



EV_1_4_Ejercicio con Turtlesim.

Dinámica y control de robots.

Integrantes:

Curiel Sánchez Héctor David

Fernández Gaeta Uriel

García Camacho Jesús Alberto

Gómez Medina Jesús Carlos

Salcedo González Alondra

Ingeniería Mecatrónica 9°B

Maestro: Carlos Enrique Morán Garabito.

24 de mayo del 2019



Objetivo

Conocer los comandos básicos del nodo de Turtlesim para manipular sus movimientos en la terminal.

Materiales

- PC.
- SO con alguna distro de Linux compatible con ROS.

Procedimiento

1.- Para comenzar abrimos la terminal y escribimos:

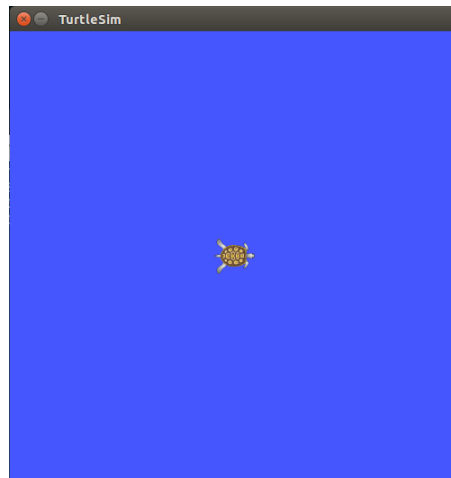
```
roscore
```

Presionamos Enter, esto lo hacemos para utilizar ROS, por lo que es necesario que esté corriendo el roscore.

2.- Luego para abrir el simulador de turtlesim ejecutamos el siguiente comando:

```
roslaunch turtlesim turtlesim_node
```

Se nos debe abrir el simulador con un turtlebot en la mitad de la pantalla del simulador.



3.- Para ver los tópicos abiertos, se ejecuta en otra terminal el siguiente comando:

```
rostopic list -v
```



Por cada turtlebot que tiene el simulador hay tres tópicos distintos que son:

- `/turtle1/cmd_vel`: recibe comandos de velocidad para mover el turtlebot.
- `/turtle1/color_sensor`: componente RGB del fondo del simulador.
- `/turtle1/pose`: posición que tiene en la imagen.

Tópicos

Los tópicos comienzan por el nombre del turtlebot que en este caso es `turtle1`.

A través del tópico `cmd_vel` pasamos el siguiente comando que enviará un mensaje de tipo `Twist` al turtlebot, de esta forma el turtlebot va a avanzar con un pequeño giro hacia la izquierda:

```
rostopic pub -1 /turtle1/cmd_vel geometry_msgs/Twist -- '[2.0,0.0,0.0]'\n'[0.0,0.0,0.3]'
```

Para darle un movimiento continuo al turtlebot podemos ejecutar:

```
rostopic pub /turtle1/cmd_vel geometry_msgs/Twist -r 1 -- '[2.0,0.0,0.0]'\n'[0.0,0.0,1.0]'
```

Como tiene una frecuencia de 1 el turtlebot no dejará de moverse.

Servicios de Turtlesim

Para obtener una lista de los servicios disponibles utilizamos el comando:

```
rosservice list
```

Para llamar los servicios utilizamos el comando:

```
rosservice call
```

Por ejemplo para llamar el servicio `spawn` pasándole como parámetros las coordenadas y un nombre podemos añadir un turtlebot adicional al simulador, escribiendo el siguiente comando:

```
rosservice call /spawn 2 2 0.5 "turtle2"
```

Para eliminar un turtlebot por su nombre podemos ejecutar el siguiente comando:

```
rosservice call /kill "turtle2"
```

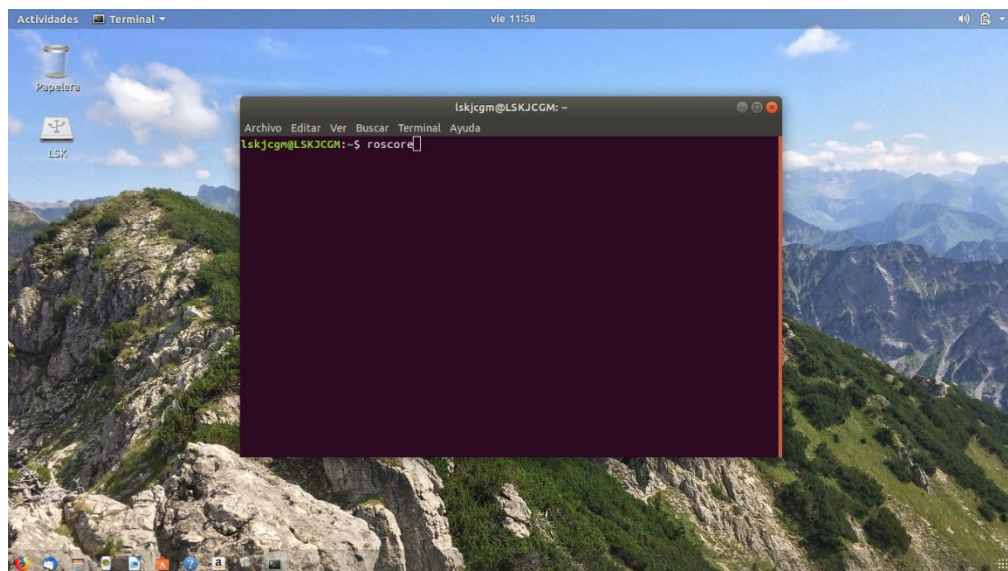
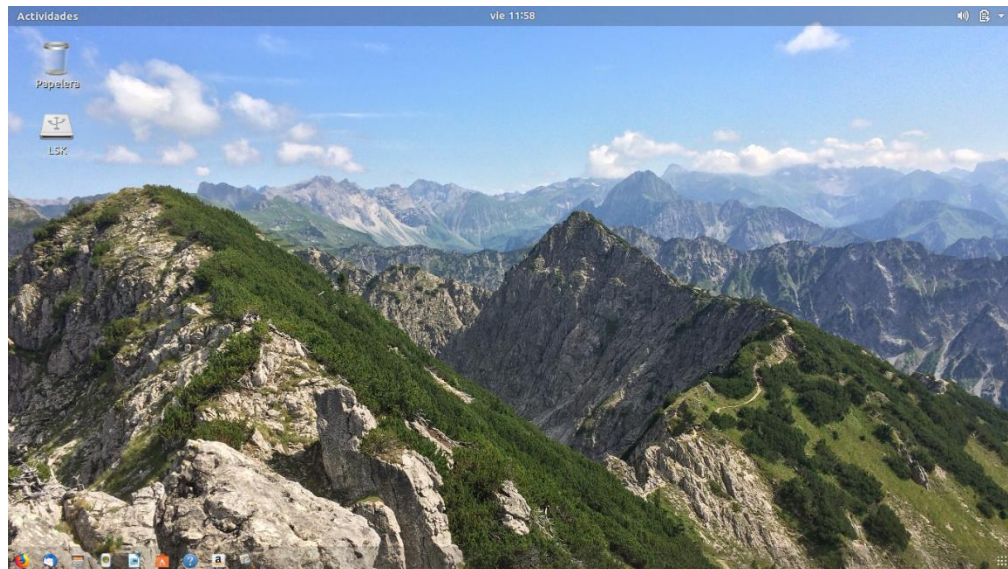


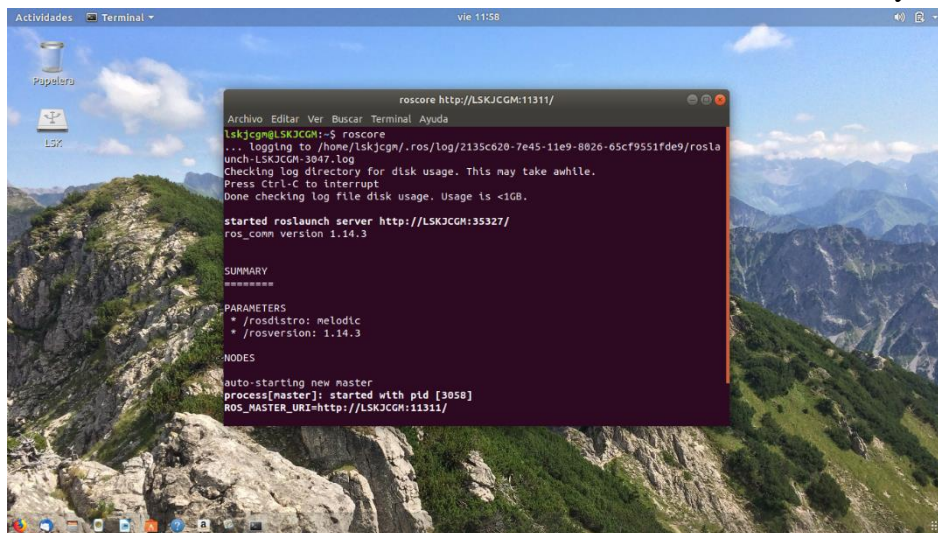
Turtlesim tiene un nodo que permite mover el turtlebot con las flechas de dirección del teclado, para ello escribimos en la terminal el siguiente comando:

```
roslaunch turtlesim turtle_teleop_key
```

Resultados

1.- Se abre una nueva terminal en Ubuntu y se ejecuta ROS.



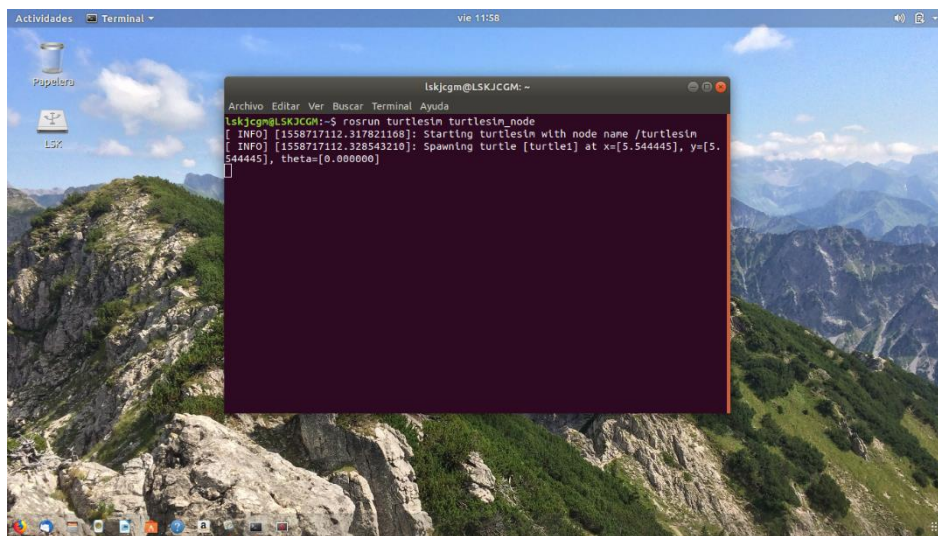


```
roscore http://LSKJCGM:11311/
lskjcg@LSKJCGM:~$ roscore
... logging to /home/lskjcg/.ros/log/2135c620-7e45-11e9-b826-65cf9551fde9/ros-launch-LSKJCGM-3047.log
Checking log directory for disk usage. This may take awhile.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://LSKJCGM:35327/
ros_comm version 1.14.3

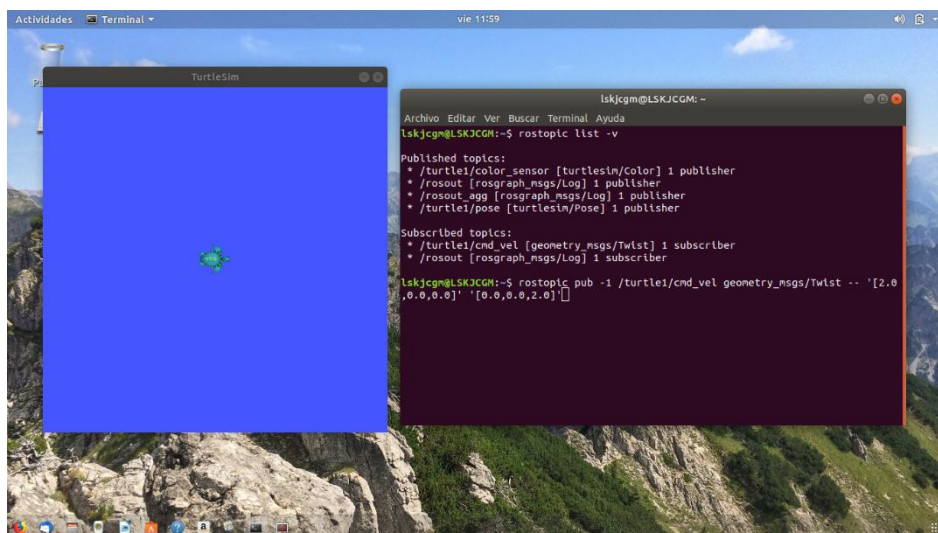
SUMMARY
=====
PARAMETERS
 * /roslistro: melodic
 * /rosversion: 1.14.3
NODES
auto-starting new master
process[master]: started with pid [3050]
ROS_MASTER_URI=http://LSKJCGM:11311/
```

2.- Se abre otra terminal y dentro de ella se ejecuta el nodo de Turtlesim.



```
lskjcg@LSKJCGM:~$ roscore
lskjcg@LSKJCGM:~$ rosrun turtlesim turtlesim_node
[ INFO] [1558717112.317821168]: Starting turtlesim with node name /turtlesim
[ INFO] [1558717112.328543210]: Spawning turtle [turtle1] at x=[5.544445], y=[5.544445], theta=[0.000000]
```

3.- En otra terminal abierta se listan los tópicos del turtlebot.

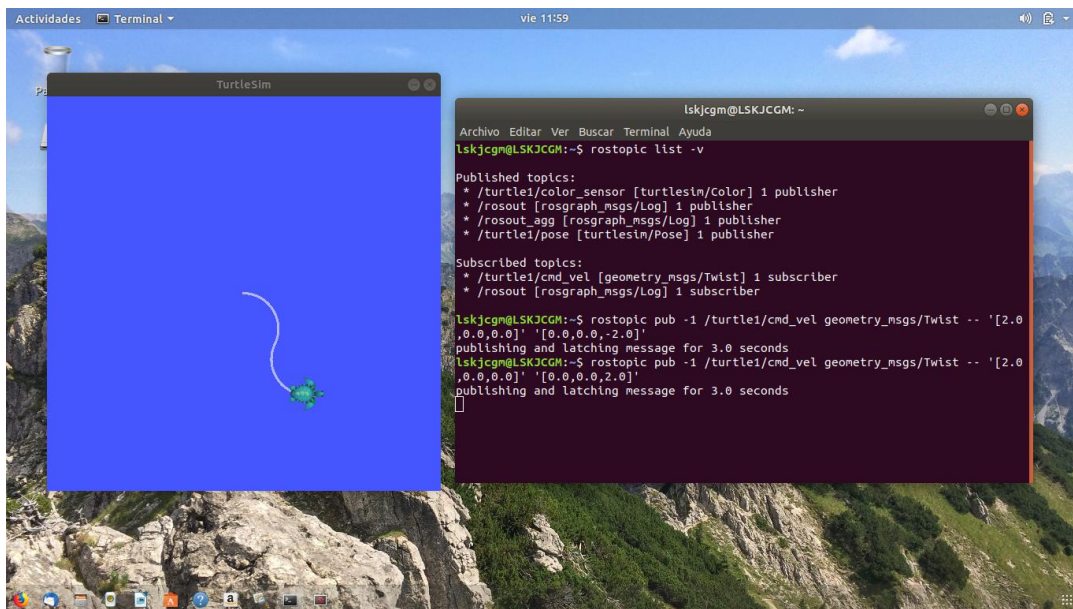
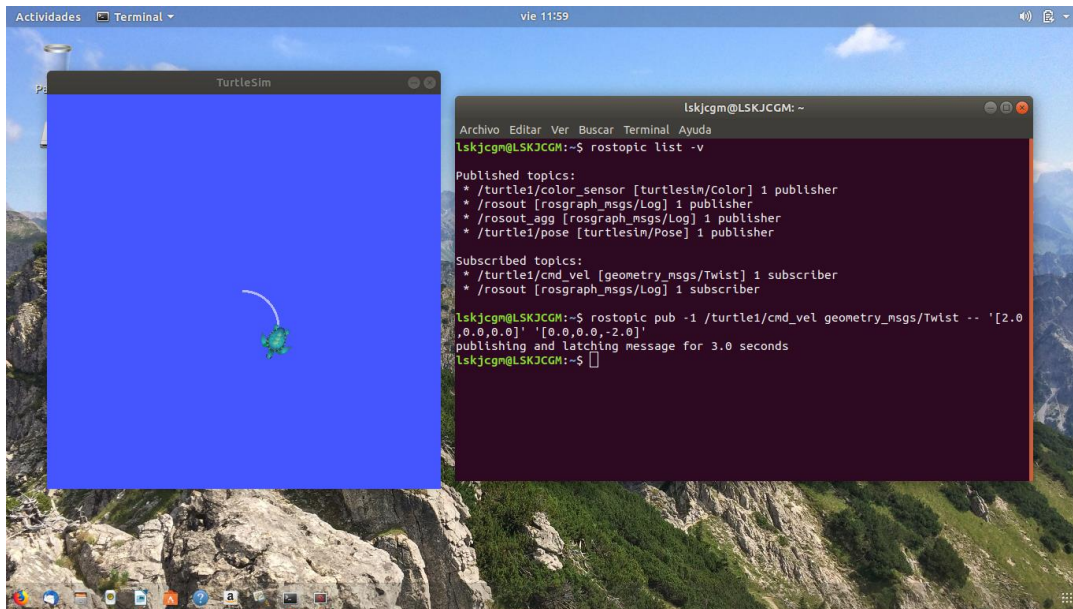


```
lskjcg@LSKJCGM:~$ rostopic list -v
Published topics:
 * /turtle1/color_sensor [turtlesim/color] 1 publisher
 * /rosout [roscpp_msgs/Log] 1 publisher
 * /rosout_agg [roscpp_msgs/Log] 1 publisher
 * /turtle1/pose [turtlesim/Pose] 1 publisher

Subscribed topics:
 * /turtle1/cmd_vel [geometry_msgs/Twist] 1 subscriber
 * /rosout [roscpp_msgs/Log] 1 subscriber

lskjcg@LSKJCGM:~$ rostopic pub -1 /turtle1/cmd_vel geometry_msgs/Twist -- '[2.0,0.0,0.0]' '[0.0,0.0,2.0]'
```

4.- Se introducen los comandos de movimiento y movemos el turtlebot.





Referencias

Jorge Juan Muñoz Morera, (2018, 28 noviembre). Turtlesim, simulador 2D para robots diferenciales en ROS. Recuperado 24 mayo, 2019, de <https://openwebinars.net/blog/turtlesim-simulador-2d-para-robots-diferenciales-en-ros/>