Hardware Acceleration of Multi-view Face Detection

Junguk Cho, Bridget Benson, and Ryan Kastner
Department of Computer and Science and Engineering
University of California, San Diego
La Jolla, California, USA
{jucho, b1benson, kastner}@cs.ucsd.edu

***Abstract*—This paper presents a parallelized architecture for hardware acceleration of multi-view face detection. In our architecture, the multi-view face detection system generates rotated image windows and their integral image windows for each classifier which perform parallel classification operations to detect non-upright (rotated) and non-frontal (profile) faces in the images. We use the training data from OpenCV to detect the frontal and profile faces based on the Viola and Jones algorithm. The proposed architecture for multi-view face detection has been designed using Verilog HDL and implemented in a Xilinx Virtex-5 FPGA. Its performance has been measured and compared with a Jones' and Viola's software implementation of multi-view face detection.**

Keywords- acceleration, classifier, face detection, FPGA, multi-view face, Verilog HDL

I. INTRODUCTION

Face detection is the act of indentifying faces of interest in images regardless of size, position, and circumstance. A successful algorithm will find the locations and sizes of all faces in the image stream that belong to a given class with no or few "false positives". Potential face detection applications include monitoring and surveillance, human computer interfaces, smart rooms, intelligent robots, and biomedical image analysis. Face detection proposed by Viola and Jones is the first approach for real-time face detection [1]. This approach utilizes the AdaBoost algorithm [2], which identifies a sequence of rectangle features that indicate the presence of a face. The Viola and Jones algorithm is most often used for face detection, e.g., in the OpenCV library [3][4], however is applicable in other domains. This algorithm requires considerable computational power due to the sheer number of rectangle features that must be identified to detect a face. One face is comprised of a substantial amount of features, which are typically computed over a window of 24×24 pixels. To reduce computation, the detection is performed in stages so that windows in an image that do not contain something that looks similar to a face do not require computation of all features. There are many proposed approaches for face detection in a wide variety of images. While they can successfully detect frontal upright faces, many natural images include rotated or profile faces that are not reliably detected in the real world. There are only a small number of techniques that address non-frontal or non-upright face detection [5-7]. Non-upright face detection, proposed by Rowley et al., [5] is the first reliable approach. They use two neural network classifiers. The first estimates the pose of a face in the image window. The second performs conventional face detection. Faces are detected in three steps: the pose of the face for each image window is first estimated; the estimated pose is then used to de-rotate the image window; the window is then evaluated by the second classifier. Schneiderman et al. [6] develop the algorithm that can reliably detect human faces with out-of-plane rotation. They develop separate detectors that are each specialized to a specific orientation of the face. They have one detector specialized to right profile faces and one that is specialized to frontal faces. To detect left profile faces, they apply the right profile detector to flipped images. Jones and Viola [7] extend this framework [1]. They handle the detection of non-upright faces using a two stage approach. In the first stage the pose of each image window is estimated using a decision tree constructed using features like those described by Viola and Jones [1][7]. In the second stage one of N pose specific detectors is used to classify the window. These methods need a substantial amount of computation. Therefore, this constitutes a bottleneck to the application of face detection in real-time.

This paper presets a parallelized architecture for the hardware acceleration of multi-view face detection. In order to detect profile (non-frontal) and rotated (non-upright) faces, the multi-view face detection system generates rotated image windows and their integral image windows for each classifier which perform parallel classification operations to detect profile and rotated faces in the images. The main contribution of our work, described in this paper, is the design and implementation of a physically feasible hardware system to accelerate the processing speed of the operations required for real-time multi-view face detection. Therefore, this work has resulted in the development of a real-time multi-view face detection system employing an FPGA implemented system designed by Verilog HDL. Its performance has been measured and compared with an equivalent software implementation. This paper is organized as follows: In Section 2, we explain the multi-view face detection algorithm. In Section 3, we describe the hardware architecture, designed with Verilog HDL, of the multi-view face detection system using block diagrams. We also present the implementation of our real-time multi-view face detection system in an FPGA. In Section 4, we show the system's performance. We conclude in Section 5.

II. MULTI-VIEW FACE DETECTION

The Viola and Jones [1][7] face detection algorithm is used as the basis of our design. While the input image is scanned across location and scale, this algorithm utilizes pattern classification to determine the presence of a face. Viola and Jones use a boosted collection of features to classify image windows by using the AdaBoost algorithm [2]. In the Adaboost

algorithm, a set of weak binary classifiers is learned from a training set. Each classifier is a simple feature made up of rectangular sums followed by a threshold as shown in Fig. 1. The simple features are designed to easily detect an edge or a line of the face. Viola and Jones [7] define the value of a two-rectangle feature as the difference between the sums of the pixels within two rectangular regions. Viola and Jones consider the two-rectangles as having the same size and shape and being horizontally or vertically adjacent, but we consider the two-rectangles as one small rectangle (gray) on top of a rectangle twice as large (covered by the white and gray areas). The advantage of representing the rectangular regions as having different sizes becomes apparent for the three and four rectangle features as these features can now be represented by 2 or 3 rectangles respectively which reduce the number of references needed to define the feature. Computation of rectangle features can be accelerated using an intermediate image representation called the integral image [1]. The integral image at location (x, y) contains the sum of the pixels above and to the left of (x, y). Using the integral image, any rectangle feature, at any scale or location, can be evaluated in constant time [1]. In order to improve computational efficiency and also reduce false positive rate, classification is divided into a cascade of classifiers. An image window is passed from one classifier in the cascade to the next as long as each classifier classifiers the window as a face. The threshold of each classifier is set to yield a high detection rate. Each stage of the cascade consists of several classifiers. Early stages have fewer classifiers while later ones have more so that easy non-face regions are quickly discarded. The window exits the cascade if it passes all stages or fails any stage. A face is detected if a window passes all stages.

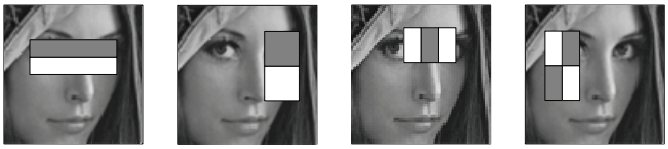

Figure 1. Example rectangle features shown relative to the enclosing detection window. The sum of the pixels which lie within the white rectnagles are subtracted from the sum of pixels which lie within the gray rectangles.

In order to detect non-upright (rotated) faces, the space of poses is divided into various classes and trained to create different detectors for each pose class by Viola and Jones [7]. In order to avoid the computational expense of having to evaluate every detector on every window in the input image, Jones and Viola use a two stage approach which first estimates the pose of the face in the window and then evaluates only the detector trained on that pose. When the pose estimator is evaluated on a non-face window, its output can be considered random. Any detector chosen to evaluate on a non-face window should return false. In Jones' and Viola's try-all-poses approach they do not use a pose estimator and try all N pose specific detectors. Jones and Viola [7] found their try-all-poses approach is more accurate than their two stage approach, but is about 5 times slower. The frontal face detector from Viola and Jones handles approximately ±15 degrees of in-plane rotation [7]. Given this, they trained 12 different detectors for frontal faces in 12 different rotation classes. Each rotation class covers 30 degrees of in-plane rotation so that together, the 12 detectors cover the full 360 degrees of possible rotations. In practice, they only needed to train 3 detectors: one for 0 degrees (which covers -15 degrees to 15 degrees of rotation), one for 30 degrees (which covers 15 degrees to 45 degrees), and one for 60 degrees (which covers 45 degrees to 75 degrees). Because the features we use can be rotated 90 degrees, any detector can also be rotated 90 degrees. So a frontal face detector trained at 0 degrees of rotation can be rotated to yield a detector for 90 degrees, 180 degrees, and 270 degrees. The same trick can be used for the 30 degree and 60 degree detectors to cover the remaining rotation classes. In order to detect non-frontal (profile) faces, the same method for rotated face detection is used. Jones and Viola [7] used the two stage approach which first classifies an image window as left or right profile and then evaluates only the detector trained on that pose. To create a left profile detector, all features of the right profile detector are simply flipped.

III. HARDWARE ARCHITECTURE / IMPLEMENTATION

We proposed a parallelized architecture of the hardware acceleration for real-time multi-view face detection. Figure 2 shows the overview of the proposed architecture for multi-view face detection. It consists of seven modules: image interface, frame grabber, image store, image scaler, Haar classifier, display, and DVI interface [8]. The image interface and DVI interface are implemented using ASIC custom chips with the FPGA board. The others are designed using Verilog HDL and implemented in an FPGA in order to perform face detection in real-time.

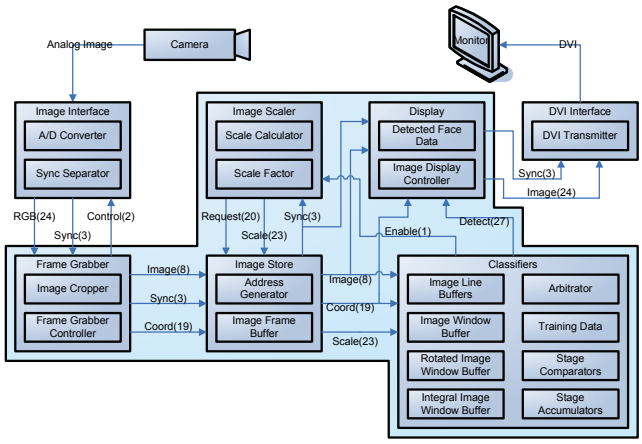

Figure 2. Block diagram of proposed multi-view face detection system.

The frame grabber module generates the control signals for the image interface, and transfers images and sync signals from the image interface module to all of the modules of the face detection system. The image store module stores the image data arriving from the frame grabber module frame by frame. The images are scaled down based on a scale factor (1.2) by the image scaler module. We use a nearest neighbor interpolation algorithm with a factor of 1.2. The scaler module for 320×240 pixel images has 14 scale factors (1.2^0~1.2^{13}), the scaler module for 640×480 pixel images has 18 scale factors (1.2^0~1.2^{17}) [8]. The image scaler module generates and transfers the address of the BRAMs containing a frame image in the image store module to request image data according to a

scale factor. The image store module transfers a pixel data to the classifier module based on the address of BRAMs required from the image scaler module. The display module stores the information of the detected faces obtained from the classifier module, and displays white squares on the faces in the image sequence. The Digital Visual Interface (DVI) specification is applied to display the processed image sequence to the LCD monitor through a DVI transmitter in the DVI interface module. This module generates the sync signals and image data for the DVI transmitter.

The classifiers module performs the classification for the multi-view face detection. It is the critical module of the whole multi-view face detection system. This module consists of the image line buffers, image window buffer, rotated image window buffers, and integral image window buffers to generate the integral image windows for each classifier, classifiers, stage accumulators, stage comparators, feature training data, stage training data, and arbitrator to perform the classification as shown in Fig. 3.

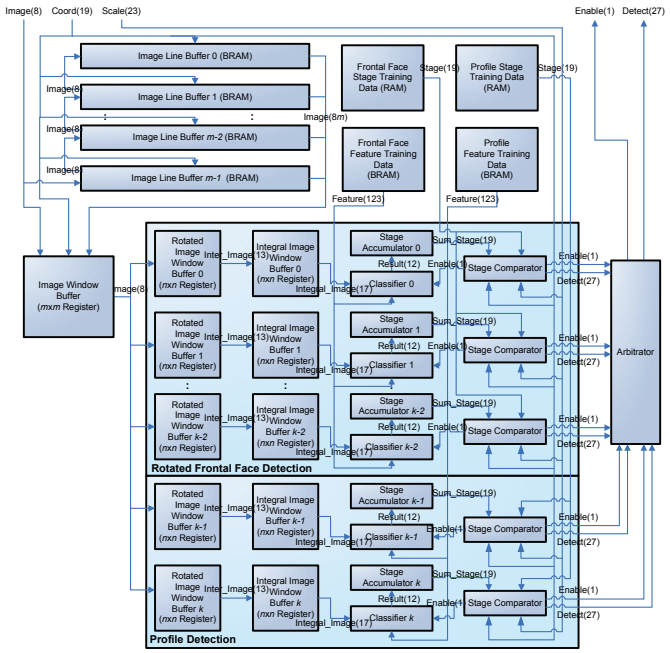

Figure 3. Block diagram of the classifiers module.

In this paper we implement the try-all-poses approach instead of a two stage system to ensure high accuracy of detection. In order to avoid the computational expense of having to evaluate every detector on every window in the input image, we design and implement a parallel architecture of multiple detectors for multi-view face detection in real-time. In order to detect the up-right (rotated) faces, we first de-rotate the image window obtained from the $m \times m$ (29×29) image window, where $m = \sqrt{n^2 + n^2}$, to generate the $n \times n$ (20×20) rotated image windows about all directions, and then generate the $n \times n$ (20×20) integral image windows using the $n \times n$ (20×20) rotated image windows for each classifier.

The integral image generation requires substantial computation. It may take a long latency delay every frame. In order to reduce memory access and processing time, we propose a specific architecture for the integral image generation [8]. This architecture stores the necessary pixels for processing each pixel and its neighboring pixels together. The image line buffers store some parts of the image. The image line buffers use dual port BRAMs where the number of BRAMs (m-1) is the same as that of the row-1 (29-1=28) in the image window buffer. Each dual port BRAM can store one line of an image. Thus, the x-coordinates of the pixels can be used as the address for the dual port BRAM.

The image window buffer stores pixel values moving from the image line buffers. Since pixels of an image window buffer are stored in registers, it is possible to access all pixels in the image window buffer simultaneously to generate the rotated image windows. Since the de-rotation of each rotated image window is performed by using the pre-calculated pixel mapping information, using a nearest neighbor interpolation algorithm, the latency of the rotation of the image window takes only one clock cycle. The pixels of a rotated image buffers are stored in registers. The integral image window buffers calculate the integral images of each rotated image buffer. There are two steps to calculate the integral images. The first calculates the accumulated values of each column of the rotated image window buffer. It takes $n/2$ (20/2=10) clock cycles using two accumulators. The second calculates the accumulated values of each row using the accumulated values of each column of the rotated image window buffer. It takes another $n/2$ (20/2=10) clock cycles. With these operations, the integral image windows are calculated from the rotated image window buffers. The total latency of the integral image generation takes n (20) clock cycles. These operations are processed with the classifications processing in the classifiers at the same time. Since the integral images which are currently calculated in the rotated image window buffer are for the classifications in the next position, it can save the latency time of the integral image generation.

We design and implement multiple classifiers for multi-view face detection. These classifiers work in parallel. The classifier has its own integral image window buffer which is generated from the rotated image buffer. In order to detect non-upright (rotated) faces, we deal with all rotation covering [-180˚, 180˚]. However, while standing, a person can tilt his or her head by [-45˚, 45˚] [9]. Since the frontal face detector from Viola and Jones handles approximately ±15 degrees of in-plane rotation, 12 detectors cover all rotation as [-180˚, 180˚], and 3 detectors cover the rotation as [-45˚, 45˚]. Thus, to detect faces of a standing person, 3 detectors are sufficient. Each detector use the training data for frontal face detection and each integral image window from the 0 degree, 30 degree, and -30 degree rotated image window, respectively. In order to detect non-frontal (profile) faces, we use a right profile detector with the integral image window from the 0 degree rotated image window. To detect a left profile faces, we simply flipped the 0 degree rotated image window to generate the integral image window for the left profile detector. The profile detectors handle out-of-plane rotations for about 3/4 view to full profile. Since the frontal upright detector handles faces from about left 3/4 view to right 3/4 view, the three detectors, right profile, frontal upright, and left profile detectors, combine to handle the full range of upright faces from left profile to right profile.

We use the training data from OpenCV [3][4] to detect the frontal [10] and profile [11] human faces based on the Viola and Jones algorithm [1]. These cascades of the training data are trained by frontal faces and profile faces of size 20×20, respectively. The cascade for frontal face detection includes a total of 22 stages, 2315 classifiers, and 4630 rectangle features. The cascade for profile face detection includes a total of 26 stages, 2609 classifiers, and 5218 rectangle features. The training data for frontal face detection are used for non-upright (rotated) face detectors to detect rotated faces. The training data for profile face detection are used for non-frontal (profile) face detectors to detect left and right profile faces. All training data for both frontal and profile faces are stored in the BRAMs.

Classifiers have rectangle features which consist of two or three rectangles of different sizes and their weight, threshold, and left and right values. Each rectangle presents four points, *x*, *y*, *x+width*, and *y+height*. The integral image value of each rectangle can be calculated using these points from the corresponding integral image window buffer simultaneously which reduces the memory access time. We design architecture of the classifier for face detection [8]. This architecture is for a classifier with only rectangular features, but can be easily extended for other features (such as the diagonal features described in Jones and Viola [7]) by adding a few more computational elements. The proposed architecture of the classifier is implemented based on a pipeline scheme. The latency for the first result of the classifier is five clock cycles. The arbitrator gathers the results of all classifiers and manages each classifier to perform the classification continuously or terminate the classification with current image window based on the result of each classifier. The arbitrator outputs the detection information and enable signal.

IV. Experiment / Result

The proposed systems for multi-view face detection are designed using Verilog HDL, synthesized using Synplify Pro, and implemented in a Virtex-5 LX330 FPGA using the ISE design suite [12]. There are two implementations: 5 classifiers to detect faces (three 0˚, 30˚, -30˚ rotated and two right and left profile faces) of standing person for both 320×240 (QVGA) resolution images and 640×480 (VGA) resolution images.

We measure the performance of the proposed parallelized architecture of hardware acceleration for multi-view face detection. We apply the implemented multi-view face detection system to a camera, which produces images consisting of 640×480 pixels at 60 frames per second. Since the system performance of face detection depends on the number of faces in the images, we test the implemented face detection system on several images which have various numbers of faces, and measure the performance as the average processing time. On 320×240 pixel images, our multi-view face detection system is capable of processing the images at speeds of an average of 68.41 ms (14.61 fps). On a 640×480 pixel images, our multi-view face detection system is capable of processing the images at speeds of an average of 258.24 ms (3.87 fps). In the software implementation by Jones and Viola [7], the two stage rotated face detector takes about 140 ms (7.14 fps) and the try-all-rotations detector takes about 660ms (1.51 fps) to process a 320×240 pixel image on a 2.8 GHz Pentium 4. The two stage profile detector process a 320×240 pixel image in about 120 ms (8.33 fps) on a 2.8 GHz Pentium 4 machine. Our multi-view face detection system has the performance improvement of 2.04 times over the Jones and Viola two stage rotated face detector, 9.67 times over their try-all-rotations detector, and 1.75 times over their two stage profile detector software implementation. Our system can also detect both rotated and profile faces in a single system whereas the Jones and Viola use a separate system for each detection. We do not compare our results to the Rowley et al. implementation [5] because it was implemented 10 years ago on a 200MHz R4400 SGI Indigo 2 and thus provide results that are too slow to offer a meaningful comparison.

V. Conclusion

We present a parallelized architecture of hardware acceleration for multi-view face detection. In our architecture, the multi-view face detection system generates rotated image windows and their integral image windows for each classifier which perform parallel classification operations to detect non-upright (rotated) and non-frontal (profile) faces in the images. We design the proposed architecture using Verilog HDL and implement the architecture in a Xilinx Virtex-5 FPGA. The parallelized architecture of multi-view face detection can have about 2 times performance improvement over the Jones and Viola two stage rotated face detector and about 9 times performance improvement over the Jones and Viola try-all-rotations detector software implementation.

References

[1] P. Viola and M. Jones, "Robust Real-Time Face Detection," *International Journal of Computer Vision*, 57(2), 137-154, 2004.

[2] Y. Freund and R. E. Schapire, "A Decision-Theoretic Generaliztion of On-Line Learning and an Application to Boosting," *Journal of Computer and System Sciences*, no. 55, pp. 119-139, 1997.

[3] Open Computer Vision Library, Sourceforge.net, January 2009. DOI=http://sourceforge.net/projects/opencvlibrary/.

[4] G. Bradski and A. Kaehler, *Learning OpenCV: Computer Vision with the OpenCV Library*, O'Reilly Media, Inc., 2008.

[5] H. Rowley, S. Baluja, and T. Kanade, "Rotation Invariant Neural Network-Based Face Detection," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 38-44, 1998.

[6] H. Schneiderman and T. Kanade, "A Statistical Method for 3D Object Detection Applied to Faces and Cars," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 746-751, 2000.

[7] M. Jones and P. Viola, "Fast Multi-view Face Detection," *Mitsubishi Electric Research Laboratories*, TR2003-096, 2003.

[8] J. Cho, S. Mirzaei, J. Oberg, and R. Kastner, "FPGA-Based Face Detection System Using Haar Classifiers," *ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, pp. 103-112, 2009.

[9] J. Kim, S. Kee, and J. Kim, "Fast Detection of Multi-View Face and Eye Based on Cascaded Classifier," *IEEE Conference on Machine Vision Applications*, pp. 116-119, 2005.

[10] R. Lienhart and J. Maydt, "An Extended Set of Haar-like Features for Rapid Object Detection," *IEEE Conference on Image Processing*, vol. 1, pp. 900-903, 2002.

[11] D. Bradley, "Profile Face Detection," *Intel Research Award Contest*, 2003.

[12] Xilinx Inc., "Virtex-5 Family Overview," February 2009. DOI=http://www.xilinx.com/.