Programming Assignment 2
Image Operations & Spatial Filtering

Objective:
In this assignment, you will add functionality to the GUI from Assignment 1 to perform the following image operations: addition, subtraction, product, negative, log transform, power (Gamma) transform, Histogram equalization, smoothing filters and sharpening filters.

Implementation details
Your application should support the following functionality:

- Storing the output of the operations to the disk (PPM format).
- Applying more than one operator to the input image. In other words, once an operation has been applied to an image, the next operation should be applied to the resulting output image and not to the original image. This process implements the so-called "cascade of image operations".
- For operations that involve multiple images, the corresponding input images may be of different dimensions. There are multiple ways to solve this problem: first resize both images to a common size and then apply the operation. This will invariably lead to accuracy issues. How can you define a "common" size for the two images? It could be - the window size, or the larger image size.
- Provide the facility to view a histogram of an image using appropriate windowing controls / canvas. In case of Histogram equalization, the user should be able to view the histograms of the source and result images, and compare the results of the operation.

Grade Distribution

Part A: Image Operations (10 points total)

The following operations should be implemented:

1. Addition, Subtraction, Product, Division (4 points)

The user should have the capability of choosing two image files required for the operation. The result of the operation should be displayed and stored in a separate file (also specified by the user). Values beyond the valid range [0,L-1] should be appropriately processed (renormalize or clamp). **Input images must be resized to the "common" size before applying the operation.**

2. Image Negative (2 point)

Perform image negative operation on a PPM file. DO NOT make assumptions regarding the maximum value (L) stored in the file. This value should be read/inferred from the file data.

3. Log Transform, Power (Gamma) transform (4 points each)

The user should have the ability to specify the 'c' and 'gamma' values for these operations. For the above operations, ensure that the values in the image can be displayed faithfully.

Part B: Histogram Equalization (10 points)

Perform histogram equalization operation on an input image. The process should be completely automated without any user driven input.

Part C: Smoothing and Sharpening Filters (10 points)

Implement the Gaussian smoothing filter. The user should input the value of the kernel size N and the standard deviation (sigma). You should then construct an N×N digital filter by sampling the 2D continuous function at appropriate locations and using the discrete mask to perform spatial filtering. Assume suitable behavior (signal repeat, zero padding) at the boundaries. Justify your choice. **(5 points)**

Implement edge detection using the Sobel operators. The output is the addition of the outputs of the individual horizontal and vertical filters. **(5 points)**

A description of the operations is given below (**for your reference only…**):

Operation	Parameters	Description
Add		Perform the addition of the intensities of the two images and write output to the specified file.
sub		Subtract second image from first image and write output to the specified file.
mul		Perform element-wise multiplication of the two images and write output to the specified file.
div		Perform element-wise division of the two images and write output to the specified file.
inv		Perform inversion operation (negative) on the current image and save output to the specified file.
log	-c <constant> -b <base>	Perform log transform to the specified base and save output to the specified file.
Example: log -c 1.25 -b 15 -i beetle.ppm -o output.ppm Perform log transform to base 15 (with multiplicative constant c = 1.25) on the file beetle-13.ppm and write the output to the file output.ppm		
pow	-c <constant> -gamma <exponent value>	Perform gamma transform on the input file and save output to the specified file.
Example: pow -c 2.0 -gamma 0.35 -i beetle.ppm -o output.ppm Perform gamma / exponential transform with c=2.0 and gamma=0.35on the file beetle-13.ppm and write the output to the file output.ppm		
gblur	-N <kernel size> -sigma <floating point value>	Implement a gaussian blur with kernel size NxN and standard deviation sigma. Save output to the specified file.
Example: gblur -N 7 -sigma 10 -i beetle.ppm -o output.ppm Construct a gaussian blur filter of size 7x7 with standard deviation = 10. Next, convolve this filter with the input image beetle-13.ppm and write the output to the file output.ppm		

Operation	Parameters	Description
sobel		Perform the sobel filtering on input image and save output to the specified file.
Example: sobel -i beetle.ppm -o output.ppm Convolve the input file beetle-13.ppm with horizontal and vertical sobel filters and write the output to the file output.ppm		

For all of the above examples, the contents of the file output.ppm should be displayed in the output panel. **Provide feature in your application to see both input & output images in the same window after applying the specified operations to compare them.**

Note that for all appropriate operations, images of different size must be resized to a common size before performing the operation.

Deliverable : For submitting your assignment, create an archive that is named as follows:

CS370_*<student_login_name>_<assignment#>***.zip**

Example: If your login is pacquiao.mandy and assignment 2 is being submitted, your folder would be named CS370_pacquiao.mandy_2.

Include the following in the archive:

a) A folder named **src** – It contains complete source code of the application including any project/solution files that you create. The instructor will not be responsible for creation of Makefiles or VS Projects/Solutions to make your project work. Instructor will compile your code and if your code does not compile to produce an executable, it will not be graded.

b) A folder named **doc** - It contains README.docx having a short description of your solution to the programming project. This should include a description of your data structures for loading and storing images. Include description of extra credit if you have attempted it. Include screenshots of your application in operation. You should submit the document as PDF file. If this document is not in proper format, or the sample output images are not from your implementations, then you will get ZERO grades for this assignment.

c) A folder named **release** – It contains your executable. The instructor should be able to run this executable.

Your project should compile and execute without errors or warnings in the latest Visual Studio environment on Digipen computers.

Failure to submit the deliverable in the required folder format will result in a zero assignment grade.

Late Submission: Refer to the course outline for the late submission guidelines.

Grade Sheet for Assignment-2:

Item	Points Possible	Points Awarded	Comments
Part A			
Addition	1		
Subtraction	1		
Product	1		
Division	1		
Negative	2		
Log Transform	2		
Power (Gamma) Transform	2		
Part B			
Histogram Equalization	10		
Part C			
Gaussian Filter	5		
Sobel Operator	5		
TOTAL GRADE	**30**		

Important Note:

Always rebuild the solution whenever there is modification in the code. The instructor should be able to rebuild/execute your project on any computer at DigiPen having the following software installed:

1. **Visual Studio Community 2022**
2. **wxWidgets-3.0.5**