Fazer download de um ficheiro num servidor para o PC, com ssh:

scp username@hostname:/path/to/remote/file /path/to/local/directory

Transferir um ficheiro do PC para o servidor:

scp /path/to/local/file username@hostname:/path/to/remote/directory

Conexão e configuração TurtleBot3 Waffle Pi

1. Abrir Terminal e ligar a um dos 5 robots

ssh user@192.168.28.[11…15] (palavra-passe: user)

2. Se ainda não houver um nó *roscore*, criá-lo (nesse terminal)

roscore

3. Num novo terminal (sem fechar o outro) ligar ao mesmo robot

ssh user@192.168.28.[11…15]

4. Nesse terminal, correr

roslaunch turtlebot3_bringup turtlebot3_robot.launch

5. Num novo terminal, sem ligação ao robot (i.e., se fazer ssh) configurar o ficheiro *.bashrc* (adicionar as seguintes linhas ao ficheiro)

export TURTLEBOT3_MODEL=waffle_pi
export TURTLEBOT3_NAME=waffle5
export TURTLEBOT3_IP=192.168.28.15
export TURTLEBOT3_NUMBER=15
export ROS_MASTER_URI=http://192.168.28.15:11311
export ROS_HOSTNAME=192.168.28.115
export ROS_IP=192.168.28.115

6. Usar este último terminal para efetuar as operações pretendidas

7. No final, correr o seguinte comando num dos terminais ligados ao robot por ssh

sudo shutdown -h now

Tirar rosbags:

rosbag record -a (para todos os tópicos)
rosbag record /<topic name> (para tópicos específicos)

Conduzir robot:

roslaunch turtlebot3_teleop turtlebot3_teleop_key.launch

Correr RVIZ:

roslaunch turtlebot3_gazebo turtlebot3_gazebo_rviz.launch

GMAPPING na simulação:

1. Num terminal meu (sem ssh), fazer

roslaunch turtlebot3_gazebo turtlebot3_world.launch

2. Num novo terminal meu, fazer

roslaunch turtlebot3_teleop turtlebot3_teleop_key.launch

3. Num novo terminal meu, fazer

roslaunch turtlebot3_gazebo turtlebot3_gmapping.launch

4. Num novo terminal meu, fazer

roslaunch turtlebot3_slam turtlebot3_slam.launch slam_methods:=gmapping

OU roslaunch turtlebot3_gazebo turtlebot3_gazebo_rviz.launch (com configurações)

OU rviz (sem configurações)
e garantir que em ‘Displays’ aparece ‘Map’ e ‘RobotModel’. Nesta secção, fazer Displays → Map → Topic → /map

GMAPPING com o robot:

1. Preparar o robot (roscore e drivers)

2. Num novo terminal meu, fazer

roslaunch turtlebot3_teleop turtlebot3_teleop_key.launch

3. Num novo terminal meu, fazer

    roslaunch turtlebot3_slam turtlebot3_slam.launch slam_methods:=gmapping

4. Guardar o mapa (num novo terminal):

    rosrun map_server map_saver -f <path and name>

5. Abrir mapa:

    rosrun map_server map_server <mapa>.pgm

Para fazer com que o robot pare:

    rostopic pub /cmd_vel geometry_msgs/Twist '[0.0, 0.0, 0.0]' '[0.0, 0.0, 0.0]'

Construir mapa no RVIZ a partir de bag:

1. roscore (sem ssh; requer alteração do ROS_MASTER_URI para máquina atual, no .bashrc)

2. roslaunch turtlebot3_slam turtlebot3_slam.launch slam_methods:=gmapping

3. rosbag play --clock ~/<path and name>

4. Guardar o mapa (num novo terminal):

    rosrun map_server map_saver -f <path and name>

5. Abrir mapa:

    rosrun map_server map_server <mapa>.pgm

Transformações de referencial (pensar em *broadcaster* e *listener*)

Representações no RVIZ:

1. roscore

2. rviz

    2.1 *Add → TF*

3. rosrun tf2_ros static_transform_publisher <x> <y> <z> <yaw> <pitch> <roll> <parent_link> <child_link>

4. Escolher *Fixed Frame*

Ver a árvore de transformações (permite também visualizar matriz de rotação):

1. rosrun tf tf_echo <parent_link> <child_link>

2. rqt

    2.1 *Plugins* → *Visualization* → *TF Tree*

Recorrendo a *Lookup Transform*, podemos obter a transformação entre o 1º e o último elementos da cadeia.

Correr *script* para dados reais

1. roscore

2. rosrun mapping occ_grid_mapping.py

3. rosrun map_server map_server <map path and name.yaml>

3. AMCL

    roslaunch turtlebot3_navigation turtlebot3_navigation.launch map_file:=$HOME/<map path and name.yaml>

4. rosbag play --clock <path and name>

Outros comandos ROS

Criar *package*:

1. cd ~/catkin_ws/src/
2. catkin_create_pkg <package_name> [depend1] [depend2] [depend3]

    Normalmente, depend1 → std_msgs, depend2 → rospy, depend3 → roscpp

3. cd ~/catkin_ws/
4. catkin_make
5. . ~/catkin_ws/devel/setup.bash

Adicionar *script* de Python à *package*:

1. touch <python_script.py>

2. Alterar a primeira linha do código para (*shebang*)

    #!usr/bin/env python3

3. Noutro terminal

    roscore

4. Tornar o código executável

    chmod +x <python_script.py>

5. Correr com

    rosrun <package_name> <python_script.py>