ECSE 420 – Parallel Computing – Fall 2019

Lab 2 – CUDA Convolution and Matrix Inversion

Due: November 3, 2019 at 11:59 pm

In this lab, you will write code for simple signal processing, parallelize it using CUDA, and write a report summarizing your experimental results. We will use PNG images and TA-provided data as the test signals.

A. Convolution:

Convolution is a slightly more complicated operation than rectification and pooling from Lab 1, but it is still highly parallelizable. For each pixel in the input image, a 3x3 convolution computes the corresponding pixel in the output image using a weighted sum of the input pixel and its neighbors. That is:

$$output[i][j] = \sum_{ii=0}^{2} \sum_{jj=0}^{2} input[i + ii - 1][j + jj - 1]w[ii][jj],$$

$$1 \leq i \leq m - 1, 1 \leq j \leq n - 1$$

where *m* is the number of rows in the input image, and *n* is the number of columns in the input image. Since we are using square weight matrices and the "valid padding" definition of convolution ($1 \leq i \leq m - 1$, $1 \leq i \leq n - 1$), the output image will be of size *m* − 2 by n − 2. Figure 3 illustrates the convolution operation for a certain weight matrix, W.

Write the code that for convolution. For this lab, you need to implement convolution using 3x3, 5x5 and 7x7 weight matrices. Before you convert the output to unsigned chars and save the file, you should clamp the output of the convolution between 0 and 255 (i.e., if a value in the output is less than 0, you should set it equal to 0, and if a value is greater than 255, you should set it equal to 255). The header file "wm.h" contains the weight matrix you should use. Analyze, discuss, and show an example image as described in the first section.

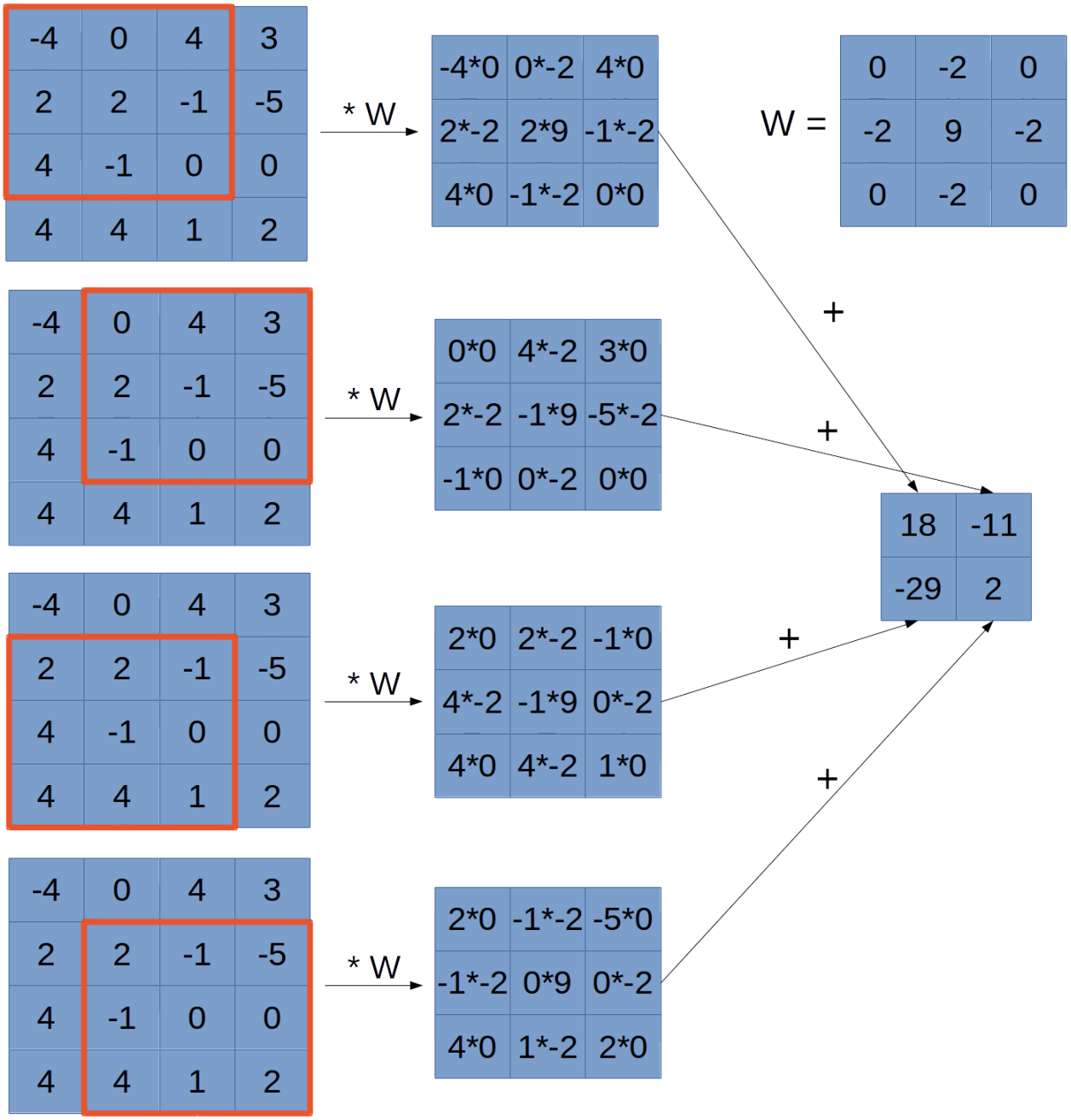

Figure 1: Convolution Example

The grader should be able to change the input arguments to your code: **input file, dimension of the weight matrix (3, 5 or 7) and no. of threads.** You could set it as a hard code inside your code, or just pass them as input arguments to your project.

When the input test image is “test.png”, the output of your code should be identical to “test_convolve.png”. These would be available on myCourses.

B. Matrix Inversion and Linear System Solution

In this part, you will make a CUDA program that inputs matrix *A*, a vector *b* and finds a vector *x* such that $Ax = b$.

You will benchmark your code against several test cases provided on MyCourses and report the performance obtained. At minimum, you should test for the testcases where the matrices are of dimension 32x32, 512x512 and 1024x1024. Furthermore, you should verify the results of your programs by multiplying the matrix *A* and the obtained vector *x* and then subtracting the vector *b* to get a vector Zero.

All the demonstrations of the code should be included in the report. The performance and correctness of the program should be clearly visible in the report.

Submission Instructions:

NO LAB DEMO. There will not be any more demos for the labs. Please submit your entire solution (along with input and output images) and the report in a zip file.

Each group should submit a single zip file with the filename Group <your group number> _Lab2.zip. (Ex – Group04_Lab2.zip).

Format for Report:

1. Must be a PDF only (No word document).
2. Must be named Group<your group number>_Lab2_Report. (Ex – Group03_Lab2_Report.pdf).
3. Must have a cover page.
4. Must follow the logical order for the lab discussions.
5. Have an appendix with **your own** code inside. Copy paste and merge the format so that the code alignment remains the same as in your IDE (MS Visual Studio).