-	-	Si	-	4
43	Hector mapping	00:00:59	-	-	Si	-	Si	-	-	Si	-	2
44	Karto mapping	00:00:35	-	-	Si	-	Si	-	-	Si	-	2
45	Karto mapping	00:00:58	-	-	Si	-	Si	-	-	-	Si	1
46	Karto mapping	00:01:12	Si	-	-	Si	-	-	Si	-	-	6
47	Karto mapping	00:01:03	Si	-	-	Si	-	-	Si	-	-	6
48	Karto mapping	00:01:09	Si	-	-	Si	-	-	Si	-	-	6
49	Karto mapping	00:01:01	Si	-	-	Si	-	-	Si	-	-	6
50	Karto mapping	00:01:22	Si	-	-	Si	-	-	Si	-	-	6

Finalmente, terminadas las 50 pruebas se evaluará cada algoritmo en función de los siguientes parámetros:

- Excelente: (5) - Representa el más alto nivel de rendimiento, calidad o satisfacción. Indica un desempeño sobresaliente en todos los aspectos evaluados.
- Muy Bueno: (4) - Indica un desempeño muy sólido y altamente satisfactorio. Aunque puede haber áreas de mejora, el resultado general es altamente positivo.
- Bueno: (3) - Significa que el desempeño es adecuado y cumple con las expectativas básicas. Puede haber algunas áreas de mejora, pero en general, el resultado es satisfactorio.
- Regular: (2) - Indica un desempeño promedio o aceptable, pero con margen significativo para mejorar. Sugiere que hay aspectos que no cumplen completamente con las expectativas.
- Deficiente: (1) - Representa un desempeño muy por debajo de las expectativas. Indica una falta de calidad, precisión o efectividad en la tarea evaluada. Requiere una revisión y mejoras significativas.

Como se evidenció en la Tabla VII, el algoritmo más viable empleado para este robot resultó ser el Karto mapping, gracias a los resultados obtenidos durante las pruebas. Este algoritmo demostró tener una mayor capacidad de detección del entorno para la navegación.

Tabla VII
DESEMPEÑO DE CADA ALGORITMO

	Tiempo	Resolución	Navegación	Detección	Precisión	Total
Karto mapping	4	4	4	4	4	20
Gmapping mapping	4	4	3	3	3	17
Hector mapping	3	4	3	3	3	16
Cartographer mapping	2	3	3	3	3	14

En la Figura 28, se aprecia el seguimiento temporal del mapeo durante intervalos de 10, 30 y 60 segundos, evidenciando los mejores resultados obtenidos en las pruebas de los cuatro algoritmos de mapeo implementados.

Es importante notar que se excluyeron los datos de 10 segundos debido a la ausencia de cambios significativos en el mapeo. Para el algoritmo de cartografía de Cartographer Mapping, se observó en la Figura 28-b que, a los 30 segundos el mapa detectó aparentemente dos entornos idénticos, pero posteriormente, en la Figura 28-c, a los 60 segundos se fusionaron los dos mapas generados, resultando en un único mapa con mayor detalle. Por otro lado, en el caso del algoritmo de mapeo de Gmapping, se constató en la Figura 28-e, que a los 30 segundos el mapeo del entorno estaba casi completo, mientras que en la Figura 28-f, a los 60 segundos se observó el mapa completamente finalizado. Respecto al algoritmo de mapeo de Karto Mapping, se registraron resultados satisfactorios donde, en la Figura 28-h, el mapa a los 30 segundos estaba notablemente cerca de completarse, y en la Figura 28-i, a los 60 segundos se mostró un mayor nivel de detalle del entorno. Finalmente, para el algoritmo de mapeo de Hector Mapping, se evidenció en la Figura 28-k, un buen progreso en el mapeo a los 30 segundos y en la Figura 28-l, a los 60 segundos se completó el mapa, aunque con la observación de ciertos detalles que no correspondían al entorno real, lo que resultó en la decisión de descartar este método de mapeo debido a la tendencia a mapear áreas inexistentes de gran tamaño debido a pequeñas perturbaciones.

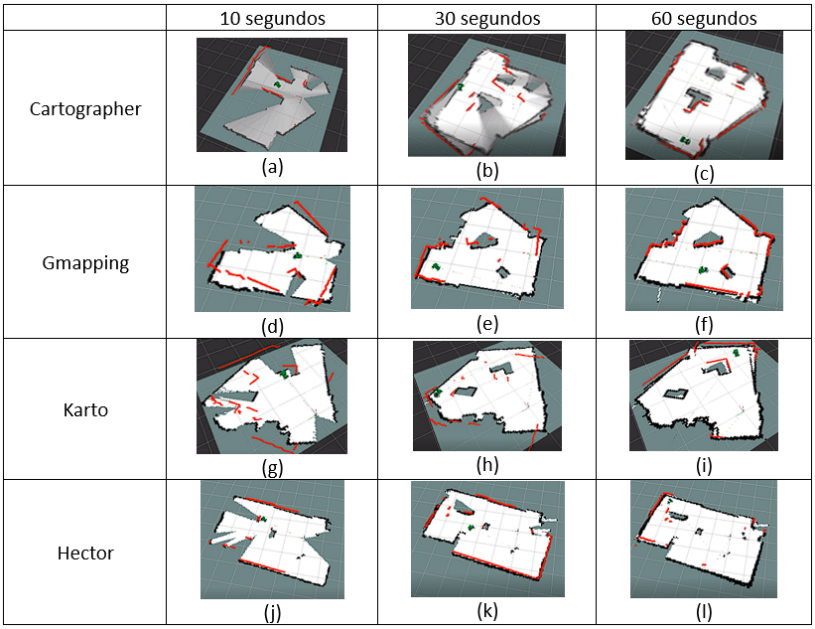
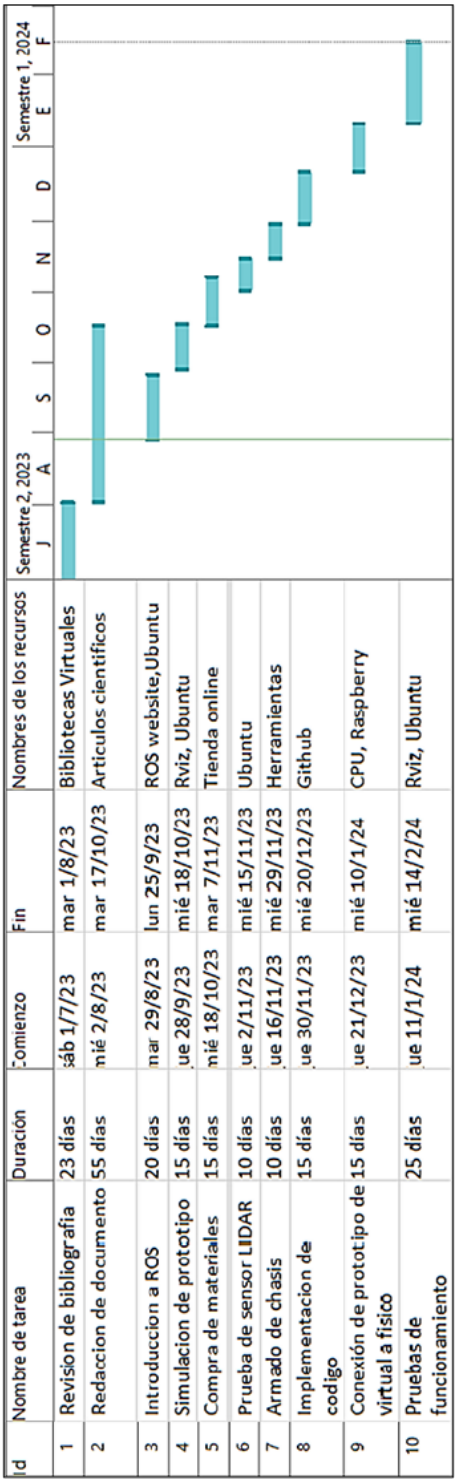


Figura 28. Comparación de tiempo de los cuatro algoritmos implementados.
Fuente: Autores.

VIII. Cronograma y actividades a desarrollar

Tabla VIII
Cronograma de actividades para implementación del anteproyecto de tesis



Id	Nombre de tarea	Duración	Comienzo	Fin	Nombres de los recursos
1	Revision de bibliografia	23 días	sáb 1/7/23	mar 1/8/23	Bibliotecas Virtuales
2	Redaccion de documento	55 días	mié 2/8/23	mar 17/10/23	Articulos cientificos
3	Introduccion a ROS	20 días	mar 29/8/23	lun 25/9/23	ROS website,Ubuntu
4	Simulacion de prototipo	15 días	jue 28/9/23	mié 18/10/23	Rviz, Ubuntu
5	Compra de materiales	15 días	mié 18/10/23	mar 7/11/23	Tienda online
6	Prueba de sensor LIDAR	10 días	jue 2/11/23	mié 15/11/23	Ubuntu
7	Armado de chasis	10 días	jue 16/11/23	mié 29/11/23	Herramientas
8	Implementacion de codigo	15 días	jue 30/11/23	mié 20/12/23	Github
9	Conexión de prototipo de virtual a fisico	15 días	jue 21/12/23	mié 10/1/24	CPU, Raspberry
10	Pruebas de funcionamiento	25 días	jue 11/1/24	mié 14/2/24	Rviz, Ubuntu

IX. Presupuestos y gastos

En la siguiente tabla de presupuestos nos indica los valores aproximados que se pudo obtener mediante cotizaciones realizadas en diferentes páginas web para poder tener un presupuesto de lo que se podría gastar para elaborar el proyecto de tesis.

Tabla IX
Tabla de presupuesto

Ítem	Descripción/Articulo	Precio Unitario	Cantidad	Precio Total
1	**Raspberry pi 4 - Modelo B**	90	1	90
2	**Motor DC**	10	2	40
3	**Chasis**	130	1	130
4	**Ruedas**	1,5	4	6
5	**Placa de expansión robot ROS**	80	1	80
6	**Sensor LIDAR - SLAM A1**	110	1	110
7	**Micro SD 64GB**	9	1	9
8	**Batería de litio 12V**	50	1	50
9	**Placa de expansión USB hub**	39	1	39
10	**Astro Pro Plus Depth Camera**	192	1	192
	Total (USD)			746

X. Conclusiones

Los sistemas implementados en la Figura 7, permitieron esquivar un 92 % de los obstáculos presentes en el espacio a mapear. El controlador Raspberry pi 4 junto con ROS permitieron implementar 2 algoritmos de navegación y 4 algoritmos de mapeo.

La estructura es capaz de soportar 3 [kg] con un deformación máxima de 0.09 [mm] y un esfuerzo máximo de 19.3 [MPa]como se muestra en la Figuras 8 y 9. Por lo que se concluye que no presentara problemas con las cargas presentes en la navegación.

El mecanismo de Ackermann planteado para un ángulo de 25 grados permite un radio de giro de 111,21 [cm] lo cual permite esquivar un obstáculo a una distancia de 45 [cm] como se verificó en la sección VII-D con una velocidad lineal 0,2 [m/s] y una velocidad angular de 0,4 [$m/s2$].

El algoritmo de mapeo que mejor funcionó fue el Karto mapping conjuntamente con el algoritmo de navegación, como se muestra en la Tabla VII. Este algoritmo permitió mapear un área de 18 [m^2] en un tiempo promedio de 1 minuto 10 segundos, como se puede verificar en las últimas 5 pruebas de la Tabla VI.

Como se muestra en las Figuras 25 y 26 el error de mapeo según mediciones corresponde a 8 %, se concluye que esta dentro del rango proyectado dentro de la investigación.

XI. Recomendaciones

Es fundamental comprender el funcionamiento del robot y discernir las necesidades operativas para garantizar su desempeño adecuado. Además, el conocimiento de las limitaciones inherentes a cada algoritmo permite la implementación de medidas que fortalezcan la capacidad del robot para resistir perturbaciones que puedan incidir en la precisión del mapeo.

Durante la utilización de algoritmos de mapeo, es imperativo evitar el empleo de colores brillantes o el negro, dado que estos pueden interferir con la capacidad del sensor LIDAR para reflejar el láser, resultando en la pérdida de información sobre la distancia a los obstáculos, lo que afecta la efectividad del mapeo.

Un aspecto crucial reside en garantizar que cada programa en ROS (Robot Operating System) tenga su propio nodo. La omisión de uno de estos nodos puede resultar en la falta de reconocimiento por parte de ROS, lo que conlleva la incapacidad de utilizar el código desarrollado.

Además, se recomienda encarecidamente asegurar los cables utilizando silicona u otro método adecuado para evitar su movimiento, lo que podría provocar fallos en la conexión y, por ende, comprometer el desempeño del robot.

REFERENCIAS

[1] J. Delmerico, E. Mueggler, J. Nitsch y D. Scaramuzza, «Active autonomous aerial exploration for ground robot path planning,» *IEEE Robotics and Automation Letters*, vol. 2, n.º 2, págs. 664-671, 2017.

[2] C. Wang, J. Cheng, J. Wang, X. Li y M. Q.-H. Meng, «Efficient object search with belief road map using mobile robot,» *IEEE Robotics and Automation Letters*, vol. 3, n.º 4, págs. 3081-3088, 2018.

[3] C. Wang, D. Zhu, T. Li, M. Q.-H. Meng y C. W. De Silva, «Efficient autonomous robotic exploration with semantic road map in indoor environments,» *IEEE Robotics and Automation Letters*, vol. 4, n.º 3, págs. 2989-2996, 2019.

[4] S. Srinivas, S. Ramachandiran y S. Rajendran, «Autonomous robot-driven deliveries: A review of recent developments and future directions,» *Transportation research part E: logistics and transportation review*, vol. 165, pág. 102 834, 2022.

[5] S. Pookkuttath, B. F. Gomez, M. R. Elara y P. Thejus, «An optical flow-based method for condition-based maintenance and operational safety in autonomous cleaning robots,» *Expert Systems with Applications*, vol. 222, pág. 119 802, 2023.

[6] K. P. Romero Bringas, «Plataforma de mapeo multiespectral basada en hardware-software abierto para aplicaciones en agricultura de precisión.,» 2023.

[7] C. Bianco y F. Porta, «Los límites de la balanza de pagos tecnológicos para medir la transferencia de tecnología en los países en desarrollo,» *Red de Indicadores de Ciencia y Tecnología-Iberoamericana e Interamericana (ed.), El estado de la ciencia*, págs. 1-13, 2003.

[8] T. Zhang, T. Qiu, Z. Pu, Z. Liu y J. Yi, «Robot navigation among external autonomous agents through deep reinforcement learning using graph attention network,» *IFAC-PapersOnLine*, vol. 53, n.º 2, págs. 9465-9470, 2020.

[9] E. Menendez, J. G. Victores, R. Montero, S. Martínez y C. Balaguer, «Tunnel structural inspection and assessment using an autonomous robotic system,» *Automation in Construction*, vol. 87, págs. 117-126, 2018.

[10] V. Prabakaran, A. V. Le, P. T. Kyaw, P. Kandasamy, A. Paing y R. E. Mohan, «sTetro-D: A deep learning based autonomous descending-stair cleaning robot,» *Engineering Applications of Artificial Intelligence*, vol. 120, pág. 105 844, 2023.

[11] D. F. Quiroga Guerra y B. L. Rubio Amaya, «Robot móvil autónomo para la siembra de semillas en el campo,» B.S. thesis, 2023.

[12] C. Papachristos, S. Khattak, F. Mascarich y K. Alexis, «Autonomous navigation and mapping in underground mines using aerial robots,» en *2019 IEEE Aerospace Conference*, IEEE, 2019, págs. 1-8.

[13] T. Verma y A. Mishra, «Development of robot model for cleaning open space,» *Materials Today: Proceedings*, vol. 22, págs. 1803-1811, 2020.

[14] M. Tomy, B. Lacerda, N. Hawes y J. L. Wyatt, «Battery charge scheduling in long-life autonomous mobile robots via multi-objective decision making under uncertainty,» *Robotics and Autonomous Systems*, vol. 133, pág. 103 629, 2020.

[15] A. S. Aguiar, F. N. dos Santos, J. B. Cunha, H. Sobreira y A. J. Sousa, «Localization and mapping for robots in agriculture and forestry: A survey,» *Robotics*, vol. 9, n.º 4, pág. 97, 2020.

[16] I. Lluvia, E. Lazkano y A. Ansuategi, «Active mapping and robot exploration: A survey,» *Sensors*, vol. 21, n.º 7, pág. 2445, 2021.

[17] J. M. Aitken, M. H. Evans, R. Worley et al., «Simultaneous localization and mapping for inspection robots in water and sewer pipe networks: A review,» *IEEE Access*, vol. 9, págs. 140 173-140 198, 2021.

[18] H. Wu, H. Yue, Z. Xu, H. Yang, C. Liu y L. Chen, «Automatic structural mapping and semantic optimization from indoor point clouds,» *Automation in Construction*, vol. 124, pág. 103 460, 2021.

[19] N. Cavalagli, A. Kita, V. Castaldo, A. Pisello y F. Ubertini, «Hierarchical environmental risk mapping of material degradation in historic masonry buildings: An integrated approach considering climate change and structural damage,» *Construction and Building Materials*, vol. 215, págs. 998-1014, 2019.

[20] N. Yoshinaga, R. Kamasaka, Y. Shibata y K. Oguri, «Pipelined FPGA implementation of a wave-front-fetch graph cut system,» en *Complex, Intelligent, and Software Intensive Systems: Proceedings of the 13th*

International Conference on Complex, Intelligent, and Software Intensive Systems (CISIS-2019), Springer, 2020, págs. 430-441.

[21] M. Peris Martorell, «Aproximación del Mapa de Disparidad Estéreo Mediante Técnicas de Aprendizaje Automático,» 2012.

[22] J. Bac y A. Zinovyev, «Lizard brain: Tackling locally low-dimensional yet globally complex organization of multi-dimensional datasets,» *Frontiers in neurorobotics*, vol. 13, pág. 110, 2020.

[23] A. Rubio Pintado et al., «Aplicación interactiva para el análisis de datos mediante mapas autoorganizados,» B.S. thesis, 2021.

[24] S. Sarikyriakidis, K. Goulianas y A. I. Margaris, «Using Self-organizing Maps to Solve the Travelling Salesman Problem: A Review,» *WSEAS Transactions on Systems*, vol. 22, págs. 131-159, 2023.

[25] M. Vidal Morales et al., «Integración de Google Maps y TSP Solvers para la definición y resolución del TSP,» B.S. thesis, 2012.

[26] A. Macario Barros, M. Michel, Y. Moline, G. Corre y F. Carrel, «A comprehensive survey of visual slam algorithms,» *Robotics*, vol. 11, n.º 1, pág. 24, 2022.

[27] P. López Torres, «Análisis de Algoritmos para Localización y Mapeado simultáneo de Objetos,» 2016.

[28] J. Huang, S. Junginger, H. Liu y K. Thurow, «Indoor Positioning Systems of Mobile Robots: A Review,» *Robotics*, vol. 12, n.º 2, pág. 47, 2023.

[29] M. N. Favorskaya, S. Mekhilef, R. K. Pandey y N. Singh, «Innovations in Electrical and Electronic Engineering Proceedings of ICEEE 2020,» *Proceedings of ICEEE*, pág. 1, 2020.

[30] D. I. Soque León, M. G. Guerra Pintado, D. A. Plaza Guingla et al., «Implementación de técnica de mapeo y localización simultáneo (SLAM) en vehículo autónomo,» Tesis doct., ESPOL. FIEC, 2020.

[31] C. Parrott, T. J. Dodd, J. Boxall y K. Horoshenkov, «Simulation of the behavior of biologically-inspired swarm robots for the autonomous inspection of buried pipes,» *Tunnelling and Underground Space Technology*, vol. 101, pág. 103 356, 2020.

[32] V. Velmurugan, L. Sharmila, D. N. Ponkumar et al., «Performance observation of a concurrent compute-intensive vision system in a human-like autonomous intelligent robot,» *Measurement: Sensors*, vol. 27, pág. 100 805, 2023.

[33] F. d'Apolito, «Obstacle Detection and Avoidance of a cost-oriented humanoid robot,» *IFAC-PapersOnLine*, vol. 51, n.º 30, págs. 198-203, 2018.

[34] X. Yang, H. Li, J. Liu et al., «A novel stereo image self-inpainting network for autonomous robots,» *Robotics and Autonomous Systems*, vol. 156, pág. 104 197, 2022.

[35] D. Fernandez-Chaves, J.-R. Ruiz-Sarmiento, A. Jaenal, N. Petkov y J. Gonzalez-Jimenez, «Robot@ VirtualHome, an ecosystem of virtual environments and tools for realistic indoor robotic simulation,» *Expert Systems with Applications*, vol. 208, pág. 117 970, 2022.

[36] S. Park, S. Yoon, S. Ju y J. Heo, «BIM-based scan planning for scanning with a quadruped walking robot,» *Automation in Construction*, vol. 152, pág. 104 911, 2023.

[37] A. Deb, Z. Wypych, J. Lonner y H. Ashrafiuon, «Design and control of an autonomous robot for mobility-impaired patients,» *Journal of Medical Robotics Research*, vol. 6, n.º 03n04, pág. 2 150 007, 2021.

[38] M. Jayaratne, D. Alahakoon y D. De Silva, «Unsupervised skill transfer learning for autonomous robots using distributed growing self organizing maps,» *Robotics and Autonomous Systems*, vol. 144, pág. 103 835, 2021.

[39] M. V. J. Muthugala, A. Vengadesh, X. Wu et al., «Expressing attention requirement of a floor cleaning robot through interactive lights,» *Automation in Construction*, vol. 110, pág. 103 015, 2020.

[40] D. Fernandez-Chaves, J.-R. Ruiz-Sarmiento, N. Petkov y J. Gonzalez-Jimenez, «ViMantic, a distributed robotic architecture for semantic mapping in indoor environments,» *Knowledge-Based Systems*, vol. 232, pág. 107 440, 2021.

[41] D. T. Son, M. T. Anh, D. D. Tu, T. H. Cuong, H. S. Phuong et al., «The practice of mapping-based navigation system for indoor robot with RPLIDAR and Raspberry Pi,» en *2021 International Conference on System Science and Engineering (ICSSE)*, IEEE, 2021, págs. 279-282.

[42] L. Nwankwo, C. Fritze, K. Bartsch y E. Rueckert, «ROMR: A ROS-based open-source mobile robot,» *HardwareX*, vol. 14, e00426, 2023.
[43] M. Wang, K.-Y. Koo, C. Liu y F. Xu, «Development of a low-cost vision-based real-time displacement system using Raspberry Pi,» *Engineering Structures*, vol. 278, pág. 115 493, 2023.
[44] C. Zhi y S. Xiumin, «Research on Cartographer Algorithm based on Low Cost Lidar,» *International Journal of engineering Research and Technology (IJERT)*, vol. 8, n.º 10, 2019.
[45] S. Jing, J. Chen, M. Li, H. Liang, P. Kannan y P. Tsiakaras, «Rechargeable non-aqueous lithium-O2 batteries: Novel bimetallic (Ni-Cu) conductive coordination polymer cathodes,» *Journal of Energy Storage*, vol. 66, pág. 107 331, 2023.
[46] C. R. Alonso, «Proyecto Fin de Grado Grado en Ingeniería de las Tecnologías Industriales,»
[47] K. Trejos, L. Rincón, M. Bolaños, J. Fallas y L. Marín, «2D SLAM Algorithms Characterization, Calibration, and Comparison Considering Pose Error, Map Accuracy as Well as CPU and Memory Usage,» *Sensors*, vol. 22, n.º 18, pág. 6903, 2022.

XII. ANEXOS

Tabla X
TABLA EXTENDIDA DE RESULTADOS DE LAS PRUEBAS REALIZADAS EN EL ESCENARIO 1

N	Algoritmo	Duración	Errores	Comentario
1	Hector mapping	00:01:01	si	No retrocede el robot solo avanzaba y movió las ruedas frontales según su necesidad y terminaba estrellándose.
2	Hector mapping	00:00:58	si	No retrocede, se ajusto la distancia de respuesta a 60 cm, no se choco pero giro muy rápido y rosaba las paredes al girar.
3	Hector mapping	00:01:11	si	No retrocede, se ajusto la velocidad lineal, no detectaba zonas con colores brillantes.
4	Hector mapping	00:01:07	si	Se ajusto la velocidad para que el robot se mueva mas lento y asi halla menos fallas al mapear.
5	Hector mapping	00:01:00	si	Se ajusto la distancia al detectar los obstaculos a menos de 10 cm, funciono mejor esquivo mejor los obstaculos pequeños pero se chocaba en la parte trasera.
6	Hector mapping	00:01:25	si	Detecto mejor los obstáculos pero reacciona tarde, tiende a chocarse a la final del giro.
7	Hector mapping	00:02:02	si	Se le bajo la velocidad linear y aumento la distancia de respuesta, reacciono mejor pero se demoraba en girar completamente.
8	Gmapping mapping	00:00:40	si	Se le ajusto la distancia de respuesta a 60 cm, sin obstaculos, consiguio mapear bien, pero se quedaba trabado en los espacios pequeños.
9	Gmapping mapping	00:02:50	si	se le ajusto la distancia de respuesta a 50 cm, logro moverse mejor en los lugares pequeños pero le tomo tiempo
10	Cartographer mapping	00:02:18	si	Se le ajusto a 0.2 m/s de velocidad lineal y 0,3 rad/s de velocidad angular, logro esquivar mejor el obstaculo y obtener un mapa con errores minimos

Tabla XI
TABLA EXTENDIDA DE RESULTADOS DE LAS PRUEBAS REALIZADAS EN EL ESCENARIO 2

N	Algoritmo	Duración	Errores	Comentarios
11	Gmapping mapping	00:00:40	si	No logro mapear por completo el entorno.
12	Gmapping mapping	00:00:35	si	Choco ligeramente la parte trasera y creo en el mapa una zona que no existe.
13	Karto mapping	00:00:58	si	Detecto la caja pero no la esquivo a tiempo.
14	Karto mapping	00:01:27	no	funciono con normalidad recontruyo el mapa por completo
15	Gmapping mapping	00:01:13	si	Se implemento el Gmmaping junto a algoritmo de navegación rrt_exploration, no reconstruyo por completo el mapa
16	Hector mapping	00:0:41	no	Mapeo rápido la zona, pero se choco contra un obstáculo.
17	Karto mapping	00:00:32	si	No logro escanear por completo el mapa y no detecto la pared correctamente.
18	Gmapping mapping	00:00:33	si	No detecto correctamente la pared, la esquivo al principio pero luego se choco con la misma pared.
19	Karto mapping	00:01:30	si	No detecto a tiempo el primer obstáculo, luego esquivo correctamente todo y escaneo por completo el entorno.
20	Karto mapping	00:00:21	si	No detecto la pared se estrello contra ella, pero logro escanear gran parte del entorno.
21	Cartographer mapping	00:01:34	si	Se recreo dos mapas encima del original, luego se distorcino todo.
22	Catographer mapping	00:02:30	no	Se distorsiono un poco la recreación del mapa, luego de unos 20 segundos logro acoplarse en uno solo, dando un buen resultado.
23	Gmapping mapping	00:01:12	no	Logro mapear correctamente.
24	Gmapping mapping	00:01:03	no	Logro esquivar a tiempo los obstáculos y mapear el entorno
25	Gmapping mapping	00:01:00	si	Logro mapear correctamente pero empujo la caja ya que no detecto el colo negro

Tabla XII
TABLA EXTENDIDA DE RESULTADOS DE LAS PRUEBAS REALIZADAS EN EL ESCENARIO 3

N	Algoritmo	Duración	Errores	Comentarios
26	Cartographer mapping	00:00:37	si	No logro mapear correctamente, pero si esquivo bien los obstáculos.
27	Cartographer mapping	00:01:14	no	Logro mapear y esquivar los obstáculos correctamente.
28	Cartographer mapping	00:01:25	no	Logro mapear y esquivar los obstáculos correctamente.
29	Cartographer mapping	00:01:17	si	Se atasco en una de las esquinas, no logro salir de ahi.
30	Gmapping mapping	00:02:28	no	Logro mapear correctamente el entorno.
31	Gmapping mapping	00:00:33	si	No logro mapear correctamente, detecto zonas inexistentes.
32	Hector mapping	00:00:48	si	No detecto a tiempo los obstáculos.
33	Hector mapping	00:01:23	no	Logro mapear correctamente alrededor, con 3 obstáculos en su entorno.
34	Karto mapping	00:00:55	no	Logro mapear el entorno de forma rápida debido a la ubicación del robot.
35	Karto mapping	00:01:04	no	Logro esquivar a tiempo lo obstáculos, permitiendo un buen mapeo.
36	Karto mapping	00:00:32	si	Golpeo un obstáculo con la parte trasera del robot y lo que ocasiono un mal mapeo.
37	Karto mapping	00:01:12	no	Logro mapear correctamente
38	Karto mapping	00:01:15	no	Logro mapear y esquivar correctamente
39	Cartographer mapping	00:03:02	si	No logro mapear todo el entorno, se atasco en medio de dos obstáculos.
40	Cartographer mapping	00:04:19	no	Se le dificulto salir de en medio de un obstáculo.
41	Cartographer mapping	00:01:58	si	Se choco con un obstáculo al terminar de girar.
42	Hector mapping	00:00:43	si	Logro navegar correctamente pero mapeo lugares inexistentes
43	Hector mapping	00:00:59	no	Al colocar los obstáculos en diagonal al robot logro mapear y pero no esquivar correctamente
44	Karto mapping	00:00:35	si	Se quedo trabado en una esquina
45	Karto mapping	00:00:58	si	Al final se golpeo con un obstáculos
46	Karto mapping	00:01:12	no	Logro mapear correctamente alrededor, con 4 obstáculos en su entorno.
47	Karto mapping	00:01:03	no	Logro mapear correctamente alrededor, con 5 obstáculos en su entorno.
48	Karto mapping	00:01:09	no	Logro mapear correctamente alrededor, con 6 obstáculos en su entorno.
49	Karto mapping	00:01:01	no	Logro mapear correctamente con una zona cerrada en la esquina.
50	Karto mapping	00:01:22	no	Logro mapear correctamente alrededor, con 7 obstáculos en su entorno de forma aleatoria.

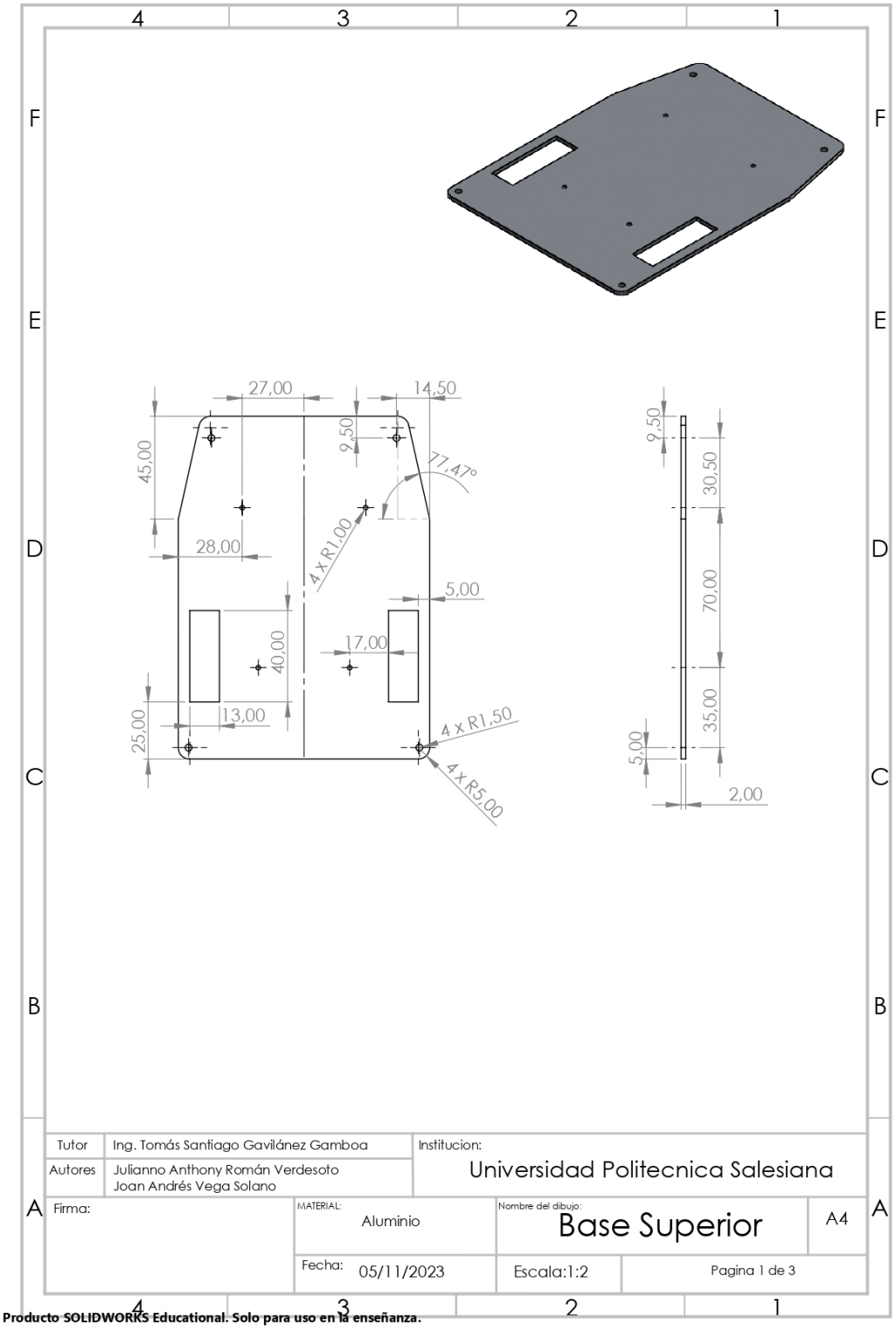


Figura 29. Plano base superior del robot de mapeo

Fuente: Autores.

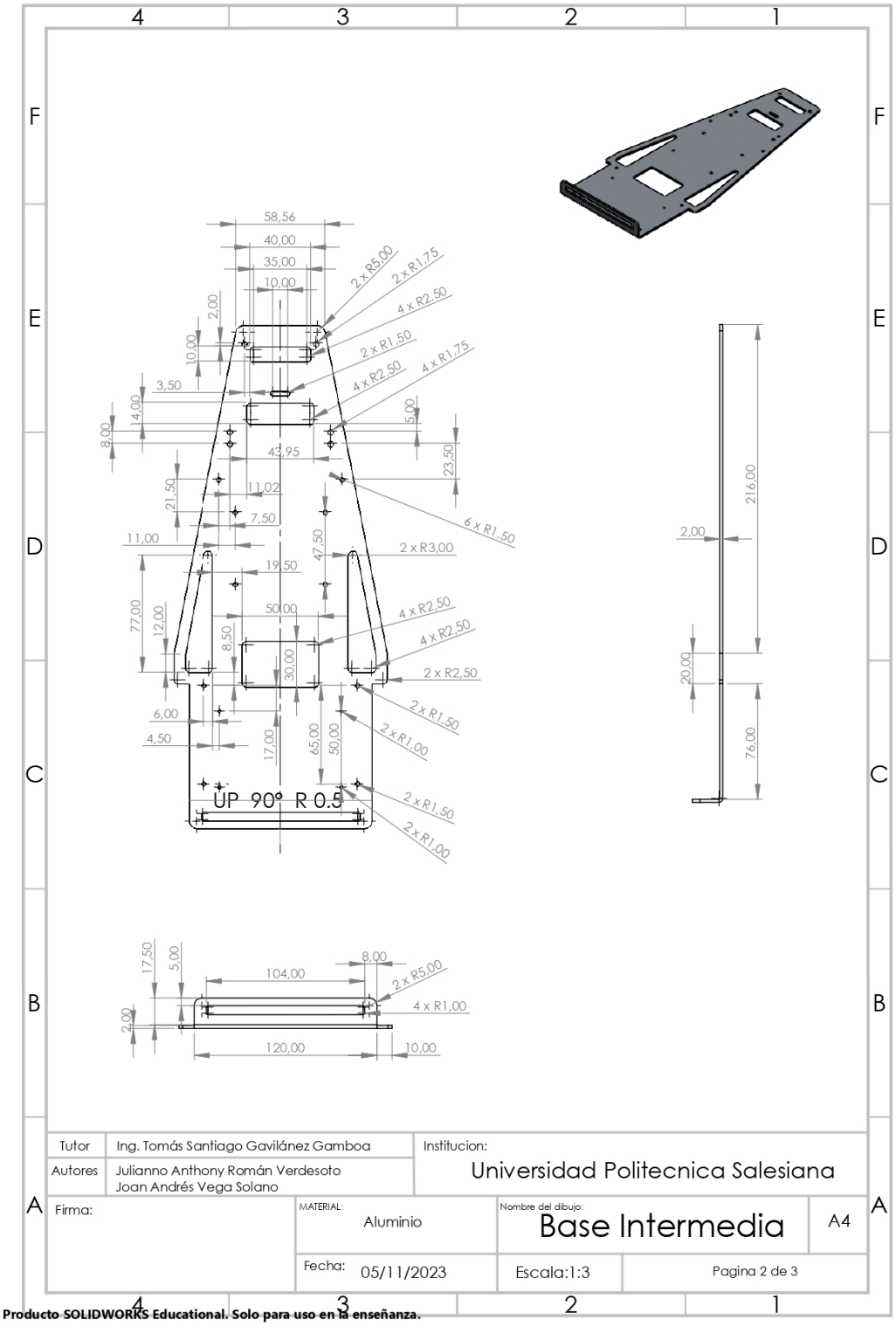


Figura 30. Plano base intermedia del robot de mapeo

Fuente: Autores.

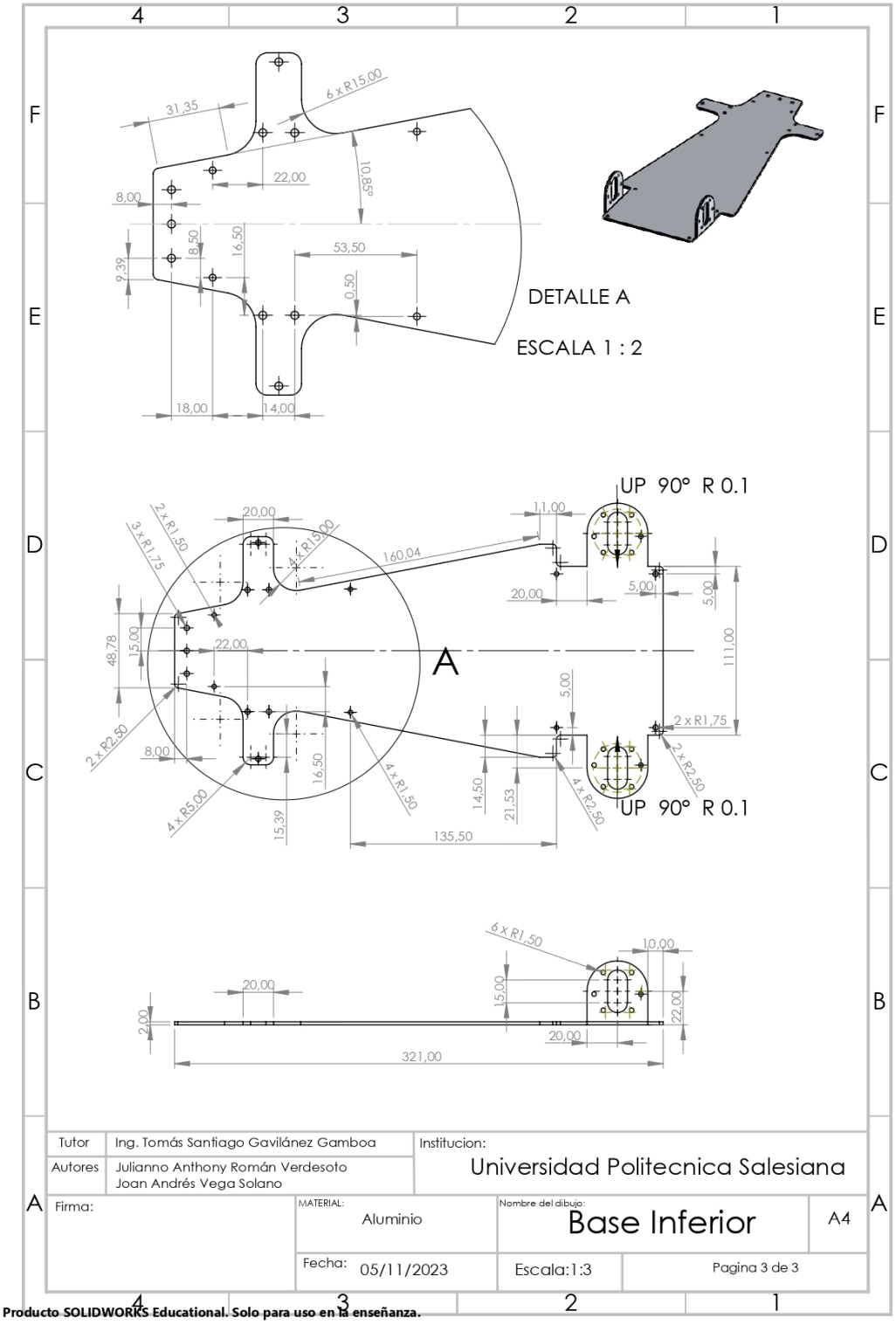


Figura 31. Plano base inferior del robot de mapeo

Fuente: Autores.