

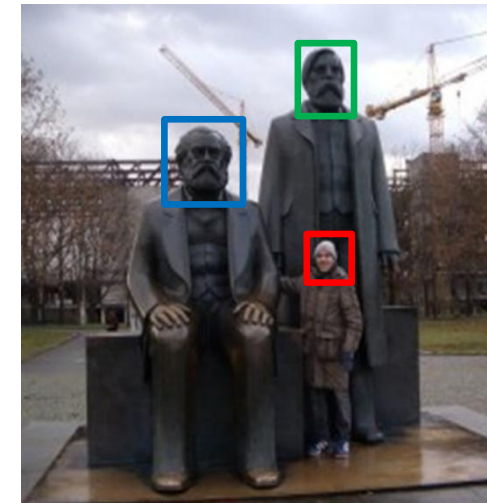
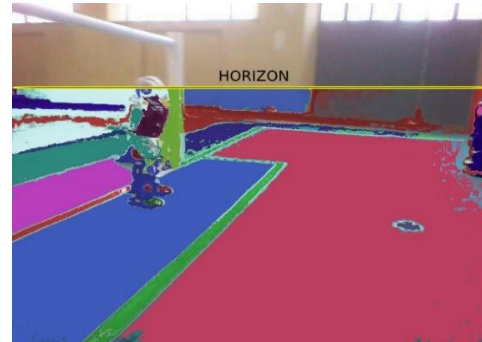
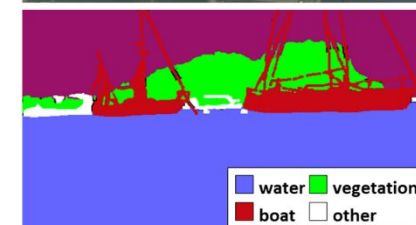


**UNIVERSITÀ DEGLI STUDI
DELLA BASILICATA**

Corso di Sistemi Informativi
A.A. 2018/19

Docente
Domenico Daniele Bloisi

ROS launch file



Maggio 2019

roslaunch

roslaunch è un tool per semplificare

- il lancio di più nodi ROS
- il settaggio dei parametri

roslaunch utilizza i cosiddetti “launch file” che sono file XML contenenti la lista dei nodi da lanciare con i rispettivi parametri

roslaunch - sintassi

```
roslaunch <package> <launch file>
```

- i launch file hanno per convenzione un nome che termina con `.launch`
- `roscore` viene automaticamente lanciato quando si esegue `roslaunch`

Esempio launch file

```
<launch>
```

```
  <node name="talker" pkg="hello_ros" type="talker" output="screen"/>
```

```
  <node name="listener" pkg="hello_ros" type="listener" output="screen"/>
```

```
</launch>
```

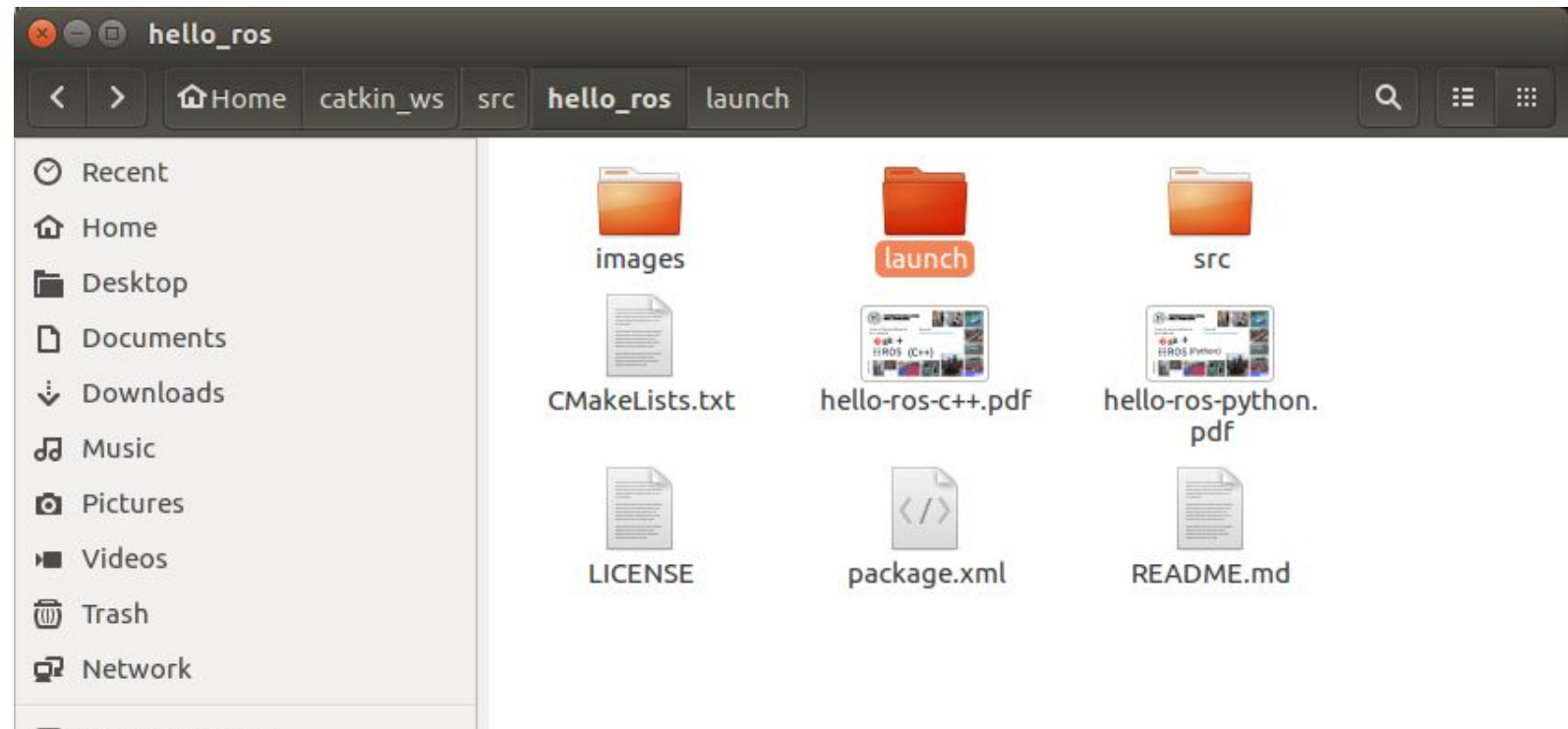
- Il tag `<node>` contiene gli attributi per specificare il nome con cui il nome verrà inserito nel grafo di ROS, il package nel quale può essere trovato e il `type`, che è il filename dell'eseguibile
- L'attributo `output` posto a "screen" indica che i messaggi di log di ROS verranno mostrati sul terminale su cui verrà eseguito il comando `roslaunch`

hello_ros: git repo recap

```
bloisi@bloisi-U36SG:~/catkin_ws/src$ git clone https://github.com/dbloisi/hello_ros.git
Cloning into 'hello_ros'...
remote: Enumerating objects: 26, done.
remote: Counting objects: 100% (26/26), done.
remote: Compressing objects: 100% (26/26), done.
remote: Total 74 (delta 13), reused 0 (delta 0), pack-reused 48
Unpacking objects: 100% (74/74), done.
Checking connectivity... done.
bloisi@bloisi-U36SG:~/catkin_ws/src$ cd hello_ros
bloisi@bloisi-U36SG:~/catkin_ws/src/hello_ros$ ls
CMakeLists.txt      hello-ros-python.pdf  LICENSE             README.md
hello-ros-c++.pdf   images                package.xml         src
bloisi@bloisi-U36SG:~/catkin_ws/src/hello_ros$
```

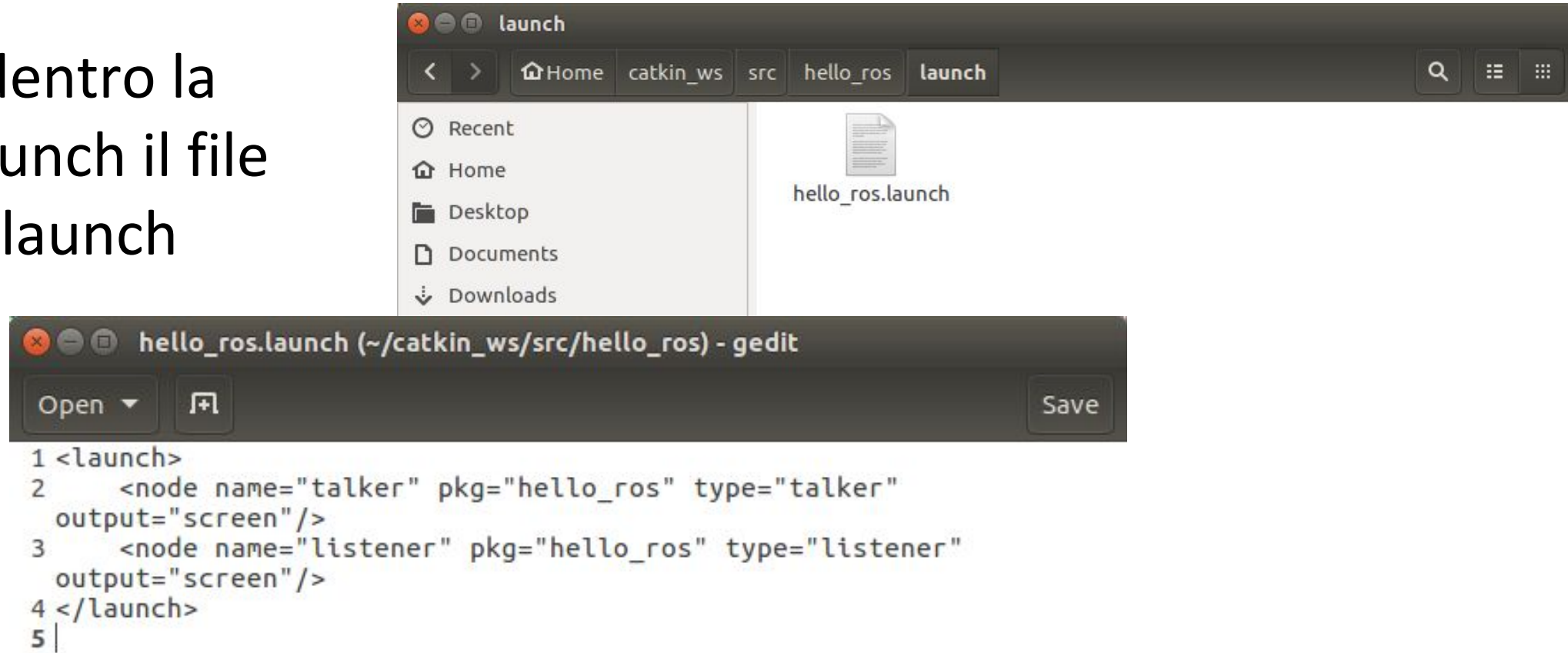
hello_ros launch file

Creiamo una cartella
launch



hello_ros launch file

Creiamo dentro la cartella launch il file hello_ros.launch



hello_ros launch file: esecuzione

```
bloisi@bloisi-U36SG: ~  
bloisi@bloisi-U36SG:~$ roslaunch hello_ros hello_ros.launch  
... logging to /home/bloisi/.ros/log/2977c8b2-716e-11e9-a68b-50465dde6884/roslau  
nch-bloisi-U36SG-6381.log  
Checking log directory for disk usage. This may take awhile.  
Press Ctrl-C to interrupt  
Done checking log file disk usage. Usage is <1GB.  
  
started roslaunch server http://localhost:41695/  
  
SUMMARY  
=====  
  
PARAMETERS  
* /rostdistro: kinetic  
* /rosversion: 1.12.14  
  
NODES  
/  
  listener (hello_ros/listener)  
  talker (hello_ros/talker)  
  
auto-starting new master  
process[master]: started with pid [6391]  
ROS_MASTER_URI=http://localhost:11311  
  
setting /run_id to 2977c8b2-716e-11e9-a68b-50465dde6884  
process[rosout-1]: started with pid [6404]  
started core service [/rosout]  
process[talker-2]: started with pid [6408]  
process[listener-3]: started with pid [6422]  
[ INFO] [1557305357.648592063]: hello world 0  
[ INFO] [1557305357.748768063]: hello world 1  
[ INFO] [1557305357.848722995]: hello world 2  
[ INFO] [1557305357.948727541]: hello world 3  
[ INFO] [1557305357.949373170]: I heard: [hello world 3]  
[ INFO] [1557305358.048732949]: hello world 4  
[ INFO] [1557305358.049583276]: I heard: [hello world 4]  
[ INFO] [1557305358.148735150]: hello world 5  
[ INFO] [1557305358.149535192]: I heard: [hello world 5]  
[ INFO] [1557305358.248783039]: hello world 6
```

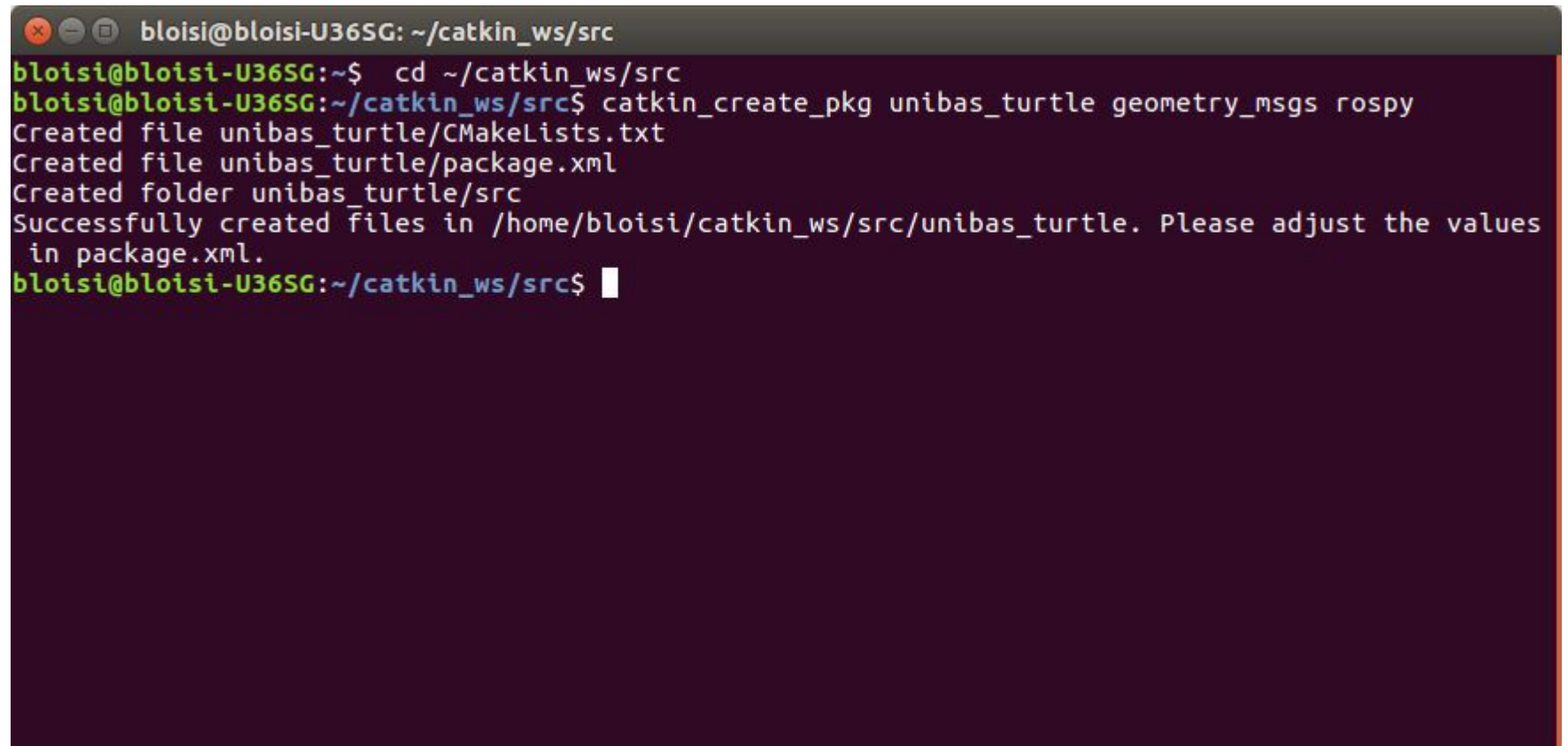

hello_ros launch file: git

```
bloisi@bloisi-U36SG: ~/catkin_ws/src/hello_ros
bloisi@bloisi-U36SG:~/catkin_ws/src/hello_ros$ ls
CMakeLists.txt      hello-ros-python.pdf  launch      package.xml  src
hello-ros-c++.pdf   images                LICENSE     README.md
bloisi@bloisi-U36SG:~/catkin_ws/src/hello_ros$ git add launch
bloisi@bloisi-U36SG:~/catkin_ws/src/hello_ros$ git commit -m "adding launch folder and launch file"
[master eeefdd0] adding launch folder and launch file
1 file changed, 5 insertions(+)
create mode 100644 launch/hello_ros.launch
bloisi@bloisi-U36SG:~/catkin_ws/src/hello_ros$ git pull
Already up-to-date.
bloisi@bloisi-U36SG:~/catkin_ws/src/hello_ros$ git push origin master
Username for 'https://github.com': dbloisi
Password for 'https://dbloisi@github.com':
Counting objects: 5, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (5/5), 519 bytes | 0 bytes/s, done.
Total 5 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 1 local object.
To https://github.com/dbloisi/hello_ros.git
03675ef..eeefdd0  master -> master
bloisi@bloisi-U36SG:~/catkin_ws/src/hello_ros$
```

Package unibas_turtle

```
$ cd ~/catkin_ws/src
```

```
$ catkin_create_pkg unibas_turtle geometry_msgs rospy
```

A terminal window with a dark purple background and a title bar that reads 'bloisi@bloisi-U36SG: ~/catkin_ws/src'. The terminal shows the following commands and output:

```
bloisi@bloisi-U36SG:~$ cd ~/catkin_ws/src
bloisi@bloisi-U36SG:~/catkin_ws/src$ catkin_create_pkg unibas_turtle geometry_msgs rospy
Created file unibas_turtle/CMakeLists.txt
Created file unibas_turtle/package.xml
Created folder unibas_turtle/src
Successfully created files in /home/bloisi/catkin_ws/src/unibas_turtle. Please adjust the values
in package.xml.
bloisi@bloisi-U36SG:~/catkin_ws/src$
```

<http://wiki.ros.org/turtlesim/Tutorials/Moving%20in%20a%20Straight%20Line>

Package unibas_turtle

```
$ cd ~/catkin_ws
```

```
$ catkin_make
```

```
bloisi@bloisi-U36SG: ~/catkin_ws
bloisi@bloisi-U36SG:~/catkin_ws/src/hello_ros/src$ cd ~/catkin_ws/
bloisi@bloisi-U36SG:~/catkin_ws$ catkin_make
Base path: /home/bloisi/catkin_ws
Source space: /home/bloisi/catkin_ws/src
Build space: /home/bloisi/catkin_ws/build
Devel space: /home/bloisi/catkin_ws/devel
Install space: /home/bloisi/catkin_ws/install
####
#### Running command: "make cmake_check_build_system" in "/home/bloisi/catkin_ws/build"
####
-- Using CATKIN_DEVEL_PREFIX: /home/bloisi/catkin_ws/devel
-- Using CMAKE_PREFIX_PATH: /home/bloisi/catkin_ws/devel;/opt/ros/kinetic
-- This workspace overlays: /home/bloisi/catkin_ws/devel;/opt/ros/kinetic
-- Using PYTHON_EXECUTABLE: /usr/bin/python
-- Using Debian Python package layout
-- Using empy: /usr/bin/empy
-- Using CATKIN_ENABLE_TESTING: ON
-- Call enable_testing()
-- Using CATKIN_TEST_RESULTS_DIR: /home/bloisi/catkin_ws/build/test_results
-- Found gmock sources under '/usr/src/gmock': gmock will be built
-- Found gtest sources under '/usr/src/gmock': gtests will be built
-- Using Python nosetests: /usr/bin/nosetests-2.7
-- catkin 0.7.14
```

<http://wiki.ros.org/turtlesim/Tutorials/Moving%20in%20a%20Straight%20Line>

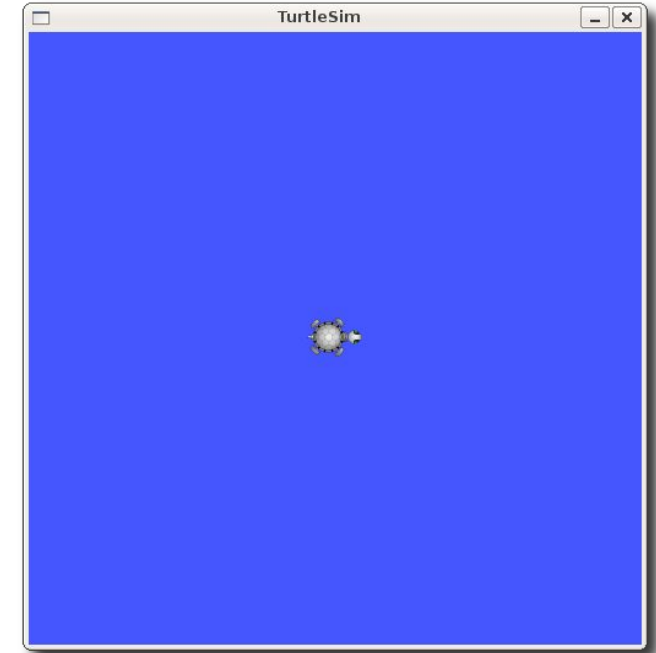
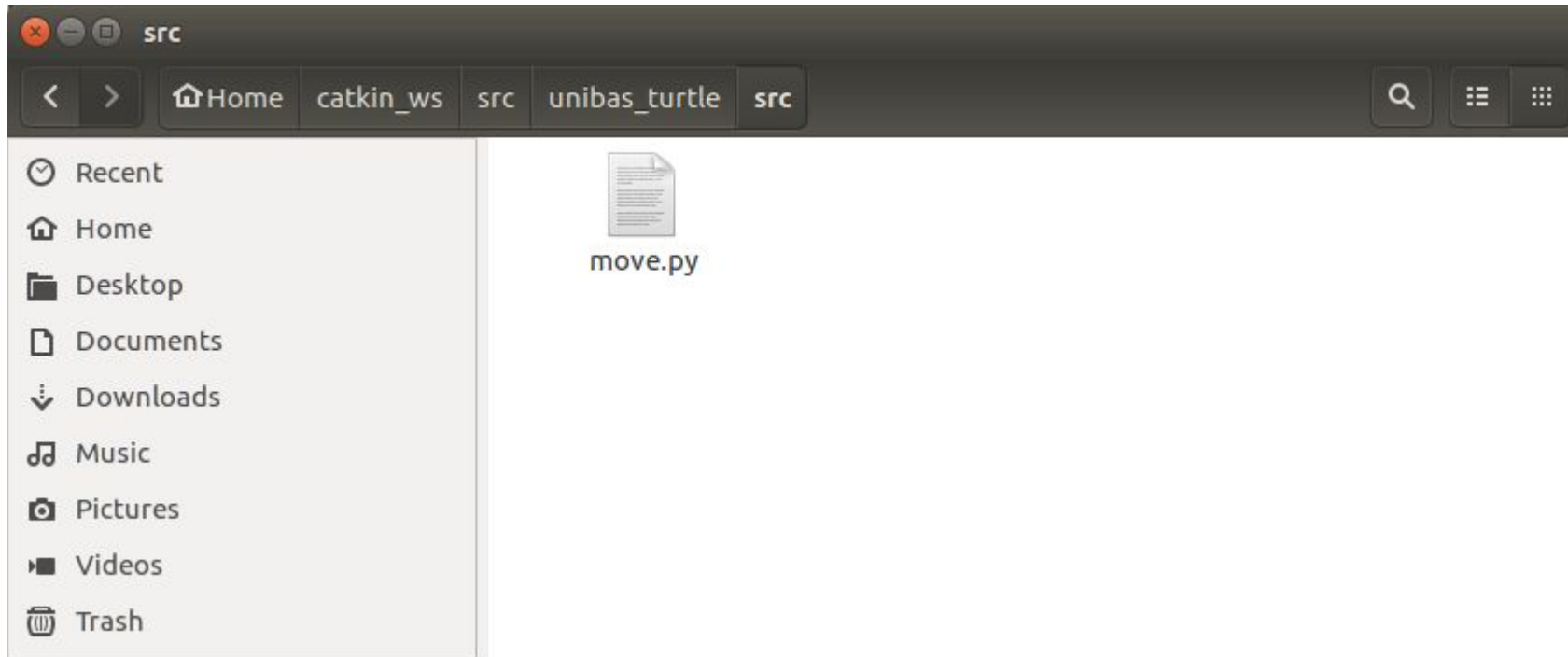
Package unibas_turtle: src

```
bloisi@bloisi-U36SG: ~/catkin_ws/src/unibas_turtle/src
bloisi@bloisi-U36SG:~/catkin_ws$ cd ~/catkin_ws/src/unibas_turtle
bloisi@bloisi-U36SG:~/catkin_ws/src/unibas_turtle$ ls
CMakeLists.txt  package.xml  src
bloisi@bloisi-U36SG:~/catkin_ws/src/unibas_turtle$ cd src
bloisi@bloisi-U36SG:~/catkin_ws/src/unibas_turtle/src$ ls
bloisi@bloisi-U36SG:~/catkin_ws/src/unibas_turtle/src$
```

<http://wiki.ros.org/turtlesim/Tutorials/Moving%20in%20a%20Straight%20Line>

Package unibas_turtle: src

Creiamo un file move.py per far muovere la tartaruga di turtlesim



<http://wiki.ros.org/turtlesim/Tutorials/Moving%20in%20a%20Straight%20Line>

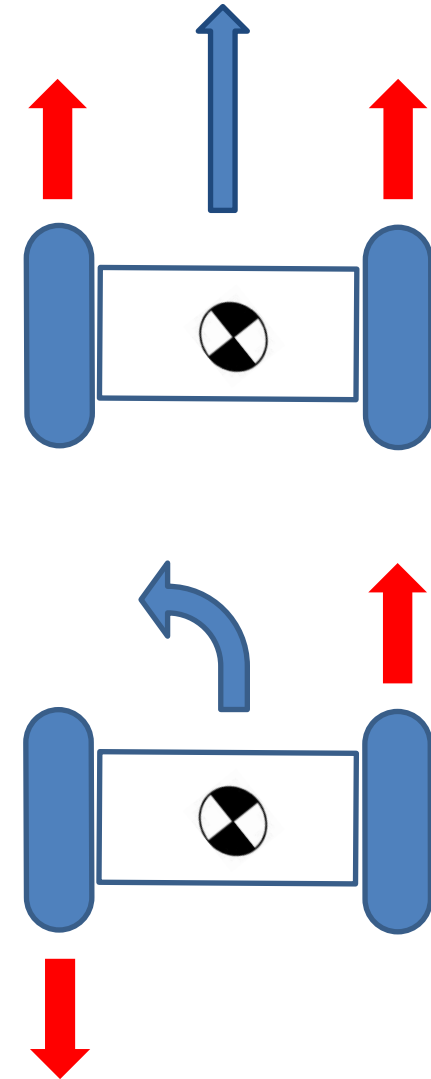
<http://wiki.ros.org/turtlesim>

idea

- Vogliamo far muovere la tartaruga controllandone la velocità
- Utilizziamo per la tartaruga il modello di un robot differenziale
- In questo modo possiamo utilizzare per il controllo la velocità lineare e la velocità angolare

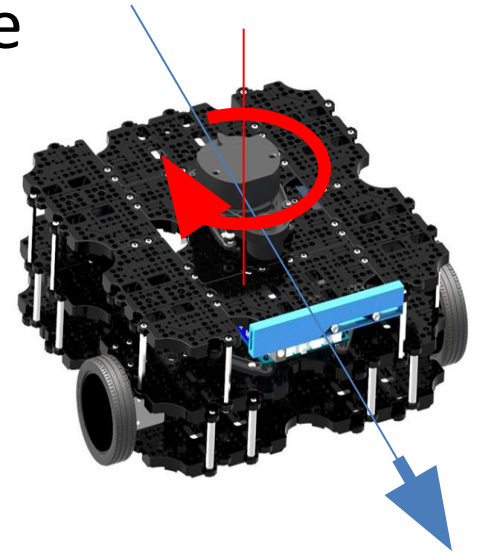
Differential drive robot

- Un robot differenziale su ruote è una base mobile avente due ruote motorizzate indipendenti
- Le ruote sono posizionate ai due lati opposti della scocca
- Il robot si muove in avanti quando entrambe le ruote girano in avanti, mentre gira sul posto quando una ruota gira in avanti e l'altra gira all'indietro



Movimento del robot

- Data la sua configurazione, un robot differenziale può muoversi solo in avanti o indietro lungo il suo asse longitudinale e può ruotare solo lungo il suo asse verticale
- Il robot non potrà muoversi di lato o verticalmente
- Per tali motivi ci bastano la componente lineare x e la componente angolare z per controllare il movimento
- Nel caso di un robot **omnidirezionale**, avremo anche una componente y per lo spostamento laterale
- **Quante componenti avremo per un robot underwater?**



Comandi di velocità in ROS

Per far muovere un robot in ROS è necessario pubblicare Twist messages sul topic cmd_vel

[geometry_msgs/Twist](#) Message

File: `geometry_msgs/Twist.msg`

Raw Message Definition

```
# This expresses velocity in free space broken into its linear and angular parts.  
Vector3  linear  
Vector3  angular
```

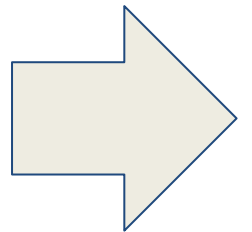
Compact Message Definition

```
geometry_msgs/Vector3 linear  
geometry_msgs/Vector3 angular
```

move.py

```
#!/usr/bin/env python
import rospy
from geometry_msgs.msg import Twist

def move():
    # Starts a new node
    rospy.init_node('robot_cleaner', anonymous=True)
    velocity_publisher = rospy.Publisher('/turtle1/cmd_vel', Twist, queue_size=10)
    vel_msg = Twist()
```



move.py

#Receiveing the user's input

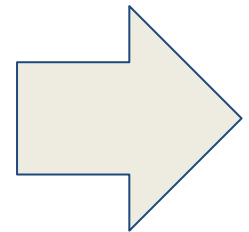
```
print("Let's move your robot")
speed = input("Input your speed:")
distance = input("Type your distance:")
isForward = input("Foward?: ")#True or False
```

#Checking if the movement is forward or backwards

```
if(isForward):
    vel_msg.linear.x = abs(speed)
else:
    vel_msg.linear.x = -abs(speed)
```

#Since we are moving just in x-axis

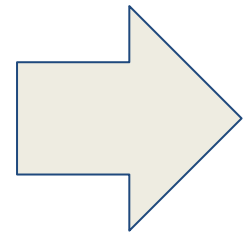
```
vel_msg.linear.y = 0
vel_msg.linear.z = 0
vel_msg.angular.x = 0
vel_msg.angular.y = 0
vel_msg.angular.z = 0
```



move.py

```
while not rospy.is_shutdown():
    #Setting the current time for distance calculus
    t0 = rospy.Time.now().to_sec()
    current_distance = 0

    #Loop to move the turtle in an specified distance
    while(current_distance < distance):
        #Publish the velocity
        velocity_publisher.publish(vel_msg)
        #Takes actual time to velocity calculus
        t1=rospy.Time.now().to_sec()
        #Calculates distancePoseStamped
        current_distance= speed*(t1-t0)
    #After the loop, stops the robot
    vel_msg.linear.x = 0
    #Force the robot to stop
    velocity_publisher.publish(vel_msg)
```



move.py

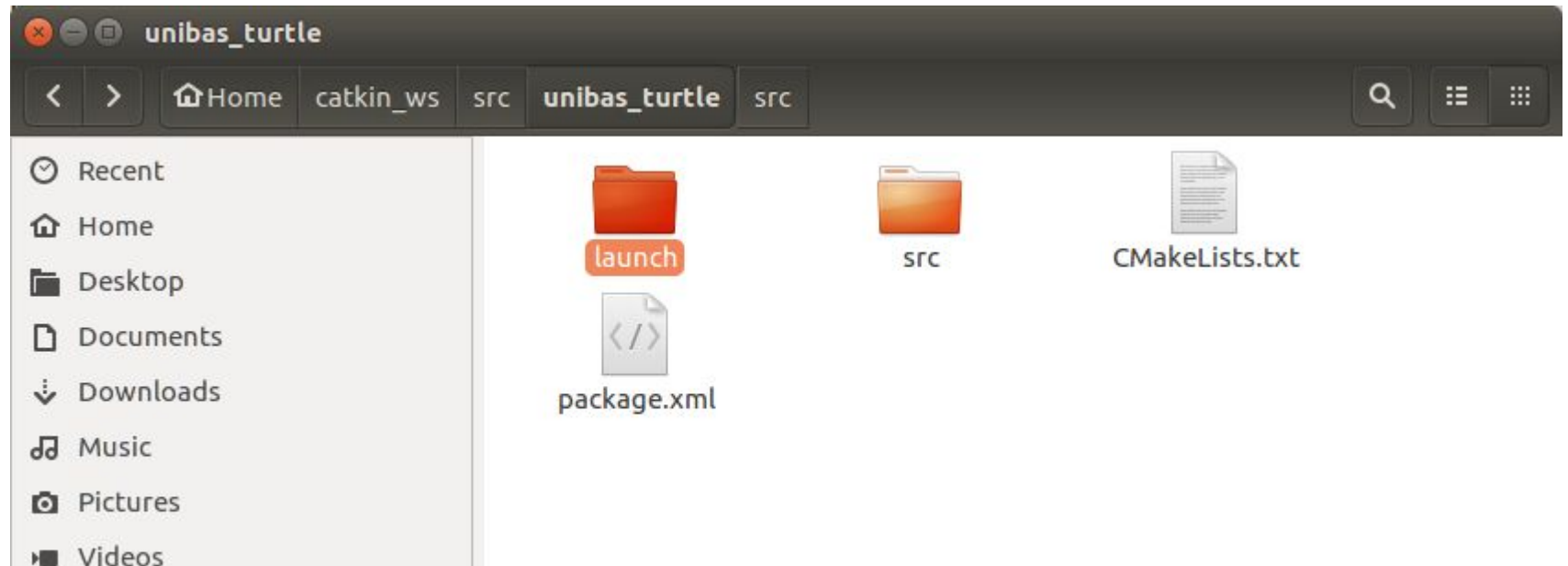
```
if __name__ == '__main__':  
    try:  
        #Testing our function  
        move()  
    except rospy.ROSInterruptException: pass
```

permessi per move.py

```
bloisi@bloisi-U36SG: ~/catkin_ws/src/unibas_turtle/src
bloisi@bloisi-U36SG:~/catkin_ws/src/unibas_turtle/src$ ls
move.py
bloisi@bloisi-U36SG:~/catkin_ws/src/unibas_turtle/src$ chmod u+x ~/catkin_ws/src/unibas_turtle/src/move.py
bloisi@bloisi-U36SG:~/catkin_ws/src/unibas_turtle/src$
```

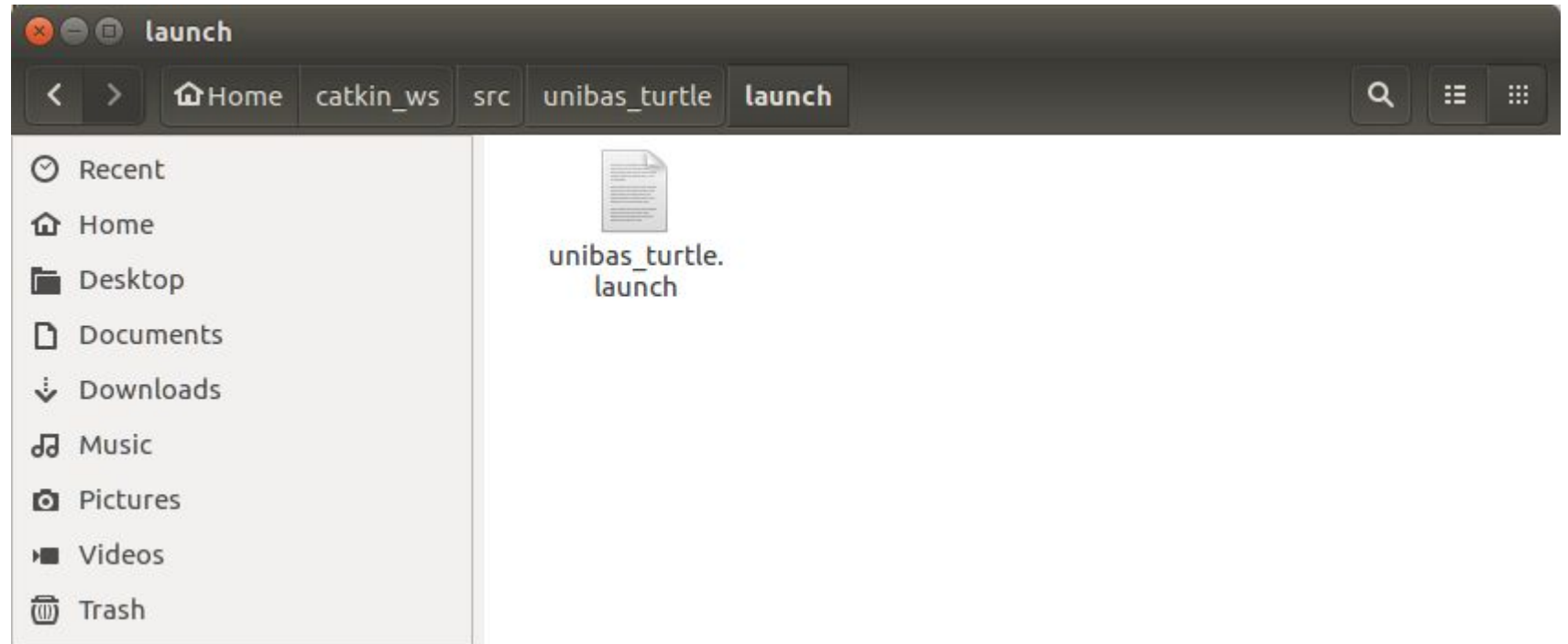
Launch file per unibas_turtle

Creiamo una
cartella launch



Launch file per unibas_turtle

Creiamo un file
unibas_turtle.launch
dentro la cartella
launch



unibas_turtle.launch

unibas_turtle.launch (~/.catkin_ws/src/unibas_turtle/launch) - gedit

Open ▼



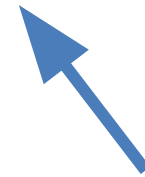
```
1 <launch>
2   <node name="turtlesim_node" pkg="turtlesim" type="turtlesim_node" output="screen"/>
3   <node name="move" pkg="unibas_turtle" type="move.py" output="screen"/>
4 </launch>
5
```

Esempio roslaunch

```
roslaunch unibas_turtle unibas_turtle.launch
```



ROS package name



launch file name

Esecuzione roslaunch

```
/home/bloisi/catkin_ws/src/unibas_turtle/launch/unibas_turtle.launch http://localhost:11311
bloisi@bloisi-U36SG:~/catkin_ws$ roslaunch unibas_turtle unibas_turtle.launch
... logging to /home/bloisi/.ros/log/3d4b327a-717d-11e9-a68b-50465dde6884/roslaunch-l
Checking log directory for disk usage. This may take awhile.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://localhost:43063/

SUMMARY
=====

PARAMETERS
* /rostdistro: kinetic
* /rosversion: 1.12.14

NODES
/
  move (unibas_turtle/move.py)
  turtlesim_node (turtlesim/turtlesim_node)

auto-starting new master
process[master]: started with pid [13029]
ROS_MASTER_URI=http://localhost:11311

setting /run_id to 3d4b327a-717d-11e9-a68b-50465dde6884
process[rosout-1]: started with pid [13042]
started core service [/rosout]
process[turtlesim_node-2]: started with pid [13047]
process[move-3]: started with pid [13054]
[ INFO] [1557311833.581376090]: Starting turtlesim with node name /turtlesim_node
[ INFO] [1557311833.587148170]: Spawning turtle [turtle1] at x=[5,544445], y=[5,5444
Let's move your robot
Input your speed:
```



Esecuzione roslaunch

```
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1G
started roslaunch server http://localhost:39428/

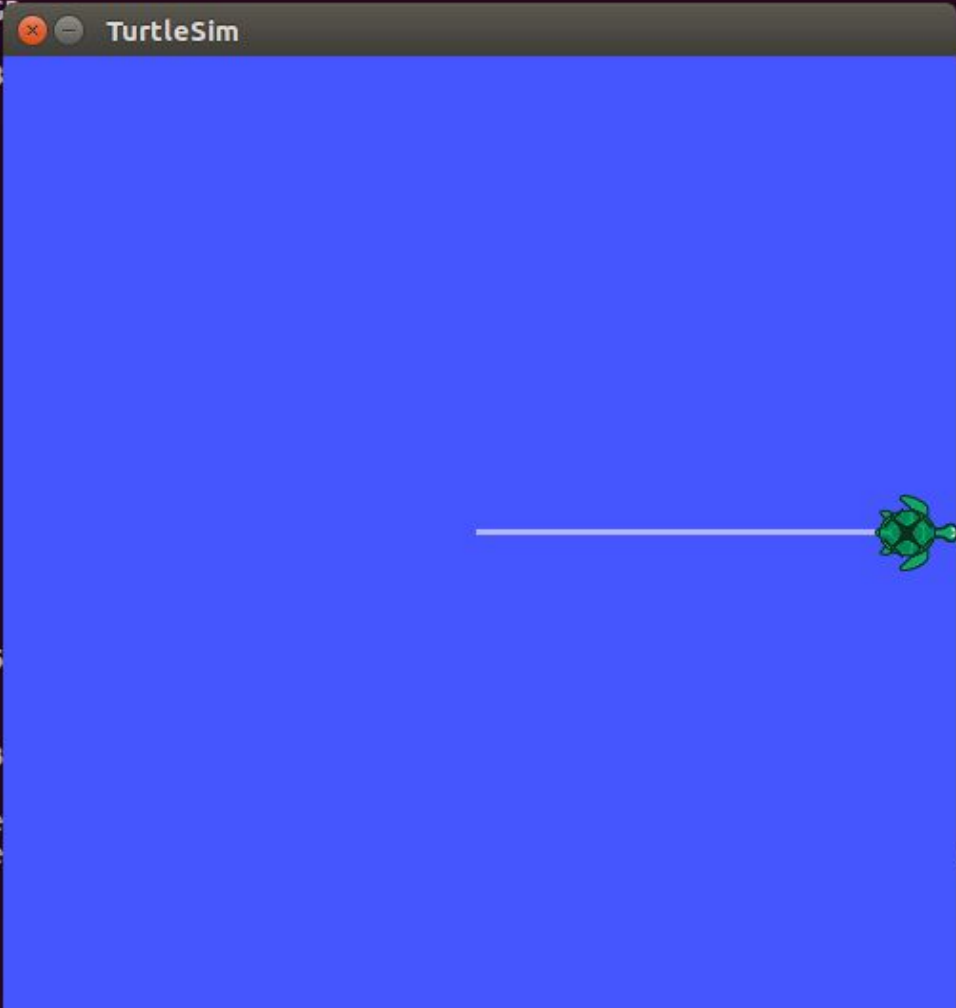
SUMMARY
=====

PARAMETERS
* /roscore: kinetic
* /rosversion: 1.12.14

NODES
/
  move (unibas_turtle/move.py)
  turtlesim_node (turtlesim/turtlesim_node)

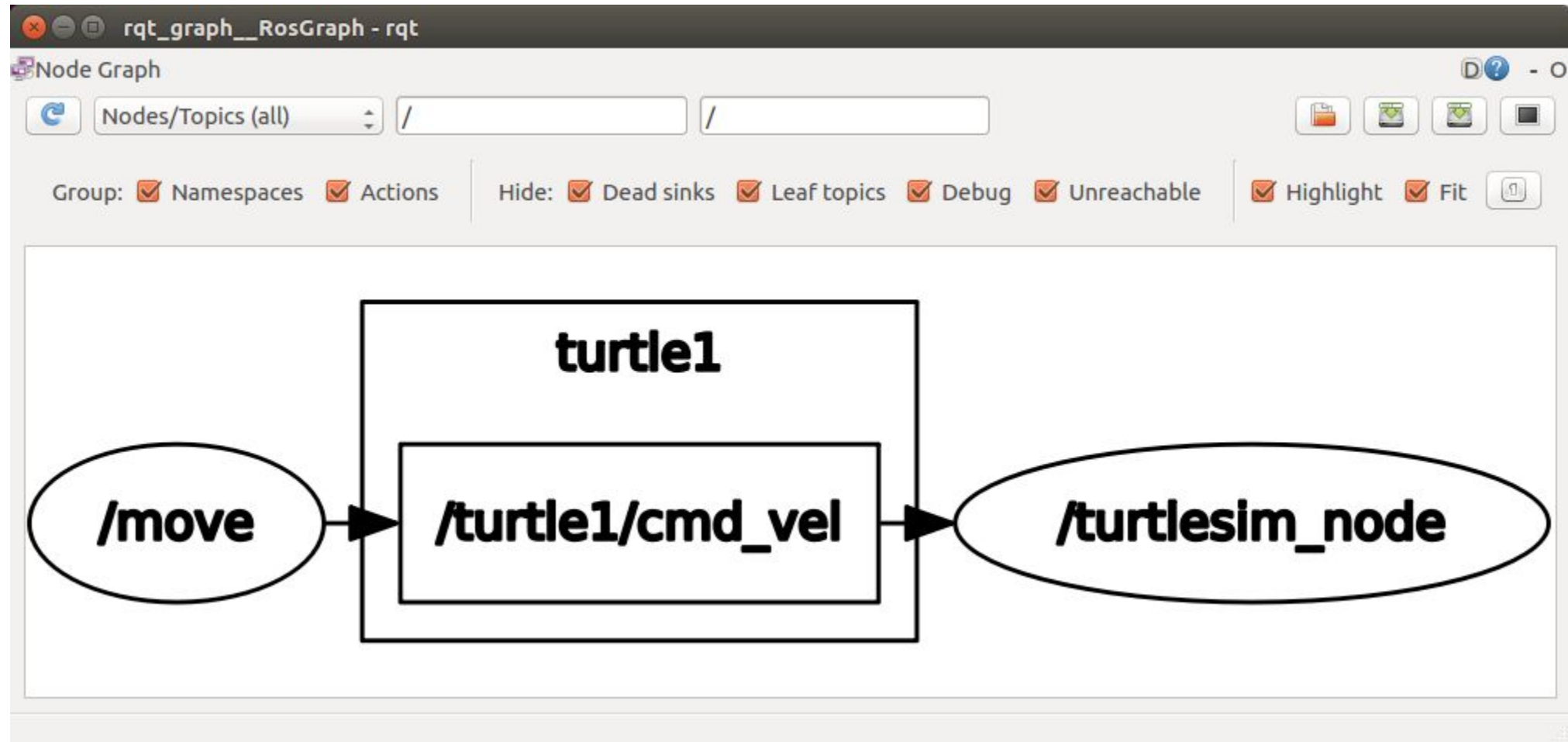
auto-starting new master
process[master]: started with pid [13456]
ROS_MASTER_URI=http://localhost:11311

setting /run_id to 4c7e0a04-717f-11e9-a68b-5046
process[roscout-1]: started with pid [13469]
started core service [/roscout]
process[turtlesim_node-2]: started with pid [13487]
process[move-3]: started with pid [13487]
[ INFO] [1557312718.258932779]: Starting turtle
[ INFO] [1557312718.264864776]: Spawning turtle
Let's move your robot
Input your speed:10
Type your distance:5
Forward?: 1
```



The image shows a terminal window with roslaunch output and a TurtleSim window. The terminal output shows the launch of the turtlesim and move packages, the starting of a new master, and the spawning of a turtle. The TurtleSim window displays a green turtle on a blue background, with a white line trailing behind it, indicating movement.

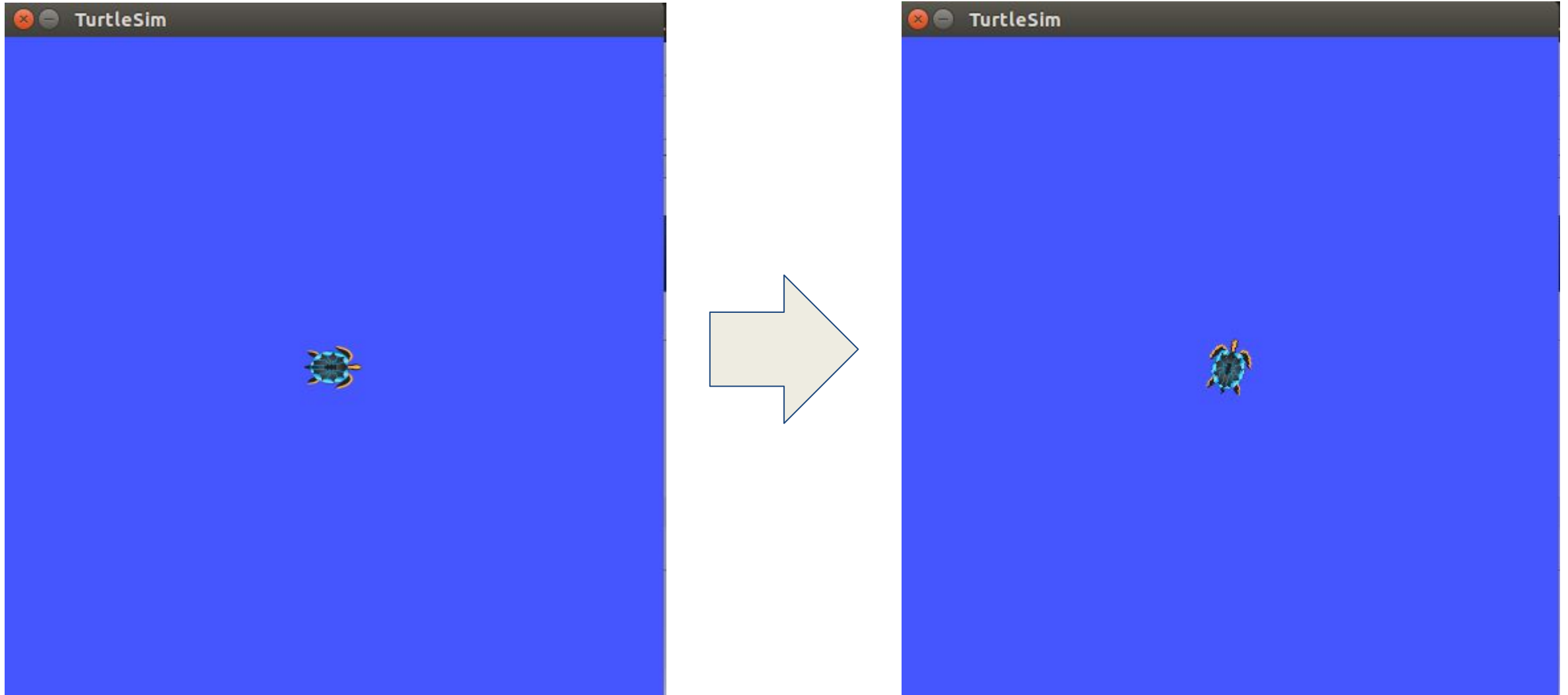
rqt_graph



topic e message

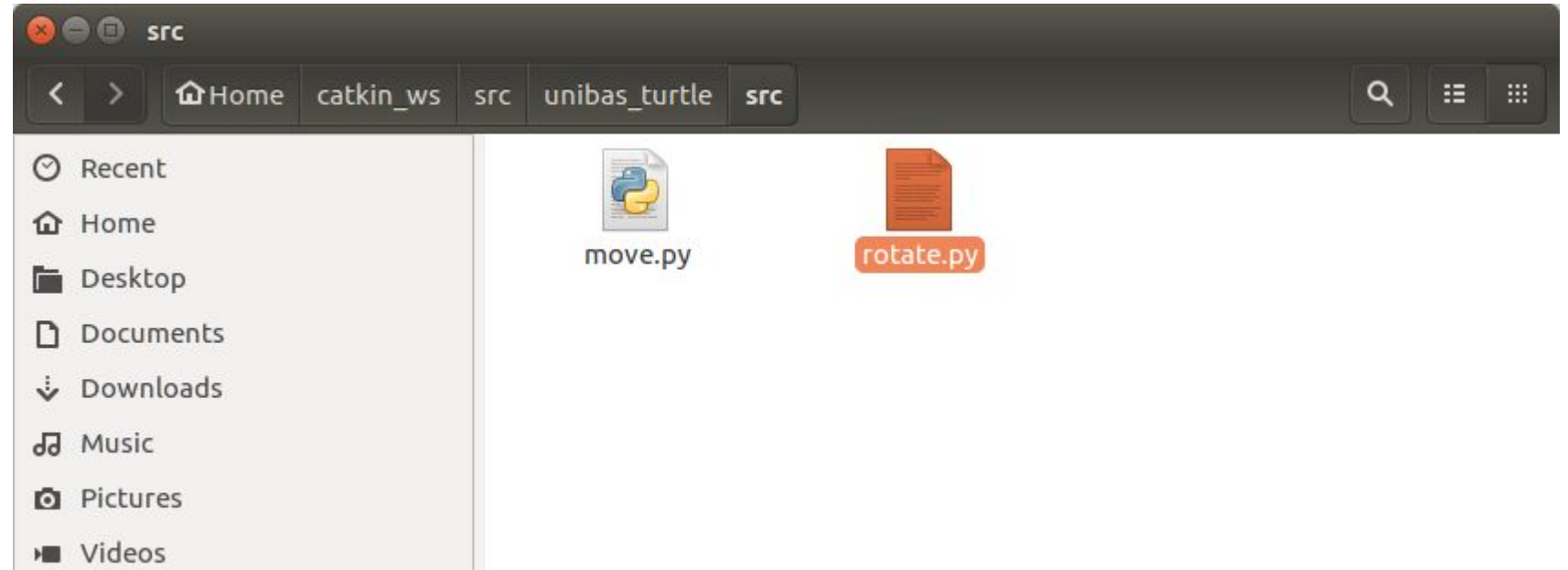
```
bloisi@bloisi-U36SG: ~  
bloisi@bloisi-U36SG:~$ rostopic info /turtle1/cmd_vel  
Type: geometry_msgs/Twist  
  
Publishers:  
* /move (http://localhost:36974/)  
  
Subscribers:  
* /turtlesim_node (http://localhost:43408/)  
  
bloisi@bloisi-U36SG:~$ rosmmsg show geometry_msgs/Twist  
geometry_msgs/Vector3 linear  
  float64 x  
  float64 y  
  float64 z  
geometry_msgs/Vector3 angular  
  float64 x  
  float64 y  
  float64 z  
  
bloisi@bloisi-U36SG:~$
```

Rotating left and right



creazione di rotate.py

Creiamo un file
rotate.py
dentro la
cartella src



rotate.py

```
#!/usr/bin/env python
```

```
import rospy
```

```
from geometry_msgs.msg import Twist
```

```
PI = 3.1415926535897
```

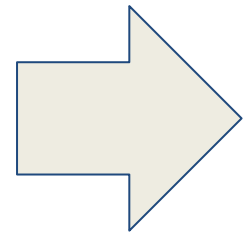
```
def rotate():
```

```
    #Starts a new node
```

```
    rospy.init_node('robot_cleaner', anonymous=True)
```

```
    velocity_publisher = rospy.Publisher('/turtle1/cmd_vel', Twist, queue_size=10)
```

```
    vel_msg = Twist()
```

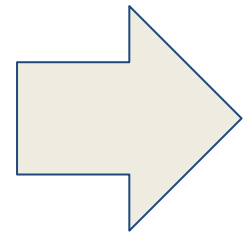


rotate.py

```
# Receiveing the user's input
print("Let's rotate your robot")
speed = input("Input your speed (degrees/sec):")
angle = input("Type your distance (degrees):")
clockwise = input("Clockwise?: ") #True or false
```

```
#Converting from angles to radians
angular_speed = speed*2*PI/360
relative_angle = angle*2*PI/360
```

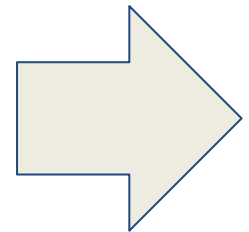
```
#We wont use linear components
vel_msg.linear.x=0
vel_msg.linear.y=0
vel_msg.linear.z=0
vel_msg.angular.x = 0
vel_msg.angular.y = 0
```



rotate.py

```
# Checking if our movement is CW or CCW
if clockwise:
    vel_msg.angular.z = -abs(angular_speed)
else:
    vel_msg.angular.z = abs(angular_speed)
# Setting the current time for distance calculus
t0 = rospy.Time.now().to_sec()
current_angle = 0

while(current_angle < relative_angle):
    velocity_publisher.publish(vel_msg)
    t1 = rospy.Time.now().to_sec()
    current_angle = angular_speed*(t1-t0)
```



rotate.py

```
#Forcing our robot to stop
```

```
vel_msg.angular.z = 0
```

```
velocity_publisher.publish(vel_msg)
```

```
rospy.spin()
```

```
if __name__ == '__main__':
```

```
    try:
```

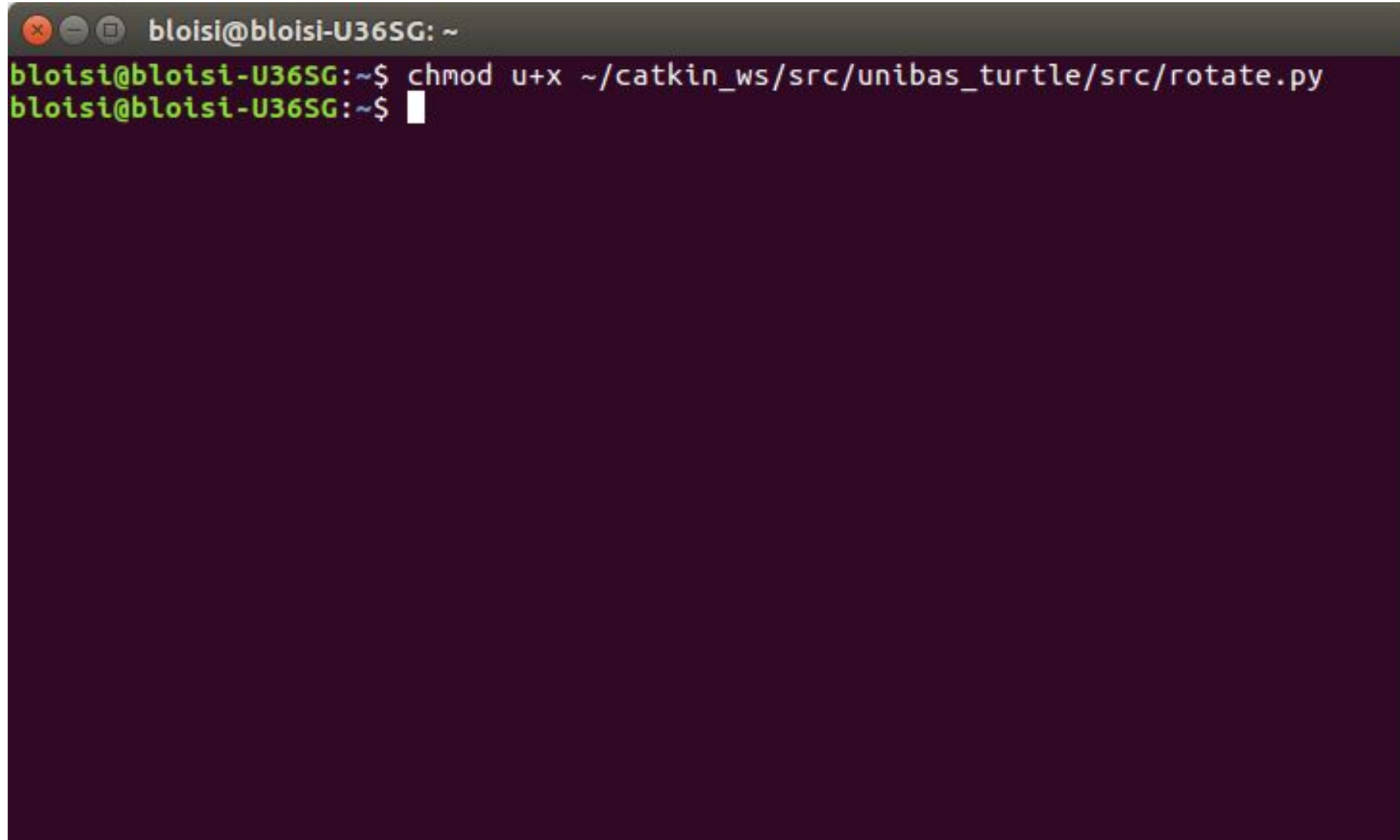
```
        # Testing our function
```

```
        rotate()
```

```
    except rospy.ROSInterruptException:
```

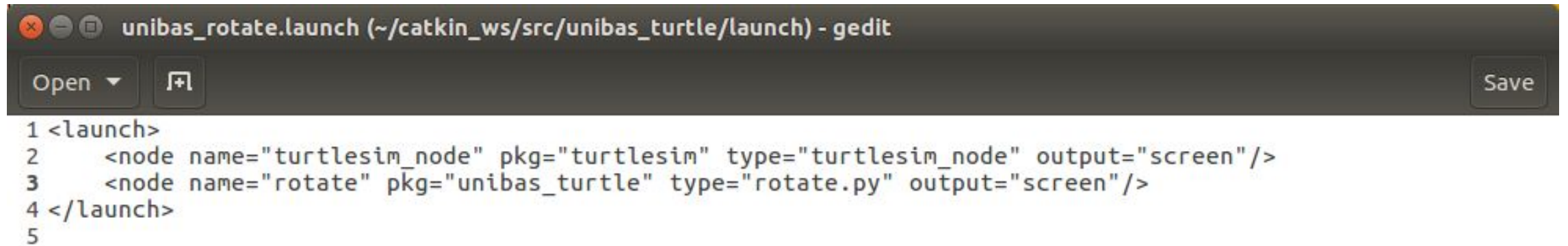
```
        pass
```


permessi per rotate.py

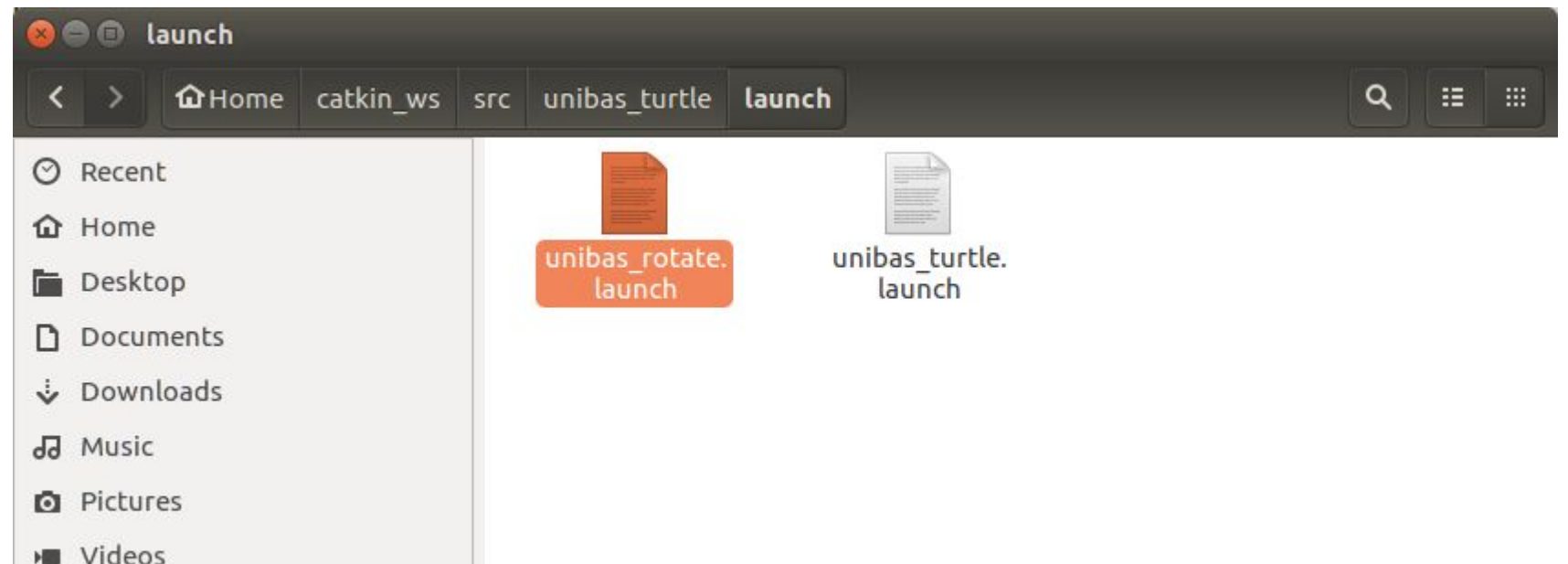
A terminal window with a dark purple background and a grey title bar. The title bar contains three window control icons (close, minimize, maximize) and the text 'bloisi@bloisi-U36SG: ~'. The terminal shows two lines of text: the first line is a command 'chmod u+x ~/catkin_ws/src/unibas_turtle/src/rotate.py' and the second line is a blank prompt. The text is in a monospaced font, with the prompt 'bloisi@bloisi-U36SG:~\$' in green and the command and its output in white.

```
bloisi@bloisi-U36SG: ~  
bloisi@bloisi-U36SG:~$ chmod u+x ~/catkin_ws/src/unibas_turtle/src/rotate.py  
bloisi@bloisi-U36SG:~$
```

launch file per il nodo rotate



```
1 <launch>
2   <node name="turtlesim_node" pkg="turtlesim" type="turtlesim_node" output="screen"/>
3   <node name="rotate" pkg="unibas_turtle" type="rotate.py" output="screen"/>
4 </launch>
5
```



esecuzione per il nodo rotate

```
/home/bloisi/catkin_ws/src/unibas_turtle/launch/unibas_rotate.launch http://localhost:1
bloisi@bloisi-U36SG:~$ roslaunch unibas_turtle unibas_rotate.launch
... logging to /home/bloisi/.ros/log/f094ced0-7188-11e9-a68b-50465dde6884/roslau
nch-bloisi-U36SG-15722.log
Checking log directory for disk usage. This may take awhile.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://localhost:35672/

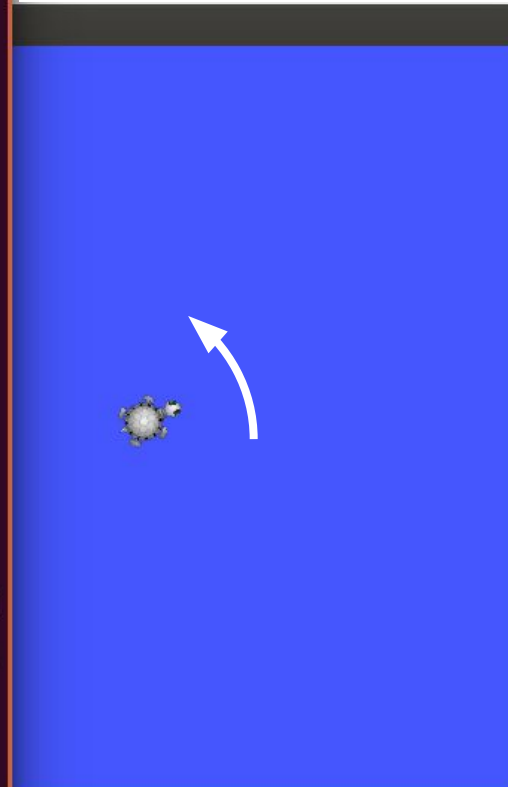
SUMMARY
=====

PARAMETERS
* /rostdistro: kinetic
* /rosversion: 1.12.14

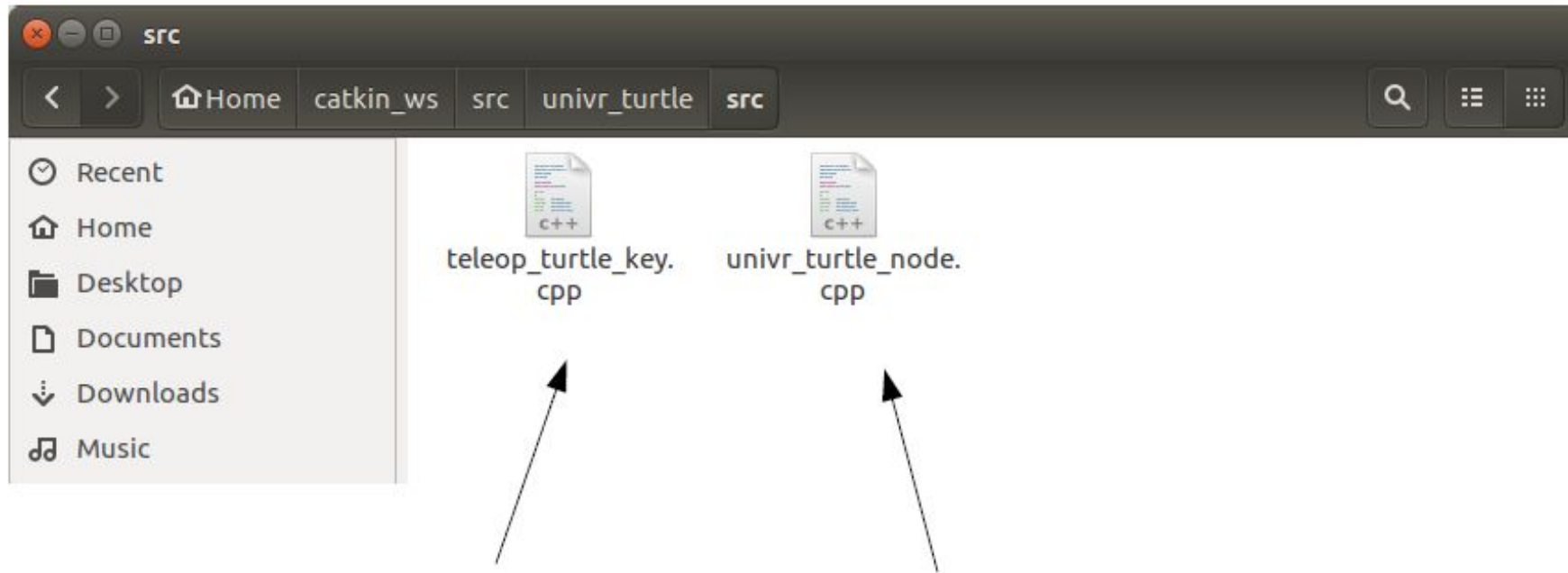
NODES
/
  rotate (unibas_turtle/rotate.py)
  turtlesim_node (turtlesim/turtlesim_node)

auto-starting new master
process[master]: started with pid [15733]
ROS_MASTER_URI=http://localhost:11311

setting /run_id to f094ced0-7188-11e9-a68b-50465dde6884
process[rosout-1]: started with pid [15746]
started core service [/rosout]
process[turtlesim_node-2]: started with pid [15753]
process[rotate-3]: started with pid [15764]
[ INFO] [1557316858.857799883]: Starting turtlesim with node name /turtlesim_nod
e
[ INFO] [1557316858.864551185]: Spawning turtle [turtle1] at x=[5,544445], y=[5,
544445], theta=[0,000000]
Let's rotate your robot
Input your speed (degrees/sec):5
Type your distance (degrees):45
Clockwise?: 0
█
```



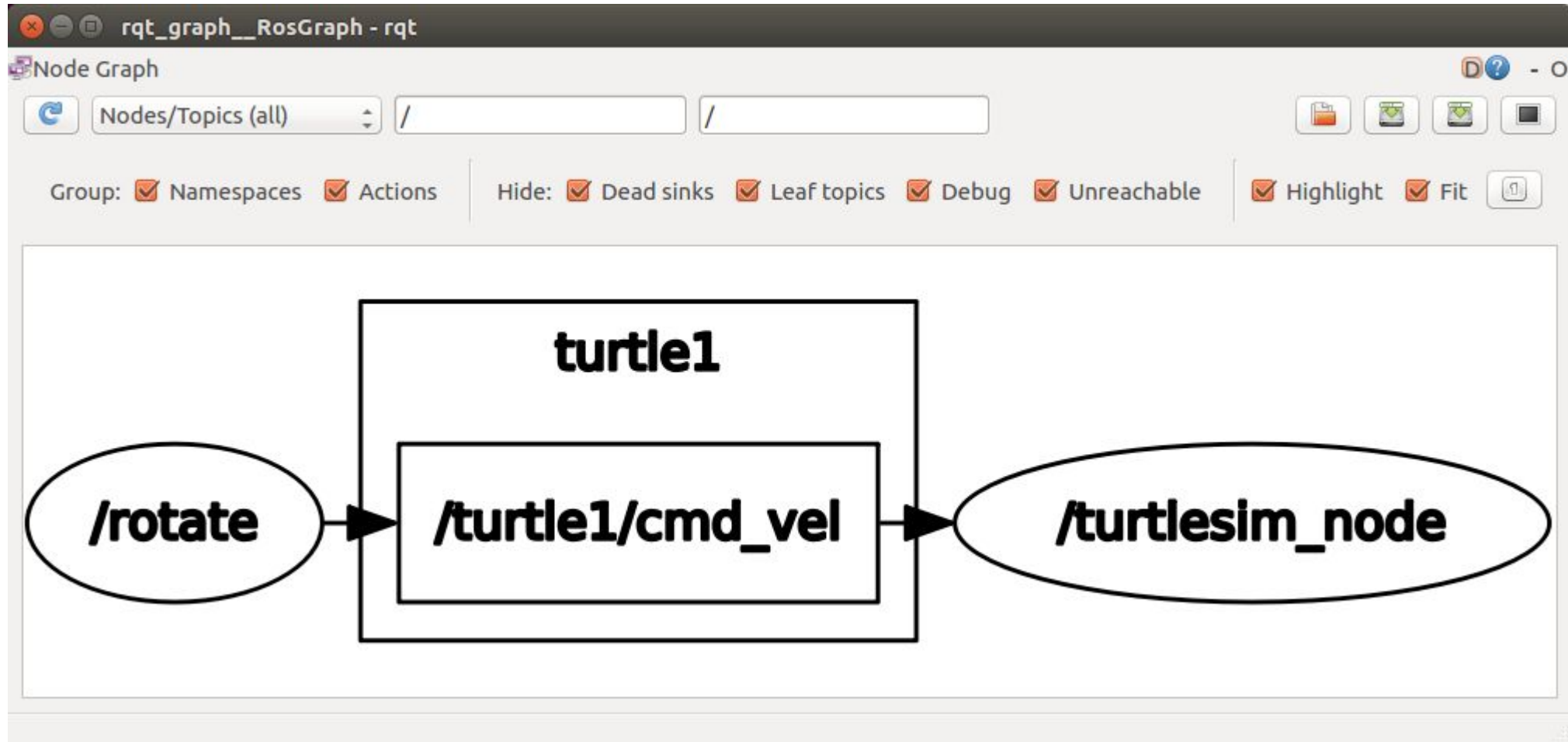
I nodi di univr_turtle



Nodo per la gestione
della teleoperazione

nodo per il
controllo in velocità
della tartaruga

rqt_graph



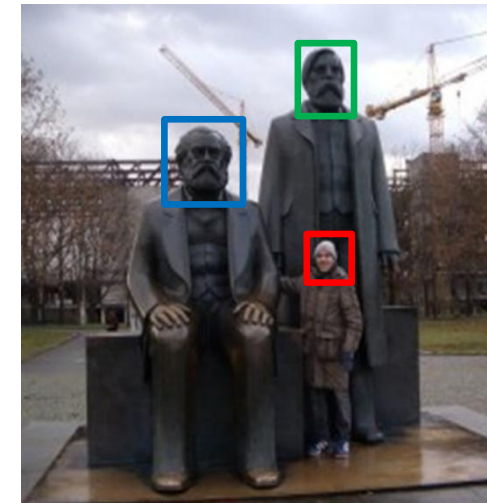
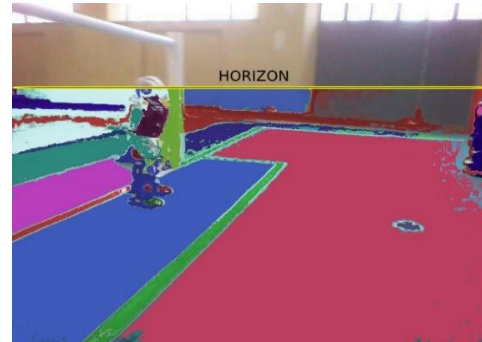
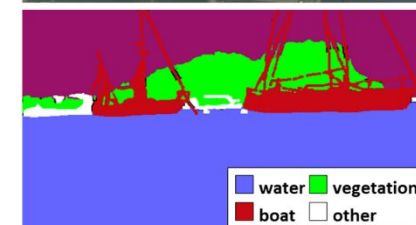


**UNIVERSITÀ DEGLI STUDI
DELLA BASILICATA**

Corso di Sistemi Informativi
A.A. 2018/19

Docente
Domenico Daniele Bloisi

ROS launch file



Maggio 2019