Experiment 1

Write a simple publisher and subscriber

Objective

- To create a package in the ROS workspace
- Write Python codes for publishing and subscribing to a topic
- Execution of master node and the two codes to publish and subscribe to a topic

Theory

Robot Operating System (ROS) is an open platform meta operating system to integrate various software capabilities of a robot. Internally, a robot consists of a processor with algorithms for navigation, multiple sensor modules and their codes, controllers to regulate various actuators, etc. ROS coordinates the communication between various parts. It considers each part of the robot as ‘nodes’ messages are communicated between the nodes using the term ‘topic.’ The codes of these are stored in a folder called ‘packages’ inside the workspace, which is the working environment of ROS. A ‘hello world’ example shows how messages are communicated between nodes. First, the steps to create a package are explained.

Procedure

1. Go to the workspace of the folder, usually catkin_ws, and in src, use the following code
   $ catkin_create_pkg ros_package_name package_dependencies
2. Build the package in the workspace using the command
   $catkin_make
3. Inside the newly created package create a folder and open a text editor using the command
   $gedit talker.py

   And copy the following code

```
 #!/usr/bin/env python3


import rospy

from std_msgs.msg import String


def publishMethod():

   pub = rospy.Publisher('talker', String, queue_size=10) # defining the publisher by topic, message
type

   rospy.init_node('publish_node', anonymous=True) # defining the ros node - publish node

   rate = rospy.Rate(10) # 10hz # fequency at which the publishing occurs

   while not rospy.is_shutdown():

      publishString = "Hello world" # varibale

      rospy.loginfo("Data is being sent")  # to print on the terminal

      pub.publish(publishString) # publishing

      rate.sleep()
```

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

4. Open another text editor for receiving using the command
   $gedit receiver.py

5. Copy the following code to the receiver.py

```
#!/usr/bin/env python3


import rospy
from std_msgs.msg import String


def subscriberCallBack(data):
    rospy.loginfo(rospy.get_caller_id() + " I recieved --  %s", data.data) #prints on terminal


def listener():
    rospy.init_node('subscriberNode', anonymous=True)
    rospy.Subscriber("talker", String, subscriberCallBack)
    rospy.spin() # the python file does not exit
if __name__ == '__main__':
    listener()
```

6. Go to the ROS workspace and open the master file
   $roscore
7. In new terminal, run the publishing node using the command
   $rosrun package_name talker.py
8. In another terminal, run the receiver node using the command
   $rosrun package_name receiver.py

   The message “Hello world” will get published in the current terminal