Instituto Tecnológico de Estudios Superiores de Monterrey

Implementación de robótica inteligente & Manchester Robotics

Manchester Robotics: Challenge 3

Profesores:

Rigoberto Cerino Jiménez

Dr. Mario Martinez

Integrantes

Daniel Castillo López A01737357

Emmanuel Lechuga Arreola A01736241

Paola Rojas Domínguez A01737136

24 de Abril de 2025

Índice

Resumen

El resumen presenta una síntesis de los aspectos más relevantes del proceso de investigación y desarrollo llevado a cabo, destacando los objetivos, la metodología empleada y los principales resultados obtenidos.

El Presente documento presenta el reto el cual se enfoca en el diseño, implementación y validación de un sistema de navegación autónoma de punto a punto para un robot móvil diferencial utilizando ROS 2. El reto consiste en aplicar control del tipo de lazo cerrado con un controlador proporcional (P), integrando técnicas de odometría y generación de trayectorias. se implementaron nodos funcionales que permiten la localización del robot mediante datos de encoders, además del cálculo del error proporcional respecto a una meta y la aplicación del control para seguir una trayectoria determinada como lo es un cuadrado de 2m *2m o alguno en cuestión con un traslado. La solución fue validada dentro de un entorno controlable, asegurando un comportamiento estable y robusto ante diversas condiciones.

Objetivos

En esta sección se presentan el objetivo general y los objetivos particulares del reto, los cuales están enfocados en la implementación de un control de lazo abierto para la trayectoria de un Puzzlebot.

Objetivo general: Desarrollar e implementar un sistema de navegación autónoma para un robot móvil del tipo diferencial utilizando programación en ROS 2, que integre un control de lazo cerrado, odometría y generación de trayectorias, validando su funcionamiento en un entorno simulado con el puzzlebot.

Objetivos específicos:

- Implementar un nodo de odometría basado en el método de dead reckoning a partir de los datos de los encoders de las ruedas.
- Diseñar e implementar un controlador proporcional (p) que permita al robot corregir su posición y orientación respecto a una meta deseada.
- Desarrollar un generador de trayectorias que gestione varios objetivos de navegación, permitiendo configurar el control específicamente las ganancias para cada objetivo de navegación.
- Validar el funcionamiento y precisión del sistema de control en una trayectoria cuadrada predeterminada y en objetivos específicos.
- Evaluar la estabilidad y precisión del sistema frente a errores de odometría, no linealidades y otras perturbaciones propias de un robot móvil diferencial.

Introducción

La introducción presenta el tema de investigación, proporcionando el contexto necesario para comprender la relevancia del reto a realizar. Se ofrece una visión general que facilita la construcción de un esquema mental sobre las metodologías utilizadas, además de los conceptos y tecnologías que se abordarán en el desarrollo del reporte.

Durante el desarrollo de este reto se trabajó con un robot móvil diferencial, específicamente el Puzzlebot, con el objetivo de implementar un sistema que le permitiera moverse de forma autónoma hacia diferentes posiciones dentro de un entorno simulado. Para lograr esto, fue necesario aplicar varios conceptos vistos en clase, como la odometría, el control en lazo cerrado y la navegación punto a punto.

Control en lazo cerrado para un robot móvil

Uno de los principales enfoques fue el uso de control en lazo cerrado, en el cual el robot utiliza su propia estimación de posición para corregir continuamente su movimiento y acercarse a el punto final de su trayectoria . Para estimar esa posición, se utilizó la información proporcionada por los encoders de las ruedas, implementando el método conocido como dead reckoning, el cual permite calcular el desplazamiento del robot integrando su velocidad lineal y angular a lo largo del tiempo. Aunque este método puede acumular errores es una buena base para comenzar a controlar el movimiento del robot ante un error.

El sistema de control que se implementó fue de tipo proporcional (P), el cual calcula el error entre la posición actual del robot y su objetivo, y en base a ese error ajusta las velocidades lineal y angular. Es un tipo de controlador efectivo, especialmente cuando se busca un comportamiento sencillo y estable.

Para darle mayor flexibilidad al sistema, se creó un nodo adicional que permite generar trayectorias. Este nodo permite al usuario ingresar varios puntos objetivo (por consola) y se encarga de enviarlos al robot uno por uno, esperando la confirmación de cada objetivo antes de continuar con el siguiente. Además, para cada punto se pueden definir diferentes ganancias del controlador, lo que permite ajustar la respuesta del sistema dependiendo del caso.

Todo el desarrollo se realizó utilizando ROS 2 y la validación heurística, siguiendo una estructura modular para facilitar la implementación y pruebas. En este reporte se explican los pasos que seguimos para llegar a una solución funcional, mostrando tanto el análisis teórico como la parte práctica del código implementado, junto con los resultados obtenidos durante las pruebas en formato de video o imagen.

Odometría de un robot móvil diferencial

En los robots móviles diferenciales, la odometría es un método fundamental para estimar la posición y orientación del robot a lo largo del tiempo, utilizando la información proporcionada por los encoders de las ruedas. Este proceso se basa en un modelo cinemático diferencial que describe cómo las velocidades individuales de las ruedas determinan el desplazamiento del robot en un plano (ManchesterRoboticsLtd, s. f.-h).

$$V = r\frac{\omega_R + \omega_L}{2} \tag{1}$$

$$\omega = r\frac{\omega_R - \omega_L}{l} \tag{2}$$

Donde:

- ω_R es la velocidad angular de la rueda derecha
- ω_L es la velocidad angular de la rueda izquierda
- V corresponde a la velocidad lineal del robot
- ω corresponde a la velocidad angular del robot
- l es la distancia entre las ruedas
- r es el radio de las ruedas

Estas velocidades se integran con el paso del tiempo para estimar la posición del robot $(x,\ y)$ y su orientación θ. Para ello, se utiliza una forma discretizada del modelo cinemático implementada con el método de Euler (ManchesterRoboticsLtd, s. f.-h).

$$x_{k+1} = x_k + V \cdot cos(\theta_k) \cdot dt \tag{3}$$

$$y_{k+1} = y_k + V \cdot cos(\theta_k) \cdot dt \tag{4}$$

$$\theta_{k+1} = \theta_k + \omega \cdot dt \tag{5}$$

Este procedimiento, conocido como dead reckoning, es útil en entornos donde el robot no tiene acceso a sensores externos como GPS. Sin embargo, hay que considerar que su precisión se ve afectada por errores acumulativos provocados por el deslizamiento de las ruedas o la discretización del tiempo. Por ello, la odometría suele complementarse con técnicas de localización más robustas.

Cálculo del error en un robot móvil diferencial.

El cálculo del error es esencial para el diseño de sistemas de control destinados a que guíen al robot hacia una posición deseada. Este proceso se conoce como estabilización de punto, y consiste en definir una pose objetivo $X_g = (x_g, y_g, \theta_g)$ que el robot debe alcanzar, comparándola con su pose actual estimada $X_r = (x_r, y_r, \theta_r)$ (ManchesterRoboticsLtd, s. f.-h).

A partir de esta comparación, se obtiene un vector error en coordenadas cartesianas y orientación:

$$e_x = x_g - x_r \tag{6}$$

$$e_y = y_g - y_r \tag{7}$$

$$e_\theta = atan2(e_y, e_x) - \theta_r \tag{8}$$

Este último término descrito en la ecuación (8), representa el ángulo entre la orientación actual del robot y la línea que conecta su posición con el objetivo. Dado que los ángulos pueden crecer indefinidamente, es común utilizar una función de envoltura para limitar los valores de e_θ dentro del rango $[-\pi, \pi]$, lo que garantiza un control estable (ManchesterRoboticsLtd, s. f.-h).

El error de distancia es una magnitud escalar que representa que tan lejos está el robot de su destino:

$$e_d = \sqrt{e_x^2 + e_y^2} \tag{9}$$

Con estos errores definidos, se puede aplicar una ley de control proporcional simple:

$$V = K_d \cdot e_d \tag{10}$$

$$\omega = K_\theta \cdot e_\theta \tag{11}$$

Donde K_d y K_θ son constantes que ajustan la respuesta del sistema. Esta forma de control es suficiente para lograr un comportamiento estable en la mayoría de los casos. El cálculo de errores, es conjunto con la odometría, permite al robot moverse de forma autónoma hacia una meta definida en un entorno conocido.

Solución del problema

En esta sección se desarrolla una solución basada en la interacción con el Puzzlebot, con el objetivo de comprender las coordenadas de la posición del robot y establecer una coordenada deseada, empleando conceptos como la odometría y la navegación punto a punto. Con esto, se plantea una estrategia que consiste en desarrollar dos nodos: uno encargado de calcular la localización del robot y otro que determine el valor del error en la distancia y el error en theta.

Se presentan los nodos obtenidos además de los elementos que tiene cada uno:

Nodo de odometría:

- Estima la posición (x,y) y orientación theta del robot usando la velocidad angular de ambas ruedas.
- Pública la odometría en el tópico /odom y convierte los resultados a una representación con cuaterniones para la orientación.

Controlador proporcional cuadrado (controller.py):

- Dirige al robot por una trayectoria cuadrada de 2 metros de lado.
- Implementa un control de avance y giro separados, priorizando la corrección angular antes del desplazamiento lineal.

Controlador general adaptable (controller 2.py):

- Dirige el robot por una trayectoria cuadrada de 2 metros de lado.
- Implementa un control de avance y giro separados, priorizando la corrección angular antes del desplazamiento lineal.

Controlador de trayectorias (path generator.py):

- Permite al usuario ingresar múltiples puntos desde consola.
- Publica cada punto en el tópico /goals, espera confirmación del controlador antes de enviar el siguiente.
- Permite modificar las ganancias Kp_lineal y Kp-angular para cada objetivo.

Validación:

- El sistema fue probado por método Heurístico, logrando que el robot siga trayectorias cuadradas y también puntos personalizados definidos por el usuario.
- La odometría y control permitieron mantener el robot en trayectorias estables alcanzando cada objetivo con una buena precisión.

Resultados

En este apartado se insertan las evidencias del funcionamiento del reto, agregando descripciones de lo que representa cada una de estas, así como un análisis de tales resultados, para saber si se consideran resultados satisfactorios o completos.

Se realizaron pruebas en el robot para validar el sistema de control. El robot completó con éxito la trayectoria cuadrada de 2 metros por lado, siguiendo siguiendo puntos predefinidos. Se observaron trayectorias estables y sin desviaciones significativas, siempre que las velocidades se mantuvieran dentro de los rangos seguros.

En trayectorias personalizadas, el generador interpretó correctamente los puntos (x, y) ingresados, generando mensajes PoseK (mensajes personalizados) adecuados. El controlador ejecuta los movimientos necesarios para llegar a los puntos con un error mínimo.

Conclusiones

Por último, se presentan los logros alcanzados a lo largo del desarrollo del reto, analizando el grado de cumplimiento de los objetivos planteados. Se evalúa si estos fueron alcanzados en su totalidad, identificando las razones detrás de su éxito o, en caso contrario, los factores que pudieron haber limitado su cumplimiento. Asimismo, se proponen posibles mejoras a la metodología utilizada, con el propósito de optimizar los resultados obtenidos.

El desarrollo de este reto permitió implementar con éxito un sistema de navegación autónoma para un robot móvil diferencial, cumpliendo en su totalidad los objetivos planteados. A través de la integración de técnicas de odometría, control de lazo cerrado y generación de trayectorias, logramos que el robot se desplazara de manera estable y precisa tanto en trayectorias predefinidas como en rutas personalizadas ingresadas por el usuario.

La implementación del nodo de odometría, basado en el método de dead reckoning, resultó fundamental para estimar correctamente la posición y orientación del robot, permitiendo al controlador proporcional aplicar las correcciones necesarias para alcanzar cada objetivo. Además, la estructura modular desarrollada proporcionó flexibilidad y robustez al sistema.

Las pruebas realizadas, tanto en la trayectoria cuadrada de 2 metros por lado como en los puntos definidos libremente por el usuario, demostraron un comportamiento estable, sin desviaciones significativas y con un error final mínimo, validando así la efectividad del diseño y la correcta sintonización de los parámetros de control.

Entre los principales aprendizajes destaca la importancia de considerar las limitaciones inherentes a la odometría, como el error acumulativo, así como la necesidad de ajustar los umbrales de error y las ganancias de control para garantizar un movimiento preciso. Asimismo, el uso de ROS 2 facilitó la estructuración ordenada del sistema en nodos independientes, promoviendo buenas prácticas de diseño en robótica distribuida.

Como propuestas de mejora, se podrían integrar el uso de controladores de tipo PI o PID para optimizar aún más la respuesta dinámica del sistema. Sin embargo, los resultados actuales reflejan un alto nivel de cumplimiento y demuestran la efectividad de la solución implementada para los desafíos planteados en el reto.

Referencias

En este apartado se anexan los elementos consultados para el desarrollo del tema de investigación.

ManchesterRoboticsLtd. (s. f.-g). *TE3002B_Intelligent_Robotics_Implementation_2025/Week3/Presentations/PDF/MCR2_Closed_Loop_Control_v3.pdf at main · ManchesterRoboticsLtd/TE3002B_Intelligent_Robotics_Implementation_2025*. GitHub. https://github.com/ManchesterRoboticsLtd/TE3002B_Intelligent_Robotics_Implementation_2025/blob/main/Week3/Presentations/PDF/MCR2_Closed_Loop_Control_v3.pdf

Freddy. (s. f.). *Implementacion-de-robotica-inteligente-2025/Implementación de Robótica Inteligente (sesion 5).pptx at main · freddy-7/Implementacion-de-robotica-inteligente-2025*. GitHub. https://github.com/freddy-7/Implementacion-de-robotica-inteligente-2025/blob/main/Implementaci%C3%B3n%20de%20Rob%C3%B3tica%20Inteligente%20(sesion%205).pptx

Freddy. (s. f.-b). *Implementacion-de-robotica-inteligente-2025/Implementación de Robótica Inteligente (sesion 8).pptx at main · freddy-7/Implementacion-de-robotica-inteligente-2025*. GitHub. https://github.com/freddy-7/Implementacion-de-robotica-inteligente-2025/blob/main/Implementaci%C3%B3n%20de%20Rob%C3%B3tica%20Inteligente%20(sesion%208).pptx