CONTROL DE NAVEGACIÓN DE UN ROBOT MÓVIL EN UN ENTORNO DINÁMICO INDOOR MEDIANTE SLAM

Cristian Muñoz Fajardo

Yojan Arvey Romero Ariza

Corporación Universitaria Autónoma del Cauca

Facultad de Ingeniería

Ingeniería Electrónica

Popayán, Colombia

2022

CONTROL DE NAVEGACIÓN DE UN ROBOT MÓVIL EN UN ENTORNO DINÁMICO INDOOR MEDIANTE SLAM

Cristian Muñoz Fajardo

Yojan Arvey Romero Ariza

Trabajo de grado para optar al título de:

Ingeniero Electrónico

Director

Mag. Yamir Hernando Bolaños Muñoz

Codirector

PhD. Pablo Eduardo Caicedo Rodríguez

Corporación Universitaria Autónoma del Cauca

Facultad de Ingeniería

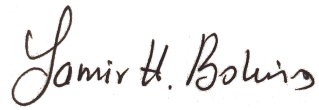
Ingeniería Electrónica

Popayán, Colombia

2022

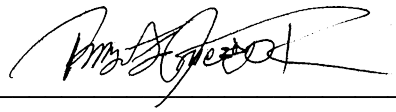
NOTA DE ACEPTACIÓN

Aprobado por el comité de grado en cumplimiento de los requisitos exigidos por la Corporación Universitaria Autónoma del Cauca para optar por el título de Ingeniero Electrónico.



Mag. Yamir Hernando Bolaños muñoz

Director



PhD. Pablo Eduardo Caicedo Rodríguez

Codirector

Jurado 1

Jurado 2

DEDICATORIAS

Dedicado especialmente a mis padres, por ser la mayor inspiración de superación y perseverancia en mi vida.

Yojan Arvey Romero Ariza

Dedico especialmente este trabajo a mis padres que a pesar de todas las dificultades siempre me han apoyado y a mis compañeros, Juan Pablo, Santiago y Yojan porque gracias al equipo que formamos culminamos satisfactoriamente este proyecto.

Cristian Muñoz Fajardo

AGRADECIMIENTOS

Agradezco inicialmente a Dios por permitirme llegar hasta este punto de mi formación como profesional, a mi director y codirector por la ayuda brindaba, a mis compañeros por la paciencia y compañía y en general a todas las personas que apoyaron en la realización de este proyecto.

Yojan Arvey Romero Ariza

Principalmente doy gracias a mis padres por apoyarme incondicionalmente a lo largo de mi formación académica, gracias a mis compañeros que de una u otra forma me apoyaron y contribuyeron en la realización de este trabajo.

Cristian Muñoz Fajardo

Índice de contenido

Índice de tablas

Índice de Figuras

Notación

OMS: Organización de Naciones Unidas

TUG: Timed Up and Go

SLAM: Simultaneous Localization and Mapping

LiDAR: Light Detection and Ranging

ROS: Robot Operating System

mmWave: Milimeter Wave

PMBM: Poisson Multi-Bernoulli

RCS: Radar Cross Section

IMU: Inertial Measurement Unit

EKF: Extended Kalman Filter

EIKF: Extended Interval Kalman Filter

AMCL: Adaptative Monte Carlo Localization

IMU: Inertial Measurement Unit

Resumen

Este trabajo presenta el diseño de un algoritmo el cual tiene como principal objetivo el realizar el seguimiento a una persona adulta en un entorno indoor dinámico, para complementar esta tarea y buscando que el sistema sea autónomo, se hace la evaluación y selección de una técnica SLAM con el fin de realizar mapeo, localización y detección de obstáculos de manera simultánea, en este caso, para la detección y segmentación de la persona se utilizó mediapipe selfie segmentation, ya que esta información es necesaria para la tarea de seguimiento.

Para la comprobación del sistema, se realizaron cuatro tipos de pruebas en un entorno indoor controlado, contando con la participación de tres sujetos de pruebas, se encontró que el sistema tiene una efectividad del 86.66% en promedio de las actividades realizadas, de estas, se pudo definir que existen dos tipos de errores principales, problemas de detección de la persona y perdida de información de localización.

Palabras claves: Seguimiento, persona adulta, SLAM, detección y segmentación.

Abstract

In this paper presents the design of an algorithm whose main objective is to monitor an adult person in a dynamic indoor environment, to complement this task and looking for the system to be autonomous, the evaluation and selection of a SLAM technique is made. In order to perform mapping, localization and detection of obstacles simultaneously, in this case, mediapipe selfie segmentation was used for the detection and segmentation of the person, since this information is necessary for the tracking task.

For the verification of the system, four types of tests were carried out in a controlled indoor environment, with the participation of three test subjects, it was found that the system has an average effectiveness of 86.66% of the activities carried out, of these, was able to define that there are two main types of errors, person detection problems and loss of location information.

Keywords: Tracking, adult person, SLAM, detection and segmentation.

CAPITULO 1: INTRODUCCIÓN

Buscando avances para la tarea de navegación en la robótica móvil surge la tecnología SLAM (Simultaneous localization and mapping), la principal característica de esta, es que haciendo uso de sensores cómo cámaras, escáneres o láser es capaz de ejecutar simultáneamente las tareas de mapeo y localización, los datos obtenidos mediante esta, contienen información que puede ser implementada para generar controles de navegación cada día más autónomos en ambientes indoor dinámicos, la navegación autónoma en la robótica móvil hace referencia a que un robot pueda circular evadiendo obstáculos y cumpliendo objetivos por un entorno sin ayuda de un operador.

Para está investigación, inicialmente se desarrolló una revisión bibliográfica sobre localización y mapeo simultáneo, donde se encontraron diferentes métodos o técnicas con las que se puede realizar este proceso, luego de esta recopilación de información, se procede a la elección del sensor y la técnica a utilizar mediante el uso de matrices de decisión, para esto fue importante considerar que se quiere hacer uso de un único sensor, es decir, el dispositivo a utilizar debe tener la capacidad de brindar información sobre el entorno, los obstáculos y de la persona a la vez, lo que llevó a la elección de una cámara RGB-D como sensor único para el sistema, de igual manera, la evaluación de la técnica se realizó con base en este último dispositivo, donde RTAB-Map (Real-Time Appearance-Based Mapping) resalta como la más adecuada para este proyecto debido a su enfoque en gráficos 3D.

Con la investigación, se busca brindar un algoritmo que genere objetivos a través de coordenadas haciendo uso de la detección y segmentación de una persona adulta, con el fin de que una plataforma robótica móvil de fabricación propia realice el seguimiento autónomo a esta, en un entorno indoor dinámico controlado, considerando la evasión de obstáculos ya sea estáticos o dinámicos, mapeo del entorno, localización actual de la plataforma y la persona con respecto a un punto inicial.

El interés profesional se encamina al incursionar en temas de nuevas tecnologías, sistemas operativos y en general temas de estudio diferentes a los vistos en el transcurso de la vida académica.

CAPITULO 2: PLANTEAMIENTO DEL PROBLEMA

Según la OMS (Organización Mundial de la Salud) la esperanza de vida de las personas ha aumentado, lo que ha generado que en los últimos años se dé un incremento en la población adulta de más de 60 años [1][2], para esta misma organización, se define una caída como un evento que lleve a un paciente a impactar con el suelo en contra de su voluntad. Estudios realizados a personas adultas determinaron que entre el 30% y 40% de las personas mayores de 60 años tienen el riesgo de sufrir una caída al año [2][3][4][5][6]. Teniendo en cuenta que la caída es un fenómeno de origen multifactorial y además presenta un riesgo de reincidencia, se han generado diferentes métodos de estimación del riesgo de caída que hoy en día son conocidos y utilizados, como la prueba timed up and go (TUG) y el test Tinetti[7][8], los cuales son realizados por profesionales de la salud.

Desde la ciencia de la electrónica, exactamente desde el área de la robótica, se han presentado algunos diseños de robots para la medicina que pueden llegar a generar un ahorro de tiempo, recursos y enriquecer la calidad de los resultados, extendiendo su cobertura con una eficiencia similar a la herramienta medica tradicional [9]. Estos desarrollos son de especial importancia para las personas de edad avanzada, debido a que permiten el monitoreo de variables fisiológicas de una forma no invasiva, ayudando así tanto física como psicológicamente a los pacientes[10].

Debido a las diferentes zonas en las que los robots se deben trasladar, se ha presentado la localización y navegación como un problema común, debido a que se requiere de algoritmos complejos ya que la mayoría de los entornos son dinámicos, donde se pueden encontrar objetos impredecibles[11][12]. En los últimos años, la tecnología GPS ha facilitado la localización en ambientes externos. No obstante, en ambientes indoor, está se vuelve inoperable[13], lo cual ha impulsado el desarrollo de nuevas tecnologías a fin de solucionar este problema [14][15][16][17], sin embargo para algunas de estas es necesario utilizar información extra, generando que el costo computacional se incremente y que algunos prototipos no sean del todo eficientes en los diferentes entornos a los que se pueden llegar a enfrentar [18], teniendo en cuenta que se busca obtener un bajo impacto al entorno, adaptando de una manera eficiente el robot a los adultos mayores sin que este altere de manera negativa las labores diarias del paciente[10][12].

De acuerdo con lo anterior se plantea la siguiente pregunta de investigación,

¿Cuál es relación existente entre el proceso de mapeo y el de navegación para una plataforma robótica en el monitoreo de adultos en un ambiente indoor dinámico controlado?

CAPITULO 3: OBJETIVOS

3.1. Objetivo general

- Desarrollar un algoritmo SLAM para el seguimiento de un adulto mediante una plataforma robótica móvil en un espacio indoor dinámico.

3.2. Objetivos específicos

- Evaluar técnicas SLAM utilizando matrices de decisión en el ámbito de diseño de la instrumentación de la plataforma robótica móvil.

- Implantar un algoritmo de control para el seguimiento de un adulto con la plataforma robótica móvil en el ambiente indoor dinámico utilizando la información generada por la técnica SLAM seleccionada.

- Evaluar el sistema robótico propuesto en un ambiente indoor dinámico controlado mediante un caso de estudio.

CAPITULO 4: ESTADO DEL ARTE

A medida que pasan los años, la localización y mapeo han ido evolucionando desde filtros de carácter genérico y mayormente lineal como el filtro de Kalman estándar. Este avance se ha dado debido a que los problemas que se presentan tienden a ser no lineales, logrando obtener algoritmos más avanzados como lo son el filtro de Kalman extendido, filtro de partículas, FastSLAM, entre otros. Posteriormente, gracias a la investigación de Smith y Cheeseman[19] se dio inicio a lo que se conoce actualmente como SLAM basándose en el método Monte Carlo Secuencial. En el SLAM se presentan dos etapas principales, la localización y mapeo, este último se divide en dos subtemas: planeamiento y seguimiento de rutas.

El SLAM es una tecnología que surge con el fin de poder identificar y mostrar la ubicación actual de un robot móvil mientras navega por un entorno desconocido[20], haciendo uso de sensores como cámaras o láser[21], funciona teniendo en cuenta la correlación entre el error de posición y el error del mapa[22], buscando planificar el movimiento del robot entre diferentes obstáculos del mundo real o representar lugares donde las personas no tienen acceso[23][24], por lo que se han presentado diferentes métodos dentro de los cuales se ha identificado que muchos presentan fallas en el momento en que se debe interactuar con objetos dinámicos[25][26], ya que muchas de estas soluciones existentes dependen de la suposición de que el entorno es en su gran mayoría estático[27], algunos de estos problemas pueden ser la falsificación de posición, mal reconocimiento de lugares ya visitados y posicionamiento erróneo del robot[28]. Para ayudar a evitar estos errores y poder contar con herramientas necesarias para SLAM se han implementado diferentes softwares dirigidos a la robótica probabilística como ROS, Orocos, CARMEN, Google Cartographer, entre otros. Hoy en día, las aplicaciones de localización y mapeo simultáneo son muy importantes en el mundo de la robótica móvil autónoma [29], por lo que se podría decir que el SLAM se ve directamente relacionado con las actividades cotidianas de las personas o actividades donde vidas humanas corran riesgo[30].

Dentro de la revisión bibliográfica realizada, se encontró que para SLAM se han diseñado diferentes metodologías por varios autores, donde se resaltan diferentes tecnologías como: (i) sensores de imagen, (ii) sensores láser, (iii) sensores de radar.

Los sensores de imagen o cámaras presentan ventajas y desventajas, debido a la reducción del coste computacional y su limitada capacidad de respuesta en los giros,

además de que permiten una amplia gama de algoritmos o métodos para tratar la información visual y generar una interacción con el entorno donde se encuentra el robot[14], por otro lado, estas pueden llegar a verse afectadas con respecto a la radiación solar, limitando su uso a entornos indoor[28], es así que las cámaras para aplicaciones SLAM se pueden dividir en: (i) cámaras monoculares, (ii) RGB-D, (iii) estereoscópicas y (iv) omnidireccionales.

La **Tabla *1***, muestra información recolectada por medio de una revisión bibliográfica, en las que se tuvo en cuenta documentos con teoría básica de cámaras y artículos donde se empleaban en algunos sistemas, es así como se encontró con información útil de cada uno de estos tipos de cámaras.

Tabla 1: *Información sensores de imagen*

Tipo de cámara	Información	Referencias
Monoculares	Solo están compuestas por el sensor de imagen y el lente, aparecieron en SLAM en el año 2007, son de bajo costo, no pueden detectar profundidad por lo que se debe estimar el mapa a escala lo cual puede generar distorsión, por lo general se utilizan para detectar esquinas; algunos trabajos indican que para el uso de estas se debe hacer el análisis de cambios generados por el movimiento en las imágenes capturadas, requieren de un entorno idealmente iluminado, pueden llegar a trabajar a 30 fps.	[31],[32], [33], [34], [35], [36]
RGB-D	Permiten generar métodos robustos para la identificación de objetos dinámicos los cuales alteran la localización y navegación simultánea en espacios indoor, tienen una gran ventaja en cuanto a continuidad y suavidad, son las primeras que se presentaron para SLAM a color, unas nacieron con otros fines como el Kinect, pero debido al funcionamiento se les ha dado uso en la robótica móvil, este tipo de cámaras genera una nube de puntos, la medición de distancia la hacen por medio del emisor y receptor láser, se catalogan como sensores de bajo costo, consumo y tamaño, la resolución varía dependiendo el tipo de cámara y trabajan desde los 30 fps a condiciones ideales, se pueden formar sistemas multicámara con estas,	[29], [37], [38], [39], [40], [41], [42][43], [44], [45], [46]

	la profundidad puede llegar a ser una variable ruidosa pero ya existen filtros que la controlan.	
Estereoscópicas	Junto con odometría, estas cámaras pueden estimar de manera precisa y confiable la localización y mapeo, no presentan el problema de detección de profundidad. En trabajos realizados se encontró que pueden adquirir datos en un ángulo de 110°, entregan una nube de puntos al controlador para realizar localización, detección y planeación de trayectorias.	[47], [48], [49]
Omnidireccionales	Son de los dispositivos que más información visual proporcionan, ya que permiten obtener visión de 360° con lo cual un fotograma tiene mucha más información, existen dos tipos de configuración de visión omnidireccional, combinar cámaras con superficies convexas buscando extender el campo de visión de la cámara “configuración catadióptrica” y configurar una serie de cámaras alrededor de la plataforma apuntando a diferentes direcciones, diferentes trabajos realizados con estas configuraciones han permitido demostrar el funcionamiento para desarrollar SLAM, aunque se tiende a presentar fallas con la localización debido a que la iluminación del ambiente no es ideal.	[50], [51], [52]

Por otro lado, se encuentran los sensores láser por sus siglas en inglés (Light Amplification by Stimulated Emission of Radiation), los cuales utilizan como base de su funcionamiento la teoría del tiempo de vuelo, teniendo en cuenta que la velocidad de las señales electromagnéticas es 0.3 m/ns[53].

Hasta el momento de esta investigación, los sensores láser que más destacan por su funcionamiento son los conocidos como LiDAR por sus siglas en inglés (Laser Imaging Detection And Ranging), es así que varios investigadores han encontrado la forma de utilizar esta tecnología para el problema de SLAM[54], teniendo en cuenta que los modelos de bajo costo de esta tecnología tienen una precisión de 2mm, una resolución de 0.9° y un ángulo de operación de 360°, lo que hace que este tipo de sensores sean utilizados en entornos indoor, en cuanto a su funcionamiento, se tiene la

posibilidad de variar la frecuencia de operación, aunque se debe resaltar que la frecuencia y el costo computacional son variables directamente proporcionales[25], siendo el costo computacional un problema que se puede llegar a disminuir con la ayuda de otros sensores. Álvarez y Jiménez[55] realizaron un sistema multisensorial en que hacían la combinación de un LiDAR RPLiDAR a2 y una cámara estereoscópica, con el fin de generar un mapa 2D del entorno y de poder hacer el seguimiento del robot, mientras se validaba el funcionamiento total del sistema, se encontró que con el sensor láser se alcanzaban errores menores al 1% mientras que con la cámara se alcanzaban errores máximos del 4%, esta tecnología LiDAR brinda la distancia absoluta y la orientación con base en la pose del robot y puede llegar a capturar entre 4000 y 8000 muestras por segundo en un rango aproximado de 12 metros[22].

Otro sensor láser que se utiliza para SLAM es el Hokuyo URG-04LX, el cual tiene un rango de operación de 240°, utiliza una fuente de luz de 785nm aproximadamente y tiene un espacio de escaneo de 4000mm, este sensor es fabricado para ser utilizado exclusivamente en entornos indoor y es catalogado de bajo costo[40], Rodríguez[56] por rapidez, exactitud y resolución angular utilizó este sensor en su sistema logrando extraer características lineales, considerando que se contaba con un coeficiente de correlación lineal aceptable, para esta propuesta, se tuvieron que realizar aproximadamente 500 muestras a distintos colores para variar la información dependiendo del color y el brillo, además se da a conocer que para cualquier planteamiento de este tipo, sin importar algoritmo o sensores es muy importante considerar los posibles errores en los modelamientos sensoriales.

La tecnología mmWave (Millimeter Wave), en los últimos años ha sido ampliamente aplicada en la localización y mapeo de plataformas robóticas móviles. No obstante, sigue siendo una tecnología bastante joven en este ámbito, aun así, se han realizado diversas investigaciones sobre el tema, debido a que realiza una propagación geométrica del entorno y puede capturar mediciones de alta resolución angular y temporal, de esta forma logrando realizar SLAM en plataformas robóticas móviles, también logra desempeñarse en ambientes extremos como niebla, humo o baja visibilidad, ambientes donde la tecnología LiDAR o cámaras pueden fallar, sin embargo, el mmWave requiere una alta complejidad computacional, impidiendo que se realice SLAM en tiempo real[57]. Por lo cual, Yu Ge y colaboradores[58] proponen un filtro SLAM de baja complejidad, que reduzca el coste computacional que el mmWave presenta, utilizando el filtro de Poisson multi-Bernoulli (PMBM), y todo mientras

mantiene el rendimiento esperado para el SLAM en tiempo real. Por otro lado, Mingsheng y colaboradores[59] proponen una combinación donde el mmWave hace las funciones de emisor el cual transmite las señales de onda milimétrica y una plataforma robótica móvil ubica estas señales e intenta navegar hasta el emisor. Yang Li y colaboradores[60] desarrollaron un sistema SLAM que ayuda a mejorar las condiciones de este con mmWave, los cuales combinan RCS que ayuda a mejorar el problema de la poca precisión del posicionamiento del SLAM, debido a la baja precisión de los datos que otorga el radar mmWave y por último se implementó un IMU, que combina nubes de puntos de escaneo continuo en “Multi-Scan”, esto ayuda con el bajo volumen de datos que el sensor mmWave presenta. De igual manera, Palacios y colaboradores[61], evaluaron un algoritmo SLAM especializado para la tecnología mmWave llamado CLAM, el cual funciona sin información inicial del despliegue de la red o el entorno y gracias a una reformulación se logra una baja complejidad computacional, esto se evaluó con un hardware mmWave de 60 GHz, el cual logra una precisión aceptable en la mayoría de los casos.

Una parte importante en la robótica móvil autónoma, son las técnicas empleadas para convertir la información de los sensores en datos que permitan la localización y mapeo del entorno en el cual se encuentra la plataforma, existen diferentes técnicas desarrolladas para esta función, como: (i) Filtro de Kalman, (ii) Filtro de Kalman extendido, (iii) Filtro de partículas, (iv) Filtro de Monte Carlo, (v) Redes neuronales, (vi) Graph-SLAM, (vii) Gmapping y (viii) RTAB-Map.

El filtro de Kalman, es uno de los primeros filtros utilizados en la localización y mapeo, sin embargo, su comportamiento no lineal lo deja obsoleto ante las condiciones de la robótica actual, donde la mayoría de los casos son no lineales, por ende, actualmente se encuentra poca bibliografía sobre el tema, por otro lado, el filtro de Kalman Extendido o EKF(Extended Kalman Filter) de carácter no lineal resulta efectivo en ambientes indoor dinámicos, por lo cual ha ganado popularidad en la localización robótica[62]. No obstante, el EKF también presenta algunos inconvenientes como puede ser el ruido del sistema, ocasionando que las características estadísticas previas del ruido detectado no se pueden predecir con precisión. Por esto Zhu-Li Ren y Colaboradores[63], proponen mejorar el Filtro de Kalman Extendido con la inclusión de SLAM adaptativo Fuzzy, el cual mostró una mejora del ruido sobre el filtro y aumentó la precisión de posicionamiento en un 53,8%, esto en comparación con el Filtro de Kalman Extendido estándar. De igual forma Vincent y Colaboradores[64], presentan una mejora para el

EKF que implica utilizar redes neuronales profundas, logrando mejorar la posición de los objetos dinámicos rastreados y un mapa 3D libre de esos objetos, por consiguiente, este método obtiene un rendimiento comparable con otros de última generación. Por otro lado, se han implementado nuevas variantes de filtro de Kalman, una de ellas es la UKF (Unscented Kalman Filter) que en comparación con el EKF presenta mejoras en precisión de localización promedio[65].

Otro algoritmo que se ha popularizado en la localización robótica es el Filtro de Partículas, ya que este puede ser modificado de diferentes formas para cumplir las necesidades que se presenten, esto debido a que el filtro tiene la capacidad de resolver problemas no-lineales y no-gaussianos. No obstante, el problema que presenta el filtro de partículas estándar es el coste computacional [66], este problema puede ser solucionado con la aplicación de un filtro marginalizado o filtro Rao-Blackwellized, el cual se compone por el filtro de partículas y el filtro de Kalman, donde el filtro de partículas resuelve el componente lineal y el filtro de Kalman el no-lineal, de esta forma se logra reducir el coste computacional global, igualmente se consigue mejorar la estimación, esto gracias al filtro de Kalman por ser un estimador óptimo[67]. Una aplicación de esto se encuentra en el trabajo de Jingwen Luo y Shiyin Qin[68] quienes proponen un filtro de intervalo combinatorio, el cual lo componen un filtro de partículas de caja mejorado (IBPF), que reduce el coste computacional y mejora el rendimiento en tiempo real del algoritmo, también se emplea para realizar la localización simultánea, además se incorporó el Filtro de Kalman de Intervalo extendido (EIKF), que es similar al Filtro de Kalman Extendido estándar en términos de estadística y rendimiento, por otro lado, el EIKF tiene una eficiente adaptabilidad y robustez y gracias a esto es el que se encargará de realizar el mapa. Por otro lado, Dhruv Talwar y Seul Jung[69] desarrollaron un algoritmo con la localización adaptativa de Monte Carlo (AMCL) y un filtro de partículas logrando un óptimo rendimiento en ambientes indoor dinámicos, debido a que el filtro pudo converger las partículas de manera rápida y de esta manera se logró obtener la posición y orientación de la plataforma móvil.

El filtro Monte Carlo, es uno de los más aplicados en la localización robótica, el cual emplea técnicas probabilísticas de los filtros de partículas, para de esta forma lograr predecir la posición y orientación de la plataforma robótica en el mapa, también existen variantes del filtro Monte Carlo más orientadas a la localización, como es el caso del AMCL (Localización adaptativa de Monte Carlo) que ajusta el número de muestras mientras se ejecuta[70], [71]. Otra variante se puede observar en el trabajo de Doherty

y colaboradores[72] donde desarrollaron un nuevo algoritmo Monte Carlo, VC-SMC (Variational Cópula Monte Carlo secuencial) el cual está basado en el filtro de partículas, este algoritmo logra realizar SLAM cuando los parámetros del modelo son desconocidos y mejora la precisión de la inferencia cuando hay asociación de datos inciertos.

Las redes neuronales son otra técnica que se está implementando hoy en día, principalmente para los sistemas que trabajan con imágenes, buscando ampliar sus campos de trabajo en la medicina, industria e incluso en trabajos militares[73], la redes neuronales generalmente se componen por tres capas: inicialmente se tiene la capa de entrada, que es la encargada de recibir la información por parte del sensor; la capa de salida, encargada de dar una orden al sistema; y entre las dos, se encuentra la capa oculta, que es donde sucede todo el control, comparación y análisis de la información que permitan evidenciar el entrenamiento de la red neuronal[74], este entrenamiento es el proceso más importante al momento de utilizar redes neuronales, ya que de este depende todo el proceso de aprendizaje, el objetivo principal del entrenamiento debe ser que el sistema pueda identificar entradas que nunca haya visto y poder reaccionar de manera adecuada ante estas[75], el funcionamiento interno de las redes neuronales se basa en que las decisiones que estas tomen van a generar unas recompensas o unas penalizaciones, buscando llegar a un momento ideal donde la red no pueda mejorar más. Uno de los principales problemas de las redes neuronales es que para aplicaciones muy grandes se puede necesitar un coste computacional muy alto, aunque para esto ya se han planteado diseños como el de Royo[76], en el que se logró disminuir el tiempo de entrenamiento y consumo de datos en un 90% y 95% respectivamente, haciendo uso de un método de aprendizaje denominado few-shot, algo que llama la atención de los investigadores para hacer uso de estas redes, es que además de permitir la detección de objetos de manera rápida y precisa[77], es una técnica que funciona por partes separadas, es decir, si una parte presenta una falla solo dejara de funcionar dicha sección de la red neuronal[78]. Cabe aclarar que las redes neuronales también se pueden utilizar con otros sensores como en el caso de Rico y Escobar[79], quienes hicieron uso de encoder's y sensores IMU para rectificar las lecturas y prevenir posibles choques de la plataforma robótica con alguna pared u objeto no detectado por la cámara estereoscópica que utilizaron como sensor principal, un ejemplo del amplio uso que se le está dando a esta técnica es la gran cantidad de intentos para la autonomía total que están realizando grandes empresas como Google, Uber, Tesla o Amazon[74].

Otra técnica para SLAM basada en imágenes es Graph-SLAM, esta se puede catalogar como offline debido a que utiliza información recolectada con anterioridad para lograr hacer SLAM con robots[80], el funcionamiento de Graph-SLAM se basa en nodos y bordes, los nodos son estimaciones de posición del robot y los bordes son mediciones que hacen los sensores con las que obligan al robot a hacer un cambio en la navegación[81], Karam y compañía[82] hicieron uso de esta técnica junto con un sensor LiDAR y un IMU para obtener un mapeo de un entorno inicialmente desconocido, logrando obtener un buen funcionamiento para detectar edificios, escaleras, paredes curvas e incluso el sistema tuvo la capacidad de moverse de manera eficiente por espacios desordenados, en algunos casos se genera el mapa confiando solo en la información dada por el sensor principal, aun así, cuando son ambientes complejos se utilizan algunos métodos para optimizar el mapa, como es el caso del diseño de Astrup y colaboradores[83], en el cual se logró realizar el mapeo de un bosque con la ayuda de un LiDAR, un IMU, una cámara estereoscópica y un GPS en conjunto con Graph-SLAM, con ayuda de algunos experimentos prácticos de este, se logró llegar a la conclusión de que la precisión se ve afectada dependiendo de la cantidad de trafico de objetos en el ambiente, más aún si estos objetos son dinámicos y de gran tamaño, ya que pueden ocupar gran parte o incluso todo el campo visual del sensor principal[84], esto ha llevado a que se hagan cambios de rol por parte del robot, en[85] se han instalado algunos puntos de referencia los cuales detectan el robot, buscando obtener mejoras en tareas de detección y en fallas por transmisión de información.

Por otro lado, Gmapping es un algoritmo reciente que tiene como base el filtro de partículas RaoBlackwelized y se considera una mejora al algoritmo Fast-SLAM[71], [86], esta técnica es capaz de realizar el mapeo por medio de cuadriculas de ocupación, el mapa se genera en escala de grises y se puede identificar espacios libres, paredes y espacios no explorados[87], actualmente Gmapping es uno de los algoritmos más utilizados en robótica móvil terrestre, en su mayoría espacios pequeños[88] y aunque se considera seguro debido a que no solo considera el movimiento del robot sino que también tiene en cuenta las lecturas hechas por el sensor, necesita sensores de odometría auxiliares para su correcto funcionamiento[89], debido a la combinación de estos sensores, esta técnica puede llegar a tener retrasos en la generación del mapa, lo que causaría que algunos espacios del entorno no se registren correctamente, evidenciándose en el mapa como espacios vacíos, este problema también se encuentra relacionado con el aumento de velocidad de las plataformas robóticas[90], como era de esperarse, también se han realizado diseños con el fin de mejorar esta técnica, es así,

que se encuentran trabajos donde se busca optimizar las funciones con el objetivo de obtener una disminución del coste computacional y requerimientos hardware[91].

Finalmente, RTAB-Map (Real Time Appearance-Based Mapping) es un algoritmo que crea mapas de entornos operando en ciclo cerrado usando nodos que permiten la recepción de información de diferentes sensores[92], además brinda métodos para realizar tareas de SLAM en tiempo real obteniendo resultados en 2D y 3D[93], [94], haciendo uso de una estimación de pose por medio de la comparación de datos obtenidos que permiten la detección de puntos de referencia y de esta forma conocer la localización de la plataforma en el entorno, generando así la posibilidad de trabajar sin utilizar información de odometría por sensores adicionales y sensores inerciales[92]. En trabajos como[95] se realiza la implementación de RTAB-Map con sensores LiDAR y cámaras RGB-D obteniendo resultados en tiempo real en diferentes entornos, además en pruebas realizadas por [96] se obtuvo que RTAB-Map brindó resultados que resaltan sobre otro tipo de algoritmos.

CAPITULO 5: MATERIALES Y EQUIPOS

En esta sección, se hará una descripción ligera de los dispositivos y medios utilizados con el fin de expresar cuál es la finalidad del uso de cada uno de estos, generando la posibilidad de que en un futuro se puedan realizar y evaluar las pruebas con el mismo principio de funcionamiento, se brindan especificaciones técnicas de estos y además se dará una idea de cómo se ejecutó la construcción de la plataforma robótica móvil.

5.1. Unidad de procesamiento

El coste computacional que requiere el SLAM puede ser demandante[18], es por lo que se recomienda utilizar un equipo con requerimientos mínimos en procesamiento y tarjeta gráfica, en este caso se utilizó un equipo con las especificaciones indicadas en **Tabla *2***.

Tabla 2: *Especificaciones computador portátil*

Característica	Valor
Marca	Acer
Procesador	Intel Core i5 8th Gen
Tarjeta gráfica	NVIDIA GTX 1050 4Gb GDDR5
RAM	20Gb
Sistema operativo	Ubuntu 20.04 LTS

5.2. Sensor

Como instrumento de captura de información para la tarea de mapeo, localización y navegación se utilizó una cámara RGB-D, luego de una comparación con otro tipo de sensores que se encuentra en la **Tabla 5** para este caso se hizo uso de una cámara Intel RealSense D415 como la de **Figura 1**, la cual cuenta con las especificaciones presentadas en **Tabla *3***

***Figura 1*:** Cámara Intel RealSense D415
Fuente propia

Tabla 3: *Tabla de especificaciones Intel RealSense D415*[97]

Característica	Especificación
Rango de funcionamiento	0.5m hasta 3m
Tipo de ambiente	Interior y exterior
Tecnología de profundidad	Estereoscópica
Resolución de profundidad	1280 x 720
Velocidad de fotogramas de profundidad	90
Resolución RGB	1920 x 1080
Velocidad de fotogramas RGB	30
Distancia mínima de profundidad	0.45m

5.3. Plataforma móvil

Para la plataforma móvil se optó por fabricar el robot desde cero, utilizando una lámina de acero que cumplirá la función de chasis, motores DC y además se utilizó un sistema de orugas para el movimiento tal y como se observa en **Figura 2**, adicional a eso se agregó un controlador, batería y otros, en **Tabla *4*** se encuentran las especificaciones completas de la plataforma móvil, los diagramas de fabricación se encuentran en los documentos anexos.

Tabla 4: *Especificaciones de plataforma móvil*

Característica	Valor
Dimensiones (cm)	33x19
Motores	DC, 9V, 9.5Kg/cm, 150RPM
Controlador	Arduino Mega2560
Rueda dentada	Metálica
Controlador de motores	Motor shield L298P
Oruga	Plástica
Material del chasis	Acero

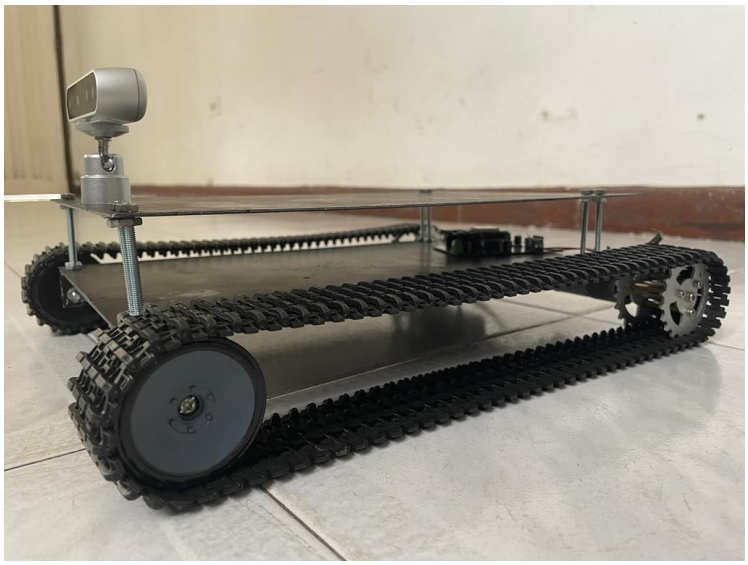


Figura 2: Prototipo de plataforma robótica móvil
Fuente propia

5.4. Software

Para la creación de algoritmos, visualización de información y pruebas, es necesario hacer uso de un conjunto de desarrollo software, para este caso, inicialmente se considera el uso de Ubuntu en su versión 20.04 LTS basado en Linux de 64 bits, sobre el cual se va a utilizar ROS, un sistema operativo cuyo principal objetivo es el desarrollo robótico. Para la visualización de información se utilizó Rviz, es una herramienta de ROS que permite visualizar datos 3D como trayectorias y nubes de puntos, es configurable a elección del usuario y además tiene adaptabilidad a diversos sensores entre ellos las cámaras RGB-D.

CAPITULO 6: METODOLOGÍA

Para realizar con éxito la investigación, se organizaron todas las tareas por hacer en términos generales, produciendo así un orden en la secuencia para estas. A continuación, se detallan todas las fases por la que paso el proyecto paso a paso hasta cumplir con los objetivos propuestos.

Dada la revisión bibliográfica, se evidenciaron diferentes métodos y técnicas que buscan dar solución a la problemática de SLAM, **Tabla 5** y **Tabla *6*** son la representación de matrices de decisión elaboradas con el fin de elegir la mejor opción que se puede utilizar para este caso en específico, estas muestran la selección del tipo de sensor y de la técnica SLAM a utilizar respectivamente, para la elaboración de estas matrices se tuvo en cuenta la información recolectada para el estado del arte, se decidió utilizar el sistema de pesos para la matriz, donde cada característica tiene un peso y la suma de los pesos de todas las características debe ser 100%, luego se define un rango de calificación (de 0 a 10 en este caso),una vez hecho esto, el peso de cada característica se multiplica por la calificación de cada opción y se suma para tener un total, los pesos y calificaciones se asignaron a criterio propio de los creadores del documento.

6.1. Selección del sensor

Para la matriz de decisión de sensores de **Tabla 5**, se tuvieron en cuenta los tipos de sensores encontrados mientras se realizó la revisión bibliográfica, principalmente se analizaron los sensores láser LiDAR, los escáner mmWave y las cámaras RGB-D, es importante resaltar que al momento que se realizó la investigación, la disponibilidad de los escáneres fue limitada, mientras que por su parte los sensores láser tienen un costo considerablemente alto lo cual generaba un incremento en el presupuesto planteado, finalmente se seleccionó las cámaras RGB-D como sensores a utilizar debido a que permiten generar una imagen en RGB que se utilizará para la detección del sujeto, generan una nube de puntos que permite la detección de obstáculos y también brindan la distancia a un punto.

Tabla 5: Matriz de decisión para tipos de sensores

Características	Peso	Sensores láser	Ponderado	Radar	Ponderado	Cámaras RGB-D	Ponderados
Velocidad de comunicación	15%	7	1.05	8	1.2	9	1.35
Coste computacional	10%	6	0.6	7	0.7	5	0.5
Consumo de corriente	10%	6	0.6	8	0.8	9	0.9
Costo	20%	1	0.2	1	0.2	5	1
Software	5%	8	0.4	9	0.45	10	0.5
Rango	5%	9	0.45	9	0.45	8	0.4
Disponibilidad	10%	3	0.3	0	0	9	0.9
Error	5%	8	0.4	5	0.25	7	0.35
Detección	10%	3	0.3	10	1	3	0.3
Interacción con el ambiente	10%	8	0.8	7	0.7	2	0.2
TOTAL	**100%**	**N/A**	**5.1**	**N/A**	**5.75**	**N/A**	**6.4**

6.2. Selección de la técnica SLAM

Utilizando el mismo método de la sección anterior, se generó una matriz de decisión que cumpla con el objetivo de determinar cuál será la técnica SLAM a utilizar para el desarrollo planteado en este documento, en este caso, se tuvieron en cuenta las técnicas que se encontraron en el proceso de la revisión bibliográfica, las cuales se evaluaron en diferentes criterios como se muestra en la **Tabla *6***, finalmente la técnica con el ponderado más alto fue RTAB-Map considerando que la adaptabilidad a la cámara RGB-D como la característica más importante, el resultado total se obtuvo debido a que es una técnica robusta haciendo uso de información 3D para las tareas de generación del mapa y localización.

Tabla 6: *Matriz de decisión para tipos de técnicas de SLAM*

Características	Peso	RTAB-map	Pond.	ORB-SLAM2	Pond.	Gmapping	Pond.	Graph-SLAM	Pond.
Adaptabilidad a sensores	25%	9	2.25	8	2	5	1.25	8	2
Compatibilidad con software	15%	10	1.5	9	1.35	9	1.35	8	1.2
Error	10%	7	0.7	8	0.8	9	0.9	6	0.6
Tiempo de ejecución	10%	10	1	10	1	10	1	10	1
Caracterización del mapeo	20%	9	1.8	7	1.4	9	1.8	8	1.6
Caracterización de localización	20%	9	1.8	7	1.4	9	1.8	8	1.6
TOTAL	**100%**	**N/A**	**9.05**	**N/A**	**7.95**	**N/A**	**8.1**	**N/A**	**8**

6.3. Calibración de sensor

La cámara Intel RealSense cuenta con la opción de calibrar desde herramientas desarrolladas directamente por Intel, para este caso, se calibró utilizando la herramienta de calibración dinámica para la serie D400, con el fin de llevar a cabo este proceso se realizó el seguimiento del paso a paso ubicado en la documentación oficial de la herramienta utilizada[98], en **Error! Reference source not found.** se muestran algunos de los puntos realizados para este proceso, con esta calibración se busca cambiar variables cuantitativas del sensor que le permitan realizar una mejor identificación de la profundidad, esto generando una mejor detección de distancia a un punto.

Figura 3: Calibración del sensor
Fuente propia

6.4. Desarrollo

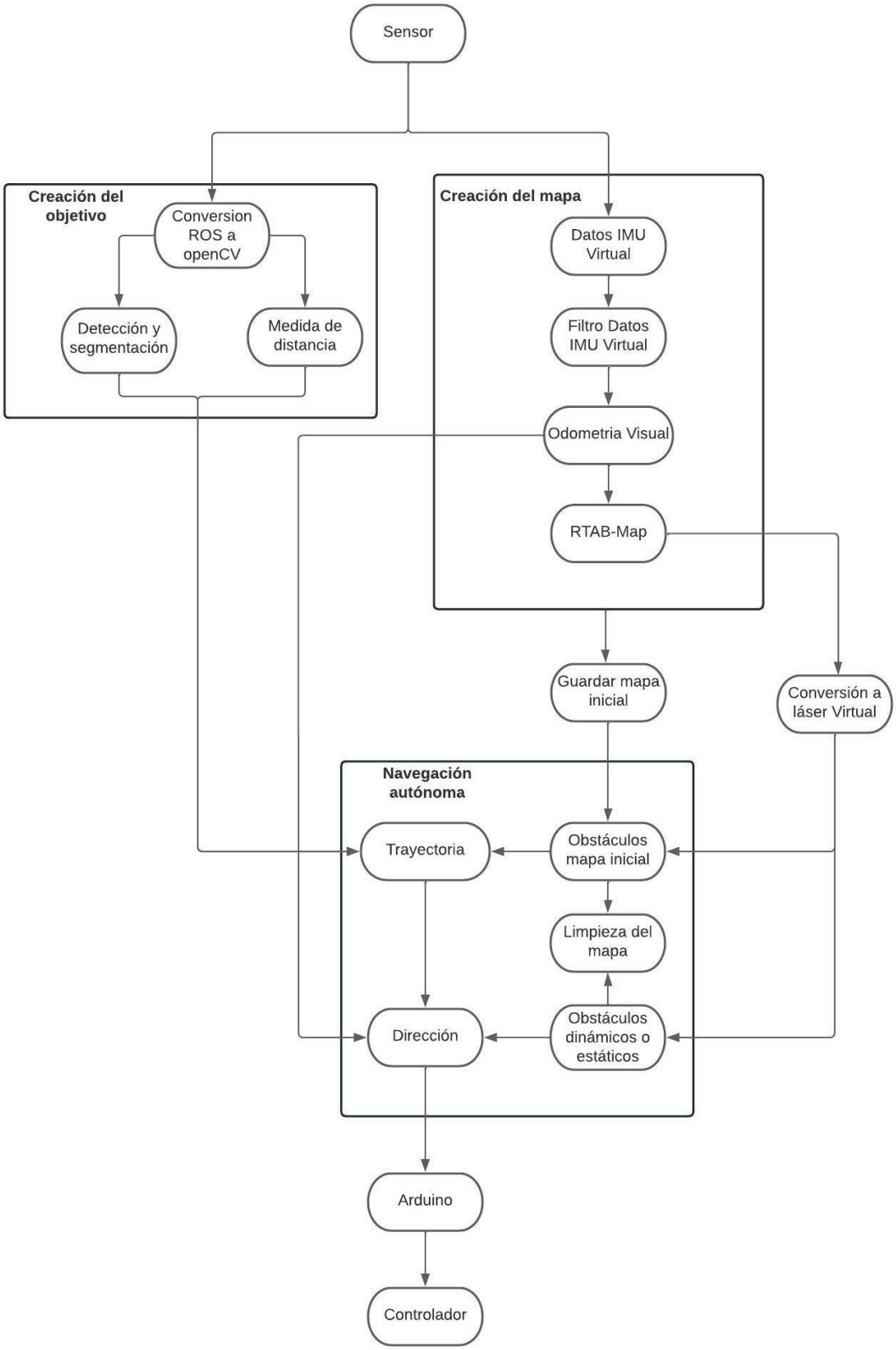


Figura 4: Diagrama de flujo general
Fuente propia

Con el fin de cumplir con los objetivos, en **Figura 4** se planteó un diagrama de flujo de todo el sistema, representando el paso a paso de todas las tareas que deben ejecutar los algoritmos propuestos, tomando como origen el sensor que es el encargado de brindar la información visual, siendo esta la base para todo el funcionamiento. Se definieron tres tareas principales para la propuesta: creación del objetivo, creación del mapa y navegación autónoma; cada una de estas tiene diferentes procesos y subtareas de las cuales depende su desempeño, en esta sección se hará una explicación detallada de cada una de estas tareas identificando las entradas y salidas de cada uno de los bloques con el fin de sustentar su uso y hacer reproducible el proyecto para trabajos futuros.

6.4.1. Creación del mapa

Considerando que para el desarrollo de las actividades es necesario hacer uso del mapa del entorno, se requiere cumplir con algunos subtemas para realizar dicha tarea, es así, que para poder llevarla a cabo se hace uso del proceso indicado en **Figura 5**.

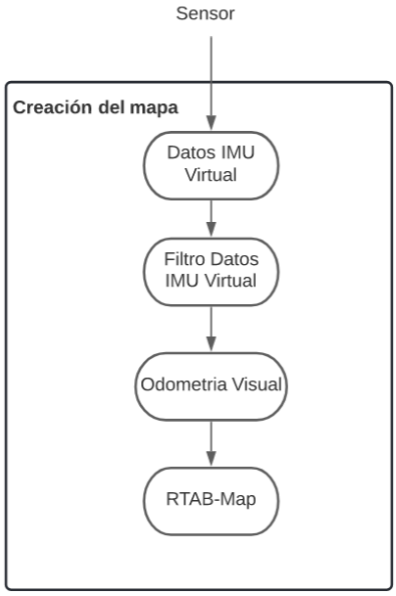


Figura 5: Diagrama de flujo de creación del mapa
Fuente propia

El mapa se crea gracias al sensor y métodos seleccionados, que por medio de una nube de puntos genera un mapa de ocupación 2D, donde los objetos como paredes, mesas, sillas, entre otros, los representa con pixeles de color negro, denotando que son objetos a los que la plataforma no podrá ir o traspasar, por otro lado el terreno libre como el piso lo representara con pixeles de color blanco, que será por donde la plataforma podrá navegar como se muestra en la **Figura 6Figura 6**. Este paso es muy importante para la navegación autónoma.

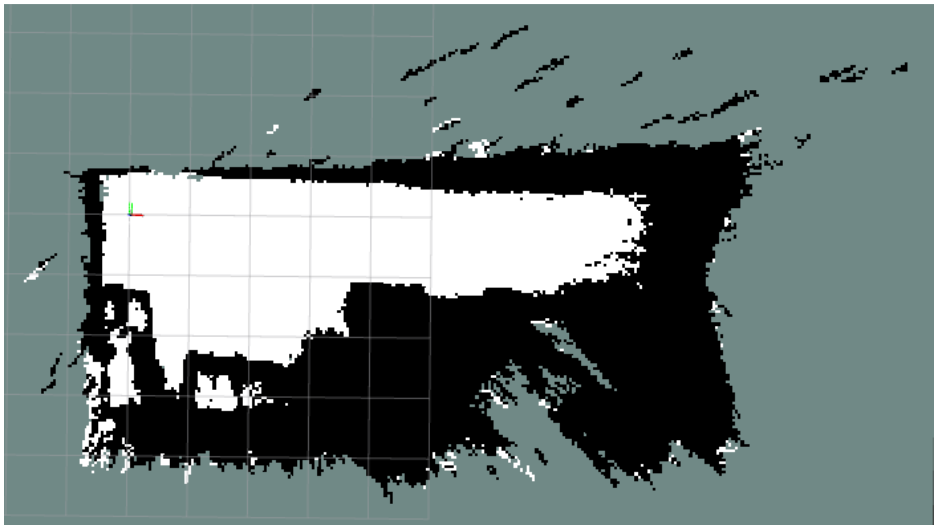
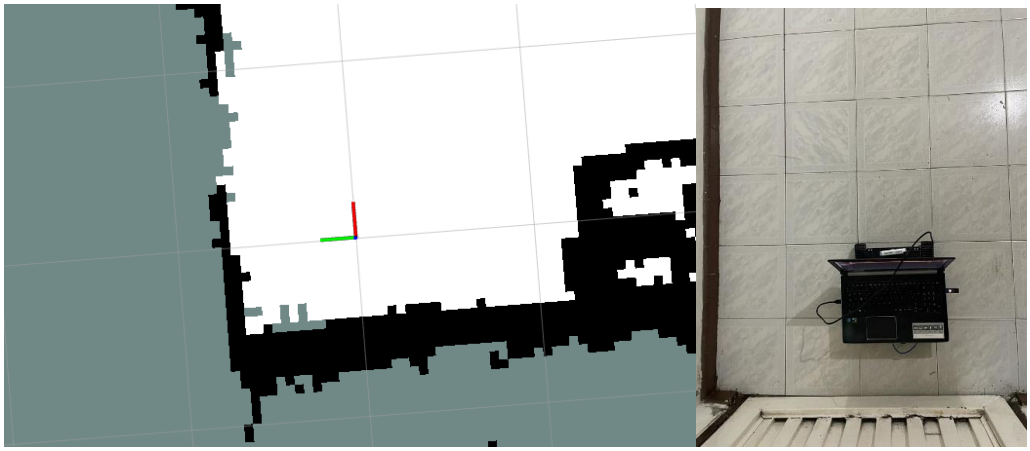


Figura 6: Imagen del mapa
Fuente propia

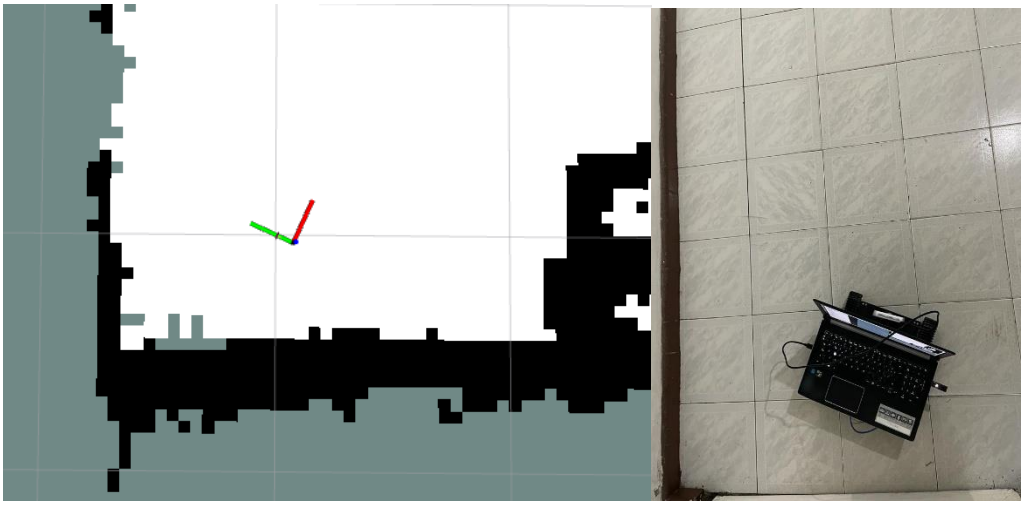
Para lograr esto, es necesario que los fotogramas que otorga el sensor sean procesados como datos IMU virtuales (velocidades angulares y aceleraciones lineales), estos son necesarios debido a que el sensor no cuenta con un dispositivo hardware que proporcione este tipo de información. En segundo lugar, se filtrarán y fusionarán los datos IMU virtuales, con esto permitiendo conocer la orientación de la plataforma móvil robótica (como se muestra en la **Figura 7**).

Tercero**,** mediante la implementación de un estimador de estado no lineal para plataformas robóticas en espacios 3D, se logra procesar los datos IMU virtuales filtrados y por medio de estos se logra rastrear el estado del robot y generar odometría visual utilizando cámaras RGB-D, esto se representa en la **Figura 8**, donde con el uso de un único sensor se obtienen coordenadas de posición X,Y y Z y valores de orientación X, Y, Z y W, que representan la orientación de la plataforma, de igual manera en la **Figura 9** se evidencia cómo estas coordenadas se representan en el mapa de ocupación 2D desde un punto inicial.

Por último, las coordenadas emitidas por la odometría visual son enviadas a la técnica SLAM seleccionada, que usando un enfoque RGB-D genera una nube de puntos 3D y por medio de esta se puede generar un mapa de ocupación 2D esencial para la navegación, por tales motivos, y en conjunto de una navegación autónoma, se logra el SLAM del método propuesto.



(a)



(b)

Figura 7: Representación de giro mediante odometría visual
Fuente propia

```
yojan@Pc-Yojan:~/catkin_ws$ rostopic echo /odometry/filtered
header:
  seq: 3400
  stamp:
    secs: 1663376145
    nsecs: 681362391
  frame_id: "odom"
child_frame_id: "camera_link"
pose:
  pose:
    position:
      x: 0.01529297266586127
      y: 0.024300182793695058
      z: -0.006339214119122971
    orientation:
      x: 0.00045819505674640174
      y: -0.0006872087423831164
      z: -0.0004097269680663394
      w: 0.9999995749625324
  covariance: [0.03610424529587132, -4.6090506872080076e-07, -2.573100986067518e
-07, -3.6611907061584276e-08, -1.4394247045413833e-05, -7.569701271418839e-05, -
4.6090506872080076e-07, 0.03610505879913546, -1.717725063767126e-07, 1.450229999
4011602e-05, 1.1967888227058123e-08, 0.0001674693120452454, -2.573100986067518e-
07, -1.7177250637671176e-07, 0.04184545817085629, 4.606241356474866e-05, -0.0001
```

Figura 8*:* Valores de odometría visual presentados desde la terminal
Fuente propia

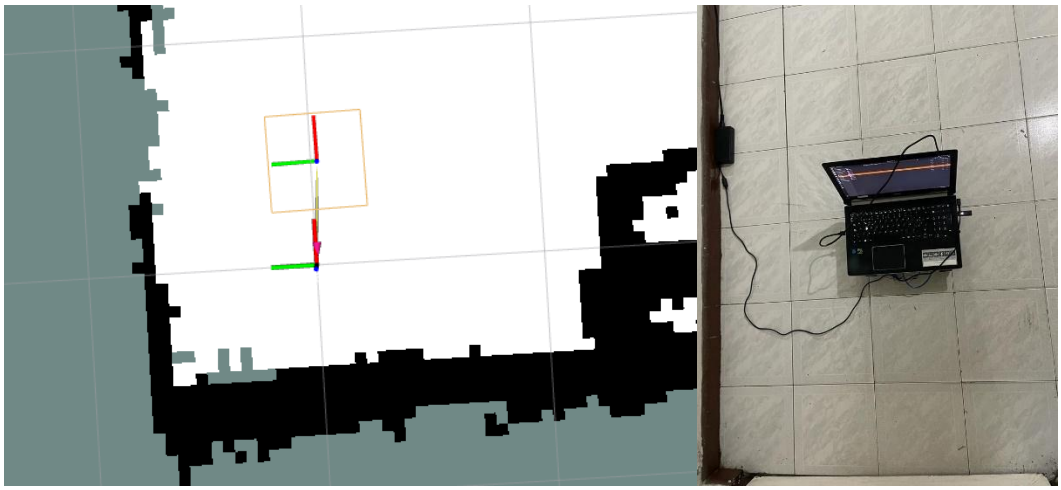


Figura 9: Representación gráfica de localización con respecto al punto inicial
Fuente propia

6.4.2. Navegación Autónoma

La navegación autónoma tomará información del sensor y odometría, emitiendo comandos de velocidad que se envían a la plataforma móvil robótica, esto es necesario para el control de movimiento ya que brinda la velocidad lineal y angular, de este modo la plataforma robótica realizará giros o se mantendrá en línea recta según sea necesario. Esto se representa en la **Figura 10**.

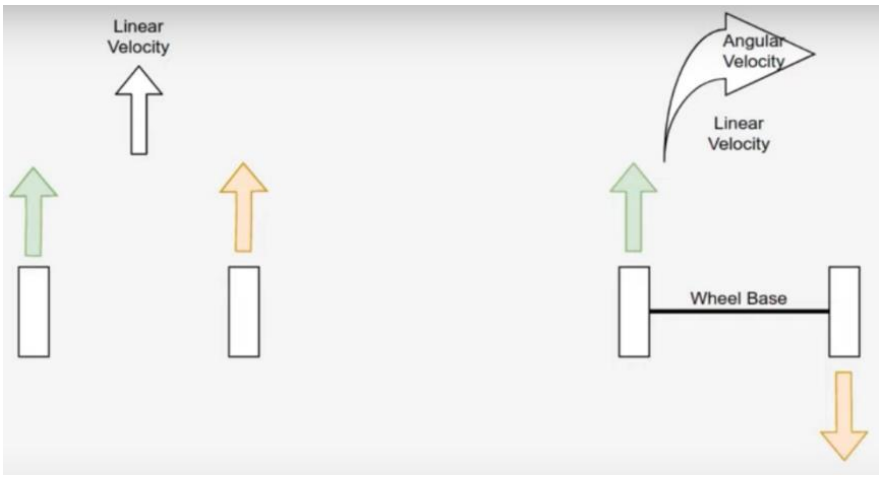


Figura 10: Representación de las velocidades lineal y angular
Fuente propia

La navegación autónoma en ROS, plantea un requisito hardware, que es la inclusión de un sensor láser plano montado en la plataforma móvil, esto en primera instancia representa un problema ya que no se cuenta con uno. No obstante, existe la posibilidad de convertir la información de la nube de puntos a un escáner láser 2D virtual, que se representa como una línea roja formada por puntos que enmarca los objetos, como se muestra en la **Figura 11**.

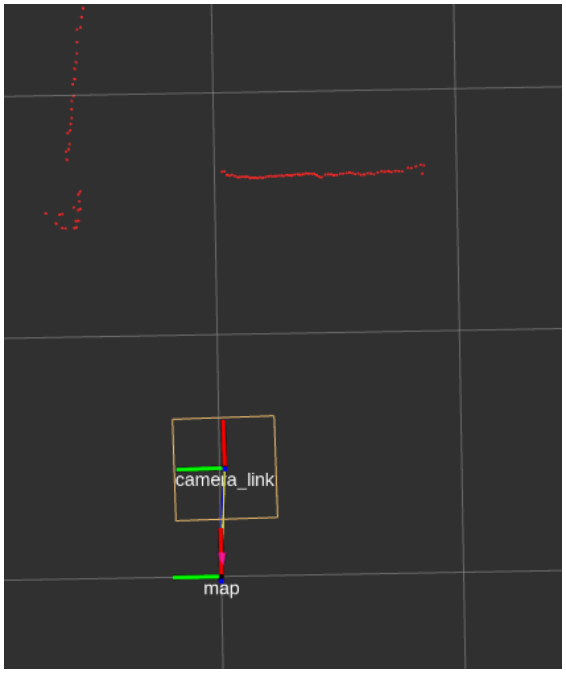


Figura 11: Representación de escáner láser virtual
Fuente propia

Por otro lado, otro de los requisitos necesarios para realizar la navegación autónoma es guardar el mapa que se generó en el apartado anterior, esto con el fin de que el método de navegación cuente con el mapa del entorno seleccionado y no deba generar uno de cero, esto se logra gracias a una herramienta que permite guardar mapas generados dinámicamente, estos se guardan en dos archivos, el .YAML que contiene los metadatos del mapa y la imagen .PNG codifica los datos de ocupación, la evidencia del mapa en este formato se encuentra en la **Figura 12**.

Figura 12: Imagen del mapa en formato .png
Fuente propia

Luego de obtener los datos de odometría visual, láser virtual y el mapa guardado, se enviarán a la navegación autónoma como se muestra en la **Figura 13,** la cual compilará estas informaciones para mover la plataforma móvil robótica a un punto deseado del mapa, esto lo logra generando una trayectoria que será la ruta por seguir y una dirección que le indicará a la plataforma hacia dónde debe girar para cumplir la trayectoria inicial.

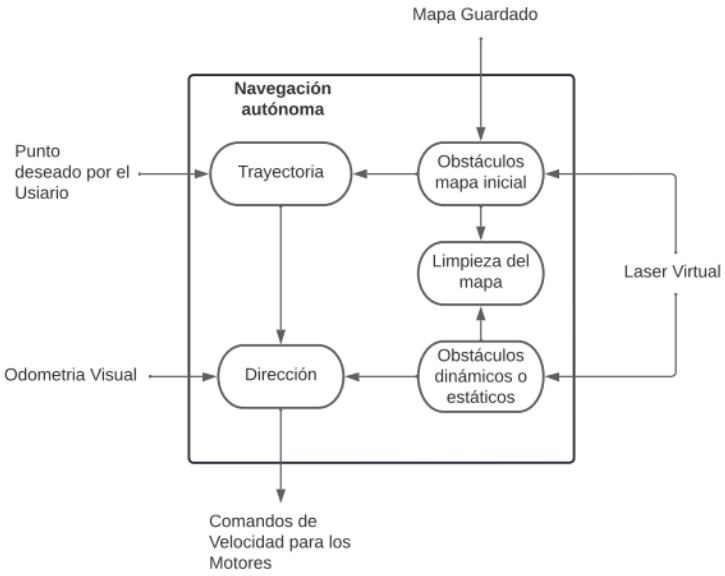
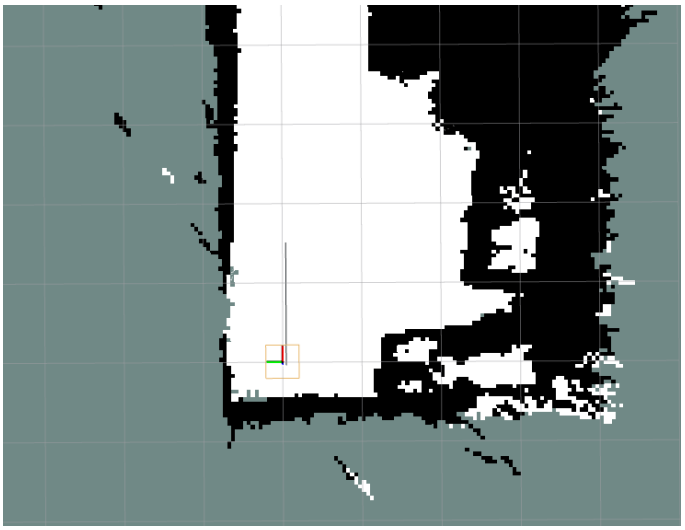
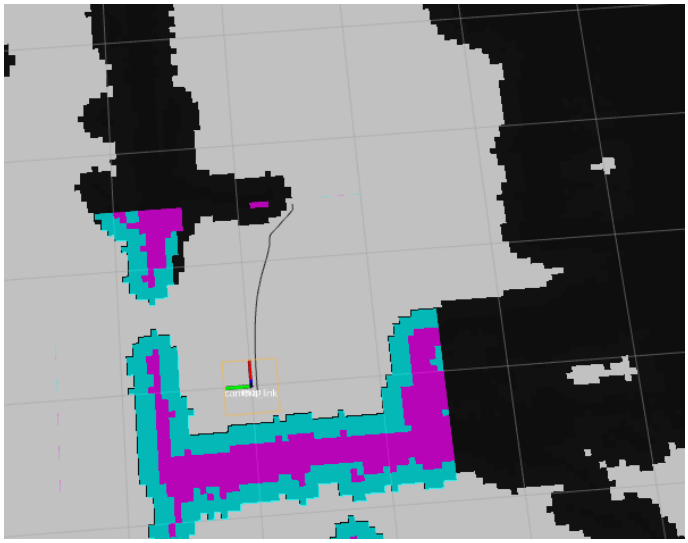


Figura 13: Diagrama de flujo de Navegación Autónoma
Fuente propia

La trayectoria es la encargada de generar la ruta al punto deseado por el usuario, evadiendo los obstáculos que aparecen en el mapa guardado. No obstante, puede modificar la ruta si la plataforma robótica se dirige a un obstáculo detectado por medio del láser virtual, esto se puede ver en la **Figura 14(a)** donde la primera trayectoria se genera en línea recta, pero si se detecta un objeto que se representa con pixeles de color rosado, modificara la ruta inicial para evadir dicho objeto como se muestra en la **Figura 14(b).**



(a)



(b)

Figura 14: Comparación de trayectoria con y sin obstáculos
Fuente propia

La dirección es la encargada de generar los comandos de velocidad que la plataforma robótica deberá usar para seguir la trayectoria generada, dichos comandos se encargarán de comunicarle a la plataforma robótica qué velocidad debe ejercer cada motor para girar en cualquier ángulo o seguir en línea recta y de esta forma lograr seguir la trayectoria dada por el usuario, lo cual se puede observar en la **Figura 15**, donde la línea de color rojo inicialmente representa que la plataforma móvil debe ir en línea recta y luego debe girar a la izquierda.

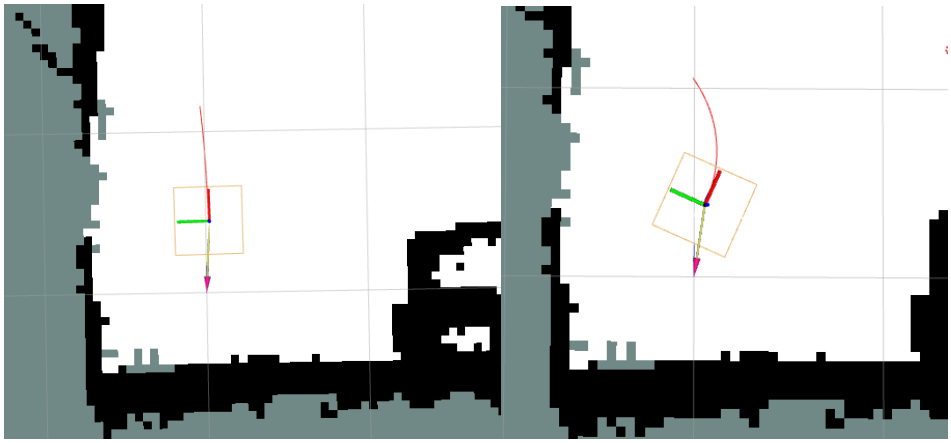


Figura 15: Representación de la estimación de movimiento
Fuente propia

Para que la trayectoria y la dirección funcionen correctamente, es necesario implementar un mapa de costos 2D (costmap)**,** el cual consiste en tomar datos del sensor láser virtual y crear una cuadrícula de ocupación 2D de color rosa en el mapa, representando los objetos

detectados, esto con el fin de que la plataforma no se acerque a dichos obstáculos ya sean estáticos o dinámicos, de igual manera, el mapa de costos se aplica al mapa inicial generando un margen que indicará un límite máximo que la plataforma tendrá para acercarse a los objetos detectados en la creación del mapa, esto se puede evidenciar en la **Figura 16**, donde los objetos detectados por el láser virtual se representan de color rosado y los objetos del mapa guardado se representan de color azul.

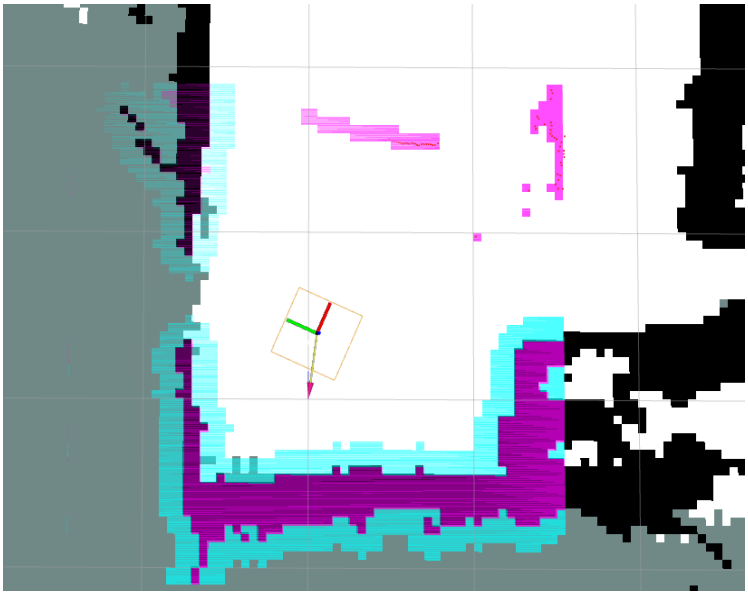
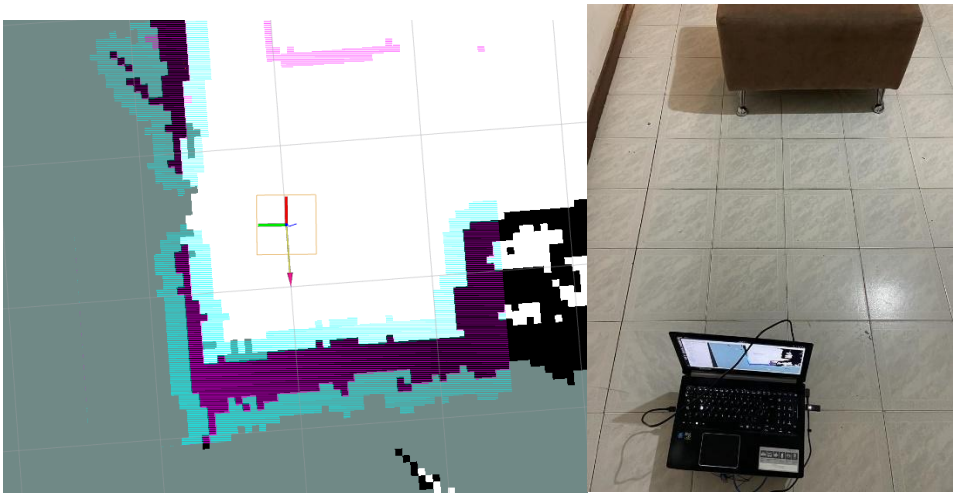


Figura 16: Detección de obstáculos
Fuente propia

Para completar la navegación, se necesita limpiar el mapa de costos de los obstáculos detectados, ya que de no hacerlo, quedarían enmarcados por todo el mapa así ya no se encuentren en esa posición, esto se logra liberando espacio, revirtiendo los mapas de costos usados y de esta forma retomando el mapa inicialmente guardado, en la **Figura 17** se muestra la comparación de la detección y limpieza de un obstáculo dinámico.



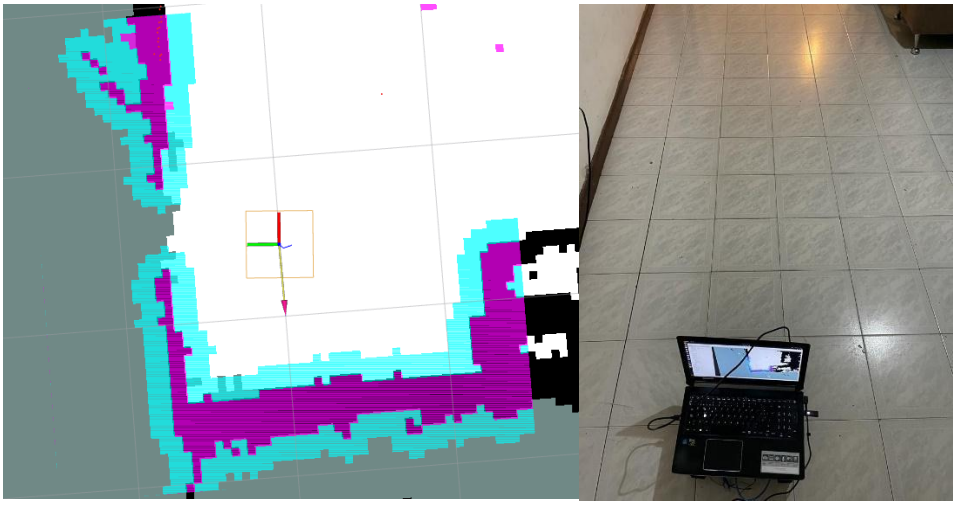


Figura 17: Generación y eliminación de obstáculos.
Fuente propia

Por último, es necesario enviar los comandos de velocidad angular y lineal al controlador, que en este caso es el Arduino Mega 2560 y al Motor Shield L298P encargado de enviarle los comandos de giro a los motores y valores de velocidad PWM.

6.4.3. Creación del objetivo de la navegación

Para el cumplimiento del segundo objetivo es de vital importancia realizar la detección de la persona, en este caso haciendo uso de herramientas de segmentación, luego se genera la conexión entre el sistema operativo ROS y el lenguaje de programación que utiliza el algoritmo de detección.

6.4.3.1 Conexión ROS con Python

La conexión de ROS con Python es necesaria debido a que el algoritmo de detección del sujeto se hará en este lenguaje, por lo que se verá reflejado en el sistema en forma de nodo, para este caso en específico, se hace uso de un nodo con la capacidad de recibir y enviar información de ROS, debido a que para el algoritmo es necesario tener como entradas las imágenes RGB y de profundidad que brinda el sensor y como salida se tiene las coordenadas que se dirigen a la trayectoria de la navegación autónoma, como se indica en la **Figura 18**.

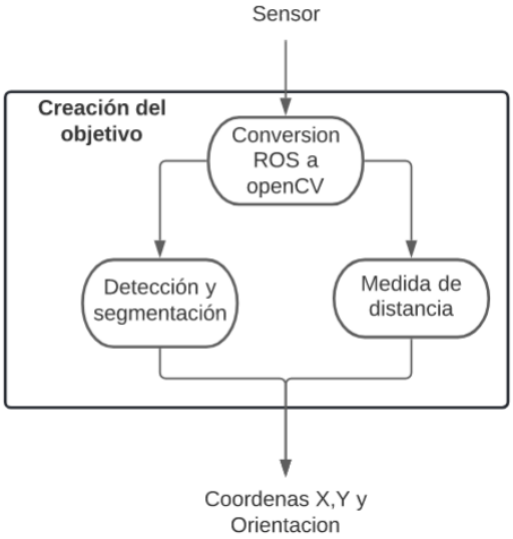


Figura 18: Diagrama de flujo de Creación del objetivo
Fuente propia

6.4.3.2. Creación de nodos ROS con Python

Existen dos tipos de nodos en ROS, los publicadores son aquellos que realizan un proceso y envían información a otro nodo, los subscriptores son los que reciben información y realizan un proceso[99], aunque los nodos se dividan en estos dos tipos, también existe la posibilidad de crear uno que cumpla con las dos funciones al mismo tiempo haciendo uso de algunas funciones que hacen parte de ROS.

6.4.3.3. Conversión de imagen ROS a imagen OpenCV

Las imágenes RGB y de profundidad generadas por el sensor inicialmente se encuentran en el nodo de ROS, debido a que estas imágenes son necesarias para la detección y seguimiento de la persona se deben enviar a Python por medio de un nodo suscriptor que recibe las dos imágenes, una vez hecho esto, se debe hacer una conversión del tipo de imagen ya que las imágenes de ROS no son compatibles con OpenCV, que es el módulo de visión por computador que se utiliza en Python, para llevar a cabo este cambio se hizo uso de una herramienta de ROS cuyo principal objetivo es realizar esta labor, una vez se haga está conversión se pueden operar las imágenes por medio del módulo OpenCV.

6.4.3.4. Segmentación de sujeto con Python.

Para la detección de la persona se utilizó el paquete mediapipe, específicamente su distribución mediapipe selfie segmentation, la cual tiene la capacidad de segmentar personas de una imagen, es decir, la salida de este proceso es una imagen en blanco y negro la cual separa en blanco el contorno de la persona y el fondo en negro[100] como se muestra en **Figura 19**, una vez detectado el sujeto, se genera un punto de interés en el interior de su contorno, el cual será utilizado para la generación de la trayectoria en el eje Y del mapa, este proceso se hace mediante condicionales de posición actual y posición anterior definiendo así la dirección hacia donde se dirige el sujeto.

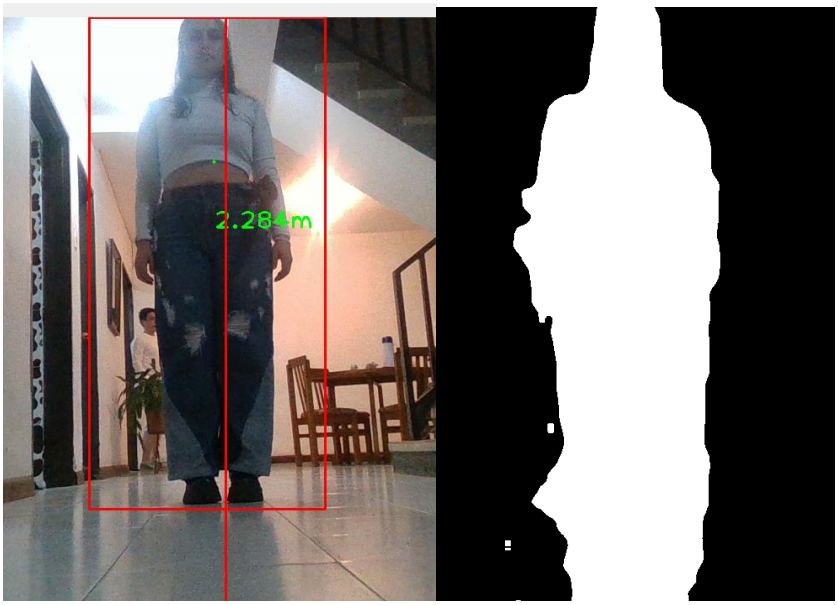


Figura 19: Imagen RGB y binaria del sujeto
Fuente propia

6.4.3.5. Detección de distancia

Para la detección de la distancia, se hizo uso de la función que brinda el nodo del sensor para esta tarea, al cual simplemente se le envían las coordenadas y este directamente brinda la distancia desde el sensor al punto seleccionado, las coordenadas que se utilizan para el proceso son las del punto de interés, asegurando así que el valor de distancia sea siempre tomado con respecto al sujeto, este valor de distancia se retorna en milímetros, por lo que solo basta con hacer la conversión a metros para definir la coordenada en el eje X del mapa.

Una vez que se tiene definido el valor de las coordenadas en los ejes X y Y, se procede a realizar el envío de estos datos a la navegación autónoma haciendo uso del nodo publicador

creado con anterioridad, además, con ayuda de la coordenada en el eje Y se puede definir la orientación objetivo, estos datos también se envían por el mismo nodo, los datos de orientación se consideran debido a que con esto se sabe en qué dirección debe llegar la plataforma robótica al objetivo, con el fin de evitar que el sujeto se deje de detectar dentro del frame a causa del movimiento.

6.5. Control de seguimiento del sujeto.

Arduino se encargará de procesar los comandos de velocidad enviados por la velocidad autónoma, los cuales son velocidad angular y lineal, donde se ocupará la velocidad lineal en X y la angular en Z, estas nos otorgaran los valores PWM que se le envían al Motor Shield y del mismo modo proporcionan la dirección de giro de los motores, debido al código implementado y al Motor Shield es necesario utilizar pines de frenado para los motores, ya que este no cuenta con comandos comunes para esta tarea.

6.6. Protocolo de pruebas.

El protocolo de pruebas se diseñó para demostrar el funcionamiento del algoritmo propuesto en este documento, se realizarán pruebas tanto de SLAM como del seguimiento de una persona en un entorno indoor dinámico, las pruebas se enfocaron en el seguimiento de la plataforma móvil a la persona y el evadir los obstáculos que se le presenten.

6.6.1. Entorno de pruebas.

Para el entorno de pruebas se utilizó un pasillo ubicado dentro de una casa el cual cuenta con unas dimensiones aproximadas de 1.7m de ancho y 9.6m de largo, con una zona un poco más amplia como se puede evidenciar en la **Figura 20**, se considera que la persona debe estar mínimo de 2 metros de distancia del sensor para evitar fallos en la detección, para esta implementación se considera controlar el entorno de tal forma que solo una persona está en él al momento de las pruebas.

Figura 20: Entorno de pruebas
Fuente propia

6.6.2. Sujetos de prueba.

Como sujetos de prueba se contó con la participación de tres personas todas mayores de edad al momento de realizar las pruebas, es importante considerar que los sujetos cumplen con una velocidad de marcha para las pruebas, la información específica de estos se encuentra en la **Tabla 7**.

Tabla 7: *Sujetos de prueba*

Sujeto	Sexo	Edad	Altura
1	Masculino	23 años	180 cm
2	Masculino	22 años	185 cm
3	Femenino	25 años	153 cm

6.6.3. Tipos de pruebas.

Para las pruebas de verificación del funcionamiento se tuvieron en cuenta cuatro actividades a realizar, para estas es necesario que el robot siempre inicie en el punto de origen determinado al momento de realizar el mapa sobre el que se trabaja y además el seguimiento al sujeto se realiza en todas las actividades cuando la separación de este con la

plataforma sea mayor a 2 metros, cada prueba tendrá un video de confirmación el cual cumple con una nomenclatura para mejor comprensión y se encuentran en el anexo pruebas.

- Actividad 1: Seguimiento en línea recta sin obstáculos

Para esta prueba el sujeto camina en línea recta frente a la plataforma y debe realizar dos pausas, a 3.3m y 5.7m aproximadamente del punto inicial del robot, para cada pausa la plataforma debe cumplir con la condición de detenerse en una distancia aproximada de 2 +/- 0.2 metros del sujeto y debe volver a moverse cuando el sujeto siga hacia adelante, el entorno que se utiliza para esta actividad se puede encontrar en ***Figura 21*****(a).**

- Actividad 2: Seguimiento en línea recta con obstáculos estáticos

Para esta prueba el sujeto camina en línea recta frente a la plataforma sin realizar pausas en el trayecto, en este caso el sistema debe generar una trayectoria la cual permita que la plataforma evada el obstáculo permitiendo continuar con el seguimiento al sujeto evitando choques, el entorno de prueba para esta actividad se encuentran en ***Figura 21*****(b)**, así mismo los objetos utilizados se encuentran especificados en **Figura *22***.

- Actividad 3: Seguimiento en línea recta con obstáculos dinámicos

Para está prueba el sujeto se mueve en línea recta hacia adelante, el sistema debe generar una trayectoria hacia el sujeto, los objetos dinámicos que se presenten en el ambiente pueden ser objetos que mueve el mismo sujeto o algún factor externo, estos objetos deben ser identificados por el sistema generando un mapa de costos local representado como píxeles de color rosa en el mapa y así mismo se deben evadir por parte de la plataforma robótica, para esta prueba se utilizó el entorno representado en ***Figura 21*****(a)** y los objetos dinámicos se encuentran especificados en **Figura *22***.

- Actividad 4: Seguimiento en zigzag sin obstáculos

Para está prueba, el sujeto realiza una marcha en zig zag por el entorno sin realizar pausas hasta un punto final, el sistema en este caso debe generar una trayectoria y seguir el sujeto hacia la dirección donde se encuentre, el entorno de prueba para esta actividad se encuentra especificado en ***Figura 21*****(a)**.

(a) **(b)**

Figura 21: Entorno de prueba sin y con obstáculos
Fuente propia

Figura 22: Objetos utilizados como obstáculos
Fuente propia

6.6.4. Nomenclatura.

Los videos capturados como evidencia de las pruebas realizadas se guardan en carpetas buscando generar un orden, inicialmente se encuentran tres carpetas, estas generadas por el número de sujetos, dentro de cada una de estas se encuentran cuatro carpetas las cuales hacen referencia a las actividades, es allí donde se encontrarán los videos que siguen la nomenclatura S#A#V#, siendo:

- S: El sujeto que realiza la actividad.

- A: La actividad realizada.
- V: El número de video realizado.

Finalmente se obtiene un total de 120 videos, los cuales resultan de 10 videos por actividad con cada sujeto.

6.6.5. Velocidad de marcha.

Para el movimiento de la persona se hace uso de un metrónomo digital, este dispositivo genera un sonido medido en pulsaciones por minuto ppm, cada vez que se genera el sonido el sujeto avanza constantemente una distancia de 30 centímetros, es así que sin tener en cuenta el error humano la velocidad de marcha es aproximadamente 0.175 m/s, en este caso haciendo uso de 35 ppm, asumiendo que se cumplen con las características anteriores, cada minuto el sujeto daría 35 pasos de 30 cm cada uno, es así que para el cálculo de la velocidad se utilizó la lógica representada en (1).

$$35\ pasos\ * 30\ cm = 1050\ cm \tag{1}$$

$$1050 \frac{cm}{min} * \frac{1\ m}{100\ cm} * \frac{1\ min}{60\ seg} = 0.175\ \frac{m}{seg}$$

CAPITULO 7: RESULTADOS

Luego de realizar los vídeos considerando cada sujeto y cada actividad, se organizó la información haciendo uso de tablas las cuales demuestran los porcentajes de funcionamiento del sistema frente a las diferentes pruebas realizadas, para la evaluación de las pruebas se tuvo en cuenta si se cumple o no con el objetivo de la actividad.

A medida que se realizaban las pruebas, se guardaron los resultados de las actividades en una tabla general, para la primera actividad, las pruebas calificadas como verdaderas son aquellas en las que la persona era detectada y mientras la distancia fuera mayor a dos metros la plataforma generaba el movimiento necesario para el seguimiento al sujeto, mientras que las pruebas calificadas como falsas fueron aquellas en las que por alguna razón el sistema pierde información de localización o detección de la persona, generando una creación errónea de la trayectoria, los datos obtenidos para esta actividad se encuentran en la **Tabla *8***.

La primera actividad se realizó en el horario de la mañana para el sujeto 3 y en la noche para los sujetos 1 y 2, obteniendo como resultado que el sistema tiene una efectividad promedio del 90% para esta, definiendo efectividad como el número de pruebas exitosas divido entre el total de pruebas realizadas.

Tabla 8: *Resultados de seguimiento en línea recta sin obstáculos*

Sujeto	Total verdaderas	Total falsas	Total pruebas	Efectividad
1	8	2	10	80 %
2	9	1	10	90%
3	10	0	10	100%

Las pruebas de la segunda actividad se hicieron en la mañana, tarde y noche para los sujetos 3, 1 y 2 respectivamente, en este caso las pruebas calificadas como positivas fueron aquellas donde se realizó la detección y evasión de los obstáculos estáticos mientras se realizaba el seguimiento al sujeto de prueba, por su parte las pruebas calificadas como falsas fueron aquellas donde ocurrió algún evento que no permitió cumplir satisfactoriamente con el seguimiento del sujeto o la evasión del obstáculo, los datos obtenidos para esta actividad se encuentran en la **Tabla *9***, en este caso se obtuvo una efectividad promedio del 90% por parte del sistema para ejecutar la actividad.

Tabla 9: *Resultados de seguimiento en línea recta con obstáculos estáticos*

Sujeto	Total verdaderas	Total falsas	Total pruebas	Efectividad
1	9	1	10	90%
2	8	2	10	80%
3	10	1	10	100%

La tercera actividad se realizó en el horario de la noche para los sujetos 1 y 2 y en la tarde para el sujeto 3, en esta actividad se catalogaron como pruebas verdaderas aquellas en las que mientras se detectaba y se seguía el sujeto también se detectaba el objeto dinámico mientras este estaba en movimiento y se evadía para cumplir con la tarea de seguimiento al sujeto, por su parte las pruebas falsas fueron aquellas en las que se presentaron acciones que no permitieron cumplir con el objetivo de la actividad, los resultados de esta se encuentran en la **Tabla *10***, en esta actividad el sistema tuvo una efectividad promedio del 86,67%

Tabla 10: *Resultados de seguimiento en línea recta con obstáculos dinámicos*

Sujeto	Total verdaderas	Total falsas	Total pruebas	Efectividad
1	9	1	10	90%
2	8	2	10	80%
3	9	1	10	90%

La cuarta y última actividad se realizó en la noche para los sujetos 1 y 2 y en la mañana para el sujeto 3, para este caso las pruebas calificadas como verdaderas fueron aquellas que generaron trayectorias de movimiento hacia el sujeto mientras éste realizaba una marcha en zigzag por el entorno de pruebas, las pruebas falsas fueron aquellas que presentaron fallas ya sea en la detección, seguimiento del sujeto o pérdida de información de localización, los datos obtenidos de las pruebas de esta actividad se encuentran en la

Tabla *11*, en este caso el sistema tuvo una efectividad de 80%.

Tabla 11: *Resultados de seguimiento en zigzag sin obstáculos*

Sujeto	Total verdaderas	Total falsas	Total pruebas	Efectividad
1	8	2	10	80%
2	8	2	10	80%
3	8	2	10	80%

Finalmente, una vez realizadas y analizadas todas las pruebas se puede evidenciar que el sistema tiene una efectividad de 86,66% en su funcionamiento general, dentro de esta evaluación se incluye el funcionamiento de:

- Detección del sujeto.
- Seguimiento del sujeto.
- Evasión de obstáculos.
- Control de movimiento.
- Desempeño de SLAM.

Por su parte, de los errores encontrados en las pruebas se encontró que la pérdida de la detección del sujeto y la pérdida de información de localización (37.5% y 43.7% respectivamente) son la principal dificultad al que se debe enfrentar el sistema y algoritmos propuestos en el documento puesto que conlleva a una errónea generación de la trayectoria de movimiento.

CAPITULO 8: DISCUSIÓN, CONCLUSIONES Y TRABAJOS FUTUROS

Con la técnica SLAM propuesta en este trabajo, se pudo identificar que el mapeo y la navegación se encuentran relacionados, ya que para realizar una correcta navegación autónoma es necesario conocer el lugar donde se encuentra y también tener la capacidad de detectar obstáculos, en este proyecto se planteó el uso de procesamiento de imágenes para detectar y segmentar a una persona adulta con el fin de convertirla en el objetivo de la navegación, como principal aporte se considera el hecho de que el sistema es robusto con una efectividad promedio del 86.66% mientras se realizan las tareas de navegación, localización, detección de obstáculos, segmentación y detección de personas, de manera simultánea con el uso de un único sensor, con el fin de seguir y monitorear las trayectorias de los sujetos en tiempo real, ejecutando control sobre el movimiento de una plataforma robótica móvil de creación propia.

Por otro lado, en las pruebas realizadas también se detectaron algunas limitaciones al proyecto entre las que se incluye: pérdida de odometría visual si se sobrepasa el límite mínimo de detección de profundidad especificado para el sensor, también se presenta problemas al momento de realizar el mapa inicial por la inclinación del sensor, ya que si este no se encuentra en paralelo con el suelo genera el mapa con una pendiente, lo cual hace que a medida que avanza la plataforma se va dirigiendo hacia abajo con relación a la base inicial del mapa, este proceso también se ve limitado en entornos donde se genera brillo sobre el piso debido a la luz, ya sea natural o artificial. Por su parte, la segmentación y detección de la persona se ve limitada cuando se disminuye la separación de la persona y la plataforma a distancias menores a dos metros y también cuando se opera en entornos con baja iluminación, llevando el sistema a establecer trayectorias erróneas.

Una vez dadas estas fortalezas y limitaciones del sistema, se comparó con algunas de las propuestas evaluadas en la revisión bibliográfica, encontrando ciertas similitudes y diferencias en algunos aspectos de la creación y del funcionamiento, como principal diferencia se encontró el uso de sensores LiDAR, IMU y láser[25][26] incluso generando sistemas multisensoriales, lo cual puede llevar a generar sistemas más robustos en temas como el mapeo y localización, por su parte, para la tarea de detección de la persona se encontró que en diversos trabajos [28],[30],[32] se hace uso de datasets ya establecidos para hacer más efectiva esta tarea. Entre las similitudes más considerables se encuentra que cuando se hace uso de cámaras RGB-D se necesita hacer uso de velocidades bajas para que así los cambios de frame a frame sean pequeños[30], teniendo en cuenta estas comparaciones se plantea que para la mejora de este proyecto en un futuro se haga uso de un sensor LiDAR con rango de uso de 360° buscando mejorar la eficiencia del mapeo, en cuanto a la segmentación y

detección de la persona se recomienda mejorar el algoritmo para poder reducir la distancia mínima, logrando que el sistema funcione en ambientes pequeños, también se plantea el uso de un sistema multisensorial en el cual las tareas se distribuyan entre diferentes sensores permitiendo ejecutarlas de una mejor manera, en cuanto a la plataforma robótica se recomienda incluir otro sistema de locomoción que permita tener un buen rendimiento en la movilidad en cualquier terreno indoor dinámico.

CAPITULO 9: ANEXOS

Los anexos para este trabajo de investigación se encuentran almacenados en una carpeta de drive, compartida con coordinación de la facultad de ingenieria electrónica por medio del correo institucional, en caso de ser requerido, se concederá el permiso en el menor tiempo posible a otros usuarios.

Dentro de la carpeta se encuentran los videos de las pruebas, manual del prototipo robótico, manual de usuario y la carpeta de desarrollo donde se debe hacer la instalación de los paquetes utilizados.

https://drive.google.com/drive/folders/1i8hPr9BtyQA02YgM4XMZqKyE4srz5TaR?usp=sharing

CAPITULO 10: REFERENCIAS

[1] T. Monge Acuña and Y. Solis Jiménez, “El Síndrome de Caídas en Personas Adultos Mayores y su Velocidad de Marcha,” *Revista Medica de Costa Rica y Centroamericana*, no. 618, pp. 91–95, 2016.

[2] J. Quiñonez Torres, “Riesgo de caídas en los pacientes adultos mayores del Hospital Geriátrico de la Policía San José , 2016,” Universidad nacional mayor de San Marcos, 2017.

[3] M. G. Riaño Castañeda, J. Moreno Gómez, L. S. Echeverria Avellaneda, L. G. Rangel Caballero, and J. C. Sánchez Delgado, “Condición física funcional y riesgo de caídas en adultos mayores,” *Revista cubana de investigaciones biomédicas*, vol. 37, La Habana, p. 10, 2018. doi: ISSN 0864-0300.

[4] L. Álvarez Rodriguez, “Síndrome de caídas en el adulto mayor,” *Revista medica de Costa Rica y centroamerica*, no. 617, pp. 807–810, 2015.

[5] A. Lavedán Santamaría, P. Jürschik Giménez, T. Botigué Satorra, C. Nuin Orrio, and M. Viladrosa Montoy, “Prevalencia y factores asociados a caídas en adultos mayores que viven en la comunidad,” *Elsevier Ltd*, vol. 47, no. 6, pp. 367–375, 2014, doi: 10.1016/j.aprim.2014.07.012.

[6] A. Smith, A. Oliveira Silva, R. Parlezani Rodrigues, M. Silva Paredes, J. de A. Nogueira, and L. Rangel Tura, “Evaluación de riesgo de caídas en adultos mayores que viven en el domicilio,” *Revista Latino-Americana de Enfermagem*, Ribeirão Preto, p. 9, 2017. doi: 10.1590/1518-8345.0671.2754.

[7] A. D. Santiago Mijangos, P. Gonzales de la cruz, L. I. Solís Alfaro, and T. Santiago Ribón, “Factores de riesgo de caídas e índice de masa corporal en el adulto mayor hospitalizado,” *Revista cuidarte*, vol. 10, no. 1, Veracruz, p. 9, Dec. 2019. doi: https://doi.org/10.15649/cuidarte.v10i1.621.

[8] G. J. Rivera Pacheco, “Incidencia de pacientes adultos mayores con riesgo de caída que acuden al servicio de Terapia Física del Centro Médico Naval Cirujano Mayor Santiago Távara,” Universidad nacional mayor de San Marcos, 2018.

[9] E. F. Pérez Reyes and I. W. Vargas Quinzo, “Desarrollo de un robot social en el área de la telemedicina para el monitoreo remoto y diagnóstico de enfermedades cardiovasculares en el adulto mayor,” Universidad nacional de Chimborazo, 2017.

[10] G. Canal Camprodon, “Adapting robot behavior to user preferences in assistive scenarios,” Universidad Politécnica de Catalunya, 2020.

[11] M. Zafra Granados, J. Martínez Rodríguez, and J. Morales Rodríguez, “Navegación autónoma de un robot móvil 4*4 en entornos naturales mediante GPS diferencial y telémetro láser tridimensional,” *uma.es*, vol. 0, Málaga, p. 15, Sep. 2015.

[12] G. Ferrer Mínguez, “Social robot navigation in urban dynamic environments,” Universidad politécnica de Catalunya, 2015.

[13] P. Trullos Pastor, “Navegación autónoma en entornos de interior basada en mapas topológicos visuales con técnicas de transformaciones reductoras (PCA y LDA),” E.T.S de Ingenieros Informaticos, 2018.

[14] P. Encalada, C. Gordón, H. Lema, D. León, and D. Chicaiza, “Navegación autónoma del robot KUKA Youbot con señalización basada en tácnicas de detección de objetos y redes neuronales profundas,” *RISTI - Revista Iberica de Sistemas e Tecnologias de Informacao*, vol. 0, no. 19, pp. 305–316, 2019.

[15] L. F. Ortiz Arroyave, “Metodología de Navegación de robots móviles para detección de vanos y discontinuidades en su trayectoria,” Instituto Tecnológico Metropolitano, 2019.

[16] R. González, F. Rodríguez, and J. L. Guzman, “Robots móviles con orugas Historia, Modelado, Localización y Control,” *Revista iberoamericana de automática e informática industrial*, vol. 12, Almería, pp. 3–12, 2015. doi: 10.1016/j.riai.2014.11.001.

[17] D. L. Ramírez Bedoya, “Diseño de una estrategia para la planeación de rutas de navegación autónoma de un robot móvil en entornos interiores usando un algoritmo de apredizaje automático,” Universidad Nacional de Colombia, 2020.

[18] J. Boal, Á. Sánchez-Miralles, and Á. Arranz, “Topological simultaneous localization and mapping: A survey,” *Cambridge University Press*, vol. 32, no. 5, Madrid, pp. 803–821, Oct. 29, 2014. doi: 10.1017/S0263574713001070.

[19] R. C. Smith and P. Cheeseman, “On the Representation and Estimation of Spatial Uncertainty,” *Int J Rob Res*, vol. 5, no. 4, pp. 56–68, 1986, doi: 10.1177/027836498600500404.

[20] L. Garzón Obregón, L. Forero Rincón, and O. Duque Suárez, “Diseño e implementación de un sistema de visión artificial usando una técnica de mapeo y localización simultánea (SLAM) sobre una plataforma robótica móvil,” *Mundo Fesc*, vol. 8, no. 16, Pamplona, pp. 8–17, Jul. 2018.

[21] S. Chen, B. Zhou, C. Jiang, W. Xue, and Q. Li, “A lidar/visual slam backend with loop closure detection and graph optimization,” *Remote Sens (Basel)*, vol. 13, no. 14, Jul. 2021, doi: 10.3390/rs13142720.

[22] A. A. Housein and G. Xingyu, “Simultaneous Localization and Mapping using differential drive mobile robot under ROS,” *J Phys Conf Ser*, vol. 1820, no. 1, 2021, doi: 10.1088/1742-6596/1820/1/012015.

[23] A. I. Martyshkin, “Motion Planning Algorithm for a Mobile Robot with a Smart Machine Vision System,” *Nexo Revista Científica*, vol. 33, no. 02, pp. 651–671, 2020, doi: 10.5377/nexo.v33i02.10800.

[24] D. Esparza, “STEREO VISION BASED SLAM IN DYNAMIC OUTDOOR ENVIRONMENTS USING DEEP LEARNING Student,” Centro de investigaciones en óptica, 2019.

[25] M. S. Bahraini, M. Bozorg, and A. B. Rad, “SLAM in dynamic environments via ML-RANSAC,” *Mechatronics*, vol. 49, pp. 105–118, 2019, doi: 10.1016/j.mechatronics.2017.12.002.

[26] F. Demin, A. Nomra, A. Boucheloukh, E. Kobzili, M. Hamerlain, and A. Bazoula, “SLAM based on adaptative SVSF for cooperative inmanned vehicles in dynamic environment,” *IFAC-PapersOnline*, vol. 52, no. 8, pp. 73–80, 2019.

[27] J. Zhang, M. Henein, R. Mahony, and V. Ila, “VDO-SLAM: A Visual Dynamic Object-aware SLAM System,” *Computer Science*, vol. 3, no. 1, May 2020, [Online]. Available: http://arxiv.org/abs/2005.11052

[28] A. Gomez-Olmedo, “Comparativa de estrategias de SLAM visual en sistemas empotrados de bajo coste,” Universidad de Alcalá, 2021.

[29] Y. Sun, M. Liu, and M. Q. Meng, “Motion removal for reliable RGB-D SLAM in dynamic environments,” *Rob Auton Syst*, vol. 108, pp. 115–127, 2018, doi: 10.1016/j.robot.2018.07.002.

[30] Á. Araneda Morales, “Simulación y análisis de algoritmos de planificación y rutas para robot móvil en Gazebo,” Universidad Andres Bello, 2020.

[31] L. Xiao, J. Wang, X. Qiu, Z. Rong, and X. Zou, “Dynamic-SLAM: Semantic monocular visual localization and mapping based on deep learning in dynamic environment,” *Rob Auton Syst*, vol. 117, pp. 1–16, 2019, doi: 10.1016/j.robot.2019.03.012.

[32] T. Solano. J. Domingo, “Real-Time Accurate Visual SLAM with Place Recognition,” Universidad de Zaragoza, 2017.

[33] J. Cremona, L. Uzal, and T. Pire, “WGANVO: Odometría visual monocular basada en redes adversarias generativas,” *Revista iberoamericana de Automática e informática industrial*, vol. 78, no. May, p. 10, 2021, doi: https://doi.org/10.4995/riai.2022.16113.

[34] A. A. Díaz Toro, “Real Time Dense Tracking and Mapping Using a Monocular Camera,” Universidad del Valle, 2018.

[35] R. Domínguez Pérez, “Estudio de Técnicas de Odometría Visual Monocular,” Universidad de Sevilla, 2019.

[36] Á. M. Lema Fulgencio, “Sistema de odometría visual para localización monocular,” Universidad de Sevilla, 2019.

[37] T. Zhang and Y. Nakamura, “PoseFusion : Dense RGB-D SLAM in Dynamic Human Environments,” *Proceedings of the 2018 international sysmposium on experimental robotics*, vol. 11, pp. 772–780, 2018.

[38] J. C. Núñez Moreno, “Análisis de movimiento humano mediante dispositivos RGB-D,” Universidad Rey Juan Carlos, 2019.

[39] M. Longrinos Garrido, “Sistema de localización y mapeo simultáneo para interiores basado en imágenes de profundidad(RGB-D),” Benemérita universidad autónoma de Puebla, 2021.

[40] C. R. Valderico, “SLAM para robots domésticos,” Universidad de Alicante, 2021.

[41] J. Civera and S. H. Lee, “RGB-D Odometry and SLAM,” *Advances in computar vision and pattern recognition*, pp. 117–144, 2019, doi: 10.1007/978-3-030-28603-3_6.

[42] Y. Zou, A. Eldemiry, Y. Li, and W. Chen, “Robust RGB-D slam using point and line features for low textured scene,” *MDPI*, vol. 20, no. 17, pp. 1–20, 2020, doi: 10.3390/s20174984.

[43] C. Debeunne and D. Vivet, “A review of visual-lidar fusion based simultaneous localization and mapping,” *isprs*, vol. 10, no. 3, p. 20, 2021, doi: https://doi.org/10.3390/ijgi10030163.

[44] R. Wang, W. Wan, Y. Wang, and K. Di, “A new RGB-D SLAM method with moving object detection for dynamic indoor scenes,” *Remote Sens (Basel)*, vol. 11, no. 10, p. 19, 2019, doi: 10.3390/rs11101143.

[45] X. Meng, W. Gao, and Z. Hu, “Dense RGB-D SLAM with multiple cameras,” *MDPI*, vol. 18, no. 7, pp. 1–12, 2018, doi: 10.3390/s18072118.

[46] Z. Teed and J. Deng, “DROID-SLAM: Deep Visual SLAM for Monocular, Stereo, and RGB-D Cameras,” pp. 1–12, 2021.

[47] I. Oliva Leal, “Implementación de un algoritmo de visión artificial basado en ORB-SLAM para un helicóptero miniatura de cuatro rotores,” Universidad autónoma del estado de Hidalgo, 2020.

[48] J. Sánchez Galarraga, “Odometría visual estereoscópica fusionada con información inercial para la determinación de la posición de los UAV respecto a su entorno,” Universidad politécnica de Madrid, 2018.

[49] J. A. Salgado Luque, “Desplazamiento Seguro y Autónomo en Entornos Poco Estructurados Basado en Técnicas de SLAM para un Vehículo Aéreo no Tripulado Desplazamiento Seguro y Autónomo en Entornos Poco Estructurados Basado en Técnicas de SLAM para un Vehículo Aéreo no Tripulado,” Universidad militar nueva granada, 2018.

[50] M. J. Martos Aranzana, “Creación de modelos mediante técnicas de Graph-SLAM visual en robótica móvil usando técnicas de apariencia Global,” *Revista Doctorado UMH*, vol. 4, no. 1, p. 18, 2018, doi: 10.21134/doctumh.v4i1.1492.

[51] Y. Berenguer, L. Payá, M. Ballesta, L. Jiménez, S. Cebollada, and O. Reinoso, “algoritmo de SLAM utilizando apariencia global de imágenes omnidireccionales,” *Actas de las XXXVIII jornadas de automática*, vol. 1, pp. 22–29, 2017.

[52] E. Martín Arias, “Evaluación De Un Algoritmo De Localización Híbrida Para Un Robot Móvil Dotado De Láser Y Cámara Omnidireccional,” Universidad politecnica de Madrid, 2020.

[53] F. Pineda Torres, “Técnicas de slam con filtros probabilísticos ; caracterización y resultados en robots móviles,” *Mundo FESC*, vol. 9, no. 18, pp. 7–15, 2019.

[54] E. Aldas Serrano, “Plataforma Robótica Móvil Para Experimentos De Mapeo Y Localización Simultanea (SLAM) En Base a Sensores De Rango,” Escuela superior politécnica de Chimborazo, 2017.

[55] E. L. Álvarez-Gutiérrez and F. R. Jiménez-López, “Generación De Mapa Global 2D Y Slam Usando Lidar Y Una Estéreo Cámara Para El Seguimiento De Movimiento De Un Robot Móvil,” *Iteckne*, vol. 16, no. 2, pp. 144–156, 2019, doi: https://doi.org/10.15332/iteckne.v16i2.2357.

[56] V. Costella and H. Rodríguez, “Caracterización de un sensor de rango láser de bajo costo previo a una implementación de Slam,” *PRISMA tecnológico*, vol. 7, no. 1, pp. 30–34, 2016.

[57] C. Xiaxuan Lu *et al.*, “See through smoke: Robust indoor mapping with low-cost mmWave radar,” *MobiSys ’20: Actas de la 18ª Conferencia Internacional sobre Sistemas, Aplicaciones y Servicios Móviles*, vol. 0, no. 0, Toronto, pp. 14–27, 2020. doi: 10.1145/3386901.3388945.

[58] Y. Ge *et al.*, “A Computationally Efficient EK-PMBM Filter for Bistatic mmWave Radio SLAM,” *Electrical engineering ans system science*, pp. 1–30, Sep. 2021, [Online]. Available: http://arxiv.org/abs/2109.03561

[59] M. Yin *et al.*, “Millimeter Wave Wireless Assisted Robot Navigation with Link State Classification,” *EESS.SP*, vol. 2, pp. 1–13, Oct. 2021, [Online]. Available: http://arxiv.org/abs/2110.14789

[60] Y. Li, Y. Liu, Y. Wang, Y. Lin, and W. Shen, “The millimeter-wave radar slam assisted by the RCS feature of the target and IMU,” *Sensors* , vol. 20, no. 18, pp. 1–27, Sep. 2020, doi: 10.3390/s20185421.

[61] J. Palacios, G. Bielsa, P. Casari, and J. Widmer, “Communication-Driven Localization and Mapping for Millimeter Wave Networks,” *IEEE INFOCOM*, vol. 1, pp. 1–9, Apr. 2018, Accessed: Jan. 30, 2022.

[62] I. Ullah, S. Qian, Z. Deng, and J. H. Lee, “Extended Kalman Filter-based localization algorithm by edge computing in Wireless Sensor Networks,” *Digital Communications and Networks*, vol. 7, no. 2, pp. 187–195, May 2021, doi: 10.1016/j.dcan.2020.08.002.

[63] Z. L. Ren, L. G. Wang, and L. Bi, “Improved Extended Kalman Filter Based on Fuzzy Adaptation for SLAM in Underground Tunnels,” *International Journal of Precision Engineering and Manufacturing*, vol. 20, no. 12, pp. 2119–2127, Sep. 2019, doi: 10.1007/s12541-019-00222-w.

[64] J. Vincent, M. Labbé, J.-S. Lauzon, F. Grondin, P.-M. Comtois-Rivet, and F. Michaud, “Dynamic Object Tracking and Masking for Visual SLAM,” *Computer vision and pattern recognition*, vol. 1, pp. 1–6, Jul. 2020.

[65] I. Ullah, X. Su, J. Zhu, X. Zhang, D. Choi, and Z. Hou, “Evaluation of localization by extended kalman filter, unscented kalman filter, and particle filter-based techniques,” *Wirel Commun Mob Comput*, vol. 1, pp. 1–15, Oct. 2020, doi: 10.1155/2020/8898672.

[66] J. Labora Gómez, “Comparative Analyses of Mobile Robots Localization Algorithms based on Particle Filters.”

[67] Y. Li *et al.*, “A SLAM with simultaneous construction of 2D and 3D maps based on Rao-Blackwekkized particle filters,” *2018 Tenth international conference on advance computational intelligence*, vol. 1, pp. 1–5, Mar. 2018, Accessed: Jan. 30, 2022.

[68] J. Luo and S. Qin, “A Fast Algorithm of SLAM Based on Combinatorial Interval Filters,” *IEEE Access*, vol. 6, pp. 28174–28192, May 2018, doi: 10.1109/ACCESS.2018.2838112.

[69] D. Talwar and S. Jung, “Particle Filter-based Localization of a Mobile Robot by Using a Single Lidar Sensor under SLAM in ROS Environment,” in *International Conference on Control, Automation and Systems*, Oct. 2019, vol. 2019-October, pp. 1112–1115.

[70] F. Matencio, “Diseño de un sistema robótico móvil para guiar e informar a pacientes en centros médicos,” Pregrado, Pontificia universidad católica del Perú, Lima, 2021. Accessed: Jan. 30, 2022.

[71] F. Salgado Castillo and A. Rodas Córdova, “Diseño e implementación de un sistema de generación de trayectoria para el control de un robot móvil, utilizando Inteligencia Artificial,” Pregrado, universidad de Azuay, Cuenca, 2021. Accessed: Jan. 25, 2022.

[72] J. D. Martin, K. Doherty, C. Cyr, B. Englot, and J. Leonard, “Variational Filtering with Copula Models for SLAM,” *IROS*, vol. 1, pp. 1–10, Aug. 2020.

[73] L. Urresti de las Alas-Pumariño, “Redes generativas para la construcción de imágenes submarinas,” Master, Universidad de Cantabria, Santander, 2020. Accessed: Jan. 22, 2022.

[74] D. Jiménez Jiménez, “Identificación Semántica de Objetos Mediante Deep Learning,” Pregrado, Universidad de Sevilla, Sevilla, 2018. Accessed: Jan. 25, 2022.

[75] J. Placed Perales, “Estrategias de Deep Learning en SLAM Activo,” Master, Universidad de Zaragoza, Zaragoza, 2019. Accessed: Jan. 23, 2022.

[76] J. Royo Miquel, “Robust Object Classification for Autonomous Ship Navigation through Spiking Neural Net-works and Meta Learning ShippingLab project,” Master, Universidad politécnica de Valencia, Lyngby, 2020. Accessed: Jan. 25, 2022.

[77] P. Encalada, C. Gordón, H. Lema, D. León, and D. Chicaiza, “Navegación autónoma del robot KUKA Youbot con señalización basada en técnicas de detección de objetos y redes neuronales profundas,” *RISTI - Revista Iberica de Sistemas e Tecnologias de Informacao*, vol. 0, no. 19, pp. 305–316, 2019.

[78] M. Díaz Saravia and J. Trejo Peraza, *Diseño se prototipo de vehículo autónomo utilizando redes neurales: aplicación para el transporte de materiales*, ITCA-FEPADE., vol. 1. Santa Tecla: Escuela de ingenieria eléctrica y electrónica, 2019. Accessed: Jan. 25, 2022.

[79] D. Escobar Montoya and J. Rico Romero, “Desarrollo de un sistema de percepción robótica para la localización y topológica en ambientes semiestructurados,” Pregrado, Universidad autónoma de occidente, Santiago de Cali, 2021. Accessed: Jan. 25, 2022.

[80] M. J. Martos Aranzana, “Creación de modelos mediante técnicas de Graph-SLAM visual en robótica móvil usando técnicas de apariencia Global,” *Revista Doctorado UMH*, vol. 4, no. 1, p. 18, 2018, doi: 10.21134/doctumh.v4i1.1492.

[81] J. Vallvé, J. Solà, and J. Andrade-Cetto, “Pose-graph SLAM sparsification using factor descent,” *Rob Auton Syst*, vol. 119, no. 1, pp. 108–118, Sep. 2019, Accessed: Jan. 30, 2022.

[82] S. Karam, V. Lehtola, and G. Vosselman, “Simple loop closing for continuous 6DOF LIDAR&IMU graph SLAM with planar features for indoor environments,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 181, pp. 413–426, Nov. 2021, doi: 10.1016/j.isprsjprs.2021.09.020.

[83] M. Pierzchała, P. Giguère, and R. Astrup, “Mapping forests using an unmanned ground vehicle with 3D LiDAR and graph-SLAM,” *Comput Electron Agric*, vol. 145, pp. 217–225, Feb. 2018, doi: 10.1016/j.compag.2017.12.034.

[84] W. Wen, L. T. Hsu, and G. Zhang, “Performance analysis of NDT-based graph SLAM for autonomous vehicle in diverse typical driving scenarios of Hong Kong,” *Sensors* , vol. 18, no. 11, Nov. 2018.

[85] S. Bhamidipati and G. X. Gao, “Multiple GPS Fault Detection and Isolation Using a Graph-SLAM Framework,” *Proceedings of the 31st international technical meeting of the satelite division of the institute of navigation*, pp. 2672–2681, Oct. 2018.

[86] K. Muñoz Peña and B. Bacca Cortes, “GUI3DXBot: An Interactive Software Tool for a Tour-Guide Mobile Robot,” *Ciencia e Ingeniería Neogranadina*, vol. 30, no. 1, pp. 59–74, Nov. 2019.

[87] A. Rahman, “Penerapan SLAM Gmapping dengan robot operating system Menggnakan laser scaner pada turtlebot,” *Rekayasa Elektrika*, vol. 16, no. 2, pp. 103–109, Jun. 2020, Accessed: Jan. 30, 2022.

[88] G. Tang, A. Shah, and K. P. Michmizos, “Spiking Neural Network on Neuromorphic Hardware for Energy-Efficient Unidimensional SLAM,” *IROS*, vol. 2, no. 1, pp. 1–6, Sep. 2019.

[89] J. F. Chow *et al.*, “Toward Underground Localization: Lidar Inertial Odometry Enabled Aerial Robot Navigation,” *IROS*, pp. 1–6, Oct. 2019.

[90] W. A. S. Norzam, H. F. Hawari, and K. Kamarudin, “Analysis of Mobile Robot Indoor Mapping using GMapping Based SLAM with Different Parameter,” *IOP Conf Ser Mater Sci Eng*, vol. 705, no. 1, Dec. 2019.

[91] T. Yong, J. Shan, R. Fan, W. Tianmiao, and G. He, “An improved Gmapping algorithm based map construction method for indoor mobile robot,” *HIGH TECHNOLOGY LETTERS*, vol. 27, no. 3, pp. 227–237, Sep. 2021, Accessed: Jan. 30, 2022.

[92] C. Kolhatkar and K. Wagle, “Review of SLAM algorithms for indoor mobile robot with LIDAR and RGB-D camera technology,” *Innovations in electrical and electronic engineering*, vol. 661, pp. 397–409, 2021.

[93] das Sargarnil, “Simultaneous Localization and Mapping (SLAM) using RTAB-Map,” Sep. 2018.

[94] C. Suat Gurel, “REAL-TIME 2D AND 3D SLAM USING RTAB-MAP, GMAPPING, AND CARTOGRAPHER PACKAGES,” 2018.

[95] A. Lorente Rogel, “COMPONENT-BASED SLAM IN RTAB-MAP,” *Robotics and mechatronics*, vol. 1, no. 1, 2021.

[96] S. Petkovi´c, L. M. Mili´c, N. Nikoli´c, D. M. Miškovi´c, and M. R. Rakovi´c, “Comparison of SLAM algorithms on omnidirectional four wheel mobile robot,” *International conference icetran*, vol. 1, no. 1, pp. 1–6, 2022.

[97] Intel, “Intel RealSense Product Family D400 Series,” 013, Apr. 2022. Accessed: Aug. 18, 2022. [Online]. Available: https://www.intelrealsense.com/wp-content/uploads/2022/05/Intel-RealSense-D400-Series-Datasheet-April-2022.pdf

[98] Intel, “Intel RealSense Product Family D400 Series Calibration Tools and API,” *Intel RealSense*, pp. 1–75, Jul. 2021.

[99] M. Luqman, “Writing a simple publisher and subscriber (python),” *ROS.org*, Sep. 30, 2021.

[100] G. Solano, “Segmentación de personas con MEDIAPIPE SELFIE SEGMENTATION,” *OMES*, Aug. 12, 2021.