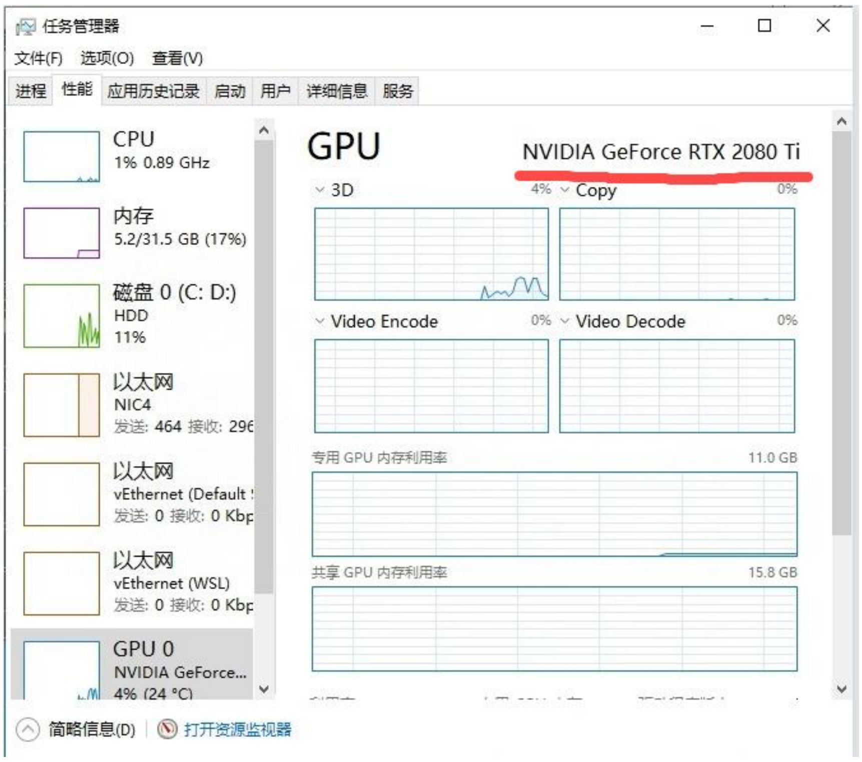
Windows 安装 Nvidia-Docker GPU 驱动 PaddlePaddle

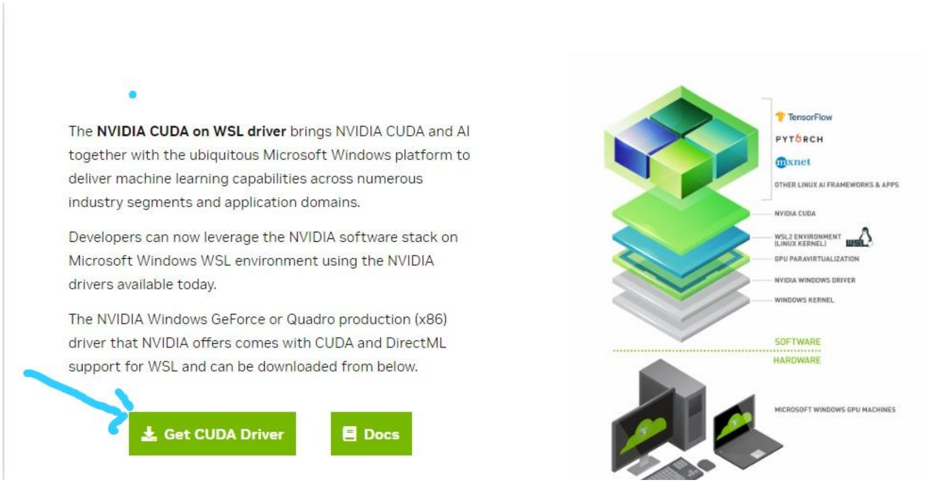
1. 安装最新显卡驱动

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:



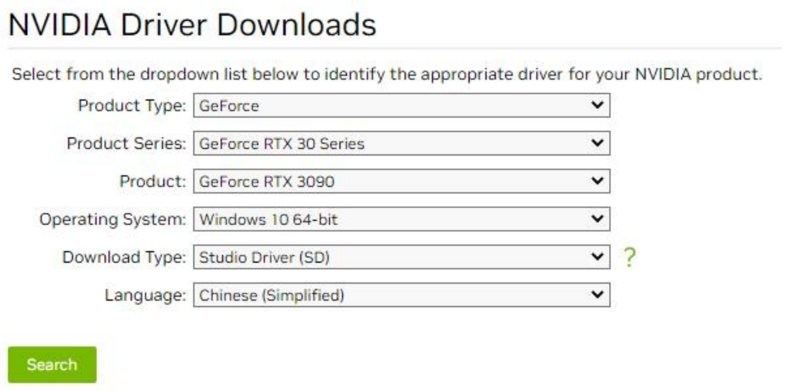
上去 Nvidia 下载自己型号的驱动: https://developer.nvidia.com/cuda/wsl



The **NVIDIA CUDA on WSL driver** brings NVIDIA CUDA and AI together with the ubiquitous Microsoft Windows platform to deliver machine learning capabilities across numerous industry segments and application domains.

Developers can now leverage the NVIDIA software stack on Microsoft Windows WSL environment using the NVIDIA drivers available today.

The NVIDIA Windows GeForce or Quadro production (x86) driver that NVIDIA offers comes with CUDA and DirectML support for WSL and can be downloaded from below.


NVIDIA Driver Downloads

Select from the dropdown list below to identify the appropriate driver for your NVIDIA product.

Product Type:	GeForce
Product Series:	GeForce RTX 30 Series
Product:	GeForce RTX 3090
Operating System:	Windows 10 64-bit
Download Type:	Studio Driver (SD)
Language:	Chinese (Simplified)

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。

2. 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：

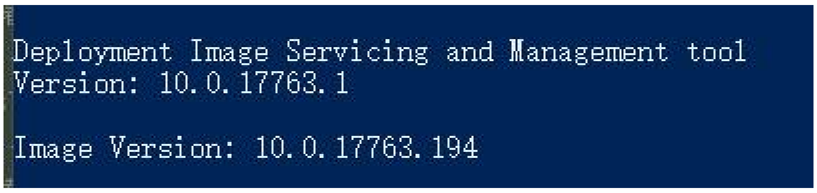
```
# 启动管理员 PowerShell
Start-Process powershell -Verb runAs
```

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



```
Deployment Image Servicing and Management tool
Version: 10.0.17763.1

Image Version: 10.0.17763.194
```

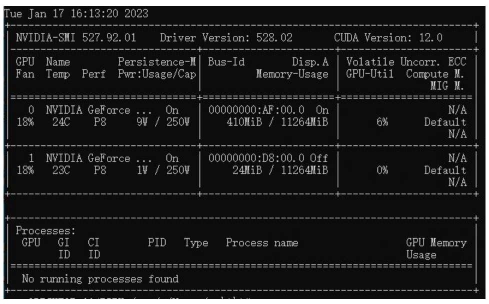
选择子系统， 推介 Ubuntu 20.04LTS
/resource/CanonicalGroupLimited.UbuntuonWindows_2004.2021.825.0.AppxBundle
双击安装即可

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
```



```
Tue Jan 17 16:13:20 2023
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 527.92.01    Driver Version: 528.02       CUDA Version: 12.0     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  NVIDIA GeForce ...  On   | 00000000:AF:00.0  On |                  N/A |
| 18%   24C    P8     9W / 250W |    410MiB / 11264MiB |      6%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   1  NVIDIA GeForce ...  On   | 00000000:D8:00.0 Off |                  N/A |
| 18%   23C    P8     1W / 250W |     24MiB / 11264MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

3. 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_insta
llers/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
```

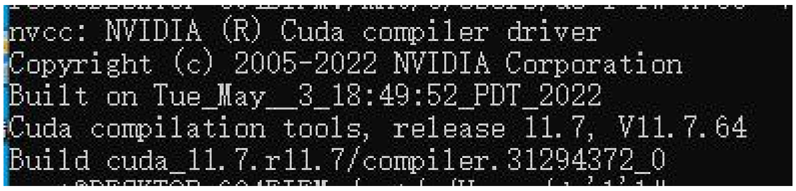
修改环境变量

```
# 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
```

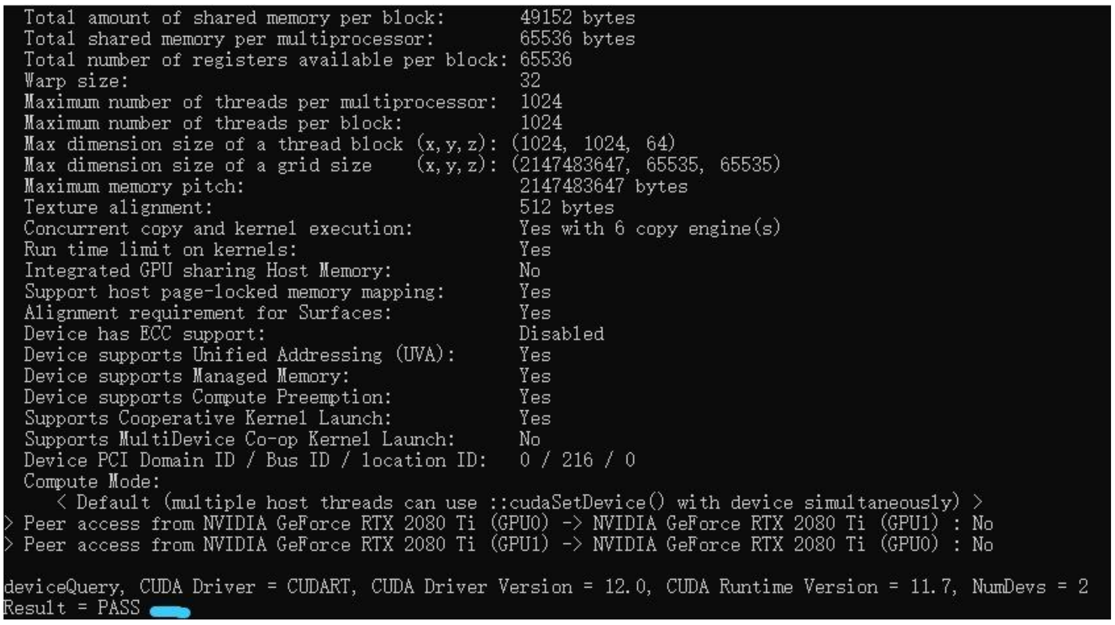


```
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2022 NVIDIA Corporation
Built on Tue_May__3_18:49:52_PDT_2022
Cuda compilation tools, release 11.7, V11.7.64
Build cuda_11.7.r11.7/compiler.31294372_0
```

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
```



```
  Total amount of shared memory per block:       49152 bytes
  Total shared memory per multiprocessor:        65536 bytes
  Total number of registers available per block: 65536
  Warp size:                                     32
  Maximum number of threads per multiprocessor:  1024
  Maximum number of threads per block:           1024
  Max dimension size of a thread block (x,y,z): (1024, 1024, 64)
  Max dimension size of a grid size    (x,y,z): (2147483647, 65535, 65535)
  Maximum memory pitch:                          2147483647 bytes
  Texture alignment:                             512 bytes
  Concurrent copy and kernel execution:          Yes with 6 copy engine(s)
  Run time limit on kernels:                     Yes
  Integrated GPU sharing Host Memory:            No
  Support host page-locked memory mapping:       Yes
  Alignment requirement for Surfaces:            Yes
  Device has ECC support:                        Disabled
  Device supports Unified Addressing (UVA):      Yes
  Device supports Managed Memory:                Yes
  Device supports Compute Preemption:            Yes
  Supports Cooperative Kernel Launch:            Yes
  Supports MultiDevice Co-op Kernel Launch:      No
  Device PCI Domain ID / Bus ID / location ID:   0 / 216 / 0
  Compute Mode:
     < Default (multiple host threads can use ::cudaSetDevice() with device simultaneously) >
> Peer access from NVIDIA GeForce RTX 2080 Ti (GPU0) -> NVIDIA GeForce RTX 2080 Ti (GPU1) : No
> Peer access from NVIDIA GeForce RTX 2080 Ti (GPU1) -> NVIDIA GeForce RTX 2080 Ti (GPU0) : No

deviceQuery, CUDA Driver = CUDART, CUDA Driver Version = 12.0, CUDA Runtime Version = 11.7, NumDevs = 2
Result = PASS
```

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 文件

删除 282 行的 90

删除 284 行的 90

4. 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.li
st | 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
```

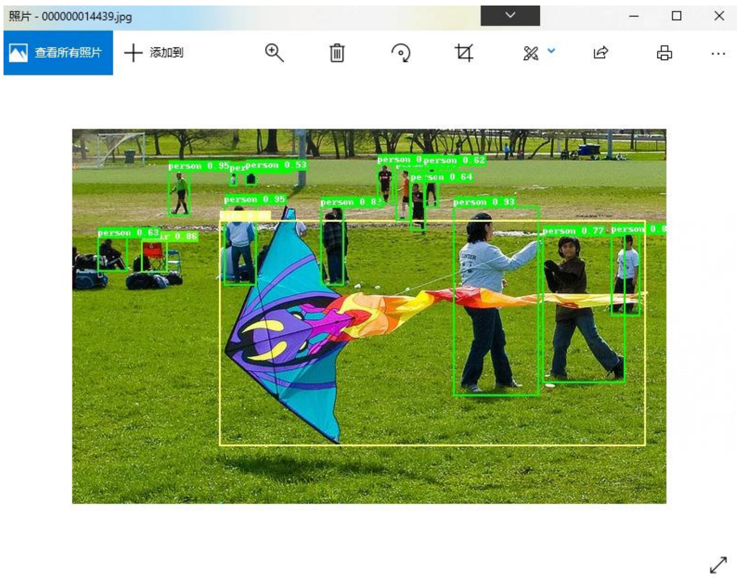
5. 使用 PaddleDetection 镜像和测试

```
# 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_co
co.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