Markerless Reconstruction and Synthesis of Dynamic Facial Expressions

Dominik Sibbing[a], Martin Habbecke[a], Leif Kobbelt[a]

[a]*RWTH Aachen University, Computer Graphics and Multimedia, Germany*

Abstract

In this paper we combine methods from the field of computer vision with surface editing techniques to generate animated faces, which are all in full correspondence to each other. The inputs for our system are synchronized video streams from multiple cameras. The system produces a sequence of triangle meshes with fixed connectivity, representing the dynamics of the captured face. By carefully taking all requirements and characteristics into account we decided for the proposed system design: We deform an initial face template using movements estimated from the video streams. To increase the robustness of the reconstruction, we use a morphable model as a shape prior to initialize a surfel fitting technique which is able to precisely capture face shapes not included in the morphable model. In the deformation stage, we use a 2D mesh based tracking approach to establish correspondences over time. We then reconstruct positions in 3D using the same surfel fitting technique, and finally use the reconstructed points to robustly deform the initially reconstructed face. We demonstrate the applicability of the tracked face template for automatic modeling and show how to use deformation transfer to attenuate expressions, blend expressions or how to build a statistical model, similar to a morphable model, on the dynamic movements.

Keywords: Face Reconstruction, Mesh Deformation, Face Animation

---

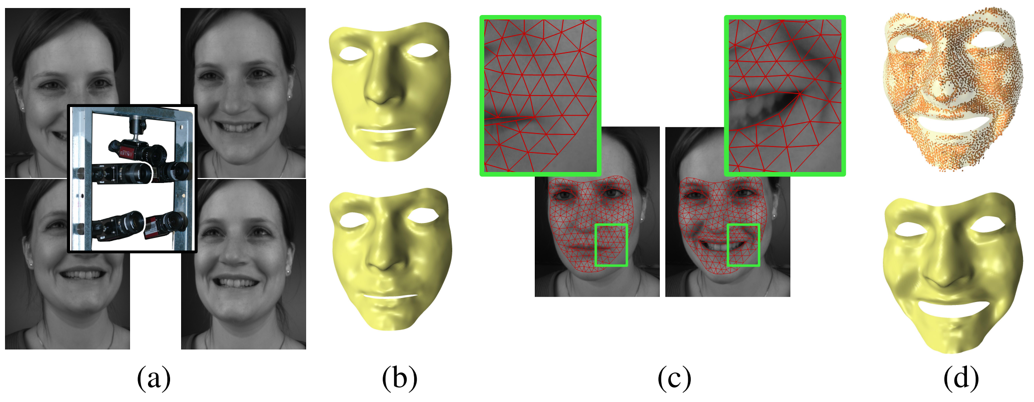

Figure 1: Workflow to reconstruct a dynamic face. (a) Views from different directions and camera rig. (b) Top: Morphable model after optimizing rigid transformation and shape parameters. Bottom: Reconstruction of the neutral face from the first image. (c) The *2D mesh tracking* establishes temporal correspondences between the frames. (d) Top: The *surfel fitting* produces a point cloud, which may contains some holes. Bottom: Result of the *3D mesh tracking*: Successfully reconstructed surfels define constraints for a non-rigid deformation of the initial face template.

1. Introduction

The dense motion capture of facial movements is an important part to generate data driven facial animations. The acquired motion data can be used to create animations for movies, computer games, or humanoid avatars which can be utilized in scientific as well as commercial applications. Standard face tracker based motion capture systems often record only sparse temporal and spatial data and need to place special markers onto the face of an actor. In many applications like psychological studies, where the original video should be as natural as possible, it is not possible to use markers or any artificial texture to ease the reconstruction process (paint, structured light, *etc.*). Modern multi-camera and computer systems offer the possibility to acquire and analyze large amounts of image data, which makes the dense reconstruction and example based synthesis of facial movements possible.

In this paper, we exploit methods from computer vision and mesh editing to compute a dense motion field for facial animations from synchronized video streams. The output of our system is a motion field represented by a predefined

Email addresses: `sibbing@cs.rwth-aachen.de` (Dominik Sibbing), `habbecke@cs.rwth-aachen.de` (Martin Habbecke), `kobbelt@cs.rwth-aachen.de` (Leif Kobbelt)

face template whose vertices move in time according to the underlying scene flow. In many applications such as the re-targeting of facial movements, expression blending or statistical analysis of motion data, it is essential to establish correspondences between vertices not only from frame to frame, but also between different data sets. Since we use a predefined face template which is fitted to an individual face, we immediately obtain the correspondences between all acquired reconstructions.

To increase robustness, we use the surface from the previous frame to initialize the 3D reconstruction for the current frame. In order to robustly obtain the reconstruction in the first frame we use a simple morphable model, with only a few shape parameters, just to initialize the face template for the first frame of the video stream. Tracking in 2D to establish correspondences in time is done by an optical flow like approach, while our 3D reconstruction is a variant of the *surfel fitting* approach [1].

In many applications, *e.g.*, the design of stimulus material for psychological studies, it is of interest to blend existing facial movements or to attenuate facial expressions. We describe how to use deformation transfer [2] as a key technique in order to transfer, blend and attenuate expressions. Furthermore we introduce a new statistical model, computed from the time varying deformations of the triangles of different face templates, representing the average of specific expressions and its main deviation modes.

The main contributions of this paper are

- A simple and easy to implement formulation of the *surfel fitting* [1] approach to reconstruct 3D geometry.

- A carefully designed reconstruction system which reconstructs facial movements from purely video data at high frame rates without using marker or artificial textures and which guarantees full temporal and inter-subject correspondences.

- A set of possible applications using the captured data to produce new facial animations by linearly combining existing ones.

- A statistical analysis of the temporal deformation data (instead of just static geometry) and a prototypical statistical model describing average dynamic expressions and their main deviation modes.

1.1. Related Work

Our system is related to previous work in the areas of facial modeling, face capture and facial animation.

A common technique to generate (caricatured) facial movements for movies and computer games is Free Form Deformation (FFD). FFD provides a framework which allows artists to drag vertices of a cage to intuitively deform the space inside the cage and thus the underlying geometry. Sophisticated methods, *e.g.*, introduced in [3, 4, 5], compute mappings in 3D space which do not induce large distortions (such as volumetric shrinking). Using such tools, artists can be very creative when producing animation sequences, but the major disadvantage is that this is very time consuming.

One way to simulate realistic movements of a human face is to use physically based methods, which usually rebuild some anatomical features of the human head with the aim of mimicking natural movements. Waters [6] simulates facial muscle contractions by abstracting the facial action units originally introduced by Ekman and Friesen [7]. However, this work only uses a few muscles to reproduce basic human emotions. Similarly, Lee *et al.* [8] build an anatomically accurate physically based head model with tissue, skull and synthetic muscles, which are used to deform the tissue to produce facial expressions. Kähler *et al.* [9] use a similar model to perform real-time deformations based on anthropometrically meaningful landmarks. Their method is also capable of simulating aging. Sifakis *et al.* [10] uses finite element methods to deform the synthetic tissues around a skull model. It also uses a set of sparse surface landmarks to track facial movements with a motion capture system.

The major problem with all biomechanical models is that it is quite difficult to build them correctly, because our anatomical knowledge about human skin, muscles, and bone structures is still limited. Thus, models sometimes require extensive tuning to produce a realistic output. Our animation technique is purely data driven and does not require this special parameter tuning.

Most data driven facial animation systems track special markers placed on an actors face [11, 10, 12, 13]. The so captured trajectories typically represent the movements of the face in a very sparse way. To improve geometric details, Bickel [12] added wrinkles to a facial base mesh, which is deformed by the motion capture data using a shell based mesh deformation method. In many applications, *e.g.*, the aforementioned psychological studies where the original videos should appear as natural as possible, markers are forbidden to use. This motioned us to focus on a markerless capture system for facial animations. Zhang *et al.* [14] reformulated space-time stereo as a global optimization problem to compute a time varying disparity map between two image sequences obtained by a structured light system. Then they fit a template mesh to the time varying depth maps and use optical flow conditions to maintain temporal correspondences in a markerless setup. Since we use pure video data instead of structured light scanners we deal with a different setup, which works at potentially higher frame rates. In the animation synthesis part the authors produce compelling facial expressions and data driven animations. However, for these steps only one input sequence of a specific subject is considered, while we also consider the blending of motion data from different subjects and an inter-subject analysis of the dynamics of faces. Ma *et al.* [15] enrich sparse motion capture data with fine geometric details by using an active sensing method. In their setup they use structured light scanning and photometric stereo to capture wrinkles and fine facial features, while motion capture markers, which make this also a marker based method, are used to track large scale deformations and to establish correspondences between faces. In a training phase polynomial displacement maps are computed to represent medium-frequency facial deformations and high-frequency facial details and used in a synthesis phase to produce new animation sequences. Since our markerless system maintains vertex correspondences between faces and through time we can use a rather simple approach, like the one proposed by Botsch *et al.* [2], to transfer (blended) deformations to other faces.

Other active sensing methods were proposed by Hernández *et al.* [16] which use multispectral photometric stereo to compute a dense normal field from untextured surfaces. Weise *et al.* [17] use active illumination based on phase-shift to reconstruct surfaces at high frame rates. A drawback of both approaches is that they are unable to maintain correspondences between vertices in time. In [18] Weise *et al.* improved their system to be able to track a generic face template. These data were used to create an actors face model in an offline process which enables to track and transfer facial movements of this actor in real time to other faces. In our application we also consider the related deformation transfer of Botsch *et al.* [2] as a key technique. Additionally we introduce a statistical expression model by performing a principal component analysis on the time varying deformations of the templates triangles.

Since we do not use active sensing methods [14, 16, 17, 15, 18], which in general need special hardware to project light patterns or colors onto an object in order to ease the reconstruction, we get by with a rather cheap, easy to get (unlike, *e.g.*, the scanner invented by Weise *et al.*, which is currently not purchasable) and simple camera setup, which potentially can produce sequences at higher frame rates.

A famous data driven approach was suggested by Blanz and Vetter [19]. They learn shape and texture parameters for a morphable model by performing a principal component analysis (PCA) on a set of laser scans of human heads. In a pure image based approach, they optimize these parameters to extract static geometry and texture of a human head from a photo. In order to decouple identities, expressions and visemes Vlasic *et al.* [20] use multilinear models. They perform a statistical analysis on the captured data to obtain a multilinear face model. With this model they were able to track facial movements from a monocular video and transfer moods and articulation to another video. Dellepiane *et al.* [21] deform a dummy head to reconstruct the shapes of human heads from images and used them for binaural rendering. Active Appearance Models (AAMs), as in [22, 23], are used to track motion through (multiview) image sequences. As with all morphable models, though, the reconstructions are always restricted to the low-dimensional space spanned by the parametric model, while our reconstruction method produces results not restricted to such parametric spaces, since the shape model is only used for the robust initialization. Moreover, we build a statistical model by analyzing the dynamics of the faces instead of static poses and use this model to synthesize expressions.

Vedula *et al.* introduced the term *dense scene flow* in [24], which was further improved by Li and Sclaroff [25]. In their pure video based approach they reformulate the optical flow problem, find corresponding pixels in time, and use disparity to find correspondences between different views. The extraction of geometric information which could be used for simple visibility tests was not considered. In [26] Furukawa extended their reconstruction approach to track vertices of a mesh reconstructed in the first image of a video stream. In order to ease the 3D stereo reconstruction and to produce compelling results they put additional paint on the faces. We designed our system such that we always have good initial solutions for the 3D reconstruction, thus we do not need to artificially texture human faces. Another difference is that we use a predefined template with fixed topology, which gives us inter-subject correspondences and simplifies deformation transfer and blending. Borshukov *et al.* [27] propose a system that is similar to ours. A scanned model of a neutral expression of an actor is tracked in time. They use optical flow to ensure temporal correspondences and use 3D stereo to triangulate 3D positions of the models vertices. Since they are using a very expensive camera setup (> 100.000 \$) and correct tracking errors in 2D and 3D manually the results look very impressive. There are other expensive commercial solutions to capture facial movements from pure video data like DI3D$^{\text{TM}}$ from Digital Imaging. Recently Bradley *et al.* [28] designed a passive, video based capture system which does not need special markers. They use 7 stereo pairs of high resolution cameras to reconstruct single patches, which are merged into a 3D point cloud. An optical flow based tracking which involves minimal user interaction is used to establish temporal correspondences. With this quite expensive system they are able to create textured animated faces in a high resolution. Our method is low-cost and the tracking part involves no manual interaction. In addition we decided to use a mesh based 2D tracking, which allows to track, *e.g.*, upper and lower lips independently, while in general a simple implementation of optical flow would smooth those contradictory up and down motions, which might cause strong drifting artifacts.

In the area of facial animation there are many different approaches for retargeting facial expressions. Noh *et al.* [29] roughly select correspondences between a source and a target mesh manually and refine them by using radial basis functions. Displacement vectors are rotated into local coordinates of a target mesh and scaled appropriately in order to take local geometric differences into account. Similar to the method proposed by Na *et al.* [30], where a base mesh is fitted by a user to a target mesh and fine geometric details such as wrinkles are transferred to the target mesh in an hierarchical approach, this technique is well suited to transfer vertex displacements to another mesh with a complete different topology. Our capture system ensures the faces to be in full inter-subject as well as temporal correspondence and we can use a straight forward and robust technique to transfer facial deformations by applying the deformation gradient of each source triangle to the target triangle as described in [31, 2]. Moreover, it is possible to blend deformations and compute a statistical expression model as we will show in the applications section.

1.2. Overview

To generate the input for our system, five synchronized cameras are mounted on a rig and calibrated (finding intrinsic and extrinsic camera parameters) to capture the dynamic facial expressions of different subjects. Each of the cameras record images at 30 FPS with a resolution of 640×480 (Figure 1a shows the rig, together with four images from the middle of a sequence). Our system processes the images after the capture, so higher frame rates > 60 FPS are possible if this is supported by the camera hardware.

The first image of each sequence shows the face in its neutral pose. In order to be able to track facial movements we need to reconstruct the face seen in the first frame. Independently optimizing point depth values using *surfel fitting* would produce a point cloud which can contain holes and outliers. We decided to use a simple morphable model to estimate the shape of the face seen in the first frame (see Section 3.1). This drastically increases to robustness of the *surfel fitting* because of the good initial surfel parameters. One requirement of our system is that inter-subject correspondences have to be maintained. This becomes possible by using a face template, containing a fixed number of vertices, which is in a one-to-one correspondence with the vertices of the morphable model. Morphable models are restricted to a space spanned by their training examples. In order to represent more general shapes we additionally non-rigidly deform the resulting model to produce a smooth face template, which is used for all further steps of the pipeline (Figure 1b).

The initial face template is deformed during the whole sequence while correspondences between the vertices of the template and the captured face are maintained. To achieve this, we combine mesh modeling techniques with multiview stereo reconstruction: 2D image-samples, placed in the first image of every view, are tracked over the entire sequence (Figure 1c). In order to establish temporal correspondences between successive frames, we use a 2D mesh based tracking approach. Simple feature tracker like the KLT tracker [32] often have the problem that features slide past each other, since their displacements are optimized independently of their local neighborhood. In the proposed *2D mesh tracking* (Section 3.2) we can control the global smoothness of the produced mapping to prevent foldovers. Shooting rays through an image-sample of the first image hits the initial face template and thereby defines an anchor point in 3D. At each step, the tracked image-samples are reconstructed using the *surfel fitting* approach [1] (top of Figure 1d). Together with its anchor point lying on the surface of the face template, a successfully reconstructed image-sample will provide a constraint which is used in the modeling step to deform the face template (Figure 1d). See Section 3.3 for a more precise description. The proposed modeling step has two advantages: First, since the tracked face template provides good initial solutions the *surfel fitting* becomes much more robust. Second, even if the *surfel fitting* should not succeed the face template can still be deformed using surrounding successfully reconstructed surfels.

Since *surfel fitting* is used to reconstruct the initial face, as well as to track facial movements we will describe this approach in the next section. The advantage of this algorithm is its simplicity since each surfel can be optimized independently from its local neighborhood.

After presenting our reconstruction results, we show the usability of such reconstructed faces by numerous applications in Section 5: The fixed topology allows for automatic placement of eyes, eyelids, lashes or denture. Deformation transfer for triangle meshes [31, 2] is employed to transfer expressions from one subject to another, attenuate an expression or blend expressions from different subjects. In the end we introduce a dynamic face model, where a principal component analysis is used to analyze the dynamic deformation of facial expressions. The resulting statistical model has similarities to the morphable model of [19], but instead of analyzing the point positions of one static shape we consider all deformation gradients of all triangles and frames. In several examples we will illustrate the presented techniques.

2. Surfel Fitting

Our 3D multi-view stereo reconstruction method is a modified version of the *surfel fitting* approach introduced by Habbecke *et al.* [1]. The input for this algorithm is a set of images obtained from calibrated cameras and an initial estimate of a surface element (surfel) defined by a point and a normal. In order to calibrate the cameras, *i.e.*, finding intrinsic parameters (focal length, principal point and distortion) and extrinsic parameters (position and orientation of the cameras), we employ the calibration method proposed by Zhang [33]. One could also use classic vertex or pixel based reconstruction methods, but in our experiments *surfel fitting* proved to be a quite robust and simple approach for multi-view stereo reconstruction.

Surfel fitting uses the surfel's associated plane and the cameras projection matrices to define a homography which naturally maps pixels from a reference image I_r over the 3D plane to a comparison image I_c. Thereby it is superior to ordinary 3D stereo reconstruction which often compare correlation windows of a fixed size in image space, since perspective distortions, induced by the two projection steps (onto the plane and into the comparison image), are handled automatically. It optimizes the parameters of the plane by minimizing differences in pixel intensities between reference and comparison images.

Given the input plane defined by the initial position $\mathbf{p} \in \mathbb{R}^3$ and normal $\mathbf{n} \in \mathbb{R}^3$, a reference image, and a set of comparison images for the plane are defined

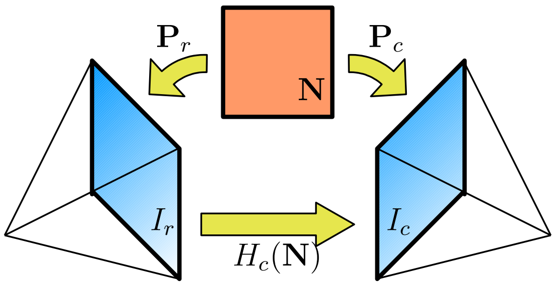

Figure 2: The 3D plane together with the image projection matrices define a homography H which maps image points from I_r to points in the image I_c.

by:

$$\begin{aligned} \mathbf{ref}(\mathbf{p}) &= I_r & r \in \{1, \dots, C\} \\ \mathbf{comp}(\mathbf{p}) &= \{I_{c_1}, \dots, I_{c_k}\} & c_1, \dots c_k \in \{1, \dots, C\} \end{aligned}$$

where C is the total number of views. The reference image can for example be chosen as the image where viewing direction and the vertex normal are closest to parallel. In all the presented steps of the tracking workflow, we have a good initial surface. Thus, the set of comparison images can be obtained by a simple visibility test using the OpenGL z-Buffer.

For simplicity we consider only one comparison image I_c in the following. Let the projection matrices for the reference image and the comparison image be

$$\mathbf{P}_r = [\mathbf{Q}_r | \mathbf{q}_r] \text{ and } \mathbf{P}_c = [\mathbf{Q}_c | \mathbf{q}_c]$$

Without loss of generality, we can transform the scene by a matrix $\mathbf{B}$ such that $\mathbf{P}'_r = \mathbf{P}_r \mathbf{B} = [\mathbf{I}_3 | \mathbf{0}]$. Together with its normal $\mathbf{n}$ we define a plane at point $\mathbf{p}$ as $\mathbf{N}^T = \left[\mathbf{n}^T, \delta\right]$, with $\delta = -\mathbf{p} \cdot \mathbf{n}$. By setting the origin of the transformed coordinate system of the reference camera to the origin of the world coordinate system, such that the plane through $\mathbf{p}$ is not passing through the origin, we can scale the parameters of $\mathbf{N}$ such that $\delta = 1$. This shows that a plane in 3D space has only three degrees of freedom. From now on we will only consider normalized planes $\mathbf{N} = \left[\mathbf{n}^T, 1\right] = [n_1, n_2, n_3, 1]$. A plane determines a homography

$$\begin{aligned} H_c(\mathbf{N}) &= \left(\delta \mathbf{Q}_c - \mathbf{q}_c \mathbf{n}^T\right) \left(\delta \mathbf{Q}_r - \mathbf{q}_r \mathbf{n}^T\right)^{-1} \\ &= \left(\delta \mathbf{Q}'_c - \mathbf{q}'_c \mathbf{n}^T\right) \\ &= \left(\mathbf{Q}'_c - \mathbf{q}'_c \mathbf{n}^T\right) \in \mathbb{R}^{3 \times 3} \end{aligned}$$

which is used to define a warping function $W_c(\mathbf{N};\hat{\mathbf{p}}) : \mathbb{R}^2 \to \mathbb{R}^2$ mapping pixels $\hat{\mathbf{p}} = (u,v)^T$ from the reference image to the comparison image by $W_c(\mathbf{N};\hat{\mathbf{p}}) = (\frac{a}{c}, \frac{b}{c})^T$ (see Figure 2). The vector $(a,b,c)^T$ is computed by the transformation of the homogeneous pixel position $(u,v,1)^T$ with the matrix $H_c(\mathbf{N})$

$$\begin{pmatrix} a \\ b \\ c \end{pmatrix} = H_c(\mathbf{N}) \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} .$$

The goal is then to find new plane parameters which minimize the energy function

$$E_c(\mathbf{N}) = \mathbf{f}^T(\mathbf{N})\mathbf{f}(\mathbf{N}) = \sum_{\hat{\mathbf{p}}\in\Omega} \left(I_c(W_c(\mathbf{N};\hat{\mathbf{p}})) - I_r(\hat{\mathbf{p}})\right)^2$$

where Ω is a square region (we used 15×15 pixels in all our experiments) in the reference image around the projection of the 3D point $\mathbf{p}$. The vector $\mathbf{f}(N) \in \mathbb{R}^{|\Omega|\times 1}$ with components $f_i = I_c(W_c(\mathbf{N};\hat{\mathbf{p}})) - I_r(\hat{\mathbf{p}})$ measures the differences in pixel intensities between the reference and comparison image evaluated at the positions $\hat{\mathbf{p}}$ and $W_c(\mathbf{N};\hat{\mathbf{p}})$ respectively.

In order to find optimal plane parameters minimizing the energy function $E(\mathbf{N})$, we employ the Levenberg-Marquard algorithm [34]. For this we need to calculate the Jacobian $\mathbf{J} = \frac{\partial \mathbf{f}}{\partial \mathbf{N}} \in \mathbb{R}^{|\Omega|\times 3}$ which is the derivative of each component of $\mathbf{f}$ with respect to N:

$$\frac{\partial f_i(\mathbf{N})}{\partial \mathbf{N}} = \nabla I_c \frac{\partial W_c(\mathbf{N};\hat{\mathbf{p}})}{\partial \mathbf{N}} \in \mathbb{R}^{1\times 3},$$

where ∇I_c is the image gradient at the warped pixel position $W_c(\mathbf{N};\hat{\mathbf{p}})$. Employing the quotient rule, the derivative of the warping function with respect to $\mathbf{N}$ can be expressed by

$$\frac{\partial W_c(\mathbf{N};\hat{\mathbf{p}})}{\partial \mathbf{N}} = \begin{bmatrix} \frac{\frac{\partial a}{\partial n_1}\cdot c - a\cdot\frac{\partial c}{\partial n_1}}{c^2} & \frac{\frac{\partial a}{\partial n_2}\cdot c - a\cdot\frac{\partial c}{\partial n_2}}{c^2} & \frac{\frac{\partial a}{\partial n_3}\cdot c - a\cdot\frac{\partial c}{\partial n_3}}{c^2} \\ \frac{\frac{\partial b}{\partial n_1}\cdot c - b\cdot\frac{\partial c}{\partial n_1}}{c^2} & \frac{\frac{\partial b}{\partial n_2}\cdot c - b\cdot\frac{\partial c}{\partial n_2}}{c^2} & \frac{\frac{\partial b}{\partial n_3}\cdot c - b\cdot\frac{\partial c}{\partial n_3}}{c^2} \end{bmatrix} \in \mathbb{R}^{2\times 3}$$

where $\frac{\partial(a,b,c)^T}{\partial n_1} = -\mathbf{q}_c' u$, $\frac{\partial(a,b,c)^T}{\partial n_2} = -\mathbf{q}_c' v$ and $\frac{\partial(a,b,c)^T}{\partial n_3} = -\mathbf{q}_c'$. Following the Levenberg-Marquard algorithm we use the first order Taylor expansion to linearize

$$\mathbf{f}(\mathbf{N} + \Delta\mathbf{N}) \approx \mathbf{f}(\mathbf{N}) + \mathbf{J}\Delta\mathbf{N}$$

and find an update $\Delta\mathbf{N}$ which minimizes $E_c(\mathbf{N} + \Delta\mathbf{N})$ by solving the augmented linear system

$$(\mathbf{J}^T\mathbf{J} + \lambda\mathbf{I}_3)\Delta\mathbf{N} = -\mathbf{J}^T\mathbf{f}(\mathbf{N}),$$

where $\mathbf{I}_3$ is the 3×3 identity matrix and λ is a damping factor which controls the convergence of the optimization: An update $\Delta\mathbf{N}$ is accepted if it leads to a decrease of the error function, *i.e.*, $E_c(\mathbf{N} + \Delta\mathbf{N}) < E_c(\mathbf{N})$. In this case the damping factor λ is divided by 10. We increase λ by a factor of 10 if the update would lead to an increased error value and reject the update $\Delta\mathbf{N}$. Initially the damping factor is set to $\lambda = 10^{-3}$. For stability reasons we normalize the pixel intensities within the small image regions by subtracting the average intensity of that region.

After minimizing the energy function, the 3D position can be obtained by shooting a ray through the center of Ω and computing the intersection with the optimized plane. Occasionally, due to noise in the images or badly textured parts in human faces, this process does not succeed at every vertex. If the plane equation is numerically ill-conditioned, the Levenberg-Marquard algorithm did not converge or if the vertex is only visible in less than 2 cameras, we discard the result. We also use a histogram based discarding criterion: If the final error, is among the 20% largest errors we discard the surfel.

3. Workflow to reconstruct a dynamic face

In order to reconstruct the dynamics of a face, the first step is to fit the face template to the individual geometry of the first frame. Then, the *2D mesh tracking* establishes temporal correspondence between pixels of successive frames by tracking image-samples distributed over regions of the first frame. Finally, 3D reconstructions of these image-samples are used as constraints to deform the face template and thereby capture the movement.

3.1. Initialization of the face template

Reconstructing a human face by just using *surfel fitting* would produce a point cloud probably containing holes and outliers. For the first frame of the video stream we overcome this by using a morphable model as a shape prior to reconstruct the face. Our face template contains a fixed number of n vertices ($\sim 8K$) and is in a one-to-one correspondence with the vertices of the morphable model (described in the next section). The basic appearance of a neutral face can be changed by adjusting the shape parameters of this model. We initialize the morphable model by fitting it to a set of user defined points. Then the face is automatically tracked in 3D by employing the temporal correspondences, established by *mesh tracking*, and using *surfel fitting* to reconstruct 3D geometry. In these later steps we do not use the morphable model anymore to predict some appearance of the face, instead we take the surface generated in the previous frame as initial solution for the next frame.

Morphable model. The morphable model we use is similar to the one introduced by Blanz and Vetter [19]. To generate it, we laser scanned about 50 faces in a neutral expression and established correspondences. By adjusting shape parameters α_i we can approximate each face of the database as a weighted sum of eigenfaces $\mathbf{m}_i \in \mathbb{R}^n$ added to an average face $\bar{\mathbf{M}} \in \mathbb{R}^n$

$$\mathbf{M} = \bar{\mathbf{M}} + \mathbf{E} \cdot \alpha,$$

where the columns of $\mathbf{E} \in \mathbb{R}^{n \times k}$ store the most significant eigenfaces $\mathbf{m}_i$. The small number of k eigenfaces are extracted by performing PCA on the laser scanned face data. If the database contains K faces, PCA extracts $K - 1$ eigenfaces, describing the main deviations from the average face. By excluding eigenfaces with small eigenvalues, the dimensionality of the face space is reduced to a small number $k < K$, while keeping the important details. To neglect high frequencies and to obtain smooth surfaces, we set $k = 15$ in all our experiments.

Initial transformation and shape. For a rough estimate of the rigid translation and rotation w.r.t. the coordinate system of the cameras as well as the shape parameters, we use a few user defined points like the corners of eyes and lips. Defining these features in at least two views allows us to triangulate the 3D location of those features. We assume the user defines a very sparse set of L feature points denoted by $\mathbf{U} = (u_{i_1}, \ldots, u_{i_L})^T$. We denote the corresponding points of the morphable model as follows

$$\begin{aligned} \mathbf{M}' &= [M_{i_1}, \ldots, M_{i_L}]^T \\ &= \bar{\mathbf{M}}' + \mathbf{E}' \cdot \alpha, \end{aligned}$$

where $\bar{\mathbf{M}}'$ and $\mathbf{E}'$ contain just the entries and rows of $\bar{\mathbf{M}}$ and $\mathbf{E}$ corresponding to $\mathbf{U}$. We alternately optimize the rigid transformation and the shape parameters of the morphable model. To compute the model's translation and rotation, the method of Iterative Closest Points [35], which minimizes the squared distances between user defined points and corresponding model points, is used. For this one needs at least 3 corresponding point pairs. In what follows, $\mathbf{M}'$ denotes the rigidly transformed morphable model.

After optimizing the rigid transformation, the new shape parameters can be obtained by minimizing the function

$$\begin{aligned} E &= E_{Shape} + \lambda E_{Ave} \\ &= (\mathbf{M}' - \mathbf{U})^2 + \lambda \cdot \alpha^T \mathbf{D} \alpha, \end{aligned}$$

where $\mathbf{D} \in \mathbb{R}^{k \times k}$ is a diagonal matrix with entries $\mathbf{D}_{i,i} = \frac{1}{\sigma_i^2}$, and σ_i is the eigenvalue of the eigenface $\mathbf{m}_i$. This second term with weight λ has a smoothing effect, because faces near the average face have a smaller energy value. Deriving this function w.r.t. the shape parameters α and setting the derivative to zero yields the linear system

$$\left[\mathbf{E}'^T \mathbf{E}' + \lambda \mathbf{D}\right] \alpha = \mathbf{E}'^T \left(\mathbf{U} - \bar{\mathbf{M}}'\right)$$

In each iteration after optimizing the rigid transformation, λ is decreased by some constant amount, such that the morphable model slowly approaches the user defined points. Since the system is augmented with the diagonal matrix $\mathbf{D}$, the shape optimization would recover the average face if the user did not specify any 3D points. But to be able to find a rigid transformation and in order to recover a good initial solution for subsequent steps we always select 4 points at the eyes and 4 points on the lips.

Improvement of transformation and shape. In the next step, we run the *surfel fitting* algorithm to calculate new depth values for the vertices of the face model. To do this for each vertex, a reference image is defined as that image with viewing direction most parallel to the vertex normal. The comparison images are obtained from a visibility test. The resulting point cloud, possibly containing some holes, is used to augment the user defined feature points $\mathbf{U}$. For each new point $u \in \mathbf{U}$, a corresponding point is obtained as the point on the face model with minimal distance to u. This results in new pairs $(\mathbf{M}', \mathbf{U})$, which are used to compute new parameters for shape and rigid transformation as described above.

3.2. Mesh Tracking in 2D

The objective of the *2D mesh tracking* is to establish temporal correspondences between successive frames. In our setup we use five video cameras to track

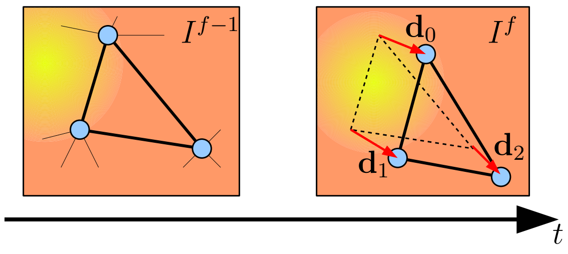

Figure 3: Displacing the triangle vertices by $\mathbf{d}_0, \mathbf{d}_1$ and $\mathbf{d}_2$ yield a similar intensity distribution within the triangle for both successive images I^{f-1} and I^f.

the dynamic facial expression through time. Each view is tracked individually. To do this, we place a 2D mesh in the first image and calculate displacements for every frame such that the mesh tracks the 2D deformation. Since we can control for global smoothness, foldovers can be prevented. Unlike, *e.g.*, [14] we do not use optical flow to establish temporal correspondences. Especially between lips it might be the case that displacement vectors of the flow field point in different directions and thus are suppressed by the smoothing term which keeps the field coherent [36]. In a mesh based approach where lips are separated and allowed to move independently this is less likely to happen.

Initialization. We project the face template, reconstructed as described in Section 3.1, into the first frame of the considered view. The projected mesh is then remeshed by the algorithm presented in [37], such that the new average edge length covers about 25 pixels. We denote the remeshed version of the mesh in the first frame as $\hat{\mathbf{S}}^1$.

Tracking. A view consists of a sequence of images

$$I^1, \ldots, I^{f-1}, I^f, I^{f+1}, \ldots$$

Given two successive images I^f and I^{f+1}, the aim is to find displacements $\mathbf{d}_i = [d_{i,x}, d_{i,y}]^T \in \mathbb{R}^2$ for every vertex of the given shape $\hat{\mathbf{S}}^f$, such that differences in the intensity distribution within each triangle of two successive images are small (see Figure 3). Consider one triangle T of $\hat{\mathbf{S}}^f$. A pixel $\hat{\mathbf{p}} = [x, y]^T \in \mathbb{R}^2$ within this triangle has barycentric coordinates $[\beta_0, \beta_1, \beta_2]$. The barycentric coordinates and the vertex displacements define a linear mapping $\pi : \mathbb{R}^2 \rightarrow \mathbb{R}^2$, which maps $\hat{\mathbf{p}}$ from the undeformed triangle of image I^f to a deformed triangle in the image I^{f+1}. If $I^f(\hat{\mathbf{p}})$ is the intensity function of an image I^f, the minimization function can be stated as

$$E_T = \sum_{\hat{\mathbf{p}} \in T} \left(I^f(\hat{\mathbf{p}}) - I^{f+1}(\pi(\hat{\mathbf{p}})\right)^2$$

If the time between two successive frames is short, the input is already close to the optimal solution and a standard Levenberg-Marquard minimization procedure is suitable to solve for the displacements d_i. Summing these energy functions for all triangles yields the global energy function

$$E_{data} = \sum_{T \in \hat{\mathbf{S}}^f} E_T = \sum_{T \in \hat{\mathbf{S}}^f} \sum_{\hat{\mathbf{p}} \in T} \left(I^f(\hat{\mathbf{p}}) - I^{f+1}(\pi(\hat{\mathbf{p}})\right)^2$$

To ensure a smooth distribution of the displacements, we introduce the additional energy term

$$E_{smooth} = \sum_{i \in V(\hat{\mathbf{S}}^f)} \left(\frac{1}{\omega_i} \sum_{j \in \mathbf{Neigh}_i} \omega_{i,j} \parallel \mathbf{d}_i - \mathbf{d}_j \parallel \right)^2$$

where $V(\mathbf{S}^f)$ is the set of vertex indices of the mesh $\hat{\mathbf{S}}^f$ and $\mathbf{Neigh}_i$ denotes the 1-neighborhood of vertex $\hat{\mathbf{p}}_i$. The standard chordal weights $\omega_{i,j} = \parallel \hat{\mathbf{p}}_i - \hat{\mathbf{p}}_j \parallel^2$, $w_i = \sum_{j \in N_i} \omega_{i,j}$ are used to set up the Laplace system. Putting both terms together, the final energy function is denoted by

$$E = E_{data} + \lambda E_{smooth}$$

where λ controls the smoothness term.

3.3. Mesh Tracking in 3D

As initialization for the *3D mesh tracking* we use again the surface we estimated for the first frame. The objective of the algorithm described in this section is to find a deformation of the face template for every frame, such that the highly detailed movements of the captured face are tracked by the template face. To achieve this, we generate image-samples in every view and track them through time. Using the *surfel fitting* of Section 2, these image-samples are reconstructed in 3D for every frame. Finally, these reconstructed 3D points are used to deform the template mesh. As stated above this increases the robustness of *surfel fitting* and decouples the reconstruction of the dynamics from the independent reconstruction of single image-samples.

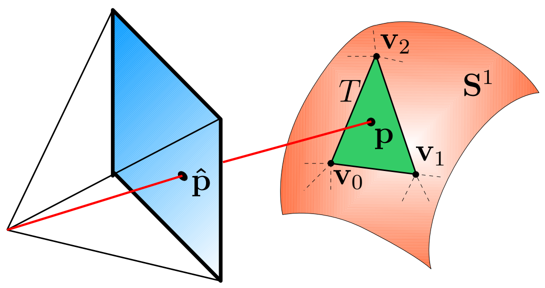

Figure 4: An anchor point of a image-sample $\hat{\mathbf{p}}$ is obtained by the intersection of a ray through $\hat{\mathbf{p}}$ with the initial face template $\mathbf{S}^1$.

Generating image-samples. Running the *2D mesh tracking* described in Section 3.2 on every view of the video sequence produces a 2D triangle mesh $\hat{\mathbf{S}}_c^f$ for each view c and frame f. Supersampling the triangles of the meshes in the first frame of the sequence generates 2D points that have barycentric coordinates w.r.t. the triangle they are placed in. For every view c this yields a set of points $\hat{\mathbf{p}}_{i,c}^1$ for the first frame, where a point can uniquely be identified by its index i and the view c it was put in. The number of image-samples is a user-defined parameter. We usually place 1600 samples in one view to obtain a dense reconstruction. The mapping π, which is defined by the deformation of a 2D mesh from one frame to the following, allows us to displace the image-samples and thereby track them through the whole sequence of a single view. This produces sequences of points $\hat{\mathbf{p}}_{i,c}^f$.

From image-samples to 3D trajectories. Section 3.1 describes how to fit a face template to the first frame of a video stream. Assume the face template to be $\mathbf{S}^1$. For each image-sample $\hat{\mathbf{p}}_{i,c}^1$, we introduce an anchor point on the surface of $\mathbf{S}^1$ by shooting a ray through $\hat{\mathbf{p}}_{i,c}^1$ and determining the intersection with $\mathbf{S}_1$. This intersection is located within a triangle T and has barycentric coordinates $[\gamma_0, \gamma_1, \gamma_2]$ with respect to T (see Figure 4). An image-sample $\hat{\mathbf{p}}_{i,c}^f$ can be reconstructed in 3D by using the *surfel fitting* approach of Section 2. Given a face template $\mathbf{S}^{f-1}$ that was already fitted to frame $f-1$, a good initial solution for the surfel position is obtained by evaluating the linear combination of the triangle vertices of $T \in \mathbf{S}^{f-1}$ weighted with $[\gamma_0, \gamma_1, \gamma_2]$. Further, the normal of this triangle is the initial plane normal for the surfel. As a reference image, we select the view c that the image-sample was initially placed in. Since the fitted template of the frame $f-1$ is already a good approximation, this mesh is well suited for visibility tests to determine the set of (multiple) comparison images. If the *surfel fitting* succeeded, the reconstructed point $\mathbf{p}^f_{i,c} \in \mathbb{R}^3$ is stored in a list denoted by $\mathbf{Succ}(f)$.

Deformation of the template mesh. In order to deform the mesh we treat the $\mathbf{p}^f_{i,c}$ as handles which drag the surface $\mathbf{S}^1$. We define two objective functions. The first function measures the squared distance between the Laplace vectors of $\mathbf{S}^1$ and those of the deformed surface $\mathbf{S}^f$

$$E_L = \sum_{\mathbf{v} \in \mathbf{S}} \parallel \Delta \mathbf{v}^f - \Delta \mathbf{v}^1 \parallel^2$$

Here, Δ denotes the discrete Laplace operator using the cotangent weights evaluated on the surface $\mathbf{S}_1$. The second function penalizes large deviation of the anchor point from the reconstructed point and can be denoted by

$$E_C = \sum_{\mathbf{p} \in \mathbf{Succ}(\mathbf{p})} \parallel \mathbf{p} - \mathbf{anchor}(\mathbf{p}) \parallel^2$$

where the anchor point $\mathbf{anchor}(\mathbf{p}^f_{i,c})$ is calculated by interpolating the vertices of the triangle T associated with $\mathbf{p}^f_{i,c}$ using the precomputed barycentric coordinates:

$$\mathbf{anchor}(\mathbf{p}^f_{i,c}) = \sum_{\mathbf{v}_t \in T} \gamma_t \cdot \mathbf{v}^f_t$$

To obtain the new vertex positions of a mesh $\mathbf{S}^f$, we solve

$$E = E_L + \lambda E_C$$

in the least-squares sense and repeat the whole procedure for the next frame $f+1$.

It is worth mentioning that this procedure can also help improve the estimated surface $\mathbf{S}^1$ of the first frame. At the end of the process described in Section 3.1, a new point cloud can be extracted by *surfel fitting*. For each surfel, we can compute an anchor point as the closest point on $\mathbf{S}^1$ w.r.t. the surfel. These pairs can then be used to deform the face template $\mathbf{S}^1$, as described above. The deformed surface does not lie in the space spanned by the morphable model and is used as the input surface for all subsequent steps of the pipeline.

4. Reconstruction results

We generated all our results using a 2.6Ghz Intel Core i7 CPU. During *2D mesh tracking*, the computation of the 2D displacements for five views of one frame took an average time of 52 seconds. The average time for the *surfel fitting* of one frame, where we optimized about 8K samples from all five views, was 50 seconds, leading to an overall computing time of less than 2 minutes per frame. Since for each surfel we need to sum over all comparison cameras in order to compute its update, the worst case complexity of the surfel fitting is $\mathcal{O}(n{\cdot}C)$ where C is the number of cameras and n the number of surfels to be reconstructed. But this is an upper bound, because we incorporated a visibility test which sometimes removes cameras from the set of comparing cameras.

We collected sequences of different subjects each performing different facial expressions for approximately 2 to 4 seconds. In Figure 5 we present five examples. It shows one of the input images together with the reconstruction of the neutral face (left column). The right column shows the result of the deformation step where the surfels act as handles to deform the neutral face (see the close up images). In all these examples, we left the parameters at a fixed setting.

5. Applications

Since we decided to use a predefined face template with a fixed mesh topology, we obtain meshes for each frame which are in full correspondence to each other. This enables a variety of applications. We demonstrate some of these applications, which are explained in more detail in this section.

5.1. Eliminating rigid transformations

Due to head movements the captured face will certainly show some rigid transformations w.r.t. the first frame. For the applications described in the following sections it is desirable to separate these rigid transformations from the deformations.

Assuming the face does not scale, we compute a translation and rotation of the face in each frame w.r.t. the face shown in the first frame. For this we employ a variant of the algorithm provided by Horn [35]. Note that the correspondences do not need to be calculated since vertex indices of our face template remain constant across frames. In general one can take all vertices into account to calculate the rigid transformations, but due to highly deformable regions on the face (like cheeks, forehead or mouth) we just consider vertices lying on the more rigid nose region. Advantages of this step are that we can easily replace the rigid transformation with different transformations, without changing the deformation itself.

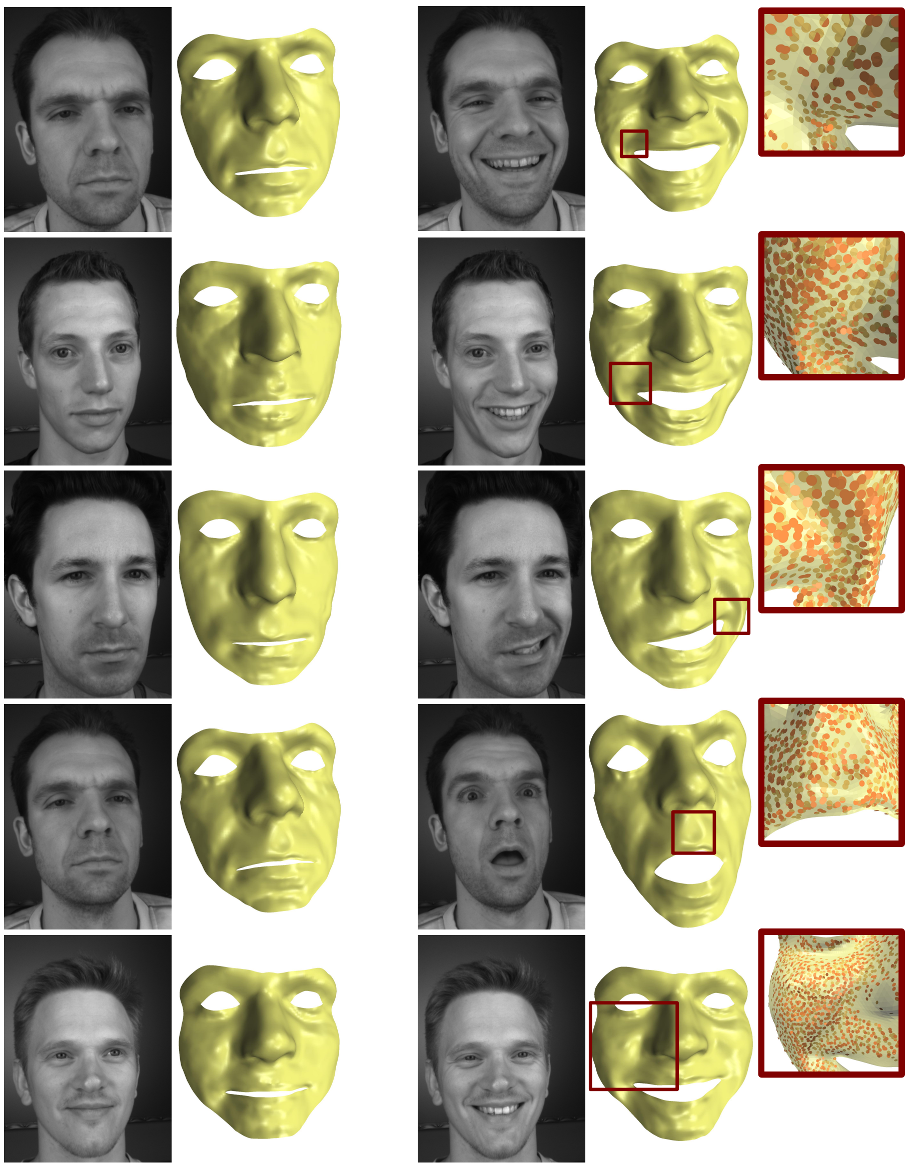

Figure 5: Reconstruction results. The left column shows input images of neutral faces and their reconstructions. The extracted surfels and their anchor points define constraints in a modeling step to deform the surface. This is possible because we established temporal correspondences by using the *2D mesh tracking* algorithm. The deformation and the surfels can be seen in the right column.

5.2. Automatic model enhancement

Since we are able to remove rigid transformations the positions of eyes, bones, *etc*. remain constant across frames. Together with the fixed topology of the mov-

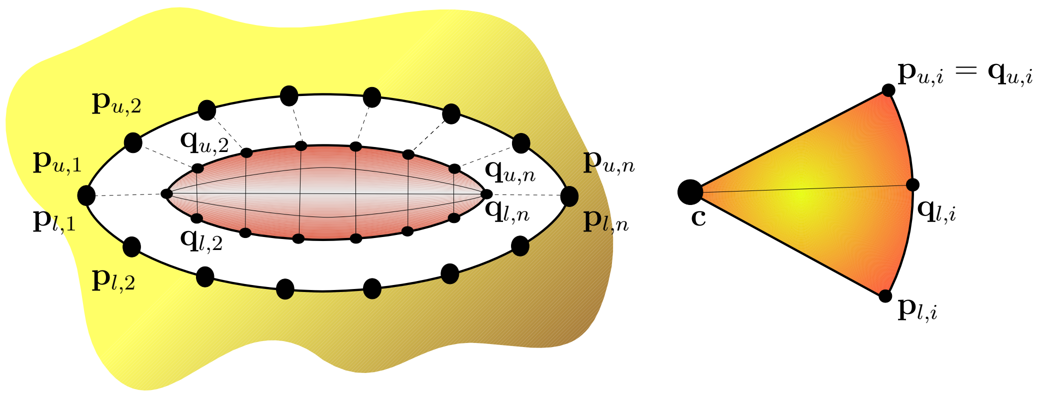

Figure 6: Left: The eyelid and the boundary of the eye have the same amount of vertices, so the eyelid can be stitched to the top boundary. Right: The closing t determines the position of each lower point $\mathbf{q}_{l,i}$ on the eyeball. If $t = 1$ the lid covers the whole eye, *i.e.*, $\mathbf{p}_{l,i} = \mathbf{q}_{l,i}$.

ing face template, this allows for the automatic placement of eyes, eye lids and lashes. In this section we describe a very simple way to model eyes. Note that this does not produce realistic looking eyes, but can be seen as a proof of concept for such an automated modeling procedure, which connects the boundary curve of the eye to lids and lashes.

As depict in Figure 6, the boundary around an eye consists of a lower curve $\{\mathbf{p}_{l,1}, \dots, \mathbf{p}_{l,n}\}$ and an upper curve $\{\mathbf{p}_{u,1}, \dots, \mathbf{p}_{u,n}\}$. We model the eyeball as a simple sphere with a radius $r = 12$mm, which is the average radius of a human eye [38]. The center $\mathbf{c}$ of the eye is placed such that the following function is minimized:

$$F = \sum_{i=l,u} \sum_{j=1}^{n} (\mathbf{c} - \mathbf{p}_{i,j})^2 - r^2$$

subject to the constraints

$$||\mathbf{p}_{i,j} - \mathbf{c}|| \geq r$$

In order to model a lid we created a small polygon mesh with an upper and lower curve containing the vertices $\{\mathbf{q}_{u,1}, \dots, \mathbf{q}_{u,n}\}$ and $\{\mathbf{q}_{l,1}, \dots, \mathbf{q}_{l,n}\}$. This mesh can easily be connected to the upper boundary curve by setting $\mathbf{q}_{u,i} = \mathbf{p}_{u,i}$ (see Figure 6). We compute aperture angles for each pair of vertices on the boundary of the eye as

$$\theta_i = \arccos\left(\frac{(\mathbf{p}_{u,i} - \mathbf{c})^T(\mathbf{p}_{l,i} - \mathbf{c})}{||\mathbf{p}_{u,i} - \mathbf{c}|| \cdot ||\mathbf{p}_{l,i} - \mathbf{c}||}\right)$$

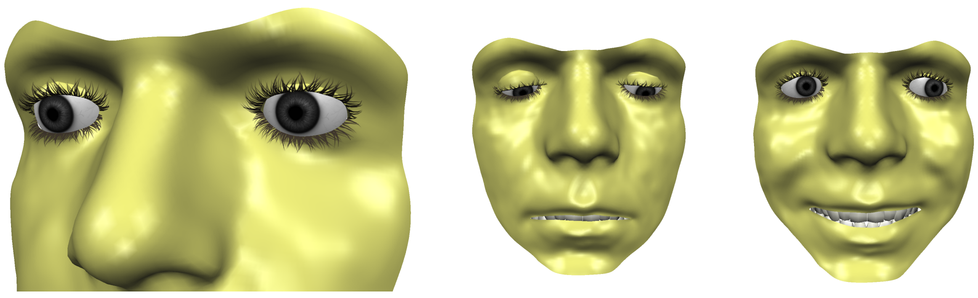
Figure 7: Simple eye model with lid and lashes. The closing of the lid depends on the viewing direction of the eye: The lower vertices of the lid are close to the upper point of the iris. In this example we also modeled teeth.

The positions of the lower points are calculated by interpolating the aperture angle. Let $t \in [0, 1]$ be the parameter controlling the closing of the eye. The lower points of the lid are computed as points lying on the eyeball (Figure 6) such that

$$t \cdot \theta_i = \arccos \left(\frac{(\mathbf{p}_{u,i} - \mathbf{c})^T (\mathbf{q}_{l,i} - \mathbf{c})}{||\mathbf{p}_{u,i} - \mathbf{c}|| \cdot ||\mathbf{q}_{l,i} - \mathbf{c}||} \right)$$

Note that the eye is closed, *i.e.*, $\mathbf{q}_{l,i} = \mathbf{p}_{l,i}$, if $t = 1$. The inner vertices of the lid are smoothly distributed over the eyeball.

Lashes can be attached to the lower points $\mathbf{q}_{l,i}$ and its mesh is textured with a semitransparent image of eyelashes. The result of our automatic modeling example is depicted in Figure 7. Here we implemented a simple look-at function for the eyes which rotates the eyeballs. The opening of the lid is calculated such that the lower points of the lid are close to the upper point of the iris. In this way the lid closes when the eye is looking down. Further modeling steps could be the automatic placement of denture or the integration into a head model.

5.3. Automatic Expression Modeling

In this section we demonstrate versatile applications where expressions, *i.e.*, the deformation of faces, themselves are manipulated. As in Section 5.2, correspondences between different faces and across frames are crucial for these applications. As a basic technique we use a variant of the deformation transfer for triangle meshes which was originally introduced by Sumner [31].

Deformation Transfer. Assume a source triangle mesh $\mathcal{S} = (V, T)$ is given, with $V = \{\mathbf{q}_1, \ldots, \mathbf{q}_n\}$ being the set of vertices and $T = \{t_1, \ldots, t_m\}$ the set of triangles. Further assume this mesh was deformed into another mesh $\mathcal{S} \to \mathcal{S}'$ where new point positions in $\mathcal{S}'$ are denoted by $\{\mathbf{q}'_1, \dots, \mathbf{q}'_n\}$. As described in [2] we can compute a deformation gradient $\mathbf{S}_t \in \mathbb{R}^{3\times3}$ for each triangle t that maps a point from the undeformed to the deformed state. This deformation gradient is given by

$$\mathbf{S}_t = (\mathbf{q}'_1 - \mathbf{q}'_3, \mathbf{q}'_2 - \mathbf{q}'_3, \mathbf{n}') \cdot (\mathbf{q}_1 - \mathbf{q}_3, \mathbf{q}_2 - \mathbf{q}_3, \mathbf{n})^{-1}$$

where $\mathbf{n}$, $\mathbf{n}'$ and $\mathbf{q}_i, \mathbf{q}'_i$ are the normals and vertex positions of an undeformed and a deformed triangle t.

The idea is to find a deformation for a target mesh $\mathcal{T} \to \mathcal{T}'$ such that the deformation gradients $\mathbf{T}_t \in \mathbb{R}^{3\times3}$ of the target mesh match those of the source mesh:

$$\mathbf{T}_t = (\mathbf{p}'_1 - \mathbf{p}'_3, \mathbf{p}'_2 - \mathbf{p}'_3, \mathbf{n}') \cdot (\mathbf{p}_1 - \mathbf{p}_3, \mathbf{p}_2 - \mathbf{p}_3, \mathbf{n})^{-1} = \mathbf{S}_t$$

Here we denote the vertices of $\mathcal{T}$ and $\mathcal{T}'$ as $\{\mathbf{p}_1, \dots, \mathbf{p}_n\}$ and $\{\mathbf{p}'_1, \dots, \mathbf{p}'_n\}$. Note that the indices of the triangles do not change since both meshes are in full correspondence. Equivalent to this we can also require that the transposed of these matrices should be the same:

$$\mathbf{T}_t^T = \mathbf{S}_t^T$$

This has the advantage that we can find an expression for $\mathbf{T}_t$ which is linear in the new point positions $\mathbf{p}'_i$:

$$\tilde{\mathbf{T}}_t^T = \underbrace{\left((\mathbf{p}_1 - \mathbf{p}_3, \mathbf{p}_2 - \mathbf{p}_3, \mathbf{n})^{-1}\right)^T \cdot \begin{pmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \\ 0 & 0 & 0 \end{pmatrix}}_{=:\mathbf{G}_t} \cdot \begin{pmatrix} \mathbf{p'_1}^T \\ \mathbf{p'_2}^T \\ \mathbf{p'_3}^T \end{pmatrix}$$

where $\mathbf{G}_t$ is the constant gradient matrix of the coordinate function of a triangle t of the target mesh $\mathcal{T}$.

In order to compute the deformation transfer for all triangles we want to find new vertex positions $\mathbf{p}'_i$ such the all deformation gradients of the target mesh are equal to the deformation gradients of the source mesh. This is expressed by the global system

$$\begin{pmatrix} \mathbf{G}_1 \\ \vdots \\ \mathbf{G}_m \end{pmatrix} \cdot \begin{pmatrix} \mathbf{p'_1}^T \\ \vdots \\ \mathbf{p'_n}^T \end{pmatrix} = \mathbf{G} \begin{pmatrix} \mathbf{p'_1}^T \\ \vdots \\ \mathbf{p'_n}^T \end{pmatrix} = \mathbf{S} = \begin{pmatrix} \mathbf{S}_1{}^T \\ \vdots \\ \mathbf{S}_m{}^T \end{pmatrix}$$

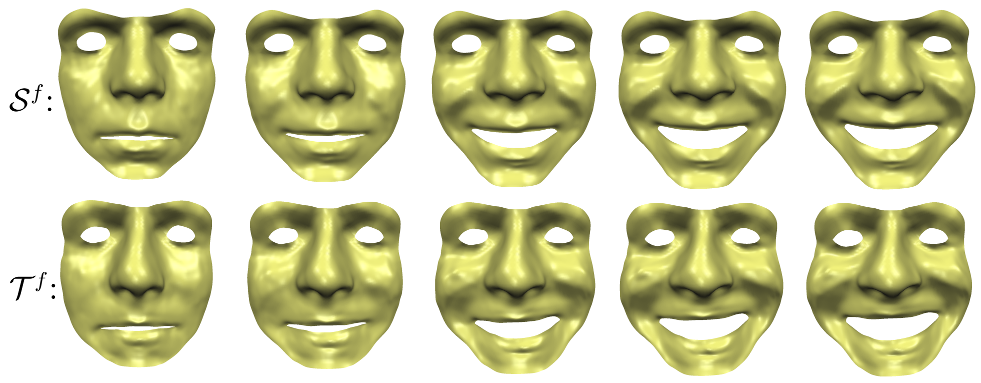

Figure 8: Expression transfer. The smile $\mathcal{S}^1, \ldots, \mathcal{S}^F$ of one face is transferred to another subject to produce a sequence $\mathcal{T}^1, \ldots, \mathcal{T}^F$.

Where $\mathbf{G} \in \mathbb{R}^{3m \times n}$ is the global gradient matrix computed from $\mathcal{T}$ and $\mathbf{S} \in \mathbb{R}^{3m \times 3}$ is a global matrix computed from the deformation $\mathcal{S} \to S'$. Since this system is over determined we solve it in the least squares sense

$$\mathbf{G}^T\mathbf{G} \begin{pmatrix} \mathbf{p_1'}^T \\ \vdots \\ \mathbf{p_n'}^T \end{pmatrix} = \mathbf{G}^T\mathbf{S}$$

This system weights the deformations per triangle in a uniform way which may lead to unwanted distortions. To avoid this, the deformation gradients are weighted by the area of the triangle which finally yields the Poisson equation describing the deformation transfer:

$$\mathbf{G}^T\mathbf{D}\mathbf{G} \begin{pmatrix} \mathbf{p}_1'^T \\ \vdots \\ \mathbf{p}_n'^T \end{pmatrix} = \mathbf{G}^T\mathbf{D}\mathbf{S}$$

where $\mathbf{D} \in \mathbb{R}^{3m \times 3m}$ is the diagonal area matrix computed from all triangles in $\mathcal{T}$. Observe that $\mathbf{G}^T\mathbf{D}\mathbf{G}$ is equal to the standard Laplace matrix of the mesh $\mathcal{T}$ using the cotangent weights and $\mathbf{G}^T\mathbf{D}$ represents the divergence operator.

Expression transfer. Using the formalism of deformation transfer, carry an expression from one face over to another is straight forward. Assume we used the markerless reconstruction technique to generate a sequence of meshes $\mathcal{S}^1, \ldots, \mathcal{S}^F$ from F frames. For each frame $f > 1$ a global deformation gradient matrix

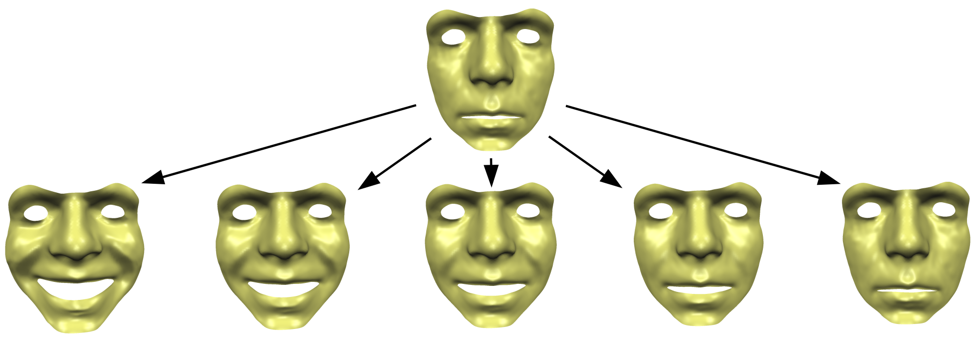
Figure 9: Expression attenuation. The upper image shows the face at time point zero. The images in the lower row show attenuated versions of one of the expressions shown in Figure 8(top) at a particular time point > 0. To generate the images from left to right, we set α to $1, \frac{3}{4}, \frac{1}{2}, \frac{1}{4}$ and 0.

$\mathbf{S}^f \in \mathbb{R}^{3m\times 3}$ can be calculated which describes the deformation $\mathcal{S}^1 \to \mathcal{S}^f$. In order to transfer this expression sequence to another face $\mathcal{T}^1$ we simply compute a new sequence of F deformed meshes $\mathcal{T}^f$ with new vertex positions $\mathbf{p}_i^f$:

$$\begin{pmatrix} \mathbf{p}_1^{f\,T} \\ \vdots \\ \mathbf{p}_n^{f\,T} \end{pmatrix} = \underbrace{(\mathbf{G}^T\mathbf{D}\mathbf{G})^{-1}\mathbf{G}^T\mathbf{D}}_{=:\mathbf{K}(\mathcal{T}^1)}\mathbf{S}^f$$

where $1 < f \leq F$ and $\mathbf{K}(\mathcal{T}^1) \in \mathbb{R}^{n\times 3m}$ is a precalculated matrix only depending on $\mathcal{T}^1$.

In Figure 8 we transferred the smile of one subject to another face.

Expression attenuation. In the previous section we used the deformation gradients S^f from a source sequence to generate a new target sequence showing the same expression as the source sequence. When using the identity matrix $\mathbf{I}_3$ as source deformation gradients per triangle, the target mesh would not change at all. By linearly blending between both possibilities we can attenuate the expression. We start with a face $\mathcal{T}^1$ in neutral position which might be identical to $\mathcal{S}^1$ and compute new vertex positions $\mathbf{p}_i^f$ as

$$\begin{pmatrix} \mathbf{p}_1^{f\,T} \\ \vdots \\ \mathbf{p}_n^{f\,T} \end{pmatrix} = \mathbf{K}(\mathcal{T}^1)\left(\alpha \cdot \begin{pmatrix} \mathbf{S}_1^{f\,T} \\ \vdots \\ \mathbf{S}_m^{f\,T} \end{pmatrix} + (1-\alpha)\cdot \begin{pmatrix} \mathbf{I}_3 \\ \vdots \\ \mathbf{I}_3 \end{pmatrix}\right)$$

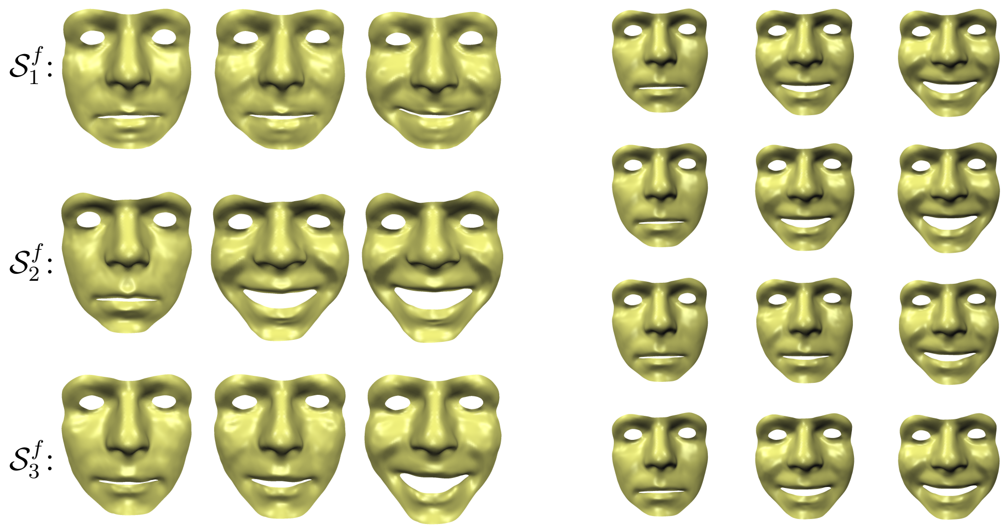

Figure 10: Expression blending. Left: Three 'smile' sequences $(\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3)$ captured by our system. Right: Each row shows images from a blended sequence at the same points in time, where $\mathcal{T}^1$ is the average shape of $(\mathcal{S}_1^1, \mathcal{S}_2^1, \mathcal{S}_3^1)$. The first row shows the average smile of $(\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3)$. The second row shows the average of $(\mathcal{S}_2, \mathcal{S}_3)$. In the third and fourth row one can see the average of $(\mathcal{S}_1, \mathcal{S}_3)$ and $(\mathcal{S}_1, \mathcal{S}_2)$.

where $\alpha \in [0, 1]$. In Figure 9 we show a smile in four different intensities by setting α to 0, $\frac{1}{3}$, $\frac{2}{3}$ and 1.

Expression blending. In this scenario we compute a weighted sum of deformation gradients from different subjects and compose a new sequence from the sum of these weighted gradients.

Besides the geometric correspondence we therefore also need temporal correspondence, *i.e.*, the same number of frames in each sequence. In our experiments the subject performed facial expressions from neutral to an expressed face. Since this took almost the same time for every subject, we immediately get temporal coherence by a simple cut, common in video editing, without interpolating between frames or scaling time.

For expression blending we consider k source sequences $\mathcal{S}_i = \{\mathcal{S}_i^1, \ldots, \mathcal{S}_i^F\}$ where $i \in 1, \ldots, k$. For each frame f in every sequence i we precompute a deformation gradient $\mathbf{S}_i^f \in \mathbb{R}^{3m \times 3}$ which describes the deformation $\mathcal{S}_i^1 \rightarrow \mathcal{S}_i^f$. Starting with a target mesh $\mathcal{T}^1$, which might be a new individual showing a neutral face, a blending between faces like done by a morphable model or simply the first mesh of one of the source sequences, we compute a new sequence from the

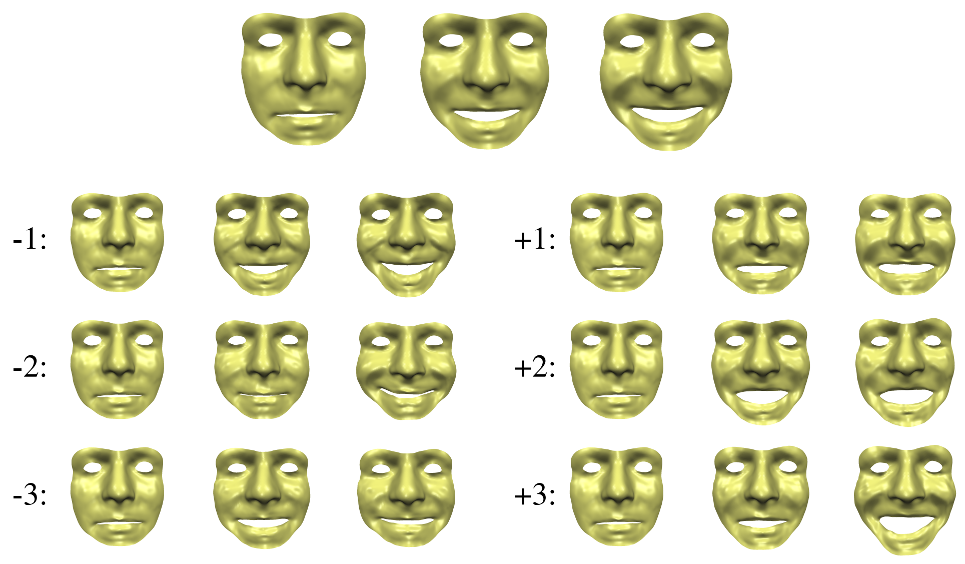

Figure 11: Principal component analysis of the deformation gradients. Top row shows the average smile of 5 subjects. The following rows show the variations produced by adding/subtracting the first, second and third eigenvector. The tree images out of each of the 7 sequences are taken at the same time points.

convex combination of the deformation gradients. This generates new meshes $\mathcal{T}^f$ with new point positions:

$$\begin{pmatrix} \mathbf{p}_1^{f\,T} \\ \vdots \\ \mathbf{p}_n^{f\,T} \end{pmatrix} = \mathbf{K}(\mathcal{T}^1) \left(\sum_{i=1}^{k} w_i \cdot \mathbf{S}_i^f \right)$$

where $\sum_{i=1}^{k} w_i = 1$, $w_i \geq 0$ are some given weights. In Figure 10 we show the result of such a convex combination of the source deformation gradients.

Statistical expression model. The idea of blending can be extended to build a statistical model for expressions. As above we assume the source sequences are in temporal correspondence, *i.e.*, each of the k sequences contains exactly F frames. Blanz and Vetter [19] built a morphable model by running a PCA on the point and color values. In this application we deal with data of a higher dimension namely the deformation gradients over all frames. Like the shape of a face can be represented by concatenating the x, y, z components of all points to a high dimensional vector, we represent one sequence by concatenating all components of the deformation gradients of all triangles at all frames, *i.e.*, for $F-1$ deformations and m triangles the produced vector has dimensionality $\mathbb{R}^{(F-1)\cdot m\cdot 9}$. Running PCA on these vectors leads to average deformations for each frame $\bar{\mathbf{S}}^f \in \mathbb{R}^{3m\times 3}$ and principal components $\mathbf{s}_j^f \in \mathbb{R}^{3m\times 3}$ with $j \in \{1,\dots,k-1\}$. Similar to the example above we calculate a whole sequence from a target mesh $\mathcal{T}^1$ representing a neutral expression by computing new point positions for $\mathcal{T}^f$ as

$$\begin{pmatrix} \mathbf{p}_1^{f\,T} \\ \vdots \\ \mathbf{p}_n^{f\,T} \end{pmatrix} = \mathbf{K}(\mathcal{T}^1)\left(\bar{\mathbf{S}}^f + \sum_{j=1}^{k'} w_j \cdot \mathbf{s}_j^f\right)$$

where we select $k' < k$ in order to omit eigenvectors with small eigenvalues and thereby to drop negligible details. In Figure 11 we demonstrate such a model obtained from five sequences of a smile: it shows the average expression together with the variations produced by adding or subtracting the first most significant eigenvectors.

6. Limitations

The main problem for our system is caused by bad illumination conditions, which will introduce noise in the image and an insufficient depth of field. Then the *surfel fitting* as well as *mesh tracking* is likely to fail. As many 3D reconstruction methods *surfel fitting* will fail in the present of specular highlights caused by a shiny skin, glasses or piercings. Our system is not capable to capture faces with a long and dense beard, although we observed that a very short beard adds some natural texture to the face which makes tracking and 3D reconstruction easier. The same thing holds for tattoos. In our experiments we observed that the natural texture of human skin is sufficient to do *surfel fitting* and *mesh tracking*. But some persons have extremely smooth skin, which causes *surfel fitting* to produce strong outliers, since the gradient of the energy functional is too shallow. The same effect would occur if subjects would use a lot of makeup.

In general we observed that *surfel fitting* sometimes produces strong outliers if the surfel is only visible by two cameras, which usually occurs at the face template's boundary, but this problem could be solved by adding more cameras to the rig. If such an outlier was not discarded, it is possible that it has still a large influence in the modeling step, since we solve the system in the least squares sense. In future work we plan to use RANSAC [39] methods to detect and eliminate outliers more robustly.

The current design of the system cannot deal with occlusion. Occluding the face for some frames of the video would induce a strong drift during the *mesh tracking* and would cause *surfel fitting* to produce surfels which do not lie on the face. Related to this, strong rotation of the head would also cause drifting artifacts during *mesh tracking*. To avoid such drifting artifacts in general, we want to use a global facial feature detector which can fix corresponding vertices of the 2D mesh to facial feature points like corners of the eyes and lips. In addition such a feature detector makes the manual selection of facial features, as needed for the first frame, redundant.

7. Discussion and Conclusion

In this paper we introduced a system for markerless reconstruction of dynamic faces. Our system is able to establish inter-subject correspondence as well as temporal correspondence. In numerous examples we show that the system performs well and that the results can be used in different applications.

Standard 3D reconstruction approaches from sparse feature points are generally quite sensitive to noise. The reason for this is that in the energy function to be minimized only a small local image region is considered. We overcome this problem by using a simple morphable model of neutral faces to estimate the more global appearance of the face seen in the first image. This generates a surface similar to the one being reconstructed, which strongly increases the robustness of the *surfel fitting*.

Instead of using the proposed *2D mesh tracking* one could also employ optical flow methods [36] to establish temporal correspondences. In our setting it is possible to adapt the topology of the 2D mesh such that it contains holes at mouth and eyes. This makes the *2D mesh tracking* superior to optical flow, since it can handle discontinuities at potentially separating regions, like eyes or lips, in a more natural way.

In general spatial-temporal correspondence can be obtained by tracking features between views and frames. We mainly had two problems with feature tracker like the KLT tracker introduced in [32] or deriving correspondences by using SIFT features proposed by Lowe [40]: First, the temporal tracking does not take the neighboring features into account. Because of that, it often produces trajectories which slide past each other. These foldovers induce high distortions in the tracked face template. To correct this, we chose the proposed *2D mesh tracking* since it allows us to control the global smoothness and prevent the features from sliding. Second, tracking features between views produces only a very sparse set of 3D features, which do not provide enough constraints for the modeling step to get reliable results. In our proposed method, we distribute a large set of (redundant) image-samples, so we are able to omit wrongly reconstructed image-samples but still end up with a large set of constraints for the modeling phase. Since each image-sample is considered independently, the 3D reconstruction is simple. By combining this computer vision technique with a simple modeling approach which fulfills each constraint in the least squares sense, a smooth surface can be produced and the robustness of the reconstruction method can be increased (visibility, good initial solutions), while simultaneously maintaining full correspondence between frames and subjects.

We showed that these correspondences allow us to automatically enhance our model by placing eyes, lids and lashes. We presented the deformation transfer in an intuitive way and used it to transfer expressions from one subject to another, to attenuate expressions or to blend expressions of different subjects. Finally we introduced a dynamic face model, built from a principal component analysis of the deformation gradients. Since the target shape can be produced by blending the shapes of different faces or be computed by a morphable model, the user can easily produce a broad variety of shapes and facial expressions from only a few examples. In our experimental validation we only used five subjects to build a model for one particular emotion. In future works we would like to build models for different emotions and incorporate more subjects.

8. Acknowledgements

This work has been supported by DFG (IRTG 1328) and the UMIC Research Centre (DFG EXC 89), RWTH Aachen University.

[1] M. Habbecke, L. Kobbelt, Iterative multi-view plane fitting, in: Vision, Modeling and Visualization, 2006.

[2] M. Botsch, R. Sumner, M. Pauly, M. Gross, Deformation transfer for detail-preserving surface editing, in: Proc. of Vision, Modeling and Visualization, 2006, pp. 357–364.

[3] P. Joshi, M. Meyer, T. DeRose, B. Green, T. Sanocki, Harmonic coordinates for character articulation, ACM Trans. Graph. 26 (3) (2007) 71.

[4] Y. Lipman, D. Levin, D. Cohen-Or, Green coordinates, ACM Trans. Graph. 27 (3) (2008) 1–10.

[5] M. Ben-Chen, O. Weber, C. Gotsman, Variational harmonic maps for space deformation, ACM Trans. Graph.

[6] K. Waters, A muscle model for animation three-dimensional facial expression, in: Computer graphics and interactive techniques, USA, 1987, pp. 17–24.

[7] P. Ekman, W. V. Friesen, Manual for the facial action coding system, Consulting Psychology Press.

[8] Y. Lee, D. Terzopoulos, K. Waters, Constructing physics-based facial models of individuals, in: In Proc. Graphics Interface, 1993, pp. 1–8.

[9] K. Kähler, J. Haber, H. Yamauchi, H.-P. Seidel, Head shop: generating animated head models with anatomical structure, in: Proc. SCA, New York, USA, 2002.

[10] E. Sifakis, I. Neverov, R. Fedkiw, Automatic determination of facial muscle activations from sparse motion capture marker data, ACM Trans. Graph. 24 (3) (2005) 417–425.

[11] C. Curio, M. Breidt, Q. C. V. M. Kleiner, M. A. Giese, H. H. Bülthoff, Semantic 3d motion retargeting for facial animation, in: Applied perception in graphics and visualization, USA, 2006, pp. 77–84.

[12] B. Bickel, M. Botsch, R. Angst, W. Matusik, M. Otaduy, H. Pfister, M. Gross, Multi-scale capture of facial geometry and motion, ACM Trans. Graph. 26 (3) (2007) 33.

[13] Williams, Lance, Performance-driven facial animation, in: Computer graphics and interactive techniques, USA, 1990, pp. 235–242.

[14] L. Zhang, N. Snavely, B. Curless, S. M. Seitz, Spacetime faces: High-resolution capture for modeling and animation, in: ACM Annual Conference on Computer Graphics, 2004, pp. 548–558.

[15] W.-C. Ma, A. Jones, J.-Y. Chiang, T. Hawkins, S. Frederiksen, P. Peers, M. Vukovic, M. Ouhyoung, P. Debevec, Facial performance synthesis using deformation-driven polynomial displacement maps, ACM Trans. Graph. 27 (5) (2008) 1–10.

[16] C. Hernández, G. Vogiatzis, G. J. Brostow, B. Stenger, R. Cipolla, Non-rigid photometric stereo with colored lights, in: Intl. Conf. on Comp. Vision, 2007.

[17] T. Weise, B. Leibe, L. V. Gool, Fast 3d scanning with automatic motion compensation, in: Proc. CVPR, 2007.

[18] T. Weise, H. Li, L. Van Gool, M. Pauly, Face/off: live facial puppetry, in: Symposium on Computer Animation, ACM, 2009, pp. 7–16.

[19] V. Blanz, T. Vetter, A morphable model for the synthesis of 3d faces, in: A. Rockwood (Ed.), Siggraph 1999, Computer Graphics Proceedings, 1999, pp. 187–194.

[20] D. Vlasic, M. Brand, H. Pfister, J. Popović, Face transfer with multilinear models, ACM Trans. Graph. 24 (3) (2005) 426–433.

[21] M. Dellepiane, N. Pietroni, N. Tsingos, M. Asselot, R. Scopigno, Reconstructing head models from photographs for individualized 3d-audio processing, Proc. Pacific Graphics 27 (7) (2008) 1719–1727.

[22] M. Odisio, M. Odisio, G. Bailly, Shape and appearance models of talking faces for model-based tracking, in: Proc. AVSP, 2003, pp. 105–110.

[23] S. C. Koterba, S. Baker, I. Matthews, C. Hu, J. Xiao, J. Cohn, T. Kanade, Multi-view aam fitting and camera calibration, in: Proc. ICCV, Vol. 1, 2005, pp. 511 – 518.

[24] S. Vedula, S. Baker, P. Rander, R. Collins, T. Kanade, Three-dimensional scene flow, in: Proc. ICCV, Vol. 2, 1999, pp. 722 – 729.

[25] R. Li, S. Sclaroff, Multi-scale 3d scene flow from binocular stereo sequences, Comput. Vis. Image Underst. 110 (1) (2008) 75–90.

[26] Y. Furukawa, J. Ponce, Dense 3d motion capture for human faces, in: Proc. CVPR, 2009.

[27] G. Borshukov, D. Piponi, O. Larsen, J. P. Lewis, C. Tempelaar-Lietz, Universal capture - image-based facial animation for ”the matrix reloaded”, in: ACM SIGGRAPH 2005 Courses, ACM, 2005, p. 16.

[28] D. Bradley, W. Heidrich, T. Popa, A. Sheffer, High resolution passive facial performance capture, ACM Trans. Graph. 29 (4) (2010) 1–10.

[29] J.-y. Noh, U. Neumann, Expression cloning, in: ACM Siggraph, ACM, 2001, pp. 277–288.

[30] K. Na, M. Jung, Hierarchical retargetting of fine facial motions, Comput. Graph. Forum 23 (3) (2004) 687–695.

[31] R. W. Sumner, J. Popović, Deformation transfer for triangle meshes, in: ACM Trans. Graph., ACM, 2004, pp. 399–405.

[32] J. Shi, C. Tomasi, Good features to track, in: Computer Vision and Pattern Recognition, 1994, pp. 593–600.

[33] Z. Zhang, A flexible new technique for camera calibration, IEEE Trans. Pattern Anal. Mach. Intell. 22 (11) (2000) 1330–1334.

[34] J. Nocedal, S. Wright, Numerical Optimization, 2nd Edition, Springer.

[35] B. K. P. Horn, Closed-form solution of absolute orientation using unit quaternions, Journal of the Optical Society of America. A 4 (1987) 629–642.

[36] B. K. Horn, B. G. Schunck, Determining optical flow, Tech. rep., Cambridge, MA, USA (1980).

[37] M. Botsch, L. Kobbelt, A remeshing approach to multiresolution modeling, in: Proc. SGP, 2004, pp. 185–192.

[38] L. Katz, Magill's Medical Guide Revised Edition, Salem Press, 1998.

[39] M. A. Fischler, R. C. Bolles, Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography (1987) 726–740.

[40] D. G. Lowe, Distinctive image features from scale-invariant keypoints, Int. Journal of Computer Vision 60.