Available online www.jocpr.com

Journal of Chemical and Pharmaceutical Research, 2014, 6(7):1308-1314

Research Article

ISSN : 0975-7384
CODEN(USA) : JCPRC5

Roberts edge detection algorithm based on GPU

Hong Xiang Gong and Liu Hao

Nanchang University, Nanchang, Jiangxi, China

ABSTRACT

With the development of the semiconductor technology, the GPU's floating point computing capacity improves rapidly. How to apply the GPU technology to the non-graphic computing field becomes a highlight in the research of high performance computing. The Roberts edge detection algorithm is a typical image processing algorithms. A fast Roberts edge detection algorithm is presented based on GPU, Texture memory technology and asynchronous data transmission are adopted to optimize the application of the algorithm. The experiment expressed that we could get a over ten times speed effect by this method than traditional Roberts edge detection algorithm.

Keywords: GPU; Roberts arithmetic; General compute

INTRUDUCTION

With the development of the semiconductor, more and more transistors are integrated in the unit area, for this case the leakage current per unit area, power consumption and heat generation are becoming more and more serious, increasing transistor clock frequency to improve the performance of CPU has been pushed to the limit. Then CPU towards multi-core direction, namely adding multiple cores in a single processor enhances the capabilities of CPU, Intel and AMD have launched a multi-core processor. In 2006, Intel introduced the core architecture based on dual processor. At the same time, the graphics processor has a high-speed development,Single precision floating-point arithmetic ability of mainstream GPU is 10 times more than the same period of CPU. Take the new kepler architecture of GPU as an example, the single precision floating point operation capacity of 3.25T flops, memory bandwidth of up to 288GB/s. At the same period the single precision floatingpoint arithmetic ability of Haswell Core i7 CPU based on Ivy Bridge architecture is…..In the software programming environment, some programming modes about multi processor and multi node cluster are proposed. For example, the multi thread programming language MPI on multi-core CPU developmentand OpenMP for computer node development. But the two languages have their own disadvantages, MPI multithreading switching cost hundreds of cycles and the computer cluster with high cost and communication overhead is also time consuming node. Microsoft Corp developed their own parallel programming language C++AMP. In order to use the GPU powerful floating-point computing power of general computation, developers must develop with the help of API graphics and it is very inconvenient. After the rise of GPU computing, the major GPU vendors have introduced the programming environment for their own GPU, including NVIDIA launched CUDA and AMD/ATI launched the Brook+, but the lack of effective means of data communication between threads. In 2008, Apple Corp proposed a cross platform heterogeneous programming OpenCL, run by a non-profit organization Khronos group, but the drawback is still a large number of API, programming is more complex, and there are some difficulties in promotion. In 2007,NVIDIA launchedcomputing unified device architecture programming Platform CUDA.

Edge detection of image can be obtained information rich, widely used in target tracking, image compression and the field of machine vision,Robertsoperator is a gradient based edge detection method, the detection effect is good, the computational complexity is moderate and is used in real-time image processing. However, Roberts operator edge detection algorithm consists of two-dimensional correlation operation, and the resolution is increased, calculation becomes large, how to enhance the efficiency of the algorithm on a specific hardware platform is the focus of current research.Literature[1,2] respectively introduces the efficient method to implement the algorithm in FPGA and DSP, but these methods are commonly used for processor(CPU,DSP and FPGA),the traditional serial algorithm can't run efficiently on GPU.

This paper adopts NVIDIA Geforce 9600GSO512 card for Roberts edge detection of gray level image with Gauss noise, the performance of Roberts edge detection based on GPU is 10 times higher than the performance achieved using the CPU and the optimized Roberts edge detection algorithm has obtained 30 times than CPU performance.

1. GPU GENERAL CALCULATIONS

The application of GPU development system comprises the unified device architecture of CUDA which is a heterogeneous system proposed by NVIDIA company and a Heterogeneous system named OpenCL proposed by Apple KhronosGroup .

2.1 The development of GPU

General-purpose computing of GPU is developing with the rendering pipeline. GPU rendering pipeline main task is to complete the 3D model to the image rendering. Commonly used graphics API programming model rendering process can be divided into several parallel processing stage and it's Separately by the different unit in the GPU rendering pipeline. At different stages of GPU rendering pipeline, the object which need to deal with are vertex, primitive, fragments, pixel. Graphics rendering process with inherent parallelism with the reason that the data correlation in the vertices, fragment and primitive is very weak so that we can use the method of computation of parallel to calculate[2]. This makes use the method to mention throughput through parallel processing becomes possible.

With the growing demand for graphics applications, fixed-line programming mode increasingly becamedifficult to adapt the demand so that the programmable rendering pipeline gradually developed. Line of programmable part concentrated in the vertex and pixel processing unit. The operation procedure is called vertex shader and pixel shader. But the vertex processing unit and a pixel processing unit hardware separation will cause load imbalance, so Starting from DX10, the industry put forward the unified shader architecture which put the vertex processors and pixel processors combined together into a unified stream processors.

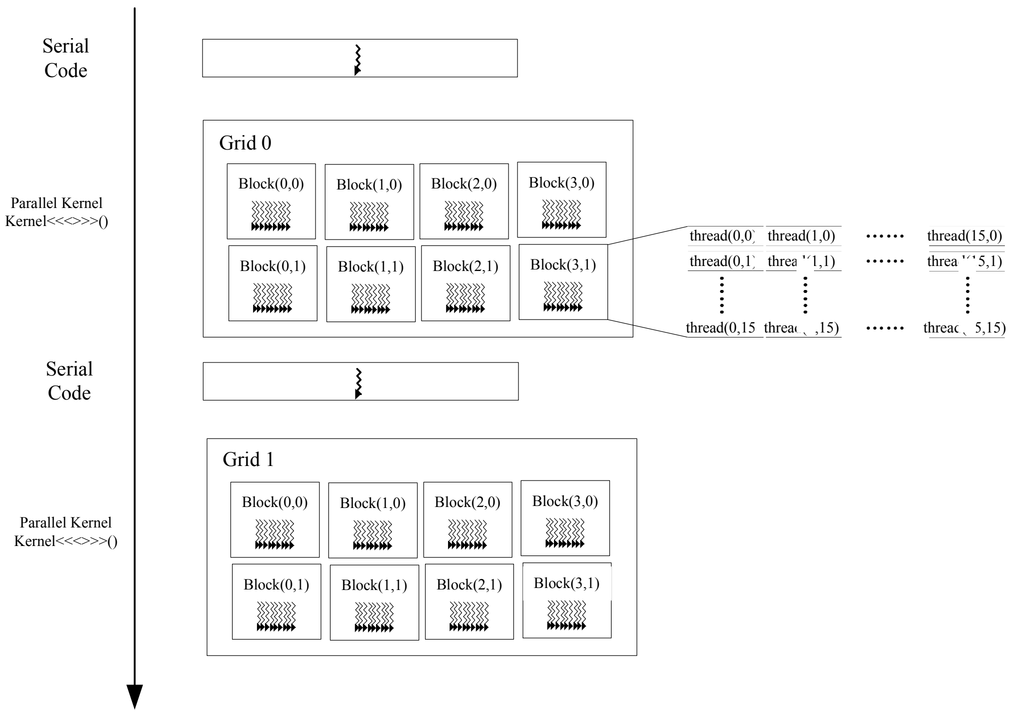


Fig.1: CUDA programming model

2.2 CUDA programming model

CUDA programming model make CPU and GPU work together, CPU is responsible for the serial section, GPU is responsible for the part of intensive and parallelizable. Usually GPU algorithm and CPU algorithm is different from kernel function. The kernel function belongs to the parallel portions of the program, by using the function type keyword _ _global_ _ statement[7].Andit indicates that the function is executed in the terminal equipment.

From the figure 1 we can seen that the entire procedure of CPU serial code is responsible for the data preparation to execution of kernel function. It is mainly include memory allocation, distribution of video memory and grid configuration. Then start the kernel function to parallel processing data in the video memory by GPU. Among them, parallel kernel function is divided into two levels of parallelism which includes parallel block in grid and parallel thread in the block. Then use the CPU serial code which is mainly responsible for the cleanup of a kernel function, and start the next kernel function. There is no overhead to start the second core function. After the execution of a kernel function, the processing of data is written back to memory. Then the CPU will display the results read back into memory and use the CPU to process other data.Finally, release the memory and memory space, exit CUDA.

2.3 CUDA execution model
NVIDIA has developed a set of NVCC compiler driven based on CUDA platform, the compiler can be embedded into the Visual Studio development platform. We can complete the compiler of GPU code when we compiling a CUDA program.

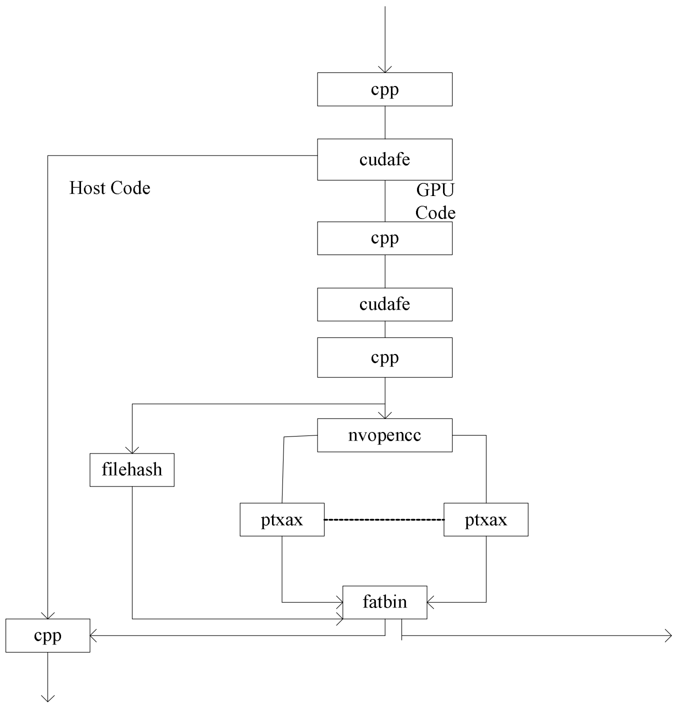


Fig.2: NVCC compiler

As can be seen from Figure 2, the whole project is first separated by cudafe as the host code and the device code, device code is compiled PTX code or binary code, and the host code is compiled by other compiler or output as a code object through calling the host compiler in the final stages[7].

3. ROBERTSOPERATERAND GAUSS FILTER KERNELS

Roberts detection of the classical operator is composed of two two template shown in Figure 3, a corresponding to the vertical edges, a corresponding to the horizontaledge[11].For the input image *f(x,y)*, the output image is obtained by the following formula:

$$g(x,y) = |\nabla f(x,y)| = \left\{\left[f(x,y+1) - f(x+1,y)\right]^2 + \left[f(x+1,y+1) - f(x,y)\right]^2\right\}^{1/2} \quad (1)$$

Templates as shown in Figure 3:

1	0
0	1

0	1
1	0

Fig.3: Roberts operator

The collected image is often accompanied with Gauss noise and usually processed by Gauss filtering, Gauss filtering is collective operation, using a template (or convolution, mask)of every pixel in the scanned image, determine the weighted average gray values of the neighborhood pixel value[3] to replace the template center pixel with template.

The general template size is 3×3 or 5×5, as shown in figure 4. Here we use the template is 3×3 ,the weight distribution in figure.

$\frac{1}{16}\times$

1	2	1
2	4	2
1	2	1

$\frac{1}{273}\times$

1	4	7	4	1
4	16	26	16	4
7	26	41	26	7
4	16	26	16	4
1	4	7	4	1

Fig.4: Gaussian filter kernels

Convolution formula of Gauss filter:

$$g(x,y)^{'}=\left\{f(x-1,y-1)+f(x-1,y+1)+f(x+1,y-1)+f(x+1,y+1)+\left[f(x-1,y)+f(x,y-1)+f(x+1,y)+f(x,y+1)\right]*2+f(x,y)*4\right\}\Big/16 \quad (2)$$

4. THE IMPLEMENT of ROBERTSEDGEDETECTION on GPU

4.1 The description of algorithm

The algorithm involves Gauss filter and Roberts edge detection and it has the dependence between the two algorithms, so we design algorithm for serial. But the part of remove the image noise after Gauss is using Gauss filtering convolution kernel and obtains sliding average of input image. Additionally, each pixel in the output image gray for the values is independent of each other and has no dependence. Therefore, parallel algorithms can be used. As for the Roberts edge detection, pixel output image is obtained by template correlation operation. In the other side, get the output image of each pixel gray values are independent of the process.Therefore, parallel algorithms can also be used. Roberts edge detection algorithm can be expressed as in the Figure 5.

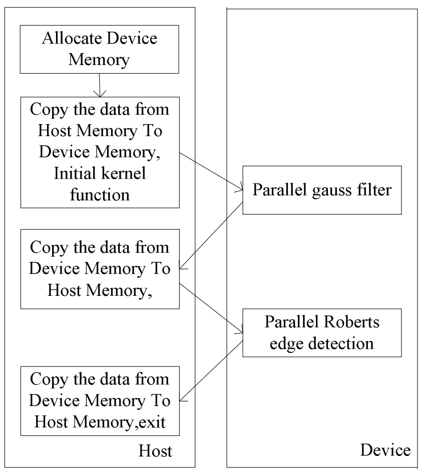

Allocate Device Memory
Copy the data from Host Memory To Device Memory, Initial kernel function
Copy the data from Device Memory To Host Memory,
Copy the data from Device Memory To Host Memory,exit
Host
Parallel gauss filter
Parallel Roberts edge detection
Device

Fig.5: Roberts edge detection algorithm based on CPU+GPU

As can be seen from Figure 5, the algorithm is using CPU and GPU parallel collaborative computing. Firstly, using the host API to allocate memory space and copy the image data from the memory to video memory. Then, configure the grid block and grid dimensions and start the kernel function. Device (GPU) paralleling to implement the Gauss filter, then copy the image data from video memory to memory and configure the next kernel. At the same time, copy the data from the memory to the video memory of next block and Start the kernel2 function. Device also parallel detection implementation of Roberts edge. When the calculation is completed, it will copy the data from the video memory to memory. The pseudo code of the algorithm can be expressed as:

```
Aogrithm1:Roberts edge detection based on GPU(using global memory)
Input: image A with Gaussian noise and Roberts operator, Gaussian filter kernel( 3×3 )
Output: image B that GPU have processed
1:i=threadidx.x+blockdim.x*blockIdx.x； j=threadidx.y+blockdim.y*blockIdx.y；
2:initial kernel1
3:if i<bmpWidth -2and j<bmpHeight -2 do
4: for m=i toi+2 do
5:        for n=j to j+2 do
6:compute the sum of the pixel    (Eq.(2))
7:        end for
8: compute the average pixel
9:   end for
10:end
11:copy the data from device to host
12:initial kernel2
13:copy the data from host to device
14:if i<bmpWidth-1and j<bmpHeight-1 do
15: compute the relevant operation of
16:end
17:copy the data from device to host
```

Resolution gray-scale image provided by *BmpWidth*BmpHeight*, we use each thread processes a pixel. If the block dimensions is(*blockDim,blockDim*), the number of processing the entire image of the desired block is *BmpWidth/blockDim*BmpHeight/blockDim*, and grid dimension is （*BmpWidth/blockDim*,*BmpHeight/blockDim*）. When*BmpWidth* and *BmpHeight* is not an integer multiple of *blockDim*, we need put the number needed to block to （*BmpWidth+blockDim-1/blockDim*）*（*BmpHeight+blockDim-1/blockDim*）to make sure all the pixels are assigned in a independent thread.

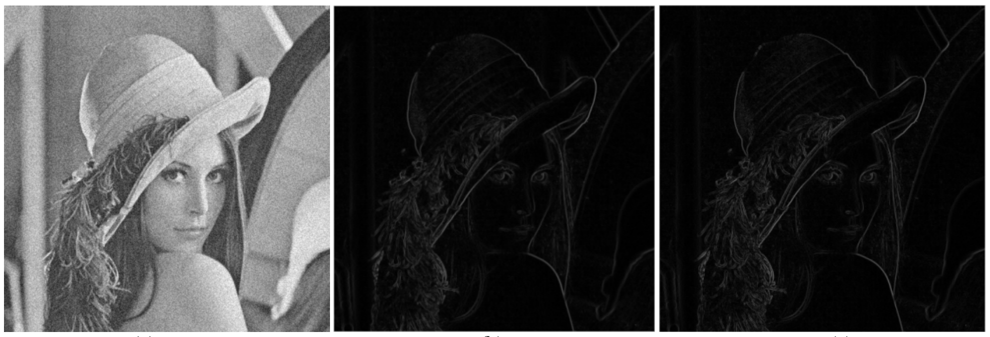
4.2Algorithm optimization

Although the GPU floating-point computing power and memory bandwidth are far more than CPU, but due to the GPU using the PCI-E bus and the host connection so that Therefore, its input and output throughput is limited by I/O bandwidth. So, we should try to reduce the amount of data transmission between CPU and GPU.

RESULTS AND DISCUSSION

The experimental platform used in this thesis is the AMD Sepron(TM) Processor 3200+, frequency is 1.8GHz, 2GB of system memory. Streaming multiprocessor is 12, CUDA core number is 48. Operating system Win7 ultimate, the whole experiment based on cuda5.0+visual studio 2008.

The resolution of gray image of 128*128, 256*256, 512*512, 1024*1024, 2048*2048, 4096*4096, 8192*8192 are chosenrespectively for Roberts edge detection based on CPU+GPU,run time determination of code, The original image and the processed image is shown in Figure 6, the experimental time as shown in table 1.



(a) (b) (c)

Fig.6: Roberts edge detection: (a) original image (b) image that have processed based on CPU (c) image that have processed based on CPU+GPU

Table 1: Roberts edge detection time of images on CPU and CPU+GPU

Image size	GPU Kernel (ms)	GPU Total (ms)	CPU Time (ms)	Rate
128*128	0.272	186.600	4.154	0.022
256*256	0.997	86.529	15.797	0.182
512*512	7.271	142.675	73.8	0.157
1024*1024	56.717	194.142	420	2.163
2048*2048	276.044	372.635	1178	3.162
4096*4096	1046.563	1242	4413	3.553
8192*8192	5260.553	7403	20730	2.800

As can be seen from figure 6, we can obtain the same visual effect when comparing the CPU+GPU heterogeneous implementation of Roberts edge detection with CPU serial implementation of Roberts edge detection. It can be seen from table 1, when processing the resolution for 4096*4096 gray image, speed can reach a maximum of 3.553. We can also conclude from table 1, we cost more time using CPU alone than CPU+GPU at a low resolution image, because the data copied from host memory to device memory through the PCI-E bus spends a lot of time, it can be seen from the NVIDIA performance analysis tool NVIDIA visual profiler.

CONCLUSION

In this paper, we release the Roberts edge detection algorithm based on GPU and introduce the development of hardware, programming model and the execution model of CUDA, and presents a method to map the traditional serial code of Roberts edge detection algorithm to the GPU and adopts CUDA stream to optimize the algorithm. The experimental results show that the method used in this paper can make full use of the characteristics of GPU. At the same time, we are easy to achieve Sobel, prettiew edge detection algorithm [17,18].

REFERENCES

[1] Sanders, Jason, and Edward Kandrot. CUDA by example: an introduction to general-purpose GPU programming. Addison-Wesley Professional, **2010**.
[2] OpenCL Website, http://www.khronos.org/opencl/
[3] GPU VSIPL: High-Performance VSIPL Implementation for GPUs
http://gpu-vsipl.gtri.gatech.edu/
[4] Fan, Zhe, et al. "GPU cluster for high performance computing." Proceedings of the 2004 ACM/IEEE conference on Supercomputing.*IEEE Computer Society*, **2004**.
[5] Farber, Rob. CUDA application design and development.Elsevier, **2011**.
[6] Zhang N, Chen Y, Wang J L. Image parallel processing based on GPU[C]//Advanced Computer Control (ICACC), *2010 2nd International Conference on. IEEE*,3: p.367-370.**2010**,
[7] Sanders, Jason, and Edward Kandrot. CUDA by example: an introduction to general-purpose GPU programming. Addison-Wesley Professional, **2010.**
[8] Roberts, Mike, et al. "A work-efficient GPU algorithm for level set segmentation." *Proceedings of the Conference on High Performance Graphics.Eurographics Association*, **2010.**
[9] Marr, David, and Ellen Hildreth. "Theory of edge detection."Proceedings of the Royal Society of London. Series B. Biological Sciences 207.1167 (**1980**): 187-217.
[10] Peli, Tamar, and David Malah. "A study of edge detection algorithms." *Computer graphics and image processing* 20.1 (**1982**): 1-21.
[11] Gonzales, R. C., & Woods, R. E. (**2002**). Digital Image Processing, 2-nd Edition.
[12] CUDA. http://developer.nvidia.com/object/cuda.html
[13] NVIDIA Corporation．CUDA programming Guide 5.0[EB/OL]．http://www.nvidia.com，**2012-08-20**．
[14] TANG, T., & LIN, Y. S. Design and Implementation of Jacobi and Laplace Algorithms on GPU Platform. *Computer Engineering & Science*, 31, pp.93-96.**2009**.
[15] Sanders J, Kandrot E. CUDA by example: an introduction to general-purpose GPU programming. Addison-Wesley Professional, **2010**.
[16] Kazakova N, Margala M, Durdle N G. Sobel edge detection processor for a real-time volume rendering system.*Proceedings of the 2004 International Symposium on. IEEE*, 2: II-913-16 Vol. 2.
2004.
[17] Vincent O R, Folorunso O. A descriptive algorithm for sobel image edge detection.*Proceedings of Informing Science & IT Education Conference (InSITE)*.pp.97-107.**2009**.

[18] Niu S, Yang J, Wang S, et al. Improvement and parallel implementation of canny edge detection algorithm based on GPU. *2011 IEEE 9th International Conference on.IEEE*,p.641-644.**2011**