Refined Inverse Rigging: A Balanced Approach to High-fidelity Blendshape Animation

STEVO RACKOVIĆ, University of Lisbon, Portugal
DUŠAN JAKOVETIĆ, University of Novi Sad, Serbia & Montenegro
CLÁUDIA SOARES, NOVA School of Science and Technology, Portugal

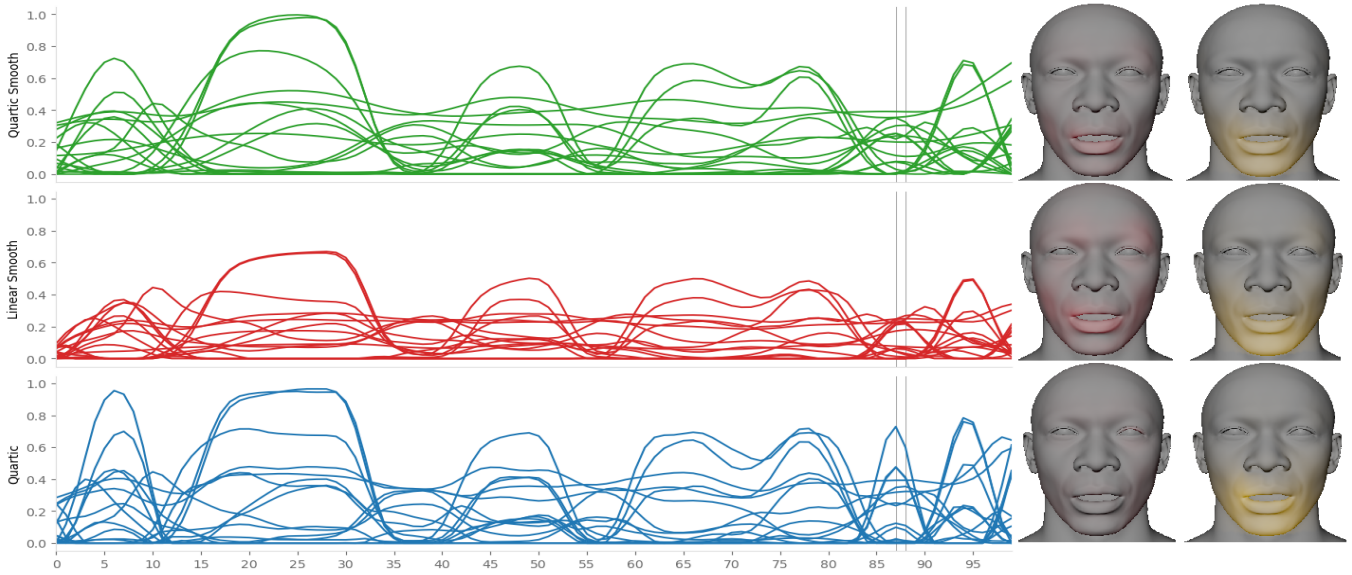

Fig. 1. Demonstrating the Efficacy of Temporally Coherent Blendshape Animation. **First row:** Our *Quartic Smooth* method captures the intricate dynamics of facial expressions by leveraging a sophisticated blendshape rig, ensuring both high-fidelity mesh reconstruction and smooth temporal transitions in animation weights. **Second row:** The *Linear Smooth* method, as proposed in [Lewis and Anjyo 2010], prioritizes temporal smoothness but simplifies the blendshape model to a linear function, resulting in a trade-off with mesh accuracy. **Third row:** The *Quartic* approach from [Racković et al. 2023] achieves a high degree of mesh fidelity by utilizing a complex blendshape model but does not account for the smoothness of frame-to-frame transitions, leading to potential discontinuities. Displayed are selected weight trajectories over 100 animation frames, with two consecutive frames magnified to showcase the mesh results. The second column employs red shading to illustrate mesh error, and the third column uses yellow to highlight discrepancies between successive frames, underscoring the balance between accuracy and smoothness in animation sequences.

In this paper, we present an advanced approach to solving the inverse rig problem in blendshape animation, using high-quality corrective blendshapes. Our algorithm focuses on three key areas: ensuring high data fidelity in reconstructed meshes, achieving greater sparsity in weight distributions, and facilitating smoother frame-to-frame transitions. While the incorporation of corrective terms is a known practice, our method differentiates itself by employing a unique combination of l_1 norm regularization for sparsity and a temporal smoothness constraint through roughness penalty, focusing on the sum of second differences in consecutive frame weights. A significant innovation in our approach is the temporal decoupling of blendshapes, which permits simultaneous optimization across entire animation sequences. This feature sets our work apart from existing methods and contributes to a more efficient and effective solution. Our algorithm exhibits a marked improvement in maintaining data fidelity and ensuring smooth frame transitions when compared to prior approaches that either lack smoothness regularization or rely solely on linear blendshape models. In addition to superior mesh resemblance and smoothness, our method offers practical benefits, including reduced computational complexity and execution time, achieved through a novel parallelization strategy using clustering methods. Our results not only advance the state-of-the-art in terms of fidelity, sparsity, and smoothness in inverse rigging but also introduce significant efficiency improvements[1].

CCS Concepts: • **Computing methodologies** → **Motion processing**; **Distributed algorithms**; Cross-validation.

Authors' Contact Information: Stevo Racković, University of Lisbon, Portugal, stevo.rackovic@tecnico.ulisboa.pt; Dušan Jakovetić, University of Novi Sad, Serbia & Montenegro, dusan.jakovetic@dmi.uns.ac.rs; Cláudia Soares, NOVA School of Science and Technology, Portugal, claudia.soares@fct.unl.pt.

SA Conference Papers '24, December 03–06, 2024, Tokyo, Japan

ACM ISBN 979-8-4007-1131-2/24/12
https://doi.org/10.1145/3680528.3687670

[1]The source code is made available at the provided GitHub repository github.com/stevorackovic/FacialAnimation/tree/master/Scripts/TimeAwareComponent

Additional Key Words and Phrases: blendshape animation, inverse rig problem, face segmentation

ACM Reference Format:
Stevo Racković, Dušan Jakovetić, and Cláudia Soares. 2024. Refined Inverse Rigging: A Balanced Approach to High-fidelity Blendshape Animation. In *SIGGRAPH Asia 2024 Conference Papers (SA Conference Papers '24), December 03–06, 2024, Tokyo, Japan.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3680528.3687670

1 Introduction

Blendshape animation, a predominant method in animating human faces, manipulates a 3D facial mesh $\mathbf{b}_0 \in \mathbb{R}^{3n}$ through interpolation among a predefined set of blendshapes $\mathbf{b}_1, \ldots, \mathbf{b}_m \in \mathbb{R}^{3n}$, where n denotes the number of mesh vertices [Lewis et al. 2014]. Representing diverse facial configurations, these blendshapes, when linearly combined with weights $\mathbf{w} = [w_1, \ldots, w_m]$, enable the creation of a wide range of expressions as

$$f_L(\mathbf{w}; \mathbf{B}) = \mathbf{b}_0 + \mathbf{B}\mathbf{w}, \tag{1}$$

with $\mathbf{B} \in \mathbb{R}^{3n \times m}$ forming a matrix of blendshape vectors. Here, the subscript L denotes the linear nature of this blendshape function. Modern advancements incorporate non-linear corrective terms into these models, enhancing realism and flexibility [Racković et al. 2023a]. For instance, a quartic blendshape function, denoted here as f_Q, integrates up to three levels of corrective terms:

$$\begin{aligned} f_Q(\mathbf{w}) = &\mathbf{B}\mathbf{w} + \sum_{\{i,j\} \in \mathcal{P}} w_i w_j \mathbf{b}^{\{ij\}} + \sum_{\{i,j,k\} \in \mathcal{T}} w_i w_j w_k \mathbf{b}^{\{ijk\}} \\ &+ \sum_{\{i,j,k,l\} \in Q} w_i w_j w_k w_l \mathbf{b}^{\{ijkl\}}. \end{aligned} \tag{2}$$

Such a detailed correction level is employed in industry-standard solutions like Metahumans[2].

This paper addresses the inverse rig problem: given a target mesh $\widehat{\mathbf{b}} \in \mathbb{R}^{3n}$, the goal is to find a set of weights $\mathbf{w}$ that accurately approximates the target. Solving the inverse rig problem is pertinent in animation, specifically for importing a 4D scanned facial animation performance into a blendshape system. It involves optimization techniques to reverse-engineer the desired output into the required inputs for the rig. The problem is vital in movie and video game production and in the increasingly popular virtual and augmented reality. Model-based solutions to this problem leverage the structure of rig functions, utilizing optimization techniques rather than relying purely on data [Çetinaslan 2016; Racković et al. 2023b]. The state-of-the-art (SOTA) model-based approach, as proposed in [Racković et al. 2023a], solves the problem by minimizing the data fidelity of the model while regularizing for sparsity and constraining the weight values to a feasible set considering that the weights only make sense on an interval $[0, 1]$,

$$\underset{\mathbf{0} \le \mathbf{w} \le \mathbf{1}}{\text{minimize}} \ \frac{1}{2} \| f_Q(\mathbf{w}) - \widehat{\mathbf{b}} \|_2^2 + \alpha \mathbf{1}^T \mathbf{w}, \tag{3}$$

employing a coordinate descent method. Prior methods [Cetinaslan and Orvalho 2020a; Lewis and Anjyo 2010; Seol et al. 2011], while effective, are confined to linear blendshape models and thus are less capable of replicating complex facial dynamics. However, a critical aspect in animation is the smoothness of frame-to-frame transitions, often overlooked in isolated frame fitting. This temporal dimension has been explored, for instance, in [Lewis and Anjyo 2010], who introduced a smoothness regularization:

$$\underset{\mathbf{0} \le \mathbf{w} \le \mathbf{1}}{\text{minimize}} \ \| f_L(\mathbf{w}) - \widehat{\mathbf{b}} \|_2^2 + \alpha \| \mathbf{w} \|_2^2 + \beta \| \mathbf{w} - \mathbf{v} \|_2^2, \tag{4}$$

where $\mathbf{v} \in \mathbb{R}^m$ represents the weight vector from the previous frame, introducing a temporal continuity constraint. While effective, these methods primarily address linear blendshape models, limiting their capability to capture the more nuanced facial expressions enabled by non-linear models. Additionally, [Tagliasacchi et al. 2015] addresses this aspect, yet their work targets hand-movements tracking, distant from our use-case due to many differences in skeletal versus blendshape models.

In this work, we bridge this gap by proposing a novel objective that harmoniously integrates both the advanced corrective terms of non-linear blendshape functions and the imperative of temporal smoothness. Our formulation extends beyond the scope of objective (4) by concurrently optimizing across all frames. This holistic approach not only ensures the fidelity of each individual frame to the target mesh but also guarantees the smoothness of transitions throughout the animation sequence, by formulating the problem

$$\begin{aligned} \underset{\mathbf{0} \le \mathbf{w} \le \mathbf{1}}{\text{minimize}} \ &\sum_{t=1}^{T} \left(\frac{1}{n} \| f_Q(\mathbf{w}^{(t)}) - \widehat{\mathbf{b}}^{(t)} \|_2^2 + \frac{\alpha}{m} \| \mathbf{w}^{(t)} \|_1 \right) \\ &+ \beta \sum_{t=2}^{T-1} \| \mathbf{w}^{(t+1)} - 2\mathbf{w}^{(t)} + \mathbf{w}^{(t-1)} \|_2^2, \end{aligned} \tag{5}$$

where T is the number of frames in the sequence, and β is a regularization parameter that balances data fidelity and smoothness constraints. Notably, our method is adaptable to various levels of corrective terms within the blendshape function, ranging from linear (f_L) to higher-order non-linear functions (such as f_Q).

The generalizability and flexibility of our approach enables it to address a wider array of animation challenges, including those with complex facial dynamics. This approach, through its comprehensive optimization across the entire animation sequence and its adaptability to various blendshape complexities, presents a significant step forward in achieving more precise facial reconstructions and ensuring smoother motion transitions in blendshape animation.

2 Related Work

2.1 Anatomically-Based versus Blendshape Models

Anatomically-based face models [Ichim et al. 2017; Sifakis et al. 2005] offer high-fidelity deformation and realistic perception. Despite their detailed representation, these models pose challenges in animation control and interpretability. In contrast, blendshape models, a standard in practical face animation due to their simplicity and ease of manipulation, have been extensively explored [Choe and Ko 2006; Choe et al. 2001; Pighin et al. 1998]. While traditionally sculpted manually, there have been developments towards automation [Bouaziz et al. 2013; Deng et al. 2006; Li et al. 2010, 2013; Moser et al. 2021; Ribera et al. 2017]. Our work assumes the existence of such a blendshape basis, focusing instead on the inverse rigging process.

[2]https://www.unrealengine.com/en-US/eula/mhc

2.2 Inverse Rig Problem and Approaches

The inverse rig problem, central to our work, involves generating animations by adjusting blendshape weights over time. Two main approaches exist: model-based and data-based. Model-based methods [Bouaziz et al. 2013; Çetinaslan 2016; Racković et al. 2023a] utilize optimization techniques, leveraging the rig structure, as opposed to data-based methods that rely on regression models trained on extensive animated data [Bailey et al. 2020; Holden et al. 2015; Seol and Lewis 2014; Song et al. 2011]. Our approach falls into the former category, focusing on a blendshape-based model-based solution. Previous works in this area typically address least squares problems with additional regularization for stability [Çetinaslan 2016; Cetinaslan and Orvalho 2020a,b], sparsity [Bouaziz et al. 2013; Neumann et al. 2013; Racković et al. 2023b], or temporal smoothness [Seol et al. 2012; Tena et al. 2011].

2.3 Direct Manipulation and Face Segmentation

Related areas include direct manipulation, demanding real-time solutions for adjusting expressions [Cetinaslan and Orvalho 2020a,b; Lewis and Anjyo 2010; Seo et al. 2011], and face segmentation for animation, where the focus varies from creating large mesh segments for inverse rig [Bailey et al. 2020; Joshi et al. 2006; Kei and Tomoyuki 2012; Marco et al. 2020; Racković et al. 2021; Reverdy et al. 2015; Tena et al. 2011] to adding secondary motion effects with smaller segments [Neumann et al. 2013; Romeo and Schvartzman 2020; Wu et al. 2016; Zoss et al. 2020]. While not our primary focus, the concept of face clustering for distributed rig inversion is also explored in this work.

2.4 Our Contributions

Our contributions to the field of blendshape animation build upon complex corrective blendshape terms that enhance the fidelity of mesh reconstructions, drawing on the methodologies established in [Racković et al. 2023; Racković et al. 2023b] and sparsity regularization to achieve a lower cardinality in the weight vectors, which aids in simplifying post-animation adjustments, as discussed in prior works [Racković et al. 2023a; Seol et al. 2011]. This leads to an integrated problem formulation for high-quality rig inversion, and additionally we recognize the critical role of temporal continuity in animation, and employ a roughness penalty regularization strategy aimed at ensuring smoother transitions between frames. Finally, we present an extensive set of results, confirming superiority of the proposed method wrt state-of-the-art solutions; Opposed to high-fidelity models, our method yields solution with similar accuracy with 85% execution time reduction, and order of difference reduction in temporal roughness; compared to simpler ones, our method's mesh reconstruction is considerably more accurate, with over 60% reduction in mesh error.

3 Refined Inverse Rigging Methodology for Blendshape Animation

In this section, we detail our approach for solving the inverse rig problem, prioritizing high accuracy in mesh reconstruction and ensuring smooth transitions between blendshape weights across frames. Unlike [Racković et al. 2023a], which focuses on single-frame analysis, our method evaluates the entire animation sequence, necessitating a matrix-based representation for weights and other elements in the objective function.

3.1 Data Fidelity and Regularization Framework

Consider an animation comprising T frames, denoted as $t = 1, \ldots, T$. We represent the weight vectors for each frame as $\mathbf{w}^{(t)}$ and assemble these into a weight matrix $\mathbf{W} = [\mathbf{w}^{(1)}, \ldots, \mathbf{w}^{(T)}] \in \mathbb{R}^{m \times T}$. The rig function for the entire sequence is then expressed as:

$$\begin{aligned} f(\mathbf{W}) =& \mathbf{B}\mathbf{W} + \sum_{t=1}^{T} \sum_{\{i,j\} \in \mathcal{P}} w_i^{(t)} w_j^{(t)} \operatorname{diag}(\mathbf{b}^{\{ij\}}) \mathbf{E}^{(t)} \\ &+ \sum_{\{i,j,k\} \in \mathcal{T}} w_i^{(t)} w_j^{(t)} w_k^{(t)} \operatorname{diag}(\mathbf{b}^{\{ijk\}}) \mathbf{E}^{(t)} \\ &+ \sum_{\{i,j,k,l\} \in \mathcal{Q}} w_i^{(t)} w_j^{(t)} w_k^{(t)} w_l^{(t)} \operatorname{diag}(\mathbf{b}^{\{ijkl\}}) \mathbf{E}^{(t)}, \end{aligned} \tag{6}$$

where $\mathbf{E}^{(t)}$ is a matrix of zeros with the t^{th} column consisting of ones. Similarly, we define the target meshes matrix $\hat{\mathbf{B}} = [\hat{\mathbf{b}}^{(1)}, \ldots, \hat{\mathbf{b}}^{(T)}] \in \mathbb{R}^{n \times T}$, with $\hat{\mathbf{b}}^{(t)}$ being the target mesh for frame t.

We propose the objective that consists of three terms:

$$\underset{0 \le \mathbf{W} \le 1}{\text{minimize}}\, E_{\text{df}} + \alpha E_{\text{sr}} + \beta E_{\text{tsr}}, \tag{7}$$

where E_{df} stands for a data fidelity term, i.e., a difference between the estimated mesh and the target mesh in vertex space, E_{sr} stands for the sparsity regularization forcing the cardinality of the weights to be low, and E_{tsr} is temporal smoothness regularizer, forcing the weights in the consecutive frames to have similar values; $\alpha, \beta \ge 0$ are corresponding regularization weights, dictating the importance of each term. Let us observe each of these terms individually.

Data Fidelity. The data fidelity term, along with the sparsity regularization term, aligns with the formulation in (3) but is now adapted to a matrix context to handle the entire animation sequence. Specifically, the data fidelity term is defined as:

$$E_{\text{df}} = \frac{1}{n} \| f(\mathbf{W}) - \hat{\mathbf{B}} \|_F^2, \tag{8}$$

where $\| \cdot \|_F$ denotes the Frobenius norm, measuring the discrepancy between the estimated mesh sequence $f(\mathbf{W})$ and the target mesh sequence $\hat{\mathbf{B}}$.

We employ a coordinate descent approach to minimize this term, focusing on a single blendshape controller, denoted as e, across the temporal dimension. In this context, we reformulate Eq. (8) into a quadratic expression:

$$E_{\text{df}} = \frac{1}{n} \mathbf{W}_e^T \Phi \mathbf{W}_e + \frac{2}{n} \mathbf{W}_e^T \Theta, \tag{9}$$

where $\mathbf{W}_e$ represents the weights corresponding to controller e over all frames. The matrices Φ and Θ are constructed as follows:

Matrix $\Phi = \operatorname{diag}([\phi^{(1)T}\phi^{(1)}, \ldots, \phi^{(T)T}\phi^{(T)}])$ encapsulates all the terms interacting with the quadratic terms of the objective. Each $\phi^{(t)}$ represents the contribution of the quadratic term of the

blendshape controller e at frame t, computed as:

$$\phi_i^{(t)} = B_{ie} + \sum_{j\in\mathcal{P}(e)} w_j^{(t)}\mathbf{b}_i^{\{je\}} + \sum_{\{j,k\}\in\mathcal{T}(e)} w_j^{(t)}w_k^{(t)}\mathbf{b}_i^{\{jke\}} + \sum_{\{j,k,l\}\in\mathcal{Q}(e)} w_j^{(t)}w_k^{(t)}w_l^{(t)}\mathbf{b}_i^{\{jkle\}}, \tag{10}$$

which considers not only the direct influence of controller e but also its interaction with other controllers in the corrective terms.

Matrix $\Theta = [\phi^{(1)T}\psi^{(1)}, \ldots, \phi^{(T)T}\psi^{(T)}]^T$ represents the linear interaction terms, where each $\psi^{(t)}$ is given by:

$$\psi_i^{(t)} = \sum_{j\neq e} w_j^{(t)}B_{ij} + \sum_{\{j,k\}\in\mathcal{P}} w_j^{(t)}w_k^{(t)}\mathbf{b}_i^{\{jk\}} + \sum_{\{j,k,l\}\in\mathcal{T}} w_j^{(t)}w_k^{(t)}w_l^{(t)}\mathbf{b}_i^{\{jkl\}} + \sum_{\{j,k,l,h\}\in\mathcal{Q}} w_j^{(t)}w_k^{(t)}w_l^{(t)}w_h^{(t)}\mathbf{b}_i^{\{jklh\}} - \hat{\mathbf{b}}_i^{(t)}. \tag{11}$$

This accounts for the contributions of all other blendshape controllers, as well as the deviation from the target mesh $\hat{\mathbf{b}}^{(t)}$ at frame t. The data fidelity term effectively quantifies and minimizes the difference between the animated mesh sequence and the target sequence, ensuring high accuracy in replicating facial expressions over time.

Sparsity Regularization. The sparsity regularization term is critical for ensuring that the animation remains computationally efficient and interpretable. It is defined as the normalized sum of all blendshape weights across the entire animation sequence. Mathematically, this is represented as:

$$E_{\text{sr}} = \frac{1}{m}\sum_{i=1}^{m}\sum_{t=1}^{T} w_i^{(t)} = \frac{1}{m}\mathbf{1}^T\mathbf{W}\mathbf{1}, \tag{12}$$

where $\mathbf{1}$ denotes a vector of ones. This encourages the model to use as few active blendshapes as possible at each frame, leading to a sparser and more interpretable set of weights. Due to the non-negativity constraints on the weights, E_{sr} is guaranteed to be non-negative. The integration of sparsity regularization enhances computational efficiency by reducing the number of active blendshapes, but also simplifies the task of manual adjustments or further processing by animators. By penalizing the sum of the weights, the model naturally gravitates towards solutions where fewer blendshapes are used to achieve the desired facial expressions, promoting a more streamlined and manageable animation process.

Temporal Smoothness Regularization. A key aspect of realistic animation is the smoothness of transitions between frames. To achieve this, we incorporate a roughness penalty function into our regularization framework. This function penalizes the squared differences between adjacent weight values across frames, effectively encouraging temporal continuity in the animation. The temporal smoothness regularization term is formulated as follows:

$$E_{\text{tsr}} = \sum_{t=1}^{T-2}\sum_{i=1}^{m} |w_i^{(t)} - 2w_i^{(t+1)} + w_i^{(t+2)}|^2 = \sum_i \mathbf{W}_i^T\mathbf{F}\mathbf{W}_i, \tag{13}$$

where $\mathbf{W}_i$ denotes the weight vector for the i^{th} blendshape across all frames. The matrix $\mathbf{F}$ is a pentadiagonal matrix defined as:

$$\mathbf{F} = \begin{bmatrix} 1 & -2 & 1 & 0 & \cdots & 0 \\ -2 & 5 & -4 & 1 & \cdots & 0 \\ 1 & -4 & 6 & -4 & \ddots & \vdots \\ 0 & 1 & -4 & 6 & \ddots & 1 \\ \vdots & \ddots & \ddots & \ddots & \ddots & -2 \\ 0 & \cdots & 0 & 1 & -2 & 1 \end{bmatrix}, \tag{14}$$

with the pattern designed to penalize the roughness or abrupt changes in the weight vectors between consecutive frames.

This regularization term thus plays a crucial role in ensuring the naturalness and fluidity of the generated animation. By minimizing E_{tsr}, our model actively works to smooth out the transitions, leading to more lifelike and appealing animations that closely mimic natural human expressions over time.

Final formulation. To synthesize the various aspects of our method into a coherent optimization framework, we formulate a comprehensive objective function that balances the need for data fidelity, sparsity, and temporal smoothness. This objective, to be minimized with respect to each blendshape controller e, encapsulates the essence of our approach:

$$\underset{\mathbf{0}\leq\mathbf{W}_e\leq\mathbf{1}}{\text{minimize}}\ \mathbf{W}_e^T\left(\frac{1}{n}\Phi + \beta\mathbf{F}\right)\mathbf{W}_e + 2\mathbf{W}_e^T\left(\frac{1}{n}\Theta + \frac{\alpha}{2m}\mathbf{1}\right). \tag{15}$$

In this equation, Φ and Θ are matrices derived from the data fidelity term, encoding the relationship between the blendshape weights and the target meshes. The matrix $\mathbf{F}$, arising from the temporal smoothness regularization term, ensures that changes in the blendshape weights are gradual over the sequence of frames. Lastly, the term involving $\mathbf{1}$, originating from the sparsity regularization, promotes solutions with fewer active blendshapes, thereby aiding in interpretability and computational efficiency. This carefully constructed objective function is central to our method, guiding the optimization process towards solutions that are not only accurate in reproducing the target facial expressions but also efficient and smooth over time. By balancing these critical aspects, our approach advances the state-of-the-art in blendshape animation, particularly in scenarios requiring high fidelity and natural motion dynamics.

3.2 Optimization Strategy

The optimization strategy used in solving the proposed objective is *coordinate descent*, a well established optimization technique that is guaranteed to produce monotonically non-increasing costs [Luo and Tseng 1992; Wright 2015]. With coordinate descent, a single component of the problem is visited at a time, and the objective is minimized in it, while keeping the other values fixed, as we do in proposed objective (15), observing only a single controller e as a variable. This decoupling in components allows us to simplify otherwise non-convex problem (5), into a constrained quadratic program, for which fast solutions and implementations are readily available [Moré and Toraldo 1989; Virtanen et al. 2020]. Coordinate descent is particularly suitable for the application in inverse rigging, since estimating one blendshape weight at a time will help avoid

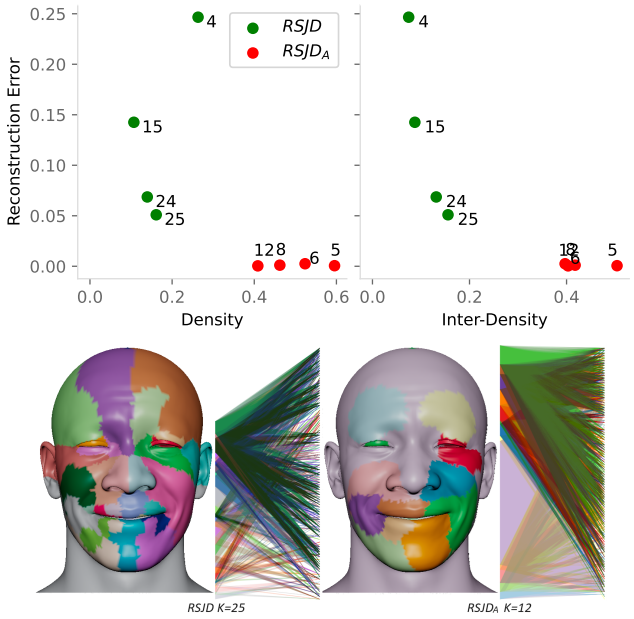

Fig. 2. **Top**: The trade-off between *Reconstruction Error* (E_R) and *Density* (left), and *Inter-Density* (right), across different clustering approaches, with annotations indicating the chosen number of clusters (K). **Bottom**: Clusters obtained using $RSJD$ with $K = 25$ (left) and $RSJD_A$ with $K = 12$ (right). A bipartite graph represents mesh-controller connections, using the same color coding, where the left partition denotes mesh vertices, and the right partition signifies the blendshape indices assigned to each cluster.

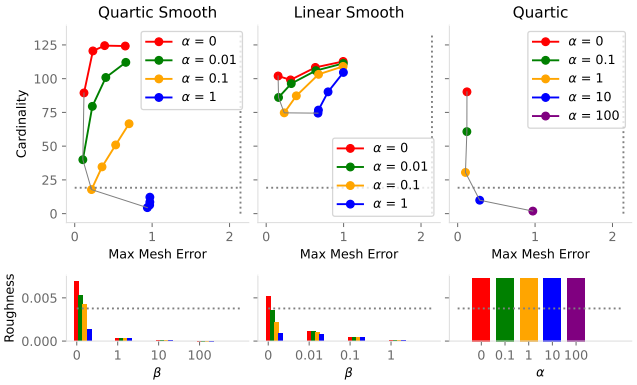

Fig. 3. Parametric Analysis of Animation Metrics During Training. **Top:** The interplay between blendshape cardinality and mesh error under various animation approaches. The color coding denotes different levels of the sparsity regularizer, α, while individual points represent a spectrum of the smoothness parameter, β, converging at $\beta = 0$ indicated by the solid gray line. The horizontal dotted line marks the cardinality of the actual animation data used as the ground truth, whereas the vertical dotted line indicates the mesh error that would result if no blendshapes were activated (all weights at 0). **Bottom:** The roughness penalty corresponding to the varying values of β along the x-axis. The color scheme is consistent with the top graph, linked to the α values. The horizontal dotted line represents the benchmark roughness penalty derived from the ground-truth animation data.

the simultaneous activation of blendshapes with canceling effects, i.e., moving the corner of the mouth up and down at the same time [Seol et al. 2011].

An important aspect of coordinate descent is the order of component updates. Even though the convergence guaranties hold with an arbitrary update order [Wright 2015], a poor choice can lead to slower convergence and a bad local minima. This problem was also studied in blendshape animation literature [Hyde et al. 2021; Racković et al. 2023a; Seol et al. 2011]. We follow the stance of [Seol et al. 2011], ordering the blendshapes by their magnitude of deformation in the mesh, in line with artist intuition of initially setting the more drastic weights before focusing on fine details.

4 Quantitative Analysis

We present the comprehensive evaluation of the proposed methodology, first describing the data and benchmarks used in our study, followed by a detailed analysis of the results. The performance of our method is compared against established benchmarks to demonstrate its efficacy in achieving high-fidelity, smooth animations. The impact of different parameters on the results is examined to provide insights into the behavior of our approach under various conditions.

4.1 Dataset Characteristics and Comparative Benchmarks

We selected the Metahuman character *Jesse* for evaluation, motivated by the high-quality and realistic nature of Metahuman blendshape models, coupled with their accessibility as publicly available resources. The *Jesse* model is equipped with $m = 133$ base blendshapes and over 400 corrective terms, encompassing three levels of correction. These elements make it an ideal candidate for demonstrating the nuanced capabilities of our method in handling complex facial animations. Originally, the meshes contain over $24,000$ vertices. To focus on the facial region, which is our primary interest, we have excluded vertices on the neck and an inactive area at the back of the skull. This results in a subsampled mesh comprising $n = 10,000$ vertices. The model is manually animated to generate a reference motion, which comprises 100 frames for the training and 100 frames for the test set. This division allows for a robust assessment of our method's performance both in learning and generalization.

In the results section, we refer to our proposed method as *Quartic Smooth*, emphasizing its unique aspects: the inclusion of corrective terms using a quartic blendshape function for enhanced realism, and the integration of smoothness regularization, setting it apart from previous model-based solutions. The first benchmark is the method described in [Lewis and Anjyo 2010], named *Linear Smooth*, which incorporates smoothness regularization but no corrective terms. It optimizes the objective defined in (4). The second benchmark is the *Quartic* algorithm from [Racković et al. 2023], which considers corrective terms, but treats each frame independently, overlooking temporal continuity. *Quartic* solves the objective in (3). Notably, while both benchmarks include weight regularization, they differ in

approach: *Quartic* uses an l_1 norm, promoting sparsity, whereas *Linear Smooth* employs a squared l_2 norm, penalizing large activations without directly reducing the number of active components.

4.2 Clustering Approach for Computational Efficiency

Given its localized nature, a significant number of vertices in the human face have limited influence on the majority of blendshape weights. Our approach incorporates a face clustering strategy to exploit this characteristic for enhanced computational efficiency. This strategy allows for a parallelized and potentially more regularized solution to the inverse rigging problem [Racković et al. 2023; Racković et al. 2021; Seol et al. 2011].

We adopt the clustering methodologies proposed in [Racković et al. 2021] (denoted as *RSJD*) and in [Racković et al. 2023] (denoted as $RSJD_A$). These methods are suitable as they generate clusters in a model-based manner. However, our approach remains flexible to be applicable with other clustering techniques, provided they meaningfully separate mesh segments and blendshape controllers.

The core idea is to treat each cluster as an independent model, decomposing the overall inverse rigging problem (15) into a series of smaller subproblems. Each subproblem focuses on a specific subset of vertices and blendshapes relevant to its cluster. This subdivision not only facilitates parallel processing but also potentially introduces additional regularization opportunities, leading to a more efficient solution. While the applied methods do segment face mesh, these segments are used for solving the rig, not for reconstructing the mesh. In particular, after the inverse rig problem is solved for each cluster independently, estimated local weights are merged into a single, global, vector of weights, and a face mesh is accordingly deformed as a whole; hence there are no discontinuities in the reconstructed meshes.

Following the recommendations in [Racković et al. 2023], we generate several instantiations of each clustering approach. The optimal configuration is selected based on a balance between *Reconstruction Error* and *Density* in the resulting clustering graphs, as illustrated in Fig. 2 (top). *Density* represents the total number of mesh vertex-blendshape pairs in all clusters, i.e., the number of edges in a bipartite graph of the clustering, while *Inter-Density* is number of edges connecting different partitions. Optimal cluster numbers are determined to minimize the model size while retaining essential information. The corresponding mesh clusters and their bipartite graph representations are detailed in Fig. 2, showcasing the effective distribution of vertices and blendshape indices across the identified clusters.

As additional comparative methods in the quantitative results analysis, we include the combination of our proposed method with the two described clustering techniques. We examine two instances of clustering: *Clustered RSJD 25* and *Clustered* $RSJD_A$ *12*, named after the clustering methods and the chosen number of clusters (K). These cases help to assess the impact of facial clustering on the performance and efficiency of our approach.

4.3 Analytical Performance Evaluation

All the considered methods have a weight regularization hyperparameter α that should be selected before comparing the results in more details. Additionally, our method, *Quartic Smooth*, as well as the benchmark *Linear Smooth*, have the smoothness regularizer β. To select the appropriate values, each method is cross-validated, as presented in Fig. 3 and 4. These results draw an expected trade-off curve between cardinality and mesh error, with the exception of the *Linear Smooth* — in this case a favorable decrease of the cardinality is never achieved, since l_2 norm in the objective tends to keep small positive values rather than setting them to zero. Optimal values for the parameters are selected so that all three presented metrics are minimized, and the selection that is used for further tests is listed in Tab. 1. By examining Tab. 2, we can compare the performance of the various methods across several key metrics. *Quartic* and *Quartic Smooth* exhibits the lowest Mesh Errors, where the former is slightly ahead in the mean value compared to the later (.013 vs .016), but it is the other way around for the max values (.073 vs .070). This showcases the superior ability of higher-level models to closely match the target mesh in the worst-case and on average. This suggests high accuracy in mesh reconstruction.

The Cardinality is lowest for the clustered approach *RSJD* (17.3), suggesting it is the most efficient in terms of blendshape usage. *Linear Smooth* on the other side has significantly more active component than any other method.

Despite its accuracy, *Quartic* has the highest Roughness Penalty ($1.1e^{-2}$), implying less smooth transitions, which is also visible in the supplementary video. All the other methods include regularization, which leads to having significantly smoother animation curves.

Linear Smooth is the fastest (.02 seconds), which is expected given its less complex nature. *Quartic* is the slowest (14.4 seconds), with large margin. *Quartic Smooth* strikes a balance between complexity and speed (2.16 seconds), offering a more efficient solution than *Quartic* while maintaining high accuracy and smoothness. This advantage over *Quartic* is due to the fact that the proposed method solves for an entire animation sequence at once, unlike *Quartic* that optimizes frame by frame. Additionally, parallelizable structure given by the clustering allows for even better execution times.

Table 1. Final parameter values for each method.

	Quartic Smooth	Linear Smooth	Quartic	Clustered $RSJD$ 25	Clustered $RSJD_A$ 12
α	0.01	0.1	1	10	1
β	0.1	0.005	/	1	1

Table 2. Average metrics values for test results (see Fig. 5).

	Max Error	Mean Error	Card.	l_1 Norm	Rough. Penalty	Exec. Time
Quartic Smooth	.070	.016	66.5	4.66	$2.5e^{-3}$	2.16
Linear Smooth	.185	.045	84.8	4.35	$3.2e^{-3}$	0.02
Quartic	.073	.013	30.1	3.87	$1.1e^{-2}$	14.4
RSJD	.308	.074	17.3	1.60	$5.5e^{-3}$	0.15
$RSJD_A$	.511	.139	45.2	1.77	$1.3e^{-3}$	0.44

Quartic Smooth demonstrates a well-balanced performance across accuracy, efficiency, and smoothness, making it a robust choice for high-quality and realistic blendshape animation. *Quartic* excels in mesh error but at the cost of high roughness and execution time, while *Linear Smooth* offers the fastest execution with compromises in accuracy. The clustered approaches offer varying trade-offs, with $RSJD_A$ showing lower execution times and roughness penalties, but at the cost of higher mesh errors.

Fig. 3 shows the trade-off between cardinality and maximum mesh error on the top row and the roughness penalty across different smoothness parameter β values on the bottom row. For *Quartic Smooth*, the top graph suggests that as the sparsity regularizer α increases, the cardinality decreases while the maximum mesh error increases. This indicates that higher sparsity comes at the cost of accuracy. In the bottom graph, the roughness penalty shows negative correlation wrt the smoothness hyperparameter β, as expected, indicating that higher weight regularization would produce smoother animation curves. For *Linear Smooth*, the cardinality does not significantly change with different values of α, indicating a potential plateau in the trade-off, where increasing mesh error does not impact the sparsity substantially. The variations in the roughness penalty across different values of β, show similar trends as in the case of *Quartic Smooth*, just with less of a contrast. For *Quartic*, increasing α leads to an increase in the mesh error but at the expense of a rapid decrease in cardinality, suggesting a strong influence of the sparsity regularizer on model complexity. However, for any choice of α, the roughness of the solution is virtually unchanged, and much higher than in the previous two cases.

The *Quartic Smooth* approach seems to strike a balance between maintaining mesh accuracy and controlling the model complexity (cardinality), compared to the *Linear Smooth* and *Quartic* approaches. The roughness penalties across *Quartic Smooth* and *Linear* are relatively low, indicating smooth transitions, with *Quartic Smooth* showing a slight advantage. However, one must consider the trade-offs between accuracy, sparsity, and temporal smoothness when selecting the appropriate parameters for each method. Fig. 1 further illustrates this comparison, showing animation curves over time, and the reconstructed meshes for the two consecutive frames. The reconstruction error for *Linear* is relatively high, while it is low for *Quartic*, which, on the other side, has bigger differences in the consecutive reconstructed meshes. Our approach strikes the right balance between the two.

Fig. 4 shows a comparative analysis of three different blendshape animation approaches: *Quartic Smooth, Clustered RSJD 25*, and *Clustered* $RSJD_A$ *12*. The analysis is based on the evolution of specific metrics over the training set while varying the sparsity regularizer (α) and the smoothness parameter (β). The *Quartic Smooth* approach in our experiments offers a favorable balance between animation accuracy and sparsity, with smooth temporal transitions. The clustered approaches may provide computational benefits, but potentially at the cost of increased mesh error, especially for higher sparsity levels. The choice between these methods may depend on the specific requirements of an animation project, such as the necessity for real-time performance (favoring clustered methods) versus the need for high-fidelity animations.

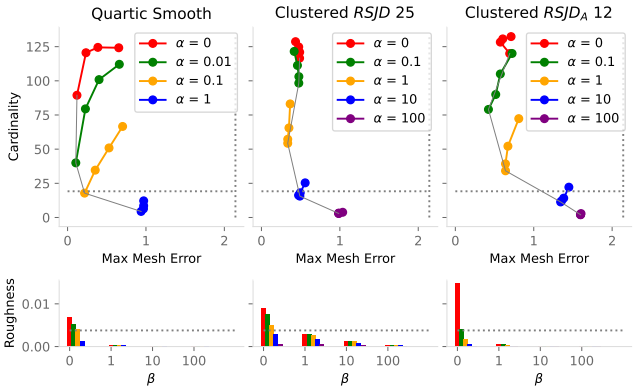

Fig. 4. Comparative Analysis of Training Metrics Across Different Parameterizations and Methodologies. In this Figure we show how applying clustering on top of our approach affects the overall results.

Additional examples comparing mesh reconstructions for all the methods across different frames are given in Fig. 6.

5 Conclusion

We have introduced *Quartic Smooth*, an advanced model-based approach for blendshape animation that adeptly balances accuracy, sparsity, and temporal smoothness. Our method demonstrates a marked improvement in mesh fidelity over traditional linear models and offers a flexible framework for high-quality applications. Our findings illustrate *Quartic Smooth*'s superior performance in creating realistic facial animations with reduced computational overhead, particularly when compared to existing linear and non-linear approaches. One downside of the proposed method is on the execution time, as it is not suitable for real-time application purposes. The introduction of face clustering techniques further augments computational efficiency, opening possibilities for real-time animation processing, yet it sacrifices some of the accuracy, as confirmed in the results section. Future work will explore the optimization of these techniques and the integration of machine learning to refine parameter selection. With its contribution to the realistic and efficient inverse rigging of facial animations, *Quartic Smooth* is poised to influence future developments in character animation within the computer graphics community.

Acknowledgments

This work was partially supported by NOVA LINCS (UIDB/04516/2020) with the financial support of FCT I.P. and Project "Artificial Intelligence Fights Space Debris" No C626449889-0046305 co-funded by Recovery and Resilience Plan and NextGeneration EU Funds, www.recuperarportugal.gov.pt. and by the Ministry of Education of the Republic of Serbia (451-03-9/2021-14/200125).

References

Stephen W. Bailey, Dalton Omens, Paul Dilorenzo, and James F. O'Brien. 2020. Fast and Deep Facial Deformations. *ACM Trans. Graph.* 39, 4 (2020).
Sofien Bouaziz, Yangang Wang, and Mark Pauly. 2013. Online Modeling for Realtime Facial Animation. *ACM Trans. Graph.* 32, 4 (2013).

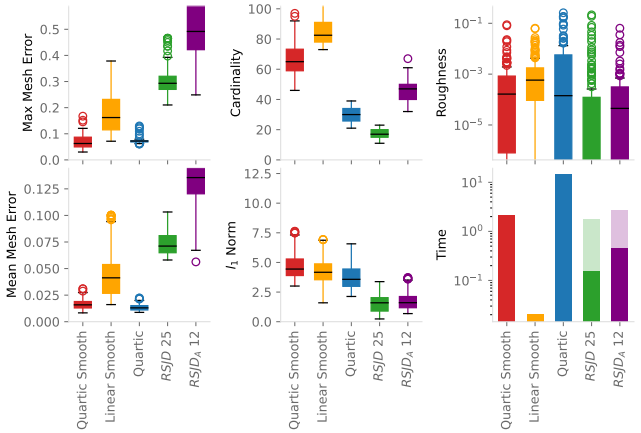

Fig. 5. Results over the test set with the selected hyperparameter values corresponding to Tab. 1. The execution time for the clustered approach is presented in solid and shaded — solid color indicates the execution time of the slowest cluster, while shaded bar shows the time of solving the clusters sequentially.

Cumhur Ozan Çetinaslan. 2016. *Position Manipulation Techniques for Facial Animation.* Ph. D. Dissertation. Faculdade de Ciencias da Universidade do Porto.

Ozan Cetinaslan and Veronica Orvalho. 2020a. Sketching Manipulators for Localized Blendshape Editing. *Graphical Models* 108 (2020), 101059.

Ozan Cetinaslan and Veronica Orvalho. 2020b. Stabilized blendshape editing using localized Jacobian transpose descent. *Graphical Models* 112 (2020), 101091.

Byoungwon Choe and Hyeong-Seok Ko. 2006. Analysis and synthesis of facial expressions with hand-generated muscle actuation basis. In *ACM SIGGRAPH 2006.*

Byoungwon Choe, Hanook Lee, and Hyeong-Seok Ko. 2001. Performance-driven muscle-based facial animation. *The Journal of Visualization and Computer Animation* 12, 2 (2001), 67–79.

Zhigang Deng, Pei-Ying Chiang, Pamela Fox, and Ulrich Neumann. 2006. Animating Blendshape Faces by Cross-Mapping Motion Capture Data. In *Proceedings of the 2006 Symposium on Interactive 3D Graphics and Games.* Association for Computing Machinery, 43–48.

Daniel Holden, Jun Saito, and Taku Komura. 2015. Learning an inverse rig mapping for character animation. In *Proceedings of the 14th ACM SIGGRAPH/Eurographics Symposium on Computer Animation.* 165–173.

David AB Hyde, Michael Bao, and Ronald Fedkiw. 2021. On obtaining sparse semantic solutions for inverse problems, control, and neural network training. *J. Comput. Phys.* 443 (2021), 110498.

Alexandru-Eugen Ichim, Petr Kadleček, Ladislav Kavan, and Mark Pauly. 2017. Phace: Physics-based face modeling and animation. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 1–14.

Pushkar Joshi, Wen C Tien, Mathieu Desbrun, and Frédéric Pighin. 2006. Learning controls for blend shape based realistic facial animation. In *ACM SIGGRAPH 2006.*

Hirose Kei and Higuchi Tomoyuki. 2012. Creating facial animation of characters via MoCap data. *Journal of Applied Statistics* 39, 12 (2012), 2583–2597.

J.P. Lewis and Ken-ichi Anjyo. 2010. Direct Manipulation Blendshapes. *IEEE Computer Graphics and Applications* 30, 04 (2010), 42–50.

John P. Lewis, Ken Anjyo, Taehyun Rhee, Mengjie Zhang, Frederic H. Pighin, and Zhigang Deng. 2014. Practice and Theory of Blendshape Facial Models. *Eurographics (State of the Art Reports)* 1, 8 (2014), 2.

Hao Li, Thibaut Weise, and Mark Pauly. 2010. Example-based facial rigging. *ACM Trans. Graph.* 29, 4 (2010), 1–6.

Hao Li, Jihun Yu, Yuting Ye, and Chris Bregler. 2013. Realtime facial animation with on-the-fly correctives. *ACM Trans. Graph.* 32, 4 (2013), 42–1.

Zhi-Quan Luo and Paul Tseng. 1992. On the convergence of the coordinate descent method for convex differentiable minimization. *Journal of Optimization Theory and Applications* 72, 1 (1992), 7–35.

Fratarcangeli Marco, Bradley Derek, A. Gruber, Zoss Gaspard, and Beeler Thabo. 2020. Fast Nonlinear Least Squares Optimization of Large-Scale Semi-Sparse Problems. *Computer Graphics Forum* 39 (2020).

Jorge J Moré and Gerardo Toraldo. 1989. Algorithms for bound constrained quadratic programming problems. *Numer. Math.* 55, 4 (1989), 377–400.

Lucio Moser, Chinyu Chien, Mark Williams, Jose Serra, Darren Hendler, and Doug Roble. 2021. Semi-supervised video-driven facial animation transfer for production. *ACM Transactions on Graphics (TOG)* 40, 6 (2021), 1–18.

Thomas Neumann, Kiran Varanasi, Stephan Wenger, Markus Wacker, Marcus Magnor, and Christian Theobalt. 2013. Sparse localized deformation components. *ACM Trans. Graph.* 32, 6 (2013), 1–10.

Frédéric H. Pighin, Jamie Hecker, Dani Lischinski, Richard Szeliski, and D. Salesin. 1998. Synthesizing realistic facial expressions from photographs. *Proceedings of the 25th annual conference on Computer graphics and interactive techniques* (1998).

Stevo Racković, Cláudia Soares, and Dušan Jakovetić. 2023. Distributed Solution of the Blendshape Rig Inversion Problem. In *SIGGRAPH Asia 2023 Technical Communications* (Sydney, NSW, Australia) *(SA '23).* Association for Computing Machinery, New York, NY, USA, Article 24, 4 pages.

Stevo Racković, Cláudia Soares, Dušan Jakovetić, and Zoranka Desnica. 2023a. High-fidelity Interpretable Inverse Rig: An Accurate and Sparse Solution Optimizing the Quartic Blendshape Model. *arXiv preprint arXiv:2302.04820* (2023).

Stevo Racković, Cláudia Soares, Dušan Jakovetić, and Zoranka Desnica. 2023b. A majorization–minimization-based method for nonconvex inverse rig problems in facial animation: algorithm derivation. *Optimization Letters* (2023), 1–15.

Stevo Racković, Cláudia Soares, Dušan Jakovetić, Zoranka Desnica, and Relja Ljubobratović. 2021. Clustering of the blendshape facial model. In *2021 29th European Signal Processing Conference (EUSIPCO).* IEEE, 1556–1560.

Clément Reverdy, Sylvie Gibet, and Caroline Larboulette. 2015. Optimal Marker Set for Motion Capture of Dynamical Facial Expressions. In *Proceedings of the 8th ACM SIGGRAPH Conference on Motion in Games.* Association for Computing Machinery, 31–36.

Roger Blanco i Ribera, Eduard Zell, J. P. Lewis, Junyong Noh, and Mario Botsch. 2017. Facial Retargeting with Automatic Range of Motion Alignment. *ACM Trans. Graph.* 36, 4 (2017).

Marco Romeo and S. Schvartzman. 2020. Data-Driven Facial Simulation. In *Computer Graphics Forum*, Vol. 39. Wiley Online Library, 513–526.

Jaewoo Seo, Geoffrey Irving, J. P. Lewis, and Junyong Noh. 2011. Compression and Direct Manipulation of Complex Blendshape Models. *ACM Trans. Graph.* 30, 6 (dec 2011), 1–10.

Yeongho Seol, J.P. Lewis, Jaewoo Seo, Byungkuk Choi, Ken Anjyo, and Junyong Noh. 2012. Spacetime Expression Cloning for Blendshapes. *ACM Trans. Graph.* 31, 2 (2012).

Yeongho Seol and J. P. Lewis. 2014. Tuning Facial Animation in a Mocap Pipeline. In *ACM SIGGRAPH 2014 Talks (SIGGRAPH '14).* Association for Computing Machinery.

Yeongho Seol, Jaewoo Seo, Paul Hyunjin Kim, J. P. Lewis, and Junyong Noh. 2011. Artist Friendly Facial Animation Retargeting. In *SIGGRAPH Asia 2011.*

Eftychios Sifakis, Igor Neverov, and Ronald Fedkiw. 2005. Automatic Determination of Facial Muscle Activations from Sparse Motion Capture Marker Data. In *ACM SIGGRAPH 2005 Papers.* Association for Computing Machinery, 417–425.

Jaewon Song, Byungkuk Choi, Yeongho Seol, and Jun yong Noh. 2011. Characteristic facial retargeting. *Computer Animation and Virtual Worlds* 22 (2011).

Andrea Tagliasacchi, Matthias Schröder, Anastasia Tkach, Sofien Bouaziz, Mario Botsch, and Mark Pauly. 2015. Robust articulated-icp for real-time hand tracking. In *Computer graphics forum*, Vol. 34. Wiley Online Library, 101–114.

J. Rafael Tena, Fernando De la Torre, and Iain Matthews. 2011. Interactive Region-Based Linear 3D Face Models. In *ACM SIGGRAPH 2011 Papers.*

Pauli Virtanen, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. 2020. SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature methods* 17, 3 (2020), 261–272.

Stephen J Wright. 2015. Coordinate descent algorithms. *Mathematical programming* 151, 1 (2015), 3–34.

Chenglei Wu, Derek Bradley, Markus Gross, and Thabo Beeler. 2016. An anatomically-constrained local deformation model for monocular face capture. *ACM Trans. Graph.* 35, 4 (2016), 1–12.

Gaspard Zoss, Eftychios Sifakis, Markus Gross, Thabo Beeler, and Derek Bradley. 2020. Data-driven extraction and composition of secondary dynamics in facial performance capture. *ACM Transactions on Graphics (TOG)* 39, 4 (2020), 107–1.

Fig. 6. Resulting meshes for selected frames (rows), and for each method (columns). Stronger red tones indicate higher mesh error.