International Journal of Mechatronics, Electrical and Computer Technology

Vol. 3(8), Jul. 2013, pp 162 - 182, **ISSN**: 2305-0543
Available online at: http://www.aeuso.org

GPU Implementation of Real-Time Biologically Inspired Face Detection using CUDA

Zeinab Farhoudi[1*], Ali Broumandnia [2], Elham Askary[1], Sara Motamed[1]

[1] Department of Computer Engineering, Islamic Azad University Science and Research Branch, Tehran, Iran, z.farhoudi@gmail.com
[2] Department of Computer engineering South Tehran branch, Islamic Azad University, Tehran, Iran, broumandnia@gmail.com

*Corresponding Author's E-mail: *z.farhoudi@gmail.com*

Abstract

In this paper massively parallel real-time face detection based on a visual attention and cortex-like mechanism of cognitive vision system is presented. As a first step, we use saliency map model to select salient face regions and HMAX C1 model to extract features from salient input image and then apply mixture of expert neural network to classify multi-view faces from nonface images. The saliency map model is a complex concept for bottom-up attention selection that includes many processes to find face regions in a visual science. Parallel real-time implementation on Graphics Processing Unit (GPU) provides a solution for this kind of computationally intensive image processing. By implementing saliency map and HMAX C1 model on a multi-GPU platform using CUDA programming with memory bandwidth, we achieve good performance compared to recent CPU. Running on NVIDIA Geforce 8800 (GTX) graphics card at resolution 640×480 detection rate of 97% is achieved. In addition, we evaluate our results using a height speed camera with other parallel methods on face detection application.

Keywords: Bottom-up Attention, CUDA, GPU, HMAX C1, Saliency map.

International Journal of Mechatronics, Electrical and Computer Technology

Vol. 3(8), Jul. 2013, pp 162 - 182, **ISSN**: 2305-0543
Available online at: http://www.aeuso.org

1. Introduction

Face detection in images is one of the most challenging and a time-consuming problem, which is widely used in many application domains, e.g., security, robotics, or surveillance. By computer performance, disciplines of image processing, such as face detection, have significantly improved. The research in this field includes methods to extract useful information for proper characterization of a visual scene [1], thus making possible its analysis for face detection. One of these techniques is by estimating the saliency face map to find the regions of faces and then use HMAX C1 model for face non-face attentional regions recognition.

In most applications in the robotics domain, robots have specific tasks and their attention is top-down based and attracted by task-relevant information in the environment. Moreover, a high speed bottom-up attentional pre-selection is the first step of efficient visual information processing due to limited computation capability. So far, there have been many bottom-up based visual attention models, the saliency map model proposed in [2] has become a standard computational model in this domain. However, due to color channel, intensity channel, image rescaling and other feature extraction processing for face detection, the saliency map generation is very compute-intensive.

To allow this biologically inspired model to be applied on mobile robots, a real-time implementation is much-needed. We implement this bottom-up attention model on multiple Graphics Processing Units (GPUs) using the CUDA Architecture [3], which provides an excellent speed-up performance of the saliency map computation. Using 4 NVIDIA GeForce 8800 (GTX) graphics cards for 640×480 resolution of the input images, the computational cost is only 3.1 ms with a frame rate of 313 fps. The saliency map generation on GPUs is approximately 8.5 times faster than the standard CPU implementations [7].

International Journal of Mechatronics, Electrical and Computer Technology

Vol. 3(8), Jul. 2013, pp 162 - 182, **ISSN**: 2305-0543
Available online at: http://www.aeuso.org

The study of the visual cortex in primate brains allowed for a quantitative object recognition model which has a strict feed forward structure. The model was first applied to the recognition of real world images by Serre et al. [8]. They achieved good performance even with a small training set. Hierarchical maximum (HMAX) C1 model for face non-face recognition was examined in [9] and get good results. Several parallel GPU implementations for various cortex-like recognition systems exist today. Recently, Mutch et al. developed a more general framework to easily cover various cortical models [16]. They interfaced CUDA from within Matlab to speed up time consuming calculations while keeping an easy to use interface.

This paper is organized as follows: In section 2, saliency map and HMAX C1 processing are described, in Section 3 the concept of GPU and CUDA is briefly introduced, in section 4 our multi-GPU implementation of bottom-up attention based on the saliency map and visual cortex C1 layer model proposed, is explained. In Section 5 our implementation is discussed and our experimental results are shown. Conclusions are given in Section 6.

2. Saliency Map

2.1. Implementation

The saliency map model first suggested by Itti and Koch [2] and then extended by Yee [3]. We have, however, made several improvements and additions to the model. The original algorithm was designed for image processing and computer vision applications. Within these applications it is assumed that there is no prior knowledge of the environment. Additionally the model is well suited for directing attention to a certain area for further face detection processing in, for example, robot vision. These models also suffer from lengthy execution times. Because of the complexity of the calculations required, especially when using high resolution images, this time can be in the order of many seconds. This is unacceptable for a system that we hope will approach interactive rates. The saliency map model [2] is presented

International Journal of Mechatronics, Electrical and Computer Technology

Vol. 3(8), Jul. 2013, pp 162 - 182, ISSN: 2305-0543
Available online at: http://www.aeuso.org

by Itti and Koch. From an input image a saliency map is generated to predict the most salient face region in the input image based on human face color, intensity and shape. Firstly, the input image is subsampled into Gabor pyramids in three channels (intensity, orientation for 0◦, 45◦, 90◦, 135◦, opponent color in red/green and blue/yellow). Then, center-surround differences are calculated for the images in the Gabor pyramids.

In this step a subtraction between the maps with fine scales and the maps with coarse scales is conducted. After a normalization of the output images of the subtraction, feature maps are generated in which the distinctive pixels with respect to their neighborhood are highlighted. Using across-scale combinations the feature maps at different scales are combined and normalized into a conspicuity map in each channel. The conspicuity maps illustrate the conspicuous pixels regarding color, intensity and orientation. The saliency map is a linear combination of the conspicuity maps. The bright pixels in the saliency map are the most salient and interesting pixels predicted by the saliency map model. Since the generation of saliency map consists of a series of linear filtering, image rescaling etc., the implementation becomes a severe problem for real-time application.

In 2005 a distributed visual attention approach on a humanoid robot is proposed in [5]. In this model 5 different channels including colors, intensity, orientation, stereo and motion are used. The saliency map attention processing is distributed on 8 PC computers and a frequency of 30 fps with input images of 320×240 pixels is achieved. In [6] a GPU based saliency map for high-fidelity selective rendering is proposed. In this implementation a motion map, depth map and habituation are integrated. Also, they use a Sobel filter instead of the complex Gabor filter to produce the orientation maps. No CUDA technology was used.

2.2. S1 and C1 Standard Visual cortex Model

The recent studies in object recognition for primate visual cortex have proved that feedforward path way of object recognition is performed in preliminary processing in the

International Journal of Mechatronics, Electrical and Computer Technology

Vol. 3(8), Jul. 2013, pp 162 - 182, **ISSN**: 2305-0543
Available online at: http://www.aeuso.org

ventral stream [10]. The first layer of the model quantitatively corresponds to the tuning properties of V1 cells along the ventral stream of visual cortex. This layer consists of two hierarchical layers of units composed of simple S1 units and complex C1 units [11]. This layer of our framework is mostly based on that of Serre and Poggio et al. [8], with small changes according to the base model which consist of four hierarchical layers of computational units composed of alternating simple S units and complex C units. This layer is summarized as follow [9].

S1 Units: A gray-value input image is first analyzed by a multidimensional array of simple S1 units which correspond to the classical simple cells in the primate visual cortex (V1) [12]. At first stage, S1 responses are obtained by applying a battery of Gabor filters to the input images, which have been shown to provide a good model of cortical simple cell receptive fields. Indeed, the S1 units extract features with a localized, oriented-edge detection on the image, with help of 2D Gabor filter, that react to line of a particular orientation, scale, and position. Gabor function can be described by the following equation:

$$F(x,y)=\exp\left(-\frac{\left(x_0^2+\gamma^2 y_0^2\right)}{2\sigma^2}\right)\times\cos\left(\frac{2\pi}{\lambda}x_0\right). \quad (1)$$

$$x_0 = x\cos\theta + y\sin\theta. \qquad y_0 = -x\sin\theta + y\cos\theta.$$

where the aspect ratio γ=0.3, the orientation θ and the effective width σ , the wavelength λ were adjusted so that the tuning calls of the corresponding S1 units match those of V1 parafoveal simple calls as closely as possible. In our model, we arranged the S1 filters to form a pyramid of scales, spanning a range of sizes from 7×7 to 21×21 pixels in steps of two pixels and considered four orientations (0°, 45°, 90°, 135°), thus leading to 32 different S1 receptive field types (8 scales×4 orientations), and the C1 layer and S1 layer parameters have been arranged as [12].

International Journal of Mechatronics, Electrical and Computer Technology

Vol. 3(8), Jul. 2013, pp 162 - 182, ISSN: 2305-0543
Available online at: http://www.aeuso.org

C1 Units: The next C1 units correspond to cortical complex cells which illustrate some tolerance to shift and size. Complex cells tend to have larger receptive field and to be more tuned than simple cells. C1 units pool over afferent S1 units from the previous layer by performing nonlinear MAX-like operation over the same orientation within the neighborhood of S1 units in both space and scale. In other words, the response r of a complex unit is the strongest response of its m afferents (x1,…,xm) from the previous S1 layer such that:

$$R = \max_{i=1\ldots m} x_i . \qquad (2)$$

The response of the C1 unit is obtained by sub-sampling adjacent S1 maps using a cell grid of size Ns×Ns= 8×8. From each grid cell, the maximum of all 32 elements gives one single measurement. As a last stage, by taking a maximum over the two scales, for each call consider the maximum value from the two maps. Again, this process is independently repeated for each of the four orientations and each scale band.

Classification stage:

Since the acquired information, resulted in processing our visual environment, in cortex is tremendous, and as a result many processes are required to recognize a typical object, this huge obtained data cannot be directly implemented and inserted into artificial systems. Therefore, PCA is employed to reduce data dimension. According to the structure of model [9], the Mixture of Experts Neural Network (ME) is employed for learning to recognize multi-view face nonface patterns. The ME networks are not biased to prefer one class of faces to another; in that, the network itself partitions the face space into subspaces and decides which subspace should be learned by which expert. The advantage of using this classification is its parallel computational instinct for increasing speed at running time.

International Journal of Mechatronics, Electrical and Computer Technology

Vol. 3(8), Jul. 2013, pp 162 - 182, ISSN: 2305-0543
Available online at: http://www.aeuso.org

3. Graphics Processing Units (GPUS)

In the last few years, the Graphics Processing Units (GPUs) have become more popular. GPU is specialized for compute-intensive, highly parallel computation. Moreover, CUDA, a new hardware and software architecture issued by NVIDIA in 2007, allows issuing and managing computations on the GPU as a data-parallel computing device without the need of mapping them in a graphics API [4]. GPUs are similar to multi-core CPUs but with two main differences (see Figure 1).

CPUs a made for speedup and GPUs for throughput. CPUs try to improve the execution of a single instruction stream while GPUs take the opposite route obtaining benefits from massively threaded streams of instructions and/or data (SIMD). The second difference is how threads are scheduled. The operating system schedule threads over different cores of a CPU in a preemptive fashion. GPUs have dedicated hardware for the cooperative scheduling of threads. CUDA has several advantages over traditional general-purpose computation on GPUs (GPGPU) using graphics APIs:

- Scattered reads – code can read from arbitrary addresses in memory
- Shared memory – CUDA exposes a fast shared memory region (up to 48KB per Multi-Processor) that can be shared amongst threads. This can be used as a user-managed cache, enabling higher bandwidth than is possible using texture lookups.
- Faster downloads and readbacks to and from the GPU
- Full support for integer and bitwise operations, including integer texture lookups

The saliency map and HMAX C1 computation consists of compute-intensive filtering (channels) in different scales, which is nevertheless highly parallelize. For real-time application we implemented the computation of our proposed model on GeForce 8800 (GTX) graphics cards of NVIDIA, which support the CUDA technology. Here, we take GeForce 8800 cards as an example. The other new products, which are compatible with the

International Journal of Mechatronics, Electrical and Computer Technology

Vol. 3(8), Jul. 2013, pp 162 - 182, **ISSN**: 2305-0543
Available online at: http://www.aeuso.org

CUDA technology, can also be used. The GeForce 8800 (GTX) consists of 16 multi-processors which consists of 8 processors each. All the processors in the same multiprocessor execute the same instruction, but with different data. This concept enables a massively parallel computation of a large amount of similar data. The GeForce 8800 (GTX) has a core clock frequency of 575 MHz and a 768 MB memory. The multi-GPU performance is strongly dependent on an efficient usage of the thread-block concept and different memories.

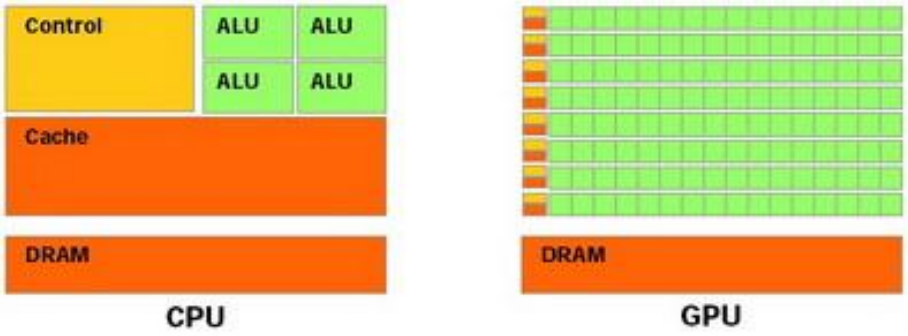


Fig. 1: Compare GPUs with multi-core CPUs architecture [4].

4. GPU Implementation of Proposed Model

A data flow diagram of our proposed model is illustrated in fig 2. The proposed bottom-up module adopts the bottom-up part of the VOCUS proposed by [2]. In the saliency-based bottom-up module, we use the skin color feature for extracting a face-like area and center-surround differences computed for the images in the Gabor pyramids. Then, we construct two feature maps (skin color and 4 orientations Gabor filter) by summing up the two scale maps found within each scale. The feature maps are based on the scale invariant saliency, which makes it possible to detect the salient regions at an image scale matching the face size. Finally, we apply HMAX C1 model to the salient regions for extracting features and classify them with mixture of expert neural network for face detection application. The

International Journal of Mechatronics, Electrical and Computer Technology

Vol. 3(8), Jul. 2013, pp 162 - 182, ISSN: 2305-0543
Available online at: http://www.aeuso.org

saliency map process, speed up face detection to find salient regions instead of windowing all input image and with HMAX C1 model presented in [16] was achieved best results. Then a multi-GPU implementation of our proposed model is described, which is divided into 9 parts, described in detail as Figure 2.

A. Initialization

The GPU should be firstly initialized. For the reason that the memory allocation in GPU takes very long, the memory is firstly allocated for different images such as the input images, the images in the Gaussian dyadic pyramids, the feature maps, the conspicuity maps, the rescaled feature and conspicuity maps at the same size as well as the saliency map. Since the filter kernel will not be changed during the saliency map computation, we also calculate the Gabor filter in the initialization phase in the CPU and then transform it into the frequency domain. Then we convert the image data type with uchar4 into float4 to achieve a high precision for the following computation. The texture memory provides an implicit possibility to do the type conversion by means of 2D-texture.

B. Data type conversion

The input image has the resolution of 640×480 and three 8-bit channels, namely red, green and blue. The image data are copied from the CPU into the global memory of the GPU. Since the global memory is not cached, it is essential to follow the right access pattern to get maximum memory bandwidth. The data type must be such that sizeof (type) is equal to 4, 8, or 16 and the variables of type must be aligned to size of bytes [3].

International Journal of Mechatronics, Electrical and Computer Technology

Vol. 3(8), Jul. 2013, pp 162 - 182, **ISSN**: 2305-0543
Available online at: http://www.aeuso.org

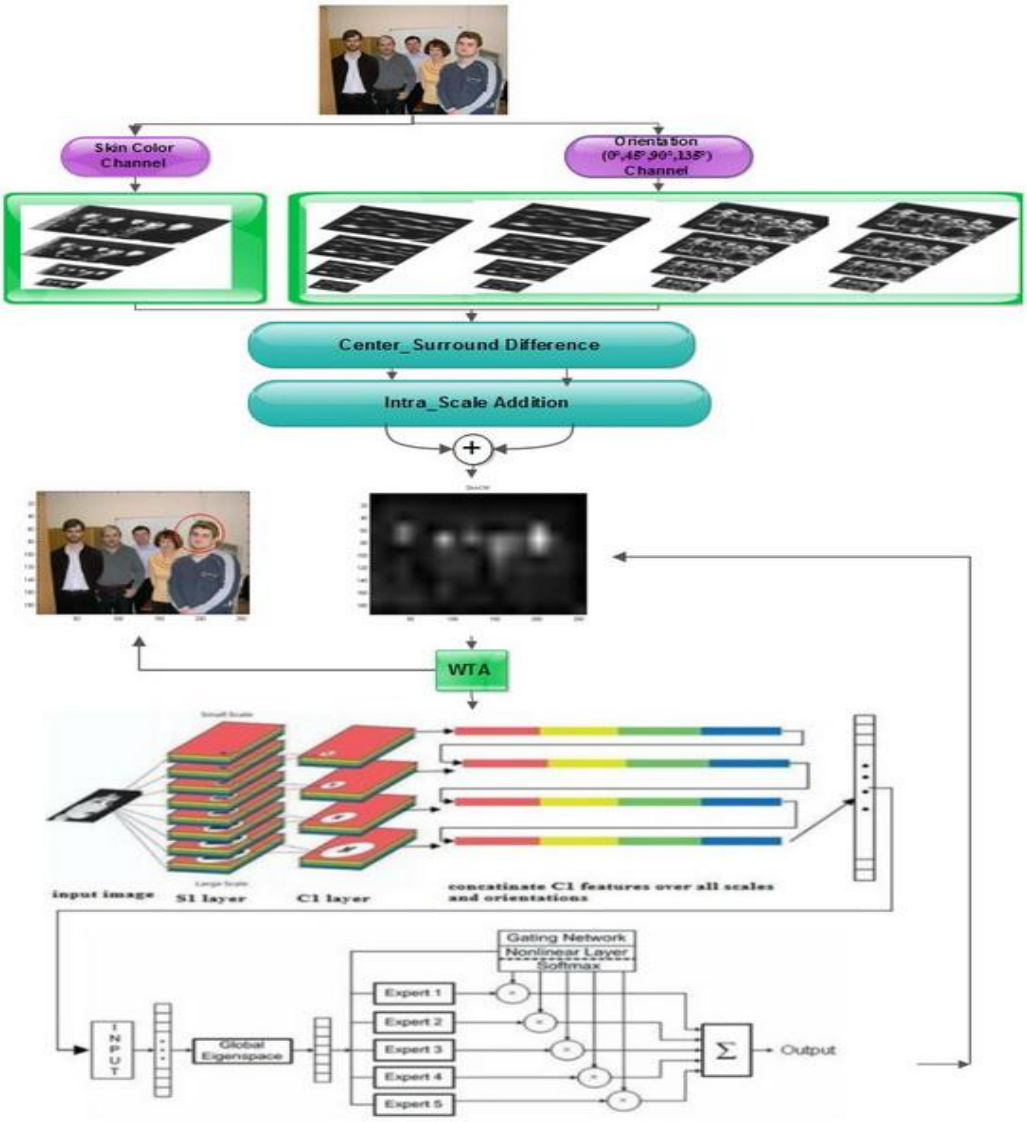


Fig. 2: System overview. The proposed system consists of three biological main layers: bottom-up visual attention based saliency map for selecting salient face regions, HMAX C1 layer to extract features from saliency regions and mixture of expert neural network for classifying face/nonface images.

International Journal of Mechatronics, Electrical and Computer Technology

Vol. 3(8), Jul. 2013, pp 162 - 182, **ISSN**: 2305-0543
Available online at: http://www.aeuso.org

C. Gaussian dyadic pyramid computation

In [14] a 6×6 separable Gaussian kernel [1 5 10 10 5 1]/32 is used for the image size reduction. A two-dimensional convolution contains 6×6 = 36 multiplications for each output pixel, while a convolution with separable filters only requires 6 + 6 = 12 multiplications for each output pixel. Therefore, we separate the Gaussian dyadic pyramid computation into two convolutions: one convolution in the horizontal direction to reduce the horizontal dimension, and one convolution in the vertical direction, respectively. For example, for the subsampling from an input image at 640×480 into an output image at 320×480, each block has 320 threads, while each grid has 480 blocks. Each thread computes only one pixel in the output image.

Attention must be paid to the threads synchronization, because the convolution in the thread n is dependent on the pixels loaded by thread n-1 and n+1. To deal with the convolution on the image border, we use [10 10 5 1]/26 on the left border and [1 5 10 10]/26 on the right border. After that, a following subsampling in the vertical direction can be similarly solved. The input image at 640× 480 is subsampled into 8 other scales: 320×240 (scale σ = 1), 160×120 (σ = 2), …, 2×1 (σ = 8).

D. C-maps computation

In the saliency map computation the RG- and BY-maps are required. According to [14], C-maps are computed as follows:

$$M_{RG}(\sigma) = \frac{r-g}{\max(r,g,b)}$$

$$M_{BY}(\sigma) = \frac{b-\min(r,g)}{\max(r,g,b)} \qquad (4)$$

International Journal of Mechatronics, Electrical and Computer Technology

Vol. 3(8), Jul. 2013, pp 162 - 182, ISSN: 2305-0543
Available online at: http://www.aeuso.org

r, g, b are the pixel values in the red, green and blue channels. The regions in which max (r, g, b) < 0.1 are set to 0. To make the computation more efficient, we integrate the computation of C-maps into the Gaussian filter convolution in the vertical direction, because the image data are already in the shared memory after the convolution. Thus, we can spare the time for loading the data from the global memory.

E. O-maps computation

Gabor filter: To compute the O-maps in different scales, a Gabor filter truncated to 19×19 pixels is used [14], which was described at section 2.2. Since a convolution with the 19×19 Gabor filter is too costly, we use FFT and IFFT to accelerate this process significantly. The Gabor filter to be convoluted images should be converted into the frequency domain using FFT at first, and multiplied with each other. Then, the result is converted from the frequency domain into the space domain using IFFT.

Using CUFFT library [15] we compute from the original Gabor filter eight FFTs with four different orientations and two different forms (sine and cosine). Due to the fact that the input image (640×480) and the subsampled image at scale 1 (320× 240) are not used for the following saliency map computation, 7 × 4 × 2 = 56 convolutions for the O-maps are needed (7 scales, 4 orientations and 2 forms). We assembly the images in 7 scales together into a Cuda-image (see Fig. 3) such that just 1 FFT and 8 IFFTs instead of 7 FFT and 56 IFFTs are computed. For an input image at 640×480, an image with 256× 256 is big enough to have all the images into itself [17].

International Journal of Mechatronics, Electrical and Computer Technology

Vol. 3(8), Jul. 2013, pp 162 - 182, **ISSN**: 2305-0543
Available online at: http://www.aeuso.org

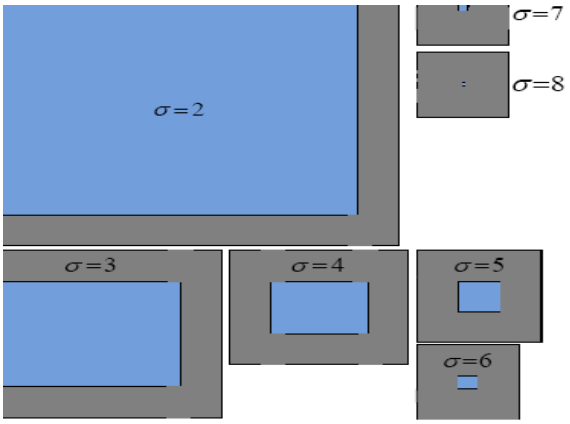


Fig. 3: The Cuda-image of 256×256 pixels. Contains seven intensity maps (scales s = 2,…,8). The blue 7 depicts the intensity maps and space is left between the maps to be applied mode texture clamp to border [17] which is shown in gray.

F. Center-surround differences

After the steps above, 9 I-maps, 18 C-maps and 36 Omaps are generated. It is followed by the center-surround differences and cross-scale combinations. In these two steps, images at different scales are subtracted and combined. In the center-surround differences, 6 feature maps in the intensity channel, 12 feature maps in the color channel and 24 feature maps in the orientation channel are computed as follows [13]:

$$RG(c,s) = N(|\, RG(c) \odot RG(s) \,|) \quad (5)$$

$$BY(c,s) = N(|\, BY(c) \odot BY(s) \,|)$$

$$O(c,s,\theta) = N(|\, O(c) \odot O(s) \,|) \quad (6)$$

With c referring to the fine scale and s indicating the coarse scale: c = {2, 3, 4}; δ = {3, 4}; s= c+δ. θ is the orientation of the Gabor filter. θ is the subtraction between two images at

International Journal of Mechatronics, Electrical and Computer Technology

Vol. 3(8), Jul. 2013, pp 162 - 182, **ISSN**: 2305-0543
Available online at: http://www.aeuso.org

different scales c and s. To execute this subtraction, the images should be enlarged or reduced into the same size and then a point-by-point subtraction is accomplished.

G. Combination into the saliency map

In the following cross-scale combinations no image rescaling is needed. It is only a question of point-by-point integration of the feature maps into conspicuity maps I, C and O as follows [2]:

$$C = \oplus_{c=2}^{4} \oplus_{s=c+3}^{c+4} (N(RG(c,s)) + N(BY(c,s))) \tag{7}$$

$$O = \sum_{\theta \in \{0^0, 45^0, 90^0, 135^0\}} \oplus_{c=2}^{4} \oplus_{s=c+3}^{c+4} (N(O(c,s))) \tag{8}$$

The saliency map is a linear combination of the normalized conspicuity maps. With the two maps conspicuity obtain the saliency map final:

$$S = \frac{C+O}{2} \tag{9}$$

H. S1 and C1 Layer

After creating saliency image map, we use WTA for weighting the salient region. Then extract the salient face region in the CPU and give as an input image to the HMAX C1 model (section 2.2). The first layer (Image Layer) uses the source image to create an image pyramid of the salient region. We create one thread for every output pixel and each thread then reads a small area to interpolate over, effectively exploiting the caching hardware. Creating one thread per output pixel is a principle which is directly reused in the S1 and C1 layers. It actually corresponds to a neuron which computes its output by weighting inputs at several dendrites.

International Journal of Mechatronics, Electrical and Computer Technology

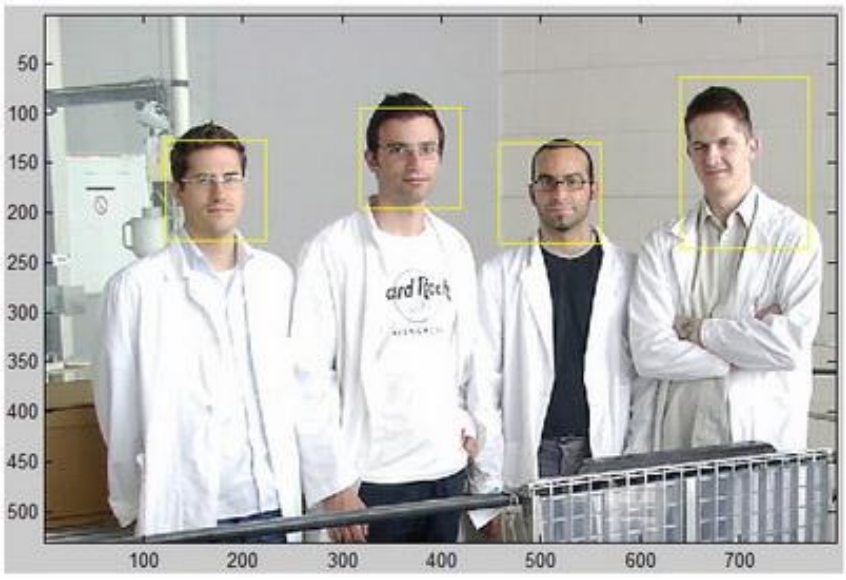
Vol. 3(8), Jul. 2013, pp 162 - 182, **ISSN**: 2305-0543
Available online at: http://www.aeuso.org

I. Classify Input Image

Finally, we apply mixture of expert neural network for classifying multi-view faces from nonface input images. This classifier was described at [9].

5. Experimental Results

We tested our multi-GPU implementation using NIVDIA GeForce 8800 (GTX) graphics cards. The computers are equipped with different CPUs and 64-bit linux systems. The computational time is the average processing time of 100 input images at a resolution of 640×480 pixels. For training we use 14000 face images with 40×40 size and 2000 nonface images with the same size. Figure 4 shows the results of our proposed model for face detection application. The most costly step is the initialization which has a computational time of 328 ms. The saliency map computation takes only about 10.6 ms with a frame rate of 94.3 fps, respectively. In the GFLOPS performance estimation, only the floating-point operations are considered. In this experiment with 100 images of 1000 faces we get 97% performance with 22 false detection rates.



International Journal of Mechatronics, Electrical and Computer Technology

Vol. 3(8), Jul. 2013, pp 162 - 182, **ISSN**: 2305-0543
Available online at: http://www.aeuso.org

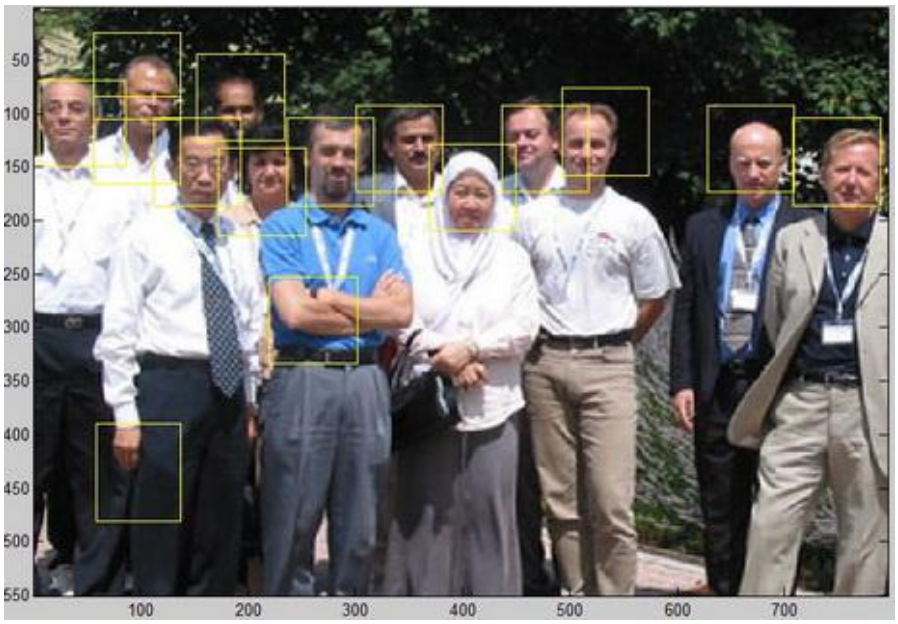


Fig. 4: The results of face detection

Table 1 shows the detailed processing time protocol. The most costly step is the initialization which has a computational time of 328ms. The total face detection task by applying this model is computed as 57ms.As we have already mentioned we benefit from using the GPU to calculate our saliency map. Both the parallel nature of this processor, and the built in image processing and texture filtering operations make are our model run faster than it would on a conventional CPU. To asses this performance gain we compared the image space portion of our model to the same algorithm with no GPU support. Figure 5 shows the time taken to process an image at a range of resolutions. The graph shows that at the maximum resolution of 768×768 which we tested our approach is approximately 70 times faster than the CPU based approach.

International Journal of Mechatronics, Electrical and Computer Technology

Vol. 3(8), Jul. 2013, pp 162 - 182, **ISSN**: 2305-0543
Available online at: http://www.aeuso.org

Table 1: Time proportion between the various stages of calculating the proposed model to detect faces in input image taken from a video camera.

Our Proposed Model	Time (ms)
Obtaining copy of the image from CPU to GPU (Initialization)	14.74
Creation of the pyramids	2.27
Gabor Filter	7.5
Image rescaling	1.48
Center-Surround differences	0.54
Convolution filters with differences in Gauss	3.32
Maximum overall	3.85
Additions to long scales	2.9
Standardization, obtaining the final map and copy to CPU	0.4
Obtaining salient regions	9.5
Resizing	3.0
Gabor Filter	7.5
Maximum over scales and neighbors	2.5
Total	**57**

International Journal of Mechatronics, Electrical and Computer Technology

Vol. 3(8), Jul. 2013, pp 162 - 182, **ISSN**: 2305-0543
Available online at: http://www.aeuso.org

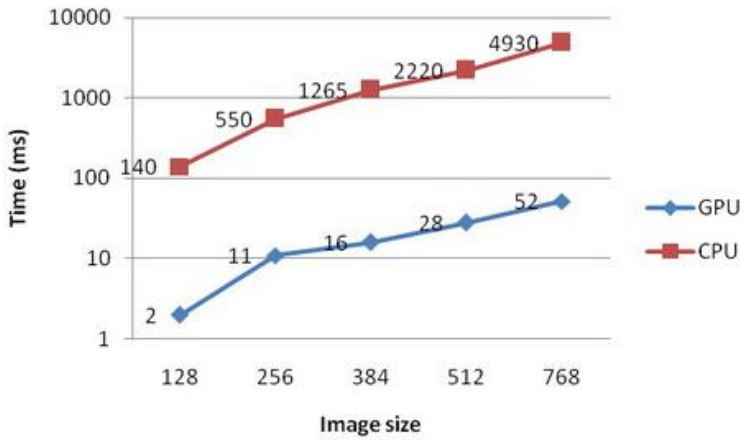


Fig. 5: Linear image filtering: NVIDIA 8800GT GPU vs. 2.8 GHz CPU.

To evaluate the performance gains of computing our proposed model, we compare the time of implementation in CUDA, a NVIDIA 9800 GTX +, with an implementation in C / C++ using the OpenCV library [20] [21] and Matlab [22], as shown in Figure 6. The implementation has a CUDA speedup of 30.6 x with respect to the implementation in C /C+ + using the OpenCV library.

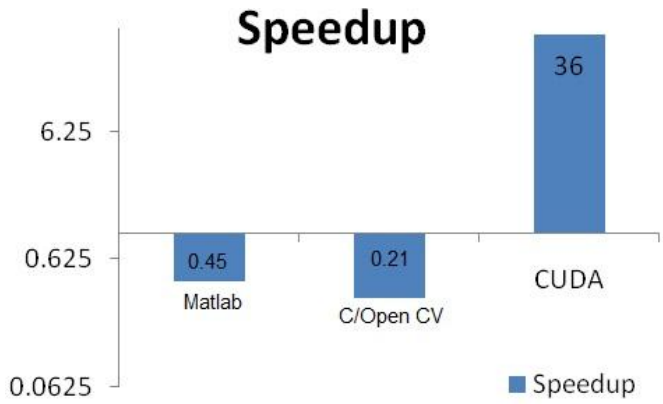


Fig. 6: Speedup implementations performed to model. The CUDA implementation has a speedup of 30.6 x in relation to the implementation in C / C + + using the OpenCV library.

International Journal of Mechatronics, Electrical and Computer Technology

Vol. 3(8), Jul. 2013, pp 162 - 182, **ISSN**: 2305-0543
Available online at: http://www.aeuso.org

Using our implementation the prerequisite of the application of bottom-up attention on mobile robots and vehicles is fulfilled. A human-like bottom-up visual attention system for our mobile robot ACE [18] is established [19].

Conclusions

Biologically motivated face detection models was implemented on massively parallel platforms such as Multi-GPU. In this paper a novel method for face detection application which implement on multi-GPU and using CUDA is presented. In the proposed model we used bottom-up attention based on the saliency map model for selecting salient regions, HMAX C1 model for feature extraction and mixture of expert neural network classifier for recognition face nonface salient images. The computationally expensive saliency map computation is elaborately implemented using CUDA technology. Different steps are properly and sophisticatedly implemented in the global memory, the shared memory and the texture memory with respect to different computation characteristics. Moreover, a multi-threaded structure accelerates the saliency map generation and brings the computational speedup into 4 NVIDIA GeForce 8800 (GTX) graphics cards.

The parallel computation power of the graphics cards is fully utilized. A frequency of 313 fps on input images at 640×480 pixels is achieved, which is about 8.5 times faster than the standard implementation on CPUs. Without any CPU load, a high accuracy and a good scalability are achieved. After saliency map, we used HMAX C1 model to extract features from salient regions and classify using mixture of expert neural network to detect faces. According to the ability to implement hierarchical model of the visual cortex of the brain, in this, we implement HMAX C1 model for parallel implementation on modern graphic processor. In experimental result show that unlike general-purpose processor, there is a nonlinear relation image size and running time of program and the running time difference between GPU and CPU at images with larger size than 512 is completely obvious.

International Journal of Mechatronics, Electrical and Computer Technology

Vol. 3(8), Jul. 2013, pp 162 - 182, **ISSN**: 2305-0543
Available online at: http://www.aeuso.org

References

[1] D. Strayer, L., F. A. Drews and W. A. Johnston, "Cell phone induced failures of visual attention during simulated driving". Journal of Experimental Psychology: Applied, 9, 23-32. 2003

[2] L. Itti, C. Koch,and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis". IEEE Transactions on Pattern Analysis and Machine Intelligence, 20(11), 1254-1259. 1998.

[3] CUDA Programming Guide Version 1.1. NVIDIA, 2007.

[4] H. Yee, S. Pattanaik, and D. P. Greenberg, "Spatiotemporal sensitivity and visual attention for efficient rendering of dynamic environments". In ACM Transactions on Graphics. ACM Press, 39.65, 2001.

[5] A. Ude, V. Wyart, L. H. Lin and G. Cheng. "Distributed Visual Attention on a Humanoid Robot". Proc. of HUMANOIDS 2005.

[6] P. Longhust, K. Debattista and A. Chalmers. "A GPU based Saliency Map for High-Fidelity Selective Rendering". AFRIGRAPH 2006, Cape Town, South Africa, 2006.

[7] R. J. Peters and L. Itti. "Applying computational tools to predict gaze direction in interactive visual environments". ACM Transactions on Applied Perception, 5(2), Article 8, 2008.

[8] T. Serre, L. Wolf, T. Poggio, "Object recognition with features inspired by visual cortex", in CVPR (2005), pp. 994–1000.

[9] Z. Farhoudi, S. Faridi, R. Ebrahimpour, S. Setayeshi, "Teacher-Directed Learning in View-Independent Face Detection with Mixture of Experts Using Cortex-Like Features". Proc. of AJIIPS, Vol 12, 2010.

[10] M. Riesenhuber, T. Poggio, "Hierarchical Models of Object Recognition in Cortex", Nature Neuroscience 2(11), 1019–1025, 1999.

[11] T. Serre, M. Kouh, C. Cadieu, U. Knoblich, G. Kreiman, T. Poggio, "A Theory of Object Recognition: Computations and Circuits in the Feedforward Path of the Ventral Stream in Primate Visual Cortex", AI Memo 2005.

[12] T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber, T. Poggio, "Robust Object Recognition with Cortex-like Mechanisms", IEEE Transactions on Pattern Analysis and Machine Intelligence 29(3), 411–426, 2007.

[13] L. Itti and C. Koch. "A comparison of feature combination strategies for saliency-based visual attention systems", In SPIE human vision and electronic imaging IV (HVEI'99), San Jose, CA pp 473-482. 1999.

[14] D. Walther and C. Koch, "Modeling attention to salient proto-objects", Science Direct. Neural Networks 19 (2006) 1395-1407.

[15] CUDA CUFFT Library. NVIDIA, 2007.

[16] J. Mutch, D. Lowe, "Multiclass object recognition with sparse, localized features", in CVPR (1), IEEE Computer Society, pp. 11–18, 2006.

International Journal of Mechatronics, Electrical and Computer Technology

Vol. 3(8), Jul. 2013, pp 162 - 182, **ISSN**: 2305-0543
Available online at: http://www.aeuso.org

[17] V. Podlozhnyuk. FFT-based 2D convolution. NVIDIA, 2007.

[18] T. Xu, K. K¨uhnlenz and M. Buss, "Looking at the Surprise: Bottom-Up Attentional Control of an Active Camera System", Proc. of ICARCV 2008.

[19] G. Lidoris, K. Klasing, A. Bauer, T. Xu, K. K¨uhnlenz, D. Wollherr and M. Buss, "The Autonomous City Explorer Project: Aims and System Overview", Proc. of IROS 2007.

[20] OpenCV Reference Manual, 2010. Availabel from: https://picoforge.int-evry.fr/projects/svn/gpucv/opencvdoc/2.1/opencv.pdf,acedidoemJulhode201155 BIBLIOGRAFIA

[21] Open CV. Intel Open Source Computer Vision Library Manual, 2011.
http://sourceforge.net/projects/opencvlibrary.

[22] Math Works. Image Processing Toolbox Manual, 2011. Available from: http://www.mathworks.com.