Python and robotics: a quick introduction to rospy

Juampe López - Python Barcelona Meetup

Robotics: state of the art

- Robotics is a hot topic - it doesn’t matter when you read this!
- IoT, Industry 4.0, AI… everybody is talking about robots and how they are gonna replace us soon - after killing us, of course.

Robotics: state of the art

Elon Musk leads 116 experts calling for outright ban of killer robots

Open letter signed by Tesla chief and Alphabet's Mustafa Suleyman urges UN to block use of lethal autonomous weapons to prevent third age of war

A killer robot from the 2014 remake of Robocop. The open letter read: 'lethal autonomous weapons will permit armed conflict to be fought at a scale greater than ever, and at timescales faster than humans can comprehend.' Photograph: Allstar/Studio Canal/Sportsphoto Ltd./Allstar

Robotics: state of the art

Robotics: state of the art

Robotics: the actual state of the art

ROS (Robot Operating System)

So what is ROS?

- It's an **open-source**, meta-operating system for your robot
- It provides **hardware abstraction** and **low-level device control**
- It's based on **message-passing between processes**
- It provides **package management**
- It allows **writing and running code across multiple computers**.

ROS (Robot Operating System)

And what about rospy?

- It's a pure **Python client library for ROS**
- It enables Python developers to **quickly interface with ROS Topics, Services, and Parameters**
- it **favors implementation speed (i.e. developer time) over runtime performance**
- actually **many of the ROS tools are written in rospy**

ROS (Robot Operating System)

Some keywords before we see the code:

- Node: *ROS is designed to be modular at a fine-grained scale: a system is typically comprised of many nodes. In this context, the term “node” is interchangeable with “software module”.* *(ROS: an open-source Robot Operating System, ICRAOSS'09)*

ROS (Robot Operating System)

Some keywords before we see the code:

- Message: *Nodes communicate with each other by passing messages. A message is a strictly typed data structure. Standard primitive types (integer, floating point, boolean, etc.) are supported, as are arrays of primitive types and constants. Messages can be composed of other messages, and arrays of other messages, nested arbitrarily deep.* *(ROS: an open-source Robot Operating System, ICRAOSS'09)*

ROS (Robot Operating System)

Some keywords before we see the code:

- Topic: *A node sends a message by publishing it to a given topic, which is simply a string such as “odometry” or “map.” A node that is interested in a certain kind of data will subscribe to the appropriate topic. There may be multiple concurrent publishers and subscribers for a single topic, and a single node may publish and/or subscribe to multiple topics.* *(ROS: an open-source Robot Operating System, ICRAOSS'09)*

ROS and rospy: publisher node example

```python
#!/usr/bin/env python
import rospy
from std_msgs.msg import String

def talker():
    pub = rospy.Publisher('chatter', String, queue_size=10)
    rospy.init_node('talker', anonymous=True)
    rate = rospy.Rate(1) # 1hz
    while not rospy.is_shutdown():
        hello_str = 'Hello, PyBCN! %s' % rospy.get_time()
        rospy.loginfo(hello_str)
        pub.publish(hello_str)
        rate.sleep()

if __name__ == '__main__':
    try:
        talker()
    except rospy.ROSInterruptException:
        pass
```

ROS and rospy: subscriber node example

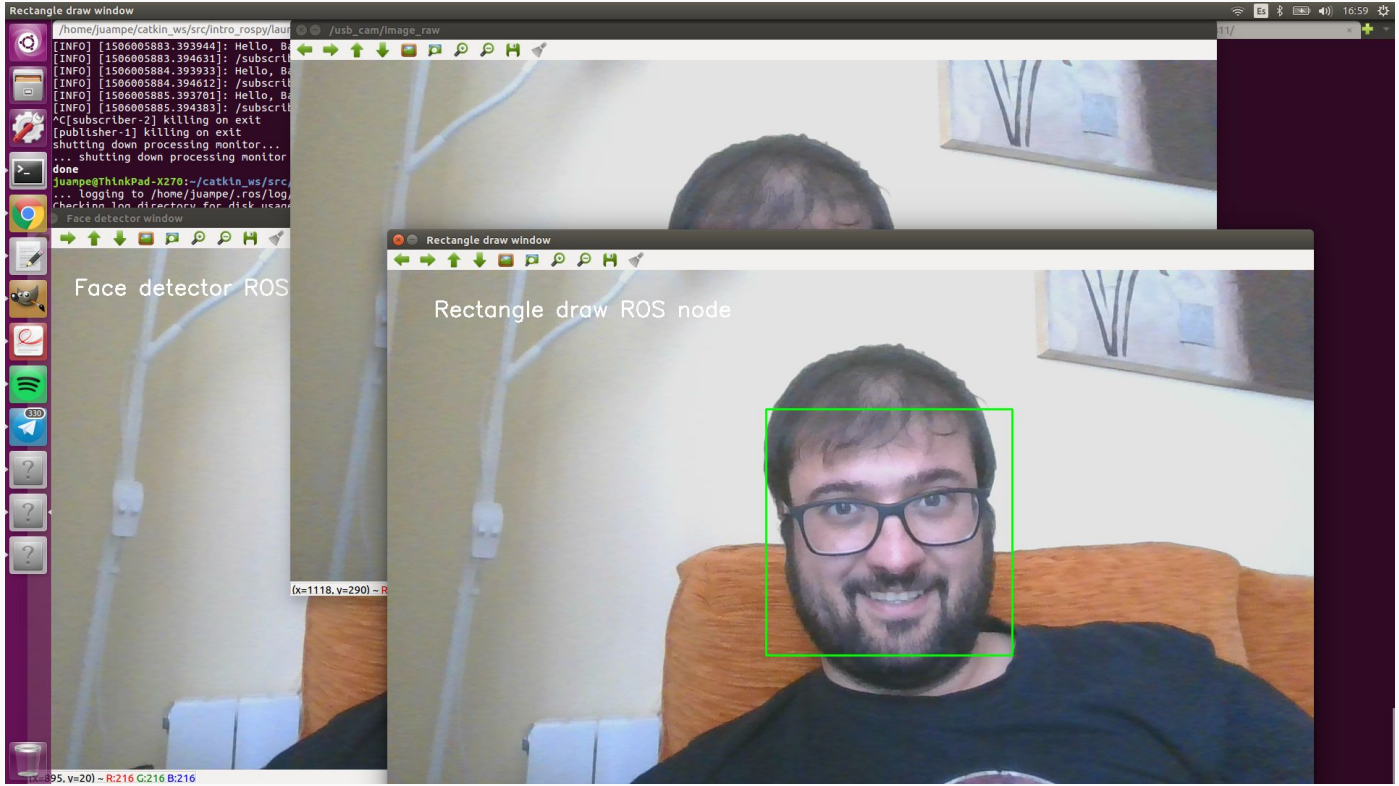
```
#!/usr/bin/env python
import rospy
from std_msgs.msg import String

def callback(data):
    rospy.loginfo(rospy.get_caller_id() + 'What I heard: %s', data.data)

def listener():
    rospy.init_node('listener', anonymous=True)
    rospy.Subscriber('chatter', String, callback)
    rospy.spin()

if __name__ == '__main__':
    listener()
```

ROS and rospy: USB camera and OpenCV demo



Links

- https://player.vimeo.com/video/146183080
- http://www.willowgarage.com/sites/default/files/icraoss09-ROS.pdf
- http://wiki.ros.org/rospy
- http://wiki.ros.org/kinetic/Installation/Ubuntu
- https://github.com/simium/intro_rospy

The end

Thank you for attending!

Questions?

Juampe López - Embedded Linux software engineer

https://github.com/simium - https://es.linkedin.com/in/jplopezcabrera