Universidad de Buenos Aires, Facultad de Ingeniería

Práctica 4 - Mapeo y SLAM con Filtro de Partículas (FASTSLAM)

Robótica Móvil - Un enfoque probabilístico

Prof. Dr. Ignacio Mas

26 de mayo de 2022

Fecha límite de entrega: 9/6/22, 23:59hs.
Modo de entrega: Enviar por el Aula Virtual del Campus el código (.m) comentado y los gráficos (.jpg ó .pdf) y/o animaciones.

1. Mapeo con poses conocidas

Un robot debe construir un mapa de grilla de ocupación (celdas $c_0, \ldots, c_n$) de un entorno unidimensional usando una secuencia de mediciones de un sensor de distancia.
Asumir el siguiente modelo de sensor: cada celda de la grilla con una distancia menor que la distancia medida se asume ocupada con una probabilidad de $p = 0{,}3$. Cada celda más allá de la distancia medida se asume ocupada con una probabilidad de $p = 0{,}6$. Las celdas ubicadas a más de $20cm$ por detrás de la distancia medida no cambian su probabilidad de ocupación.

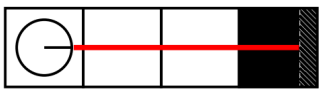

Figura 1.1: representación de grilla unidimensional

Calcular el mapa de grilla de ocupación resultante usando el modelo inverso del sensor con MATLAB/Octave. Usar un vector `m=0.5*ones(1,21)` para inicializar los valores de los *beliefs* (probabilidad de que las celdas estén ocupadas) y un vector `c=[0:10:200]`

como coordenadas de las celdas. Visualizar el *belief* resultante cada vez que el mapa se actualiza con una nueva medición usando `plot(c,m)`.

Resolución de la grilla	$10cm$
Logitud del mapa (1-D)	$2m$
posición del robot	c_0
orientación del sensor	mirando hacia c_n (ver figura)
mediciones (en cm)	101, 82, 91, 112, 99, 151, 96, 85, 99, 105
prob. *a priori*	0.5

2. Implementación de algoritmo FASTSLAM

En este ejercicio se implementará el algoritmo FASTSLAM basado en landmarks. Se asume que los landmarks son identificables por lo que el problema de asociación de datos está resuelto. El punto de partida es una estructura de código provista por la cátedra.

2.1. Notas preliminares

La estructura provista contiene los distintos componentes del algoritmo FASTLAM, para que el esfuerzo del desarrollo sea en los detalles de la implementación del algoritmo propiamente dicho. El archivo comprimido que se provee incluye las carpetas:

- **data** contiene la descripción del mundo y las lecturas del sensor
- **matlab** contiene la estructura del algoritmo FASTLAM con secciones para ser completadas
- **plots** guarda los gráficos generados como resultado

Para ejecutar el algoritmo, desde la carpeta *matlab* correr el archivo `fastslam.m`. Mientras que corre, los gráficos se van guardando en la carpeta *plots*. El algoritmo inicialmente está incompleto y no correrá correctamente. La estructura hace uso de la librería *librobotics*, escrita por Kai Arras (ASL-EPFL) para la visualización. Todas las funciones están definidas en la estructura de código.
Adicionalmente, se provee un archivo *notas_practica_fastslam.pdf* con detalles de la implementación del algoritmo FASTSLAM para usarse como guía.
Algunos consejos adicionales:

- Desactivar la visualización comentando la línea `plot_state(...)` en el script `fastslam.m` para acelerar la ejecución.
- Para probar el filtro sólo con algunos pasos, cambiar el tiempo final en el bucle *for* principal con, por ejemplo, `for t = 1:50`.

2.2. Paso de corrección FASTSLAM

Implementar el paso de corrección en el archivo `correction_step.m`. Asumir que el ruido de medición está caracterizado por la matriz diagonal cuadrada de 2×2 Q_t:

$$Q_t = \begin{pmatrix} 0{,}1 & 0 \\ 0 & 0{,}1 \end{pmatrix}$$

Nota: Con el comando *ffmpeg* (si está instalado en tu sistema operativo) se puede generar una animación de los gráficos de la carpeta *plots* de la siguiente manera:
`ffmpeg -r 10 -i fastslam_%03d.png fastslam.mp4`