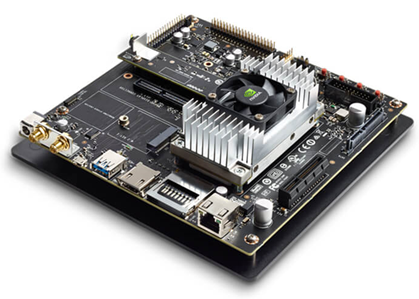
# Linux简介

## 1.1 Linux介绍

Linux是一个基于POSIX 和 UNIX的多用户、多任务、支持多线程和多 CPU 的操作系统。Linux主要应用于服务器系统、嵌入式开发、Android系统和智能硬件。

## 1.2常用的图像处理芯片

（1）Jetson TX2



（2）Jetson nano



（3）海思

# Linux安装

## 2.1安装VMware

## 2.2卸载掉Hyper-V

如果电脑中安装Hyper-V，需要先卸载掉Hyper-V，不然会死机。

参照http://www.xitongzhijia.net/xtjc/20220102/237014.html

## **2.3下载Ubuntu镜像**

## 2.4给root用户设置密码。

**$**sudo passwd root

输入密码，并确认密码。设置好密码后下次可用su root切换登录用户。

## 2.5更新软件源中的所有软件列表。

**$**apt-get update

## 2.6安装ssh服务端

**$**apt-get install openssh-server

## 2.7查看系统ip地址。

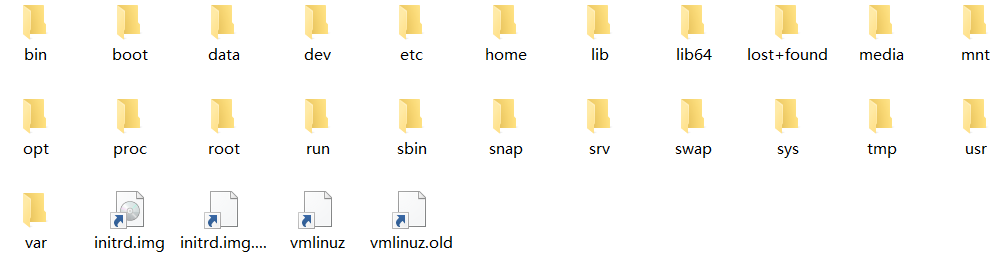
**$**ip a

## 2.8在xshell上使用root登录linux系统

如果无法登录参照

<https://blog.csdn.net/weixin_43949535/article/details/104409220>

# Linux目录结构



**bin：**全称binary，含义是二进制。该目录中存储的都是一些二进制文件，文件都是可以被运行的。

**dev**：该目录中主要存放的是外接设备，例如盘、其他的光盘等。在其中的外接设备是不能直接被使用的，需要挂载（类似window下的分配盘符）。

**etc**：该目录主要存储一些配置文件。

**home**：表示“家”，表示除了root用户以外其他用户的家目录，类似于windows下的User/用户目录。

**proc**：全称process，表示进程，该目录中存储的是Linux运行时候的进程。

**root**：该目录是root用户自己的家目录。

**sbin**：全称super binary，该目录也是存储一些可以被执行的二进制文件，但是必须得有super权限的用户才能执行。

**tmp**：表示“临时”的，当系统运行时候产生的临时文件会在这个目录存着。

**usr**：存放的是用户自己安装的软件。类似于windows下的program files。

**var**：存放的程序/系统的日志文件的目录。

**mnt**：当外接设备需要挂载的时候，就需要挂载到mnt目录下。



# 常用指令

**cd**（change directory）：切换目录

**ls**：用于显示指定工作目录下之内容(不加参数则列出当前目录下的内容。

**mkdir** ：创建目录

**touch** ：创建新文件

**rm** : 删除文件

**mv：移动文件**

**vim: 文件编辑器（详见**https://blog.csdn.net/pansaky/article/details/83755750**）**

参照<https://www.runoob.com/linux/linux-command-manual.html>

# Linux上docker的安装

## 5.1 docker的安装与使用

（1）docker的安装

**$ sudo apt install curl**

**$ curl -fsSL https://get.docker.com -o get-docker.sh**

**$ sudo sh get-docker.sh**

(参照<https://www.runoob.com/docker/ubuntu-docker-install.html>)

（2）使用docker安装mysql

**$docker pull mysql:latest**

**$docker run -itd --name mysql-test -p 3306:3306 -e MYSQL\_ROOT\_PASSWORD=123456 mysql**

## 5.2 docker常见命令

**docker run <镜像名>**

**docker run -d <镜像名> //detach 分离模式，在后台运行**

**docker run -it <镜像名> //interactive 交互模式**

**docker pull // 拉取镜像**

**docker images // 列出所有镜像**

**docker rmi // 删除镜像**

**docker ps // 列出所有正在运行的容器**

**docker ps -a //列出所有容器**

**docker stop <容器> //停止运行容器**

**docker rm <容器> //删除容器（只能删除已停止的容器）**

**docker start <容器> //把处于停止状态的容器再次开启**

# Cmake安装与Opencv编译

## ****6.1**安装cmake**

**$sudo apt-get install cmake**

## ****6.2**安装make**

**$sudo apt-get install make**

## ****6.3下载opencv3.4.5源码****

**$wget** <https://github.com/opencv/opencv/archive/3.4.5.zip>

## ****6.4解压到当前文件夹****

**$unzip opencv3.4.5.zip**

## ****6.5安装依赖环境****

**$sudo apt-get install** **build-essential libgtk2.0-dev libavcodec-dev libavformat-dev libjpeg-dev libswscale-dev libtiff5-dev**

**$sudo apt-get install libcanberra-gtk-module**

**$sudo apt-get install pkg-config**

## ****6.6进入build文件夹开始编译****

**$cmake -D CMAKE\_BUILD\_TYPE=Release –D OPENCV\_GENERATE\_PKGCONFIG=ON -D CMAKE\_INSTALL\_PREFIX=/usr/local ..**

## **6.7使用双核编译**

让make最多允许4个命令同时执行

$sudo make -j4

## **6.8**将OPENCV安装至系统中。

$sudo make install

## **6.9环境配置**

（1）将opencv.pc文件复制到pkgconfig文件夹里

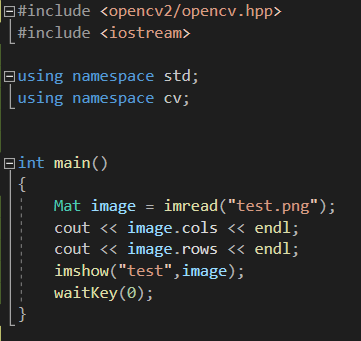
$sudo cp /usr/local/lib/pkgconfig/opencv.pc /usr/lib/pkgconfig/

(2)测试opencv版本

$pkg-config opencv --modversion

## **6.10测试程序**

（1）编写如下代码保存为cvtest.cpp



（2）用g++编译

$g++ cvtest.cpp -o cvtest.out $(pkg-config --cflags --libs opencv)

（3）执行cvtest.out

$./cvTest.out

正确输出图像的长宽，则opencv安装完成