Eurographics/ ACM SIGGRAPH Symposium on Computer Animation (2008)
M. Gross and D. James (Editors)

Pose-Space Animation and Transfer of Facial Details

Bernd Bickel[1], Manuel Lang[1], Mario Botsch[1,2], Miguel A. Otaduy[1,3], Markus Gross[1]

[1]Computer Graphics Laboratory, ETH Zurich
[2]Computer Graphics Group, Bielefeld University
[3]Department of Computer Science, URJC Madrid

Abstract

This paper presents a novel method for real-time animation of highly-detailed facial expressions based on a multi-scale decomposition of facial geometry into large-scale motion *and* fine-scale details*, such as expression wrinkles. Our* hybrid animation *is tailored to the specific characteristics of large- and fine-scale facial deformations: Large-scale deformations are computed with a fast linear shell model, which is intuitively and accurately controlled through a sparse set of motion-capture markers or user-defined handle points. Fine-scale facial details are incorporated using a novel pose-space deformation technique, which learns the correspondence of sparse measurements of skin strain to wrinkle formation from a small set of example poses. Our hybrid method features real-time animation of highly-detailed faces with realistic wrinkle formation, and allows both large-scale deformations and fine-scale wrinkles to be edited intuitively. Furthermore, our pose-space representation enables the transfer of facial details to novel expressions or other facial models.*

1. Introduction

The face gathers the foremost relevant visual characteristics of human identity and expression, and the quest for realistic computer-generated characters has acknowledged it over the years [NN99, PL06]. Sophisticated methods allow today the acquisition of detailed expressive facial deformations [ZSCS04, MHP*07, BBA*07], and high-quality real-time rendering techniques [dLE07] lay some of the components for highly-detailed facial animation for video games [BMW*06] or interaction with virtual avatars.

The complexity of facial tissue induces highly nonlinear skin bulging and wrinkling effects under expressive facial motion, which are extremely complex to model and compute. Animators and artists, however, need high fidelity face models that are easy and intuitive to control, a task for which interactive performance is crucial.

Data-driven approaches therefore avoid complex computations by blending example poses from a database of facial expressions [PSS99, BV99, VBPP05]. However, spanning the space of expressive motions requires a large and complex-to-acquire database [For03], and deriving the blending weights becomes difficult and/or expensive [JTDP03]. Facilitated by the steadily increasing CPU performance, *computational approaches* are nowadays able to realistically model facial details using complex nonlinear deformation models [SNF05, BBA*07], but are still tied to high computational cost. Bickel et al. [BBA*07] decomposed facial motion into large-scale global motion and fine-scale details, such as expression wrinkles. While the first can efficiently be computed using a linear shell deformation model, their fine-scale wrinkle synthesis still requires a computationally expensive nonlinear thin-shell model.

We propose a *hybrid animation* technique that combines computational and data-driven approaches, and thereby achieves detailed facial expressions that are intuitive to control as well as efficient to compute. Building on the two-scale decomposition of [BBA*07], we compute the large-scale motion using the same linear shell deformation, but in contrast employ a data-driven approach for learning fine-scale facial details from a small set of example poses.

While skin wrinkling is a highly nonlinear effect, there is a clear relationship to stretching and compression of the underlying skin. We therefore propose a novel pose-space deformation (PSD) technique, which in a preprocess learns the correlation of wrinkle formation to sparsely measured skin strain. At run-time, given an arbitrary facial expression, it computes the skin strain and derives from it the necessary fine-scale corrections for the large-scale deformation.

This hybrid animation technique has several advantages over purely computational or purely data-driven approaches. First, our technique is very accurate: The large-scale deformation is driven by a sparse set of positional constraints,

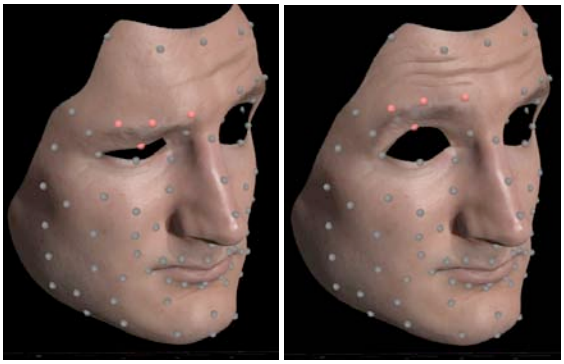


Figure 1: ***Interactive Editing of Highly-Detailed Faces.*** *The user intuitively edits a 1M-triangles face model by dragging the handle points near the eyebrow. The face deforms in real-time and produces realistic fine-scale wrinkles.*

such as motion-capture (*mocap*) markers or user-defined handle points, which are interpolated exactly and therefore enable precise control. The relation of skin strain to fine-scale wrinkles can be learned more efficiently than the relation of marker positions to a nonlinearly deformed face. As a consequence, our pose-space deformation works very well even with just a few example poses, although arbitrary accuracy can be obtained by adding more poses.

Our hybrid approach allows large-scale and fine-scale deformations to be solved as superposition of precomputed response functions. Exploiting this, we derive a highly parallel GPU implementation that achieves real-time animations of detailed facial expressions. Those animations are controlled through a set of marker positions, which in combination with real-time performance yields an intuitive editing system for realistic face animations (cf. Fig. 1).

Face models in games, virtual reality, or computer animation often appear unrealistic due to the lack of facial details. Once learned from a set of example poses, our explicit pose-space representation of fine-scale details allows us to easily transfer them onto other animated faces, thereby enhancing their visual plausibility without changing their overall motion. As such, our method provides a simple way to re-use captured or manually modeled high-res face data.

2. Related Work

Modeling, acquisition, and animation of human faces are topics largely explored in computer vision and computer graphics [NN99, PL06]. Our target problem falls into the category of facial animation, and our proposed method is kindred to techniques in character skinning, hence we focus our discussion of related work on those research areas.

One large family of methods in face animation employs model blending. Blend shapes, dating back to the early work of Parke [Par74] and commonly used in the industry today, linearly combine several poses of the same face model, either designed by an artist or acquired from a person. This method suffers from two main drawbacks: It requires a large number of poses for spanning the range of possible expressions [For03], and blending controls are hard to tune. Blend shapes have therefore been extended to automatic blending control [PSS99] and automatic segmentation [JTDP03], but those methods require computationally expensive optimizations. The FaceIK technique of [ZSCS04] allows for interactive and local blending control when the number of matched points is smaller than the number of blend shapes. Morphable models [BV99] and their extension to multilinear models [VBPP05] not only blend different expressions from the same object, but also from a large database of different subjects. The common drawback of blend shapes, however, is that they typically do not allow for intuitive, precise, and interactive control at the same time.

Another family of methods uses anatomical models activated by controls with biomechanical meaning [KGC*96, EBDP96, SNF05]. They allow extending the range of motions beyond muscle-driven expressions by incorporating external forces. Essa et al. [EBDP96] learn muscle activations for matching video-captured poses, and then interpolate the learned poses. Sifakis et al. [SNF05] learn functions for muscle activation in a more sophisticated tissue model. Furthermore, several approaches focus on specific anatomical models for wrinkle simulation [MTKL*02, ZS05, VLR05]. Among those, Wu et al. [WKMT96] designed procedural methods for wrinkle synthesis as a function of the strain measured in an underlying tissue model. Those approaches allow for realistic model behavior, but in turn require complex parameter tuning and expensive computations.

The parameter-setting complexity of anatomical models can be largely relaxed by adding user- or data-driven constraints to the deformation. Pighin et al. [PHL*98] matched corresponding vertices in a generic model to mocap markers, and then smoothly interpolated the deformation on the rest of the face. Bickel et al. [BBA*07] successfully decomposed face animations into two scales, computing the large-scale motions by a fast linear shell model driven by mocap markers, thus requiring an expensive nonlinear model only for synthesizing fine-scale wrinkles. Furthermore, they require spatial and temporal dense capturing of geometric wrinkle constraints. In contrast, our hybrid framework exploits the quasi-static nature of fine-scale wrinkles by employing an example-based model for fine-scale skin corrections, and thereby avoids spatio-temporal dense acquisition and any expensive nonlinear optimizations at run-time.

Example-based skin correction models have been successfully used for modeling arms [LCF00] or hands [KJP02, KM04]. They typically combine a fast, linear skeletal subspace deformation (SSD) [MTLT88] with a nonlinear pose-space deformation (PSD) [LCF00] that interpolates correction vectors among example poses. PSD was extended to

support weighted (i.e., per-vertex) pose space deformation (WPSD) [KM04, RLN06], which largely reduces the number of required example poses. Similar in spirit to PSD, the EigenSkin method [KJP02] performs corrections to SSD, but derives a reduced basis from a PCA of the input examples. The recently presented method of [MA07] is related to our work in the sense that it computes high-resolution deformations from a few handle vertices, but it focuses on the selection of handles given a large training dataset. Other recent methods [WPP07, WSLG07] learn example-based corrections on sparse points and assume that these corrections can be smoothly interpolated. In general, any pose-space method requires the definition of a suitable pose space, which for SSD can easily be defined using bone transformations. In the context of face models, we define a novel pose space based on local skin strain and derive corresponding PSD and WPSD techniques for fine-scale skin correction.

Several previous approaches address the transfer of whole face animations from one subject onto other faces [DM00, NN01, BBPV03, CXH03, SP04]. In contrast, our pose-space representation allows to transfer only the fine-scale details from one character onto another, which can be used to add realistic wrinkles to existing models and face animations.

3. Overview

We start with an overview of the input data and the workflow of our hybrid face animation pipeline, depicted in Fig. 2, before describing the large-scale and fine-scale deformations in more detail in the following sections.

In contrast to blend shapes, which linearly combine several facial poses, we use a single high-resolution **face mesh** $\mathcal{F}$ of the rest pose ($\approx$ 1M triangles), and generate all frames of the animation by deforming $\mathcal{F}$ from its rest pose. Following [BBA*07], we compute the large-scale facial motion using a linear thin shell model (Section 4). Its deformation is controlled through a set of $H \approx 100$ **handle vertices**, which may correspond to a set of face points tracked during an actor's performance, or may be interactively controlled by an animator. The resulting large-scale deformation successfully captures the overall facial expression, but lacks fine-scale facial details such as expression wrinkles (Fig. 2, left).

The nonlinear behavior that goes beyond the linear large-scale motion is learned in a preprocess from a set of P **example poses**. These examples are represented by high-resolution triangle meshes in full vertex correspondence with the rest pose $\mathcal{F}$. In practice, example poses can for instance be created manually by an artist, by capturing an actor's face with a high-resolution scanner [ZSCS04, MHP*07], or by reconstructing a detailed face mesh with expression wrinkles [BBA*07]. Given the example poses, the corresponding fine-scale details are extracted as the difference between the examples and the results of the large-scale deformation for the same poses, and are stored per-vertex in local tangent frames.

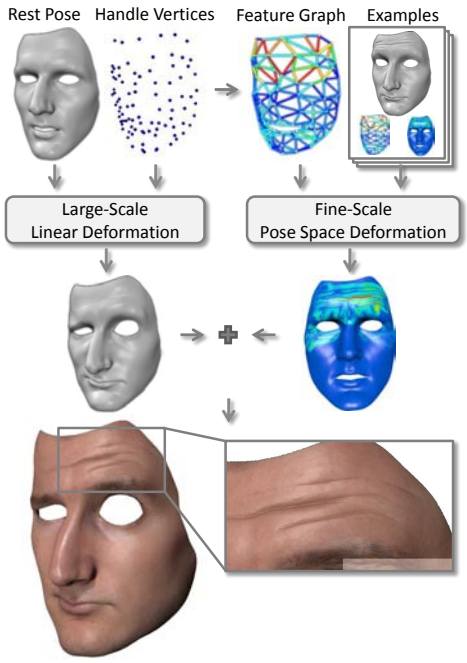


Figure 2: *Our hybrid face animation pipeline computes the large-scale facial motion from a linear deformation model, and adds fine-scale details using a pose-space deformation that learns skin corrections from a set of example poses.*

In order to interpolate the fine-scale corrections of the examples onto other poses, we define a novel pose space for facial expressions based on skin strain, and learn the formation of skin bulges and wrinkles in this space. To this end, we construct a **feature graph** from the H handle vertices, following approximately the contour of the face, but without the need to produce a consistent mesh (e.g., edges may cross each other). The feature graph's edges constitute the locations where skin strain is measured (Section 5.1). For each new pose in an animation, we compute the skin strain according to the deformed feature graph, and use this information for a pose-space correction of fine-scale facial details (Fig. 2, right, Sections 5.2 and 5.3).

4. Large-Scale Deformation

For the large-scale face deformations we employ the method of [BBA*07], which demonstrates that the displacements of a set of sparse handle vertices provide sufficient geometric constraints for capturing the large-scale facial motion in a plausible way. Given as input constraints the 3D displacements $\mathbf{u}_H \in \mathbb{R}^{H\times 3}$ of the H handle vertices, the large-scale deformation $\mathbf{u}$ is computed by minimizing a simplified, quadratic thin shell energy. This amounts to solving the corresponding Euler-Lagrange equations

$$-k_s \Delta \mathbf{u} + k_b \Delta^2 \mathbf{u} = 0, \qquad (1)$$

under the constraints $\mathbf{u}_H$ imposed by handle vertices. In this equation, k_s and k_b denote the stiffness for surface stretching and bending, respectively, and Δ is the Laplace-Beltrami operator. Discretizing the latter using the cotangent weights [MDSB03] yields the linear system

$$\begin{pmatrix} \mathbf{A} & \mathbf{A}_H \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \mathbf{u}_H \end{pmatrix} = \mathbf{0},$$

to be solved for the unknown displacements $\mathbf{u} \in \mathbb{R}^{(V-H)\times 3}$ of the $(V-H)$ free vertices.

In contrast to [BBA*07], who solved this system for each frame using a sparse Cholesky factorization [TCR], we precompute the "basis function" matrix $\mathbf{B} = -\mathbf{A}^{-1}\mathbf{A}_H$, which depends only on the rest-pose mesh $\mathcal{F}$ and the selection of handle vertices. After factorizing $\mathbf{A}$ once, each column of $\mathbf{B}$ is computed by solving a sparse linear system involving $\mathbf{A}$ and the corresponding column of $\mathbf{A}_H$. At run-time, the large-scale deformation $\mathbf{u}$ is obtained from the handle displacement $\mathbf{u}_H$ by the matrix product $\mathbf{u} = \mathbf{B} \cdot \mathbf{u}_H$, which can efficiently be computed in a parallel GPU implementation.

5. Fine-Scale Deformation in Pose Space

The linear deformation model described in the last section approximates the facial motion well at a large-scale. However, nonlinear fine-scale effects, such as the bulging produced by expression wrinkles, cannot be reproduced. These fine-scale deformations are highly nonlinear with respect to the positions of the handle vertices. However, they vary smoothly as a function of facial pose, hence we have opted for learning, as a preprocess, the fine-scale displacement $\mathbf{d}$ from a set of example poses, and then at run-time compute it by interpolation in a suitable facial pose space.

We first define a facial pose space based on a rotation-invariant feature vector of skin strain (Section 5.1), and formulate the learning problem as scattered-data interpolation in this pose-space (Section 5.2). We extend the basic method to *weighted* pose-space deformation (Section 5.3), thereby allowing for a more compact basis. Based on our explicit pose-space representation we describe the transfer of fine-scale details onto novel faces and animations (Section 5.4).

5.1. Definition of the Feature Vector

At each animation frame, the raw input data describing a pose consists of the positions of the handle vertices. This data is not invariant under rigid transformations, hence it does not constitute an effective pose descriptor, which, however, is required for the pose-space deformation method described in the next section. Bulging effects of wrinkles appear due to lateral compression of skin patches [WKMT96]. Exploiting this correlation, we suggest a feature vector that measures skin strain at various points across the face.

Based on the feature graph connecting the handle points using F edges (Fig. 2), we define the F-dimensional feature vector $\mathbf{f} = [f_1, \ldots, f_F]^T$ of a pose containing in f_i the relative stretch of the i'th feature edge, which can be regarded as a measure of strain. Specifically, given the positions of the endpoints $\mathbf{p}_{i,1}$ and $\mathbf{p}_{i,2}$, and the rest length l_i, we define

$$f_i = \left(\left\|\mathbf{p}_{i,1} - \mathbf{p}_{i,2}\right\| - l_i\right)/l_i\,.$$

This feature vector is invariant under rigid body transformations, and the feature distance $\left\|\mathbf{f} - \mathbf{f}_j\right\|$ corresponds to the Euclidean norm of relative edge stretch from the j'th example pose to the input pose. Fig. 3 shows the correlation between the feature vector and wrinkle formation. Alternatively, a per-vertex strain tensor could be used, but in practice our discrete strain approximation turned out to be sufficient.

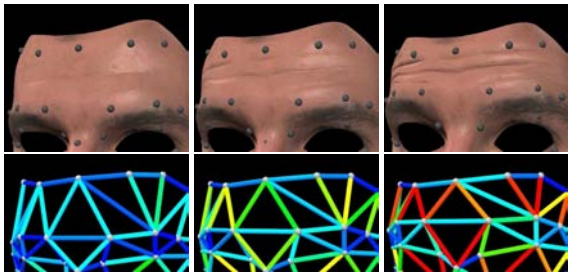


Figure 3: ***Wrinkle Formation and Skin Strain.*** *Close-up of the forehead for three input examples (top) and the corresponding strain on the feature edges (bottom), showing the correlation of lateral skin strain and wrinkle formation.*

5.2. Pose-Space Deformation

In order to exploit the connection of skin strain to wrinkle formation we want to learn their functional relation in a pose-space deformation (PSD) framework. To this end, we represent each facial expression by its rotation-invariant feature vector $\mathbf{f}$ as described in the last section. Hence, each facial expression corresponds to a point in an F-dimensional pose space, which constitutes the domain of the function we want to learn. Its range is the fine-scale detail correction $\mathbf{d}$, represented as well in a rotation-invariant manner by storing them in per-vertex local frames.

Each of the P example poses corresponds to a feature vector $\mathbf{f}_i$ with its associated fine-scale displacement $\mathbf{d}_i$. Then, PSD corresponds to a scattered data interpolation problem, for which we employ radial basis functions (RBFs). Hence, the function $\mathbf{d} : \mathbb{R}^F \to \mathbb{R}^{3V}$, mapping a facial pose to 3D fine-scale displacement, has the form

$$\mathbf{d}(\mathbf{f}) = \sum_{j=1}^{P} \mathbf{w}_j \cdot \varphi\left(\left\|\mathbf{f} - \mathbf{f}_j\right\|\right), \qquad (2)$$

where φ is a scalar basis function, and $\mathbf{w}_j \in \mathbb{R}^{3V}$ and $\mathbf{f}_j$ are the weight and feature vector for the j'th example pose. We employ the biharmonic RBF kernel $\varphi(r) = r$, since it allows for smoother interpolation of sparsely scattered example poses than locally supported kernels [CBC*01].

As a preprocess we compute the RBF weights $\mathbf{w}_j$ in one of two possible ways. If the training dataset of examples consists of only a small number P of extreme poses (e.g., modeled by an artist), we form the basis with all P poses, and compute the weights $\mathbf{w}_j$ such that the displacements of the example poses are interpolated exactly, i.e., $\mathbf{d}(\mathbf{f}_i) = \mathbf{d}_i$. This reduces to solving $3V$ linear $P \times P$ systems, which differ in their right-hand side only.

If the training dataset consists of a large number of example poses with redundant information (e.g., captured from an actor's performance), we select a compact basis of P poses, and compute the weights $\mathbf{w}_j$ that fit all T examples in a least-squares manner. This again amounts to solving $3V$ linear $P \times P$ systems. We select the poses for the basis in a greedy manner [CBC*01]: Starting with the rest-pose, we incrementally add the pose with largest error and recompute the weights. This pose selection is fully automatic, can achieve arbitrary accuracy, and allows to intuitively specify an approximation threshold.

5.3. Weighted Pose-Space Deformation

In our basic definition of PSD in Section 5.2, every input example influences all vertices of the face mesh in the same manner, since we compute a single feature distance $\|\mathbf{f}-\mathbf{f}_j\|$ per example pose j. As a consequence, the basis of example poses is required to grow exponentially to sufficiently sample the combinations of independent deformations (e.g., raising both eyebrows versus raising just one eyebrow). As an answer to this problem, we adopt a weighted pose-space deformation (WPSD) scheme [KM04].

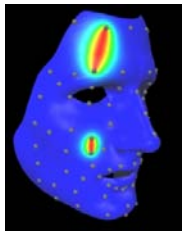
This requires to redefine (2) to compute feature distances in a per-vertex manner, replacing the Euclidean metric $\|\mathbf{f}-\mathbf{f}_j\|$ with a weighted distance metric per vertex v:

$$\|\mathbf{f}-\mathbf{f}_j\|_v := \left(\sum_{i=1}^{F} \alpha_{v,i}\,(f_i - f_{j,i})^2\right)^{1/2}, \qquad (3)$$

where $f_{j,i}$ is the strain of the i'th feature edge in the j'th pose. We exploit the fact that the components of our feature vector (i.e., relative stretch of feature edges) measure local properties for assigning weights $\alpha_{v,i}$ based on proximity to the feature edges. Specifically, for a vertex v we define the weight of the i'th feature edge as

$$\alpha_{v,i} = \frac{\bar{\alpha}_{v,i}}{\sum_i \bar{\alpha}_{v,i}}, \quad \text{with} \quad \bar{\alpha}_{v,i} = e^{-\beta(L_{v,i}-l_i)}, \qquad (4)$$

where l_i is the rest length of the feature edge, and $L_{v,i}$ is the sum of rest-pose distances from vertex v to the edge endpoints. This weight kernel is 1 on the edge, and decays smoothly everywhere else, as shown in the adjacent figure. The parameter β can be used to control the degree of decay, based on the local density of handle vertices. In our experiments we discard weights $\bar{\alpha} < 0.025$, and set β such that a vertex is influenced by at most 16 edges.



For WPSD, the computation of RBF weights slightly differs from PSD as described in Section 5.2. With function

$$\mathbf{d}_v(\mathbf{f}_i) := \sum_{j=1}^{P} \mathbf{w}_{v,j} \cdot \varphi\left(\|\mathbf{f}-\mathbf{f}_j\|_v\right), \qquad (5)$$

the interpolation problem is $\mathbf{d}_v(\mathbf{f}_i) = \mathbf{d}_{v,i}$, $\forall v \forall i$, i.e., for each vertex v the function $\mathbf{d}_v$ should interpolate v's displacements in the given example poses. Nevertheless, the weight computation still involves solving $3V$ linear $P \times P$ systems, but the matrix describing the linear system is different for each vertex v. Equivalently to PSD, the WPSD method also yields $3P$ weights to be stored per vertex.

At run-time, given the feature vector $\mathbf{f}$ of the current pose, we evaluate for every vertex the weighted feature distance to each basis pose according to (3) and compute the fine-scale displacement vector according to (5), accumulating the contributions of the precomputed RBF weights. The computations can be easily parallelized over all vertices, allowing for a highly efficient GPU implementation.

5.4. Transfer of Fine-Scale Details

Our representation of fine-scale details furthermore allows to transfer facial details onto new faces and animations. As acquiring or manually modeling high-resolution face geometry is often tedious and expensive, this provides a simple alternative for enhancing otherwise unnaturally smooth looking face models by reusing available fine-scale data.

Due to the fact that the fine-scale deformation is already defined in pose-space, controlled by local skin strain and measured in per-vertex local coordinate frames as described in Section 5.2, its transfer reduces to establishing dense correspondence with the target face model.

We establish correspondence by manually marking approx. 40 feature points on the rest-pose of the source face and the target face. After a rigid pre-alignment, the source face is deformed using the large-scale deformation method described in Section 4 to approximately fit the target face, and is then projected onto the target face, similar to [VBPP05]. Note that this is only necessary for a single frame and any other algorithm yielding dense correspondence could be used as well. As source and target model might not have the same mesh topology, we map each vertex of the target mesh to its enclosing triangle on the deformed source mesh using barycentric vertex weights.

Based on the dense correspondence of the source and target rest-pose, the feature graph with its F edges can be mapped in a straightforward manner to the target model. The nodes of the feature graph and the handle vertices on the target model do not have to match. During the animation we

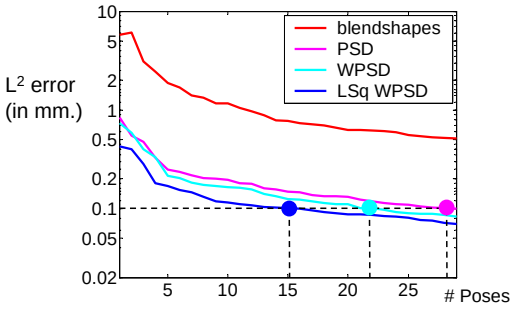


Figure 4: ***L^2 Error Vs. Size of the Basis.*** *Comparison of our implementation of local blend shapes, PSD from Section 5.2, and our novel WPSD from Section 5.3, both with interpolation of poses and with least-squares fit (LSq), measuring error over a training sequence of* 100 *frames.*

first compute the large-scale deformation and hence obtain the necessary feature graph nodes. In case of an existing animation, we do not require the large-scale deformation and instead directly map the feature graph onto the meshes.

The per-vertex RBF weights $\mathbf{w}_j$ of the source face model are transferred to the target model by linear interpolation using the barycentric coordinates of the corresponding triangle. The resulting displacements in their local coordinate frame on the target animation are obtained by evaluating (5).

6. Results and Implementation

This section presents an evaluation and performance analysis of our hybrid face animation method, as well as application experiments, such as animation from mocap data, interactive editing, and transfer of fine-scale details. Please see the accompanying video for the full model performance.

6.1. Evaluation and Comparisons

For comparisons, we have used as input 200 frames of an actor's performance, with high-resolution surface details captured with the method of [BBA*07] (See the top row of Fig. 6 for some input poses). We use as training dataset the first 100 frames, and the other 100 as ground-truth comparison data. In the training sequence the actor performs roughly 5 expressions. For the comparisons, we implemented two versions of blend shape animation. These methods use a set of blend shapes (key facial expressions) to define a linear space of facial expressions [Par74].

Global blend shapes use a single weight for each blend shape. We use our feature distance metric explained in Section 5.1 to find a convex combination of blend shapes that matches the motion capture marker positions of the actor's performance. This naive weighting scheme cannot produce expressions outside the convex combination and requires an exhaustive set of blend shapes.

Locally controlled blend shapes typically are based on a segmentation of the face into individual regions, which allows to blend regions separately and thereby alleviates the shortcomings of global blend shapes. We achieve an equivalent behavior using the locally supported, smoothly varying feature distance metric (3). This metric allows to locally interpolate blend shapes and can be evaluated in real-time. Other more sophisticated non-real time weight controls are possible [PSS99, JTDP03]. However, blend shapes in general suffer from inaccuracy problems as soon as the desired pose is not in the convex space of the example poses.

We compare the blend shape implementation with our **PSD** approach introduced in Section 5.2, and our **WPSD** method described in Section 5.3, based on both exact interpolation of P poses and on least-squares fitting to T poses.

Fig. 4 shows the decay of L^2 error for the training sequence T as more examples are added to P. PSD and WPSD achieve a significantly lower error because, in contrast to blend shapes, our face animation method splits facial geometry into large and fine-scale components, guaranteeing that marker positions of the large-scale component are always exactly interpolated.

Fig. 5 illustrates this for an input pose not present in the training dataset ($P = 6$ poses). Global and local blend shapes suffer from regions of large interpolation error (red arrows). On the other hand, global blend shapes and PSD are affected by incorrect fine-scale details due to lack of local support (blue arrows). Our WPSD model reproduces the input pose best, due to the accuracy of the linear deformation model and local support for synthesizing fine-scale details.

6.2. Performance

Compared to the nonlinear method of [BBA*07], which required approximately 20min per frame for synthesizing facial details, we obtain a performance of 4sec/frame using a pure CPU implementation on a similar machine and for the same mesh complexity, which corresponds to a speed-up of a factor of about 300.

Moreover, our method allows for efficient parallel GPU-implementation with CUDA [NVI]. The basis matrix $\mathbf{B}$, RBF weights $\mathbf{w}_{v,j}$, and feature edge weights $\alpha_{v,i}$ are stored in device memory on the GPU. Hence, we need to transfer only the feature vector $\mathbf{f}$ and the displacements of the handle vertices $\mathbf{u}_H$ at run-time. To reduce memory space and access, $\alpha_{v,i}$ is represented as a sparse matrix.

We tested the performance for a mesh of $V = 530$k vertices and more than 1M triangles, with $P = 6$ example poses, $H = 89$ handle vertices, and $F = 243$ feature edges. These settings require storing 137 floats and 24 integers per vertex. We obtain an overall performance of about 30fps on a Nvidia 8800 GTX graphics card, including large-scale deformation (4ms/frame), fine-scale deformation (13ms/frame),

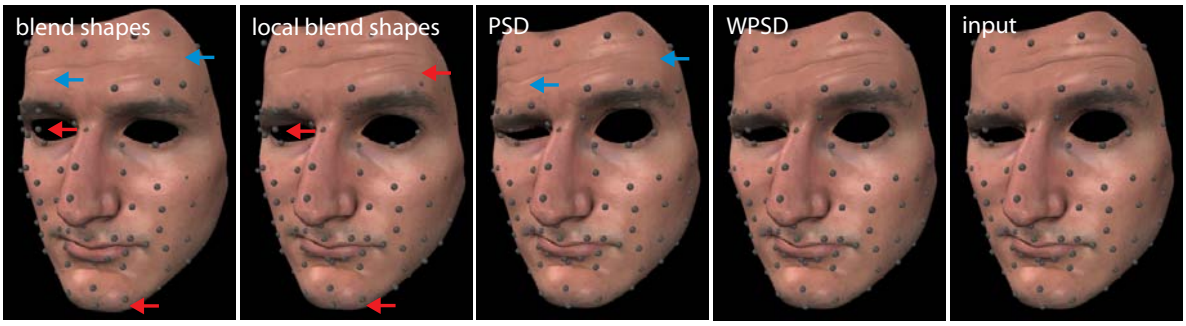


Figure 5: ***Comparison of Methods.*** *Input pose (right) not present in the training data, approximated with various methods from* 6 *examples (See top row of Fig. 6). Arrows highlight large-scale (red) and fine-scale (blue) errors.*

two normal vector updates (2×5ms/frame), and rendering (4ms/frame). With additional real-time subsurface scattering [dLE07] (36ms/frame) we obtain about 15fps. Compared to [BBA*07] this provides a very significant speed-up of 4 orders of magnitude.

6.3. Application Experiments

Animation from Mocap. Given a few poses P, our face animation method allows for high-resolution real-time performance replay from mocap markers, as shown in Fig. 6. We defined the basis for our example-based fine-scale deformation method using 6 poses from the training dataset as described in the previous section (See top row of Fig. 6). Then, the same actor performed a different sequence of expressions, captured using only mocap markers, and our method produced a high-quality performance replay in real-time (See side-by-side comparison in the video). The large-scale animation by constrained deformation provides good approximation quality of the overall face, while the fine-scale example-based correction adds high-resolution details.

Interactive Editing. The computational performance and intuitive control by handle vertices allow for interactive editing of face deformations, as shown in Fig. 1. This might especially be helpful for an artist to fine-tune a mocap sequence. With our pose-space representation of fine-scale details, nonlinear wrinkling effects are produced interactively on complex faces simply by dragging the handle points.

Wrinkle Editing. Our face animation method also allows for very simple and intuitive editing of fine-scale details, thanks to the compact pose-space representation of details using very few example poses. Fig. 7 shows wrinkle editing of an actress' performance. We used a basis with $P = 4$ poses, and modeled artificial wrinkles on 2 of them. With our hybrid face animation method, these added wrinkles are interpolated in pose-space, and seamlessly blend during the complete performance, in real-time.

Fine-Scale Transfer. Face models in games, virtual reality, or computer vision often look unrealistically smooth because high-resolution acquisition or manual modeling can be tedious and expensive [GMP*06]. Our method allows simple transfer of facial details on novel, animated faces, enhancing their visual plausibility. In Fig. 8 and the accompanying video, we show that it can be applied to a wide range of input data by transferring the acquired wrinkles of the subject shown in Fig. 6 to a female face performance captured using only sparse motion capture data, a manually designed human face rigged in Maya, and a publicly available cartoonish character rigged in Blender.

7. Discussion

We have presented a face animation method able to produce complex wrinkling effects on high-resolution meshes in real-time. Its power stems from a two-scale representation of facial deformations, and tailored approaches for computing the deformation at each scale. A constrained deformation approach allows for overall good approximation of facial expressions by smoothly interpolating a sparse set of handles. On the other hand, an example-based deformation approach allows for real-time correction of expressive details.

Along with its hybrid nature, another strength of our solution to face animation lies on the strain-based feature vector for pose representation. Wu et al. [WKMT96] already related wrinkle formation to underlying lateral skin strain, and we learn the relationship in our PSD model, but it would be interesting to further study its nonlinearity and the range of support. Our definition of feature vector has even proved competitive for very fast blend shape control.

Another advantage of our model is the simple fine-scale transfer to novel face models and animations that lack facial details. As future work, it would be interesting to build up a database of skin details across different gender, age, and skin types, that allows fast and intuitive visual enhancements of face animations. Furthermore, our method should theoretically be directly applicable to even higher-resolution data that contains very small wrinkles and pores.

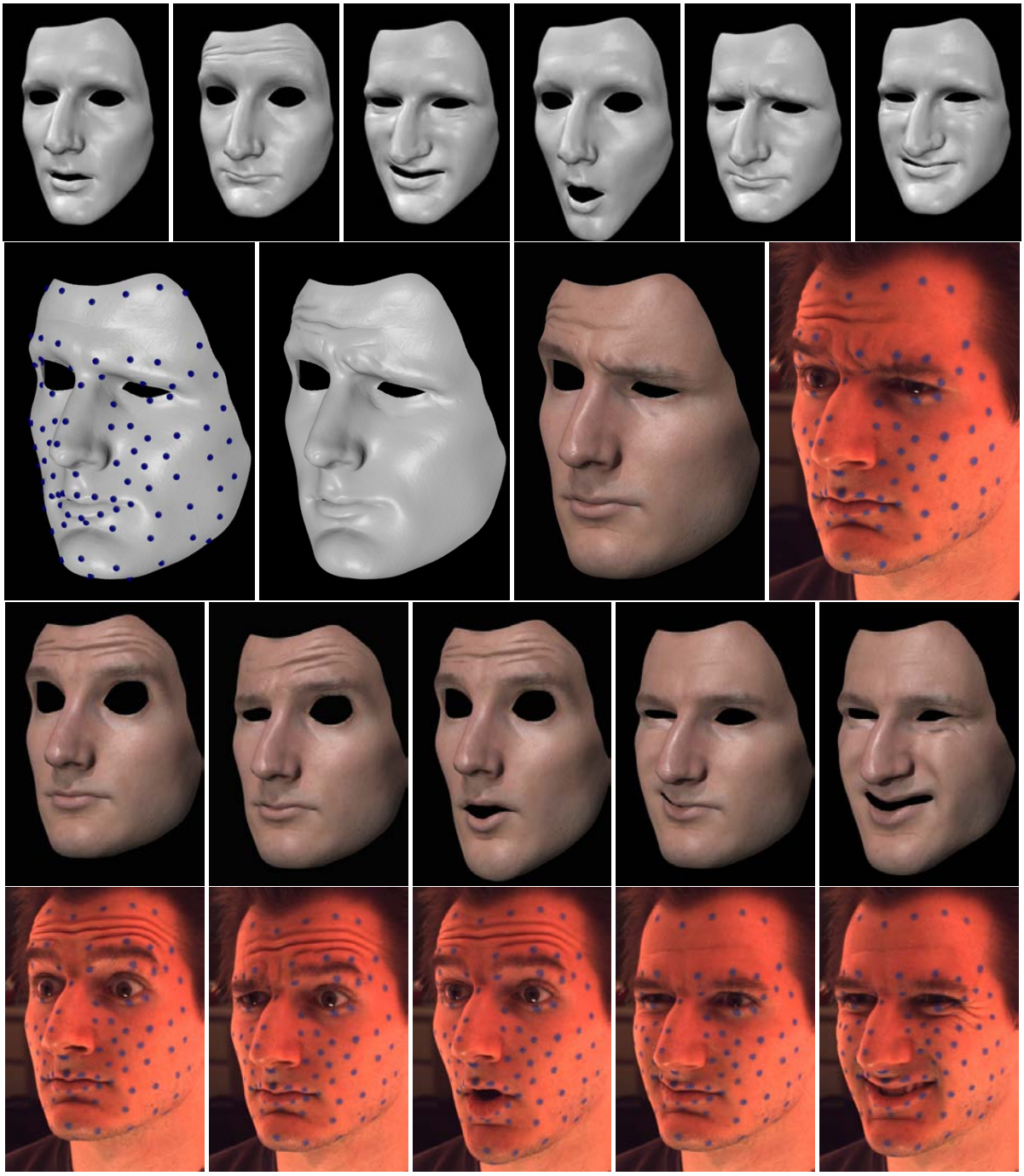


Figure 6: ***Real-Time Face Animation from Mocap.*** *First row: example poses taken from a different sequence; Second row: large-scale deformation interpolating the mocap markers, and full result after example-based fine-scale correction. Third and forth rows: more comparison results, with asymmetric deformations not present in the input examples.*

However, the method also presents some limitations. The most important one, common to all face animation methods driven by handles (e.g., mocap markers) is that the face model cannot react to forces, only to position constraints. Anatomical models can be a solution to this problem, as they react to external forces and can produce realistic deformations beyond facial expressions; however, to date they do not provide detailed deformations in real-time. A related benefit

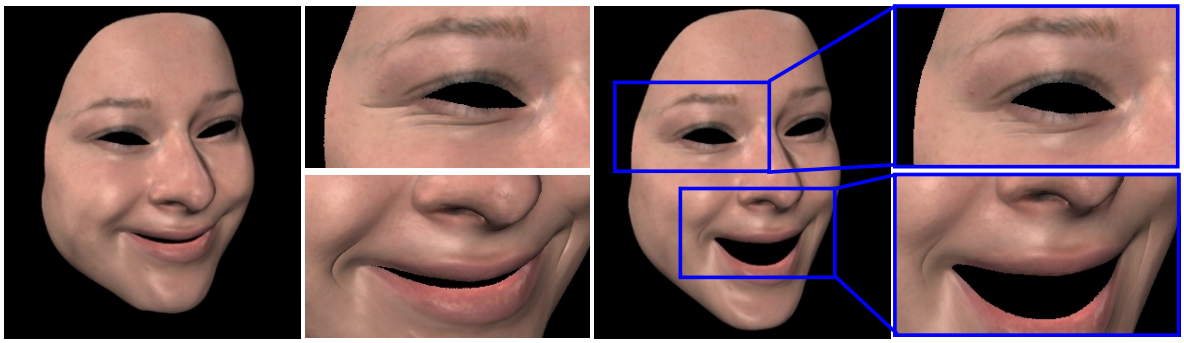


Figure 7: ***Wrinkle Editing on an Actress' Performance.*** *Given an example pose without wrinkles (left), we created wrinkles under the eyes and on the cheeks (center-left). The added wrinkles blend seamlessly during the rest of the performance, as shown in the two rightmost images and the accompanying video.*

Figure 8: ***Fine-Scale Transfer.*** *Given the example poses shown in the first row of Fig. 6, its details are transferred to a manually modeled human character (left), and a publicly available cartoonish character rigged in Blender (right).*

of anatomical models is the possibility to handle collisions naturally. Our model could suffer from self-intersections, especially if the user edits the handles to arbitrary locations, but we did not encounter such problems when the motions are restricted to facial expressions.

Anatomical accuracy would also require the addition of other important facial features like hair, teeth, or the eyes, which are outside the scope of skin deformation, but play a key role in the realism of the full face animation.

Our face animation method trades computational complexity by memory requirements, as each face vertex needs to store handle responses and learned pose displacements. Similar to recent work [MA07], we plan to investigate more compact representations, including methods for finding optimal sets of examples and handles.

Acknowledgements

We are thankful for the human character in Fig. 8 provided by Verónica Costa Orvalho from FaceInMotion and the cartoonish character animated by Christian Bocquee and modelled by Jason Pierce. This research was supported by the NCCR Co-Me of the Swiss National Science Foundation.

References

[BBA*07] BICKEL B., BOTSCH M., ANGST R., MATUSIK W., OTADUY M. A., PFISTER H., GROSS M.: Multi-scale capture of facial geometry and motion. *ACM Trans. Graph. (Proc. SIGGRAPH) 26*, 3 (2007), 33.1–33.10.

[BBPV03] BLANZ V., BASSO C., POGGIO T., VETTER T.: Reanimating faces in images and video. *Computer Graphics Forum 22*, 3 (2003), 641–650.

[BMW*06] BORSHUKOV G., MONTGOMERY J., WERNER W., RUFF B., LAU J., THURIOT P., MOONEY P., VAN NIEKERK S., RAPOSO D., DUPRAT J.-L., HABLE J., KIHLSTRÖM H., ROIZMAN D., NOONE K., O'CONNELL J.: Playable universal capture. In *ACM SIGGRAPH 06 Sketches & Applications* (2006).

[BV99] BLANZ V., VETTER T.: A morphable model for the synthesis of 3D faces. In *Proc. SIGGRAPH 99* (1999), pp. 187–194.

[CBC*01] CARR J. C., BEATSON R. K., CHERRIE J. B., MITCHELL T. J., FRIGHT W. R., MCCALLUM B. C., EVANS T. R.: Reconstruction and representation of 3D objects with radial basis functions. In *Proc. SIGGRAPH 01* (2001), pp. 67–76.

[CXH03] CHAI J.-X., XIAO J., HODGINS J.: Vision-based control of 3D facial animation. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2003), pp. 193–206.

[dLE07] D'EON E., LUEBKE D., ENDERTON E.: Efficient rendering of human skin. In *Eurographics Symposium on Rendering* (2007), pp. 147–157.

[DM00] DECARLO D., METAXAS D.: Optical flow constraints on deformable models with applications to face tracking. *International Journal of Computer Vision 38*, 2 (2000), 99–127.

[EBDP96] ESSA I., BASU S., DARRELL T., PENTLAND A.: Modeling, tracking and interactive animation of faces and heads: Using input from video. In *Proc. of Computer Animation 96* (1996), pp. 68–79.

[For03] FORDHAM J.: Middle earth strikes back. *Cinefex 92* (2003), 71–142.

[GMP*06] GOLOVINSKIY A., MATUSIK W., PFISTER H., RUSINKIEWICZ S., FUNKHOUSER T.: A statistical model for synthesis of detailed facial geometry. *ACM Trans. Graph. (Proc. SIGGRAPH) 25*, 3 (2006), 1025–1034.

[JTDP03] JOSHI P., TIEN W. C., DESBRUN M., PIGHIN F.: Learning controls for blend shape based realistic facial animation. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2003), pp. 187–192.

[KGC*96] KOCH R. M., GROSS M. H., CARLS F. R., VON BÜREN D. F., FANKHAUSER G., PARISH Y.: Simulating facial surgery using finite element methods. In *Proc. SIGGRAPH 96* (1996), pp. 421–428.

[KJP02] KRY P., JAMES D. L., PAI D. K.: EigenSkin: Real-time large deformation character skinning in hardware. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2002), pp. 153–159.

[KM04] KURIHARA T., MIYATA N.: Modeling deformable human hands from medical images. In *Proc. of ACM SIGGRAPH / Eurographics Symposium on Computer Animation* (2004), pp. 357–366.

[LCF00] LEWIS J. P., CORDNER M., FONG N.: Pose space deformations: A unified approach to shape interpolation and skeleton-driven deformation. In *Proc. SIGGRAPH 00* (2000), pp. 165–172.

[MA07] MEYER M., ANDERSON J.: Key point subspace acceleration and soft caching. *ACM Trans. Graph. (Proc. SIGGRAPH) 26*, 3 (2007), 74.

[MDSB03] MEYER M., DESBRUN M., SCHRÖDER P., BARR A. H.: Discrete differential-geometry operators for triangulated 2-manifolds. In *Visualization and Mathematics III*. Springer-Verlag, Heidelberg, 2003, pp. 35–57.

[MHP*07] MA W.-C., HAWKINS T., PEERS P., CHABERT C.-F., WEISS M., DEBEVEC P.: Rapid acquisition of specular and diffuse normal maps from polarized spherical gradient illumination. In *Proc. Eurographics Symposium on Rendering* (2007), pp. 183–194.

[MTKL*02] MAGNENAT-THALMANN N., KALRA P., LÉVÊQUE J. L., BAZIN R., BATISSE D., QUELEUX B.: A computational skin model: fold and wrinkle formation. *IEEE Trans. on Information Technology in Biomedicine 6*, 4 (2002), 317–323.

[MTLT88] MAGNENAT-THALMANN N., LAPERRIERE R., THALMANN D.: Joint-dependent local deformations for hand animation and object grasping. In *Proc. of Graphics Interface* (1988), pp. 317–323.

[NN99] NOH J.-Y., NEUMANN U.: *A Survey of Facial Modeling and Animation Techniques*. Tech. Rep. USC-TR-99-705, Univeristy of Southern Californina, 1999.

[NN01] NOH J.-Y., NEUMANN U.: Expression cloning. In *Proc. SIGGRAPH 01* (2001), pp. 277–288.

[NVI] NVIDIA: http://developer.nvidia.com/cuda.

[Par74] PARKE F. I.: *A parametric model for human faces*. PhD thesis, University of Utah, Salt Lake City, Utah, Dec. 1974.

[PHL*98] PIGHIN F., HECKER J., LISCHINSKI D., SZELISKI R., SALESIN D.: Synthesizing realistic facial expressions from photographs. In *Computer Graphics* (1998), pp. 75–84.

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

[PSS99] PIGHIN F. H., SZELISKI R., SALESIN D.: Resynthesizing facial animation through 3D model-based tracking. In *International Conference on Computer Vision (ICCV)* (1999), pp. 143–150.

[RLN06] RHEE T., LEWIS J. P., NEUMANN U.: Real-time weighted pose-space deformation on the GPU. *Computer Graphics Forum (Proc. Eurographics) 25*, 3 (2006), 439–448.

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

[SP04] SUMNER R. W., POPOVIĆ J.: Deformation transfer for triangle meshes. *ACM Trans. Graph. (Proc. SIGGRAPH) 23*, 3 (2004), 399–405.

[TCR] TOLEDO S., CHEN D., ROTKIN V.: Taucs: A library of sparse linear solvers. http://www.tau.ac.il/∼stoledo/taucs.

[VBPP05] VLASIC D., BRAND M., PFISTER H., POPOVIĆ J.: Face transfer with multilinear models. *ACM Trans. Graph. (Proc. SIGGRAPH) 24*, 3 (2005), 426–433.

[VLR05] VENKATARAMAN K., LODHA S., RAGHAVAN R.: A kinematic-variational model for animating skin with wrinkles. *Computers & Graphics 29*, 5 (2005), 756–770.

[WKMT96] WU Y., KALRA P., MAGNENAT-THALMANN N.: Simulation of static and dynamic wrinkles of skin. In *Proc. of Computer Animation* (1996), pp. 90–97.

[WPP07] WANG R. Y., PULLI K., POPOVIĆ J.: Real-time enveloping with rotational regression. *ACM Trans. Graph. (Proc. SIGGRAPH) 26*, 3 (2007), 73.

[WSLG07] WEBER O., SORKINE O., LIPMAN Y., GOTSMAN C.: Context-aware skeletal shape deformation. *Computer Graphics Forum (Proc. Eurographics) 26*, 3 (2007), 265–274.

[ZS05] ZHANG Y., SIM T.: Realistic and efficient wrinkle simulation using an anatomy-based face model with adaptive refinement. In *Computer Graphics International* (2005), pp. 3–10.

[ZSCS04] ZHANG L., SNAVELY N., CURLESS B., SEITZ S. M.: Spacetime faces: High resolution capture for modeling and animation. *ACM Trans. Graph. (Proc. SIGGRAPH) 23*, 3 (2004), 548–558.