Automated Blendshape Personalization for Faithful Face Animations Using Commodity Smartphones

Timo Menzel
timo.menzel@tu-dortmund.de
TU Dortmund University
Dortmund, Germany

Mario Botsch
mario.botsch@tu-dortmund.de
TU Dortmund University
Dortmund, Germany

Marc Erich Latoschik
marc.latoschik@uni-wuerzburg.de
Julius-Maximilians-Universität
Würzburg, Germany

ABSTRACT

Digital reconstruction of humans has various interesting use-cases. Animated virtual humans, avatars and agents alike, are the central entities in virtual embodied human-computer and human-human encounters in social XR. Here, a faithful reconstruction of facial expressions becomes paramount due to their prominent role in non-verbal behavior and social interaction. Current XR-platforms, like Unity 3D or the Unreal Engine, integrate recent smartphone technologies to animate faces of virtual humans by facial motion capturing. Using the same technology, this article presents an optimization-based approach to generate personalized blendshapes as animation targets for facial expressions. The proposed method combines a position-based optimization with a seamless partial deformation transfer, necessary for a faithful reconstruction. Our method is fully automated and considerably outperforms existing solutions based on example-based facial rigging or deformation transfer, and overall results in a much lower reconstruction error. It also neatly integrates with recent smartphone-based reconstruction pipelines for mesh generation and automated rigging, further paving the way to a widespread application of human-like and personalized avatars and agents in various use-cases.

CCS CONCEPTS

• **Computing methodologies** → **Mesh geometry models**; *Motion capture.*

KEYWORDS

virtual humans, face animation, blendshapes, personalization, deformation transfer, facial rigging

ACM Reference Format:
Timo Menzel, Mario Botsch, and Marc Erich Latoschik. 2022. Automated Blendshape Personalization for Faithful Face Animations Using Commodity Smartphones. In *28th ACM Symposium on Virtual Reality Software and Technology (VRST '22), November 29-December 1, 2022, Tsukuba, Japan.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3562939.3565622

1 INTRODUCTION

A faithful digital reconstruction of humans has various interesting use-cases throughout the media industry and beyond. Virtual actors mimicking real actors are becoming increasingly commonplace since their debut in the 1987 CGI movie Rendez-vous in Montreal, see, e.g., the late creations from the Star Wars franchise. Similarly, the fidelity of interactive Virtual Humans (VHs) [33] also significantly advanced, as demonstrated by commercial developments like Epic Games' MetaHumans. Here, high-fidelity digital reconstructions of humans open up promising applications in computer games as well as in Virtual, Augmented, and Mixed Reality (VR, AR, MR: XR for short) [7]. These applications include human-computer interactions with computer-controlled VHs, so-called virtual agents [11, 32, 37], as well as mediated human-human encounters with user-controlled VHs, so-called avatars (see, eg., [3, 21, 22, 29, 38]), in future social XRs.

Facial expressions are a central channel of non-verbal behavior. Their prominent role in social interaction has been confirmed for quite some time now [13, 40]. There is evidence that non-verbal behavior conveys the majority of information communicated [34, 42]. Facial expressions are specifically prime conveyors of "emotions, attitudes, interpersonal roles, and severity of pathology" [13, p. 50]. Overall, facial expressions are an important modality of human-human interaction. Therefore, synthesis and reconstruction of facial expressions of VHs and their effects on observers have been intensively researched [33], as was their role in non-verbal interaction between avatars confirmed (see, e.g., [41, 42]).

Faithful digital reconstruction of humans for interactive applications faces unique challenges. Most approaches capture the outer appearance using depth cameras [30] or photogrammetry [1, 15], and then rig the resulting mesh for subsequent animation by defining a weighted assignment of mesh vertices to skeleton bones for body animation and defining facial blendshapes for face animation. Today, this rigging process can be automated to a large extend by employing template models with predefined rigs: The skeletal rig can be transferred by non-rigid registration of the template to the target mesh [5], and the facial blendshapes are typically mapped from template to target using deformation transfer [47].

While generic template rigs have been shown to work well for body animation [31, 36], the template's facial blendshapes in general do not faithfully reconstruct a captured person's unique facial expressions. These differences are particularly problematic since humans are capable of detecting even subtle changes and deviations in human faces, which can potentially even lead to incongruent social cues [23, 42]. A solution to this problem is provided by using personalized blendshapes [10, 17, 20, 26], which are derived from a training set of facial expression. However, existing approaches still vary considerably in reconstruction accuracy (hence faithfulness), level of automation, ease of use and applicability.

VRST '22, November 29-December 1, 2022, Tsukuba, Japan

ACM ISBN 978-1-4503-9889-3/22/11.
https://doi.org/10.1145/3562939.3565622

Contribution: In this paper we leverage the capabilities of recent smartphone technologies (in particular Apple's ARKit) to generate personalized blendshapes for a given avatar in a fully automatic and easy-to-use manner. Our approach ideally complements recent smartphone-based avatar reconstructions (e.g., [51]) and perfectly matches the smartphone-based facial motion capturing provided in Unity 3D and the Unreal engine. Using ARKit for capturing example facial expressions, we extend the widely used example-based facial rigging [26] to extract considerably more accurate facial blendshapes from these example expressions, which are then seamlessly implanted into a target avatar using a novel formulation of deformation transfer [47]. Compared to previous approaches, our extensions lead to considerably more accurate and hence more faithful reconstructions, reducing the reconstruction error for some cases down to 10%, while still being comparable in terms of computational cost.

2 RELATED WORK

While many different approaches for face animation have been proposed, such as skeleton and joint models [46] or physics-based muscle models [49], linear blendshape models [25] are still the most widely used technique – in particular in interactive XR applications. The individual blendshapes (or morph targets) are typically based on the facial action coding system (FACS) [14], giving them an anatomical as well as semantical meaning. The FLAME model of Li et al. [28] replaces the over-complete, linearly dependent FACS blendshape basis by an orthogonal PCA basis. While this is indeed better suited for tracking and reconstruction, their basis lacks semantic meaning and is therefore not suitable for several applications. To be as widely applicable as possible, our method employs standard linear facial blendshapes.

Generating the required blendshapes by scanning an actor in all these expressions is not possible in most situations. Hence the blendshapes of a generic face rig are typically transferred to the target avatar, using for instance RBF warps [19, 35, 45], non-rigid registration [16], or variants of deformation transfer [39, 47, 48]. The blendshapes generated this way match the anatomical dimensions of the target avatar, but they typically do not faithfully reconstruct the captured person's unique facial expressions.

Higher-quality approaches therefore personalize blendshapes by capturing an actor not only in the neutral pose, but also in a couple of example expressions [8, 18, 26, 50]. The optimization involved in the underlying example-based facial rigging process [26] is ill-posed, since both the blendshape meshes of the actor and the blendshape weights of the captured expressions are unknown. The method therefore depends on good initial guesses of the blendshape weights, which restricts the poses the actor can (or has to) perform. Li et al. [26] proposed a subset of 15 facial expressions the actor should perform. These expressions consist of more than one single activated blendshape; they are instead a combination of several activated blendshapes. Carrigan et al. [9] use an even smaller set of facial expressions by packing more basic blendshapes into the training expressions – making them harder to perform though.

In contrast, Ichim et al. [20] let the actor perform a sequence of dynamic facial expressions and video-record this performance using a smartphone. They detect facial feature points in each frame and non-rigidly deform their blendshapes to better fit these landmarks. However, the user has to check and correct the detected landmarks in about 25 of 1500 frames to account for errors of the facial feature detector. We also use a smartphone to record a facial performance of the actor, but leverage an iPhone's video and depth sensor and Apple's ARKit to capture blendshape weights and 3D geometry for each frame of this performance in a robust and fully automatic manner. Since our capturing process is easy and intuitive, we currently let the actor mimic all 52 ARKit blendshapes to get a maximum of personalization.

Han et al. [18] evaluated two methods to extract personalized blendshapes from expressions scans. They recorded expressions and blendshape weights and fitted an autoencoder and a linear regression to obtain the blendshapes. However, while they achieve good results with respect to the reconstruction error, they do not compute semantically meaningful blendshapes, which disqualifies their approach for many applications. Li et al. [26] and Seol et al. [44] employ optimizations based on deformation gradients to extract personalized blendshapes from a set of training examples. Li et al. [26] fit the deformation gradients of the unknown blendshapes to the deformation gradients of the example expressions, regularized by (the deformation gradients of) the blendshapes of generic rig to ensure semantically meaningful blendshapes. Blendshape weights and blendshape meshes are solved for in an alternating optimization, requiring careful initialization for convergence. Seol et al. [44] first fit a mesh to the scanned expressions and then separate the fitted expressions into the different blendshapes based on the vertex displacements in the generic rig. In our experiments, their blendshape separation method leads to strongly damped blendshapes though. Our approach leverages ARKit's face tracking and therefore avoids the alternating optimization for blendshape weights and blendshape meshes. As we show in Section 4, our fitting also leads to considerably more accurate results compared to [26].

Several approaches further increase the reconstruction accuracy by using corrective blendshapes or corrective deformation fields. The facial animation is then computed by the initial (generic) blendshape set, but it is refined with additional shapes, which add idiosyncrasies that cannot be represented by the initial blendshapes [6, 12, 17, 20, 27]. Our work focusses on computing personalized blendshapes without extra corrective shapes, to be compatible with standard blendshape pipelines in XR engines, but corrective fields could easily be added in future work. In contrast to many previous works that use a face/head avatar only, we implant the resulting personalized facial blendshapes into a full-body avatar using a modification of deformation transfer [4, 47].

3 METHOD

In this section we present our approach for extracting personalized blendshapes from a set of scanned facial expressions and how to transfer these blendshapes to an existing avatar. Figure 1 shows an overview of our pipeline. We begin by specifying the capturing process and the resulting input data (Section 3.1). From this training data we extract personalized blendshapes (Section 3.2), which are then transferred to the full-body avatar using our seamless partial deformation transfer (Section 3.3).

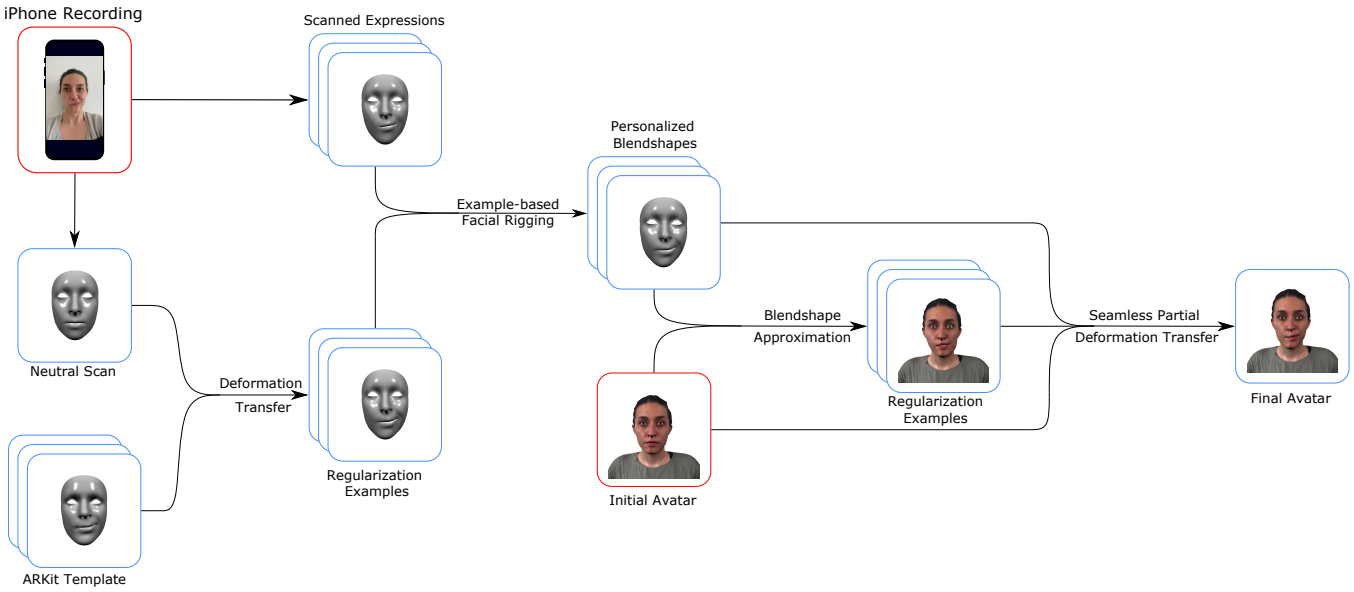

Figure 1: Our pipeline from recording example expressions to the final avatar. Red boxes indicate user input.

3.1 Input Data

Our goal is to create personalized blendshapes for an existing full-body avatar. While any method could be used to generate that avatar, we employ the smartphone-based method of Wenninger et al. [51]. As a consequence, the complete avatar generation and personalization only requires a smartphone for capturing the person, which is in stark contrast to recent approaches based on complex photogrammetry rigs [1, 15] and makes the reconstruction of personalized avatars more widely available.

We capture the training data for the blendshape personalization using a custom application running on an iPhone 12 Pro (any recent ARKit-capable iOS device could be used as well). This application guides the user through the recording session, making the whole capturing process easy and intuitive. All recording is performed by the front-facing depth and color cameras, such that the user can watch instructions and get feedback while recording.

The user first scans their own face in a neutral expression, and is then asked to perform the facial expressions corresponding to the $m = 52$ ARKit blendshapes. For each of these blendshapes, the application shows a textual and pictorial explanation of the expression to be performed. The application continuously tracks the user's face using the ARKit framework and automatically captures the facial expression when the requested expression is performed to a sufficient extent. In particular, when capturing blendshape i (for $i \in \{1, \dots, m\}$), we observe the blendshape weight $w_{i,i}$ corresponding to that blendshape and once this weight exceeds a certain threshold and reaches a maximum over time (i.e., starts decreasing), we record both the current set of blendshape weights $(w_{i,1}, \dots, w_{i,52})$ as well as the current geometry $\mathbf{S}_i$ of the ARKit face mesh, where the latter consists of $n = 1220$ vertices and 2304 triangles. While we expect the blendshape weight $w_{i,i}$ to be dominant when capturing blendshape i, other non-vanishing blendshape weights $w_{i,j}$ do not pose a problem for our reconstruction (see next section), such that the user does not have to strictly perform the requested expression in isolation (which is hardly possible).

With this automated recording procedure, the 52 requested facial expressions can be scanned in approximately 2 minutes.

3.2 Blendshape Personalization

While ARKit provides us with blendshape weights $(w_{i,1}, \dots, w_{i,m})$ and face meshes $\mathbf{S}_i$ for each expression $i \in \{1, \dots, m\}$ of the $m = 52$ ARKit blendshapes, it does not give access to the internal personalized blendshapes of the user. We therefore have to extract the personalized version of the ARKit blendshapes using a modification of example-based facial rigging [26].

The original method of Li et al. [26] fits the per-triangle deformation gradients of the unknown blendshapes to the deformation gradients of the scanned expressions in a first step, and then solves a linear least-squares system to extract the vertex positions best-matching the fitted deformation gradients [4, 47]. This two-step procedure has the disadvantage that it does not directly penalize the deviation of the reconstructed vertex positions from the scanned expressions, it only *indirectly* encourages the vertex positions to match the target expressions. In contrast, we *directly* optimize for vertex positions that best-match the observed expressions $\mathbf{S}_i$ in the least-squares sense (see Equation (1) below), which – as we will shown in Section 4.1 – leads to more accurate results.

Given the blendshape weights $(w_{i,1}, \dots, w_{i,m})$ and face meshes $\mathbf{S}_i$ for the recorded expressions $i \in \{1, \dots, m\}$, as well as the neutral face scan $\mathbf{B}_0$, we have to compute the corresponding personalized delta-blendshapes $\mathbf{B}_1, \dots, \mathbf{B}_m$. Delta-blendshapes describe the displacement from the neutral expression to a predefined facial expression – the corresponding (non-delta-)blendshape [25]. In the following, the term *blendshape* always refers to delta-blendshapes.

Our model then produces a facial expression from blendshape weights $w_1, \ldots, w_m$ as

$$\mathbf{B}_0 + \sum_{j=1}^{m} w_j \mathbf{B}_j .$$

Here, the matrix $\mathbf{B}_0 \in \mathbb{R}^{n\times 3}$ contains the $n = 1220$ vertex positions of the neutral face and the matrices $\mathbf{B}_1, \ldots, \mathbf{B}_m \in \mathbb{R}^{n\times 3}$ contain the n displacement vectors of the blendshapes, respectively.

We compute the blendshapes $\mathbf{B}_j$ by penalizing the distance from the observed training expressions $\mathbf{S}_i$, formulated as the cost function

$$E_{\text{fit}}(\mathbf{B}_1, \ldots, \mathbf{B}_m) \;=\; \sum_{i=1}^{m} \left\| \mathbf{B}_0 + \sum_{j=1}^{m} w_{i,j}\mathbf{B}_j - \mathbf{S}_i \right\|^2 . \qquad (1)$$

Since the blendshape weights $w_{i,j}$ were captured by ARKit and hence are known, minimizing (1) only requires solving a sparse linear least-squares system.

However, in order to produce semantically meaningful blendshapes the optimization has to be regularized. To this end, we transfer the blendshapes of the (non-personalized) ARKit template to the recorded neutral expression $\mathbf{B}_0$ using deformation transfer [47], resulting in the generic blendshapes $\mathbf{T}_1, \ldots, \mathbf{T}_m$. Similar to Saito [43], we add virtual triangles between upper and lower eyelids to ensure that the eyes are completely closed in the transferred eye-blink blendshapes. Our regularization energy then penalizes the deviation of the personalized blendshapes $\mathbf{B}_j$ from the generic blendshapes $\mathbf{T}_j$:

$$E_{\text{reg}} \;=\; \sum_{j=1}^{m} \left\| \mathbf{D}_j \left(\mathbf{B}_j - \mathbf{T}_j \right) \right\|^2 . \qquad (2)$$

Here, $\mathbf{D}_j$ are diagonal $(n \times n)$ matrices containing per-vertex regularization weights. These weights ensure that vertices that do not move in the template blendshape $\mathbf{T}_j$ also do not move in the personalized blendshape $\mathbf{B}_j$. They are computed as

$$\left(\mathbf{D}_j\right)_{i,i} \;=\; \frac{\max_{k=1,\ldots,n} \left\| \mathbf{t}_{k,j} - \mathbf{t}_{k,0} \right\|}{\left\| \mathbf{t}_{i,j} - \mathbf{t}_{i,0} \right\|}, \qquad (3)$$

where $\mathbf{t}_{i,j} \in \mathbb{R}^3$ is the position of the i-th vertex in the template blendshape $\mathbf{T}_j$. This results in higher regularization weights for vertices with smaller displacement magnitude in the ARKit template blendshapes. We clamp the regularization weight $(\mathbf{D}_j)_{i,i}$ to 10^5 if $\left\| \mathbf{t}_{i,j} - \mathbf{t}_{i,0} \right\| < \epsilon$ to avoid division by zero.

The personalized blendshapes are finally computed by minimizing the cost function

$$E_{\text{fit}}(\mathbf{B}_1, \ldots, \mathbf{B}_m) + E_{\text{reg}}(\mathbf{B}_1, \ldots, \mathbf{B}_m)$$

that combines the fitting term (1) and the regularization term (2), which again only involves solving a least-squares linear system.

Li et al. [26] stated that an optimization based on vertex positions leads to visible artifacts if done naively. However, our results demonstrate quantitatively and qualitatively that this is not the case for our regularization. In fact, optimizing vertex positions leads to more accurate results, as we show in Section 4. Without our regularization though, we would get clearly noticeable artifacts.

Figure 2: The personalized blendshape (left) and our approximation with generic blendshapes (right)

3.3 Blendshape Transfer

Having computed the personalized ARKit blendshapes (grey face masks in Figure 1), we now transfer them to the provided full-body avatar using an extension of deformation transfer [47]. Note that our personalized ARKit blendshapes $\mathbf{B}_j$ only specify the deformation within the face area. However, this is only a part of the deformation due to face animation, since some blendshapes (e.g. jaw open) also affect the area adjacent to the face (e.g. the neck area). Therefore, we have to adjust the adjacent area accordingly.

In a first step we approximate the personalized ARKit blendshapes with the pre-existing non-personalized blendshapes of the full-body avatar. To this end, we define a correspondence map $\mathbf{M}$ between the avatar's face and the ARKit face mesh. This is achieved by fitting the avatar template to the ARKit template using non-rigid registration [1] and selecting the closest point on the ARKit mesh for each vertex of the avatar's face. Since each full-body avatar shares the topology of the full-body template model (from [51] in our case), this mapping have to be computed only once.

To obtain the optimal blendshape weights for the approximation we use an approach similar to Lewis and Anjyo [24]. First, we use the iterative closest point algorithm (ICP) with scaling [52] to find the optimal translation, rotation, and scaling to register the ARKit face mesh to the avatar's face. Second, we compute the weights of the initial avatar blendshapes by minimizing the energy

$$E^j_{\text{approx}}(\tilde{w}_1, \ldots, \tilde{w}_k) \;=\; \left\| \sum_{i=1}^{k} \tilde{w}_i \tilde{\mathbf{B}}_i - \mathbf{M}\mathbf{B}_j \right\|^2 + \mu \sum_{i=1}^{k} \tilde{w}_i^2, \qquad (4)$$

where $\mathbf{M}$ is the pre-computed correspondence matrix that maps vertices of the ARKit face meshes to the full-body avatar. $\tilde{\mathbf{B}}_i$ denote the initial avatar's blendshapes and $\tilde{w}_i$ are the (unknown) blendshape weights. The second term penalizes large weights to avoid extreme poses. Solving a linear least-squares system results in the blendshape weights $\tilde{w}_1, \ldots, \tilde{w}_k$ for approximating the ARKit blendshape $\mathbf{B}_j$ using the initial avatar blendshapes $\tilde{\mathbf{B}}_1, \ldots, \tilde{\mathbf{B}}_k$.

The resulting approximation, which can be considered an *automatic facial retargeting* and is denoted by $\mathbf{A}_j$, is already quite close to the desired personalized blendshape $\mathbf{B}_j$ (see Figure 2). Using

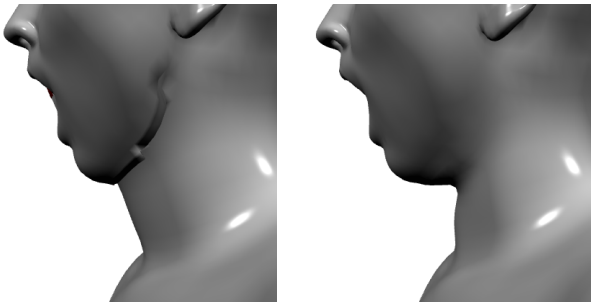

Figure 3: Seamless partial deformation transfer of the Jaw-Open blendshape: The first step transfers the deformation of the face region only (left), the second step adjusts the adjacent neck region (right).

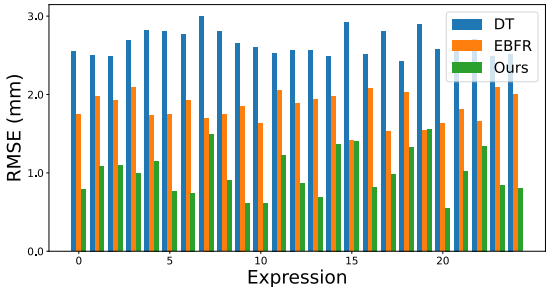

Figure 4: Root-mean-square reconstruction error of deformation transfer (blue), example-based facial rigging (orange), and our method (green) for Subject 1.

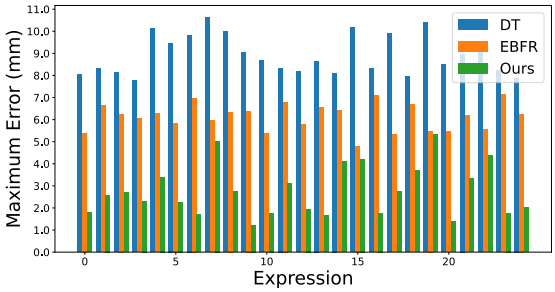

Figure 5: Maximum reconstruction error of deformation transfer (blue), example-based facial rigging (orange), and our method (green) for Subject 1.

the initial avatar's blendshapes it provides the missing information on how to deform the avatar's face in the area not covered by the ARKit face mask (e.g. the neck area under the chin). We therefore use the approximation $\mathbf{A}_j$ as regularization when transferring the personalized blendshape $\mathbf{B}_j$ to the avatar.

This transfer proceeds in two steps (see Figure 3): First, we apply deformation transfer to only the avatar's face using the precomputed correspondence mapping. Second, the vertices in the face area are fixed, and the vertices in the adjacent area (vertices that move in $\mathbf{A}_j$ but do not belong to the face area) are non-rigidly deformed from $\mathbf{A}_j$ [1]. This process seamlessly implants the personalized ARKit blendshapes to the target avatar.

The approximations $\mathbf{A}_j$ also bring facial details not included in the ARKit mesh (eye balls and teeth) to their desired position in the jth personalized avatar blendshape. However, since the ARKit mesh does not include eyeballs, the eyelid blendshapes might intersect them. We eventually repair those artifacts by moving the eyelid vertices to their closest point on the eyeball surface [2].

4 RESULTS

In the following, we present quantitative and qualitative comparisons between our blendshape personalization approach (Section 3.2), example-based facial rigging (*EBFR*, Li et al. [26]), and deformation transfer (*DT*, Sumner and Popović [47]). After that, we show comparisons of our personalized blendshapes to automatic facial retargeting and to manual facial retargeting. *Automatic facial retargeting* refers to the optimization of Equation (4). The resulting blendshape weights are the best fitting combinations of the avatar's initial blendshapes to approximate the personalized ARKit blendshapes. *Manual facial retargeting* refers to a manually optimized mapping of the ARKit blendshape set to the avatar's initial blendshape set. This mapping was hand-crafted by two PhD students in Computer Graphics with sufficient expertise in facial blendshape animation (although not being blendshape artists) and represents our best (manual) effort for retargeting the ARKit blendshapes to the initial avatar's blendshapes.

4.1 Face Mask Personalization

In order to evaluate the accuracy of the different sets of personalized ARKit blendshapes, we asked our subjects to record not only the 52 training face expressions (see Section 3.1), but also 20–30 additional test expressions, for which we record the ARKit face mesh $\mathbf{T}_i$ and the corresponding blendshape weights $w_1, \ldots, w_m$. Using these weights and the blendshapes $\mathbf{B}_1, \ldots, \mathbf{B}_m$ to be evaluated, we compute the root mean square error (RMSE) over the n vertices w.r.t. $\mathbf{T}_i$ as

$$\sqrt{\frac{1}{n}\left\| \mathbf{B}_0 + \sum_{j=1}^{m} w_j \mathbf{B}_j - \mathbf{T}_i \right\|^2}. \tag{5}$$

Our implementation of EBFR only performs the blendshape optimization step from Li et al. [26], since we already know the correct blendshape weights from ARKit. Figure 4 compares the RMSE (5) of different sets of personalized blendshapes produced with DT (blue), EBFR (orange), and our approach (green) for Subject 1. In almost all cases (except expression 19) our method yields the lowest errors. When averaging the RMEs over all expressions and all subjects the RMSE of our method is 58% of the RMSE of EBFR and 45% of the RMSE of DT (see supplementary material).

We also evaluate and compare the maximum error computed over all n vertices, since this measure allows to compare the worst parts of the reconstructed expressions. If only a particular region of the face moves in a test expression, the RMSE would be artificially reduced due to averaging over mostly unchanged vertex positions. Figure 5 shows the maximum reconstruction errors of the different

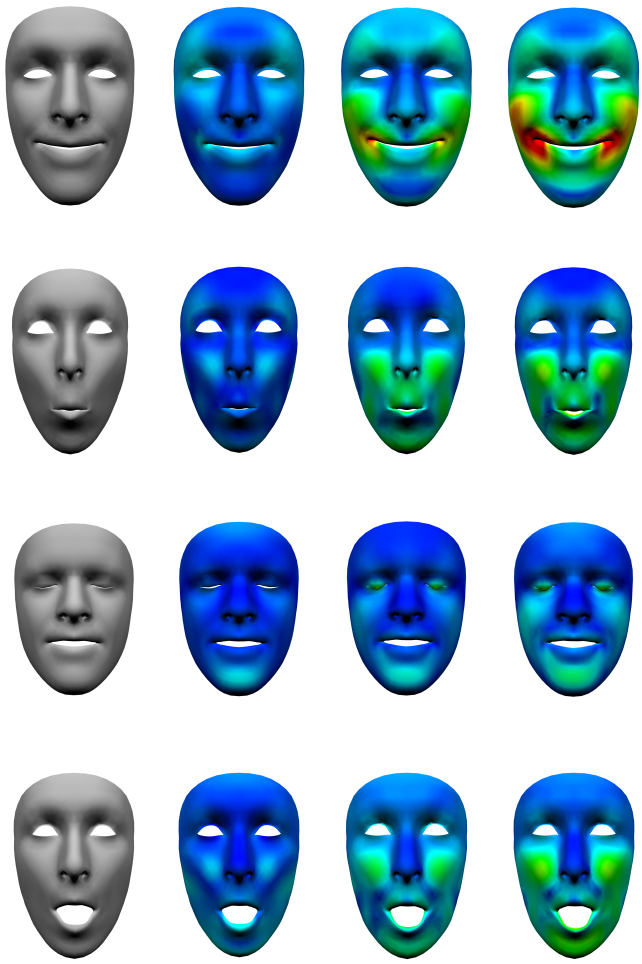

Figure 6: Color-coded reconstruction errors. From left to right: ground truth, our method, example-based facial rigging, deformation transfer. (Blue = 0mm, Red > 5mm)

approaches for Subject 1. It can be seen that DT gives the worst results in all cases. Both EBFR and our method show significant improvements, with our method consistently yielding the lowest error of all three methods. Averaging the maximum error per expression over all test expressions and all subjects, our method reduces the maximum reconstruction error of EBFR and DT down to 50% and 38%, respectively. The color-coding of reconstruction errors in Figure 6 visualizes how the three reconstruction methods differ and shows that our reconstructions are the most accurate. Results for other test subjects are shown in the supplementary material.

4.2 Avatar Blendshape Personalization

The previous section evaluated different approaches for personalizing the ARKit face mask blendshapes. In this section we evaluate the final avatars produced by implanting the personalized ARKit blendshapes through our seamless partial deformation transfer.

Method	Subject 1	Subject 2	Subject 3	Subject 4	Avg.
EBFR	1.585s	1.594s	1.602s	1.594s	1.594s
Ours	0.500s	0.499s	0.499s	0.497s	**0.499s**

Table 1: Computation times of example-based facial rigging (EBFR) and our method, including setting up and solving all required linear systems.

Figure 7 compares the approximations $\mathbf{A}_j$ (computed through automatic facial retargeting using the initial avatar blendshapes) to our final avatars with personalized blendshapes. While the approximation $\mathbf{A}_j$ give reasonable results, the expressions are more accurately reproduced by our personalized blendshapes. This is most noticeable in the mouth region, where the retargeted version does not move the mouth corners far enough or cannot properly reproduce the lip shape of the target expression.

The automatic regargeting is computed by minimzing the approximation error (4) w.r.t. the blendshape weights. Upon inspection the resulting (optimal) weights exhibit many non-zero weights, with several weights even exceeding the range $[0, 1]$. This explains why a manual retargeting is rather unlikely to reproduce the optimal results. Still the manual mapping is the default method for connecting face rigs and face tracking in game engines (see, e.g., the Unity Live Capture Plugin or the Faceware Live Client for Unity).

Figure 8 compares how well different approaches can reproduce some test expressions that have been tracking through ARKit. The manual retargeting using the initial avatar blendshapes yields the worst results, with clearly visible deviations in the mouth region. Automatic retargeting produces more accurate expressions, but the most accurate results are obtained with the personalized blendshapes. In the top row the automatic retargeting does not move the mouth corners far enough to the side, visualized through the blue cross and the green line. In the bottom row the manual and automatic retargeting do not properly close the mouth and eyes, respectively, while the personalized blendshapes do. Comparisons on the full test sequences can be seen in the accompanying video.

4.3 Computation Time

Our automatic, app-guided recording of training expressions takes about 2 minutes. Afterwards, all computations of our pipeline take approximately 35 seconds per avatar. The ARKit face masks consist of 1,220 vertices and 2,304 triangles, the avatar meshes consist of 21k vertices and 42k triangles.

Table 1 shows the computation times for EBFR and our method, measured on a desktop PC with a 10-core 3.6 GHz CPU and a Nvidia RTX 3070 GPU. These timings include setting up the linear systems and solving them through sparse Cholesky factorizations. For EBFR it also includes converting the local frames back to the new blendshape basis, and for our method it includes the computation of the regularization blendshapes using deformation transfer. On average, our method is about 3× faster than EBFR. This is mainly due to the fact that EBFR performs the optimization per triangle, and hence has to solve 2,304 linear least squares problems.

Figure 7: Comparison of the personalized ARKit blendshapes (middle) to an approximation using automatic retargeting using the initial avatar blendshapes (left) and to our final personalized blendshapes (right), where the latter yield more accurate results.

5 DISCUSSION

As described in Section 4.1, personalized blendshapes produce more accurate reconstructions of facial expressions than generic blendshapes transferred from a template rig. Our improved approach further reduces the maximum reconstruction error compared to deformation transfer and example-based facial rigging. Considering Figure 6, the mouth corners of Subject 1 (top row) are closer to the ground truth for our method, while both DT and EBFR lead to a stronger grin expression. Even these slight differences can be problematic, since they might lead to incongruent social cues [42] or create unnatural-appearing facial expressions that do not correctly convey a tracked person's actual look and feelings (Figure 8).

Our approach also has some limitations. Both the blendshape approximation and the seamless partial deformation transfer rely on correspondence mappings, which have to be computed in a preprocess once per template model. Moreover, our seamless partial deformation transfer relies on reasonable initial blendshapes of the avatar, which is used to estimate how the adjacent area of the face (e.g. the neck) deforms during facial expressions. As a consequence, our method cannot be applied to avatars without blendshapes.

6 CONCLUSION

Faithful digital reconstruction of humans has various interesting use-cases. It may become even more prominent in future social XR encounters. Here, an accurate reconstruction of facial expressions is a necessity due to the prominent role of facial expressions in non-verbal behavior and social interaction. This article presented an optimization-based approach to generating personalized blendshapes necessary for a faithful reconstruction of facial expressions and their animation. The proposed method combines a position-based optimization with a seamless partial deformation transfer. It outperforms existing solutions and overall results in a much lower reconstruction error. It also neatly integrates with recent smartphone-based reconstruction pipelines for mesh generation and automated rigging, further paving the way to a widespread application of personalized avatars and agents in various use-cases.

In the future, we would like to use the iPhone's front-facing depth sensor to capture more accurate geometry during expression scanning. Furthermore, we would like to investigate the effect that our improved personalized blendshapes have on the perceptibility of expression semantics.

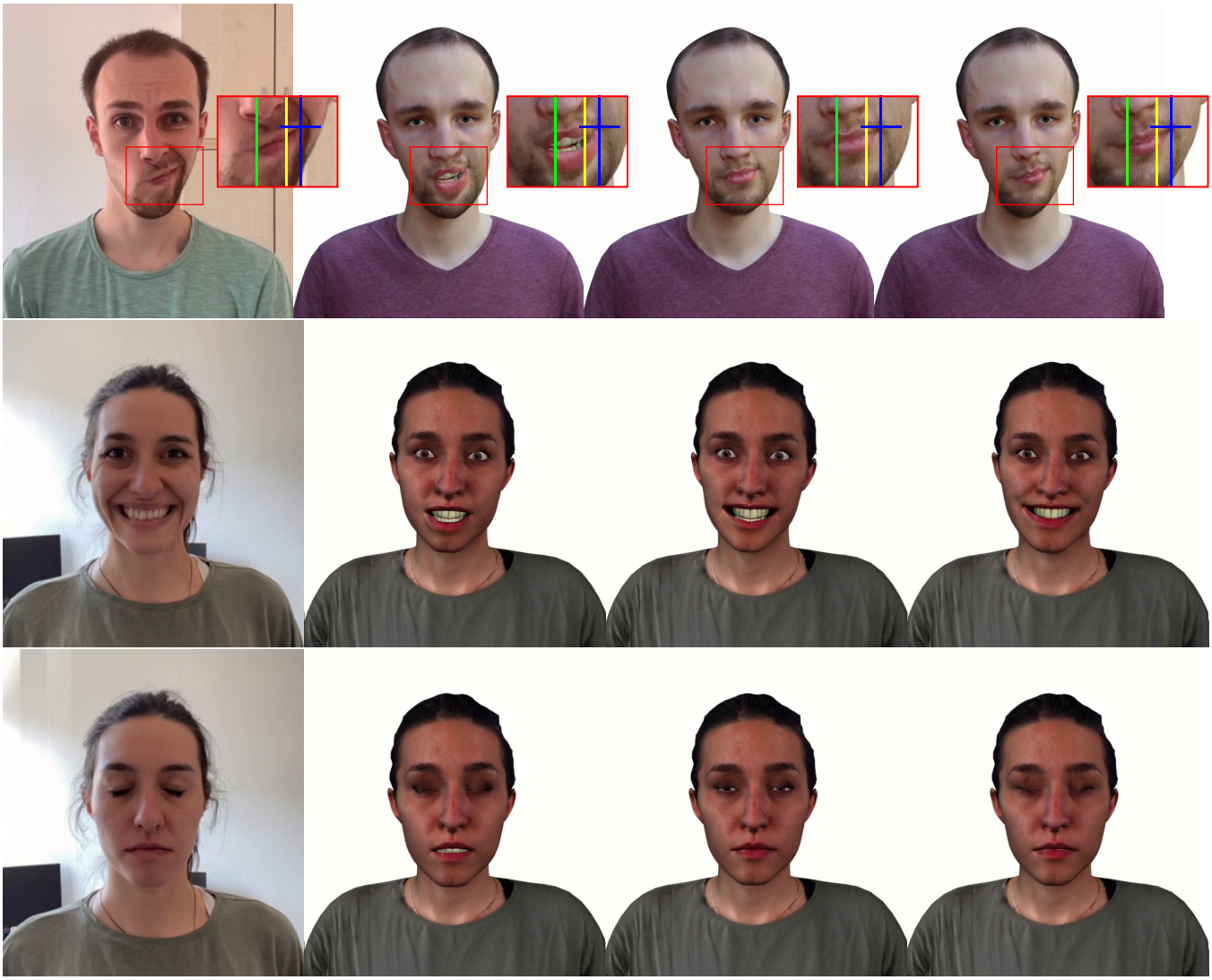

Figure 8: Comparison of the original tracked facial expression with the animated avatars. From left to right: captured image, manual retargeting, automatic retargeting, personalized blendshapes

ACKNOWLEDGMENTS

The authors are very grateful to all scanned subjects. This research was supported by the German Federal Ministry of Education and Research (BMBF) through the project VIA-VR (ID 16SV8446).

REFERENCES

[1] Jascha Achenbach, Thomas Waltemate, Marc Erich Latoschik, and Mario Botsch. 2017. Fast generation of realistic virtual humans. In *Proceedings of the 23rd ACM Symposium on Virtual Reality Software and Technology*. ACM. https://doi.org/10.1145/3139131.3139154

[2] Jascha Achenbach, Eduard Zell, and Mario Botsch. 2015. Accurate Face Reconstruction through Anisotropic Fitting and Eye Correction. In *Vision, Modeling & Visualization*. The Eurographics Association. https://doi.org/10.2312/vmv.20151251

[3] Jim Blascovich and Jeremy Bailenson. 2011. *Infinite Reality: Avatars, Eternal Life, New Worlds, and the Dawn of the Virtual Revolution*. William Morrow & Co. https://doi.org/10.1162/PRES_r_00068

[4] Mario Botsch, Robert Sumner, Mark Pauly, and Markus Gross. 2006. Deformation Transfer for Detail-Preserving Surface Editing. In *Proceedings of VMV 2006*.

[5] Sofien Bouaziz, Andrea Tagliasacci, and Mark Pauly. 2014. Dynamic 2D/3D Registration. In *Eurographics 2014 Tutorial*.

[6] Sofien Bouaziz, Yangang Wang, and Mark Pauly. 2013. Online modeling for realtime facial animation. *ACM Transactions on Graphics* 32, 4 (2013), 1–10. https://doi.org/10.1145/2461912.2461976

[7] David Burden and Maggi Savin-Baden. 2020. *Virtual humans: Today and tomorrow*. Chapman and Hall/CRC.

[8] Chen Cao, Hongzhi Wu, Yanlin Weng, Tianjia Shao, and Kun Zhou. 2016. Real-time facial animation with image-based dynamic avatars. *ACM Transactions on Graphics* 35, 4 (2016), 1–12. https://doi.org/10.1145/2897824.2925873

[9] E. Carrigan, E. Zell, C. Guiard, and R. McDonnell. 2020. Expression Packing: As-Few-As-Possible Training Expressions for Blendshape Transfer. *Computer Graphics Forum* 39, 2 (2020), 219–233. https://doi.org/10.1111/cgf.13925

[10] Dan Casas, Oleg Alexander, Andrew W. Feng, Graham Fyffe, Ryosuke Ichikari, Paul Debevec, Rhuizhe Wang, Evan Suma, and Ari Shapiro. 2015. Rapid Photorealistic Blendshapes from Commodity RGB-D Sensors. In *Proceedings of the 19th Symposium on Interactive 3D Graphics and Games*. ACM, 134–134. https://doi.org/10.1145/2699276.2721398

[11] Justine Cassell, Catherine Pelachaud, Norman Badler, Mark Steedman, Brett Achorn, Tripp Becket, Brett Douville, Scott Prevost, and Matthew Stone. 1994. Animated conversation: rule-based generation of facial expression, gesture & spoken intonation for multiple conversational agents. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*. 413–420. https://doi.org/10.1145/192161.192272

[12] Bindita Chaudhuri, Noranart Vesdapunt, Linda Shapiro, and Baoyuan Wang. 2020. Personalized Face Modeling for Improved Face Reconstruction and Motion Retargeting. In *Computer Vision – ECCV 2020*. Springer International Publishing, 142–160. https://doi.org/10.1007/978-3-030-58558-7_9

[13] Paul Ekman and Wallace V Friesen. 1969. The repertoire of nonverbal behavior: Categories, origins, usage, and coding. *Semiotica* 1, 1 (1969), 49–98. https://doi.org/10.1515/semi.1969.1.1.49

[14] Paul Ekman and Wallace V. Friesen. 1978. Facial Action Coding System. https://doi.org/10.1037/t27734-000

[15] Andrew Feng, Evan Suma Rosenberg, and Ari Shapiro. 2017. Just-in-time, viable, 3-D avatars from scans. *Computer Animation and Virtual Worlds* 28, 3-4 (2017). https://doi.org/10.1145/3084363.3085045

[16] Pablo Garrido, Levi Valgaert, Chenglei Wu, and Christian Theobalt. 2013. Reconstructing detailed dynamic face geometry from monocular video. *ACM Transactions on Graphics* 32, 6 (2013), 1–10. https://doi.org/10.1145/2508363.2508380

[17] Pablo Garrido, Michael Zollhöfer, Dan Casas, Levi Valgaerts, Kiran Varanasi, Patrick Pérez, and Christian Theobalt. 2016. Reconstruction of Personalized 3D Face Rigs from Monocular Video. *ACM Transactions on Graphics* 35, 3 (2016), 1–15. https://doi.org/10.1145/2890493

[18] Ju Hee Han, Jee-In Kim, Hyungseok Kim, and Jang Won Suh. 2021. Generate Individually Optimized Blendshapes. In *2021 IEEE International Conference on Big Data and Smart Computing (BigComp)*. IEEE. https://doi.org/10.1109/bigcomp51126.2021.00030

[19] Roger Blanco i Ribera, Eduard Zell, J. P. Lewis, Junyong Noh, and Mario Botsch. 2017. Facial retargeting with automatic range of motion alignment. *ACM Transactions on Graphics* 36, 4 (2017), 1–12. https://doi.org/10.1145/3072959.3073674

[20] Alexandru Eugen Ichim, Sofien Bouaziz, and Mark Pauly. 2015. Dynamic 3D avatar creation from hand-held video input. *ACM Transactions on Graphics* 34, 4 (2015), 1–14. https://doi.org/10.1145/2766974

[21] Marc Erich Latoschik, Florian Kern, Jan-Philipp Stauffert, Andrea Bartl, Mario Botsch, and Jean-Luc Lugrin. 2019. Not Alone Here?! Scalability and User Experience of Embodied Ambient Crowds in Distributed Social Virtual Reality. *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 25, 5 (2019), 2134–2144. https://doi.org/10.1109/TVCG.2019.2899250

[22] Marc Erich Latoschik, Daniel Roth, Dominik Gall, Jascha Achenbach, Thomas Waltemate, and Mario Botsch. 2017. The Effect of Avatar Realism in Immersive Social Virtual Realities. In *23rd ACM Symposium on Virtual Reality Software and Technology (VRST)*. 39:1–39:10. https://doi.org/10.1145/3139131.3139156

[23] Marc Erich Latoschik and Carolin Wienrich. 2022. Congruence and Plausibility, not Presence?! Pivotal Conditions for XR Experiences and Effects, a Novel Model. *Frontiers in Virtual Reality* (2022). https://doi.org/10.3389/frvir.2022.694433

[24] JP Lewis and Kenichi Anjyo. 2010. Direct Manipulation Blendshapes. *IEEE Computer Graphics and Applications* 30, 4 (2010), 42–50. https://doi.org/10.1109/mcg.2010.41

[25] J. P. Lewis, Ken Anjyo, Taehyun Rhee, Mengjie Zhang, Fred Pighin, and Zhigang Deng. 2014. Practice and Theory of Blendshape Facial Models. In *Eurographics 2014 - State of the Art Reports*. The Eurographics Association. https://doi.org/10.2312/egst.20141042

[26] Hao Li, Thibaut Weise, and Mark Pauly. 2010. Example-based facial rigging. *ACM Transactions on Graphics* 29, 4 (2010), 1–6. https://doi.org/10.1145/1778765.1778769

[27] Hao Li, Jihun Yu, Yuting Ye, and Chris Bregler. 2013. Realtime facial animation with on-the-fly correctives. *ACM Transactions on Graphics* 32, 4 (2013), 1–10. https://doi.org/10.1145/2461912.2462019

[28] Tianye Li, Timo Bolkart, Michael J. Black, Hao Li, and Javier Romero. 2017. Learning a model of facial shape and expression from 4D scans. *ACM Transactions on Graphics* 36, 6 (2017), 1–17. https://doi.org/10.1145/3130800.3130813

[29] Qiaoxi Liu and Anthony Steed. 2021. Social Virtual Reality Platform Comparison and Evaluation Using a Guided Group Walkthrough Method. *Frontiers in Virtual Reality* 2 (2021), p. 52. https://doi.org/10.3389/frvir.2021.668181

[30] Yunpeng Liu, Stephan Beck, Renfang Wang, Jin Li, Huixia Xu, Shijie Yao, Xiaopeng Tong, and Bernd Froehlich. 2015. Hybrid Lossless-Lossy Compression for Real-Time Depth-Sensor Streams in 3D Telepresence Applications. In *Advances in Multimedia Information Processing – PCM 2015*. Springer International Publishing, Cham, pp. 442–452. https://doi.org/10.1007/978-3-319-24075-6_43

[31] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. 2015. SMPL: A Skinned Multi-Person Linear Model. *ACM Transactions on Graphics* 34, 6 (2015), 248:1–248:16. https://doi.org/10.1145/2816795.2818013

[32] Birgit Lugrin, Catherine Pelachaud, and David Traum. 2021. The Handbook on Socially Interactive Agents: 20 years of Research on Embodied Conversational Agents, Intelligent Virtual Agents, and Social Robotics Volume 1: Methods, Behavior, Cognition. (2021). https://doi.org/10.1145/3477322

[33] Nadia Magnenat-Thalmann and Daniel Thalmann. 2005. Virtual humans: thirty years of research, what next? *The Visual Computer* 21, 12 (2005), 997–1015. https://doi.org/10.1007/s00371-005-0363-6

[34] David Matsumoto, Mark G Frank, and Hyi Sung Hwang. 2012. Reading people. *Nonverbal Communication: Science and Applications* (2012), 1. https://doi.org/10.4135/9781452244037

[35] Verónica Costa Orvalho, Ernesto Zacur, and Antonio Susin. 2008. Transferring the Rig and Animations from a Character to Different Face Models. *Computer Graphics Forum* 27, 8 (2008), 1997–2012. https://doi.org/10.1111/j.1467-8659.2008.01187.x

[36] Ahmed A A Osman, Timo Bolkart, and Michael J. Black. 2020. STAR: A Sparse Trained Articulated Human Body Regressor. In *European Conference on Computer Vision (ECCV)*. 598–613. https://doi.org/10.1007/978-3-030-58539-6_36

[37] Catherine Pelachaud. 2009. Modelling multimodal expression of emotion in a virtual agent. *Philosophical Transactions of the Royal Society B: Biological Sciences* 364, 1535 (2009), 3539–3548. https://doi.org/10.1098/rstb.2009.0186

[38] Tekla S Perry. 2015. Virtual reality goes social. *IEEE Spectrum* 53, 1 (2015), 56–57. https://doi.org/10.1109/MSPEC.2016.7367470

[39] Richard A. Roberts, Rafael Kuffner dos Anjos, Akinobu Maejima, and Ken Anjyo. 2021. Deformation transfer survey. *Computers & Graphics* 94 (2021), 52–61. https://doi.org/10.1016/j.cag.2020.10.004

[40] Erika L Rosenberg and Paul Ekman. 2020. *What the face reveals: Basic and applied studies of spontaneous expression using the Facial Action Coding System (FACS)*. Oxford University Press.

[41] Daniel Roth, Carola Bloch, Anne-Kathrin Wilbers, Kai Kaspar, Marc Erich Latoschik, and Gary Bente. 2015. Quantification of Signal Carriers for Emotion Recognition from Body Movement and Facial Affects. *Journal of Eye Movement Research* 4 (2015), p. 192. https://downloads.hci.informatik.uni-wuerzburg.de/2015-ecem-roth-quantification-signal-carriers.pdf

[42] Daniel Roth, Carola Bloch, Anne-Kathrin Wilbers, Marc Erich Latoschik, Kai Kaspar, and Gary Bente. 2016. What You See is What You Get: Channel Dominance in the Decoding of Affective Nonverbal Behavior Displayed by Avatars. In *Presentation at the 66th Annual Conference of the International Communication Association (ICA)*. https://downloads.hci.informatik.uni-wuerzburg.de/2016-Roth-WYSIWYG.pdf

[43] Jun Saito. 2013. Smooth contact-aware facial blendshapes transfer. In *Proceedings of the Symposium on Digital Production - DigiPro '13*. ACM Press, 13–17. https://doi.org/10.1145/2491832.2491836

[44] Yeongho Seol, Wan-Chun Ma, and J. P. Lewis. 2016. Creating an actor-specific facial rig from performance capture. In *Proceedings of the 2016 Symposium on Digital Production*. ACM. https://doi.org/10.1145/2947688.2947693

[45] Yeongho Seol, Jaewoo Seo, Paul Hyunjin Kim, J. P. Lewis, and Junyong Noh. 2011. Artist friendly facial animation retargeting, In Proceedings of the 2011 SIGGRAPH Asia Conference on - SA '11. *ACM Transactions on Graphics*, 1–10. https://doi.org/10.1145/2024156.2024196

[46] Antonio Susín Sergi Villagrasa. 2010. FACe! 3D Facial Animation System based on FACS. *IV Iberoamerican Symposium in Computer Graphics* (2010), 203–209.

[47] Robert W. Sumner and Jovan Popović. 2004. Deformation transfer for triangle meshes. *ACM Transactions on Graphics* 23, 3 (2004), 399–405. https://doi.org/10.1145/1015706.1015736

[48] Justus Thies, Michael Zollhöfer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. 2018. Face2Face: real-time face capture and reenactment of RGB videos. *Commun. ACM* 62, 1 (Dec. 2018), 96–104. https://doi.org/10.1145/3292039

[49] Keith Waters. 1987. A muscle model for animation three-dimensional facial expression. *ACM SIGGRAPH Computer Graphics* 21, 4 (1987), 17–24. https://doi.org/10.1145/37402.37405

[50] Thibaut Weise, Sofien Bouaziz, Hao Li, and Mark Pauly. 2011. Realtime performance-based facial animation. *ACM Transactions on Graphics* 30, 4 (2011), 1–10. https://doi.org/10.1145/2010324.1964972

[51] Stephan Wenninger, Jascha Achenbach, Andrea Bartl, Marc Erich Latoschik, and Mario Botsch. 2020. Realistic Virtual Humans from Smartphone Videos. In *26th ACM Symposium on Virtual Reality Software and Technology*. ACM, 1–11. https://doi.org/10.1145/3385956.3418940

[52] Timo Zinßer, Jochen Schmidt, and Heinrich Niemann. 2005. Point Set Registration with Integrated Scale Estimation. In *International Conference on Pattern Recognition and Image Processing (PRIP 2005)*.