Facial Retargeting with Automatic Range of Motion Alignment

ROGER BLANCO i RIBERA*, KAIST
EDUARD ZELL*, Bielefeld University
J.P. LEWIS, Frostbite Labs and Victoria University
JUNYONG NOH†, KAIST
MARIO BOTSCH†, Bielefeld University

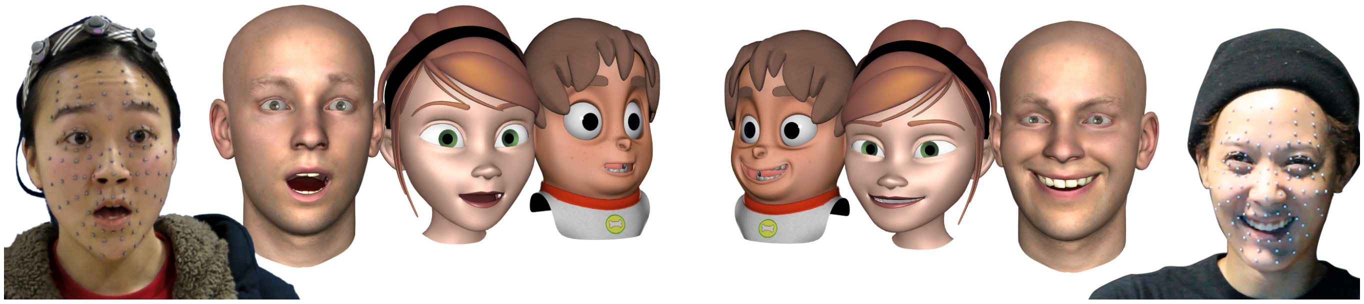

Fig. 1. Captured facial expression of two actors (*left* and *right*) with retargeting results to realistic and stylized characters. Our method automatically aligns the ranges of motion of the captured actor and the target blendshape rig, such that expressions are restored faithfully even for stylized characters.
From left to right: *The face rigs* Loki, Mery, *and* Billy, *are courtesy of* Mark Pauly, meryproject.com, *and* Jana Bergevin, *respectively. Motion capture data (*right*) courtesy of* Feel Ghood Music.

While facial capturing focuses on accurate reconstruction of an actor's performance, facial animation retargeting has the goal to transfer the animation to another character, such that the semantic meaning of the animation remains. Because of the popularity of blendshape animation, this effectively means to compute suitable blendshape weights for the given target character. Current methods either require manually created examples of matching expressions of actor and target character, or are limited to characters with similar facial proportions (i.e., realistic models). In contrast, our approach can automatically retarget facial animations from a real actor to stylized characters. We formulate the problem of transferring the blendshapes of a facial rig to an actor as a special case of manifold alignment, by exploring the similarities of the motion spaces defined by the blendshapes and by an expressive training sequence of the actor. In addition, we incorporate a simple, yet elegant facial prior based on discrete differential properties to guarantee smooth mesh deformation. Our method requires only sparse correspondences between characters and is thus suitable for retargeting marker-less and marker-based motion capture as well as animation transfer between virtual characters.

CCS Concepts: • **Computing methodologies → Animation**; *Motion capture*; *Motion processing*;

Additional Key Words and Phrases: facial animation, retargeting, blendshapes

ACM Reference format:
Roger Blanco i Ribera, Eduard Zell, J.P. Lewis, Junyong Noh, and Mario Botsch. 2017. Facial Retargeting with Automatic Range of Motion Alignment. *ACM Trans. Graph.* 36, 4, Article 154 (July 2017), 12 pages.
DOI: http://dx.doi.org/10.1145/3072959.3073674

1 INTRODUCTION

Facial animation retargeting addresses the general problem of animation transfer between virtual characters, with the transfer of performance capture to virtual characters being the main application. Recent developments in vision- and depth-sensor-based facial motion capture [Cao et al. 2014; Ichim et al. 2015; Li et al. 2013; Thies et al. 2016; Weise et al. 2011] made accurate captures of an actor, traditionally limited to big film or game studios, affordable to a much broader audience. Current real-time capture systems typically adapt a realistic generic blendshape model to the actor. Since the modified and the original character have semantically equivalent blendshapes, the captured actor performance is then transferred between the characters by directly mapping the blendshape weights. The special case of equivalent blendshapes between two characters is often named *parallel parametrization* in retargeting context.

In practice, it is uncommon to encounter facial rigs with a complete set of semantically equivalent blendshapes. Creating facial rigs for animation is time consuming and requires highly skilled artists. Therefore, a rig is carefully designed to fit the animation needs, only modeling the necessary expressions. In addition, expressive digital characters are often stylized and exaggerate the facial proportions of humans. An effective retargeting method must either transfer animation from facial motion capture markers to a blendshape rig or between faces with different blendshape sets. Several

*authors contributed equally; †corresponding author(s).

retargeting approaches generate their own parallel parametrization, by transferring the blendshapes of the character face rig to align with the actor's proportions. However, especially for stylized characters this step often fails, due to differences in ranges of motion or the shortcomings of current methods. The subsequent blendshape estimation becomes erroneous, which has been addressed so far by incorporating additional priors.

In this paper we propose a novel algorithm for creating actor-specific blendshapes with the help of a training sequence consisting of an actor's facial motions that semantically correspond to the blendshapes of the character face rig. We show that given a training sequence that sufficiently covers the actor's range of motion, it is possible to create, in an unsupervised manner, an accurate parallel parametrization – even if the facial rig and the actor differ strongly in their facial proportions. The key observation is that facial motions are similar across different stylization levels, as motivated by the facial action coding system (FACS) [Ekman and Friesen 1978]. The FACS describes facial expressions on the basis of muscle activations and is a common reference for blendshape creation for stylized and realistic human characters. Based on a new manifold alignment approach and a novel energy measuring similarity of facial expressions, we successfully align the ranges of motion of the actor and the character face rig. This subsequently leads to accurate retargeting (see Figure 1).

Our second contribution is a prior energy based on physically-inspired deformations, which can be computed efficiently even in real-time applications. Our geometric prior addresses the few artifacts that remain even in case of accurate parallel parametrizations. Both contributions are fully compatible with most previous methods, suitable for real-time applications, and produce results comparable or better than state-of-the-art offline methods [Seol et al. 2012] (Figure 15).

2 RELATED WORK

As a key element of human-centered applications, research on virtual faces and face animation has been an active field of research for decades, resulting in a wide range of publications on this topic. For a general overview we recommend the book of Parke and Waters [2008] and the more recent surveys focusing on rigging [Orvalho et al. 2012] and blendshape animation [Lewis et al. 2014]. In the following, we focus mainly on facial retargeting and assume a certain familiarity with blendshape-based facial animation (Section 3).

Cross-Mapping. To overcome problems at the transfer stage, cross-mapping methods learn directly from semantically corresponding facial expressions of the captured actor and a target face rig and synthesize new poses based on these training examples. Different learning techniques have been proposed, starting with piece-wise linear mapping [Buck et al. 2000] and locally linear embedding [Wang et al. 2004], followed by more advanced machine learning algorithms like RBFs [Deng et al. 2006], kCCA [Song et al. 2011], simplicial basis [Kholgade et al. 2011], or shared Gaussian Process Latent Variable Models (sGPLVM) [Bouaziz and Pauly 2014]. A key advantage of all cross-mapping approaches is that they are applicable to any type of character (e.g., even having different number of eyes) or any facial rig (blendshapes, muscles, etc.). Unfortunately, the performance of these methods is strongly tied to the quality and number of given training examples. Often, at least 15–20 corresponding example pairs are required for sufficient results. For a moderate facial rig with 40 blendshapes, this leads to 600–800 parameters which must be defined consistently by hand. However, even in case of consistent training examples, the resulting expressions still remain (sophisticated) interpolations of the training examples. This often leads to inaccurate results for expressions that are too different from the training examples.

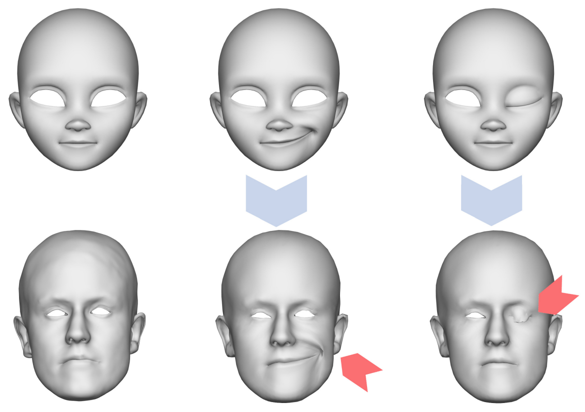
Fig. 2. Transferring blendshapes using deformation transfer leads to unnatural deformations (*center*) or broken expressions (*right*) if facial proportions are too different. Neutral expressions are shown on the *left*.
©Face rig: meryproject.com

Parallel Parametrization. The simplest way to transfer an animation from one character to another is by creating two semantically equivalent facial rigs. In this case, the animation can simply be transferred by copying the control parameters from one rig to another. For blendshape-based facial rigs, manually creating semantically corresponding sets of blendshapes is a labor-intensive task, requiring not only excellent modeling skills and anatomical knowledge of the face, but also a considerable amount of time. In order to automate this process, several approaches for transferring blendshapes from a generic face model to a neutral target have been suggested.

Given a source blendshape rig and the neutral face of a target character, Noh and Neumann [2001] suggested first to establish dense correspondences and then to transfer per-vertex displacements for each expression. This was later improved using deformation gradients [Sumner and Popović 2004] or Radial Basis Functions [Orvalho et al. 2008; Seol et al. 2012, 2011]. Several improvements have been suggested since then, ranging from incorporating examples [Li et al. 2010], adding contact constraints [Saito 2013], interactive editing [Xu et al. 2014], to iterative refinement schemes for real humans [Bouaziz et al. 2013; Ichim et al. 2015; Seol et al. 2016]. However, if the assumption that source and target models are of similar shape is violated, deformation transfer and similar methods often fail to preserve the semantics of the facial expressions (see Figure 2). The resulting proportional mismatch then leads to artifacts at the retargeting stage: exaggerated actor blendshapes cause dampened

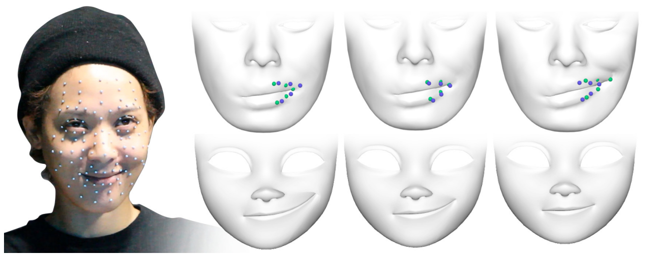
Fig. 3. Retargeting a smile (*left*) to differently personalized blendshapes (*top*). Close-ups show the positions of the captured markers (*blue*) and the corresponding vertices (*green*). Only well-matching blendshapes result in accurate retargeting (*center*). Dampened personalized blendshapes (*top left*) cause over-exaggerated retargeting results (*bottom left*), since large weights are necessary to fit the captured markers. Inversely, exaggerated blendshapes (*top right*) cause damped retargeting results (*bottom right*).
©Motion capture: Feel Ghood Music, ©Face rig: meryproject.com

animations, because smaller weights are sufficient to reach a target pose; conversely, dampened blendshapes cause larger weights and exaggerated animations – up to the point of unnatural face deformations (Figure 3).

Seol et al. [2012] address artifacts resulting from erroneous expression transfer by integrating velocities over a sequence of captured frames. In contrast, we improve the transfer process, such that the transferred blendshapes automatically adapt to the actor's range of motion. Given a motion sequence of an actor and sparse correspondences between the actor and the character model (e.g., in form of optical markers), our method automatically transfers the blendshapes of the face rig to the actor space.

Manifold-based Techniques. Aligning the ranges of motion between the actor's motion sequence and the character's blendshape rig significantly improves our expression transfer and is inspired by the success of manifold alignment methods [Pan and Yang 2010]. These approaches aim at registering two different high-dimensional data sets in a lower-dimensional embedding space. The mapping into the lower-dimensional space has to minimize the distance between the individual manifolds as well as to keep the original relationship between the data elements by preserving the geometric structure of the manifolds. Several unsupervised methods [Fan et al. 2016; Wang and Mahadevan 2009] have been proposed for various applications, including transfer learning [Pan and Yang 2010], data mining, automatic translation or image set matching [Cui et al. 2012; Pei et al. 2012]. An important aspect is the dimensionality reduction, where additional constraints ensure optimal embedding spaces. Often, transformations between embedding spaces are then solved by eigen-decomposition of the graph Laplacian [Fan et al. 2016; Wang and Mahadevan 2011, 2013]. In some sense, manifold alignment techniques aim to find a low-dimensional space where Euclidean distances better represent the similarity between the different data instances. In contrast, we want to identify character blendshapes that match the proportions and ranges of motion of the actor in the high-dimensional space. This requires the transfer of the original blendshapes to the actor space, instead of projecting into a low-dimensional space.

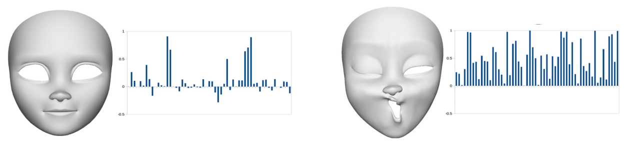
Fig. 4. Complex interaction of blendshape weights. *Left:* Weight cancellation effects lead to a valid neutral face. *Right:* Constraining all weights to the interval [0, 1] does still not guarantee valid face expressions.
©Face rig: meryproject.com

Expression Regularization. Common approaches for reducing artifacts in blendshape animation is to restrict the blendshape weights to a fixed interval [Bregler et al. 2002; Chuang and Bregler 2002] or to penalize large weights [Seo et al. 2011]. However, such heuristics do not always succeed because combinations of blendshape weights outside the specified range can still produce valid faces [Seol et al. 2011], and restricting blendshape weights to [0, 1] will not necessarily result in plausible expressions (Figure 4). This phenomenon is commonly known as *blendshape interference* [Lewis et al. 2005]. Alternatively, PCA-based priors have been proposed for direct blendshape manipulation [Anjyo et al. 2012; Lau et al. 2009] and retargeting [Seol et al. 2012]. But these approaches strongly depend on the quality and amount of training examples, where an insufficient set of example poses biases the solution towards the closest training data [Anjyo et al. 2012]. In contrast, we consider smooth skin deformation as a key factor and propose a prior that penalizes surface deformation similar to many physically-inspired facial animation methods [Barrielle et al. 2016; Bickel et al. 2007; Ichim et al. 2016]. While all these methods outperform linear blendshapes in physical accuracy, the visual improvements often do not justify the additional computation costs for many applications. We therefore reformulate the large-scale deformation energy of Bickel et al. [2007] to a suitable geometric prior for blendshape weights. Furthermore, in order to additionally achieve sparse weight activation, one can either regularize using the L_1 norm of blendshape weights [Bouaziz et al. 2013] or transfer common practices in manual key-framing [Seol et al. 2011].

3 BLENDSHAPE ANIMATION AND RETARGETING

In this section we briefly review blendshape facial animation and blendshape-based facial retargeting and set up our notation. Section 4 then introduces our improved blendshape transfer, which is a pre-processing step before the actual retargeting. The retargeting itself can be further regularized using our geometric prior (Section 5). Finally, we compare our proposed approach to state-of-the-art methods in Section 6.

Let the facial rig be given as a polygon mesh $\mathcal{M}$, consisting of N vertices, posed in neutral expression, and being equipped with K expression blendshapes that all share the connectivity of $\mathcal{M}$. We denote the vector of stacked vertex positions of the neutral face by $\mathbf{v}_0 = (\mathbf{v}_0^1, \ldots, \mathbf{v}_0^N)^\mathsf{T}$, and of the k-th blendshape by $\mathbf{v}_k = (\mathbf{v}_k^1, \ldots, \mathbf{v}_k^N)^\mathsf{T}$. Due to the coupling of the x/y/z-coordinates the blendshapes $\mathbf{v}_k$ denote $3N$-dimensional vectors.

For blendshape face animation we employ the *delta-blendshape* formulation [Lewis et al. 2014], where the neutral expression $\mathbf{v}_0$ is

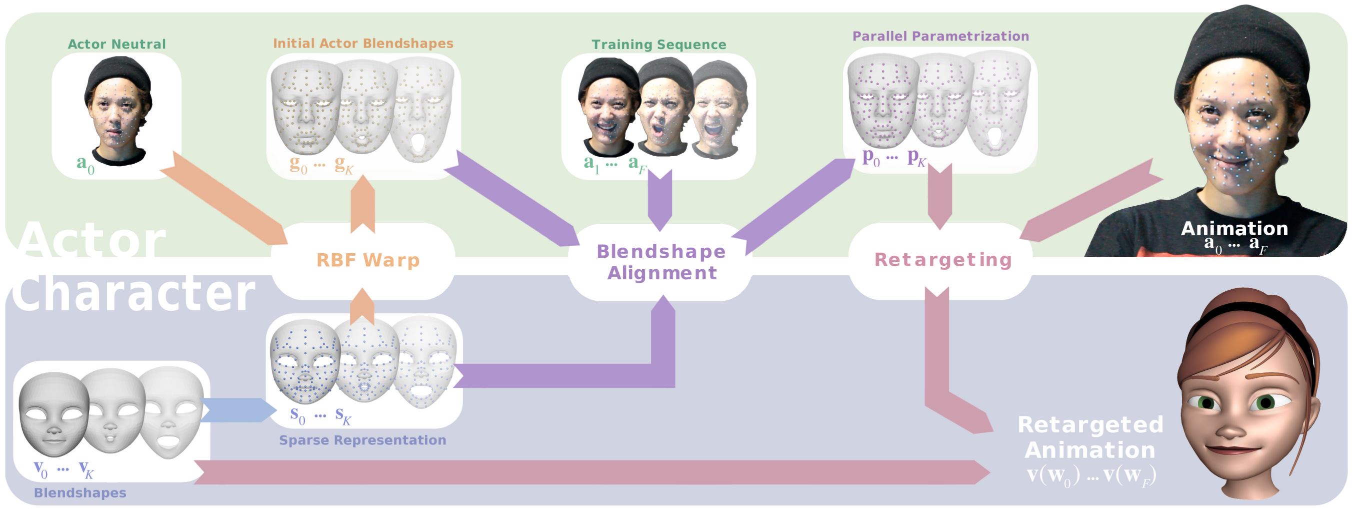

Fig. 5. Overview of facial animation retargeting pipeline, described for illustration purpose on the use case of retargeting marker-based animation. Our method addresses the problem of obtaining personalized blendshapes by aligning the range of motion of the blendshape model with the range of motion of the actor. In addition, we present a new prior for the retargeting step. ©Motion capture: Feel Ghood Music ©Face rig: meryproject.com

subtracted from the blendshape expressions $\mathbf{v}_k$ to yield a displacement field for activating a particular expression: $\delta\mathbf{v}_k = \mathbf{v}_k - \mathbf{v}_0$. New facial poses $\mathbf{v}(\mathbf{w})$ are computed by displacing the neutral face by a weighted sum of delta-blendshapes, with weights $\mathbf{w} = (w_1, \dots, w_K)^\mathsf{T}$, which can also be written in matrix form using the delta-blendshape matrix $\delta\mathbf{V} = (\delta\mathbf{v}_1, \dots, \delta\mathbf{v}_K)$:

$$\mathbf{v}(\mathbf{w}) \;=\; \mathbf{v}_0 + \sum_{k=1}^{K} w_k \delta\mathbf{v}_k = \mathbf{v}_0 + \delta\mathbf{V}\,\mathbf{w}\,. \tag{1}$$

The main application for facial retargeting is the transfer of an actor's performance capture to a virtual character, mostly using marker-based optical motion capture. We will therefore formulate our approach for this problem setting, but note that our method is not limited to marker-based retargeting, since any given facial animation or marker-less performance capture can be easily converted to a marker-based performance capture by tracing a subset of "marker vertices" through time.

The actor's performance is given as a $3M$-dimensional vector of M stacked marker positions $(\mathbf{a}^1, \dots, \mathbf{a}^M)^\mathsf{T}$ that vary over time. For a particular motion capture frame f, this data is denoted as $\mathbf{a}_f = (\mathbf{a}_f^1, \dots, \mathbf{a}_f^M)^\mathsf{T}$, and $\mathbf{a}_0$ represents a calibration frame of the actor in neutral expression. Like all retargeting methods based on parallel parameterization, we require sparse correspondences between the actor's face animation and the character's face rig. These correspondences are specified as pairs of points $\{\mathbf{a}_0^m, \mathbf{s}_0^m\}$, $m = 1, \dots, M$, on the neutral expressions of the actor and the character rig. The same set of vertices on the expression blendshape $\mathbf{v}_k$ is denoted by $\mathbf{s}_k = (\mathbf{s}_k^1, \dots, \mathbf{s}_k^M)^\mathsf{T}$. Since the number M of markers and corresponding vertices is much lower than the number of mesh vertices ($M \ll N$), the $\mathbf{s}_k$ are called the *sparse representation* of the blendshape $\mathbf{v}_k$. We employ the same delta-formulation as above for sparse blendshapes ($\delta\mathbf{s}_k = \mathbf{s}_k - \mathbf{s}_0$) and animation data ($\delta\mathbf{a}_f = \mathbf{a}_f - \mathbf{a}_0$).

The goal of any blendshape retargeting system is to compute the time-varying weights $\mathbf{w}$ that reproduce the facial expressions on the target face rig for a given actor's performance capture. This requires a set of *personalized sparse actor blendshapes* $\mathbf{p}_k = (\mathbf{p}_k^1, \dots, \mathbf{p}_k^M)^\mathsf{T}$ that are semantically equivalent to the sparse blendshapes $\mathbf{s}_k$ of the character rig. For each captured frame f, the blendshape weights $\mathbf{w}_f$ can be computed by minimizing the squared distance between the marker displacements $\delta\mathbf{a}_f$ and a weighted combination of the actor's sparse delta-blendshapes $\delta\mathbf{p}_k = \mathbf{p}_k - \mathbf{p}_0$:

$$E_{\text{Fit}}(\mathbf{w}) \;=\; \frac{1}{M} \left\| \delta\mathbf{a}_f - \sum_{k=1}^{K} w_k\, \delta\mathbf{p}_k \right\|^2 . \tag{2}$$

The required personalized actor blendshapes $\mathbf{p}_k$ are either manually created or transferred from the face rig to actor space [Orvalho et al. 2008; Seol et al. 2012; Sumner and Popović 2004]. But as discussed above and shown in Figures 2 and 3, this blendshape transfer often fails for highly different facial proportions, such as stylized characters. We therefore propose an improved approach for automatic blendshape transfer with range of motion adjustment in Section 4.

In order to resolve ambiguities, prevent over-fitting, or penalize artifacts, the above blendshape fitting process is typically regularized through additional energy terms:

$$E_{\text{Retarget}}(\mathbf{w}) \;=\; E_{\text{Fit}}(\mathbf{w}) + E_{\text{Reg}}(\mathbf{w})\,. \tag{3}$$

Typical choices for the energy $E_{\text{Reg}}(\mathbf{w})$ are (weighted combinations of) L_2 regularization $\|\mathbf{w}\|^2$ to penalize large weights [Lewis et al. 2014], L_1 regularization $\|\mathbf{w}\|_1$ for inducing sparsity [Bouaziz et al. 2013], and penalization of temporal changes $\left\|\mathbf{w}_{f-1} - \mathbf{w}_f\right\|^2$ to remove jitter [Lewis et al. 2014]. However, as we will show in Section 5, our novel geometric prior, which operates on differential mesh properties instead of on blendshape weights, prevents geometric artifacts while at the same time allowing for more expressive animation.

4 AUTOMATIC BLENDSHAPE TRANSFER

A correct set of actor-specific blendshapes $\mathbf{p}_1, \ldots, \mathbf{p}_K$, used in the fitting term (2), is a crucial component of any retargeting method. Existing approaches for transferring the character's blendshapes to the actor space compensate for proportional mismatches between character rig and actor to a certain degree. Unfortunately, they often fail to properly align the respective ranges of motion, in particular for stylized characters (see Figures 2, 3). Our approach addresses these shortcomings and is motivated by two main observations:

- Blendshapes typically define the strongest deformation caused by activating isolated facial muscles.
- Semantically equivalent expressions are highly similar for different characters, because facial muscles are consistent across humans and remain consistent even for highly stylized characters for the sake of easy expression recognition. Although semantically equivalent expressions are similar across identities, they are not equal and differ with respect to direction and magnitude.

We improve the blendshape transfer from the facial rig to the actor's proportions by aligning their facial expression manifolds. We learn the actor's expression manifold from a short, captured training sequence, consisting of F animation frames $\mathbf{a}_1, \ldots, \mathbf{a}_F$. The actor's training sequence should contain semantically equivalent expressions to the blendshapes of the facial rig; it should cover the actor's range of motion since blendshapes often correspond to extreme expressions. However, the training expressions might combine several blendshapes in arbitrary order. As introduced above, the animation data $\mathbf{a}_f$ consist of M markers with point correspondences on the facial rig. The character's sparse blendshapes $\mathbf{s}_1, \ldots, \mathbf{s}_K$ define well the expression manifold of the character rig.

Given the blendshape rig and a set of training expressions, our method computes personalized actor blendshapes based on concepts from manifold alignment (E_{Match}, Section 4.3). To this end, we measure the similarity between the character's blendshapes and actor's captured performance (Section 4.1) and extract the most important frames from the training data (Section 4.2). To regularize the alignment process, the actor-specific blendshapes should not deviate too much from an initial guess derived by RBF deformation (E_{Mesh}, Section 4.4), and relations between individual blendshapes should be preserved (E_{CEG}, Section 4.5). Figure 5 illustrates schematically our blendshape transfer in the context of facial retargeting.

Our blendshape transfer process is formulated as the minimization of the following energy (with $\alpha = 0.01$ and $\beta = 0.1$ in all shown examples) and optimizes all personalized delta-blendshapes at the same time, where the $3MK$-dimensional vector $\delta\mathbf{p} = (\delta\mathbf{p}_1, \ldots, \delta\mathbf{p}_K)^T$ contains the stacked blendshapes:

$$E_{\text{Align}}(\delta\mathbf{p}) \;=\; E_{\text{Match}}(\delta\mathbf{p}) + \alpha E_{\text{Mesh}}(\delta\mathbf{p}) + \beta E_{\text{CEG}}(\delta\mathbf{p})\,. \tag{4}$$

4.1 Facial Motion Similarity

The similarity of the ranges of motion is best visualized by considering the displacements of a single marker ($\delta\mathbf{a}_f^m = \mathbf{a}_f^m - \mathbf{a}_0^m$) over the entire training sequence and the delta-blendshapes of the corresponding vertex on the facial rig ($\delta\mathbf{s}_k^m = \mathbf{s}_k^m - \mathbf{s}_0^m$), as shown in Figure 6. In some cases clear clusters of marker displacements can be identified, indicating that different motions on that particular area occur mainly independently. For example, on the right brow (Figure 6, eyebrow) three clearly distinctive motions can be identified, corresponding to raising and lowering the brow and frowning. Considering that the captured training sequence consists of over 2000 frames, the clear separation into these motion clusters is surprising. This observation fuels our motivation to consider an expressive training sequence to generate an actor-specific parallel parametrization. Still, a clear solution may not always exist (Figure 6, lips). Even though there might be a lot of data available, it is possible that no clear one-to-one correspondence between the performance and specific blendshapes can be established. When a clear correspondence can be identified, it is best to exactly match the specific expression. However, aligning blendshapes with unrelated expressions will result in a complete loss of the semantic equivalence between the actor's performance and the retargeted animation.

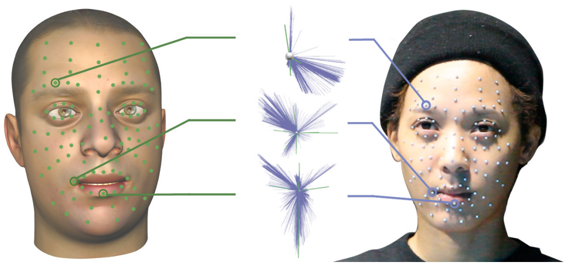
Fig. 6. Motion space comparison between an actor *(right)* and a blendshape model *(left)*. Center: Overlayed motion space in delta representation for selected markers of the actor (blue) and corresponding vertices in the blendshape model (green). Each line corresponds to a different frame/blendshape. Face rig courtesy of *Jason Osipa*. ©Motion capture: Feel Ghood Music

In order to quantify the similarities between a blendshape k and the actor's expression at frame f, we compute the Pearson Correlation Coefficient between $\mathbf{a}_f$ and $\mathbf{s}_k$. In our case the mean of a sampling set is replaced by the more meaningful neutral facial expression. The computation of the correlation coefficient $c_{k,f}$ between an actor's expression $\mathbf{a}_f$ and a sparse blendshape $\mathbf{s}_k$ then simplifies to the following equation in delta-representation:

$$c_{k,f} = \frac{\delta\mathbf{a}_f \cdot \delta\mathbf{s}_k}{\left\|\delta\mathbf{a}_f\right\| \left\|\delta\mathbf{s}_k\right\|}\,. \tag{5}$$

Figure 7 shows the resulting similarity measures between selected blendshapes and a training sequence. This simple dot-product formulation of Equation (5) is effective, because:

- Displacements of blendshapes and actor's expressions in similar directions have high correlation,
- Locality of blendshapes is considered, because vertices that do not move in blendshape $\delta\mathbf{s}_k$ cancel out the contributions of the corresponding expression $\delta\mathbf{a}_f$.

Contrast Enhancement. The similarity measure can be further improved by two heuristics: Identifying important blendshapes for a frame in the training sequence (i) is easier for more unique blendshapes and (ii) is easier for blendshapes with strongest displacement. Both properties are considered by computing a trust value for

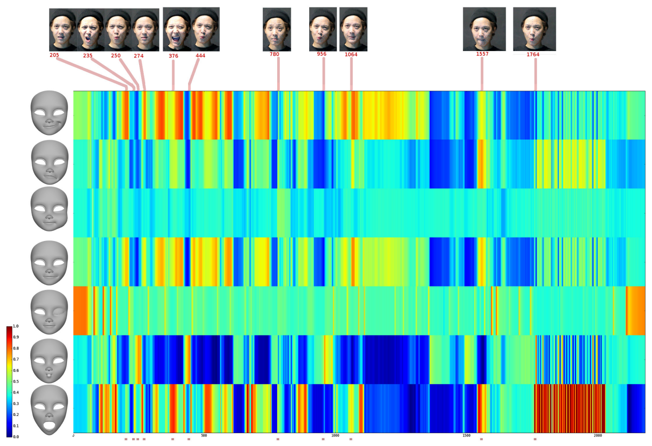

Fig. 7. Motion space similarity between selected blendshapes (*rows*) and actor's performance (*columns*) consisting of 2150 frames. Please notice the blocking structure, indicating redundancy of information.
©Motion capture: Feel Ghood Music ©Face rig: meryproject.com

each sparse blendshape. We first compute the total displacement $d_k = \|\delta \mathbf{s}_k\|$ of blendshape $\delta \mathbf{s}_k$. In a second step the sparse blendshapes are re-ordered according to their displacements d_k, such that $\delta \mathbf{s}_1$ is the blendshape with the largest displacement and $\delta \mathbf{s}_K$ the one with the smallest displacement. We then build a between-blendshape correlation matrix $\mathbf{C}(k,l) := c_{k,l}$ (see Figure 11). Finally, the trust value t_k is computed as

$$t_k \;=\; 1 - \frac{\sum_{l=1}^{k-1} c_{k,l}^{+}}{\max_{1 \le k \le K}\left(\sum_{l=1}^{k-1} c_{k,l}^{+}\right)}\,, \tag{6}$$

where similarity between sparse blendshapes is measured using the *positive* Pearson Correlation Coefficient $c_{k,l}^{+} = \max(0, c_{k,l})$. The sum $\sum_{l=1}^{k-1} c_{k,l}^{+}$ adds all $c_{k,l}^{+}$ within row k of the strictly lower triangle matrix, and the dominator is the maximum of all row sums. The sparse blendshape with the largest displacement has no entries in the strictly lower triangle matrix, such that $t_1 = 1$. All remaining sparse blendshapes will only have $t_k \approx 1$ if they are uncorrelated to sparse blendshapes with larger displacements. Hence, a blendshape modeling a subtle lip motion will likely get a very low trust value due to high correlations with other, more expressive mouth blendshapes, while a brow-raising blendshape will have a high trust value as it is mostly uncorrelated to other blendshapes.

In practice, most computed similarity values, except the blendshape with maximum displacement, will never reach a trust value of one, because even semantically equal expressions differ for different faces. We therefore propose to amplify high correlation values and reduce low correlation values using the following transformation:

$$b\left(c_{k,f}\right) \;=\; \frac{\mathrm{e}^{r c_{k,f}^{+}}}{\mathrm{e}^{r/2} + \mathrm{e}^{r c_{k,f}^{+}}}\,. \tag{7}$$

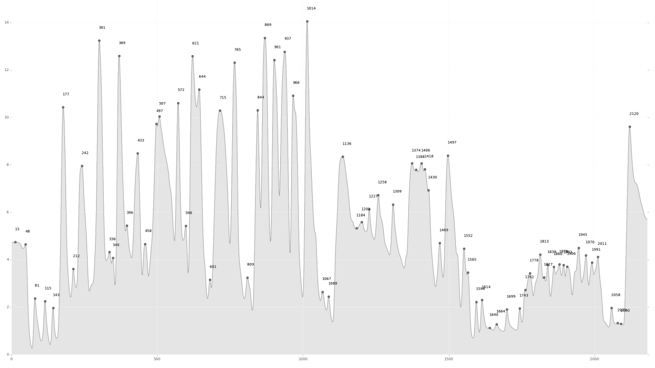

Fig. 8. Cumulative correlation function (not displacements) for identifying peak expressions within a training sequence. 82 expressions have been identified from the similarities shown in Figure 7.

The steepness r of the function is set to 15 in all our examples. Finally, we linearly interpolate between the original and the modified correlation values, depending on the trust value. The computed similarity $\tilde{c}_{k,f}$ replaces then the original correlation from Equation (5):

$$\tilde{c}_{k,f} \;=\; (1 - t_k)\, c_{k,f}^{+} + t_k\, b\left(c_{k,f}^{+}\right)\,. \tag{8}$$

4.2 Key Expression Extraction

Blendshapes represent peak expressions that we want to match to the actor's most similar expressions. After computing the similarities $\tilde{c}_{k,l}$ between blendshapes and the training sequence, we remove the temporal redundancy between consecutive frames following Coleman et al. [2008]. The correlations of each blendshape over the whole training sequence (corresponding to a row in Figure 7) are first low-pass filtered to remove some superfluous noise. The employed Gaussian kernel is kept small (three frames wide) to avoid over-smoothing fast and peak motions. Finally, all filtered rows are added together by summing over columns. In this cumulative representation of the data, we extract all local peaks in order to obtain a set of most similar facial expressions. Figure 8 shows the resulting cumulative function and the extracted peaks.

4.3 Manifold Alignment

Given a similarity measure $\tilde{c}_{k,f}$ between a sparse blendshape and an actors's expression, our goal is to fit the personalized blendshapes $\delta \mathbf{p}_k$ to the actor's expressions $\delta \mathbf{a}_f$. This means that a closed-eye blendshape should be fitted to an actor's expression with closed eye(s) in the training sequence, without being influenced by the negatively correlated eye-opening movements. Inspired by manifold alignment techniques we first consider the problem as the minimization of the following energy:

$$E_{\text{Match}}(\delta \mathbf{p}) \;=\; \frac{1}{F} \sum_{f=1}^{F} \sum_{k=1}^{K} \tilde{c}_{k,f} \left\| \delta \mathbf{p}_k - \delta \mathbf{a}_f \right\|^2\,. \tag{9}$$

However, this energy formulation would break the local support of blendshapes. Figure 9 shows an example of closing the eyes. In general eyes blink simultaneously, leading to high correlations of

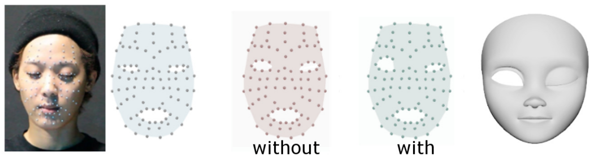

Fig. 9. Blendshapes local support. From *left* to *right*, a closed-eye frame and corresponding marker positions (*blue*), fitting results of the eye-closed blendshape using E_{Match} *without* local support (*red*) and *with* local support considered (*green*), and the original blendshape for reference.
©Motion capture: Feel Ghood Music ©Face rig: meryproject.com

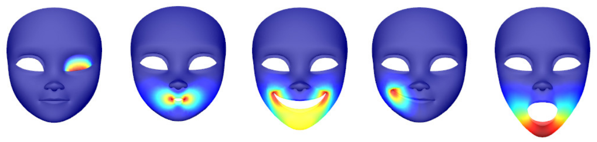

Fig. 10. Local deformations of blendshapes. Red encodes the strongest displacement within a blendshape. From *left* to *right*: left eye closed, kiss, open smile, right smile, and jaw drop. ©Face rig: meryproject.com

both eye-closing blendshapes (left and right). Without special consideration of the local support of each blendshape the alignment is distributed between both blendshapes. Including a mask of the blendshape displacements effectively disambiguates the displacement distribution (Figure 10). Similar to Seol et al. [2016] we encode local support with a soft mask vector $\mathbf{u}_k$ for blendshape k. The mask entries corresponding to the x/y/z coordinates of marker m of blendshape k are computed as $\left\|\delta \mathbf{s}_k^m\right\| / \max_{1 \le m \le M} \left\|\delta \mathbf{s}_k^m\right\|$, with $\max_{1 \le m \le M} \left\|\delta \mathbf{s}_k^m\right\|$ denoting the largest marker displacement within the blendshape $\delta \mathbf{s}_k$. Our final matching function is then

$$E_{\text{Match}}(\delta \mathbf{p}) \;=\; \frac{1}{F} \sum_{f=1}^{F} \sum_{k=1}^{K} \tilde{c}_{k,f} \left\| \delta \mathbf{p}_k - \operatorname{diag}(\mathbf{u}_k) \cdot \delta \mathbf{a}_f \right\|^2 . \qquad (10)$$

4.4 Geometric Constraint

Deformation transfer and similar automatic approaches in general do not create accurate parallel parametrizations. However, such methods preserve the original semantics and local properties of the transferred expression quite well. A transferred smile will remain a recognizable smile, although it might be too dampened or too exaggerated. Preserving the local features of an expression, e.g., the o-shape of the lips for the kiss expression, is the intention of the geometric constraint.

Based on the sparse correspondences between the neutral expressions $\mathbf{s}_0$ (of the character rig) and $\mathbf{a}_0$ (of the actor), we create an initial guess $\mathbf{g}_k$ for each personalized blendshape $\mathbf{p}_k$ using RBF-deformations [Orvalho et al. 2008; Seol et al. 2012]. To this end we first compute an RBF thin-plate spline that transforms the neutral expression $\mathbf{s}_0$ to $\mathbf{a}_0$, by placing an RBF center at every marker $\mathbf{s}_0^m$ and solving for the RBF weights. The resulting RBF function is then used to convert all delta-blendshapes $\delta \mathbf{s}_k$ to the initial guesses $\delta \mathbf{g}_k$ of the actor blendshapes. We also tried deformation transfer [Sumner and Popović 2004] for this process and can confirm the similarity of the

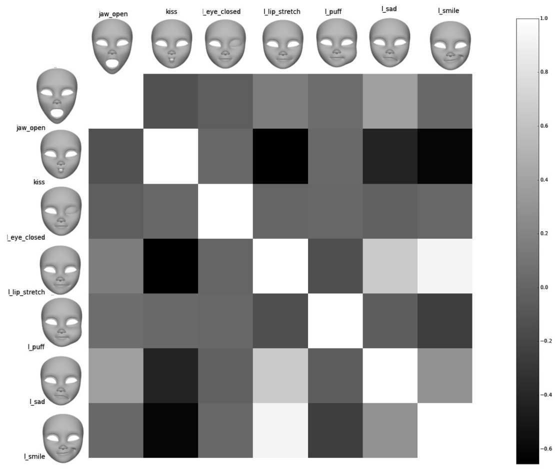

Fig. 11. Computed Pearson Correlation Coefficient between an exemplary blendshape set of the Mery character. Signed correlations are used in Section 4.5 to preserve both similarities and dissimilarities between blendshapes. Only positive correlations between animation frames and blendshapes are required for Section 4.3. ©Face rig: meryproject.com

results [Seol et al. 2012]. We preferred the RBF deformation method due to higher stability in case of degenerate meshes.

The goal of the geometric prior is to preserve the local shape properties of the initial guesses $\mathbf{g}_k$ while computing the personalized blendshapes $\mathbf{p}_k$. Local shape properties can be encoded well using per-vertex Laplacians [Botsch and Sorkine 2008], but this requires a triangle mesh, and so far the sparse blendshapes have been defined as sets of M marker points only. We therefore triangulate the marker points (in the uv-domain given by the parameterization), and add edges connecting upper/lower eyelids and lip markers to benefit from contact relationships [Saito 2013].

We formulate our geometric constraint as an energy that penalizes the change of the Laplacians between the unknown vertices of the personalized blendshape $\delta \mathbf{p}_k$ and the initial guesses $\delta \mathbf{g}_k$. This formulation is equivalent to a physically-inspired energy that minimizes bending [Bickel et al. 2007; Saito 2013]:

$$E_{\text{Mesh}}(\delta \mathbf{p}) \;=\; \frac{1}{M} \sum_{k=1}^{K} \sum_{m=1}^{M} \left\| \Delta \left(\delta \mathbf{p}_k^m - \delta \mathbf{g}_k^m \right) \right\|^2 . \qquad (11)$$

As a discretization of the Laplace operator we employ the standard cotangent weights [Pinkall and Polthier 1993].

4.5 Cross-Expression Constraint

The last energy term is responsible for maintaining the relationship between different blendshapes. If, for example, the mouth-open expression is corrected, this correction should also partly apply to the o-viseme. Inspired by application and relationship of graph and mesh Laplacians [Belkin and Niyogi 2005], we construct a *Cross-Expression Graph* that connects all blendshapes with each other. In this graph each sparse blendshape becomes a node with edges to all other sparse blendshapes and edge weights are encoded using

		blendshapes	key expressions sequence 1 (2150 frames)	key expressions sequence 2 (740 frames)	time (s)
Loki		46	73	46	18.71
Osipa		33	71	43	5.37
Mery		54	82	49	23.25
Billy		39	76	48	7.57

Table 1. Details about the sequences, extracted key expressions, and the blendshape rigs, together with timings for automatic blendshape transfer. Both sequences were captured with 99 optical markers.
©Face rigs: Mark Pauly, Jason Osipa, meryproject.com, Jana Bergevin

the *signed* similarity measure from Equation (5). We use signed correlations as edge weights (Figure 11) in order to preserve both similarities and dissimilarities between blendshapes. The weighted signed graph Laplacian [Kolluri et al. 2004; Kunegis et al. 2010] of one node is then defined as the weighted sum over its neighborhood:

$$\Delta(\delta \mathbf{s}_k) \;=\; \frac{1}{\sum_{l \neq k} |c_{k,l}|} \sum_{l \neq k} c_{k,l} \, (\delta \mathbf{s}_l - \delta \mathbf{s}_k) \, . \tag{12}$$

The graph Laplacian $\Delta(\delta \mathbf{p}_k)$ for the personalized blendshapes is defined equivalently. Similar to Equation (11), the resulting energy term penalizes dissimilarity of correlated blendshape displacements:

$$E_{\text{CEG}}(\delta \mathbf{p}) \;=\; \frac{1}{M} \sum_{k=1}^{K} \|\Delta \, (\delta \mathbf{p}_k - \delta \mathbf{s}_k)\|^2 \, . \tag{13}$$

4.6 Numerical Optimization

Optimizing for the personalized blendshapes $\delta \mathbf{p}_k$ requires minimizing E_{Align} from Equation (9) consisting of the three quadratic energies E_{Match}, E_{Mesh}, and E_{CEG}. The energies are all separable in the x/y/z coordinates of $\delta \mathbf{p}_k$, but are coupled between all blendshapes through E_{CEG}. This leads to three linear systems of size $(MK \times MK)$, which is more efficient to solve than one big linear system of size $(3MK \times 3MK)$. The entire framework was implemented within Maya [Autodesk 2016] using the Maya Python API, and all tests were performed on a computer with an Intel I7 3.4 GHz processor and 8 GB of memory. The linear systems were solved with the SciPy sparse solver [Jones et al. 2001]. Details and timings are listed in Table 1.

5 GEOMETRIC PRIOR

After computing the personalized blendshapes $\delta \mathbf{p}_k$, the blendshape weights $\mathbf{w}_f$ can be estimated for each frame $\delta \mathbf{a}_f$ of the actor's performance. Despite accurate personalized blendshapes, the flexibility provided by the facial rig might not be sufficient to faithfully reproduce the actor's expressions, leading to artifacts in the retargeted face $\mathbf{v}(\mathbf{w}_f)$. We observe that one essential property of a valid expression is a surface free of fold-overs, i.e., without strong local bending. Thus, instead of formulating an arbitrary criterion for the values of the blendshape weights (see Section 3), we derive a physically-inspired prior similar to Equation (11) that penalizes surface bending and thereby eliminates fold-overs [Bickel et al. 2007; Saito 2013]. We formulate the prior energy in terms of blendshapes and their weights, instead of in terms of vertex positions, so that our prior can be incorporated into any blendshape-based retargeting framework.

For faces, it seems natural to select the neutral pose $\mathbf{v}_0$ as the original, undeformed state. Thus our geometric prior penalizes bending between an expression $\mathbf{v}$ and the neutral face $\mathbf{v}_0$:

$$E_{\text{Prior}}(\mathbf{v}) \;=\; \frac{1}{N} \sum_{n=1}^{N} \left\|\Delta \left(\mathbf{v}^n - \mathbf{v}_0^n\right)\right\|^2 \, . \tag{14}$$

Based on Equation (1) the deformed face $\mathbf{v} = \mathbf{v}(\mathbf{w})$ can be written in terms of the blendshape weights $\mathbf{w}$. Analogously, the displacement $\mathbf{v} - \mathbf{v}_0$ can be written as $\delta \mathbf{V}\,\mathbf{w}$. Plugging this into the above prior energy and writing the Laplacian as a $(3N \times 3N)$ matrix $\mathbf{L}$, again using the cotangent weights [Pinkall and Polthier 1993], leads to the formulation of the prior energy in terms of $\mathbf{w}$:

$$E_{\text{Prior}}(\mathbf{w}) \;=\; \frac{1}{N} \, \|\mathbf{L} \, \delta \mathbf{V} \, \mathbf{w}\|^2 \, . \tag{15}$$

The combination of the fitting energy (2), the geometric prior energy (15), and a sparsity regularization $E_{\text{Sparse}}(\mathbf{w}) = \frac{1}{K} \|\mathbf{w}\|_1$, leads to our objective function for facial retargeting:

$$E_{\text{Retarget}}(\mathbf{w}) \;=\; \underbrace{E_{\text{Fit}}(\mathbf{w})}_{\text{ActorSpace}} + \mu \underbrace{E_{\text{Prior}}(\mathbf{w})}_{\text{RigSpace}} + \nu \underbrace{E_{\text{Sparse}}(\mathbf{w})}_{\text{WeightSpace}} \, . \tag{16}$$

Unlike previous approaches, our energies operate in the spaces where modifications of the input data are minimal. First, E_{Fit} is computed in the actor space, such that the incoming animation is not modified. In contrast, E_{Prior} is computed in the rig space based on the original blendshapes. Finally, in accordance with common practices in manual key-framing [Seol et al. 2011], weight activation sparsity is directly enforced on the blendshape parameter space, which simplifies any subsequent manual editing of the animation. Under the assumption that the actor's markers are saved in cm and the face rig has been uniformly scaled to roughly match the actor's head, we recommend to set $\mu = 0.3$ and $\nu = 0.6$.

The retargeting was implemented as a Maya command plugin using the Maya C++ API. We pre-compute the matrix product $\mathbf{L}\delta\mathbf{V}$ as it remains constant over time. Due to the L_1 sparsity term, the retargeting solves an Iteratively Re-Weighted Least Squares problem of size $(K \times K)$ using the Eigen library [Guennebaud et al. 2016]. On average we achieve 105 fps, which confirms that the proposed prior is very suitable for real-time applications.

6 EVALUATION

In order to evaluate our blendshape transfer algorithm and our geometric prior, we first compare each part to common alternative formulations in a realistic but simple scenario. Doing so has the advantage that contributions can be evaluated individually and limitations of each method become clearly visible. The results of this evaluation are also shown in the accompanying video.

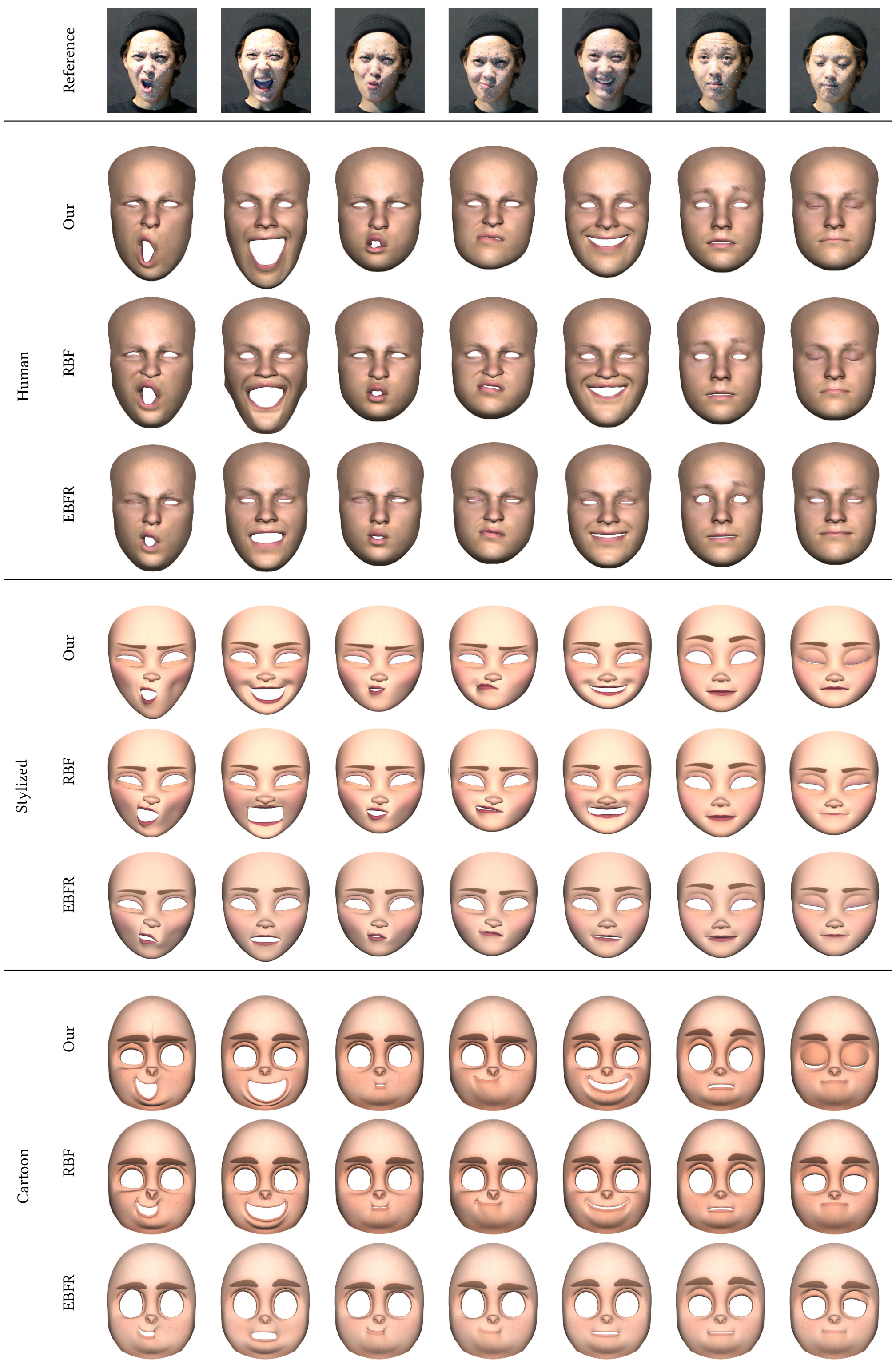

Fig. 12. Retargeting using personalized blendshapes created by our algorithm (*Our*), RBF proportion matching (*RBF*), and example-based facial rigging (*EBFR*). In all cases facial semantics are restored faithfully, and in particular for stylized characters our algorithm outperforms the other approaches.

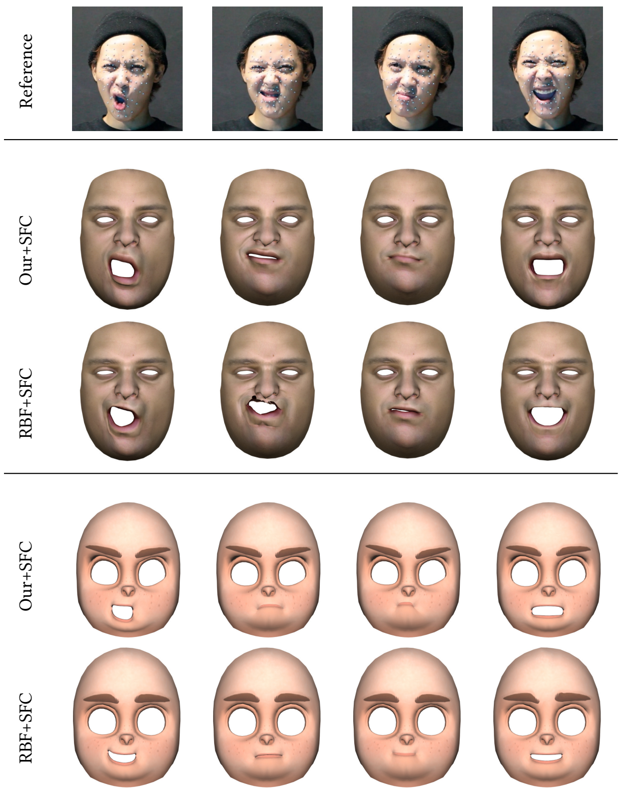

Fig. 13. Retargeting using space-time facial cloning with our personalized blendshapes (*Our+SFC*) and with the RBF proportion matching (*RBF+SFC*). Please notice the improved expressivity and fewer artifacts for our method. ©Motion capture: Feel Ghood Music ©Face rigs: Jason Osipa, Jana Bergevin

6.1 Automatic Blendshape Transfer

As described previously, RBF-deformation and deformation transfer are the most common techniques for personalizing blendshapes. We compare our algorithm to the RBF-proportion matching [Seol et al. 2012] and example-based facial rigging [Li et al. 2010]. For the example-based facial rigging we select 15 distinctive facial expressions from our captured sequence as training examples. Distinctiveness is guaranteed by clustering all expressions of the training animation using k-means. The expressions closest to the cluster centers are then chosen as examples. We initialize the corresponding blendshape weights as the result of our retargeting algorithm for these specific frames. Since this method alternatingly optimizes for blendshape geometries and blendshape weights, it requires only approximated blendshapes weights at the beginning. No peak expressions are required as examples.

We retarget the input animation to both, realistic and stylized characters. For this comparison, we only use E_{Sparse} for regularization and exclude any other prior. As shown in Figure 12, our blendshape transfer outperforms existing approaches, in particular if facial proportions differ significantly. In all cases plausible results are obtained, and in nearly all cases expression intensity is restored faithfully. In contrast, the quality of the RBF-proportion matching [Seol et al. 2012] and example-based facial rigging [Li et al. 2010] degrades with a higher degree of stylization. Figure 13 demonstrates that other retargeting algorithms, like space-time facial cloning [Seol et al. 2012], also benefit from our more accurate personalized blendshapes.

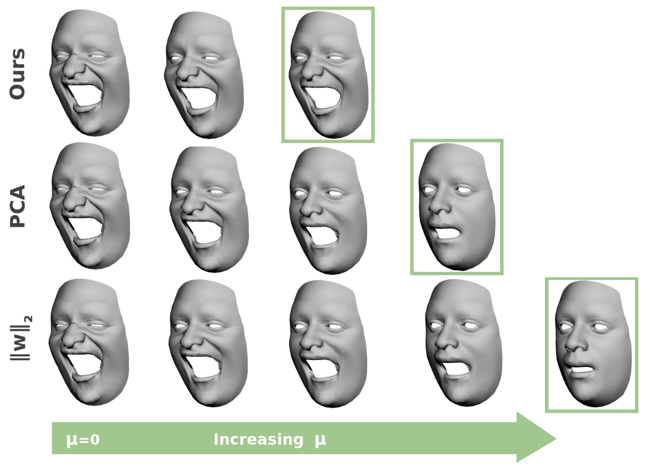

Fig. 14. Comparison of our geometric prior (*top*), the model-specific PCA prior (*middle*) and the L_2 regularization (*bottom*). From *left* to *right*, starting with a retargeting without regularization, we successively increase prior activation until a solution without self-intersections is obtained. ©Face rig: Jason Osipa

6.2 Geometric Prior

Next we compare the effectiveness of the proposed geometric prior to different well-established facial priors. To illustrate the effect of the different priors on the retargeting results, we in this section generate personalized blendshapes using the RBF proportion matching [Seol et al. 2012], since this method yielded the most artifacts in the evaluation of Section 6.1. The actual retargeting is then computed as described in Section 5, where we replace the prior energy E_{Prior} of Equation (16) by several options. Note that we employ the sparsity energy E_{Sparse} for all examples.

As a simple prior we incorporate an L_2 weight regularization $E_{\text{Prior}}(\mathbf{w}) = \|\mathbf{w}\|^2$. The combination of L_2 regularization and L_1 regularization (E_{Sparse}), also known as Elastic Net [Zou and Hastie 2005], enforces small weights but is less restrictive than non-negativity constraints. This is beneficial since there exist valid facial expressions with negative weights. In addition, we compare our geometric prior to the model-specific PCA prior [Seol et al. 2012]. The PCA is constructed from the given blendshapes only, in order to have equal input conditions. For the comparison, we identify the frames with strongest artifacts and step-wise increase μ until the geometric artifacts are removed. Figure 14 shows the effect of each prior for one representative example. Our prior converges the fastest to an artifact-free expression, because it minimizes bending at the vertex level, without being limited to the given examples as in the PCA case. In addition, Figure 15 compares our real-time capable retargeting with the offline method of Seol et al. [2012].

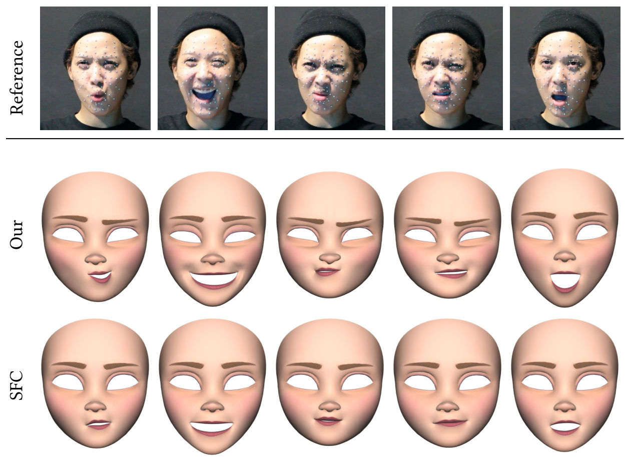

Fig. 15. Retargeting using our personalized blendshapes in combination with the geometry prior (*Our*) and space-time facial cloning (*SFC*) for offline retargeting. Semantics are restored faithfully and the high expressivity of the original actor is maintained. Especially for stylized characters our real-time algorithm outperforms even offline state of the art approaches.
©Motion capture: Feel Ghood Music ©Face rig: meryproject.com

6.3 Discussion and Limitations

In rare cases we observed inaccuracies during retargeting for our algorithm, RBF-proportion matching, and example based facial rigging. While different methods might work better for specific frames, our method performs better when considering the overall sequence. A particular strength of our method is that it also works for highly stylized characters, as long as the blendshapes approximately reflect natural facial movements.

Like any data-driven approach, the performance of our method depends on the availability of good training data, which in our case is an expressive facial sequence. However, because we do not enforce to capture isolated FACS expressions, we consider this requirement as minor: In an offline scenario the entire animation sequence can be used for training, while for real-time applications it takes only a few seconds to create such a sequence. Our results were computed using the full animation sequence as the training sequence for a fair comparison with the offline retargeting of Seol et al. [2011]. In the absence of expressive data for certain expressions, mesh constraints (Section 4.4) ensure plausible blendshapes that are similar to the well-established expression transfer methods, while blendshape constraints (Section 4.5) balance the changes occurrying to similar blendshapes.

We observed that mouth expressions are highly similar (e.g., o-viseme, kiss, mouth-open), making it very difficult to identify equivalent blendshapes across different identities in the training sequence (Figure 16). In such cases, E_{Match} tends to under-estimate the transfered blendshapes. Although this is addressed by enhancing the contrast of the similarities to obtain a better fit to the actor's expressions (Equations (7), (8)), the adjusted blendshapes might be more subtle than the actor's expressions, resulting in slightly exaggerated expressions (Figure 16, bottom) at the retargeting step. However, the user can always return to the default RBF-based deformation transfer using only E_{Mesh} and E_{CEG} (Figure 17).

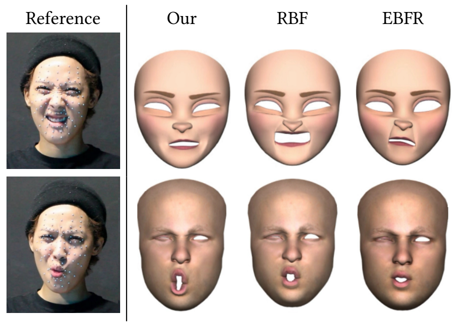

Fig. 16. Semantic changes for subtle expressions appear for all approaches. Our method allows the user to seamlessly blend between our and RBF blendshape transfer by increasing the weights of E_{Mesh} and E_{CEG}.
©Motion capture: Feel Ghood Music ©Face rigs: meryproject.com, Mark Pauly.

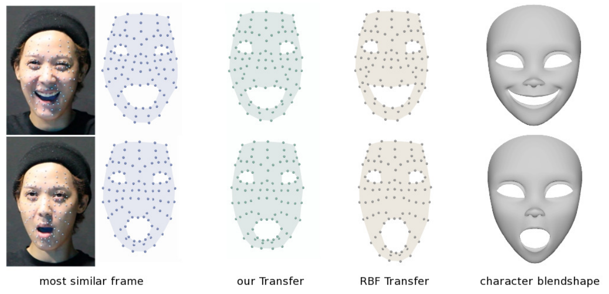

Fig. 17. Sparse blendshape comparison of the initial guess (RBF transfer) and our blendshape transfer method. Sparse blendshapes of the character rig (*right*) are adjusted to match the actor's proportions (*left*).
©Motion capture: Feel Ghood Music ©Face rigs: meryproject.com

While our approach shows convincing results on the tested blendshape rigs and animations, our alignment might not reproduce the desired result if blendshapes model a very cartoony behavior (e.g., popping eyes) or vary in the amount of facial features (e.g., different number of eyes). Cross-mapping approaches to facial animation retargeting are better suited for this type of animation.

Our method does not address the transfer of fine-scale details. In practice, it is not often desired, as fine-scale details are already encoded in the facial rig and should remain consistent when the animation is transferred from different actors.

By replacing the automatically computed correlations with manually selected weights for specific examples, our method can easily be extended to a semi-supervised method. Because our similarity measure is limited to the range $[0, 1]$, which is equivalent to the recommended weight space for blendshapes, such an extension would be straightforward.

7 CONCLUSION

Retargeting with properly corrected blendshapes is crucial if the intensity of facial expressions should be retained. We exploit the inherent similarities between facial expressions of different proportions to generate, through a combination of statistical and geometric methods, a parallel parametrization that fits the range of motion of the actor and preserves the semantic relationships and geometric properties of the character's blendshapes. Furthermore, we introduced a new prior that takes advantage of the differential mesh properties.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIP) (NRF-2017R1A2A1A05000979). Eduard Zell and Mario Botsch were supported by the Cluster of Excellence Cognitive Interaction Technology "CITEC" (EXC 277) at Bielefeld University, funded by the German Research Foundation (DFG). Roger B. R. was additionally supported by Digital Idea Inc. We thank Yeongho Seol for sharing his work for comparison. The Loki, Osipa, Mery, and Billy characters (Table 1) are courtesy of Mark Pauly, Jason Osipa, meryproject.com, and Jana Bergevin, respectively. We would like to express our most sincere gratitude to Feel Ghood Music (www.feelghood.com) and especially to Ms. Yoon Mirae for their generosity in allowing their image and captured data to be part of this work.

REFERENCES

Ken Anjyo, Hideki Todo, and J. P. Lewis. 2012. A Practical Approach to Direct Manipulation Blendshapes. *Journal of Graphics Tools* 16, 3 (2012), 160–176.

Autodesk. 2016. MAYA. (2016). www.autodesk.com/maya

Vincent Barrielle, Nicolas Stoiber, and Cédric Cagniart. 2016. BlendForces: A Dynamic Framework for Facial Animation. *Computer Graphics Forum* 35, 2 (2016).

Mikhail Belkin and Partha Niyogi. 2005. Towards a Theoretical Foundation for Laplacian-Based Manifold Methods. In *Proc. Conference on Learning Theory.* 486–500.

Bernd Bickel, Mario Botsch, Roland Angst, Wojciech Matusik, Miguel Otaduy, Hanspeter Pfister, and Markus Gross. 2007. Multi-scale Capture of Facial Geometry and Motion. *ACM Trans. Graph.* 26, 3 (2007).

Mario Botsch and Olga Sorkine. 2008. On Linear Variational Surface Deformation Methods. *IEEE Trans. Vis. Comput. Graphics* 14, 1 (2008), 213–230.

Sofien Bouaziz and Mark Pauly. 2014. *Semi-Supervised Facial Animation Retargeting.* Technical Report 202143. EPFL.

Sofien Bouaziz, Yangang Wang, and Mark Pauly. 2013. Online Modeling for Realtime Facial Animation. *ACM Trans. Graph.* 32, 4 (2013), 40:1–40:10.

Christoph Bregler, Lorie Loeb, Erika Chuang, and Hrishi Deshpande. 2002. Turning to the Masters: Motion Capturing Cartoons. *ACM Trans. Graph.* 21, 3 (2002), 399–407.

Ian Buck, Adam Finkelstein, Charles Jacobs, Allison Klein, David H. Salesin, Joshua Seims, Richard Szeliski, and Kentaro Toyama. 2000. Performance-driven Hand-drawn Animation. In *Proc. Symp. on Non-Photorealistic Animation and Rendering.* 101–108.

Chen Cao, Qiming Hou, and Kun Zhou. 2014. Displaced Dynamic Expression Regression for Real-time Facial Tracking and Animation. *ACM Trans. Graph.* 33, 4 (2014), 43:1–43:10.

Erika Chuang and Christoph Bregler. 2002. *Performance Driven Animation using Blendshape Interpolation.* Technical Report CS-TR-2002-02. Stanford University.

Patrick Coleman, Jacobo Bibliowicz, Karan Singh, and Michael Gleicher. 2008. Staggered Poses: A Character Motion Representation for Detail-preserving Editing of Pose and Coordinated Timing. In *Proc. Symp. on Computer Animation.* 137–146.

Zhen Cui, Shiguang Shan, Haihong Zhang, Shihong Lao, and Xilin Chen. 2012. Image Sets Alignment for Video-Based Face Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition.* 2626–2633.

Zhigang Deng, Pei-Ying Chiang, Pamela Fox, and Ulrich Neumann. 2006. Animating Blendshape Faces by Cross-mapping Motion Capture Data. In *Proc. Symp. on Interactive 3D Graphics and Games.* 43–48.

Paul Ekman and Wallace V. Friesen. 1978. *Facial Action Coding System: A Technique for the Measurement of Facial Movement.* Consulting Psychologists Press.

Ke Fan, Ajmal Mian, Wanquan Liu, and Ling Li. 2016. Unsupervised manifold alignment using soft-assign technique. *Machine Vision and Applications* 27, 6 (2016), 929–942.

Gaël Guennebaud, Benoît Jacob, and others. 2016. Eigen v3.3. (2016). http://eigen.tuxfamily.org

Alexandru Eugen Ichim, Sofien Bouaziz, and Mark Pauly. 2015. Dynamic 3D Avatar Creation from Hand-held Video Input. *ACM Trans. Graph.* 34, 4 (2015), 45:1–45:14.

Alexandru-Eugen Ichim, Ladislav Kavan, Merlin Nimier-David, and Mark Pauly. 2016. Building and Animating User-specific Volumetric Face Rigs. In *Proc. Symp. on Computer Animation.* 107–117.

Eric Jones, Travis Oliphant, Pearu Peterson, and others. 2001. SciPy: Open source scientific tools for Python. (2001). http://www.scipy.org/

Natasha Kholgade, Iain Matthews, and Yaser Sheikh. 2011. Content Retargeting Using Parameter-parallel Facial Layers. In *Proc. Symp. on Computer Animation.* 195–204.

Ravikrishna Kolluri, Jonathan R. Shewchuk, and James F. O'Brien. 2004. Spectral Surface Reconstruction from Noisy Point Clouds. In *Proc. Symp. of Geometry Processing.* 11–21.

Jérôme Kunegis, Stephan Schmidt, Andreas Lommatzsch, Jürgen Lerner, Ernesto W. De Luca, and Sahin Albayrak. 2010. Spectral Analysis of Signed Graphs for Clustering, Prediction and Visualization. In *Proc. Int. Conference on Data Mining.*

Manfred Lau, Jinxiang Chai, Ying-Qing Xu, and Heung-Yeung Shum. 2009. Face Poser: Interactive Modeling of 3D Facial Expressions Using Facial Priors. *ACM Trans. Graph.* 29, 1 (2009), 3:1–3:17.

J. P. Lewis, Ken Anjyo, Taehyun Rhee, Mengjie Zhang, Fred Pighin, and Zhigang Deng. 2014. Practice and Theory of Blendshape Facial Models. In *Eurographics State of the Art Reports.*

J. P. Lewis, Jonathan Mooser, Zhigang Deng, and Ulrich Neumann. 2005. Reducing Blendshape Interference by Selected Motion Attenuation. In *Proc. Symp. on Interactive 3D Graphics and Games.* 25–29.

Hao Li, Thibaut Weise, and Mark Pauly. 2010. Example-based Facial Rigging. *ACM Trans. Graph.* 29, 4 (2010), 32:1–32:6.

Hao Li, Jihun Yu, Yuting Ye, and Chris Bregler. 2013. Realtime Facial Animation with On-the-fly Correctives. *ACM Trans. Graph.* 32, 4 (2013), 42:1–42:10.

Junyong Noh and Ulrich Neumann. 2001. Expression Cloning. In *Proc. of SIGGRAPH.* 277–288.

Verónica Orvalho, Pedro Bastos, Frederic Parke, Bruno Oliveira, and Xenxo Alvarez. 2012. A Facial Rigging Survey. In *Eurographics State of the Art Reports.*

Verónica Costa Orvalho, Ernesto Zacur, and Antonio Susin. 2008. Transferring the Rig and Animations from a Character to Different Face Models. *Computer Graphics Forum* 27, 8 (2008), 1997–2012.

Sinno Jialin Pan and Qiang Yang. 2010. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering* 22, 10 (2010), 1345–1359.

Frederic I. Parke and Keith Waters. 2008. *Computer Facial Animation* (2 ed.). AK Peters Ltd.

Yuru Pei, Fengchun Huang, Fuhao Shi, and Hongbin Zha. 2012. Unsupervised Image Matching Based on Manifold Alignment. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34, 8 (2012), 1658–1664.

Ulrich Pinkall and Konrad Polthier. 1993. Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics* 2, 1 (1993), 15–36.

Jun Saito. 2013. Smooth Contact-aware Facial Blendshapes Transfer. In *Proc. Symp. on Digital Production.* 7–12.

Jaewoo Seo, Geoffrey Irving, J. P. Lewis, and Junyong Noh. 2011. Compression and Direct Manipulation of Complex Blendshape Models. *ACM Trans. Graph.* 30, 6 (2011), 164:1–164:10.

Yeongho Seol, J. P. Lewis, Jaewoo Seo, Byungkuk Choi, Ken Anjyo, and Junyong Noh. 2012. Spacetime Expression Cloning for Blendshapes. *ACM Trans. Graph.* 31, 2 (2012), 14:1–14:12.

Yeongho Seol, Wan-Chun Ma, and J. P. Lewis. 2016. Creating an Actor-specific Facial Rig from Performance Capture. In *Proc. Symp. on Digital Production.* 13–17.

Yeongho Seol, Jaewoo Seo, Paul Hyunjin Kim, J. P. Lewis, and Junyong Noh. 2011. Artist Friendly Facial Animation Retargeting. *ACM Trans. Graph.* 30, 6 (2011), 162:1–162:10.

Jaewon Song, Byungkuk Choi, Yeongho Seol, and Junyong Noh. 2011. Characteristic facial retargeting. *Computer Animation and Virtual Worlds* 22, 2-3 (2011), 187–194.

Robert W. Sumner and Jovan Popović. 2004. Deformation Transfer for Triangle Meshes. *ACM Trans. Graph.* 23, 3 (2004), 399–405.

Justus Thies, Michael Zollhöfer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. 2016. Face2Face: Real-Time Face Capture and Reenactment of RGB Videos. In *IEEE Conference on Computer Vision and Pattern Recognition.* 2387–2395.

Chang Wang and Sridhar Mahadevan. 2009. Manifold Alignment Without Correspondence. In *Proc. International Joint Conference on Artifical Intelligence.* 1273–1278.

Chang Wang and Sridhar Mahadevan. 2011. Heterogeneous Domain Adaptation Using Manifold Alignment. In *Proc. International Joint Conference on Artificial Intelligence.* 1541–1546.

Chang Wang and Sridhar Mahadevan. 2013. Manifold Alignment Preserving Global Geometry. In *Proc. International Joint Conference on Artificial Intelligence.* 1743–1749.

Yang Wang, Xiaolei Huang, Chan-Su Lee, Song Zhang, Zhiguo Li, Dimitris Samaras, Dimitris Metaxas, Ahmed Elgammal, and Peisen Huang. 2004. High Resolution Acquisition, Learning and Transfer of Dynamic 3-D Facial Expressions. *Computer Graphics Forum* 23, 3 (2004), 677–686.

Thibaut Weise, Sofien Bouaziz, Hao Li, and Mark Pauly. 2011. Realtime Performance-based Facial Animation. *ACM Trans. Graph.* 30, 4 (2011), 77:1–77:10.

Feng Xu, Jinxiang Chai, Yilong Liu, and Xin Tong. 2014. Controllable High-fidelity Facial Performance Transfer. *ACM Trans. Graph.* 33, 4 (2014), 42:1–42:11.

Hui Zou and Trevor Hastie. 2005. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67, 2 (2005), 301–320.