Práctica 3
Implementación de un nodo de ROS en la tarjeta arduino

Laboratorio de Bio-Robótica

Construcción de Robots Móviles

Objetivos

- Familiarizar al alumno con la tarjeta de desarrollo Arduino y sus herramientas de desarrollo.
- Implementar un nodo de ROS en la tarjeta Arduino usando la biblioteca `rosserial`.
- Practicar el uso de publicadores y suscriptores implementados en la tarjeta arduino.

1. Introducción

Arduino es una tarjeta electrónica de desarrollo *open-source* basada en la idea del software y hardware fáciles de usar. El hardware consiste en una tarjeta, comúnmente basada en microcontroladores Atmel AVR, con puertos analógicos y digitales de entrada/salida, además de varios módulos de comunicación y control comúnmnte usados en el desarrollo de robots móviles como generador de señales PWM, puerto RS232, comunicación I2C, entre otros.

El software Arduino consiste en un entorno de desarrollo integrado (IDE), que permite escribir programas y cargarlos en la tarjeta, y un lenguaje de programación (muy parecido al lenguaje C), específico para las tarjetas Arduino.

La biblioteca `rosserial` implementa un protocolo para empaquetar mensajes serializados estándares de ROS, además, permite multiplexar múltiples tópicos y servicios sobre dispositivos como puertos seriales o *sockets*. El paquete `rosserial_arduino` contiene extensiones específicas de Arduino para correr un cliente de la biblioteca `rosserial` en una tarjeta Arduino.

Mediante `rosserial_arduino` se puede implementar un nodo de ROS en la tarjeta Arduino Uno (que es la que se usará en este curso) que publique o se suscriba a tópicos y que atienda o llame a servicios, sin embargo, existen ciertas limitaciones. Dado que la tarjeta Arduino Uno posee sólo 2 kB de memoria RAM, no es posible enviar mensajes muy largos y el número de publicadores, suscriptores, clientes y servicios es muy limitado. Más adelante se dan instrucciones para no superar la memoria del Arduino Uno.

2. Desarrollo

2.1. Instalación del IDE de Arduino

Descargue el IDE de Arduino de la página `https://www.arduino.cc/en/Main/Software`. Seleccione la descarga de acuerdo con el sistema que tenga instalado, 32 o 64 bits.

Descomprima el archivo (se puede hacer mediante un click derecho y la opción *Extract here*), abra una terminal y cambie el directorio de trabajo a la carpeta que se acaba de extraer. Ejecute el archivo `install.sh`:

Suponiendo que el archivo descargado sea `arduino-1.8.1-linux64.tar.xz` y la carpeta de descargas esté en `~/Downloads`, los comandos para instalar el IDE serían:

```
$ cd ~/Downloads
$ tar xf arduino-1.8.1-linux64.tar.xz      #Descomprime el archivo
$ cd arduino-1.8.1
$ ./install.sh
```

Durante la instalación, seleccione siempre las opciones por default.

2.2. Instalación de la biblioteca `rosserial_arduino`

Nota. Se asume que el alumno ya tiene instalado Ubuntu 14.04 y ROS Indigo.

En una terminal, teclee los siguientes comandos:

```
$ sudo apt-get install ros-indigo-rosserial-arduino
$ sudo apt-get install ros-indigo-rosserial
```

Una vez instaladas, es necesario copiar estas bibliotecas al *sketchbook* del IDE de Arduino. La ubicación de esta carpeta se puede ver en el menú `File->Preferences`. Usualmente, el sketchbook está en `/home/user_name/Arduino`.

Una vez ubicada la ruta del sketchbook, teclee los comandos:

```
$ cd <sketchbook>/libraries
$ rm -rf ros_lib
$ rosrun rosserial_arduino make_libraries.py .
```

2.3. Código de ejemplo

Para verificar que las bibliotecas estén correctamente instaladas, escriba el siguiente código en el IDE de Arduino:

```
1  /*
2   * Ejemplo para publicar un entero que se incrementa en uno cada 500 ms
3   */
4  #include <ros.h>
5  #include <std_msgs/Int32.h>
6
7  ros::NodeHandle nh;
8
9  std_msgs::Int32 msgCounter;
10 ros::Publisher pubCounter("/counter", &msgCounter);
11
12 void setup()
13 {
14   nh.initNode();
15   nh.advertise(pubCounter);
16   msgCounter.data = 0;
17 }
18
19 void loop()
20 {
21   msgCounter.data++;
22   pubCounter.publish( &msgCounter );
23   nh.spinOnce();
24   delay(500);
25 }
```

Compile el código y cárguelo a la tarjeta. Asegúrese de seleccionar correctamente la tarjeta (Arduino Uno) en el menú `Tools->Board` y el puerto serial, en el menú `Tools->Serial Port`. El puerto al que está conectado el Arduino se puede saber mediante el comando `ls /dev`; éste desplegará una lista con los dispositivos conectados. El arduino suele tener el nombre `ttyACM0`, o bien, alguno similar.

En una terminal, ejecute el `roscore` y en otra, el siguiente comando (asegúrese de escribir el nombre del puerto correcto):

```
$ rosrun rosserial_python serial_node.py /dev/ttyACM0
```

En una tercera terminal, ejecute el comando

```
$ rostopic echo /counter
```

Si todo se ejecutó correctamente, la terminal debería mostrar una salida similar a esta:

```
data: 4943
---
data: 4944
---
data: 4945
---
data: 4946
---
data: 4947
---
```

2.4. Implementación de un subscriptor

Escriba un programa para implementar en el Arduino un nodo que se suscriba a un tópico de nombre `/period_ms`, de tipo entero, que determine el periodo de parpadeo, en milisegundos, del LED que ya está alambrado en la tarjeta. Por ejemplo, si se publica el tópico con un valor de 100, el LED debería parpadear a una frecuencia de 10 Hz.

En la página `http://wiki.ros.org/rosserial_arduino/Tutorials/Blink` se encuentra un ejemplo para implementar suscriptores. La opción `File->Examples->0.1 Basics->Blink` del IDE de Arduino despliega un ejemplo para hacer parpadear un LED.

3. Evaluación

- El programa a evaluar es el de la sección 2.4.
- Para probar el programa se publicará el tópico desde una terminal con el comando `rostopic pub`.
- El entero debe ser de cuando menos 16 bits para poder publicar periodos de varios segundos.
- El programa debe ser robusto ante periodos inválidos, por ejemplo, si se publica un dato negativo, la frencuencia debe permanecer con el valor anterior.
- Recuerde que la práctica sólo se considera entregada si todo funciona correctamente.