ACTIVIDAD SEMANA III. Manchester Robotics

Abraham Ortiz Castro
Dpto. de Ingenierías Tec de Monterrey
Tecnológico de Monterrey
Puebla, México
A01736196@tec.mx

Alan Iván Flores Juárez
Dpto. de Ingenierías Tec de Monterrey
Tecnológico de Monterrey
Puebla, México
a01736001@tec.mx

Jesús Alejandro Gómez Bautista
Dpto. de Ingenierías Tec de Monterrey
Tecnológico de Monterrey
Puebla, México
A01736171@tec.mx

Ulises Hernández Hernández
Dpto. de Ingenierías Tec de Monterrey
Tecnológico de Monterrey
Puebla, México
A01735823@tec.mx

I. Resumen

El siguiente reporte se centra en entender la odometría del robot diferencial para el desarrollo de un sistema de control robusto que permita, mediante el cálculo del error de la posición y orientación, posicionar correctamente nuestro PuzzleBot ajustando así su orientación y coordenadas para coincidir con lo deseado. Esto, orientado como una actividad previo al reto semanal, facilitará el desarrollo de lo propuesto por Manchester Robotics para alcanzar un control robusto y eficaz.

II. Objetivos.

- Identificar las coordenadas de la posición del robot en todo momento para un mejor control.
- Aprender a introducir posiciones deseadas para dirigir el movimiento del PuzzleBot.
- Calcular el error de la posición en tiempo real para ajustar la posición mediante sistemas de control.

III. Introducción.

A. Odometría de un robot móvil diferencial

La odometría en un robot móvil de tipo diferencial se basa en la estimación de la posición así como de la orientación del vehículo a partir de la información de la rotación registrada por sus ruedas a través del tiempo.

Un robot diferencial cuenta con dos ruedas de tracción con movimiento independiente, lo cual le permite al robot moverse en cualquier dirección sin cambiar su orientación. Su movimiento a grandes rasgos se basa en dos estados: cuando sus ruedas se mueven a la misma velocidad, avanzarán o retrocederán, mientras que la diferencia de velocidades entre estas hará que el robot gire.

La distancia recorrida por rueda puede estimarse al multiplicar el número de rotaciones de la rueda por su circunferencia, además, si se conoce la distancia entre ambas ruedas (equivalente a la longitud de la base del robot) la diferencia de las distancias recorridas puede ayudarnos a encontrar su orientación. Para entender mejor la odometría del robot diferencial, a continuación se mencionan algunas de las ecuaciones para la odometría de un robot diferencial.

- **Distancia total recorrida:**

$$\triangle d = \frac{d_d + d_i}{2} \tag{1}$$

  Se trata de la distancia promedio recorrida por las dos ruedas. Se basa en la suposición de que el robot se mueve a lo largo de la línea que biseca el ángulo formado por las direcciones de movimiento de las ruedas.
- **Cambio de orientación:**

$$\triangle \theta = \frac{d_d - d_i}{l} \tag{2}$$

  Se trata del cambio de orientación del robot. Se basa en la suposición de que el propio robot gira alrededor de un punto en el centro de la longitud entre ambas ruedas.
- **Cambio de coordenadas en X:**

$$\triangle x = \triangle d \cdot cos(\theta) \tag{3}$$

  Cambio en la posición en el eje de las X del robot, calculada al multiplicar la distancia total recorrida por el coseno de la orientación actual del robot.
- **Cambio de coordenadas en Y:**

$$\triangle y = \triangle d \cdot sin(\theta) \tag{4}$$

  Cambio en la posición en el eje de las Y del robot, calculada al multiplicar la distancia total recorrida por el seno de la orientación actual del robot.

Estas ecuaciones, aunque funcionales, no toman en cuenta algunos factores externos (como el deslizamiento de la rueda, el tipo de superficie y demás) que podrían arrojar estimaciones poco precisas. [1]

B. Cálculo del error en un robot móvil diferencial

Para generar un sistema robusto de navegación y control del robot, el cálculo del error en un robot móvil de tipo diferencial es fundamental, y se basa fundamentalmente en la diferencia

entre la posición actual del robot y la posición deseada u objetivo.

El error en un robot diferencial puede abordarse en dos componentes primordiales.

- **Error de posición:**

$$e_p = P_o - P_a \tag{5}$$

  Es la diferencia entre la posición deseada y su posición actual, básicamente, es la diferencia entre las coordenadas de la posición objetivo y las actuales del robot.

- **Error de orientación:**

$$e_\theta = \theta_o - \theta_a \tag{6}$$

  Es la diferencia entre la orientación deseada y su orientación actual, básicamente, es la diferencia entre el ángulo objetivo y actual del robot.

[2]

IV. Solución del problema

A. Código y lógica

Para dar solución a este proyecto se realizó la creación de tres nodos `odometry` el cual es el encargado de recibir los valores de la velocidad angular de cada una de las llantas y convertir dichos datos a desplazamiento en 'x', 'y' y theta, valores que posteriormente son enviados por un tópico del mismo nombre. El nodo `path_generator` el cuál es el nodo encargado de enviar las coordenadas correspondientes para cada una de las figuras, desde un triángulo hasta un hexágono, cuyas coordendadas se encuentran dentro de un círculo con diametro 1, estas coordendas son enviadas de igual forma por otro tópico con el mismo nombre. Por último el nodo `controller`, recibe los valores del tópico `odometria` y del `path_generator` tomando esta información y las ecuaciones 1-4 podemos calcular la posición actual, la posición objetivo, el ángulo actual y el ángulo objetivo. Para calcular el error como se muestra en las ecuacuiones 5 y 6. Segmentos de código implementado en este último nodo se muestran a continuación:

```
#Se hacen las suscripciones pertinentes
        self.subscription_odometria = self.create_subscription(
            Vector,
            'odometria',
            self.signal_callback1,
            rclpy.qos.qos_profile_sensor_data )

# Callback para recibir la posición
actual del robot
    def signal_callback1(self, msg):
        if msg is not None:
            self.Posx = msg.x
            self.Posy = msg.y
            self.Postheta = msg.theta
```

En el código anterior crea el suscriber que escucha al tópico 'odometria' con tipo de dato 'Vector', el cual contiene la posición actual del robot en los ejes 'x', 'y' y theta, para que posteriormente dentro de la función `signal_callback1` sean asignadas a otras variables locales.

```
#Se hacen las suscripciones pertinentes
        self.subscription_path = self.create_subscription(
            Path,
            'path_generator',
            self.signal_callback2,
            rclpy.qos.qos_profile_sensor_data )

def signal_callback2(self, msg):
    if msg is not None:
        self.trayectoria = [(0,0),(msg.x1, msg.y1), (msg.x2, msg.y2),
        (msg.x3, msg.y3), (msg.x4, msg.y4), (msg.x5, msg.y5),
        (msg.x6, msg.y6)]
```

Dentro de esta sección de código se crea el suscriber que escucha al tópico `path_generator` que contiene las distintas coordenadas de cada una de las figuras para ser guardadas dentro de un vector desde el cual las accesaremos posteriormente.

```
#Timer para realizar los calculos correspondientes
self.timer = self.create_timer(0.1, self.timer_callback)

def timer_callback(self):
    [...]
    # Obtener las coordenadas del punto actual en la trayectoria
    target_x, target_y = self.trayectoria[self.indice_punto_actual+1]

    target_x_ant, target_y_ant =
    self.trayectoria[self.indice_punto_actual]

    # Calcular las coordenadas polares del punto objetivo
    self.distancia = math.sqrt((target_x - self.Posx)**2 +
    (target_y - self.Posy)**2)
    self.angulo_objetivo =
    numpy.arctan2(target_y-target_y_ant, target_x-target_x_ant)

    self.errorTheta = self.angulo_objetivo - self.Postheta
```

Dentro de nuestra función `timer_callback` se toman los valores obtenidos de nuestros suscribers con anterioridad y se realizan las operaciones para el cálculo de las coordenadas actuales del robot, las coordenadas objetivo, el valor actual de theta y su valor objetivo. Para que con las restas de los mismos se obtenga el error de tanto la posición, así como del ángulo. Estos valores son posteriormente utilziados para determinar la velocidad lineal y angular que se envian al robot a través del tópico `cmd_vel` con tipo de mensaje `Twist`. De igual manera se envían los valores del error a través de un tópico con el mismo nombre.

B. ROS2

Nodos y tópicos

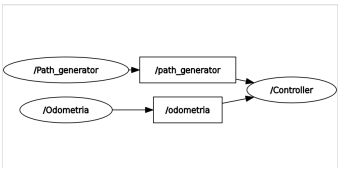

Fig. 1. rqt graph

Tópicos En las figuras 1 y 2 se pueden obsevar los nodos

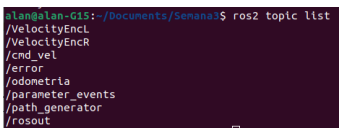

Fig. 2. Tópicos

previamente mencionados, así como los tópicos que se utilizan dentro de nuestro programa, entre los que se encuentran aquellos creados por nosotros y aquellos que ya se incluyen con el Puzzlebot.

V. Resultados

Aunque tanto el robot como las mediciones se mantienen dentro de un rango aceptable de error (no más allá del diez por ciento), hemos podido identificar el impacto que el tipo de superficie tiene en la locomoción del robot. Como se ha señalado en diversas secciones, especialmente en la parte dedicada al ”Cálculo del error”, factores como la fricción y el tipo de superficie desempeñan un papel crucial en la precisión del cálculo de la odometría.

Además, hemos observado que los acoplamientos (coples) parecen tener un cierto grado de holgura que afecta directamente al codificador (encoder). Esto se traduce en situaciones en las que, aunque el motor está en movimiento, la rueda no gira de manera sincronizada, lo que influye en el cálculo de la odometría.

A continuación se muestra la trayectoria objetivo del robot (figura 3), cuyas coordenadas corresponden a los puntos:

- (0,0.5)
- (0.75, 0.07)
- (0.75, 0.93)

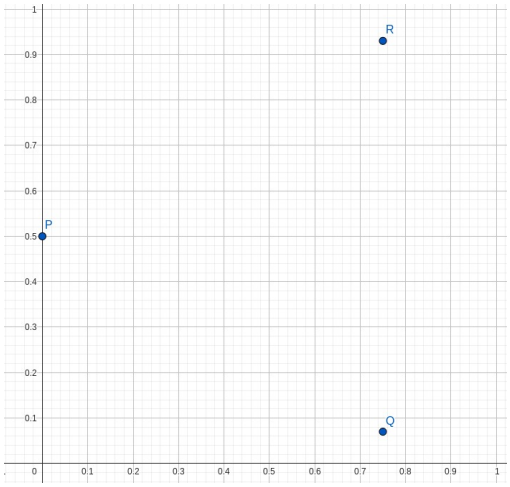

Fig. 3. Trayectoria objetivo

De igual manera se muestra el video del funcionamiento del robot siguiendo dicha trayectoria (figura 4):

VI. Conclusiones

En esta ocasión, nos centramos en mejorar la odometría de nuestro robot diferencial y calcular el error de su posición actual en comparación con la esperada, lo cual si bien ya habíamos implementado en el desafío anterior, el resultante de esta actividad fue mucho mejor estructurado. No obstante, siempre hay espacio para mejoras. Los espacios de mejora que vemos evidentes es la precisión y la consistencia de la odometría, pues si bien funcionan como se espera, ciertamente

Fig. 4. Video funcionamiento trayectoria

algunos factores externos y de cálculo afectan el desempeño del robot. En líneas generales, aunque se alcanzaron los objetivos principales de la actividad y el modelo actual es considerablemente más robusto que el anterior, es necesario seguir refinando nuestra metodología para lograr una mayor precisión y consistencia a la hora de definir la locomoción de nuestro PuzzleBot.

References

[1] Colombia, V., Montoya, J., Rios, A., MODELO CINEMÁTICO DE UN ROBOT MÓVIL TIPO DIFERENCIAL Y NAVEGACIÓN A PARTIR DE LA ESTIMACIÓN ODOMÉTRICA (2009). (pp. 191–196). link.

[2] Seguimiento de Trayectoria Mediante la Solución del Error Lateral en un Robot Tipo Diferencial. (s.f.). link.

[3] ManchesterRoboticsLtd. (s. f.). GitHub - ManchesterRoboticsLtd/TE3002B-2024: Intelligent Robotics Implementation. GitHub. link.