I have submitted the quat\_to\_euler package as a part of this assignment.

In the scripts folder of the package, you will be able to find quat\_to\_euler.py file. It is essentially a ROS Subscriber Node. Let us understand what does the code written in this file mean and how it works.

* The first line “#!/usr/bin/env python” signifies that it is python script and ROS architecture (framework) interprets it as a python file.
* Next three lines simply imports rospy and python packaged like math and numpy.
* The following line “from geometry\_msgs.msg import Quaternion”, is used to import the Quaternion Message Type form the geometry\_msgs.
* As seen, this file consist of two functions - callback() and quat\_to\_euler().
* The “quat\_to\_euler()” function is used to initialize the node. The rospy.init\_node() function initializes the quat\_to\_euler node. The anonymous = True tag is used to inform rospy to generate a unique name for the node so that multiple quat\_to\_euler nodes can exist together.” (This is necessary as ROS requires that each node must have a unique name and if same name appears it shuts the older node.)
* The next line in the “quat\_to\_euler()” function “rospy.Subscriber (“QUATERNION\_TO\_EULER”, Quaternion, callback) states that this node subscribes the QUATERNION\_TO\_EULER topic which uses message of the type geometry\_msgs.msg.Quaternion.
* Whenever a new message comes, the callback() function is invoked and the message is used as the first argument.
* rospy.spin() basically keeps the node running until it is shutdown.
* Now in the callback() function, geometry\_msgs.msg.Quaternion (basically the incoming from, or published (by some other node) message on, the topic (QUATERNION\_TO\_EULER)) is used as an argument. Here I have labelled it as Quaternion, but it could have been named anything.
* The next four lines shows how I extract information (different components of the message) form the incoming message by doing Quaternion.w, Quaternion.x etc.
* Now the next few lines use this information and computes the Euler angles, “Phi”, “Theta” and “Psi” by using the provided mathematical formulation.
* Rospy.loginfo() is used to display the output on the terminal.

To Run these files we have to follow essentially do three steps –

1. roscore
2. In new terminal launch the quat\_to\_euler node by typing “rosrun quat\_to\_euler quat\_to\_euler.py” in the terminal
3. In new terminal now publishing on the QUATERNION\_TO\_EULER topic directly from the terminal window by typing “rostopic pub QUATERNION\_TO\_EULER geometry\_msgs.msg/Quaternion x:## y:## z:## w:##”

(Where ## is a number between -1 and 1)

For multiple set of quaternion pairs (as seen in the following Fig.1) we get corresponding Euler angles (ZYX) as seen in the Fig.2. \*\*The output (PHI, THETA & PSI) is expressed in degrees.

Graphical user interface, text

Description automatically generatedText

Description automatically generated

Fig.1 Fig.2