REAL-TIME IMPLEMENTATION OF ROBUST FACE DETECTION ON MOBILE PLATFORMS

M. Rahman, Jianfeng Ren, and N. Kehtarnavaz

Department of Electrical Engineering, University of Texas at Dallas

ABSTRACT

Although many face detection algorithms have been introduced in the literature, only a handful of them can meet the real-time constraints of mobile devices. This paper presents the real-time implementation of our previously introduced face detection algorithm on a mobile device. The steps taken to achieve such a real-time implementation are discussed. Real-time comparison results with the widely used Viola-Jones face detection algorithm in terms of detection rate and processing speed are presented to demonstrate the robustness of our real-time solution.

Index Terms – Real-time face detection, mobile platform, implementation of face detection.

1. INTRODUCTION

The applications of human face detection and tracking on mobile devices has been growing recently. Face detection has been extensively examined in the image processing literature. In [1], Yang *et al.* presented a survey on various face detection algorithms. While many face detection algorithms have been reported to generate high detection rates, very few of them are suitable for real-time deployment on mobile devices such as cell-phones due to the computation and memory limitations of these devices. Normally, real-time implementations of face detection algorithms are done on PC platforms with relatively powerful CPUs and large memory sizes. Real-time software implementation on mobile platforms without using any dedicated co-processor poses its own challenges.

The examination of the existing face detection products reveal that the algorithm introduced by Viola and Jones [2] has been widely adopted. This algorithm uses a boosted cascade of simple features. It is shown that the detection rate of this algorithm is quite high but it drops noticeably for profile, rotated, tilted, or partially covered faces [3]. Although modifications have been introduced to address various face poses, e.g. [4], these modifications increase its processing time to a great extent on a mobile device.

In [5] and [6], we introduced a computationally efficient face detection algorithm based on human skin color and simple shape processing which was shown to be robust to various face poses under different lighting conditions.

As far as real-time face detection on mobile devices is concerned, appropriate implementation steps need to be made in order to achieve a real-time throughput. This paper presents such steps for real-time deployment of our face detection algorithm on a mobile platform. The steps taken are general purpose in the sense that they can be used to implement similar computer vision algorithms on any mobile device.

Section 2 provides an overview of our face detection algorithm. In section 3, the real-time implementation steps are covered. The implementation results and comparison with the Viola-Jones algorithm are presented in section 4 and the conclusions in section 5.

2. OVERVIEW OF OUR FACE DETECTION ALGORITHM

Considering that distribution of human skin color of different ethnicity forms a tight cluster in any chrominance color space, it is possible to represent this cluster by a Gaussian mixture model (GMM) as discussed in [7]. A well trained GMM in any chrominance space is capable of representing the skin tone regardless of the ethnicity. Since raw Bayer pattern images captured by a typical cell-phone camera sensor are internally transformed into the YCrCb color space for compression purposes, this color space is chosen here to avoid any internal color conversion computation.

The face detection process starts by first converting the captured image into a binary image by performing a skin color extraction using the GMM model with 1's representing pixels of the image corresponding to skin color and 0's representing non-skin color pixels. The binary image is then passed through a fast sub-block shape processing scheme which uses face size, aspect ratio and probability scoring to determine any face location in the image. To reduce the computational time, lookup tables are used to define the skin cluster area. To accommodate for different lighting conditions, a number of lookup tables corresponding to different lighting conditions are used. The lookup table based on the highest skin probability is then chosen to represent the best matching light source.

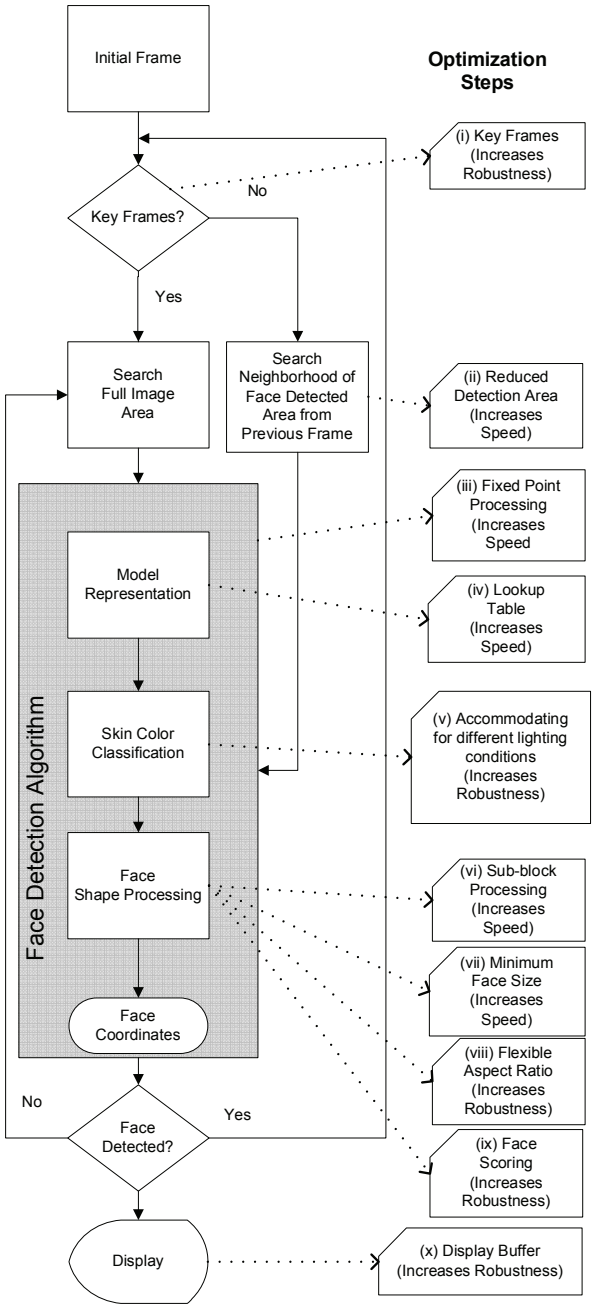

Fig. 1. Components of our face detection algorithm and the optimization steps taken for real-time deployment on mobile platforms.

Fig. 1 illustrates various components of our face detection algorithm and Fig.2 provides a sample outcome after its color and shape processing. First, skin pixels corresponding to different lighting conditions are extracted based on trained GMM models. These models are used to perform skin color extraction followed by a simple shape processing module. In case of multiple detected faces, the algorithm is designed to pick the largest face appearing in the image based on the total skin probability in detected face regions. More details of the algorithm are provided in [5] and [6].

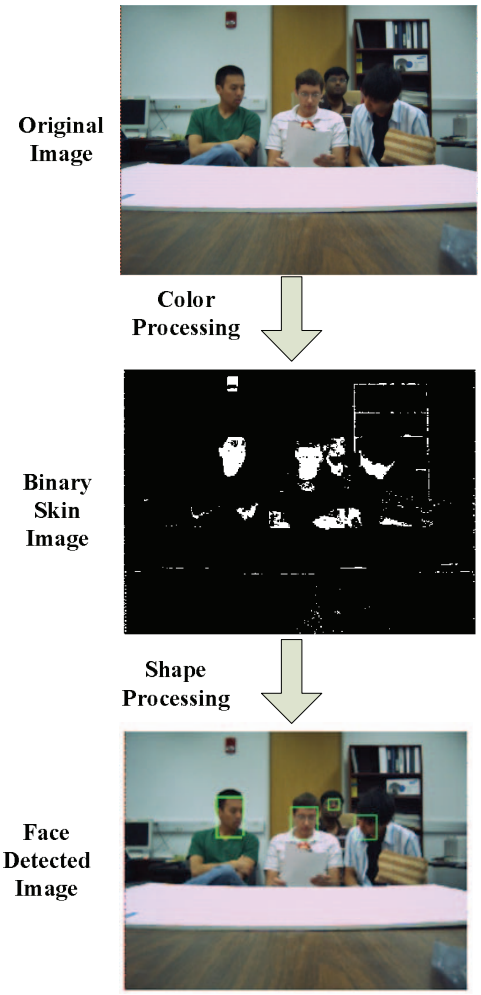

Fig. 2. Face detection using both color and shape features.

For real-time deployment on mobile devices, a number of optimizations were carried out which are described in the next section.

3. REAL-TIME IMPLEMENTATION

Both the Viola-Jones and our algorithm would not run in real-time when implemented on mobile devices due to the limited processing power of such devices. In fact, we implemented both of these algorithms on the Texas Instruments OMAP platform which is the adopted engine in many modern mobile phones. The processing time per frame

took more than a minute on this platform due to the high computation demands of these algorithms. In this section, we present the optimization steps that were taken in order to achieve a robust real-time throughput on the OMAP platform. These steps are general purpose, that is to say the same steps can be taken for real-time implementation on other mobile platforms. The optimization steps made are listed on the right side of Fig. 1.

i. **Key frame:** The concept of key frame was introduced to synchronize face detection and tracking. In the results reported in the next section, a key frame was considered every thirty frames (every one second).

ii. **Reduced detection area**: If a face was detected in a key frame, then the search region for next frames was set to be in a neighborhood area around the detected face. The full image area was used again in a next key frame or if the algorithm failed to find any face in a previous frame. This stabilized occasional abrupt changes in the face position or when a new face entered the image.

iii. **Fixed-point processing:** The great majority of mobile devices have fixed-point processors and it is quite inefficient to perform floating-point computation on fixed-point processors. During this optimization step, all the computations were redone using the fixed-point Q format.

iv. **Lookup table:** Computation of skin probability for each pixel using a GMM makes the processing time not practical for real-time deployment. To reduce the processing time, a pre-calculated lookup table containing the probability values for all CrCb combinations was used. Considering that the variation of skin color is quite small compared to the entire color space, a 50x50 lookup table was adequate to represent the skin cluster area.

v. **Accommodating for different lighting conditions:** Several lookup tables for different lighting conditions were considered to automatically choose the right lookup table by examining the total skin probabilities for all the lookup tables and selecting the one with the highest probability.

vi. **Sub-block processing**: This optimization step consisted of sub-dividing the binary skin image into blocks. The total skin area within a block was computed, and if this was greater than or equal to 50% of the block area, the block label was assigned to be skin. This step effectively reduced the size of the input image, thus significantly speeding up subsequent processing. To get the proper block size, a ROC (receiver operating characteristic) curve analysis was performed. Based on the ROC curve analysis, for a VGA resolution image, a block size of 10 and a skin probability confidence level of 2% were selected which corresponded to a detection rate of 91% [5].

vii. **Minimum face size:** By defining a minimum face size, detection of faces with a size less than the minimum size was avoided. For the results reported next, the minimum face size was set to 80x80 pixels for image resolution of 640x480, i.e. one sixth of the image height.

viii. **Flexible aspect ratio:** The standard golden ratio, which is 1.618, works well for full frontal faces but not for other face poses [8]. To accommodate for other face poses, a range of aspect ratios (0.8-1.8), was used.

ix. **Face scoring:** To cope with situations when multiple faces appeared in frames, a scoring mechanism based on the total skin probability was used to determine the best face region which in most cases corresponded to the largest size face in the image.

x. **Display buffer:** A display buffer was used to continuously draw rectangles around faces. If a face could not be detected, any previously detected face in the display buffer was used for this purpose.

4. REAL-TIME RESULTS AND COMPARISON

Our real-time implementation was carried out on the OMAP3430 mobile platform noting that it is a widely adopted platform in many modern cell-phones This platform possesses a triple core engine consisting of an ARM Cortex-A8 processor, a graphics processor, and a C6400 DSP processor. More details on this processor can be found in [9]. We tested our algorithm under different light sources, face orientations, backgrounds, and also for different people. Some sample real-time face detection outcomes are displayed in Fig. 3. Figure 4 shows a snapshot of our face detection algorithm running in real-time on the OMAP3430 mobile device.

Fig. 3. Real-time face detection outcomes.

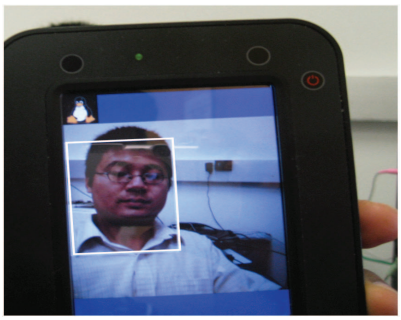

Fig. 4. Snapshot of our face detection running in real-time on OMAP3430 mobile platform.

The Viola-Jones algorithm was also optimized and implemented on the same platform as reported in [10]. Table I and II provide typical comparison outcomes between the two real-time implementations in terms of robustness and processing speed. A total of five people under different backgrounds and face orientations were tested for the comparison. Table I summarizes the overall performance for frontal, profile, partial, rotated, tilted and covered face poses.

TABLE I
COMPARISON WITH VIOLA-JONES (ROBUSTNESS)

Frontal/ Rotated/ Partial	Viola & Jones (detection/ number of frames)	Our algorithm (detection/ number of frames)
Frontal faces	189/200	185/200
Profile faces, partial faces	59/200	186/200
Rotated faces, tilted faces	97/200	165/200
Covered faces (eye, mouth)	129/200	177/200
Overall detection rate	474/800 (59.3%)	708/800 (88.5%)

For full frontal faces, the detection rates for both of the algorithms were high. However, for profile, partial, rotated, tilted, and covered faces, the detection rate for the Viola-Jones algorithm became much lower than our algorithm. For partially covered faces, the performance of the Viola-Jones deteriorated considerably. Taking into consideration all face orientations, a real-time detection rate of 59% was achieved when running the Viola-Jones algorithm as compared to a real-time detection rate of 88%when running our algorithm.

Table II summarizes the overall processing speed for both of the algorithms. Four different scenes each having 100 frames were evaluated with different skin color subjects and backgrounds. For the optimized Viola-Jones algorithm, the average processing time per frame was 90.1ms, whereas for our algorithm, the average processing time per frame was 52.9ms. Hence, considering both the robustness and processing time, our algorithm achieved a better real-time mobile device face detection throughput.

TABLE II
COMPARISON WITH VIOLA- JONES (PROCESSING TIME)

	Viola & Jones (average time required to process one frame in ms)	Our algorithm (average time required to process one frame in ms)
Videoclip 1	88.7	45.9
Videoclip 2	101.2	56.1
Videoclip 3	85.2	54.7
Videoclip 4	85.4	55.2
Average	90.1	52.9

5. CONCLUSION

This paper has presented the real-time implementation of a robust face detection algorithm on mobile devices. The optimization steps introduced have allowed the real-time implementation of the algorithm on resource limited mobile devices. The results of an actual real-time implementation on a modern mobile platform have shown that the algorithm is quite robust to various face orientations as compared to the Viola-Jones algorithm.

6. ACKNOWLEDGEMENT

This work was sponsored by the Wireless Terminal Business Unit of Texas Instruments.

7. REFERENCES

[1] M. Yang, D. Kriegman, and N. Ahuja, "Detecting faces in images: a survey," *IEEE Trans. on PAMI,* vol. 24, no. 1, pp. 34-58, Jan 2002.

[2] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," *IEEE Computer Society Conf. on Computer Vision and Pattern Recognition*, vol. 1, pp. 511-518, Dec 2001.

[3] C. Huang, H. Ai, Y. Li, and S. Lao, "High-performance rotation invariant multiview face detection," *IEEE Trans. on PAMI,* vol. 29, no. 4, pp. 671-686, Apr 2007.

[4] P. Viola and M. Jones, "Fast multi-view face detection," *Technical Report TR2003-96, Mitsubishi Electric Research Laboratories*, Jul 2003.

[5] M. Rahman, M. Gamadia, and N. Kehtarnavaz, "Real-time face-based auto-focus for digital still and cell-phone cameras," *IEEE Southwest Symposium on Image Analysis and Interpolation*, vol. 1, pp. 177-180, Mar 2008.

[6] M. Rahman, and N. Kehtarnavaz, "Real-time face-priority auto-focus for digital and cell-phone cameras," *IEEE Trans. on Consumer Electronics*, vol. 54, no.4, pp. 1506-1513, Nov 2008.

[7] R. Hsu, M. Mottaleb, and A. Jain, "Face detection in color images," *IEEE Trans. on PAMI,* vol. 24, no. 5, pp. 696-706, May 2002.

[8] Y. Chan, and S. Abu-Bakar, "Face detection system based on feature-based chrominance color information," *Proc. Inter. Conf. on Computer Graphics, Imaging and Visualization,* pp. 153-158, Jul 2004.

[9] Texas Instruments Inc., *OMAP3 Architecture from TI for MID*, Aug 2008 (available online).

[10] J. Ren, N. Kehtarnavaz, and L. Estevez, "Real-Time optimization of Viola-Jones face detection for mobile platforms," *Proceedings of Seventh IEEE Dallas Circuits and Systems Workshop,* vol. 1, no.1, pp. 1-4, Oct 2009.