High Performance Systems Programming (ECE415)
Department of Electrical & Computer Engineering University of Thessaly

Teacher: Christos D. Antonopoulos

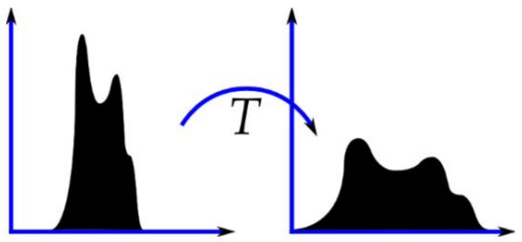
4th Laboratory Exercise

Objective: Parallelize and optimize application on GPU.

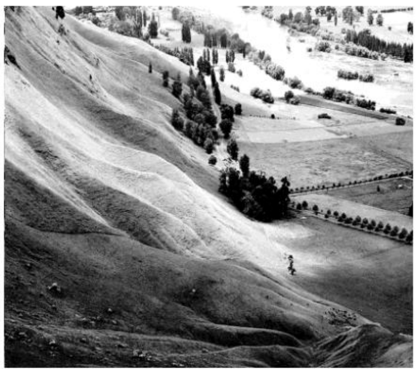
Introduction: Contrast enhancement is a very common process in image processing. As part of this task you will implement contrast enhancement using the histogram equalization process on grayscale ("black and white") images.

As a reference and starting point you are given a sequential implementation of the process in simple C. You are also given images that you can use as input to your code.

Background: A simple method of improving contrast in "black and white" images is histogram equalization. The histogram of such an image corresponds to the distribution of the number of pixels in different shades of gray. Histogram equalization essentially "spreads out" the most common tonality values, which essentially equates to smoothing the histogram. At the image level this results in areas with low local contrast gaining more contrast, but without being forced to change the overall contrast of the image.



Below you can see the results of applying the histogram equation to an image.



Briefly, the application of the histogram equalization transformation is summarized in the following steps:

a) Computation of the histogram of the original image. b) Calculation of the cumulative probability density of each shade of gray. c) Construction of a reference table which will be used to illustrate each

shade of gray in the original image to the corresponding shade in the equalized image.
d) Construction of the final image.

A detailed description of the histogram equalization process, along with a fairly informative example can be found on the wikipedia page: (http://en.wikipedia.org/wiki/Histogram_equalization).

Additionally, it is recommended that you study the code you are given very carefully. The code reads an image in pgm format, calculates its histogram, applies histogram equalization and saves the new image also in pgm format.

You can make your own images with gimp, saving them as pgm (raw format). Then you should open them with a good editor (because their size is big) and delete the comment that gimp adds at the beginning of the file (the line starting with #).

Required:
You will need to port the code you are given to the GPU. The code should be functional for any image size (provided that its device memory is sufficient
GPU). The specific code, although small, is quite receptive to optimizations that we have examined in the course and the use of parallel patterns. You can also see this lab as a competition, where the fastest implementation "wins". Of course, it goes without saying that the implementation should also be correct.

Delivery:
You must deliver:
- The final code. •

Report in which you will analyze the parallelization strategy you followed, the optimizations you applied, and their effect on the kernel runtime for each of the images you are given. Be sure to also mention any optimizations you tried without good runtime results.

Create a .tar.gz file with the above contents and name
<name1>_<AEM1>_<name2>_<AEM2>_lab5.tar.gz. Send it to e-class by 23:59 on Wednesday 14/12/2022.