International Journal of Engineering Technology, Management and Applied Sciences

www.ijetmas.com January 2015, Volume 3 Issue 1, ISSN 2349-4476

Face Detection on CUDA

Raksha Patel
Isha Vajani
Computer Department, Uka Tarsadia University,Bardoli, Surat, Gujarat

Abstract

Face Detection finds an application in various fields in today's world. However CPU single thread implementation of face detection consumes lot of time, and despite various optimization techniques, it performs poorly at real time. With the advent of General Purpose GPU (GPGPU) and growing support for parallel programming language like CUDA, it has become possible to use GPU for such computational tasks. Our design model combines conventional programming using CPU with GP-GPU programming using NVIDIA CUDA for fast face detection. The face detection algorithm can be paralyzed to implement on the GPU using CUDA technology. Viola Jones face detector is one such application which when implemented on GPU, by exploiting some of the parallel aspects of algorithm, can give optimized output. The proposed method includes the enhanced Haar-like features and uses SVM for training and classification which can efficiently and effectively be implemented on GPU.

Keywords- CUDA; GPU; Face Detection; Viola and Jones algorithm

I. Introduction

Face detection in images is quite complicated and a time-consuming problem, which found use in different disciplines, e.g., security, robotics, or advertising. Face detection is the process of detecting faces in input images. Face detection is important because it is the first step in various applications like face recognition, video surveillance, Human Computer Interaction etc. Traditionally expensive dedicated hardware was used to achieve the desired rate of detection. The first Software-based real-time face detection algorithm was proposed by Viola and Jones [1]. Parallelization is the best way to achieve faster face detection. To detect face in a given image we need to run a sub window over the image and check whether it is a face or not. This particular step of checking a sub-window for face basically consists of executing same set of instructions on each sub window. The calculation of each sub window doesn't depend on any other sub window, which makes face detection a highly parallelizable problem. Since GPU is a Single Instruction, Multiple Threads processor it is very much suitable for performing these calculations in parallel and thus saving a lot of processing time. With the help of NVIDIA CUDA toolkit it is now possible to perform general purpose computations on GPU [1].

II. Related Work

The proposed architecture is mainly based on Viola and Jones face detection algorithm.
They introduced the concept of Integral Image, Haar-like features and Support vector machine classifier for face detection. The various real time application which can be implemented parallel and analyze the performance of the system on various platforms which can be used to determine which platform suits the best for given application. This is best suitable for application which gives maximum speed up factor. This work has been intended for the exploring the suitability of applications like histogram and face detection on CUDA and various other hardware platforms like

www.ijetmas.com January 2015, Volume 3 Issue 1, ISSN 2349-4476

CPU, GPU and on CPU for various software platforms Matlab. Its performance is compared. High performance face detection is useful for computer vision, as histogram evaluation and face detection is an important step in many image processing algorithms. Our design model combines conventional programming using CPU with GP-GPU programming using NVIDIA CUDA for fast face detection. The face detection algorithm can be paralyzed to implement on the GPU using CUDA technology.

III. Challenges for implementation of an algorithm on CUDA

There are various constraints to be considered while switching over a current system on CUDA platform.

- No support of recursive function. Any recursive function must be first converted into loops.
- Bus bandwidth of GPU and CPU is a bottleneck for many applications.
- Only supported on NVIDIA's GPUs. One must have a system with NIVIDIA GPU chip with many cores on it.
- Applications which depend upon previous phase computed data also cannot be implemented efficiently on CUDA.

Thus a study of these factors has to be made before deployment of application on a CUDA platform.

IV. Viola and Jones algorithm

Viola-Jones algorithm is real-time face detection system. A face detection algorithm must possess two key features, accuracy and speed. There are three ingredients working in concert to enable a fast and accurate detection: the integral image for feature computation, Adaboost for feature selection and an intentional cascade for efficient computational resource allocation. The Viola and Jones algorithm was used for the face detection, which was divided into two parts. The first part is for training a set of classifiers. Samples weak classifiers are created. The second part it is the detection itself, when faces are detected in the input image [4]. Two components, Integral Image and Features of Viola and Jones algorithm are used in the proposed architecture [1].

A. Integral Image

Each time computing intensity values for rectangle region would be time consuming task, therefore to speed up the computation, Integral images have been used. The integral value for each pixel is the sum of all the pixels above it and to its left. Starting at the top left and traversing to the right and down, the entire image can be integrated with a few integer operations per pixel. Since calculating the sum of pixel inside a rectangle requires the integral image values of the four corners only, the feature calculation can be done in constant time [1].

For parallel Integral Image calculation we have used the concept of prefix sum. The prefix sum is an operation on array in which each element in the resulting array is obtained from the sum of the elements in the operand array up to its index.Integral image calculation is divided into two steps:

Step 1: Take each row of the image in a temporary array and calculate its prefix sum. Figure 1 shows the row wise prefix sum.

www.ijetmas.com January 2015, Volume 3 Issue 1, ISSN 2349-4476

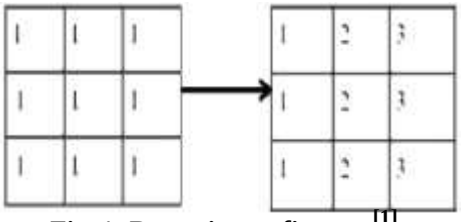

Fig. 1: Row wise prefix sum [1]

value is given by the difference between the sum of pixel intensities in the bright regions and the sum of pixel intensities in the dark regions. If the feature value is above pre-defined threshold then that feature is said to be present in searching window. The key advantage of a Haar-like feature over most other features is its calculation speed. Due to the use of *integral images*, a Haar-like feature of any size can be calculated in constant time [7].

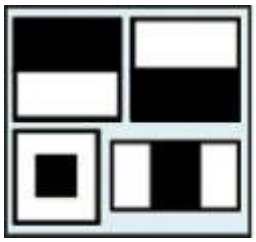

Fig.3: Haar rectangle Face [1]

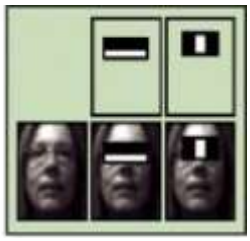

Fig. 4: Haar features on features [1]

Step 2: Take each column of the image obtained after step 1 in a temporary array and calculate its prefix sum. Figure shows the column wise image (integral image).

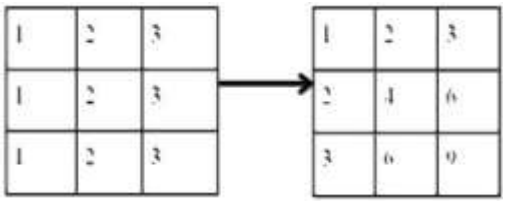

Figure 1: Column wise prefix sum [1]

Thus the proposed idea is to first calculate step 1 in parallel by assigning each row to different block. Then step 2 by assigning each column to different block. After these two steps we get the integral image.

B. Haar Features

The features used for face detection are called as Haar Features which are basically Haar wavelets. Haar wavelets are calculated over the region of adjacent rectangles. Feature

www.ijetmas.com January 2015, Volume 3 Issue 1, ISSN 2349-4476

V. Algorithm and It's GPU Implementation

Our proposed architecture uses both CPU and GPU for face detection on CUDA. CPU main work is the images convert it into grey scale and copy it to the Graphics Processor.
Proposed architecture for face detection is executed on GPU, and the language used is CUDA programming Language an extension to C programming language [1]. Algorithm for face detection is as follows:

1. Convert image into gray Scale.
2. Copy Image array to the GPU.
3. Calculate Integral Image.
4. For all Possible Sub-Window Sizes Do.
5. Create all sub-windows.
6. Assign each sub-window to different block.
7. For each sub-window Do.
8. Calculate feature for each sub-window in Parallel.
9. Classify the sub-window as face or non face using SVM.
10. Send the result back to the CPU.
11. Draw rectangle over the detected sub-window.
12. End.

The main steps involved in the algorithm are Integral Image calculation, feature extraction and SVM classification.

VI. Proposed Architecture

Our proposed architecture is shown in fig.-5. We used GPU SVM for both training and classification. Next sections give a detail about the training phase and the classification phase of the algorithm.

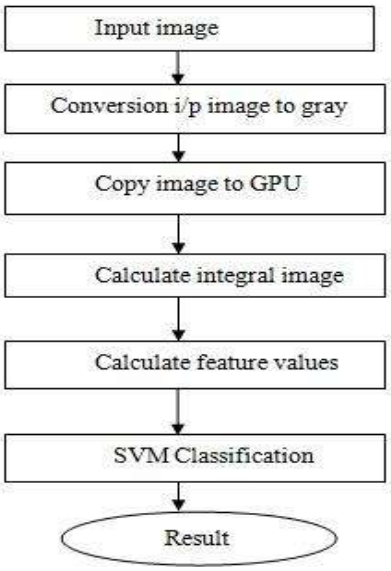

Fig.5 Proposed Architecture for face detection [1]

www.ijetmas.com January 2015, Volume 3 Issue 1, ISSN 2349-4476

A. Training Phase

FERET database is chosen as it contains a large number of images in unconstrained environment. For training face randomly chose any face from database, and for non-face created a database of non-face images.All sample images have to be converted to grey scale and resized [1].

B. Classification Phase

Classification phase will be carried out on the randomly chosen images from database. First the input image converted into grey scale. After that image copied the grey scaled image to GPU. Then a detection sub-window of size was run over the whole image. After processing the image with detection sub-window size of detection sub-window is increased by a factor. Since we are using GPU for computation and calculate features for all sub-windows in parallel and then use SVM for classifying the sub-window as face or non-face [1].

VII. Results

Implementation result has been carried out on GPU i.e. NVIDIA's GEFORCE 410M CUDA processor. The FERET face database implementation in Matlab 2013b. The output has been compared with CUDA and Matlab R2013b implementations of the same

Table 1: Execution times measurement

Algorithm	Execution Time		
	CUDA™	MATLAB	Speedup Factor
Vector Addition	130.0 ms	164.427 ms	1.26
Histogram	0.8 ms	2676.43 ms	3345.5
Face Detection	-	155.78 ms	-

The histogram calculation was performed faster for every size input image. The implementation results in execution times as well as the speed up factor measurement are shown above table.

VIII. Conclusion

There is an immense optimization in execution time of programs executed on CUDA. Various programs such as vector addition, histogram have been implemented on CUDA. Further implementations can be carried out on various images and implementations can fast face detection time can be measured. The implementation of histogram on Matlab, NVIDIA CUDA and enhances the performance and reduces execution time by performance analysis and comparative study with other implementations on Matlab and CUDA.

www.ijetmas.com January 2015, Volume 3 Issue 1, ISSN 2349-4476

Further implementations can be carried out on various images and implementations can also be carried out execution time can be measured and comparative study can be made.

XI. Future Work

Viola-Jones algorithm is real-time face detection system. A face detection algorithm must possess two key features, accuracy and speed. Viola and Jones presented a fast and robust method for face detection. The face detection in the face features are extract and execution times are measured.

Viola- Jones face detector is one such application which when implemented on GPU, by exploiting some of the parallel aspects of algorithm, can give optimized output. Future work includes implementing of Voila-Jones algorithm on CUDA GEFORCE 410M series with 48 cores.

References

1. Kailash Devrari, K.Vinay Kumar "Fast Face Detection Using Graphics Processor"(IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 2 (3), 2011.
1. Jaromír Krpec , Martin Němec " Face Detection CUDA Accelerating" The Fifth International Conference on Advances in Computer-Human Interactions,2012.
2. Bharatkumar Sharma, Rahul Thota, Naga Vydyanathan, and Amit Kale "Towards a Robust, Real-time Face Processing System using CUDA-enabled GPUs" 2009 IEEE.
3. Yi-Qing Wang, "An Analysis of the Viola-Jones Face Detection Algorithm" Published in Image Processing On Line, Vol 2, 2014. 5.http://en.wikipedia.org/wiki/classification methods
4. http://en.wikipedia.org/wiki/CUDA 7.http://en.wikipedia.org/wiki/Haar-like-features
5. Silberstein, Mark; Schuster, Assaf; Geiger, Dan; Patney, Anjul; Owens, John D. "Efficient computation of sum-products on GPUs through software-managed cache".Proceedings of the 22nd annual international conference on Supercomputing - ICS .2008
6. J.D. Owens, D. Luebke, N. Govindaraju, M. Harris,J. Krüger, A.E. Lefohn, and T.J. Purcell, "A survey of general-purpose computation on graphics hardware" on Computer Graphics Forum.
7. J.D. Owens, D. Luebke, N. Govindaraju, M. Harris,J. Krüger, A.E. Lefohn, and T.J.Purcell, "A survey of general-purpose computation on graphics hardware" on Computer Graphics Forum.
8. Lin Liu ; Jian-Ping Li ; Yuan-Yuan Huang ; Jie Lin ,"Data Classification based on Artificial Neural Networks" IEEE transactions on neural networks, vol.8, no.5,September 1997.
9. 12.http://en.wikipedia.org/wiki/FERETDB