

AMD64 架构系统环境搭建

睿尔曼智能科技（北京）有限公司

文件修订记录：

版本号	时间	备注
V1.0	2021-11-14	拟制

1. 主要配置环境预览

- 系统: Ubuntu 18.04.6
- ROS: melodic
- OpenCV 库: OpenCV 3.2.0
- Realsense D435: librealsense sdk (2.50.0) 、realsense-ros 功能包 (2.3.2)
- Marker 标记识别: Aruco 功能包
- 手眼标定: easy_handeye 功能包
- Moveit!
- RM 机械臂 ROS 功能包 (这里以 RM75 机械臂 ROS 包为例)

2. 更新清华源

2.1 备份源配置文件

```
sudo cp /etc/apt/sources.list /etc/apt/sources.list.bak
```

2.2 更改配置文件内容，替换为清华源

```
sudo gedit /etc/apt/sources.list
```

内容替换如下:

```
# 默认注释了源码镜像以提高 apt update 速度，如有需要可自行取消注释
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-updates main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-updates main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-backports main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-backports main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-security main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-security main restricted universe multiverse
# 预发布软件源，不建议启用
# deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-proposed main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-proposed main restricted universe multiverse
```

2.3 更新列表

```
sudo apt-get update
```

3. ROS 环境安装

3.1 配置 ROS 软件源

1) 添加 ROS 源

建议使用国内或者新加坡的镜像源，这样能够大大提高安装下载速度。

清华大学镜像源：

```
sudo sh -c '. /etc/lsb-release && echo "deb http://mirrors.tuna.tsinghua.edu.cn/ros/ubuntu/$DISTRIB_CODENAME main" > /etc/apt/sources.list.d/ros-latest.list'
```

2) 添加 Keys

使用如下命令获取 key：

```
sudo apt install curl  
curl -s https://raw.githubusercontent.com/ros/rosdistro/master/ros.asc | sudo apt-key add -
```

若网络不能正常下载，请使用如下镜像命令加速：

```
curl -s https://raw.fastgit.org/ros/rosdistro/master/ros.asc | sudo apt-key add -
```

3) 更新列表

```
sudo apt-get update
```

3.2 安装

桌面完整版安装：包含 ROS、rqt、rviz、机器人通用库、2D/3D 模拟器、导航以及 2D/3D 感知，依次执行如下 5 条命令（注意：第二条命令一行显示不下，实际为一行，连接处有空格）：

```
sudo apt install libvtk6-jni libvtk6-java libvtk6-dev libvtk6-qt-dev libpcl-dev  
  
sudo apt install ros-melodic-pcl-conversions ros-melodic-pcl-ros ros-melodic-perception-pcl  
ros-melodic-perception ros-melodic-rqt  
  
sudo apt install ros-melodic-desktop-full  
  
sudo apt install python-rosdep  
  
sudo apt install python-rosinstall python-rosinstall-generator python-wstool build-essential ros-melodic-catkin
```

pip 包管理工具安装：

```
sudo apt install python3-dev python3-pip python-pip python-dev
```

catkin_tools 安装：

```
sudo pip3 install -U catkin_tools
```

3.3 初始化 rosdep

在开始使用 ROS 之前你还需要初始化 rosdep。rosdep 可以方便在你需要编译某些源码的时候为其安装一些系统依赖，同时也是某些 ROS 核心功能组件所必需用到的工具。

通过 pip 包管理工具安装国内镜像加速版本 rosdepc，依次执行如下命令（如果遇到执行命令后有类似 ERROR 错误的提示信息，则再次执行出错的命令，直到没错误后再执行下一条命令）：

```
sudo pip install rosdepc
sudo rosdepc init
rosdepc update
```

3.4 环境配置

配置 ROS 环境变量，依次执行如下命令：

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

至此，ROS 环境配置完成。

3.5 测试

完成安装后跑一个测试程序-小海龟。若没有安装，执行如下命令：

```
sudo apt-get install ros-melodic-turtlesim
```

然后，在三个不同的终端分别执行以下三个命令：

```
roscore
roslaunch turtlesim turtlesim_node
roslaunch turtlesim turtle_teleop_key
```

就能跑起小海龟示例，利用键盘的上下左右键控制小海龟行走。如图：



4. 安装 MoveIt!

4.1 MoveIt! 简介

在实现机械臂的自主抓取中机械臂的运动规划是其中最重要的一部分，其中包含运动学正逆解算、碰撞检测、环境感知和动作规划等。常见机械臂的运动规划大都采用的是 ROS 系统提供的 MoveIt! 规划。

MoveIt! 是 ROS 系统中集合了与移动操作相关的组件包的运动规划库。它包含了运动规划所需要的大部分功能，同时其提供友好的配置和调试界面便于完成机器人在 ROS 系统上的初始化及调试。

官方网站：<http://moveit.ros.org/>，上边有 MoveIt! 的教程和 API 说明。

4.2 安装 MoveIt!

MoveIt! 需要安装才能使用，如果未安装，请执行如下命令进行安装。

```
sudo apt install ros-melodic-moveit
sudo apt install ros-melodic-moveit-*
```

5. 安装 OpenCV

基于 OpenCV 库，可以快速开发机器视觉方面的应用，而且 ROS 已经集成了 OpenCV 库和相关的接口功能包，执行以下命令即可安装：

```
sudo apt-get install ros-melodic-vision-opencv libopencv-dev python-opencv
```

6. Realsense D435 相机 SDK 及 ROS 包安装

6.1 librealsense sdk 部署

参考：https://github.com/IntelRealSense/librealsense/blob/master/doc/distribution_linux.md

联网执行如下命令添加 apt-key：

```
sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-key
F6E65AC044F831AC80A06380C8B3A55A6F3EFCDE || sudo apt-key adv --keyserver
hkp://keyserver.ubuntu.com:80 --recv-key F6E65AC044F831AC80A06380C8B3A55A6F3EFCDE
```

执行如下命令添加 librealsense apt 源仓库：

```
sudo add-apt-repository "deb https://librealsense.intel.com/Debian/apt-repo $(lsb_release -cs) main" -u
```

安装库：

```
sudo apt-get install librealsense2-dkms
sudo apt-get install librealsense2-utils
```

安装开发包和调试包：

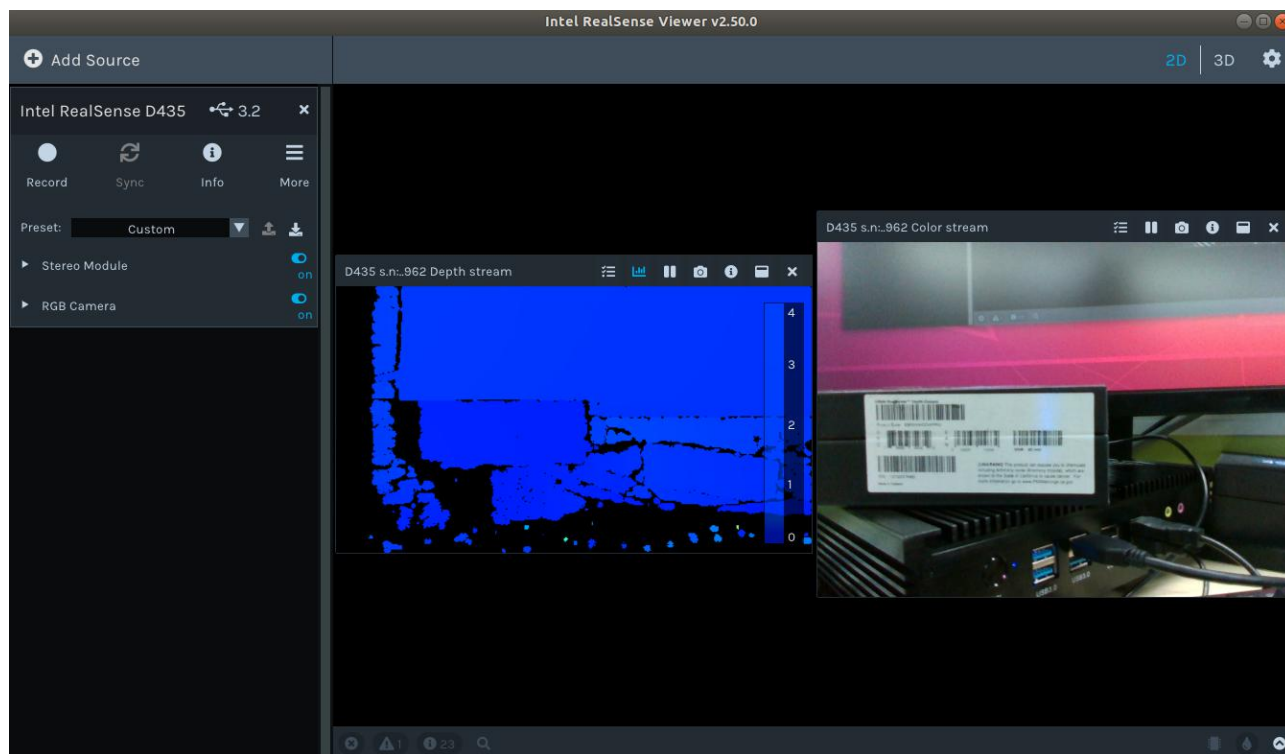
```
sudo apt-get install librealsense2-dev
```

```
sudo apt-get install librealsense2-dbg
```

安装完成之后，通过 Type-C 数据线将相机与电脑 USB3.0 接口相连，在终端软件中执行如下命令打开可视化测试界面(若未检测到相机尝试重新连接 USB):

```
realsense-viewer
```

相机连接成功后，将左侧【Stereo Module】和【RGB Camera】选择“on”状态，右侧会显示深度图像和 RGB 图像画面。鼠标移动到深度图像位置左下方画面会显示实时测量的深度距离，如图：



6.2 realsense-ros 功能包编译

进入工作空间的 src 目录下，如果没有工作空间，执行以下命令创建工作空间：

```
mkdir -p ~/catkin_ws/src && cd ~/catkin_ws/src
```

执行以下命令下载源码并安装依赖库（如果下载连接超时，可以将已下载提供的源码包解压并拷贝到~/catkin_ws/src 下，如果是拷贝的就不需要执行 git checkout 2.3.2 命令）：

```
git clone https://github.com/IntelRealSense/realsense-ros.git
cd realsense-ros/realsense2_camera
git checkout 2.3.2
sudo apt-get install ros-melodic-ddynamic-reconfigure
```

执行以下命令编译完成后即可完成功能包的安装：

```
cd ~/catkin_ws
catkin build realsense2_camera
```



```

realman@realman-laptop:~$ cd ~/catkin_ws/
realman@realman-laptop:~/catkin_ws$ catkin build realsense2_camera

Profile: default
Extending: [cached] /opt/ros/melodic
Workspace: /home/realman/catkin_ws

-----
Build Space: [exists] /home/realman/catkin_ws/build
Devel Space: [exists] /home/realman/catkin_ws/devel
Install Space: [unused] /home/realman/catkin_ws/install
Log Space: [exists] /home/realman/catkin_ws/logs
Source Space: [exists] /home/realman/catkin_ws/src
DESTDIR: [unused] None

-----
Devel Space Layout: linked
Install Space Layout: None

-----
Additional CMake Args: None
Additional Make Args: None
Additional catkin Make Args: None
Internal Make Job Server: True
Cache Job Environments: False

-----
Whitelisted Packages: None
Blacklisted Packages: None

-----
Workspace configuration appears valid.

-----
[build] Found '8' packages in 0.0 seconds.
[build] Package table is up to date.
Starting >>> realsense2_camera
Finished <<< realsense2_camera [ 1.4 seconds ]
[build] Summary: All 1 packages succeeded!
[build] Ignored: 7 packages were skipped or are blacklisted.
[build] Warnings: None.
[build] Abandoned: None.
[build] Failed: None.
[build] Runtime: 1.4 seconds total.
realman@realman-laptop:~/catkin_ws$

```

6.3 启动节点测试

在终端输入如下命令启动相关节点:

```

cd ~/catkin_ws
source devel/setup.bash
roslaunch realsense2_camera rs_camera.launch

```

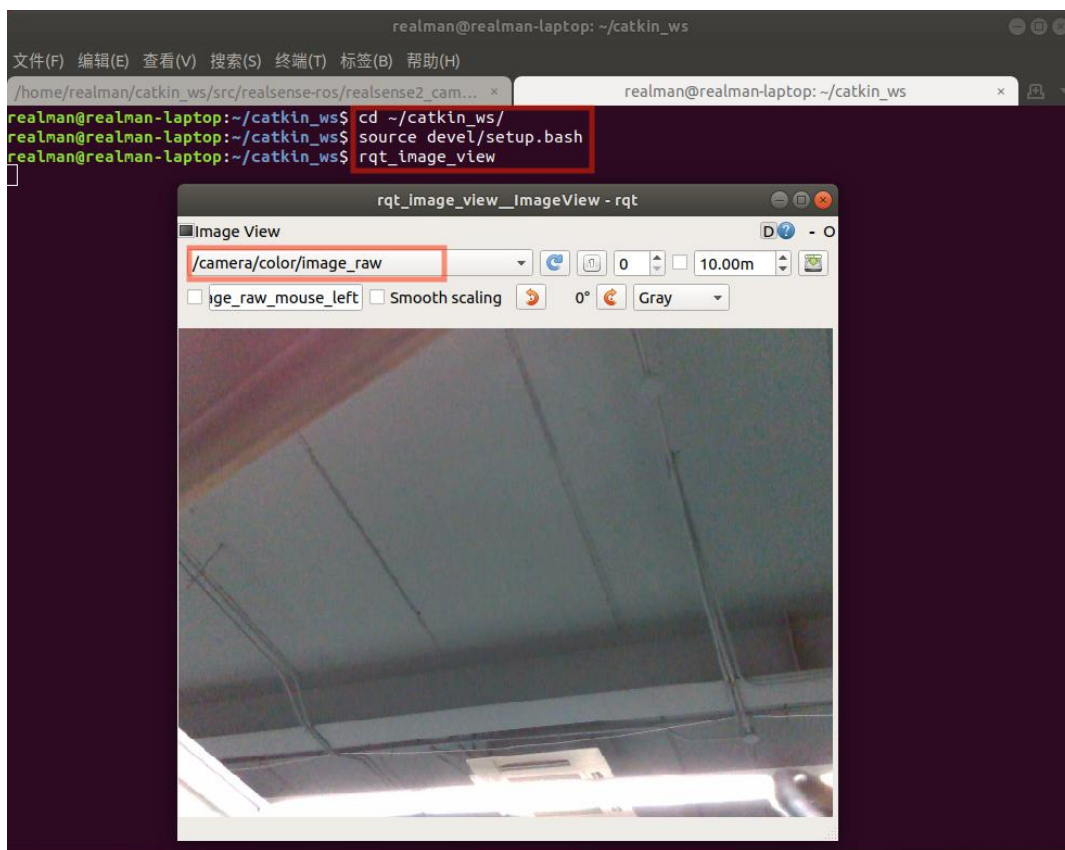
```

/home/realman/catkin_ws/src/realsense-ros/realsense2_camera/launch/rs_camera.launch http://localhost:11311
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
[INFO] [1639471966.582111583]: Built with LibRealSense v2.50.0
[INFO] [1639471966.582154314]: Running with LibRealSense v2.50.0
[INFO] [1639471966.608428472]:
[INFO] [1639471966.614174509]: Device with serial number 137322078962 was found.
[INFO] [1639471966.614247944]: Device with physical ID /sys/devices/pci0000:00/0000:00:14.0/usb3/3-1/3-1:1.0/video4linux/vi
bo0 was found.
[INFO] [1639471966.614293765]: Device with name Intel RealSense D435 was found.
[INFO] [1639471966.614935268]: Device with port number 3-1 was found.
[INFO] [1639471966.614991985]: Device USB type: 3.2
[INFO] [1639471966.618542300]: getParameters...
[INFO] [1639471966.672613411]: SetupDevice...
[INFO] [1639471966.672663406]: JSON file is not provided
[INFO] [1639471966.672699541]: ROS Node Namespace: camera
[INFO] [1639471966.672738873]: Device Name: Intel RealSense D435
[INFO] [1639471966.672779843]: Device Serial No: 137322078962
[INFO] [1639471966.672797659]: Device physical port: /sys/devices/pci0000:00/0000:00:14.0/usb3/3-1/3-1:1.0/video4linux/video
[INFO] [1639471966.672826760]: Device FW version: 05.13.00.50
[INFO] [1639471966.672855684]: Device Product ID: 0x0B07
[INFO] [1639471966.672880771]: Enable PointCloud: Off
[INFO] [1639471966.672902753]: Align Depth: Off
[INFO] [1639471966.672941290]: Sync Mode: Off
[INFO] [1639471966.673001089]: Device Sensors:
[INFO] [1639471966.675687660]: Stereo Module was found.
[INFO] [1639471966.681077192]: RGB Camera was found.
[INFO] [1639471966.681142596]: (Confidence, 0) sensor isn't supported by current device! -- Skipping...
[INFO] [1639471966.681200985]: num filters: 0
[INFO] [1639471966.681241109]: Setting Dynamic reconfig parameters.
[INFO] [1639471966.732648946]: Done Setting Dynamic reconfig parameters.
[INFO] [1639471966.735070792]: depth stream is enabled - width: 848, height: 480, fps: 30, Format: Z16
[INFO] [1639471966.735891203]: color stream is enabled - width: 640, height: 480, fps: 30, Format: RGB8
[INFO] [1639471966.735960930]: setupPublishers...
[INFO] [1639471966.739830103]: Expected frequency for depth = 30.00000
[INFO] [1639471966.786686451]: Expected frequency for color = 30.00000
[INFO] [1639471966.808855024]: setupStreams...
[INFO] [1639471966.936163645]: SELECTED BASE:Depth, 0
[INFO] [1639471966.954314706]: RealSense Node Is Up!
[WARN] [1639471967.007367169]:

```


打开一个新的终端，输入如下命令查看 topic 发布的图像：

```
rqt_image_view
```



7. 安装 aruco_ros 功能包

7.1 扩展 swap 交换空间

由于系统默认设置的 swap 交换空间不够，而编译 aruco_ros 功能包的时候需要较大的交换空间，所以需要扩展 swap 交换空间，避免编译时出现卡死的问题。

1) 在 /opt/image 中添加 2G 的 swap 交换文件，执行以下命令：

```
sudo mkdir /opt/image
cd /opt/image
sudo touch swap
```

2) 添加交换文件并设置其大小为 2G，使用如下命令：

```
sudo dd if=/dev/zero of=/opt/image/swap bs=1024 count=2048000
```

之后会返回类似结果：

```
2048000+0 records in
2048000+0 records out
2097152000 bytes (2.1 GB, 2.0 GiB) copied, 242.095 s, 8.7 MB/s
```

注意：此过程等待时间有点长，不要以为是系统死机了。

3) 创建（设置）交换空间，使用命令 `mkswap`：

```
sudo mkswap /opt/image/swap
```

返回类似结果：

```
Setting up swapspace version 1, size = 2 GiB (2097147904 bytes)
```

4) 检查现有的交换空间大小，使用命令 `free`：

```
free -m
```

返回类似结果：

	total	used	free	shared	buff/cache	available
Mem:	925	185	28	14	711	660
Swap:	0	0	0			

5) 启动新增加的 2G 的交换空间，使用命令 `swapon`：

```
sudo swapon /opt/image/swap
```

6) 确认新增加的 2G 交换空间已经生效，使用命令 `free`：

```
free -m
```

返回类似结果：

	total	used	free	shared	buff/cache	available
Mem:	925	328	56	32	541	502
Swap:	1999	0	1999			

7) 修改 `/etc/fstab` 文件，使得新加的 2G 交换空间在系统重新启动后自动生效：

```
sudo gedit /etc/fstab
```

在文件最后添加一行，内容如下：

```
/opt/image/swap    /swap    swap    defaults 0 0
```

8) 重启系统

7.2 编译 aruco_ros 功能包

进入工作空间的 `src` 目录下，执行以下命令下载源码（如果下载连接超时，可以将已下载提供的源码包解压并拷贝到 `~/catkin_ws/src` 下）：

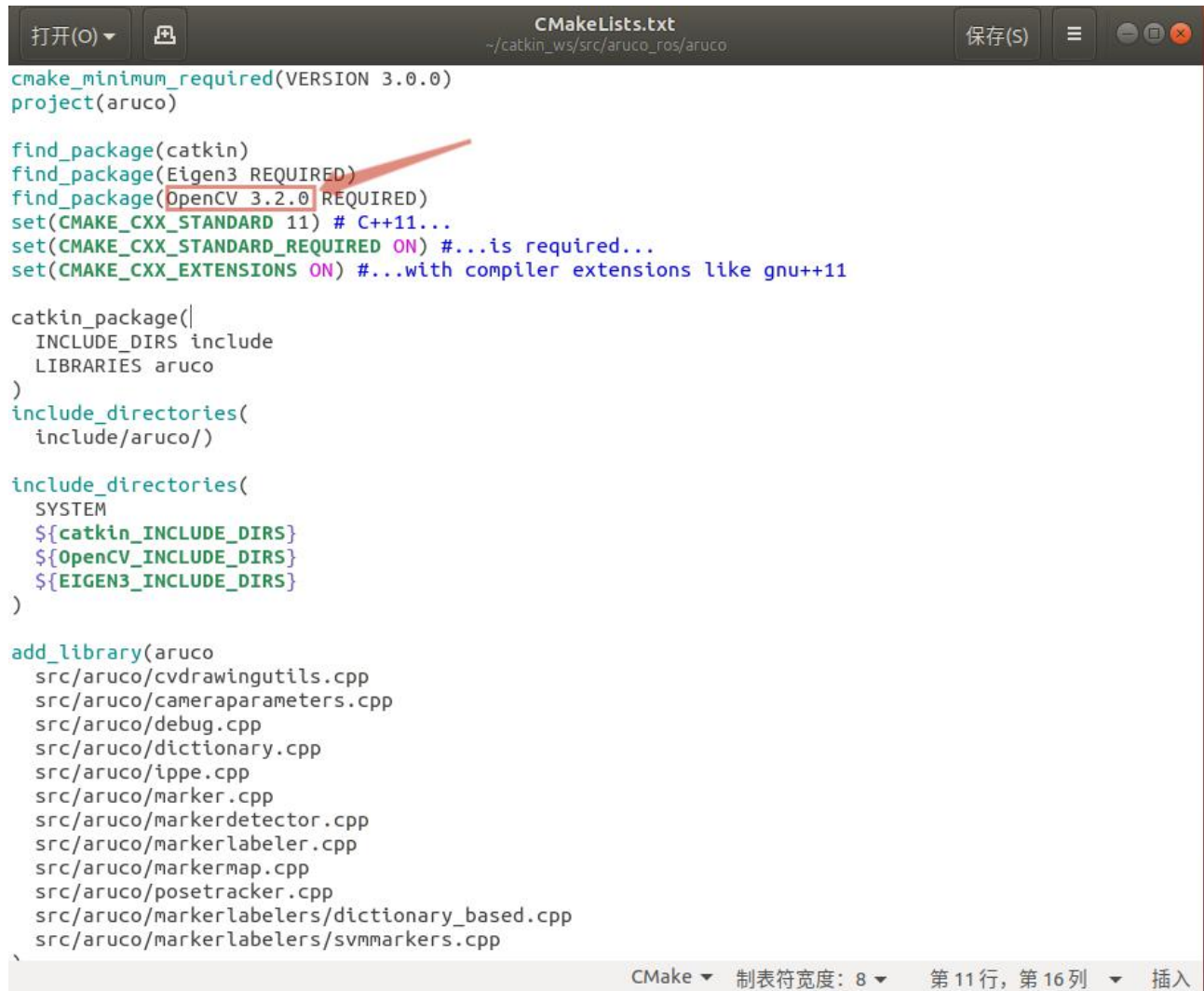
```
cd ~/catkin_ws/src
git clone https://github.com/pal-robotics/aruco_ros.git
```

由于 `aruco` 功能包编译需要依赖 `OpenCV`，默认依赖 `OpenCV4` 以上版本，而当前系统安装的为 3.2.0 版本，所以需要修改编译文件，执行以下命令打开 `aruco` 下的 `CMakeLists.txt` 进行编辑：

```
cd ~/catkin_ws/src/aruco_ros/aruco
```

gedit CMakeLists.txt

将依赖的 OpenCV 版本修改为 3.2.0，然后保存并关闭，如下图所示：



```

cmake_minimum_required(VERSION 3.0.0)
project(aruco)

find_package(catkin)
find_package(Eigen3 REQUIRED)
find_package(OpenCV 3.2.0 REQUIRED)
set(CMAKE_CXX_STANDARD 11) # C++11...
set(CMAKE_CXX_STANDARD_REQUIRED ON) #...is required...
set(CMAKE_CXX_EXTENSIONS ON) #...with compiler extensions like gnu++11

catkin_package(
  INCLUDE_DIRS include
  LIBRARIES aruco
)
include_directories(
  include/aruco/)

include_directories(
  SYSTEM
  ${catkin_INCLUDE_DIRS}
  ${OpenCV_INCLUDE_DIRS}
  ${EIGEN3_INCLUDE_DIRS}
)

add_library(aruco
  src/aruco/cvdrawingutils.cpp
  src/aruco/cameraparameters.cpp
  src/aruco/debug.cpp
  src/aruco/dictionary.cpp
  src/aruco/ippe.cpp
  src/aruco/marker.cpp
  src/aruco/markerdetector.cpp
  src/aruco/markerlabeler.cpp
  src/aruco/markermapper.cpp
  src/aruco/posetracker.cpp
  src/aruco/markerlabelers/dictionary_based.cpp
  src/aruco/markerlabelers/svmmarkers.cpp
)

```

执行以下命令编译完成后即可完成功能包的安装：

```

cd ~/catkin_ws
catkin build

```

8. 安装 easy_handeye 功能包

8.1 安装依赖库

执行以下命令安装所需依赖库：

```

sudo apt-get install libvisp-dev
sudo apt-get install ros-melodic-visp
sudo apt-get install ros-melodic-vision-visp

```

8.2 编译 easy_handeye 功能包

进入工作空间的 src 目录下，执行以下命令下载源码（如果下载连接超时，可以将已下

AMD64 架构系统环境搭建

载提供的源码包解压并拷贝到~/catkin_ws/src 下)：

```
cd ~/catkin_ws/src
git clone https://github.com/IFL-CAMP/easy_handeye.git
```

执行以下命令编译完成后即可完成功能包的安装：

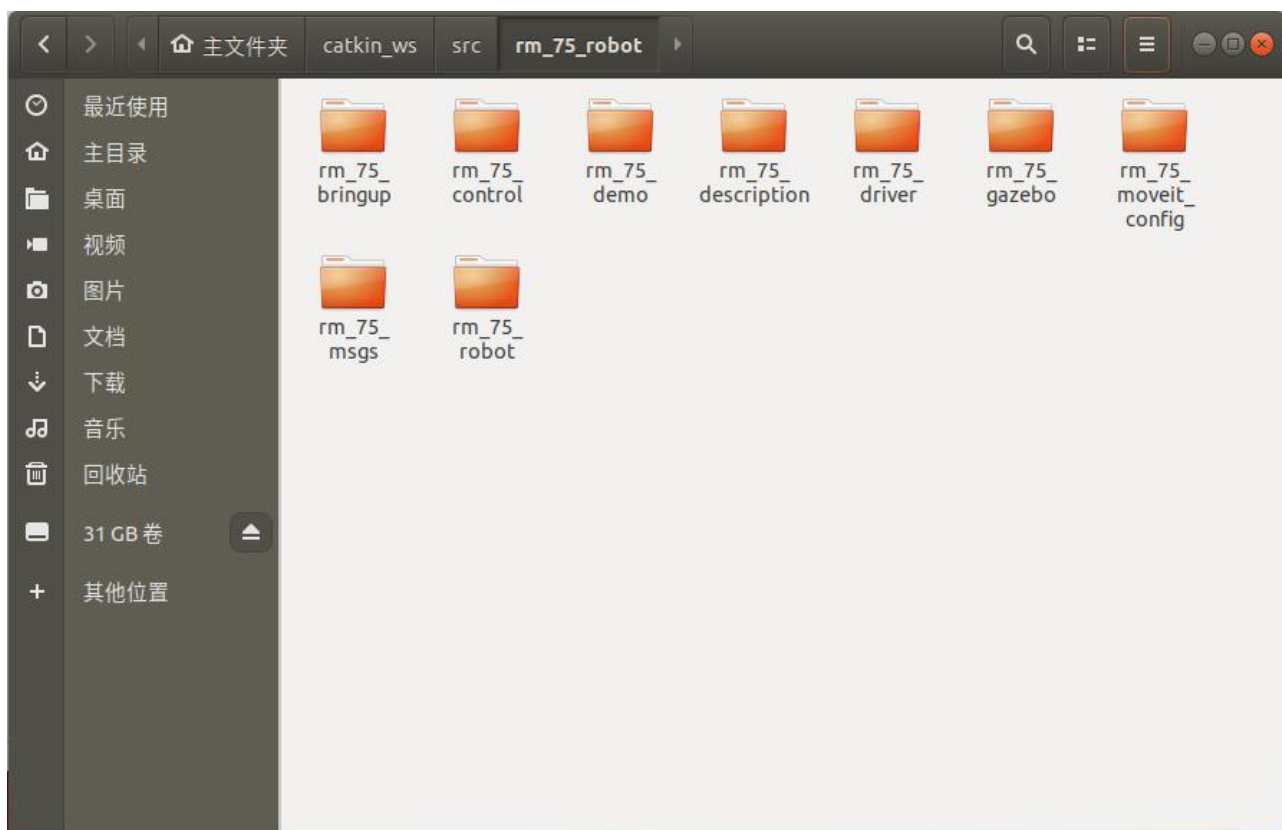
```
cd ~/catkin_ws
catkin build
```

安装启动标定程序时所需的依赖库，依次执行以下命令等待安装完成即可：

```
sudo pip install transforms3d
sudo python -m pip install opencv-contrib-python
```

9. 安装 RM 机械臂功能包

将提供的 rm_75_robot 源码包拷贝到 catkin_ws 工作空间的 src 目录下（这里以 RM75 机械臂为例，如果使用的是 RM65 机械臂则将配套的 6 轴机械臂的 ROS 包拷贝到目录下）：



使用 catkin 工具配置工作空间并进行编译，依次执行如下命令：

```
cd ~/catkin_ws
catkin build rm_75_msgs
catkin build
```