Práctica 2. Mapeado 3D de Interiores y Sigue Líneas

Miguel Cazorla, Diego Viejo, Félix Escalona, Francisco Gómez
Departamento de Ciencia de la Computación e Inteligencia Artificial
Universidad de Alicante

29 de marzo de 2021

La segunda práctica tendrá dos apartados:

- Por un lado, consistirá en el desarrollo de un programa de mapeado 3D. Se utilizará la cámara kinect que monta el turtlebot para capturar nubes de puntos mientras el robot navega por el interior de una casa simulada. Estas nubes de puntos deben ir registrándose con el objetivo de obtener un modelo 3D del entorno. Esta parte puntúa 8/10.
- Por el otro, se hará una prueba con un nuevo sistema más sencillo para usar ROS, donde se desarrollará un método para hacer un sigue-líneas. Esta parte puntúa 2/10. Es obligatoria su realización y, además, se tendrá que rellenar una encuesta.

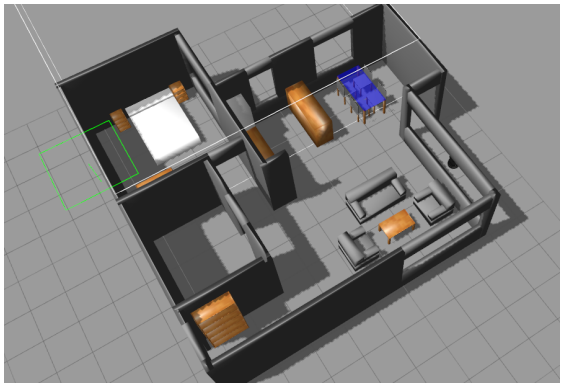

Figura 1: Mapa de la casa de la abuela Annie.

1. Mapeado 3D

El objetivo de esta primera parte es realizar el mapeado/modelo 3D de una planta de una casa, en concreto el modelo es el mismo que se usa para competiciones de robots como el Rockin. Para ello cargaremos en Gazebo el modelo

CAD de la casa en el que se insertará el ya familiar turtlebot. Como ya es conocido, turtlebot monta una cámara Kinect la cual se usará para capturar nubes de puntos. Conforme el robot se mueva por la casa, irá capturando nubes de puntos que deben ser registradas para, al final, conseguir un modelo 3D del entorno.

Para realizar la práctica se usará ROS, Gazebo y Point Cloud Library. Se creará un nodo ROS que controle el robot, así como también se encargará de la captura y registro de las nubes de puntos. El movimiento del robot puede ser teleoperado o autónomo.

1.1. Point Cloud Library y Registro

La técnica que se usará para registrar las nubes de puntos será similar a la utilizada para la generación de panorama. En la PCL existen varios métodos detectores de keypoints en nubes de puntos y varios métodos para calcular sus descriptores. Una vez obtenida esta información se utilizará alguno de los métodos que proporciona la librería para *alinear* las nubes e ir generando el modelo 3D de forma incremental. Se recomienda el uso de Voxel Grid como método de downsampling para ahorrar tiempo de cómputo.

Se deben probar algunos métodos detectores de keypoints junto con algunos métodos para calcular sus descriptores con el fin de obtener el registro de mayor calidad posible. Al menos dos de cada.

Vamos a detallar qué pasos tenemos que seguir para realizar la práctica. Vamos a suponer que el robot realiza, en su movimiento por el entorno, t tomas de datos, por lo que tendremos un conjunto de datos $X = x_1, x_2, ..., x_t$. Cada dato x_i es un conjunto de puntos 3D. Los pasos a seguir son los siguientes:

Entrada: Conjunto de datos X.

1: Transformación total $T_T = I$
2: Mapa 3D M
3: **para** i=1 hasta t-1 **hacer**
4: Extraer características $C_i = \{c_1^i, c_2^i, ..., c_m^i\}$ de los datos x_i.
5: Extraer características $C_{i+1} = \{c_1^{i+1}, c_2^{i+1}, ..., c_k^{i+1}\}$ de los datos x_{i+1}.
6: Encontrar emparejamientos $P = \{p_1, p_2, ..., p_l\}$ entre las características C_i y C_{i+1}. Cada emparejamiento es un par de características de cada conjunto de datos $p_j = \{c_a^i, c_b^{i+1}\}$
7: Obtener la mejor transformación T_i que explican los emparejamientos.
8: Obtener la transformación total $T_T = T_T * T_i$.
9: Aplicar al conjunto de datos C_{i+1} la transformación T_T para colocarlos en el sistema de coordenadas de C_1. Acumular el conjunto de datos transformados en M.
10: **fin para**
11: **devolver** M

Vamos a ir explicando en detalle los pasos a seguir.

1. Extracción de características. Este paso nos devolverá un conjunto de características C_i que será el resultado de aplicar un detector y un descriptor de características. Habrá que experimentar con las opciones disponibles para determinar cuál es el más adecuado (por tiempo de ejecución y eficacia).

2. Encontrar emparejamientos. Usaremos el método que proporciona PCL para encontrar las correspondencias. El resultado de este paso es un conjunto de emparejamiento.

3. Como hemos comentado antes, es posible que haya muchos malos emparejamientos. En este paso tenemos que determinar la mejor transformación que explica los emparejamientos encontrados. Para ello, usaremos el algoritmo RANSAC.

4. Por último, hay que construir el mapa. Como cada toma de la Kinect tiene aproximadamente 300.000 puntos, en el momento que tengamos unas cuantas tomas vamos a manejar demasiados puntos, por lo que hay que proceder a reducirlos. Para ellos, podemos usar el filtro de reducción VoxelGrid, disponible en PCL.

2. Sigue Líneas

En este apartado, se implementará un método que permita a un robot seguir un trazado (mediante una línea). Al robot se le asignará una velocidad lineal constante y, procesando la imagen, se le irá suministrando una velocidad angular en función de la posición relativa de la línea en la imagen.

Vamos a usar un sistema distinto y, pensamos, más sencillo de trabajar que con ROS. Realmente, el nuevo sistema usa ROS por debajo, pero aísla toda la instalación, conexión entre nodos, etc. Es una prueba piloto para una futura adopción de este sistema.

El nuevo sistema se llama Unibotics. Debemos ir a `https://unibotics.org/`. Primero tenemos que registrarnos, tanto en el propio sistema (Sign up) como en el foro (Forum). Cualquier duda que tengáis se va a poner en ese foro.

Para darnos de alta, nos pide un Register Code. Usaremos paizon. Una vez dentro, seleccionamos el ejercicio a realizar. En una terminal de nuestro ordenador ejecutamos: sudo docker run -it -p 8080:8080 -p 7681:7681 -p 2303:2303 -p 1905:1905 -p 8765:8765 jderobot/robotics-academy python3.8 manager.py

El ejercicio a realizar es el Follow Line. En `http://jderobot.github.io/RoboticsAcademy/exercises/AutonomousCars/follow_line/` tenéis una explicación sobre cómo se puede resolver la práctica. No hagais caso a la primera parte, de puesta en marcha, solo al apartado de Theory. El objetivo es que el robot se quede centrado en la línea.

3. Documentación a entregar

La documentación de la práctica es una parte muy importante en la puntuación final (un 60 % de ella). El código debe estar debidamente comentado, indicando qué se hace en cada punto. Además, se debe entregar una documentación (cualquier formato: PDF, HTML, etc.) con los siguientes puntos:

1. Descripción de lo que se pretende hacer. Resumen del conjunto de la práctica.

2. Descripción de cómo se ha realizado la práctica (consideraciones, problemas encontrados, etc.)

3. Implementación y código desarrollado.

4. Nube de puntos resultado del registro.

Así mismo, se pide la entrega de un vídeo donde se demuestre el funcionamiento de cada una de las partes de la práctica.

Normas de entrega de la práctica:

- La práctica se podrá realizar por parejas.
- La práctica se entregará antes de las 24 horas del domingo 23 de mayo de 2021.
- La entrega se realizará a través del Moodle de la asignatura.