Supplementary Material for the Paper: Towards High Fidelity Monocular Face Reconstruction with Rich Reflectance using Self-supervised Learning and Ray Tracing

Abdallah Dib[1*] Cédric Thébault[1*] Junghyun Ahn[1*] Philippe-Henri Gosselin[1]
Christian Theobalt[2] Louis Chevallier[1]

[1]InterDigital R&I [2]Max-Planck-Institute for Informatics

1. Implementation details

We implemented the architecture using PyTorch [1] with a GPU-enabled backend. Ray tracing is based on the method of [2], and for training we used Adam [3] as optimizer with default parameters. We used images from CelebA dataset [4] in addition to $40K$ images collected from the web for a total of $250K$ images. We keep $2K$ images for the validation. Images are aligned and cropped to a resolution of 256×256. We trained $\mathbf{E}$ for 10 epochs, then we fixed $\mathbf{E}$ and trained $\mathbf{D_1}$ and $\mathbf{D_2}$ for 5 epochs. Finally we trained all networks jointly for 5 epochs. We set our regularization weights as following: landmarks weight $\alpha_1 = 1$, $w_i = 0.002$, $w_c = 0.01$, symmetry regularizer $w_1 = 20$, $w_{2S} = 0.01$, smoothness regularizer $w_3 = 0.0001$; and for w_{2D}, we start with $w_{2D} = 0.5$, and decrease it by a factor of 2 at each epoch. For $\mathbf{E}$, we use a pre-trained *ResNet-152* with latent space dimension equal to 1000. Both $\mathbf{D_1}$ and $\mathbf{D_2}$ networks use a cascade of 7 convolution layers. Because ray tracing is very memory consuming, we use a texture resolution of 256×256 with batch size equal to 8 and input image of resolution 256×256 to fit the GPU memory (12GB on a NVIDIA GeForce RTX 2080 Ti). For the learning rates, we use $1e^{-6}$ for $\mathbf{E}$ and $1e^{-7}$ for $\mathbf{D_1}$ and $\mathbf{D_2}$. For training, it takes 15 hours to do a single epoch. During training, we use 8 samples per pixels for ray tracing the images. We experimented with different numbers of samples per pixel (spp) for ray tracing (8, 16 and 32 spp), but we did not obtain substantial improvements when using more than 8 spp, even though using 16 spp already made the training much slower. Additionally, as skin is generally not a highly specular surface, in our experiments, modeling self-geometry ray bounces did not lead to substantial gain in accuracy; thus we did not use it for training. The inference takes 54 ms (47 ms for $\mathbf{E}$ and 7 ms for $\mathbf{D_1}, \mathbf{D_2}$).

*Equal contribution

2. Vertex-based renderer implementation

In this section we provide implementation details of the vertex-based renderer that we used to compare against the ray tracer (please refer to section 5 in primary document).

The vertex based renderer computes the irradiance by evaluating spherical harmonics (SH) for each vertex of the face mesh. To model skin reflectance, we use a simplified Cook-Torrance BRDF, thus the final irradiance is the sum of diffuse and specular irradiance terms. For the diffuse term, a spatial convolution with the *half-cosine* is applied to the SH light representation. This corresponds to a multiplication of the SH coefficients (B_{lm}) of the light representation with SH coefficients (A_l) of the *half-cosine* function ([5]). For each vertex, the diffuse irradiance, $\mathcal{B}_d$, is obtained by evaluating the resulting SH:

$$\mathcal{B}_d(\mathsf{n_i}, \mathsf{c_i}) = \mathsf{c_i} \cdot \sum_{l=0}^{8} \sum_{m=-l}^{l} A_l \cdot B_{lm} \cdot Y_{lm}(\mathsf{n_i}) \qquad (1)$$

where $\mathsf{c}_i \in \mathbb{R}^3$ is the diffuse albedo of a vertex. $\mathsf{n}_i \in \mathbb{R}^3$ is the vertex normal. The specular term is similarly obtained using a spatial convolution of the SH light representation with the BRDF kernel corresponding to the roughness (which is constant in the simplified Cook-Torrance BRDF model we use). The specular irradiance, $\mathcal{B}_s$, is obtained by evaluating the resulting SH:

$$\mathcal{B}_s(\mathsf{R_i}) = \sum_{l=0}^{8} \sum_{m=-l}^{l} S_l \cdot B_{lm} \cdot Y_{lm}(\mathsf{R_i}) \qquad (2)$$

where $\mathsf{R_i}$ is the reflection direction of the viewing vector $\mathsf{W_i}$ according to the surface normal, and S_l are the SH coefficients of the BRDF function corresponding to the roughness [5]. The final irradiance $\mathcal{B}$ is equal to the sum of the diffuse and specular terms weighted by the specular intensity s_i:

$$\mathcal{B}(\mathsf{n_i}, \mathsf{c_i}, \mathsf{R_i}) = (1 - \mathsf{s_i}) \cdot \mathcal{B}_d(\mathsf{n_i}, \mathsf{c_i}) + \mathsf{s_i} \cdot \mathcal{B}_s(\mathsf{R_i}) \qquad (3)$$

$\mathsf{s}_\mathsf{i} \in \mathbb{R}$ is the specular albedo.

Finally, We use the following vertex-based photo-consistency loss to minimize during the training:

$$\mathsf{E}_{ph}(\chi) = \sum_{i=1}^{N} |\mathcal{B}(\mathsf{n}_\mathsf{i}, \mathsf{c}_\mathsf{i}, \mathsf{R}_\mathsf{i}) - \mathcal{I}^R(\Pi \circ \mathsf{C}(\mathsf{v}_\mathsf{i}))| \quad (4)$$

where N is the number of vertices, $\mathsf{C}(\mathsf{v}_\mathsf{i})$ is the projection of vertex v_i in the real image, equal to: $\mathsf{R}^{-1}(\mathsf{v}_\mathsf{i} - \mathsf{T})$. Π is the perspective camera matrix that maps a 3D vertex to a 2D pixel.

3. Mesh difference

In this section, we provide more details on how the geometric error is calculated for each method (please refer to Table 1 in primary document).

The mean difference error is computed per-vertex on the entire mesh. We implement a 3D mesh evaluation protocol similar to [6]. For computing the mesh difference, we first align a reconstructed mesh towards a ground truth (GT) mesh. Several feature points, namely, sparse correspondence points are defined on both the GT and reconstructed facial meshes, where vertices are minimally affected by the facial muscles. With the corresponding points ready on both meshes, we use a traditional least-square estimation introduced by [7] to align the two meshes. After this alignment, we compute the distance from each vertex of a mesh to the other via a fast ray-triangle intersection method [8]. The average error is computed for final difference between two meshes.

4. More comparison results

Figure 1 shows comparison results against the method of [9]. For each subject, we show the final reconstruction, estimated diffuse, specular and light for each method. The first two subjects are from the authors of [9].

Quite logically, the iterative optimization-based method of [9] achieves slightly better reconstruction results and captures more details in the estimated albedos. This is because [9] estimates and fine-tunes the facial and scene parameters specifically for each subject, while our method infers them directly without fine-tuning. Nevertheless, our method is almost on par with [9] and can successfully handle some cases where [9] falters. For instance, with the last two subjects of the Figure 1, in presence of shadows and strong expression, landmarks detector deliver less accurate initial starting points for the method of [9] which consequently gets trapped in wrong local minima. This yields poor shape and artefacts in the estimated albedos (highlighted in red boxes). Our method does not suffer from this limitation, proves to be more robust and produces visually more plausible reconstruction. We note that [9] estimates a roughness map, a parameter that we do not estimate. However, as reported by the authors, the missing statistical prior of the estimated roughness may sometimes yield to an over-fitting on this parameter.

We show more qualitative comparison against [10], [11], [12], and [13] in Figure 2. In Figure 3, we show more quantitative comparison against [10], [11], [12], and [9].

5. Face catalog

In Figure 4, we show more reconstruction results from in-the-wild images. For each subject we show the final reconstruction and the estimated diffuse, specular albedos and illumination. More results are in the accompanied video.

6. More relighting examples

Figure 5 and 6 show more relighting examples where the estimated illumination is replaced with an environment-map. More relighting results are in the accompanied video.

References

[1] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. 2017. 1

[2] Tzu-Mao Li, Miika Aittala, Frédo Durand, and Jaakko Lehtinen. Differentiable monte carlo ray tracing through edge sampling. *ACM Trans. Graph.*, 37(6):222:1–222:11, Dec. 2018. 1

[3] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 1

[4] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015. 1

[5] Dhruv Mahajan, Ravi Ramamoorthi, and Brian Curless. A theory of frequency domain invariants: Spherical harmonic identities for brdf/lighting transfer and image consistency. *IEEE transactions on pattern analysis and machine intelligence*, 30(2):197–213, 2007. 1

[6] Rohith Krishnan Pillai, Laszlo A. Jeni, Huiyuan Yang, Zheng Zhang, Lijun Yin, and Jeffrey F. Cohn. The 2nd 3d face alignment in the wild challenge (3dfaw-video): Dense reconstruction from video. In *In Proceedings of the 2019 IEEE International Conference on Computer Vision Workshops*, October 2019. 2

[7] Shinji Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Trans. Pattern Anal. Mach. Intell.*, 13:376–380, 1991. 2

[8] Tomas Möller and Ben Trumbore. Fast, minimum storage ray-triangle intersection. *J. Graph. Tools*, 2(1):21–28, Oct. 1997. 2

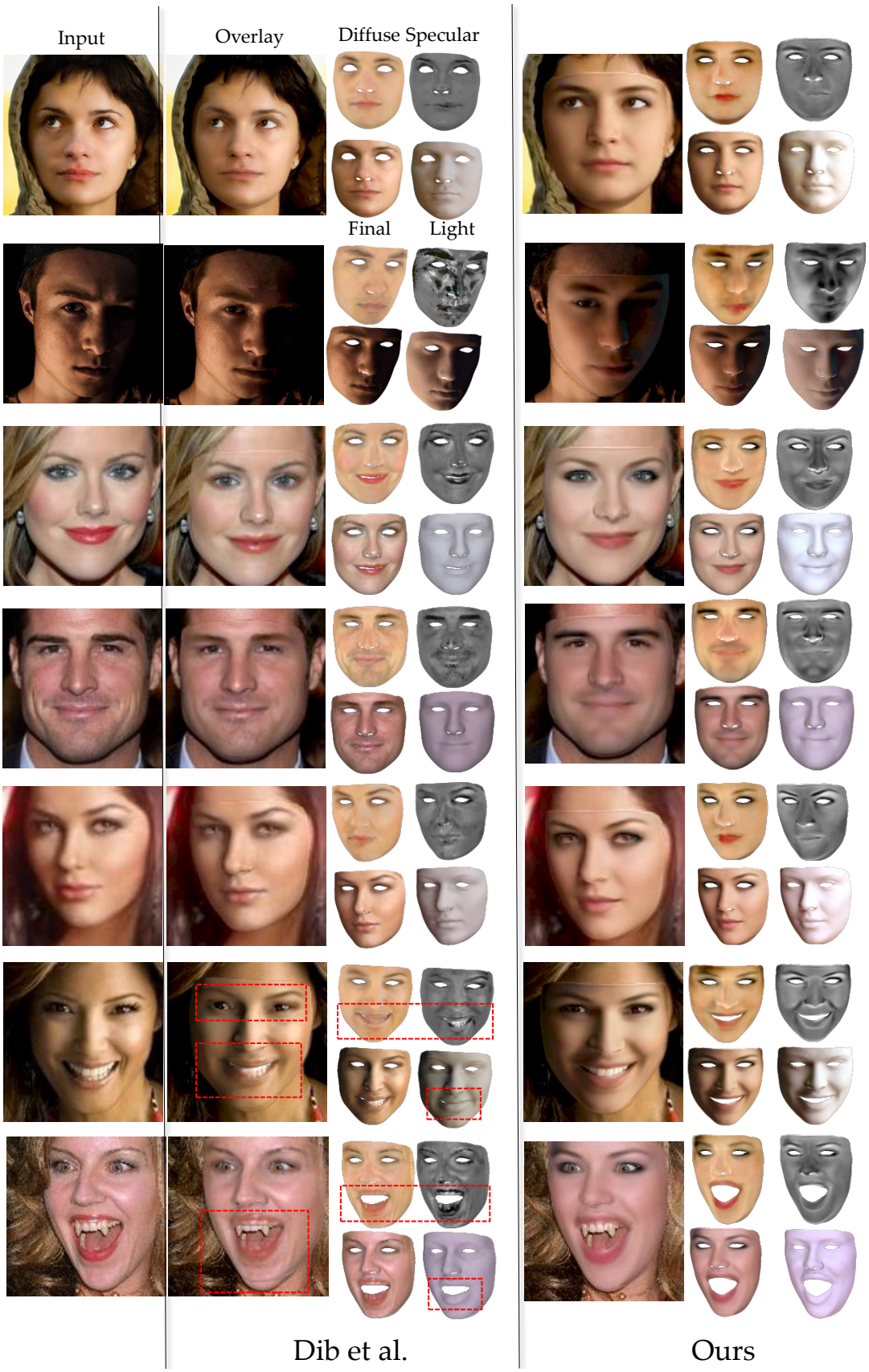


Figure 1. Comparison against [9]

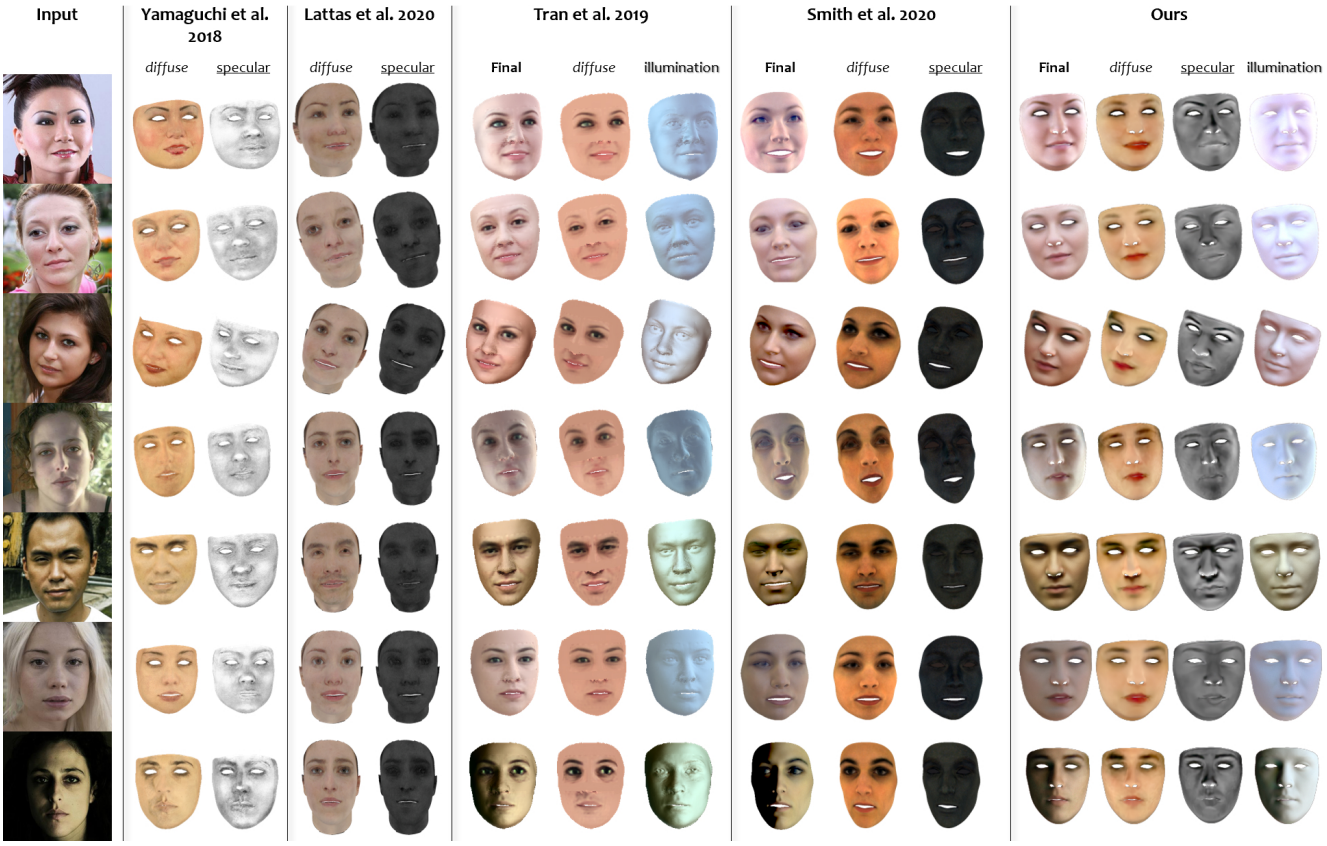


Figure 2. More visual comparisons against state-of-the-art methods.

[9] Abdallah Dib, Gaurav Bharaj, Junghyun Ahn, Cédric Thébault, Philippe-Henri Gosselin, Marco Romeo, and Louis Chevallier. Practical face reconstruction via differentiable ray tracing. *Computer Graphics Forum*, 2021. 2, 3

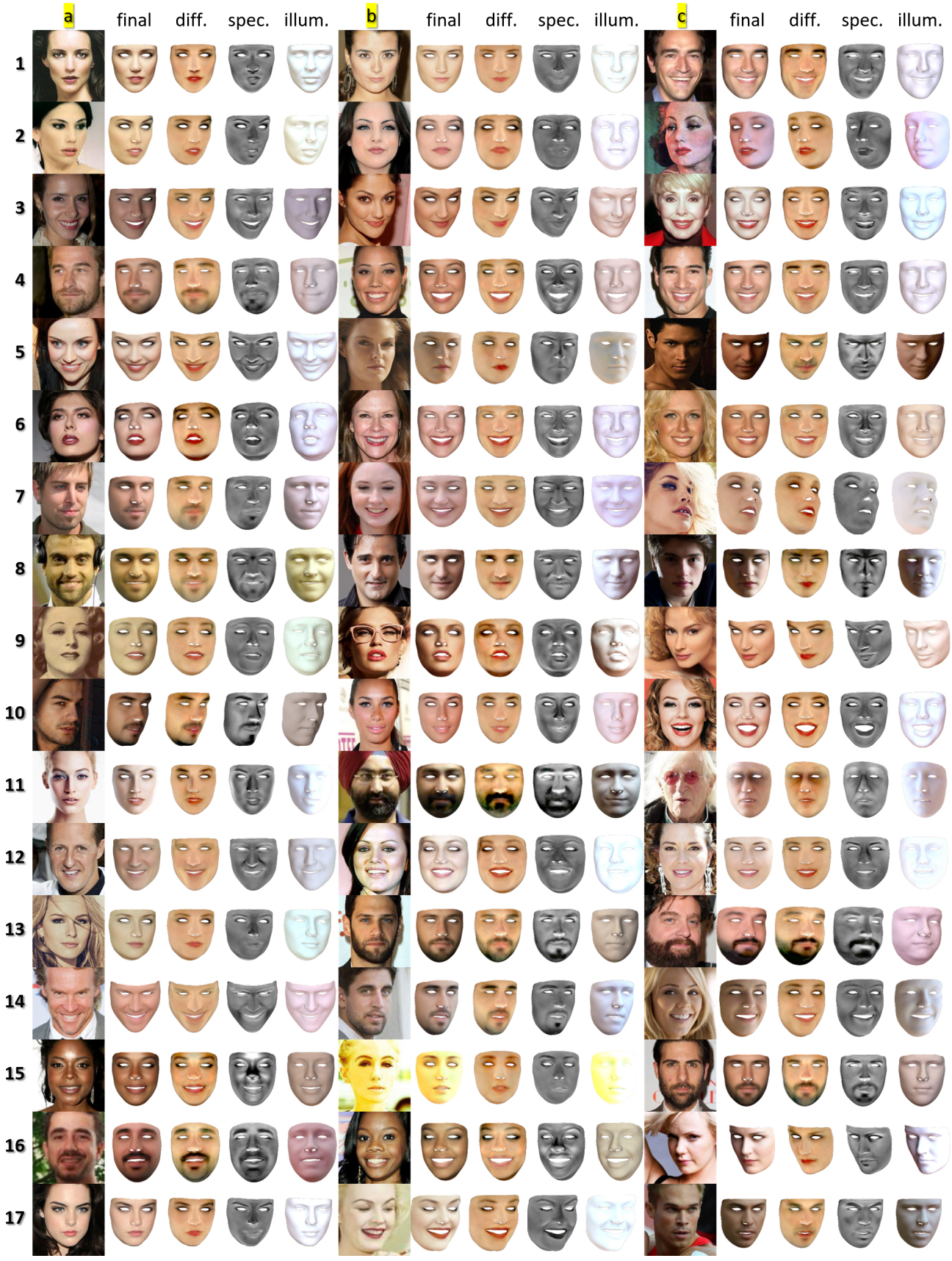
[10] S Yamaguchi, S Saito, et al. *High-fidelity Facial Reflectance and Geometry Inference from an Unconstrained Image*. ACM TOG, 2018. 2

[11] Alexandros Lattas, Stylianos Moschoglou, Baris Gecer, Stylianos Ploumpis, Vasileios Triantafyllou, Abhijeet Ghosh, and Stefanos Zafeiriou. Avatarme: Realistically renderable 3d facial reconstruction" in-the-wild". In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 760–769, 2020. 2

[12] Luan Tran, Feng Liu, and Xiaoming Liu. Towards high-fidelity nonlinear 3d face morphable model. In *In Proceeding of IEEE Computer Vision and Pattern Recognition*, Long Beach, CA, June 2019. 2

[13] William AP Smith, Alassane Seck, Hannah Dee, Bernard Tiddeman, Joshua B Tenenbaum, and Bernhard Egger. A morphable face albedo model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5011–5020, 2020. 2

Figure 3. More geometric comparisons against state-of-the-art methods.


a
final
diff.
spec.
illum.
b
final
diff.
spec.
illum.
c
final
diff.
spec.
illum.
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17

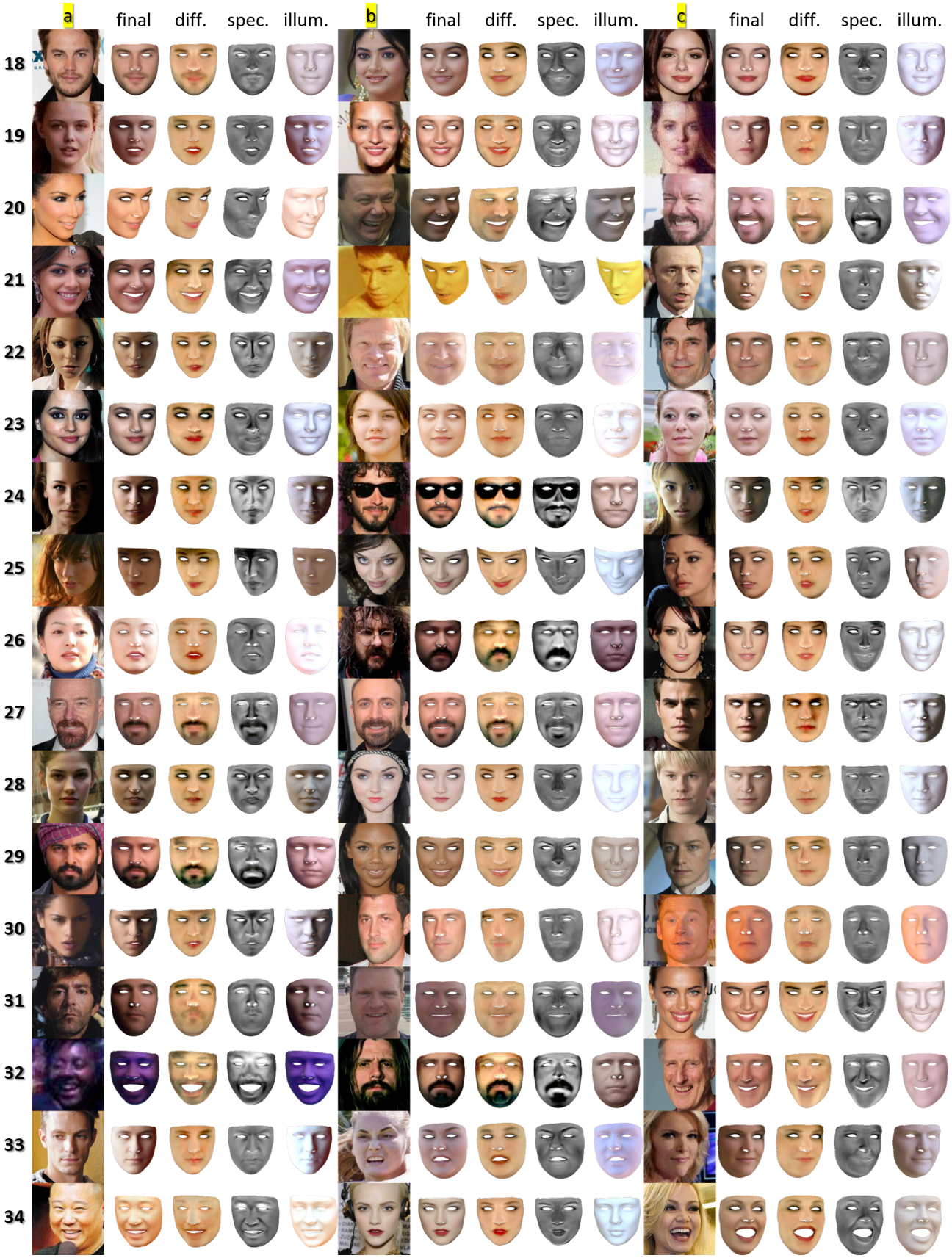


Figure 4. Face catalog of our reconstruction. For each subject, we show the input, final, diffuse, specular, and illumination. More results are in the accompanied video.

Figure 5. Relighting examples (More relighting results are in the accompanied video).

Figure 6. Relighting examples (More relighting results are in the accompanied video).