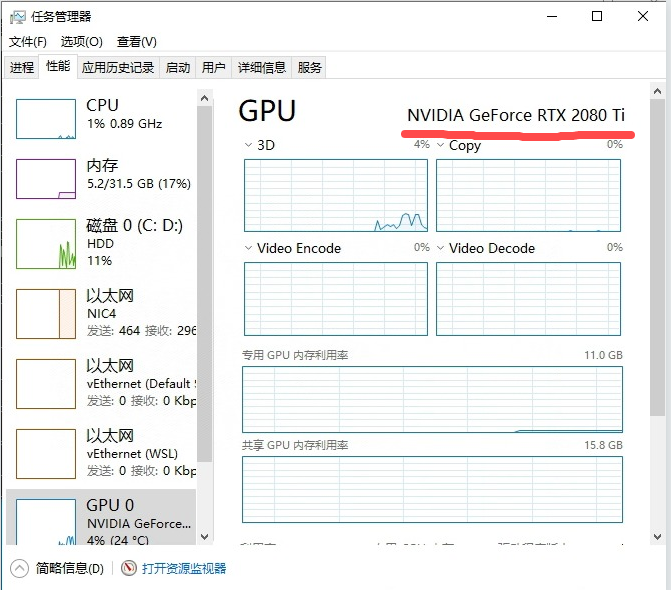
#### Windows 安装Nvidia-Docker GPU 驱动PaddlePaddle

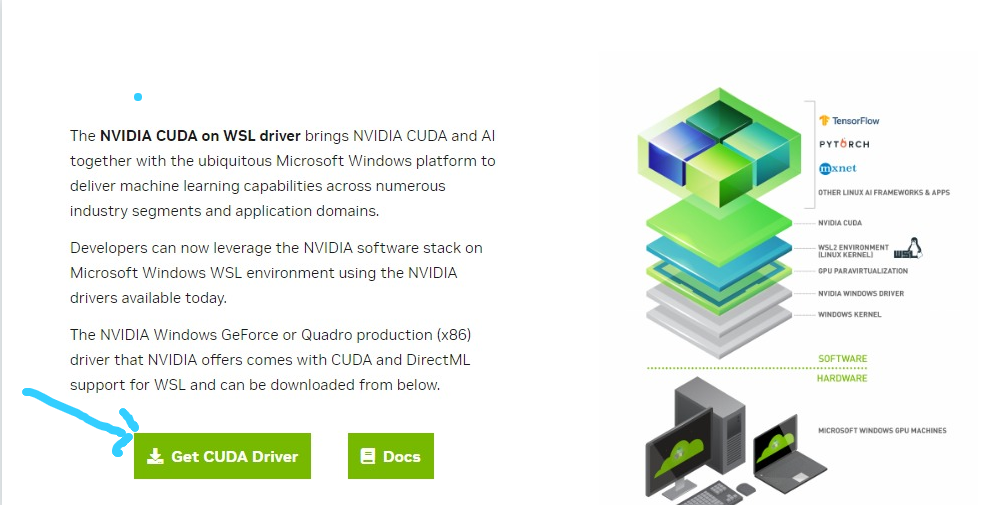
1. 安装最新显卡驱动

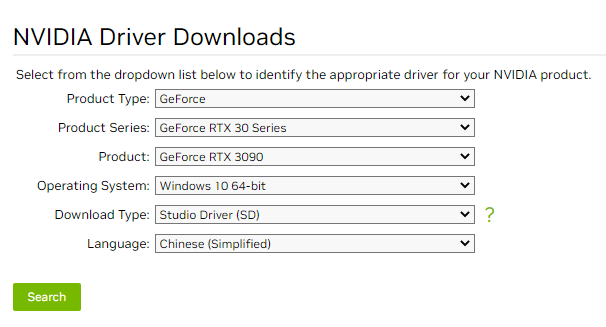
注意一定要在Windows上安装带WSL 的显卡驱动， 不要在WSL Ubuntu中安装显卡驱动。

查看自己的显卡型号:



上去Nvidia 下载自己型号的驱动: <https://developer.nvidia.com/cuda/wsl>





Download Type 选择SD  
 Game Ready Drivers: you are a gamer who prioritizes day of launch support for the latest games, patches, and DLCs.  
 Studio Drivers: you are a content creator who prioritizes stability and quality for creative workflows including video editing, animation, photography, graphic design, and livestreaming.

等待Windows 显卡驱动安装即可, 重启电脑。

1. 安装和配置WSL2

手动安装WSL

可参考《舊版WSL 的手動安裝步驟》:

<https://learn.microsoft.com/zh-cn/windows/wsl/install-manual>

下载 Linux 内核更新包&安装：

<https://wslstorestorage.blob.core.windows.net/wslblob/wsl_update_x64.msi>

开启和配置WSL,打开cmd输入指令：

# 启动管理员PowerShell

Start-Process powershell -Verb runAs

在新弹出的PowerShell中输入

# 启用 wsl 低於18362 的版本不支持WSL 2

dism.exe /online /enable-feature /featurename:Microsoft-Windows-Subsystem-Linux /all /norestart

# 启用虚拟机

dism.exe /online /enable-feature /featurename:VirtualMachinePlatform /all /norestart

# 设置 wsl 默认版本

wsl --set-default-version 2

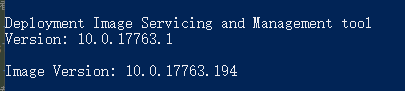
如果报错：命令列選項無效: --set-default-version

没有升级Linux 内核，文上有URL 下载安装即可

如果报错: 红色一大串

查看Windows版本是否太低

输入启动WSL可以看到，太低则升级Windows



选择子系统， 推介 Ubuntu 20.04LTS

/resource/CanonicalGroupLimited.UbuntuonWindows\_2004.2021.825.0.AppxBundle

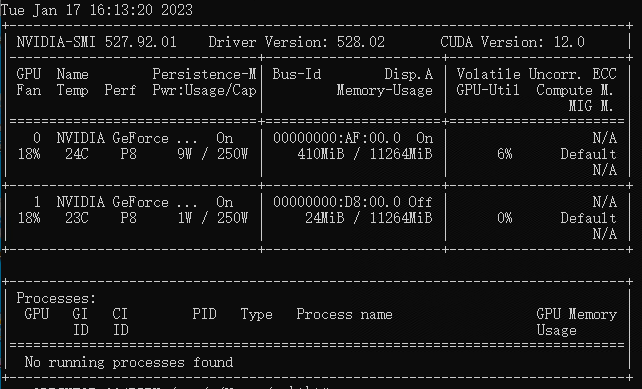
双击安装即可

可选操作

避免占用太多系统盘内容，把Linux系统搬到其他盘

安装好后继续在PowerShell 输入指令

# 子系统打包 移动到自己的文件夹  
  
# 查看列表  
wsl --list  
# wsl --export <导出的系统名称> <导出的位置>  
wsl --export Ubuntu D:\WSL\Ubuntu.tar  
# 卸载 Ubuntu   
wsl --unregister Ubuntu  
# 导入 <名字> <安装路径> <tar 路径>  
wsl --import Ubuntu C:\Ubuntu D:\backUp\Ubuntu.tar  
wsl --list  
  
# 进入子系统  
bash  
# 查看是否有显卡驱动  
nvidia-smi



1. 安装CUDA

这里安装CUDA 固定使用11.7.0版本

选择Linux -> x86\_64 -> WSL-Ubuntu -> 2.0 runfile(local)

如果apt-get安装太慢可选择更换apt源

# apt-get  
# 可选操作，如果国内网速太慢请更换国内代理  
# 备份源文件  
sudo cp /etc/apt/sources.list /etc/apt/sources.list.bak  
# 编辑源列表文件  
sudo vim /etc/apt/sources.list  
  
# 删除里面全部内容  
# 添加以下内容  
deb http://mirrors.aliyun.com/ubuntu/ bionic main restricted universe multiverse  
deb-src http://mirrors.aliyun.com/ubuntu/ bionic main restricted universe multiverse  
deb http://mirrors.aliyun.com/ubuntu/ bionic-security main restricted universe multiverse  
deb-src http://mirrors.aliyun.com/ubuntu/ bionic-security main restricted universe multiverse  
deb http://mirrors.aliyun.com/ubuntu/ bionic-updates main restricted universe multiverse  
deb-src http://mirrors.aliyun.com/ubuntu/ bionic-updates main restricted universe multiverse  
deb http://mirrors.aliyun.com/ubuntu/ bionic-backports main restricted universe multiverse  
deb-src http://mirrors.aliyun.com/ubuntu/ bionic-backports main restricted universe multivers  
deb http://mirrors.aliyun.com/ubuntu/ bionic-proposed main restricted universe multiverse  
deb-src http://mirrors.aliyun.com/ubuntu/ bionic-proposed main restricted universe multiverse  
  
# 更新一下apt-get 源  
sudo apt-get update

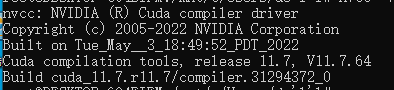
安装CUDA

注意这里有任何软连接错误请无视！！！！！

# 安装gcc Cuda 安装需要  
apt install -y build-essential  
# 提示软链接错误无需理会  
# 下载和安装  
wget https://developer.download.nvidia.com/compute/cuda/11.7.0/local\_installers/cuda\_11.7.0\_515.43.04\_linux.run  
sudo sh cuda\_11.7.0\_515.43.04\_linux.run  
  
#accept 全选安装即可

修改环境变量

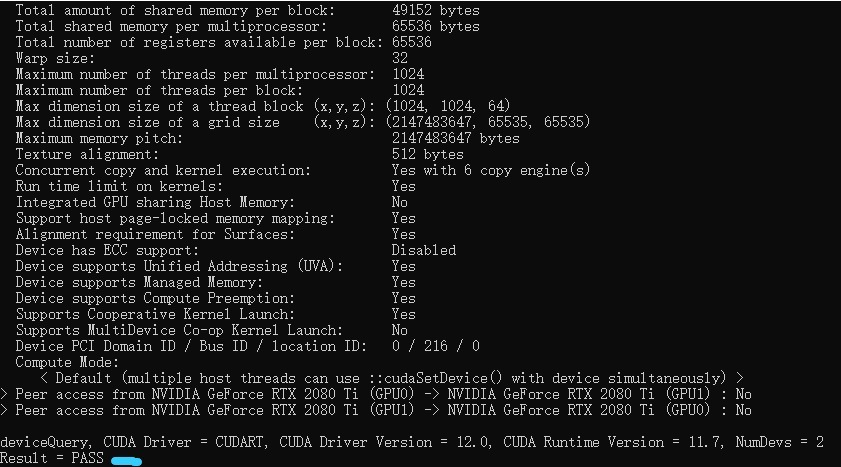
# 修改环境变量  
vim ~/.bashrc  
  
# 文件未追加  
export PATH=/usr/local/cuda/bin${PATH:+:${PATH}}  
export LD\_LIBRARY\_PATH=/usr/local/cuda/lib64${LD\_LIBRARY\_PATH:+:${LD\_LIBRARY\_PATH}}  
  
# reload 环境变量配置  
source ~/.bashrc  
  
# 检查是否生效  
nvcc -V



显示这个内容则安装成功

测试CUDA

# 测试cuda   
apt install -y git  
cd /home  
git clone https://github.com/NVIDIA/cuda-samples.git  
cd /home/cuda-samples/Samples/1\_Utilities/deviceQuery  
make  
./deviceQuery  
# 输出Pass 则成功了



I.没有Pass则显卡驱动可能未配置好，尝试重启 or 重新安装

Ii.显示cuda-samples nvcc fatal : Unsupported gpu architecture 'compute\_90'

则不支持 90算力。 进入/home/cuda-samples/Samples/1\_Utilities/deviceQuery

修改[Makefile](https://github.com/NVIDIA/cuda-samples/blob/master/Samples/1_Utilities/deviceQuery/Makefile" \o "Makefile) 文件

删除282 行的 90

删除284 行的 90

1. Nvidia Docker 安装

这里安装Nvidia Docker 安装docker 核心 以及nvidia-docker2就行

# 更新apt 源  
curl https://get.docker.com | sh   
distribution=$(. /etc/os-release;echo $ID$VERSION\_ID) \  
 && curl -s -L https://nvidia.github.io/nvidia-docker/gpgkey | sudo apt-key add - \  
 && curl -s -L https://nvidia.github.io/nvidia-docker/$distribution/nvidia-docker.list | sudo tee /etc/apt/sources.list.d/nvidia-docker.list  
sudo apt update  
sudo apt-get install nvidia-docker2  
service docker start

docker ps

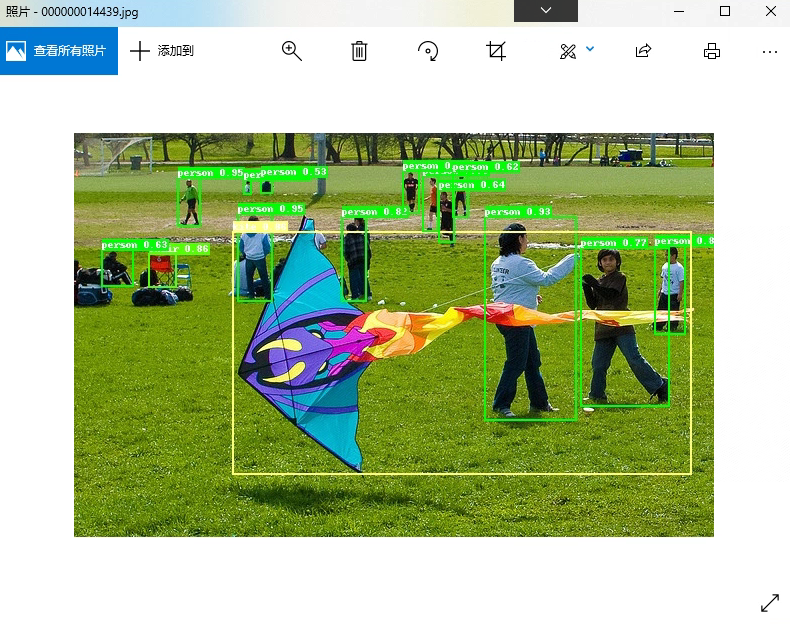
这里基本上不会有太多的问题

# nvidia-docker 测试  
sudo docker run -idt --name nvidia\_docker\_test --gpus all --shm-size 16G nvidia/cuda:11.7.1-base-ubuntu22.04  
sudo nvidia-docker start nvidia\_docker\_test  
sudo nvidia-docker attach nvidia\_docker\_test  
# 查看是否有显卡驱动  
nvidia-smi  
# 有则判定 nvidia-docker 已经成功安装和使用  
exit

1. 使用PaddleDetection镜像和测试

# 拉取PaddlePaddle项目测试  
cd /home  
git clone https://github.com/PaddlePaddle/PaddleDetection.git  
  
# 进入paddlepaddle 镜像docker  
docker run --gpus all --shm-size=1g --ulimit \  
memlock=-1 -it --name Test -v /home/PaddleDetection:/home/PaddleDetection  
--rm nvcr.io/nvidia/paddlepaddle:22.10-py3  
cd /home/PaddleDetection  
  
# 安装pip 依赖 这里使用了代理  
pip install -r requirements.txt -i https://pypi.tuna.tsinghua.edu.cn/simple  
# 在GPU上预测一张图片  
export CUDA\_VISIBLE\_DEVICES=0  
python tools/infer.py -c configs/ppyolo/ppyolo\_r50vd\_dcn\_1x\_coco.yml -o use\_gpu=true weights=https://paddledet.bj.bcebos.com/models/ppyolo\_r50vd\_dcn\_1x\_coco.pdparams --infer\_img=demo/000000014439.jpg

打开\home\PaddleDetection\output 里面有一张输出图



有任何安装问题可以联络Wechat： MoJeffrey