Facial feature detection using neural networks

Axel Christian Varchmin, Robert Rae and Helge Ritter

Neuroinformatics Group, Faculty of Technology,
University of Bielefeld, P.O. Box 100 131,
D-33501 Bielefeld
{iavarchm,robrae}@techfak.uni-bielefeld.de

Abstract. Many human–machine interfaces based on face gestures are strongly user-dependent. We want to overcome this limitation by using common facial features like eyes, nose and mouth for gaze recognition. In a first step an adaptive color histogram segmentation method roughly determines the region of interest including the user's face. Within this region we then use a hierarchical recognition approach to detect the facial features. Our system is based on a what–where neural network architecture and allows a fast and robust recognition rate. In the future we intend to use the conspicuous features for estimation of gaze directions.

1 Introduction

Evaluation of head and eye actions can significantly contribute to the realization of more powerful and easier to use man–machine interfaces. Imagine the advantages if you can watch an object from another perspective only by changing your viewing position in front of the monitor. However, existing eye tracking hardware is unpleasant to wear and very expensive because it is designed for highly precise results [1]. We are more interested in estimating the approximate gaze direction of the user for controlling basic graphical actions like zooming, rotating or transforming an object.

We aim at developing a computer vision system which determines the gaze direction depending on the head and eye orientation. Previous work on this subject [2] has shown that this is still a challenging task because of changing lighting conditions, different users and limited computing resources. Authors working on the same subject report these problems, too (e. g. [1, 3, 4, 5]). While our earlier system was strongly user-dependent we now want to overcome this limitation by using common facial features like eyes, nose and mouth. In the future we want to use this information in addition to the actual eye direction for gaze recognition.

2 System Architecture

The system consists of a SGI workstation (High Impact) and a frame grabber (Indigo^2 Video). The camera (SONY EVI-D31) is installed below the screen

pointing to the users face (see fig. 1). The advantage of this setup is a good visibility of the eyes and the nose.

We use multiple stages for the recognition task. The first step identifies the location of the users face (Sec. 2.1). Within the face region we then seek for conspicuous facial features such as nose, mouth and eyes (Sec. 2.2). A last step will use these features to compute head orientation and to estimate gaze direction by a detailed analysis of the eye region but is not implemented yet.

2.1 The Face-tracker

The face is located and tracked by combining a color segmented image with movement information. The active SONY camera allows us to compensate larger head movements by a built-in tracking algorithm.

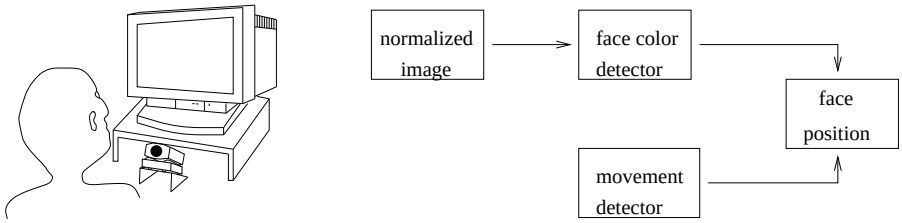

Fig. 1. L: Hardware setup; R: Face-tracking system

To locate the face region we first intensify all skin-colored pixels. The required color-classification is carried out by storing the color values that correspond to different faces in a general color-map (for details see [3]). During a session this general color-map is automatically adapted to the actual color region near the estimated nose position. This results in a color-map with increased values for colors arising with high frequency in the actual face under the actual lighting.

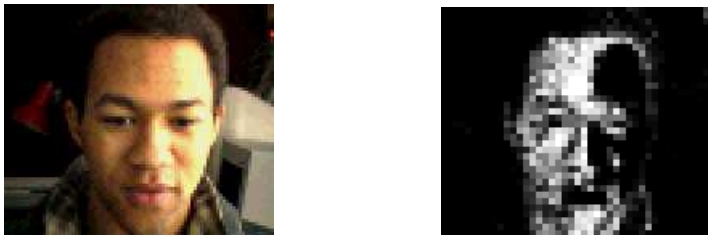

Fig. 2. Original and intensified image

Fig. 2 shows an example of the image intensification by the adapted color-map. Bright pixels correspond to high values in the color map and represent skin-colored regions. In general there is some noise in the image or there are objects in the background having face color, too. Therefore, we use movement information obtained by difference images to verify the face position estimated by the intensifier.

2.2 Facial Feature Detection

Since human head postures can differ extremely, we want to use features that can be reliably identified over a wide range of viewing angles. Suitable facial features thus are nose, mouth and eyes.

In many systems a heuristical method is used to detect these interesting points (e. g. the twinkle approach [5] for the eyes). Sometimes the results of these methods do not make sense. Therefore, we use several neural nets of the Local Linear Map type to find the correct position of the eyes and nose and further "what"-nets to verify the positions found by the first "where"-nets.

Image Feature Extraction. The image feature extraction should deliver a low dimensional and "easy to classify" vector, which means robustness against affine transformations, changes in illumination and noise. In earlier systems we applied a set of Gabor filters locally to the image [6].

The feature extraction method used in this work is motivated by the well known Eigenface approach [7, 8]. Instead of using Gabor filters we apply a set of filter kernels constructed by eigenvectors from typical facial features.

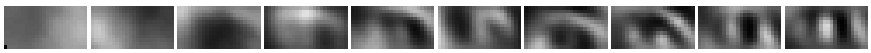

Fig. 3. The first ten "eigeneyes" ("Fine set")

To detect the eyes, we calculate the first ten "eigeneye" vectors from various images with eyes at different locations (fig. 3). For more robustness we use two sets of "eigeneyes" where the eye position varies in a smaller (fine set) and bigger region (rough set). We choose a similar feature extraction approach for the nose detection using a 2×2 grid with seven "eigennoses" (obtained from various linearly scaled face regions[1]) at each grid point. This results in a 28-dimensional feature vector.

So part of the feature classification task is already done by the feature extraction. We assume that the properties of the feature vectors lead to a high specifity and easier classification.

The Local Linear Map. For classification we use the Local Linear Map-network which is suitable for nonlinear function approximation. The nonlinear function is approximated by a set of locally valid linear mappings, for further details see e. g. [9]. In this work for a given input vector only the best match or "winner" node contributes to the output vector.

For the L-dimensional input vector $\mathbf{x}$ the output vector $\mathbf{y}$ of the net is given by:

$$\mathbf{y}_k = \mathbf{w}_k^{(out)} + A_k \left(\mathbf{x} - \mathbf{w}_k^{(in)} \right) \quad , \tag{1}$$

[1] All pixel values transformed to an interval [0..255]

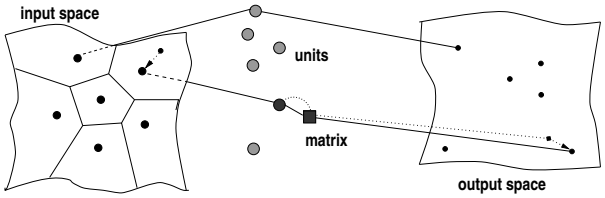

Fig. 4. Architecture of the LLM-network

where k is the winner node ($k \in \{1 \ldots N\}, N =$ number of nodes). $\mathbf{w}_k^{(in)}$ and $\mathbf{w}_k^{(out)}$ are the input and output weight vectors of node k, respectively. A_k is a $M \times L$-(Jacobian-)matrix (M is the dimensionality of the output space) which belongs to the node k. The best match node k is determined by:

$$k = \min_j(\|\mathbf{x} - \mathbf{w}_j^{(in)}\|), \qquad j = 1, \ldots, N \quad . \tag{2}$$

The LLM-net is trained in a supervised learning scheme (with $\alpha = 1, 2, \ldots, T$ examples) using the following adaption equations:

$$\begin{aligned} \Delta\mathbf{w}_k^{(in)} &= \epsilon_1 \left(\mathbf{x}^{(\alpha)} - \mathbf{w}_k^{(in)}\right) \\ \Delta\mathbf{w}_k^{(out)} &= \epsilon_2 \left(\mathbf{y}^{(\alpha)} - \mathbf{y}^{(net)}\right) + A_k \cdot \Delta\mathbf{w}_k^{(in)} \\ \Delta A_k &= \epsilon_3 \left(d_k^2\right)^{-1} \left(\mathbf{y}^{(\alpha)} - \mathbf{y}^{(net)}\right) \left(\mathbf{x}^{(\alpha)} - \mathbf{w}_k^{(in)}\right)^T \quad , \end{aligned} \tag{3}$$

where ϵ_1, ϵ_2 and $\epsilon_3 \in [0, 1]$ are adaption step-sizes.

The Facial Feature Finder. It turned out that the easiest feature to find is the nose. Heuristically we found that the nose holes are much brighter in the red-channel than all other pixels of the face. By combining this information with the knowledge that the nose is nearly in the center of the cluster, we achieve a probability $> 90\%$ to correctly determine the nose hole positions.

In case of a mismatch we determine the nose position by using an iterative correction scheme based on three neural networks. These networks are trained to estimate the position of the nose on different sized image regions. Starting from the center of the face, the first network estimates a rough correction using the 28-dimensional feature vector described above. This new point is used by the second network which further improves the position. The third net finally determines the nose position.

This position is verified by a special trained "nose recognition" neural net. The input features to this net are scalar products of the estimated nose region with 12 "eigennoses". Additionally we look for typical intensity characteristics of a nose. If both modules accept the position as a nose point further features are searched for, otherwise the image is rejected.

Using the nose position as a "landmark" greatly simplifies the detection of the remaining facial features "mouth" and "eyes" since we now can focus processing on a rather small subregion of the entire image.

The next features - mouth corners - are searched in a subsampled region of interest (16×7) below the nose. Simple "end stop" filter kernels are convolved

with this region obtained from the green channel (see fig. 5). The two highest values in the result image are taken as the mouth endpoints.

Fig. 5. L: End stop cells (filter kernels 6×4); R: Typical mouth region

The eye positions estimated from the mouth and nose points are refined by three further neural networks. These networks use a feature vector from 20 "eigeneyes" (see fig. 3) for an improvement from the initial estimation to the exact eye center.

The knowledge of the feature positions finally can be used for a validation of the "face cluster" and a first estimate for the head orientation.

3 Results and Discussion

By combining the eigenfeature approach with the described architecture of a substantial number of specialized recognition networks whose results are verified by an additional stage, we achieve a rather high robustness and recognition accuracy at the level of the entire system. Fig. 6 demonstrates the important effect of the correction stage for the estimation of the nose position (left) and of the eye position (right). As can be seen, the final target points are very accurately located on the respective facial features, although images of the depicted users had not been included in the training data sets used for the various networks in the system.

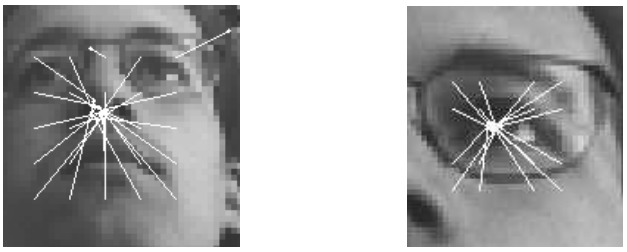

Fig. 6. Improvement by LLM-networks: Nose position (L); Eye position (R)

We have implemented the above processing stages in a system that runs at a rate of about 1 frame per second. Figs. 7a-c show the tracking of eye, nose and mouth features during a head movement sequence in front of a background containing high contrasts. Although we cannot yet present a quantitative statistical evaluation of the tracking accuracy, our experience so far indicates that the described system can track eye, mouth and nose features quite reliably and generalizes well to previously unseen users.

The implemented facial feature tracking system currently contains seven recognition networks. Although this may seem a fairly high number, evaluation of each network mapping is computationally rather undemanding (most

processing time goes into the frame grabbing and preprocessing). Even training can be done with rather moderate computational resources since the LLM networks can be trained significantly faster than, e.g., multilayer perceptrons.

The rather accurate tracking of facial features now provides robust and reliable low-dimensional features for estimating head orientation, e.g., by exploiting some geometric transformation [4] or by augmenting the system by a further identification network that is trained to learn the required transformation.

As a final, not yet implemented step, we will use the same techniques to analyze a small subregion around the estimated eye positions with a further recognition network to estimate the orientation of the eyes and thereby to achieve the final goal of obtaining an estimate of absolute gaze direction.

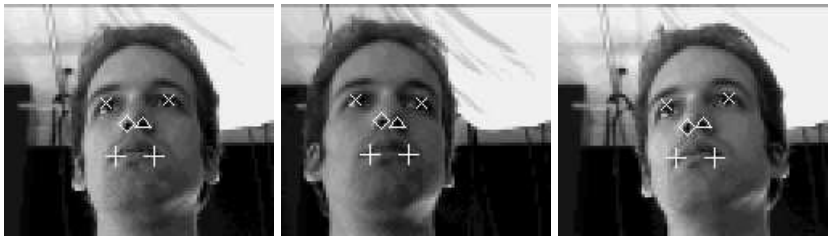

Fig. 7. Sequence of facial feature detection

References

[1] S. Baluja and D. Pomerleau: "Non-Intrusive Gaze Tracking Using Artificial Neural Networks". In J.D. Cowan, G. Tesauro and J. Alspector, editors, *Advances in Neural Information Processing Systems (NIPS) 6.*, Morgan Kaufmann Publishers, San Francisco, CA., 1994.

[2] R. Kubisch and H. Ritter: "Erkennung menschlicher Kopfhaltungen mittels künstlicher neuronaler Netze". In *Informatik Aktuell, Mustererkennung 1996, Ed. B. Jähne et. al., Springer-Verlag, Berlin, Heidelberg, 1996*, pages 109-117, 1996.

[3] B. Schiele and A. Waibel: "Gaze Tracking Based on Face-Color". In *Int. Workshop on Automatic Face- and Gesture-Recognition*, Zurich, Switzerland, June 1995.

[4] A. Gee and R. Cipolla: "Determining the gaze of faces in images". In *Image and Vision Computing Volume 12 Number 10 December 1994, Butterworth-Heinemann Ltd.*, pages 639-647.

[5] P. Ballard and G.C. Stockman: "Controlling a Computer via Facial Aspect". In *IEEE Trans. on Systems, Man, and Cybernetics, Vol. 25, No. 4, April 1995*, pages 669-677.

[6] A. Drees: "Visuelle Erkennung von Handstellungen mit neuronalen Netzen". PhD thesis, Universität Bielefeld, Technische Fakultät, September 1995.

[7] M. Turk and A. Pentland: "Eigenfaces for Recognition". In *Journal of Cognitive Neuroscience, Vol. 3 No. 1*, pages 71–86, 1997.

[8] A. Pentland, B. Moghaddam, T. Starner and M. Turk: "View-based and modular eigenspaces for face recognition". In *Proc. IEEE Computer Soc. Conf. on Computer Vision and Patt. Recog.*, 1994, pages 84–91.

[9] H. Ritter: "Learning with the Self-Organizing Map". In *Artificial Neural Networks, Vol. 1*, pages 379–384, 1991.

This article was processed using the LaTeX macro package with LLNCS style