Face Transfer with Multilinear Models

Daniel Vlasic* † Matthew Brand † Hanspeter Pfister Jovan Popović

Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology

† Mitsubishi Electric Research Laboratories

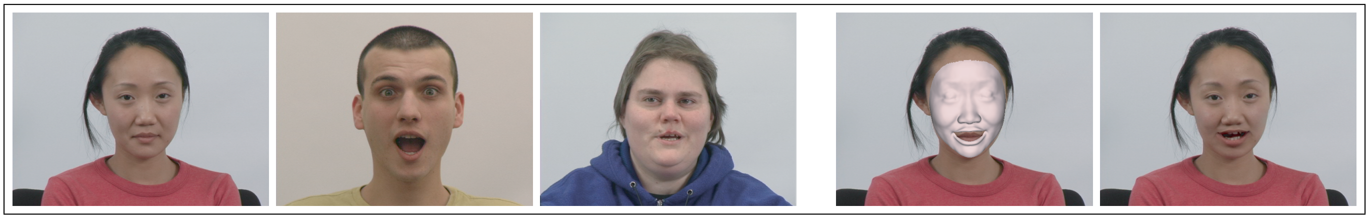

Figure 1: Face Transfer with multilinear models gives animators decoupled control over facial attributes such as identity, expression, and viseme. In this example, we combine pose and identity from the first frame, surprised expression from the second, and a viseme (mouth articulation for a sound midway between ”oo” and ”ee”) from the third. The resulting composite is blended back into the original frame.

Abstract

Face Transfer is a method for mapping videorecorded performances of one individual to facial animations of another. It extracts visemes (speech-related mouth articulations), expressions, and three-dimensional (3D) pose from monocular video or film footage. These parameters are then used to generate and drive a detailed 3D textured face mesh for a target identity, which can be seamlessly rendered back into target footage. The underlying face model automatically adjusts for how the target performs facial expressions and visemes. The performance data can be easily edited to change the visemes, expressions, pose, or even the identity of the target—the attributes are separably controllable. This supports a wide variety of video rewrite and puppetry applications.

Face Transfer is based on a multilinear model of 3D face meshes that separably parameterizes the space of geometric variations due to different attributes (e.g., identity, expression, and viseme). Separability means that each of these attributes can be independently varied. A multilinear model can be estimated from a Cartesian product of examples (identities × expressions × visemes) with techniques from statistical analysis, but only after careful preprocessing of the geometric data set to secure one-to-one correspondence, to minimize cross-coupling artifacts, and to fill in any missing examples. Face Transfer offers new solutions to these problems and links the estimated model with a face-tracking algorithm to extract pose, expression, and viseme parameters.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation; I.4.9 [Image Processing and Computer Vision]: Applications;

Keywords: Facial Animation, Computer Vision—Tracking

1 Introduction

Performance-driven animation has a growing role in film production because it allows actors to express content and mood naturally, and because the resulting animations have a degree of realism that is hard to obtain from synthesis methods [Robertson 2004]. The search for the highest quality motions has led to complex, expensive, and hard-to-use systems. This paper introduces new techniques for producing compelling facial animations that are inexpensive, practical, versatile, and well suited for editing performances and retargeting to new characters.

Face Transfer extracts performances from ordinary video footage, allowing the transfer of facial action of actors who are unavailable for detailed measurement, instrumentation, or for rerecording with specialized scanning equipment. Expressions, visemes (speech-related mouth articulations), and head motions are extracted automatically, along with a performance-driven texture function. With this information in hand, our system can either rewrite the original footage with adjusted expressions and visemes or transfer the performance to a different face in a different footage.

Multilinear models are ideally suited for this application because they can describe face variations with separable attributes that can be estimated from video automatically. In this paper, we estimate such a model from a data set of three-dimensional (3D) face scans that vary according to expression, viseme, and identity. The multilinear model decouples the three attributes (i.e., identity or viseme can be varied while expression remains constant) and encodes them consistently. Thus the attribute vector that encodes a smile for one person encodes a smile for every face spanned by the model, regardless of identity or viseme. Yet the model captures the fact that every person smiles in a slightly different way. Separability and consistency are the key properties that enable the transfer of a performance from one face to another without a change in content.

Contributions. This paper describes a general, controllable, and practical system for facial animation. It estimates a multilinear model of human faces by examining geometric variations between 3D face scans. In principle, given a large and varied data set, the model can generate any face, any expression, any viseme. As proof of concept, we estimate the model from a couple of geometric data sets: one with 15 identities and 10 expressions, and another with

[*] MIT CSAIL, The Stata Center, 32 Vassar Street, Cambridge, MA 02139, USA

16 identities, 5 expressions, and 5 visemes. Existing estimation algorithms require perfect one-to-one correspondence between all meshes, and a mesh for every possible combination of expression, viseme, and identity. Because acquiring the full Cartesian product of meshes and putting them into dense correspondence is extremely difficult, this paper introduces methods for populating the Cartesian product from a sparse sampling of faces, and for placing unstructured face scans into correspondence with minimal cross-coupling artifacts.

By linking the multilinear model to optical flow, we obtain a single-camera tracker that estimates performance parameters and detailed 3D geometry from video recordings. The model defines a mapping from performance parameters back to 3D shape, thus we can arbitrarily mix pose, identity, expressions, and visemes from two or more videos and render the result back into a target video. As a result, the system provides an intuitive interface for both animators (via separably controllable attributes) and performers (via acting). And because it does not require performers to wear visible facial markers or to be recorded by special face-scanning equipment, it is an inexpensive and easy-to-use facial animation system.

2 Related Work

Realistic facial animation remains a fundamental challenge in computer graphics. Beginning with Parke's pioneering work [1974], desire for improved realism has driven researchers to extend geometric models [Parke 1982] with physical models of facial anatomy [Waters 1987; Lee et al. 1995] and to combine them with non-linear finite element methods [Koch et al. 1996] in systems that could be used for planning facial surgeries. In parallel, Williams presented a compelling argument [1990] in favor of performance-driven facial animation, which anticipated techniques for tracking head motions and facial expressions in video [Li et al. 1993; Essa et al. 1996; DeCarlo and Metaxas 1996; Pighin et al. 1999]. A more expensive alternative could use a 3D scanning technique [Zhang et al. 2004], if the performance can be re-recorded with such a system.

Much of the ensuing work on face estimation and tracking relied on the observation that variation in faces is well approximated by a linear subspace of low dimension [Sirovich and Kirby 1987]. These techniques estimate either linear coefficients for known basis shapes [Bascle and Blake 1998; Brand and Bhotika 2001] or both the basis shapes and the coefficients, simultaneously [Bregler et al. 2000; Torresani et al. 2001]. In computer graphics, the combination of accurate 3D geometry with linear texture models [Pighin et al. 1998; Blanz and Vetter 1999] produced striking results. In addition, Blanz and Vetter [1999] presented a process for estimating the shape of a face in a single photograph, and a set of controls for intuitive manipulation of appearance attributes (thin/fat, feminine/masculine).

These and other estimation techniques share a common challenge of decoupling the attributes responsible for observed variations. As an early example, Pentland and Sclaroff estimate geometry of deformable objects by decoupling linear elastic equations into orthogonal vibration modes [1991]. In this case, modal analysis uses eigen decomposition to compute the independent vibration modes. Similar factorizations are also relied upon to separate variations due to pose and lighting, pose and expression, identity and lighting, or style and content in general [Freeman and Tenenbaum 1997; Bregler et al. 2000; DeCarlo and Metaxas 2000; Georghiades et al. 2001; Cao et al. 2003].

A technical limitation of these formulations is that each pair of factors must be considered in isolation; they cannot easily decouple variations due to a combination of more than two factors. The extension of such two-mode analysis to more modes of variation was first introduced by Tucker [1966] and later formalized and improved on by Kroonenberg and de Leeuw [1980]. These techniques were successfully applied to multilinear analysis of images [Vasilescu and Terzopoulos 2002; Vasilescu and Terzopoulos 2004].

This paper describes multilinear analysis of *three-dimensional (3D)* data sets and generalizes face-tracking techniques to create a unique performance-driven system for animation of any face, any expression, and any viseme. In consideration of similar needs, Bregler and colleagues introduced a two-dimensional method for transferring mouth shapes from one performance to another [1997]. The method is ideal for film dubbing—a problem that could also be solved without performance by first learning the mouth shapes on a canonical data set and then generating new shapes for different texts [Ezzat and Poggio 2000]. These methods are difficult to use for general performance-driven animation because they cannot change emotions of a face. Although the problem can be resolved by decoupling emotion and content via two-mode analysis [Chuang et al. 2002], all three techniques are view specific, which presents difficulties when view, illumination, or both have to change.

Our Face Transfer learns a model of 3D facial geometry variations in order to infer a particular face shape from 2D images. Previous work combines identity and expression spaces by copying deformations from one subject onto the geometry of other faces [DeCarlo and Metaxas 2000; Blanz et al. 2003; Chai et al. 2003]. Expression cloning [Noh and Neumann 2001; Sumner and Popović 2004] improves on this process but does not account for actor-specific idiosyncrasies that can be revealed by statistical analysis of the entire data set (i.e., the mesh vertex displacements that produce a smile should depend on who is smiling and on what they are saying at the same time). Other powerful models of human faces have been explored [Wang et al. 2004] at the cost of making the estimation and transfer of model parameters more difficult. This paper describes a method that incorporates all such information through multilinear analysis, which naturally accommodates variations along multiple attributes.

3 Multilinear Algebra

Multilinear algebra is a higher order generalization of linear algebra. In this section we provide insight behind the basic concepts needed for understanding of our Face Transfer system. De Lathauwer's dissertation [1997] provides a comprehensive treatment of this topic. Concise overviews have also been published in the graphics and vision literature [Vasilescu and Terzopoulos 2002; Vasilescu and Terzopoulos 2004].

Tensors. The basic mathematical object of multilinear algebra is the tensor, a natural generalization of vectors (1^{st} order tensors) and matrices (2^{nd} order tensors) to multiple indices. An N^{th}-order tensor can be thought of as a block of data indexed by N indices: $\mathscr{T} = (t_{i_1 i_2 \ldots i_N})$. Figure 2 shows a 3^{rd}-order (or 3-mode) tensor with a total of $d_1 \times d_2 \times d_3$ elements. Different modes usually correspond to particular attributes of the data (e.g, expression, identity, etc.).

Mode Spaces. A matrix has two characteristic spaces, row and column space; a tensor has one for each mode, hence we call them *mode spaces*. The $d_1 \times d_2 \times d_3$ 3-tensor in Figure 2 has three mode spaces. Viewing the data as a set of d_1-dimensional vectors stored parallel to the first axis (Figure 2b), we can define the mode-1 space as the span of those vectors. Similarly, mode-2 space is defined as the span of the vectors stored parallel to the second axis, each of size d_2 (Figure 2c). Finally, mode-3 space is spanned by vectors in the third mode, of dimensionality d_3 (Figure 2d). Multilinear algebra revolves around the analysis and manipulation of these spaces.

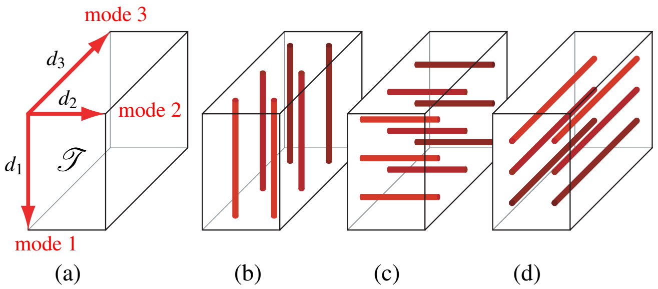

Figure 2: In (a) we show a 3^{rd}-order (3-mode) tensor $\mathscr{T}$ whose modes have d_1, d_2, and d_3 elements respectively. Depending on how we look at the data within the tensor, we can identify three mode spaces. By viewing the data as vectors parallel to the first mode (b), we define mode-1 space as the span of those vectors. Similarly, mode-2 space is spanned by vectors parallel to the second mode (c), and mode-3 space by vectors in the third mode (d).

Mode-*n* Product. The most obvious way of manipulating mode spaces is via linear transformation, officially referred to as the *mode-n product*. It is defined between a tensor $\mathscr{T}$ and a matrix **M** for a specific mode n, and is written as a multiplication with a subscript: $\mathscr{T} \times_n \mathbf{M}$. This notation indicates a linear transformation of vectors in $\mathscr{T}$'s mode-n space by the matrix **M**. Concretely, $\mathscr{T} \times_2 \mathbf{M}$ would replace each mode-2 vector **v** (Figure 2c) with a transformed vector **Mv**.

Tensor Decomposition. One particularly useful linear transformation of mode data is the N-mode singular value decomposition (N-mode SVD). It rotates the mode spaces of a *data tensor* $\mathscr{T}$ producing a *core tensor* $\mathscr{C}$, whose variance monotonically decreases from first to last element in each mode (analogous to matrix SVD). This enables us to truncate the insignificant components and get a reduced model of our data.

Mathematically, N-mode SVD can be expressed with mode products

$$\mathscr{T} \times_1 \mathbf{U}_1^\top \times_2 \mathbf{U}_2^\top \times_3 \mathbf{U}_3^\top \cdots \times_N \mathbf{U}_N^\top = \mathscr{C} \quad (1)$$

$$\Longrightarrow \quad \mathscr{T} = \mathscr{C} \times_1 \mathbf{U}_1 \times_2 \mathbf{U}_2 \times_3 \mathbf{U}_3 \cdots \times_N \mathbf{U}_N, \quad (2)$$

where $\mathscr{T}$ is the data tensor, $\mathscr{C}$ is the core tensor, and $\mathbf{U}_i$'s (or more precisely their transposes) rotate the mode spaces. Each $\mathbf{U}_i$ is an orthonormal matrix whose columns contain left singular vectors of the ith mode space, and can be computed via regular SVD of those spaces [De Lathauwer 1997]. Since variance is concentrated in one corner of the core tensor, data can be approximated by

$$\mathscr{T} \simeq \mathscr{C}_{reduced} \times_1 \check{\mathbf{U}}_1 \times_2 \check{\mathbf{U}}_2 \times_3 \check{\mathbf{U}}_3 \cdots \times_N \check{\mathbf{U}}_N \,, \quad (3)$$

where $\check{\mathbf{U}}_i$'s are truncated versions of $\mathbf{U}_i$'s with last few columns removed. This truncation generally yields high quality approximations but it is not optimal—one of several matrix-SVD properties that do not generalize in multilinear algebra. One can obtain a better approximation with further refinement of $\check{\mathbf{U}}_i$'s and $\mathscr{C}_{reduced}$ via alternating least squares [De Lathauwer 1997].

4 Multilinear Face Model

To construct the multilinear face model, we first acquire a range of 3D face scans, put them in full correspondence, appropriately arrange them into a data tensor (Figure 3), and use the N-mode SVD to compute a model that captures the face geometry and its variation due to attributes such as identity and expression.

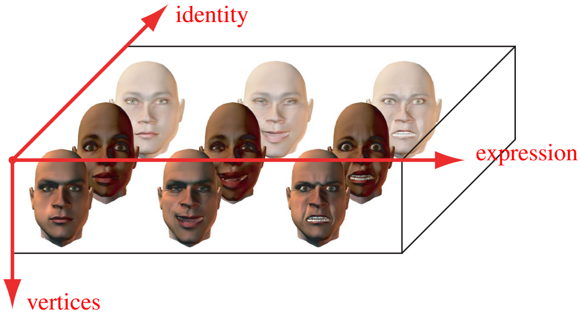

Figure 3: Data tensor for a bilinear model that varies with identity and expression; the first mode contains vertices, while the second and third modes correspond to expression and identity respectively. The data is arranged so that each slice along the second mode contains the same expression (in different identities) and each slice along the third mode contains the same identity (in different expressions). In our trilinear experiments we have added a fourth mode, where scans in each slice share the same viseme.

4.1 Face Data

We demonstrate our proof-of-concept system on two separate face models: a bilinear model, and a trilinear model. Both were estimated from detailed 3D scans ($\sim$ 30K vertices) acquired with 3dMD/3Q's structured light scanner (`http://www.3dmd.com/`) in a process similar to regular flash photography, although our methods would apply equally to other geometric data sets such as motion capture. As a preprocess, the scans were smoothed using the bilateral filter [Jones et al. 2003] to eliminate some of the capture noise. The subject pool included men, women, Caucasians, and Asians, from the mid-20s to mid-50s.

Bilinear model. 15 subjects were scanned performing the same 10 facial expressions. The expressions were picked for their familiarity as well as distinctiveness, and include neutral, smile, frown, surprise, anger, and others. The scans were assembled into a third order (3-mode) data tensor (30K vertices × 10 expressions × 15 identities). After N-mode SVD reduction, the resulting bilinear model offers 6 knobs for manipulating expression and 9 for identity.

Trilinear model. 16 subjects were asked to perform 5 visemes in 5 different expressions (neutral, smiling, scowling, surprised, and sad). The visemes correspond to the boldfaced sounds in **m**an, c**ar**, **ee**l, **too**, and **sh**e. Principal components analysis of detailed speech motion capture indicated that these five expressions broadly span the space of lip shapes, and should give a good approximate basis for all other visemes—with the possible exception of exaggerated fricatives. The resulting fourth order (4-mode) data tensor (30K vertices × 5 visemes × 5 expressions × 16 identities) was decomposed to yield a trilinear model providing 4 knobs for viseme, 4 for expression, and 16 for identity (we have kept the number of knobs large since our data sets were small).

4.2 Correspondence

Training meshes that are not placed in perfect correspondence can considerably muddle the question of how to displace vertices to change one attribute versus another (e.g. identity versus expression), and thus the multilinear analysis may not give a model with good separability. We show here how to put a set of unstructured face scans into correspondence suitable for multilinear analysis.

Despite rapid advances in automatic parameterization of meshes (e.g., [Praun and Hoppe 2003; Gotsman et al. 2003]), it took considerable experimentation to place many facial scans into detailed correspondence. The principal complicating factors are that the scans do not have congruent mesh boundaries, and the problem of matching widely varied lip deformations does not appear to be well served by conformal maps or local isometric constraints. This made it necessary to mark a small number of feature points in order to bootstrap correspondence-finding across large deformations.

We developed a protocol for a template-fitting procedure [Allen et al. 2003; Sumner and Popović 2004], which seeks a minimal deformation of a parameterized template mesh that fits the surface implied by the scan. The optimization objective, minimized with gradient descent, balances overall surface similarity, proximity of manually selected feature points on the two surfaces, and proximity of reference vertices to the nearest point on the scanned surface. We manually specified 42 reference points on a reference facial mesh and on a neutral (m-viseme) scan. After rigidly aligning the template and the scan with Procrustes' alignment, we deformed the template mesh into the scan: at first, weighing the marked correspondences heavily and afterwards emphasizing vertex proximity. For the trilinear model, the remaining m-viseme (closed-mouth) scans were marked with 21 features around eyebrows and lips, rigidly aligned to upper-face geometry on the appropriate neutral scans, and then non-rigidly put into correspondence as above. Finally, all other viseme scans were similarly put into correspondence with the appropriate closed-mouth scan, using the 18 features marked around the lips.

4.3 Face Model

Equation (3) shows how to approximate the data tensor by mode-multiplying a smaller core tensor with a number of truncated orthogonal matrices. Since our goal is to output vertices as a function of attribute parameters, we can decompose the data tensor without factoring along the mode that corresponds to vertices (mode-1), changing Equation (3) to:

$$\mathscr{T} \simeq \mathscr{M} \times_2 \check{\mathbf{U}}_2 \times_3 \check{\mathbf{U}}_3 \cdots \times_N \check{\mathbf{U}}_N \,, \tag{4}$$

where $\mathscr{M}$ can now be called the *multilinear model* of face geometry. Mode-multiplying $\mathscr{M}$ with $\check{\mathbf{U}}_i$'s approximates the original data. In particular, mode-multiplying it with one row from each $\check{\mathbf{U}}_i$ reconstructs exactly one original face (the one corresponding to the attribute parameters contained in that row). Therefore, to generate an arbitrary interpolation (or extrapolation) of original faces, we can mode-multiply the model with a linear combination of rows for each $\check{\mathbf{U}}_i$. We can write

$$\mathbf{f} = \mathscr{M} \times_2 \mathbf{w_2}^\top \times_3 \mathbf{w_3}^\top \cdots \times_N \mathbf{w_N}^\top \,, \tag{5}$$

where $\mathbf{w_i}$ is a column vector of parameters (weights) for the attribute corresponding to i^{th} mode, and $\mathbf{f}$ is a column vector of vertices describing the resulting face.

4.4 Missing Data

Building the multilinear model from a set of face scans requires capturing the full Cartesian product of different face attributes, (i.e., all expressions and visemes need to be captured for each person). Producing a full data tensor is not always practical for large data sets. For example, a certain person might have trouble performing some expressions on cue, or a researcher might add a new expression to the database but be unable reach all the previous subjects. In our case, data corruption and subsequent unavailability of a subject led to an incomplete tensor. The problem becomes more evident if we add *age* as one of the attributes, where we cannot expect to scan each individual throughout their entire lives. In all these cases, we would still like to include a person's successful scans in the model, and fill in the missing ones with the most likely candidates. This process is known as imputation.

There are many possible schemes for estimating a model from incomplete data. A naive imputation would find a complete subtensor, use it to estimate a smaller model, use that to predict a missing face, use that to augment the data set, and repeat. In a more sophisticated Bayesian setting, we would treat the missing data as hidden variables to be MAP estimated (imputed) or marginalized out. Both approaches require many iterations over a huge data set; Bayesian methods are particularly expensive and generally require approximations for tractability. With MAP estimation and naive imputation, the results can be highly dependent on the order of operations. Because it fails to exploit all available constraints, the naive imputative scheme generally produces inferior results.

Here we use an imputative scheme that exploits more available constraints than the naive one, producing better results. The main intuition, which we formalize below, is that any optimization criteria can be linearized in a particular tensor mode, where it yields a matrix factorization problem with missing values. Then we leverage existing factorization schemes for incomplete matrices, where known values contribute a set of linear constraints on the missing values. These constraints are then combined and solved in the least-squares sense.

Description. Our algorithm consists of two steps. First, for each mode we assemble an *incomplete* matrix whose columns are the corresponding mode vectors. We then seek a subspace decomposition that best reconstructs the known values of that matrix. The decomposition and the known values provide a set of linear constraints for the missing values. This can be done with off-the-shelf imputative matrix factorizations (e.g., PPCA [Tipping and Bishop 1999], SPCA [Roweis 1997], or ISVD [Brand 2002]). Typically these algorithms estimate a low-rank subspace from the complete vectors of the mode and use that to predict missing values in the incomplete columns (and/or update the subspace). In our experiments we used the standard PPCA formulation for filling in missing values, which reduces to a system of linear equations that relate unknown values to known values through the estimated mean and covariance of the vectors in the mode space. Second, the linear constraints are combined through the missing elements, because they are shared across all groups of modal vectors and must be filled in with consistent values. To that end, we collect the linear equations that determine a particular missing value in all the modes, and solve them together. For example, if two missing values co-occur in some mode vector, then they must be jointly estimated. We update the mean and covariance for each decomposition and repeat the two steps until convergence.

Evaluation. Figure 4 contrasts the results of this method with faces predicted by our generalization of the simple method proposed by Blanz and colleagues [2003]. In their formulation the same displacement vectors that make one person smile are copied over onto every other identity. Because our data set includes smiles for more than one person, we extend that approach to copy their average. In this particular example, 15% of real faces where held out of the trilinear data set and predicted by our imputation scheme and the simple averaging scheme. Note how the multilinear prediction is closer to the truth in most examples, even predicting some individual idiosyncrasies in puckers and smiles. The simple averaging scheme, however, seems to do a better job at keeping the lips sealed for closed-mouth faces (bottom row of Figure 4). We could obtain better results by preferentially weighting detail around the mouth.

In our earlier trilinear experiments, we found that ISVD-based imputations predicted how faces vary from the mean with less than 9% relative error (Frobenius norm of the total error divided by the

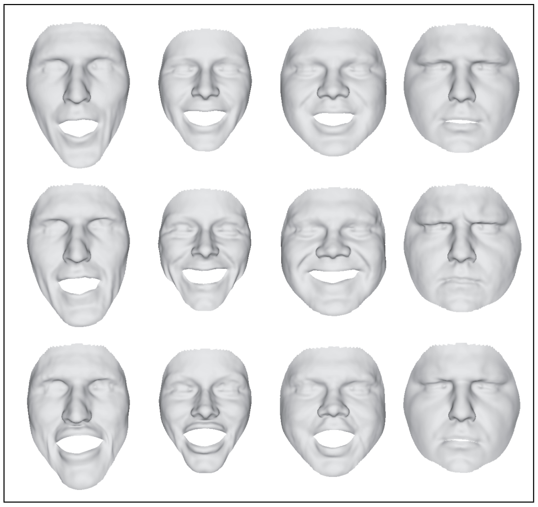

Figure 4: From top to bottom: Prediction of held-out faces with our imputation scheme (on the trilinear model), the actual face, and a simple averaging scheme.

norm of the held-out face variations) for up to 50% missing data. In general, the predictions are drawn towards the mean of the known data. Closed-mouth expressions, which are under-represented in our data and thus lie far from the mean, were not predicted as well as other expressions. That can be fixed by reweighting the data. Tests performed on synthetic data indicate that the quality of imputation increases as the data set grows in size, even if significant portions of it are missing. The reason why is that if the data is truly low-dimensional in each of the modes, the missing samples will fall within the span and density of the known ones.

Probabilistic Interpretation. The above algorithm fills in missing data by approximating the true multilinear distribution. The form of this approximation is made precise by a probabilistic interpretation, which starts from a multilinear generative model

$$\mathscr{T} = \mathscr{M} \times_2 \check{\mathbf{U}}_2 \times_3 \check{\mathbf{U}}_3 \cdots \times_N \check{\mathbf{U}}_N + \nu,$$

where $\mathscr{T}$ and $\mathscr{M}$ are the data and model tensors, $\check{\mathbf{U}}_i$ is the i-th modal subspace, and ν is a Gaussian noise source. Filling in missing data according to this model is computationally expensive. Instead, we approximate the true likelihood with a geometric average of Gaussians

$$p(\mathscr{T}|\mathscr{M},\{\check{\mathbf{U}}_i\}_{i=2}^N) \approx \prod_{j=2}^{N} q_j(\mathscr{T},\mathscr{M},\{\check{\mathbf{U}}_i\}_{i=2}^N)^{1/N}.$$

Each Gaussian $q_j(\mathscr{T},\mathscr{M},\{\check{\mathbf{U}}_i\}_{i=2}^N) \doteq \mathscr{N}(\mathscr{T}|\check{\mathbf{U}}_j\mathbf{J}_j,\sigma_j^2)$ is found by fixing $\{\check{\mathbf{U}}_i\}_{i\neq j}$ and turning the tensor Equation (4) into matrix form: $\mathbf{T}_j = \check{\mathbf{U}}_j\mathbf{J}_j$. Here, columns of $\mathbf{T}_j$ are the mode-j vectors of $\mathscr{T}$, and the columns of $\mathbf{J}_j$ are the mode-j vectors of $\mathscr{M} \times_2 \check{\mathbf{U}}_2 \cdots \times_{j-1} \check{\mathbf{U}}_{j-1} \times_{j+1} \check{\mathbf{U}}_{j+1} \cdots \times_N \check{\mathbf{U}}_N$. The resulting likelihood becomes:

$$p(\mathscr{T}|\mathscr{M},\{\check{\mathbf{U}}_i\}_{i=2}^N) \approx \prod_{j=2}^{N} \mathscr{N}(\mathscr{T}|\check{\mathbf{U}}_j\mathbf{J}_j,\sigma_j^2)^{1/N},$$

which can be maximized efficiently.

Taking logarithms and discarding constant factors such as N and σ_j, we seek to minimize the sum-squared error

$$\sum_{j=2}^{N} \|\mathbf{T}_j - \check{\mathbf{U}}_j\mathbf{J}_j\|_F^2$$

Each term of the summation presents a matrix factorization problem with missing values, where $\check{\mathbf{U}}_j$ and $\mathbf{J}_j$ are treated as unknown factors of the incomplete matrix $\mathbf{T}_j$, and are solved for using PPCA as described above.

5 Face Transfer

One produces animations from a multilinear model by varying the attribute parameters (the elements of the $\mathbf{w}_i$'s) as if they were dials, and generating mesh coordinates from Equation 5. The N-mode SVD conveniently gives groups of dials that separately control identity, expression and viseme. Within each group, the dials do not correspond to semantically meaningful deformations (such as smile or frown), but rather reflect the deformations that account for most variance. However, the dials can be "tuned" to reflect deformations of interest through a linear transform of each $\mathbf{w}_i$. This approach was successfully applied in [Allen et al. 2003] to make their body shape dials correspond to height and weight. A similar linear scheme was employed in [Blanz and Vetter 1999]. In general, dial-based systems are currently used on most of the deformable models in production, but only skilled animators can create believable animations (or even stills) with them. To give similar power to a casual user, we have devised a method that automatically sets model parameters from given video data. With this tool, a user can enact a performance in front of a camera, and have it automatically transferred to the model.

5.1 Face Tracking

To link the parameters of a multilinear model to video data, we use optical flow in conjunction with the weak-perspective camera model. Using the symmetric Kanade-Lucas-Tomasi formulation [Birchfield 1996], we express the frame-to-frame motion of a tracked point with a linear system:

$$\mathbf{Zd} = \mathbf{Z}(\mathbf{p} - \mathbf{p}_0) = \mathbf{e}\,. \tag{6}$$

Here, the 2-vector $\mathbf{d}$ describes the image-space motion of the point, also expressed as the difference between the point's true location $\mathbf{p}$ and its current best guess $\mathbf{p}_0$ (if we have no guess, then $\mathbf{p}_0$ is the location from the previous frame). Matrix $\mathbf{Z}$ and vector $\mathbf{e}$ contain spatial and temporal intensity gradient information in the surrounding region [Birchfield 1996].

Using a weak-perspective imaging model, the point position $\mathbf{p}$ can be expanded in terms of rigid head-motion parameters and non-rigid facial shape parameters, which are constrained by the multilinear model:

$$\mathbf{Z}(s\mathbf{R}\mathbf{f}_i + \mathbf{t} - \mathbf{p}_0) = \mathbf{e}\,, \tag{7}$$

where the rigid parameters consist of scale factor s, the first two rows of a 3D rotation matrix $\mathbf{R}$, and the image-space translation $\mathbf{t}$. The 3D shape $\mathbf{f}$ comes from the multilinear model through Equation (5), with $\mathbf{f}_i$ indicating the ith 3D vertex being tracked.

Solving for the pose and all the multilinear weights from a pair of frames using Equation (7) is not a well-constrained problem. To simplify the computation, we use a coordinate-descent method: we let only one of the face attributes vary at a time by fixing all the others to their current guesses. This transforms the multilinear problem into a linear one, as described below, which we solve with standard techniques that simultaneously compute the rigid pose along with the linear weights from a pair of frames [Bascle and Blake 1998; Brand and Bhotika 2001].

When we fix all but one attribute of the multilinear model, thereby making $\mathbf{f}$ linear, Equation (7) turns into

$$\mathbf{Z}(s\mathbf{R}\mathbf{M}_{m,i}\mathbf{w}_m + \mathbf{t} - \mathbf{p_0}) = \mathbf{e}\,, \tag{8}$$

where m is the mode corresponding to the non-fixed attribute. $\mathbf{w}_m$ is a vector of weights for that attribute, and $\mathbf{M}_{m,i}$ is the corresponding linear basis for the tracked vertex i obtained from

$$\mathbf{M}_m = \mathscr{M} \times_2 \mathbf{w}_2^\top \cdots \times_{(m-1)} \mathbf{w}_{(m-1)}^\top \times_{(m+1)} \mathbf{w}_{(m+1)}^\top \cdots \times_N \mathbf{w}_N^\top . \quad (9)$$

To get a well constrained solution for the per-frame pose (scale, rotation, and translation) as well as the model's attribute parameters (expression, identity, etc.), we track a number of vertices and stack the resulting linear equations into one big system. For each pair of neighboring video frames we assemble a set of linear systems, each one applying Equation (8) to one of the tracked vertices. If the currently tracked attribute varies from frame to frame (such as expression does), we solve the set of linear systems and proceed to the next pair of neighboring frames. If, on the other hand, the attribute is constant across all frames (like identity), we accumulate the mentioned linear systems from each pair of frames and solve them together as one combined system. Solving the entire system for a pair of frames and a thousand tracked points completes in about a millisecond. Taking into account several levels of multi-scale and several passes through the whole video, the tracking process averages at one frame per second.

5.2 Initialization

The method described above, since it is based on tracking, needs to be initialized with the first frame alignment (pose and all the weights of the multilinear model). We accomplish this by specifying a small number of feature points which are then used to position the face geometry. The correspondences can be either user-provided (which gives more flexibility and power) or automatically detected (which avoids user intervention). We have experimented with the automatic feature detector developed by [Viola and Jones 2001], and found that it is robust and precise enough in locating a number of key features (eye corners, nose tip, mouth corners) to give a good approximating alignment in most cases. Imperfect alignment can be improved by tracking the first few frames back and forth until the model *snaps* into a better location. Other more powerful automated alignment approaches that take in account texture and lighting, such as the one described in [Blanz and Vetter 1999], could also be adapted to multilinear models.

6 Results

Multilinear models provide a convenient control of facial attributes. Figures 5 and 6 show example manipulations of our bilinear and trilinear models.

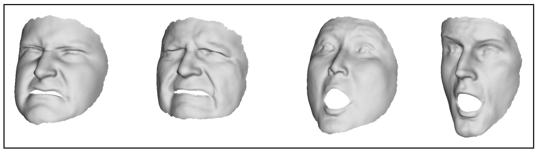

Figure 5: Several faces generated by manipulating the parameters of the bilinear model. The left two faces show our attempt of expressing disgust, an expression that was not in the database. They only differ in identity, to demonstrate the separability of our parameters. The right two faces show surprise for two novel identities, illustrating how the expression adjusts to identity.

Face Transfer infers the attribute parameters automatically by tracking the face in a video. Figure 7A shows a few example frames acquired by tracking a face outside of our data set. The resulting 3D shapes, shown below each frame, are generated by the bilinear model with the mouth shapes closed-off for texturing. Because our system tracks approximately a thousand vertices, the process is less sensitive to localized intensity changes (e.g., around the furrow above the lip). Once we obtain the 3D geometry, we can lift the texture from the video by assigning pixel colors to the corresponding mesh vertices. A simple performance-driven texture function can be obtained with the weighted sum of nearest neighbors.

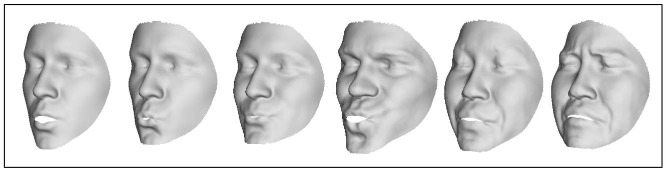

Figure 6: Faces generated by manipulating the parameters of the trilinear model. Left to right: producing the 'oo' sound, trying to whistle, breaking into a smile, two changes of identity, then adding a scowl.

All advantages of our system are combined in video rewrite applications, where a performance is lifted from a video, altered, and seamlessly rendered back into the video. In Figure 7B, we use the bilinear model to change a person's identity while retaining the expressions from the original performance. From left to right, the figure shows the original video, the recovered 3D shape, the modified 3D shape, and its textured overlay over the original video (without blending and with the simple texture model). Note how the style of smiling changes with identity. Figure 7C shows a post-production example, where a repeat performance is transferred onto the old footage. From left to right, we present the original video frame, a frame from a new video, the new geometry, and the final modified frame (without blending). Here, we combine the pose from the original video with the expression from the new video to modify original expressions.

Our most challenging face transfer uses the trilinear model to transfer expressions and visemes of a singing performance. In Figure 7D, the left two images show the frames from two input videos: a target video for a subject in our data set and a source video of a singing performance from a novel subject. The third image shows the final composite (along with the matching geometry as seen from two viewpoints) that combines the expressions and visemes from the source performance with the identity shown in the target video. For best visual results, we blend [Pérez et al. 2003] the face texture from the target video (constant throughout the sequence); the mouth texture from the source video; and the background texture, which includes the eyes peeking through the holes in the transferred geometry. In Figure 7E, we manually add a frown to the geometry in the final image; the frown texture was lifted from another frame of the same subject. With a similar technique, we can also combine facial attributes from several videos as shown in Figure 1, where the pose, expressions, and visemes are mixed from three different input videos.

7 Discussion

Perhaps our most remarkable empirical result is that even with a model estimated from a rather tiny data set, we can produce videorealistic results for new source and target subjects. Further improvements can be expected when we move to a wider variety of subjects and facial configurations.

We also see algorithmic opportunities to make aspects of the system more automatic and robust. Correspondence between scans might be improved with some of the methods shown in [Kraevoy and Sheffer 2004]. First-frame alignment would benefit from a successful application of a method such as [Blanz and Vetter 1999]. Optical flow-based tracking, which is inherently vulnerable

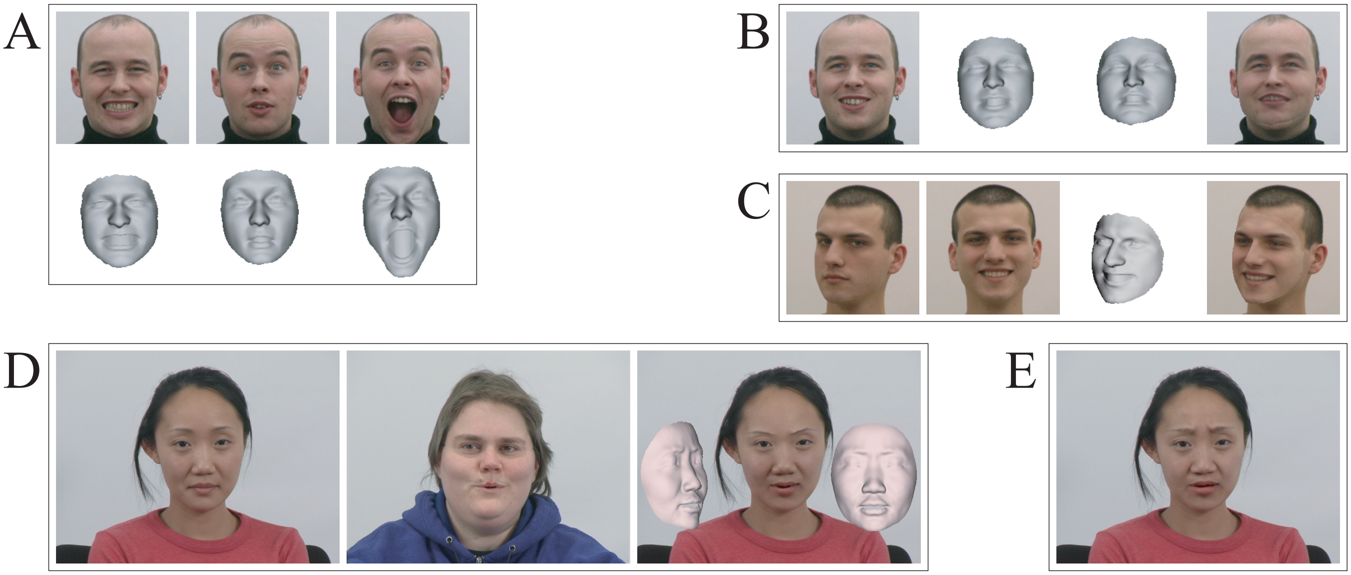

Figure 7: Several examples of our system: **(A)** A few frames of a bilinear-model tracking of a novel face and the corresponding 3D shapes below. **(B)** Changing the identity parameters of a performance tracked with the bilinear model. **(C)** Transferring a performance of a known identity from one video to another using the bilinear model. In each example the mouth gap was closed to allow for texturing. **(D)** Using the trilinear model to copy a singing performance from one video to another (left-to-right: original video, singing video, and the result). **(E)** Altering the performance from the previous example by adding a frown.

to imaging factors such as lighting, occlusions and specular reflections, can be made more robust with edge and corner constraints as demonstrated in [DeCarlo and Metaxas 1996].

In a production setting, the scan data would need to be expanded to contain shape and texture information for the ears, neck, and hair, so that we can make a larger range of head pose changes. Motions of eyes, eyelids, tongues, and teeth, are currently modeled in the texture function or not at all; this will either require more video data or a better solution. Finally, the texture function lifted from video is performance specific, in that we made no effort to remove variations due to lighting. Since we do estimate 3D shape, it may be possible to estimate and remove lighting, given sufficiently long videos.

8 Conclusion

To summarize, we have shown how to estimate a highly detailed face model from an incomplete set of face scans. The model is multilinear, and thus has the key property of separability: different attributes, such as identity and expression, can be manipulated independently. Thus we can change the identity and expression, but keep the smile. Even more useful, the new smile is in the style idiosyncratic to the new identity.

What makes this multilinear model a practical tool for animation is that we connect it directly to video, showing how to recover a time-series of poses and attribute parameters (expressions and visemes), plus a performance-driven texture function for an actor's face.

Our methods greatly simplify the editing of identity, performance, and facial texture in video, enabling video rewrite applications such as performance animation (puppetry) and actor replacement. In addition, the model offers a rich source of synthetic actors that can be controlled via video.

An intriguing prospect is that one could now build a multilinear model representing a vertex×identity×expression×viseme×*age* data tensor—without having to capture each individual's face at every stage of their life. The model would provide animators with control dials for each of the listed attributes, so they could change an actor's age along with their appearance and performance.

9 Acknowledgments

Funding for this work was provided by the MIT Oxygen Project. Our analysis was made possible by the many volunteers who agreed to have their faces scanned. The members of the MIT Graphics Group (in addition to their good looks) provided invaluable feedback throughout the entire project. Bob Sumner implemented the template-matching software used for correspondence. Ali Rahimi helped us formalize the probabilistic interpretation of our imputation algorithm. Ray Jones provided the bilateral filter code. Bryt Bradley provided a beautiful voice for our singing example. Tom Buehler shot and made all the videos.

References

ALLEN, B., CURLESS, B., AND POPOVIĆ, Z. 2003. The space of human body shapes: Reconstruction and parameterization from range scans. *ACM Transactions on Graphics 22*, 3 (July), 587–594.

BASCLE, B., AND BLAKE, A. 1998. Separability of pose and expression in facial tracking and animation. In *International Conference on Computer Vision (ICCV)*, 323–328.

BIRCHFIELD, S., 1996. KLT: An implementation of the kanade-lucas-tomasi feature tracker. http://www.ces.clemson.edu /∼stb/.

BLANZ, V., AND VETTER, T. 1999. A morphable model for the synthesis of 3D faces. In *Proceedings of SIGGRAPH 99*, Computer Graphics Proceedings, Annual Conference Series, 187–194.

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

BRAND, M. E., AND BHOTIKA, R. 2001. Flexible flow for 3D nonrigid tracking and shape recovery. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, 315–322.

BRAND, M. E. 2002. Incremental singular value decomposition of uncertain data with missing values. In *European Conference on Computer Vision (ECCV)*, vol. 2350, 707–720.

BREGLER, C., COVELL, M., AND SLANEY, M. 1997. Video rewrite: Driving visual speech with audio. In *Proceedings of SIGGRAPH 97*, Computer Graphics Proceedings, Annual Conference Series, 353–360.

BREGLER, C., HERTZMANN, A., AND BIERMANN, H. 2000. Recovering non-rigid 3D shape from image streams. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, 690–696.

CAO, Y., FALOUTSOS, P., AND PIGHIN, F. 2003. Unsupervised learning for speech motion editing. In *Eurographics/SIGGRAPH Symposium on Computer animation (SCA)*, 225–231.

CHAI, J.-X., XIAO, J., AND HODGINS, J. 2003. Vision-based control of 3D facial animation. In *Eurographics/SIGGRAPH Symposium on Computer Animation (SCA)*, 193–206.

CHUANG, E. S., DESHPANDE, H., AND BREGLER, C. 2002. Facial expression space learning. In *Pacific Conference on Computer Graphics and Applications (PG)*, 68–76.

DE LATHAUWER, L. 1997. *Signal Processing based on Multilinear Algebra.* PhD thesis, Katholieke Universiteit Leuven, Belgium.

DECARLO, D., AND METAXAS, D. 1996. The integration of optical flow and deformable models with applications to human face shape and motion estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 231–238.

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

ESSA, I., BASU, S., DARRELL, T., AND PENTLAND, A. 1996. Modeling, tracking and interactive animation of faces and heads: Using input from video. In *Computer Animation '96*, 68–79.

EZZAT, T., AND POGGIO, T. 2000. Visual speech synthesis by morphing visemes. *International Journal of Computer Vision 38*, 1, 45–57.

FREEMAN, W. T., AND TENENBAUM, J. B. 1997. Learning bilinear models for two factor problems in vision. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 554–560.

GEORGHIADES, A., BELHUMEUR, P., AND KRIEGMAN, D. 2001. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) 23*, 6, 643–660.

GOTSMAN, C., GU, X., AND SHEFFER, A. 2003. Fundamentals of spherical parameterization for 3D meshes. *ACM Transactions on Graphics 22*, 3 (July), 358–363.

JONES, T. R., DURAND, F., AND DESBRUN, M. 2003. Non-iterative, feature-preserving mesh smoothing. *ACM Transactions on Graphics 22*, 3 (July), 943–949.

KOCH, R. M., GROSS, M. H., CARLS, F. R., VON BÜREN, D. F., FANKHAUSER, G., AND PARISH, Y. 1996. Simulating facial surgery using finite element methods. In *Proceedings of SIGGRAPH 96*, Computer Graphics Proceedings, Annual Conference Series, 421–428.

KRAEVOY, V., AND SHEFFER, A. 2004. Cross-parameterization and compatible remeshing of 3D models. *ACM Transactions on Graphics 23*, 3 (Aug.), 861–869.

KROONENBERG, P. M., AND DE LEEUW, J. 1980. Principal component analysis of three-mode data by means of alternating least squares algorithms. *Psychometrika 45*, 69–97.

LEE, Y., TERZOPOULOS, D., AND WATERS, K. 1995. Realistic modeling for facial animation. In *Proceedings of SIGGRAPH 95*, Computer Graphics Proceedings, Annual Conference Series, 55–62.

LI, H., ROIVAINEN, P., AND FORCHHEIMER, R. 1993. 3-D motion estimation in model-based facial image coding. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) 15*, 6, 545–555.

NOH, J.-Y., AND NEUMANN, U. 2001. Expression cloning. In *Proceedings of SIGGRAPH 2001*, Computer Graphics Proceedings, Annual Conference Series, 277–288.

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

PARKE, F. I. 1982. Parameterized models for facial animation. *IEEE Computer Graphics & Applications 2* (Nov.), 61–68.

PENTLAND, A., AND SCLAROFF, S. 1991. Closed-form solutions for physically based shape modeling and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) 13*, 7, 715–729.

PÉREZ, P., GANGNET, M., AND BLAKE, A. 2003. Poisson image editing. *ACM Transactions on Graphics 22*, 3 (July), 313–318.

PIGHIN, F., HECKER, J., LISCHINSKI, D., SZELISKI, R., AND SALESIN, D. H. 1998. Synthesizing realistic facial expressions from photographs. In *Proceedings of SIGGRAPH 98*, Computer Graphics Proceedings, Annual Conference Series, 75–84.

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

PRAUN, E., AND HOPPE, H. 2003. Spherical parameterization and remeshing. *ACM Transactions on Graphics 22*, 3 (July), 340–349.

ROBERTSON, B. 2004. Locomotion. *Computer Graphics World* (Dec.).

ROWEIS, S. 1997. EM algorithms for PCA and SPCA. In *Advances in neural information processing systems 10 (NIPS)*, 626–632.

SIROVICH, L., AND KIRBY, M. 1987. Low dimensional procedure for the characterization of human faces. *Journal of the Optical Society of America A 4*, 519–524.

SUMNER, R. W., AND POPOVIĆ, J. 2004. Deformation transfer for triangle meshes. *ACM Transactions on Graphics 23*, 3 (Aug.), 399–405.

TIPPING, M. E., AND BISHOP, C. M. 1999. Probabilistic principal component analysis. *Journal of the Royal Statistical Society, Series B 61*, 3, 611–622.

TORRESANI, L., YANG, D., ALEXANDER, E., AND BREGLER, C. 2001. Tracking and modeling non-rigid objects with rank constraints. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, 493–450.

TUCKER, L. R. 1966. Some mathematical notes on three-mode factor analysis. *Psychometrika 31*, 3 (Sept.), 279–311.

VASILESCU, M. A. O., AND TERZOPOULOS, D. 2002. Multilinear analysis of image ensembles: Tensorfaces. In *European Conference on Computer Vision (ECCV)*, 447–460.

VASILESCU, M. A. O., AND TERZOPOULOS, D. 2004. Tensortextures: multilinear image-based rendering. *ACM Transactions on Graphics 23*, 3 (Aug.), 336–342.

VIOLA, P., AND JONES, M. 2001. Rapid object detection using a boosted cascade of simple features. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, 511–518.

WANG, Y., HUANG, X., LEE, C.-S., ZHANG, S., LI, Z., SAMARAS, D., METAXAS, D., ELGAMMAL, A., AND HUANG, P. 2004. High resolution acquisition, learning and transfer of dynamic 3-d facial expressions. *Computer Graphics Forum 23*, 3 (Sept.), 677–686.

WATERS, K. 1987. A muscle model for animating three-dimensional facial expression. In *Computer Graphics (Proceedings of SIGGRAPH 87)*, vol. 21, 17–24.

WILLIAMS, L. 1990. Performance-driven facial animation. In *Computer Graphics (Proceedings of SIGGRAPH 90)*, vol. 24, 235–242.

ZHANG, L., SNAVELY, N., CURLESS, B., AND SEITZ, S. M. 2004. Spacetime faces: high resolution capture for modeling and animation. *ACM Transactions on Graphics 23*, 3 (Aug.), 548–558.