DOI: 10.1111/cgf.70417
EUROGRAPHICS 2026 / B. Masia and J. Thies
(Guest Editors)

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

Improving Facial Rig Semantics for Tracking and Retargeting

D. Omens[1,2] A. Thurman[1] J. Yu[2] and R. Fedkiw[1,2]

[1]Stanford University, USA
[2]Epic Games, USA

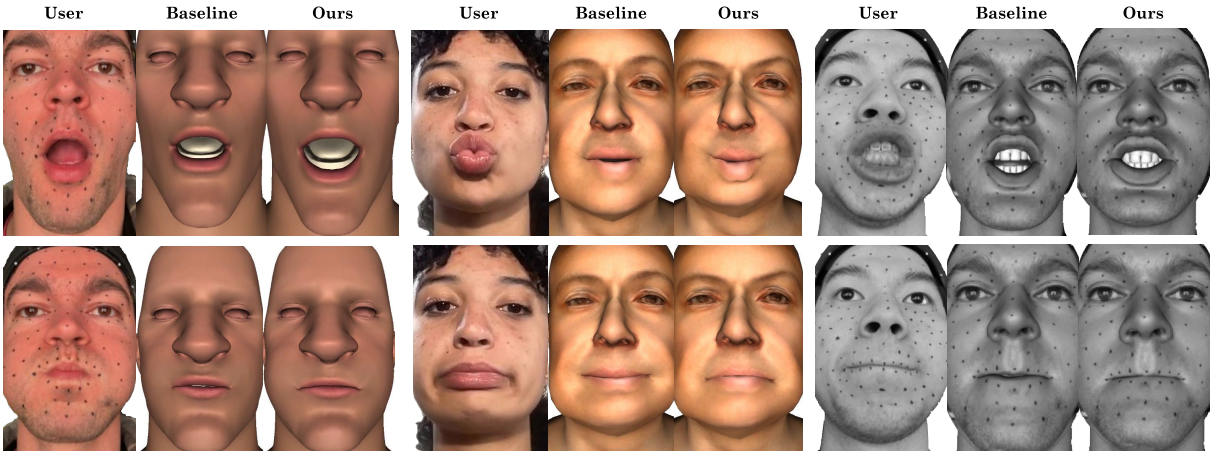

Figure 1: *We introduce a suite of tools for rig calibration to improve the semantic correctness of facial performance retargeting. Left: Retargeting to a game character rig. Middle: Retargeting to a semantic FLAME rig. Right: Retargeting to another identity using MetaHuman Animator as a baseline.*

Abstract

In this paper, we consider retargeting a tracked facial performance to other people or virtual characters. We utilize the same rig framework for both tracking and animation to remove the difficulties associated with retargeting the semantics of one framework to another. Our carefully designed set of Simon-Says expressions and regularizers is used to calibrate each rig to the motion signatures of the relevant performer or target. Although a uniform set of Simon-Says expressions can likely be used for all person-to-person retargeting, we argue that person-to-virtual-character retargeting benefits from an expression set that captures the distinct motion signature of the virtual character rig. The Simon-Says calibrated rigs tend to produce the desired expressions when exercising animation controls. Unfortunately, these well-calibrated rigs still lead to undesirable controls when tracking a performance, even though they generally produce acceptable geometry reconstructions. Thus, we propose a fine-tuning approach that modifies the rig used by the tracker to promote the output of more semantically meaningful animation controls, facilitating high efficacy retargeting. To better address real-world scenarios, the fine-tuning relies on implicit differentiation so that the tracker can be treated as a potentially non-differentiable black box. Experiments demonstrate the benefits of our calibration methods on high-fidelity expressive performance retargeting for different capture conditions, trackers, and rig frameworks.

CCS Concepts

• ***Computing methodologies*** → ***Animation;***

1. Introduction

Performance-driven facial animation, as opposed to manually-keyframed animation, is often used to improve quality, to reduce manual labor, and to facilitate real-time applications that use facial representations. The impact of performance-driven capture is significantly bolstered by the ability to transfer, or *retarget*, cap-

tured performances to other human or character models. Although facial motion retargeting has traditionally been limited to big budget feature films and AAA games due to the need for artist-driven cleanup and intervention, more robust and democratized methods would have an enormous impact on a large variety of real-time applications, especially given the surging interest in the so-called metaverse.

The computer vision community has made leaps and bounds in the reconstruction of faces using a variety of techniques [BLW*24, LSSS18, ZBT22a, RFD*24, TRT*24, DGHG*24, BZH*23], and the higher fidelity approaches tend to utilize PCA-based parameterized models, such as 3DMM [BV99] or FLAME [LBB*17], to regularize the reconstruction. These non-semantic PCA-style models are likely sufficient for reconstruction of real-world performance with different lighting and textures as well as novel views. However, retargeting that performance even if only to a younger/older version of the performer can be problematic without a semantically meaningful rig that can disentangle reconstruction geometry from semantic intention.

Geometry reconstruction is obtained by solving an inverse problem to find some parameters from an image. If this reconstruction is done by an animator, they work with semantic rigs and use their knowledge of the rig's internal parameters to determine controls; in this case, the animator is the "tracker". In parallel parameterization retargeting [ZJ24b], the controls are used to drive a new rig with different internal parameters. An expert animator would choose animation controls in such a way that includes an implicit dependence on the target's internals, but existing automated approaches do not reach this level. Some high-end trackers are designed to mimic this approach, yet expert animators still spend a great deal of time cleaning the output before it can be used for retargeting. For this reason, we aim to improve high-end trackers even further by increasing the semantic accuracy of tracked controls, resulting in retargeted performances more representative of an actor's intent (Figure 1). The maturity of reconstruction techniques means that neutral identities are well-determined, and a high-end tracker likely has highly tuned hyperparameters. Therefore, we focus on the internal rig parameters relating to expressions, stressing that motion signatures can vary greatly from person to person. We propose strategies inspired by animator mimicry for doing this rig calibration.

We summarize our novel contributions as follows:

- We provide a mathematical analysis of rigs and trackers via linearization to rigorously demonstrate the validity of our approach and the separation of geometry reconstruction and rig inversion.
- We design a refined Simon-Says expression set that covers the most important rig parameters while remaining straightforward for an average person to make. In addition, we ensure that the numerical optimization respects the various constraints expert riggers utilize in the rig design process.
- Our implicit function theorem style approach allows for modification of the tracker parameters without requiring differentiability or access to the tracker's internals.
- Our evaluations under different capture conditions, trackers, and rig frameworks employ metrics important to riggers and animators to confirm the efficacy and flexibility of our proposed methods.

2. Related Work

Facial animation rigs [OBP*12] use semantic animation controls to drive a 3D model. Some notable work on facial animation rigs includes [BCGF19, DHT*00, ZSCS04, LWP10, CBE*15, YZF*23, BODO20, YZC*23, KDP*24, MLL*24, CEM*22, KS21, LA10]. Publicly available options for facial animation software include MetaHumans in the Unreal Engine [Epi25] as well as tools suites in Maya [Aut25] and Blender [Ble25].

For a review of prior work on the retargeting of facial performances to other persons or characters, see [ZJ24b]. Although some works have found a degree of success by transferring *per-vertex* geometry displacements from one mesh to another [NN01, SP04, SLS*12, CLC*25], most focus on the underlying rig degrees of freedom. Works that find mappings between *disparate rig frameworks* (cross-mapping) [SCSN11, DCFN06, WHL*04], have found some success, but this is still a difficult problem. Some authors have aimed to address this difficult problem via neural network techniques [KJS*21, ACC*18, ZCZ22], while others have relied on constrained numerical optimization, e.g. [CCGB22] uses anatomical constraints.

Some approaches have aimed to skip the retargeting of geometry and/or rig controls entirely by instead using *image-based retargeting* followed by reconstruction or rig inversion from the retargeted image. [MCW*21] uses a network to translate an image of the performer to a semantically similar image of the target and regress animation controls from the new image. [QSA*23] and [CYSN25]transfer deformation onto arbitrary meshes using an encoder-decoder architecture which includes a FACS-based latent code for editing. [QZL*24] encodes an image of the performer into a latent space and subsequently uses a target-specific decoder to obtain animation controls. With the recent surge in popularity of LLMs, there is also some interest in audio-based retargeting followed by reconstruction or rig inversion from the retargeted audio. In this vein, [PLX*25] recently proposed a method for regressing MetaHuman animator controls for an audio signal.

High-end approaches tend to use the *same rig frameworks* on both the performer and target (parallel parameterization), aiming for a one-to-one remapping of rig control parameters. To obtain high efficacy results, the semantic intention of the performance needs to be disentangled from the geometry. [TZH*24] and [BWL*24] focus on achieving realistic person-to-person retargeting within the constrained input domain of VR headset cameras. [RJS24] proposes a new optimization framework aiming to obtain semantically meaningful animation controls that retarget well. Similar in spirit to our approach, [RZL*17] uses a Range of Motion (ROM) video to modify expression blendshapes aiming to obtain semantically meaningful animation controls. Although it focuses on reconstruction, not retargeting, [BBC*24] suggests fine-tuning the tracker in addition to modifying the tracking rig. [ZJ24a] proposes a new FACS-based rig designed to improve the retargeting efficacy of regressed animation controls. See the supplementary material for additional comments on prior work.

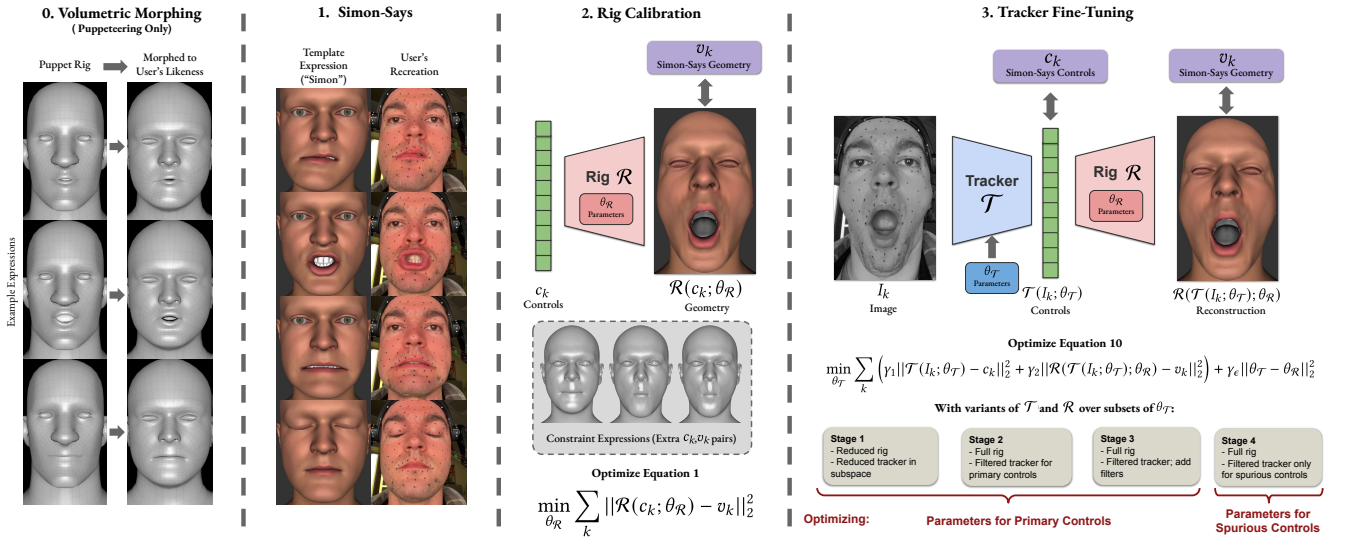

Figure 2: *A high-level overview of our proposed methods for creating and fine-tuning rigs and trackers. For puppeteering, we start with volumetrically morphing the animation rig to the user's likeness (Section 10.3). We then proceed with a Simon-Says capture session, and use that data to calibrate the rig's expressions (Section 8). Finally, to facilitate retargeting, we use our implicit differentiation to optimize the tracker's parameters using the same data that was used for rig calibration according to our optimization strategy (Section 9).*

3. Preliminaries

Animation rigs $\mathcal{R}(c;\theta_\mathcal{R})$ output geometry, driven by animation controls c and internal parameters $\theta_\mathcal{R}$. These internal parameters often consist of neutral geometry and a set of expression parameters. Importantly, the animation controls are designed to have an interpretable physical or semantic meaning so animators can understand how to articulate the rig. For a human actor, given a set of corresponding animation controls versus geometry pairs for k different expressions (c_k, v_k),

$$\min_{\theta_\mathcal{R}} \sum_k ||\mathcal{R}(c_k;\theta_\mathcal{R}) - v_k||_2^2 \tag{1}$$

can be used to determine the internal parameters $\theta_\mathcal{R}$ of the animation rig so the rig's expressions match the actor's expressions. Each of the v_k can either be sculpted by an artist or reconstructed from an image I_k chosen to correspond with c_k.

A tracker $\mathcal{T}(I;\theta_\mathcal{R})$ solves an inverse problem given an image I to find animation controls c that produce geometry when evaluated in the rig $\mathcal{R}(c;\theta_\mathcal{R})$. In this paper, we consider trackers that first perform a geometry reconstruction to obtain geometry v from an input image I and subsequently solve for animation controls c, conditioned upon the rig parameters $\theta_\mathcal{R}$. While there are some trackers that do not follow this paradigm (Section 2), they are less commonly used. Trackers usually also depend on additional parameters besides those of the rig they are conditioned upon, but since we do not change their values they are omitted from our derivations. Given a set of corresponding animation controls versus image pairs (c_k, I_k),

$$\min_{\theta_\mathcal{R}} \sum_k ||\mathcal{T}(I_k;\theta_\mathcal{R}) - c_k||_2^2 \tag{2}$$

can be used to determine the rig parameters used by the tracker such that the tracker produces the correct output for the given poses.

Regularization is typically required in tracker design to improve the ability to generalize to unseen images. This is sometimes accomplished by considering the geometry $\mathcal{R}(\mathcal{T}(I;\theta_\mathcal{R});\theta_\mathcal{R})$ obtained by evaluating the tracker output in the corresponding rig (a "reconstruction" operation). This motivates a modification of Equation 2 to

$$\min_{\theta_\mathcal{R}} \sum_k \gamma_1||\mathcal{T}(I_k;\theta_\mathcal{R}) - c_k||_2^2 + \gamma_2||\mathcal{R}(\mathcal{T}(I_k;\theta_\mathcal{R});\theta_\mathcal{R}) - v_k||_2^2 \tag{3}$$

to match both the animation controls and the geometry. Regarding terminology, a would-be tracker that outputs good geometry but poor animation controls by focusing on the second term in Equation 3 at the expense of the first is typically thought of as a reconstruction method, not an expression tracker.

A high-level overview of our proposed methods is present in Figure 2. To summarize, we first reconstruct facial geometry from a number of Simon-Says expressions, then use this data (in the form of (c_k, v_k) pairs) to calibrate the tracking rig's expressions and finally use the same data to fine-tune the tracker's parameters using our implicit differentiation. Before going into the details of these steps starting with Section 8, we present our thorough analysis of rigs and trackers and our proposed mathematical frameworks (Sections 4 - 7), beginning with a linearization exercise which motivates our approach.

4. A Motivational Linearization

For the sake of simplicity, assume that the rig $\mathcal{R}$ is linear in c implying $v = \mathcal{R}(c;\theta_\mathcal{R}) = A(\theta_\mathcal{R})c$ where $A(\theta_\mathcal{R})$ is a size $m \times n$ matrix. Stacking all the (c_k, v_k) pairs into separate matrices $C \in \mathbb{R}^{n\times k}$ and

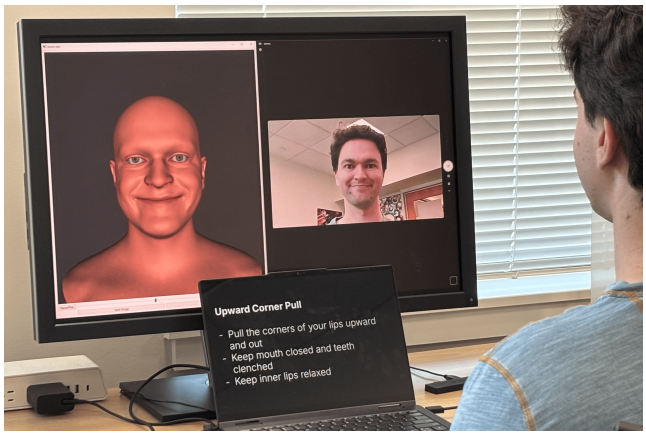

Figure 3: *Our proposed Simon-Says capture setup. Note the three key forms of input to the user: A display of the user to self-correct, a visual representation of the pose to guide non-experts, and a concise verbal description of the expression to avoid exactly replicating the imperfect visual representation.*

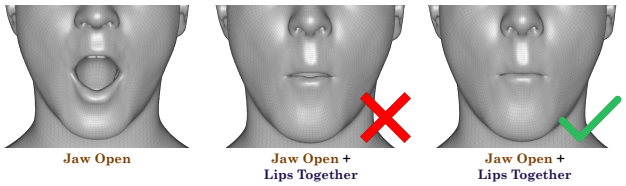

Figure 4: *Unconstrained rig calibration can lead to unacceptable results in the eyes of a technical artist. On the left is a user's calibrated "jaw open" pose. Without our corrective regularization, calibration causes the "jaw open" + "lips together" pose to be technically incorrect due to the lips not sealing (middle). With our proposed correctives, the most important combinations remain technically correct (right).*

$V \in \mathbb{R}^{m\times k}$ yields

$$V = A(\theta_{\mathcal{R}})C \implies C^T A^T(\theta_{\mathcal{R}}) = V^T \tag{4}$$

$$A(\theta_{\mathcal{R}}) = VC^T(CC^T)^{-T} = VC^T(CC^T)^{-1} \tag{5}$$

illustrating that A is unique as long as the c_k are chosen to make C^T full rank. Otherwise, the system can be modified to be full rank by adding an additional set of Levenberg-Marquardt [Lev44] regularization equations $\epsilon I_{n\times n} A^T(\theta_{\mathcal{R}}) = \epsilon A^T(\theta_0)$ where θ_0 are default values for the rig.

Assume that the tracker executes an error-free geometry reconstruction to obtain geometry v from an image I; in addition, assume that the tracker is linear in the reconstructed geometry v so that $c = \mathcal{T}(I;\theta_{\mathcal{R}}) = B(\theta_{\mathcal{R}})v$ where B is a size $n\times m$ matrix. B satisfies

$$C = B(\theta_{\mathcal{R}})V \implies CC^T(CC^T)^{-1} = B(\theta_{\mathcal{R}})VC^T(CC^T)^{-1} \tag{6}$$

$$I_{n\times n} = B(\theta_{\mathcal{R}})A(\theta_{\mathcal{R}}) \tag{7}$$

implying that B is a size $n\times m$ left inverse of A. This notion of a tracker consisting of a *geometry reconstruction followed by the "inverse" of a rig operation* (Equation 6) is conceptually useful. If C^T is not full rank, then the additional rows of C^T and V^T that were added to Equation 4 need to be included as additional columns in Equation 6 to derive Equation 7. These can be seen as additional (c_k, v_k) pairs.

In a real-world scenario, a tracker will not execute an error-free geometry reconstruction. Thus, suppose instead that the tracker erroneously reconstructs geometry $\hat{v}_k$ from the images I_k; then, the left-hand side of Equation 6 instead becomes $C = \hat{B}(\theta_{\mathcal{T}})\hat{V}$ where $\hat{B}$ rectifies errors in geometry reconstruction. Since $\hat{B} \neq B$, $\hat{B}$ must necessarily depend on different parameters. Let $\theta_{\mathcal{T}}$ be these rig parameters used by the tracker. Similar to Equations 4 and 5, this leads to

$$\hat{B}(\theta_{\mathcal{T}}) = C\hat{V}^T(\hat{V}\hat{V}^T)^{-1} \tag{8}$$

when $\hat{V}^T$ is full rank; otherwise, Levenberg-Marquardt equations of the form $\epsilon I_{mxm}\hat{B}^T(\theta_{\mathcal{T}}) = \epsilon\hat{B}^T(\theta_0)$ can be included. Note that the regularization of $\hat{V}^T$ adds columns to C and $\hat{V}$ that are entirely different from the columns that were added to C and V to regularize C^T in Equation 4. Thus, there are two reasons that $\hat{B}$ in Equation 8 is no longer a left inverse of A in Equation 5. Not only have reconstruction errors caused V to change to $\hat{V}$, but the rig-solve based regularization used to obtain Equation 7 is different from the tracker-solve based regularization used to obtain Equation 8. Since $\hat{B}$ is no longer a left inverse of A, *the rig used for the "inverse" operation in the tracker should be inconsistent with the animation rig*.

To represent this inconsistency, let $\hat{A}(\theta_{\mathcal{T}})$ represent the rig used for the "inverse" operation inside the tracker, noting that it depends on the different set of parameters $\theta_{\mathcal{T}} \neq \theta_{\mathcal{R}}$ than those used by the animation rig $A(\theta_{\mathcal{R}})$. To make this more clear, we rewrite the direct linear interpretation of the tracker $C = \hat{B}(\theta_{\mathcal{T}})\hat{V}$ to a more appropriate "rig inversion" paradigm which sets up a system of equations $\hat{A}(\theta_{\mathcal{T}})C = \hat{V}$ which is solved by the tracker to determine C. This leads to

$$\hat{A}(\theta_{\mathcal{T}}) = \hat{V}C^T(CC^T)^{-1} \neq VC^T(CC^T)^{-1} = A(\theta_{\mathcal{R}}) \tag{9}$$

where the regularization of C and $\hat{V}$ is the same as the regularization of C and V in Equation 4, so the only difference between the left-hand side and right-hand side of Equation 9 is $\hat{V}$ versus V. The separation between the tracking rig parameters $\theta_{\mathcal{T}}$ and animation rig parameters $\theta_{\mathcal{R}}$ will be crucial for setting up the optimization problem (Section 5) and implicit differentiation (Section 6).

5. Optimizing the Tracker's Parameters

Although the linearization in Section 4 provides motivation, the vast majority of state-of-the-art trackers will be nonlinear, and not readily differentiable. We only assume that the tracker has been trained to work via Equation 3, or a similar in spirit approach. Going forward, we write $\mathcal{T}(I;\theta_{\mathcal{T}})$ to stress that the tracker depends on the rig parameters in some perhaps unknown or complex fashion. Then, we propose using a modified version of Equation 3, i.e.

$$\min_{\theta_{\mathcal{T}}} \sum_k \Big(\gamma_1 ||\mathcal{T}(I_k;\theta_{\mathcal{T}}) - c_k||_2^2 + \gamma_2 ||\mathcal{R}(\mathcal{T}(I_k;\theta_{\mathcal{T}});\theta_{\mathcal{R}}) - v_k||_2^2\Big) + \gamma_\epsilon ||\theta_{\mathcal{T}} - \theta_{\mathcal{R}}||_2^2 \tag{10}$$

to fine-tune the tracker by modifying the $\theta_{\mathcal{T}}$ parameters of the underlying animation rig.

In our experience, most trackers have been trained to provide good geometric reconstruction. This means that the γ_2 term in Equation 10 will be small when $\theta_{\mathcal{T}} = \theta_{\mathcal{R}}$. However, in our experience, trackers often struggle to output semantically correct rig controls. This means that the γ_1 term in Equation 10 will typically be large when $\theta_{\mathcal{T}} = \theta_{\mathcal{R}}$. Thus, our goal is to minimize the γ_1 term while staying close to the constraint surface that keeps the γ_2 term small. The γ_ϵ term emphasizes that the rig is being fine-tuned, softly constraining the minimization of the γ_1 term on or near the γ_2 constraint surface to occur within a ball about $\theta_{\mathcal{R}}$; otherwise, $\theta_{\mathcal{T}}$ can drift too much.

6. Differentiating the Tracker

In some scenarios, one may not have access to the internal workings of the tracker. In other scenarios, the tracker may be open source yet still not readily differentiable. In either scenario, $\frac{\partial \mathcal{T}}{\partial \theta}$, which is required to backpropagate through Equation 10, cannot be computed explicitly. Thus, we make the reasonable ansatz that the tracker actually uses its internal rig to provide regularization; that is, we assume that the tracker's internal reconstruction is credible. This can be written as

$$\hat{v}(I;\theta_{\mathcal{T}}) = v(I) + \tilde{v}(I;\theta_{\mathcal{T}}) = \mathcal{R}(\mathcal{T}(I;\theta_{\mathcal{T}});\theta_{\mathcal{T}}) \tag{11}$$

where v is the ground-truth geometry and $\tilde{v}$ is an erroneous perturbation away from the ground-truth geometry; then, the ansatz that the tracker uses its internal rig to provide regularization can be restated as a claim that $\tilde{v}$ should be small regardless of θ. Differentiating Equation 11 with respect to θ leads to

$$\begin{aligned} \frac{\partial \mathcal{R}(c;\theta_{\mathcal{T}})}{\partial c}\bigg|_{c=\mathcal{T}(I;\theta_{\mathcal{T}})} \frac{\partial \mathcal{T}(I;\theta)}{\partial \theta}\bigg|_{\theta=\theta_{\mathcal{T}}} = \\ - \frac{\partial \mathcal{R}(\mathcal{T}(I;\theta_{\mathcal{T}});\theta)}{\partial \theta}\bigg|_{\theta=\theta_{\mathcal{T}}} + \frac{\partial \tilde{v}(I;\theta)}{\partial \theta}\bigg|_{\theta=\theta_{\mathcal{T}}} \end{aligned} \tag{12}$$

as an implicit equation for $\frac{\partial \mathcal{T}}{\partial \theta}$. Note that $\frac{\partial v}{\partial \theta} = 0$, since v is the ground-truth geometry.

Although one could assume that $\tilde{v}$ is always small enough to remove it from Equation 11 and its derivative from Equation 12, it is also possible to estimate $\frac{\partial \tilde{v}}{\partial \theta}$. See the supplementary material for our exploration of this idea.

7. Linear Examples

We first test our implicit differentiation approach outlined in Sections 5 and 6 in a controlled environment by using small-scale linearized examples. By doing this, we can compare numerical optimizations based on our method to the derived analytic solutions. Assuming linearity of the rig according to Equation 4 gives

$$\frac{\partial \mathcal{R}(c,\theta)}{\partial c} = A(\theta); \quad \frac{\partial \mathcal{R}(c,\theta)}{\partial \theta} = \begin{bmatrix} c^T & 0_{1x3} & 0_{1x3} \\ 0_{1x3} & c^T & 0_{1x3} \\ 0_{1x3} & 0_{1x3} & c^T \end{bmatrix} \tag{13}$$

for $A \in \mathbb{R}^{3\times 3}$ where $\frac{\partial \mathcal{R}}{\partial \theta} \in \mathbb{R}^{3\times 9}$ and θ_ℓ for $\ell = 1 \ldots 9$ are the entries of A enumerated in row-major order. Equation 10, including a γ_3 term for internal tracker reconstruction quality, can be written as

$$\begin{aligned} \min_{\theta_{\mathcal{T}}} \sum_k \Big(\gamma_1 ||\mathcal{T}(v_k;\theta_{\mathcal{T}}) - c_k||_2^2 + \gamma_2 ||A(\theta_{\mathcal{R}})\mathcal{T}(v_k;\theta_{\mathcal{T}}) - v_k||_2^2 + \\ \gamma_3 ||\hat{A}(\theta_{\mathcal{T}})\mathcal{T}(v_k;\theta_{\mathcal{T}}) - v_k||_2^2 \Big) + \gamma_\epsilon ||\theta_{\mathcal{T}} - \theta_{\mathcal{R}}||_2^2 \end{aligned} \tag{14}$$

assuming that the tracker perfectly reconstructs v from I, eliminating I entirely. Recall that $\hat{A} \neq A$ from Equation 9 and the discussion thereafter.

Motivated by the γ_1 term in Equation 14, the (c_k, v_k) pairs can be stacked together to obtain $\hat{A}^{-1}V - C$ where $V^T\hat{A}^{-T} = C^T$ can be solved separately for each column of $\hat{A}^{-T}$. Since Equation 14 solves for $\hat{A}$, not $\hat{A}^{-1}$, a better alternative is to solve $C^T\hat{A}^T = V^T$ using least squares, minimum norm, or Levenberg-Marquardt. This direct method minimizes

$$\mathcal{L}_D = \sum_i ||C^T\hat{A}^T(\theta_{\mathcal{T}})e_i - V^T e_i||_2^2 \tag{15}$$

where i loops through the rows of $\hat{A}$. Equation 15 will be used to evaluate the efficacy of various approaches to solving Equation 14.

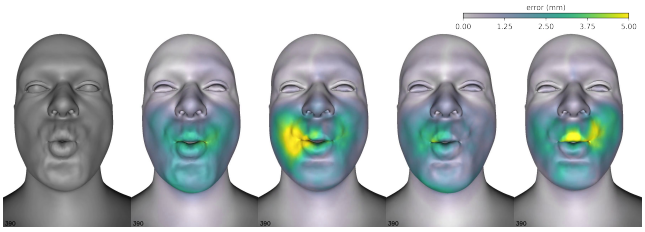

Figure 5: *Ground-truth reconstructed geometry (left) as compared to the geometry output by the tracker using θ_A, $\hat{\theta}_A$, θ_S, $\hat{\theta}_S$, respectively. This corresponds to frame 390, which was chosen because it has relatively large errors, of Figure 6.*

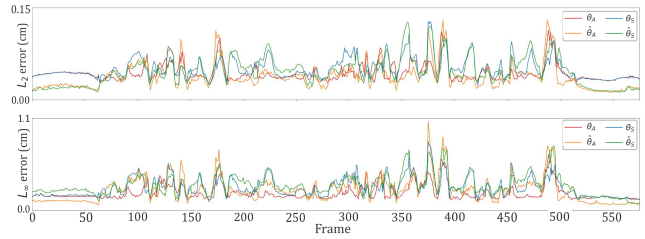

Figure 6: *The tracker was able to adequately minimize reconstruction errors on a pangram using any of θ_A, $\hat{\theta}_A$, θ_S, $\hat{\theta}_S$, even though the output controls (and thus the semantic interpretation) can vary significantly.*

7.1. Solver Efficacy

The following section explains the details of our experiments regarding the optimization of Equation 14 as compared to the direct method in Equation 15 to show that our objective function optimized using implicit differentiation is correct in variations of this analytically derivable case. Consider three sets of linearly independent data pairs (c_k, v_k) and a fourth pair (c_4, v_4) which is linearly dependent, all of which are consistent with a linear rig $A \in \mathbb{R}^{3\times 3}$.

Method	$\mathcal{L}_D$	$\mathcal{L}_{\gamma_1}+\mathcal{L}_{\gamma_2}+\mathcal{L}_{\gamma_3}$	$\|\|\hat{A}-A_2\|\|_F^2$	$\|\|\hat{A}-A\|\|_F^2$
Direct	1.71E−12	6.89E−13	0	2.77E+0
γ_1 only	4.228E−9	4.90E−12	7.72E+1	6.75E+1
γ_2 only	2.56E−8	1.66E−7	1.62E+2	1.46E+2
γ_1 & γ_ϵ	1.43E−12	7.33E−12	2.77E0	5.74E−9
γ_2 & γ_ϵ	9.94E−14	2.13E−12	2.77E0	2.05E−10

Table 1: *Optimizing Equation 15 (top row) and different loss terms in Equation 14 (other rows) on an underdetermined system (Section 7.1).*

Method	$\mathcal{L}_D$	$\mathcal{L}_{\gamma_1}$	$\mathcal{L}_{\gamma_2}$	$\|\|\hat{A}-\hat{A}_{2^*}\|\|_F^2$	$\|\|\hat{A}-A\|\|_F^2$
Direct	1.71E−12	1.20E−12	1.80E−3	2.77E0	2.77E0
γ_1 only	9.89E−9	8.56E−9	1.80E−3	6.66E+1	6.66E+1
γ_2 only	1.13E−2	1.33E−3	8.24E−9	1.13E+2	1.13E+2
γ_1 & γ_ϵ	7.19E−5	1.97E−5	1.24E−3	7.23E−6	9.26E−5
γ_2 & γ_ϵ	1.80E−3	1.33E−3	7.62E−11	1.39E−4	8.44E−9

Table 2: *Optimizing Equation 15 (top row) and different loss terms in Equation 14 (other rows) on an overdetermined system (Section 7.1).*

To simulate an underdetermined system, Table 1 shows the results of using only the first two columns of C and V with initial guess $\hat{A}=I$. The results obtained via the direct least squares method are shown for the sake of comparison. Here, A_2 is the minimum norm solution to Equation 15, unobtainable by Equation 14 without regularizing $\theta_{\mathcal{T}} \to 0$ and undesirable since $\theta_{\mathcal{T}} \to \theta_{\mathcal{R}}$ is preferred. The γ_ϵ term regularizes the unconstrained degree of freedom resulting in $\hat{A} \to A$.

Next, consider randomly perturbing the entries of V by 1% so that pairing C with $\hat{V}$ would yield a perturbation of A, to simulate errors in the tracker's reconstruction. Table 2 shows the results obtained using only the first two columns of C and $\hat{V}$ with an initial guess $\hat{A}=I$. The minimum norm solution is unobtainable via Equation 14, as discussed above. Instead, $\hat{A}_{2^*}$ is computed by specifying a third independent column of C, multiplying it by A, and adding it to $\hat{V}$ before solving $C^T\hat{A}_{2^*}^T=\hat{V}^T$. This makes $\hat{A}_{2^*}^T$ agree with the minimum norm solution in the two constrained degrees of freedom and agree with A in the remaining degree of freedom. The γ_1 and γ_2 terms converge as expected, and neither converges to A or $\hat{A}_{2^*}$, due to the unconstrained degree of freedom; however, an initial guess of $\hat{A}=A$ does result in the γ_1 term and the γ_2 terms driving $\hat{A} \to \hat{A}_{2^*}$ and $\hat{A} \to A$ respectively (as expected). The γ_ϵ term regularizes the unconstrained degree of freedom driving $\hat{A} \to A$; however, it competes with the γ_1 term that aims to drive $\hat{A} \to \hat{A}_{2^*}$ in a two-dimensional subspace.

In summary, our implicit differentiation method can be used to construct useful derivatives even under perturbation of the tracker result, given appropriate regularization. This is important, because the real-world examples will never have a perfect tracker reconstruction, which will result in perturbing the computed derivatives. Numerous additional experiments which further validate our tracker fine-tuning approach can be found in the supplementary material.

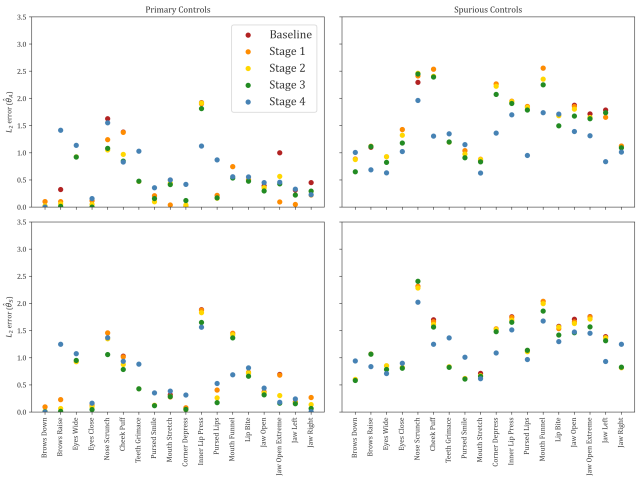

Figure 7: *Summary of the improvement in the L_2 errors according to γ_1 ($\mathcal{L}_{\gamma_1}$) throughout all four stages of our proposed refinement process using the open-source tracker and MetaHuman rig framework. The improvement in the primary controls is shown to the left, and the improvement in the spurious controls is shown to the right. Optimizing $\theta_A \to \hat{\theta}_A$ is shown on the top, and optimizing $\theta_S \to \hat{\theta}_S$ is shown on the bottom. Since Stage 4 expressions are optimized jointly to get better overall performance the accuracy on some expressions may decrease (e.g. $\hat{\theta}_S$ Spurious Controls Jaw Right). Note that data points for later stages may occlude earlier stages if their value is the same.*

8. Simon Says Animation Rig Creation

To perform rig calibration, we augment the Simon-Says approach proposed in [ZOHF24]. Our carefully chosen set of Simon-Says expressions is used to calibrate each rig to the motion signatures of the relevant user. The expression set should cover the most important rig parameters while remaining straightforward for an average person to make. In total, nineteen expressions are used for person-to-person retargeting. For person-to-game/VR character retargeting, the expressions should be chosen in a manner that captures the distinct motion signatures of the game/VR character.

For each expression, an image $\phi(\mathcal{R}(c;N,\theta_{\mathcal{R}}))$ is shown to the user. Here, ϕ represents a renderer. The user then attempts to replicate each expression. When they feel as though they are correctly reproducing the expression, an image I of their face is recorded. Visual cues enable non-experts to quickly understand each expression, especially when the cues are similar in appearance to the user. Importantly, since the animation rig is unlikely to already contain the correct motion signatures of the user, it is incorrect for the user to simply aim to match each image. Therefore, we add concise verbal descriptions for each pose (Figure 3) written with the help of a technical artist, which is not present in prior work.

Once the Simon-Says session is complete, the captured images can be used to reconstruct geometry. Due to the variety of existing geometry reconstruction approaches, we run experiments using three state-of-the-art methods: MetaHuman Animator [Epi25], MICA [ZBT22b], and SMIRK [RFD*24] (see Table 5). Given the (c,v) pairs, Equation 1 can be solved to determine rig parameters

θ_S that better capture the user's motion signature. Expert riggers typically utilize several constraints during the rig design process. For example, "jaw open" + "lips together" should properly seal the lips together. It is straightforward to incorporate such constraints by including additional (c, v) pairs, representing these constraints, into the Equation 1 solve. We added pairs based on technical artist feedback to improve upon the existing Simon-Says baseline system. Although these surrogate pairs are not user-specific, they do help with regularization (Figure 4).

9. Fine-tuning the Tracking Rig

Assuming that the performer has had their original rig θ_A calibrated to some θ_S using Simon-Says in Section 8, we further optimize that rig using Equation 10 to be more appropriate for tracking. The Simon-Says images or any other images can be used for this, as long as aspirational (I, c) pairs exist. Then, $\hat{\theta}_A$ and $\hat{\theta}_S$ will be the fine-tuned original and Simon-Says rigs, respectively.

A typical rig contains primary controls that retarget well and tweaker controls that modify the facial shapes. Unfortunately, when a tracker ignores the tweaker controls, the primary controls end up polluted as they work too hard to explain geometry that would have been better explained by tweaker controls. Thus, a tracker should aim to use the entire set of controls; however, it should lean more heavily on the primary controls as opposed to the tweaker controls. The trick is to figure out how to use just enough of the tweaker controls to obtain non-polluted values for the primary controls.

9.1. Optimization Strategy

In this section, we propose an optimization strategy for fine-tuning tracker output derived from the above motivation. As shown in Figures 6 and 5, changing $\theta_A \rightarrow \theta_S$ via Simon-Says or modifying $\theta_A \rightarrow \hat{\theta}_A$ and $\theta_S \rightarrow \hat{\theta}_S$ has little effect on the ability of a tracker to invert the rig and match the geometry; however, the tracker output and thus the semantic interpretation can vary significantly.

Our proposed strategy utilizes variations of the full tracker $\mathcal{T}$ and rig $\mathcal{R}$ to achieve this goal. Let c_D represent a subset of the rig's controls and let θ_D be the subset of rig parameters associated with those controls. $\mathcal{R}_D$ is a modified rig that can only be articulated in that limited subspace and $\mathcal{T}_D$ can only use that subspace to explain geometry. Thus, γ_1^D and γ_2^D when used in Equation 10 (instead of γ_1 and γ_2) represent weights on these modified loss terms when substituting out the rig/tracker (i.e., using $\mathcal{T}_D$ and $\mathcal{R}_D$ instead of $\mathcal{T}$ and $\mathcal{R}$). Similarly, $\mathcal{T}_\mathcal{H}$ is a filtered version of $\mathcal{T}$ which can use all controls to explain geometry but only outputs nonzero values for the relevant controls in c_D ($\gamma_1^\mathcal{H}$ represents this modification to the γ_1 term).

For a visual overview of the fine-tuning process, see Figure 2. We proceed in four stages, optimizing perturbations of Equation 10. In the first stage, for each given c_k, γ_1^D and γ_2^D are used to optimize the relevant θ_D for the expression. In the second stage, we switch from $\mathcal{T}_D$ to $\mathcal{T}_\mathcal{H}$ (replacing γ_1^D with $\gamma_1^\mathcal{H}$ and γ_2^D with $\gamma_2^\mathcal{H}$). In the third stage, again optimizing the same rig parameters, we aim to minimize the contributions of the spurious controls by adding a second $\gamma_1^\mathcal{H}$ term that selects problematic spurious controls aiming

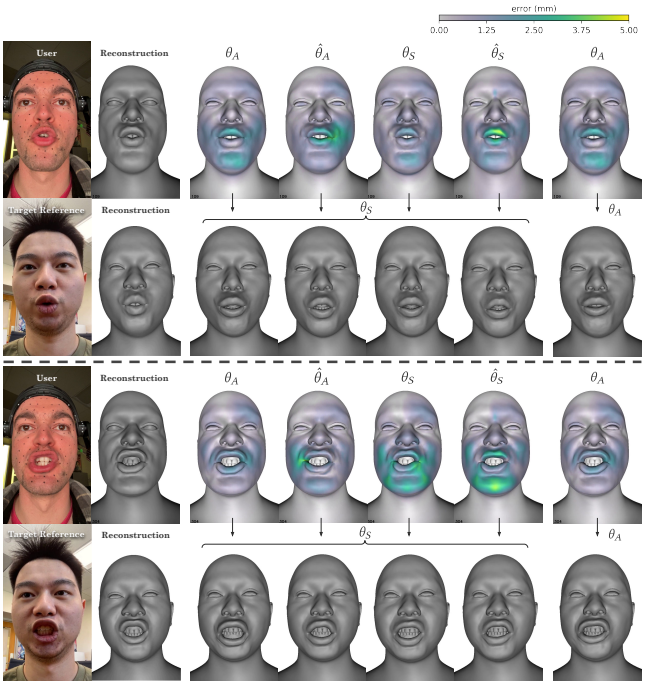

Figure 8: *Open source tracker: retargeting to another identity. The top rows show geometry errors for the different trackers, and the bottom rows show the retargeted results. Both subjects are speaking the same word but have a different motion signature. The last column shows a baseline, not using any of our approaches, for the sake of comparison. Notice the lip shapes in the baseline (rightmost) retarget versus the lip shapes of our $\hat{\theta}_S$ result as compared to the Target Reference image.*

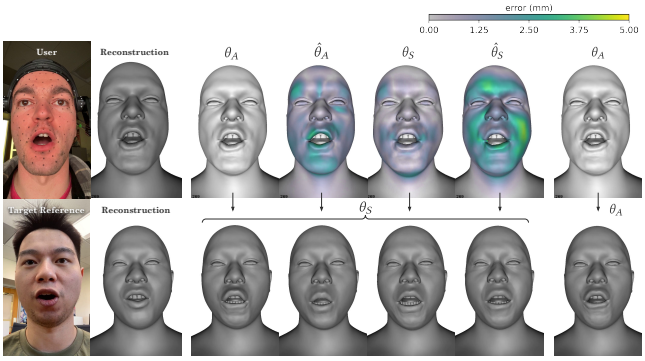

Figure 9: *Closed source tracker: retargeting to another identity. The top row shows geometry errors for the different trackers, and the bottom row shows the retargeted results. Both subjects are speaking the same word but have a different motion signature. The last column shows the high quality state-of-the-art MetaHuman Animator baseline for the sake of comparison. Notice the lower lip shape and upper lip bulge in the baseline (rightmost) retarget versus the lip shapes of our $\hat{\theta}_S$ result as compared to the Target Reference image (and reconstruction).*

to minimize their contribution. At this point in the process, the hope would be that the rig parameters θ_D corresponding to the primary controls c_D have been adjusted to explain as much of the expression as possible. In the final (fourth) stage of the process, rig parameters corresponding to spurious controls are optimized. Since the spurious controls can have a lot of overlap in rig parameters, every relevant expression should be included in the sum in Equation 10 when considering each in turn. Further implementation details to reproduce our results can be found in the supplementary material and code attachment.

Importantly, for all stages, we do not use the minimizer of the current objective function as the result. Instead, motivated by machine learning, we use the optimization simply to generate samples in parameter space and supervise optimization via the full tracker $\mathcal{T}$, choosing the resulting parameters that minimize overall controls error, not any subset or filtered version. Figure 7 shows a summary of how an open source tracker $\mathcal{T}$ improves over all stages. The strategy we present is only one of many possible ways to improve the tracker using our tools; for example, it may be beneficial to optimize the columns corresponding to spurious controls first instead of last. However, these results do demonstrate that our approach is both effective and tractable. In particular, any optimal strategy would focus on animator preferences for the specific use case.

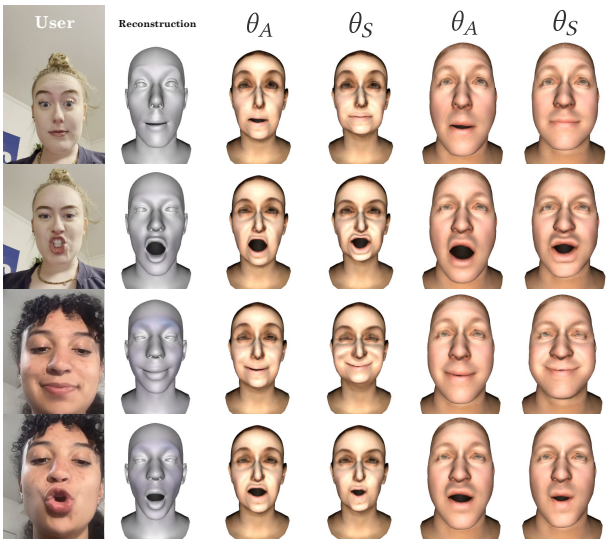

Figure 10: *Semantic FLAME framework: retargeting to another identity. Even though the geometry reconstructions are not accurate, our method (θ_S) learns the correct semantic intent from those shapes as compared to the baseline (θ_A), as long as the reconstruction method is consistent. Notice lip sealing improvements in the top row, improved lip funnel shape in the second row, and mouth corner tightness in the bottom row.*

10. Experiments

10.1. Open Source Trackers

Having shown that the optimization strategy presented in Section 9 can be used to optimize the rig parameters, this experiment demonstrates the qualitative efficacy of our methods for person-to-person retargeting. To do this, we choose a target subject, reconstruct their neutral geometry, fit it with an animation rig, calibrate that rig via Simon-Says (Section 8) then fine-tune the tracking rig (Section 9). The results shown in Sections 10.1 - 10.3 were all chosen from expressive pangrams with data that remained unseen in the Simon-Says calibration, making it quite useful for evaluating generalization. We utilize an analysis-by-synthesis tracker similar to existing trackers such as MICA [ZBT22b] or MetaHuman Animator [Epi25]. Optimization-based trackers like these are not readily differentiable; thus, fine-tuning the tracking rig requires our implicit differentiation method.

10.1.1. MetaHuman rig framework

We demonstrate results after fine-tuning of both the original parameters $\theta_A \rightarrow \hat{\theta}_A$ and Simon-Says modified rig parameters $\theta_S \rightarrow \hat{\theta}_S$ as an ablation study. As a baseline, we show the results obtained using the default MetaHuman rig (i.e. θ_A) in the tracker mapped to the default MetaHuman rig (θ_A instead of θ_S) for the target. For the sake of evaluation, we show images and geometric reconstructions of both the performer and the target making semantically similar but geometrically different expressions.

Figure 8 shows some representative results. As expected, the tracking geometry (top rows) remains similar regardless of whether θ_A, $\hat{\theta}_A$, θ_S, or $\hat{\theta}_S$ was used. In fact, as compared to the ground-truth reconstruction, the tracking geometry can sometimes worsen when modifying the tracking rig; however, these differences are unimportant, since the focus is on obtaining good results on the target. As far as the target is concerned (bottom rows), modifying $\theta_A \rightarrow \hat{\theta}_A$, $\theta_S \rightarrow \hat{\theta}_S$, and $\theta_A \rightarrow \theta_S$ all improve the results.

10.1.2. Semantic FLAME framework

To be able to use recent popular facial reconstruction techniques and compare their effectiveness, we adapt the FLAME [LBB*17] head model to be driven by a set of semantic controls. Our semantic FLAME rig is fit to a neutral identity and calibrated using Simon-Says in the same way as the MetaHuman rigs, but since the PCA-based FLAME shape parameters are not semantically separable we eschew our proposed fine-tuning approach and just present results using the Simon-Says calibration. Qualitative results for these experiments can be found in Figure 10, where θ_A examples are tracked using the original rig parameters and θ_S examples are tracked using our Simon-Says parameters.

10.2. Closed Source Trackers

We also demonstrate our methods' effectiveness in improving tracked animation controls when using a high-performance closed source tracker, the MetaHuman Animator [Epi25]. Because we capture under ideal conditions, the quality of this baseline method is impressive; nevertheless, we are able to improve the resulting semantic results. Since we cannot use $\mathcal{T}_D$ for this tracker, we skip that step during our optimization strategy. Figure 9 highlights representative results in the same manner as Figure 8.

10.3. Puppeteering a Game Rig

Whereas experiments in Sections 10.1 and 10.2 retargeted to another human, here we consider retargeting to a humanoid game character, which uses a different semantic controls space. Given the difficulties associated with remapping controls between two disparate rigs, it seems prudent to utilize the game character rig on the performer as well. Thus, we create a game character representation for the performer via volumetric morphing using the method of [ZOHF24]. We designed a set of basis expressions that better covered the rig controls of the character and calibrated those via Simon-Says. As shown in Figure 11, the low expressivity of the game/VR character rig meant that the Simon-Says results did not exactly match the characteristics of the performer; nevertheless, the vast improvement of the basis shapes led to more accurate semantic interpretation in our retargeting experiments. Like the MetaHuman experiments, we fine-tune the tracking rig according to our strategy (Section 9) after Simon-Says calibration.

Since the closed source MetaHuman Animator tracker is heavily designed to work specifically with the MetaHuman rig framework, our puppeteering experiments only use our open source tracker (Section 10.1). Also, unlike the previous examples, we use an animator's interpretation of the user's pose for reference and label pre-Simon-Says rig parameters θ_P for "puppet". For results on a pangram, see Figure 12. Although we did not expect the puppet rig to be useful without Simon-Says calibration, we show θ_P and $\hat{\theta}_P$ results as an ablation study.

10.4. Synthetic Data

Prior experiments in this section qualitatively evaluated the semantic accuracy of calibrated tracking. To further validate our approach using ground-truth animation labels, we perform synthetic data experiments. Let θ_{GT} be the rig parameters of a "ground-truth" MetaHuman identity. Starting with a different MetaHuman, we morphed its rig using the method of [ZOHF24] to match the θ_{GT} identity. The morphed rig with parameters θ_A has the same neutral identity but a different motion signature. We evaluate both an animator-sculpted set of synthetic expressions and a randomly generated set of expressions, none of which are seen as training examples.

10.4.1. Simon-Says

We used a Simon-Says expression set to generate geometry with θ_{GT} and subsequently optimized $\theta_A \rightarrow \theta_S$. For the Simon-Says (training) expressions, θ_{GT} and θ_S behaved similarly, as expected. Generalization of our Simon-Says approach can be tested by comparing the geometry produced by θ_{GT} to that produced by θ_S for other expressions. In particular, the benefits of the approach can be quantified as the decrease in errors obtained by using θ_S instead of θ_A (Table 4).

10.4.2. Tracker fine-tuning

Using the same θ_A and θ_S as the previous experiment, we apply our fine-tuning strategy on both rigs to get $\theta_A \rightarrow \hat{\theta}_A$ and $\theta_S \rightarrow \hat{\theta}_S$ as usual, using synthetic data from the θ_{GT} rig. Ground-truth geometry is generated using the θ_{GT} rig and we compare the animation curves tracked using each of $\theta_A, \hat{\theta}_A, \theta_S, \hat{\theta}_S$ to the ground-truth animation parameters. We use the open source MICA-style tracker for this experiment. Table 3 shows the average errors in rig controls across the synthetic animations. The results shown also function as an ablation study, validating the inclusion of both our Simon-Says and fine-tuning components.

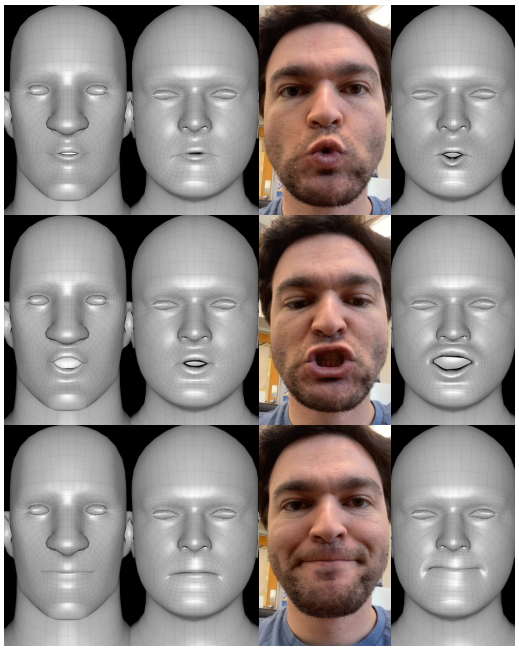

Figure 11: *Simon-Says process on specific phoneme-level controls ("OO", "CH", "M/B/P") unique to the game character rig (one on each row). Left to right: Game character rig, morphed rig, user's expression, Simon-Says results. The limited expressivity of the rig framework makes it impossible to match the user's motion signature exactly. Note that the first two columns provide very little usable input for the user, so the user needs to rely more heavily on the written description of each desired expression.*

Expression Set	Norm	θ_A	$\hat{\theta}_A$	θ_S	$\hat{\theta}_S$
Animator	L_2	1.16	1.04	1.12	**0.992**
	L_∞	0.683	0.666	0.687	**0.660**
Random	L_2	0.970	0.927	0.751	**0.748**
	L_∞	0.405	0.396	0.309	**0.307**

Table 3: *Average error in animation controls for novel synthetic expressions. We give the L_2 (euclidean distance) and L_∞ (maximum component) errors of the tracked animation controls as compared to ground-truth animation controls averaged over the dataset. We use trackers with four sets of internal parameters: Uncalibrated (θ_A), calibrated (θ_S), uncalibrated with tracker fine-tuning ($\hat{\theta}_A$), and calibrated with tracker fine-tuning ($\hat{\theta}_S$).*

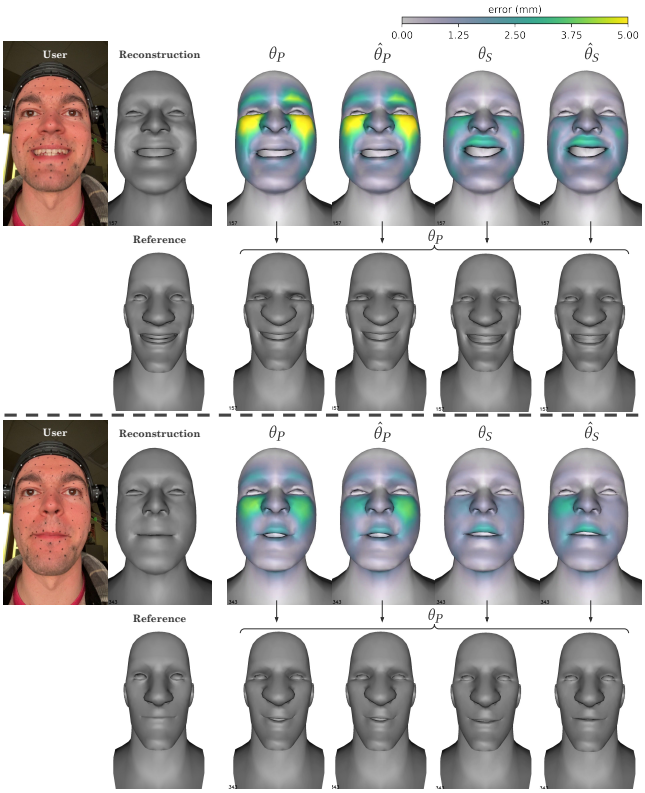

Figure 12: *Open source tracker: puppeteering a game rig. The top rows show geometry errors for the different trackers, and the bottom rows show the retargeted results. Modifying* $\theta_P \rightarrow \hat{\theta}_P$ *and* $\theta_S \rightarrow \hat{\theta}_S$ *both improve the retargeted results, as compared to the handcrafted reference pose under the reconstruction. Notice the unnecessary squinting and incorrect lower lip shape fixed in the top example, and the lip press significantly closer to the reference in the bottom example despite the reconstructions never having an accurately sealed lip press.*

10.5. Animation-focused Quantitative Metrics

From an animator's perspective there exist some key metrics by which one can quantitatively judge the quality of animation curves from automated tracking. Sparsity, the number of controls active during any given frame, is often not achieved to a sufficient degree due to trackers overutilizing spurious controls to explain the geometry reconstruction. We calculate sparsity by counting the number of controls that are above the threshold value of 0.1. Animators are also concerned with the usage of primary controls versus tweaker controls, generally preferring to animate a performance with as little usage of tweaker controls as possible. The tweaker metric is calculated as the sum of all controls on a given frame labeled as "tweaker", according to the rig definition. We evaluate the quality of our results for each tracker and rig framework according to these metrics, seeing improvement in nearly all cases (Table 5).

Expression Set	θ_A	θ_S
Animator	0.487	**0.285**
Random	0.693	**0.370**

Table 4: *Average vertex* L_2 *error in mm for novel synthetic expressions, showing that our calibrated rig with parameters* θ_S *is closer to the ground truth rig than the uncalibrated rig with parameters* θ_A.

10.6. Technical Details

We run all of our experiments on a single workstation with an AMD EPYC 7R32 processor (2.80GHz) and 128 GB RAM. The workstation is equipped with a NVIDIA A10G GPU. Our calibration is not meant to run in real-time, and only needs to be performed once per actor-rig combination. The Simon-Says calibration (Section 8) takes around 1 minute to run and the full multi-stage fine-tuning (Section 9) takes around one hour on our machine. Calibrating rigs with a very high number of internal parameters such as the MetaHuman Animator will have a larger memory requirement; our code sample minimizes this impact by masking out irrelevant parameters.

11. Conclusion

Unfortunately, many trackers produce animation curves that require significant cleanup effort by artists to obtain results that can be used effectively for retargeting. Our mathematical analysis illustrated that a tracker can be thought of as geometric reconstruction followed by rig inversion, highlighting the fact that those who merely stress the efficacy of their geometric reconstructions overlook a key component of the problem. Moreover, we showed that it can even be beneficial to relax the accuracy of the geometric reconstruction as a tradeoff for obtaining better and more usable animation curves via the rig inversion.

We showed that the rig parameters could be modified to obtain improved results for person-to-person or virtual character retargeting. To do this, we utilized a Simon-Says calibration process to personalize the performer's animation rig before modifying the tracker parameters via our implicit differentiation approach. This implicit differentiation is critical to be able to apply the method to optimization-based trackers. Although we show the efficacy of our approach, further research may be able to refine our optimization strategies or propose new ones based on the tools we provide.

Because our methods can be used to improve the realism of facial animation on digital doubles, it is important to consider the ethical implications of such technology. We condemn the use of our techniques and those related to ours for malicious intent, the spread of disinformation, or intent to cause harm. We have explicit consent from the actors in this paper for use of their likeness and encourage others to similarly respect the privacy and identity of actors, contractors, and users.

One important limitation of our work is the sensitivity to calibration quality. If the user does not accurately separate out one expression during the Simon-Says capture, it can pollute the rig

	Rig	Tracker	Sparsity (↓) Baseline	Sparsity (↓) Ours	Tweakers (↓) Baseline	Tweakers (↓) Ours
ROM	MetaHuman	OS1	0.1209	**0.0939**	0.0131	**0.0128**
	FLAME	OS2	0.1628	**0.1237**	0.0538	**0.0173**
	FLAME	OS3	0.1681	**0.0868**	0.0562	**0.0077**
	MetaHuman	MHA	0.1251	**0.0904**	0.0299	**0.0291**
	Game	OS1	**0.2044**	0.2333	0.1067	**0.0777**
Pangram	MetaHuman	OS1	0.1112	**0.1046**	0.0139	**0.0125**
	FLAME	OS2	0.1782	**0.1695**	0.0695	**0.0315**
	FLAME	OS3	**0.1568**	0.1799	0.0472	**0.0215**
	MetaHuman	MHA	**0.0956**	0.1064	0.0018	**0.0012**
	Game	OS1	0.2448	**0.2394**	0.1057	**0.0639**

Table 5: *Animation-centric metrics. OS1: L-BFGS tracker using MetaHuman Animator [Epi25] for geometry reconstruction. OS2: L-BFGS tracker using [ZBT22b] for geometry reconstruction. OS3: L-BFGS tracker using [RFD*24] for geometry reconstruction. MHA: MetaHuman Animator. The metrics, which are described in Section 10.5, are averaged over all frames.*

solve and tracker fine-tuning. This can be somewhat remedied by detecting and factoring out extraneous expressions, but further research is required in that area. Even with excellent calibration, extreme, highly asymmetrical expressions might not fare as well after our symmetric-only calibration, which is an interesting avenue for follow-up work. In addition, since our approach does not consider temporal consistency, artifacts such as tracking jitter or dirty animation curves would not be ameliorated by our approach (some of which is visible in the supplementary video).

Another key limitation of the methods presented here is the restricted domain of only trackers which are conditioned upon rig parameters, like the MetaHuman Animator. The exploration of applying rig calibration to neural trackers which skip the rig inversion process, and therefore are not conditioned upon any rig parameters, can be of great interest, especially as those methods improve to a high enough quality to be used in production. We hope that this paper may constitute a basis for future work into high efficacy retargeting and encourage those in the field to consider rig calibration as a component in the retargeting process.

References

[ACC*18] ANEJA D., CHAUDHURI B., COLBURN A., FAIGIN G., SHAPIRO L., MONES B.: Learning to generate 3d stylized character expressions from humans. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)* (2018), pp. 160–169. doi:10.1109/WACV.2018.00024. 2

[Aut25] AUTODESK, INC.: Maya, 2025. URL: https://www.autodesk.com/products/maya. 2

[BBC*24] BAERT K., BHARADWAJ S., CASTAN F., MAUJEAN B., CHRISTIE M., ABREVAYA V., BOUKHAYMA A.: Spark: Self-supervised personalized real-time monocular face capture. doi:10.1145/3680528.3687704. 2

[BCGF19] BAO M., CONG M., GRABLI S., FEDKIW R.: High-quality face capture using anatomical muscles. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019), pp. 10794–10803. doi:10.1109/CVPR.2019.01106. 2

[Ble25] BLENDER ONLINE COMMUNITY: Blender, 2025. URL: https://www.blender.org/. 2

[BLW*24] BUEHLER M. C., LI G., WOOD E., HELMINGER L., CHEN X., SHAH T., WANG D., GARBIN S., ORTS-ESCOLANO S., HILLIGES O., LAGUN D., RIVIERE J., GOTARDO P., BEELER T., MEKA A., SARKAR K.: Cafca: High-quality novel view synthesis of expressive faces from casual few-shot captures. In *ACM SIGGRAPH Asia 2024 Conference Paper*. 2024. doi:10.1145/3680528.3687580. 2

[BODO20] BAILEY S. W., OMENS D., DILORENZO P., O'BRIEN J. F.: Fast and deep facial deformations. *ACM Transactions on Graphics 39*, 4 (Aug. 2020), 94:1–15. Presented at SIGGRAPH 2020, Washington D.C. URL: http://graphics.berkeley.edu/papers/Bailey-FDF-2020-07/, doi:10.1145/3386569.3392397. 2

[BV99] BLANZ V., VETTER T.: A morphable model for the synthesis of 3d faces. In *SIGGRAPH* (1999). 2

[BWL*24] BAI S., WANG T.-L., LI C., VENKATESH A., SIMON T., CAO C., SCHWARTZ G., SARAGIH J., SHEIKH Y., WEI S.-E.: Universal facial encoding of codec avatars from vr headsets. *ACM Trans. Graph. 43*, 4 (jul 2024). 2

[BZH*23] BHARADWAJ S., ZHENG Y., HILLIGES O., BLACK M. J., ABREVAYA V. F.: Flare: Fast learning of animatable and relightable mesh avatars. *ACM Transactions on Graphics 42* (Dec. 2023), 15. doi:https://doi.org/10.1145/3618401. 2

[CBE*15] CONG M., BAO M., E J. L., BHAT K. S., FEDKIW R.: Fully automatic generation of anatomical face simulation models. In *Proceedings of the 14th ACM SIGGRAPH / Eurographics Symposium on Computer Animation* (New York, NY, USA, 2015), SCA '15, Association for Computing Machinery, p. 175–183. 2

[CCGB22] CHANDRAN P., CICCONE L., GROSS M., BRADLEY D.: Local anatomically-constrained facial performance retargeting. *ACM Trans. Graph. 41*, 4 (jul 2022). 2

[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* (New York, NY, USA, 2022), SA '22, Association for Computing Machinery. 2

[CLC*25] CHOI Y., LEE I., CHA S., KIM S., JUNG S., NOH J.: Deep-learning-based facial retargeting using local patches. *Computer Graphics Forum 44*, 1 (2025), e15263. doi:https://doi.org/10.1111/cgf.15263. 2

[CYSN25] CHA S., YOON S., SEO K., NOH J.: Neural face skinning for mesh-agnostic facial expression cloning. *Computer Graphics Forum n/a*, n/a (2025), e70009. doi:https://doi.org/10.1111/cgf.70009. 2

[DCFN06] DENG Z., CHIANG P.-Y., FOX P., NEUMANN U.: Animating blendshape faces by cross-mapping motion capture data. In *Proceedings of the 2006 Symposium on Interactive 3D Graphics and Games* (New York, NY, USA, 2006), I3D '06, Association for Computing Machinery, p. 43–48. doi:10.1145/1111411.1111419. 2

[DGHG*24] DIB A., GUSTAVO HAFEMANN L., GOT E., ANDERSON T., FADAEINEJAD A., CRUZ R. M. O., CARBONNEAU M.-A.: Mosar: Monocular semi-supervised model for avatar reconstruction using differentiable shading. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2024), pp. 1770–1780. doi:10.1109/CVPR52733.2024.00174. 2

[DHT*00] DEBEVEC P., HAWKINS T., TCHOU C., DUIKER H.-P., SAROKIN W., SAGAR M.: Acquiring the reflectance field of a human face. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques* (USA, 2000), SIGGRAPH '00, ACM Press/Addison-Wesley Publishing Co., p. 145–156. 2

[Epi25] EPIC GAMES: Metahuman animator, 2025. URL: https://www.unrealengine.com/en-US/digital-humans. 2, 6, 8, 11

[KDP*24] KAVAN L., DOUBLESTEIN J., PRAZAK M., CIOFFI M.,

ROBLE D.: Compressed skinning for facial blendshapes. In *ACM SIGGRAPH 2024 Conference Papers* (New York, NY, USA, 2024), SIGGRAPH '24, Association for Computing Machinery. doi:10.1145/3641519.3657477. 2

[KJS*21] KIM S., JUNG S., SEO K., I RIBERA R. B., NOH J.: Deep learning-based unsupervised human facial retargeting. *Computer Graphics Forum 40*, 7 (2021), 45–55. 2

[KS21] KIM J., SINGH K.: Optimizing ui layouts for deformable face-rig manipulation. *ACM Trans. Graph. 40*, 4 (July 2021). URL: https://doi.org/10.1145/3450626.3459842, doi:10.1145/3450626.3459842. 2

[LA10] LEWIS J. P., ANJYO K.-I.: Direct manipulation blendshapes. *IEEE Comput. Graph. Appl. 30*, 4 (July 2010), 42–50. URL: https://doi.org/10.1109/MCG.2010.41, doi:10.1109/MCG.2010.41. 2

[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, (Proc. SIGGRAPH Asia) 36*, 6 (2017), 194:1–194:17. URL: https://doi.org/10.1145/3130800.3130813. 2, 8

[Lev44] LEVENBERG K.: A method for the solution of certain non-linear problems in least squares. *Quarterly of Applied Mathematics 2*, 2 (1944), 164–168. 4

[LSSS18] LOMBARDI S., SARAGIH J., SIMON T., SHEIKH Y.: Deep appearance models for face rendering. *ACM Trans. Graph. 37*, 4 (jul 2018). 2

[LWP10] LI H., WEISE T., PAULY M.: Example-based facial rigging. *ACM Trans. Graph. 29*, 4 (jul 2010). 2

[MCW*21] MOSER L., CHIEN C., WILLIAMS M., SERRA J., HENDLER D., ROBLE D.: Semi-supervised video-driven facial animation transfer for production. *ACM Trans. Graph. 40*, 6 (dec 2021). 2

[MLL*24] MING X., LI J., LING J., ZHANG L., XU F.: High-quality mesh blendshape generation from face videos via neural inverse rendering. In *Computer Vision – ECCV 2024: 18th European Conference, Milan, Italy, September 29–October 4, 2024, Proceedings, Part LXX* (Berlin, Heidelberg, 2024), Springer-Verlag, p. 106–125. doi:10.1007/978-3-031-72897-6_7. 2

[NN01] NOH J.-Y., NEUMANN U.: Expression cloning. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 2001), SIGGRAPH '01, Association for Computing Machinery, p. 277–288. 2

[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), Cani M.-P., Ganovelli F., (Eds.), The Eurographics Association. doi:/10.2312/conf/EG2012/stars/183-204. 2

[PLX*25] PAN Y., LIU C., XU S., TAN S., YANG J.: Vasa-rig: Audio-driven 3d facial animation with 'live' mood dynamics in virtual reality. *IEEE Transactions on Visualization and Computer Graphics 31*, 5 (2025), 2416–2425. doi:10.1109/TVCG.2025.3549168. 2

[QSA*23] QIN D., SAITO J., AIGERMAN N., THIBAULT G., KOMURA T.: Neural face rigging for animating and retargeting facial meshes in the wild. In *SIGGRAPH 2023 Conference Papers* (2023). 2

[QZL*24] QIU F., ZHANG W., LIU C., AN R., LI L., DING Y., FAN C., HU Z., YU X.: Freeavatar: Robust 3d facial animation transfer by learning an expression foundation model. In *SIGGRAPH Asia 2024 Conference Papers* (New York, NY, USA, 2024), SA '24, Association for Computing Machinery. doi:10.1145/3680528.3687669. 2

[RFD*24] RETSINAS G., FILNTISIS P. P., DANECEK R., ABREVAYA V. F., ROUSSOS A., BOLKART T., MARAGOS P.: 3d facial expressions through analysis-by-neural-synthesis. In *Conference on Computer Vision and Pattern Recognition (CVPR)* (2024). 2, 6, 11

[RJS24] RACKOVIĆ S., JAKOVETIĆ D., SOARES C.: Refined inverse rigging: A balanced approach to high-fidelity blendshape animation. In *SIGGRAPH Asia 2024 Conference Papers* (New York, NY, USA, 2024), SA '24, Association for Computing Machinery. doi:10.1145/3680528.3687670. 2

[RZL*17] RIBERA R. B. I., ZELL E., LEWIS J. P., NOH J., BOTSCH M.: Facial retargeting with automatic range of motion alignment. *ACM Trans. Graph. 36*, 4 (jul 2017). 2

[SCSN11] SONG J., CHOI B., SEOL Y., NOH J.: Characteristic facial retargeting. *Computer Animation and Virtual Worlds 22*, 2-3 (2011), 187–194. 2

[SLS*12] SEOL Y., LEWIS J., SEO J., CHOI B., ANJYO K., NOH J.: Spacetime expression cloning for blendshapes. *ACM Trans. Graph. 31*, 2 (apr 2012). 2

[SP04] SUMNER R. W., POPOVIĆ J.: Deformation transfer for triangle meshes. *ACM Trans. Graph. 23*, 3 (aug 2004), 399–405. 2

[TRT*24] TAUBNER F., RAINA P., TULI M., TEH E. W., LEE C., HUANG J.: 3d face tracking from 2d video through iterative dense uv to image flow. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2024), pp. 1227–1237. 2

[TZH*24] TRAN P., ZAKHAROV E., HO L.-N., KARMANOV A., BERMUDEZ VENEGAS A., GOLDWHITE M., AGARWAL A., HU L., TRAN A., LI H.: Voodoo xp: Expressive one-shot head reenactment for vr telepresence. *ACM Trans. Graph. 43*, 6 (Nov. 2024). doi:10.1145/3687974. 2

[WHL*04] WANG Y., HUANG X., LEE C.-S., ZHANG S., LI Z., SAMARAS D., METAXAS D., ELGAMMAL A., HUANG P.: High Resolution Acquisition, Learning and Transfer of Dynamic 3-D Facial Expressions. *Computer Graphics Forum* (2004). doi:10.1111/j.1467-8659.2004.00800.x. 2

[YZC*23] YANG L., ZOSS G., CHANDRAN P., GOTARDO P., GROSS M., SOLENTHALER B., SIFAKIS E., BRADLEY D.: An implicit physical face model driven by expression and style. In *SIGGRAPH Asia 2023 Conference Papers* (New York, NY, USA, 2023), SA '23, Association for Computing Machinery. 2

[YZF*23] YANG H., ZHENG M., FENG W., HUANG H., LAI Y.-K., WAN P., WANG Z., MA C.: Towards practical capture of high-fidelity relightable avatars. In *SIGGRAPH Asia 2023 Conference Proceedings* (2023). 2

[ZBT22a] ZIELONKA W., BOLKART T., THIES J.: Instant volumetric head avatars. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2022), 4574–4584. URL: https://api.semanticscholar.org/CorpusID:253761096. 2

[ZBT22b] ZIELONKA W., BOLKART T., THIES J.: Towards metrical reconstruction of human faces. In *European Conference on Computer Vision* (2022). 6, 8, 11

[ZCZ22] ZHANG J., CHEN K., ZHENG J.: Facial expression retargeting from human to avatar made easy. *IEEE Transactions on Visualization and Computer Graphics 28*, 2 (2022), 1274–1287. doi:10.1109/TVCG.2020.3013876. 2

[ZJ24a] ZHU C., JOSLIN C.: A facial motion retargeting pipeline for appearance agnostic 3d characters. In *Proceedings of Computer Graphics International 2024* (New York, NY, USA, 2024), CGI 2024, Association for Computing Machinery, p. 1–15. 2

[ZJ24b] ZHU C., JOSLIN C.: A review of motion retargeting techniques for 3d character facial animation. *Computers & Graphics 123* (2024), 104037. URL: https://www.sciencedirect.com/science/article/pii/S0097849324001729, doi:https://doi.org/10.1016/j.cag.2024.104037. 2

[ZOHF24] ZHU Y., OMENS D., HE H., FEDKIW R.: Democratizing the creation of animatable facial avatars, 2024. URL: https://arxiv.org/abs/2401.16534, arXiv:2401.16534. 6, 9

[ZSCS04] ZHANG L., SNAVELY N., CURLESS B., SEITZ S. M.: Spacetime faces: high resolution capture for modeling and animation. *ACM Trans. Graph. 23*, 3 (aug 2004), 548–558. 2