Rui Huang rhhq7

For homework five, the task is to parallelize image processing with CUDA and compare the result to the result got from CPU. I implemented it at AWS(Amazon Web Service). The reason why I use AWS instead of Google Cloud Platform is that some kind of machine on AWS has GPU CUDA configured for users and I do not need to install them by myself. The following part of this report will discuss the details about my implementation, the time trend and also what I learned from it.

1. Implementation

1.1 GPU_Version

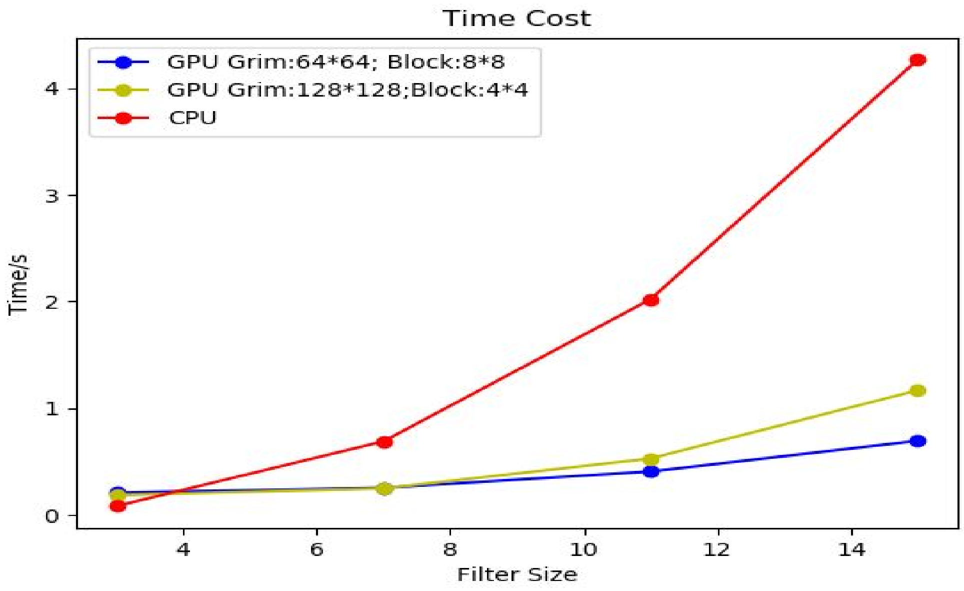
Firstly, I load lena.pgm to host memory called image_source and then malloc some memory called image_dst from host to store the result. Then I pass the address of image_source and also the address of image_dst to a function called “conv”. This function malloc two memory from device : dev_src and dev_dst. Then copy image_source and image_dst from host to device. Now, since there were copies of image_source and image_dst in device, I can use GPU to process them. For processing images on GPU, I use two different grid and block structure. For the first one, the dimension of grid is 64*64 and the dimension for block is 8*8. For the second one, the dimension of grid is 128*128 and the dimension for block is 4*4. I map each thread to a pixel in dev_src and dev_dst with the help of some parameter that CUDA provided:threadIdx, blockIdx, blockDim, and gridDim. After I map each thread to a pixel, called “center”, I stored all pixels within a window around center to a array. The size of the window is a parameter that received form the command line and it represents the size of the filter. Then I use quicksort to sort the array. After that, I can easily get the median of the array. I use the median of the array to replace the pixel “center”. When all threads finished their work, the whole process is done. Then I copy the result from device to host.

1.2 CPU_Version

Firstly, I load lena.pgm to memory. For each pixel called “center” (except pixels that near the border), I store all the pixels around center within a window to a array and use quicksort to sort them. Then I use the median of the array to replace the center. The process is almost the same as the GPU_Version. However there are two difference :first, CPU_Version doesn’t have the process of copying data between host and device. second, CPU_Version has only one thread, while GPU_Version has 512*512 threads to work.

1.3 Time trend

Below is the image that show the time trend



2. Result

Image 1-4 shows the results from GPU(Grim:64*64. Block:8*8) and 5-8 shows the results form CPU

Image 1 Filter Size : 3

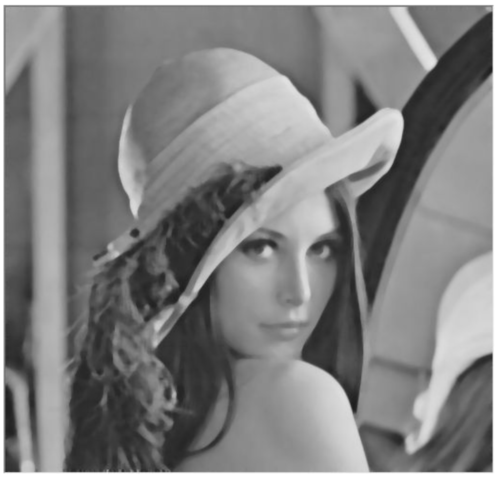


Image 2 Filter Size : 7

Image 3 Filter Size : 11

Image 4 Filter Size : 15

Image 5 Filter Size : 3

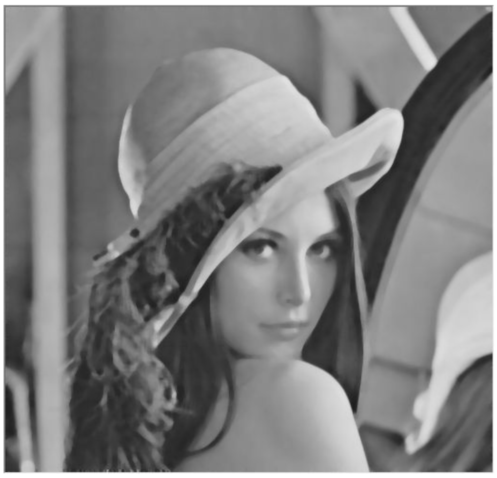


Image 6 Filter Size : 7

Image 7 Filter Size : 11

Image 8 Filter Size : 15

I load there images to memory and compare them and found that the results from GPU and CPU are absolute the same, the accuracy is 100%.

3. What I Learned

Before this homework, I had no idea about cuda and GPU programing. After finishing this homework, I knew how to use cuda and how to write using GPU. This make me a little excited. Also, by solving bugs, I know the structure of grid and block must be used properly because when I set the dimension of grid to 512*512 and block to 1*1, it produce a illegal memory access error when the size of filter goes up to 11.