UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS ELECTRÓNICA E INDUSTRIAL

MAESTRÍA EN AUTOMATIZACIÓN Y SISTEMAS DE CONTROL

Tema:

"NAVEGACIÓN AUTÓNOMA DE UN ROBOT MÓVIL OMNIDIRECCIONAL EN TRAYECTORIAS PREDETERMINADAS CON EVITACIÓN DE OBSTÁCULOS"

Trabajo de investigación, previo a la obtención del Grado Académico de Magister en Automatización y Sistemas de Control.

Autor: Ing. Santiago Álvarez T.

Director: Ing. Patricio Encalada, MSc.

Ambato - Ecuador

2019

A la Unidad Académica de Titulación de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial.

El Tribunal receptor del Trabajo de Investigación presidido por la Ingeniera Elsa Pilar Urrutia Urrutia Mg., e integrado por los señores: Ingeniero Marcelo Vladimir García Sánchez Dr., Ingeniero Franklin Wilfrido Salazar Logroño Mg., Ingeniero Carlos Diego Gordon Gallegos Dr., designados por la Unidad Académica de Titulación de la Universidad Técnica de Ambato, para receptar el Trabajo de Investigación con el tema: "NAVEGACIÓN AUTÓNOMA DE UN ROBOT MÓVIL OMNIDIRECCIONAL EN TRAYECTORIAS PREDETERMINADAS CON EVITACIÓN DE OBSTÁCULOS", elaborado y presentado por el señor Ing. Santiago Javier Alvarez Tobar, para optar por el Grado Académico de Magister en Automatización y Sistemas de Control; una vez escuchada la defensa oral del Trabajo de Investigación el Tribunal aprueba y remite el trabajo para uso y custodia en las bibliotecas de la UTA.

Ing. Elsa Pilar Urrutia Urrutia Mg.

Presidente de Tribunal

Ing. Marcelo Vladimir García Sánchez Dr.

Miembro de Tribunal

Ing. Franklin Wilfrido Salazar Logroño Mg.

Miembro de Tribunal

Ing. Carlos Diego Gordon Gallegos Dr.

Miembro de Tribunal

AUTORÍA DEL TRABAJO DE INVESTIGACIÓN

La responsabilidad de las opiniones, comentarios y críticas emitidas en el Trabajo de Investigación presentado con el tema: "NAVEGACIÓN AUTÓNOMA DE UN ROBOT MÓVIL OMNIDIRECCIONAL EN TRAYECTORIAS PREDETERMINADAS CON EVITACIÓN DE OBSTÁCULOS", le corresponde exclusivamente al señor Ingeniero Santiago Javier Alvarez Tobar, autor bajo la Dirección del Máster Patricio Germán Encalada Ruiz, Director del Trabajo de Investigación; y el patrimonio intelectual a la Universidad Técnica de Ambato.

Ing. Santiago Javier Alvarez Tobar

c.c. 1802704088

AUTOR

Ing. Patricio Germán Encalada Ruiz, MSc

c.c.1803881935

DIRECTOR

DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato, para que el Trabajo de Investigación, sirva como un documento disponible para su lectura, consulta y procesos de investigación, según las normas de la Institución.

Cedo los Derechos de mi trabajo, con fines de difusión pública, además apruebo la reproducción de este, dentro de las regulaciones de la Universidad.

Ing. Santiago Javier Alvarez Tobar

c.c.1802704088

ÍNDICE GENERAL DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

AGRADECIMIENTO

Mi agradecimiento principal a toda mi familia que ha sido el soporte fundamental de mi crecimiento personal. A mi querida Universidad Técnica de Ambato, en especial a la Facultad de Sistemas, Electrónica e Industrial, institución que se encargó de dotarme de todos los conocimientos tanto en pregrado como en posgrado, además de recibir todo el apoyo del personal Administrativo, Docente y de Servicios. A todos mis amigos quienes con sus palabras de aliento me otorgaron de la fuerza necesaria para avanzar hacia una meta más en mi vida.

Un agradecimiento especial a mi Tutor por haberme compartido sus conocimientos y brindado su apoyo para culminar este proyecto.

Santiago Javier Alvarez Tobar

DEDICATORIA

Esta tesis está dedicada a:

Dios por bendecirnos con la vida y guiarnos a los largo de nuestras exigencias.

A mi Amada Esposa quien es un pilar fundamental de mi vida, la persona que siempre se encuentra a mi lado en los buenos y malos momentos. Te amo.

A mis padres por enseñarme el correcto caminar en la vida, además se darme todo su amor y apoyo incondicional.

A mi hermana por ser mi alma gemela, quien con sus palabras de aliento siempre fortalecieron mi crecimiento.

A mi Abuelita Teresa quien con su sabiduría y su ilimitado amor siempre nos cobija con su ternura.

Y una dedicatoria especial para mi abuelito Gato, un ser único y especial que siempre guía nuestros caminos desde el cielo, una persona que nos inculco de los más grandes valores de la vida y siempre fue un gran ejemplo para todos.

Santiago Javier Alvarez Tobar

UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERIA EN SISTEMAS, ELECTRÓNICA E INDUSTRIAL / DIRECCIÓN DE POSGRADO

MAESTRÍA EN AUTOMATIZACIÓN Y SISTEMAS DE CONTROL

TEMA:

NAVEGACIÓN AUTÓNOMA DE UN ROBOT MÓVIL OMNIDIRECCIONAL EN TRAYECTORIAS PREDETERMINADAS CON EVITACIÓN DE OBSTÁCULOS

AUTOR: ING. SANTIAGO JAVIER ÁLVAREZ TOBAR

DIRECTOR: ING. PATRICIO ENCALDA, MSC

FECHA: Junio, 2018

RESUMEN EJECUTIVO

La navegación de robots hoy en día a evolucionado con propósitos de colaboración y de trabajo en diferentes espacios, haciendo uso sus modelos cinemáticos e incluso de modelos dinámicos para perfeccionar la tarea a realizar por un robot. Un robot móvil con capacidad de moverse en un lugar cumpliendo una trayectoria predefinida es muy útil para transportar objetos de un lugar hacia otro, incluyendo en este una capacidad de no colisionar con objetos en el espacio a desplazarse, logrando de esta forma llegar a la meta sin problema alguno.

Es tanto el avance de la tecnología que se pude desarrollar prototipos no tan complejos de sistemas robóticos, para realizar tareas necesarias y cotidianas por las personas, con tan solo saber que dispositivos usar y de qué forma implementarlos, basándose en leyes matemáticos o algoritmos de control se puede comandar cualquier tipo de robot ya sea móvil terrestre, aéreo o acuático; dentro del campo investigativo se estudia múltiples algoritmos de control para sistemas no lineales con los cuales se pude obtener controladores que permitan emplearse para el funcionamiento adecuado, es así que se han creado controladores desde los más simples como un control PID hasta un control avanzado como controladores predicativos, entre otros.

En esta propuesta de trabajo de titulación se presentará el desarrollo de un robot tipo omnidireccional, realizando un estudio mecánico para definir la manera adecuada de construcción en cuanto a materiales mecánicos y electrónicos y además que configuración implementar con el uso de ruedas Mecanum que nos permiten desplazar en cualquier dirección. Para lograr realizar un control autónomo se ejecuta un control interno en el robot para compensar la dinámica y por medio de un control basado en el modelo cinemático del robot emplear un control autónomo para que logre desplazarse por un perfil predefinido. Con el propósito de que el robot no colisione se incorpora un sensor Laser que permite obtener información del entorno en el que se está desplazando el robot.

Para validar el modelo cinemático, el algoritmo de navegación autónoma y la construcción del robot se realiza pruebas experimentales del robot con diferentes perfiles predefinidos, observando cual es el comportamiento del robot y como lo errores tienden a ser cero según trascurra el tiempo de ejecución.

Descriptores: Cinemática, robot móvil, tipo omnidireccional, control autónomo, navegación, control no lineal, controles avanzados, dinámica, sensor Laser, ruedas Mecanum

.

UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS ELECTRÓNICA E INDUSTRIAL /DIRECCIÓN DE POSGRADO

MAESTRÍA EN AUTOMATIZACIÓN Y SISTEMAS DE CONTROL

THEME:
AUTONOMOUS NAVIGATION OF AN OMNIDIRECTIONAL MOBILE ROBOT ON PREDEFINED TRAJECTORY WITH OBSTACLE AVOIDANCE

AUTOR: ING. SANTIAGO JAVIER ÁLVAREZ TOBAR

DIRECTOR: ING. PATRICIO ENCALDA, MSC

FECHA: Junio, 2018

EXECUTIVE SUMMARY

The navigation of robots today has evolved with the purpose of collaboration and work in different spaces, using the kinematic models and dynamic models to perfect the task to be performed by a robot. A mobile robot with the ability to move in a place fulfilling a predefined path is very useful to transport objects from one place to another, including in this a capacity not to collide with objects in the space to move, thus achieving the goal without any problem.

It is so much the advance of the technology, that could be developed not so complex prototypes of robotic systems, to realize necessary and daily tasks for the people, with only knowing what devices to use and how to implement them, based on mathematical laws or control algorithms you can command any type of robot mobile, aerial or aquatic robot; within the research field we study multiple control algorithms for nonlinear systems with which we can obtain controllers that allow us to use them for proper operation, so that controllers have been created from the simplest as a PID control to an advanced control as predictive controllers, among others.

In this proposal of work of titulación will present the development from zero of a robot type omnidireccional, realizing a mechanical study to define the suitable way

of construction as far as mechanical and electronic materials and in addition that configuration to implement with use of wheels Mecanum that allows to move in any direction. To achieve autonomous control, an internal control is executed in the robot to compensate for the dynamics and by means of a control based on the kinematic model of the robot using autonomous control so that it can move through a predefined profile. With the purpose that the robot does not collide a Laser sensor is incorporated that allows obtaining information of the environment in which the robot is moving.

For validate the kinematic model, the autonomous navigation algorithm and the robot construction, is performed experimental tests of the robot with different predefined profiles, observing what is the behavior of the robot and how errors tend to be zero as the execution time elapses.

Keywords: Kinematics, mobile robot, type omnidireccional, autonomous control, navigation, non-linear control, advanced controls, dynamics, Laser sensor, wheels Mecanum.

INTRODUCCIÓN

El siguiente trabajo Investigativo propuesto cuenta como tema: NAVEGACIÓN AUTÓNOMA DE UN ROBOT MÓVIL OMNIDIRECCIONAL EN TRAYECTORIAS PREDETERMINADAS CON EVITACIÓN DE OBSTÁCULOS, el cual está enfocado a un estudio bibliográfico de todos los recursos necesarios para implementar un robot que cumpla con la navegación segura y posteriormente construirlo, para realizar pruebas experimentales que validen la propuesta planteada. Mediante el estudio de dispositivos y partes mecánicas que se necesita para el robot omnidireccional se ve las mejores características tanto en software como en hardware para implementar un el robot omnidireccional y que cumpla trayectorias predefinidas evitando colisionar con obstáculos presentes en el trayecto. Para el desarrollo de la propuesta, el trabajo se divide en seis capítulos, incluyendo un resumen ejecutivo del proyecto y las respectivas referencias bibliográficas.

En el Capítulo I, se estudia y se realiza una justificación del tema a ser investigado, definiendo cuales son las variables contextuales que se va a resolver en el proyecto y dejando claro que es lo que va a realizar.

En el capítulo II, se realiza un estudio de los proyectos relacionados con el tema planteado, buscando trabajos similares dentro y fuera del país, respaldando así el tema y los objetivos que se plantea.

El capítulo III, se describe la metodología de investigación y desarrollo del tema, planteando preguntas en las cuales se analiza la factibilidad y recursos que existen para construir un prototipo y de qué manera. En el final de este capítulo se estudia una propuesta para el análisis de toda información obtenida, con el propósito de encontrar adversidades o inconvenientes que se puede presentar en el desarrollo.

En el capítulo IV, se ejecuta el respectivo análisis y la interpretación de la información bibliográfica obtenida, respondiendo de una forma teórica las preguntas planteadas en el capítulo III, para encontrar las soluciones al problema del trabajo investigativo.

En capítulo V se detallan las conclusiones y recomendaciones que se dan a partir de análisis bibliográfico realizado, permitiendo mediante estas conclusiones y recomendaciones dar una solución puntual al tema del proyecto.

Por último, en el capítulo VI se realiza el desarrollo de la propuesta, describiendo métodos y sistemas empleados para cumplir cada objetivo del tema de investigación. En este capítulo se describe desde la construcción de un robot Omnidireccional hasta la ejecución de las pruebas experimentales del algoritmo de navegación

CAPÍTULO I
EL PROBLEMA DE INVESTIGACIÓN

1.1. Tema de investigación

NAVEGACIÓN AUTÓNOMA DE UN ROBOT MÓVIL OMNIDIRECCIONAL EN TRAYECTORIAS PREDETERMINADAS CON EVITACIÓN DE OBSTÁCULOS

1.2. Planteamiento del problema

1.2.1. Contextualización

Dada la tendencia de la robótica en el área industrial y de servicios, la necesidad de estudiar el comportamiento de vehículos autónomos ha crecido exponencialmente. Tanto los procesos industriales como los de servicios requieren el desplazamiento de objetos dentro de un espacio interno estructurado o no estructurado, requiriendo en muchas de las ocasiones vehículos de transporte conducidos por personal humano. Sin embargo, la robótica ha venido solucionando las limitaciones del uso de operarios para incluir métodos autónomos que optimicen recursos y no dependan de usuarios calificados para operar los robots (Forlizzi & DiSalvo, 2006).

Los robots comúnmente utilizados son de tipo uniciclo u omnidireccional, los cuales presentan las características adecuadas para transporte de elementos, empuje y arrastre, movimiento en espacios reducidos y otros (Biswas & Veloso, 2010) (Siegwart, Nourbakhsh, & Scaramuzza, 2004). Específicamente, las características de los robots móviles omnidireccionales permiten un desplazamiento holonómico en todos los ejes, mientras modifican su ángulo de rotación simultáneamente (Begum, Lee, & Kim, 2010). La aplicación industrial de este tipo de asistentes robóticos comúnmente es orientada a espacios estructurados, donde señalética para ubicar el robot y brindarle información del entorno es requerida debido a las limitaciones de reconocer el espacio de trabajo.

Sin embargo, no todas las áreas de aplicación pueden incluir medios de señalética para posicionar los robots o el costo de su inclusión no representa beneficios

económicos comparados con los que se tenga con operarios humanos. Bajo este paradigma, muchas de las aplicaciones de los robots móviles tienen restricciones debido a las limitaciones de los robots en cuanto a la detección de elementos que bloquean su desplazamiento en el espacio de trabajo.

1.2.2. Árbol del problema

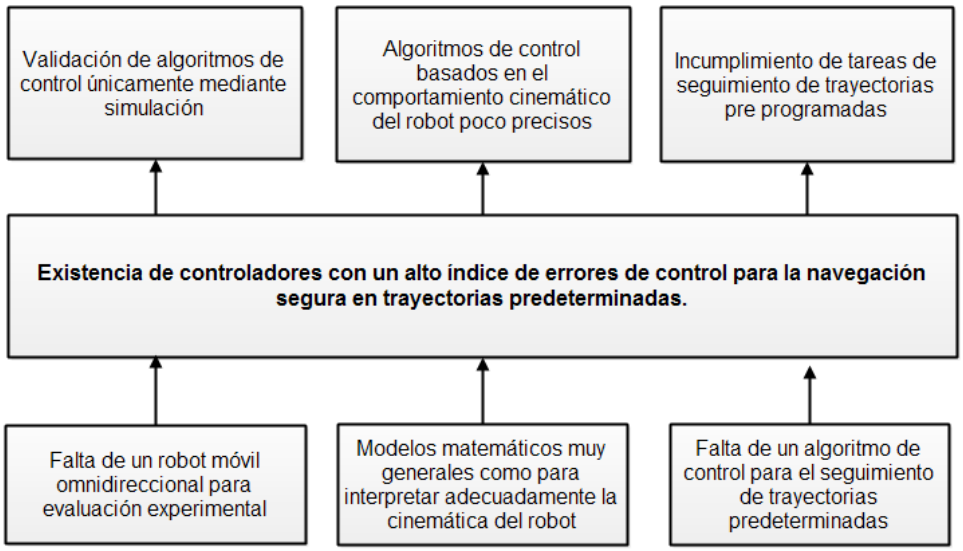


Fig. 0.1 Árbol del problema

Elaborado por: Santiago Álvarez

1.2.3. Análisis crítico

El desarrollo de la robótica en la actualidad existen prototipos robóticos de diferentes costes que cuentan con propios controladores ya pre establecidos, para las pruebas experimentales es necesario contar con un robot que cuente las configuraciones deseadas y que este apto para ser utilizado con varios controladores para tareas planteadas, en este caso particular, sin un robot móvil omnidireccional el controlador a desarrollarse no puede ser evaluado mediante resultados reales que corroboren los objetivos de diseño. Una solución a esta necesidad es el uso de simuladores de robots que permiten verificar los resultados de forma rápida, sin embargo, no reemplaza la validación sobre entornos reales.

El diseño de algoritmos de control robóticos requiere como fundamento de análisis el modelo matemático que describa el comportamiento del robot en el espacio de

interés, por otro lado, existen métodos de diseño de controlador empleados en diferentes prototipos robóticos, los cuales son diseñados en base a la configuración de cada robot.

Una de los grandes problemas de la robótica móvil, es la navegación autónoma, que, descrita de otra manera, es el seguimiento de trayectorias previamente determinadas, haciendo más complejo la generación de decisiones en el vehículo omnidireccional para que cumplan con la tarea especificada. Dicha generación de movimiento es determinada mediante un algoritmo de control, dependiendo de los objetivos planteados y considerando la cinemática del robot en base a la configuración que posee.

La complejidad y el costo de incluir sistemas para seguimiento de trayectorias en espacios no estructurados, limitan el campo de aplicación de los robots móviles holonómicos y no holonómicos (Girdhar & Dudek, 2016). Los vehículos requieren sistemas para conocer su posición y orientación con el objetivo de seguir una trayectoria deseada, problema que puede ser solucionado a través de la cinemática y la edometría empleada para el sistema del posicionamiento. Sin embargo, ante la aparición de obstáculos en el medio de trabajo, la ejecución puede variar drásticamente dañando la integridad del vehículo o el material que transporta ya que el mecanismo robótico ignora la presencia de elementos externos que impiden la ejecución de la tarea.

Aun cuando los espacios sean estructurados, las áreas de trabajo están propensas a interrumpir el paso de los vehículos con objetos que afectan la locomoción de los robots. En este caso, una cadena de accidentes laborales puede suceder reduciendo la cantidad de producción, reduciendo la fiabilidad en este tipo de sistemas o aumentando la cantidad de lesiones por parte del personal humano. Adicionalmente, dada la posible falla del transportador robótico, sistemas de supervisión en línea deben ser incluidos, reduciendo las ganancias de industrias que dependen directamente de la velocidad de ejecución de los procesos.

1.2.4. Prognosis

La no inclusión de asistentes robóticos reduce drásticamente las posibilidades de competir en el mercado industrializado actual. Sin embargo, la inclusión de robots que no satisfagan con los requerimientos mínimos de buen funcionamiento impedirá el crecimiento de producción en lugar de fortalecerlo. Tomando en cuenta estos factores, evadir la corrección de este tipo de problemas reducirá la fiabilidad en este tipo de sistemas, desencadenando en el uso de sistemas más costosos o más complejos de supervisar si y solo si la capacidad económica de la empresa facilite esta decisión.

1.2.5. Formulación del problema

¿Es posible modelar, proponer e implementar un sistema de navegación autónoma en trayectorias predeterminadas con evitación de obstáculos en un vehículo omnidireccional?

1.2.6. Preguntas directrices

- ¿Qué es un sistema de navegación autónoma?
- ¿Cómo se modela matemáticamente un robot móvil omnidireccional?
- ¿Qué control se usa para la navegación en trayectorias predeterminadas?
- ¿Cómo evitar obstáculos que se presenten en la trayectoria deseada?
- ¿Qué prototipo se emplea para validar el controlador de navegación propuesto?

1.2.7. Delimitación del problema

1.2.7.1. Límite del contenido

Área Académica: Sistemas de Control

Línea de investigación: Control y Programación de robots. Robótica Avanzada.

1.2.7.2. Limite espacial

Facultad de Ingeniería en Sistemas, Electrónica e Industrial, de la Universidad Técnica de Ambato.

1.2.7.3. Límite Temporal

La presente investigación se realizará en un lapso de 6 meses a partir de la aprobación del Honorable Consejo Académico de la Facultad de ingeniería en Sistema, Electrónica e Industrial de la Universidad Técnica de Ambato.

1.3. Justificación

Actualmente, la matriz productiva del país se centra en promover la intensidad tecnológica en la producción primaria, de bienes intermedios y finales. Pero en un orden más específico, busca articular y vincular la investigación científica, tecnológica y la educación superior al sector productivo (Secretaría Nacional de Planificación y Desarrollo, 2012). Dadas las necesidades actuales y futuras del sector productivo, se busca obtener una mejora continua de la productividad y competitividad a partir del conocimiento generado. Por ende, toda la nueva investigación desarrollada debe tener una orientación a la mejora de la economía del país a través de nuevas formas de generar riqueza y desarrollo.

La modelación cinemática del vehículo omnidireccional y la implementación en un prototipo a escala permitirá generar nuevos campos de aplicación en el seguimiento de trayectoria segura. Al distanciar la dependencia de un controlador humano, tareas como la asistencia de personas de la tercera edad, desplazamiento de cargamento, evaluación de elementos de seguridad vial y otros, pueden ser llevados a cabo de manera autónoma y ser la base investigativa para otros trabajos similares.

En este contexto, una investigación enfocada al análisis de la modelación, simulación y control de un vehículo omnidireccional para el seguimiento de

trayectorias pre-programadas es necesaria. Adicionalmente, un controlador capaz de calcular las velocidades lineales y angulares necesarias para mantener al vehículo dentro de una trayectoria especificada por computadora, con el objetivo de obtener resultados y poder graficarlos a fin de comprobar la eficiencia del controlador desarrollado. En la parte experimental, un prototipo a escala permitiría validar los resultados obtenidos en simulación y denotar en el campo real si los controladores propuestos cumplen con las características requeridas para el control adecuado del robot. La trayectoria deseada definida como una sucesión de puntos parametrizados en el tiempo para el robot es dada directamente por computador, independientemente de que este dentro de esta trayectoria, ya que el controlador será diseñado para cumplir la navegación segura evitando obstáculos que se presenten dentro de la trayectoria dada.

1.4. Objetivos

1.4.1. Objetivo general

Diseñar e implementar un algoritmo de navegación autónoma para el seguimiento de trayectorias predeterminadas con evitación de obstáculos de un robot móvil tipo omnidireccional.

1.4.2. Objetivos específicos

- Investigar los sistemas de control para la navegación autónoma de robots móviles mediante el análisis bibliográfico para determinar los métodos más utilizados.
- Modelar el robot móvil tipo omnidireccional matemáticamente analizando el comportamiento cinemático y la configuración del robot para proponer el algoritmo de navegación autónoma.
- Diseñar un algoritmo de control para la navegación autónoma usando el modelo cinemático del robot móvil tipo omnidireccional para el seguimiento de trayectorias predefinidas.

- Diseñar un algoritmo de evasión de obstáculos mediante reconstrucción tridimensional para el seguimiento de trayectoria segura.
- Construir un robot omnidireccional utilizando ruedas tipo Mecanum para probar experimentalmente los algoritmos de navegación autónoma y evitación de obstáculos.

CAPÍTULO II

2.1. Antecedentes investigativos

Internacionalmente, los temas relacionados con el objetivo de investigación presentado se describen en la presente sección. (Kundu, Mazumdera, Dhara, Lenkab, & Bhaumikc, 2016) presentan el desarrollo de un sistema de localización visual para un robot omnidireccional, usando un conjunto de marcadores para conocer la posición del robot en el plano ya que consideran que el deslizamiento producido por las ruedas provoca inexactitudes al momento de calcular el desplazamiento a través de edometría. Otros temas usando el mismo tipo de robots son presentados por (Ismael & Hedley, 2016), donde analizan y describen la modelación cinemática, el diseño y la experimentación de una plataforma omnidireccional, además de diseñar el control para el seguimiento de una esfera de color mediante un sistema de visión. Asimismo, (Yip, Ho, Chu, & Lai, 2014) presentan el desarrollo de un robot omnidireccional equipado con una cámara RGB-D para obtener información del medio donde se desplaza, pero sin el planteamiento de seguimiento de caminos o trayectorias. Con una aplicación concreta, (Tsai, Huang, & Lin, 2011) presentan la construcción, diseño y controladores para un robot onmidireccional con propósitos de extinción de fuego. Con el mismo objetivo de control, (Yang, Fan, Shi, & Hua, 2015) presentan un control no lineal para el seguimiento de objetivos con evasión de obstáculos, sin embargo, usa un vehículo no holonómico.

Los trabajos mencionados anteriormente dan a notar la variedad de aplicaciones que se le dan en la actualidad a los robot móviles, destacando la independencia del hardware y el software, a pesar, que estas investigaciones diseñan y construyen sus propios prototipos para resolver un problema muy específico, todas pudieron ser instaladas en prototipos previamente implementados, concluyendo la importancia de construir prototipos flexibles que permitan expandir sus aplicaciones, para validar uno o más controladores.

En el área de investigación nacional, se presentan tres trabajos enfocados a construcción o implementación de algoritmos de control. (Diego & Johnny, 2014) diseñan y construyen un prototipo omnidireccional para propósitos de carga con tres grados de libertad, con un alcance de hasta 0.8 m/s de velocidad. De manera similar, (Iza & Taco, 2016) plantean el análisis para la construcción de una plataforma omnidireccional para fines de investigación. Finalmente, (Pérez, 2015) presenta la implementación de algoritmos de determinación y métodos planificadores, analizando circunstancias externas como el equipo computador usado, la complejidad del entorno de trabajo, etc.

2.2. Fundamentación filosófica

La fundamentación es completamente investigativa. A través de trabajos desarrollados en la misma línea, se plantea un enfoque orientado a la generación de nueva teoría que sirva como fundamento teórico para trabajos futuros.

2.2.1. Categorías fundamentales

2.2.1.1. Redes de inclusiones conceptuales

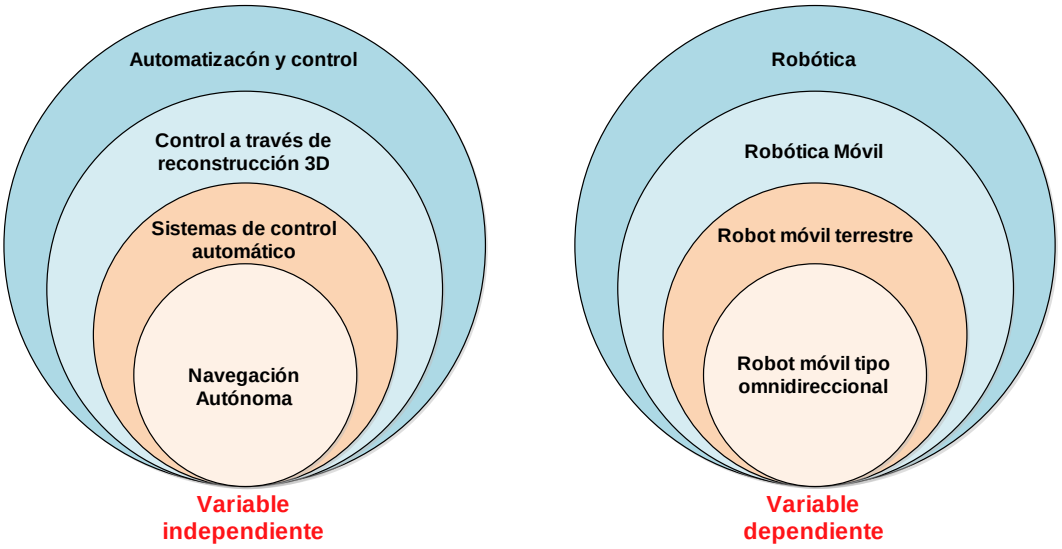


Fig. 2.1 Variables dependientes e independientes de las categorías fundamentales

Elaborado por: Santiago Álvarez

2.2.1.2. Constelación de ideas

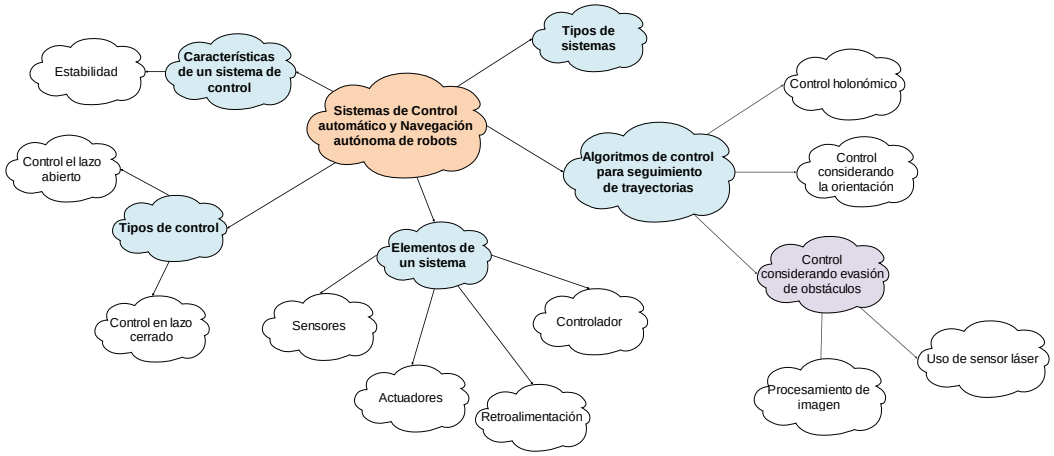


Fig. 2.2 Constelación de ideas de la Variable independiente

Elaborado por: Santiago Álvarez

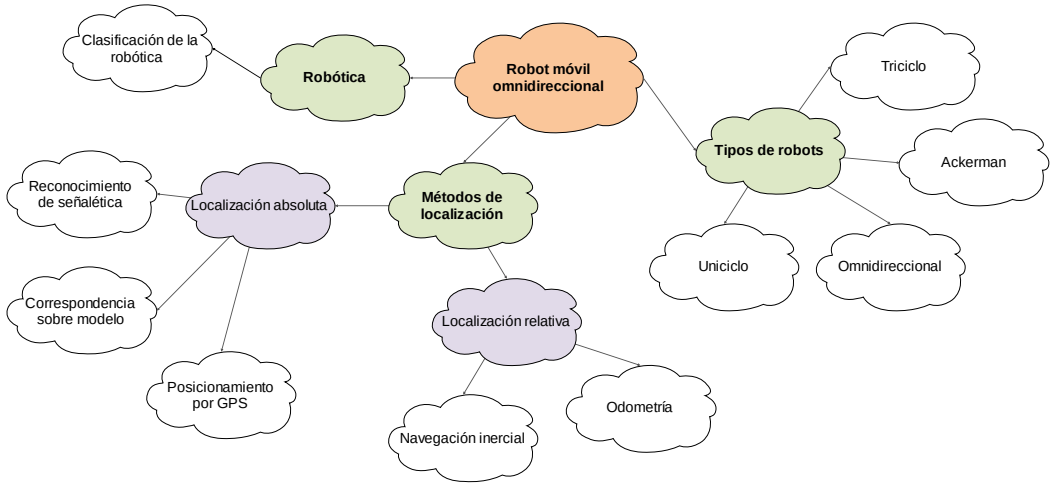


Fig. 2.3 Constelación de ideas de la Variable dependiente

Elaborado por: Santiago Álvarez

2.2.2. Automatización

La Automatización y control es una de las ramas de la ingeniería que integra tecnologías de vanguardia para solucionar problemas del ámbito industrial y de consumo. La automatización y el control se han utilizado para describir sistemas de fabricación que usan equipos o dispositivos programados, los cuales trabajan de manera total o parcialmente independiente del control humano. Más específicamente, un sistema de control es un grupo de equipos destinados a

administrar, ordenar y dirigir el comportamiento de un sistema con el objetivo de reducir la intervención humana (Ogata, 2010).

2.2.3. Sistemas de Control

Un sistema se define como el conjunto de sensores, actuadores y equipos que interactúan de manera armónica para cumplir con un objetivo común. De esta manera, un sistema está compuesto por elementos que reciben entradas y producen salidas. Por lo tanto, el comportamiento de un sistema es modificado considerablemente cuando se reemplaza uno de los componentes, así como cuando las entradas se alteran (Fig. 2.4).

Visto desde una perspectiva ingenieril, el control de un sistema se efectúa mediante componentes y señales mecánicas, hidráulicas, eléctricas, electrónicas, etc., que adquieren información del medio de trabajo y la comparan con los valores de consigna para ejecutar respuestas e intentar forzar al sistema a llegar a un resultado deseado. Por la manera de controlar el sistema, se puede dividir en control manual, control automático y una mezcla de estos dos (Ogata, 2010).

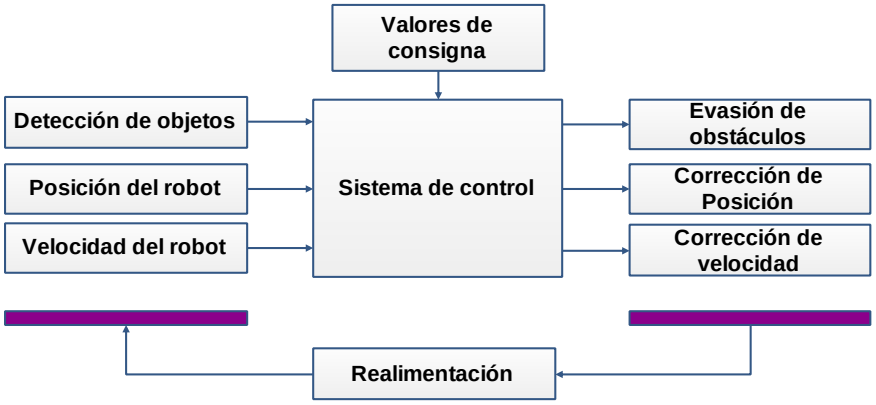


Fig. 2.4 Entradas y Salidas de un sistema de control.

Fuente: Sistemas de control (Tecnología Técnica , 2017)

2.2.3.1. Tipos de Sistemas de Control

A. Sistema de control manual

En el sistema de control manual se requiere de la intervención del operador humano para alcanzar la consigna de interés. Por tanto, la acción del hombre es necesaria para producir cambios en el funcionamiento en el sistema. Un ejemplo práctico de

este tipo de control es el frenado de un vehículo o el encendido y apagado de luces en un departamento (Chaturvedi, 2010).

B. Sistemas de control automático

En este tipo de sistemas la respuesta es provocada sin que ningún operario intervenga sobre el mismo, con excepción en la configuración de condiciones iniciales o valores deseados. En este caso, el operador humano es reemplazado por dispositivos tecnológicos que operan automáticamente al sistema, tales como relés, válvulas, actuadores, motores, etc.

C. Sistema de control semiautomático

En este tipo de sistemas, la intervención de equipos controladores y de un operario es requerida. Esta combinación se da cuando el proceso es muy complejo de controlar o el producto final es sensible o peligroso de manipular o procesar.

2.2.4. Configuraciones de los sistemas de control

Por la configuración, un sistema de control puede clasificarse en sistemas de control el lazo cerrado y sistemas de control en lazo abierto.

2.2.4.1. A. Sistemas de control a lazo abierto

El control de este tipo de sistema, la salida no tiene ningún efecto sobre la acción de control. Por tanto, este tipo de sistemas no monitorizan la variable controlada, por la que no dependen de un sensor. El control en lazo abierto no modifica o corrige su funcionamiento en función de condiciones del entorno, sino que siguen una secuencia de operación predeterminada en modo fuera de línea (Ogata, 2010). Un ejemplo práctico de este tipo de control es una licuadora, una lavadora o una puerta de garaje. Su diagrama de bloques se muestra en la Fig. 2.5.

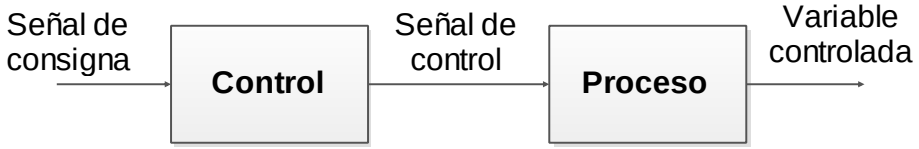


Fig. 2.5 Control en lazo abierto

Elaborado por: Santiago Álvarez

2.2.4.2. Sistemas de control en lazo cerrado

A diferencia de un control en lazo abierto, un sistema de control en lazo cerrado (Fig. 2.6) vigila permanentemente la variable de salida o controlada, actuando en función de la misma para realizar con cambio en la variable de control. Para conocer la información de salida, este tipo de controles requieren de un sensor y un algoritmo de control para mantener la salida lo más cercana posible a la señal de referencia. Una de las ventajas principales de este sistema es que la realimentación se usa para corregir las perturbaciones externas y a la variación de los parámetros internos (Kuo, 2000).

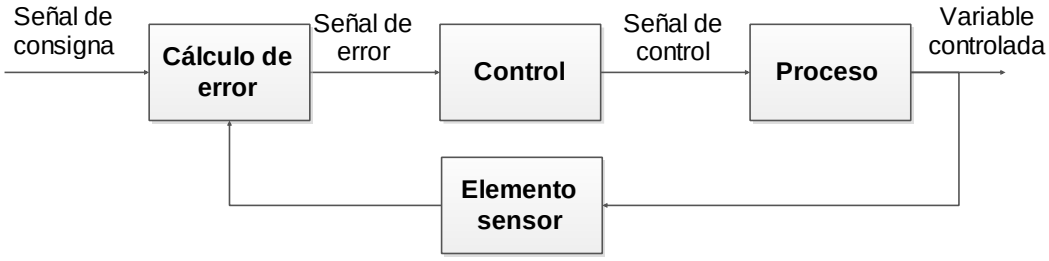


Fig. 2.6 Control en lazo cerrado

Elaborado por: Santiago Álvarez

2.2.5. Requisitos de un sistema de control

Analizando un sistema de control de manera ideal, los requisitos para un adecuado funcionamiento son:

- Ser tan eficiente como sea posible, según un criterio establecido.
- Ser sencillo y fácil de implementar, además de cómodo de operar en tiempo real.
- Garantizar la estabilidad a lo largo del tiempo
- Ser el suficientemente robusto como para corregir las perturbaciones tanto internas como externas del sistema.

2.2.6. Elementos de un sistema de control

Independientemente del tipo de tecnología usada en un sistema de control, los dispositivos mostrados a continuación forman parte casi siempre de un sistema

controlado. El diagrama de bloques mostrado en la Fig. 2.7 incluye todos los elementos que se hacen referencia (Kuo, 2000).

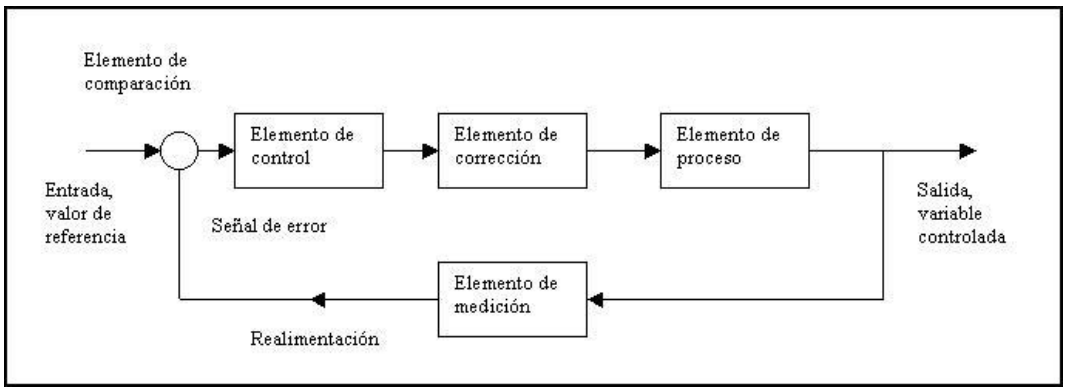


Fig. 2.7 Elementos de un sistema de control

Fuente: Sistemas de control automático (Kuo, 2000)

Generador del valor de referencia: Es la parte del sistema que genera la señal encargada de configurar el valor deseado en la salida. La señal de referencia puede ingresar directamente al controlador o a un bloque de cálculo de error, dependiendo de la configuración del sistema de control.

Transductor de la señal de salida: Consiste en un dispositivo capaz de medir el valor de la magnitud de salida en cada instante de tiempo definido por el periodo de muestreo. El transductor comúnmente consta de dos partes: el captador o elemento de primario (encargado de captar directamente la magnitud medida) y el transmisor, el cual transforma la magnitud y la reemplaza por otro tipo de señal equivalente (normalmente eléctrica o neumática).

Comparador o detector de error: Es el elemento del sistema encargado de comparar el valor de consigna con el valor devuelto por la retroalimentación.

Corrector de error: Es el equipo encargado de comparar el valor medido por el detector de errores con el valor dado por el generador de referencias, con el objetivo de corregir las acciones de control.

Elemento final de control: Es el dispositivo situado en un sistema de control con la misión de modificar la variable de salida para obtener el valor deseado.

Sistema: O conocido también como planta, es el lugar donde se desea realizar las acciones de control.

2.2.7. Robótica

La robótica es la ciencia que involucra el diseño y la fabricación de robots para usos prácticos, tanto en la industria como en servicios domésticos. La robótica conjuga muchas disciplinas tales como la informática, electrónica y la mecánica para la construcción de equipos y dispositivos que funcionen de manera automática, realizando trabajos de manera automática que resulten repetitivos o peligrosos para el ser humano (Corke, 2011). La Fig. 2.8 muestra una de las aplicaciones más comunes de los robots en la industria.

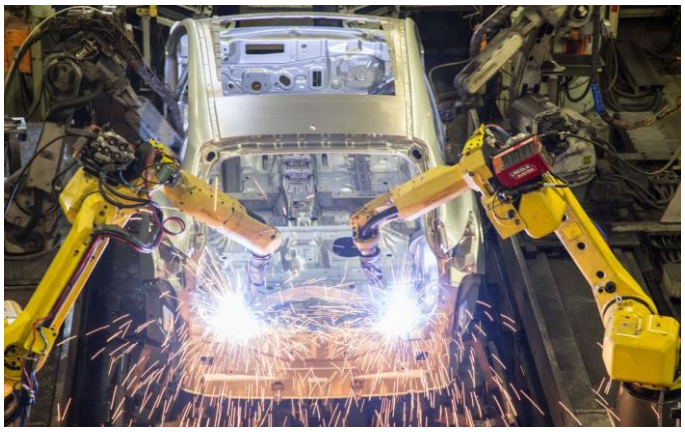


Fig. 2.8 Aplicaciones de robots

Elaborado por: Construcción de Vehículos (C-NET, 2017)

En estos días y con el desarrollo de la electrónica, la robótica ha crecido a pasos agigantados hasta llegar a campos tan específicos como la cirugía robótica. De esta manera, la robótica se ha enfocado en áreas tan delicadas como la asistencia de personas con algún tipo de discapacidad física, así como en áreas tan complejas como las militares para la defensa de poblaciones.

2.2.8. Clasificación de la robótica

La clasificación de la robótica se da a partir de distintos enfoques (Ollero Baturone, 2001). Según el momento tecnológico en el que aparece, la robótica se clasifica en generaciones (Fig. 2.9):

Primera generación: Conocidos como robots de manipulación, este tipo de robots son sistemas mecánicos multifuncionales con un sencillo sistema de control, ya sea manual o de secuencia fija.

Segunda generación: Conocidos como robots de aprendizaje, estos sistemas repiten una secuencia de movimientos que han sido ejecutados previamente por un operador humano.

Tercera generación: Definidos como robots con control sensorizado, el robot incluye una computadora interna que ejecuta las órdenes generadas desde un programa informático y las envía el robot para que realice los movimientos necesarios.

Cuarta generación: La última generación está constituida por robots inteligentes. Tienen características similares a los de la anterior generación, pero incluyendo sensores que enviar información del entorno hacia la computadora para tomar decisiones en tiempo real.

a) b) c) d)

Fig. 2.9 Generaciones de la robótica: a) primera, b) segunda, c) tercera, d) cuarta

Fuente: Generaciones de la robótica (Guillenxt, 2017)

2.2.9. Robótica móvil

La necesidad de desanclar a un robot de una estructura metálica fija ha provocado el desarrollo de la robótica móvil (Ollero Baturone, 2001). Con esto, se logra adicionalmente incrementar la autonomía del robot y extender su área de trabajo ya que su capacidad de desplazamiento lo hace útil para áreas como transporte de carga, inspección de espacios confinados, espionaje, entre otros.

Los robots móviles son un conjunto de elementos electrónicos de desplazamiento autónomo, es decir, son plataformas dotadas de sistemas de locomoción capaces de navegar a través de un determinado ambiente de trabajo, dotado de cierto nivel de autonomía para su desplazamiento portando cargas o cumpliendo con una misión (Nourbakhsh I. and Siegwart R. 2004).

Dentro de la investigación de este tipo de robots, diversos trabajos se han enfocado en su modelo cinemático, demostrándose que su estudio se limita a bajas velocidades y bajas cargas. Para incluir aplicaciones de alta velocidad y alta carga, el estudio y análisis de aplicaciones que consideran la dinámica del vehículo han sido desarrolladas.

2.2.10. Tipos de robots móviles

Por el tipo de locomoción, los robots móviles pueden clasificarse en tres grupos: por ruedas, por patas y por orugas. Aunque todos estos tipos de robots han sido estudiados ampliamente, las características que presentan los robots de ruedas los hace más fáciles de aplicar en distintas áreas de la ingeniería. Dentro de las ventajas más relevantes de los robots móviles de ruedas, el ahorro de energía, desplazamiento en terrenos lisos, capacidad de carga y rapidez de movimientos son las que más destacan (Siegwart, Nourbakhsh, & Scaramuzza, 2004).

2.2.11. Robots móviles con ruedas

Los vehículos con ruedas se presentan como la solución más eficiente y simple para conseguir la movilidad en terrenos lo suficientemente duros y sin obstáculos, permitiendo conseguir de esta manera velocidades de desplazamiento relativamente altas. Los robots móviles emplean diferentes tipos de locomoción mediante arreglos

de ruedas que les permiten adquirir características y propiedades respecto a la eficiencia energética, dimensiones, maniobrabilidad, etc. Especialmente en la maniobrabilidad, la mayor explotación de esta propiedad se consigue mediante robots omnidireccionales, el cual es capaz de trasladarse y rotar simultáneamente sobre cualquiera de los ejes del sistema de coordenadas (Ollero, 2001). A continuación, se describe la forma física y las características generales de los principales tipos de robots:

2.3. Uniciclo

Tanto la tracción como el direccionamiento vienen dados por la diferencia de velocidades de ambas ruedas laterales. Para mantener el equilibrio, un par de ruedas castor son comunes en este tipo de robots también conocidos como vehículos de direccionamiento diferencial. Comúnmente, esta configuración es la más común en robots para aplicaciones de interiores. La Fig. 2.10 presenta la disposición de ruedas de los robots móviles tipo diferencial.

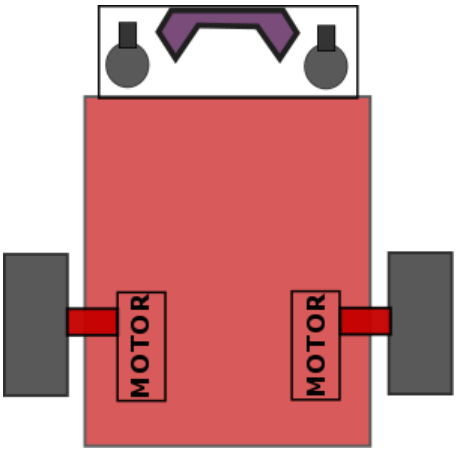


Fig. 2.10 Configuración tipo Uniciclo

Elaborado por: Santiago Álvarez

2.4. Ackerman

Conocido también como car-like, es el modelo utilizado en vehículos de cuatro ruedas convencionales. En efecto, la modificación de este tipo de vehículos (automóviles o de carga) resulta en vehículos robóticos para exteriores. En este diseño, la rueda delantera interior gira un ángulo ligeramente superior a la del exterior para eliminar los deslizamientos. Como consecuencia, la prolongación de

ambas ruedas delanteras intersecta en un punto sobre la prolongación de las ruedas delanteras, como lo indica la Fig. 2.11. Una complicación de este tipo de configuraciones es que se presentan dos ángulos de giro, uno por cada rueda, resultando el control una tarea difícil, por lo que en muchas de las ocasiones se unifica los ángulos de direccionamiento en uno solo.

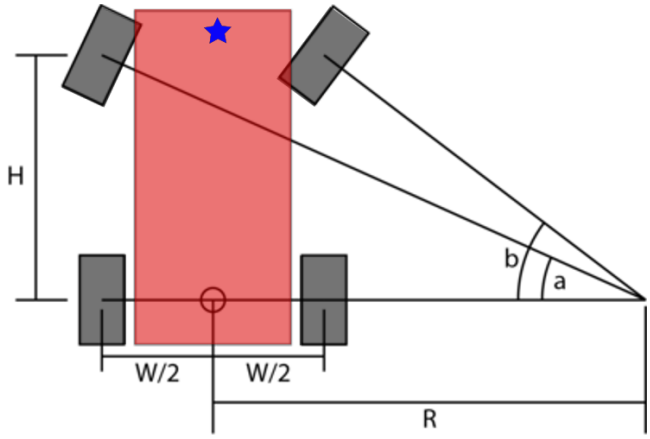


Fig. 2.11 Robot tipo Ackerman

Elaborado por: Santiago Álvarez

2.5. Triciclo

La configuración de triciclo usa la rueda delantera tanto para la tracción como para el direccionamiento. El eje trasero por su parte, incluye dos ruedas laterales que se mueven libremente. En la característica de maniobrabilidad, esta configuración presenta mejores prestaciones que la configuración de car-like. Su principal desventaja se produce ya que el centro de gravedad tiende a desplazarse al exponer al vehículo por una pendiente, por tanto, se pierde tracción. La Figura 2.12 presenta la configuración tipo triciclo.

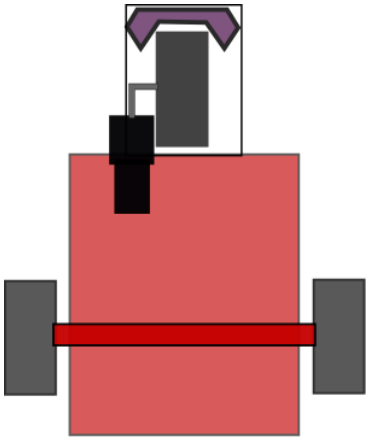


Fig. 2.12 Robot tipo triciclo

Elaborado por: Santiago Álvarez

2.6. Skid Steer

Las aplicaciones de este tipo de configuración solucionan tareas como minería, inspección y obtención de mapas de tuberías enterradas dada la disposición de las ruedas. El movimiento de este tipo de robots es el resultado de combinar las velocidades de las ruedas izquierdas con las de la derecha. La Fig. 2.13 muestra la configuración de este tipo de robots, los cuales se componen de seis ruedas.

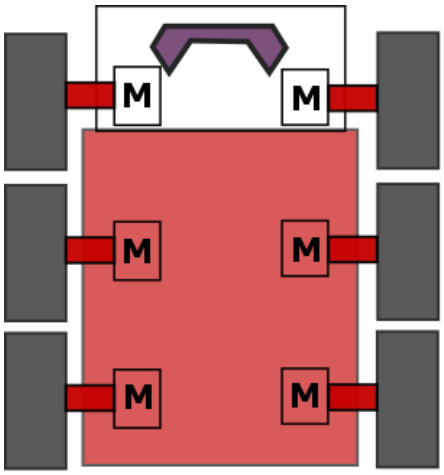


Fig. 2.13 Robot tipo Skid Steer

Elaborado por: Santiago Álvarez

2.7. Omnidireccional

Un robot omnidireccional (Fig. 2.14) es capaz de realizar movimientos en cualquiera de los componentes del plano, ya sean translaciones o rotaciones. Definido también como robot holonómico, el robot omnidireccional requiere de por lo menos tres ruedas especiales, donde el peso debe estar distribuido uniformemente sobre todas ellas. Como la capacidad de movimiento se incrementa, el control de este tipo de configuraciones se hace más complejo debido a posibles deslizamientos resultantes.

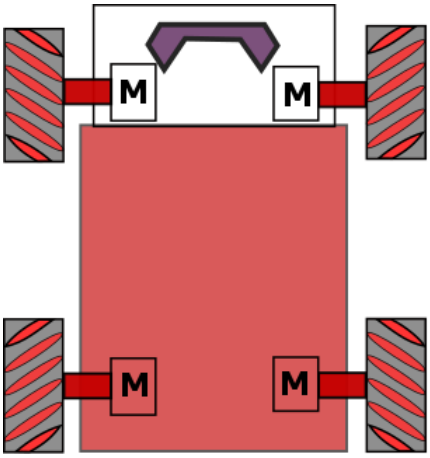


Fig. 2.14 Robot tipo Omnidireccional

Elaborado por: Santiago Álvarez

2.7.1. Algoritmos de control

La autonomía de un robot móvil se basa en el sistema de navegación automática. El control de robots móviles es un área necesaria de la robótica ya que provee seguridad en su movilidad sobre ambientes estructurados o no estructurados. Los principales problemas en el control de movimiento de robots móviles pueden ser clasificados en tres grupos: control de posición, seguimiento de caminos y seguimiento de trayectorias. En el i) control de posición, el objetivo principal es posicionar al robot en un punto de referencia programado, con una orientación final deseada. En el ii) seguimiento de caminos, el objetivo de control obliga a que un robot converja y mantenga el seguimiento de un camino establecido, sin ninguna especificación de tiempo. Finalmente, en el iii) seguimiento de trayectoria, se

requiere que el robot siga una referencia programada pero cumpliendo el seguimiento en periodos de tiempo pre-establecidos (Ollero, 2001).

2.7.2. Navegación de un robot omnidireccional

El control de robots omnidireccionales considera el arreglo de ruedas que posee. Comúnmente, la consideración para desarrollar un controlador apropiado se basa en la teoría de robots holonómicos.

2.8. Control holonómico

En relación a la movilidad, una de las formas de clasificar los robots es distinguiendo si son holonómicos o no. En resumen, los robots o sistemas holonómicos son aquellos capaces de modificar su dirección instantáneamente (lateral, frontal o posterior), sin la necesidad de rotar previamente. Un vehículo tipo car-like, por ejemplo, no es un ejemplo de arreglo holonómico ya que para poder desplazarse en el sentido lateral tiene que realizar maniobras que modifiquen la posición frontal. Del mismo modo, un robot de arreglo diferencial o uniciclo es no-holonómico ya que no puede moverse lateralmente (Suzuki, Blij, & Floreano, 2006).

Frente al sistema de dirección tradicional, el control de dirección de un vehículo es más intuitivo. En el caso de controlar el vehículo por un Joystick, la dirección de avance frontal y longitudinal será comandada directamente por los controladores análogos. Sin embargo, para la programación de un controlador autónomo la dificultad se acrecienta cuando el control no solo se enfoca en la dirección de avance sino en la distancia exacta de traslación. La principal dificultad radica en el tipo de ruedas usadas para lograr la holonomía debido al deslizamiento de las mismas. (Baker & Mackenzie, 2008).

2.9. Control considerando evasión de obstáculos

Una de las tareas más importantes que se debe desarrollar en la mayoría de las aplicaciones de los robots móviles es evitar la colisión con obstáculos que se encuentran en su camino (Fig. 2.15). Para la evasión de obstáculos, se toman en cuenta dos consideraciones: i) un completo conocimiento del área de trabajo y ii) la

inclusión de algún sistema de control que permita al robot móvil modificar su trayectoria en presencia de un obstáculo (Suzuki & Takahashi, 2011).

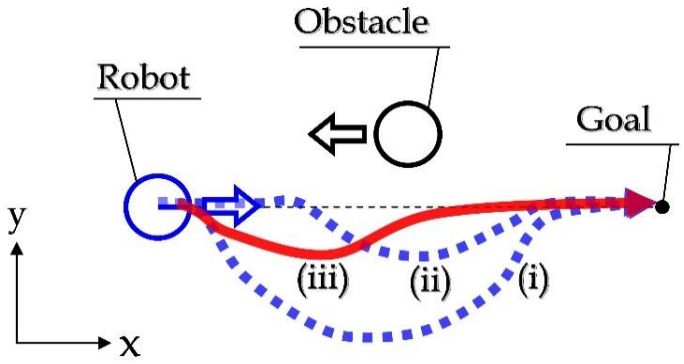


Fig. 2.15 Evasión de obstáculos de un robot

Fuente: Control con evasión de obstáculos **(Suzuki & Takahashi, 2011)**

Evasión de obstáculos mediante procesamiento de imagen: Uno de los métodos para evadir obstáculos se basa en el procesamiento de imagen. Actualmente, el uso de sensores como el Kinect o estereoscópica facilitar la ubicación de elementos que pueden irrumpir en el desplazamiento del robot sobre el espacio de trabajo. Muchos de los trabajos de bibliografía plantean el uso de este tipo de dispositivos más algoritmos de corrección de posición para ejecutar la tarea sin poner en riesgo la integridad del vehículo.

Mediante uso de sensor laser: La evasión de obstáculos requiere como base la detección de los objetos que se encuentran en frente del avance del vehículo. Muchos de los trabajos investigativos hacen uso de sensores laser, los cuales simplifican esta problemática retornando un vector de distancias que indican la proximidad de obstrucciones. Las principales desventajas de este tipo de sensores es el costo económico que representan, además de la cantidad de programación para adquirir adecuadamente la información contenida.

2.9.1. Métodos de localización

La localización de un vehículo robótico en el espacio es necesaria dado que de su posicionamiento puede ser el punto de inicio para la planificación de misiones,

posicionamiento en un punto determinado, seguimiento de trayectorias, etc., brindando retroalimentación al lazo de control. Comúnmente, la posición es determinada por el arreglo de coordenadas que el robot adopta a lo largo del tiempo con respecto a una referencia establecida, haciendo uso de la información adquirida del medio ambiente por medio de sus sensores. Entre los métodos más comunes de localización, dos grupos se presentan: la localización relativa y la absoluta (Corke, 2011).

2.9.2. Métodos de localización relativa

En estos métodos lo localización, la inclusión de sensores en el vehículo es necesaria dado que usan los datos de rotación, g iros o vibración para evaluar una posición.

A. Localización por odometría

Este tipo de localización usa el punto de partida como una referencia para estimar una posición relativa. Para calcular el estimado de movimiento, las ruedas contienen encoders que calculan la velocidad y sentido de rotación para obtener el desplazamiento angular resultante. Su principal limitación se da cuando se producen deslizamientos o la resolución de los encoders no es lo suficientemente adecuada.

B. Localización por navegación inercial

Este tipo de localización usa sensores capaces de medir aceleraciones y vibraciones. A través de cálculo matemático (integrales y derivadas), las modificaciones de movimientos se pueden calcular para estimar el avance o retroceso del robot. Sin embargo, el ruido estacionario puede afectar a largo plazo la precisión de cálculo.

2.9.3. Métodos de localización absoluta

La posición de este tipo de métodos requiere elementos externos de referencia, donde la estimación de posición es significativamente más precisa que los métodos relativos (Sen, 2014).

A. Reconocimiento por señalética

En el reconocimiento por señalética, distintas marcas son colocaras en posiciones determinadas para determinar las características del espacio de trabajo de manera fiable. La estimación de posición con este tipo de método requiere referencias que permitan dimensionar la cantidad de metros recorrida.

B. Correspondencia sobre modelos

Esta técnica funciona comparando entre la información aportada por sensores (incluidos en el vehículo) y la información contenida en bases de datos, mapas o modelos del entorno. La máxima concordancia entre esta comparación determina la posición del móvil en el entorno de trabajo.

C. Posicionamiento por GPS

La localización de equipos externos facilita la detección precisa de ubicación de un robot. El posicionamiento por GPS es el método más conocido entre los métodos de ubicación por rayos de localización activa, ya que basan su funcionamiento por satélites que orbitan el globo terráqueo enviando información constante. Sin embargo, este tipo de métodos no es eficiente para aplicaciones en interiores.

D. Posicionamiento por marcas naturales

Especialmente a gran altura, este tipo de localización usa información del entorno (edificios, casas, vías, etc) para obtener un posicionamiento estimado. Normalmente, la información posee mapas precargados con información etiquetada para lograr la comparación de ubicaciones (Fig. 2.16).

Fig. 2.16 Posicionamiento por marcas naturales.

Fuente: Google Earth (GIJN, 2017)

2.10. Hipótesis

Es posible navegar de manera autónoma usando un robot móvil omnidireccional en trayectorias predeterminadas, considerando controladores para la navegación segura.

2.11. Señalamiento de variables de la hipótesis

Variable independiente: Navegación Autónoma

Variable dependiente: Robot móvil tipo omnidireccional

CAPITULO III
METODOLOGÍA

3.1. Enfoque de la Investigación

El análisis crítico se propone como enfoque de investigación ya que estimará la precisión en la ejecución de tareas por parte del vehículo. Adicionalmente, el análisis propositivo es planteado ya que la obtención de la tesis es requerida. Finalmente, la base teórica del marco investigativo será crucial para la adquisición de los resultados finales, usándose el análisis cualitativo.

3.2. Modalidad de la Investigación

3.2.1. Investigación Bibliográfica – Documental y Experimental

La tesis tiene sustento teórico actualizado de los temas relacionados al proyecto final, por ende, su resultado será la profundización de conceptos desarrollados por bibliografía y el aporte del desarrollador. Adicionalmente, la experimentación y prueba de algoritmos propuestos serán presentadas.

3.2.2. Factibilidad del Proyecto

La propuesta es totalmente práctica y dimensionada al tiempo y recursos disponibles. Adicionalmente, se sustenta con bibliografía científica actualizada que permitirá tener resultados confiables y comprobables.

3.3. Niveles de Investigación

El nivel de investigación logrado será el explicativo debido a que se plantea una solución luego de conocer el problema. Posterior a entender la problemática de los robots con capacidad de navegación autónoma segura, se plantean soluciones que ayuden a cumplir con la tarea del robot.

3.4. Operacionalización de variables

Tabla 3-1:

Variable Independiente: *Navegación Autónoma*

Conceptualización	Dimensión	Indicador	Ítems Básicos	Técnica/ Instrumento
Es el conjunto de sensores, actuadores y equipos que interactúan de manera armónica para cumplir con un objetivo común de navegación. De esta manera, un sistema de navegación está compuesto por elementos que reciben entradas y producen salidas.	-Conjunto de elementos pasivos y activos -Navegación -Elementos que detectan variaciones físicas -Elementos que producen variaciones físicas	Tipo de elementos Estrategia de navegación Tipo de sensor Tipo de actuador	¿Qué tipo de equipos y dispositivos se ocupan para implementar un control para navegación? ¿Qué tipo de estrategia de navegación se usa? ¿Qué sensores se requieren para proveer información del estado del proceso al corrector de errores? ¿Qué tipo de actuadores se incluyen en el proceso para corregir el error de salida?	Técnica: Documental Instrumento: Ficha Bibliográfica

Elaborado por: Santiago Álvarez

Tabla 3-2:

Variable dependiente: *Robot móvil tipo omnidireccional*

Conceptualización	Dimensión	Indicador	Ítems Básicos	Técnica/ Instrumento
Es un robot con características de desplazamiento tipo holonomicas que facilitan el trabajo en espacios estructurados y no estructurados	-Desplazamiento	Velocidad	¿Cuáles son las velocidades requeridas para el movimiento de un robot omnidireccional?	Técnica: Observación
		Características	¿Qué características tiene el robot omnidireccional?	
	-Característica holonómica	Grados de libertad	¿Qué grados de libertad tiene el robot y que significan?	
		Tipo de control	¿Cómo se controla un robot omnidireccional?	
	-Espacio estructurado y no estructurado	Evitación de obstáculos	¿Cómo se evade los obstáculos en un espacio no estructurado?	Instrumento: Guía de Observación

Elaborado por: Santiago Álvarez

3.5. Recolección de información

Posterior a la primera recolección de información que se estime ayudará al desarrollo del proyecto, su depuración para seleccionar únicamente la mejor es necesaria. La recolección de más información puede ser necesaria, la cual deberá estar claramente sustentada y referenciada.

3.6. Procesamiento y análisis

Posterior a la recolección de información, el análisis de toda la bibliografía seleccionada permitirá obtener una perspectiva clara de la investigación a desarrollar. La información obtenida y el desarrollo del trabajo permitirán obtener resultados, los cuales serán analizados para demostrar que se contesta clara y objetivamente la hipótesis formulada. Los resultados adicionalmente permitirán contestar los objetivos planteados, los cuales están relacionados directamente con las conclusiones. Finalmente, se plantearán recomendaciones para extender el trabajo investigativo logrado.

CAPITULO IV

ANÁLISIS E INTERPRETACIÓN DE RESULTADOS

4.1. Análisis de Resultados

4.1.1. Dispositivos para navegación autónoma

La navegación dentro de un espacio determinado o totalmente desconocido es el método empleado para que un robot se guie dentro de su ambiente de trabajo en el cual se puede encontrar diferentes obstáculos presentes en el camino o trayectoria definida para robot móvil.

Los diferentes tipos de robots deben poseer la cualidad de reaccionar ante situaciones inesperadas como principal característica del robot, de una forma adecuada, eficaz, en cualquier tipo de entorno y más que todo sin dejar a parte la tarea propuesta hacia el robot.

Mediante la navegación de un robot se le implantan tareas como percepción de entorno por medio de sensores, planificación de una trayectoria sin obstáculos, posicionarse en un punto adecuado, guiado de vehículo a través de una referencia establecida. El control para la navegación de un robot en se emplean diferentes dispositivos que permiten retroalimentar al robot del entorno y más que todo retroalimentar la posición actual del robot. Se tiene deferentes tipos sensores y dispositivos que emplean los robots para retroalimentar el control de navegación.

4.1.1.1. Sensores Internos

Son sensores que se integran directamente en el robot en la estructura mecánica del robot, este tipo de sensores dan información del estado del robot, principalmente la posición, velocidad y aceleración del robot.

- **Sensores de Presencia:** Sensores inductivos, capacitos, ópticos, ultrasonido, contacto (Fig. 4.1).

a) Sensor Inductivo.

b) Sensor Óptico .

c) Sensor Ultrasónico

d) Sensor de Contacto.

Fig. 4.1 Sensores de Presencia.

Elaborado: Santiago Álvarez,

- **Sensores de Posición y Orientación:** encoder, potenciómetros, IMU, GPS (Fig.4.2).

a) Encoder

b) Potenciómetro

c) IMU

d) GPS

Fig. 4.2 Sensores de Posición.

Elaborado por: Santiago Álvarez.

- **Sensores de velocidad:** Tacómetros.

4.1.1.2. Sensores Externos

Los sensores externos proporcionan información del entorno donde se desplaza el robot, objetos u obstáculos presentes en la tarea del robot. Estos sensores esta ubicados en el entorno de trabajo del robot o alguno en la estructura del robot.

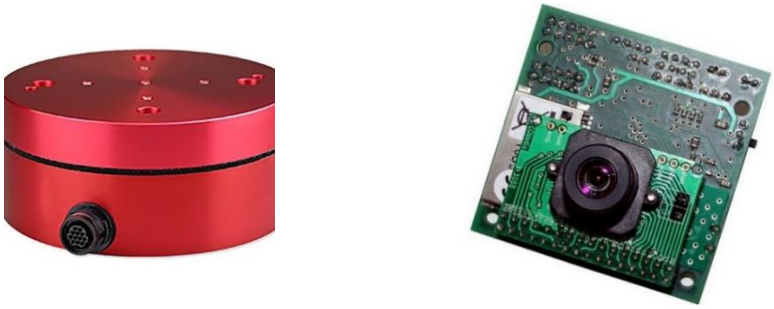
Sensores de Presencia y Proximidad: se tiene sensores Laser que son sensores ópticos de distancia que se basan en determinar el tiempo entre pulsos emitidos y los pulsos recibidos después de ser chocados o rebotados en el objeto. Dentro de este tipo de sensores se tiene los sensores LIDAR (Fig. 4.3). Proporciona distancia y Angulo de ubicación de objetos.

Fig. 4.3 Sensor de Laser.

Fuente: RPLIDAR A2M8 360° (RoboPeak, 2018)

- **Sensores Avanzados:** Imagen y Esfuerzo.

  Los sensores de esfuerzos (Fig. 4.4) se emplean para medir tensiones mecánicas y los de imagen pueden ser cámaras que proporcionan imagen del entorno en donde se encuentra el robot.



a) Sensor de Fuerza b) Cámara

Fig. 4.4 Sensores Avanzados.

Elaborado por: Santiago Álvarez.

Mediante la implementación de sensores y equipos que dan la ubicación exacta del robot se puede desarrollar un control de navegación adecuado, permitiendo al robot desplazarse dentro del entorno que se encuentre. Cada dispositivo incorporado en el robot realimenta el control para compensar el error producido por la dinámica o por el mismo ambiente.

4.1.2. Estrategia de Navegación

Una estrategia de navegación cumple con el objetivo de llevar un robot desde una posición inicial hacia otra final dentro de un ambiente totalmente desconocido o conocido por el robot. Entre las estrategias mayor empleadas para la navegación de un robot se tiene las siguientes (Alfaro, 2006):

4.1.2.1. Percepción del mundo

Se emplea sensores externos, creando un mapa o un modelo del ambiente en el cual se desplaza el robot para desarrollar la tarea de navegación, este tipo de sensores pueden ser sensores laser o detectores de obstáculos.

4.1.2.2. Generación de Camino

Se define en primer lugar una función que escala una sescuncia de puntos construida por un planificador de ruta, posteriormente a esto se realiza una discretizacion del camino con el propósito de crear un camino adecuado para el robot. Existen varios métodos para generar un camino o en otras palabras realizar una Path Planing para obtener el camino óptimo para transportar el robot desde un punto inicial hacia un final.

4.1.2.3. Seguimiento de Camino

Se realiza el desplazamiento del robot, según el camino creado mediante cualquier método de control. El objetivo es seguir un camino a una velocidad definida por el usuario o establecer una ley para que la velocidad variable de acuerdo a parámetros que se requiera.

Los tres tipos de navegación descritos anteriormente son totalmente funcionales y permiten la navegación de robots dentro de cualquier entorno. A más de tener un método de navegación para los robots, se tiene que implementar una ley de control que logre cumplir con la navegación autónoma, existen varios métodos para controlar, estos pueden

ser: PID, lógica difusa, predictivo, control no lineal, y entre otros en la Fig. 4.5 se muestra el esquema de control básico para navegación de un robot.

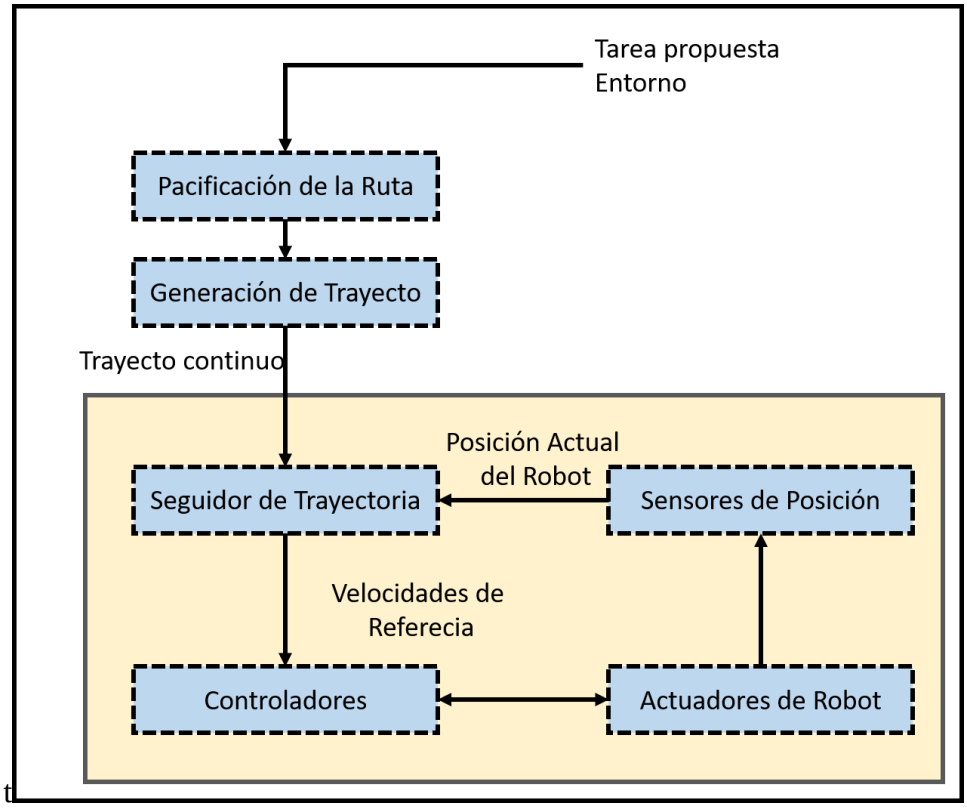


Fig. 4.5 Esquema de control de navegación básica de un robot móvil.

Elaborado por: Santiago Álvarez.

4.1.3. Sensores de Robots Para Control

Los diferentes sensores empleados para obtener información sobre la ubicación del robot en tiempo real son indispensables dentro de la navegación autónoma del robot. Mientas mejor sea el sensor, la información proporcionada será de mejor calidad para dar la ubicación adecuada el robot y lograr corregís errores en la retro alimentación de control dentro de la navegación autónoma. Los sensores más usados que permiten obtener la información del estado del robot son los siguientes.

4.1.3.1. Encoders

Los encoders son empleados para medir la posición relativa de rotación como la distancia recorrida por un eje de motor. Se emplea para realizar una medición de la distancia

recorrida, la dirección del movimiento o la posición de cualquier componente rotatorio, como un brazo robótico o un eje de llanta de un robot. Este método de localización de robots se conoce como edometría sirve directamente para saber dónde está un robot.

Existen dos tipos de encoders los absolutos y los incrementales.

A. Encoder absoluto

Este sensor es empleado para tareas de posicionamiento, son capaces de entregar valores de posición desde el momento que se encienden (Fig. 4.6).

Fig. 4.6 Encoder Absoluto.

Fuente: Encoder absoluto de una vuelta (Allen-Bradley, 2018)

B. Encoder incrementales

Los encoders incrementales (Fig. 4.7) entregan un numero de pulsos por vuelta, esto quiere decir que indican la medida de la distancia angular y lineal recorrida por el robot, debido a un desfase que proporcionan los pulsos generados se puede determinar el sentido de giro.

Fig. 4.7 Encoder Incremental.

Elaborado por: Santiago Álvarez.

4.1.3.2. Unidad de medición Inercia (IMU)

Es un dispositivo electrónico que mide y entrega información de la velocidad, orientación y fuerza gravitacional de un objeto empleando la combinación de acelerómetros y giroscopios, estos sensores son empleados más en los robots de tipo voladores por ejemplo en drones (Fig. 4.8).

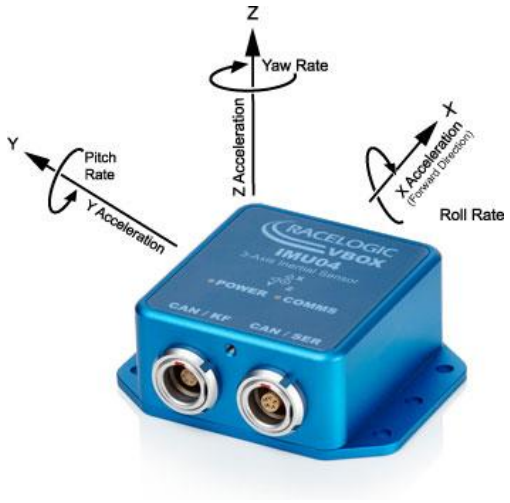


Fig. 4.8 Sensor Inercial.

Fuente: Inertial Measurement Unit (Recelogic, 2018)

4.1.3.3. Giroscopio

Es un sensor mantiene siempre la orientación hacia el norte geográfico, de esta forma el giroscopio indica la orientación preciosa o más bien entrega el desplazamiento que tiene los ejes y da como resultado la posición exacta respecto al punto de partida (Fig. 4.9).

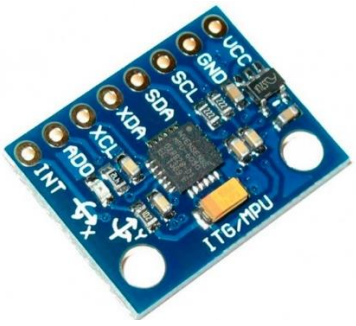


Fig. 4.9 Giroscopio-Acelerómetro.

Fuente: MPU6050 GY-521 Acelerómetro-Giróscopo (PROEMETEC, 2018)

4.1.3.4. GPS (Global Positioning System)

Es un sensor de posición que determina la latitud, longitud, altura y la velocidad del horizonte de cualquier dispositivo individual. Mediante este sensor se puede determinar de un vehículo móvil, de un robot o de un drone ubicado en un lugar remoto (Fig. 4.10).

Fig. 4.10 GPS.

Fuente: GPS 18x 5Hz (GARMIN, 2018)

4.1.4. Actuadores para Corregir Error

Un actuador es un dispositivo que cumple con la función de proporcionar una fuerza adecuada para mover otro sistema mecánico. Los múltiples actuadores usados en robótica dan energía eléctrica, neumático o hidráulica para activar el movimiento de un robot.

Existen varios tipos de actuadores para lograr mover un robot y compensar los errores de navegación estos son:

4.1.4.1. Actuadores Eléctricos

Los actuadores de este tipo los más empleados son los motores de corriente continua DC y los motores paso a paso, gracias a este tipo de motores se logra compensar la movilidad de un robot y lograr desplazarlo sobre cualquier tipo de superficie, siempre y cuando cuenten con dispositivos extras que lo permitan movilizarse.

Los motores de corriente continua regulan la velocidad se emplean para regular la velocidad de un robot con un torque proporcional al voltaje aplicado, estos motores son los más empleados en robots móviles para ser parte dela etapa de control autónomo.

Los motores a pasos existen tanto de corriente continua como alterna, debido a su alta inercia estos motores son difíciles de controlar la velocidad, este tipo de motores se

emplean para controlar el número de vueltas incluyendo hasta centésimas de vuelta para el área de la robótica y la domótica (Fig. 4.11).

Fig. 4.11 Motor de Corriente Continua.

Fuente: Motor DC Reductor 12V (INTPLUS, 2018)

4.1.4.2. Actuadores Neumáticos

Los actuadores neumáticos en robots se usan en el mando continuo, ya que su fuente de energía de estos actuadores es el aire comprimido, esos actuadores se emplean en manipuladores o gripes de precisión (Fig. 4.12).

Fig. 4.12 Actuador Neumático Cilíndrico.

Fuente: Actuadores Cilindros Neumáticos Festo (E3CreaTIC, 2017)

4.1.4.3. Actuadores Hidráulicos

Los acatadores hidráulicos (Fig. 4.13), hacen uso de la energía hidráulica especialmente son análogos, se aplican para movimientos que no sean extremadamente rápidos, una ventaja de estos actuadores es la precisión y repetitividad operando cargas importantes Fig. 4.13.

Fig. 4.13 Actuador Neumático Cilíndrico.

Fuente: Actuadores Neumáticos (S.L., 2018)

4.1.5. Velocidades de un robot omnidireccional

Un robot omnidireccional se caracteriza por su configuración en la cual posee un desplazamiento adicional que los robots de tipo uniciclo.

Para un robot uniciclo o diferencial (Fig. 4.14), las velocidades de control son únicamente la frontal y la angular logrando desplazar al robot hacia delante, atrás y rotar; mediante estas velocidades se implementa algoritmos de control para navegación de robots restringiendo ciertos movientes para realizar.

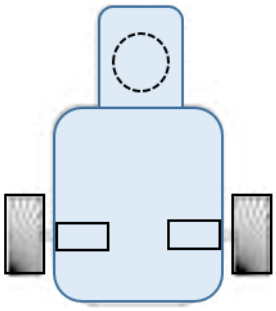


Fig. 4.14 Robot Diferencial o Uniciclo.

Elaborado por: Santiago Álvarez.

La configuración de un robot omnidireccional (Fig. 4.15), permite tener tres velocidades que son la frontal, lateral y una angular. Estas velocidades permiten al robot omnidireccional desplazase con más facilidad dentro de cualquier ambiente ya sea hacia delante, atrás, derecha, izquierda y rotar.

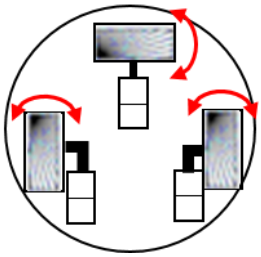


Fig. 4.15 Robot Omnidireccional.

Elaborado por: Santiago Álvarez.

4.1.6. Características de un Robot Omnidireccional

Un robot omnidireccional se caracteriza por su configuración que permite realizar desplazamientos muy diferentes a los de un robot tradicional diferencial. Existen configuraciones diferentes para un robot omnidireccional así también existen tipos de robot que emplean llantas omnidireccionales de diferentes tipos: Omnidireccional con tres ruedas, robot con cuatro ruedas tal y como se indica la Fig. 4.16. Un omnidireccional constituido por cuatro ruedas omnidireccional y de tipo Mecanum tiene la configuración adecuada para desplazar en cualquier dirección tan solo con realizar una mezcla de velocidades en cada rueda se logra tener una navegación más precisa y segura dentro de un entorno no estructurado.

Fig. 4.16 Tipos de robots Omnidireccionales.

Elaborado por: Santiago Álvarez.

4.1.7. Grados de libertad de un robot

Un robot omnidireccional datado por cuatro ruedas omnidireccionales posee una movilidad muy adecuada para desplazarse en varias direccionales esto quiere decir que con cuatro ruedas permite moverse en todas direcciones, por lo tanto, este robot tiene 3

grados de libertad es decir que permite moverse de forma lateral frontal y rotar; es por eso que no presenta ninguna restricción cinemática.

4.1.8. Control de un robot omnidireccional

Existen método de control para robots, empezando por controlarlos de forma manual para poder desplazarlos en las direcciones que permite cada robot. Un robot omnidireccional de igual manera que los muchos robots existentes se pude control de forma autónoma o manual. El control manual se lo implementa para ser manejado mediante las velocidades de comando que este permite, pueden ser ingresadas desde una aplicación remota o una interfaz gráfica.

El control automático de un robot omnidireccional se lo realiza mediante la configuración matemática es decir el modelo matemático que representa el robot, obteniendo de este las velocidades de referencia para insertar en el robot y lograr cumplir con el objetivo del robot. Existen varias maneras de realizar el control de este tipo de robots como es un control PID, control con lógica difusa, control predicativo, control cinemático, control deslizante, entre otros; tomando encuna que cada uno de ellos requiere el modelo representativo del robot omnidireccional.

4.1.8.1. Control autónomo

La cinemática inversa empelada en un control autónomo para navegación implica a que el robot sigue perfectamente las referencias deseadas de la tarea propuesta, es decir en base a la cinemática inversa del modelo matemático del robot entrega las velocidades adecuadas para cumplir con los requisitos de las referencias de la tarea propuesta, sin embargo, pero por condiciones de la dinámica o incluso de la configuración y construcción de robot no se cumplen adecuadamente o satisfactoriamente, es por eso que se desea emplear un controlador que compense ese error producido, para este desarrollo científico se hace uso de un controlador PID el cual se ha probado en diferentes procesos de control, cumpliendo satisfactoriamente con la corrección de los errores que se puede tener a continuación se presenta porque se hace uso de este tipo de controlador para corregir estos errores:

4.1.8.2. Parte Proporcional del (PID)

La etapa proporcional de un controlador es la más fácil, simplemente consiste en un engrandecer con una ganancia predefinida (K_p) variable de acuerdo al requerimiento. La parte proporcional engrandece la velocidad de respuesta y disminuye el error en estado estacionario del robot.

En la3 Fig. 4.17. se presenta la repuesta de la acción proporcional después de un cambio de escalón de entrada de referencia.

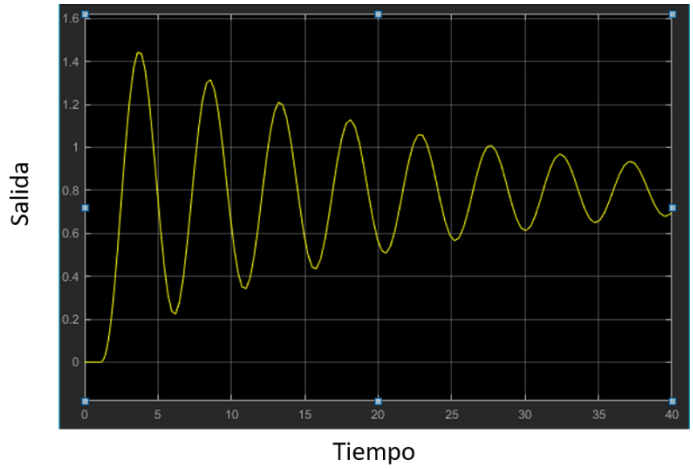


Fig. 4.17 Acción de control proporcional.

Elaborado por: Santiago Álvarez.

4.1.8.3. Parte integral

El objetivo principal de la parte integral es asegurar que la salida del robot coincida con a la referencia en estado estacionario, como se detalló anteriormente, solamente con la acción proporcional se crea un error en estado estacionario. Con esta acción un error en estaco permanente crear un aumento en la señal de control, logrando compensar ese error. Principalmente es una etapa que almacena un historial de la magnitud del error y ayudara a disminuir a cero el error producido, en la siguiente Fig. 4.18 se indica el comportamiento de esta etapa.

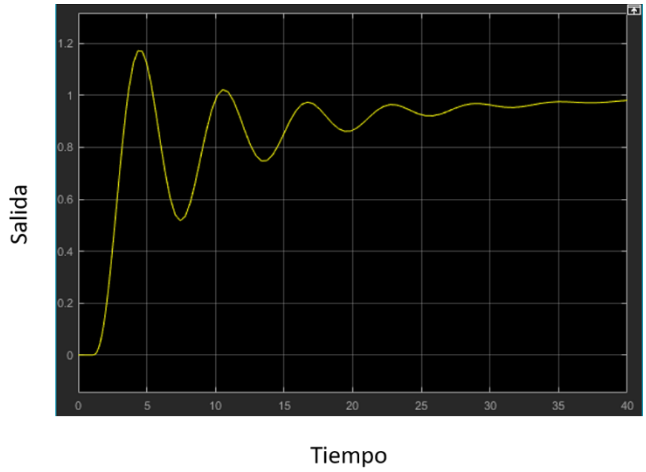


Fig. 4.18 Acción de control proporcional.

Elaborado por: Santiago Álvarez.

4.1.8.4. Acción derivativa

El propósito de la acción derivativa dentro del control PID es mejorar la estabilidad del sistema en un lazo cerrado, es decir, normalmente la acción derivativa se denomina acción de velocidad, porque, aumenta la acción de control y compensación de cada acción que interviene en el PID, cuando el tiempo de la acción derivativa aumenta, existe una inestabilidad Fig. 4.19.

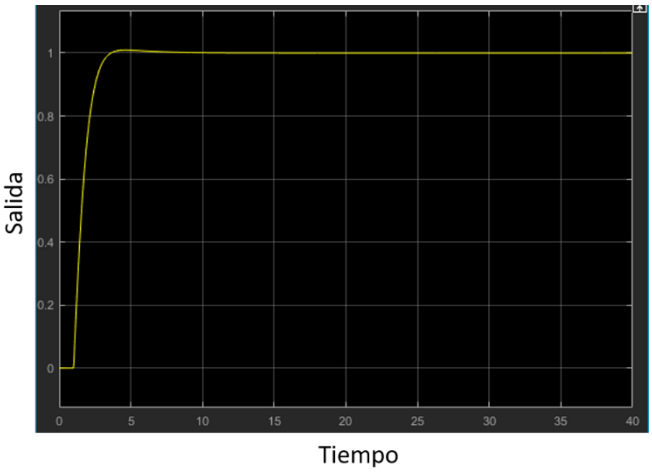


Fig. 4.19 Acción de control proporcional.

Elaborado por: Santiago Álvarez.

4.1.9. Evasión de Obstáculos

Dentro de la navegación de robots en ambientes estructurados o no estructurados se debe tomar encuentra la presencia de obstáculos que no permitan al robot desplazarse de una manera adecuada para cumplir con la tarea puesta. La manera para detectar este tipo de obstáculos es mediante la incorporación de sensores que permitan indicar la presencia o ausencia de un objeto que interrumpa la movilidad del robot.

Dentro de la navegación se puede implementar un control en el cual el robot tenga que tomar una decisión en el momento que un objeto este en su trayecto. Mediante el sensor ya sea de tipo laser o una cámara, se determina la ubicación y la distancia del objeto e inmediatamente se retroalimenta al controlador del robot para tomar una acción correctiva en su trayecto para llegar a su objetivo, una vez evadido el objeto el robot toma nuevamente su ruta para cumplir la tarea.

CAPITULO V

5.1. Conclusiones

- El análisis de los dispositivos que se incorporan dentro de un robot permite conocer y seleccionar cuales son los más apropiados que pueden ser implementados para la navegación autónoma de un robot para cumplir con la tarea propuesta.
- La metodología para utilizar una estrategia de control para la navegación de un robot se lo hace en base a sus aplicaciones y al tipo de robot a ser empleado, la estrategia de navegación se hace en foque al método de desplazar el robot de un lugar hacia otro.
- Los sensores que ocupa un robot para el desplazamiento autónomo en lugares desconocidos son seleccionados en base a su retroalimentación del proceso y que proporcione la mayor cantidad de información del robot ya sea de posición, orientación o la presencia de obstáculos en el camino.
- En el control autónomo para navegación de robots mediante el estudio se conoce que la mayoría de los actuadores existentes para robótica, cuenta con varias formas de actuar sobre el sistema, la mayoría de ellos se adecuan al tipo de robot a utilizar.
- Mediante el previo estudio de la forma de maniobrar un robot omnidireccional con el uso de velocidades lineales, hay que tener en cuenta el tipo de robot y la forma de inyectar las velocidades al robot de tal manera que pueda desplazarse en su entorno.
- Un robot omnidireccional de tipo Mecanum que se desplaza en todas direcciones mediante la combinación de velocidades, pose 3 grados de libertad apropiados para su navegación.
- La configuración de un robot omnidireccional permite establecer la forma como se puede controlar el robot, el resultado de un análisis matemático se obtiene un modelo con el cual cualquier estrategia de control será la adecuada en el momento de ejecutar una tarea para el robot omnidireccional.
- Mediante la problemática planteada se busca solucionar e implementar un control de navegación para robots omnidireccionales con capacidad de evadir obstáculos que se encuentren presentes en el trayecto del robot hacia su objetivo final.

5.2. Recomendaciones

- Dentro de un control de navegación autónoma para robots, mediante el estudio previo se ve la factibilidad de dotar al robot de sensores que le permita deslazarse adecuadamente y conocer mediante sensores su ubicación y orientación en el espacio ubicado.
- Las diferentes estrategias de control existentes sirven para trasladar a un robot de un lugar a otro, observando que la más factible para este tipo de tarea planteada en la presente investigación es la estrategia de seguimiento de trayectoria.
- Para que el control implementado se efectué de una manera idónea se debe considerar colocar sensores de velocidades, para determinar la odometría del robot, mediante esta realimentar al controlador para corregir errores de posición y orientación del robot.
- La salida del controlador genera señales que deben ser ingresadas directamente en los actuadores del robot para desplazarlo y ubicarlo en la posición deseada por la tarea, para este tipo de robot se ve la necesidad de emplear motores de corriente continua, controlados mediante una señal para variar la velocidad y el sentido de giro de cada uno de ellos.
- Para ejecutar el movimiento adecuado de un robot omnidireccional se debe emplear velocidades lineales, lateral, frontal y una angular para lograr desplazar al robot en cualquier dirección combinando cada una de ellas según el valor de referencia que genere el controlador de navegación autónoma.
- Para tener un robot omnidireccional totalmente funcional y que cumpla con los requemamientos de la tarea propuesta se aconseja emplear cuatro ruedas omnidireccionales de tipo mecanum, colocadas en una configuración “AB” para lograr que el robot sea manejable mediante las velocidades que entrega el control autónomo.
- Se recomienda proponer dentro del control de navegación una metodología para evitar obstáculos y que el robot colisione, mediante el empleo de un sensor laser que determine la ubicación de dichos objetos, obteniendo así una retroalimentación al control para saber cuándo el robot debe girar y evitar la colisión.

CAPITULO VI
LA PROPUESTA

6.1. Tema de la Propuesta

NAVEGACIÓN AUTÓNOMA DE UN ROBOT MÓVIL OMNIDIRECCIONAL EN TRAYECTORIAS PREDETERMINADAS CON EVITACIÓN DE OBSTÁCULOS

6.2. Datos Informativos

Ejecutada por: Ing. Santiago Álvarez, tesista de la Maestría en Automatización y Sistemas de Control de la Universidad Técnica de Ambato.

Beneficios: Estudiantes afines a la Maestría en Automatización y Sistemas de Control de Universidad Técnica de Ambato.

Ubicación: Facultad de Ingeniería en Sistemas, Electrónica e Industrial, de la Universidad Técnica de Ambato.

Responsables:

- Ing. Santiago Álvarez (Investigador)
- Ing. Patricio Encalada (Tutor)

Tiempo de ejecución: Marzo 2018 – Junio 2018

Financiamiento: Recursos propios del investigador.

6.3. Antecedentes de la propuesta

De acuerdo a los antecedentes referentes a este trabajo investigativo, se sabe que existen muy pocos relacionados a un control de un robot omnidireccional para navegación segura, es decir robots con control auto y evasión de obstáculos en una tarea dada, logrando culminarla sin ningún problema en un ambiente que tenga obstáculos en la trayectoria trazada, en tiempos pasados esto no era posible ya que por una parte no se tenía robots con una locomoción muy fluido como es el caso de un omnidireccional con ruedas Mecanum en configuración "AB" que le permite moverse en todas las direcciones, entonces al no poseer restricciones holonómicas se puede cumplir con rutas muy exigida para navegar en cualquier entorno, facilitando la carga o transporte de elementos. Actualmente gracias a los avances tanto en el desarrollo tecnológico se tiene sensores dotados de cualidades que permiten al robot obtener información de un ambiente que no conoce, entonces se puede crear tareas en las que el robot se deslice sobre superficies que

tengan obstáculos desarrollando algoritmos autónomos que permitan una navegación segura sin que el robot colisiones y pierda su objetivo planteado.

6.4. Justificación

Mediante el gran desarrollo de la investigación en la robótica y creación de robots totalmente autónomos, se ve la exigencia de desarrollar y unificar dispositivos tecnológicos que trabajen de manera conjunta con un robot. En los últimos años en nuestro país se ha puesto gran interés en crear sistemas de automatización e incluso la incorporación de robots que faciliten la vida cotidiana en cualquier campo aplicativo, llevando así a investigadores a desarrollar y buscar nuevas alternativas para ayudar a la personas o empresas por medio de la robótica dentro del país.

Por ultimo, la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, de la Universidad Técnica de Ambato está en constante mejoría y en busca de la excelencia, por lo que el desarrollo de esta propuesta investigativa y experimentar, incentiva a planteamientos muy parecidos con fines para continuar con nuevas investigaciones en el campo de la robótica.

6.5. Objetivos de la propuesta

6.5.1. Objetivo general

Implementar un robot omnidireccional con su modelo matemático para desarrollar el algoritmo de control que permita la navegación autónoma segura.

6.5.2. Objetivos específicos

- Construir un robot omnidireccional con 4 ruedas mecanum en configuración “AB” para desplazarse en cualquier dirección.
- Obtener el modelo matemático del robot omnidireccional analizando su configuración geométrica en el plano de referencia para determinar la cinemática en el punto de operación de interés.
- Dotar de un sensor laser al robot omnidireccional para obtener información del entorno de navegación mediante la detección de objetos presentes.
- Proponer un algoritmo de control para la navegación autónoma del robot omnidireccional mediante el modelo cinemático.

- Asegurar la navegación del robot omnidireccional mediante la detección de obstáculos mediante el sensor laser LIDAR para evitar colisiones en la trayectoria predefinida.
- Realizar pruebas experimentales para evaluar el funcionamiento del control autónomo de navegación segura del robot omnidireccional en un entorno semi-estructurado.

6.6. Análisis de factibilidad

Esta propuesta tiene factibilidad de ejecución debida que existen los recursos técnicos, operativos y económicos para desarrollar y adquirir los materiales necesarios.

6.6.1. Factibilidad operativa

Se cuenta con herramientas y métodos matemáticos para desarrollar un algoritmo de navegación autónoma segura, conjuntamente con un software apto para los cálculos respectivos durante la experimentación.

6.6.2. Factibilidad técnica

Se tiene los conocimientos técnicos adecuados, científicos e investigativos para desarrollar la presente propuesta y ejecutar cualquier solución frente a inconvenientes que se pueda presentar durante el desarrollo.

6.6.3. Factibilidad económica

El presente trabajo experimental se requiere de costos mínimos de cubrir por el investigador en adquirir y construir el robot para realizar pruebas experimentales y evaluar el desempeño del controlador desarrollado.

6.7. Fundamentación científico – técnica

6.7.1. Que es modelo cinemático

La cinemática de un robot analiza el movimiento sin considerar las fuerzas que lo generan, en otras palabras, estudia las leyes de movimiento sin tomar en cuenta las masas y pesos del robot. Estudia analíticamente la geometría o configuración que representa al robot con respecto a una referencia fija de un sistema de coordenadas.

6.7.2. Que representa el modelo cinemático

El modelo cinemático representa un modelo matemático que describe al sistema en forma de ecuaciones que permiten conocer el comportamiento de un robot en cuanto, a movimientos o desplazamientos, mediante este análisis se pude determinar posición, velocidad e incluso aceleración de cada uno de los elementos que conforman el robot, sin tener en cuenta la fuerza que genera estos movimientos.

Para definir la cinemática de un robot se debe tomar en cuenta algunas observaciones como:

- Se debe estudiar y definir bien el espacio de trabajo del robot móvil.
- Saber adecuadamente cuáles son las velocidades para que el robot se desplace en su lugar de trabajo.
- Establecer bien los grados de libertad que tiene el robot.

Existen dos tipos de cinemática, la directa y la inversa Fig. 6.1.

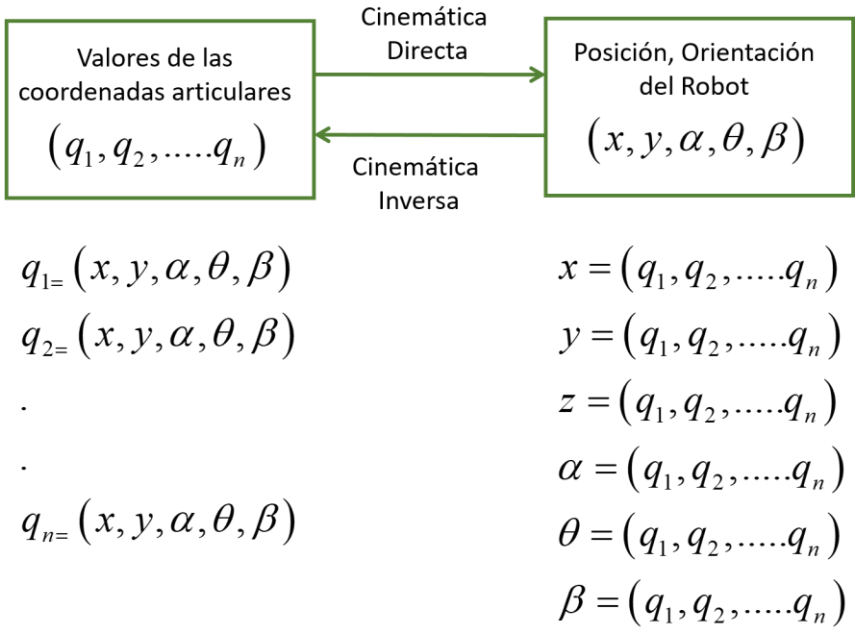


Fig. 6.1 Cinemática Directa e Inversa.

Elaborado por: Santiago Álvarez.

Donde $q_1, q_2, ...q_n$ representan los valores de cada extremidad del robot, es decir, en un brazo robótico representan los valores de los ángulos entre cada eslabón del brazo y $x, y, z, \alpha, \theta, \beta$ representa los valores del extremo operativo del robot, el cual dan la

posición en la que se encuentra en función de cada valor de los eslabones que o conforman.

6.7.3. Cinemática Directa

La cinemática directa son modelos matemáticos o ecuaciones que permiten calcular la posición de cada eslabón de un robot en base a sus componentes fijos, en otras palabras, sirve para el cálculo de la posición del actuador final o la posición y orientación de la base fija de un robot móvil a partir de la posición y velocidades de las ruedas.

6.7.4. Cinemática Inversa

Es un proceso en el cual se obtiene ecuaciones matemáticas que permiten conocer a partir de una posición del actuador calcular posición o desplazamientos de los actuadores. La mayor parte de los casos, es muy complejo encontrar el modelo cinemático inverso por lo cual se hace uso de la pseudo-invera de la matriz Jacobiana. La matriz Jacobiana determina las relaciones entre las velocidades del movimiento de cada articulación y las de un extremo del robot.

6.7.5. Cinemática de un robot móvil

El presente trabajo de investigación y experimentación de navegación autónoma segura para un robot omnidireccional, requiere controlar las velocidades y la posición de la plataforma robótica, es decir, se debe obtener un modelo matemático (cinemático) que ayude a interpretar las variables involucradas en el control, el modelo cinemático del robot está representado por las velocidades de maniobrabilidad de la plataforma, para obtener la posición y rotación de la plataforma en cualquier instante de tiempo dentro del marco referencial, para ello se debe realizar las velocidades (Siciliano & Khatib, 2008).

El modelo cinemático del robot móvil omnidireccional está compuesto por un sistema de ecuaciones conformadas por tres elementos representativos, $\dot{\boldsymbol{\delta}}(t)$ que es el vector que contiene las velocidades lineales y angulares en el punto de operación del robot omnidireccional, el vector $\mathbf{v}(t)$ que contiene las velocidades lineales y la angular de maniobrabilidad del robot omnidireccional y por último la matriz $\mathbf{M(\theta)}$, esta matriz representa una rotación homogénea del robot es decir esta matriz de transformaciones que

relaciona los estados de velocidades anteriores, también se la conoce como matriz jacobiana.

El modelo cinemático nos ayuda a obtener la posición y orientación del robot a cada instante de tiempo, lo que nos permite evaluar sus movimientos.

6.7.6. Control de desplazamiento

Con el fin de desplazar el robot omnidireccional se emplea un control en lazo cerrado que permita cumplir con la tarea deseada, corrigiendo errores en cada instante de tiempo por medio de la retroalimentación de velocidades. Esta ley de control deberá lograr que las velocidades $\mathbf{v}(t)$ del robot cumplan con las velocidades propuestas por la tarea a desarrollarse. Para cumplir esto se realiza el siguiente procedimiento de control.

- Identificar obstáculos en el trayecto.
- Modificar la posición deseada.
- Determinar el error entre el valor deseado por la tarea y el valor actual.
- Obtener la resolución del modelo cinemático inverso del robot.
- Compensar los errores producidos por la cinemática empleando un controlador.
- Obtener la cinemática directa para ver posición y orientación del robot en el plano de referencia (odometría).

El presente esquema de control será empleado para realizar la navegación segura del robot omnidireccional en espacios semiestructurados Fig. 6.2.

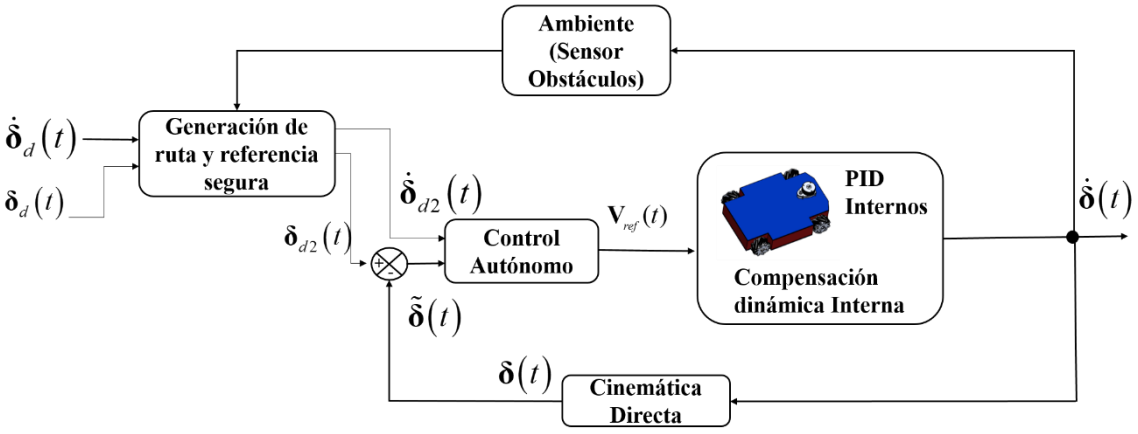


Fig. 6.2 Esquema de control para navegación de un robot Omnidireccional.

Elaborado por: Santiago Álvarez.

Como se puede observar el esquema de control detalla los pasos descritos anteriormente para realizar el control adecuado de la navegación autónoma y segura de un robot omnidireccional.

6.7.7. Odometría del Robot

Para la determinación de la posición del robot omnidireccional se emplea el sistema de medición en base a las velocidades de cada rueda del robot omnidireccional, estimando y teniendo en cuenta que se puede tener errores mínimos en distancias cortas de locomoción y mientras más grande sea la distancia recorrida por el robot, el error acumulado va ser representativo. En otras palabras, el proceso más sencillo y más utilizado es la localización de un robot en base al modelo cinemático del sistema de propulsión. La estimación de la posición de un robot omnidireccional se puede calcular mediante el sistema de ecuaciones que definiremos en el presente tema investigativo en función de la relación que hay en entre los motores del robot omnidireccional y los encoders acoplados al a las ruedas del robot.

6.8. Metodología

6.8.1.1. Diseño Mecánico

En consideración al estudio de múltiples robots existentes de tipo omnidireccional se establece una forma y tamaño adecuado para construir un robot omnidireccional, el cual fue diseñado preliminarmente en un software CAD (Solid Works), este software permite diseñar piezas y obtener planos de construcción con medidas reales, en la Fig. 6.3 se muestra el robot diseñado preliminarmente a su construcción.

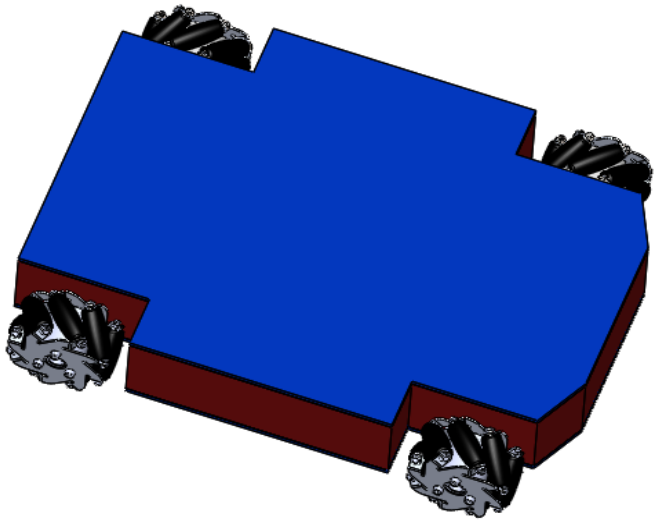


Fig. 6.3 Diseño CAD del robot omnidireccional.

Elaborado por: Santiago Álvarez.

Los planos generados para la construcción mecánica de la estructura del robot se obtienen mediante el coquizado de piezas dentro del software CAD, aquí se realiza los planos de construcción de cada parte que conforma el robot.

En la Fig. 6.4 se indica las dimensiones establecidas para la construcción del robot, acotadas correctamente en el croquis diseñado.

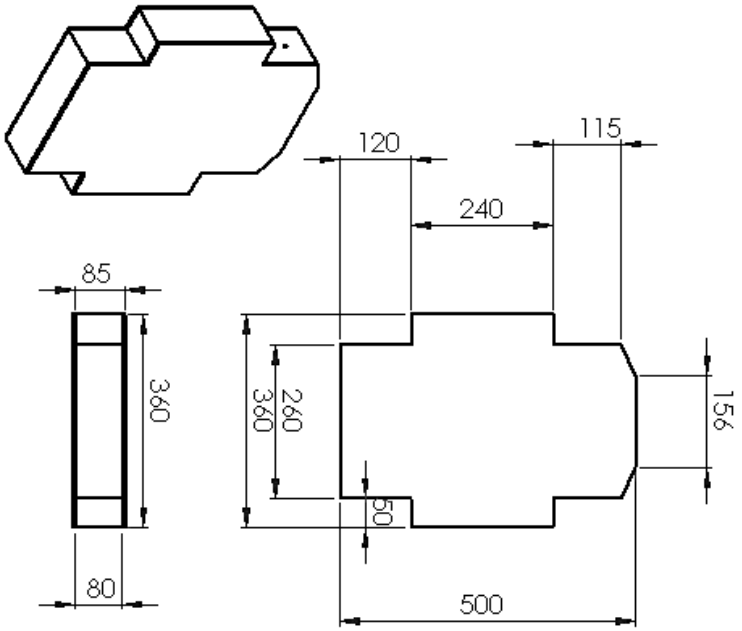


Fig. 6.4 Dimensiones del robot omnidireccional.

Elaborado por: Santiago Álvarez

En consideración al tipo de ruedas empleadas, se emplea ruedas de tipo Mecanum, con un radio de 10mm, Fig. 6.5.

Fig. 6.5 Ruedas Mecanum derecha e izquierda..

Elaborado por: Santiago Álvarez

Dentro de las diferentes configuraciones del robot tipo omnidireccional Mecanum se establece la configuración tipo AB. En el **Anexo 1** se presenta los planos y dimensiones del robot omnidireccional para la construcción.

6.8.1.2. Análisis de esfuerzos de la estructura del Robot

Posterior al diseño y coquizado de piezas del robot se realiza el análisis de fuerzas para, determinar cuál es el peso máximo q soporta la estructura del robot, logrando así especificar de qué material será construido. El análisis de la estructura mecánica y determinación del material se lo realiza en un software específico, como es el software ANSYS.

Primeramente, se realiza un mallado de toda la estructura del robot en el Software ANSYS, una malla es como un conjunto organizado de puntos formado por las intersecciones de las líneas de un sistema de coordenadas. En la Fig. 6.6 se muestra el mallado creado en el Software de la parte estructural del Robot Omnidireccional que será construido con un material de aluminio de 3mm.

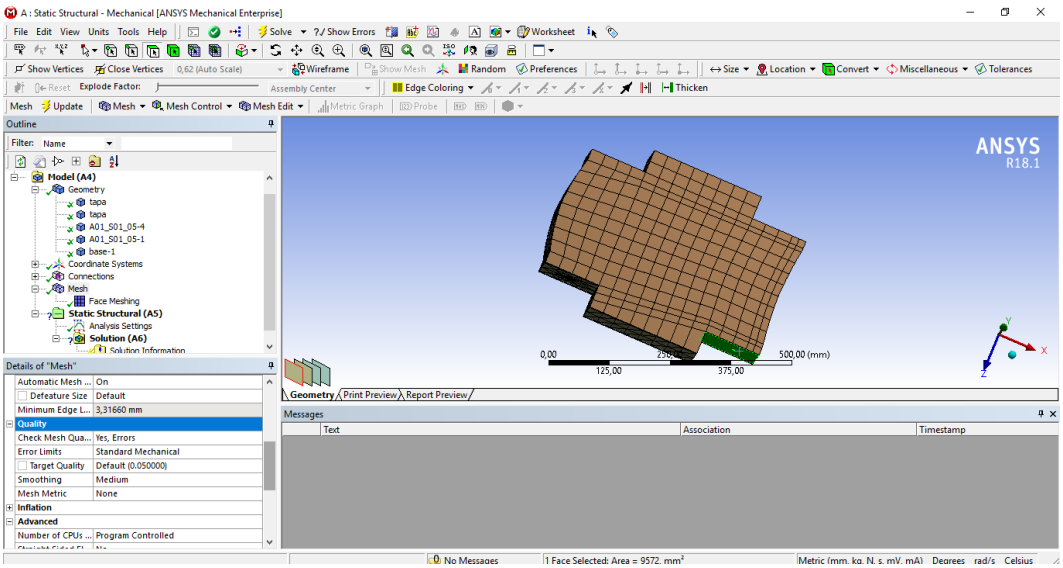


Fig. 6.6 Mallado de la estructura en ANSYS.

Elaborado por: Santiago Álvarez

Para continuar con el análisis se establece colocar soportes y la gravedad de 9,8 m/s^2 en el eje $-y$ tal y como se muestra en la Fig. 6.7.

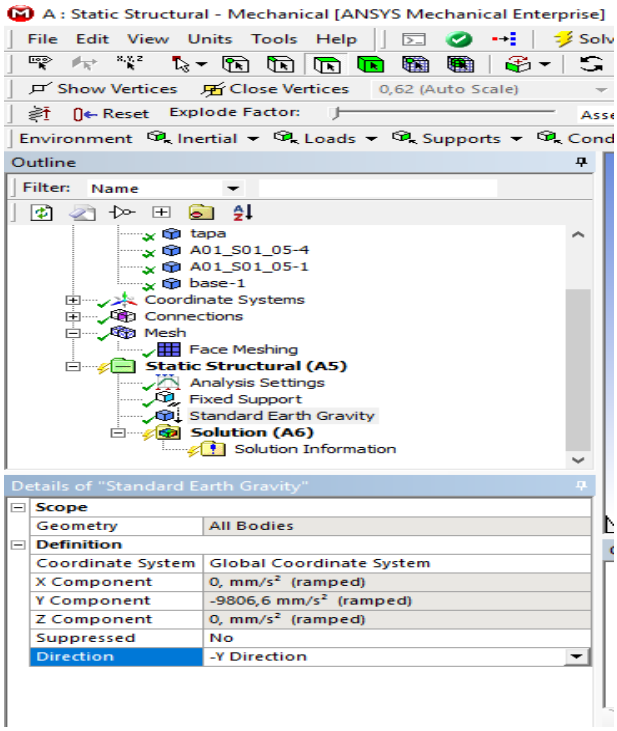


Fig. 6.7 Asignación de soportes y gravedad para el análisis.

Elaborado por: Santiago Álvarez

Se realiza el análisis de deformación en los soportes asignados previamente y con la gravedad normal, observando en la Fig. 6.8 que el resultado de deformación mínima es

de 0,0039 mm que ni al caso no llega al milímetro, se acepta que el material de construcción es el adecuado.

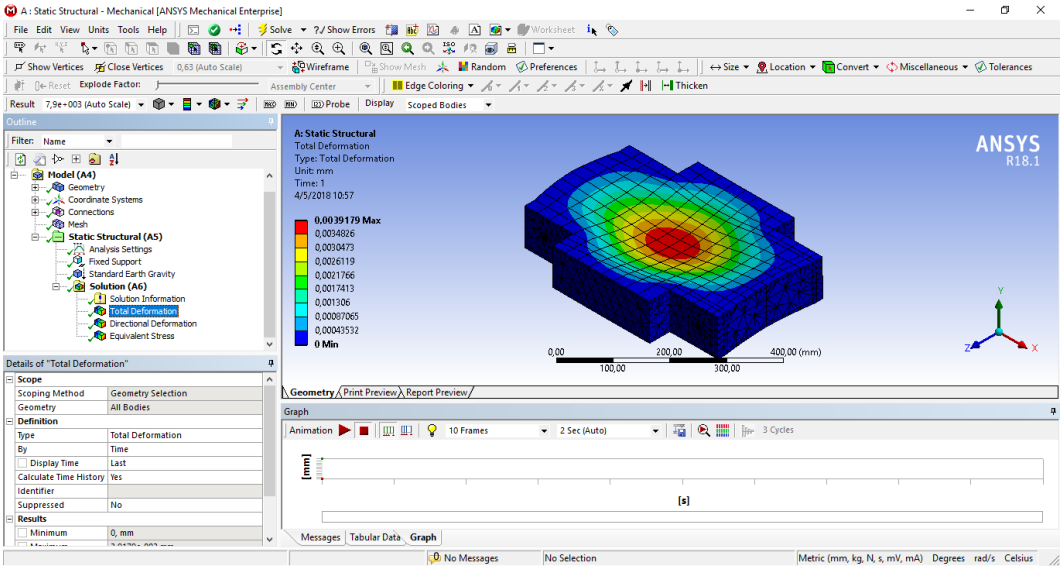


Fig. 6.8 Análisis de deformación en la estructura del Robot.

Elaborado por: Santiago Álvarez

En la Fig. 6.9 se observa la deformación direcciona en el eje y donde no llega ni al milímetro

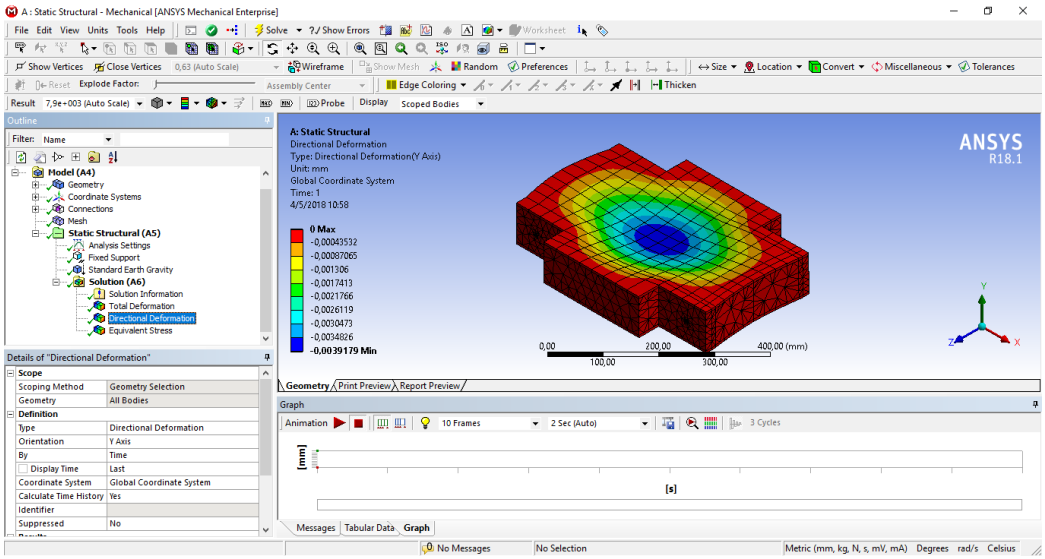


Fig. 6.9 Análisis de deformación en la estructura del Robot.

Elaborado por: Santiago Álvarez

La tensión de Von Mises es una magnitud física que es proporcional a la enérgica de distorsión, para el análisis se lo considera como indicador de un buen diseño para

materiales hechos con las aleaciones metálicas o materiales asfálticos. En la Fig. 6.10 se denota que el equivalente de von Mises es de $\sigma = 0.98MPA$.

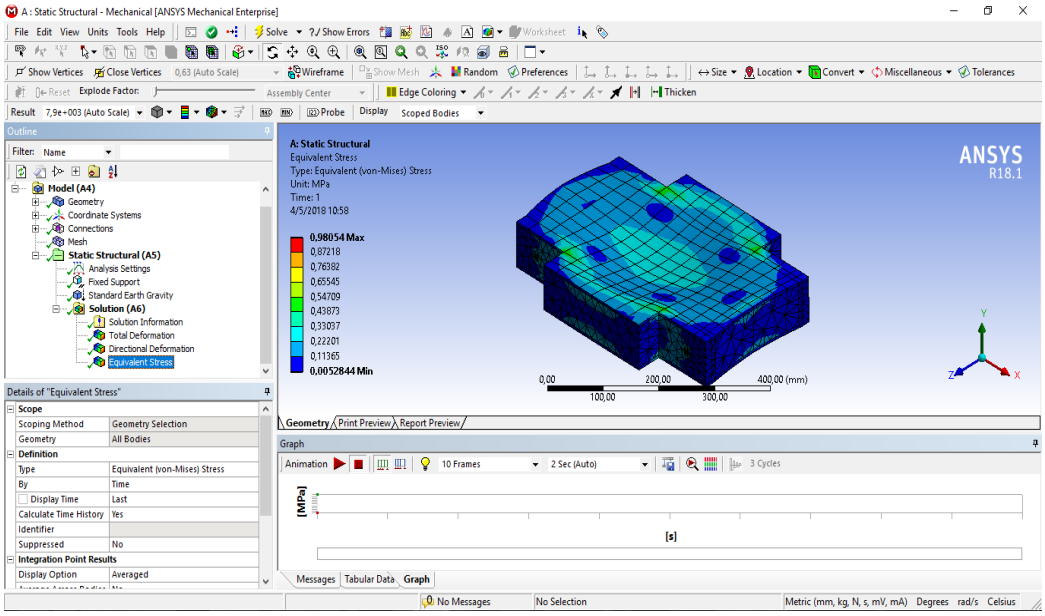


Fig. 6.10 Análisis de tensión de Von Mises.

Elaborado por: Santiago Álvarez

Con la siguiente formula de determina que S_Y/g mayor que un factor de seguridad de 5:

$$\frac{S_Y}{\sigma} \geq 5 \qquad (6.1)$$

En el caso de esta construcción es usa el aluminio con: 95Mpa en (6.1)

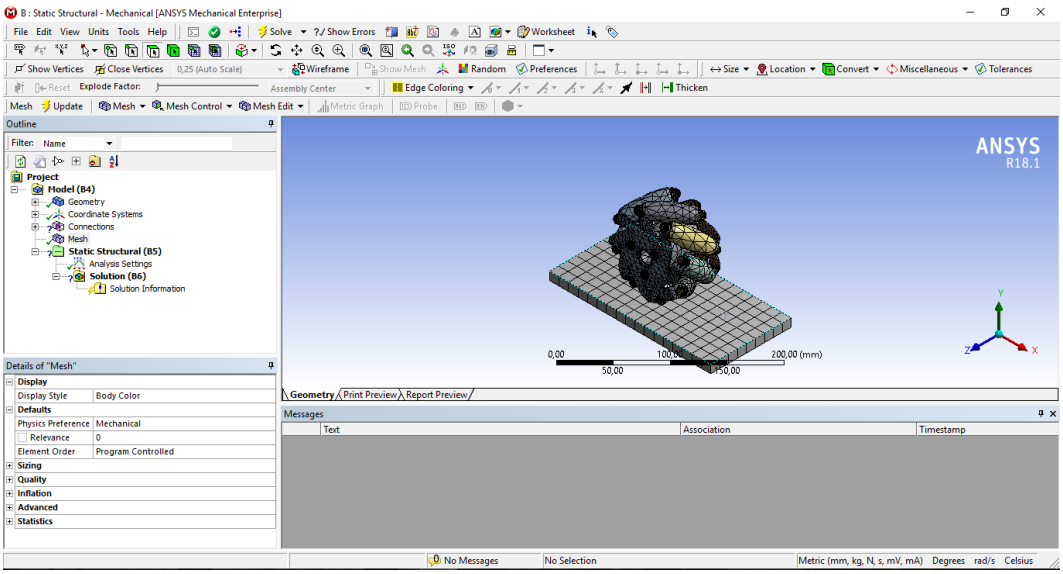
$$\frac{95M_{pa}}{0,98M_{pa}} \geq 5$$

Debido a la ecuación, se obtendrá un buen factor de seguridad lo cual la estructura está bien diseñada.

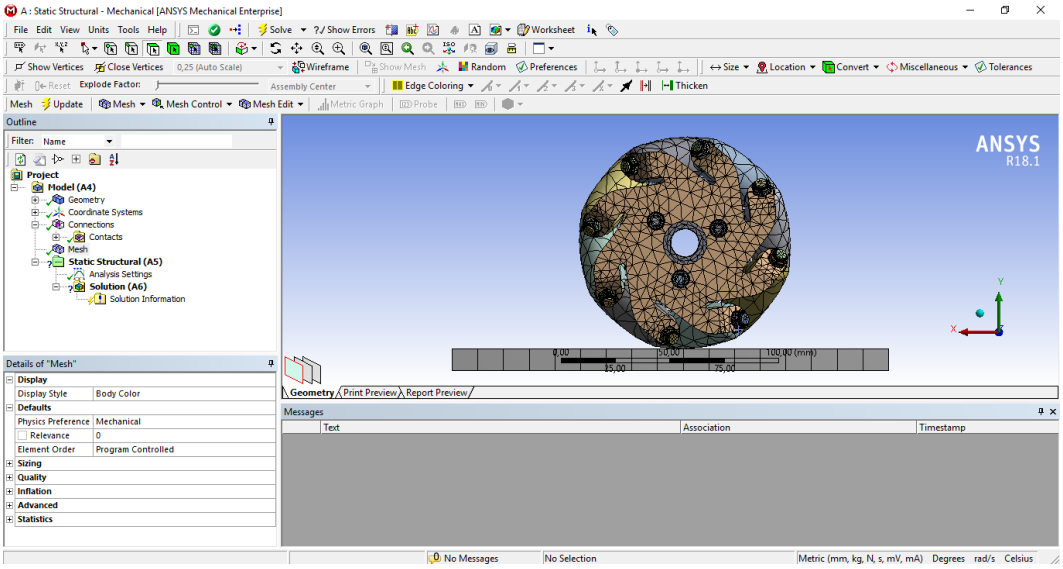
6.8.1.3. Análisis de las Ruedas del Robot

El análisis de contacto de las ruedas se lo divide el peso total del robot para cuatro en la que cada rueda va a soportar 1,75kg.

En la siguiente figura Fig. 6.11 se muestra el mallado de la rueda para realizar el análisis de contacto de cada rueda.



a) Vista de total del mallado de la rueda.



b) Vista lateral del mallado de la rueda.

Fig. 6.11 Mallado de la rueda Mecanum.

Elaborado por: Santiago Álvarez

Se procede a dar las características para ejecutar el análisis de fuerzas en la llanta omnidireccional Fig. 6.12.

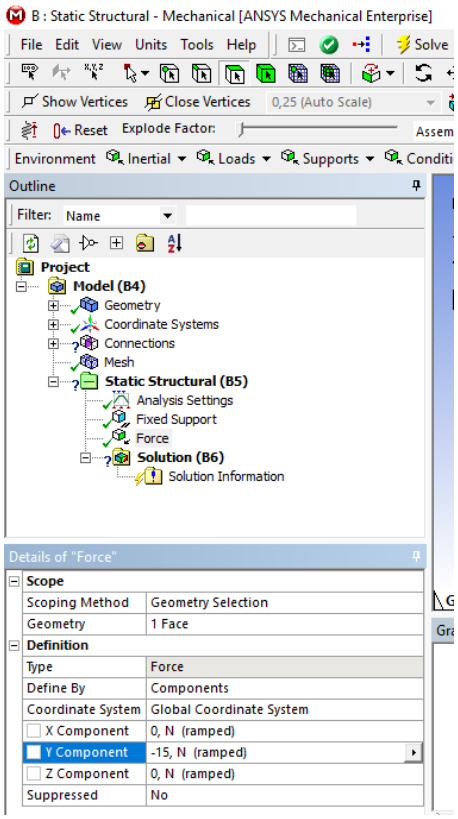
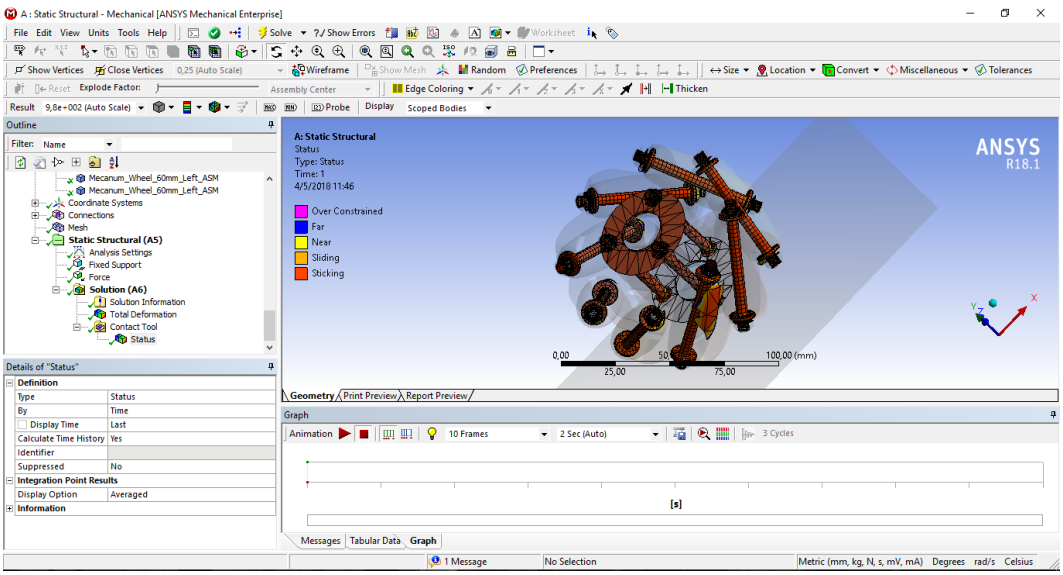


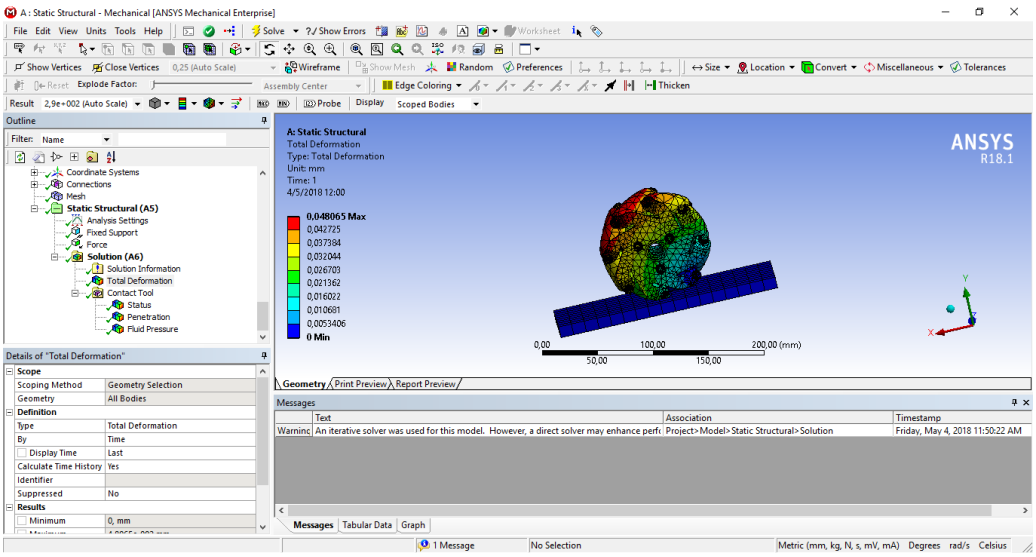
Fig. 6.12 Características para análisis de fuerzas en la rueda.

Elaborado por: Santiago Álvarez.

En la siguiente figura tenemos el estado de presión la rueda por medio del análisis de contacto Fig. 6.13.



a) Contacto de la rueda con el suelo



b) Presión ejercida de la rueda sobre el suelo

Fig. 6.13 Análisis de presión por contacto.

Elaborado por: Santiago Álvarez

6.8.1.4. Diseño Electrónico del Robot

Para el diseño electrónico del robot Omnidireccional se establece características de hardware que permita al robot comunicarse de manera inalámbrica o de forma directa con un computador, en el cual se encentra el software matemático para realizar el control autónomo. En Fig. 6.14 se indica un diagrama de bloques del hardware que conforma el robot Omnidireccional.

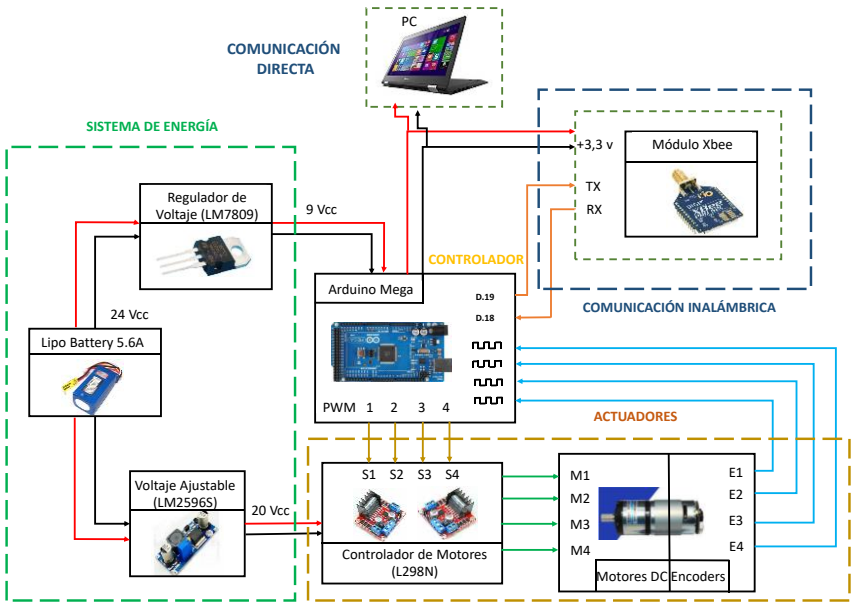


Fig. 6.14 Esquema del Hardware del robot omnidireccional..

Elaborado por: Santiago Álvarez

6.8.1.5. Sistema de Energía

Para lograr el correcto funcionamiento del Robot y de una forma autónoma se ve la necesidad de contar con un sistema que entregue los niveles de voltaje adecuado para cada etapa que conforma el Robot. El sistema de energía es compuesto por 3 dispositivos, la batería de alimentación, y dos reguladores de voltaje de corriente continua. La batería del sistema es una LIPO de 5.6A de 6s que totalmente cargada entrega un voltaje de 24 Vcc a esta salida de alimentación se agrega un regulador a 9Vcc (LM7809), este voltaje sirve para la alimentación de la placa controladora, por último, se tiene un conversor DC-DC (LM2596D) que es un dispositivo que reduce el voltaje de forma variable, para la alimentación de cada motor se emplea una regulación a 20Vcc para alimentar cada driver del motor.

6.8.1.6. Controlador

El controlador empleado para comandar el robot y ejecutar las acciones adecuadas para medir los sensores de velocidad incorporados en cada rueda del robot, es una tarjeta Arduino Mega 2560 que está conformada por 54 pines de Input/Output digitales, 16 entradas analógicas. Esta tarjeta se encarga de realizar el control interno del robot configurándole para que pueda recibir como entrada velocidades lineales que serán inyectas externamente. En esta tarjeta se encarga de enviar señales de control para cada motor mediante PWM método por el cual se varia la velocidad de cada rueda obteniendo como resultado final velocidades angulares por cada, estas velocidades serán enviadas hacia el software matemático para realimentar el control.

En el desarrollo del controlador se emplea internamente una compensación para corregir errores de velocidades aplicadas al robot. Para lograr que el robot se ajuste a las velocidades que se le envía se emplea controles PID los cuales sirven para controlar cada rueda independiente, obteniendo así una compensación de la dinámica del robot.

6.8.1.7. Control PID interno

Se propone un control interno de tipo PID dentro de la plataforma robótica con el propósito de compensar la dinámica del robot, logrando d esta manera que el robot este a las velocidades deseadas por el usuario o per el control de navegación Fig. 6.15.

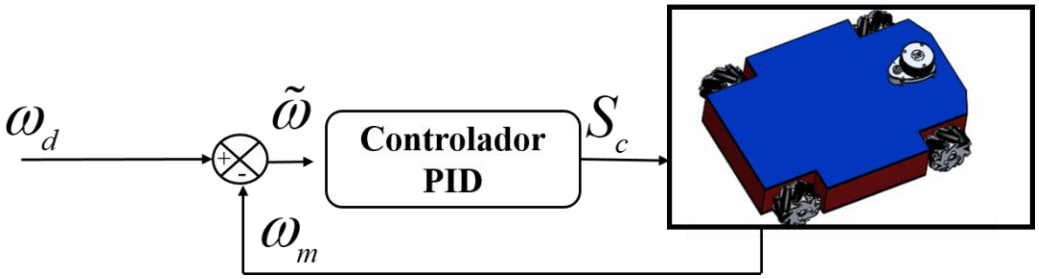


Fig. 6.15 Control interno de la plataforma Omnidireccional.

Elaborado por: Santiago Álvarez

El controlador PID propuesto está conformado de la siguiente manera:

$$S_{ci}(t) = kp_i\tilde{\omega}_i(t) + ki_i\int_0^t \tilde{\omega}_i(t)dt + kd_i\frac{d}{dt}\tilde{\omega}_i(t) \tag{6.2}$$

donde $\tilde{\omega}_i$ representa el error de la velocidad angular en la rueda i con, $i = 1,2,3,4$ para cada robot omnidireccional y está definida como $\tilde{\omega}_i(t) = \omega_{di}(t) - \omega_{mi}(t)$, ω_{di} representa la velocidad angular deseada para la rueda del robot y ω_{di} presenta la velocidad angular medida y por ultimo kp_i, ki_i, kd_i son las ganancias positivas del control PID. Mediante este control se logra cumplir con la velocidad que se desea en cada rueda del robot omnidireccional.

6.8.1.8. Actuadores

El bloque de los actuadores está conformado por sus respectivos controladores, los cuáles se encargan de manejar la pare de motores. Los motores empleados para la construcción de robot omnidireccional son de tipo DC incorporados en ellos sensores encoders que entregan una cierta cantidad de pulsos para determinar la velocidad de cada uno de ellos. Estos actuadores DC funcionan con un voltaje de 20 Vcc que entrega el sistema de energía, voltaje a adecuado para lograr mover al robot y generar una velocidad idónea para cumplir con la tarea planificad, el motor tiene una relación de relación 27:1 apto para 21 Kg/cm y con velocidad máxima de 355 rpm Fig. 6.16.

Fig. 6.16 Motor de corriente continua con encoder.

Elaborado por: Santiago Álvarez

El controlador de los motores (Dual Driver) L298N (Fig. 6.27) cuenta con las siguientes características

- Voltaje de alimentación, mínimo de 5 V.
- Posee dos entradas, una de 5V para controlar la parte lógica y otra para alimentar las salidas al motor, que pueden ser de 5Vcc o más voltaje.
- La tarjeta tiene la opción de habilitar un regulador LM7805 integrado en ella para alimentar la parte lógica directamente.
- Permite controlar dos motores, con una corriente máxima de 2 A por canal.

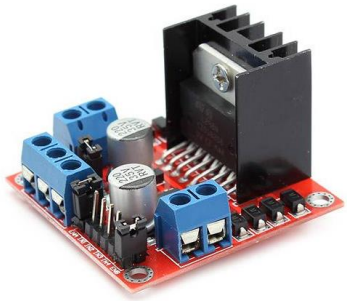


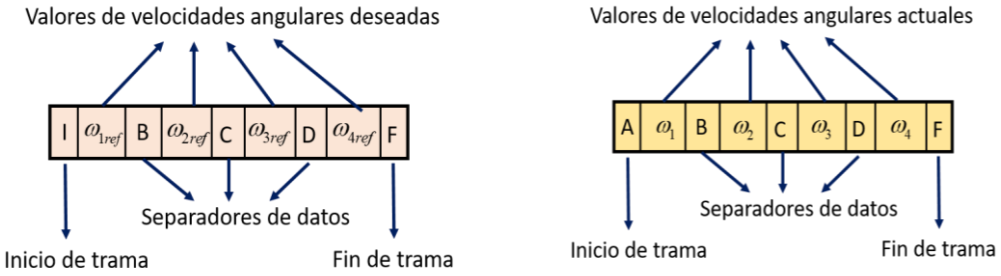
Fig. 6.17 Driver dual L298N.

Elaborado por: Santiago Álvarez

6.8.1.9. Comunicación

El sistema de comunicación se hace por medio de módulos inalámbricos XBEE S2, este tipo de dispositivos inalámbricos utilizan el protocolo IEEE 802.15.4 para crear redes punto a multipunto o para redes punto a punto dependiendo del requerimiento y la configuración de los módulos. Son aptos para transmitir gran cantidad de daros con baja latencia y una sincronización de comunicación predecible. Mediante este medio de enlace inalámbrico se puede comandar el robot inyectando velocidades angulares y recibiendo

el estado actual de cada rueda en cuanto a velocidad. Para la comunicación se emplea mediante tramas de datos la de recepción y la transmisión las cuales están conformadas como se muestra en la Fig. 6.18.



a) Esquema de la trama de datos que recibe el robot.

b) Esquema de trama de datos que retorna el robot.

Fig. 6.18 Tramas de datos para la comunicación.

Elaborado por: Santiago Álvarez

De igual manera y para mejor desempeño y desarrollo de mas aplicaciones con el robot, se implementa una comunicación directa con una PC sin la necesita de tener módulos inalámbricos, ya que mediante la gran potencia de eficiencia en enviar y recepción de datos se conecta directamente el computador a la placa de control del robot.

6.8.2. Cinemática del Robot Omnidireccional

Dentro del as consideraciones que se debe tomar en cuenta para obtener el modelo cinemático del robot omnidireccional en el plano X-Y son:

- No existe ninguna parte de la estructura del robot que sea flexible.
- El robot se desplazará sobre una superficie plana.
- Las ruedas poseen un eje de direccionamiento que es perpendicular al suelo.

Una plataforma omnidireccional con 4 ruedas omnidireccionales se considera como un cuerpo rígido ubicado dentro de un marco referencial fijo sobre el cual se desplazara con referencia en el plano $R(X,Y,Z)$ donde Z que es eje perpendicular a este plano.

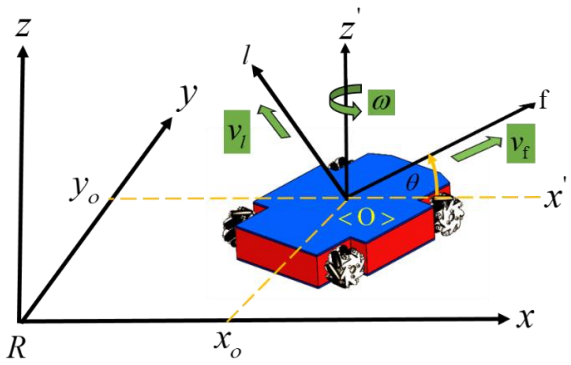


Fig. 6.19 Representación cinemática del robot omnidireccional.

Elaborado por: Santiago Álvarez

En la Fig. 6.19 se muestra como es la configuración del robot omnidireccional representado por sus respectivas velocidades, las cuáles se considera dos velocidades lineales v_f y v_l además una velocidad angular ω.

Cada velocidad lineal se dirige como eje del marco referencial unido al centro de gravedad del robot omnidireccional con dirección frontal y lateral, su velocidad angular rodea al eje Z de una manera anti horaria en forma positiva.

Para determinar la posición y velocidad del robot en cualquier instante de tiempo y la orientación se considera todos los vectores auxiliares mostrados en la Fig. 6.20.

- v_f velocidad lineal frontal en el eje X
- v_l velocidad lateral en el eje Y
- ω velocidad angular alrededor del eje Z

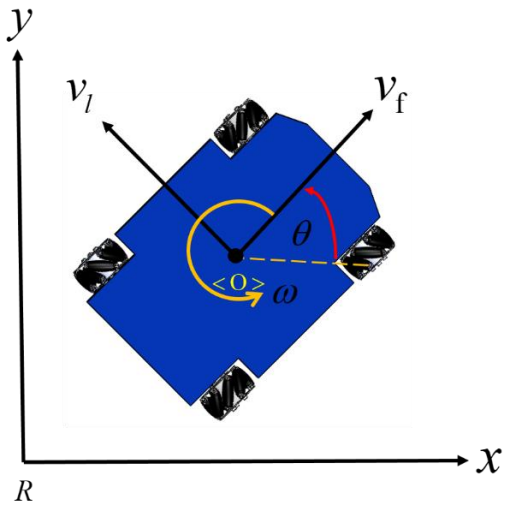


Fig. 6.20 Vectores representativos de las velocidades del robot..

Elaborado por: Santiago Álvarez

Las ecuaciones del modelo del robot vienen dadas por velocidades en el punto de operación del robot, en este caso en el centro, en función de las velocidades de maniobrabilidad de la plataforma robótica.

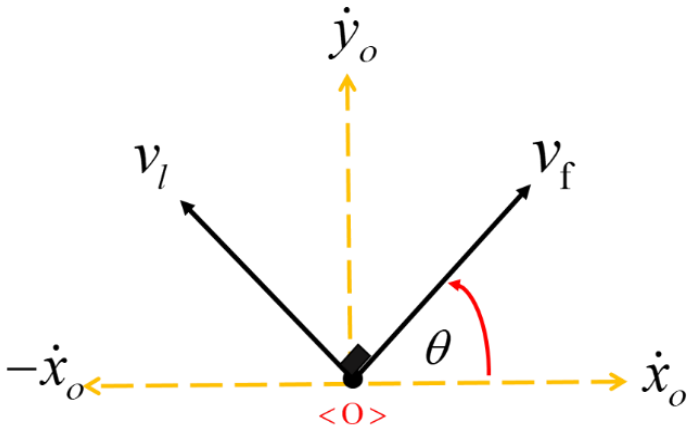


Fig. 6.21 Velocidades lineales del robot en el plano.

Elaborado por: Santiago Álvarez

Mediante la Fig. 6.21 se puede identificar la descomposición de las velocidades del robot, entonces la velocidad en el punto de operación $< O >$ del robot del eje X está dada por:

$$\dot{x}_o = v_f \cos\theta - v_l \cos(90-\theta) \tag{6.3}$$

Se sabe que por identidades trigonométricas la definición de:

$$\cos(\frac{\pi}{2}-\alpha) = \sin\alpha \tag{6.4}$$

empleando (6.4) en (6.3) se define que:

$$\dot{x}_o = v_f \cos\theta - v_l \sin\theta \tag{6.5}$$

La velocidad en el punto de operación $< O >$ del robot del eje Y está dada por:

$$\dot{y}_o = v_f \sin\theta + v_l \sin(90-\theta) \tag{6.6}$$

Por identidad trigonométrica se sabe que:

$$\sin(\frac{\pi}{2}-\alpha) = \cos\alpha \tag{6.7}$$

entonces (6.6) se define de la siguiente manera:

$$\dot{y}_o = v_f \sin\theta + v_l \cos\theta \tag{6.8}$$

La velocidad angular que representa al robot en el punto de operación se define como:

$$\omega = \dot{\theta} \tag{6.9}$$

prácticamente es la derivada del ángulo de rotación del robot.

Entonces el modelo del robot omnidireccional tomando como referencia el centro del robot se define por las ecuaciones (6.5), (6.8) y (6.9) como:

$$\begin{aligned} \dot{x}_o &= v_f \cos\theta - v_l \sin\theta \\ \dot{y}_o &= v_f \sin\theta + v_l \cos\theta \\ \dot{\theta} &= \omega \end{aligned} \tag{6.10}$$

Expresado de forma matricial el modelo cinemático se denota por:

$$\begin{bmatrix} \dot{x}_o \\ \dot{y}_o \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_f \\ v_l \\ \omega \end{bmatrix} \tag{6.11}$$

6.8.3. Modelo cinemático desplazado el punto de operación

El modelo cinemático desplazado el punto de operación representa el punto de control en donde se va a localizar el sensor laser, es decir, para cada instante de tiempo en que el robot se desplace este punto será el que va a seguir la trayectoria definida por el usuario. Para obtener el modelo cinemático del nuevo punto de control se comienza analizando las posiciones en cada eje para el robot con respecto al nuevo sistema de coordenadas operacionales.

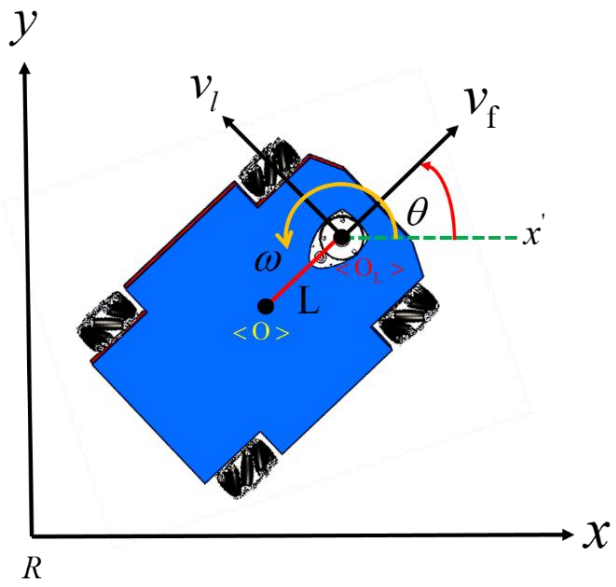


Fig. 6.22 Esquema cinemático con punto de operación desplazado.

Elaborado por: Santiago Álvarez

Como se observa en la Fig. 6.22 la distancia que se desplaza el sensor está representada por L , por lo cual se obtiene el nuevo punto de control operacional definido por $<\mathrm{O_L}>$ el cual cuenta con un nuevo vector de velocidades operacionales $\dot{\boldsymbol{\delta}} = [\dot{\delta}_x \quad \dot{\delta}_y \quad \dot{\delta}_\omega]$ las cuales se determina de la siguiente manera.

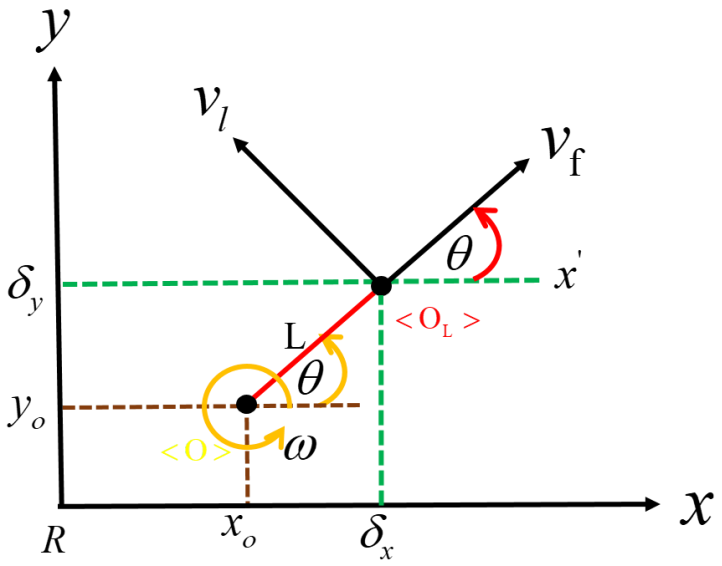


Fig. 6.23 Representación en el plano de la cinemática con el punto desplazado.

Elaborado por: Santiago Álvarez

Mediante la Fig. 6.23 se puede realizar el respectivo análisis por posición en los ejes X, Y se define de la siguiente manera:

$$\delta_x = x_o + L\cos\theta$$
$$\delta_y = y_o + L\sin\theta \tag{6.11}$$

Para obtener las velocidades de este nuevo punto de operación del robot ubicado donde se va a colocar el sensor Laser, se aplica la derivada parcial a (6.11) y (6.12) de la siguiente forma.

$$\frac{\partial \delta_x}{\partial t} = \frac{\partial x_o}{\partial t} + L\frac{\partial \cos\theta}{\partial t} \tag{6.12}$$

$$\dot{\delta}_x = \dot{x}_o - L\dot{\theta}\sin\theta \tag{6.13}$$

Se sabe que $\dot{\theta} = \omega$ de la ecuación (6.9) entonces reemplazando en (6.13) se obtiene la velocidad en el punto de control para el eje X como:

$$\dot{\delta}_x = \dot{x}_o - L\omega\sin\theta \tag{6.14}$$

Entonces, ahora reemplazando (6.5) en (6.14) se obtiene:

$$\dot{\delta}_x = v_f\cos\theta - v_l\sin\theta - L\omega\sin\theta \tag{6.15}$$

que representa la velocidad del punto de operación del robot ubicado en el sensor laser.

Ahora se aplica la derivada parcial para (6.11) del eje Y .

$$\frac{\partial \delta_y}{\partial t} = \frac{\partial y_o}{\partial t} + L\frac{\partial \sin\theta}{\partial t} \tag{6.16}$$

$$\dot{\delta}_y = \dot{y}_o + L\dot{\theta}\cos\theta \tag{6.17}$$

De igual forma se tiene que $\dot{\theta} = \omega$ de (6.9)

$$\dot{\delta}_y = \dot{y}_o + L\omega\cos\theta \tag{6.18}$$

reemplazando (6.8) en (6.18) se obtiene la velocidad de control en el punto del sensor laser para el eje Y :

$$\dot{\delta}_y = v_f\sin\theta + v_l\cos\theta + L\omega\cos\theta \tag{6.19}$$

Usando (6.15), (6.19) y (6.9) se obtiene el modelo cinemático del robot omnidireccional con el punto de operación desplazado ubicado a una distancia L del centro del robot, lugar en donde se ubicará el sensor laser:

$$\begin{aligned}\dot{\delta}_x &= v_f \cos\theta - v_l \sin\theta - L\omega \sin\theta \\ \dot{\delta}_y &= v_f \sin\theta + v_l \cos\theta + L\omega \cos\theta \\ \omega &= \dot{\theta}\end{aligned} \tag{6.20}$$

Expresando de forma matricial modelo del robot omnidireccional queda definido como:

$$\begin{bmatrix}\dot{\delta}_x \\ \dot{\delta}_y \\ \dot{\delta}_\omega\end{bmatrix} = \begin{bmatrix}\cos\theta & -\sin\theta & -L\sin\theta \\ \sin\theta & \cos\theta & L\cos\theta \\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}v_f \\ v_l \\ \omega\end{bmatrix} \tag{6.21}$$

El modelo cinemático del robot omnidireccional de una manera simplificada se expresa como:

$$\dot{\boldsymbol{\delta}}(t) = \mathbf{M}(\theta)\,\mathbf{v}(t) \tag{6.22}$$

Donde $\mathbf{M} \in \Re^{3x3}$ es la matriz de rotación homogénea del robot omnidireccional, $\mathbf{v} \in \Re^3$ es el vector de velocidades de maniobrabilidad del robot y $\dot{\boldsymbol{\delta}} \in \Re^3$ es el vector de velocidades del punto de interés del robot.

6.8.4. Velocidades de maniobrabilidad de un Robot Omnidireccional

Las velocidades de maniobrabilidad del robot omnidireccional corresponden a las velocidades angulares que se debe aplicar a cada rueda para cumplir con la velocidad lineal que se desea que el robot se mueva.

6.8.4.1. Odometría Diferencial del robot omnidireccional

Para estimar la posición del robot omnidireccional se emplea la odometría por medio de la cinemática diferencial, usando los encoders que están colocados en cada motor del robot. El control de navegación como resultado entrega velocidades lineales las cuales se deben aplicar al robot para seguir la trayectoria planeada, entonces se necesita tener una retroalimentación del robot en tanto a posición y orientación para saber en qué punto del espacio x-y se encuentra.

A. Calculo de las velocidades angulares

Como se mencionó anteriormente los encoders son sensores que captan las revoluciones de un eje de cada motor, son empleados para estimar la posición de un robot móvil. Se tomará en cuenta que un encoder da cuantas vueltas ha dado un eje dentro de un intervalo de tiempo. En un robot omnidireccional los encoders se colocan en los ejes de cada rueda, es decir, en cada uno de los grados de libertad de movimiento que pueda tener cada una de las ruedas. En esta sección se analizará la configuración concreta de las ruedas del robot omnidireccional. La configuración de ruedas y de los motores de un robot omnidireccional se la denomina cinemática del robot. Se analizará el método más sencillo y confiable denominada cinemática diferencial. Un robot omnidireccional de cinemática es aquel que cuenta con 4 ruedas motrices unidas por un eje, cada una de estas ruedas cuentan con su propio motor que la hace girar en el plano perpendicularmente.

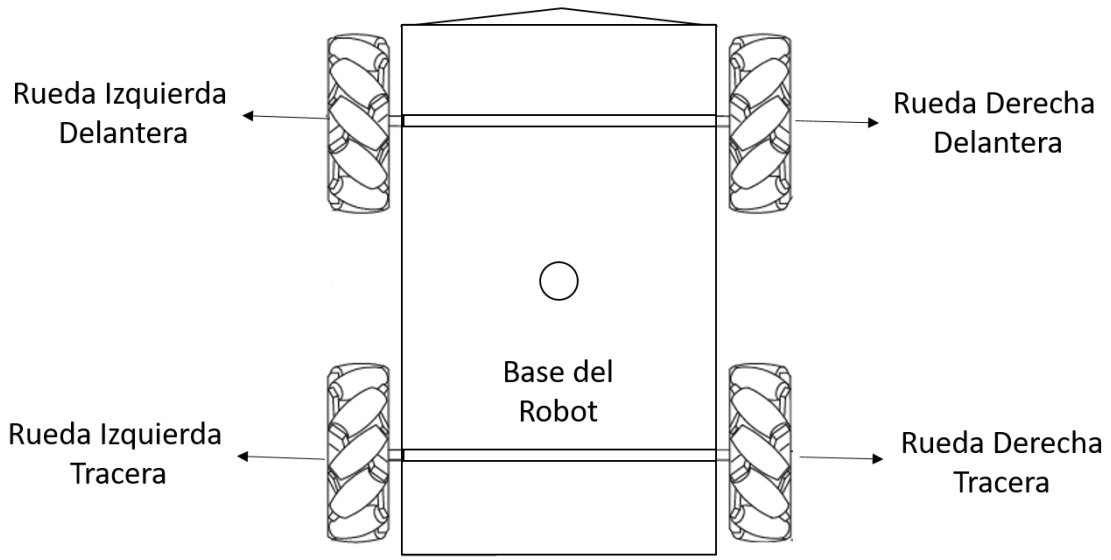


Fig. 6.24 Estructura del robot omnidireccional para el análisis de la odometría.

Elaborado por: Santiago Álvarez

Como se observa en la Fig.6.24 se observa que el robot omnidireccional cuenta con cuatro ruedas de tipo Mecanum, cada una de ellas enlazadas por un eje central, mediante estas ruedas se lograra el desplazamiento del robot móvil. Para determinar las velocidades del robot omnidireccional y lograr estimar la posición con la cinemática se hace mediante el siguiente análisis de acuerdo a la configuración del robot omnidireccional se determina matemáticamente las ecuaciones de transformación de velocidades angulares a lineales y de lineales a angulares. La Fig. 6.24. muestra cómo están distribuidas las velocidades

angulares $\omega_1, \omega_2, \omega_3,$ y ω_4 que permite al robot desplazase en cualquier dirección con ayuda de sus ruedas omnidireccionales de radio $_r$.

Para el cálculo de la velocidad angular de cada rueda se lo hace mediante la siguiente formula:

$$\omega = \frac{2\pi P_s}{P_{rev}} \left[\frac{rad}{s} \right]$$

Donde:

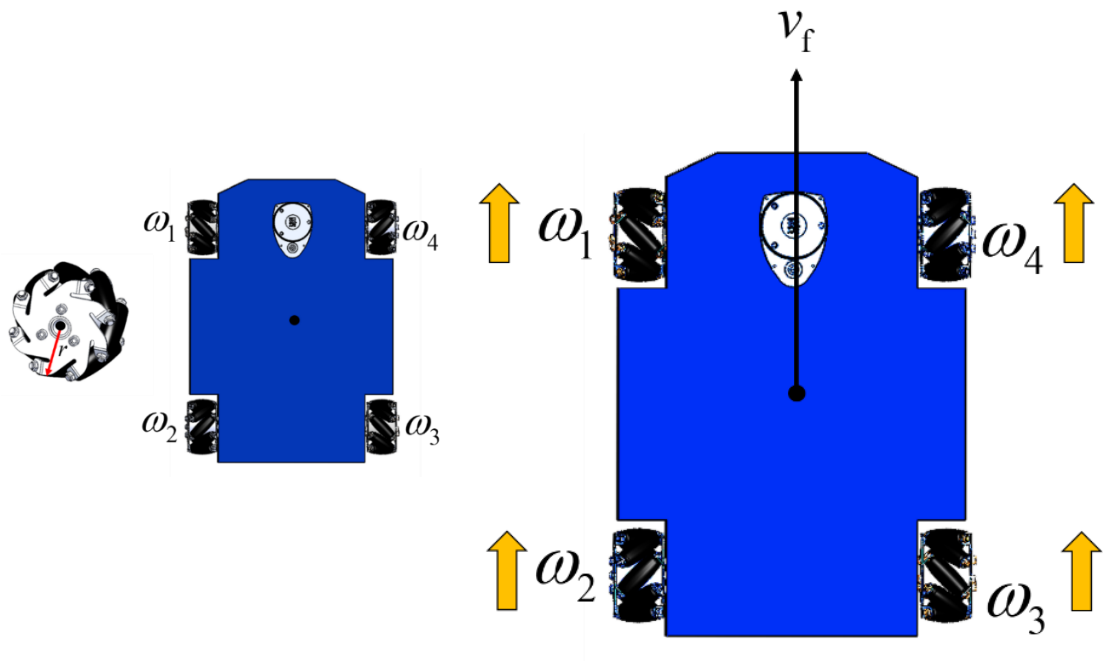
P_s = son los pulsos que da el encoder dentro de un intervalo de tiempo, está dentro de las unidades de $\frac{\#\ pulsos}{segundo}$.

P_{rev} = el número de pulsos que da en una revolución completa, viene dado por $\#\ pulsos$

Entonces la velocidad angular está dado por $\left[\frac{rad}{s} \right]$, para determinar cómo se desplaza el robot en base a cada velocidad angular de cada rueda se hace el siguiente análisis únicamente multiplicando cada velocidad angular por el radio de la rueda, entones se tiene el desplazamiento lineal que da cada rueda, empleando la cinemática diferencial se calcula el desplazamiento total del robot en el plano x-y

B. Análisis de las velocidades para el desplazamiento del robot

Para definir la velocidad lineal frontal del robot v_f el robot debe cumplir con los siguientes requerimiento de velocidades angulares Fig. 6.25.



a) Robot Omnidireccional

b) Velocidades angulares para desplazar hacia el frente el robot

Fig. 6.25 Esquema de configuración de velocidad lineal para v_f

Elaborado por: Santiago Álvarez

Entonces para obtener un movimiento lineal del robot con direcciona frontal todas las velocidades angulares deben ser positivas, es decir, que cada rueda gire en el mismo sentido para desplazar al robot hacia delante.

La configuración para lograr el movimiento frontal de robot se define como:

$$v_f(t) = \frac{\omega_1 + \omega_2 + \omega_3 + \omega_4}{4} r \tag{6.23}$$

Donde ω_1, ω_2, ω_3 y ω_4 representan las velocidades angulares de cada rueda que conforma el robot omnidireccional expresadas en $\left[rad/s \right]$ y r representa el radio de la rueda Mecanum expresado en $[m]$, entonces de acuerdo con la ecuación (6.23) el valor de v_f esta dado en $\left[m/s \right]$.

Ahora para definir la configuración de las velocidades angulares para que el robot se desplace en dirección lateral v_f tienen que estar definidas como se muestra en la Fig. 6.26.

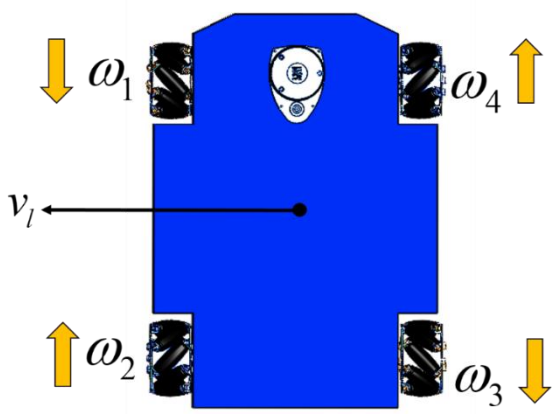


Fig. 6.26 Esquema de configuración de velocidad lineal para v_l

Elaborado por: Santiago Álvarez

La ecuación con la que se determina el valor de la velocidad lateral lineal v_l está definida de la siguiente manera:

$$v_l(t) = \frac{-\omega_1 + \omega_2 - \omega_3 + \omega_4}{4} r \tag{6.24}$$

Finalmente, para obtener la velocidad de rotación del robot omnidireccional con ruedas Mecanum en configuración “AB” se debe establecer las velocidades angulares como se muestra en la Fig. 6.27.

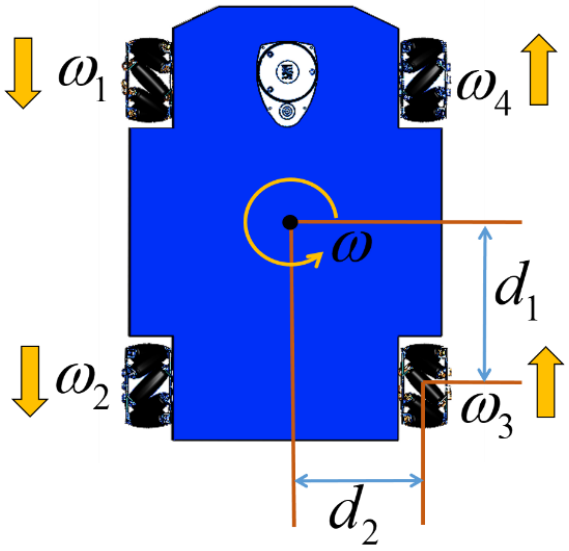


Fig. 6.27 Esquema de configuración de velocidad angulares para ω

Elaborado por: Santiago Álvarez

Como se puede apreciar las velocidades angulares de cada rueda debe tener esa configuración para obtener una velocidad angular del robot omnidireccional en sentido anti horario, la ecuación de transformación de velocidades se define como:

$$\omega(t) = \frac{-\omega_1 - \omega_2 + \omega_3 + \omega_4}{4(d_1 + d_2)} r \tag{6.25}$$

donde d_1 y d_2 son las distancias sobre el cual robot rotara y vienen dadas en $[m]$ entonces como el radio también está dado en $[m]$ el resultado de la ecuación será una velocidad angular dada en $\left[rad/s \right]$.

El sistema de ecuaciones para transformar las velocidades angulares medidas por el robot se puede expresar de forma matricial para obtener una mejor apreciación del sistema, descrita como:

$$\begin{bmatrix} v_f \\ v_l \\ \omega \end{bmatrix} = \begin{bmatrix} \frac{r}{4} & \frac{r}{4} & \frac{r}{4} & \frac{r}{4} \\ -\frac{r}{4} & \frac{r}{4} & -\frac{r}{4} & \frac{r}{4} \\ -\frac{r}{4(d_1+d_2)} & -\frac{r}{4(d_1+d_2)} & \frac{r}{4(d_1+d_2)} & \frac{r}{4(d_1+d_2)} \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix} \tag{6.26}$$

Sacando factor común $r/4$ de (6.26), se define el sistema de ecuaciones para transformar las velocidades angulares en velocidades líneas del robot en una forma matricial.

$$\begin{bmatrix} v_f \\ v_l \\ \omega \end{bmatrix} = \frac{r}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ -1 & 1 & -1 & 1 \\ -\frac{1}{d_1+d_2} & -\frac{1}{d_1+d_2} & -\frac{1}{d_1+d_2} & -\frac{1}{d_1+d_2} \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix} \tag{6.27}$$

El sistema (6.27) se puede describir de igual manera en forma de ecuación en producto de matrices, de tal manera que se defina de la siguiente forma:

$$\mathbf{v}(t) = \mathbf{T}\boldsymbol{\omega}(t) \tag{6.28}$$

Donde $\mathbf{v}(t) = \begin{bmatrix} v_f & v_l & \omega \end{bmatrix}^{\mathrm{T}}$ es el vector de velocidades lineales que se van a medir del robot, $\mathbf{T} \in \Re^{3x4}$ es una matriz de transformación y el vector $\boldsymbol{\omega}(t) = \begin{bmatrix} \omega_1 & \omega_2 & \omega_3 & \omega_4 \end{bmatrix}^{\mathrm{T}}$

conformado por las velocidades angulares que envía el robot hacia el software matemático para la retroalimentación del control.

Para lograr inyectar las velocidades angulares determinadas en base a las velocidades lineales que el controlador entrega se debe aplicar un simple despeje la ecuación (6.28), se debe tener en cuenta como es un sistema matricial se lo hace mediante la inversa de la matriz. Al no ser $\mathbf{T}$ una matriz cuadrada se aplica la pseudo-inversa, mediante este método se calcula las velocidades angulares correspondientes para que el robot omnidireccional se desplace según las velocidades que entrega el controlador implementado, esta expresión se define de la siguiente manera:

$$\boldsymbol{\omega}_{ref}(t) = \mathbf{T}^{-1}\mathbf{v}_{ref}(t) \tag{6.29}$$

Donde $\boldsymbol{\omega}_{ref} \in \Re^4$ es el vector que contendrá las velocidades de referencia que se aplicaran al robot omnidireccional para que cumpla la terea propuesta y $\mathbf{v}_{ref} \in \Re^3$ es el vector que contiene las velocidades lineales de referencia que guiaran al robot a cumplir su objetivo. Entonces aplicando la pseudo-inversa en MATLAB de la matriz $\mathbf{T}$ se obtiene la siguiente ecuación.

$$\begin{bmatrix} \omega_{ref1} \\ \omega_{ref2} \\ \omega_{ref3} \\ \omega_{ref4} \end{bmatrix} = \begin{bmatrix} \frac{1}{r} & -\frac{1}{r} & -\frac{d_1+d_2}{r} \\ \frac{1}{r} & \frac{1}{r} & -\frac{d_1+d_2}{r} \\ \frac{1}{r} & -\frac{1}{r} & \frac{d_1+d_2}{r} \\ \frac{1}{r} & \frac{1}{r} & \frac{d_1+d_2}{r} \end{bmatrix} \begin{bmatrix} v_{fref} \\ v_{lref} \\ \omega_{ref} \end{bmatrix} \tag{6.30}$$

Sacando factor como el término $\frac{1}{r}$ se obtiene el sistema de la siguiente manera:

$$\begin{bmatrix} \omega_{ref1} \\ \omega_{ref2} \\ \omega_{ref3} \\ \omega_{ref4} \end{bmatrix} = \frac{1}{r}\begin{bmatrix} 1 & -1 & -(d_1+d_2) \\ 1 & 1 & -(d_1+d_2) \\ 1 & -1 & d_1+d_2 \\ 1 & 1 & d_1+d_2 \end{bmatrix} \begin{bmatrix} v_{fref} \\ v_{lref} \\ \omega_{ref} \end{bmatrix} \tag{6.31}$$

Con el cálculo de cada velocidad del robot se hace uso de evaluar esta velocidad cada instante tiempo logrando así obtener la posición del robot, acumulando siempre el valor anterior más el actual. A esto se lo llama la integral de Euler, se tiene posición en x, posición en y, conjuntamente la rotación en que se encuentra el robot.

C. Eficiencia de control PID interno.

Para validar el control PID interno del robot omnidireccional se realiza una prueba en la que consiste enviar velocidades variantes en el tiempo, con esta prueba se pretende determinar la eficiencia del control interno. La prueba consiste en inyectar las siguientes velocidades lineales definidas como: $v_{f\ \mathrm{ref}} = \frac{2}{5}\sin\left(\frac{4}{5}t\right)\sin\left(\frac{1}{25}t^2\right) - \frac{3}{5}\cos\left(\frac{t}{2}\right)$, $v_{l\ \mathrm{ref}} = -\frac{3}{5}\cos\left(\frac{3}{4}t\right)\cos\left(\frac{1}{4}t\right)$ y $\omega_{\mathrm{ref}} = \frac{17}{20}\sin\left(\frac{7}{20}t\right)\cos\left(\frac{1}{5}t\right)$ durante un periodo de tiempo de 60 [s]. En la Fig. 6.28. presenta la velocidad frontal de referencia inyectada al robot $v_{f\ \mathrm{ref}}$ y la velocidad real del robot ejecutada v_f, de igual manera en la Fig. 6.29 se presenta las velocidades laterales del robot $v_{l\ \mathrm{ref}}$ y v_l respectivamente y por último en la Fig. 6.30 se indica la velocidad angular de referencia ω_{ref} y la velocidad angular del robot real ω. En el **Anexo 2** se tiene el código para evaluar el controlador interno del robot.

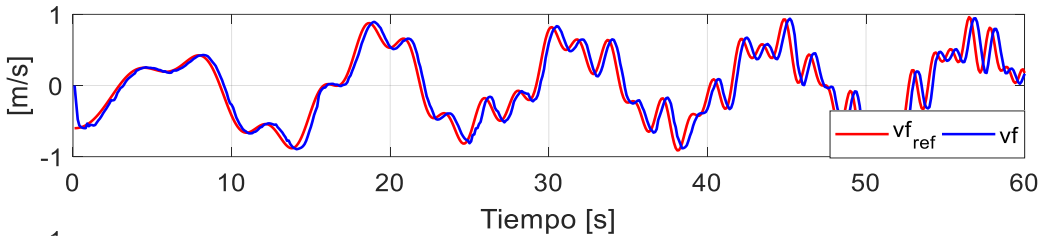


Fig. 6.28 Comportamiento de la velocidad frontal del robot ($v_{f\ \mathrm{ref}}$ vs. v_f)

Elaborado por: Santiago Álvarez

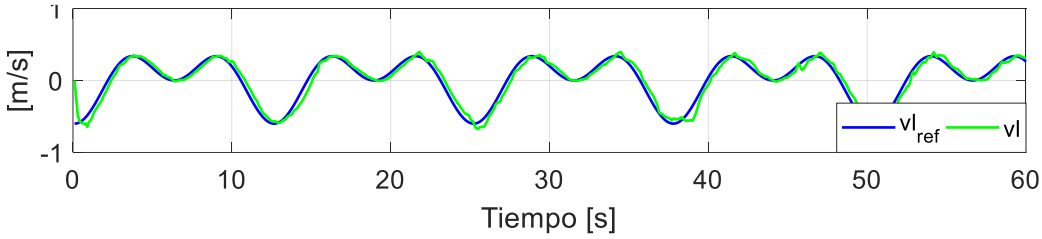


Fig. 6.29 Comportamiento de la velocidad lateral del robot ($v_{l\ \mathrm{ref}}$ vs. v_l)

Elaborado por: Santiago Álvarez

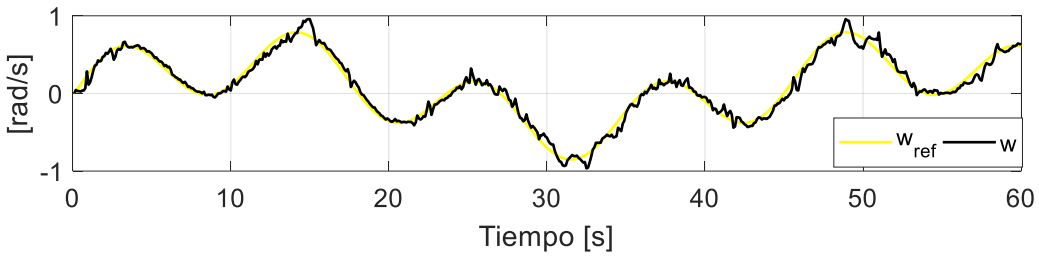


Fig. 6.30 Comportamiento de la velocidad angular del robot (ω_{ref} vs. ω)

Elaborado por: Santiago Álvarez

Para mostrar la eficiencia del control se hace énfasis al error en cada instante de tiempo, es decir, se grafica la diferencia entre los valores deseados y los valores que genera el robot. En la Fig. 6.31 se indica una gráfica de los errores producidos entre los valores de referencia y valores reales del robot.

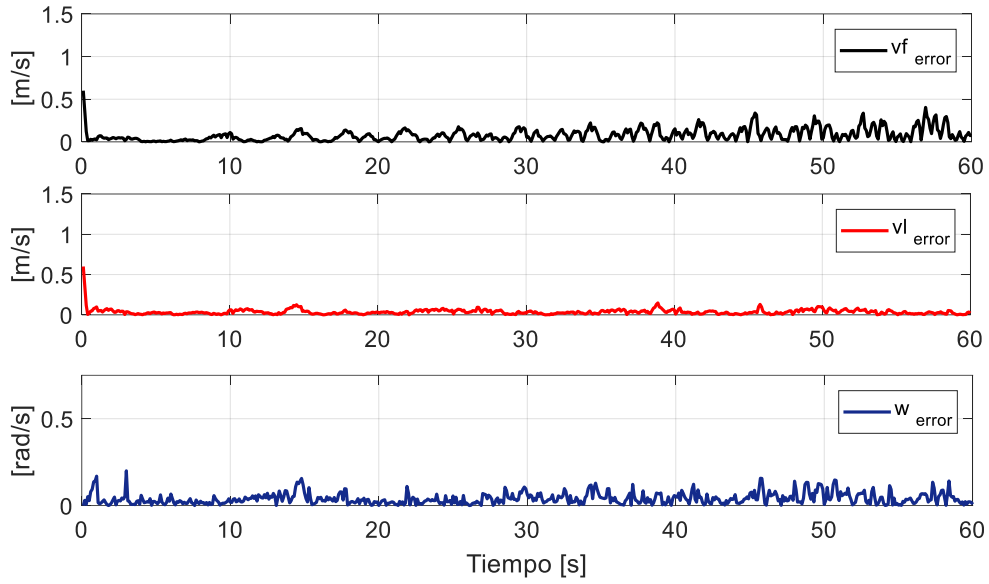


Fig. 6.31 Errores presentes entre las velocidades de referencia y las velocidades reales del robot.

Elaborado por: Santiago Álvarez

Mediante la gráfica anterior se observa que el controlador interno del robot funciona correctamente, siguiendo las velocidades de referencia inyectadas al robot de forma continua, e incluso como es un controlador que compensa perturbaciones, se tiene una eficiencia de controlador muy aceptable para el funcionamiento adecuado del robot. Los errores promedio presentados son de: $v_{f\ \text{error}} = 0.08123\ [\text{m/s}]$, $v_{\text{l error}} = 0.03644\ [\text{m/s}]$ y $\omega_{\text{error}} = 0.04035\ [\text{rad/s}]$, teniendo en cuanta que el tiempo de muestreo de las velocidades es de 100ms,espor eso que se nota el desfase de las velocidades.

D. Simulación del modelo cinemático y prueba de la odometría

Para validar el modelo cinemático (6.21) y la matriz de transformación (6.31) del robot omnidireccional se desarrolla un programa en Matlab con tiempo de simulación de 20 segundos, en el cual se escribe velocidades lineales y angular al modelo logrando ver como es la evolución del robot, para crear el programa se emplea condiciones iniciales de $x_o = 0$ [m], $y_o = 0$ [m] y $\theta_o = 0$ [rad]. Como entrada de velocidades se tiene $v_f = \begin{bmatrix} 0.65 & -0.55 \end{bmatrix}$ m/s, $v_l = \begin{bmatrix} -0.35 & -0.85 \end{bmatrix}$ m/s y $\omega = \begin{bmatrix} 0.75 & -0.8 \end{bmatrix}$ rad/s.

La Fig. 6.32 indica la evolución de la posición del robot conforme el tiempo de simulación se ejecuta, la Fig. 6.33 muestra la orientación del robot y se indica la evolución de las velocidades angulares a aplicar a cada rueda del robot en la Fig. 6.34 y por último en la Fig. 6.35 se observa el desplazamiento que realiza el robot. En el **Anexo 2** se presenta el código del programa validar de manera simulada el modelo cinemático del robot omnidireccional.

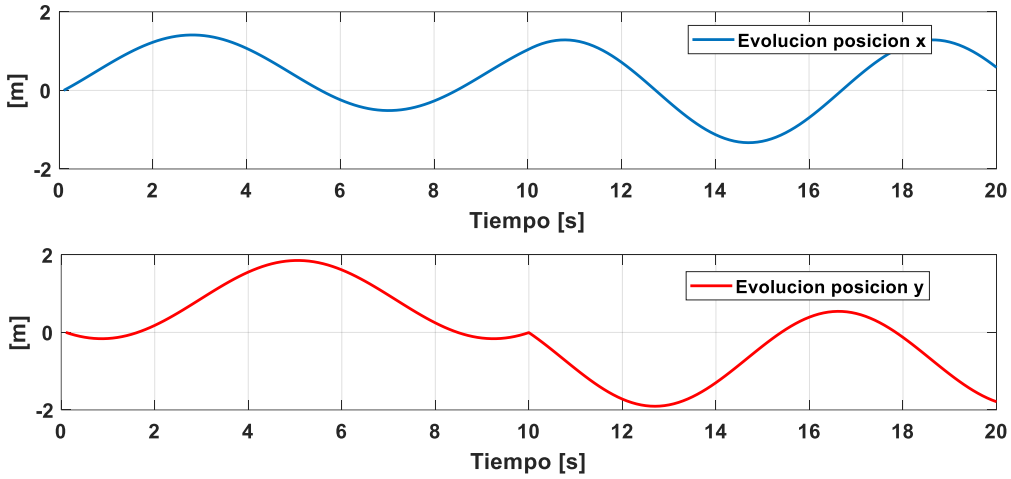


Fig. 6.32 Evolución de la posición de robot en x,y.

Elaborado por: Santiago Álvarez

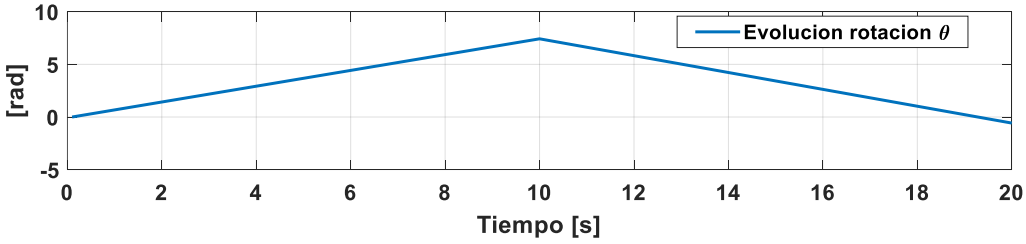


Fig. 6.33 Evolución del ángulo de orientación de robot.

Elaborado por: Santiago Álvarez

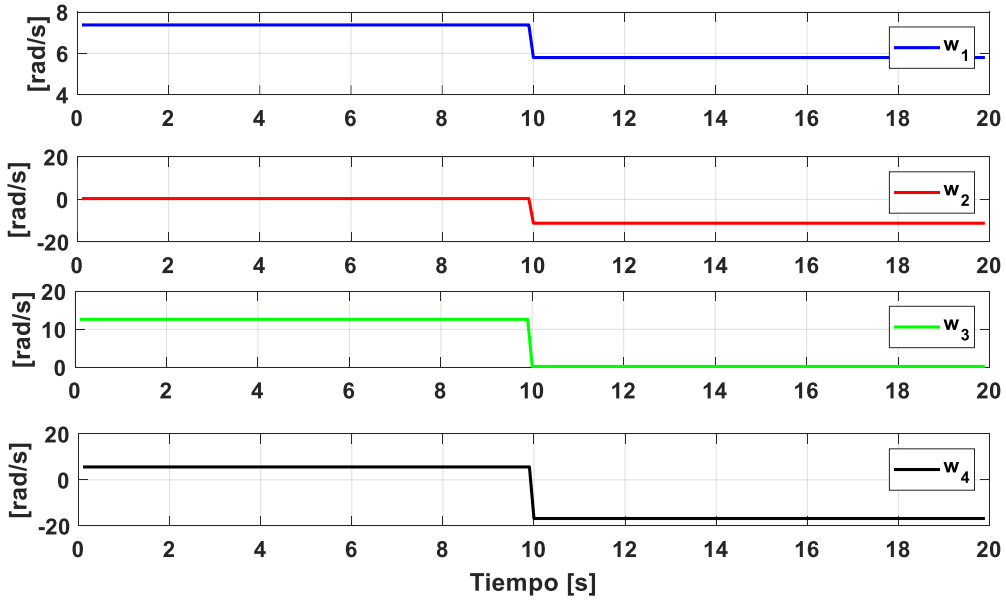


Fig. 6.34 Velocidades angulares para cada rueda del robot.

Elaborado por: Santiago Álvarez

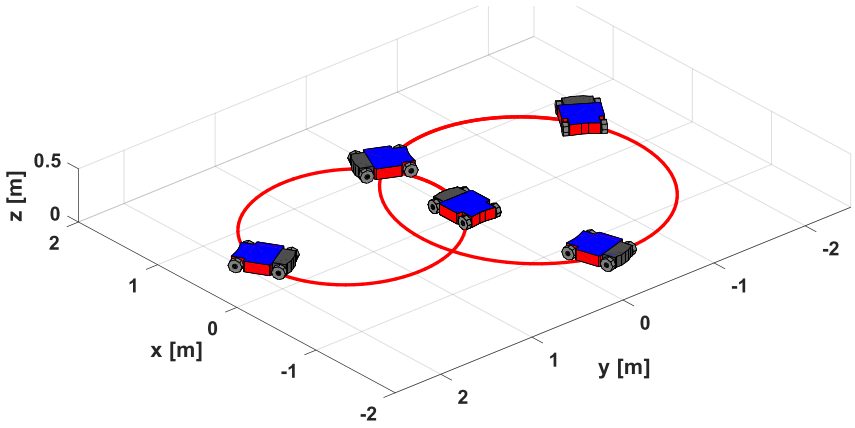


Fig. 6.35 Movimiento del robot simulado.

Elaborado por: Santiago Álvarez

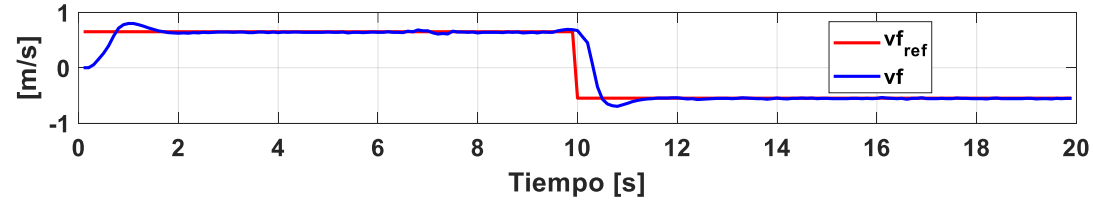
E. Prueba experimental del modelo cinemático

Para realizar esta prueba experimental, se hace uso del robot omnidireccional construido Fig. 6.36 de igual manera que la simulación se emplea las mismas condiciones descritas, en la Fig. 6.37 (a) se indica la velocidad frontal deseada y la velocidad frontal leída del robot, la Fig. 6.37 (b) indica la velocidad deseada lateral y la velocidad lateral leída del robot, además en la Fig. 6.37 (c) se muestra la velocidad angular deseada y leída del robot, en la Fig. 6.38 se indica la evolución de las velocidades angulares deseadas y las

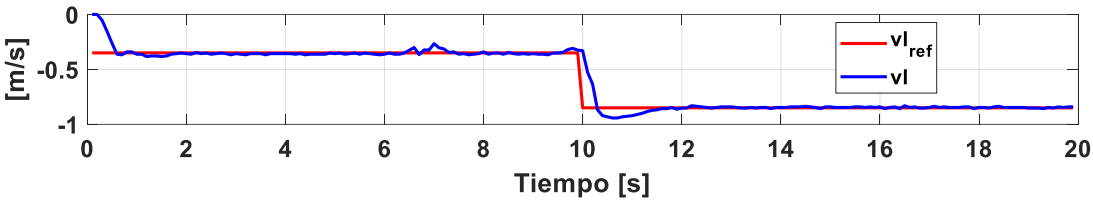
velocidades angulares correspondientes a cada rueda leídas del robot y en la Fig. 6.39 se indica la evolución de la posición del robot para X, Y y su respectiva rotación θ. Para indicar al movimiento del robot se puede observar la Fig. 6.40 que describe el trayecto recorrido por el robot al ingresar las velocidades lineales y angulares. En el **Anexo 3** se presenta el código del programa validar de forma experimental el modelo cinemático del robot omnidireccional.

Fig. 6.36 Robot omnidireccional construido.

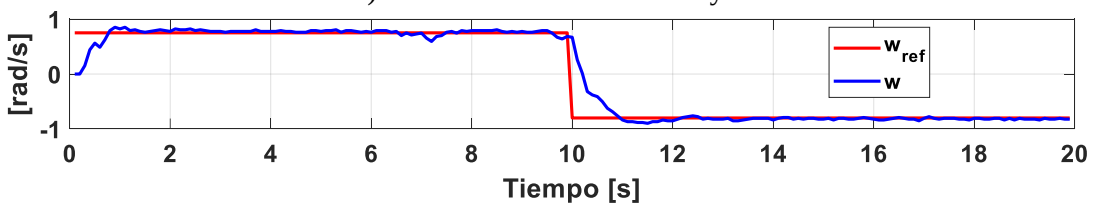
Elaborado por: Santiago Álvarez



a) Velocidad Frontal deseada y leída



b) Velocidad lateral deseada y leída



c) Velocidad angular deseada y leída.

Fig. 6.37 Velocidades de maniobrabilidad del robot escritas y leídas.

Elaborado por: Santiago Álvarez

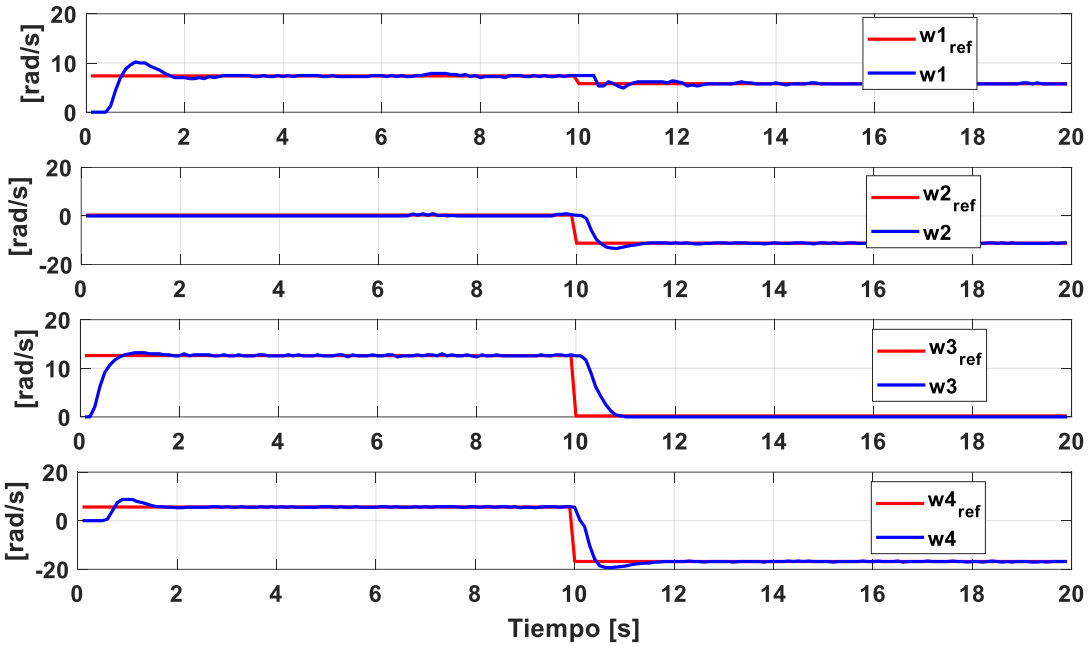
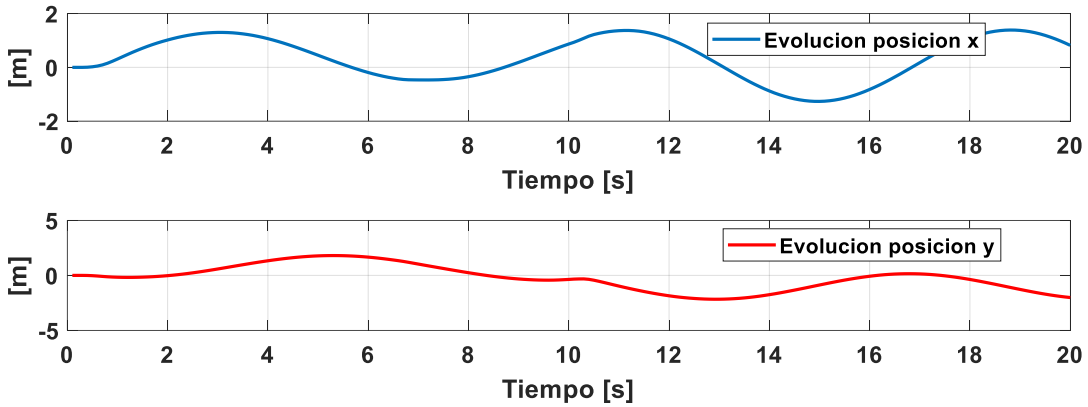
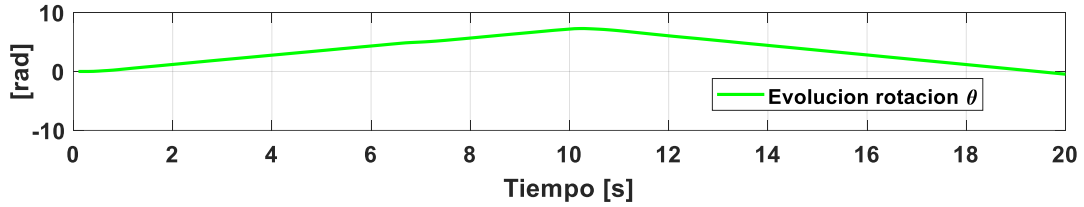


Fig. 6.38 Velocidades angulares de las ruedas del robot deseadas y leídas.

Elaborado por: Santiago Álvarez



a) Evolución de la posición del robot para X,Y



b) Evolución de la rotación del robot.

Fig. 6.39 Evolución de la posición y la orientación del robot según la cinemática.

Elaborado por: Santiago Álvarez

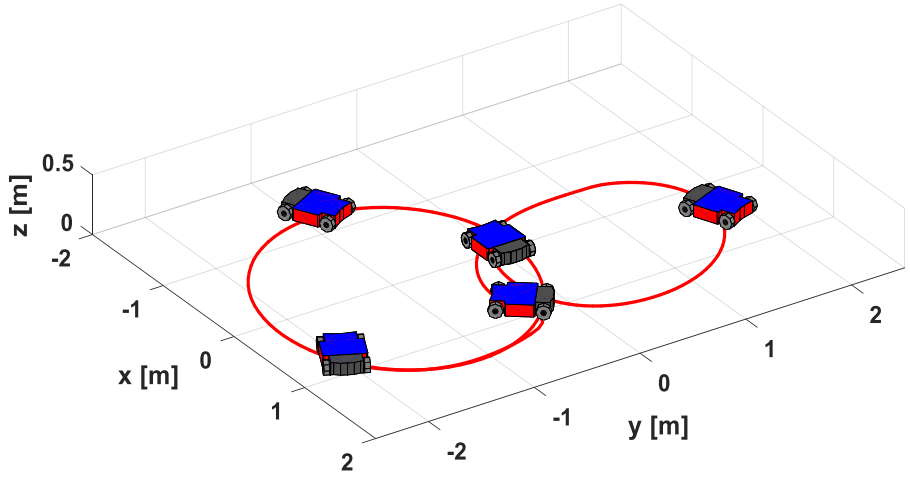


Fig. 6.40 Movimiento real descrito por el robot.

Elaborado por: Santiago Álvarez

6.8.5. Algoritmo de Navegación

Para lograr cumplir con la terea de navegación autónoma se plantea un control PID con el propósito de compensar errores durante el desplazamiento, el esquema de control se mostró en la Fig. 6.2, logrando así de esta forma que el robot se mueva de manera autónoma por una secuencia de posiciones (trayectoria deseada) $\boldsymbol{\delta}_d(t)$ predefinida. Se propone el siguiente controlador PID.

$$\mathbf{v_c}(t) = \mathbf{M}^{-1}_{(\theta)}\left(\dot{\boldsymbol{\delta}}_d + \mathbf{k}\tilde{\boldsymbol{\delta}}\right) \tag{6.32}$$

La representación (6.32) en MATLAB para ahorrar el cálculo matemático de utiliza una función (*linsolve*) que cuenta con solución de sistemas de ecuaciones lineales o no lineales quedando de forma:

$$\mathbf{v_c}(t) = linsolve\left(\mathbf{M}_{(\theta)}, \left[\dot{\boldsymbol{\delta}}_d + PID\left(\tilde{\boldsymbol{\delta}}\right)\right]\right) \tag{6.33}$$

De (6.32) $\mathbf{M}^{-1}_{(\theta)}$ representa la inversa de la matriz homogénea del modelo cinemático del robot omnidireccional, $\dot{\boldsymbol{\delta}}_d$ es el vector de parámetros de referencia de la trayectoria deseada. Si $\boldsymbol{\delta}_d = [\mathbf{x_d} \quad \mathbf{y_d} \quad \boldsymbol{\theta}_\mathbf{d}]$ entonces $\dot{\boldsymbol{\delta}}_d = [\dot{\mathbf{x}}_\mathbf{d} \quad \dot{\mathbf{y}}_\mathbf{d} \quad \dot{\boldsymbol{\theta}}_\mathbf{d}]$ y el error de control se define como:

$$\tilde{\boldsymbol{\delta}} = \left(\boldsymbol{\delta}_d - \boldsymbol{\delta}\right) \tag{6.34}$$

donde $\boldsymbol{\delta}$ es vector con datos que contiene posición y orientación actual del robot.

6.8.5.1. Análisis de Estabilidad

Para el análisis de estabilidad de tiene que el robot siga correctamente el perfil deseado, se asume un seguimiento de velocidad perfecto, de tal manera que:

$$\mathbf{v} \cong \mathbf{v_c} \tag{6.35}$$

Entonces reemplazamos en (6.32) la ecuación (6.22) obteniendo:

$$\dot{\boldsymbol{\delta}} = \mathbf{M}_{(\theta)}\mathbf{M}^{-1}{}_{(\theta)}\left(\dot{\boldsymbol{\delta}}_d + \mathbf{k}\tilde{\boldsymbol{\delta}}\right) \tag{6.36}$$

por propiedad de matrices se sabe que $\mathbf{M}_{(\theta)}\mathbf{M}^{-1}{}_{(\theta)} = \mathbf{I}$ es igual a una matriz identidad, entonces la expresión (6.36) se define como:

$$\dot{\boldsymbol{\delta}} = \left(\dot{\boldsymbol{\delta}}_d + \mathbf{k}\tilde{\boldsymbol{\delta}}\right) \tag{6.37}$$

Ahora aplicamos una deriva parcial a la ecuación (3.34), que presenta el error en cada instante de tiempo se obtiene:

$$\dot{\boldsymbol{\delta}} = \dot{\boldsymbol{\delta}}_\mathbf{d} - \dot{\tilde{\boldsymbol{\delta}}} \tag{6.38}$$

Sustituyendo en (6.38) en la ecuación (6.37) se tiene la siguiente expresión:

$$\dot{\boldsymbol{\delta}}_\mathbf{d} - \dot{\tilde{\boldsymbol{\delta}}} = \dot{\boldsymbol{\delta}}_d + \mathbf{k}\tilde{\boldsymbol{\delta}} \tag{6.39}$$

Operando la ecuación (6.39) se presenta que:

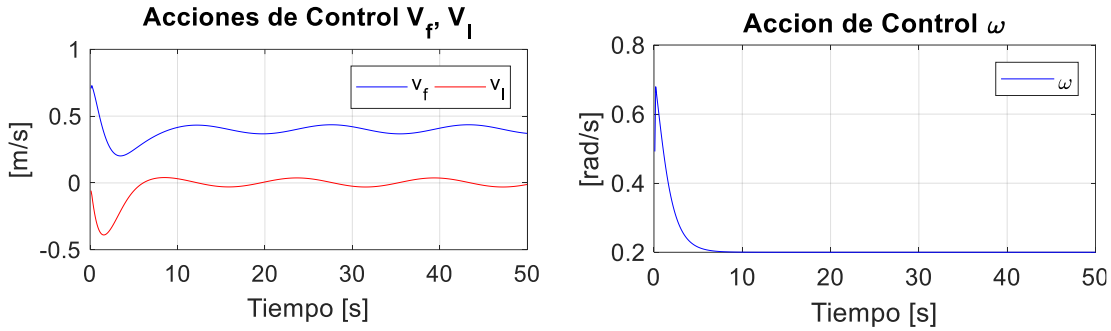
$$0 = \dot{\tilde{\boldsymbol{\delta}}} + \mathbf{k}\tilde{\boldsymbol{\delta}} \tag{6.40}$$

Siendo $\mathbf{k}$ un vector con ganancias positivas, la expresión (6.40) se cumplirá cuando $\tilde{\boldsymbol{\delta}} \to 0$, esto hace que el controlador PID sea asintóticamente estable.

A. Simulación de control autónomo (Trayectoria 1)

Para la simulación se desarrolla un programa en MATLAB que defina una trayectoria deseada para el robot conjuntamente con las referencias de velocidad y las condiciones iniciales la simulación contara con un tiempo de muestro de 100 [ms]. La trayectoria definida está dada por $\mathrm{x_d} = r * \cos\left(\frac{t}{5}\right)$ [m], $y_\mathrm{d} = r * \sin\left(\frac{t}{5}\right)$ [m] , donde r es el radio de la

circunferencia de la trayectoria y por último el ángulo deseado para cada instante de tiempo $\theta_d = \arctan\left(\frac{\dot{y}_d}{\dot{x}_d}\right)$. En Fig. 6.41 (a) se muestra la evolución de las velocidad lineal frontal y lateral aplicada al robot según el controlador entregue, en la Fig. 6.41 (b) se indica la velocidad angular suministrada al robot para corregir los errores de orientación, en la Fig. 6.42 (a) se muestra la evolución del error como tiende a cero según transcurra el tiempo, en la Fig. 6.42 (b) se presenta el error de orientación y por último la Fig. 6.43 muestra el movimiento realizado por el robot cumpliendo correctamente la trayectoria planteada. En el **Anexo 4** se presenta el código del programa para simular el controlador con varias trayectorias.

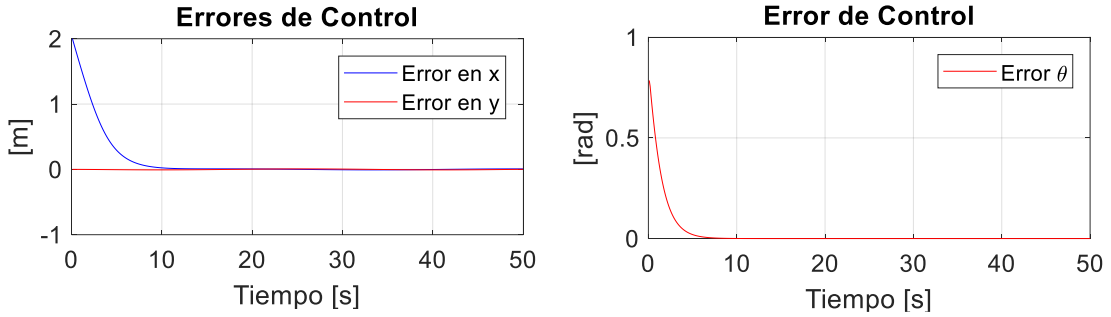


a) Evolución de Velocidades lineales v_f y v_l

b) Evolución de la velocidad angular del robot ω

Fig. 6.41 Velocidades lineales y velocidad angular de control.

Elaborado por: Santiago Álvarez



a) Error de control en x,y de la trayectoria simulada.

b) Error de control de rotación en la trayectoria simulada.

Fig. 6.42 Evolución de los errores de control en la trayectoria simulada

Elaborado por: Santiago Álvarez

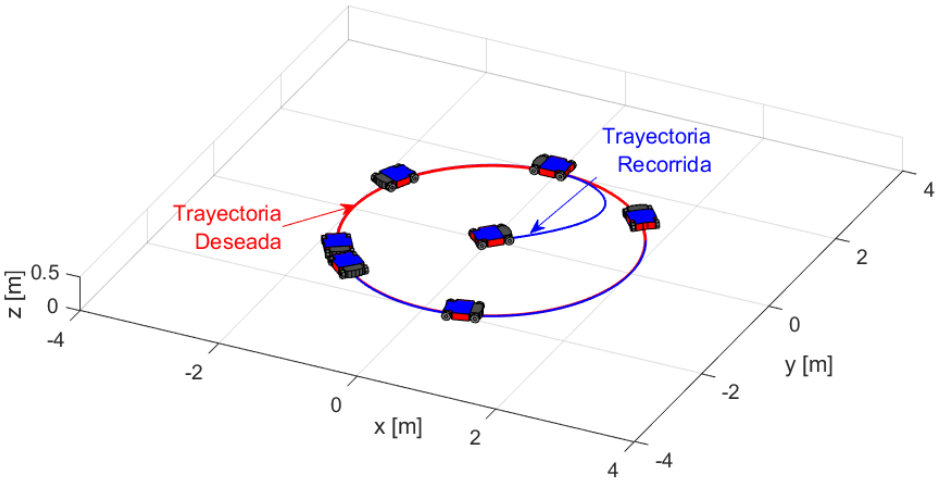


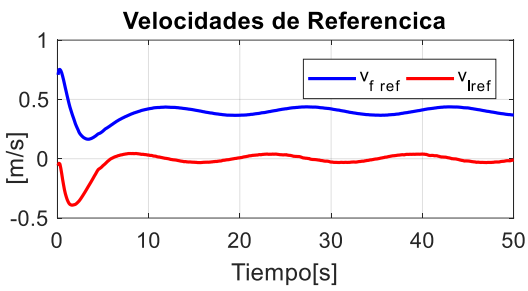
Fig. 6.43 Movimiento del robot ejecutado en la simulación.

Elaborado por: Santiago Álvarez

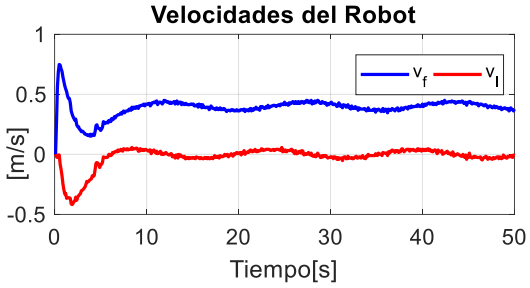
Como se puede observar el controlador mediante simulación de seguimiento de trayectoria predefinido cumple satisfactoriamente logrando establecer que $\dot{\tilde{\boldsymbol{\delta}}} \to 0$ asintóticamente.

B. Experimentación de control autónomo (Trayectoria 1)

Para verificar el funcionamiento del robot con su respectivo control se realiza de igual forma un experimento real con el prototipo, tomando en cuenta los mismos datos de simulación de la trayectoria 1 (circulo), para verificar que cumple de igual manera que la simulación. En Fig.6.44 (a) se puede observar la evolución de las velocidades lineales frontal y lateral de referencia aplicadas al robot, en la Fig. 6.44 (b) se indica las velocidades leídas del robot omnidireccional, las cuales se parecen a las velocidades inyectadas, es decir, el robot ejecuta las acciones que el controlador lo indique. En la Fig. 6.45 (a) se observa la velocidad angular de referencia que entrega el controlador para enviar al robot, de igual manera en la Fig. 6.45 (b) se observa la velocidad angular ejecutada por el robot. Los errores de control del experimento se pueden observar en la Fig. 6.46 (a) y Fig. 6.46 (b) respectivamente para posición y rotación. Por último, en la Fig. 6.47 se observa el movimiento realizado por el robot, logrando apreciar cómo sigue con la trayectoria planificada. En el **Anexo 5** se presenta el código del programa para ejecutar la experimentación del seguimiento de varias trayectorias.



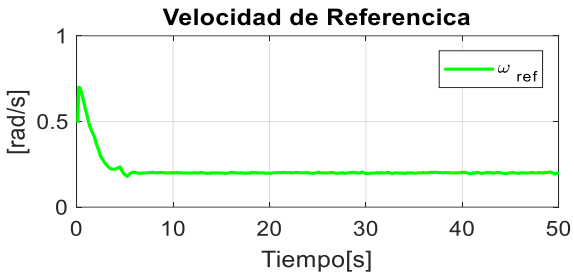
a) Evolución de velocidades de referencia $v_{f\ ref}$ y $v_{l\ ref}$



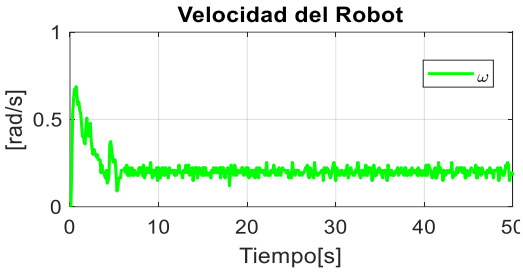
b) Evolución de las velocidades obtenidas del robot v_f y v_l

Fig. 6.44 Velocidades lineales generadas por el controlador y velocidades lineales reales del robot.

Elaborado por: Santiago Álvarez



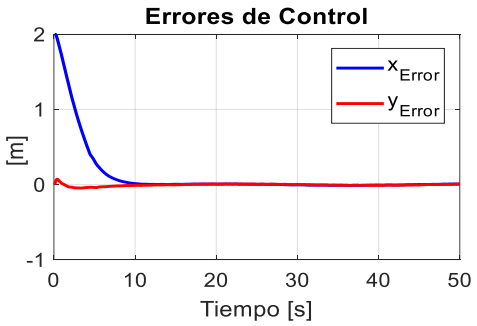
a) Evolución dela velocidad angular de referencia ω_{ref}.



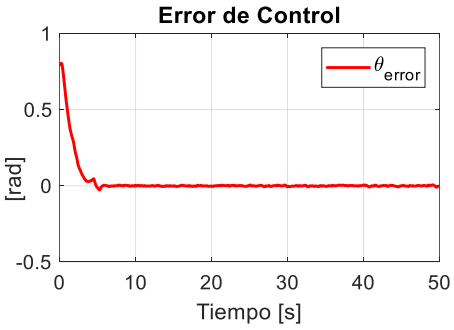
b) Evolución dela velocidad angular obtenida del robot ω.

Fig. 6.45 Velocidad angular generada por el controlador y velocidad angular real del robot.

Elaborado por: Santiago Álvarez



a) Error de control en x,y de la trayectoria ejecutada.



b) Error de control de rotación en la trayectoria ejecutada.

Fig. 6.46 Comportamiento de los errores de control del robot omnidireccional.

Elaborado por: Santiago Álvarez

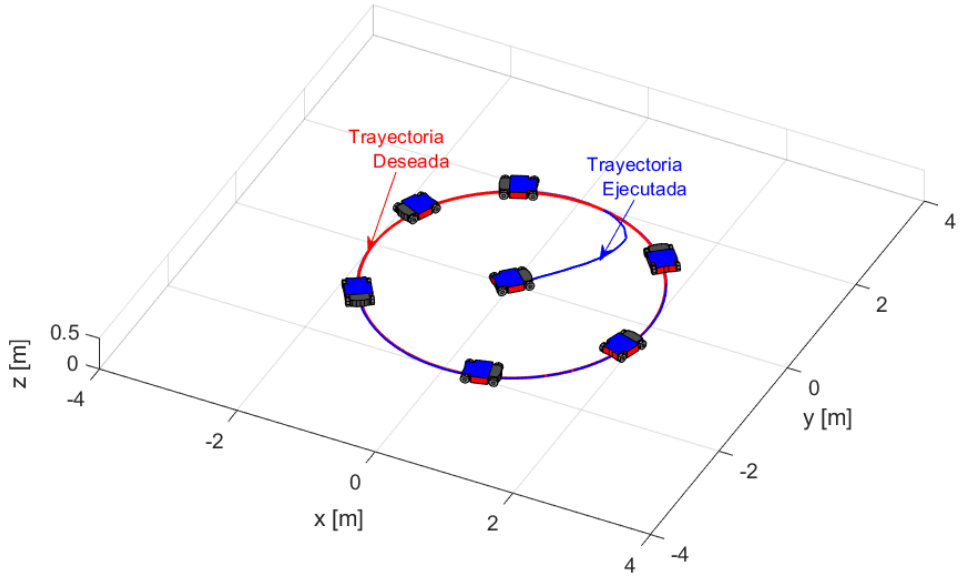
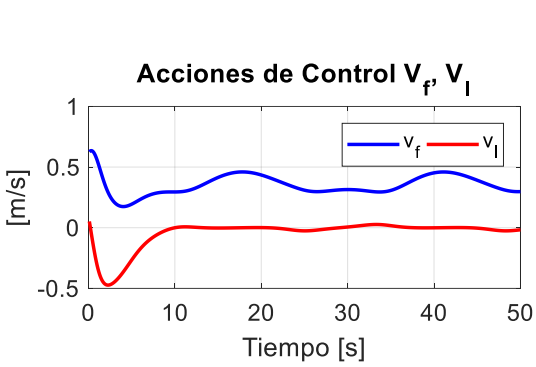


Fig. 6.47 Movimiento del robot ejecutado en la experimentación.

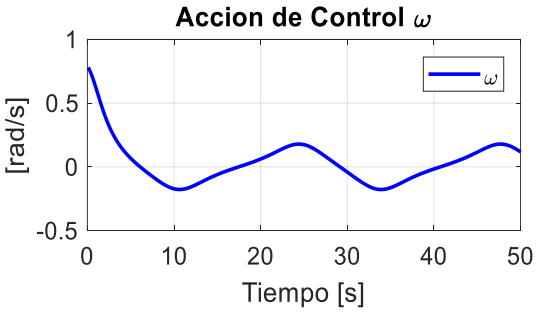
Elaborado por: Santiago Álvarez.

C. Simulación de control autónomo (Trayectoria 2)

El controlador planteado en el presente trabajo permite no solamente ejecutar una sola trayectoria, es decir, tiene la capacidad de realizar varias trayectorias que el usuario defina, es por esto que se realiza la simulación de otra trayectoria definida como $x_d = \frac{1}{2}\cos\left(\frac{2t}{5}\right) + \frac{t}{20} - 1$ [m] , $y_d = \frac{11}{50}t - 4$ [m] y un ángulo deseado para el trayecto definido como $\theta_d = \arctan\left(\frac{\dot{y}_d}{\dot{x}_d}\right)$[rad], las condiciones iniciales del robot están dadas por $x_o = -4$[m], $y_o = -3$[m] y $\theta_o = 0$ [rad] con tiempo de simulación de 50 [s]. En la Fig. 6.48 (a) se presenta las velocidades lineales que entrega el controlador, en la Fig. 6.48 (b) se ve la velocidad angular. En la Fig. 6.49 (a) se observa el comportamiento del error durante el tiempo de ejecución de la simulación y en la Fig. 6.49 (b) se encuentra el error de rotación del robot en la simulación. por último, se presenta el movimiento realizado por el robot en la Fig. 6.50. En el Anexo 5 se presenta el programa para simular varias trayectorias.



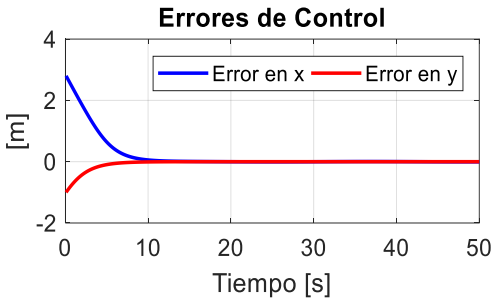
c) Evolución de velocidades lineales v_f y v_l



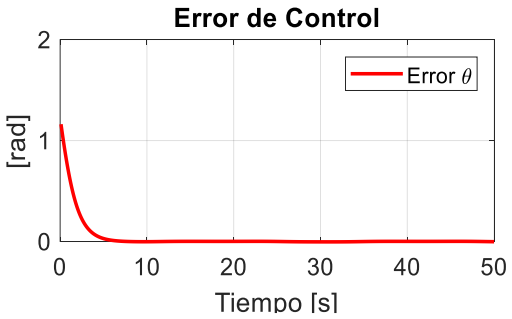
d) Evolución de la velocidad angular del robot ω

Fig. 6.48 Velocidades lineales y velocidad angular de control.

Elaborado por: Santiago Álvarez



a) Errores de control en x,y de la trayectoria 2 simulada.



b) Error de control en rotación de la trayectoria 2 simulada.

Fig. 6.49 Comportamiento de los errores de control en la trayectoria 2 simulada.

Elaborado por: Santiago Álvarez

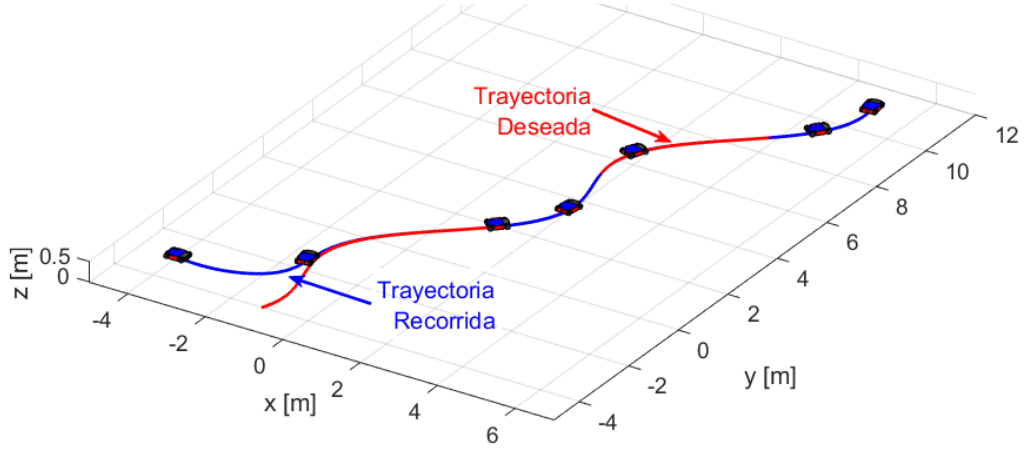
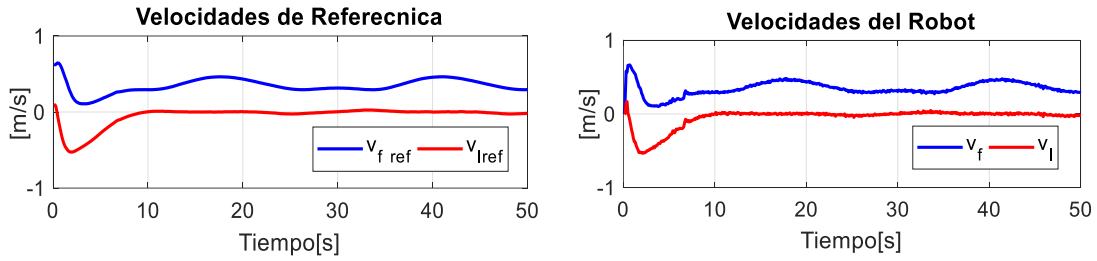


Fig. 6.50 Movimiento del robot ejecutado en la simulación.

Elaborado por: Santiago Álvarez

D. Experimentación de control autónomo (Trayectoria 2)

Para validar la simulación realizada de la trayectoria 2 se ejecuta una prueba experimental con el robot. El resultado de la experimentación se presenta a continuación: en la Fig. 6.51 (a) se observa la evolución de la velocidad lineal frontal y lateral de referencia que genera el controlador las mismas que son aplicadas al robot, en la Fig. 6.51 (b) se indica la lectura de la velocidad real del robot omnidireccional. En la Fig. 6.52 (a) se observa la velocidad angular de referencia que entrega el controlador aplicada al robot y en Fig. 6.52 (b) se observa la velocidad angular real del robot. Los errores de control del experimento se plantan en Fig. 6.53 (a) y Fig. 6.53 (b) respectivamente para posición y rotación. Por último, en la Fig. 6.54 se observa el movimiento realizado por el robot, logrando apreciar cómo sigue con la trayectoria planificada. En el **Anexo 5** se presenta el código del programa para ejecutar la experimentación del seguimiento de varias trayectorias.

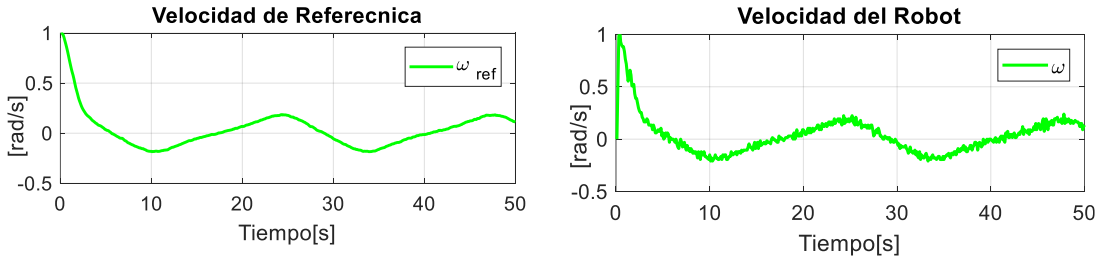


c) Evolución de velocidades de referencia $v_{f\ ref}$ y $v_{l\ ref}$

d) Evolución de las velocidades obtenidas del robot v_f y v_l

Fig. 6.51 Velocidades lineales generadas por el controlador y velocidades lineales reales del robot.

Elaborado por: Santiago Álvarez

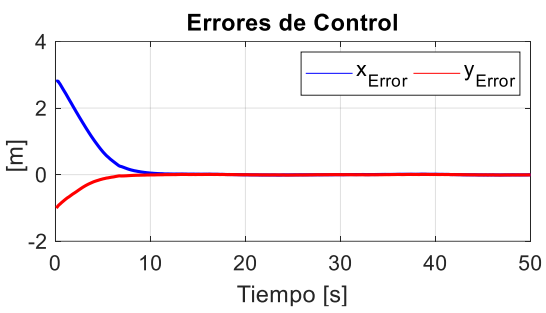


c) Evolución dela velocidad angular de referencia ω_{ref}.

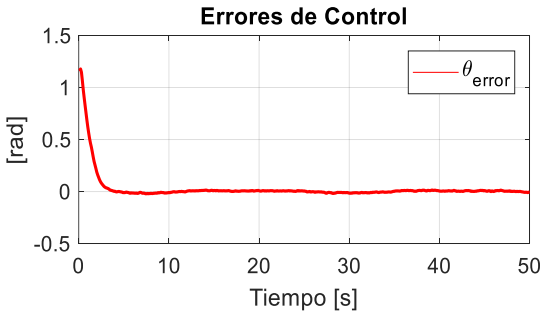
d) Evolución dela velocidad angular obtenida del robot ω.

Fig. 6.52 Velocidad angular generada por el controlador y velocidad angular real del robot.

Elaborado por: Santiago Álvarez



c) Error de control en x,y de la trayectoria ejecutada.



d) Error de control de rotación en la trayectoria ejecutada.

Fig. 6.53 Comportamiento de los errores de control del robot omnidireccional.

Elaborado por: Santiago Álvarez

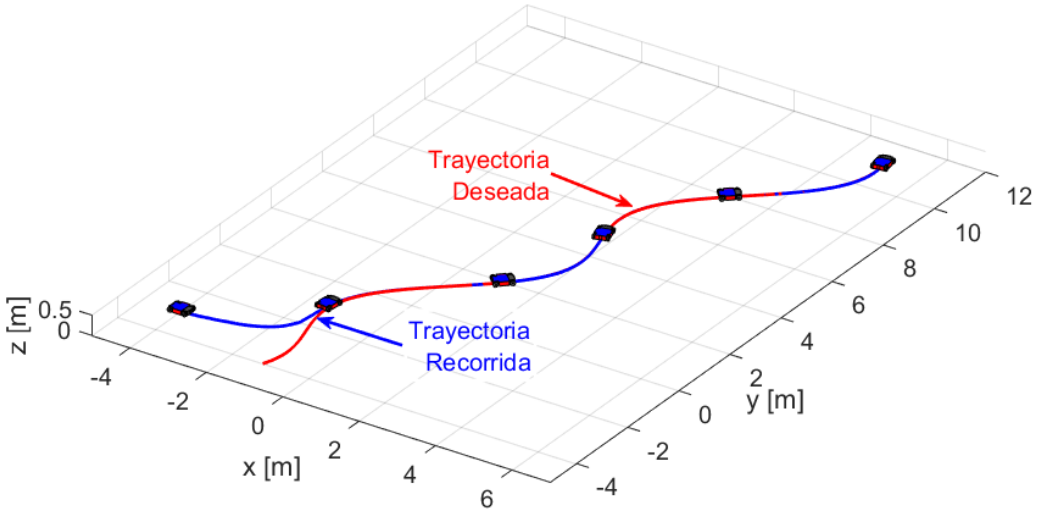


Fig. 6.54 Movimiento del robot ejecutado en la experimentación.

Elaborado por: Santiago Álvarez.

6.9. Navegación Autónoma

El método empelado para la navegación autónoma del robot omnidireccional es de manera local, es decir una vez dado su trayectoria a seguir el robot mapea su entorno para observar si existe obstáculos presentes en el camino, según el mapeo del robot da como resultado la modificación del camino a seguir por el robot. Conjuntamente con el algoritmo de seguimiento de rutas el robot omnidireccional sigue el trayecto trazado si existe un obstáculo, de ser el caso que no existiera un obstáculo el robot continuo su trayecto normal predefiniendo por el usuario.

6.9.1.1. Planificación de trayectos

Existe varios métodos para planificar o dar un trayecto al robot para que se desplace dentro de un entorno, aunque no se posea un conocimiento previo del mismo, se data entre planificación global o local.

Planificación global: Construir o planificar la ruta que lleve al robot a la meta determinada por el usuario, según las especificaciones del problema que debe resolverse, mediante esta planificación se tiene un aproximado del camino real que debe seguir el robot, ya que en la realización de esta acción no se consideran los detalles del entorno local del robot.

Planificación local: Resuelve las obstrucciones sobre la ruta global en el entorno local al robot para determinar la ruta real que guiara al robot, esta planificación local se lo hace mediante sensores, es decir, mediante el uso del sensor laser LIDAR para determinar obstáculo en el trayecto del robot móvil.

Nota: Dentro de este proyecto, se emplea el uso de trayectorias predefinidas, dadas en manera parametrizada de coordenadas polares dentro de un intervalo de tiempo, es decir la trayectoria global viene ya definida por el usuario para posteriormente modificarla de manera local por el método de evasión de obstáculos.

6.9.2. Trayectoria segura

Para cumplir con el objetivo de que el robot omnidireccional siga un trayecto predefino por el usuario o según el entorno lo requiera, se elabora un método de planificación local para que dicho robot tenga la capacidad de trasladarse de un lugar hacia otro evitando obstáculos que se pueden encontrar en su trayecto.

6.9.2.1. Método para evitar obstáculos

El método de fuerza ficticia es un método implementado para planificar caminos y facilitar la locomoción de un robot dentro de entornos no estructurados y estructurados, se basa en la generación de un campo potencial virtual el cual genera una fuerza que repela un objeto de otro. Gracias a su simplicidad matemática hace que se pueda modificarlo para ser utilizado en diferentes robots, en este caso se emplea para que un robot omnidireccional cumple una trayectoria evitando obstáculos dentro de la misma.

Se define la fuerza de repulsión entre el obstáculo como una fuerza virtual que va cambiando continuamente según la distancia que se encuentre el robot del obstáculo y depende también de las velocidades actuales en la que se encuentra la plataforma robótica.

La fuerza virtual se define como:

Si $d_{obs}(k) > d_{mx}$

$$fr(k) = 0$$

Si $d_{obs}(k) \leq d_{mx}$

$$fr(k) = k_f \frac{\eta}{\left| d_m - d_{obs}(k) \right|} \tag{6.41}$$

Donde:

$d_{obs}(k)$ representa la distancia entre el robot y el obstáculo presente en la trayectoria del robot.

d_{mx} la distancia máxima para definir la zona de repulsión entre el obstáculo y el robot, con esta constante se especifica a qué momento el robot empieza el método de evasión.

d_m es la distancia mínima que el robot va estar del objeto al momento de evadir el obstáculo para no entrar en colisión.

k_f representa una constante positiva para el radio de cobertura de la fuerza ficticia.

η es una constante que varía de acuerdo a la velocidad máxima del robot y la velocidad actual, se hace un promedio de las velocidades lineales actuales del robot con las velocidades máximas y se define dela siguiente manera.

$$\eta = \lambda \frac{v_f(k) + v_l(k)}{v_{f_{\max}} + v_{l_{\max}}} \tag{6.42}$$

Donde λ, es una constante que pesa el promedio de las velocidades máximas y actuales, esta constante se define como $0 < \lambda \leq 1$.

El la Fig.6.55 se puede observar cómo se comporta el método de fuerza virtual para evadir obstáculos dentro de una trayectoria definida.

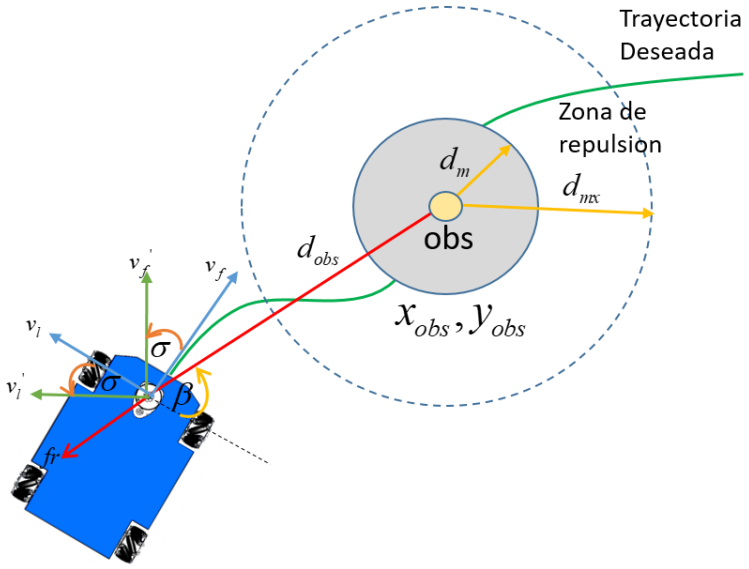


Fig. 6.55 Método de evasión de obstáculo por fuerza virtual

Elaborado por: Santiago Álvarez

Mediante el esquema de la Fig.6.55 se puede identificar a β que es el ángulo de detección del obstáculo, mediante este ángulo se puede definir la dirección de evasión del obstáculo, σ es el ángulo en que el robot va a rotar en base a la fuerza ficticia calculada por (6.41) mientras que v_f' y v_l' son las velocidades lineales modificadas en función a la fuerza que se genera por presencia del obstáculo.

Para lograr que el robot cambie su posición actual en base a la fuerza virtual se emplea la siguiente mitología, empelando una rotación en función al ángulo α hacia las velocidades deseadas del robot omnidireccional.

$$\begin{bmatrix} \dot{\delta}_{xd}'(k) \\ \dot{\delta}_{yd}'(k) \end{bmatrix} = \varepsilon \begin{bmatrix} \cos(\sigma(k)) & -\sin(\sigma(k)) \\ \sin(\sigma(k)) & \cos(\sigma(k)) \end{bmatrix} \begin{bmatrix} \dot{\delta}_{xd}(k) \\ \dot{\delta}_{yd}(k) \end{bmatrix} \tag{6.43}$$

La constante ε se emplea para reducir la velocidad de giro de la plataforma en el instante de evadir el obstáculo, esta constante está dentro de $0 < \varepsilon < 1$, pero si $fr = 0$ entonces $\varepsilon = 1$ que representa el seguimiento normal del trayecto predefinido.

El sentido de evasión viene definido por:

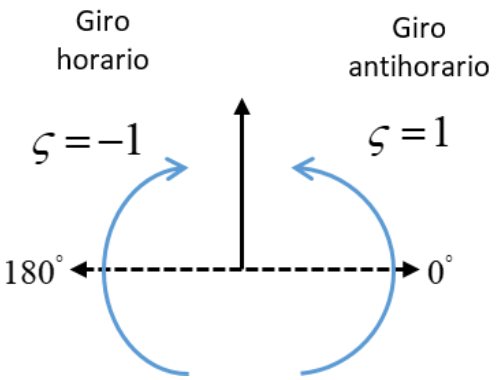


Fig. 6.56 Definición de sentido de evasión.

Elaborado por: Santiago Álvarez

según las Fig.6.56 si:

$$\frac{\pi}{2} \geq \beta > \frac{3}{2}\pi \quad \text{entonces } \varsigma = 1$$

$$\frac{\pi}{2} < \beta \leq \frac{3}{2}\pi \quad \text{entonces } \varsigma = -1$$

Para definir el ángulo de rotación para que el robot evada el obstáculo se calcula de la siguiente manera:

$$\sigma(k) = f_r * \varsigma \tag{6.44}$$

6.9.2.2. Detección de obstáculos con LASER

EL sensor Laser que usara en el presente proyecto cuanto con un SDK para desarrollar aplicaciones propias y poder usarlas para conveniencia de cada usuario, en este caso se desarrolló una DLL para leer los datos que genera y poder procesar en el software de MATLAB. En la Fig. 6.57 se observa de qué manera el sensor detecta putos en un radio de 360º.

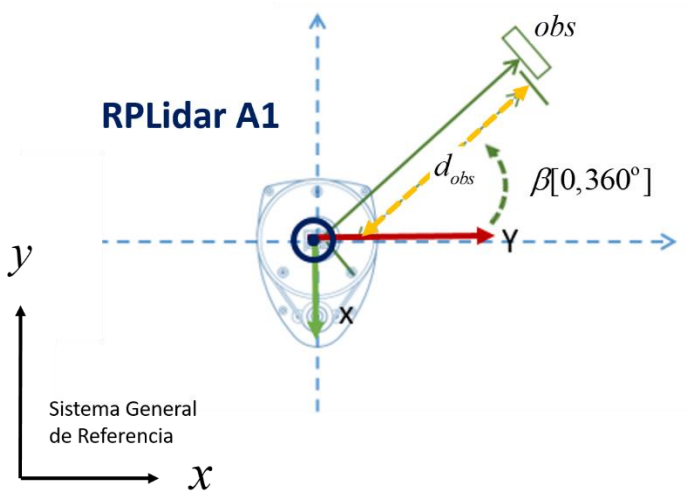


Fig. 6.57 Datos que genera el sensor RPLidar A1.

Elaborado por: Santiago Álvarez

Donde d_{obs} representa la distancia hacia un objeto y β representa el ángulo con que incide hacia el objeto el frente del láser. Para obtener las coordenadas de dicho objeto detectado se emplea las ecuaciones siguientes:

$$x_{obs} = d_{obs}\cos(\beta) \tag{6.45}$$

$$y_{obs} = d_{obs}\sin(\beta) \tag{6.46}$$

Los datos obtenidos se pueden graficar obteniendo un escaneo de objetos presentes en el entorno donde se encuentra el robot, el Fig. 6.58 se observa los datos en MATLAB del sensor Laser.

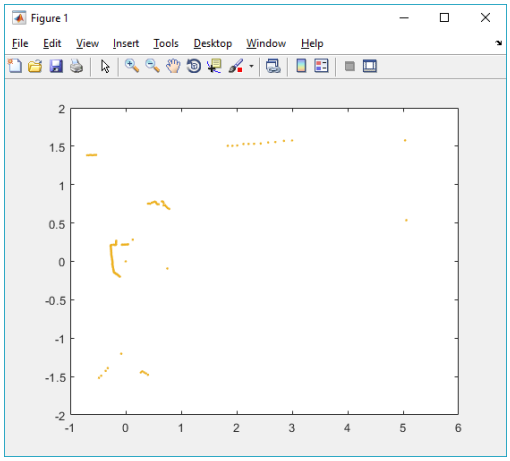


Fig. 6.58 Escaneo 2D con el sensor Laser RPLidar A1 en MATLAB.

Elaborado por: Santiago Álvarez

A. Experimentación de trayectoria segura (Trayectoria 1)

Para el proceso de experimentación con el robot omnidireccional evitando obstáculos, se plantea una prueba en la que el robot tenga que seguir la trayectoria definida como: $x_d = r\cos\left(\frac{t}{5}\right)$ [m], $y_d = r\sin\left(\frac{t}{5}\right)$ [m] , donde r es el radio de la circunferencia a seguir a demás se tiene que las velocidades del trayecto están definidas como: $\dot{x}_d = -\frac{r}{5}\sin\left(\frac{t}{5}\right)$, $\dot{y}_d = \frac{r}{5}\cos\left(\frac{t}{5}\right)$ y el Angulo deseado durante la trayectoria está definido por $\theta_d = \arctan\left(\frac{\dot{y}_d}{\dot{x}_d}\right)$.

El experimento realizado se lo ejecuta durante un tiempo de 60[s] con un periodo de muestreo de 100 [ms], en la Fig. 6.59 se presenta el robot listo para la prueba experimental en un lugar que cuenta con obstáculos presentes en el trayecto.

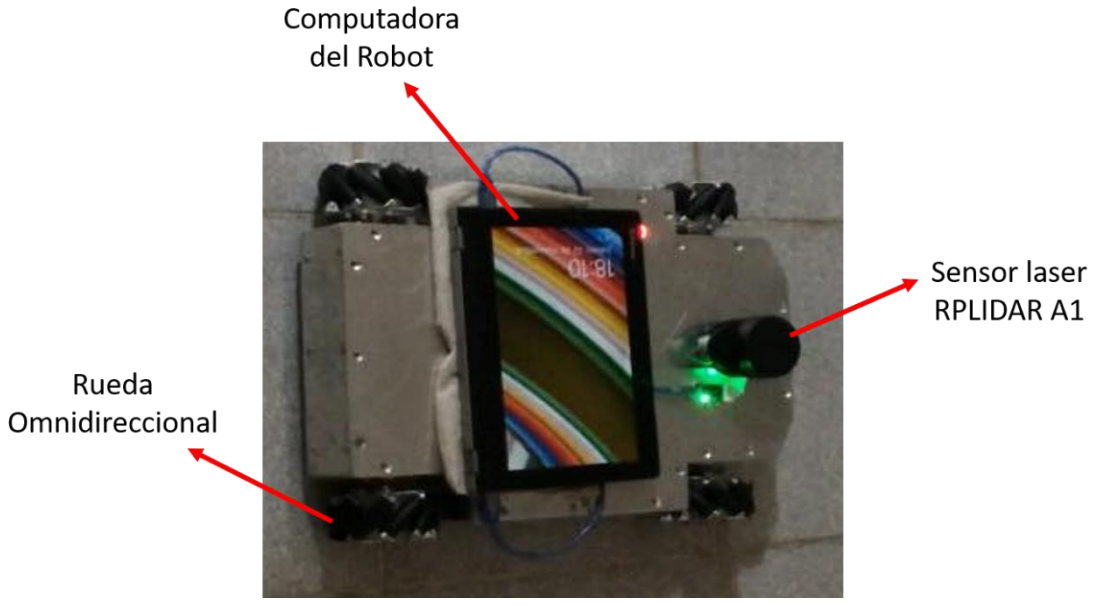


Fig. 6.59 Robot omnidireccional con sensor laser.

Elaborado por: Santiago Álvarez

En la Fig. 6.60 se presenta el movimiento ejecutado por el robot dentro del ambiente con obstáculos, se puede observar que en presencia de obstáculos el robot cambia su trayectoria para no colisionar con el objeto en ruta, una vez que el robot detecta que ha evitado el obstáculo vuelve nuevamente a su trayecto original, es decir, retorna a ruta para cumplir con la trayectoria planificada.

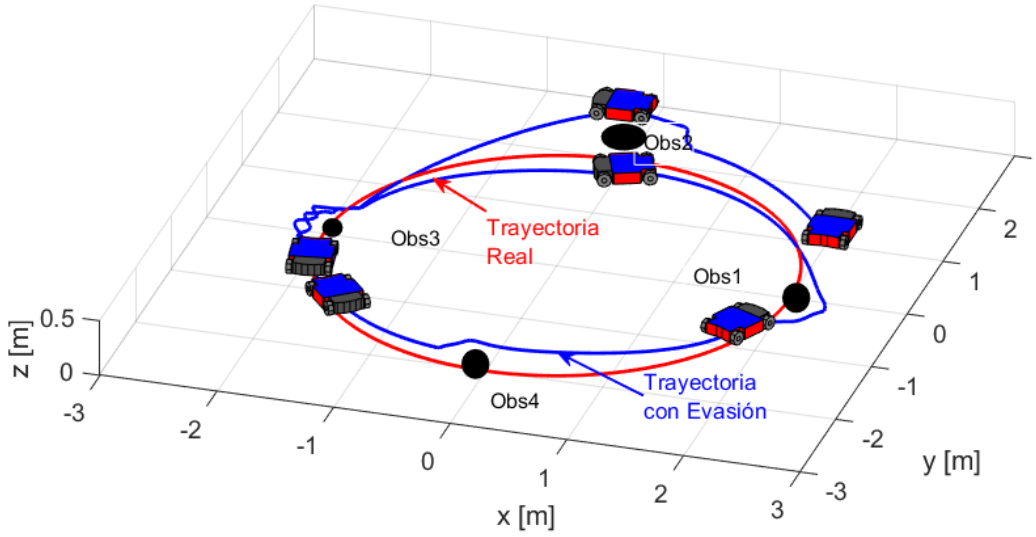
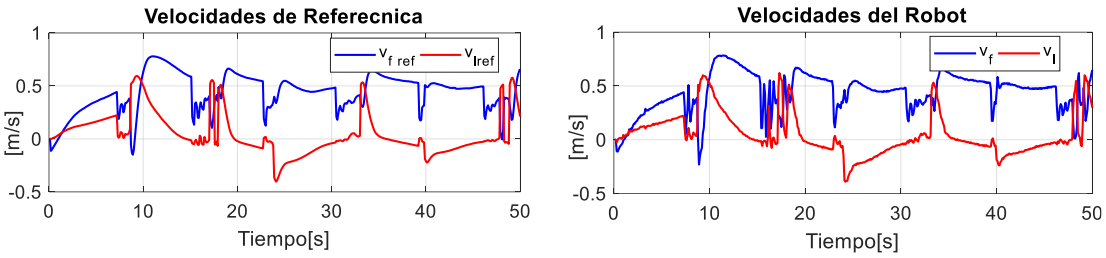


Fig. 6.60 Movimiento ejecutado por el robot omnidireccional.

Elaborado por: Santiago Álvarez

La velocidad de referencia generada por el controlador se puede observar en la Fig. 6.61 (a), en la Fig. 6.61 (b) se aprecia la velocidad real del robot omnidireccional, tanto para velocidad frontal velocidad lateral.



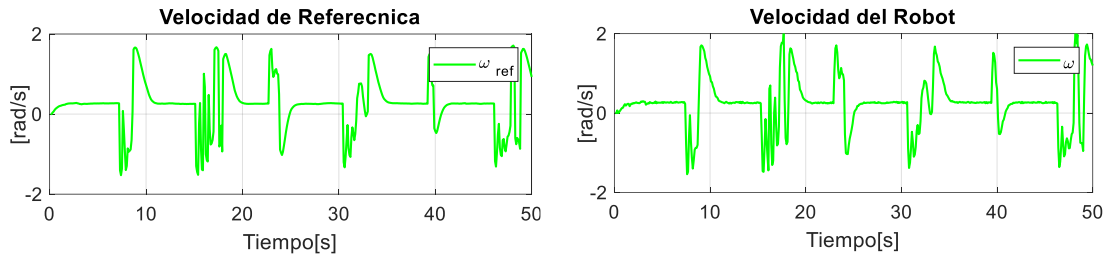
a) Evolución de velocidades de referencia $v_{f\ ref}$ y $v_{l\ ref}$

b) Evolución de las velocidades obtenidas del robot v_f y v_l

Fig. 6.61 Velocidades lineales del robot omnidireccional de la experimentación.

Elaborado por: Santiago Álvarez

En la Fig. 6.62 (a) se observa la velocidad angular que genera el controlador y es enviada hacia el robot, mientas que en la Fig. 6.62 (b) se observa la velocidad angular que ejecuta el robot durante su movimiento en el ambiente para cumplir con la trayectoria predeterminada.



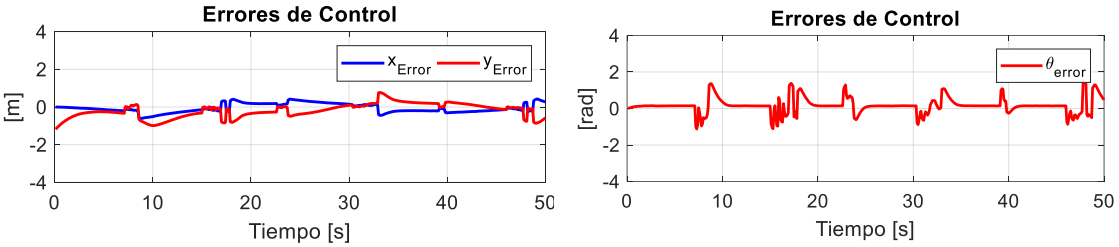
a) Evolución dela velocidad angular de referencia ω_{ref}.

b) Evolución dela velocidad angular obtenida del robot ω.

Fig. 6.62 Velocidad angular del robot omnidireccional en la experimentación.

Elaborado por: Santiago Álvarez

Con el fin de validar el control y observar como el robot retoma la ruta deseada se presenta los errores de control, en la Fig. 6.63 (a) se observa los errores $\tilde{x}$, $\tilde{y}$ durante el tiempo de ejecución de trayectoria. En la Fig. 6.63 (b) se describe el error de rotación $\tilde{\theta}$, como se observa en las figuras los errores durante la presencia de obstáculos aumentan, mientras que una vez que el robot evite el obstáculo retornan a ser cero, es decir, el robot omnidireccional cumple con la trayectoria predefinida por el usuario esquivando objetos.



a) Error en x,y de la trayectoria ejecutada evitando obstáculos.

b) Error de rotación en la trayectoria ejecutada evitando obstáculos

Fig. 6.63 Comportamiento de los errores del robot omnidireccional

Elaborado por: Santiago Álvarez

La trayectoria modificada por el algoritmo planteado versus la trayectoria original se observa en la Fig. 6.64, apreciando que en presencia de obstáculos la trayectoria deseada cambia mientras el robot se encuentre en peligro de colisión, de igual manera se presenta los datos del Laser obtenidos.

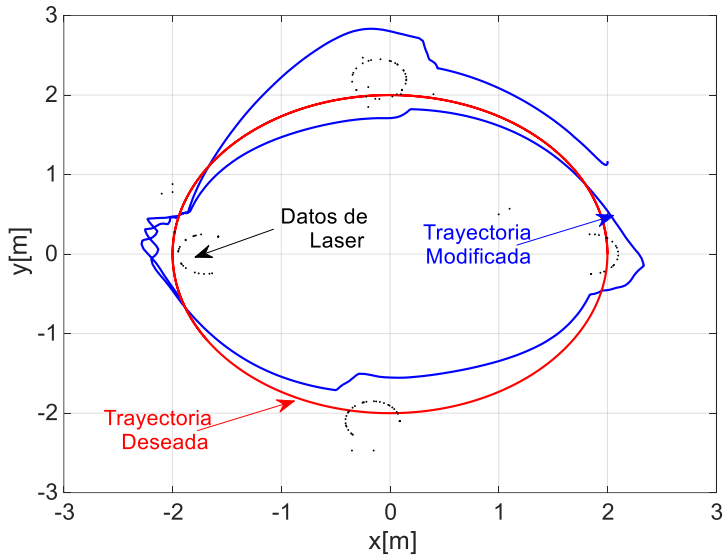


Fig. 6.64 Trayectoria modifica vs. trayectoria deseada.

Elaborado por: Santiago Álvarez

B. Experimentación de trayectoria segura (Trayectoria 2)

Con la ejecución de varios experimentos se valida que el algoritmo planteado en este trabajo, para observar el funcionamiento adecuado. Es por eso que se ejecuta otro experimento con el robot para evitar obstáculos, la prueba consiste en seguir la trayectoria descrita como: $x_d = 0.8\cos(0.27t) + 0.13t - 2$, $y_d = 0.2t - 4$ y como ángulo deseado se tiene $\theta_d = \arctan\left(\frac{\dot{y}_d}{\dot{x}_d}\right)$, teniendo en cuenta que para la velocidad se utiliza la derivada tanto para $\dot{x}_d$, $\dot{y}_d$, en el **Anexo 8** se presenta el programa con las constantes definidas para evitar obstáculos.

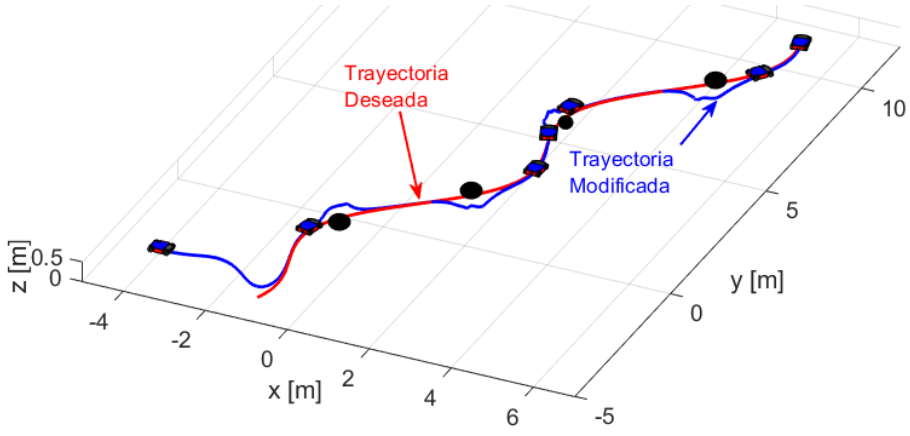
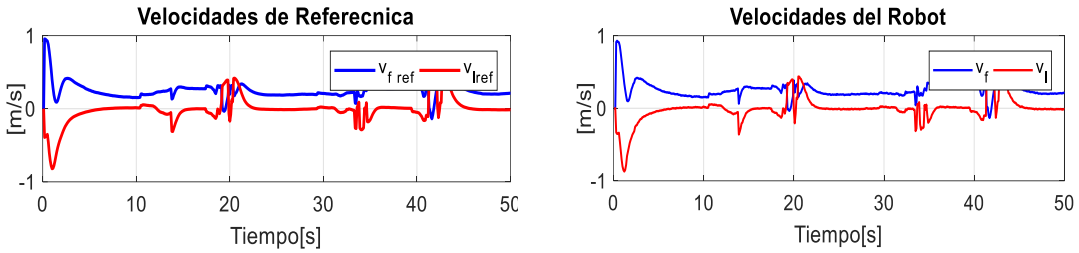


Fig. 6.65 Movimiento ejecutado por el robot omnidireccional.

Elaborado por: Santiago Álvarez

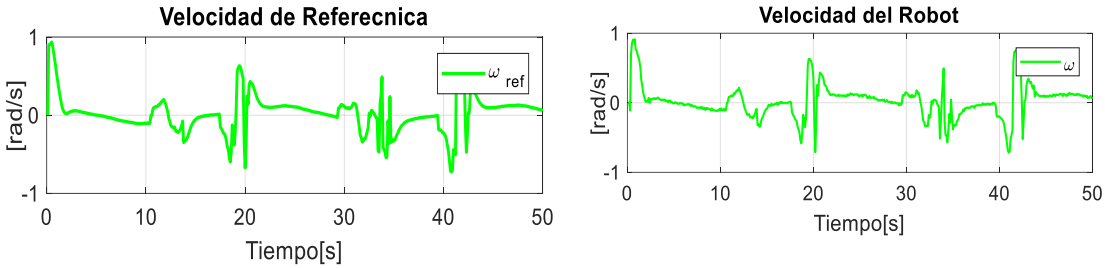


a) Evolución de velocidades de referencia $v_{f\ ref}$ y $v_{l\ ref}$

b) Evolución de las velocidades obtenidas del robot v_f y v_l

Fig. 6.66 Velocidades lineales del robot omnidireccional de la experimentación.

Elaborado por: Santiago Álvarez

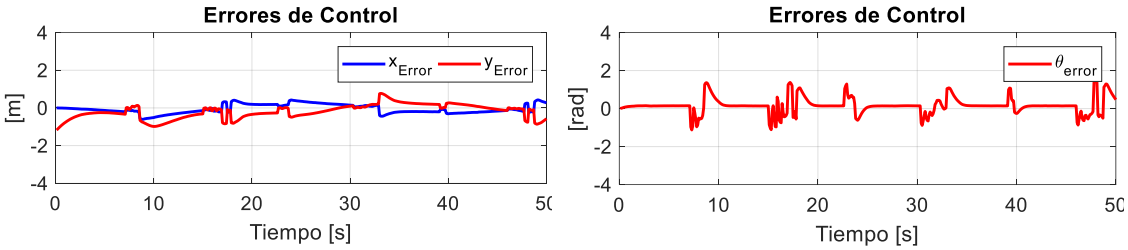


a) Evolución dela velocidad angular de referencia ω_{ref}.

b) Evolución dela velocidad angular obtenida del robot ω.

Fig. 6.67 Velocidad angular del robot omnidireccional en la experimentación.

Elaborado por: Santiago Álvarez



a) Error en x,y de la trayectoria ejecutada evitando obstáculos.

b) Error de rotación en la trayectoria ejecutada evitando obstáculos

Fig. 6.68 Comportamiento de los errores del robot omnidireccional

Elaborado por: Santiago Álvarez

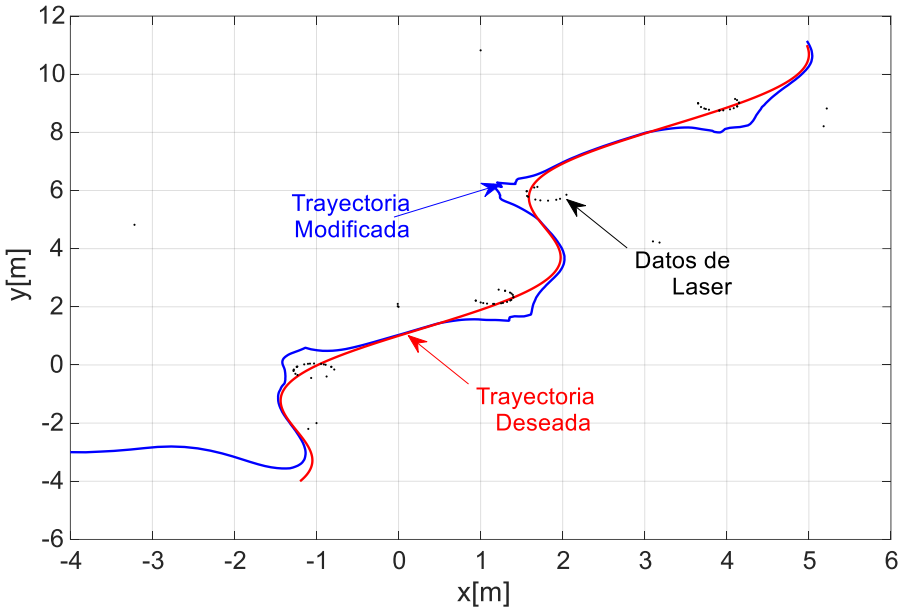


Fig. 6.69 Trayectoria modifica vs. trayectoria deseada

Elaborado por: Santiago Álvarez

6.9.3. Interfaz Gráfica Programada

Con el propósito de realizar pruebas de validación se desarrolla una interfaz para interactuar con el robot y a demás probar el algoritmo de control, tanto para simular y experimentar. Esta interfaz es desarrollada desde MATLAB permitiendo conectarnos con el robot y poder enviar las velocidades de maniobrabilidad para que el robot se desplace Fig. 6.70. En el Anexo 9 se encuentra la programación para el desarrollo de la interfaz de operación.

Fig. 6.70 Interfaz en MATLAB para probar el algoritmo de control.

Elaborado por: Santiago Álvarez

La interfaz cuenta con cuatro opciones para el usuario clasificadas de la siguiente manera: simulador, control manual, control automático y navegación segura.

Simulador: esta opción permite realizar la simulación del algoritmo de seguimiento de trayectoria predeterminada con un menú de selección de tres trayectorias diferentes, en esta pantalla se grafica el movimiento del robot según a la trayectoria seleccionada y por último se grafica las velocidades de control y los errores producidos de control Fig.6.71.

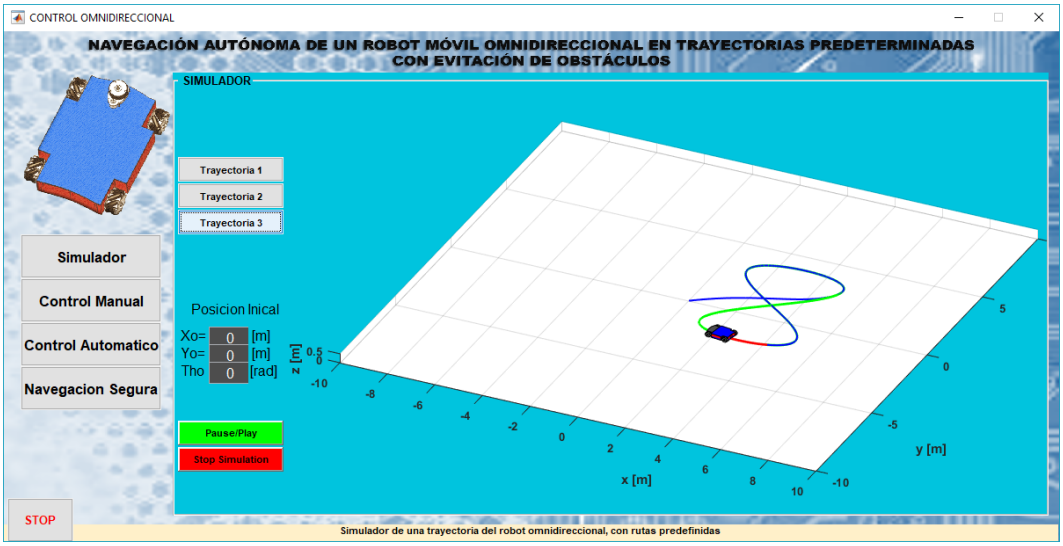


Fig. 6.71 Interfaz en MATLAB opción para simular las trayectorias.

Elaborado por: Santiago Álvarez

Control Manual: esta opción permite ya interactuar con el robot, es decir, aquí se pide insertar velocidades al robot para observar el movimiento que realiza según la velocidad que se ingrese. La pantalla cuenta con tres botones principales que permiten conectar al robot, ejecutar el programa de envío de velocidades y detener al robot, en la Fig. 6.72 se puede observar los indicadores como cambian al momento de conectares con el robot y mostrar en pantalla la velocidades enviadas y recibidas. Mediante la cinemática directa se pude saber la ubicación del robot en el plano referencial.

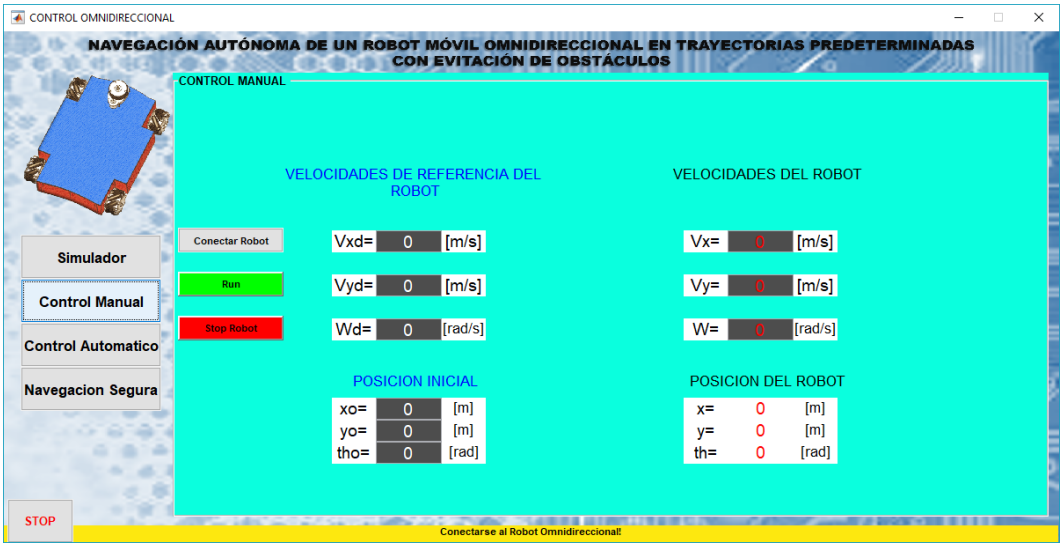


Fig. 6.72 Interfaz en MATLAB, control manual del robot Omnidireccional.

Elaborado por: Santiago Álvarez

Control Automático: con el propósito de ver experimentalmente como el robot se incorpora al momento ejecutar un trayectoria predefinida y probada en simulación se procede a realiza experimentalmente el proceso Fig. 6.73.

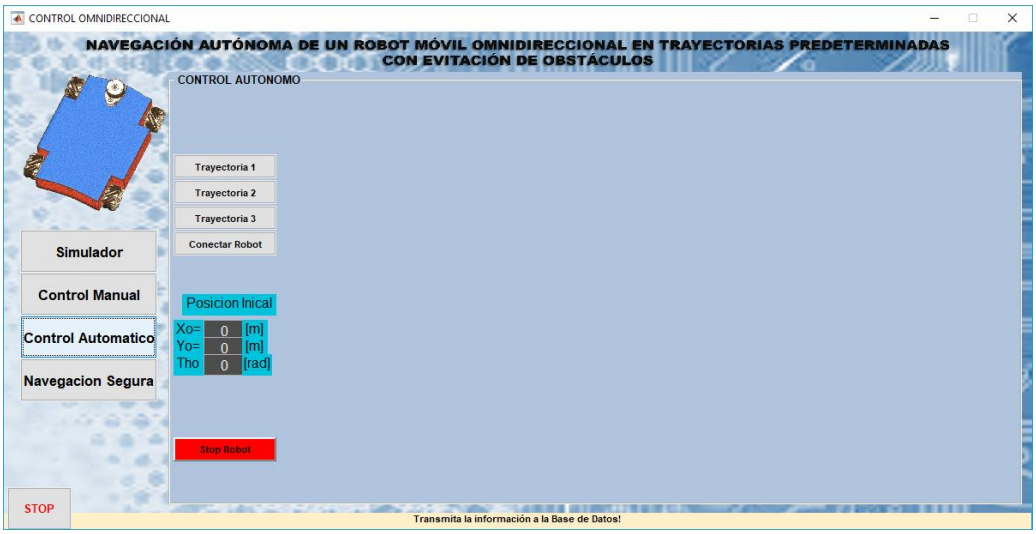


Fig. 6.73 Interfaz en MATLAB, control autónomo del robot Omnidireccional.

Elaborado por: Santiago Álvarez

De igual manera que la opción de control manual esta opción cuenta con botones de conectarse al robot para ejecutar la terea que se seleccione en cuanto a trayectorias, una vez realizada la tarea se indicara el movimiento realizado por el robot en pantalla y las gráficas de las velocidades del robot y los errores de control ya con pruebas realiza con el prototipo físico.

Navegación Segura: esta opción final ya sería la parte completa del presente trabajo investigativo, esta opción permite probar el robot experimentalmente dentro de un lugar semiestructurado evitando obstáculos para no colisionar con objetos y realizar adecuadamente la tarea propuesta, en esta opción de la interfaz se indicará la circulación del robot como se realizado, las velocidades de maniobrabilidad y por último los errores producido durante el trayecto seleccionado Fig. 6.74.

Fig. 6.74 Interfaz en MATLAB, navegación segura con evasión de obstáculos del robot Omnidireccional.

Elaborado por: Santiago Álvarez

6.9.4. Conclusiones y recomendaciones

A. Conclusiones

- Conforme a las distintas bibliografías analizadas, se permitió establecer la mejor manera para la construcción del robot omnidireccional, permitiendo seleccionar las llantas adecuadas en una configuración que consienta al robot desplazarse de una manera propicia sobre la superficie de trabajo. De igual manera en cuanto a la construcción mediante estas bases consultadas se pudo seleccionar el sensor LIDAR para determinar los obstáculos dentro de la trayectoria predefinida para el robot.
- La construcción del robot omnidireccional con materiales resistentes y de un tamaño apto para soportar carga, permitió realizar pruebas experimentales, las cuales validan el control autónomo para trayectorias predefinidas y además evadir obstáculos que están dentro de la tarea propuesta al robot, la configuración de las ruedas mecanum consintieron que el robot se desplace sin problema alguno sobre una superficie plana, en cualquier dirección según las velocidades que se le inyecte.
- Mediante el estudio de métodos para obtener el modelo cinemático de un robot omnidireccional, se destacó el método de matrices rotacionales el cual permitió obtener de una forma rápida la matriz jacobina que representa la cinemática de la plataforma, la cual mapea las velocidades del punto de operación del robot y las velocidades de maniobrabilidad.
- La odometría empleada mediante codificadores rotativos y las respectivas matrices homogéneas de transformación, directas e inversas facilitaron identificar la posición y la velocidad de desplazamiento del robot omnidireccional sobre el sistema de referencia en el que el robot actúa.
- Con el empleo del modelo cinemático del robot omnidireccional en el presente trabajo se pudo plantear una ley de control para el seguimiento autónomo de trayectorias predefinidas, corroborando que dicho controlador propuesto satisface el cumplimento de la tarea prepuesta. El controlador establecido permite validar de manera experimental y simulada el seguimiento de rutas que el usuario predefina.

- La detección de obstáculos mediante el sensor RPLidar permitió dotar al robot de la habilidad de evitar obstáculos dentro de su trayecto para llegar a un punto deseado, modificando de manera matemática la ruta cuando existe un obstáculo en la trayectoria original, formando así rutas seguras en tiempo real, es decir, mientras el robot interactúa en el medio de forma autónoma.
- Con el software MATLAB se logra realizar los cálculos matemáticos necesarios para que el robot actué de manera autónoma. Mediante el software se logra la interacción del robot con el sensor laser para evitar obstáculos realizando los cálculos debidos para corregir errores y modificar el trayecto con el propósito de evitar obstáculos.
- Las pruebas experimentales con el robot omnidireccional validan de mejor manera el modelo cinemático obtenido y el controlador propuesto, estableciendo una eficacia en el controlador y la corrección de velocidades para la operación correcta del robot omnidireccional. Mediante estas pruebas de igual forma se puede ajustar de mejor manera las ganancias del controlador para compensar los errores presentes en la tarea.

B. Recomendaciones

- Mediante este trabajo se propone a futuros investigadores tomar como referencia para la elaboración de algoritmos de control basados en la cinemática del robot omnidireccional con el propósito de plantear nuevos controladores para ejecutar más tareas e incluso dotar de mayor capacidad al robot para interactuar en ambientes más hostiles no estructurados.
- Plantear trabajos en los que el robot omnidireccional pueda realizar tareas de transporte de objetos empelando manipuladores robóticos, es decir, agregar a la plataforma uno o dos brazos robóticos para manipular y transportar objetos de un lugar hacia otro.
- Se recomienda saturar bien las velocidades del robot, es decir, saber cuáles son las velocidades máximas y mínimas de la plataforma para no tener inconvenientes con la parte física del robot.
- Para validar el buen funcionamiento de un robot, realizar pruebas a máximas escalas, inyectando diferentes velocidades para observar el comportamiento y

verificar que responde adecuadamente a cada comando que el usuario envié hacia la plataforma.

- Para mejorar la edometría de un robot que trabaje en espacios con mayor campo de desplazamiento, es recordable agregar sensores inerciales e incluso Las pruebas experimentales con el robot omnidireccional validan de mejor manera el modelo cinemático obtenido y el controlador propuesto, estableciendo una eficacia en el controlador y la corrección de velocidades para la operación correcta del robot omnidireccional. Mediante estas pruebas de igual forma se puede ajustar de mejor manera las ganancias del controlador para compensar los errores presentes en la tarea.

REFERENCIAS

Alfaro, T. (2006). *inf.* Obtenido de utfsm: https://www.inf.utfsm.cl/~noell/SMMC/mat2006/Cap3.pdf

Allen-Bradley. (02 de 05 de 2018). *rockwellautomation*. Obtenido de https://ab.rockwellautomation.com/es/Motion-Control/Single-Turn-High-Performance-Absolute-Encoder

Baker, A., & Mackenzie, I. (2008). Omnidirectional Drive Systems Kinematics and Control. *FIRST Robotics Conference*, 1-15.

Begum, A., Lee, M., & Kim, Y. J. (2010). A Simple Visual Servoing and Navigation Algorithm for an Omnidirectional Robot. *Human-Centric Computing (HumanCom)*, 10-16.

Biswas, J., & Veloso, M. (2010). WiFi localization and navigation for autonomous indoor mobile robots. *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, 4379-4384.

Chaturvedi, D. K. (2010). *Modeling and simulation of Systems using MATLAB and Simulink.* Boca Raton: Taylor & Francis.

C-NET. (1 de 11 de 2017). *Los robots de Nissan crean hasta 113 carros por hora* . Obtenido de https://www.cnet.com/es/imagenes/robots-nissan-crean-hasta-113-carros-por-hora-fotos/

Corke, P. (2011). *Robotics, Vision and Control.* Brisbane: Springer International Publishing AG 2017.

Diego, B., & Johnny, Y. (2014). *Diseño y fabricación de un prototipo de plataforma móvil omnidireccional para la empresa SIMYM.* Sangolquí: Universidad de las Fuerzas Armadas - ESPE.

E3CreaTIC. (2017). *casahermes*. Obtenido de Tienda actuadores neumaticos FESTO: https://casahermes.co/tienda/actuadores-cilindros-neumaticos-festo/

Forlizzi, J., & DiSalvo, C. (2006). Service robots in the domestic environment: a study of the roomba vacuum in the home. *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*, 258-265.

GARMIN. (10 de 04 de 2018). *buy.garmin*. Obtenido de https://buy.garmin.com/en-US/US/p/13195

GIJN. (1 de 11 de 2017). *Get Your Google Earth On: A Trainer's Guide*. Obtenido de https://gijn.org/2017/09/11/get-your-google-earth-on-a-trainers-guide/

Girdhar, Y., & Dudek, G. (2016). Modeling curiosity in a mobile robot for long-term autonomous exploration and monitoring. *Autonomous Robots*, 1267–1278.

Guillenxt. (1 de 11 de 2017). *Otra buena definición de Robótica*. Obtenido de http://www.guillenxt.com/2012/03/las-5-generaciones-de-la-robotica.html

INTPLUS. (10 de 05 de 2018). *superrobotica*. Obtenido de http://www.superrobotica.com/S330010.htm

Ismael, O. Y., & Hedley, J. (2016). Analysis, Design, and Implementation of an Omnidirectional Mobile Robot Platform. *American Scientific Research Journal for Engineering, Technology, and Sciences*, 195-209.

Iza, J., & Taco, L. (2016). *INVESTIGACIÓN DE LA MANIOBRABILIDAD DE UNA PLATAFORMA ROBÓTICA CON SISTEMA DE TRACCIÓN OMNIDIRECCIONAL E IMPLEMENTACIÓN EN EL PROYECTO DE INVESTIGACIÓN "TELE – OPERACIÓN BILATERAL CORPORATIVO DE MÚLTIPLES MANIPULADORES MÓVILES" APROBADO POR EL CONSORCIO.*. Latacunga: Universidad de las Fuerzas Armadas-ESPEL.

Kundu, A. S., Mazumdera, O., Dhara, A., Lenkab, P. K., & Bhaumikc, S. (2016). Scanning Camera and Augmented Reality Based Localization of Omnidirectional Robot for Indoor Application. *2016 IEEE International Symposium on Robotics and Intelligent Sensors, IRIS 2016*, 17-20.

Kuo, B. C. (2000). *Sistemas de Control Automático.* Illinois: Prentice Hall.

Ogata, K. (2010). *Ingenieria de control moderna.* Madrid: PEARSON EDUCACIÓN.

Ollero Baturone, A. (2001). *ROBÓTICA; MANIPULADORES Y ROBOTS MÓVILES.* Madrid: MARCOMBO, S.A.

Ollero, A. (2001). *Robótica Manipuladores y robots móviles.* Madrid: Marcombo.

Pérez, V. D. (2015). *Implementación de algoritmos de determinación de rutas para el Robotino de Festo.* Quito: Escuela Politécnica Nacional.

PROEMETEC. (05 de 05 de 2018). *PROEMETEC*. Obtenido de https://www.prometec.net/producto/mpu-gy-521-acelerometro-giroscopo/#

Recelogic. (10 de 05 de 2018). *racelogic.support*. Obtenido de https://racelogic.support/VBOX_Mining/Hardware_Info/IMU

RoboPeak. (06 de 2018). *robotshop*. Obtenido de https://www.robotshop.com/en/rplidar-a2m8-360-laser-scanner.html

S.L., D. (01 de 05 de 2018). *diteico*. Obtenido de Catalogo Actuadores Neumaticos: http://diteico.com/catalogos/catalogos-actuadores-neumaticos/

Secretaría Nacional de Planificación y Desarrollo. (2012). *Transformación de la Matriz Productiva: Revolución productiva a través del conocimiento y el talento humano.* Quito: SENPLADES.

Sen, S. (2014). Outdoor Robot Location and Navigation Based on RTK in Intelligence Community. *School of Control Science and Engineering*, 1-12.

Siciliano, B., & Khatib, O. (2008). *Handbook of Robotics.* Berlin: Springer-Verlag Berlin.

Siegwart, R., Nourbakhsh, I., & Scaramuzza, D. (2004). *Introduction to autonomous Mobile robots.* London: Massachisetts Institute of Techlonogy.

Suzuki, M., Blij, J. v., & Floreano, D. (2006). Omnidirectional Active Vision for Evolutionary Car Driving. *Intelligent Autonomous Systems*, 1-8.

Suzuki, T., & Takahashi, M. (2011). Obstacle Avoidance for Autonomous Mobile Robots Based on Position Prediction Using Fuzzy Inference. *Computer and Information Science*, 1-11.

Tecnología Técnica . (2 de 11 de 2017). *Sistemas de control*. Obtenido de http://www.tecnologia-tecnica.com.ar/sistemadecontrol/index%20sistemasdecontrol.htm

Tsai, C.-C., Huang, H.-C., & Lin, S.-C. (2011). FPGA-Based Parallel DNA Algorithm for Optimal Configurations of an Omnidirectional Mobile Service Robot

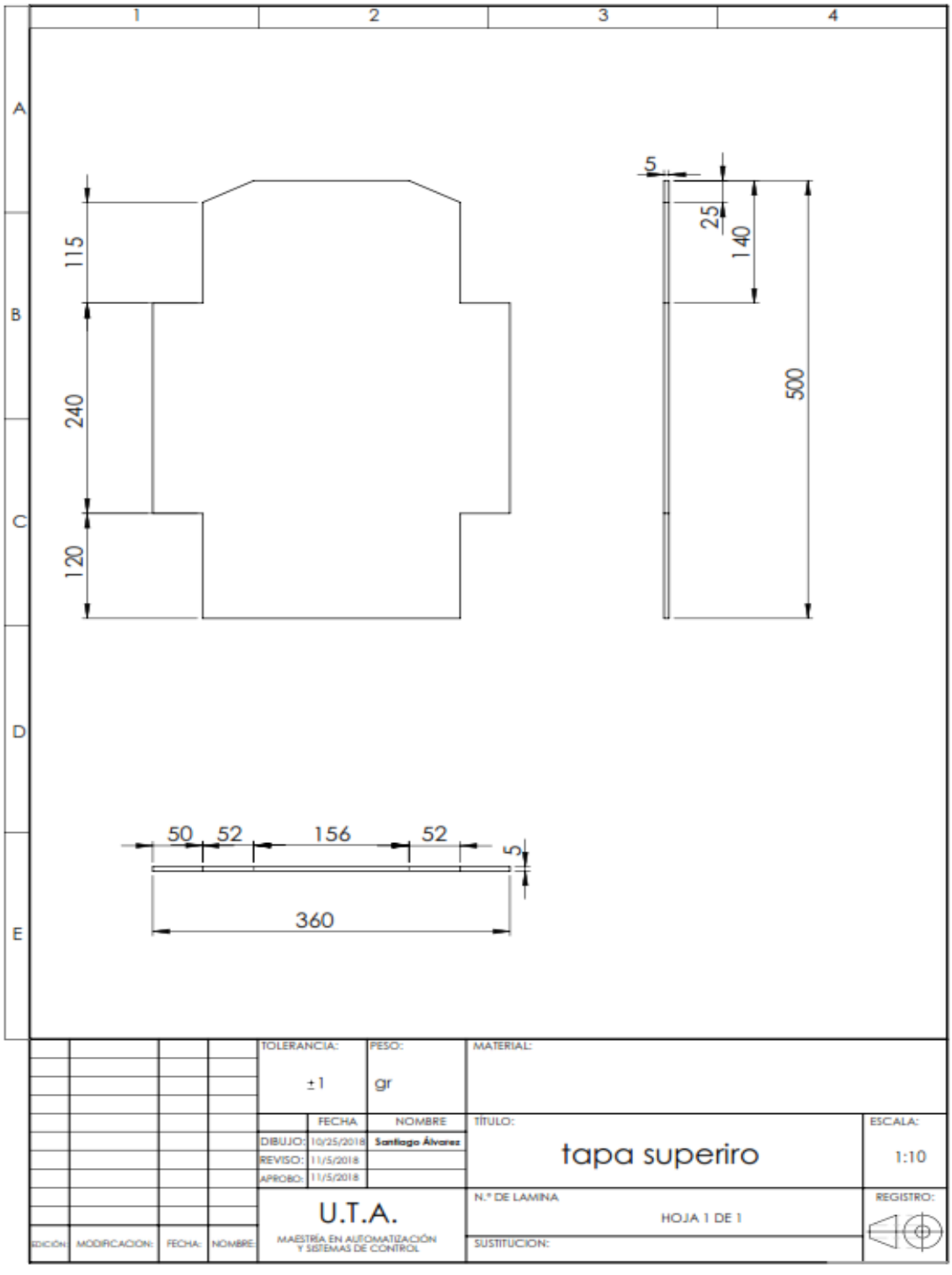
Performing Fire Extinguishment. *IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS*, 1016-1025.

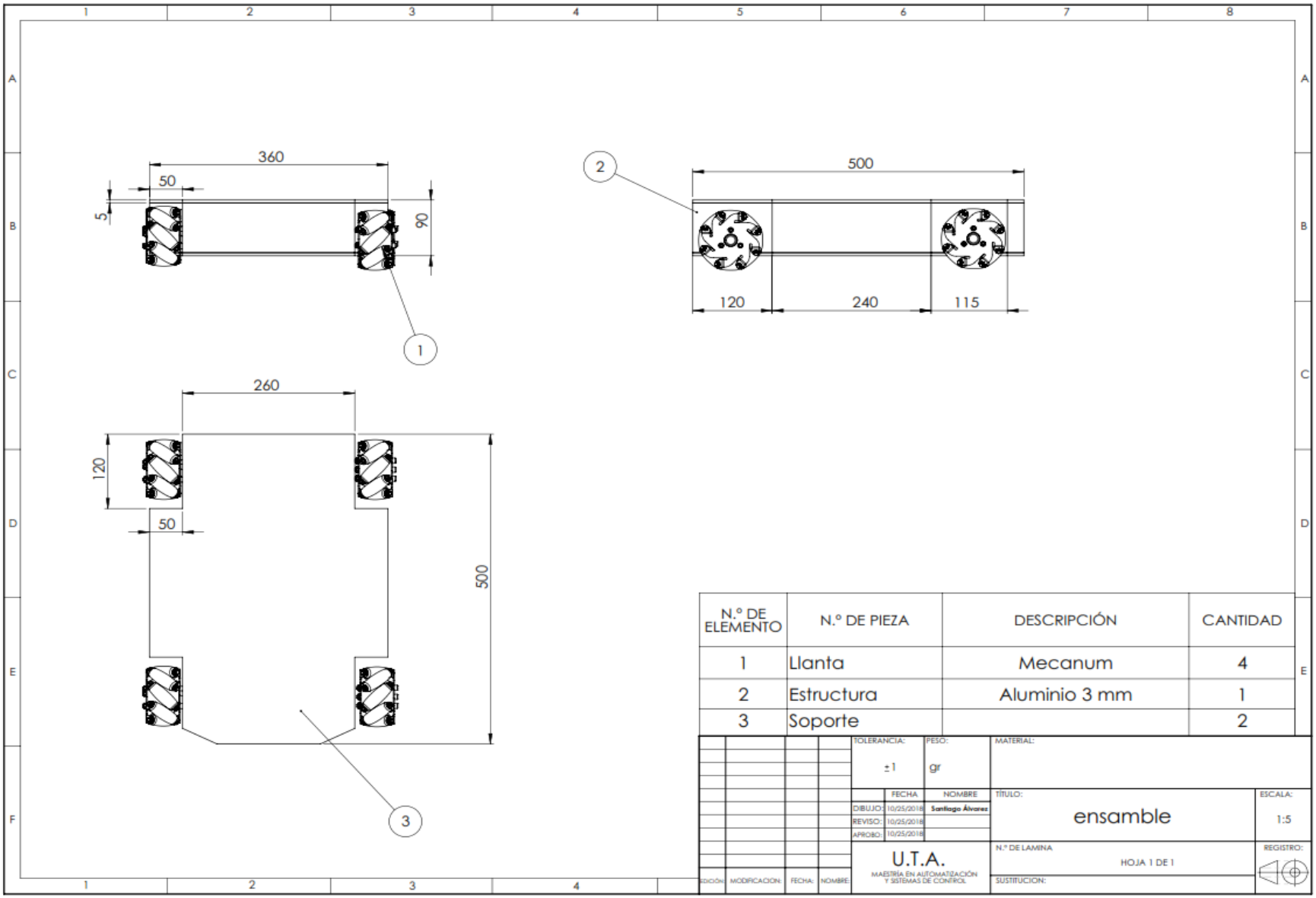
Yang, H., Fan, X., Shi, P., & Hua, C. (2015). Nonlinear Control for Tracking and Obstacle Avoidance of a Wheeled Mobile Robot With Nonholonomic Constraint. *IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY*, 1-6 (Accepted Article).

Yip, H. M., Ho, K. K., Chu, M. H., & Lai, K. W. (2014). Development of an Omnidirectional Mobile Robot using a RGB-D Sensor for Indoor Navigation. *The 4th Annual IEEE International Conference on Cyber Technology in Automation, Control and Intelligent Systems*, 162-167.

Anexos

Anexo 1. Planos de construcción del Robot Omnidireccional

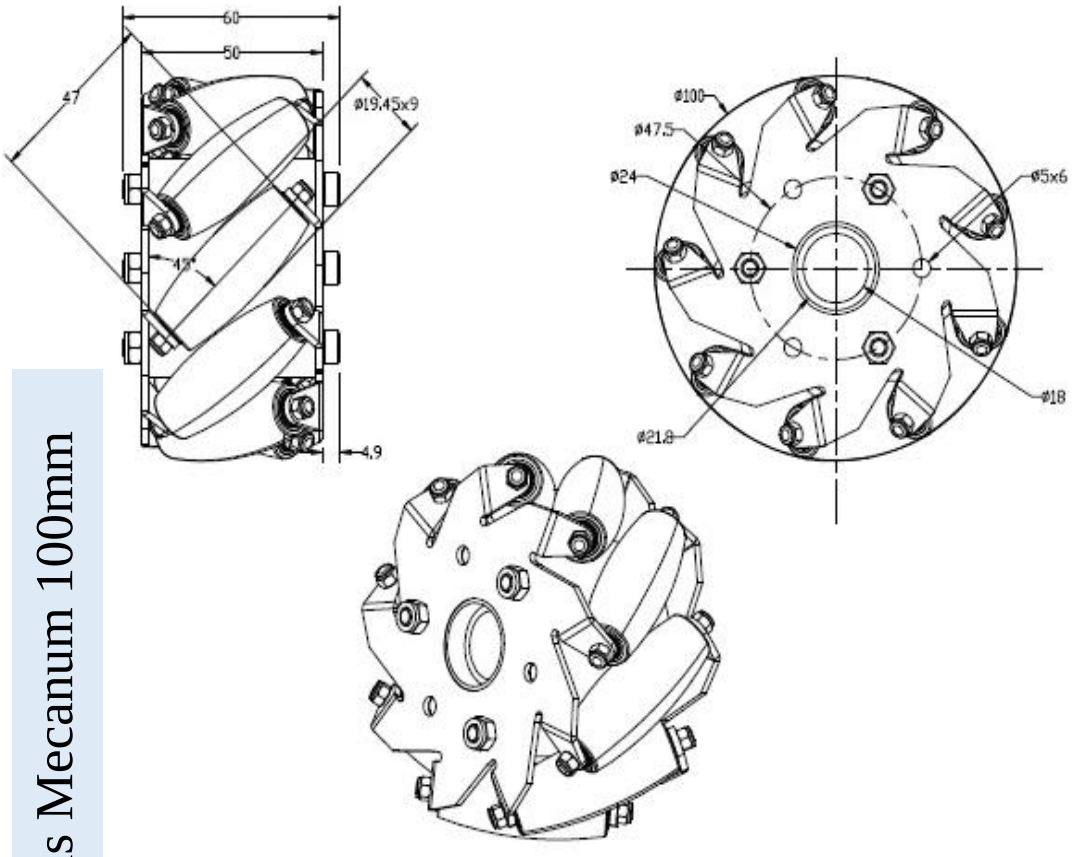


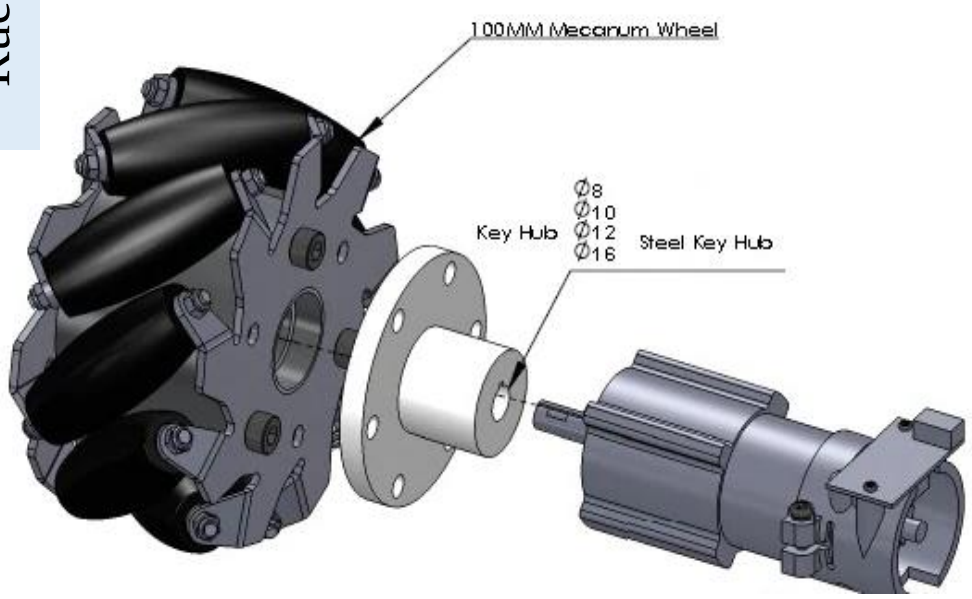


N.° DE ELEMENTO	N.° DE PIEZA	DESCRIPCIÓN	CANTIDAD
1	Llanta	Mecanum	4
2	Estructura	Aluminio 3 mm	1
3	Soporte		2

TOLERANCIA: ±1	PESO: gr	MATERIAL:	
	FECHA / NOMBRE	TÍTULO: ensamble	ESCALA: 1:5
DIBUJO: 10/25/2018 Santiago Álvarez			
REVISO: 10/25/2018			
APROBO: 10/25/2018			
U.T.A. MAESTRÍA EN AUTOMATIZACIÓN Y SISTEMAS DE CONTROL		N.° DE LAMINA HOJA 1 DE 1	REGISTRO:
EDICIÓN / MODIFICACIÓN / FECHA / NOMBRE		SUSTITUCION:	

Ruedas Mecanum 100mm





Especificaciones:	
Diámetro: 100mm	**Rodamiento de rodillos:** rodamiento de bolas
Ancho: 50 mm	**Material del cuerpo:** aleación de aluminio
Número de rodillos: 9	**Material del rodillo:** pp + PE
Número de rodamientos: 18	**Espaciador material:** nylon
Número de placas: 2	**Longitud del rodillo:** 47mm
Peso neto: 400g	**Capacidad de carga:** 15 kg

Anexo 2. Programación para verificar la eficiencia del control PID interno

Mediante el siguiente programa se verifica el funcionamiento del control PID interno del Robot

```
clear all;close all;clc
% conectar al robot omnidireccional
delete(instrfind({'Port'},{'COM19'}));
%selecionar el puerto del robot
S=serial('COM19');
% se configura la velocidad a 19200 Baudios
set(S,'Baudrate',19200);
% se configura bit de parada a uno
set(S,'StopBits',1);
% se configura que el dato es de 8 bits, debe estar entre 5 y 8
set(S,'DataBits',8);
% se configura sin paridad
set(S,'Parity','none');
% caracter con que finaliza el envio
set(S,'Terminator','CR/LF');
% 5 segundos de tiempo de espera
set(S,'Timeout',5);
fopen(S)
%imprimir en oantalla
disp('Robot Conectado')
pause(1)
%medidas del Robot
d1=0.165;
d2=0.185;
r=0.05; %radio
L=0.17;%distancia de ubicacion de Laser
%estableco el tiempo de muestreo
ts=.1 ; tfin=60; %500
t=[0:ts:tfin];
%definicion de velocidades de refencia
vf_ref=0.4*sin(0.8*t).*sin(0.04*t.*t)-0.6*cos(t/2);
vl_ref=-0.6*cos(1.5*t/2).*cos(1.5*t/6);
w_ref=0.85*sin(0.35*t).*cos(0.2*t);
%Inicio
for k=1:length(t)
% inicia medicipon de tiempo
tic
% calcular velocidades para las ruedas
V_ref=[vf_ref(k) vl_ref(k) w_ref(k)]';
%Matriz de transformacion de velocidades lineales a velocidades
angulares
%independientes
T_inv=[[1/r   -1/r    -(d1+d2)/r];...
      [1/r    1/r    -(d1+d2)/r];...
      [1/r   -1/r     (d1+d2)/r];...
      [1/r    1/r     (d1+d2)/r]];
%transformacion de de velocidades lineales a angualares
W_rf=T_inv*V_ref;
%separo las velocidades angulares para cada llanta
w1_ref(k)=W_rf(1);
w2_ref(k)=W_rf(2);
w3_ref(k)=W_rf(3);
w4_ref(k)=W_rf(4);
%aplicar al robot omnidireccional
```

```
Tx =
strcat('I',num2str(W_rf(1)),'B',num2str(W_rf(2)),'C',num2str(W_rf(3)),
'D',num2str(W_rf(4)),'F');
fprintf(S,Tx);
%esperrar el tiempo de muestreo
while toc < ts
end
fprintf(S,'G');
Rx=fscanf(S)
wm1(k) = str2num(Rx((strfind(Rx,'A')+1):(strfind(Rx,'B')-1)));
wm2(k) = str2num(Rx((strfind(Rx,'B')+1):(strfind(Rx,'C')-1)));
wm3(k) = str2num(Rx((strfind(Rx,'C')+1):(strfind(Rx,'D')-1)));
wm4(k) = str2num(Rx((strfind(Rx,'D')+1):(strfind(Rx,'F')-1)));
%calcular las velocidades lineales en base a las angulares
vf(k) =0.25*r*(wm1(k)+wm2(k)+wm3(k)+wm4(k));
vl(k) =0.25*r*(-wm1(k)+wm2(k)-wm3(k)+wm4(k));
w(k)=0.25*r*(-wm1(k)-wm2(k)+wm3(k)+wm4(k))/(d1+d2);
%calculo del error absoluto
error_vf(k)=abs(vf_ref(k)-vf(k));
error_vl(k)=abs(vl_ref(k)-vl(k));
error_w(k)=abs(w_ref(k)-w(k));
end
%calculo del error total para cada velocidad
err_vf_T=mean(error_vf);err_vf_min=min(error_vf);err_vf_max=max(error_
vf);
err_vl_T=mean(error_vl);err_vl_min=min(error_vl);err_vl_max=max(error_
vl);
err_w_T=mean(error_w);err_w_min=min(error_w);err_w_max=max(error_w);
%almacenar datos
save('eficiencia_PID');
fprintf(S,'I0B0C0D0F');
fclose(S);
disp("fin")
  %% Grafica de velocidades
  figure
  subplot(3,1,1);
  plot(vf_ref(1:600),'r'); hold on;
  plot(vf(1:600),'b'); hold on;
  legend('vf_r_e_f','vf')
  xlabel('Tiempo [s]'), ylabel('[m/s]'),grid on
  subplot(3,1,2);
  plot(vl_ref(1:600),'r'); hold on;
  plot(vl(1:600),'b'); hold on;
  legend('vl_r_e_f','vl');
  xlabel('Tiempo [s]'), ylabel('[m/s]'),grid on
  subplot(3,1,3);
  plot(w_ref(1:600),'r'); hold on;
  plot(w(1:600),'b'); hold on;
  legend('w_r_e_f','w');
  xlabel('Tiempo [s]'), ylabel('[rad/s]'),grid on
  %% Grafica de errores
  figure
  subplot(3,1,1);
  plot(error_vf(1:600),'r'); hold on;
  legend('vf _e_r_r_o_r')
  xlabel('Tiempo [s]'), ylabel('[m/s]'),grid on
  subplot(3,1,2);
  plot(error_vl(1:600),'r'); hold on;
  legend('vl _e_r_r_o_r');
  xlabel('Tiempo [s]'), ylabel('[m/s]'),grid on
  subplot(3,1,3);
```

```
plot(error_w(1:600),'r'); hold on;
legend('w _e_r_r_o_r','w');
xlabel('Tiempo [s]'), ylabel('[rad/s]'),grid on
```

Anexo 3. Programación para simular la cinemática del Robot

```
%limpiar variables
clear all;close all;clc
%medidas del Robot
d1=0.165;
d2=0.185;
r=0.05; %radio
L=0.17;%distancia de ubicación del Laser
%tiempo de muestreo
ts=.1 ; tfin=20; %500
t=[0:ts:tfin];
%POSICION INICIAL DEL ROBOT
x(1)=0; %posicion inicial robot en X
y(1)=0; %posicion inicial robot en Y
th(1)=0; %rotacion inicial robot en th
%Inicio
for k=1:length(t)
% inicia medición de tiempo
tic
%velocidades de referencia
if k<100
    vf(k)=0.65;          %en m/s
    vl(k)=-0.35;          %en m/s
    w(k)=0.75;          %en rad/s
else
    vf(k)=-0.55;          %en m/s
    vl(k)=-0.85;          %en m/s
    w(k)=-0.8;          %en rad/s
end
% calcular velocidades para las ruedas
V_ref=[vf(k) vl(k) w(k)]';
%Matriz de transformación de velocidades lineales a velocidades
angulares
%independientes
T_inv=[[1/r   -1/r    -(d1+d2)/r];...
     [1/r    1/r    -(d1+d2)/r];...
     [1/r   -1/r     (d1+d2)/r];...
     [1/r    1/r     (d1+d2)/r]];
%transformación de velocidades lineales a angulares
W_rf=T_inv*V_ref;
%separo las velocidades angulares para cada llanta
w1(k)=W_rf(1);
w2(k)=W_rf(2);
w3(k)=W_rf(3);
w4(k)=W_rf(4);
%calculo la velocidad del robot en el punto de interés en base a la
cinemática
xp(k) = vf(k)*cos(th(k))-vl(k)*sin(th(k)) -L*w(k)*sin(th(k));
yp(k) = vf(k)*sin(th(k))+vl(k)*cos(th(k))-L*w(k)*cos(th(k));
% calculo la nueva posición del robot en base a la euler
x(k+1) = x(k)+(ts)*xp(k);
y(k+1) = y(k)+(ts)*yp(k);
th(k+1) = th(k)+w(k)*(ts);
end
%fin del programa
disp("fin")
%%Graficar las variables y el movimiento ejecutado por el robot
paso=70; fig=figure;
set(fig,'position',[100 60 980 600]);
axis equal;  cameratoolbar
```

```
%definiciones ejes para la grafica
view(25,88)
axis([-2 2 -2.5 2.5 0 0.5]);grid on, hold on
%establezco la escala del robot para graficar
Robot_Dimension(1);hold on
%graficar el robot en condiciones iniciales
Rt=Omni_3D(x(1),y(1),th(1),0.05);hold on
H1 = plot(x(1),y(1),'m');
H2=plot(x(1),y(1),'b','LineWidth',2);   %es la trayectoria que hace
%graficar el movimiento estroboscópico del robot
H1 = plot(x,y,'r');
nn=1;
Rt=Omni_3D(x(nn),y(nn),th(nn),0.05);hold on
nn=50;
Rt=Omni_3D(x(nn),y(nn),th(nn),0.05);hold on
nn=100;
Rt=Omni_3D(x(nn),y(nn),th(nn),0.05);hold on
nn=150;
Rt=Omni_3D(x(nn),y(nn),th(nn),0.05);hold on
nn=150;
Rt=Omni_3D(x(end),y(end),th(end),0.05);hold on
%% graficar la evolucion de la posicion
figure
subplot(3,1,1);
plot(x);
legend('Evolucion posicion x')
xlabel('Tiempo [s]'), ylabel('[m]'),grid on
subplot(3,1,2)
plot(y,'r');
legend('Evolucion posicion y')
xlabel('Tiempo [s]'), ylabel('[m]'),grid on
subplot(3,1,3)
plot(th);
legend('Evolucion rotacion \theta')
xlabel('Tiempo [s]'), ylabel('[rad]'),grid on
%% graficar los valores de velocidades enviadas vs recibidas
figure
subplot(4,1,1);
plot(w1,'b'); hold on;
legend('w_1')
xlabel('Tiempo [s]'), ylabel('[rad/s]'),grid on
subplot(4,1,2);
plot(w2,'r'); hold on;
legend('w_2')
xlabel('Tiempo [s]'), ylabel('[rad/s]'),grid on
subplot(4,1,3);
plot(w3,'g'); hold on;
legend('w_3')
xlabel('Tiempo [s]'), ylabel('[rad/s]'),grid on
subplot(4,1,4);
plot(w4,'k'); hold on;
legend('w_4')
xlabel('Tiempo [s]'), ylabel('[rad/s]'),grid on
```

Anexo 4. Programación para validar experimentalmente la cinemática del Robot

```
%limpiar variables
clear all;close all;clc
% conectar al robot omnidireccional
delete(instrfind({'Port'},{'COM19'}));
%seleccionar el puerto del robot
S=serial('COM19');
% se configura la velocidad a 19200 Baudios
set(S,'Baudrate',19200);
% se configura bit de parada a uno
set(S,'StopBits',1);
% se configura que el dato es de 8 bits, debe estar entre 5 y 8
set(S,'DataBits',8);
% se configura sin paridad
set(S,'Parity','none');
% caracter con que finaliza el envío
set(S,'Terminator','CR/LF');
% 5 segundos de tiempo de espera
set(S,'Timeout',5);
fopen(S)
%imprimir en pantalla
disp('Robot Conectado')
pause(1)
%medidas del Robot
d1=0.165;
d2=0.185;
r=0.05; %radio
L=0.17; %distancia de ubicación de Laser
%estableco el tiempo de muestreo
ts=.1 ; tfin=60; %500
t=[0:ts:tfin];
%POSICION INICIAL DEL ROBOT
x(1)=0; %posición inicial robot en X
y(1)=0; %posición inicial robot en Y
th(1)=0; %rotation inicial robot en th
%Inicio
for k=1:length(t)
% inicia medición de tiempo
tic
%velocidades de referencia
if k<200
    vf_ref(k)=0.65;          %en m/s
    vl_ref(k)=-0.35;          %en m/s
    w_ref(k)=0.75;          %en rad/s
else
    vf_ref(k)=-0.55;          %en m/s
    vl_ref(k)=-0.85;          %en m/s
    w_ref(k)=-0.8;          %en rad/s
end
% calcular velocidades para las ruedas
V_ref=[vf_ref(k) vl_ref(k) w_ref(k)]';
%Matriz de transformación de velocidades lineales a velocidades
angulares
%independientes
T_inv=[[1/r   -1/r    -(d1+d2)/r];...
     [1/r    1/r    -(d1+d2)/r];...
     [1/r   -1/r     (d1+d2)/r];...
     [1/r    1/r     (d1+d2)/r]];
%transformación de de velocidades lineales a angulares
W_rf=T_inv*V_ref;
```

```
%separo las velocidades angulares para cada llanta
w1_ref(k)=W_rf(1);
w2_ref(k)=W_rf(2);
w3_ref(k)=W_rf(3);
w4_ref(k)=W_rf(4);
%concatenar valores para enviar al robot
Tx=strcat('I',num2str(W_rf(1)),'B',num2str(W_rf(2)),'C',num2str(W_rf(3
)),'D',num2str(W_rf(4)),'F');
%imprimo en el puerto las velocidades para el robot
fprintf(S,Tx);
pause(0.065);
%Lectura de los datos del robot
fprintf(S,'G');
Rx=fscanf(S)
%separa velocidades recibidas del robot
w1(k) = str2num(Rx((strfind(Rx,'A')+1):(strfind(Rx,'B')-1)));
w2(k) = str2num(Rx((strfind(Rx,'B')+1):(strfind(Rx,'C')-1)));
w3(k) = str2num(Rx((strfind(Rx,'C')+1):(strfind(Rx,'D')-1)));
w4(k) = str2num(Rx((strfind(Rx,'D')+1):(strfind(Rx,'F')-1)));
%cálculo de la velocidad frontal actual
vf(k) =0.25*r*(w1(k)+w2(k)+w3(k)+w4(k));
%cálculo de la velocidad lateral actual
vl(k) =0.25*r*(-w1(k)+w2(k)-w3(k)+w4(k));
%cálculo de la velocidad angular actual
w(k)=0.25*r*(-w1(k)-w2(k)+w3(k)+w4(k))/(d1+d2);
%esperar que transcurra el tiempo de muestro
while toc < ts
end
%calculo la velocidad del robot en el punto de interes en base a la
cinematica
xp(k) = vf(k)*cos(th(k))-vl(k)*sin(th(k)) -L*w(k)*sin(th(k));
yp(k) = vf(k)*sin(th(k))+vl(k)*cos(th(k))-L*w(k)*cos(th(k));
% calculo la nueva posición del robot en base a la euler
x(k+1) = x(k)+(toc)*xp(k);
y(k+1) = y(k)+(toc)*yp(k);
th(k+1) = th(k)+w(k)*(toc);

end
%fin del programa
%detener el robot
fprintf(S,'I0B0C0D0F');
fclose(S);
disp("fin")
%%Graficar las variables y el movimiento ejecutado por el robot
paso=70; fig=figure;
set(fig,'position',[100 60 980 600]);
axis equal;  cameratoolbar
%definicionde ejes para la grafica
view(25,88)
axis([-2 2 -2.5 2.5 0 0.5]);grid on, hold on
%establecer la escala del robot para graficar
Robot_Dimension(1);hold on
%graficar el robot en condiciones iniciales
Rt=Omni_3D(x(1),y(1),th(1),0.05);hold on
H1 = plot(x(1),y(1),'m');
H2=plot(x(1),y(1),'b','LineWidth',2);   %es la trayectoria que hace
%graficar el movimiento estroboscopico del robot
H1 = plot(x,y,'r');
nn=1;
Rt=Omni_3D(x(nn),y(nn),th(nn),0.05);hold on
nn=50;
```

```
Rt=Omni_3D(x(nn),y(nn),th(nn),0.05);hold on
nn=100;
Rt=Omni_3D(x(nn),y(nn),th(nn),0.05);hold on
nn=150;
Rt=Omni_3D(x(nn),y(nn),th(nn),0.05);hold on
nn=150;
Rt=Omni_3D(x(end),y(end),th(end),0.05);hold on
%% graficar la evolucion de la posicion
figure
subplot(3,1,1);
plot(x);
legend('Evolucion posicion x')
xlabel('Tiempo [s]'), ylabel('[m]'),grid on
subplot(3,1,2)
plot(y);
legend('Evolucion posicion y')
xlabel('Tiempo [s]'), ylabel('[m]'),grid on
subplot(3,1,3)
plot(th);
legend('Evolucion rotacion \theta')
xlabel('Tiempo [s]'), ylabel('[rad]'),grid on
%% graficar los valores de velocidades enviadas vs recibidas
figure
subplot(4,1,1);
plot(w1_ref,'r'); hold on;
plot(w1,'b'); hold on;
legend('w1_r_e_f','w1')
xlabel('Tiempo [s]'), ylabel('[rad/s]'),grid on
subplot(4,1,2);
plot(w2_ref,'r'); hold on;
plot(w2,'b'); hold on;
legend('w2_r_e_f','w2')
xlabel('Tiempo [s]'), ylabel('[rad/s]'),grid on
subplot(4,1,3);
plot(w3_ref,'r'); hold on;
plot(w3,'b'); hold on;
legend('w3_r_e_f','w3')
xlabel('Tiempo [s]'), ylabel('[rad/s]'),grid on
subplot(4,1,4);
plot(w4_ref,'r'); hold on;
plot(w4,'b'); hold on;
legend('w4_r_e_f','w4')
xlabel('Tiempo [s]'), ylabel('[rad/s]'),grid on
%%graficar velocidades lineales deseadas y actuales
figure
subplot(3,1,1);
plot(vf_ref,'r'); hold on;
plot(vf,'b'); hold on;
legend('vf_r_e_f','vf')
xlabel('Tiempo [s]'), ylabel('[m/s]'),grid on
subplot(3,1,2);
plot(vl_ref,'r'); hold on;
plot(vl,'b'); hold on;
legend('vl_r_e_f','vl');
xlabel('Tiempo [s]'), ylabel('[m/s]'),grid on
subplot(3,1,3);
plot(w_ref,'r'); hold on;
plot(w,'b'); hold on;
legend('w_r_e_f','w');
xlabel('Tiempo [s]'), ylabel('[rad/s]'),grid on
```

Anexo 5. Programación para la simulación del controlador para varias trayectorias

Para realizar la simulación del controlador desarrollado se realiza varias pruebas, mediante el presente código se realiza la simulación de múltiples trayectorias, seleccionando cual es la que se va a realizar.

```
%limpiar variables
clear all;close all;clc
%medidas del Robot
d1=0.165;
d2=0.185;
r=0.05; %radio
L=0.17;%distancia de ubicacion de Laser
%%DATOS PARA TRAYECTORIA 1
%tiempo deejecucion y tiempo de muestro
ts=.1 ; tfin=50;
t=[0:ts:tfin];
%Condiciones Iniciales para Trayectoria 1
x(1)=0; %posicion inicial robot en X
y(1)=0; %posicion inicial robot en Y
th(1)=pi/4; %rotacion inicial robot en th
% trayectoria de un circulo
radio=2;
xd = radio*cos(0.2*t);       xd_p = -radio*0.2*sin(0.2*t);      xd_2p
= -radio*0.2*0.2*cos(0.2*t);
yd = radio*sin(0.2*t);       yd_p =  radio*0.2*cos(0.2*t);      yd_2p
= -radio*0.2*0.2*sin(0.2*t);
%angulo deseada durante la trayectoria
thd= atan2(yd_p,xd_p);
thd_p = (1./((yd_p./xd_p).^2+1)).*((yd_2p.*xd_p-
yd_p.*xd_2p)./xd_p.^2); % velocidad deseada rotacion
thd_p(1) = 0;
%%DATOS PARA TRAYECTORIA 2
%tiempo deejecucion y tiempo de muestro
% ts=.1 ; tfin=50;
% t=[0:ts:tfin];
% %POSICION INICIAL DEL ROBOT
% x(1)=-4; %posicion inicial robot en X
% y(1)=-3; %posicion inicial robot en Y
% th(1)=0; %rotacion inicial robot en th
% xd = 0.8*cos(0.27*t)+0.13*t-2;
% xd_p = -0.8*0.27*sin(0.27*t)+0.13*ones(1,length(t));
% xd_2p = -0.8*0.27*0.27*cos(0.27*t);yd = 0.3*t-4;
% yd_p =0.3*ones(1,length(t));yd_2p = 0;
%%angulo deseado durante la trayectoria
%thd= atan2(yd_p,xd_p);
%thd_p = (1./((yd_p./xd_p).^2+1)).*((yd_2p.*xd_p-
yd_p.*xd_2p)./xd_p.^2); % velocidad deseada rotacion
for k=1:length(xd)
%calculo de errores
error_x(k)=xd(k)-x(k);
error_y(k)=yd(k)-y(k);
error_th(k)=Ang180(thd(k)-th(k));
%vector de errores
Errores=[error_x(k) error_y(k) error_th(k)];
%vectorde velocidades dependiendo de la trayectoria
Vref=[xd_p(k) yd_p(k) thd_p(k)];
%Matriz de rotacion homogenea
M = [[cos(th(k)) -sin(th(k))  -L*sin(th(k))  ];...
```

```
    [sin(th(k))  cos(th(k))  -L*cos(th(k))  ];...
    [0             0          1  ]];
kp=0.5;%ganancia de control Pid
Vpid=kp*tanh([error_x(k) error_y(k) error_th(k)]);
%mas ganancia para rotacion
Vpid(3)=Vpid(3)*1.5;%ganancia editable
%ley de control para seguimiento de trayectoria
Vel=linsolve(M,[(Vref'+Vpid')]);
%Separo las velocidades leidas del robot
vf(k)=Vel(1);%en m/s
vl(k)=Vel(2);%en m/s
w(k)=Vel(3);%en rad/s
% aplicar las velocidades al robot y obtengo la velocidad de cada eje
del robot con su cinematica
xp(k) = vf(k)*cos(th(k))-vl(k)*sin(th(k)) -L*w(k)*sin(th(k));
yp(k) = vf(k)*sin(th(k))+vl(k)*cos(th(k))-L*w(k)*cos(th(k));
%Calculo de la nueva posicion del robot con Euler
x(k+1) = x(k)+(ts)*xp(k);
y(k+1) = y(k)+(ts)*yp(k);
th(k+1) = th(k)+w(k)*(ts);
end
disp('fin')
%%
%%Graficar las varibles y el movimiento del robot
paso=50; fig=figure;
set(fig,'position',[100 60 980 600]);
axis equal;  cameratoolbar
%definicionde ejes para la grafica
view(25,30)
% axis Trayectoria 1
axis([-4 4 -4 4 0 0.5]);grid on, hold on
% axis Trayectoria 2
% axis([-5 7 -5 12 0 0.5]);grid on, hold on
Robot_Dimension(1);hold on
Omni_3D(0,0,0,0.2);hold on
plot(x(1),y(1),'m');
plot(x(1),y(1),'b','LineWidth',2);
plot(xd,yd,'r','LineWidth',2);
%graficar movimiento estroboscopico para Trayectoiria 1
k=1;Omni_3D(x(k),y(k),th(k),0.2);hold on
k=80;Omni_3D(x(k),y(k),th(k),0.2);hold on
k=200;Omni_3D(x(k),y(k),th(k),0.2);hold on
k=250;Omni_3D(x(k),y(k),th(k),0.2);hold on
k=350;Omni_3D(x(k),y(k),th(k),0.2);hold on
k=450;Omni_3D(x(k),y(k),th(k),0.2);hold on
k=501;Omni_3D(x(k),y(k),th(k),0.2);hold on
%graficar movimiento estroboscopico para Trayectoiria 2
% k=1;Rt=Omni_3D(x(k),y(k),th(k),0.2);hold on
% k=100;Omni_3D(x(k),y(k),th(k),0.2);hold on
% k=200;Omni_3D(x(k),y(k),th(k),0.2);hold on
% k=300;Omni_3D(x(k),y(k),th(k),0.2);hold on
% k=400;Omni_3D(x(k),y(k),th(k),0.2);hold on
% k=501;Omni_3D(x(k),y(k),th(k),0.2);hold on


%Graficar los resultados
figure(2);
%graficar la velocidad lineales del robot
subplot(2,2,1);
plot(vf(1:length(t)-1),'b'); hold on;
plot(vl(1:length(t)-1),'r'); hold on;
```

```
legend('v_f','v_l')
xlabel('Tiempo [s]'), ylabel('[m/s]')
title('Acciones de Control V_f, V_l'); grid on
%graficar la velocidad angular del robot
subplot(2,2,2);
plot(w(1:length(t)-1),'b'); hold on;
xlabel('Tiempo [s]'), ylabel('[rad/s]')
legend('\omega')
title('Accion de Control \omega'); grid on
%graficar errores de control en x y
subplot(2,2,3);
plot(error_x(1:length(t)-1),'b'); hold on;
plot(error_y(1:length(t)-1),'r'); hold on;
xlabel('Tiempo [s]'), ylabel('[m]')
legend('Error en x','Error en y')
title('Errores de Control'); grid on
%graficar error decontrol de rotacion
subplot(2,2,4);
plot(error_th(1:length(t)-1),'r'); hold on;
xlabel('Tiempo [s]'), ylabel('[rad]')
legend('Error \theta')
title('Error de Control'); grid on
```

Anexo 7. Programación para lectura de datos del Sensor RPLidar

Con el siguiente código se verifica el funcionamiento para la opctencion de datos del sensor Laser

```
%limpiar variables
clc, close all, clear all;
%cargar libreria
loadlibrary('./radarDLL.dll','./codigo.h');
%enviar puerto com del LIDAR
calllib('radarDLL','start','com11');
pause(2)
%estado del Laser
state = calllib('radarDLL','getStatus');
%esperar a que el Laser este trabajando
while (state ~= 0)
    state = calllib('radarDLL','getStatus');
    pause(0.5)
end
%pausa
pause(2)
%verificar si hay errores en el Sensor
errors = calllib('radarDLL','getErrors');
%pausa de 5 segundos para empezar a obtener datos
disp('En 5 segundos empieza a obtener valores');
pause(5)
%inicialización de variable
n = 1;
figure(1)
while n < 1000
    %empezar a medir tiempo
    tic
    %la lectura viene de la forma
    %angulo-distancia-calidad:angulo-distancia-calidad:.....
    values = calllib('radarDLL','getValues');
    %incremento variable de obtencion de datos
    n = n+1;
    %separar los datos leídos del sensor
    Datos=textscan(values,'%f','Delimiter',':');
    %transformar a tipo de datos dobles los valores obtenidos
    D=double(Datos{1});
    %inicializo variable para analizar datos obtenidos por el sensor
    k=1;
    d_obs=0;
    beta=0;
for i=1:3:length(D)
    %transformar a metros la distancia de los puntos detectados
    aux=D(i+1)/1000;
    %si distancia es menor o igual a 0.2m y la calidad mayor al
    %40%p guardar los valores caso contrario no guardarlos
    if (aux<0.2) && (D(i+2)>40) && (D(i)<=90 ||(D(i)<=360 &&
D(i)>=270))
    [aux D(i)];
    d_obs(k)=aux;
    %transformar a radianes el ángulo
    beta(k)=-D(i)*pi/180;
    k=k+1;
    end
end
%inicialización de variables para almacenar los puntos
```

```
x_obs=0;
y_obs=0;
for k=1:length(d_obs)
    %calculo para la coordenada en 'x'
    x_obs(k)=d_obs(k)*cos(beta(k));
    %calculo para la coordenada en 'y'
    y_obs(k)=d_obs(k)*sin(beta(k));
end
axis([-1 1 -1 1]);
%ploteo los puntos de un obstáculo detectado dentro del rango
especificado
H=plot(x_obs,y_obs,'.');hold on
pause(0.01)
delete(H)
%pausa hasta que el tiempo de muestreo se cumple en 100ms
while toc<0.1
end

end
%detiene el sensor
calllib('radarDLL','stop');
pause(5)
%descarga la libreria del laser
unloadlibrary radarDLL
```

Anexo 8. Programación para la experimentación del control con evasión de obstáculos.

Mediante esta programación el robot ejecuta una trayectoria predefinida en la que se encuentren obstáculos y el robot de forma autónoma evite cada uno de ellos.

```
%limpiar variables
clear all;close all;clc
% conectar al robot omnidireccional
delete(instrfind({'Port'},{'COM19'}));
%selecionar el puerto del robot
S=serial('COM19');
% se configura la velocidad a 19200 Baudios
set(S,'Baudrate',19200);
% se configura bit de parada a uno
set(S,'StopBits',1);
% se configura que el dato es de 8 bits, debe estar entre 5 y 8
set(S,'DataBits',8);
% se configura sin paridad
set(S,'Parity','none');
%caracter con que finaliza el envio
set(S,'Terminator','CR/LF');
%5 segundos de tiempo de espera
set(S,'Timeout',5);
fopen(S)
%imprimir en pantalla
disp('Robot Conectado')
%COnenctar laser LIDAR
%cargar libreria
loadlibrary('./radarDLL.dll','./codigo.h');
%enviar puerto com del LIDAR
calllib('radarDLL','start','com11');
pause(2)
%estado del Laser
state = calllib('radarDLL','getStatus');
%esperar a que el Laser este trabajando
while (state ~= 0)
    state = calllib('radarDLL','getStatus');
    pause(0.5)
end
%pausa
pause(2)
%verificar si hay errores en el Sensor
errors = calllib('radarDLL','getErrors');
%pausa de 5 segundos para empezar a obtener datos
disp('En 5 segundos empieza a obtener valores');
pause(5)


%medidas del Robot
d1=0.165;
d2=0.185;
r=0.05; %radio
L=0.17;%distancia de ubicacion de Laser
%estableco el tiempo de muestreo
ts=.1 ; tfin=50; %500
t=[0:ts:tfin];

%%DATOS PARA TRAYECTORIA 1
```

```
%POSICION INICIAL DEL ROBOT
% x(1)=0; %posicion inicial robot en X
% y(1)=0; %posicion inicial robot en Y
% th(1)=pi/4; %rotacion inicial robot en th
%%trayectoria 1 Circulo
% radio=2;
% xd = radio*cos(0.2*t);       xd_p = -radio*0.2*sin(0.2*t);
xd_2p = -radio*0.2*0.2*cos(0.2*t);
% yd = radio*sin(0.2*t);       yd_p =  radio*0.2*cos(0.2*t);
yd_2p = -radio*0.2*0.2*sin(0.2*t);
% %angulo deseada durante la trayectoria
% thd= atan2(yd_p,xd_p);
% thd_p = (1./((yd_p./xd_p).^2+1)).*((yd_2p.*xd_p-
yd_p.*xd_2p)./xd_p.^2); % velocidad deseada rotacion
% thd_p(1) = 0;
%%DATOS PARA TRAYECTORIA 2
%POSICION INICIAL DEL ROBOT
x(1)=-4; %posicion inicial robot en X
y(1)=-3; %posicion inicial robot en Y
th(1)=0; %rotacion inicial robot en th
% trayectoria linea senoidal
xd = 0.8*cos(0.27*t)+0.13*t-2;xd_p = -
0.8*0.27*sin(0.27*t)+0.13*ones(1,length(t));xd_2p = -
0.8*0.27*0.27*cos(0.27*t);
yd = 0.3*t-4;yd_p =  0.3*ones(1,length(t));yd_2p = 0;
%%angulo deseada durante la trayectoria
thd= atan2(yd_p,xd_p);
thd_p = (1./((yd_p./xd_p).^2+1)).*((yd_2p.*xd_p-
yd_p.*xd_2p)./xd_p.^2); % velocidad deseada rotacion
%%INICIO CONSTANTES PARA EVASION DE OBSTACULOS%%
% parametros para evasion
 dmin=0.3;%es la distancia mínima
 dmax=0.5;%distancia máximA
 vf_max=0.7;%velocidad frontal maxima
 vl_max=0.7;%velocidad lateral maxima
 p=0.9;%constante para promedio de las velocidades máximas y actuales
 kf=0.5;%representa una constante para radio de cobertura
 vf(1)=0;%velocidad inicial del robot
 vl(1)=0;;%velocidad inicial del robot

for k=1:length(xd)
tic
%%LECTURA DE DATOS DEL LASER LIDAR
    %la lectura viene de la forma
    %angulo-distancia-calidad:angulo-distancia-calidad:.....
    values = calllib('radarDLL','getValues');
    %incremento variable de obtencion de datos
    n = n+1;
    %separar los datos leidos del sensor
    Datos=textscan(values,'%f','Delimiter',':');
    %transformar a tipo de datos dobles los valores obtenidos
    D=double(Datos{1});
    %inicializo variabl para analizar todos los datos obtenidos por el
sensor
    k=1;
    d_obs=0;
    beta=0;
for i=1:3:length(D)
    %transformo a metros la distancia de los puntos detectados
    aux=D(i+1)/1000;
```

```
    %si la distancia en menos o igual a 0.2m y la calidad de medida
semayor al
    %40%p guardar los valorescaso contrario no guardarlos
    if (aux<0.2) && (D(i+2)>40) && (D(i)<=90 ||(D(i)<=360 &&
D(i)>=270))
        [aux D(i)];
        d_obs(k)=aux;
        %transformar a radianes el angulo
        beta(k)=-D(i)*pi/180;
        k=k+1;
    end
end
%inicializacion de variables para almacenar los puntos
x_obs=0;
y_obs=0;
for k=1:length(d_obs)
    %calculo para la coordenada en 'x'
    x_obs(k)=d_obs(k)*cos(beta(k));
    %calculo para la coordenada en 'y'
    y_obs(k)=d_obs(k)*sin(beta(k));
end

%calculo de niu, pesalas velocidades para elradio de cobertura
n(k)=p*(ul(k)+um(k))/(umax+umax2);
% calculo de la fuerzade repulsion
    if (d_obs(k)>dmax)
        F(k)=0;
    else
        F(k)=kf*n(k)/abs(dmin-d_obs(k));
    end
%definicion para pesar la fuerza de rotacion
   if (F(k)==0)
        epsilon(k)=1;
    else
        epsilon(k)=0.3;
   end
%   constante para definir sentido de evasion
      if (beta(k)>90)
          sb(k)=-1;
      else
          sb(k)=1;
      end
%angulo de rotacion
sigma(k)=sb(k)*(abs(F(k)));
%matriz de rotacion para el robot
rot=[[cos(sigma(k)) -sin(sigma(k))];...
       [sin(sigma(k))  cos(sigma(k))]];
%definimos la velocidades del robot de acuerdo
%a la trauectoria modificada con la nueva rotacion
velo=epsilon(k)*rot*[xd_p(k);yd_p(k)];
%velx, velx, velth
Vref=[velo(1) velo(2) 0]';
%angulo de rotacion nuevo
thd2(k)=atan2(velo(2),velo(1));

%calculo de errores
error_x(k)=xd(k)-x(k);
error_y(k)=yd(k)-y(k);
error_th(k)=Ang180(thd2(k)-th(k));
%vector de errores nuevos rotados para evitar obstaculo
%moifcacion de errroes
```

```
error_mod=epsilon(k)*rot*[x_err(k);y_err(k)];
x_err2(k)=error_mod(1);
y_err2(k)=error_mod(2);
Errores=[x_err2(k) y_err2(k) 1.2*error_th(k)]';

%Matriz de rotacion homogenea
M = [[cos(th(k)) -sin(th(k))  -L*sin(th(k))  ];...
    [sin(th(k))  cos(th(k))  -L*cos(th(k))  ];...
    [0             0          1  ]];
kp=0.7;%ganancia de control Pid
Vpid=kp*tanh(0.8*Errores);%control proporcional
%mas ganancia para rotacion
Vpid(3)=Vpid(3)*2.5;%mas ganancia a la rotacion
%ley de control paraseguimiento de trayectoria
velo=linsolve(M,[(Vref'+Vpid')]);
% Separar las velocidad de maniobrabilidad del robot
vf_ref(k)=velo(1);
vl_ref(k)=velo(2);
w_ref(k)=velo(3);
%vector de velcidadesde refecnia obtenidas por la ley decontrol
Vref=[vf_ref(k);vl_ref(k);w_ref(k)];
%envio velodicades al robot funcion Robot(com,vector) y lectura en Vel
Vel=Robot(S,Vref);
pause(0.002);
while toc < ts        %espero que haya pasado tiempo de muestreo para
enviar las acciones de control
end
toc
%Separo las velocidades leidas del robot
vf(k)=Vel(1);%en m/s
vl(k)=Vel(2);%en m/s
w(k)=Vel(3);%en rad/s
% aplicar las velocidades al robot y optengo la velocidad de cada eje
del robot con su cinematica
xp(k) = vf(k)*cos(th(k))-vl(k)*sin(th(k)) -L*w(k)*sin(th(k));
yp(k) = vf(k)*sin(th(k))+vl(k)*cos(th(k))-L*w(k)*cos(th(k));
%Calculo de la nueva posicion del robot con Euler
x(k+1) = x(k)+(ts)*xp(k);
y(k+1) = y(k)+(ts)*yp(k);
th(k+1) = th(k)+w(k)*(ts);
end
%poner en estop el carro
fprintf(S,'I0B0C0D0F');
fclose(S);
%detiene el sensor
calllib('radarDLL','stop');
pause(5)
%descarga la libreria del radar
unloadlibrary radarDLL
disp('fin')
%almacenar datos
% save('control_trayectoria_PID_experimental');
%%
%%Graficar las varibles y el movimeitno ejecutado por el robto
paso=50; fig=figure;
set(fig,'position',[100 60 980 600]);
axis equal;  cameratoolbar
%definicionde ejes para la grafica
view(25,30)
% axis Trayectoria 1
% axis([-4 4 -4 4 0 0.5]);grid on, hold on
```

```
% axis Trayectoria 2
axis([-5 7 -5 12 0 0.5]);grid on, hold on
Robot_Dimension(1);hold on
Rt=Omni_3D(0,0,0,0.2);hold on
H3 = plot(x(1),y(1),'m');
H4=plot(x(1),y(1),'b','LineWidth',2);   %es la trayectoria que hace
H1=plot(xd,yd,'r','LineWidth',2);
for k=1:paso:length(xd)
    drawnow
    delete(Rt);
    delete(H3);
    delete(H4);
    Rt=Omni_3D(x(k),y(k),th(k),0.2);hold on
    H4=plot(x(1:k),y(1:k),'b','LineWidth',1.5);   %es la trayectoria
que hace
end
%graficar movimiento estroboscopico para Trayectoiria 1
% k=1;Rt=Omni_3D(x(k),y(k),th(k),0.2);hold on
% k=100;Rt=Omni_3D(x(k),y(k),th(k),0.2);hold on
% k=250;Rt=Omni_3D(x(k),y(k),th(k),0.2);hold on
% k=300;Rt=Omni_3D(x(k),y(k),th(k),0.2);hold on
% k=350;Rt=Omni_3D(x(k),y(k),th(k),0.2);hold on
% k=450;Rt=Omni_3D(x(k),y(k),th(k),0.2);hold on
%graficar movimiento estroboscopico para Trayectoiria 2
k=1;Rt=Omni_3D(x(k),y(k),th(k),0.2);hold on
k=100;Rt=Omni_3D(x(k),y(k),th(k),0.2);hold on
k=200;Rt=Omni_3D(x(k),y(k),th(k),0.2);hold on
k=300;Rt=Omni_3D(x(k),y(k),th(k),0.2);hold on
k=400;Rt=Omni_3D(x(k),y(k),th(k),0.2);hold on
k=501;Rt=Omni_3D(x(k),y(k),th(k),0.2);hold on
%%
%Graficar los resultados
figure(2);
%graficar errores de control en x y
subplot(1,2,1);
plot(error_x(1:500),'b'); hold on;
plot(error_y(1:500),'r'); hold on;
xlabel('Tiempo [s]'), ylabel('[m]')
legend('x_E_r_r_o_r','y_E_r_r_o_r')
title('Errors de Control'); grid
%graficar error decontrol de rotacion
subplot(1,2,2);
plot(error_th(1:500),'r'); hold on;
xlabel('Tiempo [s]'), ylabel('[rad]')
legend('\theta_e_r_r_o_r')
title('Errores de Control'); grid
%%
figure
subplot(2,2,1);
plot(vf_ref(1:500),'b'); hold on;
plot(vl_ref(1:500),'r'); grid on
legend('v_f _r_e_f','v_l_r_e_f')
xlabel('Tiempo[s]'), ylabel('[m/s]')
title('Velocidades de Referecnica'); grid on
grid on
subplot(2,2,2);
plot(vf(1:500),'b'); hold on;
plot(vl(1:500),'r'); grid on
legend('v_f','v_l')
xlabel('Tiempo[s]'), ylabel('[m/s]')
title('Velocidades del Robot'); grid on
```

```
subplot(2,2,3);
plot(w_ref(1:500),'g'); grid on
legend('\omega _r_e_f')
xlabel('Tiempo[s]'), ylabel('[rad/s]')
title('Velocidad de Referecnica'); grid on
subplot(2,2,4);
plot(w(1:500),'g'); grid on
legend('\omega')
xlabel('Tiempo[s]'), ylabel('[rad/s]')
title('Velocidad del Robot'); grid on
save('experimento fin1')
```

Función: `Robot(com,vector)`

```
function [Vel] = Robot(PorT,Vo)
%medidas del Robot
d1=0.165;
d2=0.185;
R=0.1; %radio

%Matriz de transformacion de velocidades lineales a velocidades
angulares
%independientes
Tft= [[1/R   -1/R    -(d1+d2)/R];...
     [1/R    1/R    -(d1+d2)/R];...
     [1/R   -1/R     (d1+d2)/R];...
     [1/R    1/R     (d1+d2)/R]];
W=Tft*Vo;
w1=num2str(W(1));
w2=num2str(W(2));
w3=num2str(W(3));
w4=num2str(W(4));
%creo una string para enviar
Tx = strcat('I',w1,'B',w2,'C',w3,'D',w4,'F');
fprintf(PorT,Tx);
% pause(0.001);
pause(0.065);
fprintf(PorT,'G');
%leo las velocidades del robot
Rx=fscanf(PorT);
%separo y convierto las velocidades leídas de robot
wm1 = str2num(Rx((strfind(Rx,'A')+1):(strfind(Rx,'B')-1)));
wm2 = str2num(Rx((strfind(Rx,'B')+1):(strfind(Rx,'C')-1)));
wm3 = str2num(Rx((strfind(Rx,'C')+1):(strfind(Rx,'D')-1)));
wm4 = str2num(Rx((strfind(Rx,'D')+1):(strfind(Rx,'F')-1)));
vf =0.25*R*(wm1+wm2+wm3+wm4);
vl =0.25*R*(-wm1+wm2-wm3+wm4);
w=0.25*R*(-wm1-wm2+wm3+wm4)/(d1+d2);
Vel=[vf vl w];
end
```

Anexo 9. Programación para la interfaz de usuario, para controlar el robot omnidireccional

Con la Interfaz de usuario se prende la fácil manipulación del robot, cuenta con varias opciones descritas en el presente trabajo.

```
%limpio variables
close all; clc; clear;warning off;
%variables de inicio
Control_Panel=1;
trayectoria=1;
Stop=0; %boton de stop simulacion
Pause=0; %boton de Pause simulacion
Run=0; %boton de run Manual
k=1;vf(1)=0;vl(1)=0;w(1)=0;
% Colores
colorGris= [250/255 240/255 230/255];
% Figura
f = figure('Name','CONTROL OMNIDIRECCIONAL','Position',[50 80 1200
580]','NumberTitle','off');
set(f, 'ToolBar', 'none', 'MenuBar', 'none','Resize','off');
axes('Position', [0 0 1 1]); axis off;
im=imread('imj.png','png');
image(im),axis off;
% Panel de mensajes inferior
panelMensajes = uicontrol('Style','text','String','Inertaz de Control
Robot Omnidireccional','Position',[0 0 1200 20],'FontWeight',
'bold','BackgroundColor',[100/255 240/255 201/255]);
%% Definicion de Botones Principales Para los Paneles
btnSimulador = uicontrol('String','Simulador','Position',[20 300 160
50], 'FontSize',13,'FontWeight',
'bold','Callback','Control_Panel=1;pestania');
btnControl_manual = uicontrol('String','Control Manual','Position',[20
250 160 50],'FontSize',13, 'FontWeight',
'bold','Callback','set(btnDesconectar,''visible'',''off'');set(btnCone
ctar,''visible'',''on'');Control_Panel=2;pestania;');
btnControl_Auto = uicontrol('String','Control
Automatico','Position',[20 200 160 50],'FontSize',13, 'FontWeight',
'bold','Callback','set(btnDesconectar2,''visible'',''off'');set(btnCon
ectar2,''visible'',''on'');Control_Panel=3;pestania');
btnNavegacion_Segura = uicontrol('String','Navegacion
Segura','Position',[20 150 160 50],'FontSize',13,'FontWeight',
'bold','Callback','set(btnDesconectar3,''visible'',''off'');set(btnCon
ectar3,''visible'',''on'');Control_Panel=4;pestania');
btnStopfin=   uicontrol('String', 'STOP','Position', [5 0 75 50],
'FontWeight','bold','FontSize',10,'ForegroundColor','r',...
    'Callback', 'close all, clear all');
panelSimulador = uipanel(f,'Title',' SIMULADOR
',’FontSize’,10,'BackgroundColor',[0/255 196/255
222/255],'Position',[0.16 .05 0.83 0.87],'FontWeight', 'bold');
panelControl_Manual = uipanel(f,'Title','CONTROL
MANUAL','FontSize',10,'BackgroundColor',[10/255 255/255
222/255],'Position',[0.16 .05 0.83 0.87],'FontWeight', 'bold');
panelControl_Automatico = uipanel(f,'Title',' CONTROL AUTONOMO
',’FontSize’,10,'BackgroundColor',[176/255 196/255
222/255],'Position',[0.16 .05 0.83 0.87],'FontWeight', 'bold');
panelNavegacion_Segura = uipanel(f,'Title','NAVEGACION
SEGURA','FontSize',10,'BackgroundColor',[255/255 250/255
225/255],'Position',[0.16 .05 0.83 0.87],'FontWeight', 'bold');
```

```
%% Programacion Panel Simulador
btnT1 = uicontrol('String','Trayectoria
1','parent',panelSimulador,'Position',[5 380 120 28], 'FontWeight',
'bold','Callback','Stop=0,Pause=0,trayectoria=1;control_trayectoria_PI
D');
btnT2 = uicontrol('String','Trayectoria
2','parent',panelSimulador,'Position',[5 350 120 28], 'FontWeight',
'bold','Callback','Stop=0,Pause=0,trayectoria=2;control_trayectoria_PI
D');
btnT3 = uicontrol('String','Trayectoria
3','parent',panelSimulador,'Position',[5 320 120 28], 'FontWeight',
'bold','Callback','Stop=0,Pause=0,trayectoria=3;control_trayectoria_PI
D');
%boton de paro de simulacion
btnSt = uicontrol('String','Stop
Simulation','parent',panelSimulador,'Position',[5 50 120 28],
'FontWeight', 'bold','BackgroundColor','r',...
'Callback','Stop=1;if exist(''ploter'',''var''),delete(ploter),end;if
exist(''ploter3D'',''var''),delete(ploter3D),end;deleteObjetos');
%boton de pause de sumulacion
btnPause =
uicontrol('String','Pause/Play','parent',panelSimulador,'Position',[5
80 120 28], 'FontWeight', 'bold','BackgroundColor','g',...
'Callback','Pause=not(Pause)');
%Texto de titulo ingresar posision inical
uicontrol('Style','text','String','Posicion
Inical','FontSize',12,'Position', [15 220 110
25],'BackgroundColor',[0/255 196/255
222/255],'parent',panelSimulador);
% texto de ingreso de posicion inicial en X
uicontrol('Style','text','String', 'Xo=','FontSize',12,'Position', [5
190 35 25],'BackgroundColor',[0/255 196/255
222/255],'parent',panelSimulador);
txt_Pox=
uicontrol('Style','edit','String',0,'FontSize',12,'ForegroundColor','w
','BackgroundColor',[0.3 0.3 0.3],'Position', [40 190 50
25],'parent',panelSimulador);
uicontrol('Style','text','String', '[m]','FontSize',12,'Position', [
85 190 30 25],'BackgroundColor',[0/255 196/255
222/255],'parent',panelSimulador);
% texto de ingreso de posicion inicial en Y
uicontrol('Style','text','String', 'Yo=','FontSize',12,'Position', [5
170 35 25],'BackgroundColor',[0/255 196/255
222/255],'parent',panelSimulador);
txt_Poy=
uicontrol('Style','edit','String',0,'FontSize',12,'ForegroundColor','w
','BackgroundColor',[0.3 0.3 0.3],'Position', [40 170 50
25],'parent',panelSimulador);
uicontrol('Style','text','String', '[m]','FontSize',12,'Position', [85
170 30 25],'BackgroundColor',[0/255 196/255
222/255],'parent',panelSimulador);
% texto de ingreso de posicion inicial en th
uicontrol('Style','text','String', 'Tho=','FontSize',12,'Position', [5
150 35 25],'BackgroundColor',[0/255 196/255
222/255],'parent',panelSimulador);
txt_Poth=
uicontrol('Style','edit','String',0,'FontSize',12,'ForegroundColor','w
','BackgroundColor',[0.3 0.3 0.3],'Position', [40 150 50
25],'parent',panelSimulador);
```

```
uicontrol('Style','text','String', '[rad]','FontSize',12,'Position',
[85 150 35 25],'BackgroundColor',[0/255 196/255
222/255],'parent',panelSimulador);
%****************************fin panel
simulador**********************
%% Programacion Panel Control Manual
btnConectar = uicontrol('String','Conectar
Robot','parent',panelControl_Manual,'Position',[5 300 120 28],
'FontWeight', 'bold','Callback','comunicacion');
btnDesconectar = uicontrol('String','Desconectar
Robot','parent',panelControl_Manual,'Position',[5 300 120 28],
'FontWeight', 'bold','Callback','desconectar');
%boton de paro de simulacion
btnSt = uicontrol('String','Stop
Robot','parent',panelControl_Manual,'Position',[5 200 120 28],
'FontWeight', 'bold','BackgroundColor','r',...
'Callback','Stop=1;Run=0;Detener;deleteObjetos');
%boton de pause de sumulacion
btnRun =
uicontrol('String','Run','parent',panelControl_Manual,'Position',[5
250 120 28], 'FontWeight', 'bold','BackgroundColor','g',...
'Callback','Run=not(Run);Manual;Stop=0;');
%Texto de titulo ingresar velocidades al robot
uicontrol('Style','text','String','VELOCIDADES DE REFERENCIA DEL
ROBOT','FontSize',13,'ForegroundColor','b','Position', [100 350 350
50],'BackgroundColor',[10/255 255/255
222/255],'parent',panelControl_Manual);
% texto de ingreso de Velocidad X
uicontrol('Style','text','String', 'vfd=','FontSize',14,'Position',
[180 300 50 25],'BackgroundColor','w','parent',panelControl_Manual);
txt_vfd=
uicontrol('Style','edit','String',0,'FontSize',14,'ForegroundColor','w
','BackgroundColor',[0.3 0.3 0.3],'Position', [230 300 75
25],'parent',panelControl_Manual);
uicontrol('Style','text','String', '[m/s]','FontSize',14','Position',
[ 305 300 50 25],'BackgroundColor','w','parent',panelControl_Manual);
% texto de ingreso de Velocidad Y
uicontrol('Style','text','String', 'vld=','FontSize',14,'Position',
[180 250 50 25],'BackgroundColor','w','parent',panelControl_Manual);
txt_vld=
uicontrol('Style','edit','String',0,'FontSize',14,'ForegroundColor','w
','BackgroundColor',[0.3 0.3 0.3],'Position', [230 250 75
25],'parent',panelControl_Manual);
uicontrol('Style','text','String', '[m/s]','FontSize',14','Position',
[ 305 250 50 25],'BackgroundColor','w','parent',panelControl_Manual);
% texto de ingreso de Velocidad Angular
uicontrol('Style','text','String', 'Wd=','FontSize',14,'Position',
[180 200 50 25],'BackgroundColor','w','parent',panelControl_Manual);
txt_Wd=
uicontrol('Style','edit','String',0,'FontSize',14,'ForegroundColor','w
','BackgroundColor',[0.3 0.3 0.3],'Position', [230 200 75
25],'parent',panelControl_Manual);
uicontrol('Style','text','String',
'[rad/s]','FontSize',13','Position', [ 305 200 50
25],'BackgroundColor','w','parent',panelControl_Manual);
%Texto Leer las Velocidades del Robot
uicontrol('Style','text','String','VELOCIDADES DEL
ROBOT','FontSize',13,'ForegroundColor','k','Position', [500 350 350
50],'BackgroundColor',[10/255 255/255
222/255],'parent',panelControl_Manual);
% texto de ingreso de Velocidad X
```

```
uicontrol('Style','text','String', 'vf=','FontSize',14,'Position',
[580 300 50 25],'BackgroundColor','w','parent',panelControl_Manual);
txt_vfr=
uicontrol('Style','edit','String',vf,'FontSize',14,'ForegroundColor','
r','BackgroundColor',[0.3 0.3 0.3],'Position', [630 300 75
25],'parent',panelControl_Manual);
uicontrol('Style','text','String', '[m/s]','FontSize',14','Position',
[ 705 300 50 25],'BackgroundColor','w','parent',panelControl_Manual);
% texto de ingreso de Velocidad Y
uicontrol('Style','text','String', 'vl=','FontSize',14,'Position',
[580 250 50 25],'BackgroundColor','w','parent',panelControl_Manual);
txt_vlr=
uicontrol('Style','edit','String',vl,'FontSize',14,'ForegroundColor','
r','BackgroundColor',[0.3 0.3 0.3],'Position', [630 250 75
25],'parent',panelControl_Manual);
uicontrol('Style','text','String', '[m/s]','FontSize',14','Position',
[ 705 250 50 25],'BackgroundColor','w','parent',panelControl_Manual);
% texto de ingreso de Velocidad Angular
uicontrol('Style','text','String', 'W=','FontSize',14,'Position', [580
200 50 25],'BackgroundColor','w','parent',panelControl_Manual);
txt_Wr=
uicontrol('Style','edit','String',w,'FontSize',14,'ForegroundColor','r
','BackgroundColor',[0.3 0.3 0.3],'Position', [630 200 75
25],'parent',panelControl_Manual);
uicontrol('Style','text','String',
'[rad/s]','FontSize',13','Position', [ 705 200 50
25],'BackgroundColor','w','parent',panelControl_Manual);
% Posicion Inicial del robot
uicontrol('Style','text','String','POSICION
INICIAL','FontSize',13,'ForegroundColor','b','Position', [100 115 350
50],'BackgroundColor',[10/255 255/255
222/255],'parent',panelControl_Manual);
% Mostrar la Cinematica del Robot
%Posicion en X
uicontrol('Style','text','String', 'xo=','FontSize',14,'Position',
[180 110 50 25],'BackgroundColor','w','parent',panelControl_Manual);
txt_Xo=
uicontrol('Style','edit','String','0','FontSize',14,'ForegroundColor',
'w','BackgroundColor',[0.3 0.3 0.3],'Position', [230 110 75
25],'parent',panelControl_Manual);
uicontrol('Style','text','String', '[m]','FontSize',13','Position', [
305 110 50 25],'BackgroundColor','w','parent',panelControl_Manual);
%Posicion en Y
uicontrol('Style','text','String', 'yo=','FontSize',14,'Position',
[180 85 50 25],'BackgroundColor','w','parent',panelControl_Manual);
txt_Yo=
uicontrol('Style','edit','String','0','FontSize',14,'ForegroundColor',
'w','BackgroundColor',[0.3 0.3 0.3],'Position', [230 85 75
25],'parent',panelControl_Manual);
uicontrol('Style','text','String', '[m]','FontSize',13','Position', [
305 85 50 25],'BackgroundColor','w','parent',panelControl_Manual);
%rotacion
uicontrol('Style','text','String', 'tho=','FontSize',14,'Position',
[180 60 50 25],'BackgroundColor','w','parent',panelControl_Manual);
txt_tho=
uicontrol('Style','edit','String','0','FontSize',14,'ForegroundColor',
'w','BackgroundColor',[0.3 0.3 0.3],'Position', [230 60 75
25],'parent',panelControl_Manual);
uicontrol('Style','text','String', '[rad]','FontSize',13','Position',
[305 60 50 25],'BackgroundColor','w','parent',panelControl_Manual);
x(1)=str2double((get(txt_Xo, 'String')));
```

```
y(1)=str2double((get(txt_Yo, 'String')));
th(1)=str2double((get(txt_tho, 'String')));
%Texto Mostrar Posiicon del robot
uicontrol('Style','text','String','POSICION DEL
ROBOT','FontSize',13,'ForegroundColor','k','Position', [500 115 350
50],'BackgroundColor',[10/255 255/255
222/255],'parent',panelControl_Manual);
% Mostrar la Cinematica del Robot
%Posicion en X
uicontrol('Style','text','String', 'x=','FontSize',14,'Position', [580
110 50 25],'BackgroundColor','w','parent',panelControl_Manual);
txt_X=
uicontrol('Style','text','String',x(k),'FontSize',14,'ForegroundColor'
,'r','BackgroundColor','w','Position', [630 110 75
25],'parent',panelControl_Manual);
uicontrol('Style','text','String', '[m]','FontSize',13','Position', [
705 110 50 25],'BackgroundColor','w','parent',panelControl_Manual);
%Posicion en Y
uicontrol('Style','text','String', 'y=','FontSize',14,'Position', [580
85 50 25],'BackgroundColor','w','parent',panelControl_Manual);
txt_Y=
uicontrol('Style','text','String',y(k),'FontSize',14,'ForegroundColor'
,'r','BackgroundColor','w','Position', [630 85 75
25],'parent',panelControl_Manual);
uicontrol('Style','text','String', '[m]','FontSize',13','Position', [
705 85 50 25],'BackgroundColor','w','parent',panelControl_Manual);
%rotacion
uicontrol('Style','text','String', 'th=','FontSize',14,'Position',
[580 60 50 25],'BackgroundColor','w','parent',panelControl_Manual);
txt_th=
uicontrol('Style','text','String',th(k),'FontSize',14,'ForegroundColor
','r','BackgroundColor','w','Position', [630 60 75
25],'parent',panelControl_Manual);
uicontrol('Style','text','String', '[rad]','FontSize',13','Position',
[705 60 50 25],'BackgroundColor','w','parent',panelControl_Manual);
%**********************fin panel control manual***************
%% Programacion Panel Control Autonomo
btnConectar2 = uicontrol('String','Conectar
Robot','parent',panelControl_Automatico,'Position',[5 290 120 28],
'FontWeight', 'bold','Callback','comunicacion');
btnDesconectar2 = uicontrol('String','Desconectar
Robot','parent',panelControl_Automatico,'Position',[5 290 120 28],
'FontWeight', 'bold','Callback','desconectar');
btnT1 = uicontrol('String','Trayectoria
1','parent',panelControl_Automatico,'Position',[5 380 120 28],
'FontWeight', 'bold','Callback','Stop=0,Pause=0,trayectoria=1;if
exist(''ploter'',''var''),delete(ploter),end;if
exist(''ploter3D'',''var''),delete(ploter3D),end;control_trayectoria_P
ID_experimental');
btnT2 = uicontrol('String','Trayectoria
2','parent',panelControl_Automatico,'Position',[5 350 120 28],
'FontWeight', 'bold','Callback','Stop=0,Pause=0,trayectoria=2;if
exist(''ploter'',''var''),delete(ploter),end;if
exist(''ploter3D'',''var''),delete(ploter3D),end;control_trayectoria_P
ID_experimental');
btnT3 = uicontrol('String','Trayectoria
3','parent',panelControl_Automatico,'Position',[5 320 120 28],
'FontWeight', 'bold','Callback','Stop=0,Pause=0,trayectoria=3;if
exist(''ploter'',''var''),delete(ploter),end;if
exist(''ploter3D'',''var''),delete(ploter3D),end;control_trayectoria_P
ID_experimental');
```

```
%tiempo de simulacion ingresar
uicontrol('Style','text','String','Tiempo','FontSize',10,'Position',
[130 380 50 28],'BackgroundColor',[0/255 196/255
222/255],'parent',panelControl_Automatico);
txt_tfin2 =
uicontrol('style','edit','String',10','parent',panelControl_Automatico
,'Position',[130 350 50 28], 'FontWeight', 'bold');
%boton de paro de simulacion
btnSt = uicontrol('String','Stop
Robot','parent',panelControl_Automatico,'Position',[5 50 120 28],
'FontWeight', 'bold','BackgroundColor','r',...
'Callback','Stop=1;if exist(''ploter'',''var''),delete(ploter),end;if
exist(''ploter'',''var''),delete(ploter),end;Detener;deleteObjetos');
%Texto de titulo ingresar posision inical
uicontrol('Style','text','String','Posicion
Inical','FontSize',12,'Position', [15 220 110
25],'BackgroundColor',[0/255 196/255
222/255],'parent',panelControl_Automatico);
% texto de ingreso de posicion inicial en X
uicontrol('Style','text','String', 'Xo=','FontSize',12,'Position', [5
190 35 25],'BackgroundColor',[0/255 196/255
222/255],'parent',panelControl_Automatico);
txt_Pox2=
uicontrol('Style','edit','String',0,'FontSize',12,'ForegroundColor','w
','BackgroundColor',[0.3 0.3 0.3],'Position', [40 190 50
25],'parent',panelControl_Automatico);
uicontrol('Style','text','String', '[m]','FontSize',12,'Position', [
85 190 30 25],'BackgroundColor',[0/255 196/255
222/255],'parent',panelControl_Automatico);
% texto de ingreso de posicion inicial en Y
uicontrol('Style','text','String', 'Yo=','FontSize',12,'Position', [5
170 35 25],'BackgroundColor',[0/255 196/255
222/255],'parent',panelControl_Automatico);
txt_Poy2=
uicontrol('Style','edit','String',0,'FontSize',12,'ForegroundColor','w
','BackgroundColor',[0.3 0.3 0.3],'Position', [40 170 50
25],'parent',panelControl_Automatico);
uicontrol('Style','text','String', '[m]','FontSize',12,'Position', [85
170 30 25],'BackgroundColor',[0/255 196/255
222/255],'parent',panelControl_Automatico);
% texto de ingreso de posicion inicial en th
uicontrol('Style','text','String', 'Tho=','FontSize',12,'Position', [5
150 35 25],'BackgroundColor',[0/255 196/255
222/255],'parent',panelControl_Automatico);
txt_Poth2=
uicontrol('Style','edit','String',0,'FontSize',12,'ForegroundColor','w
','BackgroundColor',[0.3 0.3 0.3],'Position', [40 150 50
25],'parent',panelControl_Automatico);
uicontrol('Style','text','String', '[rad]','FontSize',12,'Position',
[85 150 35 25],'BackgroundColor',[0/255 196/255
222/255],'parent',panelControl_Automatico);
%********************fin panel control
autónomo***********************
%% Programación Panel Control navegacion Segura
btnConectar3 = uicontrol('String','Conectar Robot y
Sensor','parent',panelNavegacion_Segura,'Position',[5 290 150 28],
'FontWeight', 'bold','Callback','comunicacion2');
btnDesconectar3 = uicontrol('String','Desconectar
Robot','parent',panelNavegacion_Segura,'Position',[5 290 150 28],
'FontWeight', 'bold','Callback','desconectar2');
set(btnConectar3,'Visible','on');
```

```
set( btnDesconectar3,'Visible','off');
btn3T1 = uicontrol('String','Trayectoria
1','parent',panelNavegacion_Segura,'Position',[5 380 120 28],
'FontWeight', 'bold','Callback','Stop=0,Pause=0,trayectoria=1;if
exist(''ploter'',''var''),delete(ploter),end;if
exist(''ploter3D'',''var''),delete(ploter3D),end;control_trayectoria_P
ID_experimental_eva');
btn3T2 = uicontrol('String','Trayectoria
2','parent',panelNavegacion_Segura,'Position',[5 350 120 28],
'FontWeight', 'bold','Callback','Stop=0,Pause=0,trayectoria=2;if
exist(''ploter'',''var''),delete(ploter),end;if
exist(''ploter3D'',''var''),delete(ploter3D),end;control_trayectoria_P
ID_experimental_eva');
btn3T3 = uicontrol('String','Trayectoria
3','parent',panelNavegacion_Segura,'Position',[5 320 120 28],
'FontWeight', 'bold','Callback','Stop=0,Pause=0,trayectoria=3;if
exist(''ploter'',''var''),delete(ploter),end;if
exist(''ploter3D'',''var''),delete(ploter3D),end;control_trayectoria_P
ID_experimental_eva');
%tiempo de simulacion ingresar
uicontrol('Style','text','String','Tiempo','FontSize',10,'Position',
[130 380 50 28],'BackgroundColor',[0/255 196/255
222/255],'parent',panelNavegacion_Segura);
txt_tfin3 =
uicontrol('style','edit','String',10','parent',panelNavegacion_Segura,
'Position',[130 350 50 28], 'FontWeight', 'bold');
%boton de paro de simulacion
btnSt3 = uicontrol('String','Stop
Robot','parent',panelNavegacion_Segura,'Position',[5 50 120 28],
'FontWeight', 'bold','BackgroundColor','r',...
'Callback','Stop=1;if exist(''ploter'',''var''),delete(ploter),end;if
exist(''ploter'',''var''),delete(ploter),end;detener;deleteObjetos');
%Texto de titulo ingresar posision inical
uicontrol('Style','text','String','Posicion
Inical','FontSize',12,'Position', [15 220 110
25],'BackgroundColor',[0/255 196/255
222/255],'parent',panelNavegacion_Segura);
% texto de ingreso de posicion inicial en X
uicontrol('Style','text','String', 'Xo=','FontSize',12,'Position', [5
190 35 25],'BackgroundColor',[0/255 196/255
222/255],'parent',panelNavegacion_Segura);
txt_Pox3=
uicontrol('Style','edit','String',0,'FontSize',12,'ForegroundColor','w
','BackgroundColor',[0.3 0.3 0.3],'Position', [40 190 50
25],'parent',panelNavegacion_Segura);
uicontrol('Style','text','String', '[m]','FontSize',12,'Position', [
85 190 30 25],'BackgroundColor',[0/255 196/255
222/255],'parent',panelNavegacion_Segura);
% texto de ingreso de posicion inicial en Y
uicontrol('Style','text','String', 'Yo=','FontSize',12,'Position', [5
170 35 25],'BackgroundColor',[0/255 196/255
222/255],'parent',panelNavegacion_Segura);
txt_Poy3=
uicontrol('Style','edit','String',0,'FontSize',12,'ForegroundColor','w
','BackgroundColor',[0.3 0.3 0.3],'Position', [40 170 50
25],'parent',panelNavegacion_Segura);
uicontrol('Style','text','String', '[m]','FontSize',12,'Position', [85
170 30 25],'BackgroundColor',[0/255 196/255
222/255],'parent',panelNavegacion_Segura);
% texto de ingreso de posicion inicial en th
```

```
uicontrol('Style','text','String', 'Tho=','FontSize',12,'Position', [5
150 35 25],'BackgroundColor',[0/255 196/255
222/255],'parent',panelNavegacion_Segura);
txt_Poth3=
uicontrol('Style','edit','String',0,'FontSize',12,'ForegroundColor','w
','BackgroundColor',[0.3 0.3 0.3],'Position', [40 150 50
25],'parent',panelNavegacion_Segura);
uicontrol('Style','text','String', '[rad]','FontSize',12,'Position',
[85 150 35 25],'BackgroundColor',[0/255 196/255
222/255],'parent',panelNavegacion_Segura);
%****fin panel control autónomo con evasión de obstaculos************
```

Anexo 9.1. Función comunicación con el robot

```
%comunicacion con el robot
delete(instrfind({'Port'},{'COM19'}));
S=serial('COM19');
%configuracion ouerto
set(S,'Baudrate',19200);          % se configura la velocidad a 9600
Baudios
set(S,'StopBits',1);             % se configura bit de parada a uno
set(S,'DataBits',8);             % se configura que el dato es de 8
bits, debe estar entre 5 y 8
set(S,'Parity','none');          % se configura sin paridad
set(S,'Terminator','CR/LF');     % caracter con que finaliza el envÃ-o
set(S,'Timeout',5);              % 5 segundos de tiempo de espera
fopen(S)
%revisar estatus y activar banderas e indicadores
if( S.status == 'open')
set(btnConectar,'Visible','off');
set( btnDesconectar,'Visible','on');
set(btnConectar2,'Visible','off');
set( btnDesconectar2,'Visible','on');
set(btnConectar3,'Visible','off');
set( btnDesconectar3,'Visible','on');
Control_Panel=5;pestania;
else
set(btnConectar,'Visible','off');
set( btnDesconectar,'Visible','on');
set(btnConectar2,'Visible','off');
set( btnDesconectar2,'Visible','on');
set(btnConectar3,'Visible','off');
set( btnDesconectar3,'Visible','on');
Control_Panel=6;pestania;
delete(S);
end
```

Anexo 9.2. Función desconectar el robot y sensor

```
%desconectar Robot
if exist('S')
fclose(S);
delete(S);
%Stop the sensor
calllib('radarDLL','stop');
pause(2)
%unload the dynamic link library
unloadlibrary radarDLL
set(btnConectar,'Visible','on')
set( btnDesconectar,'Visible','off')
```

```
set(btnConectar2,'Visible','on')
set(btnDesconectar2,'Visible','off')
set(btnConectar3,'Visible','on')
set(btnDesconectar3,'Visible','off')
Control_Panel=6;pestania;
end
```