ROBUST FACE DETECTION IN PATIENT TRIAGE IMAGES

Niyati Chhaya and Tim Oates
University of Maryland, Baltimore County, 1000 Hilltop Circle, Baltimore, Maryland, U.S.A.

Keywords: Face detection, Feature extraction, Triage images.

Abstract: Disaster events like the attack on the World Trade Center in New York City in 2001 and the earthquake in Haiti in 2010 result in a desperate need for family and friends to obtain information about missing people. This can be facilitated by automatically extracting textual descriptors of patients from images taken at emergency triage centers. We show that existing face detection algorithms, a necessary precursor to extracting such descriptors, perform poorly on triage images taken during disaster simulations, and propose and demonstrate the superiority of an ensemble-based face detection method that is a combination of robust skin detection and simple pattern matching face detection techniques. The addition of a template-based eye detection method further improves the accuracy of our face detection method.

1 INTRODUCTION

Disaster events like the attack on the World Trade Center in New York City in 2001 and the earthquake in Haiti in 2010 have both local and non-local effects. Locally, there is significant loss of life and property. Non-locally, there is a desperate need for information about the status of friends and family who might have been directly impacted. An increasingly common practice in disaster preparedness drills, which are required annually of hospitals in the United States, is the use of digital cameras to take pictures of patients as they are triaged. Privacy concerns, as well as the sometimes graphic nature of the images when injuries are severe, makes it impractical to make the images publicly accessible. However, descriptive features extracted from the images – hair color and style, clothing color and style, distinctive features like tattoos, gender, ethnicity, age, etc. – can be made publicly available and searchable.

To extract descriptive features from triage images, one must first locate the patient's face. This proves to be a difficult task given that triage images are taken in highly variable lighting conditions from many different viewing angles and distances, and the patients' faces are often obstructed with bandages, dirt, debris, and blood. In this paper we propose a novel ensemble based face detection algorithm that combines skin detection and face detection based on pattern matching that works well (in terms of precision and recall) for our corpus of images. We also show that state-of-the-art face detection algorithms perform poorly on a corpus of patient triage images taken during a disaster preparedness drill. Finally, we apply template based eye detection along with the proposed face detection algorithm to show how locating eyes in the image gives better results for the resultant face detection. Figure 1 shows the overview of the proposed ensemble based algorithm.

The remainder of the paper is organized as follows. Section 2 reviews related work for skin, face, and eye detection. Our proposed approach is described in Section 3. Section 4 presents our dataset and empirical results, and Section 5 concludes and points to future work.

2 RELATED WORK

Even in the domain of disaster response where faces can be obstructed in a variety of ways, good skin detection is essential. Probabilistic models such as (Elgammal et al., 2009) compute the probability that a pixel corresponds to skin given, for example, it's RGB values. Others approaches (Vassili et al., 2003) make hard decisions. For example, Peer *et al.* (Peer et al., 2003) declare all pixels that meet the following conditions to correspond to skin:

$$\begin{aligned} R > 95, G > 40, B > 20 \\ max(R,G,B) - min(R,G,B) > 15 \\ |R-G| > 15, R > G, R > B \end{aligned} \quad (1)$$

Chhaya N. and Oates T. (2011).
ROBUST FACE DETECTION IN PATIENT TRIAGE IMAGES.
In *Proceedings of the International Conference on Computer Vision Theory and Applications*, pages 326-333
DOI: 10.5220/0003319303260333

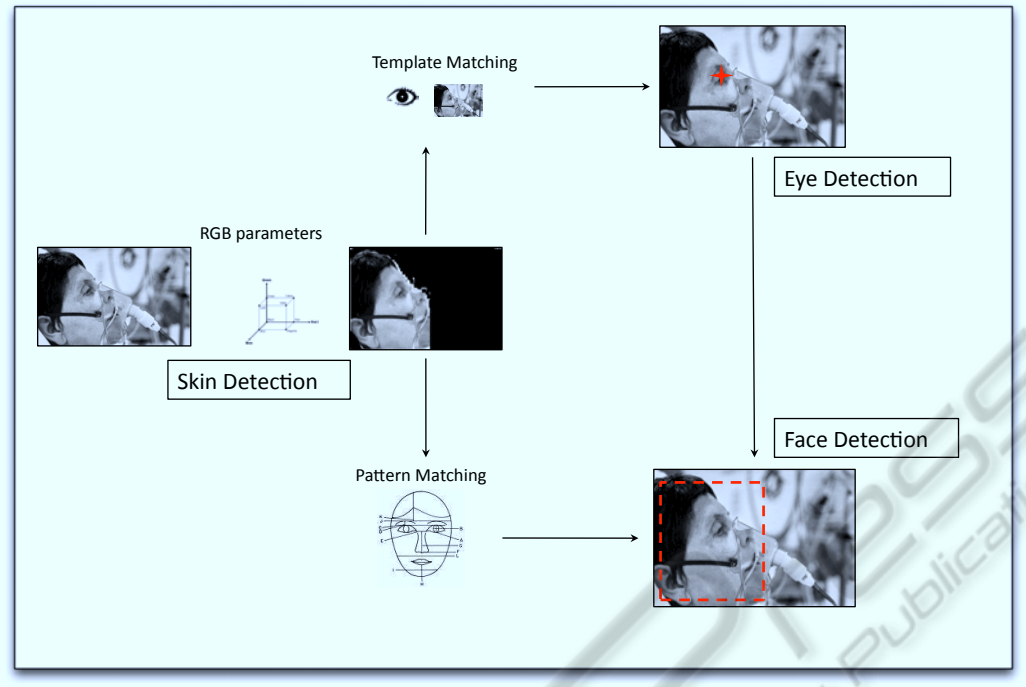

Figure 1: An overview of the processing done on each patient image for face detection.

This algorithm is a component of ours. Once relevant pixels are found, they can be grouped into distinct regions that may correspond to different body parts or different people (Aznaveh et al., 2008; Ravichandran and Ananthi, 2009; Phung et al., 2005).

In this work we are concerned with face region detection (Yang et al., 2002) (finding contiguous regions in an image that mostly contain faces), which is contrasted with face recognition (determining the identity of a person based on an image of their face) and face detection (marking the locations of faces, but not their spatial extent). Some approaches use combinations of facial features for face region localization. For example, (Pai et al., 2006), which we incorporate in our algorithm, uses the height to width ratio of the face and the presence of eyes and a mouth. Ratios in the range $[0.75, 1.75]$ are acceptable, and eye and mouth locations are found using color histograms and shape features. The baseline against which we compare our results is the Viola-Jones face detection algorithm (Viola and Jones, 2001) which uses a feature representation based on Haar wavelets and trains a discriminative model using AdaBoost (Collins et al., 2000). This algorithm has state-of-the-art performance on commonly used datasets.

Many face detection algorithm can take advantage of information on the location of eyes. There are numerous approaches to eye localization, such as the use of active IR (Peng et al., 2005) which is not feasible for disaster events due to the requirement for specialized equipment. Others use eye templates extracted from images either to train supervised classifiers or for matching. We incorporate a template matching algorithm in our approach. Combination approaches such as locally selective projection with SVMs (Zheng and Wang, 2008), rotation invariant Zernike moments with SVMs (Kim and Kim, 2008), and face circle fitting with dark pixel filters (Lin and Yang, 2004) have also been studied.

Interestingly, many of the most commonly used datasets in face detection were originally developed for face recognition, and thus are relatively uniform. For example, the FERET face dataset (Phillips et al., 2000) contains images taken from frontal and left/right profile views, the Yale dataset (Yal, 2006) contains frontal images with glasses and various facial expressions, and the MIT dataset (MIT, 2000) contains images of 10 different people taken under different lighting conditions and poses. All of these datasets lack diversity along a number of dimensions: race, open vs. closed eyes, presence of facial hair, tattoos, etc. This is part of the reason that existing algorithms, developed and tested on uniform data, perform poorly on images from disaster events that vary along all of these dimensions and many others.

3 PROPOSED FACE DETECTION ALGORITHM

As shown in Figure 1, the input patient image undergoes a sequence of processing events. The face detection task is treated as a combined result from a color–based skin detector, pattern matching–based face detector and a template matching–based eye detector. The system takes an input image and cleans up the background using skin detection, this processed image is then fed into the face detection algorithm. Using this combination the system needs to find the face only within the detected skin region. To further improve face detection accuracy, eye detection is used. After the skin detection step, eye detection is performed, this gives us eye coordinates. This output is fed into the face detection algorithm and then pattern matching is done with the already known eye coordinates. The system is hence a combination of interlocked algorithms where the feedback from eye detection is used to improve the performance of the face detection algorithm. Each component of this ensemble system is explained here.

Referring back to Figure 1, first, skin detection is done. The skin detection algorithm of Peer *et al.* (Peer et al., 2003) is used, modified slightly so that $max(R,G,B) - min(R,G,B) > 10$ rather than $max(R,G,B) - min(R,G,B) > 15$ to allow for more variation in ethnicity, obstruction by dirt and blood, etc.

As mentioned, skin detection allows us to eliminate the background and focus on those regions that are most likely to contain a face, which is done with the face detection algorithm in (Pai et al., 2006) that uses the height to width ratio of the skin region and the presence and location of eyes and a mouth. If any of these features are present in the image the algorithm computes a bounding rectangle around the face. We relaxed the maximum allowed height to width ratio from 1.75 to 1.8 to allow for more variation. The eye localization algorithm uses color histograms in the $Y'C_BC_R$ color space. Likewise for mouth localization.

For eye detection, we implemented a template-based eye detection algorithm given that many of the patients have their eyes closed, thus confounding color-based approaches. Fifty templates were extracted from images not used in our evaluation. Some of them are shown in Figure 2. For matching we use two different methods and combine the results. The first is the normalized sum of squared differences (SSD) between the template and image patches; the second is the normalized cross correlation (NCC). NCC is useful when lighting conditions vary, as they do in our images. Equation 2 and Equation 3 give expressions for SSD and NCC, respectively.

$$ssd(u,v) = \Sigma_{x,y}(f(x,y) - t(x-u,y-v))^2 \quad (2)$$

$$ncc(u,v) = \frac{1}{n-1}\Sigma_{x,y}\frac{(f(x,y)-\bar{f})(t(x,y)-\bar{t})}{\sigma_f \sigma_t} \quad (3)$$

In the above equations, n is the number of pixels, f is the image, t is the template, and the summation is over positions x,y under the template at position u,v. σ is the standard deviation.

(a) (b) (c)

Figure 2: Example eye templates.

Because patients can be, and are, in many different orientations in our images, we rotate each eye template and use the rotated templates as independent templates for comparison. The rotation is done around the center of the eye template in 30 degree increments, producing twelve different templates from each original. Other transformations, such as scaling, were tested, but had little impact on the results. Empirically, the template-based method works better than the built-in method of (Pai et al., 2006), so to improve face localization we first locate the eyes using templates and then run the face detection algorithm.

This system being an ensemble of algorithms, treats the face as a combination of different features on a given skin area, whereas the Viola-Jones approach, the system is looking for a complete set of Haar-like features that match the face. Hence, if there is a partial face or a face with a different orientation, our system performs considerably better than the standard face detection algorithms that are trained specially for full upright straight face images.

4 EXPERIMENTAL RESULTS

Recall that our problem domain is images of patients taken during disaster events. Much of our data came from drills conducted by the Bethesda Hospitals' Emergency Preparedness Partnership (BHEPP), which is a collaboration between the National Navel Medical Center, the National Institutes of Health Clinical Center, and the Suburban Hospital in Bethesda, Maryland. One of the goals of BHEPP is to develop a transportable, proven model for defining

Table 1: Dataset Summary.

Source	Count	Remark
CMAX	22	BHEPP disaster drill
Web	6	
Lost Person Finder	22	Female
Lost Person Finder	26	Male
Lost Peron Finder	21	Child
Total	97	

best practices in casualty care management through ongoing multi-agency collaboration, integration, and training. Part of the training is an annual Collaborative Multi-Agency Exercises (CMAX) drill in which a disaster event is simulated, complete with actors playing the role of disaster victims. Patients are taken to emergency medical care centers for treatment where they are triaged and a primary record is created, including an image of the patient. In addition, some of our images come from the National Institutes of Health Lost Person Finder (LPF) project, which is a web-based system for re-uniting family and friends with victims during disaster events.

Figure 3 shows sample images from our dataset and contrasts them with images often used in face detection and recognition work. Note that the faces in the BHEPP images are in different orientations, fill different amounts of the field of view, are obstructed by bandages and other visible signs of injury, have open and closed eyes, are surrounded by other patients and medical personnel, etc. In contrast, images from standard face datasets are remarkably uniform. As we will see shortly, state-of-the art face detection algorithms perform exceptionally poorly on the BHEPP images, necessitating a new approach. Table 1 gives a summary of the 97 images used in our experiments.

We compared four different face detection approaches as described below:

- OpenCV: The state-of-the-art Viola-Jones face detection algorithm (Viola and Jones, 2001) implemented in the OpenCV toolkit
- Skin + OpenCV: Skin detection is done before running the Viola-Jones algorithm
- Skin + FaceDetect: Skin detection is done before running the face detection algorithm of *Pai et al.* (Pai et al., 2006)
- Skin + Eye + FaceDetect: This is the same as Skin + FaceDetect except the template-based eye detection method is used

For setting a standard testing platform, we asked humans to manually annotate our image set for face detection. They were asked to draw the tightest bounding rectangle around the face region in each image. Figure 4 shows examples of these annotated images. These annotations were used as reference images and results from each of the different methods are compared against the annotated reference images. For numerical analysis, we use precision/recall values and the F-Score, defined as:

$$Precision(P) = \frac{Annotated \cap Program}{Program}$$

$$Recall(R) = \frac{Annotated \cap Program}{Annotated} \quad (4)$$

$$F - Score = \frac{2 \times P \times R}{P + R}$$

We calculate precision and recall values by counting the amount of pixel overlap between the human annotations and the bounding boxes produced by the algorithms. If the result from our algorithm falls completely inside the human-annotated area then we have perfect precision and a lower recall value. If the result from our algorithm completely contains the human-annotated rectangle then we have perfect recall and lower precision. And, finally, if there is no overlap between the regions from human annotation and that from program then precision and recall are 0. There is no distinction between no overlap and no detection (no rectangle marked) in the results.

4.1 OpenCV

On testing face detection using OpenCV on our images we got extreme results in most cases. Large numbers of images had no detection of the face (precision/recall=0) whereas some images had perfect detection of the face (precision/recall=1). Table 2 summarizes the results. Due to OpenCV's failure to detect faces in almost 50% of the images, the average F-Score is also low (0.47) for this algorithm.

Table 2: Results: OpenCV.

Property	Number of Images
Total Images	97
No Detection	48
Perfect Detection	28
Average Recall	0.49
Average Precision	0.47
Average F-Score	0.47

The images with perfect face detection are all frontal clean face images, very similar to those from

(a) Source: BHEPP

(b) Source: BHEPP

(c) Source: BHEPP

(d) OpenCV test image: Lena

(e) Images from Yale face database

(f) Profile test image from CMU face database

Figure 3: Images from BHEPP disaster drills (a - c) compared to images often used in face detection and recognition work (d - f).

(a)

(b)

(c)

Figure 4: Annotated Images.

Table 3: Results: Skin + OpenCV.

Property	Number of Images
Total Images	97
No Detection	37
Perfect Detection	31
Average Recall	0.6
Average Precision	0.58
Average F-Score	0.59

standard face databases. As OpenCV is trained using Haar wavelets for frontal face features it fails for anything outside that frame. Cascading of Haar features, done as a part of the OpenCV system, also does not improve performance for non-frontal images. Another observation is that a cluttered background greatly affects the performance of OpenCV face detection. In some images, non-face regions were detected as faces.

4.2 Skin + OpenCV

The performance of OpenCV showed us that changing the background distracts the algorithm. Our next experiment involved first doing skin detection and then continuing to run OpenCV face detection on the images with only the skin area visible. This method helped us eliminate background noise. As seen from the results, the overall performance of the algorithm improved. Also the number of no-detections went down from 48 images in the previous experiment to 37 images in this experiment. Table 3 summarizes these results.

4.3 FaceDetect

We first performed skin detection on all images and then applied the pattern–matching–based face detection algorithm (FaceDetect) on the detected areas. The results in Table 4 show that there were very

Table 4: Results: Skin + FaceDetect.

Property	Number of Images
Total Images	97
No Detection	7
Perfect Detection	0
Average Recall	0.66
Average Precision	0.72
Average F-Score	0.65

Table 5: Results: Skin + Eye + FaceDetect.

Property	Number of Images
Total Images	97
No Detection	1
Perfect Detection	1
Average Recall	0.77
Average Precision	0.83
Average F-Score	0.78

few images with no face detection at all. Though at the same time, the number of images with perfect face detection is zero. The more frontal face images give high recall values but the precision is not perfect. Also, we observe that most face detected regions cover only part of the actual face and tend to be towards one side of the face rather than covering the complete face area.

4.4 Eye + FaceDetect

The results of face detection from the previous section showed that the detected face tends to be shifted on either side of the actual face. In this experiment, we first implemented eye detection on detected skin areas and then did the face detection with the eye locations known. Table 5 summarizes the results for all images. The number of no–detections reduces to just one. Also, the average precision/recall value is very high as compared to that from prior results. The increase in average F-Score shows that there is an improvement in precision, recall, or both.

4.5 Comparisons

In this section we compare the results of all the above methods. Figure 5 shows results when only OpenCV is used and when OpenCV is used after skin detection. From the graph we observe that the average performance improves and the number of no-detections also decreases.

Figure 6 shows results from both methods using the pattern–matching–based face detection. As seen from the graph, average performance for both of these

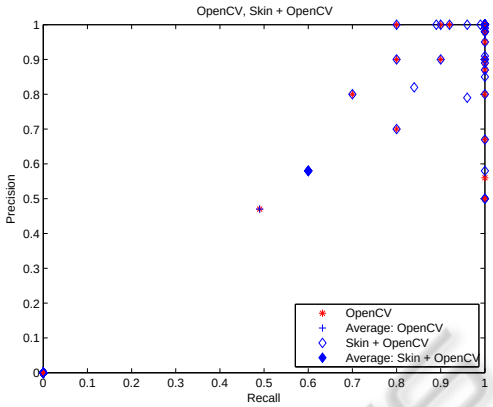

Figure 5: Graph: OpenCV, Skin + OpenCV – Each point in the graph represents precision/recall value of one image from the database.

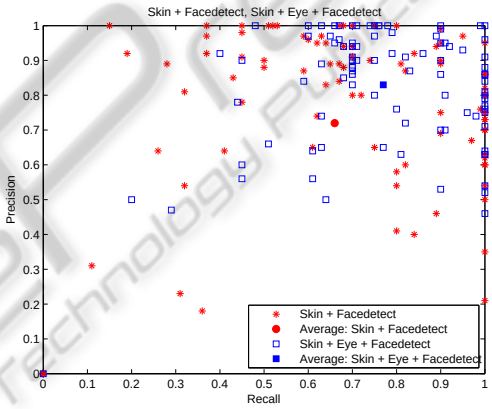

Figure 6: Graph: Skin + FaceDetect, Skin + Eye + FaceDetect – Each point in the graph represents precision/recall value of one image from the database.

methods is better than OpenCV and also that the average after Eye + FaceDetect improves as compared to that of FaceDetect.

Table 6: Results: Average F-Score Value.

Method	Avg. F-Score value
OpenCV	0.47
Skin + OpenCV	0.59
Skin + Facedetect	0.65
Skin + Eye + Facedetect	0.78

4.6 Eye Detection

Figure 8 shows the detected face area for a given image using different methods. The complete rectangle is the annotated reference region, the – rectangle is from FaceDetect result and the -. rectangle shows

Table 7: Results: Eye Detection–Without template rotation.

Image	Count	Correct Detected(%)	False Positives	No detected	Templates
Open Eyes	67	51 (76.12)	13	3	8
Closed Eyes	10	4 (40.00)	1	5	3
Half Open	15	7 (46.67)	1	7	4
Not Visible	5	0 (-)	1	0	0
All	97	62 (63.91)	16	15	15

Table 8: Results: Eye Detection–With template rotation.

Image	Count	Correct Detected (%)	False Positives	Not detected	Templates
Open Eyes	67	58 (86.57)	8	1	6
Closed Eyes	10	8 (80.00)	1	1	3
Half Open	15	14 (93.33)	1	0	3
Not Visible	5	0 (-)	1	0	0
All	97	80 (82.48)	11	2	12

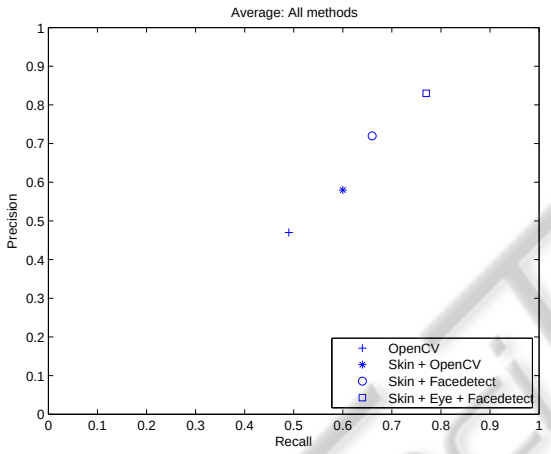

Figure 7: Graph: All Methods - Average.

Figure 8: Face Detection Result: — : Human annotated - - - : Skin + FaceDetect .-: Skin + Eye + FaceDetect.

result of the Eye + FaceDetect algorithm. As seen from the image, FaceDetect detects partial face and FaceDetect done after eye detection detects a more accurate face region. These results are reflected in Figure 7 where average precision and recall from all methods are shown. The graph shows an almost linear improvement in performance. Table 6 compares average F-Scores from all methods. These numbers also show improvement as the methods change. The best performance is shown using the Skin + Eye + FaceDetect method.

We performed an accuracy test to test the eye detection algorithm by itself. The images under the non–visible eyes section are those images where the person is either wearing sunglasses or has a bandage around the eye region.

Tables 7 and 8 show results of eye detection using template matching with normal templates and using rotated templates respectively. As observed from the results, eye detection with template rotation performs better than that without any rotation. The principal reason behind this result is the presence of large number non–upright face images in the dataset.

5 CONCLUSIONS

We began with comparing standard face databases against the dataset from real life mass disaster situation. The results from the previous section show that an ensemble–based face detection algorithm performs better than using off–the–shelf standard face detection algorithms. Relaxing parameters for skin detection helps in accommodating a wider range of skin color and noise in the skin area, hence resulting in improved performance. Performance of face detection algorithms improves drastically with elimination of unwanted background regions. Also, the face detection algorithm that we use gives better accuracy if eye detection is done first. Template–matching–based eye detection using a normalized sum of squared dif-

ference and normalized cross correlation is effective for this database. The key aspect of the algorithm is rotation of templates for comparison. The rotation allows accurate detection of eyes in images where the person is lying down or not in an upright position. Hence, we have both face to eye detection and eye to face detection. This allows us to check the performance of each algorithm and use the other one as a feedback mechanism to improve performance.

We developed a mesh–like structure with the algorithms, where each algorithm acts as a feedback for the other, hence validating the performance and aiding to improve performance of individual algorithms. Our work also illustrates how algorithms meant for standard face databases can be modified for real–life noisy databases. We use individual unrelated methods at each stage of detection such as color model analysis for skin detection, pattern–matching for face detection and template–matching for eye detection, all of these finally coming together for an improved face detection algorithm.

REFERENCES

2000, *MIT Face Recognition Database*. Online.

2006, *Yale Face Database*. Online.

Aznaveh, M. M., Mirzaei, H., Roshan, E., and Saraee, M. (2008). A new color based method for skin detection using rgb vector space.

Collins, M., Schapire, R. E., and Singer, Y. (2000). Logistic regression, adaboost and bregman distances. In *Conference on Learning Theory*, pages 158–169.

Elgammal, A. M., Muang, C., and Hu, D. (2009). Skin detection. In *Encyclopedia of Biometrics*, pages 1218–1224. Springer US.

Kim, H.-J. and Kim, W.-Y. (April 2008). Eye detection in facial images using zernike moments with svm. *Electronics and Telecommunications Research Institute*, 30:335–338.

Lin, D.-T. and Yang, C.-M. (2004). Real-time eye detection using face-circle fitting and dark-pixel filtering. In *International Conference on Multimedia and Expo*, pages 1167–1170.

Pai, Y.-T., Ruan, S.-J., Shie, M.-C., and Liu, Y.-C. (2006). A simple and accurate color face detection algorithm in complex background. *IEEE International Conference on Multimedia and Expo*, 0:1545–1548.

Peer, P., Kovac, J., and Solina, F. (2003). Human skin color clustering for face detection.

Peng, K., Chen, L., Ruan, S., and Kukharev, G. (2005). A robust and efficient algorithm for eye detection on gray intensity face. In *International Conference on Advances in Pattern Recognition (2)*, pages 302–308.

Phillips, P. J., Moon, H., Rizvi, S. A., and Rauss, P. J. (2000). The feret evaluation methodology for face-recognition algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(10):1090–1104.

Phung, S. L., Bouzerdoum, A., and Chai, D. (2005). Skin segmentation using color pixel classification: Analysis and comparison. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(1):148–154.

Ravichandran, K. and Ananthi, B. (2009). Color skin segmentation using k-means cluster. *International Journal of Computational and Applied Mathematics*, 4:153–157.

Vassili, V. V., Sazonov, V., and Andreeva, A. (2003). A survey on pixel-based skin color detection techniques. In *in Proc. Graphicon-2003*, pages 85–92.

Viola, P. and Jones, M. (2001). Robust real-time object detection. In *International Journal of Computer Vision*.

Yang, M.-H., Kriegman, D. J., and Ahuja, N. (2002). Detecting faces in images: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1):34–58.

Zheng, Y. and Wang, Z. (2008). Robust and precise eye detection based on locally selective projection. In *International Conference on Pattern Recognition*, pages 1–4.