Vision, Modeling, and Visualization (2015)
D. Bommes, T. Ritschel, and T. Schultz (Eds.)

Data Driven 3D Face Tracking Based on a Facial Deformation Model

Dominik Sibbing and Leif Kobbelt

RWTH Aachen University, Germany

Abstract

We introduce a new markerless 3D face tracking approach for 2D video streams captured by a single consumer grade camera. Our approach is based on tracking 2D features in the video and matching them with the projection of the corresponding feature points of a deformable 3D model. By this we estimate the initial shape and pose of the face. To make the tracking and reconstruction more robust we add a smoothness prior for pose changes as well as for deformations of the faces. Our major contribution lies in the formulation of the smooth deformation prior which we derive from a large database of previously captured facial animations showing different (dynamic) facial expressions of a fairly large number of subjects. We split these animation sequences into snippets of fixed length which we use to predict the facial motion based on previous frames. In order to keep the deformation model compact and independent from the individual physiognomy, we represent it by deformation gradients (instead of vertex positions) and apply a principal component analysis in deformation gradient space to extract the major modes of facial deformation. Since the facial deformation is optimized during tracking, it is particularly easy to apply them to other physiognomies and thereby re-target the facial expressions. We demonstrate the effectiveness of our technique on a number of examples.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [COMPUTER GRAPHICS]: Three-Dimensional Graphics and Realism—Animation; I.4.8 [IMAGE PROCESSING AND COMPUTER VISION]: Scene Analysis—Tracking

1. Introduction

In many industrial movie and computer game productions, facial performance capture has evolved to an indispensable part of the production pipeline. Modern systems are used to (re)animate the face of an actor or to re-target her facial expressions to another computer animated character. In general, such systems require special equipment, such as large and well calibrated camera arrays, laboratory lighting conditions or expensive (marker-based) 3D scanning and tracking devices. From the usability point of view a system would be preferable which tracks 3D faces from simple videos captured with a consumer level camera.

In this paper we propose such a system which captures the facial performance of an actor from simple 2D images. Since this problem is ill-conditioned, we use the motion data contained in a large database of different facial expressions as a deformation prior to stabilize the tracking process. Different to other approaches we derive a deformation model from the database, which separates the facial movements from the individual face geometries. During the tracking this automatically ensures that we do not blend between different faces as a conventional shape model would do, but only deform an individual face according to our deformation model. One big advantage of this is, that the computed facial deformations can directly be applied to other faces, which instantaneously enables re-targeting of facial animations. Furthermore we analyze the deformation data in our database and compute a general time dependent movement model for facial expressions which is used as a temporal prior for facial movements. Similar to a shape model [BV99], which is able to reconstruct plausible facial shapes not contained in the input database, this movement model is also applicable to new persons and requires no individual training per person. Potential applications include the generation of personalized avatars for computer games where individual facial movements are recorded by a consumer-level webcam before transferred to a virtual character. Another scenario are video chats, which require only a small bandwidth. Since our deformation model encodes facial movements by only a few parameters, it is possible to transmit those parameters to reconstruct and visualize an abstract version of the speaker instead of transmitting the complete video stream.

The paper is organized as follows: after discussing the related work and giving details about our database, Section 4 defines the deformation space for faces and explains how the deformation parameters are related to the appearance of a face. In Section 5 we detail the actual tracking process which tries to find deformation parameters and a rigid transformation such that projected feature points are close to feature points observed by a 2D Active Appearance Model (AAM) and such that the previously reconstructed motion sequence is continued in a plausible way according to our motion model. In Section 6 we show how the computed deformation can be used for facial re-targeting and before concluding the paper we present results in Section 7.

Contributions

- Instead of optimizing shape parameters and thereby blending between different faces, we run our optimization in a deformation space representation, which decouples the dynamic motion from the individual shapes of the faces, and which makes re-targeting of facial animations particularly easy.
- Our system is purely image based and computationally not involved. It can easily be integrated in a facial feature tracker like AAMs, which runs at high frame rates even on mobile devices.
- Our system uses a general deformation and motion model derived from a large database. It does not need an individual training phase per persons, which makes it interesting for simple consumer level applications.

2. Related work

3D Motion capture. In many industrial movie or game productions facial performance capture is still based on motion capture. Therefore special cameras track the movements of 3D marker points placed on the actors face. Sifakis et al. deduce muscle activities from the sparse motion data obtained with such systems and feed the resulting forces in a finite element simulation to drive a carefully designed, complex and anatomically plausible muscle model [SNF05]. Curio et al. computed blendshape weights from the motion capture data to drive a morphable shape model [CBK*06]. Using such sparse sets of markers, wrinkles and fine details within the facial movements disappear. Bickel et at. suggest a thin shell based mesh deformation approach to simulate the appearance of wrinkles in a physically plausible manner [BBA*07]. All systems use special camera systems to track the 3D motion of marker points, while our system only needs a simple video as input.

Dense reconstruction. With increased computational power, dense reconstruction and tracking methods have moved into focus and can roughly be divided into active sensing and passive sensing systems. Zhang et al. use a structured light scanner and compute a time varying disparity map for two input video streams by solving a global optimization problem [ZSCS04]. Fitting a generic face template to the 3D movements, they are able to capture the facial performance of an actor. Such scanners usually have the problem of a low temporal resolution. This problem was tackled by Weise et al. [WLVP09] by using a modified structured light scanner capable of capturing at 30 fps. Ma et al. use polarized light emitted from multiple directions (lightstage) to separately measure a diffuse and a specular normal map [MHP*07]. In [ARL*09] this technique was further improved to compute high quality renderings of the facial performance of an actor in an offline process. Other active sensing methods [HVB*07] use multispectral photometric stereo to compute a dense normal field. Weise et al. propose a real-time facial performance capture system, which requires as input depth maps obtained from a Kinect sensor [WBLP11]. While our temporal prior is similar to theirs, we use a general model independent of the tracked person and require only 2D videos as input.

In comparison to active sensing methods passive methods usually reconstruct the facial movements from simple video data. The main advantage might be, that the actor is not distracted by changes of the illumination conditions at high frequencies. Vedula et al. introduced dense scene flow as a three dimensional, time dependent vector field [VBR*99]. Employing an expensive, complex and carefully calibrated camera setup, combining optical flow and 3D stereo reconstruction can lead to impressive and realistic reconstructions of human actors as demonstrated in [BPL*05]. Li and Sclaroff improved the early work of Vedula by simultaneously computing depth and optical flow from binocular stereo sequences [LS08]. Furukawa and Ponce us a multiview stereo system and define filters over the local neighborhood of a vertex to smoothly track the vertices of a triangular mesh [FP09]. In a recent work, Bradley et al. use a multiview stereo setup with well controlled illumination conditions to separately reconstruct single patches of a human face, from which the face is reassembled [BHPS10]. One main disadvantage of such systems is that they usually need large computation times to track the facial performance.

Passive as well as active sensing methods both only work under laboratory conditions, with well calibrated systems, while our system is also applicable using videos produced by consumer grade hardware.

Vision based capture systems infer the 3D facial movements from a single video and usually involve a priori knowledge about the facial shape. Marker based methods [WL90, LO05] produce quite robust motion parameters used to deform a morphable model. In [XBMK04] Xiao et al. use a non-rigid-structure-from-motion approach to augment a 2D Active Appearance Model (AAM) [CET01, MB04] with additional 3D shape modes in order to track 3D facial features. Similar to [BV99, LRF93] and [DM96] Pighin et al. build a 3D morphable model and used it to track facial expressions in an analysis by synthesis approach [PSS99]. Chuang and Bregler present a framework to manually model facial animations assisted by the shape and motion information of a large database [CB02]. Their system involves an expensive search in the database and does not run in real-time. Chai et al. track facial features to extract 2D motion control signals, which are mapped to high quality 3D facial expressions using a preprocessed database [CXH03]. While all systems model facial expressions as some sort of shape combination, we use a deformation model extracted from the motion data itself. This decouples the shape of the face from its dynamic motion and thereby reduces the number of parameters to be optimized and is user independent.

For a more complete overview about facial performance capture see [PL06].

3. Database of dynamic facial expressions

To collect the motion data for our deformation model, we used a rig of 4 cameras to capture moving faces at 40 frames per second with a resolution of 780 $\times$ 580. We recorded 60 subjects performing 20 facial expressions related to the extreme poses suggested in [EF78]. On average a facial expression took 2 seconds to perform, which leads to a total number of about 96.000 frames to be reconstructed. To robustly and automatically perform this reconstruction and tracking task for this amount of frames we used a system similar to the one suggested in [SHK11] and generated for each subject and each dynamic facial expression a sequence of meshes M_f. Here f represents the frame number within the respective dynamic facial expressions. Sliding a time window of length k over an animation with F frames we can extract $F-k+1$ animation snippets of equal length k. Each animation in our database is split into such sets of animation snippets, each containing exactly k frames. Since images are taken at a rather coarse resolution the resulting meshes do not show detailed features like e.g. wrinkles.

4. Deformation space Representation

Given a triangle mesh A and a deformed version A' having the same mesh connectivity, we can compute a deformation gradient $\mathbf{S}(t) \in \mathbb{R}^{3\times 3}$ for each triangle t. This matrix represents the change of shape relative to a reference shape. The precondition to compute deformation gradients is that both meshes are in full correspondence. That is why we designed our facial capture system to be able to produce the required consistent mesh topology where each mesh has m triangles and n vertices. The concatenation of all gradients to a matrix $\mathbf{S} \in \mathbb{R}^{3m\times 3}$ encodes the deformation of A to A' [BSPG06].

Using deformation transfer [BSPG06] the change of the shape can be applied to another face. Since deformation gradients encode this change and thereby represent the motion independent from the underlying facial geometry, we encode each reconstructed mesh M_f in a sequence by its deformation gradient $\mathbf{S}_f$ w.r.t. the neutral pose M_0 of the respective person. We do this for all sequences of our database. The resulting matrix $\mathbf{S}_f$ for each frame can be represented as a $9 \cdot m$ dimensional vector. We feed the entire database of 96.000 frames into a Principal Component Analysis (PCA). By this we extract the average deformation gradient $\bar{\mathbf{S}} \in \mathbb{R}^{3m\times 3}$ and a matrix of the l most important eigen-gradients $\mathbf{S}_{eg} \in \mathbb{R}^{3m\times 3l}$ encoding the main deviations from $\bar{\mathbf{S}}$ (Fig. 1). Given l deformation parameters $(s_1,\ldots,s_l)$, we assemble a $3l \times 3$ parameter matrix

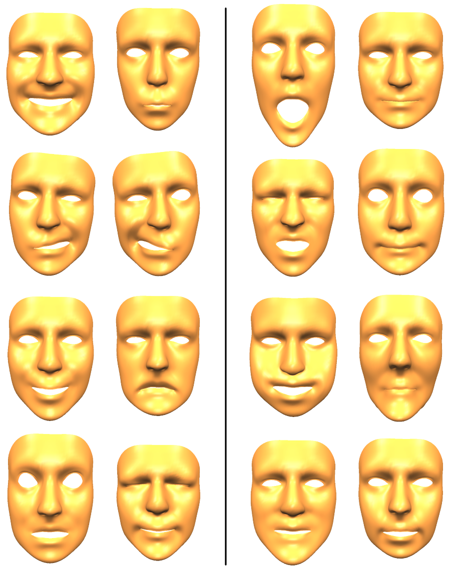


Figure 1: *The eight largest eigen-gradients represent meaningful facial actions. Each pair of a left and right face show the lower and upper range of one deformation parameter.*

$$\mathbf{s} = \begin{bmatrix} s_1 & & \\ & s_1 & \\ & & s_1 \\ & \cdots & \\ s_l & & \\ & s_l & \\ & & s_l \end{bmatrix}$$

and express an deformation gradient $\mathbf{S}$ as a linear combination of eigen-gradients:

$$\mathbf{S} = \bar{\mathbf{S}} + \mathbf{S}_{eg} \cdot \mathbf{s}$$

We approximate each deformation gradient $\mathbf{S}_f$ by such a low dimensional parameter matrix $\mathbf{s}_f$, represented as a point $\mathbf{s}_f = (s_{f,1},\ldots,s_{f,l})^T$ in deformation space. In the following derivation we switch between the matrix and vector notation where appropriate, but since a deformation gradient $\mathbf{S}$ is always encoded as a matrix it will become clear from the context which notation is meant. An animation snippet then is a trajectory in this deformation space represented as a $k \cdot l$ dimensional vector $(\mathbf{s}_1^T,\ldots,\mathbf{s}_k^T)^T$.

4.1. Deformation space to shape space Transformation

For our face tracking algorithm we need to be able to compute the actual geometry, i.e. the vertex positions of

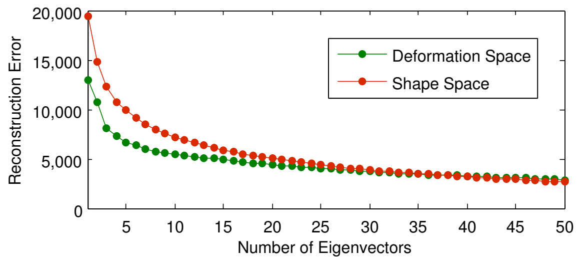


Figure 2: *Approximation error obtained using different numbers of eigenvectors in a standard shape space and in our suggested deformation space. For a small number of parameters the deformation space is able to approximate faces significantly better than a standard shape model.*

a mesh, given a point in deformation space. As described in [BSPG06], deformation gradients $\mathbf{S}$ computed from a pair (A, A') can be applied to a new mesh B – having the same topology as A – to produce a mesh B' showing a similar deformation. The vertex positions $\mathbf{p}' \in \mathbb{R}^{n\times 3}$ of the mesh B' are computed by solving the linear system

$$\mathbf{G}^T\mathbf{D}\mathbf{G}\mathbf{p}' = \mathbf{G}^T\mathbf{D}\mathbf{S} \ ,$$

where $\mathbf{G} \in \mathbb{R}^{3m\times n}$ and $\mathbf{D} \in \mathbb{R}^{3m\times 3m}$ are the gradient and area matrix of B. Observe that $\mathbf{G}^T\mathbf{D}\mathbf{G}$ evaluates to the well known cotangent Laplace matrix. Since this matrix does not have full rank, one normally augments the linear system with additional constraints, e.g., by fixing some vertex to a certain position in space. Instead of fixing the position of an arbitrary vertex, we suggest to use a Tikhonov regularization [PTVF07], which adds the identity matrix $\mathbf{I}$ scaled by a small value ε to the system. We select $\varepsilon = trace(\mathbf{G}^T\mathbf{D}\mathbf{G}) \cdot 10^{-6}$, which affects the solution of the system only slightly:

$$(\mathbf{G}^T\mathbf{D}\mathbf{G} + \varepsilon \cdot \mathbf{I})\mathbf{p}' = \mathbf{G}^T\mathbf{D}\mathbf{S} \ .$$

Now the system is invertible and we can express the new vertex positions as a linear combination of deformation parameters $\mathbf{s}$:

$$\begin{aligned} \mathbf{p}' &= (\mathbf{G}^T\mathbf{D}\mathbf{G} + \varepsilon \cdot \mathbf{I})^{-1}\mathbf{G}^T\mathbf{D}(\bar{\mathbf{S}} + \mathbf{S}_{eg} \cdot \mathbf{s}) \\ &= \mathbf{M}_{DS} \cdot \mathbf{s} + \mathbf{b}_{DS} \end{aligned}$$

with

$$\begin{aligned} \mathbf{M}_{DS} &= (\mathbf{G}^T\mathbf{D}\mathbf{G} + \varepsilon \cdot \mathbf{I})^{-1}\mathbf{G}^T\mathbf{D}\mathbf{S}_{eg} & \in \mathbb{R}^{n\times 3l} \\ \mathbf{b}_{DS} &= (\mathbf{G}^T\mathbf{D}\mathbf{G} + \varepsilon \cdot \mathbf{I})^{-1}\mathbf{G}^T\mathbf{D}\bar{\mathbf{S}} & \in \mathbb{R}^{n\times 3} \ . \end{aligned}$$

Note that the gradient and area matrices need to be derived only once from the reference mesh – i.e. the mesh of the neutral face of a specific person – and so $\mathbf{M}_{DS}$ and $\mathbf{b}_{DS}$ have to be precomputed once for each person but the deformation gradient model $(\bar{\mathbf{S}}, \mathbf{S}_{eg})$ itself is independent from the individual person.

A standard shape model, computed from the distribution of the vertex positions, simultaneously encodes individual shapes as well as deformations. Therefore the variance in such a model is higher than the variance of the pure deformation gradient data, where we factor out the “shape” component leading to more coherence. This in turn results in a more compact representation in deformation space and thus requires fewer parameters for the actual tracking procedure. In Figure 2 we compare the approximation power of the deformation gradient space with that of shape space. For this we successively used more and more eigenvectors to approximate the faces in our database and measured the error (in mm) based on the accumulated vertex distances between reconstruction and original mesh. To compute the matrices $\mathbf{M}_{DS}$ and $\mathbf{b}_{DS}$ we used the mesh of the neutral face of the respective person. When employing only a few eigen-gradients, the deformation space representation actually shows a much better approximation error than a standard shape model which allows us to use fewer parameters during tracking. Thus our deformation space representation is a very natural model for face tracking where the deformation state of the face changes over the sequence but not the underlying physiognomy.

5. Expression tracking

The input to our tracking procedure is a sequence of facial images. To get the initial shape of the neutral face, as seen in the first frame, one can, e.g., use an approach similar to [BV99] to optimize shape parameters of a morphable model obtained from the neutral expressions of the database. From such a reconstruction we compute the constant gradient and diagonal matrices $\mathbf{G}$ and $\mathbf{D}$, and thereby $\mathbf{M}_{DS}$ and $\mathbf{b}_{DS}$.

To each frame of our video sequence we fit an 2D AAM [MB04] in order to detect the facial features around the eyes, the nose and the mouth. This captures the individual features of the person to be tracked. The remaining part of the tracking algorithm is completely independent from the individual face. Let the 2D feature positions in the current frame f be $\mathbf{a} = (\mathbf{a}_1, \dots, \mathbf{a}_{|a|})$. For the rigid motion we allow three degrees of freedom for the translation $\mathbf{t} = (t_x, t_y, t_z)^T$ and three degrees of freedom for the rotations, parametrized by Euler angles $\boldsymbol{\omega} = (\alpha, \beta, \gamma)^T$, such that rigid motion parameters are defined by $\mathbf{r} = (\alpha, \beta, \gamma, t_x, t_y, t_z)^T$. Given the 2D feature positions we want to compute the most probable deformation parameters $\mathbf{s}$ as well as rigid motion parameters $\mathbf{r}$, such that

1. The projections of the 3D feature points – i.e. vertices of the resulting 3D mesh template corresponding to the 2D features – are close to the points $\mathbf{a}$ in image space.
2. The sequence $\tilde{\mathbf{r}} = (\mathbf{r}_{f-k+1}^T, \dots, \mathbf{r}_{f-1}^T)^T$ of rigid motion parameters already computed for the $k-1$ previous frames is smoothly extended
3. The sequence $\tilde{\mathbf{s}} = (\mathbf{s}_{f-k+1}^T, \dots, \mathbf{s}_{f-1}^T)^T$ of deformation parameters encoding the already computed animation of the $k-1$ previous frames is reasonably continued

Note that the resolution of the AAM is independent of the resolution of the 3D mesh template and its vertex positions only drive the tracking process. Since increasing the resolution of the AAM effectively increases the number of energy terms while the number of parameters stays constant, the optimization is expected to be more stable and accurate. Similar to Weise [WBLP11], we formulate this optimization as a maximum a posteriori estimation

$$(\mathbf{s},\mathbf{r})^* = \arg\max_{\mathbf{s},\mathbf{r}} p(\mathbf{s},\mathbf{r}|\mathbf{a},\tilde{\mathbf{s}},\tilde{\mathbf{r}})$$

where $p(\cdot|\cdot)$ represents the conditional probability. Using Bayes' rule and the independence of rigid motion and deformation we can split this into three probabilities

$$(\mathbf{s},\mathbf{r})^* = \arg\max_{\mathbf{s},\mathbf{r}} \underbrace{p(\mathbf{a}|\mathbf{s},\mathbf{r})}_{\text{likelihood}} \cdot \underbrace{p(\mathbf{r},\tilde{\mathbf{r}})}_{\text{rigid prior}} \cdot \underbrace{p(\mathbf{s},\tilde{\mathbf{s}})}_{\text{shape prior}}$$

and iteratively maximize this probability by computing the minimum of its negative logarithm:

$$E(\mathbf{s},\mathbf{r}) = -\ln p(\mathbf{a}|\mathbf{s},\mathbf{r}) - \ln p(\mathbf{r},\tilde{\mathbf{r}}) - \ln p(\mathbf{s},\tilde{\mathbf{s}})$$

In what follows we explain the individual terms of this energy function. Derivatives w.r.t. the parameters $\mathbf{s}$ and $\mathbf{r}$ are given in the Appendix.

5.1. Likelihood of the AAM

Intuitively, the observed facial feature positions $\mathbf{a}_i$ are a probable explanation of the projected feature points $\mathbf{q}_i$, if the distances between those points in image space are small. This can be modeled by the probability

$$p(\mathbf{a}|\mathbf{s},\mathbf{r}) = \prod_{i=1}^{|a|} \frac{1}{2\pi\sigma_{aam}} e^{-\frac{\|\mathbf{q}_i-\mathbf{a}_i\|^2}{(2\sigma_{aam})^2}} \quad ,$$

where σ_{aam} controls the kernel size of the probability function. The negative logarithm of this expression evaluates to an energy term of the form

$$E_{aam}(\mathbf{s},\mathbf{r}) = \lambda_{aam} \cdot \sum_{i=1}^{|a|} \|\mathbf{q}_i - \mathbf{a}_i\|^2$$

where λ_{aam} is a constant dependent on the kernel parameter σ_{aam}. In order to compute the 2D positions $\mathbf{q}_i$ we only need to project those 3D vertices $\mathbf{p}_j$ of the face model, corresponding to the facial feature points $\mathbf{a}_i$. Therefore we define a selection matrix $N \in \mathbb{R}^{|a|\times n}$ as

$$N_{i,j} = \begin{cases} 1 & \text{if } \mathbf{a}_i \text{ corresponds to } \mathbf{p}_j \\ 0 & \text{otherwise} \end{cases}$$

By taking the respective rows $N_{i,\cdot}$ we can directly compute the 3D positions of the i-th facial feature point given a set of deformation parameters

$$\mathbf{p}_j = \left[\, N_{i,\cdot} \cdot (\mathbf{M}_{DS} \cdot \mathbf{s} + \mathbf{b}_{DS}) \,\right]^T$$

Then the points are transformed according to the rigid motion parameters and projected into the image by using the camera's intrinsic calibration matrix $\mathbf{K} \in \mathbb{R}^{3\times 3}$, such that

$$\mathbf{q}_i = \left(\frac{a}{c}, \frac{b}{c}\right)^T$$

where

$$\begin{pmatrix} a \\ b \\ c \end{pmatrix} = \mathbf{K} \cdot (\mathbf{R}_0 \cdot \mathbf{R}_\alpha \cdot \mathbf{R}_\beta \cdot \mathbf{R}_\gamma \cdot \mathbf{p}_j + \mathbf{t})$$

In order to avoid the problem of a Gimbal lock, we compute an initial rotation $\mathbf{R}_0 \in \mathbb{R}^{3\times 3}$ for the first frame of the sequence and start the tracking with $\alpha = \beta = \gamma = 0$.

5.2. Rigid motion prior

We consider the rigid motion of an animation as probable, if linear and rotational acceleration are small. Acceleration can be estimated by computing the second derivative of the position and rotation w.r.t. time. Instead of using a higher order interpolation taking $k-1$ previous frames into account we decided to use a simple backwards scheme to estimate the acceleration from two previous frames only:

$$\begin{aligned} \ddot{\mathbf{t}} &= \frac{\mathbf{t} - 2\mathbf{t}_{f-1} + \mathbf{t}_{f-2}}{dt^2} \\ \ddot{\boldsymbol{\omega}} &= \frac{\boldsymbol{\omega} - 2\boldsymbol{\omega}_{f-1} + \boldsymbol{\omega}_{f-2}}{dt^2} \end{aligned}$$

The probability of a plausible rigid movement, i.e. with low acceleration, then is given by

$$p(\mathbf{r},\tilde{\mathbf{r}}) = \frac{1}{(2\pi)^3 \cdot \sqrt{\sigma_t \cdot \sigma_\omega}} \cdot e^{-\frac{\|\ddot{\mathbf{t}}\|^2}{2\sigma_t} - \frac{\|\ddot{\boldsymbol{\omega}}\|^2}{2\sigma_\omega}} \quad ,$$

which again boils down to a simple energy term E_{rigid} when applying the negative logarithm

$$E_{rigid}(\mathbf{r},\tilde{\mathbf{r}}) = \lambda_t \|\ddot{\mathbf{t}}\|^2 + \lambda_\omega \|\ddot{\boldsymbol{\omega}}\|^2 \quad ,$$

where λ_t and λ_ω are constants controlling the influence of this energy term and depending on the kernel parameters σ_r and σ_ω.

5.3. Shape deformation prior

In Section 3 we explained how to split the animations in our database into snippets. Each snippet can be represented by concatenating its sequence of deformation parameters $(\tilde{\mathbf{s}}^T, \mathbf{s}^T)^T$ to a $k \cdot l$ dimensional vector. Similar to Weise et al. [WBLP11] we compute a Gaussian Mixture Model to estimate the likelihood of a given animation snippet w.r.t. our observed animation data stored in the database. The main difference to the original approach of Weise is that our deformation gradient model is independent of individual faces and needs to be computed only once.

A Gaussian Mixture Model is able to approximate complex data distributions by a set of local Gaussian models. Given weights π_i, center positions of local Gaussian kernels $\boldsymbol{\mu}_i$ and covariance matrices $\boldsymbol{\Sigma}_i$ one can compute the probability of a point $\mathbf{x} \in \mathbb{R}^d$ as a sum of normal distributions

$$p_{GM}(\mathbf{x}) = \sum_i \pi_i \frac{1}{\sqrt{(2\pi)^d |\boldsymbol{\Sigma}_i|}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)}$$

In our setting the goal is to compute a probability value for a given animation snippet. The snippet representation still contains a lot of redundancy and with typical values for

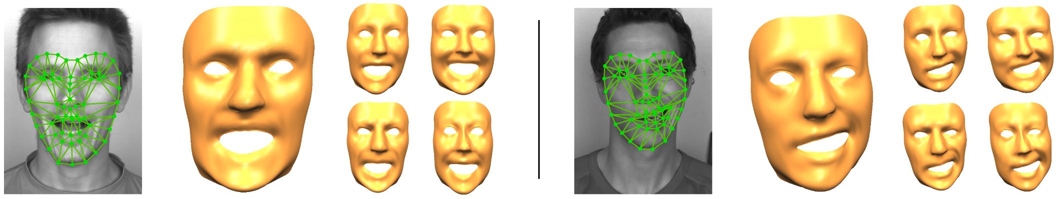


Figure 3: *Our tracking approach allows to directly transfer an expression to other faces. In this example we demonstrate the re-targeting of facial expressions by carrying over a tracking result of an intermediate frame to four different facial shapes.*

$k = 3$ and $l = 15$ it is of a rather high dimension. Silverman theoretically derived that the number of samples, needed to compute a density distribution function, grows exponentially with the dimensions of the samples [Sil86]. So we decided to learn the Gaussian Mixture Model in a space of reduced dimensionality obtained by a PCA and thereby stabilize the learning procedure.

Training the parameters of the mixture model. Performing a PCA on the snippet coefficients $\mathbf{y} = (\tilde{\mathbf{s}}^T, \mathbf{s}^T)^T$ results in a vector $\bar{\mathbf{y}} \in \mathbb{R}^{k \cdot l}$ representing the average coefficient and a matrix $\mathcal{A} \in \mathbb{R}^{(k \cdot l) \times d}$ storing the main variations modes w.r.t. the average deformation. Keeping 99% of the energy reduces the dimensionality with the above parameters to $d = 20$. We define the sample points $\mathbf{x}$ to be the compact representation of the snippet coefficients $\mathbf{y}$

$$\mathbf{x} = \mathcal{A}^T \cdot (\mathbf{y} - \bar{\mathbf{y}})$$

and employ the EM Algorithm [Bil98] to robustly learn the parameters $\pi_i \in \mathbb{R}$, $\boldsymbol{\mu}_i \in \mathbb{R}^d$ and $\boldsymbol{\Sigma}_i \in \mathbb{R}^{d \times d}$ in this reduced space. Finally we define the last energy term as

$$E_{shape}(\mathbf{s}, \tilde{\mathbf{s}}) = -\ln(p_{GM}(\mathcal{A}^T \cdot (\mathbf{y} - \bar{\mathbf{y}})))$$

6. Re-Targeting Facial Animations

Mapping the tracking result $(\mathbf{s}, \mathbf{r})$ of a single frame to another physiognomy encoded as a mesh B is peculiar easy. Therefore we only need to precompute the matrix $\mathbf{M}_{DS}$ and the vector $\mathbf{b}_{DS}$ depending on the shape of B (cf. Section 4.1) and compute the new vertex positions of B as

$$\mathbf{p}(\mathbf{s}, \mathbf{r}) = \mathbf{R}_\alpha \cdot \mathbf{R}_\beta \cdot \mathbf{R}_\gamma \cdot (\mathbf{M}_{DS} \cdot \mathbf{s} + \mathbf{b}_{DS}) + \mathbf{t}$$

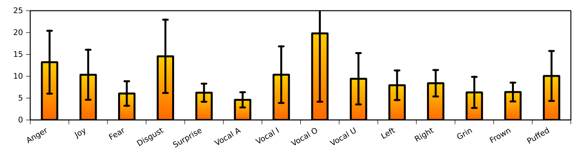
In Figure 3 we exemplarily re-target a facial expression from an intermediate frame to the faces of four different persons.

7. Results

We run all our experiments on an Intel® Core^{TM} i7 CPU with 2.67GHz and used a Stingray F-046B camera for the capture. Since the suggested system has only to fulfill a few constraints (AAM vertex positions) and is minimally parametrized using the deformation space formulation, our Levenberg-Marquard [PTVF07] implementation only needs 16 milliseconds on average to optimize the energy function in each frame. Using the approach of Matthews [MB04] fitting an AAM to one frame takes 20ms. By parallelizing the 3D face tracking in a given frame with the AAM computation of the next frame we obtain a frame rate of 40 fps with one frame latency.

Our experiments indicated that using rather small snippets ($k = 3$) is a good choice to predict the facial movement. This is not surprising when considering the fact that facial movements are quite fast: even at capture rates of 40 frames it takes only 3 to 5 frames to change a facial expression from, e.g., "neutral" to "surprised". For all our experiments we used 45 Gaussians to train the Gaussian Mixture Model.

In Figure 4 and the accompanying video we used sequences of a length of up to 10 seconds and show results produced with our system. In each experiment we excluded the facial movements of the tracked person from the database. The left image in each example shows one image of a video sequence overlayed with the detected facial feature points. The middle image shows the result produced by our tracking approach. Since we used our own multiview stereo capture system to acquire the database we are able to compare our result with a ground truth 3D stereo reconstruction obtained from a 4 cameras rig (right image). We use the stereo reconstruction of the neutral face as reference mesh to initialize the tracking procedure. One can see that the resulting deformations are quite similar to the ground truth reconstructions. To quantify this a bit more, we took the tracking result of one subject performing different facial expressions and measured in each frame the error as the average distance between vertices of the tracked mesh and their corresponding vertices of the stereo reconstruction. This allows us to



compute an average error and its standard deviation for each sequence of approximately 2 seconds length, as shown in the diagram. Except for the expression "Vocal O" we observe a rather small average error of about 10-15 millimeters.

Limitations Due to the resolution of our template meshes, the presented approach is not intended to generate high quality 3D models from videos and fails to reproduce details like e.g. wrinkles. In this paper we focus on the benefit of our deformation model for tracking and re-targeting of facial expressions and only incorporate a simplistic AAM model, which is not trained for large head rotations and does not incorporate visibility checks. Due to this the tracking can become unstable if large head rotations are involved, the head pose is badly estimated or if the AAM tracking fails, such that features are not traced accurately and begin to dither in image space.

8. Conclusions

We presented a 3D tracking procedure for facial expressions. Instead of blending between different facial shapes collected from various persons, we optimize deformations of the individual neutral face such that projected facial feature points match with the feature points of an AAM. Our formulation decouples the facial motion from the individual shapes and allows us to compute a general motion model for facial expressions from a given database, which in turn can be used to track new persons not contained in the database. In future work it would be interesting to incorporate this model into a stereo system and to track additional details of the facial activity and geometry.

References

[ARL*09] ALEXANDER O., ROGERS M., LAMBETH W., CHIANG M., DEBEVEC P.: The Digital Emily project: photoreal facial modeling and animation. In *ACM SIGGRAPH Courses* (NY, USA, 2009), ACM, pp. 12:1–12:15. 2

[BBA*07] BICKEL B., BOTSCH M., ANGST R., MATUSIK W., OTADUY M., PFISTER H., GROSS M.: Multi-scale capture of facial geometry and motion. *TOG 26*, 3 (2007), 33. 2

[BHPS10] BRADLEY D., HEIDRICH W., POPA T., SHEFFER A.: High resolution passive facial performance capture. *TOG 29*, 4 (2010). 2

[Bil98] BILMES J.: *A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models*. Tech. rep., 1998. 6

[BPL*05] BORSHUKOV G., PIPONI D., LARSEN O., LEWIS J. P., TEMPELAAR-LIETZ C.: Universal capture - image-based facial animation for "The Matrix Reloaded". In *ACM SIGGRAPH Courses* (2005), ACM. 2

[BSPG06] BOTSCH M., SUMNER R., PAULY M., GROSS M.: Deformation transfer for detail-preserving surface editing. In *Vision, Modeling and Visualization* (2006), pp. 357–364. 3, 4

[BV99] BLANZ V., VETTER T.: A Morphable Model for the Synthesis of 3D Faces. In *Siggraph 1999* (1999), Rockwood A., (Ed.), pp. 187–194. 1, 2, 4

[CB02] CHUANG E., BREGLER C.: *Performance driven facial animation using blendshape interpolation*. Tech. rep., Stanford University, 2002. 3

[CBK*06] CURIO C., BREIDT M., KLEINER Q. C. V. M., GIESE M. A., BÜLTHOFF H. H.: Semantic 3D motion retargeting for facial animation. In *Applied perception in graphics and visualization* (USA, 2006), pp. 77–84. 2

[CET01] COOTES T. F., EDWARDS G. J., TAYLOR C. J.: Active Appearance Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence 23* (2001), 681–685. 2

[CXH03] CHAI J.-X., XIAO J., HODGINS J.: Vision-based control of 3D facial animation. In *Symposium on Computer animation* (2003), SCA '03, Eurographics Association, pp. 193–206. 3

[DM96] DECARLO D., METAXAS D.: The Integration of Optical Flow and Deformable Models with Applications to Human Face Shape and Motion Estimation. In *CVPR* (Washington, DC, USA, 1996), CVPR '96, IEEE Computer Society. 2

[EF78] EKMAN P., FRIESEN W. V.: Manual for the Facial Action Coding System. *Consulting Psychology Press* (1978). 3

[FP09] FURUKAWA Y., PONCE J.: Dense 3D Motion Capture for Human Faces. In *Proc. CVPR* (2009). 2

[HVB*07] HERNÁNDEZ C., VOGIATZIS G., BROSTOW G. J., STENGER B., CIPOLLA R.: Non-Rigid Photometric Stereo with Colored Lights. In *Intl. Conf. on Comp. Vision* (2007). 2

[LO05] LIN I.-C., OUHYOUNG M.: Mirror MoCap: Automatic and efficient capture of dense 3D facial motion parameters from video. *The Visual Computer 21*, 6 (2005), 355–372. 2

[LRF93] LI H., ROIVAINEN P., FORCHEIMER R.: 3-D Motion Estimation in Model-Based Facial Image Coding. *IEEE Transactions on Pattern Analysis and Machine Intelligence 15* (1993), 545–555. 2

[LS08] LI R., SCLAROFF S.: Multi-scale 3D scene flow from binocular stereo sequences. *CVIU 110*, 1 (2008), 75–90. 2

[MB04] MATTHEWS I., BAKER S.: Active Appearance Models Revisited. *Int. J. Comput. Vision 60*, 2 (2004), 135–164. 2, 4, 6

[MHP*07] MA W.-C., HAWKINS T., PEERS P., CHABERT C.-F., WEISS M., DEBEVEC P. E.: Rapid Acquisition of Specular and Diffuse Normal Maps from Polarized Spherical Gradient Illumination. In *Rendering Techniques* (2007), Eurographics Association, pp. 183–194. 2

[PL06] PIGHIN F., LEWIS J.: Performance-driven facial animation. In *ACM SIGGRAPH 2006 Courses* (2006), ACM. 3

[PSS99] PIGHIN F. H., SZELISKI R., SALESIN D.: Resynthesizing Facial Animation through 3D Model-based Tracking. In *ICCV* (1999), pp. 143–150. 3

[PTVF07] PRESS W. H., TEUKOLSKY S. A., VETTERLING W. T., FLANNERY B. P.: *Numerical Recipes 3rd Edition: The Art of Scientific Computing*, 3 ed. Cambridge University Press, NY, USA, 2007. 4, 6

[SHK11] SIBBING D., HABBECKE M., KOBBELT L.: Markerless reconstruction and synthesis of dynamic facial expressions. *CVIU 115*, 5 (2011), 668–680. 3

[Sil86] SILVERMAN B.: *Density Estimation for Statistics and Data Analysis*. Monographs on Statistics and Applied Probability. Taylor & Francis, 1986. 6

[SNF05] SIFAKIS E., NEVEROV I., FEDKIW R.: Automatic determination of facial muscle activations from sparse motion capture marker data. *TOG 24*, 3 (2005), 417–425. 2

[VBR*99] VEDULA S., BAKER S., RANDER P., COLLINS R., KANADE T.: Three-Dimensional Scene Flow. In *Proc. ICCV* (September 1999), vol. 2, pp. 722–729. 2

[WBLP11] WEISE T., BOUAZIZ S., LI H., PAULY M.: Realtime Performance-based Facial Animation. *ACM Transactions on Graphics (SIGGRAPH 2011)* (2011). 2, 5

[WL90] WILLIAMS, LANCE: Performance-driven facial animation. In *Computer graphics and interactive techniques* (USA, 1990), pp. 235–242. 2

[WLVP09] WEISE T., LI H., VAN GOOL L., PAULY M.: Face/Off: live facial puppetry. In *Symposium on Computer Animation* (2009), ACM, pp. 7–16. 2

[XBMK04] XIAO J., BAKER S., MATTHEWS I., KANADE T.: Real-Time Combined 2D+3D Active Appearance Models. In *CVPR* (June 2004), vol. 2, pp. 535–542. 2

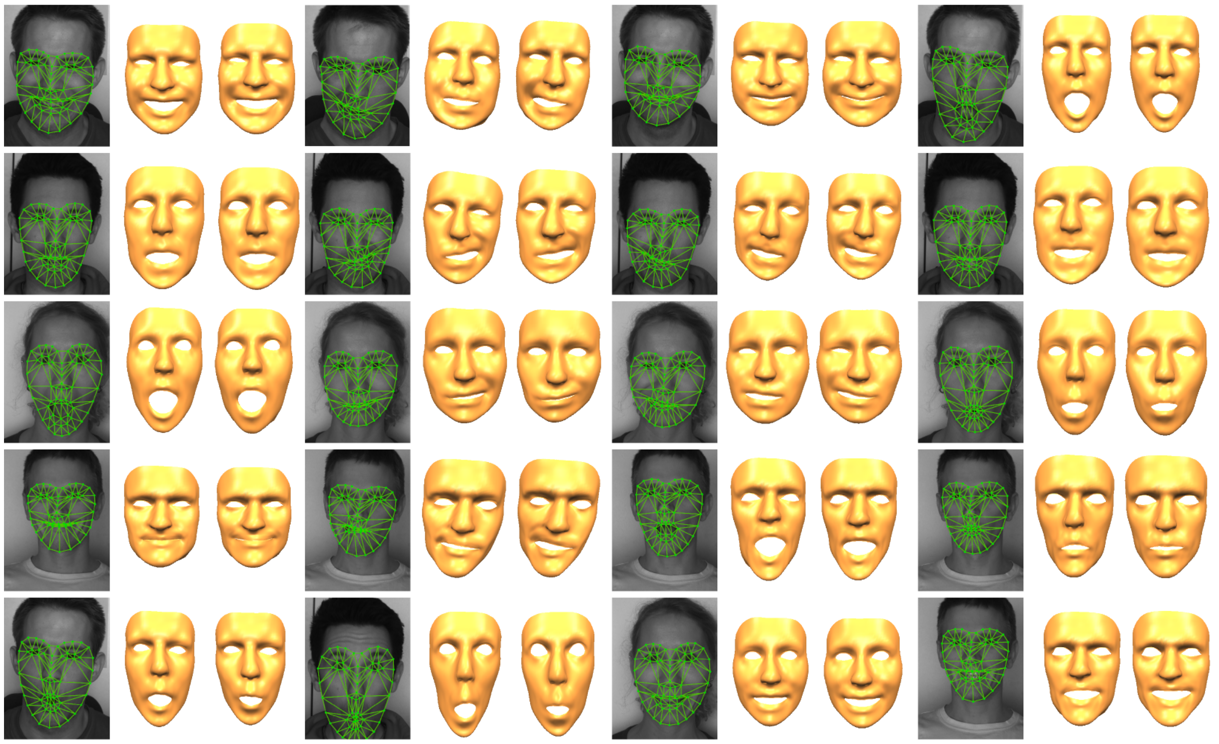


Figure 4: *Results produced with our system. Each block contains an image of the video sequence (left), the result produced with our system (middle) and the corresponding 3D stereo reconstruction as a ground truth comparison. The results of our system are quite similar to the baseline.*

[ZSCS04] ZHANG L., SNAVELY N., CURLESS B., SEITZ S. M.: Spacetime Faces: High-Resolution Capture for Modeling and Animation. In *ACM Annual Conference on Computer Graphics* (2004), pp. 548–558. 2

Appendix

Gradient of $E_{aam}(\mathbf{s},\mathbf{r})$. The gradient of the projected vertex positions can be computed as

$$\frac{\partial \mathbf{q}_j}{\partial x} = \left[\begin{array}{c} \frac{\frac{\partial a}{\partial x}\cdot c - \frac{\partial c}{\partial x}\cdot a}{c^2} \\ \frac{\frac{\partial b}{\partial x}\cdot c - \frac{\partial c}{\partial x}\cdot b}{c^2} \end{array} \right]$$

where the derivatives w.r.t. the parameter $\mathbf{r}$ and $\mathbf{s}$ are given by

$$\begin{aligned} \frac{\partial \begin{pmatrix} a \\ b \\ c \end{pmatrix}}{\partial \alpha} &= \mathbf{K}\cdot\mathbf{R}_0\cdot\frac{\partial \mathbf{R}_\alpha}{\partial \alpha}\cdot\mathbf{R}_\beta\cdot\mathbf{R}_\gamma\cdot\mathbf{p}_j \\ \frac{\partial \begin{pmatrix} a \\ b \\ c \end{pmatrix}}{\partial s_i} &= \mathbf{K}\cdot\mathbf{R}_0\cdot\mathbf{R}_\alpha\cdot\mathbf{R}_\beta\cdot\mathbf{R}_\gamma\cdot\begin{pmatrix} \mathbf{M}_{DS}(j,3\cdot i) \\ \mathbf{M}_{DS}(j,3\cdot i+1) \\ \mathbf{M}_{DS}(j,3\cdot i+2) \end{pmatrix} \end{aligned}$$

Finally the gradient of the first energy term is

$$\frac{\partial E}{\partial \mathbf{x}} = \lambda_{aam}\cdot\sum_{i=1}^{|a|} 2\|\mathbf{q}_i - \mathbf{a}_i\|\cdot\frac{\partial \mathbf{q}_j}{\partial \mathbf{x}}$$

Gradient of $E_{rigid}(\mathbf{r},\tilde{\mathbf{r}})$. We exemplarily evaluate the gradient of the energy measuring rigid accelerations w.r.t. ω:

$$\frac{\partial E_{rigid}}{\partial \boldsymbol{\omega}} = \frac{2\lambda_{\boldsymbol{\omega}}}{dt^4}(\boldsymbol{\omega} - 2\boldsymbol{\omega}_{f-1} + \boldsymbol{\omega}_{f-2})$$

Gradient of $E_{shape}(\mathbf{s},\tilde{\mathbf{s}})$. Similar to separating the snippet coefficients $\mathbf{y}\in\mathbb{R}^{k\cdot l}$ into a part $\tilde{\mathbf{s}}$ (previous frames) and a part $\mathbf{s}$ (actual frame) we similarly divide the matrix $\mathcal{A} = \begin{bmatrix} \mathcal{A}_{\tilde{\mathbf{s}}} \\ \mathcal{A}_{\mathbf{s}} \end{bmatrix} \in \mathbb{R}^{k\cdot l\times d}$ and the vector $\bar{\mathbf{y}} = \begin{bmatrix} \bar{\mathbf{y}}_{\tilde{\mathbf{s}}} \\ \bar{\mathbf{y}}_{\mathbf{s}} \end{bmatrix} \in \mathbb{R}^{k\cdot l}$, where $\mathcal{A}_{\mathbf{s}}\in\mathbb{R}^{l\times d}$ and $\bar{\mathbf{y}}_{\mathbf{s}}\in\mathbb{R}^{l}$. Then

$$\begin{aligned} f(\mathbf{y}) &= (\mathbf{x}-\boldsymbol{\mu}_i)^T\boldsymbol{\Sigma}_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i) \\ &= \mathbf{s}^T\boldsymbol{E}_i\mathbf{s} + 2\cdot\mathbf{s}^T\boldsymbol{F}_i + \boldsymbol{b}_i \end{aligned}$$

where

$$\begin{aligned} \boldsymbol{E}_i &= \mathcal{A}_{\mathbf{s}}\cdot\boldsymbol{\Sigma}_i^{-1}\cdot\mathcal{A}_{\mathbf{s}}^T \in \mathbb{R}^{l\times l} \\ \boldsymbol{F}_i &= \mathcal{A}_{\mathbf{s}}\cdot\boldsymbol{\Sigma}_i^{-1}\cdot\boldsymbol{\lambda}_i \in \mathbb{R}^{l} \\ \boldsymbol{b}_i &= \boldsymbol{\lambda}_i^T\cdot\boldsymbol{\Sigma}_i^{-1}\cdot\boldsymbol{\lambda}_i \in \mathbb{R} \quad \text{and} \\ \boldsymbol{\lambda}_i &= \mathcal{A}_{\tilde{\mathbf{s}}}^T\cdot\tilde{\mathbf{s}} + \mathcal{A}^T\cdot\bar{\mathbf{y}} - \boldsymbol{\mu}_i \in \mathbb{R}^d \end{aligned}$$

Then the derivative of the last energy term becomes

$$\frac{E_{shape}(\mathbf{s},\tilde{\mathbf{s}})}{\partial \mathbf{s}} = \frac{1}{p_{GM}(\mathbf{x})}\cdot\sum_i \pi_i \frac{e^{-\frac{1}{2}f(\mathbf{y})}}{\sqrt{(2\pi)^d|\boldsymbol{\Sigma}_i|}}\cdot(\mathbf{E}_i\mathbf{s} + \mathbf{F_i}) \quad .$$