Technical report
Juan Pablo Güitrón Leal
September 2018

Summary

The following document is a report of the second multi-core programming assignment. This second assignment consists on adding a blur effect to an image using CPU, CPU with threads, and GPU. This document shows an analysis and comparison between the execution times of each of the algorithms used.

Introduction:

Parallel programming is the use of computational resources to execute a task simultaneously. Parallel programming solves problems faster than sequential programming. However, there are many problems that can not be parallelized. The objective of this second assignment was to add a blur effect to images, we need to analyze the difference of running time between a sequential and a parallel algorithm. In order to obtain this effect we must use a convolution matrix. In this assignment we're using a 5X5 matrix, which get the averaged color of the neighbors for each pixel.

The program consists of 4 files:

- blur.cpp: Add sequentially the blur to an image.
- blur_wt.cpp: Add blur to an image with the CPU using threads.
- blur.cu: Add blur to an image using the GPU.
- Common.h: Used to display error messages that occurred on the GPU.

Development:

A total of 440 tests were performed in 12 minutes.

- **Characteristics of the system used:**
    - Device: Laptop
    - Model: HP Pavilion 15-cx0001la Notebook
    - Processor (CPU): Intel Core i5-8300H
        - Frequency: 2.3 GHz
        - Cores: 4
    - Video card (GPU): NVIDIA GeForce GTX 1050
        - Memory: 4 GB GDDR5
        - Cores: 640 cuda cores
        - Frequency: 1354 MHz

- **Selection of threads and GPU dimensions:**
  - To select the best dimensions and number of threads in the execution of the algorithm used by CUDA, several tests were performed to evaluate which one had the least execution time.
  - For the selection of the dimensions, 60 tests were carried out with a 3840 * 2400 using 6 different thread configuration, in the following table we can see the results.

	GPU_16_16	GPU_32_32	GPU_16_64	GPU_2_256	GPU_64_16	GPU_256_2
	114.413132	111.428917	113.711349	170.120163	103.069489	106.304413
	109.272079	108.222099	111.510025	166.626663	102.793777	107.627151
	107.999352	109.631104	109.645409	161.049271	103.298424	105.671333
	108.64817	106.955917	110.777435	154.401428	102.495949	106.133163
	114.37101	111.627487	111.319199	153.690674	109.319069	108.907104
	111.41111	108.249527	111.597748	154.171707	105.20517	109.104729
	109.659004	107.983459	111.266708	153.681763	104.088074	106.751244
	108.385933	106.739586	111.72805	154.975006	107.233154	107.311943
	110.190804	106.287857	114.246407	153.403778	107.475754	106.059875
Time	113.82312	114.660194	110.156883	154.513306	110.600037	110.572212
Average	110.8173714	109.1786147	111.5959213	157.6633759	105.5578897	107.4443167

- **CPU threads**
  - In the next table we can see that using the 4 cores of the computer reduces a lot the execution time.

CPUWT 1	CPUWT 2	CPUWT 3	CPUWT 4
4212.09082	2513.615723	1596.422363	1368.228516
4250.931641	2249.595703	1597.337524	1366.091431
4228.140625	2267.815186	1593.146606	1367.830811
4204.175293	2280.650391	1594.187866	1377.897339
4239.374023	2276.848389	1593.958862	1369.441895
7555.201172	2276.091553	1594.17981	1365.015625
4249.460449	2250.784668	1598.684692	1366.893066
4254.249512	2250.077637	1598.506592	1569.972168
4231.893555	2253.547607	1593.441162	1359.746094

4227.726074	2264.109131	1600.406494	1366.188721

Comparison of times (Times in milliseconds):

The following table shows a comparison of execution times between the CPU and GPU (CUDA), changing the image resolution.

	CPU	CPUWT	GPU
640*400	116.1658522	63.9921584	90.4252219
2560*1600	1849.540613	632.4142946	96.1430428
3840 * 2400	4074.062598	1387.730567	105.5578897
7200*4050	13045.44736	4306.70415	161.4078583

- **Performance improvement factor**

  The following table shows the performance improvement factor (speed-up) which is defined as S (n) = T (1) / T (n).

	CPUWT	GPU
640*400	1.815313862	1.284662064
2560*1600	2.924571169	19.23738379
3840 * 2400	2.935773482	38.59552904
7200*4050	3.029102281	80.82287629

Conclusions:

As expected, the results obtained showed that GPU with threads runs faster than the other methods. This analysis also demonstrated the importance of knowing when to use parallel programming, because in small cases it may slow down the execution of the program. We could clearly see this in the first test, in which an image of 600 * 400 was used and the program was executed slower in the GPU than in the CPU with threads, this is because the administration of these threads takes a while, so it is not always worth to use them.

By performing tests with images of different sizes, we can see a considerable change in time between the CPU and GPU, but the difference can be seen more clearly appreciated if you change the size of the blur window because most of the the calculations depend on this matrix.

It is important that the blur window is not of even size, since if it is of even size it does not have an exact center in the pixels and the colors of the image can be affected. Another thing that was important to consider in this assignment are the edges, since the pixels of the edges do not have the same pixels as the central ones.