Fast Face Recognition Approach Using a Graphical Processing Unit "GPU"

Yousri Ouerhani, Maher Jridi, Ayman Alfalou

Fast Face Recognition Approach Using a Graphical Processing Unit "GPU"

Yousri Ouerhani, Maher Jridi and Ayman AlFalou, Senior Member, IEEE
Département d'optoélectronique, Laboratoire L@bISEN
20 rue Cuirassé Bretagne CS 42807, 29228 Brest Cedex 2, France
e-mail: {yousri.ouerhani, maher.jridi and ayman.alfalou}@isen.fr

***Abstract*—In this manuscript, we present an implementation of a correlation method for face recognition application on GPU. Our correlator is based on the famous "4*f*" setup and the use of a Phase Only Filter (POF). Traditionally, the correlation approach is implemented using optical components for real-time application. Unfortunately, optical implementation is complex and has exorbitant price. To cope with these drawbacks and in order to benefit from the accuracy of the correlation method, we propose in this work to implement the correlation using GPU. To this end, we will take an interest in the mathematical aspect of the correlation method to identify the processing to be implemented on GPU. Simulations results about the implementation of the face recognition application on GPU showed the efficiency of our proposed design. Moreover, comparison between GPU and CPU in terms of execution time have been made and shows that, to identify one face among 4, GPU Nvidia Geforce 8400 GS is 3 times faster than the Intel Core 2 CPU 2.00 GHZ.**

I. Introduction

Over the last few years, interest in face recognition has increased for military and civil applications. The trend in these applications is to realize the face recognition in a low computing time. In this context, we have turned to techniques such as optical correlation which benefit from parallelism provided by optical components. In the literature, there have been many approaches implementing face recognition methods according to specific applications. In [1], the authors proposed a tutorial on the various correlation techniques. It has been shown that these techniques are efficient in terms of detection (very good recognition rate with low false alarm rate) and low computing time.

In [2], the authors use the correlation technique to recognize the sign language and retranscribe it into a written or oral language in order to make mobile communication possible between a deaf-mute and people who don't know the sign language.

In some applications and for convenience and safety reasons (recognition of people in the subway) the correlator must be physically separated from cameras filming the scene. Thus it is necessary to save and/or transmit images to the correlator. However applying the conventional methods of images compression and encryption is very restrictive because it leads to decrease the correlator performances. To cope with this problem, the authors in [3] proposed many compression and encryption techniques adapted to face recognition using correlation methods.

Traditionally, optical devices are used to achieve the correlation. The low processing time is the major advantage of these devices. These devices are not portable and have an exorbitant price for civil applications. To solve this problem, we propose to implement this correlation method on digital electronic components. In fact, algorithms of image processing are implemented on different targets such as DSP, CPU and GPU in order to obtain a high quality with a low execution time. Unfortunately, it is often difficult to combine the high resolution and the reduced execution time. In this work a special interest is given to the implementation of the correlation technique for the face recognition on GPU. This kind of processor has been proposed for two reasons: to solve the compromise quality/time and reduce the cost of obtained circuit. In fact, results reported in several recent scientific references for different applications show that the parallel architecture of GPU allows pipelined computing and consequently reduces constraints for real-time applications [4]. It should be outlined that the principle of the proposed face recognition approach based on correlation technique is to compare the degree of similarity between the target object (face to recognize) with several references objects. The correlation technique is performed using the mathematical model based on Fourier Transform FT and detailed in [1].

This manuscript is organized as follows: in section II, a performance comparison between GPUs and CPUs is made to justify the choice of GPU. Section III is devoted to the description of the Fast Fourier Transform (FFT) algorithm and 4*f* setup to be used in the correlation technique. In section IV, GPU experimental results are presented in order to show the efficiency of the proposed approach. Finally, we conclude this paper by presenting new research directions to improve the proposed design.

II. Why GPU?

A. Evolution

Today, CPU frequency no longer follows Moore's law due to the increase of the CPU consumed energy and physical limitation [5]. To cope with this problem, researchers over the word propose to use many parallel processors. One interesting solution using multiprocessors for graphical processing is the GPU. Since 2003, researchers have attempted to use GPU originally designed for computing 3D functions. This includes lighting effects, object transformations, and 3D motion [6].

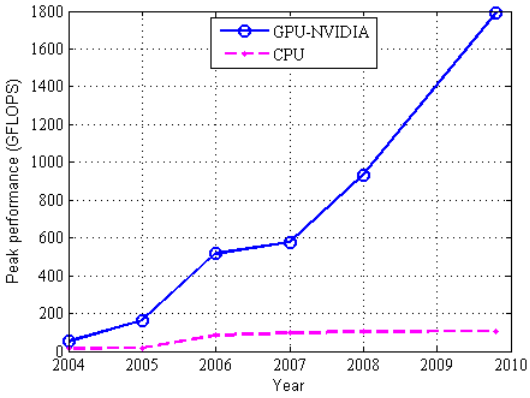

Fig. 1. CPU versus GPU evolution

Today, GPU is used for intensive computing and can be considered as multiple cores with a software layer that allows parallel computing. Contrary to CPU, the state of the art of GPU shows that the performances in term of execution time are in constant evolution. We extended the study of [7] to estimate the performances of GPU and CPU in 2010. The results of this study are summarized in figure 1 which shows that in 2010 the ratio between the GPU and CPU maximum operating frequencies is more than 15. This means that image processing through GPU is faster than on CPU. The computing speed is measured by GFLOPS (Giga FLoating point Per Second) which is equal to 10^9 floating-point arithmetic operations per second. The GPU family used in this comparison is NVIDIA.

B. Architecture

GPU architecture is composed of several multiprocessors, themselves containing several physical devices. For example, the GPU card used in this work is an Nvidia 8400 GS graphics card. As shown in figure 2, this GPU has only one Stream Multiprocessor (SM) with 8 graphic processors named Stream Processor (SP) and 2 Super Functions Unit (SFU) which are specific units used to compute elementary functions (exponential, logarithm, sine, cosine, ...). Each multiprocessor (SM) can process parallel groups of threads, called warps. On the other hand, each SM has a memory of 16 KB size that is shared by the processors within the multiprocessor. The instruction Fetch/Dispatch is a scheduler that dispatches instruction and data to be executed and saved on cache memory Instruction L1 and Data L1. Each multiprocessor is based on Single Input Multiple Data (SIMD) architecture, to perform intensive computation of highly parallelizable algorithms. On one clock cycle, each processor executes one instruction [8]. Consequently, the execution time of algorithms on GPU is determined by (1).

$$Time = \frac{Number\ of\ threads}{number\ of\ processors} \quad (1)$$

Finally, it is important to notice that to program the GPU, we should use the development toolkit CUDA [9] that allows

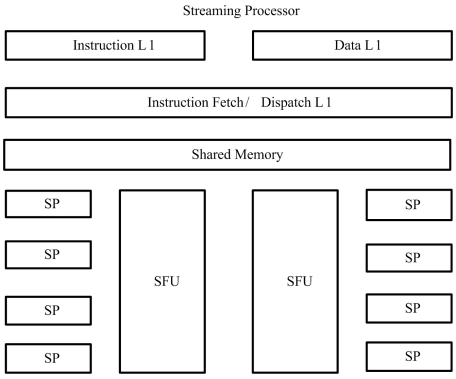

Fig. 2. GPU Architecture

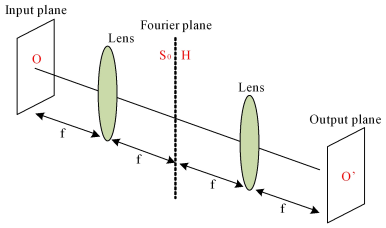

Fig. 3. $4f$ optical setup

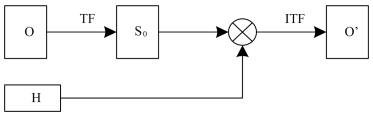

Fig. 4. $4f$ setup

a massively parallel programming. The goal of the implementation of the face recognition on GPU is double: first, we take advantage of the parallel processing to optimize the execution time. Second, we ensure a high face detection quality. Before detailing our numerically approach, we begin by introducing the principle of the correlation method used to face recognition.

III. FFT-$4f$ Setup: Correlation Method

A. Principle

One way of the realization of the $4f$ setup is the optical one shown in figure 3. According to this figure, the $4f$ setup, implementing the correlation, is an optical system composed of two convergent lenses and three planes: input, Fourier and (output) correlation planes. The 2D target-object O is illuminated by a monochromatic wave. A first convergent lens performs the Fourier transform S_0 of the input object at the Fourier plane. Then, a specific correlation filter H is put using specific optoelectronic devices [10]. Next, a second convergent lens performs the inverse Fourier transform (IFT) at the output plane of the system to get the correlation plane.

As we said previously, in order to reduce the complexity of the optical implementation and to ensure the configurability of the correlation method this optical setup is replaced by a digital one illustrated in figure 4. However, to implement the

TABLE I
PROCESSING TIME OF MATRIX MULTIPLICATION ON GPU (MS)

Matrix size	16x16	32x32	64x64	128x128	256x256
Execution Time	0.092	0.1121	0.2502	1.2713	9.1691

correlation setup on GPU we should examine the possibility of implementing the 2D Fourier Transform. One well-known algorithm of 2D-FT is the 2D Fast Fourier Transform 2D-FFT. In order to simplify the study of this algorithm we take an interest in the 1D-FFT defined by the next equation:

$$F_j = \sum_{k=0}^{n-1} x_k e^{-\frac{2i\pi}{n}jk} \tag{2}$$

Where $(x_0, ..., x_{n-1})$ is the input vector and n is the length of this vector. The form of (2) is equivalent to the matrix multiplication form shown in (3) where $w = e^{-\frac{2i\pi}{n}jk}$.

$$\begin{pmatrix} F_0 \\ F_1 \\ F_2 \\ \vdots \\ F_{n-1} \end{pmatrix} = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & \mathrm{w} & \cdots & \mathrm{w}^{(n-1)} \\ 1 & \mathrm{w}^2 & \cdots & \mathrm{w}^{2(n-1)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \mathrm{w}^{(n-1)} & \cdots & \mathrm{w}^{(n-1)^2} \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{n-1} \end{pmatrix} \tag{3}$$

By using (3) we can reach to the 2D-FFT by replacing the input vector by the image matrix and by computing two times the multiplication of the input image by the matrix w.
To evaluate the performance, in term of computing time, matrix multiplication for different matrix sizes applied to input image and w matrix has been made. The computing times (expressed in ms) of these multiplications are shown in Table I. According to these results, for a 256x256 matrix multiplication the execution time on GPU is about 9 ms. However, as we said before, to implement the 4*f* setup on a GPU, one 2D-FFT and one 2D-IFFT applied to image size of 256x256 are required. Consequently, 4 matrix multiplications are required and the total execution time reaches 36 ms. On the other side, to implement the face recognition application, the matrix multiplication is used with some additional processings related to the decision of the recognition. Consequently, the overall execution time is a heavy burden for real-time application. In order to impove the algorithm computing time we use CUFFT library provided by CUDA. In fact, CUDA FFT is based on Fastest Fourier Transform in the West (FFTW) algorithm [11]. The FFTW uses many algorithms such as Cooley-Tukey algorithm [12], prime factor algorithm [13], Rader's algorithm for prime sizes [14] and split-radix algorithm [15] in order to achieve best performances. By using this library, the 2D-FFT followed by a 2D-IFFT execution time is less than 8 ms for image size of 256x256. This algorithm will be used in the implementation of the face recognition approach.

IV. Face Recognition

To validate the digital implementation of the correlation method (4*f* setup), a face recognition application is considered.

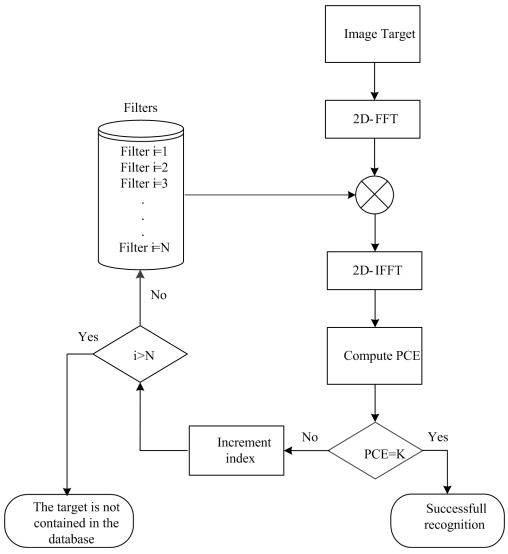

Fig. 5. Face recognition algorithm

A. Principle algorithm

The principle of the adapted correlation algorithm implementing the face recognition application on a GPU NVIDIA is shown in figure 5. Basically, this technique is based on the multiplication of the spectrum of the target image by a correlation filter, made from a reference image. The result is a more or less intense central correlation peak depending on the degree of similarity between the target object and the image reference [1]. Finally, to make a decision about the recognition, the criterion called Peak to Correlation Energy (PCE) is computed on the GPU and compared to a fixed threshold. The PCE is defined as the ratio between correlation peak energy and correlation plane energy [16].

$$PCE = \frac{C_{\varphi}^2}{\sum_i C_{x_i}^2} \tag{4}$$

Where C_{φ}^2 is the correlation peak energy and $\sum_i C_{x_i}^2$ is the correlation plane energy. Using the face recognition algorithm two decisions are possible: if the PCE is superior to the threshold K, the target face is recognized as the reference face used to manufacture the correlation filter; otherwise, a counter will increment the index of the correlation filter to be used for the next correlation. This operation will continue until obtaining index higher than the number of the filters saved in the database. In this case, we get the following decision: the target face is not contained in our database.

B. Test and validation

Implementation results are shown using only 4 images but the proposed face recognition method can supports much more images.
Many simulations have been conducted using 256x256 gray scale images encoded on 8 bits. The first two images shown in figure 6 are downloaded from [17].
In the case of successful recognition, an example of corre-

Fig. 6. Input images

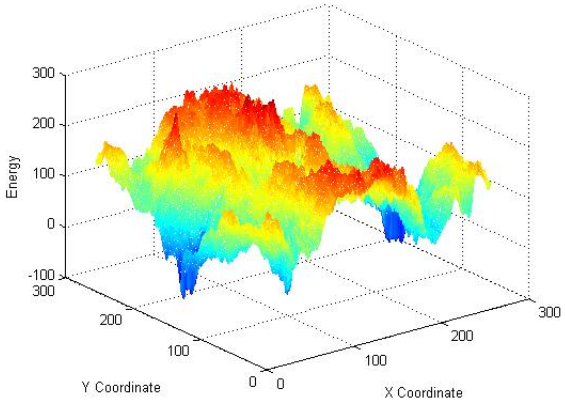

Fig. 8. Correlation plan for no similar images

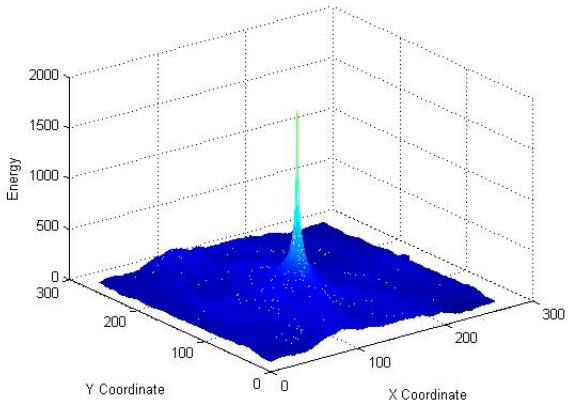

Fig. 7. Correlation plane: target image is similar to the reference image

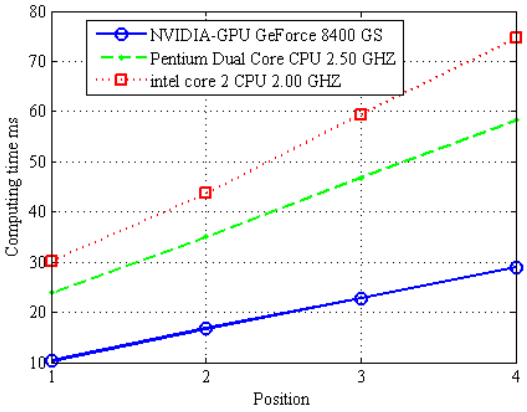

Fig. 9. Comparison of processing time for face recognition on GPU Nvidia Geforce 8400 GS and CPU

lation plane is shown in figure 7. In this output plane, i.e., correlation plane, we have an intense correlation peak located at the center. That means a sharp depending on the degree of similarity between the target and the reference images. In the case of failed recognition (references images is not contained in the data base) the correlation result is shown in figure 8. In this correlation plane there is no correlation peak i.e. there is no similarity between the target and the reference images. In all cases, the execution time depends on the position of the reference filter, i.e., index i of the filter in the database. In the case of GPU implementation using 4 filters the computing times vary between 10.34 ms and 28.99 ms. Moreover, to show the speed-up using GPU, comparisons between GPU and CPU implementation are done. To achieve this comparison, a Matlab model for the face recognition algorithm has been developed for the CPU implementation. Comparisons in terms of execution time between these two implementations are illustrated in figure 9. We can notice that the GPU Nvidia Geforce 8400 GS execution is 3 times faster than an Intel Core 2 CPU 2.00 GHZ and 2.5 times faster than a Pentium Dual Core CPU 2.50 GHZ. An extended study of this method applied to a database length of about 1000 images shows that the execution time is lower than 6 s when the position of the reference image is the last one or not contained in the database.

ACKNOWLEDGEMENT

The authors thank Dominique Maratray for her help and advice.

V. CONCLUSION

Through this paper, we have proposed and validated an approach implementing the correlation method digitally on a GPU NIVIDIA for a face recognition application. The choice of GPU target is comforted by comparisons between a GPU and CPU implementations. It has been shown that the execution time using GPU manufactured in 2006 is three times lower than recent CPU. This speed-up grows exponentially with the image sizes. Our future works for the digital implementation of face recognition will be devoted to FPGA implementation to take benefits from the speed and the configurability.

REFERENCES

[1] A. Alfalou and C. Brosseau, *Understanding Correlation Techniques for Face Recognition: From Basics to Applications,* in *Face Recognition,* In-Tech, ISBN 978-953-7619-00-X, 2010.

[2] M. Elbouz, A. Alfalou and H. Hamam, *Mobil phone camera Recognition Sign Language using a segmented multidecision filter adapted to the Parallel virtual machine (PVM),* International Journal of Hybrid Information Technology : IJHIT Journal, vol. 1, No 2, pp. 101-108, 2008.

[3] A. Alfalou, M. Elbouz, M. Jridi and A. Loussert, *A new simultaneous compression and encryption method for images suitable to optical correlation,* Optics and Photonics for Counterterrorism and Crime Fighting V, edited by Colin Lewis, Proc. of SPIE, vol. 7486, 74860J-1-8, 2009.

[4] J.D. Owens, M. Houston, D. Luebke, S. Green, J.E. Stone and J.C. Phillips, *GPU Computing,* Proc. of IEEE, vol. 96, pp. 879-899, May 2008.

[5] S.H. Park, D.R. Shires and B.J. Henz, *Coprocessor Computing with FPGA and GPU,* 3rd ed. DoD HPCMP Users Group Conference. Seattle, WA, pp. 366-370, July 2008.

[6] K. Moreland and E. Angel, *The FFT on a GPU,* Proc. of Graphics Hardware, pp. 112-136, 2003.

[7] E Stephanie and K. Ronan, *Etat de l'art des processeurs du marche dans le contexte du codec H.264,* Technical report Projet TransMedi@, Institut Telecom/Telecom-Bretagne, Departement Informatique, Ver. 1.0, May 2009.

[8] C. Deqi, L. Ningfang and C. Weiliang *Study of ARFTIS reconstruction model on GPU,* ISECS International Colloquium on Computing, Communication, Control and Management, vol. 2, pp. 192 - 195, 8-9 Aug. 2009.

[9] NVIDIA, NVIDIA CUDA Programming guide version 2.0, Available on : http://www.developer.download.nvidia.com

[10] A. Alfalou, G.Keryer and J. L. de Bougrenet, *Optical implementation of segmented composite filtering,* Applied Optics, vol. 38, pp. 6129-6135, 1999.

[11] M. Frigo and S.G. Johnson, *The Design and Implementation of FFTW3,* Proc. of IEEE, vol. 93, no. 2, pp. 216-231, 2005.

[12] J.W. Cooley and J.W. Tukey, *An algorithm for the machine computation of the complex Fourier series,* Mathematics of Computation, vol. 19, pp. 297-301, April 1965.

[13] A.V. Oppenheim and R.W. Schafer, *Discrete-time Signal Processing,* Englewood Cliffs, NJ 07632: Prentice-Hall, 1989.

[14] C.M. Rader, *Discrete Fourier transforms when the number of data samples is prime,* Proc. of IEEE, vol. 56, pp. 1107-1108, June 1968.

[15] P. Duhamel and M. Vetterli, *Fast Fourier transforms: a tutorial review and a state of the art,* Signal Processing, vol. 19, pp. 259-299, April 1990.

[16] J.L. Horner, *Metrics for assessing pattern-recognition performance,* Applied Optics, vol. 31, pp. 165-166, 5 March 1991.

[17] Face Databases From Other Research Groups, Available on : http://www.ecse.rpi.edu/%7Ecvrl/database/other_Face_Databases.htm