1. Getting started with Ubuntu 16.04 in VirtualBox

***Note that for this course, we will be using ROS Kinetic, which is deemed more stable and more established as compared to other versions when using the TB3 and OM systems. Hence, please take note that it is strictly required to install Ubuntu version 16.04, as this version can better support ROS Kinetic.**

VirtualBox installation

1. First we need to download VirtualBox (**v6.1.4; to be safe**) from https://www.virtualbox.org/wiki/Download_Old_Builds_6_1
2. Run the executable and follow the prompts to complete the installation.

Create an Ubuntu virtual machine

1. Download the latest Ubuntu release from http://releases.ubuntu.com/16.04/?_ga=2.245247622.286617492.1585822024-1634746839.1585822024. Download Ubuntu version 16.04.6 LTS (Xenial Xerus); click on 64-bit PC (AMD64) desktop image to download desktop image
2. Open VirtualBox and click **New**
3. Type the Name for the virtual machine, like Ubuntu 16. VirtualBox will try to predict the Type and Version based on the name you enter. Otherwise, select:
    - Type: Linux
    - Version: Ubuntu (64-bit)

   and click **Next**.
4. Next we need to specify how much memory to allocate the virtual machine. According to the Ubuntu system requirements we need 2GB, but ***I'd recommend more if your host can handle it***. Basically the higher you can set the memory without severely impacting your host machine, the better the performance of the guest machine. If you're not sure, stick with 2048MB (i.e. 2 GB)
5. On the Hardware screen select **Create a virtual hard disk now** and click **Create**
6. Accept the default option **VDI** for **Hard disk file type** and click **Next**
7. Next we are prompted for Storage on physical hard disk. The options are Dynamically allocated and Fixed size. We'll use the default of **Dynamically allocated**. Click **Next**
8. Choose the hard disk size and storage location. The Ubuntu system requirements recommend 25GB; change to 25GB. Remember, we choose Dynamically allocated as our storage option in the last step, so we won't consume all this disk space immediately. Rather, VirtualBox will allocate it as required, up to the maximum 25GB we specified. Click **Create**
9. The wizard will finish and we are returned to the main VirtualBox window. Click **Settings**
10. In the left pane select **Storage**, then in the right select the CD icon with the word *Empty* beside it
11. Under *Attributes* click the CD icon and select **Choose a disk file** and browse to the downloaded file ubuntu-16.04.6-desktop-amd64.iso
12. Click **OK** to close the *Settings* dialog window. The virtual machine should now be ready to start

Install Ubuntu

In VirtualBox your VM should be showing as *Powered Off*, and the optical drive configured to point to the Ubuntu ISO file we downloaded previously.

1. In VirtualBox, click **Start**. VirtualBox will launch a new window with the vm and boot from the iso
2. Click **Start** and you will be taken to an installation window
3. Click **Install Ubuntu**
4. Select Download updates while installing Ubuntu and click Continue
5. On the next screen accept the default of Erase disk and install Ubuntu and click Install Now
6. You will be prompted with a warning saying the changes will be written to disk. Click **Continue**
7. Select your timezone and click Continue
8. Select your keyboard layout. I accepted the default of English (US) and click Continue
9. Enter a username and password, then click Continue
10. The Ubuntu installation may take several minutes to run
11. When the installation is finished you will be prompted to restart. Save and close anything else you may have open and click **Restart Now**
12. When the vm reboots, press Enter in the vm to proceed
13. If all went well the VM should boot to the Ubuntu login screen. Enter your password to continue
14. Ubuntu should run normally in the VirtualBox environment. If everything is far too small, you can adjust the screen by selecting ***View > Scaled Mode***

2. Instructions for ROS and Relevant Packages Setup

Refer to README for command instructions (easier to copy and paste) to install ROS and other packages

The following script will allow you to simplify the ROS installation procedure. Run the following command in a terminal window. The terminal application can be found with the Ubuntu search icon on the top left corner of the screen, or you can use shortcut key for terminal is `Ctrl-Alt-T`. After install ROS, please reboot Remote PC.

- `$ sudo apt-get update`
- `$ sudo apt-get upgrade`
- `$ wget https://raw.githubusercontent.com/ROBOTIS-GIT/robotis_tools/master/install_ros_kinetic.sh && chmod 755 ./install_ros_kinetic.sh && bash ./install_ros_kinetic.sh`

***Note that if sudo upgrade cannot work, type command:**

- `$ sudo rm /var/lib/apt/lists/lock`
- `$ sudo rm /var/cache/apt/archives/lock`
- `$ sudo rm /var/lib/dpkg/lock`
- `$ sudo dpkg --configure -a`

***Note that if bash cannot work, type command:**

- `$ /bin/cp /etc/skel/.bashrc ~/`

Installing Dependent ROS Packages for TB3 control on Remote PC

The next step is to install dependent packages for TurtleBot3 control on Remote PC:

- `$ sudo apt-get install ros-kinetic-joy ros-kinetic-teleop-twist-joy ros-kinetic-teleop-twist-keyboard ros-kinetic-laser-proc ros-kinetic-rgbd-launch ros-kinetic-depthimage-to-laserscan ros-kinetic-rosserial-arduino ros-kinetic-rosserial-python ros-kinetic-rosserial-server ros-kinetic-rosserial-client ros-kinetic-rosserial-msgs ros-kinetic-amcl ros-kinetic-map-server ros-kinetic-move-base ros-kinetic-urdf ros-kinetic-xacro ros-kinetic-compressed-image-transport ros-kinetic-rqt-image-view ros-kinetic-gmapping ros-kinetic-navigation ros-kinetic-interactive-markers`
- `$ cd ~/catkin_ws/src/`
- `$ git clone https://github.com/ROBOTIS-GIT/turtlebot3_msgs.git`
- `$ git clone https://github.com/ROBOTIS-GIT/turtlebot3.git`
- `$ cd ~/catkin_ws && catkin_make`

***Ensure that catkin make runs successfully. Note that if catkin cannot work, run the following commands in the terminal window:**

- `$ sudo apt install catkin`
- `$ source /opt/ros/kinetic/setup.bash`
- `$ source ~/catkin_ws/devel/setup.bash`
- `$ cd ~/catkin_ws && catkin_make`

***If catkin still cannot work, run:**

- `$ gedit ~/.bashrc`

Under ~/.bashrc, add/edit the following lines wherever applicable (i.e. if they do not exist):

```
# Set ROS Kinetic
source /opt/ros/kinetic/setup.bash
source ~/catkin_ws/devel/setup.bash

# Set ROS Network
export ROS_HOSTNAME=xxx.xxx.xxx.xxx
export ROS_MASTER_URI=http://${ROS_HOSTNAME}:11311

# Set ROS alias command
alias cw='cd ~/catkin_ws'
alias cs='cd ~/catkin_ws/src'
alias cm='cd ~/catkin_ws && catkin_make'
```

After which, save, exit the bashrc and type the following command in the terminal:

`$ source ~/.bashrc`

Installing Dependent ROS Packages for OM control on Remote PC

Install dependent packages for OpenMANIPULATOR-X. Run the following commands in a terminal window:

- `$ sudo apt-get install ros-kinetic-ros-controllers ros-kinetic-gazebo* ros-kinetic-moveit* ros-kinetic-industrial-core`
- `$ cd ~/catkin_ws/src/`
- `$ git clone https://github.com/ROBOTIS-GIT/DynamixelSDK.git`
- `$ git clone https://github.com/ROBOTIS-GIT/dynamixel-workbench.git`
- `$ git clone https://github.com/ROBOTIS-GIT/dynamixel-workbench-msgs.git`
- `$ git clone https://github.com/ROBOTIS-GIT/open_manipulator.git`
- `$ git clone https://github.com/ROBOTIS-GIT/open_manipulator_msgs.git`
- `$ git clone https://github.com/ROBOTIS-GIT/open_manipulator_simulations.git`
- `$ git clone https://github.com/ROBOTIS-GIT/robotis_manipulator.git`
- `$ cd ~/catkin_ws && catkin_make`

Installing Simulation Packages for TB3 on Remote PC

- `$ cd ~/catkin_ws/src/`
- `$ git clone https://github.com/ROBOTIS-GIT/turtlebot3_simulations.git`
- `$ cd ~/catkin_ws && catkin_make`

Installing SLAM packages for TB3 on Remote PC

- `$ sudo apt-get install ros-kinetic-frontier-exploration ros-kinetic-navigation-stage`

Installing Manipulation Packages for TB3 on Remote PC

- `$ cd ~/catkin_ws/src/`
- `$ git clone https://github.com/ROBOTIS-GIT/turtlebot3_manipulation.git`
- `$ git clone https://github.com/ROBOTIS-GIT/turtlebot3_manipulation_simulations.git`
- `$ cd ~/catkin_ws && catkin_make`

Upgrading Gazebo to version 7.16

Follow the following steps to update your Gazebo to version 7.16.0:

- `$ sudo sh -c 'echo "deb http://packages.osrfoundation.org/gazebo/ubuntu-stable` `lsb_release -cs` `main" > /etc/apt/sources.list.d/gazebo-stable.list'`
- `$ wget http://packages.osrfoundation.org/gazebo.key -O - | sudo apt-key add -`
- `$ sudo apt-get update`
- `$ sudo apt-get install gazebo7 -y`

Installing AutoRace and Other Dependent Packages for TB3 on Remote PC

- `$ cd ~/catkin_ws/src/`
- `$ git clone https://github.com/ROBOTIS-GIT/turtlebot3_autorace.git`
- `$ cd ~/catkin_ws && catkin_make`
- `$ sudo apt-get install ros-kinetic-image-transport ros-kinetic-cv-bridge ros-kinetic-vision-opencv python-opencv libopencv-dev ros-kinetic-image-proc`

3. Instructions for Basic Testing of ROS and Other Packages to Ensure Installation is Right

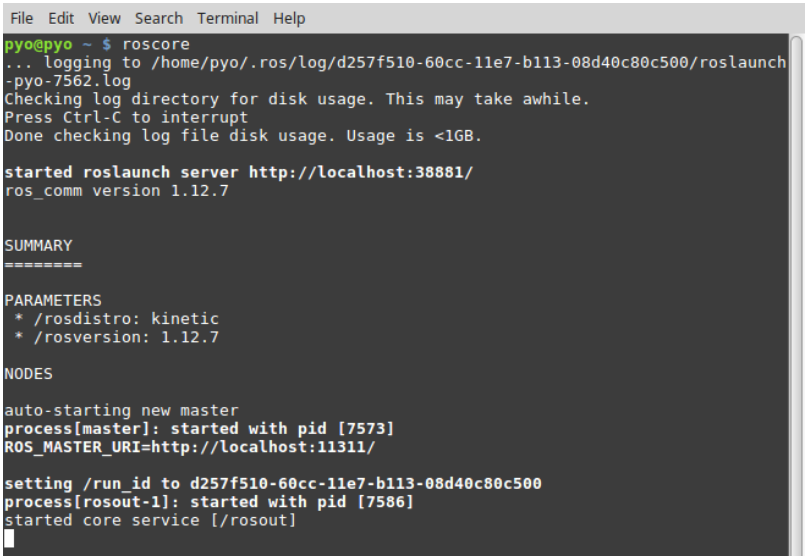
Testing of ROS

***Note that if you have any issues in these steps, please reinstall ROS from scratch.**

To test if ROS is installed properly and works correctly; Steps:

1. Open a new terminal window (*Ctrl + Alt + t*) and **enter $ `roscore`**

*roscore is the command that runs the ROS master (aka server)



2. **Run turtlesim_node** in the turtlesim package

Open a new terminal window and enter:

```
$ rosrun turtlesim turtlesim_node
```

*rosrun is the basic execution command of ROS; used to run a single node in the package

A window will pop out with a turtle in the middle:

3. **Run turtle_teleop_key** in the turtlesim package

Open a new terminal window and enter:

```
$ rosrun turtlesim turtle_teleop_key
```

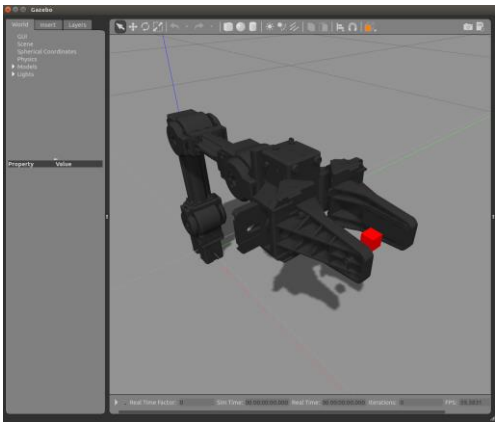
You can use any of arrow keys on the keyboard (←, → , ↑, ↓) to move the turtle

Testing of Gazebo

***Note that for the very first time you launch any Gazebo world, the system will take a while. Be patient and allow time for the system to load the world. Ensure also that the robot model (e.g. OM or TB3) is loaded within the Gazebo world.**

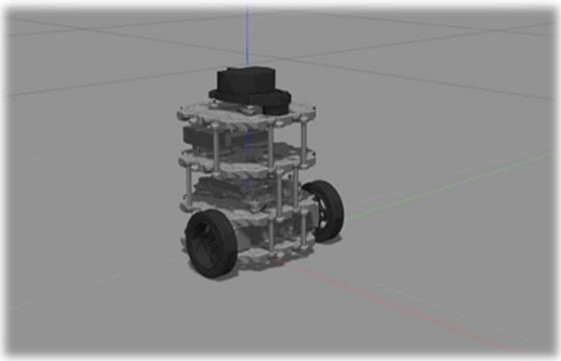
Loading OM in Empty World; type command in terminal:

- `$ roslaunch open_manipulator_gazebo open_manipulator_gazebo.launch`
- This will load OpenManipulator-X on Gazebo simulator within the Empty World; example shown in picture below:



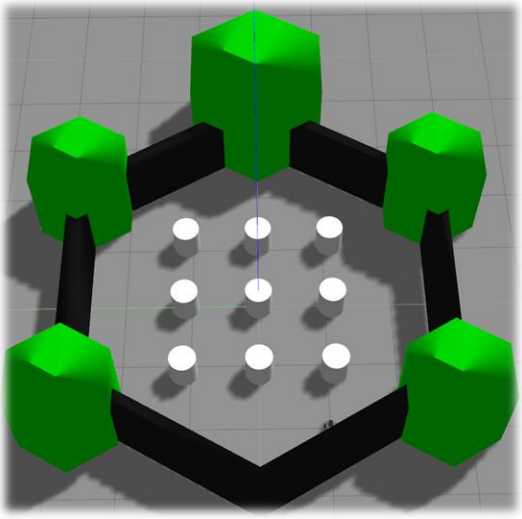
Loading TB3 Burger in Empty World; type command in terminal:

- `$ export TURTLEBOT3_MODEL=burger`
- `$ roslaunch turtlebot3_gazebo turtlebot3_empty_world.launch`
- Example shown in picture below:



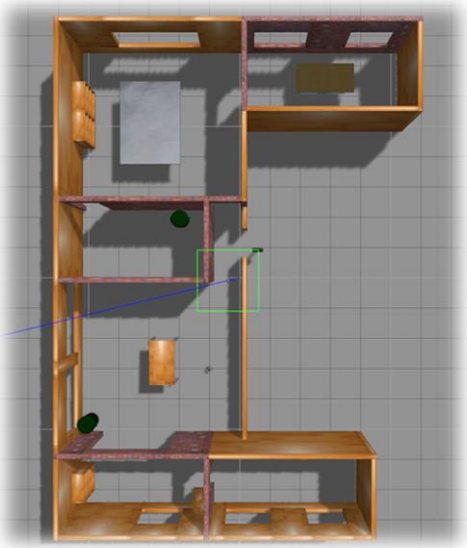
Loading TB3 Burger in TB3 World; type command in terminal:

- `$ export TURTLEBOT3_MODEL=burger`
- `$ roslaunch turtlebot3_gazebo turtlebot3_world.launch`
- Example shown in picture below:



Loading TB3 Burger in House World; type command in terminal:

- `$ export TURTLEBOT3_MODEL=burger`
- `$ roslaunch turtlebot3_gazebo turtlebot3_house.launch`
- Example shown in picture below:



***Note that if you have any issues loading the Gazebo worlds and/or the robot models, please reinstall the Dependent ROS Packages for OM and TB3, and the Simulation Packages for TB3.**

Testing of SLAM Nodes

***Note that if you have any issues with the SLAM application, please reinstall the SLAM Packages for TB3.**

First, launch Gazebo in TurtleBot3 World

- `$ export TURTLEBOT3_MODEL=burger`
- `$ roslaunch turtlebot3_gazebo turtlebot3_world.launch`

Second, launch SLAM node in a new terminal

- `$ export TURTLEBOT3_MODEL=burger`
- `$ roslaunch turtlebot3_slam turtlebot3_slam.launch slam_methods:=gmapping`
- Initial state will be as below image (i.e. **the required state; if you did not observe this, reinstall the SLAM packages for TB3**)
  - ❖ Note that if after the reinstallation and your SLAM module still don't work, close all terminals, manually install it in a new terminal:
    - ➢ `$ sudo apt install ros-kinetic-slam-gmapping`
    - ➢ Try launching Gazebo and the SLAM node again