Lab3 CUDA Basic

Nov, 2022 Parallel Programming

Overview

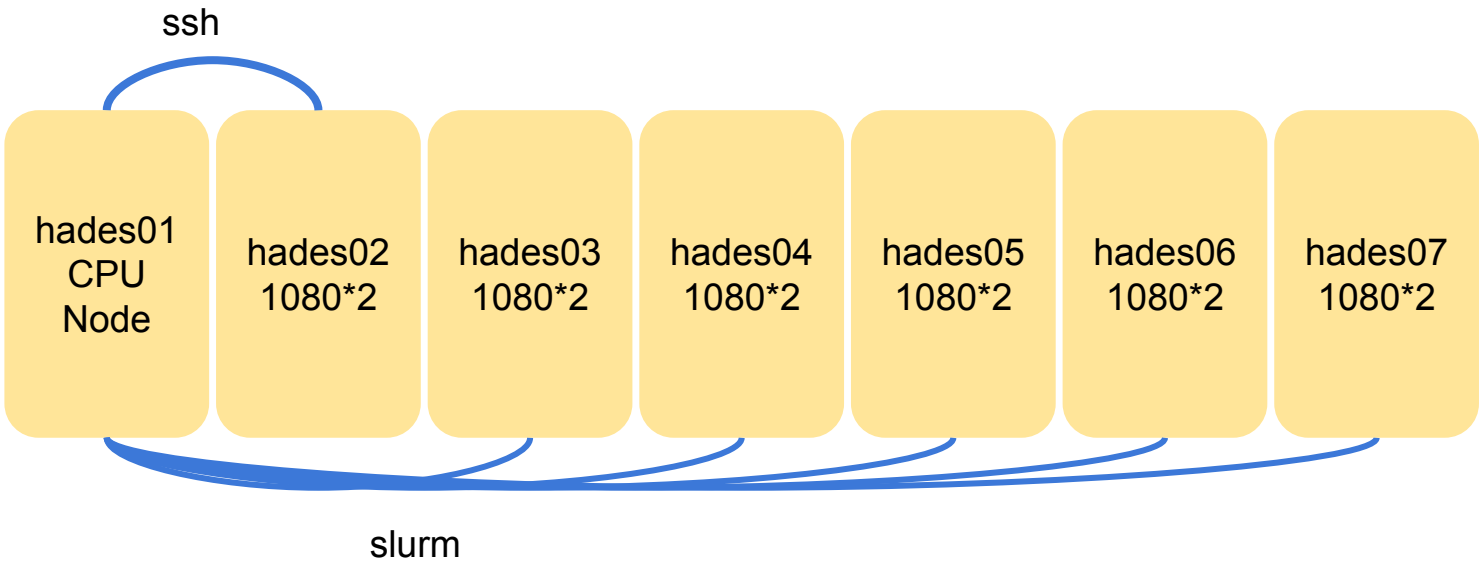
- Platform guide
  - Hades
  - NCHC Container
- Tools
- Assignment

Platform Guide (Hades)

GPU Cluster

- Host: hades.cs.nthu.edu.tw
- Account: same as apollo
- Password: same as apollo

GPU Cluster


ssh
hades01
CPU
Node
hades02
1080*2
hades03
1080*2
hades04
1080*2
hades05
1080*2
hades06
1080*2
hades07
1080*2
slurm

Job Scheduler

- SLURM
- Course partition: pp22

```
[enmingw32@hades01 ~]$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
pp22*        up       5:00      5   idle hades[03-07]
```

- Limitation
  - 2 gpus
  - 5 minutes

Access Resource

- hades02
  - ssh hades02
  - If you want to specify which GPU to use.
  - `export CUDA_VISIBLE_DEVICES=<gpu id>`
  - eg. `export CUDA_VISIBLE_DEVICES=1`
  - eg. `export CUDA_VISIBLE_DEVICES=0,1`
- hades[03-07]
  - Slurm
  - Access gpus with flag `--gres=gpu:<number of gpu>`
  - eg. `srun -n 1 --gres=gpu:1 ./executable`
  - eg. `srun -n 1 --gres=gpu:2 ./executable`
  - `Two GPUs will be executed on the same node.`

Platform Guide (NCHC CT)

NCHC Container

- Webpage: https://portal.apps.edu-cloud.nchc.org.tw
- Tutorial: https://hackmd.io/@enmingw32/pp22-nchc
- Register your account first
- GPU: RTX 3070
- Total available GPUs: 28
- Please stop your container if you aren't using it

課程名稱	課程程度	建立時間	操作
PP22 1GPU	基礎	2022 / 10 / 21	⋮
PP22 2GPUs	基礎	2022 / 11 / 01	⋮
PP22 noGPU	基礎	2022 / 11 / 03	⋮

Register

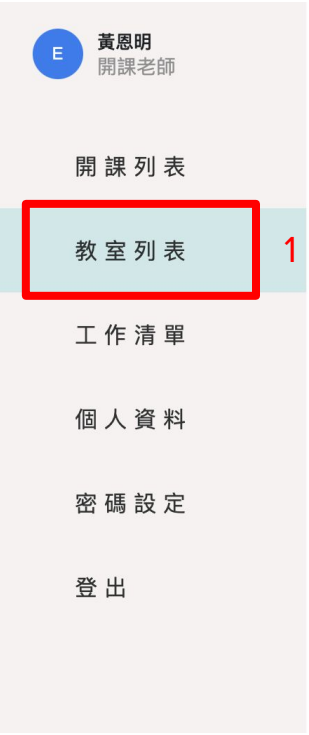

1
NARLabs 財團法人國家實驗研究院
國家高速網路與計算中心
National Center for High-performance Computing
課程介紹
基礎課程
登入 / 註冊
歡迎來到NCHC.ai
Sign in with Google
OR
2
學生註冊
管理員 / 教師申請
By continuing, you agree to NCHC.ai's Terms of Service, Privacy Policy


歡迎註冊 NCHC.ai 帳戶
Your gmail in this sheet
請輸入您欲註冊的信箱
* 中文姓名
請輸入中文姓名
* 密碼
請輸入您的密碼
* 密碼確認
請再次輸入您的密碼
註冊
取消
Enter whatever you want

Login

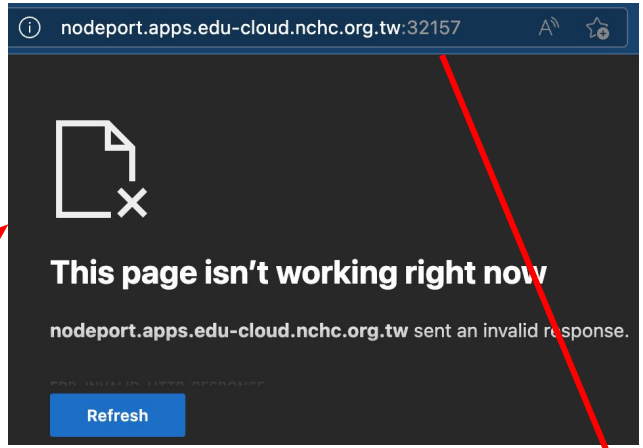

NCHC.ai
國家高速網路與計算中心
National Center for High-performance Computing
課程介紹
基礎課程
登入 / 註冊
NCHC.ai
歡迎來到NCHC.ai
Sign in with Google
OR
學生註冊
管理員 / 教師申請
By continuing, you agree to NCHC.ai's Terms of Service, Privacy Policy

Start Container



Access the Container

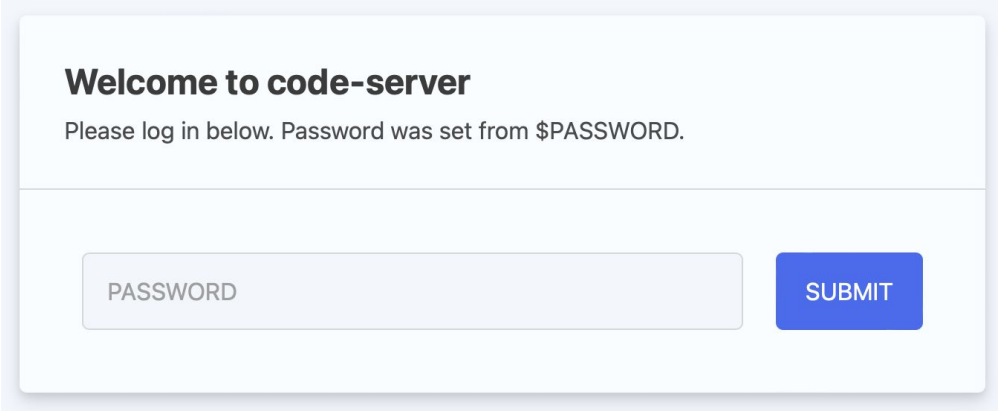
- SSH
  - The port number will be different
- Codeserver (the web version of vscode)
  - Click the “code-server” button



MobaXterm or Terminal:
ssh root@nodeport.apps.edu-cloud.nchc.org.tw -p 32157

User & Password

- User: root
- Password: student
  - The password of ssh and code-server is the same



First-time Setup Script

Open terminal in the container:

```
bash <(curl -s https://apollo.cs.nthu.edu.tw/pp22/setup-remote.sh)
```

The script will execute the following commands:

- Set proper bash config for homework and lab judger (e.g., hw3-2-judge)
- Generate ssh key and install it on Apollo (you will be prompted to enter your Apollo account name and password)

Run this script only once, even you relaunched your container (since your personal data will be kept).

```
#################################
##     Installing SSH Key.     ###
#################################
Enter your username on apollo: enmingw32
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/root/.ssh/id_rsa.pub"
The authenticity of host '140.114.91.183 (140.114.91.183)' can't be established.
ECDSA key fingerprint is SHA256:ITdmu4BUC9SDIl2+zFrENz2LyV7H+9jIdOMWGYvQFVc.
Are you sure you want to continue connecting (yes/no)? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
enmingw32@140.114.91.183's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'enmingw32@140.114.91.183'"
and check to make sure that only the key(s) you wanted were added.

  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  4681  100  4681    0     0   253k      0 --:--:-- --:--:-- --:--:--  253k
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 18583  100 18583    0     0   955k      0 --:--:-- --:--:-- --:--:--  955k
#################################
##       Install completed     ###
## Please relogin the container ###
#################################
root@78052b1f-9e60-49a7-a9ad-791bdeb43087-59cddb9d78-jmc2s:~#
```

Stop your container

- Please stop your container if you aren't using it;
  otherwise, other students may not have enough GPU resources
- Your files located under $HOME (/root/)
  will be preserved

NO MINING

Educational use only

Please cherish the computing resources we provided

Compile & run

- Compiler nvcc
  - `nvcc [options] <inputfile>`
  - eg. `nvcc cuda_code.cu -o cuda_executable`
- Run
  - Refer to the previous slide

Practice

- In this practice, you can try to run the **`deviceQuery`** on
  - hades02 or NCHC container
  - scheduler
- Compile

Hades
```
cp -r /home/pp22/share/lab3/deviceQuery $HOME
```
NCHC
```
cp -r /tmp/pp22-dataset/pp22/share/lab3/deviceQuery $HOME
cd $HOME/deviceQuery
nvcc deviceQuery.cpp -o deviceQuery
```

- Run it on
  - hades02 or NCHC container
  - scheduler (via srun)
- How many CUDA cores on GTX1080 ?

Tools

nvidia-smi

- NVIDIA System Management Interface program
- You can query details about
  - gpu type
  - gpu utilization
  - memory usage
  - temperature
  - clock rate
  - ...

nvidia-smi example

```
# michael1017 @ hades02 in ~ [15:08:34]
$ nvidia-smi
Thu Nov 12 15:08:36 2020
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 450.57       Driver Version: 450.57       CUDA Version: 11.0     |
|-------------------------------+----------------------+----------------------+
| 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  GeForce GTX 1080    On   | 00000000:4B:00.0 Off |                  N/A |
|  0%   37C    P8     7W / 200W |      1MiB /  8119MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   1  GeForce GTX 1080    On   | 00000000:4D:00.0 Off |                  N/A |
|  0%   44C    P8    14W / 200W |      1MiB /  8117MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

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

cuda-memcheck

- This tool checks memory errors of your program, and it also reports hardware exceptions encountered by the GPU. These errors may not cause program to crash, but they could result in unexpected program behavior and memory misusage.
- Useful for debugging
- Error types
  - cuda-memcheck

cuda-memcheck

```
cudaFree(device_t);
cudaFree(device_t); // free an address twice, error
```

```
[mewtwo@hades02 HW4_cuda_sobel]$ cuda-memcheck ./sobel input/candy.bmp out.bmp
========= CUDA-MEMCHECK
========= Program hit cudaErrorInvalidDevicePointer (error 17) due to "invalid device pointer" on CUDA API call to cudaFre
e.
=========     Saved host backtrace up to driver entry point at error
=========     Host Frame:/usr/lib64/nvidia/libcuda.so.1 [0x32f6a3]
=========     Host Frame:./sobel [0x454c0]
=========     Host Frame:./sobel [0x356f]
=========     Host Frame:/usr/lib64/libc.so.6 (__libc_start_main + 0xf5) [0x21b35]
=========     Host Frame:./sobel [0x36ff]
=========
========= ERROR SUMMARY: 1 error
```

cuda-gdb

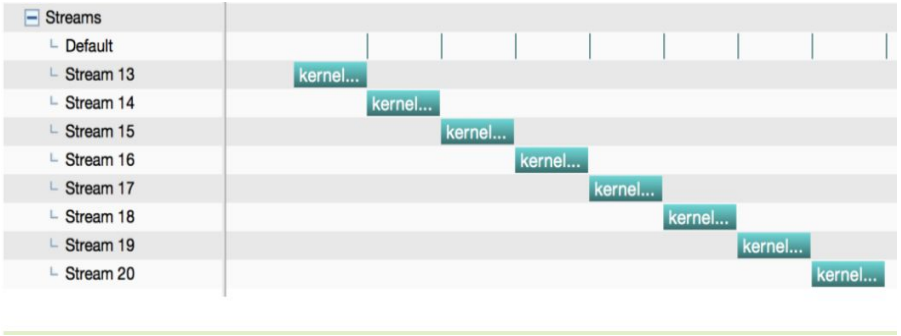
- cuda-gdb tutorial

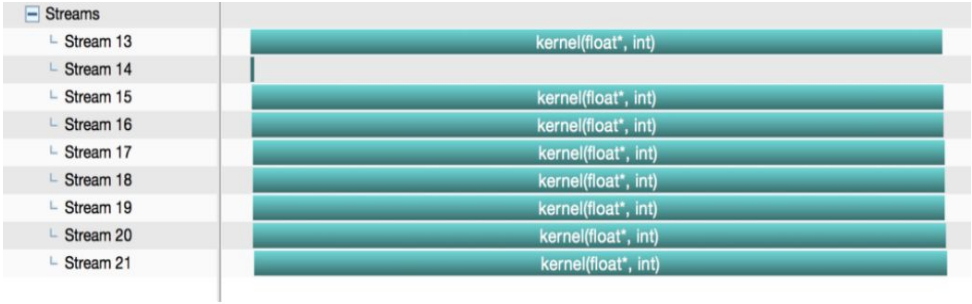
nvprof & nsight-compute

- A CUDA profiler provides feedback to optimize CUDA programs
  - `nvprof ./lab3 in.png out.png`
  - `-o <FILE>` to save result to a file
  - `-i <FILE>` to read result from a file
- GTX 1080: nvprof
- RTX 3070: nsight-compute (command: `ncu`)

nvvp

- [nvvp-tutorial]
- GUI version of nvprof
- Useful for the stream optimization
  - Timeline



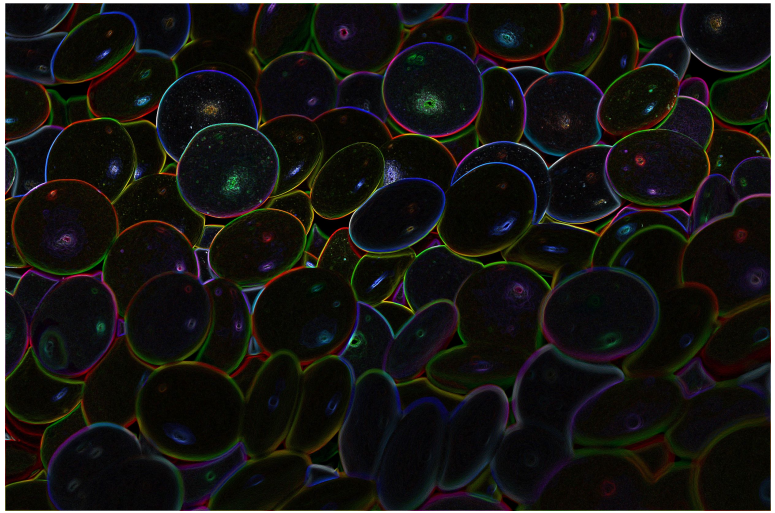


nvvp is useful for checking the concurrency of stream

Lab3 Assignment

Problem Description

- Edge Detection: Identifying points in a digital image at which the image brightness changes sharply



Sobel Operator

- Used in image processing and computer vision, particularly within edge detection algorithms.
- Uses two 3x3 kernels gx, gy which are convolved with the original image to calculate approximations of the derivatives - one for horizontal changes, and one for vertical.

5x5 Variation

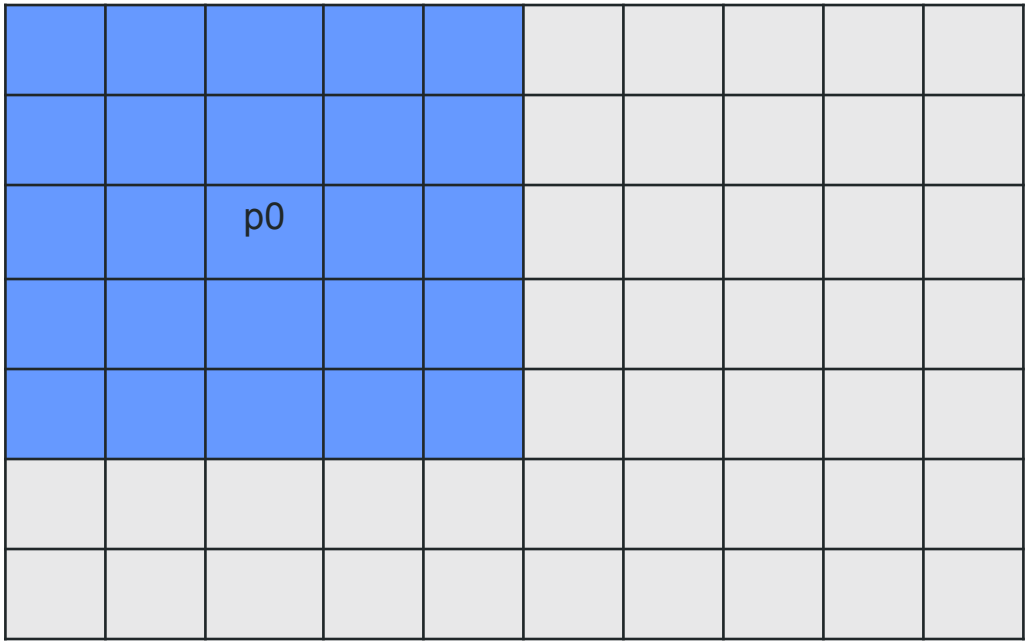
- We use this kernel instead of the 3x3 one in this lab

$$g_x = \begin{pmatrix} -1 & -2 & 0 & 2 & 1 \\ -4 & -8 & 0 & 8 & 4 \\ -6 & -12 & 0 & 6 & 12 \\ -4 & -8 & 0 & 8 & 4 \\ -1 & -2 & 0 & 2 & 1 \end{pmatrix},$$

$$g_y = \begin{pmatrix} -1 & -4 & -6 & -4 & -1 \\ -2 & -8 & -12 & -8 & -2 \\ 0 & 0 & 0 & 0 & 0 \\ 2 & 8 & 12 & 8 & 2 \\ 1 & 4 & 6 & 4 & 1 \end{pmatrix}$$

Convolution Calculation

- Iterate through the width and height of the image
- For each pixel, multiply the filter matrix with original image element-wisely and sum them up.



Hints

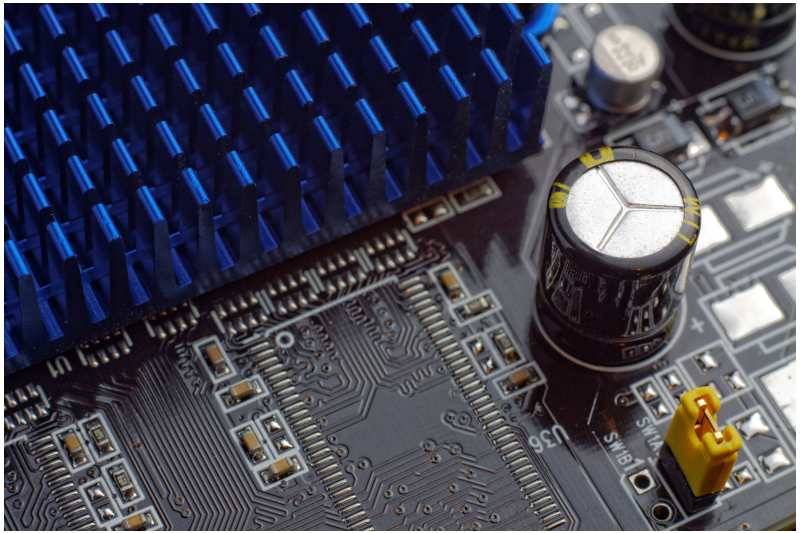
- Refer to sample code

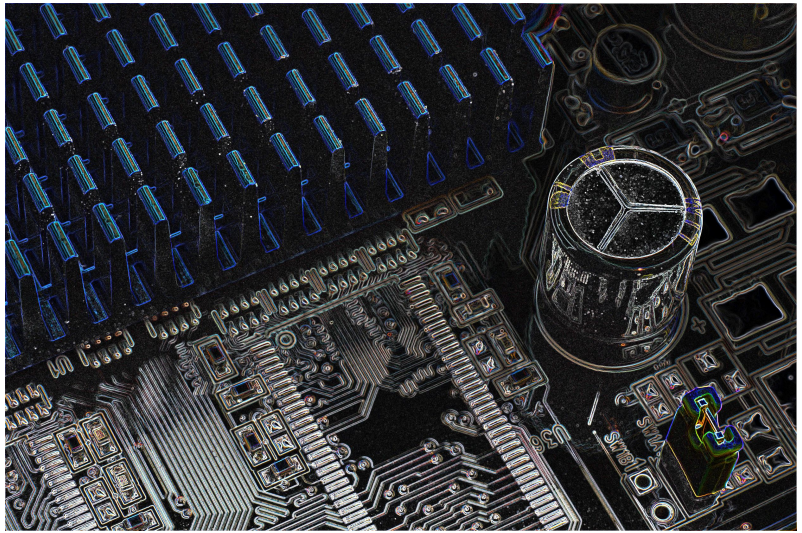
```
/* Hint 6 */
// parallel job by blockIdx, blockDim, threadIdx
for (y = 0; y < height; ++y) {
    for (x = 0; x < width; ++x) {
        for (i = 0; i < MASK_N; ++i) {
            adjustX = (MASK_X % 2) ? 1 : 0;
            adjustY = (MASK_Y % 2) ? 1 : 0;
            xBound = MASK_X /2;
            yBound = MASK_Y /2;

            val[i*3+2] = 0.0;
            val[i*3+1] = 0.0;
            val[i*3] = 0.0;

            for (v = -yBound; v < yBound + adjustY; ++v) {
                for (u = -xBound; u < xBound + adjustX; ++u) {
                    if ((x + u) ≥ 0 && (x + u) < width && y + v ≥ 0 && y + v < height)
                        R = s[channels * (width * (y+v) + (x+u)) + 2];
                        G = s[channels * (width * (y+v) + (x+u)) + 1];
                        B = s[channels * (width * (y+v) + (x+u)) + 0];
                        val[i*3+2] += R * mask[i][u + xBound][v + yBound];
                        val[i*3+1] += G * mask[i][u + xBound][v + yBound];
                        val[i*3+0] += B * mask[i][u + xBound][v + yBound];
                    }
                }
            }
        }
```

Sample Result





Preparation

- TA provides CPU version, Makefile, and hint
- File is located at `/home/pp22/share/lab3`
- Please do NOT copy the testcases.
- `lab3.cu` is cpu version (you need to rewrite it with cuda!)
- You can follow hints to write

How to run

- hades02 & NCHC
  - `./lab3 <input> <output>`
  - `CUDA_VISIBLE_DEVICES=0 ./lab3 <input> <output>`
- hades[03-07]
  - `srun -n 1 ./lab3 <input> <output>`
  - `srun -n 1 --gres=gpu:1 ./lab3 <input> <output>`
- Compare your result with the answer
  - `png-diff <output image> <answer image>`

Hints

- Malloc memory on GPU
- Copy original image to GPU
- Put filter matrix on device memory (or declare it on device)
- Copy filter matrix to shared memory
  (don’t let only one thread do it)
- Parallelize the sobel computing
- Copy the results from device to host
- Free unused address

Submission

- judge will execute your code with single process, single GPU
- submit your code and Makefile (optional) to eeclass before 11/17 23:59
- use lab3-judge
- We support lab3-judge, hw3-2-judge and hw3-3-judge on NCHC, but the final result is based on Hades.
- Try to write your code on NCHC, and perform fine tune on Hades to avoid heavy queueing delay.