PROGRAMACIÓN DE ROBOTS

Programación de robots móviles
PRÁCTICA 5.- Localización odométrica de robots móviles Lego Mindstorm EV3

El objetivo de la práctica consiste en realizar la localización del robot Lego usando su odometría, primero calculándola offline, y después calculándola online.

1.- Planteamiento del problema

Implanta un programa que mueva al robot recorriendo lo más exactamente posible un cuadrado (puedes usar una loseta del suelo del lab como guía) de manera continua, al menos una vez completa, y tomando muestras de los sensores propioceptivos y del tiempo. Para recorrer ese cuadrado hay que realizar dos tipos de movimiento:

- Para recorrer los tramos rectos, arranca los motores A y C a potencia igual a 50 durante un cierto tiempo;
- Para hacer los giros apaga el motor C durante otro cierto tiempo (dejando el A a máxima potencia).

El esquema general de tu programa debe ser parecido al que sigue:

1. Parte de inicialización. Debes definir los parámetros necesarios para tu código (por ejemplo, potencia de los motores, tiempos de giro, etc.).

2. Control de movimiento. Esta parte debe encargarse de planificar el movimiento del robot produciendo como salida los valores de actuación deseados para cada motor, de tal modo que sea capaz de seguir la trayectoria cuadrada pedida. Ajusta los valores de tiempo de tramo recto y de giro manualmente para tu robot hasta que consigas que siga el cuadrado lo mejor posible.

3. Cálculo de la odometría (sólo para el apartado 3). Esta parte debe estimar la *pose* actual del robot dada la anterior y las muestras de los sensores. La pose inicial se toma igual a cero para sus tres componentes.

4. Recogida de datos. Se deben guardar los datos necesarios para el cálculo posterior de la odometría para el apartado 2 (off-line), y la pose estimada en tiempo real en el robot para el apartado 3 (on-line), esencialmente, los valores del tiempo de simulación y de los sensores.

2.- Localización off-line

Una vez ejecutado el programa del apartado 1 satisfactoriamente, vamos a estimar offline la posición cartesiana que ha tenido el robot en cada momento de muestreo, usando los datos de tiempo y encoders que éste ha grabado en fichero, y el siguiente modelo cinemático:

$$\Delta x = \frac{\Delta\theta_l \cdot R + \Delta\theta_r \cdot R}{2} \cdot cos(\theta_0)$$
$$\Delta y = \frac{\Delta\theta_l \cdot R + \Delta\theta_r \cdot R}{2} \cdot sin(\theta_0)$$
$$\Delta\theta = \frac{\Delta\theta_r \cdot R - \Delta\theta_l \cdot R}{D}$$

donde R es el radio de la rueda del robot, y D la distancia entre ruedas (puedes medirla en el lab). ¿A qué modelo cinemático de los vistos en clase corresponde?

Ahora, escribe una función en Matlab (fuera del programa principal) que calcule la odometría del robot en un paso y que tenga la siguiente cabecera:

```
function [x1,y1,theta1]=odometry(x0,y0,theta0,t0,rotl0,rotr0,t1,rotl1,rotr1)
```

donde (*x0*, *y0*, *theta0*) es la última pose calculada para el robot, en metros y radianes, *t0* el tiempo (en milisegundos) cuando se calculó esa pose, y *rotl0* y *rotr0* lo que medían los sensores de rotación de cada rueda en aquel momento, en grados. En el primer paso todos esos valores serán 0. La rutina debe rellenar (*x1*, *y1*, *theta1*) con la nueva pose del robot, en metros y radianes, calculada en el momento actual mediante la medición del tiempo en milisegundos (que está en *t1*) y la de los encoders en grados (que están en *rotl1* y *rotr1*).

Si el incremento de tiempo entre una llamada y la anterior (es decir, *t1-t0*) es igual a cero, la rutina deberá devolver la misma pose que se le ha pasado, como si no hubiera habido movimiento. Dibuja en una gráfica la trayectoria seguida por el robot según esta función (todas sus poses, incluidas las orientaciones, representadas como segmentos de 1 milímetro orientados en la dirección del movimiento). Ten en cuenta que para ello debes hacer un nuevo *script* (no el que controla el robot) sino otro que emplee la rutina `odometry` y los datos del fichero generado en el apartado 1.

3.- Localización on-line

Basándote en la rutina que has tenido que implementar en Matlab para calcular la odometría en cada iteración, modifica ahora el código del apartado 1 para que el robot calcule, en tiempo real, su pose mediante el mismo modelo cinemático.

Recoge los datos devueltos de una ejecución de esta rutina (graba un vídeo del robot recorriendo el cuadrado, a vista de pájaro, en el que se aprecie bien la trayectoria seguida sobre el suelo) y superpón en una misma gráfica los resultados de la odometría calculada por Matlab para ese experimento y de la odometría calculada por el robot, y saca conclusiones con su comparación, así como con la comparación con la trayectoria real del robot en el vídeo.