Histogram Equalization

Objective

1. Understand how Histogram Equalization is applied to images.
2. Write an optimized GPU code in CUDA that provides the same functionality of the histogram equalization from OpenCV but can perform the algorithm faster.

The project can be developed in groups of a maximum of 3 students.

Part 1: Setting up and running the baseline code

The code for the project can be found in the folder on the discovery cluster:

```
/scratch/gutierrez.jul/GPUClass/FINPROJ/heq
```

Copy these files to your personal folder. In the new folder you will find a Makefile which will be used to compile the code by running:

```
make all
```

To clean files that are generated, we have to use the command:

```
make clean
```

Go over all the code to understand how every part works. Please observe what the kernel of the GPU is doing in this baseline code. It's your job to make it work correctly.

To be able to execute you need to load the module to support OpenCV by running this command or adding it to your .bashrc file:

```
module load opencv
```

Execute the command using the following notation:

```
./heq <input image>
```

For example:

```
./heq input/bridge.png
```

This command will output the following to the command line:

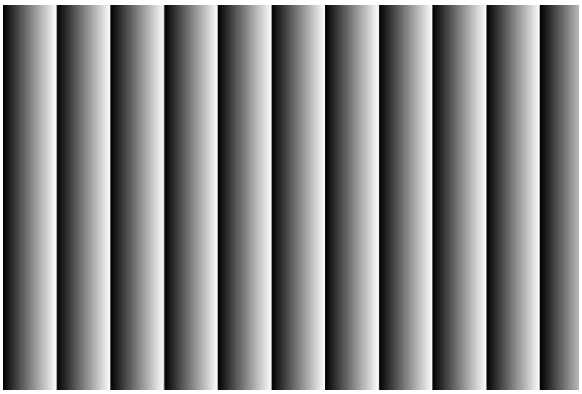
```
Kernel Execution Time: 0.178080 ms
CPU execution time: 17.4714 ms
GPU execution time: 10.5595 ms
Percentage difference: 81.1443%
```

This command will also output the following images:

Input_baw.jpg (Input Black and White version)

Output_cpu.jpg (Output from the OpenCV function)

Output_gpu.jpg (Output from the GPU function)



Given the application will output multiple files, the easiest way to visualize them is by using FileZilla to copy them to your computer (particularly when looking at images). The configuration for this application should be as follows:

1. Host: discovery.neu.edu

2. Username: <Username>
3. Password: <Password>
4. Port: 22
5. Click Quickconnect
6. Now you should be able to copy files from your computer (Local site to the left) to the server (Remote site to the right).

Part 2: Writing your GPU Program

To better understand how to implement a histogram equalization, please review the following link:

http://www.programming-techniques.com/2013/01/histogram-equalization-using-c-image.html

The following are a couple of suggestions in the development of the code:

1. When you are developing your code, the first thing you should do is write a code that is doing the correct calculation, and once it's working correctly, you try to optimize it.
2. Divide the algorithm into stages, and possibly run different kernels for each stage.
3. Remember that we should always consider the time it takes to copy the data to the GPU. If it's taking really long, how can you improve it? Running NVPROF could provide a good insight for this.
4. Test code with different sized images. What's the impact in the performance? What about an all-black image?
5. The goal is to have a fast implementation of the algorithm that has over 98% accuracy when compared to the OpenCV implementation.
6. Start working on it as soon as possible.
7. Last class you will present those results.