**TURTLEBOT GUIDE**

**INSTALLATION:**

1. Assemble the Turltebot kit if it is not already assembled using the “Turtlebot Assembly Instructions” PDF
2. Install Ubuntu on two laptops (I was using version 12.10, the latest version). One laptop will be riding on the Turtlebot and the other will be the workstation PC from which the Turtlebot will be controlled from.
3. Here is the current guide (as of 3/19/2013) for installing ROS and the Turtlebot software on the two laptops. Make sure you have internet access when installing this software. The most up to date instructions can be found at: <http://ros.org/wiki/turtlebot/Tutorials/groovy/Installation>

I did have some alterations to these instructions (although these instructions may not be current, these instructions worked for me), these alterations can be found in red. The instructions are for the laptop you will be using on the Turtlebot mainly (it will specifically say Workstation if the following instructions are for the workstation)

## Installation

### Configure your Ubuntu repositories

Configure your Ubuntu repositories to allow "restricted," "universe," and "multiverse." You can [follow the Ubuntu guide](https://help.ubuntu.com/community/Repositories/Ubuntu) for instructions on doing this (if needed).

### Setup your sources.list

Setup your computer to accept software from packages.ros.org. ROS Groovy does **not** support Lucid, Maverick or Natty.

**Ubuntu 12.10 (Quantal)**

* + sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu quantal main" > /etc/apt/sources.list.d/ros-latest.list'

### Set up your keys

* wget http://packages.ros.org/ros.key -O - | sudo apt-key add -

### Installation

First, make sure your Debian package index is up-to-date:

* sudo apt-get update

There are many different libraries and tools in ROS. We provided four default configurations to get you started. You can also install ROS packages individually.

* **Desktop-Full Install: (Recommended)** : ROS, [rqt](http://ros.org/wiki/rqt), [rviz](http://ros.org/wiki/rviz), robot-generic libraries, 2D/3D simulators, navigation and 2D/3D perception
  + sudo apt-get install ros-groovy-desktop-full

### Initialize rosdep

Before you can use ROS, you will need to initialize rosdep. rosdep enables you to easily install system dependencies for source you want to compile and is required to run some core components in ROS.

sudo rosdep init

rosdep update

### Environment setup

It's convenient if the ROS environment variables are automatically added to your bash session every time a new shell is launched:

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

source ~/.bashrc

If you have more than one ROS distribution installed,*~/.bashrc*must only source the*setup.bash*for the version you are currently using.

### Getting rosinstall

[rosinstall](http://ros.org/wiki/rosinstall) is a frequently used command-line tool in ROS that is distributed separately. It enables you to easily download many source trees for ROS packages with one command.

To install this tool on Ubuntu, run:

sudo apt-get install python-rosinstall

### Preparation

In addition, you will need to install the following pre-requisites for building the create, kobuki and turtlebot stacks:

> sudo apt-get install ros-groovy-ecl-tools ros-groovy-ecl-lite ros-groovy-ecl-core ros-groovy-ecl-navigation ros-groovy-rqt-robot-plugins

> sudo apt-get install ros-groovy-zeroconf-avahi ros-groovy-multimaster-experimental ros-groovy-depthimage-to-laserscan ros-groovy-openni-camera

> sudo apt-get install ros-groovy-joystick-drivers ros-groovy-map-store

> sudo apt-get install python-wstool

### Catkin Stacks

> mkdir ~/turtlebot-catkin

> cd ~/turtlebot-catkin

> wstool init src https://raw.github.com/turtlebot/turtlebot/master/turtlebot\_catkin.rosinstall

> source /opt/ros/groovy/setup.bash

> catkin\_make -DCMAKE\_INSTALL\_PREFIX=/opt/turtlebot/groovy

> cd build; sudo make install

### Rosbuild Stacks

> rosws init ~/turtlebot-rosbuild /opt/turtlebot/groovy

> cd ~/turtlebot-rosbuild

> rosws merge https://raw.github.com/turtlebot/turtlebot/master/turtlebot.rosinstall

> rosws update

> source setup.bash

> rosmake kobuki\_desktop turtlebot turtlebot\_apps turtlebot\_viz

### Convenience

For convenience, you may wish to source your setup.sh script from your .bashrc so that your environment is ready as soon as you log in. e.g.

> echo "source ~/turtlebot-rosbuild/setup.bash" >> ~/.bashrc

## Network Time Protocol

Clock synchronization is important for ROS. Chrony has been found to be the best ntp client over lossy wireless.

1. Install Chrony
   * sudo apt-get install chrony
2. manually sync NTP
   * sudo ntpdate ntp.ubuntu.com

### Source Installation

The software currently uses environment variables to determine the hardware configuration. Export the following variables before you source setup.bash.

> echo "export TURTLEBOT\_BASE=create" >> ~/turtlebot-rosbuild/setup.sh

> echo "export TURTLEBOT\_STACKS=circles" >> ~/turtlebot-rosbuild/setup.sh

> echo "export TURTLEBOT\_3D\_SENSOR=kinect" >> ~/turtlebot-rosbuild/setup.sh

> source ~/turtlebot-rosbuild/setup.bash

**You will also need to install SSH capabilities on the Turtlebot laptop:**

sudo apt-get install openssh-server

## Workstation Installation (Now we are installing the software on the PC that will be controlling the Turtlebot)

## Overview

## Network Time Protocol

1. Install Chrony
   * sudo apt-get install chrony
2. manually sync NTP
   * sudo ntpdate ntp.ubuntu.com

### Configure your Ubuntu repositories

Configure your Ubuntu repositories to allow "restricted," "universe," and "multiverse." You can [follow the Ubuntu guide](https://help.ubuntu.com/community/Repositories/Ubuntu) for instructions on doing this.

### Setup your sources.list

Setup your computer to accept software from packages.ros.org. ROS Groovy does **not** support Lucid, Maverick or Natty.

**Ubuntu 12.10 (Quantal)**

* + sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu quantal main" > /etc/apt/sources.list.d/ros-latest.list'

### Set up your keys

* wget http://packages.ros.org/ros.key -O - | sudo apt-key add -

### Installation

First, make sure your Debian package index is up-to-date:

* sudo apt-get update

There are many different libraries and tools in ROS. We provided four default configurations to get you started. You can also install ROS packages individually.

* **Desktop-Full Install: (Recommended)** : ROS, [rqt](http://ros.org/wiki/rqt), [rviz](http://ros.org/wiki/rviz), robot-generic libraries, 2D/3D simulators, navigation and 2D/3D perception
  + sudo apt-get install ros-groovy-desktop-full

or [click here](apt:ros-groovy-desktop-full?refresh=yes)

### Initialize rosdep

Before you can use ROS, you will need to initialize rosdep. rosdep enables you to easily install system dependencies for source you want to compile and is required to run some core components in ROS.

sudo rosdep init

rosdep update

### Environment setup

It's convenient if the ROS environment variables are automatically added to your bash session every time a new shell is launched:

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

source ~/.bashrc

If you have more than one ROS distribution installed,*~/.bashrc*must only source the*setup.bash*for the version you are currently using.

### Getting rosinstall

[rosinstall](http://ros.org/wiki/rosinstall) is a frequently used command-line tool in ROS that is distributed separately. It enables you to easily download many source trees for ROS packages with one command.

To install this tool on Ubuntu, run:

sudo apt-get install python-rosinstall

In addition, you will need to install the following debs for turtlebot (please update this if you find any errors):

> sudo apt-get install ros-groovy-turtlebot ros-groovy-turtlebot-apps ros-groovy-turtlebot-viz

One last step if you have a kobuki base, you'll need to add kobuki's udev rules (you'll need your sudo password):

> . /opt/ros/groovy/setup.bash

> rosrun kobuki\_ftdi create\_udev\_rules

### Convenience

For convenience, you may wish to source your setup.sh script from your .bashrc so that your environment is ready as soon as you log in. e.g.

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

If you have overlaid your own development environment on top of the installed debs, point to that setup script instead.

## Network Configuration – MAKE SURE BOTH LAPTOPS ARE ON THE SAME NETWORK

**Description:** Get turtlebot and your workstation chatting to each other.  
  
**Tutorial Level:** INTERMEDIATE

ROS requires bidirectional networking between all computers attached to the network and does not have security built in. Using a VPN is recommended.

For this tutorial you must know the IP of the netbook on TurtleBot (IP\_OF\_TURTLEBOT). You will also need the IP of your personal computer/workstation (IP\_OF\_WORKSTATION). If the hostname is properly resolved on both computers, you don't need to set this.

Ultimately you will need to configure ROS\_MASTER\_URI and ROS\_HOSTNAME correctly with these ip addresses to ensure the ros communication channels can find each other.

## Determining IP addresses

### IfConfig

Get the ip addresses of both your turtlebot pc and the workstation pc using the following:

In a terminal, type

* ifconfig

You will see something like:

* lo Link encap:Local Loopback
* inet addr:127.0.0.1 Mask:255.0.0.0
* inet6 addr: ::1/128 Scope:Host
* UP LOOPBACK RUNNING MTU:16436 Metric:1
* RX packets:6658055 errors:0 dropped:0 overruns:0 frame:0
* TX packets:6658055 errors:0 dropped:0 overruns:0 carrier:0
* collisions:0 txqueuelen:0
* RX bytes:587372914 (587.3 MB) TX bytes:587372914 (587.3 MB)
* wlan1 Link encap:Ethernet HWaddr 48:5d:60:75:58:90
* inet addr:**10.0.129.17** Bcast:10.0.129.255 Mask:255.255.254.0
* inet6 addr: fe80::4a5d:60ff:fe75:5890/64 Scope:Link
* UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
* RX packets:101983 errors:0 dropped:0 overruns:0 frame:0
* TX packets:37244 errors:0 dropped:0 overruns:0 carrier:0
* collisions:0 txqueuelen:1000
* RX bytes:49326141 (49.3 MB) TX bytes:7588044 (7.5 MB)
* the network interface for the wireless card is wlan1
* the IP address of the computer is 10.0.129.17

### Confirm Connectivity

Ensure that you can ping between machines using the following:

Ping from the laptop to the workstation and back by IP, or fully resolved hostname. From TurtleBot laptop ping workstation/desktop using IP\_OF\_WORKSTATION

* ping IP\_OF\_WORKSTATION

**Note:** Replace IP\_OF\_TURTLEBOT with actual IP address of TurtleBot that was determined in previous step.

From workstation ping TurtleBot using IP\_OF\_TURTLEBOT

* ping IP\_OF\_TURTLEBOT

## Turtlebot Setup (On the Turtlebot laptop)

### Source Installation

You should export the variables inside your workspace setup script.

> echo export ROS\_MASTER\_URI=http://IP\_OF\_TURTLEBOT:11311 >> ~/rosbuild-catkin/setup.sh

> echo export ROS\_HOSTNAME=IP\_OF\_TURTLEBOT >> ~/rosbuild-catkin/setup.sh

## Workstation Setup (On the Workstation laptop)

You should export the variables inside your workspace setup script. Note that the meaning of the ROS\_MASTER\_URIchanges here - the master is in the turtlebot!

> echo export ROS\_MASTER\_URI=http://IP\_OF\_TURTLEBOT:11311 >> .bashrc

> echo export ROS\_HOSTNAME=IP\_OF\_WORKSTATION >> .bashrc

## I had to run *source .bashrc* on both the workstation laptop for the environment variables to update

## Verification

The following section is not strictly necessary. However, if there is a problem with the ROS networking setup between the TurtleBot and workstation, it will be easier to identify it early.

Be sure to relaunch turtlebot if you have manually set these variables.

### Verify from TurtleBot to ROS master

On TurtleBot laptop, make sure it can contact ROS master by running:

> rostopic list

If you get the following error:

ERROR: Unable to communicate with master!

If you see this, your turtlebot has not yet been started - refer to the [TurtleBot Bringup](http://ros.org/wiki/turtlebot_bringup/Tutorials/groovy/TurtleBot%20Bringup) tutorial.

Also make sure ROS\_MASTER\_URI is set correctly

> echo $ROS\_MASTER\_URI

### Verify between ROS nodes on TurtleBot

In a **new** terminal on TurtleBot laptop run:

> rostopic echo /diagnostics

If you don't get a response a warning like

WARNING: topic [/diagnostics] does not appear to be published yet

Check that ROS\_HOSTNAME is set correctly on TurtleBot laptop.

> echo $ROS\_HOSTNAME

### Verify from workstation to TurtleBot

Open a **new** command line terminal on workstation and run:

> rostopic list

If you don't see list of topics check the value of ROS\_MASTER\_URI.

On workstation run:

> rostopic echo /diagnostics

If you don't get a warning that topic has not been published, then verify that ROS\_HOSTNAME is set correctly on the TurtleBot laptop.

### Verify from TurtleBot to Workstation

Finally, check that TurtleBot laptop can get data from ROS node running on workstation.

On workstation run:

> rostopic pub -r10 /hello std\_msgs/String "hello"

On TurtleBot run

> rostopic echo /hello

The message "hello" begin printed about 10 times a second. If not, check the ROS\_HOSTNAME setting on the workstation computer.

## Turtlebot Bringup – AFTER ALL INSTALLATION IS DONE

Close the lid of your TurtleBot laptop, place it on the TurtleBot and connect the USB cables. Press the power button for your mobile base. If you have a **Create**, the power button is on the top left, hidden behind the edge of the lowest deck and a green light will turn on. If you have a **Kobuki**, the power button is a switch on the left hand side and it will chirp and flash led's excitedly as soon as you turn it on.

**NOTE: ALL OF THE FOLLOWING COMMANDS ARE RUN FROM THE WORKSTATION COMPUTER**

Firstly ssh into your TurtleBot computer from your workstation computer – **NEW TERMINAL**

ssh turtlebot@IP\_OF\_TURTLEBOT

The format for SSH is userName@IP\_OF\_TURTLEBOT

The instructions then differ depending on whether you have a prepared usb derived release, from debs, or from source.

I had to run the following command to set permissions on the USB port that the robot is plugged into (everything has to be connected and turned on before running this (and the following) commands).

Sudo chmod 777 /dev/USB01

### Source Installation Instructions

### NEW TERMINAL ON WORKSTATION

> source ~/turtlebot-rosbuild/setup.bash

> roslaunch turtlebot\_bringup minimal.launch

### Initial Bringup

If your turtlebot came with a netbook or a USB drive installer, the turtlebot software may already be running on the turtlebot netbook. You can check by running rosnode list on the netbook. If you get a list of nodes (instead of an error) ROS is already running. Note that ROS will only start if the netbook is connected to the network - if ROS is notrunning check this first.

Otherwise, you may see something like this:

> rosnode list

ERROR: Unable to communicate with master!

If this is the case you can manually start the turtlebot ROS nodes from a terminal on the netbook.

> roslaunch turtlebot\_bringup minimal.launch

## Workstation Bringup

### Preparation

This assumes you have finished the [workstation install](http://ros.org/wiki/turtlebot/Tutorials/groovy/Workstation%20Installation) and also the [network configuration](http://ros.org/wiki/turtlebot/Tutorials/groovy/Network%20Configuration) that enables turtlebot and workstation to communicate.

### Dashboard

/!\ To see what is happening on the TurtleBot please keep the turtlebot\_dashboard up as much as possible when running the robot.

And the following command to launch the Create dashboard. – **NEW TERMINAL**

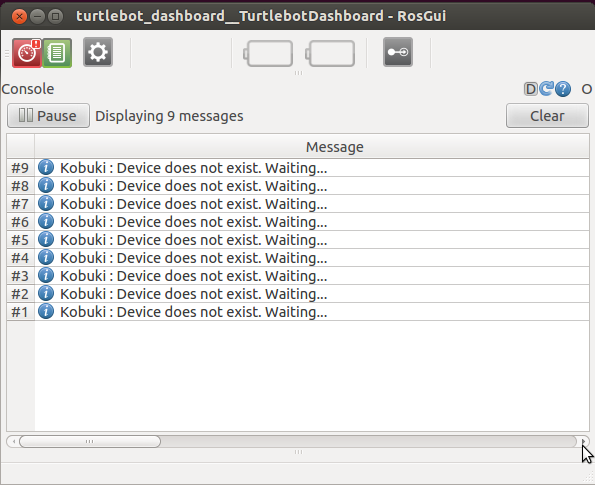
> roslaunch turtlebot\_dashboard turtlebot\_dashboard.launch

You should see the TurtleBot dashboard appear.

**I HAD TO USE THE FOLLOWING COMMAND TO BRING UP THE DASHBOARD (this is run from the workstation computer):**

**Rosrun rqt\_gui rqt\_gui –s “create\_dashboard”**

**When that screen comes up, put the robot in “Full mode” by clicking on the settings icon and clicking “Full mode:”**



## Keyboard Teleop

**Description:** Keyboard teleoperation of a turtlebot.  
  
**Keywords:** keyboard teleop  
  
**Tutorial Level:** BEGINNER  
  
**Next Tutorial:** [Joystick Teleop](http://ros.org/wiki/turtlebot_teleop/Tutorials/Joystick%20Teleop) 

This assumes that you have a TurtleBot which has already been brought up in this tutorial. See [TurtleBot Bringup](http://ros.org/wiki/turtlebot_bringup/Tutorials/TurtleBot%20Bringup) if you have not brought up the TurtleBot.

## On the TurtleBot

**NEW TERMINAL**

roslaunch turtlebot\_teleop keyboard\_teleop.launch

Key presses in this terminal can now be used to control the robot.

## SLAM Map Building with TurtleBot

**Description:** How to generate a map using gmapping  
  
**Tutorial Level:** BEGINNER  
  
**Next Tutorial:** [Autonomous Navigation of a Known Map with TurtleBot](http://ros.org/wiki/turtlebot_navigation/Tutorials/Autonomously%20navigate%20in%20a%20known%20map)

### On the TurtleBot Open a new terminal on the workstation computer and ssh into the turtlebot laptop

* Run the gmapping demo app

roslaunch turtlebot\_navigation gmapping\_demo.launch

### On your Workstation

This assumes you have ROS on your workstation and ROS\_MASTER\_URI has been set to point to your turtlebot.

* Launch rviz:

roslaunch turtlebot\_rviz\_launchers view\_navigation.launch

### On the TurtleBot

Drive the robot around using telop (keyboard)

* Save the map to file:

rosrun map\_server map\_saver -f /tmp/my\_map

**Note:** Do not close the gmapping launch until after saving the map.

* After it says "done" press Ctrl-C [#4843](https://code.ros.org/trac/ros-pkg/ticket/4843)

**CLOSE THE TERMINAL**

## Autonomous Navigation of a Known Map with TurtleBot

**Description:** This tutorial describes how to use the TurtleBot with a previously known map.  
  
**Tutorial Level:** BEGINNER

## Prior Setup

This tutorial assumes you have a map of your work area setup. Such as the one generated by the previous tutorial.

This assumes that you have a TurtleBot which has already been brought up in the [turtlebot bringup tutorials](http://ros.org/wiki/turtlebot_bringup/Tutorials).

## Launch the amcl app

### On the WORKSTATION (ssh into the Turtlebot)

Run the navigation demo app passing in your generated map file.

* roslaunch turtlebot\_navigation amcl\_demo.launch map\_file:=/tmp/my\_map.yaml

### On your Workstation

This assumes you have ROS on your workstation and ROS\_MASTER\_URI has been set to point to your turtlebot.

Launch rviz:

* # Groovy or later
* roslaunch turtlebot\_rviz\_launchers view\_navigation.launch

### In RVIZ

#### Localize the TurtleBot

When starting up, the TurtleBot does not know where it is. To provide him its approximate location on the map:

1. Click the "2D Pose Estimate" button
2. Click on the map where the TurtleBot approximately is and drag in the direction the TurtleBot is pointing.

You will see a collection of arrows which are hypotheses of the position of the TurtleBot. The laser scan should line up approximately with the walls in the map. If things don't line up well you can repeat the procedure.

#### Send a navigation goal

With the TurtleBot localized, it can then autonomously plan through the environment.

To send a goal:

1. Click the "2D Nav Goal" button
2. Click on the map where you want the TurtleBot to drive and drag in the direction the TurtleBot should be pointing at the end.

This can fail if the path or goal is blocked.

If you want to stop the robot before it reaches it's goal, send it a goal at it's current location.