Taller 5: ROS y V-Rep

Introducción a la Robótica Movil

April 12, 2018

1 Introdución

El objetivo de este taller es familiarizarlos con el entorno de desarrollo **ROS** (Robot Operating System), el simulador **V-Rep** e introducir las herramientas básicas para el desarrollo de sistemas de navegación complejos.

1.1 Descarga e Instalación de V-Rep

Para este Taller es necesario descargar el simulador **V-Rep** de la pagina oficial. Una vez descargado el simulador es necesario copiar el archivo **libv_repExtRosInterface.so** en la carpeta de **V-Rep**. Descargarlo de la página de la materia. Este archivo configura el simulador para poder trabajar con **ROS** y los paquetes requeridos para el taller.

1.2 Configurar variables de entorno de ROS

Dado que uno podría tener varias versiones de **ROS** instaladas es necesario seleccionar la versión que se utilizará y configurar las variables de entorno correspondientes. Para esto es necesario ejecutar las siguientes líneas de comando **por única vez**:

```
$ echo "source /opt/ros/kinetic/setup.bash" >> ~/.bashrc
$ source ~/.bashrc
```

De esta manera, las variables de entorno de **ROS** serán accesibles para toda **nueva** consola de comandos. Se recomienda cerrar y volver a abrir toda consola de comandos abierta para asegurar la carga de las variables de entorno necesarias por **ROS**.

NOTA: Para confirmar que las variables de entorno estén correctamente configuradas es posible ejecutar:

```
$ rosversion -d
```

Debería responder: "**kinetic**".

1.3 Configurar el Catkin workspace

Catkin es el nombre del sistema de compilación y manejo de paquetes integrado en **ROS**. Los workspaces de **Catkin** facilitan trabajar con varios paquetes al

mismo tiempo y definen un entorno unificado para la ejecución de diversos nodos.
La materia provee de un workspace de **Catkin** con los paquetes necesarios para el taller, **este workspace aún no se encuentra inicializado**. Para poder configurar el workspace y compilar los paquetes primero se debe **ejecutar en la raíz del mismo (/catkin_ws/)** el siguiente comando:

```
$ catkin init
```

Es posible listar los paquetes presentes en el entorno de desarrollo **Catkin** utilizando el comando:

```
$ catkin list
```

Para compilar los paquetes presentes se debe utilizar el comando:

```
$ catkin build
```

Como resultado se crearán varias carpetas. Un workspace se compone principalmente de 3 directorios:

- **/build**: Contiene los archivos generados durante la compilación de los paquetes. **Catkin** utiliza un sistema de compilación llamado **CMake**[1].
- **/devel**: Provee un entorno de desarrollo en el cual los nodos de los paquetes se encuentras disponibles para ser ejecutados.
- **/src**: Contiene los paquetes con los cuales se trabajará. Estos proveen definiciones de mensajes, nodos, directivas de compilación y establecen dependencias entre paquetes.

Aún cuando los paquetes hayan compilado de manera exitosa, es posible tener varios workspaces **Catkin** creados. Por lo que es necesario configurar las variables de entorno del workspace actual ejecutando **en la raíz del workspace**:

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

NOTA: Esta configuración solo es válida en la consola de comandos en la que se ejecuto, por lo que si se cierra la consola es necesario volver a ejecutar el comando de configuración para **el workspace con el que se desea trabajar**.

Ejercicio 1: Interfaz con V-Rep

La materia provee de una versión de **V-Rep** previamente configurada para comunicarse con **ROS** de manera de poder publicar y suscribirse a tópicos. Para establecer la comunicación correctamente es necesario iniciar primero el **ROS Master** ejecutando **en una consola aparte**:

```
$ roscore
```

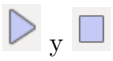
Una vez iniciado el **ROS Master**, es posible ejecutar **V-Rep** ejecutando **desde su directorio raíz**:

```
$ ./vrep.sh
```

[1]Popular sistema de compilación capaz de trabajar en numerosos sistemas operativos. https://cmake.org/.

Una vez abierto el **V-Rep** deben **cargar la escena** preparada para la ejercitación clickeando en: **File → Open scene...** y abrir el archivo: **/catkin_ws/src/ros_intro/vrep/ros_intro.ttt**

La escena presentada se compone de un único robot **Pioneer**, para iniciar y detener la simulación deben hacer click en los botones:

 y

Luego de haber iniciado la simulación de la escena, se pide:

a) Utilizar comandos de **ROS** para descubrir que tópicos se suscribe y publica el **V-Rep**.

b) Controlar el robot enviando comandos de velocidad (**geometry_msgs/Twist**).

c) Enviar comandos de velocidad al robot de manera que se encuentre **constantemente** girando en circulo.

Ejercicio 2: Nodo de tele-operación

Implementaremos un nodo de **ROS** que nos permita enviar comandos de velocidad al robot simulado en **V-Rep** utilizando el teclado. Para esto primero es necesario tener el workspace de **Catkin** correctamente configurado (**revisar sección: 1.3**).

Dentro del paquete **ros_intro** (ubicado en **/catkin_ws/src/ros_intro**) existe un directorio **/src/** donde encontrarán el código de fuente "esqueleto" de un nodo para traducir eventos del teclado a comandos de velocidad. Este nodo se llama **keys_to_twist** y se suscribe a los tópicos **'/keyboard/keyup'** y **'/keyboard/keydown'**. Los mensajes de dichos tópicos se reciben por los métodos **on_key_up(..)** y **on_key_down(..)** presentes en el archivo **keys_to_twist.cpp**.

Para compilar el nodo ejecuten desde la raíz del workspace **Catkin**:

```
$ catkin build
```

Para iniciar el nodo de manera que pueda comunicarse con **V-Rep** correctamente utilizar:

```
$ roslaunch ros_intro ros_intro_vrep.launch
```

NOTA: Utilizar **Ctrl + C** para detener la ejecución de cualquier nodo.

Se pide:

a) Completar las áreas marcadas de código en los métodos **on_key_up(..)** y **on_key_down(..)** de manera de publicar un **mensaje Twist** por el tópico **'/robot/cmd_vel'** (**revisar las slides de las clases**).

NOTA: Las unidades en **ROS** son en **metros por segundo** para la velocidad lineal y **radianes por segundo** para la velocidad angular.

b) Iniciar el **ROS Master**, Abrir el simulador **V-Rep**, iniciar la simulación y ejecutar los nodos utilizando el **comando roslaunch correspondiente**. Visualizar el comportamiento del robot en la simulación ¿Que nodos están tomando parte del sistema en ese momento?.

c) Teniendo en cuenta lo visto en la primera parte de la materia:
¿Qué nodo esta operando los motores de las ruedas? ¿Cúal es la diferencia en la forma de controlar el robot si la comparamos con el taller anterior?

Ejercicio 3: Visualizar en RViz

RViz es muy interesante para visualizar información que viaja por los tópicos del sistema. Para iniciar RViz es necesario ejecutar el comando:

```
$ rviz
```

Iniciar el **ROS Master**, Abrir el simulador **V-Rep**, iniciar la simulación y ejecutar los nodos utilizando el **comando roslaunch**:

a) Visualizar los marcos de referencia publicados por el sistema:
Hacer click en Add → Seleccionar en la lista: TF → OK

NOTA: RViz necesita que se defina algún marco de referencia como **"fijo"**. Seleccionen el marco fijo utilizando la opción desplegable:
Global Options → Fixed Frame → Seleccionar el marco deseado..

b) Listar los tópicos activos y visualizar la información odométrica provista por **V-Rep**:
Hacer click en Add → Seleccionar la paleta: "By topic" → Seleccionar Odometry

c) Utilizar el teclado para enviar comandos de velocidad al robot mientras se cambia el **"marco fijo"** en **RViz**. ¿Que sucede con los marcos **no fijos** al moverse el robot? ¿Como es que se mueven?.

Para visualizar el árbol de transformaciones, utilizar el comando:

```
$ rosrun rqt_tf_tree rqt_tf_tree
```

Anexo: Paquetes instalados para este taller

Adicional a los archivos provistos por la materia existen varios paquetes adicionales instalados en las computadoras del laboratorio. En caso de querer reproducir el taller les dejamos especificado la configuración y paquetes instalados en las computadoras del laboratorio. **No daremos instrucciones especificas para la instalación de estos paquetes**, por lo que deberán investigar en las páginas correspondientes.

- **Sistema operativo:** Ubuntu 16.04
- **ROS:** Versión Kinetic (por defecto en Ubuntu 16.04)
- **Catkin Tools:** Herramientas adicionales para el manejo de workspaces. https://catkin-tools.readthedocs.io/

- **libSDL:** Librería para el manejo de entradas de teclado (requerido por el paquete **keyboard**).
  https://www.libsdl.org/
- **V-Rep:** http://www.coppeliarobotics.com/
- **V-Rep Ros Interfase:** Plugin para **V-Rep**
  http://www.coppeliarobotics.com/helpFiles/en/rosInterf.htm