ACCELERATED SEAM CARVING USING CUDA

Prathmesh Savale[1], Fardeen Ahmed[2], Kalpesh Dusane[3], Swapnil Dhage[4]

[1]*Student, Department of Computer Engineering, Sinhgad Academy of Engineering, Pune, Maharashtra, India*
[2]*Student, Department of Computer Engineering, Sinhgad Academy of Engineering, Pune, Maharashtra, India*
[3]*Student, Department of Computer Engineering, Sinhgad Academy of Engineering, Pune, Maharashtra, India*
[4]*Student, Department of Computer Engineering, Sinhgad Academy of Engineering, Pune, Maharashtra, India*

Abstract

The purpose of this review paper is to show the difference between executing the seam carving algorithm using sequential approach on a traditional CPU (central processing unit) and using parallel approach on a modern CUDA (compute unified device architecture) enabled GPU (graphics processing unit). Seam Carving is a content-aware image resizing method proposed by Avidan and Shamir of MERL.[1] It functions by identifying seams, or paths of least importance, through an image. These seams can either be removed or inserted in order to change the size of the image. It is determined that the success of this algorithm depends on a lot of factors: the number of objects in the picture, the size of monotonous background and the energy function. The purpose of the algorithm is to reduce image distortion in applications where images cannot be displayed at their original size. CUDA is a parallel architecture for GPUs, developed in the year 2007 by the Nvidia Corporation. Besides their primary function i.e. rendering of graphics, GPUs can also be used for general purpose computing (GPGPU). CUDA enabled GPU helps its user to harness massive parallelism in regular computations. If an algorithm can be made parallel, the use of GPUs significantly improves the performance and reduces the load of the central processing units (CPUs). The implementation of seam carving uses massive matrix calculations which could be performed in parallel to achieve speed ups in the execution of the algorithm as a whole. The entire algorithm itself cannot be run in parallel, and so some part of the algorithm mandatorily needs a CPU for performing sequential computations.

Keywords: *Seam Carving, CUDA, Parallel Processing, GPGPU, CPU, GPU, Parallel Computing.*

---***---

1. INTRODUCTION

The field of multimedia processing is today one of the most important areas in computer science. We cannot imagine music and videos without image processing. The amount of data that our device is supposed to process is enormous for rendering as well as processing. Even the quality of the image depends on the quantity of this data. The average figure today presented with 5-10 millions of pixels, the so-called HD video (High-Definition video, video with higher resolution than standard video), each of the 25 images represented by about 2 million pixels, which means 50 million thereof, in a second. All these new amounts of data require greater processing power of the available devices. The mere display requires a lot of processing power. Before it is decided to upgrade the hardware devices, it should be checked whether all the potential of the current devices is exploited or not. This thesis focuses on the utilization of computational power of graphics processing units. To see a comparison of computational capabilities Seam Carving algorithm is chosen. The algorithm represents one new approach for image processing; in addition, as compared with some of the popular algorithms for image processing it is not entirely trivial. There is also a high degree of parallelism associated with implementation of this algorithm, which is crucial to exploit the computing power of GPUs. The following section presents the algorithm itself, then CUDA environment, which allows us to write programs to run on the GPU, at the end of the grazing implementation of the algorithm itself.

2. SEAM CARVING

Seam Carving is one of the 3 content aware image resizing algorithms which became commercially popular after it was provided as a tool for image retargeting in Adobe Systems image editing software Photoshop. Every day a variety of devices are used that use their own dimensions and display, therefore images need to be adjusted to be shown on different devices. For such adjustments, mostly used technique is to reduce the image to maintain the aspect ratio as long as one of the dimensions does not correspond to the display (Transform). Another option is to reduce the image from both the dimensions till it corresponds to the display (Scaling). Disadvantages of these algorithms and advantage of Seam Carving are shown in Figure 1. Seam Carving algorithm when resizing takes into account the content of the image. It can be seen in Figure 1 that the removed portions were monotonous background and not the main parts of the picture, which are the portions of grass and sky. Thus seam carving is better than scaling and transform. The following explains the operation of the algorithm, which leads to such results.

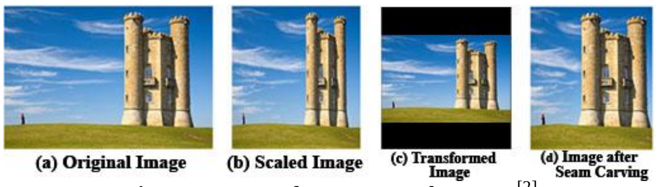

Fig 1: Image Adjustment techniques [2]

3. CUDA

CUDA (Compute Unified Device Architecture) is an open parallel architecture, introduced in 2007, developed by Nvidia. It is a computational drive, which is used in graphics processors, and is accessible to software developers through the standard programming languages. It mainly uses C as an extensing language for CUDA, along with other languages such as Perl, Python, Java, Fortran and Matlab environment. CUDA gives developers access to a limited set of command and GPU resources. CUDA exploits the potential of the powerful graphics processing units in modern computers. The main difference in the architecture of CPU and GPU is in the proportion of transistors that they designed with, as well as the number of tasks that they can perform in parallel. The first difference can be seen in Figure 2. The orange color is highlighted memory part, to control yellow, and green represents arithmetical logical part of the process unit.

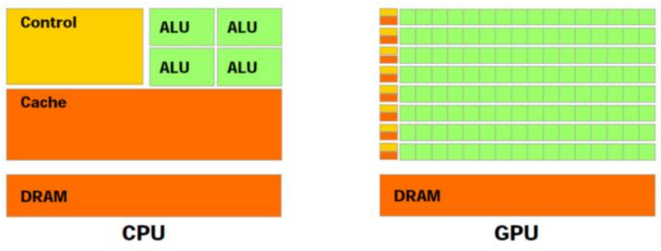

Fig 2: Comparison of CPU and GPU [8]

3.1 Implementation using CUDA

Figure 3 shows an example of the use of graphics processor. The operation is divided into four phases:
1. Transfer of data from main memory to GPU memory
2. CPU commands GPU to perform the task
3. Graphics processing unit processes the data
4. Transfer of data from GPU memory to main memory

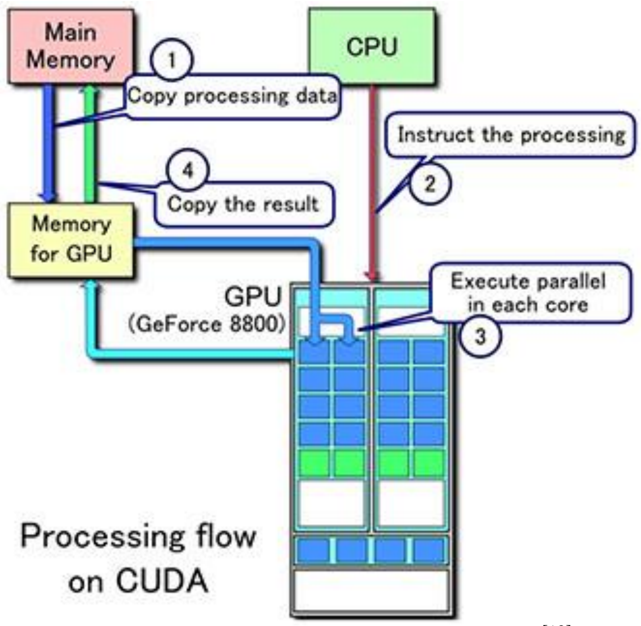

Fig 3: Sequence of events when using GPU[10]

3.2 CUDA Programming Model

Software code written for CUDA system is divided into two units, the parallel and the serial code. The serial code is carried out on the CPU and contains mainly commands to transfer data to the GPU and vice versa. Otherwise it may be a serial code which includes a part of the algorithm, which is implemented on the CPU. In CUDA programming model, CPU is known as the host and GPU is known as the device. A piece of code that runs on the GPU is known as thread. A collection of threads, known as a block runs on a single CUDA processor, where all threads within the block run in parallel. The image to be processed is divided into such blocks. A collection of blocks is known as a grid. A C-like function that runs on the GPU is known as kernel. The kernel call includes the specification of gridsize, blocksize and other specifications that are required. Figure 4 shows the block diagram of CUDA programming model.

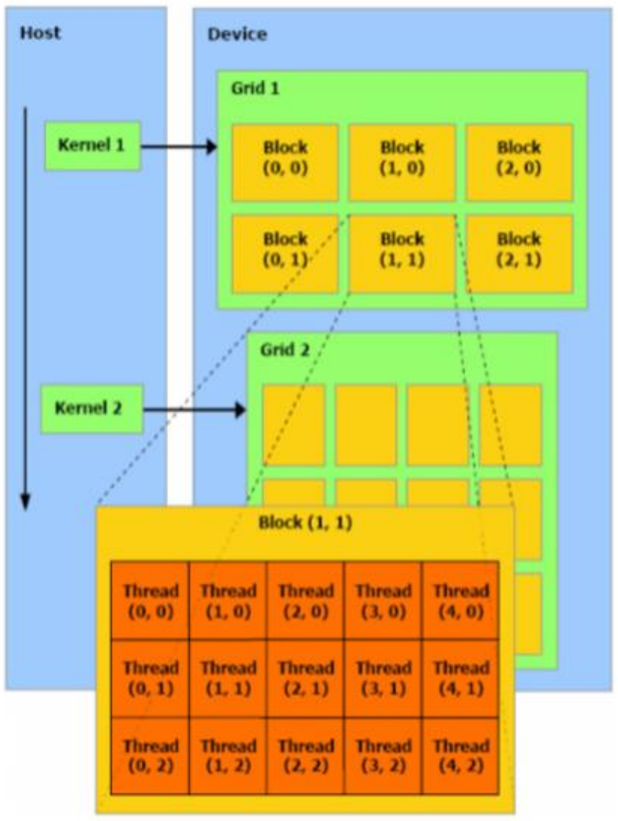

Fig 4: CUDA Programming Model [8]

4. PROPOSED SYSTEM

The following image shows our proposed system for implementing seam carving, it can be very well observed that seam carving like any other image processing algorithms implements humongous matrix calculations, viz. the conversion of a particular image to a 2D array based on the RGB values of the pixels as well as construction of energy maps and gradient tables [2]. If we are to use a parallel approach in implementing these massive calculations on a GPU with a parallel architecture like

CUDA we are sure to achieve computational speedups on a grand scale.

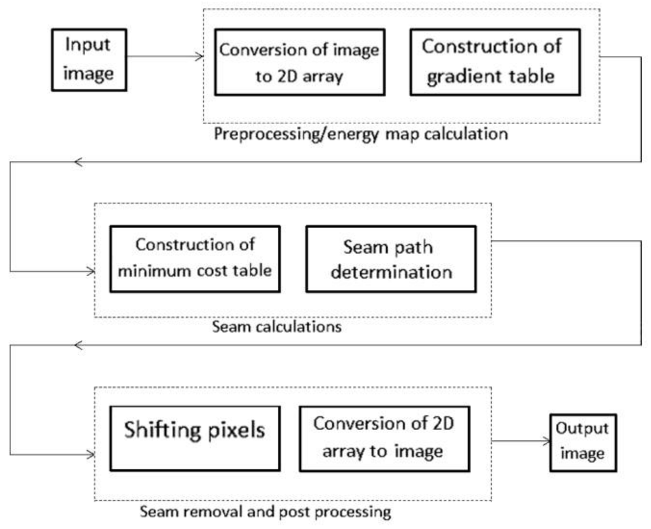
Input image
Conversion of image to 2D array
Construction of gradient table
Preprocessing/energy map calculation
Construction of minimum cost table
Seam path determination
Seam calculations
Shifting pixels
Conversion of 2D array to image
Output image
Seam removal and post processing

Fig 5: Block Diagram for Seam Carving

5. FUTURE SCOPE

1. Content aware image resizing algorithms also include seam insertion and object removal, which can be integrated within our proposed system of seam carving.
2. Real time image as well as video resizing can be implemented on TEGRA K1 (192-core) powered mobile and handheld devices which support CUDA.

6. CONCLUSION

Parallel implementation of seam carving algorithm will be faster than the sequential implementation. Thus using CUDA for parallel programming exploits the potential of GPUs which gives ten times the speedups due to massive parallelism in image processing algorithms.

ACKNOWLEDGEMENTS

We would like to sincerely thank Prof. G. S. Gurjar, our guide from Sinhgad Academy of Engg. and Mr. P. Desai, our mentor from Persistent Systems Pvt. Ltd. for their support and encouragement.

REFERENCES

[1]. Shai Avidan and Ariel Shamir. “Seam Carving for Content-Aware Image Resizing”, The Interdisciplinary Center & MERL.
[2]. [Online] http://en.wikipedia.org/wiki/Seam_carving
[3]. Ronald Duarte and Resit Sendag. “Run-time Image and Video Resizing Using CUDA-enabled GPUs”, Department of Electrical and Computer and Biomedical Engineering, University of Rhode Island, Kingston, RI
[4]. Stas Goferman, Lihi Zelnik-Manor and Ayellet Tal. “Context-Aware Saliency Detection”
[5]. Anindya Sarkar, Lakshmanan Nataraj and B. S. Manjunath. “Detection of Seam Carving and Localization of Seam Insertions in Digital Images”, Vision Research Laboratory University of California, Santa Barbara Santa Barbara, CA 93106
[6]. Michael Rubinstein, Ariel Shamir and Shai Avidan. “Context-Aware Saliency Detection”, ACM Transactions on Graphics, Vol. 27, No. 3, Article 16, Publication date: August 2008.
[7]. Weiming Dong, Ning Zhou, Jean-Claude Paul and Xiaopeng Zhang. “Optimized Image Resizing Using Seam Carving and Scaling”, ACM Transactions on Graphics 29, 5 (2009) 10 p.
[8]. [Online] CUDA Toolkit Documentation http://docs.Nvidia.com/cuda/cuda-c-programming-guide/#axzz3GnJjIKFk
[9]. [Online] http://en.wikipedia.org/wiki/CUDA
[10]. Jianbin Fang, Ana Lucia Varbanescu and Henk Sips. “A Comprehensive Performance Comparison of CUDA and OpenCL”,Parallel and Distributed Systems Group Delft University of Technology Delft, the Netherlands.