MSc Thesis

Automatic Facial Action Detection from Face Video

MMI-MS-2004-01

Graduation Committee:

Advisor: M. Pantic, TU Delft
Advisor: I. Patras, TU Delft

Member: J.F. Cohn, University of Pittsburgh
Member: C.J. Veenman, TU Delft

Chair: E.A. Hendriks, TU Delft

MSc Candidate:

M.F. Valstar

Facial Action Unit Detection in Face Video

M.F. Valstar

Abstract

Enabling computer systems to correctly analyse human behavior is an unsolved problem in Artificial Intelligence. Recognizing human facial expressions automatically by a robot or computer is an important aspect of this analysis. A system that enables fast and robust facial action unit (AU, i.e. atomic facial signal) recognition would have many applications in behavioral science, medicine, security and man machine interaction. By detecting these AUs we can detect any facial expression, because every facial expression can be expressed as a composition of AUs. This work investigates the possibility to detect AUs using temporal templates and tracked facial micro features as a data representation. Temporal Templates is a novel data representation for the detection of facial action from face video. Two systems have been developed, one using temporal template data representation and a two-stage k-Nearest Neighbor-rulebase classifier and the other using the tracked facial micro features data representation and a Support Vector Machine classifier. Both systems have been evaluated on two databases, namely the well-known Cohn-Kanade facial expression database and the recently developed MMI-Face-Database. The experimental results suggest that both data representations are useful for the detection of AUs. We report results for the detection of 21 AUs using temporal templates and results for the detection of 15 AUs using tracked facial micro features.

1 Background

1.1 Research motivations

The human face is involved in an impressive variety of different activities. It houses the majority of our sensory apparatus: eyes, ears, mouth and nose, allowing the bearer to see, hear, taste and smell. Apart from these biological functions, the human face provides a number of signals essential for interpersonal communication in our social life. The face houses the speech production apparatus and is used to identify other members of the species, to regulate the conversation by gazing or nodding, and to interpret what has been said by lip reading. It is our direct and naturally preeminent means of communicating and understanding somebody's affective state and intentions based on the shown facial expression [1]. Personality, attractiveness, age and gender can be also derived from someone's face. Thus the face is a multi-signal sender/receiver capable of tremendous flexibility and specificity.

Automating the analysis of facial signals, especially rapid facial signals (i.e. facial muscle actions), would be highly beneficial for fields as diverse as security, behavioral science, medicine, communication, and education. In security contexts, facial expressions play a crucial role in establishing or detracting from credibility. In medicine, facial expressions are the direct means to identify when specific mental processes are occurring, for instance when somebody is sleepy, upset or bored. In education, pupils' facial expressions inform the teacher of the need to adjust the instructional message. As far as natural interfaces between humans and computers (PCs / robots / machines) are concerned, facial expressions provide a way to communicate basic information about needs and demands to the machine. In fact, automatic analysis of rapid facial signals seem to have a natural place in various vision sub-systems, including automated tools for gaze and focus of attention tracking, lip reading, bimodal speech processing, face / visual speech synthesis, face-based command issuing, and facial affect processing. Where the user is looking (i.e., gaze tracking) can be effectively used to free computer users from the classic keyboard and mouse. Also, certain facial signals (e.g., a wink) can be associated with certain commands (e.g., a mouse click) offering an alternative to traditional keyboard and mouse commands. The human capability to 'hear' in noisy environments by means of lip reading is the basis for bimodal (audiovisual) speech processing that can lead to the realization of robust speech-driven interfaces. To make a believable 'talking head' (avatar) representing a real person, tracking the person's facial signals and making the avatar mimic those using synthesized speech and facial expressions is compulsory. The human ability to read emotions from someone's facial expressions is the basis of facial affect processing that can lead to expanding interfaces with emotional communication and, in turn, to obtaining a more flexible, adaptable, and natural interaction between humans and machines. It is this wide range of principle driving applications that has lent a special impetus to the research problem of automatic facial expression analysis and produced a surge of interest in this research topic [2].

Altough humans are perfectly capable to estimate a persons affectionate state given only a static image, there is undoubtedly more information about facial behavior contained in time-dynamic observations from face video. Therefore we will pursue in this work approaches that aim to benefit from this additional information.

1.2 FACS

Rapid facial signals are movements of the facial muscles that pull the skin, causing a temporary distortion of the shape of the facial features and of the appearance of folds, furrows, and bulges of skin. The common terminology for describing rapid facial signals refers either to culturally dependent linguistic terms indicating a specific change in the appearance of a particular facial feature (e.g., smile, smirk, frown, sneer) or to the linguistic universals describing the activity of specific facial muscles that caused the observed facial appearance changes. There are several methods for linguistically universal recognition of facial changes based on the facial muscular activity [3]. >From those, the facial action coding system (FACS) proposed by Ekman et al. [4, 5] is the best known and most commonly used system. It is a system designed for human observers to describe changes in the facial expression in terms of visually observable activations of facial muscles. The changes in the facial expression are described with FACS in terms of 44 different Action Units (AUs), each of which is anatomically related to the contraction of either a specific facial muscle or a set of facial muscles. Examples of different AUs are given in Fig. 1 . Along with the definition of various AUs, FACS also provides the rules for visual detection of AUs and their temporal segments (onset, apex, offset) in a face image. Using these rules, a FACS coder (that is a human expert having a formal training in using FACS) decomposes a shown facial expression into the AUs that produce the expression. Although FACS provides a good foundation for AU-coding of face images by human observers, achieving AU recognition by a computer is by no means a trivial task. A problematic issue is that AUs can occur in more than 7000 different complex combinations [3], causing bulges (e.g., by the tongue pushed under one of the lips) and various in- and out-of-image-plane movements of permanent facial features (e.g., jetted jaw) that are difficult to detect in 2D face images [2].

1.3 Automating FACS

Few approaches have been reported for automatic recognition of AUs in images of faces. Some researchers described patterns of facial motion that correspond to a few specific AUs, but did not report on actual recognition of these AUs. Examples of such works are the studies of Mase [6], Black and Yacoob [7], and Essa and Pentland [17]. Almost all other efforts in automating FACS coding addressed the problem of automatic AU recognition in face video. To detect 6 individual AUs in face image sequences free of head motions, Bartlett et al. [9] used a $61 \times 10 \times 6$ feed-forward neural network. They achieved 91% accuracy by feeding the pertinent network with the results of a hybrid system combining holistic spatial analysis and optical flow with local feature analysis. To recognize 8 individual AUs and 4 combinations of AUs in face image sequences free of head motions, Donato et al. [10] used Gabor wavelet representation and independent component analysis. They reported a 95.5% average recognition rate accomplished by their method. To recognize 8 individual AUs and 7 combinations of AUs in face image sequences free of head motions, Cohn et al. [11] used facial feature point tracking and discriminant function analysis. They achieved an 85% average recognition rate by their method. Tian et al. [12] used lip tracking, template matching and neural networks to recognize 16 AUs occurring alone or in combination in nearly frontal-view face image sequences. They reported an 87.9% average recognition rate attained by their method. Braathen et al. [13] reported on automatic recognition of 3 AUs using particle filtering for 3D tracking, Gabor filters, Support Vector Machines, and Hidden Markov Models to

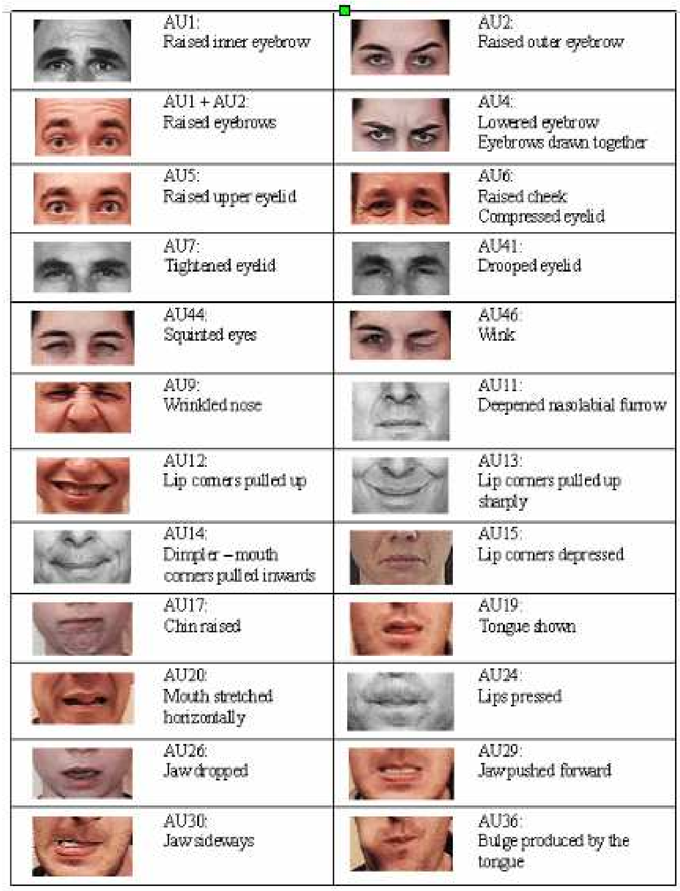

AU1: Raised inner eyebrow
AU2: Raised outer eyebrow
AU1 + AU2: Raised eyebrows
AU4: Lowered eyebrow Eyebrows drawn together
AU5: Raised upper eyelid
AU6: Raised cheek Compressed eyelid
AU7: Tightened eyelid
AU41: Drooped eyelid
AU44: Squinted eyes
AU46: Wink
AU9: Wrinkled nose
AU11: Deepened nasolabial furrow
AU12: Lip corners pulled up
AU13: Lip corners pulled up sharply
AU14: Dimpler – mouth corners pulled inwards
AU15: Lip corners depressed
AU17: Chin raised
AU19: Tongue shown
AU20: Mouth stretched horizontally
AU24: Lips pressed
AU26: Jaw dropped
AU29: Jaw pushed forward
AU30: Jaw sideways
AU36: Bulge produced by the tongue

Figure 1: Some examples of Facial Action Units (AUs)

analyze an input face image sequence having no restriction placed on the head pose.

In contrast to all these approaches to automatic AU detection, which deal only with frontal-view face images and cannot handle temporal dynamics of AUs, Pantic and Patras [14] addressed the problem of automatic detection of AUs and their temporal segments (onset, apex, offset) from profile-view face image sequences. They used particle filtering to track 15 fiducial facial points in an input face-profile video and temporal rules to perform both automatic segmentation and recognition of temporal segments of 23 AUs occurring alone or in a combination in the input video sequence. They achieved an 88% average recognition rate by their method.

The only work reported up to date that addresses automatic AU coding from static face images is the work of Pantic and Rothkrantz [15]. It concerns an automated system for AU recognition in static frontal- and/or profile-view color face images. A multidetector approach to facial feature localization is utilized to spatially sample the face profile contour and the contours of the facial components such as the eyes and the mouth. From the contours of the facial features, 10 profile-contour fiducial points and 19 fiducial points of the contours of the facial components are extracted. Based on these, 32 individual AUs occurring alone or in combination are recognized using rule-based reasoning. With each scored AU, the utilized algorithm associates a factor denoting a certainty with which the pertinent AU is scored. A recognition rate of 86% is achieved by the method.

In summary, four critical issues in automated FACS coding can be distinguished. The first one concerns dealing with head pose variations. Most systems have limited capabilities to overcome the problems caused by variations in head pose. Xiao et. al. [16] reported on a method for recovering the full motion (3 rotations and 3 translations) of the head from an input video using a cylindrical head model. Using the recovered motion parameters the face region can be stabilized and processed with a facial expression analyzer. The second issue concerns the temporal aspects of facial expressions. Timing of different AUs, the speed of the activation of an AU, and onset-apex-offset detection, are just three examples of the temporal aspects of facial action. Researchers in automatic FACS coding are just starting to pursue these issues. Third, occlusions in the face often pose a problem to automatic FACS coding. Beards, moustaches and glasses complicate the detection of AUs. The robustness to these occlusions vary for each system but most publications do not mention how well the proposed system copes with this particular problem. Only Essa and Pentland report a system that is able to handle distractions like facial hair and glasses. The last issue is the lack of a publicly available set of properly annotated reference images and videos. This makes it difficult to benchmark the various systems that have been developed in the past. The current standard is the Cohn-Kanade database (see section 4.1), but there are some serious limitations to this dataset which makes it unsuitable as a proper benchmark set as explain in section 4.1.

In this work, we do not investigate the problems of head pose and the temporal dynamics of facial actions. Regarding the problem of occlusion, the developed systems are robust to the distraction caused by glasses, and in the case of temporal templates data representation the method is also robust to distractions caused by facial hair. Furthermore, we have started the construction of a publicly available database containing videos and still images of subjects displaying all possible facial actions (see section 4.1). We believe that this database is very well suited for benchmarking different automatic FACS coding systems.

Table 1 summarizes the AU detection systems that are discussed above.

1.4 Outline of the paper

The purpose of this work is to investigate new data-representation/classification combinations in search of a way to robustly detect as many different AUs as possible. For this goal, we will use the novel application of the temporal template data representation to facial action detection and we will compare this with a

Table 1: Summary of AU detection systems

System	Head Pose	Face Database	Data representation	Classifier	AUs	Rec. Rate
Bartlett et. al. 1999 [9]	Frontal	System specific	Optical flow, local feature analysis	Neural Network	6	91%
Donato et. al. 1999 [10]	Frontal	System specific	Gabor wavelets	Independent Component Analysis	8	95.5%
Cohn et. al. 1999 [11]	Frontal	Cohn-Kanade	Tracked Facial Features	Discriminant Function Analysis	8	85%
Tian et. al. 2001 [12]	Frontal	Cohn-Kanade	Tracked Facial Features, template matching	Neural Network	16	87.9%
Braathen et. al. 2002 [13]	Frontal	System specific	Tracked Facial Features, Gabor wavelets	Support Vector Machines, Hidden Markov Model	3	87.1%
Pantic and Patras 2004 [14]	Profile	MMI-Face-DB	Tracked Facial Features	Expert System	23	88%
Pantic and Rothkrantz 2004 [15]	Frontal, Profile	System Specific	Contour Regression	Expert System	32	86%

standard data representation, namely, tracked facial features.

The remainder of the paper is organized as follows: Section 2 describes the two data representations whose properties we have studied. Section 3 describes different pattern classifiers we have used. In section 4 we present the experiments carried out on two different datasets: the Cohn-Kanade Database and the MMI-Face-DB. This section also includes a discussion of the experimental results. Finally, in section 5 we present our conclusions and suggestions for future research.

2 Data representation

Two different data representations have been examined for the purpose of AU detection in face video. The first relates to temporal templates, a 2-dimensional representation of a movement in a 3-dimensional space (two spatial dimensions and the dimension of time). Section 2.1 elaborates on temporal templates and how we construct them from input image sequences. A well known deficiency of temporal templates is motion self occlusion [18]. In section 2.1.3 we propose Multilevel Motion History Imaging (MMHI), a new temporal-template-based representation technique that records multiple motion instances at the same coordinates of an image sequence.

The second data representation that we investigated concerns fiducial facial points and the distances between them. In frontal face video we track 20 fiducial facial points and calculate parameters such as distances between specific pairs of these points or spatial deviations of points from their original position. These parameters alter with AU activation. Section 2.2 elaborates on these points and the different features we have used for AU detection.

2.1 Temporal Templates

Bobick and Davis first introduced temporal templates [18]. They are 2D images constructed from image sequences, showing where and (in the case of Motion History Images) when motion occured in an image sequence. They effectively reduce a 3-dimensional spatio-temporal input space (video) to a 2-dimensional output space (image).

If the temporal template is constructed without preserving the information about the time when the movement occurred, we refer to it as a Motion Energy Image (MEI). If instead we preserve the temporal motion history information by assigning different intensities to different moments of motion, we refer to the resulting image as a Motion History Image (MHI). In our system we will only use MHIs since we are interested in the timing of the facial motion caused by the activation of AUs.

2.1.1 Face Image Sequence Registration

To be able to construct temporal templates we either need the background to be static or the motion of the object of interest to be separable from the background. Furthermore, to be able to compare separate temporal templates in a meaningfull way, the faces in the image sequences must have the same position, scale and orientation. Hence, to construct useful comparable temporal

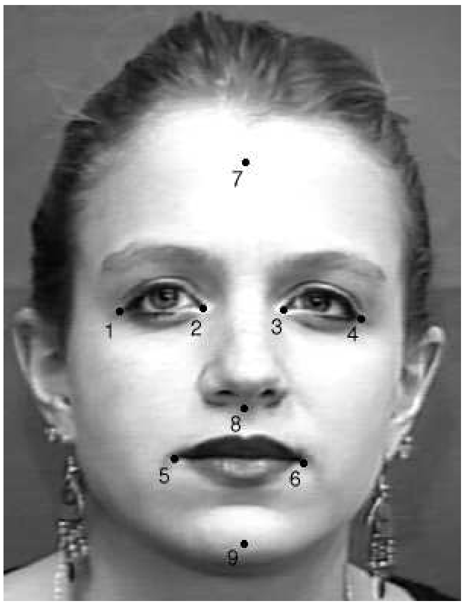


Figure 2: Facial points used for the registration of images for temporal template construction

templates, we need the input face image sequences to be registered in two ways. First, all rigid head movements within one image sequence must be eliminated. Second, all utilized image sequences must have the faces in the same position, orientation and scale.

To achieve both registration, we first select by hand 9 facial points from the first frame of the image sequence (Fig. 2). These points are then tracked in all subsequent frames using a condensation based template tracking technique [19]. The size of the image patch around the tracked points that is used as the template has an impact on the tracking performance. Experimental trials revealed that the tracker perfomed best using a large image patch (100x100 pixels for our images). For registration of each frame with respect to the first frame we apply an affine transformation, using stable facial points whose spatial position remains the same even if a facial muscle contraction (AU activation) occurs. Otherwise, if we would use some other points, we could not be sure whether the movement of a point is due to unwanted rigid head motion or due to the activation of one or more AUs. We call this process intra-registration, as the registration is performed within one image sequence.

The second registration is performed between different image sequences and we will therefore refer to it as inter-registration. All image sequences are registered with respect to a predefined set of facial points. Otherwise faces in different image sequences could have different positions as well as variations in size, making a template-based comparison impossible. This inter-registration process is also carried out by an affine transformation. Under the assumption that all image sequences begin with a neutral facial expression, the transformation matrix is computed by comparing the neutral position of the facial points taken from the first frame of the current image sequence with the predefined position of the same facial points.

2.1.2 Temporal Template Construction

Once properly registered, the available image sequences are used to construct temporal templates. Since we do not employ MEIs in this work, we will only elaborate on the construction of MHIs. Suppose our image sequence consists of k frames. Let $I(x,y,t)$, $t = 1...k$, be a sequence of pixel intensities of the t'th frame and let $D(x,y,t)$ be the binary image indicating regions of motion that results from pixel-intensity-change detection, that is by thresholding

$$|I(x,y,t) - I(x,y,t-1)| > th_I \tag{1}$$

where x and y are the spatial coordinates of picture elements and th_I is the intensity difference threshold between two images for motion detection, a parameter that has to be determined experimentally.

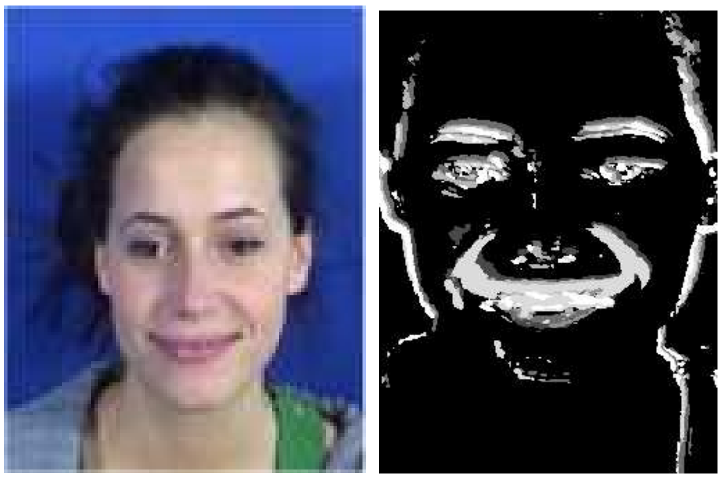
In a MHI, say H_τ, the pixel intensity is a function of the temporal history of motion at that point with τ being the period of time to be considered. Bobick and Davis' implementation of the MHI is as follows:

$$H_\tau(x,y,t) = \begin{cases} \tau & D(x,y,t) = 1 \\ max([H_\tau(x,y,t-1) - 1], 0) & otherwise \end{cases} \tag{2}$$

Bobick and Davis studied body gestures. In their problem definition it is not known when the movement of interest begins or ends. Therefore they need to vary the observed period τ and try to classify all the resulting MHIs. Because we assume that the beginning and end of a facial expression are known and coincide with the duration of an image sequence, we do not need to vary τ. Because of this we are able to normalize the temporal behavior by distributing the grey values in the MHI over the full range of our output image (0-255, assuming that we are using 8 bit greylevel images). Thus, variations in display duration of an AU are canceled out, which makes it possible to compare facial expressions that have a different period but are otherwise identical.

Initially the image sequences may have a different number of frames. So, while the MHIs are temporally normalized, the number of history levels in them may still differ from one image sequence to another. To be able to compare the sequences properly we want to create all MHIs containing the same fixed number of history levels n_h. Therefore the image sequence is sampled to $n_h + 1$ frames. The number of history levels is experimentally determined to be the number that, when used to construct the MHIs, results in the highest recognition rate. Using the known parameter n_h we modify the MHI operator into:

$$H(x,y,t) = \begin{cases} s * t & D(x,y,t) = 1 \\ H(x,y,t-1) & otherwise \end{cases} \tag{3}$$


(a) Apex frame of an image sequence displaying a smile

(b) MHI constructed from the image sequence that contained the frame shown in fig. a

Figure 3: MHI construction.

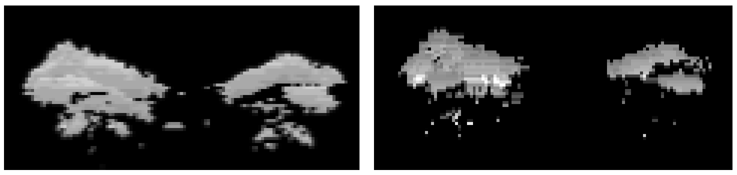


Figure 4: MHI of brows moving up (left) and down (right)

where $s = 255/n_h$ is the intensity step between two history levels and $H(x, y, t) = 0$ for $t < 0$. Fig. 3 shows an image of a smile (AU6 + AU12 + AU25) at apex and the corresponding MHI.

2.1.3 Multilevel Motion History Images

A drawback innate to temporal templates as proposed by Bobick and Davis is the problem of motion self occlusion due to overwriting. Let us explain this problem by giving an example. Let us denote an upward movement of the eyebrows as action A_1 and a downward movement of the eyebrows back to the neutral position as action A_2. Figure 4 shows one MHI of the brows moving up and one MHI of the brows moving down. Both actions produce apparent motion in the facial region above the neutral position of the eyebrows. If A_2 follows A_1 in time and if the motion history of both actions is recorded within a single

MHI, then the motion of history of action A_2 overwrites the motion history of A_1; the information about the motion of action A_1 is utterly lost. To overcome this problem, we propose a way to record the motion history at multiple time intervals, constructing Multilevel Motion History Images (MMHIs), instead of recording the motion history once for the entire image sequence and constructing a single MHI.

In a MMHI, we want to encode motion occurring at different time instances t on the same spatial location $\{x, y\}$, such that it is uniquely decodable later on. To do so, we use a simple bitwise coding scheme. If motion occurs at time instance t at a pixel $\{x, y\}$, we add to the current value of the MMHI $M(x, y, n-1)$ a value that is uniquely related to t. Using this coding scheme results in the following representation for a MMHI constructed from an image sequence of n frames:

$$M(x, y, n) = \sum_{t=1}^{n} D(x, y, t)\, 2^{t-1} \qquad (4)$$

Because of the bitwise coding scheme, we are able to separate multiple motions occurring at the same position in the classification stage.

2.2 Fiducial Facial Points and Distances

Many AUs can be described by specific movements of a few characteristic facial points. Raising or lowering the eyebrows, dropping the jaw, parting the lips, etc. can all be detected by applying a temporal analysis of spatial relations between fiducial facial points.

2.2.1 Face Model

A face model is described by the set P of fiducial facial points. The choice of points consents to two requirements. First, spatio-temporal changes in the position of point P_i in the $\{x, y\}$ plane of a frontal face video must provide information about the activation of at least one AU. The corners of the eyebrows, eyes and mouth are all good examples of points that provide such information. The second requirement is that a point is tracable using a state of the art point tracker. Although the motion of a point positioned on the cheek would give us very valuable information about the activation of AU6 indeed, it is impossible to track such a point due to the lack of 'facial landmarks' surrounding such a point.

Our face model is described by a set of 20 fiducial facial points, as shown in Fig. 5 . Allthough all points seemed to be traceable, it turned out during the experiments that the points F, F1, G and G1 are difficult to track robustly. Motion of point N does not by itself contribute to the detection of AUs. However, it is a stable point [15] tracked extremely robustly. This allows us to use it for registration purposes and as a reference point for calculating various distances to other fiducial facial points. We will elaborate on this in section 2.2.3.

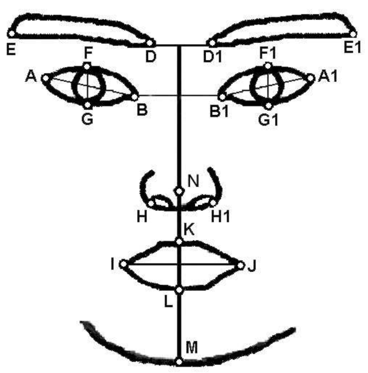


Figure 5: Frontal face model consisting of 20 fiducial facial points

2.2.2 Facial Point Tracker

At this moment the initial positions of these points have to be selected manually in the first frame. The positions in all subsequent frames are determined with a tracker using Particle Filtering with Factorized Likelihoods [20]. Particle Filtering with Factorized Likelihoods is an extension to the Auxiliary Particle Filtering theory introduced by Pitt and Shephard [21], which itself is an extension to the classical particle filtering, or Condensation, proposed by Isard and Blake [19]. We will provide a short overview of how this tracker works. An introduction to Condensation and Particle Filtering with Factorized Likelihoods is given. For detailed descriptions of the various methods please refer to the relevant papers.

Condensation The main idea of particle filtering is to maintain a particle based representation of the *a posteriori* probability $p(\alpha \mid Y)$ of the state α given all the observations Y up to the current time instace. This means that the distribution $p(\alpha \mid Y)$ is represented by a set of pairs $\{(s_k, \pi_k)\}$ such that if s_k is chosen with probability equal to π_k, then it is as if s_k was drawn from $p(\alpha \mid Y)$. In the particle filtering framework our knowledge about the *a posteriori* probability is updated in a recursive way. Suppose that we have a particle based representation of the density $p(\alpha^- | Y^-)$, that is, we have a collection of K particles and their corresponding weights (i.e. $\{(s_k^-, \pi_k^-)\}$). Then, the Condensation Particle Filtering can be summarized as follows:

1. Draw K particles s_k^- from the probability density that is represented by the collection $\{(s_k^-, \pi_k^-)\}$.

2. Propagate each particle s_k^- with the transition probability $p(\alpha|\alpha^-)$ in order to arrive at a collection of K particles s_k.

3. Compute the weights π_k for each particle as follows,

$$\pi_k = p\left(y \mid s_k\right) \tag{5}$$

Then normalise so that $\sum_k \pi_k = 1$.

This results in a collection of K particles and their corresponding weights (i.e. $\{(s_k, \pi_k)\}$ which is an approximation of the density $p\left(\alpha|Y\right)$.

Factorized Likelihoods Classical particle filtering has three major drawbacks. The first drawback is that a large amount of particles that result from sampling from the proposal density $p\left(\alpha|Y^-\right)$ (i.e. step 1 of the Condensation algorithm) might be wasted because they are propagated into areas with small likelihood, as determined in step 3 of the Condensation algorithm. The second problem is that the above scheme ignores the fact that while a particle $s_k = \langle s_{k1}, s_{k2} ... s_{kN}\rangle$ might have low likelihood, it can easily happen that parts of it might be close to the correct solution. Finally, the third problem is that the estimation of the particle weights does not take into account the interdependencies between the different parts of the state α.

Particle filtering with factorized likelihoods [20] attempts to solve these problems in one step, given the case that the likelihood can be factorized, that is in the case that $p\left(y|\alpha\right) = \prod_i p\left(y|\alpha_i\right)$. It uses a proposal distribution $g\left(\alpha\right)$ the product of the posteriors of each α_i given the observations, that is $g\left(\alpha\right) = \prod_i p\left(\alpha_i|y\right)$, from which we draw samples s_k. These samples are then assigned weights π_k, using the same proposal distribution. We now find π_k and s_k as follows:

1. Propagate all particles s_k^- via the transition probability $p\left(\alpha_i|\alpha^-\right)$ in order to arrive at a collection of K sub-particles μ_{ik}. Note, that while s_k^- has the dimensionality of the state space, the μ_{ik} have the dimensionality of the partition i.

2. Evaluate the likelihood associated with each sub-particle μ_{ik}, that is let $\lambda_{ik} = p(y|\mu_{ik})$.

3. Draw K particles s_k^- from the probability density that is represented by the collection $\{(s_k^-, \lambda_{ik}\pi_k^-)\}$.

4. Propagate each particle s_k^- with the transition probability $p\left(\alpha_i|\alpha^-\right)$ in order to arrive at a collection of K sub-particles s_{ik}. Note, that s_{ik} has the dimensionality of the partition i.

5. Assign a weight π_{ik} to each subparticle as follows, $w_{ik} = \frac{p(y|s_{ik})}{\lambda_{ik}}$, $\pi_{ik} = \frac{w_{ik}}{\sum_j w_{ij}}$. With this procedure, we have a particle-based representation for each of the N posteriors $p\left(\alpha_i \mid y\right)$. That is, we have N collections $(s_{ik},)\, \pi_{ik}$, one for each i.

6. Sample K particles from the proposal function $g(\alpha) = \prod_i p(\alpha_i \mid Y)$. This is approximately equivalent to constructing each particle $s_k = \langle s_{k1} ... s_{ki} ... s_{kN} \rangle$ by sampling independently each s_{ik} from $p(\alpha_i \mid Y)$.
7. Assign weights π_k to the K samples as follows:

$$\pi_k = \frac{p(s_k|Y^-)}{\prod_i p(s_{ik}|Y^-)} = \frac{\sum_l p(s_k|s_l^-) p(s_l^-|Y^-)}{\prod_i \sum_l p(s_{ik}|s_l^-) p(s_l^-|Y^-)} \tag{6}$$

The weights are normalized to sum up to one. With this, we end up with a collection $\{(s_k, \pi_k)\}$ that is a particle-based representation of $p(\alpha|Y)$.

In the above algorithm , steps 1 to 3 represent the essence of the modification of the Condensation Particle filtering made by the Auxiliary Particle Filtering. By first propagating all particles at time $t-1$ we can evaluate their likelihood λ_k. Particles with high λ_k, that is, particles which end up at areas with high likelihood when propagated with the transition density, are favored in step 3, overcoming the first major drawback of classical particle filtering.

Rigid and morphologic observation models In steps 2 and 5 of the PFFL the likelihood and weight of a sub-particle are determined by applying an observation model. For the system described in this paper we use two different models. Both models are robust color-based observation models for template-based tracking. The first model is suitable for the tracking of rigid motion of the template around a facial micro feature. The second model however, allows for minor morphologic transformations of the template. The models are initialized in the first frame of an image sequence when a set of N windows are centered around the facial micro-features that the user pointed and that will be tracked for the rest of the image sequence. Let us denote with o_i the template feature vector, which contains the RGB color information at window i.

We need to define $p(y|\alpha_i)$. Let us denote with q_i the template feature vector that contains the RGB color information at the window around α_i. We use a color-based difference between the vectors o_i and q_i that is invariant to global changes in the intensity as follows:

$$c_1(o_i, q_i) = \left(\frac{o_i}{E\{o_i\}} - \frac{q_i}{E\{q_i\}} \right) \tag{7}$$

where $E\{x\}$ is the mean operator on x. It is easy to show that the color difference vector $c_1(o_i, q_i)$ is invariant to global changes in the light intensity. Finally, we define the scalar color distance using a robust function ρ. Let us denote with j the pixel index and with $c_{1j}(o_i, q_i)$ the color difference at pixel j. The scalar color distance is then defined as:

$$d_c(o_i, q_i) = E_j\{\rho(c_{1j}(o_i, q_i))\} \tag{8}$$

where the robust function that has been used in our experiments is the L_1 norm (see section 3.1).

The second model allows for non-rigid deformations of the initial template, as mentioned above. Let us denote this unknown transformation with $\phi : N^2 \rightarrow N^2$, a transformation that gives the correspondence between the pixel coordinates of the color template c and the image patch $y(\alpha_i)$. The distance metric d_m for the second model contains two terms: the first term is similar to the distance measure for the rigid observation model, only now we take the minimum color distance for all possible deformations ϕ. The second term, $d_s(\phi)$, is a measure of the shape deformation that is introduced by the transformation ϕ. The distance measure is the minimum over all possible transformations, formally:

$$d_m(y(\alpha_i), o) = min\left(d_c(o, y(\alpha_i, \phi)) + \lambda d_s(\phi)\right) \qquad (9)$$

where the first term is used to penalize large color-based distances, the second term is used to penalize large shape deformations and the parameter λ controls the balance between the two terms. For details on the distance term $d_s(\phi)$ and the transformation ϕ, please read [22]. Finally, the observation likelihood reads:

$$p(y|\alpha_i) = e^{-d(y(\alpha_i), o_i)} \qquad (10)$$

where $d(x, y)$ is either the distance measure d_c defined in (8) or the distance measure d_m defined in (9) depending on which observation model is applied.

2.2.3 Parametric Representation

After tracking n fiducial facial points in an image sequence containing l frames, we obtain a set of coordinates P with dimensionality lxn. In order to extract parameters that are invariant to rigid head motions within one image sequence we first intra-register all frames within one sequence by subtracting point N from the coordinates of all facial points. Variations in the relative positions of the facial points between different subjects[1] are minimized by applying a transformation T. The transformation in question is obtained by comparing facial points B, B1 and N with their corresponding points in a pre-defined 'normal' face. Thus, the registered points P_i' are obtained as:

$$P_i' = T(P_i - N) \qquad (11)$$

From the set of points $P = \langle P_1 ... P_n \rangle$ we extract a set of parameters F with dimensionality $l * d$. The parameters we extract are given in table 2, based on the rules for AU activation as described in [15].

3 Pattern Classification

We used several different learning algorithms for the classification of data patterns existing in our input data into the requested output classes (AUs). The

[1]Except for identical twins without scars, no two faces are the same. Hence, for example the distances between the mouth corners varies with every subject. We map every subject to a 'normal' face to make the comparison between subjects possible.

Table 2: Parametric representation of changes in facial feature points

Feature	Features for facial point features
$Edist\,(x, y)$	Euclidian distance between the points x and y
$EdistIncrease\,(x, y,)\,n$	The Euclidian distance increase between points x and y at frame n relative to their distance at frame 1
$xDistFromN\,(x)$	The vertical distance between point x at frame $n$an point x at frame 1
$yDistFromN\,(x)$	The horizontal distance between point x at frame $n$an point x at frame 1

utilized algorithms vary from very simple (k-Nearest Neighbour) to state-of-the-art (Support Vector Machines with Probabilistic Active Learning) learning algorithms. However, a complex classifier like Support Vector Machines (SVMs) might not be always well suited for the task at hand, depending on the data representation, as we will show in section 4.

3.1 k-Nearest Neighbour

The employed k-Nearest Neighbour (kNN) algorithm is straightforward: for a test sample it uses a distance metric to compute which k (labeled) training samples are "nearest" to the sample tested for. It then casts a majority vote on the labels of the nearest neighbours to decide the class of the test sample. Parameters of interest are the distance metric being used and k, the number of neighbours to consider. Apart from the domain expertise, there is no easy way to determine what distance metric to use or what value k should take.

Some examples of distance metrics are the Minkowski distance:

$$L_m\left(\boldsymbol{a}, \boldsymbol{b}\right) = \left(\sum_{i=1}^{d} |a_i - b_i|^m\right)^{1/m} \tag{12}$$

of which the well known Euclidian distance (L_2 norm) and Manhattan or city-block distance (L_1 norm) are special cases. The Manhattan distance is the sum of the distances of the projections of the points on a set of predefined perpendicular axes. The Tanimoto metric, which find most use in taxonomy, is defined as

$$D_{Tanimoto}\left(S_1, S_2\right) = \frac{n_1 + n_2 - 2n_{12}}{n_1 + n_2 - n_{12}} \tag{13}$$

where n_1 and n_2 are the number of elements in sets S_1 and S_2, respectively, and n_{12} is the number of elements that is in both sets. Another distance that will prove useful in combintation with a MMHI data representation is the Chamfer distance. It is defined as follows:

$$D_{Chamfer}(\boldsymbol{a}, \boldsymbol{b}) = \sum_{i=1}^{l} min_k |a_i - b_k| + \sum_{i=1}^{m} min_k |b_i - a_k| \tag{14}$$

where $\boldsymbol{a}$ and $\boldsymbol{b}$ are sets with cardinality l and m, respectively. When applied to Multilevel Motion History Images, the entries in the sets $\boldsymbol{a}$ and $\boldsymbol{b}$ indicate which history levels are active (see section 2.1.3). Although kNN is non-linear in concept and has achieved good results in a wide range of problems, there are a number of major drawbacks. The first major drawback is that for each test sample to classify, the distance of the test sample to each and every sample in the training set has to be computed after which all these distances have to be sorted. Besides the problem of a slow classification process, this also means that for the classifier to work, all traindata needs to be stored.

The second major drawback is that while kNN works very good when the train data is a good estimation of the underlying probability density function, it's generalisation ability is not optimal when we have a limited number of training examples.

3.2 Sparse Network of Winnows

A Sparse Network of Winnows (SNoW) [23] is an information processing structure that consists of an input layer of nodes and an output layer of target nodes. It learns a sparse network of linear functions in which the target concepts (classes) are represented as linear functions over a common feature space.

Nodes in the input layer of the network represent simple relations over the input and are being used as the input features. Given a set of relations (i.e. types of features) that may be of interest in a set of input features, feature set is mapped into a set of features which are *active* (present) in it; this representation is presented to the input layer of SNoW and propagated to the target nodes.[2] Target nodes are linked via weighted edges to (some of the) input features. Let $A_t = \{i_1, ..., i_m\}$ be the set of features that are active in an example and are linked to the target node t . Then the target node is *active* if and only if $\sum_{i \in A_t} w_i^t > \theta_t$, where w_i^t is the weight on the edge connectiong the ith feature to the target node t, and θ_t is the threshold.

The training algorithm is on-line and mistake-driven. Several update rules can be used within SNoW. The most succesful update rule and the only one used in our experiments is a variant of Littlestone's Winnow update rule, a multiplicative update rule tailored to the situation in which the set of input features is not known a priori. The Winnow update rule has, in addition to the threshold θ_t at the target t, two update parameters: a promotion parameter $\alpha > 1$ and a demotion parameter $0 < \beta < 1$. These parameters are used to update the current representation of the target t (the set of weights w_i^t), but

[2]Features may take either binary values, just indicating if the features are active, or real values, reflecting the strength of the activation in question. In our experiments, we used only binary values.

only when a mistake in the prediction is made. Let $A_t = \{i_1, ..., i_m\}$ be the set of active features that are linked to the target node t. If the algorithm predicts -1 (that is, $\sum_{i \in A_t} w_i^t \leq \theta_t$) and the real label is 1 (false negative), the active weights in the current example are promoted in a multiplicative fashion: $\forall i \in A_t$, $w_i^t \leftarrow \alpha w_i^t$. And if the algorithm predicts 1 ($\sum_{i \in A_t} w_i^t > \theta_t$) while the real label is -1 (false positive), the active weights in the current example are demoted: $\forall i \in A_t$, $w_i^t \leftarrow \beta w_i^t$. All other weights are unchanged.

The key feature of the Winnow update rule is that the number of examples it requires to learn a linear function grows linearly with the number of relevant features and only logarithmically with the total number of features. This is what makes SNoW appealing for use in combination with temporal templates: although the temporal templates have a large dimension, typically only specific regions (pixels) within the temporal templates are important (and thus active, in SNoW sense) for certain AUs.

3.3 Expert Systems

Knowledge-based systems or Expert Systems have formed a sub-field of artificial intelligence for some three decades now. It investigates knowledge models and reasoning techniques that might assist a human decision maker. Expert systems have been defined in various ways, but all definitions agree that expert systems are artificial means used to emulate the decision-making ability of a human expert.

Expert systems have two main parts, namely, a knowledge base and an inference engine. The knowledge base contains knowledge about the problem domain, usually in the form of heuristic rules. The inference engine uses the rules to infer appropriate conclusions based on relevant portions of the knowledge base and a set of facts that form the current input to the system.

3.4 Support Vector Machine

Support Vector Machines (SVMs) have proven to be extremely efficient classifiers, achieving classification rates unparaleled by any other classifier in domains as diverse as marine biology, face detection and speech recognition. They are sparse, non-linear and generalize very well given only a small training set. But probably the most important aspect is the well-founded mathematical theory [24] on which the classifier is based. It is inspired by statistical learning theory that performs structural risk minimization on a nested set structure of separating hyperplanes. SVMs may be used for regression, binary and multi-class classification. The essence of SVMs can be summarized in three steps: maximizing the separating hyperplane margin, mapping the input space to a (hopefully) linearly separable feature space and applying the 'kernel trick' to combine the first two steps in a computationally efficient way.

Maximizing the margin of the separating hyperplane ($\boldsymbol{w} + b$) results in a high generalization ability. Generalization bounds can be found in [24, 25, 26]. To wit, it is the problem of finding the hyperplane that maximizes the distance

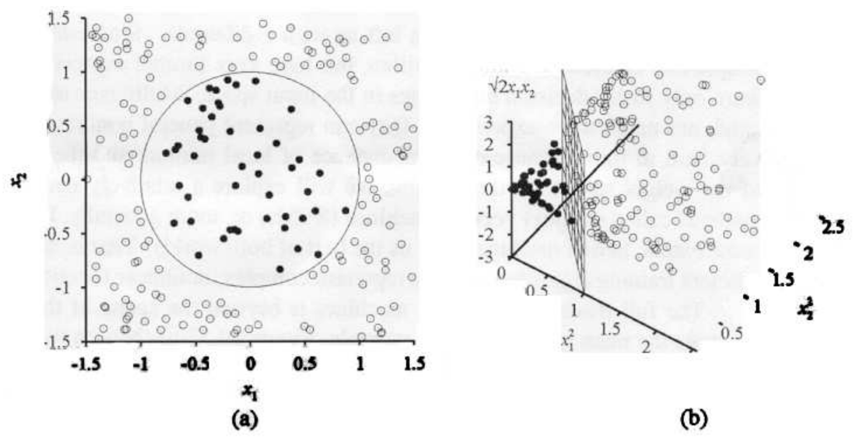


Figure 6: (a) A two-dimensional training with positive examples as black dots and negative examples as white dots. The true decision boundary, $x_1^2 + x_2^2 \leq 1$, is also shown. (b) The same data after mapping into a three dimensional feature space $\left(x_1^2, x_2^2, \sqrt{2}x_1x_2\right)$

between the support vectors (SVs) and $\boldsymbol{w}$. We refer to the margin of the function output as the *functional margin*. The generalisation bounds and optimisation theories use instead the *geometric margin,* which is the functional margin of a normalised weight vector $w/\left\|w\right\|$. Hence, we can equally well optimise the geometric margin by fixing the functional margin to be equal to 1 and minimising the norm of the weight vector (resulting in a canonical hyperplane). Suppose we are given a set of examples $(\boldsymbol{x}_1, y_1), \ldots (\boldsymbol{x}_l, y_l)$, $\boldsymbol{x} \in R^N$, $y_i \in \{-1, +1\}$ we achieve the maximal margin hyperplane with geometric margin $\gamma = 1/\left\|w\right\|_2$ as follows:

$$\begin{array}{cc} minimize_{\boldsymbol{w},b} & \langle \boldsymbol{w} \cdot \boldsymbol{w} \rangle \\ subject\,to & y_{i(\langle \boldsymbol{w} \cdot \boldsymbol{x}_i \rangle + b) \geq 1, i=1\ldots l} \end{array} \tag{15}$$

Off course, most real-world problems are not linearly separable in input space. To enrich the hypothesis space of the learning algorithm presented above and to overcome this problem, in step 2 we map each input sample $\boldsymbol{x}$ to its representation in feature space $\Phi(\boldsymbol{x})$ in which we can apply our learning algorithm to find the maximal margin hyperplane. More often than not, this feature space is of (much) higher dimension than the input space. Perhaps the most illustrative example is the problem of finding a way to linearly separate black dots within a circle from white dots outside that circle (see Fig. 6).

The third step is probably the most important step. Before we can apply the pertinent 'kernel trick', we must realize that the margin optimization problem can be represented in it's Lagrangian dual representation as follows:

$$\begin{array}{cc} maximize_{\alpha \in \Re^m} & W(\boldsymbol{\alpha}) = \sum_{i=1}^m \alpha_i - \sum_{i=1}^m \alpha_i \alpha_j y_i y_j \langle \boldsymbol{x}_i \cdot \boldsymbol{x}_j \rangle \\ subject\,to & \sum_{i=1}^m \alpha_i y_i = 0 \\ and & \alpha_i \geq 0,\, i = 1, \ldots, l \end{array} \tag{16}$$

The decision function of our classifier is now found as:

$$f(\boldsymbol{x}) = sgn\left(\sum_{i=1}^{m} \alpha_i y_i \langle \boldsymbol{x} \cdot \boldsymbol{x}_i \rangle + b\right) \tag{17}$$

where b is the bias of the hyperplane and $\langle \boldsymbol{x} \cdot \boldsymbol{x}_i \rangle$ is the inner product of test sample $\boldsymbol{x}$ and the ith trainsample $\boldsymbol{x}_i$. This representation has the remarkable property that the data only appear inside an inner product. Maximizing the margin and evaluating the decision function both require the computation of the dot product $\langle \Phi(\boldsymbol{x}), \Phi(\boldsymbol{x}_i) \rangle$ in a high-dimensional space. There exist special functions, or kernels, with the following property:

$$\langle \Phi(\boldsymbol{x}), \Phi(\boldsymbol{x}_i) \rangle = K(\boldsymbol{x}, \boldsymbol{x}_i) \tag{18}$$

Substituting these so-called Mercer kernels into inner products of the objective and the decision functions (16) and (17) can greatly reduce the computation time, as it eliminates the need to explicitly transform the input space to the feature space. The patterns which we want to detect using our maximal margin classifier do not need to coincide with the input $\boldsymbol{x}$, we might as well apply our decision function (17) directly on $\Phi(x)$. Substituting (18) for the inner product, the decision function in feature space directly becomes

$$f(\boldsymbol{x}) = sgn\left(\sum_{i=1}^{m} y_i \alpha_i k(\boldsymbol{x}, \boldsymbol{x}_i) + b\right) \tag{19}$$

The sparseness of SVMs results from the fact that only a small fraction of the α_i coefficients have nonzero values. The name of the SVM comes from those vectors $\boldsymbol{x}_i$ with nonzero α_i which are known as the support vectors. This sparse set of support vectors fully define the decision function, giving rise to extremely fast classification and little space needed to store the decision function.

Unfortunately, in many practical situations, a separating hyperplane does not exist. To allow for possibilities of violating (17), slack variables $\zeta_i \geq 0$ are introduced. The optimization problem is now defined as:

$$\begin{array}{cc} minimize & \tau(\boldsymbol{w}, \zeta) = \langle \boldsymbol{w}, \boldsymbol{w} \rangle + C \sum_{i=1}^{l} \zeta_i \\ subject\,to & y_i(\langle \boldsymbol{x}_i \cdot \boldsymbol{w} \rangle + b) \geq 1 - \zeta_i,\ i = 1, \ldots, l \\ and & \zeta_i \geq 0,\ i = 1, \ldots, l \end{array} \tag{20}$$

It can be shown that minimizing the first term in (20) amounts to minimizing a bound on the VC-dimension and minimizing the second term corresponds to minimizing the classification error [24]. This minimization can be posed as a constrained quadratic programming problem, with solutions in the form of (17).

3.4.1 Probabilistic Active Learning

Because of the large amount of samples used in our experiments (over 15.000), validation with a leave-one-out scheme or even a leave-one-session-out scheme, where one session contains samples belonging to one video, becomes an intractable problem. To overcome this problem, we implemented a Probabilistic Active Learning (PAL) algorithm [27]. PAL iteratively improves the SV set, using only a small subset of the training samples at every iteration to build the support vector classifier. The algorithm estimates the likelihood that a new example belongs to the actual support vector set and selects a set of p new points according to this likelihood, which are then used along with the current set of SVs to obtain the new SV set. The likelihood of an example being a SV is estimated using a combination of two factors: the margin of the particular example with respect to the current hyperplane and the degree of confidence that the current set of SVs spans the actual hyperplane that separates the complete training set in feature space best[3]. This confidence factor c, which varies with each iteration, can be also seen as a measure for how close the current hyperplane is to the actual hyperplane. Therefore c can be used as an indication of how to choose samples to be used in the next iteration: close to the current hyperplane (high c) or far away from the hyperplane (low c). So instead of being randomly chosen from the set of training examples, the new set of samples for each iteration is generated according to a probability $P_{\xi(x,f(x))}$ where $\xi(x, f(x))$ denotes the event that example x is an SV. If $\langle \boldsymbol{w}, b \rangle$ is the current separating hyperplane, we have:

$$P_{\xi(x,f(x))} = \begin{cases} c & if\, y\left(\langle \boldsymbol{w} \cdot \boldsymbol{x} \rangle + b\right) \leq 1 \\ 1-c & otherwise \end{cases} \tag{21}$$

Here c is the above mentioned confidence factor. This factor is estimated as follows. Let the current set of SVs be denoted by $S = \{s_1, s_2, ..., s_l\}$. Also, consider an integer k (say, $k = \sqrt{l}$). For every $\boldsymbol{s}_i \in S$, compute the set of k nearest points in the train set $\boldsymbol{x}$. Among the k nearest neighbors, let k_i^+ and k_i^- number of points have labels +1 respectively -1. The confidence factor c is then defined as:

$$c = \frac{2}{lk} \sum_{i=1}^{l} min\left(k_i^+, k_i^-\right) \tag{22}$$

This results in an adaptive algorithm that starts by finding the general location of the separating hyperplane and then proceeds with fine tuning in order to delimit the exact location of $\boldsymbol{w}$.

4 Experimental Evaluation

In this section we will present various experiments that were undertaken to evaluate the performance of AAU detectors based upon of the two data representations described in section 2 in combination with the classifiers described

[3]Note that we use the confidence in having found the actual hyperplane, as opposed to Mitra et al. [27], who find the confidence that the current set of SVs is the actual set of SVs. The issue here is that the set of SVs that spans the actual hyperplane does not need to be unique.

in section 3. We will organize the remainder of this section as follows: section 4.1 describes the two data sets that we have used for our experiments. Section 4.2 describes the experimental setup for the experimental evaluation of AU detection using temporal templates, while section 4.3 describes the experiments conducted based upon tracked fiducial facial points. Some combinations of data representations and classifiers seemed a priori unfruitful and have not been evaluated.

4.1 Face Databases

To develop and evaluate face analysis applications, large collections of training and test data are needed. While motion records are necessary for studying temporal dynamics of facial expressions, static images are important for obtaining information on the configuration of facial expressions which is essential, in turn, for inferring the related meaning (e.g., in terms of emotions). Therefore both static face images and face videos are needed.

Also, while the researchers of machine analysis of facial affect are interested in facial expressions of emotions, the researchers of machine analysis of facial muscle actions are interested in facial expressions produced by activating a single facial muscle (AU) or by activating a combination of AUs. Therefore both kinds of training and test material are needed.

In a frontal-view face image, facial actions such as showing the tongue or pushing the jaw forwards represent out-of-plane non-rigid movements which are difficult to detect. Such AUs are clearly observable in a profile view of the face. On the other hand, changes in the appearance of the eyes and eyebrows are clearly observable from a frontal facial view. Therefore both frontal and profile facial views are necessary for the research on machine analysis of facial expressions.

In spite of repeated references to the need for a comprehensive, readily accessible reference set of face images that could provide a basis for benchmarks for all different efforts in the research on machine analysis of facial expressions, no database of images exists that is shared by all diverse facial-expression-research communities. In general, only isolated pieces of such a facial database exist. An example is the unpublished database of Ekman-Hager Facial Action Exemplars. It has been used by several research groups in the States to train and test their methods for facial action detection from face image sequences. The facial databases made publicly available, but still not used by all diverse facial-expression-research communities, are the Cohn-Kanade AU-coded Face Expression Image Database [28], the PIE face database [29], the AR database [30], and the JAFFE database [31]. From these, the most comprehensive and the most commonly used database in the research on automated facial expression analysis is the Cohn-Kanade face database.

None of these existing face databases contains images of faces in profile view, none contains images of all possible single-AU activations, and none contains both static face images and face videos. Also, the metadata (labels) associated with each database object does not usually identify the temporal segments (on-

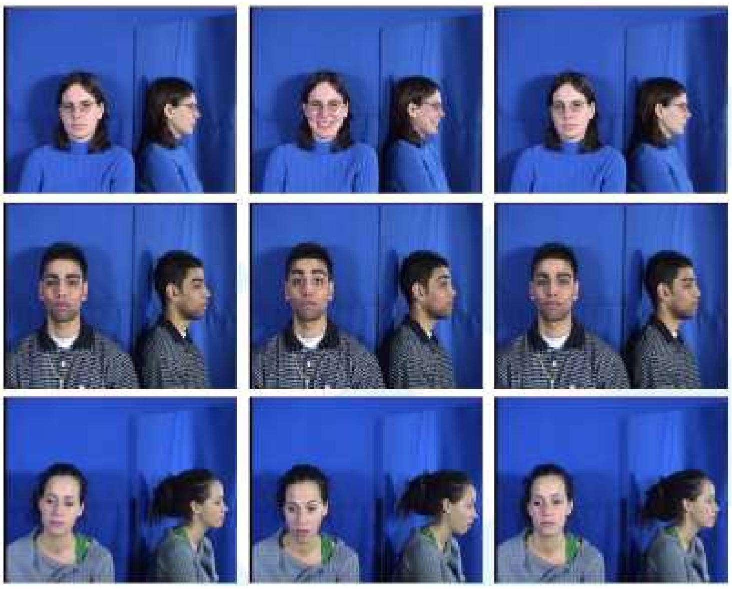


Figure 7: Three samples of recordings (sessions) contained in the MMI-Face DB. The three columns show the first, middle and last frames of each session

set, apex, offset) of AUs and emotion facial displays shown in the face video in question. Finally, none of these databases is either easily accessible or easily searchable. Once a permision for the usage is issued, large, unstructured files of material are sent.

MMI-face-Database In an attempt to address the issues on face databases presented above, we started the development of the MMI-face-DB. It presently consists of over 4000 videos and 600 static images depicting facial displays of 31 adults being 18 to 35 years old; 50% female, 81% being Caucasian, 14% Asian and 5% African. All facial displays are posed and the recordings are made under constant lighting conditions. All but seven facial expression videos were recorded in profile and frontal view simultaneously (using a mirror). Two FACS experts coded the database. When in doubt, decisions were made by consensus.

The database contains a large amount of videos where the activation of individual AUs has been recorded. In the cases where this was not possible, expressions produced by the activation of the least possible number of AUs were recorded. For example, AU16 (lower lip depressed) depresses the lower lip and automatically parts the lips causing AU25 (lips parted) to be activated as well. Figure 7 shows the first, middle and last frames of three recordings (sessions) of the MMI-Face-DB.

Cohn Kanade Database Another database that we used in our experiments is the Cohn-Kanade face database. This database contains over 2000 videos of the facial displays produced by 210 adults being 18 to 50 years old, 69% female, 81% caucasian, 13% African and 6% from other ethnic groups. All facial displays are posed and the recordings are made under constant lighting conditions. Only real expressions were recorded, which means that many AUs never occur alone. Many recordings contain the date/time stamp recorded over part of the face. This occurrence is unwanted, for it causes (M)MHI activation and it foils the point tracking algorithm. Hence, as explained below, when tested on Cohn-Kanade database samples, our algorithms achieved unsatisfactory results in some instances.

4.2 Temporal Templates experiments

For detection of AUs from MHI- and MMHI-represented face image sequences, we compared two classification schemes: (i) a two-stage classifier combining a kNN-based and a rule-based classifier, and (ii) a SNoW classifier. The first method is straightforward, while the second method seemed very appropriate to the problem at hand. SNoW is a sparse multi-class classifier specifically tailored to large-scale classification problems with a very large number of features, which is exactly what we needed.

4.2.1 Combined kNN/rule-based classifier

We have developed a two-stage classifier, consisting of a kNN classifier for the initial classification, followed by a second-stage rule-based classifier, which tries to correct a number of common mistakes made in the first stage. The kNN variable k and the distance metric were experimentally determined. Best results for MHI input data were achieved using the simple Euclidian, or L_2, norm (12). In the case of MMHI-based data representation we have used the Chamfer distance (14).

Though it gives a good indication of the AUs shown in a given sample, the kNN algorithm confuses commonly AUs that have partially the same (M)MHI. To address this drawback, we created a set of rules based on the knowledge of a human FACS coder. We defined facial regions in which the presence of motion characterizes certain AU activation (see Fig. 8). For example, the presence of motion in region R_2 is characteristic for the activation of AU2. We calculate this activity $act\,(R_i)$ in facial region R_i as follows:

$$act\,(R_i) = \begin{cases} \frac{1}{N}\sum_{\boldsymbol{x},\boldsymbol{y}\in R_i}\left\lceil\frac{H(\boldsymbol{x},\boldsymbol{y},n_l)}{255}\right\rceil & MHI-data \\ \frac{1}{N}\sum_{j\in R_i}\left\lceil\frac{|A_j|}{n_l}\right\rceil & MMHI-data \end{cases} \tag{23}$$

where H is the MHI operator defined in (3), n_l is the number of history levels in each (M)MHI, $|A_j|$ the cardinality of active history levels of the j-th pixel in a MMHI and N the number of pixels in the facial region R_i. The facial regions are positioned relative to the same facial points that we used for the registration

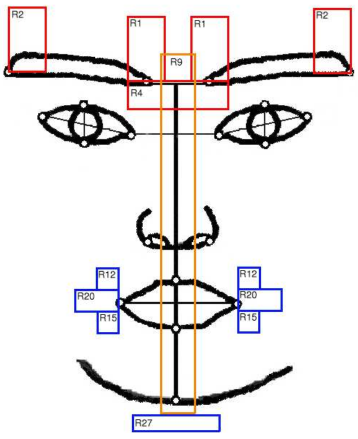


Figure 8: Facial regions for meassurement of temporal templates activity. The numbers of the regions correspond to Action Units

of image sequences as described in section 2.1.1. Using the activation of these regions we built a set of rules. With these rules it is possible to correctly reclassify samples that the kNN algorithm misclassified at first. For example, the kNN classifier often confuses AU4 and AU1+AU4. Both produce activity in the same part of the (M)MHI (in regions R_1 and R_4 as illustrated in Fig. 8), but AU4 causes the eyebrows to move inward and downward, while AU1+AU4 first causes an inward and downward movement of the eyebrows followed by an upward movement of the brows. This results in a high activation between the brows and a relatively low activation above the inner corners of the brows. Hence, the rule used to resolve the confusion in question is defined as follows. If the kNN classifier encodes AU4 and it is true that

$$\frac{act\left(R_1\right)}{\left(act\left(R_2\right)+act\left(R_4\right)\right)} > th_3 \bigwedge act\left(R_4\right) > th_4 \tag{24}$$

where th_j are thresholds that are automatically defined during the training phase, then AU1+AU4 will be the final classification of the pertinent input sample.

Experimental evaluation showed that the optimal kNN parameter k should be $k = 3$ and the (M)MHI constructor threshold $th_I = 0.19$ (equation (1)).

Table 4 shows the results of tests performed on a leave-one-out basis on the MMI-face-DB and table shows the results for the same test performed on the Cohn-Kanade database. Unfortunately, as mentioned in section 4.1, the Cohn-Kanade database has fewer AUs occurring alone, resulting in fewer AUs that can be recognized. The recognition rates for this database are somewhat higher. This is probably due to a larger number of samples per AU and fewer target classes providing less confusion possibilities.

Results for MMHI are, overall, lower than they are for MHI. This is because of the definition of MMHI. The distance measure used makes it difficult to find

Table 3: SNoW detection results. Column 2 lists the number of positive samples in the dataset

Action Unit	nr	Recognition rate positive samples	Recognition rate negative samples
1	34	0.85	0.85
2	22	0.64	0.97
4	37	0.76	0.85
6	18	0.50	0.89
12	21	0.67	0.94
17	36	0.33	0.85
25	49	0.94	0.23
26	4	0.25	0.94
27	7	0.29	1.00
total:	228	0.58	0.84

the desired nearest neighbor of a sample when multiple levels are activated on the same position in a MMHI. Furthermore, as we do not have any examples displaying possible confusion caused by motion self-occlusion, we were not able to show the increased resolution of MMHI with respect to MHI in occasions where confusions caused by motion self-occlusions do occur.

4.2.2 SNoW classifier

For each AU to detect a binary SNoW net is trained using an equal number of positive and negative samples. Each trained SNoW net is evaluated on the whole dataset on a leave-one-out basis. The MHI data was constructed from the Cohn-Kanade database; samples containing individual AUs as well as samples containing more than one AU were allowed in the training and test sets. The number of examples the SNoW algorithm requires to learn a linear function grows linearly with the number of relevant features. Yet, although we down-scaled our images, the results suggest that there still are too many relevant features present in our sample MHIs for the number of samples in the dataset; AUs with a small number of positive samples do not have good recognition rates for the detection of positively labeled samples. Table 3 shows the results for the SNoW classifier.

4.3 Facial points experiments

For detection of AUs based upon tracked facial points we compared the results of classification achieved with PAL-trained SVMs to those achieved using kNN. For the evaluation of both detectors we tracked 20 facial micro features (described in section 2.2.3) in 167 image sequences from the MMI-Face-DB displaying different facial expressions. We evaluated two different approaches. In the first approach we used expert knowledge to define 2-5 different parameters for detection of

Table 4: Recognition rates for the MMI-face-DB using a two-stage kNN-rulebase on temporal template data. The second column shows the number of samples of the specified action unit contained in the database

Action Units	nr	MHI		MMHI	
		Recall	Precision	Recall	Precision
1+2	10	0.9	0.75	0.9	0.75
2	6	0.5	1.0	0.5	0.6
1+4	6	0.5	0.5	0.5	0.5
4	12	0.67	0.57	0.75	0.47
6	10	0.7	0.88	0.7	0.88
9	11	0.82	1	0.45	1
8+25	10	0.6	0.6	0.4	0.57
10+25	10	0.9	0.9	0.8	0.89
11+25	10	0.7	0.88	0.7	0.88
12+25	10	1.0	0.5	0.8	0.47
14	11	0.27	0.43	0.18	0.4
15	8	0.37	0.25	0.75	0.18
16+25	10	0.6	0.43	0.5	0.24
17	10	0.6	0.67	0.7	0.54
18	10	0.7	0.64	0.6	0.55
20	10	0.6	0.67	0.6	0.86
22+25	10	0.6	0.67	0.4	1
25	9	0.33	0.23	0.33	0.38
25+26	11	0.36	0.27	0.46	0.31
27	10	0.8	1	0.7	1
26+30L	9	0.33	0.75	0.33	0.75
26+30R	9	0.67	0.67	0.67	0.75
26+36T	12	0.50	0.86	0.42	0.71
26+36B	10	0.60	0.75	0.3	0.75
26+36L	9	0.56	0.45	0.44	0.44
26+36R	10	0.40	0.4	0.6	0.75
Total:	253	0.61	0.59	0.56	0.54
Recognitionrate:		0.61		0.56	

each of 15 different AUs. Every classifier is thus trained with a different set of features and the activation of AUs is highly independent of each other. In contrast, the second approach tries to exploit the correlation between various AUs by using all (48) features. These are the same features that were used in the first approach, only now used all together. This approach makes it possible to detect AUs indirectly that have high correlation with other AUs. Using this approach we can detect AUs 6 and 9 in addition to the 15 AUs we were allready able to detect.

Table 5: Recognition rates for the Cohn-Kanade database using a two-stage kNN-rulebase on temporal template data. The second column shows the number of samples of the specified action unit contained in the database

Action Units	nr	MHI		MMHI	
		Recall	Precision	Recall	Precision
1+2	21	0.52	0.85	0.45	0.75
1+4	21	0.61	0.52	0.67	0.39
4	21	0.38	0.44	0.29	0.46
6	61	1	0.88	0.89	0.87
12+25	59	0.63	0.79	0.31	0.82
12	27	0.78	0.45	0.54	0.32
15	4	0.25	0.09	0	0
17	26	0.54	0.74	0.46	0.43
20+25	18	0.11	0.67	0.11	0.29
25	42	0.64	0.73	0.88	0.5
27	42	0.93	0.89	0.78	0.92
Total:	344	0.68	0.68	0.58	0.59
Recognition rate:		0.69		0.58	

4.3.1 SVM validation

Training data was generated from 169 videos (from now on called 'sessions') from the MMI-face-DB (15 different subjects). The training data consisted not only of the 17 AUs to be detected, but of all possible AUs. This way we could prove that our system will work in real-life situations where people can display any facial expression, allthough we still are not able to recognize them all. Validation was performed using a leave-one-session-out cross-validation strategy. This seems fair, as each session consists of an image sequence containing one facial expression. Therefore, every session has a highly spatio-temporally correlated pattern. A leave-one-out strategy where we would leave one frame out instead would lead to enormous loss of generalization, because the frame is embedded in the time dynamic pattern of all the other frames within that video and classifcation of that frame would become almost trivial.

The SVM classifier detects AUs per frame. In this work we will not consider timing aspects of AU activation but instead focus on the presence of AUs within an entire video, consistent with existing literature. Therefore we add a decision layer that adaptively computes a threshold, favoring the recall over the precision of the classification of AUs in facial video. First, we apply the cross-validation scheme on our data set. This results in a set of AU activation predictions for every frame in every facial video. Suppose the SVM detemined that a test sample $\boldsymbol{x}$ has n frames where a certain AU is active. Let's say that $\boldsymbol{N}_p$ is a vector containing the number of active frames as predicted by the SVM for all videos where the AU under investigation is present and $\boldsymbol{N}_n$ the vector containing the number of active frames as for all videos where that AU is not present. Let

Table 6: Validation results for AU detection using SVM on selected features. Columns two and three list the number of positive/negative image sequences per AU.

AU	Truth	Prediction	Cl. Rate	Recall	Precision
1	13/154	13/154	1.00	1.00	1.00
2	10/157	10/157	1.00	1.00	1.00
4	24/143	12/155	0.87	0.50	0.50
10	15/152	0/167	0.91	0.00	0.00
12	11/152	8/155	0.98	0.73	0.89
13	10/157	10/153	0.98	1.00	0.71
15	10/157	0/157	0.94	0	0
16	8/66	2/66	0.92	0.25	1.00
18	13/154	9/148	0.94	0.69	0.60
20	10/157	6/140	0.87	0.6	0.26
22	8/159	2/152	0.92	0.25	0.22
25	75/92	71/76	0.88	0.95	0.82
26	17/150	12/149	0.96	0.71	0.92
27	10/157	10/157	1.00	1.00	1.00
30	5/108	4/79	0.73	0.8	0.12

$\boldsymbol{N}_{n<p}$ be the vector of elements of $\boldsymbol{N}_n$ that are smaller than $min\left(\boldsymbol{N}_p\right)$. The threshold that is used to decide wether the test sample $\boldsymbol{x}$ contains the AU under investigation is now found as:

$$th = max\left(\boldsymbol{N}_{n<p}\right) + \frac{\left(min\left(\boldsymbol{N}_p\right) - max\left(\boldsymbol{N}_{n<p}\right)\right)}{2} \tag{25}$$

The results presented for the various validation studies on the facial point data are organized in tables as follows: the first three columns of the tables represent the AU, the number of positive/negative samples for that AU as recorded by the FACS coder (the ground truth for our validation studies) and the number of correctly predicted positive/negative samples. Columns four to six list the classification rate, recall and precision for the AU. The classification rate is the number of correctly classified samples divided by the total number of samples. The recall is defined as the number of positive samples from the ground truth that are correctly classified divided by the total number of positive samples found in the ground truth. Precision is defined as the correctly classified positive samples divided by the sum of correctly classified positive samples and the number of false positives. So, recall tells us how often the classifier retrieves a positive example, while the precision is a measure of how reliable the results are.

Table 6 shows the results for the cross-validation on the MMI-Face-DB using the selected features for each AU.

Table 7 shows the results for the cross-validation on the MMI-Face-DB using all features for a selection of AUs. As we can see, using all features enables

Table 7: Validation results for AU detection using SVM on all features. Columns two and three list the number of positive/negative image sequences per AU.

AU	Truth	Prediction	Cl. Rate	Recall	Precision
1	13/154	11/122	0.80	0.8462	0.2558
6	16/151	15/129	0.86	0.94	0.41
9	10/157	10/156	0.99	1.00	0.91
10	15/152	14/142	0.93	0.93	0.58
26	17/150	16/151	0.83	0.94	0.37
27	10/157	7/134	0.84	0.70	0.23

Table 8: kNN validation of the MMI-Face-DB for ><DEFANGED.2909 three selected AUs

AU	Truth	Prediction	Cl. Rate	Recall	Precision
1	13/154	13/153	0.99	1	0.93
25	75/92	73/13	0.52	0.97	0.48
16	10/150	12/98	0.66	0.71	0.19

us to detect AUs we could not detect using the selected features. However, performance drops for some other AUs. Most notably, the classifier performs pretty bad for AU1 and AU27, in shrill contrast with the performance of the SVM-PAL classifier using selected features.

4.3.2 kNN validation

To compare our SVM-PAL classification scheme with the cumbersome kNN classifier, we applied leave-one-session-out cross validation on the MMI-Face-DB data for three AUs. We used all Facial Fiducial Point features for this validation. The first AU, AU1, is very easy to detect using the fiducial facial point data representation, the second AU, AU25, is a good representation of an AU of 'average recognition difficulty' and the last AU we performed the comparison study on is AU16, which was extremely difficult to detect using our SVM-PAL classifier. As we can see from table 8, kNN shows similar results. The precision of kNN is far lower than the precision found for SVM-PAL. Therefore, the results from the SVM-PAL are more reliable.

5 Conclusions and Future Research

The purpose of this work was a twofold. The first goal was to determine the effectiveness of the Temporal Template and Fiducial Facial Points data representations for the purposes of AU recognition. The second goal was to evaluate on classification schemes which would be effective for the AU recognition tasks, using the aforementioned data representations.

We have shown that the Temporal Template data representation is suitable for AU recognition tasks, altough the configuration in which we used Temporal Templates in this work is open for improvement. The nature of Temporal Templates makes it very useful for a temporal analysis of AU activation, a topic we have not addressed in this work. Onset/offset detection would be an area we wish to investigate further using Temporal Templates. Data representation for temporal analysis of facial expression dynamics, we believe Multilevel Motion History Images (MMHIs) are more appropriate than the original Motion History Images (MHIs). MMHIs retain all motion information of the temporal segment for which they are defined, in contrast with MHIs that can lose motion information by motion self occlusion. Unfortunately, we failed to prove this in this work. We believe this is due to two facts. First, in our test data there were no cases where motion self-occlusion could cause confusion (e.g., in hand gesture recognition, waving the hand would be confused with moving the hand from left to right only due to motion self occlusion). The second problem when using MMHIs is choosing an appropriate classification mechanism. The kNN classifier using a Chamfer distance metric is not a good classifier for this data representation.

The results in section 4.3 show clearly that the Fiducial Facial Points data representation is very well suited for the task of AU detection and thus for the task of facial expression analysis. If we select for every AU the proper feature parameters, results are both precise and reliable. The fact that it is possible to detect AUs on a per frame basis makes this data representation also suitable for temporal analysis dynamics of facial expressions. It may seem strange at first that the SVM-PAL results using all features are sometimes lower than the results for the same classification scheme using selected features. The reason for this is that when we look at all the features, the activation of AUs becomes correlated. In order to train a system using all features that will generalize well in real world applications, we would need a very complete database of facial expressions that covers all possible combinations of AUs. This is not feasible in practice. We therefore suggest that some care has to be taken when using all features and that where possible the selected features should be used.

We have shown that Support Vector Machines are a good classifier in combination with Fiducial Facial Point data for the task of AU detection. It clearly outperforms kNN in terms of generalisation capability, classification time and required memory. Sparse Networks of Winnows (SNoWs) are not usefull in the current setting. They either need more training data, or we need to apply feature selection before training ><DEFANGED.2910 the SNoW.

In the near future we plan to do more experiments. We plan to do a large-scale comparison between the MMI-Face-DB and the Cohn-Kanade-DB for the SVM-PAL classifier based upon the facial point data. Also, we want to test the SVM-PAL classifier using all feature parameters for more AUs.

As allready mentioned above, both data representations have potential to prove useful for temporal analysis of facial expression dynamics. The classification and interpretation of facial expressions in terms of onset, apex and offset of AUs is one of the research topics we will be addressing in the future. For this temporal analysis, Hidden Markov Models is one of the learning algorithms we want to investigate further. Temporal analyses on larger time scales, that is, interpretations of facial behavior in terms of attitudes like bordom and attentiveness as well as in terms of mood is one of our future aims as well.

The ultimate goal is providing means for better man-machine interaction. Facial expression analysis is an important variable in this process. Yet, we believe that we can improve the understanding between humans and machines even more by including novel modalities for non-verbal communication. The fusion of video and audio data will therefore receive a lot of research efforts in the near future.

References

[1] M. Lewis and J.M. Haviland-Jones, *Handbook of Emotions*, New York, Guilford Press, 2000

[2] M. Pantic, “Face for interface: survey and critical issues”, The Encyclopedia of Multimedia Technology and Networking, M. Pagani (ed.), Idea Group Publishing, 2005, accepted for publication

[3] K.R. Scherer, and P. Ekman, Eds. (1982). *Handbook of methods in nonverbal behavior research.* Cambridge, Cambridge University Press, 1982

[4] P. Ekman, W.V. Friesen, *The Facial Action Coding System: A Technique for the Measurement of Facial Movement*, San Francisco: Consulting Psychologist, 1976

[5] P. Ekman, W.V. Friesen, J.C. Hager, *Facial Action Coding System (FACS)*, Weidenfeld & Nicolson, London, 2002

[6] K. Mase, “Recognition of facial expression from optical flow”, *IEICE Trans.*, vol. E74, no. 10, pp. 3474-3484, 1991

[7] M.J. Black and Y. Yacoob, “Recognizing facial expressions in image sequences using local parameterized models of the image motion”, *Int. J. Comput. Vis.* vol. 25, no. 1, pp. 23-48, 1997

[8] I. Essa and A. Pentland, “Coding analysis interpretation recognition of facial expressions”, *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, pp. 757-763, July 1997

[9] M.S. Bartlett, J.C. Hager, P. Ekman, and T.J. Sejnowski, “Measuring facial expressions by computer image analysis”, *Psychophysiology*, 36, 253-263, 1999

[10] G. Donato, M.S. Bartlett, J.C. Hager, P. Ekman, and T.J. Sejnowski, “Classifying Facial Actions”, *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 21 no. 10, pp. 974-989, 1999

[11] J.F. Cohn, A.J. Zlochower, J. Lien, and T. Kanade, "Automated face analysis by feature point tracking has high concurrent validity with manual faces coding", *Psychophysiology*, vol. 36, pp. 35-43, 1999

[12] Y. Tian, T. Kanade and J.F. Cohn, "Recognizing action units for facial expression analysis" *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 23, no. 2, pp. 97-115, 2001

[13] B. Braathen, M.S. Bartlett, G. Littlewort, E. Smith and J.R. Movellan, "An Approach to Automatic Recognition of Spontaneous Facial Actions", *Proc. IEEE Int'l Conf. Face and Gesture Recognition*, pp 345-350, 2002

[14] M. Pantic, and I. Patras, "Temporal modeling of facial actions from face profile image sequences", *Proc. Int'l Conf. Multimedia and Expo*, 2004

[15] M. Pantic and L.J.M. Rothkrantz, "Facial Action Recognition for Facial Expression Analysis from Static Face Images", *IEEE ><DEFANGED.2911 Trans. Systems, Man, and Cybernetics*, Part B, vol. 34, no. 3, pp. 1449-1461, 2004

[16] J. Xiao and T. Moriyama and T. Kanade and J.F. Cohn. "Robust Full Motion Recovery of Head by Dynamic Templates and Re-registration Techniques" *Int'l Journal of Imaging Systems and Technology*, vol. 13,pp. 85-84, 2003.

[17] I. Essa, A. Pentland, "Coding, analysis, interpretation and recognition of facial expressions" *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19(7),pp. 757-763, 1997

[18] A.F. Bobick and J.W. Davis, "The Recognition of Human Movement using Temporal Templates", *IEEE trans. Pattern Analysis and Machine Intelligence*, vol 23, pp. 257-267, 2001

[19] M. Isard and A. Blake, "Condensation - Conditional Density Propagation for Visual Trackint", *Int'l. J. Computer Vision*, pp. 5-28, 1998

[20] I. Patras and M. Pantic, "Particle filtering with factorized likelihoods for tracking facial features", *Proc. IEEE Int's Conf. on Automatic Face and Gesture Recognition*, pp. 97-102, 2004

[21] M.K. Pitt and N. Shephard. Filtering via simulation: auxiliary particle filtering. *J. American Statistical Association,* vol. 94, nr. 446, pp. 590-599, 1999

[22] I. Patras and M. Pantic, "Tracking deformable motion", *unpublished*, 2004

[23] M.H. Yang, D. Roth and N. Ahuja, "A SNoW-Based Face Detector", *Advances in Neural Information Processing Systems*, vol. 12, pp. 855-861, 2000

[24] V.N. Vapnik, *The nature of statistical learning theory*, New York, Springer-Verlag, 2nd edition, 1999

[25] B. Scholkopf and A.J. Smola, *Learning with Kernels*, Cambridge Massachusetts, London England, MIT press, 1999

[26] N. Cristianini, J. Shawe-Taylor, *Support Vector Machines*, Cambridge, England, Cambridge University Press 2000

[27] P. Mitra, C.A. Murthy, S.K. Pal, "A probabilistic active learning support vector learning algorithm", *IEEE trans. on Pattern Analysis and Machine Intelligence*, pp. 413-418 2004

[28] T. Kanade, J. Cohn and Y. Tian, "Comprehensive database for facial expression analysis", *Proc. IEEE Int.'l Conf. Face and Gesture Recognition*, pp. 46-53, 2000

[29] R. Gross, J. Shi, "The CMU Motion of Body (MoBo) Database", *Tech. report CMU-RI-TR-01-18,* Robotics Institute, Carnegie Mellon University, June, 2001

[30] A.M Martínez and R. Banavente, "The AR face database", *CVC Tech. Report #24*, June 1998.

[31] M.J. Lyons and J. Budynek and S. Akamatsu, "Automatic Classification of Single Facial Images", *IEEE trans. on Pattern Analysis and Machine Intelligence*, vol. 21(12), pp. 1357-1362, 1999.

[32] L. Maat, R. Sondak and P. Gaja, "MMI Face Database", *technical report MMI-BS-2004-02*, Delft, 2004