Proyecto 1: Generación de trayectorias en ROS

Héctor H. Huipet Hernández
Instituto Tecnológico Autónomo de México

I. INTRODUCCIÓN

Este proyecto funcionó de tutorial para familiarizarse con el entorno de trabajo de ROS así como de recordar e integrar la programación en el lenguaje C++ con comandos de ROS para poder cumplir con la tarea de desplazar al robot dada una pose final elegida por el usuario.

La instalación y configuración del entorno de trabajo de ROS se realizó mediante el estudio y práctica de los primeros tres capítulos del libro-tutorial *"A gentle introduction to ROS"* [1]. El primer capítulo será cubierto en el marco teórico.

El segundo capítulo explica la configuración e instalación en Ubuntu 14.04 y la versión de ROS Indigo. Experiencias de compañeros de clase mostraron que algunas librerias no eran compatibles con Ubuntu 16.04 y que la versión de mac también presentaba inconsistencias para la instalación.

El tercer capítulo trata fundamentalmente de programar tres "paquetes" con funciones diferentes. El primer paquete es un simple programa *Hello World* con el que se muestra el uso de la salida estándar de información de texto. Más allá de la simpleza de la aplicación, se explica por primera vez la composición de un paquete de ROS y se da un ejemplo base de los documentos anexos que lo componen: **package.xml** y **CmakeLists.txt**. Básicamente package define al paquete en cuanto a dependencias de otros elementos de ROS así como la información para identificar al paquete. CmakeLists es un script para Cmake, el cual continene una serie de instrucciones de construcción del paquete así como de los ejecutables pertinentes.

El documento restante se presenta de la siguiente forma. La sección dos, expone un breve resumen sobre los conocimientos básicos de ROS, así como teoría de trayectorias. La sección tres, muestra algunos experimentos con el código del proyecto. Por último, la seccion cuatro concluye realizando un resumen de lo encontrado y las lecciones aprendidas en el desarrollo de este proyecto.

II. MARCO TEÓRICO

ROS siglas de *Robot Operating System* fue desarrollado por la comunidad de robótica para ayudar a los desarrolladores de software a tener un meta-sistema operativo de código abierto. ROS abstrae los controles de dispositivos de bajo nivel, interacciones de hardware, paso de mensajes entre procesos y manejo de paquetes. También provee de librerias para construir aplicaciones a través de multiples tipos de computadoras.

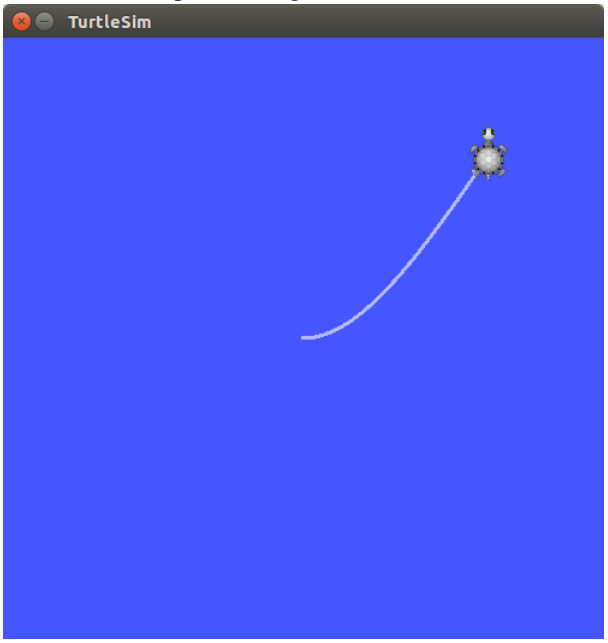
El concepto más utilizado en ROS y en este documento es el de paquete, un paquete es una unidad funcional que define un conjunto de operaciones que en el momento que se instancia para su ejecución cambia de nombre a nodo y se integra al ambiente provisto por ROS para comunicarse o recibir comunicación de otros nodos. Un paquete puede instanciarse multiples veces mientras se realice un cambio de nombre.

Para este proyecto la funcionalidades más representativas fueron las de publicar y subcribirse a un tópico. Un tópico es un canal de información en ROS dónde se puede observar y/o escribir información para un determinado fin. Un paquete que publica información requiere de un objeto que contiene el nombre del tópico al cual va a publicar, asi como el tipo de información que va a transmitir. Un paquete que se subscribe a un tópico requiere también de un objeto que conoce el nombre del tópico así como el tipo de mensajes que se publican ahí. El primero se ejecuta puede proseguir a realizar otras operaciones, mientras que el segundo se mantiene al pendiente de los mensajes que lleguen al tópico.

III. EXPERIMENTOS

Los experimentos del proyecto se presentan a continuación.

Tomando como inicio la pose predeterminada por turtlesim_node (x: 5.54, y: 5.54, ángulo: 0). Se procede a realizar un desplazamiento a el punto (9,9) con orientación de 90 grados en un tiempo de 7 segundos



```
ubuntu-win@ubuntuwin-Virtual: ~/rosProject
Ingrese pose final:
Valor en X: 9
Valor en Y: 9
Angulo: 1.57
Ingrese tiempo de ejecucion (segundos): 7
[ INFO] [1472789497.315098492]: La pose inicial es = (5.54,5.54) direccion=0.00
[ INFO] [1472789497.620085753]: La pose actual es = (5.73,5.56) direccion=0.17 d
istancia al obj: 4.75
 •••
[ INFO] [1472789504.614941257]: La pose actual es = (8.97,8.95) direccion=1.40 d
istancia al obj: 0.06
[ INFO] [1472789504.914734758]: La pose actual es = (8.97,8.95) direccion=1.45 d
istancia al obj: 0.06
tiempo final: 5.31
tiempo final: 5.54
ubuntu-win@ubuntuwin-Virtual:~/rosProject$
```

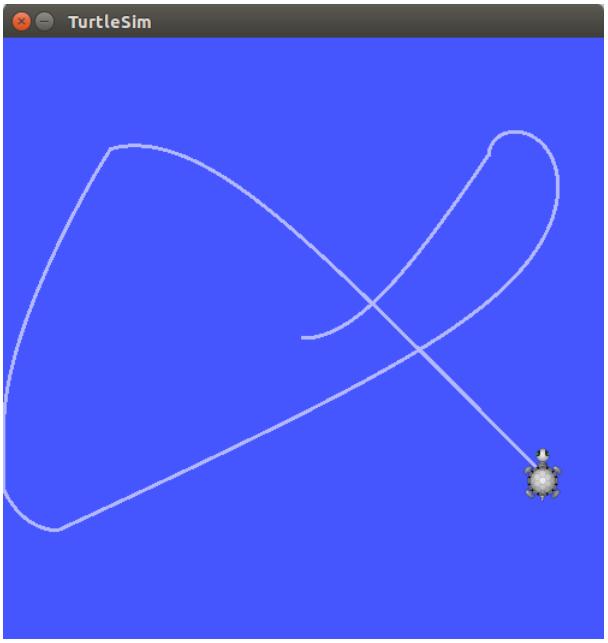
Posteriormente se vuelve a correr el programa ahora para desplazarse a la posición x:1, y:2, con ángulo 180 grados y una velocidad de 8 segundos:

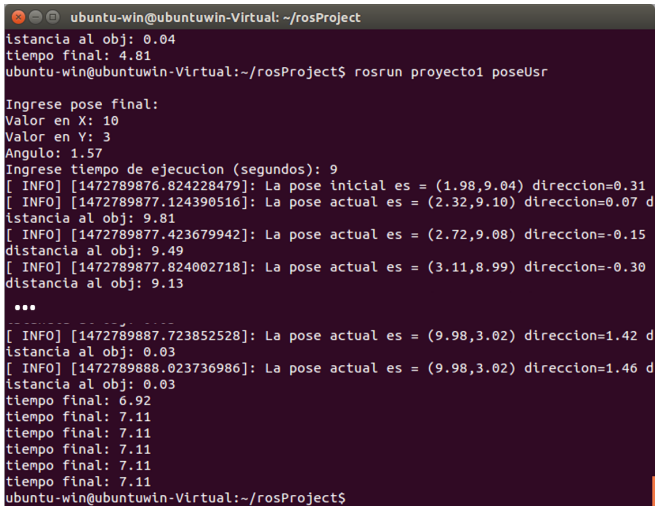

```
ubuntu-win@ubuntuwin-Virtual: ~/rosProject
ubuntu-win@ubuntuwin-Virtual:~/rosProject$ rosrun proyecto1 poseUsr

Ingrese pose final:
Valor en X: 1
Valor en Y: 2
Angulo: 3.14
Ingrese tiempo de ejecucion (segundos): 8
[ INFO] [1472789695.613592539]: La pose inicial es = (8.97,8.95) direccion=1.57
[ INFO] [1472789695.913041504]: La pose actual es = (9.14,9.26) direccion=0.69 d
istancia al obj: 10.90
 •••
[ INFO] [1472789708.112972938]: La pose actual es = (1.00,1.99) direccion=2.96 d
istancia al obj: 0.01
[ INFO] [1472789708.413175502]: La pose actual es = (1.00,1.99) direccion=3.01 d
istancia al obj: 0.01
tiempo final: 7.40
tiempo final: 7.60
tiempo final: 7.60
tiempo final: 7.60
ubuntu-win@ubuntuwin-Virtual:~/rosProject$
```

El siguiente movimiento es hacia x: 2, y:9 con ángulo de 0 grados y en tiempo de 6 segundos, cabe mencionar que como esta construido el algoritmo de trayectoria hay una pequeña colisión con una frontera del mundo, sin embargo puede llegar a la pose indicada.

```
ubuntu-win@ubuntuwin-Virtual: ~/rosProject
tiempo final: 7.60
tiempo final: 7.60
ubuntu-win@ubuntuwin-Virtual:~/rosProject$ rosrun proyecto1 poseUsr

Ingrese pose final:
Valor en X: 2
Valor en Y: 9
Angulo: 0
Ingrese tiempo de ejecucion (segundos): 6
[ INFO] [1472789779.924383605]: La pose inicial es = (1.00,1.99) direccion=3.15
[ INFO] [1472789780.224448159]: La pose actual es = (0.63,2.08) direccion=2.72 d
istancia al obj: 7.05
[ INFO] [1472789780.524498224]: La pose actual es = (0.27,2.35) direccion=2.33 d
istancia al obj: 6.87
 •••
[ INFO] [1472789786.924779001]: La pose actual es = (1.98,9.04) direccion=3.35 d
istancia al obj: 0.04
[ INFO] [1472789787.324988668]: La pose actual es = (1.98,9.04) direccion=4.55 d
istancia al obj: 0.04
[ INFO] [1472789787.624576456]: La pose actual es = (1.98,9.04) direccion=6.17 d
istancia al obj: 0.04
tiempo final: 4.81
ubuntu-win@ubuntuwin-Virtual:~/rosProject$
```

Por último se inidca una pose en x:10, y:3 con ángulo nuevamente de 90 grados y con un tiempo de 9 segundos:





IV. CONCLUSIONES

Para desarrollar este proyecto se reforzaron conocimientos básicos de programación en C++ y se aprendieron nuevos objetos y comandos de ROS para poder comunicarse entre los componentes y lograr el objetivo del proyecto.

Las dificultad más sobresaliente resultó ser el manejo de los mensajes Twist para manipular al robot, sobretodo el cálculo del ángulo que éste debía tomar para dirigirse a la pose meta.

La combinación de objetos de publicación y subscripción en un sólo nodo al principio fue conceptualmente desafiante debido a que los comportamientos son asíncronos, sin embargo, la contribución y el diálogo entre los compañeros del curso mostró una forma sencilla de sincronizar ambos elementos.

La manera en la que se resolvió el proyecto no fue la óptima, ya que en clase se mostró que calculando la trayectoria previamente, se puede subdividir ésta en intervalos que pueden ser recorridos a distintas velocidades según se necesite. Debido a que en este caso la velocidad se calcula sólo en el inicio del algoritmo, el programa tiene problemas para ajustarse cuando la velocidad es muy pequeña y los desplazamientos producen una trayectoria errática que muy dificil termina en la pose deseada.

Cabe señalar que en los experimentos se observó reportado un tiempo igual o menor al indicado por el usuario. Una teoría sobre el porque sucedió así es debido a que todas las pruebas se realizaron sobre una máquina virtual de linux Ubuntu 14.04, sin embargo no pudieron hacerse pruebas concluyentes sobre lo reportado por la libreria time.h y el método clock().

V. REFERENCIAS

[1] Jason M O'Kane. *A gentle introduction to ROS*. 2014.