robotcraft2017_patrol ROS Package Installation

David Portugal, Ingeniarius Ltd.

16 June 2017

1. Install Ubuntu 16.04 Linux (64 bits) - Xenial Distribution

Download the Operating System from: ftp://ftp.dei.uc.pt/pub/linux/ubuntu/releases/16.04.2/ubuntu-16.04.2-desktop-amd64.iso

and record it to a USB-stick (https://www.ubuntu.com/download/desktop/create-a-usb-stick-on-windows) or to a CD/DVD.

Boot your PC from the USB/CD/DVD.

We recommend to download the updates while installing Ubuntu and install third-party software for graphics and Wi-Fi hardware.

Make sure that you install Ubuntu in English language.

We do not recommend installing Ubuntu in a Virtual Machine, unless you have at least 4GB of RAM and 4 CPU cores dedicated to your virtual machine.

2. Run the Ubuntu updates

After a successful installation of the OS, please install the latest updates.
Open the gnome-terminal and type:

```
sudo apt upgrade
```

3. Install ROS Kinetic Kame

Follow the instructions at: http://wiki.ros.org/kinetic/Installation/Ubuntu

Please choose the desktop-full installation

4. Install dependencies for robotcraft2017_patrol

Open the gnome-terminal and type:

```
sudo apt install ros-kinetic-move-base ros-kinetic-nav-core
ros-kinetic-amcl ros-kinetic-map-server
```

5. Setup your ROS catkin workspace

In the terminal type:

```
mkdir ~/catkin_ws

cd ~/catkin_ws

wstool init src

catkin_make

source devel/setup.bash
```

Open the “.bashrc” configuration file from the terminal with:

```
gedit ~/.bashrc
```

and add the following two lines at the of the file:

```
source ~/catkin_ws/devel/setup.bash

export ROS_WORKSPACE=~/catkin_ws
```

Save the file and exit.

To test if your changes were successful type “`roscd`” in a new terminal, and your current directory will change to “`catkin_ws`”.

6. Download the robotcraft2017_patrol framework into your Workspace:

In the terminal type:

```
roscd

cd src

git clone https://github.com/ingeniarius-ltd/robotcraft2017_patrol
```

7. Compile the robotcraft2017_patrol framework in your Workspace:

In the terminal type:

```
roscd; catkin_make
```

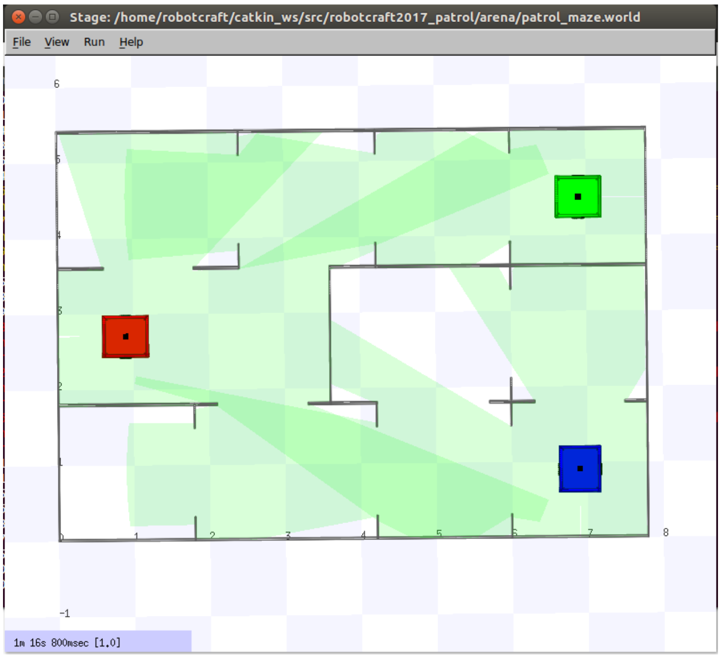
If you have carefully followed all the above steps, you should see a full compilation running in your workspace until 100%.

8. Start the robotcraft2017_patrol framework:

In the terminal type:

```
roslaunch robotcraft2017_patrol robotcraft2017.launch
```

You will see the Stage simulator starting up with 3 robots ready to patrol!


Stage: /home/robotcraft/catkin_ws/src/robotcraft2017_patrol/arena/patrol_maze.world
File
View
Run
Help
6
5
4
3
2
1
0
1
2
3
4
5
6
7
8
-1
1m 16s 800msec [1.0]

9. Run the *patrolbot* example:

The patrolbot initial example consists in sending a navigation goal to each of the 3 robots. It also exemplifies how robots can communicate by publishing data in the "/communication" topic, and subscribing data from it.

To run the example with three robots simultaneously, type in the terminal:

```
roslaunch robotcraft2017_patrol start_robots.launch
```

You will see the robots moving in the Stage simulator!

10. Controlling the behavior of each robot

The “patrolbot.cpp” code in the “src” folder of “robotcraft2017_patrol”, provides example code that you should carefully analyze.

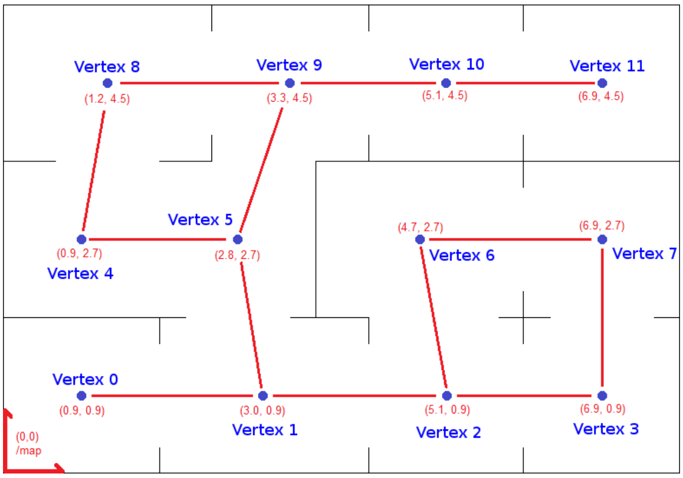
To run each robot separately, you should re-launch the robotcraft2017_patrol framework (**step 8**) and then type in separate terminals:

```
rosrun robotcraft2017_patrol patrolbot _robot_id:=0

rosrun robotcraft2017_patrol patrolbot _robot_id:=1

rosrun robotcraft2017_patrol patrolbot _robot_id:=2
```

11. Programming the behavior of the robots

Your assignment as a Robotcrafter will be to choose together with your group an effective patrolling strategy, and program the collective behavior of robots to frequently visit all points of the environment (*i.e.* all vertices of the graph).

Below you can check the 12 vertices that your robots should visit:



(“patrol_maze_graph” image in the “arena” folder)

Please modify the “patrolbot.cpp” source code to implement the desired behavior.

Remember: you will need to recompile the code with “`roscd; catkin_make`” to test your changes.

12. Useful Tips

You should install terminator to manage the multiple consoles for starting/testing the different ROS nodes:

```
sudo apt install terminator
```

In order to monitor the load of your computer, you should install the multiload indicator for Ubuntu:

```
sudo apt install indicator-multiload
```

Happy Programming!