EUROGRAPHICS 2026 / B. Masia and J. Thies
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 45 (2026), Number 2

CANRIG: Cross-Attention Neural Face Rigging with Variable Local Control

Arad Mohammadi[1,2,*], Sebastian Weiss[2,*], Jakob Buhmann[2], Loic Ciccone[2], Robert W. Sumner[1,2], Derek Bradley[2], Martin Guay[2]

[1] ETH Zürich, Switzerland [2] DisneyResearch|Studios, Switzerland *Equal Contribution

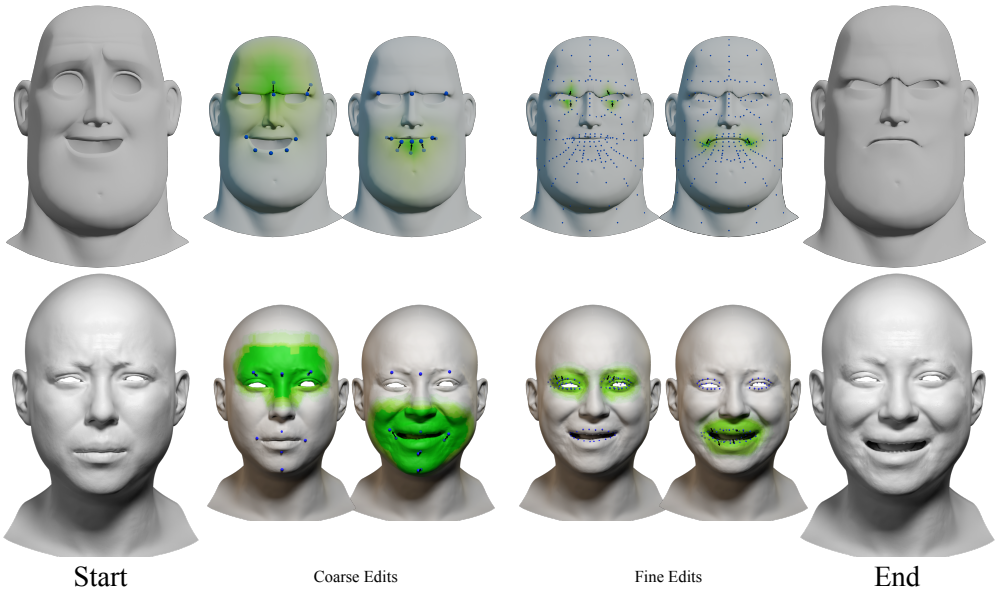


Figure 1: *An example of an editing workflow with our shape-preserving model that supports additive deformation layers, user-defined editing regions (green), and arbitrary layouts of control points (blue) for both coarse and fine edits. Given an initial shape (left), the user performs two coarse deformations on the forehead and mouth using a few sparse control points. The result is then used as the base for a new layer, allowing for precise, fine-grained edits. The user can then increase the density of control points and reduce the editing region to manipulate specific areas, such as the eyelids and mouth corners, to achieve the final, desired shape (right). ©Disney/Pixar*

Abstract

Facial animation is one of the most labor-intensive aspects of animation and VFX, as traditional rigging consumes weeks of expert time and forces animators to spend countless hours manipulating hundreds of controls to achieve varied expressions. This technical complexity creates a barrier between artistic vision and execution, limiting creative exploration and iteration. In this paper, we introduce CANRig*, a fully automated neural facial rigging approach that simplifies the process of creating and editing facial poses by benefiting from global correlations learned from data. Unlike existing neural face models that either sacrifice local control or demand extensive manual region setup, our method introduces continuous local control through a novel conditioning mechanism that operates on a variable region. By modeling deformation as cross-attention between control handles and mesh vertices—modulated by a user-defined region—we enable seamless transitions from precise local adjustments to broad global changes. We further expand our method with a shape-preserving workflow that enables iterative edits, guaranteeing that changes remain untouched even as controls are reconfigured. Our method delivers the best of both worlds: the automation and naturalness of neural methods with the granular control that professional animators demand, and we demonstrate its effectiveness across multiple applications in both animation and high-end visual effects pipelines.*

CCS Concepts

• ***Computing methodologies*** → ***Shape modeling; Animation; Machine learning;***

1. Introduction

Creating compelling facial animations is notoriously challenging and time-consuming. The traditional workflow is highly manual, requiring significant effort to rig a character and then painstakingly coordinate a multitude of controls to achieve a desired pose. In contrast, learned face rigs present a powerful paradigm shift [DZX*23, QSA*23, VRWW20, BCR*21, CPH*25, GZC*16, AC24, SSR20, RdGM20]. Thanks to advances in deep learning, neural rigs can automatically learn the complex nonlinear rig from raw data and enable faster animation by leveraging global correlations. These capabilities unlock profound applications, including rapid posing, performance editing, and animatable asset transfer to different software without manual rigging work [YLL*25, AGB*23, GXM*25].

However, a critical limitation persists in current neural rigging methods. The global correlations learned from data must be broken down into local face regions for artistic editing. For example, when an artist edits the lip shape they do not expect the eyebrows to move, but such a correlation may exist in the training data. As such, existing methods typically create this local control by naively dividing the face into a fixed set of predefined regions before training [RSJ*21, MCY19]. At runtime, local control is then constrained to those regions. This lack of user-specified control over the deformation area at runtime prevents artists from fully realizing their creative vision, forcing them to work within the rigid boundaries of the rig. To overcome this, we introduce *CANRig*, *a novel neural architecture that provides dynamic, user-controlled locality*. Our model provides continuous control over the region of influence, allowing an artist to precisely define the area for any given control handle at runtime.

Our approach achieves this natural and precise local control by formulating deformation as a cross-attention between a set of control handles and the vertices of the mesh. The attention matrix is then constrained by per-vertex weights, derived from the user-specified local radius. During training, our model is exposed to a wide range of continuous local regions, enabling it to learn and infer natural deformations with a controllable area of influence. Furthermore, to provide a pleasant and seamless workflow, we address a significant challenge that arises when transitioning between different control radii during a sequence of progressive edits. Without a solution, changing the control radius can cause the previous pose to suddenly deform, destroying the artist's prior work. We solve this by designing a novel *shape-preserving bias* into our network architecture, which ensures that previous edits are maintained as the artist fluidly changes the control configuration, providing a truly robust and non-destructive editing experience. Additionally, the shape-preservation also results in zero error inversion when editing captured performances in live-action.

We design CANRig for both animation and high-end visual effects. In animation, the facial rigs have relatively low resolution (*e.g.* 1000s of vertices) and attention from all control handles to all vertices is feasible (Figure 1, top row). In high-end visual effects, face models tend to be higher resolution (*e.g.* up to 100,000s vertices), and a dense attention matrix becomes infeasible. There, we propose a simple extension of our network to model facial deformation using a common patch blendshape system [CCGB22] with hundreds of small patches instead of per-vertex deltas (Figure 1, bottom row), allowing the same powerful neural rig architecture in a wide variety of applications. Finally, we showcase the capabilities of CANRig in several production scenarios, including quick posing from an existing frame, post-production performance editing, facial expression transfer, and iterative continuous editing.

In summary, we propose

- **A novel rigging architecture** that enables continuous, user-controlled deformations without pre-defined localities, and without manually painted weights.
- **An iterative editing workflow** that facilitates progressive, shape-preserving modifications, allowing artists to refine complex geometries through successive operations.
- **Cross-domain versatility**, demonstrating our system's effectiveness and scalability in both high-end feature animation and live-action production pipelines.

2. Related Work

In the following we present related work in different fields of controllable facial deformation, starting with more generic morphable face models, then moving on to traditional and more modern facial rigging approaches.

2.1. Morphable Face Models

For a long time, researchers have shown interest in modeling the deformation space of human faces. Initial methods analyzed 3D facial scans and built the first 3D Morphable Models (3DMMs) [BV99, VBPP05, LBB*17]. These models were not explicitly designed for artistic control, rather as parametric models for downstream tasks like fitting faces to images [FFBB21], however 3DMMs can be considered as rudimentary facial rigs.

Recent approaches have extended classical 3DMMs with powerful neural representations for face modeling and editing. For example, Chandran *et al.* [CGB20] proposed Semantic Deep Face Models (SDFM), one of the first deep morphable face models offering nonlinear deformation and explicit separation of identity and expression. Giebenhain *et al.* [GKG*23] propose a Neural Parametric Head Model (NPHM) that represents identity as a canonical signed distance field and facial expressions as a learned deformation field. Zheng *et al.* [ZZY*25] introduce ImFace++, a nonlinear 3DMM built from implicit neural fields, which learns separate deformation fields for identity and expression plus a refinement displacement field, and even a "neural blend-field" that adaptively blends local shape details. These neural 3DMMs achieve high-fidelity reconstruction and explicitly disentangle identity and expression, which helps preserve identity under edits. Style-based generative models have also been applied. Yan *et al.* [YWW*25] uses a StyleGAN-like decoder conditioned on latent codes, achieving state-of-the-art 3D-aware face reconstruction with disentangled controls for identity, expression, and appearance. In the implicit domain, Chen *et al.* [COBG23] decompose facial deformations into multiple local implicit fields centered on semantic handles, with 3DMM coefficients modulating each field. Aneja *et al.* [ATDN23] use differentiable rendering and CLIP-based losses in a self-supervised framework for text-guided editing of textured 3D morphable face models allowing high-quality texture synthesis and expressive 3D face animation in

a single forward pass. Potamias *et al.* [PTPZ24] uses a 3D diffusion model that allows localized shape editing based on diffusion inpainting. [TPO*24] enables fine-grained local editing by using a patch-based Transformer or MLP-Mixer to overwrite encoded geometry via sparse control-point displacements. However [TPO*24] lacks user-specified locality and its global backbone means edits can influence other regions. Although morphable face models are very powerful tools for exploring facial deformation, they are not generally appropriate for production-level facial rigging and animation.

2.2. Traditional Face Rigging

The most common approach to facial rigging is through blendshapes [OBP*12, LAR*14, LA10], where the facial deformation is created by linearly interpolating static facial expressions. The expressions in a blendshape model often follow specific muscle Action Units (AUs), defined by the historic Facial Action Coding System (FACS) [EF78]. The main problem with blendshape models is that hundreds and often thousands of shapes are required to achieve realistic facial animation. Nevertheless, many facial rigs used in production today are still largely based on blendshapes.

Over the years, researchers proposed several improvements in the space of linear face rigging. Example-Based Facial Rigging (EBFR) [LWP10] is an approach to automatically compute a blendshape rig given a number of example posed expressions, reducing the rig construction cost. Some methods aimed to provide more flexibility in the blendshape models by defining local editing regions [TDlTM11, NVW*13]. Other approaches explored free-form expression posing [LCXS09] and enriching blendshape rigs with physical simulation for added realism [KBB*17]. Overall, facial rigging has always been an important problem in computer graphics research, but modern approaches focus more on deep neural methods learned from data.

2.3. Neural Rigs

Neural animation rigs learned from data have advanced facial animation pipelines in recent years. Qin *et al.* [QSA*23] present Neural Face Rigging (NFR), an end-to-end system that trains an autoencoder on both synthetic and real data, with a latent expression space aligned with FACS-style parameters to allow artist-friendly edits. Earlier work like RigNet [XZK*20] similarly shows that mesh-based deep networks can predict complete rigs (skeletons and skinning weights) for articulated characters from geometry alone. Building on captured data, Zhao *et al.* [ZWG*23] automatically infer semantic joint positions and blendshape parameters from 4D facial sequences, enabling ultra-fast rig generation with AU-like semantics. Cha *et al.* [CYSN25] enables mesh-agnostic and fine-grained facial expression transfer via predicting per-vertex skinning weights. Real-time neural avatar frameworks demonstrate these ideas in practice. For example, Raina *et al.* [RTT*25] constructs a hybrid mesh–volumetric head model rigged by a prism lattice and a 3DMM, then distills it into a mesh with neural textures for fast rendering; the resulting avatar runs at 60 fps on mobile devices while preserving high visual fidelity.

The two closest related methods to the proposed CANRig are the Shape Transformer [CZG*22], a global transformer-based autoencoder from sparse handles to a full shape, and Animatomy [CEM*22], which uses local one-dimensional blendshapes to simulate virtual muscles (we refer to the video for a demo). In Table 1 we compare the features of these two techniques with ours. Shape Transformer encodes all handle edits into a global latent code and can thus not preserve local edits. Animatomy's locality is limited by the predefined muscle layout and cannot be changed by the artist directly. For global correlations, an additional autoencoder MLP is employed. In contrast, our method allows both global and local edits, with the locality freely defined by the artist at runtime. Additionally, we introduce iterative editing that allows for non-destructive layering and lossless editing of existing performances.

Table 1: *Comparison of the features offered by the Shape Transformer [CZG*22] and Animatomy [CEM*22] compared to the proposed CANRig. Shape Transformer is only global, Animatomy supports both global edits and local edits restricted to predefined virtual muscles. Our rig can freely and dynamically adapt from global to local and also supports iterative editing.*

Method	global edit	local edit	iterative edit
Shape Transformer	✓	✗	✗
Animatomy	✓*	✓*	✗
CANRig (ours)	✓	✓	✓

* Animatomy is global with an MLP and local up to the size of its predefined regions.

3. Method

We begin by describing the core architecture of our model (Figure 2), which uses a set of handles to predict the deformations of a 3D mesh. Subsequently, we introduce modifications to our model to enable a non-destructive iterative editing workflow.

3.1. Attention-based Local Deformation

The goal of our work is to enable local deformations of a facial mesh by directly manipulating a set of handles, which act as control points. To achieve this, we formulate shape deformations as a cross-attention between handles and vertices, with a modification to enforce a user-defined and flexible locality. The standard cross-attention layer is unsuitable for predicting localized deformations because it allows every handle to influence every vertex, an inherent property of attention that creates global dependencies. While these dependencies are great for fast posing, they do not permit user-controlled local refinement.

To overcome this limitation, we introduce a method based on local influence masks, which limits the deformation to user-defined local regions using a multiplicative mask in the attention layer. To implement this, we define S as the target 3D shape, which we want to reconstruct, with N vertices. The canonical shape, S_0, is defined as the character in a neutral pose with its mouth open. Next, we define a set of handles, denoted as H, which serve as the control points. Each handle, h, is specified by three components: its position on the canonical shape $p_h \in \mathbb{R}^3$, a user-defined displacement relative to this position $\Delta_h \in \mathbb{R}^3$, and an influence mask $M_h \in [0,1]^N$. The influence mask determines how much each handle's displacement

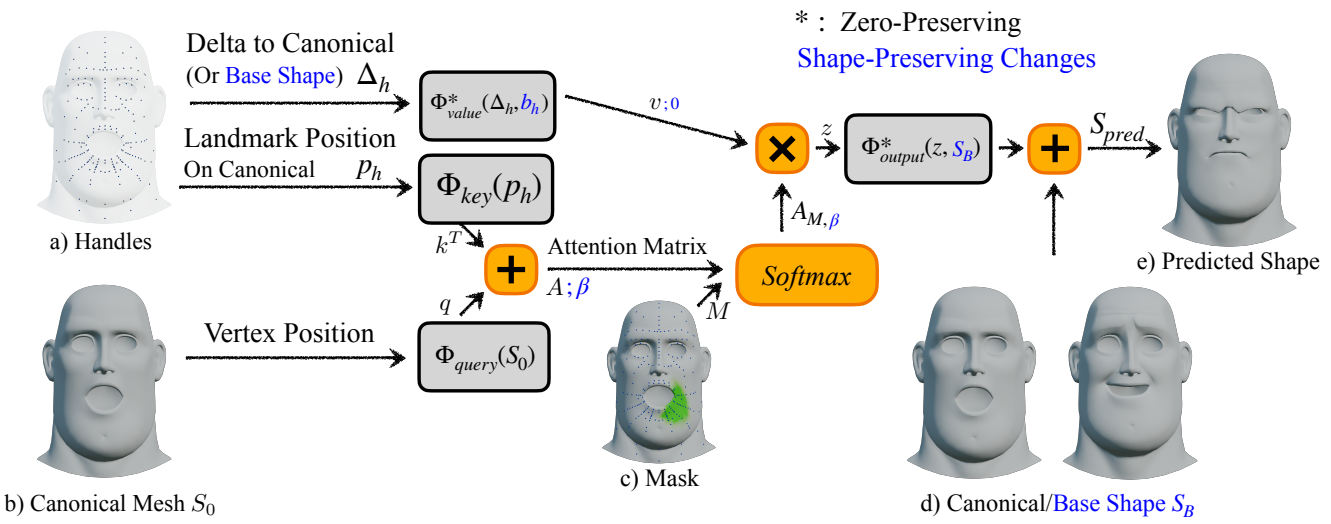


Figure 2: *The user moves the handles a), which are defined by their position on a canonical mesh and a delta position representing the user's edit. We encode the handles with learned MLPs to form the keys and values of the attention mechanism, while the canonical mesh b) provides the query. The attention then correlates the handle displacements with the mesh vertices to predict the full face deformation. To control the influence and ensure locality, the attention is masked so that handles only affect vertices within an artist-specified radius c), see Section 3.1. The output deformations are then added to the canonical mesh d) resulting in the final predicted shape e) (after multiple edits). Note that the value and output MLPs (*) are zero-preserving which we will discuss in Section 6.1. The extensions to support iterative edits from arbitrary base shapes, see Section 3.2, are marked in blue. ©Disney/Pixar*

affects the vertices. More formally, the influence of h on vertex v_i is computed as:

$$M_{h,v_i} = 0.5\Big(1 - \tanh\big(d(h,v_i) - r_h)\gamma_h\big)\Big)\,, \tag{1}$$

where $d(h,v_i)$ is a distance measure between h and v_i, γ_h is a falloff parameter, and r_h is a radius. These parameters are further illustrated in Figure 3. γ_h and r_h can be tuned by the user at runtime to control the locality of the edit (see Figure 4). We compute $d(h,v_i)$ using a weighted geodesic distance on the mesh, where each edge is weighted by its 3D Euclidean distance. For more discussion on the distance measure, we refer to Appendix A.

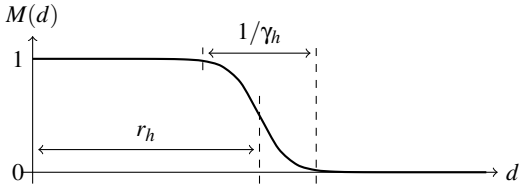


Figure 3: *The influence of a handle h on a vertex at distance d is controlled by the handle's radius r_h and falloff parameter γ_h.*

Next, as shown in Figure 2, the *single layer* cross attention connects the handles to the vertices. We construct the attention queries from the vertices of the canonical shape S_0, while the keys and values are derived from the handle positions (p_h) and their displacements (Δ_h), respectively. We then multiply the attention matrix by the attention mask M, an $N \times H$ matrix that concatenates the individual influence masks M_h. When handles compete for the same vertex, M acts as an additional constraint in the attention mechanism. Given A as the attention matrix, our masked attention is formulated as follows:

$$A_M = \sigma_M(A) = \frac{\exp(A)M}{\sum_{h=1}^{H}\exp(A_h)M_h} \quad \in \mathbb{R}^{N\times H}\,. \tag{2}$$

For simplicity, we omit the multi-head attention dimension in the description above.

In our implementation, we use multi-layer perceptrons (MLPs) [Hay94], Φ, to process the attention layer inputs and outputs. A problem with vanilla MLPs is that when a user creates a new handle, the inherent bias term in the MLP can cause undesired deformations, as shown in Figure 6. In other words, a zero displacement by the user may not translate to a zero deformation in the MLP output. To address this issue, we enforce a *zero-preservation* property by:

1. Removing the *bias* parameter in the value and output MLPs, Φ^*_{value} and Φ^*_{output}, and
2. Using an activation function where $a(0) = 0$ (we use GELU [HG16] for Φ^*_{value} and Tanh for Φ^*_{output}).

Finally, the output deformations will be added with S_0 to form the final prediction, S_{pred}. We then train our network to reconstruct the full face S using an MSE loss:

$$\mathcal{L}_{\text{L2}} = \|S - S_{\text{pred}}\|_2^2\,. \tag{3}$$

Figure 4: *Deformations using a single control handle (blue) with two varying user-defined editing region, indicated by 0 1. Note that with a small radius, the eyelids remain unaffected, whereas with a large radius, the eyelids move along with the eyebrow. ©Disney/Pixar*

3.2. Shape-preserving Iterative Editing

Initial testing with artists on the task of pose authoring indicated that controlling the influence of multiple control points at the same time was challenging and time-consuming. Recent work in the context of body posing [AGB*23] has introduced a concept of previous pose, or base pose, which preserves the previous edits of the user in an iterative workflow. This concept can also be used for editing a captured performance (see Section 7.2). The ability to bias our model towards preserving the captured frame enables direct manipulation without a lossy solve or manual setup. However, a challenge in implementing this concept is the lack of authoring data, since we do not have pairs of faces with user edits, but only the final movements. While Agrawal *et al.* [AGB*23] solved this with data augmentation, we found that their solution led to unnatural deformations in the context of dense face meshes. Rather than relying on data augmentation, we achieve a preservation towards the base shape, denoted as S_B, entirely via architecture and loss function designs, which are described below and highlighted in blue in Figure 2.

Architecture. First we modify the model's input and output to be relative to the base shape rather than the canonical shape. In this way, Φ^*_{value} now takes handle deltas relative to the S_B as input and Φ^*_{output} generates vertex deltas from S_B. The model still requires the base shape and corresponding handle positions as input so that its output is aware of the previous shape. Thus, we pass this information to Φ^*_{value} and Φ^*_{output}, by multiplying their inputs with handle positions on base shape, denoted as b_h, and base shape respectively. To match the dimension, we use a single linear layer to transform the Euclidean positions to the latent dimension of the Φ^*_{value} and Φ^*_{output}. Note that this way of modulation maintains the *zero-preservation* property of the model while also improving the model performance over no modulation, as illustrated in detail in Figure 8.

Attention to base shape. Our second change is a novel control mechanism: we incorporate a new attention column, β, to guide the model towards the base shape. This new column, β, has the effect that when the handle influences are small, the base shape control, β, will dominate the softmax normalization, resulting in a strong bias towards inferring the base shape. Conversely, when handle influences are large, their attention values will dominate. Note that in the extreme case of having only a single handle, without β, the softmax normalization results in an equal influence over all vertices within the radius. However, adding β, which effectively is a constraint active at all times, resolves this issue of normalizing over a single entry, thus, properly maintaining the actual weight of the influence mask.

Adding β results in a dimensionality discrepancy in the attention matrix. To ensure dimensional consistency, we append an extra row of zeros to the input *value*. Note that choosing zeros is zero-preserving, and prevents the added β column from introducing any unwanted deformation. More formally, the attention output o is now given by:

$$o = \sigma_M([A;\beta]) * [v;0] \; . \tag{4}$$

Note that we keep β fixed during training, but can be controlled by the user to control the amount of bias towards the base shape when editing. In a practical implementation, we additionally clamp the influence of a handle to zero if it is below a small threshold (in our case 10^{-2}).

Shape-preserving loss. We further enforce a bias towards preserving the base shape by introducing a shape-preserving loss, L_{sp}, which smoothly blends the base and target shapes, guiding the model to respect previous edits while still learning the desired modifications. The loss is defined as:

$$L_{sp} = \|S - (S_{\text{pred}} \odot \hat{M} + S_B \odot (1 - \hat{M}))\|_2^2 \; . \tag{5}$$

Here, $\hat{M}$ is a per-vertex blend mask where values range from 0 to 1 and $\odot$ denoting element-wise multiplication. The value of $\hat{M}$ for each vertex is determined by the maximum influence of all active handles. This ensures that regions highly influenced by user edits (where $\hat{M}$ is close to 1) are predominantly learned from the predicted shape, while regions with minimal handle influence (where $\hat{M}$ is close to 0) are biased towards the base shape.

4. Scalability to High-Resolution Meshes

While meshes typically encountered in feature animation (see Figure 2 and Figure 4) have a fairly low resolution (≈6,000 vertices), in a live action setting, the shapes coming from studio capture setups [BHB*11] feature much higher resolution details and intricate wrinkles (≈100,000 vertices). Representing these details in the attention system would require a drastic increase in the network capacity. To keep the system lightweight and real-time, we instead adopt local patch blendshapes as an intermediate representation.

Patch blendshapes (PBS) [CCGB22] is a local blendshape system capable of representing a wide range of expressions of a character

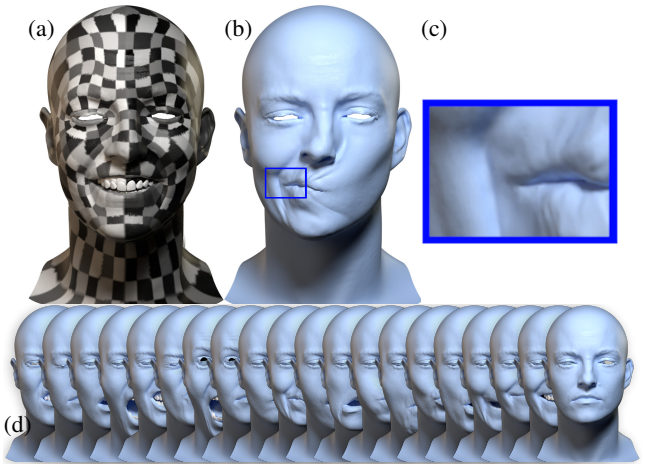


Figure 5: *Patch Blendshapes (PBS) [CCGB22] split the face into small regions (a), each equipped with a small blendshape system derived from 20 basis shapes (d). Predicting patch blend weights instead of per-vertex positions allows our method to scale to high-resolution meshes (b) that can preserve high-frequency details (c) from the input scans (d). ©Disney*

and capturing fine details from just a small number B of basis shapes. Its core idea is to split the face into P small regions, or patches, as shown in Figure 5a. Each patch then represents a small blendshape system of dimension B, with local basis shapes extracted from the complete B basis shapes (see Figure 5d). To avoid seams between the patches, they are expanded by three vertex rings and the positions are smoothly interpolated in this overlapping region, resulting in the final shape Figure 5b.

The locality and ease of setup inherent to PBS align perfectly with our goals, making it an ideal candidate for adapting our model to high-resolution meshes. In our implementation, we follow Chandran *et al.* [CCGB22] with $P = 457$ patches and $B = 19$ basis shapes. The B basis shapes are extreme expressions extracted from the training data along with a single neutral shape (the rightmost in Figure 5d).

Formally, the proposed local attention encoder predicts an output shape $S_{\text{pred}} \in \mathbb{R}^{P\times B}$, i.e. we treat the patches as "vertices" and the blendshape dimension as spatial dimension. We convert the predicted patch blendshapes into per-vertex blendshapes $S_{\text{vbs}} \in \mathbb{R}^{N\times B}$,

$$S_{\text{vbs}} = \mathbf{W}_{\text{layout}} S_{\text{pred}} \,, \tag{6}$$

where $W_{\text{layout}} \in \mathbb{R}^{N\times P}$ is a sparse matrix that encodes how much each vertex i is influenced by patch j given the current patch layout. By representing the canonical, neutral face as a vector $\mathbf{S}_0 \in \mathbb{R}^{3N}$ and combining the B basis shapes into a matrix $\mathbf{S}_{\text{basis}} \in \mathbb{R}^{B\times 3N}$ of offsets to the neutral shape, we can reconstruct the final shape as follows:

$$S_{\text{output}} = \mathbf{S}_{\text{neutral}} + S_{\text{vbs}}\mathbf{S}_{\text{basis}} \,. \tag{7}$$

The advantage of the PBS system over direct vertex prediction for high-resolution meshes is that small details like expression wrinkles in the eye or mouth region (see Figure 5c) do not need to be learned by the attention networks—they are contained in the basis shapes $\mathbf{S}_{\text{basis}}$. Note that since we define the basis shapes as offsets to the neutral shape, an all-zero network prediction $S_{\text{pred}} = 0$ will result in exactly the neutral shape, playing well together with the *zero-preservation* introduced in Section 3.1. In the *Iterative Editing* scenario, the attention mechanism predicts a delta in PBS shape over the base shape expressed as PBS weights.

Furthermore, since the encoder operates on a patch level instead of a vertex level and the number of queries is an order of magnitude lower than even the low-resolution case, we replace Φ^*_{output} by a dedicated per-patch MLP. This allows it to independently learn how to transform the latent information from the handle into its patch blend weight versus a global, implicit MLP that is conditioned on the canonical mesh. While the per-patch MLP increases the number of trainable weights by a factor of P, the impact on the inference time is negligible and it produces more consistent movement prediction. We refer to Section 6.4 for a comparison. For training, we combine the MSE loss on the predicted shapes from Section 3.1 with a regularizer on the PBS weights:

$$\mathcal{L} = \|S_{\text{shape-gt}} - S_{\text{shape-pred}}\|_2^2 + \|S_{\text{pbs-gt}} - S_{\text{pbs-pred}}\|_2^2 \,. \tag{8}$$

5. Training and Implementation Details

All of our models are trained on proprietary subject-specific datasets of facial motion. This includes hand-crafted motion for animated characters sourced from production movies, consisting of dialogue-heavy frames and performance-captured motion for live-action characters, which is highly diverse and includes expressive facial workout routines. Table 2 shows the number of frames for each character. Notice how dataset sizes and mesh resolutions can vary, highlighting the robustness and generality of our method. Table 3 also shows the effect of dataset size on the model reconstruction error. This implies that if a limited animation dataset were similarly expressive, such a large volume would not be necessary.

Table 2: *The size of the dataset (i.e. number of training shapes) for each character*

Type	ID	Dataset Size	Mesh Resolution
Animation	A1	100478	6032
Animation	A2	50526	6323
Animation	A3	70034	15000
Live Action	L1	1010	95035
Live Action	L2	679	95035
Live Action	L3	773	95035

To train our network, we sample random vertices to be our control handles, with varying influence regions. The network is trained to reconstruct a target face from these handles. For shape-preservation, we sample a random base shape from the dataset 90% of the time. The remaining 10% of the time, we use the previous frame of the target shape as the base shape to expose the model to more subtle deformations.

We set the input and output latent dimensions of the attention layer to 64. Similarly, each MLP within the network has a latent dimension

Table 3: *The impact of training dataset size on the reconstruction error using 1000 random and extreme samples for one character. As the dataset size increases, the error consistently decreases. Furthermore, by training on a smaller but more expressive dataset (similar to the live-action case) and removing redundant frames, we can achieve a comparable level of error.*

Training Data Size	Test Reconstruction Euclidean Error	
	Random Samples	Extreme Samples
1000	0.540mm	0.845mm
1000 Expressive	0.481mm	0.713mm
5000	0.483mm	0.759mm
10000	0.465mm	0.739mm
50000	0.458mm	0.713mm
100000	0.454mm	0.710mm

of 64 and a total of 3 layers each. We use the Tanh activation function for Φ_{output} and the GELU activation function [HG16] for other MLPs (we refer to Section 6.2 for the detailed explanation of why Tanh was chosen for Φ_{output}). We use Adam [KB15] as the optimizer with a learning rate of 10^{-3}. The networks typically converge after around one hour of training time on an NVIDIA RTX 4090 GPU, depending on the character and number of available frames. The model's inference time is 9 ms for low-resolution animation data and 23 ms for high-resolution live-action meshes on the NVIDIA RTX 4090 GPU. This demonstrates the model's capability to be integrated into commercial editing software tools.

6. Ablations and Comparisons

In this section, we ablate our models to evaluate the impact of different design choices made throughout the architecture. This includes evaluating the effects of zero-preservation, the selected activation functions, number of control handles, and the modulation method introduced in our approach. Furthermore, we quantitatively compare our method against Shape Transformer [CZG*22], one of the state-of-the art models in shape modeling.

6.1. Preserving Zero Edits

Section 3.1 introduced *zero-preserving* MLPs to avoid jumps if new handles are added. Figure 6 shows that with a bias in the MLP, jumps are clearly visible when new handles are added, even if they have a zero deformation. Designing the MLPs to be *zero-preserving* avoids these unwanted deformations.

6.2. Activation Function

Figure 7 compares the use of GELU or Tanh as the activation function for the output branch. While we found that the choice of activation function has no impact on the reconstruction metrics on test data, we observed a strong influence for out-of-distribution data, such as large manual edits from users. Due to GELU being unbounded, large edits also lead to unbounded deformations, see Figure 7a. In contrast, Tanh restricts the output to the range $[-1,+1]$. Hence, using it in the inner layers of Φ^*_{output} constrains the network's

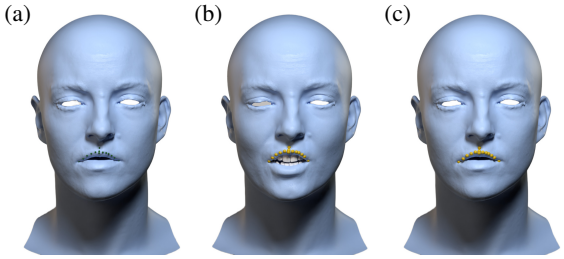
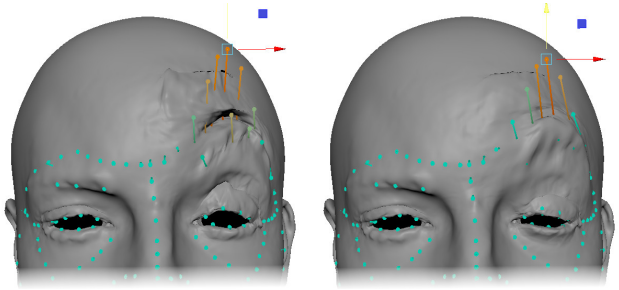


Figure 6: *When modifying a base shape (a) without zero-preserving MLPs, the shape of the mouth changes suddenly as soon as the user activates the handles in yellow (b). By designing MLPs to preserve zero displacements, activating new controls preserves the shape of the original mouth (c). ©Disney*



(a) Only GELU (b) Tanh in the output

Figure 7: *Comparison of the activation function for the output MLP: While having no impact on the reconstruction quality, using Tanh (b) instead of GELU (a) for the output MLP limits the range of motion to a more reasonable scale for extreme user edits. ©Disney*

predictions within a bounded region and prevents deformations from producing extreme out-of-distribution poses that could introduce artifacts, see Figure 7b.

6.3. Handles

The control handles and radii in our model can be assigned to any arbitrary vertex and value, as the model was exposed to random pairs during training. However, transitioning from a fixed starting pose A to a fixed target pose B using varying numbers of handles and radii may result in different levels of accuracy. Table 4 shows a comparison of the results when reaching the target pose with different handle and radius settings. This suggests that to accurately reconstruct the target pose, a higher number of handles with smaller radii is required. However, beginning the edit process with a small number of handles and a large radius provides a solid initial reconstruction, which can then be fine-tuned with smaller radius to reach the target pose more accurate.

6.4. Modulation

The Output MLP of our model (see Section 3.1 and Figure 2) processes the output of the attention layer and predicts the final deformation. Figure 8 shows the behavior if a single handle is edited, as

Table 4: *The impact of number of control handles and edit radius on the reconstruction error using 1000 random samples for one character. As we increase the number of control handles, the error decreases, demonstrating that more handles allow for a more accurate reconstruction of the target face, specifically the fined grained details. Furthermore, choosing an excessively small radius hinders the reconstruction, as fewer handles are able to influence each point on the face mesh.*

No. Handles	Test Reconstruction Euclidean Error		
	Small	Medium	Large
20	0.96mm	0.75mm	0.71mm
75	0.62mm	0.48mm	0.49mm
200	0.51mm	0.46mm	0.47mm

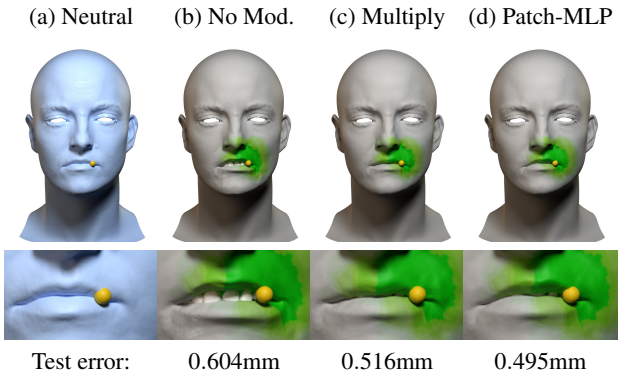


Figure 8: *Comparison of different ways to condition the implicit Output-MLP with the input shape, visualized qualitatively using a single-handle edit and quantitatively using the reconstruction loss on the test set: (b) no modulation of Φ^*_{output}, (c) multiplying the base shape as in Section 3.2, per-patch MLPs as in Section 4. ©Disney*

well as overall reconstruction loss on the test set for different configurations of conditioning this MLP. First, multiplying the canonical shape position, see Section 3.2, leads to a more plausible mouth shape (Figure 8c vs. b). Second, for the PBS-parameterized faces in a live-action setting, Section 4, using per-patch MLPs further reduces the reconstruction loss (Figure 8d vs. c).

6.5. Comparisons to Shape Transformer

As the closest method to CANRig in terms of input modality, Shape Transformer [CZG*22] also reconstructs a full shape from sparse input handles. Table 5 shows the reconstruction quality of the validation set using 265 uniformly distributed handles. Since Shape Transformer does not support iterative editing, all handles are defined relative to a canonical shape, we also disable the base shape in CANRig here. As seen in the table, our method consistently outperforms Shape Transformer with a $2-3\times$ lower Euclidean reconstruction error. Additionally, Shape Transformer required 14 hours of training time vs. CANRig with only 1 hour.

Table 5: *Quantitative comparison on the reconstruction quality from 265 uniformly distributed handles on the held-out test set for three live-action datasets. Our proposed method consistently reconstructs the shapes with a lower average Euclidean error (the standard deviation is shown in brackets) than the Shape Transformer by Chandran et al. [CZG*22]. For a fair comparison, we disable iterative editing and reconstruct relative to the canonical shape.*

Subject	# shapes	Shape Transformer	CANRig (Ours)
L1	90	0.652*mm* (0.194)	0.285*mm* (0.060)
L2	86	0.663*mm* (0.226)	0.326*mm* (0.081)
L3	90	0.405*mm* (0.065)	0.163*mm* (0.032)

7. Results

To evaluate our model's capabilities, we designed use cases that simulate various scenarios an artist might encounter when using our system. Together, these scenarios examine both the technical performance of the model and its practical usability in production. In the following subsections, we present each use case in detail.

7.1. Pose A to Pose B Workflow

In this experiment, our objective is to deform an initial pose A, into a target pose B. This use case measures how accurately the model transforms one state into another, as well as how easily a user can generate animations between the two poses. Figure 9 shows several examples of the transformations across 6 different characters. The accompanying video further demonstrates the complete process of transitioning from pose A to pose B. Our model can accurately reconstruct the target shapes with a relative low error, showing the effectiveness of our method and its ability to capture fine-grained geometric details using an arbitrary set of handles.

7.2. Clip Editing

Another use case is clip editing, which has significant applications in both animation and live-action production. After a performance is captured or animated, artists may need to adjust or correct specific frames within a longer clip. To achieve this, a user can define a region of interest using handles and an influence mask, perform the desired edit on that region, and then seamlessly integrate the modified frame back into the clip. This scenario highlights our model's capability for targeted frame-level manipulation within a video sequence, where it can smoothly integrate an edited frame while preserving temporal coherence. This capability is demonstrated in two production scenarios: one showing single-frame editing and smooth integration in a motion-capture sequence (Figure 10), and the other illustrating dialogue replacement from a source to a target sequence (Figure 11).

7.3. Clip Editing – Comparison to Related Methods

To compare our method quantitatively to related works, we repeat the clip editing scenario using two other methods: Shapeformer [CZG*22] (adjusted to the full-head topology) and global blendshapes (GBS). First, by using the iterative layering, CANRig

Figure 9: *Demonstration of the A-to-B workflow: For each subject, we demonstrate in two situations a base shape A (column 1 and 4) can be edited using the displayed handles to arrive at the target shape B (column 3 and 6). The network prediction is given in column 2 and 5, colored with the Euclidean error to the target shape, using a scale from 0mm 10mm. The rows follow the ordering of Table 2. ©Disney/Pixar*

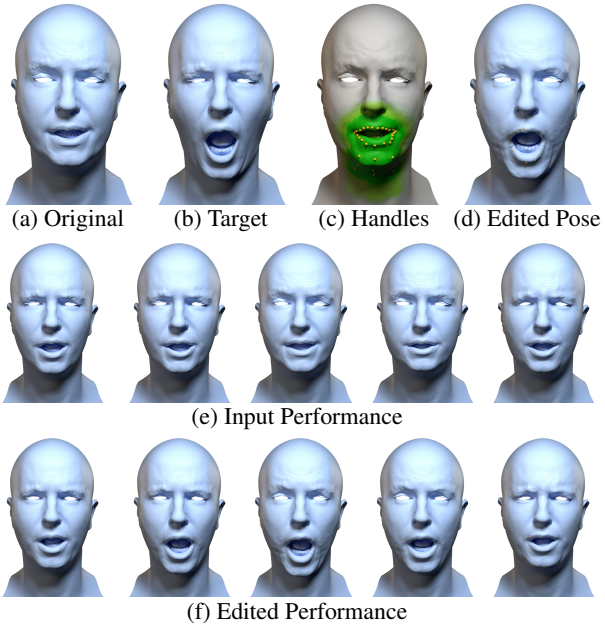


Figure 10: *A common task in production is to start from a captured sequence (e) and edit a single frame (a), e.g. to make it more readable or expressive. In this example we want to open the mouth wider as in shape (b). For that purpose, we select handles around the mouth and chin (c) and move them towards the target pose, giving us shape (e) that fully preserves the non-edited regions. This edit can then be smoothly integrated into the performance (f) by fading the handle edit in and out. We refer to the video for the full sequence. ©Disney*

Table 6: *Quantitative comparison between CANRig, ShapeTransformer [CZG*22] and global blendshapes (GBS) for dialog replacement. Averaged over the test sequence we measure the error to invert the shape into the rig, the difference to the target mouth shape, and the change to the rest of the face. Images for an example frame and the masks used to evaluate the differences can be found in Figure 12.*

Task	CANRig	Shapeformer	GBS
Rig Inversion	0.0*mm**	0.98*mm*	0.38*mm*
Difference to target mouth	1.01*mm*	2.14*mm*	1.43*mm*
Change to the rest of the face	0.0*mm*	0.35*mm*	0.28*mm*

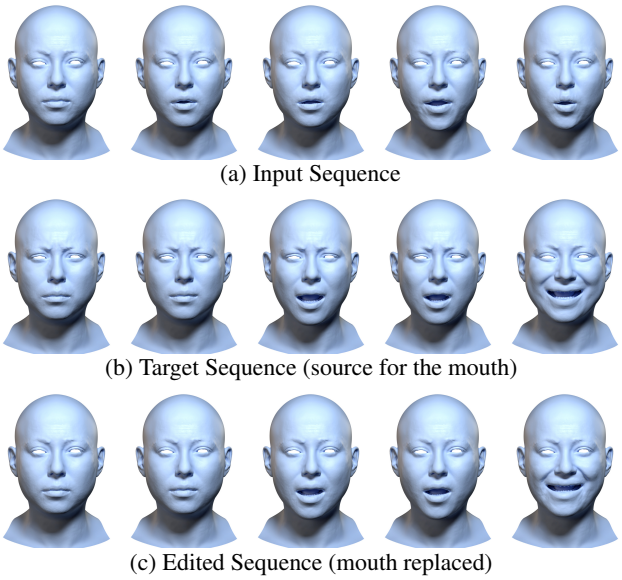
*Assuming PBS is used for face reconstruction as well.

simply uses the input shape as the base shape, leading to zero rig inversion error. Shapeformer and GBS use landmarks to deform the neutral face to the input shape, leading to up to 1mm error (Figure 12 and Table 6, 1st row). Next, the mouth area is replaced by adding new handles (CANRig) or moving existing ones (Shapeformer, GBS), where CANRig again leads to the smallest difference to the target mouth shape since the handles don't conflict with existing ones of the rest of the face. Finally, since CANRig is local to the influence region of the handles, anything outside of the influence region (see Figure 12 1st column, 3rd row) is guaranteed to not change the base shape. Shapeformer and GBS, however, are global and introduce changes to the rest of the face. Interestingly, since the global blendshapes (the same as for the PBS decoder) are chosen to be as semantically separate, GBS actually performs better in the locality tests than Shapeformer.

7.4. Shape-preserving Iterative Editing

In this experiment, we demonstrate the capability of our model for iterative edits that preserve the deformations of the starting shape. First, we show a simple scenario in Figure 13. By starting from different base shapes and applying an identical edit, using only a few local handles, our model smoothly integrates the local deformation into the edited region, while preserving the non-edited region of the base shapes in the mouth region.

Then, to highlight the non-destructive property of our approach more systematically, we re-evaluate the Pose A to Pose B experiment in an iterative manner. As shown in Figure 14, after each edit, the reconstruction error decreases, leading to an increasingly accurate reconstruction. This improvement is especially significant for poses, where target shape is far from the initial shape or even out-of-distribution from the training data, e.g. row 1 and 4. Quantitatively, we show this trend of iteratively improving the edit in



(a) Input Sequence

(b) Target Sequence (source for the mouth)

(c) Edited Sequence (mouth replaced)

Figure 11: *Given an input sequence (a), we want to replace the mouth with the shapes from the target sequence (b), e.g. for dialog replacement in dubbing. We use the same pipeline as in Figure 10, i.e. select a set of handles around the mouth and move them to the target shape. Here, we apply it to every frame, resulting in the edited sequence (c). We refer to the video for the full sequence. ©Disney*

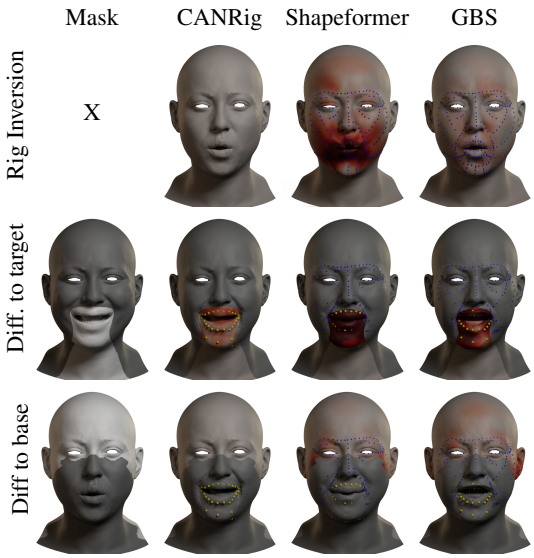


Figure 12: *Comparison to related methods on frame 40 of the test sequence with replaced mouth (Figure 11, last column). First column: target/base shape with masks to compute the differences as reported in Table 6. All error images use a scale from 0mm 5mm. ©Disney*

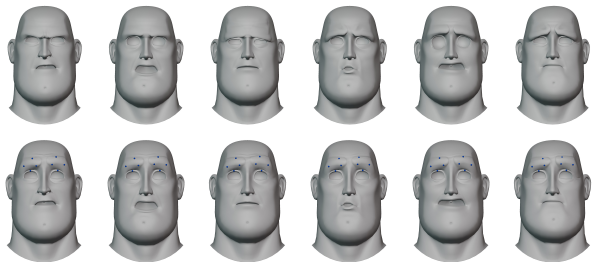


Figure 13: *Starting with 6 distinct base shapes (1st row), applying an identical local edit (eyebrow raise) with a few handles, results in distinct final shapes (2nd row). Note that the deformation smoothly integrated into the editing region, while the non-edited parts, such as the mouth and chins region remain unchanged. ©Disney/Pixar*

Figure 15. The error decreases with each edit and converges with a residual error that could potentially be removed with a more fine grained handles outline.

7.5. Artist Feedback

CANRig is currently being tested for production and below is the feedback we received from artists so far:

Advantages

- The implementation is very rigging friendly.
- The handles are very interesting to manipulate in ways that normal controls cannot offer.
- It is very exciting that the deformations are highly specific to the actor.
- It is nice that the handles have adjustable areas of influence.

Limitations

- While some areas succeed greatly at adhering to their handles (such as the lips), other areas may need a small “cleanup” pass, such as the eyelids.
- The rig is a little opaque as to how the controls affect the mesh since it is data-driven.

8. Limitations and Discussion

Our simple yet effective approach to learning a direct manipulation rig via attention enables fast posing and local refinement. However, for creating a complex pose from scratch, artists still need to select many handles and managing them simultaneously can be time-consuming. While our iterative editing capability allows to break this down into simpler steps, a fundamental challenge for the artist is selecting the right handles. Therefore, in the future, when integrating CANRig into existing artist tools, we envision ways to combine our handles into semantically meaningful groups that can be controlled together. For instance, a jaw bone could be used to move multiple handles around the jaw, or parametric curves that outline the lips and eyes could help artists to move the handles in a coordinated manner.

The introduction of the base shape enables the artist to do iterative editing. While this is a core feature of our proposed system, it holds

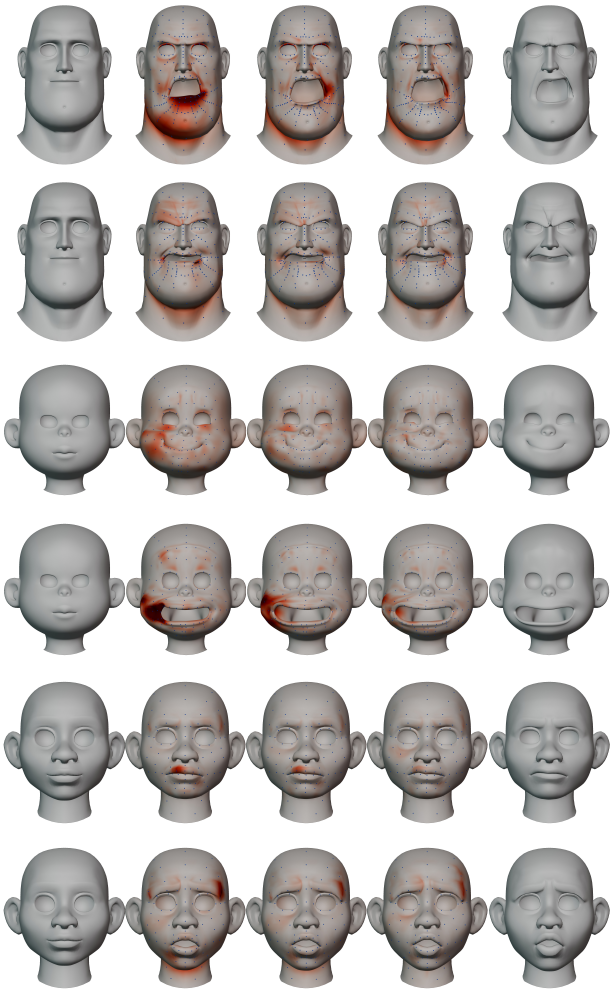


Figure 14: *The effect of shape-preserving iterative editing. Starting from the neutral face (column 1), using the displayed handles, our system predicts the target face (column 5) with a single edit (column 2). By updating the base shape to the intermediate results, additional edits can be applied, leading to a more accurate reconstruction of the target face. The Euclidean reconstruction error relative to the target face is represented by color, using a scale from 0mm 10mm. ©Disney/Pixar*

its own limitations. When the artist chooses a base shape that is conflicting too much with the desired pose, the bias might be too strong, requiring the artist either to lower the bias strength or change the base shape. While both options are supported in CANRig, it comes at the cost of additional authoring.

One of the strengths of CANRig is the ability to stay in model and prohibit unfeasible shapes, which means the deformations do not break the model and preserve the character's identity. This is a crucial requirement in the live-action VFX scenario. On the other hand, in the animation domain, artists often need to push the rig beyond the range of the training data. While we have seen that the iterative editing workflow allows us to reach more extreme poses in

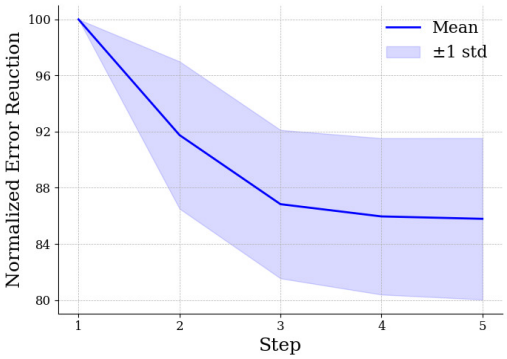


Figure 15: *The plot shows the normalized reduction of reconstruction error (L2 distance) using the shape-preserving iterative editing approach. Errors are normalized by the error after the first edit and averaged over 1,000 samples. Because the first edit already achieves a strong reconstruction, we highlight the improvements from subsequent edits. This result demonstrate that our model can further reduce the error until convergence using multiple layers of edits, while remaining non-destructive, as the error does not increase with additional iterations.*

multiple steps, control over very extreme poses is still a challenge for our rig.

9. Conclusion and Future Work

Creating facial expressions for animated and live-action characters has always been a complex task in movie and video game production. Developing the rigs used by artists requires skilled modelers, and the rigs may lack intuitive controls.

In this work, we presented CANRig, an automated direct manipulation rigging method that supports dynamic selection of the editing region, enabling both fine and coarse deformations and offering artists precise control over their work. We achieve this by formulating deformation as a cross-attention between handles and user-defined mesh regions at runtime.

Furthermore, to reduce the amount of control manipulation, as well as to reduce the error of inversion to zero with captured performances, we introduced an additional shape-preserving architecture. This helps to preserve previous edits and performances while still offering the flexibility to change the shape.

While our work focused on creating a dedicated neural rig for each character, we believe our rig can be generalized to multiple characters. In the future, we plan on extending CANRig to retarget a character's performance onto another, or to adapt a pre-trained version of CANRig to a new character for which only a small number of training shapes exist; as is often the case with new animation productions and fictional characters, which are manually sculpted. Broadening the generalizability of these results to different characters is beyond the scope of this paper and represents a promising direction for future work.

Acknowledgments

The authors would like to thank Disney/Pixar for providing the data used in this work, with special thanks to Arnold Moon and Mark Meyer for their assistance. Note CANRIG was not used in the making of ©Pixar movies. This project was partially supported by Innosuisse and we are grateful for their support.

References

[AC24] ARCELIN B., CHAVEROU N.: Audio2Rig: Artist-oriented deep learning tool for facial and lip sync animation. In *ACM SIGGRAPH 2024 Talks* (2024), ACM. 2

[AGB*23] AGRAWAL D., GUAY M., BUHMANN J., BORER D., SUMNER R. W.: Pose and skeleton-aware neural IK for pose and motion editing. In *SIGGRAPH Asia 2023 Conference Papers* (2023), ACM. 2, 5

[ATDN23] ANEJA S., THIES J., DAI A., NIESSNER M.: ClipFace: Text-guided editing of textured 3D morphable models. In *SIGGRAPH '23 Conference Proceedings* (2023), ACM. doi:10.1145/3588432.3591566. 2

[BCR*21] BAI Z., CUI Z., RAHIM J. A., LIU X., TAN P.: Riggable 3D face reconstruction via in-network optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2021), pp. 6216–6225. 2

[BHB*11] BEELER T., HAHN F., BRADLEY D., BICKEL B., BEARDSLEY P., GOTSMAN C., SUMNER R. W., GROSS M.: High-quality passive facial performance capture using anchor frames. *ACM Transactions on Graphics (TOG) 30*, 4 (2011), 75:1–75:10. 5

[BV99] BLANZ V., VETTER T.: A morphable model for the synthesis of 3D faces. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques* (1999), ACM, pp. 187–194. 2

[CCGB22] CHANDRAN P., CICCONE L., GROSS M., BRADLEY D.: Local anatomically-constrained facial performance retargeting. *ACM Transactions on Graphics (TOG) 41*, 4 (2022), 1–14. 2, 5, 6

[CEM*22] CHOI B., EOM H., MOUSCADET B., CULLINGFORD S., MA K., GASSEL S., KIM S., MOFFAT A., MAIER M., REVELANT M., LETTERI J., SINGH K.: Animatomy: An animator-centric, anatomically inspired system for 3D facial modeling, animation and transfer. In *SIGGRAPH Asia 2022 Conference Papers* (2022), ACM. 3

[CGB20] CHANDRAN P., GROSS M., BEELER T.: Semantic deep face models. In *2020 International Conference on 3D Vision (3DV)* (2020), IEEE, pp. 345–354. 2

[COBG23] CHEN C., O'TOOLE M., BHARAJ G., GARRIDO P.: Implicit neural head synthesis via controllable local deformation fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2023), pp. 416–426. 2

[CPH*25] CORIGLIANO D., PETER D., HUBER N. B., THOMASZEWSKI B., SOLENTHALER B.: NeuRiPhy: Neural baking of physics-based deformations for facial rigs. *Proceedings of the ACM on Computer Graphics and Interactive Techniques 8*, 4 (2025), 59:1–59:18. 2

[CYSN25] CHA S., YOON S., SEO K., NOH J.: Neural face skinning for mesh-agnostic facial expression cloning. *Computer Graphics Forum 44*, 2 (2025), e70009. 3

[CZG*22] CHANDRAN P., ZOSS G., GROSS M., GOTARDO P., BRADLEY D.: Shape transformers: Topology-independent 3D shape models using transformers. *Computer Graphics Forum 41*, 2 (2022), 195–207. 3, 7, 8, 9

[DZX*23] DING Z., ZHANG X., XIA Z., JEBE L., TU Z., ZHANG X.: DiffusionRig: Learning personalized priors for facial appearance editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2023), pp. 12736–12746. 2

[EF78] EKMAN P., FRIESEN W. V.: *Facial Action Coding System: A Technique for the Measurement of Facial Movement*. Consulting Psychologists Press, Palo Alto, CA, 1978. 3

[FFBB21] FENG Y., FENG H., BLACK M. J., BOLKART T.: Learning an animatable detailed 3D face model from in-the-wild images. *ACM Transactions on Graphics (TOG) 40*, 4 (2021), 88:1–88:13. 2

[GKG*23] GIEBENHAIN S., KIRSCHSTEIN T., GEORGOPOULOS M., RÜNZ M., AGAPITO L., NIESSNER M.: Learning neural parametric head models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2023), pp. 21003–21012. 2

[GXM*25] GUO Z., XIANG J., MA K., ZHOU W., LI H., ZHANG R.: Make-It-Animatable: An efficient framework for authoring animation-ready 3D characters. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2025), pp. 10783–10792. 2

[GZC*16] GARRIDO P., ZOLLHÖFER M., CASAS D., VALGAERTS L., VARANASI K., PÉREZ P., THEOBALT C.: Reconstruction of personalized 3D face rigs from monocular video. *ACM Transactions on Graphics (TOG) 35*, 3 (2016), 28:1–28:15. 2

[Hay94] HAYKIN S.: *Neural Networks: A Comprehensive Foundation*. Prentice Hall PTR, 1994. 4

[HG16] HENDRYCKS D., GIMPEL K.: Gaussian error linear units (GELUs). *arXiv preprint arXiv:1606.08415* (2016). 4, 7

[KB15] KINGMA D. P., BA J.: Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)* (2015). 7

[KBB*17] KOZLOV Y., BRADLEY D., BÄCHER M., THOMASZEWSKI B., BEELER T., GROSS M.: Enriching facial blendshape rigs with physical simulation. *Computer Graphics Forum 36*, 2 (2017), 75–84. 3

[LA10] LEWIS J. P., ANJYO K.-I.: Direct manipulation blendshapes. *IEEE Computer Graphics and Applications 30*, 4 (July-Aug. 2010), 42–50. doi:10.1109/MCG.2010.41. 3

[LAR*14] LEWIS J., ANJYO K., RHEE T., ZHANG M., PIGHIN F., DENG Z.: Practice and theory of blendshape facial models. *Eurographics 2014 - State of the Art Reports* (2014). 3

[LBB*17] LI T., BOLKART T., BLACK M. J., LI H., ROMERO J.: Learning a model of facial shape and expression from 4D scans. *ACM Transactions on Graphics (TOG) 36*, 6 (2017), 194:1–194:17. 2

[LCXS09] LAU M., CHAI J., XU Y.-Q., SHUM H.-Y.: Face poser: Interactive modeling of 3D facial expressions using facial priors. *ACM Transactions on Graphics (TOG) 29*, 1 (2009), 3:1–3:17. 3

[LWP10] LI H., WEISE T., PAULY M.: Example-based facial rigging. *ACM Transactions on Graphics (TOG) 29*, 4 (2010), 32:1–32:6. 3

[MCY19] MA C., CHEN L., YONG J.: AU R-CNN: Encoding expert prior knowledge into R-CNN for action unit detection. *Neurocomputing 355* (2019), 35–47. 2

[NVW*13] NEUMANN T., VARANASI K., WENGER S., WACKER M., MAGNOR M., THEOBALT C.: Sparse localized deformation components. *ACM Transactions on Graphics (TOG) 32*, 6 (2013), 179:1–179:10. 3

[OBP*12] ORVALHO V., BASTOS P., PARKE F., OLIVEIRA B., ALVAREZ X.: A facial rigging survey. In *Eurographics 2012 - State of the Art Reports* (2012), The Eurographics Association, pp. 183–204. 3

[PTPZ24] POTAMIAS R. A., TARASIOU M., PLOUMPIS S., ZAFEIRIOU S.: ShapeFusion: A 3D diffusion model for localized shape editing. In *Proceedings of the European Conference on Computer Vision (ECCV)* (2024), Springer. 3

[QSA*23] QIN D., SAITO J., AIGERMAN N., GROUEIX T., KOMURA T.: Neural face rigging for animating and retargeting facial meshes in the wild. In *SIGGRAPH '23 Conference Proceedings* (2023), ACM. doi:10.1145/3588432.3591556. 2, 3

[RdGM20] RADZIHOVSKY S., DE GOES F., MEYER M.: FaceBaker: Baking character facial rigs with machine learning. In *ACM SIGGRAPH 2020 Talks* (2020), ACM, pp. 1–2. 2

[RSJ*21] RACKOVIC S., SOARES C., JAKOVETIC D., DESNICA Z., LJUBOBRATOVIC R.: Clustering of the blendshape facial model. In *Proceedings of the 29th European Signal Processing Conference (EUSIPCO)* (2021), IEEE, pp. 1556–1560. 2

[RTT*25] RAINA P., TAUBNER F., TULI M., TEH E. W., FERREIRA K.: PrismAvatar: Real-time animated 3D neural head avatars on edge devices. *arXiv preprint arXiv:2502.07030* (2025). 3

[SSR20] SONG S. L., SHI W., REED M.: Accurate face rig approximation with deep differential subspace reconstruction. *ACM Transactions on Graphics (TOG) 39*, 4 (2020), 34:1–34:13. 2

[TDlTM11] TENA J. R., DE LA TORRE F., MATTHEWS I.: Interactive region-based linear 3D face models. *ACM Transactions on Graphics (TOG) 30*, 4 (2011), 76:1–76:10. 3

[TPO*24] TARASIOU M., POTAMIAS R. A., O'SULLIVAN E., PLOUMPIS S., ZAFEIRIOU S.: Locally adaptive neural 3d morphable models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2024), pp. 1867–1876. 3

[VBPP05] VLASIC D., BRAND M., PFISTER H., POPOVIĆ J.: Face transfer with multilinear models. *ACM Transactions on Graphics (TOG) 24*, 3 (2005), 426–433. 2

[VRWW20] VESDAPUNT N., RUNDLE M., WU H., WANG B.: JNR: Joint-based neural rig representation for compact 3D face modeling. In *Proceedings of the European Conference on Computer Vision (ECCV)* (2020), Springer, pp. 389–405. 2

[XZK*20] XU Z., ZHOU Y., KALOGERAKIS E., LANDRETH C., SINGH K.: RigNet: Neural rigging for articulated characters. *ACM Transactions on Graphics (TOG) 39*, 4 (2020), 58:1–58:14. 3

[YLL*25] YAN H., LUO K., LI W., LIANG Y., LI S., HUANG J., GUO C., TAN P.: PoseMaster: Generating 3D characters in arbitrary poses from a single image. *arXiv preprint arXiv:2506.21076* (2025). 2

[YWW*25] YAN P., WARD R. K., WANG D., TANG Q., DU S.: Stylemorpheus: Learning a StyleGAN-based 3D-aware morphable face model with a disentangled style space. *Neurocomputing 654* (2025), 131329. 2

[ZWG*23] ZHAO Z., WENG D., GUO H., HOU J., ZHOU J.: Facial auto rigging from 4D expressions via skinning decomposition. In *Proceedings of the 31st ACM International Conference on Multimedia* (2023), pp. 6101–6109. 3

[ZZY*25] ZHENG M., ZHANG H., YANG H., CHEN L., HUANG D.: ImFace++: A sophisticated nonlinear 3D morphable face model with implicit neural representations. *IEEE Transactions on Pattern Analysis and Machine Intelligence 47*, 2 (2025), 994–1012. doi:10.1109/TPAMI.2024.3480151. 2

Appendix A: Choice of Distance Measure

As described in Section 3.1, we chose a weighted geodesic distance measure for determining the influence radius of a control handle. Empirically, this approach is better than straight-line Euclidean distances or unweighted geodesic distances because it better preserves both the spatial and topological relationships of the mesh. Especially, the topological relationship can be crucial in regions such as the face, where the upper and lower lip might be close together in space but should not move together for a small influence radius.

Additionally, we tested another way of determining the influence radius, based on a correlation prior. To do so, we precomputed the correlation $corr(v_i, v_j)$ between each vertex pair v_i and v_j. When integrating the weighted geodesic distance along the edges, we multiply this with an additional scaling of $1 - |corr(v_i, v_j)|$. With this scaling, not only are the topological and spatial relationships taken into account, but also their prior correlation based on the data. This results in more correlated vertices being covered first when increasing the radius of a single handle. Figure 16 illustrates this effect, showing how the influence region initially covers the forehead rather than the upper eyebrows, unlike the weighted geodesic approach in Figure 4. Also note how the nose is not covered at all since that deformation was very uncorrelated in the data and can only be moved by selecting a handle on the nose.

While the precomputed correlation could potentially simplify the task the model has to learn, we did not observe any improvements (or deficits) in performance. However, from a user perspective, this way of controlling the influence region offers an interesting property by more clearly visualizing which regions can or cannot be moved together. From a UI-perspective, the visualization of the attention weights can offer a similar feedback to the user to understand the influence of a control handle on the vertices. Nevertheless, since the computation of the correlation matrix is challenging for high-resolution meshes, we defaulted to the simpler weighted geodesic distance measure in CANRig. Yet, one can also expose this way of shaping the influence region as an additional option to the user at runtime, since CANRig is not limited to the masks it was trained with.

Appendix B: Detailed Equations

In this section, we present the full mathematical formulations used throughout this paper. We begin with the equations for the non-iterative model in Section 3.1, followed by the formulations for the iterative approach in Section 3.2.

$$q = \Phi_{\text{query}}(S_0) \qquad \in \mathbb{R}^{K\times N\times F_i} \qquad (9a)$$
$$k = \Phi_{\text{key}}(p_h) \qquad \in \mathbb{R}^{K\times H\times F_i} \qquad (9b)$$
$$v = \Phi^*_{\text{value}}(\Delta_h) \qquad \in \mathbb{R}^{K\times H\times F_o} \qquad (9c)$$
$$A = q * k^T \qquad \in \mathbb{R}^{K\times N\times H} \qquad (9d)$$
$$A_M = \sigma_M(A) = \frac{\exp(A)\odot M}{\sum_{l=1}^{L}\exp(A_h)M_h} \qquad \in \mathbb{R}^{K\times N\times H} \qquad (9e)$$
$$z = A_M * v \qquad \in \mathbb{R}^{K\times N\times F_o} \qquad (9f)$$
$$S_{pred} = S_0 + \Phi^*_{\text{output}}(z) \qquad \in \mathbb{R}^{N\times D} \qquad (9g)$$

Figure 16: *Deformations using a single control handle (blue) with two varying user-defined editing regions, indicated by 0 1, that are additionally weighted based on a correlation prior. ©Disney/Pixar*

Here, F_i, F_o are the attention input and output latent dimensions, and K is the number of attention heads.

Similarly, the equations for the shape-preserving model are given by:

$$q = \Phi_{\text{query}}(S_0) \qquad \in \mathbb{R}^{K\times N\times F_i} \qquad (10a)$$
$$k = \Phi_{\text{key}}(p_h) \qquad \in \mathbb{R}^{K\times H\times F_i} \qquad (10b)$$
$$v = \Phi^*_{\text{value}}(\Delta_h, b_h) \qquad \in \mathbb{R}^{K\times H\times F_o} \qquad (10c)$$
$$A = q * k^T \qquad \in \mathbb{R}^{K\times N\times H} \qquad (10d)$$
$$A_{M,\beta} = \sigma_M(A;\beta) = \left[\frac{\exp(A)\odot M}{\sum_{l=1}^{L}\exp(A_h)M_h};\beta\right] \qquad \in \mathbb{R}^{K\times N\times H+1} \qquad (10e)$$
$$z = A_{M,\beta} * [v;0] \qquad \in \mathbb{R}^{K\times N\times F_o} \qquad (10f)$$
$$S_{pred} = S_B + \Phi^*_{\text{output}}(z, S_B) \qquad \in \mathbb{R}^{N\times D} \qquad (10g)$$

Appendix C: Additional Results

Pose A to B First, we present additional visual results of the A to B experiment for our animated characters (see Figure 17). In all cases, the target pose B can be hit accurately. In some cases such as the one in fifth row, where the target pose is more extreme, the error might be higher. However, the residual error can be further improved by our iterative editing capabilities, as shown in the next section.

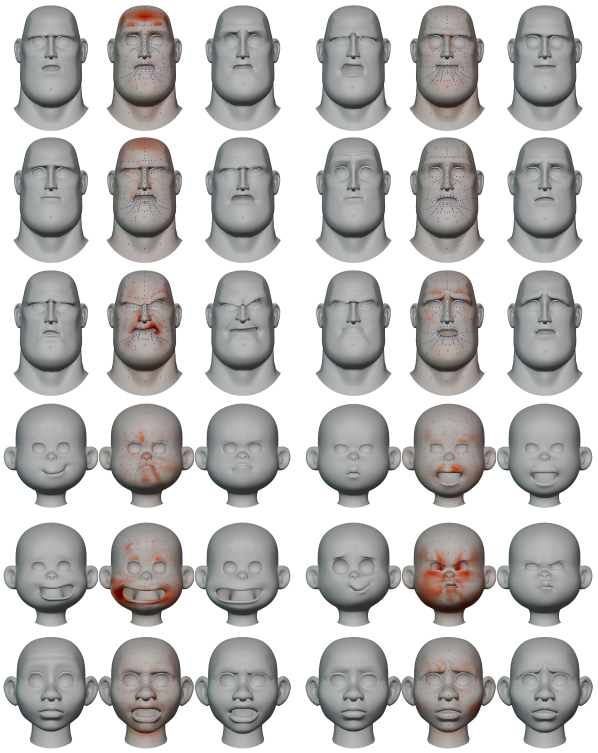


Figure 17: *Additional examples of the A to B test. The base shape is given in the first and fourth columns, the target pose in the third and sixth columns, and the reconstruction is given in the second and fifth columns. The error scale is from 0mm*  *10mm. ©Disney/Pixar*

Iterative Editing Secondly, we give additional visual results of the shape-preserving iterative editing in Figure 18). In all cases, we start from the neutral pose and reconstruct the target iteratively. By updating the base shape after each editing layer and bringing it closer to the target shape, the reconstruction error is reduced in each step. This is especially prominent for the second and fifth row that show more extreme target poses.

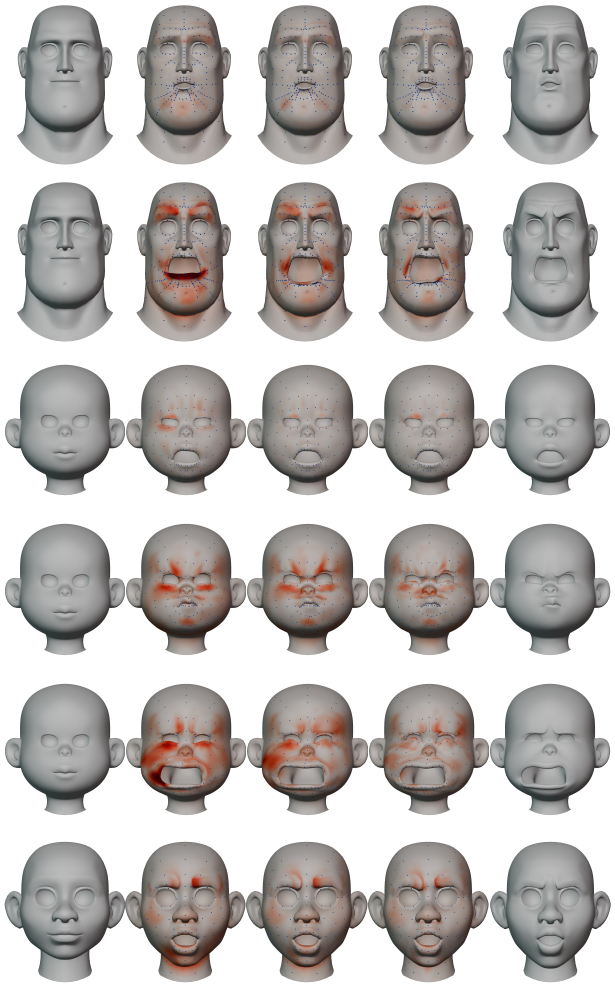


Figure 18: *Additional examples of iterative editing. Starting from the neutral face (column 1), our system predicts the target face (column 5). Column 2 shows the prediction with a single edit. By updating the base shape after each layer, additional edits can be applied, leading to a better reconstruction of the target face. The Euclidean reconstruction error relative to the target face is represented by color, using a scale from 0mm*  *10mm. ©Disney/Pixar*