



МИНОБРНАУКИ РОССИИ

**Федеральное государственное бюджетное образовательное учреждение
высшего образования**

“МИРЭА - Российский технологический университет”

РТУ МИРЭА

Институт кибернетики

Кафедра проблем управления

ЛАБОРАТОРНАЯ РАБОТА №4

по дисциплине

Программное обеспечение мехатронных и робототехнических систем

Тема работы: “Осуществление публикации и чтения данных внутри
информационной среды ROS”

Студент группы КРБО-03-17

Орлов Д.Л., 

Преподаватель:

Морозов А.А. _____

ЦЕЛЬ И ЗАДАНИЕ

Цель работы: Получение навыков установки и настройки ROS (Robot operating system), ознакомление с архитектурой ROS, создание узла, осуществляющего публикацию данных и чтение данных из информационной среды ROS. **Задание:** Установить и настроить ROS. Создать узлы Publisher и Subscriber для публикации и чтения данных из информационной среды ROS.

ХОД РАБОТЫ

Был выбран дистрибутив ROS Melodic. Для начала настроим систему, для того что бы мы могли принимать программное обеспечение, выполнив команду:

```
sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu
$(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-
latest.list'
```

Настроим ключи доступа в системе для правильной загрузки:

```
sudo apt-key adv --keyserver hkp://ha.pool.sks-keyservers.net:80 -
-recv-key C1CF6E31E6BADE8868B172B4F42ED6FBAB17C654
```

Убедимся в том, что мы имеем последние версии индексов пакетов:

```
sudo apt-get update
```

Теперь установим сам пакет ROS. Так как мы решили использовать ROS вместе с симуляцией, установим полную версию.

```
sudo apt-get install ros-melodic-desktop-full
```

Далее необходимо инициализировать rosdep:

```
sudo rosdep init rosdep  
update
```

Настроим переменное окружение так, чтобы оно добавлялось в наш сеанс bash при каждом запуске новой оболочки:

```
echo "source /opt/ros/melodic/setup.bash" >> ~/.bashrc source  
~/.bashrc
```

Далее при помощи команды запустим ROS и добавим узел для симуляции черепашки, таким образом убедимся в правильности установки (рисунок 1).

```
roscore  
roslaunch turtlesim turtlesim_node
```

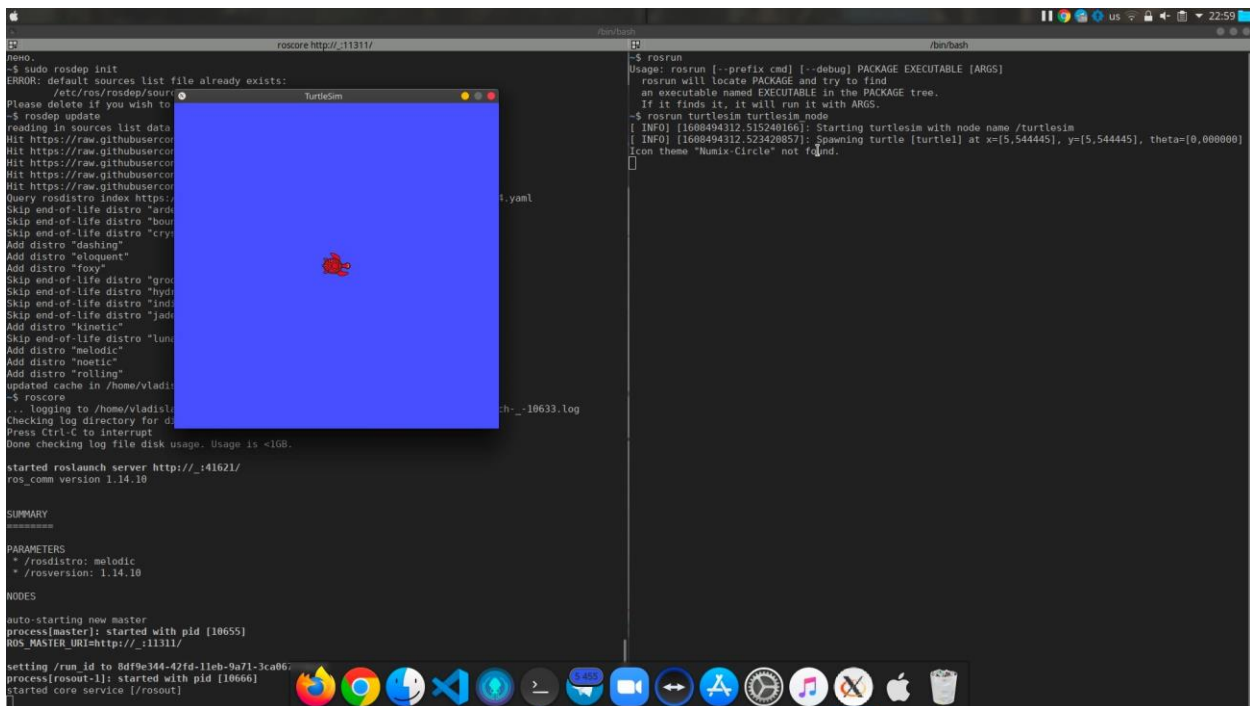


Рисунок 1 - Запуск ROS и turtlesim

Теперь необходимо создать рабочее пространство для создания своего пакета ROS.

Создадим папку рабочего пространства.

```
mkdir -p ~/ros_workspace/src
```

Проинициализируем рабочее пространство и соберем пакет.

```
cd ~/ros_workspace && catkin_init_workspace && catkin_make
```

Создадим новый пакет OrlovDL.

```
cd ~/ros_workspace/src && catkin_create_pkg OrlovDL std_msgs rospy
```

Создадим файлы скриптов Publisher и Subscriber. Сделаем их исполняемыми.

```
mkdir ~/ros_workspace/src/scripts  
cd ~/ros_workspace/src/scripts  
touch OrlovDL_publisher.py OrlovDL_subscriber.py  
chmod +x OrlovDL_publisher.py OrlovDL_subscriber.py
```

Напишем код скриптов (см. приложение) и пересоберем пакет.

```
cd ~/ros_workspace && catkin_init_workspace && catkin_make
```

Запустим ROS, ноду-публикатора и ноду-слушателя в трёх терминалах, таким образом проверим работу (рисунок 2).

```
roscore  
roslaunch OrlovDL OrlovDL_publisher.py  
roslaunch OrlovDL OrlovDL_subscriber.py
```

```
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://_:37873/
ros_comm version 1.14.10

SUMMARY
=====
PARAMETERS
* /rostdistro: melodic
* /rosversion: 1.14.10

NODES
auto-starting new master
process[master]: started with pid [11987]
ROS_MASTER_URI=http://_:11311/

setting /run_id to e690b9c6-43af-11eb-aefb-3ca067f50adf
process[roscout-l]: started with pid [11998]
started core service [/roscout]
^C[roscout-l] killing on exit
[master] killing on exit
shutting down processing monitor...
... shutting down processing monitor complete
done

~/ros_workspace$ roscore
... logging to /home/vladislav/.ros/log/33b39c46-43b0-11eb-aefb-3ca067f50adf/roslaunch-
-12332.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://_:43857/
ros_comm version 1.14.10

SUMMARY
=====
PARAMETERS
* /rostdistro: melodic
* /rosversion: 1.14.10

NODES
auto-starting new master
process[master]: started with pid [12342]
ROS_MASTER_URI=http://_:11311/

setting /run_id to 33b39c46-43b0-11eb-aefb-3ca067f50adf
process[roscout-l]: started with pid [12353]
started core service [/roscout]
```

```
[INFO] [1608571709.667106]: Test message. Time now: 1608571709
[INFO] [1608571710.667138]: Test message. Time now: 1608571710
[INFO] [1608571711.667132]: Test message. Time now: 1608571711
[INFO] [1608571712.667112]: Test message. Time now: 1608571712
[INFO] [1608571713.667150]: Test message. Time now: 1608571713
[INFO] [1608571714.667183]: Test message. Time now: 1608571714
[INFO] [1608571715.667139]: Test message. Time now: 1608571715
[INFO] [1608571716.667140]: Test message. Time now: 1608571716
[INFO] [1608571717.667135]: Test message. Time now: 1608571717
[INFO] [1608571718.667110]: Test message. Time now: 1608571718
[INFO] [1608571719.667091]: Test message. Time now: 1608571719
[INFO] [1608571720.667117]: Test message. Time now: 1608571720
[INFO] [1608571721.667076]: Test message. Time now: 1608571721
[INFO] [1608571722.667081]: Test message. Time now: 1608571722
[INFO] [1608571723.667101]: Test message. Time now: 1608571723
[INFO] [1608571724.667111]: Test message. Time now: 1608571724
[INFO] [1608571725.667136]: Test message. Time now: 1608571725
[INFO] [1608571726.667149]: Test message. Time now: 1608571726
[INFO] [1608571727.667140]: Test message. Time now: 1608571727
[INFO] [1608571728.667157]: Test message. Time now: 1608571728
[INFO] [1608571729.667108]: Test message. Time now: 1608571729
[INFO] [1608571730.667104]: Test message. Time now: 1608571730
[INFO] [1608571731.666980]: Test message. Time now: 1608571731
[INFO] [1608571732.667193]: Test message. Time now: 1608571732
[INFO] [1608571733.666768]: Test message. Time now: 1608571733
[INFO] [1608571734.666158]: Test message. Time now: 1608571734
```

```
[INFO] [1608571709.668700]: /hello_world_subscriber_13124_1608571657768 I heard msg: 'Test message. Time now: 1608571709'
[INFO] [1608571710.668763]: /hello_world_subscriber_13124_1608571657768 I heard msg: 'Test message. Time now: 1608571710'
[INFO] [1608571711.668641]: /hello_world_subscriber_13124_1608571657768 I heard msg: 'Test message. Time now: 1608571711'
[INFO] [1608571712.668564]: /hello_world_subscriber_13124_1608571657768 I heard msg: 'Test message. Time now: 1608571712'
[INFO] [1608571713.668902]: /hello_world_subscriber_13124_1608571657768 I heard msg: 'Test message. Time now: 1608571713'
[INFO] [1608571714.668809]: /hello_world_subscriber_13124_1608571657768 I heard msg: 'Test message. Time now: 1608571714'
[INFO] [1608571715.668926]: /hello_world_subscriber_13124_1608571657768 I heard msg: 'Test message. Time now: 1608571715'
[INFO] [1608571716.668780]: /hello_world_subscriber_13124_1608571657768 I heard msg: 'Test message. Time now: 1608571716'
[INFO] [1608571717.668947]: /hello_world_subscriber_13124_1608571657768 I heard msg: 'Test message. Time now: 1608571717'
[INFO] [1608571718.668634]: /hello_world_subscriber_13124_1608571657768 I heard msg: 'Test message. Time now: 1608571718'
[INFO] [1608571719.668623]: /hello_world_subscriber_13124_1608571657768 I heard msg: 'Test message. Time now: 1608571719'
[INFO] [1608571720.668957]: /hello_world_subscriber_13124_1608571657768 I heard msg: 'Test message. Time now: 1608571720'
[INFO] [1608571721.668624]: /hello_world_subscriber_13124_1608571657768 I heard msg: 'Test message. Time now: 1608571721'
[INFO] [1608571722.668684]: /hello_world_subscriber_13124_1608571657768 I heard msg: 'Test message. Time now: 1608571722'
[INFO] [1608571723.668628]: /hello_world_subscriber_13124_1608571657768 I heard msg: 'Test message. Time now: 1608571723'
[INFO] [1608571724.668840]: /hello_world_subscriber_13124_1608571657768 I heard msg: 'Test message. Time now: 1608571724'
[INFO] [1608571725.668955]: /hello_world_subscriber_13124_1608571657768 I heard msg: 'Test message. Time now: 1608571725'
[INFO] [1608571726.668973]: /hello_world_subscriber_13124_1608571657768 I heard msg: 'Test message. Time now: 1608571726'
[INFO] [1608571727.668912]: /hello_world_subscriber_13124_1608571657768 I heard msg: 'Test message. Time now: 1608571727'
[INFO] [1608571728.668875]: /hello_world_subscriber_13124_1608571657768 I heard msg: 'Test message. Time now: 1608571728'
[INFO] [1608571729.668599]: /hello_world_subscriber_13124_1608571657768 I heard msg: 'Test message. Time now: 1608571729'
[INFO] [1608571730.668624]: /hello_world_subscriber_13124_1608571657768 I heard msg: 'Test message. Time now: 1608571730'
[INFO] [1608571731.668488]: /hello_world_subscriber_13124_1608571657768 I heard msg: 'Test message. Time now: 1608571731'
[INFO] [1608571732.669127]: /hello_world_subscriber_13124_1608571657768 I heard msg: 'Test message. Time now: 1608571732'
[INFO] [1608571733.669244]: /hello_world_subscriber_13124_1608571657768 I heard msg: 'Test message. Time now: 1608571733'
[INFO] [1608571734.667792]: /hello_world_subscriber_13124_1608571657768 I heard msg: 'Test message. Time now: 1608571734'
```

Рисунок 2 - Демонстрация работы Publisher и Subscriber

ВЫВОД

В ходе лабораторной работы были получены практические навыки по установке и настройке среды ROS-Melodic. Была настроена рабочая среда для создания новых проектов. Был создан и собран новый проект “ OrlovDL”, который реализует публикацию и чтение данных в информационную среду ROS.

ПРИЛОЖЕНИЕ

OrlovDL_publisher

```
#!/usr/bin/env python import
rospy from std_msgs.msg import
String

def talker():
    Pub = rospy.Publisher('hello_pub', String, queue_size=10)
    rospy.init_node('OrlovDL_publisher', anonymous=True)    r =
    rospy.Rate(1)

    while not rospy.is_shutdown():
        msg = "Test message. Time now: " + str(rospy.get_time())
        [:-3]
    rospy.loginfo(msg)
    OrlovDL.publish(msg)
    r.sleep()

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

OrlovDL_subscriber

```
#!/usr/bin/env python import
rospy from std_msgs.msg import
String

def callback(data):
    rospy.loginfo(rospy.get_caller_id() + " I heard msg: '%s'",
    data.data)

def listener():
```

```
    rospy.init_node(' OrlovDL_subscriber', anonymous=True)
rospy.Subscriber("hello_pub", String, callback)    rospy.spin()

if __name__ == '__main__':
    listener()
```