TÉCNICO LISBOA

Robotics Course 2021/2022

Instituto Superior Técnico

LAB 1 - Software Installation Guide

November 2021

1 Introduction

For LAB 1, students will perform tests with the Niryo One requires students to connect through ssh to the real robot. However, there are only a few robots on each lab session, thus below it is also shown how to install a simulator for Niryo One, on *Ubuntu 16.04*. Students can validate their model from home using the simulator and then perform tests using the real Niryo One.

Figure 1: Niryo One Robot Arm

2 Niryo One Simulator

The Niryo One Simulator will be used as a prior validation by the students of their tested model in the Niryo ROS Simulation Environment. From here, students can interactively choose joint angles to find the positions or the other way around. However, this tool requires two main components:

- **Ubuntu 16.04** - It is recommended that students use this version, since it is the only one officially supported by Niryo.
- **ROS Kinetic** - There are many different ROS versions, but it is required to install ROS Kinetic on Ubuntu for this simulator.

2.1 Ubuntu 16.04

For Windows and Mac users, students can use a the software provided by IST and install *VMWare*(instructions for installation are also in the link) to create a Virtual Machine for the *.iso* file of *Ubuntu 16.04*. From here, it is only needed to proceed with the normal installation of Ubuntu.

2.2 ROS Kinetic

Once installed *Ubuntu 16.04*, it is now recommended to install ROS Kinetic, which is a version of ROS, a widely used open-source robotics Middleware for development and simulations. The download and instructions for the installation can be found here - *ROS Kinetic*.

2.3 Niryo One Simulation Environment

Now that *ROS Kinetic* is installed, it is now possible to install the ROS Niryo One Simulation Envirionment, where students can experiment with different settings on the robot arm in real-time. The Download and installation instructions can be found here - *ROS Niryo One Simulator*.

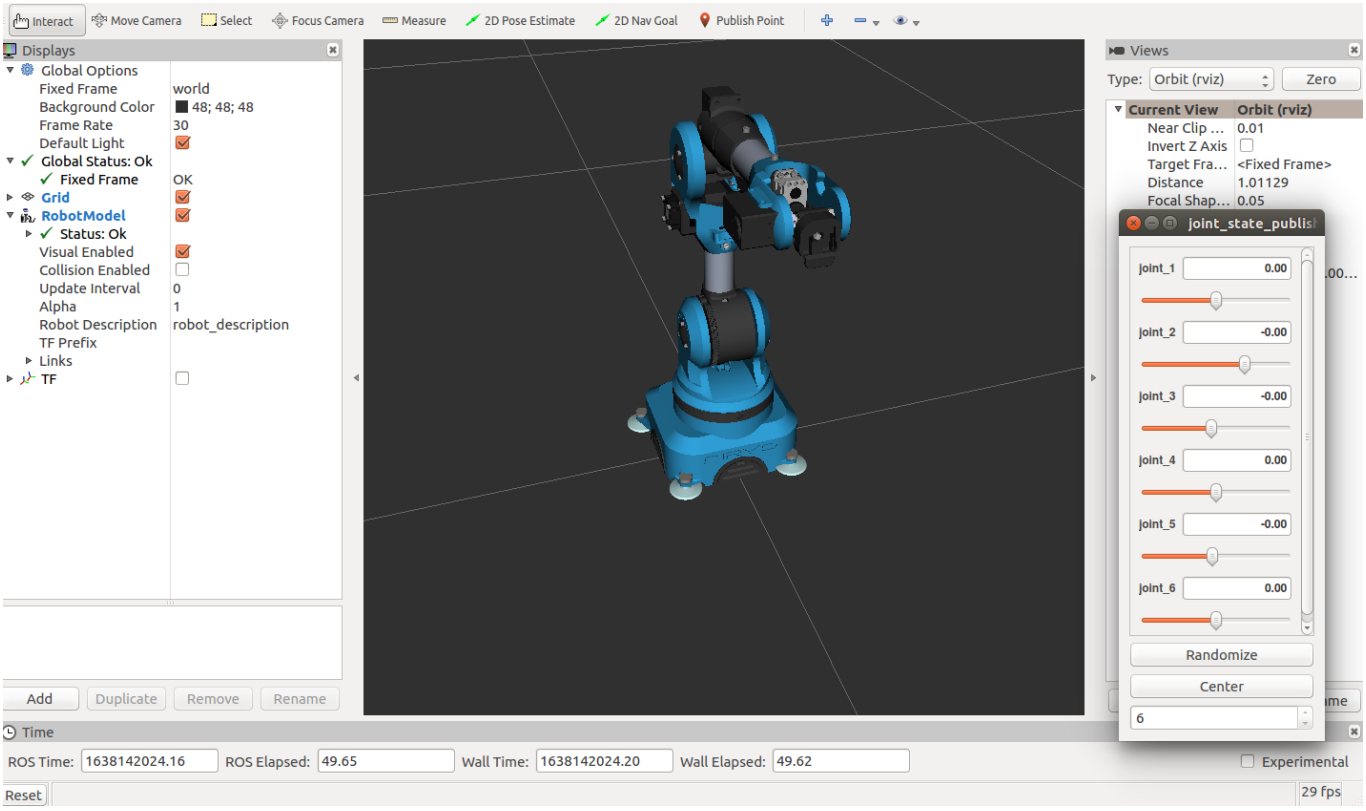

Figure 2: Niryo One Simulator in RVIZ

From Figure 2, it shown the simulator environment on RVIZ when launching the main testing environment by typing **"roslaunch niryo_one_description display.launch"** in the Ubuntu terminal. Not only will RVIZ open but a set o scroll-bars to control the robot joint angles. This simulator is based on ROS, thus it is advised that students perform some level of research on this subject, more specifically on the topic of ROS Nodes, where students can use them to find specific information regarding the robot simulations, like the Pose on the final point with a command similar to - *rostopic echo /topicname*. More information regarding these topics can be found here- *Niryo one ROS Stack*, *ROS General Info* and *rostopics*.

2.4 Niryo One Simulator w/Python API

If students want to run their code with the Niryo One Simulator, the following steps have to be taken previously.

1. Proceed to the "src" folder inside "catkin_ws" folder where Niryo is installed.
2. Open to the *niryo_one_python_api* folder and go to the examples folder.
3. Now its needed to create a ROS Package. Open the terminal here and type - **catkin_create_pkg NiryoPy std_msgs rospy roscpp**.
4. Once the package is created, type **"cd ~/catkin_ws"** and then **catkin_make**.
5. Add the package to the ROS Environment - **". ~/catkin_ws/devel/setup.bash"**.

6. Go to the package created, inside the "examples" folder and create a scripts section - **mkdir scripts**. Here copy the code provided to a new file .py inside it. (it is recommended that students install previously VS Code to simplify code editing).

7. Now open a terminal window in this scripts folder a type "ls", and see if the python file name is green. If not, it means that it is not executable and it is necessary to type **chmod +x <name_of_python_file>**.

8. Now its all set and students should proceed this next procedure whenever they want to perform tests. First open a new terminal a open the ROS simulator compatible with the Python API - **"roslaunch niryo_one_bringup desktop_rviz_simulation.launch"** and then on the previous terminal inside the scripts file type - **"rosrun niryo_one_python_api <name_of_python_file"**.

2.5 Common Installation Errors

- **ROS Kinetic** - When Students first try to install ROS Kinetic, in some cases a error shows that *Unable to locate package ros-kinetic-desktop-full*, which can be solved by updationg the system after setting up the keys and sources to download as seen here - *Unable to locate package ros-kinetic-desktop-full*.

- **Niryo One Simulator** - When performing the *roslaunch* for the Nyrio One Simulator, an error may appear stating that *Missing joint_state_publisher_gui*, to solve this, it is just require to install this package as seen here - *Missing joint_state_publisher_gui*.

3 Python Code - Niryo Linux Simulator

Example code to move Niryo from his base position

```python
#!/usr/bin/env python
import sys
import rospy
import time
sys.path.insert(1,'/path/to/folder/niryo_one_python_api')

from niryo_one_api import *

rospy.init_node('niryo_one_example_python_api')

print("--- Start")

n = NiryoOne()
n.calibrate_auto()
c = 0
try:
    print('test counting'.format(c))
    c = c + 1
    joints = n.get_joints()
    print(joints)
    joints = [0.1, 0.1, 0.1, 0.1, 0.1, 0.1]
    n.move_joints(joints)
except NiryoOneException as e:
    print(e)
```

4 Rostopics List for the full behaviour of Niryo

Once the simulator is launched, students should perform in a new terminal the command - **rostopic list** - and see if all the topics correspond to the below list, to understand if all nodes are active. If not, an error might show on the terminal window when launching the simulator, with a warning such as *"No module found"*. From here students should perform a simple command - "pip install <package>". This only works if previously students have installed *pip*.

```
/attached_collision_object
/clicked_point
/client_count
/collision_object
/connected_clients
/diagnostics
/execute_trajectory/cancel
/execute_trajectory/feedback
/execute_trajectory/goal
/execute_trajectory/result
/execute_trajectory/status
/initialpose
/joint_states
/joy
/joy/set_feedback
/move_base_simple/goal
/move_group/cancel
/move_group/display_contacts
/move_group/display_planned_path
/move_group/feedback
/move_group/goal
/move_group/monitored_planning_scene
/move_group/ompl/parameter_descriptions
/move_group/ompl/parameter_updates
/move_group/plan_execution/parameter_descriptions
/move_group/plan_execution/parameter_updates
/move_group/planning_scene_monitor/parameter_descriptions
/move_group/planning_scene_monitor/parameter_updates
```

```
/move_group/result
/move_group/sense_for_plan/parameter_descriptions
/move_group/sense_for_plan/parameter_updates
/move_group/status
/move_group/trajectory_execution/parameter_descriptions
/move_group/trajectory_execution/parameter_updates
/niryo_one/blockly/break_point
/niryo_one/blockly/highlight_block
/niryo_one/blockly/save_current_point
/niryo_one/commander/robot_action/cancel
/niryo_one/commander/robot_action/feedback
/niryo_one/commander/robot_action/goal
/niryo_one/commander/robot_action/result
/niryo_one/commander/robot_action/status
/niryo_one/current_tool_id
/niryo_one/hardware_status
/niryo_one/joystick_interface/is_enabled
/niryo_one/kits/conveyor_1_feedback
/niryo_one/kits/conveyor_2_feedback
/niryo_one/learning_mode
/niryo_one/max_velocity_scaling_factor
/niryo_one/robot_state
/niryo_one/rpi/digital_io_state
/niryo_one/sequences/execute/cancel
/niryo_one/sequences/execute/feedback
/niryo_one/sequences/execute/goal
/niryo_one/sequences/execute/result
/niryo_one/sequences/execute/status
/niryo_one/sequences/sequence_autorun_status
/niryo_one/software_version
/niryo_one/steppers_reset_controller
/niryo_one/tool_action/cancel
/niryo_one/tool_action/feedback
/niryo_one/tool_action/goal
/niryo_one/tool_action/result
/niryo_one/tool_action/status
/niryo_one_follow_joint_trajectory_controller/command
/niryo_one_follow_joint_trajectory_controller/follow_joint_trajectory/cancel
/niryo_one_follow_joint_trajectory_controller/follow_joint_trajectory/feedback
/niryo_one_follow_joint_trajectory_controller/follow_joint_trajectory/goal
/niryo_one_follow_joint_trajectory_controller/follow_joint_trajectory/result
/niryo_one_follow_joint_trajectory_controller/follow_joint_trajectory/status
/niryo_one_follow_joint_trajectory_controller/state
/niryo_one_matlab/command
/niryo_one_matlab/result
/niryo_one_vision/compressed_video_stream
/pickup/cancel
/pickup/feedback
/pickup/goal
/pickup/result
/pickup/status
/place/cancel
/place/feedback
/place/goal
/place/result
/place/status
/planning_scene
/planning_scene_world
/rosout
/rosout_agg
/tf
/tf2_web_republisher/cancel
/tf2_web_republisher/feedback
/tf2_web_republisher/goal
/tf2_web_republisher/result
/tf2_web_republisher/status
/tf_static
/trajectory_execution_event
```