Beyond Volumetric Albedo — A Surface Optimization Framework for Non-Line-of-Sight Imaging: Supplementary Material

Chia-Yin Tsai, Aswin C. Sankaranarayanan, and Ioannis Gkioulekas
Carnegie Mellon University

1. Introduction

In this supplement, we cover the following topics:

- In Section 2, we prove and discuss Proposition 1. In particular, we detail the GGX reflectance model, and derive the associated derivative expressions. We then derive the expression for the derivative with respect to mesh vertices. Finally, we discuss computing derivatives while correctly accounting for visibility terms.
- In Section 3, we discuss in detail the rendering algorithm used for estimating the transient and its derivatives. Also, we discuss extensively our optimization procedure and related engineering issues.
- In Section 4, we show additional experimental results. In particular, we provide quantitative metrics for the shape reconstruction experiments in the main paper. We additionally evaluate performance for different numbers of measurements and levels of noise.

2. Differentiating transients

2.1. Reflectance model

We assume that the NLOS surface has a spatially-uniform BRDF, which we model using the GGX microfacet BRDF [29]. We first define the surface normal $\hat{\boldsymbol{n}}$, incoming and outgoing directions $\boldsymbol{\omega}_i$ and $\boldsymbol{\omega}_o$, and *half-vector* $\mathbf{h} = (\boldsymbol{\omega}_i + \boldsymbol{\omega}_o)/\|\boldsymbol{\omega}_i + \boldsymbol{\omega}_o\|$. Then, the BRDF model is:

$$f_s\left[\boldsymbol{\pi}\right](\hat{\boldsymbol{n}}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o) = f_{\text{GGX}}\left[\alpha\right](\hat{\boldsymbol{n}}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o) = \frac{F_{\text{Cook-Torrance}}(\boldsymbol{\omega}_o, \mathbf{h}) \cdot D_{\text{GGX}}(\hat{\boldsymbol{n}}, \mathbf{h}) \cdot G_{\text{Smith}}(\hat{\boldsymbol{n}}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o)}{4}, \tag{14}$$

where

$$D_{\text{GGX}}(\hat{\boldsymbol{n}}, \mathbf{h}) = \frac{\alpha^2}{\pi\left[\left\langle\hat{\boldsymbol{n}}, \mathbf{h}\right\rangle^2(\alpha^2 - 1) + 1\right]^2}, \tag{15}$$

$$G_{\text{Smith}}(\hat{\boldsymbol{n}}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o) = G_1(\hat{\boldsymbol{n}}, \boldsymbol{\omega}_i) \cdot G_1(\hat{\boldsymbol{n}}, \boldsymbol{\omega}_o), \tag{16}$$

$$G_1(\hat{\boldsymbol{n}}, \boldsymbol{\omega}) = \frac{2}{\left\langle\hat{\boldsymbol{n}}, \boldsymbol{\omega}\right\rangle + \sqrt{\alpha^2 + (1 - \alpha^2)\left\langle\hat{\boldsymbol{n}}, \boldsymbol{\omega}\right\rangle^2}}, \tag{17}$$

and $F_{\text{Cook-Torrance}}(\boldsymbol{\omega}_o, \mathbf{h})$ is the Fresnel reflection of an ideal reflector, independent of α. This leads to a one-dimensional reflectance parameterization, $\boldsymbol{\pi} = \{\alpha\}$, where the parameter $\alpha \in [0, 1]$ controls surface roughness.

2.2. Proposition 1

Proposition 1 of the main paper is a direct consequence of the Equations (18) and (21) we derive below.

Reflectance derivatives. Differentiating the surface integral of Equation (1) of the main paper with respect to reflectance is straighforward. By observing that the integration measure is independent of the reflectance parameters $\boldsymbol{\pi}$, and under weak continuity conditions that are known to be satisfied for radiometric integrals [2], we can simply exchange the order of differentiation and integration to obtain

$$\left.\frac{\partial I\left(t; \boldsymbol{l}, \boldsymbol{s}\right)}{\partial \boldsymbol{\pi}}\right|_{\boldsymbol{v}, \boldsymbol{\pi}} = \int_{\mathcal{S}_{\text{NLOS}}} \overbrace{W\left(\boldsymbol{x}; t\right) \left.\frac{\partial f\left(\boldsymbol{x}, \hat{\boldsymbol{n}}\left(\boldsymbol{x}\right)\right)}{\partial \boldsymbol{\pi}}\right|_{\boldsymbol{v}, \boldsymbol{\pi}}}^{g_r(\boldsymbol{x}, \hat{\boldsymbol{n}}(\boldsymbol{x}))} \cdot v\left(\boldsymbol{x}, \boldsymbol{l}\right) v\left(\boldsymbol{x}, \boldsymbol{s}\right) \mathrm{d}A\left(\boldsymbol{x}\right), \tag{18}$$

where

$$\left.\frac{\partial f\left(\boldsymbol{x}, \hat{\boldsymbol{n}}\left(\boldsymbol{x}\right)\right)}{\partial \boldsymbol{\pi}}\right|_{\boldsymbol{v}, \boldsymbol{\pi}} = \left.\frac{\partial f_s\left[\boldsymbol{\pi}\right]\left(\hat{\boldsymbol{n}}\left(\boldsymbol{x}\right), \hat{\boldsymbol{\omega}}_{\boldsymbol{l}}\left(\boldsymbol{x}\right), \hat{\boldsymbol{\omega}}_{\boldsymbol{s}}\left(\boldsymbol{x}\right)\right)}{\partial \boldsymbol{\pi}}\right|_{\boldsymbol{v}, \boldsymbol{\pi}} \cdot \frac{\left\langle-\hat{\boldsymbol{\omega}}_{\boldsymbol{l}}\left(\boldsymbol{x}\right), \hat{\boldsymbol{n}}\left(\boldsymbol{l}\right)\right\rangle \left\langle\hat{\boldsymbol{\omega}}_{\boldsymbol{l}}\left(\boldsymbol{x}\right), \hat{\boldsymbol{n}}\left(\boldsymbol{x}\right)\right\rangle}{\left\|\boldsymbol{x} - \boldsymbol{l}\right\|^2} \cdot \frac{\left\langle-\hat{\boldsymbol{\omega}}_{\boldsymbol{s}}\left(\boldsymbol{x}\right), \hat{\boldsymbol{n}}\left(\boldsymbol{s}\right)\right\rangle \left\langle\hat{\boldsymbol{\omega}}_{\boldsymbol{s}}\left(\boldsymbol{x}\right), \hat{\boldsymbol{n}}\left(\boldsymbol{x}\right)\right\rangle}{\left\|\boldsymbol{x} - \boldsymbol{s}\right\|^2}, \tag{19}$$

and the derivatives $\partial f_s/\partial \boldsymbol{\pi}$ can be computed analytically from Equations (14)-(17),

$$\frac{\partial f_s}{\partial \alpha} = \frac{\partial f_{\text{GGX}}\left[\alpha\right]\left(\hat{\boldsymbol{n}}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o\right)}{\partial \alpha}. \tag{20}$$

We used symbolic differentiation to compute this derivative, and then implemented the resulting expression in our rendering code.

Surface derivatives. In contrast to the reflectance case, differentiating the surface integral of Equation (1) of the main paper with respect to mesh vertices is complicated by the fact that the integration measure $A\left(\boldsymbol{x}\right)$ is now a function of these same mesh vertices. An additional complicating factor is the presence of the binary and discontinuous visibility terms v in the integrand. We can ameliorate the second problem by making the approximation that the visibility terms are independent of the mesh geometry. This approximation is justified by the fact that the visibility terms have non-zero derivatives only on a zero-measure part of the surface (that is, on occluding contours [13]), and is common in computer vision and graphics problems [1, 22, 14].

Under this assumption, we can directly apply the derivation in Section 3.2 of Delaunoy and Prados [4]. Then, the derivative of Equation (1) of the main paper with respect to each vertex $\boldsymbol{v}_i$ becomes:

$$\begin{aligned}
&\left.\frac{\partial I\left(t;\boldsymbol{l},\boldsymbol{s}\right)}{\partial \boldsymbol{v}_i}\right|_{\boldsymbol{v},\boldsymbol{\pi}} \\
&= \sum_{k\in J_i} \int_{\mathcal{T}_k} \overbrace{\left[g_{s1}\left(\boldsymbol{x},\hat{\boldsymbol{n}}\left(\boldsymbol{x}\right);i,k\right) + g_{s2}\left(\boldsymbol{x},\hat{\boldsymbol{n}}\left(\boldsymbol{x}\right);i,k\right)\right]}^{g_s(\boldsymbol{x},\hat{\boldsymbol{n}}(\boldsymbol{x});i,k)} \\
&\qquad \cdot v\left(\boldsymbol{x},\boldsymbol{l}\right) v\left(\boldsymbol{x},\boldsymbol{s}\right)\, \mathrm{d}A\left(\boldsymbol{x}\right),
\end{aligned} \tag{21}$$

where

$$g_{s1}\left(\boldsymbol{x},\hat{\boldsymbol{n}}\left(\boldsymbol{x}\right);i,k\right) = \nabla_{\boldsymbol{x}} g(\boldsymbol{x},\hat{\boldsymbol{n}}_k)\phi_i(\mathbf{x}), \tag{22}$$

$$g_{s2}\left(\boldsymbol{x},\hat{\boldsymbol{n}}\left(\boldsymbol{x}\right);i,k\right) = -\frac{\mathbf{e}_{k,i}}{2A_k} \wedge \left[g(\boldsymbol{x},\hat{\boldsymbol{n}}_k)\hat{\boldsymbol{n}}_k + g_{\hat{\boldsymbol{n}}}(\boldsymbol{x},\hat{\boldsymbol{n}}_k)\right]. \tag{23}$$

J_i corresponds to the set of triangles that contain vertex i. $\nabla_{\boldsymbol{x}}$ is the gradient of g with respect to $\boldsymbol{x}$, $\phi_i(\boldsymbol{x})$ is the linear interpolating basis function. $\mathbf{e}_{k,i}$ is the opposite edge of vertex i in the face k pointing counterclockwise. Operator $\wedge$ denotes the cross product. Finally,

$$g_{\hat{\boldsymbol{n}}} = \nabla_{\hat{\boldsymbol{n}}} g(\boldsymbol{x},\hat{\boldsymbol{n}}_k) - \langle \nabla_{\hat{\boldsymbol{n}}} g(\boldsymbol{x},\hat{\boldsymbol{n}}_k), \hat{\boldsymbol{n}}_k\rangle\, \hat{\boldsymbol{n}}_k. \tag{24}$$

In the following, we detail some quantities that are used in the above expressions. For simplicity, we show the expression for the case of confocal imaging ($\boldsymbol{l} = \boldsymbol{s}$). The derivation for non-confocal case is straightforward, only involves longer expressions. As defined in Equations (1) and (2) of the main paper,

$$\begin{aligned}
g(\boldsymbol{x},\hat{\boldsymbol{n}}(\boldsymbol{x})) &= W\left(\boldsymbol{x};t\right) f\left(\boldsymbol{x},\hat{\boldsymbol{n}}\left(\boldsymbol{x}\right)\right) \\
&= W\left(\boldsymbol{x};t\right) f_s\left(\hat{\boldsymbol{n}}\left(\boldsymbol{x}\right),\hat{\boldsymbol{\omega}}_{\boldsymbol{l}}\left(\boldsymbol{x}\right),\hat{\boldsymbol{\omega}}_{\boldsymbol{s}}\left(\boldsymbol{x}\right)\right) \\
&\quad \cdot \frac{\left\langle -\hat{\boldsymbol{\omega}}_{\boldsymbol{l}}\left(\boldsymbol{x}\right),\hat{\boldsymbol{n}}\left(\boldsymbol{l}\right)\right\rangle \left\langle \hat{\boldsymbol{\omega}}_{\boldsymbol{l}}\left(\boldsymbol{x}\right),\hat{\boldsymbol{n}}\left(\boldsymbol{x}\right)\right\rangle}{\left\|\boldsymbol{x}-\boldsymbol{l}\right\|^2} \\
&\quad \cdot \frac{\left\langle -\hat{\boldsymbol{\omega}}_{\boldsymbol{s}}\left(\boldsymbol{x}\right),\hat{\boldsymbol{n}}\left(\boldsymbol{s}\right)\right\rangle \left\langle \hat{\boldsymbol{\omega}}_{\boldsymbol{s}}\left(\boldsymbol{x}\right),\hat{\boldsymbol{n}}\left(\boldsymbol{x}\right)\right\rangle}{\left\|\boldsymbol{x}-\boldsymbol{s}\right\|^2}, \\
&= W\left(\boldsymbol{x};t\right) f_s\left(\hat{\boldsymbol{n}}\left(\boldsymbol{x}\right),\hat{\boldsymbol{\omega}}_{\boldsymbol{l}}\left(\boldsymbol{x}\right),\hat{\boldsymbol{\omega}}_{\boldsymbol{s}}\left(\boldsymbol{x}\right)\right) g_0(\boldsymbol{x},\hat{\boldsymbol{n}}(\boldsymbol{x})))
\end{aligned} \tag{25}$$

The temporal importance W function is an unit rectangular function. This means that the derivatation will only be non-zero on the boundary. To make g defferentiable, we substitute W with a Gaussian that has full width at half max equal to the temporal resolution of the sensor. Then,

$$\begin{aligned}
W\left(\boldsymbol{x};t\right) &= \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{(\tau(\mathbf{x})-t)^2}{2\sigma^2}) \\
\frac{\partial W\left(\boldsymbol{x};t\right)}{\partial \boldsymbol{x}} &= \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{(\tau(\mathbf{x})-t)^2}{2\sigma^2}) \frac{2[\tau(\mathbf{x})-t]}{\sigma^2} \hat{\boldsymbol{\omega}}_{\boldsymbol{l}} \\
\frac{\partial W\left(\boldsymbol{x};t\right)}{\partial \hat{\boldsymbol{n}}} &= \mathbf{0}
\end{aligned} \tag{26}$$

In addition, the terms $\nabla_{\boldsymbol{x}} g(\boldsymbol{x},\hat{\boldsymbol{n}}_j)$ and $\nabla_{\hat{\boldsymbol{n}}} g(\boldsymbol{x},\hat{\boldsymbol{n}}_j)$ can be computed through chain rule, as follows

$$\begin{aligned}
g_0(\boldsymbol{x},\hat{\boldsymbol{n}}) &= \frac{\langle \hat{\boldsymbol{\omega}}_{\boldsymbol{l}}(\boldsymbol{x}),\hat{\boldsymbol{n}}\rangle^2 \langle \hat{\boldsymbol{\omega}}_{\boldsymbol{l}}(\boldsymbol{x}),\hat{\boldsymbol{n}}(\boldsymbol{l})\rangle^2}{\|\boldsymbol{x}-\boldsymbol{l}\|^4} \\
\frac{\partial g_0(\boldsymbol{x},\hat{\boldsymbol{n}})}{\partial \boldsymbol{x}} &= \frac{2\,\langle \hat{\boldsymbol{\omega}}_{\boldsymbol{l}}(\boldsymbol{x}),\hat{\boldsymbol{n}}(\boldsymbol{l})\rangle \langle \hat{\boldsymbol{\omega}}_{\boldsymbol{l}}(\boldsymbol{x}),\hat{\boldsymbol{n}}\rangle}{\|\boldsymbol{x}-\boldsymbol{l}\|^5} \\
&\quad \cdot [\hat{\boldsymbol{n}}(\boldsymbol{l})\,\langle \hat{\boldsymbol{\omega}}_{\boldsymbol{l}}(\boldsymbol{x}),\hat{\boldsymbol{n}}\rangle + \hat{\boldsymbol{n}}\,\langle \hat{\boldsymbol{\omega}}_{\boldsymbol{l}}(\boldsymbol{x}),\hat{\boldsymbol{n}}(\boldsymbol{l})\rangle \\
&\quad +4\,\langle \hat{\boldsymbol{\omega}}_{\boldsymbol{l}}(\boldsymbol{x}),\hat{\boldsymbol{n}}(\boldsymbol{l})\rangle \langle \hat{\boldsymbol{\omega}}_{\boldsymbol{l}}(\boldsymbol{x}),\hat{\boldsymbol{n}}\rangle \hat{\boldsymbol{\omega}}_{\boldsymbol{l}}(\boldsymbol{x})] \\
\frac{\partial g_0(\boldsymbol{x},\hat{\boldsymbol{n}})}{\partial \hat{\boldsymbol{n}}} &= \frac{2\,\langle \hat{\boldsymbol{\omega}}_{\boldsymbol{l}}(\boldsymbol{x}),\hat{\boldsymbol{n}}(\boldsymbol{l})\rangle^2 \langle \hat{\boldsymbol{\omega}}_{\boldsymbol{l}}(\boldsymbol{x}),\hat{\boldsymbol{n}}\rangle}{\|\boldsymbol{x}-\boldsymbol{l}\|^4} \hat{\boldsymbol{\omega}}_{\boldsymbol{l}}(\boldsymbol{x})
\end{aligned} \tag{27}$$

Finally, the reflectace function is also a function of $\boldsymbol{x}$ and $\hat{\boldsymbol{n}}$. Notice that $\boldsymbol{\omega}_i$ and $\boldsymbol{\omega}_o$ are function of $\boldsymbol{x}$. We use the symbolic differentiation to compute the derivative with respect to $\boldsymbol{x}$ and $\hat{\boldsymbol{n}}$, respectively.

2.3. Accounting for visibility changes

Delaunoy and Prados [4] derived the gradient of a function that contains visibiliy terms for the case of surfaces represented as triangular meshes. In particular, they show how to differentiate energy functionals of the form,

$$E(\mathcal{S}) = \int_{\mathcal{S}} \langle \mathbf{g}(\boldsymbol{x},\hat{\boldsymbol{n}}(\boldsymbol{x})),\hat{\boldsymbol{n}}(\boldsymbol{x})\rangle\, v(\boldsymbol{x},\mathbf{s}_0)\, \mathrm{d}A\left(\boldsymbol{x}\right), \tag{28}$$

where $\mathbf{s}_0$ is the camera center for the line-of-sight imaging case. The visibility function checks whether a point is visible to the camera. Their derivative expression can be directly adapted to our setting in the case of confocal imaging $\boldsymbol{l} = \boldsymbol{s}$. In that case, we observe that the image formation model of Equation (1) of the main paper simplifies to

$$I\left(t;\boldsymbol{l}\right) = \int_{\mathcal{S}_{\text{NLOS}}} g\left(\boldsymbol{x},\hat{\boldsymbol{n}}\left(\boldsymbol{x}\right)\right) v\left(\boldsymbol{x},\boldsymbol{l}\right)\, \mathrm{d}A\left(\boldsymbol{x}\right), \tag{29}$$

where

$$g\left(\boldsymbol{x},\hat{\boldsymbol{n}}\left(\boldsymbol{x}\right)\right) = f_s\left(\hat{\boldsymbol{n}}\left(\boldsymbol{x}\right),\hat{\boldsymbol{\omega}}_{\boldsymbol{l}}\left(\boldsymbol{x}\right),\hat{\boldsymbol{\omega}}_{\boldsymbol{l}}\left(\boldsymbol{x}\right)\right) \cdot \frac{\left(\left\langle -\hat{\boldsymbol{\omega}}_{\boldsymbol{l}}\left(\boldsymbol{x}\right),\hat{\boldsymbol{n}}\left(\boldsymbol{l}\right)\right\rangle\right)^2 \left(\left\langle \hat{\boldsymbol{\omega}}_{\boldsymbol{l}}\left(\boldsymbol{x}\right),\hat{\boldsymbol{n}}\left(\boldsymbol{x}\right)\right\rangle\right)^2}{\left\|\boldsymbol{x}-\boldsymbol{l}\right\|^4}. \tag{30}$$

If we define

$$\mathbf{g}\left(\boldsymbol{x},\hat{\boldsymbol{n}}\left(\boldsymbol{x}\right)\right) = f_s\left(\hat{\boldsymbol{n}}\left(\boldsymbol{x}\right),\hat{\boldsymbol{\omega}}_{\boldsymbol{l}}\left(\boldsymbol{x}\right),\hat{\boldsymbol{\omega}}_{\boldsymbol{l}}\left(\boldsymbol{x}\right)\right) \cdot \frac{\left(\left\langle -\hat{\boldsymbol{\omega}}_{\boldsymbol{l}}\left(\boldsymbol{x}\right),\hat{\boldsymbol{n}}\left(\boldsymbol{l}\right)\right\rangle\right)^2 \left\langle \hat{\boldsymbol{\omega}}_{\boldsymbol{l}}\left(\boldsymbol{x}\right),\hat{\boldsymbol{n}}\left(\boldsymbol{x}\right)\right\rangle}{\left\|\boldsymbol{x}-\boldsymbol{l}\right\|^4} \cdot \hat{\boldsymbol{\omega}}_{\boldsymbol{l}}\left(\boldsymbol{x}\right), \tag{31}$$

then we can rewrite Equation (29) as

$$I\left(t;\boldsymbol{l}\right) = \int_{\mathcal{S}_{\text{NLOS}}} \left\langle \mathbf{g}\left(\boldsymbol{x},\hat{\boldsymbol{n}}\left(\boldsymbol{x}\right)\right),\hat{\boldsymbol{n}}\left(\boldsymbol{x}\right)\right\rangle v\left(\boldsymbol{x},\boldsymbol{l}\right)\, \mathrm{d}A\left(\boldsymbol{x}\right), \tag{32}$$

which now matches the form of Equation (28).

We briefly describe the derivative expression derived by Delaunoy and Prados [4] for functionals of the form of Equation (28). The derivative contains three terms, terms due to the variation of the normal, term due to the tetrahedra of the visible adjacent triangles, and the term due to the movement of the crepuscular cone. Please refer to [4] for the details.

$$\frac{\partial E(\mathcal{S})}{\partial \boldsymbol{v}_i} = \mathbf{G}_i^{\text{norm}} + \mathbf{G}_i^{\text{int}} + \mathbf{G}_i^{\text{horiz}} \tag{33}$$

$$\mathbf{G}_i^{\text{norm}} = -\sum_{k\in J_i} \frac{\mathbf{e}_{k,i}}{2A_k} \wedge \int_{\mathcal{T}_k} P_{\hat{\boldsymbol{n}}_k}(D_{\hat{\boldsymbol{n}}_k}\mathbf{g}(\mathbf{x},\mathbf{n}_k)^T\hat{\boldsymbol{n}}_k) v(\boldsymbol{x},\mathbf{s}_0) ds \tag{34}$$

$$\mathbf{G}_i^{\text{int}} = \sum_{k\in J_i} \hat{\boldsymbol{n}}_k \int_{\mathcal{T}_k} \nabla\cdot\mathbf{g}(\mathbf{x},\hat{\boldsymbol{n}}_k)\phi_i(\boldsymbol{x}) ds \tag{35}$$

$$\mathbf{G}_i^{\text{horiz}} = \sum_{\mathbf{H}_{i,k}} \frac{1}{2}\int_0^1 \left\{[p(T(\mathbf{y}(u))) - p(\mathbf{y}(u))] \left[\frac{\mathbf{y}(u)}{\|\mathbf{y}(u)\|^4} \wedge \mathbf{H}_{i,k}\right](1-u)\right\} du \tag{36}$$

$P_{\hat{\boldsymbol{n}}_k}(\cdot)$ is the projection on the othogonal plane to $\hat{\boldsymbol{n}}_k$. A_k is the area of face k. $\mathbf{H}_{i,k}$ is the vector such that $[\boldsymbol{v}_i, \boldsymbol{v}_i + \mathbf{H}_{i,k}]$ is the edge of triangle k generating the horizon. $\mathbf{y}$ corresponds to points sampled on the edge $\mathbf{H}_{i,k}$. $T(\boldsymbol{x})$ corresponds to the point located behind $\boldsymbol{x}$ in the direction of the viewpoint. $\mathbf{g}(\boldsymbol{x}) = p(\boldsymbol{x})\frac{\boldsymbol{x}}{\boldsymbol{x}_z^3}$

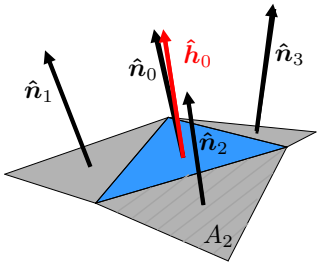


Figure 8. **Normal smoothness.** For surface smoothness regularization, we optimize vertices around a face so that the normal is aligned with the weighted normal around the face.

2.4. Surface Regularization

When optimizing geometry, we follow Delaunoy and Prados [4] and augment the loss function $E\left(\boldsymbol{v},\boldsymbol{\pi}\right)$ of Equation (7) of the main paper, with a normal smoothing regularization term

$$\mathcal{R}\left(\boldsymbol{v}\right) = \sum_{k=1}^{T} A_k \left(1 - \hat{\boldsymbol{h}}_k \cdot \hat{\boldsymbol{n}}_k\right), \tag{37}$$

where A_k and $\hat{\boldsymbol{n}}_k$ are the area and face normal, respectively, of the k-th mesh triangle, and $\hat{\boldsymbol{h}}_k$ is the weighted average of the face normals of all triangles in its neighborhood N_k,

$$\hat{\boldsymbol{h}}_k = \frac{\sum_{i\in N_k} A_i\hat{\boldsymbol{n}}_i}{\left\|\sum_{i\in N_k} A_i\hat{\boldsymbol{n}}_i\right\|}. \tag{38}$$

A visualization of this regularization term is shown in Figure 8. Its derivative with respect to mesh vertices is provided by Delaunoy and Prados [4].

3. Stochastic estimation and optimization

In this section, we provide details relating to the rendering and optimization algorithms discussed in Section 4.2 in the main paper.

3.1. Monte Carlo rendering with stratified area sampling

We first discuss the rendering algorithm we use to estimate transients $I\left(t;\boldsymbol{l},\boldsymbol{s}\right)$, and their derivatives with respect to surface, $\frac{\partial I}{\partial \boldsymbol{v}}$, and with respect to reflectance, $\frac{\partial I}{\partial \boldsymbol{\pi}}$ (Equations (1), (9), and (10) in the main paper, respectively). We present the algorithm in the context of estimating $I\left(t;\boldsymbol{l},\boldsymbol{s}\right)$, but the discussion applies exactly for the $\frac{\partial I}{\partial \boldsymbol{v}}$ and $\frac{\partial I}{\partial \boldsymbol{\pi}}$ cases. Algorithm 1 shows an overview of our rendering procedure.

We use Monte Carlo integration to approximate the surface integral of Equation (1): We first use any probability distribution μ on $\mathcal{S}_{\text{NLOS}}$ to sample a set of points $\{\boldsymbol{x}_j \in \mathcal{S}_{\text{NLOS}}, j = 1, \dots, J\}$. Then, we form an estimate as in Equation (11) of the main paper, reproduced here for convenience:

$$\langle I \rangle = \sum_{j=1}^{J} \frac{g\left(\boldsymbol{x}_j, \hat{\boldsymbol{n}}\left(\boldsymbol{x}_j\right)\right) v\left(\boldsymbol{x}_j, \boldsymbol{l}\right) v\left(\boldsymbol{x}_j, \boldsymbol{s}\right)}{\mu\left(\boldsymbol{x}_j\right)}, \tag{39}$$

In the context of light transport simulation, this is referred to as *Monte Carlo rendering* [23, 19, 5]. We note that Monte Carlo integration by sampling points $\boldsymbol{x}$ on $\mathcal{S}_{\text{NLOS}}$ is equivalent to the *area sampling* strategy described by Veach and Guibas [25]. The fact that we use area sampling is also the reason for the presence of the shading terms $\left\langle -\hat{\boldsymbol{\omega}}_{\boldsymbol{l}}\left(\boldsymbol{x}\right), \hat{\boldsymbol{n}}\left(\boldsymbol{l}\right)\right\rangle \left\langle \hat{\boldsymbol{\omega}}_{\boldsymbol{l}}\left(\boldsymbol{x}\right), \hat{\boldsymbol{n}}\left(\boldsymbol{x}\right)\right\rangle / \left\|\boldsymbol{x}-\boldsymbol{l}\right\|^2$ and $\left\langle -\hat{\boldsymbol{\omega}}_{\boldsymbol{s}}\left(\boldsymbol{x}\right), \hat{\boldsymbol{n}}\left(\boldsymbol{s}\right)\right\rangle \left\langle \hat{\boldsymbol{\omega}}_{\boldsymbol{s}}\left(\boldsymbol{x}\right), \hat{\boldsymbol{n}}\left(\boldsymbol{x}\right)\right\rangle / \left\|\boldsymbol{x}-\boldsymbol{s}\right\|^2$ in the estimate of Equation (39) (see Equation (9) in Veach and Guibas [25]).

We note that standard path sampling algorithms (e.g., path tracing [9], bidirectional path tracing [24], Metropolis light transport [26]) typically use *directional sampling* instead of area sampling. However, area sampling is advantageous in our setting for two reasons: First, when the NLOS surface $\mathcal{S}_{\text{NLOS}}$ is small compared to the LOS surface $\mathcal{S}_{\text{LOS}}$, directional sampling will result in a large number of missed rays, greatly reducing rendering efficiency. This has also been observed by Klein et al. [12], who report an order of magnitude efficiency improvement when using area sampling instead of directional sampling.

Second, area sampling facilitates *stratified sampling* [6]. We start by noting that, when $\mathcal{S}_{\text{NLOS}}$ is a triangular mesh, Equation (1) can be decomposed into a sum of per-triangle integrals,

$$I = \sum_{k=1}^{T} \int_{\mathcal{T}_k} g\left(\boldsymbol{x}, \hat{\boldsymbol{n}}\left(\boldsymbol{x}\right)\right) v\left(\boldsymbol{x}, \boldsymbol{l}\right) v\left(\boldsymbol{x}, \boldsymbol{s}\right) \, \mathrm{d}A\left(\boldsymbol{x}\right), \tag{40}$$

where $\mathcal{T}_k$ is the k-th mesh triangle, and $\mathcal{S}_{\text{NLOS}} = \bigcup_{k=1}^{T} \mathcal{T}_k$. We can then estimate I as

$$\langle I \rangle = \sum_{k=1}^{T} \sum_{j=1}^{\lceil J/T \rceil} \frac{g\left(\boldsymbol{x}_j^k, \hat{\boldsymbol{n}}\left(\boldsymbol{x}_j^k\right)\right) v\left(\boldsymbol{x}_j^k, \boldsymbol{l}\right) v\left(\boldsymbol{x}_j^k, \boldsymbol{s}\right)}{\mu\left(\boldsymbol{x}_j^k\right)}, \tag{41}$$

where for each k, $\left\{\boldsymbol{x}_j^k \in \mathcal{T}_k, j = 1, \dots, \lceil J/T \rceil\right\}$. When μ is the uniform distribution, Equation (39) uses J points $\boldsymbol{x}_j$ uniformly sampled from the entire mesh $\mathcal{S}_{\text{NLOS}}$. By contrast, Equation (41) splits the J points into T equal-size sets, with points in the k-th set uniformly sampled from a *stratum* corresponding to the triangle $\mathcal{T}_k$. This stratification procedure can signficantly reduce the variance of $\langle I \rangle$ [15, 21], by up to a factor of $1/T$ if the integrand of Equation (40) is approximately constant within each triangle—as is the case for meshes with very fine triangulation. Empirically, we observed that stratified sampling reduces variance by an order of magnitude compared to uniform sampling.

An additional advantage of stratified sampling becomes evident when we consider the visibility terms v in Equations (39) and (41). For large meshes, visibility tests can account for the bulk of the rendering computational cost. Therefore, it is critical that these tests be performed using highly-optimized libraries specifically designed for this task [28, 17]. When using stratified sampling, the ray bundles $\left\{\boldsymbol{s} \to \boldsymbol{x}_j^k\right\}$ and $\left\{\boldsymbol{l} \to \boldsymbol{x}_j^k\right\}$ for all points of the same k can be treated as *coherent ray packets* for the purposes of visibility testing [20, 27]. Empirically, we found that this results in a fourfold rendering acceleration, highlighting another advantage of our triangle-based stratification procedure.

We conclude this discussion by noting one important disadvantage of our area sampling procedure (either stratified or otherwise): When the reflectance f_s of the underlying surface becomes very specular, then the estimate of Equation (39) becomes very inefficient. This is because, when $\boldsymbol{x}_j$ is sampled uniformly on $\mathcal{S}_{\text{NLOS}}$ or a triangle $\mathcal{T}_k$, the half-vector corresponding to the path $\boldsymbol{l} \to \boldsymbol{x}_j \to \boldsymbol{s}$ will, with high probability, deviate significantly from the normal $\hat{\boldsymbol{n}}\left(\boldsymbol{x}_j\right)$. As a result, most of the sampled paths will have very low integrand values $g\left(\boldsymbol{x}_j, \hat{\boldsymbol{n}}\left(\boldsymbol{x}_j\right)\right)$. The effect of this can be observed in Section 6.1 of the main paper, when optimizing for the NLOS case under very specular BRDF (GGX with $\alpha = 0.1$). In the future, we plan to overcome this by incorporating into our framework area sampling techniques that can account for BRDF effects through half-vector importance sampling [10]. An alternative direction is to combine area and directional sampling techniques using multiple importance sampling [25].

3.2. Optimization pipeline

In this section, we describe in detail our optimization pipeline. We show an overview in Algorithm 2, and discuss below the various components and strategies we use to make convergence faster and more robust.

Stochastic gradient descent. We alternatingly optimize for reflectance and surface vertices using stochastic gradient descent (SGD). In particular, we use the Adam [11] variant of SGD, in order to take advantage of the adaptive, per-parameter, learning-rate scheduling.

When optimizing for surface vertices, Adam ordinarily would maintain $3 \cdot V$ independent learning rates, where V is the number of vertices, and each vertex is a three-dimensional vector. Empirically, we found it beneficial to constrain all 3 coordinates of the same vertex to have the same learning rate, reducing the independent learning rates to V—one per 3D vertex vector. To achieve this, we modi-

Algorithm 1 Monte Carlo rendering for estimating transient I, or its derivatives $\frac{\partial I}{\partial \boldsymbol{v}}$, $\frac{\partial I}{\partial \boldsymbol{\pi}}$.

Require: Integrand function $G \in \{g, g_s, g_r\}$.
Require: Mesh $\mathcal{S}_{\text{NLOS}} = \bigcup_{k=1}^{K} \mathcal{T}_k$.
Require: Reflectance parameters $\boldsymbol{\pi}$.
Require: Virtual source $\boldsymbol{l}$ and detector $\boldsymbol{s}$.
Require: Number of rendering samples J.

function RENDER($G, \mathcal{S}_{\text{NLOS}}, \boldsymbol{\pi}, \boldsymbol{l}, \boldsymbol{s}, J$)
  $\langle I \rangle \leftarrow 0$. ▷ Initialize estimate.
  for $k \in \{1, \ldots, K\}$ **do** ▷ Select triangle $\mathcal{T}_k$.
    ▷ Compute triangle quantities.
    $A \leftarrow \textbf{ComputeTriangleArea}(\mathcal{T}_k)$.
    $\hat{\boldsymbol{n}} \leftarrow \textbf{ComputeTriangleNormal}(\mathcal{T}_k)$.
    for $j \in \{1, \ldots, \lceil J/K \rceil\}$ **do**
      ▷ Uniformly sample a point on the triangle.
      $\boldsymbol{x} \leftarrow \textbf{SamplePointUniformly}(\mathcal{T}_k)$.
      ▷ Evaluate the function to be integrated.
      $y \leftarrow \textbf{EvaluateIntegrand}(G; \boldsymbol{\pi}, \boldsymbol{x}, \hat{\boldsymbol{n}}, \boldsymbol{l}, \boldsymbol{s})$.
      ▷ Perform visibility tests.
      $v_l \leftarrow \textbf{EvaluateVisibility}(\mathcal{S}_{\text{NLOS}}, \boldsymbol{x}, \boldsymbol{l})$.
      $v_s \leftarrow \textbf{EvaluateVisibility}(\mathcal{S}_{\text{NLOS}}, \boldsymbol{x}, \boldsymbol{s})$.
      $\langle I \rangle \leftarrow \langle I \rangle + y \cdot v_l \cdot v_s \cdot A / \lceil J/K \rceil$.
    end for
  end for
  return $\langle I \rangle$.
end function

fied Adam to update the learning rate for each vertex using the total magnitude of the 3D gradient for that vertex's coordinates. For the learning rate update rules, we refer to the original Adam publication [11].

Coarse-to-fine surface optimization. We use a coarse-to-fine scheme to accelerate the surface optimization. We begin with a mesh of a resolution equivalent to the scanning resolution on the visible wall $\mathcal{S}_{\text{LOS}}$, as produced by a volumetric reconstruction procedure. We then iterate between evolving the mesh for N gradient-descent iterations, and increasing the number of vertices by 1.25. As discussed in the main paper, to increase the number of vertices, we first use El Topo [3] to create a non-intersecting version of the evolved mesh, then perform isotropic remeshing with an increased number of target vertices.

Continuation. As discussed in the main paper and in Section 2.4 of this supplement, we use a regularized loss function for surface optimization,

$$E(\boldsymbol{v}, \boldsymbol{\pi}) + \lambda \mathcal{R}(\boldsymbol{v}). \tag{42}$$

Following common practice in regularized optimization [7], we adopt a *continuation* scheme and gradually decrease the weight λ of the regularization term: We start with an initial value λ_0, and decrease λ by a factor of 1.25 every time we increase the surface resolution. This gives more and more emphasis on the data term in stead of the regularization.

Rendering sample budgeting. We follow the so-called *increasing precision* strategy proposed by Pfeiffer and Sato [18]: As the optimization proceeds, we increase the number of samples J used for rendering the gradients with respect to surface and reflectance. Intuitively, as the optimization gets closer to a local minimum, and therefore the true gradient becomes smaller, the gradient-descent procedure becomes more sensitive to variance in the gradient estimates; therefore, reducing this variance by increasing the rendering samples can help convergence. In our implementation, we use an initial number of J_0 samples, and we increase this by a factor of 1.25 every time we alternate between surface and reflectance optimization.

4. Additional Experiments

We show additional results from experiments using synthetic data.

Quantitative evaluation of shape reconstructions. To quantify the improvement in surface reconstruction in Figure 5 of the main paper, we compare in Table 9 initialization and final reconstruction to ground-truth using two metrics: 1) the loss function of Equation (7) in the main paper; 2) the mean vertex-to-surface error. In all cases, our method provides significant improvement, often by one or more orders of magnitude. We note that mean vertex-to-surface error is not representative of the difference in resolution between the two meshes, because of the vastly different number of vertices in the initialization and final reconstructions. The following extreme example illustrates the issue: A mesh consisting of just one triangle whose three vertices happened to be exactly on the ground-truth surface would have zero error, despite not capturing the underlying shape.

Number of measurements. We compare reconstructions obtained by the technique of O'Toole et al. [16] and our procedure, using measurements from confocal scans ($\boldsymbol{l} = \boldsymbol{s}$) at 5 different scanning resolutions within the same scanning area on the visible wall $\mathcal{S}_{\text{LOS}}$. In particular, we consider resolutions: 16×16, 32×32, 64×64, 128×128, and 256×256 scan points.

Figure 10 shows the results. We note that, for the 128×128 and 256×256 cases, we found it beneficial to initialize our optimization procedure using a reconstruction obtained by applying the algorithm of O'Toole et al. [16] on only 64×64 measurements. Despite the loss in detail, this initialization provided better coverage on the NLOS surface.

We observe that, even in the case of 16×16 measured transients, our algorithm can recover some of the detail on the surface of the ground-truth mesh (e.g., texture on the

		bunny	armadillo	bear	bust	einstein	skull	soap
Loss function	init	$5.01 \cdot 10^{-4}$	$1.4 \cdot 10^{-3}$	$1.7 \cdot 10^{-3}$	$7.17 \cdot 10^{-4}$	$3.8 \cdot 10^{-3}$	$1.3 \cdot 10^{-3}$	$2.4 \cdot 10^{-3}$
	result	$1.13 \cdot 10^{-5}$	$2.3 \cdot 10^{-5}$	$2.57 \cdot 10^{-5}$	$6.49 \cdot 10^{-5}$	$9.65 \cdot 10^{-5}$	$4.31 \cdot 10^{-5}$	$2.08 \cdot 10^{-4}$
Vertex distance	init	$3.1 \cdot 10^{-3}$	$6.6 \cdot 10^{-3}$	$6 \cdot 10^{-3}$	$3.4 \cdot 10^{-3}$	$8.1 \cdot 10^{-3}$	$6.8 \cdot 10^{-3}$	$2.3 \cdot 10^{-3}$
	result	$9.95 \cdot 10^{-4}$	$3.4 \cdot 10^{-3}$	$1.2 \cdot 10^{-3}$	$1.9 \cdot 10^{-3}$	$2.2 \cdot 10^{-3}$	$2 \cdot 10^{-3}$	$3.55 \cdot 10^{-4}$

Figure 9. **Surface reconstruction metrics:** For each shape in Figure 5 of the main paper, we compare the initialization and final reconstruction to ground-truth using two metrics: 1) the loss function of Equation (7) in the main paper (*loss function*; 2) the mean vertex-to-surface error (*vertex distance*).

bunny's leg), despite the fact that the initialization has no visible details. We also see that, with just 32×32 measured transients, our algorithm can reconstruct more detail than what is possible using the volumetric reconstruction from O'Toole et al. [16] with 256×256 measured transients. That is, we can obtain a more detailed reconstruction even though we are using 64 times fewer measurements.

Noise levels. In this experiment, we evaluate the performance of our method as a function of the amount of noise contaminating our transient measurements. We use synthetic transients, and simulate SPAD noise using the method of Hernandez et al. [8]. The simulated noise contains three components, ambient noise, Poisson noise, and SPAD jitter.

Figure 11 shows reconstructions under three different noise levels, corresponding to different number of laser pulses M. Even though performance deteriorates as noise increases, we see that our proposed method can take SPAD jitter into account and is robust to ambient and Poisson noise.

We note that, as the noise level increases, the level of detail we can recover decreases. In this case, continuing to increase the spatial resolution of the mesh can be counterproductive, as the amount of noise means that gradient estimates have very high variance: We simply have not measured enough photons in order for our measurements to sufficiently regularize the surface $\mathcal{S}_{\text{NLOS}}$. This is shown in Figure 12, where we show the evolution of the mesh as we continue to upsample it at higher spatial resolutions. In practice, this implies that, for higher noise levels, we need to stop the optimization procedure at earlier resolution levels.

References

[1] Ben Appleton and Hugues Talbot. Globally minimal surfaces by continuous maximal flows. *IEEE transactions on pattern analysis and machine intelligence*, 28(1):106–118, 2006. 2

[2] James Arvo. *Analytic methods for simulated light transport*. PhD thesis, PhD thesis, Yale University, 1995. 1

[3] Tyson Brochu and Robert Bridson. El topo, 2009. https://www.cs.ubc.ca/labs/imager/tr/2009/eltopo/eltopo.html. 5

[4] Amaël Delaunoy and Emmanuel Prados. Gradient flows for optimizing triangular mesh-based surfaces: Applications to 3d reconstruction problems dealing with visibility. *IJCV*, 95(2):100–123, 2011. 2, 3

[5] Philip Dutré, Kavita Bala, and Philippe Bekaert. *Advanced global illumination*. AK Peters, Ltd., 2006. 4

[6] George Fishman. *Monte Carlo: concepts, algorithms, and applications*. Springer Science & Business Media, 1996. 4

[7] Elaine T Hale, Wotao Yin, and Yin Zhang. Fixed-point continuation for l_1-minimization: Methodology and convergence. *SIAM Journal on Optimization*, 19(3):1107–1130, 2008. 5

[8] Quercus Hernandez, Diego Gutierrez, and Adrian Jarabo. A computational model of a single-photon avalanche diode sensor for transient imaging. *arXiv*, preprint arXiv:1703.02635, 2017. 6

[9] James T Kajiya. The rendering equation. In *ACM Siggraph Computer Graphics*, volume 20, pages 143–150, 1986. 4

[10] Anton S Kaplanyan, Johannes Hanika, and Carsten Dachsbacher. The natural-constraint representation of the path space for efficient light transport simulation. *ACM Transactions on Graphics (TOG)*, 33(4):102, 2014. 4

[11] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR*, 2015. 4, 5

[12] Jonathan Klein, Martin Laurenzis, Dominik L Michels, and Matthias B Hullin. A quantitative platform for non-line-of-sight imaging problems. In *Proceedings BMVC*, pages 1–12, 2018. 4

[13] Jan J Koenderink. What does the occluding contour tell us about solid shape? *Perception*, 13(3):321–330, 1984. 2

[14] Hsueh-Ti Derek Liu, Michael Tao, and Alec Jacobson. Paparazzi: Surface editing by way of multi-view image processing. *ACM TOG*, 2018. 2

[15] Don P Mitchell. Consequences of stratified sampling in graphics. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 277–280. ACM, 1996. 4

Algorithm 2 Surface and reflectance optimization pipeline.

Require: Initial NLOS vertices $\boldsymbol{v}_0$.
Require: Initial number of NLOS vertices V_0.
Require: Initial NLOS triangles $\boldsymbol{T}_0$.
Require: Initial reflectance parameters $\boldsymbol{\pi}_0$.
Require: Initial learning rates $\boldsymbol{\eta}_{s,0}$, $\boldsymbol{\eta}_{r,0}$.
Require: Measured transient $\tilde{I}$.
Require: Virtual source $\boldsymbol{l}$ and detector $\boldsymbol{s}$.
Require: Initial regularization weight λ_0.
Require: Initial number of rendering samples J_0.
Require: Number of iterations N.

Initialization.

$\boldsymbol{v} \leftarrow \boldsymbol{v}_0$.
$V \leftarrow V_0$.
$\boldsymbol{T} \leftarrow \boldsymbol{T}_0$.
$\boldsymbol{\pi} \leftarrow \boldsymbol{\pi}_0$.
$\lambda \leftarrow \lambda_0$.
$\boldsymbol{\eta}_s \leftarrow \boldsymbol{\eta}_{s,0}$.
$\boldsymbol{\eta}_r \leftarrow \boldsymbol{\eta}_{r,0}$.
$J \leftarrow J_0$.
while not converged **do**

Gradient-descent optimization.

$\mathcal{S}_{\text{NLOS}} \leftarrow \mathbf{CreateMesh}(\boldsymbol{v}, \boldsymbol{T})$.
▷ Update reflectance.
$\boldsymbol{\pi} \leftarrow \mathbf{OptReflectance}(\boldsymbol{\pi}, \boldsymbol{\eta}_r, \mathcal{S}_{\text{NLOS}}, \tilde{I}, \boldsymbol{l}, \boldsymbol{s}, N, J)$.
▷ Update surface.
$\boldsymbol{v}\prime \leftarrow \mathbf{OptSurface}(\boldsymbol{v}, \boldsymbol{\eta}_s, \boldsymbol{\pi}, \boldsymbol{T}, \tilde{I}, \boldsymbol{l}, \boldsymbol{s}, \lambda, N, J)$.

Geometry processing.

▷ Deform mesh.
$(\boldsymbol{v}, \boldsymbol{T}) \leftarrow \mathbf{ElTopo}(\boldsymbol{v}, \boldsymbol{v}\prime, \boldsymbol{T})$.
▷ Increase mesh resolution.
$V \leftarrow V \cdot 1.25$.
$(\boldsymbol{v}, \boldsymbol{T}) \leftarrow \mathbf{IsoRemesh}(\boldsymbol{v}, \boldsymbol{T}, V)$.

Update parameters.

$\lambda \leftarrow \lambda / 1.25$. ▷ Continuation.
$J \leftarrow J \cdot 1.25$. ▷ Increasing precision.
end while

[16] Matthew O'Toole, David B Lindell, and Gordon Wetzstein. Confocal non-line-of-sight imaging based on the light-cone transform. *Nature*, 555(7696):338, 2018. 5, 6

[17] Steven G Parker, James Bigler, Andreas Dietrich, Heiko Friedrich, Jared Hoberock, David Luebke, David McAllister, Morgan McGuire, Keith Morley, Austin Robison, et al. Optix: a general purpose ray tracing engine. *ACM TOG*, 29(4):66, 2010. 4

[18] Gustavo T Pfeiffer and Yoichi Sato. On stochastic optimization methods for monte carlo least-squares problems. *arXiv preprint arXiv:1804.10079*, 2018. 5

[19] Matt Pharr, Wenzel Jakob, and Greg Humphreys. *Physically*

Algorithm 3 Adam routines for optimizing NLOS mesh vertices and reflectance.

Require: Initial NLOS vertices $\boldsymbol{v}_0$.
Require: Initial learning rates $\boldsymbol{\eta}_0$.
Require: Reflectance parameters $\boldsymbol{\pi}$.
Require: NLOS triangles $\boldsymbol{T}$.
Require: Measured transient $\tilde{I}$.
Require: Virtual source $\boldsymbol{l}$ and detector $\boldsymbol{s}$.
Require: Regularization weight λ.
Require: Number of iterations N.
Require: Number of rendering samples J.

function OptSurface($\boldsymbol{v}_0, \boldsymbol{\eta}_0, \boldsymbol{\pi}, \boldsymbol{T}, \tilde{I}, \boldsymbol{l}, \boldsymbol{s}, \lambda, N, J$)
  $\boldsymbol{v} \leftarrow \boldsymbol{v}_0$. ▷ Initialize estimate.
  $\boldsymbol{\eta} \leftarrow \boldsymbol{\eta}_0$. ▷ Initialize learning rates.
  for $n \in \{1, \ldots, N\}$ **do**
    $\mathcal{S}_{\text{NLOS}} \leftarrow \mathbf{CreateMesh}(\boldsymbol{v}, \boldsymbol{T})$.
    ▷ Render required quantities.
    $I \leftarrow \mathbf{Render}(g; \mathcal{S}_{\text{NLOS}}, \boldsymbol{\pi}, \boldsymbol{l}, \boldsymbol{s}, J)$.
    $\boldsymbol{I}_{\boldsymbol{v}} \leftarrow \mathbf{Render}(g_s; \mathcal{S}_{\text{NLOS}}, \boldsymbol{\pi}, \boldsymbol{l}, \boldsymbol{s}, J)$.
    $\boldsymbol{R}_{\boldsymbol{v}} \leftarrow \mathbf{ComputeReguGradient}(\mathcal{S}_{\text{NLOS}})$.
    ▷ Compute gradient.
    $\boldsymbol{g} \leftarrow \left(I - \tilde{I}\right) \boldsymbol{I}_{\boldsymbol{v}} + \lambda \boldsymbol{R}_{\boldsymbol{v}}$
    ▷ Perform Adam updates.
    $\boldsymbol{v} \leftarrow \mathbf{AdamUpdateParameters}(\boldsymbol{v}, \boldsymbol{\eta}, \boldsymbol{g})$
    $\boldsymbol{\eta} \leftarrow \mathbf{AdamUpdateLearningRates}(\boldsymbol{v}, \boldsymbol{\eta}, \boldsymbol{g})$
  end for
  return $\boldsymbol{v}$.
end function

Require: Initial reflectance parameters $\boldsymbol{\pi}_0$.
Require: Initial learning rates $\boldsymbol{\eta}_0$.
Require: NLOS mesh $\mathcal{S}_{\text{NLOS}}$.
Require: Measured transient $\tilde{I}$.
Require: Virtual source $\boldsymbol{l}$ and detector $\boldsymbol{s}$.
Require: Number of iterations N.
Require: Number of rendering samples J.

function OptReflectance($\boldsymbol{\pi}_0, \boldsymbol{\eta}_0, \mathcal{S}_{\text{NLOS}}, \tilde{I}, \boldsymbol{l}, \boldsymbol{s}, N, J$)
  $\boldsymbol{\pi} \leftarrow \boldsymbol{\pi}_0$. ▷ Initialize estimate.
  $\boldsymbol{\eta} \leftarrow \boldsymbol{\eta}_0$. ▷ Initialize learning rates.
  for $n \in \{1, \ldots, N\}$ **do**
    ▷ Render required quantities.
    $I \leftarrow \mathbf{Render}(g; \mathcal{S}_{\text{NLOS}}, \boldsymbol{\pi}, \boldsymbol{l}, \boldsymbol{s}, J)$.
    $\boldsymbol{I}_{\boldsymbol{\pi}} \leftarrow \mathbf{Render}(g_r; \mathcal{S}_{\text{NLOS}}, \boldsymbol{\pi}, \boldsymbol{l}, \boldsymbol{s}, J)$.
    ▷ Compute gradient.
    $\boldsymbol{g} \leftarrow \left(I - \tilde{I}\right) \boldsymbol{I}_{\boldsymbol{\pi}}$
    ▷ Perform Adam updates.
    $\boldsymbol{\pi} \leftarrow \mathbf{AdamUpdateParameters}(\boldsymbol{\pi}, \boldsymbol{\eta}, \boldsymbol{g})$
    $\boldsymbol{\eta} \leftarrow \mathbf{AdamUpdateLearningRates}(\boldsymbol{\pi}, \boldsymbol{\eta}, \boldsymbol{g})$
  end for
  return $\boldsymbol{\pi}$.
end function

based rendering: From theory to implementation. Morgan Kaufmann, 2016. 4

[20] Matt Pharr, Craig Kolb, Reid Gershbein, and Pat Hanrahan. Rendering complex scenes with memory-coherent ray tracing. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 101–108, 1997. 4

[21] Adrien Pilleboue, Gurprit Singh, David Coeurjolly, Michael Kazhdan, and Victor Ostromoukhov. Variance analysis for monte carlo integration. *ACM Transactions on Graphics (TOG)*, 34(4):124, 2015. 4

[22] Steven M Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *CVPR*, 2006. 2

[23] Eric Veach. *Robust monte carlo methods for light transport simulation*. Stanford University Stanford, 1998. 4

[24] Eric Veach and Leonidas Guibas. Bidirectional estimators for light transport. In *Photorealistic Rendering Techniques*, pages 145–167. Springer, 1995. 4

[25] Eric Veach and Leonidas J Guibas. Optimally combining sampling techniques for monte carlo rendering. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 419–428. ACM, 1995. 4

[26] Eric Veach and Leonidas J Guibas. Metropolis light transport. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 65–76. ACM Press/Addison-Wesley Publishing Co., 1997. 4

[27] Ingo Wald, Philipp Slusallek, Carsten Benthin, and Markus Wagner. Interactive rendering with coherent ray tracing. In *Computer graphics forum*, volume 20, pages 153–165. Wiley Online Library, 2001. 4

[28] Ingo Wald, Sven Woop, Carsten Benthin, Gregory S Johnson, and Manfred Ernst. Embree: a kernel framework for efficient cpu ray tracing. *ACM TOG*, 33(4):143, 2014. 4

[29] Bruce Walter, Stephen R Marschner, Hongsong Li, and Kenneth E Torrance. Microfacet models for refraction through rough surfaces. In *Proceedings of the 18th Eurographics conference on Rendering Techniques*, pages 195–206. Eurographics Association, 2007. 1

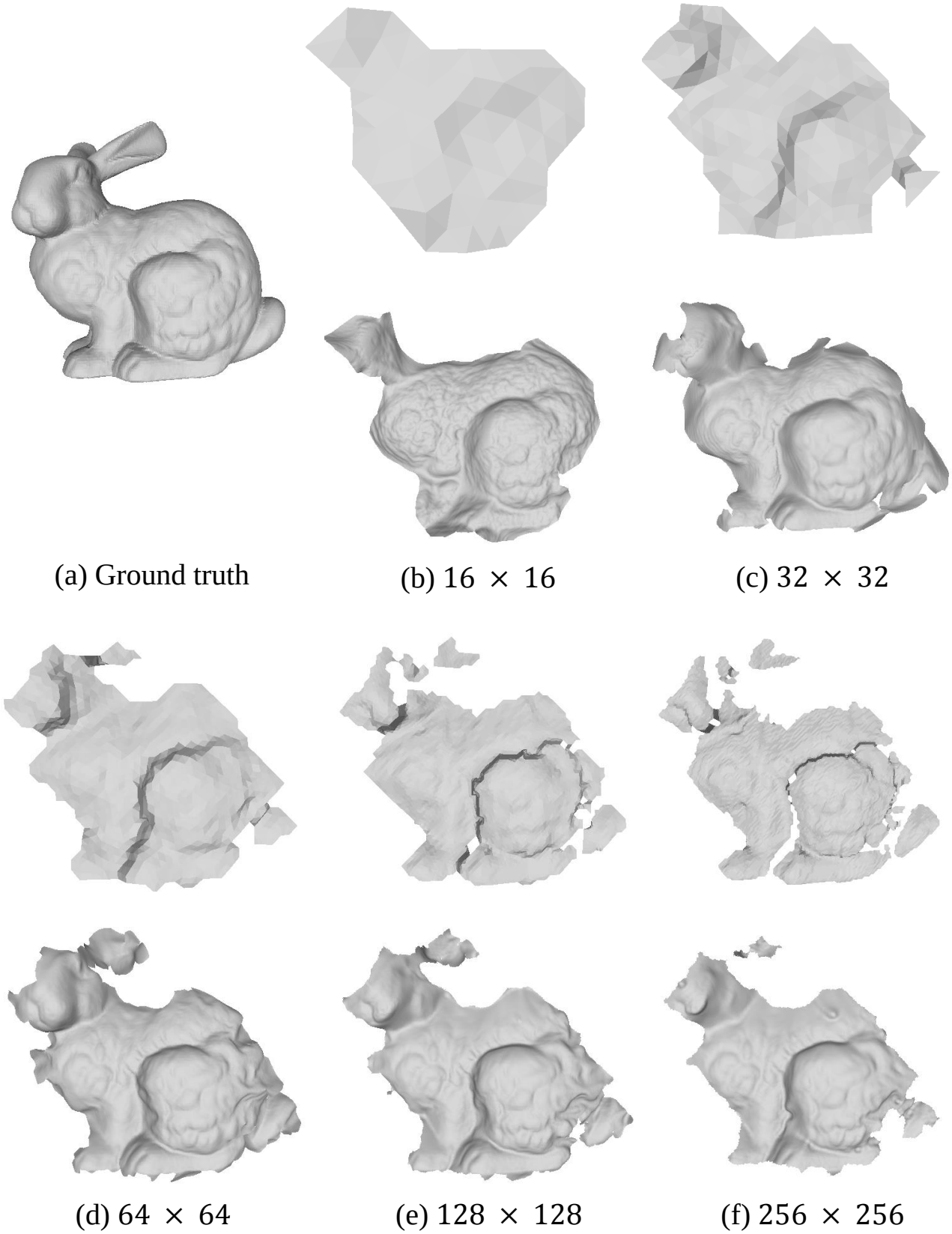
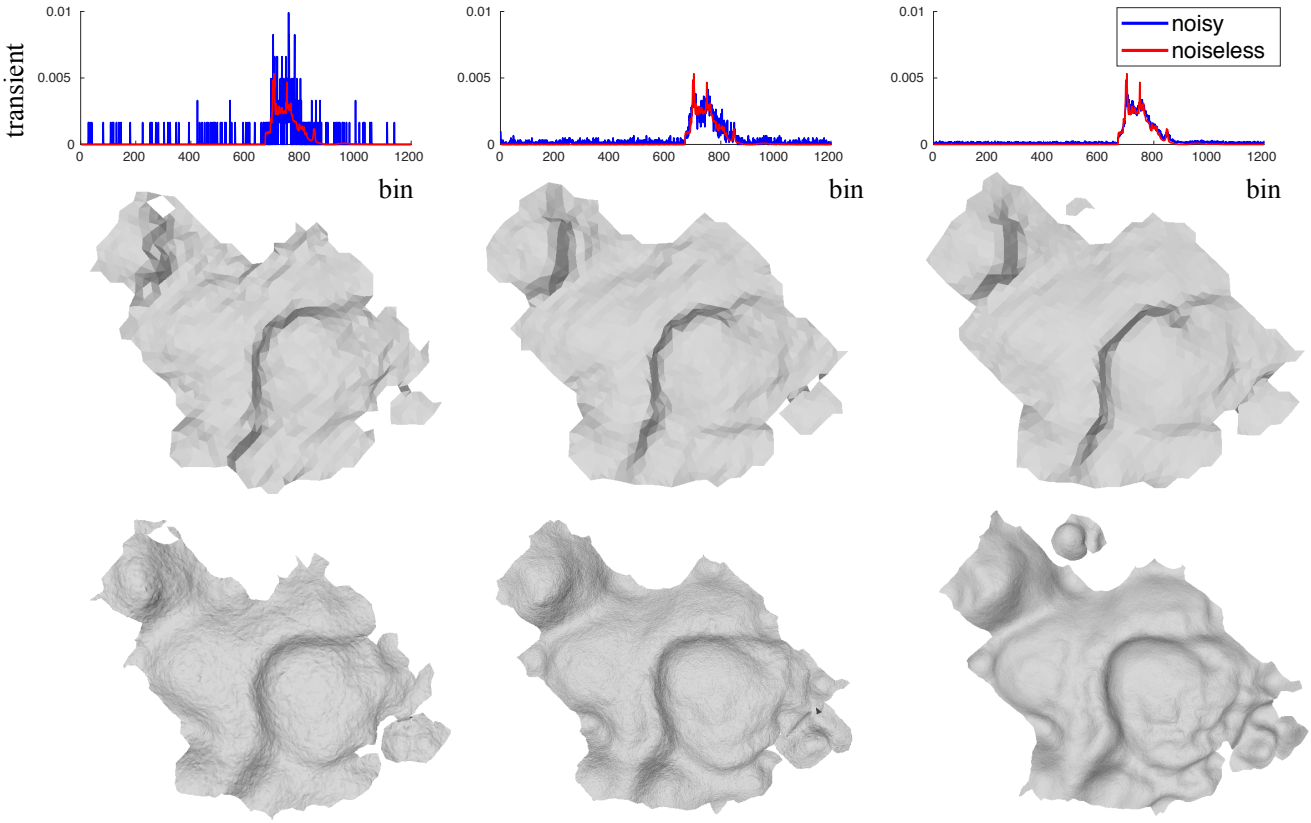


Figure 10. **Surface optimization using different number of measurements:** We perform experiments for different numbers of measured transients (i.e., scanned points on the LOS surface $\mathcal{S}_{\text{LOS}}$). When the number of measurements is very small, the initialization does not recover any discernible shape, whereas our surface optimization framework still recovers details of the ground-truth mesh. As the number of measurements increases, the level of detail of both the initialization and our reconstruction increases; in all cases our method significantly improves the final surface reconstruction.



(a) M = 200 (b) M = 2000 (c) M = 20000

Figure 11. **Noise experiment:** We simulate measurement noise by including ambient light, Poisson noise, and SPAD jitter. (Top) Sample of the noisy transients. (Middle) Initialization (Bottom) Our recovered results.

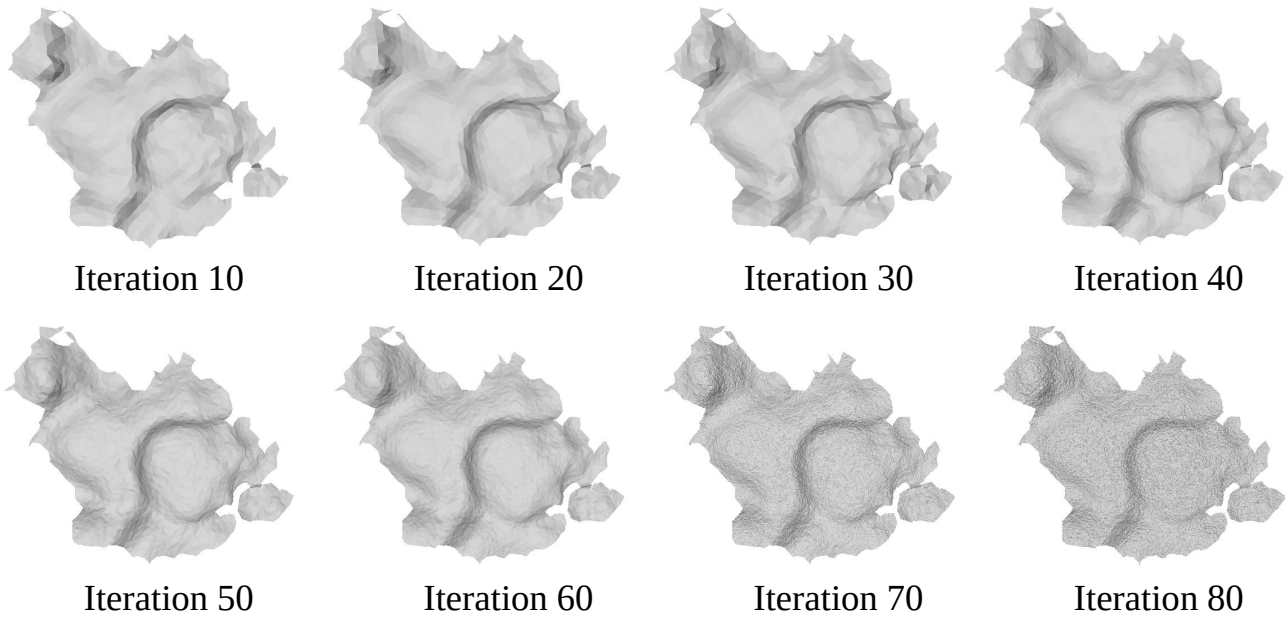


Figure 12. **Noise experiment:** We show the progression of the surface optimization for the case of the highest noise ($M = 200$). The number of mesh vertices increases with iteration. As the spatial resolution of the surface increases, the effect of noise is to reduce local surface quality, even if the global surface shape remains acceptable.