Path-Space Manipulation of Physically-Based Light Transport

Thorsten-Walther Schmidt [1] Jan Novák [1] Johannes Meng [1] Anton S. Kaplanyan [1] Tim Reiner [1]
Derek Nowrouzezahrai [2] Carsten Dachsbacher [1]

[1] Karlsruhe Institute of Technology [2] Université de Montréal

Figure 1: *Transport manipulation in the* GARAGE *scene. Before/after close-ups (right): removing indirect highlights caused by the car, re-directing sunlight after it refracts through the windows, moving and rotating a glossy interreflection, and altering the mirror reflection.*

Abstract

Industry-quality content creation relies on tools for lighting artists to quickly prototype, iterate, and refine final renders. As industry-leading studios quickly adopt physically-based rendering (PBR) across their art generation pipelines, many existing tools have become unsuitable as they address only simple effects without considering underlying PBR concepts and constraints. We present a novel light transport manipulation technique that operates directly on path-space solutions of the rendering equation. We expose intuitive direct and indirect manipulation approaches to edit complex effects such as (multi-refracted) caustics, diffuse and glossy indirect bounces, and direct / indirect shadows. With our sketch- and object-space selection, all built atop a parameterized regular expression engine, artists can search and isolate shading effects to inspect and edit. We classify and filter paths on the fly and visualize the selected transport phenomena. We survey artists who used our tool to manipulate complex phenomena on both static and animated scenes.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Raytracing; I.3.6 [Computer Graphics]: Methodology and Techniques—Interaction techniques

Keywords: global illumination, artistic light transport editing

Links: DL PDF WEB VIDEO

ACM Reference Format
Schmidt, T., Novák, J., Meng, J., Kaplanyan, A., Reiner, T., Nowrouzezahrai, D., Dachsbacher, C. 2013. Path-Space Manipulation of Physically-Based Light Transport. ACM Trans. Graph. 32, 4, Article 129 (July 2013), 8 pages. DOI = 10.1145/2461912.2461980 http://doi.acm.org/10.1145/2461912.2461980.

DOI: http://doi.acm.org/10.1145/2461912.2461980

1 Introduction

Physically-based rendering (PBR) systems simulate, store, and operate on the same radiometric quantities and operators (e.g. NVIDIA's MentalRay, Radiance [1998], PBRT [2010], Arnold by Solid Angle Inc.). The availability of efficient PBR systems with progressive pre-visualization has promoted rapid adoption of PBR standards in the feature-film and gaming industries [McAuley et al. 2012; Křivánek et al. 2010]. As such, artists are becoming increasingly familiar with technical PBR concepts.

Increasing artists' productivity by minimizing iteration quantity and time depends on the flexibility of their lighting tools more than on the underlying PBR system. Many existing tools either target non-PBR systems or consider only very specific PBR effects (Sec. 2). We present an artistic lighting tool, built on physically-based global illumination solutions, that enables rapid and intuitive selection and manipulation of light transport, including effects that result from complex light paths (see Fig. 1). We integrate atop industrial digital content creation (DCC) tools and support various PBR algorithms.

Our approach interactively clusters light paths according to user selected feature(s) and provides a visualization of clustered paths (Fig. 2). This instant feedback links an artist's selection to its underlying PBR constructs. To cope with the complexity of light transport, we complement selection with subpath ranking and filtering based on path type and/or on path–object interactions.

In contrast to previous work, our manipulation operates entirely on path space, rather than targeting specific shading phenomena. We present two complementary editing concepts, each of which permits light transport editing from a different perspective (Fig. 3): *path retargeting* operates *directly* on paths, allowing the user to select and transform grouped paths; *path-proxy linking*, on the other hand, *indirectly* edits the path space according to edits of the scene. These two approaches enable editing of complex shading effects such as multiply-refracted caustics, indirect lighting, reflection, and shadows. They are purposefully redundant, as some editing tasks can be completed with both path retargeting or path-proxy linking, although, depending on the task, one approach may be more intuitive or efficient.

Considering the entire path space poses several technical challenges, as we make no prior assumptions on the form of transport. We discuss how to render manipulated path spaces with bidirectional, importance-sampled global illumination (GI) algorithms.

Our paper makes the following contributions:

Selection-related mechanisms:

- path selection on surfaces with stroke- and region-sketching,
- automatic clustering, classification and ranking of transport effects based on light paths and their neighborhoods.

Manipulation-related mechanisms:

- two complementary editing concepts for artistic light transport manipulation: *path retargeting* and *path-proxy linking*,
- consistent rendering of edited transport with PBR algorithms,
- a more general path-space editing solution, in which many existing techniques become special cases of our approach.

2 Previous Work

We review PBR editing approaches, including selection, manipulation, and visualization of light transport. Our survey focuses exclusively on PBR techniques; NPR editing is outside our scope.

Light Transport Manipulation. We begin by considering editing tools for direct illumination effects (most predominantly shadows), before addressing prior work on editing more complex effects.

Poulin and Fournier [1992] use user-modified highlights and shadows to infer light positions in a scene. Several works similarly allow users to drag, paint, and re-color shadows, automatically adjusting light parameters to satisfy the requested edits [Barzel 1997; Pellacini et al. 2002; Pellacini et al. 2007]. Kerr et al.'s [2010] method directs the path of direct illumination along curvy volumetric frusta. *EnvyLight* [Pellacini 2010] allows users to sketch image-space features to select and edit direct illumination from natural lighting; our system couples sketch-based selection with path-space filtering in order to facilitate feature selection. Recent techniques employ appearance-based interfaces for all-frequency direct lighting manipulation with inverse shading [Okabe et al. 2007] or precomputed editable visibility representations [Obert et al. 2010]. Lee et al. [2006] edit shading independently per object. We also enable selection and filtering of path space according to scene object IDs. This allows users to constrain edits to specific objects, as in Lee et al.'s work, but also to isolate effects *between* a subset of objects.

Ritschel et al. [2009] let users specify constraints on mirror surfaces to infer modified reflections. Nowrouzezahrai et al. [2011] designed a tool for artistic manipulation of volumetric lighting which, like ours, is integrated into existing industrial pipelines; this facilitated and promoted its adoption in the industry. Obert et al. [2008] use a painting interface to edit the intensity and color of indirect light, all while enforcing physical constraints on the resulting appearance. Tabellion and Lamorlette [2004] use shader falloff-function editing on the hue of indirect color bleeding effects.

Interactive On-Surface Signal Deformation (OSSD) [Ritschel et al. 2010] is a flexible tool for manipulating shadows, caustics and indirect illumination with a click-and-drag interface. OSSD requires surface parameterizations (computed on the fly), nearly precluding its applicability to animated/deforming scenes and restricting edits to signals *on surfaces*. In contrast, our approach manipulates path-space segments, naturally supporting complex transport effects and animations without the need for any surface parameterization.

While powerful, all prior work either support limited types of transport effects or rely on specialized rendering engines or precomputation, impacting their adoption in large-scale production pipelines.

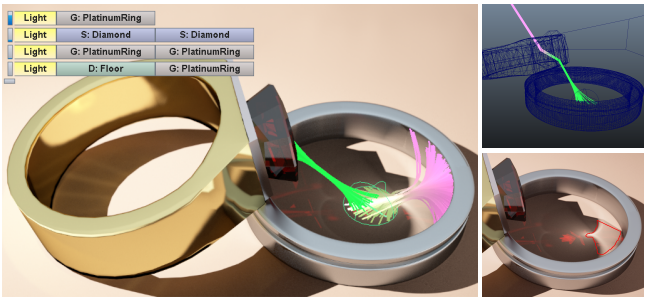

Figure 2: *Left: our interactive GI preview suggests a list of path classifications (using our extended notation) ranked by the contribution to the selected region; paths matching the top two classifications are visualized as bundles. Right: a caustic multiply refracted by the gemstone; sketch-based selection of the ring's caustic.*

Visualization and User Interfaces (UIs). We overview recent developments in visualization for goal-based light manipulation alongside interaction techniques used in the aforementioned works. *BendyLights* [Kerr et al. 2010] visualizes a user-deformable lighting volume used to control direct illumination. Reflection editing [Ritschel et al. 2009] and OSSD [Ritschel et al. 2010] both use custom UIs to specify spatial and directional constraints during editing. We also use a customized UI to intuitively expose transport manipulation operations for directional and spatial selection/editing. We do so while respecting the UI and interaction concepts of the DCC we chose to build our system atop. This shrinks the learning curve and builds on artists' training and skills.

Prior art used sketch-, click-and-drag, and paint-based editing concepts. Kerr and Pellacini [2009] show that, while paint interfaces are suitable in some cases, *direct* and *indirect* manipulation are better suited to typical editing operations. As such, we devise direct and indirect interaction techniques to manipulate light transport.

While light transport can be represented mathematically in many forms, few works explicitly use *visually* informative representations of transport. Reiner et al. [2012] are an exception, developing and validating (via user survey) several methods of visualizing spatially- and angularly-varying radiometric quantities. Our work is influenced by their approach, however we extend these inspection approaches to more complex path-space analysis, as well as exposing several approaches to now *manipulate* the visualized quantities.

3 Goals and Overview

We will identify the goals of our tool and outline how our method integrates into standard modeling and lighting workflows. We target six design requirements, while simultaneously balancing between artistic controllability and plausibility:

- **WYSIWYG Editing:** an interactive preview of the progressive GI solution is needed to better inform artists at edit time.
- **Consistency:** edits must respect the PBR behavior by default, e.g. moving a shadow must also affect indirect lighting (unless explicitly disabled). Layered edits must also be supported.
- **Animation:** keyframing of editing operations should be supported, with edits extending naturally to animated scenes.
- **Usability:** edits should be easy to learn and intuitive, building on widely adopted user interfaces and workflows.
- **Generality:** all light transport manipulations should operate in a unified manner, e.g. caustic and diffuse indirect light selection and editing should be exposed under the same UI.
- **Rendering:** the edited light transport solution should be computable using robust (ray tracing-based) GI methods.

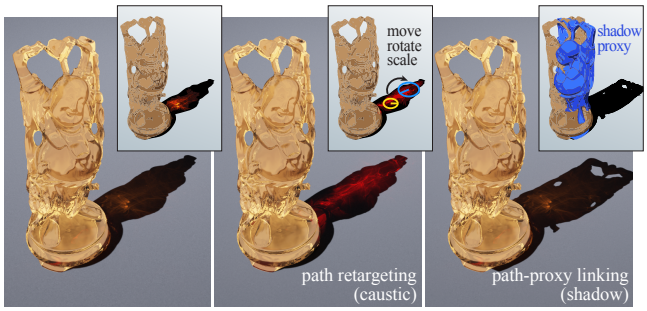

Figure 3: *Left to right: original render; using path retargeting to displace light paths forming the caustic; path-proxy linking creates a proxy instance of the Buddha that affects only the shadow.*

We implement our method as a plug-in for a professional DCC system and employ stochastic progressive photon mapping (SPPM) [Hachisuka and Jensen 2009] as our rendering backend during editing. SPPM balances quality with immediate interactive and progressive feedback and, as with any ray tracing-based backend, allows us to trivially gather path sample information. We also overlay a visualization of light transport paths (Sec. 4.3 and Fig. 2) to facilitate the inspection and editing of selected phenomena.

A typical editing session starts with selection and filtering of a transport effect. The user defines a region of interest by sketching, placing a bounding volume around the target area, or selecting objects that interact with the transport. Our system then automatically ranks incident light paths within the selection by their contribution and the surrounding transport. At any point, the user may override our tool's suggested rankings and either manually select from an automatically generated list of transport paths or specify paths using our parameterized regular expression notation (Sec. 4.1).

We present two complementary editing concepts, each of which manipulates transport according to a different rationale (Fig. 3): *path retargeting* operates on the space of transport paths, allowing the user to transform (e.g. translate, rotate, or scale) path vertices; *path-proxy linking* indirectly edits light transport for a particular transport phenomenon according to external scene adjustments, e.g. artists can "displace" a shadow-casting object and our tool automatically creates an "invisible" proxy that only affects its shadow(s).

4 Filtering and Selection of Light Transport

Targeted and deliberate transport editing requires a precise and intuitive way to select and filter light transport effects in the scene. Internally, we employ an extension of Heckbert's path notation [1990] that includes object IDs, providing more detailed differentiation of individual path interactions (Sec. 4.1). Expert users can select transport according to this notation, but we also expose more intuitive semi-automatic selection techniques (Sec. 4.2). In every case, our system analyzes and ranks the most significant phenomena in a selected region, all while visualizing the selected light transport according to our informative visualization scheme (Sec. 4.3).

4.1 Extended Path Classification

We overview our parameterized regular expression syntax for path classification. We distinguish between diffuse (D), glossy (G), and specular (S) interactions. We additionally classify interactions as reflections (superscript $\square^R$) or transmissions ($\square^T$), and store the ID of the object at the interaction. An important addition compared to other variants of Heckbert's notation is that we define a token X_P corresponding to an *arbitrary* surface interaction (D, G, or S) that (optionally) lies in a selected sub-region of $P \subseteq \mathbb{R}^3$.

This extended notation is powerful enough to classify all surface-based editable transport phenomena. For example, $L_i S_j^T S_j^T X_P E$ corresponds to a visible transmissive caustic through object #j, lit by light source #i, where the final interaction (before the eye vertex) lies in a (user specified) region P. To simplify user interaction, these expressions can easily be converted into textual descriptions, e.g. "caustic through [name of object]".

4.2 Intelligent Selection

We believe an intuitive way of selecting a shading effect is to define a region on a surface where it is visible (i.e. $L(D|G|S)^* X_P E$). As such, we allow users to sketch or place bounding volumes (Fig. 2) to delimit regions of interest. We then collect transport paths whose vertices (i.e. photons) fall into this region, and compute a flux-weighted histogram according to the paths' classifications. We subsequently offer two ways of ranking these paths (both exposed in the UI): *contribution ranking* orders histogram bins by decreasing flux and presents the first entries of this sorted list to the artist (the first element is chosen as default), while *discrimination ranking* compares the transport in the selected region with the transport in its surrounding neighborhood, using the rationale that something about the selected light transport is special or notable compared to its surrounding. To this end, we compute a second, outer histogram in an enlarged (50% by default) region around the original region of interest and compare it to the first histogram: we order path classifications by their (decreasing) relative flux in the histograms, i.e., for every classification, we divide its accumulated flux in the outer histogram by that in the inner one (as we use convex bounding volumes, paths in the inner region are guaranteed to be included in the outer region as well). We similarly select and manipulate eye subpaths, e.g. a mirror reflection $X_P SE$, by shooting importons (particles carrying importance) from the camera.

Light transport effects can also be selected according to the order in which scene objects interact in the desired path, permitting a more visual approach to building the desired path specification. Here, we are limited to contribution ranking, since no region is specified.

Many photons must be collected (in the region of interest) to form meaningful statistics in complex scenes with many objects; we accumulate photons over many frames. Alternatively, an importance-driven metric can be used during accumulation. In complex scenes, histogram size can be reduced by merging similar bins according to high-level descriptors (as in Ritschel et al. [2010]).

4.3 Light Path Visualization

We detail our novel path visualization method used to provide immediate feedback to the artist after path selection. Naively rendering path segments (i.e. with line segments) results in visual clutter and we are motivated by work in the information visualization community on *force-directed edge bundling* [Holten and van Wijk 2009]. This approach performs (expensive) physical simulations to improve the visualization of complex graphs by drawing contracted edges. We present a variant more suitable to our problem that, in contrast, is efficient to compute and readily generalizes to 3D paths.

After light (photons) or eye vertices (importons) have been selected/filtered, we cluster their path segments into bundles as follows: two segments are bundled if they are equidistant from the X token in terms of interactions, and their interactions match in our path notation. The number of visualized segments is user-definable.

Next, we compute the medial axis of each bundle. Every segment is then rendered as a quadratic curve contracted towards the medial axis, while its endpoints remain at the original photon positions; the strength of the contraction is defined by a global user-controlled parameter. The user can also choose to contract the

curves across segments separated only by interaction types. This more effectively distinguishes between different sources of illumination in more complex scenarios, especially when the region of interest is populated only by photons after several indirect bounces from the light. We render the curves with colors coded according to the interacting objects and lit using Bank's model [1994].

User Interface. Fig. 2 illustrates our selection tools and light path visualization. We use standard DCC gizmos (widgets) and overlay additional UI elements atop modeling and GI rendering preview viewports. All options and parameter input fields are integrated into the DCC's standard menus and dialogs (see accompanying video).

5 Manipulation of Light Transport

We now focus on our interactive manipulation tools. As mentioned earlier, our design goals include consistency and generality: the manipulated shading should remain as physically plausible as possible, without restricting artistic freedom. To this end, we start from a physically-based GI solution and always provide interactive feedback. We begin by detailing our *path retargeting* (explicitly altering path segments), and then describe *path-proxy linking*, which enables manipulation by linking transformed scene objects to appropriate light transport phenomena. Both of these strategies additionally support brightness scaling/offsetting and hue editing (in the spirit of [Obert et al. 2008]). In Sec. 5.3 we describe how our edits generalize to animated scenes, as well as multiple layered edits.

5.1 Path Retargeting

Path retargeting provides focused control over distinct lighting features, e.g. dragging a caustic or moving a reflection. The underlying concept is to choose a light (e.g. diffuse indirect light LD^RX_P) or an eye subpath (e.g. mirror reflection X_PS^RE), and then retarget (move) the subpath's endpoints. Retargeting alters path segments and thus implicitly affects secondary effects, such as indirect occlusion and interreflection. The following discussion assumes manipulation of a light path; eye paths are handled analogously.

Retargeting operates as follows: first, the transport phenomenon is selected (Sec. 4.2), defining the light subpaths in our extended path notation that will be affected during retargeting. If the user chose to specify a region of interest using a UI gizmo, then its center will act as a *source anchor*, serving as the origin of the retargeting transformation. The transformation is an affine mapping defined by placing a *target gizmo* (Fig. 2) in the scene. Note that the user can choose either to retarget only path vertices within the region of interest, or all vertices matching the classification filter. We illustrate the usefulness of this latter option, for example, when a glass sphere that is causing a caustic is retargeted. If the sphere is moved, the caustic may migrate outside the source region. Only when matching the path classification will the caustic also be retargeted. Both source and target regions are independent of the object causing the phenomenon, however their transformations can also be keyframed, and optionally specified relative to the object's location.

We give an example of retargeting in the context of photons: after selecting a caustic path $L_iS_j^TS_j^T(X|X_P)$, displacing the target node translates only the path vertices (photons) in the selected region P whose path matches this filter. Our preview provides interactive feedback to the user, and any new path can of course undergo further manipulations. Note that we move path vertices but maintain the energy throughput of the original path; for glossy and specular BSDFs, the throughput of manipulated paths is often zero. Such a manipulation can also be seen as a local tangent space transformation at the interaction before X such that the throughput (if recomputed) equals the original path throughout (see App. A).

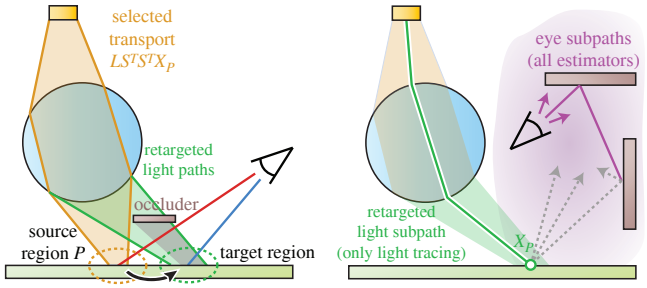

Figure 4: *Left: Retargeted paths (shown here for light subpaths) interact with scene geometry. If we apply the inverse transformation to the blue camera ray (yielding the red ray) and continue tracing the path, we would miss the occlusion. Right: In bidirectional path tracing, we only allow manipulated path segments to be created with unidirectional sampling. Here, the light path must be constructed with light tracing up to X_P and only afterwards can we combine estimators using multiple importance sampling.*

5.1.1 Integration with Light Transport Algorithms

Retargeting is defined to operate in the direction in which either light or importance propagate and can thus be trivially applied to light or eye subpaths, respectively. However, a fixed-length path can be constructed from several pairings of light and eye subpaths, each of different length. Therefore we have to ensure that all potential bidirectional Monte Carlo estimators consistently interpret the manipulations. Unfortunately, applying a manipulator in the opposite direction of its definition is non-trivial (Fig. 4, left). We proceed to describe our practical solution to this problem that ensures consistency and easily integrates into SPPM or bidirectional path tracing (BDPT) [Veach and Guibas 1994; Lafortune and Willems 1993] at little extra computation cost. In App. A we discuss a more robust and theoretical approach that might perform better in some cases, but its integration into existing renderers is non-trivial.

We again provide an example for manipulated light paths, noting that eye path manipulation behaves analogously. BDPT constructs light and eye subpaths and then connects them. We can readily apply retargeting to light paths, but inconsistencies may occur when we trace the same path from the camera (i.e. when we use an estimator that attempts to construct the manipulated segment as an eye subpath; Fig. 4, left). Since it is not clear how to properly retarget eye paths, we must ensure that we employ only Monte Carlo estimators that construct the manipulated segment(s) in the same direction as the manipulator's operation (Fig. 4, right).

For example, if we displace a caustic with a manipulator, we must ensure that only the (light-traced) unidirectional estimators up to the manipulated vertex are used. Once we reach the manipulated vertex, we have a subpath that can connect to the eye using all estimators. This ensures consistent results but reduces the set of available estimators for some subpaths, potentially locally decreasing the efficiency of multiple importance sampling in BDPT.

SPPM is a special case where each path can be constructed using only a single estimator, avoiding any inconsistencies. For instance, caustics are manipulated by transforming light paths, affecting only photon tracing, and then density estimation connects to the eye paths. Analogously, reflection editing is an example of manipulating eye paths connected to light paths using density estimation.

5.1.2 Increasing Artistic Flexibility

Path retargeting can also be generalized: in our prototype we bend path segments (see Fig. 5 for an example) which generalizes Bendy-

Figure 5: *Bending paths to form "BendyCaustics" off the mirror.*

Lights [Kerr et al. 2010] to arbitrary transport. OSSD [Ritschel et al. 2010] "displaces" surface shading effects, however, topology constraints limit editing flexibility (e.g. caustics cannot move from an object to another disconnected one). In contrast, we work in path space where edits transparently transfer across objects (surface signals are just slices of the light field), including objects undergoing topological changes by animation/deformation/fracturing. We also automatically update secondary illumination effects, e.g. indirect shadows, to maintain PBR consistency.

5.2 Path-Proxy Linking

Our second manipulation concept is motivated by the observation that certain shading edits can be best conceived and achieved through manipulation of the scene elements, instead of explicitly editing light paths. For example, an object's shadow silhouette can be intuitively edited by rotating or scaling the object. The idea of path-proxy linking[1] is to offer the possibility to specify different object transformations for individual components of the light transport, and can thus be seen as a generalized light linking technique (but not a super-set). Path-proxy linking can also be used for *subtractive* shading phenomena, such as shadows, which are best described by a lack of light transport rather than with a type of path; such edits would not be possible using path retargeting[2]. We also believe that manipulating a shadow using path-proxy linking is more intuitive: users select direct light on a surface, pick the shadow casting object, and are then presented with a proxy object used only for shadow computation (Fig. 3, right) and can be transformed using standard gizmos.

As usual, an edit always starts by selecting and filtering the targeted transport phenomenon. After potentially using the visualization to help understand which scene objects are involved in the generation of the selected shading effect, the user selects one or more objects influencing the selected transport to be manipulated. Our method automatically creates a proxy object (virtual instance), for every selected object, that can undergo affine transformations while affecting only light transport of the selection filter. Every proxy is also linked to the selection gizmo, and once the selection is modified, the manipulated transport updates as well. We also permit multiple proxies per object, e.g. to separately manipulate shadows and indirect illumination. The geometry itself is not duplicated; when the object geometry is modified, all proxies are updated accordingly. Note that every proxy has a unique object ID, which is required to distinguish the different light paths during GI computation.

5.2.1 Efficient Integration into Global Illumination Systems

Efficient path-proxy linking requires two modest changes to acceleration structures and GI algorithms, which we detail below.

[1]The term "*path-proxy linking*" was chosen to explicitly indicate that light paths become linked to geometric proxies in the scene.

[2]Photon mapping can simulate shadows with "shadow photons" carrying negative energy, which would facilitate path retargeting of subtractive effects. We avoid this to remain compatible with more GI methods.

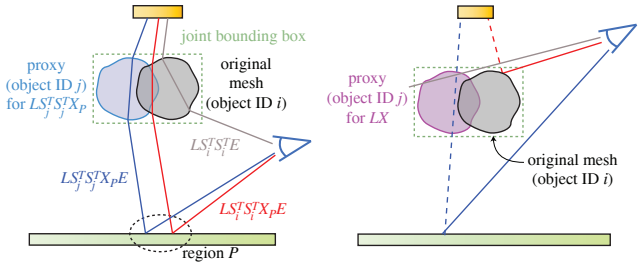

Figure 6: *Left: path-proxy linking for refractive caustics; paths are discarded if they match the filter but have not been created with the proper proxy. Right: next event estimation for eye subpaths is also discarded when the wrong instance is used; primary rays are always tested against the original mesh.*

Accelerator. We use a bounding volume hierarchy (BVH) with two hierarchical levels. First we build a BVH over the polygons of each object in the scene. These BVHs form the lower level of the hierarchy. Then we compute the *joint bounding box* of each object and all its proxies. Finally, we construct the upper level of the hierarchy as a BVH over the joint bounding boxes, where each leaf contains the path classifications and affine transformations for the proxies of the corresponding object. This two-level hierarchy enables efficient affine transformations, as only the upper level needs to be updated when manipulating the proxies.

Global Illumination Computation. At the core of every ray tracing based GI method, paths are constructed between light sources and sensor points. No special treatment is required for paths that interact only with non-manipulated objects. For a path interacting with a manipulated object, consider the example in Fig. 6 (left): the user selects a refractive caustic $LS_i^TS_i^TX_P$ and creates a proxy for object i with the new ID j. All refractive caustic paths will now be computed using only the proxy, not the original object (splitting path space into disjoint sub-spaces). To achieve this, we need to slightly modify the tracing of paths. When a segment intersects a joint bounding box containing the original object and N proxies, the path should interact with only one of the $N+1$ representations. Since we do not yet know the complete path's classification, we probabilistically pick one representation and continue with construction. Once we can determine whether the path's classification matches the linked filter, we re-weight the path's contribution by $N+1$. Otherwise, we discard the path. In our example, the prefix of the red path $LS_i^TS_i^TX_PE$ matches the type of interaction, but uses the original mesh instead of the proxy and so will be discarded; the prefix of the blue path $LS_j^TS_j^TX_PE$ matches and uses the proxy for this type of light transport; thus this path contributes to the image. The gray path $LS_i^TS_i^TE$ used the original object and since it does not match the proxy's filter, it also contributes to the image.

Fig. 6 (right) demonstrates how path-proxy linking works with path tracing and next event estimation. Here, the user selects direct light manipulation, i.e. the filter is LX, and a proxy (ID j) has been created for this filter. When the original mesh (ID i) is chosen for the blue path, then the path is discarded: the prefix of the path LDE matches that filter LX, but does not use the proxy linked to it. When a primary ray intersects a joint bounding box, we always use the original mesh to construct paths, i.e. the red path is created as expected, while the gray ray will not intersect geometry.

To summarize, if a path uses the original mesh then the intersecting segment is a primary ray or the path remains valid only if it does not match any filter of the proxies; if a path intersects one of the proxies, it is valid only if its prefix matches the proxy's filter.

Figure 7: *An example of path-proxy linking with shadow and caustic proxies in the* BUNNY *scene. Left: original rendering without manipulation. Right: the shadow is displaced, rotated, and enlarged to cover more of the face, and the caustic is moved, squished, and rotated to highlight the bunny's expression of "evil intent."*

5.3 Multiple Edits and Animated Scenes

Two important requirements for artistic transport manipulation are that edits can be applied to animations, and also be animated (e.g. keyframed) themselves. The first is implicitly met since our method works on transport paths and manipulation is both affected by, and transferred to, static and dynamic objects. Moreover, all selection and manipulation gizmos can be animated in the same way as objects in the scene. All attributes can be keyframed or bound to simulations (e.g. a target gizmo can be bound to a rigid-body simulation). The accompanying video illustrates transport manipulation in dynamic scenes with changing topology. Multiple edits are supported in both path retargeting and path-proxy linking and can also be combined. Path retargeting manipulations do not commute, i.e. the order they are applied in matters.

6 Results and Example Edits

We implemented our prototype editing system as an Autodesk Maya™ 2012 plug-in. All videos and timings were recorded on a PC with a 3.20GHz Intel Core i7 CPU, 8GB of RAM and an NVIDIA GeForce GTX 580 card with 3GB of VRAM. Interactive previews progressively update at 5 to 40 frames per second, depending on the scene complexity and the viewport resolution.

In Fig. 7 path-proxy linking is used to position and scale the shadow (LX_P paths) and refractive caustic ($LS^TS^TX_P$) independently, i.e. using two proxy objects. Fig. 8 shows several sequential manipulations in a simple box scene, combining both retargeting and linking and illustrating secondary effects, which would be tedious to edit with light linking techniques (where multiple lights would need to be generated, some in order to induce fake indirect light).

Fig. 1 illustrates sequential edits, here on $LS^TG^RX_P$ and LS^TX_P paths (glossy reflection off the car and sunlight through the window), a rotation and scaling of a glossy reflection on the floor, as well as reflection editing (X_PS^RE paths). Reflection editing is not possible with light linking; the glossy reflection on the floor can of course be repositioned with light linking, however, this would lack precise control over its shape and orientation, increasing artist effort (see the video for our edits). Converged renderings took several hours in our instrumented SPPM implementation (which is slower than pure SSPM due to the additional path bookkeeping).

Another scene from our video is shown in Fig. 9, where we retargeted a refractive caustic onto a mug that shatters. Notice how the changing indirect illumination is plausibly computed from the manipulated light transport. Fig. 10 demonstrates light manipulation using path retargeting in a complex architectural scene. We applied two edits to LD^RX_P paths, i.e. indirect diffuse illumination, "stretching" the color bleeding across the floor and down the stairs. Similar effects can be obtained with light linking, but would entail a significant effort. We note that lighting design becomes increasingly complex and cluttered when light linking is overused.

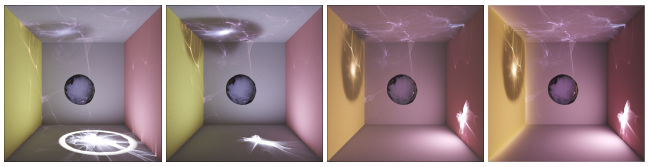

Figure 8: *Bumpy sphere lit by a static spot light. Left to right: original render, path-proxy linking enlarges the shadow, retargeting moves the caustic to the right wall, and retargeting stretches the diffuse indirect lighting from the left wall ($L(.)^*D_{\text{leftwall}}$).*

Domain Expert Feedback. We conducted a survey to assess the usefulness and potential of our manipulation tools and our selection and visualization approaches. Our survey included five technical directors (TDs) at our local film academy, all specializing in lighting and shading. They attested to being very familiar with PBR techniques and having significant experience with graphics packages such as V-Ray, Autodesk's Maya and Max, Side Effects' Houdini, and Pixar's RenderMan. Each TD confirmed that PBR is rapidly being adopted in production, making existing tools obsolete. For fine-tuning and light manipulation they traditionally use light linking, but they also pointed out that it can be tedious and time-consuming in complex scenes, yet often the only available effective tool.

During TD interviews, we demonstrated the use of our tools in the BUNNY, JEWELRY (Fig. 2), GARAGE, and BUDDHA (Fig. 3) scenes. Afterwards, we let the TDs experiment with our tools for a few minutes and then presented them with several editing tasks: we asked them to use manipulators to retarget the caustic in the BUNNY scene (i.e. to focus it, change its color, and move it to a more visually pleasing location). All TDs were able to perform these actions in less than a minute. In the BUDDHA scene we asked them to reshape the caustic with path retargeting and rotate the shadow with path-proxy linking. These editing tasks were accomplished in less than two minutes. The TDs were pleased with the secondary effects (e.g. inter-reflected caustics) being consistent with the edits.

TDs also identified several instances where our tools simplified lighting design, such as being able to freely manipulate caustics without iteratively tweaking material properties like the index of refraction. They additionally identified potential extensions: for instance, film production may employ and reuse fixed lighting rigs across several scenes, and these rigs could also be adapted on a per-scene basis according to artistic requirements using our tools.

Interactive feedback (selection, visualization and manipulation) coupled with instantaneous GI preview was positively received by every TD. Sketch-based selection was generally preferred for gizmo placement. TDs also stated that our visualization served as a powerful tool—even on its own—for increasing scene understanding, and that bundling path lines reduces visual clutter in complex lighting scenarios. They reported that they do not know of any visualization tools that provide more than a photon density map on surfaces, and hence could imagine using our visualization as a stand-alone tool to identify where light is coming from and "debug" scenes. Finally, we asked them to provide feedback on the main components, namely visualization, path retargeting, and path-proxy linking.

TDs discussed the importance of matching even the smallest details set by art direction and stated that our tool's support for multiple and independent local edits would significantly simplify these tasks. Furthermore, the ability to apply these edits while having access to a progressive GI preview facilitates the artistic iteration process. TDs also emphasized that our approach propagates indirect PBR shading effects after editing operations, avoiding approximations of segmenting these effects and simply pre-baking them into textures.

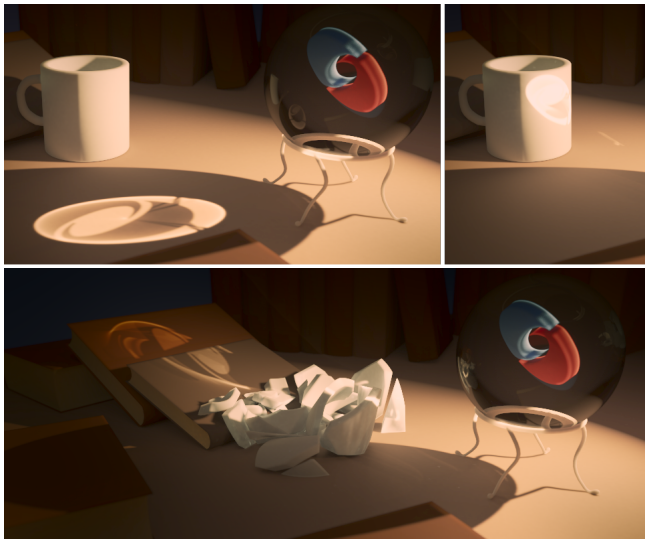

Figure 9: *In reading order: the first frame of the* MUG *animation, without any manipulation. The caustic is manipulated with path retargeting, causing indirect lighting in front of the mug. The mug has been shattered, and the manipulated caustic now appears on a stack of books in the background (see the accompanying video).*

All TDs emphasized the importance of animation support, and confirmed that the ability to keyframe our editing operations was an essential design decision. Workflow-wise, they suggested that artists should be permitted to pick a set of the most important keyframes, which could be cached to speed up previews between them.

7 Conclusion and Future Work

We presented an interactive shading editing tool for intuitive selection, visualization, and manipulation of PBR. Path-space processing, semi-automatic selection approaches, ranking and visualization of selected light transport components, and direct- and indirect-manipulation techniques combine to form the core of our tool.

By integrating into an existing DCC system, artists can easily familiarize themselves with our tool and quickly complete complex lighting design tasks in PBR contexts. Operating directly on path-space solutions of the rendering equation enables the manipulation of complex transport phenomena, including secondary shading effects. Many of our ideas complement existing editing metaphors (e.g. light linking) and, given the feedback from our small survey of experienced users, we believe that our tool can be of great help in the current industrial-strength PBR art pipelines.

We can also imagine situations where our method does not provide a suitable solution. We briefly discuss some artifacts that may arise with exaggerated manipulations. In contrast to Ritschel et al.'s approach [2009], the shape of, say, a caustic may change during manipulation, since we re-trace manipulated photons. Excessive manipulation may also cause completely unlit (source) regions and detachment of shadows. As we always begin manipulation from an initial GI solution (to promote a physically plausible final result), we also cannot create completely new lighting scenarios, which light linking can easily achieve. Reiner et al. [2012] show that transport visualization is challenging with no single technique that works well in all cases. Indeed, we found visualization of diffuse indirect light to be difficult, as this phenomenon is less localized than, e.g. caustics. However, bundling helps even for diffuse light when multiple manipulators are used, as paths are contracted and different transport bundles are isolated.

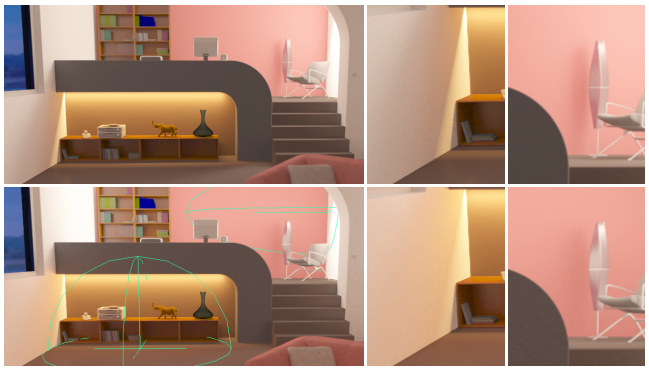

Figure 10: *Manipulating diffuse GI in the* LIVING ROOM *scene. Top to bottom: original GI solution; indirect diffuse light above the shelf and on the back wall are manipulated with path retargeting.*

Of the use cases we cooperatively identified with TDs, we primarily focused on those where intuitive PBR manipulation seemed most viable. Intelligent selection and retargeting/linking handle these (primarily directional) effects quite well. We note that path retargeting can be used—albeit less intuitively—to manipulate smooth diffuse light (see Fig. 10), e.g. by changing its gradient.

Several interesting directions of future work can stem from our initial investigation. Path-proxy linking can be more closely integrated into existing light linking tools, or extended with more flexible proxy generation operations. Path retargeting also has limitations: strong displacements can result in self-intersections, which we could imagine avoiding with back-propagation (akin to inverse kinematics) of the manipulated segment deformation. Second, our retargeting manipulator is unidirectional, reducing the performance of bidirectional GI. Finally, extending our manipulations to participating media poses several interesting challenges.

Acknowledgements. This work was supported by the DFG grant DA 1200/1-1 and the Intel Visual Computing Institute, Saarbrücken, Germany. We thank the TDs and Volker Helzle from Filmakademie Baden-Württemberg, Martin Tillmann for implementing "BendyCaustics", and NVIDIA for hardware donations.

References

BANKS, D. C. 1994. Illumination in diverse codimensions. In *SIGGRAPH'94*, 327–334.

BARZEL, R. 1997. Lighting controls for computer cinematography. *Journal of Graphics Tools 2*, 1, 1–20.

HACHISUKA, T., AND JENSEN, H. W. 2009. Stochastic progressive photon mapping. *ACM Transactions on Graphics (Proc. SIGGRAPH) 28*, 141:1–141:8.

HECKBERT, P. S. 1990. Adaptive radiosity textures for bidirectional ray tracing. *Computer Graphics (Proc. SIGGRAPH) 24*, 4, 145–154.

HOLTEN, D., AND VAN WIJK, J. J. 2009. Force-directed edge bundling for graph visualization. *Computer Graphics Forum 28*, 3, 983–990.

KERR, W. B., AND PELLACINI, F. 2009. Toward evaluating lighting design interface paradigms for novice users. *ACM Transactions on Graphics (Proc. SIGGRAPH) 28*, 3, 1–9.

KERR, W. B., PELLACINI, F., AND DENNING, J. D. 2010. Bendylights: artistic control of direct illumination by curving light

rays. *Computer Graphics Forum (Proc. EG Symposium on Rendering) 29*, 4, 1451–1459.

KŘIVÁNEK, J., FAJARDO, M., CHRISTENSEN, P. H., TABELLION, E., BUNNELL, M., LARSSON, D., AND KAPLANYAN, A. 2010. Global illumination across industries. In *ACM SIGGRAPH Courses*.

LAFORTUNE, E. P., AND WILLEMS, Y. D. 1993. Bi-directional path tracing. In *Proc. Conference on Computational Graphics and Visualization Techniques*, 145–153.

LARSON, G. W., AND SHAKESPEARE, R. 1998. *Rendering with radiance: the art and science of lighting visualization*. Morgan Kaufmann Publishers.

LEE, C. H., HAO, X., AND VARSHNEY, A. 2006. Geometry-dependent lighting. *IEEE Transactions on Visualization and Computer Graphics 12*, 2, 197–207.

MCAULEY, S., HILL, S., HOFFMAN, N., GOTANDA, Y., SMITS, B., BURLEY, B., AND MARTINEZ, A. 2012. Practical physically-based shading in film and game production. In *ACM SIGGRAPH Courses*.

NOWROUZEZAHRAI, D., JOHNSON, J., SELLE, A., LACEWELL, D., KASCHALK, M., AND JAROSZ, W. 2011. A programmable system for artistic volumetric lighting. *ACM Transactions on Graphics (Proc. SIGGRAPH) 30*, 4, 29:1–29:8.

OBERT, J., KRIVÁNEK, J., PELLACINI, F., SÝKORA, D., AND PATTANAIK, S. N. 2008. iCheat: A representation for artistic control of indirect cinematic lighting. *Computer Graphics Forum (Proc. EG Symposium on Rendering) 27*, 4, 1217–1223.

OBERT, J., PELLACINI, F., AND PATTANAIK, S. N. 2010. Visibility editing for all-frequency shadow design. *Computer Graphics Forum 29*, 4, 1441–1449.

OKABE, M., MATSUSHITA, Y., SHEN, L., AND IGARASHI, T. 2007. Illumination brush: Interactive design of all-frequency lighting. In *Proc. Pacific Graphics*, 171–180.

PELLACINI, F., TOLE, P., AND GREENBERG, D. P. 2002. A user interface for interactive cinematic shadow design. *ACM Transactions on Graphics (Proc. SIGGRAPH) 21*, 3, 563–566.

PELLACINI, F., BATTAGLIA, F., MORLEY, K., AND FINKELSTEIN, A. 2007. Lighting with paint. *ACM Transactions on Graphics 26*, 2.

PELLACINI, F. 2010. envyLight: an interface for editing natural illumination. *ACM Transactions on Graphics (Proc. SIGGRAPH) 29*, 4, 34:1–34:8.

PHARR, M., AND HUMPHREYS, G. 2010. *Physically Based Rendering, Second Edition: From Theory To Implementation*. Morgan Kaufmann Publishers.

POULIN, P., AND FOURNIER, A. 1992. Lights from highlights and shadows. In *Proc. ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, 31–38.

REINER, T., KAPLANYAN, A., REINHARD, M., AND DACHSBACHER, C. 2012. Selective inspection and interactive visualization of light transport in virtual scenes. *Computer Graphics Forum (Proc. Eurographics) 31*, 2, 711–718.

RITSCHEL, T., OKABE, M., THORMÄHLEN, T., AND SEIDEL, H.-P. 2009. Interactive reflection editing. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia) 28*, 5, 129:1–129:7.

RITSCHEL, T., THORMÄHLEN, T., DACHSBACHER, C., KAUTZ, J., AND SEIDEL, H.-P. 2010. Interactive on-surface signal deformation. *ACM Transactions on Graphics (Proc. SIGGRAPH) 29*, 4, 36:1–36:8.

TABELLION, E., AND LAMORLETTE, A. 2004. An approximate global illumination system for computer generated films. *ACM Transactions on Graphics (Proc. SIGGRAPH) 23*, 3, 469–476.

VEACH, E., AND GUIBAS, L. 1994. Bidirectional estimators for light transport. In *Proc. EG Workshop on Rendering*, 147–162.

VEACH, E., AND GUIBAS, L. J. 1997. Metropolis light transport. In *SIGGRAPH'97*, 65–76.

VEACH, E. 1998. *Robust monte carlo methods for light transport simulation*. PhD thesis. AAI9837162.

A Robust Bidirectional Transport

Our tools can also be formulated in Veach's transport framework [1998], enabling the seamless use of our approach in light transport methods such as path tracing, (full) BDPT and Metropolis Light Transport [1997].

Path-proxy linking. Here, different scene representations are used for different types of paths. For sampling (as in BDPT), where path length is unknown prior to path construction, Russian roulette should be applied to decide which representation to use: once a path is formed, it is rejected if it does not exist for the chosen representation. If, after subpath identification, we know that a representation will not match, it is not selected. Our probabilistic proxy/path selection (Sec. 5.2.1) increases noise since new paths can be discarded; this increase of noise can be eliminated if *all* $N+1$ proxies are traced against and only the path generated by the one valid mesh is retained. This scheme would however increase computation cost.

Path retargeting. Retargeting effectively rotates *a surface's outgoing tangent frame* such that an outgoing segment points towards the user-specified target (Fig. 11, left). Similarly to modified/bumpmapped shading normals, this introduces a non-symmetric BRDF [Veach 1998] for bidirectional transport. Given a surface point with normal N, BSDF f, and a path with incident and exitant directions ω_i and ω_o, path retargeting defines a rotation $\mathbf{R}$ of the tangent frame: the modified BSDF is

$$f'(\omega_i \to \omega_o) = f(\omega_i \to \mathbf{R}(\omega_o))$$

and its adjoint is derived similarly to [Veach 1998, Sec. 5.3.2] as

$$f'^*(\omega_i \to \omega_o) = f(\mathbf{R}^T(\omega_o) \to \omega_i)|\mathbf{R}^T(\omega_o) \cdot N|/|\omega_o \cdot N|.$$

The adjoint BSDF f'^* should be used when the sampling and manipulation directions are inverted. Retargeting constrained to a source region must be properly treated using rejection sampling for paths from the direction opposite the manipulation direction. If e.g. an eye subpath is constructed for a path that has been modified from the light direction, the inverse source-target transformation must be applied to ensure that the original light subpath hits the source region (Fig. 11, right); otherwise such a path is rejected.

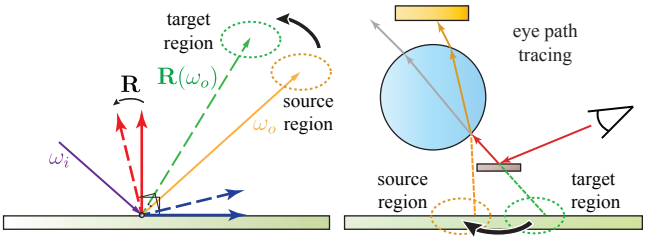

Figure 11: *Left: path retargeting can be viewed as a rotation of either the incident or exitant tangent frames, depending on the manipulation (whether light or eye subpaths are modified) and the "direction" of the sampling method (either importance or radiance). Right: when we trace opposite to the manipulation direction, the inverse transformation has to be applied at each interaction to check for potential manipulations from another side.*