Supplementary Material:
MEGANE: Morphable Eyeglass and Avatar Network

Junxuan Li[1,2]*, Shunsuke Saito[2], Tomas Simon[2], Stephen Lombardi[2], Hongdong Li[1], Jason Saragih[2]
[1]Australian National University, [2]Meta Reality Labs Research
junxuan-li.github.io/megane

S1. Data Acquisition

In this section, we describe how we capture and process the *Eyeglasses* dataset.

S1.1. Eyeglasses Dataset

Data Capture We capture the *Eyeglasses* dataset consisting of 43 eyeglasses. The lenses from the eyeglasses were removed before capturing. We place the eyeglasses in a well lit indoor room. In addition, we place a AR-checker board with green background under the eyeglasses, as shown in Fig. S1. For each eyeglasses, we capture around 70 images from different view points with a hand-held DSLR camera. The camera intrinsics are calibrated in advance and fixed during the entire capture. We use the OpenCV detector [4] and COLMAP [10] for camera extrinsic and intrinsic calibration.

Mesh Extraction We employ NeuS [11] to reconstruct the 3D mesh of each eyeglasses from the aforementioned multi-view capture. Specifically, we use the official NeuS implementation and its default hyper-parameters to train the network. NeuS was trained for 300k iterations with an NVIDIA V100 GPU, which takes around 8 hours. Once trained, a 3D mesh of the glasses can be extracted using marching cubes [7] with a grid resolution of 512. We denote the meshes of the eyeglasses as

$$\mathcal{M}_i \in \mathbb{R}^{3\times M_i}, \quad \mathcal{V}_i \in \mathbb{R}^{3\times V_i}, \tag{S1}$$

where $\mathcal{V}_i$ are the vertices and $\mathcal{M}_i$ are the faces of the i-th glasses.

Mesh Canonicalization We deform these eyeglasses into a canonical space such that they are spatially aligned across different eyeglasses. We label a set of key points for each eyeglasses on 2D images, and then triangulate these 2D points to get the 3D key points $p_i \in \mathbb{R}^{3\times 20}$ on the eyeglasses mesh $\{\mathcal{M}_i, \mathcal{V}_i\}$. We connect these key points to form a skeleton and apply Bounded Biharmonic Weights (BBW) [5] to deform the mesh into a canonical space using linear blend skinning (LBS). Denote the linear blend skinning weights computed by BBW as $\boldsymbol{M}_i \in \mathbb{R}^{20\times V_i}$ for eyeglasses i; we optimize the transformation of the skeletons $\boldsymbol{T}_i \in \mathbb{R}^{3\times 20}$ such that the L2 distance between the transformed key points and the average key points is minimized as follows:

$$\boldsymbol{T}_i = \arg\min_{\boldsymbol{T}_i} ||p_i^g - \hat{p}||_2^2, \tag{S2}$$

where the p_i^g are the key points after applying the transformation; the transformed vertices of eyeglasses is given by

*Work done while Junxuan Li was an intern at Reality Labs Research.

Figure S1. Our setup for capturing the *Eyeglasses* dataset.

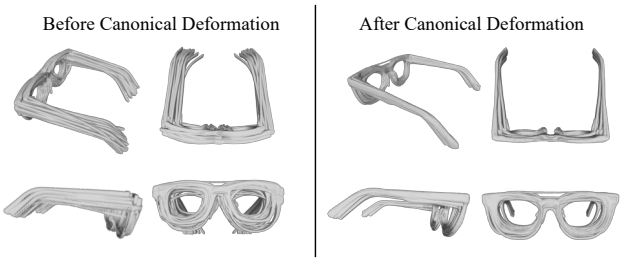

Figure S2. **Meshes of 43 eyeglasses.** The left side shows 43 eyeglasses extracted from NeuS, without spatial alignment. The right side shows the meshes in the canonical space.

Figure S3. **Eyeglasses Registration on Face.** The top row shows the projection of canonical eyeglasses mesh on face with only rigid pose alignment. The bottom row shows eyeglasses mesh after non-rigid registration. The red color denote the detected segmentation of eyeglasses on face. The blue color in the figure denotes the projection of mesh. The mesh after person-dependent deformations align accurately with the observed images.

$\mathcal{V}_i^g = \boldsymbol{T}_i \boldsymbol{M}_i$. Fig. S2 shows the effect of alignment. On the left are the extracted meshes of all 43 glasses, and the right is the deformed and transformed canonical meshes, where they are aligned based on the average key points.

Glasses Registration on Face We now register the reconstructed meshes in the canonical space to fit the image data captured in the *Faces with Eyeglasses* dataset. In this step we aim to model the person-dependent deformations of different eyeglasses on different people. For j-th subject wearing i-th eyeglasses, we compute the LBS weights of eyeglasses as $\boldsymbol{M}_i$. We choose one frame with neutral facial expression and regular eyeglasses position and fit the deformation of eyeglasses to this frame. We optimize a transformation and deformation matrix $\boldsymbol{A}_{ij}$ such that the transformed/deformed mesh $\mathcal{V}_{ij}^g = \boldsymbol{A}_{ij}\boldsymbol{M}_i$ has minimum key points loss and segmentation error:

$$\boldsymbol{A}_{ij} = \underset{\boldsymbol{A}_{ij}}{\arg\min} \left(||p_{ij}^g - p_j||_2^2 + ||\boldsymbol{I}_{\text{seg}}^g - \boldsymbol{I}_{\text{seg}}|| \right), \quad \text{(S3)}$$

where p_j are the detected glasses key points on face wearing eyeglasses images; and $\boldsymbol{I}_{\text{seg}}$ is the glasses segmentation on face wearing eyeglasses images; $\boldsymbol{I}_{\text{seg}}^g$ is the rendered segmentation mask of the deformed eyeglasses mesh $\{\mathcal{M}_i, \mathcal{V}_{ij}^g\}$.

We use stochastic gradient descent with an Adam optimizer [6] to update the skeleton transformations with a learning rate of 10^{-3} for 1000 iterations. The registration process takes around 20 minutes for each eyeglasses. As shown in Fig. S3, the deformed mesh after registration is aligned accurately with the observed images.

S2. Training and Losses

In this section, we explain the loss function and training procedures in detail.

We denote all the ground truth camera images and associated processed assets for a frame i as $\boldsymbol{I}^i$, which includes: the canonical mesh of the i-th eyeglasses $\{\mathcal{M}_i, \mathcal{V}_i^g\}$; the deformed i-th mesh on j-th subject as $\{\mathcal{M}_i, \mathcal{V}_{ij}^g\}$; the mask of the canonical mesh $\boldsymbol{I}_{\text{seg}}^g$; the mask of the deformed mesh $\boldsymbol{I}_{ij\text{seg}}^g$; the observed image $\boldsymbol{I}$; glasses segmentation of observed image $\boldsymbol{I}_{\text{seg}}$. We provide the exact formulation of each loss described in the main paper below as follows:

$$\mathcal{L}_{\text{L1}} = ||\boldsymbol{I}' - \boldsymbol{I}||, \quad \text{(S4)}$$
$$\mathcal{L}_{\text{vgg}} = \text{VGG}(\boldsymbol{I}', \boldsymbol{I}), \quad \text{(S5)}$$
$$\mathcal{L}_{\text{gan}} = \text{GAN}(\boldsymbol{I}', \boldsymbol{I}), \quad \text{(S6)}$$

where $\boldsymbol{I}'$ is the reconstructed image from volume rendering; and we follow the implementation of $\text{VGG}(\cdot), \text{GAN}(\cdot)$ in [2]. In the notation below, we use prime $\boldsymbol{I}'$ to denote the rendered results and notations without prime $\boldsymbol{I}$ to denote the corresponding ground-truth.

$$\mathcal{L}_{\text{c}} = \text{chamfer}(\mathcal{V}_i^{g'}, \mathcal{V}_i^g) + \text{chamfer}(\mathcal{V}_{ij}^{g'}, \mathcal{V}_{ij}^g), \quad \text{(S7)}$$
$$\mathcal{L}_{\text{m}} = ||\boldsymbol{I}_{\text{seg}}^{g'} - \boldsymbol{I}_{\text{seg}}^g|| + ||\boldsymbol{I}_{ij\text{seg}}^{g'} - \boldsymbol{I}_{ij\text{seg}}^g||, \quad \text{(S8)}$$
$$\mathcal{L}_{\text{s}} = ||\boldsymbol{I}_{\text{seg}}^{'} - \boldsymbol{I}_{\text{seg}}||, \quad \text{(S9)}$$

where $\text{chamfer}(\cdot)$ is the chamfer distance between two point clouds; $\mathcal{V}_i^{g'}, \mathcal{V}_{ij}^{g'}$ are the positions of eyeglasses primitives before and after person-dependent deformations respectively; and $\boldsymbol{I}_{\text{seg}}^{g'}, \boldsymbol{I}_{ij\text{seg}}^{g'}, \boldsymbol{I}_{\text{seg}}^{'}$ are the rendered eyeglasses mask, and segmentation of the corresponding eyeglasses deformations.

An l_2-regularization is also applied to the facial deformation terms

$$\mathcal{L}_{\text{L2}} = ||\delta \boldsymbol{s}||_2^2 + ||\delta \boldsymbol{R}||_2^2 + ||\delta \boldsymbol{t}||_2^2. \quad \text{(S10)}$$

The training of the network $\mathcal{A}_{\text{spec}}$ relies on estimated normals $\boldsymbol{n}$. For each eyeglasses mesh $\{\mathcal{M}_{ij}^g, \mathcal{V}_{ij}^g\}$, we extract its per-vertex surface normal and learn normals inside each primitive such that the predicted normals are coherent with the ones on the closest vertices.

During morphable geometry training, we first train the face model $\mathcal{G}_f$ on face only dataset following [2]. We then jointly train other models on face wearing glasses dataset. Likewise, during relightable appearance training, we first train the face model $\mathcal{A}_f$ on face only dataset; then we train

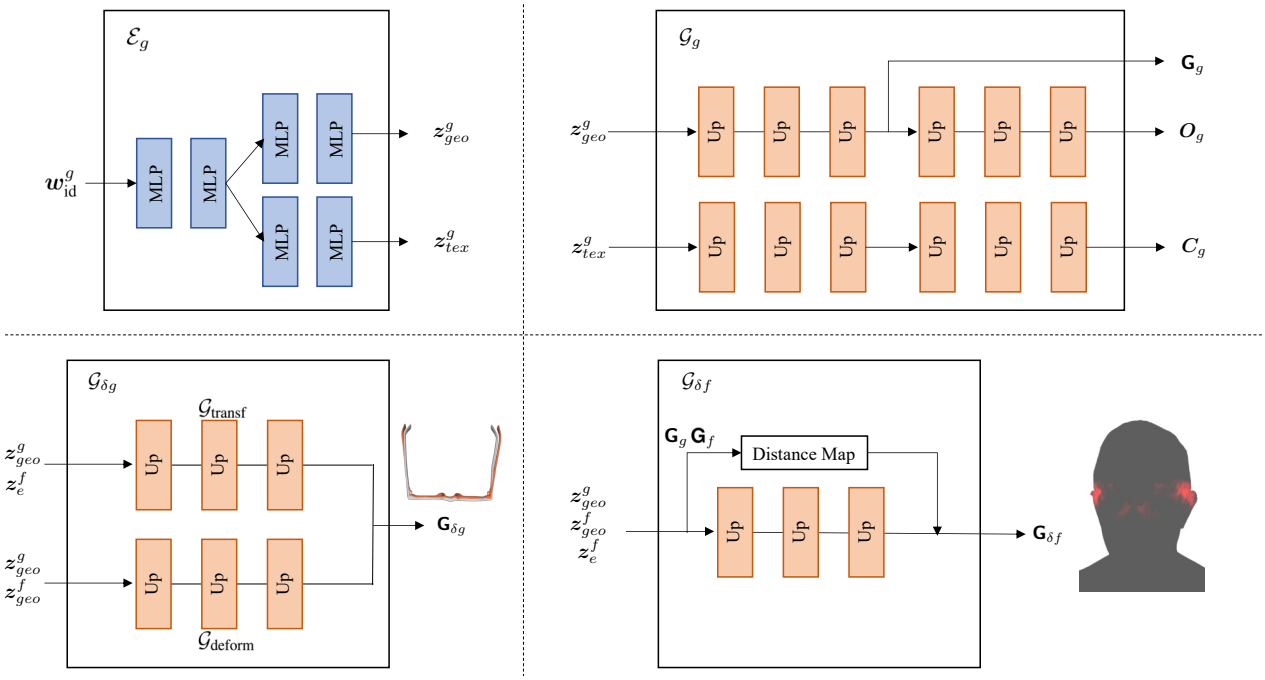

Figure S4. **Morphable Geometry Networks.** We illustrate the network architectures of $\mathcal{E}_g, \mathcal{G}_g, \mathcal{G}_{\delta g}, \mathcal{G}_{\delta f}$. The "MLP" in the figure denotes a linear layer followed by a leaky-ReLU with 0.2 negative slope. The "Up" denotes an up-sampling layer consists of a transpose convolutional layer (4×4 kernel, stride 2), followed by a leaky-ReLU. The "Distance Map" computes the l_2 distance between each of the glasses primitives to its closest face primitives.

other modules on the face wearing glasses dataset. We empirically found that the pretraining of the face modules is critical for stable training of the remaining modules including the interactions between faces and glasses.

S3. Networks Architectures

In this section, we provide the architectures of Morphable Geometry Networks and Relightable Appearance Networks in Fig. S4 and Fig. S5 separately.

S4. Lens Insertion

We propose to model lenses as a postprocess by introducing an analytical model instead of jointly modeling them with eyeglasses frames from image observations. The advantage of this analytical model of lens is that it yields plausible and photorealistic reflection and refraction for any prescription and doesn't required large dataset of lens for training. As shown in Fig. S6, we can even control the prescriptions of eyeglasses and intensity of the reflections.

Without loss the generality, we focus on the left lens for explanation. A similar formulation is applied to the right lens.

Lens boundary. We denote the detected key points of glasses in the image space. We first triangulate these key points to 3D. As shown in Fig. S7, the key points for left lens are not precise enough to draw lens. We therefore iteratively subdivide points and find the closest primitive positions. We apply the subdivision several times to obtain a fine lens boundary as shown in Fig. S8. With the estimated lens boundary, it is trivial to define a triangle mesh m of lens by connecting the lens center with each boundary point.

Lens ray-marching for refraction and reflection. During ray marching, lens refraction and reflection only happens on those rays that are intersect with the lens mesh. Given the intersection point $\boldsymbol{p}$ of the camera ray $\boldsymbol{d}$ and lens mesh m, the distorted camera ray $\boldsymbol{d}'$ is given by

$$\boldsymbol{d}' = \frac{\boldsymbol{p} - \boldsymbol{c}'}{||\boldsymbol{p} - \boldsymbol{c}'||_2}, \tag{S11}$$

$$\boldsymbol{c}' = \frac{f(\boldsymbol{c} - \boldsymbol{o})}{f + u} + \boldsymbol{o}, \quad u = (\boldsymbol{c} - \boldsymbol{o})\boldsymbol{n}^T, \tag{S12}$$

where $\boldsymbol{c}$ is the camera position; f is the lens focal length, which can be derived from prescriptions; $\boldsymbol{n}$ is the normal of lens mesh m; $\boldsymbol{o}$ is the optical center of lens, where we use the average of the lens boundary for approximation.

To compute the reflection direction, we approximate the lens as a sphere with radius r as shown in Fig. S8. The

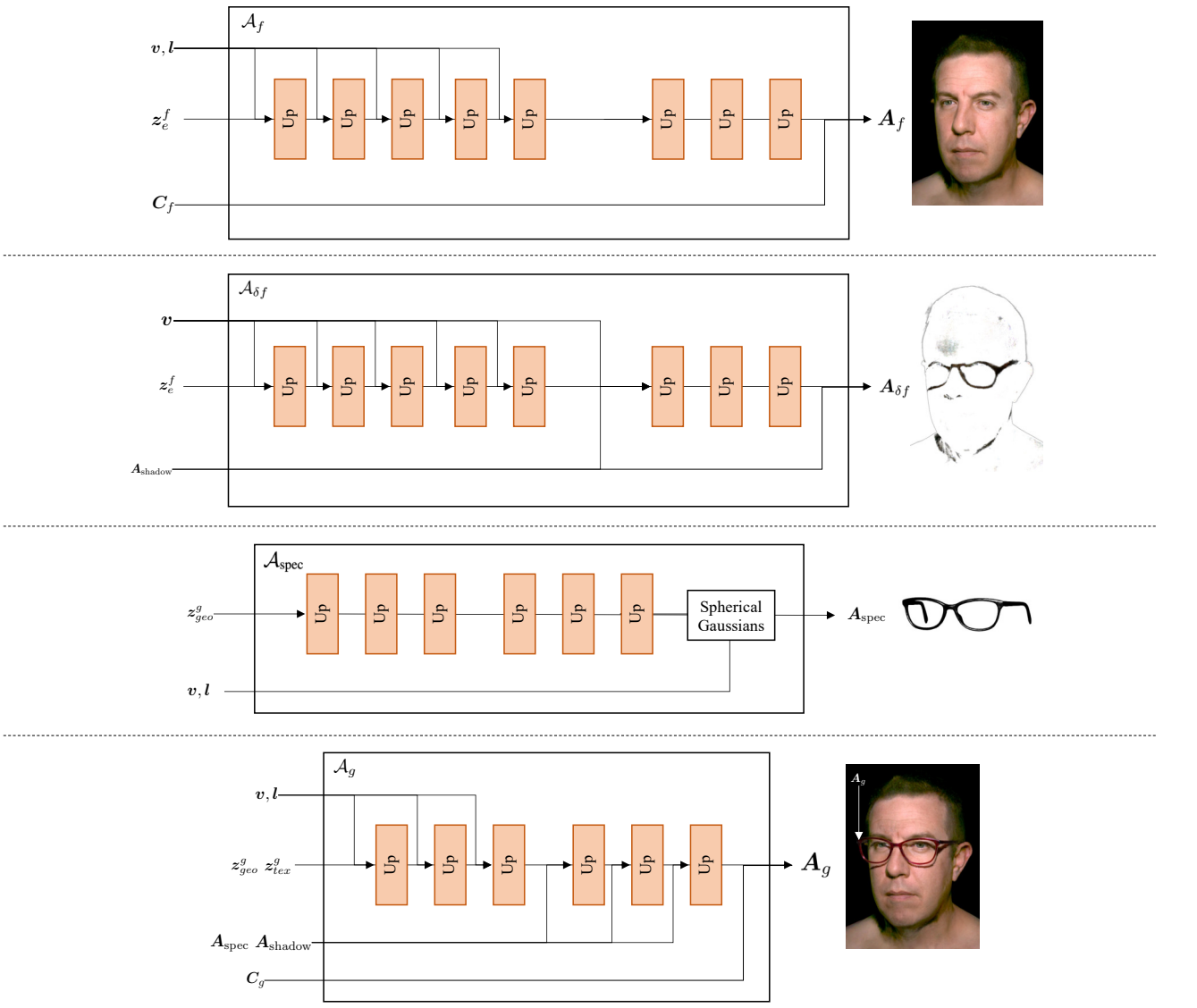

Figure S5. **Relightable Appearance Networks.** We illustrate the network architectures of $\mathcal{A}_f, \mathcal{A}_{\delta f}, \mathcal{A}_{\text{spec}}, \mathcal{A}_g$. The "Up" denotes the same operation as in Fig. S4. The "Spherical Gaussians" takes normal $\boldsymbol{n}$, view direction $\boldsymbol{v}$, and light direction $\boldsymbol{l}$ as input, and computes three specular lobes via $s = \exp(r(\frac{\boldsymbol{l}+\boldsymbol{v}}{||\boldsymbol{l}+\boldsymbol{v}||_2}\boldsymbol{n}^T - 1))$. In our experiments, we choose the following three roughness terms $r = \{64, 128, 1000\}$.

reflection ray $\boldsymbol{d}''$ can be computed as

$$\boldsymbol{d}'' = \boldsymbol{d} - 2(\boldsymbol{d}\boldsymbol{r}^T)\boldsymbol{r}, \quad \boldsymbol{d} = \frac{\boldsymbol{p}-\boldsymbol{c}}{||\boldsymbol{p}-\boldsymbol{c}||}, \tag{S13}$$

$$\boldsymbol{r} = \frac{\boldsymbol{p}-\boldsymbol{o}'}{||\boldsymbol{p}-\boldsymbol{o}'||}, \quad \boldsymbol{o}' = \boldsymbol{o} - r\boldsymbol{n}. \tag{S14}$$

When the camera ray is intersect with lens, we updated ray directions to a refraction ray and a reflective ray, and proceed the volume rendering, except when the ray does not intersect with any primitives, we query the sphere-mapped environment map. Then the reflection and refraction are blended as

$$I = \alpha I_{\text{refra}} + \beta I_{\text{refle}}, \tag{S15}$$

where α, β is the ratio of refraction and reflection respectively.

S5. Few-Shot Reconstruction

Our method is fully differentiable. Hence, once trained, we can use only a few images to reconstruct the geometry and material of novel glasses unseen during training.

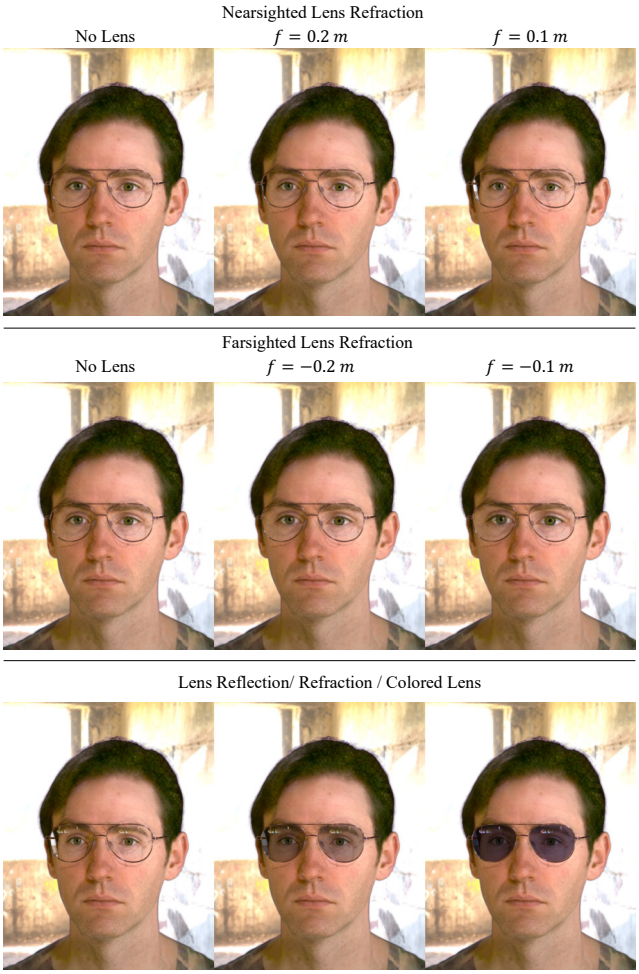

Figure S6. **Lens insertion.** Top row shows nearsighted lens refraction with different focal lengths. Second row shows farsighted lens refraction with different focal lengths. Bottom row shows the combination effects of the lens reflection, refraction and colored lens insertion.

Figure S7. **Lens boundary estimation.** We demonstrate how the coarse boundary from glasses key points is refined to a finer lens boundary. Left is the six key points detected from images. Right is the fine lens boundary after one subdivision. In our experiments, we repeat the proposed subdivision three times.

Specifically, we optimize the geometry latent code and appearance latent code via photometric loss:

$$z^g_{\text{geo}}, z^g_{\text{tex}} = \arg\min_{z^g_{\text{geo}}, z^g_{\text{tex}}} \sum ||\boldsymbol{I}' - \boldsymbol{I}||, \tag{S16}$$

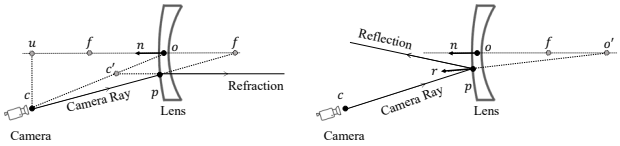

Figure S8. **Lens refraction and reflection.** The left and right shows how the ray refraction and reflection works in our lens insertion modeling.

where $\boldsymbol{I}$ is the observed images; and $\boldsymbol{I}'$ is the rendered images using the latent code $z^g_{\text{geo}}, z^g_{\text{tex}}$.

S6. Implementation details of comparisons

In this section, we explain implementation details of our comparisons including the modifications we make to GeLaTO [8] and GIRAFFE [9] to support our own dataset.

S6.1. Implementation of GeLaTO [8]

GeLaTO is not open sourced, and their training dataset is also not released. Therefore, we implement their method following their paper and train using our *Faces with Eyeglasses* dataset. We use the detected segmentation of glasses as a ground-truth foreground mask. To align the three billboards proposed in their work, we use the detected 3D key points as in our training pipeline for fair comparison.

S6.2. Implementation of GIRAFFE [9]

GIRAFFE proposed a compositional neural radiance field that supports adding and changing objects in a scene. However, the official implementation only supports objects within the same category. For a fair comparison, we adapt their method to support adding generative objects in multiple categories.

Specifically, their official implementation supports only two models: a background model and a object model. We extend this to support three different models for a background, faces, and glasses. To further facilitate the decomposition of faces and glasses, we combine *Faces only* and *Faces with Eyeglasses* datasets for training. Without this, we observe that GIRAFFE learns to model faces and glasses in a single model as they are always co-located.

S6.3. Implementation of Envmap Relighting

Following Bi *et al*. [1], we represent environmental lights as a set of distant lights, and compute shadow features for each light source. Due to the linearity of light transport, we can synthesize faces and eyeglasses under arbitrary environmental lights by linearly blending contributions of each light. Note that the global intensity and color balance may not be consistent between ours and Lumos because Lumos

does not release their tone mapping function or global intensity scale.

S7. Limitations

While our model successfully models the deformation residuals on glasses conditioned by face identity and expressions, the initial position of glasses and subtle motions caused by facial expression changes are entangled. Future work could address this limitation by incorporating more fine-grained data capture and loss functions to facilitate disentanglement. Another limitation is that our current framework infers relighting results under a single point light. On one hand, due to the linearity of light transport [3], we can synthesize physically plausible relighting under natural illuminations by weighted sum of multiple point light sources. On the other hand, running the relighting network for each point light source is too expensive for real-time use. As demonstrated for face relighting [1], distilling our point-light based model to an efficient student model should be possible for real-time use cases.

References

[1] Sai Bi, Stephen Lombardi, Shunsuke Saito, Tomas Simon, Shih-En Wei, Kevyn Mcphail, Ravi Ramamoorthi, Yaser Sheikh, and Jason Saragih. Deep relightable appearance models for animatable faces. *ACM Transactions on Graphics (TOG)*, 40(4):1–15, 2021. 5, 6

[2] Chen Cao, Tomas Simon, Jin Kyu Kim, Gabe Schwartz, Michael Zollhoefer, Shun-Suke Saito, Stephen Lombardi, Shih-En Wei, Danielle Belko, Shoou-I Yu, et al. Authentic volumetric avatars from a phone scan. *ACM Transactions on Graphics (TOG)*, 41(4):1–19, 2022. 2

[3] Paul Debevec, Tim Hawkins, Chris Tchou, Haarm-Pieter Duiker, Westley Sarokin, and Mark Sagar. Acquiring the reflectance field of a human face. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 145–156, 2000. 6

[4] Sergio Garrido-Jurado, Rafael Muñoz-Salinas, Francisco José Madrid-Cuevas, and Manuel Jesús Marín-Jiménez. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, 47(6):2280–2292, 2014. 1

[5] Alec Jacobson, Ilya Baran, Jovan Popovic, and Olga Sorkine. Bounded biharmonic weights for real-time deformation. *ACM Trans. Graph.*, 30(4):78, 2011. 1

[6] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 2

[7] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '87, page 163–169, New York, NY, USA, 1987. Association for Computing Machinery. 1

[8] Ricardo Martin-Brualla, Rohit Pandey, Sofien Bouaziz, Matthew Brown, and Dan B Goldman. Gelato: Generative latent textured objects. In *European Conference on Computer Vision*, pages 242–258. Springer, 2020. 5

[9] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11453–11464, 2021. 5

[10] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1

[11] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *Advances in Neural Information Processing Systems*, 34:27171–27183, 2021. 1