磁盘分区：

swap：交换空间 /：根目录 /home：主目录

磁盘挂载：

1. 获取磁盘 UUID：

```
sudo blkid
```

查找想要后续进行挂载的磁盘，并且获取其 UUID，TYPE 信息

2. 创建挂载点：

创建目录作为分区或硬盘的挂载点。例如，如果要将分区挂载到 `/media/data`，请使用以下命令创建目录：

```
sudo mkdir /media/data
```

3. 编辑 `/etc/fstab` 文件：

`/etc/fstab` 文件包含有关应在启动时挂载的分区的的信息。使用以下命令打开文件进行编辑：

```
sudo nano /etc/fstab
```

在文件的末尾添加一行，格式如下：

```
UUID=<partition_UUID> /media/data <filesystem_type> defaults 0 0
```

将 `<partition_UUID>` 替换为在步骤 1 中标识的分区或硬盘的 UUID，将 `/media/data` 替换为在步骤 2 中创建的挂载点，并将 `<filesystem_type>` 替换为分区的文件系统类型（例如，`ext4`、`ntfs`）。

4. 保存并关闭 `/etc/fstab` 文件：

使用 ctrl+x 并输入 y 保存并退出

5. 重启:

换源并更新软件包：

软件与更新，换源

```
sudo apt update
sudo apt upgrade
```

WiFi问题：

```
sudo apt install flex bison
sudo apt install git
sudo apt-get install make
git clone
https://git.kernel.org/pub/scm/linux/kernel/git/iwlwifi/backport-
iwlwifi.git
cd backport-iwlwifi
sudo make defconfig-iwlwifi-public
sudo make
sudo make install
git clone https://mirrors.tuna.tsinghua.edu.cn/git/linux-
firmware.git
cd linux-firmware/
sudo cp iwlwifi-* /lib/firmware/
reboot
```

声卡问题：

```
lsmod | grep snd_hda_intel
```

- step 1:

```
sudo gedit /etc/modprobe.d/alsa-base.conf
```

添加：

```
options snd-hda-intel model=generic
options snd-hda-intel dmic_detect=0
```

- step 2:

```
sudo gedit /etc/modprobe.d/blacklist.conf
```

添加：

```
blacklist snd_soc_skl
reboot
```

安装显卡驱动相关指令：

```
查看：
nvidia-smi
ubuntu-drivers devices
```

在软件与更新中附加驱动改动

弃用： ~~sudo apt-get remove --purge nvidia* 卸载已安装的驱动，不演示了~~ ~~sudo ubuntu-drivers autoinstall 自动安装，但是我的当时安装会报错，所以我并没有用这个~~ ~~sudo apt install nvidia-driver-530 上一个我没有用，因此用的是这一个，后面的530是驱动版本号，看你自己想要选择哪一个~~

```
reboot    重启
再次输入nvidia-smi，可以显示即可
```

显示器问题：

进入bios，将SG/PEG修改为discreate模式

安装pip：

```
sudo apt install python3-pip python3-dev
python3 -m pip install --upgrade pip
```

安装gcc：

```
sudo apt install gcc
```

安装miniconda：

官网安装指令:

```
mkdir -p ~/miniconda3
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-
x86_64.sh -O ~/miniconda3/miniconda.sh
bash ~/miniconda3/miniconda.sh -b -u -p ~/miniconda3
rm -rf ~/miniconda3/miniconda.sh
~/miniconda3/bin/conda init bash
~/miniconda3/bin/conda init zsh
重启终端即可
```

1. 相关指令:

https://blog.csdn.net/menc15/article/details/71477949

1.1. 获取版本号

```
conda --version
```

或

```
conda -V
```

1.2. 获取帮助

```
conda --help
conda -h
```

查看某一命令的帮助，如 update 命令及 remove 命令

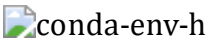
```
conda update --help
conda remove --help
```

同理，以上命令中的 `--help` 也可以换成 `-h`。

1.3. 环境管理

查看环境管理的全部命令帮助

```
conda env -h
```


conda-env-h

创建环境

```
conda create --name your_env_name
```

输入 `y` 确认创建。

创建制定 python 版本的环境

```
conda create --name your_env_name python=2.7
conda create --name your_env_name python=3
conda create --name your_env_name python=3.5
```

创建包含某些包的环境

```
conda create --name your_env_name numpy scipy
```

创建指定 python 版本下包含某些包的环境

```
conda create --name your_env_name python=3.5 numpy scipy
```

列举当前所有环境

```
conda info --envs
conda env list
```

进入某个环境

```
activate your_env_name
```

退出当前环境

```
deactivate
```

复制某个环境

```
conda create --name new_env_name --clone old_env_name
```

删除某个环境

```
conda remove --name your_env_name --all
```

1.4. 分享环境

如果你想把你当前的环境配置与别人分享，这样 ta 可以快速建立一个与你一模一样的环境（同一个版本的 python 及各种包）来共同开发/进行新的实验。一个分享环境的快速方法就是给 ta 一个你的环境的 `.yml` 文件。

首先通过 `activate target_env` 要分享的环境 `target_env`，然后输入下面的命令会在当前工作目录下生成一个 `environment.yml` 文件，

```
conda env export > environment.yml
```

小伙伴拿到 `environment.yml` 文件后，将该文件放在工作目录下，可以通过以下命令从该文件创建环境

```
conda env create -f environment.yml
```

`.yml` 是这个样子的
 这里写图片描述

当然，你也可以手写一个 `.yml` 文件用来描述或记录你的 python 环境。

1.5. 包管理

列举当前活跃环境下的所有包

```
conda list
```

列举一个非当前活跃环境下的所有包

```
conda list -n your_env_name
```

为指定环境安装某个包

```
conda install -n env_name package_name
```

如果不能通过 conda install 来安装，可以用 pip 直接安装。Pip 在 Miniconda 中已安装好，不需要单独为每个环境安装 pip。如需要用 pip 管理包，activate 环境后直接使用即可。

2. 添加环境变量：

```
sudo gedit ~/.bashrc
export PATH=/home/XXX/anaconda3/bin:$PATH（XXX为自己的用户名）（在文件末尾处添加该语句）
source ~/.bashrc
```

3. 换源：

源的网站：[[https://mirror.tuna.tsinghua.edu.cn/]]

注意：部分机器换源后可能导致 conda 无法使用，此时不再进行换源即可

```
sudo gedit ~/.condarc
```

添加：

```
channels:
  - defaults
show_channel_urls: true
default_channels:
  - https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/main
  - https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/r
  - https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/msys2
custom_channels:
  conda-forge: https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud
  msys2: https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud
  bioconda: https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud
  menpo: https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud
  pytorch: https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud
  pytorch-lts: https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud
  simpleitk: https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud
  deepmodeling: https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud/
  运行 conda clean -i 清除索引缓存，保证用的是镜像站提供的索引
```

4. conda安装cuda，pytorch等：

```
conda install pytorch torchvision cudatoolkit=11.8
```

在cudnn官网下载cudnn对应版本： https://developer.nvidia.com/rdp/cudnn-archive#a-collapse805-111

将下载解压的文件中的lib中的所有文件均复制到conda的虚拟环境中的lib中

弃用：

~~安装cuda：~~ ~~nvidia-smi查看最高支持版本~~ ~~搜索cuda官网，选择对应版本 https://developer.nvidia.com/cuda-toolkit-archive~~ ~~复制粘贴官网指令到终端运行~~ ~~选择continue->输入accept->按空格取消Driver选项->选择Install安装。~~ ~~sudo gedit ~/.bashrc~~ ~~export PATH=$PATH:/usr/local/cuda/bin~~ ~~export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/cuda/lib64~~ ~~export LIBRARY_PATH=$LIBRARY_PATH:/usr/local/cuda/lib64~~ ~~source ~/.bashrc~~ ~~查看版本：nvcc -V~~~~ ~~安装cudnn：~~ ~~搜索cudnn官网，选择对应版本 https://developer.nvidia.com/rdp/cudnn-archive#a-collapse805-111~~ ~~修改文件夹权限：~~ ~~cd /usr/local/cuda~~ ~~sudo chmod 666 include~~ ~~下载tar，解压到当前文件夹~~ ~~使用官网的提示文档下载：https://docs.nvidia.com/deeplearning/cudnn/install-guide/index.html#installlinux-tar~~ ~~查看版本号（下载文件在local中）：sudo cat /usr/local/cuda/include/cudnn_version.h | grep~~

~~CUDNN_MAJOR -A 2~~ ~~安装pytouch：~~ ~~pytouch官网：https://pytorch.org/get-started/locally/~~ ~~复制粘贴官网给的命令~~ ~~官网给的命令如果安装比较慢可以使用此命令：pip install -i https://pypi.tuna.tsinghua.edu.cn/simple torch torchvision torchaudio~~

安装编程软件 (python)：

官网：

1. vscode: https://code.visualstudio.com/Download 下载deb并使用安装器安装**（推荐）**
2. pycharm:

- 官网下载：https://www.jetbrains.com/pycharm/download/#section=linux 下载tar.gz
- 指令安装： sudo snap install [pycharm-professional|pycharm-community] --classic

3. snap商店（即Ubuntu software）：打开安装即可

安装tensorflow：

vscode中使用pip进行安装

安装opencv (C++，3.3.1):

https://blog.csdn.net/wss794/article/details/124850094

1. 下载依赖：

```
sudo apt-get install build-essential
sudo apt-get install cmake git libgtk2.0-dev pkg-config libavcodec-
dev libavformat-dev libswscale-dev
sudo apt-get install python-dev python-numpy libtbb2 libtbb-dev
libjpeg-dev libpng-dev libtiff-dev libjasper-dev libdc1394-22-dev
```

- 报错E: 无法定位软件包 libjasper-dev： sudo apt install libjasper1 libjasper-dev（仍然报错）
- 解决方式：

```
sudo add-apt-repository "deb http://security.ubuntu.com/ubuntu
xenial-security main"
sudo apt-get update
sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys
```

```
3B4FE6ACC0B21F32
sudo apt update
sudo apt install libjasper1 libjasper-dev
sudo apt-get install python-dev python-numpy libtbb2 libtbb-dev
libjpeg-dev libpng-dev libtiff-dev libjasper-dev libdc1394-22-dev
```

2. 进行安装：

1. 将下载并解压好的opencv放好位置，进入下载并解压好的opencv文件夹中

2. 暂时注释掉：与 conda 有关的所有代码:

```
sudo gedit ~/.bashrc
source ~/.bashrc
```

3. 进入opencv/modules/videoio/src/cap_ffmpeg_impl.hpp文件，添加：

```
#define AV_CODEC_FLAG_GLOBAL_HEADER (1 << 22)
#define CODEC_FLAG_GLOBAL_HEADER AV_CODEC_FLAG_GLOBAL_HEADER
#define AVFMT_RAWPICTURE 0x0020
```

4. 在opencv-3.3.1文件夹下新建build文件夹(注意不是安装包opencv的目录下):

```
cd build
sudo cmake ..
sudo make -j8
sudo make install
```

3. 进行配置：

3.1. 确认路径：

```
sudo find / -iname opencv.pc
路径为：/usr/local/lib/pkgconfig/opencv.pc
```

3.2. 路径添加：

```
sudo gedit /etc/profile.d/pkgconfig.sh（pkgconfig.sh为一个新的文件）
```

- 添加：

```
export PKG_CONFIG_PATH=/usr/local/lib/pkgconfig:$PKG_CONFIG_PATH
```

- 环境：

```
source /etc/profile
```

- 验证：

```
pkg-config --libs opencv
```

3.3. 配置动态库：

```
sudo gedit /etc/ld.so.conf.d/opencv.conf（新建的文件）
添加：
    /usr/local/lib
sudo ldconfig
```

3.4. 验证：

```
cd ~/opencv/samples/cpp/example_cmake
cmake .
make
./opencv_example
```

ROS 安装：

首先默认换源等操作已经完成。

1. 配置公钥：

```
sudo apt-key adv --keyserver 'hkp://keyserver. Ubuntu. Com: 80' --
recv-key C 1 CF 6 E 31 E 6 BADE 8868 B 172 B 4 F 42 ED 6 FBAB 17 C
```

```
654
```

2. 添加 ROS 源：

```
sudo sh -c '. /etc/lsb-release && echo "d{2d3b390d-f615-41d0-a830-
7bbdcedbd397}eb http://mirrors.ustc.edu.cn/ros/ubuntu/ `lsb_release
-cs` main" > /etc/apt/sources.list.d/ros-latest.list'
sudo sh -c '. /etc/lsb-release && echo "deb
http://mirrors.tuna.tsinghua.edu.cn/ros/ubuntu/ `lsb_release -cs`
main" > /etc/apt/sources.list.d/ros-latest.list'
sudo apt-get update
sudo apt-get upgrade
```

3. 设置编码：

```
sudo apt install locales
sudo locale-gen en_US en_US.UTF-8
sudo update-locale LC_ALL=en_US.UTF-8 LANG=en_US.UTF-8
export LANG=en_US.UTF-8
```

4. 进行安装：

4.1. 注意版本：

Ubuntu	Ros1	Ros2
16.04 LTS	Kinetic LTS	Ardent
18.04 LTS	Melodic LTS	Dashing LTS
20.04 LTS	Noetic LTS	Foxy LTS
22.04 LTS	Noetic Ninjemys	Humble Hawksbill

```
sudo aptitude install ros-noetic-desktop-full
```

5. 设置环境变量：

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

6. 安装依赖：

```
sudo apt install python3-rosinstall python3-rosinstall-generator
python3-wstool -y
sudo apt install python3-roslaunch -y
```

7. 测试：

- 重启终端
- 启动 ROS: `roscore` 如果打印以下内容，则认为成功：

```
root@DESKTOP-7PBML8U:/home/ssr# roscore
... logging to /root/.ros/log/ad1b121e-a611-11ee-89e9-
57487a114ada/roslaunch-DESKTOP-7PBML8U-822.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://DESKTOP-7PBML8U:37425/
ros_comm version 1.16.0


SUMMARY
========

PARAMETERS
 * /rosdistro: noetic
 * /rosversion: 1.16.0

NODES

auto-starting new master
process[master]: started with pid [832]
ROS_MASTER_URI=http://DESKTOP-7PBML8U:11311/

setting /run_id to ad1b121e-a611-11ee-89e9-57487a114ada
process[rosout-1]: started with pid [842]
started core service [/rosout]
```

- 再开一个控制台窗口，输入小海龟启动命令：

```
rosrun turtlesim turtlesim_node
```

- 然后再开一个窗口，启动小海龟控制节点：

```
rosrun turtlesim turtle_teleop_key
```

- 使用方向键可以控制小乌龟移动，则认为配置成功

Docker 安装：

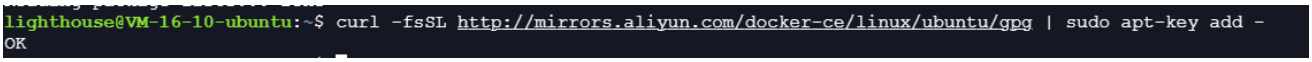
建议使用国内云，参考博客 https://www.cnblogs.com/Can-daydayup/p/16472375.html

1. 卸载可能存在的或者为安装成功的Docker版本

```
sudo apt-get remove docker docker-engine docker-ce docker.io
```

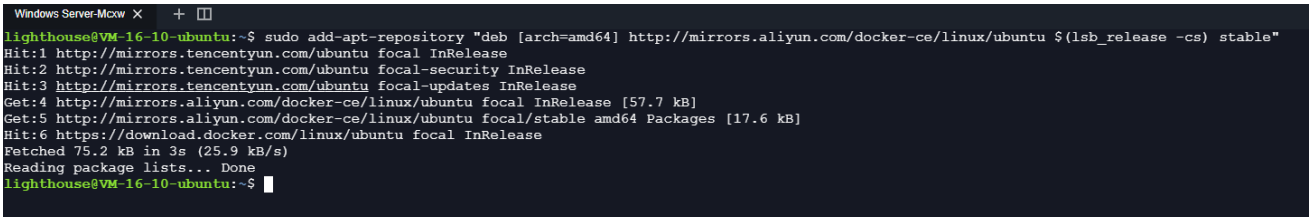
2. 添加阿里云的GPG密钥

```
curl -fsSL http://mirrors.aliyun.com/docker-ce/linux/ubuntu/gpg |
sudo apt-key add -
```



3. 使用以下命令设置存储库

```
sudo add-apt-repository "deb [arch=amd64]
http://mirrors.aliyun.com/docker-ce/linux/ubuntu $(lsb_release -cs)
stable"
```



4. 安装最新版本的Docker(飞速安装)

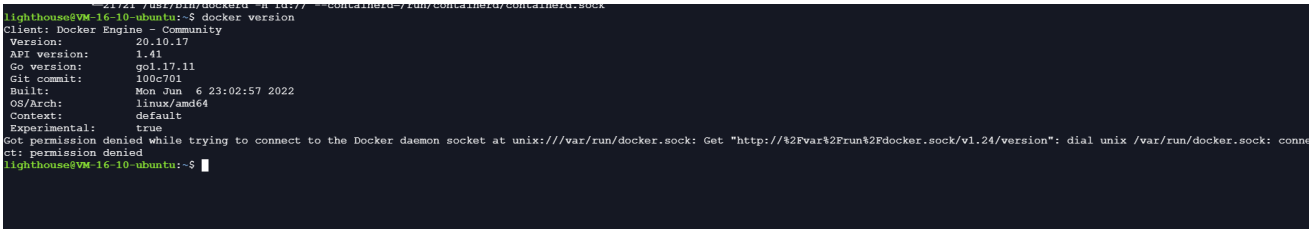
```
sudo apt-get update`
```

```
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-compose-plugin
```

5. 验证Docker是否安装成功

5.1. 查看版本号：

```
docker version
```

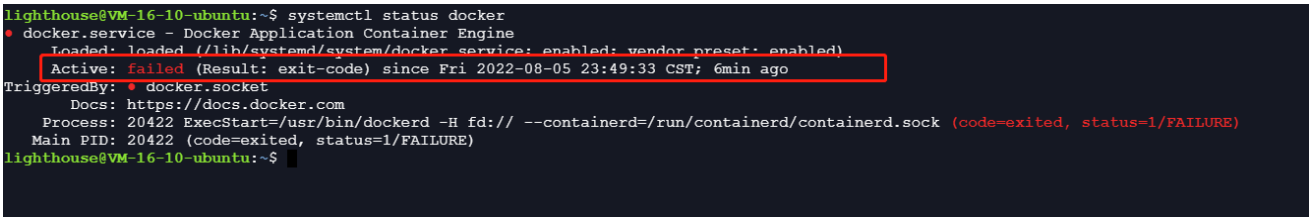


6. 启动Docker

6.1. 安装完成后，运行如下命令验证 Docker 服务是否在运行

```
systemctl status docker
```

未运行：



6.2. 运行以下命令启动Docker服务

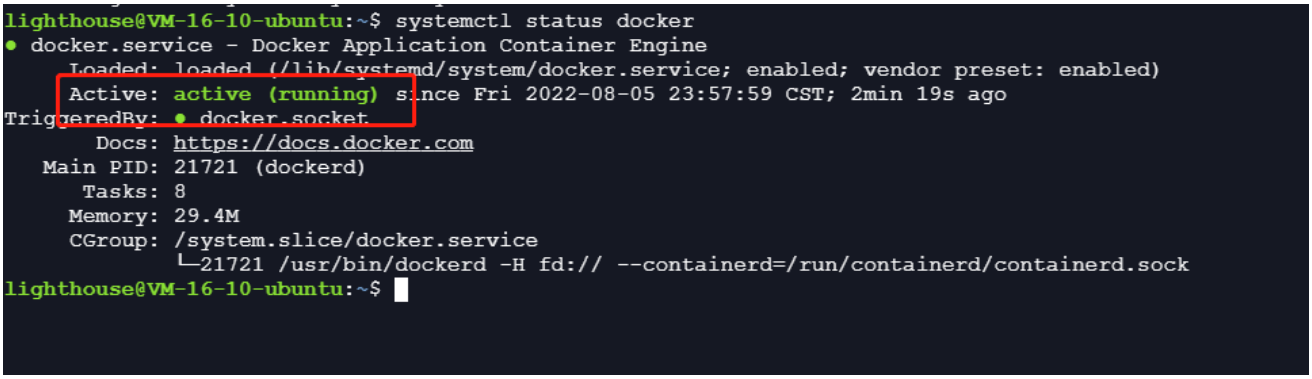
```
sudo systemctl start docker
```

6.3. 设置Docker服务在每次开机时自动启动

```
sudo systemctl enable docker
```

6.4. 查看docker运行状态

```
systemctl status docker
```



7. 验证Docker是否运行正常

注意：执行下面的命令会下载一个Docker测试镜像，并在容器中执行一个“hello-world”样例程序。

```
sudo docker run hello-world
```

如果你看到类似下方的输出，那么祝贺你，Docker能够正常运行在你的Ubuntu系统中了。

```
ubuntu@VM-16-10-ubuntu:~$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest:
sha256:53f1bbee2f52c39e41682ee1d388285290c5c8a76cc92b42687eecf38e0af
3f0
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working
correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker
Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which
runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client,
```

```
which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container
with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/
For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

时间同步问题：

Step 1:

- Windows：控制面板，日期和时间, 同步

Step 2:

- Ubuntu：

```
sudo apt-get install ntpdate
sudo ntpdate time.windows.com
sudo hwclock --localtime --systohc
```

卸载Ubuntu引导项：

```
diskpart
list disk
select disk 0
list partition
select partition 1
assign letter=J
管理员打开写字板，打开j盘文件，删除ubuntu文件夹
remove letter=J
```