Gaussian Shadow Casting for Neural Characters - Supplemental Document

Luis Bolanos[1] Shih-Yang Su[1] Helge Rhodin[1,2]
[1]The University of British Columbia [2]Bielefeld University

This supplemental document supplies additional details on the mathematical derivation and training details to aid future work and possible extensions. It also contains extra figures showcasing more results for the experiments described in the main document, as well as demonstrating our relighting methods on other datasets.

1. HDRi Relighting

We are able to relight, not only by changing a single, primary light source direction, but by using a high dynamic range image (HDRi) that defines the environment illumination. For HDRi relighting, we rely on the Gaussian density model to query visibility and pair it with a diffuse reflectance model. We cast multiple secondary rays for each pixel from the surface of the model towards to the environment map, typically 64 secondary rays per pixel. The first of which is important sampled towards the brightest region in the HDRi, i.e. the sun. The rest are sampled according to diffuse reflection, with probability proportional to the cosine angle between normal and sample direction. This distribution can be attained by setting the ray direction as the unit surface normal and adding a random point on the unit sphere. This is a simple yet effective model that serves our purpose by sampling rays that contribute most with high likelihood, considering both the brightest region in the environment map and those that contribute most to diffuse reflection. In the same manner, the Gaussian shadow casting could also be integrated into a physically accurate illumination model by including a full BRDF function and an unbiased importance sampling method for reducing variance.

Relighting using HDRis can be seen in the Supplemental Video and the teaser in the main paper.

Relighting on MonoPerfCap and Animal Datasets. Our Gaussian method can also be used as a standalone relighting tool for datasets captured under uniform illuminations. These datasets have very few lighting and shading effects which allows for the direct interpretation of the neural color field as the albedo. We can simply fit the Gaussian density model and relight the learned avatars with HDRis. We test this paradigm on a monocular sequence from the MonoPerfCap dataset [6]. See Figure 1 for the results.

Figure 1. **HDRi Relighting on MonoPerfCap.** Our Gaussian relighting method can work on recordings done under uniform illumination, even monocular datasets.

Figure 2. **HDRi Relighting on Animal Dataset.** Our method can extend beyond human avatars as we do not necessitate any templates.

Likewise, due to the template-less nature of our implementation, we are able to learn bodies with Gaussian density models for non-human characters. We test this using the Animal dataset [3] as seen in Figure 2.

2. Novel Poses on Outdoor Sequence

We showcase more novel-pose results on a real sequence captured outdoors in bright daylight Figure 3. Our explicit lighting module results in more accurate shadows compared to the baselines. The baselines, which overfit to the training set, are highly inconsistent with small perturbations in pose leading to large changes in the shadow.

3. Novel-View Rendering Results

We also test the ability to render novel views using our method. We find that due to the explicit nature of our shading computations, the lighting and shadows are still accurate. The baselines fail to produce accurate illuminations and are much more susceptible to small pose variations leading to large changes, as seen in Figure 4. In contrast to the training cameras, the novel view is backlit with large

Figure 3. **Outdoors in sunlight (Real Sequence 1).** Novel poses rendered using DANBO [4], NPC [5] and our method. Our method has more consistent lighting and shadows whereas the baselines suffer from large shadow changes from small pose variations.

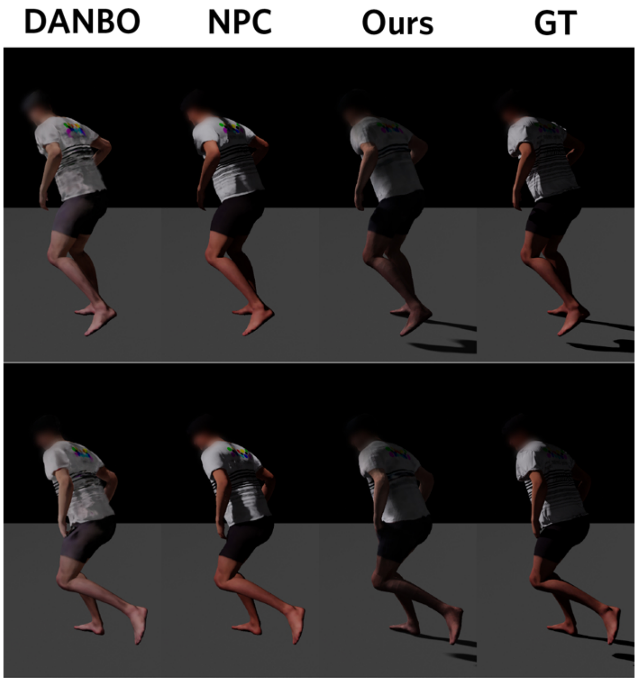

Figure 4. **Novel-view Synthesis (Synthetic Sequence).** Due to the explicit nature of our lighting module, our models are robust to changes in view. The baselines are highly pose-dependant with small changes in pose affecting the image drastically.

parts of the trunk in shadow. Consequently, the quantitative analysis in Table 1 demonstrates even larger performance gains than the novel pose evaluation in the main document. The result further highlights the importance of our explicit shadow-casting method.

4. Training Details

4.1. Loss Functions & Regularizations

Our primary objective is the accurate reconstruction of the neural character renders $\hat{\mathbf{I}}$ and the training images $\mathbf{I}$. We use a standard photometric reconstruction loss between the pixel color values in the training image, $\mathbf{c}$, and reconstruction, $\hat{\mathbf{c}}$.

Table 1. **Novel-pose in novel-view synthesis (all test frames).** As our method is explicit, large changes is view direction still result in accurate shadowing unlike the baselines.

	Novel View		
	PSNR↑	SSIM↑	LPIPS↓
DANBO	18.85	0.773	0.179
NPC	<u>20.64</u>	<u>0.823</u>	<u>0.157</u>
Ours	**27.19**	**0.873**	**0.153**

$$\mathcal{L}_{\text{RGB}} = |\hat{\mathbf{c}} - \mathbf{c}| \tag{1}$$

We augment this by also using a mask loss that operates between the integrated accumulation, $\hat{\rho}$, and the foreground mask, ρ.

$$\mathcal{L}_{\text{mask}} = |\hat{\rho} - \rho| \tag{2}$$

Our final photometric loss is one we dub the grey loss. Its objective is to initialize the RGB head of the NeRF to output a light grey value such that when the standard RGB loss starts to have an influence on the training, it is not prone to getting stuck in a local minimum with shadows already learnt caused by an initialization resulting in darker color values. It also provides enough time for the light direction to optimize and fit before the NeRF overfits to the shadows as we interpolate between the grey loss and the RGB loss.

$$\mathcal{L}_{\text{grey}} = |\hat{\mathbf{c}} - 0.75| \tag{3}$$

To fit the Gaussians, we introduce a Gaussian Density loss which minimizes the squared distance between the Gaussian density function at a given query location, $\mathbf{G}(\mathbf{x})$, and the density head of the NeRF at the same query location, $\mathbf{D}(\mathbf{x})$.

$$\mathcal{L}_{\text{gDensity}} = ||\mathbf{G}(\mathbf{x}) - \mathbf{D}(\mathbf{x})||^2 \tag{4}$$

We regularize our Gaussians by supervising their mean and standard deviations. The standard deviation regularization limits the size of the Gaussians to approximately be within 2.5 and 50 centimeters, while the mean regularization prevents Gaussians from drifting too far from the bone centers, $\mathbf{b}$. These loss functions are visualized in Figure 5.

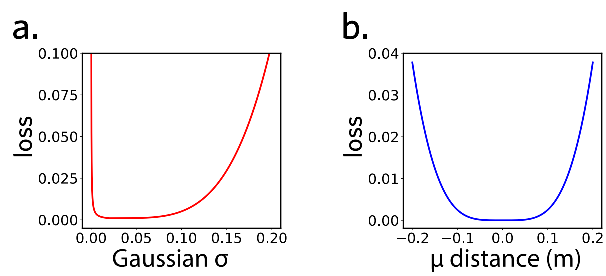

Figure 5. **Regularization on Gaussian Density Model.** a) The regularization function for the standard deviations of the Gaussians, constraining the size. b) The regularization function for the means of the Gaussians, keeping the Gaussians close to the center of the bones.

$$\mathcal{L}_{\text{gSigma}} = \begin{cases} \frac{2e-5}{\sigma} & \sigma \leq 0.02 \\ 100(\sigma - 0.02)^4 + 0.001 & \sigma > 0.02 \end{cases} \tag{5}$$

$$\mathcal{L}_{\text{gMean}} = (100(\mu - \mathbf{b})^4 + 1)^{\frac{1}{4}} - 1 \tag{6}$$

We also regularize the ambient intensity, $\hat{\mathbf{L}}_{\text{amb}}$, to be somewhat dark values to prevent the model from setting a bright ambient value and learning all of the shadows as part of the NeRF color.

$$\mathcal{L}_{\text{amb}} = ||\hat{\mathbf{L}}_{\text{amb}} - 0.1||^2 \tag{7}$$

We adopt the SDF-based density field from VolSDF [7] for improved normals, and regularize the SDF network using an Eikonal loss [2] $\mathcal{L}_{\text{Eikonal}}$ to predict proper level sets,

$$\mathcal{L}_{\text{Eikonal}} = ||\hat{\mathbf{n}_s}(\mathbf{x}) - 1||_2^2 \tag{8}$$

and use a curvature loss, $\mathcal{L}_{\text{Curvature}}$, to smoothen out the geometry by minimizing the difference between neighbouring normals,

$$\mathcal{L}_{\text{Curvature}} = ||\hat{\mathbf{n}_s}(\mathbf{x}) - \hat{\mathbf{n}}_s(\mathbf{x} + \epsilon)||^2, \tag{9}$$

with ϵ a small random perturbation.

Our total loss is the sum of all of these losses and regularization terms, each with a weighing function, $\alpha_i(t)$, for loss term i and training iteration t. Training takes around 20 hours on an NVIDIA RTX 3090, similar to Relighting4D [1].

4.2. Scheduled Learning

Our scheduled learning can be split into 3 segments. Denoting a change in the weights for each of the loss terms throughout the training using linear interpolation.

Segment 1: Density Fitting $\sim 1k$ Iterations. In this step, the main goal for the model is to train the neural field's density to fit the silhouette of the character. It begins with high weights for only $\mathcal{L}_{\text{mask}}$ and $\mathcal{L}_{\text{grey}}$ alongside the regularizers for curvature, $\mathcal{L}_{\text{Curvature}}$, and Eikonal constraints $\mathcal{L}_{\text{Eikonal}}$. We need this first stage as our shading computations rely on accurate depth maps, normal maps and accurate Gaussian fits, which the latter requires an accurate density field to fit to. Training the RGB head directly from the start results in many artifacts that the network cannot recover from due to the deferred nature of our shading computations.

Segment 2: Gaussian Density Model Fitting $\sim 4k$ Iter. This segment marks the addition of the Gaussian density loss, $\mathcal{L}_{\text{gDensity}}$, and its regularizers, $\mathcal{L}_{\text{gMean}}$ and $\mathcal{L}_{\text{gSigma}}$. At which point the parameters of the Gaussians, $\mathbf{G}$, are optimized to fit to the pretrained NeRF's density.

Segment 3: Light Fitting & RGB Fitting This step switches from using the grey loss to the RGB loss. Iterpolation between the two ensures a smooth transition between purely optimizing for the silhouette and our target color reconstruction. In our experiments, $1k$ iterations were sufficient to fully optimize the light direction, at which point the diffuse and Gaussian shadow computation is fairly accurate, allowing the neural color field to learn color without shadows, more closely resembling the albedo, see the Supplemental Video.

Weight Modulation: As previously mentioned, our total loss is the sum of all of our loss terms each with a weighing function $\alpha_i(t)$ which modulates the weight during training to allow the previously mentioned stages to train properly. We plot the value of each of the weighing functions over the training iterations in Figure 6. They are linearly interpolating between two values over a number of iterations with a hold-off period.

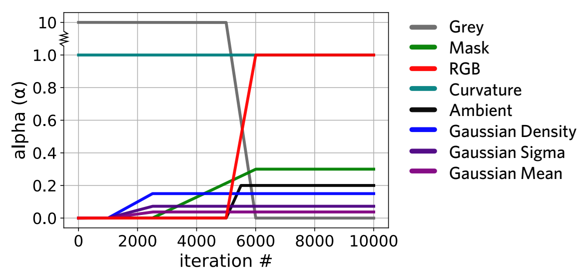

Figure 6. **Weight Modulation.** How the weights $\alpha_i(t)$ change throughout the training. First focusing on fitting a grey silhouette and the Gaussian model, later transitioning to fir the RGB alongside ambient regularization.

5. Analytical Gaussian Integral

Our goal is to derive a 1D function $G^r(t)$ that represents the density along a ray $\mathbf{r}$ that we can integrate to acquire the cast shadow. We start our derivation from the 3D anisotropic Gaussian $\mathbf{G}(\mathbf{x})$ that we use to approximate the body density field,

$$\begin{aligned} \mathbf{G}(\mathbf{x}) &= \mathcal{C} \exp\left[-0.5(\mu - \mathbf{x})^T \Sigma^{-1} (\mu - \mathbf{x})\right] \\ &= \mathcal{C} \exp\left[-0.5(\mu^T \Sigma^{-1} \mu - 2\mu^T \Sigma^{-1} \mathbf{x} + \mathbf{x}^T \Sigma^{-1} \mathbf{x})\right]. \end{aligned} \tag{10}$$

We can infer the density along the ray, $G^r(t)$, by parameterizing the 3D positions along the ray by the distance t from the origin $\mathbf{r}_o$, with $\mathbf{x} = \mathbf{r}_o + t \cdot \mathbf{r}_d$, and substituting it into

Equation 10,

$$
\begin{aligned}
&G^r(t)\\
&= \hat{\mathcal{C}} \exp\left[-0.5(t^2\mathbf{r}_d^T\Sigma^{-1}\mathbf{r}_d - 2t\mathbf{r}_d^T\Sigma^{-1}(\mathbf{r}_o-\mu))\right]\\
&= \hat{\mathcal{C}} \exp\Big[-0.5\mathbf{r}_d^T\Sigma^{-1}\mathbf{r}_d\left(t^2 - 2t\frac{\mathbf{r}_d^T\Sigma^{-1}(\mathbf{r}_o-\mu)}{\mathbf{r}_d^T\Sigma^{-1}\mathbf{r}_d}\right)\Big],
\end{aligned}
\tag{11}
$$

where $\mathbf{r}_d$ is the ray direction, and $\hat{\mathcal{C}}$ consists of the terms that are constants with respect to the ray distance t, separated by the equality $\exp(a+b) = \exp(a)\exp(b)$,

$$
\hat{\mathcal{C}} = \mathcal{C}\exp\left[-0.5(\mu-\mathbf{r}_o)^T\Sigma^{-1}(\mu-\mathbf{r}_o)\right]. \tag{12}
$$

We then reorganize Equation 11 by substituting $\hat{\sigma} = \frac{1}{\sqrt{\mathbf{r}_d^T\Sigma^{-1}\mathbf{r}}}$ and $\bar{\mu} = \frac{\mathbf{r}_d^T\Sigma^{-1}(\mathbf{r}_o-\mu)}{\mathbf{r}_d^T\Sigma^{-1}\mathbf{r}_d}$,

$$
\begin{aligned}
G^r(t) &= \hat{\mathcal{C}}\exp\Big[-0.5\frac{(t^2-2t\bar{\mu})}{\bar{\sigma}^2}\Big]\\
&= \hat{\mathcal{C}}\exp\Big[-0.5\frac{(t^2-2t\bar{\mu}+\bar{\mu}^2)-\bar{\mu}^2}{\bar{\sigma}^2}\Big]\\
&= \hat{\mathcal{C}}\exp\Big[-0.5\frac{(t-\bar{\mu})^2-\bar{\mu}^2}{\bar{\sigma}^2}\Big]\\
&= \bar{\mathcal{C}}\exp\left[-\frac{(\bar{\mu}-t)^2}{2\bar{\sigma}^2}\right],
\end{aligned}
\tag{13}
$$

where, again, $\bar{\mathcal{C}}$ absorbs the terms that are constant to t,

$$
\bar{\mathcal{C}} = \mathcal{C}\exp\left[-0.5\left((\mu-\mathbf{r}_o)^T\Sigma^{-1}(\mu-\mathbf{r}_o) - \frac{\bar{\mu}^2}{\bar{\sigma}^2}\right)\right],
\tag{14}
$$

and we arrive at $G^r(t)$ that assumes the form of a 1D Gaussian density function with mean $\bar{\mu}$, std $\bar{\sigma}$, and scaling factor $\bar{\mathcal{C}}$. As the integral through a Gaussian can be computed in closed form through the error function, this enables analytical integration of the density along the ray, which in turn enables our method to cast shadows efficiently.

References

[1] Zhaoxi Chen and Ziwei Liu. Relighting4d: Neural relightable human from videos. In *ECCV*, 2022. 4

[2] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes. In *ICML*, 2020. 4

[3] Ruilong Li, Julian Tanke, Minh Vo, Michael Zollhofer, Jurgen Gall, Angjoo Kanazawa, and Christoph Lassner. Tava: Template-free animatable volumetric actors. In *ECCV*, 2022. 1

[4] Shih-Yang Su, Timur Bagautdinov, and Helge Rhodin. Danbo: Disentangled articulated neural body representations via graph neural networks. In *ECCV*, 2022. 2

[5] Shih-Yang Su, Timur Bagautdinov, and Helge Rhodin. Npc: Neural point characters from video. In *ICCV*, 2023. 2

[6] Weipeng Xu, Avishek Chatterjee, Michael Zollhöfer, Helge Rhodin, Dushyant Mehta, Hans-Peter Seidel, and Christian Theobalt. Monoperfcap: Human performance capture from monocular video. *ACM Trans. Graph.*, 37(2):27:1–27:15, May 2018. 1

[7] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. In *NeurIPS*, 2021. 4