IET Computer Vision

Special Section: Computer Vision for the Creative Industries

IET Journals
The Institution of Engineering and Technology

Interactive facial animation with deep neural networks

ISSN 1751-9632
Received on 6th October 2019
Revised 2nd April 2020
Accepted on 3rd June 2020
E-First on 13th August 2020
doi: 10.1049/iet-cvi.2019.0790
www.ietdl.org

Wolfgang Paier[1] ✉, *Anna Hilsmann*[1], *Peter Eisert*[1,2]
[1]*Vision & Imaging Technologies, Fraunhofer HHI, Berlin, Germany*
[2]*Visual Computing, Humboldt University, Berlin, Germany*
✉ *E-mail: wolfgang.paier@hhi.fraunhofer.de*

Abstract: Creating realistic animations of human faces is still a challenging task in computer graphics. While computer graphics (CG) models capture much variability in a small parameter vector, they usually do not meet the necessary visual quality. This is due to the fact, that geometry-based animation often does not allow fine-grained deformations and fails in difficult areas (mouth, eyes) to produce realistic renderings. Image-based animation techniques avoid these problems by using dynamic textures that capture details and small movements that are not explained by geometry. This comes at the cost of high-memory requirements and limited flexibility in terms of animation because dynamic texture sequences need to be concatenated seamlessly, which is not always possible and prone to visual artefacts. In this study, the authors present a new hybrid animation framework that exploits recent advances in deep learning to provide an interactive animation engine that can be used via a simple and intuitive visualisation for facial expression editing. The authors describe an automatic pipeline to generate training sequences that consist of dynamic textures plus sequences of consistent three-dimensional face models. Based on this data, they train a variational autoencoder to learn a low-dimensional latent space of facial expressions that is used for interactive facial animation.

1 Introduction

Modelling and rendering of virtual human faces is still a challenging task in computer graphics. Complex deformation and material properties and effects like occlusion and disocclusions make it difficult to develop efficient representations that are capable of generating high-quality renderings. Especially frameworks, which try to capture eyes and the oral cavity in a holistic manner suffer from these problems. A geometry-based approach would have to model eyeballs and eyelids, as well as tongue, teeth and gums explicitly, which is prone to artefacts, low-visual quality and likely to require heavy manual intervention. On the other hand, purely image-based approaches are becoming more and more popular with the rise of deep convolutional networks, which are capable of capturing the complex appearance changes in image space alone. However, this comes at the cost of long training times, high hardware requirements and large amounts of training data. Futhermore, simple operations like changing the three-dimensional (3D) pose of a model become more difficult again. A more sensible solution should make use of both approaches. Geometry-based representations can already compensate for a large amount of variance, which is usually caused by rigid motion and large-scale deformations, but they fail in face regions, which exhibit complex as well as fine motions or strong changes in texture. Therefore, we propose a hybrid approach between classical geometry-based modelling and image-based rendering using dynamic textures (Fig. 1). In general, working in texture space provides several advantages, like a fixed resolution, less variance that is induced by motion and especially a fixed topology that even allows us to define regions of interest for generating local models that can be animated independently (e.g. mouth or eyes). This helps in reducing the model complexity by a large amount, as we can train smaller neural networks on selected regions. In our experiments, we focus on the eye and mouth area, which are represented by convolutional neural networks as local non-linear models that are able to capture the highly non-linear appearance changes. Our networks have a variational autoencoder (VAE) structure with two separate encoding and decoding paths for texture and geometry. Additionally, we use an adversarial loss, to improve the visual quality of the generated textures such that even fine details are visible. We designed our system in a way such that it can generate a 2D expression manifold for each region of interest (i.e. mouth and eyes). This enables us to create a simple interface for interactive facial animation in real time (Fig. 2).

The remainder of this paper is organised as follows: in Section 2, we present an overview of existing technologies and methods that are related to facial animation and facial modelling. Section 3 gives an overview of the proposed framework. In Section 4, we describe the data acquisition process for our training and test data. Section 5 contains a detailed description of our hybrid model and in the last two sections, we present and discuss our results.

2 Related work

Animation and performance capture of human faces have been an active area of research for decades.

Classical approaches for modelling facial expressions, for example [1–3], have been used for many years. They are usually built upon a linear basis of blendshapes and even though their expressive power is limited, they became popular due to their robustness and simplicity of use.

For example in [4], Li *et al.* improve blendshape-based performance capture and animation systems by estimating corrective blendshapes on the fly. Cao *et al.* [5] present an automatic system for creating large-scale blendshape databases using captured RGB-D data and Garrido *et al.* [6] use blendshapes to create a personalised model by adapting it to a detailed 3D scan using sparse facial features and optical flow. The generated face model is then used to synthesise plausible mouth animations of an actor in a video according to a new target audio track. Since blendshape models cannot capture teeth and the mouth cavity, they use a textured 3D proxy to render teeth and the inner mouth. A more sophisticated facial retargeting system was proposed by Thies *et al.* [7] who implemented a linear model that can represent expressions, identity, texture and light. With their model, it is possible to capture a wide range of facial expressions of different people under varying light conditions. However, similar to Garrido *et al.* [6], they had to use a hand-crafted model to visualise the oral cavity, which also demonstrates the weakness of blendshape-based models. While being useful to represent large-scale geometry

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

deformations and rigid motion, they usually lack expressive power to capture complex deformations and occlusions/disocclusions that occur around eyes, mouth and in the oral cavity.

To circumvent these limitations, several image-based approaches like [8–17] have been developed. For example in [13], a low-cost system for facial performance transfer is presented. Using a multilinear face model, they are able to track an actor's facial geometry in the 2D video, which enables them to transfer facial performances between different persons via image-based rendering. Similar techniques were used in [12, 9, 7]. A different way of image-based animation was proposed by Casas *et al.* in [11] or Paier *et al.* in [8, 10]. They exploit the fact that low-quality 3D models with high-quality textures are still capable of producing high-quality renderings. Based on previously captured video and geometry, they create a database of atomic geometry and texture sequences that can be re-arranged to synthesise novel performances. While Casas and co-workers use pre-computed motion graphs to find suitable transitions points, Paier and co-workers proposed a spatio-temporal blending to generate seamless transitions between concatenated sequences.

The big advantage of these approaches is that they make direct use of captured video data to synthesise new video sequences, which results in high-quality renderings. Geometry is only used as a proxy to explain rigid motion, large-scale deformation and perspective distortion. However, the high-visual quality comes at the cost of high-memory requirements during rendering, since textures have to be stored for each captured frame in the database.

With the advent of deep neural networks, more powerful generative models have been developed. Especially, VAEs [18] and generative adversarial networks (GANs) [19] gain popularity, since they are powerful enough to represent large distributions of high-dimensional data in high quality (e.g. faces images or textures). For example in [20–22], GANs are used to learn latent representations of images that can be used to manipulate the content of images (e.g. facial expression) in a controlled and meaningful way. More recent methods like [23, 24] are even capable of generating believable facial animations from a single photo using GANs. For example, Pumarola *et al.* [23] present a GAN-based animation approach that can generate facial expressions based on an action unit coding scheme. The action unit weights define a continuous representation of the facial expressions. The loss function is defined by a critic network that evaluates the realism of generated images and fulfilment of the expression conditioning. Another example is [25], where Huang and Khan propose a method to synthesise believable facial expressions using GANs that are conditioned on sketches of facial expressions.

While deep models were initially used to generate realistic images or modify them, GANs are also capable of performing image to image translation tasks, as demonstrated in [26, 27]. In these examples, the output images of GANs are conditioned on source images, which usually represent the 'content' and the task of the GAN is generating new images that show the same content (e.g. facial expressions) but with a different style or appearance (e.g. identity). After training, it can be used to animate, for example, a virtual character or a face by recording the desired facial expressions and using the GAN to translate the video.

A different method was presented by Lombardi *et al.* [28] and Slossberg *et al*. [29]. Instead of working in image space alone, they use a 3D model with texture. A deep neural network is trained to represent geometry and texture with a 128-dimensional vector. They use a high-quality face capture rig with 40 synchronised machine vision cameras to reconstruct consistent 3D face geometry for every frame. Based on the 3D geometry and the average texture, they train a VAE that is capable of synthesising geometry and a view-dependent texture, which allows rendering high-quality face sequences in 3D. They also proposed a facial expression editing method, where they allow artistic control over facial expressions by applying position constraints on vertices. While this seems to be a sensible solution at first glance, our system aims at providing high-level control over captured facial expressions to design new facial animation sequences, rather than editing facial expressions on vertex level. Furthermore, we want to provide a solution that can work with standard hardware.

3 System overview

This section gives a high-level overview of the proposed animation framework, see Fig. 3. Our system basically consists of three phases: capture/tracking (offline), training (offline) and animation/rendering(interactive/live).

In the first phase, we capture a set of multiview stereo image pairs in order to compute an approximate deformable geometry model of the face. We need not capture all fine details in geometry since most of the visual information will be represented by dynamic textures. This is especially true for complex areas like mouth and eyes, where accurate 3D modelling would need heavy manual intervention. Our deformable face model is only responsible to capture rigid motion, large-scale deformation and provide consistent texture space parameterisation. The advantage of this approach is that we can do this in an automatic way without manual intervention. Since most visual information will be stored in texture space, we also need to create a texture model. We do this by recording a few sequences of multiview video, where our actress shows some general facial expressions and speaks a set of words in order to capture most visemes and transitions between them. Before we start extracting face textures, we estimate the 3D face pose and geometry deformation parameters for each captured image frame. In the second phase, we train a deep auto-encoder network to learn a latent representation of face texture and geometry. In the last phase, we use this latent representation to interpolate realistically between different facial expressions (in geometry as well as in texture space), which allows us to create new performances by concatenating short sequences of captured geometry/texture in a motion-graph like a manner. Furthermore, we provide a simple user interface for direct high-level manipulation of facial expressions based on the latent expression features. Due to the fast evaluation of our neural model on the GPU, we can provide a graphical user interface for interactive animation and facial expression editing in the real time.

4 Acquisition of data

Our proposed face representation is based on a deformable 3D geometry model with consistent topology and dynamic textures. While the geometry acts as a 3D proxy to represent rigid motion and large-scale deformation of the face, our texture model aims at

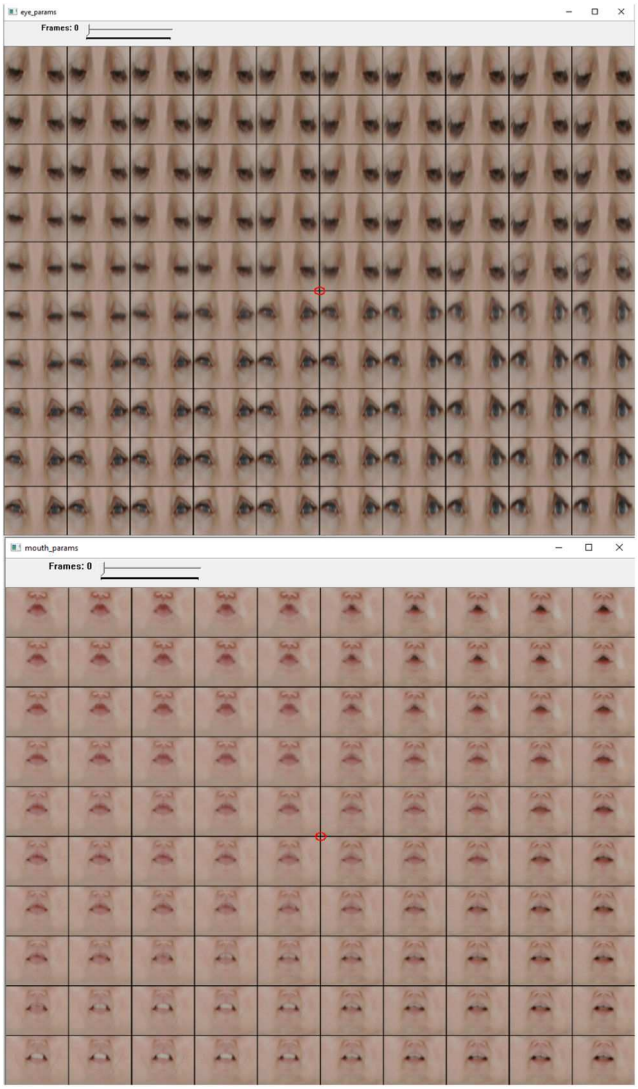

Fig. 2 *Screenshot of our implemented animation interface. There are two separate windows for mouth (bottom) and eye (top) control. We use the 2D parameter vector for mouth and eyes to display the learned manifold of facial expressions. By clicking on the displayed expression in the respective window it is possible to change the facial expression of our model*

generating high-quality dynamic textures with fine details and realistic micromovements. The capture process for data acquisition consists of two parts: first, we capture 18 general facial expressions as well as 15 expressions that show visemes, with a D-SLR camera rig that consists of four stereo pairs (Fig. 4). Second, we record multiview face video with high-resolution machine vision cameras in a diffuse light setup. For this part, we capture video data for the dynamic texture model. Similar as before, we capture one video, where our actress displays general facial expressions and two more videos, where our actress speaks a set of selected words to capture visemes and transitions between different visemes. In the last video, our actress displayed some general facial expressions and speech which allows us to evaluate the performance on unseen data. The training data consists of ~2160 frames and the evaluation sequence has 250 frames.

4.1 Preprocessing

During preprocessing, we compute the 3D face geometry for each static facial expression that was captured by the D-SLR rig. This is achieved using a three-stage reconstruction pipeline: first, we extract and match SIFT [30] features using the matching approach described in [31]. Knowing corresponding points in all camera pairs, we compute an initial 3D point cloud (Fig. 5, leftmost), which serves as input for the screened Poisson mesh reconstruction approach [32] to generate an initial 3D mesh (Fig. 5, middle). We

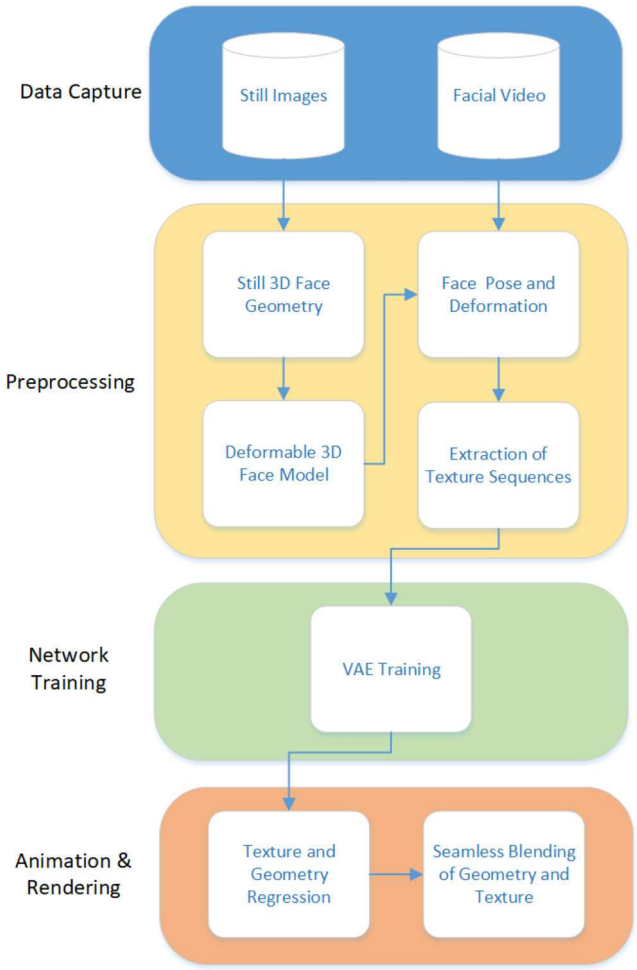

Fig. 3 *Schematic overview of the proposed framework*

refine the initial 3D mesh by computing dense stereo reconstructions [33] based on optical flow and improve the accuracy of the initial mesh by computing vertex offsets that minimise the difference between a vertex on the initial mesh and the closest vertices on all dense stereo reconstructions. We regularise the deformation field by minimising the uniform Laplacian of the deformation field. The result of this first step is shown in Fig. 5 (right).

4.2 Deformable mesh registration

In order to create an animatable face model, we need registered face meshes with consistent topology. For this purpose, we use an existing template mesh [34] and deform it such that it matches the geometry of all reconstructed 3D face meshes. During mesh registration, we minimise three error terms simultaneously: the point-to-plane distance between template geometry and target geometry $\mathscr{E}_{\text{geo}}$ (2), the optical flow $\mathscr{E}_{\text{img}}$ (3) and the distance between automatically detected facial feature points [35] and the projected 2D location of corresponding 3D mesh-landmarks on the template mesh $\mathscr{E}_{\text{lm}}$ (4). We optimise the following parameters during mesh registration: similarity $\mathcal{S}$ is represented by a 7D parameter vector $\boldsymbol{s} = [t_x, t_y, t_z, r_x, r_y, r_z, s]^{\text{T}}$ that contains three parameters for translation (t_x, t_y, t_z), three parameters for rotation (r_x, r_y, r_z) and s for uniform scale. The blendshape weight vector $\boldsymbol{b} = [b_0, \ldots b_9]^{\text{T}}$ is a 10D column vector and vertex offsets $\boldsymbol{o} = [o_x, o_y, o_z, \ldots, o_x^n, o_y^n, o_z^n]^{\text{T}}$ represent a 3D offset for each vertex in the template mesh with n vertices

$$\boldsymbol{x}_{\boldsymbol{s},\boldsymbol{b},\boldsymbol{v}} = \mathcal{S}(\mathcal{X}_0 + \mathcal{B}\boldsymbol{b} + \boldsymbol{o}) \tag{1}$$

Fig. 4 *Samples of static expressions captured by D-SLRs*

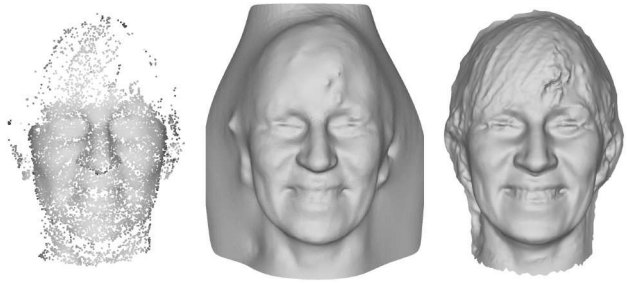

Fig. 5 *Results of the mesh reconstruction, from left to right: initial point cloud, initial 3D mesh, refined and cleaned 3D mesh*

The column vector $\boldsymbol{x}_{\boldsymbol{s},\boldsymbol{b},\boldsymbol{v}} = [x, y, z, \ldots, x_n, y_n, z_n]^{\mathrm{T}}$ contains x, y and z coordinates of all vertices of the deformed mesh. $\mathcal{X}_0$ corresponds to the mean shape of the mesh template and $\mathcal{S}$ represents the similarity transform that is applied in the last step on all deformed and refined vertices. $\boldsymbol{x}_i$ refers to the 3D position $[x_i, y_i, z_i]^{\mathrm{T}}$ of the ith transformed vertex. $\mathcal{B}$ refers to the blendshape base (column vectors) of the mesh template to adapt the overall shape

$$\mathcal{E}_{\mathrm{geo}}(\boldsymbol{s},\boldsymbol{b},\boldsymbol{v}) = \sum_i \left| (\boldsymbol{x}_i - \boldsymbol{v}_i)\boldsymbol{n}_{v_i} \right|^2 \tag{2}$$

$\mathcal{E}_{\mathrm{geo}}$ refers to the squared point-to-plane distance, with $\boldsymbol{x}_i$ being the transformed position of vertex i, v_i being the corresponding point on the target mesh and $\boldsymbol{n}_{v_i}$ being the surface normal of $\boldsymbol{v}_i$. We select $\boldsymbol{v}_i$ by choosing the closest point on the target mesh using a *kd*-tree

$$\mathcal{E}_{\mathrm{img}}(\boldsymbol{s},\boldsymbol{b},\boldsymbol{v}) = \sum_c^C \sum_{p \in I} \left| \mathcal{I}_c(p) - \mathcal{I}_{\boldsymbol{s},\boldsymbol{b},\boldsymbol{v}}(p) \right|^2, \tag{3}$$

where $\mathcal{E}_{\mathrm{img}}$ refers to the sum of all squared intensity differences at all rendered pixel p between the synthesised image $\mathcal{I}_{\boldsymbol{s},\boldsymbol{b},\boldsymbol{v}}$ and the captured image $\mathcal{I}_c$, with c being the camera index and C being the number of all cameras

$$\mathcal{E}_{\mathrm{lm}}(\boldsymbol{s},\boldsymbol{b},\boldsymbol{v}) = \sum_c^C \sum_i \left| \boldsymbol{m}_{c,i} - \hat{\boldsymbol{m}}_{c,i} \right|^2, \tag{4}$$

the above equation refers to the sum of squared distances between the projected 2D position $\boldsymbol{m}_{\boldsymbol{c},\boldsymbol{i}}$ of all mesh landmarks and the corresponding detected 2D landmark position $\hat{\boldsymbol{m}}_{\boldsymbol{c},\boldsymbol{i}}$ in all input images. We further assume that the mesh template is annotated and the locations of distinct facial features (e.g. eyes, nose, mouth corners) on the mesh are available (i.e. barycentric coordinates and triangle id of landmarks on the triangle mesh).

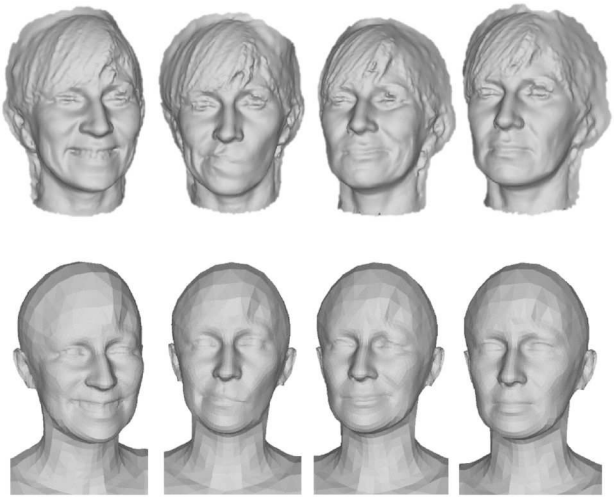

Fig. 6 *Examples of the adapted 3D face template with corresponding multiview 3D reconstructions*

The mesh registration process itself consists of two steps: first, the mesh template is registered with the reconstructed neutral face geometry. For this step, we minimise geometry error $\mathcal{E}_{\mathrm{geo}}$ and the distance between corresponding landmark positions $\mathcal{E}_{\mathrm{lm}}$ by adjusting similarity $\boldsymbol{s}$, blendshapes weights $\boldsymbol{b}$ and vertex offsets $\boldsymbol{o}$. The blendshape weight $\boldsymbol{b}$ is related to the SMPL [34] template model, which provides ten blendshapes to adjust the general shape. Fine adjustments to the face shape are realised by computing per-vertex offset $\boldsymbol{o}$ that is regularised via the contangent Laplacian δ_i [36, 37]

$$\delta_i = \sum_{\{i,j\} \in E} w_{i,j}(\boldsymbol{x}_j - \boldsymbol{x}_i), \tag{5}$$

with x_i being the position of vertex i and $\boldsymbol{x}_j$ being the position of one of its one-ring neighbours

$$w_{i,j} = \frac{\omega_{i,j}}{\sum_{\{i,k\} \in E} \omega_{i,k}} \tag{6}$$

$$\omega_{i,j} = \cot\alpha + \cot\beta \tag{7}$$

After the template mesh is adapted to the neutral expression, we start registering the template to all other facial expressions. We do this by deforming the adapted neutral template to match the reconstructed geometry of all other facial expressions. During this phase, we only adapt rigid motion $(t_x, t_y, t_z, r_x, r_y, r_z)$ and vertex offsets $\boldsymbol{o}$, since we have no blendshapes that could simulate facial expressions. With the adapted neutral template, we additionally use the optical flow error (3) to improve the semantic consistency

between the registered face meshes. To regularise the strong deformations, we minimise the contangent mesh Laplacian (5) on vertex offsets as well as on the vertex positions. This ensures a smooth deformation field as well as an artefact-free and smooth mesh surface. The registration process is treated as non-linear optimisation task, which is solved using the iterative Gauss–Netwon method. With the 33 registered template meshes, we create a linear face deformation model consisting of 20 basis vectors, which are computed using principal component analysis (PCA). This linear model is used in all subsequent steps for capturing face motion, deformation and extracting textures from the captured multiview video stream Fig. 6.

4.3 Model-based face tracking and texture extraction

The training data for our hybrid model is extracted from the previously recorded multiview video stream and consists of a deformable geometry proxy and high-resolution dynamic textures. We process the multiview video data in two steps: first, we track the 3D head pose and deformation based on face landmarks and optical flow. Knowing the approximate 3D head geometry at every timestep, we extract a sequence of textures using an approach [10] that was adapted for the usage in our pipeline. To obtain the 3D face geometry for the first frame of each sequence, we assume a neutral expression and optimise rigid motion parameters $(r_x, r_y, r_z, t_x, t_y, t_z)$ to minimise the distance between all detected landmarks and the projected 2D landmark positions (4) of the annotated face model. All following frames are processed by minimising landmark error $\mathscr{E}_{\text{lm}}$ as well as the intensity difference $\mathscr{E}_{\text{img}}$ between captured images and the rendered face model. To evaluate the intensity difference, we use the first frame of each camera as a texture for our face model and compute the difference between the synthesised image (given the pose and blendshape parameters) and the captured target image. As before, we treat the tracking as optimisation task that is solved iteratively using the Gauss–Newton method.

Knowing the accurate pose and face shape for each frame in the captured multiview videos allows us to extract a dynamic texture sequence for each video. The extracted texture sequences, capture all information that is not represented by geometry, which includes, for example, fine motions and deformations (e.g. micromovements, wrinkles) as well as changes in texture and occlusions/disocclusions (e.g. eyes and oral cavity). Together, geometry and texture represent almost the full appearance of the captured face in each frame. For the texture extraction, we rely on a graph-cut-based approach with a temporal consistency constraint [10]. This method, simultaneously optimises three data terms (8) to generate a visually pleasing (i.e. no spatial or temporal artefacts) sequence of textures from each video

$$\mathrm{E}_{\text{tex}}(C) = \sum_{t}^{T}\sum_{i}^{N} \mathscr{D}\left(f_i^t, c_i^t\right) + \lambda \sum_{i,j \in \mathscr{N}} \mathscr{V}_{i,j}\left(c_i^t, c_j^t\right) + \eta \mathscr{T}\left(c_i^t, c_i^{t-1}\right) \tag{8}$$

C denotes the set of source camera ids for all triangles. The first term $\mathscr{D}(f_i, c_i)$ is a measure for the visual quality of a triangle f_i textured by the camera c_i. It uses the heuristic $\mathscr{W}(f_i, c_i)$, which is the area of f_i projected on the image plane of the 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

$$\mathscr{D}(f_i, c_i) = \begin{cases} 1 - \mathscr{W}(f_i, c_i) & f_i \text{is visible} \\ \infty & f_i \text{is occluded} \end{cases} \tag{9}$$

$$\mathscr{W}(f_i, c_i) = \frac{\text{area}(f_i, c_i)}{\sum_{c_j} \text{area}(f_i, c_i)} \tag{10}$$

Fig. 7 *Upper half of this figure shows the rendered model with and without highlighted animation regions. The lower half of this picture shows that we can simply select the modified face region in texture space by defining a rectangular region of interest. Similarly, we select the corresponding vertices of interest by comparing the texture coordinate of each vertex with the region of interest in texture space*

$\mathscr{V}_{i,j}(c_i, c_j)$ represents a spatial smoothness constraint, which relates the sum of colour differences along the common edge $e_{i,j}$ of two triangles f_i and f_j that are textured from different cameras c_i and c_j

$$\mathscr{V}_{i,j}(c_i, c_j) = \begin{cases} 0 & c_i = c_j \\ \Pi_{e_{i,j}} & c_i \neq c_j \end{cases} \tag{11}$$

$$\Pi_{e_{i,j}} = \int_{e_{i,j}} \| I_{c_i}(x) - I_{c_j}(x) \| \mathrm{d}x \tag{12}$$

the last term $\mathscr{T}(c_i, c_j)$ in (8) ensures temporal smoothness by penalising the change of the source camera c_i of a triangle f_i

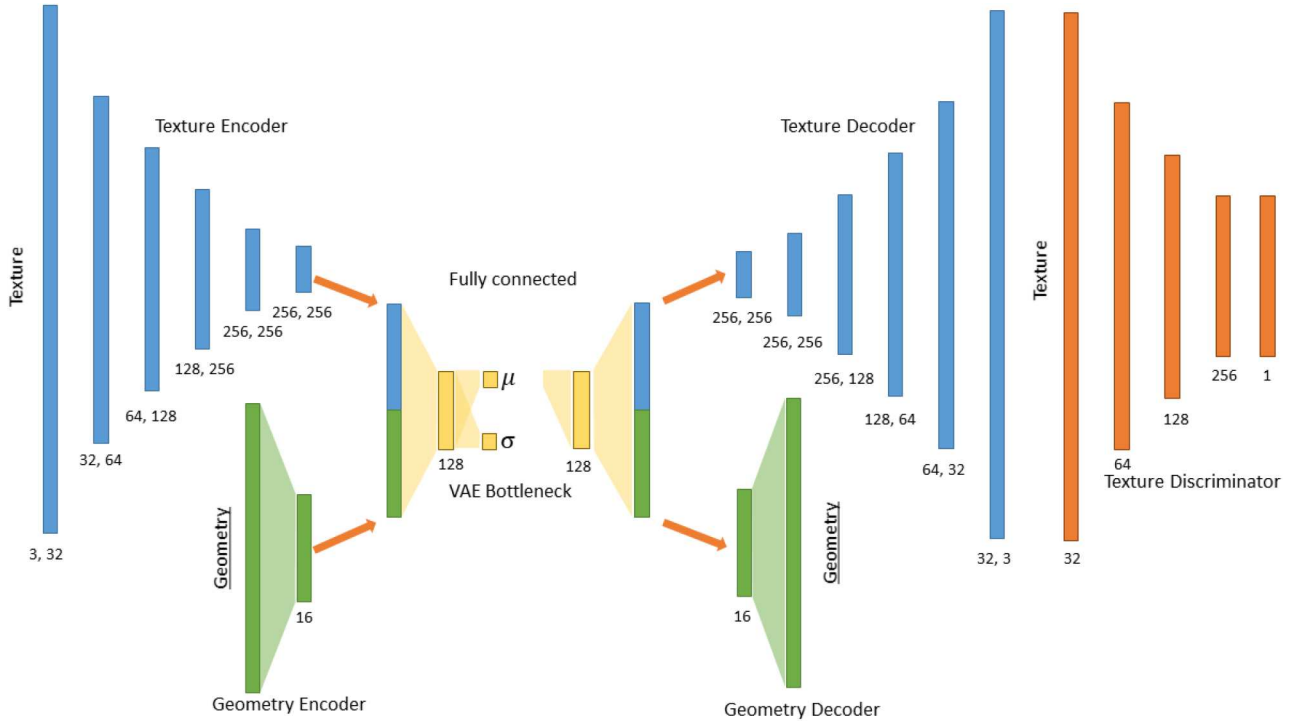

Fig. 8 *This figure shows the used network architecture. The network consists of roughly six parts. A convolutional texture encoder/decoder(blue). A geometry encoder/decoder(green). A fully connected bottleneck (yellow), that combines information of texture and geometry into a latent code vector (μ) and deviation (σ). A texture discriminator network (orange) tries to classify textures as real or synthetic*

Table 1 Final errors after 100 epochs

Data	Training error	Validation error
eye geometry	0.000274	0.000280
mouth geometry	0.000676	0.000759
eye texture	0.015	0.0163
mouth texture	0.0116	0.0125

between consecutive time steps. Without such a term, the extracted dynamic textures are not temporally consistent, i.e. the source camera of a certain triangle can change arbitrarily between two consecutive texture frames, which results in disturbing flickering in the resulting texture sequence

$$\mathcal{T}(c_i^t, c_i^{t-t}) = \begin{cases} 0 & c_i^t = c_i^{t-t} \\ 1 & c_i^t \neq c_i^{t-t} \end{cases} \tag{13}$$

The selection of source cameras C for all mesh-triangles in all frames is solved simultaneously using the alpha expansion algorithm [38]. In the last step, we assemble the reconstructed textures by sampling texel colours from the optimal source cameras. Seams between triangles are concealed using Poisson image editing [39].

5 Neural face model

Our hybrid face model consists of two animatable regions (i.e. mouth and eyes), while the rest of the face model is deformed automatically in order to ensure a consistent facial geometry. Both animatable regions are represented with deep neural networks that are capable of simultaneously generating a detailed local texture as well as the vertex positions of the local geometry proxy. The remaining face geometry is modelled with a small set of blendshapes and a static texture. This approach has several advantages: first, most relevant information appears around mouth and eyes. Decoupling them gives our model a higher expressive power while keeping the overall complexity and the amount of necessary training data as low as possible. The complexity of deformation and material properties of these two regions is much higher, which makes the usage of deep neural networks a sensible choice, while the remaining facial regions can be represented easily with a linear model. Furthermore, training smaller networks requires less data, less hardware and less time which makes this approach even more appealing in practice.

5.1 Network structure and training

Fig. 7 shows a schematic representation of the face model components. Both local non-linear models are represented as deep VAE [18] (Fig. 8). Similar to [28] we use separate encoder/decoder paths for texture (Fig. 8, blue) and geometry (Fig. 8, green) that are joined at the bottleneck (Fig. 8, yellow) into a single fully connected layer that outputs the latent representation of an expression. In contrast to [28], we use an adversarial discriminator (Fig. 8, orange) on the reconstructed texture to improve the realism of the reconstructed textures (e.g. eye-lashes, wrinkles).

The encoder–decoder path for the local texture model consists of six convolutional blocks. Each texture encoding block consists of two consecutive convolution–batchnorm–LeakyReLU layers, where the first layer performs downscaling using a stride of two. The first convolution block outputs a filter map with 32 channels. In each following convolution block, the number of channels is doubled. Each texture decoding block consists of a transposed convolution–batchnorm–LeakyReLU followed by a convolution–batchnorm–LeakyReLU layer. The number of channels for each filter bank can be found in Fig. 8. The vertex encoding and decoding layer consist of a linear layer followed by batchnorm and a LeakyReLU activation. The fully connected layers are merging texture and geometry information and output a 2D latent code that is regularised by the variational lower bound [18]. This guarantees a smooth latent space and latent features that are normally distributed ($\mu = 0$ and $\sigma = 1$), which give a well-defined parameter space that can be used for manual facial animation, interpolation

Fig. 9 *Possible facial expressions, reconstructed by our model. We randomly combined different expressions of eye and mouth area*

and editing. That means, our face model can be animated by providing 2D parameters for eyes and mouth region, which results in four parameters per facial expression. Our discriminator network is based on the PatchGAN structure as described in [40]. It consists of four convolutional blocks. Each block consists of a 2D convolution with stride 2, batchnorm and LeakyReLU layer. The final layer is a 1×1 convolution that outputs a single channel feature map where a value close to zero means, that the discriminator believes that this image region is fake, while a value close to one signals that the image patch is real. The advantage of using smaller patches is that the discriminator classifies based on smaller features and details, which forces the texture generator to create a sharper texture with more details. We use a batch-size of four, a leakiness of 0.2 for all ReLUs in our network and use the Adam optimiser with default parameters to train our network for 100 epochs with a fixed learning rate of 0.0001. Table 1 shows the final training errors after 100 epochs on training data and unseen data. The geometry error is the mean absolute deviation over all

Fig. 10 *Selected facial expressions that show the advantages of our hybrid approach. Below every facial expression, we show the cropped and enlarged mouth and eye area. While the geometry model cannot capture details and occlusions/disocclusions, we show that the texture model is capable of capturing and reproducing teeth, tongue, eye-lashes or wrinkles*

vertex coordinates given in metre. The texture error is given as the mean absolute error over all pixel, with normalised values that range from 0 to 1. While the texture error is quite similar for the eye and the mouth region, the geometry error for the eye region is almost three times lower than for the mouth region. This is most probably caused by the fact, that the eye geometry does not vary as much as the mouth geometry and therefore expressions for the eye region are mainly captured in texture space. This would also explain, why the texture error for the eye region is higher than for the mouth region.

5.2 Animation and rendering

The rendering of our face model can be performed with most available rendering engines since the output of our model is a textured triangle mesh. The animation procedure is performed in three steps. First, we evaluate both local face models with their corresponding animation parameters. These animation parameters can be generated either by using the encoder part of the trained networks or by a user that modifies the animation parameters manually by selecting the desired facial expressions on the implemented animation interface (Fig. 2). In order to show possible facial expressions in the animation control windows, we sample the expression parameter space at regular grid locations with coordinate-values between −4 and +4. The textures of these samples are then arranged as a 2D grid and act as a rough preview for the animator. Second, we make sure that the reconstructed texture and the geometry fit seamlessly into the animated mesh. We perform this integration with Poisson image and mesh editing. Since the reconstructed texture lies already in GPU memory, we use the highly optimised 2D convolution operations of the machine learning framework to perform realtime Poisson image editing on the texture. This can be implemented by creating a single 2D convolution layer with a $3 \times 3 \times 3 \times 3$ filter tensor, where each i,i, 3×3 sub-tensor contains the weights of a Laplacian filter kernel while all other elements are set to zero. Using this convolution operation, we compute the target response tensor L_t by convolving the generated texture. Then, we replace all border pixel values in the reconstructed texture with its values from the original texture and iteratively update the reconstructed texture by minimising the difference between the L_t and L. We use 25 blending iterations in our experiments. The integration of both sub-meshes is performed on the CPU. Therefore, we adjust the blendshape weights of the underlying face model to minimise the geometry difference between the full face model and the two local geometry reconstructions. Similar as before, we integrate the reconstructed geometry by performing Laplacian blending on all unmodified vertices while the updated positions of the mouth and eye region stay constant and serve as border constraint. Finally, we render the textured mesh with OpenGL.

6 Experimental results and discussion

In this section, we present more results that were generated with the proposed model. In Fig. 9, we show the rendered face model with 16 different facial expressions. We randomly combined different expressions of eye area and mouth region that were seamlessly integrated in the head model. For our experiments, we captured our actress with eight synchronised video cameras (5120 × 3840@25 fps) and eight D-SLR cameras (5184 × 3456). While we used high-quality hardware for our data capture, our method is not restricted to this exact hardware configuration. A minimal capture setup could, for example, consist of three video cameras alone to record the subject's face from the left side, from the right side and frontally. Instead of creating a personalised face model, it is also possible to employ an existing blendshape model (e.g. [5]) or to use a single 3D reconstruction of the subject's face (e.g. neutral expression) that is deformed based on the optical flow in

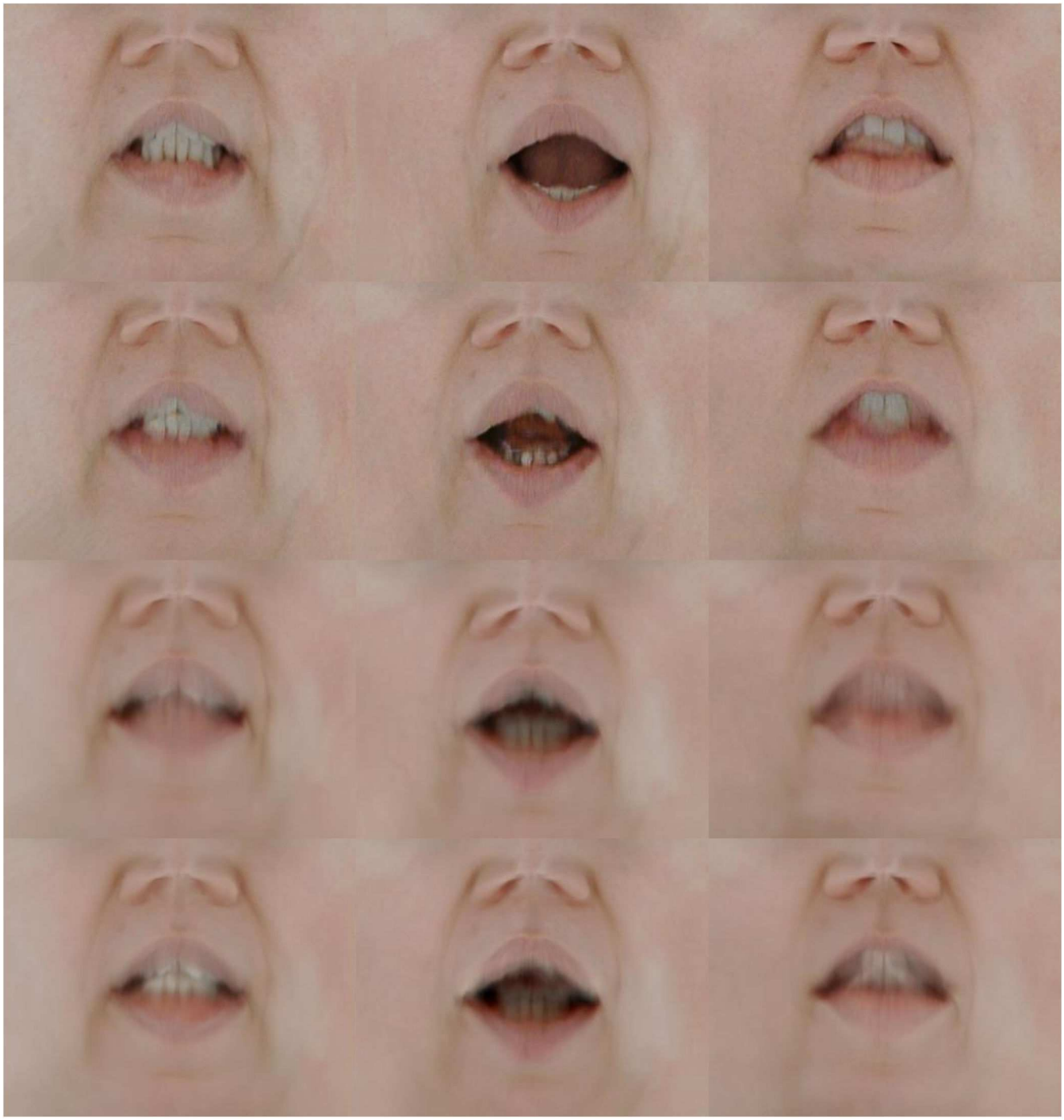

Fig. 11 *Comparison of the achieved texture quality for PCA, GPLVM and VAE. Each column corresponds to an expression. The rows correspond to (top to bottom): ground truth, VAE, GPLVM and PCA results*

the video sequences (e.g. [8, 41]). This is possible due to the fact that our method does not rely on accurate and detailed 3D geometry for animation. This is demonstrated in Fig. 10, where we present facial expressions with strong deformations and dissocclusions (e.g. teeth and tongue appear in the opened mouth which is not represented by geometry). While the data-processing and training have to be done in an offline learning phase, we can animate and render interactively at a rate of 20 frames per second. Rendering and training performed on a regular desktop computer with 64 GB Ram, 2.6 GHz CPU (14 cores with hyperthreading) and one GeForce RTX2070 graphics card. The animated eye region has a size of 360×100 pixel and needs ∼2.5 GB of graphic memory during training. The mouth area is 470×380 pixel big and needs ∼4.5 GB of GPU memory during training, which means that both models can be trained simultaneously on one GPU. Our face model is based on the SMPL mesh template [34]. While we adapt geometry and compute the personalised face geometry model automatically, we used a semiautomatic mesh unwrapping technique to add texture coordinates to the mesh and annotate face landmarks on the mesh. Since these are tasks that need to be done only once, we do not consider it as a strong limitation. Currently, we use a neural texture model that does not account for view dependency of textures. While this is not a major limitation in most cases, it can cause erroneous perspective distortions if the geometry is not accurate enough and the rendering viewpoint differs much from the camera's viewpoint from which the texture was generated.

In order to assess the effectiveness of our VAE regarding the representation of dynamic textures, we compare the reconstruction quality of our model with two other well-known methods, namely PCA and the Gaussian process latent variable model (GPLVM) [42]. PCA-based representations are a popular choice in computer-vision and computer-graphics, especially when dealing with high-dimensional data that can be well explained by a linear

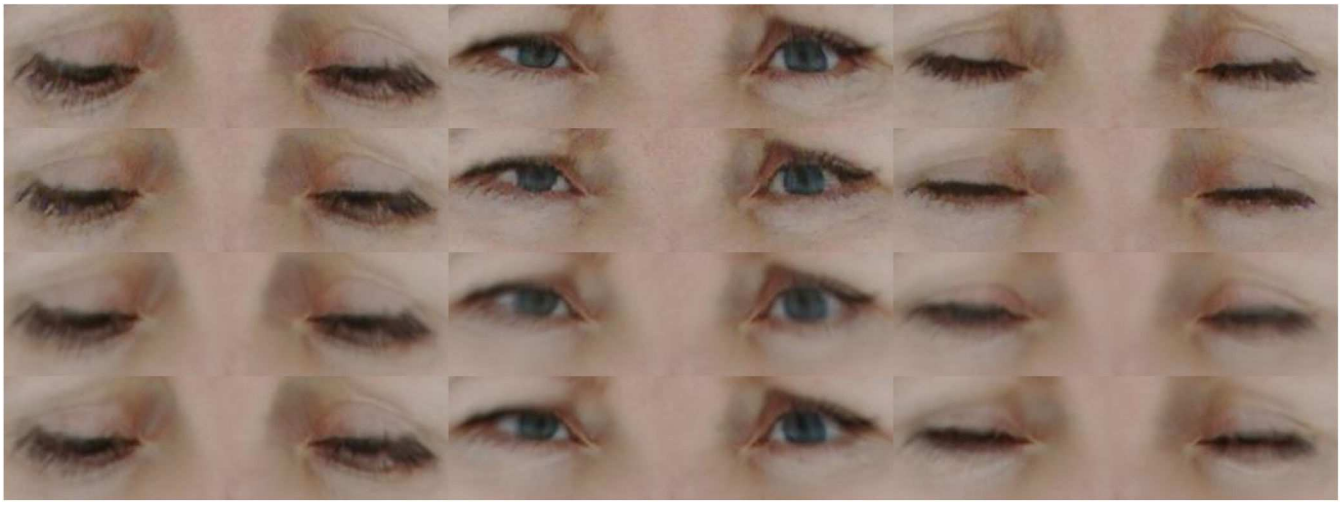

Fig. 12 *This figure compares the achieved texture quality between PCA, GPLVM and VAE. Each column corresponds to an expression. The rows correspond to (top to bottom): ground truth, VAE, GPLVM and PCA results*

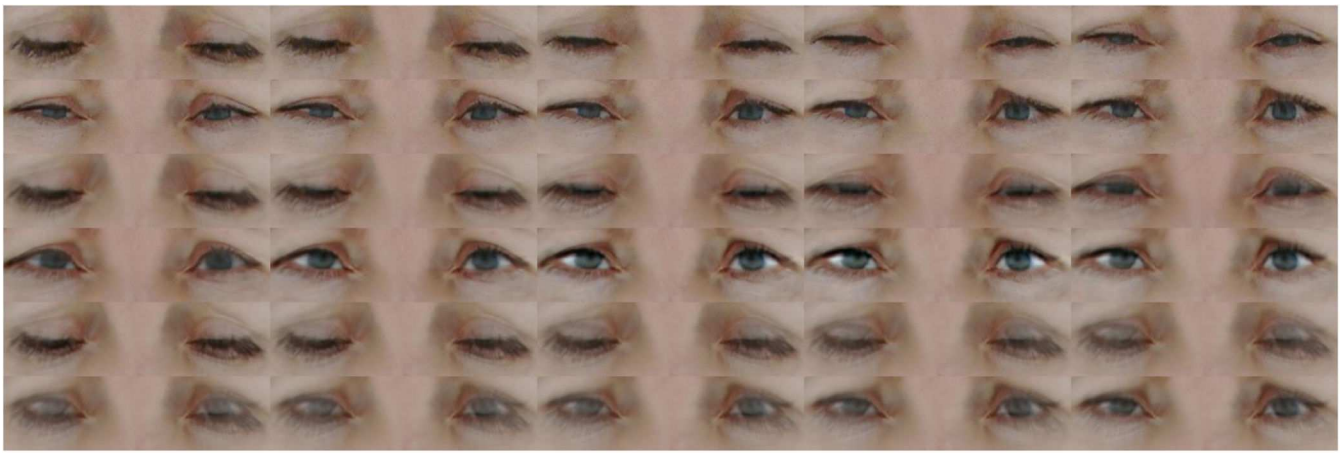

Fig. 13 *Comparison of the interpolation capabilities of a PCA-, GPLVM- and VAE-based texture representation. We synthesise intermediate expressions to simulate a transition from closed to open eyes. The two top rows show the expressions that are synthesised by our proposed model. The rows in the middle show expressions that are synthesised by the GPLVM-based approach and the two bottom rows correspond to the PCA-based texture interpolation. While the VAE-based approach shows realistic intermediate expressions, PCA and GPLVM textures exhibit ghosting and/or blur*

combination of ortho-normal base vectors and a constant offset. While PCA-based models assume a linear relationship between data and its latent representation, GPLVMs overcome this limitation and are also capable of learning non-linear relations between data and latent variables. For our experiments, we trained PCA-based models as well as GPLVMs on the mouth and eye textures and compared the reconstructed textures with the ones generated by our VAE, see Figs. 11 and 12. For both non-linear models, we used a 2D latent space. For the linear PCA model, we chose the number of bases such that the number of model parameters is approximately the same as in the VAE. As the VAE has ∼9.6 million parameters, we use 20D PCA basis. Looking at the reconstructed mouth textures, the most obvious difference between the results of PCA/GPLVM and VAE is that the textures of the latter are less blurry and do not suffer from ghosting artefacts. The blurriness is caused by the fact that we capture small motions, deformations as well as occlusions/disocclusions in texture space. This strategy greatly simplifies the general process of facial performance capture, but in order to produce high-quality results we need a more sophisticated texture model that is actually capable of representing complex changes in texture space. Comparision of the mouth and eye textures shows that eye textures are in general more detailed and less blurry for all three models. This might be caused by the fact that eye movements and blinks are too fast to be captured by our cameras, which results in clusters of similar eye states rather than the actual motion. Therefore, even the simple linear PCA model is able to reconstruct artefact-free textures. However, as we want to use our model not only for the reconstruction of already captured data but also for generating intermediate expressions (e.g. when changing from one facial expression to another), we conducted another experiment where we use all three models to interpolate between closed and open eyes, see Fig. 13. To synthesise the intermediate expressions, we performed a linear interpolation in the latent space. The results show that even though all three models are able to faithfully reconstruct the original eye textures, only the VAE-based model is able to synthesise a texture sequence that shows a realistic transition from closed to open eyes. The texture sequence that is generated by the PCA model contains ghosting artefacts because the interpolation in latent space essentially results in a linear interpolation of corresponding texel values and the GPLVM synthesises a texture sequence that actually shows moving eyelids but textures are much blurrier than the VAE textures.

7 Conclusion

We presented a new method for video-based animation of human faces. The key idea of the method is to allow animators to use deep neural networks in simple way to control facial expressions of a virtual human character. The simple representation on a 2D grid makes it easy to create believable animations, even for untrained users. The data-driven approach allows generating realistic facial animations and renderings since our neural face model is learned completely from captured data of our actress. Facial animation parameters can be easily generated from real data by encoding sequences of extracted texture and geometry, which enables a user

to create a database of realistic animation sequences in a motion capture-like way. However, unlike motion capture, the facial expressions can be represented holistically in a low-dimensional latent space that allows easy manipulation. The representation of texture and geometry data with autoencoders allows for smooth and artefact-free interpolation between facial expressions, which enables a user to perform interesting operations with captured video and geometry data like concatenation or looping of atomic sequences to create novel animation sequences that have not been captured in that way. Apart from that, we also compare the VAE-based texture representation with a PCA and GPLVM-based approach and show that in contrast to PCA/GPLVM, our proposed method generates high-quality texture sequences that do not exhibit strong blur or ghosting artefacts.

8 Acknowledgments

This work was partly funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No 762021 (Content4All) as well as the Germany Federal Ministry of Education and Research under grant agreement 03ZZ0468 (3DGIM2).

9 References

[1] Cootes, T.F., Edwards, G.J., Taylor, C.J.: 'Active appearance models', *IEEE Trans. Pattern Anal. Mach. Intell.*, 1998, **23**, pp. 484–498

[2] Blanz, V., Vetter, T.: 'A morphable model for the synthesis of 3d faces'. Proc. 26th Annual Conf. on Computer Graphics and Interactive Techniques. SIGGRAPH '99, New York, NY, USA,, 1999, pp. 187–194. http://dx.doi.org/10.1145/311535.311556

[3] Vlasic, D., Brand, M., Pfister, H., *et al.*: 'Face transfer with multilinear models', *ACM Trans. Graph.*, 2005, **24**, (3), pp. 426–433. http://doi.acm.org/10.1145/1073204.1073209

[4] Li, H., Yu, J., Ye, Y., *et al.*: 'Realtime facial animation with on-the-fly correctives', *ACM Trans. Graph.*, 2013, **32**, (4), pp. 42:1–42:10. http://doi.acm.org/10.1145/2461912.2462019

[5] Cao, C., Weng, Y., Zhou, S., *et al.*: 'Facewarehouse: A 3d facial expression database for visual computing', *IEEE Trans. Vis. Comput. Graphics*, 2014, **20**, (3), pp. 413–425. http://dx.doi.org/10.1109/TVCG.2013.249

[6] Garrido, P., Valgaert, L., Wu, C., *et al.*: 'Reconstructing detailed dynamic face geometry from monocular video', *ACM Trans. Graph.*, 2013, **32**, (6), pp. 158:1–158:10. http://doi.acm.org/10.1145/2508363.2508380

[7] Thies, J., Zollhöfer, M., Nießner, M., *et al.*: 'Real-time expression transfer for facial reenactment', *ACM Trans. Graph. (TOG)*, 2015, **34**, (6), pp. 1–14

[8] Paier, W., Kettern, M., Hilsmann, A., *et al.*: 'A hybrid approach for facial performance analysis and editing', *IEEE Trans Cir Syst Video Technol.*, 2017, **27**, (4), pp. 784–797. https://doi.org/10.1109/TCSVT.2016.2610078

[9] Fechteler, P., Paier, W., Hilsmann, A., *et al.*: 'Real-time avatar animation with dynamic face texturing'. Proc. 23rd IEEE Int. Conf. on Image Processing (iCIP 2016), Phoenix, Arizona, USA, 2016

[10] Paier, W., Kettern, M., Anna, H., *et al.*: 'Video-based facial re-animation'. Proc. 12th European Conf. on Visual Media Production, November, 2015

[11] Casas, D., Volino, M., Collomosse, J., *et al.*: '4d video textures for interactive character appearance', *Comput. Graph. Forum*, 2014, **33**, (2), pp. 371–380

[12] Fechteler, P., Paier, W., Eisert, P.: 'Articulated 3D model tracking with on-the-fly texturing'. Proc. 21st IEEE Int. Conf. on Image Processing (iCIP 2014), Paris, France, 2014, pp. 3998–4002

[13] Dale, K., Sunkavalli, K., Johnson, M.K., *et al.*: 'Video face replacement'. ACM Transactions on Graphics (Proc SIGGRAPH Asia), Hong Kong, 2011, vol. **30**

[14] Lipski, C., Klose, F., Ruhl, K., *et al.*: 'Making of 'who cares?' HD stereoscopic free viewpoint video'. Proc. of the Eighth European Conf. on Visual Media Production, London, November 2011

[15] Kilner, J., Starck, J., Hilton, A.: 'A comparative study of free-viewpoint video techniques for sports events'. Proc. Third European Conf. on Visual Media Production, London, November 2006

[16] Borshukov, G., Montgomery, J., Werner, W., *et al.*: 'Playable universal capture'. ACM SIGGRAPH 2006 Sketches, Boston, August 2006

[17] Carranza, J., Theobalt, C., Magnor, M.A., *et al.*: 'Free-viewpoint video of human actors', *ACM Trans. Graph.*, 2003, **22**, (3), pp. 569–577

[18] Kingma, D.P., Welling, M.: 'Auto-encoding variational Bayes'. Banff, Canada, 2013

[19] Goodfellow, I.J., Pouget.Abadie, J., Mirza, M., *et al.*: 'Generative adversarial nets'. Proc. 27th Int. Conf. on Neural Information Processing Systems - Volume 2 (NIPS'14), Cambridge, MA, USA, 2014, pp. 2672–2680. Available at: http://dl.acm.org/citation.cfm?id=2969033.2969125

[20] Radford, A., Metz, L., Chintala, S.: 'Unsupervised representation learning with deep convolutional generative adversarial networks', 2015

[21] Lample, G., Zeghidour, N., Usunier, N., *et al.*: 'Fader networks: manipulating images by sliding attributes', CoRR, Long Beach, CA, USA, 2017, abs/1706.00409. Available at: http://arxiv.org/abs/1706.00409

[22] Kulkarni, T.D., Whitney, W., Kohli, P., *et al.*: 'Deep convolutional inverse graphics network', CoRR, Montréal, Canada, 2015, abs/1503.03167. Available at: http://arxiv.org/abs/1503.03167

[23] Pumarola, A., Agudo, A., Martinez, A.M., *et al.*: 'Ganimation: anatomically-aware facial animation from a single image'. Proc. European Conf. on Computer Vision (ECCV), Munich, Germany, 2018

[24] Geng, J., Shao, T., Zheng, Y., *et al.*: 'Warp-guided gans for single-photo facial animation', *ACM Trans. Graph.*, 2018, **37**, (6), pp. 231:1–231:12. Available at: http://doi.acm.org/10.1145/3272127.3275043

[25] Huang, Y.M., Khan, S.: 'Dyadgan: generating facial expressions in dyadic interactions'. The IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) Workshops, Honolulu, Hawaii, July 2017

[26] Bansal, A., Ma, S., Ramanan, D., *et al.*: 'Recycle-gan: unsupervised video retargeting'. ECCV, Munich, Germany, 2018

[27] Webber, S., Harrop, M., Downs, J., *et al.*: 'Everybody dance now: tensions between participation and performance in interactive public installations'. Proc. Annual Meeting of the Australian Special Interest Group for Computer Human Interaction(OzCHI '15)New York, NY, USA, 2015, pp. 284–288. Available at: http://doi.acm.org/10.1145/2838739.2838801

[28] Lombardi, S., Saragih, J.M., Simon, T., *et al.*: 'Deep appearance models for face rendering'. CoRR, Montréal, Canada, 2018, abs/1808.00362. Available at: http://arxiv.org/abs/1808.00362

[29] Slossberg, R., Shamai, G., Kimmel, R.: 'High quality facial surface and texture synthesis via generative adversarial networks', CoRR, Munich, Germany, 2018, abs/1808.08281. Available at: http://arxiv.org/abs/1808.08281

[30] Lowe, D.G.: 'Distinctive image features from scale-invariant keypoints', *Int. J. Comput. Vision*, 2004, **60**, (2), pp. 91–110. Available at: https://doi.org/10.1023/B:VISI.0000029664.99615.94

[31] Furch, J., Eisert, P.: 'An iterative method for improving feature matches'. 2013 Int. Conf. on 3D Vision - 3DV 2013, June 2013, pp. 406–413

[32] Kazhdan, M., Hoppe, H.: 'Screened Poisson surface reconstruction', *ACM Trans. Graph.*, 2013, **32**, (3), pp. 29:1–29:13. Available from: http://doi.acm.org/10.1145/2487228.2487237

[33] Blumenthal.Barby, D., Eisert, P.: 'High-resolution depth for binocular image-based modelling', *Comput. Graph.*, 2014, **39**, pp. 89–100

[34] Loper, M., Mahmood, N., Romero, J., *et al.*: 'SMPL: A skinned multi-person linear model', *ACM Trans Graphics (Proc SIGGRAPH Asia)*, 2015, **34**, (6), pp. 248:1–248:16

[35] Kazemi, V., Sullivan, J.: 'One millisecond face alignment with an ensemble of regression trees'. Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition, Columbus, Ohio, USA, 2014, pp. 1867–1874

[36] Pinkall, U., Polthier, K.: 'Computing discrete minimal surfaces and their conjugates', *Exp. Math.*, 1993, **2**, (1), pp. 15–36

[37] Meyer, M., Desbrun, M., Schröder, P., *et al.*: '*Discrete differential-geometry operators for triangulated 2-manifolds*' (Springer-Verlag, Germany, 2002, pp. 35–57)

[38] Boykov, Y., Veksler, O., Zabih, R.: 'Fast approximate energy minimization via graph cuts', *IEEE Trans. Pattern Anal. Mach. Intell.*, 2001, **23**, (11), pp. 1222–1239. Available from: https://doi.org/10.1109/34.969114

[39] Pérez, P., Gangnet, M., Blake, A.: 'Poisson image editing', *ACM Trans. Graph.*, 2003, **22**, (3), pp. 313–318. Available at: http://doi.acm.org/10.1145/882262.882269

[40] Isola, P., Zhu, J.Y., Zhou, T., *et al.*: 'Image-to-image translation with conditional adversarial networks', arxiv, 2016

[41] Kettern, M., Hilsmann, A., Eisert, P.: 'Temporally consistent wide baseline facial performance capture via image warping'. Proc. Vision, Modeling, and Visualization Workshop 2015, Aachen, Germany, October 2015

[42] Lawrence, N.D.: 'Gaussian process latent variable models for visualisation of high dimensional data'. Proc. of the 16th Int. Conf. on Neural Information Processing Systems (NIPS'03), Cambridge, MA, USA, 2003, pp. 329–336