Reto semanal 3. Manchester Robotics

Oscar Ortiz Torres A01769292

Yonathan Romero Amador A01737244

Ana Itzel Hernández García A01737526

Implementación de Robótica Inteligente

Grupo 501

Jueves 24 de Abril del 2025

Resumen

Este proyecto tuvo como objetivo diseñar e implementar un sistema de navegación y control para el robot móvil Puzzlebot, capaz de seguir trayectorias definidas por el usuario mediante un sistema de control PID en lazo cerrado. Se desarrollaron tres nodos principales:

- `pathGenerator_ROSario`
- `controller_ROSario`
- `localization_ROSario`

Los resultados muestran que el robot fue capaz de seguir con precisión la trayectoria deseada, logrando un buen control del error lineal. Aunque hubo algunas limitaciones en la precisión angular, el sistema cumplió con los objetivos propuestos, demostrando su viabilidad y sentando las bases para futuras mejoras.

Objetivos

El objetivo de este challenge es desarrollar e implementar un sistema de navegación y control para el Puzzlebot, para el seguimiento de trayectorias definidas por el usuario, esto implementando un sistema de control de lazo cerrado tipo PID, siendo robusto ante perturbaciones.

- Implementar un nodo generador de trayectorias que permite al usuario definir caminos personalizados mediante parámetros de puntos, área y velocidades.
- Validar la factibilidad de los puntos considerando el comportamiento dinámico del robot y los parámetros ingresados por el usuario.
- Integrar un mensaje personalizado en el tópico `/pose` que incluye la información adicional sobre las velocidades.

Introducción

El control en lazo cerrado en un robot móvil se crea en base a las velocidades angulares de las ruedas del robot, el cual determina la velocidad lineal y angular del robot. El control en lazo cerrado es la medición de las velocidades para poder obtener la posición del robot, así comparándolas con la posición deseada para calcular el error que se le alimenta al controlador para poder eliminarlo.(Bañó, n.d)

Un controlador PID aplicado a un robot móvil es el encargado de determinar la velocidad angular de las ruedas para poder llegar al punto deseado, así disminuyendo las imperfecciones que genere el robot o el terreno. Dando robustez a la trayectoria asegurándose que el robot siga la trayectoria deseada.(Cortés et al., n.d)

Profundizando en el cálculo del error, este se calcula como la diferencia entre nuestra referencia y nuestra medición real. Es a esto a lo que se le aplica el PID, a este error para variar la intensidad a la cual el robot recibe los errores, así aumentando o disminuyendo su velocidad.

Solución del problema

El nodo `pathGenerator_ROSario` se encarga de la verificación de los parámetros dados por el usuario para la generación de la trayectoria, por lo que verifica dada punto calculando que estén dentro del área dada, además el código se encarga de verificar que estas velocidades están dentro de los valores permitidos para evitar picos de voltaje.

Diseño e implementación del nodo

- Lectura de parameter file: el nodo lee un `.yaml` file el cual contiene el vector de valores (X, Y) y el área de trabajo del robot. También obtiene la velocidades mínimas y máximas lineales y angulares.

- Mensaje Personalizado: se subscribe al tópico `pose` para publicar: Los puntos destinos ya verificados, lasa velocidades maximas y minimas lineales y angulares
- Retroalimentación: El nodo pública en `consola` todo los eventos sucedidos dentro del nodo, dando los errores ocurridos durante la ejecución y los valores aceptados.

El nodo encargado de la implementación de un controlador de lazo cerrado es el `controller_ROSario`, tiene la tarea de hacer que el robot siga una trayectoria formada por puntos objetivos, primero implementando una rotación y luego una traslación hacia cada punto con uso del controlador PID para ambos movimientos.

Diseño e implementación del nodo

- Puntos objetivos: se subscribe al tópico `/pose` que publica: Un punto destino (`msg.path.position.x, msg.path.position.y`), velocidad máxima y mínima de la parte lineal y la velocidad máxima y mínima de la parte angular. Cada punto se guarda en la variable `path_queue`.
- Lectura de la odometría: Se suscribe al tópico `/odom` para obtener la posición actual (X,Y) y la orientación del robot. Con el uso de transforms3d se convierte la orientación (θ) en cuaternión y el ángulo a yaw (θ).
- El ciclo de control: se ejecuta control_loop() como una maquina de estados con 3 modos, para el estado 0 se ejecuta la rotación, donde se calcula el ángulo del robot y el punto objetivo con el uso de PID, cuando

el error angular es menor a 10° pasa al siguiente estado. El estado 1 es de traslación donde se calcula la distancia al objetivo y cuánto avanza con el uso del PID, cuando el error de distancia es menor a 10 cm pasa al siguiente estado. Por último, está el estado 2 de reposo, donde se detiene el robot y espera a el siguiente punto.

- Controlador PID: se utilizan dos, un PID lineal y un PID angular que incluyen anti-windup para la saturación de la parte integral del controlador.
- Publicación de velocidades: Por último, el nodo publica la velocidad en formato Twist en el tópico `/cmd_vel`, donde `angular.z` para girar y `linear.z` para avanzar

El nodo `localization_ROSario`, es el encargado de implementar la odometría diferencial para la estimación en tiempo real de la posición (X,Y) y la orientación (θ) del puzzlebot a partir de las velocidades angulares de ambas ruedas.

Diseño e implementación del nodo

- Suscripción a los encoders: se suscribe a los tópicos `VelocityEncR` y `VelocityEncL`, los cuales contienen las velocidades angulares de la rueda derecha e izquierda.
- Cálculo de las velocidades: se convierten las velocidades de cada rueda en velocidades tangenciales usando el radio de las ruedas y con el uso del modelo cinemático del robot se obtienen las velocidades lineal y angular.

- Estimación de la posición y orientación: se utiliza la velocidad angular y lineal, se actualizan las coordenadas (x,y) y el ángulo θ en cada iteración integrando el delta de tiempo.
- Publicación de odometría: los datos estimados de posición y orientación se publica en el tópico `/odom` como mensaje `nav_msgs/Odometry`, de igual manera se incluye la velocidad lineal y angular para el uso de los otros nodos.

Resultados

De acuerdo a lo planteado anteriormente, ejecutamos los nodos obteniendo resultados favorables, se logró exitosamente el recorrido de la trayectoria deseada, el movimiento del robot siguió los puntos [(0.86602, 0.5), (-0.07366, 0.84202), (1.15734, 1.05908), (1.88234, -0.19665)], se confirmó el correcto funcionamiento de la parte de control.

Como evidencia de los resultados obtenidos, en la Figura 1 se encuentra la gráfica que representa la trayectoria seguida por el robot para su comparación con la trayectoria deseada, así como un video demostrativo del comportamiento del sistema en tiempo real en la parte de Anexos. Con dichos elementos podemos visualizar que el desplazamiento del robot fue consistente con lo esperado.

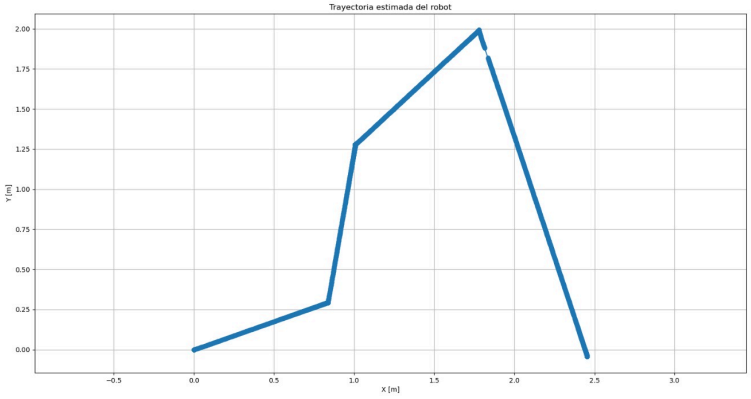

Figura 1. Gráfica del desplazamiento del robot con las coordenadas deseadas.

Conclusiones

Como se puede observar en los resultados obtenidos, el manejo del error lineal fue satisfactorio, logrando un error de distancia mínimo. Esto demuestra un buen desempeño en el control del desplazamiento del sistema. Sin embargo, se identifican oportunidades de mejora en el manejo del ángulo, ya que no se alcanzó el nivel de precisión esperado en la orientación, por lo que las trayectorias no fueron del todo satisfactorias.

A pesar de estos desafíos, se puede concluir que el reto fue abordado de manera exitosa, cumpliendo con los objetivos planteados inicialmente y sentando las bases para futuras optimizaciones del sistema.

Bibliografía o referencias

Cortés, U., Castañeda, A., Benítez, A., & Díaz, A. (n.d.). *Control de Movimiento de un Robot Móvil Tipo Diferencial Robot ůβot-32b*.

https://amca.mx/memorias/amca2015/files/0058_JuAT3-01.pdf

Bañó, A. (n.d.). *Análisis y Diseño del Control de Posición de un Robot Móvil con Tracción Diferencial*.

https://repository.unad.edu.co/bitstream/handle/10596/5786/An%C3%A1lisis%20y%20Dise%C3%B1o%20del%20control%20de%20Posici%C3%B3n%20de%20un%20Robot%20M%C3%B3vil%20con%20tracci%C3%B3n%20diferencial.pdf

Anexos

Video Demostrativo: Video Reto Semana 3.mp4

Repositorio Github: https://github.com/oscarOT09/close_loop_control_ROSario