Tracking and Learning Graphs and Pose on Image Sequences of Faces*

Thomas Maurer* and Christoph von der Malsburg*+

*Ruhr-Universität Bochum, Institut für Neuroinformatik, D–44780 Bochum, Germany
+University of Southern California, Dept. of Computer Science and Section for Neurobiology, Los Angeles, USA

Abstract

We demonstrate a system capable of tracking, in real world image sequences, landmarks such as eyes, mouth, or chin on a face. In the standard version, knowledge previously collected about faces is used for finding the landmarks in the first frame. In a second version, the system is able to track the face without any prior knowledge about faces and is thus applicable to other object classes. By using Gabor filters as visual features, and by both avoiding limiting assumptions and many parameters our tracking tool is simple and easy to use. As a first application the tracking results are used to estimate the pose of a face.

1 Introduction

In the last decade there has been much development in the area of face recognition: Systems are being built to track a person in a scene, extract the head and recognize the face. Especially recognition of frontal views has become robust against changes of expression [1] and shows good performance even on large databases with hundreds of images [2, 3]. Indeed, the first face recognition systems have already left the laboratory and are being deployed under realistic conditions [4]. But these successful systems need (at least approximately) frontal face images. When the head is rotated appreciably ($> 30°$) from the frontal view in any direction, the performance of all systems decreases dramatically. Although different attempts have been made to overcome this problem by learning some sort of transformation from one pose or view to another [5, 6, 7], none of those systems seems to have the capability of view-independent face recognition for practical applications. Nearly all systems work on static images taken from different views, and then apply time intensive methods like flow field computation (e.g., [6]) to find corresponding points in these images, or simply match corresponding points by hand. But as long as we don't have easy-to-use tools to feed a computer with the knowledge of how a face (or, for that matter, any object) looks like from all directions together with the information on point correspondence between views, there will be no chance to learn 3D representations. This is also true in the general case of 3D object recognition.

Consequently, if we want to achieve progress with 3D face and object recognition, we should provide computers with the same input that natural systems have: image sequences continuous in time. If image sequences are taken with sufficiently high frame rates, it should be possible to track the points over the whole sequence, there then being only small changes from frame to frame. In this way, the correspondence problem should be much easier to solve than in sparse static images. This is the strategy we will follow in this paper.

2 Brief description of the face finding system

As the basic visual feature we use a local image descriptor in the form of a vector called 'jet' [1]. Each component of a jet is the filter response of a Gabor wavelet of specific frequency and orientation, extracted at some point $\mathbf{x}$. We typically use wavelets of four different frequencies and eight different orientations, for a total of 32 complex components. Such a jet describes the area surrounding $\mathbf{x}$. A set of jets taken from different positions form a graph which describes an object in the image. To compare jets and graphs, similarity functions are defined: The normalized dot product of two jets yields their similarity, the sum over jet similarities being the similarity of graphs of identical geometry. Additionally, distortions of the grid can be taken into account to compute graph similarity [1]. Features and similarities defined in this way are robust against changes in illumination and contrast. In order to create an appropriate face graph for a gallery of 50–80 frontal faces of equal size, we place the nodes by hand at landmarks that we subjectively consider important to the task, e.g., at the center of the eyes or at the tip of the nose [2].

*Supported by the German Federal Ministry of Science and Technology and by the US Army Research Lab.

0-8186-7713-9/96 $5.00

Figure 1. Sequence of 50 frames of a face rotating from frontal view to right profile and back to frontal. Only frames 1, 10, 20, 30, 40, and 50 are shown, the nodes are placed and tracked completely automatically. The light spots on the glasses caused inaccuracies by shifting the eye nodes opposite to the movement direction, and the nodes on the hair did not have enough local structure to track their position properly.

There are typically 20–40 nodes forming a face graph. Such a gallery constitutes the general face knowledge for frontal faces. For a new frontal face of approximately the same size these nodes of the graph can be found automatically, even if this person is not in the gallery [2].

3 Tracking of face graphs

Given a sequence of images taken with a sufficient frame rate (>10 Hz) and showing the head of a human subject moving and rotating, the task is to track the landmarks as represented by the graph's nodes during the sequence. In order to acquire starting points for these nodes in the first frame, we told the subjects to initially look straight into the camera. Then, the process for finding landmarks in frontal faces described above [2] can be used to place the graph on the first image automatically. The nodes of this graph were chosen *ad hoc* and are not necessarily optimal for tracking; how to find optimal nodes will be described in the next section. Although we demonstrate our system on faces here, we don't want to be restricted to this case. Therefore, we avoid using any model knowledge or assumptions on the tracking device; only continuity in time (frame rate high enough) is needed. We do not make use of the relative movement of the nodes; they are tracked independently, and the edges of the graphs are for illustration only. To find the corresponding node positions in the new frame, only the jets extracted in the actual frame are used, i.e., the system has one single graph in memory which is matched on a new frame, then replaced, and so on. In this way we get a general tracking device, which can be further optimized for different applications by including additional constraints.

To compute the displacement of a single node in two consecutive frames we use a method developed for disparity estimation in stereo images, based on [8] and [9]. The strong variation of the phases of the complex filter responses is used explicitly to compute the displacement with subpixel accuracy [10]. By writing the response J_j to the jth Gabor filter in terms of amplitude a_j and phase ϕ_j a similarity function can be defined as

$$S(\mathbf{J}, \mathbf{J}', \mathbf{d}) = \frac{\sum_j a_j a'_j \cos(\phi_j - \phi'_j - \mathbf{d} \cdot \mathbf{k}_j)}{\sqrt{\sum_j a_j^2 \sum_j a_j'^2}}.$$

Let $\mathbf{J}$ be the jet at some position $\mathbf{x}$ in frame n, and $\mathbf{J}'$ the jet at the same position $\mathbf{x}$ in the next frame $n+1$, the displacement $\mathbf{d}$ of the corresponding point can be found by maximizing the similarity S with respect to $\mathbf{d}$, the $\mathbf{k}_j$ being the wavevectors associated with the filter generating J_j. Because the estimation of $\mathbf{d}$ is only precise for small displacements, i.e., large overlap of the Gabor jets, large displacement vectors are treated as a first estimate only and the process is iterated. In this way displacements up to half the wavelength of the kernel with the lowest frequency used can be computed (see [10] for details). For our Gabor kernels the maximal displacement is 6–7 pixels. As already mentioned in the introduction, a much larger range would help only in the special case of a purely translational movement, in all other cases larger displacements are associated with greater changes in the image, and then the corresponding node position might not be found anyway. But if fast frontoparallel motion should cause problems this could be easily repaired by including the assumption of continuity in the motion, i.e., by starting the computation of $\mathbf{d}_{n+1}$ not at $\mathbf{x}_n$, but at $(\mathbf{x}_n + \mathbf{d}_n)$.

So for all nodes of the graph in frame n the displacement vectors with respect to frame $n+1$ are computed, then a graph is created with its nodes at these new corresponding positions in the new frame, and all stored jets (which had been extracted in frame n) are replaced by those extracted at the corresponding node positions in frame $n+1$. But here we have a problem: Although the displacements have been determined as floats, the jets can be extracted at (integer) pixel positions only, resulting in a systematic rounding error. To compensate for this subpixel error $\triangle\mathbf{d}$, the phases of the

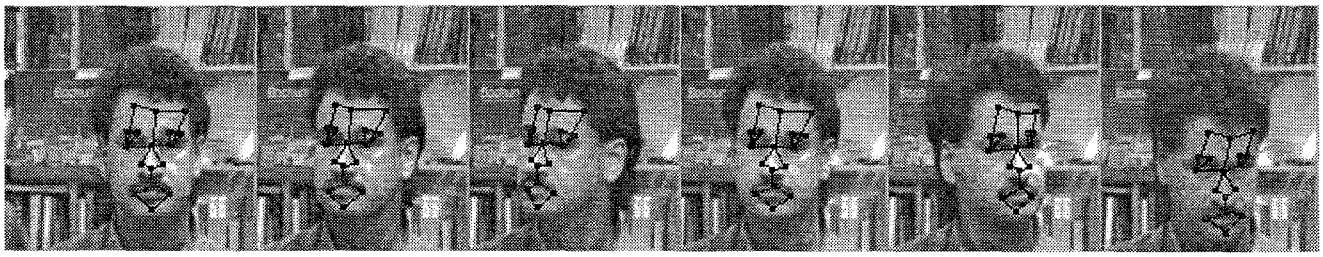

Figure 2. Sequence of 50 frames of a person looking in different directions, with highly structured background. Only the frames 1, 10, 20, 30, 40, and 50 are shown. Here the initial landmark finding misplaced the node on the left eye, and this position is held during the whole sequence.

complex Gabor filter responses must be shifted according to

$$\triangle\phi_j = \triangle\mathbf{d} \cdot \mathbf{k}_j \,,$$

then they will look as if they were extracted at the correct subpixel position. In this way Gabor jets can be tracked with subpixel accuracy without any further bookkeeping of rounding errors. This is an additional advantage of using Gabor jets in image processing; subpixel accuracy is a more difficult problem in most other image processing methods. Fig. 1 and 2 show graphs tracked over 50 frames without and with structured background.

A few remarks on computation time: Nearly the whole processor time per frame is needed for the wavelet convolution, ca. 4 sec on a SUN Sparc-20 (60 MHz) for a frame of 128 × 128 pixels. The average time per frame for the computation of the displacement vectors is less than 50 ms. So if we had the hardware to do the wavelet convolution in real time, the whole face tracking would work in real time, too.

4 Learning graphs

Given an image sequence, instead of starting with an initial graph in the first frame (placed automatically or, in the case of a new object, by hand), one can simply track *all* pixels from the first frame over a part of the sequence (more precisely, put a node on every pixel in the first frame and track all nodes) and try to create a graph from that subset of nodes which could be tracked well, i.e., those that appear to be attached to the surface of the moving object. A first attempt could be to track all nodes forth and back on the image sequence and then choose those coming closest to their starting coordinates in the first frame. But this task is easier to fulfill by nodes in regions moving with lower velocity than by nodes in regions moving faster, and it is trivially fulfilled by nodes on the background. The path the nodes have taken during tracking must be taken into account as well. Therefore we should preferably select nodes with larger movement. So our algorithm is as follows: We assume the object motion to have one major direction, which is the normal case for a short sequence. Now we track all nodes from the first to the last frame, compute the average movement direction (movement vector averaged over all nodes, normalized to length 1), and compute the movement m_i of every node i made in this direction. Then we track all nodes back to the first frame on the same images and compute the deviations d_i of the end positions from the starting positions, which would ideally be zero. Now the nodes can be labeled with their cost values $c_i = d_i - m_i$, "good" nodes having low costs. To create a graph on the object consisting of nodes with minimal cost values, all nodes are sorted with respect to their cost and, starting with the best node, all nodes spatially too close to the good ones in the start frame are deleted (to avoid the jets having too much overlap, i.e., to keep redundancy small). Finally, the best N nodes are connected by a minimal spanning tree resulting in the graph optimal for tracking the object in this sequence. The number N of nodes is the only free parameter of the algorithm; we have chosen $N = 20$ in all examples (see Fig. 3). To summarize, this system can extract graphs of objects and track them completely automatically, without knowing anything *a priori* about those objects. On the other hand, when applied to faces, we get different graphs for different persons, and this makes it less interesting for face recognition applications, because the correspondence between different faces is needed there.

5 A possible application: Pose estimation

Given the tracking of facial landmarks with the frontal view of a face as starting frame, how can we determine the pose in the other frames of an image sequence? First of all, our pose estimator should not rely on specific nodes such as, e.g., the eye nodes, to remain independent of the detailed structure of the graph (i.e., how many nodes it has), and to have the flexibility for using this tool with predefined as well as learned graphs. As another advantage one could expect more robustness and fault tolerance by avoiding strong de-

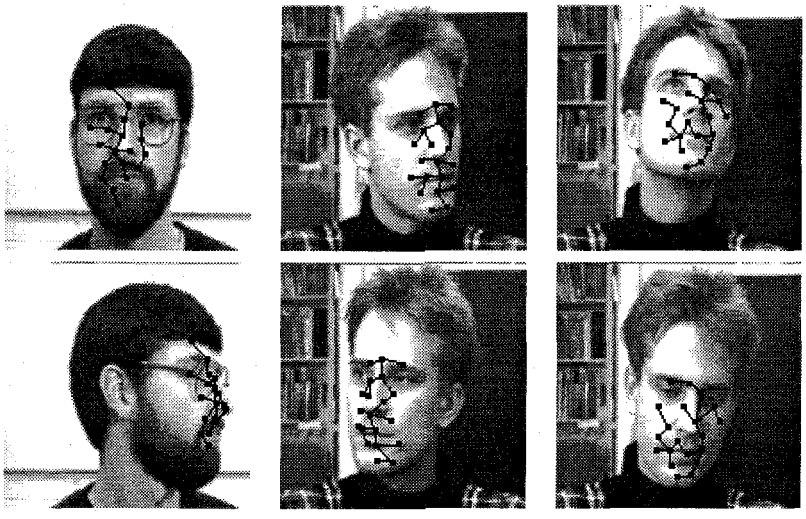

Figure 3. Graphs learned and tracked in three different sequences, the upper image being the start frame, the lower frame 30. While the graph has been learned only on the first 15 frames in the example on the left (it is the same sequence as shown in Fig. 1), it was tracked further to frame 50 looking exactly like frame 1. In the two examples on the right the graphs have been learned and tracked on the whole sequence of 30 frames. Only small inaccuracies can be found, e.g., the node on the chin in the middle example.

pendencies on the fate of single nodes. So, our first simple pose estimator works as follows: We assume the facial landmarks to lie approximately on a flat surface. This assumption turns out to fit very well (we will test it quantitatively below). We determine the optimal affine transformation from the graph in frame 1 onto the graph in frame n. Therefore, we write all node coordinates with respect to the center of mass of their graph and determine the optimal 2×2-matrix projecting the frontal graph onto the other one by solving the overdetermined linear equation system with singular value decomposition. This 2×2-matrix contains the information about the pose of the face in frame n. By simple calculations the angles of rotation in plane, horizontal and vertical rotation in depth[1], and the relative scale are extracted from this matrix leading to the pose. To visualize the quality of the pose estimation, we apply the affine transformation so determined to the frontal face graph and plot it onto the rotated face. In Fig. 4, pose estimation results for two image sequences with relatively large head rotation angles in depth are shown. To investigate the quality of the assumption that the landmarks are lying on a plane, we selected a dozen face images with the head rotated in different directions and placed the landmarks by hand in order to exclude the influence of tracking errors. Then the affine transformation between the frontal view and those test views were computed and applied to the frontal graph. The average distance of the transformed nodes from their correct (i.e., manually defined) position was only 2 pixels, the maximal distance 5 pixels (the node on the tip of the nose). So the assumption of the facial landmarks lying on a flat plane is quite good, at least as long as the face does not come too close to the camera. To make the system robust against tracking errors, a node with a deviation larger than 5 pixels is assumed to be an uncorrectly tracked outlier and is discarded, and the pose determination is repeated without it. Therefore the system will work quite well as long as the majority of the nodes is tracked properly.

[1]To be precise: Only the cosines of the two in-depth rotation angles. Their sign is determined by the displacement of the center of mass.

6 Comparison with related work

Many researchers are active in the areas of tracking as well as pose estimation, so we cannot even make an attempt to give a complete overview here. Very often, the face or head is tracked as a whole by segmentation from motion or by fitting an ellipse to the contour of the head without knowing much about the internal features of the face. Contrary to that, one class of systems analyses explicitly the motion of facial features to estimate facial expressions and emotions like [11, 12, 13]. Nearly all of them work with different specialized detectors for eyes, mouth etc., and need faces of frontal view or at least not too far from it; they also differ in robustness with respect to scale changes or difficulties as beards and glasses. Recently in [14] a facial tracking device was presented which fits an ellipsoid to the facial flow field and estimates the pose. In contrast to these examples, our system is a combination of (i) a powerful frontal landmark finding system which can easily incorporate new examples, thereby avoiding that beard, glasses or race cause problems, and (ii) a quite general tracking device. Our local features, the Gabor-based jets, are neither manually designed for a special task nor restricted to faces. More related to our system is the work of Chellappa: In [15] graphs with Gabor-based jets were tracked in image sequences of a laboratory, not faces. Because they did not use the phase of the filter responses, they were not accurate enough to track the nodes independently, and so they tracked the whole twodimensional rigid graph, which restricted their tool to special cases. Yao & Chellappa [16] track nodes in-

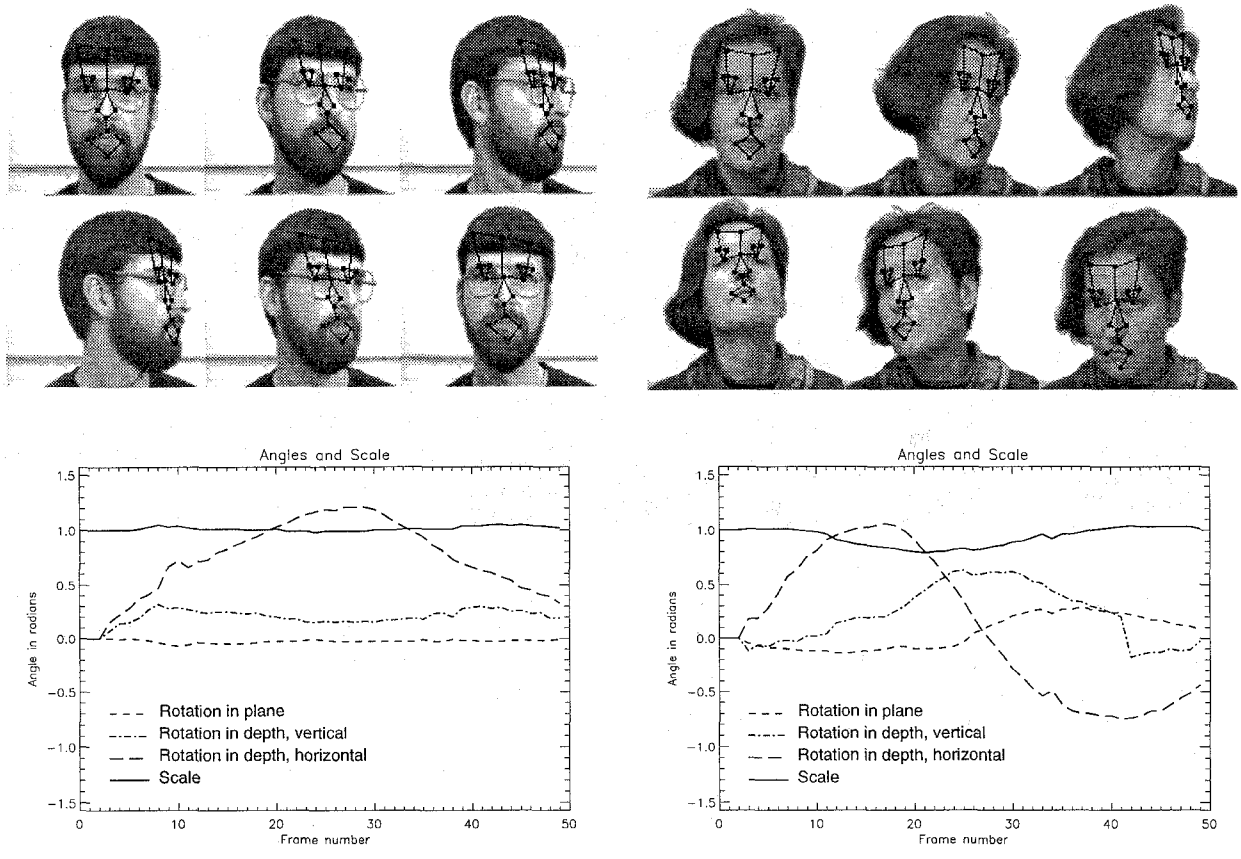

Figure 4. Pose estimation results on two image sequences with relatively large head rotation angles in depth, the one on the left being the sequence from Fig. 1. Instead of showing the nodes tracked independently as in Fig. 1, we have plotted the transformed frontal graph onto the rotated face according to the estimated pose. The assumption of the face being a flat plane becomes less accurate for larger head rotation angles, especially at the nose. As a consequence the estimated angles for extreme poses like profile are too small. On the left hand side, the system erroneously reports a vertical rotation in depth: This is explained firstly by the fact that faces even in the upright frontal view are not perfectly vertical, and secondly by the light spots on the glasses shifting the eye nodes opposite to the movement direction.

dependently by replacing the Gabor-based jets by a local correlation method. The tracking results are always compared to estimates resulting in a feedback loop; nodes can automatically be deleted and generated. They apply their system to lab and landscape scenes to determine the egomotion of the camera. We do not know how easily their system could be applied to moving faces.

Regarding pose estimation we have to distinguish two different problems: Do we have to perform the task on a single face image of unknown pose, or on an image sequence with the pose known in at least one frame. Many researchers have concentrated on the first task (e.g., [17, 18, 19]) because with a perfect system for the first one can solve the second task, too, simply by applying it to all frames separately. But to recognize the pose in a single isolated frame has the great disadvantage that the system can only identify poses it has been trained on before (ignoring possible interpolation between learned poses). Therefore we have chosen the second alternative: Starting with the frontal pose in the first frame our system then can recognize arbitrary poses without any training, and can work as a tool providing the training data for a single-frame pose estimator automatically.

7 Conclusion and Outlook

We have demonstrated a system capable of tracking face graphs and simultaneously estimating the pose in natural image sequences. By tracking the nodes of the graphs in-

dependently, and by using only information contained in the actual frame to find the corresponding nodes in the next frame, we have avoided detailed assumptions about structure or motion. Additionally, by taking Gabor jets as visual features, we have avoided manually designed filters different for eyes, mouth, etc. As a consequence, our system is very homogenous and easy to understand and has practically no free parameters to be tuned. Despite this simplicity we have demonstrated pose estimation and facial feature tracking for larger head rotation angles than many others with their sophisticated systems, although those were designed and tuned only for this task and have many parameters. Of course our general tracking scheme has to be only the starting point to do structure from motion, i.e., to estimate the depth profile of the face, which can then be used by the tracking module itself to become robust enough for every day tasks. On the other hand it can serve as a basic tool to build 3D representations by collecting the information at the nodes tracked, e.g., to learn features that are robust with respect to variation of view point.

Acknowledgements

We would like to thank Laurenz Wiskott for the frontal facial landmark finder, Jan Vorbrüggen for the image acquisition system as well as for his never ending patience in proofreading papers, Karsten Brauer for his practical help, and last not least our system administrator Michael Neef for transforming our Unix workstations into user-friendly machines.

References

[1] M. Lades, J.C. Vorbrüggen, J. Buhmann, J. Lange, C. von der Malsburg, R.P. Würtz, W. Konen, *Distortion Invariant Object Recognition in the Dynamic Link Architecture*, IEEE Trans. Comp., Vol. 42, No. 3, p. 300-311, 1993.

[2] L. Wiskott, J.M. Fellous, N. Krüger, C. von der Malsburg, *Face Recognition and Gender Determination*, Proc. of the International Workshop on Automatic Face- and Gesture-Recognition (IWAFGR), p. 92, Zürich, 1995.

[3] M. Turk & A. Pentland, *Eigenfaces for Recognition*, Journal of Cognitive Neuroscience, Vol. 3, No. 1, p. 71, 1991.

[4] W. Konen & E. Schulze-Krüger, *ZN-Face: A system for access control using automated face recognition*, IWAFGR, p. 18, Zürich, 1995.

[5] T. Maurer & C. von der Malsburg, *Single-View Based Recognition of Faces Rotated in Depth*, IWAFGR, p. 248, Zürich, 1995.

[6] T. Poggio & D. Beymer, *Learning networks for face analysis and synthesis*, IWAFGR, p. 160, Zürich, 1995.

[7] T. Maurer & C. von der Malsburg, *Learning Feature Transformations to Recognize Faces Rotated in Depth*, ICANN, Vol. 1, p. 353, Paris, 1995.

[8] D.J. Fleet & A.D. Jepson, *Computation of component image velocity from local phase information*, Int. Journal of Computer Vision, Vol. 5, No. 1, p. 77, 1990.

[9] W.M. Theimer & H.A. Mallot, *Phase-based binocular vergence control and depth reconstruction using active vision*, CVGIP: Image Understanding, Vol. 60, No. 3, p. 343, 1994.

[10] L. Wiskott, *Labeled Graphs and Dynamic Link Matching for Face Recognition and Scene Analysis*, Verlag Harri Deutsch, Thun, Frankfurt a. Main, Reihe Physik, Vol. 53, 1995.

[11] M.J. Black & Y. Yacoob, *Tracking and recognizing rigid and non-rigid facial motions using local parametric model of image motion*, Proc. Int. Conf. on Comp. Vision, p. 374, IEEE Computer Society, Cambridge, MA, 1995.

[12] I. Essa, T. Darrell, A. Pentland, *Tracking facial motion*, Proc. of the Workshop on Motion of Nonrigid and Articulated Objects, p. 36, IEEE Computer Society, 1994.

[13] A. Saulnier, M.L. Viaud, D. Geldreich, *Real-Time Facial Analysis and Synthesis Chain*, IWAFGR, p. 86, Zürich, 1995.

[14] S. Basu, I. Essa, A. Pentland, *Motion Regularization for Model-based Head Tracking*, submitted to Int. Conf. on Pattern Recognition, Vienna, 1996.

[15] S. Chandrashekhar, C. von der Malsburg, R. Chellappa, *Recursive Tracking of Image Points Using Labeled Graph Matching*, Proc. of the IEEE Int. Conf. on Systems, Man and Cybernetics, Virginia, 1991.

[16] Y. Yao & R. Chellappa, *Tracking a Dynamic Set of Feature Points*, IEEE Trans. on Image Proc., Vol. 4, No. 10, p. 1382, 1995.

[17] A. Pentland, B. Moghaddam, T. Starner, *View-based and Modular Eigenspaces for Face Recognition*, IEEE Conf. on Comp. Vision and Pattern Recognition, 1994.

[18] T. Poggio & K.K. Sung, *Example-based Learning for View-based Human Face Detection*, ARPA, Vol. 2, p. 843, 1994.

[19] N. Krüger, M. Pötzsch, T. Maurer, M. Rinne, *Estimation of Face Position and Pose with Labeled Graphs*, submitted to British Machine Vision Conf., 1996.