



Faculty of Media, Science and Technology

National Centre for Computer Animation (NCCA)

Msc Computer Animation and Visual Effects

August / 2026

MSc Master Project 25/26

**Implementing and Validating Physically Based Rendering Techniques in an
Extensible Renderer**

By

Gabriela Zambrano Dávila

Abstract	1
1. Introduction	2
1.1 Motivation	2
1.2 Aim and scope	2
1.3 Contributions	3
1.4 Research questions	3
1.5 Thesis structure	3
2. Background and related work	4
2.1 Rendering equation and Monte Carlo path tracing	4
2.2 Variance reduction and practical image reconstruction	4
2.3 Physically based shading	5
2.4 Rendering architectures and research frameworks	6
3. System design and methodology	7
3.1 Layered architecture	7
3.2 Design principles	8
3.3 Validation ladder	8
3.4 Experimental protocol	8
4. Implementation	10
4.1 Path tracer	10
Per-bounce sequence	11
4.2 Principled BSDF	13
Material state and lobe structure	13
Evaluation, probability and sampling	14
Diffuse and GGX reflection	15
Clearcoat, sheen and transmission	16
4.3 Environment lighting	17
Radiance and importance representations	17
Direction mapping and probability	18
Compensation and infinite-light representation	19
4.4 Physical camera, AOVs and denoising	19
Thin-lens camera	20
First-hit AOVs	21
OIDN post-processing	21

4.5 Production-scene conversion pipeline	22
5. Evaluation and results	24
5.1 Controlled path-tracer validation	24
Furnace validation	24
Why the matched-seed results are identical	25
Multi-bounce integration	25
AOV outputs	26
5.2 BSDF validation	27
White-furnace behaviour	27
Sampling consistency	27
GTR1 clearcoat distribution	28
Smooth-glass equivalence	28
5.3 Sampling and noise experiments	29
Environment sampling	29
Sampling configuration	31
Firefly clamping	32
OIDN denoising	32
5.4 Isolated feature validation	33
Material capability	33
Physical camera	34
Focused BSDF behaviour	35
Transmission and visibility	36
5.5 Imported and production-scene validation	38
5.5.1 Blender-authored LEGO scene	38
5.5.2 Country Kitchen	39
5.6 Scale-dependent failure case study	43
6. Blender Cycles comparison and discussion	45
6.1 Comparison protocol	45
6.2 Comparison set	46
6.3 Evaluation	46
6.4 Critical discussion	50
7. Conclusion	52
References	53

Implementing and Validating Physically Based Rendering Techniques in an Extensible Renderer

Abstract

Physically based rendering research is commonly presented as equations and high-level algorithms, but integrating those ideas into a working renderer also requires consistent sampling methods and their associated probability densities, numerical safeguards and adherence to framework conventions. This thesis addresses that implementation gap through custom Mitsuba 3 plugins including a path tracer, HDR environment emitter, Disney-style Principled BSDF and physical thin-lens camera, supported by auxiliary image outputs and optional Intel Open Image Denoise processing.

Validation progresses from comparisons with Mitsuba reference components and statistical tests to isolated feature scenes, a Blender-authored LEGO model and the production-scale Country Kitchen interior. The completed pipeline is also compared with Blender Cycles using closely matched geometry, cameras, illumination and material parameters. Reference comparisons, furnace tests and distribution checks support the correctness of the tested integrator and material configurations. Multiple importance sampling, environment importance sampling, stratification and lobe-aware sampling reduce variance when suited to the transport distribution, while clamping and denoising improve presentation with acknowledged bias or reconstruction effects. Production-scale testing exposed an environment shadow-ray truncation defect missed by smaller scenes, demonstrating the complementary value of component and integration tests. The resulting images reproduce the principal physical and visual behaviour of the reference systems and are often close to Cycles; remaining differences are explained by material and light translation, filtering, colour handling and documented approximations. The work contributes a reproducible method for turning established rendering research into validated, extensible renderer components.

1. Introduction

1.1 Motivation

Physically based rendering creates images by simulating how light travels through a scene, interacts with surfaces, and reaches the camera. Since Kajiya (1986) introduced the rendering equation, research has established methods for Monte Carlo integration, material scattering and camera simulation (Pharr et al., 2023; Veach, 1997). Translating them into software nevertheless requires decisions often absent from the high-level descriptions: sampling procedures and probability densities must agree, numerical edge cases must be controlled, and components must follow the host renderer's conventions.

Rendering systems are therefore often layered rather than created entirely from first principles. Framestore's *Freak*, for example, is an in-house path tracer built on Intel Embree, which supplies optimised acceleration structures, traversal and primitive intersection so that application development can concentrate on light transport and production requirements (Framestore, n.d.; Wald et al., 2014). *Freak* was not initially developed as an entirely new renderer. During *Gravity* and the work that followed, Framestore progressively implemented its own shaders, lights, integrators and volumes around Arnold until Arnold was used primarily for geometry intersection, making its eventual replacement with Embree a practical next step (Failes, 2020). This project draws inspiration from that gradual process at a much smaller scale. Its purpose is not to optimise the underlying ray-tracing core, but to implement and test the surrounding transport, shading, lighting and camera components in order to understand how they interact as a complete rendering system. Mitsuba 3 has a broader role than Embree, additionally providing XML scene loading, textures, samplers, films, plugin interfaces and vectorised execution through Dr.Jit (Jakob et al., 2022a, 2022b). The analogy is therefore not one of architectural equivalence or project scale, but of progressively developing specialised rendering behaviour on trusted infrastructure.

This thesis evaluates whether established techniques can be integrated coherently within that framework and whether their outputs agree with theory, reference implementations and controlled external comparisons, rather than judging plausibility alone.

1.2 Aim and scope

The aim is to implement, integrate and validate established physically based rendering techniques within Mitsuba, then evaluate them in controlled and production-style scenes relative to Mitsuba reference components and Blender Cycles. The system includes a path tracer, HDR environment emitter, Disney-style Principled BSDF, physical thin-lens camera, auxiliary image outputs and optional OIDN processing. Evaluation advances from synthetic reference and statistical tests to a Blender-authored LEGO model, the Country Kitchen interior and a final Cycles comparison.

Success is assessed through coherent physically based behaviour and clearly explained differences rather than exact pixel agreement with Cycles. Exact agreement would be unrealistic because the systems differ in material models, lighting conventions, filters, execution and post-processing. The intended outcome is a validated research renderer rather than a production-system replacement.

1.3 Contributions

The principal contribution lies in implementation, integration and experimental validation:

- An implementation of established path-tracing, importance-sampling, physically based shading and camera techniques as custom Mitsuba plugins.
- An account of the design decisions required to integrate these techniques coherently within an existing renderer framework.
- A structured validation methodology progressing from reference and statistical tests to Blender-authored assets and a production-scale interior.
- A critical evaluation of correctness, variance, visual quality, practical trade-offs and known approximations.
- An end-to-end comparison with Blender Cycles using closely matched scenes.

1.4 Research questions

This investigation is organised around four research questions.

- **RQ1 / Implementation correctness:** How can established rendering techniques be implemented as extensible renderer components, and to what extent do their outputs agree with theoretical expectations and Mitsuba reference implementations?
- **RQ2 / Sampling and practical trade-offs:** How do sampling strategies, including multiple importance sampling, HDR environment importance sampling, stratification and BSDF-lobe sampling, and practical treatments such as clamping and denoising affect convergence, variance, bias and final image quality?
- **RQ3 / Production-scene behaviour:** What does progressive testing from synthetic scenes to Blender-authored assets and a production-scale interior reveal about component correctness, integration limitations and scale-dependent failures?
- **RQ4 / Blender Cycles comparison:** How closely can the custom pipeline reproduce the practical visual behaviour of Blender Cycles under controlled, closely matched scene conditions, and how can the remaining differences be explained?

1.5 Thesis structure

Chapter 2 presents the required theory and related systems. Chapter 3 defines the architecture and validation methodology; Chapter 4 explains the custom components; and Chapter 5 evaluates them from isolated tests through production integration, including a scale-dependent visibility failure. Chapter 6 compares the completed pipeline with Cycles. Chapter 7 answers the research questions, states the limitations and identifies future work.

2. Background and related work

2.1 Rendering equation and Monte Carlo path tracing

Physically based rendering describes image formation as radiance transport. Recursive ray tracing represented visibility, shadows, ideal reflection and refraction (Whitted, 1980); Kajiya's rendering equation unified direct and indirect illumination (Kajiya, 1986):

$$L_o = L_e + \int \Omega f_s \times L_i \times |n \cdot \omega_i| d\omega_i$$

Outgoing radiance is the sum of emitted radiance and incident radiance scattered towards the observer. The BSDF relates incident and outgoing directions, while the cosine accounts for projected surface area. Because incident radiance may itself arise from another interaction, the equation is recursive.

Path tracing estimates this generally intractable integral by constructing random camera paths. For samples drawn from probability density $p(X)$, Monte Carlo integration uses

$$\hat{I} = (1/N) \sum [f(X_i) \div p(X_i)]$$

The estimator is unbiased when the distribution covers the non-zero integrand and each contribution is divided by its correct density. Standard error falls as $1/\sqrt{N}$, so halving it generally requires four times as many samples. Path tracing can represent colour bleeding, glossy reflection and multi-bounce transmission, but finite-sample noise makes path generation, weighting and termination as important as the transport equation itself (Pharr et al., 2023; Veach, 1997).

2.2 Variance reduction and practical image reconstruction

Importance sampling reduces variance by directing samples towards directions expected to contribute more light. Next-event estimation explicitly samples a light source and tests its visibility, which is useful when emitters are unlikely to be reached through unconstrained path continuation. BSDF sampling instead selects directions according to the material response and is effective for concentrated glossy or specular reflection. Multiple importance sampling (MIS) combines these strategies and reduces the influence of samples produced by a poorly suited strategy. The power heuristic of Veach and Guibas (1995) is:

$$w_i = (n_i p_i)^2 \div \sum_j (n_j p_j)^2$$

where p_i is the probability density of a sampling strategy and n_i its sample count. Correct weighting requires the evaluated contributions and their probability densities to remain consistent (Pharr et al., 2023; Veach & Guibas, 1995).

HDR environment maps use a separate importance distribution. Uniform spherical sampling can waste samples when most illumination is concentrated in a few bright regions. For an equirectangular environment map, sampling importance can be expressed as:

$$P(x, y) \propto L(x, y) \times \sin(\theta)$$

where $L(x, y)$ is pixel luminance and $\sin(\theta)$ accounts for the solid angle represented by each image row. The resulting environment-light samples can then be combined with BSDF samples through MIS (Pharr et al., 2023).

Russian roulette probabilistically terminates paths with low throughput after a chosen depth. Surviving paths are reweighted by their continuation probability to preserve the expected result. Independent sampling allows arbitrary sample counts, while stratified sampling distributes samples more evenly across the sampling domain (Pharr et al., 2023; Veach, 1997).

Clamping and denoising address image quality rather than sampling. Clamping suppresses extreme contributions but introduces bias. OIDN reconstructs a cleaner image and can use beauty, albedo and shading-normal AOVs to preserve boundaries. However, denoised output cannot establish transport correctness, so raw and processed results are evaluated separately (Áfra, 2026).

2.3 Physically based shading

A BRDF describes same-hemisphere reflection, while a BSDF also includes transmission. Physically plausible models are normally non-negative, reciprocal and energy-conserving. Lambertian reflection is view-independent:

$$f_d = \rho \div \pi$$

The division by π keeps its hemispherical integral within the incident energy and provides a useful validation baseline (Pharr et al., 2023).

Microfacet models treat a rough surface as microscopic planar facets. Cook–Torrance reflection has the form

$$f_r = [D(h) \times F(\omega_i, h) \times G(\omega_i, \omega_o)] \div [4 \times \ln \cdot \omega_i \times \ln \cdot \omega_o]$$

(Cook & Torrance, 1982) D describes facet orientations, F the Fresnel-reflected fraction and G masking and shadowing. GGX produces long-tailed highlights; Smith terms model microfacet visibility; and Schlick's approximation estimates Fresnel response efficiently (Heitz, 2014; Schlick, 1994; Walter et al., 2007). Walter et al. (2007) extended the framework to transmission and stressed that sampling and PDFs must match.

Single-scattering microfacet models omit interreflection between facets and can therefore lose energy at high roughness without implying an integrator error. This distinction is central to the later furnace tests (Kulla & Conty, 2017).

Disney's Principled model offers artist-facing controls for varied production materials (Burley, 2012). It combines Burley diffuse, GGX-style specular reflection, metallic response, anisotropy, clearcoat, sheen and dielectric transmission. Metallic response colours specular reflection and suppresses conventional diffuse; anisotropy uses different roughnesses along the tangent axes; clearcoat adds a narrow secondary lobe; sheen contributes at grazing angles; and smooth transmission selects reflection or refraction by Fresnel probability. The model prioritises controllability over exact layered-material simulation: clearcoat and sheen are additive, the implemented smooth transmission omits rough glass and absorption, and single-scattering GGX loses energy as roughness increases (Burley, 2015). These limitations frame the later implementation and validation.

2.4 Rendering architectures and research frameworks

Mitsuba 3's research architecture is based on interface contracts. Each plugin type exposes operations appropriate to its role: integrators query the scene, BSDFs and emitters provide evaluation and sampling operations, sensors generate rays, and films receive image samples. Other parts of the system interact with these interfaces rather than depending on a particular implementation, supporting algorithmic variation within a consistent rendering pipeline (Jakob et al., 2022b).

Dr.Jit provides the execution model behind these interfaces. Rendering state is represented using vectorised arrays containing many rays or surface interactions. Operations on these arrays are recorded, specialised and just-in-time compiled into data-parallel CPU or GPU kernels (Jakob et al., 2022a). The `@dr.syntax` transformation converts compatible Python loops and conditionals into traced control flow, while masks represent rays that follow different paths or terminate at different times (Realistic Graphics Lab, n.d.).

This enables Python plugins to participate in vectorised rendering instead of invoking the Python interpreter separately for every ray. It does not accelerate arbitrary Python code: implementations must use Dr.Jit array types, masking and supported control-flow patterns. Mitsuba therefore combines the accessibility of Python-level algorithm development with compiled execution of the resulting numerical operations.

3. System design and methodology

3.1 Layered architecture

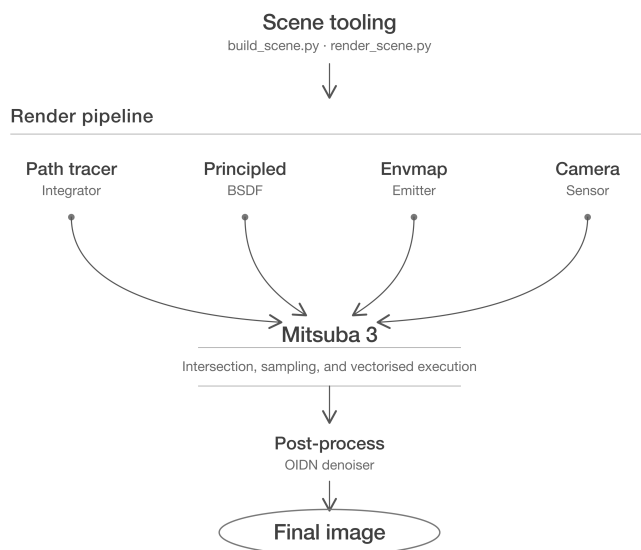
Building on the framework described in Section 2.4, the project is organised into four layers:

Layer	Responsibility
Mitsuba and Dr.Jit	Scene representation, ray intersection, plugin execution and vectorised computation.
Custom rendering plugins	Light transport, material scattering, environment lighting and camera simulation.
Scene and evaluation tooling	Scene conversion, scripted configuration, test execution and evidence generation.
Post-processing	Optional Intel Open Image Denoise processing using beauty, albedo and normal buffers

[Table 3.1.a (p. 7) — *Renderer Layers and Responsibilities*]

During rendering, the custom sensor generates camera rays, the integrator queries scene intersections and invokes the active BSDF and emitter plugins, and the film stores the resulting beauty image and auxiliary outputs. Scene and evaluation tools configure this process and collect its results but do not participate directly in light transport.

OIDN forms a separate final stage. It consumes the completed beauty, albedo and shading-normal buffers; depth is retained for diagnostics. Keeping this operation outside the estimator establishes a clear architectural boundary between rendered evidence and subsequent image reconstruction (Áfra, 2026).



[Figure 3.1.a (p. 7) — *Layered Architecture of the Rendering System*. Note: Showing the Mitsuba/Dr.Jit foundation, custom plugins, scene and evaluation tooling, and optional denoising stage.]

3.2 Design principles

Four principles guided the implementation. First, controlled substitution allows a custom plugin to be exchanged with a Mitsuba reference component while other scene conditions remain fixed. Second, optional extensions remain baseline-compatible: physical exposure, transparent shadows, clamping and AOV generation are disabled or neutral unless requested. Third, each material's `eval()`, `pdf()` and `sample()` operations must describe the same active lobes and normalised probabilities, preventing incorrect weighting within mixed BSDFs. Finally, scene conversion, camera insertion and diagnostic variants are scripted for reproducibility, while approximations such as thin transparent shadows, radiance clamping and cross-renderer material conversion are identified explicitly.

3.3 Validation ladder

Validation proceeds cumulatively from isolated numerical checks to authored production scenes.

Stage	Purpose and principal evidence
1. Reference equivalence	Compare custom components against matched Mitsuba implementations, including path tracing and dielectric transport
2. Statistical and physical checks	Test furnace behaviour, PDF normalisation, sampling distributions and convergence over independent seeds
3. Isolated feature validation	Exercise camera depth of field, bokeh, exposure, material lobes, transmission and visibility in controlled scenes
4. Imported-asset validation	Reconstruct a Blender-authored LEGO scene and compare the custom renderer with a Cycles reference
5. Production-scale integration	Combine the custom components in the Country Kitchen, containing approximately 85 converted materials, glazing, complex occlusion and multiple sensors

[Table 3.3.a (p. 8) — Five-Stage Validation Ladder]

These stages answer different questions. Numerical tests can expose biased sampling and incorrect weighting but lack the scale, occlusion and asset diversity of a complete interior; a plausible production render cannot independently prove estimator correctness. The Kitchen demonstrated this complementarity: an environment shadow-ray distance error passed smaller, open scenes but failed when distant room geometry had to occlude sampled environment directions. Direct-only and window-transmission controls isolated the defect.

3.4 Experimental protocol

Each evaluation section is reconstructed by an evaluator rather than using one universal render preset. Scene preparation and production-scene transformations are also scripted to avoid manually diverging XML variants. Each evaluator records the settings relevant to its claim, including resolution, sampler, samples per pixel, pass count, seeds, path depths, clamping, visibility options and post-processing state.

Quantitative measurements use linear floating-point EXR data; PNG files are display derivatives. Masks restrict object-specific measurements to valid pixels—for example, an eroded primary-hit region excludes furnace silhouettes and background. Noise and convergence are evaluated across independent seeds, separating between-seed variation from spatial variation. Independent sampling is used when stratification would alter an exact sample budget, while high-sample results may combine independently seeded passes to reduce peak memory without changing the total budget.

Raw, unclamped renders provide the primary transport evidence. Clamped and OIDN outputs are labelled and reported separately. Cross-renderer comparisons match geometry, camera, resolution and nominal sample budget where practicable. Residual differences are interpreted alongside the component tests, and render times are reported as contextual measurements rather than equivalent-performance benchmarks.

Canonical runs produce images, measurements and machine-readable metadata containing their configuration, software versions, Mitsuba variant, hardware, repository state and relevant file hashes. Quick runs use separate output routes, while exploratory and legacy results remain outside the canonical thesis dataset.

4. Implementation

4.1 Path tracer

The path tracer (`integrators/path_tracer.py`) defines the transport estimator while Mitsuba supplies scene handling, intersections, standard plugin interfaces and film reconstruction. This boundary avoids reimplementing renderer infrastructure and allows the custom integrator to be paired with either the custom materials and emitter or Mitsuba reference plugins (Jakob et al., 2022b).

Mitsuba provides	The custom integrator controls
Camera rays, film and reconstruction	Path-state initialisation and continuation
BVH (bounding volume hierarchy) intersection	When intersections and light samples are requested
BSDF and emitter <code>eval()/pdf()/sample()</code> interfaces	How their results are weighted and combined
Ray construction and offsetting	Termination and returned channels

[Table 4.1.a (p. 10) — Division of Responsibilities Between Mitsuba and the Custom Integrator]

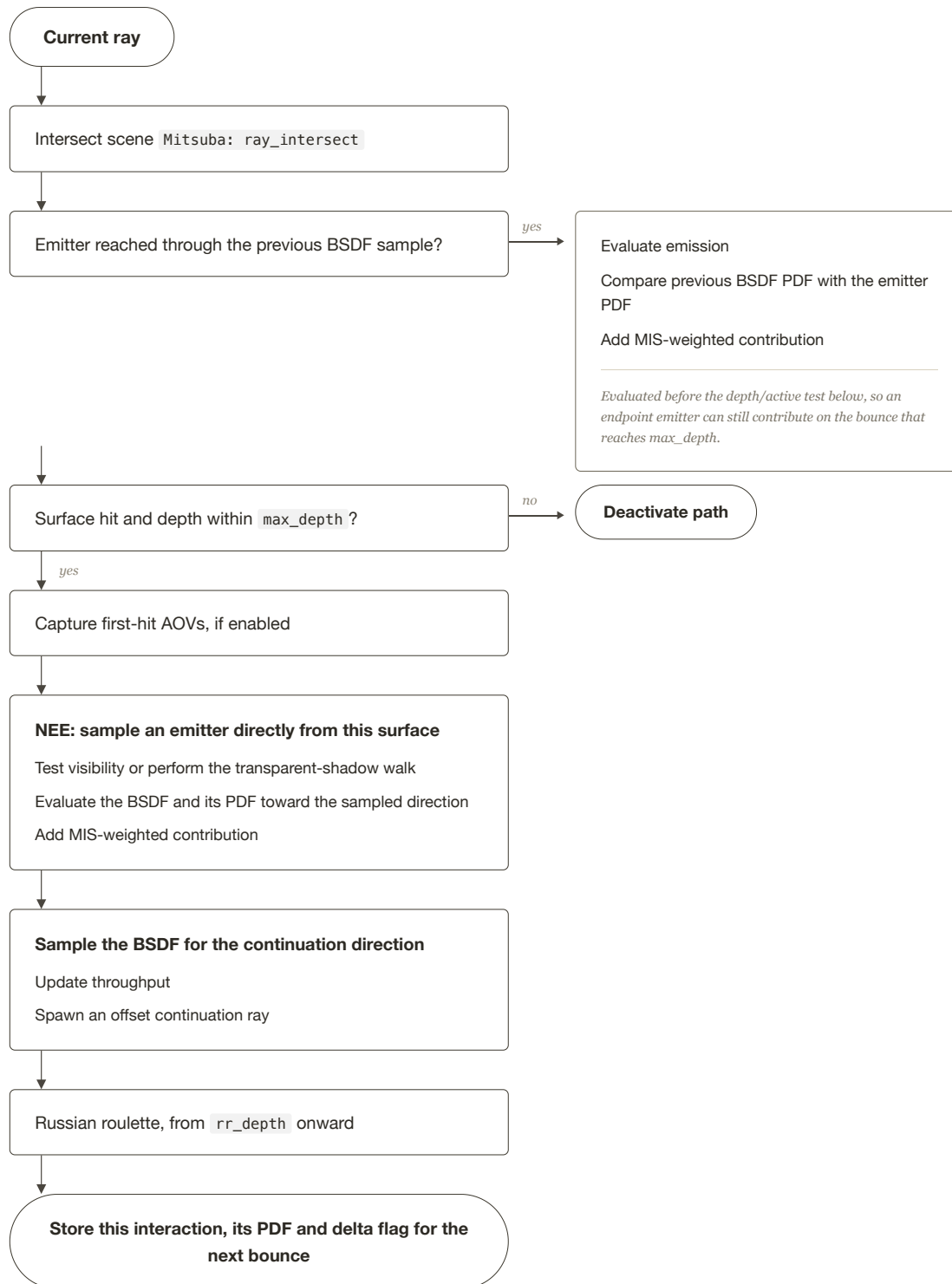
Vectorised execution. Dr.Jit traces a wavefront of active ray lanes rather than executing a separate Python path for each sample. The `@dr.syntax` transformation converts the `while active:` loop into vectorised operations, while masks prevent terminated lanes from contributing (Jakob et al., 2022a; Realistic Graphics Lab, n.d.).

State	Meaning
ray	Current path segment
throughput	Multiplicative weight accumulated along the path (β in Chapter 2's notation)
result	Radiance accumulated for the pixel sample so far (L in Chapter 2's notation)
depth	Current bounce count
active	Mask identifying paths still being traced
prev_si, prev_bsdf_pdf, prev_delta	The previous interaction, its BSDF-sampling PDF and whether that event was delta scattering, needed to weight an emitter reached during the current bounce

[Table 4.1.b (p. 10) — Path-Tracing State Variables]

The previous-interaction state is retained because emission reached by a BSDF-sampled continuation must be weighted against the probability of obtaining the same connection through direct-light sampling.

Per-bounce sequence



[Figure 4.1.a (p. 11) — Per-Bounce Sequence of the Custom Path Tracer]

Direct illumination has two complementary estimators: next-event estimation samples an emitter from the current surface, whereas the BSDF route evaluates emission after a continuation ray reaches a light or escapes to the environment. The two probabilities are combined with the power heuristic:

$$w_a = p_a^2 / (p_a^2 + p_b^2)$$

The NEE contribution has the form:

$$L_{\text{NEE}} = T \times f_s \times W_e \times w_e$$

where T is throughput, f_s is the BSDF value, W_e is Mitsuba's emitter-sampling weight and w_e is the MIS weight (Pharr et al., 2023; Veach & Guibas, 1995). Emission is evaluated before the depth check so that a path reaching a light at `max_depth` still records its endpoint contribution.

Delta events require special treatment. A light reached after ideal reflection or refraction receives weight one because emitter sampling cannot reproduce the discrete direction. Conversely, a delta emitter sample on the NEE route is not compared with a continuous BSDF proposal and also receives weight one.

For continuation, the BSDF weight multiplies throughput and Mitsuba spawns an offset ray. `max_depth` provides a deterministic bound. From `rr_depth`, Russian roulette derives a continuation probability from throughput; surviving paths are divided by that probability to preserve the expected radiance (Pharr et al., 2023; Veach, 1997).

Optional controls intervene locally in the baseline loop:

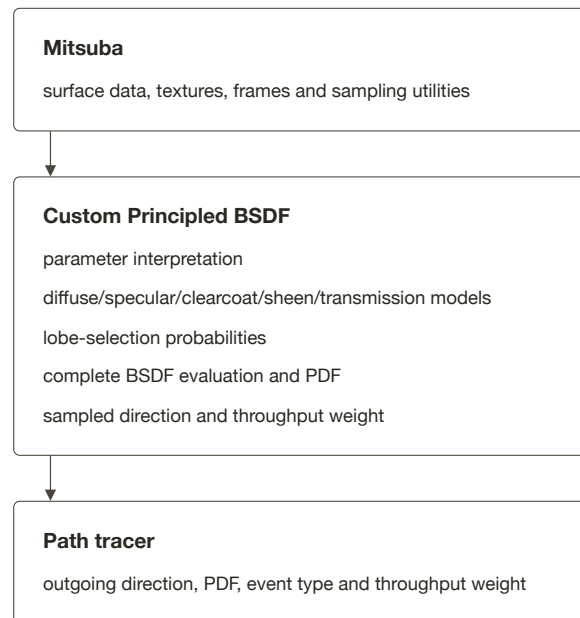
Feature	Intervenes at	Effect
<code>transparent_shadows</code>	NEE visibility test	Walks the shadow ray through delta-transmissive surfaces instead of using Mitsuba's binary occlusion result, up to a configurable pass-through budget
<code>hide_from_camera</code>	Emitter-hit evaluation, primary rays only	Suppresses a named shape's directly visible emission; its illumination through NEE and indirect bounces is unaffected
<code>firefly_clamp</code>	Emitter-hit and NEE accumulation	Caps individual contributions before they are added to result, deliberately trading bias for reduced outlier variance
<code>with_aovs</code>	After intersection, at depth zero	Records first-hit albedo, normal and depth without altering path continuation

[Table 4.1.c (p. 12) — Optional Path-Tracer Controls and Their Effects]

`opaque_shadow_shapes` modifies transparent shadows by forcing selected materials to remain opaque. Matching uses the BSDF ID rather than the shape ID because Mitsuba can merge compatible meshes while retaining their material references. All controls default to off or empty, leaving the baseline estimator unchanged. Transparent shadows and firefly clamping are deliberately approximate and are evaluated as such in Chapters 5 and 6.

4.2 Principled BSDF

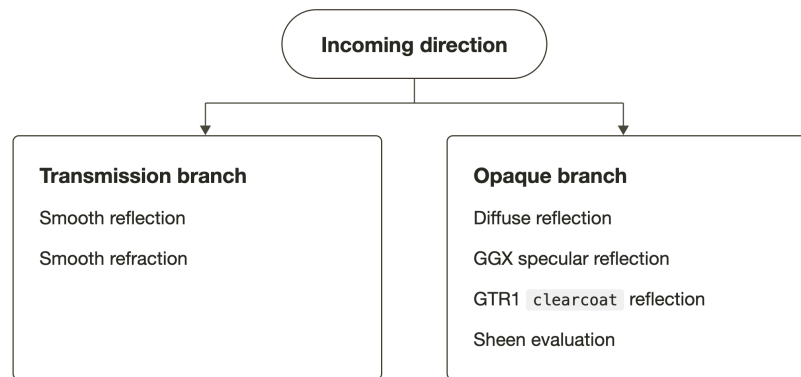
The integrator contains no material-specific branches: it only calls the BSDF interface. The custom plugin (`bsdfs/principled.py`) uses Mitsuba's interactions, texture evaluation, shading frames and sampling utilities to implement diffuse, dielectric, metallic, clearcoat, sheen, anisotropic and smooth-transmission behaviour behind that common interface.



[Figure 4.2.b (p. 13) — Principled BSDF Transmission and Opaque-Branch Structure]

Material state and lobe structure

Colour, roughness, metallic, specular, anisotropy, clearcoat, sheen and transmission parameters may be constants or textures and are evaluated at each interaction. IOR, thin-sheet behaviour and the Lambert/Burley diffuse choice are configuration values. Transmission first selects between a smooth dielectric event and the opaque mixture; the opaque branch contains diffuse, GGX specular and GTR1 clearcoat proposals, while sheen contributes to evaluation without a separate sampling branch (Burley, 2012).



[Figure 4.2.c (p. 14) — Tangent-Aligned Frame for Anisotropic GGX Sampling]

The first random variate chooses the top-level branch and is remapped to zero-to-one within the selected interval. Clearcoat receives an explicit share of the opaque branch, and the remainder is divided between specular and diffuse sampling:

$$p_{\text{clear}} + p_{\text{spec}} + p_{\text{diff}} = 1$$

The exact opaque-branch probabilities mirror the code:

```

cc_prob = clearcoat × clamp(
    0.05 + 0.20 × clearcoat_gloss,
    0.05,
    0.25
)
p_clear = cc_prob
p_spec  = spec_prob × (1 - cc_prob)
p_diff  = (1 - spec_prob) × (1 - cc_prob)

```

For non-metals, `spec_prob` is:

$$\text{clamp}(0.8 \times \text{metallic} + 0.1, 0.1, 0.9)$$

It becomes one for effectively metallic materials and zero only when both metallic and specular response are absent. Thus any active specular lobe remains sampleable without preventing a fully diffuse configuration.

Evaluation, probability and sampling

Mitsuba exposes three related BSDF operations:

```

eval(si, wo) → scattering contribution in direction wo
pdf(si, wo)  → probability of generating direction wo
sample(...)  → selected direction, PDF, event flags and weight

```

During NEE, `eval()` and `pdf()` assess an emitter-supplied direction; `sample()` instead generates the continuation (Jakob et al., 2022b). For a continuous event, it returns:

$$\text{sample weight} = \text{complete BSDF value} / \text{complete mixture PDF}$$

The complete mixture is required even when a particular lobe proposed the direction, because other lobes can also contribute value and probability there:

$$\begin{aligned} p_{\text{opaque}}(w_o) = & (1 - \text{transmission}) \times \\ & [\\ & \quad p_{\text{clear}} \times p_{\text{cc}}(w_o) \\ & + p_{\text{spec}} \times p_{\text{GGX}}(w_o) \\ & + p_{\text{diff}} \times p_{\text{diffuse}}(w_o) \\ &] \end{aligned}$$

Equivalent mixture expressions exist independently in `sample()` and public `pdf()`. Their agreement is therefore established through matched implementation and the statistical tests in Chapter 5, rather than guaranteed by one shared helper. Dr.Jit evaluates candidate lobes vectorially and uses masks to select the event for each lane.

Diffuse and GGX reflection

The diffuse proposal uses cosine-weighted hemisphere sampling. Lambert mode evaluates:

$$\text{base_colour} / \pi$$

Burley diffuse modifies this value with incident, outgoing and half-vector grazing terms, beginning with:

$$\text{FD90} = 0.5 + 2 \times \text{roughness} \times \cos^2(\theta_h)$$

Diffuse response fades with metallic and is gated by dielectric Fresnel reflection so that interface reflection is not also assigned to the diffuse term (Burley, 2012). A separate query returns unlit `base_colour` for the albedo AOV.

The principal specular lobe uses Cook–Torrance GGX (Cook & Torrance, 1982; Walter et al., 2007) with Smith masking-shadowing (Heitz, 2014). Mitsuba supplies microfacet evaluation and sampling; the plugin derives directional roughness and Fresnel colour. Perceptual roughness becomes:

$$\alpha = \max(\text{roughness}^2, 0.0001)$$

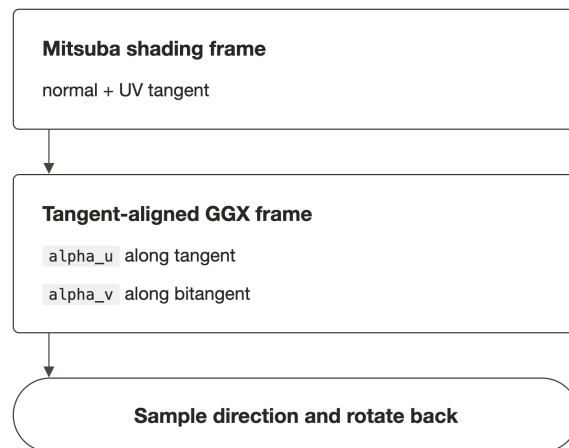
Anisotropy splits this value along the tangent axes:

$$\text{aspect} = \sqrt{1 - 0.9 \times \text{anisotropic}}$$

$$\alpha_u = \max(\alpha / \text{aspect}, 0.0001)$$

$$\alpha_v = \max(\alpha \times \text{aspect}, 0.0001)$$

The UV derivative defines the tangent-aligned GGX frame; a degenerate derivative falls back to the local x-axis. Directions are transformed into this frame for evaluation and sampling, then returned to Mitsuba's shading frame.



[Figure 4.2.c (p. 16) — Transformation From the Mitsuba Shading Frame to a Tangent-Aligned GGX Frame]

Fresnel reflection uses Schlick's approximation (Schlick, 1994):

$$F = F_0 + (1 - F_0) \times (1 - \cos(\theta))^5$$

For dielectrics:

$$F_0 = 0.08 \times \text{specular}$$

Metallic blends F_0 towards 'base_colour' while suppressing diffuse reflection, allowing the same GGX implementation to represent dielectric and metallic surfaces.

Clearcoat, sheen and transmission

Clearcoat is an additive Disney GTR1 lobe. Because Mitsuba's standard microfacet utility does not implement GTR1, the plugin provides distribution evaluation, inverse-CDF sampling and the corresponding outgoing-direction PDF. Its roughness is:

```

alpha_cc = max(
    0.1 * (1 - gloss) + 0.001 * gloss,
    0.0001
)
  
```

This maps the control from a broader coat to a narrow highlight. A dedicated clearcoat proposal samples the concentrated lobe. Sheen is an additive, tinted grazing response included in every complete opaque evaluation, but directions are proposed by diffuse, GGX or clearcoat sampling.

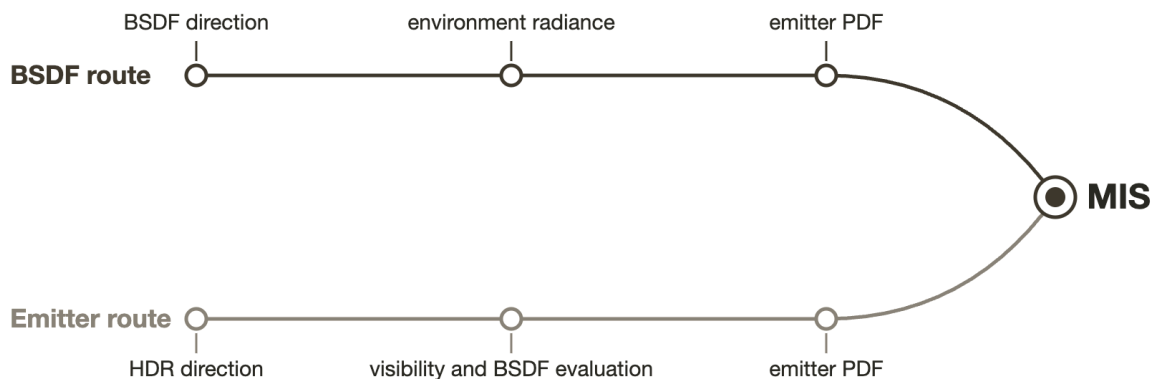
Transmission is a discrete smooth-dielectric branch. Fresnel selects ideal reflection or refraction; Mitsuba supplies the refracted direction and relative IOR, while total internal reflection becomes a reflective event. Transmission includes the radiance-mode squared-IOR correction and a `base_colour` tint. For `thin=True`, the direction passes straight through a zero-thickness sheet, the net relative IOR is one, and the two-interface reflectance is approximated by (Burley, 2015):

$$F_{\text{thin}} = 2F / (1 + F)$$

Ideal reflection and refraction have zero ordinary solid-angle `eval()` and `pdf()` values. `sample()` instead returns their discrete probability, throughput and `DeltaReflection` or `DeltaTransmission` flag. The flags let MIS and transparent-shadow handling distinguish delta transport without introducing knowledge of the material inside the integrator.

4.3 Environment lighting

The custom emitter (`emitters/envmap.py`) represents distant illumination with a latitude-longitude HDR image (Pharr et al., 2023). It evaluates radiance when a camera or BSDF ray escapes and proposes directions for NEE through Mitsuba’s emitter interface.

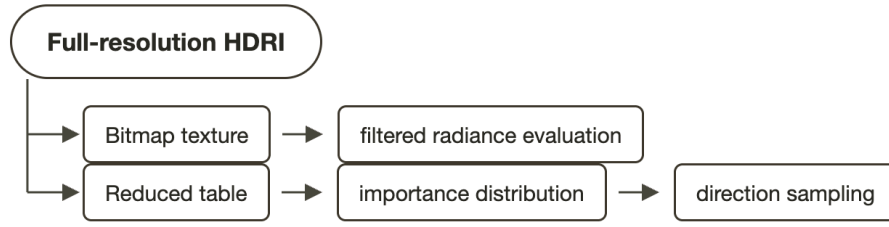


[Figure 4.3.a (p. 17) — Complementary Environment-Lighting Routes Combined by Multiple Importance Sampling]

Correct MIS requires `_radiance()`, `sample_direction()` and `pdf_direction()` to describe the same directional distribution. Mitsuba supplies image loading, transforms, direction records and `DiscreteDistribution2D`; the plugin supplies preprocessing, spherical mapping and solid-angle conversion.

Radiance and importance representations

The HDRI is loaded twice. A full-resolution, bilinearly filtered Mitsuba texture provides linear radiance, while a reduced table, 512×256 by default, defines the sampling distribution. Reducing the table lowers preprocessing and storage cost without reducing evaluated image resolution.



[Figure 4.3.b (p. 18) — Separate HDRI Representations for Radiance Evaluation and Importance Sampling]

Mean or maximum pooling constructs the reduced table. Maximum pooling is the default because it preserves small bright sources that averaging can dilute. Rec. 709 luminance is multiplied by the latitude–longitude solid-angle factor:

$$\text{importance}(u,v) = \text{luminance}(u,v) \times \sin(\theta)$$

This avoids over-weighting polar texels. Mitsuba normalises and samples the resulting distribution. Disabling importance sampling retains the same radiance evaluation but samples directions uniformly over the sphere.

Direction mapping and probability

The sampled cell probability and intra-cell remainder reconstruct continuous texture coordinates:

$$u = (\text{cell_x} + \text{remainder_x}) / W$$

$$v = (\text{cell_y} + \text{remainder_y}) / H$$

They map to spherical angles as:

$$\theta = \pi v$$

$$\phi = 2\pi u$$

The local direction is transformed by `to_world`; the inverse mapping lets radiance and PDF evaluation recover the same HDR location from a world-space direction. Rotation therefore changes lighting orientation without rebuilding the distribution.

The table supplies probability mass per cell, but MIS requires density per unit solid angle. Sampling and PDF evaluation both use:

$$p_\omega(d) = p_{\text{cell}} \times (W \times H) / [2\pi^2 \sin(\theta)]$$

The first factor converts cell mass to UV-area density and the latitude–longitude Jacobian converts it to solid angle (Pharr et al., 2023). Clamping `sin(θ)` to 0.0001 stabilises the poles. Identical conversion in both functions keeps emitter- and BSDF-sampled MIS probabilities consistent.

Radiance uses the full-resolution texture multiplied by a global scale, which changes intensity without changing relative importance. Direct samples return:

$$\text{emitter weight} = \text{HDR radiance} / \text{emitter PDF}$$

The integrator combines this weight with throughput, BSDF response and MIS.

Compensation and infinite-light representation

Optional compensation subtracts the mean importance to concentrate samples in brighter regions:

compensated importance =

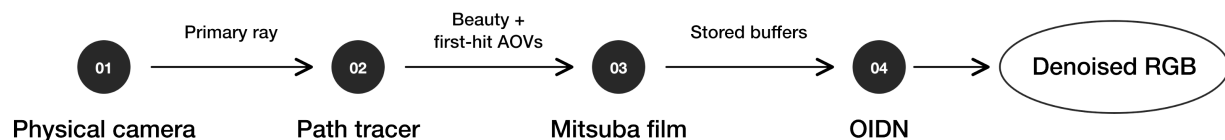
$$\max(\text{importance} - \text{mean}(\text{importance}), \epsilon)$$

A positive floor preserves support, evaluated radiance is unchanged, and the option is enabled by default. This is a simplified heuristic inspired by Karlik et al. (2019), applied after luminance has been solid-angle weighted. It is neither the paper's derived optimum nor its normal-independent, balance-heuristic configuration, and is therefore evaluated only as an empirical proposal modification.

Although the environment is infinite, Mitsuba's `DirectionSample3f` represents a finite position and distance. The plugin initially places a virtual endpoint using a radius of 1,000,000 and a shadow distance twice that value; `set_scene()` can only enlarge the radius from the scene bounds. The non-decreasing safe default is essential because `set_scene()` was not invoked for this Python emitter in the tested Mitsuba configuration. This scale-dependent design decision is validated by the Kitchen failure case in §5.6.

4.4 Physical camera, AOVs and denoising

The final image stage connects three components: `cameras/physical_camera.py` generates thin-lens primary rays, the integrator records first-hit AOVs, and `denoisers/oidn.py` processes completed buffers. Only the first two participate in sample generation and film accumulation; OIDN is a separate reconstruction step and does not affect transport or MIS.



[Figure 4.4.a (p. 19) — Rendering and Denoising Data Flow]

Thin-lens camera

`PhysicalCamera` implements Mitsuba's sensor interface while explicitly constructing projection and lens geometry. For film coordinates (u,v) , it first obtains the horizontally defined pinhole direction:

$$\begin{aligned}x &= -(2u - 1) \times \tan(\text{fov}/2) \\y &= -(2v - 1) \times \tan(\text{fov}/2) / \text{aspect} \\d &= \text{normalize}(x, y, 1)\end{aligned}$$

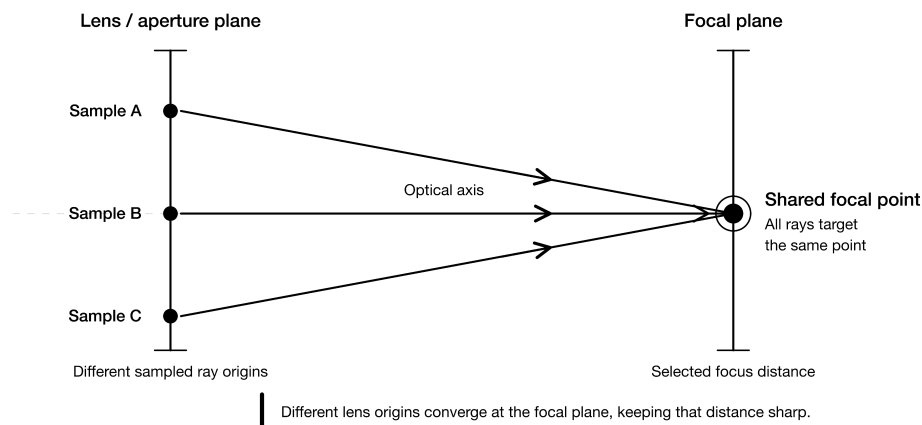
Dividing the vertical coordinate by film aspect preserves framing for non-square images (Pharr et al., 2023). The ray intersects a focal plane perpendicular to the forward axis:

$$\begin{aligned}t_{\text{focus}} &= \text{focus_distance} / d.z \\ \text{focus_point} &= d \times t_{\text{focus}}\end{aligned}$$

A second sample selects the lens origin, and the final ray targets the shared focal point:

$$\begin{aligned}\text{lens_origin} &= \text{aperture_sample} \times \text{aperture_radius} \\ \text{ray_direction} &= \text{normalize}(\text{focus_point} - \text{lens_origin})\end{aligned}$$

Thin-lens focusing for one pixel



[Figure 4.4.b (pp. 20) — Thin-Lens Ray Construction for a Single Pixel]

Different lens samples therefore agree at the focal plane but displace objects in front of or behind it, producing a circle of confusion after accumulation. Origins transform to world space as points and directions as vectors. Setting `aperture_radius=0` recovers pinhole projection. The sensor returns unit weight, so aperture size controls depth of field without changing exposure.

`_sample_aperture()` uses concentric disk sampling for fewer than three blades and equal-area triangular wedges for polygonal apertures (Pharr et al., 2023). The first coordinate selects a wedge:

$$\text{scaled sample} = \text{sample.x} \times \text{blade count}$$

Its fractional remainder is reused for uniform square-root barycentric sampling within that wedge. `aperture_rotation` rotates the polygon and hence the bokeh orientation.

This is a geometric thin-lens rather than a complete photographic camera. It excludes aperture-dependent exposure, compound-lens distortion, vignetting, diffraction and analytic ray differentials. The restricted model allowed projection, orientation, aspect ratio, focus and aperture sampling to be validated directly.

First-hit AOVs

With `with_aovs` enabled, zero-initialised buffers are written only at:

$$\text{first_hit} = \text{active AND depth} == 0$$

The integrator stores:

- albedo from `bsdf.eval_diffuse_reflectance()`;
- the world-space shading normal `si.sh_frame.n`;
- primary-ray distance `si.t`.

Later bounces cannot overwrite camera-visible features, and misses remain zero even when environment radiance contributes to beauty. The Principled BSDF provides unlit `base_colour` through the common albedo query, avoiding material-specific access in the integrator.

The linear EXR stores beauty RGB, albedo RGB, normal XYZ and depth in ten floating-point channels. Normals retain signed world-space values and depth remains unbounded scene distance; remapping occurs only for display. Because AOVs use the same stochastic lens rays as beauty, their defocused boundaries remain aligned.

OIDN post-processing

The wrapper passes contiguous float32 colour, albedo and normal buffers to OIDN's RT filter; depth remains diagnostic and is not supplied. It uses `hdr=True` for unbounded linear radiance, keeps signed world-space normals unchanged and correctly treats sampled auxiliary buffers as noisy. Three-channel input falls back to colour-only processing (Áfra, 2026).

OIDN-bound renders use a box reconstruction filter. Keeping each Monte Carlo sample within its assigned pixel avoids weighted splatting of one noisy path across neighbours and produced more suitable inputs for denoising.

`denoise_composite()` restores bright pixels after filtering using:

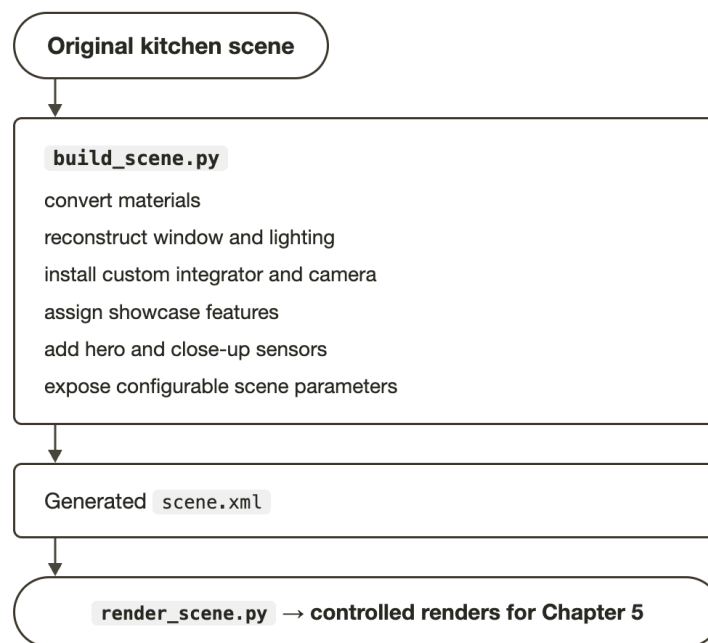
$$(R + G + B) / 3 > \text{emitter_threshold}$$

The default threshold is 2.0. Pixels passing this test are copied from the original render into the denoised result. `denoise_composite_aov_render()` combines automatic channel extraction with this restoration procedure.

Brightness is only a proxy for emitter identity: the procedure can restore bright reflections or outliers and miss dim emitters. A dedicated primary-hit emission AOV would isolate emitted radiance more reliably. Chapter 5 therefore evaluates the heuristic explicitly and Chapter 6 retains its scene dependence as a limitation.

4.5 Production-scene conversion pipeline

The Kitchen scene integrates all custom plugins at production scale. `build_scene.py` regenerates `scene.xml` from untouched `scene_original.xml` through named, ordered transformations, making every conversion and approximation repeatable rather than embedding them in a manually edited final scene.



[Figure 4.5.a (p. 22) — Deterministic Country Kitchen Scene-Conversion Pipeline]

Material conversion. Mitsuba diffuse, plastic, conductor and dielectric materials become parameterisations of `principled_bsdf`. Colours and bitmap textures are preserved where possible; microfacet alpha is converted to perceptual roughness, conductor eta and k to normal-incidence reflectance, and dielectric IOR and thin state are retained. Unsupported bump wrappers are flattened and alpha masking becomes solid because the custom BSDF has no cut-out opacity. These are documented, comparable inputs rather than claims of exact model equivalence.

Lighting and camera reconstruction. The source's emitting window rectangle is converted into smooth thin glazing, allowing the environment to illuminate the room through actual transmission. Practical rectangles and the pendant bulb are reconstructed from Blender. The hero sensor uses the Blender-exported transform and converted field of view, then replaces the Mitsuba sensor with `physical_camera`.

Feature placement assigns extensions selectively to plausible existing objects:

Feature	Production-scene use
Clearcoat	Kettle, cooker, lamp and radio plastics
Sheen	Cushion, towel and table mats
Anisotropy	Steel pot and knife blade
Thin transmission	Window glazing

[Table 4.5.a (p. 23) — Placement of Custom Material Features in the Country Kitchen Scene]

This tests whether isolated behaviours remain usable with textures, mesh tangents, complex lighting and interacting materials. The generated scene includes the hero view and five close-ups of the stove, table, radio, towel and island, exposing subtle material behaviour without removing it from the production context.

`render_scene.py` registers the plugins and exposes the controlled render parameters, including sensor, resolution, samples, seeds, pass accumulation, path depth, AOVs, environment intensity, visibility and camera settings. It preserves multi-channel EXR output and supports independently seeded pass averaging, providing one repeatable route for the Chapter 5 integration evidence.

5. Evaluation and results

5.1 Controlled path-tracer validation

The path tracer was validated before the custom BSDF so that transport and material errors could be separated. The comparisons retain Mitsuba geometry, materials, camera, sampler, filter and path-depth settings; only the integrator changes. Furnace statistics use a sphere mask eroded by two pixels at 128×128 to exclude partially covered silhouettes.

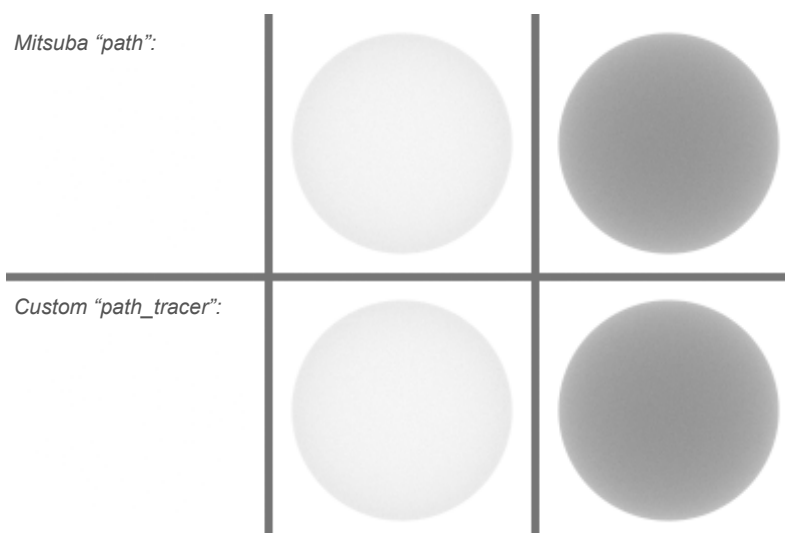
Furnace validation

A diffuse white furnace tests preservation of unit radiance, while Mitsuba's directional roughconductor provides a GGX reference.

Material	Setting	Mitsuba path	Custom path_tracer
Diffuse, 256 spp	Mean	0.999950	0.999950
Diffuse, 1024 spp	Mean	1.000044	1.000044
GGX	Roughness 0.0	1.000001	1.000001
GGX	Roughness 0.5	0.881552	0.881552
GGX	Roughness 1.0	0.366848	0.366848

[Table 5.1.a (p. 24) — Eight-Seed Mean Furnace Radiance for the Reference and Custom Integrators]

Diffuse radiance remains at unity, and its mean spatial standard deviation falls from 0.01582 at 256 spp to 0.00795 at 1024 spp, matching inverse-square-root convergence. The identical GGX means show that the roughness-dependent loss belongs to the shared single-scattering model, not the custom integrator.



[Figure 5.1.a (p. 24) — Matched-Seed Furnace Comparison Between the Reference and Custom Integrators. Note: Mitsuba reference above, custom path tracer below.]

Why the matched-seed results are identical

With the same seed, camera rays and components, both integrators consume random numbers in the same order in these simple furnace paths. The diffuse renders are therefore pixel-identical, while the roughness-0.5 GGX masked MAE is only 5.6×10^{-9} . This correlation is strong equivalence evidence but not an independent convergence test. A separate control used different seeds and recorded Mitsuba's PathIntegrator and the custom Python PathTracer, also ruling out accidental duplicate plugin loading.

Material	Reference/custom seeds	Mean difference	Masked MAE
Diffuse, 256 spp	1000 / 0	0.000276	0.01775
Diffuse, 1024 spp	1000 / 0	0.000072	0.00886
GGX, roughness 0.5	43 / 56	0.000307	0.02586
GGX, roughness 1.0	43 / 56	0.000479	0.01975

[Table 5.1.b (p. 25) — Independent-Seed Path-Tracer Control Results]

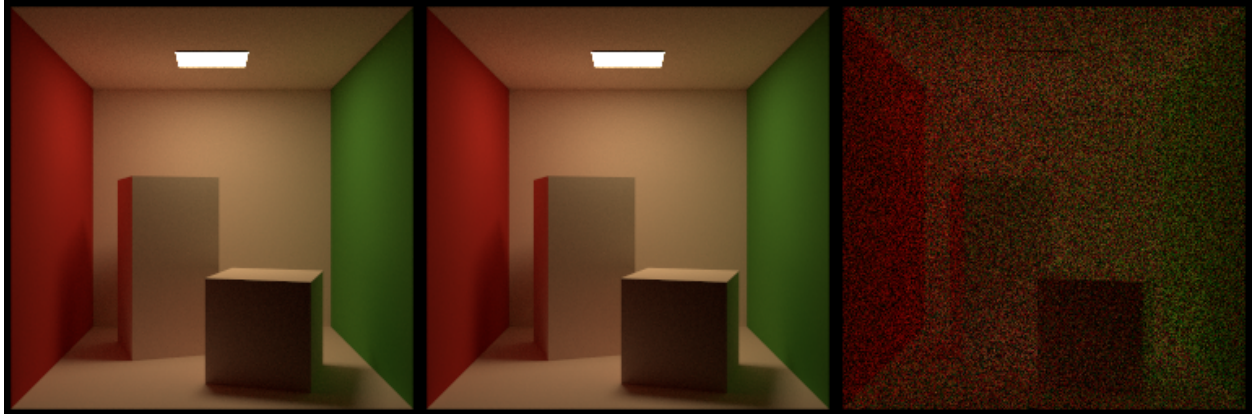
Nonzero MAE confirms independent stochastic samples, but the means and spatial deviations remain closely matched; the diffuse cross-seed MAE also approximately halves from 256 to 1024 spp.



[Figure 5.1.b (p. 25) — Independent-Seed GGX Furnace Comparison. Note: Reference seed 43, custom seed 56 and their absolute difference amplified by 10.]

Multi-bounce integration

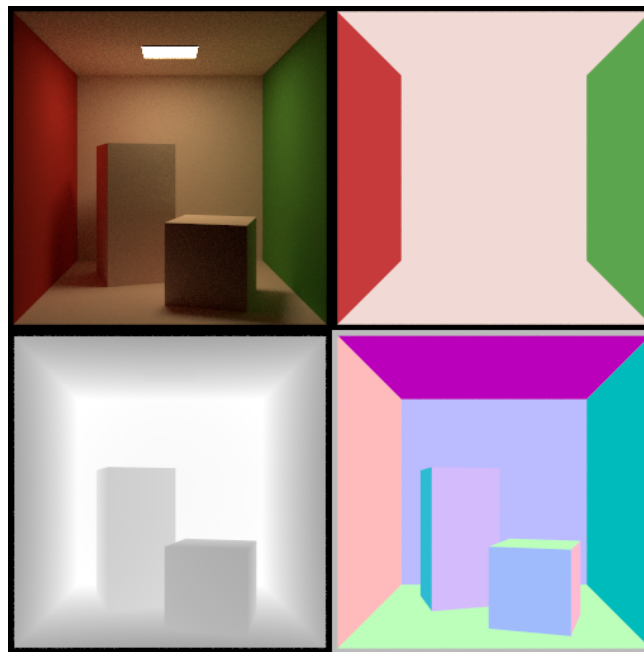
The Cornell box adds indirect illumination, occlusion, colour bleeding and multiple bounces. Eight 128-spp passes gave 1024 effective spp per integrator. Mean radiance was 0.147063 for Mitsuba and 0.147058 for the custom path tracer (-0.0035%), with MAE 0.000856 and RMSE 0.001622.



[Figure 5.1.c (p. 26) — Cornell-Box Comparison of the Reference and Custom Path Tracers. Mitsuba reference, custom path tracer, and absolute difference amplified $\times 20$.]

The matched-seed, independent-seed and Cornell results establish reference-level transport behaviour while confirming that the plugins are distinct implementations.

AOV outputs



[Figure 5.1.d (p. 26) — Cornell-Box Beauty and Arbitrary Output Variables]

Cornell-box AOV output. Beauty and albedo are shown on the top row; primary-hit depth and world-space shading normals are shown below. The albedo and normal passes provide auxiliary surface information for denoising, while depth supports geometric inspection.

The Cornell-box control also verified primary-hit albedo, shading-normal and depth channels without changing the default rendering path. Albedo and normals guide OIDN in separating surface detail from noise; depth supports geometry and distance diagnostics.

5.2 BSDF validation

With transport validated independently, the Principled BSDF was tested for furnace behaviour, sampling/PDF consistency and agreement with Mitsuba smooth dielectric transmission. Visual parameter behaviour follows in § 5.4.

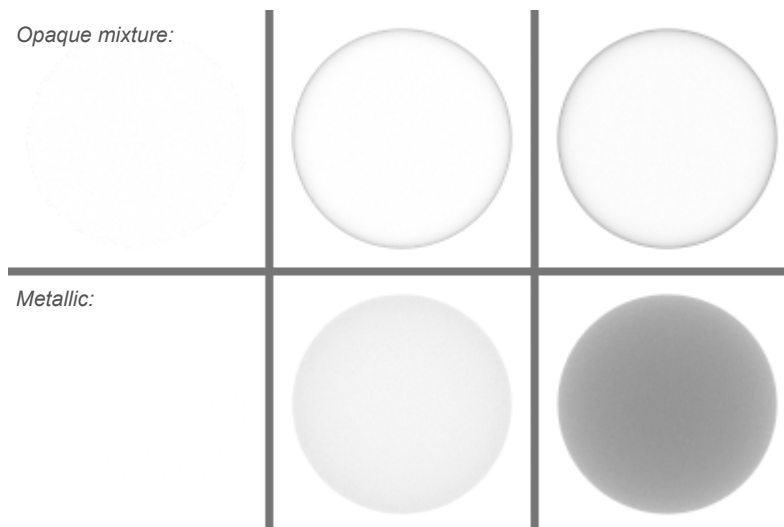
White-furnace behaviour

An opaque material was placed in a uniform white environment. Table 5.2.a reports eight-seed means from 256-spp renders; uncertainty is the between-seed standard deviation.

Material configuration	Roughness 0.0	Roughness 0.5	Roughness 1.0
Lambert control	0.99995 ± 0.00013	0.99995 ± 0.00013	0.99995 ± 0.00013
Default opaque mixture	0.99990 ± 0.00012	0.98688 ± 0.00014	0.95611 ± 0.00015
Metallic	1.00000 ± 0.00000	0.88155 ± 0.00025	0.36685 ± 0.00024

[Table 5.2.a (p. 27) — Mean White-Furnace Radiance for Representative Principled BSDF Configurations]

Lambert remains at unity for all roughness values. Loss in the other configurations therefore originates in the rough microfacet term. The metallic values reproduce § 5.1's Mitsuba roughconductor results, identifying the high-roughness loss as a limitation of single-scattering GGX rather than disagreement between the custom evaluator and sampler.



[Figure 5.2.a (p. 27) — White-Furnace Response of Default Opaque and Metallic Principled Materials. Note: Top: default opaque mixture. Bottom: metallic. Columns: roughness 0.0, 0.5 and 1.0.]

Sampling consistency

Chi-squared tests compared directions from `sample()` with the distribution predicted independently by `pdf()`, testing whether lobe selection and the complete mixture PDF describe the same model.

Configuration	p-value	Result
Diffuse	0.934	Pass
Plastic	0.893	Pass
Metallic	0.171	Pass
Mixed dielectric/metallic	0.828	Pass
Broad clearcoat	0.848	Pass
Clearcoat with all three lobes	0.909	Pass
Clearcoat over metallic base	0.572	Pass

[Table 5.2.b (p. 28) — Chi-Squared Sampling-Consistency Results at $\alpha = .01$]

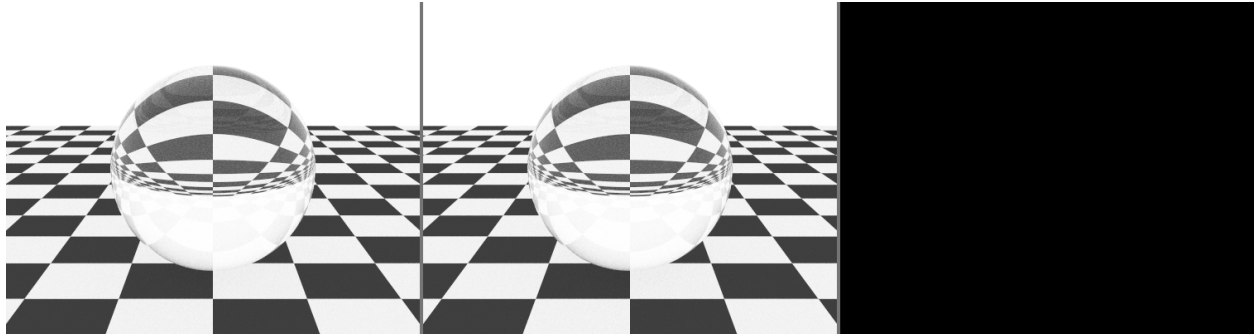
All representative diffuse, specular and clearcoat mixtures passed, including cases that exercise three-lobe probabilities. Extremely sharp distributions reached the test grid’s resolution limit: Mitsuba’s own sharp Principled case also failed at 201 cells, while clearcoat gloss 0.95 changed from failure at 201 to $p = 0.0956$ at 401 with unchanged implementation and samples. Such cases are therefore inconclusive when the lobe is narrower than the numerical grid.

GTR1 clearcoat distribution

No directly replaceable Mitsuba GTR1 clearcoat exists, so the distribution was checked independently. Analytic density integration returned unity for widths from 0.1 to 0.001, with maximum error below 1.6×10^{-11} ; sampled histograms agreed with integrated bin probabilities. A deliberately mis-scaled control was rejected. Thus both density normalisation and the sampling transformation were validated.

Smooth-glass equivalence

Smooth transmission was compared with Mitsuba dielectric at IOR 1.5 using four independently seeded sets of 200,000 samples across normal, grazing and critical-angle cases. Reflection frequencies, radiance-mode transmission weights and total internal reflection agreed. Both implementations returned zero for ordinary `eval()` and `pdf()` queries on discrete events, following Mitsuba’s delta-BSDF convention (Jakob et al., 2022b). Their furnace mean was 0.999992 ± 0.000027 , and the checkerboard transmission renders were pixel-identical, with zero MAE, RMSE and maximum difference.



[Figure 5.2.b (p. 29) — Smooth-Glass Equivalence With Mitsuba Dielectric. Note: Left: Mitsuba dielectric. Centre: custom smooth transmission. Right: absolute difference amplified by 50.]

These tests establish consistency of the principal diffuse, microfacet, mixture, clearcoat and smooth-transmission mechanisms while separating genuine single-scattering energy loss from implementation error.

5.3 Sampling and noise experiments

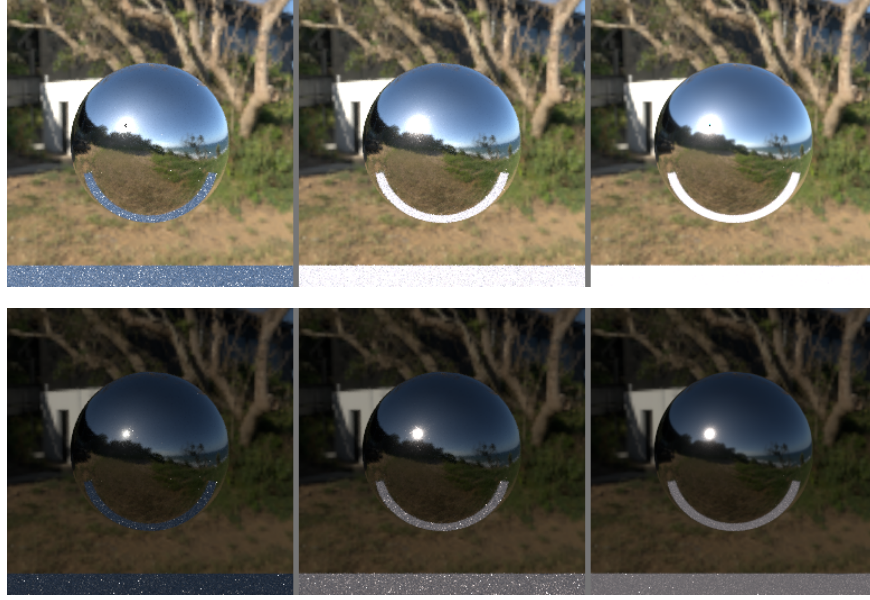
This section distinguishes variance-reduction methods that preserve the expected result from clamping and denoising, which suppress noise through bias or reconstruction.

Environment sampling

Uniform-sphere and luminance importance sampling were compared with 24 matched seeds at 16 spp. The brightest 1% of Venice Sunset and Sundowner Overlook contains 9.1% and 73.5% of HDRI energy respectively. Noise is the across-seed mean per-pixel standard deviation within three-pixel-eroded geometry masks at 256×256.

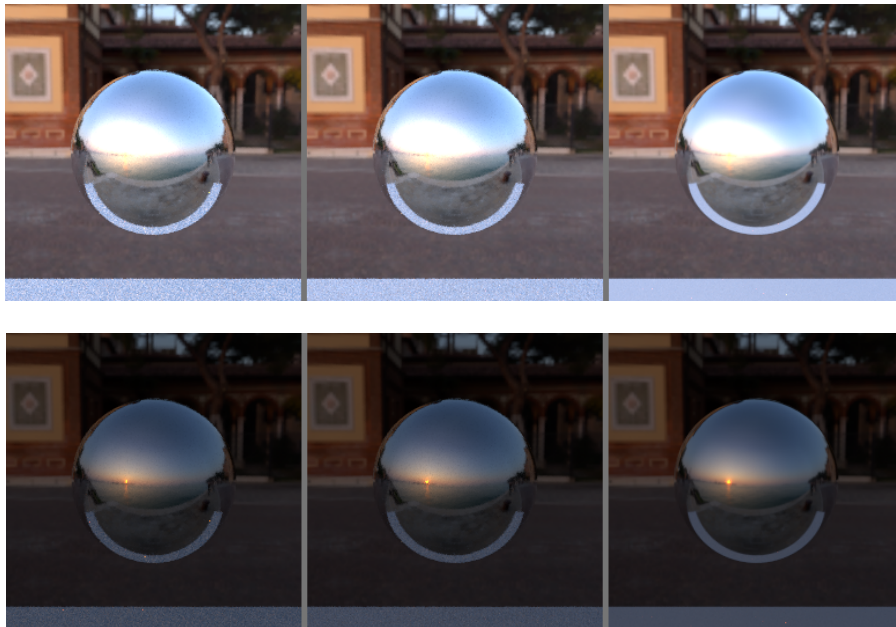
HDRI	Region	Uniform SD	Importance SD	Reduction
Venice	Sphere	0.05329	0.04046	24.1%
Venice	Floor	0.09776	0.07148	26.9%
Sundowner	Sphere	2.08976	0.47294	77.4%
Sundowner	Floor	4.55619	0.75335	83.5%

[Table 5.3.a (p. 29) — Environment-Sampling Noise Under Uniform and Importance Sampling. Note: The experiments use Studio Kontrast 04 (Wronkowski, n.d.), Venice Sunset (Zaal, n.d.-b), Sundowner Overlook (Barresi, n.d.) and Rogland Clear Night (Zaal, n.d.-a), obtained from Poly Haven under its CC0 licence.]



[Figure 5.3.a (pp. 30) — Effect of Environment Importance Sampling in Sundowner Overlook. Note: uniform sampling, importance sampling, and the 24-seed importance-sampled mean (left to right). The bottom row repeats the images three stops darker to reveal bright outliers and residual noise.]

The benefit grows with illumination concentration. Directly visible backgrounds are unchanged because they do not use direction sampling. Venice's mean difference was -0.0023% ; Sundowner's wider difference interval still included zero because uniform sampling occasionally produced extreme contributions. The experiment therefore demonstrates lower variance without evidence of systematic bias.



[Figure 5.3.b (p. 30) — Effect of Environment Importance Sampling in Venice Sunset. Note: uniform sampling, importance sampling, and the 24-seed importance-sampled mean (left to right).]

Sampling configuration

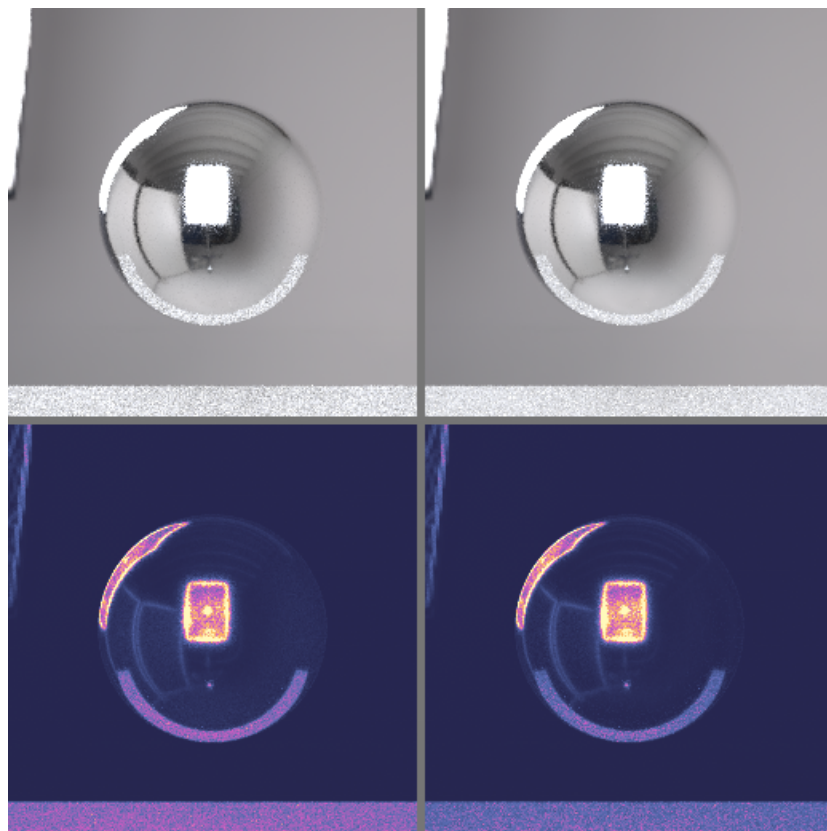
At 16 spp, stratification reduced mean per-pixel standard deviation from 0.17252 to 0.14797 (14.2%). Max-pooled CDF construction reduced noise by 7.6%, higher resolution by 4.5%, and the combined 512×256 max-pooled configuration by 11.6%. These changes are complementary, although not necessarily additive in every scene.

Mean-subtraction compensation produced a scene-dependent result:

HDRI	Noise change
Studio Kontrast	-12.9%
Venice Sunset	-5.2%
Sundowner Overlook	-5.8%
Rogland Clear Night	+2.0%

[Table 5.3.b (p. 31) — Effect of Mean-Subtraction Compensation on Noise Across HDR Environments]

Because compensation improved three environments but worsened Rogland slightly, it remains an optional, scene-dependent optimisation.



[Figure 5.3.c (p. 31) — Effect of Mean-Subtraction Compensation in Studio Kontrast. Note: Above: Studio Kontrast Noise change. Bottom row: Heatmap-style visualization.]

Firefly clamping

At 256 spp, the unclamped maximum was 4294.44. Clamp 3 reduced it to 5.95 but lowered whole-image and geometry means by 40.0% and 61.2%; clamp 50 reduced it to 82.27 while lowering those means by 32.4% and 50.7%. The threshold applies separately to NEE and emitter-hit terms at every bounce, so several capped contributions can accumulate beyond it. Clamping removes outliers effectively but introduces substantial scene-dependent bias and therefore remains disabled by default.



[Figure 5.3.d (p. 32) — Effect of Firefly Clamping. Note: Firefly clamping at 256 spp: unclamped, clamp 3, and clamp 50 (left to right), shown three stops darker. Lower thresholds suppress bright outliers more strongly but introduce greater bias and loss of image brightness.]

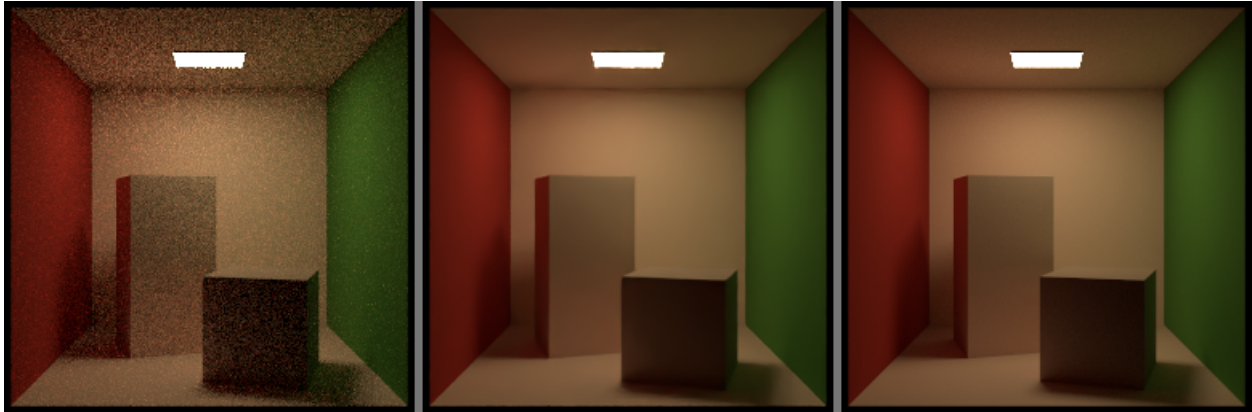
OIDN denoising

OIDN (Áfra, 2026) was evaluated against an independently accumulated 2048-spp Cornell reference, excluding the directly visible emitter restored from the noisy input.

spp	Noisy	Denoised	Reduction
16	0.02439	0.00874	64.2%
64	0.01272	0.00592	53.4%
256	0.00653	0.00409	37.3%

[Table 5.3.c (p. 32) — OIDN Denoising Error by Sample Count]

RMSE fell at every sample count, with diminishing benefit as the input converged. Denoised 16 spp (0.00874) lay between undenoised 128 spp (0.00939) and 256 spp (0.00653), although any effective sample multiplier is scene- and reference-dependent. Gaussian-filtered, spatially correlated noise improved by only about 1.1% over the full image, versus about 12.9% for box-filtered input, confirming that reconstruction filtering affects denoiser performance.



[Figure 5.3.e (pp. 33) — OIDN Denoising Compared With an Independently Accumulated 2048-spp Reference. Note: noisy 16 spp input, OIDN-denoised result, and independently accumulated 2048 spp reference (left to right). Denoising substantially reduces noise while closely preserving the reference scene structure and illumination.]

Unbiased sampling improvements should be preferred first; clamped and denoised images are practical approximations, not evidence of improved estimator correctness.

5.4 Isolated feature validation

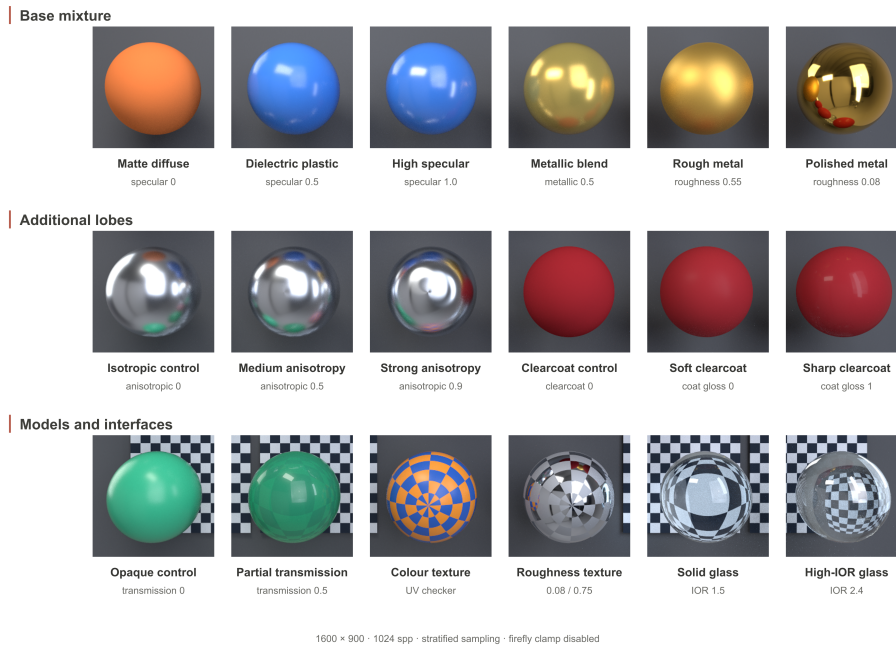
The following controls test whether material, camera, transmission and visibility parameters produce the intended observable changes after their numerical foundations have been established.

Material capability

Figure 5.4.a places eighteen Principled configurations under shared geometry and illumination. It spans diffuse and dielectric mixtures, rough and polished metals, anisotropy, clearcoat, partial transmission, texture-driven colour and roughness, and solid glass at two IORs. Increasing metallic response and lowering roughness sharpen and colour reflections; anisotropy elongates the highlight; textures follow sphere UVs; and IOR alters glass response. The board demonstrates interface range, while § 5.2 establishes numerical correctness.

Material capability map

Eighteen Principled BSDF configurations under shared geometry and illumination

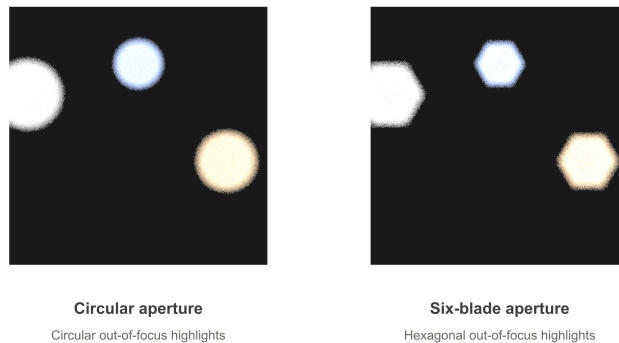


[Figure 5.4.a (p. 34) — Principled BSDF Material Capability Map. Note: Eighteen configurations rendered at 1600 × 900 and 1024 spp with shared geometry, illumination and an unbiased, unclamped path tracer.]

Physical camera

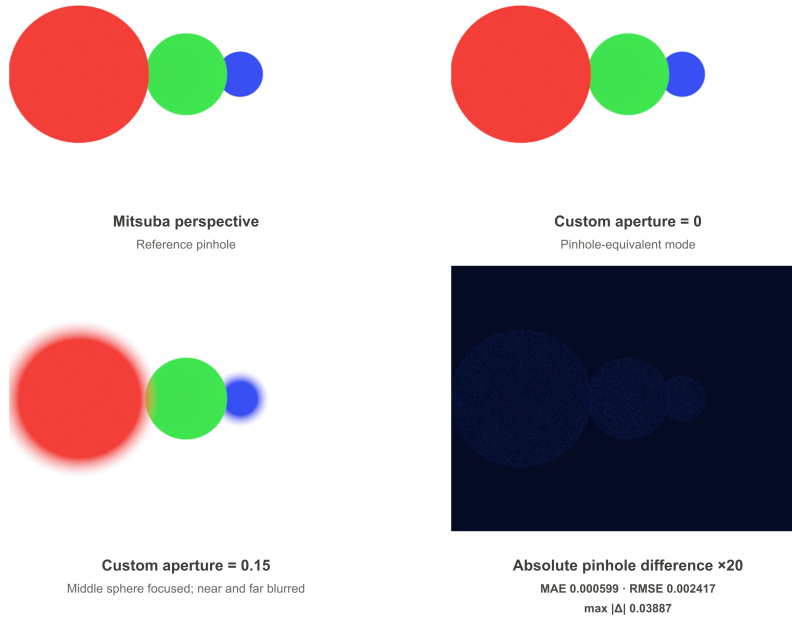
At zero aperture, the custom camera was compared with Mitsuba's perspective sensor using the same scene, film and seed. MAE was 0.000599 and RMSE 0.002417, with no systematic displacement or mirroring. At aperture radius 0.15 and focus distance 5, the middle sphere remains sharp while near and far spheres blur. Bright-point controls show circular highlights for circular sampling and consistently oriented hexagons for six blades.

Aperture-shape validation



[Figure 5.4.b (pp. 34) — Physical-Camera Projection, Aperture-Shape Validation. Note: Circular and six-blade bokeh]

Projection and focal-plane validation



[Figure 5.4.c (pp. 35) — Physical-camera validation: reference and custom pinhole projections, finite-aperture depth of field, and the pinhole absolute difference amplified by 20.]

Focused BSDF behaviour

Matched-position controls remove the spatial confound of earlier multi-sphere scenes. Anisotropic GGX elongates the disc highlight along the tangent direction. With specular disabled, Burley/Lambert remains 1.000 at normal incidence for every roughness; at 80° incidence and exit, ratios of 0.782, 1.422 and 2.253 for roughness 0.1, 0.5 and 0.9 show the expected transition to rough-surface retro-reflection.

Soft and sharp clearcoat change 8.85% and 3.23% of pixels above 0.01 respectively, consistent with concentration into a narrower highlight. White and fully tinted sheen each affect about 2.5%, mainly at the silhouette, with tint changing the grazing response's colour. These fractions locate effects rather than measure physical error.

Anisotropic GGX



Isotropic · 0.0



Anisotropic · 0.8

Elongation follows the disc tangent field.

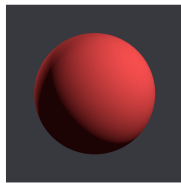
Burley / Lambert response

roughness 0.1					roughness 0.5					roughness 0.9				
	0°	30°	60°	80°		0°	30°	60°	80°		0°	30°	60°	80°
0°	1.000	1.000	0.989	0.853	0°	1.000	1.000	1.008	1.033	0°	1.000	1.000	1.027	1.214
30°	1.000	1.000	0.990	0.871	30°	1.000	1.000	1.014	1.124	30°	1.000	1.000	1.037	1.377
60°	0.989	0.990	0.981	0.874	60°	1.008	1.014	1.031	1.198	60°	1.027	1.037	1.083	1.538
80°	0.853	0.871	0.874	0.782	80°	1.033	1.124	1.198	1.422	80°	1.214	1.377	1.538	2.253

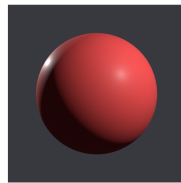
incidence ⊥ exit angle →

Normal incidence: 1.000 at every roughness
80°/80°: 0.782 · 1.422 · 2.253

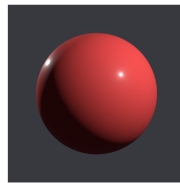
Clearcoat controls



Off



Soft · gloss 0

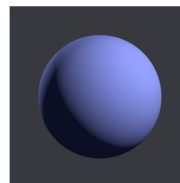


Sharp · gloss 1

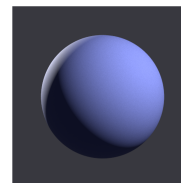
Changed pixels above 0.01
soft 8.85% · sharp 3.23%

Matched position and base response isolate the coat.

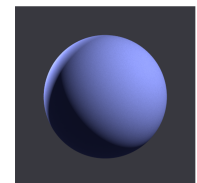
Sheen controls



Off



White · tint 0



Tinted · tint 1

Changed pixels above 0.01
white 2.47% · tinted 2.52%

Response remains concentrated at the silhouette.

[Figure 5.4.d (p. 36)] — Focused BSDF controls: isotropic and anisotropic GGX, Burley/Lambert angle ratios, and matched-position clearcoat and sheen comparisons. All rendered panels use 256 spp.

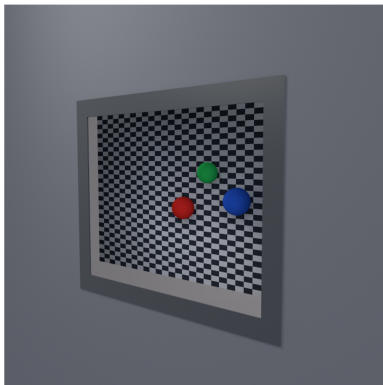
Transmission and visibility

The upper row distinguishes an unchanged open aperture, a thin sheet with straight-through transmission and Fresnel reflection, and a closed slab with thickness-dependent refraction. This complements § 5.2's dielectric equivalence by separating thin and solid interfaces visually.

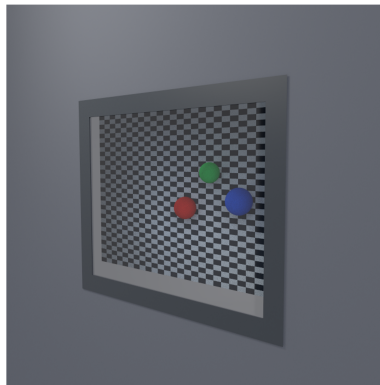
Below, receiver mean luminance is 0.9121 without a pane, zero with binary visibility, and 0.8876 (97.31% of unobstructed) with transparent shadows. This validates an opt-in production approximation, not refracted NEE: shadow rays remain straight and omit Fresnel loss, tint and refraction.

Transmission models

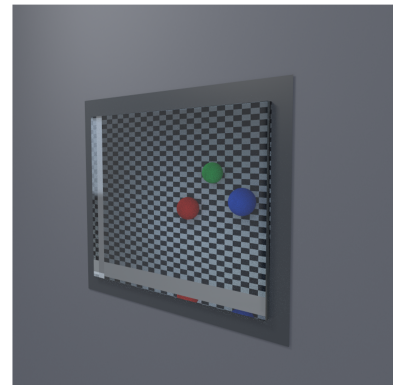
Thin-sheet transmission and finite-thickness refraction under shared scene conditions



Open aperture
No optical interface



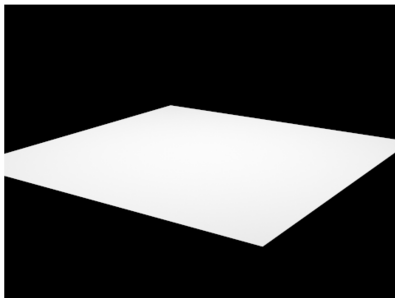
Thin-sheet glazing
Straight-through transmission + Fresnel reflection



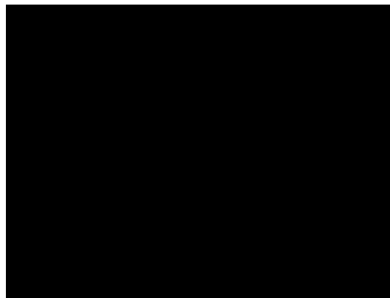
Solid glass slab
Refraction through modelled thickness

Direct-light receiver test

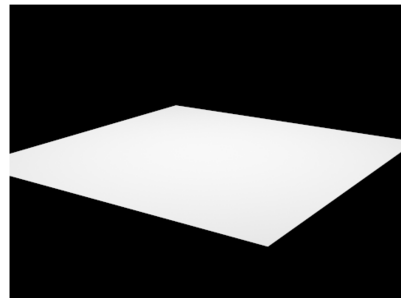
Opt-in NEE visibility approximation through a transmissive pane



No pane
Unobstructed reference



Pane; feature off
Binary NEE visibility



Pane; feature on
Straight-through NEE approximation

Receiver luminance: 0% off → 97.31% of reference on

[Figure 5.4.d (pp. 37) — Thin and Solid Transmission With Direct-Light Visibility Controls. Note: Open, thin-sheet and solid-glass transmission above; unobstructed, blocked and opt-in straight-through direct-light visibility below. Glazing uses 1024 spp and visibility panels 256 spp.]

The controls therefore behave coherently in isolation before being combined in imported and production scenes.

5.5 Imported and production-scene validation

5.5.1 Blender-authored LEGO scene

The LEGO 856 Bulldozer is the first complete-pipeline test using an externally authored scene. It was rendered in Cycles, exported through the Mitsuba-Blender add-on and reconstructed with the custom path tracer, physical camera, environment emitter and Principled BSDF. Camera and environment transforms use Mitsuba’s resolved matrices rather than assumed rotation composition.

The export contains 439 PLY meshes, nine materials and one HDR environment; all references resolved. Exported and reconstructed primary-ray silhouettes achieved IoU 1.000 with no differing pixels at 256×256 , confirming geometry and camera preservation independently of shading.

Comparison	MAE	RMSE	Luminance ratio
Full frame	0.001031	0.006380	0.999675
Geometry region	0.005411	0.014689	0.997757

[Table 5.5.1.a (p. 38) — Quantitative Comparison of the Cycles and Reconstructed LEGO Renders]

The final 1080×1080 , 1024-spp comparison used maximum depth 8 without clamping or denoising. The reconstruction closely matches Cycles framing, environment orientation, colours and brightness. Replacing the zero-specular custom material with Mitsuba diffuse changed the result only at floating-point precision ($\text{RMSE } (1.25 \times 10^{-8})$), confirming the diffuse response.



[Figure 5.5.1.a (p. 38) — Blender Cycles and Custom-Pipeline LEGO Comparison. Note: Blender Cycles reference (left) and full custom-plugin reconstruction (right). Both images use matched resolution, path depth and sample count. The LEGO 856 Bulldozer model by Heinzelnisse (2014) is the first complete-pipeline test using an externally

authored scene. It was rendered in Cycles, exported through the Mitsuba Blender add-on (Mitsuba Renderer Project, n.d.) and reconstructed with the custom path tracer, physical camera, environment emitter and Principled BSDF.]

Differences remain local and traceable. The importer reduces the rubber band’s blended BSDF to a weighted flat colour, changing brightness and saturation; materials named transparent and Red_Glass were exported as diffuse and cannot test transmission. Fine-detail differences may partly reflect the different sampling and reconstruction-filter configurations: the recorded Cycles reference uses tabulated Sobol sampling and a Gaussian filter, while the custom reconstruction uses independent sampling and Mitsuba’s Gaussian filter implementation (Blender Foundation, 2024). The scene therefore validates geometry, camera, environment and simple-material integration, not arbitrary Blender shader compatibility.

5.5.2 Country Kitchen

The Country Kitchen, originally created by Jay-Artist and distributed as a Mitsuba 3 scene by Bitterli (2016), is the principal production-scale integration test, combining authored geometry, varied materials, interior/exterior illumination, complex visibility and multiple cameras.

Its deterministic reconstruction converts 85 of 86 source materials and preserves referenced textures while mapping diffuse, plastic, conductor and dielectric inputs to supported Principled configurations. After required lighting geometry is added, the scene grows from 299 shapes and 86 materials to 301 shapes and 87 top-level definitions, 86 of which use the custom BSDF. Unsupported alpha masks become opaque diffuse materials. The pipeline also installs the custom integrator, importance-sampled environment and physical camera, while replacing the artificial window emitter with explicit thin glazing.

Integration check	Result
Referenced assets	316 references; none missing
Source materials converted	85
Final Principled definitions	86 of 87
Scene geometry	301 shapes
Physical-camera sensors	6
Rebuilt XML	Byte-identical
Render and AOV validity	PASS for all six views

[Table 5.5.2.a (p. 39) — Country Kitchen Scene-Integration Checks]



[Figure 5.5.2.a (p. 40) — Country Kitchen Integration Render. Note: produced using the custom path tracer, Principled BSDF, physical camera and importance-sampled environment emitter. The scene contains 301 shapes and 87 material definitions, including explicit thin window glazing.]

The corrected hero view shows coherent room-scale illumination across textures, metals, glass, practical emitters and indirect transport. Five close-ups isolate material and camera behaviour.



[Figure 5.5.2.b (p. 40) — Stove Close-Up: Metallic Response and Depth of Field]



[Figure 5.5.2.c (p. 41) — Dining-Table Close-Up: Glass Transmission and Contact Shadows]



[Figure 5.5.2.d (p. 41) — Radio Close-Up: Texture Mapping and Fine Geometry]



[Figure 5.5.2.e (p. 42) — Towel Close-Up: Textured Diffuse Response]



[Figure 5.5.2.f (p. 42) — Kitchen-Island Close-Up: Material Variety and Close Occlusion]

All views use 1280 × 720, four independently seeded 256-spp passes (1024 effective spp), stratification, clamp 50 and transparent shadows. Albedo and normal AOVs support OIDN, while raw linear and denoised outputs are both retained. The scene demonstrates integrated operation and, crucially, exposed the scale-dependent visibility defect examined next.

5.6 Scale-dependent failure case study

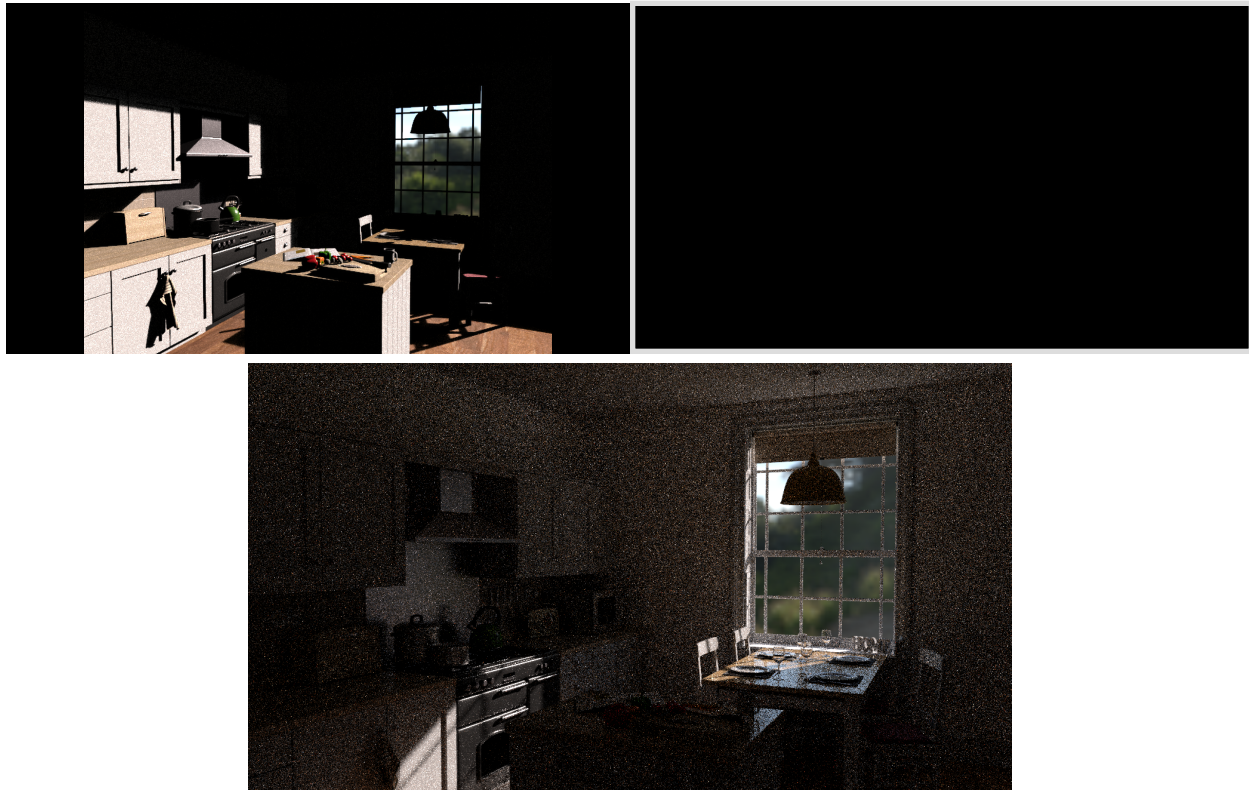
The first Kitchen render contained strong illumination inconsistent with the window direction, suggesting unintended emitters, an opening or faulty environment visibility. Disabling practical lights left the error; removing indirect bounces showed it in direct lighting; and making the window opaque still left the room illuminated. The contribution therefore bypassed the intended opening.

Diagnostic step	Observation	Conclusion
Environment only	Unexpected illumination remained	Practical emitters were not responsible
Direct illumination only	Inconsistent lit regions remained	The problem was not indirect transport
Opaque window	Interior remained illuminated	Environment visibility bypassed the enclosure
Opaque window after correction	Image was exactly black	Occlusion was restored
Transmission restored	Light re-entered through the window	Intended light path remained functional

[Table 5.6.a (p. 43) — Diagnostic Isolation of the Country Kitchen Visibility Defect]



[Figure 5.6.a (p. 43) — Progressive Isolation of the Country Kitchen Lighting Defect. Note: the initial directional inconsistency, environment-only illumination (increased scale), and the bounce-free direct-light diagnostic. Illumination remains in regions without a valid path to the window.]



[Figure 5.6.b (p. 44) — Opaque-Window Visibility Control Before and After the Shadow-Ray Correction. Decisive visibility control. Before correction, illumination remains with the window made opaque; after correcting the environment shadow-ray distance, the same sealed configuration is exactly black. Restoring pane transmission re-establishes illumination through the intended opening.]

The environment emitter set shadow-ray distance to twice a stored bounding-sphere radius. Its default radius of 1 produced shadow rays only 2 scene units long, while the host `set_scene()` callback expected to replace it was not invoked for this Python emitter. Walls farther than two units were therefore missed and the environment was incorrectly considered visible.

The correction uses a safe default radius of 10^6 and allows the callback only to increase it. The corrected direct-only opaque-window render is exactly black; restoring transmission admits light through the intended glazing. Earlier open or small scenes lacked sufficiently distant occluders, so their success is consistent with this scale-dependent failure. The case demonstrates that component tests and production-scale integration expose different error classes.

6. Blender Cycles comparison and discussion

This final comparison asks whether the custom pipeline reproduces coherent Cycles behaviour under closely matched conditions and whether differences can be explained; pixel identity is not expected between distinct material and transport systems.

6.1 Comparison protocol

Property	Comparison configuration
Geometry and cameras	Shared geometry, transforms, UVs, resolution and converted camera parameters
Sampling	1024 spp; Cycles in one run and the custom renderer as four 256-spp passes
Path depth	Maximum depth 17; custom Russian roulette from depth 3
Output	Full-float linear EXR, Box filter and Standard display transform
Denoising	Disabled for direct comparison
Clamping	Disabled in matched controls; value 50 retained separately for production output
Hardware	CPU execution on the same machine

[Table 6.1.a (p. 45) — Blender Cycles Comparison Protocol]

Environment images, intensities and orientations were matched. Blender specifies area-light power as radiant flux in watts; for the one-sided Lambertian emitters used here, it was converted to radiance using $L = P \div (\pi \times A)$ (Blender Foundation, 2024; Pharr et al., 2023). Material parameters were matched where possible, but Blender’s diffuse-glossy mixtures and layered Principled model are not identical to the custom BSDF.

Render time is contextual only. Kitchen views took 317- 420 s in Cycles (mean 363 s) and 194-242 s in the complete custom pipeline (mean 208 s), including scene loading, four passes, AOVs and OIDN. This establishes practical execution, not an intrinsic speed advantage: the custom system relies on Mitsuba/Dr.Jit and implements a smaller feature set.

6.2 Comparison set

Scene	Principal evidence
Material grid	Diffuse, metallic, roughness, clearcoat, sheen, anisotropy, textures and transmission
HDR environment	Environment orientation, directional illumination, shadow and reflection
Physical camera	Framing, focal plane and foreground/background defocus
Country Kitchen	Complete transport, converted materials, glazing, practical lights and complex occlusion

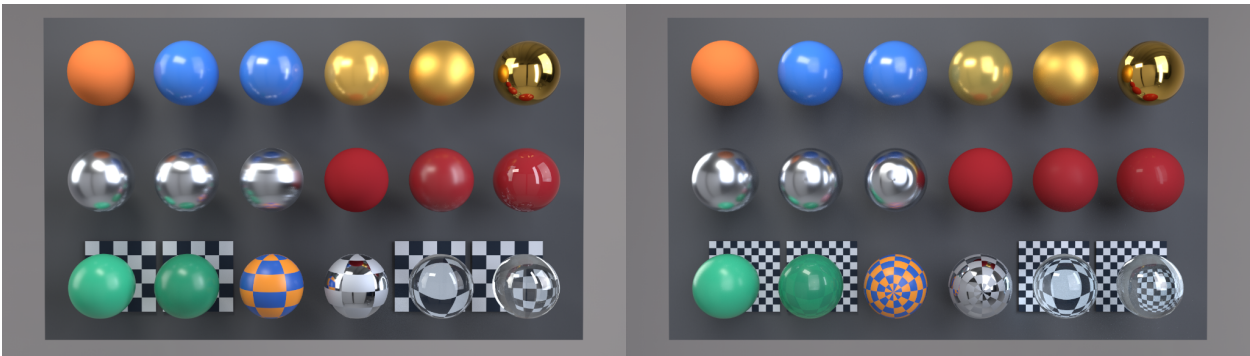
[Table 6.2.a (p. 46) — Comparison Scenes and Principal Evidence]

The first three scenes isolate materials, lighting and camera behaviour; the Kitchen combines all plugins in a hero view and five close-ups. Unclamped Kitchen pairs provide the matched comparison, while clamped and AOV-guided denoised images are reported separately as practical outputs.

6.3 Evaluation

Agreement is strong at all four scales: the custom pipeline preserves intended appearance, spatial structure and physical behaviour without reproducing Cycles pixel-for-pixel.

Figure 6.3.a shows the same diffuse, plastic, metallic, rough, polished, coated and transmissive categories, with mean RGB differing by 3.6%. This is substantial agreement for an independently implemented BSDF. Local highlight differences are consistent with the custom model’s single-scattering GGX, additive GTR1 clearcoat and smooth-only transmission, compared with Blender’s layered Principled model and its documented multiscatter behaviour (Blender Foundation, 2024).



[Figure 6.3.a (p. 46) — Material-Grid Comparison Between Blender Cycles and the Custom Renderer. Note: Cycles (left) and custom renderer (right) material grids.]

The HDR comparison reproduces solar position, horizon, panorama reflection and overall illumination; mean RGB differs by 1.4%. This validates the emitter's lookup, coordinate conversion, radiance evaluation and directional sampling together. Remaining noise differences reflect filtering and importance-sampling implementation rather than orientation.



[Figure 6.3.b (p. 47) — Sundowner Overlook Environment Comparison]

The physical-camera comparison is closest: framing, focal plane and relative defocus are preserved, with mean RGB differing by 0.04% and MAE 0.0011. This strongly supports the thin-lens projection and aperture sampling.



[Figure 6.3.c (p. 47) — Physical-Camera Comparison of Framing and Defocus]

The Kitchen combines all custom plugins across approximately 85 converted materials, explicit glazing, practical lights, complex occlusion and six sensors. Its complete composition, daylight direction, indirect illumination and major material identities remain coherent. Similarity in the isolated HDR scene does not guarantee identical interior brightness because Kitchen illumination must transmit through glazing and undergo multiple material interactions.

Close-ups retain reflective metals, background transmission through glass, radio texture and fine geometry, the towel pattern, contact shadows and material separation; island mean luminance is within 4.4% of Cycles. This demonstrates transfer from synthetic controls to a realistic authored scene.



[Figure 6.3.d (p. 48) — Unclamped Country Kitchen Hero Comparison. Note: Cycles (left) and custom renderer (right) . Unclamped Kitchen hero comparison.]



[Figure 6.3.e (p. 48) — Denoised Country Kitchen Production Outputs. Note: Blender Cycles with integrated albedo/normals-guided Open Image Denoise is shown on the left. The custom renderer's contribution-clamped, AOV-guided OIDN result is shown on the right. Both preserve the same composition and lighting structure; differences in interior brightness and local material response reflect their distinct transport, material and stabilisation policies.]





[Figure 6.3.f (pp. 48–49) — Country Kitchen Close-Up Comparisons: Stove, Table, Radio, Towel, and Island]

Remaining differences are local and explainable. Deliberate anisotropy affects the pot and knife, added clearcoat affects the kettle, and custom glass and straight-through transparent shadows differ from Cycles transport. The Cycles cupboard includes gloss while its converted counterpart has zero specular response, explaining the absent practical-light reflection in the towel view. View-dependent luminance differences therefore cannot be reduced to one exposure offset.

Unclamped custom renders show stronger fireflies near windows and glossy/transmissive paths. Clamp 50 and AOV-guided OIDN produce cleaner presentation at the acknowledged cost of clamp bias. The limitations affect particular lobes and difficult paths rather than preventing credible production-style rendering.

6.4 Critical discussion

The evidence supports a limited positive answer to RQ4: the system approaches Cycles closely where models and inputs are comparable, while the remaining discrepancies are traceable.

Category	Current limitation	Observed consequence
Material model	Single-scattering GGX and additive clearcoat/sheen	Different highlight energy and shape, particularly on metals and coatings
Transmission	Smooth interfaces without absorption, rough glass or dispersion	Different glass appearance and bright transmitted paths
Sampling	Sheen is not sampled directly; bright glossy and transmissive paths retain high variance	More severe fireflies in unclamped Kitchen renders
Stabilisation	Contribution clamping is biased	Cleaner output but a 15–83% view-dependent reduction in mean luminance
Visibility	Transparent shadows approximate straight pass-through rather than refracted transport	Practical light delivery through glazing without exact dielectric visibility
Conversion	Blender shader networks are collapsed into single custom materials	Lost glossy mixtures, bump wrappers and alpha masking
Workflow	Dependence on external assets, Mitsuba interfaces and Dr.Jit execution	Reduced portability and high-resolution memory constraints

[Table 6.4.a (p. 50) — Current Rendering Limitations and Observed Consequences]

These limitations explain why disagreement changes by object and view rather than indicating a single lighting error: the isolated HDR result agrees closely, while local materials and paths differ. Raw comparisons are therefore more informative than one aggregate error.

The lower measured custom runtime likewise establishes practicality, not general superiority. Mitsuba and Dr.Jit supply core infrastructure, and the custom feature set is smaller than Cycles'. Most importantly, the comparison confirms the validation ladder: isolated tests established numerical behaviour, while production integration revealed the shadow-distance defect, conversion losses and clamp bias. Priorities are therefore multiple-scattering energy compensation, rough transmission, direct sheen sampling, Blender-network translation and a dedicated emission AOV. Overall, close agreement combined with explicit, reproducible explanations for differences is the central result.

7. Conclusion

This project translated established physically based rendering techniques into extensible Mitsuba plugins and validated them from numerical controls to production-style scenes. Its contribution is the implementation, integration and reproducible evaluation process rather than new rendering theory or a standalone renderer.

Regarding RQ1, reference-integrator, furnace, PDF-normalisation, statistical-sampling and dielectric comparisons support the tested path tracer, HDR emitter, Principled BSDF and physical camera. They also show that correct equations require consistent sampling, PDFs and framework conventions.

Regarding RQ2, MIS, environment importance sampling, stratification and lobe-aware sampling reduce variance when matched to transport distributions. Independent multi-pass accumulation enables high sample totals under memory constraints. Clamping and OIDN improve presentation but remain separate from raw evidence because clamping is biased and denoising reconstructs radiance.

Regarding RQ3, the validation ladder was essential. The LEGO asset tested reconstruction and compatibility; the 85-material Kitchen combined glazing, complex occlusion, environment illumination and multiple sensors, revealing an environment shadow-ray truncation error absent from smaller tests. Component correctness alone therefore does not ensure system correctness.

Regarding RQ4, the pipeline reproduced Cycles' principal material, environment, camera and interior behaviour, often closely. Remaining differences follow from material/light translation, filtering, post-processing and documented approximations rather than one global integrator fault; pixel identity is neither expected nor required.

The principal limitations are single-scattering energy loss, additive clearcoat and sheen, smooth-only transmission, approximate transparent shadows and restricted DCC interoperability. Future work should prioritise energy compensation, rough transmission and absorption, direct sheen sampling, fuller Blender translation and systematic performance profiling.

References

- Áfra, A. T. (2026). Intel Open Image Denoise (Version 2.5.0) [Computer software]. <https://github.com/RenderKit/oidn/tree/v2.5.0>
- Barresi, D. (n.d.). Sundowner Overlook [HDR image]. Poly Haven. https://polyhaven.com/a/sundowner_overlook
- Bitterli, B. (2016). Rendering resources. <https://noobody.org/resources/>
- Blender Foundation. (2024). Blender 4.3 manual. <https://docs.blender.org/manual/en/4.3/>
- Burley, B. (2012). Physically-based shading at Disney. Walt Disney Animation Studios. <https://disneyanimation.com/publications/physically-based-shading-at-disney/>
- Burley, B. (2015). Extending the Disney BRDF to a BSDF with integrated subsurface scattering. In Physically based shading in theory and practice: SIGGRAPH 2015 course notes. https://blog.selfshadow.com/publications/s2015-shading-course/burley/s2015_pbs_disney_bsdf_notes.pdf
- Cook, R. L., & Torrance, K. E. (1982). A reflectance model for computer graphics. ACM Transactions on Graphics, 1(1), 7–24. <https://doi.org/10.1145/357290.357293>
- Failes, I. (2020, March 24). Meet Freak, Framestore's new in-house renderer. before & afters. <https://beforeandafters.com/2020/03/24/meet-freak-framestores-new-in-house-renderer/>
- Fascione, L., Hanika, J., Pieké, R., Villemin, R., Hery, C., Gamito, M. N., Emrose, L., & Mazzone, A. (2018). Path tracing in production. In ACM SIGGRAPH 2018 courses (Article 15). Association for Computing Machinery. <https://doi.org/10.1145/3214834.3214864>
- Framestore. (n.d.). Freak. Retrieved August 16, 2026, from <https://www.framestore.com/technology/freak>
- Heizelnisse. (2014, January 17). Lego 856 Bulldozer [3D model]. Blend Swap. <https://www.blendswap.com/blend/11490>
- Heitz, E. (2014). Understanding the masking-shadowing function in microfacet-based BRDFs. Journal of Computer Graphics Techniques, 3(2), 48–107. <https://jcgt.org/published/0003/02/03/>
- Hu, C. (2026). pyoidn (Version 2.5.0.1) [Computer software]. <https://pypi.org/project/pyoidn/2.5.0.1/>
- Intel Corporation. (2026). Intel Open Image Denoise documentation (Version 2.5.0). <https://www.openimagedenoise.org/documentation.html>
- Jakob, W., Speierer, S., Roussel, N., & Vicini, D. (2022a). Dr.Jit: A just-in-time compiler for differentiable rendering. ACM Transactions on Graphics, 41(4), Article 124. <https://doi.org/10.1145/3528223.3530099>
- Jakob, W., Speierer, S., Roussel, N., Nimier-David, M., Vicini, D., Zeltner, T., Nicolet, B., Crespo, M., Leroy, V., & Zhang, Z. (2022b). Mitsuba 3 renderer (Version 3.8.0) [Computer software]. <https://github.com/mitsuba-renderer/mitsuba3/tree/v3.8.0>

- Kajiya, J. T. (1986). The rendering equation. *ACM SIGGRAPH Computer Graphics*, 20(4), 143–150. <https://doi.org/10.1145/15886.15902>
- Karlík, O., Šik, M., Vévoda, P., Skřivan, T., & Křivánek, J. (2019). MIS compensation: Optimizing sampling techniques in multiple importance sampling. *ACM Transactions on Graphics*, 38(6), Article 151. <https://doi.org/10.1145/3355089.3356565>
- Kulla, C., & Conty, A. (2017). Revisiting physically based shading at Imageworks. In *Physically based shading in theory and practice: SIGGRAPH 2017 course*. https://blog.selfshadow.com/publications/s2017-shading-course/imageworks/s2017_pbs_imageworks_slides.pdf
- Mitsuba Renderer Project. (n.d.). Mitsuba Blender add-on [Computer software]. GitHub. Retrieved August 16, 2026, from <https://github.com/mitsuba-renderer/mitsuba-blender>
- Pharr, M., Jakob, W., & Humphreys, G. (2023). *Physically based rendering: From theory to implementation* (4th ed.). MIT Press. <https://pbr-book.org/4ed/contents>
- Realistic Graphics Lab. (n.d.). Dr.Jit documentation (Version 1.3.1). <https://drjit.readthedocs.io/en/v1.3.1/>
- Schlick, C. (1994). An inexpensive BRDF model for physically-based rendering. *Computer Graphics Forum*, 13(3), 233–246. <https://doi.org/10.1111/1467-8659.1330233>
- Veach, E. (1997). *Robust Monte Carlo methods for light transport simulation* [Doctoral dissertation, Stanford University]. Stanford Graphics Laboratory. https://graphics.stanford.edu/papers/veach_thesis/
- Veach, E., & Guibas, L. J. (1995). Optimally combining sampling techniques for Monte Carlo rendering. In *Proceedings of the 22nd annual conference on computer graphics and interactive techniques* (pp. 419–428). Association for Computing Machinery. <https://doi.org/10.1145/218380.218498>
- Wald, I., Woop, S., Benthin, C., Johnson, G. S., & Ernst, M. (2014). Embree: A kernel framework for efficient CPU ray tracing. *ACM Transactions on Graphics*, 33(4), Article 143. <https://doi.org/10.1145/2601097.2601199>
- Walter, B., Marschner, S. R., Li, H., & Torrance, K. E. (2007). Microfacet models for refraction through rough surfaces. In J. Kautz & S. Pattanaik (Eds.), *Rendering techniques 2007: 18th Eurographics Symposium on Rendering* (pp. 195–206). Eurographics Association. <https://doi.org/10.2312/EGWR/EGSR07/195-206>
- Whitted, T. (1980). An improved illumination model for shaded display. *Communications of the ACM*, 23(6), 343–349. <https://doi.org/10.1145/358876.358882>
- Wronkowski, G. (n.d.). Studio Kontrast 04 [HDR image]. Poly Haven. https://polyhaven.com/a/studio_kontrast_04
- Zaal, G. (n.d.-a). Rogland Clear Night [HDR image]. Poly Haven. https://polyhaven.com/a/rogland_clear_night
- Zaal, G. (n.d.-b). Venice Sunset [HDR image]. Poly Haven. https://polyhaven.com/a/venice_sunset