A Graphical Model based Solution to the Facial Feature Point Tracking Problem

Serhan Coşar, Müjdat Çetin*

Sabancı University, Faculty of Engineering and Natural Sciences, Orhanlı - Tuzla, 34956 İstanbul, Turkey

Abstract

In this paper a facial feature point tracker that is motivated by applications such as human-computer interfaces and facial expression analysis systems is proposed. The proposed tracker is based on a graphical model framework. The facial features are tracked through video streams by incorporating statistical relations in time as well as spatial relations between feature points. By exploiting the spatial relationships between feature points, the proposed method provides robustness in real-world conditions such as arbitrary head movements and occlusions. A Gabor feature-based occlusion detector is developed and used to handle occlusions. The performance of the proposed tracker has been evaluated on real video data under various conditions including occluded facial gestures and head movements. It is also compared to two popular methods, one based on Kalman filtering exploiting temporal relations, and the other based on active appearance models (AAM). Improvements provided by the proposed approach are demonstrated through both visual displays and quantitative analysis.

Keywords: Facial feature tracking, Graphical models, Temporal and Spatial models, Occlusion detector, Human-Computer interaction, Facial expression analysis

*Corresponding author. Tel: +90 216 483-9594, Fax: +90 216 483-9550.
Email address: serhancosar@su.sabanciuniv.edu, mcetin@sabanciuniv.edu (Müjdat Çetin)

1. Introduction

1.1. Motivation & Problem Statement

During the past decades, facial expression analysis has attracted significant interest in the scientific community due to its importance for human-computer interfaces in a number of contexts including smart environments, virtual reality, video conferencing, model-based facial image coding.

A facial expression analysis system can be divided into three parts: detection, tracking, and recognition[1]. The tracking part can be defined as a bridge between the detection and recognition part, hence it is a very important part of a facial expression analysis system. However facial feature tracking is a very challenging problem because each facial expression is generated by non-rigid object deformations and these deformations are person-dependent. Other than the complex movement of these facial components, there is the movement of the head that makes the problem even more complicated. In addition to these *internal* problems, there are also *external* problems caused by events such as external occlusions, etc.

Motivated by these observations, we propose a new solution to the facial feature tracking problem described above.

1.2. Previous Work

To provide a solution to the tracking problems discussed above, a variety of methods have been proposed. Earlier pieces of work in this domain are based on low-level image processing techniques. There are methods based on template matching which use the intensity pattern as the template and search for a region that has the closest pattern [1, 2, 3, 4, 5]. There are many methods in which

[1] A facial system that operates on static, single-image frames would not have the tracking part. However, in order to fully exploit the temporal structure of the data for better performance, tracking is required.

Gabor features are used for tracking [6, 7, 8, 9, 10] in which the disparity of a point from one frame to the next is estimated in terms of phase differences of Gabor jets. In [6, 7], the motion of facial features are modeled statistically, and a Kalman filter exploiting such temporal information is used for tracking. Active contours or curve evolution methods have been used to track the shape of facial features. In such methods, intensity values [11, 12, 13, 14] or Gabor features [15] are usually used as part of the data term in curve evolution. Similarly, in [16, 17, 18] 3-D wire-frame models are constructed and iteratively deformed using intensity values. Spatial relations are taken into account by constraining the shape of facial features using subspace representations, such as principal component analysis (PCA), in the shape space, leading to so-called active shape models (ASM) [11, 12, 13]. This idea has been combined with subspace representations of appearance, resulting in active appearance models (AAM) [11, 15, 18]. There is not much work that models spatial and temporal relations together. There is one paper [19] which uses a directed graphical model for tracking of contours modeling facial feature motion. The graphical model is based on non-Gaussian distributions.

Recent methods discussed up to this point have varying levels of success in handling data quality limitations and real-world conditions such as occlusion and head movement. Earlier works become fragile when the data quality is low because of noise, etc. For such cases, temporal relations provide very useful information. In most of the methods it is assumed that the images are frontal face images, and occlusion or head movement are not directly addressed. Some methods have a solution to head movement but in a limited range of head pose variations [5, 6, 7, 8, 9, 10]. To handle real-world conditions, spatial relations are utilized using training-based shape constraints. Although they are not adequate for occlusion, these methods are more robust to head pose variations but within the limits of the training sets. The more the number of training images with pose variations, the more robust the technique becomes under pose variations. Of course, a training phase is an additional step that causes extra computational

load to the system and this is a disadvantage. On the other hand, in systems which do not incorporate spatial relations, as the methods that use only temporal relations, this lack of relations also causes drifts and poor performance. For this reason these relations should be considered as well. In this manner, there are some methods in which ASM is extended using temporal relations [20].

1.3. Contributions of this Paper

In this paper facial feature point tracking is performed in a graphical model-based framework that incorporates both temporal and spatial information about the feature points. There has been some methods that incorporate temporal and spatial relations [20, 19]. Unlike these methods, we propose an approach based on graphical models that can effectively handle both temporal and spatial information within a single, unified framework. This framework is the main contribution of this paper. The temporal relation in the model is similar to Kalman filtering type models. A point-to-point interaction model is constructed for spatial relationships. The overall model is formed as a Markov random field (MRF). It is based on a parametric statistical model in which the probability densities involved are Gaussian and belief propagation algorithm is used for inference. The parametric nature of the models makes the method computationally efficient. Constructing both the temporal and spatial relations allows us to have the advantages of both approaches in a coherent framework. The spatial connections between points allow the tracking to continue reasonably well by exploiting the information from neighboring points, even if a point disappears from the scene or cannot be observed, e.g. due to occlusions. Another advantage of the spatial connections is to build a whole model by binding the feature points and prevent possible drifts occurring because of head movements. The feature values from video sequences are based on Gabor filters that are sensitive to different poses, orientations and different feature sizes. Also, an occlusion detector based on the Gabor filter outputs is proposed to automatically detect occluded points. The proposed tracker has been applied on real video data under various conditions including occluded facial gestures and 3-D head movements, and performance of

the proposed approach is demonstrated through both visual displays and quantitative analysis.

The rest of this paper is organized as follows. Section 2 briefly introduces graphical models and describes the components of the proposed framework. In Section 3, we explain our algorithmic solution of the posed tracking problem . Then, Section 4 focuses on the treatment of real-world conditions. In Section 5, experimental results are presented demonstrating the performance of the proposed approach as well the improvements it provides over two existing methods. Conclusions are drawn in Section 6.

2. Graphical Models for Facial Feature Tracking

2.1. Overview

Graphical models can be defined as a combination of graph theory and probability theory. They provide a tool for approaching two major problems of engineering: uncertainty and complexity. Efficient probabilistic inference algorithms defined on graphs make this structure computationally attractive.

Generally a graph G is defined by a set of nodes V, and a corresponding set of edges E. The neighborhood of a node $s \in V$ is defined as $N(s) = \{t|(s,t) \in E\}$. The models are divided into two main categories: directed and undirected graphs. Directed graphs are graphs in which there is a causal relation between random variables. In undirected graphs the relation is bidirectional. A sample undirected graph, or MRF, is shown in Fig. 1. The graphical models of interest in this paper are undirected.

We associate each node $s \in V$ with an unobserved, hidden random variable x_s which can be drawn from a wide range of probability distributions, and a noisy local observation y_s. Sometimes the observations are represented as separate nodes connected to the nodes corresponding to the hidden variables. Let

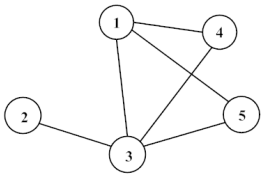

Figure 1: An example for an undirected graph.

$x = \{x_s | s \in V\}$ and $y = \{y_s | s \in V\}$ denote the sets of all hidden and observed variables, respectively. Considering pairwise MRFs, a graphical model encodes the factorization of a joint probability density function $p(x, y)$ as shown below.

$$p(x, y) = \frac{1}{Z} \prod_{(s,t) \in E} \psi_{s,t}(x_s, x_t) \prod_{(s) \in V} \psi_s(x_s, y_s) \tag{1}$$

Here, $\psi_{s,t}(x_s, x_t)$ is the edge compatibility function between hidden variables, and $\psi_s(x_s, y_s)$ is the node compatibility function.

The graphical model that is used in the proposed method consists of three sub-models: a temporal model, an observation model, and a spatial model. A representative version of the model that can be used only for two facial features is illustrated in Fig. 2 (An example set of facial feature points is shown in Fig. 3). Considering real-world coordinates, the hidden variables should ideally be 3-D points that move in the real-world coordinate system. However, because of the ill-posed structure of the problem (2-D observations, 3-D hidden variables) the tracking problem becomes quite challenging. To simplify this, the feature points are taken as 2-D points that move on the camera plane (see Section 4.2). The hidden variables x_s are vectors, each with four elements: x-coordinates, y-coordinates, velocity of the point in the x direction, and velocity of the point in the y direction. The observed variables y_s are vectors, each with two elements; x-coordinates and y-coordinates of observed data. So in this notation, $x_i(t)$ means hidden variable of the ith feature point at discrete time point t and $y_j(t)$ is the observed variable of the jth feature point at discrete time point t.

The relations between nodes are represented by edge compatibility functions, as in (1). In this paper, these functions are selected as Gaussian distributions. In the next subsections, construction of edge and node compatibility functions in the sub-models will be explained in detail.

2.2. Temporal Model

The temporal model captures the temporal behavior of each facial feature point. In this paper, these relations are modeled using linear dynamics. Since this modeling is the same for each feature, for simplicity, we suppress the subscript notation in the variable $x_i(t)$ and define $x(t)$ as the state variable of a generic feature point.

$$x(t+1) = A \cdot x(t) + w(t) \qquad w(t) \backsim N(0, Q) \tag{2}$$

Here, A is the state transition matrix and $w(t)$ is the process noise, which is normally distributed with mean zero and covariance matrix Q, and is independent of $x(t)$. It is assumed that the points move with a constant velocity. Therefore the state transition matrix is selected as:

$$A = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3}$$

The temporal connection between two nodes involves a Gaussian distribution and the edge compatibility function can be defined as follows:

$$\begin{aligned} \psi_{t+1,t}(x(t+1), x(t)) =& p(x(t+1)|x(t)) = N(A \cdot x(t), Q) \\ =& \alpha \exp\left\{(x(t+1) - A \cdot x(t))^T Q^{-1} (x(t+1) - A \cdot x(t))\right\} \end{aligned} \tag{4}$$

2.3. Spatial Model

The spatial model contains one of the main contributions of this paper. This part models the spatial relations between facial features. Here a model is introduced which brings in prior information about the expected spatial distances

between feature points. The spatial model between ith and jth feature points is represented as a Gaussian distribution as follows. For simplicity of notation, we consider the spatial relation at a single time point and neglect the time arguments in the variables $x_i(t)$, $x_j(t)$.

$$\psi_{i,j}(x_i, x_j) = \alpha \exp\{\left[\ x_i - (x_j - \mu_{\triangle x})\ \right]^T \Sigma^{-1} \left[\ x_i - (x_j - \mu_{\triangle x})\ \right]\} \qquad (5)$$

where $\mu_{\triangle x}$ represents the mean of the random vector $\triangle x = (x_j - x_i)$ and Σ represents its covariance matrix. Since the hidden variables, x_i, x_j, are four-element vectors, the random vector, $\triangle x$, is also a four-element vector $\triangle x = [\ \triangle x_x \quad \triangle x_y \quad \triangle x_u \quad \triangle x_v\ ]^T$ where $\triangle x_x$ and $\triangle x_y$ represent the spatial differences between facial features on the x-axis and y-axis respectively; $\triangle x_u$ and $\triangle x_v$ represent the differences between velocity components in the x and y directions, respectively. As mentioned in Section 2.1, the proposed method tracks the locations of the projection of the feature points on the camera plane. Hence, the spatial differences are simply the 2-D spatial distances on the camera plane.

When it is not possible to observe certain data because of occlusion, noise etc., incorporating spatial constraints becomes very important. For instance; when an eye corner point cannot be observed properly because of occlusion, it is hard to continue tracking with uncertain data. But, since this point is spatially connected with the other eye corner point (see Fig. 2), the spatial information from neighboring points can prevent drifts and make accurate tracking possible.

2.4. Observation Model

This subsection is about the relations between the hidden variables and the corresponding noisy observations. The relation is modeled in a linear form as follows. Again, for simplicity of notation, we neglect the subscript notation and define $x(t)$ and $y(t)$ as the state variable and observation corresponding to a generic feature point.

$$y(t) = C \cdot x(t) + v(t) \qquad v(t) \backsim N(0, R) \qquad (6)$$

Here, C is the observation matrix and $v(t)$ is the observation noise, which is normally distributed with mean zero and covariance matrix R, and is independent of $x(t)$. The observation matrix is selected as follows:

$$C = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \tag{7}$$

As a result, we have

$$\psi_s(x(t), y(t)) = p(y(t)|x(t)) = N(C \cdot x(t), R) \tag{8}$$

Extraction of $y(t)$ from the video data is explained in detail in the next section.

3. Algorithmic Solution

3.1. Data Preprocessing

To extract the 2-D feature point observations from images, some data preprocessing is performed. For any given image from a video sequence, the feature point of interest is searched in a search region by comparing groups of pixels with a template patch that is selected as a reference for the corresponding feature. This comparison is based on the Gabor filter outputs of both the template image and the given image region. Gabor wavelets are a time-frequency representation tool used in many computer vision problems. They achieve robust performance for feature extraction under illumination and appearance variations.

In this paper, the Gabor filters are selected as in [21]. Then as in [6], image regions are convolved with 24 filters consisting of 6 different orientations and 4 different wavelengths. The magnitude and phase of the complex outputs of the filtering at different spatial locations are compared using the similarity metric in [21]. This produces similarity values for every point in the convolution region. The location of the best match, with the highest similarity value, is used as the observation data for the corresponding feature point. The overall flow of the preprocessing is illustrated in Fig. 4. We select the search region around the estimated location of feature point from the previous frame. Template patches

could be based on marked first frame or the previous frame. In [6], the feature point of interest is searched by comparing with the template patch that is obtained from the previous frame. But our experiments show that this kind of consecutive comparison causes drifts and error accumulation. Thus in our work, we generate the template patches from the first frame of the sequence. Extreme changes in the appearance of facial features, for instance because of facial gestures, may result in the failure of the feature extraction process. One way to address this issue might be to use a bank of template patches that represent the appearances for different facial gestures. But, in practice, because of the variability of facial gestures, constructing such a database is a hard problem. Rather than using such a complex feature extraction process, we use one template patch for each facial feature (generated from the first frame of the sequence) and benefit from the power of our statistical model when feature extraction fails.

In this work we focus on facial features located in parts of the face with an inhomogeneous texture pattern. This is motivated by the observation that in the context of facial expression analysis, feature points in inhomogeneous regions such as eye corners and lips have proven to be more important than points in regions with homogeneous texture. For such points, Gabor filters provide an appropriate mechanism for feature extraction. If one is interested in other features, such as those in more homogeneous parts of the face including the cheeks and the forehead, then one would need to use a feature extractor appropriate for tho se features.

The output value of the similarity metric also provides quantitative information about how similar the best match in the search region and the initial feature point are. This information can be used to detect frames in which there is an occlusion in the region of interest of a feature point. The occlusion detection part, another contribution of this paper, is explained in Section 4.1 in detail.

3.2. Loopy Belief Propagation Algorithm

In many computer vision applications, the main goal is to find the marginal conditional density function $p(x_s|y)\ \forall s$. In our work we want to find the marginal conditional density function of each feature point at each time index (Fig. 2). Up to this point we used the notation $x_i(t)$ for the ith feature point at discrete time point t. Here, we combine these indices and use the notation x_s, where s represents both feature point and temporal indices.

For graphs which are acyclic or tree–structured, the desired conditional distributions can be directly calculated by a local message–passing algorithm known as *belief propagation* (BP) [22]. In chain-structured graphs, this algorithm is equal to Kalman filtering. For cyclic graphs, Weiss [23] as well as others showed that BP produces excellent empirical results in many cases. BP is typically described as a parallel message-passing algorithm which is iteratively applied. In particular, in the nth iteration, each node $t \in V$ calculates a message $m^n_{t,s}(x_s)$ to be sent to each neighboring node $s \in N(t)$:

$$m^n_{t,s}(x_s) = \alpha \int_{x_t} \psi_{s,t}(x_s, x_t)\psi_t(x_t, y_t) \times \prod_{u \in N(t)\setminus s} m^{n-1}_{u,t}(x_t)dx_t \tag{9}$$

Each node combines these messages with its own observation and produces its own conditional density function:

$$p(x_s|y) = \alpha\psi_s(x_s, y_s) \prod_{t \in N(s)} m^n_{t,s}(x_s) \tag{10}$$

In our graphical model, the algorithm is applied and the marginal conditional density function $p(x_s|y)$ is updated at each time step for each feature point. These message-passing operations can be analytically computed only in special cases, such as that of Gaussian or discrete random variables. In our model, we use Gaussian densities, which appear to provide accurate statistical descriptions for our problem. In this case, the two steps of the algorithm stated above simplify to updating means and covariances. If non-Gaussian densities were used, one would have to resort to approximate, possibly non-parametric iterative methods, which would be slower than our approach. To make the proposed

tracker usable in real-time applications, each update step is done using the current and the previous data (and not the future data). As a result the algorithm used becomes a filtering (rather than smoothing) algorithm. For the details of the update equations for the Gaussian case please see [24].

4. Tracking under Real-World Conditions

As a natural human action people move their head, hands all the time. Thus, in real-world conditions there can be external occlusion problems (because of moving hands etc. in front of the face) and/or problems due to 3-D head movement. To build a facial feature tracker that is robust to real-world conditions, these cases should be handled properly. The proposed method is designed in a way to deal with these problems. This is one of the contributions of this paper. The proposed solutions for these problems are explained in the following subsections.

4.1. External Occlusions

Considering real-world scenarios of computer vision applications, the occlusion problem can be divided into two categories: external occlusion and self-occlusion. Self-occlusion can be defined as the case when the object of interest occludes (parts of) itself. Most of the time this occurs because of the movements (e.g. rotation) of the object. Contrarily, external occlusion is the occlusion that is caused by other objects occluding the object of interest. In the context of this paper, self-occlusion can be defined as the occlusion when there is out-of-plane motion of the head. This case is handled in the next subsection. In this subsection the external occlusion problem is considered.

When occlusion occurs, the observed data become useless for the occluded feature point and should be disregarded. Therefore, we first need to detect the occlusion. We have observed that when occlusion occurs, the similarity values in our Gabor filter-based search begin to drop (illustrated in Fig. 5). By thresholding similarity values, this quantitative information can be used to detect the

occlusions of various features in the video frames. When an occlusion is detected for a feature point at a particular time instant, the observed data for the feature point is disregarded. Although one might be concerned that this lack of data will have a dramatic effect on tracking, the proposed tracker can continue tracking using the information contributed by the temporal dynamics and the spatial constraints without incorporating any observation at that time instant.

4.2. 3-D Head Movements

To deal with the 3-D movements of the head, ideally (and unlike the development up to this point) a framework in which the feature points are represented as 3-D points would be needed. In a real-world coordinate system, facial feature points are 3-D points, but the observations are 2-D videos taken from a monocular camera. This produces the need for a mechanism to associate 2-D observations to 3-D real-world points.

Going from 2-D to 3-D is a hard problem which does not have an exact solution because of its ill-posed structure. In the literature, there are two main approaches to this problem. One approach is stereo vision. In stereo vision applications, one needs to take advantage of stereo cameras. Since in this paper a monocular camera is used, this approach is out of the scope. The other approach is to use a monocular camera and try to extract 3-D information. There are a number of proposed methods in the literature to obtain 3-D information from motion [25, 26, 27, 28]. These methods are mostly based on point-correspondence and assume that there is one moving object in the scene. This assumption is clearly not satisfied in our problem, as we are interested in tracking *multiple* facial features on a head, which itself can be moving as well. Also, the point-correspondence can be very fragile in the case of out-of-plane head movements. Considering these potential issues, it has been decided not to use this approach. In [29, 30, 31, 32], the 3-D model of the object of interest is used as prior knowledge to extract 3-D information. However, given that face

has a non-rigid, person-dependent structure, constructing a generic, accurate 3D model of the face is not a trivial task. Because of the complexities of these methods, it has also been decided not to pursue this path. Instead, we develop a simpler solution matched to the structure of the facial feature tracking problem.

To suppress this deficiency of the 2-D data for 3-D movements, incorporating side information about the head pose can be a solution. If this information is obtained somehow and incorporated into the model, then our model that uses 2-D states can be adapted to the pose of the head and our method can continue tracking in the case of 3-D head movements. We can take advantage of what we will call *reliably tracked points* to extract partial information about the head pose [2]. We define *reliably tracked points* as the feature points that do not move significantly due to facial expressions. Namely, their motion is almost due entirely to head pose change. We assume that these points are not occluded, hence we can track them reliably.

The motion of inner eye corners is mostly because of head pose change. Since these points are in the middle of the face, the self-occlusion problem for these points can be negligible, provided that the head does not go through extreme 3-D rotations. Thus, inner eye corner points can be selected as *reliably tracked points* and they can be used to get the head pose information.

Consider a moving plane in space. Let there be n 3-D points on the plane each represented by $P_i = [X_i, Y_i, Z_i]^T,\ i = 1, ..., n$, and let $p_i = [x_i, y_i]^T$ be the projection of each point onto the camera plane. In the perspective camera

[2] If an external head pose tracker is available, our algorithm could exploit the accurate head pose information provided by such a tracker. In the context of this paper, we assume that such a tracker is not available, and demonstrate how our algorithm can incorporate partial information provided through the use of reliably tracked points.

model, 3-D points and their projections are related by

$$x_i = f\frac{X_i}{Z_i} \qquad\qquad y_i = f\frac{Y_i}{Z_i} \tag{11}$$

where f denotes the focal length. Since the points are on the plane, the 2-D motion of each point p_i on the camera plane can be represented by the well known "optical flow" equation [33] as

$$\begin{pmatrix} \dot{x}_i \\ \dot{y}_i \end{pmatrix} = \begin{pmatrix} a \\ b \end{pmatrix} + \begin{pmatrix} c & d \\ e & f \end{pmatrix} \begin{pmatrix} x_i \\ y_i \end{pmatrix} + \begin{pmatrix} gx_i^2 + hx_iy_i \\ hy_i^2 + gx_iy_i \end{pmatrix} \tag{12}$$

For simplicity, the last term of (12) can be neglected and the motion of each point p_i can be represented as an affine motion. By this assumption, affine camera model (i.e. scaled orthography) and affine invariances can be used. Under affine transformation, the ratio of distances between collinear points is preserved [34]. Consider Fig. 6. Let line l go through an affine transformation $\mathcal{A}$ and become line l'. Since the transformation is affine, the ratio between distances is preserved:

$$\frac{|PQ|}{|QR|} = \frac{|P'Q'|}{|Q'R'|} \tag{13}$$

Assuming the face is a planar surface and facial feature points are moving on this plane, this invariance can be used to adapt to distances between feature points to the new head pose. Let points P,Q,R represent the locations of three feature points at the reference head pose, with spatial connections between Q and R. Then assume that, P',Q',R' are the new point locations after the head pose changed. If P',Q' points are assumed to be reliably tracked points then the expected spatial distance between Q' and R' can be found based on the affine ratio relationship in (13), and the parameters of the spatial model in (5) can be updated accordingly.A more general approach can be to assume feature points are nonplanar and to use a perspective camera model. In this way, projective invariance (i.e. cross-ratio) can be used to update the coordinates of the mean vector [33]. But to update the coordinates using cross-ratio, there is a need of three reliable points that are on the same line (see Chapter 10 of [33]). Since it is difficult to find three reliable points on the face that are on the same line and since feature points are almost on a plane, using affine ratio is expected to be

sufficient and appropriate for the particular problem of interest in this paper. Therefore, affine-ratio is used in our work.

In the case of head rotations around x-axis and y-axis the inner eye corner points can be occluded by other feature points because of the head movement. In this case our tracker would work accurately up to the point where what we use as reliable points remain unoccluded.

5. Experimental Results

5.1. Setup & Parameter Selection

The data used in our experiments are grayscale videos with 640x480 resolution. Videos are recorded with a rate of 30 frames per second without compression. Assuming the point coordinates and velocities are independent of each other, all covariance matrices in the framework are constructed in diagonal form. The covariance matrices Q and R in (4) and (8), respectively, are selected as follows:

$$Q = \begin{bmatrix} 16 & 0 & 0 & 0 \\ 0 & 16 & 0 & 0 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 4 \end{bmatrix} \qquad R = \begin{bmatrix} 4 & 0 \\ 0 & 4 \end{bmatrix} \tag{14}$$

The Gaussian densities involved in the spatial model are formed in $information\ form$. It is assumed that the variance of the distances on x and y axes are 4 pixels. The parameters of the covariance matrix are set according to the spatial constraints. For the case in which there are constraints only on x and y axis positions, the inverse covariance matrix (Σ^{-1}) in (5) for the corresponding spatial model is selected as follows:

$$\Sigma^{-1} = \begin{bmatrix} 1/4 & 0 & 0 & 0 \\ 0 & 1/4 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{15}$$

By setting the last two diagonal elements of the inverse covariance matrix zero, we do not impose any prior information on relative velocities of feature points.

The x and y coordinates of the mean vector ($\mu_{\triangle x}$) in the spatial model are selected using the 2-D distance between the feature points in the first frame of the sequence.

For the initial covariances of each hidden variable, we choose large values reflecting the assumption that the initial uncertainty is large, and letting the algorithm refine these values based on observed data. In particular, we initialize the prior covariance for each feature point as follows:

$$\Sigma_{x_0} = \begin{bmatrix} 100 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 \\ 0 & 0 & 25 & 0 \\ 0 & 0 & 0 & 25 \end{bmatrix} \tag{16}$$

An ideal way of determining the parameters above would be to learn them from the experimental data in each scenario. Here, we illustrate the use of learning by providing a simple example. Of course, in a learning task we would need sufficient amount of labeled training data providing the ground truth for feature points. A manual feature point marking procedure is performed to compose this ground truth data. Consider determining the parameters of the spatial model describing the relationship between the inner and outer corner points of the right eye. In Fig. 7, the histogram of 2-D spatial distances (in pixels) on the x-axis for this particular scenario is shown. The histogram is constructed by using 1109 sample images. The mean and variance of this data are given in Table 1. This information can be used to set the parameters of the Gaussian density in the spatial model. However, the parameter selection from learning will produce the need of ground truth data of each feature point for every scenario of interest. For the illustrative set of experiments in this paper, we do not carry out such a detailed learning process for all the feature points, but rather we select the covariance matrices as in (14), (15) and specify the x and y coordinates of the mean vector simply using the manually-marked first frame in each sequence, as described in the beginning of this section. In practical use of our approach in a real tracking scenario, one may of course not have access to the marking of the initial frame for the test sequence, and would use parameters learned from

offline training data as described above, in the latter portion of this section.

	Mean	Var
The spatial distance on x-axis	55.6177	4.119

Table 1: The mean and variance of the distribution in Fig. 7

5.2. Basic Experiments

Based on the main motivations of this paper, the proposed method is tested in scenarios involving facial expressions. A number of distinct facial gestures are selected for the experiments. These include mouth opening, eye closure, smiling, and eye (wide) opening. The sequence consisting of such gestures is recorded continuously, and also includes natural eye blinking movements. 16 feature points are selected including 2 points for each eye brow, 4 points for each eye and 4 points for the mouth. In Fig. 8, the selected feature points and the spatial connections, we have decided to exploit for such an experiment are illustrated. In Table 2, the spatial constraints of the corresponding connections are shown.

The results for a sequence in which there are facial expressions described above and there is no head movement are shown in Fig. 9-b. To make a comparison, an algorithm that uses only temporal relations and that is based on Kalman filtering [6] is also applied to the same sequence and the results are shown in Fig. 9-a. It can be clearly seen that the proposed method gives more accurate results, while drifts occur for the other algorithm. Most of the drifts occur in the eye region points because eye blinking and eye closure cause the observed region to change rapidly. Since the other algorithm does not use spatial relations, these rapid changes result in loss of tracking performance. To perform a quantitative evaluation, the Euclidean errors of the trackers for estimating the position of each feature point are evaluated by using the ground truth data. Ground truth data were labelled by one person. But, for accuracy, the markings were checked and corrected by another person. The quantitative errors (in pixel) of the tracker

Connected Couples	$\triangle x$			
	$\triangle x_x$	$\triangle x_y$	$\triangle x_u$	$\triangle x_v$
1-2	•	•	-	-
4-3	•	•	-	-
5-8	•	-	-	-
7-8	•	-	-	-
11-8	•	-	-	-
6-9	•	-	-	-
8-9	-	•	-	-
10-9	•	-	-	-
12-9	•	-	-	-
13-15	•	-	-	-
14-15	-	•	-	-
16-15	•	-	-	-

Table 2: The selected spatial constraints of the corresponding spatial connections shown in Fig. 8. Dots indicate the constraint used.

in [6] and the proposed tracker are shown in Table 3. Due to the imperfection of ground truth, because of manual marking, errors of 2-3 pixels can be negligible. The quantitative results confirm our observations above. The overall errors show that the results of the proposed method are more accurate. The movement of eye lid features provides information about the status of the eye (blinking, closure,... etc). For this reason, a tracking algorithm in a facial expression recognition system should robustly track eye lid features. The method in [6] makes very large errors for the eye lid feature points whereas the errors of the proposed method are negligibly low.

5.3. Experiments Involving External Occlusion

The purpose of this experiment is to test the performance of our facial feature tracker under occlusion and the performance of our occlusion detector, described in Section 4.1, in facial expression sequences. Here, a sequence that consists of the occlusion of a mouth feature point and an eye corner point, as well as facial gestures such as mouth opening and wide opening eyes is used.

Errors	Method in [6]		Proposed method	
Feature Point No.	Mean	Var	Mean	Var
1	2.9515	1.6815	3.2449	2.1996
2	5.2828	4.4396	5.3056	3.7691
3	2.7997	1.6815	3.0503	1.9411
4	3.5312	1.8122	2.895	1.6714
5	21.8701	202.6687	3.3094	4.2972
6	7.0232	74.1023	6.6992	68.102
7	2.6495	2.8636	6.605	6.1585
8	2.6055	3.9925	2.7635	3.789
9	2.823	3.05	3.3881	5.7874
10	2.736	3.4941	3.747	3.3388
11	43.8116	552.5629	4.7928	32.6585
12	16.8808	187.4258	4.1475	22.1235
13	1.9145	1.0547	3.1186	2.1213
14	4.2243	4.4301	4.6351	4.5915
15	5.2805	5.5958	5.3156	13.8288
16	6.0641	22.4314	5.1734	19.5062
Overall	8.278	173.3266	4.2619	22.5933

Table 3: Quantitative errors of the tracker in [6] and the proposed tracker in the sequence shown in Fig. 9.

There are also natural eye blinking movements in the sequence. The proposed tracker's performance in such a sequence is illustrated in Fig. 10-b. In Fig. 10-a, the results of an existing technique [6] is shown. To make a proper comparison, the same occlusion detector in our approach is also used for the technique in [6].

It can be observed that in the case of data loss because of occlusion, spatial constraints in our approach enable the tracking to continue. Without the spatial relations, the positions of the points cannot be accurately estimated, as demonstrated by the results of [6] in Fig. 10-a. When the occlusion occurs, the information coming from the data term of the occluded feature is neglected in

both tests. The proposed method continues to track using the spatial information and temporal information, but the other algorithm fails to track because only temporal information is insufficient for accurate tracking. Using only temporal relationships will cause the feature points to go on with constant velocity (due to the nature of the linear, constant velocity dynamical model used). The drifts are because of this lack of information. These drifts can also be observed in the quantitative results for the corresponding feature points in Table 4. Here, the ground truth data for occluded feature points are obtained by marking the likely positions of the points as estimated by the person performing the labeling. The errors for occluded points are also included in the quantitative results. The overall errors in Table 4 show that the proposed method is more capable of tracking in scenarios including external occlusion. One may obtain a rough assessment of how occlusion affects the performance of our tracker by comparing the results in Table 3 and 4. As explained in Section 4.1, the best match similarity outputs for the corresponding feature points provide quantitative information about occlusion. These similarity values for four eye corner points are plotted in Fig. 11. It can be clearly seen that thresholding the similarity values below 0.95 (red line) can detect the occlusion.

Up to this point the proposed method has been compared with the method in [6], which involves no spatial constraints. To compare the performance of the proposed tracker this time with a technique involving spatial constraints, Active Appearance Models [11], a widely known technique used for facial expression analysis, is also applied to the same sequence. The results of the AAM method are shown in Fig. 10-c. The AAM-API library [35] is used for implementation of the AAM method. A training set that consists of the same facial gestures described above is composed from the Cohn-Kanade database [36] and AAM is trained with these images. The results show that AAM cannot be sufficient for tracking facial features under external occlusion. The main reason of this is that the AAM method demonstrated here does not use the occlusion information and it is dependent on the training set. Since external occlusion is a case

Errors	Method in [6]		Proposed method	
Feature Points No.	Mean	Var	Mean	Var
1*	74.9623	899.8539	13.5749	38.8698
2	8.5802	22.9464	8.5421	23.485
3	10.0582	26.2468	9.3382	22.7377
4	10.2098	13.2184	10.5673	14.6818
5	8.4403	14.7268	8.247	15.274
6	7.7024	14.3648	6.5375	12.4466
7*	33.0846	156.0525	6.4674	12.8102
8	5.0784	8.7588	5.1565	7.9788
9	5.7576	10.2903	5.8035	9.764
10	4.9655	8.3805	4.5125	7.7788
11	4.989	10.4318	5.2705	10.3697
12	6.3485	12.1798	6.3323	12.1276
13	6.7574	20.5591	6.668	20.4128
14*	23.6016	656.7814	7.9048	19.8647
15	6.2903	17.2305	5.9772	15.3571
16	10.9803	64.1293	10.9873	64.254
Overall	14.2379	326.6216	7.6179	65.8467

Table 4: Quantitative errors of the tracker in [6] and the proposed tracker in the sequence shown in Fig. 10. The points marked with "*" correspond to the points that are occluded in a particular time of this sequence.

that cannot be fully represented by sample images, including some images in which the facial components are occluded in the training set cannot be sufficient for general occlusion scenarios. However, there are methods in which AAM is used together with occlusion information by robust error functions [37] or detecting outliers [38]. Also, there is some drift on the unoccluded left eye brow which is because of the limits of the training set. As a result it can be concluded that the performance of the proposed method is better than the performance of the methods in [11] and [6] under external occlusion.

5.4. Experiments Involving Head Movement

Head movement is an important issue in facial feature tracking problems. In this section, we present results on the application of the proposed method on sequences that include rotation around x-axis (θ_x), rotation around y-axis (θ_y) and translation on z-axis (T_z), which are types of "out-of-plane" motion, as well as mouth opening gesture. In these types of head movements, it is clear that the appearance and edge structure of some feature points will undergo a major change. Since the template patches are generated from the first frame of the sequence (Section 3.1), it may be difficult to extract correct observations. However, with the use of our occlusion detector, these incorrect observations can be omitted and our statistical model can continue tracking by incorporating both spatial and temporal information.

Here, the left and right inner eye corner points are selected as *reliably tracked points* and they are used to obtain some information regarding the head pose (by affine invariance). The head pose in the first frame of the sequence is selected as the reference head pose. The results of our approach for the "out-of-plane" head motion sequences are shown in Fig. 12-b, Fig. 13-b and Fig. 14-b. We also present results of the method in [6] and the AAM method in [11] on the same sequences. These results are given in Fig. 12-a, Fig. 13-a, Fig. 14-a and in Fig. 12-c, Fig. 13-c, Fig. 14-c respectively. The quantitative results for these experiments are given in Table 5, Table 6 and Table 7. First, we note that our method outperforms the method in [6] in the overall errors. A detailed analysis is given below.

While the overall performance of our method is better than the other methods in these experiments, we also observe that that our method has certain limitations in getting the 3-D head pose information based on reliably tracked points (left and right eye inner corners). In particular, for sequences that contain θ_x and θ_y motion (Fig. 12-b and 13-b) drifts occur, especially for eye feature points. As explained in Section 4.2 when the feature points that are taken as

Errors	Method in [6]		Proposed method	
Feature Points No.	Mean	Var	Mean	Var
1	7.7547	6.0941	7.9711	8.0411
2	6.2688	7.7507	6.0041	6.7594
3	3.8367	4.4322	3.6256	4.6126
4	3.4234	2.4056	3.1331	2.42
5	10.0875	41.3224	9.1781	34.0289
6	10.7619	55.7967	8.9935	38.574
7	4.5767	13.1218	6.6986	20.99
8	4.8406	16.6608	3.9675	5.3031
9	5.2186	29.9649	6.2543	7.704
10	13.0865	108.2725	13.5364	55.4233
11	30.2639	614.7551	5.4531	10.9155
12	6.818	28.3884	6.0795	29.5396
13	2.263	1.8954	2.6063	2.0324
14	3.6134	3.9564	6.8595	14.5613
15	16.4026	159.9598	13.8698	105.1156
16	4.81	11.4739	4.7797	11.6641
Overall	8.3766	115.0337	6.8131	32.5431

Table 5: Quantitative errors of the tracker in [6] and the proposed tracker in the sequence shown in Fig. 12.

reliably tracked points cannot be tracked well, the proposed method can lose some accuracy in tracking. Because of the head movement in these sequences, in particular in the sequence in Figure 13, there occurs self-occlusion on the eye feature points and this affects the tracking performance of the reliable points. For this reason, head pose information cannot be obtained very accurately based on the *reliably tracked points* approach, and drifts occur in some feature points. This result can also be clearly seen in the quantitative results. For the proposed method, the errors of the inner eye corner points in θ_y motion sequence (Table 6) are much bigger than the errors in previous experiments. This increases the errors in other feature points as well. For this reason, the overall error of the proposed method for this sequence is also much bigger than the overall errors in

Errors	The method in [6]		The proposed method	
Feature Points No.	Mean	Var	Mean	Var
1	8.5874	3.6709	8.9523	31.5737
2	3.6896	3.7212	4.9679	10.2338
3	4.8703	8.1034	15.4902	362.119
4	24.6714	1149.1813	23.0897	1000.3765
5	67.5781	1576.4321	21.6335	107.3603
6	36.9113	2676.2937	22.5722	988.0865
7	25.9111	274.3535	13.0587	72.5457
8	70.0671	1642.8216	12.9896	83.3209
9	15.9406	271.5229	18.3194	528.204
10	3.9114	5.3791	29.3009	1403.7654
11	49.5711	1158.2236	9.4916	64.7557
12	37.645	1565.7496	23.9482	929.2606
13	2.3963	1.652	7.4228	64.9385
14	3.8976	8.3061	7.9202	40.0682
15	3.3282	3.5374	3.7145	3.5187
16	7.1848	24.0822	6.8959	18.7617
Overall	22.8851	1151.871	14.3605	414.3141

Table 6: Quantitative errors of the tracker in [6] and the proposed tracker in the sequence shown in Fig. 13.

previous experiments. On the other hand, in the T_z motion sequence (Fig. 14 and Table 7), except a small drift in the left mouth corner, there is no significant error in any feature point. Although the errors of the proposed method increase in sequences involving occlusion of reliably tracked feature points, the overall errors of the method in [6] are larger in general. This can also be observed in Fig. 12, 13 and 14 by visually comparing the proposed method and the method in [6]. The method in [6] suffers due to lack of spatial information for eye feature points especially in θ_x and T_z motion sequences (Fig. 12-a and 14-a). Besides, comparing the proposed method with the AAM method in [11] shows that the AAM method fails to fit on the face in the case of head movement, especially when self-occlusion occurs. The reason of failure of the AAM method is again because the technique does not use the occlusion information and cannot adapt

Errors	The method in [6]		The proposed method	
Feature Points No.	Mean	Var	Mean	Var
1	10.0086	4.2787	10.2397	3.2059
2	2.5805	2.0985	2.9314	2.5856
3	6.2958	4.2993	5.5194	3.8763
4	3.6411	1.7353	4.1643	1.7968
5	58.6924	2312.5091	6.7089	32.8487
6	5.8923	11.2332	4.1303	4.9833
7	3.7021	2.9566	4.1804	4.4403
8	2.3474	3.5049	3.0666	2.3461
9	12.2222	38.909	4.8284	9.4
10	13.7604	112.8223	5.1722	29.9485
11	12.8666	40.4387	6.1376	8.7468
12	32.9484	182.0606	3.9329	6.9234
13	2.2256	1.2406	3.6931	1.921
14	3.4794	5.3327	5.4026	11.4034
15	40.1732	303.3658	17.9888	45.3846
16	5.1407	10.4991	5.2426	10.0153
Overall	13.4985	438.8398	5.8337	23.8907

Table 7: Quantitative errors of the tracker in [6] and the proposed tracker in the sequence shown in Fig. 14.

well to scenarios not covered in the training set. As in the case of external occlusions, composing a training set that can fully represent head movements is very hard. Because of this, the AAM method can be useful for facial feature tracking only within the limits of the scenarios covered by the training set.

The *reliably tracked points* approach may have limitations when these points are occluded (e.g., because of large head movements). But, the approach is valid for certain amount of head movement. For instance, in the θ_y motion sequence in Fig. 13-b, there are no drifts in the beginning. But when head turns too much (there occurs self-occlusion on the eye feature points), drifts begin to occur (frame no. 147 and 297). Despite the fact that there can occur drifts in the results of the proposed method, considering the results of all three

methods, the proposed method appears to be a better tracker. If the head pose information can be obtained from an external source then these results will be improved even further.

For further evaluation we have used the Boston head tracking database [39], which includes similar head movements. The results of the proposed method, the method in [6], and the AAM method are shown in Fig. 15. It can be seen that the results of the proposed method are better than the results of the other methods. The method in [6] loses tracking of some eye feature points because temporal information is not enough when the head turns. Again, the AAM method in [11] cannot track facial features accurately since it cannot adapt to head movements not covered in the training set. On the other hand, our method deals well with this kind of head movement. Except some small drifts in eye feature points, there are no significant drifts in the other feature points. This result also shows that although the *reliably tracked points* approach may have limitations, it outperforms the other techniques, and usually deals well with 3-D head movements.

In addition to the experiments shown here, we have also applied our method to sequences in which video resolution is low, there are illumination changes, sequences in which there are in-plane head movements, and as a real-world application sequences taken in a vehicle environment [40]. For these extensive experiments please refer to [41].

6. Conclusion

In this paper a facial feature point tracker that can be used in applications such as human-computer interfaces and facial expression analysis systems is proposed. The proposed tracker is based on a graphical model framework. The facial features are tracked through video streams by incorporating statistical relations in time as well as spatial relations between feature points. By con-

structing both the temporal and spatial relations using graphical models, the proposed method exploits information from both types of relationships in a single, coherent framework. The role of the spatial connections is to build a whole model by binding the feature points and prevent possible drifts occurring due to a number of factors including data quality limitations, occlusions, and head movements. In the case of occlusions, the data in the occluded region become useless. To overcome this, a Gabor feature based occlusion detector is developed and spatial constraints are utilized to prevent drifts because of occlusions. The tracking problem posed in a graphical model framework is solved using a message passing algorithm.

The performance of the tracker is evaluated under various conditions by comparing with two different popular methods which are methods based on Kalman filtering and AAM. Both qualitative visual results and a quantitative evaluation are presented. Our results contain examples demonstrating both successful and imperfect tracking performance of the proposed method. An interesting quantitative result could be the amount of drift present in a feature point before the model breaks down. But, since the spatio-temporal constraints in the proposed method can ensure recovery even in some challenging scenarios, it is not straightforward to characterize this quantitatively. Based on the results we have obtained, it can be concluded that the proposed method provides a promising framework for facial feature tracking. It is a robust tracker for facial expression tracking especially in challenging scenarios involving head movements and occlusions.

Acknowledgements

This work was partially supported by a Turkish Academy of Sciences Distinguished Young Scientist Award, the Turkish State Planning Organization under the DRIVE-SAFE project, an by a graduate scholarship from the Scientific and Technological Research Council of Turkey.

[1] R.-S. Wang, Y. Wang, Facial feature extraction and tracking in video sequences, in: IEEE First Workshop on Multimedia Signal Processing, 1997, pp. 233–238.

[2] J.-W. Kima, M. Song, I.-J. Kim, Y.-M. Kwon, H.-G. Kim, S. C. Ahn, Automatic fdp/fap generation from an image sequence, in: The 2000 IEEE International Symposium on Circuits and Systems, Proceedings. ISCAS Geneva., 2000, pp. 40 – 43.

[3] K.-W. Wong, K.-M. Lam, W.-C. Siu, K.-M. Tse, Face segmentation and facial feature tracking for videophone applications, in: Proceedings of 2001 International Symposium on Intelligent Multimedia, Video and Speech Processing, 2001, pp. 518 – 521.

[4] C. Tomasi, T. Kanade, Detection and tracking of point features, Tech. Rep. CMU-CS-91-132, Carnegie Mellon University (April 1991).

[5] F. Bourel, C. Chibelushi, A. Low, Robust facial feature tracking, in: Proc. 11th British Machine Vision Conference, Bristol, England, 2000, pp. 1:232–241.

[6] G. H, J. Q, Information extraction from image sequences of real-world facial expressions, Mach. Vis. Appl. 16 (2) (2005) 105–115.

[7] J. Q, Y. X, Real-time eye, gaze, and face pose tracking for monitoring driver vigilance, Real-Time Imaging 8 (5) (2002) 357–377.

[8] J. Wieghardt, R. P. Würtz, C. von der Malsburg, Gabor-based feature point tracking with automatically learned constraints, in: Proceedings ECCV 2002, Copenhagen, 2002.

[9] R. S. Feris, R. M. C. Junior, Locating and tracking facial landmarks using gabor wavelet networks, in: ICAPR '01: Proceedings of the Second International Conference on Advances in Pattern Recognition, Springer-Verlag, London, UK, 2001, pp. 311–320.

[10] R. Feris, J. Cesar, R.M., Tracking facial features using gabor wavelet networks, in: Proceedings XIII Brazilian Symposium on Computer Graphics and Image Processing, 2000, pp. 22–27.

[11] T. F. Cootes, G. J. Edwards, C. J. Taylor, Active appearance models, in: ECCV '98: Proceedings of the 5th European Conference on Computer Vision-Volume II, Springer-Verlag, London, UK, 1998, pp. 484–498.

[12] K.-W. Wan, K.-M. Lam, K.-C. Ng, An accurate active shape model for facial feature extraction, Pattern Recogn. Lett. 26 (15) (2005) 2409–2423.

[13] Y. Tong, Y. Wang, Z. Zhu, Q. Ji, Facial feature tracking using a multi-state hierarchical shape model under varying face pose and facial expression, in: 18th International Conference on Pattern Recognition, 2006, pp. 283 – 286.

[14] T. Chen, R. R. Rao, Audio-visual integration in multimodal communication, in: Proc. IEEE, 1998, pp. 837–852.

[15] C. Hu, R. Feris, M. Turk, Real-time view-based face alignment using active wavelet networks, in: IEEE International Workshop on Analysis and Modeling of Faces and Gestures, 2003, pp. 215 – 221.

[16] J. Ahlberg, Using the active appearance algorithm for face and facial feature tracking, in: IEEE ICCV Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems, 2001, pp. 68–72.

[17] T. Goto, S. Kshirsagar, N. Magnenat-Thalmann, Automatic face cloning and animation using real-time facial feature tracking and speech acquisition, IEEE Signal Processing Magazine 18 (2001) 17–25.

[18] J. Xiao, S. Baker, I. Matthews, T. Kanade, Real-time combined 2d+3d active appearance models, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Vol. 2, 2004, pp. 535 – 542.

[19] C. Su, L. Huang, Spatio-temporal graphical-model-based multiple facial feature tracking, EURASIP Journal on Applied Signal Processing 13 (2005) 2091–2100.

[20] A. M. Baumberg, D. C. Hogg, An efficient method for contour tracking using active shape models, in: In Proceeding of the Workshop on Motion of Nonrigid and Articulated Objects. IEEE Computer Society, 1994, pp. 194–199.

[21] D. S. Bolme, Elastic bunch graph matching, Master's thesis, Colorado State University (2003).

[22] J. Pearl, Probabilistic Reasoning in Intelligent Systems, Morgan Kaufman, San Mateo, 1988.

[23] Y. Weiss, Correctness of local probability propagation in graphical models with loops, Neural Comput. 12 (1) (2000) 1–41.

[24] E. Sudderth, Embedded trees: Estimation of gaussian processes on graphs with cycles, Master's thesis, MIT (2002).

[25] A. Azarbayejani, B. Horowitz, A. Pentland, Recursive estimation of structure and motion using relative orientation constraints, in: Computer Vision and Pattern Recognition, 1993. Proceedings CVPR '93., 1993 IEEE Computer Society Conference on, 1993, pp. 294–299.

[26] C. S. Wiles, A. Maki, N. Matsuda, Hyperpatches for 3d model acquisition and tracking, IEEE Trans. Pattern Anal. Mach. Intell. 23 (12) (2001) 1391–1403.

[27] R. Subbarao, P. Meer, Y. Genc, A balanced approach to 3d tracking from image streams, in: ISMAR '05: Proceedings of the 4th IEEE/ACM International Symposium on Mixed and Augmented Reality, IEEE Computer Society, Washington, DC, USA, 2005, pp. 70–78.

[28] M. Cordea, E. Petriu, N. Georganas, D. Petriu, T. Whalen, 3d head pose recovery for interactive virtual reality avatars, Instrumentation and Measurement Technology Conference, 2001. IMTC 2001. Proceedings of the 18th IEEE 1 (2001) 72–77 vol.1.

[29] L. Vacchetti, V. Lepetit, P. Fua, Stable real-time 3d tracking using online and offline information, IEEE Transactions on Pattern Analysis and Machine Intelligence 26 (2004) 1385–1391.

[30] M. Pupilli, A. Calway, Real-time camera tracking using known 3d models and a particle filter, in: Pattern Recognition, 2006. ICPR 2006. 18th International Conference on, Vol. 1, 2006, pp. 199–203.

[31] H. Sidenbladh, M. J. Black, D. J. Fleet, Stochastic tracking of 3d human figures using 2d image motion, in: In European Conference on Computer Vision, 2000, pp. 702–718.

[32] E. Sudderth, M. Mandel, W. Freeman, A. Willsky, Visual hand tracking using nonparametric belief propagation, in: Computer Vision and Pattern Recognition Workshop, 2004. CVPRW '04. Conference on, 2004, pp. 189–189.

[33] E. Trucco, A. Verri, Introductory Techniques for 3-D Computer Vision, Prentice Hall PTR, Upper Saddle River, NJ, USA, 1998.

[34] D. A. Forsyth, J. Ponce, Computer Vision: A Modern Approach, Prentice Hall, 2002.

[35] M. Stegmann, B. Ersboll, R. Larsen, Fame-a flexible appearance modeling environment, Medical Imaging, IEEE Transactions on 22 (10) (2003) 1319–1331.

[36] T. Kanade, J. Cohn, Y.-L. Tian, Comprehensive database for facial expression analysis, in: Proceedings of the 4th IEEE International Conference on Automatic Face and Gesture Recognition (FG'00), 2000, pp. 46 – 53.

[37] R. Gross, I. Matthews, S. Baker, Active appearance models with occlusion, Image and Vision Computing 24 (1) (2006) 593–604.

[38] M. Storer, P. M. Roth, M. Urschler, H. Bischof, J. A. Birchbauer, Active appearance model fitting under occlusion using fast-robust PCA, in: In:

Proc. International Conference on Computer Vision Theory and Applications (VISAPP), 2009, pp. 130–137.

[39] M. L. Cascia, S. Sclaroff, Fast, reliable head tracking under varying illumination, In IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (1998) 322–336.

[40] H. Abut, H. Erdoğan, A. Erçil, B. Çürüklü, H. Koman, F. Taş, A. Argunşah, S. Coşar, B. Akan, H. Karabalkan, E. Çökelek, R. Fıcıcı, V. Sezer, S. Danış, M. Karaca, M. A. M. Uzunbaş, K. Eritmen, C. Kalaycıoğlu, M. Imamoğlu, C. Karabat, M. Peyiç, Data collection with 'uyanik': Too much pain; but gains are coming, in: Proc. Biennial on DSP for In-Vehicle and Mobile Systems, Istanbul, Turkey, 2007.

[41] S. Coşar, Facial feature point tracking based on a graphical model framework, Master's thesis, Sabancı University (January 2008).

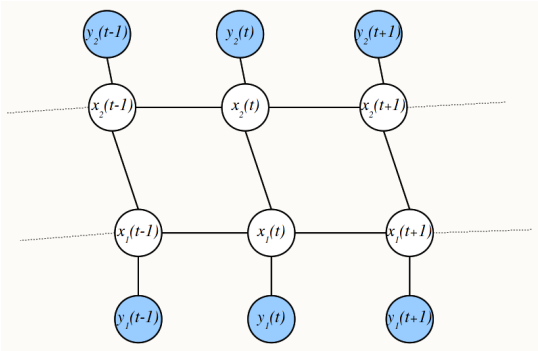

Figure 2: The graphical model used in our framework for the particular case of two feature points.

Figure 3: An example set of facial feature points.

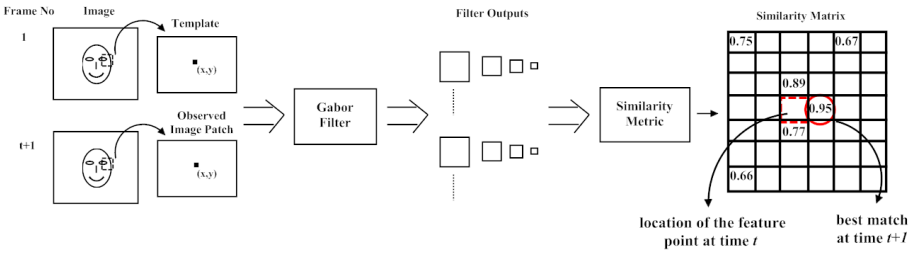

Figure 4: The overall flow of the preprocessing

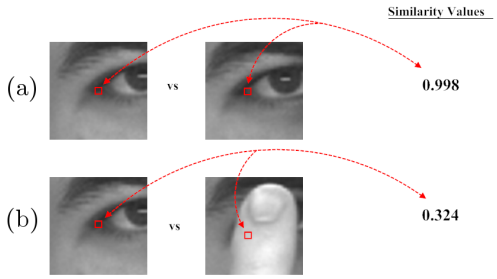

Figure 5: Similarity value outputs of the eye corner feature point when (a) there is no occlusion, (b) there is occlusion

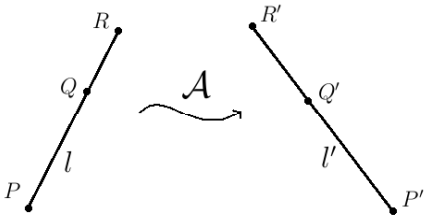

Figure 6: Illustration of line l going through an affine transformation.

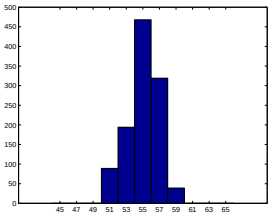

Figure 7: The distribution of the spatial distances on x-axis between the right eye corner feature points for a scenario involving facial expressions without any head movement and/or occlusion.

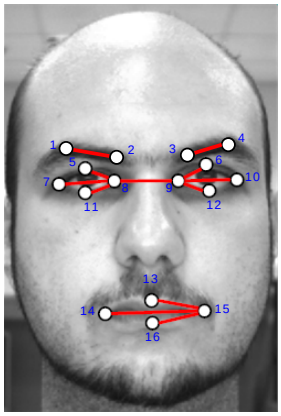

Figure 8: The selected feature points and the spatial connections for the experiments.

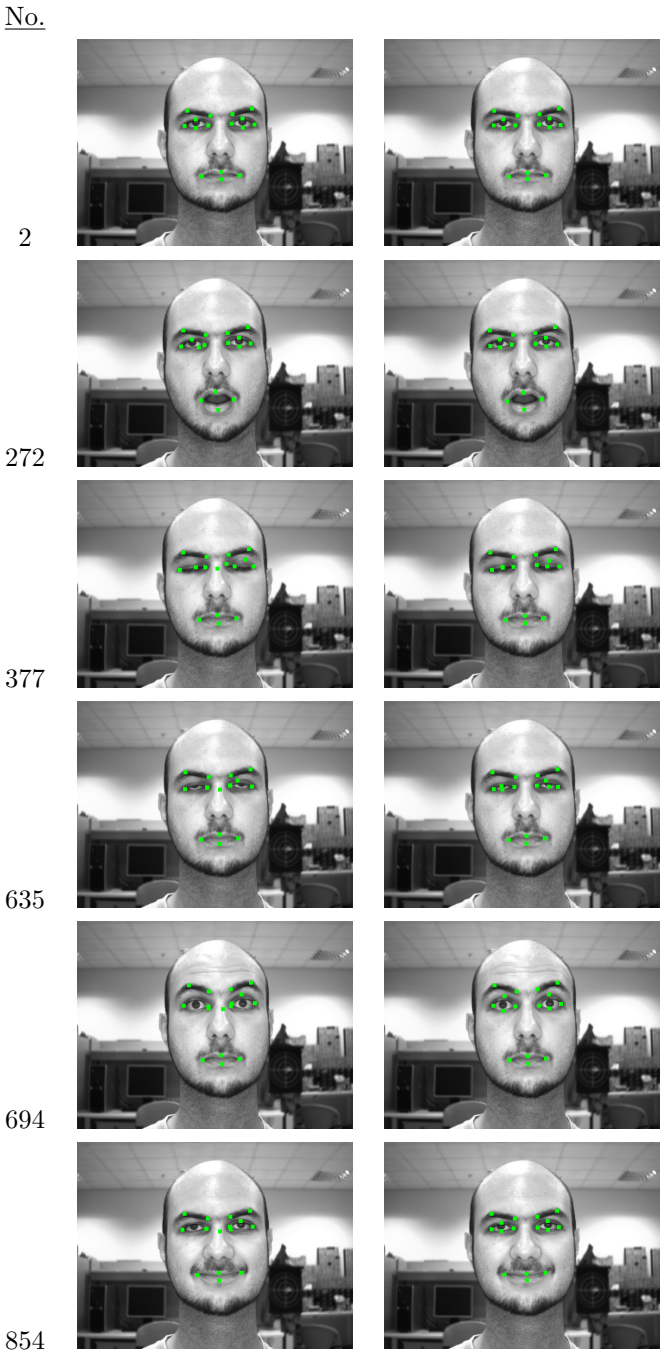

Figure 9: Tracking results of (a) the method in [6] (b) the proposed method for a sequence that includes facial gestures.

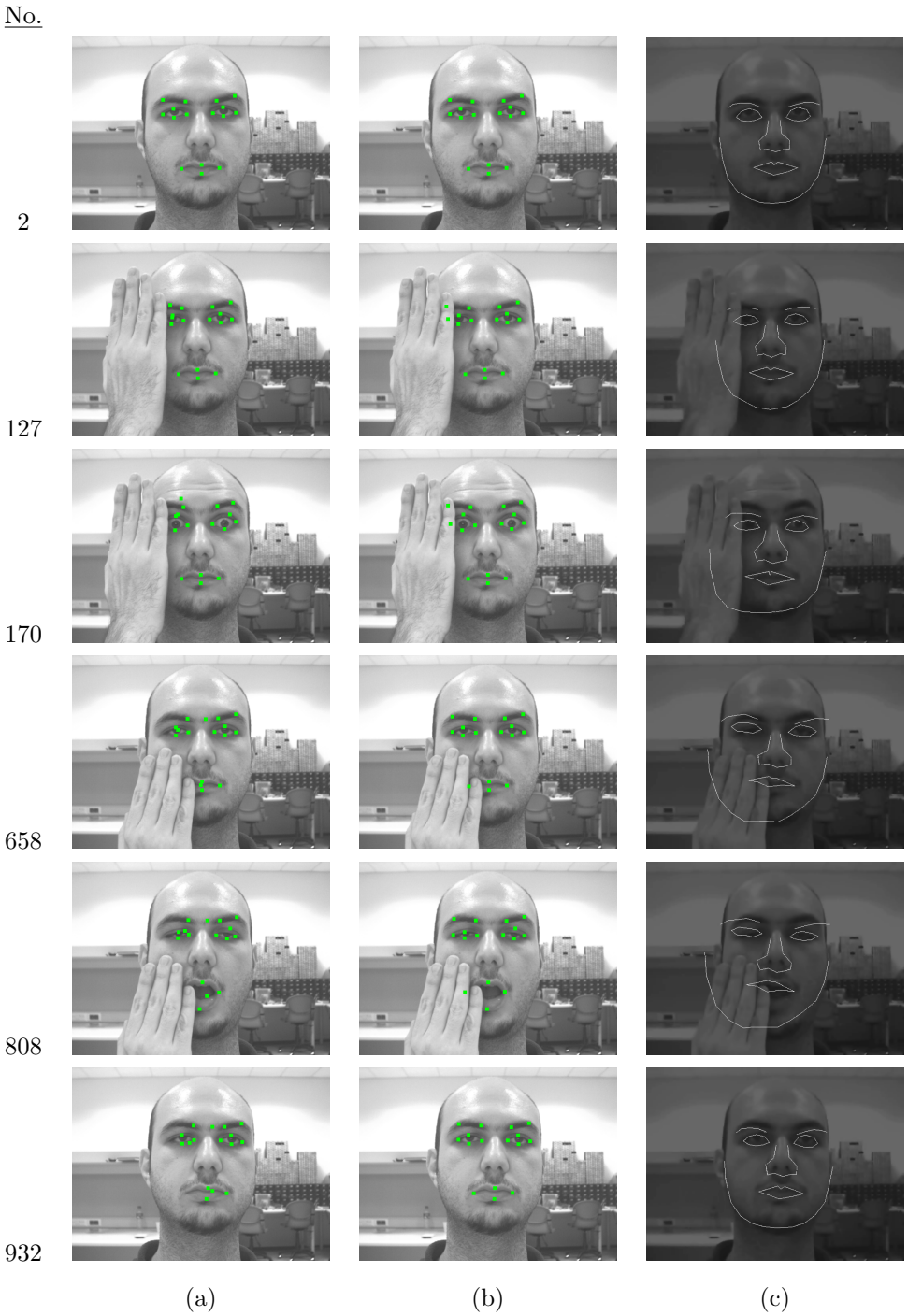

Figure 10: Tracking results of (a) the method in [6] (b) the proposed method (c) the AAM method in [11] for a sequence that includes facial gestures and external occlusion by hand.

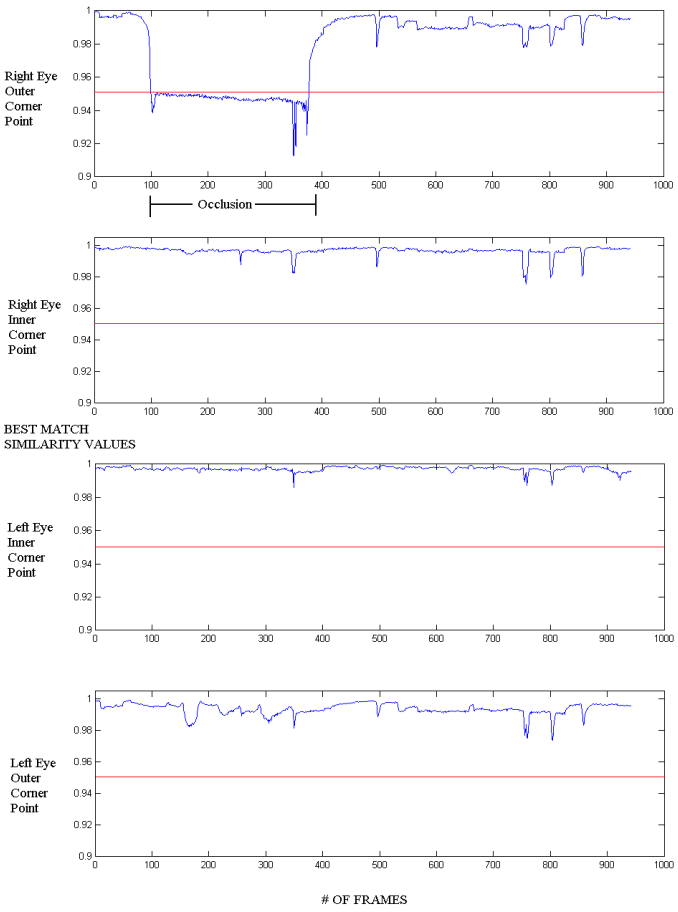

Figure 11: The similarity outputs of four eye corner points for the sequence shown in Fig. 10. Red line represents the threshold value.

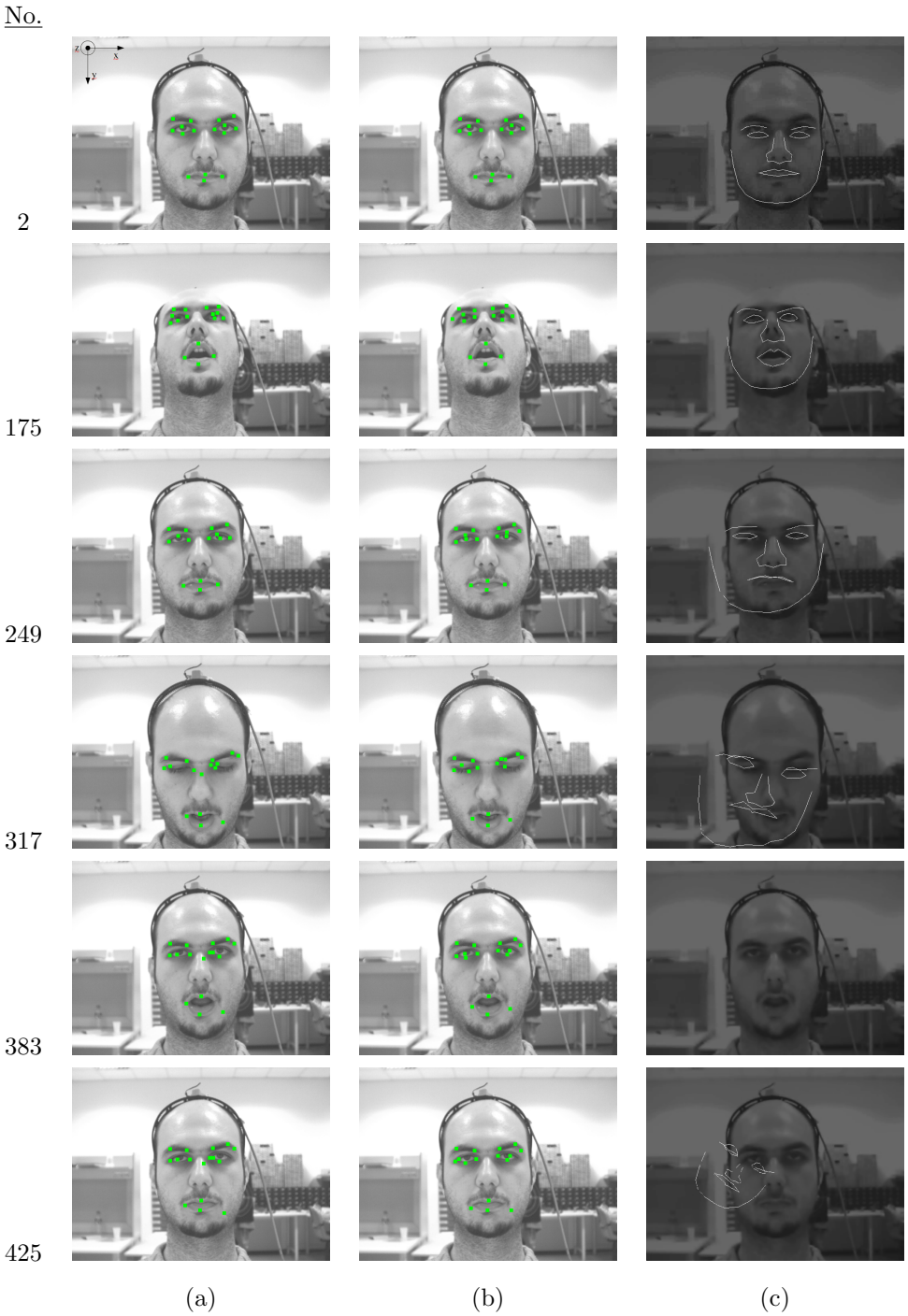

Figure 12: Tracking results of (a) the method in [6] (b) the proposed method (c) the AAM method in [11] for a sequence that includes rotation around x-axis (θ_x) and mouth opening.

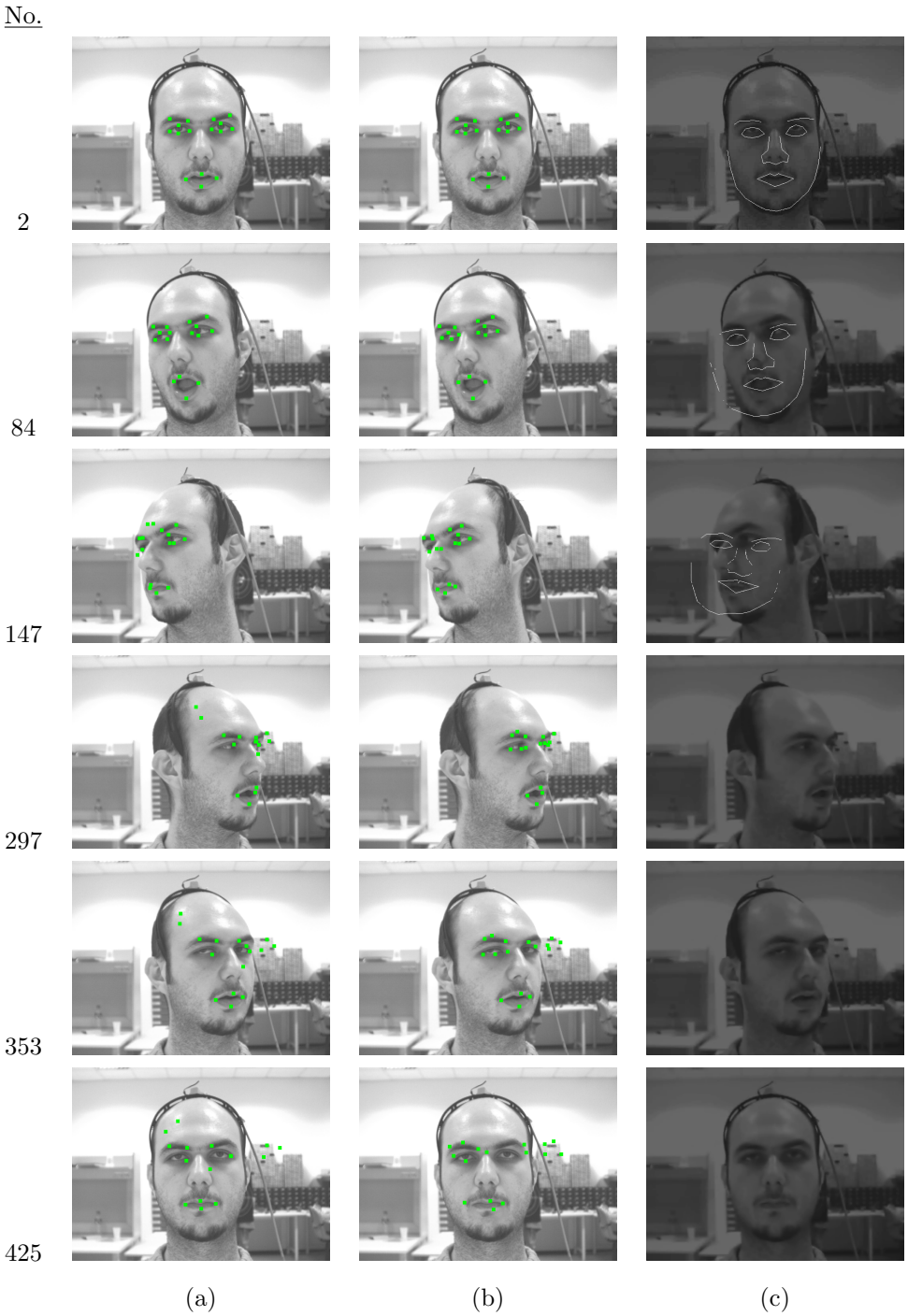

Figure 13: Tracking results of (a) the method in [6] (b) the proposed method (c) the AAM method in [11] for a sequence that includes rotation around x-axis (θ_x) and mouth opening.

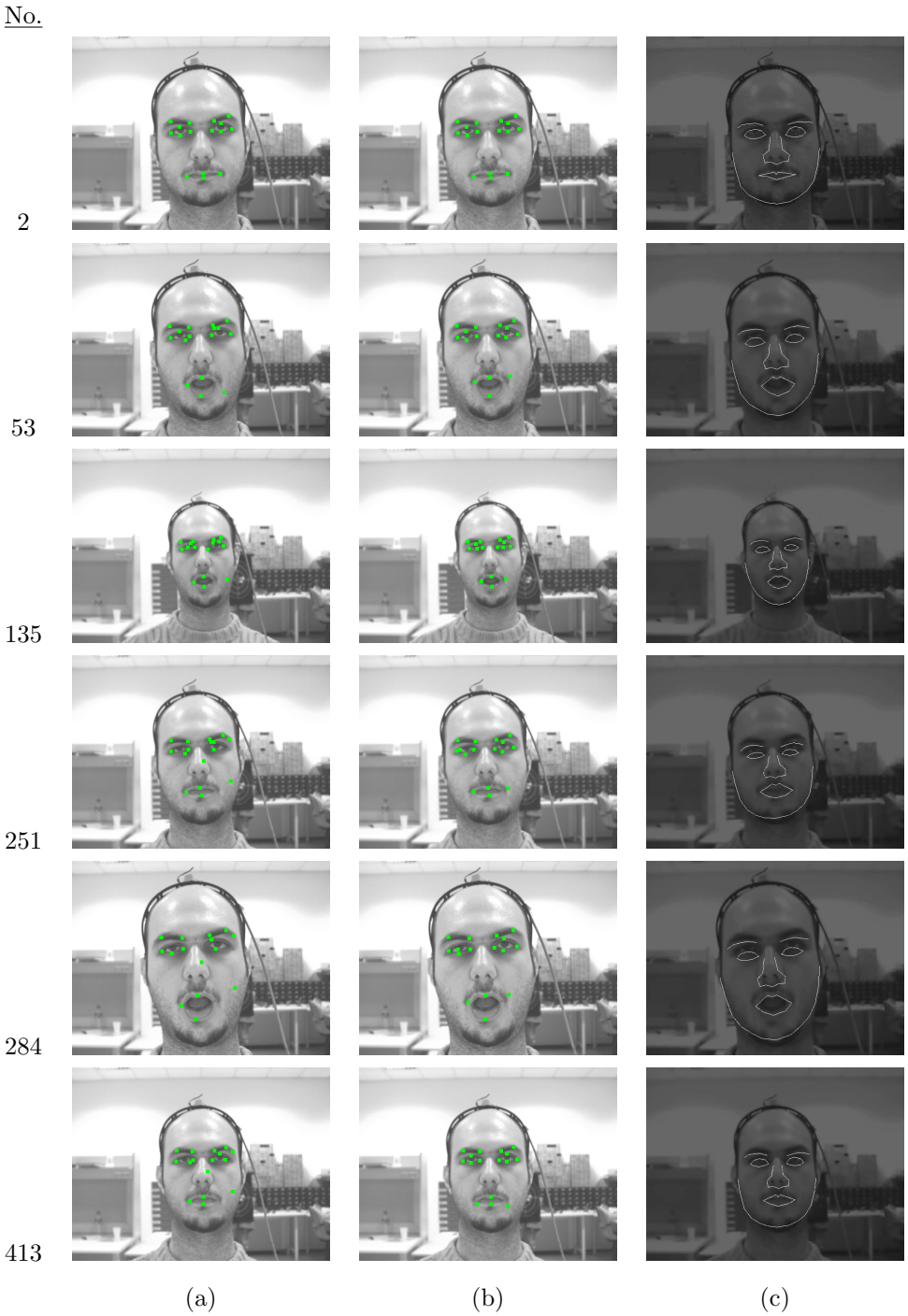

Figure 14: Tracking results of (a) the method in [6] (b) the proposed method (c) the AAM method in [11] for a sequence that includes translation on z-axis and mouth opening.

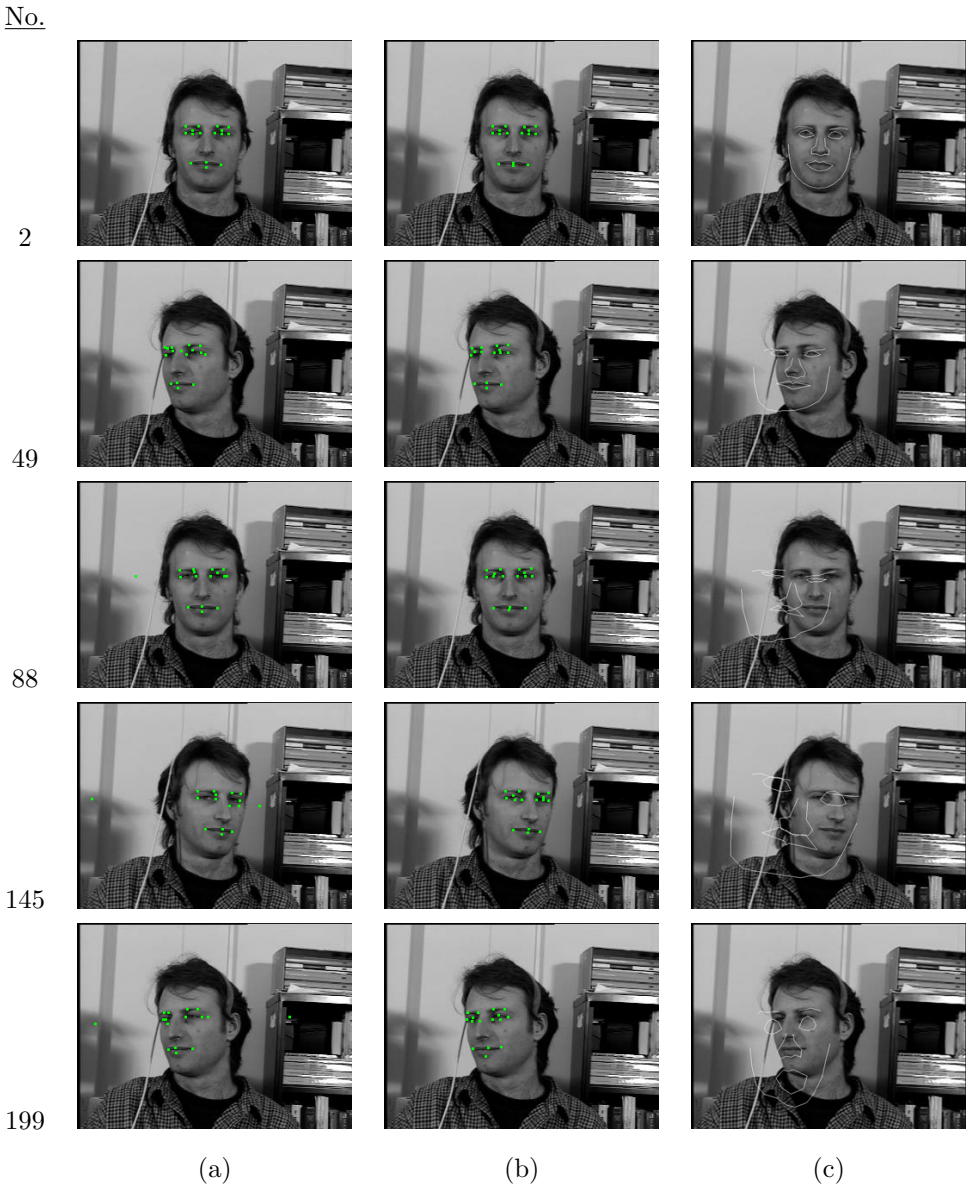

Figure 15: Tracking results of (a) the method in [6] (b) the proposed method (c) the AAM method in [11] for a sequence from Boston head tracking database [39].