CMSC 427 Computer Graphics
Programming Assignment 1: Image Processing
Version: Sep. 22, 2014, 10:30am
Due Date: Sunday, Oct. 12, 2014 @ 9PM
Project Submission:

> **Please email a zip file to the TA at jgbrad1(at)cs.umd.edu before the due date. Read below for further instructions on what must be included.**

Project Description
In about a 5-10 year time frame, I expect you will see single-shot high dynamic range (HDR) sensors become a standard feature on mobile phone cameras. This means your camera will not take several photos at several exposures, but one single image where intensity values for red, green, and blue can have an arbitrary range and are represented by floating point values.

In this assignment, you will implement some basic image processing operations that are common in any image editing software (e.g. Photoshop, GIMP, etc). They are

- Image crop
- Image mirror (over the x-axis and the y-axis)
- Gaussian blur along the X axis
- Gaussian blur along the Y axis
- Gaussian blur in both directions (X and Y axis)
- Scale an image up (the effect seen when zooming in)
- Scale an image down (the effect seen when zooming out)
- Nearest neighbor interpolation
- Bilinear interpolation

but, you'll perform these operations, not on standard RGB images, but HDR images.

Since you're dealing with HDR images, you'll also need to implement a tone mapping algorithm, which allows you to convert an HDR image to a low dynamic range (LDR) image.

- You'll be implementing the tone mapping algorithm from Reinhard SIGGRAPH 2002 http://www.cs.utah.edu/~reinhard/cdrom/. We will discuss this algorithm in class, but you are encouraged to read the original paper.

The images are loaded into memory from a Radiance RGBE HDR image by using an implementation from Greg Ward (see rgbe.txt for details). The image data is stored row by row such that you can access the red, green, and blue channels of a single pixel at position (x,y) by accessing the data at image_data[3*(y*image_width + x)], image_data[3*(y*image_width + x)+1], and image_data[3*(y*image_width + x)+2] respectively.

Code Requirements
Your program must be executable from the command line. To execute your code from the command line, please read the comments inside the *main()* function or take a look at the included README file, which contains examples of how your program may be executed from the command line.

Project Submission:
A zip file containing the following should be sent in an email to the TA. The .zip file should be named as follows: YourFirstName_YourLastName.zip

cmsc427.cpp (your implementation will be here, you may not add any more files)
cmsc427.hpp (you may add function declarations here as needed)
cmsc427.pro
rgbe.c
rgbe.h

The project can be compiled and run using Qt Creator or from the command line using qmake followed by make (or nmake on windows).

Project Grading
The TA will use an undisclosed image and set of command line options to test your implementation so we encourage you to find images online in Radiance HDR format to test your approach. We have provided a file cmsc427.hdr in the starter code, which provides an HDR image for initial testing.

Your program will be graded based upon successful calls to it from the command line and a correct implementation. Your code must be well commented. Note that it will also be judged subjectively based on the simplicity and clarity of implementation. An implementation that is easy to understand, but has few minor bugs will be scored more highly than a messy implementation with the same number of minor bugs.

Points for this project will be distributed as follows:

- Image crop (5%)
- Image mirror over the x-axis (5%)
- Image mirror over the y-axis (5%)
- Gaussian blur along X axis (10%)
- Gaussian blur along Y axis (10%)
- Gaussian blur in both directions (5%)
- Scale an image up (10%)
- Scale an image down (10%)
- Nearest neighbor interpolation (10%)
- Bilinear interpolation (10%)
- Tone mapping (20%)