COMMUNICATION-AWARE FACE DETECTION USING NOC ARCHITECTURE

Hung-Chih Lai, Radu Marculescu, Marios Savvides, and Tsuhan Chen

Department of Electrical and Computer Engineering
Carnegie Mellon University, Pittsburgh, PA 15213, USA
{hlai, radum, marioss, tsuhan}@cmu.edu

Abstract. Face detection is an essential first step towards many advanced computer vision, biometrics recognition and multimedia applications, such as face tracking, face recognition, and video surveillance. In this paper, we proposed an FPGA hardware design with NoC (Network-on-Chip) architecture based on an AdaBoost face detection algorithm. The AdaBoost-based method is the state-of-the-art face detection algorithm in terms of speed and detection rates and the NoC provides high communication capability architecture. This design is verified on a Xilinx Virtex-II Pro FPGA platform. Simulation results show the improvement in speed 40 frames per second compared to software implementation. The NoC architecture provides scalability so that our proposed face detection method can be sped up by adding multiple classifier modules.

Keywords: Face detection, Hardware Architecture, Network-on-Chip.

1 Introduction

Face detection is the process of finding all possible faces in a given image or a video sequence. More precisely, face detection has to determine the locations and sizes of all possible human faces. It is a more complex case than face localization in which the number of faces is already known. On the other hand, face detection is the essential first step towards many advanced computer vision, biometrics recognition and multimedia applications, such as face tracking, face recognition, and video surveillance.

Due to scale, rotation, pose and illumination variation, face detection involves many research challenges. How to detect different scales of faces, how to be robust to illumination variation, how to achieve high detection rate with low false detection rates are only few of all issues a face detection algorithm needs to consider.

Face detection techniques have been researched for years and much progress has been proposed in literature. Most of the face detection methods focus on detecting frontal faces with good lighting conditions. According to Yang's survey [6], these methods can be categorized into four types: knowledge-based, feature invariant, template matching and appearance-based.

Knowledge-based methods use human-coded rules to model facial features, such as two symmetric eyes, a nose in the middle and a mouth underneath the nose [12].

Feature invariant methods try to find facial features which are invariant to pose, lighting condition or rotation [10]. Skin colors, edges and shapes fall into this category. Template matching methods calculate the correlation between a test image and pre-selected facial templates [13]. The last category, appearance-based, adopts machine learning techniques to extract discriminative features from a pre-labeled training set. The Eigenface method [7] is the most fundamental method in this category. Recently proposed face detection algorithms such as support vector machines [11], neural networks [9], statistical classifiers [2,8] and AdaBoost-based face detection [1] also belong to this class.

Recently, appearance-based methods have achieved good results in terms of performance and speed. However, these methods are software-centric algorithms which do not take the hardware aspect into consideration. For instance, the state-of-the-art method in terms of speed and performance, AdaBoost-based face detection, requires inefficient memory space to store integral images (integral image will be discussed later). Besides, high demand of random data access of memory is necessary. For a 320x240 image, AdaBoost-based face detection needs 25 bits for one pixel, (a total of 1.92 Mbits), to store the whole integral image. Moreover, even the AdaBoost-based method uses a cascade architecture to reduce colloquial unnecessary operations; for an image of the same size, this method still requires accessing 25 bits of data over 21 million times for only the original scale and the first three classifier stages. (Assume that 41 features are in the first three stages, and each feature requires 8 points accessing)

In this paper, we adopt the AdaBoost-based face detection with NoC (Network-on-a-chip) architecture to satisfy the requirement of high data access rate. The NoC architecture [16, 17] is proposed with parallel and scalable links for communication within large VLSI implementation in a silicon chip. NoCs adopt the routing strategy from computer network to allow components to communicate data from source nodes through routers to destination nodes. There are many choices of mapping and routing methods which will not be discussed here. In this paper, we manually map the task and use the XY routing method.

2 Previous work

2.1 Theory of AdaBoost Technique

The AdaBoost-based face detection method uses a set of simple rectangular features to detect faces. Three kinds of rectangles are used, a two-rectangle feature, a three-rectangle feature and a four-rectangle feature, as shown in Fig. 1.

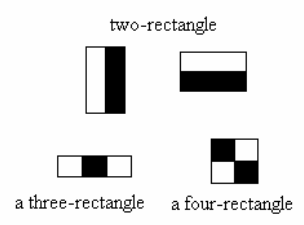

Fig. 1. Example of simple features

For a given feature, the sum of pixels in the white rectangles is subtracted from the sum of pixels in the black rectangles. Based on these simple features, the AdaBoost-based face detection method proposed three significant contributions which make face detection fast. The block diagram is shown in Fig. 2. First, by utilizing the integral image representation, the simple features can be rapidly computed in linear time. Second, a machine learning method, based on AdaBoost, trains an over-completed feature set and obtains a reduced set of critical features. Third, a cascade architecture of increasingly complex classifiers was proposed to quickly eliminate background windows and focus on more face-like windows.

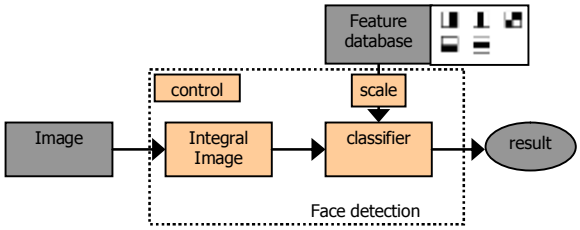

Fig. 2. Block diagram of AdaBoost-based face detection

Integral Image:

An image is first converted into its corresponding integral image. The integral image at location (x, y) is the sum of all pixel values above and to the left of (x, y), inclusive [1].

$$ii(x,y)=\sum_{m=0}^{x}\sum_{n=0}^{y}i(m,n) \tag{1}$$

where ii is the integral image and i is the original image. Any rectangular sum can be computed efficiently by using integral images.

$$\begin{aligned} &sum(\text{one rectangle}) \\ &= ii(a)+ii(d)-ii(b)-ii(c) \end{aligned} \tag{2}$$

Using the integral image, any rectangular sum can be computed in four-point access. Therefore, no matter what the size or the location of feature is, a two-rectangle

feature, three-rectangle feature and four-rectangle feature can be computed in six, eight and nine data references, respectively.

AdaBoost Learning:

AdaBoost is a machine learning technique that allows computers to learn based on known dataset. Recall that there are over 180,000 features in each sub-window. It is infeasible to compute the completed set of features, even if we can compute them very efficiently. According to the hypothesis that small number of features can be associated to form an effective classifier [1], the challenge now turns to be how to find these features. To achieve this goal, the AdaBoost algorithm is designed, for each iteration, to select the single rectangular feature among the 180,000 features available that best separates the positive samples from the negative examples, and determines the optimal threshold for this feature. A weak classifier h_j (x) thus consists of a feature f, a threshold Θ_j and a parity p_j indicating the direction of the inequality sign:

$$h_j(x) = \begin{cases} 1 & if \quad p_j f_j(x) < p_j \theta_j \\ 0 & otherwise \end{cases} \tag{3}$$

The final strong classifier is shown in Eq. (4). Therefore, given a test sub-window x, the strong classifier classifies the sub-window as a face if the output is one.

$$h(x) = \begin{cases} 1, & if \quad \sum_{t=1}^{T} \alpha_t h_t(x) > \theta \\ 0, & otherwise \end{cases} \tag{4}$$

Cascade Architecture:

Along with the integral image representation, the cascade architecture is the important property which makes the detection fast. The idea of the cascade architecture is that simple classifiers, with fewer features (which require a shorter computational time) can reject many negative sub-windows while detecting almost all positive objects. The classifiers with more features (and more computational effort) then evaluate only the sub-windows that have passed the simple classifiers. Fig. 3 shows the cascade architecture consisting of n stages of classifiers. Generally, the later stages contain more features than the early ones.

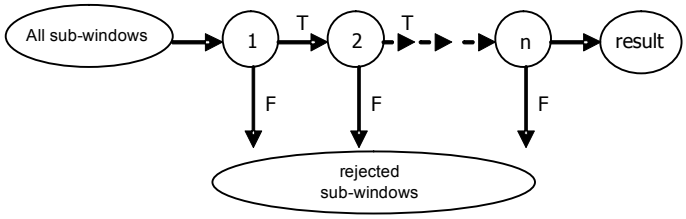

Fig. 3. Cascade architecture of classifier

2.2 Hardware implementation of AdaBoost-based face detection

Several studies have been proposed in hardware implementation for face detection. In one neural network implementation [14], high-frame-rate face detection is achieved. However, in this specific example, only 965 windows were evaluated for a 300x300 pixel image. In another hardware implementation, a good detection rate is achieved using a statistical classifier [13]. However, due to the large memory requirement, the hardware implementation needed to constraint the extracted features to a reduced model size

There are fewer hardware studies involving AdaBoost-based face detection. However, most of the proposed solutions do not list the complete parameters that affect the detection speed. For example, the implementation using System Generator® and ISE® [3] does not mention the cost of accessing data. The other implementation on handheld camera [4] assumes the bandwidth and latency of bus and interface are guaranteed. Another parallel architecture [5] is proposed taking the advantage of parallel processing and can achieve 52 fps running at 500 MHz clock cycles. However, the size of the test images used to achieve 52 fps was not mentioned.

3 Architecture

Our NoC-based face detection framework consists of four parts: the image integrator, memory, the feature database, and the feature classifier. Each component is attached to an on-chip router to form the on-chip network, which is shown in Fig. 4. The main reason for choosing NoC as our architecture is that NoC is a scalable architecture which can allow us to work with higher resolution images and reach higher detection speeds by attaching several duplicated modules.

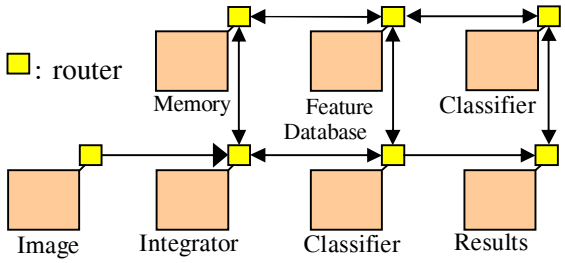

Fig. 4. NoC-based face detection framework

The Image Integrator:

The target image resolution in this paper is 320 by 240 pixels. Therefore, we utilize one 320 length FIFO with 32-bit width to provide the integral data of the previous line. The accumulator performs the horizontal sum and stores it in the FIFO while the adder performs the vertical sum of the output of the accumulator and the FIFO. Fig. 5 shows the design of the integral image using System Generator®. The synthesis result

of this implementation achieves a maximum frequency of 163.074 MHz and a total gate count of 8590 gates.

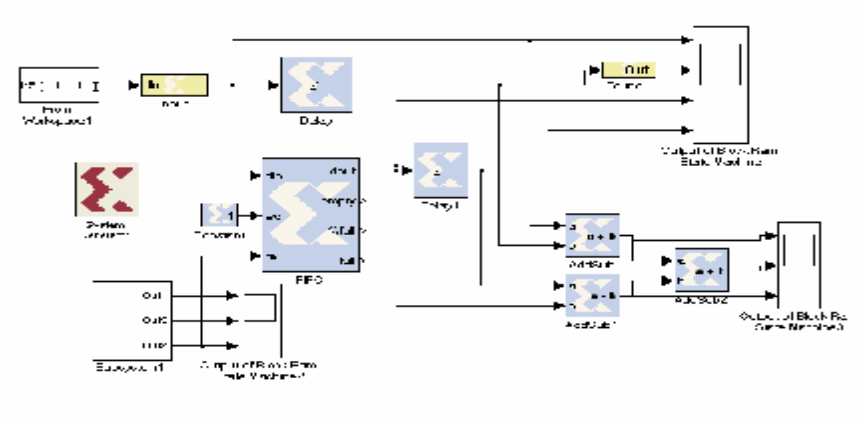

Fig. 5. Integral image design in System Generator

Memory:

The size of integral images is large, and the minimum number of bits needed to store a pixel of an integral image varies greatly from the top-left pixel to the bottom-right pixel. As a result, we plan to use an external memory for integral image storage. The external memory will be accessed by a memory controller which is connected to the on-chip network through an on-chip router.

The Feature Database:

This database includes a large set of features that characterize faces. Sub-windows in the original image that strongly respond to features in the database will be classified as faces. On the other hand, since faces in images may have different sizes, we also need to scale the features in order to identify various scales of faces.

The Feature Classifier:

Once we locate a target sub-window in the original image, the feature classifier will search for the best feature in this sub-window by calculating the difference between a feature and an equal-size searching block, which can be done by Equation (2) in constant time. If the difference is less than a specified threshold, we say that the pattern matches that feature. If a target pattern matches all features in the feature database, this pattern will be marked as a face. In order to exploit the advantage of the on-chip network, we can duplicate the feature classifier and each classifier searches for a sub-set of features in parallel.

We can also apply the cascade classification to split this procedure into several stages, with the fewer but more important features for the first stage and the other features for the last stage. Furthermore, several classifiers can be employed in the NoC architecture to achieve parallelism by processing several target patterns or detecting several features simultaneously.

4 ON-CHIP ROUTER ARCHITECTURE

We implemented the on-chip router by using System Generator®. However, we realize that using System Generator for the on-chip router is not that efficient because it consists of many control signals for the arbiter, crossbar switch, and channel controller.

The architectural overview of our on-chip router is shown in Fig. 6. Generally, each router has four pairs of input/output Routing Channels and one pair of input/output Local Channels, and each channel has its own Buffer. The Channel Controller extracts the source and destination addresses from an incoming packet, determines which output channel the packet should be delivered to, and sends the connection request to the Arbiter. The arbiter can decide which input channel has the right to send the packet through the Crossbar Switch, and then establish a link between the input channel and its corresponding output channel. In this NoC-based application, we will use deterministic XY routing and specify buffer length for each channel according to traffic congestion.

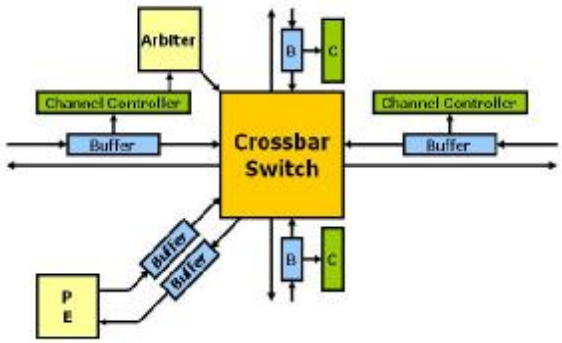

Fig. 6. Architectural overview of on-chip router

Fig. 7 shows our preliminary on-chip router design in the Xilinx System Generator. We used FIFOs as input and output buffers and implemented the crossbar switch by a set of MUXs. Channel controllers are accomplished by several control signals and the arbiter is implemented by writing a Matlab code in MCode block.

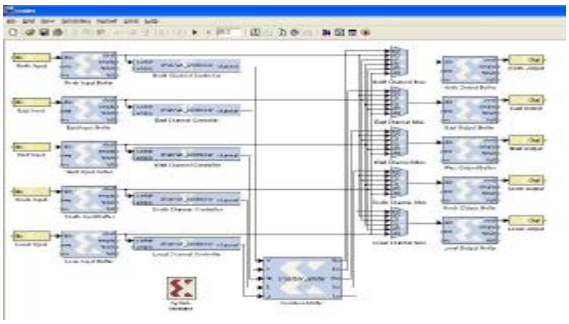

Fig. 7. On-chip router design in System Generator

5 EXPERIMENTAL RESULTS

We modify the on-chip router and attach the router to the image integrator. In addition, a ROM which stores the original image and an internal memory which stores the integral image are also connected to on-chip network. More specifically, we can now read the original image from the ROM, calculate the corresponding integral image, and then store the integral image into memory. All these operations are achieved through on-chip network. These integrated components can be synthesized by Xilinx ISE and the synthesis result achieve a maximum frequency of 152.996 MHz.

Our proposed work is verified on a Xilinx Virtex-II Pro XC2VP30 FPGA on Xilinx® VirtexII Pro board, as shown in Fig. 8. This FPGA contains 200K equivalent logic gates and 2.44Mb of on-chip memory. Using both Modelsim® simulation software and ISE® synthesize tool, a bitstream file is generated for FPGA verification. For more information about Xilinx FPGA, please refer to [15].

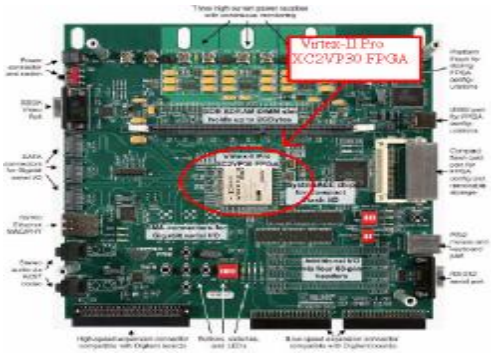

Fig. 8. Xilinx Virtex-II Pro system board

Input: Images with face(s), e.g. Fig. 9(a).
Output: Images with face(s) marked, e.g. Fig. 9(b)
Description: Our objective is to locate and mark where the faces are with acceptable accuracy. Right now, we take the internal memory into consideration and implement 44 features and we believe that a detection rate of over 70% could be achieved in normal images. Input and output images will be read and written through the Compact Flash (CF) interface of FPGA board.

(a) (b)

Fig. 9. (a). An input image. (b). An output image

6. CONCLUSION

In this paper, we proposed an FPGA hardware implementation with NoC (Network-on-Chip) architecture based on the AdaBoost face detection algorithm. The AdaBoost method is the state-of-the-art face detection algorithm and the NoC provides high communication capability architecture. This design is verified on a Xilinx Virtex-II Pro FPGA platform. Our results achieve rates of 40 images per second for 320 by 240 resolution images which outperforms than current software implementation.

The evaluation of the performance in terms of detection rate, the optimization of logical circuit will be done in future work.

ACKNOWLEDGMENT

This research is supported by Carnegie Mellon CyLab. The authors would also like to thank Xilinx, Inc., for providing FPGA hardware and software tools by Xilinx University Program.

References

1 P. Viola and M J. Jones, “Rapid object detection using a boosted cascade of simple features,” in Proc. CVPR 2001

2 H. Schneiderman and T. Kanade, "Object detection using the statistic of parts," *Int. J. Computer Vision* 2004
3 Y. Wei, X. Bing, and C. Chareonsak, "FPGA implementation of adaboost algorithm for detection of face biometrics," In *Int. Workshop on Biomedical Circuits & Systems* 2004
4 Ming Yang, Ying Wu, Crenshaw, J.; Augustine, B.; Mareachen, R.;" Face detection for automatic exposure control in handheld camera" in *ICVS* 2006.
5 Theocharides, T.; Vijaykrishnan, N. Irwin, M.J., "A parallel architecture for hardware face detection," *Emerging VLSI Technologies and Architectures*, 2006
6 M.-H. Yang, D. Kriegman and N. Ahuja, "Detecting Faces in Images: A Survey," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 24, no. 1, pp. 34-58, Jan. 2001.
7 M. Turk and A. Pentland, "Face recognition using eigenfaces," *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 586–591.
8 H. Schneiderman and T. Kanade, "A Statistical Method for 3D Object Detection Applied to Faces and Cars," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 746-751, June 2000.
9 H.A. Rowley, S. Baluja and T. Kanade, "Neural Network-Based Face Detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 20, no. 1, pp. 23-38, Jan. 1998.
10 K.K. Sung and T. Poggio, "Example-Based Learning for View-Based Human Face Detection," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 20, no. 1, pp. 39-51, Jan. 1998.
11 Kepenekci, B.; Akar, G.B, "Face classification with support vector machine," Proc. *IEEE Conf. Signal Processing and Communications Applications*, pp. 583-586, April 2004.
12 G. Yang and T.S. Huang, "Human Face Detection in Complex Background," Pattern Recognition, Vol, 27, no.9, pp. 712-735, 1997
13 A. Lanitis, C.J. Taylor and T.F. Cootes, "An automatic face identification System Using Flexible Appearance Models," Image and Vision Computing, Vol. 13, no. 5, pp. 393-401, 1995
14 Theocharis G. Theocharides,G. M. Link, V.Narayanan, M. J. Irwin, and W. Wolf, "Embedded Hardware Face Detection," in Procedings, VLSI Design 2004, IEEE, 2004.
15 http://www.xilinx.com/products/silicon_solutions/fpgas/virtex/virtex_ii_pro_fpgas/index.htm
16 A. Jantsch and H. Tenhunen (Eds.), "Networks on Chip," Kluwer Academic Publishers, 2003.
17 J. Hu, R. Marculescu, "Energy-Aware Communication and Task Scheduling for Network-on-Chip Architectures under Real-Time Constraints," in Proc. Design, Automation and Test in Europe Conf., Paris, France, Feb. 2004.