Computers & Graphics
Vol. 22, No. 3
to appear 1998.

Digital Watermarking of MPEG-4 Facial Animation Parameters

Frank Hartung, Peter Eisert, and Bernd Girod

Telecommunications Institute I
University of Erlangen-Nuremberg
Cauerstrasse 7, 91058 Erlangen, Germany
Phone +49 9131 85 7116
Fax +49 9131 85 8849
{*hartung,eisert,girod*}*@nt.e-technik.uni-erlangen.de*

The enforcement of intellectual property rights is difficult for digital data. One possibility to support the enforcement is the embedding of digital watermarks containing information about copyright owner and/or receiver of the data. Watermarking methods have already been presented for audio, images, video, and polygonal 3D models. In this paper, we present a method for digital watermarking of MPEG-4 facial animation parameter data sets. The watermarks are additively embedded into the parameter values and can be retrieved either from the watermarked parameters or from video sequences rendered using the watermarked animation parameters for head animation. For retrieval from rendered sequences, the facial animation parameters have to be estimated first. We use a model-based approach for the estimation of the facial parameters that combines a motion model of an explicit 3D textured wireframe with the optical flow constraint from the video data. This leads to a linear algorithm that is robustly solved in a hierarchical framework with low computational complexity. Experimental results confirm the applicability of the presented watermarking technique.

Key words: digital watermarking, facial animation, facial animation parameters, 3D head model, computer graphics, model-based video coding, MPEG-4

1 Introduction

3D models have extensively been used for the synthesis of naturally looking images and video. For rendering of animated video sequences using 3D models, usually 3 separate data sets are utilized: a data set describing the basic shape of the 3D object (e.g. a wireframe or a spline surface), a data set describing the surface brightness and color (the texture), and a data set describing the temporal variation of the model (the animation parameters). For the special case of animated head-and-shoulder sequences, the International Standardization Organization (ISO) has standardized a format for the representation and animation of such 3D models [1].

Each of the three mentioned data sets (shape, texture, animation parameters) can be regarded as an own entity where own associated intellectual property rights may exist. For example, a 3D wireframe model may be generated from a 3D laser scan, and an according texture pattern may be painted by an artist. Later on, different animation parameter sets may be added to the model in order to animate different sequences with the model. In this case, the intellectual property rights for shape, texture and motion of the model may certainly belong to different parties, and need to be protected separately.

As with all other digital multimedia data, copy protection is difficult to achieve for 3D model and animation data. Every copy of digital data has the same fidelity as the original. Thus, mechanisms are desirable to enforce intellectual property rights. One possible solution to the problem is digital watermarking [2]. Digital watermarking allows to robustly embed copyright information, as well as information about the receiver of data, into digital data. It does not prevent copying, but can help to identify and trace back illegal copies of multimedia data. A large variety of watermarking methods for formatted text [3], images [4–7], video [8–11] and audio [12] have already been presented. Since textures of 3D models are often represented as a 2D image, most of the presented methods for images are also applicable to watermarking of texture images. A few contributions have also been made for watermarking of other data, for example for static 3D polygonal models [13]. No contributions have however been made that suggest solutions for the embedding of data into model animation parameters.

In this paper, a method is proposed that allows to embed digital watermarks into MPEG-4 facial animation parameters (in the following called FAPs). The method derives from ideas of spread spectrum communications. The watermark can either be retrieved directly from the watermarked parameters, or from the video sequence rendered from the model using the watermarked parameters. In the latter case, the animation parameters have to be estimated from the sequence, and the watermark is then retrieved from the estimated parameters.

The proposed method can be used to embed information into MPEG-4 FAPs. For example, a FAP sequence that is available for download on the WWW can be marked with invisible copyright information and/or with information about the downloading client. This information can later on be retrieved either from the watermarked FAPs, or from a video sequence rendered from them.

This paper is organized as follows: in Section 2, the concept of MPEG-4 head animation is explained. In Section 3, a method is explained that allows the estimation of FAPs, e.g. MPEG-4 FAPs, from natural or synthesized video sequences. In Section 4, the proposed method for watermarking of FAPs is described. In subsection 4.1, the embedding of the watermark is shown. Subsections 4.2 and 4.3 describe the retrieval of the watermarks from the parameters and the rendered video sequences, respectively. In Section 5, the applicability of the scheme is demonstrated by experimental results.

2 MPEG-4 Face Animation

The video compression standards MPEG-1 (ISO/IEC 11172) and MPEG-2 (ISO/ IEC 13818) are widely known and used for video storage and distribution. The new standard MPEG-4 (ISO/IEC 14496) [1] does also supply tools for video and still image compression, but has additional functionalities and features beyond waveform-based compression, such as mechanisms for face animation. The tool of MPEG-4 face animation allows the compression of head-and-shoulder scenes down to bit-rates of very few kbit/s. This is beyond today's possibilities of waveform-based compression.

A generic face is predefined in MPEG-4 and can be animated with facial animation parameters (FAPs). Additionally, a particular face can be transmitted if desired using so-called facial definition parameters (FDPs).

A set of 66 FAPs is defined in MPEG-4, Annex C [1]. It includes global head motion parameters (e.g. head pitch and yaw angles) and local face motion parameters (e.g. opening of eyelids, opening of lips, movement of innerlip corners). In total, there are 16 FAPs controlling the jaw, chin, inner lips and cornerlips; 12 FAPs controlling the eyeballs, pupils and eyelids; 8 FAPs controlling the eyebrows; 4 FAPs controlling the cheeks; 5 FAPs controlling the tongue; 3 FAPs controlling the global head rotation; 10 FAPs controlling the outer lip positions; 4 FAPs controlling the nose; and 4 FAPs controlling the ears. It is possible to define new, macro-style FAPs which are linear combinations of the 66 predefined FAPs. It is also valid to transmit only a subset of the defined FAPs in order to animate a head model.

The output of the MPEG-4 decoding process is a stream of decoded FAPs. It is left to the implementation how to generate the FAPs at the transmitter side, and how to animate the 3D head model with the FAPs at the receiver side.

3 Analysis of Facial Animation Parameters from Video Sequences

While the MPEG-4 standard does not specify how to generate FAPs, we need a method which determines FAPs from video sequences in order to retrieve embedded watermarks from rendered sequences. A suitable method has been presented in [14,15] and is outlined in the following. We assume that the human head is locally deformable and that facial expressions can be represented by a linear combination of small elementary local movements. These movements are described by MPEG-4 FAPs. We further assume that we have a 3D model of the person that we want to determine FAPs from, for example acquired by a 3D laser scanner. In the following, we regard the FAPs as a time-varying k_{max}-dimensional vector

$$\mathbf{FAP}(t) \tag{1}$$

where k_{max} is the number of transmitted FAPs ($k_{max} \leq 66$), t is the integer time index of the video frame, and $FAP_k(t)$ is the k-th FAP at time t ($k \in \{1 \ldots k_{max}\}$). Thus, $FAP_{k=const}(t)$ denotes a sequence of a certain FAP value over time, while $\mathbf{FAP}(t = const.)$ denotes all FAPs at a given time instant.

In our algorithm, all FAPs are estimated simultaneously using a hierarchical optical flow based method. The optical flow constraint is combined with the parameterized 3D motion equations for each object point. The determination of the motion as a function of the FAPs is simplified due to the use of triangular B-splines for the head surface construction. In this way we only have to model the motion of a small number of control points. The three-dimensional scene used for parameter estimation and rendering of the synthetic images consists of a camera model and a head model.

3.1 Camera Model

For the camera model we use perspective projection where the 3D coordinates of an object point $[x\ y\ z]^T$ are projected into the image plane according to

$$X = X_0 - f_x \frac{x}{z}$$

$$Y = Y_0 - f_y \frac{y}{z}. \tag{2}$$

Here, f_x and f_y denote the focal length multiplied by scaling factors in x- and y-direction, respectively. These scaling factors transform the image coordinates into pixel coordinates X and Y. In addition, they allow the use of non-square pixel geometries. The two parameters X_0 and Y_0 describe the image center and its translation from the optical axis due to inaccurate placement of the CCD-sensor in the camera.

3.2 Head Model

The estimated animation parameters are constrained by a 3D model of the head and facial expressions. Like other well-known facial models [16,17], our head model also consists of a number of triangles onto which texture is mapped to obtain a photorealistic appearance [14]. The shape of the surface is, however, modeled by second order triangular B-splines [18] that allow to locally control the shape with a small number of control points $\mathbf{c}_i$. The coordinates of the vertices $\mathbf{v}_j$ in the mesh are then defined by the positions of the control points $\mathbf{c}_i$ according to

$$\mathbf{v}_j = \sum_{i \in I_j} N_{ji} \mathbf{c}_i, \quad \text{such that} \quad \sum_{i \in I_j} N_{ji} = 1, \tag{3}$$

with N_{ji} being the precalculated basis functions.

B-splines are well-suited for the modeling of facial skin [19] and constrain the motion of neighboring vertices which simplifies modeling of facial expressions. For the parameterization of facial expressions we adopt the MPEG-4 FAP definition tables [1]. By changing the facial animation parameters, the control points of the spline surface are moved which results in a new shape for the 3D model. Two examples of expressions rendered with this head model can be seen in Fig. 1.

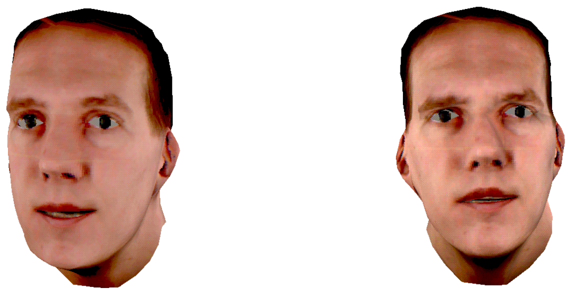

Fig. 1. Illustration of different synthesized facial expressions.

Our FAP estimation algorithm estimates the facial animation parameters from two successive frames of a natural or rendered video sequence. To avoid error accumulation in the long-term parameter estimation, a feedback loop is used [20,21] as depicted in Fig. 2. The model of the head is moved according to the

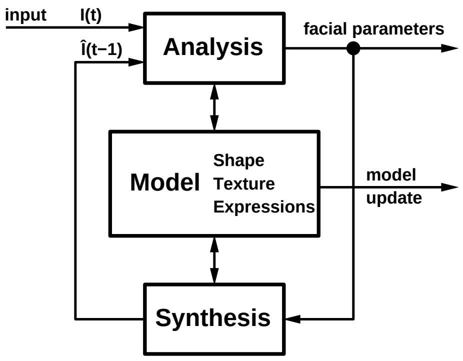

Fig. 2. Feedback structure of the coder.

parameters estimated from video frames I(t) and I(t-1), and a synthetic image is generated by rendering the model with modified shape and position. The estimation is then performed between the actual camera frame and the synthetic image of the previous frame which assures that no large misalignment of the model occurs. The feedback loop is also used in our hierarchical approach to compensate the facial motion before starting the next iteration on a higher resolution level. To reduce errors that are caused by linearizations in the algorithm, we first estimate the parameters with subsampled images. This rough estimate is then used to compensate the motion in the synthetic image leading to a new frame that is closer to I(t). These steps are repeated on higher resolution images resulting in more and more accurate facial parameters.

For the motion estimation the whole image is used by setting up the optical flow constraint equation

$$I_X \cdot u + I_Y \cdot v + I_t = 0 \tag{4}$$

where $[I_X\ I_Y]$ is the gradient of the intensity at point $[X\ Y]$, u and v the velocity in x- and y-direction and I_t the intensity gradient in temporal direction.

Instead of computing the optical flow field by using additional smoothness constraints and then extracting the motion parameter set from this flow field, we estimate the facial animation parameters from (4) together with the 3D motion equations of the head's object points [22]. For a rigid body motion the motion equation can be easily written in terms of a translation and a rotation. When allowing the body to be flexible, this is no longer possible and the motion equation must be set up at each object point independently. The trajectory of an object point is, however, not independent of its neighbors but is constrained by the head model that describes motion of surface points as a function of facial animation parameters.

Local deformations caused by facial expressions are taken into consideration by changing the facial animation parameters. These FAPs determine the position of all object points in the synthetic image by applying different transformations as shown in Fig. 3. First the control points of the surface are moved

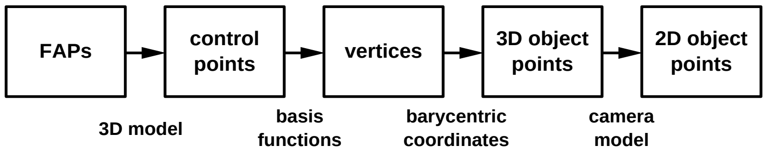

Fig. 3. Transformation from FAPs to image points.

according to the given FAPs. Using the basis functions of the splines the position of the vertices can be calculated from the control points. Three vertices form a triangle and the 3D motion of all object points inside this triangle specified by their barycentric coordinates is determined. The 2D image point is finally obtained by projecting the 3D point into the image plane. These transformations, that are all linear except for the projection, must be incorporated into our algorithm to obtain a second constraint that can be used together with (4).

The new control point position $\mathbf{c}_i'$ can be determined from the position $\mathbf{c}_i$ in the previous frame by

$$\mathbf{c}_i' = \mathbf{c}_i + \sum_k \Delta FAP_k(t) \;\; \mathbf{d}_{ik} \tag{5}$$

where

$$\Delta FAP_k(t) = FAP_k(t) - FAP_k(t-1) \tag{6}$$

is the change of facial animation parameter k between the two frames that are estimated by the algorithm and $\mathbf{d}_{ik}$ the 3D direction vector of the corresponding movement.

Strictly speaking, (5) is just valid for translations. If a number of control points are rotated around given axes by some action units, the description for the motion of control points becomes much more complicated due to the combination of rotation (defined by rotation matrices) and translation. The order of these operations can no longer be changed and the use of matrix multiplication results in a set of equations that is nonlinear in the parameters that have to be estimated. However, we can use the linear description (5) also for rotation, if we assume that the rotation angles between two successive frames are small. Then, the trajectory of a control point i that is rotating around the center O can be approximated by its tangent $\mathbf{d}_{ik}$ as shown in Fig. 4. This tangent differs for all object points, but we have to set up (5) for all points anyhow because of local deformations in the surface.

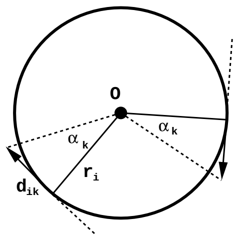

Fig. 4. Approximation of rotations.

For a rotation with the angle α_k

$$\alpha_k = \Delta FAP_k(t) \cdot s_k, \tag{7}$$

that is defined by the facial animation parameter $\Delta FAP_k(t)$ and the corresponding scaling factor s_k the length of the translation vector $\mathbf{d}_{ik}$ can be calculated by

$$|\mathbf{d_{ik}}| = \Delta FAP_k(t) \cdot r_i \cdot s_k. \tag{8}$$

Here, r is the distance between the object point and the given rotation axis. With this assumption (the direction of $\mathbf{d}_{ik}$ is specified by the direction of the tangent), (5) can also be used for rotation leading to a simple linear description for the FAPs that can be estimated very efficiently. The small error caused by the approximation vanishes after some iterations in the feedback structure shown in Fig. 2.

Having modeled the shift in control points, the motion of the vertices of the triangular mesh can be determined using (3) and the local motion of an object point $\mathbf{x}$ is calculated from that using

$$\mathbf{x} = \sum_{m=0}^{2} \lambda_m \mathbf{v}_m = \sum_{j \in J} (\sum_{m=0}^{2} \lambda_m N_{mj}) \mathbf{c}_j, \tag{9}$$

where λ_m are the barycentric coordinates of the object point in the triangle that encloses that point. The motion equation for a surface point can be represented as

$$\mathbf{x}' = \mathbf{x} + \sum_k \Delta FAP_k(t)\, \mathbf{t}_k = \mathbf{x} + \mathbf{T} \cdot \mathbf{\Delta FAP(t)}, \tag{10}$$

where $\mathbf{t}_k$'s are the new direction vectors of the corresponding facial animation parameter calculated from $\mathbf{d}_k$ by applying the linear transforms (3) and (9). $\mathbf{T}$ combines all the vectors in a single matrix and $\mathbf{\Delta FAP}(t)$ is the vector of all $\Delta FAP_k(t)$. Let $\mathbf{t}_x$, $\mathbf{t}_y$ and $\mathbf{t}_z$ be the row vectors of this matrix. The components of (10) are given by:

$$x' = x(1 + \frac{1}{x}\mathbf{t}_x \cdot \mathbf{\Delta FAP}(t)) \tag{11}$$
$$y' = y(1 + \frac{1}{y}\mathbf{t}_y \cdot \mathbf{\Delta FAP}(t)) \tag{12}$$
$$z' = z(1 + \frac{1}{z}\mathbf{t}_z \cdot \mathbf{\Delta FAP}(t)). \tag{13}$$

Dividing (11) and (12) by (13), inserting the camera model (2) and using a first order approximation leads to

$$\begin{aligned} u = X' - X &\approx -\frac{1}{z}(f_x\mathbf{t}_x + (X - X_0)\mathbf{t}_z)\mathbf{\Delta FAP}(t) \\ v = Y' - Y &\approx -\frac{1}{z}(f_y\mathbf{t}_y + (Y - Y_0)\mathbf{t}_z)\mathbf{\Delta FAP}(t). \end{aligned} \tag{14}$$

Together with (4) a linear equation at each pixel position can be set up

$$\frac{1}{z}\left[I_X f_x \mathbf{t}_x + I_Y f_y \mathbf{t}_y + (I_X(X - X_0) + I_Y(Y - Y_0))\mathbf{t}_z\right]\mathbf{\Delta FAP}(t) = I_t \tag{15}$$

with z being the depth information coming from the model. We obtain an overdetermined system that can be solved in a least-squares sense with low computational complexity. The size of the system depends directly on the number of implemented FAPs. Since in fact the incremental change of the FAPs is estimated from frame to frame, rather than their absolute value, we have to start with an imaginary video frame I(-1) which shows the head model in relaxed state, i.e.

$$FAP_k(-1) = 0 \quad \forall\, k \tag{16}$$

We then gain the absolute values of the FAPs for each frame by cumulative addition over ΔFAP for all frames from I(-1) up to the current frame. The described method estimates the FAPs with very satisfying accuracy [15].

4 Watermarking of MPEG-4 Facial Animation Parameters

For embedding of watermark data into the FAPs, we adopt a spread spectrum approach [23] that has been applied similarly to image and video watermarking before [5,8,9]. The idea is to apply small changes to the FAPs that seem random and are not conspicuous but correlate with a secret pseudo-random key. The correlation can be exploited to retrieve the embedded information later on. The watermark information to be embedded may be any arbitrary 1D or 2D bit pattern. Here we regard the watermark as a 2D bit pattern

$$WM(m,n) \in \{-1,+1\}, \quad m \in 0 \ldots m_{max}, \quad n \in 0 \ldots n_{max}. \qquad (17)$$

Please note that we denote the two possible states of a bit by -1 and $+1$, and not by 0 and $+1$, in order to have a mean-free signal.

4.1 Watermark Embedding

One basic concept of watermarking is to distribute one bit of watermark information over more than one FAP. For ease of understanding, we embed one bit of watermark information into a $M \times N$ block of FAPs, that means into M consecutive FAPs over N consecutive time instants, but we could also distribute one bit of watermark information over $M \times N$ randomly chosen FAPs from the whole set of FAPs, $FAP_k(t)$.

Before embedding, the stretched bit pattern is modulated, i.e. multiplied, with a filtered pseudo noise pattern $PN(k,t)$. This pseudo noise pattern is the secret key used for embedding, and the modulation makes the modulation product difficult to detect and manipulate. The pseudo-noise pattern can be generated by any random number generator that produces binary output values -1 and $+1$. For a discussion about the choice of a good pseudo random generator, please refer to [5]. We use a filtered pseudo noise pattern that is lowpass filtered along t, e.g. with a 9-tap lowpass filter, in order to decrease the noisiness in the watermarked FAPs which could result in annoying noisy behaviour of the head model.

Before finally adding the modulated bit pattern to the FAPs, we apply an amplitude adaptation in order to limit visible distortions. We found it a good choice to limit the maximum deviation of the watermarked FAPs from the unwatermarked FAPs to 3 % of the dynamic range for local FAPs like lip movement, and 1 % of the dynamic range for global FAPs like head rotation. We denote this amplitude adaptation by a factor of $\beta(k)$ which is an amplitude factor for FAP_k.

Putting all the above together, we obtain the watermarked FAPs from the unwatermarked FAPs by

$$FAP_k^{WM}(t) = FAP_k(t) + [WM(\lfloor k/M \rfloor, \lfloor t/N \rfloor) \cdot PN(k,t) \cdot \beta(k)] \quad (18)$$

where $\lfloor \ldots \rfloor$ is the floor operator.

Since the FAPs are a highly compressed representation of human face expressions, and since their data volume is very low, the signal space which can be used to embed information is also very limited. This results in the fact that the information embedded into the FAPs is not robust enough to resist all possible kinds of attacks, unless $M \times N$ is increased very much, that is, the data rate for the watermark information is decreased very much. However, already for choices of M and N like the ones that were used for the experiments in Section 5, namely $M = 2$ and $N = 76$, there is a certain robustness, and manipulation of single FAPs or DCT-based video compression of sequences rendered from the FAPs are not sufficient to impair the watermark.

The method according to (18) can easily be applied to streaming or fixed-length FAPs sequences in order to embed information in real-time, like the ID of a downloading client for download on the WWW.

The data rate r_{WM} of the watermark (the number of bits per second) depends on the number k_{max} of transmitted FAPs, frame rate R (number of FAPs sets per second), and the number $M \times N$ of FAPs that one bit of watermark information is embedded into and can be calculated as

$$r_{WM} = \frac{k_{max} \times R}{M \times N}. \quad (19)$$

4.2 Watermark Retrieval from the Facial Animation Parameters

Retrieval of the embedded watermark from the watermarked FAPs is done by subtraction of the unwatermarked FAPs from the watermarked FAPs, subsequent correlation with the same filtered PN sequence that has been used for embedding (and which is the secret key), summation over the window for each watermark bit, and thresholding as a bit decision. Put into an equation, the reconstructed watermark $\overline{WM}$ is

$$\overline{WM}(m,n) = sign \left[\sum_{k=m \cdot M}^{(m+1)\cdot M-1} \sum_{t=n \cdot N}^{(n+1)\cdot N-1} \left(FAP_k^{WM}(t) - FAP_k(t) \right) \cdot PN(k,t) \right] (20)$$

If the watermarked FAPs have been used to animate a head model and render a video sequence, and if the video sequence is available but the FAPs are not, the watermark cannot directly be retrieved. It is however possible to estimate the FAPs from the rendered sequence, for example with the method outline in Section 3, and to use the estimated FAPs to retrieve the watermark. Of course this is only possible if the video sequence is rendered such that the FAPs can be estimated. For example, mouth and eye of the model should be visible. The needed steps are

- Estimation of the FAPs from the rendered sequence
- Retrieval of the watermark from the estimated FAPs as described in Section 4.2

5 Experimental Results

In order to verify the applicability of the proposed watermarking method, several experiments were conducted.

In the experiments, we have embedded a binary information of 3 bytes into two different FAP streams of size $k_{max} = 17, t = 228$ (since the frame rate is 25 frames per second, these FAP streams represent sequences of $\frac{228}{25} = 9.12$ seconds each). Thus, $M = 2$ and $N = 76$, and each bit of watermark information has been embedded into 2 parameters over 76 frames, i.e., into 152 FAPs. The first bit of watermark information has thus been embedded into FAPs 1 and 2 over the first 76 frames, the second bit into FAPs 3 and 4 over the first 76 frames; and the last bit into FAPs 15 and 16 over frames 153 to 228. The data rate r_{WM} of the watermark is 2.8 bit/s (the frame rate is 25 Hz), or one byte every 2.9 s. This is not very much, but is sufficient for many applications. For example, an internet IP address (4 bytes) of a downloading client can be embedded every 11.6 seconds. Please note that only a subset of 17 out of the 66 admissible FAPs has been used here. If all possible FAPs are transmitted, more information can be embedded.

In the first experiment, we tried to retrieve the watermark information directly from the watermarked FAPs according to (20). It should be noted that in this case the watermarked FAPs were known, and the FAPs did not have to be estimated, as for the following experiments. For both used FAP sequences, it was possible to retrieve the watermarks without errors, as expected.

In the second experiment, the two watermarked FAP streams have been used to render two different animated head models into video sequences. The background was static in both cases, and the used resolution was 352×288 pixels. Figures 5 and 6 show representative frames from the two sequences. The FAPs

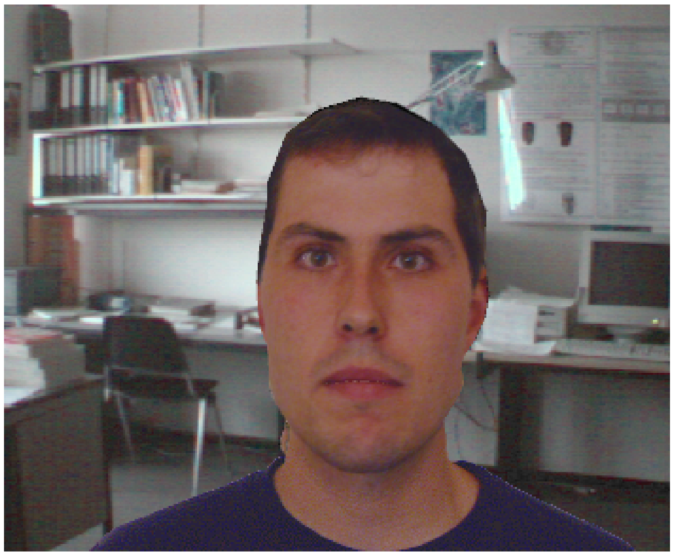

Fig. 5. One frame of a video sequence which was rendered from the watermarked FAP stream 1.

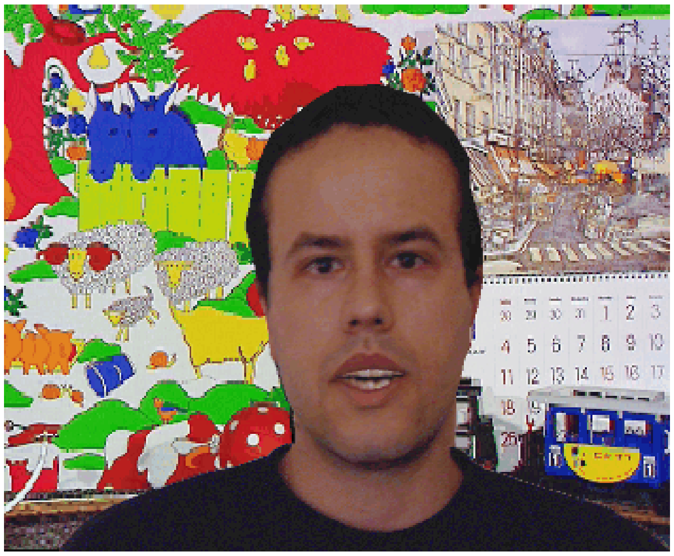

Fig. 6. One frame of a second video sequence which was rendered from the watermarked FAP stream 2.

were then estimated from the video sequences as described in Section 3, and the watermarks were retrieved from the estimated watermarked FAPs. For both used FAP sequences, it was possible to retrieve the watermarks without errors.

In the third experiment, we have used the rendered sequences of the second experiment and have compressed/decompressed them using MPEG-2 video compression. The bit-rate was chosen to 250 kbit/s for the first sequence and 600 kbit/s for the second sequence. Figures 7 and 8 show representative frames from the two sequences. The FAPs were then estimated from the compres-

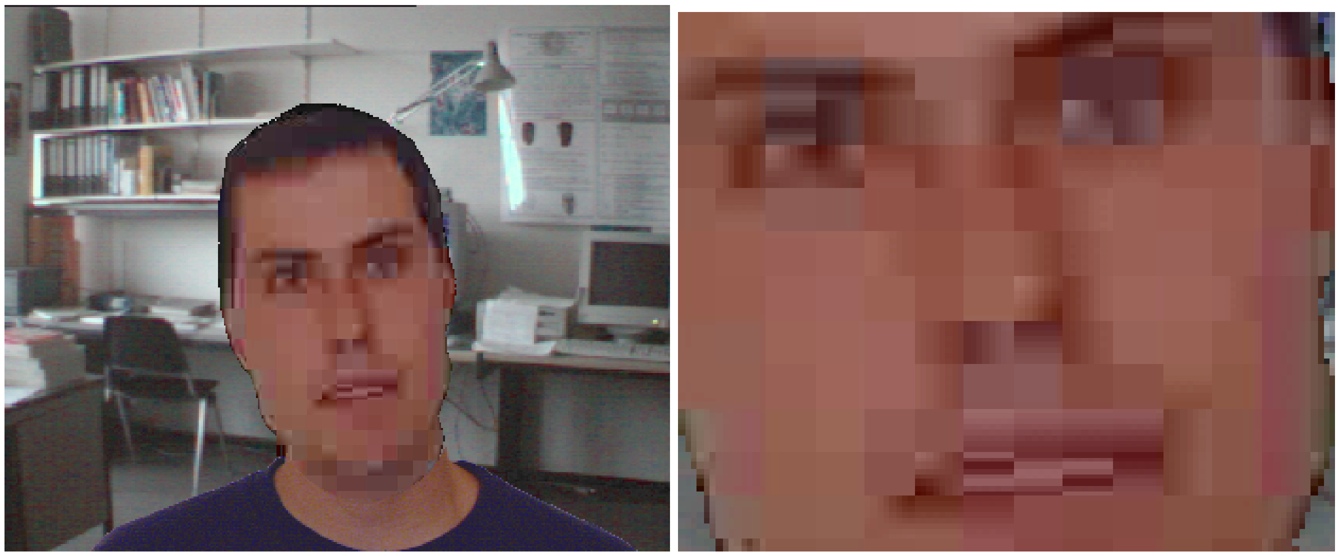

Fig. 7. One frame of the first video sequence from the second experiment after additional MPEG-2 compression at 250 kbit/s. The enlarged detail on the right demonstrates the compression artefacts.

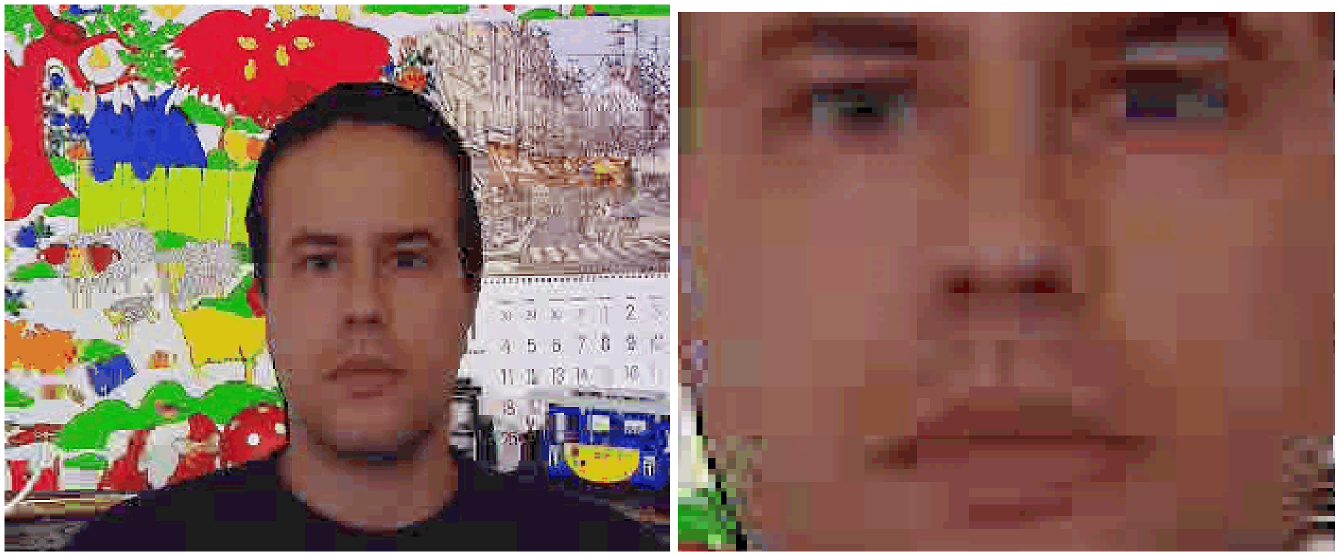

Fig. 8. One frame of the second video sequence from the second experiment after additional MPEG-2 compression at 600 kbit/s. The enlarged detail on the right demonstrates the compression artefacts.

sed/decompressed video sequences as described in Section 3, and the watermarks were retrieved from the estimated watermarked FAPs. For both used

FAP sequences, it was possible to retrieve the watermarks without errors for the chosen parameters. This successful retrieval from the MPEG-2 compressed sequences demonstrates that there is inherent robustness in the watermark.

We noticed however that, depending on the bit-rate of video compression and the data-rate of the embedded watermark, bit errors may occur. This is demonstrated in Fig. 9 where we have repeated the third experiment for the first FAP stream and for different choices of M and N (different tilings of the FAPs-time-plane) and, thus, for different watermark data rates r_{WM}. Shown here is the bit error rate of the watermark information vs. the data rate of the watermark information.

Though there is clearly a tendency to lower bit error rates of the watermark information for low data rates, sometimes bit error rates up to 0.2 may still occur, due to estimation errors in the FAPs estimation. Thus, it is advisable to protect the watermark information with an error correcting code.

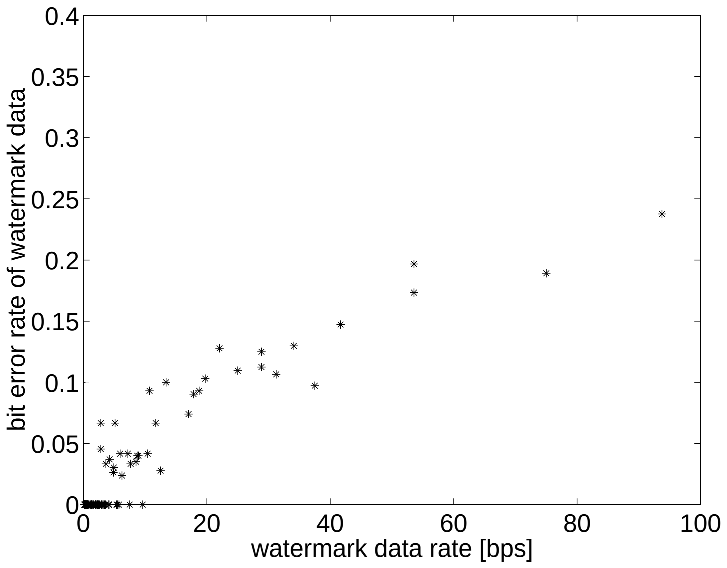

Fig. 9. Bit error rates of watermarks retrieved from a rendered and MPEG-2 encoded video sequence for different choices of the watermark data-rate.

6 Conclusions

We have presented a spread spectrum watermarking method which allows to embed information such as copyright information into facial animation parameters (FAPs) as they are used in MPEG-4 face animation. This allows to enforce intellectual property rights (IPR) for FAPs in a similar way as it has previously been proposed for audio, images, video [2], and 3D polygonal models [13]. The watermark is embedded into the FAPs and can be retrieved from the watermarked FAPs or from a video sequence rendered with the watermarked FAPs. For the estimation of the FAPs from video sequences we have presented a low complexity gradient-based approach. This algorithm exploits the shape, texture, and motion information from a 3D triangular B-spline head model whos facial expressions are modeled according to the MPEG-4 standard. We have shown results confirming the applicability of the scheme. Applications are for example marking and protection of MPEG-4 FAP data available on the internet or broadcasted in video delivery networks.

References

[1] ISO/IEC 14496-2, Coding of Audio-Visual Objects: Visual (MPEG-4 video), Committee Draft, October 1997.

[2] H. Berghel and L. O'Gorman. Protecting ownership rights through digital watermarking. *IEEE Computer*, pages 101–103, July 1996.

[3] J. Brassil, S. Low, N. Maxemchuk, and L. O'Gorman. Electronic marking and identification techniques to discourage document copying. *IEEE Journal on Selected Areas in Communications*, 13(8):1495–1504, October 1995.

[4] E. Koch and J. Zhao. Towards robust and hidden image copyright labeling. In *Proceedings of 1995 IEEE Workshop on Nonlinear Signal and Image processing*, Neos Marmaras, Greece, June 1995.

[5] I. Cox, J. Kilian, T. Leighton, and T. Shamoon. Secure spread spectrum watermarking for multimedia. Technical Report 95-10, NEC Research Institute, Princeton, NJ, USA, 1995.

[6] M. Kutter, F. Jordan, and F. Bossen. Digital signature of color images using amplitude modulation. In *Proceedings of Electronic Imaging 1997 (EI 97), San Jose, USA*, February 1997.

[7] D. Kundur and D. Hatzinakos. A robust digital image watermarking method using wavelet-based fusion. In *Proceedings IEEE International Conference on Image Processing 1997 (ICIP 97), Santa Barbara, CA*, volume 1, pages 544–547, October 1997.

[8] F. Hartung and B. Girod. Digital watermarking of raw and compressed video. In N. Ohta, editor, *Digital Compression Technologies and Systems for Video Communications*, volume 2952 of *SPIE Proceedings Series*, pages 205–213, October 1996.

[9] F. Hartung and B. Girod. Digital watermarking of uncompressed and compressed video. *Signal Processing, Special issue on copyright protection and access control for multimedia services*, vol. 66, no. 3, 1998.

[10] G.C. Langelaar, R.L. Lagendijk, and J. Biemond. Real-time labeling methods for MPEG compressed video. In *Proceedings 18th Symposium on Information Theory in the Benelux, Veldhoven, The Netherlands*, May 1997.

[11] M Swanson, B. Zhu, and A. Tewfik. Multiresolution video watermarking using perceptual models and scene segmentation. In *Proceedings IEEE International Conference on Image Processing 1997 (ICIP 97), Santa Barbara, CA*, volume 2, pages 558–561, October 1997.

[12] L. Boney, A.H. Tewfik, and K.H. Hamdy. Digital watermarks for audio signals. In *Proceedings EUSIPCO 1996*, Trieste, Italy, September 1996.

[13] R. Ohbuchi, H. Masuda, and M. Aono. Embedding data in three-dimensional polygonal models. In *Proceedings ACM Multimedia '97, Seattle, USA*, November 1997.

[14] P. Eisert and B. Girod. Facial expression analysis for model-based coding of video sequences. In *Proceedings Picture Coding Symposium (PCS 97)*, pages 33–38, Sep. 1997.

[15] P. Eisert and B. Girod. Model-based estimation of facial expression parameters from image sequences. In *Proceedings International Conference on Image Processing*, 2:418–421, Oct. 1997.

[16] F. I. Parke. Parameterized models for facial animation. *IEEE Computer Graphics and Applications*, pages 61–68, Nov. 1982.

[17] M. Rydfalk. *CANDIDE: A Parameterized Face.* PhD thesis, Linköping University, 1978. LiTH-ISY-I-0866.

[18] G. Greiner and H. P. Seidel. Modeling with triangular B-splines. *ACM/IEEE Solid Modeling Symposium*, pages 211–220, 1993.

[19] M. Hoch, G. Fleischmann, and B. Girod. Modeling and animation of facial expressions based on B-splines. *Visual Computer*, 1994.

[20] R. Koch. Dynamic 3-D scene analysis through synthesis feedback control. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(6):556–568, Jun. 1993.

[21] P. Eisert and B. Girod. Model-based 3D motion estimation with illumination compensation. In *Proceedings International Conference on Image Processing and its Applications*, 1:194–198, Jul. 1997.

[22] H. Li, P. Roivainen, and R. Forchheimer. 3-D motion estimation in model-based facial image coding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(6):545–555, Jun. 1993.

[23] Paul G. Flikkema. Spread-spectrum techniques for wireless communications. *IEEE Signal Processing*, 14(3):26–36, May 1997.