MESSAGE FILTERS

ROBOTICS

POLITECNICO
MILANO 1863

MESSAGE FILTERS

Useful to synchronize multiple topics

Need topics with header and timestamp

Can synchronize with exact time or approximate time

Camera topics synchronization has a custom version

MESSAGE FILTERS (without policy)

```
message_filters::Subscriber<geometry_msgs::Vector3Stamped> sub1(n, "topic1",
1);
  message_filters::Subscriber<geometry_msgs::Vector3Stamped> sub2(n, "topic2",
1);
  message_filters::TimeSynchronizer<geometry_msgs::Vector3Stamped,
geometry_msgs::Vector3Stamped> sync(sub1, sub2, 10);
  sync.registerCallback(boost::bind(&callback, _1, _2));
```

Create the subscriber

Create the time synchronizer

Bind it with the callback

MESSAGE FILTERS (with policy)

```
typedef
message_filters::sync_policies::ExactTime<geometry_msgs::Vector3Stamped,
geometry_msgs::Vector3Stamped> MySyncPolicy;
```

Create the policy

```
message_filters::Synchronizer<MySyncPolicy> sync(MySyncPolicy(10), sub1, sub2);
  sync.registerCallback(boost::bind(&callback, _1, _2));
```

Bind it with the callback

Create the time synchronizer with the policy

ROSPY

ROBOTICS

POLITECNICO
MILANO 1863

ROSPY

Python 2.7

With some changes ROS works also with 3.6

Files saved in the script folder

Start node with same syntax as c++ node but using the name of the executable instead of the node:

rosrun package_name python_file.py

ROSPY (publisher)

```
import rospy
from std_msgs.msg import String
```

← Standard rospy include and std_msgs include

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

← Main function

← Start the tolker

ROSPY (publisher)

```
def talker():
    pub = rospy.Publisher('chatter', String, queue_size=10)
    rospy.init_node('talker', anonymous=True)
    rate = rospy.Rate(10)
    while not rospy.is_shutdown():
        hello_str = "hello world %s" % rospy.get_time()
        rospy.loginfo(hello_str)
        pub.publish(hello_str)
        rate.sleep()
```

Create the publisher

Initialize the node

Set the loop rate

Keep spinning

Rospy version of ROS_INFO

Publish the message

ROSPY (subscriber)

```
import rospy
from std_msgs.msg import String

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

Standard include

Main function

ROSPY (subscriber)

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

def listener():


    rospy.init_node('listener', anonymous=True)

    rospy.Subscriber("chatter", String, callback)

    rospy.spin()
```

Callback function

Init the node

Create the subscriber

ROSPY (bag operation)

```
import rosbag
bag = rosbag.Bag('test.bag')
for topic, msg, t in bag.read_messages(topics=['chatter', 'numbers']):
    print msg
bag.close()
```

import lib to handle bag files

Import the bag file

Filter specific topic

Cycle through the topic and messages