A REAL-TIME FACE TRACKER FOR COLOR VIDEO

S. Spors and R. Rabenstein

Telecommunications Laboratory
University of Erlangen-Nuremberg
Cauerstrasse 7, 91058 Erlangen, Germany
E-mail: {spors, rabe}@LNT.de

ABSTRACT

This paper presents a face localization and tracking algorithm which is based upon skin color detection and principle component analysis (PCA) based eye localization. Skin color segmentation is performed using statistical models for human skin color. The skin color segmentation task results in a mask marking the skin color regions in the actual frame, which is further used to compute the position and size of the dominant facial region utilizing a robust statistics-based localization method. To improve the results of skin color segmentation a foreground/background segmentation and an adaptive background update scheme were added. Additionally the derived face position is tracked with an Kalman filter. To overcome the problem of skin color ambiguity an eye detection algorithm based upon the principle component analysis (PCA) is presented.

1. INTRODUCTION

The processing of face images receives more and more interest in the field of image analysis. The task of facial image analysis includes the face localization, the recognition of human faces and the analysis of mimics or facial expressions. Face localization is needed as preprocessing step for many applications, including such as object-orientated image coding, security control, expression recognition and intelligent man-machine interaction. Various methods for face detection were published in the past. The methods can roughly be diverted into four classes, according to [1]: *Knowledge-Based Methods* are rule-based approaches which try to model intuitive knowledge of facial features. In general these facial features have to be extracted by a preprocessing step first. *Feature Invariant Methods* utilize invariant (scale, orientation, lighting) features for face detection. One of the most frequent used feature for this purpose is skin color. *Template Matching Methods* use manually defined patterns of the whole face or the facial features separately which are matched against the input image. *Appearance Based Methods* try to find the relevant characteristics of face images by machine learning from a training set. Often, face tracking systems exploit a single localization technique to locate and track the users face. While one modality is able to track a user under optimal conditions, they also have typical failures in unconstrained scenes.

To arrive at a more robust facetracker under real-time constraints, we present a system which combines feature invariant (skin color) and appearance based methods (eye detection). First, face detection and tracking is discussed. Then we show how eye detection and tracking can be performed. Next, the complete face tracking algorithm is presented. Finally, some results showing the performance of the algorithm are discussed.

2. FACE DETECTION AND TRACKING

Face localization is performed using the statistical properties of human skin color. Many recent publications have proven that human skin color is a powerful feature for face detection. To improve the robustness of color segmentation a foreground/background segmentation step is introduced before the color segmentation is carried out. Next the detected foreground pixels are processed further by skin color segmentation. The resulting mask is the basis for face localization. The derived center position and size of the face is then tracked with an Kalman filter. Skin color modeling and segmentation is performed using the YCrCb color space, that provides separation between the luminance (Y) and the chrominance (Cr,Cb) components.

2.1. Foreground segmentation

Because, in general, the background of a still image is not available, foreground segmentation can only be performed for an image sequence. Additionally it is necessary, that the background can be captured before the interacting person is entering the scene. This can simply be done in the startup sequence of face tracking.
Foreground segmentation is executed by computing the Euclidean distance between the captured background image and the actual image for each pixel:

$$e = \sqrt{(Y - Y_{ref})^2 + (Cr - Cr_{ref})^2 + (Cb - Cb_{ref})^2} \quad (1)$$

where the subscript 'ref' denotes the reference image. A pixel is expected to be in the foreground if its Euclidean distance e from the reference image exceeds a given threshold ϵ:

$$\text{pixel is } \begin{cases} \text{foreground}, & \text{if } \quad e \geq \epsilon \\ \text{background}, & \text{otherwise} \end{cases} \quad (2)$$

The results from foreground segmentation are stored in a binary mask marking the detected foreground pixels for further processing by skin color segmentation.

2.2. Skin color segmentation

Skin color segmentation uses statistical models to model the characteristics of human skin color. Among several models, the histogram based color model described in [2] was chosen for our algorithm.

2.2.1. Skin color models

Histogram color models utilize histograms to model the color distribution of a certain image class, discarding the spatial information. These models are therefore solely based upon the statistical color properties of the selected image class. The histogram model is built using a two-dimensional histogram with 32 bins in each dimension. The model is fed with data from a set of hand labeled training images. In our case two classes of pixels were considered: skin and non-skin pixels. Given skin and non-skin histograms, the histogram counts are converted into estimates for the discrete probability distributions $\hat{P}(CrCb|\text{skin})$ and $\hat{P}(CrCb|\text{non-skin})$ in the usual manner:

$$\hat{P}(CrCb|\text{skin}) = \frac{c_s[CrCb]}{T_s}, \tag{3a}$$

$$\hat{P}(CrCb|\text{non-skin}) = \frac{c_n[CrCb]}{T_n} \tag{3b}$$

where $c_s[CrCb]$, $c_n[CrCb]$ denote the pixel counts for a certain $CrCb$ color pair in the skin and non-skin histograms and T_s, T_n are the total pixel counts contained in the skin and non-skin histograms, respectively. The statistical investigations showed, that the human skin colors cluster in a small region of the color space and that there is a significant degree of separation between the skin and non-skin image classes. These results prove, that human skin color is a powerful feature for face detection.

2.2.2. Skin color segmentation

The color segmentation step classifies the pixels of an given input image into skin and non-skin pixels. Only the pixels that were identified as foreground pixels are processed further by skin color segmentation. The result is a binary mask, that marks the skin color areas in a given input image. This mask provides the basis for the face localization algorithm described in the next section. Using the skin and non-skin color histogram model $\hat{P}(CrCb|\text{skin})$ and $\hat{P}(CrCb|\text{non-skin})$ a skin pixel classifier can be built. The estimated conditional probability $\hat{P}(CrCb|\text{skin})$ cannot be used for skin detection directly. For skin detection, the conditional probability $\hat{P}(\text{skin}|CrCb)$ is needed, which gives us the probability that a given pixel belongs to a skin area, based on the color information. This conditional probability is the basis for skin color segmentation. A given pixel is classified as skin pixel, if the conditional probability $\hat{P}(\text{skin}|CrCb)$ is greater than a preselected threshold θ for the $CrCb$ color pair of this pixel:

$$\hat{P}(\text{skin}|CrCb) \geq \theta \tag{4}$$

Using the Bayes rule the conditional probability $\hat{P}(\text{skin}|CrCb)$ can be computed from the color histograms in the following way:

$$\hat{P}(\text{skin}|CrCb) = \frac{\hat{P}(CrCb|\text{skin})\,\hat{P}(\text{skin})}{\hat{P}(CrCb|\text{skin})\,\hat{P}(\text{skin}) + \hat{P}(CrCb|\text{non-skin})\,\hat{P}(\text{non-skin})} \tag{5}$$

where $\hat{P}(\text{skin})$ and $\hat{P}(\text{non-skin})$ are the prior probabilities for skin and non-skin. Although there is no knowledge of $P(\text{skin})$ for a given input image, a reasonable choice for the prior probability of skin is the ratio of the total skin pixels in the histogram model to the total of all the pixels in the training data $\hat{P}(\text{skin}) = T_s/(T_s + T_n)$. Since the skin and non-skin models are disjunct to each other, the prior probability $\hat{P}(\text{non-skin})$ can be computed from $\hat{P}(\text{non-skin}) = 1 - \hat{P}(\text{skin})$.

2.3. Face localization

The face localization is performed by a robust statistics based method as described in [3]. Starting point for the algorithm is the mask derived from the color segmentation performed on the input image as described in the previous sections. Based upon this mask two one-dimensional projected histograms along the x- and y-axis of the mask are computed. The center position and size of the dominant face in an input image is estimated based on the means and standard deviations of trimmed versions of the projected histograms.

2.4. Eye localization and tracking

2.4.1. Eye localization

The method used here is based on the principle component analysis (PCA) which is better known as eigenface analysis. PCA is mostly used for the localization and recognition of faces [4, 5]. The research in [6] shows that PCA also provides a powerful framework for locating eyes. The aim of the PCA is to find the relevant characteristics of eyes from a set of training images. The basic idea is to use a unitary transform which transforms a given input image into a lower dimensional space. In general, a unitary transform matrix $\mathbf{\Phi}$ is applied to a given vector $\mathbf{f}$

$$\mathbf{b} = \mathbf{\Phi}^T\,\mathbf{f} \tag{6}$$

resulting in the vector $\mathbf{b}$, containing the transform coefficients. The vector $\mathbf{f}$ can be obtained simply by rearranging the image luminance components $Y_{y,x}$ by row ordering. The unitary transform matrix is build using the Karhunen-Loève transformation (KLT) as described in e.g. [7]. According to the eigenfaces used for face detection, the vectors of the transform matrix $\mathbf{\Phi}$ are called eigeneyes. Figure 1 shows the first eight eigeneyes computed from a set of 20 training images.

Eye detection is performed utilizing the inverse transformation:

$$\mathbf{f} = \mathbf{\Phi}\,\mathbf{b} \tag{7}$$

The simplest method to search for the eyes in an given image is to browse through the whole image, compute a reconstruction $\mathbf{r}$ for each truncated input vector $\mathbf{u}$ and evaluate the reconstruction error ϵ. This reconstruction is computed using the forward transform (6) resulting in the transform coefficients $\mathbf{b}$ which are then used to compute the reconstruction $\mathbf{r}$ using the inverse transform (7). The basic idea behind this scheme is, that the eigeneye basis $\mathbf{\Phi}$ provides the best reconstruction results for eye like regions and thus minimal reconstruction errors. The best match between reconstruction $\mathbf{r}$ and truncated input vector $\mathbf{u}$, provided through the position of the minimal reconstruction error $\epsilon_{\min}$ is an eye candidate. The reconstruction error $\epsilon_{y,x}$ is defined as follows:

$$\epsilon_{y,x} = \frac{\|\mathbf{u}_{y,x} - \mathbf{r}_{y,x}\|^2}{\sqrt{\|\mathbf{u}_{y,x}\|^2\;\|\mathbf{r}_{y,x}\|^2}} \tag{8}$$

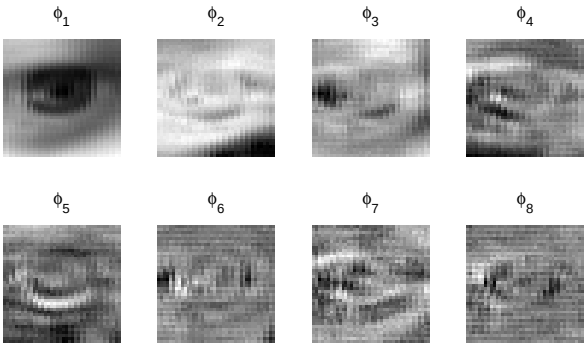

Fig. 1. Eigeneyes computed from a set of 20 training images

Although this algorithm is suitable for finding the desired feature, it is computationally quite unfeasible. There are three methods to reduce the computational complexity. The first method is not to use all of the derived eigeneyes for eye detection. This can be done the following way: Equation (7) can be rewritten as a sum

$$\mathbf{f}' = \sum_{i=1}^{L_a} \boldsymbol{\phi}_i \; b_i \tag{9}$$

where $\boldsymbol{\phi}_i$ contains the i-th column of $\boldsymbol{\Phi}$ and b_i denotes the i-th component of the vector $\mathbf{b}$. Thus an approximation of $\mathbf{f}$ can be obtained, if instead of N_a columns from $\boldsymbol{\Phi}$ we select only $L_a \leq N_a$ columns of $\boldsymbol{\Phi}$. The second method to lower the computational complexity is to use a downsampled search strategy. The eigenvectors $\boldsymbol{\phi}_i$ and the input image are subsampled after a suitable anti-aliasing filter. The search is then performed using the downsampled vectors resulting in a significant reduction of computational complexity. The third method utilizes the results from face detection to reduce the number of points that have to be searched in the input image. Only the non-skin classified points are searched within the detected facial region. Using these methods the eye localization can be performed in about 0.3 s on a SGI O2 workstation, while typically about 6000 positions per eye are searched.

2.4.2. Eye tracking

Although the computational effort for the PCA detection scheme is highly reduced through downsampling, it is not suitable for a real time implementation with high frame rates on the given hardware. To reduce the computational complexity further, the eye detection and tracking task is divided into two steps: First the eye is detected using the algorithms described in the previous sections. Once the position of the eyes is known, they are tracked using a luminance-adapted block matching technique, as described in [8]. This provides robust eye localization through PCA and fast tracking using block matching. The block matching is, like the PCA, performed on the luminance components of the image. In order to cope with temporal luminance and size changes of the eye, a reference eye pattern and the extracted eye region from the previous frame are used for matching the eye regions. The block distance $\eta_{y,x}$ is defined the following way

$$\eta_{y,x} = \frac{3}{4} \left\| \mathbf{p}_{\text{ref}} - \mathbf{u}_{y,x} \right\|^2 + \frac{1}{4} \left\| \mathbf{p}_{\text{tmp}} - \mathbf{u}_{y,x} \right\|^2 \tag{10}$$

where $\mathbf{u}_{y,x}$ is a shifted block from the current frame , $\mathbf{p}_{\text{tmp}}$ denotes the extracted eye region from the previous frame and $\mathbf{p}_{\text{ref}}$ the reference eye pattern. The best match is found by searching for the minimum block distance $\eta_{\min}$ in the facial area. After the matching process, the luminance of each reference eye pattern $\mathbf{p}_{\text{ref}}$ is adjusted by the difference of the means of the detected eye region in the current frame and the reference pattern in order to compensate temporal changes in brightness

$$\mathbf{p}_{\text{ref}}[k+1] = \mathbf{p}_{\text{ref}}[k] + (\mu_{\mathbf{p}_{\text{ref}}[k]} - \mu_{\mathbf{u}_{\min}[k]}) \tag{11}$$

where $\mathbf{u}_{\min}[k]$ denotes the region with the best block match and $\mu_{\mathbf{p}_{\text{ref}}[k]}, \mu_{\mathbf{u}_{\min}[k]}$ denote the appropriate block mean values.

2.5. Face tracking

The face localization algorithm from the previous section provides the center position and the size of a detected face in the input frame. Because of noise in the image acquisition, the position and size located by the localization algorithms are disturbed by noise. As a result of these noise terms the localization result plotted on the screen appears restless. Seeing these problems in the context of state estimation, the measurements can be improved by applying a state estimation technique. State estimation utilizes a prior knowledge of the measurements and the system dynamics to provide a reliable estimate of the true system state. In the context of face localization the state is identified as the center position or the size of the object tracked. Among various state estimation techniques the Kalman filter is used for tracking purposes in our system. The Kalman filter easily takes into account many important factors such as sequential time updates, measurement accuracy and target maneuver models. The linear Kalman filter uses linear systems, disturbed by noise terms to model the system dynamics as well as the measurements. The derivation of the Kalman filter equations can be found in e.g. [9].
To model the system dynamics a motion model for the tracked object is needed. The linear motion model used here implies that the object moves with constant speed with respect to the Cartesian coordinate system used.

3. REAL-TIME IMPLEMENTATION

Figure 2 shows a block diagram of the complete face tracking algorithm. It is implemented on an SGI O2 workstation, providing real-time operation with 25 frames per second. The implementation is based upon the IRIX '*Video Library*' utilizing the O2 standard video hardware. Once a new frame is captured, its color information is subsampled to reduce the data that has to be processed. A subsampling factor of four was proven to be suitable. As next step, the foreground segmentation is carried out. To cope with changes in the background image captured at the beginning of the tracking session, the adaptive background scheme from [8] was added. Skin color segmentation is then performed on the detected foreground pixels. Based upon the skin color segmentation mask, the position and size of the dominant skin color region in the input frame is computed. Simultaneously the users eyes are located and tracked. As final step, the center position and size of the face is tracked with a Kalman filter and the results are plotted into the live video capture on the screen in real-time.

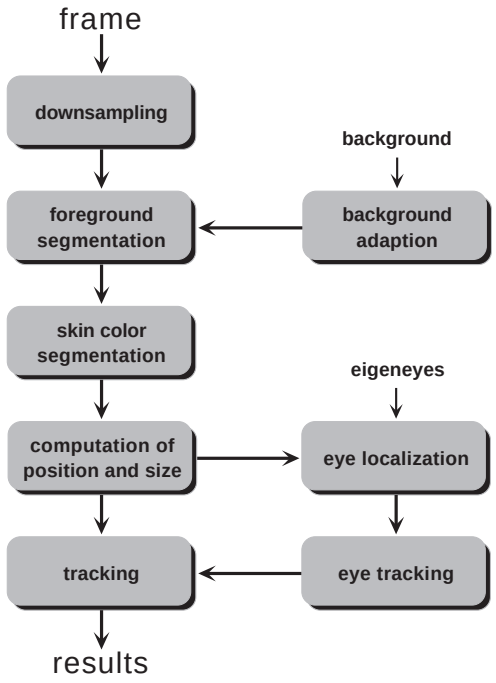

Fig. 2. Block diagram of the facetracking algorithm

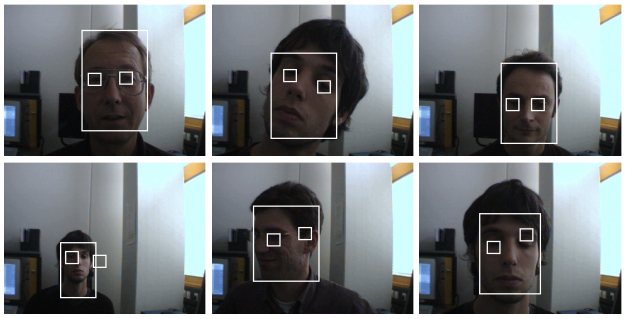

Fig. 3. Results of the facetracking algorithm

4. RESULTS

Figure 3 shows some snapshots taken from tracking sessions. The top row shows successful face and eye detection results. The face positions and sizes of different users were detected successful in all cases. The eyes of the users were localized, even wearing glasses or in tilted position. The bottom row shows some failure modes of the eye localization system. The left image is a result of strong size variation between the training set and the input image. In the center picture the eyes of the user are partially occluded and thus eye localization fails. The right image shows an inaccurate eye localization result.

Besides these more qualitative experiments we also conducted some experiments on the position accuracy of the algorithm. For this purpose experiments using a model railway with non-ambiguous color have been carried out. The skin color based localization scheme, trained on the color of the model railway, was used to localize and track the model railway. The results show that the algorithm provides a high position accuracy when tracking objects with non-ambiguous color.

5. CONCLUSION

This paper presented a face localization and tracking algorithm which is based upon skin color detection and PCA based eye localization. Skin color segmentation is performed using statistical models for human skin color. To this end, a set of hand segmented skin color training images is needed to train the skin color models. Although the presented skin color segmentation and face localization scheme is feasible under controlled lighting conditions, one major problem are changing and non-uniform lighting conditions. This is due to the still unsolved color constancy problem on image acquisition through video hardware. Eye localization is realized by matching an reconstructed image against the original input image. The reconstructed image is computed utilizing the eigeneye basis and therefore the reconstruction works best for eye like regions. The eye localization algorithm works robust and reliable in typical scenes. Once the face and eye positions are localized they are tracked by the algorithm. The center position and size of the facial area is tracked using a linear Kalman filter. The object motion model used for the Kalman filter assumes linear motion of the tracked object. The use of more complex motion models can further improve the quality of object tracking. Eye tracking is performed using a luminance-adapted block matching technique. In general this algorithm works robust and efficient.

6. REFERENCES

[1] D. Kriegman M.-H. Yang and N. Ahuja, "Detecting faces in images: A survey," *IEEE Transactions on Pattern analysis and Machine intelligence*, 2000, To be published.

[2] M. J. Jones and J. M. Rehg, "Statistical color models with application to skin detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, 1998, vol. 1, pp. 274–280.

[3] R. J. Quian, M. I. Sezan, and K. E. Matthews, "A robust real-time face tracking algorithm," in *Proceedings of the 1998 IEEE International Conference on Image Processing*, 1998, vol. 1, pp. 131–135.

[4] M. Kirby and L. Sirovich, "Application of the Karhuenen - Loève procedure for the characterization of human faces," in *IEEE Transactions on Pattern analysis and Machine intelligence*, 1990, vol. 12, pp. 103–108.

[5] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of Cognitive Neuroscience*, vol. 3, no. 1, pp. 71–86, 1991.

[6] K. Talmi and J. Liu, "Eye and gaze tracking for visually controlled interactive stereoscopic displays," *Image Communication*, vol. 14, no. 10, pp. 799–810, 1999.

[7] A.K. Jain, *Fundamentals of digital image processing*, Prentice-Hall, 1989.

[8] K. Talmi L. P. Bala and J. Liu, "Automatic detection and tracking of faces and facial features in video sequences," *Picture Coding Symposium 1997, 10-12 September 1997, Berlin*, 1997.

[9] Robert Grover Brown and Patrick Y.C. Hwang, *Introduction to random signals and applied Kalman filtering*, John Wiley & Sons, 1997.