Laboratory Report 2

Azamat Turganbayev

1. Introduction

Robot Operating System (ROS) is an open-source set of software frameworks for the software development of robots. It gives administrations planned for a heterogeneous computer cluster such as equipment reflection, low-level device control, execution of commonly used functionality, message-passing between processes, and package management. Running sets of ROS-based forms are spoken to in a graph architecture where preparation takes place in nodes which will get, post, and multiplex sensor information, control, state, planning, actuator, and other messages. In spite of the significance of reactivity and low latency in robot control, ROS isn't a real-time working framework (RTOS). In any case, it is conceivable to coordinate ROS with real-time code. The need of support for real-time frameworks has been tended to within the creation of ROS 2, a major modification of the ROS API which can take advantage of present day libraries and advances for center ROS capacities and include back for real-time code and inserted framework equipment.

2. Application of ROS in robots

As the robotics field is developing with tremendous speed, many engineers and manufacturers apply ROS in modeling the robots. Below you can find the examples of such robots.

A. Clearpath Robotics

Clearpath is a Canadian company founded in 2009. The main application can be founded in unmanned ground vehicles, unmanned surface vehicles (on the water), and industrial vehicles field. Most robots are based on ROS and can be programmed easily. Below is the JACKAL UGV.

Figure 1. Jackal UGV from Clearpath Robotics

B. Pal Robotics

Pal Robotics was founded in Barcelona in 2004. This company focuses on humanoid robots. In addition, they also manufacture mobile manipulators. This is the only company that can afford producing and selling these types of robots.

Figure 2. TALOS Robot from Pal Robotics

3. Publisher - Subscriber

As a main exercise of Lab 2, the main idea was to write a code for a simple Publisher - Subscriber. The aim was to make a continuous loop with the student id number and a topic name with the student's surname. 2 options for completing this task were available: via C++ or Python. This task was done through Python. Below you can see the images of **output terminal** and codes for **talker.py** and **listener.py**.

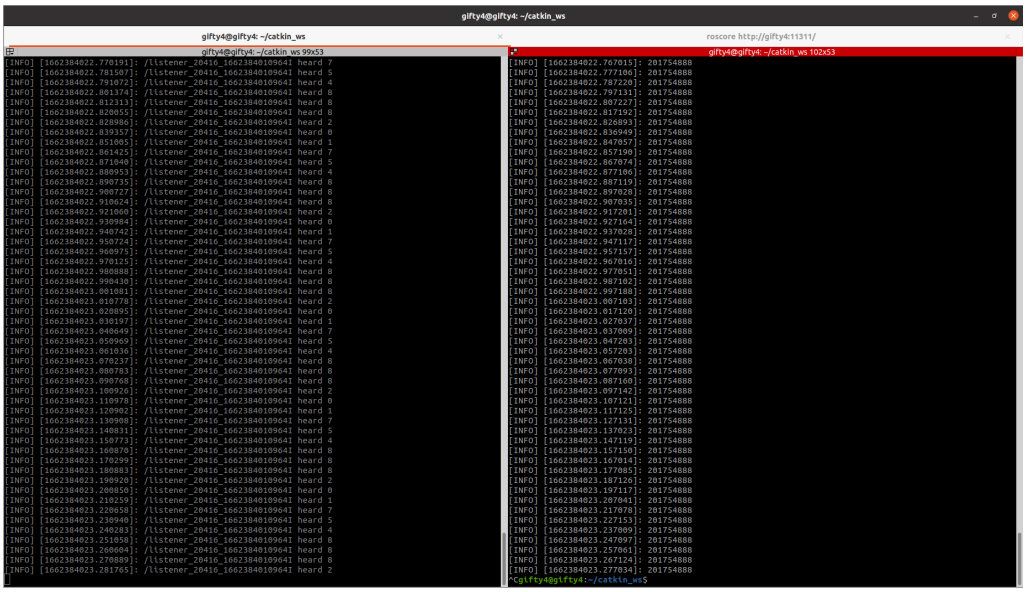

Figure 3. Output in Terminal

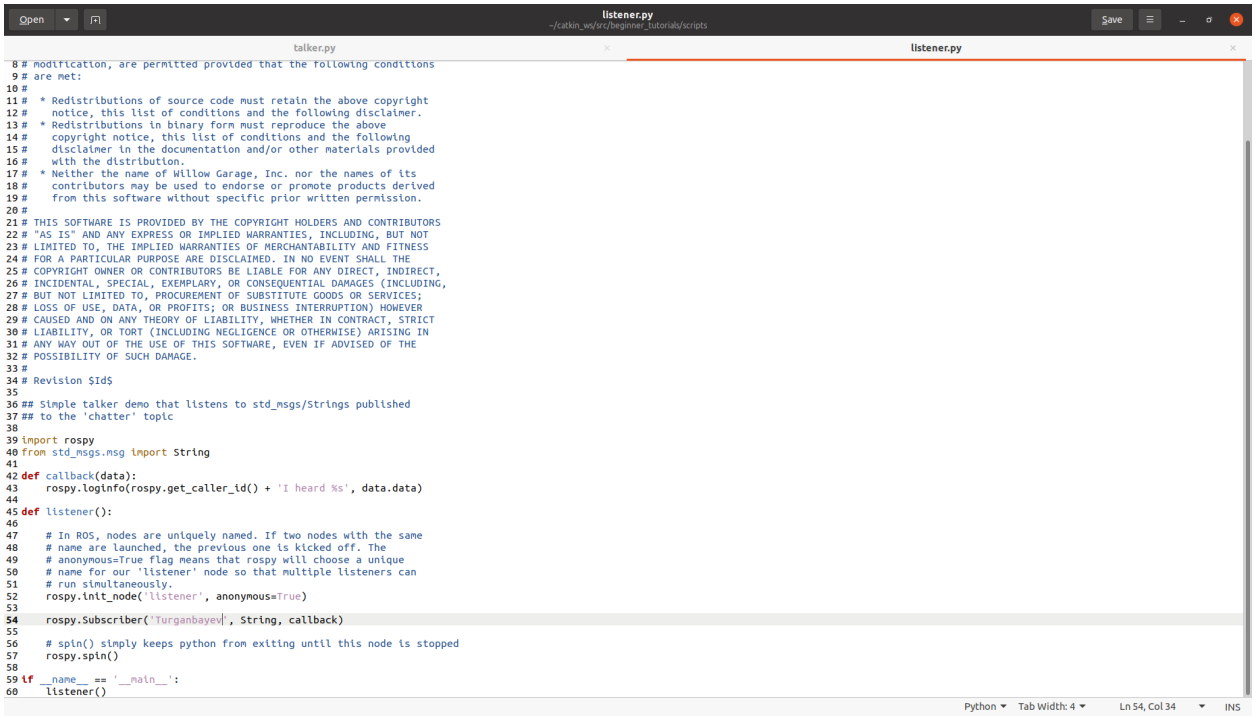

Figure 4. Listener.py code

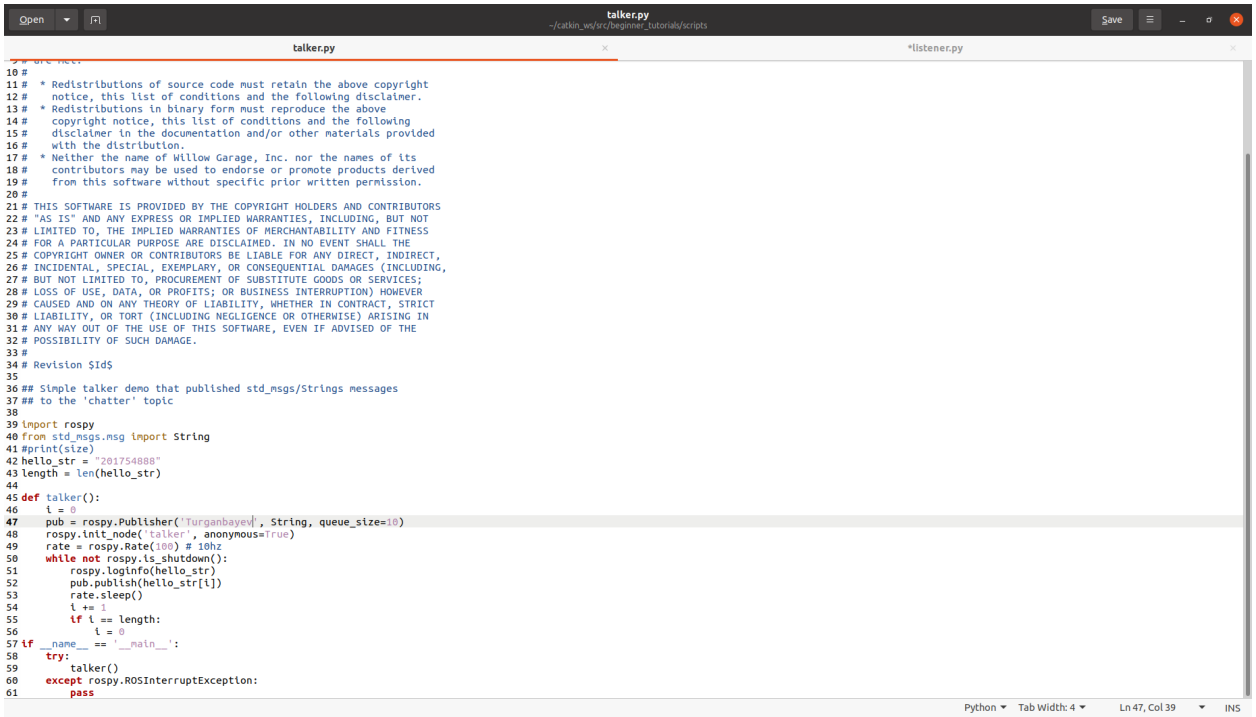

Figure 5. Talker.py code