Objects as volumes: A stochastic geometry view of opaque solids

Supplementary Material

A. Volumetric representation of point clouds

In this section, we discuss how our theory relates to prior work on modeling point clouds, and how their combination enables volume rendering of point clouds.

Stochastic Poisson surface reconstruction (SPSR). Sellán and Jacobson [52] propose a stochastic model for opaque solids as excursion sets of Gaussian processes conditioned on point clouds. Using our notation, their model corresponds to Equations (24)–(25) as

$$\mathrm{v}_{\mathrm{SPSR}}(\boldsymbol{x}) = 1 - \mathrm{o}_{\mathrm{SPSR}}(\boldsymbol{x}) \equiv \Psi_{\mathrm{Gaussian}}(s(\boldsymbol{x})\,\mathrm{f}(\boldsymbol{x})), \tag{31}$$

where Ψ_{Gaussian} is the CDF for the zero-mean, unit-variance Gaussian distribution; and the mean implicit function $\mathrm{f}(\boldsymbol{x})$ and scale $s(\boldsymbol{x})$ are parameterized as functionals of a point cloud. For volume rendering, they use the occupancy $\mathrm{o}_{\mathrm{SPSR}}$ in place of the attenuation coefficient σ [52, Section 8.2]. Our theory disambiguates the occupancy and corresponding attenuation coefficient, which from Equation (16) (assuming perfectly anisotropic projected area) becomes:

$$\sigma_{\mathrm{SPSR}}(\boldsymbol{x},\boldsymbol{\omega}) \equiv \frac{\|\nabla\,\mathrm{v}_{\mathrm{SPSR}}(\boldsymbol{x})\|}{\mathrm{v}_{\mathrm{SPSR}}(\boldsymbol{x})} \cdot |\boldsymbol{\omega}\cdot\boldsymbol{n}(\boldsymbol{x})|\,. \tag{32}$$

Volume rendering of point clouds. To highlight the importance of disambiguating vacancy and the attenuation coefficient, we use single-bounce volume rendering to synthesize images of a point cloud scene from SPSR [52] under point light illumination. In Figure 7, we compare the images resulting from using the attenuation coefficient our theory predicts in Equation (32), versus using the vacancy in Equation (31) in place of the attenuation coefficient. Using the attenuation coefficient results in images that more accurately convey the appearance of the opaque solid underling the point cloud.

We use the same scene for an additional experiment that highlights the importance of enforcing reciprocity in volumetric representations of opaque solids. We consider variant of Equation (32) that replaces the perfectly anisotropic projected area with the ReLU-based one from NeuS [63]:

$$\sigma_{\mathrm{SPSR,NeuS}}(\boldsymbol{x},\boldsymbol{\omega}) \equiv \frac{\|\nabla\,\mathrm{v}_{\mathrm{SPSR}}(\boldsymbol{x})\|}{\mathrm{v}_{\mathrm{SPSR}}(\boldsymbol{x})} \cdot \mathrm{ReLU}(-\boldsymbol{\omega}\cdot\boldsymbol{n}(\boldsymbol{x}))\,. \tag{33}$$

We use the attenuation coefficients of Equations (32) and (33) to synthesize images with two volume rendering algorithms: volumetric path tracing (trace rays starting from the camera) and volumetric light tracing (trace rays starting from the light source) [57]. Under a physically-plausible, reciprocal volumetric representation, both algorithms should produce identical images. We show in Figure 7 that using our reciprocal attenuation coefficient in Equation (32) produces images that are a lot closer to each other (up to discretization and noise artifacts) than those using the non-reciprocal attenuation coefficient in Equation (33) that NeuS suggests.

B. Global illumination representation

We briefly discuss the global illumination term in the volume rendering equation (5). In *deterministic* light transport algorithms, global illumination appears as the outgoing radiance in the conservation of radiance equation. From the *reflection equation*, the outgoing radiance equals:

$$L_{\mathrm{o}}(\boldsymbol{x},\boldsymbol{\omega}_{\mathrm{o}}) = \int_{\mathcal{H}^2(\boldsymbol{x})} f_{\mathrm{r}}(\boldsymbol{x},\boldsymbol{n}(\boldsymbol{x}),\boldsymbol{\omega}_{\mathrm{o}},\boldsymbol{\omega}_{\mathrm{i}}) L_{\mathrm{i}}(\boldsymbol{x},\boldsymbol{\omega}_{\mathrm{i}}) \langle\boldsymbol{\omega}_{\mathrm{i}},\boldsymbol{n}(\boldsymbol{x})\rangle\,\mathrm{d}\boldsymbol{\omega}_{\mathrm{i}}, \tag{34}$$

where $\mathcal{H}^2(\boldsymbol{x}) \equiv \left\{\boldsymbol{\omega}\in\mathcal{S}^2 : \boldsymbol{\omega}\cdot\boldsymbol{n}_{\boldsymbol{x}} \geq 0\right\}$ is the positive hemisphere and f_{r} is the bidirectional distribution reflection function (BRDF) at point $\boldsymbol{x}$. In *volumetric* light transport algorithms, global illumination appears as the conditional expectation of the outgoing radiance in the volume rendering equation (5). Dropping the distinction between expected and actual radiance to ease notation, from the *in-scattering equation* [23, Section 3.1], the outgoing radiance equals:

$$L_{\mathrm{o}}(\boldsymbol{x},\boldsymbol{\omega}_{\mathrm{o}}) = a(\boldsymbol{x}) \int_{\mathcal{S}^2} f_{\mathrm{p}}(\boldsymbol{x},\boldsymbol{\omega}_{\mathrm{o}},\boldsymbol{\omega}_{\mathrm{i}}) L_{\mathrm{i}}(\boldsymbol{x},\boldsymbol{\omega}_{\mathrm{i}})\,\mathrm{d}\boldsymbol{\omega}_{\mathrm{i}}, \tag{35}$$

where $a\in[0,1]$ is the albedo and f_{p} is the phase function at point $\boldsymbol{x}$. Jakob et al. [25, Section 3] prove that reciprocal *exponential* transport requires that the attenuation coefficient σ and phase function f_{p} satisfy for all directions $\boldsymbol{\omega}_{\mathrm{o}},\boldsymbol{\omega}_{\mathrm{i}}$:

$$\sigma(\boldsymbol{x},\boldsymbol{\omega}_{\mathrm{o}}) f_{\mathrm{p}}(\boldsymbol{x},\boldsymbol{\omega}_{\mathrm{o}},\boldsymbol{\omega}_{\mathrm{i}}) = \sigma(\boldsymbol{x},\boldsymbol{\omega}_{\mathrm{i}}) f_{\mathrm{p}}(\boldsymbol{x},\boldsymbol{\omega}_{\mathrm{i}},\boldsymbol{\omega}_{\mathrm{o}}). \tag{36}$$

Lastly, Heitz et al. [23, Section 5] show that, when the attenuation coefficient has the form of Equation (16) in Definition 5, the phase function f_{p} relates to the BRDF f_{r} of the underlying deterministic geometry as:

$$f_{\mathrm{p}}(\boldsymbol{x},\boldsymbol{\omega}_{\mathrm{o}},\boldsymbol{\omega}_{\mathrm{i}}) = \frac{1}{\sigma_{\mathrm{D}}^{\perp}(\boldsymbol{x},\boldsymbol{\omega}_{\mathrm{o}})} \cdot \int_{\mathcal{S}^2} f_{\mathrm{r}}(\boldsymbol{x},\boldsymbol{m},\boldsymbol{\omega}_{\mathrm{o}},\boldsymbol{\omega}_{\mathrm{i}}) |\boldsymbol{\omega}_{\mathrm{o}}\cdot\boldsymbol{m}|\,\mathrm{D}_{\boldsymbol{x}}(\boldsymbol{m})\,\mathrm{d}\boldsymbol{m}. \tag{37}$$

Equation (37) expresses the phase function as the expected value of the *foreshortened* BRDF with respect to the distribution of normals D used also to define the projected area $\sigma_{\mathrm{D}}^{\perp}$ in Equation (14). We make three observations: 1. The phase function in Equation (37) satisfies the reciprocity relation in Equation (36). 2. Heitz et al. [23] derived Equation (37) in the context of microflake models for stochastic

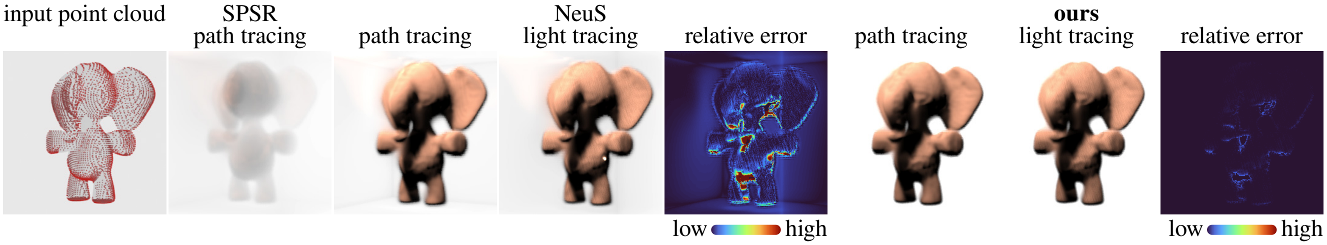

Figure 7. Comparison of point-cloud volume renderings on a scene from SPSR [52]. Using the vacancy $\mathrm{v}_{\mathrm{SPSR}}$ in place of the attenuation coefficient as SPSR suggests results in images that do not convey the appearance of an opaque solid. Using a non-reciprocal attenuation coefficient $\sigma_{\mathrm{SPSR,NeuS}}$ as NeuS suggests results in significant differences between images from different rendering algorithms. Using the reciprocal attenuation coefficient σ_{SPSR} our theory suggests results in accurate images under both rendering algorithms.

microparticle geometry. The similarity between the attenuation coefficient in our theory (Definition 5) and that in microflake models allows us to apply Equation (37) also for stochastic solid geometry. 3. As in the microflake model for stochastic microparticle geometry, we can use different BRDFs f_{r} in Equation (37) (e.g., specular or Lambertian) to derive corresponding phase functions for the stochastic solid geometry. Then, different distribution of normals D correspond to different levels of "roughness" of the underlying geometry, in analogy with the distribution of normals in microfacet BRDF models (e.g., GGX [60] or Oren-Nayar [46] models for rough specular or Lambertian, respectively, reflectance). However, reciprocity requires using the same distribution of normals for both the phase function and attenuation coefficient, and thus jointly constrains geometry and global illumination in the volume rendering equation (5).

Integrated directional encoding. Neural rendering pipelines typically represent the global illumination in Equation (5) as a neural field with positional and directional inputs. This representation makes it difficult to exactly enforce Equations (35) and (37). Verbin et al. [58] showed how to approximately do so with an *integrated directional encoding* that computes moments of directional inputs with respect to the distribution of normals describing surface roughness. Equation (37) suggests that we can combine our volumetric representation with integrated directional encoding, by using the same distribution of normals D for both.[5] This approach links the geometry and global illumination terms in the volume rendering equation (5), as reciprocity requires.

In particular, Verbin et al. [58] use a distribution of normals $\mathrm{D}_{\boldsymbol{x}}$ that equals the von-Mises Fisher distribution with center at the normal $\boldsymbol{n}(\boldsymbol{x})$ and concentration parameter $\kappa(\boldsymbol{x}) \equiv {}^{\alpha(\boldsymbol{x})}\!/_{1-\alpha(\boldsymbol{x})}$, where $\alpha(\boldsymbol{x}) \in [0,1]$ is a spatially varying anisotropy parameter.[6] Using Equation (14), the corresponding projected area equals:

$$\sigma^{\perp}_{\mathrm{vMF}}(\boldsymbol{x},\boldsymbol{\omega}) \approx \frac{\alpha(\boldsymbol{x})+1}{2}|\boldsymbol{\omega}\cdot\boldsymbol{n}(\boldsymbol{x})|^{\alpha(\boldsymbol{x})}. \qquad (38)$$

The proof of Equation (38) is exactly analogous to that by Han et al. [21, Section 7.2], and uses the same approximations as in the derivation of the equations for the integrated directional encoding [58, Equation (8), Appendix A]. It is worth comparing $\sigma^{\perp}_{\mathrm{vMF}}$ in Equation (38) to $\sigma^{\perp}_{\mathrm{mix}}$ and $\sigma^{\perp}_{\mathrm{SGGX}}$ in Equations (17) and (45): all three projected area terms become fully anisotropic and isotropic at the limit values $\alpha = 1$ and $\alpha = 0$, respectively, and produce different interpolations between these extremes at intermediate values.

Novel-view synthesis experiments. We demonstrate the importance of linking representations for geometry and global illumination in the volume rendering equation Equation (5) through *proof-of-concept* experiments on novel-view synthesis. We use Ref-NeRF [58] as a baseline, and modify its architecture to use our attenuation coefficient in Equation (26) with projected area as in Equation (38) (Figure 8). Importantly, our modification uses the same roughness ρ (equivalently, anisotropy α) for both the projected area and integrated directional encoding. The qualitative and quantitative results in Figure 9 and Tables 4 and 5 show that our modifications improve normal and appearance predictions.

Implementation details. We briefly discuss important implementation details for our multi-view synthesis experiments. We build on the Ref-NeRF implementation in the Multi-NeRF codebase [37]. We modify the normal loss to compare with the level-set normals of the vacancy, ${}^{\nabla\mathrm{v}}\!/_{\|\nabla\mathrm{v}\|}$, instead of the attenuation coefficient, ${}^{\nabla\sigma}\!/_{\|\nabla\sigma\|}$—our theory shows that the two are *not* equal (Equation (16)) and the former should better represent the underlying geometry.

Multi-NeRF takes advantage of the large memory capacity of TPUs and uses large batch sizes during training. We had access to only GPUs with 16 GB of memory, so we had to reduce the batch size from the default 2^{14} to 2^{9} rays per batch. We also followed a suggestion in the Github reposi-

[5] A caveat to this approach is that, whereas our Definition 5 and Equation (37) use D as a distribution of *normals*, the integrated directional encoding of Verbin et al. [58] uses it as a distribution of *reflected directions*. This discrepancy is analogous to the difference between the Blinn-Phong and Phong BRDFs, which randomize normals (or equivalently, half-way directions) versus reflected directions, respectively [10]. Ramamoorthi and Hanrahan [49, Appendix] show that, for the von-Mises Fisher distribution that Equation (38) and the integrated directional encoding use, this discrepancy introduces only a small approximation error.

[6] Verbin et al. [58] parameterize the von-Mises Fisher distribution using an *unbounded* roughness parameter $\rho \in [0,\infty)$, which we replace with our *bounded* anisotropy parameter $\alpha \equiv {}^{1}\!/_{\rho(\boldsymbol{x})+1}$.

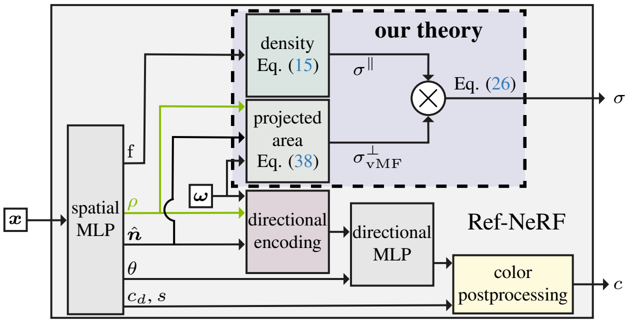

Figure 8. Visualization of our modifications to Ref-NeRF [58].

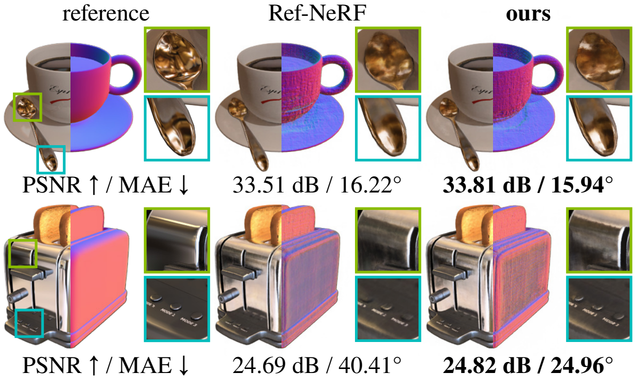

Figure 9. Qualitative results for the Shiny Blender dataset.

Table 4. PSNR values on the Shiny Blender dataset.

Shiny Blender	Ref-NeRF	**ours**
TEAPOT	**44.07**	43.35
TOASTER	24.69	**24.82**
CAR	**29.07**	28.84
BALL	32.48	**32.79**
COFFEE	33.51	**33.81**
HELMET	29.31	**30.31**
mean	32.19	**32.32**
median	30.89	**31.55**

Table 5. Mean average error (MAE) values for learned normals on the Shiny Blender dataset.

Shiny Blender	Ref-NeRF	**ours**
TEAPOT	48.48	**42.86**
TOASTER	40.41	**24.96**
CAR	21.68	**18.06**
BALL	98.94	**38.18**
COFFEE	16.22	**15.94**
HELMET	50.62	**21.46**
mean	46.06	**26.91**
median	44.45	**23.21**

tory of Multi-NeRF to reduce the learning rate when using smaller batch sizes. Instead of scheduling the learning rate to decay from 2×10^{-3} to 2×10^{-5}, we use a schedule from 5×10^{-4} to 5×10^{-6}. These settings lead to worse performance than what Ref-NeRF reports [58]. Therefore, for a fair comparison, we run Ref-NeRF with our training settings. We use the default 250 k iterations, which on an NVIDIA V100 Tensor Core GPU takes 18 hours.

C. Relationship to other prior work

Occupancy networks. Mescheder et al. [35] and Niemeyer et al. [40] propose an alternative to volume rendering for stochastic opaque solid models. They define an occupancy and vacancy model as in Equation (24) similar to NeuS,

$$\mathrm{v}_{\mathrm{ONet}}(\boldsymbol{x}) \equiv \Psi_{\mathrm{logistic}}(s\,\mathrm{f}(\boldsymbol{x})). \tag{39}$$

where the mean implicit function $\mathrm{f}(\boldsymbol{x})$ is a (trainable) neural field. However, instead of converting this occupancy to an attenuation coefficient σ for volume rendering, they define what we term the *maximum a posteriori* (MAP) opaque solid $\mathcal{O}_{\mathrm{MAP}} \equiv \left\{\boldsymbol{x} \in \mathbb{R}^3 : \mathrm{v}(x) \le 1/2\right\}$. This approach is equivalent to setting at each point $\boldsymbol{x}$ the random indicator function $\mathrm{I}(\boldsymbol{x})$ equal to its MAP estimate $\hat{\mathrm{I}}_{\mathrm{MAP}}(\boldsymbol{x}) \equiv \mathbf{1}_{\{\mathrm{v}(x)\le 1/2\}}$—hence the term MAP opaque solid.

Mescheder et al. [35] proceed to render $\mathcal{O}_{\mathrm{MAP}}$ using *deterministic* intersection queries and rendering algorithms. This approach is analogous to setting, during deterministic rendering, the free-flight distance $t^*_{\boldsymbol{x},\boldsymbol{\omega}}$ equal to its MAP estimate $t^*_{\boldsymbol{x},\boldsymbol{\omega},\mathrm{MAP}} \equiv \operatorname{argmax}_{t\in[0,+\infty)} \mathrm{p}^{\mathrm{ff}}_{\boldsymbol{x},\boldsymbol{\omega}}(t)$. By contrast, volume rendering (Equation (5)) computes the expectation over the free-flight distance $t^*_{\boldsymbol{x},\boldsymbol{\omega}}$. By using point estimation instead of expectation, the MAP approach discards uncertainty information available in $\mathrm{o}(\boldsymbol{x})$ about the scene. Additionally, recent work has demonstrated the benefits of volumetric representations for novel-view synthesis [7, 36, 58] and 3D reconstruction [45, 63, 66], motivating our focus on volumetric representations.

Discretization approaches. Another approach for deriving volumetric representations for opaque solids is to discretize a ray and define the free-flight distribution using discrete occlusion events [8, 45, 55]. We specialize this discussion to exponential transport. Along each ray, these approaches are discrete-time approximations to the continuous-time binary Markov process $\mathrm{I}_{\boldsymbol{x},\boldsymbol{\omega}}(t)$ underlying our model [42]. Alternatively, but equivalently, they correspond to a piecewise-constant approximation to the attenuation coefficient σ associated with the indicator function, as we explain below.

We consider a segment $[t_0, t_N]$ along a ray $\boldsymbol{r}_{\boldsymbol{x},\boldsymbol{\omega}}(t)$, which we break into N segments $[t_n, t_{n+1}]$, $n = 0, \dots, N-1$, such that $t_0 < t_1 < \cdots < t_N$. (These segments can have unequal lengths.) The probability that a ray will *not* terminate within segment $[t_n, t_{n+1}]$ equals:

$$\mathrm{V}_n \equiv \mathrm{T}_{\boldsymbol{x},\boldsymbol{\omega}}(t_n, t_{n+1}) \tag{40}$$

$$= \exp\left\{-\int_{t_n}^{t_{n+1}} \sigma(\boldsymbol{r}_{\boldsymbol{x},\boldsymbol{\omega}}(s), \boldsymbol{\omega})\,\mathrm{d}s\right\}, \tag{41}$$

where we used Definition 1 and expanded the notation for transmittance in Equations (1) and (6). The quantity V_n is a discrete analogue of the vacancy v of the continuous-time Markov process $\mathrm{I}_{\boldsymbol{x},\boldsymbol{\omega}}(t)$: it equals the probability that $\mathrm{I}_{\boldsymbol{x},\boldsymbol{\omega}}(t) = 0$ for all $t \in [t_n, t_{n+1}]$, and thus all points along the ray segment are vacant. We also define a discrete occupancy as $\mathrm{O}_n \equiv 1 - \mathrm{V}_n$. With this notation, assuming Markovianity, the probability that a ray terminates at segment n after traveling through the previous segments—i.e., the discrete analogue of the free-flight distribution—equals:

$$\mathrm{P}_n^{\mathrm{ff}} \equiv \mathrm{O}_n \prod_{m=0}^{n-1} \mathrm{V}_m = (1 - \mathrm{V}_n) \prod_{m=0}^{n-1} \mathrm{V}_m \,. \tag{42}$$

Lastly, if we approximate the attenuation coefficient on the segment $[t_n, t_{n+1}]$ as a constant $\bar{\sigma}_n$, then from Equation (40) the discrete vacancy becomes:

$$\mathrm{V}_n = \exp\{-\bar{\sigma}_n \cdot |t_{n+1} - tn|\}. \tag{43}$$

Equations (42)–(43) correspond exactly to the model for ray tracing through discretized opaque geometry in previous literature [8, 45, 55, 56]. As we discuss in Appendix D, they are also equivalent to the common procedure for numerically *approximating* the free-flight distribution when volume rendering continuous opaque geometry [34, 36].

Importantly, most previous discretization-based approaches model not the (piecewise-linear) attenuation coefficient $\bar{\sigma}$ in Equation (43), but directly the discrete vacancy V_n (or occupancy O_n). They do so using either a grid [8, 55, 56], or a neural field [45]—i.e., $\mathrm{V}_n := \mathrm{neural\_net}(\boldsymbol{r}_{\boldsymbol{x},\boldsymbol{\omega}}(t_n))$.

Unfortunately, directly modeling the discrete vacancy in this way does not account for the ray-segment length term $|t_{n+1} - tn|$ in Equation (43), which is problematic when considering segments of varying length (e.g., in multi-resolution approaches, or when ray discretization is non-uniform). As an example, we consider the discrete free-flight distribution $\mathrm{P}_n^{\mathrm{ff}}$ in Equation (42) for the same ray $\boldsymbol{r}_{\boldsymbol{x},\boldsymbol{\omega}}(t)$, first at discretization resolution Δt, then at a finer resolution ${}^{\Delta t}\!/_{2}$. In the latter case, $\mathrm{P}_n^{\mathrm{ff}}$ will equal the product of twice as many V_n terms as in the first case. If these terms are directly provided by, e.g., a neural field that does not account for the change in ray-segment length as resolution changes, then increasing discretization resolution results in a much lower value for the discrete free-flight distribution of the same ray. By contrast, we can elide this problem by using the neural field to output the attenuation coefficient $\bar{\sigma}_n$, then computing the vacancy terms V_n through Equation (40) while accounting for varying ray-segment lengths.

D. Implementation details

We discuss implementation details for our 3D reconstruction experiments. Our volumetric neural rendering pipeline builds on top of the NeuS pipeline [63].

Quadrature algorithm. An important algorithmic component of volumetric rendering pipelines is the numerical integration (i.e., quadrature) along a ray of the free-flight distribution and expected radiance terms in the volume rendering equation (5). Quadrature algorithms work in two stages: first, they sample locations along the ray; second, they approximate the free-flight distribution and expected radiance along the ray segments between samples. Several quadrature algorithms have appeared in both graphics [18, 27, 30, 41, 43] and vision [7, 36, 45, 58, 63, 66]. In particular, the volumetric representations in prior work we discussed in Section 4 all use their own quadrature algorithms. Unfortunately, this makes it difficult to experimentally compare different volumetric representations on equal terms: empirically, we have found that performance differences between different representations are often primarily due to the different underlying quadrature algorithms, rather than the different attenuation coefficient expressions. Therefore, to ensure a fair and informative comparison, in our experiments we use for all volumetric representations the same quadrature algorithm, which combines a sampling technique inspired by Oechsle et al. [45] with the numerical approximation by Max [34] (Equations (42)–(43)). We have found that this quadrature algorithm performs well for all representations we tested.

The sampling component of our quadrature algorithm, which we visualize in Figure 10, works as follows: We first intersect a ray with a bounding sphere of the scene. Then, we divide the ray into 1024 segments, evaluate the mean implicit function f at the endpoints of each segment, and check for sign changes between the endpoints. The free-flight distribution p^{ff} is concentrated within the segment containing the first sign change of f, and Oechsle et al. [45] suggest placing most samples in this segment. We follow their suggestion and sample a total of 64 points in three sets—one-third of the sample points within the segment with the first sign change, another third in the part of the ray before this segment, and the remaining third in the part of the ray after. If there is no segment with a sign change, we allocate the entire set of 64 samples along the entire ray. For each of the three sample sets, we select the sample locations within their corresponding ray intervals using the equidistant sampling comb approach of Kettunen et al. [27]: we deterministically place the samples at equidistant intervals, then shift them all by the same uniformly-sampled random offset.

Network architecture. Our pipeline mimics NeuS [63], using two multi-layer perceptrons (MLPs) to encode the mean implicit function and volume emission, with the addition of another small MLP to encode the anisotropy parameter.

The mean implicit function MLP has 8 hidden layers with 256 neurons per layer and $\mathrm{SoftPlus}$ activations with $\beta = 100$. A skip connection combines the output of the fourth layer with the original input. It takes as input the spatial position, and outputs the mean implicit function value and a 256-dimensional feature vector.

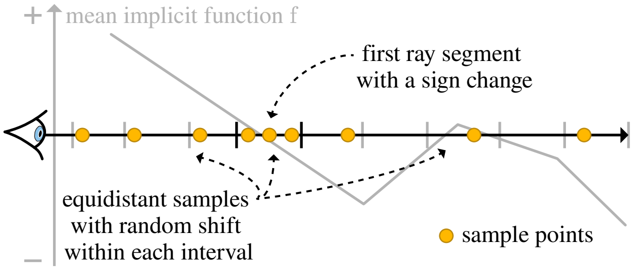

Figure 10. Visualization of our importance sampling algorithm, whoch concentrates points within the first ray segment at whose endpoints the mean implicit function f has opposite signs.

The volume emission MLP has 4 hidden layers with 256 neurons per layer and ReLU activations. It takes as input the spatial position, viewing direction, unit normal of the isosurface of the vacancy function, and the 256-dimensional feature vector from the mean implicit function MLP.

The anisotropy MLP has 1 hidden layer with 256 neurons and sigmoid activation. It takes as input the 256-dimensional feature vector from the mean implicit function MLP.

All positional and directional inputs to the MLPs use frequency encoding [36], with 6 frequencies for positional and 4 frequencies for directional inputs. As in NeuS, all linear layers use weight normalization [51] to stabilize training.

Background radiance field. To enable surface reconstruction on unbounded scenes, such as those in the BlendedMVS dataset, we follow the approach of NeuS [63] and VolSDF [66]: We perform surface reconstruction only inside a bounding sphere, and treat its exterior as a neural radiance field. We use the *inverted sphere parameterization* from NeRF++ [69] for the exterior of the sphere. To learn the background radiance field, we sample along each ray an additional 32 points on the exterior of the bounding sphere, uniformly with respect to inverse distance in the range $[0, {}^1\!/_R)$, where $R = 3$ is the bounding sphere radius. As in VolSDF, we do not learn a background radiance field for the DTU scenes.

Regularization and initialization. We follow NeuS [63] and use eikonal regularization [20], and geometric initialization [4] for the mean implicit function f.

Training details. Our training protocol is the same as in NeuS [63]. We train using the ADAM optimizer [28] for 300 k iterations with 512 rays per batch. During the first 5 k iterations, we control the learning rate with a linear warmup from 0 to 5×10^{-4}; for the remaining iterations, we use a cosine-decay schedule until the learning rate reaches a minimum of 2.5×10^{-5}. Training on an NVIDIA V100 Tensor Core GPU training takes 11 hours without the background radiance field, and 12 hours with it.

E. Additional quantitative results

We provide complete versions of Tables 2–3 in Tables 6–7 (for Table 2) and Tables 8–9 (for Table 3).

Table 6. Chamfer distances on the DTU dataset.

DTU	VolSDF	NeuS	**ours**
24	2.25	3.57	**1.92**
37	3.19	4.02	**2.35**
40	**1.94**	1.99	1.96
55	1.61	1.71	**1.11**
63	1.87	2.04	**1.83**
65	2.12	2.37	**2.01**
69	1.61	1.70	**1.30**
83	1.60	2.33	**1.53**
97	2.18	2.38	**1.62**
105	1.73	3.17	**1.50**
106	0.94	1.07	**0.71**
110	1.91	**1.09**	1.56
114	1.51	1.16	**0.83**
118	**1.36**	1.37	1.52
122	**1.74**	1.83	1.78
mean	1.84	2.17	**1.57**
median	1.74	1.99	**1.56**

In Table 9, we evaluate an additional option for the distribution of normals: We use the *SGGX distribution* that Heitz et al. propose for stochastic microparticle geometry—specifically its *surface-like* version [23, Equations (13)–(14)]. Using our notation and a reparameterization in terms of a (spatially varying) anisotropy parameter $\alpha(\boldsymbol{x}) \in [0, 1]$, this distribution and corresponding projected area equal:

$$\mathrm{D}_{\boldsymbol{x},\mathrm{SGGX}}(\boldsymbol{m}) \equiv \frac{1}{\mathrm{D}(\alpha(\boldsymbol{x}))\left(1-\alpha(\boldsymbol{x})^2|\boldsymbol{m}\cdot\boldsymbol{n}(\boldsymbol{x})|^2\right)^2}, \quad (44)$$

$$\sigma^{\perp}_{\mathrm{SGGX}}(\boldsymbol{x},\boldsymbol{\omega}) \equiv$$

$$\frac{1}{\Sigma(\alpha(\boldsymbol{x}))}\sqrt{\alpha(\boldsymbol{x})^2|\boldsymbol{\omega}\cdot\boldsymbol{n}(\boldsymbol{x})|^2 + (1-\alpha(\boldsymbol{x})^2)}\,, \quad (45)$$

with normalization coefficients

$$\mathrm{D}(\alpha) \equiv \frac{\alpha + \operatorname{arctanh}(\alpha) - \alpha^2 \operatorname{arctanh}(\alpha)}{2\alpha\pi}, \quad (46)$$

$$\Sigma(\alpha) \equiv 1 + \left(\frac{1}{\alpha} - \alpha\right)\operatorname{arcsinh}\left(\frac{\alpha}{\sqrt{1-\alpha^2}}\right). \quad (47)$$

Comparing the SGGX projected area of Equation (45) to the mixture one in Equation (17), we see that—the normalization coefficients aside—they behave similarly: $\sigma^{\perp}_{\mathrm{SGGX}}$ is a sum of squared terms, whereas $\sigma^{\perp}_{\mathrm{mix}}$ is a sum of linear terms. At the limits $\alpha = 1$ and $\alpha = 0$, they both reduce to fully anisotropic and isotropic, respectively. Heitz et al. [23] provide detailed justification for the use of the SGGX distribution in the context of stochastic microparticle geometry. In our context of stochastic solid geometry, the linear mixture performed better in our ablation study (Table 9), so we used it for the rest of our experiments.

Table 7. Chamfer distances on the NeRF Realistic Synthetic dataset.

NeRF RS	VolSDF	NeuS	**ours**
CHAIR	**0.0105**	0.0182	0.0196
LEGO	0.0354	0.0555	**0.0156**
DRUMS	0.6081	0.9897	**0.1769**
FICUS	**0.0279**	0.0796	0.0436
HOTDOG	0.1654	0.2104	**0.1295**
MATERIALS	0.0231	0.0444	**0.015**
MIC	0.9141	**0.1214**	0.4344
SHIP	0.2351	0.0901	**0.0711**
mean	0.252	0.201	**0.113**
median	0.100	0.085	**0.057**

Table 8. Chamfer distances on the DTU dataset when using different implicit function distributions Ψ for the density $\sigma^{\parallel}$.

Ψ model	logistic	Laplace	**Gaussian**
24	2.73	**1.92**	1.99
37	3.56	3.65	**3.08**
40	**1.94**	2.32	2.28
55	1.77	1.65	**1.64**
63	1.86	1.76	1.76
65	2.67	2.60	**2.45**
69	1.73	1.58	**1.31**
83	1.85	1.94	**1.69**
97	**1.82**	2.13	1.83
105	1.90	2.04	**1.74**
106	1.09	0.98	0.98
110	1.98	1.92	**1.76**
114	1.29	1.43	**0.96**
118	**1.39**	1.54	1.67
122	2.11	1.89	**1.59**
mean	1.98	1.96	**1.78**
median	1.86	1.92	**1.74**

F. Proofs

We provide proofs for our theory in Sections 3 and 4.

F.1. Exponential transport in opaque solids

We begin with the proof of Theorem 4. Our proof requires background on continuous-time discrete-space Markov processes, for which we refer to Norris [42] as one of many excellent textbooks on this subject matter. We provide pointers to specific parts throughout the proof.

To sketch the proof: The free-flight distribution $\mathrm{p}^{\mathrm{ff}}_{\boldsymbol{x},\boldsymbol{\omega}}(t)$ is the probability density that, starting from $\mathrm{I}(\boldsymbol{x}) = 0$, the first $0 \to 1$ transition of the indicator function along the ray (i.e., first intersection) happens at distance t. For this distance to be an exponential random variable, the counting process of $0 \to 1$ transitions must be a *Poisson process*. Then, the continuous-time discrete-space random process $\mathrm{I}_{\boldsymbol{x},\boldsymbol{\omega}}(t)$ must be Markovian,[7] and Equation (12) follows from the Kolmogorov equations [42] after enforcing reciprocity and reversibility.

Table 9. Chamfer distances on the DTU dataset when using different distributions of normals D for the projected area $\sigma^{\perp}$.

D	delta	delta	mixture	**mixture**	SGGX
model	(ReLU)		(const.)	**(var.)**	(var.)
24	3.57	2.73	2.43	2.16	**2.10**
37	4.02	3.56	4.16	3.40	**3.32**
40	1.99	1.94	1.94	**1.76**	1.83
55	1.71	1.77	1.85	**1.43**	1.64
63	2.04	1.86	1.85	**1.60**	1.80
65	2.37	2.67	2.19	**1.97**	2.34
69	1.70	1.73	1.57	**1.54**	1.43
83	2.33	1.85	1.79	1.55	**1.49**
97	2.38	**1.82**	2.25	1.91	2.20
105	3.17	1.90	1.85	**1.53**	1.82
106	1.07	1.09	0.99	1.32	**0.89**
110	1.90	1.98	1.89	**1.59**	1.79
114	1.16	1.29	1.37	1.26	**1.15**
118	1.37	1.39	1.75	**1.31**	1.35
122	1.83	2.11	**1.73**	1.85	1.95
mean	2.17	1.98	1.97	**1.75**	1.81
median	1.99	1.86	1.85	**1.59**	1.80

Proof. We extend Definition 1 to define the *conditional transmittance*:

$$\overline{\mathrm{T}}_{\boldsymbol{x},\boldsymbol{\omega}}(t) \equiv \Pr_{\mathcal{O}}\left\{t^*_{\boldsymbol{x},\boldsymbol{\omega}}(t) \geq t \,\middle|\, \mathrm{I}(\boldsymbol{x}) = 0\right\}, \tag{48}$$

the *conditional free-flight distribution*:

$$\overline{\mathrm{p}}^{\mathrm{ff}}_{\boldsymbol{x},\boldsymbol{\omega}}(t) \equiv -\frac{\mathrm{d}\,\overline{\mathrm{T}}_{\boldsymbol{x},\boldsymbol{\omega}}}{\mathrm{d}t}(t), \tag{49}$$

and the *conditional attenuation coefficient*:

$$\overline{\sigma}(\boldsymbol{x},\boldsymbol{\omega}) \equiv \overline{\mathrm{p}}^{\mathrm{ff}}_{\boldsymbol{x},\boldsymbol{\omega}}(0). \tag{50}$$

Under the assumption of exponential transport, the transmittance equals the conditional transmittance:

$$\mathrm{T}_{\boldsymbol{x},\boldsymbol{\omega}}(t) \overset{\text{exponential}}{=} \overline{\mathrm{T}}_{\boldsymbol{x},\boldsymbol{\omega}}(t). \tag{51}$$

and likewise for the free-flight distribution and attenuation coefficient [9, Supplementary Section 1, Equation (68)]. Thus, we work with the conditional quantities for this proof. **First jump time.** The restriction $\mathrm{I}_{\boldsymbol{x},\boldsymbol{\omega}}$ of the indicator function I along the ray $\boldsymbol{r}_{\boldsymbol{x},\boldsymbol{\omega}}$ is a

[7] Readers familiar with the derivation of exponential transport for stochastic microparticle geometry will recognize this argument: In that setting, the free-flight distribution describes the distance to the first collision with a microparticle. If microparticle locations are independent, then counting collisions is a Poisson process, and the distance to the first collision is an exponential random variable. In the stochastic solid geometry setting, we replace collisions with $0 \to 1$ transitions, and the assumption of independent particle locations with the Markovianity assumption of Equation (11).

continuous-time discrete-space (binary) stochastic process. Then, the conditional free-flight distance $t^*_{\boldsymbol{x},\boldsymbol{\omega}} \equiv \max\{t \in [0,\infty) : \mathrm{V}_{\boldsymbol{x},\boldsymbol{\omega}}(t) = 1 \,|\, \mathrm{I}_{\boldsymbol{x},\boldsymbol{\omega}}(0) = 0\}$ equals the *first jump time* $\mathrm{J}_{\boldsymbol{x},\boldsymbol{\omega}} \equiv \min\{t \in [0,\infty) : \mathrm{I}_{\boldsymbol{x},\boldsymbol{\omega}}(t) = 1 \,|\, \mathrm{I}_{\boldsymbol{x},\boldsymbol{\omega}}(0) = 0\}$ of this process; that is, the distance travelled along the ray until the *first* $0 \to 1$ transition of $\mathrm{I}_{\boldsymbol{x},\boldsymbol{\omega}}$, or equivalently until we first move from outside to inside the opaque solid [42, Section 2.2]. Consequently, the conditional free-flight distribution $\overline{\mathrm{p}}^{\mathrm{ff}}_{\boldsymbol{x},\boldsymbol{\omega}}$ equals the probability density function of $\mathrm{J}_{\boldsymbol{x},\boldsymbol{\omega}}$.

Markov property. From Equation (7), exponential transport requires that the conditional free-flight distance $t^*_{\boldsymbol{x},\boldsymbol{\omega}}$, and thus the first jump time $\mathrm{J}_{\boldsymbol{x},\boldsymbol{\omega}}$, be exponential random variables. This is equivalent to the process $\mathrm{I}_{\boldsymbol{x},\boldsymbol{\omega}}$ being *Markovian*, that is, satisfying Equation (11) [42, Section 2.6].

Transition coefficients and generator matrix. The Markov property implies that we can associate with the process $\mathrm{I}_{\boldsymbol{x},\boldsymbol{\omega}}$ a *generator matrix* [42, Section 2.8]:

$$\mathbf{Q}_{\boldsymbol{x},\boldsymbol{\omega}}(t) \equiv \begin{bmatrix} -\,\sigma^{01}_{\boldsymbol{x},\boldsymbol{\omega}}(t) & \sigma^{01}_{\boldsymbol{x},\boldsymbol{\omega}}(t) \\ \sigma^{10}_{\boldsymbol{x},\boldsymbol{\omega}}(t) & -\,\sigma^{10}_{\boldsymbol{x},\boldsymbol{\omega}}(t) \end{bmatrix}, \tag{52}$$

where $\sigma^{01}_{\boldsymbol{x},\boldsymbol{\omega}}$ and $\sigma^{10}_{\boldsymbol{x},\boldsymbol{\omega}}$ are nonnegative *transition rates*. The transition rate $\sigma^{01}_{\boldsymbol{x},\boldsymbol{\omega}}$ is the rate of the exponential distribution of the first jump time $\mathrm{J}_{\boldsymbol{x},\boldsymbol{\omega}}$, and thus we can relate it to the conditional free-flight distribution as:

$$\overline{\mathrm{p}}^{\mathrm{ff}}_{\boldsymbol{x},\boldsymbol{\omega}}(t) = \sigma^{01}_{\boldsymbol{x},\boldsymbol{\omega}}(t) \exp\left(-\int_0^t \sigma^{01}_{\boldsymbol{x},\boldsymbol{\omega}}(s)\,\mathrm{d}s\right). \tag{53}$$

Analogously, the transition rate $\sigma^{10}_{\boldsymbol{x},\boldsymbol{\omega}}$ is the rate of the exponential distribution of the first jump time for $1 \to 0$ transitions (moving from inside to outside the opaque solid). Comparing Equations (7) and (53), we see that the transition rate $\sigma^{01}_{\boldsymbol{x},\boldsymbol{\omega}}$ is the restriction of the attenuation coefficient σ along the ray $\boldsymbol{r}_{\boldsymbol{x},\boldsymbol{\omega}}$. Even though we need only $\sigma^{01}_{\boldsymbol{x},\boldsymbol{\omega}}$ and not $\sigma^{10}_{\boldsymbol{x},\boldsymbol{\omega}}$ for exponential transport, our proof characterizes both.

Kolmogorov equations. To proceed, we will use the *transition probabilities* of the process $\mathrm{I}_{\boldsymbol{x},\boldsymbol{\omega}}$ for all $i, j \in \{0,1\}$:

$$\overline{\mathrm{P}}^{ij}_{\boldsymbol{x},\boldsymbol{\omega}}(t) \equiv \Pr(\mathrm{I}_{\boldsymbol{x},\boldsymbol{\omega}}(t) = j \,|\, \mathrm{I}_{\boldsymbol{x},\boldsymbol{\omega}}(t) = i \,|\,), \tag{54}$$

and the corresponding *transition probability matrix*:

$$\overline{\mathbf{P}}_{\boldsymbol{x},\boldsymbol{\omega}}(t) \equiv \begin{bmatrix} \overline{\mathrm{P}}^{00}_{\boldsymbol{x},\boldsymbol{\omega}}(t) & \overline{\mathrm{P}}^{01}_{\boldsymbol{x},\boldsymbol{\omega}}(t) \\ \overline{\mathrm{P}}^{10}_{\boldsymbol{x},\boldsymbol{\omega}}(t) & \overline{\mathrm{P}}^{11}_{\boldsymbol{x},\boldsymbol{\omega}}(t) \end{bmatrix}. \tag{55}$$

Because $\mathrm{I}_{\boldsymbol{x},\boldsymbol{\omega}}$ is Markovian, this matrix satisfies the *Kolmogorov forward equation* [42, Theorem 2.8.2]:

$$\frac{\mathrm{d}\overline{\mathbf{P}}_{\boldsymbol{x},\boldsymbol{\omega}}}{\mathrm{d}t}(t) = \overline{\mathbf{P}}_{\boldsymbol{x},\boldsymbol{\omega}}(t) \cdot \mathbf{Q}_{\boldsymbol{x},\boldsymbol{\omega}}(t). \tag{56}$$

To proceed, we will derive a version of Equation (56) that uses the unconditional *probability matrix*:

$$\mathbf{P}_{\boldsymbol{x},\boldsymbol{\omega}}(t) \equiv \begin{bmatrix} \Pr(\mathrm{I}_{\boldsymbol{x},\boldsymbol{\omega}}(t) = 0) \\ \Pr(\mathrm{I}_{\boldsymbol{x},\boldsymbol{\omega}}(t) = 1) \end{bmatrix} = \begin{bmatrix} \mathrm{v}_{\boldsymbol{x},\boldsymbol{\omega}}(t) \\ \mathrm{o}_{\boldsymbol{x},\boldsymbol{\omega}}(t) \end{bmatrix}. \tag{57}$$

Using marginalization, we can relate $\mathbf{P}_{\boldsymbol{x},\boldsymbol{\omega}}$ and $\overline{\mathbf{P}}_{\boldsymbol{x},\boldsymbol{\omega}}$ as:

$$\mathbf{P}_{\boldsymbol{x},\boldsymbol{\omega}}(t) = \overline{\mathbf{P}}_{\boldsymbol{x},\boldsymbol{\omega}}(t)^{\top} \cdot \mathbf{P}_{\boldsymbol{x},\boldsymbol{\omega}}(0). \tag{58}$$

Differentiating with respect to distance, we have:

$$\frac{\mathrm{d}\mathbf{P}_{\boldsymbol{x},\boldsymbol{\omega}}}{\mathrm{d}t}(t) = \frac{\mathrm{d}\overline{\mathbf{P}}_{\boldsymbol{x},\boldsymbol{\omega}}}{\mathrm{d}t}(t)^{\top} \cdot \mathbf{P}_{\boldsymbol{x},\boldsymbol{\omega}}(0). \tag{59}$$

Using the Kolmogorov forward equation (56):

$$\frac{\mathrm{d}\mathbf{P}_{\boldsymbol{x},\boldsymbol{\omega}}}{\mathrm{d}t}(t) = \mathbf{Q}_{\boldsymbol{x},\boldsymbol{\omega}}(t)^{\top} \cdot \overline{\mathbf{P}}_{\boldsymbol{x},\boldsymbol{\omega}}(t)^{\top} \cdot \mathbf{P}_{\boldsymbol{x},\boldsymbol{\omega}}(0). \tag{60}$$

Lastly, using Equation (58), we arrive at what we term the *unconditional Kolmogorov equation*:

$$\frac{\mathrm{d}\mathbf{P}_{\boldsymbol{x},\boldsymbol{\omega}}}{\mathrm{d}t}(t) = \mathbf{Q}_{\boldsymbol{x},\boldsymbol{\omega}}(t)^{\top} \cdot \mathbf{P}_{\boldsymbol{x},\boldsymbol{\omega}}(t). \tag{61}$$

Relating ray to local quantities. An important property of the unconditional Kolmogorov equation (61) is that it is *local*, that is, it uses quantities defined at only one point, $\boldsymbol{r}_{\boldsymbol{x},\boldsymbol{\omega}}(t)$. In particular, even though our derivation so far has considered a ray starting at point $\boldsymbol{x}$ and traveling at direction $\boldsymbol{\omega}$, Equation (61) requires knowledge of only the direction of the ray not its history (origin $\boldsymbol{x}$ or distance t).

We will use this property to rewrite Equation (61) in terms of *local* quantities. We associate with every point $\boldsymbol{x} \in \mathbb{R}^3$ and direction $\boldsymbol{\omega} \in \mathcal{S}^2$ the *local* transition rates $\sigma^{01}(\boldsymbol{x},\boldsymbol{\omega})$ and $\sigma^{10}(\boldsymbol{x},\boldsymbol{\omega})$, generator matrix

$$\mathbf{Q}(\boldsymbol{x},\boldsymbol{\omega}) \equiv \begin{bmatrix} -\,\sigma^{01}(\boldsymbol{x},\boldsymbol{\omega}) & \sigma^{01}(\boldsymbol{x},\boldsymbol{\omega}) \\ \sigma^{10}(\boldsymbol{x},\boldsymbol{\omega}) & -\,\sigma^{10}(\boldsymbol{x},\boldsymbol{\omega}) \end{bmatrix}, \tag{62}$$

and unconditional probability matrix

$$\mathbf{P}(\boldsymbol{x}) \equiv \begin{bmatrix} \mathrm{v}(\boldsymbol{x}) \\ \mathrm{o}(\boldsymbol{x}) \end{bmatrix}. \tag{63}$$

It follows that we can relate ray to local quantities as:

$$\sigma^{01}_{\boldsymbol{x},\boldsymbol{\omega}}(t) = \sigma^{01}(\boldsymbol{r}_{\boldsymbol{x},\boldsymbol{\omega}}(t), \boldsymbol{\omega}), \tag{64}$$

$$\sigma^{10}_{\boldsymbol{x},\boldsymbol{\omega}}(t) = \sigma^{10}(\boldsymbol{r}_{\boldsymbol{x},\boldsymbol{\omega}}(t), \boldsymbol{\omega}), \tag{65}$$

$$\mathbf{Q}_{\boldsymbol{x},\boldsymbol{\omega}}(t) = \mathbf{Q}(\boldsymbol{r}_{\boldsymbol{x},\boldsymbol{\omega}}(t), \boldsymbol{\omega}), \tag{66}$$

$$\mathbf{P}_{\boldsymbol{x},\boldsymbol{\omega}}(t) = \mathbf{P}(\boldsymbol{r}_{\boldsymbol{x},\boldsymbol{\omega}}(t)). \tag{67}$$

From Equations (7), (53) and (64), we recognize the local transition rate $\sigma^{01}(\boldsymbol{x},\boldsymbol{\omega})$ as the attenuation coefficient $\sigma(\boldsymbol{x},\boldsymbol{\omega})$. Additionally, for any quantity $(*)$:

$$\frac{\mathrm{d}(*)_{\boldsymbol{x},\boldsymbol{\omega}}}{\mathrm{d}t}(t) = \boldsymbol{\omega} \cdot \nabla_{\boldsymbol{x}}(*)(\boldsymbol{r}_{\boldsymbol{x},\boldsymbol{\omega}}). \tag{68}$$

Local form of the Kolmogorov equation. Using the correspondences between ray and local quantities in Equations (64)–(68), we can rewrite the unconditional Kolmogorov equation Equation (61) in *local form*:

$$\boldsymbol{\omega} \cdot \nabla_{\boldsymbol{x}} \mathbf{P}(\boldsymbol{x}) = \mathbf{Q}(\boldsymbol{x},\boldsymbol{\omega})^{\top} \cdot \mathbf{P}(\boldsymbol{x}). \tag{69}$$

Expanding $\mathbf{Q}$ and $\mathbf{P}$ from Equations (62) and (63), and using $\mathrm{o}(\boldsymbol{x}) = 1 - \mathrm{v}(\boldsymbol{x})$, we arrive at a scalar equation that we term the *local form of the unconditional Kolmogorov equation*:

$$\begin{aligned}&\boldsymbol{\omega} \cdot \nabla \mathrm{v}(\boldsymbol{x}) = \\ &\qquad - \mathrm{v}(\boldsymbol{x})\, \sigma^{01}(\boldsymbol{x}, \boldsymbol{\omega}) + (1 - \mathrm{v}(\boldsymbol{x}))\, \sigma^{10}(\boldsymbol{x}, \boldsymbol{\omega}).\end{aligned} \tag{70}$$

Exponential transport requires that Equation (70) hold for rays passing through point $\boldsymbol{x}$ along any direction $\boldsymbol{\omega}$. In particular, if we consider a ray in direction $-\boldsymbol{\omega}$, we arrive at:

$$\begin{aligned}&- \boldsymbol{\omega} \cdot \nabla \mathrm{v}(\boldsymbol{x}) = \\ &\qquad - \mathrm{v}(\boldsymbol{x})\, \sigma^{01}(\boldsymbol{x}, -\boldsymbol{\omega}) + (1 - \mathrm{v}(\boldsymbol{x}))\, \sigma^{10}(\boldsymbol{x}, -\boldsymbol{\omega}).\end{aligned} \tag{71}$$

Enforcing reciprocity. At any point $\boldsymbol{x}$ and direction $\boldsymbol{\omega}$, Equations (70) and (71) provide two linear constraints on the four transition rates $\sigma^{01}(\boldsymbol{x}, \boldsymbol{\omega}), \sigma^{01}(\boldsymbol{x}, -\boldsymbol{\omega}), \sigma^{10}(\boldsymbol{x}, \boldsymbol{\omega}), \sigma^{10}(\boldsymbol{x}, -\boldsymbol{\omega})$ as functions of the vacancy $\mathrm{v}(\boldsymbol{\omega})$ and its directional derivative. Enforcing reciprocity provides an additional linear constraint through the requirement of Equation (8):

$$\sigma^{01}(\boldsymbol{x}, \boldsymbol{\omega}) = \sigma^{01}(\boldsymbol{x}, -\boldsymbol{\omega}). \tag{72}$$

Equations (70)–(72) form an *underdetermined* linear system on the four transition rates that, after requiring nonnegativity for the transition rates, admits the family of solutions:

$$\begin{bmatrix} \sigma^{01}(\boldsymbol{x}, \boldsymbol{\omega}) \\ \sigma^{01}(\boldsymbol{x}, -\boldsymbol{\omega}) \\ \sigma^{10}(\boldsymbol{x}, \boldsymbol{\omega}) \\ \sigma^{10}(\boldsymbol{x}, -\boldsymbol{\omega}) \end{bmatrix} = \begin{cases} \begin{bmatrix} \frac{\boldsymbol{\omega} \cdot \nabla \mathrm{v}(\boldsymbol{x}) + \tau(\boldsymbol{x}, \boldsymbol{\omega})(1 - \mathrm{v}(\boldsymbol{x}))}{\mathrm{v}(\boldsymbol{x})} \\ \frac{\boldsymbol{\omega} \cdot \nabla \mathrm{v}(\boldsymbol{x}) + \tau(\boldsymbol{x}, \boldsymbol{\omega})(1 - \mathrm{v}(\boldsymbol{x}))}{\mathrm{v}(\boldsymbol{x})} \\ \frac{2\boldsymbol{\omega} \cdot \nabla \mathrm{v}(\boldsymbol{x}) + \tau(\boldsymbol{x}, \boldsymbol{\omega})(1 - \mathrm{v}(\boldsymbol{x}))}{1 - \mathrm{v}(\boldsymbol{x})} \\ \tau(\boldsymbol{x}, \boldsymbol{\omega}) \end{bmatrix}, & \text{if } \boldsymbol{\omega} \cdot \nabla \mathrm{v}(\boldsymbol{x}) \geq 0, \\ \begin{bmatrix} \frac{-\boldsymbol{\omega} \cdot \nabla \mathrm{v}(\boldsymbol{x}) + \tau(\boldsymbol{x}, \boldsymbol{\omega})(1 - \mathrm{v}(\boldsymbol{x}))}{\mathrm{v}(\boldsymbol{x})} \\ \frac{-\boldsymbol{\omega} \cdot \nabla \mathrm{v}(\boldsymbol{x}) + \tau(\boldsymbol{x}, \boldsymbol{\omega})(1 - \mathrm{v}(\boldsymbol{x}))}{\mathrm{v}(\boldsymbol{x})} \\ \tau(\boldsymbol{x}, \boldsymbol{\omega}) \\ \frac{-2\boldsymbol{\omega} \cdot \nabla \mathrm{v}(\boldsymbol{x}) + \tau(\boldsymbol{x}, \boldsymbol{\omega})(1 - \mathrm{v}(\boldsymbol{x}))}{1 - \mathrm{v}(\boldsymbol{x})} \end{bmatrix}, & \text{if } \boldsymbol{\omega} \cdot \nabla \mathrm{v}(\boldsymbol{x}) < 0, \end{cases} \tag{73}$$

where $\tau(\boldsymbol{x}, \boldsymbol{\omega}) = \tau(\boldsymbol{x}, -\boldsymbol{\omega}) \geq 0$ is a free variable. We can verify this solution by computing the range and null space of the linear system of Equations (70)–(72). Focusing on the attenuation coefficient $\sigma \equiv \sigma^{01}$, we can write succinctly:

$$\begin{aligned}&\sigma(\boldsymbol{x}, \boldsymbol{\omega}) = \sigma(\boldsymbol{x}, -\boldsymbol{\omega}) = \\ &\qquad \frac{|\boldsymbol{\omega} \cdot \nabla \mathrm{v}(\boldsymbol{x})| + \tau(\boldsymbol{x}, \boldsymbol{\omega})(1 - \mathrm{v}(\boldsymbol{x}))}{\mathrm{v}(\boldsymbol{x})}.\end{aligned} \tag{74}$$

Enforcing reversibility. Equation (74) includes the free variable τ whose value we cannot resolve by enforcing only exponential transport (i.e., exponential free-flight distributions) and reciprocity. We can determine the value of τ by considering the following: For any two points $\boldsymbol{x}$ and $\boldsymbol{y}$, Equation (70) allows computing the vacancy v of one point from that of the other, by integrating along the linear segment connecting the two points. As the two vacancies correspond to probabilities of the same underlying random field I, we should arrive at consistent results whether we integrate in the direction from $\boldsymbol{x}$ towards $\boldsymbol{y}$, or in the reverse direction from $\boldsymbol{y}$ towards $\boldsymbol{x}$. Concretely, using the ray notation:

1. In one direction, we start at $\boldsymbol{r}_{\boldsymbol{x}, \boldsymbol{x} \rightarrow \boldsymbol{y}}(0) = \boldsymbol{x}$ with initial condition $\mathrm{v}_{\boldsymbol{x}, \boldsymbol{x} \rightarrow \boldsymbol{y}}(0) = \mathrm{v}(\boldsymbol{x})$, and use Equation (70) to integrate $\mathrm{v}_{\boldsymbol{x}, \boldsymbol{x} \rightarrow \boldsymbol{y}}(t)$ with respect to t along the ray until we reach $\boldsymbol{r}_{\boldsymbol{x}, \boldsymbol{x} \rightarrow \boldsymbol{y}}(\|\boldsymbol{y} - \boldsymbol{x}\|) = \boldsymbol{y}$. The integration result should be $\mathrm{v}_{\boldsymbol{x}, \boldsymbol{x} \rightarrow \boldsymbol{y}}(\|\boldsymbol{y} - \boldsymbol{x}\|) = \mathrm{v}(\boldsymbol{y})$.
2. In the reverse direction, we start at $\boldsymbol{r}_{\boldsymbol{y}, \boldsymbol{y} \rightarrow \boldsymbol{x}}(0) = \boldsymbol{y}$ with initial condition $\mathrm{v}_{\boldsymbol{y}, \boldsymbol{y} \rightarrow \boldsymbol{x}}(0) = \mathrm{v}(\boldsymbol{y})$, and use Equation (70) to integrate $\mathrm{v}_{\boldsymbol{y}, \boldsymbol{y} \rightarrow \boldsymbol{x}}(t)$ with respect to t along the ray until we reach $\boldsymbol{r}_{\boldsymbol{y}, \boldsymbol{y} \rightarrow \boldsymbol{x}}(\|\boldsymbol{y} - \boldsymbol{x}\|) = \boldsymbol{x}$. The integration result should be $\mathrm{v}_{\boldsymbol{y}, \boldsymbol{y} \rightarrow \boldsymbol{x}}(\|\boldsymbol{y} - \boldsymbol{x}\|) = \mathrm{v}(\boldsymbol{x})$.

We term this property *reversibility*,[8] as it corresponds to enforcing consistency between the process $\mathrm{I}_{\boldsymbol{x}, \boldsymbol{\omega}}(t)$ and its directional reverse $\mathrm{I}_{\boldsymbol{x}, -\boldsymbol{\omega}}(t)$. Reversibility along all rays is necessary for the underlying indicator function I to represent a solid, as otherwise views of this solid from different rays would have inconsistent geometry. From Equation (70), reversibility holds only if the right-hand side of this equation is proportional to the vacancy $\mathrm{v}(\boldsymbol{x})$. In turn, from Equation (73), this is only true if $\tau(\boldsymbol{x}, \boldsymbol{\omega}) \propto {}^{\mathrm{v}(\boldsymbol{x})}/_{1 - \mathrm{v}(\boldsymbol{x})}$. Setting $\tau(\boldsymbol{x}, \boldsymbol{\omega}) := 0$ produces the *minimal* transition rates that guarantee reversibility. Then, Equation (74) simplifies to:

$$\sigma(\boldsymbol{x}, \boldsymbol{\omega}) = \sigma(\boldsymbol{x}, -\boldsymbol{\omega}) = \frac{|\boldsymbol{\omega} \cdot \nabla \mathrm{v}(\boldsymbol{x})|}{\mathrm{v}(\boldsymbol{x})}. \tag{75}$$

This concludes the proof. □

F.2. Stochastic implicit geometry

We now prove Proposition 7.

Proof. Equation (25) follows from the definitions of the vacancy function and the indicator function relative to the implicit function in Definition 6:

$$\begin{aligned}\mathrm{v}(\boldsymbol{x}) &= \Pr(\mathrm{I}(\boldsymbol{x}) = 0) &(76)\\ &= \Pr(\mathrm{G}(\boldsymbol{x}) > 0) &(77)\\ &= 1 - \mathrm{cdf}_{\mathrm{G}(\boldsymbol{x})}(0) &(78)\\ &= 1 - \Psi(-s\, \mathrm{f}(\boldsymbol{x})) &(79)\\ &= \Psi(s\, \mathrm{f}(\boldsymbol{x})), &(80)\end{aligned}$$

where in the last step, we used the assumption that Ψ is symmetric. Equation (24) is a consequence of the property $\mathrm{o}(\boldsymbol{x}) = 1 - \mathrm{v}(\boldsymbol{x})$. Lastly, Equation (26) follows by combining Equations (15) and (25), and the property that the PDF ψ is the derivative of the CDF Ψ. This concludes the proof. □

[8] This property is different from the usual notion of time reversibility in continuous-time Markov processes [42, Section 3.7].

We close this subsection by adapting Proposition 7 to the case of a spatially varying scale $s(\boldsymbol{x})$. This case is required for adaptive shell representations [64] and stochastic Poisson surface reconstruction [52] (Appendix A). Equations (24) and (25) remain unchanged:

$$\mathrm{o}(\boldsymbol{x}) = \Psi(-s(\boldsymbol{x})\,\mathrm{f}(\boldsymbol{x})), \tag{81}$$

$$\mathrm{v}(\boldsymbol{x}) = \Psi(s(\boldsymbol{x})\,\mathrm{f}(\boldsymbol{x})). \tag{82}$$

Combining Equations (15) and (82) results in:

$$\sigma^{\|}(\boldsymbol{x}) = \frac{\psi(s(\boldsymbol{x})\,\mathrm{f}(\boldsymbol{x}))\|s(\boldsymbol{x})\nabla\,\mathrm{f}(\boldsymbol{x}) + \nabla s(\boldsymbol{x})\,\mathrm{f}(\boldsymbol{x})\|}{\Psi(s(\boldsymbol{x})\,\mathrm{f}(\boldsymbol{x}))} \,. \tag{83}$$

F.3. Density for logistic distribution

Lastly, we prove Equation (28). This equation follows easily from Equation (26) using the properties of the logistic CDF:

$$\begin{aligned}
\psi_{\text{logistic}}(s) &\equiv \frac{\mathrm{d}\,\Psi_{\text{logistic}}}{\mathrm{d}s}(s) && (84)\\
&= \Psi_{\text{logistic}}(s)\cdot(1-\Psi_{\text{logistic}}(s)) && (85)\\
&= \Psi_{\text{logistic}}(s)\cdot\Psi_{\text{logistic}}(-s). && (86)
\end{aligned}$$