Programming Assignment VI:OpenCL Programming

The purpose of this assignment is to familiarize yourself with OpenCL programming.

1 Problem Statement

A histogram is a statistic that shows frequency of a certain occurrence within a data set. The histogram of an image provides a frequency distribution of pixel values in the image. If the image is a color image, the pixel value can be the luminosity value of each pixel or the individual red (R), green (G), and blue (B). More about image histogram can be found at http://en.wikipedia.org/wiki/Image_histogram.

In this problem, you need to use OpenCL to parallelize the implementation of the image histogram, which needs to calculate the pixel values (RGB) in an image. The input is a one-dimension array consisting the R/G/B values, each of which ranges from 0 to 255, of each pixel of the image.

Below shows a serial implementation of the image histogram. The implementation can be downloaded at http://people.cs.nctu.edu.tw/~ypyou/courses/PP-f17/assignments/HW5/image-histogram.cpp.

```
#include "stdio.h"
#include "stdlib.h"
#include "string.h"
#include <fstream>
#include <iostream>

unsigned int * histogram(unsigned int *image_data, unsigned int _size)
   {

        unsigned int *img = image_data;
        unsigned int *ref_histogram_results;
        unsigned int *ptr;

        ref_histogram_results = (unsigned int *)malloc(256 * 3 * sizeof
           (unsigned int));
        ptr = ref_histogram_results;
        memset (ref_histogram_results, 0x0, 256 * 3 * sizeof(unsigned
           int));

        // histogram of R
        for (unsigned int i = 0; i < _size; i += 3)
        {
                unsigned int index = img[i];
                ptr[index]++;
        }

        // histogram of G
        ptr += 256;
        for (unsigned int i = 1; i < _size; i += 3)
        {
                unsigned int index = img[i];
                ptr[index]++;
        }

        // histogram of B
        ptr += 256;
        for (unsigned int i = 2; i < _size; i += 3)
        {
                unsigned int index = img[i];
                ptr[index]++;
        }
```

```
        return ref_histogram_results;
}

int main(int argc, char const *argv[])
{

        unsigned int * histogram_results;
        unsigned int i=0, a, input_size;
        std::fstream inFile("input", std::ios_base::in);
        std::ofstream outFile("xxxxxx.out", std::ios_base::out);

        inFile >> input_size;
        unsigned int *image = new unsigned int[input_size];
        while( inFile >> a ) {
                image[i++] = a;
        }

        histogram_results = histogram(image, input_size);
        for(unsigned int i = 0; i < 256 * 3; ++i) {
                if (i % 256 == 0 && i != 0)
                        outFile << std::endl;
                outFile << histogram_results[i]<< ' ';
        }

        inFile.close();
        outFile.close();

        return 0;
}
```

2 Input/Output

2.1 Input

The first line represents the size of the one-dimension array, say N, followed by N/3 lines, each of which indicates a list of R, G, and B values (separated by a space) of a pixel. The following example indicates that the input image has three pixels, values of which are (255 0 0), (0 100 255), and (255 255 255), respectively.

```
9
255 0 0
0 100 255
255 255 255
```

2.2 Output

You will need to output a file, named `StudentID.out`, which contains the values pointed by `ref_histogram_results` in the sample code. The output file contain three lines for the R/G/B values, respectively, and each line contains 256 values separated by a space.

3 Requirements

- Your submitted solution contains two source files, which are named `histogram.cpp and histogram.cl`.
- The `histogram.cl` must be the OpenCL kernel that you wrote.

4 Development Environment

4.1 Building the OpenCL environment on your own computer

If you have a nVidia or AMD GPU, you can build your own development environment by installing the corresponding SDK.
https://developer.nvidia.com/cuda-downloads
http://developer.amd.com/tools-and-sdks/heterogeneous-computing/amd-accelerated-parallel-processing-app-sdk/downloads/

4.2 Using CUDA/OpenCL Server in SSLAB

We have set up 4 servers for this assignment. You can login and use one of the servers to work on your assignment. The GPU(s) in the server is an nVidia 1060. We will grade your implementation on these servers.

4.2.1 Login

Server IP: 140.113.215.195 Port: 32403,32404,32414,32428
ID: u+Number Part of Student ID
Password: u+Number Part of Student ID
(You can use "passwd" to change your password)

4.2.2 Compilation

```
g++ -lOpenCL histogram.cpp -o histogram
```

If the message below shows, do not bother with it.The OpenCL environment works as expected.
"/opt/cuda/lib64/libOpenCL.so.1: no version information available"

5 Submission

Be sure to upload your zipped source codes, which includes no folder, to e-Campus system by the due date and name your file as "HW5_xxxxxxx.zip", where xxxxxxx is your student ID.

Due Date: 23:59, January 2, 2018

6 References

- http://www.kimicat.com/opencl-1/opencl-jiao-xue-yi
- http://www.khronos.org/opencl/