MASSACHUSETTS INSTITUTE OF TECHNOLOGY
ARTIFICIAL INTELLIGENCE LABORATORY
and
CENTER FOR BIOLOGICAL AND COMPUTATIONAL LEARNING
DEPARTMENT OF BRAIN AND COGNITIVE SCIENCES

A.I. Memo No. 1494 June, 1994
C.B.C.L. Memo No. 100

Towards an Example-Based Image Compression Architecture for Video-Conferencing

Sebastian Toelg* and Tomaso Poggio

Abstract

Very-low bandwidth video-conferencing, which is the simultaneous transmission of speech and pictures (face-to-face communication) of the communicating parties, is a challenging application requiring an integrated effort of computer vision and computer graphics. This paper consists of two major parts. First, we present the outline of a simple approach to video-conferencing relying on an example-based hierarchical image compression scheme. In particular, we discuss the use of example images as a model, the number of required examples, faces as a class of semi-rigid objects, a hierarchical model based on decomposition into different time-scales, and the decomposition of face images into patches of interest. In the second part, we present several algorithms for image processing and animation as well as their experimental evaluation. Among the original contributions of this paper is an automatic algorithm for pose estimation and normalization. Experiments suggest interesting estimates of necessary spatial resolution and frequency bands. We also review and compare different algorithms for finding the nearest neighbors in a database for a new input as well as a generalized algorithm for blending patches of interest in order to synthesize new images. Extensions for image sequences are proposed together with possible extensions based on the interpolation techniques of Beymer, Shashua and Poggio (1993) between example images. Finally, we outline the possible integration of several algorithms to illustrate a simple model-based video-conference system.

This report describes research done within the Center for Biological and Computational Learning in the Department of Brain and Cognitive Sciences and at the Artificial Intelligence Laboratory at the Massachusetts Institute of Technology. This research is sponsored by grants from the Office of Naval Research under contracts N00014-92-J-1879 and N00014-93-1-0385; and by a grant from the National Science Foundation under contract ASC-9217041 (this award includes funds from ARPA provided under the HPCC program). Additional support is provided by the North Atlantic Treaty Organization, ATR Audio and Visual Perception Research Laboratories, Mitsubishi Electric Corporation, Sumitomo Metal Industries, and Siemens AG. Support for the A.I. Laboratory's artificial intelligence research is provided by ARPA contract N00014-91-J-4038. S. Toelg was supported by a postdoctoral fellowship from the Deutsche Forschungsgemeinschaft while he was at MIT. Parts of this paper were written and edited while S. Toelg was at the Institut für Neuroinformatik, Ruhr-University Bochum, Germany, and later at the Computer Vision Lab., Center for Automation Research, University of Maryland, College Park.

*Current address: Computer Vision Laboratory, Center for Automation Research, Bd. 115, University of Maryland, College Park, MD 20742, USA

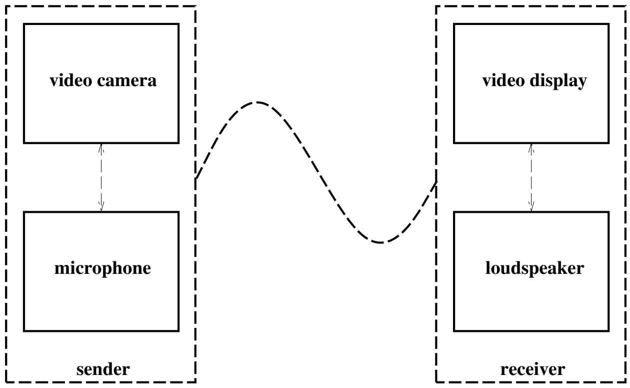

Figure 1: Simple scheme to illustrate a video-conference system. Only one direction of data flow is indicated. For transmission in the reverse direction a mirror symmetric system is used. The dashed arrows between the video and the audio channel indicate a possible coupling.

1 Introduction and motivation

Video-conferencing requires the simultaneous transmission of speech and pictures of the communicating parties in real time and with high quality (see Figure 1). We choose this apparently technical problem as a paradigm to investigate new concepts related to the representation of 3-D objects without explicit volumetric or structural information. In particular, we present a hierarchical architecture for model-based image compression of semi-rigid objects. Human faces are typical representatives of this object class.

This paper consists of two major parts. The first part outlines our approach to video-conferencing (Section 3) and it is introduced with a brief overview of work on face images and especially face recognition. Section 4 presents a specific architecture for a video-conference system based on the previous approach. Several algorithms for image processing and animation are described and experimental results are discussed. A novel robust algorithm for automatic pose estimation and normalization in the presence of strong facial expressions is presented and experimental results are discussed in detail. At the end of Section 4, possible extensions to the video-conferencing scheme by interpolation between example images are reviewed and discussed.

2 Related topics

Conventional model-based image compression methods are briefly reviewed to emphasize their potential problems and to point out their differences from the approach presented here. We review work on images of faces and then focus our discussion on previous work on face recognition that is of interest for our example-based approach to video-conferencing. Finally, we list specific differences between processing for face recognition and for video-conferencing.

2.1 Conventional and model-based compression

The objective here is not to give a comprehensive survey of image compression techniques, but rather to sketch a few key ideas.

Most existing image coding techniques are designed to exploit statistical correlations inherent to images or to time sequences. Conventional waveform coding techniques are known to introduce unnatural coding errors when reproducing images at very low bit rates. The source of this problem lies in the use of statistical correlations which are not related to image content and cannot make use of context information. However, such general techniques are very successful for lower compression ratios. Popular examples are JPEG (Joint Photographic Expert Group) for single images and MPEG for motion sequences, which have also been implemented in hardware. JPEG applies the discrete cosine transform (DCT) to blocks of 8×8 pixel and yields good results for compression ratios up to about $1 : 25$. The algorithm is symmetrical, i.e., computational costs for compression and for decompression are about the same.

It is widely accepted that only model-based compression schemes have the potential of very high compression rates while retaining high image fidelity. In model-based coding schemes the sender (encoder) and the receiver (decoder) contain a common specific model or special knowledge about the objects that are to be coded. The encoder analyzes the input images and estimates model parameters. The decoder synthesizes and reconstructs images from these parameters using the internal model or knowledge. With this kind of coding, very low bit rates can by realized since basically only the analyzed model parameters are transmitted.

The general approach of such model-based coding techniques — being still an active research topic (cf. [1, 20, 44, 24, 25], to give some examples) — for faces is to use a volumetric 3-D facial model (such as polygonal wire frame model). Full face images under standard view can be projected onto the wire frame model for reconstruction by using texture mapping techniques. Two major difficulties are inherent to this approach. Firstly, the generation of realistic 3-D model for individual faces is very difficult in itself. Almost all available automatic techniques yield either poor results for faces or are not applicable to video-conferencing because they require controlled or artificial conditions (structured light illumination, laser scanner, marks attached to the face surface, etc.). Secondly, difficulties arise from the necessity to register the 3-D model and the images used for texture mapping. Also, 3-D motion parameters have to be computed precisely from image data.

2.2 Work on face images and recognition

A good survey of the state of the art on image processing of faces is given in [12]. However, most of the work with faces in computer vision was done on processing for recognition [33, 6, 61, 36, 39, 3, 2, 4, 15, 28, 9] and some work on different kinds of classification tasks [21, 22, 14]. Almost all of this work treats face recognition as a static problem approached by pattern recognition techniques applied to single static images. Only recently the attention of researchers has shifted to the temporal aspect of facial expressions by using optical flow in sequences of face images [41, 42, 25, 65].

We do not intend to give a comprehensive overview of face recognition here. Rather, we will summarize some ideas that are relevant to our work. Recently, a systematic comparison of typical approaches (feature-based versus template-based techniques) to face recognition was carried out by *Brunelli & Poggio* [13, 15]. Several other approaches to face recognition have also been presented (for example [6, 61, 36, 4]).

The first approach is influenced by the work of *Kanade* [33] and uses a vector of geometrical features for recognition. First the eyes are located by computing the normalized cross-correlation coefficient with a single eye template at different resolution scales. The image is then normalized in scale and translation. By independently localizing both eyes, small amounts of head rotation can be compensated by aligning the eye-to-eye axis to be horizontal. The vertical and horizontal integral projection of two directional edge maps (components of the intensity gradient) are used to extract features, such as position and size of nose and mouth, as well as eyebrow position and thickness. Assumptions about natural constraints of faces, such as bilateral symmetry and average facial layout, are used at this stage. A total of 35 geometrical features are extracted. Recognition is then performed with a Bayes classifier applied to the vector of geometric features.

The second approach uses template matching by correlation and can be regarded as an extension of the pioneering work of *Baron* [6]. Images of frontal views of faces are normalized as described above. Each person is represented by four rectangular masks centered around the eyes, nose, and mouth, respectively. The relative position of these masks is the same for all persons. The normalized cross-correlation coefficients of the four masks are computed by matching the novel image against the database. Recognition is done by finding the highest cumulative score. Some preprocessing is applied to the grey-level images to decrease the sensitivity of correlation to illumination effects. An interesting finding is that recognition performance is stable over a resolution range of $1:4$ (within a Gaussian pyramid). This indicates that quite small templates can be used, thus making correlation feasible at very low computational cost.

Gilbert & Yang presented a real time face recognition system using custom VLSI hardware [28]. Their system is based on the template-matching recognition scheme outlined by *Brunelli & Poggio* [13, 15].

In most of the work with faces the images are normalized so that the faces have the same position, size and orientation after manually locating the eyes and mouth. Normalization is often achieved by alignment of the $\top$ spanned by both eyes and the mouth.

2.3 Face recognition vs. video-conferencing

Since much work has been done on face recognition, we want to make use as much as possible of the available experience. On the other hand, there are several significant differences between the problems of face recognition and of video-conferencing. These issues have important implications for our approach and will be discussed in the sequel — Table 1 summarizes the most important differences.

In face recognition the task is in general to match a new face image against a gallery of stored images of different persons. The low-level processing should extract significant features and a subsequent classification should identify the appropriate person and should determine if there is a good match in the database at all. Recognition should be invariant against variations of an individual face (emotional condition, not shaved recently, etc.), but should be highly discriminative to differences between individuals. To achieve this goal, several images taken under different views and illumination conditions are commonly stored in the database. In many applications it is feasible to acquire the example images under (or at least close to) standardized conditions, such as frontal view and neutral facial expression. Moreover, during the recognition phase, it is often possible to repeatedly take snapshot images until one comes close enough to the standard conditions (as mentioned in [28] for instance).

A different paradigm applies for applications where recognition is used for validation or verification only. For instance, in an access control system, there may be prior information about the person's identity available, e.g., by means of a personalized key or code-card. Such a system has only to decide whether the match with a selected entry in the database is good enough in order to verify the identity of the actual person. An exhaustive search for the absolute optimum of the cost function — that is commonly used in recognition — is not feasible here. Other means for normalizing and thresholding the cost function have to be utilized. Of course, the rate of false positive decision should be very low.

In video-conferencing the challenge is to achieve high fidelity (high resolution, realistic colors and quasi real time) transmission between communicating parties, while keeping the required transmission bandwidth as small as possible. A reasonable assumption is that the identity of the person is known and does not change throughout a session (i.e., video-conference call). This assumption allows to exploit knowledge and examples accumulated during previous sessions of the same person. As opposed to face recognition systems, a video-conference system must be very sensitive in explicitly detecting even minor individual variations. The detected variations (either with respect to the previous images, or with respect to similar example images) have to be parameterized and coded to facilitate efficient transmission but without sacrifice of detailed reconstruction.

For a video-conference system, we cannot significantly restrict the range of admitted head poses or limit the variety of facial expressions. The only applicable assumptions arise from the physical and anatomical limitations of the human face, head and body.

Another difference is rooted in the more passive character of our task. We cannot repeatedly acquire new images until we have a suitable one, but we must process

Table 1: The most important differences between face recognition and video-conferencing (transmission for reconstruction of images of 3-D objects) are summarized here.

face recognition	video-conferencing
task is comparison and matching against (large) example database	task is "best" possible reconstruction with small channel capacity
examples in database are from different faces	all examples are of the same face
discrimination of intra-individual features is important	additional information for identification is available
should be invariant to individual variations	individual variations are most important
roughly standard pose of all faces	no a priori standard pose possible
roughly standard facial expressions possible	all facial expressions must be admitted
deals with static images	quasi-continuous sequence of images
image acquisition can be repeated	no repetition, real time required

the incoming sequence of images. Moreover, we cannot direct a person to behave in a certain way (e.g., to obtain approximately standard conditions) — as is possible for recognition, at least while building the database. On the other hand, the high sampling rate within the incoming image sequence allows one to exploit smoothness in time (due to the inertia of physical objects). Thus, the differences between successive frames are small and finding correspondences is easier than for recognition. The algorithms need not start from scratch for each new image, but can rely on predictions derived from previous images to increase stability and coding efficiency. We have good reasons to expect that real time operation (video frame rate) can be achieved in the very near future with an affordable platform.

3 Outline of the approach

In this section we outline our approach to video-conferencing. Due to limited space the scope will be restricted to five topics that have strongest impact upon the system architecture presented in Section 4.

3.1 Examples as model

Several conventional approaches for model-based coding of face images are reported in the literature. Most of them rely on an explicit metric 3-D model (wire frame model) of the specific face [1, 44]. These volumetric models are obtained from image data under different viewpoints or from laser range-scanners. Shape-from-motion algorithms are known to be not very stable and quite noise sensitive. Recently, several structure-from-motion algorithms have been demonstrated to yield good results from real image sequences. However, they crucially depend on stable features that can be accurately localized and tracked in the images. In face images, features of this kind are not present in sufficient number or quality. In some work auxiliary marks (like white points in [1]) were attached to the person's skin. While such aids may be useful for research purposes they are certainly not acceptable for a commercial video-conference system. On the other hand, laser range-scanners are very expensive, comparatively slow, and difficult to handle (due to subtle scanning mechanics). The problems are even more severe for systems that have aligned video cameras to simultaneously record images for texture mapping.

In contrast to these more conventional concepts for video-conferencing we want to avoid the detour of explicit 3-D models. In this paper we advocate a model-based coding scheme to reconstruct images of 3-D objects (e.g., faces) directly from images. The model is based on a set of 2D example images of a person's face. We assume that the set of example images is acquired at the beginning of a session; possibly the system may fall back upon an example database from previous sessions. The set of example images is initially transmitted to the receiver (by means of conventional image compression techniques). During the subsequent continuous transmission the examples are already stored on the sender (encoder) and on the receiver (decoder) side. Therefore, approaches of this kind are also called "memory-based".

The stored example images span a high dimensional space of different poses, viewpoints, facial expressions, and also illumination conditions. As suggested for instance by *Poggio & Brunelli* ([46]) object images can be generated by interpolating between a small set of example images. They described this interpolation in terms of learning from examples [45, 47, 48].

To accomplish the interpolation for image reconstruction, two different approaches are conceivable. The first is related to a new approach to computer graphics [46]. This method has been proposed to synthesize new images from examples attributed with specific and given parameters. For instance, the image sequence of a walking person can be interpolated over time from a few images showing distinct postures; here the parameter is simply time (cf. [46]). In computer graphics we can interactively choose the right examples and tailor the parameterization to synthesize images that are close enough to what we want by interpolation in a relatively low-dimensional parameter space. Some applications for special animation effects in movies can also be found in [64].

The second approach has the same memory-based flavor and is in fact provably almost equivalent. The first step is to find the nearest neighbors, i.e., the most similar examples according to some appropriate distance measure, to the novel view within the database. The following step may then estimate the weight for each neighbor to yield the best interpolation (that is, the closest weighted combination to the novel image) between these examples. There may also be higher dimensional cases where better interpolation can be achieved if examples other than the nearest neighbors are used [10]. For the purpose of video-conferencing this second approach seems more natural. It is in fact not immediately obvious how details of facial expressions should be parameterized in a video-conference system. Moreover, the first method requires explicit estimation of these parameters in addition to pose. The second approach avoids the bottleneck of predetermined parameterization, but selects the basis for interpolation in a more adaptive way. Thus, it is potentially more flexible at the expense of a data-dependent higher dimensional parameter space.

To evaluate the feasibility of this concept in the preliminary implementation of this paper we will consider only the nearest neighbor in the database. *Beymer, Shashua & Poggio* [10] have provided a preliminary evaluation of the two approaches for video-conferencing.

3.2 Number of examples

The major objection to the example-based approach for a video-conference system might be an excessive requirement of memory to store the example database common to the sender (encoder) and the receiver (decoder) side.

Due to today's semiconductor technology, however, fast memory is affordable in abundance. For instance, standard RAM chips with 16 MB can already accommodate 256 images of full size (256×256 pixels) without any further compression. Therefore, storage capacity does not cause an insuperable problem.

On the other hand, the costs for initial transmission of the examples should be kept as low as possible. Thus, an interesting question is to what extent the number of examples can be reduced without restricting the variety of expressions that can be synthesized.

Loosely speaking, one can think of a high dimensional space that is defined or spanned by the example images. The dimension of this space is related to the number of distinct face images that can be generated. In other words, we want to reduce the number of examples used as nearest neighbors or for interpolation without reducing the dimensionality of this example space.

There are several possibilities. The common thread is to divide the abstract example space into lower dimensional subspaces. This, however, relies on the assumption that various properties of face images are separable, i.e., certain aspects of the images are to a large extend independent of others. For reconstruction, a face image can then be composed of intermediate results obtained within the subspaces.

Clearly, the concrete separation into subspaces must be subjected to experimental justification. The next sections discuss some ways to reduce the number of examples, i.e., reduce the costs for initial transmission and storage, while at the same time preserving the dimensionality of the example space.

3.3 Faces as semi-rigid objects

Human heads/faces are representatives of a special class of objects; we call them semi-rigid objects. The name accounts for the fact that these objects are not thoroughly rigid, but at a larger scale still approximately retain their shape. A prerequisite is that a decomposition of the object dynamics into the motion of the object as a whole (e.g., translation of center of mass and rotation about an axis through this point) and the variation of the object's shape makes sense. Moreover, the dynamic range of variations in the object shape is small compared to the total object extension. In other words, the variation in shape can be formulated as a perturbation to a standard or average shape. Since the object shape is subjected to variations, we cannot expect to find points on the object surface that are fixed with respect to any object-centered reference frame, e.g., as is defined by the skull. Therefore, it will, in general, not be possible to infer the object's position based on observations of localized feature points on the surface.

An experimental finding for face images is that facial expressions as well as detailed features and the overall shape of the head become apparent at different ranges of spatial frequency. This is demonstrated in Figure 4. Based on this observation, we conjecture that the pose of the semi-rigid head can be estimated from the low-frequency images alone, thus discarding the variations due to facial expressions.

In face recognition commonly labeled feature points like eyes, corners of the mouth, etc. (see Figure 3 for illustration) are used to compensate for the pose. This requires detection and accurate localization of corresponding points in a new image and the example images in the database. Reliable detection and precise localization of such predefined and labeled features in one image are already difficult. Furthermore, inferring the pose from correspondence of such feature points across distinct images is prone to errors in the presence of facial expressions. For instance, the pupils of the eyes may move by more than 1 cm to both sides due to changes in direction of gaze and vergence movements as is depicted in Figure 3. Also, the pupils may temporarily disappear when the eyelids are closed during twinkling or blinking. The corners of the mouth are rather unreliable feature points for pose estimation, since they are subjected to substantial movements with respect to the head due to facial expressions and during normal speech.

To circumvent the problems sketched above, we propose an adaptive strategy for robust pose estimation and compensation that is suitable for video-conferencing. Details are described in Section 4.1. In a nutshell the main ideas are: to make use only of the low-frequency bands within a coarse-to-fine refinement technique to estimate the pose; to use the constant brightness constraint to find correspondence for all image points; to rely on lower level criteria to select adequate correspon-

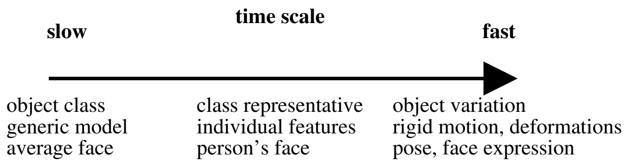

Figure 2: Description at different time-scales gives rise to a hierarchical system architecture (see text for details).

dence points based on a confidence measure; and, finally, to use robust statistics to estimate model parameters within the multiresolution refinement.

3.4 Description on different time scales

In what follows we consider the object model on different time-scales. Based on this decomposition, we derive a hierarchical architecture for a video-conference system. At least three different time-scales should be distinguished as is depicted in Figure 2.

Notice that the general idea is by no means restricted to video-conferencing, but carries over to recognition. This claim is reflected in the three lines under the arrow in Figure 2 leading from a general formulation (top) to the specific example of faces and video-conferencing (bottom).

It is a reasonable assumption that the object class a system has to deal with remains the same over a "long" period of time or does not change at all. Thus, on the slowest time scale we can assume that we have prior knowledge about the object class. This suggests the use of a generic object model such as an average or prototypical face. Some appropriate model assumptions at this level are: a rough model of the 3-D shape of a face/head (e.g., a quadric surface, see [53, 54] for details); human heads exhibit approximately bilateral symmetry; the set of constituents of a face (two eyes, nose, mouth, two ears, etc.); stable features of constituents (the pupil is round and darker than the white eye bulbus, teeth are white, the holes in the nose are dark, the relative size between different parts, etc.); the geometric arrangement of constituents (nose is between mouth and eyes, and vertically centered between both eyes, the ears are far apart, etc.); the variability of constituents in color and in geometry (eyes are fairly fixed in the head, location of pupil changes with direction of gaze, height of an open mouth does not exceed its width, etc.).

On the intermediate time-scale we are concerned with a specific representative of the class. The object model is refined by accounting for individual features that are specific to a person's face. Typical features that are fairly stable are, for example: the 3-D shape of the head insofar as determined by the skull (excluding the region around the mouth, of course); the color of eyes, hair, and teeth; to some extent also the taint and the texture of the skin; typical dynamics of facial expressions and miming, defects (e.g., scars left from operations or accidents) or irregularities (e.g., moles or stained spots) in the appearance.

Finally, on the fastest time-scale we have to deal with the variations of an individual object instance. Even for many non-rigid objects it makes sense to decompose the description of the object dynamics. For instance, as the motion (translation and rotation) of a local reference frame (e.g., center of mass as origin) [1] and a description of the non-rigid dynamics with respect to this object centered reference frame. For the video-conference system we therefore want to separate the estimation of the global pose from the dynamics of the more local facial expressions.

3.5 Decomposition into patches of interest

A further way to reduce the number of examples and the storage requirements is to subdivide the normalized face images into disjunctive or overlapping subregions. Such subregions clipped from the original image may have arbitrary shapes to allow for maximal generality. They are called patches of interest (POI) in the sequel — as opposed to the rectangular regions of interest (ROI) commonly used. This concept is suitable to animate fine facial movements and facial expression by blending the subregions together to reconstruct a composite image at the receiver side. The most important POIs are located around the eyes and the mouth. Note that because of the arbitrary shape, different POIs for the eye pupil, eyelid, and eyebrows could be used as well as different POIs for the corners of the mouth and teeth. In Section 4.3 two algorithms to find the nearest neighbor of an image region are described. A versatile algorithm for blending possibly overlapping POIs of arbitrary shape together is described in Section 4.4.1.

Interestingly, the number of subimages (pasted into a base image of a face) needed to achieve realistic animation is often surprisingly small. This fact has also been pointed out in the literature (see [24] for example). A realistic animation of an eye blink can be achieved by using only four distinct subimages if the eyeball is fixed. *Duffy* noted that only five different images are required for satisfactory simulation of natural eye movements. However, this should be considered as a lower bound to achieve realistic animation of the position of the eyeball (without interpolation between the examples). Also, some examples for different pupil diameters will be required. Moreover, in experiments on lip-reading it has been demonstrated that the essence of a conversation can be picked up from a sequence of images alone. Remarkable, however, is that a set of only 19 visually distinguishable images (showing particular arrangements of lip, tongue, and teeth) is sufficient [30]. Although for realistic video-conferencing a larger number of images may be required, this result is very encouraging.

The advantage of having distinct example patches for different regions of a face is manifold. First of all instead of requiring separate example images of the whole face for all possible poses, lip-shapes, eye positions, etc., a face can be reconstructed from a smaller number of example patches for subregions. For instance, we want to synthesize various face images with eye blinks and speech. Assuming the above mentioned number of ex-

[1] More generally, we may admit not only rigid transformations; the local reference frame may be non-orthogonal and time-varying.

ample images for the subregions (let's say 20 different examples for the mouth and 5 distinct images for each eye) a total of 500 different combinations can be obtained. Thus using only 31 (including a base image for the whole face) examples a much larger number of face images can be animated. Notice that the gain in the number of possible combinations will increase dramatically the more we can subdivide the high dimensional space of possible faces into lower dimensional subspaces.

In addition to significantly reducing the number of needed example images, the memory required to store the POIs is much less than for the whole face images, since the memory required for a POI is roughly proportional to the percentage of the area it covers in the image. For the images shown in this paper (for instance left the image in Figure 5) this yields approximately 1% for each eye POI and about 2–3% for the region around the mouth.

Another benefit of utilizing POIs stems from geometric constraints of the imaging process. The subregions of POIs usually will be projections (perspective projection in general) of a small patch of the 3-D face surface. For most POIs the corresponding 3-D patch will have little depth structure and therefore can be well approximated by a plane. Each POI can be subjected to separate transformations that account for the global head pose before the subregions are blended together. Such transformations can be affine transformations in the image plane or even projective transformations of a plane in 3-D. Even if the 3-D surface is not exactly a plane, perspective distortions and occlusion will be much less of a problem for smaller patches. The advantage is that it may not be necessary to have example subimages for all the variety of head poses. Rather, only a few samples may be sufficient. This is because intermediate views can be generated by appropriate transformations with good fidelity. Thus, fewer examples are required to synthesize faces with the same variety of expressions.

Probably the most compelling argument for decomposing face images into subregions has to do with the nearest neighbors procedure. The most conspicuous facial features to a human observer cover only a small fraction of the overall face image, e.g., we are very sensitive to even minor variations around the eyes. Any procedure to assess similarity between face images that relies on the whole face will be less sensitive to those local variation. Take, for instance, normalized cross-correlation of the full images. The correlation value is likely to be dominated by small illumination differences affecting the whole image rather than by local variations in the "sensitive" regions around the eyes.

Notice that many of these arguments carry over to conventional model-based approaches. Using a physical 3-D model may remedy problems of perspective and occlusion. But still, the appropriate texture (e.g., mouth, eyes open or closed) to be mapped is required. This texture is supplied as a 2-D grey-level or color image in a standard view. The task of generating this image is essentially the same as in the example-based approach advocated in this paper.

4 System architecture

In this section several algorithms that have been developed so far within the novel framework for videoconferencing will be presented and discussed in more detail. Before doing so, we will sketch a possible architecture for a very simple system. The intention is not to present a working system, but to outline its major components so that the developed algorithms can be seen in an appropriate context.

A simple system architecture should comprise the following components:

1. compute normalized pose of new facial images relative to a given example and estimate pose parameters
2. find nearest neighbors within an example database of subimages, e.g., regions around eyes, mouth, nose, etc.
3. transmit model parameters, such as pose parameters and index numbers of the nearest neighbors
4. reconstruct the face image on the receiver side by blending the regions of the subimages together
5. transform the composed face image into the pose of the original image on the sender side

A desirable extension to this simple scheme is to "interpolate" each subimage between several suitable examples (see [10]). Also, adequate ways to update the example database automatically have to be devised.

4.1 Automatic and robust pose estimation

In what follows we describe a novel algorithm for automatic pose estimation and normalization of new face images relative to a given example, i.e., a reference image. Some of the techniques used in this approach to pose estimation are somewhat established in motion estimation and are reviewed briefly for the sake of completeness. We will emphasize, however, the original parts of our algorithm, based on the discussion of Section 3.

The new algorithm can be sketched as follows. Using a restricted affine model for the transformation, four parameters (translation, fronto-parallel rotation, and scale) are estimated in a correspondence scheme on the coarse resolution levels of a Laplacian pyramid only. Local confidence measures for correspondence are used as statistical weights during least squares error (LSE) fit of these parameters. Off-plane rotation is handled by separate example images. The error between the unconstrained measured displacement vector field and affine motion transformation at higher resolutions can used to assess the similarity between both images (see Section 4.3.2).

4.1.1 Analysis in spatial frequency bands

All computation is performed in a hierarchical data structure of the kind originally proposed by *Tanimoto & Pavlidis* to speed up various image processing operations [57]. A comprehensive overview of multiresolution image processing and pyramid structures is given by *Rosenfeld* [50].

For each image we compute a multiresolution pyramid, where $I(x, y, t)$ denotes the discrete grey-level image. The correspondence algorithm can be applied using either Gaussian or Laplacian pyramids. We compute these pyramids, adopting the algorithms proposed by *Burt* [16] and *Burt & Adelson* [17]. A multiresolution pyramid is a stack of N levels of progressively smaller versions of the original image. Let l denote the level within the pyramid and let G^l be the reduced image at the l-th level. The bottom of the pyramid is the original image itself, i.e., $G^0 = I$. The image array G^{l+1} at a higher level is a lowpass filtered and subsampled copy of its predecessor G^l. The images that form the Gaussian pyramid are computed recursively by applying a `reduce` operator to the previous level:

$$G^{l+1} = \texttt{reduce}\ G^l.$$

This procedure is iterated until the highest level N is reached. The `reduce` operator performs a convolution with a smoothing kernel and a subsequent sampling at every second image location in G^l, i.e., every second row and column is skipped. Thus, the size in each direction of the image array reduces roughly by a factor of two between successive levels. As a consequence, the spatial resolution and image size shrinks to a quarter. In our implementation we use a separable and symmetric 5×5 smoothing kernel. Its 1D coefficients are derived from the binomial distribution, e.g., $\frac{1}{16}(1, 4, 6, 4, 1)$.

A Gaussian pyramid consists of an ordered set of lowpass filtered versions of the original image. For the previously discussed type of Gaussian pyramids, the filter bandlimit reduces by an octave (factor of two) from level to level. A Laplacian pyramid may be regarded as a stack of bandpass filtered image "incarnations". The name arises from the fact that the Laplacian edge detector [2] commonly used in image enhancement can be approximated by the difference of Gaussians [40]. The optimal ratio — the one that leads to the best approximation — of standard deviations for inhibiting and excitatory Gaussian about is 1.6. An efficient algorithm to compute a pyramid of bandpass filtered images having a ratio of $\sqrt{2.0}$ is the DOLP transform (difference of lowpass transform) proposed by *Crowley & Stern* [23]. However, here we construct a Laplacian pyramid from the difference of images at adjacent levels of the Gaussian pyramid as proposed by *Burt & Adelson*. Therefore, the ratio of the standard deviations is 2.0. This results in a broader filter bandwidth. This is favorable to achieve consistent motion estimation across several frequency bands (pyramid levels) since there is significantly more overlap between adjacent bands. The center frequency of the bandpass changes by an octave between levels.

The levels L^l of the Laplacian pyramid can be generated from the Gaussian pyramid by using an `expand` operator:

$$L^l = G^l - \texttt{expand}\ G^{l+1},$$

were we define $L^N = G^N$ for the highest level N. The `expand` operator may be thought to do basically the reverse of `reduce`. Its effect is to expand an image array G^{l+1} to an array having twice the linear extent by interpolating values at intermediate locations at level l between given sample points in G^{l+1}.

The upper limit for the storage requirement of such a pyramid is $\frac{4}{3}S$, were S is the memory required for the original image. Moreover, the computational costs increase linearly with S.

4.1.2 Estimation of local transformation and confidence values

Let us suppose that we have two similar face images of the same person. We name the first image $E = E(\mathbf{x}, n)$, indicating that it is one of the example images in the database (n is an index number), and a second new image $I = I(\mathbf{x}, t)$ acquired by the video camera at time t (with $\mathbf{x} = (x, y)^\top$ being the location in the image). Our task is now to bring the new image I to the closest possible alignment with E. Moreover, we want to obtain a robust estimate for the transformation parameters despite the fact that both facial expressions may be significantly different (see Figure 3 for examples).

Our novel algorithm that achieves this goal can be subdivided into two main steps. Firstly, we describe a differential technique for local alignment of small image patches in two images. Secondly, we present an algorithm that fits the parameters of a restricted affine model to describe the global transformation between the two faces due to different poses in both images. Even though the discussion here uses the terminology tailored to our video-conference system, the results and algorithms can be generalized to other problems.

In our derivation we follow the lines of *Lucas & Kanade* who first proposed a differential algorithm for image registration and stereo [37, 38]. However, more recently related techniques have been presented in various flavors in the context of motion estimation or optical flow computation [26, 31, 43, 32, 34, 29, 55]. A comprehensive survey and comparison of differential and other optical flow techniques is given by *Barron, Fleet & Beauchemin* [7]. Unfortunately, they do not consider coarse-to-fine methods that are essential to extend the velocity range of differential techniques.

We assume that at a sufficiently low resolution both images are locally similar and that a small patch in I can be approximated as being a shifted version of a corresponding patch in E. That is: $I(\mathbf{x}) = E(\mathbf{x} + \mathbf{d}(\mathbf{x}))$, were $\mathbf{d}(\mathbf{x})$ is the local displacement vector that we want to estimate. In the case of optical flow techniques E and I are two consecutive images taken from a motion sequence and $\mathbf{d} = \mathbf{v} \cdot \Delta t$ depends on the instantaneous local velocity $\mathbf{v}$ and the time interval Δt.

In order to align both image patches we have to search for the displacement $\mathbf{d}$ that minimizes a distance measure between the patches in I and E. A typical measure is the L_2-norm of the grey-levels over a certain neighborhood Ω centered around the image point $\mathbf{x}$. Moreover, we assume that $\mathbf{d}(\mathbf{x})$ varies only smoothly and thus can be modeled to be constant over Ω. This is reasonable since we apply this procedure to bandlimited images anyway. In addition we want to allow for a weighting function

[2] Formally this is the Laplacian operator applied to a Gaussian convolution kernel of standard deviation σ: $(\nabla^2 G_\sigma) * I(x, y)$.

$W(\mathbf{x}) \geq 0$, which gives us the freedom to emphasize the central region of Ω over the periphery. In order to find $\mathbf{d}$ we can formulate a least squares problem. Thus, we want to minimize:

$$e = \sum_{\mathbf{x}\in\Omega} W(\mathbf{x})\, \|I(\mathbf{x}) - E(\mathbf{x} + \mathbf{d}(\mathbf{x}))\|_2^2 . \qquad (1)$$

We approximate $E(\mathbf{x} + \mathbf{d}(\mathbf{x}))$ by a Taylor expansion truncated after the linear term and differentiate the error e with respect to $\mathbf{d}$. The displacement that minimizes (1) is the solution of the equation system

$$\mathbf{D}\mathbf{d} = \mathbf{c} \quad \text{, where} \quad \mathbf{c} = \begin{pmatrix} \sum W^2(\mathbf{x}) I_x(\mathbf{x}) \Delta I(\mathbf{x}) \\ \sum W^2(\mathbf{x}) I_y(\mathbf{x}) \Delta I(\mathbf{x}) \end{pmatrix} \qquad (2)$$

and

$$\mathbf{D} = \begin{pmatrix} \sum W^2(\mathbf{x}) I_x^2(\mathbf{x}) & \sum W^2(\mathbf{x}) I_x(\mathbf{x}) I_y(\mathbf{x}) \\ \sum W^2(\mathbf{x}) I_x(\mathbf{x}) I_y(\mathbf{x}) & \sum W^2(\mathbf{x}) I_y^2(\mathbf{x}) \end{pmatrix} .$$

Here we have introduced the abbreviations $\Delta I(\mathbf{x}) = I(\mathbf{x}) - E(\mathbf{x})$ for the difference of the intensity values and $I_x = \partial I/\partial x$ for the partial derivative, for I_y respectively. Indication of the explicit dependence of $\mathbf{D}, \mathbf{d}$, and $\mathbf{c}$ on $\mathbf{x}$ is omitted.

In our implementation we use a five tap central difference mask to approximate these spatial derivatives, e.g., the coefficients are $\frac{1}{12}(-1, 8, 0, -8, 1)$. This is a reasonable compromise between computational cost and goodness of the approximation provided that the signal is sufficiently bandlimited. The spatial neighborhood Ω is a 5×5 square centered around the actual image point.

The important fact is, that in addition to the estimated local displacement $\mathbf{d}(\mathbf{x})$, we can obtain an associated reliable measure of its correctness. This confidence measure $k(\mathbf{x})$, as it will be called in the sequel, is used in the second step to weight each displacement vector when we fit the parameters of a low order polynomial model for the global transformation.

In the rest of this section two questions will be discussed: a) what are the solutions of (2), and b) what is the optimal confidence measure for our purposes. Some of the following items have been addressed in individual papers on optical flow techniques, but we think it is worthwhile to repeat them in the context of our specific application.

Note that the matrix $\mathbf{D}$ has two important properties that will be exploited in the sequel. Firstly, it is symmetric, i.e., $\mathbf{D} = \mathbf{D}^\top$. Therefore, $\mathbf{D}$ has two real eigenvalues $\lambda_1, \lambda_2 \in \mathbb{R}$ and the corresponding eigenvectors are orthogonal if the eigenvalues are distinct, i.e., $\lambda_1 \neq \lambda_2$. Secondly, $\mathbf{D}$ is positive semi-definite (the quadratic form $\mathbf{x}\mathbf{D}\mathbf{x} \geq 0 \,\forall\, \mathbf{x} \in \mathbb{R}^2$ and $\mathbf{x} \neq 0$) as can be verified by Sylvester's criterion [35]. Consequently the eigenvalues are nonnegative ($\lambda_1, \lambda_2 \geq 0$). The eigenvalues are computed as the roots of the characteristic quadratic polynomial in our implementation. Let $\lambda_{\min} = \min(\lambda_1, \lambda_2)$ be the smaller eigenvalue, and $\lambda_{\max} = \max(\lambda_1, \lambda_2)$ be the larger one.

In order to solve (2) for the displacement $\mathbf{d}$ three different cases have to be distinguished:

1. If $\det(\mathbf{D}) \neq 0$ the inverse $\mathbf{D}^{-1}$ exists and (2) can be solved for the 2-D displacement $\mathbf{d}$. However, in practice the determinant has to exceed a certain threshold to ensure stable results: $\det(\mathbf{D}) > \tau_{\det}$.
   If the matrix $\mathbf{D}$ is singular, i.e. $\det(\mathbf{D}) = \lambda_{\min} \cdot \lambda_{\max} \leq \tau_{\det}$, we must distinguish the following two cases.
2. If $\lambda_{\max} > 0 \;\wedge\; \lambda_{\min} = 0$, i.e., $\lambda_{\max} \geq \tau_{\max} \;\wedge\; \lambda_{\min} \leq \tau_{\min}$ in our implementation, we have
   $$\begin{aligned} \det(\mathbf{D}) &= \textstyle\sum W^2(\mathbf{x}) I_x^2(\mathbf{x}) \sum W^2(\mathbf{x}) I_y^2(\mathbf{x}) \\ &\quad - \left(\textstyle\sum W^2(\mathbf{x}) I_x(\mathbf{x}) I_y(\mathbf{x})\right)^2 \qquad = 0. \end{aligned}$$
   This is satisfied if $I_x(\mathbf{x}) = \text{const} \cdot I_y(\mathbf{x}) \;\forall\; \mathbf{x} \in \Omega$. The interpretation is that the image intensities within the region Ω lie on a plane and consequently all spatial gradients have the same direction. This situation represents the well-known aperture problem and we can only determine the normal component of the displacement:
   $$\mathbf{d}_n(\mathbf{x}) = \frac{\Delta I(\mathbf{x})}{\|\nabla I(\mathbf{x})\|} \frac{\nabla I(\mathbf{x})}{\|\nabla I(\mathbf{x})\|} .$$
3. If $\lambda_{\max} = 0$ all the entries in $\mathbf{D}$ are zero. The situation $\lambda_{\max} < \tau_{max}$ may occur in practice if the image does have insufficient texture within the region Ω. Consequently, the spatial gradients nearly vanish and we cannot determine any component of displacement.

The confidence measure $k(\mathbf{x})$ associated with the displacement vector $\mathbf{d}(\mathbf{x})$ can be derived from the entries in the matrix $\mathbf{D}(\mathbf{x})$. Several ways to do this have been proposed in the literature:

1. *Simoncelli, Adelson & Heeger* presented a Bayesian framework for optical flow computation [55]. They emphasized the relevance of the trace of the spatial derivative matrix for the probability distributions of velocity vectors. Here, we have for the trace of $\mathbf{D}$:
   $$\begin{aligned} \operatorname{tr}(\mathbf{D}) &= \lambda_1 + \lambda_2 \\ &= \sum W^2(\mathbf{x}) I_x^2(\mathbf{x}) + \sum W^2(\mathbf{x}) I_y^2(\mathbf{x}) \end{aligned}$$
2. It is obvious from the previous discussion that the larger $\det(\mathbf{D})$ is, the more stable is the solution of the linear system (2).
3. *Uras et al.* [63] proposed the smallest condition number [3] $\kappa(\mathbf{H})$ of the matrix $\mathbf{H}$ as an accuracy criterion. The matrix $\mathbf{H}$ is the Hessian of image intensity $I(\mathbf{x}, t)$. It arises from an optical flow technique using second order constraints to recover the 2-D-velocity locally (see also [29]).
   Based on this approach *Toelg* developed a refined and robust algorithm that is used in an active vision system [60, 59]. However, in extensive experiments the magnitude of the determinant $\det(\mathbf{H})$,

[3] The condition number is defined as the ratio between the largest and the smallest absolute eigenvalue of a matrix (cf. [49]). A matrix is ill-conditioned if its condition number κ is too large, and it is singular if κ is infinite.

i.e., the spatial Gaussian curvature in the intensity image, turned out to be a better confidence measure. This finding is in accordance with the more recent results discussed in [7].

Here, we may use the condition number of the matrix $\mathbf{D}$: $\kappa(\mathbf{D}) = \lambda_{\max}/\lambda_{\min}$ as a measurement for reliability.

4. Here, we advocate the magnitude of the smallest eigenvalue $\lambda_{\min} = \min(\lambda_1, \lambda_2)$ as an appropriate confidence measure. This is along the lines of the practical results reported in [7].

A brief justification for this choice will be given. We are only interested in the first case for solving (2) where the full 2-D displacement vector can be recovered reliably. Using $\lambda_{\min}$ as a confidence measure gives a lower bound for the determinant of $\mathbf{D}$, since $\det(\mathbf{D}) \geq \lambda_{\min}^2$. Of course this also gives a lower bound for the trace of $\mathbf{D}$, since $\mathrm{tr}(\mathbf{D}) \geq 2\lambda_{\min}$. Moreover, a larger $\lambda_{\min}$ gives rise to a condition number $\kappa(\mathbf{D})$ closer to unity in the implementation. This is because $\lambda_{\max}$ is bounded from above due to spatial lowpass filtering and due to the limited range of intensity values. It is interesting to note that for any 2×2 matrix, the characteristic equation can be written as: $\lambda^2 - \lambda \mathrm{tr}(\mathbf{D}) + \det(\mathbf{D}) = 0$. We conclude that taking the magnitude of the smallest eigenvalue $\lambda_{\min}$ as a confidence measure implies all the other discussed criteria.

4.1.3 Fitting global parameters for the pose model

We will now derive the second step of the pose estimation algorithm. The local displacement vectors $\mathbf{d}(\mathbf{x})$ and their associated confidence measures $k(\mathbf{x})$ are used to estimate parameters for the global pose transformation model.

We assume an affine model for the displacement vector field. This is a reasonable assumption, since we estimate the pose using low spatial frequency images of the face only. We want to discard facial expressions that have very little effect on these low resolution images as discussed in Section 3.3. We will outline only the basic idea in this section. The reader is referred to Appendix A for mathematical details.

The estimated affine displacement field $\hat{\mathbf{d}}(\mathbf{x}_i)$ is determined at any image location $\mathbf{x}_i = (x_i, y_i)^\top$ by six model parameters in the general case:

$$\hat{\mathbf{d}}(\mathbf{x}_i) = \mathbf{A}\mathbf{x}_i + \mathbf{t} \tag{3}$$

with

$$\mathbf{A} = \begin{pmatrix} b_x & c_x \\ b_y & c_y \end{pmatrix} \quad \text{and} \quad \mathbf{t} = \begin{pmatrix} a_x \\ a_y \end{pmatrix}. \tag{4}$$

Suppose we have n image points $\mathbf{x}_i$ $(i = 1, \ldots, n)$ with a measured displacement $\mathbf{d}(\mathbf{x}_i) = (d_x(x_i, y_i), d_x(x_i, y_i))^\top$ and associated confidence measures $k(\mathbf{x}_i)$.

In general (3) cannot be satisfied exactly for all points. Instead, we want to find the parameter vector $\mathbf{p} = (a_x, b_x, c_x, a_y, b_y, c_y)^\top$ (cf. equation (14) on page 20) that minimizes the error between the measured displacement field $\mathbf{d}(\mathbf{x}_i)$ and the fitted affine displacement field $\hat{\mathbf{d}}(\mathbf{x}_i)$. We assume Gaussian statistics of the process and use the L_2-norm as a distance measure. So, we want to minimize the sum of the squared differences (SSD) over all image points:

$$e = \sum_i w_i^2 \left\| \mathbf{d}(\mathbf{x}_i) - \hat{\mathbf{d}}(\mathbf{x}_i) \right\|_2^2, \tag{5}$$

where we assume a weight w_i^2 for each data point $\mathbf{x}_i$. These weights are computed as the values of a monotonic function $w_i^2 = s(k(\mathbf{x}_i))$ of the associated confidence measures. The function $s(.)$ must be nonlinear to decrease the range, which turned out to be too large. We utilize a sigmoid-like characteristics. In our experiments the choice of $w_i^2 = \sqrt{k(\mathbf{x}_i)}$ worked very well.

The solution for this weighted least squares problem is found by the weighted pseudo-inverse as given in (18). In the general case this leads to six equations for six parameters as given in (19) and (20).

The affine model cannot account for perspective effects and occlusions such as occur during off-plane rotation of the face. This kind of head movement must be handled by separate example images. Therefore, we do not want to allow for any component of shear and reduce the degree of freedom in the affine model. To admit only translation, scale and in-plane rotation in the model we impose additional constraints on the transformation matrix (see Appendix A.2 for details):

$$\mathbf{A} = \mathbf{S}\mathbf{R} - \mathbf{I} \tag{6}$$

with

$$\mathbf{S} = \begin{pmatrix} s & 0 \\ 0 & s \end{pmatrix} \quad , \quad \mathbf{R} = \begin{pmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{pmatrix}. \tag{7}$$

$\mathbf{I}$ is the identity matrix, s denotes the isotropic scale factor, and α is the angle of rotation in the image plane. These constraints result in a coupling between the equations for the general case. The system can be reduced to four equations (see (26) and (27)) in the four model parameters a_x, a_y, s, α (see (25) on page 21 for definition).

Further simplification can be achieved by introducing a new barycentric coordinate system (see Appendix A.4). The origin of this new reference frame coincides with the weighted center if gravity of the image. Expressed in this new reference frame the equation systems take a much simpler form.

For the sake of generality, the case of a full affine transformation is derived first. The new equations for the general case with six free parameters are given in (37) and (38) on page 22. They can be directly solved for the translation parameters a'_x, a'_y. For the remaining parameters only two decoupled 2×2 systems have to be solved.

However, even more significant is the advantage of the new reference system in the constrained case where we do not allow for any component of shear (see Appendix A.4.2). We obtain a 4×4 equation system. As it is obvious from (42) the corresponding matrix is diagonal. Hence, the system can be solved directly and the solution can be written in closed form (see (44) – (46)).

The size of the face images varies because of changes in the distance between the camera and the person's head.

We model these variations by a scale factor s in the image plane. Mathematically this is only correct for a size variation due to a change in the focal length of the lens while the distance is retained (no change of perspective). Since we apply this algorithm only to low spatial frequency images, the influence of perspective distortions and self-occlusion is mostly negligible. Our experiments demonstrated that this approximation is sufficient for a reasonable distance range (about $\pm 20\%$ around the ideal position) and for typical ratios of the focal length and the viewing distance.

4.1.4 Hierarchical control structure

The pose estimation algorithm is embedded in a coarse-to-fine control structure. Computation starts at the highest level N within the Laplacian pyramid (cf. Section 4.1.1). At this level with lowest resolution all model parameters are initialized. On each pyramid level the following steps are performed in sequence:

1. The local displacement vector field $\mathbf{d}(\mathbf{x})$ and the associated confidence field $k(\mathbf{x})$ are computed in the way described in Section 4.1.2.
2. The global transformation parameters of a constrained affine model are estimated by the weighted least squares fit derived in Section 4.1.3 and the appendix.
3. The residual affine parameters estimated at level l and the parameters propagated from a higher lever $l+1$ are combined. As is shown in Appendix A.8 the refined affine parameters at level l are simply obtained as the sum of the propagated parameters and the residual parameters.
4. The estimated model parameters are then propagated from a higher pyramid level $l+1$ to the next lower level l. The relation between the parameters is derived in Appendix A.7. The result is:
$$\mathbf{A}^l = \mathbf{A}^{l+1} \quad \text{and} \quad \mathbf{t}^l = 2 \cdot \mathbf{t}^{l+1}; \qquad (8)$$
thus, only the translation vector $\mathbf{t}^{l+1}$ has to be multiplied by two. The coefficient of the matrix $\mathbf{A}^{l+1}$ are retained.
5. At the current level l the original image I_{orig} is warped according to the propagated model parameters. The warp operation remaps the intensity values according to: $I_{\text{warped}}(\mathbf{x}, t) = I_{\text{orig}}(\mathbf{x} + \hat{\mathbf{d}}(\mathbf{x}), t)$. Note that the addresses $\mathbf{x} + \hat{\mathbf{d}}(\mathbf{x})$ in I_{orig} will in general not coincide with integer coordinates of the image array. The intensity values in I_{warped} have to be interpolated over a small region centered at this address. Only bilinear interpolation is used for the bandlimited pyramid images. The warped image I_{warped} is in closer alignment with the example image E.
6. Steps 1. – 5. are repeated on successively lower pyramid levels, i.e., on successively higher frequency bands of the original images. The process is terminated at a given pyramid level. In our implementation we terminated the refinement at level 2 or 3, e.g., at 1/16 or 1/64 of the original image size.
7. After the refinement is terminated, the estimated affine parameters are propagated (see Appendix A.7) to the bottom level of the pyramid which has the same size and resolution as the original image.
8. To obtain a pose compensated face, the new image I is warped according to the pose parameters. However, here we use bicubic interpolation (e.g., Lagrangian interpolation [11, 5] or bicubic splines) for the warping, since we do not want the high frequency components of the original image to be suppressed and to reduce aliasing effects.

4.1.5 Experimental results

Using real image data, we will now present some typical experimental results to demonstrate the robustness and versatility of the algorithm for pose estimation and compensation.

The two images in Figure 5 might typically occur during a video-conferencing session. Here the left image is a reference image that would be stored in the example database of standardized face images. The right image represents a new video frame acquired during the session. Its pose parameters have to be estimated with respect to the reference image and the new image has to be standardized in pose to facilitate further processing. Figure 6 depicts the resulting images after automatic pose compensation. The estimation is only continued up to a final pyramid level. Subsequently, the parameters are propagated to the resolution of the original image. After estimation on level 3 or 2 no significant change is apparent when estimating at higher resolution. Table 2 gives the values of the corresponding pose parameters. The diagrams in Figure 7 show different graphic representations of the data in Table 2. It is obvious that the parameters estimated at higher levels converge to their bottom value at the highest resolution. Furthermore, the largest adjustments already happen at the higher levels having low resolutions.

Figure 8 shows another more “difficult” pair of images. In addition to the different poses of the faces, there is a significant difference in the facial expressions, e.g., the smile in the reference image with the mouth opened, the teeth visible, shifted corners of the mouth, and different direction of gaze as compared to the more neutral right image. In Figure 9 the resulting images after estimation and pose compensation up to the final resolution level are displayed. By visual inspection no significant change happens after level 2. The corresponding parameter values in Table 4 as well as the diagrams in Figure 10 confirm this finding.

However, it is notable that the curves are not monotonic and not as smooth as in Figure 7. This suggests some caution and a closer examination. Figure 11 shows the pose parameters obtained at each resolution level. The four diagrams depict the results after an increasing number of iterations (1, 2, 3, 5, respectively) at each level before proceeding to the estimation at the next higher resolution level. Table 5 gives the corresponding parameter values. The most dominant change occurs between one and two iterations per level. This observation indicates that one pass per level might not be sufficient to achieve the best possible alignment between both images, especially when dealing with more "difficult" images as in Figure 8. For visual assessment Figure 12 shows the pose compensated images for different numbers of iterations. Although the numerical parameter values for one and a larger number of iterations differ — especially for vertical translation and in-plane rotation — the visual appearance is quite similar. Instructive is a comparison with Table 3 which represents the data for the more similar (in facial expression, not in pose) image pair in Figure 5. Here, the parameter variation for an increasing number of iterations per level is much less.

To be on the safe side, two or three iterations per level will improve the estimation and compensation. Although the difference might not be noticeable by visual inspection, it will be beneficial for further processing steps.

To demonstrate the range of variation the algorithm can cope with, Figure 15 gives images that have been aligned with various reference images. Only two examples with strong facial expression for the reference images are reproduced here. The reference images used in Figure 16 and 17 are similar to the faces in the lower row of Figure 3, which are likely to overtax most feature based algorithms. However, the results are quite convincing, despite the fact that only one pass (number of iterations $i = 1$) has been performed and the final estimation level is $l = 2$. The range in pose is mainly limited by border effects of the filtering within the Laplacian pyramid. The range would be extended provided that the face is smaller with respect to the image size. Notice that the pose estimation algorithm is symmetrical with respect to both images, i.e., it does not make a difference whether the new image or the reference image exhibits a strong facial expression. Moreover, the algorithm works well with strong expressions and deformations in both images. More examples with a variety of different face images and also different backgrounds are presented in [58].

4.2 Discussion of pose estimation

The pose estimation and normalization algorithm described in this section can be located at an intermediate level of complexity among models describing global motions of a face in images. The idea is to make some general assumptions about the object that give rise to a parameterized model. This model will be valid, i.e., a good enough approximation for our purpose, only for a limited range of poses and transformations. Some other linear models in increasing order of complexity (f,n), i.e., number of free parameters f and minimum number of corresponding image points n required, are:

1. pure translation in the image plane (2,1)
2. translation and rotation in the image plane (3,2)
3. constrained affine transformation (4,2) – the model used in the algorithm
4. full affine transformation in the image plane (6,3) – correct for motion of a plane under orthographic projection
5. projective transformation of a plane (8,4) – correct for motion of a plane under perspective projection

Rotation in the image plane thoroughly compensates for rotation of the head in space about an axis parallel to the optical axis of the camera (in-plane rotation). Although not exactly correct in the mathematical sense, in practice translation in the image plane compensates for translation of the head in space parallel to the image plane. In the usual situation for video-conferencing, however, a person will fixate the video display and hence will keep the direction of gaze directed toward the nearby camera. Thus, shifting the head in space will most likely be accompanied by a compensatory rotation of the head. The change of the image plane scale factor can — to a reasonable approximation — take care of distance variations between head and camera; the requirements of weak perspective[4] are sufficiently well satisfied by standard imaging geometry. However, numerous experiments showed that more complex transformations do not yield natural appearing face images and are very difficult or even not accessible for subsequent processing steps, e.g., finding nearest neighbors. For instance, allowing full affine transformation, the additional components of shear can lead to severe distortions of a face. Similarly, for points distant from the plane defining a projective transformation severe distortions occur. Moreover, occlusion effects become very obvious even for small angles of off-plane rotation of a head (e.g., rotation around the neck). For these reasons off-plane rotations are better treated by separate example images. Taking these observations into account, the constrained affine transformation in the image plane is a good compromise between the achieved reduction in the necessary number of example images and the image fidelity.

A further step is to use more prior structural information (than the approximation by a plane) about faces/heads in order to increase the range of transformations resulting in natural looking images. Recently, an algorithm has been presented that applies the model of a general quadric surface to map two images of faces taken under perspective projection onto each other [53, 54]. Using a projective framework, a constructive proof is given that in general nine corresponding reference points and the epipoles in two views of an approximately quadric surface determine the correspondences for all other image points of that surface. Encouraging results have been achieved for real face images. This algorithm, however, still requires manual selection of the reference points. Also, its robustness does not meet the high standards of the simpler algorithm presented in this paper. On the other hand, it has been demonstrated that the quadric surface model can successfully deal with small amounts of off-plane rotation of the head as long as occlusion effects are not too severe.

Although the images used here have a fairly homogeneous background (some structure from the fabric and due to illumination), the pose normalization algorithm

[4] For weak perspective, the depth of objects along the line of sight is small compared with the viewing distance.

would still work in the presence of a moderate amount of textured background surrounding the face. More robustness could be achieved by a preceding segmentation step. Suitable techniques for segmentation of the face from the static background are readily available, such as color segmentation [2, 4] and integral projection of directional information [15]. These techniques perform grey-level based static segmentation of single images, whereas motion segmentation in front of the static background exploits relative motion due to the unavoidable small jitter or movements of a person's head (a robust motion segmentation algorithm is described in [60, 58], for instance). Combinations of these techniques should be considered in order to achieve greater generality.

4.3 Finding the nearest neighbors

In order to find the nearest neighbor(s) in an example database for the subimages extracted from incoming face images, several approaches are conceivable. The important issue here is the similarity measure used for this assessment. We will describe only the two ways to find nearest neighbors that we have implemented.

4.3.1 Template matching in a multiresolution hierarchy

One way to asses the similarity between images or subimages is based on template matching by means of the normalized linear cross-correlation coefficient:

$$C_N = \frac{< E- < E >>< I- < I >>}{\sigma(E)\sigma(I)} \quad (9)$$

$$= \frac{< EI > - < E >< I >}{\sigma(E)\sigma(I)}. \quad (10)$$

where E is an example image and I is a new image that is already normalized in pose. $< . >$ denotes the average operator and $\sigma(.)$ the standard deviation over the image intensity values. The value range is $C_N \in [-1.0, 1.0]$. If E and I are identical we have complete positive correlation $C_N = 1.0$. If $C_N \approx 0.0$, then the images are uncorrelated. The use of this standard technique is suggested by the good results reported for face recognition by other researchers (cf. [6, 13, 28]).

Our implementation performs normalized linear cross-correlation within a multiresolution hierarchy. Correlation is computed between corresponding levels of Laplacian pyramids for the new images and the examples for the subregion. Computation starts at a high pyramid level at low resolution. Cross-correlation is performed for a small window of horizontal and vertical shifts. For each example image the optimal correlation value within the shift window is chosen to achieve more robustness against small distortions. The location of this optimal correlation is propagated to the next lower pyramid level and defines the center of the shift window at the next higher resolution. The size of the shift window may either be constant for all pyramid levels or may increase with the spatial resolution.

The results obtained so far are encouraging. Experiments have been conducted using about 20 example images either for the whole face or for the same subregion around the eye. The new images (taken from a motion sequence), for which the nearest neighbor had to be found, were similar to one of the examples, but were not included in the example database. All face images were previously normalized in pose using the robust algorithm presented in Section 4.1. For all test images the hierarchical template matching algorithm picked the image as a nearest neighbor that appeared most similar to human observers. Robustness against small residual shifts between the images is achieved by choosing the center of the shift window according to the optimum at the previous level. Instead of using different frequency bands within a Laplacian pyramid we also tried correlation using gradient-magnitude images within a Gaussian pyramid of the images. This kind of preprocessing before correlation has been reported to yield superior recognition results [13]. There was no difference in the chosen nearest neighbors. However, differences in the normalized correlation coefficients were less pronounced for images with added random noise.

4.3.2 Fit error of the pose model

Another approach to find the nearest neighbors is more closely related to the automatic pose estimation algorithm described in Section 4.1 and Appendix A. The general idea is to make use of the displacement vector fields between a new image and the example images. The most similar example will be the one having the smallest sum of vector magnitudes taken over the entire region. To be more specific, what is used is the remaining displacement vector field after aligning both images according to the constrained affine transformation that takes care of different poses. In other words, the information used to asses similarity between images is the sum of squared differences between the measured displacement field and the fitted affine displacement field at each pyramid level (see also (5)). If the variation between two images is only due to different poses as defined by the affine model, then both displacement fields will be identical and both images will be assessed to be similar.

Two ways to compute this similarity measure have been considered. The simplest way is to discard the weights assigned to each displacement vector (expressing confidence in the data) and to compute the homogeneous error of the fit. The formulas for doing so are derived in Appendix A.5. The second way is by computing the weighted errors of the fit as derived in Appendix A.6 for the three different model cases. The error can be expressed in terms of the estimated model parameters. For the weighted error only two additional sums over the squares of the measured displacement components have to be computed in addition to the terms already computed to estimate the model parameters. This facilitates an efficient implementation. The summed errors still have to be normalized to account for different image sizes or for the individual weights. The criterion used to asses image similarity is the mean deviation of the measured displacement field from the fitted affine displacement field, i.e., expressed as the variance (see Appendices A.5 and A.6 for details).

Some experimental results of this approach will be discussed now. The last two columns give in Table 2 give the weighted and the homogeneous variance for the image pair in Figure 5; so does Table 4 for Figure 8. These data suggest the following generalizations. The variances are bound between a theoretical lower limit of 0.0 and an upper limit of about 1.5 that is due to the gradient technique used to compute displacement vectors. There are two exceptions. Firstly, the weighted variances at the highest pyramid level are usually larger than at the next lower level. This is because the initial pose estimation is not accurate enough and the variance is dominated by errors due to misalignment. Secondly, at the highest resolution the variances are in general smaller than at the previous level. This phenomenon may have two explanations. Either the images are very similar and the alignment is significantly better at highest resolution; the images are rather different and the computation of the displacement vectors fails because the convergence range of the gradient algorithm is exceeded. This suggests using only the significant intermediate pyramid levels to asses image similarity. Indeed, comparing the data in Tables 2 and 4 shows that the variances for the significant levels are always smaller for the more similar images in Figure 5 that for the images in Figure 8 which exhibit rather different facial expressions.

Previous results (see Section 4.1.5) indicate that even better alignment can be achieved at a given pyramid level if more than one iteration of the pose estimation algorithm is performed. Table 3 and 5 give the variances for multiple iterations. The variances tend to decrease slightly if more iterations per level are done. Although only the data for the bottom level is given in the Tables, this result holds also for intermediate levels.

4.4 Reconstruction of face images

4.4.1 Blending patches of interest together

We now consider the problem of reconstructing a composite face image from a patchwork of example subregions.

In computer graphics texture mapping techniques have been applied to map an image that is a frontal view of a face onto a 3-D wire frame model of the surface. Quite realistic animation of facial details such as eye movements or speech can be achieved by blending sequences of eye and mouth subimages into a base image at appropriate positions prior to mapping (see for instance [24]). Both base and subimage are static frontal views of the face.

As observed by *Duffy* [24], simply pasting a subimage, e.g., a rectangular region of the mouth, yields fairly unsatisfactory results due to visible discontinuities at the edges of the pasted area. These discontinuities are caused by variations in brightness and color between base and subimages as well as by minor changes in facial shape (misalignment) occurring for real subjects. To remedy these shortcomings *Duffy* proposed a transition zone located around the bounding box of the rectangular subimage. Within this transition zone the values of the composite image are computed by weighted averaging between corresponding values of base image and subimage. A weighting function having linear dependence in position is applied. In this way, significantly better animation, i.e., less spurious effects, can be obtained.

However, this simple way of blending a subimage into a base image is not flexible enough for our demands. Let us mention only its most important shortcomings: it requires a rectangular region of fixed size and location, the width and location of the transition zone is very critical in order to achieve realistic results, and there is no straightforward generalization to many (possibly overlapping) subimages because of geometrical constraints.

We will now explain a more general algorithm for blending, i.e., seamlessly merging, several image regions to form a composite image. The essential requirement is to preserve important details of the individual source images without introducing artifacts by the blending process. Two factors are relevant for choosing the width of the transition zone. If the transition zone is narrow as compared to the image features, then the boundary will still be noticeable in the composite image, although it will appear blurred. On the other hand, if the transition zone is too wide, then features from several source images may appear superimposed, similar to a multi-exposure in photography. These conflicting requirements cannot be fulfilled simultaneously in general, i.e., for images covering a wide range of spatial frequencies. A suitable transition width can be found only if the spatial frequency band of the images is relatively narrow.

To overcome this problem *Burt & Adelson* [18, 19] proposed a multiresolution approach for merging images [5]. First, each source image is decomposed into a set of bandpass filtered component images. In the next step, the component images are merged separately for each band to form mosaic images by weighted averaging within a transition zone. The transition width corresponds approximately to a half wave length of the band center frequency. Finally, these bandpass mosaic images are simply summed to obtain the desired composite image. Thus, the transition zone always matches the size of the image features. This technique has been formulated for pairs of static source images and demonstrated to yield superior results over simpler techniques in several applications [18, 19].

We adopt this idea and give a more general formulation that applies to any finite number of source images and to time sequences of images. The subregions of face images will be called patches of interest (POI). As opposed to rectangular regions of interest (ROI) a POI may have arbitrary shape. Moreover, several POIs may overlap and can be arranged in a stack. In this pseudo 3-D structure only the top patches contribute to the composite image.

Our blending algorithm takes two kinds of inputs. Firstly, an indexed set of subimages compatible with the corresponding POIs. These subregions of facial example images are previously normalized in pose. These examples comprise, among others, base images of the face under various off-plane rotation views and a variety of subimages of facial details like different mouth shapes and different states of eye movement and eye blinks. Secondly, a sequence of index images that describe the composite face image appearance over time — thus both have

[5] Amnon Shashua provided valuable contributions to our discussion and the relevant papers.

the same size. Each pixel value in these index images refers to the POI that should dominate the composite image at the corresponding position.

The procedure consists of the following steps:

1. For each example subimage associated with the POI having the index number i generate a Laplacian pyramid LE_i consisting of bandpass filtered images LE_i^l using the procedure described in Section 4.1.1[6]. This has to be done only initially and the pyramids can be stored for fast access. The storage requirement is only 4/3 of the original image.
2. For each index image X_n in the time sequence collect the set of index numbers $\mathcal{N}$ of all referenced example subimages. Simultaneously, build a binary mask image M_i for each index number. A pixel is assigned the value 1 at positions having the corresponding index value in the index image and 0 everywhere else.
3. Generate a Gaussian pyramid GM_i for each mask image M_i included in the index set $\mathcal{N}$.
4. The entries in the GM_i pyramid are used as weights for the corresponding bandpass filtered example subimages in LE_i^l. For each band (level l of the pyramid) a mosaic LC^l image is computed in the following way:
$$LC^l = \sum_{i \in \mathcal{N}} GM_i^l \cdot LE_i^l,$$
where the sum is taken only over the examples included in the index set $\mathcal{N}$. This significantly increases efficiency if a large number of potential example subimages is used as required for realistic animation.
5. Finally, the procedure of generating the Laplacian pyramids is reversed to obtain the composite image C. This is achieved by the following iterative procedure starting the highest level N of the mosaic pyramid LC:
$$GC^{l-1} = LC^{l-1} + \texttt{expand}\ GC^l,$$
where $GC^N = LC^N$ and $C = GC^0$.

Since the POIs associated with each index image may change between frames, it is desirable to have a smooth transition between successive frames in an animation. An adequate way to accomplish this is to apply a low-pass filter to the binary mask image M_i before generating the Gaussian pyramid GM_i. Weighting the past few mask images with an exponentially decaying weighting function can be implemented very efficiently in a recursive way (see [60] for algorithm). However, application of such a filter may require an additional normalization step of the pixel values in the mosaic images. This is because in general it cannot be guaranteed that the sum of all weights for each image location is equal to unity.

The above sketched algorithm has been implemented and a very promising, realistic animation of face images have been obtained. This approach to blending images could also be successfully combined with texture mapping techniques.

[6] Indication of the individual pyramid level l is omitted if procedures are applied homogeneously to all levels.

4.4.2 Recovery of original pose

The last processing step before displaying the reconstructed face image is to transform the image reassembled from normalized examples to the pose of the original input image. This requires reversal of the transformation of the pose compensation performed on the sender side from the transmitted pose parameters. Inverting the mapping of the image warping (that is computing the mapping from image 2 to image 1 if the mapping from 1 to 2 is given) in not trivial in general [64]. However, due to the parametric model applied here, the parameters for warping the normalized pose face image to the original pose can be computed easily from the transmitted pose parameters describing the alignment with the reference image.

In Appendix B a closed-form solution is derived to obtain the inverse mapping parameters from the original parameters. To demonstrate the inverse mapping, in Figure 13 the reference image depicted in Figure 5 is warped towards the pose of the right image; this is the reverse of the pose compensation.

Figure 14 summarizes the processing steps of a simplistic video-conference system:

- normalizing the pose of a new face image,
- finding the most similar example out of a database of normalized images,
- reconstructing the face image given an index number (in the database) and inversion of the pose normalization.

4.4.3 Interpolation between examples

In this section we point out possible extensions of the system architecture outlined in Section 4. Instead of using the nearest neighbors only, it is natural to "interpolate" novel views between examples from the database as already mentioned in Section 3.1. Extending previous results [45, 47, 48, 46], recent work of *Beymer, Shashua & Poggio* [10] presents the mathematical formulation and experimental demonstrations of several versions of such an approach. The feasibility of interpolation between images has been successfully demonstrated for the multidimensional interpolation of novel human face images.

So far, the examples are selected manually for training. For applications in video-conferencing the process of picking adequate examples from the database for subsequent interpolation obviously has to be automated; this becomes an even more significant issue for higher dimensional interpolation. Various strategies are conceivable and we will suggest some — still subject to experimental evaluation. For 1D interpolation (morphing between a pair of similar images) an exhaustive search for the two database images that allow for the best interpolation result, i.e., the result that comes closest to the novel image, appears reasonable. But, for higher dimensional interpolation (at least four examples are needed for 2-D interpolation) this strategy seems to be prohibitive due to excessive computational costs (combinatorial explosion).

In order to reduce the search space for the interpolation basis, i.e., the examples that span the space of possible interpolated views, we suggest the following strategy: i) Find the nearest neighbor to the novel view in the database (cf. Section 4.3). ii) Restrict the search space to images that are within a certain "distance" from the novel view or the nearest neighbor; the later is more efficient since the distances between all examples can be precomputed. Of course this presents the problem of finding a suitable metric to define the distance. Potential candidates are provided by the algorithms described in Sections 4.3.1 and 4.3.2. However, other (e.g., feature based) metrics used for recognition should also be considered. iii) The maximum distance could be chosen to contain only the number of examples required for interpolation of a given dimensionality. Note that this approach may result in ambiguities if more than one example image has the same distance — this cannot be excluded in a high-dimensional space. Alternatively, one can chose the maximal distance so that more than the required examples are included in the search space. Subsequently, the number is reduced by abandoning redundant images, for instance using a leave-one-out strategy that keeps only the examples providing the best result.

The argument for the strategy of taking the nearest neighbor as a interpolation basis is not obvious and requires a better practical understanding of the interpolation algorithms (see [10]). The algorithm consists of two major steps. First, correspondence vector fields are estimated, which capture the geometrical relations between the novel image and the examples in the best possible way. In the second step, the texture information is pixelwise interpolated between the intensity values in the examples referenced by the correspondence vector fields. In theory, optimal vector field interpolation for the first step in general cannot be obtained using just the nearest neighbors. On the other hand, because of illumination effects, occlusions and distortions it is likely that the nearest neighbors contain the most similar texture.

The interpolation technique has been applied to images of whole faces. A natural extension is to apply the interpolation technique separately to patches of interest (POI), as proposed in Section 3.5. By means of the generalized blending technique described in Section 4.4.1 new views can be composed. We expect a large potential in combining these two concepts. Here are two examples: i) The location of the iris and pupil of the eye (as it changes with the direction of gaze) may be interpolated from four examples or even from two examples if we disregard the minor vertical movements. Additional example sets may be used to account for varying pupil size. ii) Realistic synthesis of eye blinks may require not many more than two examples.

We will now suggest a further extension of the multidimensional interpolation algorithm. So far, the same coefficients are used for interpolating an approximated geometric relation (correspondence vector fields) between the examples and the novel image as well as for the pixelwise interpolation of the intensity (texture) information (see Sections 4.1. and 4.2 in [10]). These coefficients are estimated to yield the "best" possible approximation, e.g., in the least square sense, for the correspondence vector field of the novel image with respect to the example(s). While this coupling between the coefficients for geometric and texture interpolation makes sense for certain applications of 1D interpolation, e.g., for frame rate conversion (see [8]), it is not necessarily the best approach for more general cases.

We suggest exploiting the freedom of adjusting the coefficients for geometric and texture interpolation independently. A straightforward way to do this is to estimate the optimal coefficients for geometric interpolation first (as before) and subsequently to optimize a second set of coefficients for texture interpolation. The second step uses the previously recovered geometric relations to access the intensity information at corresponding locations in the example images. The coefficients for the second step could be found by least square minimization applied to the intensity values, for example. While only doubling the number of parameters, we expect even more realistic "rendering" of novel images, especially if the example images are captured under small variations in the illumination (direction and intensity). A further amendment could include "virtual" example images that could partially compensate for small lighting changes. In the simplest case, the coefficients for a completely black and a white image could be used to adjust the average brightness of the synthesized image. More elaborated versions would use several additional "virtual" examples showing slowly varying intensity in different directions. Adjusting the corresponding coefficients will to some extent simulate changes in illumination direction. Of course, this is not correct in the strict physical sense that would require multiplication of the surface reflection by the illumination intensity, where both may be functions of the relative angles. However, for small variations the linear compensation will give reasonable results at a very low computational cost.

Two recent achievements should be mentioned in the context of generating new views from a small number of model views or example images. *Shashua & Toelg* [53] showed that a nominal quadric transformation for all image points, i.e., a transformation assuming that an object surface can be approximated by a quadric, can be successfully applied to register two face images. All parameters of the transformation (the quadric and the relative camera geometry) can be recovered from only nine corresponding points over two views [53, 54]. Alternatively, the transformation can be recovered using only four corresponding points and a given conic in one view (encompassing the face in our application) [54].

This algorithm is relevant here for two purposes. Firstly, it can be used as a preprocessing step to facilitate pixelwise correspondence, i.e., bringing two views into closer alignment. This step is essential for views that are too different to be directly accessible to standard dense correspondence algorithms; a small number of distinct feature points can easily be found in the two initial views. Secondly, the transformation according to the quadric surface model is described by a few parameters (at most 17 are needed). In a video-conference system only these parameters need to be transmitted in order to synthesize novel views of a face from one given example, provided that the views are not to different (self-occlusion, etc.). The nominal quadric transformation is significantly more general than simpler transformations commonly used (e.g., affine, or transformation due to a plane) and superior registration results have been obtained with face images (see [53, 54] for examples).

The second achievement is a generalization of the "linear combination of views" result obtained by *Ullman & Basri* [62] that relates three orthographic views of a 3-D object (ignoring self-occlusion). Recently, *Shashua* [52, 51] proved that the image coordinates of corresponding points over any three perspective views (uncalibrated pinhole camera) of a 3-D object are related by a pair of trilinear equations. The 17 independent coefficients of this trilinear form can be recovered linearly from 9 corresponding points over all three views. Once these coefficients are recovered and full correspondence between the two model views has been established, the corresponding locations in the novel (third view) can be obtained uniquely for all other points.

The direct approach of using the trilinear result to generate new views has several theoretical advantages over classical structure from motion methods as well as over methods to recover non-metric structure (see [51] for a detailed discussion). Moreover, processes that are known to be unstable in the presence of noise, such as recovering the epipolar geometry, are avoided. The trilinear algorithm proved to be significantly more stable in the presence of errors in the image measurements. So far, the trilinear algorithm has been evaluated only in computer simulations and using re-projection of isolated points in real imagery, though an implementation to transform dense images is planned for the near future [7].

Although [52, 51] emphasis is given to the task of recognition of 3-D objects, the trilinear method may have interesting applications in an example-based video-conference system. Only 17 parameters are needed to represent a new view with respect to two model views — as opposed to only one model view for the nominal quadric transformation. The two model views, or examples as we called them earlier, are available on the sender and the receiver side. The same algorithm for achieving full correspondence between these reference images is applied on both sides. To encode a third view, the sender solves for the 17 parameters by using many points to increase robustness; this can be done using a least squares approach. Transmitted, however, are only the 17 parameters needed for reconstruction.

5 Conclusions and outlook

The concept of an alternative approach to video-conferencing that is sketched in the first part of this paper appears to be very promising. Several algorithms have been presented that form modules in a system architecture. Each of these modules has proved to be robust under realistic conditions. Much further work on integration and refinement of the system is required. Once a more elaborated system based on our approach is available, it will be interesting to compare its performance with state-of-the-art systems utilizing traditional model-based approaches.

For an automatic video-conference system, i.e., a system that does not require any human intervention, additional components are obviously required. However, many suitable algorithms are already known and described in the literature. The most important components are briefly discussed in the sequel.

The separation of the face and the uncovered background can be achieved by the methods sketched at the end of Section 4.2. Recently an algorithm for human face segmentation by fitting an ellipse to the head has been described [56]. This algorithm is robust enough to deal with images having moderately cluttered backgrounds.

Another, more critical problem is the automatic selection and positioning of the POIs in the face image. This task is significantly simplified by the robust pose normalization presented in this paper. In the normalized face images knowledge about the average facial geometry is easily applicable to define relevant regions. For individual faces, regions of high surface structure can be detected by means of texture analysis techniques (e.g., high gradient or high spectral energy). The generalized blending algorithm described in Section 4.4.1 to some extent smooths out visible discontinuities at edges between adjacent POIs. It is desirable, however, to locate boundaries within regions of low surface texture and not at conspicuous facial features — loosely speaking, we apply a reversed edge detector. For this purpose orientation selective filters (like Gabor filters, wavelets, or steerable filters [27]) may be the way to go. They make it possible to seek for appropriate locations depending on the orientations of boundary lines that are approximately given.

An important step is the automatic acquisition of the example database. Here at least two distinct tasks have to be distinguished. During the initialization phase examples have to be acquired that span the largest possible range of facial expressions and poses. However, extreme poses and expressions may be disregarded at this stage. Moreover, two cases have to be considered. In the standard case no prior examples for a person are available when a person uses the system for the first time. Then all examples have to be transmitted initially using conventional image compression, e.g., JPEG. In the second case prior examples are available from previous sessions. New examples have to be acquired on the sender side and it has to be determined which of the old examples are still compatible with the current situation. This is necessary to update changes in facial hair, for example. The gain is that usually only some new examples may have to be transmitted to the receiver side. However, the computational cost of the evaluation on the sender side may be higher.

During the subsequent transmission phase of the session the task is somewhat different. In general, novel images should be approximated in terms of the examples with sufficient accuracy. Precautions should be taken to

[7] Amnon Shashua, personal communication, March 1994.

detect whenever the best possible reconstruction (based on the available examples) is not satisfactory. In these rare cases, e.g., for unusually strong expressions, (compressed) new image data have to be transmitted. At this point it is not clear whether this additional image data for non-standard cases should supplement the standard database as an additional example; this has to be subjected to experimental evaluation.

Acknowledgements

I want to thank all the people at CBCL and at the AI Lab. for providing a creative atmosphere and a pleasant working environment. In particular, I want to thank Amnon Shashua for his advice and for numerous scintillating discussions. Thanks also to A. Rosenfeld for correcting the final version of this paper and for his valuable comments.

While S. Toelg worked at MIT he was supported by a postdoctoral fellowship from the Deutsche Forschungsgemeinschaft.

References

[1] K. Aizawa, H. Harashima, and T. Saito. Model-based analysis synthesis image coding (MBASIC) system for a person's face. *Signal Processing: Image Communication*, 1(2):139–152, Oct. 1989.

[2] S. Akamatsu, T. Sasaki, H. Fukamachi, N. Masui, and Y. Suenaga. An accurate and robust face identification scheme. In *Proc. 11 IARP Intl. Conf. on Pat. Recog.*, pages 217–220, The Hague, the Netherlands, Aug. 1992. IEEE Computer Society.

[3] S. Akamatsu, T. Sasaki, H. Fukamachi, and Y. Suenaga. A robust face identification scheme – KL expansion of an invariant feature space. In *Intelligent Robots and Computer Vision X: Algorithms and Techniques*, volume 1607, pages 71–84, Boston, MA, Nov. 1991. SPIE—The International Society for Optical Engineering.

[4] S. Akamatsu, T. Sasaki, H. Fukamachi, and Y. Suenaga. Automatic extraction of target images for face identification using the sub-space classification method. *IEICE Transactions on Information and Systems*, 1993. Appears in special section on machine vision & applications.

[5] D. H. Ballard and C. M. Brown. *Computer Vision*. Prentice-Hall, 1985.

[6] R. J. Baron. Mechanisms of human facial recognition. *International Journal of Man Machine Studies*, 15:137–178, 1981.

[7] J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of optical flow techniques. Technical Report RPL-TR 9107, Queen's University, Kingston, Ontario, Robotics and Perception Laboratory, July 1992.

[8] J. R. Bergen and R. Hingorani. Hierarchical motion-based frame rate conversion. Technical report, David Sarnoff Research Center, Apr. 1990.

[9] D. Beymer. Face recognition under varying pose. Technical Report AI Memo 1461 and CBIP Paper 89, Artificial Intelligence Laboratory, MIT and Center for Biological Information Processing, Whitaker College, Dec. 1993.

[10] D. Beymer, A. Shashua, and T. Poggio. Example based image analysis and synthesis. Technical Report AI Memo 1431 and CBIP Paper 80, Artificial Intelligence Laboratory, MIT and Center for Biological Information Processing, Whitaker College, Nov. 1993.

[11] I. N. Bronstein and K. A. Semendjajew. *Taschenbuch der Mathematik*. Verlag Harri Deutsch, 1980.

[12] V. Bruce and M. Burton. *Processing Images of Faces*. ABLEX Publishing-Corporation, Norwood, NJ, 1992.

[13] R. Brunelli and T. Poggio. Face recognition: Features versus templates. Technical Report TR 9110-04, Istituto per la Ricerca Scientifica e Tecnologica, Oct. 1992.

[14] R. Brunelli and T. Poggio. Caricatural effects in automated face perception. *Kybernetik / Biol. Cybernetics*, 69:235–241, 1993.

[15] R. Brunelli and T. Poggio. Face recognition: Features versus templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1993. accepted for publication.

[16] P. J. Burt. Fast filter transforms for image processing. *Computer Graphics and Image Processing*, 16:20–51, 1981.

[17] P. J. Burt and E. H. Adelson. The Laplacian pyramid as a compact image code. *IEEE Transactions on Communications*, 31(4):532–540, Apr. 1983.

[18] P. J. Burt and E. H. Adelson. A multiresolution spline with application to image mosaics. *ACM Transactions on Graphics*, 2(4):217–236, Oct. 1983.

[19] P. J. Burt and E. H. Adelson. Merging images through pattern decomposition. In *Applications of Digital Image Processing VIII*, volume 575, pages 173–181. SPIE—The International Society for Optical Engineering, 1985.

[20] C. S. Choi, H. Harashima, and T. Takebe. Analysis and synthesis of facial expressions in knowledge-based coding of facial image sequences. In *Proc. ICASSP*, pages 2737–2740, May 1991.

[21] G. Cottrell and M. Fleming. Face recognition using unsupervised feature extraction. In *Proc. Intl. Neural Network Conf. (INNC)*, Paris, 1990.

[22] G. W. Cottrell and J. Metcalfe. EMPATH: Face, emotion, and gender recognition using holons. In *Proc. NIPS 3*, 1991.

[23] J. L. Crowley and R. M. Stern. Fast computation of the difference of low-pass transform. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(2):212–222, Mar. 1984.

[24] N. D. Duffy. Animation using image samples. In V. Bruce and M. Burton, editors, *Processing Images of Faces*, pages 179–201. ABLEX Publishing-Corporation, Norwood, NJ, 1992.

[25] I. A. Essa. Visual interpretation of facial expressions using dynamic modeling. Technical Report TR 235, Perceptual Computing Group, Media Laboratory, MIT, July 1993.

[26] C. L. Fennema and W. B. Thompson. Velocity determination in scenes containing several moving objects. *Computer Graphics and Image Processing*, 9:301–315, 1979.

[27] W. T. Freeman and E. H. Adelson. The design and use of steerable filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(9):891–906, 1991.

[28] J. M. Gilbert and W. Yang. A real-time face recognition system using custom VLSI hardware. In *Proc. of Computer Architectures for Machine Perception Workshop*, Dec. 1993. accepted for publication.

[29] F. Girosi, A. Verri, and V.Torre. Constraints for the computation of optical flow. In *Proc. IEEE Workshop on Visual Motion*, pages 116–124, Irvine, CA, Mar. 1989. IEEE Computer Society Order Number 1903.

[30] R. L. Height. Lip reader trainer: Computer program for the hearing impaired. In *Proc. Johns Hopkins first national search for applications of personal computing to aid the handicapped*, pages 4–5, Los Alamitos, CA, 1981. IEEE Computer Society.

[31] B. Horn and B. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[32] B. K. P. Horn. *Robot Vision*. MIT Press, Cambridge, Mass., 1986.

[33] T. Kanade. *Picture Processing System by Computer Complex and Recognition of Human Faces*. Unpublished Ph.D. thesis, Dept. of Information Science, Kyoto Univ., 1973.

[34] J. K. Kearney, W. B. Thompson, and D. L. Boley. Optical flow estimation: An error analysis of gradient-based methods with local optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(2):229–244, Mar. 1987.

[35] G. A. Korn and T. M. Korn. *Mathematical Handbook for Scientists and Engineers*. McGraw-Hill Book Company, 1968.

[36] M. Lades, J. C. Vorbrüggen, J. Buhmann, J. Lange, C. v. d. Malsburg, R. P. Würtz, and W. Konen. Distortion invariant object recognition in the dynamic link architecture. *IEEE Transactions on Computers*, 1992. accepted for publication.

[37] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. DARPA Image Understanding Workshop*, pages 121–130, 1981.

[38] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. International Joint Conference on Artificial Intelligence*, pages 674–679, Vancouver, 1981.

[39] B. S. Manjunath, R. Chellappa, and C. v. d. Malsburg. A feature based approach to face recognition. Technical Report CAR-TR-604 and CS-TR-2834, Computer Vision Laboratory, Center for Automation Research , Univ. of Maryland, Jan. 1992.

[40] D. Marr and E. Hildreth. Theory of edge detection. *Proc. R. Soc. Lond. B*, 207:187–217, 1980.

[41] K. Mase. Recognition of facial expression from optical flow. *IEICE Transactions*, E 74(10):3474–3483, Oct. 1991.

[42] K. Mase and A. Pentland. Automatic lipreading by optical flow analysis. *Systems and Computers in Japan*, 22(6):67–76, July 1991.

[43] H.-H. Nagel. Displacement vectors derived from second-order intensity variations in image sequences. *Computer Vision, Graphics, and Image Processing*, 21:85–117, 1983.

[44] Y. Nakaya, Y. C. Chuah, and H. Harashima. Model-based/waveform hybrid coding for videotelephone images. In *Proc. ICASSP*, pages 2741–2744, May 1991.

[45] T. Poggio. A theory of how the brain might work. In *Cold Spring Harbor Symposia on Quantitative Biology*, volume LV, pages 899–910. Cold Spring Harbor Laboratory Press, 1990.

[46] T. Poggio and R. Brunelli. A novel approach to graphics. Technical Report AI Memo 1354 and CBIP Paper 71, Artificial Intelligence Laboratory, MIT and Center for Biological Information Processing, Whitaker College, Feb. 1992.

[47] T. Poggio and S. Edelman. A network that learns to recognize three-dimensional objects. *Nature*, 343:263–266, 1990.

[48] T. Poggio and F. Girosi. Networks for approximation and learning. *Proceedings of the IEEE*, 78(9):1481–1497, Sept. 1990.

[49] W. H. Press, S. A. Teulolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C*. Cambridge Univ. Press, 2nd edition, 1992.

[50] A. Rosenfeld, editor. *Multiresolution Image Processing and Analysis*, volume 12 of *Information Sciences*. Springer-Verlag, 1984.

[51] A. Shashua. Algebraic functions for recognition. Technical Report AI Memo 1452, Artificial Intelligence Laboratory, MIT, Jan. 1994. Submitted to PAMI, Jan. 1994.

[52] A. Shashua. Trilinearity in visual recognition by alignment. In *Proc. 3rd European Conf. on Computer Vision*, pages 479–484, Stockholm, Sweden, May 1994. Springer-Verlag.

[53] A. Shashua and S. Toelg. The quadric reference surface: Applications in registering views of complex

3D objects. In *Proc. 3rd European Conf. on Computer Vision*, pages 407–416, Stockholm, Sweden, May 1994. Springer-Verlag. Also CAR-TR-702 and CS-TR-3220,Computer Vision Laboratory, Center for Automation Research , Univ. of Maryland, Feb. 1994.

[54] A. Shashua and S. Toelg. The quadric reference surface: Theory and applications. Technical Report AI Memo 1448 and CBCL Paper 85, Artificial Intelligence Laboratory, MIT and Center for Biological and Computational Learning, Whitaker College, June 1994. Also submitted for publication to International Journal of Computer Vision.

[55] E. P. Simoncelli, E. H. Adelson, and D. J. Heeger. Probability distributions of optical flow. In *IEEE Proc. of CVPR*, pages 310–315, Maui, Hawaii, June 1991.

[56] S. A. Sirohey. Human face segmentation and identification. Technical Report CAR-TR-695 and CS-TR-3176, Computer Vision Laboratory, Center for Automation Research , Univ. of Maryland, Nov. 1993.

[57] S. Tanimoto and T. Pavlidis. A hierarchical data structure for picture processing. *Computer Graphics and Image Processing*, 4(2):104–119, 1975.

[58] S. Toelg. Video conferencing: An application of learning. In *Proceedings of the CBCL Learning Day*, Endicott House, MIT, Apr. 1993.

[59] S. Tölg. Gaze control for an active camera system by modeling human pursuit eye movements. In D. P. Casasent, editor, *Intelligent Robots and Computer Vision XI: Algorithms, Techniques, and Active Vision*, volume 1825, pages 585–598, Boston, MA, Nov. 1992. SPIE—The International Society for Optical Engineering.

[60] S. Tölg. *Strukturuntersuchungen zur Informationsverar-
beitung in neuronaler Architektur am Beispiel der Modellierung von Augenbewegungen für aktives Sehen*, volume 197 of *Fortschritt-Berichte VDI, Reihe 10: Informatik/Kommunikationstechnik.* VDI Verlag, Düsseldorf, 1992. ISBN 3-18-149710-X.

[61] M. Turk and A. Pentland. Face recognition using eigenfaces. In *IEEE Proc. of CVPR*, pages 586–591, Maui, Hawaii, June 1991.

[62] S. Ullman and R. Basri. Recognition by linear combinations of models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(10):992–1006, 1991. Also MIT AI Memo 1051, 1989.

[63] S. Uras, F. Girosi, A. Verri, and V. Torre. A computational approach to motion perception. *Kybernetik / Biol. Cybernetics*, 60:79–87, 1988.

[64] G. Wolberg. *Digital Image Warping.* IEEE Computer Society Order Number 1944, 1990.

[65] Y. Yacoob and L. S. Davis. Computing spatio-temporal representations of human faces. In *IEEE Proc. of CVPR*, Seattle, WA, June 1994.

A Hierarchical estimation of global pose from local displacements

We assume an affine model for the displacement vector field. The affine displacement field $\hat{\mathbf{d}}(\mathbf{x}_i)$ is determined at any image location $\mathbf{x}_i = (x_i, y_i)^\top$ by six model parameters:

$$\hat{\mathbf{d}}(\mathbf{x}_i) = \mathbf{A}\mathbf{x}_i + \mathbf{t} \tag{11}$$

with

$$\mathbf{A} = \begin{pmatrix} b_x & c_x \\ b_y & c_y \end{pmatrix} \quad \text{and} \quad \mathbf{t} = \begin{pmatrix} a_x \\ a_y \end{pmatrix}. \tag{12}$$

Suppose we have n image points $\mathbf{x}_i \in \Omega$ $(i = 1, \ldots, n)$ with a measured displacement $\mathbf{d}(\mathbf{x}_i) = (d_x(x_i, y_i), d_x(x_i, y_i))^\top$.

We obtain an overdetermined linear system with $2n$ equations and six unknowns that can be written in matrix form:

$$\mathbf{M}\mathbf{p} = \mathbf{d} \tag{13}$$

with

$$\mathbf{M} = \begin{pmatrix} 1 & x_1 & y_1 & & & \\ \vdots & \vdots & \vdots & & \vdots & \\ 1 & x_i & y_i & \cdots & 0 & \cdots \\ \vdots & \vdots & \vdots & & \vdots & \\ 1 & x_n & y_n & & & \\ & & & 1 & x_1 & y_1 \\ & \vdots & & \vdots & \vdots & \vdots \\ \cdots & 0 & \cdots & 1 & x_i & y_i \\ & \vdots & & \vdots & \vdots & \vdots \\ & & & 1 & x_n & y_n \end{pmatrix} \in \mathbb{R}^{2n \times 6} \quad , \quad \mathbf{p} = \begin{pmatrix} a_x \\ b_x \\ c_x \\ a_y \\ b_y \\ c_y \end{pmatrix} \in \mathbb{R}^6 \quad \text{and} \quad \mathbf{d} = \begin{pmatrix} d_x(x_1, y_1) \\ \vdots \\ d_x(x_i, y_i) \\ \vdots \\ d_x(x_n, y_n) \\ d_y(x_1, y_1) \\ \vdots \\ d_y(x_i, y_i) \\ \vdots \\ d_y(x_n, y_n) \end{pmatrix} \in \mathbb{R}^{2n}. \tag{14}$$

In general this system cannot be solved exactly. Instead, we want to find the parameter vector $\mathbf{p}$ that minimizes the error between the measured displacement field $\mathbf{d}(\mathbf{x}_i)$ and the fitted affine displacement field $\hat{\mathbf{d}}(\mathbf{x}_i)$. We assume Gaussian statistics of the process and use the L_2-norm as a distance measure. So, we want to find

$$\min_{\mathbf{d}} e = \sum_i w_i^2 \left\| \mathbf{d}(\mathbf{x}_i) - \hat{\mathbf{d}}(\mathbf{x}_i) \right\|_2^2, \tag{15}$$

where we allow for a weight w_i^2 for each data point $\mathbf{x}_i$. These weights may account for the confidence that we associate with the measurement $\mathbf{d}(\mathbf{x}_i)$.

With

$$\mathbf{W} = \begin{pmatrix} w_1 & \cdots & 0 & & \vdots & \\ \vdots & w_i & \vdots & \cdots & 0 & \cdots \\ 0 & \cdots & w_n & & \vdots & \\ & \vdots & & w_1 & \cdots & 0 \\ \cdots & 0 & \cdots & \vdots & w_i & \vdots \\ & \vdots & & 0 & \cdots & w_n \end{pmatrix} \in \mathbb{R}^{2n \times 2n} \tag{16}$$

the solution of the minimization problem (15) formally is

$$\mathbf{p} = \mathbf{M}^* \mathbf{d}, \tag{17}$$

where $\mathbf{M}^* = (\mathbf{M}^\top \mathbf{W}^2 \mathbf{M})^{-1} \mathbf{M}^\top \mathbf{W}^2$ is the weighted pseudo-inverse. This can be written as

$$\underbrace{\mathbf{M}^\top \mathbf{W}^2 \mathbf{M}}_{\mathbf{B}} \mathbf{p} = \underbrace{\mathbf{M}^\top \mathbf{W}^2 \mathbf{d}}_{\mathbf{b}} \quad \Rightarrow \quad \mathbf{B}\mathbf{p} = \mathbf{b} \tag{18}$$

Now, $\mathbf{b} \in \mathbb{R}^6$ and $\mathbf{B} \in \mathbb{R}^{6 \times 6}$ is a square matrix. The system can be solved for the parameter vector $\mathbf{p} = \mathbf{B}^{-1}\mathbf{b}$ provided that $\det(\mathbf{B}) \neq 0$ and therefore the inverse $\mathbf{B}^{-1}$ exists. For reasons of numerical accuracy and stability one would generally prefer to solve the overdetermined system by means of the computationally more costly singular value decomposition (SVD) (cf. [49]). However, our relatively simple model is well-behaved and it turn out that in the implemented case the matrix inversion is trivial.

A.1 General case

In the general case of an affine displacement field with six free parameters we have $\mathbf{p}_6 = \mathbf{B}_6^{-1}\mathbf{b}_6$ with

$$\mathbf{B}_6 = \begin{pmatrix} \sum w_i^2 & \sum w_i^2 x_i & \sum w_i^2 y_i & & \vdots & \\ \sum w_i^2 x_i & \sum w_i^2 x_i^2 & \sum w_i^2 x_i y_i & \cdots & 0 & \cdots \\ \sum w_i^2 y_i & \sum w_i^2 x_i y_i & \sum w_i^2 y_i^2 & & \vdots & \\ & \vdots & & \sum w_i^2 & \sum w_i^2 x_i & \sum w_i^2 y_i \\ \cdots & 0 & \cdots & \sum w_i^2 x_i & \sum w_i^2 x_i^2 & \sum w_i^2 x_i y_i \\ & \vdots & & \sum w_i^2 y_i & \sum w_i^2 x_i y_i & \sum w_i^2 y_i^2 \end{pmatrix}, \tag{19}$$

$$\mathbf{p}_6 = \mathbf{p} \quad \text{and} \quad \mathbf{b}_6 = \begin{pmatrix} \sum w_i^2 d_x(x_i, y_i) \\ \sum w_i^2 d_x(x_i, y_i) x_i \\ \sum w_i^2 d_x(x_i, y_i) y_i \\ \sum w_i^2 d_y(x_i, y_i) \\ \sum w_i^2 d_y(x_i, y_i) x_i \\ \sum w_i^2 d_y(x_i, y_i) y_i \end{pmatrix}. \tag{20}$$

A.2 No shear

If we admit only translation, scale and rotation and do not allow for any component of shear, $\mathbf{A}$ in (12) takes the form

$$\mathbf{A} = \mathbf{SR} - \mathbf{I} \tag{21}$$

with

$$\mathbf{S} = \begin{pmatrix} s & 0 \\ 0 & s \end{pmatrix} \quad , \quad \mathbf{R} = \begin{pmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{pmatrix} \quad \text{and} \quad \mathbf{I} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \tag{22}$$

Consequently (11) becomes

$$\hat{\mathbf{d}}(\mathbf{x}_i) = (\mathbf{SR} - \mathbf{I})\mathbf{x}_i + \mathbf{t}. \tag{23}$$

Therefore, we have the constraint for the parameters:

$$b_x = c_y = s \cdot \cos\alpha - 1 \quad \text{and} \quad -b_y = c_x = s \cdot \sin\alpha. \tag{24}$$

The scale s and the angle of rotation α in the image plane are derived from $\mathbf{A}$ as

$$s = \sqrt{\det(\mathbf{A} + \mathbf{I})} \quad \text{and} \quad \alpha = \arccos\left(\frac{b_x + 1}{s}\right). \tag{25}$$

Of course, $s = 1$ and $\alpha = 0$ yields a constant displacement field.

With the constraints in (24), the linear system (18) can be simplified by adding the 2nd and 6th row and subtracting the 5th from the 3rd row of (19) and (20). This gives $\mathbf{p}_4 = \mathbf{B}_4^{-1}\mathbf{b}_4$ with

$$\mathbf{B}_4 = \begin{pmatrix} \sum w_i^2 & \sum w_i^2 x_i & \sum w_i^2 y_i & 0 \\ \sum w_i^2 x_i & (\sum w_i^2 x_i^2 + \sum w_i^2 y_i^2) & 0 & \sum w_i^2 y_i \\ \sum w_i^2 y_i & 0 & (\sum w_i^2 x_i^2 + \sum w_i^2 y_i^2) & -\sum w_i^2 x_i \\ 0 & \sum w_i^2 y_i & -\sum w_i^2 x_i & \sum w_i^2 \end{pmatrix}, \tag{26}$$

$$\mathbf{p}_4 = \begin{pmatrix} a_x \\ b_x \\ c_x \\ a_y \end{pmatrix} \quad , \text{and} \quad \mathbf{b}_4 = \begin{pmatrix} \sum w_i^2 d_x(x_i, y_i) \\ \sum w_i^2 d_x(x_i, y_i) x_i + \sum w_i^2 d_y(x_i, y_i) y_i \\ \sum w_i^2 d_x(x_i, y_i) y_i - \sum w_i^2 d_y(x_i, y_i) x_i \\ \sum w_i^2 d_y(x_i, y_i) \end{pmatrix}. \tag{27}$$

A.3 Pure translation

If we admit only pure translation ($c_y = b_x = b_y = c_x = 0$) the displacement field is of course constant $\hat{\mathbf{d}}(\mathbf{x}_i) = \mathbf{t}$. In this case we have $\mathbf{B}_2\mathbf{p}_2 = \mathbf{b}_2$, where

$$\mathbf{B}_2 = \begin{pmatrix} \sum w_i^2 & 0 \\ 0 & \sum w_i^2 \end{pmatrix} \quad , \quad \mathbf{p}_2 = \begin{pmatrix} a_x \\ a_y \end{pmatrix} \quad , \text{and} \quad \mathbf{b}_2 = \begin{pmatrix} \sum w_i^2 d_x(x_i, y_i) \\ \sum w_i^2 d_y(x_i, y_i) \end{pmatrix} \tag{28}$$

are derived from the 1st and 4th row of the general case equation system (19) and (20). Since $\mathbf{B}_2 = \sum w_i^2 \mathbf{I}$ we can solve $\mathbf{B}_2\mathbf{p}_2 = \mathbf{b}_2$ directly:

$$\mathbf{p}_2 = \frac{1}{\sum w_i^2}\mathbf{b}_2; \tag{29}$$

which is equivalent to

$$a_x = \frac{\sum w_i^2 d_x(x_i, y_i)}{\sum w_i^2} \quad \text{and} \quad a_y = \frac{\sum w_i^2 d_y(x_i, y_i)}{\sum w_i^2}. \tag{30}$$

A.4 New reference frame

As we will see, several expressions can be simplified significantly if we express the affine displacement field in a new frame of reference. In order to do so, we introduce a new function

$$\hat{\mathbf{d}}'(\mathbf{x}'_i) = \mathbf{A}'\mathbf{x}'_i + \mathbf{t}' = \mathbf{A}\mathbf{x}_i + \mathbf{t} = \hat{\mathbf{d}}(\mathbf{x}_i) \tag{31}$$

expressing the affine displacement field in terms of $\mathbf{A}'$ and $\mathbf{t}'$. We performed a coordinate transform, so that the origin of the new reference frame coincides with the center of gravity of the image:

$$\mathbf{x}'_i = \mathbf{x}_i - \bar{\mathbf{x}} \quad \text{with} \quad \bar{\mathbf{x}} = \frac{\sum w_i^2 \mathbf{x}_i}{\sum w_i^2}. \tag{32}$$

In the new reference frame the equation systems take a much simpler form, since

$$\sum_i w_i^2 \mathbf{x}'_i = \sum_i w_i^2 \mathbf{x}_i - \bar{\mathbf{x}} \sum_i w_i^2 = 0. \tag{33}$$

Using (31) and (15) we now want to minimize the expression

$$e = \sum_i w_i^2 \left\| \mathbf{d}(\mathbf{x}_i) - \hat{\mathbf{d}}'(\mathbf{x}'_i) \right\|_2^2 \tag{34}$$

with respect to the new parameters a'_x, b'_x, c'_x and a'_y, b'_y, c'_y.

A.4.1 General case

From (31) and (32) we can directly derive the old parameters from the new ones:

$$\mathbf{A}'\mathbf{x}'_i + \mathbf{t}' = \mathbf{A}'(\mathbf{x}_i - \bar{\mathbf{x}}) + \mathbf{t}' = \mathbf{A}'\mathbf{x}_i + (\mathbf{t}' - \mathbf{A}'\bar{\mathbf{x}}) = \mathbf{A}\mathbf{x}_i + \mathbf{t} \tag{35}$$

Comparison of the coefficients on both sides yields (and similarily for a_y, b_y, c_y):

$$a_x = a'_x - b'_x \bar{x} - c'_x \bar{y} \quad , \quad b_x = b'_x \quad \text{and} \quad c_x = c'_x. \tag{36}$$

Because of (33), in the new reference frame (19) and (20) simplify to

$$\mathbf{B}'_6 = \begin{pmatrix} \sum w_i^2 & 0 & 0 & & \vdots & \\ 0 & \sum w_i^2 {x'_i}^2 & \sum w_i^2 x'_i y'_i & \cdots & 0 & \cdots \\ 0 & \sum w_i^2 x'_i y'_i & \sum w_i^2 {y'_i}^2 & & \vdots & \\ & \vdots & & \sum w_i^2 & 0 & 0 \\ \cdots & 0 & \cdots & 0 & \sum w_i^2 {x'_i}^2 & \sum w_i^2 x'_i y'_i \\ & \vdots & & 0 & \sum w_i^2 x'_i y'_i & \sum w_i^2 {y'_i}^2 \end{pmatrix} \tag{37}$$

and

$$\mathbf{b}'_6 = \begin{pmatrix} \sum w_i^2 d_x(x_i, y_i) \\ \sum w_i^2 d_x(x_i, y_i) x'_i \\ \sum w_i^2 d_x(x_i, y_i) y'_i \\ \sum w_i^2 d_y(x_i, y_i) \\ \sum w_i^2 d_y(x_i, y_i) x'_i \\ \sum w_i^2 d_y(x_i, y_i) y'_i \end{pmatrix}. \tag{38}$$

From which we obtain a'_x directly:

$$a'_x = \frac{\sum w_i^2 d_x(x_i, y_i)}{\sum w_i^2} \tag{39}$$

and by inversion the remaining 2×2 matrix we get the solutions for b'_x, c'_x in closed form:

$$b'_x = \frac{\sum w_i^2 {y'_i}^2 \sum w_i^2 d_x(x_i, y_i) x'_i - \sum w_i^2 x'_i y'_i \sum w_i^2 d_x(x_i, y_i) y'_i}{\sum w_i^2 {x'_i}^2 \sum w_i^2 {y'_i}^2 - (\sum w_i^2 x'_i y'_i)^2} \tag{40}$$

$$c'_x = \frac{\sum w_i^2 {x'_i}^2 \sum w_i^2 d_y(x_i, y_i) y'_i - \sum w_i^2 x'_i y'_i \sum w_i^2 d_y(x_i, y_i) x'_i}{\sum w_i^2 {x'_i}^2 \sum w_i^2 {y'_i}^2 - (\sum w_i^2 x'_i y'_i)^2}. \tag{41}$$

The solutions for a'_y, b'_y, c'_y are found similarly.

A.4.2 No shear

In the new reference frame (26) and (27) simplify to

$$\mathbf{B}'_4 = \begin{pmatrix} \sum w_i^2 & 0 & 0 & 0 \\ 0 & (\sum w_i^2 {x'_i}^2 + \sum w_i^2 {y'_i}^2) & 0 & 0 \\ 0 & 0 & (\sum w_i^2 {x'_i}^2 + \sum w_i^2 {y'_i}^2) & 0 \\ 0 & 0 & 0 & \sum w_i^2 \end{pmatrix} \tag{42}$$

and

$$\mathbf{b}'_4 = \begin{pmatrix} \sum w_i^2 d_x(x_i, y_i) \\ \sum w_i^2 d_x(x_i, y_i) x'_i + \sum w_i^2 d_y(x_i, y_i) y'_i \\ \sum w_i^2 d_x(x_i, y_i) y'_i - \sum w_i^2 d_y(x_i, y_i) x'_i \\ \sum w_i^2 d_y(x_i, y_i) \end{pmatrix}. \tag{43}$$

$\mathbf{B}'_4$ is now diagonal and $\mathbf{B}'_4 \mathbf{p}'_4 = \mathbf{b}'_4$ can be directly solved for the parameters:

$$a'_x = \frac{\sum w_i^2 d_x(x_i, y_i)}{\sum w_i^2} \quad , \quad a'_y = \frac{\sum w_i^2 d_y(x_i, y_i)}{\sum w_i^2}, \tag{44}$$

$$b'_x = c'_y = \frac{\sum w_i^2 d_x(x_i, y_i) x'_i + \sum w_i^2 d_y(x_i, y_i) y'_i}{\sum w_i^2 {x'_i}^2 + \sum w_i^2 {y'_i}^2}, \tag{45}$$

and

$$-b'_y = c'_x = \frac{\sum w_i^2 d_x(x_i, y_i) y'_i - \sum w_i^2 d_y(x_i, y_i) x'_i}{\sum w_i^2 {x'_i}^2 + \sum w_i^2 {y'_i}^2}. \tag{46}$$

A.5 Homogeneous error of the fit

In order to assess how well the estimated affine motion $\hat{\mathbf{d}}'(\mathbf{x}'_i)$ describes the measured displacement field $\mathbf{d}(\mathbf{x}_i)$ let us now derive the homogeneous error of the fit. Setting all weights $w_i^2 = 1$ we obtain from (34):

$$he = \sum_i \left\| \mathbf{d}(\mathbf{x}_i) - \hat{\mathbf{d}}'(\mathbf{x}'_i) \right\|_2^2 = \sum_i \left(\mathbf{d}(\mathbf{x}_i) - (\mathbf{A}'\mathbf{x}'_i + \mathbf{t}') \right)^2 \tag{47}$$

Note, that because of the L_2-norm, e can be decomposed into the errors of the x and y component:

$$he = he_x + he_y. \tag{48}$$

The mean deviation of the measured displacements from the fitted affine displacement field is estimated by the homogeneous variance

$$\mathrm{var}_h = \frac{he}{n-2}. \tag{49}$$

In the general case we obtain for the error of the x component:

$$\begin{aligned} he_x = & \sum d_x^2(x_i, y_i) \\ & - 2a'_x \sum d_x(x_i, y_i) - 2b'_x \sum d_x(x_i, y_i) x'_i - 2c'_x \sum d_x(x_i, y_i) y'_i \\ & + 2a'_x b'_x \sum x'_i + 2a'_x c'_x \sum y'_i + 2b'_x c'_x \sum x'_i y'_i \\ & + {a'_x}^2 n + {b'_x}^2 \sum {x'_i}^2 + {c'_x}^2 \sum {y'_i}^2, \end{aligned} \tag{50}$$

and similarly for he_y, where n is the number of data points.

A.6 Weighted error of the fit

The mean deviation of the measured displacements $\mathbf{d}(\mathbf{x}_i)$ from the fitted affine displacement field $\hat{\mathbf{d}}'(\mathbf{x}'_i)$ is estimated by the weighted variance approximated by

$$\mathrm{var}_w \approx \frac{we}{\sum w_i^2}. \tag{51}$$

From (34) we get the weighted error of the fit:

$$we = \sum_i w_i^2 \left\| \mathbf{d}(\mathbf{x}_i) - \hat{\mathbf{d}}'(\mathbf{x}'_i) \right\|_2^2 = \sum_i w_i^2 \left(\mathbf{d}(\mathbf{x}_i) - (\mathbf{A}'\mathbf{x}'_i + \mathbf{t}') \right)^2. \tag{52}$$

Note, that we can be decomposed again into the errors of the x and y components:

$$we = we_x + we_y. \tag{53}$$

A.6.1 General case

In the general case we get for the error of the x component:

$$\begin{aligned} we_x &= \sum w_i^2 d_x^2(x_i, y_i) \\ &\quad - 2a'_x \sum w_i^2 d_x(x_i, y_i) - 2b'_x \sum w_i^2 d_x(x_i, y_i) x'_i - 2c'_x \sum w_i^2 d_x(x_i, y_i) y'_i \\ &\quad + 2a'_x b'_x \sum w_i^2 x'_i + 2a'_x c'_x \sum w_i^2 y'_i + 2b'_x c'_x \sum w_i^2 x'_i y'_i \\ &\quad + {a'_x}^2 \sum w_i^2 + {b'_x}^2 \sum w_i^2 {x'_i}^2 + {c'_x}^2 \sum w_i^2 {y'_i}^2, \end{aligned} \tag{54}$$

and similarly for we_y. Since $\sum w_i^2 x'_i = \sum w_i^2 y'_i = 0$ and with the solution for a'_x several terms cancel out and we get

$$\begin{aligned} we_x &= \sum w_i^2 d_x^2(x_i, y_i) + 2b'_x c'_x \sum w_i^2 x'_i y'_i \\ &\quad - 2b'_x \sum w_i^2 d_x(x_i, y_i) x'_i - 2c'_x \sum w_i^2 d_x(x_i, y_i) y'_i \\ &\quad - {a'_x}^2 \sum w_i^2 + {b'_x}^2 \sum w_i^2 {x'_i}^2 + {c'_x}^2 \sum w_i^2 {y'_i}^2, \end{aligned} \tag{55}$$

A.6.2 No shear

With $c_y = b_x$ and $b_y = -c_x$ we get from (53) and (55)

$$\begin{aligned} we &= \sum w_i^2 d_x^2(x_i, y_i) + \sum w_i^2 d_y^2(x_i, y_i) \\ &\quad - 2b'_x \left(\sum w_i^2 d_x(x_i, y_i) x'_i + \sum w_i^2 d_y(x_i, y_i) y'_i \right) \\ &\quad - 2c'_x \left(\sum w_i^2 d_x(x_i, y_i) y'_i - \sum w_i^2 d_y(x_i, y_i) x'_i \right) \\ &\quad - ({a'_x}^2 + {a'_y}^2) \sum w_i^2 + ({b'_x}^2 + {c'_x}^2) \left(\sum w_i^2 {x'_i}^2 + \sum w_i^2 {y'_i}^2 \right). \end{aligned} \tag{56}$$

This can be further simplified using solutions for b'_x, c'_x:

$$\begin{aligned} we &= \sum w_i^2 d_x^2(x_i, y_i) + \sum w_i^2 d_y^2(x_i, y_i) \\ &\quad - ({a'_x}^2 + {a'_y}^2) \sum w_i^2 - ({b'_x}^2 + {c'_x}^2) \left(\sum w_i^2 {x'_i}^2 + \sum w_i^2 {y'_i}^2 \right). \end{aligned} \tag{57}$$

A.6.3 Pure translation

In the case of pure translation ($c_y = b_x = b_y = c_x = 0$) we see from (36) that $a_x = a'_x$ and $a_y = a'_y$. The error of the fit is then given by

$$we = \sum w_i^2 d_x^2(x_i, y_i) + \sum w_i^2 d_y^2(x_i, y_i) - (a_x^2 + a_y^2) \sum w_i^2. \tag{58}$$

A.7 Propagation of affine parameters from coarse to fine levels

In the sequel we derive, how to propagate the affine motion parameters from a coarse pyramid level to the next finer level. The affine displacement field

$$\hat{\mathbf{d}}^l(\mathbf{x}_i^l) = \mathbf{A}^l \mathbf{x}_i^l + \mathbf{t}^l \tag{59}$$

at the level l of the pyramid is determined at any image location $\mathbf{x}_i^l = (x_i^l, y_i^l)^\top$ by six parameters:

$$\mathbf{A}^l = \begin{pmatrix} b_x^l & c_x^l \\ b_y^l & c_y^l \end{pmatrix} \quad \text{and} \quad \mathbf{t}^l = \begin{pmatrix} a_x^l \\ a_y^l \end{pmatrix}. \tag{60}$$

For the previous coarser level $l+1$ with lower resolution we have accordingly

$$\hat{\mathbf{d}}^{l+1}(\mathbf{x}_i^{l+1}) = \mathbf{A}^{l+1} \mathbf{x}_i^{l+1} + \mathbf{t}^{l+1}. \tag{61}$$

Now, since the sampling grid of level l has twice the density than at the coarser level $l+1$, the coordinates of corresponding points and their displacements are related by

$$\mathbf{x}_i^l = 2 \cdot \mathbf{x}_i^{l+1} \quad \text{and} \quad \hat{\mathbf{d}}^l(\mathbf{x}_i^l) = 2 \cdot \hat{\mathbf{d}}^{l+1}(\mathbf{x}_i^{l+1}). \tag{62}$$

Inserting this into (59) leads to

$$\hat{\mathbf{d}}^{l+1}(\mathbf{x}_i^{l+1}) = \mathbf{A}^l \mathbf{x}_i^{l+1} + \frac{1}{2} \mathbf{t}^l. \tag{63}$$

Comparing this with (61) yields

$$\mathbf{A}^l = \mathbf{A}^{l+1} \quad \text{and} \quad \mathbf{t}^l = 2 \cdot \mathbf{t}^{l+1}. \tag{64}$$

Therefore, to propagate the affine parameters from level $l+1$ to the next finer level l we have only to double the translation vector $\mathbf{t}^{l+1}$. The coefficients of matrix $\mathbf{A}^{l+1}$ remain unchanged.

A.8 Combining the affine parameters at one level

The combination of the initial affine parameters on level l (propagated from level $l+1$) with the parameters of the residual affine motion on level l (estimated from the residual OF at level l) is now considered. The refined displacement field $\hat{\mathbf{d}}_r^l$ at level l is given by

$$\hat{\mathbf{d}}_r^l = \hat{\mathbf{d}}^l + \tilde{\mathbf{d}}^l, \tag{65}$$

where $\tilde{\mathbf{d}}^l(\mathbf{x}_i^l) = \tilde{\mathbf{A}}^l\mathbf{x}_i^l + \tilde{\mathbf{t}}^l$ is the residual affine displacement field estimated at level l. With (59) we get because of the linearity of the affine transformation:

$$\hat{\mathbf{d}}_r^l(\mathbf{x}_i^l) = (\mathbf{A}^l\mathbf{x}_i^l + \mathbf{t}^l) + (\tilde{\mathbf{A}}^l\mathbf{x}_i^l + \tilde{\mathbf{t}}^l) = (\mathbf{A}^l + \tilde{\mathbf{A}}^l)\mathbf{x}_i^l + (\mathbf{t}^l + \tilde{\mathbf{t}}^l) = \mathbf{A}_r^l\mathbf{x}_i^l + \mathbf{t}_r^l. \tag{66}$$

Therefore, the refined affine parameters $(\mathbf{A}_r^l, \mathbf{t}_r^l)$ at level l are given as the sum of the propagated parameters $(\mathbf{A}^l, \mathbf{t}^l)$ and the residual parameters $(\tilde{\mathbf{A}}^l, \tilde{\mathbf{t}}^l)$ estimated at level l.

B Inversion of affine pose transformation

The transformation defined in (11) and (12) gives us the displacements from the example image E to the new image I. This allows us to warp the face in I towards the normalized pose of the example image E. The position of a point $\mathbf{x}_i' = (x_i', y_i')^\top$ in I is derived from the affine displacement field $\hat{\mathbf{d}}(\mathbf{x}_i)$ that maps $\mathbf{x}_i \rightarrow \mathbf{x}_i'$ by

$$\mathbf{x}_i' = \hat{\mathbf{d}}(\mathbf{x}_i) + \mathbf{x}_i = (\mathbf{A} + \mathbf{I})\mathbf{x}_i + \mathbf{t}. \tag{67}$$

Here, the prime indicates that we use image I as a reference frame.

On the receiver side we are faced with the problem of reversing this pose normalization. We have a face image E with normalized pose (indexed in the database, or interpolated between several examples) and we want to generate an image I according to the transmitted pose parameters of our model. We are now looking for the displacement field $\hat{\mathbf{d}}'(\mathbf{x}_i')$ that maps $\mathbf{x}_i' \rightarrow \mathbf{x}_i$. For our affine transformation model there is a closed form solution. Provided that $\det(\mathbf{A} + \mathbf{I}) \neq 0$ the inverse matrix exists, and we obtain from (67) the location $\mathbf{x}_i$ in the normalized image E by

$$\mathbf{x}_i = (\mathbf{A} + \mathbf{I})^{-1}\mathbf{x}_i' - (\mathbf{A} + \mathbf{I})^{-1}\mathbf{t}. \tag{68}$$

With the definition

$$\hat{\mathbf{d}}'(\mathbf{x}_i') = \mathbf{A}'\mathbf{x}_i' + \mathbf{t}' = \mathbf{x}_i - \mathbf{x}_i' \tag{69}$$

we conclude from (68) that

$$\mathbf{A}' = (\mathbf{A} + \mathbf{I})^{-1} - \mathbf{I} \quad \text{and} \quad \mathbf{t}' = -(\mathbf{A} + \mathbf{I})^{-1}\mathbf{t}. \tag{70}$$

Therefore, expressed in terms of the model parameters, $\mathbf{A}'$ is given by

$$\mathbf{A}' = \frac{1}{\det(\mathbf{A} + \mathbf{I})} \begin{pmatrix} c_y + 1 & -c_x \\ -b_y & b_x + 1 \end{pmatrix} - \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \tag{71}$$

Figure 3: Depicted are several facial expressions that may occur during a video-conferencing session. All images exhibit a roughly frontal view of the face. Conventional approaches to normalize face images utilize labeled feature points, like the pupils of the eyes or the tips of the mouth. The first image shows a neutral facial expression with gaze in the forward direction. This is the standard case that most face recognition systems are designed to cope with. The second row demonstrates the movement of both eyes due to changes in direction of gaze (conjugate eye-movements); vergence movements (disjunctive eye-movements) alter the distance between the pupils. The positions of these points (centers of the pupils) may differ by more than 1 cm on either side of the forward direction. This is a large fraction of the inter-ocular distance of about 7 cm. The last row depicts the movement of the corners of the mouth due to skin deformations caused by facial expressions and normal speech. As is evident, estimating the pose based on the correspondence of these points is rather unreliable if facial expressions are admissible. Nevertheless, such feature points are commonly used for normalizing face images with moderate deviations from neutral expressions. Finally, the pupils may entirely disappear when the eyelid is closed during twinkling or blinking. A pose estimation method relying on the correct detection of these feature points would be led astray.

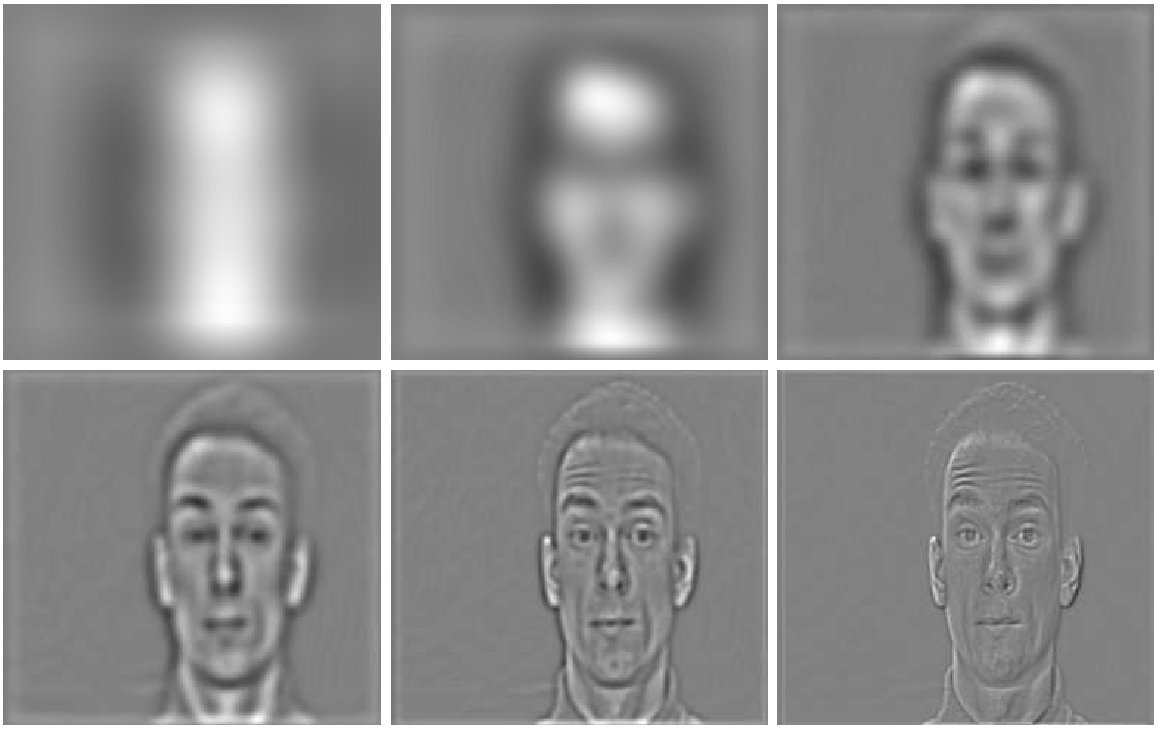

Figure 4: Band-pass filtered images of a face (in front of a dark background). A Difference-of-Gaussians (DOG) filter approximating a Laplacian-of-Gaussian operator of width σ ("Mexican-hat filter") is applied to the original image. The images are arranged in descending order of σ starting with $\sigma = 64.0$ pixels for the upper left image and ending with $\sigma = 2.0$ pixels for the lower right image; σ increases by one octave between consecutive images. Facial expressions and details of the face (eyes, mouth, etc.) are most conspicuous in the high-frequency images, whereas the overall position, size, and orientation of the head appear dominant in the low-frequency bands.

Figure 5: Two face images with similar expressions, but different pose and size, used for demonstrating the robust pose estimation and compensation algorithm. The left image is the reference image, e.g., stored in the example database, the right one is a new frame of the incoming data stream that has to be normalized in pose. All images used in the sequel are 255×320 pixels in size and digitized with 8-bit quantization. The focal length of the camera was approximately 16 mm and the camera distance was about 1.2 m. The face on the right side is inclined by 8–9° and is about 10% smaller than that in the reference image. These values are obtained by direct reading from the images.

Figure 6: Here the right image of Figure 5 is transformed to resemble the pose of the reference image. Pose parameters have been computed automatically by the algorithm described in Section 4.1. The six images show how the results depend on the resolution level where the estimation is terminated. Subsequently, the pose parameters are extrapolated to the original resolution and the images are warped accordingly. The lowest resolution is level 5 (upper left); the original image resolution is at level 0 (lower right). By visual inspection it is obvious that level 3 (upper right) already achieves good alignment of the face with the reference image. The resulting image becomes stationary and estimation at higher resolutions does not lead to significant improvement. For more quantitative details see Figure 7 and Table 2.

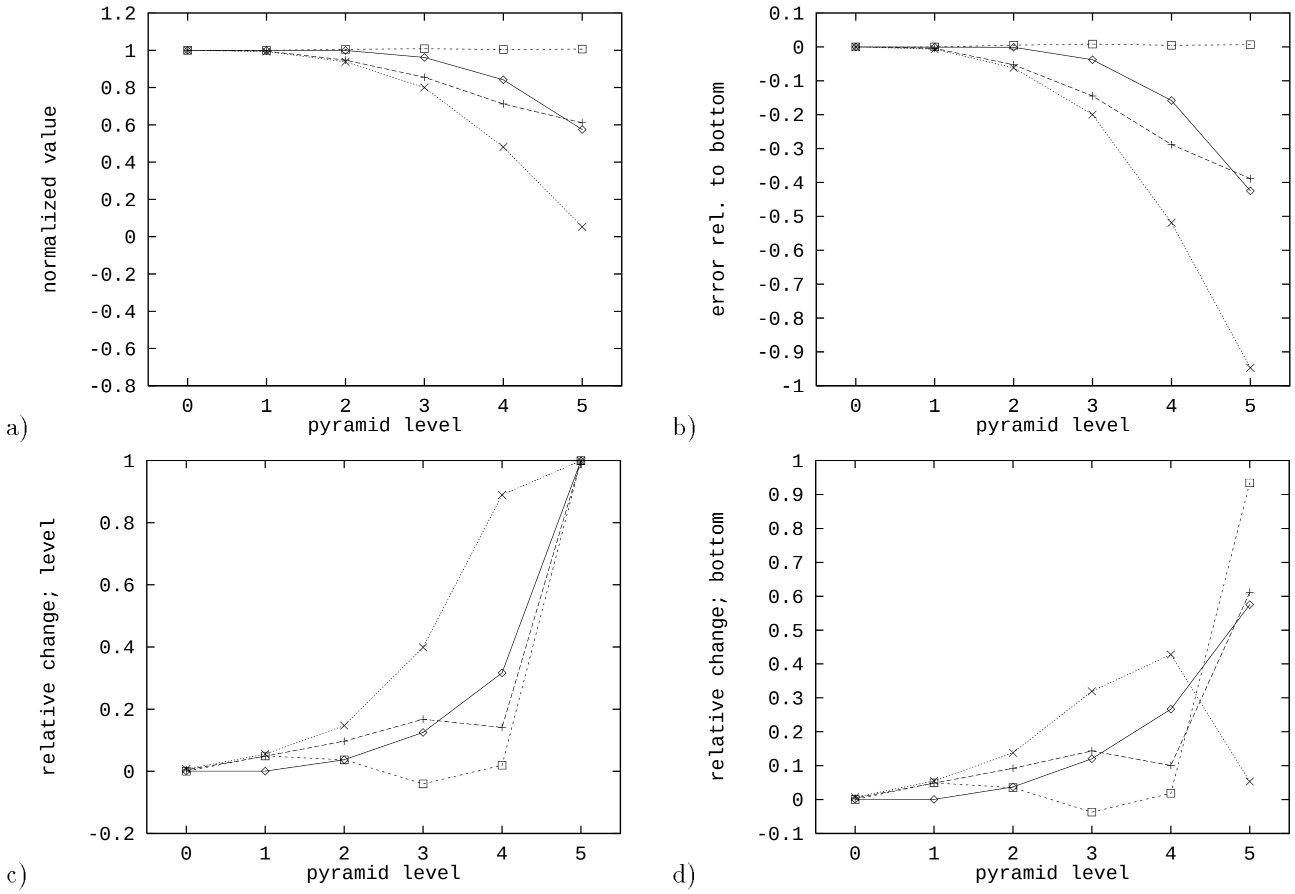

Figure 7: $a_x : \diamond$ $a_y : +$ $s : \square$ $\alpha : \times$
Different graphic representations of the estimated pose parameters given in Table 2. Investigated is the dependence of the accuracy on the pyramid level where the estimation is terminated. The diagrams show: a) pose parameters at each final level normalized by their bottom values (level 0); b) error to highest resolution estimate normalized by the bottom value; c) relative change between successive levels normalized by the value of the lower current level; d) relative change between successive levels normalized by the bottom value. The notation is the same as is used throughout the text: a_x and a_y are the horizontal and vertical translation, respectively; s is the scale factor; α is the angle of in-plane rotation. The more careful analysis confirms the results already discussed in Figure 6. The estimates converge to the values at the highest resolution. At level 2 the estimated parameters are already very close to the values obtained at the highest resolution (level 0). The largest changes in the parameters happen already at the high pyramid levels with low resolutions (see lower row).

Table 2: Pose parameters estimated by the algorithm. The values in each row correspond to the images depicted in Figure 6, where l is the final level for estimation before the parameters are propagated to the original resolution at level 0. The notation for the parameters is the same as throughout the text: a_x and a_y are the horizontal and vertical translation, respectively; s is the scale factor; α is the angle of in-plane rotation in radians. The last two columns give the weighted variance (var_w) and the homogeneous variance (var_h) as discussed in Section 4.3.

l	a_x	a_y	s	α	var_w	var_h
5	-23.59	26.40	0.9162	0.0072	0.3901	0.4009
4	-34.52	30.74	0.9146	0.0654	0.2871	0.6221
3	-39.46	36.92	0.9179	0.1088	0.4027	0.7876
2	-40.98	40.90	0.9148	0.1275	0.7118	1.0114
1	-41.00	43.00	0.9104	0.1350	1.2381	1.0830
0	-41.20	43.18	0.9102	0.1359	0.9217	0.7124

Table 3: Pose parameters at the bottom level ($l = 0$) depending on the number of iterations i per level estimated for the image pair in Figure 5. The estimates vary very little for increasing number of iterations. The largest changes (for a_y and α between $i = 1$ and 2) are on the order of 5%. var_w and var_h are the weighted and the homogeneous variance, respectively.

i	a_x	a_y	s	α	var_w	var_h
1	-41.20	43.18	0.9102	0.1359	0.9217	0.7124
2	-41.29	45.41	0.9076	0.1465	0.8998	0.6925
3	-41.34	46.14	0.9067	0.1505	0.8969	0.6860
5	-41.40	47.00	0.9057	0.1557	0.8959	0.6778
10	-41.46	47.40	0.9055	0.1580	0.8984	0.6778

Figure 8: Two other face images. This example is somewhat more "difficult" as compared to Figure 5, since the facial expressions in both images are quite different. Again, the left image represents the reference pose. An inclination of 6–7° and an increase in size of about 7–8% can be measured directly by comparing both images.

Figure 9: See caption of Figure 6. The results in this figure are for the two images in Figure 8. By visual inspection no significant change occurs for final estimation levels higher than two (lower left).

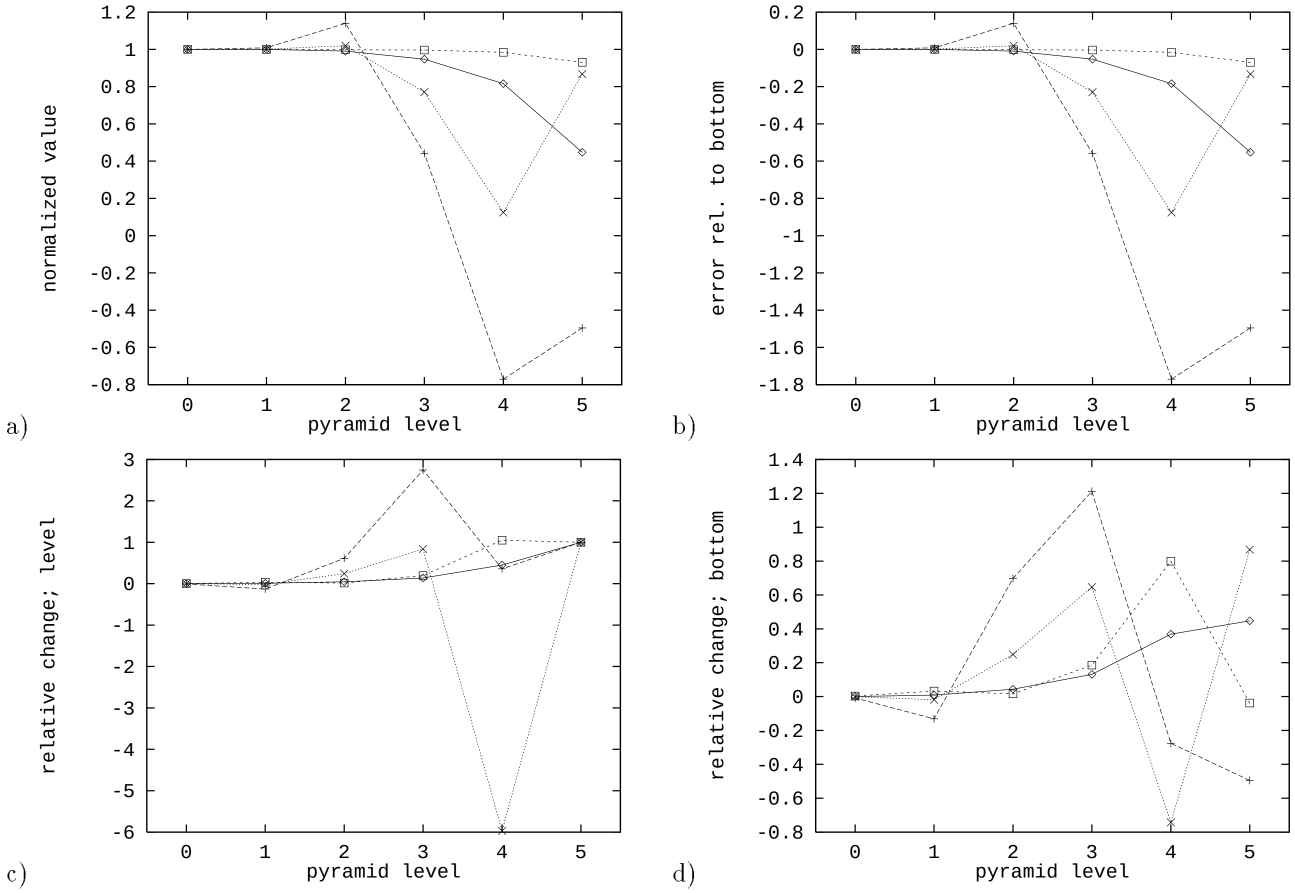

Figure 10: $a_x : \diamond \quad a_y : + \quad s : \square \quad \alpha : \times$
Different graphic representations of the estimated pose parameters given in Table 4. See also caption of Figure 7. Again, the parameters converge to the values at highest resolution (upper row) and the largest changes take place at the high pyramid levels representing low frequencies (lower row). However, this tendency is not as pronounced as in Figure 7.

Table 4: Pose parameters for the resulting images in Figure 9. See also caption of Table 2.

l	a_x	a_y	s	α	var_w	var_h
5	-36.56	-1.95	0.9972	0.0580	0.4995	0.5100
4	-66.64	-3.03	1.0549	0.0083	0.3232	0.7876
3	-77.34	1.74	1.0683	0.0516	0.6429	1.0447
2	-80.90	4.48	1.0695	0.0682	0.9457	1.0312
1	-81.62	3.96	1.0718	0.0670	1.4389	1.0743
0	-81.66	3.93	1.0721	0.0669	1.0295	0.7762

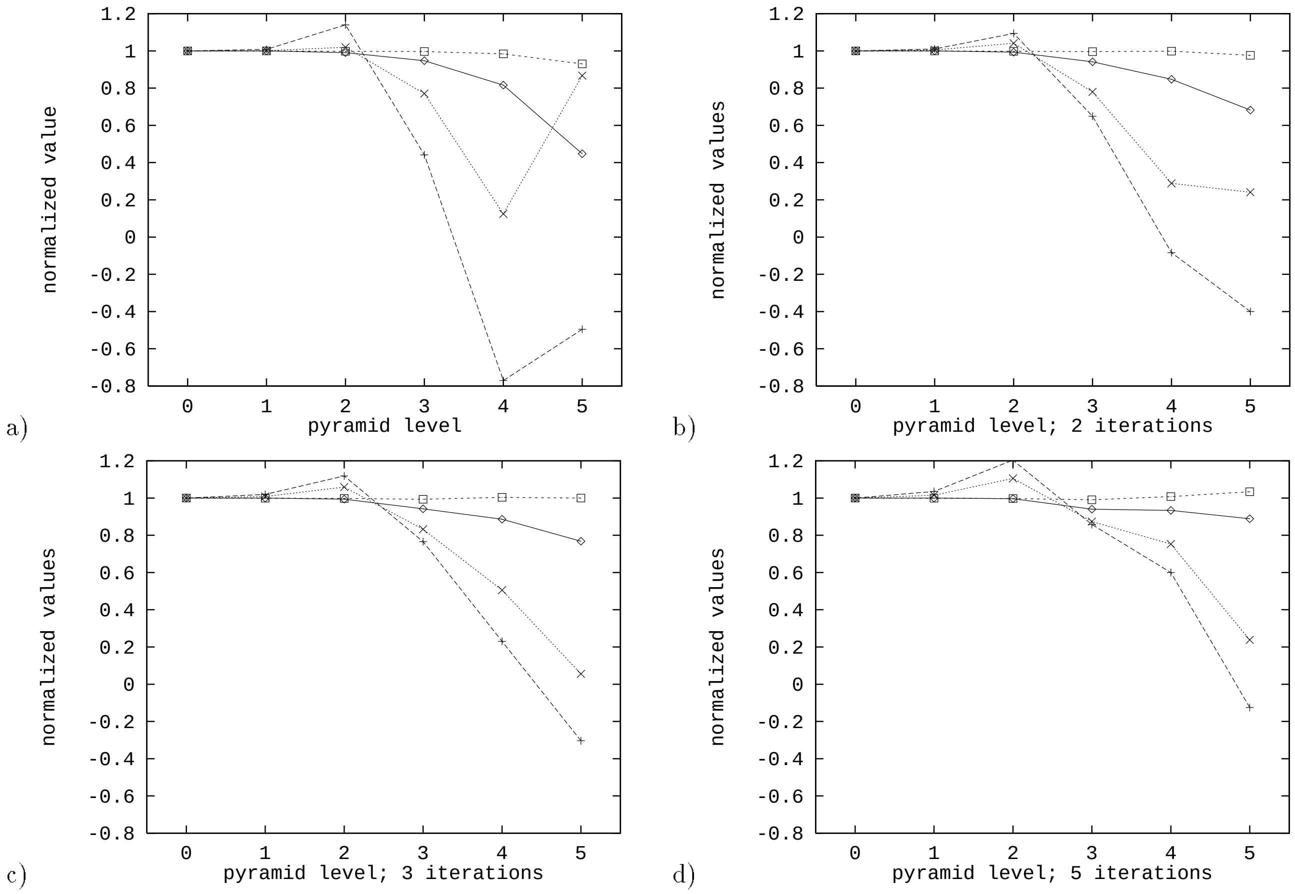

Figure 11: $a_x : \diamond \quad a_y : + \quad s : \square \quad \alpha : \times$

More detailed analysis of the parameters depending on the number of iterations (1, 2, 3, 5, respectively) per level before the estimation is continued at the next higher resolution level. The values are normalized by the bottom value. Table 5 gives the corresponding absolute parameter values. Diagram a) is identical to Figure 10 a). The major improvement is achieved by performing two iterations instead of one pass since the convergence to the bottom values becomes smoother.

Table 5: Estimated pose parameters at the bottom level ($l = 0$) as functions of the number of iterations i per level. The values correspond to the left data points in Figure 11. Note the significant change in a_y and α between $i = 1$ and 2, whereas the variation for larger number of iterations is fairly small.

i	a_x	a_y	s	α	var_w	var_h
1	-81.66	3.93	1.0721	0.0669	1.0295	0.7762
2	-87.49	12.88	1.0728	0.1135	1.0090	0.7641
3	-88.70	14.17	1.0737	0.1204	1.0163	0.7668
5	-89.00	13.82	1.0740	0.1191	1.0126	0.7652
10	-87.62	11.33	1.0716	0.1056	0.9987	0.7618

Figure 12: Illustration of the results depending on the number of iterations per level: left $i = 1$; middle $i = 3$; right $i = 10$. Careful visual inspection is required to perceive the differences although the values in Table 5 differ.

Figure 13: The image transformation can be inverted due to the parametric model used here. Given the pose parameters used to align the new image with the reference image, the inverse parameters can be computed as described in the text. Note that this is not possible for a general mapping. For illustration, the reference image in Figure 8 is transformed to the pose of the new image, i.e., the mapping is done in the reverse direction.

Figure 14: Sequence of images to exemplify the processing steps of a simple video-conference system. The left image is a newly acquired video frame. The middle image is the most similar normalized example image found in the database. The pose parameters of the new image with respect to the reference example are estimated and transmitted together with the index number for the example. On the receiver side the stored normalized example is transformed towards the pose of the new image on the sender side (right image).

Figure 15: Several images to evaluate and demonstrate the robustness of the algorithm and the range of poses it can deal with. The horizontal and vertical translation in the first two rows amount to about 110 and 50 pixels, respectively. In the third row an in-plane rotation of 15–20° to each side (measured from the vertical line) can be read from the images. Finally, the variation in size in the last row is in the range of 25%. Other examples with different backgrounds and a variety of distinct poses and facial expressions can be found in [58].

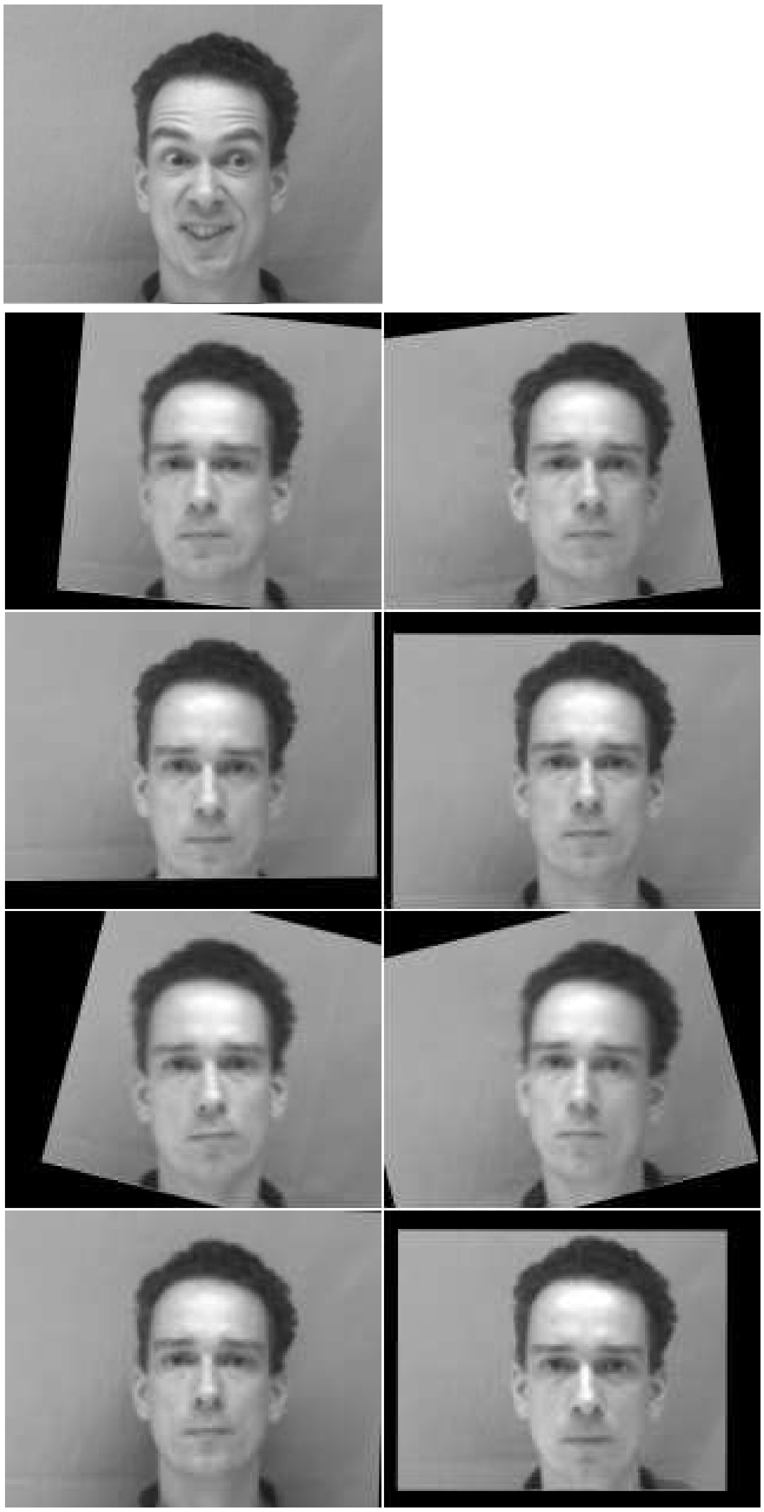

Figure 16: Pose compensated versions of the original images depicted in Figure 15 are in corresponding positions. The image in the first row is the reference image defining the intended pose. The rather distinct facial expression as compared to the new images is noteworthy. These results are obtained with only one iteration ($i = 1$) per level and the estimation is terminated at level two ($l = 2$). The reference image resembles the images in the last row of Figure 3. Conventional algorithms relying on localized feature points would very likely produce unreliable results.

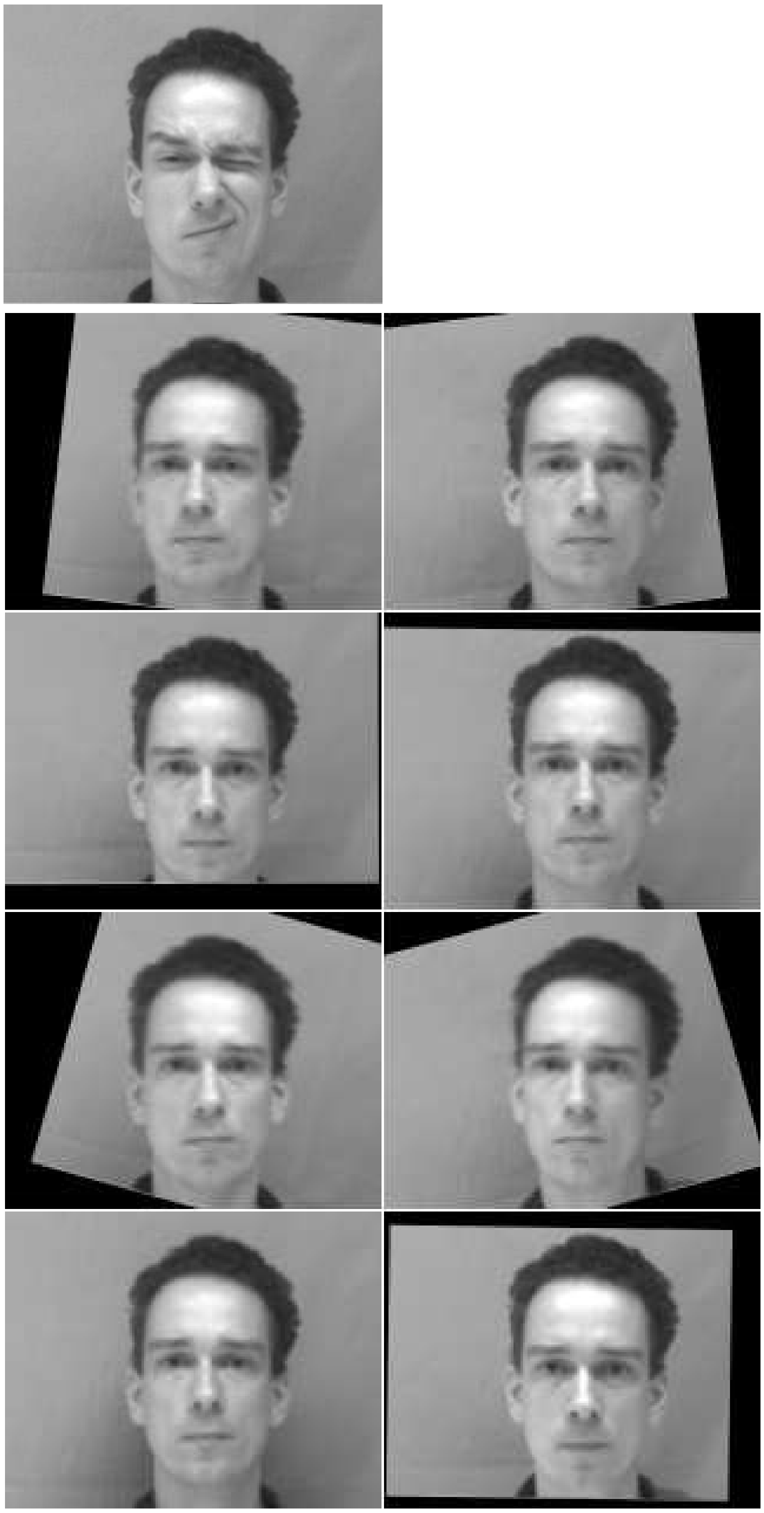

Figure 17: Results similar to Figure 16, but for a reference image with one eye closed and with severe distortions around the mouth.