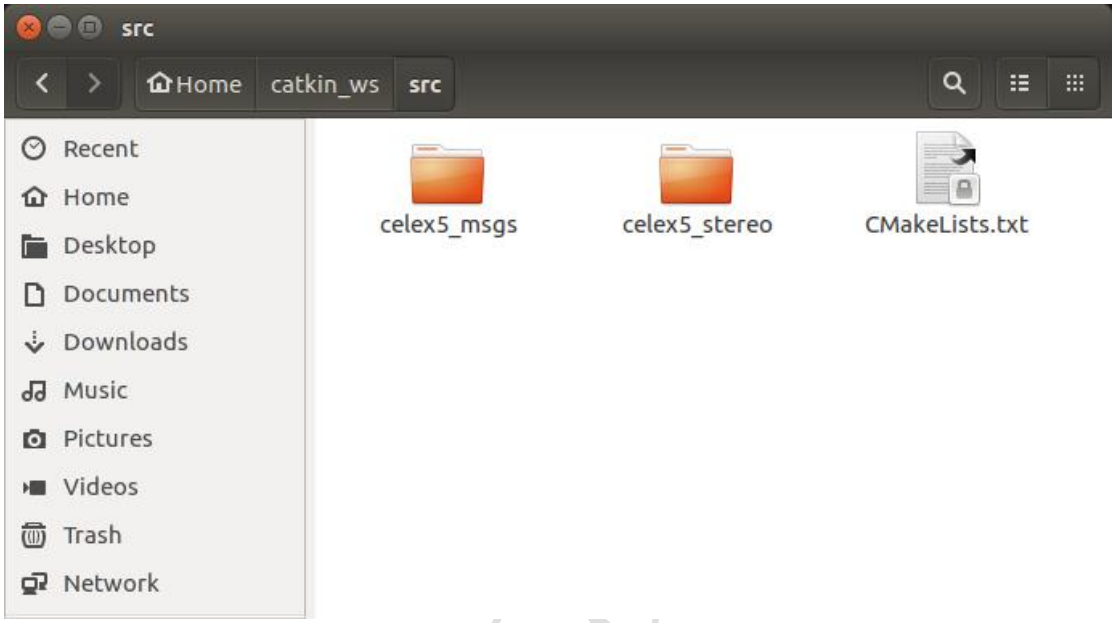
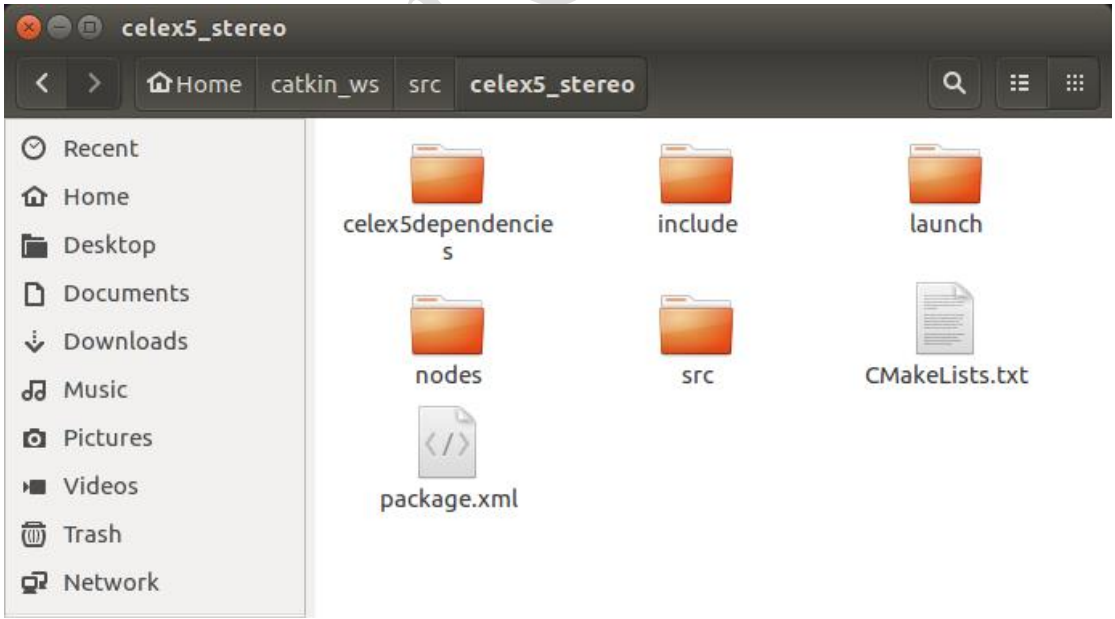
1 Introduction

The sample code file in the ROS environment under the release directory "**Sample for ROS**" mainly includes two package packages (celex5_msgs and celex5_stereo), wherein the ***celex5_msgs*** is a custom ROS message package, and the ***celex5_stereo*** is a CeleX5 Stereo function package. This example is compiled and run under Ubuntu 16.04 based on the Kinetic version of the ROS environment.



The CeleX5 Stereo function package includes the following five folders and files.

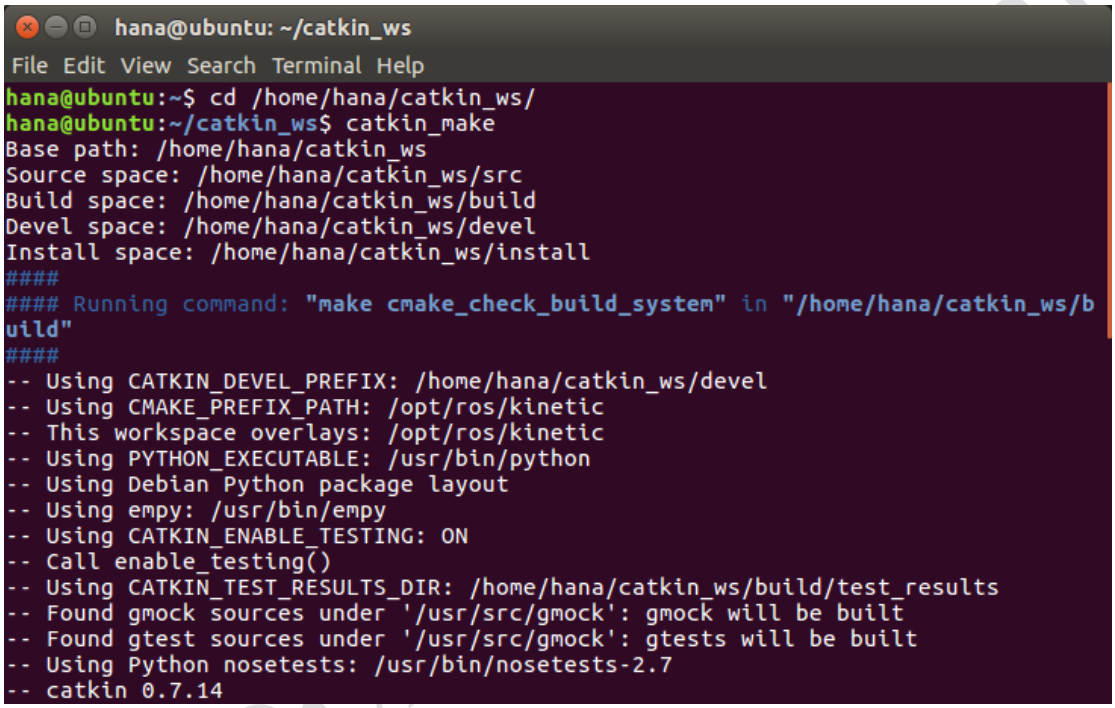


- **celex5dependencies：** This folder includes the API header files and library files.（for Ubuntu 16.04）
- **include：** This folder holds the header file.
- **launch：** The roslaunch startup file is stored in this file.
- **nodes：** This folder holds the startup node file for rosrun.

- ✧ **src**：The source file is stored in this folder.
- ✧ **CMakeLists.txt**：Used for compilation of executable files.
- ✧ **Package.xml**：Describe the properties of the package.

2 Compile ROS Sample Package

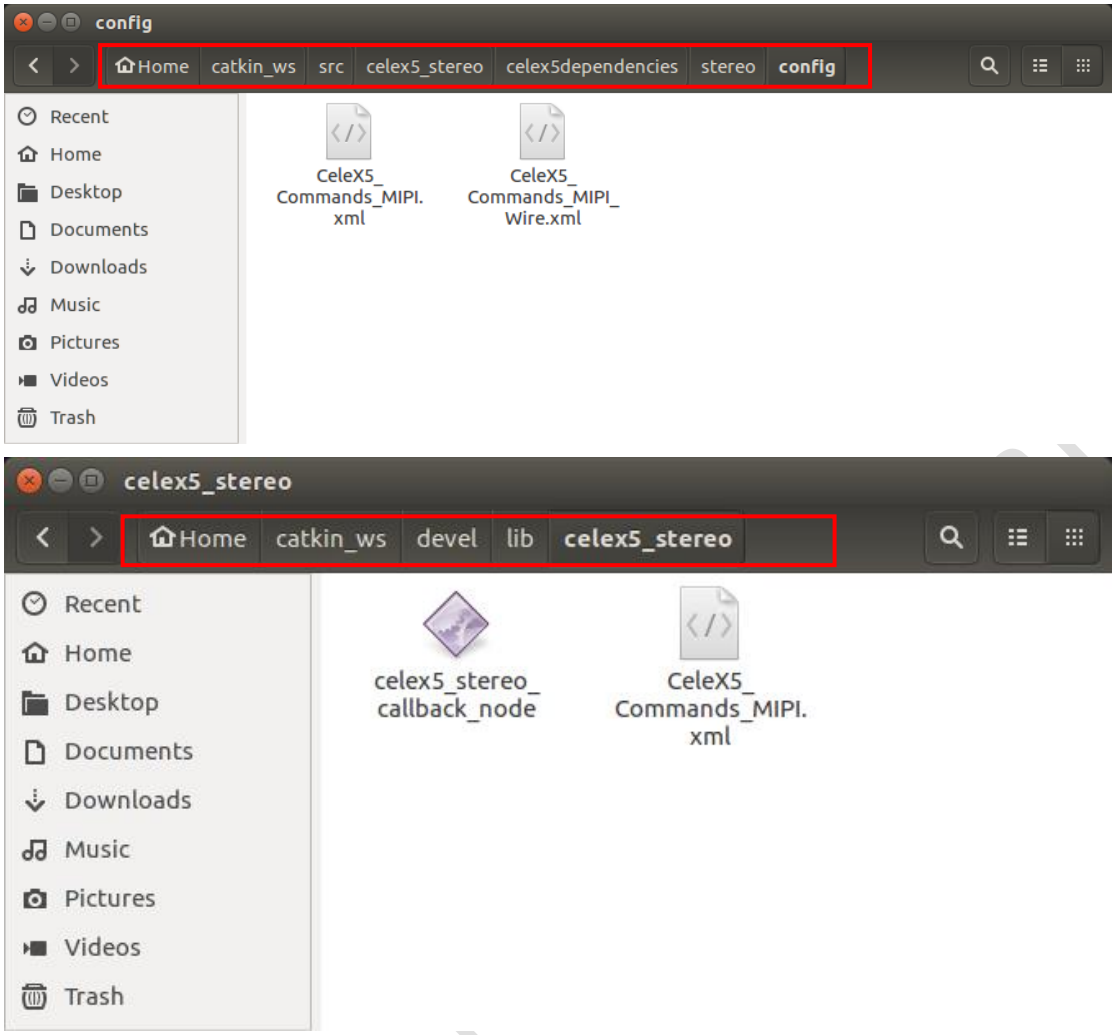
Place the ***celex5_msgs*** and ***celex5_stereo*** folders in the ***src*** directory of the created ROS workspace to compile. As shown in the following figure, the current ROS workspace is named ***catkin_ws***. Enter the workspace and use the ***catkin_make*** command to compile all catkin projects in the ***src*** directory. (**Note**: Compilation depends on OpenCV, users need to configure the OpenCV environment under Ubuntu, this example uses OpenCV version 3.3.0)



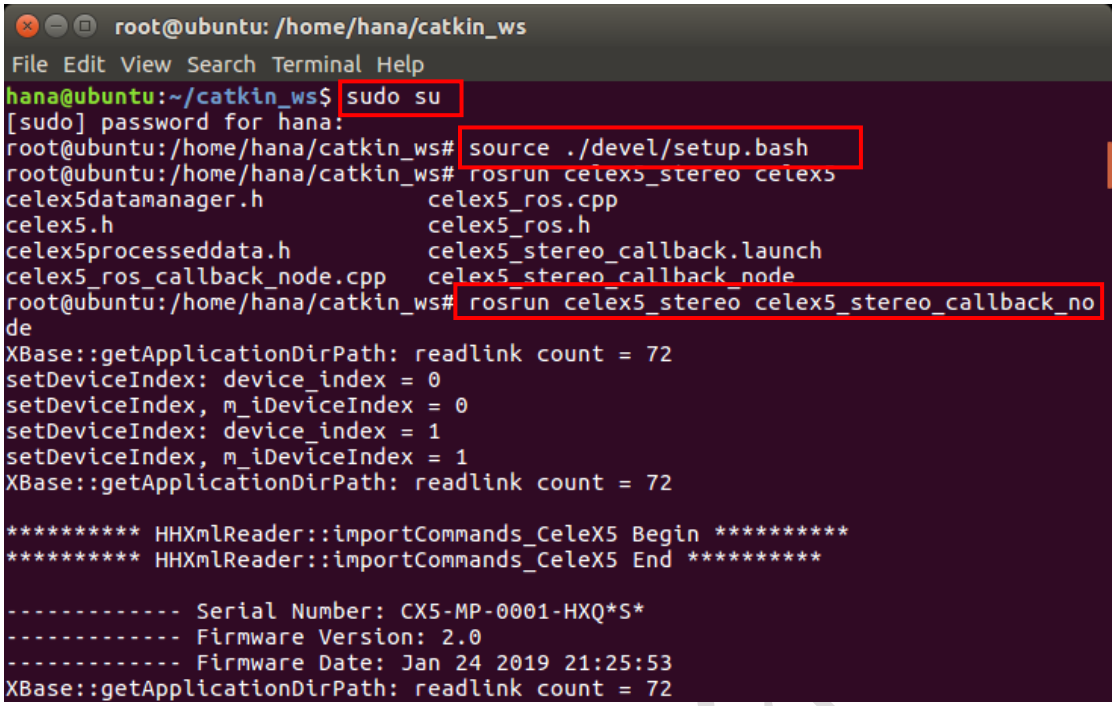
3 Run ROS Sample Package

After compilation, the executable files ***celex5_stereo_callback_node*** will be generated in the */devel /lib/celex5_stereo/* directory of the workspace.

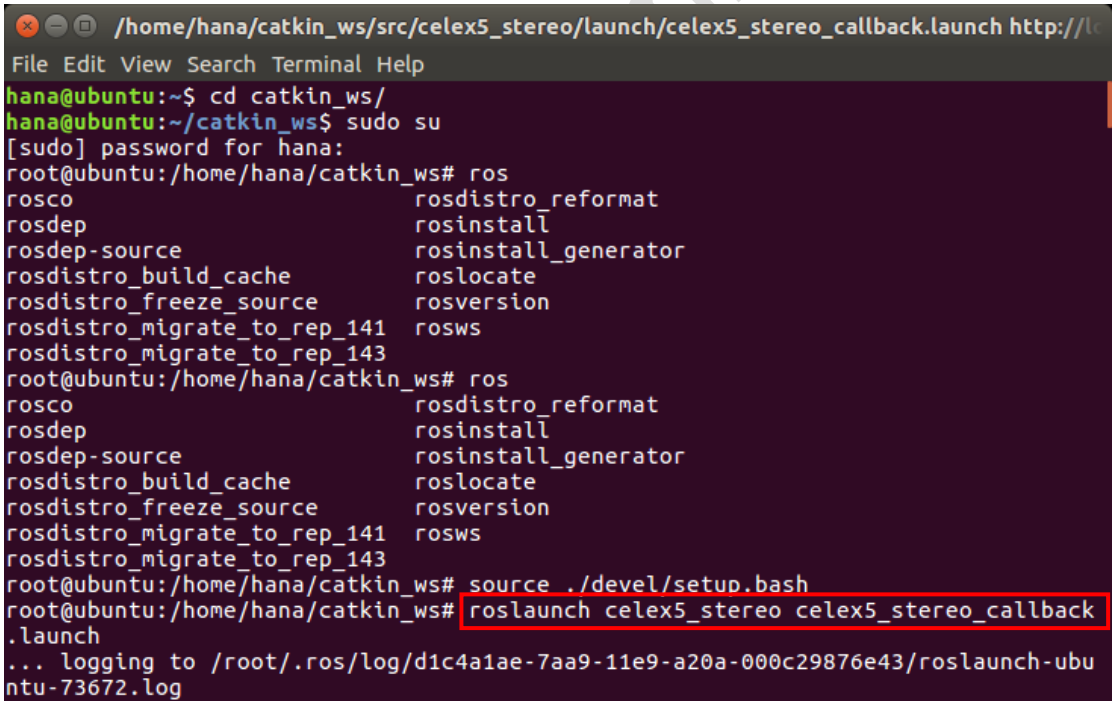
The users need to copy the necessary .xml configuration file (/home/YOUR_WORKSPACE/src/celex5_stereo/celexdependencies/stereo/config/*.xml) to the executable directory (/home/ YOUR_WORKSPACE/devel/lib/celex5_stereo/).



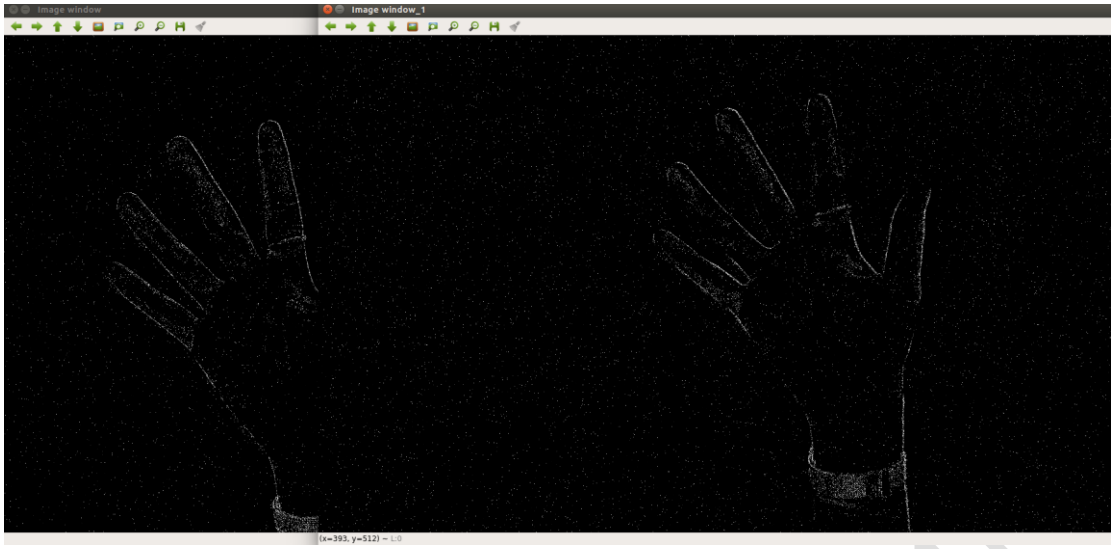
Before running the ROS package, you must first run ***roscore***. Then we can use ***rosrun*** or ***roslaunch*** to run the node. Since the CeleX5 needs to read and write to USB devices, so you need to use root permissions. In order to ensure the successful running of the node program, we first enter the root permission directly.



If there is a package *** not found or rosrun and other commands is invalid, you can use ***source ./devel/setup.bash*** to refresh the environment.



You can see two image windows after rosrun or roslaunch the node.



You can also view image information published by /imgshow and /imgshow1 via rviz.