DOI: 10.1111/cgf.13781
Eurographics Symposium on Rendering 2019
T. Boubekeur and P. Sen
(Guest Editors)

Volume 38 (2019), Number 4

Global Illumination Shadow Layers

François Desrichard, David Vanderhaeghe, Mathias Paulin

IRIT, Université de Toulouse, CNRS, INPT, UPS, UT1, UT2J, France

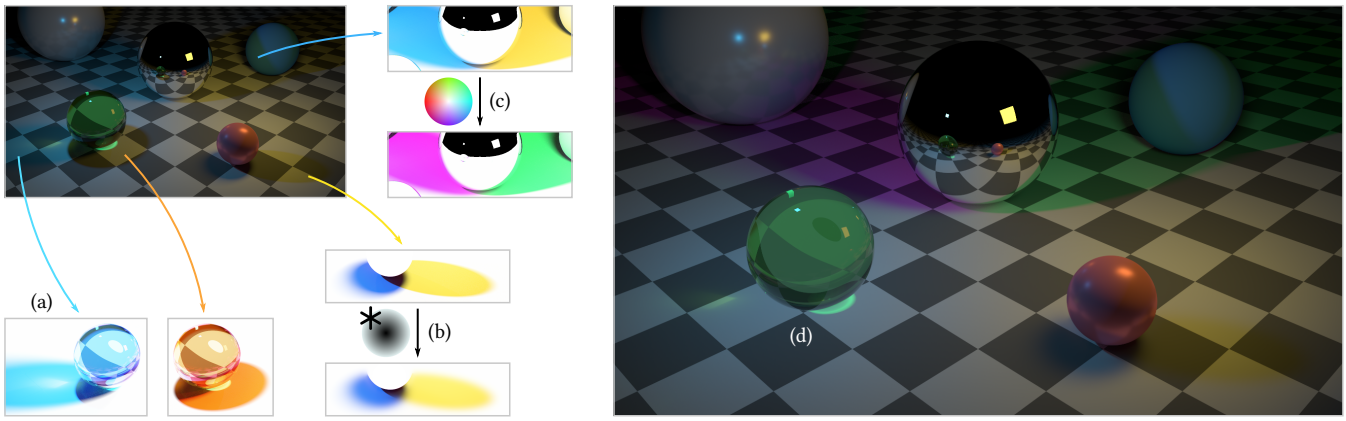


Figure 1: *(a) Our approach computes shadow layers under global illumination, taking into account direct and indirect shadows for each light / object pair. Compositing tasks involving shadows such as (b) shape smoothing, (c) color grading, and (d) removal are easily achieved.*

Abstract

Computer graphics artists often resort to compositing to rework light effects in a synthetic image without requiring a new render. Shadows are primary subjects of artistic manipulation as they carry important stylistic information while our perception is tolerant with their editing. In this paper we formalize the notion of global shadow, generalizing direct shadow found in previous work to a global illumination context. We define an object's shadow layer as the difference between two altered renders of the scene. A shadow layer contains the radiance lost on the camera film because of a given object. We translate this definition in the theoretical framework of Monte-Carlo integration, obtaining a concise expression of the shadow layer. Building on it, we propose a path tracing algorithm that renders both the original image and any number of shadow layers in a single pass: the user may choose to separate shadows on a per-object and per-light basis, enabling intuitive and decoupled edits.

CCS Concepts

• ***Computing methodologies*** → ***Rendering; Ray tracing;*** *Non-photorealistic rendering;*

1. Introduction

In cinematographic production, the post-processing of a 3D render involves thorough editing of light effects. Lighting artists typically agree with compositing artists on a set of useful images to export aside from the main render. These images are called Arbitrary Output Variables (AOVs), and may contain any useful by-product of the rendering process. For example, exporting diffuse and specular light contributions in separate AOVs offers artists the freedom to manipulate each component individually during compositing, without requiring a new render of the shot.

Shadows are an important source of artistic expression. In today's compositing pipelines, they commonly undergo intensity correction, color grading and shape smoothing. These manipulations are made possible by the human tolerance to non-realistic edits: perceptual studies show that the eye does not fully account for physical correctness [HBM*14] nor shape congruence [SSMK05]. However, inspecting the output of state-of-the-art renderers, we underline several liabilities in the way they isolate the shadow component of a render. Most notably, they are limited to the extraction of shadows created by direct lighting.

To overcome the limitations of current renderers, we define the *shadow layer* associated with an object of the scene. A shadow layer contains the radiance lost on the camera film due to the presence of the considered object. By adding the shadow layer to the original image, which we call the *main layer*, the radiance lost because of the object is recovered, and shadow is canceled. Our definition takes into account indirect shadows, namely shadows created by indirect illumination. The shadow layer can be freely manipulated before being composited with the main layer, allowing for a variety of edits as illustrated in Figure 1.

This paper makes the following contributions:

- A general and intuitive definition of shadow layer which is compatible with existing global illumination algorithms.
- A concise characterization of the shadow layer in the path space that is amenable to Monte-Carlo integration.
- A path tracing algorithm rendering both the main layer and any number of decoupled shadow layers in a single pass.

The shadow layer can be computed as the difference between two altered renders of the scene, generated using any global illumination algorithm (Section 3.1). Although very general and intuitive, this first method does not provide user control on the content of the shadow layer and requires two renders per considered object, which is intractable in a realistic production context.

To devise a more versatile and efficient solution, we translate the previous definition in the path space formulation of the light transport equation (Section 3.2). We show that shadow is concisely expressed as the unoccluded radiance contribution from a subset of all light paths in the scene, and that the rendering of shadow layers is amenable to Monte-Carlo integration.

We propose a variation of the path tracing algorithm that renders the main layer along with any number of shadow layers in a single rendering pass (Section 4). Our path tracer extracts shadow layers on a per-object and per-light basis to decouple the different lighting effects in the scene. It reuses part of the information that is computed but eventually discarded when using a standard path tracer. Enabling the export of shadow layers impacts render time with a factor between 1.1 and 1.3. Sampling budget is now distributed among all generated images; therefore, we measure how exporting additional shadow layers affects convergence (Section 5.2).

2. Related Work

Inverse design methods The propagation of light from a source and its interactions with objects in a scene are complex and hardly anticipated. A whole body of work investigates inverse methods, where lighting is automatically set up to meet a given goal. Painting and sketching metaphors are extensively used for their familiarity [PBMF07], most notably by Poulin *et al.* [PF92, PF95, PRJ97]. The relationship between a direct shadow and the occluded light source can also be inverted to allow for intuitive edits, such as dragging across a surface [PTG02]. Alternatively, the automated method proposed by Bousseau *et al.* [BCRA11] optimizes directional lighting to emphasize surface features. These methods formulate a user's objective and optimize direct lighting to best match it. Conversely, we are interested in shadows as a byproduct of global illumination, and offer direct control over them during compositing.

Artistic control of light transport In many occasions, lighting artists need to deviate from realistic simulation of light transport to create unique effects. In RayPortals [SMVP17], the continuity of light propagation can be bypassed with teleporting surfaces; linear transformations of light paths were also used to retarget lighting throughout the scene [SNM*13]. With iCheat [OKP*08], the user has affine control over the light transport coefficients in the scene. Nowrouzezahrai *et al.* [NJS*11] break down and edit the emissive properties of participating media. Our method also alters light propagation to isolate and render global shadows, but user control is deferred to post-processing.

BendyLights [KPD10] offers a variation on the spot light primitive, where the emission shape is deformed using a set of control points. Obert *et al.* [OPP10] focus on direct directional lighting and interactively transform the visibility of objects in the scene, allowing for a variety of edits on the corresponding shadows. Our work shares a similar motivation but focuses on global illumination, without any constraint on lighting.

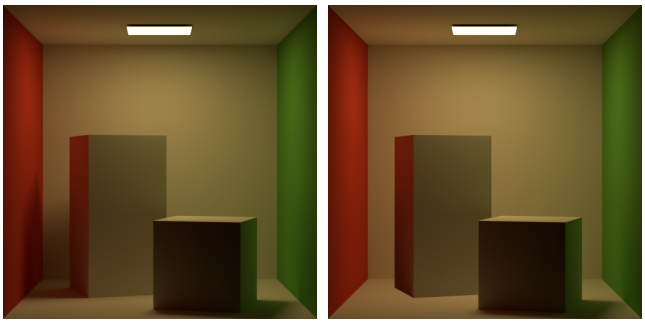
Surface manipulation Because most lighting effects are primarily visible on surfaces, it is intuitive to edit them directly on the scene's geometry. Mattausch *et al.* [MIW13] propose to deform the boundary of an object's direct shadow via control points projected on the receiving surface. Mirror reflections can also be edited by enforcing a constraint on both reflective and reflected objects [ROTS09]; more generally, any signal defined on the surface of an object such as textures or shadows can be relocated using a locally invertible transformation [RTD*10]. Instead, and much like Stylized Shadows [DCFR07], our work focuses on screen-space edits but takes into account indirect shadowing.

3. From Image-Based to Path Space Shadow Layers

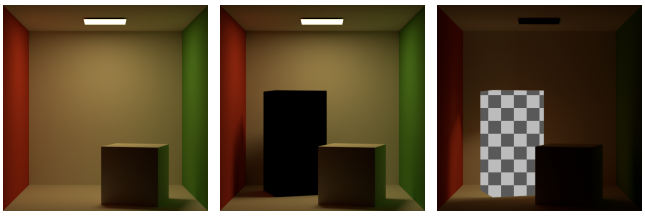
Intuitively, one often defines a shadow as an area that lacks light relative to its surroundings. Computer graphics artists are used to working with synthetic objects whose extent and purpose are clearly identified, and naturally associate a shadow with the object that produces it. We follow this approach and settle a permissive definition of an object: object $\mathbf{O}$ is characterized by its surface $\mathcal{M}_\mathbf{O} \subset \mathcal{M}$, where $\mathcal{M}$ is the set of all surfaces inside the scene. It is up to the user to choose a proper surface, and we make no further assumption on its properties. For example, $\mathcal{M}_\mathbf{O}$ might be non-manifold or display a disconnected topology.

We separate objects in two kinds, assuming that the user is interested in editing a particular shadow. As found in literature, we call *caster* the object $\mathbf{C}$ that produces this shadow by occluding incident light, and *catcher* the object exhibiting a local loss of radiance on its surface as a result. In the case of self-shadowing, caster and catcher are the same object.

In this section, we describe the acquisition of the shadow layer associated with a particular caster using two different methods. The first method computes it as the difference between two altered renders of the scene. It introduces the intuition, concepts and definitions of our approach, but lacks user control and computational efficiency. The second method overcomes these limitations; based on a characterization of shadow in the path space formulation of light transport, it enables the rendering of several shadow layers in one pass, supports self-shadowing and per-light separation of shadows.



(a) Main layer I (b) Removed shadow $I+S$



(c) Invisible T (d) Black body B (e) Shadow layer S

Figure 2: *(b) Removing the shadow created by the box in the background. (e) The shadow layer, defined as the difference $S = T - B$, is added back to the main layer I using a matte.*

3.1. Image-Based Acquisition

The influence of the caster on light propagation is primarily dictated by the extent of its surface $\mathcal{M}_{\mathbf{C}}$, and further modeled by its Bidirectional Scattering Distribution Function (BSDF). In order to quantify the amount of light occluded by the caster, we consider the difference between two renders of the scene: one where $\mathbf{C}$ is invisible, and one where it is considered a black body at 0° K.

In practice, rendering the image where $\mathbf{C}$ is invisible simply means removing it from the scene. For the consistency of our derivations however, we consider that its surface remains, but replace the BSDF by a Dirac distribution that transmits incident radiance unaltered in a straight line. Considering $\mathbf{C}$ a black body at temperature 0° K implies that the BSDF absorbs all incoming radiative energy, and re-emits none.

The image T where the caster is invisible is the result of light propagating without the interference of $\mathbf{C}$ (Figure 2c). Conversely in the black body image B (Figure 2d), all radiance reaching the caster is absorbed instead of scattering on the surface. As such, the caster's only effect on the light field is to create shadows. The difference between these two renders quantifies the amount of radiance lost on the camera film because of the presence of $\mathcal{M}_{\mathbf{C}}$ inside the scene, as seen in Figure 2e. Any direct or indirect shadow created by the caster can be removed by adding the shadow layer to the main layer (Figure 2b).

This definition yields an acquisition method that is agnostic to the algorithm used for the render, but it has two shortcomings. The first one is that the shadow catcher being considered is invariably the camera film itself, which creates inconsistencies and prevents the extraction of self-shadowing. As seen in Figure 2d, the image B is completely black where the caster is directly visible. This means that after subtraction from T, the resulting shadow layer $S = T - B$ is equal to T inside the object's silhouette, showing the back wall through the box. Adding the whole shadow layer to the main layer would create artifacts in the area, which is why a matte of the object is used to limit the area of influence of the shadow layer.

But the main liability of this approach is that two additional renders are needed per shadow layer, aside from the main layer. This is a substantial computational overhead for a single object: in a realistic context, artists need to retain maximum flexibility during compositing and tag multiple objects whose shadow they want to edit. Given N such tagged casters, the previous method requires $2N+1$ renders in total, which becomes impractical. Our path tracing implementation solves both issues (Section 4).

3.2. Characterization on the Path Space

To obtain the shadow layers of multiple casters in a single render pass, we translate the previous definition in the path integral formulation of the light transport equation. Using the definitions and notations of Veach [Vea97], a light path $\bar{x}$ of length k formed of $k+1$ vertices x_i is denoted

$$\bar{x} = \mathrm{x}_0\, \mathrm{x}_1 \ldots \mathrm{x}_k$$

with light propagating from x_0 to x_k. The set of all paths of finite length is called the path space Ω. It contains all geometric paths that can be formed inside a given scene, with no consideration for mutual visibility between the successive vertices. The measurement contribution function f_j determines the throughput of a given light path. The goal of Monte-Carlo algorithms is to sample and integrate the fraction of the path space that intersects a virtual light sensor on the camera. The measured radiance I_j on the sensor is expressed as the integral over Ω of path contributions $f_j(\bar{x})$ weighted by the area-product measure $\mathrm{d}\mu(\bar{x})$:

$$I_j = \int_\Omega f_j(\bar{x})\, \mathrm{d}\mu(\bar{x}) = \int_\Omega f_j\, \mathrm{d}\mu$$

Replacing the caster's BSDF to obtain T_j and B_j changes f_j into $f_{T\,j}$ and $f_{B\,j}$ respectively. As we ensured that $\mathbf{C}$ is geometrically present inside the scene even when invisible, the area-product measure does not change (see Appendix A). We express the measurements of the altered renders as

$$T_j = \int_\Omega f_{T\,j}\, \mathrm{d}\mu \quad \text{and} \quad B_j = \int_\Omega f_{B\,j}\, \mathrm{d}\mu$$

Let us write the difference between these two measurements:

$$T_j - B_j = \int_\Omega f_{T\,j}\, \mathrm{d}\mu - \int_\Omega f_{B\,j}\, \mathrm{d}\mu = \int_\Omega (f_{T\,j} - f_{B\,j})\, \mathrm{d}\mu \qquad (1)$$

we then introduce the subset of Ω comprised of all paths with at least one vertex lying on $\mathcal{M}_{\mathbf{C}}$:

$$\Omega_{\mathbf{C}} = \{\bar{x} = \mathrm{x}_0\, \mathrm{x}_1 \ldots \mathrm{x}_k \mid \exists\, i \in [\![0,\, k]\!], \mathrm{x}_i \in \mathcal{M}_{\mathbf{C}}\}$$

and separate Equation (1) in two terms:

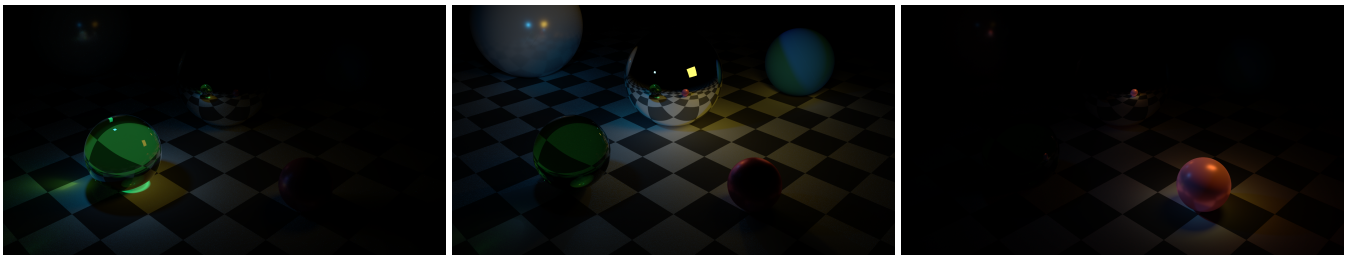


Figure 3: *Scattered layers associated with the front three spheres of Figure 1. In our approach, scattered layers are computed as $I - B$ and contain the radiance impact of an object on the scene. These layers are usually defined using Heckbert notation and rendered inside AOVs.*

$$T_j - B_j = \int_{\Omega_\mathbf{C}} (f_{T\,j} - f_{B\,j})\,\mathrm{d}\mu + \int_{\Omega\setminus\Omega_\mathbf{C}} (f_{T\,j} - f_{B\,j})\,\mathrm{d}\mu$$

As $f_{T\,j}$ and $f_{B\,j}$ are both equal to the original f_j outside of $\Omega_\mathbf{C}$, the second term cancels out. Furthermore, $f_{B\,j}$ is null on $\Omega_\mathbf{C}$ by definition of a black body, meaning that if we denote S_j the difference,

$$S_j = T_j - B_j = \int_{\Omega_\mathbf{C}} f_{T\,j}\,\mathrm{d}\mu \tag{2}$$

which reads that the shadow layer contains the contribution of all light paths encountering the caster, considering that the latter is invisible. It is a positive quantity expressing the local loss of radiance on the sensor due to the caster. This definition is concise and intuitive, and because it is an integration over a subset of Ω, we build on Monte-Carlo algorithms to obtain it in a very general manner, coherently with other phenomena such as depth of field.

In the same way, subtracting the black body render from the main layer yields the unaltered contribution of all paths encountering the caster. We call this image the *scattered layer* (Figure 3); following the same type of derivations, it is expressed as

$$I_j - B_j = \int_{\Omega_\mathbf{C}} f_j\,\mathrm{d}\mu$$

This kind of layer is usually defined using Heckbert notation [Hec90]. Many renderers implement Heckbert notation in the form of light path expressions [Gri16], which share some of their syntax with regular expressions. For instance, the previous layer could be rendered as a separate AOV using the expression `.*<[RT].'caster'>.*`.

4. Efficient and Practical Implementation

As an input, the user tags N objects as shadow casters of interest $(\mathbf{C}_i)_{i\in[\![1,\,N]\!]}$. Building on path tracing for its simplicity and widespread use, we present an algorithm rendering the N shadow layers $S_{\mathbf{C}_i}$ at the same time as the main layer I.

4.1. One-Pass Extraction

The algorithm must be able to sample Ω according to the original contribution function f, but also each subspace $\Omega_{\mathbf{C}_i}$ for $i \in [\![1,\,N]\!]$ while measuring contribution with the corresponding $f_{T\,\mathbf{C}_i}$. To do so, we modify the standard path tracing algorithm on two points:

- During the propagation of a path in the scene, we decide which layer it should contribute to in an unbiased manner, according to its successive encounters with objects.
- When gathering direct lighting at a vertex, the shadow ray may ignore occlusion in the way. As a consequence, normally discarded light paths can be reused by the algorithm.

Propagation For each sample, the path tracing algorithm incrementally builds a path starting from the camera by propagating rays inside the scene. Initially, a path propagates as usual and contributes to the main layer. The first time a caster $\mathbf{C}_i$ is encountered, two possibilities arise: as the path now belongs to $\Omega_{\mathbf{C}_i}$, throughput may be measured according to either f or $f_{T\,\mathbf{C}_i}$. Measuring with f implies scattering normally on the surface, while measuring with $f_{T\,\mathbf{C}_i}$ is equivalent to skipping the intersection. Both outcomes have a probability $p = 1/2$, ensuring an unbiased behavior.

- If the ray skips $\mathbf{C}_i$, the latter becomes the *assigned caster* of the path. Contributions will go to $S_{\mathbf{C}_i}$ for the rest of the propagation, and $\mathbf{C}_i$ will always be skipped. No further choice is given to the path when intersecting another caster.
- If the ray scatters normally on $\mathbf{C}_i$, the path still contributes to the main layer. It will interact normally with $\mathbf{C}_i$ for the rest of the propagation. $\mathbf{C}_i$ can never be considered as a potential assigned caster again. However, another caster intersected later can still be assigned to the path.

Each performed choice introduces a normalization factor of $1/p$ in radiance contributions.

Shadow rays At each vertex, the path tracing algorithm gathers direct lighting from a random source. It performs a visibility test that must remain coherent with the history of the path.

- If the path has an assigned caster $\mathbf{C}_i$, the shadow ray skips it during the visibility test, just like propagation. Any direct lighting contribution goes to the shadow layer $S_{\mathbf{C}_i}$.
- If the current path does not have an assigned caster:
  - If no object was hit when testing for visibility, the contribution goes to the main layer.
  - Otherwise, the first occluder in the way is skipped if it is a tagged caster $\mathbf{C}_i$ that has not been discarded for assignment. If the shadow ray connects with the source, the contribution goes to the corresponding shadow layer $S_{\mathbf{C}_i}$.

Different cases are illustrated in Figure 4.

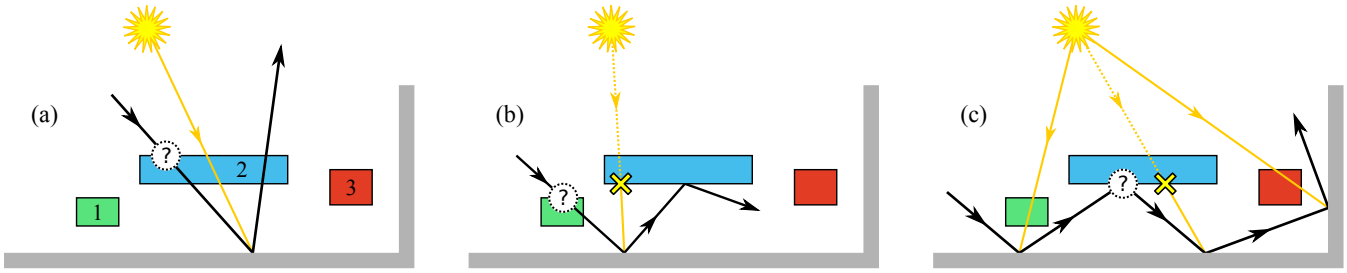


Figure 4: *Three examples of path construction with objects **1**, **2**, and **3** as casters. (a) The blue caster **2** is hit and assigned to the path; it is skipped during propagation and shadow ray testing. (b) The green caster **1** is assigned to the path; caster **2** is thus considered a normal object for both path construction and shadow rays. (c) The ray hits **2**, which is not assigned to the path. Further intersection and shadow ray tests consider it a normal object. For shadow rays, other casters are skipped and the radiance contribution goes to their shadow layer.*

4.2. User Control

Shadow Catchers As stated in Section 3, a shadow catcher is any object exhibiting a loss of radiance on its surface because of a caster. In the image-based acquisition, the catcher being considered is the camera film itself, and a matte had to be used to obtain coherent compositing results when removing shadow (Figure 2).

By default, we change this behavior and measure lost radiance on the surface of objects in the scene: a ray cannot skip a caster before it has scattered on a surface. For greater artistic freedom, we allow the user to define their own set of admissible catchers. In the same spirit, a ray starts propagating normally, and cannot skip a caster before is has scattered on a catcher. Additionally, we provide the option to discard self-shadowing for a caster that is also a catcher.

Per-light separation An object casts as many shadows as there are light sources in the scene; these shadows can overlap in the corresponding layer and become tedious to edit. We allow the user to separate the shadows per-light by creating a layer for each light and caster pair. If there are L sources in the scene, $N \times L$ layers are thus available for decoupled editing.

Direct and indirect components The location of direct shadows can easily be anticipated by the user from the relative position of the light and caster. However, indirect shadowing implying multiple light bounces, it is less predictable. We allow the user to separate the two components inside separate layers.

5. Results and Comparison

We compare the shadow layers produced by our approach against available renderers, and demonstrate various edits possible using our decomposition. We then assess the computational overhead of our approach compared to standard path tracing.

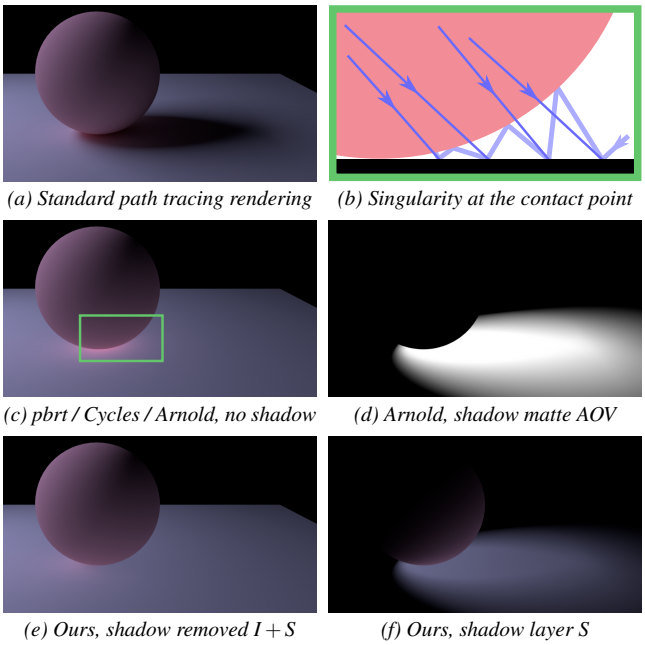
5.1. Improved Shadow Editing

Our method renders separate shadow layers that can be added back to the main layer to remove shadows. Many renderers do the opposite, proposing an option to turn off shadows per-object to generate an image without shadows. Figure 5a displays a simple scene calibrated for an equal result using three renderers, where we then disabled shadow casting from the sphere:

- pbrt version 3, path tracer, material parameter "shadowalpha"
- Blender 2.78 Cycles, object option "Shadow"
- Arnold for Maya 3.1.2, shape option "Cast Shadows"

Inspecting the source code of pbrt-v3, we reckon that all three underlying path tracing algorithms systematically ignore occlusion when shooting shadow rays for direct lighting. This behavior shares the same motivation as our modification of shadow rays described in Section 4; however without altered propagation, it is not sufficient to account for indirect shadows.

Moreover, light is overestimated near the contact point between the sphere and the base plane with all three renderers, as seen in Figure 5c: after each bounce, the area light placed at the top left



(a) Standard path tracing rendering *(b) Singularity at the contact point*

(c) pbrt / Cycles / Arnold, no shadow *(d) Arnold, shadow matte AOV*

(e) Ours, shadow removed $I + S$ *(f) Ours, shadow layer S*

Figure 5: *Comparison of shadow extraction with different renderers. Conventional path tracers exhibit a singularity at the contact point between the sphere and the base plane. Our method correctly extracts the shadow of the sphere, which can then be edited.*

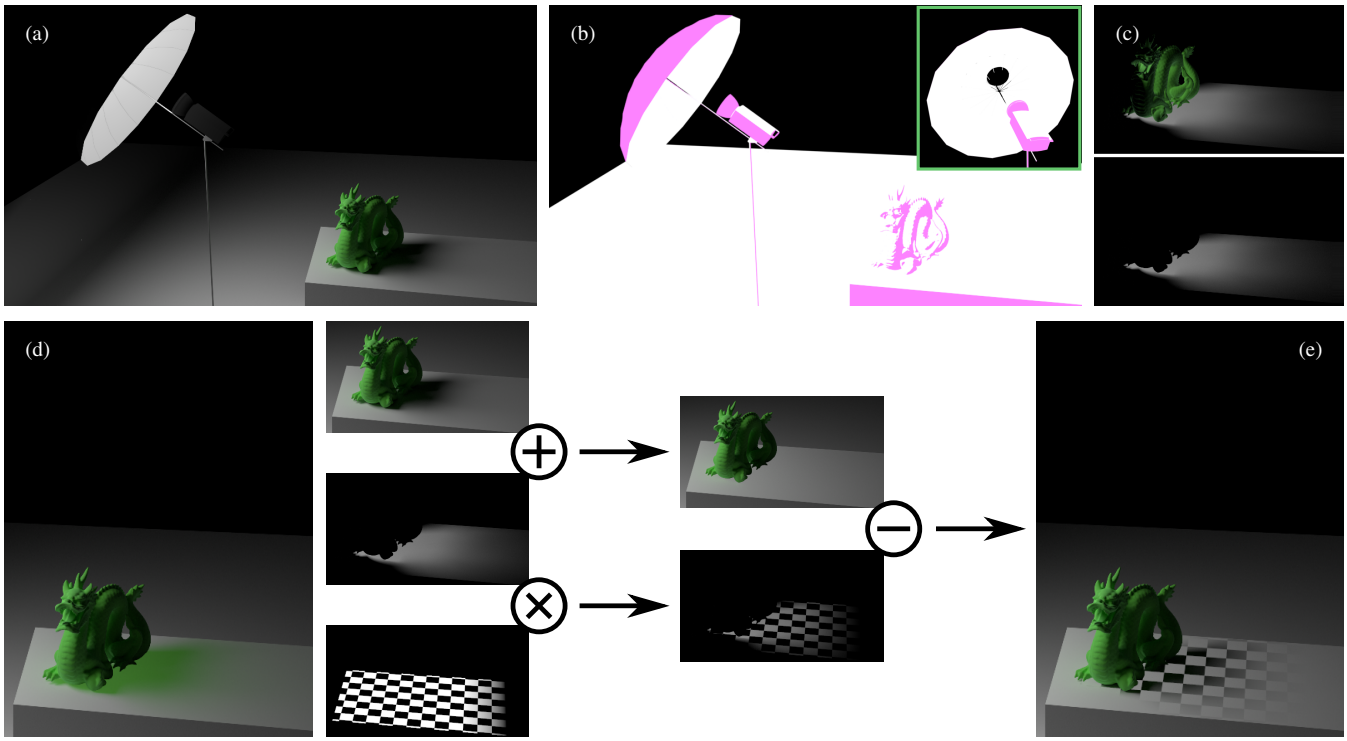


Figure 6: *(a) Dragon scene, indirect lighting. (b) Arnold's shadow matte fails to distinguish indirect shadows (white indicates shadow). Backfacing surfaces (colored in pink for clarity) are considered lit, and only one area near the apex of the umbrella is correctly marked. The rest of the scene is considered to be in shadow. (c) Our shadow layer contains exclusively indirect shadows, with (top) or without (bottom) self-shadowing enabled. (d) Shadow layer edited to change color. (e) A compositing graph enables versatile editing of the shadow.*

of the sphere contributes a near-constant amount of radiance (Figure 5b). Conversely, by adding the sphere's shadow layer S back to the main layer I, we obtain a coherent result where shadow has been removed from the image (Figure 5e). Light bleeding on the back of the sphere is captured without overestimating light contribution, as seen in the shadow layer of Figure 5f.

We determined experimentally that Arnold's shadow matte AOV (Figure 5d) represents the ratio of occluded shadow rays. This shadow matte is often used as an empirical factor to drive compositing effects such as smoothing, while our shadow layer has an intrinsic physical meaning.

To underline the absence of indirect shadows in standard renderers, we set up Figure 6. A single light source from the lamp is reflected before reaching the dragon; thus, whereas the triangle count is around 215,000, the scene's complexity is that of light transport. Arnold's shadow matte AOV is presented for comparison and fails to detect the presence of indirect shadow. The rear of the dragon and the side of the stand are not considered in the shadow matte since they backface the source, and are rendered in black by Arnold; we colored them in pink for illustration purpose. The top of the stand is uniformly white as direct lighting is occluded. The inside of the umbrella is mostly white as direct lighting is occluded by the reflector of the lamp; only a small area near the apex is correctly marked as lit, as shown in the inset. On the contrary, our algorithm correctly computes indirect shadow and allows the user to edit it.

Several edits are performed in Figure 1, which contains two area lights hovering above the foreground spheres. The scene has very few triangles as the spheres are parametric primitives, but they create complex patterns such as caustics and indirect shadows. Shadow removal of the green glass sphere is achieved by adding the corresponding shadow layer to the main layer. To color grade the mirror's shadow, we rotate the hue of the shadow ratio $I/(I+S)$. This quantity, also called *visibility ratio* [OPP10], is useful for user editing; the V-Ray renderer generates a *raw shadow* AOV following a similar equation. Finally, the shadow ratio of the red plastic ball is blurred, with self-shadowing discarded.

The Moana Island scene (Figure 7) showcases complex geometry (48M triangles, 0.5M curves) and lighting (23 light sources). The key lights are a sky dome, providing uniform lighting over the hemisphere, and the sun, whose distant emission creates strong shadows. In the part of the scene presented, two plants occlude most of the radiance and hide details in their shadow. Traditional color grading affects the whole image and must be fine-tuned by the user; instead, compositing the shadow layer brightens the scene, with a physically coherent and localized result. Because self-shadowing was discarded for all casters, the foliage remains untouched. The alternative view in Figure 8 was rendered with only the sun and hemispheric lights put into separate shadow layers, without self-shadowing. While the sand has a high-frequency texture, the shadow ratio is mostly uniform in the shaded areas and allows the user to transform the shape of shadows.

Figure 7: *Left: the hibiscus (left) and pandanus (top) occlude most of the light coming from the sun, in the top right corner. Here, grading the color histogram would affect the whole image. Right: adding our shadow layers recovers lost radiance locally and brightens the scene.*

5.2. Performance Analysis

We implemented our method in pbrt-v3, and compared it to the standard path tracing algorithm; the results are presented in Table 1. For each scene, N is the number of rendered shadow layers, without per-light or direct / indirect shadow separation; where $N = 0$, the standard path tracer was used. The first column displays the total number of samples computed at each pixel; this sampling budget is shared among all the layers that are being generated. The benchmarking machine runs an Intel Xeon E5-2630 v4 processor with 20 threads at 2.20 GHz, and has 64 GiB RAM.

When rendering at least one shadow layer along with the main layer, we observe an increase in render time of a factor varying between 1.1 and 1.3. It is mainly due to the additional intersection tests that must be performed when skipping a caster repeatedly during propagation or direct light gathering. As N increases, we observe that the render time keeps increasing slightly. Indeed, managing several images takes some additional time, for example when looping over the reconstruction filter's support after each measure. Storing additional layers has a predictable footprint on memory, costing the equivalent of a full resolution image per layer.

However, the variations in the number of radiance contributions that each image receives mean that time alone is not a sufficient factor for comparison. Assessing render quality using Root-Mean-Square Error (RMSE) or relative RMSE is inaccurate on the shadow layers, as they mostly contain null values. We thus compute the Structural Similarity (SSIM) [BSS04] relative to a reference render involving at least 16 times more samples, with a radius of 5 and $\sigma = 1.5$. A SSIM of 1 means indistinguishable images, and 0 no similarity. Given a fixed sample budget, SSIM decreases with the number of shadow layers being rendered. When many shadow layers span a large image-space support, all images particularly suffer from undersampling.

We also provide the percentage of Zero Radiance Paths (ZRP) that are simulated but eventually discarded as they bring no energy. This percentage systematically decreases with increasing N, which confirms that computing all images in one pass allows us to reuse otherwise lost information.

Scene	N	Samples	Time	SSIM	ZRP
Teaser (Figure 1)	0	2048	15' 27"	0.903	26%
	1	2048	17' 27"	0.901 / 0.983	17%
	2	2048	18' 07"	0.896 / 0.956	12%
	3	2048	18' 28"	0.896 / 0.956	10%
Dragon (Figure 6)	0	4096	27' 53"	0.927	91%
	1	4096	33' 33"	0.889 / 0.952	90%
Island (Figure 7)	0	1024	34' 30"	0.992	87%
	1	1024	42' 35"	0.992 / 0.996	81%
	3	1024	46' 05"	0.992 / 0.875	80%
	5	1024	47' 20"	0.992 / 0.864	79%
Flowers (Figure 8)	0	256	03' 19"	0.901	15%
	1	256	03' 28"	0.899 / 0.838	12%

Table 1: *Performance results for the render of N shadow layers. The overhead between $N = 0$ and $N = 1$ is mainly due to additional intersection tests. As N increases, the various images are less converged compared to the reference, and their handling takes processing power. The SSIM is given for the main layer / the shadow layer with minimum similarity to the reference.*

Scene	N	Time	SPP	$\overline{\text{Var}}$	Var	Time	$\overline{\text{SPP}}$
Teaser (Fig. 1)	0	15'	2048	0.003	0.01	18' 24"	2321
	1	15'	1824	0.004	0.01	20' 08"	2318
	2	15'	1728	0.004	0.01	21' 46"	2360
	3	15'	1696	0.004	0.01	22' 14"	2371
Dragon (Fig. 6)	0	30'	9856	0.153	0.1	34' 58"	8122
	1	30'	8256	0.153	0.1	41' 44"	8139
Island (Fig. 7)	0	30'	960	0.007	0.01	39' 36"	1277
	1	30'	768	0.007	0.01	51' 11"	1289
	3	30'	704	0.008	0.01	54' 16"	1310
	5	30'	704	0.008	0.01	55' 56"	1317
Flowers (Fig. 8)	0	10'	832	0.271	0.5	4' 01"	291
	1	10'	784	0.271	0.5	4' 15"	291

Table 2: *The middle column shows the number of samples contributing to all layers, and the mean variance of the main layer for equal-time runs. The right column shows the computation time for a target pixel variance in the main layer, and the average sampling.*

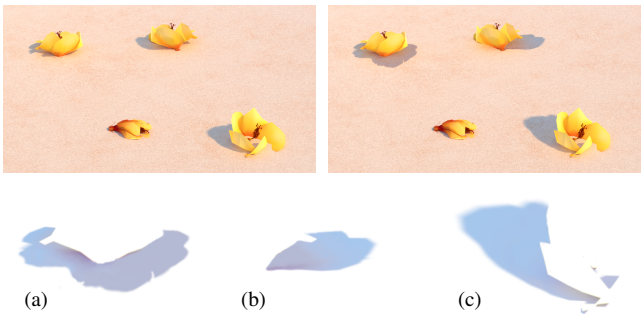


Figure 8: *Transforming the shape of shadows using the shadow ratio $I/(I+S)$. Top left: main layer. Top right: the result is coherent with the fine-grained sand texture. Bottom: the ratio was edited by (a) painting directly, (b) mirroring and inpainting the occluded part, (c) using a cage deformation.*

While fixing the sampling budget and activating the rendering of shadow layers corresponds to a typical use case, we study more in-depth the interaction of sample count, rendering time and pixel variance. To do so, we fix time or variance in the main layer to a target and compare our implementation to standard path tracing; results are found in Table 2. For fixed variance, pixels can adaptively use up to 4 times the initial sampling budget of Table 1.

In the Teaser, fixed-time performance is comparable to that of Table 1. However with a target variance, both integrators consume additional samples in regions with difficult light transport such as the caustics under the glass sphere, where our algorithm also picks up indirect shadows. The Dragon scene contains only indirect lighting and shadows that cause very slow convergence for both algorithms, as they rely on path tracing. Whereas the Flowers view from Figure 8 contains mostly direct shadows that have little performance impact, the full Island scene is the most challenging: across the deep foliages, many intersection tests are performed to recover missing light from the sources on the other side.

6. Limitations and Extensions

When light is occluded by more than one caster such as in Figure 9, the resulting shadow cannot be assigned to only one of them, which creates inconsistencies. One solution is to export a shadow layer with both casters considered as one; this is the case in Figure 1 between the mirror and the two background spheres. However, tracking interactions between any two casters among N total yields $\binom{N}{2} = N(N-1)/2$ additional layers. Likewise, accounting for all possible interactions between N casters would involve 2^N layers. Methods that give more precise control over light transport [SMVP17, SNM*13, OKP*08] do not encounter this limitation. In the future, we would like to design an automated way to correctly recover shadows created by multiple casters.

Because we heavily rely on the notion of surface to define an object and derive shadow layers from it, our work does not apply to participating media. Generalizing our definition to participating media would give us further insights on what is sufficient to characterizes shadows; we believe that the black body behavior can be reproduced by modifying the absorption coefficient.

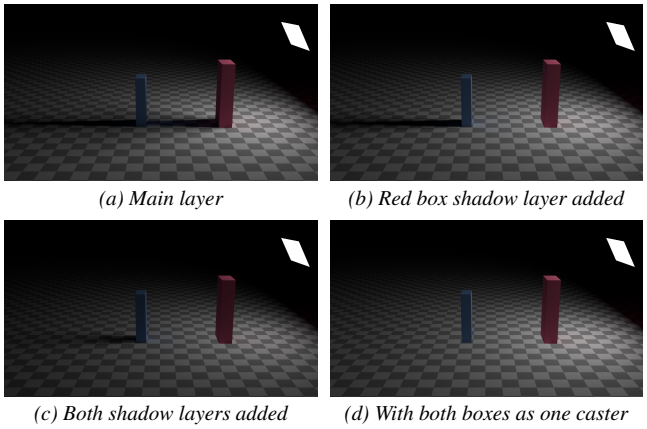
While we implemented our approach using path tracing, we give hints on how to adapt the computation of shadow layers to other rendering algorithms. The formulation as the difference between two altered renders works out of the box, but for a single pass rendering, one needs to adapt two parts of the algorithm: (1) ray propagation and (2) path integration. For instance in bidirectional path tracing, (1) implies altering propagation from the camera and light, as devised for path tracing. (2) means that sub-paths connection is given the possibility to ignore an occluding caster, much like direct lighting; however, only full paths exhibiting a coherent propagation should be formed, *i.e.* a camera sub-path that skipped caster $\mathbf{C}_i$ cannot be connected to a light sub-path that scattered on $\mathbf{C}_i$.

The content of a shadow layer remains coherent in the presence of animations, but artistic edits must still be keyframed to follow the action. Taking time into account would be a natural extension to our work: ideally, the exported layers should include a differential information expressing how shadows move and deform in the adjacent frames of an animation, and support topology changes.

7. Conclusion

We proposed a coherent definition of an object's shadow in a global illumination context, and a characterization of this definition in the path space formulation of light transport. This allowed us to devise an efficient method to render it while reusing some of the computation performed by a path tracing algorithm. Given a fixed sampling budget, it overcomes the limitations of current renderers and supports the extraction of both direct and indirect shadows with a predictable computational overhead factor between 1.1 and 1.3.

Along with these performance improvements, our solution provides the user with full artistic freedom when extracting and editing global shadows. The light source, caster and catcher interacting to create shadow can be freely specified, and their definition naturally follows the semantic of objects in the scene.



(a) Main layer *(b) Red box shadow layer added*

(c) Both shadow layers added *(d) With both boxes as one caster*

Figure 9: *Artifact due to shadow interaction. Adding the shadow layer of the red box creates a new shadow for the blue box. Hence when adding both shadow layers, our approach shows a residual shadow. One solution is to consider both objects as the same caster.*

Acknowledgements

This work benefited from the support of project CaLiTrOp ANR-16-CE33-0026 of the French National Research Agency (ANR). The authors want to thank Nicolas Mellado, Loïc Barthe and the anonymous reviewers for their valuable comments, and the artists and technical experts who motivated this work: Élodie Fraysse, Benjamin Legros, Cyril Corvazier, and Philippe Llerena.
The Cornell Box was adapted to pbrt by Benedikt Bitterli; the Dragon comes from the Stanford 3D Scanning Repository. Moana Island Scene made available by Walt Disney Animation Studios.

Appendix A: Path Integral Formulation

We rewrite here some useful derivations for clarity [Vea97]. From the usual area measure A on the set of surfaces $\mathcal{M}$, the area-product measure μ is defined for a path $\bar{x} = x_0\, x_1 \ldots x_k$ as

$$d\mu(x_0\, x_1 \ldots x_k) = dA(x_0) \ldots dA(x_k)$$

This implies that if $\mathcal{M}$ does not change, neither do A nor μ. The measurement contribution function f_j is the product

$$\begin{aligned} f_j(\bar{x}) = {} & L_e(x_0 \to x_1) G(x_0 \leftrightarrow x_1) \\ & \cdot \prod_{i=1}^{k-1} \varphi(x_{i-1} \to x_i \to x_{i+1}) G(x_i \leftrightarrow x_{i+1}) \\ & \cdot W_e^j(x_{k-1} \to x_k) \end{aligned} \tag{3}$$

where $L_e(x_0 \to x_1)$ is the radiance emitted at x_0 towards x_1, and $W_e^j(x_{k-1} \to x_k)$ the importance leaving x_k towards x_{k-1}. We are interested in the middle term, which is the product for all three-point configurations $x_{i-1} \to x_i \to x_{i+1}$ of the BSDF φ and the geometric term G that accounts for mutual visibility and solid angles.

To obtain $f_{T\,j}$, the original BSDF $\varphi(x_{i-1} \to x_i \to x_{i+1})$ is replaced by $\delta(x_{i-1} \to x_i \to x_{i+1})$ whenever $x_i \in \mathcal{M}_\mathbf{C}$ in Equation (3). In turn, $\delta(x_{i-1} \to x_i \to x_{i+1})$ is equal to 1 when x_{i-1}, x_i and x_{i+1} are colinear and arranged in this order, and null otherwise. For $f_{B\,j}$, the original BSDF is replaced by the null function whenever $x_i \in \mathcal{M}_\mathbf{C}$, implying that the whole product becomes null.

References

[BCRA11] BOUSSEAU A., CHAPOULIE E., RAMAMOORTHI R., AGRAWALA M.: Optimizing environment maps for material depiction. *Computer Graphics Forum 30*, 4 (2011), 1171–1180. doi:10.1111/j.1467-8659.2011.01975.x. 2

[BSS04] BOVIK A. C., SHEIKH H. R., SIMONCELLI E. P.: Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing 13*, 4 (April 2004), 600–612. doi:10.1109/TIP.2003.819861. 7

[DCFR07] DECORO C., COLE F., FINKELSTEIN A., RUSINKIEWICZ S.: Stylized shadows. In *Proc. 5th International Symposium on Non-photorealistic Animation and Rendering* (New York, NY, USA, 2007), NPAR '07, ACM, pp. 77–83. doi:10.1145/1274871.1274884. 2

[Gri16] GRITZ L.: OSL, 2016. Accessed 2019-04-03. URL: https://github.com/imageworks/OpenShadingLanguage/wiki/OSL-Light-Path-Expressions. 4

[HBM*14] HECHER M., BERNHARD M., MATTAUSCH O., SCHERZER D., WIMMER M.: A comparative perceptual study of soft-shadow algorithms. *ACM Trans. Appl. Percept. 11*, 2 (July 2014). doi:10.1145/2620029. 1

[Hec90] HECKBERT P. S.: Adaptive radiosity textures for bidirectional ray tracing. *SIGGRAPH Comput. Graph. 24*, 4 (Sept. 1990), 145–154. doi:10.1145/97880.97895. 4

[KPD10] KERR W. B., PELLACINI F., DENNING J. D.: Bendylights: Artistic control of direct illumination by curving light rays. *Computer Graphics Forum 29*, 4 (2010), 1451–1459. doi:10.1111/j.1467-8659.2010.01742.x. 2

[MIW13] MATTAUSCH O., IGARASHI T., WIMMER M.: Freeform shadow boundary editing. *Computer Graphics Forum 32*, 2pt2 (2013), 175–184. doi:10.1111/cgf.12037. 2

[NJS*11] NOWROUZEZAHRAI D., JOHNSON J., SELLE A., LACEWELL D., KASCHALK M., JAROSZ W.: A programmable system for artistic volumetric lighting. *ACM TOG 30*, 4 (July 2011), 29:1–29:8. doi:10.1145/2010324.1964924. 2

[OKP*08] OBERT J., KŘIVÁNEK J., PELLACINI F., SÝKORA D., PATTANAIK S.: icheat: A representation for artistic control of indirect cinematic lighting. *Computer Graphics Forum 27*, 4 (2008), 1217–1223. doi:10.1111/j.1467-8659.2008.01260.x. 2, 8

[OPP10] OBERT J., PELLACINI F., PATTANAIK S.: Visibility editing for all-frequency shadow design. *Computer Graphics Forum 29*, 4 (2010), 1441–1449. doi:10.1111/j.1467-8659.2010.01741.x. 2, 6

[PBMF07] PELLACINI F., BATTAGLIA F., MORLEY R. K., FINKELSTEIN A.: Lighting with paint. *ACM TOG 26*, 2 (June 2007). doi:10.1145/1243980.1243983. 2

[PF92] POULIN P., FOURNIER A.: Lights from highlights and shadows. In *Proc. Symposium on Interactive 3D Graphics* (New York, NY, USA, 1992), I3D '92, ACM, pp. 31–38. doi:10.1145/147156.147160. 2

[PF95] POULIN P., FOURNIER A.: Painting surface characteristics. In *Rendering Techniques '95* (1995), Hanrahan P. M., Purgathofer W., (Eds.), Springer Vienna, pp. 160–169. doi:10.1007/978-3-7091-9430-0_16. 2

[PRJ97] POULIN P., RATIB K., JACQUES M.: Sketching shadows and highlights to position lights. In *Proc. Computer Graphics International* (June 1997), pp. 56–63. doi:10.1109/CGI.1997.601272. 2

[PTG02] PELLACINI F., TOLE P., GREENBERG D. P.: A user interface for interactive cinematic shadow design. *ACM TOG 21*, 3 (July 2002), 563–566. doi:10.1145/566654.566617. 2

[ROTS09] RITSCHEL T., OKABE M., THORMÄHLEN T., SEIDEL H.-P.: Interactive reflection editing. *ACM TOG 28*, 5 (Dec. 2009), 129:1–129:7. doi:10.1145/1618452.1618475. 2

[RTD*10] RITSCHEL T., THORMÄHLEN T., DACHSBACHER C., KAUTZ J., SEIDEL H.-P.: Interactive on-surface signal deformation. *ACM TOG 29*, 4 (July 2010), 36:1–36:8. doi:10.1145/1778765.1778773. 2

[SMVP17] SUBILEAU T., MELLADO N., VANDERHAEGHE D., PAULIN M.: Rayportals: A light transport editing framework. *Vis. Comput. 33*, 2 (Feb. 2017), 129–138. doi:10.1007/s00371-015-1163-2. 2, 8

[SNM*13] SCHMIDT T.-W., NOVÁK J., MENG J., KAPLANYAN A. S., REINER T., NOWROUZEZAHRAI D., DACHSBACHER C.: Path-space manipulation of physically-based light transport. *ACM TOG 32*, 4 (July 2013), 129:1–129:11. doi:10.1145/2461912.2461980. 2, 8

[SSMK05] SATTLER M., SARLETTE R., MÜCKEN T., KLEIN R.: Exploitation of human shadow perception for fast shadow rendering. In *Proc. 2nd symposium on Applied perception in graphics and visualization* (2005), APGV05, Association for Computing Machinery, p. 131–134. doi:10.1145/1080402.1080426. 1

[Vea97] VEACH E.: *Robust Monte Carlo Methods for Light Transport Simulation*. PhD thesis, Stanford University, Stanford, CA, USA, 1997. AAI9837162. 3, 9