Computers & Graphics (2024)

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

SparseSoftDECA
Efficient High-Resolution Physics-Based Facial Animation from Sparse Landmarks

Wagner Nicolas[a,*], Schwanecke Ulrich[b], Botsch Mario[a]

[a] *TU Dortmund University, Otto-Hahn-Str. 16, 44227 Dortmund, Germany*
[b] *University of Applied Sciences RheinMain, Kurt-Schumacher-Ring 18, 65197 Wiesbaden, Germany*

ARTICLE INFO

Article history:
Received March 7, 2024

Keywords: Facial Animation, Deep Learning, Physics-based Simulation

ABSTRACT

Facial animation on computationally limited systems still heavily relies on linear blendshape models. Nonetheless, these models exhibit common issues like volume loss, self-collisions, and inaccuracies in soft tissue elasticity. Furthermore, personalizing blendshapes models demands significant effort, but there are limited options for simulating or manipulating physical and anatomical characteristics afterwards. Also, second-order dynamics can only be partially represented.

For many years, physics-based facial simulations have been explored as an alternative to linear blendshapes, however, those remain cumbersome to implement and result in a high computational burden. We present a novel deep learning approach that offers the advantages of physics-based facial animations while being effortless and fast to use on top of linear blendshapes. For this, we design an innovative hypernetwork that efficiently approximates a physics-based facial simulation while generalizing over the extensive DECA model of human identities, facial expressions, and a wide range of material properties that can be locally adjusted without re-training.

In addition to our previous work, we also demonstrate how the hypernetwork can be applied to facial animation from a sparse set of tracked landmarks. Unlike before, we no longer require linear blendshapes as the foundation of our system but directly operate on neutral head representations. This application is also used to complement an existing framework for commodity smartphones that already implements high resolution scanning of neutral faces and expression tracking.

1. Introduction

Currently, research in the realm of head avatars and facial animation primarily revolves around achieving photorealistic outcomes using neural networks[1, 2, 3, 4]. These approaches require substantial computational resources for operation. However, a significant challenge lies in accommodating less powerful hardware configurations and scenarios where geometry-based processing is necessary. In such cases, various adaptations of linear blendshape models [5] remain the conventional choice for production.

Despite decades of intensive research and refinement of linear facial models, they still exhibit known limitations, including physically implausible distortions, volume loss, anatomically impossible expressions, the absence of volumetric elasticity, and self-intersections. To address these issues, physics-based simulations have been proposed, which mitigate most ar-

*Corresponding author
e-mail: nicolas.wagner@tu-dortmund.de (Nicolas Wagner)

tifacts associated with linear blendshapes and introduce a range of additional capabilities [6, 7, 8, 9, 10, 11, 12]. Researchers have explored applications in fields such as medicine, involving the visualization of weight changes, paralysis, or surgical procedures, as well as visual effects like aging, zombifications, gravity alterations, and second-order effects. Moreover, recent work has demonstrated that simulations incorporating detailed material information result in significantly more realistic facial animations compared to linear models [10].

However, it is important to note that physics-based facial animation models typically impose a substantial computational burden, leading to a considerable body of literature dedicated to acceleration techniques. Much of this research has focused on evaluating simulations within manually constructed subspaces [13] or learned subspaces [14, 15] and corrective blendshapes [7]. Among these approaches, learned subspace methods have proven to be more versatile and adaptable [14], which is why they have already found successful application in full-body animations [15]. Nevertheless, there is currently no method that effectively extends these advancements in fast physics-based simulations to facial animations. The principal contribution of this work is closing this gap with a deep learning approach which we call SoftDECA.

SoftDECA introduces an innovative neural network designed to animate facial expressions while closely adhering to a dynamic physics-based model. Our approach possesses universal applicability, as it can accommodate a wide range of physics-based facial animations. However, our specific emphasis lies in approximating a combination of cutting-edge anatomically plausible and volumetric finite element methods (FEM) [6, 7, 8, 16]. For this, we propose a novel adaption of hypernetworks [17] which yields inference times of about 10ms on consumer-grade CPUs and has the same programming interface as standard linear blendshapes. More precisely, we train SoftDECA to be applied as an add-on to arbitrary human blendshape rigs that follow the Apple ARKit system .

Furthermore, SoftDECA offers straightforward deployment without the necessity for intricate customizations or retraining efforts due to our extensive compilation of training examples. This comprehensive dataset encompasses a substantial domain of the intended FEM model and amalgamates data from various sources. These sources include CT head scans to capture head anatomy, 3D head reconstructions representing diverse head shapes (utilizing DECA as outlined in [18]), and facial expressions recorded as ARKit blendshape weights from dyadic conversational scenarios. The resulting training dataset ensures SoftDECA's capacity for robust generalization across a spectrum of human identities, facial expressions, and the extensive parameter space of the targeted FEM model. In contrast to earlier methods [14, 15], the ability to generalize across simulation parameters makes extensive and efficient artistic interventions possible, with SoftDECA even supporting localized material adjustments.

As an additional contribution, we present a novel layered head model (LHM) that represents all training instances in a standardized way. Unlike fully or partially tetrahedralized volumetric meshes conventionally used for FEM, the LHM has additional enveloping wraps around bones, muscles, and skin. Based on these wraps, we describe a data-driven fitting procedure that positions muscles and bones within a neutral head while avoiding intersections of the various anatomic structures. A characteristic that was mostly not of concern in previous manually crafted physics-based facial animations but can otherwise lead to numerical instabilities in our automated training data generation approach.

This paper is an extension to the previously presented SoftDECA [19]. Here, we additionally introduce the adapted SparseSoftDECA, which maps sparsely observed facial landmarks into plausible facial expressions with respect to the foundational physics-based simulation. Again, SparseSoftDECA is trained to exhibit a high degree of generalization, accommodating a variety of head shapes and landmark positions. As before, we present a pipeline for generating extensive training data that densely samples the input domains.

The animation via facial landmarks offers the advantage of eliminating the need for blendshape generation entirely. All that is required for animating a person's face is SparseSoftDECA and the neutral head shape which can be easily obtained. For instance, Wenniger et al. [20] have demonstrated the quick acquisition of a neutral head shape in just a few minutes solely based on smartphone videos.

Furthermore, SparseSoftDECA inherently supports personalized animations when facial landmarks can be reliably tracked. Achieving this level of personalization, such as through linear blendshapes, typically demands several of additional scans for each individual.

2. Related Work

2.1. Personalized Anatomical Models

Algorithms for generating personalized anatomical models can be categorized into two main paradigms: *heuristic-based* and *data-driven*. In the realm of heuristic-based approaches, Anatomy Transfer [21] employs a space warp on a template anatomical structure to conform to a target skin surface, deforming the skull and other bones only through an affine transformation. A similar approach is presented by Gilles et al. [22], incorporating statistical validation of bone shapes derived from artificially deformed bones. In both [7] and [23], an inverse physics-based simulation is utilized to reconstruct anatomical structures from multiple 3D expression scans. Saito et al. [24] focus on simulating the growth of soft tissue, muscles, and bones. In [25], a complete musculoskeletal biomechanical model is fitted based on sparse observations, however, no qualitative evaluation is conducted.

Primarily, concerns such as data privacy or potential radiation exposure keep the number of data-driven anatomy fitting approaches small. The recent OSSO method [26, 27] predicts body skeletons from 2000 DXA images. These images do not contain precise 3D information and bones are placed within the body by predicting solely three anchor points per bone group. Additionally, intersections between skin and bones are not resolved. In [28], skin-bones intersections are addressed and also

the musculature is fitted. Instead of fitting anatomical structures directly, encapsulating wraps are placed within a body. However, this approach relies on a BMI regressor rather than accurate medical imaging [29]. Also in [27], skeletons do not intersect but are not placed based on medical imaging either.

A more accurate facial model, developed by Achenbach et al. [30], combines CT scans with optical surface scans using a multilinear model (MLM) that maps between skulls and faces bidirectionally. Despite its accuracy, this model does not prevent self-intersections and solely focuses on fitting bones. Building upon the data from [30] and extending the concept of a layered body model [28], we formulate a statistical layered head model encompassing musculature while mitigating self-intersections.

Fig. 1: All components of the layered head model template $\mathcal{T}$. Skin $S_\mathcal{T}$, skin wrap $\hat{S}_\mathcal{T}$, muscles $M_\mathcal{T}$, muscles wrap $\hat{M}_\mathcal{T}$, skull $B_\mathcal{T}$, and the skull wrap $\hat{B}_\mathcal{T}$.

2.2. Physics-Based Facial Animation

Various paradigms for animating faces have been developed in the past [31, 32, 33, 34]. Dominating the field are data-driven models [5, 7, 35], which have witnessed significant advancements with the application of deep learning techniques [36, 37, 1, 18, 38, 3]. Linear blendshapes [5] remain prevalent in demanding applications and scenarios lacking computationally rich hardware due to their simplicity and speed. Physics-based simulations, although addressing issues of blendshape models like implausible contortions and self-intersections, are less commonly used due to their inherent complexity and computational demands. Sifakis et al.'s [39] pioneering work represents the first fully physics-based volumetric facial animation, employing a personalized tetrahedron mesh with limited resolution due to an involved dense optimization problem. The Phace system [6] successfully overcame this limitation through an improved simulation. Art-directed physics-based facial animations additionally employ a muscle representation based on B-splines [16, 40, 8]. Animations can then be controlled via trajectories of spline control points. A solely inverse model for determining physical properties of faces is presented in [41].

Hybrid methodologies incorporate surface-based physics into linear blendshapes to enhance the intricacy of facial expressions [11, 42, 9, 43]. Nevertheless, due to their design, these approaches are unable to represent volumetric effects. The introduction of volumetric blendshapes [7] represents a hybrid solution that amalgamates the structure of linear blendshapes with volumetric physical and anatomical plausibility. However, achieving real-time performance necessitates the utilization of extensive personalized corrective blendshapes.

Considering soft bodies in general, deep learning approaches have been investigated to approximate physics-based simulations. For instance, in [15, 44] the SMPL (Skinned Multi-Person Linear Model) proposed in [45] was extended with secondary motion. Recently, [12, 10, 9] developed methods to learn the particular physical properties of objects and faces. However, these approaches must be retrained for unseen identities and are slow in inference. A fast and general approach for learning physics-based simulations is introduced in [14]. Unfortunately, they focused on reflecting the dynamics of single objects with limited complexity. We present a real-time capable deep learning approach to physics-based facial animations that does not need to be retrained and maintains the control structure of standard linear blendshapes. Additionally, none of the previously described deep learning methods tackle the challenging creation of facial training data, which we also address in this work.

3. Method

The cornerstone of the SoftDECA animation system lies in a novel layered head representation (Section 3.1). Building upon this foundation, we formulate a physics-based facial animation system (Sections 3.2 & 3.3) and illustrate how to distill it into a defining dataset (Section 3.4). Utilizing this dataset, we train a newly devised hypernetwork (Section 3.5) capable of real-time approximation of the animation system. In addition to our previous work [19], we enhance SoftDECA to be directly addressable by sparse landmarks, rendering it entirely independent of linear blendshapes if desired (Section 3.6).

3.1. Layered Head Model

3.1.1. Structure

We define a head $\mathcal{H} = \rho_\mathcal{H}(\mathcal{T})$ with a neutral expression through a component-wise transformation $\rho_\mathcal{H}$ of a layered head model template

$$\mathcal{T} = \left(S_\mathcal{T}, B_\mathcal{T}, M_\mathcal{T}, \hat{S}_\mathcal{T}, \hat{B}_\mathcal{T}, \hat{M}_\mathcal{T}\right), \tag{1}$$

comprising six triangle meshes. $S_\mathcal{T}$ delineates the skin surface, encompassing the eyes, mouth cavity, and tongue. $B_\mathcal{T}$ denotes the surface of all skull bones including the teeth. $M_\mathcal{T}$ represents the surface of all muscles, along with the cartilages of the ears and nose. $\hat{S}_\mathcal{T}$ is the skin wrap, i.e. a closed wrap that envelopes $S_\mathcal{T}$. $\hat{B}_\mathcal{T}$ is the skull wrap that encloses $B_\mathcal{T}$ and $\hat{M}_\mathcal{T}$ is the muscle wrap that encloses $M_\mathcal{T}$. For simplicity, other anatomical structures are omitted. The template structures $S_\mathcal{T}, B_\mathcal{T}$, and $M_\mathcal{T}$ were artistically designed, while the skin, skull, and muscle wraps $\hat{S}_\mathcal{T}, \hat{B}_\mathcal{T}$, and $\hat{M}_\mathcal{T}$ were generated by shrink-wrapping the same sphere as closely as possible to the corresponding surfaces without intersections. The complete template is depicted in Figure 1.

The shared triangulation among the wraps of the LHM allows to also define a soft tissue tetrahedron mesh $\mathbb{S}_\mathcal{T}$ (between the skin and muscle wraps) and a muscle tissue tetrahedron mesh $\mathbb{M}_\mathcal{T}$ (between the muscle and skull wraps). For this

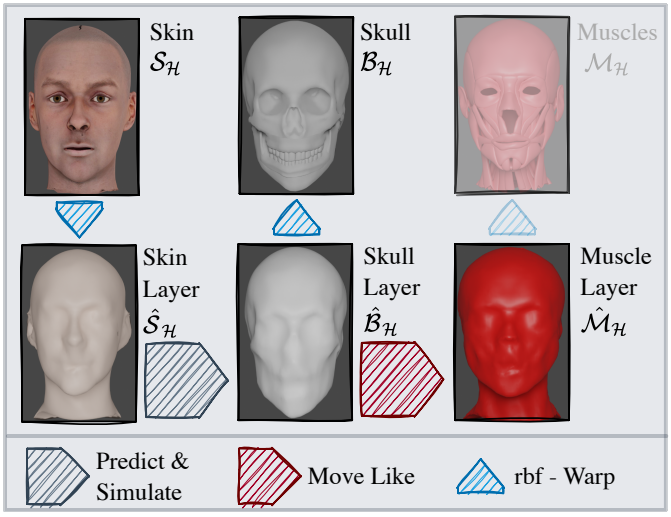

Fig. 2: a) Procedural overview of the layered head model fitting algorithm.

purpose, each triangular prism spanned between corresponding wrap faces is canonically split into three tetrahedra. The complexities of all template components are detailed in the appendix. In the subsequent sections, we denote the number of vertices in a mesh as $|\cdot|_v$ and the number of faces as $|\cdot|_f$.

3.1.2. *Fitting*

Later on, generating training data involves determining

$$\left(S, B, M, \hat{S}, \hat{B}, \hat{M}\right) = \rho_\mathcal{H}(\mathcal{T}) \tag{2}$$

when only the skin surface S of the head $\mathcal{H}$ is known. For this purpose, we employ a hybrid approach that places the skull wrap in a data-driven manner, while the remaining template components are fitted using heuristics to ensure anatomical plausibility and avoid self-intersections.

Starting with the fitting of the skin wrap, we set

$$\hat{S} = \mathrm{rbf}_{S_\mathcal{T} \to S}\left(\hat{S}_\mathcal{T}\right). \tag{3}$$

Here, the RBF function denotes a space warp based on triharmonic radial basis functions [46], calculated from the template skin surface $S_\mathcal{T}$ to the target S and applied to the template skin wrap $\hat{S}_\mathcal{T}$. Due to the construction of RBFs, the skin wrap undergoes a semantically consistent warp, adhering closely to the targeted skin surface.

Following, we fit the skull wrap $\hat{B}$ by first evaluating a linear regressor D that predicts distances from the vertices of $\hat{S}$ to the corresponding vertices of $\hat{B}$. Then, we minimize with projective dynamics [47]

$$\begin{aligned}\arg\min_X\, & w_\mathrm{rect} E_\mathrm{rect}\left(X,\ \hat{S}_\mathcal{T}\right) + w_{\mathrm{dist}_2} E_{\mathrm{dist}_2}\left(X,\ \hat{S},\ D\left(\hat{S}\right)\right) \\ & + w_\mathrm{curv} E_\mathrm{curv}\left(X,\ \hat{B}_\mathcal{T}\right).\end{aligned} \tag{4}$$

In this optimization, E_{dist_2} ensures the adherence to the predicted distances, E_curv represents a curvature regularization for the skull wrap, and E_rect prevents shearing between corresponding faces of the skin and skull wraps. The distances are set to a minimum value if they fall below a threshold, thereby preventing skin-skull intersections. For formal descriptions of the energy components, please refer to the appendix. The optimization is initialized with $X = \hat{S} - D(S) \cdot n(\hat{S})$, where $n(\hat{S})$ denotes area-weighted vertex normals. The linear regressor D is trained on the dataset from [48] (SKULLS), which correlates CT skull measurements with optical skin surface scans. For a visual illustration of the training process of the linear regressor please refer to Wagner et al. [19].

The muscle wrap $\hat{M}$ is placed almost at the same absolute distances between corresponding vertices of the skin and skull wraps as in the template. Only ten percent of the relative distance changes compared to the template are incorporated, assuming that the muscle mass in the facial area is only moderately influenced by body weight and skull size.

The skull mesh is placed by setting

$$B = \mathrm{rbf}_{\hat{B}_\mathcal{T} \to \hat{B}}(B_\mathcal{T}). \tag{5}$$

The characteristics of the RBF space warp ensure that the skull mesh remains enclosed within the skull wrap, provided the wrap has sufficient resolution. While the muscle mesh could be positioned similarly, it is not utilized further in our pipeline.

Finally, the tetrahedron meshes representing soft and muscle tissue $\mathbb{S}$ and $\mathbb{M}$ are constructed as described before. On average, the complete fitting pipeline takes about 500ms on an AMD Threadripper Pro 3995wx processor. Figure 2 visualizes the overall fitting process.

3.2. *SoftDECA Animation System*

Building upon the LHM representation, we now introduce the SoftDECA animation system by, first, revisiting the concept of linear blendshapes. Subsequently, we derive the dynamic physics-based facial simulation system, which forms the core of SoftDECA.

In a linear blendshape model, n surface blendshapes

$$\left\{S^i\right\}_{i=1}^n \tag{6}$$

animate a facial expression S_t as a linear combination

$$S_t = \sum_{i=1}^N \mathbf{w}_t^i S^i, \tag{7}$$

where the blending weights $\mathbf{w}_t$ determine the contribution of each blendshape to the expression at frame t.

To achieve the same animation with a physics-based model ϕ, one typically employs either forward or inverse simulations. Without loss of generality, we consider inverse simulations in the following. Here, the expression S_t is converted into the (in the Euclidean sense) closest $\phi-$plausible solution by $\phi^\dagger$ to

$$T_t = \phi^\dagger(S_t,\ \mathbf{p}), \tag{8}$$

where $\mathbf{p}$ is a vector of material and simulation parameters on which ϕ depends. For including second-order effects as well, Equation (8) expands to

$$T_t = \phi^\dagger(\gamma\, S_t + 2\alpha\, T_{t-1} - \beta\, T_{t-2},\ \mathbf{p}). \tag{9}$$

The SoftDECA animation system operates in a similar manner, but the right-hand side of Equation 9 is approximated by a computationally efficient neural network f.

Ensuing, we will elucidate our implementation of $\phi^\dagger$ and the process of generating representative examples. However, please note that SoftDECA is not confined to a specific implementation of $\phi^\dagger$.

3.3. Physics-Based Simulations

We implement anatomically plausible inverse physics $\phi^\dagger$ as a projective dynamics energy $E_{\phi^\dagger}$. At this, state-of-the-art FEM models [8, 6, 41] are merged by applying separate terms for soft tissue, muscle tissue, the skin, the skull, and auxiliary components.

3.3.1. Energy

Considering the soft tissue $\mathbb{S}$, we closely follow the model of [6] and impose

$$E_{\mathbb{S}} = w_{\text{vol}} \sum_{\mathsf{t}\in\mathbb{S}} E_{\text{vol}}(\mathsf{t}) + w_{\text{str}} \sum_{\mathsf{t}\in\mathbb{S}} \mathbb{1}_{\sigma_{F(\mathsf{t})}>\epsilon} E_{\text{str}}(\mathsf{t}), \tag{10}$$

which for each tetrahedron t penalizes change of volume and strain, respectively. Strain is only accounted for if the largest eigenvalue $\sigma_{F(\mathsf{t})}$ of the stretching component of the deformation gradient $F(\mathsf{t}) \in \mathbb{R}^{3\times3}$ grows beyond ϵ.

To reflect the biological structure of the skin, we additionally formulate a dedicated strain energy

$$E_S = \sum_{\mathsf{t}\in S} E_{\text{str}}(\mathsf{t}) \tag{11}$$

on each triangle t of the skin which, to the best of our knowledge, has not been done before.

For the muscle tetrahedra $\mathbb{M}$, we follow Kadleček et al. [41] that capturing fiber directions for tetrahedralized muscles is in general too restrictive. Hence, only a volume-preservation term

$$E_{\mathbb{M}} = w_{\text{vol}} \sum_{\mathsf{t}\in\mathbb{M}} E_{\text{vol}}(\mathsf{t}) \tag{12}$$

is applied for each tetrahedron in $\mathbb{M}$.

The skull is not tetrahedralized as it is assumed to be non-deformable even though it is rigidly movable. The non-deformability of the skull is represented by

$$E_B = \sum_{\mathsf{t}\in B} E_{\text{str}}(\mathsf{t}) + \sum_{\mathsf{x}\in B} E_{\text{curv}}(\mathsf{x}, B)\,, \tag{13}$$

i.e. a strain E_{str} on the triangles t and mean curvature regularization on the vertices x of the skull B. We do not model the non-deformability as a rigidity constraint due to the significantly higher computational burden.

To connect the muscle tetrahedra as well as the eyes to the skull, connecting tetrahedra are introduced similar to the sliding constraints in [6]. For the muscle tetrahedra , each skull vertex connects to the closest three vertices in $\mathbb{M}$ to form a connecting tet. For the eyes, connecting tetrahedra are formed by connecting each eye vertex to the three closest vertices in B. On these connecting tetrahedra , the energy E_{con} with the same constraints as in Equation (10) is imposed. By this design, the jaw and the cranium are moved independently from each other through muscle activations but the eyes remain rigid and move only with the cranium.

Finally, the energy

$$E_{\text{inv}} = \sum_{\mathsf{x}\in S} E_{\text{tar}}(\mathsf{x},\, S_t) \tag{14}$$

of soft Dirichlet constraints is added, attracting the skin surface S vertices to the targeted expression S_t.

The weighted sum of the aforementioned energies gives the total energy

$$\begin{aligned} E_{\phi^\dagger} = {} & w_{\mathbb{S}} E_{\mathbb{S}} + w_{\mathbb{M}} E_{\mathbb{M}} + w_{\text{B}} E_B + w_{\text{mstr}} E_{\text{mstr}} \\ & + w_{\text{S}} E_S + w_{\text{con}} E_{\text{con}} + w_{\text{inv}} E_{\text{inv}} \end{aligned} \tag{15}$$

of the inverse model $\phi^\dagger$. Altogether, $\phi^\dagger$ results in an expression T_t that in a Euclidean sense is close to the target S_t but is plausible w.r.t. the imposed constraints.

3.3.2. Collisions

Finally, self-intersections are resolved between colliding lips or teeth in a subsequent projective dynamics update as in [49]. The decisive characteristic of this approach is that no gaps can occur after the resolution of self intersections. For example, in the case of a lip collision, the corresponding lower and upper lip points are simulated to the same position.

3.3.3. Parameters

The construction of $\phi^\dagger$ also implies parts of the parameter vector $\mathbf{p}$. As such, the dynamics parameters α, β, γ, weights w_* of all the constraints, but also other attributes of the constraints are considered. For example, the target volume in E_{vol} or scaling factors of the skull bones are included. We also add constant external forces like gravity strength and direction into $\mathbf{p}$. An overview of all parameters we use and the corresponding value ranges is given in the appendix.

3.4. Training Data

According to the definition of the animation system in Equation (9), a comprehensive training dataset $\mathcal{D}$ should include examples that link various facial expressions generated through linear blendshapes to the corresponding surfaces conforming to ϕ. Moreover, to encompass dynamic effects, the exemplary facial expressions should form coherent sequences. This dataset also needs to encompass a range of diverse head shapes and simulation parameters.

In the following, we describe a pipeline for creating instances of such a dataset, which can be roughly divided into six high-level steps.

1. We commence by randomly selecting a neutral skin surface S from DECA [18], an extensive high-resolution face model. Specifically, we pick an image at random from the Flickr-Faces-HQ [50] dataset and let DECA determine the corresponding neutral head shape along with a latent representation $\mathbf{h}$.

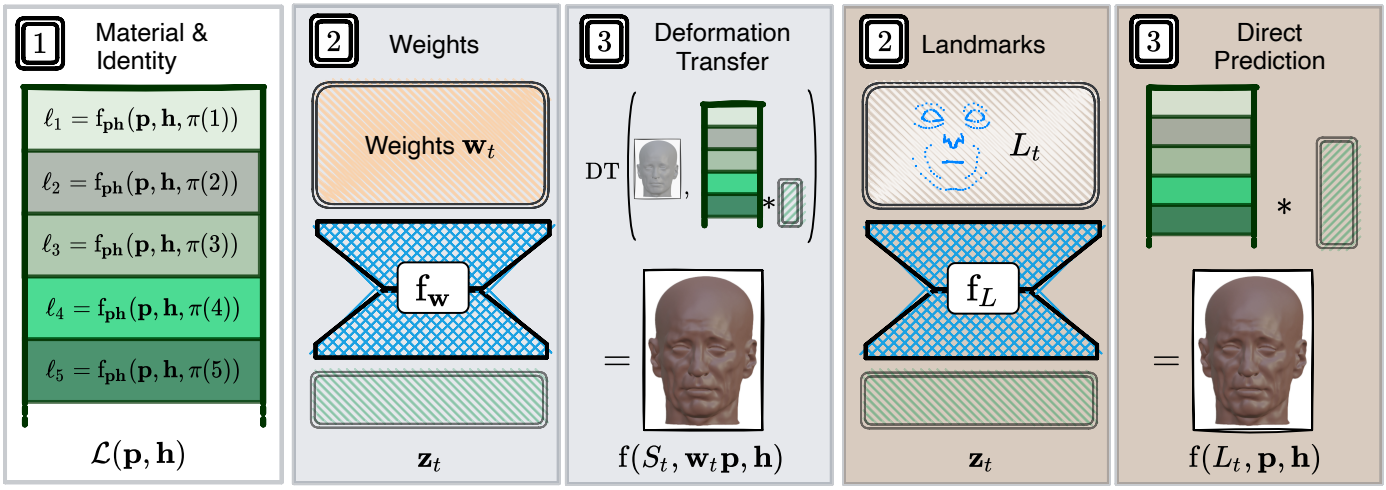

Fig. 3: An overview of SoftDECA and SparseSoftDECA facial animations. In Step 1), for both, the hyper-tensor and the dynamic parameters are determined once for an animation. Subsequently, steps 2-3 are repeatedly evaluated per frame and either map blendshapes weights to deformation gradients (SoftDECA) or landmarks to vertex position (SparseSoftDECA).

2. The template LHM $\mathcal{T}$ is aligned with the skin surface S as described in Section 3.1.
3. Deformation transfer [51] is applied to map ARKit surface-based blendshapes to S.
4. An expression sequence $\mathbf{S} = (S_t)_{t=0}^m$ of length $m+1$ is generated by applying a sequence of linear blendshape weights $\mathbf{w} = (\mathbf{w}_t)_{t=0}^m$. These blendshape weights are derived from 8 approximately 10 minutes long dyadic conversations recorded using a custom iOS app.
5. As the final step before generating the ϕ-plausible counterpart of $\mathbf{S}$, it is necessary to sample simulation parameters within appropriate domains. We expect the user to specify lower and upper bounds for continuous parameter beforehand. Then, for each continuous entry in $\mathbf{p}$, a value is independently sampled from a uniform distribution between the specified bounds. Discrete parameters are treated similarly, without specific constraints.
6. Finally, $\mathbf{T} = \left(\phi^\dagger(S_t, \mathbf{p})\right)_{t=0}^m$ is computed and $(\mathbf{T}, \mathbf{S}, \mathbf{w}, \mathbf{p}, \mathbf{h})$ is appended to $\mathcal{D}$. Evaluating one time step takes approximately 10 seconds on an AMD Threadripper Pro 3995wx.

3.5. Hypernetwork

3.5.1. Architecture & Training

Having training data, we can now design a computationally efficient neural network f to approximate the physics-based simulation from Equation 9. Irrespective of a particular architecture, the training goal implied by $\mathcal{D}$ is to optimize on each frame

$$\min_{\mathrm{f}} \sum_{(\mathbf{T},\mathbf{S},\mathbf{w},\mathbf{p},\mathbf{h})\in\mathcal{D}} \sum_{t=0}^{m} \|T_t - \mathrm{f}(S_t, \mathbf{w}_t, \mathbf{p}, \mathbf{h})\|_2 \,. \tag{16}$$

In words, f is trained to approximate the ϕ-conformal expressions from the the linearly blended expressions S_t, the blending weights $\mathbf{w}_t$, simulation parameters $\mathbf{p}$, and the head descriptions $\mathbf{h}$. Hence, leaving out dynamic effects to begin with, the probably most naive approach would be to learn f to directly predict vertex positions. However, this would not allow the usage of personalized blendshapes at inference time that have not been used in the curation of $\mathcal{D}$. Therefore, we separate f into two high-level components

$$\mathrm{f}(S_t, \mathbf{w}_t, \mathbf{p}, \mathbf{h}) = \mathrm{DT}(S_t, \mathrm{f}_{DG}(\mathbf{w}_t, \mathbf{p}, \mathbf{h}))\,, \tag{17}$$

where DT is a deformation transfer function as in [52] that applies 3×3 per-face deformation gradients (DGs) predicted by $\mathrm{f}_{DG}(\mathbf{w}_t, \mathbf{p}, \mathbf{h}) \in \mathbb{R}^{|S|_f\times 9}$ to the linearly blended S_t. By doing so, f can also be applied to a facial expression S_t which has been formed by unseen personalized blendshapes while still achieving close approximations of $\phi^\dagger$. Fortunately, the evaluation of DT is not more than efficiently finding a solution to a pre-factorized linear equation system.

To implement the DG prediction network f_{DG}, we evaluated multiple network architectures such as set transformers [53], convolutional networks on geometry images, graph neural networks [54], or implicit architectures [55], but all have exhibited substantially slower inference speeds while reaching a similar accuracy as a multi-layer perceptron (MLP). Nevertheless, a plain MLP does not discriminate between inputs that change per frame t and inputs that have to be computed only once. Therefore, we propose an adaptation of a hypernetwork MLP [17] to implement f_{DG} in which the conditioning of f_{DG} with respect to the simulation parameters as well as the DECA identity is done by manipulating network parameters. Formally, we implement

$$\mathrm{f}_{DG}(\mathbf{w}_t, \mathbf{p}, \mathbf{h}) = \mathbf{z}_t\mathcal{L}(\mathbf{p}, \mathbf{h}), \tag{18}$$

where $\mathcal{L}(\mathbf{p}, \mathbf{h}) \in \mathbb{R}^{32\times|S|_f\times 9}$ returns a tensor that only has to be calculated once for all frames and $\mathbf{z}_t = \mathrm{f}_{\mathbf{w}}(\mathbf{w}_t) \in \mathbb{R}^{32}$ is the result of a small standard MLP that processes the blending weights at every frame t. Each matrix $\ell_i \in \mathbb{R}^{32\times 9}$ in $\mathcal{L}(\mathbf{p}, \mathbf{h})$ corresponds to a face in S and the entries are calculated as

$$\ell_i = \mathrm{f_{ph}}(\mathbf{p}, \mathbf{h}, \pi(i))\,. \tag{19}$$

Again, $\mathrm{f}_{\mathbf{ph}}$ is a small MLP and π is a trainable positional encoding. Please consult the appendix for detailed dimensions of all networks and see Figure 3 for a structural overview of f.

3.5.2. Localization

The architecture described above offers extensive possibilities for artistic user interventions at inference time. For instance, different simulation parameters $\mathbf{p}_i$ can be used per triangle i by changing Equation (19) to

$$\ell_i = \mathrm{f}_{\mathbf{ph}}(\mathbf{p}_i, \mathbf{h}, \pi(i)), \tag{20}$$

which enables a localized application of different material models. The DT function ensures that the models are smoothly combined.

3.5.3. Dynamics

Given that locally differing simulation parameters are not reflected in the training data, existing approaches to integrate dynamics in deep learning [14, 15], cannot be adopted. Therefore, we again use the hypernetwork concept to achieve a piecewise-linear dynamics approximation. More precisely, we recursively extend f to

$$\begin{aligned}\mathrm{f}(S_t, \mathbf{w}_t, \mathbf{p}, \mathbf{h}) = \quad & \boldsymbol{\gamma} \odot \mathrm{DT}(S_t, \mathrm{f}_{DG}(\mathbf{w}_t, \mathbf{p}, \mathbf{h})) \\ & + 2\boldsymbol{\alpha} \odot \mathrm{f}(S_{t-1}, \mathbf{w}_{t-1}, \mathbf{p}, \mathbf{h}) \\ & - \boldsymbol{\beta} \odot \mathrm{f}(S_{t-2}, \mathbf{w}_{t-2}, \mathbf{p}, \mathbf{h}),\end{aligned} \tag{21}$$

where $\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma} \in \mathbb{R}^{32 \times |S|_v}$ contain per-vertex dynamics parameters. The first row of Equation (21) is the same as in Equation (17) but the second and third rows allow for dependencies on the previous two frames. Each entry of $\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}$ is calculated as in Equation (20) but with dedicated MLPs $\mathrm{f}_\alpha, \mathrm{f}_\beta, \mathrm{f}_\gamma$. As a result, $\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}$ are again not time-dependent and only have to be calculated once.

3.6. Sparse Animation Control

Previously, we assumed that SoftDECA is supposed to map an expression S_t generated by linear blendshapes (Equation (7)) into a $\phi^\dagger$-plausible expression T_t (Equation (8)). In the following, we now assume that only temporally consistent landmarks $L_t \in S_t$ can be observed per frame t. At the same time, we no longer require S_t to be derived from a specific linear blendshape system for inference. We refer to the *adapted* variant which processes landmarks instead of blendshape weights as SparseSoftDECA. In other words, SparseSoftDECA can create personalized animations from tracked landmarks requiring only a neutral scan as input. In this section, we first explain the adaptation of the physics model to the sparse input. Subsequently, which training data is required for SparseSoftDECA is discussed. Finally, we described changes in the hypernetwork topology of SoftDECA to allow landmarks to be used as input.

3.6.1. Adapted Physics-Based Simulation

The foundation of SparseSoftDECA is a modified physics-based model $\varphi^\dagger$ which in principle optimizes the same energy as $\phi^\dagger$. However, the targeted landmarks are enforced by simultaneously optimizing for

$$E_{\mathrm{lmk}} = \sum_{\mathsf{x} \in L} E_{\mathrm{tar}}(\mathsf{x}, L_t). \tag{22}$$

In our experiments, it has proven beneficial to keep the previous target energy E_{inv} as a regularization term. Otherwise, since L_t is usually only a sparse observation of S_t, i.e. $|L|_v \ll |S|_v$, solely non-uniformly distributed actuation signals would act in $\varphi^\dagger$ which would cause distortions.

In summary, $\varphi^\dagger$ is composed by the overall energy

$$\begin{aligned}E_{\varphi^\dagger} = \; & w_{\mathbb{S}} E_{\mathbb{S}} + w_{\mathbb{M}} E_{\mathbb{M}} + w_{\mathrm{B}} E_B + w_{\mathrm{mstr}} E_{\mathrm{mstr}} \\ & + w_{\mathrm{S}} E_S + w_{\mathrm{con}} E_{\mathrm{con}} \\ & + w_{\mathrm{reg}} E_{\mathrm{inv}} + w_{\mathrm{lmk}} E_{\mathrm{lmk}},\end{aligned} \tag{23}$$

where w_{reg} indicates the strength of the regularization and is included in the parameter vector $\mathbf{p}$.

3.6.2. Adapted Training Data

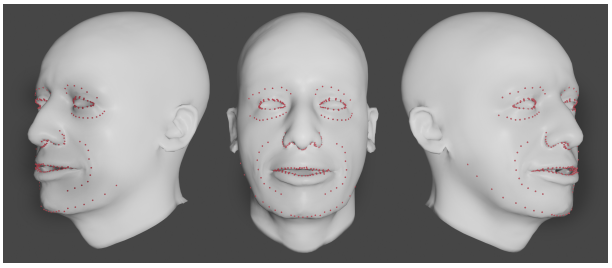

Fig. 4: The set of landmarks used for SparseSoftDECA.

To generate training data for SparseSoftDECA we, basically follow the same data generation pipeline as described in Section 3.4. Merely the steps 4 and 6 must be adjusted to produce training instances with landmarks rather than blendshape weights.

Concerning step 4, we have extended the custom iOS app such that not only weight vector $\mathbf{w}_t$ but also about 150 corresponding landmarks L_t are captured by Apple's ARKit. These landmarks mainly represent the contours of a face and are visualized in Figure 4. Contrary to the blendshape weights, the captured landmarks are tailored to the recorded head.

Concerning step 6, a training instance is now formed as $(\mathbf{T}, \mathbf{S}, \mathbf{L}, \mathbf{p}, \mathbf{h})$ where

$$\begin{aligned}\mathbf{L} &= (\sigma(L_t))_{t=0}^m, \\ \mathbf{T} &= (\varphi^\dagger(\sigma(L_t), S_t, \mathbf{p}))_{t=0}^m.\end{aligned} \tag{24}$$

Here, σ is an augmentation function which serves two purposes. On the one hand, the landmarks must be personalized to account for the difference between the recorded and simulated head shape S drawn in Step 1 of the data generation pipeline. On the other hand, the notably larger domain as opposed to the blendshape weights requires a denser sampling in the training set, as we will show empirically in Section 4.3. Therefore, σ is composed of a deformation transfer [52] that accomplishes the personalization followed by a coordinate-wise Gaussian noise to achieve a robust domain coverage.

3.6.3. Adapted Hypernetwork

For SparseSoftDECA, the efficient hypernetwork topology presented earlier for SoftDECA (Section 3.5) is fundamentally preserved. However, so far, SoftDECA focused on deforming a linear blended surface according to specified material properties. Since SparseSoftDECA is intended to reconstruct a facial expressions without being tied to a particular linear blendshape system, neither the linear blended surface S_t nor the blendshape weights w_t can be utilized as input for the adapted hypernetwork. For the same reason, mesh coordinates can be predicted directly without the intermediate step of forming and resolving deformation gradients. Formally, the static hypernetwork f of SparseSoftDECA is implemented as

$$\mathrm{f}(L_t, \mathbf{p}, \mathbf{h}) = \mathrm{f}_L(L_t)\mathcal{L}(\mathbf{p}, \mathbf{h}), \tag{25}$$

where $\mathcal{L}(\mathbf{p}, \mathbf{h}) \in \mathbb{R}^{32\times|S|_v\times 3}$ returns a tensor that only has to be calculated once for all frames and $\mathrm{f}_L(L_t) \in \mathbb{R}^{32}$ is the result of a small standard MLP that processes the landmarks at every frame t. The dynamic variant is derived as before in Equation (21). A structural overview is given in Figure 3.

3.7. Personalized Animation From Commodity Smartphones

We will release SparseSoftDECA trained on the skin topology used in Wenninger et al. [20]. In their work, they demonstrate how to quickly create high-resolution (face) avatars from a single smartphone video. Combining both the high resolution avatars and our models allows for computationally efficient realistic facial animation with real-time tracking even on low budget hardware. Due to the compatibility with ARKit and software based thereon, SoftDECA and SparseSoftDECA can readily be deployed in environments from Apple, Unity, and many more.

4. Experiments

Prior to outlining the accuracy and efficiency of SoftDECA (Section 4.2), we first evaluate the precision of the LHM fitting (Section 4.1). Afterwards, we examine both quantitatively and qualitatively SparseSoftDECA (Section 4.3).

4.1. LHM Fitting

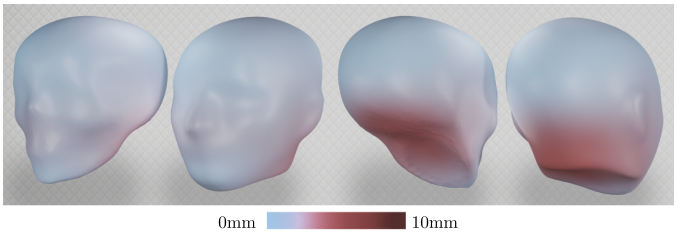

Fig. 5: The per-vertex mean L2-error of the LHM fitting.

The fitting process of the LHM involves the data-driven positioning of the skull wrap and the subsequent heuristic fitting of the muscle wrap. Our evaluation focuses on the critical fitting of the skull wrap using the CT SKULLS dataset from [48], consisting of 43 instances. To assess precision, a leave-one-out validation is conducted, measuring vertex-wise L2 errors.

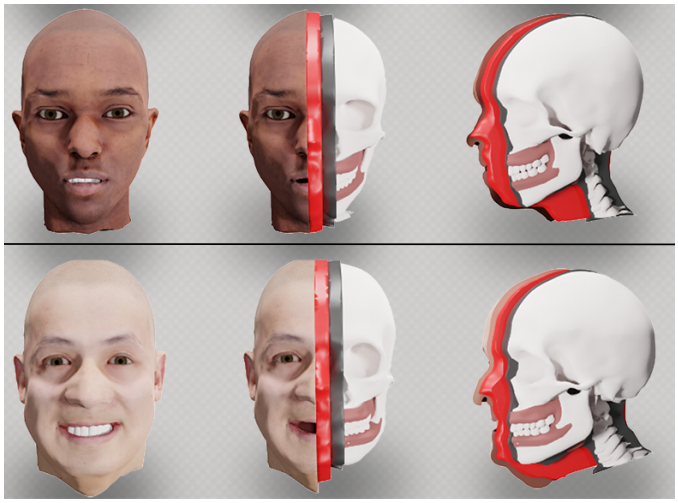
Fig. 6: Exemplary fits of the LHM components skull wrap, muscle wrap, and skull.

Prior methods positioning the skull within the head primarily rely on sparse soft tissue statistics derived from a few points on the skull [7, 56]. We evaluate our approach against the multilinear model (MLM) proposed by Achenbach et al. [30, 48] which demonstrated more robust and precise positioning through the capture of dense soft tissue statistics represented as radii of spheres surrounding the skull.

Both models fall short of achieving medical-grade positioning, exhibiting errors ranging between approximately 2 mm and 4 mm. The MLM demonstrates higher precision with a mean error of 1.98 mm, surpassing our approach, which positions the skull with an average error of 3.83 mm. However, the MLM lacks collision prevention, posing a potential issue for physics-based simulations. Moreover, our fitting algorithm produces significant errors primarily in regions of lesser importance for facial simulations, as depicted in Figure 5. Notably, errors are concentrated in the back area of the skull, where the rectangular constraints of our fitting procedure may no longer align well with the skin wrap. Figure 6 provides visual examples of the fitting process.

4.2. SoftDECA

4.2.1. Dataset & Training

To train and evaluate f, we construct a dataset comprising 500k training and test instances using the pipeline detailed in Section 3.4. The parallelized creation of the dataset spanned five days and necessitated one terabyte of storage. To address the disparate sizes of the parameter spaces, 75% of the generated data consists of static instances where all parameters except the dynamic ones α, β, γ are sampled. The remaining 25% of the data is dynamically simulated, resulting in the generation of 6250 dynamic sequences, each with a length of 16 frames. To initiate dynamic sequences with a reasonable velocity, a longer sequence of length 2048 is pre-simulated with fixed dynamics parameters. For each dynamic sequence, a random observed velocity from the long sequence is drawn as the initialization. The dataset is divided into 90% for training and 10% for testing, ensuring no repetition of the same identity, simulation parameters, or facial expression in both sets.

During training, the Adam optimizer executes 200k update steps with a learning rate of 0.0001, linearly decreasing to 0.00005, and a batch size of 128. The training specifications result in an approximate runtime of 8 hours on an NVIDIA A6000. The relatively brief training duration can be attributed to the efficient network design and less noisy training data compared to scenarios typically encountered in image-based deep learning.

4.2.2. Quantitative Analysis

We quantitatively evaluate SoftDECA based on the L2 reconstruction error with respect to the targeted physics-based simulation and the computational runtimes. Additionally, we compare SoftDECA against Subspace Neural Physics (SNP) [14] and SoftSMPL [15] architectures adapted for facial simulations, recognized as state-of-the-art methods for rapid approximations of physics-based simulations. An overview of all results is provided in Table 1. The reported runtimes represent averages of ten runs measured on a consumer-grade Intel i5 12600K processor. All implementations rely on PyTorch[1].

SoftDECA outputs precise approximations for both static and dynamic animations, showcasing average test reconstruction errors of only 0.22 mm and 0.41 mm, respectively. The results underscore SoftDECA's capacity to generalize effectively across diverse human identities, facial expressions, and simulation parameters. However, the test data fully stems from unpersonalized blendshapes, necessitating further assessment using an external dataset obtained from 3DScanstore[2].

The external data is compromised of 20 to 35 scanned facial expressions for each of seven human identities. We create personalized ARKit blendshapes per head using example-based facial rigging [57]. Subsequently, a test dataset is generated as before. Despite the possibility that the 3DScanstore examples may not align with the DECA distribution, the reconstruction error experiences only a marginal increase to 0.44 mm.

Noteworthy is SoftDECA's swift performance, with an average runtime of 7.45 ms for static frames and 9.87 ms for dynamic frames. This rapid processing makes SoftDECA an appealing choice for resource-demanding applications. Additionally, in scenarios where unseen personalized blendshapes are undesirable, we explored a variant of SoftDECA directly predicting vertex positions. This alternative achieves an accuracy of 0.16 mm and can be executed at an accelerated pace of 0.71 ms per frame.

4.2.3. Static Comparisons

In static simulations, SoftDECA is compared with SoftSMPL, as SNP is exclusively tailored for approximating dynamic effects. The key distinction between the SoftDECA and SoftSMPL architectures lies in the choice between our hypernetwork MLP and a conventional MLP. Originally designed for full-body applications, SoftSMPL takes a motion descriptor as input, summarizing a body and its state. In our case, this translates to blendshape weights, simulation parameters, and the identity code. To maintain consistent inference times, we employ identical network dimensions for the standard MLP as those in the hypernetwork. Consequently, the SoftSMPL MLP experiences a notable increase in the reconstruction error, averaging 1.67 mm. We also explore a larger MLP that achieves a comparable reconstruction error to SoftDECA, however, this results in a substantial increase in runtime to 46.61ms.

Another canonical alternative to the hypernetwork is a standard MLP that does not map to all DGs simultaneously but is evaluated face-wise. This approach yields a low reconstruction error of 0.17 mm, yet it comes with a higher runtime of 34.92 ms. Other architectures like CNNs, GNNs, or transformers could not be evaluated in real-time on a consumer-grade CPU with sufficient accuracy. For CNNs and GNNs, this is due to the fundamental sparse convolutions that are depended on very deep network layers to represent global effects (CNN, GNN). Further, transformer architectures usually require an attention mechanism with quadratic runtime but even optimized set transformer [53] involve significantly more operations than standard MLPs.

4.2.4. Dynamic Comparisons

For dynamic simulations, we compare SoftDECA with SoftSMPL and SNP. Unlike SoftDECA, both SoftSMPL and SNP perform dynamic computations in a latent space rather than directly on vertices. Further, SoftSMPL incorporates a recurrent GRU network [58], while SNP relies solely on a standard MLP. For this comparison, we only consider the *larger* network design mentioned earlier, as our primary focus is on evaluating the accuracy of our dynamic approximation rather than comparing runtimes. At this, both SoftSMPL and SNP exhibit slightly improved reconstruction errors at 0.22 mm and 0.24 mm, respectively. However, since both methods do not operate vertex-wise, they are not suitable for handling locally varying parameters of the dynamic simulation.

4.2.5. Qualitative Analysis

A visual illustration of SoftDECA's capabilities is given in Figure 7, presenting a comparison between SoftDECA predictions and the targeted physics-based facial simulation. For example, in (a), it is evident that while collisions are not guaranteed to be entirely eliminated, they are largely mitigated. In (b), a localized increase in triangle strain on the skin around the cheeks results in the formation of wrinkles in that region. The result in (c) demonstrates the incorporation of external effects as heightened gravity. A *surgical manipulation* is shown in (d), where the jaw is lengthened along the vertical axis in the neutral state while maintaining the head's volume. The representation of a humanoid alien in (e) illustrates SoftDECA's robustness even outside the DECA distribution. This robustness is primarily achieved by transferring DGs instead of directly predicting vertex positions. Our interpretation of zombification in (f) is realized by expanding the skin area, highlighting SoftDECA's capability to closely approximate high-frequency details. Lastly, in (g-h), we depict the simulation of different weight additions in a non-linear manner, raising the soft tissue volume by 20% and 40%, respectively. Given the extensive training domain

[1] https://pytorch.org
[2] https://www.3dscanstore.com

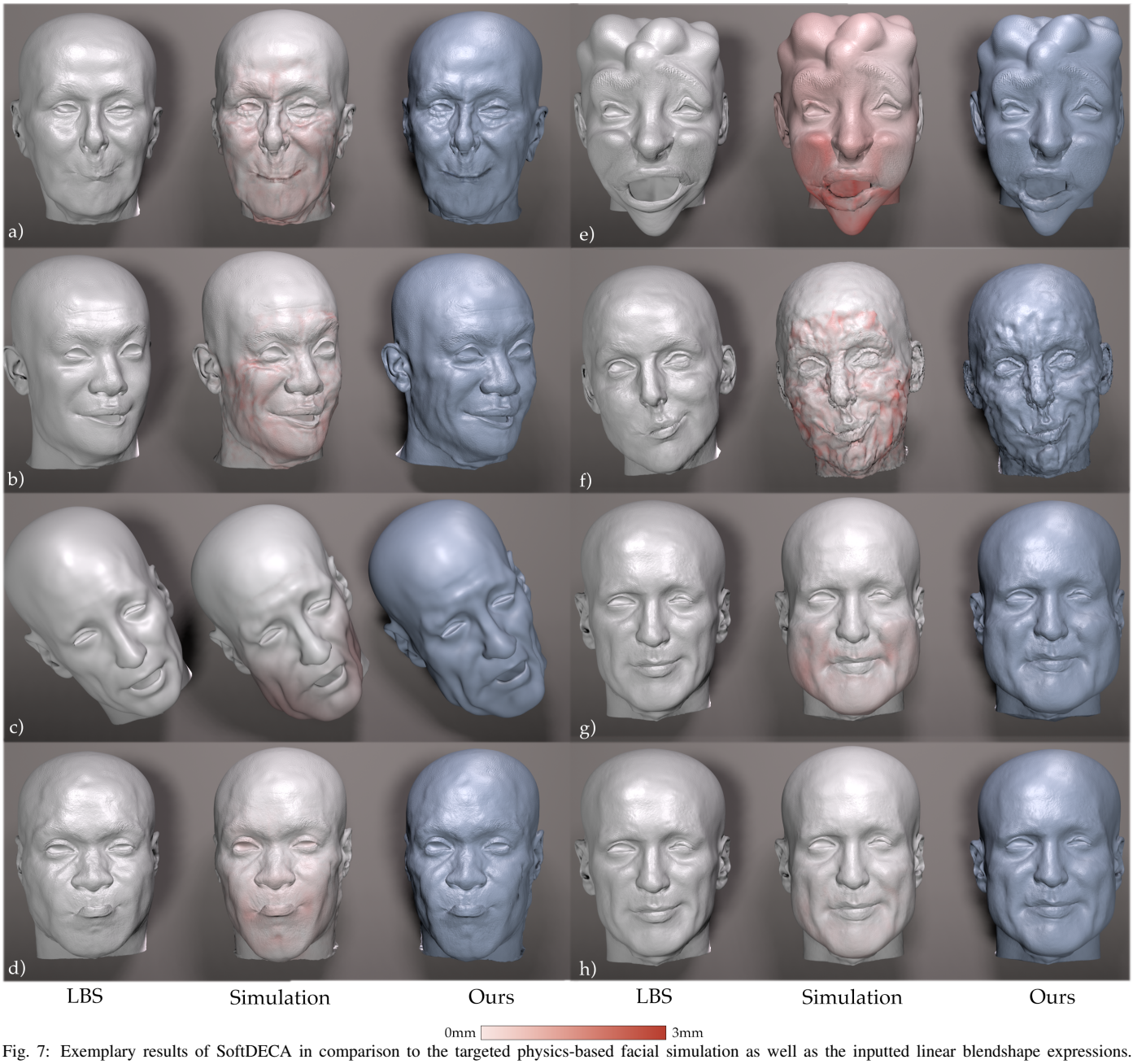

Fig. 7: Exemplary results of SoftDECA in comparison to the targeted physics-based facial simulation as well as the inputted linear blendshape expressions. Reconstruction errors are plotted on the simulated expressions.

Model	**Ours**			**SoftSMPL**			**SNP**	**Ablation**	
	Static	Dynamic	External	Static (Small)	Static (Large)	Dynamic	Dynamic	Face-wise	Only Vertices
Error in mm	0.23	0.41	0.44	1.67	0.16	0.22	0.14	0.17	0.16
Time in ms	7.45	9.87	7.45	7.62	46.61	47.39	46.61	34.92	0.72

Table 1: SoftDECA test results in comparison to adapted SNP [14] and SoftSMPL [15] architectures as well as ablations. The runtimes are averages measured on a consumer-grade Intel i5 12600K processor. External refers to the 3Dscanstore dataset. Small and large correspond to the size of the inspected MLP.

Fig. 8: Exemplary results of SparseSoftDECA (right) in comparison to the targeted physics-based facial simulation (left) as well as the inputted landmarks (red dots). Additionally, in b), the combination of SparseSoftDECA with skin textures is displayed. In the last row of b), Gaussian noise has been applied to the landmarks.

Model	Ours		Ablation	
	Same Identity	Other Identity	With Noise	Without Noise
Error in mm	0.54	0.62	0.55	0.73

Table 2: SparseSoftDECA test results using both the same and a different head shape for personalization. Additionally, we investigate the influence of applying noise to the facial landmarks in the training set.

of SoftDECA, many other effects can be animated efficiently which are not displayed in Figure 7. Additional results, including dynamic effects, are available in the supplementary material video.

4.3. SparseSoftDECA

4.3.1. Dataset & Training

For the training and assessment of SparseSoftDECA, we create a dataset consisting of 500k training and test examples by following the procedure outlined in Section 3.6.2. Specifically, we simulate 50 distinct sets of facial expressions for each of 10,000 randomly selected identities. The dataset is divided into 90% for training and 10% for testing, ensuring that neither the same identity nor the same facial landmarks appear in both sets. To further rigorously evaluate the robustness of SparseSoftDECA in the face of incorrect and noisy inputs, as well as its generalization capacities, we extend σ in Equation (24). In contrast to training examples, for test examples the process of personalizing the landmarks applies a separate test identity.

The training process and hyperparameters used are consistent with those described in Section 4.2.1.

4.3.2. Quantitative Analysis

SparseSoftDECA demonstrates the ability to closely mimic sparse landmark-guided simulations, as illustrated in Table 2. Whether personalization involves the same individual or a different one appears to be almost irrelevant. The minimal L2-errors of 0.54 mm and 0.62 mm affirm the robustness of SparseSoftDECA in handling erroneous and noisy inputs. We also investigated the influence of training data augmentation with Gaussian noise (standard deviation of 0.01). A slight improvement of the error from 0.73 mm to 0.55 mm can be observed.

In general, the errors observed are greater compared to those of SoftDECA. This can be attributed to the increased complexity of the task. Previously, the learning focus was primarily on changes in simulation properties, whereas now the learning task involves predicting entire facial expressions.

4.3.3. Qualitative Analysis

The images depicted in Figure 8 illustrate landmarks, corresponding simulations, and predictions generated by SparseSoftDECA. In b), skin textures are exhibited aside of the geometry to demonstrate the quality of the final animation result. For the last row of b), Gaussian noise was applied to the landmarks, while all other examples are free of noise. On one hand, the reproduction quality evident from the measured test errors is visually confirmed. On the other hand, the benefits of physics-based simulations are reemphasized, highlighting their capacity to transform even highly noisy landmark inputs into anatomically plausible facial expressions. The principal advantage, however, is that all expressions were generated using only sparse landmarks as input and no underlying blendshapes had to laboriously sculpted. As a side effect, no blendshapes need to be stored, which can greatly reduce the memory footprint depending on the type of animation.

To observe the temporal consistency of SparseSoftDECA we kindly refer the reader to the attached video.

5. Limitations

Although SoftDECA inherits most of the advantages of physics-based facial animations, it lacks the intrinsic handling of interactive effects such as wind or colliding objects. Moreover, although we allow for extensive localized artistic interventions, mixtures of material properties have not been part of the training data. Incorporating such mixtures into the training data is difficult as it is hard to define an adequate mixture distribution. Nonetheless, the smooth material blending of SoftDECA visually appears to be a sufficient approximation.

Despite SparseSoftDECA differing from SoftDECA in that it is not constrained by a specific set of blendshape weights, it operates on a predefined set of landmarks. However, this limitation could potentially be overcome in future research by implementing a training process that utilizes randomly selected landmark sets. In general, identifying an optimal set of landmarks is left to future work.

6. Conclusion

In this work, we have presented SoftDECA, which provides a computationally efficient approximation to physics-based facial simulations, even on consumer-grade hardware. With a few exceptions, most simulation capabilities are retained, such as dynamic effects, volume preservation, wrinkle generation, and many more. SoftDECA's runtime performance is attractive for high-performance applications and low-cost hardware. In addition, it is versatile as it supports different head shapes, facial expressions, and material properties. The ability to make local adjustments after training makes it a valuable framework for artistic customization.

Our future goals for improving SoftDECA are twofold. On the one hand, we want to refine the anatomical model to achieve an even more accurate representation, especially for structures such as the trachea and esophagus. On the other hand, latest results demonstrate the efficient learning of contact deformations [59]. Given that people often touch their face several times a day, introducing a contact treatment for more realistic gestures could significantly improve immersion.

In continuation of the earlier presentation of SoftDECA [19], this work also includes the introduction of SparseSoftDECA. SparseSoftDECA enables blendshape-free facial animation based on sparse landmarks and exhibits the same generalization characteristics as SoftDECA. SparseSoftDECA seamlessly integrates with the avatar generation pipeline proposed by

Wenninger et al. [20], making it straightforward to deploy.

References

[1] Cao, C, Simon, T, Kim, JK, Schwartz, G, Zollhoefer, M, Saito, SS, et al. Authentic volumetric avatars from a phone scan. ACM Transactions on Graphics (TOG) 2022;41(4):1–19.

[2] Grassal, PW, Prinzler, M, Leistner, T, Rother, C, Nießner, M, Thies, J. Neural head avatars from monocular RGB videos. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022, p. 18653–18664.

[3] Athar, S, Xu, Z, Sunkavalli, K, Shechtman, E, Shu, Z. RigNeRF: Fully Controllable Neural 3D Portraits. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022, p. 20364–20373.

[4] Zielonka, W, Bolkart, T, Thies, J. Instant volumetric head avatars. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2023, p. 4574–4584.

[5] Lewis, JP, Anjyo, K, Rhee, T, Zhang, M, Pighin, FH, Deng, Z. Practice and theory of blendshape facial models. Eurographics (State of the Art Reports) 2014;1(8):2.

[6] Ichim, AE, Kadleček, P, Kavan, L, Pauly, M. Phace: Physics-based face modeling and animation. ACM Transactions on Graphics (TOG) 2017;36(4):1–14.

[7] Ichim, AE, Kavan, L, Nimier-David, M, Pauly, M. Building and animating user-specific volumetric face rigs. In: Symposium on Computer Animation. 2016, p. 107–117.

[8] Cong, MD. Art-directed muscle simulation for high-end facial animation. Stanford University; 2016.

[9] Choi, B, Eom, H, Mouscadet, B, Cullingford, S, Ma, K, Gassel, S, et al. Animatomy: an Animator-centric, Anatomically Inspired System for 3D Facial Modeling, Animation and Transfer. In: SIGGRAPH Asia 2022 Conference Papers. 2022, p. 1–9.

[10] Yang, L, Kim, B, Zoss, G, Gözcü, B, Gross, M, Solenthaler, B. Implicit neural representation for physics-driven actuated soft bodies. ACM Transactions on Graphics (TOG) 2022;41(4):1–10.

[11] Barrielle, V, Stoiber, N, Cagniart, C. Blendforces: A dynamic framework for facial animation. In: Computer Graphics Forum; vol. 35. 2016, p. 341–352.

[12] Srinivasan, SG, Wang, Q, Rojas, J, Klár, G, Kavan, L, Sifakis, E. Learning active quasistatic physics-based models from data. ACM Transactions on Graphics (TOG) 2021;40(4):1–14.

[13] Brandt, C, Eisemann, E, Hildebrandt, K. Hyper-reduced projective dynamics. ACM Transactions on Graphics (TOG) 2018;37(4):1–13.

[14] Holden, D, Duong, BC, Datta, S, Nowrouzezahrai, D. Subspace neural physics: Fast data-driven interactive simulation. In: Proceedings of the 18th annual ACM SIGGRAPH/Eurographics Symposium on Computer Animation. 2019, p. 1–12.

[15] Santesteban, I, Garces, E, Otaduy, MA, Casas, D. SoftSMPL: Data-driven Modeling of Nonlinear Soft-tissue Dynamics for Parametric Humans. In: Computer Graphics Forum; vol. 39. 2020, p. 65–75.

[16] Cong, M, Fedkiw, R. Muscle-based facial retargeting with anatomical constraints. In: ACM SIGGRAPH 2019 Talks. 2019, p. 1–2.

[17] Ha, D, Dai, A, Le, QV. Hypernetworks. arXiv preprint arXiv:160909106 2016;.

[18] Feng, Y, Feng, H, Black, MJ, Bolkart, T. Learning an animatable detailed 3D face model from in-the-wild images. ACM Transactions on Graphics (TOG) 2021;40(4):1–13.

[19] Wagner, N, Botsch, M, Schwanecke, U. SoftDECA: Computationally Efficient Physics-Based Facial Animations. In: Proceedings of the 16th ACM SIGGRAPH Conference on Motion, Interaction and Games. 2023, p. 1–11.

[20] Wenninger, S, Achenbach, J, Bartl, A, Latoschik, ME, Botsch, M. Realistic virtual humans from smartphone videos. In: Proceedings of the 26th ACM Symposium on Virtual Reality Software and Technology. 2020, p. 1–11.

[21] Ali-Hamadi, D, Liu, T, Gilles, B, Kavan, L, Faure, F, Palombi, O, et al. Anatomy transfer. ACM Transactions on Graphics (TOG) 2013;32(6):1–8.

[22] Gilles, B, Reveret, L, Pai, DK. Creating and animating subject-specific anatomical models. In: Computer Graphics Forum; vol. 29. 2010, p. 2340–2351.

[23] Kadleček, P, Ichim, AE, Liu, T, Křivánek, J, Kavan, L. Reconstructing personalized anatomical models for physics-based body animation. ACM Transactions on Graphics (TOG) 2016;35(6):1–13.

[24] Saito, S, Zhou, ZY, Kavan, L. Computational bodybuilding: Anatomically-based modeling of human bodies. ACM Transactions on Graphics (TOG) 2015;34(4):1–12.

[25] Schleicher, R, Nitschke, M, Martschinke, J, Stamminger, M, Eskofier, BM, Klucken, J, et al. BASH: Biomechanical Animated Skinned Human for Visualization of Kinematics and Muscle Activity. In: VISIGRAPP (1: GRAPP). 2021, p. 25–36.

[26] Keller, M, Zuffi, S, Black, MJ, Pujades, S. OSSO: Obtaining Skeletal Shape from Outside. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022, p. 20492–20501.

[27] Keller, M, Werling, K, Shin, S, Delp, S, Pujades, S, C. Karen, L, et al. From Skin to Skeleton: Towards Biomechanically Accurate 3D Digital Humans. In: ACM TOG, Proc. SIGGRAPH Asia. 2023,.

[28] Komaritzan, M, Wenninger, S, Botsch, M. Inside Humans: Creating a Simple Layered Anatomical Model from Human Surface Scans. Frontiers in Virtual Reality 2021;2:694244.

[29] Maalin, N, Mohamed, S, Kramer, RS, Cornelissen, PL, Martin, D, Tovée, MJ. Beyond BMI for self-estimates of body size and shape: A new method for developing stimuli correctly calibrated for body composition. Behavior Research Methods 2021;53(3):1308–1321.

[30] Achenbach, J, Brylka, R, Gietzen, T, zum Hebel, K, Schömer, E, Schulze, R, et al. A multilinear model for bidirectional craniofacial reconstruction. In: Proceedings of the Eurographics Workshop on Visual Computing for Biology and Medicine. 2018, p. 67–76.

[31] Ichim, AE, Bouaziz, S, Pauly, M. Dynamic 3D avatar creation from hand-held video input. ACM Transactions on Graphics (TOG) 2015;34(4):1–14.

[32] Bradley, D, Heidrich, W, Popa, T, Sheffer, A. High resolution passive facial performance capture. In: ACM SIGGRAPH 2010 papers. 2010, p. 1–10.

[33] Zhang, L, Snavely, N, Curless, B, Seitz, SM. Spacetime faces: High-resolution capture for modeling and animation. In: Data-Driven 3D Facial Animation. Springer; 2008, p. 248–276.

[34] Parke, FI. Control parameterization for facial animation. In: Computer Animation'91. 1991, p. 3–14.

[35] Lewis, JP, Mooser, J, Deng, Z, Neumann, U. Reducing blendshape interference by selected motion attenuation. In: Proceedings of the 2005 symposium on Interactive 3D graphics and games. 2005, p. 25–29.

[36] Zheng, Y, Abrevaya, VF, Bühler, MC, Chen, X, Black, MJ, Hilliges, O. Im avatar: Implicit morphable head avatars from videos. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022, p. 13545–13555.

[37] Garbin, SJ, Kowalski, M, Estellers, V, Szymanowicz, S, Rezaeifar, S, Shen, J, et al. VolTeMorph: Realtime, Controllable and Generalisable Animation of Volumetric Representations. arXiv preprint arXiv:220800949 2022;.

[38] Song, SL, Shi, W, Reed, M. Accurate face rig approximation with deep differential subspace reconstruction. ACM Transactions on Graphics (TOG) 2020;39(4):34–1.

[39] Sifakis, E, Neverov, I, Fedkiw, R. Automatic determination of facial muscle activations from sparse motion capture marker data. In: ACM SIGGRAPH 2005 Papers. 2005, p. 417–425.

[40] Bao, M, Cong, M, Grabli, S, Fedkiw, R. High-quality face capture using anatomical muscles. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019, p. 10802–10811.

[41] Kadleček, P, Kavan, L. Building accurate physics-based face models from data. Proceedings of the ACM on Computer Graphics and Interactive Techniques 2019;2(2):1–16.

[42] Bickel, B, Lang, M, Botsch, M, Otaduy, MA, Gross, MH. Pose-Space Animation and Transfer of Facial Details. In: Symposium on Computer Animation. 2008, p. 57–66.

[43] Kozlov, Y, Bradley, D, Bächer, M, Thomaszewski, B, Beeler, T, Gross, M. Enriching facial blendshape rigs with physical simulation. In: Computer Graphics Forum; vol. 36. 2017, p. 75–84.

[44] Casas, D, Otaduy, MA. Learning nonlinear soft-tissue dynamics for interactive avatars. Proceedings of the ACM on Computer Graphics and Interactive Techniques 2018;1(1):1–15.
[45] Loper, M, Mahmood, N, Romero, J, Pons-Moll, G, Black, MJ. SMPL: A Skinned Multi-Person Linear Model. ACM Trans Graphics (Proc SIGGRAPH Asia) 2015;34(6):248:1–248:16.
[46] Botsch, M, Kobbelt, L. Real-time shape editing using radial basis functions. In: Computer graphics forum; vol. 24. 2005, p. 611–621.
[47] Bouaziz, S, Martin, S, Liu, T, Kavan, L, Pauly, M. Projective dynamics: Fusing constraint projections for fast simulation. ACM Transactions on Graphics (TOG) 2014;33(4):1–11.
[48] Gietzen, T, Brylka, R, Achenbach, J, Zum Hebel, K, Schömer, E, Botsch, M, et al. A method for automatic forensic facial reconstruction based on dense statistics of soft tissue thickness. PloS one 2019;14(1):e0210257.
[49] Komaritzan, M, Botsch, M. Projective skinning. Proceedings of the ACM on Computer Graphics and Interactive Techniques 2018;1(1):1–19.
[50] Karras, T, Laine, S, Aila, T. A style-based generator architecture for generative adversarial networks. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019, p. 4401–4410.
[51] Botsch, M, Sumner, R, Pauly, M, Gross, M. Deformation transfer for detail-preserving surface editing. In: Vision, Modeling & Visualization. 2006, p. 357–364.
[52] Sumner, RW, Popović, J. Deformation transfer for triangle meshes. ACM Transactions on Graphics (TOG) 2004;23(3):399–405.
[53] Lee, J, Lee, Y, Kim, J, Kosiorek, A, Choi, S, Teh, YW. Set transformer: A framework for attention-based permutation-invariant neural networks. In: International conference on machine learning. 2019, p. 3744–3753.
[54] Scarselli, F, Gori, M, Tsoi, AC, Hagenbuchner, M, Monfardini, G. The graph neural network model. IEEE transactions on neural networks 2008;20(1):61–80.
[55] Mildenhall, B, Srinivasan, PP, Tancik, M, Barron, JT, Ramamoorthi, R, Ng, R. Nerf: Representing scenes as neural radiance fields for view synthesis. Communications of the ACM 2021;65(1):99–106.
[56] Beeler, T, Bradley, D. Rigid stabilization of facial expressions. ACM Transactions on Graphics (TOG) 2014;33(4):1–9.
[57] Li, H, Weise, T, Pauly, M. Example-based facial rigging. ACM Transactions on Graphics (TOG) 2010;29(4):1–6.
[58] Chung, J, Gulcehre, C, Cho, K, Bengio, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:14123555 2014;.
[59] Romero, C, Casas, D, Chiaramonte, MM, Otaduy, MA. Contact-centric deformation learning. ACM Transactions on Graphics (TOG) 2022;41(4):1–11.
[60] Botsch, M, Kobbelt, L, Pauly, M, Alliez, P, Lévy, B. Polygon mesh processing. CRC press; 2010.

Appendix A. Simulation Parameters

In the following, we describe all simulation parameters that haven been sampled during the creation of the SoftDECA training data. Moreover, we state the sampling range for each parameter. This list is not complete in the sense that SoftDECA is not committed to it. However, these parameters already provide a comprehensive test of SoftDECA's capabilities and allow for extensive individualization opportunities.

- *Dynamics* We sample each of the parameters α, β, γ that steer the dynamic second order effects in a range from 0 to 2.
- *Constraint Weights* All weights w_* associated with the constraints of $\phi^\dagger$ are sampled between 0.001 and 100.
- *Volume* The target determinant in the volume energy E_{vol} is sampled from 0.5 to 1.5.
- *Maximum Strain* We allow a varying amount of maximum soft tissue strain by adjusting the ϵ from 0.7 to 1.3.
- *Gravity* An additional gravity force is applied in a range from standard earth's gravity up to two times the standard. Further, the gravity direction is sampled.
- *Skull* We incorporate changes in the skull bones by sampling coordinate-wise scaling factors for both the cranium and jaw in the range from 0.5 to 1.5.

Appendix B. Energies

In the following, we formally state all energies under optimization.

Volume & Strain

$$E_{\text{vol}}(t) = (\det(\mathrm{F}(t)) - 1)^2 \tag{B.1}$$

$$E_{\text{str}}(t) = \min_{R \in SO(3)} \|\mathrm{F}(t) - R\|_F^2 \tag{B.2}$$

$\mathrm{F}(t)$ denotes the deformation gradient of a tetrahedron t, $R \in SO(3)$ the optimal rotation, and $\|\cdot\|_F^2$ the Frobenius norm.

Bending

$$E_{\text{curv}}(x, B) = A_x \|\Delta x - R\Delta b_x\|^2 \tag{B.3}$$

The matrix $R \in SO(3)$ denotes the optimal rotation keeping the vertex Laplacian Δx as close as possible to its initial value Δb_x. The vertex Laplacian is discretized using the cotangent weights and the Voronoi areas A_x [60].

Soft Dirichlet

$$E_{\text{tar}}\left(x, S_{\text{exp}}\right) = \|x - s_x\|^2, \tag{B.4}$$

attracts each vertex x of the skin surface S to the corresponding vertex s_x from the target expression S_{exp}.

Fitting Distances

$$E_{\text{dist}_2}\left(X, \hat{S}, D\left(\hat{S}\right)\right) = \sum_{x \in X} (\|x - s_x\| - d_x)^2 \tag{B.5}$$

ensures that for each vertex $x \in X$ the predicted distance $d_x \in D(\hat{S})$ is adhered to.

Appendix C. Template Layered Head Model

Table C.3 states the cardinality of each component of the layered head model template. By subdividing the wrap meshes or the triangle prisms between the wraps, the resolution of the template tetrahedron meshes can easily be adjusted. We will provide a mapping between the DECA and our topology.

Mesh	$S_{\mathcal{T}}$	$B_{\mathcal{T}}$	$M_{\mathcal{T}}$	$\hat{S}_{\mathcal{T}}$
#Vertices	35621	14572	16388	7826
#Faces / #Tetrahedrons	71358	28856	32370	15648

Mesh	$\hat{B}_{\mathcal{T}}$	$\hat{M}_{\mathcal{T}}$	$\mathbb{S}_{\mathcal{T}}$	$\mathbb{M}_{\mathcal{T}}$
#Vertices	7826	7826	49852	
#Faces / #Tetrahedrons	15648	15648	123429	73681

Table C.3: Template dimensions.

Appendix D. Network Dimensions

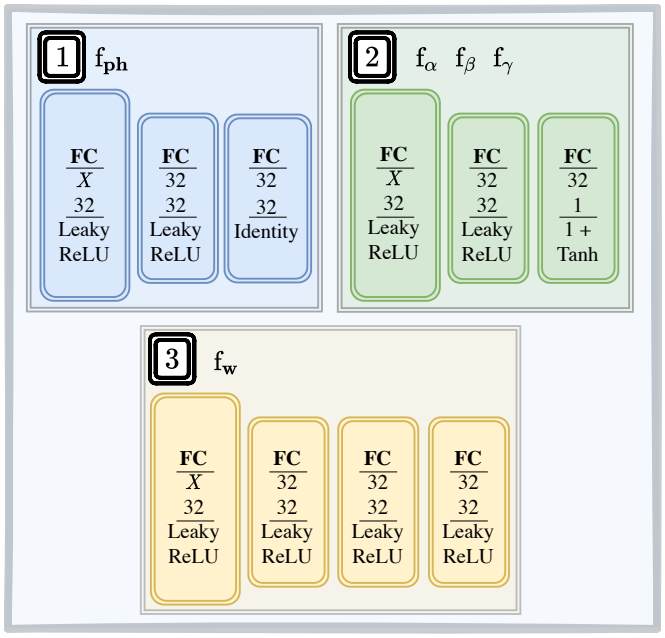

Fig. D.9: Network dimensions. Each fully connected layer (FC) is represented as a box. For each FC, the input and output dimensions are stated as well as the applied activation function.