UNIVERSIDAD SANTO TOMÁS
PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA

Diseño y construcción de un robot móvil autónomo para localización y mapeo simultáneos (SLAM) en ambientes cerrados domésticos.

TATIANA ANDREA ROZO MANRIQUE
BRAYAN STIVEN PALLARES OLIVARES

UNIVERSIDAD SANTO TOMÁS
INGENIERÍA ELECTRÓNICA
BOGOTÁ D.C
2021

Diseño y construcción de un robot móvil autónomo para localización y mapeo simultáneos (SLAM) en ambientes cerrados domésticos.

TATIANA ANDREA ROZO MANRIQUE
BRAYAN STIVEN PALLARES OLIVARES

Proyecto de grado presentado como requisito para optar al título de INGENIERO ELECTRÓNICO

UNIVERSIDAD SANTO TOMÁS
INGENIERIA ELECTRÓNICA
BOGOTÁ D.C
2021

UNIVERSIDAD SANTO TOMÁS

Ingeniería Electrónica

Este proyecto de grado fue aceptado para la carrera de Ingeniero(a) Electrónico(a) en Junio 2021

Director: Juan Manuel Calderon Chavez
Doctorado Electric Engineering
University of South Florida
Magíster en Ingeniería Electrónica y de Computadores
Universidad de los Andes

Co-director: José Guillermo Guarnizo Marín
Doctorado en Automática, Robótica e Informática Industrial
Universidad politécnica de Valencia - Internacional
Maestría en Automatización Industrial
Universidad Nacional de Colombia sede Bogotá

Co-director: Edgar Camilo Camacho Poveda
Magíster en Ingeniería Electrónica
Universidad Nacional

Jurados: Jhon Erik Navarrete Gómez
Magíster en Ingeniería Electrónica
Universidad Santo Tomas

Sindy Paola Amaya
Magíster en Ingeniería Electrónica y de Computadores
Universidad de los Andes

Sustentación proyecto de grado: 28 Junio 2021, BOGOTÁ D.C

Brayan Stiven Pallares Olivares _____

Tatiana Andrea Rozo Manrique _____

NOTA DE ACEPTACIÓN

El trabajo de grado titulado « **Diseño y construcción de un robot móvil autónomo para localización y mapeo simultáneos (SLAM) en ambientes cerrados domésticos.**» realizado por los estudiantes **Brayan Stiven Pallares Olivares** y **Tatiana Andrea Rozo Manrique** cumple con los requisitos exigidos por la **Universidad Santo Tomás** para optar al título de **Ingeniero Electrónico.**

Juan Manuel Calderon Chavez _____

José Guillermo Guarmizo Marín _____

Edgar Camilo Camacho _____

Jhon Erik Navarrete Gómez _____

Sindy Paola Amaya _____

DEDICATORIA

A nuestra familia y amigos por acompañarnos en cada momento, motivarnos a ser mejores personas e inspirarnos a alcanzar nuevos conocimientos.

AGRADECIMIENTOS

A Dios. A nuestros padres y familias por ser los principales promotores de nuestros sueños, siempre confiar en nuestras capacidades y guiarnos en cada decisión. A nuestros directores de proyecto de grado por brindarnos todos sus conocimientos, experiencia, apoyo y paciencia en el desarrollo de este proyecto. A todos los docentes de la universidad por los conocimientos otorgados. A nuestros compañeros y amigos con los cuales compartimos esta importante etapa de nuestra vida. ¡Muchas gracias a todos!

Índice general

Índice de cuadros

Índice de figuras

Glosario

Se presenta un glosario, con las palabras con mayor relevancia dentro del proyecto.

- **SLAM:** Simultaneous localization and mapping.
- **PID:** Proporcional, integral y derivativo.
- **AMR:** Autonomous Mobile Robots.
- **ROS:** Robot Operating System.
- **ABC:** Artificial Bee Colony.
- **RTAB-Map:** Real Time Appearance Based Mapping.
- **EKF:** Extended Kalman filter.
- **MPC:** Model Predictive Control.
- **WM:** Working memory.
- **STM:** Short-term memory.
- **LTM:** Long-term memory.
- **TEB Planner:** Trayectoria de banda elástica cronometrada.

Resumen

Este proyecto de grado presenta el diseño y desarrollo de un robot móvil omnidireccional que brinde capacidades de movilidad a robots sociales y actuadores robóticos. Se detalla el diseño mecánico, electrónico y de software necesarios para el funcionamiento del robot. Se realizan los cálculos de cinemática y odometría para el movimiento del robot y se implementa el algoritmo de SLAM, RTAB-Map, que le permite al robot realizar mapas de entornos domésticos, localizarse y navegar en ellos. Como producto final se presenta: el diseño y modelo 3D del robot, un repositorio con los paquetes desarrollados en ROS y el robot construido.

Introducción

Actualmente se están desarrollando estudios en robótica para dar solución a todo tipo de problemáticas, desde robots de producción en grandes fábricas hasta pequeñas aspiradoras inteligentes en casa. Uno de los campos de la robótica que está tomando más relevancia es la robótica móvil, ya que para que la mayoría de los robots sean aplicables en entornos reales, es necesario que estos cuenten con capacidades de movilidad.

La robótica móvil surgió para dar solución principalmente a 3 preguntas ¿Dónde estoy?, ¿A dónde quiero ir?, ¿Como llegar ahí? Al dar solución a estas 3 preguntas es posible dotar con capacidades de movimiento a un robot. Como parte de la solución a estas preguntas surgen los métodos de localización y mapeo simultáneos (SLAM). Estos métodos implementan algoritmos de mapeo, que le brinda al robot información de su entorno para mediante algoritmos de localización, poder determinar su posición actual. Por último, implementan un algoritmo de navegación que indica la ubicación objetivo del robot y busca la mejor forma de llegar a este.

Los robots móviles solucionan los problemas de movilidad a los que se puede enfrentar la robótica, pero la mayoría de las soluciones suelen ser costosas, limitando su aplicabilidad. A lo largo de este proyecto se plantea desarrollo de un robot móvil omnidireccional orientado al costo y de fácil implementación, que brinde capacidades de movimiento a robots sociales y manipuladores robóticos.

Este trabajo de grado se ha desarrollado gracias al soporte, apoyo y

dirección del Grupo de investigación y Desarrollo en Robótica de la Universidad Santo Tomas (G.E.D.). El grupo G.E.D. se enfoca principalmente al desarrollo de tres líneas de investigación denominadas Sistemas Inteligentes, Visión Artificial y Robótica. Dentro de estas líneas de investigación el grupo ha desarrollado diferentes proyectos de investigación enfocados en robótica cooperativa, robótica educativa, inteligencia artificial y robots humanoides. El grupo ha participado activamente de la iniciativa educativa y de robótica denominada "RoboCup" en donde se diseñó y desarrolló un equipo de fútbol de robots para competir en la liga "Small Size" y actualmente enfoca su participación en la liga "@Home" tal como se muestra en los trabajos [1], [2]. Se desarrollaron estrategias de control para la reducción de la fuerza de impacto en robots humanoides cuando estos caen o saltan [3], [4], [5], [6]. Además han trabajado en el uso de la robótica para incentivar a estudiantes de colegio a optar por carreras afines a la ciencia, matemáticas e ingeniería [7], [8], [9]. Actualmente se desarrollan proyectos de investigación enfocados en la robótica de enjambre [10], [11], [12] y sistemas de aprendizaje automático tal como [13], [14], [15], [16].

Capítulo 1

Planteamiento del problema

Por años se ha soñado con robots capaces de realizar tareas y labores domésticas, cuidar de las personas dependientes, atender a los ancianos y demás. Gracias a los avances tecnológicos de los últimos años y el surgimiento de la robótica social, este sueño está cada vez más cerca de ser una realidad. La robótica social es el estudio de robots capaces de interactuar y comunicarse entre ellos, con los seres humanos y con el medio ambiente, estas características han hecho que aumente el interés por desarrollar estos robots para diferentes aplicaciones, algunas de ellas como: guías turísticos, robots de vigilancia, robots de servicio, robots de rehabilitación y ayuda, entre muchas otras [17] [18].

La robótica social ha hecho que los robots pasen de estar inmersos en grandes fábricas a habitar en nuestros hogares, haciendo necesario el desarrollo de métodos que les permitan navegar de forma segura, confiable y natural alrededor de los humanos [19]. Debido a que los robots sociales se desempeñan en gran cantidad de tareas y la mayoría de ellas sin supervisión humana, no es practico que estos sean operados a distancia o configurados con rutas de tránsito preestablecidas, es necesario brindarles capacidades de navegación autónoma.

Con el fin de dar solución a este problema, la robótica social se ha apoyado en los avances y herramientas entregados por la robótica móvil [20], dentro de esta rama de la robótica encontramos los robots móviles

autónomos (AMRs), los cuales son capaces de moverse de forma independiente, evadir obstáculos, calcular rutas e identificar su ubicación en tiempo real. Estos robots pueden ser programados con diversos algoritmos y técnicas dependiendo del tipo de sensores utilizados y el ambiente a explorar. Dada esta relación entre la robótica social y la robótica móvil, y la problemática anteriormente mencionada, se plantea la siguiente pregunta:

¿Cómo diseñar una plataforma robótica móvil capaz de crear mapas de entornos cerrados domésticos y navegar a partir de ellos, con el fin de brindar capacidad de movimiento autónomo a un robot social?

Capítulo 2

Justificación

Anteriormente se evidenció el interés por el desarrollo de robots sociales, uno de los objetivos por los cuales se está trabajando en ellos es para disminuir el tiempo que emplean las personas realizando tareas caseras cotidianas.

Según el informe presentado en junio del año 2019 por el departamento de trabajo de los Estados Unidos de América, acerca del uso del tiempo por parte de los ciudadanos estadounidenses, se encontró que en un día común el 84 por ciento de las mujeres y el 69 por ciento de los hombres gastan gran parte de su tiempo realizando actividades del hogar, empleando en promedio 1.78 horas diarias en realizar estas actividades [21].

Otro de los objetivos por los cuales se está trabajando en el desarrollo de robots sociales, es ayudar a las personas ancianas o dependientes que requieren de supervisión y apoyo en el desarrollo de sus actividades cotidianas, ya que se estima que para el año 2050, el número de personas dependientes sobrepase el número de personas disponibles para prestar sus cuidados [22]. La robótica social tiene la capacidad de solucionar estos problemas, ya que estos robots son capaces de interactuar con los seres humano y con el entorno, ya han sido usados para dar solución a múltiples problemas y cada día serán usados para dar solución a más.

Los robots sociales son muy útiles en tareas de manipulación,

reconocimiento de objetos e interacción con seres humanos. Aún con estas cualidades, estos robots no serían aplicables a entornos reales de no contar con habilidades de navegación, ya que el robot se va a encontrar en ambientes domésticos y debe reconocer la presencia de personas para evitar lastimarlas de alguna forma, manteniendo un espacio seguro y confiable [23].

Es en el área de la navegación donde los robots móviles destacan, ya que son capaces de crear mapas, evadir obstáculos y calcular rutas en tiempo real [24]. Todas estas características son muy útiles en los entornos donde se desempeñan los robots sociales, puesto que la mayoría de estos trabajan en entornos cerrados domésticos. Debido a que cada entorno es único, los robots se encontrarán en ambientes desconocidos, donde es necesario que realicen de forma autónoma la identificación del medio. Esta es la razón principal por la cual los robots sociales hacen uso de las herramientas y técnicas de la robótica móvil y por la cual se plantea en este proyecto la construcción de una plataforma móvil robótica para aplicaciones de robótica social.

Por otra parte, se considera una mejor alternativa realizar la construcción del robot en su totalidad, en lugar de adquirir un modelo comercial, debido a que esto limitaría la posibilidad de probar diferentes técnicas de mapeo y dificultaría el acceso al hardware.

Capítulo 3

Impacto Social

Según [22], el número de personas mayores de 65 años pasará de unos 524 millones en 2010 a casi 1.500 millones en 2050. El aumento del número de personas mayores va acompañado de un incremento de la edad media de la población. En [25] se estima que en 2050 el porcentaje de personas mayores dependientes en EE.UU. será del 34 %, en la UE del 49 % y en Japón del 76 %.

La elevada proporción de personas mayores dependientes es una gran preocupación porque conlleva problemas económicos y sanitarios. En un futuro próximo, será necesario el uso de robots, porque no habrá suficientes personas para satisfacer las necesidades de los ancianos. En la actualidad, se están investigando y desarrollando robots que interactúan con el ser humano en diferentes ámbitos como guías turísticos [26], robots de vigilancia [27], robots de servicios [28], robots de búsqueda y rescate [29], [30] robots de desinfección [31] y rehabilitación [32], entre muchos otros.

Desafortunadamente, el desarrollo del robot social suele ser costoso, como mencionan en [33] y en [34], lo que limita su aplicabilidad. Para resolver el problema de la asistencia a las personas mayores mediante robots, es necesario desarrollar robots orientados al costo que sean accesibles a la población en general, fáciles de construir, programar e implementar.

Este documento presenta el diseño de un robot orientado al costo que proporciona capacidades de movilidad a los robots sociales y manipuladores estáticos, proporcionando una plataforma móvil para facilitar y reducir el costo de implementación de un robot social en entornos domésticos interiores.

Capítulo 4

Antecedentes

Cada día se logran grandes avances y desarrollos en robótica, convirtiéndose en algo cada vez más cotidiano. Hoy en día se cuenta con robots desempeñando tareas complejas en diferentes áreas, tales como: desactivación de minas antipersonal y artefactos explosivos, búsqueda y rescate de sobrevivientes en zonas de desastre, exploración extraterrestre, apoyo en el hogar y la industria, entre otras.

A lo largo de la historia, la robótica se ha enfocado en los sectores industriales más desarrollados, logrando de esta forma una producción masiva en industrias tales como la automovilística, metalúrgica, química y demás. En los últimos años este enfoque ha cambiado, buscando la forma de aplicar la robótica a entornos más sociales y cotidianos, creando de esta forma nuevas ramas, técnicas y herramientas que faciliten la aplicación de la robótica a nuevos entornos [35]. Es así como surge la robótica social, que busca crear robots con la capacidad de interactuar y comunicarse de forma natural.

4.1. Robot omnidireccional

Los robots móviles, por otro lado, tienen la capacidad de moverse en diferentes ambientes, aun así, estos deben equiparse con herramientas que les permitan percibir información de su entorno y tomar decisiones

basándose en ésta. Es por esto que los robots móviles son equipados con cámaras y sensores, la información entregada por estos sensores es tratada mediante diferentes técnicas con el fin de crear mapas de los entornos en los que el robot navega. Un ejemplo del proceso de diseño y construcción de un robot móvil autónomo es mostrado en [36], donde se comparan diferentes tipos de llantas omnidireccionales con el fin de encontrar las que mejor se adapten a esta tarea, también muestra el desarrollo del diseño mecánico para realizar la cinemática del robot a partir de sus dimensiones. Una vez se ha construido el robot, se realizó un controlador digital para controlar el movimiento, implementando diferentes modos de operación: modo seguidor de línea, modo de control remoto y modo autónomo. Para concluir se realizó una comparativa entre las diferentes alternativas de configuración del robot y los propósitos para los que puede ser empleado.

En [37] se presenta un diseño de robot móvil autónomo realizado a partir de 3 módulos: chasis, control y aplicación. Se hizo uso de servomotores DC, cada uno con una potencia de 150 W, para el control de estos se utilizó un controlador PID. Se equipó al robot con múltiples sensores para que este sea capaz de navegar en el entorno. Para localizar el robot se implementó un algoritmo para *filtro de Kalman extendido* (EKF) en conjunto con las señales obtenidas de los encoders de cada llanta y el sensor Kinect. Se realizaron pruebas midiendo el error de posición manualmente, encontrando que el error variaba entre 1 y 4 mm, determinando que el robot efectivamente es capaz de seguir rutas, en un futuro el robot será equipado con más sensores a fin de aumentar la fiabilidad. En [38] se presenta la construcción de un robot móvil omnidireccional inteligente que utiliza cuatro ruedas Mecanum, cada llanta cuenta con un motor DC de engranajes precisos, se utilizaron controladores de motor LMD 18200, en este trabajo explican las etapas y materiales necesarios para la construcción del robot mecánica y electrónicamente, muestran las acciones de movimiento generales y el control de los motores en el modo básico.

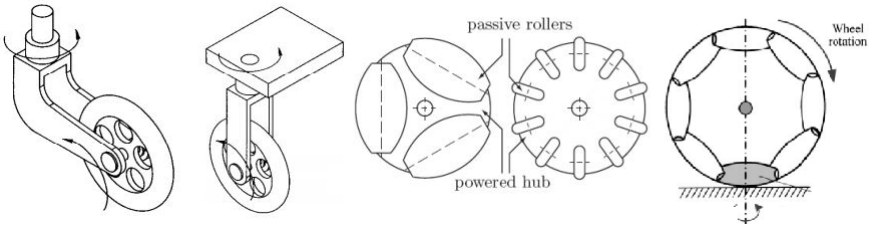

Figura 4.1. Diferentes tipos de llantas omnidireccionales [36].

Es común encontrar llantas omnidireccionales en los diseños de robots móviles autónomos. Existen diferentes configuraciones de ruedas tal como se puede evidenciar en la figura 4.1, estas llantas le brindan versatilidad a los robots, haciéndolos capaces de realizar múltiples movimientos. Un tipo de llantas omnidireccionales muy usadas son las conocidas como "Mecanum Wheels", en [39] se hace uso de estas llantas en un robot, logrando que este se mueva a lo largo de un camino predeterminado, al mismo tiempo que es capaz de rotar sobre su centro, de esta forma se demostró la funcionalidad de estas llantas y su cinemática. En la figura 4.2 se muestran dos diagramas realizados para calcular las cinemáticas de robots con configuraciones de 3 y 4 ruedas.

Al hacer uso de las llantas mecanun es importante considerar el cálculo de la cinemática del robot ya que estas ruedas no se comportan como las llantas comunes, en [40] se presenta una plataforma que utiliza cuatro ruedas Mecanum, se realiza el cálculo de la cinemática directa e inversa del robot mediante ecuaciones cinemáticas similares a las obtenidas a partir de resultados experimentales, con estos cálculos se logró que la plataforma realice movimientos omnidireccionales complejos alcanzando cualquier dirección entre 0 y 360 grados sin cambiar su orientación.

En [42] se muestra el desarrollo y verificación del modelo matemático de una plataforma robótica móvil que utiliza llantas omnidireccionales para lograr el movimiento en cualquier dirección, el modelo matemático fue utilizado para lograr un control autónomo de un robot móvil desarrollado como robot de servicio de oficina. Se concluyó que se

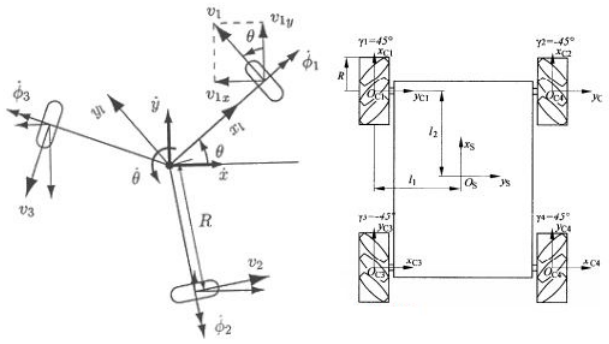

Figura 4.2. Ejemplo diagrama cinemático de dos configuraciones diferentes de robots móviles [41] [36].

pueden lograr 81 combinaciones diferentes de movimiento de las ruedas, donde 51 movimientos se pueden usar para realizar movimientos de la plataforma y los 30 restantes son movimientos no funcionales, es decir, no producen movimiento alguno. En [43] se realiza el seguimiento de la trayectoria de un robot móvil omnidireccional con 3 ruedas. Se establece el modelo cinemático y se realizan algoritmos de *Model Predictive Control* (MPC), el sistema se diseñó para lograr la estabilización de puntos y el seguimiento de la trayectoria. Se realizó un análisis comparativo entre el robot con ruedas omnidireccionales y la estructura tradicional de doble rueda. Las pruebas muestran la estabilización de puntos y el seguimiento de trayectoria logrados. Otro trabajo con robots de 3 ruedas se evidencia en [41], dónde detallan los cálculos pertinentes para calcular la cinemática de este robot y diseñar el controlador del mismo, se realizaron varios experimentos para determinar el desempeño de los controladores de movimiento bajo varias condiciones, al analizar los resultados de las pruebas se determinó que el mejor controlador era el controlador PI diseñado. En [44] también se realiza el control de un robot omnidireccional de 3 ruedas, pero en este caso con un controlador *inverse input-output linearization based control*, se realizaron todos los cálculos pertinentes para el desarrollo de este. El diseño del controlador fue probado haciendo que el robot se moviera siguiendo una ”forma de ocho” a una velocidad constante, obteniendo resultados positivos ya que

el error obtenido fue menor al deseado.

En [45] se describe el proceso para la construcción de un robot móvil capaz de planear rutas. Una vez construido el robot, se comparó con otros, realizando pruebas en las cuales se evaluó la funcionalidad de cada uno, se determinó que el robot que mejor se desempeñó fue el diseñado, ya que cuenta con alta precisión y su tiempo de respuesta es más rápido.

4.2. Localización y Mapeo Simultáneos (SLAM)

Como se mencionó anteriormente los robots móviles autónomos hacen uso de múltiples sensores y algoritmos con el fin de ser capaces de navegar, una de las técnicas más utilizadas para esta tarea es el *Real-Time Appearance-Based Mapping* (RTAB-Map), la cual fue desarrollada por dos estudiantes de la universidad de Sherbrooke en el año 2013, RTAB-Map es un enfoque SLAM basado en gráficos RGB-D, estéreo y Lidar, este método utiliza la probabilidad para determinar cuándo una imagen proviene de una ubicación nueva y hace uso de un optimizador para reducir errores [46] [47].

La figura 4.3 muestra el resultado obtenido por Mathieu Labbé y François Michaud creadores del método RTAB-Map, que en [48] intentan solucionar el problema de mapear grandes entornos, ya que los recursos del robot son limitados, el tamaño de los mapas que pueden crear también lo son. Otro problema de mapear grandes entornos surge cuando el robot se enciende en un entorno no mapeado, ya que no es capaz de ubicarse porque no encuentra similitudes con las zonas mapeadas.

Para dar una solución y aumentar la capacidad de mapeo de los robots, Mathieu Labb'e y François proponen el uso de un enfoque de gestión de la memoria [49], considerar sólo porciones del mapa para reducir los requisitos computacionales y permitir el mapeo multisesión que hace posible unir mapas de diferentes áreas. Como se muestra en la figura 4.3 en la que se han unido cinco mapas diferentes.

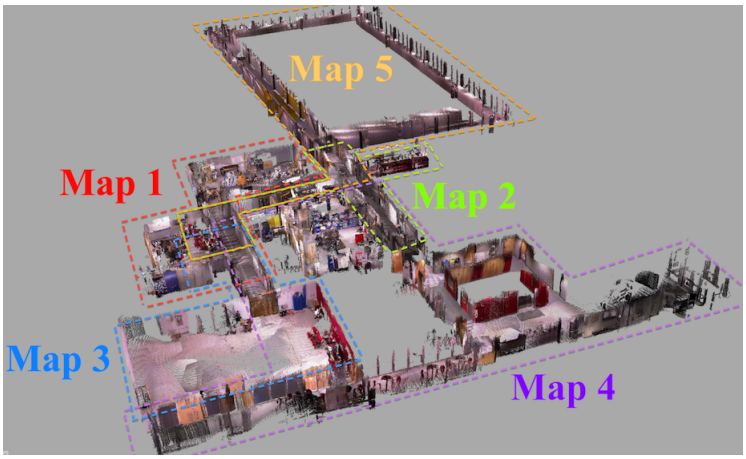

Figura 4.3. Mapa de demostración creado usando RTAB-Map [50].

Este algoritmo es utilizado en robots para todo tipo de tareas, en [51] se hace uso de esta técnica para equipar a un robot móvil con la capacidad de navegar en entornos desconocidos, esto con el fin de ayudar en operaciones de búsqueda y rescate. Esta técnica puede ser implementada en conjunto con diferentes algoritmos para mejorar su rendimiento, una muestra de esto es su implementación con el algoritmo de *Loop Closure* [52]. Otro tipo de trabajos muestran el uso de técnicas de SLAM en diferentes robots, como en [53] donde se examina cómo los humanos trabajan con robots móviles no tripulados teleoperados con el fin de crear mapas 3D, se equipó al robot con una cámara RGB-D y se realizó odometría visual para la generación de los mapas, se llegó a la conclusión que el sistema teleoperado tuvo éxito al extender la perspectiva humana de sus entornos.

En el artículo [54] se desarrolló un enfoque de planeación de ruta híbrida, donde el algoritmo debe encontrar la mejor ruta de acuerdo con la energía que consumirá, la suavidad del camino, la seguridad vial, la longitud del camino y las restricciones de duración de la tarea. Se propuso un nuevo algoritmo híbrido con la cooperación del algoritmo *Artificial Bee Colony* y el algoritmo *Probabilistic Roadmap*. Con el algoritmo propuesto se obtuvo una eficiencia mayor a los algoritmos *Probabilistic Roadmap* (PRM) y *Artificial Bee Colony* (ABC).

Capítulo 5

Objetivos

5.1. Objetivo General

Diseñar y construir un robot móvil autónomo, capaz de realizar tareas de mapeo y navegación en ambientes cerrados domésticos de una planta, que brinde capacidad de desplazamiento a robots sociales.

5.2. Objetivos Específicos

- Diseñar y construir un robot omnidireccional que permita controlar de forma independiente la velocidad angular de cada llanta.
- Implementar los algoritmos para el cálculo de la cinemática y la odometría del robot en el sistema de cómputo abordo del mismo.
- Implementar los algoritmos para mapeo, localización y navegación, en el sistema de cómputo abordo del robot.
- Medir la precisión espacial en la localización y el mapeo, en ambientes domésticos cerrados.

Capítulo 6

Marco teórico

6.1. Robótica

Los robots son una herramienta valiosa que ya está siendo utilizada en diferentes áreas como: trabajo, estudio, salud y hogar, etc [55]. Por ello se hace énfasis en el desarrollo de robots colaborativos, capaces de apoyar el desarrollo humano, potencializar la automatización, aumentar la productividad y ayudar a ejecutar eficientemente las actividades industriales o de servicio.

6.1.1. Robótica móvil

La robótica móvil se especializa en el desarrollo de robots capaces de desplazarse por diferentes tipos de entornos, bien sea en tierra, agua o aire [56]. Las características que poseen los robots móviles se centran en su capacidad de realizar desplazamientos y navegación por distintos ambientes. Para equipar el robot con la capacidad de movimiento se requiere de todo un esquema que le permita procesar la información obtenida por los sensores, almacenar las características del entorno y aprender de él. La robótica móvil surge para solucionar los problemas de movilidad de los robots en diferentes ambientes, haciéndolos útiles en una gran variedad de tareas como puede ser la asistencia de ancianos y personas dependientes [28], la detección de víctimas en zonas de desastre

[29], la desinfección de superficies [31] y muchas más actividades [26] [27] .

6.1.2. Robótica social

La robótica social estudia a los robots que son capaces de interactuar con los seres humanos, con el entorno y con ellos mismos [57]. La capacidad de interacción de estos robots facilita la integración de la robótica en entornos domésticos, brindando ayuda para el desarrollo de tareas cotidianas o servicio a personas con limitaciones que requieran una atención especial.

6.2. Cinemática

La cinemática es el estudio del comportamiento básico de los sistemas mecánicos. En un robot móvil la cinemática es necesaria para comprender el comportamiento mecánico del robot y así diseñar sistemas capaces de controlar y determinar su movimiento. La cinemática se divide en dos: cinemática directa y cinemática inversa [58] [59].

Para realizar el cálculo de la cinemática se debe determinar la configuración del robot, tipos de llantas a utilizar y medidas. La figura 6.1 muestra la configuración del robot diseñado con cuatro llantas mecanum. A partir de esta configuración se realiza el cálculo de la cinemática directa e inversa del robot [59].

6.2.1. Cinemática directa

Se denomina cinemática directa al modelo que determina las velocidades de un robot Vx, Vy, W, a partir de las velocidades individuales de cada llanta y la configuración del robot [60]. La ecuación 6.1 describen este cálculo.

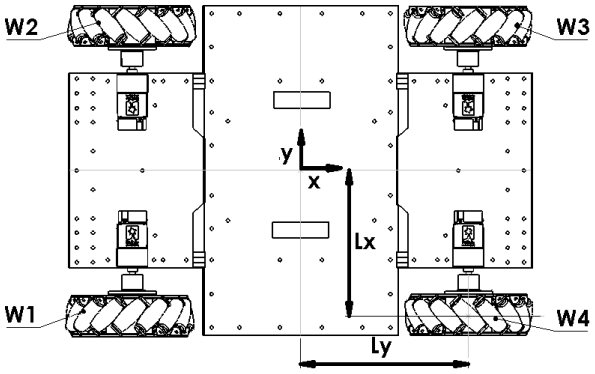

Figura 6.1. Descripción ejes para la cinemática.

$$\begin{bmatrix} Vx \\ Vy \\ Wz \end{bmatrix} = \frac{r}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ -1 & 1 & 1 & -1 \\ -\frac{1}{lx+ly} & \frac{1}{lx+ly} & -\frac{1}{lx+ly} & \frac{1}{lx+ly} \end{bmatrix} \begin{bmatrix} W1 \\ W2 \\ W3 \\ W4 \end{bmatrix} \tag{6.1}$$

6.2.2. Cinemática inversa

La cinemática inversa es el modelo a partir del cual se determina la velocidad a la cual se debe mover cada una de las llantas, para que el robot se desplace a determinada velocidad [61]. La ecuación 6.2 describen este cálculo.

$$\begin{bmatrix} W1 \\ W2 \\ W3 \\ W4 \end{bmatrix} = \frac{1}{r} \begin{bmatrix} 1 & -1 & -(lx+ly)) \\ 1 & 1 & (lx+ly) \\ 1 & 1 & -(lx+ly) \\ 1 & -1 & (lx+ly) \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ \omega \end{bmatrix} \tag{6.2}$$

6.3. Slam

La localización y mapeo simultáneos (SLAM) es una técnica utilizada en la robótica que permite la construcción de mapas de entornos desconocidos. Esta técnica permite identificar la ubicación del robot y

realizar mapas para usos posteriores [62]. Al conocer las características de su entorno será posible que el robot tenga una navegación eficiente.

Existen gran variedad de técnicas de SLAM para todo tipo de sensores y entornos, cada una aplica diferentes métodos para lograr la localización y mapeo simultáneos, algunas de las más usadas son: RatSLAM, VSLAM, Gmapping, REAL-TIME APPEARANCE-BASED MAP- PING (RTAB-MAP), ORB SLAM, Large-Scale Direct MonocularSLAM (LSD-SLAM).

6.3.1. RTAB-MAP

El mapeo en tiempo real basado en la apariencia (RTAB-MAP) es la técnica de SLAM más utilizada en la actualidad. Esta técnica fue desarrollada por dos estudiantes de la universidad de Sherbrooke en 2013 e integrada al sistema operativo para robótica (ROS) [63]. RTAB-Map se basa en RGB-D, estéreo y Lidar, utiliza la estadística para determinar cuándo una imagen proviene de una ubicación nueva y utiliza optimizadores para reducir errores [64] [65]. RTAB-Map es utilizada en robots para todo tipo y en gran variedad de tareas.

6.4. Odometría

Es el estudio que permite calcular el movimiento que realiza un robot, de acuerdo con las órdenes dadas, va recopilando información de los análisis de imágenes, integración de datos de sensores, cálculo de movimientos de acuerdo a un mapa; para luego procesar la información durante su navegación y estimar su ubicación [62].

La odometría visual se realiza a partir de un análisis de acuerdo a las imágenes captadas por el movimiento de la cámara. Se puede realizar con ayuda de un sensor como la cámara Intel Realsense T265, puesto que emite la posición y orientación actual 200 veces por segundo. La cámara cuenta con dos ojos de pez lentes con sensor IMU combinado 1635 FOV y BMI055 equipadas [66].

6.5. ROS

Robot Operating System (ROS) está compuesto por una biblioteca de hardware y software, que respaldan el avance de sistemas de robots. Una de sus características más relevantes es su plataforma de código abierto, por lo que ha ayudado con marcos de desarrollos de software para robots [67].

ROS funciona como la principal unidad informática, permite el intercambio de información entre diferentes dispositivos de forma eficiente, además proporciona herramientas que facilitan las comunicaciones entre cada proceso. Este middleware proporciona diferentes paquetes de controladores de hardware, también se vincula con otras bibliotecas apropiadas para la programación de robótica [68].

Capítulo 7

Diseño e Implementación

El diseño del robot se realiza teniendo en cuenta 3 requerimientos básicos: primero, el robot debe ser construido con materiales de fácil obtención, segundo, las dimensiones del robot le deben permitir moverse con facilidad por la mayoría de los entornos domésticos, tercero, el robot debe permitir la fácil integración de manipuladores robóticos o robots sociales.

El diseño del robot se divide en 3 partes esenciales: mecánica, electrónica y software.

7.1. Electrónica

La electrónica necesaria para el funcionamiento del robot se divide en 2, electrónica de alimentación y electrónica de procesamiento. La electrónica de alimentación consiste en una batería de 14.8 voltios a 4400mAh, 4 celdas y 65.1Wh que alimenta directamente a los motores, y un regulador que reduce el voltaje a 5 voltios para el resto de los componentes. La electrónica de procesamiento consiste en: una tarjeta UP Squared que se encarga de ejecutar el software necesario para el funcionamiento del robot, una cámara de seguimiento Realsense T265, una cámara de profundidad Realsense D435 y una tarjeta de desarrollo STM32F4DISCOVERY que recibe las ordenes de movimiento del robot y

las envía a 4 drivers Pololu TB9051 para que cada una accione un motor DC Pololu 37D.

La interconexión entre estos elementos puede ser detallada en la figura 7.1, donde las líneas continuas representan suministro de energía y la línea punteada representa suministro de información .

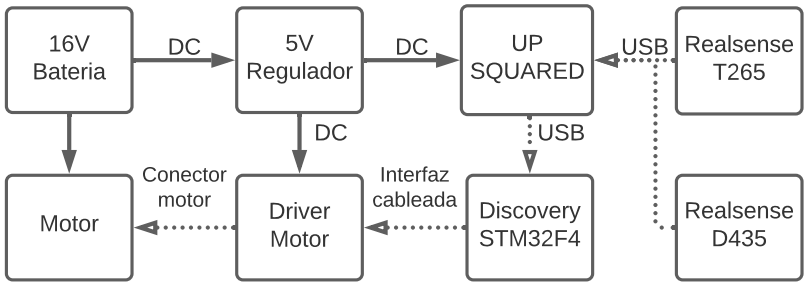

Figura 7.1. Esquema electrónico del robot. Fuente: Autor.

La conexión entre la tarjeta Discovery, los drivers y los motores es realizada mediante un circuito impreso PCB. La figura 7.2 muestra el esquema de componentes de la PCB y su conexión con un motor, el esquema se repite de igual forma para cada uno de los 4 motores.

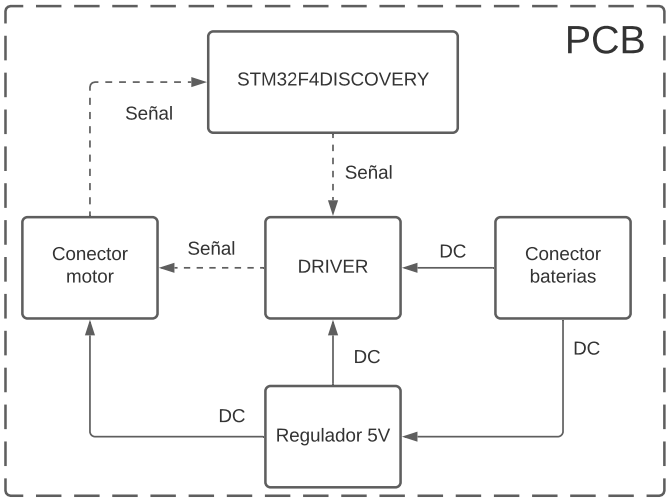

Figura 7.2. Esquemático PCB. Fuente: Autor.

La batería del robot se conecta directamente a la PCB, suministrando el voltaje de entrada del driver y pasando por un regulador de 5v que baja el voltaje para alimentar el circuito lógico del driver y el encoder de los motores. La tarjeta STM32F4FISCOVERY envía 2 PWM al driver para que este genere las señales de salida y accione los motores. A su vez el encoder de los motores envía dos señales a la tarjeta STM32F4FISCOVERY para que esta calcule la velocidad actual de cada motor.

La figura 7.3 muestra la PCB construida e integrada con la tarjeta STM32F4DISCOVERY y los drivers, el modelo a profundidad con el diseño de la capa superior e inferior puede ser visto en el anexo A, las medidas finales de la PCB son de 195.6 mm por 139.4 mm.

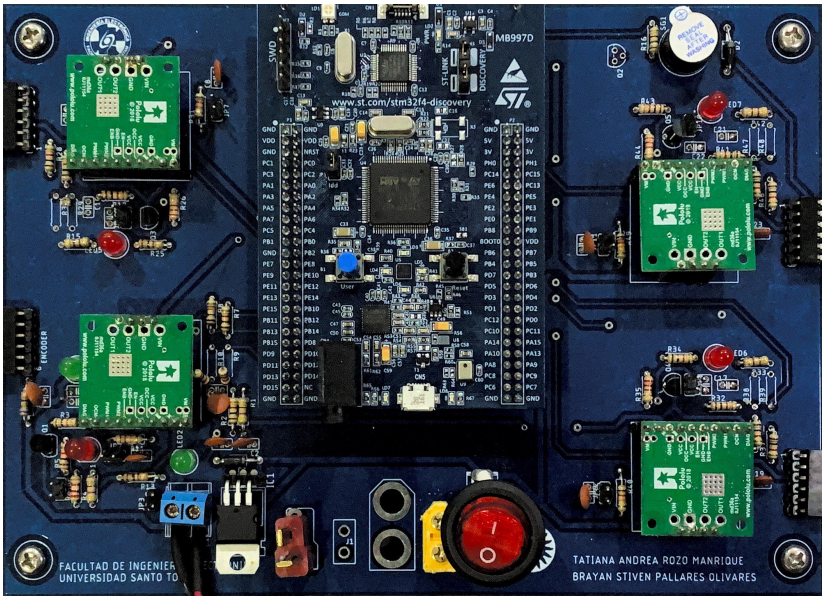

Figura 7.3. PCB. Fuente: Autor.

Se utilizan motores DC Pololu 37D de 12 voltios con encoder de 3200 pulsos por revolución. El encoder convierte el movimiento en una señal eléctrica mediante la cual es posible determinar la velocidad y la posición de cada llanta. Se seleccionaron estos motores ya que su tamaño es reducido, brindan hasta 21 kg*cm de torque y su voltaje de funcionamiento es bajo. Las especificaciones de estos motores pueden ser

Motorreductor Pololu 12V 50:01	
Tensión nominal	12 V
Corriente a plena carga	5.5 A
Corriente sin carga	0.2 A
Relación de transmisión	50:1
Velocidad sin carga	200 RPM
Potencia Max	10 W

Cuadro 7.1. Descripción de motores Pololu 37D. Fuente: Autor.

vistas en el cuadro 7.1.

Los motores son controlados por drivers Pololu TB9051, estos son drivers de puente H que permiten el control bidireccional de los motores. Tienen un rango de operación de 4.5 V a 28 V entregando hasta 2.6 A de corriente continua y 5 A de corriente pico. Los drivers proporcionan múltiples protecciones contra subtensión, sobre corriente, sobrecalentamiento y tensión inversa. Las especificaciones de este driver pueden ser detalladas en la tabla 7.2.

TB9051FTG	Min	Max
Tensión de alimentación del motor	4.5 V	28 V
Corriente de salida	2.6 A	5 A (a pico)
Frecuenta PWMs	-	20 kHz
VCC	5 V	

Cuadro 7.2. Descripción del driver TB9051FTG. Fuente: Autor.

El firmware del robot se ejecuta en una tarjeta STM32F4DISCOVERY, esta tarjeta cuenta con la capacidad de un microcontrolador STM32F407, incorpora la herramienta de depuración ST-LINK/V2-A, acelerómetro digital ST-MEMS, micrófono digital, DAC de audio, LED's, pulsadores y conector micro USB. Cuenta con múltiples canales PWM, contadores internos y módulo de comunicación en serie por USB, características que le permiten controlar los motores mediante los drivers y comunicarse con

STM32F4DISCOVERY	
Microcontrolador	STM32F407VGT6 Arm
Memoria Flash	1 Mbyte
RAM	192 Kbytes
Micrófono	Digital omnidireccional con sensor de audio ST-MEMS
Acelerómetro	3 ejes ST MEMS
Audio	DAC
Fuente de alimentación externa para aplicaciones	3 V y 5 V
Depurador	ST-LINK/V2-A
Compatibility	IAR Embedded Workbench®, MDK-ARM y STM32CubeIDE

Cuadro 7.3. Descripción de la tarjeta STM32F4DISCOVERY.

el sistema de cómputo del robot. Las principales características de esta tarjeta pueden ser detalladas en la tabla 7.3.

El computador principal del robot es una tarjeta UP Squared, esta tarjeta cuenta con un procesador Intel Atom x5-E3940, 8Gb de Ram, 3 USB 2.0, 2 USB 3.0, salida de HDMI, conector de Ethernet, módulo de Wifi y demás. Esta tarjeta se usa en el robot debido a su reducido tamaño, pero puede ser sustituida por cualquier ordenador con capacidad para ejecutar el sistema operativo para robótica (ROS) [69].

7.1.1. Sensores

Intel® RealSense D435

La cámara que se muestra en la figura 7.4 es la Intel® RealSense D435 es una cámara de profundidad con un rango mínimo de 0.105 metros y un máximo de 10 metros, con una resolución de 1280x720 de profundidad estéreo activa. Hasta 90 Fotogramas por segundo (fps). A la vez esta cuenta con módulo de RGB con una resolución de 1920x1080 y

una velocidad de 30fps [70][71]. El amplio campo de visión y el sensor de obturación global la convierten en una de las mejores opciones para navegación y reconocimiento de objetos, al tener un sensor de obturación proporciona una gran sensibilidad a la poca luz.

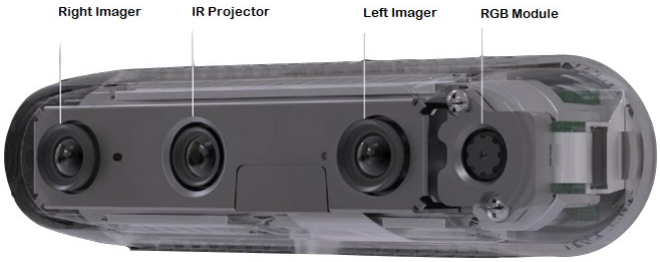

Figura 7.4. Cámara D435 [71].

Intel® RealSense T265

La cámara Intel® RealSense T265 se observa en la figura 7.5. Cuenta con dos lentes de ojo de pez para la detección de características, al unir las dos imágenes se puede estimar la profundidad densa, aunque los resultados no son óptimos. Los lentes están optimizados para un amplio campo de visión de seguimiento en lugar de precisión de profundidad, y hay textura proyectada sobre el medio ambiente para ayudar con el relleno de profundidad [72][73].

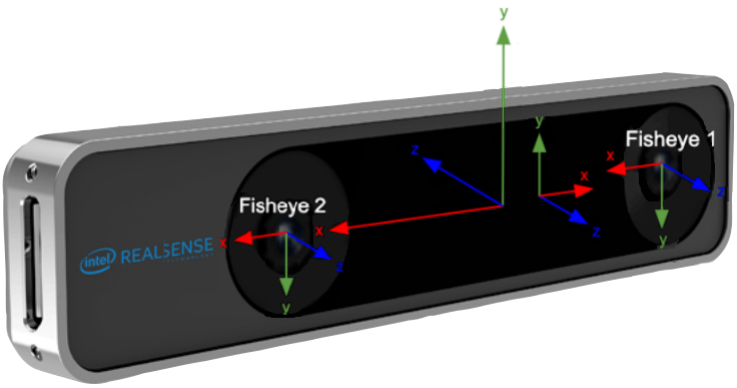

Figura 7.5. Cámara T265 [73].

7.2. Mecánica

7.2.1. Materiales

El robot se diseña en aluminio y MDF ya que se busca que sea resistente, liviano y fácil de construir. En la estructura se hace uso de perfil estructural de aluminio de 20x20mm, 20x40mm, 20x60mm y 20x80mm cortados a diferentes largos, las superficies del robot son diseñadas usando laminas de MDF de 4mm, La figura 7.6 muestra el modelo 3D de los perfiles de aluminio utilizados y sus medidas.

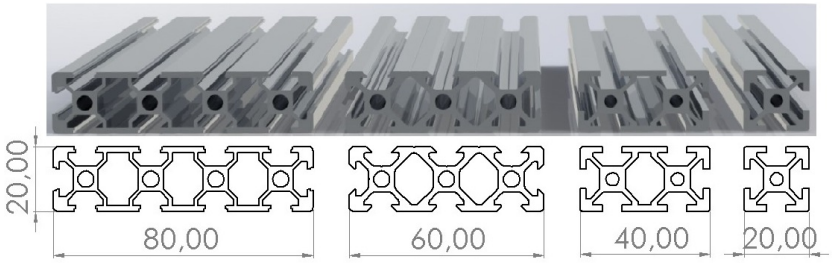

Figura 7.6. Sección inferior del robot. Fuente: Autor.

7.2.2. Llantas

Uno de los componentes fundamentales del robot son las llantas, ya que son estas las que le permiten al robot moverse de forma omnidireccional. Se seleccionan llantas Mecanum, estas llantas cuentan con rollers pasivos a una inclinación de 45 grados alrededor de su eje. Combinando 4 llantas mecanum de forma que se cancele la inclinación de sus rollers, es posible determinar combinaciones de movimientos de las llantas que le permitan al robot moverse en cualquier dirección sin necesidad de modificar su orientación, característica muy útil en ambientes domésticos ya que el robot puede alejarse fácilmente de obstáculos [36].

7.2.3. Distribución del robot

El diseño final del robot se divide en 5 niveles los cuales se agrupan en 3 secciones, la sección 1 agrupa los niveles 0 y 1, la sección 2 el nivel 2, y la sección 3 los niveles 3 y 4. La figura 7.7 muestra el diseño final del robot con sus niveles. En esta sección se detalla el proceso de diseño de cada sección, su propósito y componentes.

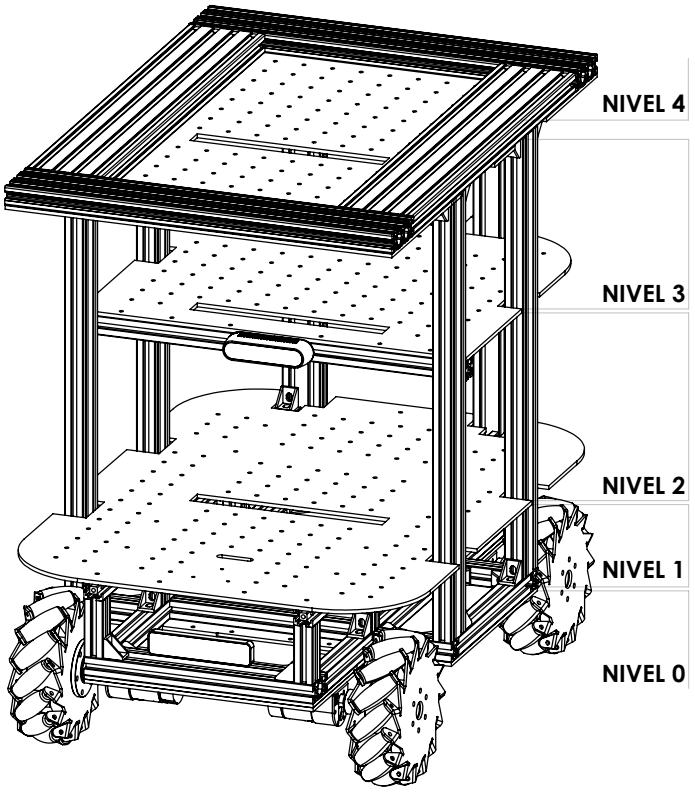

Figura 7.7. Plano del robot con sus niveles. Fuente: Autor.

Se diseño una primera versión de la sección 1 con estructura rectangular de perfil de aluminio, base central de MDF y llantas mecanum de 100 mm de diámetro. Esta versión fue construida como se muestra en la figura 7.8 para realizar pruebas de funcionalidad determinando que, el robot debe contar con más espacio para ubicar elementos electrónicos y soportar más niveles.

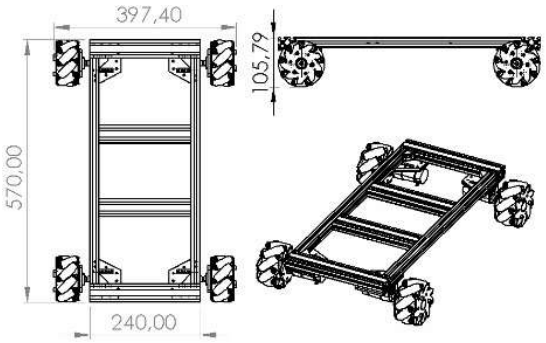

Figura 7.8. Primera versión de la sección 1. Fuente: Autor.

Esta primera versión de la sección 1 fue modificada remplazando las llantas mecanum de 100 mm de diámetro por unas de 152 mm, agregando una saliente a cada lado del robot y cubriendo toda la base con piezas diseñadas de MDF. Estas modificaciones brindan más espacio para ubicar elementos electrónicos y mayor estabilidad para soportar más niveles. El diseño final de la primera sección puede ser visto a detalle en la figura 7.9, esta sección agrupa los niveles 0 y 1 del robot. El nivel 0 es la parte inferior del robot donde se aseguran los motores y el nivel 1 la parte superior donde se ubica toda la electrónica necesaria para el funcionamiento del robot.

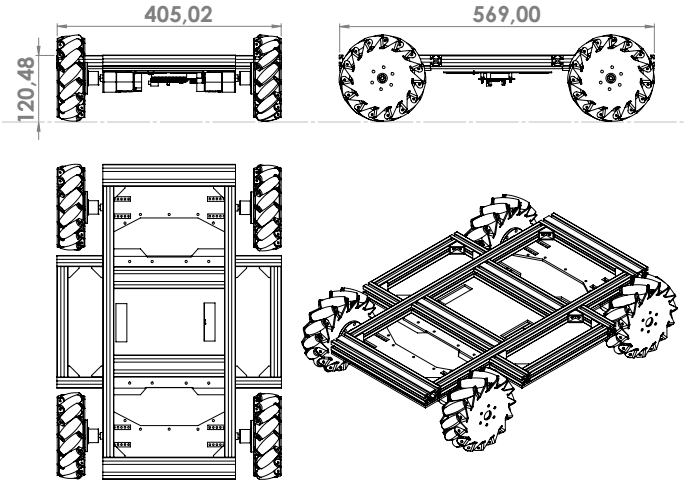

Figura 7.9. Planos de diseño final de la sección 1. Fuente: Autor.

La sección 2 se diseña cubriendo la sección 1 con una lámina de MDF, sobre esta superficie se ubica el computador principal del robot, se pueden agregar sensores y demás elementos necesarios. La lamina de MDF se esta sección cuenta con perforaciones para asegurar los diferentes elementos y rendijas para pasar cables. La figura 7.10 muestra el robot diseñado hasta la sección 2 con sus medidas en milímetros.

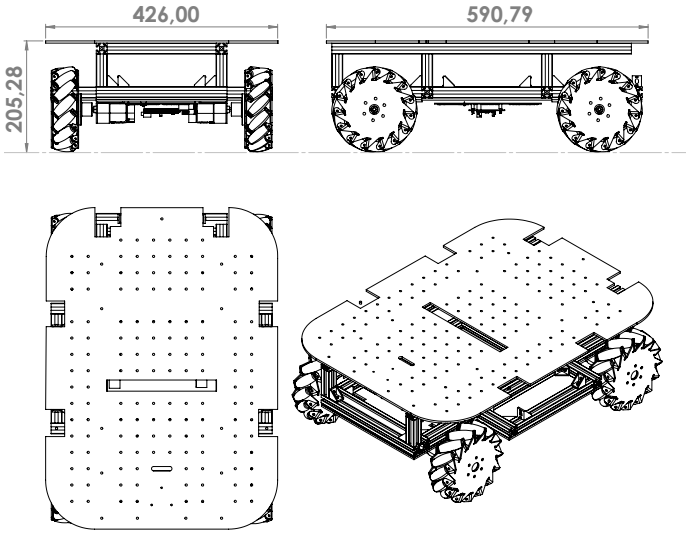

Figura 7.10. Plano del robot con dos secciones. Fuente: Autor.

La sección 3 agrupa los niveles 3 y 4, el nivel 3 es un nivel intermedio en el robot donde se ubica la cámara principal, en esta ubicación la estructura del robot no interfiere con el campo visual de la cámara siendo capaz de ver los obstáculos muy cercanos al robot. Adicionalmente este nivel cuenta con una lámina de MDF donde se pueden añadir diferentes elementos. El nivel 4 es la parte superior del robot, cuenta con perfiles de aluminio de 20x60 mm y 20x80 mm donde se pueden asegurar fácilmente manipuladores robóticos, robots sociales y cualquier tipo de elemento. Con este nivel la altura final del robot es de 600 mm, a esta altura un manipulador robótico es capaz de abrir la mayoría de las puertas y alcanzar objetos sobre superficies como escritorios o muebles. La sección 3 puede ser vista a detalle en la figura 7.11 con sus medias en milímetros.

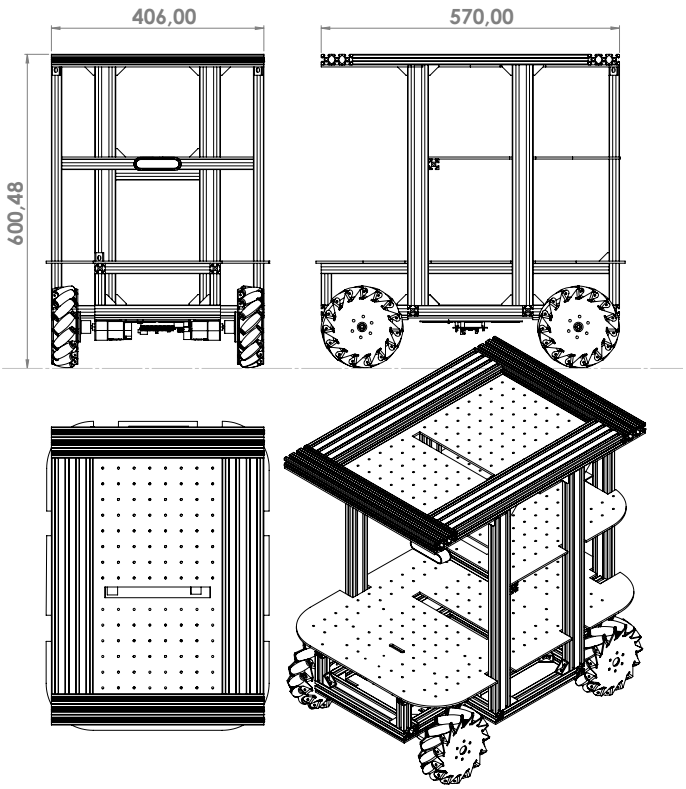

Figura 7.11. Plano del robot con tres secciones. Fuente: Autor.

7.3. Software

7.3.1. Firmware

Uno de los componentes principales del software es el firmware ya que es el encargado de ejecutar las funciones esenciales del robot, como lo son: controlar, medir la velocidad de cada motor, ejecutar cálculos de cinemática y odometría. Estos procesos son realizados dentro de una interrupción que se dispara con una frecuencia de 50hz.

Para medir la velocidad de cada motor se configuran 4 contadores duales de la tarjeta STM32F4DISCOVERY como entradas de encoders. Los contadores toman valores desde -10.000 hasta 10.000 y el encoder genera 3200 pulsos por revolución, por lo tanto, con un par de vueltas el contador se reinicia. El reinicio del contador se refiere a que su valor salta al extremo opuesto, es decir, de forma ascendente el contador pasa de 10.000 a -10.000 como muestra la figura 7.12 y de forma descendente el contador pasa de -10.000 a 10.000 como muestra la figura 7.13.

Figura 7.12. Comportamiento contador ascendente. Fuente: Autor.

El valor de los contadores es leído cada 50hz determinando la diferencia entre las medidas para calcular la velocidad. Si la diferencia entre medidas es mayor a 10.000 o menor a -10.000 indica que hubo un reinicio de los contadores, por lo tanto, el ultima valor obtenido debe ser

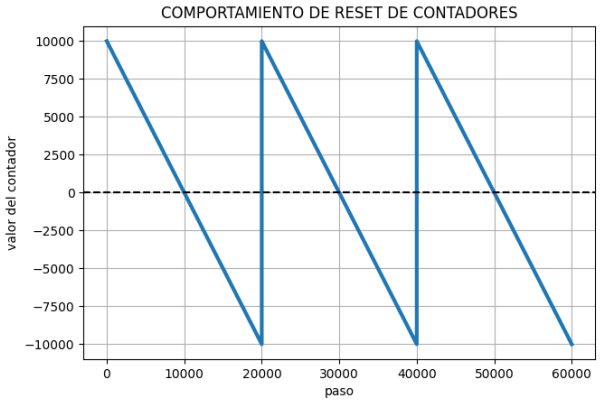

Figura 7.13. Comportamiento contador descendente. Fuente: Autor.

modificado sumando o restando 10.000. Sabiendo que cada encoder genera 3200 pulsos por revolución, se realiza una regla de tres para determinar la velocidad de cada llanta en radianes por segundo.

Para controlar la velocidad de los motores se diseña un controlador PID. En primer lugar, se determina la función de transferencia de los motores mediante el método de identificación de Ziegler Nichols [74]. La ecuación 7.1 muestra la función de transferencia determinada.

$$\frac{30}{0{,}06S + 1} \tag{7.1}$$

Se hace uso del toolbox de Simulink "Discrete PID Controller" para determinar un controlador apropiado para el sistema, encontrando que un controlador PI brinda buenos resultados. La ecuación en diferencias y constantes del controlador se presentan en la ecuación 7.2.

$$\begin{gathered} Y(n) - Y(n-1) = R(n) K_p - R(n-1) \cdot \left(K_p + K_i T_s\right) \\ K_p = 0{,}016 \\ K_i = 0{,}274 \\ T_s = 0{,}02 \end{gathered} \tag{7.2}$$

El controlador PI se exporta en lenguaje C para ser incorporado al firmware en la tarjeta Discovery, con este es posible controlar la velocidad individual de cada motor.

7.3.2. Estructura de software.

El sistema de software se divide en 2, el software que se ejecuta al interior del robot y el que se ejecuta remotamente. Al interior del robot, el software se ejecuta en las tarjetas STMDiscovery y UP SQUARED, estas dos tarjetas se comunican mediante comunicación serial. El software que se ejecuta de forma remota se comunica mediante ROS con el robot, por lo tanto, puede ser ejecutado en cualquier dispositivo computacional con la capacidad de funcionar con ROS.

La figura 7.14 muestra los diferentes componentes de software, la forma en la que estos se comunican y el hardware en el que se ejecutan.

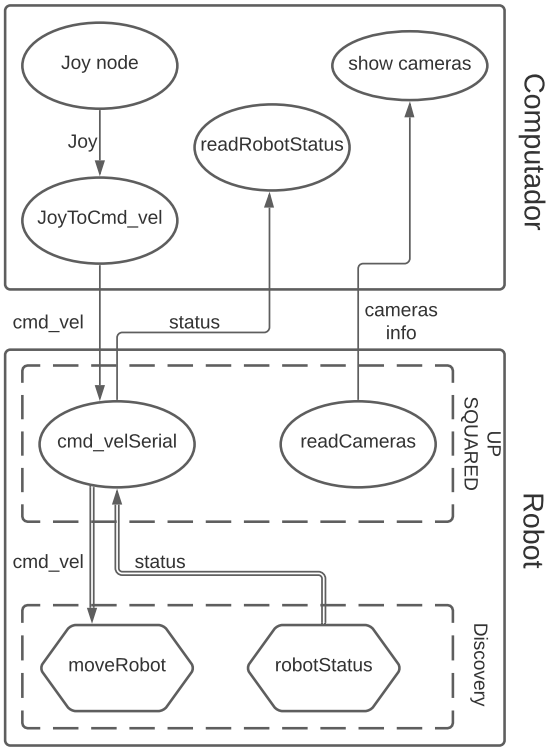

Figura 7.14. Diagrama esquemático del funcionamiento hardware-software.

Donde se muestran los componentes de software que se ejecutan en el robot y remotamente. Los contornos ovalados representan paquetes de

ROS y los contornos hexagonales programas locales. Las líneas dobles representan comunicación vía serial y las lineas sencillas mediante mensajes de ROS.

Remotamente se ejecutan 4 paquetes de ROS:

- **Joy_node:** Lee la información de un joystick y la publica como una matriz mediante el mensaje Joy de ROS.
- **JoyToCmd_vel:** Lee el mensaje Joy y determina una velocidad objetivo a partir de la información recibida, esta velocidad objetivo es publicada como un mensaje cmd_vel que contiene las velocidades lineales y angulares en cada eje.
- **show_cameras:** Lee la información de las cámaras y la muestra en pantalla en tiempo real a través del programa Rviz.
- **readRobotStatus:** Lee la información de posición y velocidad actual del robot proveniente del mensaje "Odom".

En el robot se ejecutan 2 paquetes de ROS, la tarjeta UP SQUARED es la encargada de ejecutar estos paquetes:

- **Cmd_velSerial:** Lee la velocidad objetivo del robot proveniente del mensaje cmd_vel y la envía mediante comunicación serial a la tarjeta Discovery, también recibe de la tarjeta Discovery el estado actual del robot y la publica a través del mensaje "Odom".
- **readCameras:** Lee la información de las cámaras conectadas a la UP SQUARED y la publica en ROS para que pueda ser leída remotamente.

En la tarjeta Discovery se ejecutan 2 programas locales:

- **MoveRobot:** Lee el puerto serial obteniendo la velocidad objetivo del robot, calcula la cinemática y envía la velocidad objetivo a cada controlador de motor.
- **RobotStatus:** Calcula la odometría del robot y la envía vía serial.

Capítulo 8

Cálculo e implementación de la cinemática y odometría

CINEMÁTICA

El cálculo de la cinemática y odometría del robot son muy útiles para determinar los movimientos y posición del mismo. Para realizar estos cálculos se parte de la configuración del robot, teniendo en cuenta principalmente el tipo, tamaño y ubicación de las llantas con respecto al robot. La figura 8.1 muestra la configuración del robot diseñado.

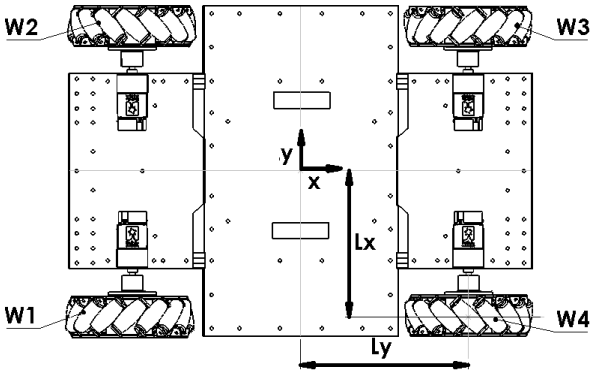

Figura 8.1. Descripción ejes para la cinemática. Fuente: Autor.

La cinemática inversa nos permite determinar la velocidad a la cual se

debe mover cada llanta para que el robot se mueva a la velocidad objetivo, la ecuación 8.1 muestra el cálculo final de la cinemática inversa. El cálculo completo del modelo para un robot con llantas mecanum es mostrado en [75] y [61].

$$\begin{aligned} W_1 &= \tfrac{1}{r}\left(v_x - v_y - \left(l_x + l_y\right)\omega\right) \\ W_2 &= \tfrac{1}{r}\left(v_x + v_y + \left(l_x + l_y\right)\omega\right) \\ W_3 &= \tfrac{1}{r}\left(v_x + v_y - \left(l_x + l_y\right)\omega\right) \\ W_4 &= \tfrac{1}{r}\left(v_x - v_y + \left(l_x + l_y\right)\omega\right) \end{aligned} \tag{8.1}$$

Donde v_x es la velocidad en x, v_y es la velocidad en y y ω es la velocidad rotacional. Los términos l_x, l_y se refieren a la distancia desde el centro de las ruedas al centro del robot en los ejes x y y. r se refiere al radio de las ruedas, estas medidas pueden ser vistas gráficamente en la descripción de la figura 8.1.

La cinemática directa nos permite estimar la velocidad real a la que se está moviendo el robot, a partir de la velocidad de cada llanta. La ecuación 8.2 muestra el cálculo de la cinemática directa para determinar la velocidad angular y las velocidades lineales en X y Y del robot.

$$\begin{aligned} v_x\left(t\right) &= \left(W_1 + W_2 + W_3 + W_4\right)\tfrac{r}{4} \\ v_y\left(t\right) &= \left(-W_1 + W_2 + W_3 - W_4\right)\tfrac{r}{4} \\ \omega\left(t\right) &= \left(-W_1 + W_2 - W_3 + W_4\right)\tfrac{r}{4\left(l_x + l_y\right)} \end{aligned} \tag{8.2}$$

Con el cálculo de la cinemática inversa, determinamos la velocidad objetivo de cada llanta para enviarla al controlador PID, mientras, la cinemática directa toma la velocidad medida de cada llanta por los encoders y determina la velocidad a la que se está moviendo el robot. Teniendo el cálculo de ambas cinemáticas podemos pasar a calcular la odometría.

La odometría nos permite estimar la posición actual del robot con respecto al punto de partida. En este caso la odometría del robot es medida a partir de la cinemática directa, tomando la velocidad entregada por esta y multiplicándola por el tiempo de refresco de la medida, para estimar la odometría del robot.

El cálculo de la cinemática y la odometría son programados en la tarjeta Discovery y como se menciona en la sección 7.3.1, estos cálculos hacen parte del firmware del robot. Para la cinemática inversa se programan las ecuaciones tomando como variables de entrada las velocidades objetivo del robot (v_x, v_y , ω). Para la cinemática directa se toman como variables de entrada la velocidad medida de cada llanta (Wn). La odometría es medida a partir de la cinemática directa, se toma el cálculo de la cinemática directa y se multiplica por la frecuencia con la que esta se recibe (0.2 ms), de esta forma pasamos de la velocidad de la cinemática directa a la distancia de la odometría.

Se tomó la decisión de programar estos cálculos en la tarjeta Discovery para reducir la cantidad de cálculos que ejecuta el sistema de cómputo.

Capítulo 9

Implementación de algoritmos de mapeo y localización

Para que el robot sea capaz de conocer el entorno en el que se encuentra y ubicarse en él, se integra un algoritmo de localización y mapeo simultáneo en el robot. Existen múltiples algoritmos de localización y mapeo simultáneos, en el robot se implementa RTAB-Map ya que es un algoritmo de fácil implementación, el robot cuenta con los sensores necesarios para ejecutar este algoritmo, el algoritmo suministra el mapa 2D y 3D del entorno, y cuenta con múltiples parámetros que permiten ajustar la precisión del mapa.

RTAB-Map se puede ejecutar con tan solo una cámara de profundidad que le suministre la información visual y de profundidad del entorno. Para esto el robot está equipado con la cámara realsense D435 que brinda la información RGB y de profundidad de cada píxel. La figura 9.1 muestra la imagen suministrada por la cámara de profundidad vista en Rviz.

El algoritmo crea tres memorias, la memoria de trabajo (WM), la memoria de corto plazo (STM), y la memoria de largo plazo (LTM). El algoritmo funciona combinando la información de la cámara y la odometria como una tupla, para ser almacenada en la STM. La STM almacena la información hasta que se alcanza su capacidad máxima, una

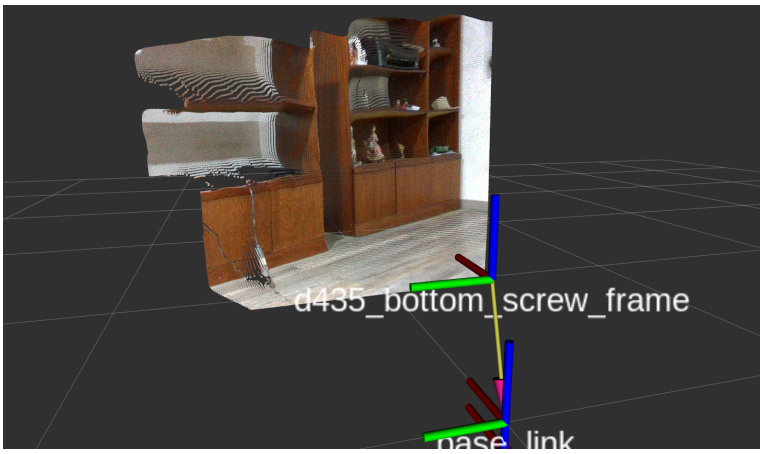

Figura 9.1. Imagen de la cámara de profundidad en Rviz. Fuente: Autor.

vez se alcanza esta capacidad, los registros más antiguos van pasando a la WM, de forma que en la STM siempre se encuentran los registros más recientes. El tamaño de la STM es determinado por la velocidad a la que se mueve el robot y con qué frecuencia ingresa un nuevo dato.

Las características de cada imagen son extraídas mediante un algoritmo de "bolsa de palabras visuales". Las características de la imagen actual son comparadas con las características de las imágenes en la WM, de esta forma se localiza el robot y se detectan los cierres de bucle. La STM no es utilizada para detectar cierres de bucle ya que se pueden presentar inconsistencias al ver imágenes consecutivas.

Una vez el tamaño de la WM hace que el tiempo de procesamiento para encontrar coincidencias, sea mayor a un umbral, las localizaciones con menos características pasan de la WM a la LTM. Dejando en la WM solo las imágenes con más probabilidad de generar una localización. Cuando se detecta una localización con la WM, las imágenes vecinas son extraídas de la LTM para ayudar a mejorar la localización.

Para que el algoritmo de RTAB-Map brinde buenos resultados es importante que la posición estimada del robot sea muy cercana a la real. Para esto no basta con la odometría de las llantas (odometría medida) ya que esta odometría puede presentar inconsistencias con respecto a la

posición real del robot, estas inconsistencias se pueden presentar porque alguna llanta pierda fricción, el terreno sea desigual o haya algún obstáculo que interrumpa el correcto movimiento del robot. Para mejorar la odometría de las llantas y obtener con mayor precisión la posición real, se une esta odometría con la odometría suministrada por la cámara T265. Esta cámara cuenta con giroscopio, acelerómetro y dos lentes de ojo de pez, que combinados mediante inteligencia artificial proporcionan una estimación de odometría muy cercana a la real y al unirse con la odometría de las llantas se perfecciona aún más (odometría mejorada). La figura 9.2 muestra visualmente la posición estimada del robot con respecto al punto de origen, mediante la odometría mejorada.

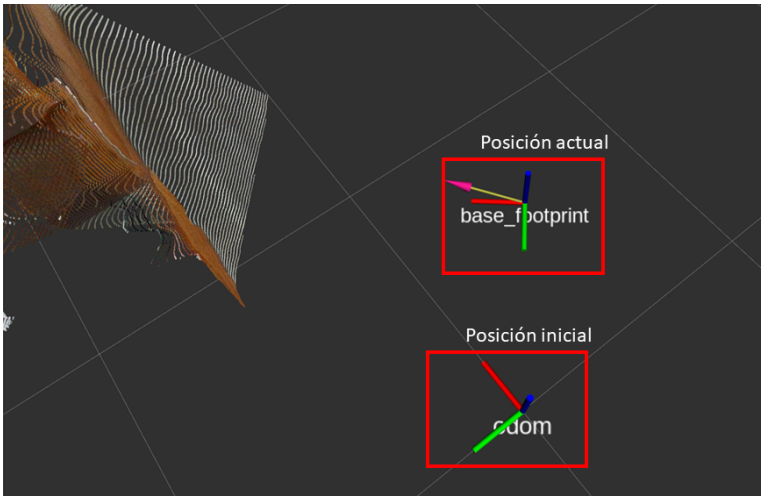

Figura 9.2. Posición actual y posición inicial vistas en Rviz. Fuente: Autor.

El paquete RTAB-Map de ROS guarda la información del mapeo en una base de datos como "rtabmap.db", de forma que si se desea crear el mapa de una nueva ubicación se genera un archivo nuevo y si se desea continuar con un mapa anterior se carga el mismo. El algoritmo es capaz de detectar el cierre de bucle y realizar corrección en el mapa en tiempo real. La cámara realsense D435 proporciona la información de imagen y profundidad utilizada para crear el mapa 3D, este mapa es proyecto al

suelo para crear el mapa 2D. El proceso de mapeo puede ser visto en el enlace [1].

La figura 9.3 muestra el resultado final de un mapa en 2D y la figura 9.4 del mapa en 3D.

Figura 9.3. Mapa en 2D. Fuente: Autor.

Figura 9.4. Mapa 3D. Fuente: Autor.

Para la navegación del robot se utiliza el generador de trayectoria de banda elástica cronometrada "TEB Planner", este algoritmo se presenta

[1] Prueba de mapeo: https://www.youtube.com/watch?v=1MHaJJRSefk

como un avance al generador de trayectoria “EB Planner”. EB planner realiza la discretización del mapa y genera un conjunto finito de poses para cumplir la trayectoria y TEB Planner añade el tiempo de ejecución de esa pose para obtener una mejor trayectoria.

Cuenta con múltiples parámetros que se configuraron, como: La evasión de obstáculos, ancho de los muros, radio de giro, multiplicador de costo, aceleración máxima del robot, velocidad máxima, velocidad en reversa, espacio del robot.

Adicionalmente, es posible modificar la heurística para la generación de la trayectoria. Para lanzar la navegación se debe lanzar primero la localización, de esta forma el robot conoce su ubicación y el entorno, siendo capaz de determinar una mejor trayectoria. La posición objetivo puede ser posicionada directamente desde Rviz. La figura 9.5 muestra al robot en Rviz moviéndose hacia la posición objetivo, el proceso puede ser visto a detalle en el siguiente enlace [2]. El repositorio completo con todos los paquetes de ROS y sus instrucciones de ejecución pueden ser visto en el siguiente enlace [3] .

Se configuraron los siguientes parámetros: La evasión de obstáculos, ancho de los muros, el radio de giro , el multiplicador de costo, la aceleración se fijo, la aceleración angular, la velocidad máxima, la velocidad en reversa, espacio del robot.

En la figura 9.6 se muestra el flujo de ejecución para el mapeo, la localización y navegación. Primero se debe lanzar el nodo maestro, luego se lanzan los nodos necesarios para leer un joystick conectado mediante bluetooth, convierte sus botones en una velocidad objetivo ”*cmd_vel*” y la envía por serial al hardware del robot para que este ejecute sus movimientos. Luego se ejecuta la cámara de profundidad y la cámara de seguimiento. Ya con estos nodos lanzados se procede a ejecutar el mapeo, luego de tener el mapa se ejecuta la localización y por último la navegación.

[2]Prueba de navegación: https://www.youtube.com/watch?v=SxnY7qiEfQA

[3]Repositorio: https://github.com/BrayanPallares/autonomousMobileRobot_pkg

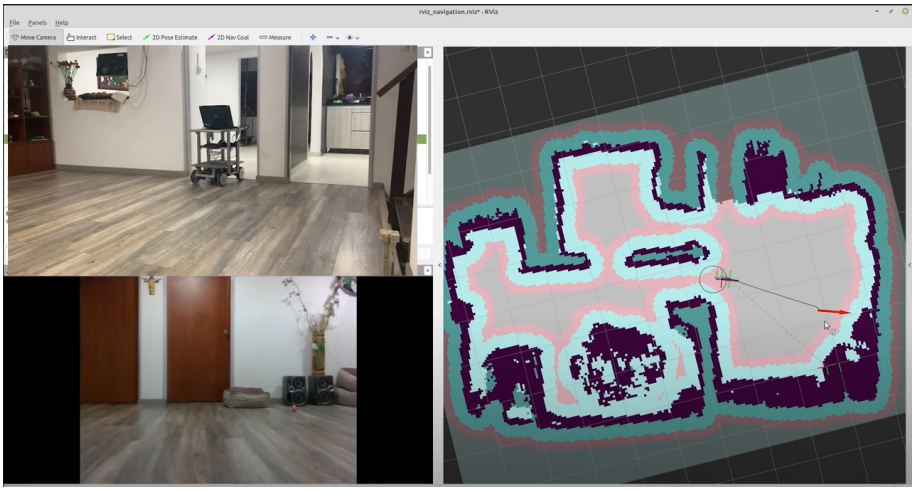

Figura 9.5. Robot navegando. Fuente: Autor.

Lanzar ROS	•Roscore
Lanzar movimiento manual del robot	•roslaunch robot_pkg launchJoyCmdvel.launch •roslaunch robot_pkg launchRobotJoy.launch
Lanzar cámara de profundidad	•roslaunch camera_pkg camera_d435.launch
Lanzar cámara de seguimiento	•roslaunch camera_pkg camera_t265.launch
Lanzar mapeo	•roslaunch rtabmap_pkg mapping.launch
Lanzar localización	•roslaunch rtabmap_pkg localization.launch
Lanzar Navegación	•roslaunch navigation_pkg teb_planner.launch

Figura 9.6. Flujo de ejecución.

Capítulo 10

Ejecución de pruebas y resultados

El robot se construye siguiendo el diseño realizado, el proceso de construcción puede verse con más detalle en el anexo C. La figura 10.1 muestra el diseño final del robot y su construcción.

Figura 10.1. Robot diseñado y robot real. Fuente: Autor.

Se realizan múltiples pruebas para comprobar el correcto

funcionamiento de los componentes del robot. En primer lugar, se comprueba que la velocidad medida por los encoders sea cercana a la real, para esto se hace girar los motores a velocidad constante, la velocidad real de los encoders es medida por un tacómetro digital y comparada con la medida de los encoders. La tabla 10.1 muestra la comparación de ambas medidas en las 4 llantas.

	Medida referencia (Tacómetro digital) [RPM]	Medida por programa (Encoder) [RPM]	Error[%]
W1	4,964	5,007	0,864
	30,434	30,264	0,559
W2	5,005	5,033	0,554
	30,104	30,305	0,668
W3	4,998	5,036	0,748
	29,992	30,168	0,586
W4	4,976	5,002	0,514
	29,981	30,105	0,414

Cuadro 10.1. Prueba encoders.

Para validar el controlador PID, se utiliza una velocidad objetivo variable y se observa la respuesta de los motores. La figura 10.2 muestra la velocidad objetivo de los motores en rojo y la velocidad medida de cada motor en cían, azul, verde y negro.

Comprobando el correcto funcionamiento de los motores, encoders, controladores PI y demás elementos. Con el controlador diseñado se obtuvo un tiempo de respuesta de 118ms y un tiempo de estabilización de 390ms.

Para comprobar el cálculo de la cinemática directa e inversa del robot, se genera una secuencia de velocidades del robot. Estas velocidades se utilizan para determinar la velocidad de cada llanta mediante la cinemática inversa, la cinemática directa realiza el cálculo inverso, determinando las velocidades del robot a partir de la velocidad de cada llanta. De esta forma hacemos que el robot se mueva a determinada

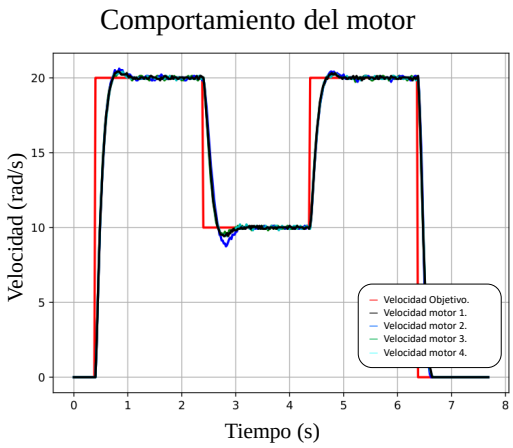

Figura 10.2. Respuesta del motor al cambio de velocidad.

velocidad mediante la cinemática inversa y comprobamos que se esté moviendo a esta velocidad mediante la cinemática directa. La figura 10.3 muestra 2 pruebas realizadas, donde la línea roja representa la velocidad enviada a la cinemática inversa (velocidad objetivo) y la línea verde representa la velocidad medida por la cinemática directa. la primera prueba, se realizó enviando la misma velocidad objetivo para cada una de las velocidades en simultaneo, y la segunda enviando una a la vez.

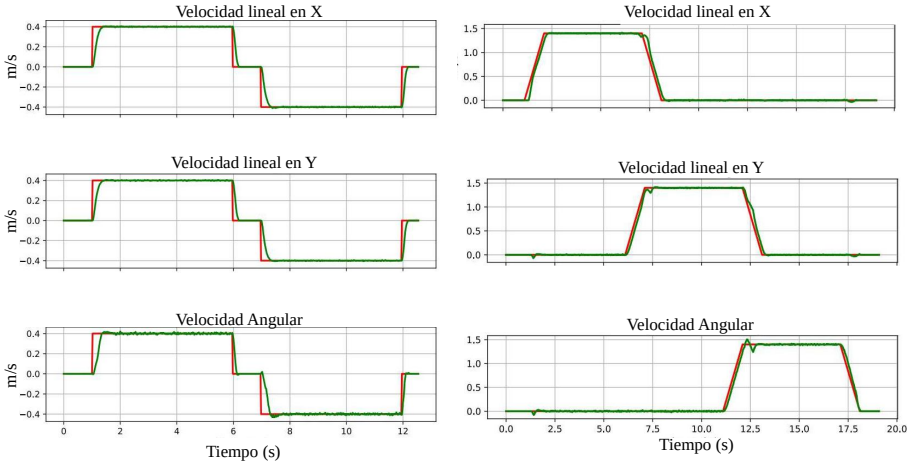

Figura 10.3. Velocidad objetivo y real del robot. Fuente: Autor.

Con esta prueba se evidencia el correcto funcionamiento de las cinemáticas, ya que la velocidad objetivo de la cinemática inversa es muy cercana a la velocidad medida por la cinemática directa.

La odometría medida se comprueba realizando trayectorias con el robot y comparando la trayectoria real con la medida. La trayectoria medida hace referencia a la odometría medida a partir de la velocidad de las llantas con la cinemática directa y la trayectoria real hace referencia a la odometría entregada por la cámara de seguimiento T265. La figura 10.4 muestra la posición del robot al realizar 2 trayectorias, una trayectoria circular y una cuadrada. El color verde representa la trayectoria objetivo, el color azul representa la trayectoria medida y el color rojo representa la trayectoria real del robot.

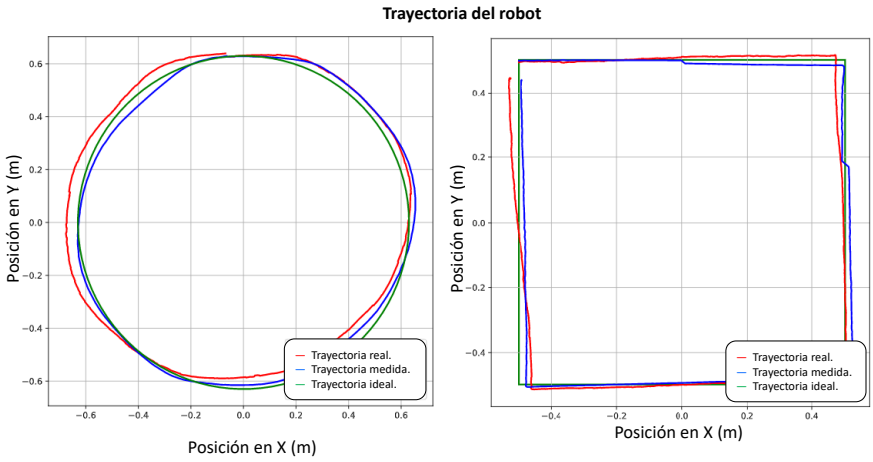

Figura 10.4. Trayectorias ejecutadas por el robot. Fuente: Autor.

Se calcula la diferencia promedio entre las trayectorias tomando la distancia euclidiana de punto a punto. Esta prueba nos muestra que la diferencia promedio entre la trayectoria real y la ideal es de 0.052 metros, y entre la trayectoria real y la medida es de 0.0325. La tabla 10.2 muestra esta medida y la correspondiente para la trayectoria circular y cuadrada.

	Trayectoria real vs Trayectoria ideal [m]	Trayectoria real vs Trayectoria medida [m]
Trayectoria circular	0,066	0,043
Trayectoria cuadrada	0,038	0,022
Promedio	0,052	0,033

Cuadro 10.2. Error promedio de las trayectorias.

Capítulo 11

Evaluación del sistema

Se probaron diferentes parámetros en los algoritmos: En el algoritmo de mapeo se limitó el sensor D435 a tomar la nube de puntos en 2m, ya que si se deja por defecto queda en 10m y esto puedo ocasionar ruido en el mapa 2D. En el algoritmo de navegación se probaron los planeadores DWA, EBand, TEB y el de ROS, se evidencio que el que mejor generaba ruta para robots omnidireccionales era el algoritmo TEB.

La evaluación del sistema se basa en el análisis de la precisión de los algoritmos de localización y mapeo, se realizan 2 pruebas, primero, se evalúa la precisión en el mapeo, para esto se realiza el mapa de una planta y se toma la medida del largo y ancho de las zonas, comparando la medida en el mapa creado y en el espacio real. La tabla 11.1 muestra las distancias medidas en metros y el entre las medidas. La figura 11.1 muestra las medidas del mapa creado en rojo y del entorno real en azul.

	Mapeado		Real		Error	
Espacio	Ancho	Largo	Ancho	Largo	Ancho	Largo
Sala	4,21	3,74	4,37	3,83	3,66 %	2,35 %
Comedor	3,47	3,63	3,63	3,92	4,41 %	7,40 %
Cocina	2,33	2,26	2,43	2,33	4,12 %	3,00 %
Lago espacio	6,12	9,86	6,19	9,66	1,13 %	2,07 %
Promedio error					3,33 %	3,71 %

Cuadro 11.1. Medidas del entorno mapeado y real.

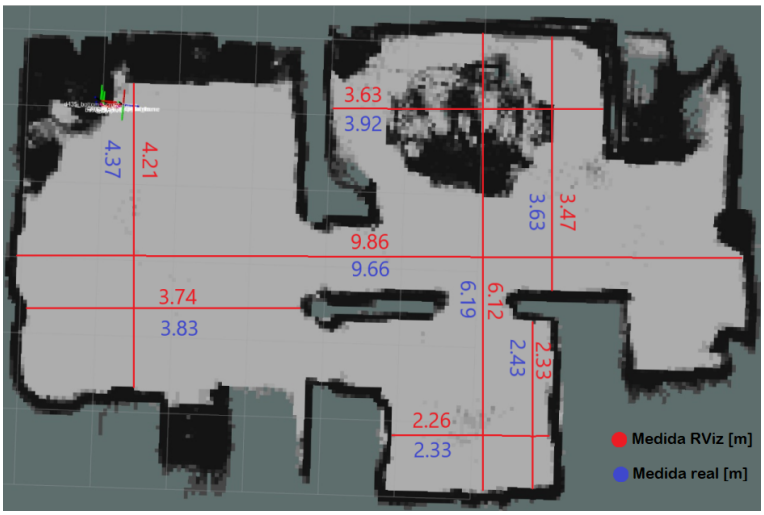

Figura 11.1. Mapa con las medidas en RViz y las medidas reales. Fuente: Autor.

La segunda prueba consiste en lanzar RTAB-Map en modo de localización, en diferentes ubicaciones para tomar medidas de la localización en el mapa y la localización en el espacio real, las medidas son tomadas midiendo la distancia del robot a las paredes en metros. La tabla 11.2 muestra los resultados obtenidos.

Donde se muestra la posición estimada en el mapa, y la posición real del robot en X y Y. Las posiciones son tomadas midiendo la distancia en metros del robot a las paredes más cercanas. La columna de error muestra el error porcentual en cada eje y la columna "d_E"muestra la distancia euclidiana entre los puntos (error en metros).

	Mapa		Real		Error		d_E
	x (m)	y (m)	x (m)	y (m)	x (m)	y (m)	
Medida 1	1,36	2,12	1,42	2,24	4,23 %	5,36 %	0,13
Medida 2	2,63	1,02	2,65	1,00	0,75 %	2,00 %	0,03
Medida 3	1,22	1,26	1,20	1,28	1,67 %	1,56 %	0,03
Medida 4	2,64	3,72	2,85	3,80	7,37 %	2,11 %	0,22
Medida 5	2,52	3,49	2,44	3,32	3,28 %	5,12 %	0,19
Medida 6	2,91	3,89	3,02	3,82	3,67 %	2,04 %	0,14
Medida 7	3,06	4,40	3,11	4,31	1,67 %	1,99 %	0,10
Medida 8	3,30	4,90	3,45	4,81	4,43 %	1,96 %	0,18
Medida 9	3,54	5,11	3,38	5,40	4,92 %	5,51 %	0,34
Medida 10	3,79	5,91	3,68	5,72	2,80 %	3,32 %	0,22
Promedio					3,45 %	3,20 %	0,16

Cuadro 11.2. Medidas x y y de la localización y real.

Conclusiones

Se diseñó y construyó un robot móvil omnidireccional que permite controlar de forma independiente la velocidad de cada llanta. El robot diseñado es de fácil construcción, ya que este hecho con materiales de fácil obtención es omnidireccional ya que cuenta con llantas Mecanum y controla la velocidad de cada llanta mediante un controlador PID.

Se realizan pruebas para comprobar que el robot sea capaz de controlar la velocidad de cada llanta, primero, se prueba que el robot sea capaz de medir la velocidad de cada llanta, encontrando que el error en la medida es de menos del 1 % para cualquier velocidad y cualquier llanta, segundo, se comprueba el controlador PID comparando la velocidad objetivo con la real, en esta prueba se encontró que los motores cuentan en promedio con un tiempo de respuesta de 118ms y un tiempo de establecimiento de 390ms.

Se realizó el cálculo de la cinemática y odometría del robot en el sistema de cómputo abordo de este. Se comprobó el correcto cálculo de las cinemáticas al ejecutar movimientos con la cinemática inversa y comprobarlos con la cinemática directa. La odometría se verificó al ejecutar trayectorias y medir la diferencia entre la posición real y la medida, encontrando que la diferencia en promedio entre ambas es de 0.033 metros.

Se implementaron los algoritmos de mapeo, localización y navegación, en el sistema de cómputo abordo del robot. Se realizaron pruebas para verificar su exactitud, encontrando que estos algoritmos funcionan correctamente, presentando un error en el mapeo inferior al

4 %.

El error en la precisión de los algoritmos de mapeo, localización y navegación se puede deber a las zonas con pocas características en el entorno, ya que el robot presenta dificultades al localizarse en estas zonas. La precisión puede ser mejorada al agregar un sensor Lidar de 360 grados que ayude en la localización sin depender de las características de la imagen.

El robot construido es de fácil implementación y programación ya que utiliza tecnología estándar (ROS). Los planos de diseño y el repositorio de software son presentados para que el robot sea fácil de operar, replicar y modificar.

Apéndice A

PCB - Visor Gerber

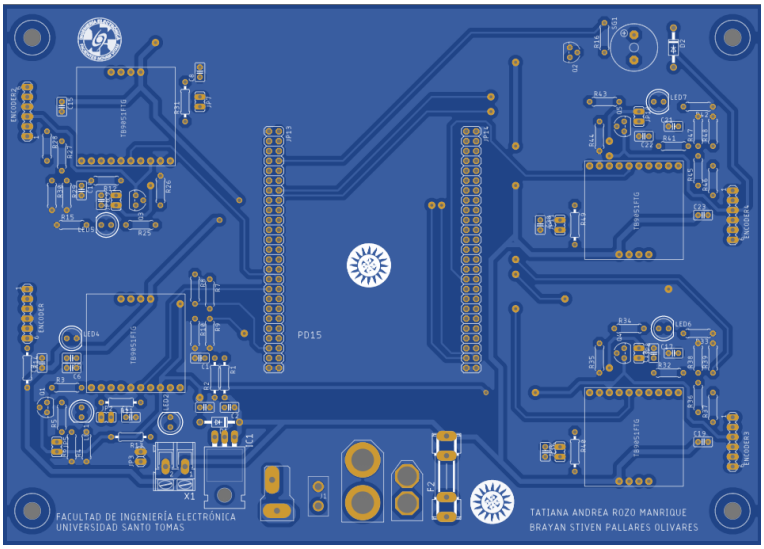

Figura A.1. Visor Gerber de la capa TOP. Fuente: Autor.

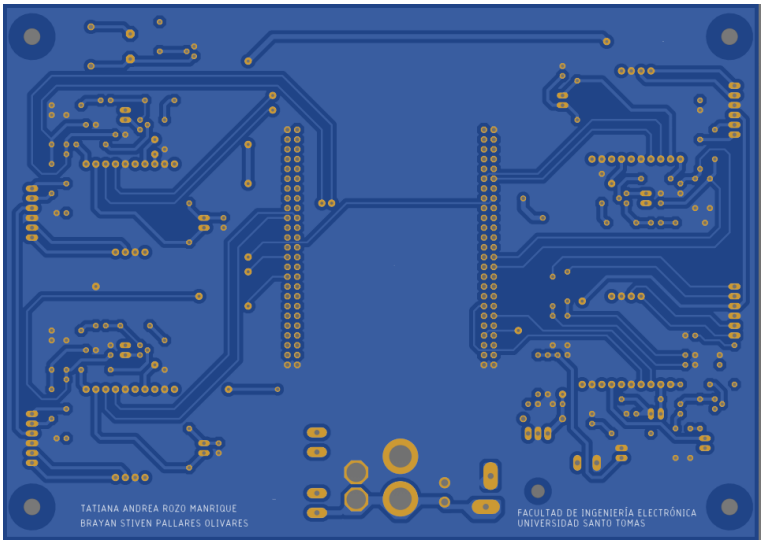

Figura A.2. Visor Gerber de la capa BOT. Fuente: Autor.

Apéndice B

Diseño del robot

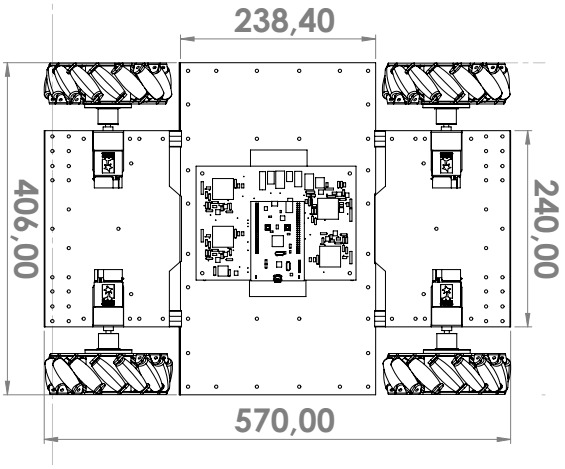

Figura B.1. Plano del robot con un nivel. Fuente: Autor.

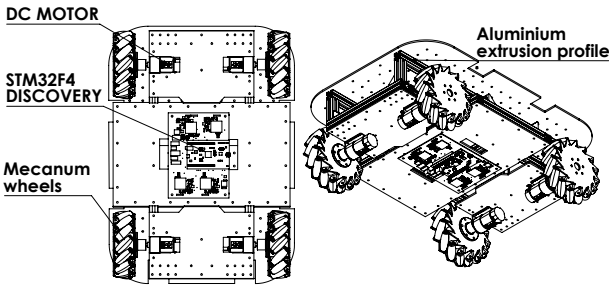

Figura B.2. Plano 3D del robot con tres niveles. Fuente: Autor.

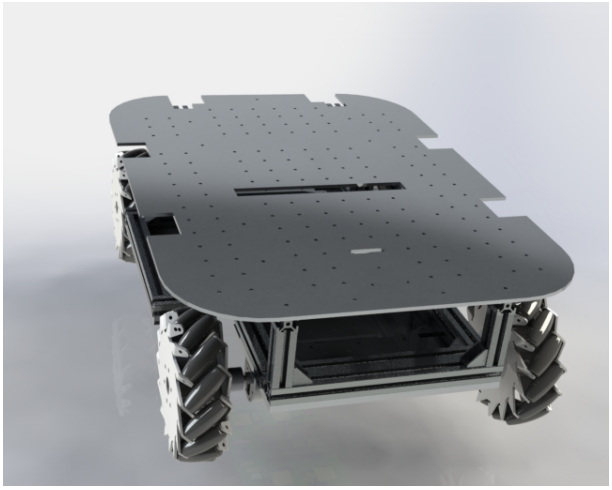

Figura B.3. Modelo 3D del robot con 3 niveles. Fuente: Autor.

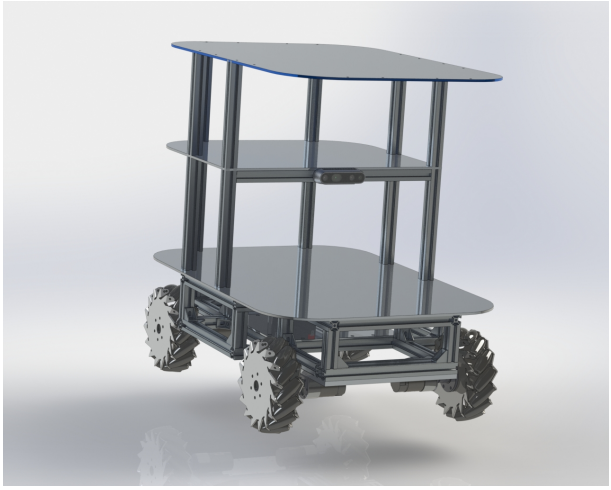

Figura B.4. Modelo 3D del robot final. Fuente: Autor.

Apéndice C

Proceso de construcción

Figura C.1. Primera versión del robot. Fuente: Autor.

Figura C.2. Primera versión del robot y electrónica. Fuente: Autor.

Figura C.3. Comparación llantas versión 1 y versión 2. Fuente: Autor.

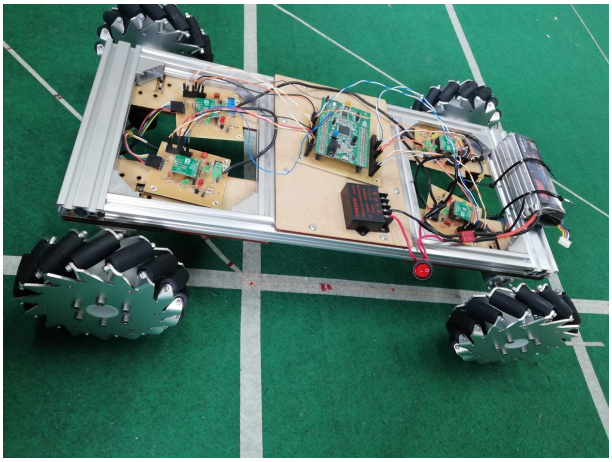

Figura C.4. Robot con llantas de la segunda versión. Fuente: Autor.

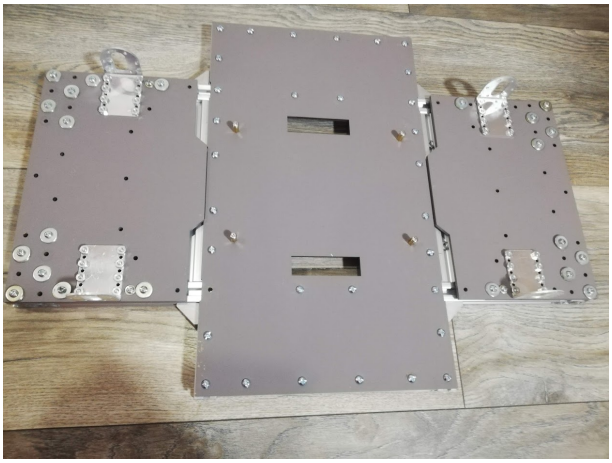

Figura C.5. Primera pieza segunda versión. Fuente: Autor.

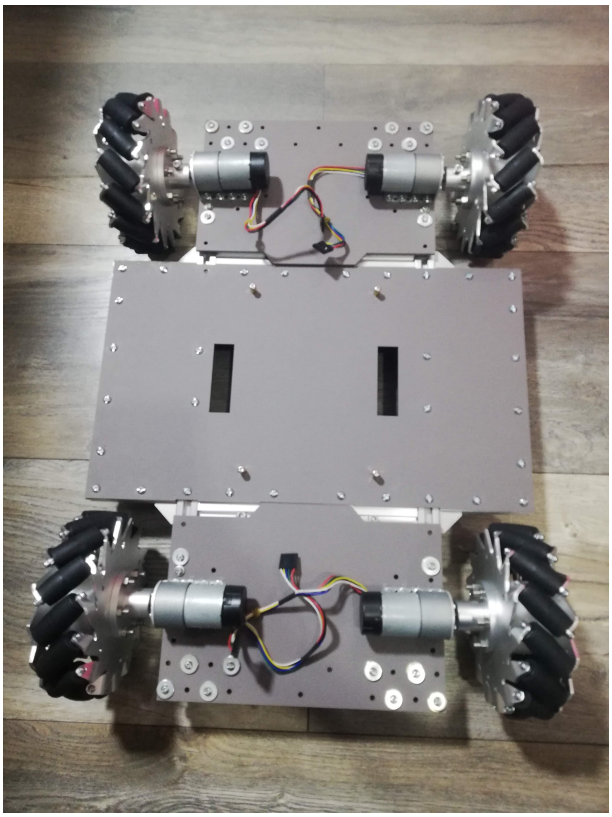

Figura C.6. Segunda versión con llantas. Fuente: Autor.

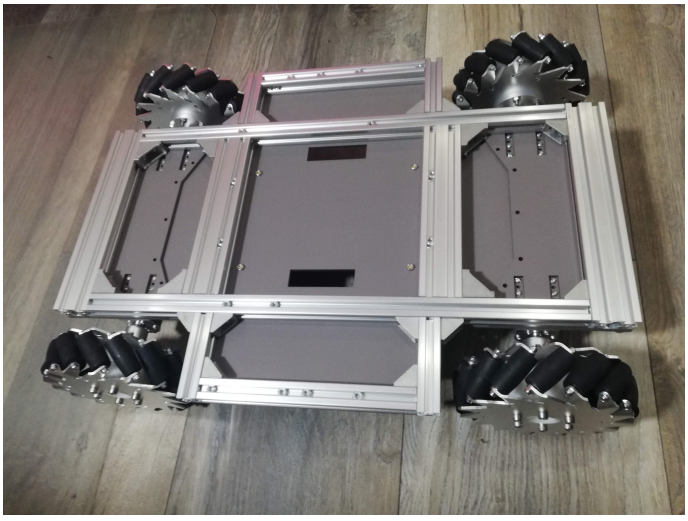

Figura C.7. Vista superior del robot segunda versión. Fuente: Autor.

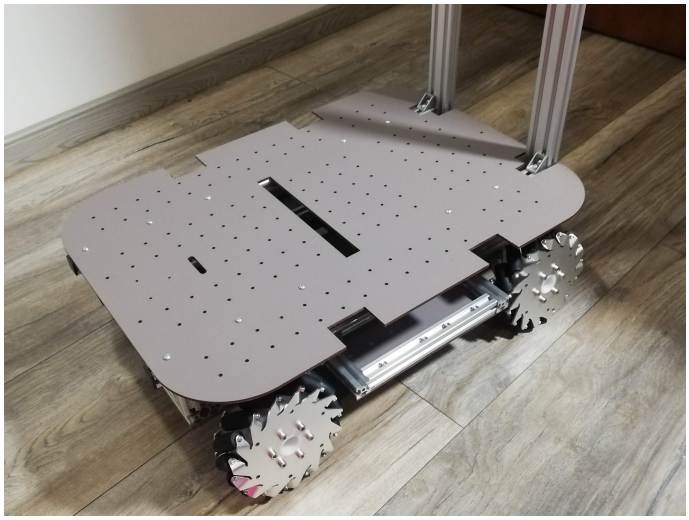

Figura C.8. Vista superior del robot con 3 niveles. Fuente: Autor.

Figura C.9. Robot final construido. Fuente: Autor.

Bibliografía

[1] Carlos Quintero, Saith Rodríguez, Katherín Pérez, Jorge López, Eyberth Rojas, and Juan Calderón. Learning soccer drills for the small size league of robocup. In *Robot Soccer World Cup*, pages 395–406. Springer, 2014.

[2] Saith Rodríguez, Eyberth Rojas, Katherín Pérez, Jorge López, Carlos Quintero, and Juan Calderón. Fast path planning algorithm for the robocup small size league. In *Robot Soccer World Cup*, pages 407–418. Springer, 2014.

[3] Gustavo A Cardona, Wilfrido Moreno, Alfredo Weitzenfeld, and Juan M Calderon. Reduction of impact force in falling robots using variable stiffness. In *SoutheastCon 2016*, pages 1–6. IEEE, 2016.

[4] Ercan Elibol, Juan Calderon, Martin Llofriu, Carlos Quintero, Wilfrido Moreno, and Alfredo Weitzenfeld. Power usage reduction of humanoid standing process using q-learning. In *Robot Soccer World Cup*, pages 251–263. Springer, 2015.

[5] Ercan Elibol, Juan Calderon, Martin Llofriu, Wilfrido Moreno, and Alfredo Weitzenfeld. Analyzing and reducing energy usage in a humanoid robot during standing up and sitting down tasks. *International Journal of Humanoid Robotics*, 13(04):1650014, 2016.

[6] Juan Calderon, Gustavo A Cardona, Martin Llofriu, Muhaimen Shamsi, Fallon Williams, Wilfrido Moreno, and Alfredo Weitzenfeld. Impact force reduction using variable stiffness with an optimal

approach for falling robots. In *Robot World Cup*, pages 404–415. Springer, 2016.

[7] Juan M Calderon, Eyberth R Rojas, Saith Rodriguez, Heyson R Baez, and Jorge A Lopez. A robot soccer team as a strategy to develop educational iniciatives. In *Latin American and Caribbean Conference for Engineering and Technology, Panama City, Panama*, 2012.

[8] Jose Guillermo Guarnizo Marin, Glen Camilo Ortega Díaz, Andrés Felipe Téllez Rodríguez, and Édgar Camilo Camacho Poveda. Entorno pedagógico para la enseñanza en básica primaria mediante el uso de sistema robótico comercial. *Ingeniería*, 26(1), 2021.

[9] Heyson Báez, Katherin Perez, Eyberth Rojas, Saith Rodriguez, Jorge López, Carlos Quintero, and Juan Manuel Calderón. Application of an educational strategy based on a soccer robotic platform. In *2013 16th International Conference on Advanced Robotics (ICAR)*, pages 1–6. IEEE, 2013.

[10] Carolina Higuera, Fernando Lozano, Edgar Camilo Camacho, and Carlos Hernando Higuera. Multiagent reinforcement learning applied to traffic light signal control. In *International Conference on Practical Applications of Agents and Multi-Agent Systems*, pages 115–126. Springer, 2019.

[11] Wilson O Quesada, Jonathan I Rodriguez, Juan C Murillo, Gustavo A Cardona, David Yanguas-Rojas, Luis G Jaimes, and Juan M Calderón. Leader-follower formation for uav robot swarm based on fuzzy logic theory. In *International Conference on Artificial Intelligence and Soft Computing*, pages 740–751. Springer, 2018.

[12] Jose León, Gustavo A Cardona, Andres Botello, and Juan M Calderón. Robot swarms theory applicable to seek and rescue operation. In *International Conference on Intelligent Systems Design and Applications*, pages 1061–1070. Springer, 2016.

[13] Juan Calderon, Alexa Obando, and Diego Jaimes. Road detection algorithm for an autonomous ugv based on monocular vision. In *Electronics, Robotics and Automotive Mechanics Conference (CERMA 2007)*, pages 253–259. IEEE, 2007.

[14] Jose Guillermo Guarnizo and Luis Fernando Niño. Clonal selection algorithm applied to object recognition in mobile robots. In *MLDM (1)*, pages 49–62, 2019.

[15] Sindy Amaya and Armando Mateus. Tasks allocation for rescue robotics: a replicator dynamics approach. In *International Conference on Artificial Intelligence and Soft Computing*, pages 609–621. Springer, 2019.

[16] Yeison Suarez, Carolina Higuera, and Edgar Camilo Camacho. Inverse reinforcement learning application for discrete and continuous environments. In *International Conference on Advanced Engineering Theory and Applications*, pages 345–355. Springer, 2019.

[17] P Nuñez, P Bustos, E Jaramillo, Pilar Bachiller, and Ismael García-Varea. Robots sociales para la mejora de la calidad de vida de las personas dependientes. In *VI Congreso Iberoamericano de Tecnologías de Apoyo a la Discapacidad IBERDISCAP*, volume 1, pages 94–103, 2011.

[18] Wolfram Burgard, Armin B Cremers, Dieter Fox, Dirk Hähnel, Gerhard Lakemeyer, Dirk Schulz, Walter Steiner, and Sebastian Thrun. Experiences with an interactive museum tour-guide robot. *Artificial intelligence*, 114(1-2):3–55, 1999.

[19] Mikael Svenstrup, Thomas Bak, and Hans Jørgen Andersen. Trajectory planning for robots in dynamic human environments. In *Trajectory Planning for Robots in Dynamic Human Environments*, I E E E International Conference on Intelligent Robots and Systems. Proceedings, pages 4293–4298. IEEE Press, 2010.

[20] C. Wang, Y. Li, S. S. Ge, and T. H. Lee. Adaptive control for robot navigation in human environments based on social force model. In

2016 IEEE International Conference on Robotics and Automation (ICRA), pages 5690–5695, May 2016.

[21] U.S. Department of labor. American time use survey — 2018 results, 06 2019.

[22] United Nations Department of Economic and Social Affairs. *World Population Ageing 2019: Highlights*. United Nations, 2019.

[23] Rachid Alami, Alin Albu-Schäffer, Antonio Bicchi, Rainer Bischoff, Raja Chatila, Alessandro De Luca, Agostino De Santis, Georges Giralt, Jérémie Guiochet, Gerd Hirzinger, et al. Safe and dependable physical human-robot interaction in anthropic domains: State of the art and challenges. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1–16. IEEE, 2006.

[24] Roland Siegwart, Illah R Nourbakhsh, and Davide Scaramuzza. Autonomous mobile robots. *A Bradford Book*, 2:15, 2011.

[25] Priska Flandorfer. Population ageing and socially assistive robots for elderly persons: The importance of sociodemographic factors for user acceptance. *International Journal of Population Research*, page 13, 2012.

[26] Biel Piero E. Alvarado Vásquez and Fernando Matía. A tour-guide robot: Moving towards interaction with humans. *Engineering Applications of Artificial Intelligence*, 88:103356, 2020.

[27] A. K. Telkar and B. Gadgay. Iot based smart multi application surveillance robot. In *2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA)*, pages 931–935, 2020.

[28] Sangwon Park. Multifaceted trust in tourism service robots. *Annals of Tourism Research*, 81:102888, 2020.

[29] Gustavo A Cardona and Juan M Calderon. Robot swarm navigation and victim detection using rendezvous consensus in search and rescue operations. *Applied Sciences*, 9(8):1702, 2019.

[30] Juan M Calderon, Gustavo A Cardona, Juan Ramirez-Rugeles, and Eduardo Mojica-Nava. Visual victim detection and quadrotor-swarm coordination control in search and rescue environment. *International Journal of Electrical & Computer Engineering (2088-8708)*, 11(3), 2021.

[31] Evan Ackerman. Autonomous robots are helping kill coronavirus in hospitals. *IEEE Spectrum*, 11, 2020.

[32] Liang Tang, Guanjun Liu, Min Yang, Feiyang Li, Fangping Ye, and Chenyu Li. Joint design and torque feedback experiment of rehabilitation robot. *Advances in Mechanical Engineering*, 12(5), 2020.

[33] G. Muhammad M. S. Islam, M. M. Rahman and M. Shamim Hossain. Design of a social robot interact with artificial intelligence by versatile control systems. *IEEE Sensors Journal*, pages 1–1, 2021.

[34] Mordechai Ben-Ari and Francesco Mondada. *Robots and Their Applications*, pages 1–20. Elements of Robotics. Springer, Cham., 01 2018.

[35] X.Q. Chen, Y.Q. Chen, and J.G. Chase. Mobiles robots - past present and future. In XiaoQi Chen, Y.Q. Chen, and J.G. Chase, editors, *Mobile Robots*, chapter 1. IntechOpen, Rijeka, 2009.

[36] Ioan Doroftei, Victor Grosu, and Veaceslav Spinu. Omnidirectional mobile robot - design and implementation. In Maki K. Habib, editor, *Bioinspiration and Robotics*, chapter 29. IntechOpen, Rijeka, 2007.

[37] Jun Qian, Bin Zi, Daoming Wang, Yangang Ma, and Dan Zhang. The design and development of an omni-directional mobile robot oriented to an intelligent manufacturing system. *Sensors*, 17(9):2073, 2017.

[38] Jefri Efendi Mohd Salih, Mohamed Rizon, Sazali Yaacob, Abdul Adom, and Mohd Mamat. Designing omni-directional mobile robot with mecanum wheel. *American Journal of Applied Sciences*, 3, 05 2006.

[39] A. Gfrerrer. Geometry and kinematics of the mecanum wheel. *Comput. Aided Geom. Des.*, 25(9):784–791, December 2008.

[40] Bing Qiao Hamid Taheri and Nurallah Ghaeminezhad. Kinematic model of a four mecanum wheeled mobile robot, 2015.

[41] TA Baede. Motion control of an omnidirectional mobile robot. *Traineeship report DCT*, 2006, 2006.

[42] N. Tlale and M. de Villiers. Kinematics and dynamics modelling of a mecanum wheeled mobile platform. In *2008 15th International Conference on Mechatronics and Machine Vision in Practice*, pages 657–662, Dec 2008.

[43] Chengcheng Wang, Xiaofeng Liu, Xianqiang Yang, Fang Hu, Aimin Jiang, and Chenguang Yang. Trajectory tracking of an omni-directional wheeled mobile robot using a model predictive control strategy. *Applied Sciences*, 8:231, 02 2018.

[44] Jun Qian, Bin Zi, Daoming Wang, Yangang Ma, and Dan Zhang. The design and development of an omni-directional mobile robot oriented to an intelligent manufacturing system. *Sensors*, 17(9):2073, 2017.

[45] R. K. Panda and B. B. Choudhury. An effective path planning of mobile robot using genetic algorithm. In *2015 IEEE International Conference on Computational Intelligence Communication Technology*, pages 287–291, Feb 2015.

[46] M. Labbé and F. Michaud. Rtab-map as an open-source lidar and visual slam library for large-scale and long-term online operation. *Field Robotics*, 36:416–446, 09 2019.

[47] Mathieu Labbé and François Michaud. Memory management for real-time appearance-based loop closure detection. In *2011 IEEE/RSJ international conference on intelligent robots and systems*, pages 1271–1276. IEEE, 2011.

[48] M. Labbé and F. Michaud. Online global loop closure detection for large-scale multi-session graph-based slam. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2661–2666, 2014.

[49] M. Labbé and F. Michaud. Appearance-based loop closure detection for online large-scale and long-term operation. *IEEE Transactions on Robotics*, 29(3):734–745, 2013.

[50] Mathieu Labbe and François Michaud. Online global loop closure detection for large-scale multi-session graph-based slam. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2661–2666. IEEE, 2014.

[51] Joshua Samuel P Siy, Robert Keith C Chan, and Renann G Baldovino. Implementation of a real-time appearance-based mapping in a fully autonomous urban search robot. In *2018 IEEE 10th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM)*, pages 1–4. IEEE, 2019.

[52] Sagarnil Das. Simultaneous localization and mapping (slam) using rtab-map. *ArXiv*, abs/1809.02989, 2018.

[53] M.S. Achmad, Gigih Priyandoko, R. Roali, and Mohd Daud. Tele-operated mobile robot for 3d visual inspection utilizing distributed operating system platform. *International Journal of Vehicle Structures and Systems*, 9, 09 2017.

[54] N. ALPKIRAY, Y. TORUN, and O. KAYNAR. Probabilistic roadmap and artificial bee colony algorithm cooperation for path planning. In *2018 International Conference on Artificial Intelligence and Data Processing (IDAP)*, pages 1–6, Sep. 2018.

[55] German Carro Fernandez, Sergio Martin Gutierrez, Elio Sancristobal Ruiz, Francisco Mur Perez, and Manuel Castro Gil. Robotics, the new

industrial revolution. *IEEE Technology and Society Magazine*, 31(2):51–58, 2012.

[56] Aníbal Ollero Baturone. *Robótica: manipuladores y robots móviles*. Marcombo, 2005.

[57] Frank Hegel, Claudia Muhl, Britta Wrede, Martina Hielscher-Fastabend, and Gerhard Sagerer. Understanding social robots. In *2009 Second International Conferences on Advances in Computer-Human Interactions*, pages 169–174, 2009.

[58] Gang-Tae Bae, Seung-Won Kim, Dongeun Choi, Changhyun Cho, Woo-Sub Lee, and Sung-Chul Kang. Omni-directional power-assist-modular(pam) mobile robot for total nursing service system. In *2017 14th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, pages 832–834, 2017.

[59] Hamid Taheri, Bing Qiao, and Nurallah Ghaeminezhad. Kinematic model of a four mecanum wheeled mobile robot. *International journal of computer applications*, 113(3):6–9, 2015.

[60] Debashree Sengupta, Neha Jain, and C. S. Kumar. An educational website on kinematics of robots. In *2013 IEEE Fifth International Conference on Technology for Education (t4e 2013)*, pages 24–27, 2013.

[61] E. Maulana, M. A. Muslim, and V. Hendrayawan. Inverse kinematic implementation of four-wheels mecanum drive mobile robot using stepper motors. In *2015 International Seminar on Intelligent Technology and Its Applications (ISITIA)*, pages 51–56, 2015.

[62] Federico Andrade, Martin Llofriu, Gonzalo Tejera, and Facundo Benavides. Slam estado del arte, 2007.

[63] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. Ros: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, page 5. Kobe, Japan, 2009.

[64] M. Labbé and F. Michaud. Memory management for real-time appearance-based loop closure detection. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1271–1276, 2011.

[65] Mathieu Labbé and François Michaud. Rtab-map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation: LabbÉ and michaud. *Journal of Field Robotics*, 36, 10 2018.

[66] Evgeny Tsykunov, Valery Ilin, Stepan Perminov, Aleksey Fedoseev, and Elvira Zainulina. Coupling of localization and depth data for mapping using intel realsense t265 and d435i cameras. *arXiv preprint arXiv:2004.00269*, 2020.

[67] Nicolás Alvarez Casadiego, Mario Armando Segura Albarracin, et al. Implementación de algoritmos filtro de kalman y filtro de partículas para localización y mapeo simultáneo aplicado a un robot móvil en ambientes interiores con variaciones de iluminación.

[68] Kanjanapan Sukvichai, Kandith Wongsuwan, Nut Kaewnark, and Piyamate Wisanuvej. Implementation of visual odometry estimation for underwater robot on ros by using raspberrypi 2. In *2016 International Conference on Electronics, Information, and Communications (ICEIC)*, pages 1–4. IEEE, 2016.

[69] M. Quigley. Ros: an open-source robot operating system. In *ICRA 2009*, 2009.

[70] Intel RealSense. *D400 Series Product Family*, 2019.

[71] Intel RealSense. Intel real sense depth camera d435, November 2008.

[72] Intel RealSense. *Tracking Camera*, 2019.

[73] Intel RealSense. Intel realsense tracking camera t265, November 2008.

[74] PM Meshram and Rohit G Kanojiya. Tuning of pid controller using ziegler-nichols method for speed control of dc motor. In *IEEE-international conference on advances in engineering, science and management (ICAESM-2012)*, pages 117–122. IEEE, 2012.

[75] Bing Qiao Hamid Taheri and Nurallah Ghaeminezhad. Kinematic model of a four mecanum wheeled mobile robot, 2015.