EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

Tecnológico Nacional de México

Centro Nacional de Investigación y Desarrollo Tecnológico

Tesis de Maestría

Navegación de un robot omnidireccional basada en Lógica Difusa Tipo-2

presentada por

Ing. Francisco Javier Valdepeña Rivera

como requisito para la obtención del grado de

Maestro en Ciencias de la Computación

Director de tesis

Dr. Dante Mújica Vargas

Codirector de tesis

Dr. Noé Alejandro Castro Sánchez

Cuernavaca, Morelos, México. Enero de 2022.

cenidet
Centro Nacional de Investigación y Desarrollo Tecnológico

EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

Centro Nacional de Investigación y Desarrollo Tecnológico
Departamento De Ciencias Computacionales

Cuernavaca, Mor., 14/diciembre/2021

OFICIO No. DCC/204/2021
Asunto: Aceptación de documento de tesis
CENIDET-AC-004-M14-OFICIO

DR. CARLOS MANUEL ASTORGA ZARAGOZA
SUBDIRECTOR ACADÉMICO
PRESENTE

Por este conducto, los integrantes de Comité Tutorial del C. FRANCISCO JAVIER VALDEPEÑA RIVERA, con número de control M19CE065, de la Maestría en Ciencias de la Computación, le informamos que hemos revisado el trabajo de tesis de grado titulado **"NAVEGACIÓN DE UN ROBOT OMNIDIRECCIONAL BASADA EN LÓGICA DIFUSA TIPO 2"**, y hemos encontrado que se han atendido todas las observaciones que se le indicaron, por lo que hemos acordado aceptar el documento de tesis y le solicitamos la autorización de impresión definitiva.

DR. DANTE MÚJICA VARGAS
Director de tesis

DR. NOÉ ALEJANDRO CASTRO SÁNCHEZ
Codirectora de Tesis

DR. MÁXIMO LÓPEZ SÁNCHEZ
Revisor

DR. JUAN GABRIEL GONZÁLEZ SERNA
Revisor

S.E.P. CENTRO NACIONAL DE INVESTIGACIÓN Y DESARROLLO TECNOLÓGICO
RECIBIDO
15 DIC 2021
CIENCIAS COMPUTACIONALES

C.c.p. Depto. Servicios Escolares.
Expediente / Estudiante
JGGS/ibm

cenidet
Interior Internado Palmira S/N, Col. Palmira,
C.P. 62490, Cuernavaca, Morelos
Tel. (01) 777 3 62 77 70, ext. 3201.
e-mail: dcc@cenidet.tecnm.mx
www.tecnm.mx | www.cenidet.tecnm.mx

México 2021 Año de la Independencia

EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

Centro Nacional de Investigación y Desarrollo Tecnológico
Departamento De Ciencias Computacionales

Cuernavaca, Mor., 14/diciembre/2021

OFICIO No. DCC/204/2021
Asunto: Aceptación de documento de tesis
CENIDET-AC-004-M14-OFICIO

DR. CARLOS MANUEL ASTORGA ZARAGOZA
SUBDIRECTOR ACADÉMICO
PRESENTE

Por este conducto, los integrantes de Comité Tutorial del C. FRANCISCO JAVIER VALDEPEÑA RIVERA, con número de control M19CE065, de la Maestría en Ciencias de la Computación, le informamos que hemos revisado el trabajo de tesis de grado titulado **"NAVEGACIÓN DE UN ROBOT OMNIDIRECCIONAL BASADA EN LÓGICA DIFUSA TIPO 2"**, y hemos encontrado que se han atendido todas las observaciones que se le indicaron, por lo que hemos acordado aceptar el documento de tesis y le solicitamos la autorización de impresión definitiva.

DR. DANTE MÚJICA VARGAS
Director de tesis

DR. NOÉ ALEJANDRO CASTRO SÁNCHEZ
Codirectora de Tesis

DR. MÁXIMO LÓPEZ SÁNCHEZ
Revisor

DR. JUAN GABRIEL GONZÁLEZ SERNA
Revisor

S.E.P. CENTRO NACIONAL DE INVESTIGACIÓN Y DESARROLLO TECNOLÓGICO
RECIBIDO
15 DIC 2021
CIENCIAS COMPUTACIONALES

C.c.p. Depto. Servicios Escolares.
Expediente / Estudiante
JGGS/ibm

cenidet
Interior Internado Palmira S/N, Col. Palmira,
C.P. 62490, Cuernavaca, Morelos
Tel. (01) 777 3 62 77 70, ext. 3201.
e-mail: dcc@cenidet.tecnm.mx
www.tecnm.mx | www.cenidet.tecnm.mx

México 2021 Año de la Independencia

Dedicatoria

Esta tesis se la dedico a mis padres ***Elvia Rivera Gómez*** y ***Francisco Javier Valdepeña Rivera*** por su constante compromiso, cariño y apoyo incondicional durante esta etapa de mi vida. A mis profesores que me motivaron e inspiraron a desarrollarme en el área de ciencia de la computación. A mis amigos Ángel Arturo Rendón Castro, Antonio Luna Álvarez y Virna Viridiana Vela Rincón por compartirme sus conocimientos y brindarme asesoría en temas relacionados a mi proyecto.

Agradecimientos

Agradezco al Consejo Nacional de Ciencia y Tecnología (CONACYT) por el apoyo otorgado durante el desarrollo de esta tesis para la obtención de grado de maestro en ciencias de la computación, mediante su sistema de becas de posgrado. Al Centro Nacional de Investigación y Desarrollo Tecnológico (CENIDET), perteneciente al TECNM, por haber permitido realizar los estudios de una Maestría en Ciencias en sus instalaciones.

Agradezco a mi director de tesis el Dr. Dante Mújica Vargas por su apoyo durante el desarrollo del tema de tesis, así como su guía en el tema de la Teoría Difusa y asesoría brindada durante mi formación como maestro. A mi co-director el Dr. Noé Alejandro Castro Sánchez por sus comentarios y aportaciones para el desarrollo de tema de tesis. Al comité revisor conformado por el Dr. Juan Gabriel González Serna y el Dr. Máximo López Sánchez, por sus aportaciones y recomendaciones durante el desarrollo esta investigación. De igual manera al Dr. Miguel Ángel Ruíz Jaimes por su recomendación y apoyo durante mi formación en este programa de maestría.

Resumen

En este trabajo se presenta el diseño, desarrollo y prueba de un sistema de navegación para un robot omnidireccional basado en Lógica Difusa Tipo-2. Este trabajo se divide en tres apartados principales: El primer apartado es la adquisición de información a través de un mapa de profundidad, utilizando una cámara estéreo, el segundo apartado es un sistema de inferencia mediante la Lógica Difusa Tipo-2 y el tercero es su aplicación sobre un robot omnidireccional con ruedas tipo *mecanum*. La adquisición de información se realiza mediante la obtención de las distancias entre objetos por medio de los mapas de profundidad del mismo entorno y utilizando odometría visual para obtener las poses dentro de la trayectoria. El sistema de inferencia utiliza Lógica Difusa Tipo-2 en un controlador que maneja la incertidumbre del desplazamiento del robot omnidireccional, en específico es un sistema difuso de intervalo tipo-2 que utiliza la visión estéreo para controlar las velocidades lineales y angulares del robot. La implementación de este sistema de navegación se ejecuta en un robot omnidireccional con ruedas *mecanum* sobre una plataforma embebida NVIDIA Jetson TX2 y una tarjeta *BeagleBone Blue* para el control cinemático, dicho prototipo se probó en un escenario delimitado y conocido.

Las métricas para la evaluación del sistema de navegación son la longitud de trayectoria cubierta, distancia media de la meta, evaluación de curvatura en trayectorias para determinar la suavidad del desplazamiento en un robot. Se plantearon experimentos para evaluar un escenario con obstáculos estáticos y dinámicos para analizar las trayectorias generadas por el sistema.

Los resultados cuantitativos, muestran que el control en una trayectoria generada y seguida, para llegar a una meta resulta eficiente al implementar la visión como tipo de navegación, teniendo un rendimiento mayor a 0.80 en escenarios con obstáculos estáticos y dinámicos, con la ruta más corta obtenida por el algoritmo PRM un planificador de trayectoria.

Palabras clave: Lógica Difusa Tipo-2, Visión estéreo, Algoritmo planificador de trayectoria, Sistema embebido, robot omnidireccional, Odometría visual, Modelo cinemático dinámico.

Abstract

In this work is proposed a navegation system based on Fuzzy Logic Type-2 for an omniderational. The work is divided into three main sections: The acquisition of information through a depth map, using a stereo camera to obtain the distances of the objects and visual odometry. The inference system uses Type-2 Fuzzy Logic in a controller that handles the uncertainty of the omnidirectional robot movement, specifically it is a type-2 interval diffuse system that uses stereo vision to control the linear and angular speeds of the robot. Finally, the implementation is executed on an embedded NVIDIA Jetson TX2 platform and a Bleaglebone Blue card creating a prototype that was equipment on RoboMaster S1 and tested in a defined and well-known scenario.

The proposed navigation system was evaluated with metrics for spatial dimensions such as covered path length, mean distance from the target evaluation of curvature in paths to determine the smoothness of the displacement in a robot. For the experiments, real time navigation with static and dynamic obstacles to analyze the trajectories generated by the system.

The quantitative results show that the control proposed in a trajectory generated and followed, to reach a goal is efficient when implementing the vision as a type of navigation, having a performance greater than 0.80 in scenarios with static and dynamic obstacles, with the route shortest obtained by the PRM algorithm a trajectory planner.

Keywords: Fuzzy Logic Type-2, Stereo vision, Trajectory planning algorithm, Embedded system, omnidirectional robot, Visual odometry, dynamic kinematic model.

Índice General

Índice de Tablas

Índice de Figuras

Capítulo 1

Introducción

En la robótica móvil existen diferentes accionamientos en desplazamientos tales como, holonómicos y no-holonómicos, este último tiene la desventaja de no desplazarse en todas las direcciones posibles, con respecto a un plano en x, y. Los holonómicos por el contrario tienen la capacidad de desplazarse en todas direcciones en cualquier instante de tiempo. En este sentido el funcionamiento de este tipo robot, proporciona un desempeño eficiente en escenarios donde los obstáculos impidan obtener trayectorias lo más cortas posible a un punto meta determinado. El objetivo es generar un control para la navegación de este tipo de accionamiento robótico para desplazarse sobre una trayectoria obtenida de un nodo inicial hacia una meta. Un sistema de navegación requiere información del entorno para así generar algún control robótico, esta información puede ser obtenida por diferentes sensores como por ejemplo, cámaras, infrarrojos, dispositivos de medición inercial, ultrasónicos, LIDAR, entre otros. Los sistemas de navegación basados en información visual son los más implementados por sus diferentes niveles de complejidad, un sistema basado en visión monocular requiere un menor rendimiento computacional que el de visión estéreo, pero este último aporta más información, tales como distancias hacia objetos captados en puntos específicos de visión, reconstrucción del entorno en un mapa de profundidad digital, generación de odometría visual en tiempo real y esto en cada fotograma captado, por lo cual, resulta conveniente para procesarla en un sistema de control mejor definido para controlar el desplazamiento robótico dentro de una trayectoria con obstáculos estáticos o dinámicos en entorno.

La complejidad vista en diferentes trabajos con el uso de navegación visual de tipo estereoscópica es la alta demanda de recurso computacional ya que se requiere de sistemas embebidos para lograr trabajar grandes cantidades de información visual en tiempo real y el procesar dicha información dentro de un control de navegación con Lógica Difusa la complejidad aumenta, es por ello que se genera un desempaquetado de la información y solo se busca implementar ciertas características que logren detectar la

presencia y distancia del objeto en el entorno. Es por ello que un enfoque de solución a la incertidumbre que se puede generar de esta información es la Lógica Difusa y en especial en este trabajo es la Tipo-2, exportando la experiencia humana a través de reglas que definen mejor un comportamiento robótico.

En esta tesis, se presenta un sistema de navegación sobre un robot de tipo omnidireccional basado en un control con Lógica Difusa Tipo-2 y visión estéreo procesada en un sistema embebido. Este robot móvil utiliza la información de un mapa de profundidad para detectar la distancia de los objetos en los puntos de visión centro, derecho e izquierdo, para después implementarla en el sistema de inferencia, con Lógica Difusa Tipo-2 y generar el control de las velocidades lineales y angulares del robot, al mismo tiempo la generación de la odometría para obtener las poses y seguir la trayectoria hacia la meta.

La implementación del sistema de inferencia con Lógica Difusa Tipo-2 se realizó en el sistema embebido Jetson TX2 que lleva a cabo las tareas de detección en inferencia en tiempo real. La ventaja de implementar este tipo de *hardware* es la robustez de las tareas que se requieren para el sistema de navegación, la visión estéreo y el control difuso Tipo-2 que requieres realizar diversas operaciones matemáticas.

1.1. Planteamiento del problema

La implementación de robots de tipo no-holonómico, son muy utilizados por su facilidad y baja complejidad de implementación pero presenta desventajas en cuanto a sus desplazamientos, ya que no son capaces de desplazarse en todas direcciones en cualquier instante de tiempo. Esta desventaja no es vista en los tipo holonómico, pero el desarrollar un sistema de navegación para controlar los desplazamientos en un entorno con obstáculos estáticos y dinámicos de este robot móvil no es algo trivial. El problema que se planea resolver es el control de los desplazamientos de un robot móvil holonómico, este control robótico requiere de un sistema el cual pueda utilizar la información obtenida de un sensor ya sea visual o de cualquier otro tipo y procesarlas con técnicas avanzadas de manejo de incertidumbre en modelos cinemáticos para poder realizar los desplazamientos de un robot omnidireccional. Para lograr un buen control de los desplazamientos del robot móvil se propone, implementar y evaluar un algoritmo inteligente que sea capaz de desplazar al robot en un espacio con obstáculos.

Este trabajo de tesis tiene por objeto desarrollar un sistema de navegación para un robot omnidireccional pueda trabajar con la incertidumbre inherente del entorno real, capaz de evitar colisiones y realizar un seguimiento trayectorias exacto. El algoritmo propuesto se basará en Lógica Difusa Tipo-2 e implementará visión estéreo para realizar

la adquisición de información.

1.1.1. Hipótesis

El uso de Lógica Difusa Tipo-2 para el control de robots móviles, especialmente para holonómicos (omnidireccionales) podría permitir una mejor navegación y control, utilizando la información visual de una cámara estéreo para la detección de proximidad a los obstáculos, para brindar una reacción en tiempo real durante la navegación.

1.1.2. Delimitación del problema

El problema se enfocará en implementar un robot omnidireccional con ruedas *mecanum* En el contexto de la robótica móvil se pretende resolver la deficiencia que tienen robots de tipo no-holonómico, para desplazarse en diferentes direcciones durante cualquier instante de tiempo, en este sentido se propone la implementación de un robot de configuración omnidireccional con ruedas tipo *Mecanum*, ya que este tipo de robot es capaz de desplazarse en todas direcciones, sin la necesidad de realizar movimientos complejos de dirección y una disminución de la velocidad para controlar el seguimiento de trayectorias no-lineales.

1.1.3. Complejidad

- Es un enfoque a la navegación robótica reciente. Existen pocos artículos implementando Lógica Difusa Tipo-2 en navegación.
- El control de basado Lógica Difusa Tipo 2 es más complejo que el de Tipo 1.
- La implementación de un sistema operativo robótico para la configuración de la GPU y el algoritmo de navegación.

1.2. Objetivos

1.2.1. Objetivo general

Proponer, implementar y evaluar un algoritmo para la navegación autónoma de un robot omnidireccional, basado en Lógica Difusa Tipo-2.

1.2.2. Objetivos específicos

- Analizar y comprender la formalización matemática del movimiento omnidireccional.
- Analizar y comprender la conceptualización de la Teoría Difusa Tipo-2 enfocada a la navegación autónoma.

- Entender los datos inerciales que un dispositivo de medición inercial evalúa, tomando como ejemplo la tarjeta *Beaglebone Blue*.
- Hacer el modelado del sistema de navegación autónoma, incorporando la metodología solución.
- Realizar la configuración y programación en las tarjetas *Beaglebone Blue* y GPU NVIDIA Jetson TX2.
- Evaluar resultados durante la navegación del robot omnidireccional mediante métricas cuantitativas.

1.3. Alcances y Limitaciones

1.3.1. Alcances

- Procesar en tiempo real los datos de profundidad de la cámara ZED.
- Controlar la velocidad y aceleración del robot omnidireccional.
- Determinar la pose del robot a partir del sensor inercial.
- Evasión de obstáculos mediante el uso de una imagen de profundidad.
- Implementar el Controlador Difuso Tipo-2 sobre una GPU NVIDIA Jetson TX2.

1.3.2. Limitaciones

- Parámetros circunstanciales.

  Condiciones de luz anormales.

  Obstáculos imprevistos en un corto lapso de tiempo de reacción.
- Condiciones insólitas en los caminos que afecten la adherencia de la rueda a la superficie.

  Suelos con superficies rugosas.

  Suelos con superficies deslizantes.
- Se limitará de tiempo de navegación para evitar acumular el error.
- Navegará solo en espacios cerrados.

1.4. Justificación

El motivo de implementar Lógica Difusa Tipo-2 para la creación del sistema de navegación autónoma sobre un robot omnidireccional tipo *mecanum*, es que tiene la capacidad de

trabajar mejor la incertidumbre que se obtiene de los sensores visuales e inerciales. Dado el buen desarrollo de cada uno de los bloques del sistema difuso es como trabajará mejor sobre este tipo de robot móvil. La implementación de una cámara ZED como medio de visión estéreo proporcionará más precisión para reaccionar y evitar obstáculos, siendo más exacto en el control de la velocidad lineal y angular del robot. Para tener el robot funcionando correctamente y de mejor medida es utilizando una plataforma embebida como GPU NVIDIA Jetson TX2, logrando un procesamiento más eficiente sin perder información durante el proceso de navegación.

1.5. Organización de la Tesis

En este documento de Tesis se organiza de la siguiente manera:

En el Capítulo 2 se realiza una descripción de los conceptos teóricos los cuales son necesarios para comprender, accionamiento omnidireccional, modelos matemáticos para comprender el comportamiento cinemático de un robot omnidireccional con ruedas tipo *mecanum*, la implementación de visión estéreo para evasión de obstáculos en robot móviles y fundamentos de la Lógica Difusa Tipo-2 para entender el método implementado en este trabajo.

En el Capítulo 3 se presenta un análisis de los trabajos más relacionados con robots móviles omnidireccionales, implementación de sensores de visión estéreo en un robot móvil, técnicas de control cinemático e implementación de modelos difusos Tipo-2 para control robótico.

En el Capítulo 4 se describe a detalle el sistema para la navegación autónoma de un robot omnidireccional, enfocando las partes de adquisición de información, controlador difuso e implementación del sistema de navegación.

En el Capítulo 5 se explican los experimentos realizados, se da una evaluación conforme a conceptos y métricas cuantitativas.

Finalmente en el Capítulo 6, se exponen las conclusiones obtenidas, recomendaciones para trabajos futuros, así como un análisis de los objetivos y alcances plateados en esta investigación.

Capítulo 2

Marco Teórico

En esta sección se detallan los conceptos básicos, algoritmos, software implementado para entender y desarrollar el tema de tesis propuesto.

2.1. Modelo cinemático

El modelo cinemático describe las relaciones geométricas que están presentes en el robot móvil. Donde la relación entre los parámetros de control y su comportamiento está en función del robot físico y su entorno. Un modelo cinemático se describe mediante las velocidades del sistema y se presenta con un conjunto de ecuaciones diferenciales de primer orden.

En un modelo dinámico los parámetros a considerar se encuentra la física del movimiento, energía para desplazamiento, masa del sistema e inercia, así como los parámetros de velocidad. Estos modelos dinámicos vienen representados por ecuaciones diferenciales de segundo orden *(Klančar y cols., 2017)*.

En una simulación los parámetros físicos no tienen demasiada importancia ya que se definen de manera global en una sección del código que describe al robot y su entorno resultando más factible implementar un sistema con ecuaciones de primer orden. En cambio, cuando se realiza una experimentación en un entorno real y tomando en cuenta este accionamiento omnidireccional resulta más preciso tener un control de las variables físicas para los movimientos en este robot móvil y la elección de desplazamiento de cada motor. Una opción de modelo dinámico es en el que no se toman en cuenta los pares gravitacionales, a través del método Euler-Lagrange. La energía cinética del robot omnidireccional se determina por la energía rotacional de cada rueda y tanto como de traslación y rotación del robot omnidireccional. Resultando en la siguiente representación del modelo Lagrangiano L

$$L = \frac{1}{2}\left[M\left((\dot{x}_G)^2 + (\dot{y}_G)^2\right) + I_g\theta^2 + \sum_{i=1}^{4} m_i\omega_i^2\right] \tag{2.1}$$

donde, los parámetros a considerar son masa del robot M, el momento inercial I_g del robot sobre el eje z, el momento de inercia de cada rueda m_i donde, $i = 1, 2, 3, 4$, las velocidades traslacionales $(\dot{x}_G,\ \dot{y}_G)$ sobre el marco G, la velocidad del robot θ sobre el eje z y angular de cada rueda ω_i.

2.2. Modelo cinemático omnidireccional

La cinemática directa se puede utilizar para determinar la pose actual del robot omnidireccional a partir de la velocidades angulares y lineales reales del robot en cualquier instante de la navegación sobre una trayectoria planificada. La representación gráfica del robot omnidireccional se observa en la Figura 2.1.

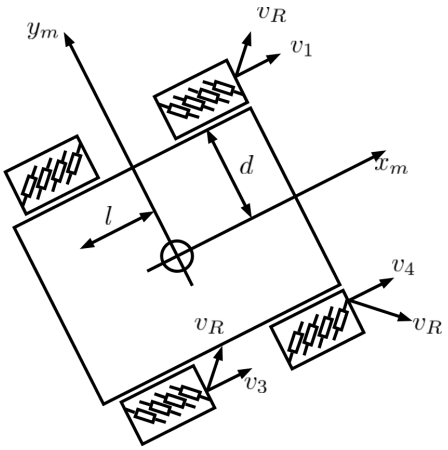


Figura 2.1: Estructura y variables de un robot omnidireccional con ruedas tipo *mecanum*.

donde las variables físicas del robot son: y_m posición en y, x_m posición en x, d distancia entre la rueda y el centro del cuerpo, l distancia frontal hacia el centro del robot, v_R Velocidad angular, v_1 y lineal del robot.

Con el propósito de controlar la base del robot omnidireccional se modeló una serie de ecuaciones para representar la cinemática, tal como se muestra a continuación *(Klancar y cols., 2017)*.

$$\begin{bmatrix} V_x \\ V_y \\ \omega_z \end{bmatrix} = \frac{R}{4L}\begin{bmatrix} -L & L & -L & L \\ L & L & L & L \\ 1 & -1 & -1 & 1 \end{bmatrix}\begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix} \tag{2.2}$$

donde, L es la representación del diámetro del robot omnidireccional resultado de la

suma de l_1 y l_2, distancias al centro del extremo frontal y reverso, suponiendo simetría puede ser el doble de l, ω_i, $i = (1, 2, 3, 4)$ velocidad angular de cada rueda, R es el radio de la rueda. Efectuando esta ecuación resulta en lo siguiente.

$$\begin{bmatrix} V_x \\ V_y \\ \omega_z \end{bmatrix} = \frac{R}{4} \begin{bmatrix} -1 & 1 & -1 & 1 \\ 1 & 1 & 1 & 1 \\ \frac{1}{L} & -\frac{1}{L} & -\frac{1}{L} & \frac{1}{L} \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix} \tag{2.3}$$

donde, la distancia L es tomada en cuenta durante el desplazamiento del robot, ω es la velocidad angular de la rueda $i(i = 1...4)$. Si se desea conocer la velocidad del robot, tenemos que calcular la velocidad angular ω de cada rueda(es decir, la solución de la velocidad inversa), de la siguiente forma.

$$\begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix} = \frac{1}{R} \begin{bmatrix} 1 & 1 & -L \\ 1 & -1 & L \\ 1 & -1 & -L \\ 1 & 1 & L \end{bmatrix} \begin{bmatrix} V_x \\ V_y \\ \omega_z \end{bmatrix} \tag{2.4}$$

donde, ω_i, $i(1, 2, 3, 4)$ es la velocidad angular de la rueda *mecanum* y V es la velocidad lineal del robot móvil en las coordenadas $(x,\ y,\ z)$. Esta expresión puede expresarse de la siguiente manera.

$$\begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \end{bmatrix} = \begin{bmatrix} R\omega_1 \\ R\omega_2 \\ R\omega_3 \\ R\omega_4 \end{bmatrix} = \begin{bmatrix} 1 & 1 & -L \\ 1 & -1 & L \\ 1 & -1 & -L \\ 1 & 1 & L \end{bmatrix} \begin{bmatrix} V_x \\ V_y \\ \omega_z \end{bmatrix} \tag{2.5}$$

Dado que se necesita conocer la velocidad angular de la base robótica omnidireccional desde un punto p central de la base móvil, se utiliza la siguiente expresión.

$$V_r = \sqrt{V_x^2 + V_y^2} \tag{2.6}$$

donde, V_r, es la velocidad angular obtenida durante el desplazamiento rotacional del robot en los ejes $(x,\ y)$. Realizando la operación con las velocidades lineales $V_x,\ V_y$ en las dos componentes del plano donde esté el robot *mecanum*.

2.3. Navegación en robots móviles

La navegación en robots móviles es una acción en la que a partir de un nodo o punto de inicio, el robot debe ser capaz de desplazarse a una meta establecida, tomando en cuenta que el robot tendrá que evadir cualquier obstáculo que se encuentre en dicho escenario. En un robot móvil esta tarea comprende una localización espacial propia y la planeación que debe realizar para llegar a la meta, de este modo el robot requiere de una representación del ambiente e interpretación del mismo. En la robótica móvil esta tarea queda unificada por las siguientes 3 etapas *(de Dios Flores Méndez, 2015)*.

- Percepción del mundo: Se refiere a la creación de mapas a partir de la información de un sensor.
- Planificación de ruta: Este cálculo de trayectoria está divido por una identificación de posición y una generación de la ruta a partir del mapa creado.
- Seguimiento de la trayectoria: En esta etapa el movimiento del robot es generado a base de la trayectoria y realizando un control de velocidades y posicionamiento en cada instante de tiempo.

2.4. Odometría visual

La odometría visual es una técnica que se emplea para estimar la posición de un robot en un espacio, mediante la implementación de una cámara monocular o de tipo estereoscópica. Dicha técnica está constituida por 5 etapas para generar la odometría visual. En primera instancia se obtiene una imagen con la cámara instalada en robot móvil, la siguiente etapa corresponde a la detección de características de la imagen captada, son métodos como los de detectores de puntos clave en una imagen, en la tercera etapa se realiza un emparejamiento de las características en imágenes sucesivas, lo siguiente es tomar estas parejas de puntos y remover aquellos que se formen parte de ruido o asociaciones incorrectas entre características. En la etapa final se concatenan las posiciones encontradas por cada par de imágenes sucesivas y que donde sus puntos resulten correctos, para obtener una trayectoria o conjunto de poses para desplazarse *(A. P. González, A. P. González)*.

En la navegación visual mediante visión estéreo la etapa final de estimación de movimiento con utilización de los puntos correctos obtenidos en cada par de imágenes sucesivas se emplea la estimación 3D-3D el cual utiliza la información de un arreglo de dos cámaras, en específico la profundidad para cada punto de la imagen visualizada en un mapa de profundidad como se visualiza en la siguiente Figura 2.2.

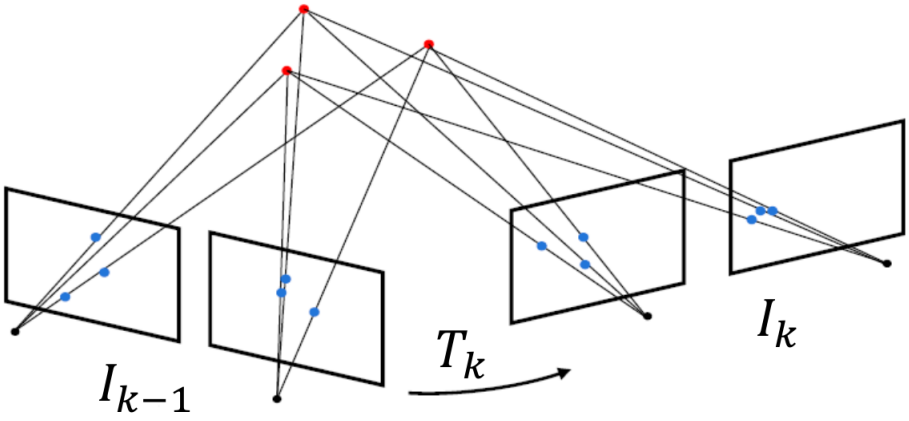


Figura 2.2: Correspondencia de características entre dos imágenes sucesivas para el caso de estimación 3D-3D (*A. P. González, 2015*).

En este caso cada par de imágenes f_{k-1} f_k están especificadas en tres dimensiones. El movimiento de la cámara queda descrito por la matriz de transformación T_k. En la visión estéreo cada imagen 3D I_k está descrita por dos imágenes 2D $I_{izquierda}$ e $I_{derecha}$ donde se puede generar una triangulación

2.5. Navegación inercial

En la navegación inercial se emplean sensores los cuales aprovechan la propiedad de la inercia, es decir, la resistencia a un cambio de momento, como por ejemplo movimiento angular en el caso de un giroscopio y cambios en movimiento lineal para un acelerómetro. En general este tipo de sensores se acopla una IMU la cual junto con una computadora dedicada es capaz de obtener y procesar este tipo de datos. Estos datos se pueden trabajar de diferentes maneras, es decir, si se tienen mediciones de aceleración lineal **a**, para tres ejes ortogonales, se puede obtener la velocidad lineal del objeto $\dot{\mathbf{p}}$ mediante la integración de esta aceleración con respecto al tiempo y para obtener la posición, **p** se integra con respecto al cambio de la velocidad *(Castro, 2019)*.

$$\mathbf{p} = \int \dot{\mathbf{p}} dt = \int \int \ddot{\mathbf{p}} dt = \int \int \mathbf{a} dt \tag{2.7}$$

donde, **p**, $\dot{\mathbf{p}}$, $\ddot{\mathbf{p}}$ son vectores de aceleración, velocidad respectivamente. En la navegación es importante controlar la velocidad angular ω para ello se tiene que calcular la posición angular θ del robot para cada instante, para realizar esta integración se requiere un modelo para la representación de la orientación del sistema. Una forma de representar la implementación de este tipo de datos inerciales se describen en la Figura 2.3.

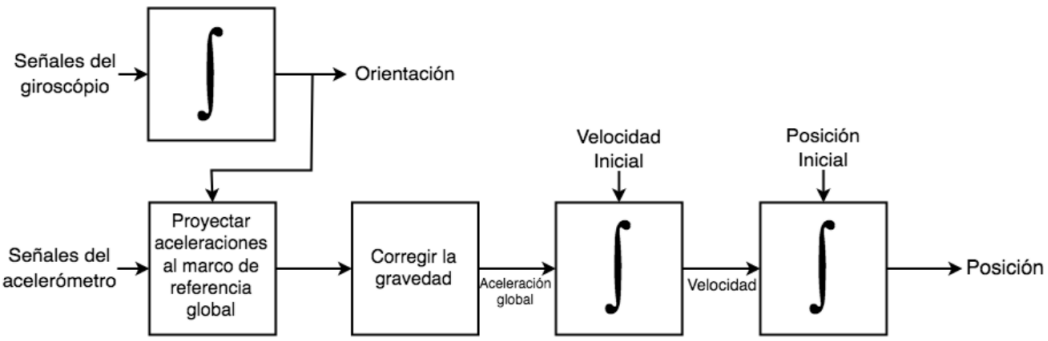


Figura 2.3: Diagrama de un sistema de medición inercial simplificado (*Castro, 2019*).

2.6. Teoría Difusa

Es una teoría propuesta por el científico en computación Lofti Zadeh en 1965, Lofti definió los conjuntos difusos como una clase de conjuntos con grados de pertenencia desde 0 a 1 *(Ponce-Cruz y cols., 2016)*. Esta teoría trata de modelar la incertidumbre que sen encuentra relacionada al razonamiento natural humano, el cual es expresado con palabras y oraciones lingüisticas a través de expresiones matemáticas, el razonamiento difuso es un razonamiento que no es exacto ni inexacto y para ello se deben comprender tres conceptos básicos:

- **Variable Lingüística (TS).** Variable cuyos valores son palabras u oraciones en un lenguaje natural o artificial en lugar de numérico. Por ejemplo, la estatura se puede describir como se presenta a continuación.

  TS(estatura)={Alto, Mediano, Bajo}={A,M,B}.

- **Proposición difusa**. Declaración expresada en un lenguaje natural o artificial. A diferencia de las proposiciones lógicas clásicas, puede adoptar un valor de verdad del intervalo [0, 1]. Por ejemplo, la temperatura es caliente.
- **Regla Lingüística.** Sentencia IF-THEN que se compone de dos partes: causa o proposición (IF), consecuencia o proposición difusa (THEN).

Un conjunto difuso es caracterizado por la función de pertenencia, la cual asigna a cada elemento un grado de pertenencia, el conjunto difuso F está definido por un conjunto de pares ordenados *(Abiyev y cols., 2019)*:

$$F = \{(x, \mu_F(x)) | x \in X\} \quad \mu_F \in [0, 1] \tag{2.8}$$

donde,x es un elemento del universo U y μ_F es la función de pertenencia que se asigna a un grado de pertenencia $\mu_F(x)$ para cada elemento x de F *(Mendel, 2001)*.

2.6.1. Conjunto Difuso Tipo-1

Usualmente el razonamiento humano en la toma de decisiones no es definido con métodos matemáticos, así que, los números difusos pueden ser usados para resolver problemas sencillos y avanzados que lidian con condiciones ambiguas. Los conjuntos difusos se utilizan para describir la falta de claridad en función de los grados de pertenencia y se pueden usar en muchas situaciones reales con términos lingüísticos *(Ponce-Cruz y cols., 2016)*.

Diagrama de bloques de los sistemas de Lógica Difusa Tipo-1

Un sistema de Lógica de Difusa Tipo-1 basado en reglas contiene cuatro componentes como se muestran en la Figura 2.4.

- Fuzzificación: convierte un valor de lectura puntual a un dominio difuso por medio de la función de pertenencia para posteriormente activar el sistema de inferencia.
- Base de conocimiento: los antecedentes y los consecuentes se relacionan entre sí por las funciones de lenguaje IF-THEN almacenadas en la base de reglas canónicas, utilizando las reglas necesarias de forma coherente; además, determina la las formas, cantidad y umbrales de los conjuntos que se trabajaran para cada variable.
- Inferencia: este bloque asigna a la entrada difusa una salida difusa de acuerdo con las reglas establecidas y operadores.
- Defuzzificación: asignación de una entrada difusa de tipo 1 en una salida puntual o en forma de palabra.

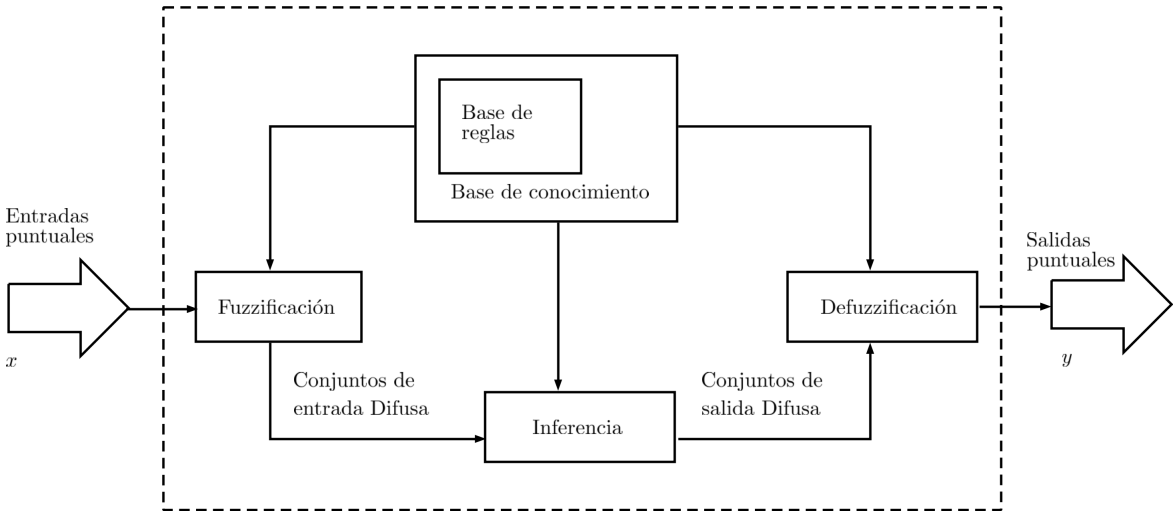


Figura 2.4: Diagrama a bloques de la Lógica Difusa Tipo-1 (*Mendel, 2001*).

2.6.2. Lógica Difusa Tipo-2

En algunos contextos es posible identificar fuentes de incertidumbre asociadas al lenguaje natural, donde se puede ponderar dicha incertidumbre con grados de pertenencia a su vez difusos, considerándolos de esta manera como incertidumbre difusa Tipo-2. En general, un conjunto difuso Tipo-2 es una generalización de un conjunto difuso en cuanto se asocia a una fuente secundaria de incertidumbre relacionada con la definición de una palabra o conjunto A. Dicha fuente adicional de incertidumbre se presenta como función de pertenencia secundaria asociada a cada valor de $x \in X$(o universo de discurso), la representación de la Lógica Difusa Tipo-2 se muestra en la Figura 2.5 *(Rodríguez y Huertas, 2016)*.

$$\hat{A} : X \to F([0,1]) \tag{2.9}$$

donde, $F([0,1])$ es el conjunto de todos los conjuntos difusos Tipo-1 que pueden ser definidos dentro dentro del universo $x \in X$.
Los conjuntos difusos Tipo-2, cuya función de pertenencia secundaria es un intervalo, es decir, la incertidumbre asociada al lenguaje es tratada de manera equitativa, estos son llamados *conjuntos difusos Tipo-2 de intervalo*, ya que los grados de pertenencia asignados a los elementos del universo son un intervalo.
Los conjuntos difusos Tipo-2 de intervalo son expresados por dos funciones de pertenencia donde una representa el grado de pertenencia en X y la otra da una ponderación a cada uno de los conjuntos difusos Tipo-1, definido de la siguiente manera.

$$\tilde{A} = \{(x, \mu_{\tilde{A}}(x)) | x \in X\} \tag{2.10}$$

donde $\mu_{\tilde{A}}(x)$ es el grado de pertenencia de x en $\tilde{A}$ y $\mu_{\tilde{A}}$ es la función de pertenencia asociada al conjunto $\tilde{A}$, que define el grado al que pertenece cada elemento del universo X perteneciente al conjunto difuso Tipo-2.
Por otra parte, el conjunto difuso primario J_x esta ponderada por el conjunto difuso $f_x(u)$ como una función de pertenencia secundaria.

$$\tilde{A} = \{((x,y), J_x, f_x(u)) | x \in X; u \in [0,1]\} \Leftrightarrow f_x(u) = 1 \tag{2.11}$$

donde x es la variable primaria, J_x es un intervalo $[0,1]$ que representa la pertenencia primaria de x, u es la variable secundaria y $f_x(u)$ es la función secundaria de x.

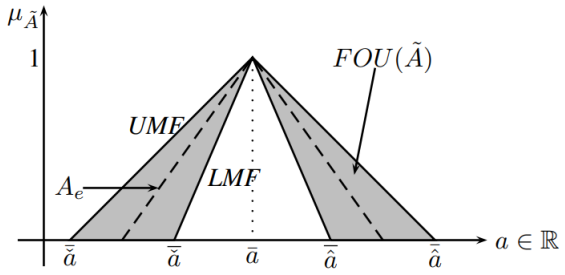


Figura 2.5: Conjunto Difuso Tipo-2 *(Rodríguez y Huertas, 2016)*.

La huella de incertidumbre (FOU) de los conjuntos difusos Tipo-2 define la incertidumbre de $\tilde{A}$ como la unión de todas las pertenencias primarias J_x.

$$FOU(\tilde{A}) = \bigcup_{x \in X} J_x \tag{2.12}$$

Esta se encuentra limitada por dos funciones de pertenencia: función de pertenencia superior $\bar{\mu}_{\tilde{A}(x)}$ (UMF) y función de pertenencia inferior $\underline{\mu}_{\tilde{A}(x)}$ (LMF) donde son conjuntos de Tipo-1 y tienen e conjuntos empotrados A_e; en consecuencia el grado de pertenencia de cada elemento de un conjunto difuso Tipo-2 es un intervalo $[\underline{\mu}_{\tilde{A}(x)}, \bar{\mu}_{\tilde{A}(x)}]$

Diagrama a bloques de sistemas de Lógica Difusa Tipo-2

En la Figura 2.6 se presenta el diagrama a bloques, el cual es similar a un sistema de Lógica Difusa Tipo-1, con la diferencia de que se trabaja con dos funciones de pertenencia y se introducen los siguientes componentes.

- Reducción de tipo: En algunos sistemas se requiere transformar las salidas difusas de Tipo-2 al motor de inferencia Tipo-1 y el conjunto se denomina reducción de tipo.

- Defuzzificación: Una vez que las salidas se han reducido, la defuzzificación determina el valor puntual que se introducirá en algún actuador.

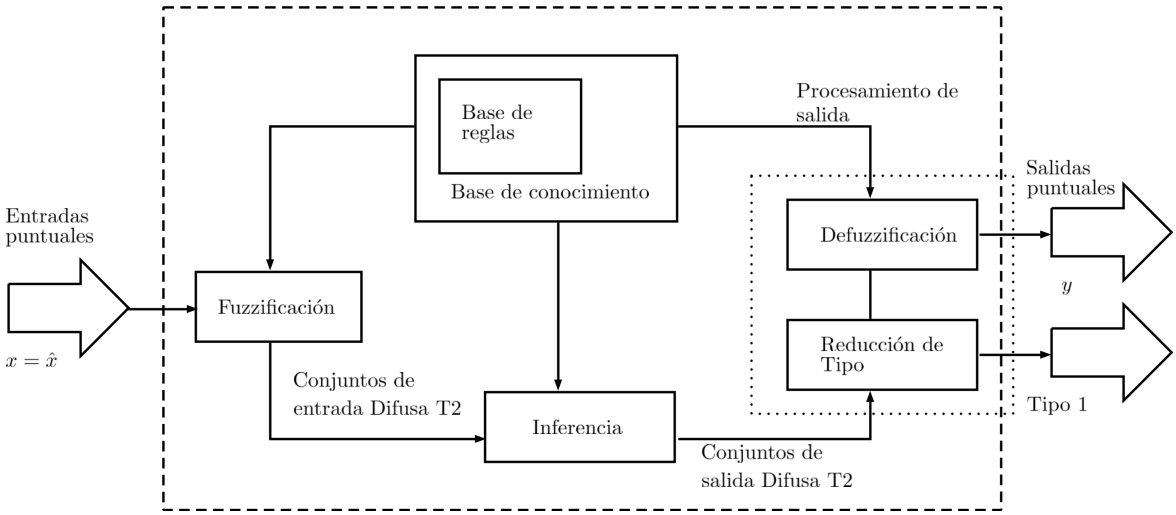


Figura 2.6: Diagrama a bloques de un sistema de Lógica Difusa Tipo-2 *(Mendel, 2001)*.

2.7. Planificación de trayectoria

La planificación de trayectorias es una tarea que se enfoca en encontrar el camino o linea de un punto inicial a una meta en un espacio libre. Este tipo de algoritmo utiliza un mapa conocido por el robot para realizar este cálculo. El robot emplea este tipo algoritmo para alcanzar de forma ordenar y precisa un nodo objetivo en específico y no solo en realizar desplazamientos aleatorios con respecto a los obstáculos del escenario como algunos sistemas de navegación que no buscan probar la generación de trayectorias en robots móviles *(Klančar y cols., 2017)*.

El algoritmo planificador de trayectorias calcula el camino más factible para evitar colisión con algún obstáculo en el escenario, por lo general se obtienen n cantidad de trayectorias en un escenario cada algoritmo planificador tiene su criterio para determinar la ruta con menos distancia entre un punto inicio y una meta. Existen diversas opciones de selección en la que un algoritmo planificador optimiza una trayectoria, algunas de esas opciones son.

- La longitud de la trayectoria es la más corta de un nodo a otro.
- El camino adecuado es aquel en el que el robot puede recorrerlo en el menos tiempo.
- El nodo meta debe encontrarse lo más lejos de un obstáculo.
- Una ruta debe ser lo más recta sin tener rotaciones o giros bruscos.

- Una trayectoria debe considerar restricciones de movimiento, lo cual considera que no todas las direcciones son posibles o no-holonómicas.

Los planificadores de trayectorias pueden categorizarse en dos grandes grupos, los planificadores geométricos o los basados en control.

Los planificadores geométricos toman en cuenta restricciones geométricas del entorno y cinemáticas del sistema de navegación. Consideran que cualquier ruta puede convertirse en una trayectoria dinámicamente factible. Un ejemplo de este tipo de algoritmo es el método de ruta probabilística o PRM, el cual emplea una búsqueda de rutas entre puntos o nodos de un escenario *(Klančar y cols., 2017)*, como se muestran en la Figura 2.7.

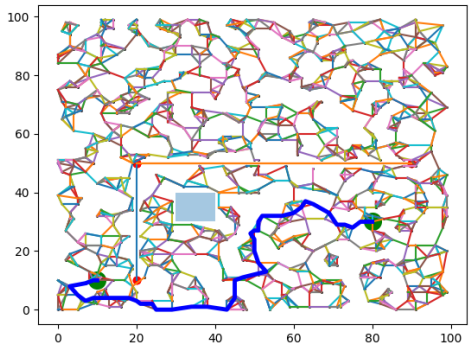


Figura 2.7: Representación gráfica de una navegación con el planificador PRM *(KaleabTessera, 2019)*.

El algoritmo maneja dos etapas. En la primera construye un esquema donde se distribuyen los nodos en todo el mapa esto son colocados en el orden aleatorio que un algoritmo de inicialización genere, la segunda etapa el nodo inicio donde se encuentra el robot conecta a la meta utilizando un método de búsqueda con la restricción de la distancia más corta entre nodos generados.

Por otro lado, los planificadores basados en control consideran restricciones diferenciales, se basan en la propagación del estado en lugar de generar una simple interpolación para generar movimientos. Este tipo de planificadores no requieren una función de dirección, pero todos algunos la emplean si el usuario lo requiere.

2.8. Sistema Operativo Robótico

El Sistema Operativo Robótico es un sistema con una amplia colección de herramientas y librerías que tienen como objetivo facilitar el desarrollo de sistemas robóticos permitiendo crear robots complejos y de tareas robustas dentro de una amplia variedad de plataformas robóticas. Su idea básica es hacer que la codificación de proyectos robóticos sea reproducible,

modificable y escalable, de esta forma se puede emplear en otra plataforma robótica, claro adecuando las necesidades del proyecto *(Martinez y Fernández, 2013)*.

Este Sistema Operativo Robótico está basado en una arquitectura de nodos con una topología centralizada sobre un nodo maestro, estos nodos pueden mandar o recibir información de diferentes tipos de *hardware* en especial de sensores, actuadores, controladores y planificadores de trayectorias. La comunicación se establece utilizando tópicos, el cual es el encargado de transportar mensajes entre nodos, una representación simple de esta arquitectura se muestra a continuación en la Figura 2.8.

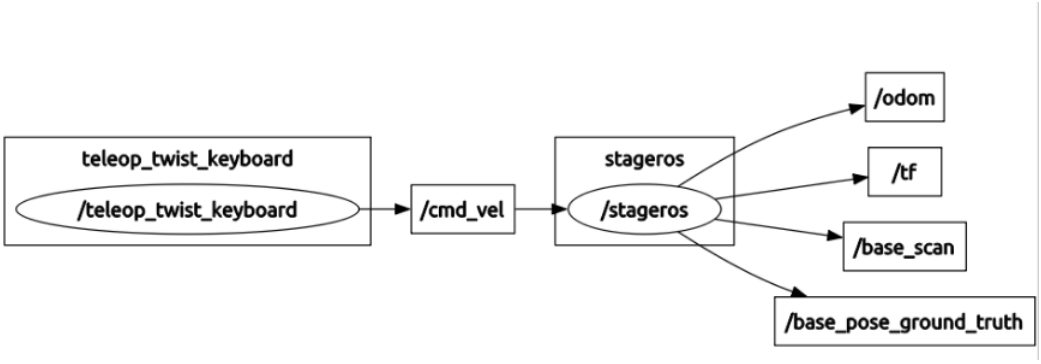


Figura 2.8: Transferencia de información entre nodos *(Martinez y Fernández, 2013)*.

El sistema además proporciona herramientas para generación de sistemas simulados entre los que destacan Gazebo, V-REP, *Webots*, STDR y *Stage*. Así como también Rviz la cual funciona como un graficador de información sensorial tanto en sistemas simulados como en robots físicos. El implementar este tipo de herramientas ayuda a probar las aplicaciones robóticas para después pasar a la parte física.

2.9. Discusión

En este Capítulo se presentó un panorama general de temas relacionados con este trabajo de tesis, los cuales son importante conocer para comprender el objetivo a que se llegó con el trabajo, de los que podemos mencionar como la formalización matemática para la cinemática del robot omnidireccional, el entender el concepto principal que es la navegación y el control que es una tarea esencial para mantener unificada sus etapas, la técnica que se empleó para desarrollar el control que se basa en Lógica Difusa Tipo-2. Añadiendo que el implementar un sistema como ROS en el desarrollo de este robot omnidireccional es donde todo se monta y comunica entre sí.

La información presentada es fundamental para entender cada una de las etapa de la metodología solución que se plantea en este trabajo, desde la adquisición, el sistema de inferencia e implementación sobre una plataforma robótica.

Capítulo 3

Estado del Arte

3.1. Antecedentes

Los trabajos que se muestran a continuación son aquellos que se desarrollaron en el Centro Nacional de Investigación y Desarrollo Tecnológico (CENIDET) y abordan temas relacionados con este trabajo de investigación.

Evaluación de las técnicas SLAM disponibles en ROS *(Borreguero, 2017)*

En este trabajo de maestría, se tuvo como objetivo cumplir el estudio comparativo de las metodologías de localización, mapeo y navegación en ROS. Se utilizaron algoritmos como: *Gmapping*, *HectorSLAM* y *CRSM SLAM*, estos algoritmos se enfocaban a mapeo y localización simultáneo, mientras que el paquete *Frontier Exploration* trabajaba la navegación del robot, junto con la telometría.
Los resultados obtenidos señalaron a *Gmapping* como la técnica SLAM que mejor se desempeñó en la generación de mapas aproximados a entornos reales, del mismo modo tuvo la capacidad de recuperar la pose del robot en la mayoría de casos.
La técnica *Gmapping* se basa en un Filtro de Partículas *Rao-Blackwellized* para el aprendizaje de mapas, el cual toma una primera observación, para proponer una distribución de campana. El Filtro de Partículas *Rao-Blackwellized* utiliza la factorización del propio filtro para realizar el mapeo y localización, matemáticamente expresado en la siguiente ecuación.

$$p(x_{1:t}, m | z_{1:t}, u_{1:t-1}) \tag{3.1}$$

donde, x indica la trayectoria del robot, m el mapa, z las observaciones y u las mediciones de odometría. Para generar la odometría se realiza una factorización que permite estimar la trayectoría y justo después el mapa, dada esa trayectoria.

Evaluación del algoritmo Theta* para planeación de trayectorias *(Fernández, 2019)*

En este documento se presentó la implementación del algoritmo Theta*, sobre ROS para planificación de trayectorias, del mismo modo una comparativa con A* y Dijkstra. Con el objeto de planificar trayectorias cortas en un tiempo de ejecución aceptable. El algoritmo fue probado sobre un robot de configuración diferencial, en 5 mapas diferentes y se evaluaron 27 trayectorias en función del tiempo con sus respectivas distancias generadas, obteniendo como resultado del 97 % de las trayectorias y metas alcanzadas en el 80 % de los casos.

La implementación del algoritmo para la planificación de trayectorias mostró expectativas altas para este segmento de la navegación autónoma.

Fusión de Información de Pose en Robots Móviles con Visión *(Olán, 2013)*

En esta tesis de maestría, se mostraron dos métodos de fusión de información para el mejoramiento de estimación de pose en un robot móvil que cuente con un sistema de visión, debido a la problemática que plantea sobre el menor grado de confiabilidad hacia un sistema visual.

El sistema de visión por computadora implementó una cámara estéreo donde dicha información fue utilizada para establecer una pose en robots móviles. Algunos de los métodos empleados fueron: Intersección de Covarianzas y el Filtro de Partículas.

Ambos métodos ayudaron en un 100 % en la estimación de la pose a través de la estimación visual. Resultando mejor el Filtro de Partículas sobre el de Intersección de Covarianzas.

Fusion de Datos de Sensores Mediante Lógica Difusa, Aplicación en Robótica móvil *(Arellano, 2014)*

Este trabajo de tesis, utilizó una fusión de datos sensoriales los cuales son usados en la navegación autónoma de robots móviles, en especial empleó datos inerciales y mecánicos mediante el uso de ecuaciones cinemáticas. Dado que la fusión de sensores trae problemas en la afinación de odometría, este trabajo se complementó con el uso de teoría difusa para generar esta fusión de información.

En esta tesis se planteó un esquema para minimizar los errores de odometría ocasionado por errores sistemáticos (como las diferencias entre diámetros de las ruedas) o no sistemáticos (plano de tierra o deslizamiento). La fusión sensorial consistió en insertar datos de sensores ultrasónicos, VGA, encoders y un acelerómetro para determinar la posición XY, sobre un sistema difuso de tipo Mandami dentro una simulación MATLAB-SIMULINK. Las reglas y combinaciones de la tesis fueron programadas en la interfaz de LabVIEW®.

Discusión

En este conjunto de tesis realizadas en CENIDET se localizaron enfoques al control robótico de robots con accionamiento diferencial, pero no dirigido a robots omnidireccionales. *(Arellano, 2014)* se enfocó a la fusión de sensores con teoría difusa para evitar problemas de odometría, utilizando robots de tipo diferencial, mediante un sistema de lógica difusa de tipo Mandami. *(Fernández, 2019)* realizó un trabajo enfocado a algoritmos de planificación de trayectorias, resultando en un análisis comparativo del algoritmo A* y Dijkstra con respecto al algoritmo Theta*.

No se encontraron trabajos relacionados con navegación autónoma tomando algún enfoque específico en Teoría Difusa de Tipo-2.

3.2. Trabajos relacionados

A continuación, se presenta una introducción de artículos recientes que representaron técnicas, métodos y/o algoritmos considerados relevantes y útiles para el desarrollo del tema de tesis.

Control de un robot omnidireccional usando el sistema *Z-Number* Difuso *(Abiyev y cols., 2019)*

En este artículo el problema se demostró una navegación autónoma y la planificación de trayectorias. Se propuso un sistema de inferencia difusa basado en *Z-Number* para un robot de tipo omnidireccional. Se implementó en el mecanismo de inferencia y en el de control.

Dicho mecanismo de inferencia hizo uso de las *Reglas Mandami* como planificador del comportamiento robótico en el módulo de toma de decisiones, Figura 3.1.

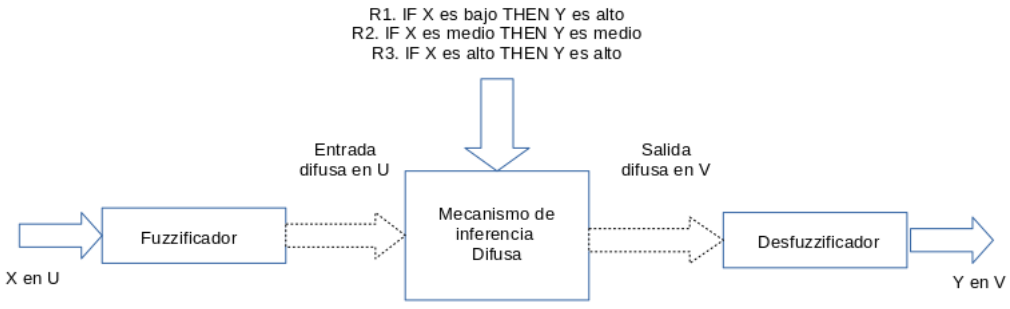


Figura 3.1: Sistema difuso Mandami *(Abiyev y cols., 2019)*.

Determina las trayectorias y se mueve basado el control PID difuso, utilizando el error

de seguimiento de posición e_p y el error de seguimiento de ángulo e_ω de dirección. Un número difuso A es un subconjunto de R tal que $\mu_A : R \rightarrow [0, w]$ los cuales satisfacen las siguientes propiedades *(Abiyev y cols., 2019)*.

1. μ_A es simi-continua superior.
2. $\mu_A = 0$ fuera de un intervalo $[a,\ b]$, donde a, b$\in R$

La función de pertenencia se formuló como.

$$\mu_A(x) = \begin{cases} \mu_A^L(x), & a \leq x \leq b \\ 1, & b \leq x \leq c \\ \mu_A^R(x), & b \leq x \leq c \\ 0, & otherwise \end{cases} \tag{3.2}$$

donde, $\mu_A^L(x) : [a, b] \rightarrow [0, 1]$ es continuo y estrictamente incremental, $\mu_A^R(x) : [c, d] \rightarrow [0, 1]$ es continuo y estrictamente decreciente.

Controladores en Lógica Difusa para la navegación de robots móviles *(Masmoudi y cols., 2016)*

En este trabajo se desarrolló un controlador integral proporcional (PI) pero implementando Teoría Difusa Tipo-1 (FuzzyPI), a la problemática de navegación en robots omnidireccionales sobre la planificación de trayectorias en rutas grandes y sus cambios de ángulos o direcciones. Durante las pruebas del controlador Difuso, demostró que el algoritmo FuzzyPI tiene complicaciones al controlar la velocidad lineal y angular del robot, por ello se optimizó las salida de ambas señales del controlador *Mandami* con un bloque *Fuzzy adaptive Tuner* ω resultando a la salida, la comparación que se muestra en la Figura 3.2.

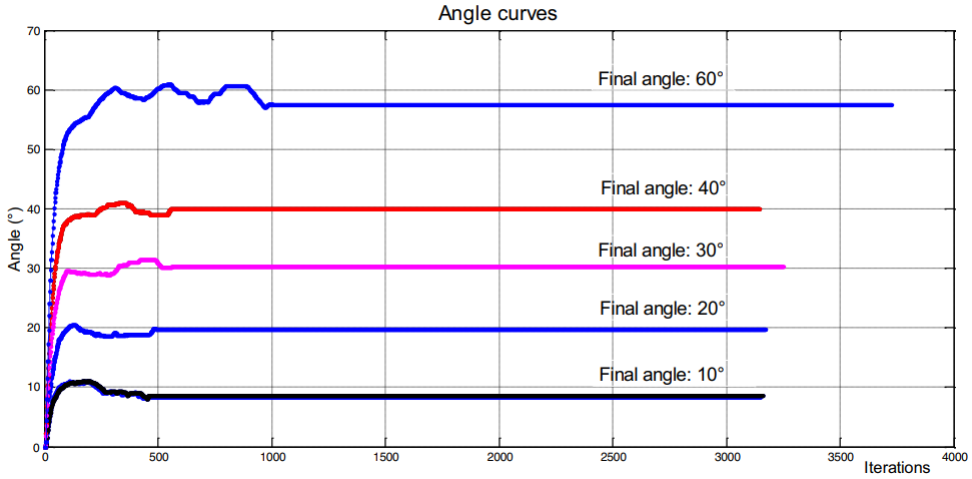


Figura 3.2: Controlador PI Difuso *(Masmoudi y cols., 2016)*.

En este artículo se emplearon conceptos de control robótico así como muestran el desempeño de los controles proporcional integral; visto desde el punto difuso. Al igual las desventajas de emplearlo, desarrollaron una propuesta de optimización para dicho controlador y muestran los resultados obtenidos. Tiene como utilidad conocer técnicas tradicionales de control hacia la optimización con Lógica Difusa.

Sistema embebido para el control de movimiento en un robot móvil omnidireccional *(Mamun y cols., 2018)*

En este artículo se propuso un planificador y controlador de trayectorias en robots omnidireccionales, enfocándose a la problemática de desplazamiento de este tipo de robots, dicho control fue implementado en un sistema embebido que se compone de una tarjeta FPGA que se encarga de la cuadratura de decodificadores y la comunicación del robot omnidireccional. El robot no se manejó con datos odométricos, empleó sensores inerciales, tales como giroscopios y acelerómetros. Esto para ayudar a obtener las constantes de aceleración y velocidad, logrando una mejor navegación. Este sistema cuenta con dos partes:

- Sistema integrado con comunicación a un servidor remoto utilizando un enlace inalámbrico.
- Una ruta proporcional integral (PI) ajustada difusamente.

El contenido de dicho articulo se puede utilizar como referencia para el manejo tecnologías más sofisticadas como la FPGA y demás que se desempeñen mejor en el control.

Control Difuso para un robot omnidireccional *(Abiyev y cols., 2017)*

En este artículo se trató la problemática de la navegación en robot omnidireccionales de 4 ruedas debido a que los desplazamientos suelen ser caóticos si se evalúa con respecto al tiempo, debido a la carga de error en el control, se propone la implementación del método *Z-Number* en un controlador PID para el tratamiento de señales como velocidad lineal y angular. El método *Z-Number* utiliza un enfoque difuso, cada regla sigue una composición *max-min*, como se muestra en la siguiente fórmula.

$$\mu(y) = \max_{x_1,x_2} \min\{\mu_{x_1}(x_1), \mu_{x_2}(x_2), \mu_R(x_1, x_2, y)\} \tag{3.3}$$

Del mismo modo, el método para la defuzzificación de la salida con la siguiente expresión matemática.

$$y = \frac{\sum_{i=1}^{n} \mu(y_i) * y_i}{\sum_{i=1}^{n} \mu(y_i)} \tag{3.4}$$

donde, y es la salida defuzzificada y $\mu(y_i)$ indica la salida del sistema difuso. Se puede tomar como comparativa contra el algoritmo de navegación propuesto. Al igual que funciona para conocer más a detalle el funcionamiento de un control difuso omnidireccional.

Navegación autónoma basada en maniobras bajo estimación de posturas humanas para un robot omnidireccional KUKA Youbut *(Guamani, 2019)*

En este trabajo su objetivo general fue implementar un sistema de navegación autónoma bajo estimación de posturas humanas, en este trabajo la problemática estaba enfocado a la autonomía ya que este tipo de robot en la industria se desempeña bajo el control humano para tareas muy específicas, se busca la implementación de un aprendizaje controlado tomando como base el seguimiento de objetos para el control y navegación del robot es por ello que se toman las posturas humanas a seguir como trayectoria de referencia, emplea detección y reconocimiento de objetos como técnicas para la adquisición de información.

Las tecnologías que se emplearon fueron en un entorno robótico, es decir, ROS ya que cuenta con las paqueterías y librerías necesarias para trabajar este tipo de proyecto. Uno de los métodos empleados para la navegación a partir del reconocimiento y detección de puntos clave es el algoritmo SURF el cuál es invariante a la escala, este tipo de algoritmos se desempeña eficientemente a entornos no controlados. Lo que se puede obtener de este trabajo es la implementación de algoritmos para la detección de objetos.

Control de seguimiento de trayectoria de retroalimentación de salida de un robot móvil omnidireccional *(Andreev y cols., 2019)*

Este artículo implementó algoritmos de planificación o trazado de trayectorias para robots omnidireccionales donde, el problema principal a resolver fue el control de un robot omnidireccional de 3 ruedas durante la navegación en una trayectoria planificada en el cual se destaca la ventaja de los desplazamientos en cualquier dirección a diferencia de un robot de tipo diferencial. El robot de accionamiento diferencial debe rotar para cambiar su dirección de desplazamiento. La forma en que resuelve este problema es la aplicación de un principio de cuasivarianza para sistemas autónomos de ecuaciones diferenciales.

Definición 5: La cuasivarianza de una muestra aleatoria simple es un estimador insesgado de la varianza poblacional *(C. G. González y Felpeto,2006)*.

La propiedad global de estabilidad asintótica uniforme del conjunto de todos los puntos de equilibrio del sistema de circuito cerrado se ha demostrado mediante la construcción de un controlador con retroalimentación en la salida. Dicho de otra manera, implementa este tipo de circuito ya que le proporciona un re-ajuste de la pose en un robot omnidireccional.

Uso de un Controlador Neuronal Difuso Tipo-2 para el control de navegación de robots evolutivos *(Lin y cols., 2019)*

En este trabajo se realizó un enfoque en el control de robots móviles, por el hecho de que existen algoritmos que en un cierto lapso de tiempo no son óptimos para controlar la trayectoria planificada, es por ello que se plantea la problemática de la utilización de controladores poco eficientes en sistemas robóticos diferenciales, para ello se propone en el artículo un algoritmo de optimización de enjambre para control y evasión de obstáculos. En el trabajo se tuvo como entradas la distancia entre la pared del robot (d) y los sensores de sonda (S^b), la salida es la velocidad para ambas ruedas. El algoritmo que optimizan en esta publicación fue el PSO a una versión DGPSO el cual manejó mejor los valores de velocidad angular, con el siguiente modelo matemático.

$$V_i(n+1) = \omega x V_i(n) + C_1 x rand_1 x (P_i - X_i(n)) + C_2 x rand_2 x (P_{G_{best}} = X_i(n)) \tag{3.5}$$

$$X_i(n+1) = X_i(n) + V_i(n+1) \tag{3.6}$$

donde, $j = 1, 2, ...M$ es el número de la regla en campo de inferencia difusa $i = 2, ...n$ es el número (máximo) de entrada en la representación visual. $V_i(n)$ representa la i-ésima partícula, $P_{G_{best}}$ es el mejor valor global, ω representa el peso inercial, C_1 indica el parámetro cognitivo, C_2 representa el parámetro social, y $rand_1$, $rand_2$ son números aleatorios en $[0, 1]$.

Control de un robot futbolista usando arboles de comportamiento y Lógica Difusa para la 12^a Conferencia Internacional sobre la Aplicación de Sistemas Difusos y Computación Flexible *(Abiyev y cols., 2016)*

En este artículo se enfocó a la problemática de la toma de decisiones en robots para competencias de *soccer* presentando como propuesta un algoritmo basado en árboles, teniendo como objetivo diseñar e implementar un control robótico. Tomando decisiones frente a obstáculos durante su navegación. En la utilización de árboles de decisión, se le da un enfoque difuso para tener más flexibilidad.

Adicionalmente el mismo enfoque difuso se centra en la planificación y seguimiento de la trayectoria, mediante reglas difusas si-entonces y un mecanismo de inferencia para encontrar la trayectoria óptima evitando los obstáculos. Como resultado en este artículo se obtuvo un control para robots de accionamiento omnidireccional en 4 ruedas, demostrando su efectividad en competencias como lo es el fútbol.

Algoritmo de navegación a bordo en ambientes controlados a partir de procesamiento de imágenes *(Barrero, 2016)*

En este artículo se atendió a la problemática de planificación de trayectorias en robots móviles de accionamiento diferencial, ya que un robot debe ser capaz de desplazarse eficientemente en entornos no-controlados y el primer paso es la navegación sobre espacios con un cierto número de obstáculos y limites de zona. Durante la construcción del entorno se utilizaron formas geométricas para definir los posibles obstáculos. Para ello se propone en el artículo, un algoritmo adecuado para la navegación con datos obtenidos de una cámara.

Mediante los datos obtenidos de la cámara se prosiguió a realizar un procesamiento para comunicar al sistema embebido y este realiza una decisión para esquivar el obstáculo o seguirlo hasta llegar un su meta. Se implementaron cálculos de odometría y generación de matrices de transformación para la toma de decisiones en robótica móvil.

El sistema embebido estuvo compuesto de una *Raspberry* Pi 2 Modelo B, ya que trabaja con una amplia compatibilidad de micro-cámara para proyectos de robótica.

Navegación autónoma de un robot móvil usando técnicas probabilísticas de localización y mapeo basados en métodos secuenciasles Monte Carlo *(Barrero, 2016)*

En este artículo se ejecutaron algunas técnicas de mapeo y localización simultáneo que suelen ser muy robustas en cuanto a procesamiento computacional y para robots industriales este tipo de técnicas ayuda en tener un control más específico del entorno ya que aprende conforme va navegando. Pero tiene complicaciones con forma transcurre el tiempo y son la acumulación de ruido durante la navegación, este tipo de ruido requiere de otros métodos de filtrado para controlar la planificación de la trayectoria y que su desplazamiento no se vea afectado durante largos periodos de tiempo. En el control de trayectorias que se propuso maneja algoritmos heurísticos y probabilísticos, como por ejemplo los métodos secuenciales de Monte Carlo, este tipo de algoritmos es muy utilizado en robots móviles ya que brinda un control al ruido que se puede generar durante el mapeo simultáneo de las técnicas SLAM. Como propuesta se establece una mejora a este algoritmo para reducir el error de estimación durante la navegación con técnicas SLAM y el algoritmo Monte Carlo.

Implementación de sistemas de navegación autónomo en robot móvil experimental para reconstrucción y exploración de entornos desconocidos *(Velázquez, 2016)*

Este trabajo se enfocó en la navegación autónoma en robots de tipo diferencial atendiendo a la problemática de control que se requiere en desplazamientos autónomos para tareas industriales utilizando robots móviles capaces de reaccionar ante cualquier obstáculo y

ambientes no controlados. En el artículo se emplearon técnicas de mapeo y localización simultánea (SLAM) para establecer un nivel de autonomía frente a trayectorias y evasión de obstáculos dinámicos y estáticos. Recordar que el uso de técnicas SLAM requiere un alto desempeño computacional, ya que realiza dos tareas de forma simultánea para tener un control durante la navegación en el entorno. Como resultado de la implementación de estas técnicas para resolver el problema de navegación en los robots diferenciales, se realizó un prototipo para mapear el interior de un centro de investigación y se realizó la navegación cargando los mapas ya obtenidos al grabar el instituto.

Control de un robot omnidireccional de ruedas *mecanum* con una estrategia basada en un sistema de Lógica Difusa Tipo-2 *(Cuevas y cols., 2020)*

En este artículo se tuvo como objetivo de mantener una ubicación y comportamiento específico en un robot omnidireccional mediante el uso de Lógica Difusa Tipo-2, se describe de manera teórica el sistema difuso como también sus conceptos básicos que lo componen dentro de esta teoría de conjuntos difusos, la experimentación del sistema se realizó de manera simulada sobre el software V-REP *Simulator* y Matlab-Simulink sobre el robot *YouBot* de KUKA *Robotics*. Se implementan ecuaciones de cinemática básica para robots omnidireccionales de 4 ruedas y un modelo difuso para control de velocidades en motores. El control está basado en odometría inercial y cinemática para realizar desplazamientos dentro del escenario simulado. En la Figura 3.3, se muestra el sistema de control difuso empleando la dos entradas $Error_x$ y *Change Error* y las salidas V_x y V_y.

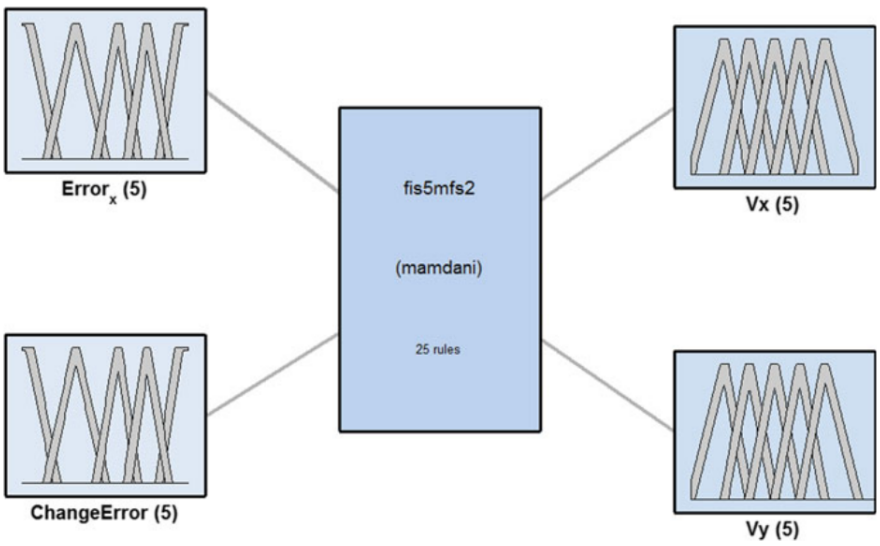


Figura 3.3: Sistema de intervalo difuso tipo-2 con dos entradas y dos salidas *(Cuevas y cols., 2020)*.

Control de trayectoria de un robot omnidireccional de servicio basado en un modelo de control predictivo de un sistema de inferencia neuro-difuso adaptativo *(Pang y cols., 2020)*

En este artículo se propuso e implementó un modelo de control predictivo (MPC) basado en un sistema de inferencia neuro-difuso adaptativo (ANFIS) para realizar el control de un robot omnidireccional de servicio y esto para el seguimiento de trayectorias, dentro del modelo se realiza un ajuste manual a la función de costo para obtener un resultado satisfactorio. Para mejorar el rendimiento y precisión del modelo se implementa un sistema neuro-difuso adaptativo para el ajuste del peso de la función y reducir el error en el proceso de seguimiento de trayectoria. Para realizar los experimentos de esta metodología solución al seguimiento de trayectorias se utilizó un entorno simulado. El diagrama a bloques de la metodología solución se muestra a continuación en la Figura 3.4.

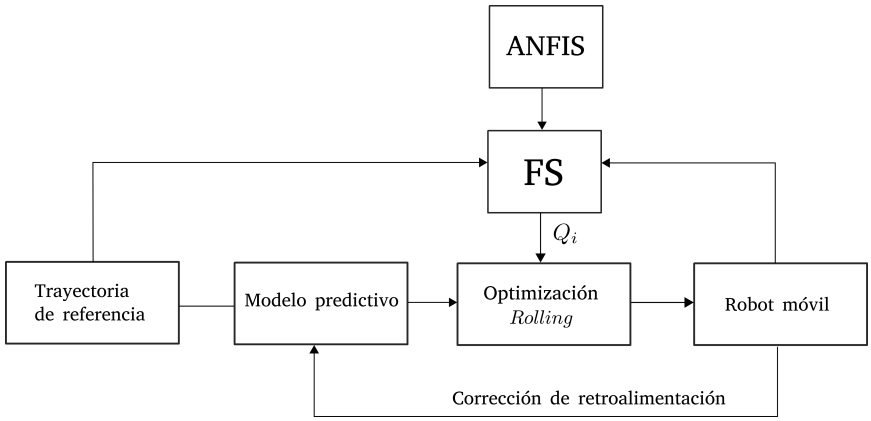


Figura 3.4: Diagrama del sistema de control *(Pang y cols., 2020)*.

donde Q_i es el peso ajustable para sistema difuso entrenado.

Odometría basada en visión para la localización de robots omnidireccionales en interiores *(Patruno y cols., 2020)*

En este artículo se propuso un sistema de localización de robots móviles en interiores, la plataforma que se implementó fue de tipo omnidireccional, dicho sistema tiene un enfoque basado en visión e implementando el uso de odometría visual es como se realizó la navegación en interiores. La odometría visual es capaz de devolver la pose relativa de un vehículo de guiado automático de tipo omnidireccional, empleando una cámara monocular casi perpendicular al suelo utiliza descriptores para visualizar puntos clave tales como SURF, SIFT, BRIEF los cuales detectan las características de las imágenes y mediante un filtrado basado en información estadística y geométrica se rechazan las coincidencias incorrectas captadas entre los puntos detectados de los fotogramas

consecutivos, de esta manera se extrae un mejor valor para la pose del robot dentro del escenario. El diagrama a bloques de dicho trabajo se representa a continuación en la Figura 3.5.

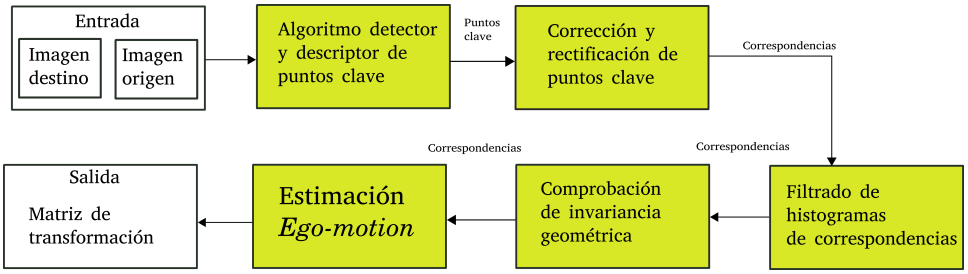


Figura 3.5: Diagrama del sistema de localización propuesto *(Patruno y cols., 2020)*.

OxIOD: Base de datos para odometría de inercia profunda *(C. Chen y cols., 2018)*

En este articulo se propuso la creación de una base de datos inerciales etiquetada, para el ámbito de la inteligencia artificial en el área de la robótica. Tiene como propósito ayudar a problemas de planificación de trayectorias en robot móviles donde, se requiere tener una base de entrenamiento para configurar sensores de tipo inercial. Se utilizaron datos de celulares ya que la mayoría cuenta con una IMU integrada, las pruebas se enfocaron a desplazamientos en diversas profundidades, es decir, se colocaron sobre varias alturas durante el desplazamiento, los objetos que sirvieron para definir la altura fueron,

- Dentro del bolsillo de pantalón.
- Sobre un carro de carga.
- Dentro de una mochila de mano.
- Sosteniendo el dispositivo mientras camina.

A partir de estos datos generados de 158 secuencias, sumando mas de 40km durante la navegación. Lo siguiente es el etiquetado inercial, a partir de las unidades de medición del sensor IMU (acelerómetros, giroscopios y magnetómetros). Este trabajo tuvo como objetivo el poder tener un conjunto de datos con lo que métodos como las redes neuronales puedan trabajar de mejor manera, al entrenar este tipo de información.

Seguimiento de trayectoria en Omni-Robot con el método de odometría *(Fahmizal y cols., 2019)*

En este artículo se presentó el seguimiento de trayectorias definidas para robot de accionamiento omnidireccional, utilizando el método de odometría. El sistema de odometría tiene como objetivo estimar la posición relativa a la posición inicial del robot, para

estimar los cambios de posición a lo largo del tiempo. El sensor del codificador giratorio se utiliza para contar el movimiento del robot omnidireccional en las coordenadas x e y en el proceso de cálculo de odometría. Se enfocan en el problema de control en la etapa de planificación de trayectorias, este artículo contiene los puntos teóricos y matemáticos para estabilizar el robot omnidireccional durante la navegación, logrando un error mínimo del 5 %. La posible utilidad de esta información funcionará para lograr el control del robot y evitar la propagación del error en lapsos de tiempo largos.

Investigación sobre co-simulación de plataforma móvil omnidireccional basada en ADAMS y Matlab *(Tang y cols., 2017)*

El problema que se trabajó en este tesis fue la navegación sobre una plataforma omnidireccional para implementando un control del desplazamiento holonómico, resolviendo a su vez la desventaja que tienen los robots de accionamiento diferencial al tener un gran radio de giro y un movimiento inflexible hacia cualquier dirección que se quiera desplazar, esto trae incapacidades con respecto a las otras direcciones al igual que las colisiones con obstáculos dentro del espacio, además de que hay la posibilidad de que no funcione completamente dentro del espacio, es por ello que plantea la solución utilizando robots omnidireccionales, los cuales no tienen la desventaja del desplazamiento, estos a su vez son más flexibles y libres para moverse hacia cualquier dirección. Ahora bien en el artículo lo que se propone es diseñar un modelo de robot omnidireccional en un entorno 3D con herramientas como Matlab y SolidWorks y mediante un control PID difuso lograr el control de las ruedas. Obteniendo como resultado una co-simulación para validar los resultados en plataformas físicas. Lo que se puede obtener de dicho artículo son las métricas de evaluación para evaluar el algoritmo que se propone con esta propuesta de tesis.

Seguimiento y dirección de un robot con ruedas *mecanum* utilizando control de lógica difusa *(Fahmizal y Kuo, 2017)*

Este artículo presentó algunos sistemas de seguimiento de trayectoria y ajuste de rumbo de un robot de rueda tipo *mecanum*, utilizó un controlador de lógica difusa (FLC) y un sensor de unidad de medición de inercial (IMU). Se sabe que cuando un robot de ruedas *mecanum* no se programa de forma correcta el algoritmo de control puede provocar que las ruedas resbalen. En esta condición de deslizamiento, las características como la posición y orientación del robot a menudo se desvían más de lo esperado. Para corregir el control y así no afectar el movimiento en las rueda *mecanum* el sensor IMU proporciona la medida para ajustar la dirección del robot. La implementación de la IMU como sensor de orientación generalmente se logra usando un filtro complementario. El FLC, por otro lado, se requiere forzar el robot de rueda *mecanum* a seguir una trayectoria

dada una desviación mínima de orientación. Como resultado el sistema FLC y el sensor IMU proporcionan un esquema de navegación robusto. La implementación de la propuesta solución en este artículo se logró utilizando un microcontrolador incorporado para el control de movimiento y la adquisición de la información mediante datos IMU. Lo utilizable de este artículo son varios puntos, tales como métricas de evaluación como diseño y un sistema difuso.

Simulación de modelado de robot con ruedas *Mecanum* basada en software y control difuso *(Jamali y cols., 2017)*

En este artículo se planteó un control en robots de tipo omnidireccional utilizando SIMULINK y Matlab para crear el sistema PID difuso. Este diagrama a bloques representa el control y ajuste que se realizó para cada una de las cuatro ruedas *mecanum* se tiene tres sistemas difusos, cada uno con una entrada y una salida que continúa con una compuerta que une estas señales que se procesan mediante el uso de una función de activación para asignarle la velocidad angular a cada rueda del robot omnidireccional. El aporta las métricas necesarias, debido a que solo es una simulación y no tiene la finalidad para algo físico, la parte teórica resulta útil. En la Figura 3.6 se muestra el diagrama a bloques de sistema propuesto para la simulación

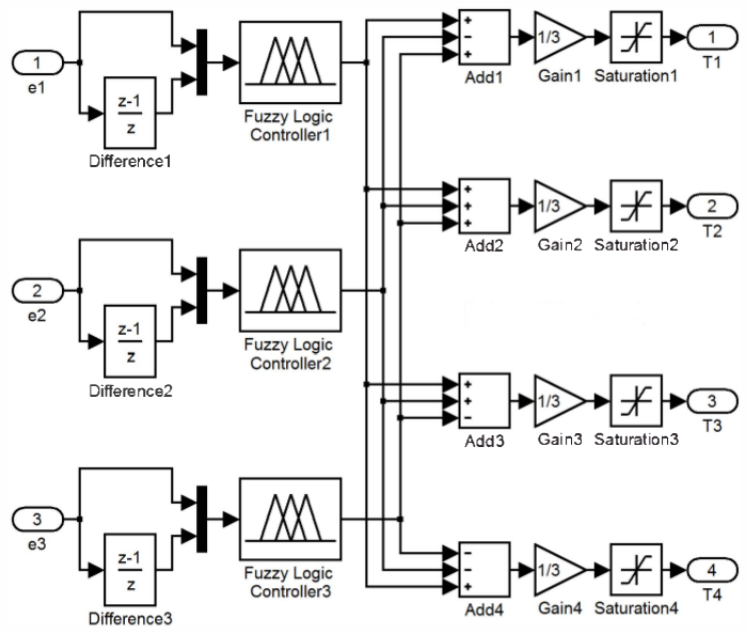


Figura 3.6: Diagrama a bloques del controlador difuso tipo Mamdani *(Jamali y cols., 2017)*.

Simulación de la navegación de un robot omnidireccional con ruedas *mecanum* basada en el algoritmo A* en Unity 3D *(Li y cols., 2019)*

En este artículo se trabajó una aplicación simulada de un robot omnidireccional, resulta útil ya que cuenta con la formalización matemática y física para configurar una plataforma omnidireccional, puesto que usa el algoritmo A* para el enfoque de planificación de trayectorias, las métricas para evaluar la navegación de este tipo de robots resulta de gran ayuda. Este artículo al igual que los anteriores tiende a mirar al enfoque de desplazamiento en robots de móviles e implementan un robot omnidireccional simulado en Unity3D con el propósito de generar una configuración y navegación con este tipo de plataforma robótica adaptando un algoritmo A* para la obtención de las trayectorias. En el artículo se implementó una mejora del algoritmo A* para desempeñar una mejor navegación ya que este tipo de robot tiende a requerir un buen control en sus desplazamiento para no generar derrapes durante su navegación. Se tomarán las métricas de evaluación así como también se realizará la comparación de dicho sistema, con el propuesto en este trabajo de tesis.

3.2.1. Discusión del Estado del Arte

En la mayoría de los artículos se propusieron métodos de control, planificación, navegación, mapeo y localización sobre plataformas de robots móviles, del mismo modo las propuestas se enfocaban en robots de accionamiento omnidireccional. Cada uno proporciona enfoques distintos a las etapas de la metodología solución: Adquisición de información del entorno, algoritmos implementados y tipo de robot móvil implementado. Una breve discusión de estos temas, a continuación.

- Adquisición de información para el control robótico:
  En la mayoría de los artículos se obtenía la información mediante cámaras y procesada con algoritmos de puntos clave en una imagen origen y destino, sensores infrarrojos, los cuales realizaban una detección en profundidad de los obstáculos que podrían encontrarse en la trayectoria. Así mismo otros emplearon, técnicas de triangulación para calcular la distancia aproximada con respecto al obstáculo visualizado. Algunos otros emplearon técnicas sin un enfoque de visión computacional, sino que mediante sensores de tipo ultrasónico, acelerómetros y giroscopios, cada señal era procesada de distinta manera pero se tenía un fin único y era la estimación de la pose del robot con respecto a un sistema de coordenadas globales.
- Algoritmos, métodos o técnicas implementadas:
  La mayoría de los algoritmos que se emplearon en los artículos consultados fueron acerca de Lógica Difusa desde la Tipo-1 a Tipo-2, así como también redes neuronales difusas para realizar un control más robusto y preciso. Otros autores optaron

por técnicas de mapeo y localización simultáneo comúnmente conocidas como SLAM, estas técnicas resultaron ser precisas en tareas de navegación teniendo un buen manejo del ruido que se puede generar, pero con la deficiencia de que eran sistemas automatizados y no autónomos y bastante complejos. También se utilizaron técnicas más simples pero eficientes como las redes neuronales, en robots diferenciales.

- Plataformas de *Hardware*:
  El tipo de robot móvil más empleado fue el diferencial, junto con algunos de tipo omnidireccional.
  Las plataformas empleadas en la mayoría de publicaciones eran microcontroladores como lo son las tarjetas *Raspberry* y Arduino UNO, sobre estas plataformas se trabajaban controles para robots diferenciales. Otros optaban por sistemas embebidos de mayor escalabilidad de procesamiento como las GPU de NVIDIA junto con sensores de tipo LIDAR. Así también empleaban tarjetas FPGA para procesos paralelos haciendo uso del paradigma de la Lógica Difusa, en los artículos se propusieron varios sistemas difusos basados en el modelo de tipo *Mandami*.
- Entornos de navegación. En cuanto a los espacios de prueba, la mayoría era sobre industrias o instituciones educativas, los cuales eran espacios cerrados con áreas amplias para desplazarse, no se encontraron prototipos funcionando en exteriores, por factores dinámicos dentro del entorno que afectara la navegación del robot.

Tabla 3.1: Resumen del Estado del Arte

Articulo	Objetivo	Método / Algoritmo	Resultado	Utilidad
Algoritmo de navegación a bordo en ambientes controlados a partir de procesamiento de imágenes *(Barrero,2016)*.	Implementar un algoritmo para la navegación con cámara a bordo en un sistema digital.	Una metodología basada en redes neuronales de tipo propagación hacia atrás.	Generación de trayectorias, realización y clasificación de superficies típicas.	Referencia para la planificación de trayectorias y evaluación de los algoritmos para la navegación.
Navegación autónoma basada en maniobras bajo estimación de posturas humanas para un robot omnidireccional Kuka youbot *(Guamani,2019)*.	Implementar un sistema de navegación autónoma bajo estimación de posturas humanas.	Una metodología basada en algoritmos de detección de posturas humanas en un entorno de trabajo ROS.	Se obtiene puntos de interés, eligiendo las posturas mediante algoritmo SURF.	Referencia para la evaluación de algoritmos de detección de puntos de interés.
Navegación autónoma de un robot móvil usando técnicas probabilísticas de localización y mapeo basados en métodos Monte Carlo secuenciales *(Flores y cols.,2017)*.	Implementar un algoritmo para la navegación autónoma usando técnicas probabilísticas de localización.	Técnicas probabilísticas para la localización y los métodos Monte Carlo para el mapeo de la zona.	Uso de herramientas practicas, robustaz y flexibles para la navegación autónoma.	Referencia para el uso de lenguaje de Matlab.
Implementación de sistemas de navegación autónomo en robot móvil experimental para reconstrucción y exploración de entornos desconocidos *(Velázquez,2016)*.	Implementar un sistema de navegación autónomo en un robot móvil de tracción experimental.	Usar técnicas SLAM, un stack de navegación en ROS y un control de movimiento.	Hacer uso de herramientas del sistema operativo robótico, para el control de los recursos.	Experimentación de técnicas SLAM para el mapeo y localización del robot omnidireccional.

Resumen del Estado del Arte (continuación)

Articulo	Objetivo	Método / Algoritmo	Resultado	Utilidad
Control de seguimiento de trayectoria de retroalimentación de salida de un robot Omni-móvil *(Andreev y cols.,2019)*	Resolver el problema de trazado de trayectorias.	Desarrollo de un principio de cuasi-invarianza para sistemas autónomos de ecuaciones diferenciales.	Esquema de control de trayectoria global para un robot omnidireccional.	Referencia para el control autónomo de un robot omnidireccional.
Uso de un controlador difuso neural tipo-2 para el control de la navegación de robots evolutivos *(Lin y cols.,2019)*	Implementar una neurona difusa Tipo-2 en un control robótico.	Optimización de algoritmo PSO para robot omnidireccion.	Mejor resultado en el control automático. Trayectorias mejor definidas.	Control automático implementado teoría difusa Tipo-2.
Control difuso para un robot omnidireccional *(Abiyev y cols.,2017)*	Diseñar controlador difuso Tipo-1 en robot omnidireccional.	Controlador PID basado en Z-number	Mejor tratamiento de señales de un control robótico.	Referencia para la evaluación del control automático.
Sistema embebido para el control de movimiento de un robot omnidireccional *(Mamun y cols.,2018)*	Implementar un planificador y controlador para un robot omnidireccional.	Desarrollo de metodología para el control de un robot omnidireccional.	Mejor control de un robot omnidirectional.	Referencia para la evaluación de los sensores.
Control de un robot omnidireccional utilizando un sistema difuso basado en *Z-Number* *(Abiyev y cols.,2019).*	Diseñar un sistema de inferencia para el control de un robot omnidirectional.	Optimización de un controlador PID.	Implementación compleja para un sistema difuso Tipo-2.	Poca referencia para la evaluación de señales en los sensores.
Diseño de controladores difusos para navegación en robots omnidireccionales *(Masmoudi y cols.,2016).*	Diseñar un controlador difuso para evitar obstaculos.	Optimizar el algoritmo Fuzzy_PI.	Mejora en la navegación por espacios desconocidos.	Referencia para la evaluación del controladorFuzzy_PI.

Resumen del Estado del Arte (continuación)

Artículo	Objetivo	Método / Algoritmo	Resultado	Utilidad
Control de un robot futbolista usando árboles de comportamiento y Lógica Difusa *(Abiyev y cols., 2016)*	Implementar un control para la navegación de un robot *soccer* a base de teoría Difusa y métodos de árboles de expansión.	Emplea un sistema de tipo Mandami, junto técnicas *Random forest*.	Un control eficiente para robots móviles holonómicos de 4 ruedas.	Referencia para conocer la forma en que construye el controlador difuso para un robot omnidireccional de 4 ruedas.
Control de un robot omnidireccional de ruedas *mecanum* con una estrategia basada en un sistema de Lógica Difusa Tipo-2 *(Cuevas y cols., 2020)*	Implementar un control de ubicación y comportamiento de un robot omnidireccional de ruedas *mecanum*.	Lógica Difusa Tipo-2 para el control de la cinemática y velocidad de motores durante los desplazamientos.	Simulación de los desplazamientos omnidireccionales en V-REP *Simulator*.	V-REP *Simulator* y Matlab Simulink implementando el *YouBot* KUKA *Robotics*.
Control de trayectoria de un robot omnidireccional de servicio basado en un modelo de control predictivo de un sistema de inferencia neuro-difuso adaptativo *(Pang y cols., 2020)*	Control del seguimiento de trayectorias en robots móviles de tipo omnidireccional.	Un modelo de control predictivo basado en un sistema de inferencia neuro-difuso adaptativo.	Experimentación en una simulación, el resultado fue la implementación de un sistema de inferencia para el control de la función de costo.	MATLAB SIMULINK, modelado de sistemas ANFIS.
Odometría basada en visión para la localización de robots omnidireccionales en interiores *(Patruno y cols., 2020)*	Localización de un robot omnidireccional en interiores.	Algoritmos de detectores de puntos clave (SURF, SIFT, BRIEF) filtros de información estadística y geométrica.	Prototipo funcional de un robot omnidireccional de 3 ruedas con localización basada en visión monocular.	Robot omnidireccional de 3 ruedas. Dispositivos no especificados.
OxIOD: Base de datos para odometría bajo datos inerciales *(C. Chen y cols., 2018)*	Recopilar y etiquetar datos inerciales para la implementación en el área de inteligencia artificial.	Metodología de recolección de datos inerciales, mediante dispositivos móviles.	Bases de datos robustas y etiquetadas de datos inerciales para la utilización en sistemas de aprendizaje.	Utilidad para comparación de datos inerciales obtenidos con la computadora *Beaglebone Blue*.

Tabla 3.2: Resumen del Estado del Arte (continuación)

Articulo	Objetivo	Método / Algoritmo	Resultado	Utilidad
Seguimiento de trayectoria en Omni Robot con el método de odometría *(Fahmizal y cols., 2019)*	Seguimiento de una trayectoria definida por un robot de tipo omnidireccional.	Método de odometría para estimar la posición relativa a la posición inicial del robot, aplica un control PID para las velocidades.	Se obtiene una trayectoria definida por el método de odometría y estimación de posiciones durante el tiempo.	Se emplear para pruebas con el control PID sobre la plataforma física y utilizar las métricas para evaluar trayectorias.
Investigación sobre co-simulación de plataforma móvil omnidireccional basada en ADAMS y Matlab *(Tang y cols., 2017)*	Crear un control PID difuso para robots omnidireccionales para desplazamiento y navegación del robot	Control PID difuso e implementación de una simulación de dinámica multicuerpo (ADAMS).	Una simulación del control de trayectoria en robots omnidireccionales.	La implementación de simuladores para realizar pruebas rápidas del sistema difuso propuesto.
Seguimiento y dirección de un robot con ruedas *mecanum* utilizando control de lógica difusa *(Fahmizal y Kuo, 2017)*	Implementar un control de trayectoria y orientación para un robo omnidireccional, basado en Lógica Difusa.	Algoritmo basado en Lógica Difusa Tipo-1	Establecer la trayectoria para el robot omnidireccional utilizando solo datos inerciales con la mínima desviación en el rumbo.	La utilización del modelo para obtener los datos sensoriales
Simulación de modelado de robot con ruedas *mecanum* basada en software y control difuso *(Jamali y cols., 2017)*	Implentar un control de navegación para un robot omnidireccional mediante Lógica Difusa.	Modelo de Lógica Difusa tipo Mamdani sobre una plataforma Simulada con SolidWorks.	Simulación de control robótico omnidireccional, trayectorias controladas hasta cierto lapso de tiempo	Experimentación y comparación con el sistema difuso propuesto, utilización de algunas métricas.
Simulación de la navegación de un robot omnidireccional con ruedas *mecanum* basada en el algoritmo A* en Unity 3D *(Li y cols., 2019)*	Implementar un algoritmo para la navegación de un robot omnidireccional.	Algoritmo A* para evitar obstáculos	Fusión de datos sensoriales tipo LIDAR y el algoritmo A*	Métricas de evaluación en trayectorias para evaluar el control de navegación.

Capítulo 4

Metodología

4.1. Sistema propuesto

En este apartado se describe el sistema propuesto; como se menciona en el resumen se describe por una adquisición de información basada en visión estéreo, inferencia implementando la Lógica Difusa Tipo-2 sobre un robot omnidireccional con movimientos controlados por un bloque que utiliza un algoritmo de cinemática dinámica. En esta sección se presentan los algoritmos estudiados para cumplir con el objetivo general y objetivos específicos, en la Figura 4.1 se muestra el sistema propuesto.

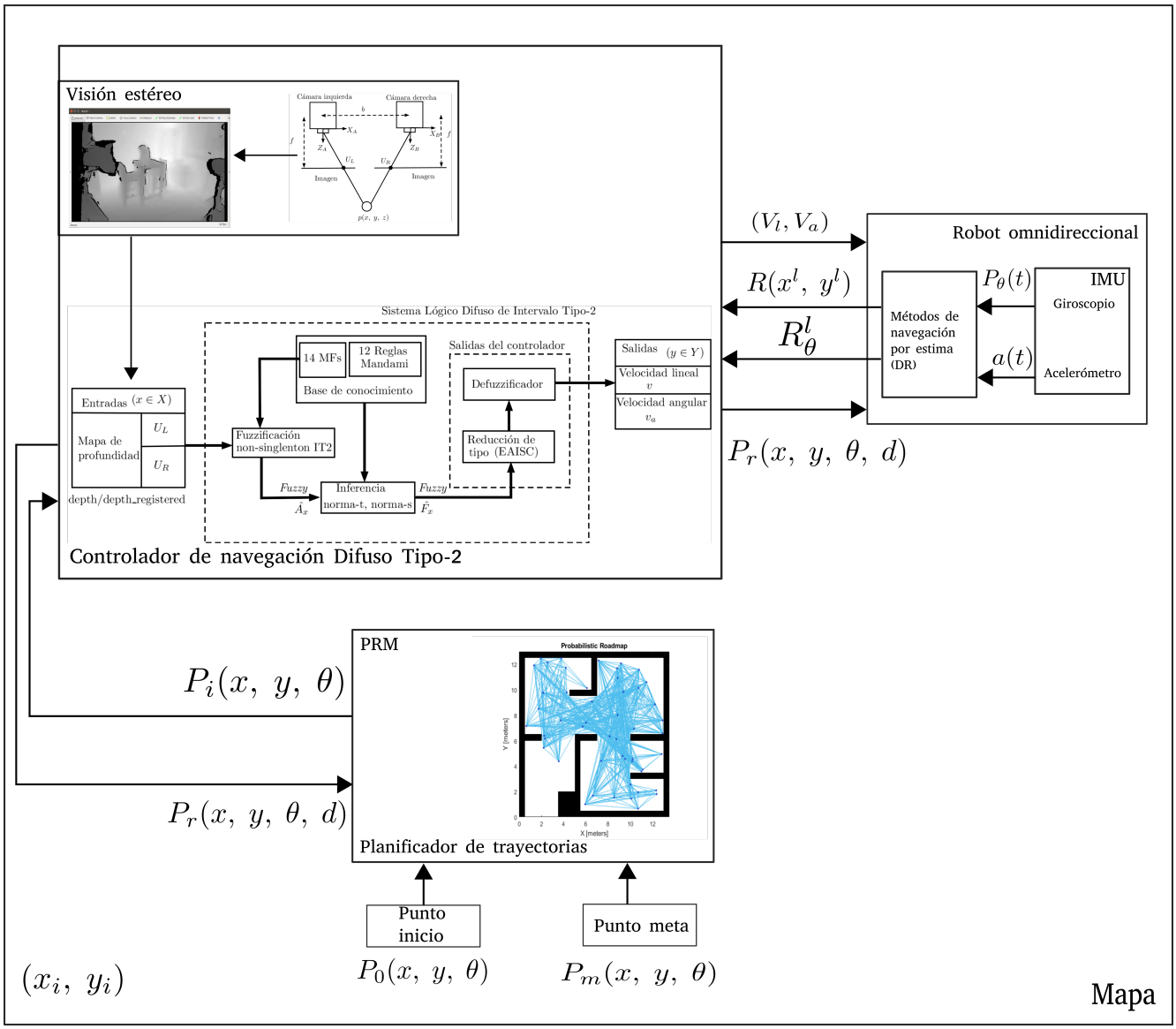


Figura 4.1: Metodología para la navegación de robot omnidireccional.

4.1.1. Adquisición de información

Para realizar la tarea de control durante la navegación de un robot móvil de tipo omnidireccional en un entorno con obstáculos, es necesario contar con sensores que ayuden a detectar los objetos que se encuentran en el escenario. Se implementó la navegación visual con ayuda de una cámara ZED que utiliza la visión estéreo y a través del mapa de profundidad se puede obtener las distancias del centro de las cámaras al objeto visualizado, haciendo uso de las expresiones básicas de geometría epipolar se desglosa la información del mapa de profundidad para utilizarla como entrada en un sistema.

La navegación visual en este proyecto fue mediante el uso de una cámara estéreo, la cual utilizando sus librerías se pudo acceder a la información que obtiene del entorno en específico la *sl::Mat*, la cual almacena los datos de triangulación que capta el sistema de visión.

Visión estéreo

Los sistemas de visión estéreo son sensores confiables y eficientes para la robótica móvil ya que permite extraer la información del entorno de corto a largo alcance entre $0.3 - 2.5m$, como ejemplo la cámara ZED. Este tipo de cámaras se implementa para tareas de navegación móvil, procesamiento de imágenes, detección de características, entre otras *(Krishnan y Kollipara, 2014)*. Los sistemas de visión estéreo implementan el cálculo de las distancias entre objetos del mundo real, producto de la triangulación de la imágenes haciendo uso de funciones de la geometría epipolar, esto con el objetivo de evitar colisiones durante la navegación en tiempo real, ver Figura 4.2.

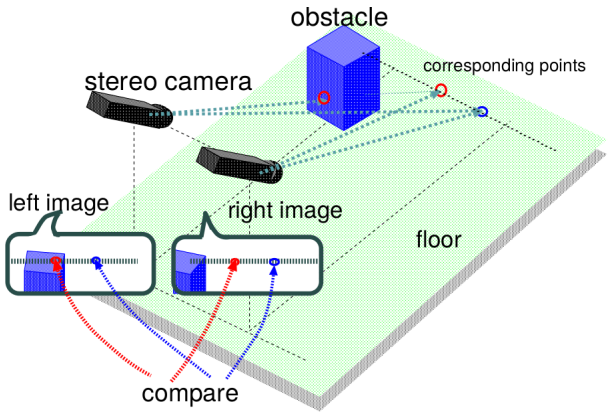


Figura 4.2: Ejemplo de un sistema de visión estéreo *(Kumano y cols., 2000)*.

El sistema de visión estéreo visualiza la información del entorno en forma de mapas de profundidad, dentro del cual se realizan procesos geométricos para generar distancias a ciertos puntos del entorno, dichos puntos captados por ambas cámaras, poseen una distancia hacia del objeto al sensor, dentro sistema de navegación de un robot móvil determina hacia donde desplazarnos evitando colisiones, la representación de dichas características se muestran en la Figura 4.3.

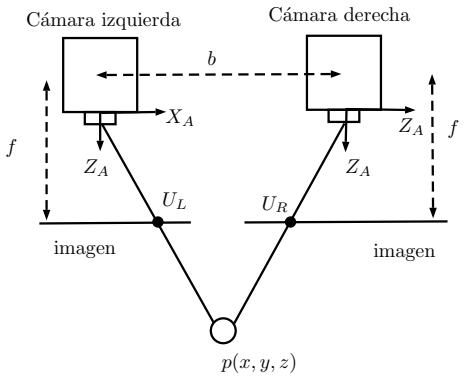


Figura 4.3: Esquema matemático de la visión estéreo *(Islam y cols., 2018)*.

donde, b es la distancia entre las dos cámaras, f es la distancia focal de la cámara, X_A es el eje x de la cámara, Z_A es el eje óptico de la cámara, P es el punto en el mundo real, definido por coordenadas $(X,\ Y,\ Z)$, U_L es la proyección del punto P en el mundo real adquirida por la cámara izquierda, U_R es la proyección del punto P en el mundo real adquirida por la cámara derecha. Para calcular estos dos últimos valores se aplican las siguientes ecuaciones *(Z. Chen y cols., 2018)*.

$$U_L = f * \frac{X}{Z} \tag{4.1}$$

$$U_R = \frac{(X - b)}{Z} \tag{4.2}$$

La distancia al punto de profundidad central que realiza un sistema visión estéreo al generar una triangulación entre la imagen derecha e izquierda se obtiene de la siguiente manera.

$$p_z = \frac{b * f}{X_A - Z_A} \tag{4.3}$$

donde, b es la distancia entre cámaras y $X_A - Z_A$ es la disparidad del punto proyectado, es decir, la diferencia entre coordenadas de la imagen izquierda y derecha. El valor se puede interpretar como a mayor disparidad el punto estará más lejos al objeto, mientras si el valor es cercano a 0, el objeto se encuentra más lejos de la cámara.

Para implementar la información visual del sistema estereoscópico en el algoritmo de navegación móvil, consistió en tener el cálculo del movimiento de la cámara entre imágenes sucesivas, aplicando odometría visual la representación gráfica se expresaría como en la Figura 4.4.

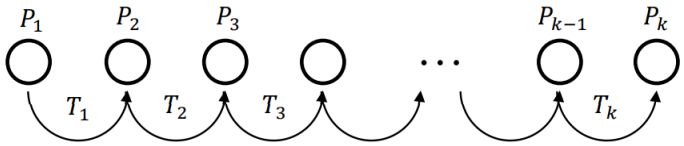


Figura 4.4: Relación entre la matriz de transformación y posición de la cámara en cada fotograma *(A. P. González, 2015)*.

$$T_k = \begin{bmatrix} R_{k,k-1} & t_{k,k-1} \\ 0 & 1 \end{bmatrix} \tag{4.4}$$

donde, T_k es la matriz de movimiento de $4 * 4$, $R_{k,k-1}$ la matriz de rotación $3 * 3$ entre el momento k y el antecesor $k-1$ y $t_{k,k-1}$ el vector de traslación de $3 * 1$ entre el momento k y $k-1$. La posición actual de la cámara, por tanto la pose del robot omnidireccional queda expresada como.

$$p_k = T_k P_{k-1} \tag{4.5}$$

con P_0 siendo la posición inicial de la cámara, en coordenadas homogéneas. Para el cálculo de R y t se utiliza la descomposición en valores singulares. Esta implementación se utilizó para calcular la pose del robot en cada momento de desplazamiento durante la navegación.

Implementación de datos inerciales

Para implementar la información inercial la cual se utilizó con el fin de analizar el comportamiento de una navegación en robots móviles de tipo omnidireccional (ver Figura 4.5), se midieron estas trayectorias las cuales fueron comparadas con las obtenidas por la navegación visual. La información de la IMU se utilizó para generar un control de la base robótica y de igual manera mantener dicho control en cada pose generada durante la navegación, esta parte queda descrita en el bloque de control omnidireccional del robot que se realizó en tres etapas, las cuales se describirán a continuación.

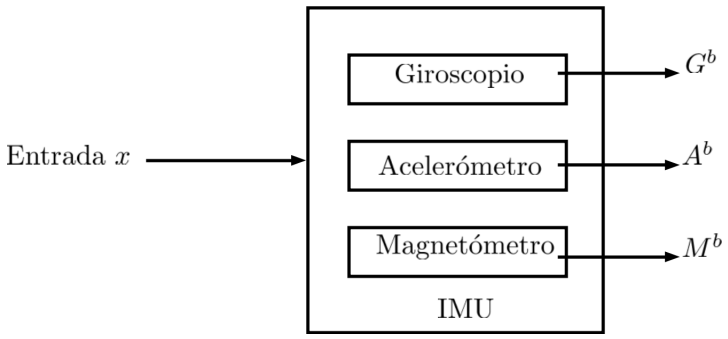


Figura 4.5: Representación gráfica de un sensor de tipo IMU, activado por una entrada x.

Una vez calibrado el sensor de tipo IMU se puede comenzar obteniendo información como posición, velocidad, actitud. Pose la cual se calcula a través de transformación de coordenadas y con métodos de navegación por estimación. El primer paso es actualizar las velocidades una vez iniciada la navegación hacia un punto meta. Esto directamente se obtiene con la siguiente expresión.

$$\begin{bmatrix} V^b_{x,m} \\ V^b_{y,m} \end{bmatrix} = \begin{bmatrix} V^b_{x,m-1} \\ V^b_{y,m-1} \end{bmatrix} + \begin{bmatrix} \triangle T & 0 \\ 0 & \triangle T \end{bmatrix} \begin{bmatrix} a^b_{x,m} \\ a^b_{y,m} \end{bmatrix} \tag{4.6}$$

donde, V^b es la velocidad, a^b es la aceleración obtenida por el acelerómetro en el momento actual durante la navegación y $\triangle T$ es simplemente el tiempo del propio sistema de navegación inercial.

Seguido de la actualización de la pose S^b_I en el sistema de coordenadas de navegación, se calcula integrando las velocidades estimadas. La pose se actualiza mediante la expresión.

$$S^b_{I_m} = S^b_{I_{m-1}} + \int_{t_{m-1}}^{t_m} V^b_m dt \tag{4.7}$$

donde, $S^b_{I_m}$es la pose del robot, V^b_m es la velocidad en el instante m, $S_{I^b_{m-1}}$ es la pose en el momento $m-1$ y el intervalo $[t_m, t_{m-1}]$ es la longitud del paso de actualización de pose, $\triangle T = t_m - t_{m-1}$.

Para determinar la actitud del robot omnidireccional implica calcular los ángulos de pitch, roll y yaw, para ello es necesario resolver la siguiente ecuación.

$$\dot{E} = \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{varphi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\phi \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{4.8}$$

donde p, q, r son vectores de velocidades angulares medidas por los giroscopios en el eje x, y y z respectivamente. $\dot{\phi}$, $\dot{theta}$ y $\dot{\varphi}$ son vectores de las derivadas de los ángulos *roll*, *pitch* y *yaw* respectivamente.
La siguiente expresión es una diferencial discreta, la cual debe resolverse para determinar los ángulos de Euler, para ello se a empleado el método de *Rugen-Kutta*, el cual que está dado por la expresión.

$$y_i + 1 = y_i + \frac{k1 + 2 * k2 + 2 * k3 + k4}{6} * h \tag{4.9}$$

donde, $k1 = f(x_i, y_i)$, $k2 = f(x_i + \frac{1}{2} * h, y_i + \frac{1}{2} * k1 * h)$, $k3 = f(x_i + \frac{1}{2} * h, y_i + \frac{1}{2} * k2 * h)$, $k4 = f(x_i + hy_i + k3 * h)$ y h expresa el tiempo de muestreo.
Los términos x_i, y y_i se ajustan a la expresión anterior, de la siguiente forma.

$$x_i = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{4.10}$$

$$y_i = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} \phi \\ \theta \\ \varphi \end{bmatrix} \tag{4.11}$$

$$h = dt = 0.01 \tag{4.12}$$

Y por lo tanto.

$$f_i = \begin{bmatrix} f_0 \\ f_1 \\ f_2 \end{bmatrix} = \begin{bmatrix} x_0 & x_1 \sin y_0 \tan y_1 & x_2 \cos y_0 \tan y_1 \\ 0 & x_1 \cos y_0 & -x_2 \sin y_0 \\ 0 & x_1 \sin y_0 \sec y_1 & x_2 \cos y_0 \sec y_1 \end{bmatrix} \tag{4.13}$$

De esta forma se obtienen los ángulos de Euler a partir de las varias interacciones en un tiempo de muestreo determinado, pero como el RK calcula la actitud actual a partir de la anterior.
Para obtener la orientación de la plataforma se ha utilizado las medidas del campo magnético en el plano XY, estas medidas de calibración previa describen el momento de rotar a la plataforma sobre un eje fijo, debido a este comportamiento la ecuación que se emplea para calcular el ángulo de orientación o *Headinges*

$$Hed = \tan^{-1} \frac{(M_y)}{(M_x)} \tag{4.14}$$

donde, *Hed* es el ángulo de orientación con respecto al norte magnético o *Heading*, M_y medida calibrada del campo magnético en el eje y, M_x medida calibrada del campo magnético en el eje x.

La ecuación (4.15) y (4.16) es aplicable solo cuando el movimiento de la plataforma se realiza en el plano horizontal sin embargo puede ser aplicable a un *drone* si se requiere la compensación de las medidas del campo magnético, previo a la realización del cálculo del ángulo de orientación, se definen de la siguiente manera.

$$M_{xc} = M_x \cos\theta + M_z \sin\theta \tag{4.15}$$

$$M_{yc} = M_x \sin\phi \sin\theta + M_y \cos\phi - M_z \sin\phi \cos\theta \tag{4.16}$$

donde, M_{xc} es la medida del campo magnético en el eje x compensado con los ángulos de inclinación, M_{yc} es la mediad del campo magnético en el eje y compensado con los ángulos de inclinación, M_x Es la medida original del campo magnético sin compensación en el eje x, M_y es la medida original del campo magnético sin compensación en el eje y, M_z es la medida original del campo magnético sin compensación en el eje z, θ ángulo de inclinación en *pitch*, ϕ ángulo de inclinación en *roll*. Lo siguiente es utilizar la ecuación (4.17) para realizar el nuevo cálculo de la orientación.
Para estimar la posición de la plataforma móvil se emplea la medida de los acelerómetros, suprimiendo los efectos de la gravedad mediante la siguiente formalización matemática.

$$g_b = \begin{bmatrix} g_x \\ g_y \\ g_z \end{bmatrix} = C_b^n = \begin{bmatrix} 0 \\ 0 \\ b \end{bmatrix} = \begin{bmatrix} -g\sin\theta \\ g\cos\theta\sin\phi \\ g\cos\theta\cos\phi \end{bmatrix} \tag{4.17}$$

$$a = \frac{\partial^2 r}{\partial t^2} + g \tag{4.18}$$

$$a_{Ib} = a_b - (\omega + \Omega) x v_b - g_b \tag{4.19}$$

donde, a_{Ib} es la aceleración considerando la plataforma de un cuerpo idealmente inercial, a_b es la aceleración medida por el acelerómetro, Ω cantidad de rotación del cuerpo debido a la rotación de la tierra, ω es la cantidad de rotación del cuerpo debido al propio cuerpo, g_b vector gravedad del cuerpo, C_b^n matriz de cambio de coordenadas del cuerpo a inerciales.
Dado que los giroscopios miden siempre Ω y ω entonces se tiene.

$$(\omega + \Omega) = \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{4.20}$$

Reemplazando las ecuaciones (29) y (30) en la ecuación (31) se obtienen las aceleraciones,

suprimiendo las rotaciones y la gravedad y se representa por las ecuaciones.

$$a_{Ibx} = a_{bx} + v_{by} * r - v_{bz} * q + g \sin\theta \tag{4.21}$$

$$a_{Iby} = a_{by} - v_{bx} * r + v_{bz} * p - g \cos\theta \sin\phi \tag{4.22}$$

$$a_{Ibz} = a_{bz} + v_{bx} * q - v_{bz} * p - g \cos\theta \cos\phi \tag{4.23}$$

donde, a_{Ibx}, a_{Iby}, a_{Ibz}, representan la aceleración sin los efectos de rotación y de gravedad en los ejes x, y y z respectivamente, a_{bx}, a_{by}, a_{bz} es la aceleración medida directamente en los ejes x, y, z respectivamente, v_{bx}, v_{by}, v_{bz} representa la velocidad del robot en los ejes x, y y z, p, q, r las velocidades angulares del robot en los ejes x, y y z, θ es el ángulo *pitch*, *phi* es el ángulo *roll*, g es la gravedad.
Una vez corregidas completamente las medidas de las aceleraciones, se procede a realizar la integración de la señal, con lo cual se obtiene la velocidad, la misma que será integrada nuevamente para obtener el desplazamiento del robot omnidireccional.

4.1.2. Sistema de Inferencia

En este bloque se describe el método basado en Lógica Difusa Tipo-2 el cual se encargará del control y desempeñarse en situaciones con incertidumbre. Se trasladó la experiencia sobre navegación móvil todo lo referente al control de los desplazamientos mediante el ajuste de las velocidades a un conjunto de reglas las cuales reciben de entrada la información de profundidad en la que se extraen las distancias entre objetos (U_L, U_R) e indican que tanto se encuentra un objeto de la cámara derecha e izquierda del robot. El controlador se realizó con la librería PyIT2FLS de *(Islam y cols., 2018)*.

Sistema Lógico Difuso de Intervalo Tipo-2 (IT2FLS)

El sistema propuesto tuvo constantes actualización para dotarlo de una no tan compleja base de conocimiento, donde el robot puede desempeñarse, tanto en ambientes con obstáculos estáticos como dinámicos, haciendo uso de sus características como robot omnidireccional. Se inició con una base de conocimiento compleja y se fue reduciendo hasta ajustarla a 12 reglas que describen correctamente las situaciones planteadas en los objetivos. La ventaja que tuvo el implementar este tipo de técnica es su versatilidad para adaptarse en entornos donde los obstáculos se pueden llegar a desplazar, al igual que durante la ejecución, el tiempo generado es corto a comparación de otros sistemas. El IT2FLS se compone por 5 elementos principales: Fuzzificación, Inferencia, Base de conocimiento, Reducción de tipo y Defuzzificación, como se visualiza en la Figura 4.6.

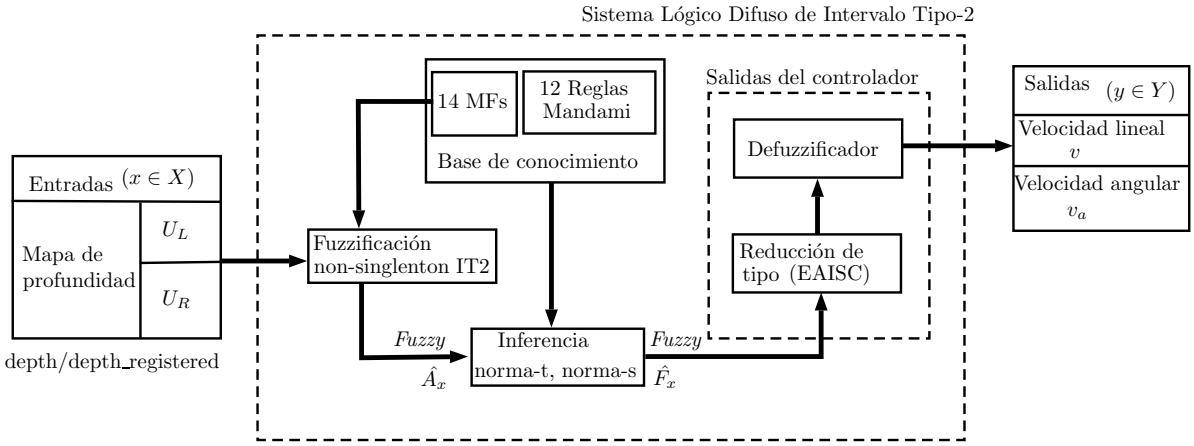
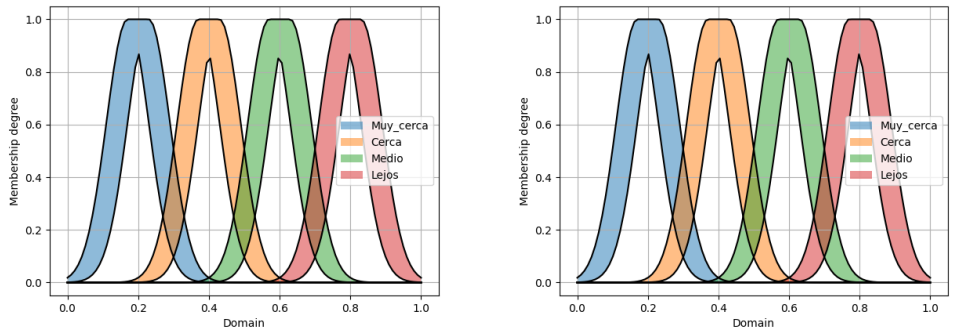


Figura 4.6: Sistema Lógico Difuso de Intervalo Tipo-2 propuesto.

Fuzzificación

En los IT2FLS existen tres tipos de fuzzificadores: Singlenton, No Singlenton Tipo-1 y No Singlenton Tipo-2. Para este caso se implementó una fuzzificación No-Singlenton Tipo-2 que transforma la medición $x_i = x_i'$ en un numero difuso tipo-2, es decir en un Sistema Lógico Difuso de Intervalo Tipo-2 cuyas funciones de pertenencia superior e inferior de la huella de incertidumbre es un número difuso tipo-1.

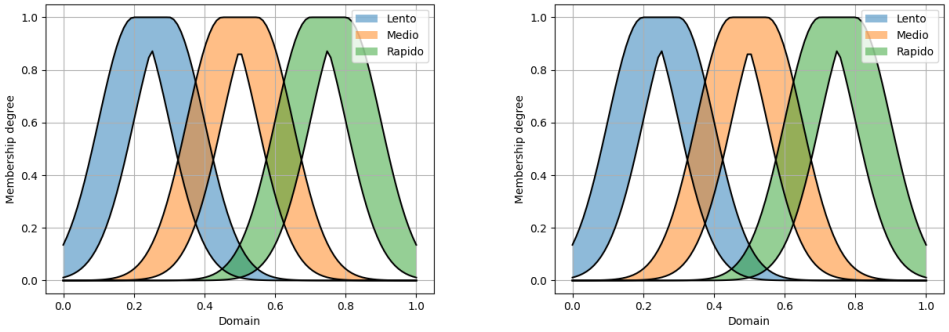
Se manejó una fuzzificación con una huella de incertidumbre delimitada por dos funciones de tipo *gaussiano* por el motivo que este tipo de conjuntos describen mejor comportamientos de la vida real, así como tienen buen desempeño en situaciones no controladas como se muestra en el trabajo de *(Wu, 2018)*. Para los conjuntos se tomó como entrada el mapa de profundidad de la cámara ZED, del cual se extrajeron los valores de los puntos U_L y U_R los cuales representan las distancias de objetos hacia la cámara, trazadas por la visión derecha e izquierda del sensor. Dichos valores se le envían en tiempo real al sistema el cual dependiendo de la distancia en que se encuentre ajusta y controla la velocidad en que desplaza y en ese sentido toma una decisión a hacia que dirección desplazarse para esquivarlo. El sistema se inicializa como una velocidad lineal y angular constantes hacia delante de 0 a 0.25m/s y 0-2.631rad/s respectivamente, si al iniciar un objeto se encuentra a una distancia considerable que pueda colisionar con el, el controlador ajusta la velocidad y le indica a que posición desplazarse para evitar el obstáculo y para lograr este desplazamiento se aplica la navegación visual para determinar la pose próxima y más cercana al punto para retomar la trayectoria actualizada. Las entradas quedan definidas como se muestran en la Figura 4.7.



(a) (b)

Figura 4.7: Representación gráfica de la entradas del sistema difuso Tipo-2, (a) Sensor derecho, (b) Sensor izquierdo.

Para las salidas del IT2FLS que indican la velocidad lineal y angular del robot se definieron 3 conjuntos difusos delimitados por dos funciones del tipo *gaussiano* con niveles de rápido, medio, lento, ver Figura 4.8.



(a) (b)

Figura 4.8: Conjuntos de la salida del sistema difuso Tipo-2.

Las expresiones para la funciones de pertenencia de las entradas*gaussianas* quedan expresadas como se muestran a continuación.

$$\mu_{\underline{A_i}}(x) = exp\left(\frac{(c_i - x)^2}{2\sigma_i^2}\right) \tag{4.24}$$

$$\mu_{\overline{A_i}}(x) = exp\left(\frac{-(c_i - x)^2}{2\sigma_i^2}\right) \tag{4.25}$$

donde, $c_i\{0.2, 0.4, 0.6, 0.8\}$ es igual a los datos centrales de cada uno de los conjuntos de

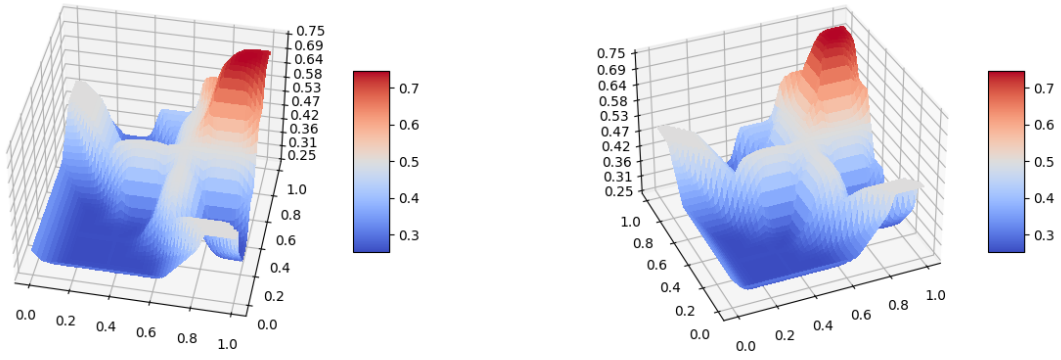
entradas, con una desviación estándar $\sigma\{0.5\}$ con la misma magnitud entre conjuntos, para cada valor de x. Para los valores establecidos para las salidas el valor de $c_i\{0.25, 0.5, 0.75\}$. En cuanto a la definición de las reglas se realizaron un listado las situaciones posibles en las que el robot omnidireccional podría enfrentarse. El conjunto de reglas se depuró y solo se implementaron 12 para brindarle un preciso sistema de inferencia. Las reglas se muestran el Tabla 4.1.

Tabla 4.1: Conjunto de reglas para el controlador difuso Tipo-2.

No.	Regla
1	Si (SD es muy cerca) y (SI es muy cerca) entonces (Vx es lento) (Vy es rápido)
2	Si (SD es cerca) y (SI es cerca) entonces (Vx es lento) (Vy es rápido)
3	Si (SD es medio) y (SI es medio) entonces (Vx es normal) (Vy es lento)
4	Si (SD es lejos) y (SI es lejos) entonces (Vx es rápido) (Vy es normal)
5	Si (SD es muy cerca) y (SI es cerca) entonces (Vx es lento) (Vy es rápido)
6	Si (SD es cerca) y (SI es muy cerca) entonces (Vx es lento) (Vy es rápido)
7	Si (SD es medio) y (SI es cerca) entonces (Vx es normal) (Vy es rápido)
8	Si (SD es cerca) y (SI es medio) entonces (Vx es normal) (Vy es rápido)
9	Si (SD es medio) y (SI es lejos) entonces (Vx es normal) (Vy es normal)
10	Si (SD es lejos) y (SI es medio) entonces (Vx es normal) (Vy es normal)
11	Si (SD es lejos) y (SI es cerca) entonces (Vx es lento) (Vy es normal)
12	Si (SD es cerca) y (SI es lejos) entonces (Vx es lento) (Vy es normal)

La forma en que inicialmente se definieron las reglas fueron simulando situaciones probables en la que el robot podría encontrarse durante la navegación, conforme transcurrían las pruebas se comenzó a especificar mejor de tal manera que no afectó su funcionalidad ya obtenida. Las reglas 1, 2, 5 y 6 se enfoca en situaciones donde el robot se encuentre muy cerca al objeto, en específico las reglas 1 y 2 a objetos que se encuentre directo a su campo de visión del robot, como puede suceder al inicio de la ejecución del sistema y se requiera hacer un desplazamiento inmediato, las reglas 5 y 6 dado que se requería un desplazamiento rotacional cuando se requiera visualizar una nueva trayectoria meta es posible que se encuentre un objeto al girar y requiera actuar rápido. Las reglas 3, 7, 8 y 9 definen situaciones en las que un objeto tiene una distancia media a la definida como máxima por el sistema de visión estéreo, las reglas 8 y 9 describe una situación en la que otro objeto pueda generar una colisión justo después de evadir el objeto inmediato cercano al robot de esta manera se previene ajustando y evaluando el siguiente punto de desplazamiento. Las reglas 4, 10, 11 y 12 describen situaciones en distancias lejanas con pocos obstáculos dentro del campo de visión y donde los desplazamientos para evitar colisiones son movimientos cortos e inmediatos, como por ejemplo en las reglas 11 y 12 las cuales representan entornos de visión donde un objeto cercano está ligeramente dentro de la trayectoria del robot y este debe realizar un ligero desplazamiento ya

sea izquierda o derecha para evitar y re-calcular la siguiente pose para ingresar a la trayectoria más próxima a la meta.Una manera de representar la salida de un sistema difuso es mediante una superficie 3D en la que se combina la entrada, es decir, cada una de las variables de entrada con sus conjuntos descritos por sus funciones de pertenencia correspondientes y el conjunto de reglas que se implementaron para el sistema como base de conocimiento, ver Figura 4.9.



(a) (b)

Figura 4.9: Representación gráfica de la entradas del sistema difuso Tipo-2, (a) Sensor derecho, (b) Sensor izquierdo.

Bloque de Inferencia

El bloque de inferencia transforma de un Sistema Lógico Difuso Tipo-1(T1FS) en una salida T1FS. Aquí es donde múltiples antecedentes de las reglas se conectan por operaciones de norma-t (mínimo o producto). Los grados de pertenencia de los conjuntos de entrada son combinados con los de salida usando la composición *sup-star*. Teniendo en cuento esto, múltiples entradas pueden ser combinadas usando las operaciones de norma-t o durante la defuzzificación por suma de pesos. En un IT2FLS es muy similar, ya que mapea la entrada del conjunto tipo-2 en una salida. Para el controlador se implementó norma-t y norma-s, obteniendo el área de FOU para que esta interpretada por el algoritmo de reducción de tipo dichas expresiones se muestran a continuación.

$$\mu_{\hat{A}\cap\hat{B}}(u_i) = max\{0,\ \mu_{\hat{A}}(u_i) + \mu_{\hat{B}}(u_i) - 1\} \tag{4.26}$$

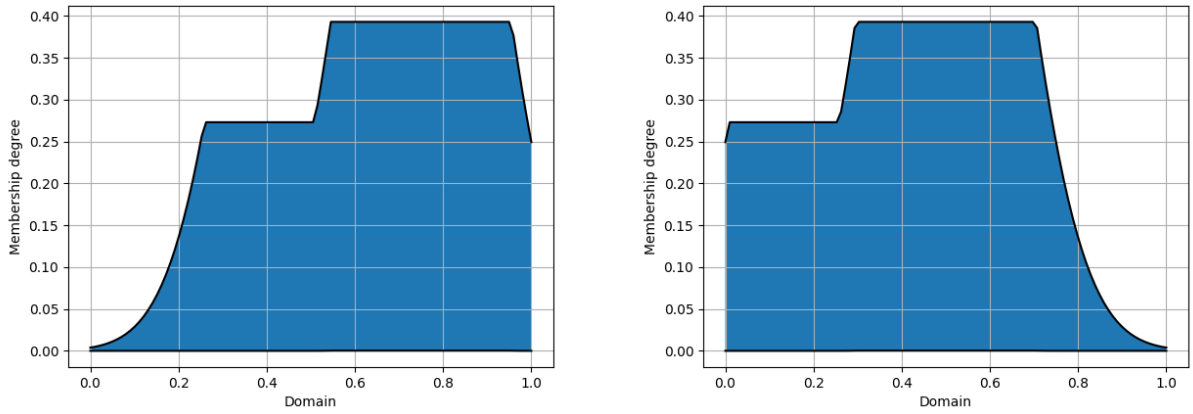
$$\mu_{\hat{A}\cup\hat{B}}(u_i) = min\{1, \mu_{\hat{A}} + \mu_{\hat{B}}(u_i)\} \tag{4.27}$$

donde, $\hat{A}$ y $\hat{B}$ son conjuntos difusos tipo-2, con funciones de pertenencia $\mu_{\hat{A}}(u_i)$ y $\mu_{\hat{B}}(u_i)$ definidas en el universo u_i, se tiene un conjunto de intersección $\hat{A}_i \cap \hat{B}_i$ para la norma-t

y un conjunto unión $\hat{A}_i \cup \hat{B}_i$ para la norma-s (Cuchango y Méndez, 2010).

Reducción de tipo

Para procesar la salida dentro de la defuzzificación es necesario conocer los valores que generan un cambio dentro de la huella de incertidumbre haciendo uso de *yl* y *yr*, los cuales se obtienen con algoritmos de reducción de tipo aplicados a la FOU de salida, ejemplo en forma gráfica Figura 4.10.



(a) (b)

Figura 4.10: Representación de la FOU en ambas salidas del sistema de control.

El algoritmo implementado para la reducción de tipo fue el (EIASC) algoritmo mejorado Iterativo con condición de paro, el cual al encontrarlo en la literatura y analizando el algoritmo en ejecución se notó una mejora en tiempo que un algoritmo de convergencia cuadrática *(Arellano, 2014)*. Este tipo de reducción es más rápido en ejecución que otros algoritmos, como por ejemplo el algoritmo Karnik-Mendel o el de Centros de conjuntos, la razón de implementar un algoritmo con menor tiempo de ejecución es debido a que el control de navegación debe ser en tiempo real y optar por uno que de resultados al momento indica que el control se puede realizar en momentos que el robot necesite tener tomar decisiones rápidas ante obstáculos. La razón por la cual el algoritmo tiene un mejor desempeño es principalmente a su bloque de inicialización, cuenta además de una condición de paro, para evitar iteraciones de más y realiza una búsqueda en sentido opuesto. El proceso iterativo del algoritmo EIASC, se muestra a continuación.

Algorithm 1 Algoritmo EIASC

procedure CÁLCULO DE yl Y yr
Input: Xl (vector límites inferiores x), Xr (vector límetes superiores de x), Wl (vector límites inferiores), Wr (vector límites superiores).
Output: y
  $a = Xl * Wl, b = sum(Wl), ly = length(Xl)$
  while $1 < ly \quad \& \quad yl > Xl(l)$ **do**
    $a = a + Xl(l) * (Wr(l) - Wl(l))$
    $b = b + Xr(l) - Wl(l)$
    $yl = a/b, l = l + 1$
  $a = Xr * Wl, b = sum(Wl), ly = length(Xr)$
  $r = ly, b = sum(Wl), ly = length(Xr)$
  while $r > 0 \quad \& \quad yr < Xr(r)$ **do**
    $a = a + Xr(r) * (Wr(r) - Wl(r))$
    $b = b + Wr(r) - Wl(r)$
    $yl = a/b, r = r - 1$
  Return: $y = (yr + yl)/2$

Defuzzificación

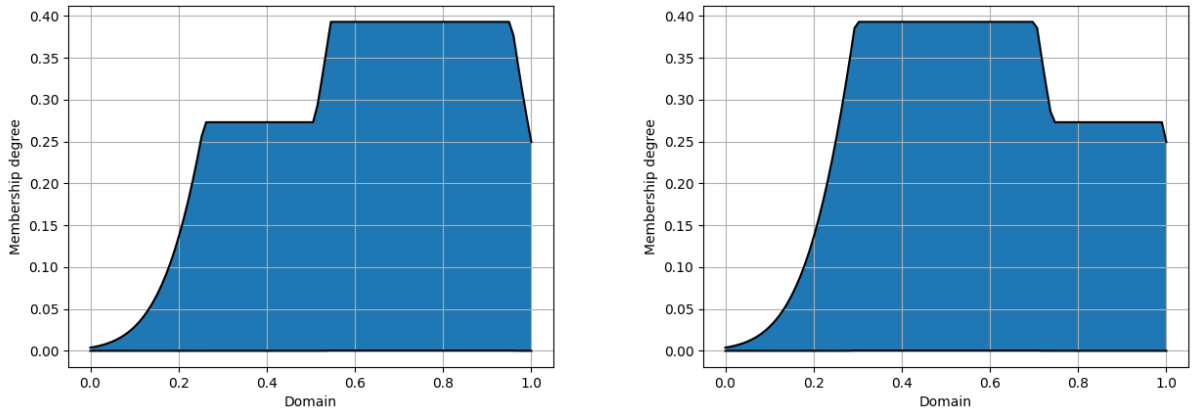
Para realizar la defuzzificación se tuvo que programar la expresión (24) lo que expresa es un promedio de los puntos de cambio obtenidos en la reducción de tipo.

$$y = \frac{yl + yr}{2} \tag{4.28}$$

donde, yr y yl indican los puntos de cambio del conjunto difuso Tipo-2. De esta forma se obtiene un primer valor de salida que puede ser utilizado como resultado del sistema, lo que puede indicar fácilmente en un control para un robot móvil en la velocidad lineal y angular.

Un ejemplo de salida para la FOU de la Figura 4.11 para ambas salidas del sistema, en un momento durante la navegación con un obstáculo cercano al robot, con valores de entrada en ese mismo punto de la navegación. Se tendría como entrada un mapa de profundidad el cual utiliza distancias a un objeto donde cercano a 1 es una distancia corta al robot y un valor cercano 0 indica cierta lejanía con el robot, El valor de la entrada U_l indica la distancia del objeto al robot, en la región del punto de referencia $p(x,\ y,\ z)$ a la izquierda del ángulo máximo de alcance de visión de la cámara, U_r indica la distancia del objeto al robot, pero la región derecha. Esta información ayuda al controlador ajustar las velocidades para tomar una decisión a que parte desplazarse sin tener colisión, ambas distancias pueden ser calculadas tomando como referencia el punto $p(x,\ y,\ z)$ del sistema de visión estéreo. El valor de entrada para U_l es de 0.745 y

U_r es 0.923.



(a) (b)

Figura 4.11: Representación de la FOU en ambas salidas del sistema de control, en una posición del escenario (x : 97.214, y : −95.1147).

El robot detecta que se encuentra con un objeto más cercano por la derecha pero un cierto riesgo de otro objeto de la parte izquierda el robot hace el reajuste de las velocidades V_x : 0.55075, V_ω : 0.54069 lineales y angulares, respectivamente. V_x : 0.13768m/s y V_ω : 1.4228rad/s lo que demuestra una disminución de la velocidad angular en las ruedas para desplazarse de forma segura, la velocidad lineal igual se disminuye y se desplaza al siguiente punto del campo de visión seguro siguiendo una nueva trayectoria al nodo cercano a la meta.

La visión estéreo es un sensor que da una mejor estimación de la distancia en la que se encuentra un objeto y esto es necesario en control de navegación ya que se puede evadir obstáculos de forma más anticipada como se describe en la obtención de información. Este tipo de sistema de visión puede generar un mapa de profundidad el cual calcula las distancias y las agrupa en valores de entre 0 y 255 dependiendo de que tan lejano o cercano este un objeto dentro de su campo de visión, ver Figura 4.12.

(a) (b)

Figura 4.12: Campo de visión del robot con un obstáculo por la región derecha del robot, la distancia al objeto es de z : 34.4736cm.

Planificador de trayectorias

Se implementó un algoritmo para calcular las trayectorias de un punto inicio a un punto meta dentro del mapa que se requiere demostrar el control de la navegación durante el seguimiento de trayectoria. Como se describe en el apartado del sistema propuesto la navegación en robots móviles se representa por un bloque de planificación, control del desplazamiento, seguimiento de trayectorias *(Valverde Moreno, 2020)*. Dado que se tiene un control mediante un sistema de visión estéreo es necesario para poder navegar de un punto a otro se necesita un algoritmo planificador de trayectorias el cree la ruta más corta posible a la meta y mediante la odometría visual determinar el punto de la pose del robot en cada momento dentro de la trayectoria. Se experimentó con algoritmos de planificación de rutas geométricas ya que se cuenta con un escenario el cual es conocido en cada punto del entorno y este tipo de algoritmos ademas de ser rápidos en calcular las trayectorias, no son demasiado complejos para implementar característica que ayuda a tener un desempeño en tiempo real durante la navegación. El algoritmo que se implemento para realizar esta tarea fue el planificador de hoja de ruta probabilística (PRM), el cual se describe a continuación *(Klančar y cols., 2017)*.

Algorithm 2 Algoritmo PRM

procedure GENERACIÓN DE RUTAS
Input: $G(V, E)$
Output: $line(x, y, color, b)$
    $D = 1$
    $maxIter = 200$
    $M = M^n$
    $j = 1$
    while $j <= maxIter$ **do**
        $xRand = 10 * rand(1, 2)$
        $M = [M; xRand]$
        $con = C^n$
        for $i = 1 : size(M, 1)$ **do**
            $d = norm(M(i, :) - xRand)$
            if $d < D \& d > eps$ **then**
                $con = [con; xRand, M(i, :)]$
        for $i = 1; size(con, 1)$ **do**
    $line(con(i, [1, 3]), con(i, [2, 4]), ' color', ' b')$

El planificador de hoja de ruta probabilística (PRM) es el método utilizado para buscar rutas entre puntos de inicio y meta. El primer paso es la fase de aprendizaje donde se realiza una hoja de ruta o un gráfico no dirigido del espacio libre. En el segundo paso, el punto de inicio y objetivo actual se conectan al gráfico y se utiliza algún algoritmo de búsqueda de ruta de gráfico para encontrar la ruta óptima. El algoritmo PRM agrupa los nodos en componentes conectados. Se debe almacenar el componente conectado al que pertenece el nodo. Para realizar la agrupación, los nodos se recopilan dentro de un radio fijo del nodo generado aleatoriamente y se van interconectando evaluando la conexión menos costosa de recorrer.

Algoritmo de control basado en el modelo cinemático de un robot omnidireccional

El algoritmo está basado en un modelo de cinemática dinámica en robots omnidireccionales pero con el tipo de rueda *mecanum*. Utiliza las expresiones del modelo dinámico y mediante la expresión (35) se obetienen las velocidades de cada rueda. De este algoritmo se obtienen poses, en otras palabras la odometría del robot en base a pose generadas. Es importante la sintonización de parámetros, debido a que las características físicas del robot influyen al modelo cinemático ya que son consideradas para calcular las velocidades.

Algorithm 3 Algoritmo Modelado Dinámico

procedure CÁLCULO DE LAS VELOCIDADES
Input: $T_{m_1}, T_{m_2}, T_{m_3}, T_{m_4}$
Output: X_G, Y_G, θ
$X_G(1) = x_{g_i}, Y_G(1) = y_{g_i}, \theta = \theta_i$
$X_G(0) = 0, Y_G(0) = 0, \omega = 0$
$v_{r_1}(1) = 0, v_{r_3}(1) = 0, v_{r_3}(1) = 0, v_{r_4}(1) = 0$
$T_{m_1}(1) = 0, T_{m_3}(1) = 0, T_{m_3}(1) = 0, T_{m_4}(1) = 0$
$t = 0, \quad t_k = t$
while $t \quad \neq \quad t_f$ **do**
$Leer : T_{m_1}, T_{m_2}, T_{m_3}, T_{m_4}$
$\ddot{X}_r, \ddot{Y}_R, \theta$
$\dot{X}_R(t_k) = \dot{X}_R(t_k - 1) + \ddot{X}_R(t_k) \cdot t_s$
$\dot{Y}_R(t_k) = \dot{Y}_R(t_k - 1) + \ddot{Y}_R(t_k) \cdot t_s$
$\dot{\theta}(t_k) = \dot{\theta}(t_k - 1)\ddot{\theta}(t_k) \cdot t_s$
$v_R = WheelMecanum([\dot{X}_R, \dot{Y}_R, \dot{\theta}])$
$t = t + t_s$
end while
$\theta_G(t) = \theta_G(t_k) + \dot{\theta}_G(t_k) \cdot (t - t_k)$
$X_G(t) = X_G(t_k) + \dot{X}_G(t_k) \cdot (t - t_k)$
$Y_G(t) = Y_G(t_k) + \dot{Y}_G(t_k) \cdot (t - t_k)$

donde, $T_{m_i}(t_k)$, $i(1, ..., 4)$ son los torques que se buscan que tengan los motores, estos valores se definen y se ajustan para tener más control con el algoritmo, $\ddot{X}_R(tk)$, $\ddot{Y}_R(t_k)$ y $\ddot{\theta}$ son aceleraciones definidas. $\dot{X}_R(t_k)$, $\dot{Y}_R(t_k)$ y $\dot{t_k}$ son las velocidades producto de las integrales en la aceleración. El algoritmo proporciona la salida con las poses del robot omnidireccional. Los datos de entrada definen mediante un signo hacia donde se realiza el movimiento, la función *WheelMecanum* es la expresión (35) para obtener la velocidad de cada rueda, se obtendrán 4 velocidades, con la velocidad total del robot, como parámetro.

4.1.3. Implementación

Para la implementación de los dispositivos de *hardware* se utilizó una tarjeta NVIDIA Jetson TX2 para generar el control mediante visión utilizando la cámara ZED, una tarjeta *BeagleBone Blue* para realizar el control de la base omnidireccional ya que el Robot *Robomaster S1* no tiene la capacidad de funcionar sin los componentes que lo componen, además de que no se puede programar con dispositivos externos. Por esta razón se desarrolló un control de la base omnidireccional reconfigurando el control que se tenía con las ruedas *mecanum* mediante un algoritmo de cinemática dinámica para cada uno de sus desplazamientos omnidireccionales.

Robot omnidireccional Robomaster S1

Robomaster S1 es un robot omnidireccional creado para sacar el máximo a los variantes desplazamientos que un robot omnidireccional puede realizar con las ruedas tipo *mecanum*, ver Figura 4.13. Las características del robot Robomaster S1, se muestran en la Tabla 4.2.

Tabla 4.2: Especificaciones de las características del robot omnidireccional

Robot omnidireccional Robomaster S1	
Característica	**Descripción**
Velocidad angular máxima	104,72 ~ rad/s
Dimensiones	35 cm x 25 cm x 15 cm
Tipo de control de chasis	ESC / FOC
Tipo de ruedas	*Mecanum*
Capacidad de la batería	2400 mAh
Voltaje de carga promedio	11,7 V
Conectividad	Wifi / Red local, por medio de un enrutador

Figura 4.13: Robomaster S1 DJI *(Company, 2020)*.

GPU Jetson TX2

Para lograr que el algoritmo de control del robot funcione en tiempo real, de modo que pueda procesar información de una cámara estéreo, es utilizando una GPU en este caso la Jetson TX2, ver Figura 4.14, la cual es utilizada en aplicaciones de control y navegación autónoma de robots móviles, fueron sus características que fue la razón por la que resultó la mejor opción para la tarea, como se describen en la Tabla 4.3.

Tabla 4.3: Características de la GPU NVIDIA Jetson TX2

Caraterísticas	Descripción
GPU	256-core NVIDIA PascalTM GPU arquitectura con 256 NVIDIA CUDA cores
CPU	Dual-Core NVIDIA Denver 2 64-Bit CPU Quad-Core ARM Cortex -A57 MPCore
Memoria	8GB 128-bit LPDDR4 1866 MHx - 59.7 GB/s
Almacenamiento	32GB eMMC 5.1
Energía	7.5W / 15W

Figura 4.14: GPU Jetson TX2 (Jetson TX2, *s.f.*).

BeagleBone Blue

Para tener un control del robot omnidireccional se implementó la tarjeta con IMU integrada *BeagleBone Blue*, (ver Figura 4.15) para utilizar tanto los datos inerciales como parte de control en desplazamientos rotacionales de la plataforma así como sus velocidades velocidades y desplazamientos, como se mencionó anteriormente, el utilizar una plataforma ya armada como la *Robomaster S1* tiene desventajas ya que no tiene la capacidad de implementar dispositivos externos a los que los que poseé de fábrica, así como para generar el control de navegación y para poder implementar su plataforma omnidireccional, fue necesario reconfigurar el control con un algoritmo de cinemática dinámica, dentro de este dispositivo. Las características de la *BeagleBone Blue* se muestran en la Tabla 4.4.

Tabla 4.4: Características de la *BeagleBone Blue*.

Característica	Descripción
Procesador	Octavo OSD3358 SiP con TI Sitara AM3358 a 1 GHz con 2 MCU PRU de 32 bits y 200 MHz
Memoria RAM	512 MB de RAM DDR3 (en SiP); Flash eMMC de 4 GB; puerto MicroSD
Inalámbrico	802.11bgn (2,4 GHz) y Bluetooth 4.1 BLE a través de TI WiLink 1835; ext. antenas Interfaces de radio GPS y DSM2
Control de motor	Servomotor 8 x 6V salidas (4 A reg.) 4 x DC salidas de motor (4 A reg.) 4 x entradas encoders
Sensores	IMU (acelerómetro, giroscopio y magnetómetro) para 9 ejes.
Energía de entrada	Conector JST-XH de batería LiPo de 2 celdas (2S) con equilibrio: conector de 9-18 VCC; micro USB
Sistema Operativo	Debian con kernel en tiempo real; Ubuntu Core; admite ROS, ArduiPilot, Cloud9 IDE en NodeJS

Figura 4.15: Tarjeta *BeagleBone Blue (*BeagleBoard.org - blue, *s.f.)*.

Cámara ZED

Se implementó una cámara ZED, (ver Figura 4.16) para desarrollar la visión del robot omnidireccional, como dispositivo para la obtención de información en el entorno resultó más eficiente que la detección mediante una sola cámara dado que al obtener un mapa profundidad del entorno en tiempo real, se pudo determinar las distancias próximas dentro de la trayectoria del robot, además de que describe mejor la profundidad en el eje z que el implementar una sola cámara. Dado que requiere una GPU para procesar toda la información obtenida, el uso de la Jetson TX2 beneficio considerablemente esta tarea y aceleró el proceso. Las características de la cámara ZED se describen en la Tabla 4.5.

Tabla 4.5: Características de la Cámara ZED

Caraterísticas	Descripción
Detección 3D	Detección 3D de largo alcance hasta una distancia de 20 my con mayor precisión de corta
Cámara dual 4MP	Captura de video 2K 3D en escenarios con poca luz.
Captura de video	Captura video HD 1080p a 30 FPS o WVGA a 100FPS.
Tasa de actualización de pose	Hasta 100 Hz
Tecnología	Odometría visual basada en profundidad en tiempo real y SLAM
Energía	5V / 380mA

Figura 4.16: Cámara ZED (ZED Stereo Camera, *s.f.*).

Ya mencionados los dispositivos de *hardware* requeridos para construir el robot omnidireccional, tanto el módulo inteligente, el control de la base robótica y la obtención de información visual. Para el desarrollo del sistema se contó con las siguientes especificaciones de *software*, ROS Medolic configurado para la Jetson TX2, *Open Motion Planning Library*, SDK ZED python, PyIT2FLS, Debian-*BeagleBone-Blue*, rospy *in* ROS, VNC *Viewer* NVIDIA. El *hardware* complementario es para el desarrollo del sistema se muestra en la Tabla 4.6.

Tabla 4.6: Dispositivos complementarios y sus especificaciones.

Nombre	Características	Descripción	Imagen
Display Raspberry Pi.	5' pulgadas, HD *Touchscreen.*	Dispositivo para visualizar información y comandos de ejecución.	
Batería Hyundai.	8100 Mah de salida, 19V/3.5A.	Fuente de energía para la GPU y Display.	

Se utilizó material como conectores de 4 micro-pin JST para realizar conexiones con la *BeagleBone Blue*, conectores para el *Display* tipo HDMI, conectores *plug* para realizar la

conexión a la fuente de energía, entre otro dispositivo periférico como un teclado para ejecutar algunos comandos para inicializar la conexión entre dispositivos. La visualización se realizó en una computadora HP, Procesador i3 8va Generación, 8GB de RAM.

En la Figura 4.17, se representan en forma de diagrama de bloque las conexiones que se realizaron con los mencionados dispositivos de *hardware*.

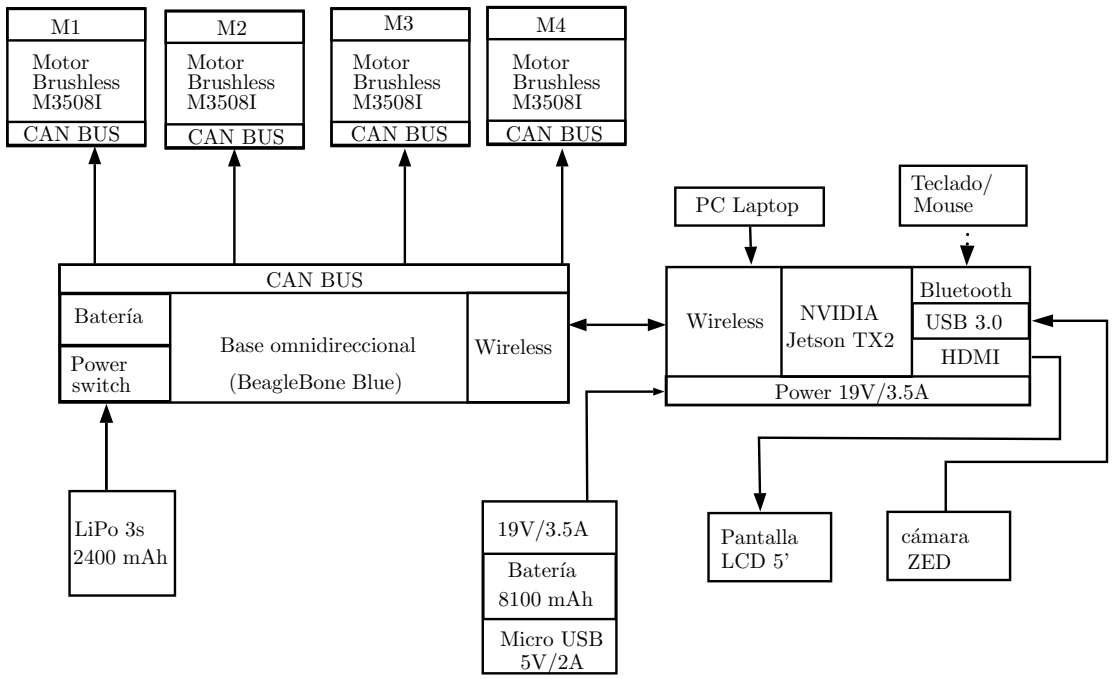
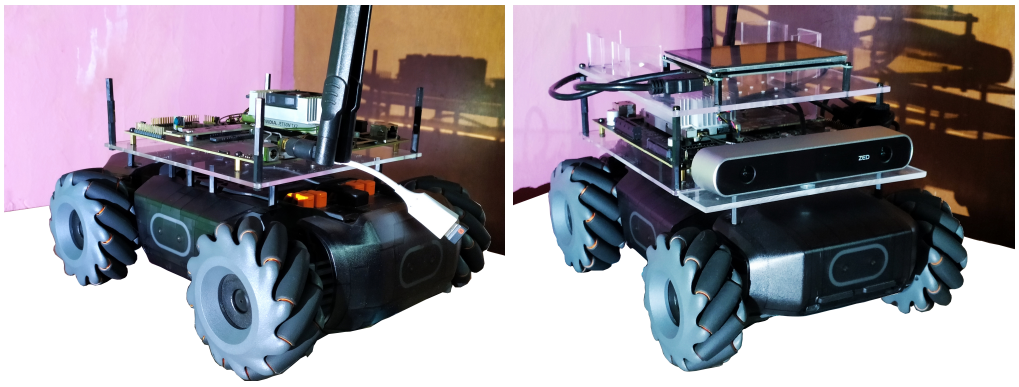


Figura 4.17: Diagrama de conexiones del robot omnidireccional tipo *mecanum*.

Del esquema anterior, las conexiones que implementaron para el desarrollo de la plataforma omnidireccional en la que se desempeñó el algoritmo propuesto, se clasifica en dos bloques principales, el bloque de control omnidireccional el cual es implementado en la tarjeta *BeagleBone Blue* aplicando la ecuaciones de cinemática robótica descritas en la Sección Modelo cinemático y el bloque de control de navegación mediante Lógica Difusa Tipo-2 implementado por la GPU Jestson TX2. En la *BeagleBone Blue* hay cuatro conexiones de tipo físico y una quinta de tipo inalámbrica. Se conectan los motores M3508I a través del conector CAN con conectores de 4 pines mini-JST, una conexión inalámbrica haciendo uso de nodos en ROS, una entrada de energía en forma de *plug* para conectar la salida de la batería de 19v/3.5A. El segundo bloque describe las conexiones de los periféricos de la Jetson TX2, salvo la pantalla LCD 5' pulgadas, los demás dispositivos solo se emplearon para hacer las configuraciones del algoritmo de navegación, ya que se visualizaron los datos en una computadora HP de manera inalámbrica. La conexión mediante nodos de ROS, vincula ambos módulos para su funcionamiento en conjunto. En la Figura 4.18 se observa el robot armado físicamente.



(a) (b)

Figura 4.18: (a) Plataforma robótica montada con la tarjeta NVIDIA Jetson TX2 y base ominidireccional. (b) Robot omnidireccional completo tipo *Mecanum*.

4.2. Instalación y ejecución de los nodos dentro del Sistema Operativo Robótico (ROS)

Al conectar los componentes del robot omnidireccional, se implementó la estructura de paquetes y nodos de ROS, tomando como base algunos paquetes disponibles dentro del Sistema Operativo Robótico distribución *Melodic*. Dado que ROS proporciona una extensa documentación de los paquetes disponibles. Además de que la navegación visual mediante una cámara ZED, proporciona la visualización de la información censada a través de RVIZ.
Para iniciar con las configuraciones básicas para trabajar con el robot omnidireccional sobre ROS, se encendió el robot y se inicializó ROS en una terminal. Dado que ya se cargaron las variables de entorno esenciales para la comunicación no es necesario correr las lineas de código ya que después de la propia instalación de ROS se inicializaron.

```
$ cd  /catkin_ws
$ catkin_make -DCMAKE_BUILD_TYPE=Release
$ source ./devel/setup.bash
```

En la sentencias anteriores se compila el espacio de trabajo, del propio proyecto para generar los archivos ejecutables.

```
$ source /opt/ros/melodic/setup.bash
$ roscore
```

En las instrucciones anteriores, la primera linea carga el archivo setup.bash con las

primeras configuraciones de comunicación y después se inicializa el nodo *Master* con la segunda linea.

```
$ rosrun bbblue_drivers imu_pub_node
$ rosrun robot_upstart install mecanum/launch/core.launch
```

Las dos lineas anteriores se ejecutan en dos diferentes terminales para ejecutar los archivos de comunicación. La primera linea ejecuta los *drivers* para incorporar el sensor IMU de la tarjeta *BeagleBone Blue*. El segundo inicializa la comunicación entre los nodos de la IMU y la cámara ZED.

```
$ roslaunch zed_display_rviz display_zed.launch
$ roslaunch mecanum_robot navigation.launch
```

Las dos instrucciones anteriores ejecutan múltiples nodos para generar una navegación omnidireccional. Del mismo modo se ejecutan en dos terminales distintas. La primera instrucción ejecuta la visión a través de la cámara ZED además de que se puede visualizar los datos durante la navegación.

4.3. Discusión

En este capítulo se describió el sistema de navegación así como cada una de sus etapas que lo componen, se muestra la construcción y un diagrama a bloques de las conexiones generadas para desarrollar el robot omnidireccional, dicho diagrama esquematiza de manera general las conexiones entre los componentes de *hardware*, la cámara ZED, GPU Jetson TX2, *BeagleBone Blue* y la plataforma Robomaster S1 así como la versión final del robot ensamblado. Se muestran los comandos que se ejecutaron para iniciar el funcionamiento de la visión estéreo como el inicio del control de la base robótica con la *BeagleBone Blue*, al ejecutar los comandos en el orden indicado el sistema comienza a adquirir la información de la cámara y de la odometría del robot en el espacio, junto con la lectura de datos hacia el sistema en la Jetson TX2.

Capítulo 5

Pruebas y Resultados

En esta sección se detallan los experimentos para evaluación del sistema y resultados obtenidos, del mismo modo se describen las métricas que se emplearon para validar los resultados de los experimentos, las cuales se implementan para evaluar la navegación en robots móviles.

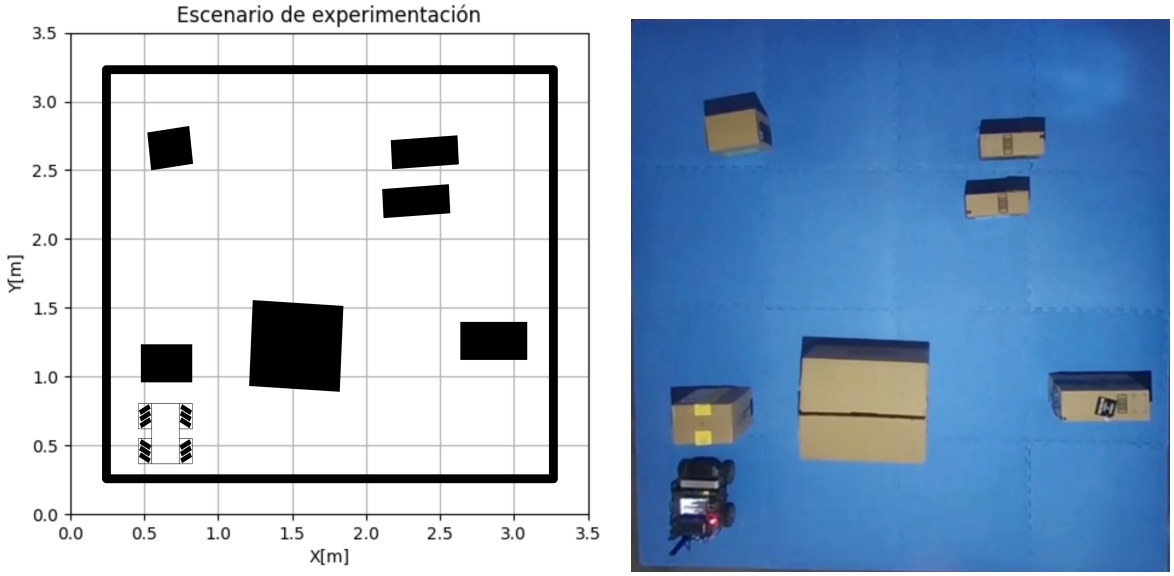
5.1. Entorno de experimentación

Para realizar la implementación de este sistema de navegación fue necesario contar con una plataforma de *software* robótica capaz de agrupar y controlar varios dispositivos. La forma de implementar este tipo de arquitectura física fue a través de los nodos que permite configurar el Sistema Operativo Robótico (ROS), este provee controladores, librerías, administración en forma de paquetes entre emisores y receptores, al igual que otras funciones. Para poder utilizar este sistema en la Jetson TX2 se instaló una versión la cual queda documentada como prueba porque la arquitectura no está pensada en ARM sino hasta la versión ROS2, es por ello que se implementa una fracción de este sistema operativo.

En cuanto al software implementado fue la versión Xubuntu 18.4 para una PC que sirvió como nodo central para monitoreo del funcionamiento durante la navegación de este robot, Ubuntu Jetson TX2 la cual es una arquitectura que maneja el JetPack que se instaló en la Jetson, este requirió de una actualización de los controladores de la GPU. ROS *Melodic* en la Jetson TX2 y en la PC para operar la comunicación entre nodos el algoritmo de navegación se creo bajo la arquitectura de ficheros de este sistema y el cual fue lanzado el 23 de Mayo de 2018.

La arquitectura de *hardware* para el monitoreo del robot móvil es una *Laptop* HP, con un procesador Core i3 Serie 7020U, RAM 8GB, la cual solo funcionó para genera un monitoreo de los datos durante la navegación y visión de información de la cámara ZED. El escenario es de forma cuadrada de 2.54$\mathbf{m}$ por lado, lo cual da un área de 5.9049$\mathbf{m^2}$

ideal para desplazarse en cortos lapsos de tiempo debido al costo energético por el *hardware* implementado, tal y como se observa en la Figura 5.1.



(a) (b)

Figura 5.1: Ejemplo del escenario de experimentación.

5.2. Métricas de evaluación

El desempeño de un robot puede ser evaluado en términos cuantitativos. Sin embargo, hay que tener en cuenta que las trayectorias del robot tienen una constante actualización en un entorno dinámico o cambiante. Para ello se planten dos métricas, Periodo de control (LeM) y Energía de curvatura (B_e).

Métricas de seguridad

Describen la seguridad del robot mientras se desplaza generando una trayectoria, teniendo en cuenta la distancia entre los obstáculos que puedan existir en el escenario.

Métricas para dimensiones espaciales

La trayectoria hacia la meta se considera en sus dimensiones de tiempo y espacio. En general, se supone que una trayectoria menos costosa hacia la meta, siempre que sea posible, es una linea con longitud mínima y curvatura cero entre el punto inicial (x, y) y el punto final (x_n, y_n) cubierto en el tiempo mínimo. **La longitud de la trayectoria cubierta** es una métrica que mide la longitud de toda la trayectoria cubierta por el

vehículo desde el punto inicial hasta la meta. Asumiendo el punto inicial como (x_1, y_1) y la meta como (x_i, y_i). Se puede calcular con la siguiente expresión *(Muñoz-Ceballos y Valencia-Velásquez, 2016)*.

$$P_L = \sum_{i=1}^{n-1} \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2} \tag{5.1}$$

donde, (x_i, y_i), $i(i = 1, 2, 3...)$ son puntos de trayectoria a en plano cartesiano.
Distancia media de la meta (Mgd): Esta métrica se aplica en robot los cuales se requiere comparar con una trayectoria definida. Como aspecto importante es determinar la calidad del sistema de navegación del robot. Para evaluar la calidad de la ejecución de la trayectoria, se analiza la distancia media entre el robot y el objetivo. La diferencia más significativa es si la distancia recorrida es más corta. Está definida por el cuadrado de la proximidad a la meta l_n integrado a lo largo de la trayectoria y normalizando por el número total de puntos n:

$$l_n = min(\sqrt{(x_i - x_n)^2 + (y_i - y_n)^2} \tag{5.2}$$

donde x_i, y_i es la posición inicial del robot y x_n, y_n es la posición final de la trayectoria.

$$Mgd = \frac{\int_0^1 l_n^2}{n} ds \tag{5.3}$$

donde, l_n es la proximidad hacia la meta dentro de la trayectoria, n es el numero total de posiciones dentro de la trayectoria *(Muñoz-Ceballos y Valencia-Velásquez, 2016)*.
Dado que se obtuvo un trayectoria generada con el control difuso una métrica que mida **error cuadrático medio** para saber que tanto se aproxima a la trayectoria planteada, nos dijo que tanto evitó el algoritmo los obstáculos en los escenarios y puesto que no se tiene una suavizado de la trayectoria con la métrica tiene que evaluar la posición con respecto al tiempo en cada punto del trayecto. La métrica CMSE viene dada como.

$$RMSE = \frac{1}{|f_i|} \cdot \sum_{\forall p_i \in T} (p_i - p')^2 \tag{5.4}$$

donde, f_i es una posición dentro de la trayectoria y f' es la posición deseada o esperada para demostrar un control estable y una navegación segura dentro del espacio de simulación *(Borreguero, 2017)*.

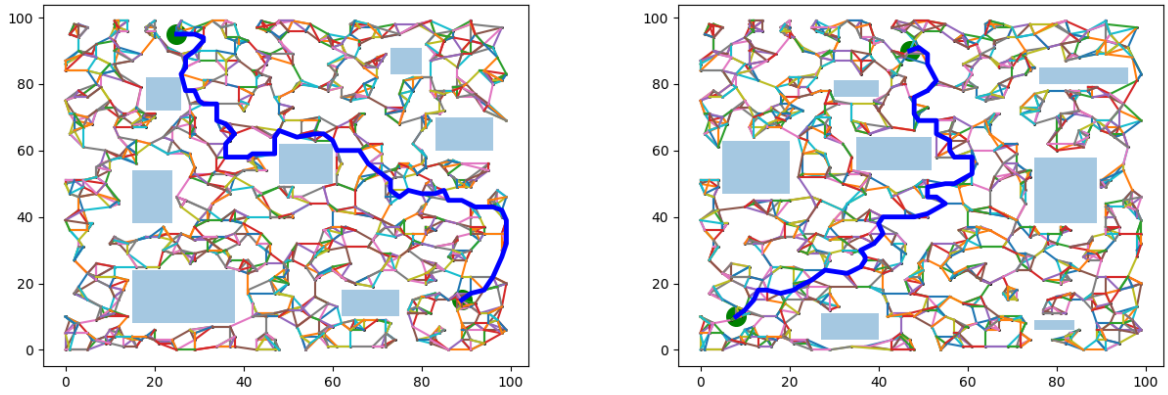
5.3. Pruebas

Los experimentos realizados para evaluar en conjunto al sistema propuesto para la navegación autónoma basada en Lógica Difusa Tipo-2 se dividieron en: control del seguimiento de trayectorias y evasión de obstáculos mediante el controlador basado en Lógica Difusa Tipo-2, navegación en entornos con obstáculos estáticos utilizando el algoritmo propuesto y en escenarios con objetos dinámicos.

5.3.1. Planificador de trayectorias

Para determinar la mejor trayectoria de un entorno, en el que se conocen los puntos meta relativos para poder desplazarse, es necesario planificar la trayectoria con algoritmos que puedan realizar la conexión de los nodos con menor costo y en menor distancia, esto al ir sumando estas trayectorias entre nodos se va obteniendo una trayectoria total la cual es definida por el algoritmo planificador como la mejor y más corta para llegar a la meta destino. El algoritmo que se planteó para esta tarea, es el PRM *(Probabilistic roadmap)*. Dado que este algoritmo tiene un buen funcionamiento en espacios no superiores a 10 metros su utilidad para un experimento de robótica en tiempo real resulta factible. Para esta prueba se configuró el algoritmo para un escenario con obstáculos estáticos esto debido que es más fácil graficar el comportamiento del algoritmo dentro del entorno, los experimentos con obstáculos dinámicos solo se muestra el cambio de la trayectoria inicia y se va graficando en tiempo real frente a situaciones dinámicas *(Klančar y cols., 2017)*.

Al obtener estas trayectorias se definió como la mejor trayectoria para llegar a la meta y por consiguiente el algoritmo de navegación propuesto se evaluó sobre de esta información, en la Figura 5.2 se muestran los experimentos generados en dos diferentes escenarios con el algoritmo planificador PRM.



(a) (b)

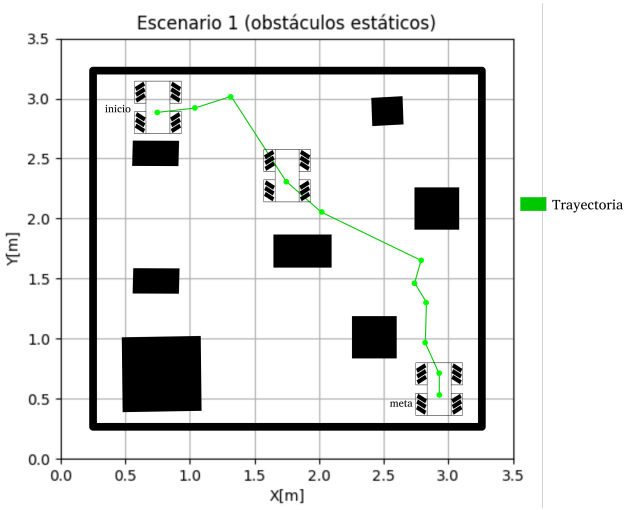
Figura 5.2: Experimentación sobre Escenario 1 y 2 con el algoritmo PRM.

Para el experimento anterior la distancia recorrida tomando como simulación el entorno trabajado se tiene para el escenario 1, $Distanciamediadelameta = 2.74944m$, para el escenario 2, $Distanciamediadelameta = 3.974m$. En los experimentos de definió la posición de cada obstáculo el algoritmo se generó puntos de manera aleatoria de los cuales se obtuvo una distancia de nodo a nodo y mediante una clasificación que realiza el algoritmo donde toma los nodos más cercanos dentro del radio del propio nodo pose este se va desplazando hasta alcanzar a la meta deseada. El algoritmo del robot va generando la trayectoria de esta manera y utilizando el control es capaz de realizar el seguimiento y en algunos experimentos como se observó el replantear un reajuste de trayectoria para recortar las distancias finales. Es como de esta manera se definió la mejor trayectoria para desplazarse en este entorno con obstáculos.

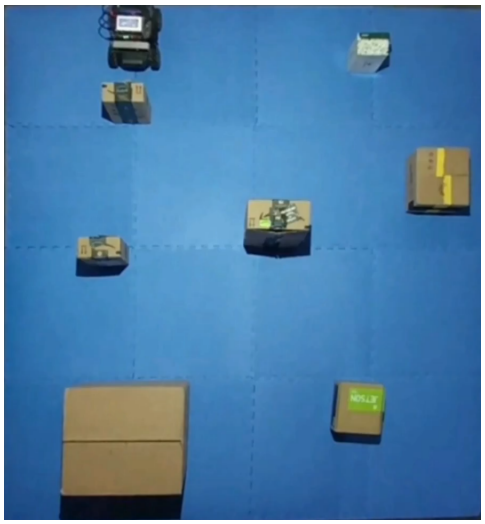
5.3.2. Experimentación de la navegación entre objetos estáticos

El algoritmo propuesto para el control dentro de la navegación de robots omnidireccionales se muestran a la izquierda la mejor trayectoria la cual es obtenida por el algoritmo planificador en escenarios con obstáculos estáticos y a la derecha se muestra el entorno real graficado a imagen binaria, como se observa en la Figura 5.3.

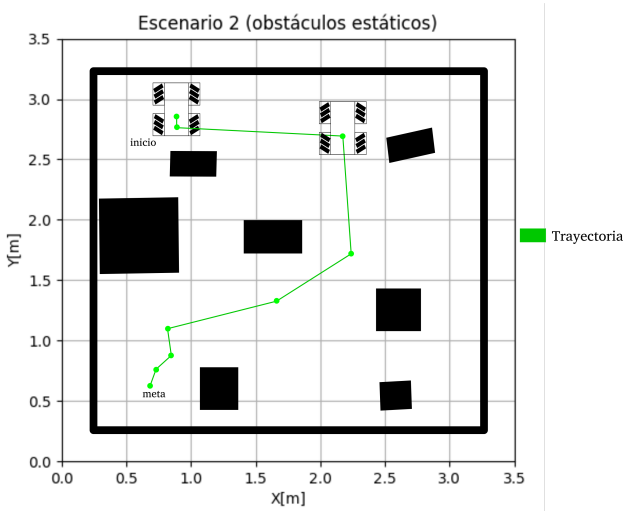
El objetivo de realizar este experimento es visualizar tanto la navegación utilizando la visión estéreo en un control de navegación basado en Lógica Difusa Tipo-2. Lo siguiente es determinar con las métricas de Longitud de trayectoria cubierta, como la Distancia media de la meta (Mgd), la cual sirve para demostrar si el algoritmo pudo realizar una planificación y seguimiento de trayectoria, tanto como la evasión de los obstáculos así como ir ajustando las velocidades lineales y angulares del robot.


Escenario 1 (obstáculos estáticos)
inicio
meta
Trayectoria
Y[m]
X[m]
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5

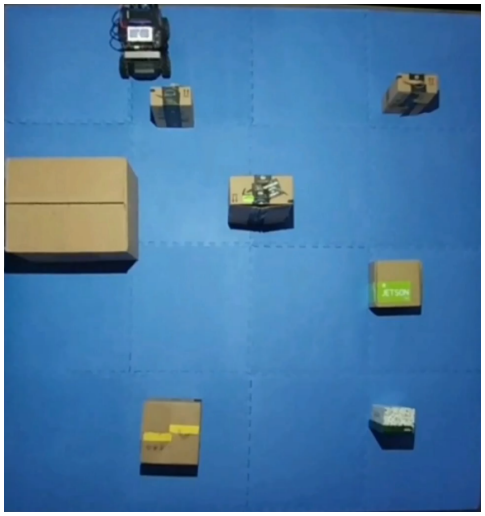
(a)



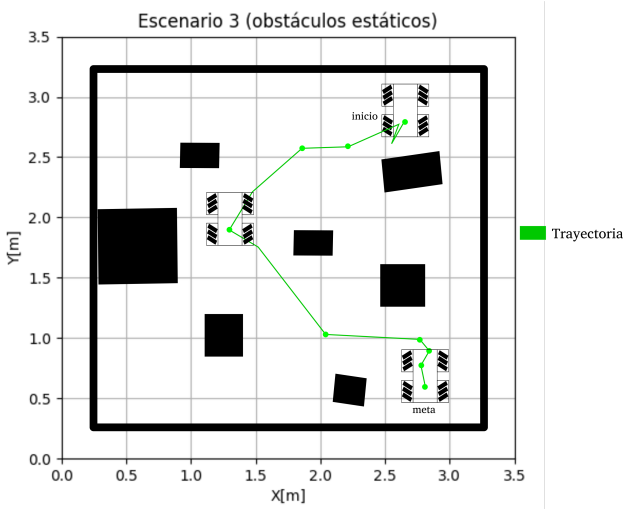
(b)


Escenario 2 (obstáculos estáticos)
inicio
meta
Trayectoria
Y[m]
X[m]
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5

(c)



(d)


Escenario 3 (obstáculos estáticos)
inicio
meta
Trayectoria
Y[m]
X[m]
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5

(e)

(f)

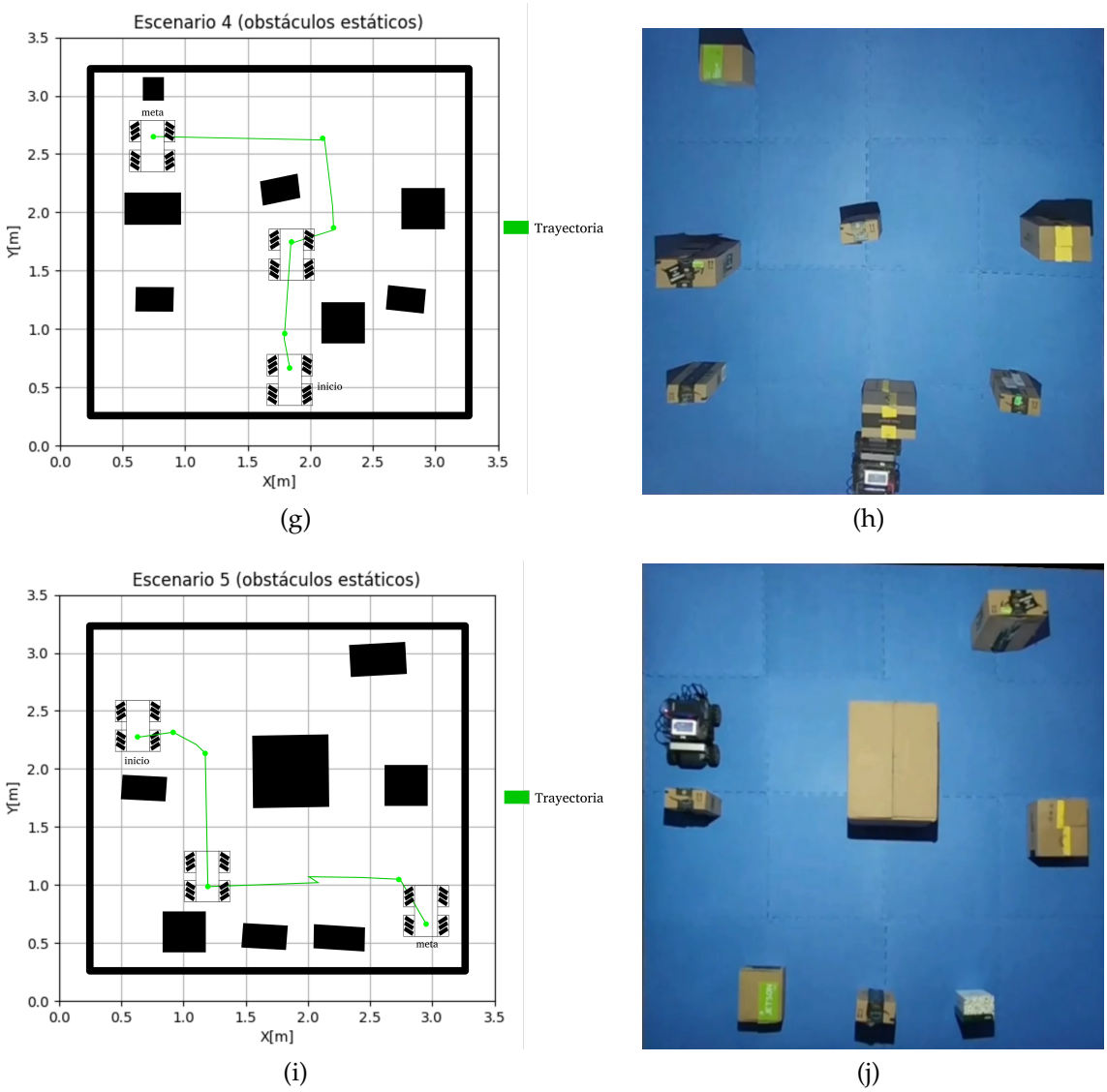


Figura 5.3: Experimentación sobre un escenario con obstáculos estáticos.

En la representación visual se representan las trayectorias escaladas obtenidas durante la experimentación en el entorno real dentro del apartado de la derecha de cada figura que muestra la trayectoria, en estos experimentos se muestran trayectorias cortas en un entorno con más de 4 objetos fijos, de esta manera se exige al robot a determinar la ruta más corta y segura para desplazarse evitando las colisiones con dichos objetos, dado al tipo de accionamiento del robot móvil sus desplazamientos fueron más directos y sin generar demasiadas curvas lo que generaría una complejidad computacional alta y lo cual se reflejaría en las métricas utilizadas. Los resultados cuantitativos de las trayectorias generadas se presentan en la Tabla 5.1.

Tabla 5.1: Resultados del Algoritmo Difuso Tipo-2 en los 5 trayectorias con 7 obstáculos estáticos.

Trayectoria	Tiempo (milisegundos)	Distancia de trayectoria (metros)	Distancia media de la meta (metros)	Error Cuadrático Medio (ECM)	Velocidad media (m/s)
T1	2.371	3.0494	3.178	0.824	0.24353
T2	2.619	3.097	3.153	0.835	0.24869
T3	3.8931	3.0263	3.1029	0.88703	0.24654
T4	2.9173	2.00417	2.01872	0.87394	0.24639
T5	2.1357	2.7243	1.71247	0.8987	0.24437

Analizando la tabla anterior, la característica del tiempo de ejecución del algoritmo muestra una respuesta rápida en reacción para controlar los desplazamiento del robot hacia la meta, aunado que el sistema embebido ejecuta solo dicha tarea y el control omnidireccional lo procesa la *BeagleBone Blue*. Como se mencionó, en esta experimentación la trayectorias fueron cortas, como se ve en la columna de distancia de trayectoria, esto para visualizar el nivel de respuesta en un entorno con una cantidad mayor de obstáculos y teniendo como primer análisis empleando el sistema propuesto. La distancia media de la meta en la mayoría de los resultados indicó que el robot realizó el seguimiento preciso, destacando la T3 y T4 ya que se obtuvo una mejor trayectoria y su seguimiento hacia la meta se completó sin desviación y colisión. El valor de error con respecto a la trayectoria trazada fue muy cercano al que el robot realizó en el seguimiento llegando muy cerca al 0.9 de similitud. Otro punto a destacar son las velocidades que se tienen durante la navegación y es que estas se mantuvieron tan cerca una de otra ya que le era fácil desplazarse sin tratar de ocasionar una colisión, es por ello que los valores resultaron tan parecidos.

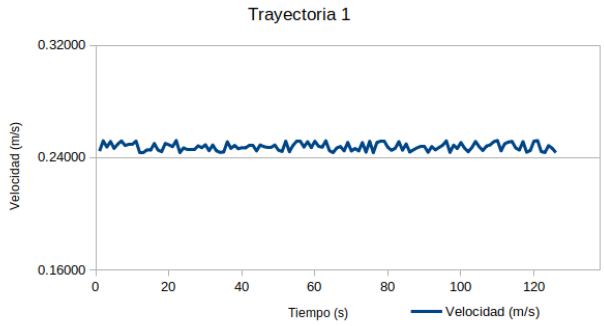
Resultados comparativos entre trayectoria del sensor IMU y Visión estéreo

Como punto adicional se realizaron experimentos para evaluar otro tipo de navegación utilizando el sensor inercial, haciendo uso de cada sensor en los mismos escenarios y así como sus objetivos meta. Se decidió representarlas con una tabla donde se agrupe la distancia de las trayectorias generada por ambos métodos de navegación y con la medición del error se establecerá la mejor opción en este tipo de experimentos, el resumen de esta experimentación se muestran en la Tabla 5.2.

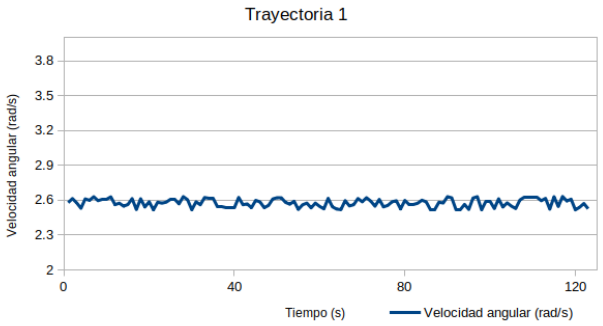
Tabla 5.2: Resultados de trayectorias generadas mediante el sensor inercial IMU, de los primeros 5 experimentos en el entorno con objetos estáticos.

Trayectoria	Distancia de trayectoria (metros)	Error Cuadrático Medio (ECM)
T1	1.924	0.8714
T2	1.387	0.8675
T3	2.4223	0.8853
T4	2.1607	0.8894
T5	2.7833	0.88187

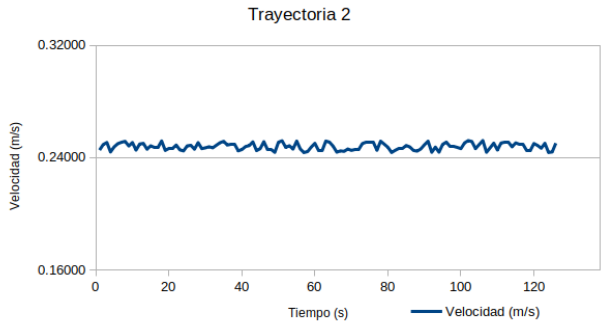
En estos primeros 5 experimentos ambas técnicas resultaron ser muy similares ya el error cuadrático tiende a 1 y la trayectoria por otro lado se llevo unos centímetros más a la generada mediante visión. Estos experimentos se realizaron con los mismos tiempos de navegación y objetos. A continuación en la Figura 5.4, se muestran las gráficas de las velocidades lineales y angulares que se midieron en cada experimento, lo cual de manera visual resultan ser valores no muy dispersos a velocidades constantes dado que el robot se encontraba en situaciones donde tenía que reaccionar rápido.



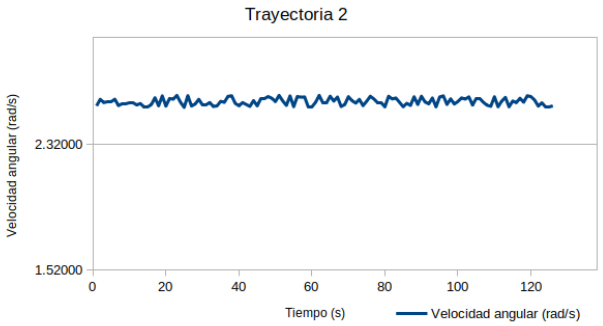
(a)



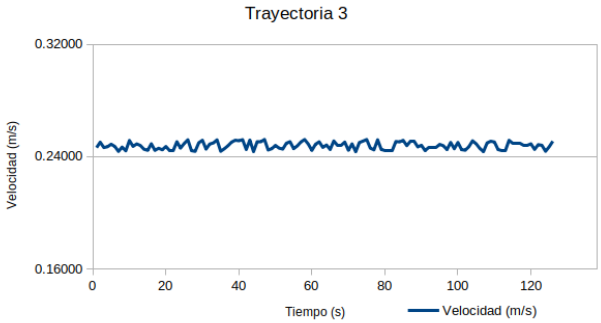
(b)



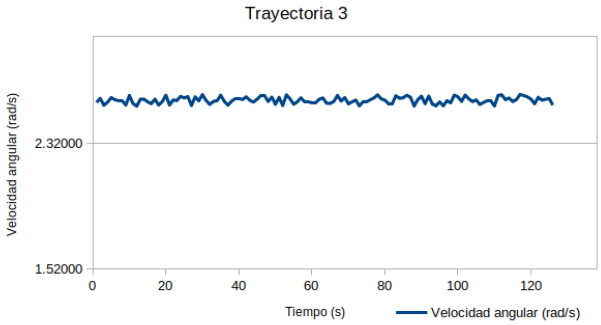
(c)



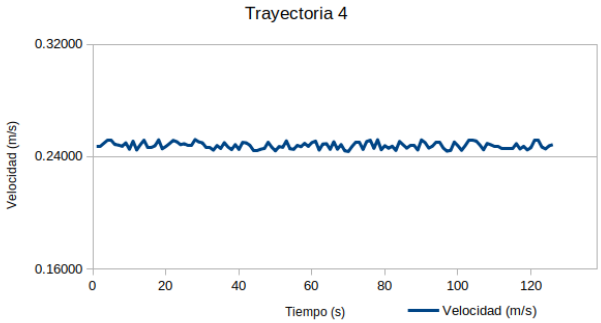
(d)



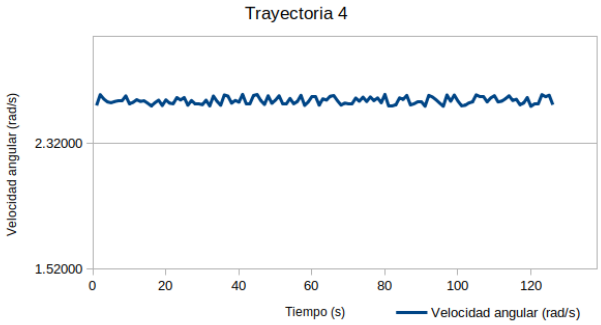
(e)



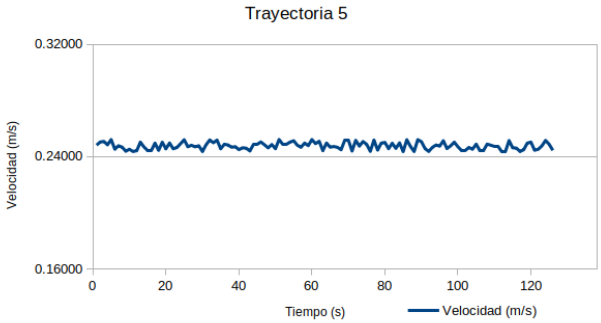
(f)



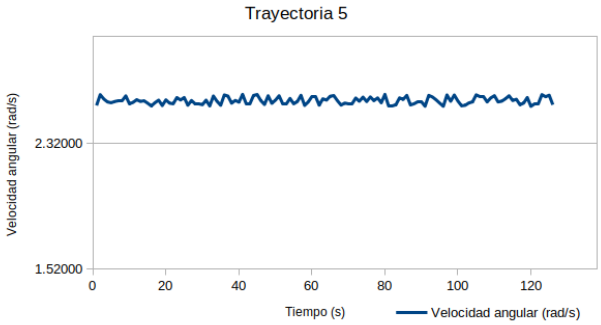
(g)



(h)



(i)



(j)

Figura 5.4: Velocidades de la experimentación en un escenario estático.

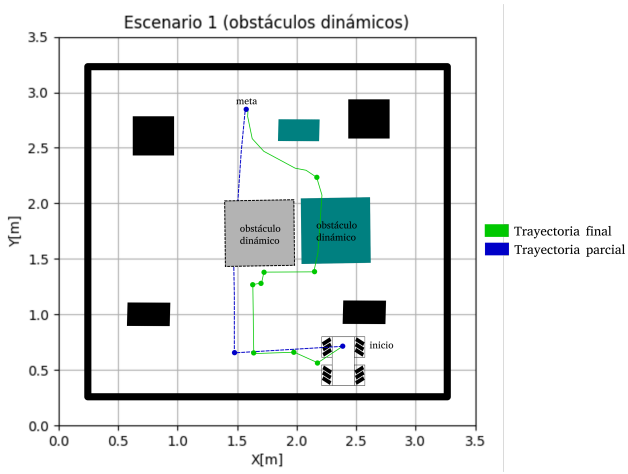
Durante estas pruebas no se hubo colisiones con obstáculos ya que en todo momento se tuvo las velocidades estables y junto con el sistema de visión estéreo se pudo evadir anticipadamente los obstáculos dentro del escenario.

5.4. Evaluación y análisis de los resultados

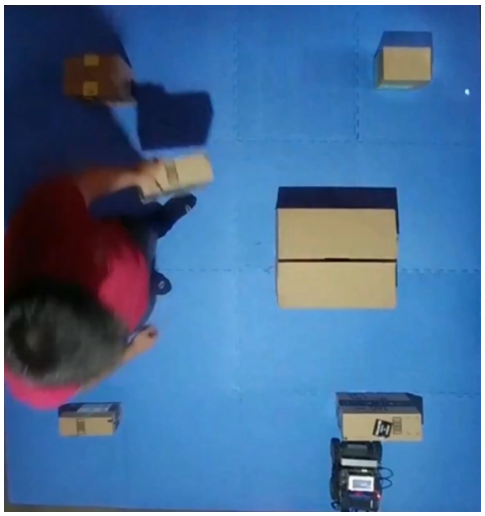
Los resultados se representan en la Tabla 5.1 donde se mide el control difuso tipo-2 por medio de la métrica de trayectoria media de la meta donde indica el seguimiento que realizó el robot en cada instante de tiempo. En este sentido un sistema de navegación robótica en la etapa de seguimiento de trayectoria debe realizar desplazamientos con mediante odometría visual y eso implementando ecuaciones de cinemática móvil se calcula la pose próxima más cercana a la cual desplazar y seguir con la trayectoria planificada, de este modo se evaden los obstáculos. En la navegación sobre el entorno descrito anteriormente, se plantearon 5 trayectorias, donde esas fueron determinadas por el algoritmo de navegación propuesto, se realizó su seguimiento y se obtuvo una distancia que indica lo recorrido, esta es calculada por medio de la sumatoria de poses en cada instante de tiempo, obteniendo así una distancia en función del mapa definido.

Experimentación de la navegación entre objetos dinámicos

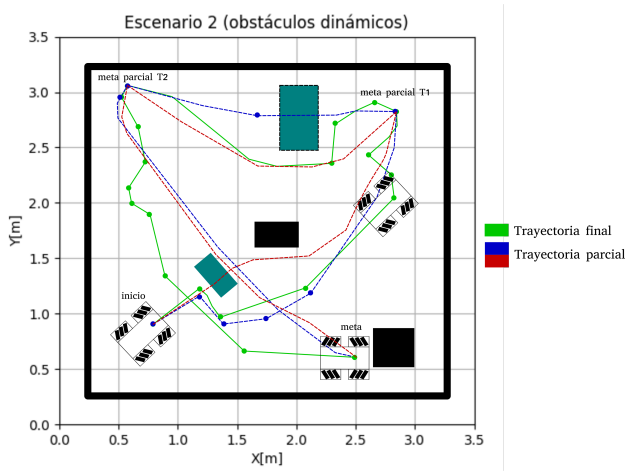
La experimentación se realizó en un entorno cuadrado de área $6.45m^2$ con un número de 6 obstáculos rectangulares con tamaños de entre $698.88cm^3$ a $11246.4cm^3$, donde los obstáculos dentro de este escenario fueron desplazados en instantes de tiempo aleatorio, de esta manera el robot demostró una evasión y re-cálculo de la trayectoria hacia la meta en algunos entornos no se emplearon todos los obstáculos sino que con menos de 6, estos se desplazaban entre el robot y su objetivo. La razón de este experimento es determinar que tan diferente es la trayectoria a la meta y como se desempeña el control durante el seguimiento de esa trayectoria en cada momento que un objeto se interpone en el camino y este afecta a su trayectoria inicial realizando un re-calculo de la misma.
La representación visual tanto en un gráfico como en una imagen del escenario real se muestran en la Figura 5.5, donde se agregó la etiqueta a cada ruta que el algoritmo tenía la posibilidad de desplazarse tomando en cuanta los objetos estuvieran frente a el o no.


Escenario 1 (obstáculos dinámicos)
meta
obstáculo dinámico
obstáculo dinámico
inicio
Trayectoria final
Trayectoria parcial
Y[m]
X[m]

(a)

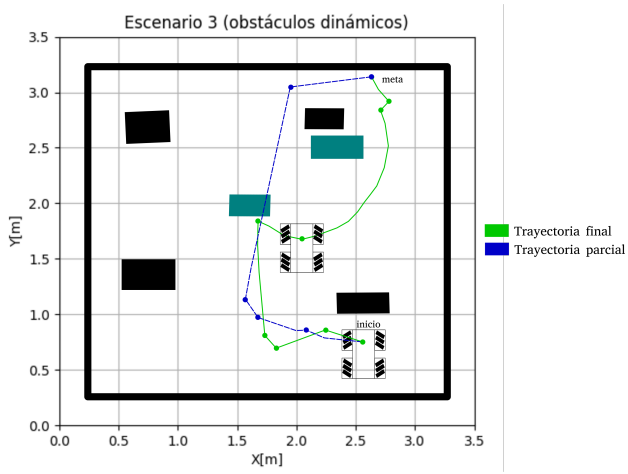


(b)

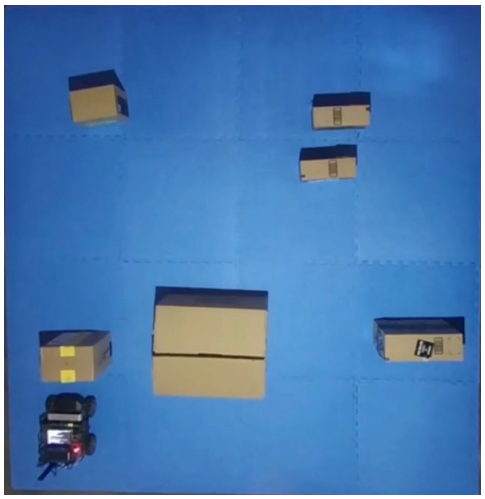

Escenario 2 (obstáculos dinámicos)
meta parcial T2
meta parcial T1
inicio
meta
Trayectoria final
Trayectoria parcial
Y[m]
X[m]

(c)

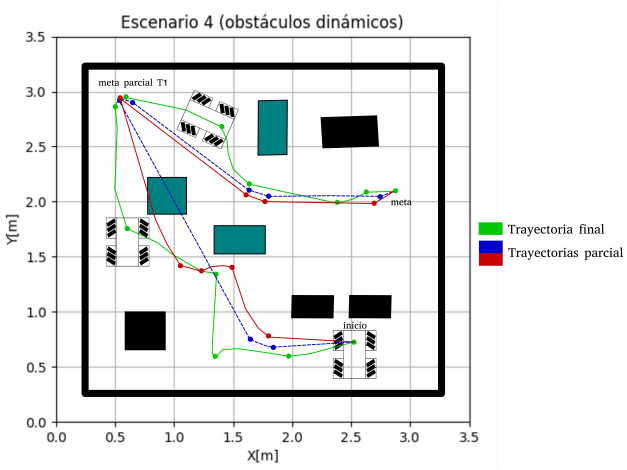
(d)


Escenario 3 (obstáculos dinámicos)
meta
inicio
Trayectoria final
Trayectoria parcial
Y[m]
X[m]

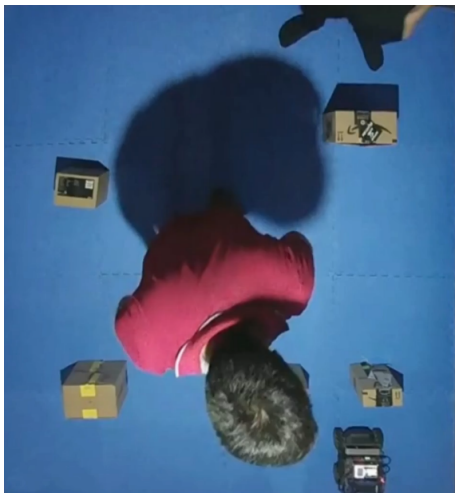
(e)



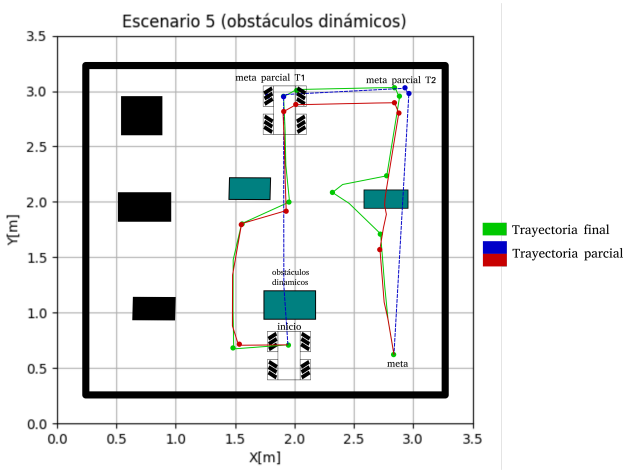
(f)



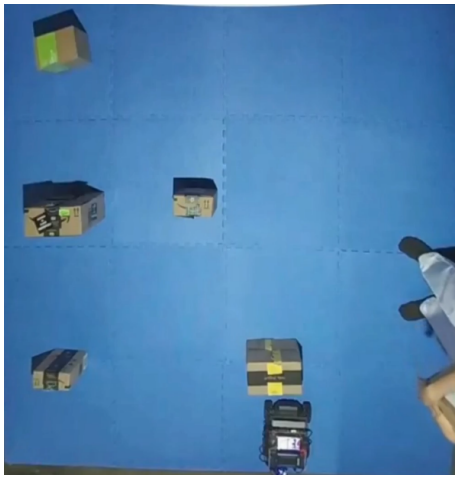
(g)



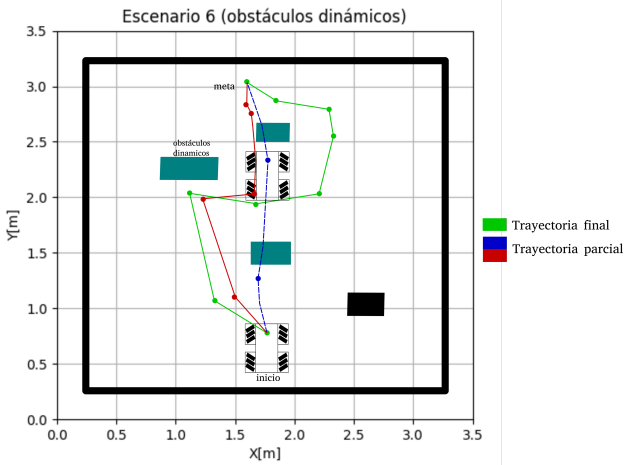
(h)



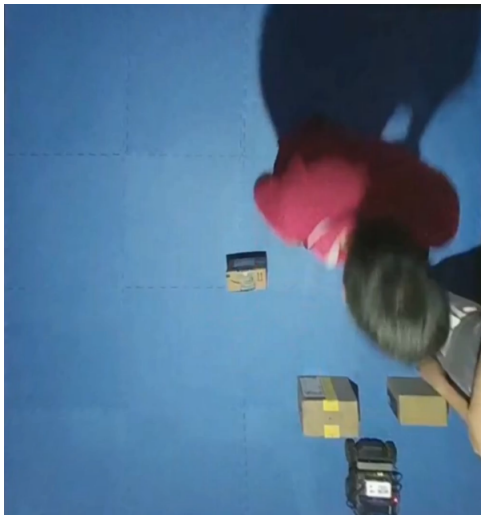
(i)



(j)



(k)



(l)

Figura 5.5: Experimentación sobre un escenario con obstáculos dinámicos.

En esta experimentación y como se muestra de manera visual con la figura anterior, se colocaron más de un punto meta en los experimentos, T2, T4 y T5 esto con el fin de analizar en un lapso de tiempo más largo el funcionamiento del control propuesto, dado que la trayectoria final es más larga se pudo observar su comportamiento sumado de nodo a nodo.

En el Experimento 2, se pudo analizar el comportamiento si un obstáculo cercano y dinámico está próximo a colisionar fue en este experimento donde se visualizó un desplazamiento en reversa para reajustar la trayectoria y avanzar a la meta tomando una nueva trayectoria más corta es un intervalo de tiempo que se puede observar en el gráfico de velocidades entre los 60 y 80 segundos del experimento.

Resultados

La experimentación con objetos dinámicos dentro del mismo escenario se realizó en dos partes una parte se enfocó en medir la relación que la mejor trayectoria y el algoritmo de navegación tenían tanto como evasión de obstáculo tomando una nueva trayectoria, como distancia generadas, correlaciones entre poses durante la navegación y la reacción del robot ante objetos dinámicos en función de las velocidades obtenidas del controlador difuso tipo-2, el resumen de esta experimentación se muestran en la Tabla 5.3.

Tabla 5.3: Resultados del Algoritmo Difuso Tipo-2 en los 6 trayectorias con 6 obstáculos dinámicos.

Trayectoria	Tiempo (milisegundos)	Distancia de trayectoria (metros)	Distancia media de la meta (metros)	Error Cuadrático Medio (ECM)	Velocidad media (m/s)
T1	2.2173	3.8871	3.8783	0.8217	0.24683
T2	2.98413	3.8317	3.9371	0.8625	0.24662
T3	3.01727	3.12547	3.23176	0.8649	0.2471
T4	2.8364	2.8729	2.8927	0.9037	0.24768
T5	2.8631	3.4278	3.5231	0.89913	0.2472
T6	2.9817	2.8971	3.0047	0.86173	0.24934

Algo que resalta, en los datos obtenidos son primeramente las distancias que se generaron para cada una de las trayectorias. La distancia de la trayectoria la cual representa la longitud cubierta por la planificación y la la distancia media de la meta, ambas fueron diferentes en la mayoría de los experimentos, esto se debió a que el robot al planificar y seguir una nueva trayectoria está captando su entorno y cualquier objeto al entrar a su campo de visión este toma la mejor decisión para evadirlo de manera controlada, si el algoritmo solo contase con un planificador de trayectoria este podría colisionar frente a este tipo de situaciones ya que la trayectoria calcula la trayectoria menos costosa en distancia sin tomar en cuenta el entorno. De igual manera los valores obtenidos en las

Tabla 5.4: Resultados con la comparación de trayectorias generadas por la visión estéreo e inercial en un entorno dinámico.

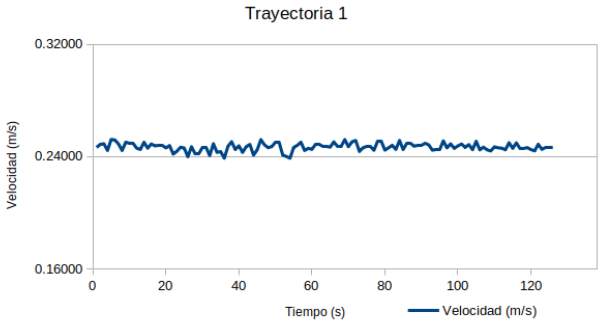
Trayectoria	Distancia de trayectoria (metros)	Error Cuadrático Medio (ECM)
T1	3.9571	0.8647
T2	3.9821	0.8025
T3	3.54134	0.80109
T4	2.9892	0.864201
T5	3.6871	0.8034
T6	3.00587	0.81013

velocidades lineales durante la navegación se mantuvieron lo más estables posible lo que indicaba que no habría desplazamientos muy inmediatos y con riesgo de colisionar.

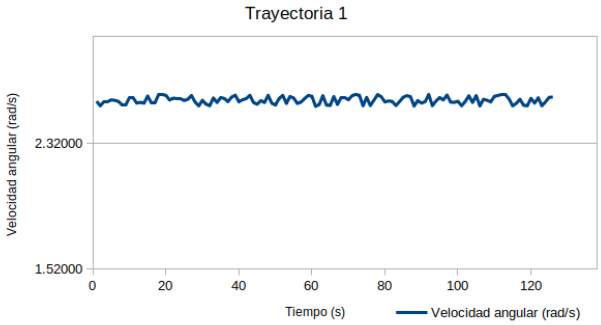
Resultados comparativos entre trayectoria del sensor IMU y Visión estéreo

Al igual que se realizó esta experimentación en los entornos con objetos estáticos se evaluaron en estos escenarios, sin embargo al evaluar las trayectorias donde los obstáculos son dinámicos se obtuvieron desplazamientos donde las distancias generadas por un sensor IMU, resultaran un poco largas a comparación de la navegación con visión estéreo y esto debido a que se mientras hubiese definido trayectorias más largas el sensor puede llegar a almacenar algo de ruido en cada desplazamiento que realiza, el resumen de esta experimentación se muestran en la Tabla 5.4.

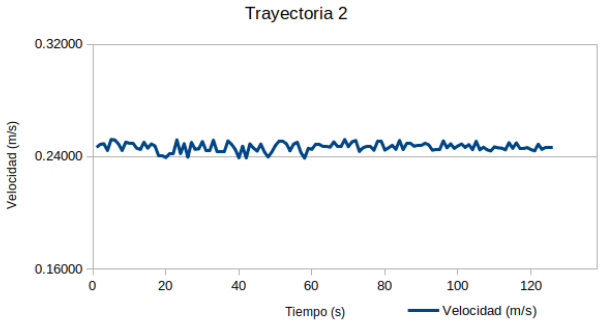
En estos últimos resultados la distancia de la trayectoria se vio un tanto diferente a la generada por la visión estéreo esto debido a que en este ejemplo los objetos son dinámicos la medida de comparación mostró una ligera diferencia entre trayectorias generadas. En conclusión a estos resultados obtenidos en los dos tipos de navegación, la visión estéreo resultó la más viable para utilizar, dado que proporciona más información del entorno que el sensor IMU por su cuenta. Sin embargo, las ventajas de diseñar e implementar un buen control con un sensor IMU se basan en utilizar sus datos en un algoritmo de cinemática móvil esto proporcionaría una mejor precisión y control en escenarios con obstáculos dinámicos.


Trayectoria 1
0.32000
0.24000
0.16000
Velocidad (m/s)
0
20
40
60
80
100
120
Tiempo (s)
Velocidad (m/s)

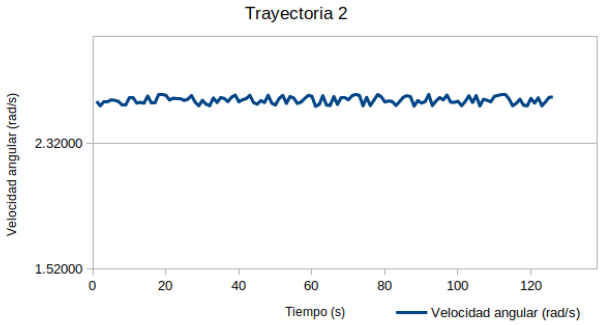
(a)


Trayectoria 1
2.32000
1.52000
Velocidad angular (rad/s)
0
20
40
60
80
100
120
Tiempo (s)
Velocidad angular (rad/s)

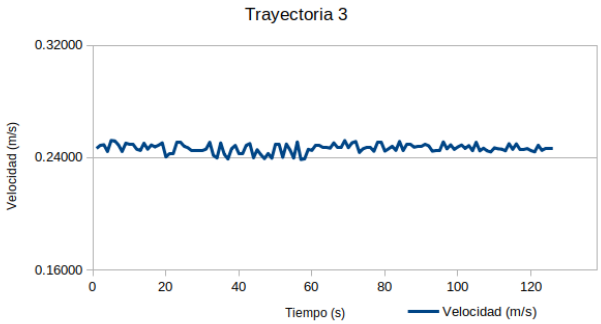
(b)


Trayectoria 2
0.32000
0.24000
0.16000
Velocidad (m/s)
0
20
40
60
80
100
120
Tiempo (s)
Velocidad (m/s)

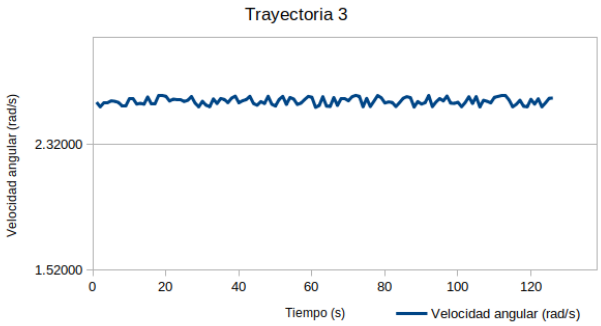
(c)


Trayectoria 2
2.32000
1.52000
Velocidad angular (rad/s)
0
20
40
60
80
100
120
Tiempo (s)
Velocidad angular (rad/s)

(d)


Trayectoria 3
0.32000
0.24000
0.16000
Velocidad (m/s)
0
20
40
60
80
100
120
Tiempo (s)
Velocidad (m/s)

(e)


Trayectoria 3
2.32000
1.52000
Velocidad angular (rad/s)
0
20
40
60
80
100
120
Tiempo (s)
Velocidad angular (rad/s)

(f)

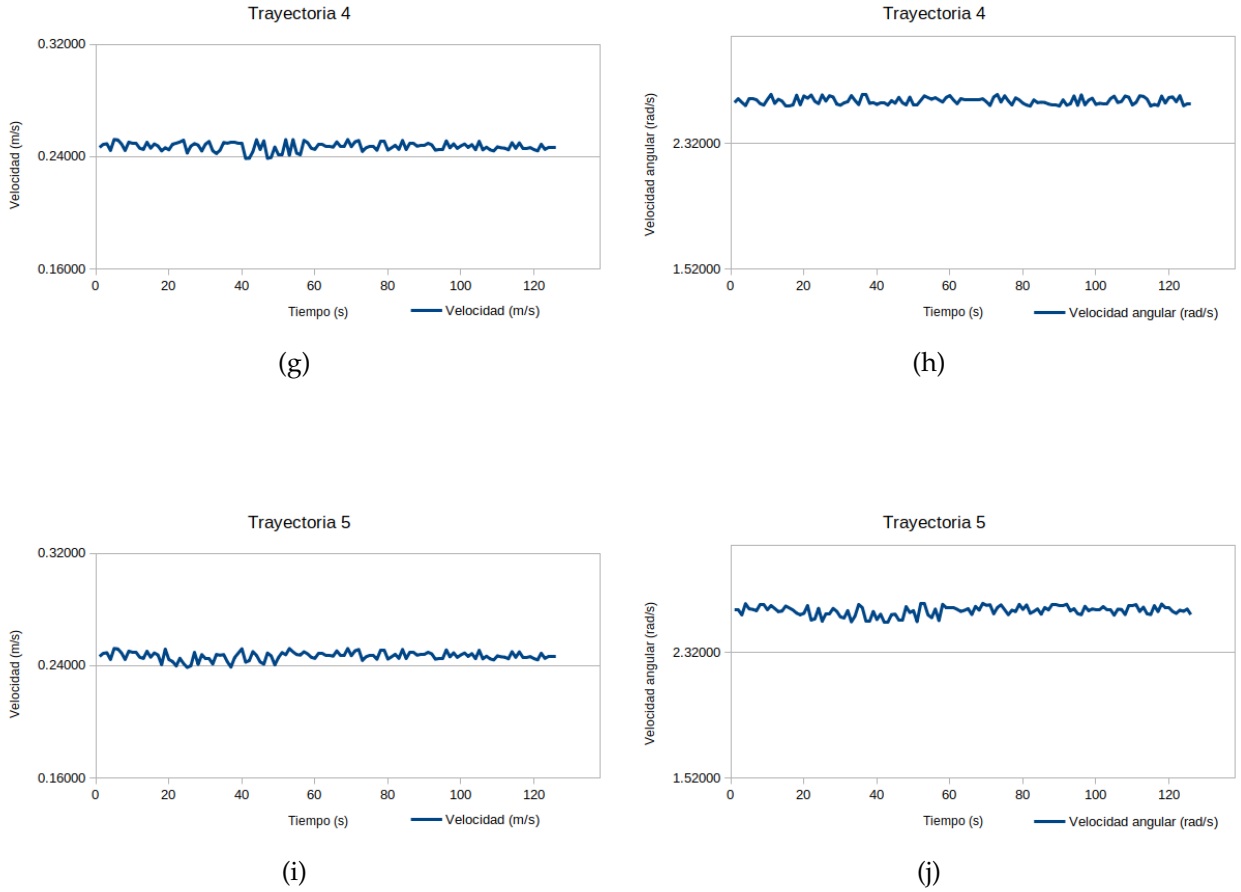


Figura 5.6: Velocidades obtenidas en la experimentación en escenarios con obstáculos dinámicos.

5.5. Discusión

En este Capítulo se detalló la parte del entorno de experimentación, así como las métricas para utilizadas para la evaluación del sistema de navegación para un robot omnidireccional basado en Lógica Difusa Tipo-2. Los experimentos se diseñaron y estructuraron para 3 categorías de evaluación, el primero es una prueba al algoritmo planificador de trayectoria, después la experimentación del controlador difuso tipo-2, y los dos últimos experimentos se enfocaron a dos tipos de escenarios, con obstáculos estáticos y dinámicos. En estos dos últimos experimentos se evaluó un sistema de navegación inercial para comparar las trayectorias de una odometría visual a una inercial. Como resultado principal en cada prueba se llegó a la meta establecida dentro del escenario las distancias entre los nodos inicio y fin se obtuvieron las trayectorias más cortas posibles. Como conclusión del tipo de navegación visual implementada, el sensor estéreo requirió un alto nivel de procesamiento para realizar la detección y evasión de obstáculos en cada momento por

lo que los trayectos por desventaja del *hardware* resultaron cortos de tiempo de ejecución, significativamente entre 5 a 10 minutos para realizar las pruebas con la energía completa. En general esta navegación resultó eficiente y práctica para implementar la Lógica Difusa Tipo-2 por su ventaja de manipulación de información sensorial, añadiendo que al usar el sensor inercial por su bajo costo computacional y energético es una buena opción para buscar una mejora en un futuro en el control de este tipo de robot.

Capítulo 6

Conclusiones

En este capítulo se realiza el análisis de los objetivos cumplidos después de implementar el sistema sobre un robot omnidireccional construido y realizar la experimentación junto con la discusión de resultados. Del mismo modo se realiza un análisis del trabajo futuro de este proyecto de tesis.

Tabla 6.1: Objetivos y alcances realizados

Objetivos	Actividad
Analizar y comprender la formalización matemática del movimiento omnidireccional.	Se estudió los diversos modelos cinemáticos robóticos para comprender la matemática de cada desplazamiento, el cuál ayudó en modificar e implementar un control de movimiento, en el robot omnidireccional de tipo dinámico para las físicas del robot.
Analizar y comprender la conceptualización de la Teoría Difusa Tipo-2 enfocada a la navegación.	Se estudió la teoría de los conjuntos tipo 1 y 2 (intervalo y general), se realizaron implementaciones con controles difusos de tipo 1, para comprender su función y desarrollar el IT2FLS.

Comprender y experimentar los datos inerciales que un dispositivo de medición inercial evalúa, implementando la tarjeta *BeagleBone Blue.*	Se implementaron dos sensores por separado uno de tipo inercial y la visión estéreo, para obtener la odometría del robot y generar un seguimiento de trayectorias, al final se implementó una navegación por visión estéreo como adquisición de información, para utilizar mapas de profundidad y generar el control de la navegación.
Diseñar el modelado del sistema de navegación autónoma, incorporando la metodología de solución.	Se generó un diagrama a bloques del sistema completo, para explicar cada parte de funcionamiento en el algoritmo de navegación para el robot omnidireccional.
Realizar la configuración y programación en las tarjetas *BeagleBone Blue* y GPU NVIDIA Jetson TX2 con la cámara ZED.	Se estudió y experimentó con una plataforma embebida NVIDIA Jetson TX2, para cargar el sistema de navegación basado en Lógica Difusa Tipo-2.
Evaluar los resultados durante la navegación del robot omnidireccional mediante métricas cuantitativas.	Se evaluó el sistema con métricas de la literatura enfocada a navegación en robots móviles entre ellas de tipo espacial para determinar trayectorias cortas y suaves de un punto inicio a uno meta.
Alcances	Actividad
Procesar en tiempo real los datos de profundidad de la cámara ZED.	Se utilizó un mapa de profundidad generado en tiempo real al cual se le extrajo información del escenario y obstáculos mediante los métodos del SDK y una expresión matemática.
Controlar la velocidad y aceleración del robot omnidireccional.	Se controló la velocidad con la salida del sistema difuso tipo-2 y mediante comunicación entre nodos se controló la base del robot omnidireccional generando su odometría visual para el seguimiento de trayectorias.

Determinar la pose del robot a partir del sensor inercial.	Se implementó el sensor IMU para generar trayectorias y controlando el seguimiento de las mismas haciendo uso de datos inerciales.
Evasión de obstáculos mediante el uso de una imagen de profundidad.	Se programó un módulo que utilizó el mapa de profundidad generado por la cámara ZED para obtener las distancias de los objetos al robot y así determinar el siguiente desplazamiento más corto a la meta para evadir el obstáculo.
Implementar el Controlador Difuso Tipo-2 sobre una GPU NVIDIA Jetson TX2.	Se diseñó y desarrolló un controlador IT2FLS con 2 entradas del mapa de profundidad, 12 reglas Mamdani, 14 MFs gaussianas tipo-2, reductor de tipo EIASC y dos salidas la cuales representan velocidad lineal y angular; este sistema se configuró sobre la GPU Jetson TX2.

6.1. Productos

Durante el desarrollo de este proyecto de tesis de maestría se obtuvieron los siguientes productos

- Reporte del estado del arte: documento en el que se presentan los trabajos relacionados con este proyecto de tesis. Este se entregó en segundo semestre.
- Diseño, desarrollo e implementación del algoritmo de navegación para un robot omnidireccional basado en Lógica Difusa Tipo-2.
- Algoritmo de navegación para entornos con obstáculos estáticos y dinámicos, con un robot omnidireccional.
- Artículo presentado en el CORE 2019. Titulado: *Comparative analysis of interest point detectors algorithms on Robotic Operative System*
- Artículo presentado en el congreso de la UNAM CEIAAIT, Titulado: Análisis comparativo de trayectorias en robots
- Artículo presentado en la Cuarta Jornada de Ciencia y Tecnología del CENIDET, Titulado: Navegación de un robot diferencial basada en una Red Neuronal Multicapa con el método *Backpropagation*.

- Artículo aceptado y publicado por el Encuentro Nacional de Computación 2021, Titulado: *Navigation of an omnidirectional robot with Type-I Fuzzy Logic using Takagi-Sugeno model*.

6.2. Aportaciones

Como aportaciones de tema desarrollado en este tema de tesis se obtuvieron:

- Controlador IT2FLS para la navegación de un robot omnidireccional.
- Prototipo completo del robot omnidireccional.
- Algoritmo de navegación para la implementación de entornos cerrados y con escenarios con objetos estáticos y dinámicos.

6.3. Trabajos futuros

Este tipo de trabajo se puede complementar con la implementación de fusión de datos sensoriales para determinar de manera más precisa la pose de un robot y calcular trayectorias con la menor distancia hacia la meta, al igual de que la navegación en diferentes escenarios cerrados con un numero no definido de obstáculos. De igual manera en la parte del control difuso tipo-2 experimentar con otros tipo de algoritmos de reducción de tipo más eficientes ya que en la literatura se encontró otro tipo de algoritmos con un buen desempeño en tiempo. La razón de desarrollar este tipo de robots es que en un futuro sirvan para desempeñar tareas cotidianas o dentro de un instituciones o empresas que requieran realizar tareas de traslado de equipo dentro de la empresa, es por ello que este trabajo se puede considerar para dar inicio a ese tipo de sistemas más complejos.

Referencias

Abiyev, R. H., Akkaya, N., y Gunsel, I. (2019). Control of omnidirectional robot using z-number-based. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *49*, 238-252. doi: 10.1109/TSMC.2018.2834728

Abiyev, R. H., Günsel, I., Akkaya, N., Aytac, E., Çağman, A., y Abizada, S. (2016). Robot Soccer Control Using Behaviour Trees and Fuzzy Logic. *ELSEVIER*, *102*, 477-484. doi: https://doi.org/10.1016/j.procs.2016.09.430

Abiyev, R. H., Günsel, I. S., Akkaya, N., Aytac, E., Çağman, A., y Abizada, S. (2017). Fuzzy control of omnidirectional robot. (Turkey). *ELSEVIER*, *120*, 608-616. doi: https://doi.org/10.1016/j.procs.2017.11.286

Andreev, A. S., Peregudova, O. A., y cols. (2019). On output feedback trajectory tracking control of an Omni-Mobile robot. (Russia). *ELSEVIER*, *52*(8), 37-42. doi: https://doi.org/10.1016/j.ifacol.2019.08.045

Arellano, I. J. A. G. (2014). *Fusión de datos de sensores mediante lógica difusa, aplicación en robótica móvil*. Cuernavaca, Morelos.: Centro Nacional de Investigación y Desarrollo Tecnológico.

Barrero, A. F. (2016). Algoritmo de navegación a bardo en ambientes controlados a partir de procesamiento de imágenes. (Colombia). *Tekhnê*, *12*(2), 23-34.

BeagleBoard.org - blue. (s.f.). `https://beagleboard.org/blue`. (Accessed: 2020-02-01)

Borreguero, L. D. E. L. (2017). *Evaluación de las técnicas slam disponibles en ros*. Cuernavaca, Morelos.: Centro Nacional de Investigación y Desarrollo Tecnológico.

Castro, S. A. H. (2019). *Sistema de odometría visual e inercial con un marcador*.

Chen, C., Zhao, P., Lu, C. X., Wang, W., Markham, A., y Trigoni, N. (2018, 09). Oxiod: The dataset for deep inertial odometry. *Cornell University*, 7. doi: https://arxiv.org/abs/1809.07491

Chen, Z., Mendes, A., Yan, Y., y Chen, S. (Eds.). (2018). *Intellingent Robotics and Applications*. 11th International Conference, ICIRA 2018. doi: 10.1007/978-3-319-97589-4

Company, D. (2020). *Dji robomaster ep.* Shenzhen, China. Descargado de `https://www.dji.com/mx/company?site=brandsite&from=footer`

Cuchango, H. E. E., y Méndez, J. J. S. (2010). A fuzzy inference system based on boolean relations. *UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS*, *15*, 52-66.

Cuevas, F., Castillo, O., y Cortés-Antonio, P. (2020). Omnidirectional four wheel mobile robot control with a type-2 fuzzy logic behavior-based strategy. *Intuitionistic and Type-2 Fuzzy Logic Enhancements in Neural and Optimization Algorithms: Theory and Applications.*, 49-62.

de Dios Flores Méndez, J. (2015). *Robot omnidireccional para slam y navegación en ros*.

Fahmizal, y Kuo, C.-H. (2017). Trajectory and heading tracking of a mecanum wheeled robot using fuzzy logic control. *2016 International Conference on Instrumentation, Control and Automation (ICA)*, 54-60.

Fahmizal, Rijalussalam, D. U., Budiyanto, M., y Mayub, A. (2019). Trajectory tracking pada robot omni dengan metode odometry. *JNTETI*, *8*, 35-45.

Fernández, I. A. M. R. (2019). *Evaluación del algoritmo theta* para la planeación de trayectorias*. Cuernavaca, Morelos.: Centro Nacional de Investigación y Desarrollo Tecnológico.

Flores, C., Arturo, I., y cols. (2017). Implementación de un robot móvil para optimizar el flujo documentario en el área de logística usando tecnologías de robótica autónoma basada en la plataforma de software libre ROS. (Perú). *TECNIA*, *27*(2), 93. doi: https://doi.org/10.21754/tecnia.v27i2.179

González, A. P. (2015). Implementación de odometría visual utilizando una cámara estereoscópica. *Universidad de Chile, Facultad de ciencias físicas y matemáticas departamento de ingeniería eléctrica*, 1-88.

González, C. G., y Felpeto, A. B. (2006). *Tratamiento de datos*. España: Ediciones Díaz de Santos.

Guamani, E. S. B. (2019). *Navegación autónoma basada en maniobras bajo estimación de posturas humanas para un robot omnidireccional kuka youbot*. Universidad Técnica de Ambato, Ecuador: Universidad Técnica de Ambato. Facultad de Ingeniería en Sistemas, Electrónica e Industrial. Carrera de Ingeniería Electrónica y Comunicaciones.

Islam, A., Hossan, M. T., y Jang, Y. M. (2018). Convolutional neural network scheme-based optical camera communication system for intellingent internet of vehicles. *International Journal of Distributed Sensor Networks*, *14*, 1-19.

Jamali, P., Tabatabaei, M., Sohrad, O., y Seifipour, N. (2017). Software based modeling, simulation and fuzzy control of a mecanum wheeled mobile robot. *RSIIISM International Conference on Robotics and Mechatronics*, 200-205.

Jetson TX2. (s.f.). https://www.nvidia.com/es-la/autonomous-machines/embedded-systems/jetson-tx2/ (Accessed: 2020-02-01)

KaleabTessera. (2019). *Prm-path-planning.* Descargado 2019, de *https : //github.com/KaleabTessera/PRM − Path − Planning*

Klancar, G., Zdesar, A., Blasic, S., y Skrjanc, I. (2017). *Wheeled mobile robotics from fundamentals towards autonoumous systems.* United Kingdom: Butterworth-Heinemann.

Klančar, G., Zdešar, A., Blažič, S., y Škrjanc, I. (Eds.). (2017). *Wheeled mobile robotics From fundamentals towards autonomous systems*. Butterworth-Heinemann.

Krishnan, A. B., y Kollipara, J. (2014). Intelligent indoor mobile robot navigation using stereo vision. *Signal and Image Processing An International Journal*, *5*, 1-9.

Kumano, M., Ohya, A., y Yuta, S. (2000). Obstacle avoidance of autonomous mobile robot using stereo vision sensor. *University of Tsukuba*, *1*, 1-6.

Li, Y., Dai, S., Shi, Y., Zhao, L., y Ding, M. (2019). Navigation simulation of a mecanum wheel mobile robot based on an improved a* algorithm in unity 3d. *Sensors 2019*, *19*, 1-44.

Lin, C.-J., Lin, H.-Y., y Yu, C.-Y. (2019). Using a Type-2 Neural Fuzzy Controller for Navigation Control of Evolutionary Robots. (Taiwan). *Sensor and materials*, *31*(9), 2735-2751. doi: https://doi.org/10.18494/SAM.2019.2343

Mamun, M. A. A., Tariq, M., y cols. (2018). Embedded system for motion control of an omnidirectional mobile robot. *IEEE Access*, *6*, 6722-6739. doi: https://ieeexplore.ieee.org/document/8262666

Martinez, A., y Fernández, E. (2013). *Learning ros for robotics programming*.

Masmoudi, M. S., Krichen, N., Masmoudi, M., y Derbel, N. (2016). Fuzzy logic controllers design for omnidirectional mobile robot navigation. *ELSEVIER*, *49*, 901-919. doi: https://doi.org/10.1016/j.asoc.2016.08.057

Mendel, J. M. (2001). *Uncertain rule-based fuzzy logic systems: introduction and new directions*. Prentice Hall PTR Upper Saddle River.

Muñoz-Ceballos, N. D., y Valencia-Velásquez, J. A. (2016). Benchmark framework for mobile robots navigation algorithms. *Revista Facultad de Ingeniería (Fac. Ing.)*, *23*, 65-73.

Olán, I. C. L. G. (2013). *Fusión de información de pose en robots móviles con visión*. Cuernavaca, Morelos.: Centro Nacional de Investigación y Desarrollo Tecnológico.

Pang, F., Luo, M., Xu, X., y Tan, Z. (2020, Noviembre). Path tracking control of an omni-directional service robot based on model predective control of adaptive neural-fuzzy inference system. , *11*, 18. doi: doi.org/10.3390/app11020838

Patruno, C., Colella, R., Nittia, M., Reno, V., Mosca, N., y Stella, E. (2020, Febrero). A vision-based odometer for localization of omnidirectional indoor robots. , *20*, 25. doi: 10.3390/s20030875

Ponce-Cruz, P., Molina, A., y MacCleery, B. (2016). *Fuzzy logic type 1 and type 2 based on labview^{TM} fpga*. Springer.

Rodríguez, M., y Huertas, Y. (2016). Metodología para el Diseño de Conjuntos Difusos Tipo-2 a partir de opiniones de Expertos.. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 5. doi: 10.1109/TSMC.2016.2834618

Tang, W., Liu, Y., Liu, C., Gu, J., Ji, Y., y Liu, X. (2017). Research on co-simulation of omni-directional mobile platform based on adams and matlab. *International Forum on Management, Education and Information Technology Application (IFMEITA 2016)*, 560-568.

Valverde Moreno, S. (2020, nov.). Implementación de sensores 3d para la navegación de robots móviles y vehículos autónomos. , *33*, Pág. 176–186. doi: 10.18845/tm.v33i7.5492

Velázquez, C. A. (2016). Implementación de sistemas de navegación autónomo en robot móvil experimental para reconstrucción y exploración. (Colombia). *IV Congreso internacional de ingeniería mecatrónica y automatización - CIIMA 2015*, 72-84.

Wu, D. (2018). A brief tutorial on interval type-2 fuzzy sets and systems. *School of Automation, Huazhong University of Science and Technology*, 1-13.

ZED stereo camera. (s.f.). `https://www.stereolabs.com/zed/`. (Accessed: 2020-02-01)

Anexos

EDUCACIÓN

CORE

Centro de Investigación en Computación del Instituto Politécnico Nacional

Awards the present

Certificate of Participation

to Francisco Javier Valdepeña Rivera

for his participation with the presentation **"Comparative analysis of interest point detectors algorithms on Robotic Operative System."** in the 19th International Congress on Computer Science Core 2019 held at the Instituto Politécnico Nacional in Mexico City, on 5 – 9 August 2019

Dr. Marco Antonio Ramírez Salinas
CIC General Director

Elizabeth López Lozada
Presidente del Comité Organizador

ISSN 1870-4069

Comparative Analysis of Interest Point Detectors Algorithms on Robotic Operative System

Francisco Javier Valdepeña Rivera[1], Dante Mújica Vargas[2], Miguel Ángel Ruíz[1]

[1] Universidad Politécnica del Estado de Morelos
Jiutepec, Morelos, Mexico
[2] CENIDET-TecNM, Depto. Ciencias computacionales
Cuernavaca, Morelos, Mexico
{vrfo150252, mruiz}@upemor.edu.mx, dantemv@cenidet.edu.mx

Abstract. Applications of robotic vision have had great advances within the artificial intelligence through the processing of images, as well as the automated systems (robots). A comparative analysis of some interest point detector algorithms will be performed, the next analysis will be about a robotic operating system called ROS, by means of a 2D object detector system. For this purpose, a physical architecture will be carried out to carry out the experimentation within a controlled work environment, in order to demonstrate which algorithm will work best in the future for the development of object recognition systems, implementing this system on Robots.

Keywords: detector, interest points, ROS, artificial vision.

1 Introduction

The artificial vision in these days has been realized through systems that detect characteristics of the objects and classify them by means of neural networks which decide what type of object is, by means of a comparison of weights or learning units which are updated every that an iteration within algorithm.

Deep learning is based on a set of algorithms for machine learning that attempts to model high-level abstractions in data using architectures composed of multiple non-linear transformations [1].

But there is also the way to segment artificial vision in various stages which are essential to divide the number of processes into more concise processes, such as those shown in Figure 1. This is to perform optimal recognition in robotic systems, contrary to only have neural networks for recognition what it is to load a neural network with points or characteristics that are a stage of artificial vision can achieve, that the recognition is more precise and with a minimum of variety to the conditions in which find the object.

In this way it is necessary to take into account which are those algorithms that can extract the most important characteristics to be able to make a good detection, so after having said information the recognition is achieved instantaneously with a neural network or applying the concept of deep learning.

It is proposed to analyze and compare object detection algorithms under rotation, translation and light conditions to conclude which of them is the best for the task and

Universidad Nacional Autónoma de México
Facultad de Estudios Superiores Cuautitlán

Otorgan la presente

Constancia

A:

Francisco Javier Valdepeña Rivera, Dante Mújica Vargas y Miguel Ángel Ruíz Jaimes

Por su participación en del

2º Congreso Estudiantil de Inteligencia Artificial Aplicada a la Ingeniería y Tecnología (CEIAAIT)

Realizado el 13 y 14 de noviembre en el Aula Magna de Ingeniería, Campo Cuatro.

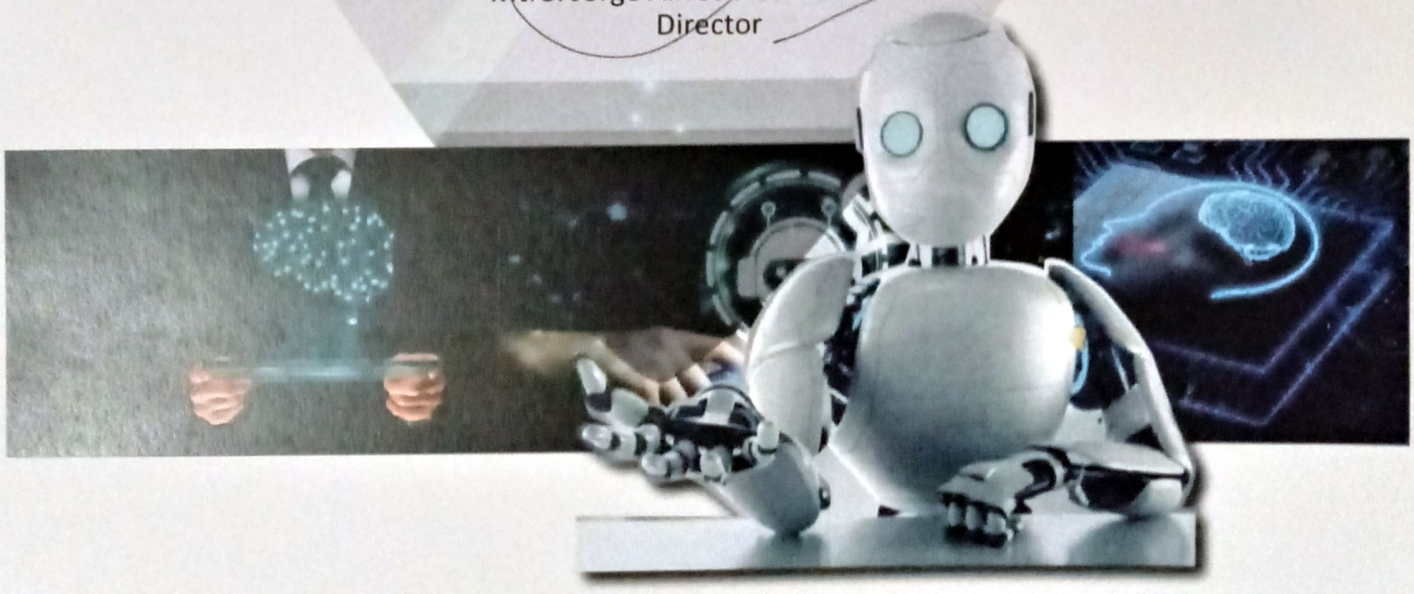
Con la ponencia titulada:

Análisis comparativo de trayectorias en robots

"Por mi raza hablará el espíritu"

Cuautitlán Izcalli, Estado de México, noviembre de 2019.

Mtro. Jorge Alfredo Cuéllar Ordaz
Director



Análisis comparativo de trayectorias en robots

1st Francisco Valdepeña
Tecnológico Nacional de México/CENIDET.
Cuernavaca, Morelos, México
francisco.valdepena19ca@cenidet.edu.mx

2nd Dante Mújica-Vargas
Tecnológico Nacional de México/CENIDET
Cuernavaca, Morelos, México
dantemv@cenidet.edu.mx

3rd Miguel Ángel Ruíz Jaimes
Universidad Politécnica del Estado de Morelos
Jiutepec, Morelos
mruiz@upemor.edu.mx

***Resumen*—En este artículo se analiza la planificación de trayectorias para robots, empleando los algoritmos: Control simple de trayectorias, Control por una trayectoria de referencia, Control basado en la optimización del enjambre de partículas y Control predictivo basado en modelos. Se realizó una evaluación cuantitativa basada en el Error Cuadrático Medio (ECM), Similitud coseno y el Coeficiente de Correlación de Pearson (CCP). Los sucesos experimentales indican que el Control predictivo basado en modelos, se desempeña mejor a trayectorias no lineales, con ECM $<$ 0.1, Similitud coseno $>$ 0.8 y CCP $<$ 0.02, en un rango continuo de [0-1].**

***Index Terms*—Análisis de trayectorias, Robots móviles, Algoritmos de control de trayectoria**

I. Introduction

En la robótica móvil, una trayectoria es un vector de poses en un robot móvil necesita para manejar sobre coordenadas generalizadas en el espacio.

$$\mathbf{q}_{ref}(t) = [x_{ref}(t), y_{ref}(t), \varphi_{ref}(t)]^T \tag{1}$$

La mayoria de veces la trayectoria siempre se define en un intervalo de tiempo finito $t \in [0, T]$, es decir una trayectoria tiene un punto de inicio y un punto fin.
El control para el seguimiento de una trayectoria es un mecanismo que asegurará que la trayectoria del robot $\mathbf{q}(t)$ está lo más cerca posible de la referencia $\mathbf{q}_{ref}(t)$ a pesar de dificultades encontradas.
Al realizar un control de seguimiento de trayectoria, lo más lógico es imaginar la trayectoria de referencia como un movimiento a seguir.
En este documento se realiza un análisis comparativo del control de trayectorias sobre los algoritmos: Control basado en la optimización de ejambre de partículas (PSO), Control predictivo basado en modelos (MPC), Control simple de trayectorías y Control para una trayectoria de referencia [1].
En la sección 2, se describen conceptos sintetizados de la teoría del modelado de sistemas móviles, junto con su formalización matemática y los algoritmos de control de trayectorias a evaluar. En la sección 3, se desglosa la experimentación, métricas a emplear y los resultados del análisis. En la sección 4, se emplea una optimización al algoritmo con mejor desempeño. En la sección 5, se da una conclusión del análisis.

II. Marco teórico

II-A. Modelado de robots móviles

La locomoción es el proceso de mover un sistema autónomo de un lugar a otro. Los modelos de movimiento pueden describir la cinemática del robot, esto es interesante ya que puede ser descrito matemáticamente, tal y como se muestra en la siguiente expresión [1].

$$\mathbf{q}(t) = \begin{bmatrix} x(t) \\ y(t) \\ \varphi(t) \end{bmatrix} \tag{2}$$

donde, $\mathbf{q}(t)$ es la trayectoria del robot, $x(t)$ y $y(t)$ son coordenadas locales en los ejes X e Y, $\varphi(t)$ es la orientación del robot.
En un modelado cinemático el vector pose de una trayectoria en un robot se describe geométricamente, por una matriz de rotación ($\mathcal{R}(\varphi)$), definida en la siguiente expresión [1].

$$\mathcal{R}(\varphi) = \begin{bmatrix} \cos\varphi & \sin\varphi & 0 \\ -\sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

donde, cada columna de la matriz representa una relación entre las coordenadas locales del robot ($[x(t), y(t), z(t)]$) y las coordenadas globales de referencia ($[x_{ref}(t), y_{ref}(t), z_{ref}(t)]$), en un intervalo de tiempo.
La pose de un robot de configuración diferencial, sobre un plano de coordenadas globales está representada en la Figura 1.

II-B. Control de robots móviles

El control de la pose de un robot esta basada a una pose de referencia, la cual no esta predefinida y el robot puede elegir cualquier ruta factible para llegar a la meta. Sin embargo, esta ruta puede ser definida explícitamente y adaptada durante el movimiento o ser implícita aplicando un algoritmo de control de trayectorias, para lograr las poses de referencia [2].
En una pose, la orientación indica la disparidad a una referencia, esto en la mayoría de algoritmos es manejado mediante un error de estimación, entre la orientación de referencia y la

Navegación de un robot diferencial basada en una Red Neuronal Multicapa con el método de retropropagación

Francisco Valdepeña * **Dante Mújica-Vargas** **

* *Tecnológico Nacional de México/CENIDET, Cuernavaca, Morelos, México, (e-mail: francisco.valdepena19ca@cenidet.edu.mx).*
** *(e-mail: dantemv@cenidet.edu.mx)*

Resumen: En este artículo se implementó una red neuronal multicapa para la navegación en robots de accionamiento diferencial, basada en el método de retropropagación, con el objetivo de evitar diversos obstáculos en un área simulada utilizando *CoppeliaSim Simulator*. En la simulación se definieron tres diferentes escenarios con obstáculos y para la detección de objetos se instalaron seis sensores de proximidad de tipo escaner láser, para calcular la distancia del obstáculo al robot.
En dicha implementación se evaluó un error cuadrático medio entre capas para entrenamiento se obtuvo un $ECM < 0.01$ entre capas durante el entrenamiento.

Palabras clave— Red Neuronal Multicapa, Algoritmo *Backpropagation*, navegación robótica, accionamiento diferencial.

1. INTRODUCCIÓN

El emplear técnicas de entrenamiento y aprendizaje como las redes neuronales para la solución de la navegación en robots moviles, ha sido muy empleada en diversas investigaciones. Este problema ha empleando diversos métodos, tales como clasificadores, algoritmos genéticos, algoritmos basados en teoría difusa, así como diferentes sensores entre ellos, sensores ultrasónicos, infrarrojos, cámaras, escaneres láser. Cada técnica ha resultado en un avance significativo para este problema. Para que un robot móvil sea capaz de desplazarse en un ambiente no controlado, de manera autónoma, sin que presente problemas obtenidos por la ambigüedad o incertidumbre de los escenarios y la cual suele ser sensada, donde se requiere un modelo previo del escenario y esto a través de redes neuronales, no es necesario ya que empleando solo información obtenida en plena ejecución es posible entrenar y enseñarle al robot a desplazarse en dichos escenarios.
Una red neuronal funciona apartir de datos de entrada optenidos de diferentes sensores, en este artículo se experimentó con sensores de proximidad, es decir, sensores de tipo escaner laser, los cuales mapean un ángulo limitado de información de la distancia de un objeto al robot móvil y tomando como base dicha distancia el robot es capaz de reaccionar a dicho objeto según sea la próxima dirección sin obstáculo.
La navegación se define como una metodología que permite al robot móvil poder desenvolverse en un entorno con obstáculos, en decir, es la capacidad de reaccionar frente a situaciones inesperadas, la principal cualidad es desenvolverse, de modo eficaz, en entornos no estructados que presenten un desorden de objetos y donde no sea sencillo navegar con libertad. La navegación se puede estructurar a manera de bloques como en la Figura 1 (V. et al. (2009)).

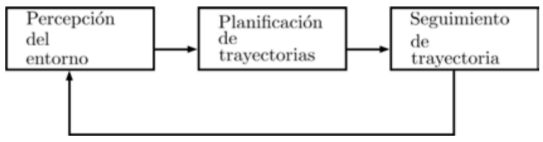


Figura 1. Metodología de la navegación.

Las tareas involucradas en la navegación de un robot móvil son: la percepción del entorno, de modo que le permita crear una abstracción del mundo, mediante diversos sensores, como por ejemplo los sensores infrarrojos para la proximidad entre objetos; la planificación de trayectorias, para alcanzar un destino, utilizando métodos de trazado de trayectorias y el seguimiento de la ruta.
En este documento se implementa una red neuronal artificial multicapa con el método *Backpropagation*, para realizar una navegación evitando obstáculos y siguiendo una trayectoria, sobre una plataforma de simulación utilizando un robot de accionamiento diferencial.
En la sección 2, se describen conceptos y formalizaciones matemáticas del modelo de una red neuronal multicapa y el método *Backpropagation*, describiéndolo en un algoritmo. En la sección 3, se desglosa la experimentación y resultados en los tres tipos de escenarios simulados. En la sección 4 se da una conclusión de la experimentación y

ENC 2021
9-11 agosto
Morelia, Michoacán

SMCC

Encuentro Nacional de Computación

CONSTANCIA

A:

Francisco Javier Valdepeña, Dante Mújica Vargas, Miguel Ángel Ruíz Jaimes y Antonio Luna Alvarez

Por su asistencia al *Encuentro Nacional de Computación*, así como la presentación del artículo titulado "*Navigation of an omnidirectional robot with Type-I Fuzzy Logic using Takagi-Sugeno model*" en el "**Taller de Inteligencia Artificial: Inteligencia Artificial: Métodos, Algoritmos y Aplicaciones**".

Morelia, Michoacán, México
9-11 de agosto 2021

Dra. Karina Mariela Figueroa Mora
Presidenta SMCC

Dra. María Lucia Barrón Estrada
Organizadora General

2021 Mexican International Conference on Computer Science (ENC) | 978-1-6654-2612-1/21/$31.00 ©2021 IEEE | DOI: 10.1109/ENC53357.2021.9534808

Navigation of an omnidirectional robot based on Takagi-Sugeno Logic Model Type-I

Francisco Valdepeña*
Departamento en Ciencias de Computación
Tecnológico Nacional de México/CENIDET
Cuernavaca, Morelos, México
francisco.valdepena19ca@cenidet.edu.mx
*Corresponding author

Dante Mújica Vargas
Departamento en Ciencias de Computación
Tecnológico Nacional de México/CENIDET
Cuernavaca, Morelos, México
dante.mv@cenidet.tecnm.mx

Antonio Luna-Alvarez
Departamento en Ciencias de Computación
Tecnológico Nacional de México/CENIDET
Cuernavaca, Morelos, México
jesus.luna18ce@cenidet.edu.mx

Miguel Ángel Ruíz-Jaimes
Depto. de Ingeniería en informática e Ingeniería en Electrónica y Telecomunicaciones
Universidad Politécnica del Estado de Morelos
Jiutepec, Morelos, México
mruiz@upemor.edu.mx

***Abstract*—In this article we present an algorithm for control and navigation of an omnidirectional mecanum wheels with based on a Takagi-Sugeno Fuzzy Model Type-I, aiming to avoid various obstacles in each environment defined. To achieve this, the robot receives the data from 3 proximity sensors and readjusts its direction and speeds to control its navigation. This article presents research work results obtained from the implementation and experimentation of the fuzzy algorithm. In the simulation, the duration was defined. The robot used was a KUKA youBot, to which 3 proximity sensors were installed to detect the various obstacles within the simulated scenario.**

***Index Terms*—Fuzzy algorithm, Takagi-Sugeno Fuzzy Model Type I, Simulation, Autonomous navigation, Omnidirectional robot, Proximity sensors.**

I. Introducción

Dentro de la navegación robótica se encuentra el accionamiento omnidireccional el cual representa una solución bien pensada para realizar tareas de las cuales el desplazamiento se realice lo más libre en cualquier dirección e instante de tiempo. Una plataforma particular de un robot omnidireccional es la que implementa ruedas tipo *mecanum*[1] o sueca, la cual consta de un rueda compuesta de rodillos pasivos que se montan en ángulos fijos para proporcionar una fuerza fuera del eje. El uso de cuatro ruedas permite al robot realizar movimientos omnidireccionales sin la necesidad de un sistema de dirección convencional, ver Figura 1.

Existen varios estudios en estado del arte donde demuestrán que la implementación de Lógica Difusa puede superar los desafíos que la rueda *mecanum* provoque. En [2], se implementó un Sistema Inteligente de Control de Lógica Difusa Distribuido (SICLDD) en un sistema de vehículos autónomos guiados con ruedas *mecanum* para lograr una mayor confiabilidad y reducir la complejidad de la síntesis del controlador. En [3], se propuso un control de modo deslizante de terminal no singular utilizando redes de ondas difusas para el seguimiento de trayectorias y estabilización de un vehículo omnidireccional con ruedas *mecanum*.

En este trabajo se implementa un sistema difuso de tipo Takagi-Sugeno para el control y navegación de un robot omnidireccional llamado KUKA youBot, este modelo es alimentado por 3 sensores de proximidad que se enfocan en la visión frontal y laterales del robot donde respuesta de los sensores afectan el control de la velocidad en los motores, al interactuar con un obstáculo, para ello se generaron 22 reglas que se dictaminan la nueva pose en trayectoria.

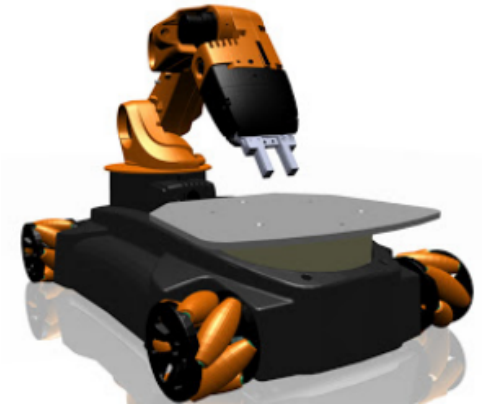


Fig. 1. Forma de un robot omnidireccional, con ruedas tipo *mecanum* [4].

En la sección 2, se describen conceptos y formalizaciones

Universidad Tecmilenio

La Universidad Tecmilenio, Campus Cuernavaca
otorga el presente **Reconocimiento a**:

Mtro. Francisco Javier Valdepeña Rivera

Torneo Nacional de Robótica Tecmilenio

Por su valiosa participación como Juez en el Torneo Nacional de Robótica de la Preparatoria Tecmilenio en el periodo Enero-Mayo 2021

Miguel Ángel Cardona Ahumada
Director General

Fiorella Ramírez Aguilar
Directora de Preparatoria

María Elena Hernández Tapia
Líder Docente

Temixco, Mor., a 18 de noviembre de 2021.