Video-Based Facial Re-Animation

Wolfgang Paier
Fraunhofer HHI
wolfgang.paier
@hhi.fraunhofer.de

Markus Kettern
Fraunhofer HHI
markus.kettern
@hhi.fraunhofer.de

Anna Hilsmann
Fraunhofer HHI
anna.hilsmann
@hhi.fraunhofer.de

Peter Eisert
Fraunhofer HHI/
Humboldt University Berlin
peter.eisert
@hhi.fraunhofer.de

ABSTRACT

Generating photorealistic facial animations is still a challenging task in computer graphics, and synthetically generated facial animations often do not meet the visual quality of captured video sequences. Video sequences on the other hand need to be captured prior to the animation stage and do not offer the same animation flexibility as computer graphics models. We present an inexpensive method for video-based facial animation, which combines the photorealism of real videos with the flexibility of CGI-based animation by extracting dynamic texture sequences from existing multi-view footage. To synthesize new facial performances, these texture sequences are concatenated in a motion-graph-like way. In order to ensure realistic appearance, we combine a warp-based optimization scheme with a modified cross dissolve to prevent visual artefacts during the transition between texture sequences. Our approach makes photorealistic facial re-animation from existing video footage possible, which is especially useful in applications like video editing or the animation of digital characters.

Categories and Subject Descriptors

H.5.1 [**Multimedia Information Systems**]: Animations; I.4.8 [**Scene Analysis**]: Tracking

Keywords

facial animation, facial texture, geometric proxy, tracking

1. INTRODUCTION

The creation of realistic virtual human characters and especially of human faces is one of the most challenging tasks in computer graphics. The geometric and reflective properties of the human face are very complex and hard to model. Deformation is induced by complex interactions of a large number of muscles as well as several layers of tissue and skin. Reflective properties vary from diffuse to highly specular areas and also show subsurface scattering effects. Furthermore, humans are very good at interpreting faces, such that even slight deviations from the expected visual appearance are perceived as wrong or unnatural facial expressions. Therefore, synthetically generated facial expressions are often not perceived as realistic as real video sequences. Video sequences on the other hand need to be captured in advance, and editing the captured facial performances is difficult.

In this paper, we present a video-based approach to re-animate an actor's face. Our approach does not try to model the facial appearance with low dimensional statistical models as this would drop important details that cannot be represented in a low dimensional space. We rather base our method on real video footage which is transformed to dynamic texture sequences to achieve a photorealistic appearance. These dynamic textures are then processed in a way that allows for seamless concatenation of texture sequences according to an animator's input, enabling the creation of novel facial video performances. The presented approach can for example be used in computer games or video editing applications where it allows to conveniently synthesize realistic facial performances from a database of source sequences, to seamlessly transfer a facial performance between different video sequences, to re-arrange a sequence of facial video or to re-animate a digital character (e.g. in video games).

For the extraction of the dynamic texture sequences containing the facial performance (section 4), we use a pre-created 3D proxy model to consistently track the head pose in one or multiple synchronized video streams (section 4.2). The proxy mesh is a-priorily created using a single shot of the actor's head using a multi-view D-SLR camera rig and a dense image-based 3D-reconstruction method (section 4.1). After dynamic texture extraction (section 4.3), the dynamic textures can be used to synthesize novel facial performances by seamlessly concatenating several texture sequences (section 5). We use a combination of warp-based image registration (section 5.1) and cross dissolve blending (section 5.2) to create seamless transitions between consecutive texture sequences.

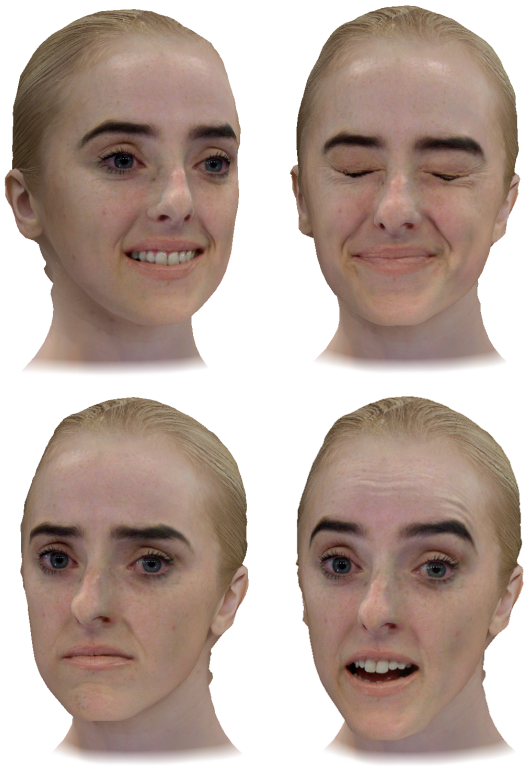

Figure 1: Examples of our results. The proxy head model is rendered with different facial expressions and from different viewpoints

2. RELATED WORK

Performance capture is a popular technique to make the performance of an actor reusable. It can for example be exploited to drive the skeletal animation of a human character, to transfer facial expressions by matching the face geometry [31], or to aid the creation of more realistic video dubbings [15] and speech driven animation [12, 2]. In this paper, we mainly focus on facial performance capture. A detailed survey on this topic can for example be found in [26]. Our approach is related to marker-less facial motion capture like [21, 30, 11, 5, 4, 10] since we use an optical flow-based technique to accurately track the head pose in the video stream. Borshukov et al. [5] use a highly sophisticated capture setup consisting of 8 infrared cameras plus 3 synchronized, high definition color cameras to capture an actor's facial performance in an ambient lighting setup. Based on 70 small retro-reflective markers, they drive a previously scanned 3D model of the actor's face to obtain a time-consistent animation mesh and to extract dynamic textures. Mesh sequences and dynamic textures are then used to perform a motion graph like animation of the actor's face. A different approach is presented by Garrido et al. [13], where a blend shape model is personalized by matching it to a detailed 3D-scan of the actor's face. This blend shape model is then used to track the facial performance using a combination of sparse facial features and optical flow. Alexander et al. [1] describe an approach to create extremely realistic digital characters but their approach also requires a highly sophisticated capture setup and additional human effort.

These approaches achieve highly realistic results using complex and expensive capture setups. In contrast, our approach keeps the complexity as low as possible while at the same time aiming at photorealistic animations. This makes our approach a valuable option for low profile productions to create photorealistic facial animations without the need for sophisticated facial performance capture setups or professional 3D animators. We also do not rely on a fully animatable 3D model of the actor's face, but on a roughly approximated geometric model with only a few degrees of freedom. In the experiments presented in this paper, we only use a single blendshape to account for mouth opening and closing. The necessary expressivity is achieved by using photorealistic dynamic textures which add fine details and facial movements. Similar strategies have been used for other applications in [22, 7, 11, 19, 29, 9, 23]. Using image-based rendering, these methods create photorealistic renderings from novel viewpoints though the used geometry is only a rough approximation. Pushing this idea further, Xu et al. [32] presented a system for the synthesis of novel full body performances from multi-view video. They use performance capture to obtain pose and geometry for each video frame. Based on this data, they render a synthetic video performance according to a user provided query viewpoint and skeleton-sequence, even if the exact body pose is not represented in the database. However, they mention that while their approach is appropriate for skeletal animation, facial animation has to be handled separately.

As humans are very sensitive to inconsistencies in the appearance of other human faces, we specifically concentrate on facial re-animation, in contrast to the previously mentioned papers. Inspired by the aforementioned advances in image- and video-based rendering, our approach is based on real video footage to achieve photorealistic results. Video-based facial animation has only recently found attention in the literature. Paier et al. [23] presented a system for facial re-targeting, i.e. transferring short sequences of a facial performance between different videos. This can be used in video editing to fine tune the timing of facial actions or to exchange similar facial expressions in order create a flawless shot from already captured shots. In contrast to [23], we focus on synthesizing completely new sequences from short clips of facial expressions of an actor. Furthermore, by extracting dynamic textures and the use of an approximate head model that allows for jaw movements, novel animations can be rendered from arbitrary viewpoints.

Our re-animation strategy is also related to the idea of motion graphs [20, 17, 8] that have already been successfully used for skeletal or surface-based animation of human character and faces [5]. We capture several video sequences of a facial performance and split them up into short clips that contain single actions or facial expressions (e.g. smile, talk, looking surprised or angry). These clips are transformed to texture space and are concatenated in order to compose a novel facial performance. Similar to [18, 25], we also find smooth transitions between different facial sequences because directly switching from one texture sequence to another would create obvious artifacts in the synthesized facial video

(e.g. sudden change of facial expression or illumination). For this purpose, we use a geometric image registration technique to compensate tracking errors and changes in the facial expression as well as a modified cross dissolve to smoothly blend all remaining color differences.

3. SYSTEM OVERVIEW

The workflow of the presented system consists of two main steps (see figure 2). First, in a pre-processing phase a database of dynamic textures, each containing a certain facial expression/action, is created. This step has to be done only once in advance. After this database has been set up, new facial videos can be created in real time according to user input by seamlessly concatenating selected dynamic textures and rendering them using an approximate model of the actor's head.

The input data for the extraction of dynamic textures consists of a multi-view video stream that contains several facial performances of an actor (see figure 5) as well as a calibrated 360° multi-view set of still images showing the actor's head with a neutral expression (see figure 3). First, we reconstruct the head geometry based on the still images and run a semi-automatic mesh unwrapping technique to generate texture coordinates. The extracted head geometry will be used to consistently track the head pose in the multi-view video streams. It is an approximation of the true geometry since it is almost static and allows for dominant deformations only (in this paper the only possible deformation is jaw movement). More subtle deformations will be expressed by dynamic textures. The following steps process the multi-view video stream only. We label several facial expressions/actions (e.g. neutral, happy, talking, ...) by storing a tag as well as the first and the last frame of each facial performance. Then, using the extracted head geometry, the face is tracked through all frames in all annotated sequences and temporally consistent textures are created for each multi-view frame. These pre-processing steps need to be performed only once in advance and are detailed in section 4.

Input to the synthesis of facial videos is a user defined sequence of facial expression labels. Based on these labels, the processed dynamic texture sequences are combined to re-animate the face either by directly rendering the mesh with animated textures (e.g. games or virtual reality applications) or by rendering the sequence to a target video. In order to ensure a seamless concatenation of the texture sequences, we apply a two stage blending approach. First, in a pre-defined transition window at the end of the current sequence and the beginning of the following one, we adjust the motion using a warp-based optimization technique. Finally, we apply a cross dissolve-based color adjustment. Details on the the synthesis of novel facial expression sequences, i.e. facial re-animation, are given in section 5.

4. CREATING A DATABASE OF DYNAMIC TEXTURES

This section explains how we extract a database of dynamic textures from a multi-view video stream. First, the actor is captured in neutral position using a calibrated 360° multi-view setup of D-SLR cameras to generate a 3D head model (section 4.1). Then, several facial performances are captured

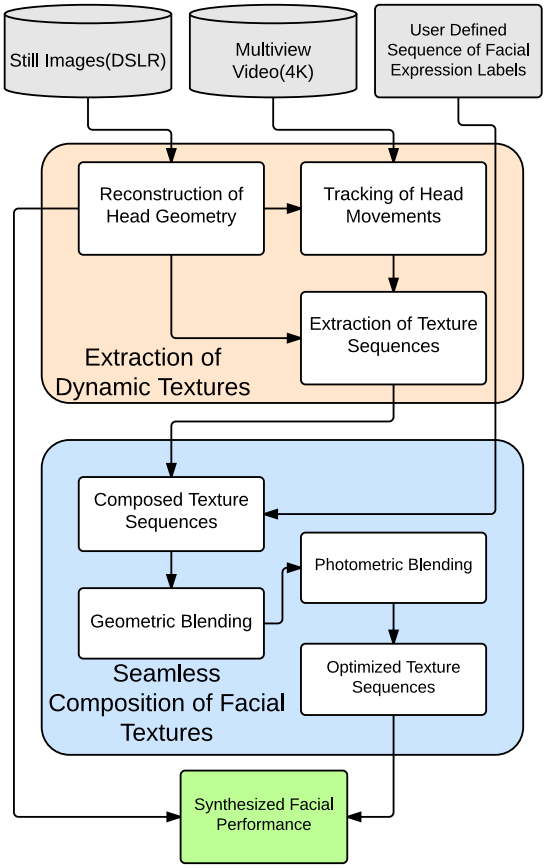

Figure 2: Schematic system overview

with a multi-view video setup, and the 3D head model is used to track the 3D pose and orientation of the actor's head and jaw movements in each sequence (section 4.2). This allows us to extract temporally consistent textures (section 4.3) from the multi-view video sequences in order to set up a database of dynamic textures.

4.1 Generation of 3D Head Models

For the extraction of dynamic textures from multi-view video, a 3D representation of the actor's head is generated from a calibrated multi-view set of D-SLR images. We employ a state-of-the-art image-based dense stereo matching scheme to estimate depth-maps for each D-SLR pair [3]. The resulting 2.5D models are then registered using an iterative closest point approach and merged into a complete 3D head model (see figure 4). This method provides an accurate reconstruction of the facial geometry and a realistic approximation of the actor's hair.

The reconstructed head geometry is almost static and only used as a geometric proxy in all following processing steps. Typically, a neutral facial expression is used for the proxy mesh since this provides a reasonable approximation of the head geometry for most facial expressions. Finally, a semiautomatic method for mesh unwrapping is used to create texture coordinates for each vertex. Note that this step needs

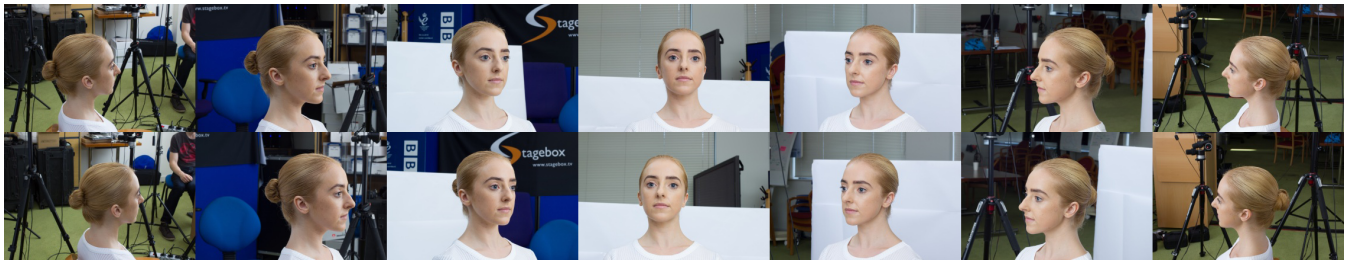

Figure 3: Samples of the still images used to reconstruct the head geometry

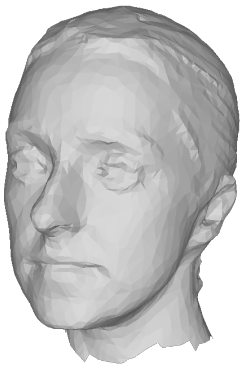

Figure 4: Image of the geometric proxy used for tracking, texture extraction, and rendering

to be done only once for each actor, and can then be used to process all video sequences of this actor.

4.2 Head Motion Tracking

In order to enable the extraction of temporally consistent textures from video streams displaying facial performances, the 3D model used for texture extraction should follow this motion as closely as possible. However, correctly tracking the subtle geometric deformations of a face is considered to be a very hard computer vision problem (e.g. [14]) and even state-of-the-art methods may quickly lose track due to the manifold deformations or large head rotations occurring in natural facial performances, producing visually disturbing artifacts during re-animation. In order to allow for photo-realistic animations, the overall idea of our approach is to express all subtle facial deformations by animating the texture rendered upon the geometry instead of modeling them in 3D. The only type of deformation we consider impossible to represent by texture alone is jaw movement since it largely deforms the face boundary where strong depth discontinuities would severely hamper the results of any approach relying on texture alone.

Thus, the rigid head motion and jaw movement have to be separated from deformations due to facial expressions. We achieve this by tracking the actor's face with the original proxy mesh and a single blend shape for downwards jaw movement which is easily created using 3D modeling software. Note that the method described below is not limited to a single blend shape and could also be used to track a full-blown blend shape model.

The tracking procedure is preceded by selecting a set of key-frames from the video stream and matching the proxy mesh to these key-frames via a small set of fiducial points either obtained from a facial feature extractor (e.g. [28]) or by manual annotation. In order to maximize semantic consistency of the extracted textures, we minimize the image difference between each frame and a reconstruction of that frame obtained by warping the last key-frame according to the estimated motion and deformation of the proxy mesh, resulting in an analysis-by-synthesis approach. The rigid motion of the head proxy model in frame s is defined by a rotation R^s around the model's center point and a translation $\mathbf{t}^s$. The jaw movement will be parametrized by a blend shape factor λ^s. Since we are working in a calibrated multi-view setup, each camera c also has a pose $(R_c, \mathbf{t}_c)$ and its projection of a 3D point $\mathbf{x}^s$ on the surface of the proxy model in frame s is given by

$$\begin{bmatrix} u \\ v \end{bmatrix} = \Psi_c \left(R_c^T \left(R^s \mathbf{x}^s + \mathbf{t}^s - \mathbf{t}_c \right) \right) \tag{1}$$

$$\Psi_c \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \mathbf{c}_c - \operatorname{diag}(\mathbf{f}_c) \begin{bmatrix} \frac{x}{z} \\ \frac{y}{z} \\ z \end{bmatrix} \tag{2}$$

where $\mathbf{c}_c$ and $\mathbf{f}_c$ denote the camera's principal point and scaled focal length, respectively. The position of point $\mathbf{x}^s$ in model space is defined by

$$\mathbf{x}^s = \mathbf{v} + \lambda^s \mathbf{b} \tag{3}$$

where $\mathbf{v}$ is the position in the original proxy mesh, $\mathbf{b}$ is the corresponding offset for blend shape animation and λ^s is the animation coefficient for jaw movement. Note that for a model with more than one blend shape, λ^s would be replaced by a vector of coefficients and $\mathbf{b}$ by a matrix containing the vertex offsets.

Since we estimate each frame by modifying a rendered version of its preceding frame, we may assume the rotation update for each new frame to be small enough to be approximated linearly. Thus, we can express the motion of a 3D model point in world coordinates as

$$\mathbf{w}^s = \mathbf{w}^{s-1} + \Delta^s_{\mathbf{x}} \tag{4}$$

$$= \Delta^s_R R^{s-1} \mathbf{x}^s + t^{s-1} + \Delta^s_t \tag{5}$$

$$\Delta^s_R = \begin{bmatrix} 1 & -r^s_z & r^s_y \\ r^s_z & 1 & -r^s_x \\ -r^s_y & r^s_x & 1 \end{bmatrix} \tag{6}$$

$$\Delta^s_{\mathbf{x}} = \left(\Delta^s_R - I \right) R^{s-1} \mathbf{x}^s + \Delta^s_t \tag{7}$$

This motion induces an offset for the projected position $\mathbf{u} = \begin{bmatrix} u & v \end{bmatrix}^T$ corresponding to $\mathbf{x}^s$ in the image of camera c which we represent by its first order approximation

$$\Delta^s_{\mathbf{u},c}\left(\boldsymbol{\theta}^s\right) \approx J_{\Psi_c} R_c^T \Delta^s_{\mathbf{x}} \tag{8}$$

where $\boldsymbol{\theta}^s = [\Delta^s_r, \Delta^s_t, \lambda^s]^T$ is the parameter vector consisting of the changes in model rotation, translation and jaw movement. J_Ψ is the Jacobian of the projection function given by

$$J_{\Psi_c} = \operatorname{diag}\left(\mathbf{f}_c\right) \begin{bmatrix} -\frac{1}{z} & 0 & \frac{x}{z^2} \\ 0 & -\frac{1}{z} & \frac{y}{z^2} \end{bmatrix} \tag{9}$$

for a 3D point $\begin{bmatrix} x & y & z \end{bmatrix}^T = R_c^T \left(R^s \mathbf{x}^s + \mathbf{t}^s - \mathbf{t}_c\right)$ in camera space.

Substituting (3) and (6) into (8) yields

$$\Delta^s_{\mathbf{u},c}\left(\boldsymbol{\theta}^s\right) = J_{\Psi_c} R_c^T \begin{bmatrix} \left[R^{s-1}\mathbf{v}\right]_\times \\ I_3 \\ -\left(R^{s-1}\mathbf{b}\right)^T \end{bmatrix}^T \boldsymbol{\theta}^s \tag{10}$$

where $[\mathbf{a}]_\times$ denotes the skew-symmetric cross-product matrix of vector $\mathbf{a}$. Note that this 2D motion offset is now expressed as linearly dependent on the parameters of the model's motion in 3D space given by $\boldsymbol{\theta}^s$. We can derive a matrix representing the induced motion of all pixels in an image area Ω by

$$\begin{bmatrix} \vdots \\ \Delta^s_{\mathbf{u},c} \\ \vdots \end{bmatrix}_{\mathbf{u}\in\Omega} = A_c \boldsymbol{\theta}^s \tag{11}$$

with each pair of rows in A_c given by

$$A_{\mathbf{u},c} = J_{\Psi_c} R_c^T \begin{bmatrix} -\left[R^{s-1}\mathbf{v}\right]_\times & I_3 & -R^{s-1}\mathbf{b} \end{bmatrix} . \tag{12}$$

The established relation between 3D and 2D motion is used to explain the optical flow between two images I^s_c and J^s_c. This amounts to minimizing the error

$$\mathcal{E}_{\mathrm{f}}(\boldsymbol{\theta}^s) = \sum_c \sum_{\mathbf{u}\in I} \left\| \left(\nabla I^s_{\mathbf{u},c}\right)^T \Delta^s_{\mathbf{u},c}\left(\boldsymbol{\theta}^s\right) - \left(J^s_{\mathbf{u},c} - I^s_{\mathbf{u},c}\right) \right\|^2 \tag{13}$$

with $\nabla I^s_{\mathbf{u},c}$ being the image gradient of I^s_c at pixel $\mathbf{u}$.

As described above, to prevent drifting errors, we use a rendered version of frame I^0_c, the last key-frame, as image J^s_c. This rendered version is created by projecting I^0_c onto the texture of the model mesh at its pose in I^0_c and then rendering the mesh with the estimate of its current pose. If Ω is the area covered by the rendered mesh, $\mathcal{E}_{\mathrm{f}}$ can be minimized in closed form by solving the system of linear equations given by

$$\nabla I^s_c A_c \boldsymbol{\theta}^s = J^s_c - I^s_c \tag{14}$$

evaluated in region Ω.

This yields a linearized estimate of the image variations induced by the parameters $\boldsymbol{\theta}^s = \begin{bmatrix} (\Delta^s_r)^T & (\Delta^s_t)^T & \lambda^s \end{bmatrix}^T$. Since this relation is in truth a non-linear one, we resort to an iterative optimization approach. Observe that (14) represents the set of normal equations for this non-linear problem so iteratively solving it and updating the rendered image J^s and the depth map for obtaining A_Ω results in a Gauss-Newton optimization. This process typically converges to yielding very small parameter updates within less than 10 iterations.

4.3 Dynamic Texture Generation

This step uses the results of the tracking procedure to transform the multi-view video streams (see figure 5) into a single stream of texture mosaics. Such a representation has several advantages. First, it can easily be integrated into existing rendering engines. Second, it eases the process of editing facial expressions as all texture information is merged into a single image. Finally, it reduces redundancy as in texture space only relevant data is stored and unnecessary data is dropped (e.g. background and overlaps).

Since our setup consists of multiple video cameras it is necessary to decide for each triangle f_i, from which camera it should receive its texture. This can be formulated as a labeling problem, estimating a camera label c_i for each triangle f_i of the proxy mesh.

In order to create an optimal sequence of texture mosaics for each facial expression/facial action, we employ a discrete optimization scheme minimizing an objective function (15) that consists of three terms, each corresponding to one visual quality criterion [23]: high visual quality, low visibility of seams and no temporal artifacts (e.g. flickering caused by rapidly changing source cameras).

$$\begin{aligned} \mathcal{E}_{\mathrm{t}}(C) = & \sum_t^T \sum_i^N \mathcal{D}(f_i^t, c_i^t) \\ & + \lambda \sum_{i,j\in\mathcal{N}} \mathcal{V}_{i,j}(c_i^t, c_j^t) \\ & + \eta \mathcal{T}(c_i^t, c_i^{t-1}) \end{aligned} \tag{15}$$

where C denotes the set of camera labels for all triangles. The first term $\mathcal{D}(f_i, c_i)$ measures the visual quality of a triangle f_i in camera c_i and uses a quality measure $\mathcal{W}(f_i, c_i)$, which is the area of f_i projected on the image plane of camera c_i relative to the sum of $area(f_i, c_i)$ over all possible c_i to ease the choice of the weighting factors η and λ:

$$\mathcal{D}(f_i, c_i) = \begin{cases} 1 - \mathcal{W}(f_i, c_i) & f_i \text{ is visible} \\ \infty & f_i \text{ is occluded} \end{cases} \tag{16}$$

$$\mathcal{W}(f_i, c_i) = \frac{area(f_i, c_i)}{\sum\limits_{c_j} area(f_i, c_j)} \tag{17}$$

The second term $\mathcal{V}_{i,j}(c_i, c_j)$ in (15) adds a spatial smoothness constraint to the objective function (15) which relates to the sum of color differences along the common edge $e_{i,j}$ of two triangles f_i and f_j that are textured from two cameras c_i and c_j.

$$\mathcal{V}_{i,j}(c_i, c_j) = \begin{cases} 0 & c_i = c_j \\ \Pi_{e_{i,j}} & c_i \neq c_j \end{cases} \tag{18}$$

$$\Pi_{e_{i,j}} = \int_{e_{i,j}} \left\| I_{c_i}(x) - I_{c_j}(x) \right\| dx \tag{19}$$

Figure 5: Samples of the multi-view video capture showing different facial expressions

Finally, a temporal smoothness term $\mathcal{T}(c_i, c_j)$ is added to the objective function. Without such a term, the resulting dynamic textures are not necessarily temporally consistent, i.e. the source camera of a certain triangle can change arbitrarily between two consecutive texture frames resulting in visually disturbing flickering in the resulting texture sequence. $\mathcal{T}$ increases the overall cost if the source camera c_i of a triangle f_i changes between two consecutive time steps.

$$\mathcal{T}(c_i^t, c_i^{t-t}) = \begin{cases} 0 & c_i^t = c_i^{t-t} \\ 1 & c_i^t \neq c_i^{t-t} \end{cases} \tag{20}$$

Finally, we employ a simple but effective global color matching [27] together with Poisson blending [24] modified for the usage in texture mosaics to conceal remaining seams without unnecessarily blurring the resulting texture or adding ghosting artifacts (which can be caused by simpler approaches like alpha-blending).

In case the video footage alone is not sufficient to create 360° dynamic textures, we allow the filling of missing regions and areas of low spatial resolution (caused by the viewing angle) with texture data from the D-SLR capture.

5. SYNTHESIS OF FACIAL VIDEOS

In the previous stage (section 4), we created a set of independent texture sequences. Each sequence represents a facial expression or action like smiling, talking, laughing, looking friendly or angry. We can now use the extracted dynamic textures to create photorealistic facial performances by playing the texture sequences on a static 3D model of the head like a video. This creates the photorealistic illusion of a talking mouth, blinking eyes or wrinkles caused by a smile without the need to model all fine deformations in the geometry. Furthermore, by looping and concatenating several texture sequences, longer and more complex sequences can be synthesized. This type of animation strategy is closely related to motion graphs [20]. In the context of motion graphs, edges in the graph would correspond to facial actions, and vertices to expression states.

Since the extracted dynamic textures have been captured separately and in a different order, simple concatenation of independent dynamic textures would create visual artifacts at the transition between two sequences. These artifacts are due to small tracking errors, changing illumination (e.g. caused by head movement) and differences in the facial expression at the end of one sequence and the beginning of another (see figure 6).

Figure 6: Intensity difference at a transition point. Bottom-left: previous frame, bottom-right: current frame, top: color difference at transition frame

Therefore, at this stage, the independent texture sequences from the pre-processing phase (section 4) are brought into connection by defining transition rules between the separate sequences. Between each pair of texture sequences, a two stage blending strategy is employed: first, the geometric misalignment between the last frame $\mathcal{T}_{last,t-1}$ of the previous texture sequence and the first frame $\mathcal{T}_{first,t}$ of the next sequence is corrected, before the remaining color differences are blended by a cross dissolve.

5.1 Geometric Blending

The geometric misalignment is compensated by calculating a 2D warp $\mathcal{W}(\mathcal{T}, \Phi)$ that maps $\mathcal{T}_{last,t-1}$ on $\mathcal{T}_{first,t}$, minimizing

$$\underset{\Phi}{argmin} \, \|\mathcal{T}_{first,t} - \mathcal{W}(\mathcal{T}_{last,t-1}, \Phi)\|^2 + \lambda \mathcal{R}(\Phi), \tag{21}$$

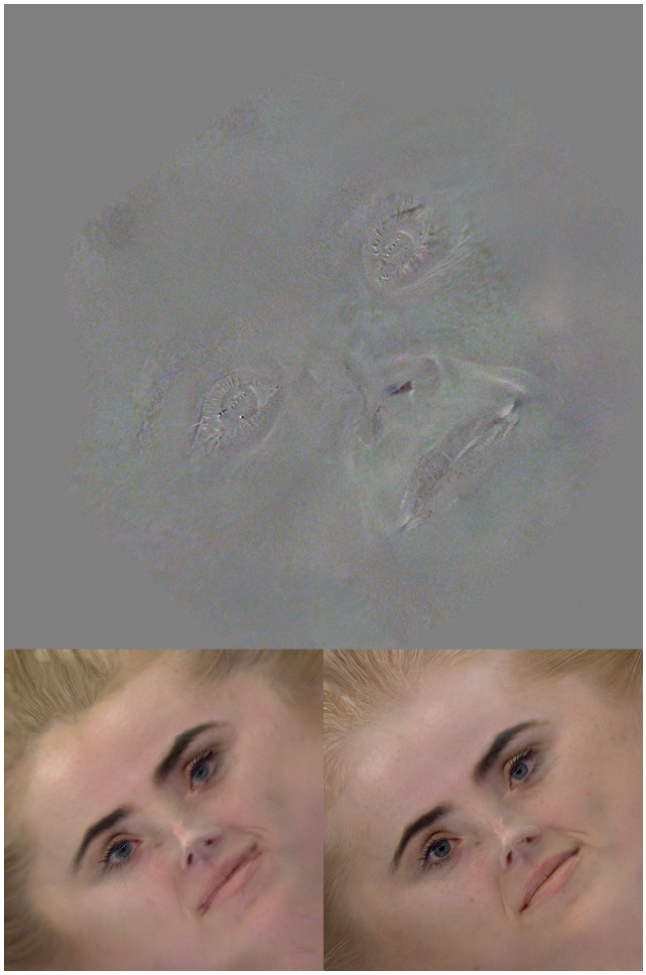

Figure 7: Impact of geometric warping. Bottom-left: 50% cross dissolve without geometric warp (artifacts around the lips and the eyes), bottom-right: with geometric warp compensation, top: Color differences after geometric image warp. No strong edges are visible around eyes and mouth.

with $\mathcal{R}$ being a regularization term weighted by a scalar factor λ. Similar to [16], we model the geometric image deformation of $\mathcal{T}_{last,t-1}$ with regard to $\mathcal{T}_{first,t}$ as a regular deforming 2D control mesh with Barycentric interpolation between vertex position, i.e. the warping function is parametrized by a vector Φ containing the control vertex displacements, and the regularization term is based on the mesh Laplacian.

Based on the estimated warp, the motion in the last frames of $\mathcal{T}_{\ldots,t-1}$ and the first of $\mathcal{T}_{\ldots,t}$ are deformed gradually to ensure that the transition frames of both sequences are identical. This deformation process is distributed over several frames. We use a rather high number of frames n=60 (at 59 fps) to perform the geometric deformation because the additional motion per frame should be as low as possible to make it barely noticeable.

5.2 Anisotropic Cross Dissolve

The geometric texture alignment reduces ghosting artifacts during blending (see figure 7). However, color differences between $\mathcal{T}_{last,t-1}$ and $\mathcal{T}_{first,t}$ can still exist. These can be caused by changing lighting conditions as the head usually moves during the capturing process, surface deformations (e.g. wrinkles that appear or disappear) and remaining misalignments that could not be fully compensated by the image warping (see figure 7). Though the remaining discrepancies are not disturbing in the still image, they become apparent when re-playing the texture sequences. Therefore, an additional cross dissolve blending is performed in parallel to the geometric deformation. The cross dissolve is also distributed over a large number of frames in order to achieve a slow and smooth transition. The number of frames has to be chosen carefully: if the number of frames used for the transition is too small, the resulting transition can become apparent due to sudden changes in shading or specularities. On the other hand, if the number of frames is too large, ghosting artifacts can appear because the cross dissolve adds high frequency details while the face deforms (e.g. specularities on the closed eye, lip line on a opened mouth, etc.).

Therefore, we use an anisotropic cross dissolve that allows for multiple blending speeds within the same texture. For example, a fast blending (e.g. the blending finishes after 4 frames) is used in regions with high frequency differences (e.g. eyes and mouth) whereas slow blending speed (e.g. the blending finishes after 40 frames) is applied in smooth regions with mainly low frequency differences (e.g. skin regions). The faster cross dissolve does not create disturbing effects because blending small misalignments with cross dissolve results in a sensation of movement [18]. This small but fast movement is barely noticeable in contrast to a slowly appearing or disappearing ghosting effect caused by an isotropic cross dissolve. The anisotropic cross dissolve is realized by providing an additional speed-up factor s for each texel. For this purpose, a static binary map $\mathcal{S}$ was used to mark regions of increased blending speed. To ensure a smooth spatial transition between regions of different blending speeds, $\mathcal{S}$ is blurred in order to create intermediate regions where s changes gradually from slow to fast. For our experiments, a single binary map was created manually in texture space

6. RESULTS AND DISCUSSION

Experimental Results and Discussion

This section presents still images of our proposed re-animation technique. Note that the results can best be evaluated in motion and we therefore also refer to the video in the supplementary material.

For our experiments, we recorded different facial performances of an actress with 4 calibrated UHD-Cameras (Sony F55) at 59 Hz. In order to capture mainly intensity changes in the texture that are induced by the facial expressions (e.g. wrinkles) we captured under homogeneous lighting conditions. We annotated 18 clips in the captured multi-view video footage and transformed them to dynamic texture sequences. Figure 8 shows an example of a reconstructed texture mosaic. A 3D head geometry proxy of the actress was created a-priorily using one shot of a multi-view still camera rig consisting of 14 D-SLR cameras (Canon EOS 550D) (see figure 3).

Figure 8: Example of a reconstructed head texture.

The presented approach was used to optimize the first/last n=60 frames of each texture sequence to allow for a seamless transition between the different texture sequences or to loop a single sequence multiple times (e.g. idle). A non-optimized implementation of our system tracks up to 4 frames per second, takes several seconds for one texture mosaic and approximately 10 seconds for the generation of an optimized texture sequence transition. This is sufficient as we consider the main purpose of our presented system as an offline processing tool (e.g. video editing or creation of video-based animations for digital characters in games).

To demonstrate the interactive usability of our re-animation approach, the visual quality of the resulting face sequences and the usage for free viewpoint rendering, we implemented a free viewpoint GUI, in which a user can arbitrarily change the order of facial performance sequences while at the same time changing the viewpoint (see figures 9 and 10 as well as the accompanying video). This demonstrates that our approach can be used as a video editing tool to seamlessly exchange or recompose the facial performance of an actor in post production or to conveniently create an optimized set of dynamic textures that allow for rendering photorealistic facial performances for virtual agents or digital characters (e.g. in computer games).

The use of a low-dimensional model results in a very stable tracking, which is important in order to generate realistic dynamic textures. Tracking errors would directly influence the visual quality of the synthesized facial performance because they result in an additional movement of the whole face when the textured model is rendered. The low dimensionality of the model is compensated for by applying warping and blending to the textures at the transition point when concatenating sequences with different facial expressions. This compensation in texture space is possible as long as the deformation can be described as an image warp. This describes the trade-off between geometric and textural changes: Geometric changes are needed for large-scale changes, e.g. viewpoint and global illumination changes, jaw movements and strong deformations, whereas textural changes are especially well suited in regions with small-scale, detailed and more subtle movements, e.g. fine wrinkles that form around the eyes or mouth. The geometric model should therefore be of low dimensionality to ensure a robust tracking

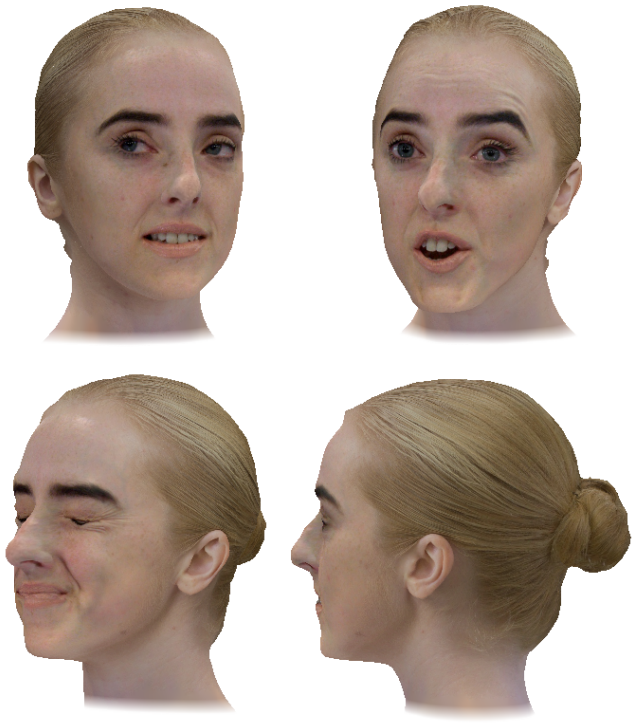

Figure 9: Different re-animated facial expressions and modified camera orientations demonstrating free viewpoint capabilities.

but have enough freedom to model large-scale changes that cannot be described as a textural warp.

The results show that by using real video footage to express subtle facial motion and details, highly realistic facial animation sequences of an actor can be achieved. In addition, our approach requires only little additional data: an approximation of the actor's head together with a calibrated video stream is sufficient to perform the presented facial re-animation technique. We aim at keeping the manual effort as low as possible, i.e. the user is only required once to select a few fiducial points in a single keyframe in order to initialize the head motion tracking. All subsequent trackings can then be initialized automatically using standard feature detection/matching (e.g. SIFT).

Lighting variations present a general limitation of image-based approaches as these variations are captured in the textures. To address this, we captured under homogeneous lighting conditions. Global lighting conditions can then still be modified during rendering using the approximate geometry as demonstrated for example by Bradley et al. [6]. While this generally follows the overall concept of our approach (i.e. global geometry and lighting changes can be modeled geometrically while subtle details are captured in the texture), the database could also be extended by different lighting conditions. Furthermore, some visual artifacts might arise from the use of a geometry that does not capture all shape details. While the simple geometric proxy could simulate the perspective distortions sufficiently well in our experiments, wide viewpoint changes in combination with

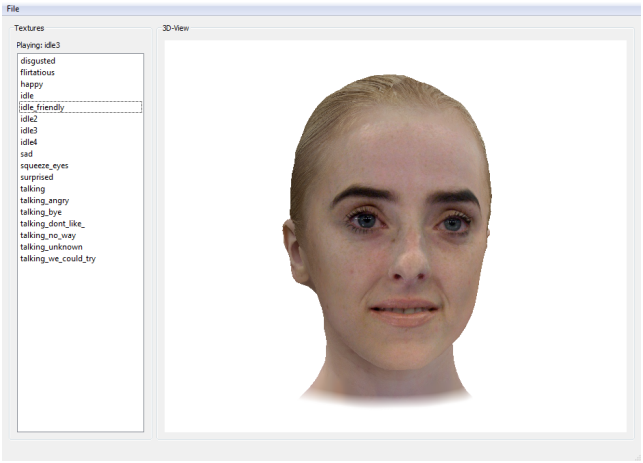

Figure 10: Screenshot of the implemented user interface for re-animation. Possible texture clips are displayed in the list box on the left side.

Figure 11: Example of artifacts caused by the roughly approximated geometry during a speech sequence: wrong silhouette (right), wrong projection of teeth and eyes (left and right)

strong deformations can reveal inaccuracies in the geometry (e.g. wrong projection of the texture or errors at the silhouette), see figure 11.

7. CONCLUSION

We presented a photorealistic method for video-based synthesis of facial performances. Our method does not require highly sophisticated performance capture hardware or complex statistical models which makes it a valuable option for low profile productions to create photorealistic facial re-animations based on captured video footage. The presented approach enables the animator to create novel facial performance sequences by simply providing a sequence of facial-expression-labels that describe the desired facial performance. This makes it easy to create novel facial videos, even for untrained users. Deformations caused by facial expressions are encapsulated in dynamic textures that are extracted from real video footage. New facial animation sequences are then synthesized by clever concatenation of dynamic textures rendered upon the geometry, instead of modeling all fine deformations in 3D. In order to create seamless transitions between consecutive sequences, we perform a geometric and photometric optimization of each sequence. Through the extraction of dynamic textures from real video footage and the definition of transition rules between independent textures, our approach combines the photorealism of real image data with the ability to modify or re-animate recorded performances. Possible applications of our approach range from video editing applications to the animation of digital characters.

Future Work

In our experiments, we manually selected transition points at the beginning and at the end of each dynamic texture. While this successfully demonstrates the visual quality of synthesized facial animations achieved with our approach, it would be desirable to directly switch from one sequence to another (e.g. starting to laugh while talking) without necessarily finishing the first one. This could be achieved for example by analyzing the facial video sequences in order to extract optimal transition points where it is possible to directly switch from one sequence to another, similar to a motion-graph-based approach. This would allow to create more complex animation graphs. Furthermore, we plan to add a better handling of different lighting situations. For example by supporting global light changes based on the approximate geometry or by capturing facial expressions under different lighting situations to extend the texture database. For animation, appropriate textures could then be selected based on the target expression and desired lighting conditions and it would be possible to blend between different light conditions during the animation. Another possible extension is the definition of local regions in texture space allowing for an independent animation of multiple face parts (e.g. eyes and mouth) at the same time.

Acknowledgment

This work is partially funded by the European Commission, H2020-644629 AutoPost.

8. REFERENCES

[1] O. Alexander, M. Rogers, W. Lambeth, M. Chiang, and P. Debevec. Creating a photoreal digital actor: The digital emily project. In *European Conference on Visual Media Production (CVMP)*, London, UK, 2009.

[2] Marie-Odile Berger. Realistic Face Animation From Sparse Stereo Meshes. In *International Conference on Auditory-Visual Speech Processing 2007 - AVSP 2007*, Hilvarenbeek, Netherlands, 2007.

[3] D. Blumenthal-Barby and P. Eisert. High-resolution depth for binocular image-based modelling. *Computers & Graphics*, 39:89–100, 2014.

[4] G. Borshukov, D. Piponi, O. Larsen, J. P. Lewis, and C. Tempelaar-Lietz. Universal capture - image-based facial animation for "the matrix reloaded". In *ACM SIGGRAPH Courses*, 2005.

[5] George Borshukov, Jefferson Montgomery, Witek Werner, Barry Ruff, James Lau, Paul Thuriot, Patrick Mooney, Stefan Van Niekerk, Dave Raposo, Jean-Luc Duprat, John Hable, Håkan Kihlström, Daniel Roizman, Kevin Noone, and Jeff O'Connell. Playable universal capture. In *ACM SIGGRAPH 2006 Sketches*, SIGGRAPH '06, New York, NY, USA, 2006. ACM.

[6] Derek Bradley, Wolfgang Heidrich, Tiberiu Popa, and Alla Sheffer. High resolution passive facial performance capture. In *ACM SIGGRAPH 2010 Papers*, SIGGRAPH '10, pages 41:1–41:10, New York, NY, USA, 2010. ACM.

[7] J. Carranza, C. Theobalt, M.A. Magnor, and H.-P. Seidel. Free-viewpoint video of human actors. In *ACM SIGGRAPH*, 2003.

[8] D. Casas, M. Tejera, J.-Y. Guillemaut, and A. Hilton. 4d parametric motion graphs for interactive animation. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, I3D '12, pages 103–110, New York, NY, USA, 2012. ACM.

[9] D. Casas, M. Volino, J. Collomosse, and A. Hilton. 4d video textures for interactive character appearance. *Computer Graphics Forum*, 33(2):371–380, 2014.

[10] J.-X. Chai, J. Xiao, and Jessica J. Hodgins. Vision-based control of 3d facial animation. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2003.

[11] Peter Eisert and Jürgen Rurainsky. Geometry-assisted image-based rendering for facial analysis and synthesis. *Sig. Proc.: Image Comm.*, 21(6):493–505, 2006.

[12] Tony Ezzat, Gadi Geiger, and Tomaso Poggio. Trainable videorealistic speech animation. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '02, pages 388–398, New York, NY, USA, 2002. ACM.

[13] P. Garrido, L. Valgaert, C. Wu, and C. Theobalt. Reconstructing detailed dynamic face geometry from monocular video. *ACM Transactions on Graphics*, 32(6):158:1–158:10, 2013.

[14] P. Garrido, L. Valgaert, C. Wu, and C. Theobalt. Reconstructing detailed dynamic face geometry from monocular video. *ACM Trans. Graph.*, 32(6):158:1–158:1F0, November 2013.

[15] P. Garrido, L. Valgaerts, H. Sarmadi, I. Steiner, K. Varanasi, P. Perez, and C. Theobalt. Vdub: Modifying face video of actors for plausible visual alignment to a dubbed audio track. In *Eurographics 2015*, pages –, 2015.

[16] A. Hilsmann and P. Eisert. Tracking deformable surfaces with optical flow in the presence of self-occlusions in monocular image sequences. In *CVPR Workshops, Workshop on Non-Rigid Shape Analysis and Deformable Image Alignment (NORDIA)*, pages 1–6. IEEE Computer Society, June 2008.

[17] P. Huang, A. Hilton, and J. Starck. Human motion synthesis from 3d video. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 1478–1485, June 2009.

[18] I. Kemelmacher-Shlizerman, E. Shechtman, R. Garg, and S. M. Seitz. Exploring photobios. *ACM Trans. Graph.*, 30(4):61:1–61:10, July 2011.

[19] J. Kilner, J. Starck, and A. Hilton. A comparative study of free-viewpoint video techniques for sports events. In *European Conference on Visual Media Production (CVMP)*, 2006.

[20] L. Kovar, M. Gleicher, and F. Pighin. Motion graphs. In *Proc. of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '02, pages 473–482, New York, NY, USA, 2002. ACM.

[21] K. Li, Q. Dai, R. Wang, Y. Liu, F. Xu, and J. Wang. A data-driven approach for facial expression retargeting in video. *IEEE Transactions on Multimedia*, 16:299–310, 2014.

[22] C. Lipski, F. Klose, K. Ruhl, and M. Magnor. Making of who cares hd stereoscopic free viewpoint video. In *European Conference on Visual Media Production (CVMP)*, 2011.

[23] W. Paier, M. Kettern, and P. Eisert. Realistic retargeting of facial video. In *Proc. of the 11th European Conference on Visual Media Production*, CVMP '14, pages 2:1–2:10, New York, NY, USA, 2014. ACM.

[24] P. Pérez, M. Gangnet, and A. Blake. Poisson image editing. *ACM Trans. Graph.*, 22(3):313–318, July 2003.

[25] F. Pighin, J. Hecker, D. Lischinski, R. Szeliski, and D. H. Salesin. Synthesizing realistic facial expressions from photographs. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '98, pages 75–84, New York, NY, USA, 1998. ACM.

[26] F. Pighin and J. Lewis. Facial motion retargeting. In *ACM SIGGRAPH Courses*, 2006.

[27] E. Reinhard, M. Ashikhmin, B. Gooch, and B. Shirley. Color transfer between images. *IEEE Computer Graphics Applications*, 21(5):34–41, 2001.

[28] J. M. Saragih, S. Lucey, and J. F. Cohn. Deformable model fitting by regularized landmark mean-shift. *Int. J. Comput. Vision*, 91(2):200–215, January 2011.

[29] A. Schödl and I. A. Essa. Controlled animation of video sprites. In *SCA '02: Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 121–127, New York, NY, USA, 2002. ACM.

[30] D. Sibbing, M. Habbecke, and L. Kobbelt. Markerless reconstruction and synthesis of dynamic facial expressions. *Computer Vision and Image Understanding*, 115(5):668–680, 2011.

[31] T. Weise, H. Li, L. Van Gool, and M. Pauly. Face/off: Live facial puppetry. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '09, pages 7–16, New York, NY, USA, 2009. ACM.

[32] F. Xu, Y. Liu, C. Stoll, J. Tompkin, G. Bharaj, Q. Dai, H.-P. Seidel, J. Kautz, and C. Theobalt. Video-based characters: Creating new human performances from a multi-view video database. In *ACM SIGGRAPH 2011 Papers*, SIGGRAPH '11, pages 32:1–32:10, New York, NY, USA, 2011. ACM.