Taller 1 Robótica

Santiago Martínez Castaño
Departamento de Ingeniería
Eléctrica y Electrónica
Universidad de los Andes, Bogotá
s.martinezc@uniandes.edu.co

Martin David Galvan Castro
Departamento de Ingeniería
de Sistemas y Computación
Universidad de los Andes, Bogotá
md.galvan@uniandes.edu.co

Santiago Montaño Díaz
Departamento de Ingeniería
Eléctrica y Electrónica
Universidad de los Andes, Bogotá
vs.montano@uniandes.edu.co

I. Introducción

La solución del Taller 1 permite familiarizarse con los conceptos básicos de ROS, así como una puesta en práctica en la creación de nodos, suscripción y publicación a tópicos, creación de servicios y la interconexión de estos. El desarrollo está basado al rededor de una simulación en V-REP para el robot Turtlebot2, para esta se han creado nodos que permiten manipular el robot por su ambiente, graficar su posición en el espacio y recrear recorridos anteriores guardados.

El presente informe se divide en 2 partes principales: En la sección II se presenta el desarrollo de los nodos mencionados anteriormente, así como su conexión a la simulación y una breve explicación a manera de macroalgoritmo. Esta sección permitirá entender cómo se ha desarrollado la solución a los problemas planteados sin entrar en mayor detalle del código. Luego, en la sección III se explicará al usuario cómo correr la simulación y comprobar el funcionamiento de los nodos desarrollados, así como las herramientas adicionales requeridas para su correcta ejecución.

II. Desarrollo

Punto 1 - /turtle_bot_teleop

El nodo **/turtle_bot_teleop** permite al usuario manipular el robot Turtlebot2 en el entorno de simulación de V-REP a través de las flechas del teclado. Este comportamiento se logra a través de la herramienta 'pynput' para capturar las acciones del usuario en el teclado [1], la instalación de esta herramienta se explicará en la sección III-A de este informe.

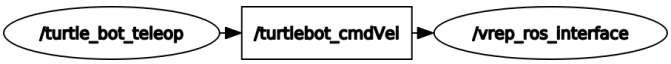

Figura 1. Grafo de conexiones del nodo /turtle_bot_teleop

El comportamiento del nodo se ha resumido en la Figura 1, dónde se observa al nodo **/turtle_bot_teleop** publicando en el tópico **/turtlebot_cmdVel** a través de un mensaje de tipo **geometry_msgs/Twist**, este tipo de mensaje contiene 2 vectores de 3 posiciones con información lineal y angular en los 3 ejes del espacio. Así el nodo publica la velocidad lineal y angular deseada según las teclas presionadas en el teclado por el usuario.

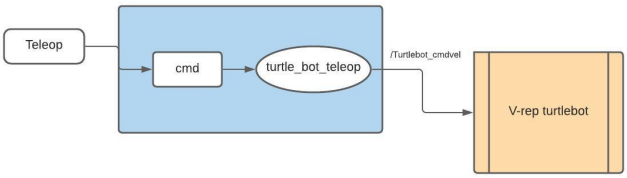

Figura 2. Macroalgoritmo del nodo /turtle_bot_teleop

Punto 2 - /turtle_bot_position

Para observar la posición del robot Turtlebot2 en el plano XY a través de una gráfica en tiempo real se ha construido el nodo **/turtle_bot_position**. Esta gráfica se ha construido con la librería 'matplotlib' [2], cuya instalación se explicará en la sección III-A de este informe. Con esta herramienta se logra graficar la trayectoria del robot y su posición actual, representada como un punto en la gráfica.

Figura 3. Grafo de conexiones del nodo /turtle_bot_position

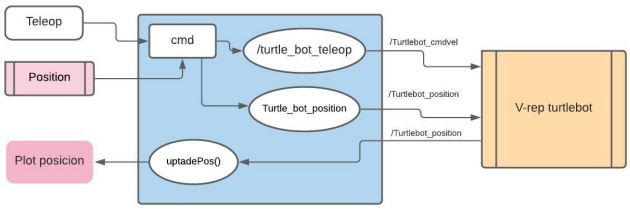

Figura 4. Macroalgoritmo del nodo /turtle_bot_position

La funcionalidad deseada de este nodo se obtiene a través de las conexiones mostradas en la Figura 3, dónde se observa que el nodo **/turtle_bot_position** se suscribe al tópico **/turtlebot_position** del cual obtiene un mensaje de tipo **geometry_msgs/Twist** con la información de la posición XY del robot. Este mensaje permite actualizar la gráfica cada vez que exista un cambio de posición, representando la trayectoria del robot en tiempo real.

Punto 3 - Complemento de /turtle_bot_teleop

Como complemento al nodo presentado en el **Punto 1** se ha añadido la funcionalidad de preguntar al usuario si desea guardar el recorrido del robot, en caso de que la respuesta sea afirmativa, se guarda en un archivo de texto la secuencia de acciones que realizó el usuario durante el recorrido del robot. El nombre del archivo en el que se guardará este recorrido también es especificado por el usuario.

Punto 4

Para cumplir con la funcionalidad deseada, se crearon dos nodos, estos son el nodo Servidor y el nodo Cliente. El nodo servidor se encarga de recibir un String con el nombre del archivo a ejecutar y verificar la existencia en la carpeta **results**. En caso de encontrarlo le enviara al cliente la ruta del archivo, en caso contrarió enviara un mensaje diciendo que no lo encontró.

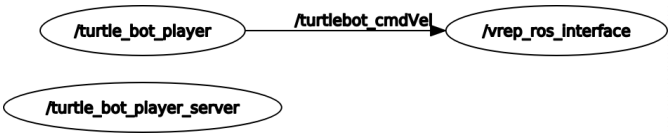

Figura 5. Grafo de conexiones del servicio

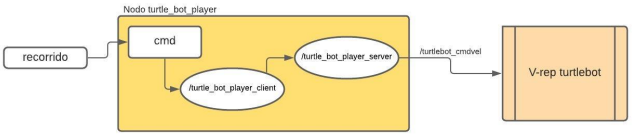

Figura 6. Macroalgoritmo /turtle_bot_player

El nodo cliente se encarga de recibir el nombre del archivo, y leerlo linea por linea. Primero obtiene las velocidades a las cuales se hizo el recorrido y después lee linea por linea, se procesa el texto de la linea para traducirlo a un comando y después se manda un mensaje por el tópico **/turtlebot_cmdVel**.

III. Instrucciones de Ejecución

Para comprobar el funcionamiento de los nodos creados es necesario que el dispositivo que ejecuta la simulación disponga de herramientas de terceros que han sido empleadas en el código de esta solución. Por esto, en la sección III-A se explicará cómo instalar todas las librerías adicionales requeridas para el funcionamiento de los nodos en ROS. Posterior a esto, en la sección III-B, se explicará cómo ejecutar la simulación y comprobar el funcionamiento de los nodos desarrollados.

Es importante que maneja la versión de Python3 en el equipo dónde se va a correr la simulación, y que tenga instalado el manejador de paquetes de Python3 'pip3' para la instalación de las siguientes librerías. Si está utilizando la máquina virtual del curso de robótica ya contará con esta versión de Python. Si no lo tiene, pude obtenerlo con el siguiente comando:

```
$ sudo apt install python3-pip
```

III-A. Herramientas adicionales

Librería *pynput* [1]

Esta librería se utiliza para capturar las acciones del usuario en el teclado, permitiendo manipular el robot a través de las flechas del teclado en el nodo **/turtle_bot_teleop**. Para instalar esta librería utilice el siguiente comando en una terminal:

```
$ pip3 install pynput
```

Librería *matplotlib* [2]

Esta librería se utiliza para graficar datos en Python, y ha sido utilizada en el nodo **/turtle_bot_position** para representar en tiempo real la posición del robot. Adicionalmente se uso la librería PyQT5 que permite a *matplotlib* ser usada de manera concurrente. Para instalar esta librería utilice los siguientes comandos en una terminal:

```
$ pip3 install matplotlib
$ pip3 install pyqt5
```

III-B. Prueba de funcionamiento de los nodos

Para comprobar el funcionamiento de los nodos es necesario correr primero la simulación de V-REP para el Turtlebot2, para esto abra una terminal y corra el *roscore* de ROS:

```
$ roscore
```

En otra terminal diríjase a la carpeta en donde se encuentra el simulador V-REP y ejecute el script *vrep.sh*. Si está utilizando la máquina virtual del curso de robótica será:

```
$ cd Documents/Programas/
$ cd V-REP_PRO_EDU_V3_6_2_Ubuntu18_04/
$ ./vrep.sh
```

Cuando V-REP se haya ejecutado abra la escena de la simulación en el menú File→Open. Ejecute la simulación para comprobar el funcionamiento de los nodos.

Ahora descomprima el archivo *taller1_grupo4.tar.gz* y compile el paquete ROS utilizando los siguientes comandos en su entorno de trabajo para ROS, suponiendo que es *catkin_ws*:

```
$ tar -xzvf taller1_grupo4.tar.gz
$ cd catkin_ws/
$ source devel/setup.bash
```

Nodo /turtle_bot_teleop

Para ejecutar este nodo abra una terminal y corra el nodo con dicho nombre perteneciente al paquete de ROS **turtle_bot_4**:

```
$ rosrun turtle_bot_4 turtle_bot_teleop.py
```

Nodo /turtle_bot_position

Para ejecutar este nodo abra una terminal y corra el nodo con dicho nombre perteneciente al paquete de ROS **turtle_bot_4**:

```
$ rosrun turtle_bot_4 turtle_bot_position.py
```

Para que la gráfica se guarde de manera correcta, es necesario detener la simulación en V-REP. Después se puede hacer una interrupción por teclado.

Nodos /turtle_bot_player cliente y servidor

Para ejecutar estos nodos es primero ejecutar el nodo cliente con:

```
$ rosrun turtle_bot_4
      turtle_bot_player_server.py
```

Despues se puede ejecutar el nodo cliente de la siguiente forma, donde ¡nombreArchivo¿es el nombre del archivo a abrir. Este tiene que estar entre comillas, como se muestra a continuación:

```
$ rosrun turtle_bot_4
turtle_bot_player_client.py "<nombreArchivo>"
```

REFERENCIAS

[1] M. Palmér, *Librería pynput para Python*, 2016. Disponible en https://pypi.org/project/pynput/ - Versión: 1.7.1.

[2] J. D. Hunter and M. Droettboom, *Librería matplotlib para Python*, 2010. Disponible en https://pypi.org/project/matplotlib/ - Versión: 3.3.1.