CSE 560 - GPU Computing
Winter 2017

Edge Detection of Images on GPU

Due date: 23:59:59, XX January 2017

Max. Marks: 10

- Submit a zipped folder of your solutions for the assignment and the naming convention should be **YourRollNo_asg_1.zip**.
- A report is mandatory along with code submission to receive any credit. Report should include - a brief description of your GPU code design, CPU and GPU result images, and a plot for the timing data.
- Disclaimer : Your code should be written by you and be easy to read. You are NOT permitted to use any code that is not written by you. (Any code provided by the TA/instructor can be used with proper credits within your program).

Prewitt operator is a rudimentary but yet an effective image processing tool for edge detection in images. Prewitt operator consists of two $3X3$ matrices K_x and K_y, for separate gradient computation along two directions (X and Y axis of 2D image plane) of image.

$$K_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad \& \quad K_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \tag{1}$$

For edge detection, input image is represented as a 2D matrix I, where each entry of I corresponds to pixel value of the image. Next we perform convolution between Prewitt operators and image. Convolution is a weighted sum of values of each pixel and its neighboring pixels, that is, a weighted sum of 9 values. The weights are determined by a matrix called *kernel* (a good resource to get started with kernels and convolution on images).

In our case kernels are Prewitt operators. So for every pixel in image we get 2 values from K_x and K_y, g_x and g_y (say). Then we compute the values for each pixel e_i of output image E as

$$e_i = \lceil \sqrt{g_x^2 + g_y^2} \rceil \tag{2}$$

To scale the value of e_i between the range $0-255$ (valid greyscale range) we use the following formulation and this gives us transformed pixel value, e_i' as

$$e_i' = \frac{e_i - e_{min}}{e_{min} - e_{max}} \times 256 \tag{3}$$

where e_{max} and e_{min} are maximum and minimum pixel value (among e_is) respectively.

Understand the given CPU code to generate the output image E for given images in "*images/*" folder.

1. Generate the image depicted the detected edges on the GPU. Divide the workload in a 2D thread configuration. (only the computation part needs to be done on the device, image loading and saving may be done on the host) [6marks]

2. Generate timing data for your code (GPU time vs. CPU time) by varying the image size (image of different sizes are available in the "*images/*" folder). Use image size N = 256, 1024, 2048, and 4096. In the GPU timing, report time taken for kernel execution, as well as total time taken (kernel + memory transfer). [4marks]

Bonus Tasks

1. Use texture memory as an alternative to global memory to read image. [3 marks]

2. Document your observations and compare performance between global memory version and texture memory version of the kernel. [1 mark]

Notes for the given CPU code :

1. Untar and navigate to Assignment_1 folder :

```
$ tar -zxvf Assignment_1.tar.gz
$ cd Assignment_1/
```

2. Run make. This will generate the executable edgeDetect

3. Set the library path :

```
$ export LD_LIBRARY_PATH=./devil/lib:$LD_LIBRARY_PATH
```

4. Run the code, ./edgeDetect, with -s argument