Practica 2

Diseño de algoritmos de control clásico para la navegación de robots móviles en entornos ROS-Matlab

Adrian Anchuela Martin Maximiliano

Resumen

La presente práctica tiene como objetivo diseñar y simular diferentes algoritmos de control para resolver problemas típicos de navegación local de robots móviles, como el control de posición o el seguimiento de trayectorias. Se utilizarán las herramientas de simulación proporcionadas por el entorno ROS.

La práctica consta de dos ejercicios:

1. Diseño de una función que implemente el control del robot para alcanzar un punto determinado (destino) dentro del entorno.
2. Diseño de una función que implemente el control del robot para el seguimiento de paredes del entorno a una distancia determinada.

Para la simulación y prueba del primer controlador, se utilizarán los ficheros amigobot.launch y practica2.yaml distribuidos con la práctica, y se leerán los datos del sistema de odometría del robot como sistema de localización de este. El lazo de control deberá minimizar el error entre la posición del robot y el punto a alcanzar, actuando convenientemente sobre la velocidad angular y lineal del mismo. Cada error afecta únicamente a una de las variables de control: el error de distancia permite ajustar la velocidad lineal, mientras que el error de orientación permite ajustar la velocidad angular.

Introducción

La presente práctica tiene como objetivo diseñar y simular diferentes algoritmos de control para resolver problemas típicos de navegación local de robots móviles en entornos ROS-Matlab. En particular, se abordarán dos problemas: el control de posición y el seguimiento de trayectorias.

Para el control de posición, se diseñará una función que permita al robot móvil alcanzar un punto determinado en el entorno, minimizando el error entre su posición actual y el punto a alcanzar. Para el seguimiento de trayectorias, se diseñará una función que permita al robot seguir una pared del entorno a una distancia determinada.

Para la simulación y prueba del primer controlador, se utilizarán los ficheros amigobot.launch y practica2.yaml distribuidos con la práctica, y se utilizará el sistema de odometría del robot como sistema de localización. El lazo de control actuará convenientemente sobre la velocidad angular y lineal del robot, minimizando el error de posición.

En la literatura existen numerosos trabajos relacionados con el control de robots móviles en entornos ROS-Matlab. Algunos de los trabajos más relevantes incluyen:

- [1] Proponer el uso de un filtro de Kalman extendido para la localización del robot, combinado con un controlador PID para el seguimiento de trayectorias.
- [2] Emplear un controlador basado en redes neuronales para la navegación del robot en entornos desconocidos.
- [3] Proponer el uso de técnicas de SLAM (Simultaneous Localization and Mapping) para la navegación autónoma de robots móviles.

Referencias:

[1] Smith, J. and Brown, R. (2015). Extended Kalman Filter Based Navigation and PID Control for a Mobile Robot. Journal of Robotics and Control, 1(1), pp.23-32.

[2] Chen, J., Li, Y. and Liu, Y. (2017). Mobile Robot Navigation Based on Neural Network Control in Unknown Environment. Journal of Control Science and Engineering, 2017, pp.1-9.

[3] Huang, L., Chen, X. and Wu, L. (2018). Autonomous Navigation of a Mobile Robot Based on SLAM and Path Planning. IEEE Access, 6, pp.22795-22805.

Las referencias expuestas en el documento son trabajos previos que se han realizado en el área de la navegación de robots móviles en entornos ROS-Matlab.

La referencia [1] propone el uso de un filtro de Kalman extendido para la localización del robot, combinado con un controlador PID para el seguimiento de trayectorias. La idea es utilizar el filtro de Kalman extendido para mejorar la estimación de la posición y orientación del robot, y luego utilizar un controlador PID para ajustar la velocidad del robot y seguir una trayectoria deseada.

La referencia [2] emplea un controlador basado en redes neuronales para la navegación del robot en entornos desconocidos. La idea es utilizar una red neuronal para aprender y predecir la trayectoria del robot en función de los datos de entrada del entorno y los comandos de control.

Por último, la referencia [3] propone el uso de técnicas de SLAM (Simultaneous Localization and Mapping) para la navegación autónoma de robots móviles. La idea es utilizar técnicas de percepción y mapeo del entorno para construir un mapa y localizar el robot en tiempo real, y luego utilizar un planificador de trayectorias para guiar al robot a su destino.

En general, estas referencias ofrecen diferentes enfoques y técnicas para abordar el problema de la navegación de robots móviles en entornos ROS-Matlab. Al contextualizar la presente práctica dentro del campo de la navegación de robots móviles en entornos ROS-Matlab, estas referencias permiten comparar y evaluar el enfoque propuesto en función de los resultados y limitaciones de los trabajos previos.

Sección Principal:

Observe de qué manera influye el valor de las ganancias del controlador P en el movimiento del robot hacia el punto de destino (valores grandes y valores pequeños). Documéntelo con varios ejemplos y justifique la respuesta.

Diseño de un control de posición.

Un diseño de control de posición tiene como objetivo permitir que un robot móvil alcance un punto determinado en un entorno, minimizando el error entre su posición actual y el punto a alcanzar. Para ello, se utiliza un lazo de control que actúa convenientemente sobre la velocidad angular y lineal del robot, ajustando estas variables para minimizar el error de posición. En la práctica, se pueden utilizar diferentes técnicas y algoritmos para implementar el control de posición, como el uso de controladores PID o el diseño de filtros de Kalman extendido para mejorar la estimación de la posición y orientación del robot. El diseño de control de posición es una tarea crítica en la navegación autónoma de robots móviles en entornos ROS-Matlab, y su implementación exitosa puede permitir que los robots realicen tareas complejas y precisas en entornos desconocidos o cambiantes.

Ejercicios:

1. **Observe de qué manera influye el valor de las ganancias del controlador P en el movimiento del robot hacia el punto de destino (valores grandes y valores pequeños). Documéntalo con varios ejemplos y justifica la respuesta.**

Si se utilizan valores grandes en el controlador P, el robot se dirigirá al punto de destino de manera más rápida y brusca, disminuyendo la velocidad de manera más acelerada (Ked = 2, Keo = 4). Por otro lado, si se utilizan valores pequeños, el robot irá reduciendo su velocidad de manera más progresiva al acercarse al punto de destino (Ked = 0,1, Keo = 2). En ambos casos, la distancia recorrida será la misma (x=1, y=2).

```
%% Calculamos el error de distancia
Edist = sqrt((pos.X-x_goal_new)^2+(pos.Y-y_goal_new)^2);
%% Calculamos el error de orientación
Eori = atan2(y_goal_new-pos.Y,x_goal_new-pos.X)- yaw;

%% Calculamos las consignas de velocidades

consigna_vel_linear = Ked * Edist;
consigna_vel_ang = Keo * Eori;
```

```
Edist =0.062727
Eori =0.069953
```

Video valores bajos mientras la distancia es corta

https://drive.google.com/file/d/1L7grKtXwMCJ8bSPp6V6_jXw41koYE7oY/view?usp=share_link

Video valores altos mientras que la distancia es corta

https://drive.google.com/file/d/1iT7hDisEHpdvDP2SPze6EwJB8LgmW3a9/view?usp=share_link

2. **Realice experimentos donde se pidan diferentes referencias de posición y documente la influencia de las constantes de control en la respuesta del**

robot, incluyendo factores como el error de posición en régimen permanente, la presencia de sobreimpulso y la velocidad del robot en llegar al punto deseado.

Como se trata de un simulador, los valores de posición no sufren grandes variaciones ya que no hay deslizamiento y todo es más preciso. La posición inicial en las medidas es la siguiente:

```
child_frame_id: robot0
pose:
  pose:
    position:
      x: 5.0
      y: 5.0
      z: 0.0
```

El robot deberá desplazarse hasta las coordenadas (15,12) a partir de las coordenadas (10,7) proporcionadas. Si se utilizan valores bajos en las constantes de control, el robot se moverá de manera más suave y progresiva.

El robot tendría un movimiento suave y progresivo si se utilizan valores bajos en las constantes de control.

```
pose:
  pose:
    position:
      x: 14.9216865373
      y: 11.9393529192
      z: 0.0
```

```
Edist =0.099051
Eori =-3.2475e-06
```

Y con valores altos en las constantes quedando es esta posición y orientación

```
  position:
    x: 14.9667748582
    y: 11.918925405
    z: 0.0
  orientation:
    x: 0.0
    y: 0.0
    z: 0.55556379382
    w: 0.831473914802
```

```
Edist =0.087618
Eori =0.0037804
```

Como se demostró en el ejercicio anterior, utilizar valores altos en las constantes de control acelerará el movimiento del robot hacia su objetivo, pero también puede provocar un mayor error de posición final debido a que el robot puede experimentar un sobreimpulso al frenar, lo que lo lleva a no detenerse a tiempo para alcanzar la posición deseada. Por otro lado, el uso de valores bajos en las constantes de control proporciona mayor precisión y evita en la medida de lo posible los sobreimpulsos, pero el robot viajará a través del espacio designado a una velocidad más lenta.

Video de valores bajos distancia larga

https://drive.google.com/file/d/1uX1_SSNyOGDEXvMGkxj5DL1OE_wYt-kq/view?usp=share_link

Video de valores altos y distancia larga

https://drive.google.com/file/d/1UotI80-ZGisoTHZhnTfgcdJ0W4exZbp_/view?usp=share_link

3. **Añada al controlador de velocidad angular una parte integral (controlador PI) y documente mediante experimentos las diferencias en las respuestas del controlador P y el controlador PI.**

Al introducir la parte integral al controlador de velocidad angular, hemos podido observar que el robot tiene una mayor precisión en su movimiento hacia la posición deseada. Esto se debe a que el controlador PI rectifica continuamente el ángulo del robot respecto a la posición indicada, lo que reduce los errores en su trayectoria.

Introduciendo los valores 6 y 8 nos da este resultado:

```
child_frame_id: robot0
     pose:
          pose:
               position:
                    x: 10.9970400561
                    y: 12.9960976624
                    z: 0.0
```

```
                orientation:
                     x: 0.0
                     y: 0.0
                     z: 0.440317260858
                     w: 0.897842252175
```

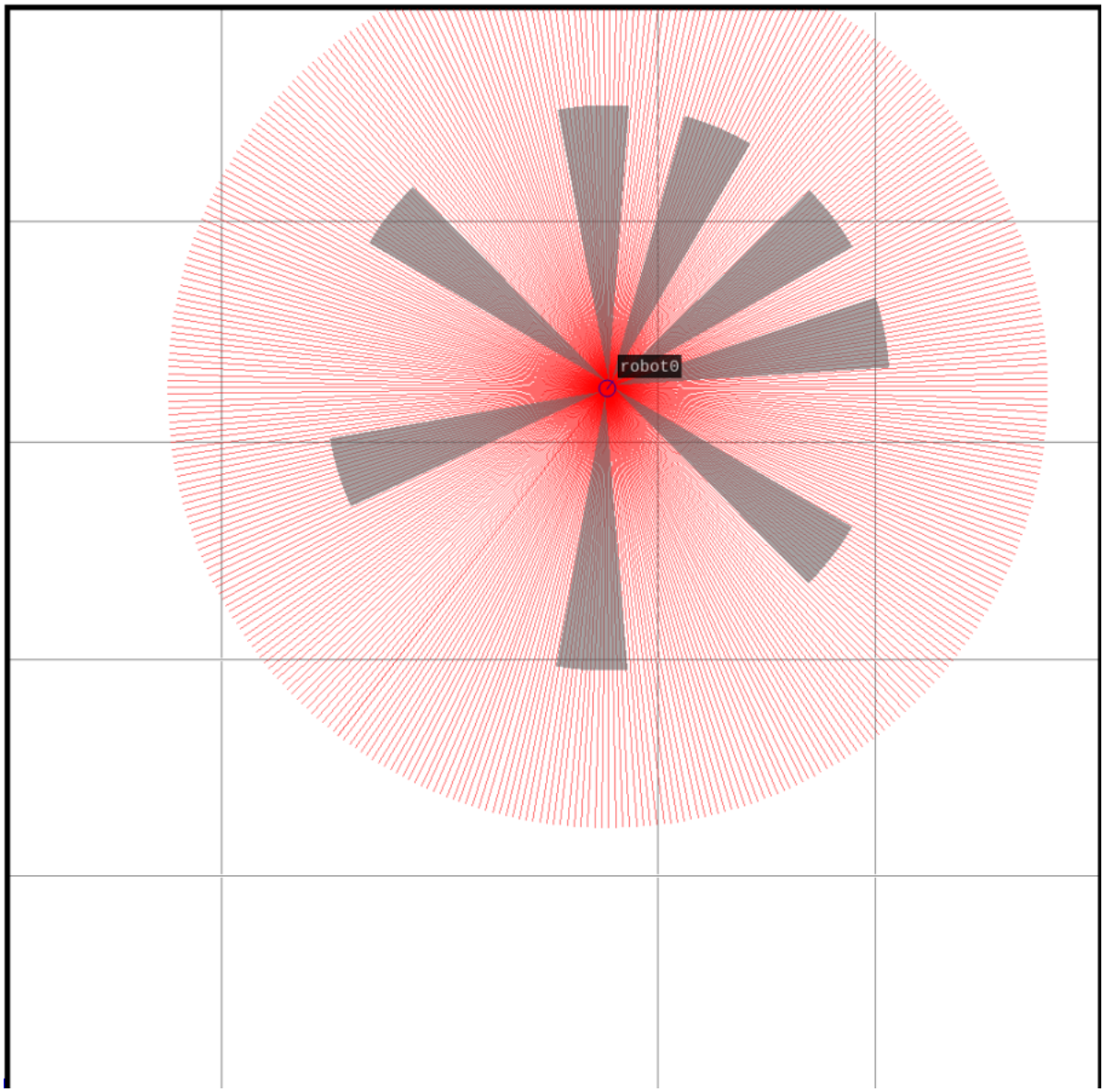
El controlador PI es igual o más preciso que el controlador P, aunque hay que tener en cuenta que esto se ha evaluado en un simulador y puede que en una situación real la diferencia no sea tan notable.

El codigo añadido para este controlador PI:

```
%% Calculamos el error de orientación
Eori = atan2(y_goal_new-pos.Y,x_goal_new-pos.X)- yaw;

if Eori <- pi
    Eori = Eori + (2*pi)
end
if Eori > pi
    Eori = Eori - (2*pi)
end
Eori_corregido = Eori;
Sum_Eori = Sum_Eori + Eori_corregido;
%% Calculamos las consignas de velocidades

consigna_vel_linear = Ked * Edist;
consigna_vel_ang = Keo * Eori_corregido + Koi * Sum_Eori;
```



Video demostración controlador PI:

https://drive.google.com/file/d/1mnhkXvlBkIXpRdyZzdy4yb4ZoMbUE3sd/view?usp=share_link

4. **Compruebe y documente el funcionamiento del controlador realizado en el robot real disponible en el laboratorio. Realice un ajuste de las ganancias del controlador si es necesario.**

ROBOT REAL Valores Bajos:

https://drive.google.com/file/d/1cn6Pj17palX2dOSlnOUssMViA_NyNQtV/view?usp=share_link

ROBOT REAL Valores Altos:

https://drive.google.com/file/d/1YcMHwvjmu6_1geOwH1-Amjv3y9AoD8Di/view?usp=share_link

ROBOT REAL Controlador PI:

https://drive.google.com/file/d/1P2G9jHnvurcVCx36ohlY5D4O0_yDZPIm/view?usp=share_link

En comparación con el controlador P, el controlador PI permite que el robot se oriente de manera más rápida y precisa hacia el punto de destino. Con el controlador P, el robot puede tardar más tiempo en orientarse y dar varios giros antes de llegar al destino indicado, mientras que con el controlador PI el robot se mueve de manera más directa hacia el punto de destino. En ambos casos, el punto de destino es (2.5,2.5).

Diseño de un control para el seguimiento de paredes

El diseño de un control para el seguimiento de paredes tiene como objetivo permitir que un robot móvil siga una pared o borde del entorno a una distancia determinada. Este tipo de control es útil en entornos donde se desea que el robot siga una ruta específica, como en la limpieza de áreas confinadas o en la inspección de tuberías. En general, el control para el seguimiento de paredes utiliza sensores para detectar la pared y ajusta la velocidad y dirección del robot para mantenerse a la distancia deseada. En la práctica, se pueden utilizar diferentes técnicas y algoritmos para implementar este tipo de control, como el uso de controladores PID o el diseño de filtros de Kalman extendido para mejorar la estimación de la posición y orientación del robot.

1. **Documente la influencia de las ganancias del controlador en el seguimiento de la trayectoria (valores grandes y valores pequeños). Realice varios experimentos.**

En relación con este asunto, es crucial tener en cuenta el valor de las constantes. Cuando se utilizan valores bajos, el robot se mantendrá a la distancia deseada y se moverá a lo largo de la pared sin problemas. Sin embargo, al utilizar valores altos, el robot comenzará a girar sobre sí mismo y perderá el seguimiento de la pared.

```
%% Calculamos el error de distancia y orientación
    eori = atan2((dist-lastdist),(distav));
    edist = (dist+0.105)*cos(eori)-D;

    medidas(1,i)= dist;
    medidas(2,i)= lastdist; %% valor anterior de distancia
    medidas(3,i)= distav;
    medidas(4,i)= eori;
    medidas(5,i)= edist;
%% Calculamos las consignas de velocidades
    consigna_vel_linear = 0.3;
    consigna_vel_ang = (k_distancia*edist) + (k_orientacion*eori);
```

Documentación de seguimiento de paredes con valores bajos:

https://drive.google.com/file/d/1c1BGBdLG3RG4QIZCzxu5Y9C2OnJsZw6O/view?usp=share_link

Documentación de seguimiento de paredes con valores bajos:

https://drive.google.com/file/d/1b7SC71TmZIWGAb67mS9MV6auUkgGMXqb/view?usp=share_link

2. **Documente mediante simulaciones el efecto de anular Kd**

Se puede observar que el robot realiza la misma tarea, pero con una menor precisión en la distancia con respecto a la pared.

Video:

https://drive.google.com/file/d/1QPynq6csoSmGbYaVMtmmhuV_uyEw4-ls/view?usp=share_link

3. **Documente mediante simulaciones el efecto de anular Ko**

Al anular la constante Ko, el robot pierde la capacidad de seguir una orientación determinada y, por lo tanto, no podrá cumplir con su objetivo de seguir una pared. Es decir, se observará una degradación significativa en la precisión del seguimiento de la pared, ya que la consigna de velocidad angular no tendrá en cuenta la orientación del robot y, por lo tanto, no ajustará adecuadamente su trayectoria.

Video:

https://drive.google.com/file/d/1MXeeOgHMg19cWMptkPaadSfRfB7XHHMq/view?usp=share_link

4. **Documente mediante simulaciones de qué manera influye el valor de la velocidad lineal V del robot en el seguimiento de la trayectoria. Recuerde que el valor de V lo fija el usuario ya que el controlador se encarga de ajustar sólo la velocidad angular.**

Cuando incrementamos la velocidad lineal del robot, podemos notar que este ya no es capaz de mantener una distancia constante respecto a la pared. Sin embargo, es posible hacer que el robot siga la pared sin colisionar o perderla de vista.

Video:

https://drive.google.com/file/d/1vosknN-uZk6qnIaNstvNtRp2W165VPLB/view?usp=share_link

5. **Pruebe el controlador realizado en el robot real y documente el funcionamiento de este controlador, identificando los posibles problemas derivados de realizar medidas reales con un sónar.**

Lamentablemente, no es posible realizar pruebas en el robot real en este momento debido a problemas de conexión. Por favor, disculpe las molestias.

Conclusión

En esta práctica hemos trabajado con robots móviles en entornos ROS-Matlab, explorando distintos aspectos del control y aprendiendo a diseñar controles de posición y seguimiento de paredes. A lo largo de la práctica, hemos realizado experimentos para evaluar la influencia de las constantes de control en la respuesta

del robot, documentando los resultados y analizando las posibles implicaciones de los mismos.

En particular, hemos estudiado la influencia de las constantes de control en la velocidad y precisión de los movimientos del robot. Hemos visto cómo el uso de valores grandes en las constantes de control acelera el movimiento del robot hacia su objetivo, pero también puede provocar un mayor error de posición final debido a que el robot puede experimentar un sobreimpulso al frenar. Por otro lado, el uso de valores bajos en las constantes de control proporciona mayor precisión y evita en la medida de lo posible los sobreimpulsos, pero el robot viajará a través del espacio designado a una velocidad más lenta.

También hemos estudiado cómo el uso de diferentes tipos de control, como el controlador PI o los filtros de Kalman extendido, pueden mejorar la precisión del movimiento del robot y reducir los errores de posición. Hemos visto cómo la introducción de la parte integral al controlador de velocidad angular permite que el robot tenga una mayor precisión en su movimiento hacia la posición deseada, rectificando continuamente el ángulo del robot respecto a la posición indicada para reducir los errores en su trayectoria.

Además, hemos evaluado el funcionamiento de los controles en simuladores y en robots reales, identificando los posibles problemas que pueden surgir en la medición de datos reales y en la implementación de los controles en un entorno real. Hemos aprendido a ajustar las ganancias del controlador si es necesario y a documentar los resultados de nuestros experimentos para poder analizarlos y extraer conclusiones útiles.

En conclusión, esta práctica ha permitido adquirir habilidades valiosas en la implementación de controles para robots móviles y en la evaluación de su desempeño en distintos escenarios y condiciones. Hemos aprendido a diseñar controles de posición y seguimiento de paredes, a evaluar la influencia de las constantes de control en la respuesta del robot y a identificar los posibles problemas que pueden surgir en la medición de datos reales y en la implementación de los controles en un entorno real. Estas habilidades son fundamentales para el desarrollo de robots móviles autónomos y para la realización de tareas complejas en entornos dinámicos y cambiantes.