ROS In-class Progress Check Correction

By Victor Chen

1. What are ROS Nodes?

   ROS nodes are the basic computational processes that are capable of performing certain calculations and functionalities and communicating through the ROS topics or services.

2. What are ROS Topics?

   ROS topics is a method to support the communication between ROS nodes. The ROS topics allow multiple ROS nodes to publish data on them and send the data to the receivers.

3. What are ROS Workspaces?

   ROS workspaces are the directories that contain ROS packages. Each package consists of one or multiple nodes that are for performing the tasks on a certain functionality, and the workspace is just a combination of all these packages.

4. What are ROS services?

   ROS services are also a method to support the communication for the ROS node. It allows nodes to create multiple clients and request for calculations or actions on the server, and often with data callback received.

5. When would you use a ROS service vs a ROS topic?

   ROS topic is mainly used if you’re aiming for communication between ROS nodes, such as when passing information between nodes. ROS Service, on the other hand, is mainly for functionalities that the node wants to perform. For example, a node can send a request to the ROS server in order to perform specific actions.

6. What do you have to do every time you open a new terminal in order to use ROS? Hint: it will produce the error ROS2 command not found. Bonus: How do you get around this?

   You have to use the following line to first source the ROS2:

   ```
   > source /opt/ros/humble/setup.bash
   ```

   And then we need the following line to activate the workspace:

   ```
   > source ~/<workspaceName>/install/setup.bash
   ```

7. I have just completed writing a node in my_robot_controller in a workspace called ros2_ws what does my path look like?

   Begins from the workspace, the path should look like this:
   ros2_ws/src/my_robot_controller/my_robot_controller/<nodeFile>

8. If I were to have successfully established a ROS environment and run both talker and listener what would running rqt_graph produce?

   The rqt_graph should produce two nodes, the first node talker (as a publisher) is directed to a topic for communication; and the topic will direct to the second node listener (as a subscriber).

9. When you create a new node what do you need to do in order to run a new node called tester found in the node_tester package and where would you run it?

   Run the following lines in the /src/node_tester/node_tester/:

   ```
   > touch tester.py
   > chmod +x tester.py
   ```

   Then in /src/node_tester/setup.py, add the following line:
   “tester = node_tester.tester:main”

10. What do you need to source to run custom nodes?

You need to source the setup.bash of the workspace, which is:

```
> source /install/setup.bash
```

11. If I have created a node called test_node in my_robot_controller and would like to execute it through the command line how would I make it executable from the command line with the ros2 functionalities? Name it tester. Hint: you must add something to a .py file.

I will add the following line to my setup.py in entry_points:
“test_node = my_robot_controller.tester:main”

I’m assuming test_node is the node name and tester is the file name.

12. What packages do you need to import for every node?

rclpy

13. What are the arguments for ros publisher and subscriber?

For creating ROS publisher:

- The topic it’s publishing to;
- The data type that it’s passing;
- Amount of data stored historically when connection break;

For creating ROS subscriber:

- The topic it’s subscribing to;
- The data type that topic is passing;
- The callback function to be executed every time when new data is received;
- Amount of data received historically when connection break;

14. What does ros spin do and why do you need it?

ROS spin (rclpy.spin(node)) keeps the node active and able to receive callbacks even after the program finishes execution, and it keeps the terminal alive until being stopped by Ctrl+C.

15. What is a call back?

Call back is a function that is executed whenever the subscriber receives new information from the topic it’s subscribing to.

16. How do I see the ROS Topics running?

Run the following command in the terminal:

```
> ros2 topic list
```

17. I noticed there is a topic called geometry message. How can I see what information is on that topic?

Run the following command in the terminal:

```
> ros2 topic echo <topicName>
```

18. Once I know the name of a topic how do I know the message type of it?

Run the following command in the terminal:

```
> ros2 topic info <topicName>
```

19. What is the first thing you should do if you run into an error?

The first thing I will do is to review the error message, and try to figure out the potential quick fixes. If these quick fixes are not available or not working, likely I will be looking them up on Google or StackOverflow. If I still cannot figure out the solution, I will be asking my teammates, TAs, and instructors.

20. In setup.py I add the line “test_node = my_robot_controller.my_first_node:main” what is the executable name, what is the package name, and what is my node name?

    Executable name: my_first_node.py
    Package name: my_robot_controller
    Node name: test_node

21. How do you edit a python file in the terminal?

    Run the following command in terminal (requires vim):

    ```
    > vim <fileName>
    ```

22. What does chmod +x do?

    This command makes the file listed able to be executed.

23. What is a src folder and why is it necessary?

    src folder is the folder that stores all the codes, and it’s necessary because it separates the part where we can edit our code with the functional part used by the system (such as install).

24. How do you create a ros package?

    Run the following command in terminal:

    ```
    > ros2 pkg create <packageName>
    ```

25. Why should you include --symlink in colcon build?

    Because with this option, the updated code in the nodes will be automatically integrated when running, so you don’t have to run colcon build every time after you make an update. This makes debugging these nodes much easier.

Bonus: Create a Ros node that will log : “ UAV is Awesome”?