DIFFERENTIABLE SHADOW FIELDS: RAY-CALIBRATED VISIBILITY FOR INVERSE RENDERING

Anonymous authors
Paper under double-blind review

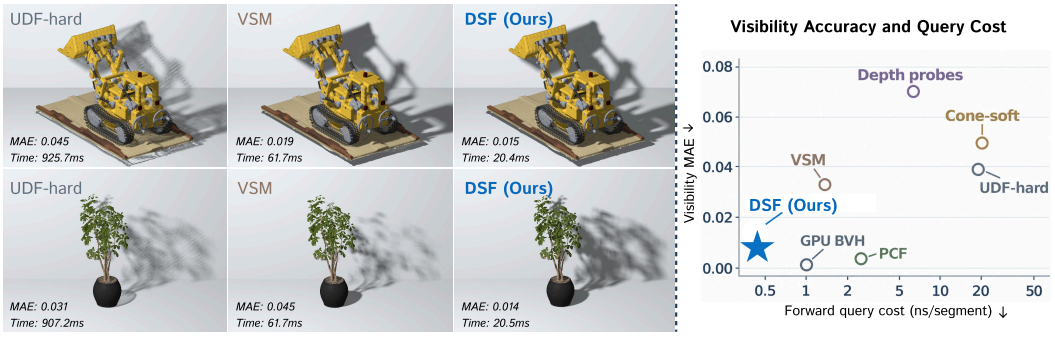


Figure 1: **Visibility as reusable scene state.** Left: Lego/Ficus shadows; Time is visibility-query time per image (ms), excluding PBR shading. Right: visibility error versus forward query cost; lower-left is better. Table 1 reports the common-segment benchmark and preparation costs.

ABSTRACT

Visibility is fundamental to rendering and inverse rendering (IR), where occlusion must be repeatedly evaluated as lighting, geometry, or scene parameters change. Ray tracing provides accurate visibility, but requires repeated geometric intersection tests and can become costly under large numbers of queries. Precomputed methods such as shadow maps are efficient, but light-dependent, update-sensitive, and not naturally differentiable with respect to light positions. We introduce Differentiable Shadow Fields (DSF), a ray-calibrated, light-independent visibility representation for IR. DSF amortizes scene geometry into a shared scalar field and evaluates visibility along finite segments between surface points and arbitrary light endpoints. Starting from an unsigned distance field, we calibrate the representation and its differentiable query function using ray-traced visibility supervision. This allows DSF to approximate ray-traced visibility while avoiding source-mesh intersection tests at inference time. DSF supports arbitrary light queries without rebuilding and remains differentiable with respect to light positions. Experiments characterize visibility accuracy and the costs of preparation, queries, and light updates, and demonstrate shadow-driven light localization, reuse across inverse-rendering pipelines, and compositional scene editing.

1 INTRODUCTION

Inverse rendering (IR) recovers materials and illumination from images, where cast shadows can be confused with dark reflectance or weak lighting. Modeling visibility helps constrain this ambiguity by relating image appearance to geometric occlusion (Zhang et al., 2021; Jin et al., 2023). Given a geometry proxy, visibility specifies whether light can reach a surface point. It therefore provides a geometric constraint for explaining shadows, rather than leaving their appearance entirely to material and lighting estimation.

Methods such as GS-ID (Du et al., 2025) jointly estimate localized lights and scene appearance. Moving a light changes both its incident directions and its occlusion across the scene. Visibility must therefore explain the observed shadows while providing useful gradients as light positions

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

change. Repeated queries make their cost important throughout optimization. A shared geometric representation can serve both light inference and subsequent relighting, and support geometry updates during scene editing.

These requirements are difficult to meet with a single visibility operator. Ray tracing resolves occlusion from geometry, but a binary intersection test alone provides no useful local gradient across shadow boundaries; differentiating visibility requires additional machinery (Li et al., 2018; Loubet et al., 2019). Shadow maps offer inexpensive lookup and have differentiable variants (Reeves et al., 1987; Worchel & Alexa, 2023), but their stored state depends on the light configuration. For effective inverse rendering, visibility should be evaluated efficiently across the scene while providing accurate gradients with respect to each queried light direction.

For efficiency, we encode scene geometry in a light-independent distance field, decoupling geometric representation from individual light configurations and enabling scalable visibility queries in complex scenes. For reliable light-direction gradients, however, the raw distance field is insufficient (as illustrated in Section 4.2): empirically, its induced visibility exhibits systematic bias and leads to inaccurate light estimation. We therefore calibrate the field directly through the final visibility query, allowing it to adapt to the evaluator rather than preserving exact metric distance.

We introduce Differentiable Shadow Fields (DSF), which represents visibility between a surface point $\boldsymbol{x}$ and a light endpoint $\boldsymbol{p}$ as $V(\boldsymbol{x}, \boldsymbol{p}) = Q(F_G; \boldsymbol{x}, \boldsymbol{p})$. We first store unsigned distances to scene geometry on a discrete 3D grid shared across light configurations, forming the geometric field D_G. Ray-traced supervision then jointly calibrates bounded corrections to the grid values and the query parameters, yielding the visibility-oriented field F_G. The evaluator Q samples and aggregates interpolated field values along the finite segment between the endpoints. At fixed grid resolution and sampling budget, queries operate on the field without traversing the source mesh. The same field supports differentiable light inference and fast forward queries for relighting. Rigid motion reuses object fields through coordinate transforms; nonrigid deformation rebuilds only the direct object field, preserving the calibrated background without additional training. Through the common endpoint-query interface Q, DSF serves as a portable visibility module for different IR methods.

We evaluate this representation through visibility prediction, physical light-position recovery, and reuse. Common-query benchmarks evaluate visibility accuracy and query cost (Figure 1), while ablations examine calibration. Integration with TensoIR, IRGS, R3DG, and LumiGauss tests the shared interface across appearance representations, while independent object fields demonstrate composition under rigid motion and nonrigid deformation.

Our contributions are:

- **Ray-calibrated finite-segment visibility.** DSF jointly calibrates a shared spatial field and its endpoint-conditioned query against ray-traced visibility, optimizing the representation for finite-segment visibility rather than metric distance reconstruction.
- **Endpoint differentiation through the spatial field.** DSF computes light-endpoint gradients through local spatial field variations along the queried segment, explicitly linking visibility differentiation to the geometry-derived representation.
- **Reuse across inverse-rendering methods.** DSF exposes geometric visibility through a common endpoint-conditioned interface, enabling its integration into inverse-rendering methods with different scene and appearance representations.

2 Related Work

Inverse rendering couples geometry, materials, and lighting through the rendering equation, with visibility encoding occlusion. NeRV and NeRFactor learn visibility for relighting (Srinivasan et al., 2021; Zhang et al., 2021); other systems use density marching, Gaussian ray tracing, baked responses, or learned transfer (Jin et al., 2023; Gu et al., 2025; Gao et al., 2024; Liang et al., 2024; Guo et al., 2024; Kaleta et al., 2025). Some jointly model shadowing, indirect illumination, and appearance (Zeng et al., 2023; Bi et al., 2024; Chen et al., 2025). SSD-GS refines visibility using geometry and material features for shading decomposition (Zheng et al., 2026); RadioGS constrains learned radiance through physically based rendering (Han et al., 2026). PTIR-GS optimizes Gaussian materials and lighting via multi-bounce path tracing (Zhu et al., 2026). GS-ID jointly estimates

local lighting and appearance using learned shadow directions for training and shadow maps for relighting (Du et al., 2025). DSF isolates geometric visibility in a shared field with a common endpoint query for light inference and relighting, independent of the host's appearance model.

Representing visibility and ray queries. Precomputed radiance transfer stores responses to changing illumination (Sloan et al., 2002), and shadow fields compose independently represented lights and occluders (Zhou et al., 2005). ComGS uses surface probes for object–scene composition and introduces SynCom (Gao et al., 2026). Neural Visibility of Point Sets predicts visibility from reusable point features and view directions (Wang et al., 2025a); NeuralPVS estimates region-level visible sets for culling (Wang et al., 2025b). Neural intersection functions and N-BVH learn ray responses from geometry (Fujieda et al., 2023; Weier et al., 2024; Fujieda et al., 2025), with extensions to parameterized deformations (Kao et al., 2026). ODFs predict first-hit distances, which can be compared with endpoint distances to answer finite visibility queries (Ying et al., 2024). These methods establish precedents for reuse and positioned queries. DSF parameterizes visibility through local scalar values shared across segments and calibrated through their final differentiable queries. Gaussian Shadow Casting and RAGA address shadows for neural characters and avatar–scene interactions (Bolanos et al., 2024; Mir et al., 2026b); DSF tests modular field reuse under rigid motion and local field rebuilding under nonrigid deformation.

Distance fields and differentiable visibility. Distance fields encode proximity to geometry (Frisken et al., 2000) and support sphere tracing and shadow approximations (Hart, 1996; Quilez, 2010; Wright, 2015; Tan et al., 2022). Differentiable sphere tracing supports implicit-surface rendering (Liu et al., 2020); edge sampling and reparameterization address visibility discontinuities (Li et al., 2018; Loubet et al., 2019; Vicini et al., 2022; Bangaru et al., 2022). Spherical-cap walks provide unbiased visibility-gradient estimators (Wu et al., 2025), and shadow-ray supervision can recover geometry (Ling et al., 2023). DSF uses distance as an initialization, then fits the field to visibility outputs. Its derivative belongs to the calibrated surrogate; we evaluate its utility through recovery from ray-traced targets.

Light-space and cached responses. Shadow maps and filters store light-dependent depth or moments (Williams, 1978; Reeves et al., 1987; Donnelly & Lauritzen, 2006; Annen et al., 2007), with differentiable variants for inverse problems (Worchel & Alexa, 2023; 2025). They support relighting by updating the light-dependent state. Lyu et al. (2021) instead approximate visibility with spherical harmonics and sphere occluders. Deep Gaussian Shadow Maps cache light-dependent volumetric transmittance (Mir et al., 2026a); neural visibility caching predicts visibility for light sampling (Bokšanský & Meister, 2025). Radiance caches approximate illumination (Müller et al., 2021; Bauer et al., 2026), while neural render proxies learn transport to image pixels for relighting with fixed cameras and materials (Sancho et al., 2026). DSF stores a geometry-derived field independent of the queried light. Geometry acceleration structures also share this property; DSF combines it with a calibrated surrogate for endpoint differentiation and reuse.

3 METHOD

Our goal is to answer a simple visibility question: given a surface point $\boldsymbol{x}$ and a light endpoint $\boldsymbol{p}$, is the finite segment between them blocked by scene geometry? We denote the corresponding soft visibility by $V(\boldsymbol{x}, \boldsymbol{p}) \in [0, 1]$, where values near one indicate a clear light path and values near zero indicate occlusion.

Vectors use bold italic lowercase symbols and matrices bold upright capitals; scalar variables are italic and descriptive labels upright. Figure 2 gives an overview of the DSF pipeline. We distinguish two field states. We first construct a geometric distance field D_G from the scene proxy, whose values encode unsigned distance to geometry. We calibrate this grid once per scene against ray-traced visibility to obtain the final DSF field F_G. The two fields share the same spatial grid but have different semantics: D_G is the direct (uncalibrated) distance field, whereas F_G contains calibrated scalar values optimized for the downstream visibility query.

Using the calibrated field, visibility is written as

$$V(\boldsymbol{x}, \boldsymbol{p}) = Q(F_G; \boldsymbol{x}, \boldsymbol{p}), \tag{1}$$

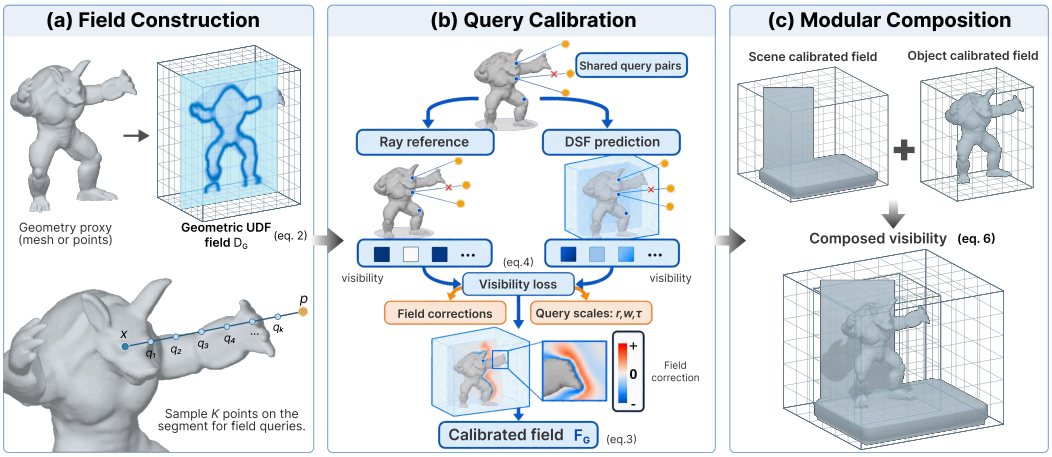


Figure 2: **DSF overview.** (a) Initialize a spatial field from geometry. (b) Match ray-traced visibility on shared segments by calibrating the field and query scales. (c) Query scene and object fields together to compose visibility without rebuilding the scene field.

where Q samples and aggregates F_G along the finite segment from $\boldsymbol{x}$ to $\boldsymbol{p}$.

3.1 Field Construction

Geometric initialization. We first construct a geometric field D_G on a dense grid covering the padded scene bounds. Each grid vertex stores its unsigned distance to the geometry proxy, forming an unsigned distance field (UDF) (Frisken et al., 2000). For any 3D location $\boldsymbol{q}$ inside the grid, the field value is obtained by trilinear interpolation, which we denote by $d_0(\boldsymbol{q}) = D_G(\boldsymbol{q})$. We use unsigned rather than signed distances because reconstructed meshes may be open or inconsistently oriented, while visibility only requires proximity to potential occluders rather than an inside–outside classification.

Finite-segment sampling. Given a surface point $\boldsymbol{x}$ and a light endpoint $\boldsymbol{p}$, only geometry along the finite segment between them can block the light. After excluding a surface clearance δ and a light clearance δ_l, we place K uniformly spaced midpoint samples $\boldsymbol{q}_1, \ldots, \boldsymbol{q}_K$ on the remaining interval (Eq. (A.1)). Segments no longer than $\delta + \delta_l$ are treated as visible. Reading the geometric field at these locations gives $d_{0,k} = D_G(\boldsymbol{q}_k)$, where smaller $d_{0,k}$ indicates that the segment passes closer to scene geometry.

3.2 Calibrating the Field through Its Queries

The geometric field D_G provides a useful prior, but accurate geometric distances do not necessarily yield accurate visibility. Without calibration, finite sampling can miss thin occluders, while interpolation and smooth aggregation can bias the effective blocking boundary. We therefore calibrate the field through its final visibility predictions rather than fitting geometric distance alone.

Visibility-oriented field. At each grid vertex $\boldsymbol{v}_i$, we introduce a trainable raw parameter ρ_i and define the calibrated value

$$f_i^{\text{grid}} = \max\left(0,\, d_i^{\text{base}} + \frac{\ell}{2}\tanh\rho_i\right), \qquad d_i^{\text{base}} = D_G(\boldsymbol{v}_i). \tag{2}$$

Here, ℓ is the voxel-cell diagonal; i indexes grid vertices, whereas k indexes segment samples. The continuous field $F_G(\boldsymbol{q})$ is obtained by trilinearly interpolating these corrected vertex values f_i^{grid}. The correction is limited to half a voxel-cell diagonal, preserving geometry as a strong prior while allowing the field to compensate for errors introduced by discretization and segment evaluation. After calibration, F_G is no longer required to represent exact metric distance.

For a segment sample $\boldsymbol{q}_k$ inside the grid, we read $f_k = F_G(\boldsymbol{q}_k) = \text{TriLerp}(\{f_i^{\text{grid}}\}; \boldsymbol{q}_k)$, where TriLerp interpolates the corrected values at the eight vertices of the containing cell. With $\boldsymbol{f} =$

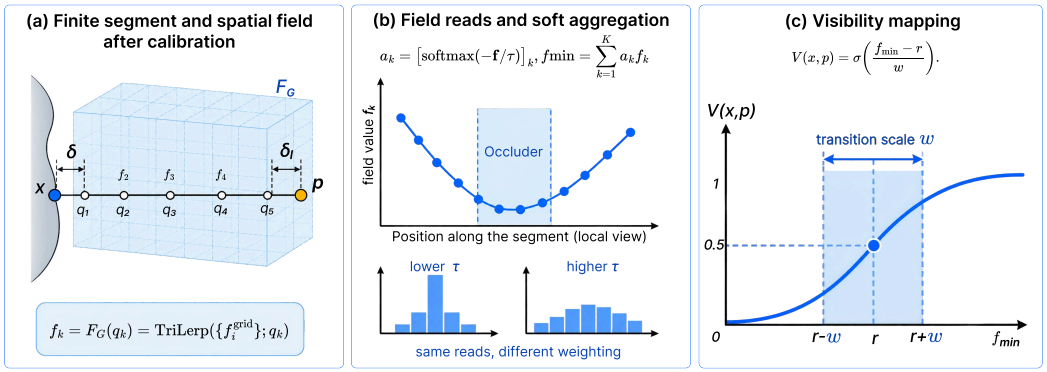


Figure 3: **From field samples to visibility.** (a) Read field values along the segment, with endpoint clearances δ and δ_l. (b) Emphasize low values near occluders; τ controls weighting. (c) Map $f_{\min}$ to visibility using threshold r and transition width w.

$(f_1, \ldots, f_K)^\top$, we aggregate the sample values and map the result to soft visibility:

$$a_k = \left[\mathrm{softmax}(-\boldsymbol{f}/\tau)\right]_k, \qquad f_{\min} = \sum_{k=1}^{K} a_k f_k, \qquad V(\boldsymbol{x}, \boldsymbol{p}) = \sigma\left(\frac{f_{\min} - r}{w}\right). \tag{3}$$

Here, σ is the logistic sigmoid; $\tau > 0$ controls the softness of the minimum, r the visibility threshold, and $w > 0$ the transition width. These positive scales are parameterized by Eq. (A.3). Figure 3 illustrates the sampling, aggregation, and visibility mapping.

Visibility supervision. For sampled surface–light pairs $(\boldsymbol{x}, \boldsymbol{p})$, we obtain binary visibility $V_{\mathrm{ray}}(\boldsymbol{x}, \boldsymbol{p})$ from ray tracing on the same declared segment, including the same endpoint clearances, and optimize

$$\mathcal{L}_{\mathrm{cal}} = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{p})} \left| V(\boldsymbol{x}, \boldsymbol{p}) - V_{\mathrm{ray}}(\boldsymbol{x}, \boldsymbol{p}) \right| + \frac{\lambda}{N} \sum_{i=1}^{N} |\rho_i|. \tag{4}$$

$\lambda \geq 0$ weights the correction penalty, N is the number of grid vertices, and $\boldsymbol{\rho} = (\rho_1, \ldots, \rho_N)^\top$. Gradients through the visibility query jointly update the local field parameters $\boldsymbol{\rho}$ and the query parameters (r, w, τ). Sampling details and the training recipe are provided in Appendix A.1.

Illumination queries. Point and moving lights are queried directly through their endpoints. Area lights are evaluated using multiple sampled endpoints, while distant illumination is converted to a finite endpoint at the scene bounding-box exit (Eq. (A.4)).

3.3 Endpoint Gradients and Modular Composition

The calibrated visibility query supports two useful capabilities beyond forward prediction: gradients with respect to the light endpoint and composition across independently constructed fields.

Endpoint gradients. With $\boldsymbol{x}$, the field, query scales, and clearances fixed, moving $\boldsymbol{p}$ changes the sample positions, yielding an almost-everywhere light-endpoint derivative for $\|\boldsymbol{p} - \boldsymbol{x}\| > \delta + \delta_l$:

$$\frac{\partial V}{\partial \boldsymbol{p}} = \frac{V(1-V)}{w} \sum_{k=1}^{K} a_k \left(1 + \frac{f_{\min} - f_k}{\tau}\right) \nabla F_G(\boldsymbol{q}_k)^\top \frac{\partial \boldsymbol{q}_k}{\partial \boldsymbol{p}}. \tag{5}$$

$\partial V / \partial \boldsymbol{p}$ is a row derivative, ∇F_G is a column vector, and $\partial \boldsymbol{q}_k / \partial \boldsymbol{p}$ is the 3×3 Jacobian. The Jacobian accounts for the change in segment direction and length as the light moves; its closed-form expression is given in Eq. (A.2). This derivative belongs to the smooth DSF visibility surrogate rather than binary geometric visibility. Unlike direct endpoint regression with Hash-MLP (Müller et al., 2022), DSF obtains endpoint gradients through spatial field gradients along the queried segment. Geometry therefore structures both visibility prediction and its differentiation. We evaluate whether it provides useful physical gradients through shadow-driven light localization in Section 4.2.

Table 1: **Visibility accuracy and reuse cost.** Four scenes (0.31–1.42M triangles), 72 point lights and 16 area emitters. Point/Area MAE: mean absolute visibility error for point/area lights against ray tracing. Occ. IoU: overlap of predicted/reference occluded queries (visibility < 0.5). Query: time for 16,384 point-light segments. Prep.: one-time scene preparation, including calibration. Update: state rebuilding when a light moves. 18 lights: estimated update and query time across 18 positions, excluding preparation. Ray tracing provides the reference. PCF: percentage-closer filtering.

Operator	Point MAE ↓	Occ. IoU ↑	Area MAE ↓	Query/batch (μs)	Prep. (s)	Update (ms)	18 lights (ms)
Ray tracing (GPU BVH)	0.0000	1.000	0.0000	636.8	0.16	0.00	11.46
Shadow maps (PCF) (Reeves et al., 1987)	0.0041	0.981	0.0025	486.7	0.00	9.72	183.66
Hard distance (Hart, 1996)	0.0224	0.908	0.0245	3320.6	78.86	0.00	59.77
VSM (Donnelly & Lauritzen, 2006)	0.0102	0.891	0.0076	120.1	0.01	11.37	206.78
Cone-soft (Quilez, 2010)	0.0296	0.758	0.0243	3499.8	78.86	0.00	63.00
Density (Mildenhall et al., 2020)	0.0424	0.853	0.0510	3577.2	78.86	0.00	64.39
Hash-MLP (Müller et al., 2022)	0.0274	0.810	0.0201	197.4	243.49	0.00	3.55
DSM (Worchel & Alexa, 2023)	0.0104	0.886	0.0078	119.0	0.01	13.51	245.27
Moment bounds (Worchel & Alexa, 2025)	0.0112	0.885	0.0084	140.4	0.01	17.01	308.76
DSF Texture (ours)	0.0054	0.961	0.0037	101.2	859.11	0.00	1.82

Modular composition. Independently constructed fields can be combined without merging their grids. For opaque geometry, a light path is visible only if it is visible with respect to every component. We compose the soft predictions using the corresponding minimum. For a rigid coordinate map $T(\boldsymbol{q}) = \mathbf{R}\boldsymbol{q} + \boldsymbol{t}$, with rotation $\mathbf{R}$ and translation $\boldsymbol{t}$,

$$V_{\text{full}}(\boldsymbol{x}, \boldsymbol{p}) = \min\big(V_{\text{scene}}(\boldsymbol{x}, \boldsymbol{p}), V_{\text{obj}}\big(T^{-1}(\boldsymbol{x}), T^{-1}(\boldsymbol{p})\big)\big)\,. \tag{6}$$

The inverse map $T^{-1}(\boldsymbol{q}) = \mathbf{R}^\top(\boldsymbol{q} - \boldsymbol{t})$ transforms the query segment into the object's local coordinate system, allowing its field to be evaluated without rebuilding. Each component retains its own grid and calibration.

For rigid motion, only T changes, so both scene and object fields are reused. For nonrigid deformation, we rebuild only the object's direct field from the deformed geometry while keeping the calibrated background fixed; no additional training is required (Figure 6).

3.4 Execution Backends

DSF currently exposes three execution backends. Fixed is a reference differentiable implementation of Eq. (3), used to evaluate endpoint gradients and inverse optimization. It has not been aggressively optimized for GPU throughput. Sphere and Texture are forward-oriented implementations; Texture exploits hardware-filtered 3D sampling for high-throughput visibility evaluation; execution details are given in Appendix A.1.3.

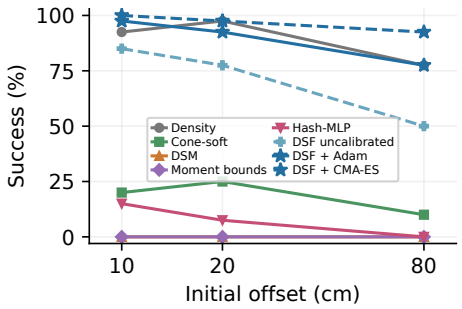
4 Experiments

We evaluated visibility prediction, inverse inference, and representation reuse on four FIPT indoor scenes (Wu et al., 2023) and four TensoIR-Synthetic scenes. External validation used Stanford-ORB scans (Kuang et al., 2023) and SynCom compositions (Gao et al., 2026). Unless varied explicitly, standalone DSF used a 1024^3 grid calibrated once for 10k updates before evaluation.

4.1 Visibility Accuracy and Query Cost

We tested prediction accuracy on common finite segments, then measured the cost of reusing each representation as the light changes.

Accuracy on common queries. Table 1 compares 72 held-out point lights and 16 area emitters across four scenes, using identical query segments and emitter samples against a shared ray reference. Differentiable shadow mapping (DSM) and moment bounds use official implementations (Worchel & Alexa, 2023; 2025); query parameters are selected on separate validation lights.



Operator / optimizer	Success (%)	Median (cm)
Density 2020	89.2	2.0
Cone-soft 2010	18.3	7.9
Hash-MLP 2022	7.5	16.5
DSM 2023	0.0	20.0
Moment bounds 2025	0.0	20.0
DSF uncalibrated	70.8	2.3
DSF + Adam (ours)	89.2	1.0
DSF + CMA-ES (ours)	96.7	0.9

Figure 4: **Recovering light positions from shadows.** Left: success versus initial position error. Right: success and median final position error over 120 starts for 20 lights. Success requires error below 5 cm; uncalibrated DSF uses Adam; calibrated DSF shares one objective across optimizers.

Implementation details and the Hash-MLP control appear in Appendix A.2.1. Percentage-closer filtering (PCF) averages shadow-map depth comparisons, retaining explicit geometric occlusion tests while filtering their results. It achieves the lowest approximation error, but its queries cost $4.8\times$ as much as DSF's and its maps must be updated as lights move. DSF achieves lower point and area mean absolute error (MAE) than the remaining approximate operators in Table 1.

Query cost and reuse. Each query tests visibility between a surface point and a light endpoint. Query time covers a batch of 16,384 such segments, excluding preparation, light updates, and error evaluation. DSF and the bounding volume hierarchy (BVH) reuse scene state as lights move, giving DSF an online advantage over shadow maps that require rebuilding. Total-cost savings depend on how many queries reuse the prepared state. Texture queries take 101.2 μs per batch, versus 119.0 μs for DSM and 636.8 μs for the tested BVH. The differentiable Fixed backend is evaluated in Sections 4.2 and 4.5. Appendix A.2.1 details the protocol, per-scene results, and batch-size checks.

Dependence on source mesh size. Table 3 (right) fixes the surface and DSF field while subdividing the mesh. A $169\times$ increase in triangles raises the tested BVH query time by $2.77\times$, while DSF Texture query times remain nearly constant. This isolates tessellation complexity (Appendix A.6.5).

Real scanned geometry. We additionally evaluated fourteen Stanford-ORB objects, using 512^3 fields and eighteen held-out lights per object. Direct-field and PCF comparisons isolate visibility error on the same scanned geometry; DSF achieved mean MAE 0.0093, versus 0.0497 for the direct field and 0.0111 for PCF, which has higher binary occlusion intersection over union (IoU). Per-object results appear in Appendix A.2.3.

4.2 Differentiable Light Optimization

We recovered twenty point-light positions from 4,096 fixed visibility observations per light. Each method received the same ray-traced targets and 120 paired starts at 10, 20, and 80 cm offsets. Success requires a final position error below 5 cm. The supplementary video shows how DSF optimizes light positions from shadow observations.

Matched operators. Figure 4 compares DSF with other baselines for light-position recovery and evaluates calibration and optimizer choice. Initial offset denotes the Euclidean distance between the initial and GT light positions. All gradient-based methods use Adam, with learning rates selected on separate validation lights; calibrated DSF is also evaluated with gradient-free CMA-ES. With Adam, calibrated DSF achieves higher success and lower median position error than its uncalibrated counterpart, showing that calibration improves gradient-based light recovery. Among gradient-based methods, DSF matches the highest success rate and achieves the lowest median position error. Adam follows local endpoint gradients, whereas CMA-ES explores a population of candidate light positions. Using the same calibrated DSF objective, CMA-ES achieves the best recovery overall at higher optimization cost. Although Density and DSF-Adam achieve the same success rate, DSF localizes lights more precisely, with a lower median position error. Appendix A.3 details baseline configurations, optimization budgets, runtime costs, and uncertainty estimates.

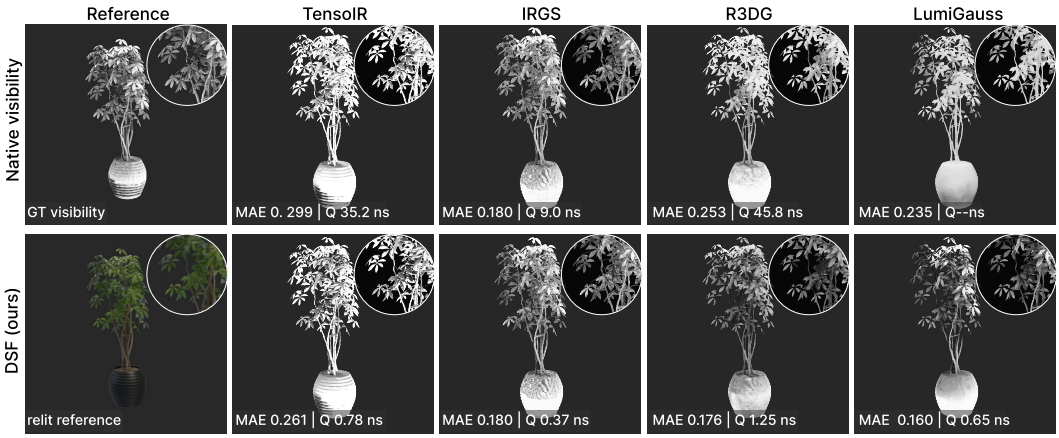


Figure 5: **Visibility across inverse-rendering hosts.** Ficus under bridge illumination: native visibility (top) and DSF (bottom), with visibility and relit references at left. MAE measures foreground visibility error; Q gives average query cost from Table 2.

Table 2: **Visibility reuse across hosts.** Native/DSF compare the host's original visibility with our field. PSNR measures relit-image fidelity to ground truth; MAE measures foreground visibility error against ray tracing, weighted by incident illumination. Query is average time per query segment.

			PSNR ↑		MAE ↓		Query (ns) ↓	
Host	Native visibility	Adaptation	Native	**DSF**	Native	**DSF**	Native	**DSF**
TensoIR (Jin et al., 2023)	Density marching	Retrained	28.58	28.58	0.1780	**0.1720**	35.2	0.78
IRGS (Gu et al., 2025)	Gaussian ray tracing	Frozen	30.64	30.64	0.0935	**0.0930**	9.0	0.37
R3DG (Gao et al., 2024)	Baked visibility	Frozen	21.14	**21.53**	0.1511	**0.1358**	45.8	1.25
LumiGauss (Kaleta et al., 2025)	Learned transfer	Frozen	23.57	**24.00**	0.1971	**0.0916**	–[†]	0.65

[†]Native LumiGauss contracts learned transfer, with no comparable per-segment query cost.

4.3 Reuse across Inverse-Rendering Systems

We tested DSF in four hosts: TensoIR's density marching (Jin et al., 2023), IRGS's Gaussian ray tracing (Gu et al., 2025), R3DG's baked visibility (Gao et al., 2024), and LumiGauss's learned transfer (Kaleta et al., 2025). We measured relighting PSNR and visibility MAE on four TensoIR-Synthetic scenes under five held-out environment maps. We retrained the NeRF-based TensoIR to test whether DSF can preserve rendering quality while accelerating training; the other host checkpoints remained frozen. Additional timing tests showed $3.4\times$ faster BRDF training in TensoIR. Each DSF field was built from the corresponding baseline's reconstructed geometry: an extracted mesh for TensoIR and IRGS or a Gaussian proxy for R3DG and LumiGauss. DSF evaluates visibility on the host's reconstruction and retains its geometric limitations.

Table 2 shows that DSF maintains or improves relighting PSNR while reducing visibility MAE, with the largest reduction in LumiGauss. DSF also achieves lower reported per-segment query costs where comparable native measurements are available. Figure 5 illustrates the visibility improvements. These results demonstrate reuse across heterogeneous inverse-rendering systems while preserving rendering quality. Integration details and timing protocols appear in Appendix A.4.

4.4 Modular Visibility Composition

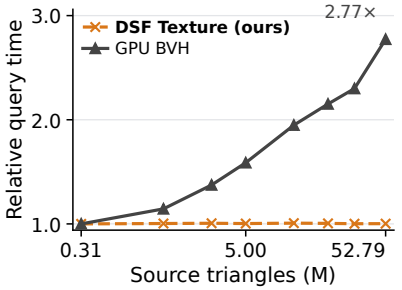
Rigid edits. We inserted and moved a 96^3 direct object field in the calibrated kitchen scene and moved the light, reusing both fields (Figure 6b–d). Against rays through the combined geometry, visibility MAE averaged 0.0630, with queries taking 5.1–5.4 ms per frame. Appendix A.5 extends the evaluation to 108 scene–object–pose–light configurations and analyzes thin-structure errors. On sixteen SynCom scene–object pairs, four independently calibrated object fields were reused across scenes. Replacing only ComGS's object cast-shadow queries preserved similar aggregate RGB accuracy over 48 views, with materials, lighting, and self-occlusion unchanged (Appendix A.5.4).



Figure 6: **Modular scene editing.** (b–d) Rigid edits reuse both fields; (e) deformation rebuilds only the object field. Vis. FPS measures visibility queries only.

Table 3: **Calibration and mesh complexity.** Left: visibility MAE; bullets mark fitted field corrections/query scales (T: Texture; F: Fixed). Right: normalized query time versus mesh triangle count.

	Fitted		Area	Point	
Configuration	Field	Scales	T	T	F
Direct	–	–	0.0520	0.0492	0.0904
Scales only	–	•	0.0176	0.0191	0.0235
Field only	•	–	0.0225	0.0211	0.0267
DSF (ours)	•	•	0.0037	0.0054	0.0059



Nonrigid edits. For deformation, we rebuilt only the 48^3 direct object field, reusing the query-scale settings and calibrated background without retraining (Figure 6e). After each update, visibility queries ran at 185 FPS. Appendix A.5.3 reports update costs and accuracy.

4.5 Ablation Studies

What should be calibrated? Table 3 (left) compares four configurations: no calibration, field corrections $\boldsymbol{\rho}$ only, query scales (r, w, τ) only, and joint calibration. Field corrections adapt local values, whereas query scales adjust how those values translate into visibility. All configurations share the 1024^3 initialization; fitted variants use 10k updates on identical teacher–student segments. Calibrating either component reduces error, while fitting both gives the lowest aggregate point and area MAE under matched query settings (Appendix Table 25). This indicates that local field adaptation and global query adjustment are complementary. Joint calibration also gives the lowest median light-position error among the four controls (1.01 cm; Appendix Table 27). The main table additionally applies the validation-selected Texture scales to full DSF; controls retain base scales.

Field and evaluator. Calibration also improves cone-soft visibility, benefiting other evaluators (Appendix Table 28). The fixed-sample differentiable backend (Fixed) supports light-position optimization, while the texture-based adaptive backend (Texture) reduces forward query cost (Appendix Table 31). Higher grid resolutions reduce calibrated visibility error at greater memory cost, while increasing the fixed-sample count improves accuracy at greater query cost (Appendix A.6).

5 Limitations and Conclusion

Limitations. Our current implementation prioritizes visibility calibration and reuse. Dense grids allocate the same resolution to occupied and empty regions, increasing memory costs for large scenes. An adaptive sparse grid could retain fine resolution near geometry while representing empty space coarsely, reducing memory without uniformly coarsening the field. DSF also incurs a one-time calibration cost per scene, amortized over subsequent queries and light changes. Future work could reduce this setup cost through more informative training-pair sampling and earlier stopping.

Conclusion. DSF represents finite-segment visibility through a shared, light-independent field and an endpoint-conditioned evaluator. Ray-traced supervision calibrates the field through its visibility queries. This representation supports forward prediction, light-position inference, and reuse across lights, rendering hosts, and object edits through rigid composition and local field updates.

AI USE STATEMENT

In this work, we used generative AI tools to assist with refining research hypotheses and paper positioning, providing feedback on methodology and experimental design, supporting implementation and debugging, and discussing the interpretation and presentation of experimental results. We additionally used generative AI tools for literature search and summarization, scientific figure editing and layout, and drafting or editing portions of the manuscript for clarity, concision, and readability.

Experimental data and quantitative results were obtained from the datasets, rendering procedures, and evaluations described in this paper. Generative AI was not used to fabricate measured results or independently establish mathematical or empirical claims. All mathematical formulations, algorithmic designs, implementation changes, experimental protocols, quantitative results, citations, and conclusions were reviewed and verified by the authors. AI-assisted code was manually inspected and tested against controlled cases or reference implementations where applicable, and literature suggestions and factual statements were checked against the original sources. The authors take full responsibility for the final content of this work, including text, code, figures, claims, and other artifacts produced with the assistance of generative AI.

ETHICS STATEMENT

This work studies a general-purpose computational representation for differentiable visibility in rendering and inverse graphics. It does not involve human subjects, personally identifiable information, or sensitive personal data. Our experiments use synthetic or previously released scene assets and benchmarks as described in the paper and supplementary material. The proposed method is intended to improve the efficiency and differentiability of visibility queries for applications such as rendering, inverse rendering, simulation, and scene editing. We are not aware of ethical concerns specific to the proposed technique beyond those associated with these general application areas.

REPRODUCIBILITY STATEMENT

We provide the mathematical formulation of Differentiable Shadow Fields (DSF), including the construction of the underlying distance-field representation, endpoint-conditioned visibility query, calibration procedure, refinement mechanism, and endpoint derivatives, in the main paper. The experimental section and appendix describe the scene preparation and query-generation protocols, ray-traced reference construction, evaluation metrics, baseline configurations, and timing methodology. In particular, Appendix A.2.1 provides additional details of the visibility benchmark and matched comparisons. We report the principal hyperparameters required to reproduce our results, including field resolutions, segment sampling settings, calibration and refinement configurations, as well as the hardware and evaluation settings used for accuracy and performance measurements. These details are intended to enable independent reproduction of the reported experiments.

REFERENCES

Thomas Annen, Tom Mertens, Philippe Bekaert, Hans-Peter Seidel, and Jan Kautz. Convolution shadow maps. In *EGSR*, 2007.

Sai Praveen Bangaru, Michaël Gharbi, Tzu-Mao Li, Fujun Luan, Kalyan Sunkavalli, Miloš Hašan, Sai Bi, Zexiang Xu, Gilbert Bernstein, and Frédo Durand. Differentiable rendering of neural SDFs through reparameterization. In *SIGGRAPH Asia Conference Papers*, 2022.

David Bauer, Qi Wu, Hamid Gadirov, and Kwan-Liu Ma. GSCache: Real-time radiance caching for volume path tracing using 3D gaussian splatting. *IEEE Transactions on Visualization and Computer Graphics*, 32(1):901–911, 2026. doi: 10.1109/TVCG.2025.3634634.

Zoubin Bi, Yixin Zeng, Chong Zeng, Fan Pei, Xiang Feng, Kun Zhou, and Hongzhi Wu. GS^3: Efficient relighting with triple gaussian splatting. In *SIGGRAPH Asia Conference Papers*, 2024.

Jakub Bokšanský and Daniel Meister. Neural visibility cache for real-time light sampling. *Journal of Computer Graphics Techniques (JCGT)*, 14(2), 2025. arXiv:2506.05930.

Luis Bolanos, Shih-Yang Su, and Helge Rhodin. Gaussian shadow casting for neural characters. In *CVPR*, 2024. URL https://github.com/LuisBolanos17/GaussianShadowCasting.

Hongze Chen, Zehong Lin, and Jun Zhang. GI-GS: Global illumination decomposition on gaussian splatting for inverse rendering. In *ICLR*, 2025.

William Donnelly and Andrew Lauritzen. Variance shadow maps. In *I3D*, 2006.

Kang Du, Zhihao Liang, Yulin Shen, and Zeyu Wang. GS-ID: Illumination decomposition on gaussian splatting via adaptive light aggregation and diffusion-guided material priors. In *ICCV*, 2025.

Sarah F. Frisken, Ronald N. Perry, Alyn P. Rockwood, and Thouis R. Jones. Adaptively sampled distance fields: A general representation of shape for computer graphics. In *SIGGRAPH*, pp. 249–254, 2000.

Shin Fujieda, Chih-Chen Kao, and Takahiro Harada. Neural intersection function. In *High-Performance Graphics*, 2023. arXiv:2306.07191.

Shin Fujieda, Chih-Chen Kao, and Takahiro Harada. LSNIF: Locally-subdivided neural intersection function. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 2025. doi: 10.1145/3728295.

Jian Gao, Chun Gu, Youtian Lin, Zhihao Li, Hao Zhu, Xun Cao, Li Zhang, and Yao Yao. Relightable 3D gaussians: Realistic point cloud relighting with BRDF decomposition and ray tracing. In *ECCV*, 2024.

Jian Gao, Mengqi Yuan, Yifei Zeng, Chang Zeng, Zhihao Li, Zhenyu Chen, Weichao Qiu, Xiao-Xiao Long, Hao Zhu, Xun Cao, and Yao Yao. ComGS: Efficient 3D object-scene composition via surface octahedral probes. In *ICLR*, 2026. arXiv:2510.07729.

Chun Gu, Xiaofei Wei, Zixuan Zeng, Yuxuan Yao, and Li Zhang. IRGS: Inter-reflective gaussian splatting with 2D gaussian ray tracing. In *CVPR*, 2025.

Yijia Guo, Yuanxi Bai, Liwen Hu, Ziyi Guo, Mianzhi Liu, Yu Cai, Tiejun Huang, and Lei Ma. PRTGS: Precomputed radiance transfer of gaussian splats for real-time high-quality relighting. In *ACM Multimedia*, 2024.

Kyu Beom Han, Jaeyoon Kim, Woo Jae Kim, Jinhwan Seo, and Sung eui Yoon. Radiometrically consistent gaussian surfels for inverse rendering. In *ICLR*, 2026. URL https://arxiv.org/abs/2603.01491.

John C. Hart. Sphere tracing: A geometric method for the antialiased ray tracing of implicit surfaces. *The Visual Computer*, 12(10), 1996.

Haian Jin, Isabella Liu, Peijia Xu, Xiaoshuai Zhang, Songfang Han, Sai Bi, Xiaowei Zhou, Zexiang Xu, and Hao Su. TensoIR: Tensorial inverse rendering. In *CVPR*, 2023.

Joanna Kaleta, Kacper Kania, Tomasz Trzciński, and Marek Kowalski. LumiGauss: Relightable gaussian splatting in the wild. In *WACV*, 2025.

Chih-Chen Kao, Grzegorz Makowski, Shin Fujieda, and Takahiro Harada. Voxel deformation-aware neural intersection function. In *Eurographics 2026 – Short Papers*, 2026. doi: 10.2312/egs.20261004.

Zhengfei Kuang, Yunzhi Zhang, Hong-Xing Yu, Samir Agarwala, Shangzhe Wu, and Jiajun Wu. Stanford-ORB: A real-world 3D object inverse rendering benchmark. In *NeurIPS*, 2023. URL https://stanfordorb.github.io/.

Tzu-Mao Li, Miika Aittala, Frédo Durand, and Jaakko Lehtinen. Differentiable monte carlo ray tracing through edge sampling. *ACM Transactions on Graphics*, 37(6), 2018.

Zhihao Liang, Qi Zhang, Ying Feng, Ying Shan, and Kui Jia. GS-IR: 3D gaussian splatting for inverse rendering. In *CVPR*, 2024.

Jingwang Ling, Zhibo Wang, and Feng Xu. ShadowNeuS: Neural SDF reconstruction by shadow ray supervision. In *CVPR*, 2023.

Shaohui Liu, Yinda Zhang, Songyou Peng, Boxin Shi, Marc Pollefeys, and Zhaopeng Cui. DIST: Rendering deep implicit signed distance function with differentiable sphere tracing. In *CVPR*, 2020.

Guillaume Loubet, Nicolas Holzschuch, and Wenzel Jakob. Reparameterizing discontinuous integrands for differentiable rendering. *ACM Transactions on Graphics*, 38(6), 2019.

Linjie Lyu, Marc Habermann, Lingjie Liu, Mallikarjun B R, Ayush Tewari, and Christian Theobalt. Efficient and differentiable shadow computation for inverse problems. In *ICCV*, 2021.

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

Aymen Mir, Riza Alp Guler, Jian Wang, Gerard Pons-Moll, and Bing Zhou. Animated 3DGS avatars in diverse scenes with consistent lighting and shadows. *arXiv preprint arXiv:2601.01660*, 2026a.

Aymen Mir, Riza Alp Guler, Jian Wang, Peter Wonka, Bing Zhou, and Gerard Pons-Moll. RAGA: Real time ray traced gaussian shadow casting for 3DGS avatar-scene interaction. In *European Conference on Computer Vision*, 2026b. URL https://arxiv.org/abs/2606.29329.

Thomas Müller, Fabrice Rousselle, Jan Novák, and Alexander Keller. Real-time neural radiance caching for path tracing. *ACM Transactions on Graphics*, 40(4), 2021.

Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics*, 41(4):102:1–102:15, 2022. doi: 10.1145/3528223.3530127.

Inigo Quilez. Soft shadows in raymarched SDFs. https://iquilezles.org/articles/rmshadows/, 2010.

William T. Reeves, David H. Salesin, and Robert L. Cook. Rendering antialiased shadows with depth maps. In *SIGGRAPH*, 1987.

Sergio Sancho, Alexander Rath, Marco Manzi, Pascal Chang, Amit H. Bermano, Derek Nowrouzezahrai, Markus Gross, and Marios Papas. Neural render proxies for interactive and differentiable lighting. *Computer Graphics Forum*, 45(4), 2026. doi: 10.1111/cgf.70533.

Peter-Pike Sloan, Jan Kautz, and John Snyder. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. In *SIGGRAPH*, 2002.

Pratul P. Srinivasan, Boyang Deng, Xiuming Zhang, Matthew Tancik, Ben Mildenhall, and Jonathan T. Barron. NeRV: Neural reflectance and visibility fields for relighting and view synthesis. In *CVPR*, 2021.

Yu Wei Tan, Nicholas Chua, Clarence Koh, and Anand Bhojan. RTSDF: Real-time signed distance fields for soft shadow approximation in games. In *GRAPP*, 2022.

Delio Vicini, Sébastien Speierer, and Wenzel Jakob. Differentiable signed distance function rendering. *ACM Transactions on Graphics*, 41(4), 2022.

Jun-Hao Wang, Yi-Yang Tian, Baoquan Chen, and Peng-Shuai Wang. Neural visibility of point sets. In *SIGGRAPH Asia Conference Papers*, 2025a. doi: 10.1145/3757377.3763869. URL https://arxiv.org/abs/2509.24150.

Xiangyu Wang, Thomas Köhler, Jun Lin Qiu, Shohei Mori, Markus Steinberger, and Dieter Schmalstieg. NeuralPVS: Learned estimation of potentially visible sets. In *SIGGRAPH Asia Conference Papers*, 2025b. doi: 10.1145/3757377.3763822. URL https://arxiv.org/abs/2509.24677.

Philippe Weier, Alexander Rath, Élie Michel, Iliyan Georgiev, Philipp Slusallek, and Tamy Boubekeur. N-BVH: Neural ray queries with bounding volume hierarchies. In *ACM SIGGRAPH 2024 Conference Papers*, 2024.

Lance Williams. Casting curved shadows on curved surfaces. In *SIGGRAPH*, 1978.

Markus Worchel and Marc Alexa. Differentiable shadow mapping for efficient inverse graphics. In *CVPR*, 2023.

Markus Worchel and Marc Alexa. Moment bounds are differentiable: Efficiently approximating measures in inverse rendering. *ACM Transactions on Graphics*, 44(4), 2025. doi: 10.1145/3730899.

Daniel Wright. Dynamic occlusion with signed distance fields. Advances in Real-Time Rendering, ACM SIGGRAPH Courses, 2015.

Lifan Wu, Nathan Morrical, Sai Praveen Bangaru, Rohan Sawhney, Shuang Zhao, Chris Wyman, Ravi Ramamoorthi, and Aaron Lefohn. Unbiased differential visibility using fixed-step walk-on-spherical-caps and closest silhouettes. *ACM Transactions on Graphics*, 2025. doi: 10.1145/3731174. URL https://research.nvidia.com/labs/prl/publication/wu2025diffvisibility/.

Liwen Wu, Rui Zhu, Mustafa B. Yaldiz, Yinhao Zhu, Hong Cai, Janarbek Matai, Fatih Porikli, Tzu-Mao Li, Manmohan Chandraker, and Ravi Ramamoorthi. Factorized inverse path tracing for efficient and accurate material-lighting estimation. In *ICCV*, 2023.

Zhi Ying, Nicholas Edwards, and Mikhail Kutuzov. Efficient visibility approximation for game AI using neural omnidirectional distance fields. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 7(1), 2024. doi: 10.1145/3651289.

Chong Zeng, Guojun Chen, Yue Dong, Pieter Peers, Hongzhi Wu, and Xin Tong. Relighting neural radiance fields with shadow and highlight hints. In *ACM SIGGRAPH Conference Proceedings*, 2023.

Xiuming Zhang, Pratul P. Srinivasan, Boyang Deng, Paul Debevec, William T. Freeman, and Jonathan T. Barron. NeRFactor: Neural factorization of shape and reflectance under an unknown illumination. *ACM Transactions on Graphics*, 40(6), 2021.

Iris Zheng, Guojun Tang, Alexander Doronin, Paul Teal, and Fang-Lue Zhang. SSD-GS: Scattering and shadow decomposition for relightable 3D gaussian splatting. In *ICLR*, 2026.

Kun Zhou, Yaohua Hu, Stephen Lin, Baining Guo, and Heung-Yeung Shum. Precomputed shadow fields for dynamic scenes. *ACM Transactions on Graphics*, 24(3), 2005.

Junke Zhu, Hao Zhang, Yutian Zhu, Ang Li, Chenxiao Hu, Meng Gai, Fei Zhu, Zhangjin Huang, and Sheng Li. PTIR-GS: Path-traced inverse rendering with global illumination in 3D gaussian fields. *arXiv preprint arXiv:2606.09606*, 2026. URL https://arxiv.org/abs/2606.09606.

A APPENDIX

CONTENTS

The appendix follows the method and experimental sections of the main paper.

A.1 Method Details **p. 14**
Sampling and derivatives; calibration; execution backends.
A.2 Visibility Prediction **p. 16**
Common-segment benchmark; independent endpoints; real scanned objects.
A.3 Differentiable Light Optimization **p. 21**
Queries, optimizers, selection criteria, and recovery metrics.
A.4 Reuse Across Inverse-Rendering Systems **p. 22**
Shared protocol; host adapters; timing boundaries; relighting examples.
A.5 Modular Composition **p. 25**
Rigid editing; error attribution; nonrigid updates; SynCom validation.
A.6 Ablations and Additional Analyses **p. 30**
Calibration; evaluators; resolution; proxy geometry; query costs; endpoint stress.

A.1 METHOD DETAILS

A.1.1 SAMPLING AND ENDPOINT DERIVATIVES

Sample placement. For the visibility query in Eq. (3), let $L = \|\boldsymbol{p}-\boldsymbol{x}\|$. If $L \le \delta+\delta_{\mathrm{l}}$, the segment is treated as visible. Otherwise, we place K samples at the midpoints of equal intervals between the start and light clearances:

$$\boldsymbol{q}_k = \boldsymbol{x} + \big(\delta + u_k(L-\delta-\delta_{\mathrm{l}})\big)\,\hat{\boldsymbol{v}}, \qquad u_k = \tfrac{k-1/2}{K}, \qquad \hat{\boldsymbol{v}} = \tfrac{\boldsymbol{p}-\boldsymbol{x}}{L}. \tag{A.1}$$

Here, $k = 1,\dots,K$; the upright subscript l denotes the light endpoint, distinct from the voxel-cell diagonal ℓ. Both clearances are measured along the segment. The start clearance defaults to $\max(5\,\mathrm{mm},\ 1.25\ell)$ to skip the source surface and its interpolation neighborhood, unless the host prescribes a shadow-ray $t_{\min}$, in which case we match it; the light clearance defaults to $1\,\mathrm{mm}$. Experiment-specific endpoints and clearances are stated in Appendix A.2.1 and the application protocols.

Endpoint Jacobian. We differentiate Eq. (A.1) with $\boldsymbol{x}$, δ, and δ_{l} held fixed. Writing $\boldsymbol{v} = \boldsymbol{p} - \boldsymbol{x}$ and $\beta_k = \delta - u_k(\delta+\delta_{\mathrm{l}})$, the sample position is $\boldsymbol{q}_k = \boldsymbol{x} + \alpha_k\boldsymbol{v}$ with $\alpha_k = u_k + \beta_k/\|\boldsymbol{v}\|$, so

$$\frac{\partial \boldsymbol{q}_k}{\partial \boldsymbol{p}} = \alpha_k\,\mathbf{I} - \beta_k\,\frac{\boldsymbol{v}\boldsymbol{v}^{\top}}{\|\boldsymbol{v}\|^3}. \tag{A.2}$$

Here, $\mathbf{I}$ is the 3×3 identity matrix. Substituting Eq. (A.2) into Eq. (5) accounts for the change in segment direction and length as the endpoint moves, giving the expression evaluated by the backward kernel. Together with the field derivative, this Jacobian gives the derivative of the fixed-sample surrogate almost everywhere. It is not a derivative guarantee for binary geometric visibility.

A.1.2 CALIBRATION AND BOUNDARY HANDLING

Calibration parameterization. Figure 3 illustrates field sampling and aggregation; Figure 7 details the visibility mapping and supervision. The query scales in Eq. (3) are bounded multipliers of cell-diagonal-scaled defaults,

$$(r, w, \tau) = (s_r m_r,\ s_w m_w,\ s_\tau m_\tau)\,\ell, \qquad m_\bullet = \tfrac{1}{2} + \sigma(\eta_\bullet) \in (\tfrac{1}{2}, \tfrac{3}{2}), \tag{A.3}$$

where $\bullet \in \{r, w, \tau\}$ and $\eta_r, \eta_w, \eta_\tau$ are shared trainable scalars, distinct from the local field parameters ρ_i, with defaults $(s_r, s_w, s_\tau) = (0.6,\ 0.2,\ 0.2)$, so the query scales remain positive and bounded. The vertex correction in Eq. (2) is bounded by $\ell/2$; the raw parameters ρ_i themselves are unconstrained.

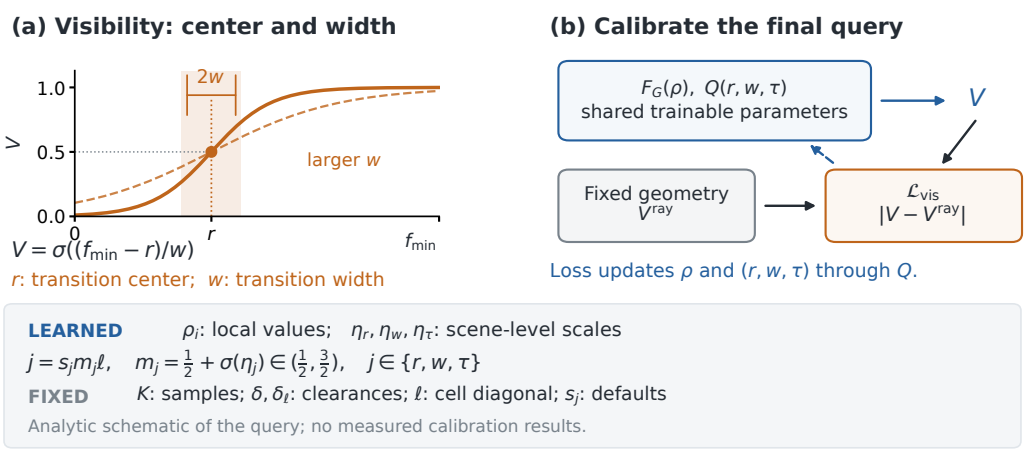


Figure 7: **Visibility mapping and query-level calibration.** (a) Radius and width control the sigmoid. (b) Ray labels supervise field corrections and query scales. Analytic schematic.

Training recipe. Calibration is supervised by binary visibility from the exact ray caster on sampled point–light pairs. Surface points are sampled uniformly on the proxy mesh, independently of any evaluation view. Before endpoint clearances, raw light samples follow the deployment distribution: half lie on the scene's emitter surfaces and half are free-space points with clearance from geometry. We minimize Eq. (4): an L_1 loss between the field's soft visibility and the teacher's binary visibility, with a mean absolute penalty on the raw vertex parameters ρ_i; gradients reach $\boldsymbol{\rho}$ and the three multipliers through the PyTorch reference implementation of Eq. (3). The current fields use Adam at a learning rate of 10^{-2} for 10k updates of 65,536 pairs, with 1024 samples per segment and a two-million-point mesh sampling pool. Training uses seed 7. For the common-segment comparisons and calibration ablations, the sampled receiver is moved 5 cm toward the sampled light, and the light endpoint is moved 8 cm toward the receiver. These explicit endpoints are passed to both the ray teacher and the field evaluator with zero additional clearance and no teacher-only intersection tolerance. The 8 cm endpoint shift preserves the sampled segments used by the original deployment recipe; held-out queries use the actual endpoints specified in Appendix A.2.1. Application-specific recipes are stated separately below. Table 1 reports the preparation cost of these fields. The common-segment recipe uses base radius and width scales of 0.25 and 0.10 at all evaluated resolutions.

Out-of-bounds convention. Samples outside the modeled box return the box diagonal, a large free-space distance, rather than zero. This avoids interpreting out-of-bounds samples as surface crossings. For a point $\boldsymbol{x}$ inside the box and a unit direction $\boldsymbol{\omega}$, we query the endpoint $\boldsymbol{p}_{\text{exit}} = \boldsymbol{x} + t_{\text{exit}}\boldsymbol{\omega}$, where

$$t_{\text{exit}} = \min_{a \in \{1,2,3\}} t_a, \qquad t_a = \begin{cases} (b_a^{\max} - x_a)/\omega_a, & \omega_a > 0, \\ (b_a^{\min} - x_a)/\omega_a, & \omega_a < 0, \\ +\infty, & \omega_a = 0. \end{cases} \tag{A.4}$$

Here, $\boldsymbol{b}^{\min}$ and $\boldsymbol{b}^{\max}$ are the padded box bounds. This reduction assumes all modeled occluders lie within the box; finite sampling and endpoint clearances still apply.

A.1.3 Execution Backends and Implementation

Execution backends. All three backends query the stored field without source-mesh intersections. The Fixed backend (`exact` in code) evaluates Eq. (3) at the samples of Eq. (A.1). Its custom backward recomputes these positions and returns the light-endpoint derivative in Eq. (5) without retaining a per-sample autograd graph. Softmax weights use maximum-logit subtraction for numerical stability.

Sphere and Texture are forward-only approximations of the fixed-sample query: they use adaptive field-guided stepping and a hard minimum instead of soft aggregation. Sphere reads F_G by trilinear

interpolation and advances by the stored value clamped to $[\ell/2, 8\ell]$, tracks a hard minimum m, and stops at the light, after K steps, or when $m < r - 3w$. Texture uploads the calibrated vertex values to a hardware-filtered 3D texture, using the same adaptive stepping rule. The texture is uploaded after field edits; directional batches evaluate Eq. (A.4) inside the kernel. Fixed supplies derivatives; Texture is the fast forward path, with Sphere as a non-texture alternative. Fixed is intended as a correctness-oriented differentiable implementation rather than a throughput-optimized GPU kernel. It directly evaluates the fixed-sample formulation of Eq. (3) and its endpoint derivative in Eq. (5). In contrast, Texture is engineered for forward throughput using hardware-filtered 3D texture reads and adaptive stepping. The large runtime gap between Fixed and Texture therefore reflects both evaluator differences and implementation maturity, and should not be interpreted as an intrinsic cost of differentiable DSF queries. Table 31 compares their accuracy and cost.

Implementation settings. Table 4 separates engine defaults from evaluation settings. Scales use the cell diagonal ℓ; the training recipe above specifies the common-segment overrides. The teacher is an Embree binary ray caster on the geometry proxy. Preparation costs are reported in Table 1.

Table 4: **Implementation settings.** Query scales are engine defaults; calibration and evaluation overrides are specified in the corresponding protocols.

Quantity	Setting	Note
Grid resolution	1024^3	384^3/768^3 comparisons in Appendix A.6
Box padding	10% of the mesh extent	samples outside read as free space
Collision radius r	$0.6\,\ell$	multiplier fitted in $[\frac{1}{2}, \frac{3}{2}]$
Transition width w	$0.2\,\ell$	multiplier fitted in $[\frac{1}{2}, \frac{3}{2}]$
Soft-min temperature τ	$0.2\,\ell$	multiplier fitted in $[\frac{1}{2}, \frac{3}{2}]$
Residual bound	$\ell/2$	$\tanh$-bounded per grid vertex
Start clearance δ	$\max(5\,\mathrm{mm}, 1.25\,\ell)$	experiment-specific overrides
Light clearance δ_l	1 mm	experiment-specific overrides
Samples per segment K	1024	Fixed; Texture budget stated per experiment
Sphere-tracing step	clipped to $[\ell/2, 8\ell]$	at most K steps
Calibration	Adam, lr 10^{-2}, 10k updates	65,536 point–light pairs per iteration
Calibration loss	L_1 to binary teacher visibility	mean absolute penalty on raw vertex parameters

A.2 Visibility Prediction

A.2.1 Common-Segment Benchmark

Inputs and reference. We reused the four indoor meshes and frozen 1024^3 fields calibrated for 10k updates. For each scene, two of the twenty previously generated novel light positions are reserved for baseline validation; the remaining eighteen are test positions. The same 16,384 receivers are used at every test light. Receivers retain their stored 3 mm normal offset and are moved 5 cm along the ray to define the explicit query start. All operators receive these starts and the actual light endpoints, with no additional hidden start or light-end trimming. The reference uses two-sided CPU triangle intersection on these exact segments; GPU BVH predictions are independently checked against it. This test evaluates the declared offset segments, not visibility in the omitted near-surface interval.

For four test light positions per scene, we additionally constructed a square emitter in the xz plane with side length 2% of the scene bounding-box diagonal. We evaluated 2,048 receivers using the same 4×4 emitter samples for every operator and the ray reference. Area visibility is the uniform mean of these sixteen segment values; it is not BRDF-weighted radiance. Thus these errors isolate the operator and contain no difference in emitter-sampling budgets. Point-query occlusion IoU thresholds predictions at 0.5, with zero visibility denoting occlusion. Metrics average lights equally within each scene and then average scenes.

Operators and validation. The GPU BVH uses the existing two-sided CUDA closest-hit implementation; it is not a hardware any-hit kernel. All shadow maps share the official `diffshadow` rasterization and cube-texture conventions. PCF uses nine depth comparisons; VSM uses filtered first and second moments; DSM additionally uses the author's differentiable raster antialiasing. VSM and DSM use a 7×7 box filter and the library's 10^{-4} variance floor. Moment bounds use four normalized depth moments, the same positive box filter, and the official `diffmoments` CUDA solver.

Raster antialiasing is omitted for these moments because it can produce invalid moment sequences. All maps use 2048^2 cube faces, near distance 10^{-4}, and far distance three scene diagonals. Higher moments and the earlier midpoint-bound configuration are not used as the representative baseline here.

Each baseline is selected using 1,024 receivers at each of the two validation lights, independently of test labels. Depth biases are 0, h, or $2h$, where h is the scene diagonal/1024. Moment biases are 10^{-5}, 10^{-4}, or 10^{-3}, with lower, midpoint, or upper occlusion bounds. Nonfinite validation configurations are rejected. Hard-distance thresholds are 0.1, 0.2, 0.3, or 0.5 cell diagonals; cone coefficients are 16, 32, 64, or 128; density amplitudes are 10, 50, or 100 with widths 0.25, 0.5, or 1 cell edge. These direct-field controls use the uncalibrated 1024^3 UDF and 1,024 midpoint samples. Hard distance thresholds the minimum sampled distance, inspired by distance-based intersection tests (Hart, 1996); it is not adaptive sphere tracing. Density uses exponential transmittance (Mildenhall et al., 2020) with a Gaussian UDF-to-density profile, $\sigma_{\mathrm{dens}}(d) = a_{\mathrm{dens}} \exp[-d^2/(2w_{\mathrm{dens}}^2)]$, where a_{dens} and w_{dens} are its amplitude and width; it is a controlled density surrogate, not a trained NeRF. DSF Fixed uses 1,024 samples; Texture has a 256-step budget. After field calibration, Texture base radius and width scales are selected jointly on these same eight validation lights, using one shared pair across scenes. Candidates are radius scales $0.02, 0.04, 0.08, 0.125, 0.1875, 0.25, 0.3125$ and width scales $0.01, 0.025, 0.05, 0.1, 0.2$. Field values and learned multipliers remain frozen. Fixed retains the query scales used during training. Table 5 reports the selected settings.

Table 5: **Validation-selected query parameters.** Two reserved lights per scene; configurations remain fixed for all test queries. DSF shares one radius/width pair across scenes, in cell-diagonal units; depth biases are in scene units.

Scene / operator	Selected parameters
kitchen / PCF (Reeves et al., 1987)	depth bias=0
kitchen / VSM (Donnelly & Lauritzen, 2006)	depth bias=0
kitchen / DSM (Worchel & Alexa, 2023)	depth bias=0
kitchen / Moment bounds (Worchel & Alexa, 2025)	depth bias=0, bias=1e-05, blend=0
kitchen / Hard distance (Hart, 1996)	threshold=0.3
kitchen / Cone-soft (Quilez, 2010)	cone=32
kitchen / Density (Mildenhall et al., 2020)	amplitude=100, width=0.25
bathroom / PCF (Reeves et al., 1987)	depth bias=0
bathroom / VSM (Donnelly & Lauritzen, 2006)	depth bias=0
bathroom / DSM (Worchel & Alexa, 2023)	depth bias=0
bathroom / Moment bounds (Worchel & Alexa, 2025)	depth bias=0, bias=1e-05, blend=0
bathroom / Hard distance (Hart, 1996)	threshold=0.3
bathroom / Cone-soft (Quilez, 2010)	cone=16
bathroom / Density (Mildenhall et al., 2020)	amplitude=100, width=0.5
bedroom / PCF (Reeves et al., 1987)	depth bias=0
bedroom / VSM (Donnelly & Lauritzen, 2006)	depth bias=0
bedroom / DSM (Worchel & Alexa, 2023)	depth bias=0
bedroom / Moment bounds (Worchel & Alexa, 2025)	depth bias=0, bias=1e-05, blend=0
bedroom / Hard distance (Hart, 1996)	threshold=0.3
bedroom / Cone-soft (Quilez, 2010)	cone=32
bedroom / Density (Mildenhall et al., 2020)	amplitude=100, width=0.5
livingroom / PCF (Reeves et al., 1987)	depth bias=0
livingroom / VSM (Donnelly & Lauritzen, 2006)	depth bias=0
livingroom / DSM (Worchel & Alexa, 2023)	depth bias=0
livingroom / Moment bounds (Worchel & Alexa, 2025)	depth bias=0, bias=1e-05, blend=0
livingroom / Hard distance (Hart, 1996)	threshold=0.3
livingroom / Cone-soft (Quilez, 2010)	cone=64
livingroom / Density (Mildenhall et al., 2020)	amplitude=100, width=0.25
All scenes / **DSF Texture (ours)**	radius base=0.125, width base=0.01

Learned endpoint control. Hash-MLP directly predicts segment visibility from its two endpoints. A shared multiresolution hash encoding (Müller et al., 2022) is read at the start, midpoint, and end; these features and the normalized endpoints feed an MLP with four width-128 hidden layers. The encoding has sixteen levels with two features each, resolutions 16–2048, and up to 2^{19} hash entries per level. We used the official tiny-cuda-nn implementation. This is a direct learned control, not a reproduction of N-BVH or NVC. Each scene receives the same ray-supervised training sampler as DSF: 10k steps, batch size 65,536, and seed 7. Adam minimizes binary cross-entropy with a cosine learning-rate schedule from 10^{-3} to 10^{-4}. Every 500 steps, the two reserved validation lights select the checkpoint; test lights are never used for selection. Evaluation uses the unchanged point and area queries and timing protocol.

Table 6: **Learned endpoint control.** Hash-MLP (control) on the unchanged common-segment test set; query time is for 16,384 point segments. Step is selected using the two reserved validation lights.

Scene	Point MAE	Occ. IoU	Area MAE	Query (μs)	Prep. (s)	Step
kitchen	0.0529	0.787	0.0286	197.7	221.6	9500
bathroom	0.0128	0.808	0.0106	201.3	239.5	10000
bedroom	0.0164	0.833	0.0203	195.5	239.3	9500
livingroom	0.0274	0.812	0.0209	195.3	273.5	8000

Timing and preparation. For query timing, inputs and outputs stay on the CS RTX PRO 6000 GPU. Every query measurement uses two warmups and seven repetitions; we report the median synchronized wall time and independently record CUDA events. Query timing includes endpoint arithmetic, output allocation, and the visibility evaluation, but excludes transfers, reference computation, CPU metrics, compilation, and state construction. All map filtering is charged to the light-update stage. The eighteen-light workload estimate (Table 1) sums separately measured updates and median cached-query times at eighteen positions with fixed receivers; it excludes one-time scene preparation. Each light update is timed once after loading that light, whereas query timing uses warmed repetitions. A separate repeated-update-plus-query diagnostic rebuilds the same light after query warmups; its different cache state can give lower times and it is not used for this estimate. This is an operator workload, not full-frame rendering FPS. Preparation includes measured setup for maps and BVH. DSF preparation includes field construction and calibration, timed separately on another machine. Offline deployment-parameter sweeps are excluded from preparation costs. Hash-MLP preparation includes the auxiliary UDF used by the matched training sampler, ray-teacher construction, training, and checkpoint validation and saving. The auxiliary UDF is discarded for inference. Table 30 reports grid-value storage and resolution costs.

Table 9 compares a common timing boundary at three batch sizes. Nanoseconds per segment are batch-average throughput, not independent single-ray latency.

Tables 7 and 8 report per-scene results.

Table 7: **Common-segment evaluation by scene.** Query cost is synchronized wall time for 16,384 point queries; the 18-light estimate sums separately measured updates and queries. Fixed rows report accuracy only.

Scene / operator	Point MAE	Area MAE	Query (μs)	18 lights, est. (ms)
kitchen / Ray tracing (GPU BVH)	0.0000	0.0000	733.6	13.20
kitchen / PCF (Reeves et al., 1987)	0.0081	0.0032	492.7	232.21
kitchen / VSM (Donnelly & Lauritzen, 2006)	0.0179	0.0096	121.9	255.02
kitchen / DSM (Worchel & Alexa, 2023)	0.0182	0.0099	119.4	318.32
kitchen / Moment bounds (Worchel & Alexa, 2025)	0.0204	0.0114	140.8	357.55
kitchen / Hard distance (Hart, 1996)	0.0739	0.0832	3200.5	57.61
kitchen / Cone-soft (Quilez, 2010)	0.0490	0.0296	3337.2	60.07
kitchen / Density (Mildenhall et al., 2020)	0.1012	0.1426	3418.8	61.54
kitchen / **DSF Texture (ours)**	0.0139	0.0085	89.7	1.62
kitchen / **DSF Fixed (ours)**	0.0144	0.0066	–	–
bathroom / Ray tracing (GPU BVH)	0.0000	0.0000	514.4	9.26
bathroom / PCF (Reeves et al., 1987)	0.0029	0.0021	484.3	172.51
bathroom / VSM (Donnelly & Lauritzen, 2006)	0.0080	0.0063	120.6	196.19
bathroom / DSM (Worchel & Alexa, 2023)	0.0082	0.0064	119.2	228.43
bathroom / Moment bounds (Worchel & Alexa, 2025)	0.0082	0.0065	140.7	297.73
bathroom / Hard distance (Hart, 1996)	0.0061	0.0046	3299.0	59.38
bathroom / Cone-soft (Quilez, 2010)	0.0163	0.0131	3487.4	62.77
bathroom / Density (Mildenhall et al., 2020)	0.0244	0.0093	3564.1	64.15
bathroom / **DSF Texture (ours)**	0.0024	0.0016	90.1	1.62
bathroom / **DSF Fixed (ours)**	0.0028	0.0015	–	–

A.2.2 Independent Endpoints and Finite Segments

We evaluated the four current 1024^3 controls on 72 held-out lights, with 10^5 independently sampled receivers per light. The reserved validation lights are excluded. Light positions are shared with the main benchmark, while test-view receivers are sampled separately for each light rather than sharing

Table 8: **Common-segment evaluation by scene.** Query cost is synchronized wall time for 16,384 point queries; the 18-light estimate sums separately measured updates and queries. Fixed rows report accuracy only.

Scene / operator	Point MAE	Area MAE	Query (μs)	18 lights, est. (ms)
bedroom / Ray tracing (GPU BVH)	0.0000	0.0000	729.4	13.13
bedroom / PCF (Reeves et al., 1987)	0.0027	0.0024	489.9	154.89
bedroom / VSM (Donnelly & Lauritzen, 2006)	0.0071	0.0065	119.2	177.66
bedroom / DSM (Worchel & Alexa, 2023)	0.0072	0.0067	118.9	202.33
bedroom / Moment bounds (Worchel & Alexa, 2025)	0.0077	0.0073	138.7	279.45
bedroom / Hard distance (Hart, 1996)	0.0043	0.0056	3488.2	62.79
bedroom / Cone-soft (Quilez, 2010)	0.0195	0.0231	3684.3	66.32
bedroom / Density (Mildenhall et al., 2020)	0.0161	0.0259	3762.1	67.72
bedroom / **DSF Texture (ours)**	0.0021	0.0024	107.3	1.93
bedroom / **DSF Fixed (ours)**	0.0024	0.0029	–	–
livingroom / Ray tracing (GPU BVH)	0.0000	0.0000	569.8	10.26
livingroom / PCF (Reeves et al., 1987)	0.0028	0.0023	479.9	175.01
livingroom / VSM (Donnelly & Lauritzen, 2006)	0.0078	0.0078	118.5	198.26
livingroom / DSM (Worchel & Alexa, 2023)	0.0081	0.0081	118.3	232.03
livingroom / Moment bounds (Worchel & Alexa, 2025)	0.0085	0.0086	141.4	300.30
livingroom / Hard distance (Hart, 1996)	0.0054	0.0046	3294.7	59.31
livingroom / Cone-soft (Quilez, 2010)	0.0337	0.0315	3490.5	62.83
livingroom / Density (Mildenhall et al., 2020)	0.0281	0.0263	3563.8	64.15
livingroom / **DSF Texture (ours)**	0.0033	0.0021	117.6	2.12
livingroom / **DSF Fixed (ours)**	0.0038	0.0024	–	–

Table 9: **Batch-size timing audit.** Both synchronized wall and CUDA-event times include the query API. Inputs remain on the GPU. Larger batches repeat the same 16,384 receivers for one held-out light per scene. The per-segment average is throughput, not single-query latency.

Operator	Segments	Wall (μs)	Event (μs)	ns/segment
Ray tracing (GPU BVH)	16,384	603.3	594.8	36.82
Ray tracing (GPU BVH)	65,536	643.8	635.4	9.82
Ray tracing (GPU BVH)	262,144	1127.9	1119.7	4.30
PCF (Reeves et al., 1987)	16,384	488.2	477.1	29.79
PCF (Reeves et al., 1987)	65,536	920.5	909.0	14.05
PCF (Reeves et al., 1987)	262,144	3505.2	3492.3	13.37
VSM (Donnelly & Lauritzen, 2006)	16,384	121.8	111.8	7.43
VSM (Donnelly & Lauritzen, 2006)	65,536	201.4	191.3	3.07
VSM (Donnelly & Lauritzen, 2006)	262,144	673.9	662.3	2.57
DSM (Worchel & Alexa, 2023)	16,384	124.2	114.0	7.58
DSM (Worchel & Alexa, 2023)	65,536	204.1	193.9	3.11
DSM (Worchel & Alexa, 2023)	262,144	672.3	660.4	2.56
Moment bounds (Worchel & Alexa, 2025)	16,384	139.8	129.4	8.53
Moment bounds (Worchel & Alexa, 2025)	65,536	234.7	224.0	3.58
Moment bounds (Worchel & Alexa, 2025)	262,144	811.6	798.8	3.10
Hard distance (Hart, 1996)	16,384	3311.6	3301.1	202.12
Hard distance (Hart, 1996)	65,536	13104.0	13093.2	199.95
Hard distance (Hart, 1996)	262,144	50648.7	50633.1	193.21
Cone-soft (Quilez, 2010)	16,384	3517.7	3507.4	214.70
Cone-soft (Quilez, 2010)	65,536	13929.8	13919.3	212.55
Cone-soft (Quilez, 2010)	262,144	53643.9	53630.1	204.64
Density (Mildenhall et al., 2020)	16,384	3591.8	3581.2	219.23
Density (Mildenhall et al., 2020)	65,536	14231.7	14219.9	217.16
Density (Mildenhall et al., 2020)	262,144	54840.2	54827.7	209.20
DSF Texture (ours)	16,384	96.4	87.6	5.88
DSF Texture (ours)	65,536	116.5	108.1	1.78
DSF Texture (ours)	262,144	296.6	287.4	1.13

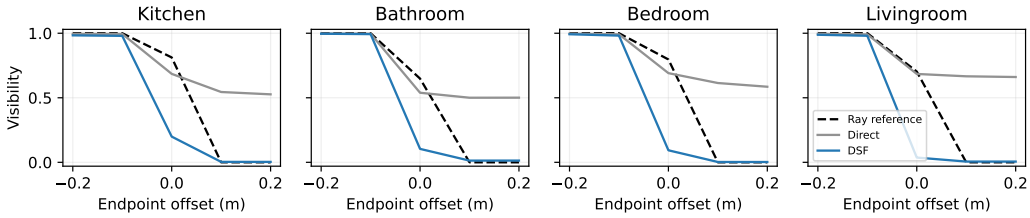


Figure 8: **Visibility across an occluder.** Mean Fixed response on current fields; endpoint offsets are relative to the first ray hit.

its smaller common receiver set. After a 3 mm normal offset, each origin moves 5 cm toward its light; the actual light endpoint is retained. All operators and the ray teacher use this explicit segment with zero additional trimming. Fields and learned query multipliers remain frozen, with common base radius/width scales of 0.25/0.10. Texture uses a 256-step cap and Fixed uses 1,024 samples. Metrics average equally over lights within each scene and then over the four scenes.

Table 10: **Independent endpoint accuracy.** Current 1024^3 controls on 72 held-out lights, with 100k receivers per light and a common ray reference.

Configuration	MAE ↓	Occlusion IoU ↑	Acc. (%)
Direct, Texture	0.0465	0.896	96.99
Direct, Fixed	0.0861	0.381	90.90
Scales only, Texture	0.0170	0.916	99.08
Scales only, Fixed	0.0220	0.920	98.05
Field only, Texture	0.0170	0.930	99.45
Field only, Fixed	0.0235	0.943	99.40
DSF (ours), Texture	0.0056	0.945	99.56
DSF (ours), Fixed	0.0034	0.965	99.74

Table 11: **Per-scene visibility MAE.** All four controls share query endpoints and base query scales.

Scene / backend	Direct	Scales only	Field only	**DSF**
Kitchen, Texture	0.1153	0.0468	0.0353	0.0108
Kitchen, Fixed	0.2051	0.0645	0.0490	0.0061
Bathroom, Texture	0.0314	0.0076	0.0130	0.0032
Bathroom, Fixed	0.0600	0.0094	0.0220	0.0022
Bedroom, Texture	0.0134	0.0058	0.0074	0.0028
Bedroom, Fixed	0.0277	0.0053	0.0083	0.0019
Livingroom, Texture	0.0258	0.0077	0.0123	0.0055
Livingroom, Fixed	0.0514	0.0089	0.0146	0.0036

Finite-endpoint behavior. Figure 8 moves the endpoint across the first occluder on up to 256 fixed rays per scene. Offsets are relative to the first ray hit, and every prediction is compared with the same finite segment's ray visibility. This tests the response to endpoint distance; an exact first-hit depth can also distinguish the two sides of an occluder.

A.2.3 Real Scanned Objects

We used one scanned mesh for each of the fourteen Stanford-ORB objects (Kuang et al., 2023). Each released mesh has 200,000 triangles; we retained all triangles and normalized its bounding-box diagonal to one. This evaluates visibility on real captured geometry. The ray targets are computed from these scans, independently of the dataset's RGB images. For each object, twenty independently sampled light endpoints are split into two validation and eighteen test lights. A common set of 16,384 surface receivers is used at each test light. Query origins have a 0.001 normal offset followed by a 0.002 offset toward the light; there is no additional endpoint shortening.

We constructed a 512^3 field with 10% padding along each bounding-box axis, plus 0.001. Joint calibration uses 10k Adam updates at 10^{-2}, batches of 65,536, and an L_1 visibility loss plus 10^{-3} times the mean absolute raw residual. Training samples a separate two-million-point surface pool and light endpoints uniformly in direction, with radius uniform on $[0.65, 1.5]$. Fixed uses 1,024 samples during calibration; Texture uses 256 steps. The direct-field control queries the uncalibrated UDF with the same Texture evaluator. Each field's deployment radius and width scales are selected on the two validation lights from $\{0.04, 0.125, 0.25\} \times \{0.01, 0.025, 0.1\}$. PCF uses 2048^2 cube faces and nine comparisons, with depth bias selected from zero, one, or two field-cell diagonals. All methods share the test segments and two-sided ray reference. Timing follows Appendix A.2.1.

Table 12: **Visibility on Stanford-ORB scans.** Fourteen objects, 18 held-out lights per object, and 16,384 segments per light. Fields use 512^3 grids; PCF uses 2048^2 faces. Query times are in μs.

Object	**DSF Texture (ours)**			Direct UDF			PCF		
	MAE	IoU	Query	MAE	IoU	Query	MAE	IoU	Query
baking	0.0076	0.989	58.3	0.0372	0.959	65.1	0.0093	0.995	488.7
ball	0.0106	0.984	62.3	0.0736	0.900	66.6	0.0143	0.993	482.1
blocks	0.0065	0.991	59.9	0.0494	0.941	65.7	0.0082	0.996	484.7
cactus	0.0103	0.984	59.5	0.0398	0.946	64.2	0.0145	0.992	481.2
car	0.0098	0.986	63.4	0.0577	0.931	68.6	0.0126	0.994	482.4
chips	0.0079	0.989	59.8	0.0691	0.910	71.0	0.0089	0.996	481.8
cup	0.0066	0.992	58.9	0.0518	0.947	67.8	0.0074	0.997	480.3
curry	0.0055	0.991	58.4	0.0120	0.983	60.2	0.0078	0.996	479.5
gnome	0.0108	0.983	59.7	0.0557	0.929	64.1	0.0152	0.992	480.6
grogu	0.0082	0.990	60.0	0.0623	0.928	66.8	0.0090	0.996	483.5
pepsi	0.0162	0.977	60.0	0.0299	0.965	62.8	0.0112	0.994	484.8
pitcher	0.0083	0.990	60.2	0.0517	0.945	68.8	0.0084	0.997	481.3
salt	0.0106	0.982	60.7	0.0469	0.939	64.8	0.0142	0.993	480.5
teapot	0.0107	0.984	61.1	0.0588	0.924	67.8	0.0141	0.993	479.6
Macro-average	0.0093	0.986	60.2	0.0497	0.939	66.0	0.0111	0.994	482.2

A.3 Differentiable Light Optimization

Queries and initialization. We reused the twenty fixed light positions from the application test, with 4,096 front-facing mesh receivers per light. Each receiver retains its 3 mm normal offset and is moved 5 cm along the true light direction to define a fixed query endpoint. This endpoint remains fixed throughout optimization. CPU ray tracing and every operator evaluate the same explicit segment to the light, with no additional endpoint trimming. The same stored initialization directions are used for all methods: two directions at each of 10, 20, and 80 cm, yielding 120 paired starts.

Operators. Operator definitions and forward-query settings are given in Appendix A.2.1; below we specify their use for light optimization. DSM uses the official differentiable shadow-map implementation; the moment-bound adapter uses the official solver and four cube-map moments, as in Table 1. We verified the DSM adapter against the author query API for both predictions and light-position gradients. Maps are rebuilt inside the gradient graph at every step and use 2048^2 faces. Density and cone-soft retain their validation-selected Table 1 settings. DSF uses the same newly calibrated fields as the forward benchmark, with its 1,024-sample Fixed backend and trained query scales. Uncalibrated DSF uses the same Fixed evaluator with the initial distance field and base query scales, without fitting either component. Hash-MLP reuses the frozen Table 1 checkpoints, differentiating through the hash encoding and MLP to optimize only the light endpoint. It uses the same observations, initial positions, loss, and Adam protocol. All operators retain their forward-selected query settings: this tests reuse for inference, rather than each method's best inverse-rendering configuration. DSM represents the variance-shadow-map family in this test; PCF, hard distance, and the tested BVH do not expose ordinary light-position gradients.

Optimization and model selection. Adam runs for at most 300 steps, stopping after 40 steps without an objective improvement exceeding 10^{-7}. For each operator, one global learning rate is selected from 0.005 and 0.02 using the eight reserved validation lights, a 20 cm initialization, and

100 optimization steps. For DSM and moment bounds, a follow-up validation sweep also tests 10^{-4}, 5×10^{-4}, and 10^{-3}; the selected rates remain unchanged. The zero success rates denote failed recovery within the 5 cm threshold, not zero learning rates: DSM uses 0.02 and moment bounds uses 0.005. The rate decays linearly to one tenth of its initial value. Selection uses mean validation position error among numerically stable candidates and precedes test optimization. Nonfinite test losses or gradients count as unsuccessful runs. The returned position minimizes the observed visibility loss along the trajectory; ground-truth position is never used to select an iterate.

Gradient-free control. CMA-ES uses population eight, initial standard deviation 5 cm, and at most 1,200 objective evaluations. Both DSF optimizers use the same Fixed objective. Their iteration and evaluation budgets differ and are reported explicitly.

Metrics and timing. Table 13 includes synchronized optimization cost with warmed kernels. Shadow-map updates are included. Success intervals bootstrap the twenty light clusters, retaining all starts within each sampled cluster; they do not treat starts as independent scenes. Table 14 breaks down recovery by scene.

Table 13: **Matched light recovery and optimization cost.** 120 paired starts; intervals bootstrap 20 light clusters. Step reports median per-case average time per Adam iteration or CMA-ES generation; Total is mean optimization time per start. Hash-MLP was timed on an RTX 4090; the other rows use an RTX PRO 6000.

Operator	Success (%, 95% CI)	Median (cm)	LR	Step (ms)	Total (s)
Density	89.2 [79.2, 96.7]	2.04	0.02	2.9	0.46
Cone-soft	18.3 [6.7, 31.7]	7.85	0.005	3.1	0.64
DSM	0.0 [0.0, 0.0]	20.00	0.02	42.8	2.05
Moment bounds	0.0 [0.0, 0.0]	20.00	0.005	51.1	2.34
Hash-MLP	7.5 [2.5, 13.3]	16.46	0.02	0.7	0.08
DSF uncalibrated	70.8 [59.2, 82.5]	2.35	0.02	10.2	1.33
DSF + Adam (ours)	89.2 [80.8, 96.7]	1.01	0.02	10.1	1.68
DSF + CMA-ES (ours)	96.7 [93.3, 99.2]	0.93	–	39.3	4.87

Table 14: **Matched light recovery by scene.** Success (%, error below 5 cm) / median position error (cm), over 30 paired starts per scene.

Operator	Kitchen	Bathroom	Bedroom	Livingroom
Density	83.3 / 2.1	86.7 / 2.0	100.0 / 2.2	86.7 / 1.0
Cone-soft	26.7 / 5.2	0.0 / 13.8	10.0 / 6.9	36.7 / 5.8
DSM	0.0 / 20.0	0.0 / 20.0	0.0 / 20.0	0.0 / 20.0
Moment bounds	0.0 / 20.0	0.0 / 20.0	0.0 / 20.0	0.0 / 20.0
Hash-MLP	3.3 / 19.4	13.3 / 12.5	3.3 / 20.0	10.0 / 13.1
DSF uncalibrated	73.3 / 2.9	43.3 / 6.9	96.7 / 1.0	70.0 / 2.6
DSF + Adam (ours)	93.3 / 1.4	73.3 / 0.9	100.0 / 0.7	90.0 / 1.0
DSF + CMA-ES (ours)	96.7 / 1.2	93.3 / 0.9	100.0 / 0.5	96.7 / 0.8

A.4 Reuse Across Inverse-Rendering Systems

A.4.1 Shared Evaluation Protocol

All hosts use the four TensoIR-Synthetic scenes (hotdog, armadillo, ficus, lego) and the five held-out environment maps of the TensoIR relighting protocol (bridge, city, fireplace, forest, night) over the 200 test views, except TensoIR, which renders every second view. Relit is PSNR of the relit image against the dataset ground truth. Shadow is the mean absolute error of the rendered visibility against a mesh-traced reference over all foreground pixels of all test views and environments: for every test pixel the reference is the unoccluded fraction of incident irradiance, $\sum_k V_k L_k \cos\theta_k / \sum_k L_k \cos\theta_k$ over 4096 Fibonacci directions with a 2 mm surface offset on the authors' released meshes, and the host's own visibility is reduced with the host's own sampling weights, so that both arms of a host

are compared on the quantity the host actually integrates. Table 2 reports quality. IRGS, R3DG, and LumiGauss quality is re-evaluated with frozen host checkpoints. IRGS and R3DG ran on an RTX 4090 and LumiGauss on an RTX PRO 6000, with each native/DSF pair on the same hardware. These three hosts use 384^3 fields: IRGS uses its extracted mesh with 10k calibration steps, while R3DG and LumiGauss retain direct Gaussian-proxy fields. No external geometry enters their field construction.

A.4.2 Host-Specific Integration

TensoIR. The native arm is the authors' released checkpoint. The DSF arm is retrained from scratch with the official configuration, with the visibility term of the direct-light shading replaced by a field query; indirect illumination keeps the host's marching during training, and at evaluation the transmittance call is replaced as well, so the evaluated model contains no marching. The grid is built from the mesh exported from the native checkpoint. Because this host is retrained rather than hot-swapped, the two arms differ in training as well as in the operator. Both arms were timed on the same GPU over 100 test views and five lights: the visibility operation (direct visibility and transmittance) takes 1556 ms per frame with marching and 365 ms with the field. These timings measure evaluation-time visibility, not training throughput; training retains the host's indirect-light marching. In a separate training-speed test, jointly replacing visibility and indirect-light evaluation increased BRDF-stage throughput from 7.1 to 24 iterations/s ($3.4\times$); this configuration differs from the visibility-only replacement used for the reported quality comparison.

IRGS. Both arms share one checkpoint. The host computes incident visibility inside its rendering equation by tracing rays against the Gaussians; the DSF arm replaces the visibility output with a field query at the same sample directions and weights. The native trace also returns material and normal attributes for indirect illumination, so the DSF arm retains this trace. Native Occl. includes this joint tracing work, while DSF Occl. times the field query alone; their ratio is not an end-to-end speedup: the added query increases the host's rendering cost. CUDA events time the component operations; synchronized wall time around the rendering call excludes image I/O and metric computation. Three untimed renders per light provide warmup and a repeatability control. The grid is built from the TSDF mesh extracted from the checkpoint with the extractor's default connected component filtering. The field is calibrated against mesh-traced supervision for 10k steps with 1024 samples per training segment; this does not update the host checkpoint.

R3DG. The host bakes per-Gaussian visibility once over hemispherical Fibonacci directions with a BVH ray tracer. Both arms then use the same rendering loop to multiply incident light by stored visibility. The DSF arm replaces the bake with field queries on the same directions and reuses the native checkpoint without any gradient step. Table 15 reports amortized bake cost. The field uses direct Gaussian-proxy construction without mesh-traced calibration.

LumiGauss. The native renderer contracts learned per-Gaussian spherical-harmonic transfer with the environment, with occlusion already folded into the fit. DSF reuses the checkpoint and appearance model but evaluates field visibility at shading time. This adds an operation absent from the native renderer and increases rendering cost. Environment orientation is calibrated once on the native arm and reused unchanged for DSF.

A.4.3 Timing Boundaries

Visibility-stage operations. Occl. is the visibility computation per rendered frame; for hosts that bake visibility it is the bake time amortized over the frames of the test set. Operation timings are reported in Table 15. R3DG visibility baking is amortized over 1000 frames. Native LumiGauss Occl. measures transfer retrieval and its SH contraction with illumination, not a standalone ray query. IRGS timing uses 20 evenly spaced test views under each of five lights per scene, with warmup and repeatability checks excluded; quality still uses all test views.

Per-segment query cost. The Query column of Table 2 normalizes cost per segment. For the native operators, the per-frame visibility time of Table 15 is divided by the number of segments the host issues per frame: TensoIR counts every transmittance query (44.2 M per frame on average), IRGS shades each pixel with nonzero alpha along 768 incident directions (111 M per frame), and

Table 15: **Host-dependent costs.** Visibility-stage operation costs within each host; R3DG baking is amortized over 1000 frames.

Host	Arm	Visibility operation (ms)
IRGS	native	1005.508
IRGS	**DSF (ours)**	56.504
R3DG	native	3.351
R3DG	**DSF (ours)**	0.302
LumiGauss	native	0.142
LumiGauss	**DSF (ours)**	44.193

R3DG bakes 384 directions per Gaussian (73 M per bake). The IRGS native value includes the attribute gathering of its joint trace. For DSF, each host's field is queried with the same set per scene, 10^5 points sampled on the TensoIR mesh with 256 hemisphere directions each, in a single call of the Texture backend on one RTX PRO 6000 (median of seven repetitions). Per scene the DSF values range from 0.33 to 1.61 ns; the spread across hosts reflects the stored field, since the Gaussian-proxy fields of R3DG hold smaller distances and require more steps than the calibrated mesh field of IRGS. Inside a host the same queries cost more when issued in small chunks: on the lego fields, calls of 10^5 segments add 1.3 to 1.7 ns per segment over a single call.

A.4.4 Relighting Examples

Figures 9 and 10 compare native and DSF relighting across all four scenes under bridge illumination, complementing the visibility maps in Figure 5.

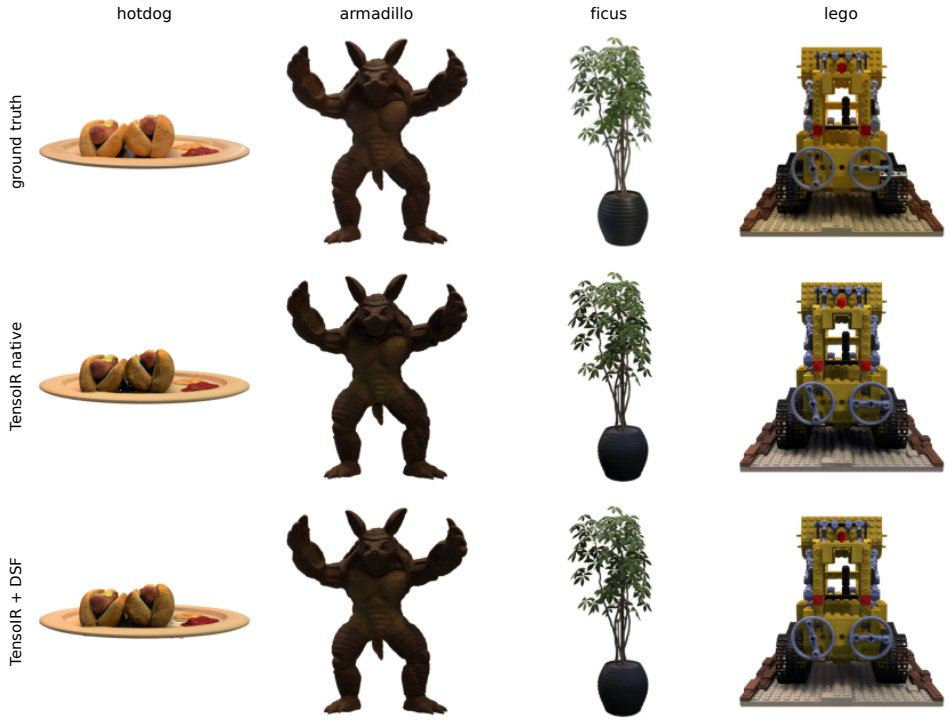


Figure 9: **TensoIR relighting: bridge, view 100.** Rows: ground truth, native, and DSF-retrained models. Columns: four test scenes.



Figure 10: **R3DG relighting: bridge, view 100.** Rows: ground truth, native, and DSF. Columns: four test scenes.

A.5 Modular Composition

A.5.1 Rigid Object Editing

The 0.6 m object has a separately constructed 96^3 field, built in 0.61 seconds. Four predetermined rigid-object/light states use a 1280×640 view and sixteen samples of a spherical light with 3 cm radius. Each frame queries both fields and combines segment visibility using Eq. (6) before emitter averaging. The scene field is the application's frozen 10k-calibrated 1024^3 checkpoint.

For the reference, primary intersections use the complete scene and transformed object meshes, and shadow rays traverse their union. Table 16 reports full-frame visibility MAE, the IoU of newly added shadows on scene surfaces, and the object-only shadow-footprint IoU. These measure distinct effects: a faithful object field does not remove errors already present in scene visibility. The field query and ray reference retain their respective surface and light-clearance conventions, so the union comparison includes this deployment approximation. The original third-party mesh and all poses are unchanged between the reference and prediction. Warm visibility-stage timing uses five repetitions and excludes primary ray intersections, material shading, and image output. Figure 6 uses supersampled PBR detail views of the first three states with a shared exposure and mesh vertex colors as object basecolor; its FPS labels retain this benchmark resolution.

Table 16: **Composition against union geometry.** Four fixed-field states with sixteen emitter samples. IoU measures added shadows or object footprints; timing covers visibility queries at 1280×640.

State	Union MAE	Added-shadow IoU	Object footprint IoU	Visibility (ms)
1	0.0511	0.866	0.899	5.20
2	0.0506	0.823	0.908	5.16
3	0.0855	0.760	0.847	5.45
4	0.0648	0.274	0.941	5.08

A.5.2 Composition Stress Tests and Error Attribution

Kitchen and bathroom each use three object shapes, normalized to 0.6 m height and represented by independent 96^3 direct fields. The frame has 1.2 cm bars; the torus introduces a hole, and the armadillo supplies an irregular surface. A geometry-only rule selects an upward-facing floor point near the horizontal median of floor samples. The three nearest existing test lights are fixed before visibility evaluation. Each object uses yaw $0°$, $45°$, or $90°$, with its base at the floor or 3 cm above it. Six poses and three lights give eighteen states per scene–object pair and 108 states in total.

Each state evaluates 10^4 scene receivers within 1.5 m horizontally of the placement and 10^4 area-uniform object-surface receivers. Scene points within 3 mm of the object mesh are excluded geometrically. Visibility averages sixteen fixed samples of a 3 cm-radius light. Queries use Texture with a 2 cm light trim. The scene retains its 5 cm start trim; object fields retain their grid-scale default, $\max(5\,\mathrm{mm}, 1.25\,\ell)$, where ℓ is their cell diagonal. The union ray reference starts at the same 3 mm normal-offset point and uses a 1 mm light tolerance. Thus this test includes the fields' deployment approximation. Its sampled-receiver MAE is distinct from the image-pixel MAE of Table 16.

For each segment, the scene-oracle control computes $\min(V^{\mathrm{ray}}_{\mathrm{scene}}, V^{\mathrm{DSF}}_{\mathrm{object}})$; the object-oracle control substitutes the opposite term. We check that the minimum of both binary ray terms equals a separate ray query through the complete union. Substitution and composition precede emitter aggregation. Added-shadow IoU pools intersection and union counts across all eighteen preset states per scene–object pair, retaining zero-positive states (Table 18).

Table 17: **Composition error attribution.** Union-reference MAE over 108 states. DSF separates scene/object receivers; oracles replace one visibility term on scene receivers.

Scene / object	**DSF scene (ours)**	**DSF object (ours)**	Scene oracle	Object oracle
Kitchen / armadillo	0.0600	0.1569	0.0014	0.0586
Kitchen / torus	0.0595	0.1653	0.0010	0.0585
Kitchen / thinframe	0.0600	0.1320	0.0014	0.0586
Bathroom / armadillo	0.0117	0.1400	0.0073	0.0044
Bathroom / torus	0.0095	0.1758	0.0051	0.0044
Bathroom / thinframe	0.0076	0.1198	0.0032	0.0044

Table 18: **Added-shadow IoU on scene receivers.** Counts pool 18 states per scene/object; added shadows threshold the scene-only minus union visibility at 0.5.

Scene / object	**DSF (ours)** IoU	Scene oracle	Object oracle
Kitchen / armadillo	0.8732	0.8783	0.9942
Kitchen / torus	0.9333	0.9424	0.9899
Kitchen / thinframe	0.2853	0.2853	0.9875
Bathroom / armadillo	0.8898	0.8898	1.0000
Bathroom / torus	0.9271	0.9271	1.0000
Bathroom / thinframe	0.5621	0.5621	1.0000

Table 19: **Attribution across four rendered states.** Union-reference MAE and added-shadow IoU; each oracle replaces one visibility term before emitter averaging.

State	**DSF (ours)** MAE	**DSF (ours)** IoU	Scene oracle IoU	Object oracle IoU
1	0.0511	0.8662	0.9048	0.9562
2	0.0506	0.8225	0.8986	0.9104
3	0.0855	0.7604	0.8348	0.8573
4	0.0648	0.2736	0.9005	0.2906

The thin-frame shadow errors largely follow the object term: replacing object visibility raises its added-shadow IoU from 0.285 to 0.988 in kitchen and from 0.562 to 1.000 in bathroom. Object-surface visibility also remains less accurate than scene-surface visibility across all six scene–object

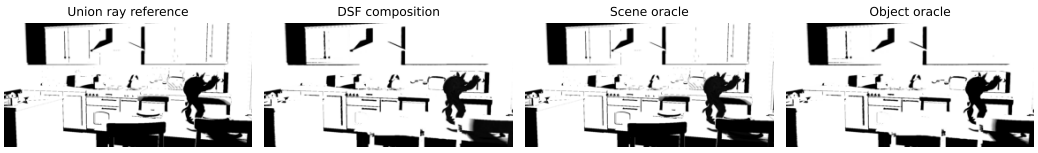


Figure 11: **Composition error attribution, state 4.** Union ray reference, DSF, scene oracle, and object oracle; quantitative results in Table 19.

pairs (DSF MAE 0.120–0.176 versus 0.008–0.060). These controls localize the approximation error without separating object-grid resolution, sampling, and start-clearance effects.

The original lowest-overlap rendered state has a different failure mode. Its added-shadow IoU rises from 0.274 to 0.900 with scene-ray substitution, but only to 0.291 with object-ray substitution. Thus the scene term is the dominant limitation for that edit, and its low overlap cannot be attributed to the inserted object field alone.

A.5.3 Nonrigid Object Updates

Figure 6e keeps the camera, light, and object pose of panel (d), and deforms the object in its local coordinates. For normalized height $u \in [0, 1]$, we rotate the horizontal coordinates by θu^2 and add bu^2 to the horizontal x coordinate, with $\theta = 25^\circ\big(b/(0.22\,\mathrm{m})\big)$. Topology and vertex colors are preserved; normals are recomputed. Each frame constructs a fresh direct object field from the deformed mesh and uploads its texture. The object field uses geometric distance initialization with prescribed query scales, so neither its initial construction nor subsequent shape updates require training. The background 1024^3 field retains its one-time 10k-step calibration throughout the sequence.

Resolution–accuracy–speed tradeoff. We evaluated 96^3, 64^3, 48^3, and 32^3 object grids over the same 25-frame closed cycle, $b_t = 0.11[1 - \cos(2\pi t/24)]\,\mathrm{m}$ for $t = 0, \ldots, 24$. Geometry, light samples, and ray references are shared across resolutions. Each grid follows the same cell-relative query-scale rule and uses the deformed mesh's bounding box; no per-resolution tuning or training is performed. Table 20 reports mean object-visibility MAE on scene receivers, grid size, and synchronized timing at 1280×640. MAE and update time are averaged over all 25 frames; FPS is the reciprocal of mean time, and update speedup is T_{96}/T_R. The 48^3 grid uses one eighth as many voxels as 96^3 and updates $3.83\times$ faster. Figure 6e shows its maximum-deformation frame, with cycle-mean timings. The PBR illustration is supersampled using the same exposure and crop as panels (b–d).

Table 20: **Object-grid resolution tradeoff.** Means over a 25-frame deformation cycle. Update speedup is relative to 96^3; Figure 6e uses 48^3.

Object grid	Voxels	MAE ↓	Update (ms) ↓	Update speedup ↑	Vis. FPS ↑	Full FPS ↑
DSF 96^3 **(ours)**	884,736	0.0072	603.1	$1.00\times$	183	0.92
DSF 64^3 **(ours)**	262,144	0.0105	236.1	$2.55\times$	184	1.41
DSF 48^3 **(ours)**	110,592	0.0140	157.5	$3.83\times$	185	1.59
DSF 32^3 **(ours)**	32,768	0.0188	115.6	$5.22\times$	185	1.71

Update attribution at 96^3. A separate five-frame test increases b uniformly from 0.066 to 0.22 m to compare a rebuilt 96^3 field with its undeformed counterpart and break down update costs. Table 21 evaluates its final frame against rays through the deformed object and scene union. Reusing the undeformed object grid provides a control on the identical queries. Rebuilding improves both object

visibility and its shadow footprint. Object MAE and footprint IoU use scene receivers; union MAE covers all receivers.

Table 21: **Visibility after object deformation.** Final-frame MAE with 96^3 object fields; footprint IoU uses scene receivers.

Object field	Object MAE	Union MAE	Footprint IoU
DSF, Unchanged (ours)	0.0346	0.1033	0.437
DSF, Rebuilt (ours)	0.0074	0.0868	0.824

Table 22 reports synchronized wall time on the CS RTX PRO 6000 workstation after warmup. Grid updates include mesh serialization, Open3D CPU distance-grid construction, and GPU texture upload. Complete frames additionally include deformation, normal updates, scene and object BVH construction, primary intersections, data transfers, visibility composition, and PBR shading to an image buffer. One-time loading, JIT compilation, reference evaluation, and image-file export are excluded. Grid construction is included in the grid-update row; FPS is the reciprocal of mean frame time across the five updates. The same timing boundaries apply to the resolution sweep above.

Table 22: **Nonrigid update and rendering costs.** Five-frame means with a 96^3 object grid at 1280×640; complete frames include updates and PBR rendering.

Operation	Time (ms)	FPS
Object geometry and normals	5.55	–
Distance-grid construction	591.87	–
Grid update, including upload	608.09	–
PBR frame after update	471.39	2.12
Complete deformation frame	1086.83	0.92
Rigid PBR frame	459.38	2.18

A.5.4 External Composition on SynCom

Proxy-geometry visibility. We used all four scenes and four objects released with ComGS (Gao et al., 2026). Poisson reconstruction (depth ten) converts the trained 2D Gaussian assets to meshes, retaining Gaussians with opacity above 0.1 and trimming the lowest 3% of Poisson densities. The scene fields use the 512^3/10k configuration in Appendix A.2.3; half of training light endpoints instead sample within the scene bounds, rejecting locations less than 0.01 from the proxy. Object fields are constructed independently at 96^3, calibrated once per object, and transformed using the released poses. Object calibration uses 10k Adam updates at 10^{-2}, batches of 65,536, and 1,024 samples, with the same visibility loss and residual penalty. Half of the training segments connect independently sampled exterior endpoints with radii in $[0.65, 2.5]$. The other half start at an exterior endpoint and pass through a uniform target in $[-0.5, 0.5]^3$, extending beyond the object. Coordinates use the normalized object frame. An independent set of 32,768 ray-labeled segments selects Texture radius and width scales from $\{0.04, 0.125, 0.25, 0.5\} \times \{0.01, 0.025, 0.1\}$. The resulting four fields are shared across all scenes and poses.

Each of the sixteen compositions receives eight independently sampled point lights at a radius of 2.5 transformed object diagonals. We report two receiver sets separately: 8,192 uniformly sampled scene points and 8,192 points nearest the object center, selected from a 131,072-point surface pool. Composed MAE compares the minimum of the two field queries against rays through their combined proxy geometry. Conditional added-shadow IoU uses ray-reference scene visibility to isolate the object field's error; intersections and unions are pooled over lights within each case. These geometric targets are distinct from the RGB targets below.

Native ComGS rendering. We additionally rendered the sixteen released compositions at three predetermined views each (0000, 0024, and 0048). The native renderer retains its frozen scene and object appearance, environment maps, object self-occlusion, and composition logic. Only the object cast-shadow query changes: native surface probes, a calibrated 96^3 DSF object field, or two-sided rays through the same reconstructed object proxy. The ray variant is a geometric diagnostic, not

Table 23: **Composition on SynCom proxy geometry.** Sixteen configurations, eight lights each. MAE measures composed visibility; added-shadow IoU conditions on ray-reference scene visibility. Each receiver set has 8,192 points; query time covers both sets.

Configuration	Uniform receivers		Near-object receivers		Query
	MAE	Added IoU	MAE	Added IoU	(μs)
artwall / bottle	0.0500	1.000	0.0489	0.986	87.7
artwall / horse	0.0520	0.961	0.0612	0.927	86.7
artwall / kettle	0.0525	0.955	0.0560	0.985	86.9
artwall / toy	0.0527	0.960	0.0561	0.976	86.9
attic / bottle	0.0478	0.864	0.0545	0.981	86.9
attic / horse	0.0497	0.955	0.0621	0.956	86.3
attic / kettle	0.0461	0.983	0.0587	0.982	87.9
attic / toy	0.0497	0.960	0.0608	0.974	87.1
forest / bottle	0.0583	0.667	0.1242	1.000	91.1
forest / horse	0.0537	0.875	0.1199	0.969	90.6
forest / kettle	0.0566	1.000	0.1229	0.972	91.7
forest / toy	0.0579	1.000	0.1245	0.974	90.3
room / bottle	0.0723	0.968	0.0664	0.990	88.8
room / horse	0.0710	0.929	0.0679	0.962	86.7
room / kettle	0.0760	1.000	0.0718	0.976	86.2
room / toy	0.0754	1.000	0.0758	0.977	89.0
Macro-average	0.0576	0.942	0.0770	0.974	88.2

ground-truth radiance. We compared all 48 rendered views per method with the official SynCom RGB targets using full-image PSNR and MAE, averaged equally over views. Object fields use only proxy-ray supervision; no parameters are fitted to these RGB images. The camera loader follows the official image-size and field-of-view conventions, including the 1270×700 room images.

Table 24: SynCom RGB composition on three predetermined views per scene–object pair (48 views). Native ComGS (Gao et al., 2026) materials, lighting, and object self-occlusion are retained; only cast-shadow visibility changes. Proxy ray tracing is a diagnostic using the extracted object mesh. All errors use official RGB images.

	Native SOP		**DSF (ours)**		Proxy ray (diagnostic)	
Scene / object	PSNR ↑	MAE ↓	PSNR ↑	MAE ↓	PSNR ↑	MAE ↓
Artwall / Bottle	23.15	0.0299	22.82	0.0311	22.82	0.0311
Artwall / Horse	24.16	0.0295	23.98	0.0297	24.00	0.0297
Artwall / Kettle	22.56	0.0321	23.19	0.0304	23.18	0.0305
Artwall / Toy	24.83	0.0249	25.49	0.0239	25.48	0.0239
Attic / Bottle	25.52	0.0314	25.90	0.0308	25.90	0.0308
Attic / Horse	26.29	0.0304	26.34	0.0303	26.37	0.0303
Attic / Kettle	25.18	0.0331	25.27	0.0330	25.26	0.0330
Attic / Toy	25.73	0.0318	25.87	0.0316	25.88	0.0316
Forest / Bottle	21.40	0.0542	21.40	0.0542	21.40	0.0542
Forest / Horse	21.15	0.0554	21.04	0.0558	21.04	0.0558
Forest / Kettle	21.04	0.0562	21.14	0.0559	21.14	0.0558
Forest / Toy	20.96	0.0561	21.03	0.0559	21.03	0.0559
Room / Bottle	23.44	0.0300	23.56	0.0297	23.56	0.0297
Room / Horse	24.13	0.0280	24.05	0.0281	24.06	0.0281
Room / Kettle	22.77	0.0321	22.76	0.0322	22.76	0.0322
Room / Toy	23.18	0.0304	23.28	0.0303	23.27	0.0303
Mean	23.47	0.0366	23.57	0.0364	23.57	0.0364

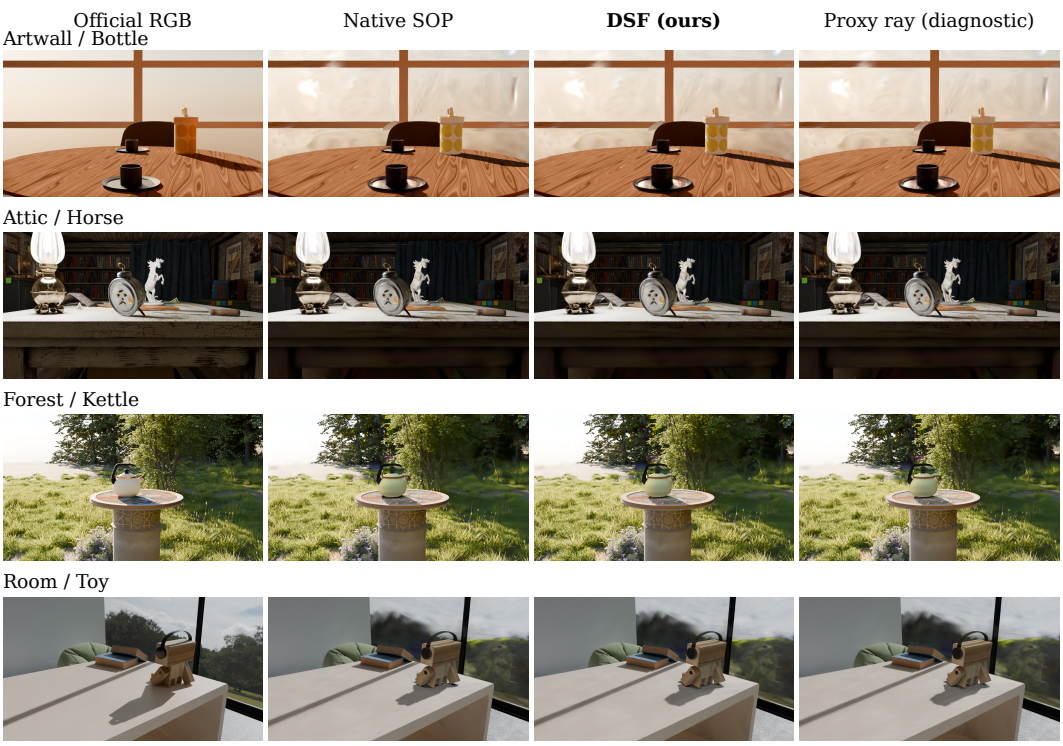


Figure 12: SynCom (Gao et al., 2026) composition (view 0000; Room/Toy uses 0024). Rows cover four scenes with different inserted objects. Native SOP, DSF, and proxy-ray rendering share materials and lighting and differ in cast-shadow visibility; proxy rays use the extracted object mesh.

A.6 Ablations and Additional Analyses

A.6.1 Field and Query-Scale Calibration

Calibration controls. The four controls isolate the two fitted components: Direct fits neither, Scales only fits the three global query parameters (r, w, τ), Field only fits $\boldsymbol{\rho}$, and Joint fits both. They share the 1024^3 distance initialization, evaluator, and initial radius and width scales of 0.25 and 0.10 cell diagonals. Fitted variants use the same 10k-update recipe and training queries, changing only the optimized parameter set. Validation-query hashes match, and independent test queries are reused byte-for-byte. Table 25 evaluates the common segments used in Table 1. Texture controls keep base radius and width scales fixed at 0.25 and 0.10, retaining learned multipliers wherever scales are fitted. The deployment-scale selection in the main comparison is not applied here.

Paired calibration effects. Table 26 compares the current controls on the 72 independent endpoint lights. We bootstrapped 10,000 draws, resampling lights within each scene and averaging the four scene means. Joint calibration lowers MAE relative to every control with both backends; all six intervals exclude zero. These intervals quantify light-sampling variability for fixed geometries and trained fields.

Calibration and light recovery. The four current controls share the 20 lights, 4,096 observations per light, and 120 starts at 10, 20, and 80 cm from Appendix A.3. Each uses Fixed with 1,024 samples and its trained query scales. The same eight validation lights select a global Adam learning rate per control from 0.005 and 0.02; all select 0.02. Optimization uses the same 300-step budget and stopping rule as the main comparison. No fields are retrained for inference.

Joint gives the lowest median position error against ray observations (1.01 cm), versus 1.24 cm for scales-only and 1.06 cm for field-only. Field-only succeeds on one more start than Joint. Replacing

Table 25: **Common-segment calibration ablation.** Each learned variant uses 10k updates with identical teacher/student segments. Texture uses the same base radius/width scales across variants; unlike Table 1, no additional deployment-scale selection is applied.

Scene	Variant	Point (T)	Area (T)	Point (F)	Area (F)
kitchen	Direct	0.1166	0.1522	0.2044	0.2917
kitchen	Scales only	0.0513	0.0469	0.0656	0.0721
kitchen	Field only	0.0473	0.0605	0.0570	0.0787
kitchen	**DSF (ours)**	0.0215	0.0138	0.0144	0.0066
bathroom	Direct	0.0316	0.0080	0.0600	0.0138
bathroom	Scales only	0.0086	0.0056	0.0103	0.0049
bathroom	Field only	0.0138	0.0053	0.0221	0.0056
bathroom	**DSF (ours)**	0.0039	0.0025	0.0028	0.0015
bedroom	Direct	0.0170	0.0192	0.0354	0.0425
bedroom	Scales only	0.0072	0.0106	0.0068	0.0085
bedroom	Field only	0.0093	0.0127	0.0106	0.0133
bedroom	**DSF (ours)**	0.0035	0.0051	0.0024	0.0029
livingroom	Direct	0.0316	0.0285	0.0619	0.0567
livingroom	Scales only	0.0092	0.0075	0.0111	0.0093
livingroom	Field only	0.0140	0.0115	0.0171	0.0142
livingroom	**DSF (ours)**	0.0060	0.0036	0.0038	0.0024

Table 26: **Paired endpoint calibration effects.** Joint-minus-control MAE; 95% intervals bootstrap lights within each of four fixed scenes.

Control / backend	Δ MAE	95% low	95% high	Joint wins
Direct, Texture	-0.0409	-0.0655	-0.0210	72/72
Scales only, Texture	-0.0114	-0.0238	-0.0036	66/72
Field only, Texture	-0.0114	-0.0158	-0.0076	71/72
Direct, Fixed	-0.0826	-0.1197	-0.0519	72/72
Scales only, Fixed	-0.0186	-0.0363	-0.0062	71/72
Field only, Fixed	-0.0200	-0.0294	-0.0126	72/72

ray observations with each control's own predictions at the true light isolates recovery under self-generated observations; these results are reported alongside physical recovery in Table 27.

Table 27: **Calibration and light recovery.** Current controls share 120 starts and the Fixed evaluator. Ray targets test physical recovery; self targets use each operator's own visibility. Success requires error below 5 cm.

Configuration	Ray success (%)	Median (cm)	Self success (%)	Median (cm)
Direct	70.8	2.35	86.7	0.16
Scales only	85.8	1.24	90.0	0.12
Field only	**90.0**	1.06	93.3	0.09
DSF (ours)	89.2	**1.01**	90.0	0.11

A.6.2 Field and Evaluator Substitutions

Table 28 uses the current checkpoints and the common point- and area-light queries of Appendix A.2.1. Hard distance, cone-soft, and density use 1,024 midpoint samples. Their parameters are selected per scene on the two reserved validation lights, using the candidate sets in Appendix A.2.1. Cone-soft is also evaluated on the calibrated field with independently validation-selected coefficients (Table 29). DSF Fixed and Texture retain the base radius and width scales of 0.25 and 0.10, without the extra Texture deployment tuning used in Table 1. The calibrated field improves cone-soft predictions, while the Fixed DSF query gives the lowest point and area MAE among these controls.

Table 28: **Field and evaluator substitutions.** Common-segment MAE; direct-query and cone-soft parameters use reserved validation lights. DSF Fixed/Texture retain the shared base query scales.

Operator	Point MAE	Area MAE
Hard distance (Hart, 1996)	0.0224	0.0245
Cone-soft (Quilez, 2010)	0.0296	0.0243
Density (Mildenhall et al., 2020)	0.0424	0.0510
DSF field, cone-soft	0.0066	0.0053
Direct field, Fixed	0.0904	0.1012
DSF Fixed (ours)	0.0059	0.0034
DSF Texture (ours)	0.0087	0.0063

Table 29: **Selected evaluator parameters.** Hard thresholds use cell diagonals; density widths use the maximum cell edge.

Scene / operator	Threshold	Cone	Amplitude	Width
Kitchen, direct hard	0.3	–	–	–
Kitchen, direct cone	–	32	–	–
Kitchen, direct density	–	–	100	0.25
Kitchen, joint cone	–	128	–	–
Bathroom, direct hard	0.3	–	–	–
Bathroom, direct cone	–	16	–	–
Bathroom, direct density	–	–	100	0.5
Bathroom, joint cone	–	128	–	–
Bedroom, direct hard	0.3	–	–	–
Bedroom, direct cone	–	32	–	–
Bedroom, direct density	–	–	100	0.5
Bedroom, joint cone	–	128	–	–
Livingroom, direct hard	0.3	–	–	–
Livingroom, direct cone	–	64	–	–
Livingroom, direct density	–	–	100	0.25
Livingroom, joint cone	–	128	–	–

A.6.3 Resolution and Execution Backends

The 384^3 and 768^3 fields are freshly calibrated for 10k updates using the same matched-segment recipe as the current 1024^3 fields. All resolutions share base radius and width scales of 0.25 and 0.10 and the same test queries. Table 30 reports Texture accuracy, query cost, and float32 grid-value storage; the latter excludes optimizer state, additional field buffers, and workspace. Table 31 varies Fixed samples and compares the Sphere and Texture approximations on the current 1024^3 fields. Higher Fixed sample counts reduce aggregate prediction error at greater query cost. Timings average warmed 16,384-segment batches from one held-out light per scene on the RTX PRO 6000.

Table 30: **Grid-resolution comparison.** Texture with shared base query scales and 10k joint calibration. Values memory is one float32 grid, excluding optimizer state and workspace. Timing uses one 16,384-segment batch per scene.

Resolution	Direct point	Joint point	Joint area	Values (MiB)	ns/segment
384^3	0.0709	0.0235	0.0208	216	3.69
768^3	0.0468	0.0104	0.0083	1728	4.23
1024^3	0.0492	0.0087	0.0063	4096	5.77

Table 31: **Execution backends at 1024^3.** Common-segment MAE and warmed joint-field query-API cost; timing averages one 16,384-segment batch per scene. Texture represents the optimized forward deployment path, whereas Fixed is retained primarily for gradient validation and inverse optimization experiments.

Backend	Direct point	Joint point	Joint area	ns/segment
Fixed 192	0.0947	0.0205	0.0198	119.21
Fixed 512	0.0905	0.0060	0.0040	271.63
Fixed 1024	0.0904	0.0059	0.0034	542.31
Sphere	0.0492	0.0087	0.0063	18.33
Texture	0.0492	0.0087	0.0063	5.77

A.6.4 Sensitivity to Proxy Geometry

We retained 10% or 1% of faces, removed 5% of faces, or applied 1 or 3 cm vertex noise to each scene. Each of the 20 proxy fields uses 768^3 resolution and 10k matched-segment calibration against its own proxy rays. Direct and joint fields share query settings and are evaluated against rays through the original geometry; proxy ray casting isolates geometry error. Keeping 10% of faces changes aggregate error little; stronger decimation and perturbations degrade accuracy. Calibration improves over direct fields except under 3 cm vertex noise, where the corrupted teacher also limits the calibrated representation (Table 32).

Table 32: **Geometry-proxy sensitivity at 768^3.** Point/area MAE against original geometry (D: direct; J: joint). Joint uses 10k proxy-ray calibration; both field variants use Texture and identical base query scales.

Proxy	Point D	Point J	Point rays	Area D	Area J	Area rays
Original	0.0468	0.0104	0.0000	0.0577	0.0083	0.0000
10% faces	0.0468	0.0104	0.0003	0.0577	0.0082	0.0003
1% faces	0.0495	0.0151	0.0074	0.0607	0.0130	0.0074
5% faces removed	0.0536	0.0196	0.0117	0.0584	0.0082	0.0003
1 cm noise	0.0714	0.0433	0.0258	0.0810	0.0426	0.0248
3 cm noise	0.2024	0.2257	0.1782	0.1991	0.2190	0.1692

A.6.5 Query Cost and Source-Mesh Complexity

Table 33 reports forward query costs for GPU BVH, DSF Texture, and smoothed ray tracing without BVH. All methods use identical 16,384-segment batches. After two warmups, seven synchronized repetitions measure query-API time, including allocation and random sampling but excluding compilation and field loading. Hardware is identified in the table. These batch-average costs are not single-ray latency. Texture retains the ablation query scales; Table 1 additionally selects deployment scales and averages over more light positions.

The differentiable ray-based control scans mesh triangles without BVH or hardware RT and smooths binary visibility with Gaussian light-endpoint perturbations ($\sigma = 5\,\mathrm{cm}$). Antithetic score-function estimates use $M = 32$ or 128 traced segments per query to obtain light-position gradients. This is a custom unaccelerated estimator, not an optimized differentiable ray tracer. Table 34 compares this estimator with DSF Fixed ($K = 1024$) on the same RTX 4090, timing forward plus a vector–Jacobian product with respect to the light position and excluding optimizer updates.

For N_q queries, total cost includes preprocessing, $T_\mathrm{total} = T_\mathrm{build} + T_\mathrm{calibration} + N_\mathrm{q}\, t_\mathrm{query}$. Table 1 reports preparation separately: low query cost alone does not imply lower total runtime than BVH tracing.

Source-mesh tessellation. We subdivided each bedroom triangle into n^2 coplanar triangles using a uniform barycentric lattice, with $n \in \{1, 2, 3, 4, 6, 8, 10, 13\}$. The eight meshes range from 312,369 to 52,790,361 faces without adding geometric detail. They share one frozen 1024^3/10k field and 262,144 surface–emitter segments. The field is reused without reconstruction or recalibration;

Table 33: **GPU forward-query costs by capability.** Times in ns/segment for 16,384-segment batches. Smoothed RT uses no BVH; M counts traced samples per query.

	Forward-only visibility		Smoothed RT (no BVH)	
Scene	GPU BVH	**DSF Texture (ours)**	$M = 32$	$M = 128$
Kitchen	41.53	5.07	99748.53	327314.03
Bathroom	33.44	5.43	39418.33	131759.07
Bedroom	41.12	6.26	22071.15	72288.34
Livingroom	31.82	6.31	41966.80	138476.57

Forward-only group: RTX PRO 6000; smoothed RT: RTX 4090.

Table 34: **Differentiable visibility query costs.** Forward plus backward in ns/segment on RTX 4090, using identical 16,384-segment batches. Smoothed RT uses no BVH; M counts traced samples per query.

Scene	Smoothed RT ($M = 32$)	Smoothed RT ($M = 128$)	**DSF Fixed (ours)**
Kitchen	97383.87	328873.04	1524.90
Bathroom	39413.34	132149.60	1235.47
Bedroom	22079.40	71709.16	1474.61
Livingroom	41951.14	138080.68	1520.25

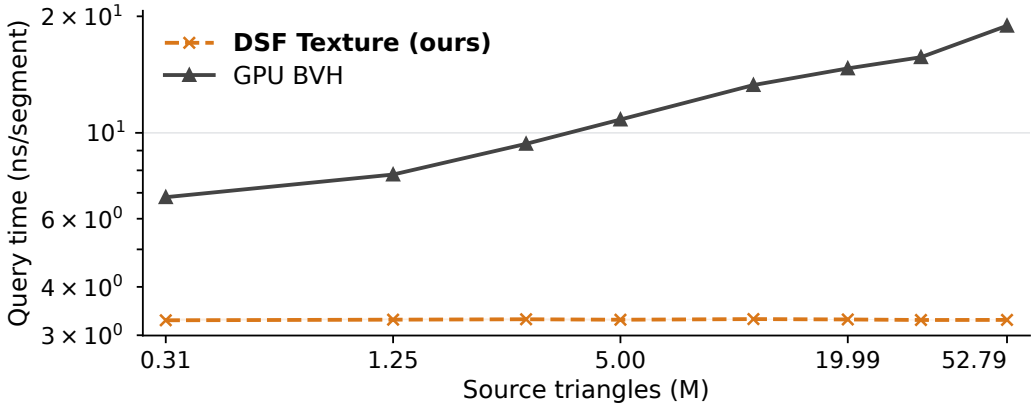


Figure 13: **Visibility-query cost versus source triangle count.** DSF Texture and GPU BVH query times for an unchanged surface and field. Bands show the interquartile range; Table 3 shows normalized costs.

the BVH is built for each tessellation. DSF Texture has a 1024-step cap. Both DSF Texture and BVH use a 5 cm start clearance and a 2 cm light clearance after the same 3 mm normal offset.

Figure 13 and Table 35 report synchronized query-API wall time on the same RTX PRO 6000, including output allocation and conversion. After warmup, each of 15 rounds times three calls per backend in randomized order; preprocessing, input preparation, and compilation are excluded. The GPU BVH returns closest hits with backface culling disabled, which are tested against segment length. Separate visible and occluded timings use fixed subsets of 123,846 and 138,298 queries, respectively. Surface area and bounds agree within floating-point tolerance; CPU ray-reference labels remain unchanged under subdivision, and the GPU BVH matches the reference on every query. DSF Texture predictions are identical across tessellations. Increasing the face count 169× raises BVH query time to 2.77× its original value, while DSF Texture remains nearly unchanged. This control tests dependence on source triangle count at fixed geometry, not equal-accuracy scaling as new geometric detail is introduced.

Table 35: **Source-mesh tessellation control.** Bedroom with an unchanged surface and a shared 1024^3/10k field. Query times are medians over 15 timing rounds; BVH uses closest-hit tracing.

		Query (ns/segment)		Visibility MAE
Triangles	BVH build (s)	BVH	**DSF Texture (ours)**	**Texture**
312,369	0.06	6.82	3.28	0.0420
1,249,476	0.29	7.80	3.29	0.0420
2,811,321	0.73	9.37	3.30	0.0420
4,997,904	1.25	10.82	3.29	0.0420
11,245,284	3.15	13.29	3.30	0.0420
19,991,616	5.41	14.67	3.30	0.0420
31,236,900	8.21	15.68	3.28	0.0420
52,790,361	15.67	18.91	3.29	0.0420

A.6.6 Near-Geometry Endpoint Stress

We sampled eight light endpoints per scene at 3–10 cm from geometry, using camera-facing surface normals. Candidates must lie inside the scene bounds and have an unobstructed short segment from the source surface; lights are at least 30 cm apart. Each light uses up to 10^4 front-facing receivers, excluding original segments shorter than 11 cm. Origins receive the same 3 mm normal and 5 cm light-direction offsets as the common benchmark; no light-end trim is applied. All four current controls remain frozen.

Table 36 reports equal-light mean errors. Joint and scales-only have similar Fixed error, while scales-only has lower Texture error. Thus the larger ordinary-endpoint gain does not transfer uniformly to near-geometry queries. Tables 37 and 38 report per-scene results and query-weighted strata by original segment length and grazing angle (normal–direction cosine below 0.1).

Table 36: **Near-geometry endpoints.** Current frozen controls on 32 lights at 3–10 cm from geometry; all operators share the reference segments.

Configuration	MAE ↓	Occlusion IoU ↑	Acc. (%)
Direct, Texture	0.0630	0.930	98.61
Direct, Fixed	0.1256	0.377	86.33
Scales only, Texture	0.0387	0.854	96.64
Scales only, Fixed	0.0265	0.940	98.90
Field only, Texture	0.0573	0.816	95.66
Field only, Fixed	0.0537	0.866	96.97
DSF (ours), Texture	0.0418	0.829	95.97
DSF (ours), Fixed	0.0264	0.884	97.33

Table 37: **Per-scene visibility MAE.** All four controls share query endpoints and base query scales.

Scene / backend	Direct	Scales only	Field only	**DSF**
Kitchen, Texture	0.0957	0.0446	0.0575	0.0335
Kitchen, Fixed	0.2150	0.0423	0.0629	0.0211
Bathroom, Texture	0.0305	0.0094	0.0180	0.0099
Bathroom, Fixed	0.0710	0.0091	0.0226	0.0043
Bedroom, Texture	0.0505	0.0523	0.0587	0.0419
Bedroom, Fixed	0.0769	0.0244	0.0394	0.0252
Livingroom, Texture	0.0756	0.0486	0.0948	0.0820
Livingroom, Fixed	0.1396	0.0302	0.0900	0.0550

Table 38: **Stress-test error strata.** MAE weighted by query count within each stratum.

Configuration	< 1 m	1–3 m	≥ 3 m	Grazing
Direct, Texture	0.0566	0.0638	0.0632	0.1556
Direct, Fixed	0.0970	0.1130	0.1377	0.1906
Scales only, Texture	0.0306	0.0509	0.0309	0.1918
Scales only, Fixed	0.0204	0.0281	0.0261	0.0904
Field only, Texture	0.0544	0.0778	0.0429	0.2710
Field only, Fixed	0.0422	0.0645	0.0473	0.2065
DSF (ours), Texture	0.0447	0.0645	0.0254	0.2561
DSF (ours), Fixed	0.0210	0.0420	0.0158	0.1634