Real-Time Head Pose Estimation on Mobile Platforms

Jianfeng Ren[1], Mohammad Rahman[1], Nasser Kehtarnavaz[1], and Leonardo Estevez[2]
[1]Department of Electrical Engineering, University of Texas at Dallas
[2]Wireless Terminal Business Unit, Texas Instruments

ABSTRACT

Many computer vision applications such as augmented reality require head pose estimation. As far as the real-time implementation of head pose estimation on relatively resource limited mobile platforms is concerned, it is required to satisfy real-time constraints while maintaining reasonable head pose estimation accuracy. The introduced head pose estimation approach in this paper is an attempt to meet this objective. The approach consists of the following components: Viola-Jones face detection, color-based face tracking using an online calibration procedure, and head pose estimation using Hu moment features and Fisher linear discriminant. Experimental results running on an actual mobile device are reported exhibiting both the real-time and accuracy aspects of the developed approach.

Keywords: Head pose estimation, mobile platform, real-time implementation, face detection

1. INTRODUCTION

Many computer vision applications require a face pose estimation module; for example, identifying the head gesture during conversation [1], designing a smart room that monitors its occupants' activities [2], deploying a driver assisted system [3]. Recently due to the growth in the use of mobile devices, face pose estimation on mobile devices has become of interest. Normally, the real-time implementation of head pose estimation algorithms is reported on PC platforms with relatively powerful processors and large memory sizes. The real-time software implementation of head pose estimation on mobile platforms without using any dedicated co-processor poses its own challenges.

Although many face pose estimation algorithms have been introduced in the literature, most suffer from one or more of these limitations: predefined assumptions about the environment, high computational complexity and low accuracy. Recently, Murphy-Chutorian et al. [4] presented a survey on various face pose estimation algorithms and pointed out that further improvements need to be made to get a robust real-time pose estimation system. When it comes to relatively resource limited mobile platforms as compared to PC platforms, it becomes more challenging to meet the real-time aspect while maintaining reasonable head pose estimation accuracy. This paper presents a robust real-time head pose estimation algorithm by using the Hu moment shape features [5] in an online training manner to classify a face pose into one of the following five poses: center/up/down/left/right.

The remaining part of the paper is organized as follows: Section 2 includes an overview of the existing head pose estimation algorithms. The introduced real-time head pose estimation algorithm is then presented in section 3. Real-time experimental results on a mobile platform are then provided in section 4 and finally the conclusions are stated in section 5.

2. OVERVIEW OF EXISTING HEAD POSE ESTIMATION ALGORITHMS

In the past few years, much research has been done on head pose estimation. The existing approaches can be categorized into two general categories of appearance-based and model-based approaches. The interested reader is referred to [4] for details on these approaches.

Head pose estimation is normally done after face detection. Appearance-based approaches consider head pose estimation as a multi-class classification problem. The orientation of detected faces gets classified into several different poses. These approaches use the detected/tracked faces as the input, and extract various features such as Gabor-wavelet and Hu moments. A classification is then applied. Some of the commonly used classifiers include support vector machines, neural networks, and Adaboost cascade classifiers. Generally, the training is performed offline and the recall is done online.

Model-based approaches mainly use face features such as the mouth, eyes and/or nose to determine the right pose. Even though many of these features are simple, the difficulty lies in detecting these features with high precision and accuracy, in particular when faces appear small in captured images.

In this paper, the emphasis is placed on head pose estimation running in real-time on mobile devices. The main attribute of the developed solution is that it does not require any offline training. In what follows, various components of our head pose estimation system are discussed.

3. REAL-TIME HEAD POSE ESTIMATION SYSTEM

The developed real-time head pose estimation solution consists of four main components: Viola-Jones face detection for the front faces, online color calibration, color-based face tracking, and finally online head pose estimation.

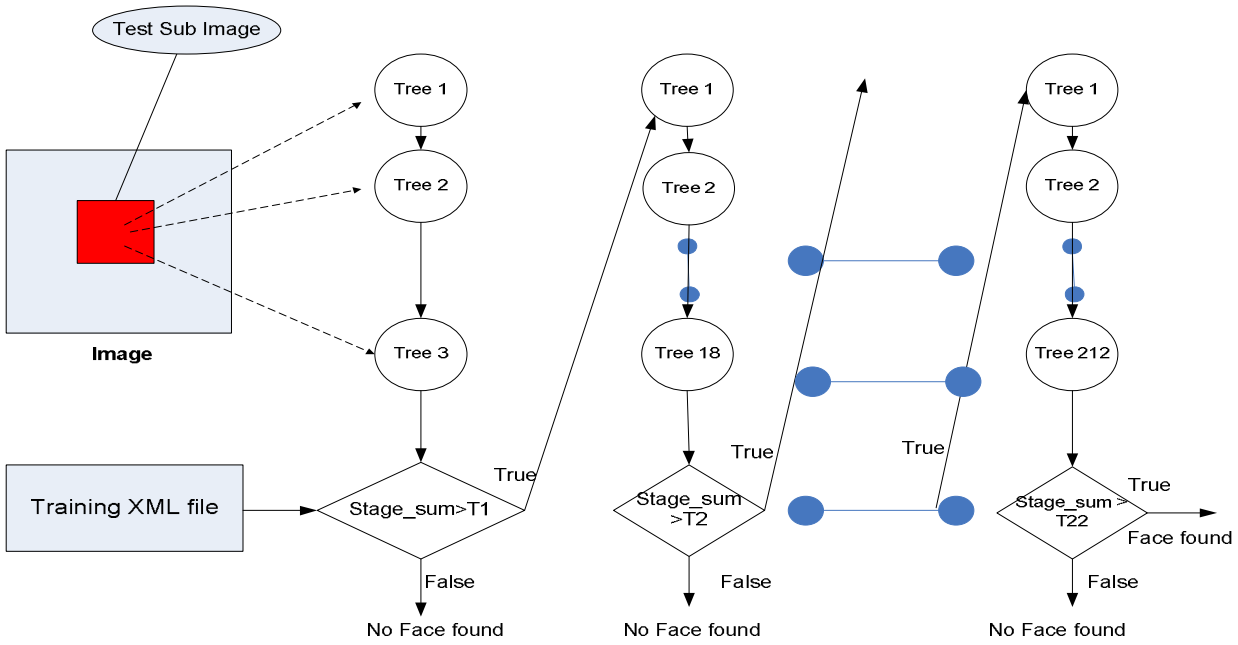

Fig. 1: Face detection algorithm based on Viola-Jones object detection.

3.1 Viola-Jones face detection

The developed head pose estimation system requires face detection first. Among the face-based detection algorithms, the one based on the Viola-Jones object detection approach has been shown to be most robust to environmental lighting changes [6-8]. Let us briefly state the real-time implementation of the Viola-Jones face detection on mobile platforms which we previously reported in [9].

For detection, a so called integral image for the entire image frame is computed. Then, each subimage with different positions and sizes is tested against all trees/stages in the classifier. Figure 1 provides an overview illustration of the algorithm. First, the available classifier parameters are read into one data structure such as a binary tree or an array. In the implementation reported here, we used the classifier parameters for frontal view faces. It should be mentioned that for profile faces or other face orientations, the corresponding classifier parameters can be used. The classifier selected for frontal view faces consists of 22 stages with each stage comprising different numbers of trees ranging from 3 to 212. For each subimage to be examined, its corresponding features are computed. Viola and Jones proposed four different rectangular features within a subimage as shown in Fig. 2. During the training process, the number of rectangular features within one 24x24 block is about 18,000. After training, each tree does the comparison for one rectangular feature. Therefore, during each stage, each tree is applied to the subimage under testing. This will generate one value to be compared with a threshold of that tree. If the value is less than the threshold of that tree, the left value of the tree gets accumulated. Otherwise, the right value gets accumulated. For each stage, if the stage sum is less than the stage threshold (T# in the figure, where # indicates the number of stages), then the testing ends indicating that the tested subimage does not contain any face. Otherwise, it continues to go through all the trees/stages until the last one. If one subimage goes through all the stages and the final result is 1, this indicates the subimage is a face.

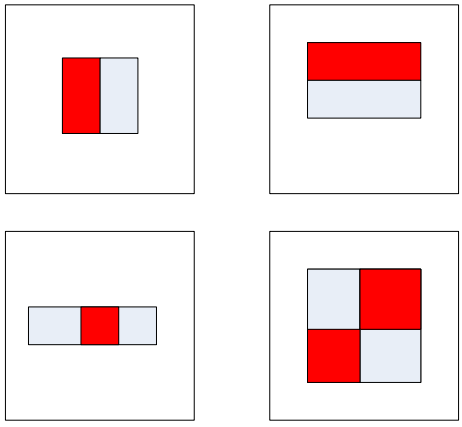

Fig. 2: Rectangular features shown for a tested subimage

This algorithm has been implemented in hardware in digital camera products. In [9], we presented a software-based

implementation of the Viola-Jones face detection algorithm on the TI OMAP3430 mobile device. We considered a number of optimization techniques including data reduction, search reduction and numerical reduction to be able to run the Viola-Jones algorithm in real-time. After incorporating all these optimizations, we were able to run the Viola-Jones face detection algorithm on this device every 90ms for VGA resolution video frames.

3.2 Online color calibration

After faces are detected, a skin color-based look-up table is used to identify the skin face area. We previously reported the details of our skin color-based face detection in [10].

Figure 3(a) shows a sample detected face area (outer box) and the area from which skin samples are collected (inner box). To separate skin pixels from non-skin pixels, the k-means clustering algorithm is applied to the inner area pixels. Figures 3(b) and 3(c) show the face area from which data samples are collected and the segmented skin area after clustering, respectively. A skin color model is then trained online and a lookup table is generated using the chrominance values of the skin pixels as exemplified in Fig. 3(d).

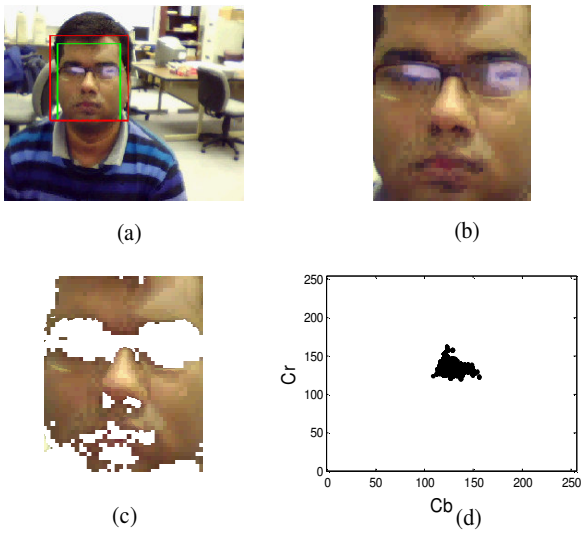

Fig 3: (a) Detected face area, (b) face area from which skin samples are collected, (c) segmented skin area after clustering, and (d) skin color cluster in Cb-Cr chrominance space.

3.3 Color tracking

In [10], we reported a robust color tracking algorithm to detect faces in video streams captured by the OMAP3430 mobile device.
Figure 4 shows the flow chart of our online color calibration and color tracking algorithm. At the beginning, few of initial frames are skipped because, in many camera systems, auto white balance and auto exposure have not reached a stable state. After this initial warm-up time, the feature based face detection algorithm is started and allowed to run for next five frames. The median of these five frames is obtained in order to avoid any ambiguity caused by false alarms and then the detected face area is used for the online calibration of the skin color model.

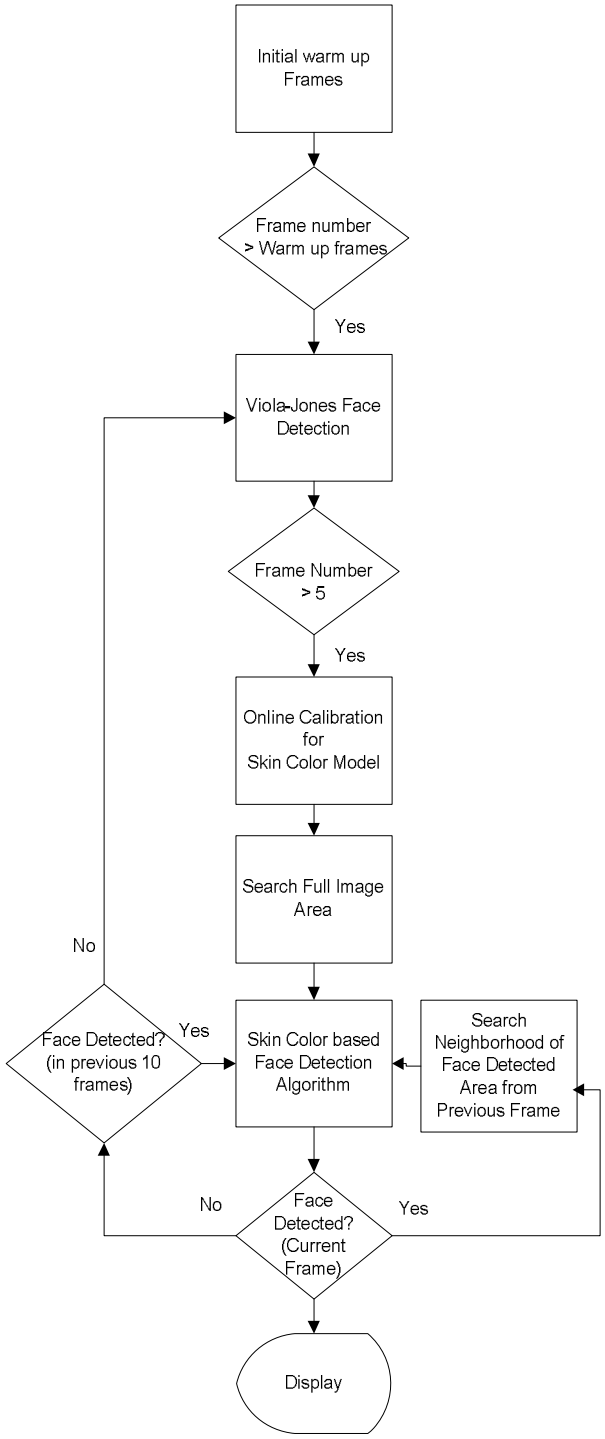

Fig. 4: Components of our hybrid face detection algorithm.

As soon as training is finished, the skin color based algorithm is started. If a face is detected, then the search region for next frames is arranged to be in a neighborhood area around the detected face. The full image area is used again in a next key frame or if the algorithm fails to find any face in a previous frame. This stabilizes occasional abrupt changes in the face position or when a new face enters the image. If the skin color based algorithm fails to detect any face for 10 consecutive frames, then the Viola-Jones algorithm is executed again to recalibrate the skin color model assuming that the lighting condition has changed or a new person has entered the frame.

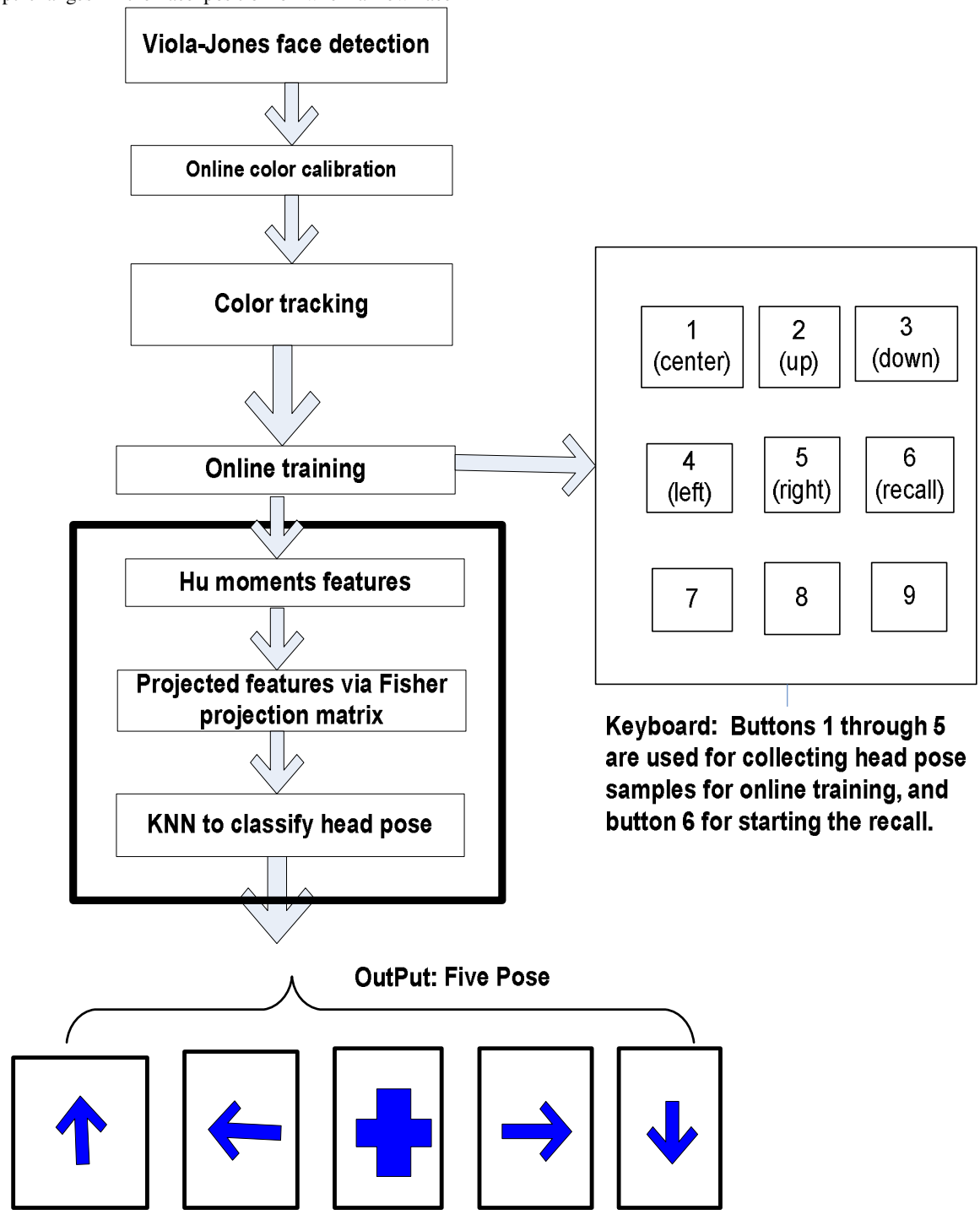

Fig 5: Components of the developed real-time head pose estimation system.

3.4 Online training

As illustrated in Fig. 5, in our approach, an online training is carried out first by collecting five different poses of a subject whose face poses are to be identified. Hu moments from each pose are captured by pushing the keyboard buttons 1 through 5 of the mobile device. During this online training procedure, feature vectors are obtained by projecting Hu moments via the Fisher linear discriminant matrix [11]. The k-nearest neighbor (KNN) algorithm is then applied to provide representative feature vectors for each pose. The recall gets started by pushing the button 6.

Let $\omega_1, \omega_2, \cdots, \omega_L$ and $N_1, N_2, \cdots, N_L$ denote the pose classes and the number of sample images in each class, respectively. Let $M_1, M_2, \cdots, M_L$ and M denote the class means and the overall mean, respectively. The within-class and between-class scatter matrices are computed as follows:

$$\Sigma_w = \sum_{i=1}^{L} \mathrm{E}\{(y - M_i)(y - M_i)^t\} \quad (1)$$

$$\Sigma_b = \sum_{i=1}^{L} \mathrm{E}\{(M_i - M)(M_i - M)^t\} \quad (2)$$

Fisher linear discriminant provides the projection matrix Ψ that maximizes $\Psi^t \Sigma_b \Psi \Big/ \Psi^t \Sigma_w \Psi$. This ratio is maximized when Ψ consists of the eigenvectors of the matrix $\Sigma_w^{-1}\Sigma_b$. The projected features via the projection matrix are then used during the recall phase.

4. EXPERIMENTAL RESULTS

In this section, the above computationally efficient head pose estimation algorithm is implemented on the OMAP mobile platform. We selected this platform since 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. Figure 6 and Figure 7 show a snapshot of the head pose estimation algorithm running in real-time on the PC and on the OMAP3430 mobile device.

The developed head pose estimation algorithm achieved more than 90 percent correct pose estimation when considering individual frames as part of live video streams running on the OMAP3430 platform. The outcome of a typical test experiment for three subjects appears in Table 1 based on individual frames in the corresponding video stream. As can be seen from this table, an average estimation accuracy of 94.1% was resulted. A sample confusion matrix based on individual frames of a video stream is shown in Table 2 exhibiting the origins of the misclassified cases.

Table 1: Head pose estimation accuracy

Subject	Detection Rate
Subject 1	91.2%
Subject 2	96.8%
Subject 3	94.2%
Average	94.1%

Table 2: Sample confusion matrix

	Center	Up	Down	Left	Right
Center	74	9	25	0	0
Up	3	69	0	0	0
Down	0	0	33	0	0
Left	0	0	0	100	0
Right	0	0	0	0	127

It is evident from Table 2 that the misclassification is primarily caused between the center-down and center poses. The reason for this is that the captured face image still retains most of the frontal facial features when tilting the head down. This problem was addressed by using majority voting. That is to say a time or moving window was used and the classification outcome was considered for a number of consecutive frames. Then, the pose with the majority number of frames was selected. This majority voting approach led to 100% correct pose estimation on live video streams.

The breakdown of the processing time for each part of the head pose estimation process is as follows: Viola-Jones face detection 90ms, online calibration (worst case 250ms, depends on detected face size), color-based face tracking 70ms and head pose estimation 10ms. These times are listed in Table 3. It should be noted that the Viola-Jones face detection and the on-line calibration are only run for the first few frames, which is to say after an initial warm-up time, the color tracking takes over.

Table 3: Computational breakdown of head pose estimation components during recall

Component	Processing Time
Viola-Jones face detection	90ms
Online calibration	At most 250ms, depends on face size
Color tracking	50ms for VGA resolution
Head Pose estimation	10ms

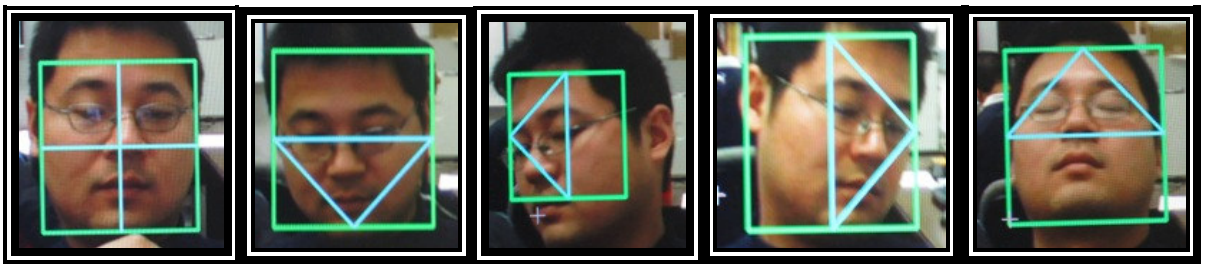

Fig 6: Snapshot of five head poses for a video stream running on PC platform.

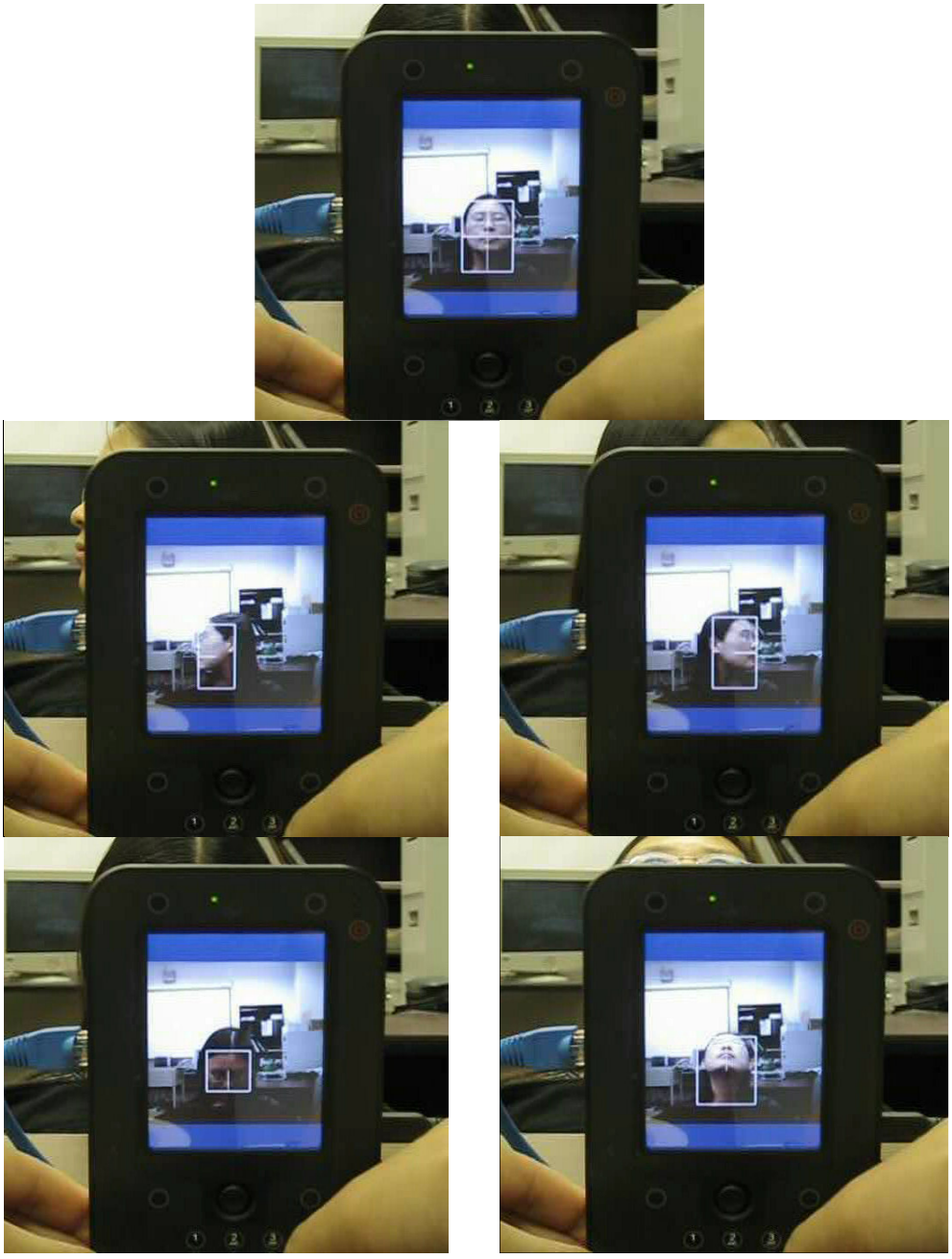

Fig 7: Snapshot of five head poses running in real-time on mobile platform (OMAP3430).

5. CONCLUSION

In this paper, a robust head pose estimation approach for real-time deployment on mobile devices is presented. The developed solution consists of three parts: (1) Viola-Jones face detection, (2) color tracking based on an online calibration procedure, and (3) a computationally efficient head pose estimation algorithm. An actual implementation on a mobile device is performed demonstrating both the robustness and real-time aspects of the introduced solution.

6. ACKNOWLEDGEMENT

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

7. REFERENCES

[1] M. Trivedi, "Human movement capture and analysis in intelligent environments," **Machine Vision and Applications**, vol. 14, no. 4, pp. 215–217, 2003.

[2] S. Baker, I. Matthews, J. Xiao, R. Gross, T. Kanade, and T. Ishikawa, "Real-time non-rigid driver head tracking for driver mental state estimation," **Proceedings of 11th World Congress Intelligent Transportation Systems**, Oct 2004.

[3] S. Fujie, Y. Ejiri, K. Nakajima, Y. Matsusaka, and T. Kobayashi, "A conversation robot using head gesture recognition as para-linguistic information," **Proceedings of 13th IEEE International Workshop on Robot and Human Communication**, pp.159-164, Sep 2004.

[4] E. Murphy-Chutorian and M. Trivedi "Head pose estimation in computer vision: A survey," **IEEE Trans. on PAMI**, vol. 31, no. 4, pp. 607-626, Feb 2009.

[5] R. Sivaramakrishna and N. Shashidharf, "Hu's moment invariants: how invariant are they under skew and perspective transformations?" **Proceedings of IEEE Conference on Communications, Power and Computing**, pp.292 – 295, May 1997.

[6] P. Viola and M. Jones, "Rapid Object Detection Using a Boosted Cascade of Simple Features," **Proceedings of IEEE CVPR**, vol. 1, pp. 511-518, April 2001.

[7] OpenCV[online] http://www.intel.com/technology/computing/opencv/overview.htm

[8] X. Tang, Z. Ou T. Su and P. Zhao, " Cascade AdaBoost Classifiers with stage features optimization for cellular Phone embedded face detection system" **Lecture Notes in Computer Science**, vol. 3612, Springer, 2005.

[9] 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**, pp. 1-4, Oct 2009.

[10] M. Rahman, J. Ren, N. Kehtarnavaz, "Real-time implementation of robust face detection on mobile platform", **Proceedings of IEEE ICASSP Conference**, vol.1, pp. 1353-1356, April 2009.

[11] C. Liu, H. Wechsler, "Gabor Feature Based Classification using the enhanced fisher linear discriminant model for face recognition", **IEEE Transactions on Image Processing**, vol. 11, no.4, pp. 467-476, April 2002.