Install Mint17 + ROS Indigo +Gazebo3+ Qt5.3.0

by Danping Zou: dpzou@sjtu.edu.cn
written by Huizhong Zhou: zhz218@hotmail.com

Please do install an individual linux system instead of using VMware.

1. ROS Indigo

see http://wiki.ros.org/hydro/Installation/Ubuntu

Notice: some changes has to be made according to the version of Linux and ROS, see below

1) Setup your souces.list

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

Note: 'trusty' is the ubuntu version of Mint17

2) Set up your keys

```
 wget https://raw.githubusercontent.com/ros/rosdistro/master/ros.key -O - | sudo apt-key add -
```

3) Installation

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

```
 sudo apt-get update
```

Install the lastest version 'Indigo'. Don't install the full version, but the desktop version.

```
 sudo apt-get install ros-indigo-desktop
```

4)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
```

5)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/indigo/setup.bash" >> ~/.bashrc
source ~/.bashrc
```

Note: you can edit the bashrc file using “gedit ~/.bashrc”

6)Getting rosinstall & others

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
```

```
sudo apt-get install ros-indigo-image-view
```
you can check you img topic using:(`rosrun image_view image_view image:=/image_raw`)

getting build tools
```
sudo apt-get build-essential
```

2. Gazebo 3
see http://gazebosim.org/wiki/3.0/install#Gazebo_in_different_deb_packages

1)
```
   sudo sh -c 'echo "deb http://packages.osrfoundation.org/gazebo/ubuntu trusty main" > /etc/apt/sources.list.d/gazebo-latest.list'
```
 Note: you have to find out your ubuntu version, here for mint 17 is 'trusty'

 2) Retrieve and install the keys for the Gazebo repositories.

```
   wget http://packages.osrfoundation.org/gazebo.key -O - | sudo apt-key add -
```

3) Update apt-get and install Gazebo.

```
   sudo apt-get update
   sudo apt-get install gazebo3
```
# For developers that works on top of gazebo, one extra package
```
   sudo apt-get install libgazebo-dev
```
(Please do this step)

4) Check your installation

```
   gzserver
```

5) The first time gzserver is executed requires the download of some models and it could take some time, please be patient. Wait until you see a message like Publicized address: ... and then execute a gazebo client:

```
    gzclient
```

3. Qt 5.3

see: http://qt-project.org/downloads
1)download Qt 5.3.0(Qt Library + Qt Creator) for Linux and click to install
!!! Don't use apt-get, since it installs the old version. Also don't use 5.3.1, there can be errors.

2)Environment Setup
open ~/.bashrc, insert this line
```
export CMAKE_PREFIX_PATH=<your qt path>/Qt5.3.0/5.3/gcc_64/lib/cmake:
$CMAKE_PREFIX_PATH
```
Note: if 32bit version is installed, then is gcc instead of gcc_64

4. Getting started

1)ROS Environment
for detail,see http://wiki.ros.org/ROS/Tutorials

a) create catkin workspace
```
        mkdir -p ~/catkin_ws/src
        cd ~/catkin_ws/src
        catkin_init_workspace
```
Note: initialize under src file instead of the root

b) create a new catkin packages (optional)
```
        cd ~/catkin_ws/src
        catkin_create_pkg <package name> <dependence 1>  < dependence 2> ...
```
e.g. catkin_create_pkg hello roscpp rospy std_msgs geometry_msgs

c) enable rosrun <package>
open ~/.bashrc, insert this line
```
        export ROS_PACKAGE_PATH = $ROS_PACKAGE_PATH:<your package path1>:<your
package path2>
```

Note: for the following example, export ROS_PACKAGE_PATH = $ROS_PACKAGE_PATH: <file path> /catkin_ws/src/sjtu_drone

5. for the sjtu_drone example

1)Download and Compiling

```
cd <catkin_ws>/src
git clone https://<your username>@bitbucket.org/dannis/sjtu_drone.git
cd <catkin_ws>
catkin_make
```

Here <catkin_ws> is the path of the catkin work space. Please refer to the tutorial about how to create a catkin work space in ROS.

2)Run
The simplest way is calling

```
cd <where you check out the code>
export ROS_PACKAGE_PATH=`pwd`:$ROS_PACKAGE_PATH
```
(If you have already added the package path into the bashrc file, then there is no need to do this step here.)
```
roslaunch sjtu_drone start.launch
```

or running the different parts of the package step by step

```
cd <where you check out the code>
export ROS_PACKAGE_PATH=`pwd`:$ROS_PACKAGE_PATH
roscore #to start the ROS server
rosrun sjtu_drone start_gzserver <world file> #run the gazebo server and loading the world file
rosrun sjtu_drone start_gui #run the gazebo client
rosrun sjtu_drone spawn_model # generate a quadrotor in the scene
rosrun sjtu_drone drone_keyboard # run the keyboard controller to control the quadrotor
```

6 Editing your package with Qt Creator

a) link your qt
```
sudo ln -s <QT_PATH>/tools/bin/qtcreator /usr/bin
```

b) open qt creator with the environment path
```
sh -i -c qtcreator
```

c) open project
----open the CmakeList.txt under your package file
----choose the Build Location to be under your package file
e.g. ../catkin_ws/src/hello/build
----edit your Run Cmake to be like this:
```
-DCATKIN_DEVEL_PREFIX=<your workspace path>/catkin_ws/devel
-DCMAKE_BUILD_TYPE=Debug
```

d)create a src file under the package file and new a cpp, edit
e.g. ../catkin_ws/src/hello/src/main.cpp

e) edit your package Cmakelists.txt

```
## Find catkin macros and libraries
## if COMPONENTS list like find_package(catkin REQUIRED COMPONENTS xyz)
## is used, also find other catkin packages
find_package(catkin REQUIRED COMPONENTS
  geometry_msgs
  roscpp
  rospy
  std_msgs
  cv_bridge
)
```
!!!! add the ROS package that you need(e.g. cv_bridge is a ros package to tranform ros img to

opencv img)

```
## System dependencies are found with CMake's conventions
# find_package(Boost REQUIRED COMPONENTS system)
find_package(OpenCV REQUIRED)
!!!! if you need to use opencv, add this

## Declare a cpp executable
add_executable(hello src/main.cpp)
!!!! add the cpp file here, there is no need to add head file here

## Specify libraries to link a library or executable target against
target_link_libraries(hello
  ${catkin_LIBRARIES}
  ${OpenCV_LIBRARIES}
)
!!!! if you need to use Opencv, also specify lib here
```

Note: if you want to include opencv, you have to edit your package Cmakelist.txt