METAHUMAN-ANYTHING: VISION-LANGUAGE FACE SIMILARITY GROUNDING FOR ENGINE-FRIENDLY DIGITAL HUMAN SCULPTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Automated creation of editable 3D facial assets is increasingly important for digital human production in games and animation. However, image-based 3D face reconstruction methods typically output point clouds, meshes, or parametric face models rather than native characters that can be edited directly in animation engines. MetaHuman's mesh-conforming tools fit a native face template to reconstructed meshes, a step that can alter facial geometry and reduce identity fidelity. We present MetaHuman-Anything, an automated, vision–language-model-guided framework that directly edits native MetaHuman parameters to match a reference portrait. At each iteration, a frozen vision–language Reasoner identifies differences between the reference and the current render. A state-conditioned Grounder translates this analysis into grouped parameter targets and gated updates, while a learned renderer provides feedback for iterative refinement. We construct a corpus pairing rendered faces with editable MetaHuman states to train the framework. On an external set of 102 portraits, our method improves mean target-to-Unreal-Engine-render ArcFace identity similarity by 118.2% over the strongest mesh-conversion baseline under matched fixed-appearance conditions, while producing directly editable native MetaHuman characters.

1 INTRODUCTION

Three-dimensional character modeling is an essential part of game and animation production. Creating a digital human face with a specific identity requires skilled artists to shape its facial features and appearance and deliver an editable, animatable asset. As the number of characters grows, this process demands substantial time and labor. Automating personalized face creation from reference portraits with AI can reduce the work required for each character. To enter a production pipeline, the generated face must also be directly editable in the target engine.

Image-based 3D face reconstruction has advanced in facial geometry and visual detail. DECA, EMOCA, and MICA map images into their respective face-model parameter spaces (Feng et al., 2021; Daněček et al., 2022; Zielonka et al., 2022); other methods produce meshes or point-based representations. These outputs support reconstruction and rendering, but they do not directly specify an editable MetaHuman state. Using them as MetaHuman assets in Unreal Engine requires a separate fitting or conversion step, and the resulting engine render can differ from the reference portrait. The relevant production outcome is therefore the rendered native asset.

Recent vision–language-model (VLM)-guided avatar systems such as SmartAvatar (Huang-Menders et al., 2025) show that visual analysis and rendering feedback can jointly control a parametric character generator. MetaHuman's mesh- and scan-fitting tools provide a route from external geometry to native characters (Epic Games, n.d.). Creating a similar MetaHuman face automatically from a reference portrait also requires translating visual analysis into edits of native parameters and revising those edits after observing the rendered result.

This task presents three challenges. First, a single portrait does not uniquely determine a high-dimensional MetaHuman state: different parameter settings can produce similar images. Second, the native state includes continuous face coefficients and appearance controls alongside discrete asset choices. A visual judgment such as the current face being too narrow must lead to a valid

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

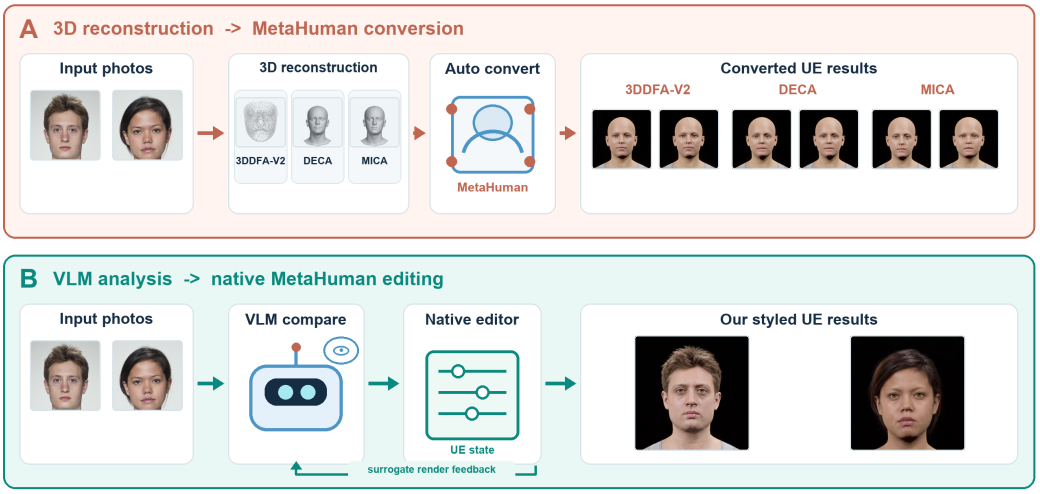

Figure 1: Two routes from a portrait to MetaHuman. Reconstruction baselines convert estimated geometry into a native character (top); our VLM-guided editor adjusts native parameters using rendered feedback (bottom). Cases 141 and 030 illustrate the routes. The lower styled portraits are unscored; quantitative comparisons use matched fixed-appearance UE renders.

update in the appropriate parameter group. Third, a parameter change alone does not show whether the face has moved closer to the reference; the next edit must depend on the updated render.

Learning these edits also requires supervision in the native parameter space. Existing face resources commonly pair images with meshes or coefficients of research face models, whereas a native editor needs paired editable MetaHuman states and rendered images. Converting source faces into MetaHumans can also compress differences among identities. A scalable data pipeline must therefore check identity preservation and sample diversity in the converted DNA space.

We present MetaHuman-Anything, a VLM-guided framework that automatically edits native MetaHuman face parameters to match a reference portrait. A frozen Reasoner compares the reference with the current preview and describes their visual differences. Ten learned queries then pool group-indexed reasoning representations from the answer's hidden states. A Grounder combines the reference image, current native state, and reasoning features to predict parameter targets and update gates for each group. A frozen surrogate renderer produces the next preview from the updated state, allowing the Reasoner to propose another correction. The final state remains a native MetaHuman asset that can be edited and rendered in Unreal Engine.

To provide native supervision, we develop an automated pipeline from the Google Generative aNthropometric Model (GNM) (Ploumpis et al., 2026) to MetaHuman. It generates diverse source faces, converts them in batches into MetaHuman Character and DNA assets, filters identities in the converted native space, and pairs editable states with engine renders. The resulting corpus contains 2,000 identities and 15,000 rendered images.

We evaluate identity preservation using final Unreal Engine renders. On 102 external portraits under matched fixed-appearance conditions, the selected editor improves mean facial identity similarity by 118.2% relative to the strongest reconstruction-then-conversion baseline. This improvement in the final native Character directly addresses the conversion gap that motivates our approach.

Contributions. We make three contributions:

- **Visual reasoning as native MetaHuman edits.** Our VLM-guided editor combines visual differences with the current native state to predict grouped parameter targets and update gates, then uses rendered feedback to refine the character.
- **Scalable native supervision.** Our automated GNM-to-MetaHuman pipeline filters identities in the converted native space and pairs editable states with engine renders, providing supervision for native face editing.

- **Identity preservation in final engine assets.** We evaluate the faces rendered by Unreal Engine and obtain higher identity similarity than reconstruction-then-conversion baselines under controlled appearance conditions.

2 Related Work

Parametric face reconstruction. FLAME provides articulated identity and expression parameters (Li et al., 2017); RingNet, DECA, EMOCA, and MICA infer facial shape or appearance under different supervision assumptions (Sanyal et al., 2019; Feng et al., 2021; Daněček et al., 2022; Zielonka et al., 2022). FaceVerse and implicit morphable models extend representational detail and capacity (Wang et al., 2022; Zheng et al., 2022b). These approaches motivate explicit parameter supervision. Our output coordinates are native MetaHuman identity and appearance controls, conditioned on an existing state. Mapping outputs from another face model to this space introduces a separate conversion problem.

Photo-to-game-face authoring. Face-to-Parameter Translation optimizes editable game-character facial controls through a learned imitator of a nondifferentiable engine (Shi et al., 2019). Its follow-up learns a faster, single-pass parameter translator (Shi et al., 2020). MeInGame transfers reconstructed shape and texture into game character faces (Lin et al., 2021). These studies establish automated photo-to-game-face authoring. Our framework targets MetaHuman Character/DNA states and learns state-conditioned, groupwise edit targets and gates from reference images and Reasoner outputs.

Animatable avatars and production compatibility. Neural avatars combine learned appearance with deformation models (Grassal et al., 2022; Zheng et al., 2022a; Zielonka et al., 2023), while GaussianAvatars binds rendering primitives to a parametric face (Qian et al., 2024). Recent single-image systems further reduce acquisition requirements (Qiu et al., 2025; Kirschstein et al., 2025; Liao et al., 2025). Animation capability and native-engine compatibility measure different properties. We therefore evaluate parameter recovery, rendered resemblance, and successful construction of a renderable MetaHuman asset separately.

Multimodal editing. Instruct-NeRF2NeRF uses language-guided image editing to update a neural scene (Haque et al., 2023); HeadSculpt and TADA explore controllable head or avatar generation (Han et al., 2023; Liao et al., 2024). Our editor compares a reference with its current render and converts that comparison into native parameter actions.

Engine-native authoring and parameter corpora. Production engines ship authoring tools that fit or sculpt MetaHuman assets from scans and meshes; these tools operate in an offline artist loop. Our editor complements them as an automated post-process, applying iterative reference-guided adjustment directly to the same native assets. Parameter-driven corpora such as FaceVerse (Wang et al., 2022) pair rendered images with the parameters of a research face model; neither their renders nor their records are bound to a production asset graph. The corpus in Section 4 links every image to the editable Character, FaceState, and DNA records of a live MetaHuman asset, so supervision and evaluation transfer directly to engine operations.

3 Method

3.1 Task and Native State

Given a reference portrait I^* and an initial editable MetaHuman state S_0, our goal is to produce a native state whose rendered face approaches the portrait in identity and visible appearance. At iteration t, we represent the state as

$$S_t = (\mathbf{x}_t, \mathbf{a}_t, \mathbf{c}_t, \gamma_t), \tag{1}$$

where $\mathbf{x}_t$ contains 1,397 FaceModel coefficients, $\mathbf{a}_t$ contains 15 continuous appearance controls, $\mathbf{c}_t$ contains nine categorical appearance fields, and γ_t is the gender field. Of the coefficients, 1,345 vary in the training split; the other 52 are restored as train-set constants when writing the complete

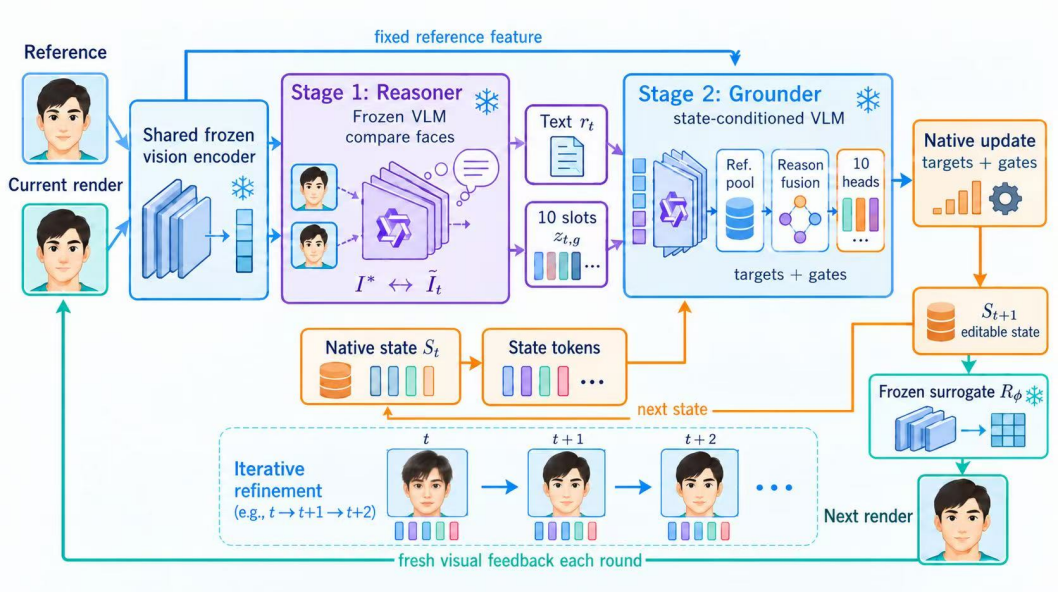

Figure 2: MetaHuman-Anything's closed loop. A frozen Reasoner compares the reference with the current render. A state-conditioned Grounder uses trainable adapters and heads on a frozen VLM backbone to predict grouped native targets and gates. The frozen surrogate renders the updated state; both state and preview enter the next iteration.

native state. The ten prediction groups are facial identity coefficients, skin, freckles, iris, eyelashes, textures, hair, eyebrows, facial hair, and gender. These groups organize output heads: continuous controls receive group-level updates, whereas categorical fields retain individual execution gates.

A frozen surrogate renderer R_ϕ maps a state to a fixed-view preview $\widetilde{I}_t = R_\phi(S_t)$. At each iteration, the editor compares $(I^*, \widetilde{I}_t)$, updates the native state, and renders the new state. Both S_{t+1} and $\widetilde{I}_{t+1}$ enter the next iteration, while I^* remains fixed. After the final iteration, the parameters are written to a MetaHuman Character and rendered in Unreal Engine for evaluation. Figure 2 summarizes these two feedback paths; its illustrated faces and iterations are schematic.

3.2 Constructing the Native Face-State Corpus

We call our paired Unreal Engine renders and editable MetaHuman states the *native face-state corpus*. We construct it from the Google Generative aNthropometric Model (GNM) head model Ploumpis et al. (2026) through a staged conversion and selection pipeline. GNM identity parameters first produce source head meshes. Batch MetaHuman conform maps each mesh to a Character and DNA representation, from which we export the native FaceModel coefficients and render the resulting face. An identity identifier links source geometry, converted DNA, editable coefficients, appearance settings, and rendered images. The editor outputs continuous coefficients for an unseen portrait; the identifier serves as a data index.

Conform can compress distinctions among source faces. We therefore train a lightweight mapping from GNM identity features to predicted DNA features and use source and predicted-native features together to preselect candidates before costly conversion. A low-cost Posed-DNA pass supplies actual converted geometry for further screening. Candidates retained for full Character and State-DNA export are then assessed in the converted space using rendered-face identity features and native geometry descriptors. Final selection and auditing use the post-conversion renders, State-DNA geometry, and identity consistency across frontal and $\pm 30^\circ$ views. This procedure yields 2,000 engine-valid, gender-balanced native identities with unique DNA assets.

We next vary appearance without changing the selected base identity DNA. Each identity receives seven or eight controlled appearance configurations, covering continuous controls and categorical

assets. Every rendered sample stores its parent identity, native coefficient vector, appearance configuration, and image. This produces a corpus of 15,000 state–render pairs. Section 4 gives its identity-disjoint split and evaluation protocol; the conversion and post-DNA audits establish which native states enter that protocol.

3.3 Visual Reasoning and State-Conditioned Grounding

The editor makes two passes over a shared Qwen3-VL-8B-Instruct backbone Bai et al. (2025). In the Reasoner pass, language adapters are disabled. The frozen model views the reference image and current preview, then generates a comparison r_t describing their remaining visual difference. Ten learned queries pool its generated answer-token hidden states into group-indexed reasoning slots $\mathbf{z}_{t,g}$. Both the text and slots are refreshed after every update.

In the Grounder pass, trainable Q/K/V/O LoRA adapters are active. Its backbone input contains the reference image I^*, 16 encoded tokens for the current native state S_t, an identity token projected from a frozen ArcFace (Deng et al., 2019) embedding of I^*, and the Reasoner text r_t. The reasoning slots condition the group heads through residual projections. Thus the reference image anchors the target, the state tokens describe the current editable asset, and evidence from the current preview reaches the Grounder through the Reasoner outputs.

The Grounder is initialized from a static, single-pass reference-to-state predictor (the static Grounder). Ten learned visual queries pool reference-image tokens; their group-specific outputs combine with the <REG> hidden state to form $\mathbf{b}_{t,g}$ for group g. A projection of <REASON_REG> provides a global text vector $\mathbf{u}_t$. Each parameter head receives

$$\mathbf{h}_{t,g} = \mathbf{b}_{t,g} + f_g^{\text{text}}(\mathbf{u}_t) + f_g^{\text{slot}}(\mathbf{z}_{t,g}), \tag{2}$$

where f_g^{text} and f_g^{slot} are learned group-specific residual projections. Their final layers are initialized to zero, so the iterative model initially reproduces the static Grounder's proposal and learns how reasoning changes that proposal. The ten slots and heads share an index order; their attention weights and projections are learned from native-state supervision.

3.4 Native Targets, Gates, and Rendering Feedback

The group heads predict continuous targets and deltas, categorical targets, and change gates. Delta outputs support consistency supervision; execution uses the target and gate. For the coefficient group and each of four continuous appearance groups, $\alpha_{t,g} = \sigma(\ell_{t,g})$ controls movement from the current normalized values toward the predicted target:

$$\mathbf{s}_{t+1,g} = \text{clip}_{[-6,6]}\left(\mathbf{s}_{t,g} + \alpha_{t,g}(\widehat{\mathbf{s}}^*_{t,g} - \mathbf{s}_{t,g})\right). \tag{3}$$

The clip applies in normalized continuous coordinates. For a categorical field or gender, its individual gate selects the predicted category when the gate probability is at least 0.5; otherwise the current value is retained. A training threshold used to label continuous gates is distinct from this discrete execution threshold. The field assignments, valid categories, and normalization are detailed in Appendix A.1.

After execution, $R_\phi(S_{t+1})$ supplies the next preview while S_{t+1} becomes the next numerical state input. The Reasoner recomputes the comparison against I^*, so each action depends on the previous action's rendered effect. This loop uses a frozen approximation of the fixed-view Unreal Engine renderer. The external identity score instead uses an Unreal Engine render of the final native Character.

3.5 Training the Renderer and Editor

We first train the surrogate renderer on paired native states and Unreal Engine images. Its objective combines pixel reconstruction, structural similarity, perceptual similarity, and face-identity preservation:

$$\mathcal{L}_{\text{render}} = \mathcal{L}_1 + 0.2(1 - \text{SSIM}) + 0.1\mathcal{L}_{\text{LPIPS-VGG}} + 0.1\mathcal{L}_{\text{ArcFace}}. \tag{4}$$

The identity term has a warm-up, and the validation-selected renderer is frozen before editor training. Its architecture and held-out renderer audit are reported in Appendix A.2.

Editor training proceeds in three stages after static-Grounder pretraining. First, an interface stage trains the reasoning projections and slots while the static Grounder is fixed. Second, joint adaptation updates the reasoning projections, group heads, gates, and Grounder LoRA adapters. Third, rollout replay adds states reached by the editor's own predictions, allowing training on the inputs encountered after an imperfect update. The zero-initialized reasoning residuals keep the static proposal available at the start of adaptation.

Native supervision covers identity coefficients, continuous appearance values, categorical assets, gender, absolute targets, and executed next states. Additional terms enforce consistency of predicted deltas and identity, train the update gates, and compare frozen-surrogate renders of both predicted target and executed next states with the desired image. The categorical image path uses a straight-through approximation. Identity-only Flickr-Faces-HQ (FFHQ) portraits Karras et al. (2019) add a target-render identity objective while contributing no native-state labels. At inference, each round refreshes both the Reasoner output and surrogate preview before applying the next update. Loss weights, sampling mixtures, optimization settings, and rollout schedule are specified in Appendix A.3.

4 Data and Evaluation Protocol

The native face-state corpus contains 15,000 fixed-view images from 2,000 identities, with seven or eight appearance variants per identity. Its identity-disjoint training, validation, and test splits contain 10,497/1,494/3,009 images from 1,400/200/400 identities, respectively. Original 512×512 RGBA images are resized to 256×256 RGB. Identity statistics, appearance statistics, and categorical vocabularies are computed from training records only. The frozen parameter-to-image renderer maps native states to the image domain used for closed-loop training and internal evaluation.

Static-Grounder and iterative-editor training sample different-identity, same-identity, and real-identity-interpolation pairs in a 6:1:3 ratio. Validation uses frozen different-identity pairs, with each target paired with a current state from another identity; all models share these pair artifacts. Iterative evaluation records updates 1, 2, 4, and 8 from each fixed initial state. The one-shot baseline predicts a target state in one pass.

For a state S, standardized active-identity and continuous-appearance errors are

$$e_x(S) = \frac{\|\boldsymbol{z}_x - \boldsymbol{z}_x^*\|_1}{1345}, \qquad e_a(S) = \frac{\|\boldsymbol{z}_a - \boldsymbol{z}_a^*\|_1}{15}, \qquad e_{\text{comb}}(S) = \frac{e_x(S) + e_a(S)}{2}. \tag{5}$$

The iterative parameter-selection score is the mean next-state e_{comb} over rollout lengths 1, 2, 4, and 8 on the fixed different-identity validation pairs; lower is better. We report the corresponding proxy-image score separately. For identity evaluation, we extract ArcFace embeddings from a target portrait and a rendered face, then compute their cosine similarity (ID-Sim; higher is better). We distinguish surrogate-preview ID-Sim used internally from Unreal-Engine-render ID-Sim (UE ID-Sim) used for final-asset evaluation. The one-shot baseline uses its one-step score.

We sample portraits from FFHQ for identity-only training and form separate validation and test subsets. A 10,000-image cache supplies optional training examples. The validation subset contains 100 portraits, each paired with four fixed initial native states (400 trajectories), and supports iterative-editor checkpoint selection by surrogate-render identity similarity. The test subset remains outside checkpoint selection.

For the native-engine comparison, we use the Face Research Lab London Set (DeBruine & Jones, 2017). We take one neutral frontal photograph for each of its 102 identities from the archive's neutral-front subset, retaining the source portrait identifiers. Every method uses the same Unreal Engine camera, lighting, and fixed appearance template. UE ID-Sim is measured on final Character renders with a frozen identity evaluator separate from the one used during training. A supplementary 100-portrait FFHQ UE comparison appears in Appendix B.2.

The static Grounder is selected by native-corpus parameter error with a 1% tolerance, followed by native-corpus and FFHQ validation identity similarity. The iterative editor is selected by its four-step identity similarity on the FFHQ validation subset. The London portraits provide the external Unreal Engine comparison.

5 EXPERIMENTS

5.1 COMPARISON WITH RECONSTRUCTION AND CONVERSION

We compare final editable assets on 102 neutral-front London portraits, one per identity. 3DDFA-V2 (Guo et al., 2020), DECA, and MICA reconstruct a face and then automatically convert it to a native MetaHuman DNA/Character asset. Our iterative editor writes native facial coefficients directly. Unreal Engine 5.8 renders every asset with the same camera, lighting, and fixed light-skin, hairless female appearance. We measure UE ID-Sim between each target portrait and its final Character render. Holding appearance fixed focuses the comparison on facial identity after the complete reconstruction-to-engine or native-editing pipeline.

Table 1: London external test (102 identities). Mean UE ID-Sim ↑ under the same fixed appearance. The selected editor was chosen on FFHQ identity validation before its separate London UE run.

Method	UE ID-Sim ↑
3DDFA-V2 → MetaHuman	0.0171
DECA → MetaHuman	0.0163
MICA → MetaHuman	0.0747
Ours, initial iterative editor	0.1286
Ours, selected iterative editor	**0.1631**

The selected editor reaches 0.1631 versus 0.0747 for MICA, the strongest conversion baseline (Table 1). The paired mean gain is 0.0883 from unrounded scores, a 118.2% relative increase; a 10,000-resample identity bootstrap gives a 95% interval of $[0.0734, 0.1038]$. Our editor scores higher than MICA for 91 of 102 portraits. Because the comparison scores the rendered MetaHuman Characters rather than intermediate reconstructions, this gain directly supports native editing as a way to preserve identity in the asset delivered to the engine. The initial editor already reaches 0.1286, showing that the advantage is present before identity-focused replay.

We also evaluate closed-set identity retrieval using the same final UE renders. For each method, we rank its 102 fixed-appearance renders for each London portrait by ArcFace cosine similarity. Our selected editor retrieves the correct identity at rank 1 for 66 of 102 portraits (64.7%), compared with 18 (17.6%) for MICA-to-MetaHuman. The corresponding mean similarities for our editor are 0.1631 for matched identities and 0.0099 for mismatched identities. The final native assets therefore retain more target-identity information under this fixed-appearance protocol. Appendix B.1 reports all six evaluated conditions.

Figure 3 illustrates native MetaHuman appearance editing with author-configured skin and hair. Table 1 evaluates all methods under the same fixed appearance. The numbers in Figures 1 and 3 are London portrait identifiers.

5.2 ABLATION STUDIES

We ablate the static Grounder's language adapters and group attention against the full static-Grounder configuration, which uses identity input, FFHQ sampling probability 0.10, and target-render identity-loss weight 0.05. All three configurations use the same checkpoint-selection validation protocol, with one training run per configuration. A separate matched identity-input comparison at higher FFHQ sampling appears in Appendix B.4.

Table 2: Structural ablations of the full static Grounder on native face-state corpus validation (1,494 fixed pairs). P-Err is mean normalized parameter error e_{comb} (Eq. 5); I-Loss is surrogate-render $L_1 + 0.2(1 - \text{SSIM}) + 0.1\text{LPIPS}$. Lower is better; bold marks the best recorded value.

Grounder variant	P-Err ↓	I-Loss ↓
Full static Grounder	**0.4554**	**0.0537**
Without language adapters	0.4583	0.0563
Without group attention	0.4772	0.0650

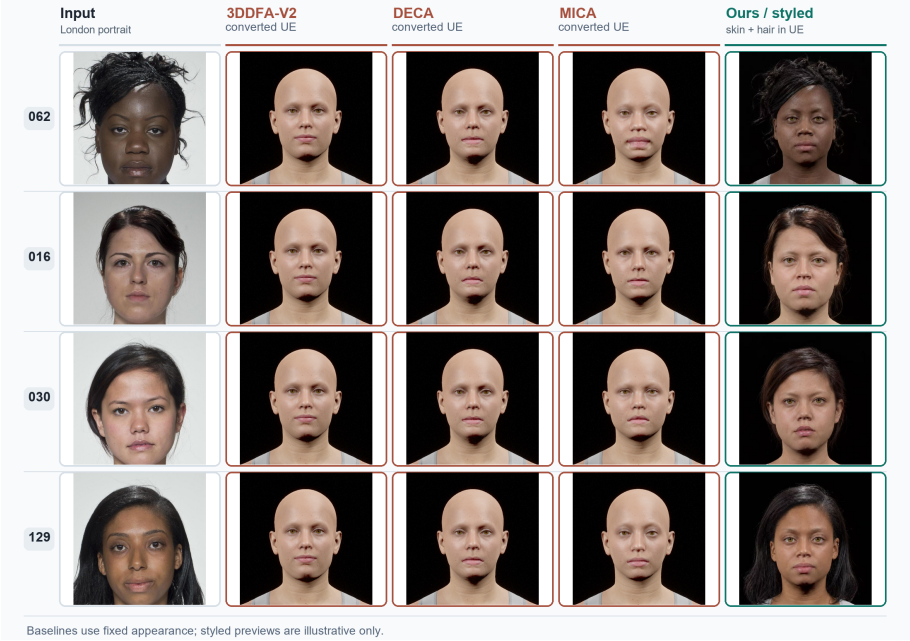

Figure 3: Four London portraits (062, 016, 030, and 129). Reconstruction baselines show converted UE assets under fixed appearance; our native assets show illustrative skin and hair styling. Quantitative UE ID-Sim uses fixed appearance for every method (Table 1).

The full static Grounder has the lowest parameter error and image loss (Table 2). Removing language adapters raises P-Err by 0.0029 and I-Loss by 0.0026; removing group attention raises them by 0.0218 and 0.0113. Both components therefore contribute to parameter recovery and render agreement in these runs, with the larger degradation following removal of group attention. These improvements address the native-editing objective: converting portrait evidence into a state whose rendered face matches the target.

5.3 Iterative Refinement and Identity Analysis

We compare fresh, absent, and shuffled Reasoner outputs on the selected editor, using the same 102 London portraits, eight update rounds, native initialization, and fixed-appearance UE rendering. The trained weights remain fixed, and all conditions use the learned gates; Appendix B.3 details the inference protocols.

Fresh reasoning reaches 0.16307 UE ID-Sim, compared with 0.11514 without reasoning: a paired gain of 0.04793 (95% bootstrap interval $[0.03816, 0.05792]$), or a 41.6% relative increase. It improves 85 of 102 portraits. Supplying Reasoner output thus improves the selected editor's final UE identity similarity under the fixed-appearance protocol.

Shuffled reasoning reaches 0.16155. Fresh reasoning has a higher observed mean by 0.00153 (95% interval $[-0.00022, 0.00315]$) and scores higher on 64 of 102 portraits. The supported inference is that supplying Reasoner output improves performance; the additional effect of pairing that output with its own portrait remains unresolved.

Separately, increasing the initial editor from one to eight rounds improves UE ID-Sim from 0.12714 to 0.12860 (paired gain 0.00145; 95% interval $[0.00044, 0.00246]$), providing evidence for the benefit of continued native editing.

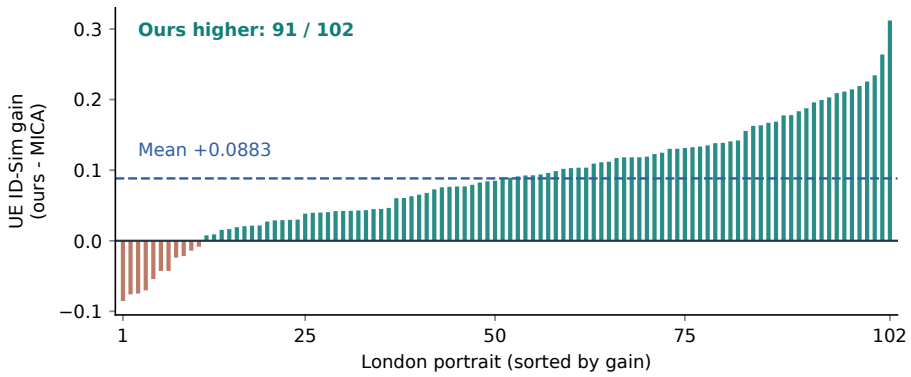

Figure 4: Per-portrait UE ID-Sim gain on London 102: selected editor minus MICA. Each bar represents one identity; positive values favor our editor. Portraits are sorted by gain, not by training step. The dashed line marks the mean gain, $+0.0883$.

The selected editor scores higher than MICA for 91 of 102 portraits. Its median paired gain is 0.0893, close to the mean of 0.0883. The identity benefit in final UE renders therefore extends across most portraits rather than being driven by a few large gains, reinforcing the advantage of optimizing the native asset directly.

Identity-focused training raises FFHQ validation ID-Sim from 0.1285 to 0.1839 across selected runs. This comparison measures the combined change in identity sampling, loss weighting, and replay training; the inference controls above hold each tested checkpoint fixed. Tables 6 and 7 provide the training-checkpoint results.

6 Limitations and Responsible Use

The London comparison measures facial identity from frontal UE renders under a fixed appearance template, separating face matching from hair and skin selection. We verified Character construction, saving, and rerendering. Multi-view appearance and animation are separate evaluation targets.

Identifiable digital humans can enable impersonation and non-consensual avatar creation. Deployment requires consent, provenance, access control, disclosure, and compliance with source-image and engine licenses.

7 Conclusion

MetaHuman-Anything turns visual comparisons between a reference portrait and the current render into gated edits of native MetaHuman face parameters. An automated GNM-to-MetaHuman pipeline supplies paired editable states and engine renders for training. On 102 external London portraits, the selected editor reaches 0.1631 UE ID-Sim versus 0.0747 for the strongest reconstruction-and-conversion baseline under the same fixed appearance. These results show that optimizing directly in the native parameter space preserves more facial identity in the finished Unreal Engine asset.

Reproducibility Statement

Sections 4–5 specify the evaluation protocol; Appendices A–B provide model configurations, checkpoints, and supplementary results.

AI Use Statement

ClawsGO Science Agent and Codex assisted with manuscript drafting and editing, LaTeX preparation, figure layout, and cross-checking recorded experiment values against available reports and

code. Reported measurements come from the recorded experiments. The authors are responsible for the scientific claims, final manuscript, and this disclosure.

REFERENCES

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, et al. Qwen3-VL technical report. *arXiv preprint arXiv:2511.21631*, 2025. doi: 10.48550/arXiv.2511.21631.

Radek Daněček, Michael J. Black, and Timo Bolkart. Emoca: Emotion driven monocular face capture and animation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20311–20322, 2022.

Lisa DeBruine and Benedict Jones. Face research lab london set. figshare dataset, 2017. URL https://doi.org/10.6084/m9.figshare.5047666.

Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4690–4699, 2019.

Epic Games. Mesh to metahuman. https://dev.epicgames.com/documentation/metahuman/mesh-to-metahuman, n.d. Accessed September 2026.

Yao Feng, Haiwen Feng, Michael J. Black, and Timo Bolkart. Learning an animatable detailed 3D face model from in-the-wild images. *ACM Transactions on Graphics*, 40(4):88:1–88:13, 2021. doi: 10.1145/3450626.3459936.

Philipp-Wenzel Grassal, Malte Prinzler, Titus Leistner, Carsten Rother, Matthias NieSSner, and Justus Thies. Neural head avatars from monocular RGB videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18632–18643, 2022.

Jianzhu Guo, Xiangyu Zhu, Yang Yang, Fan Yang, Zhen Lei, and Stan Z. Li. Towards fast, accurate and stable 3d dense face alignment. In *Computer Vision – ECCV 2020*, pp. 152–168. Springer International Publishing, 2020. doi: 10.1007/978-3-030-58529-7_10.

Xiao Han, Yukang Cao, Kai Han, Xiatian Zhu, Jiankang Deng, Yi-Zhe Song, Tao Xiang, and Kwan-Yee K. Wong. Headsculpt: Crafting 3d head avatars with text. In *Advances in Neural Information Processing Systems*, 2023.

Ayaan Haque, Matthew Tancik, Alexei A. Efros, Aleksander Holynski, and Angjoo Kanazawa. Instruct-nerf2nerf: Editing 3d scenes with instructions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 19740–19750, 2023.

Alexander Huang-Menders, Xinhang Liu, Andy Xu, Yuyao Zhang, Chi-Keung Tang, and Yu-Wing Tai. SmartAvatar: Text- and image-guided human avatar generation with VLM AI agents. *arXiv preprint arXiv:2506.04606*, 2025. doi: 10.48550/arXiv.2506.04606.

Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4401–4410, 2019.

Tobias Kirschstein, Javier Romero, Artem Sevastopolsky, Matthias Nießner, and Shunsuke Saito. Avat3r: Large animatable gaussian reconstruction model for high-fidelity 3d head avatars. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025.

Tianye Li, Timo Bolkart, Michael J. Black, Hao Li, and Javier Romero. Learning a model of facial shape and expression from 4D scans. *ACM Transactions on Graphics*, 36(6):194:1–194:17, 2017. doi: 10.1145/3130800.3130813.

Tingting Liao, Hongwei Yi, Yuliang Xiu, Jiaxiang Tang, Yangyi Huang, Justus Thies, and Michael J. Black. Tada! text to animatable digital avatars. In *International Conference on 3D Vision*, pp. 1508–1519, 2024.

Tingting Liao, Yujian Zheng, Adilbek Karmanov, Liwen Hu, Leyang Jin, Yuliang Xiu, and Hao Li. Soap: Style-omniscient animatable portraits. In *ACM SIGGRAPH Conference Papers*, 2025. doi: 10.1145/3721238.3730691.

Jiangke Lin, Yi Yuan, and Zhengxia Zou. MeInGame: Create a game character face from a single portrait. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.

Stylianos Ploumpis, Jan Bednarik, Gaspard Zoss, Ruslan Guseinov, et al. GNM head: A generative anthropometric model of the human head. *arXiv preprint arXiv:2607.23687*, 2026. doi: 10.48550/arXiv.2607.23687.

Shenhan Qian, Tobias Kirschstein, Liam Schoneveld, Davide Davoli, Simon Giebenhain, and Matthias Nießner. GaussianAvatars: Photorealistic head avatars with rigged 3D gaussians. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20299–20309, 2024.

Lingteng Qiu, Xiaodong Gu, Peihao Li, Qi Zuo, Weichao Shen, Junfei Zhang, Kejie Qiu, Weihao Yuan, Guanying Chen, Zilong Dong, and Liefeng Bo. Lhm: Large animatable human reconstruction model from a single image in seconds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025.

Soubhik Sanyal, Timo Bolkart, Haiwen Feng, and Michael J. Black. Learning to regress 3D face shape and expression from an image without 3D supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7763–7772, 2019.

Tianyang Shi, Yi Yuan, Changjie Fan, Zhengxia Zou, Zhenwei Shi, and Yong Liu. Face-to-parameter translation for game character auto-creation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019.

Tianyang Shi, Zhengxia Zou, Yi Yuan, and Changjie Fan. Fast and robust face-to-parameter translation for game character auto-creation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020. doi: 10.1609/aaai.v34i02.5537.

Lizhen Wang, Zhaoqi Chen, Tao Yu, Chenguang Ma, Jing Li, and Yebin Liu. Faceverse: A fine-grained and detail-controllable 3D face morphable model from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20333–20342, 2022.

Yufeng Zheng, Victoria Fernández Abrevaya, Marcel C. Bühler, Xu Chen, Michael J. Black, and Otmar Hilliges. IMavatar: Implicit morphable head avatars from videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13545–13555, 2022a.

Yufeng Zheng, Ruiping Yang, Yao Huang, and Yizhou Chen. Imface: A nonlinear 3d morphable face model with implicit neural fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20343–20352, 2022b.

Wojciech Zielonka, Timo Bolkart, and Justus Thies. Towards metrical reconstruction of human faces. In *European Conference on Computer Vision*, pp. 250–269. Springer, 2022. doi: 10.1007/978-3-031-19778-9_15.

Wojciech Zielonka, Timo Bolkart, and Justus Thies. Instant volumetric head avatars. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4574–4584, 2023.

A Implementation and Reproducibility Details

A.1 State fields and image preprocessing

The native state has 1,397 facial coefficients, of which 1,345 vary in the training split and 52 are train-constant, 15 continuous appearance fields, nine categorical appearance fields, and gender. The ten group indices, in order, are `identity_coefficients`, `skin`, `freckles`, `iris`, `eyelashes`, `textures`, `hair`, `eyebrows`, `facial_hair`, and `gender`. The continuous assignments are skin (`skin_u`, `skin_v`, `skin_roughness`), freckles (`freckles_density`,

freckles_strength), iris (iris_rotation, iris_primary_u/v, iris_secondary_u/v, iris_color_blend, pupil_dilation), and eyelashes (eyelashes_melanin, eyelashes_redness, eyelashes_roughness). The nine categorical fields are face and body texture indices, freckles mask, iris pattern, eyelashes type, hair asset, eyebrows asset, mustache asset, and beard asset. The identity-coefficient group and four appearance groups therefore receive continuous gates; each categorical field and gender receive individual gates. The term ten groups refers to representation and output-head order, not ten identical scalar gates.

The native face-state corpus stores 512×512 RGBA renders. RGB copies are resized to 256×256 with Lanczos sampling and mapped to $[-1, 1]$ for the renderer. Numeric normalization and categorical vocabularies are derived from the 1,400-identity training split. Continuous execution is clipped to $[-6, 6]$ normalized units. For training pairs, the probabilities of different-identity, same-identity, and real-identity-interpolation examples are 0.6, 0.1, and 0.3. Interpolation uses one, two, or three donors with probabilities 0.5, 0.25, and 0.25 and samples small, medium, and large interpolation strengths from $[0.1, 0.3]$, $[0.3, 0.6]$, and $[0.6, 1.0]$. Validation and test each use fixed, different-identity pair manifests.

A.2 Frozen surrogate renderer and held-out audit

The surrogate renderer separately encodes coefficients and continuous appearance with MLPs and categorical appearance and gender with embeddings, projects fused features to a $4 \times 4 \times 512$ map, and uses residual upsampling to produce a 256×256 RGB image with a $\tanh$ output. Its objective is pixel L_1 (weight 1), $1 - \mathrm{SSIM}$ (0.2), LPIPS-VGG (0.1), and identity loss (0.1; 1,000-step warm-up). The frozen validation-selected step-93,808 checkpoint has SHA256 bd6a10f80485705b134bc6bb925f2b945e81ebdaf130f66434dd88ac4db2ac51. On 1,494 validation images it achieves L_1 0.02042, SSIM 0.92476, LPIPS-VGG 0.05523, and ID-Sim 0.89515. On the separate 3,009-image native face-state corpus test split, a renderer-versus-UE audit records L_1 0.02027, SSIM 0.92554, and LPIPS-VGG 0.05482. The stable image-reconstruction scores support using this frozen renderer for closed-loop training.

A.3 Grounder inputs, heads, and optimization

The selected iterative editor uses Qwen3-VL-8B-Instruct in BF16, with frozen base visual and language weights and language-attention LoRA on q_proj, k_proj, v_proj, and o_proj (rank 16, alpha 32, dropout 0.05). The Reasoner uses the same base with adapters disabled, greedy decoding, and at most 96 generated answer tokens. The Grounder has 16 state tokens (state encoder width 1,024; token dimension 256; categorical embedding dimension 16), one target-image ArcFace token (512-dimensional normalized embedding projected through a trainable MLP to the backbone width), and ten learned visual and ten learned answer-token group queries, each with four-head attention. The identity extractor is frozen. The effective selected configuration sets head.direct_state_fusion=false; no exact native-state vector bypasses the state-token path into the output head. The identity projection uses hidden width 1,024, dropout 0.1, and residual scale 0.02. Its text and slot residuals have zero-initialized final linear layers.

The supervised loss weights for coefficient, continuous, categorical, gender, absolute-target, and next-state terms are one each; delta and identity consistency are 0.1 each, image 0.1, and gate 0.2. Continuous gate supervision regards a normalized target change greater than 0.02 as active and warms up over 200 steps. The renderer image loss evaluates both predicted target and executed next state (fractions 0.5 and 0.5), using a straight-through discrete estimator at temperature one. The selected identity-focused replay uses an FFHQ identity-only batch probability 0.5 and target-render identity-loss weight 0.5, warmed up over 500 steps. Identity-only examples do not provide supervised target native coefficients. The static Grounder and initial iterative editor used probability 0.1 and identity-loss weight 0.05.

The full static Grounder trains for 35,000 steps on two devices at batch size 16 per device; step 33,600 is selected. Iterative-editor training first trains a reasoning interface while evaluating the pretrained static Grounder without gradients, then jointly adapts reasoning projections, heads, gates, and LoRA, and then trains with rollout replay. The selected replay has seed 42, four devices, batch size eight per device, 20 epochs/7,000 steps, gradient checkpointing, and AdamW ($\beta_1 = 0.9$,

$\beta_2 = 0.999$, weight decay 10^{-4}). Its learning rates are 10^{-4} for reasoning modules, 2×10^{-5} for heads, and 10^{-5} for LoRA. Replay refreshes every 250 steps, holds at most 128 cases, uses at most two generations, and is sampled with probability 0.5. Step 7,000 is selected on FFHQ identity validation and evaluated separately on the London portraits. The selected checkpoint SHA256 is `dd52924c4d38c03f65ddeaf3128829bdf9ef0d806508c48ada4f3f9c8b7b79e3`.

Checkpoint selection used surrogate-renderer step 93,808 and one-shot-regressor step 17,500; the latter uses an ImageNet-initialized ResNet-50. The full static Grounder used identity input, FFHQ sampling probability 0.10, and target-render identity-loss weight 0.05; it initialized the iterative editor. The initial iterative-editor replay stopped at step 3,840, with step 2,800 selected for the initial-editor comparison. A separate replay completed 7,000 steps with FFHQ sampling probability 0.50 and identity-loss weight 0.50; its step-7,000 checkpoint was selected by FFHQ identity validation for the later evaluation. Native-corpus validation parameter error was 0.4534 at step 2,800 and 0.4544 at step 7,000 (Table 6).

A.4 One-shot baseline and native conversion

The one-shot regressor is an ImageNet-initialized ResNet-50 with heads for native coefficients, continuous appearance, nine categorical fields, and gender. It sees the target portrait alone and outputs one state. Its state loss combines Smooth L_1 coefficient and continuous errors with nine categorical cross-entropies and gender supervision. Coefficient consistency and a frozen surrogate-renderer image L_1 each have weight 0.1. The image loss renders predicted continuous, categorical, and gender values, with straight-through hard-forward discrete values; it does not use ground-truth appearance values. Mild affine augmentation is applied to inputs while image-loss targets remain unaugmented.

For the London comparison, 3DDFA-V2, DECA, and MICA meshes are converted into native MetaHuman DNA/Character assets; the iterative editor writes native coefficients directly. Each method is rendered by UE 5.8 under the same camera, lighting, and fixed light-skin, hairless female appearance. UE ID-Sim is computed from the target portrait and final Character render. Baseline conversion, asset creation, and UE rendering complete for all 102 London samples. The initial and selected iterative-editor checkpoints have separate complete UE runs; the selected run produces 102/102 parameter exports, Character assets, and renders.

Initial-editor inference averaged 16.19 seconds per London portrait, followed by 1.98 seconds to write native coefficients. UE conform took approximately 45 seconds per converted baseline asset; upstream baseline inference was not timed under a common end-to-end protocol.

B Supplementary Evaluation

B.1 London closed-set identity retrieval

For each method, every London portrait queries the gallery of that method's 102 fixed-appearance UE renders. We rank each gallery by the same ArcFace cosine similarity used for UE ID-Sim in Section 5. Matched and mismatched means respectively summarize the 102 corresponding portrait–render pairs and the 10,302 noncorresponding pairs within each method. Table 3 gives the complete retrieval results. All 102 portraits and 612 method-specific UE renders yielded face embeddings.

Table 3: London closed-set identity retrieval with 102 portraits and one fixed-appearance UE render per identity in each method-specific gallery. Matched and mismatched columns give mean target-to-render ArcFace cosine similarity; Top-1 is the fraction of queries for which the corresponding render ranks first. All entries reuse the UE renders and evaluator of the main London comparison.

UE asset	Matched ↑	Mismatched	Top-1 ↑
3DDFA-V2 → MetaHuman	0.0171	0.0148	1/102 (1.0%)
DECA → MetaHuman	0.0163	0.0094	2/102 (2.0%)
MICA → MetaHuman	0.0747	0.0122	18/102 (17.6%)
Selected editor, fresh reasoning	**0.1631**	0.0099	**66/102 (64.7%)**
Selected editor, no reasoning	0.1151	0.0048	62/102 (60.8%)
Selected editor, shuffled reasoning	0.1615	0.0096	64/102 (62.7%)

The selected editor places the correct UE render first for 66 portraits, compared with 18 for the strongest reconstruction-and-conversion baseline. Its matched mean is also higher than its mismatched mean (0.1631 versus 0.0099), showing that the final assets carry discriminative identity information in this cross-domain retrieval task. The 102-person gallery defines the retrieval setting; these rates are not open-set verification accuracy.

B.2 Exploratory FFHQ UE comparison

The supplementary UE comparison uses 100 FFHQ portraits under the same fixed-appearance template and UE ID-Sim protocol. Six portraits also occur in the training cache. This run compares the initial editor with the three conversion baselines.

Table 4: Supplementary FFHQ UE comparison (100 portraits). Mean UE ID-Sim ↑ under fixed appearance; the best recorded score is bold.

Method	UE ID-Sim ↑
3DDFA-V2 → MetaHuman	-0.0015
DECA → MetaHuman	0.0102
MICA → MetaHuman	0.0685
Ours, initial iterative editor	**0.1166**

The initial editor scores 0.1166 versus 0.0685 for MICA. The same ranking on these FFHQ portraits adds evidence that editing native parameters preserves more identity in the final engine render than reconstruction followed by conversion.

B.3 Inference controls on final native assets

Selected-editor reasoning controls. The selected-editor intervention study uses the training-step-7,000 checkpoint evaluated in Table 1. Fresh reasoning, no reasoning, and shuffled reasoning each produce 102 valid target–UE-render pairs, with eight updates and normal learned gates. Fresh recomputes the Reasoner output at every round; no reasoning removes that input; shuffled exchanges outputs between paired identities at each round. The fresh scores match the selected editor's main London evaluation. Paired intervals use 10,000 identity-level bootstrap resamples and are pointwise 95% intervals without multiplicity adjustment.

The shuffled runner uses batches of two, with 50 same-gender pairs and one cross-gender pair; the fresh runner processes one portrait at a time. This protocol changes both output assignment and inference batch size. Across the 100 identities in same-gender pairs, fresh-minus-shuffled UE ID-Sim is $+0.00141$ (95% interval $[-0.00033, 0.00307]$). The three-condition comparison in Section 5.3 reports both reasoning contrasts for the selected editor.

Inference rounds on the initial editor. The inference-round comparison uses the initial editor selected at training step 2,800, with the same 102 London portraits, native initialization, and fixed-appearance UE protocol. One round gives 0.12714 and eight rounds give 0.12860; the paired gain is 0.00145 ($[0.00044, 0.00246]$), with improvements on 65 portraits. This comparison keeps the checkpoint fixed while extending the number of native-state updates.

B.4 Identity-input comparison at high FFHQ sampling

The full static Grounder already uses the identity token, with FFHQ sampling probability 0.10 and identity-loss weight 0.05. To isolate the token's effect, two additional static-Grounder runs use the same higher settings, 0.50 and 0.50, and differ only in whether the identity token is supplied. These optional runs are distinct from the full static Grounder and the structural ablations in Table 2.

Table 5: Matched identity-input comparison on checkpoint-selection validation. Native denotes native face-state corpus validation (1,494 fixed pairs); FFHQ denotes the identity-validation set (100 portraits, four initial states each). P-Err is normalized parameter error (Eq. 5); I-Loss is surrogate-render $L_1 + 0.2(1 - \mathrm{SSIM}) + 0.1\mathrm{LPIPS}$; ID-Sim follows Section 4. The two runs use FFHQ batch probability and identity-loss weight 0.50 each; bold marks the better recorded value for each metric.

Static-Grounder variant	Native P-Err ↓	Native I-Loss ↓	Native ID-Sim ↑	FFHQ ID-Sim ↑
With identity token	**0.4571**	**0.0538**	**0.8724**	**0.1262**
Without identity token	0.4851	0.0578	0.8384	0.0772

With the identity token, native P-Err falls from 0.4851 to 0.4571 and FFHQ ID-Sim rises from 0.0772 to 0.1262 under matched 0.50/0.50 training settings. Image-derived identity information therefore helps the Grounder translate a reference portrait into more accurate native parameters. The structural ablations in Table 2 use the full Grounder's original 0.10/0.05 settings.

B.5 Cross-model checkpoint-selection validation

Table 6 compares the recorded selected checkpoints. The identity-focused iterative checkpoint has the highest FFHQ ID-Sim, while the one-shot regressor has the lowest native-state parameter error. Rendered identity and parameter error thus answer different questions; identity-guided checkpoint selection matches the goal of creating a recognizable native Character from a portrait.

Table 6: Recorded checkpoint-selection validation. Native face-state corpus validation uses 1,494 fixed different-identity pairs; FFHQ identity validation uses 100 portraits and four initial states each. Param is normalized state error ↓; image is surrogate-render error ↓; ID is surrogate-preview ID-Sim ↑. The one-shot row uses one step; iterative rows average updates 1, 2, 4, and 8.

Model (selected step)	Native param ↓	Native image ↓	Native ID ↑	FFHQ ID ↑
One-shot regressor (17,500)	**0.3793**	**0.0523**	0.8599	0.0059
Static Grounder (33,600)	0.4554	0.0537	**0.8631**	0.1278
Iterative editor, initial replay (2,800)	0.4534	0.0536	–	0.1285
Iterative editor, identity-focused replay (7,000)	0.4544	0.0534	–	**0.1839**

B.6 Recorded checkpoints within the selected replay run

The five recorded native-corpus checks for the identity-focused iterative replay vary little in four-step mean normalized next-state error (Table 7). FFHQ ID-Sim rises from 0.14158 at step 1,400 to 0.18389 at step 7,000, a 29.9% relative increase. This trajectory motivates selecting the later checkpoint by rendered identity when the target task is portrait resemblance in a native Character.

Table 7: Recorded validation checkpoints within one identity-focused iterative-editor replay run. Native param is four-step mean normalized next-state error ↓; FFHQ ID is four-step surrogate-preview ID-Sim ↑. A dash means no FFHQ ID value was recorded for that checkpoint; bold marks the best recorded value in each column.

Training step	Native param ↓	FFHQ ID ↑
1,400	**0.45298**	0.14158
2,800	0.45446	–
4,200	0.45470	–
5,600	0.45368	–
7,000	0.45442	**0.18389**