Scientific Journal of Impact Factor(SJIF): 3.134

e-ISSN(O): 2348-4470
p-ISSN(P): 2348-6406

International Journal of Advance Engineering and Research Development

Volume 2,Issue 5, May -2015

Face Detection on a parallel platform using CUDA technology

Patel Raksha R.
M.Tech Student, Computer Department CGPIT, UTU, Bardoli, Gujarat, India
Ms. Isha K. Vajani
Assistant Professor, Computer Department, CGPIT, UTU, Bardoli, Gujarat, India

Abstract: *Face Detection finds an application in various fields like live video surveillance, bio-metrics, augmented reality, etc. However, CPU single thread implementation of face detection consumes lot of time, and despite various optimization techniques, it performs poorly at real time. Our design model combines conventional programming using CPU with multi core 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 algorithm when exploited in parallel, it gives optimized output. The speed up factor for various algorithms has been calculated. The algorithms involving rigorous computations can perform better on parallel platforms when broken up into parallel and executed concurrently.*

Keywords: *CUDA; GPU; Face Detection; Viola and Jones algorithm*

I. Introduction

Face detection in images is quite complicated and a time-consuming problem it has which found use in different disciplines, e.g., security, robotics and 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. Since all the mentioned applications are often used in real time, so there is a need of a fast face detection system. 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 - Jones [1]. It has now become the default standard for real-time face detection. However it doesn't suit well for images with high resolution, hence we need to look for high performance face detection solutions for fast face detection with reasonable cost. 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].

Face Detection Methods

There a large number of techniques and algorithms are suggested by researchers for face detection. Methods can be categorized in different ways. So many methods are combination of two or more algorithms and techniques [2].

1. Knowledge Based Method

These methods are based upon simple rules. Features of the face are described into simple rules. Rules normally are relationships between the features of the face. For example there may be a rule intensity in central region of the face is uniform and the also eyes on the face are symmetric. The technique also called top down [3].
Yang and Huang introduced a knowledge based method in 1994. It detects faces in three stages or levels. At first stage high level rules are applied. In second stage the histogram and equalization methods are used for edge detection and finally eyes and mouth features are found. Knowledge based method introduced by Kouropoulos and Pitas. Generalization of these methods for moving images and faces with poses also decrease the accuracy and performance [3].

2. Template Based Method

A larges number of templates of faces are coded into the system. The input face is matched with already stored. Faces are located by correlation. Also models are required to map the problem. The approach is simple to implement but not successful

for face detection and cannot deal with variations in scales, poses and shape. A number of methods proposed to overcome the variations in poses and illumination. P. Sinha used a set of image invariants to describe face pattern. The brightness of different parts of the face like eyes and nose changes while illumination conditions. The said methods calculated the pair wise ratios of brightness and projected directions. A face is localized by satisfying all conditions for dark and bright regions [3][4].

3. Feature Based Method

It is also known as bottom up approach. The main focus is to find the invariant features of the face in first step. The results are collected by integrating them. A number of different methods are proposed a number of researchers based on facial feature, texture, skin color and combination of different features. Edge detection, segmentation and histogram commonly used for extracting facial features, textures, skin color and other features from the images. Feature based are not successful when faces have different poses and illumination problems. Occluded faces with other objects are also difficult to detect [5] .

4. Appearance Based Method

Machine learning techniques are widely used for learning face and none face classes. There is no need of model for representation of image separately. All appearance based methods have common properties like classification of face and non face classes, preprocessing, learning and post processing of the images [5].

5. Viola and Jones Method

The Viola-Jones object detection framework is the first object detection framework to provide competitive object detection rates in real-time proposed in 2001 by Paul Viola and Michael Jones [5]. 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.

II. Comparisons of Face Detection Methods

Here a comparison of different methods is given. The accuracy is calculated face detected by a methods divided by total faces.

Table 1: Comparisons of Face Detection Methods

Methods	Faces Tested	False Detection	Accuracy %
Knowledge based method	136	3	88.5
Template based method	136	5	90.3
Feature based method	136	13	81.9
Appearance based method	136	20	60
Viola and Jones Method	136	2	99.8

The various face detection methods comparisons are shown above figure. 136 images are taken from the any source and this various method applied to 136 images. The 136 images in faces are tested. These images in faces are present or not. True detection means windows in face are present and False detection means windows in face is not present. Face detection methods of accuracy are shown above table [5]. The Viola-Jones method is better then the other methods.

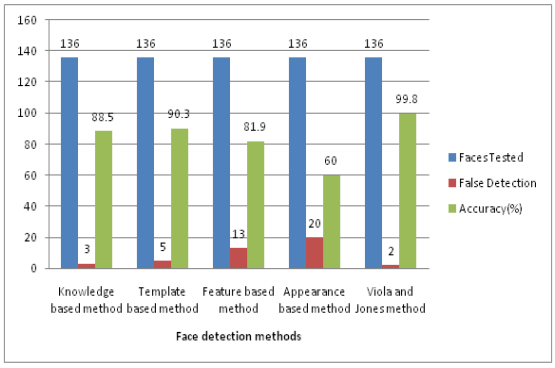

Fig.:1 Comparisons of Face Detection Methods

The various face detection methods comparisons are shown below figure. 136 images are taken from the any source and this various method applied to 136 images. The 136 images in faces are tested. False detection means non-face detected. Thus Viola-Jones algorithm is better then the other algorithms [5].

Introduction to CUDA

CUDA (Compute Unified Device Architecture) is a parallel computing platform and programming model created by NVIDIA and implemented by the graphics processing units (GPUs) that they produce.
The CUDA platform is accessible to software developers through CUDA accelerated libraries, compiler directives and extensions to industry-standard programming languages, including C, C++ and Fortran [11].
CUDA gives developers access to the virtual instruction set and memory of the parallel computational elements in CUDA GPUs. Using CUDA, the latest NVIDIA GPUs become accessible for computation like CPUs. Unlike CPUs, however, GPUs have a parallel throughput architecture that emphasizes executing many concurrent threads slowly, rather than executing a single thread very quickly. This approach of solving general-purpose problems on GPUs is known as GPGPU (General Purpose Graphics Processing Unit) [11].

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 NVIDIAs GPUs. One must have a system with NVIDIA GPU chip with many cores on it.
- Applications which depend upon previous phase computed data also cannot be implemented anciently on CUDA.
-

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

Process Flow of CUDA

On start CUDA's compiled code runs like any other application. Its primary execution is happening in CPU. When kernel call is made, application continues execution of non-kernel function on CPU. In the same time, kernel function does its execution on GPU. This way we get parallel processing between CPU and GPU. This is also known as heterogeneous programming. Memory move between host and device is primary bottleneck in application execution. Execution on both is halted until this operation completes [14].

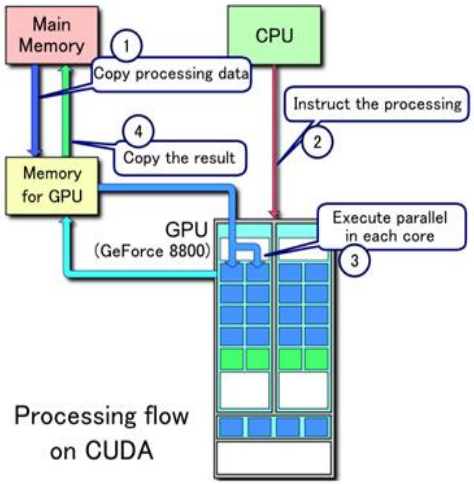

Fig.2: Process Flow of CUDA

The logical steps can be listed as below:
1. Copy input data from CPU memory to GPU memory
2. Load GPU program and execute, caching data on chip for performance
3. Copy results from GPU memory to CPU memory.

III. Proposed Work

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. Algorithm for face detection is as follows:

1. Convert input 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 cascade.
10. Send the result back to the CPU.
11. Draw rectangle over the detected sub-window.
12. End.

Our proposed architecture is shown in figure-3. We used GPU cascade for both training and classification. Next sections give a detail about the training phase and the classification phase of the algorithm.

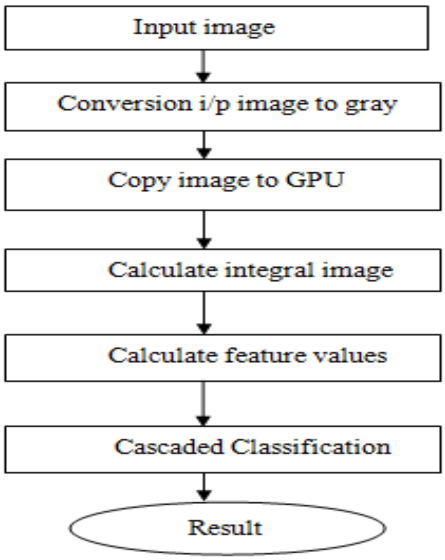

Fig.3: Block diagram of proposed system

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 .

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 cascaded for classifying the sub-window as face or non-face.

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 work- 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
Viola - Jones presented a fast and robust method for face detection.

The Viola - Jones face detection algorithm has mainly 4 stages [2]:
1. Haar Features Selection
2. Creating Integral Image
3. Adaboost Training algorithm
4. Cascaded Classiffiers

1. Haar Features Selection

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. Haar rectangle features figure shown as below.

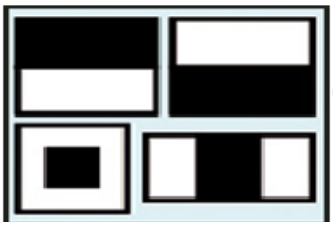

Fig.4: Haar rectangle features

Feature 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. Haar features on face figure shown as below .

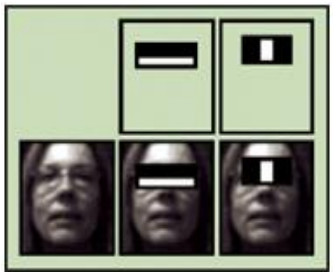

Fig.5: Haar features on face

The four features applied in this algorithm are applied onto a face.
Rectangle features Value = (pixels in black area) - fi(pixels in white area)

2. Creating 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 [7].
Row wise prefix sum figure shown as below.

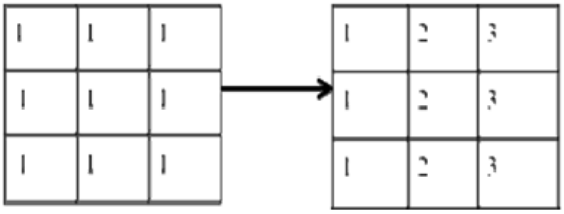

Fig.: 6: Row wise prefix sum

Column wise prefix sum figure shown as below.

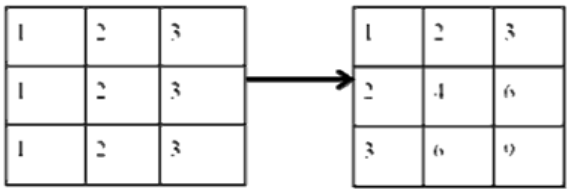

Fig.:7: Column wise prefix sum

3. AdaBoost Algorithm

A classifier has to be trained from a number of available discriminating features within a specific sub window in order to detect an object. The possible positions and scales of the five different feature types as shown in figure 3.2 produce about 90,000 possible

alternative features within a sub window size of 24x24 pixels. Therefore, a small set of features which are the best describe of the object, has to be selected. Adaboost is a
technique to select a good classification functions, such that a final strong classifier will be formed, which is in fact, a linear combination of all weak classifiers.

4. The Cascaded Classifier

The essential standard of the Viola-Jones face detection calculation is to sweep the
detector ordinarily through the same picture each one time with another size. Regardless of the fact that a picture ought to hold one or more faces it is evident that an unreasonable expansive measure of the assessed sub-windows might even now be negatives (non-faces). Instead of finding faces, the algorithm discards non-faces.

Cascaded Classifier figure shown as below .

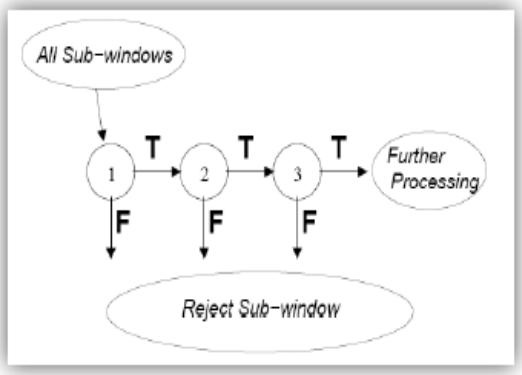

Fig.8: Cascaded Classifier

Results and Analysis

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. 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 below table.

Table 2: Execution times measurement

Algorithms	Execution Time		
	CUDA™	MATLAB	Speedup
Vector Addition	130.0 ms	164.427 ms	1.264
Matrix Multiplication	11.123 ms	14.772 ms	1.328
Histogram	0.8 ms	267.643 ms	334.5

Algorithms	Execution Time		
	CUDA™	MATLAB	Speedup
Face Detection	10.1 ms	155.78 ms	15.423

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

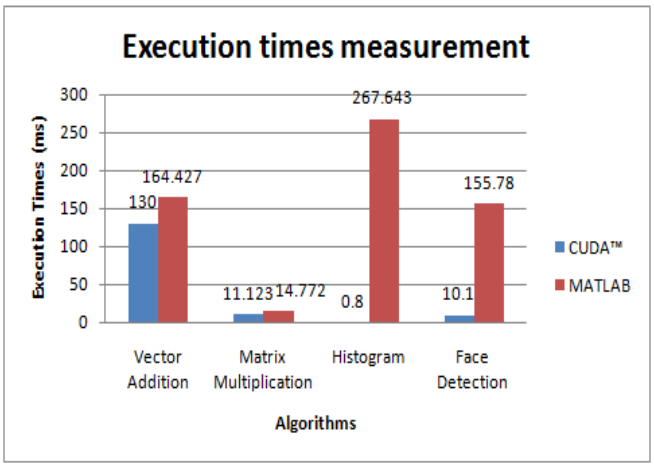

Figure 5.1: Comparison of CPU execution time with GPU execution time

IV. Conclusion:

There is an immense optimization in execution time of programs executed on CUDA platform. Various programs such as vector addition, matrix multiplication, and face detection application have been implemented on CUDA and MATLAB. It concluded that implementation of various programs on NVIDIA CUDA enhance the performance and reduces execution time.

Future Extension:

The future work now will be to work on higher complexity algorithm for reducing time complexity. The extraction of facial feature points, (eyes, nose, mouth) plays an important role in many applications, such as face recognition, face detection, expression recognition, facial animation and head pose determination which can be effectively implemented on CUDA platform to obtain speed up.

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.

2. Viola–Jones object detection framework - Wikipedia, the free encyclopedia.htm

3. Bharatkumar Sharma, Rahul Thota, Naga Vydyanathan, and Amit Kale "Towards a Robust, Real-time Face Processing System using CUDA-enabled GPUs" 2009 IEEE.

4. Yi-Qing Wang, "An Analysis of the Viola-Jones Face Detection Algorithm" Published in Image Processing On Line, Vol 2, 2014.

5. Divyarajsinh N. Parmar, Brijesh B. Mehta "Face Recognition Methods & Applications" International Journal of Computer Science Technology & Applications, Vol 4 (1), 2012

6. http://en.wikipedia.org/wiki/Haar-like-featuress

7. Paul Viola, Michael J. Jones, "Robust Real-Time Face Detection" International Journal of Computer Vision 57(2), 2004, USA(e-thesis)

8. http://en.wikipedia.org/wiki/classification methods

9.http://en.wikipedia.org/wiki/Sport_vector_machine

10. 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.

11. http://en.wikipedia.org/wiki/CUDA

12. Owens, John D., et al. "A Survey of general-purpose computation on graphics hardware." Computer graphics forum. Vol. 26. No. 1. Blackwell Publishing Ltd, 2007.

13. NVIDIA CUDA Compute Unified Device Architecture Programming Guide

14.http://www.nvidia.com/object/CUDAhome_new.html

15.http://hub.vscse.org/site/resource/CUDA_04.pdf

16. Jaromír Krpec , Martin Němec " Face Detection CUDA Accelerating" The Fifth International Conference on Advances in Computer-Human Interactions,2012.

17. 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

18. http://matrix/Matrix/Multiplication/step-by-step.htmls

19.http://en.wikipedia.org/wiki/FERETdatabase

20. http://en.wikipedia.org/wiki/Speedup

21. Shivashankar J.Bhutekar, Arati K. Manjaramkar "Parallel face Detection and Recognition on GPU" (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5 (2), 2014, 2013-2018

