Facultad de INGENIERÍA Sede Bogotá | UNIVERSIDAD NACIONAL DE COLOMBIA

Hector SLAM con RPLIDAR A1- Kobuki

ROS Kinetic

Alejandro Ojeda Olarte
Ricardo Ramirez

Facultad de INGENIERÍA Sede Bogotá | UNIVERSIDAD NACIONAL DE COLOMBIA

Contenido

Descripción

SLAM 1.1

El Mapeo y Localización Simultáneos o SLAM (Simultaneous Localization and Mapping) por sus siglas en inglés, es una técnica para construir un mapa de un entorno desconocido en el que se encuentra, a la vez que estima su trayectoria al desplazarse dentro de este entorno.

Hector SLAM 1.2

Hector SLAM es un algoritmo para SLAM 2D, para los cuales también se encuentran Gmapping, KartoSLAM, CoreSLAM y LagoSLAM. Hector SLAM permite realizar mapeo sin uso de odometría, por medio de información de alta resolución y alta frecuencia de láser. Este algoritmo es considerado el estado del arte en cuestión de mapeo basado en filtrado de partículas. De igual manera para uso con ROS cuenta con soporte para uso con el RPLidar y el Hokuyo UTM-30LX.

RPLIDAR A1 1.3

El RPLIDAR A1 es un LIDAR de bajo costo que permite hacer captura 2D de 360 grados, desarrollado por RoboPeak, y actualmente producido por SLAMTEC. Tiene una frecuencia de 5.5Hz hasta 10Hz, capturando 360 puntos por cada revolución. Tiene un rango de captura de 12m y puede tomar hasta 8000 datos por segundo. Cuenta con alimentación de 5V.

Imagen obtenida de los datos técnicos del fabricante.

Kobuki 1.4

El Kobuki es una plataforma de desarrollo para robótica móvil de bajo costo. Es manufacturado por Yujin Robot. Este es un robot diferencial que llega hasta los 70cm/s y girá a un máximo de 180 deg/s. Su capacidad de carga útil en piso liso es de 5 Kg y en tapete es de 4 Kg. Cuenta con un puerto USB B para uso con un computador o tarjeta de desarrollo con puerto USB; así como con un puerto paralelo por los pines RX/TX. Su batería es de Litio-Ion de 2200mAh (4S1P).

Instalación de librerías en ROS Kinetic

Primero, comenzando desde el workspace (que en este caso se tomará como *catkin_ws* del usuario *SU_USUARIO*). Su terminal debería verse similar a lo siguiente:

```
~/home/SU_USUARIO/catkin_ws/$ cd src/
```

Por temas de simplicidad se omitirá la ruta de la consola y sólo se tomará desde `$`.

Primero se instalan los paquetes que se instalan desde apt.

```
$ sudo apt-get install ros-kinetic-kobuki ros-kinetic-kobuki-core
$ sudo apt-get install ros-kinetic-map-server
```

En esta carpeta se descarga la librería de Simulación de Kobuki y de Hector SLAM para RPLidar.

```
$ git clone https://github.com/yujinrobot/kobuki_desktop
$ git clone https://github.com/NickL77/RPLidar_Hector_SLAM
```

Luego se procede a reconstruir el espacio de trabajo:

```
$ cd ..
$ catkin_make
```

Una vez terminado el proceso es necesario realizar el comando *source* para que nuestra terminal esté actualizada con el paquete recién construido. Esto se debe realizar en cada terminal.

```
$ source devel/setup.bash
```

De esta manera ya se tiene instalado los paquetes.

Configuración

Conexión remota 3.1

Es necesario tener dos dispositivos para la operación remota efectiva con el kobuki, una que corresponde al maestro, desde donde se controlará el movimiento y se registra el SLAM, y otra donde está conectado el robot y el RPLidar. Se usá la guía para conexión remota para este fin.

RPLIDAR 3.2

Luego se configura el RPLidar acorde a la guía de uso. Dado que la librería de Hector SLAM ya viene con rplidar_ros, **no es necesario realizar los pasos de instalación, solo configuración**. Es importante conectar primero este dispositivo y luego otorgar los permisos de lectura y escritura del puerto, para que (acorde a la configuración del paquete) quede en `ttyUSB0.`

Kobuki 3.3

El Kobuki que se encuentra a disposición en el laboratorio tiene una estructura para adaptar sensores y alojar una computadora portátil, como se ve a continuación.

Se debe verificar si el robot está cargado previamente. Para cargarlo se conecta el adaptador con el que viene su alimentación.

Contiguo al puerto de carga se encuentra el botón para encender el robot. Este emite un sonido como el que se tiene a continuación. Sonido de Inicio.mp4. Luego en los puertos, se conecta el cable USB.

Ejecución

Inicialmente, en una ventana de la consola de comandos o Terminal, comenzamos a correr ROS. Esto se debe realizar en el dispositivo maestro.

```
$ roscore
```

Posteriormente, en el dispositivo esclavo, se conecta el Kobuki con el puerto USB, y se corre el comando para cargar el control del robot en una consola diferente.

```
$ roslaunch kobuki_node minimal.launch
```

Al correr este comando, cuando el robot se conecta con el programa se escucha el siguiente sonido Sonido de Conexion.mp4.

Posteriormente, se corre el comando que habilita el uso del RPLidar en otra consola más.

```
$ roslaunch rplidar_ros rplidar.launch
```

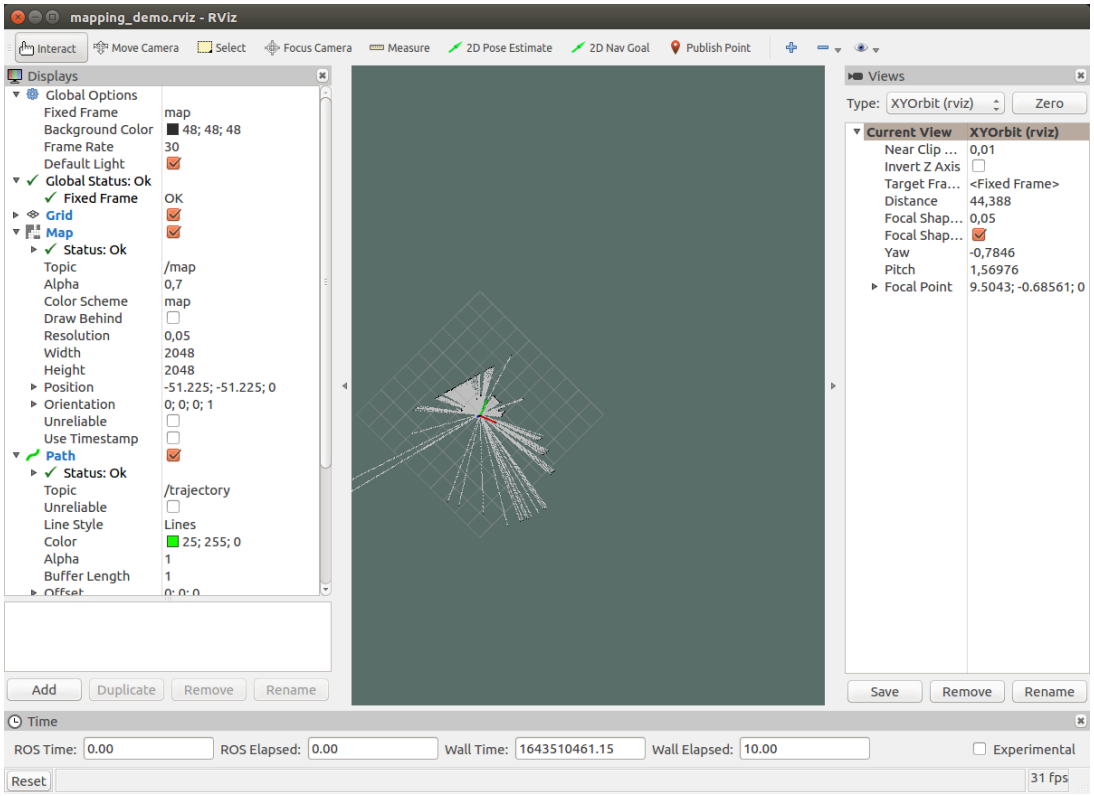
Finalmente, en el dispositivo maestro se corre en dos consolas diferentes a la que está ejecutando ROS, los siguientes comandos. El primero es para teleoperación del Kobuki:

```
$ roslaunch kobuki_keyop keyop.launch
```

El segundo, comienza a ejecutar el algoritmo Hector SLAM, alimentándose del tópico del RPLidar.

```
$ roslaunch hector_slam_launch tutorial.launch
```

Este comando ejecuta una ventana de RViz donde se puede observar el mapa que se va generando, donde en negro aparecen los obstaculos, en gris el espacio por el que el robot puede moverse, y en una línea verde, la trayectoria que registra el robot.



Paralelo a este proceso, si se desea guardar todos los tópicos que están corriendo, se recomienda usar la función de rosbag.

```
$ rosbag record -a
```

Después del tiempo que se capture la muestra que se desea tomar, terminamos la tarea utilizando Ctrl + C. Este archivo se almacena en un formato .bag, con un nombre acorde a la fecha, por ejemplo “ 2022-01-06-11-57.bag ”.

Finalmente cuando se quiera guardar el mapa, sea en vivo o por reproducción del archivo bag, se procede a mover una consola a la carpeta donde se quiere guardar el mapa, que será en dos archivos, uno en yaml y otro en pgm. Allí se corre el siguiente comando:

```
$ rosrun map_server map_saver -f Nombre_del_archivo
```

Donde Nombre_del_archivo corresponde al nombre con el que quedan guardados los archivos.

Referencias

- Ojeda A. Taller Modelo Cinemático. Taller 1 - Robótica y Control Servo-Visual. Universidad Nacional de Colombia.
- Saat, S., Abd Rashid, W. N., Tumari, M. Z. M., & Saealal, M. S. (2020, April). Hectorslam 2d Mapping for Simultaneous Localization and Mapping (slam). In *Journal of Physics: Conference Series* (Vol. 1529, No. 4, p. 042032). IOP Publishing.
- Nagla, S. (2020, December). 2D Hector SLAM of Indoor Mobile Robot using 2D Lidar. In *2020 International Conference on Power, Energy, Control and Transmission Systems (ICPECTS)* (pp. 1-4). IEEE.
- About Kobuki. Iclebo. http://kobuki.yujinrobot.com/about2/
- http://wiki.ros.org/hector_mapping
- https://automaticaddison.com/how-to-build-an-indoor-map-using-ros-and-lidar-based-slam/#Save_the_Map