Vision, Modeling, and Visualization (2017)
M. Hullin, R. Klein, T. Schultz, A. Yao (Eds.)

# LiteMaker: Interactive Luminaire Development using Progressive Photon Tracing and Multi-Resolution Upsampling

Katharina Krösl[†1,2], Christian Luksch[1], Michael Schwärzler[1] and Michael Wimmer[2]

[1]VRVis Research Center, Austria
[2]TU Wien, Institute of Computer Graphics and Algorithms, Austria

**Figure 1:** *Typical lighting-design scenarios rendered in* HILITE *[VRV], exhibiting complex geometries, materials, and multiple luminaires from which a physically plausible lighting simulation has to be calculated and visualized. The design of luminaire models for such lighting concepts is currently based on an iterative development cycle, resulting in a tedious and time-consuming design workflow due to offline lighting simulations required for verification. We introduce an interactive filtering technique efficiently approximating the ground-truth results for luminaire intensity profiles, allowing for the first time for an interactive workflow in luminaire development and lighting design.*

**Abstract**

*Industrial applications like luminaire development (the creation of a luminaire in terms of geometry and material) or lighting design (the efficient and aesthetic placement of luminaires in a virtual scene) rely heavily on high realism and physically correct simulations. Using typical approaches like CAD modeling and offline rendering, this requirement induces long processing times and therefore inflexible workflows. In this paper, we combine a GPU-based progressive photon-tracing algorithm to accurately simulate the light distribution of a luminaire with a novel multi-resolution image-filtering approach that produces visually meaningful intermediate results of the simulation process. By using this method in a 3D modeling environment, luminaire development is turned into an interactive process, allowing for real-time modifications and immediate feedback on the light distribution. Since the simulation results converge to a physically plausible solution that can be imported as a representation of a luminaire into a light-planning software, our work contributes to combining the two former decoupled workflows of luminaire development and lighting design, reducing the overall production time and cost for luminaire manufacturers.*

**CCS Concepts**
•***Computing methodologies*** $\rightarrow$ *Ray tracing; Image processing; Mesh geometry models;*

## 1. Introduction

In this paper, we present a solution for interactive light planning that combines *progressive photon tracing* (PPT) and a novel multi-resolution image-filtering approach. By integrating our algorithm into a 3D modeling environment, we facilitate rapid iterations in the luminaire development process and are therefore able to speed up current workflows in light planning.

In the industry-driven process of creating and selling lighting concepts, two main aspects in the development rely heavily on physically reliable simulation and visualization – *luminaire development* and *lighting design*. The latter, the placement of virtual luminaires in a 3D scene under the consideration of light conditions, aesthetic aspects and costs, has recently received increased scientific attention [SOL*16,SW14,LHH*13]. It has been shown [VRV]

† kroesl@vrvis.at
Authors Version. The definitive version will appear at http://diglib.eg.org/.

that this design process can be carried out in a completely interactive way using novel methods from the field of Visual Computing (see Figure 1 for examples). The first step in light planning, though – the development and simulation of luminaires – still suffers from the fact that only computationally expensive offline rendering algorithms are sufficiently accurate to capture complex effects of lighting reflector optics and lenses. Additionally, the corresponding 3D modeling and simulation processes are usually performed in different tools and are hence completely decoupled. This results in a costly and time-consuming development process of luminaires, which is not integrated in or directly connected to the lighting-design process.

These shortcomings as well as the advances in lighting design have created the need for a similarly interactive process in the field of luminaire development: This would not only combine interactive modeling, immediate simulation and real-time visualization during the luminaire development process, it would also allow coupling it with interactive lighting design, making it possible for the first time for a designer to alter the internal parameters and therefore the light distribution behavior of a luminaire *during* its placement. One of the major challenges in reaching this goal is to maintain physical correctness (a requirement in any light-planning application) while achieving interactivity.

In this paper, we therefore introduce an interactive global-illumination algorithm that simulates the light distribution of a luminaire, and produces meaningful intermediate results (for a visualization at interactive frame rates) that converge to a physically correct solution. To achieve this, we combine our novel multi-resolution image-filtering techniques with an interactive, progressive photon-tracing algorithm. This approach is integrated into a luminaire modeling and visualization tool that enables the user to interactively edit the geometry of the model or change its material properties, and get intermediate preview results rendered and displayed at interactive frame rates. This instant feedback possibility accelerates luminaire development significantly. Furthermore, the results of our algorithm are photometric representations of luminaires that can be directly imported into and used as virtual light sources in a light-planning software, allowing designers to adapt the light distribution behavior of a luminaire for a specific scene. Consequently, our approach transforms two independent workflows into one flexible, interactive process. The contributions of this paper are the following:

- A progressive photon-mapping technique for interactive visualization of intermediate results by using
- a multi-resolution image-filtering method, smoothing the noisy areas while preserving details on edges. This allows
- a coupling of the previously independent tasks of 3D modeling, simulation and visualization, and therefore
- the introduction of a completely interactive, flexible workflow that spans from luminaire development to lighting design.

The remainder of this paper is organized as follows: In Chapter 2 we discuss related work in photon mapping, image filtering and the state-of-the-art tools and techniques in lighting design. Chapter 3 gives an overview of the main components of our solution. In Chapter 4 we show how we adapt a progressive photon tracer to suit our specific use case. We present our novel multi-resolution image-filtering approach in Chapter 5 and discuss our results in Chapter 6. Finally, Chapter 7 gives a conclusion based on the presented results and an outlook on future work.

## 2. Related Work

**Photon Mapping**, proposed by Jensen [Jen96], is a rendering algorithm that is able to handle effects of light paths including specular surfaces (like caustics) very efficiently. The idea is to trace photons from a light source through the scene and store their hits on diffuse surfaces in a so-called photon map. The radiance is then estimated by the density of the photons. Walter [Wal98] studied this algorithm extensively. Garcia et al. [GHUPS14] analyze common filter kernels for density estimation and present new estimators with improved filter kernels. Despite all improvements in this area, most radiance estimation methods still increase the computation time of photon mapping due to costly calculation of the radiance estimate. Photon-relaxation approaches as presented by Spencer and Jones [SJ13a, SJ13b] try to distribute photons uniformly in a local area. During photon relaxation, feature-detection methods preserve high-frequency details in illumination information. Keller and Wald [KW00] introduce an importance-driven algorithm affecting the deposition of photons by storing them only in areas of high visual importance. Selective photon tracing [DBMS02] uses so-called pilot photons to detect regions requiring illumination updates in dynamic scenes. The regions are then refined by generating similar light paths using a quasi-random sequence. Günther et al. [GWS04] apply this method in order to exclusively refine light paths of caustics. The adaptive importance shooting technique, presented by Liu and Zheng [LZ14], calculates error values of neighbors in order to construct an adaptive cumulative distribution function, which is used to reflect photons, upon surface intersection, into directions of areas of interest. In order to overcome the memory constraint of the photon map, which limits the accuracy of radiance estimation, Hachisuka et al. introduced progressive photon mapping [HOJ08] and stochastic progressive photon mapping [HJ09], a more generalized formulation that allows typical distributed ray-tracing effects. Recently, Kaplanyan et al. [KD13] improved the convergence properties of progressive photon mapping by introducing an adaptive bandwidth selection. State-of-the-art progressive photon-mapping methods have reached interactive frame rates on the GPU [HJ10, MLM13]. Georgiev et al. [GKDS12] reformulated photon mapping as a sampling technique for rendering algorithms based on the *Monte Carlo* method. This allows the combination of photon mapping with general global-illumination algorithms based on ray tracing in one framework. The result is a robust light-transport algorithm for arbitrary scenes.

**Image Filtering.** There has been a lot research in the field of image processing involving guidance images for edge-preserving smoothing algorithms that achieve visually promising results. Petschnigg et al. [PSA*04] present a variety of techniques to enhance images and synthesize new images with improved quality from a pair of photographs, one taken with flash to capture details, and the other one taken without flash to accurately capture ambient illumination. They introduce a joint bilateral filter for ambient image denoising. He et al. [HST13] use the concept of this joint bilateral filter to develop a guided image-filtering technique that can use the input image itself as guidance image. Zhang et al. [ZSXJ14] developed a "rolling guidance filter" to smooth images by using a scale-aware image filter that also takes a guidance image into account. Cho et al. [CLKL14] propose a structure-preserving image decomposition operator based on the bilateral filter, introduced by Tomasi and Manduchi [TM98].A similar approach is introduced by Kniefacz and Kropatsch [KK15]. They propose using a separable Gaussian range filter or a symmetric nearest-neighbor filter as guided filter, to produce faster results in comparison to the rolling guidance filter [ZSXJ14]. State-of-the-art image-filtering approaches perform very well in noise-reduction scenarios. The proposed smooth-and-restore method by Kniefacz and Kropatsch [KK15] seems to be the most promising in terms of image quality and speed. However, since all of these image-filtering techniques require more calculations than a simple bilateral filter, they also have a higher computation time.

**Lighting Design.** Commonly used software products in the field of lighting design are Relux [Rel], DIALux [DIA] or AGi32 [Lig]. They combine CAD modeling with tools for measurement and evaluation of lighting solutions. The luminaire data is taken from photometric measurements provided by luminaire manufacturers, typically in the IES [Ill] or EULUMDAT [Sto] file format. The light simulation of Dialux and AGi32 is based on the radiosity method [GTGB84], while Relux uses the open-source Radiance renderer by Greg Ward [War]. Modeling and simulation are performed consecutively in these tools. HILITE, a prototype with a new approach for lighting design, is currently being developed in an academic environment [VRV]. It combines physically based real-time rendering with an interactive lighting simulation for complex architectural environments. It uses a many-light global-illumination solution baked in light maps [LTH*13] and filtered environment maps for rendering glossy material effects [LTM*14].

## 3. Application Overview

In this work, we present an algorithm to simulate the light distribution of luminaires that performs at interactive frame rates, produces meaningful intermediate results, and converges to a physically accurate solution. Our application, as shown in Figure 2, consists of two major parts: a rendering stage and a simulation stage, which are executed in parallel. The great benefit of this parallelism is that it allows real-time editing of the luminaire model while the next iteration of the lighting simulation is being calculated. Each modification of geometry or material properties (which can be done after getting the first previews) instantly restarts the simulation, turning luminaire development from an iterative into an interactive process without unproductive waiting times.

### 3.1. Rendering Stage

In the rendering stage, the light-source geometry is visualized as a 3D scene enclosed by a cube that shows the light distribution (see Figure 3, bottom left). Our algorithm works iteratively, and progressively refines the result. Between iterations, the user can modify the geometry and material properties of the luminaire by manipulating the 3D model. After each iteration of the simulation, a preview of the final result is rendered and shown in the render view.

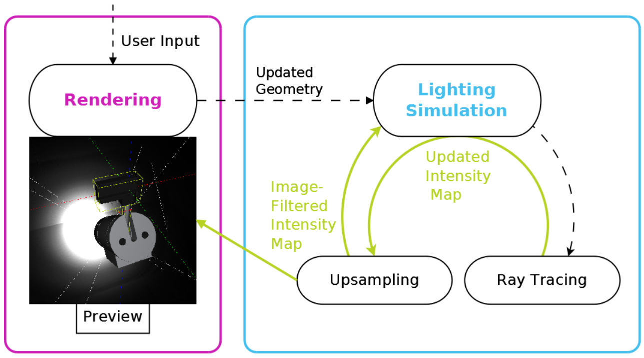



**Figure 2:** *The main components of our application are a rendering stage visualizing the light distribution of a luminaire, and a simulation stage in which this lighting information is calculated.*

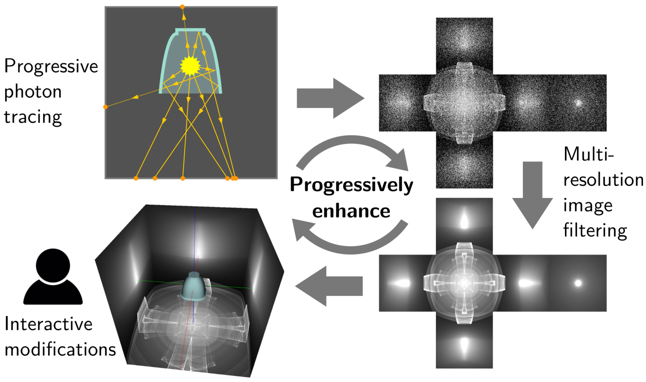



**Figure 3:** *Our algorithm uses progressive photon tracing to emit photons into the scene. The resulting intensity map is processed in an image-filtering step to obtain a smooth image that can then be shown as preview in the render view, where the user can make modifications to the luminaire. The intermediate results produced by our algorithm are progressively enhanced with each iteration.*

### 3.2. Simulation Stage

The simulation's main components are a photon-tracing process and a multi-resolution image-filtering method. Both are executed at each iteration. Figure 3 shows the workflow of the algorithm.

**Interactive, Progressive Photon Tracing.** First, rays are cast from the light-emitting part of the light source and reflected on the light-source geometry. Photons are only stored at intersections with the surrounding cube map, which results in a cube-map texture that represents the light distribution of the luminaire. The obtained cube map then functions as input for the image-filtering procedure. By designing our algorithm in a progressive manner, we enable the user to interactively make modifications to the scene, which is taken into account as soon as the next iteration of the light simulation starts.

**Image Filtering.** Our multi-resolution image-filtering technique smoothes a noisy cube map while preserving details like shadow borders. For this purpose a mipmap image pyramid is constructed and processed from top to bottom. Each texel value is compared to a user-defined global threshold. Values that lie below this threshold are replaced by an interpolated value from the next higher mipmap level in our upsampling step. Our whole image-filtering method operates on 2D textures. We implemented an elaborate border-handling method to calculate the correct coordinates at cube-map borders. We define the neighbors for every face and specify the necessary transformation (whether to switch or invert coordinate axes) and offset for each mipmap level to get the corresponding neighbor texel.

## 4. Photon Tracing

Our light-simulation algorithm is based on the concept of photon tracing. This algorithm has been chosen due to its strengths in simulating complex interreflections and caustics. These effects are decisive in luminaire development and should be depicted as soon as possible in the visualization. Our implementation casts importance-sampled rays from the light-emitting part of the luminaire geometry into the scene, each transporting the same amount of light energy. Upon intersection with the geometry, a Russian Roulette [Jen01] strategy based on the surface reflectivity of the BRDF for the incoming direction is used to determine if the ray-traversal is continued. Consequently, an importance-sampled reflection direction is calculated according to the BRDF in order to maintain photons with equal amount of light-energy. We have chosen to use the BRDF model of Kurt et al. [KSKK10], since it allows a detailed description of highly reflective surfaces and matches measured BRDFs more accurately than previous models. In contrast to standard photon-mapping algorithms, our application does not require a typical photon-map data structure. Instead, we accumulate ray hits in a surrounding cube map in the form of a photon count in the nearest texel. This cube-map texture will be referred to as *intensity map* and represents the light distribution of a luminaire. During each iteration, additional photons are cast, and the photon count in the intensity map is increased. Since all photons represent an equal amount of light energy, the photon count of a texel is directly related to the light energy received. A radiance density estimate with the same quality as standard photon mapping could be calculated by applying a radial image filter. However, sharp edges and fine details would get blurred. Instead of such a basic image filter, we apply our novel multi-resolution upsampling technique.

## 5. Multi-Resolution Image Filtering

Our multi-resolution image-filtering technique is applied to smooth the intensity map while preserving details like borders of intensity regions (areas with similar texel values). This way, our algorithm is able to show meaningful previews in the interactive visualization and modeling view while the simulation runs in parallel, converging to a physically plausible solution.

### 5.1. Classification of Texel Values

Our goal is to obtain a high-resolution intensity map image that has sharp edges at borders of intensity regions, and a low variance in homogeneous regions. In order to reach that goal in the affected image regions, it takes a certain overall amount of photons and rendering time (that is too large to be used in an interactive setup) to get a corresponding even photon distribution. At the early stages of a simulation run, the distributed light energy is rather uneven, and texels that have a significantly lower photon count than their neighbors show up as noise in the image. To still get images that can be shown as previews of the light distribution, these regions have to be properly smoothed by replacing the low photon counts by values that more accurately represent the illumination in this region. To distinguish between texels whose values should be replaced and texels whose values should not be modified, we classify them as *stable* or *unstable*, depending on their photon count. This classification is done by a configurable threshold value – comparable to a photon-count parameter needed for the density estimation in common photon-mapping algorithms with a nearest-neighbor search.

### 5.2. Image Pyramid and Pull-Push

We implemented a pull-push algorithm for our multi-resolution image-filtering approach, similar to the one developed by Gortler et al. [GGSC96] for their Lumigraph system. Different resolutions of the same texture are used to build up a mipmap image pyramid, as shown in Figure 4. During the pull phase, values of the higher-resolution textures are summed up to get approximations for lower resolutions. In the push phase, lower resolution values are used to upsample values for higher resolutions, and fill in regions at higher resolutions that do not have sufficiently high values. Instead of refining the multi-resolution buffer at some locations as proposed by Nicols and Wyman [NW09], we use multiple mipmap levels and store the number of photons that hit the area of a texel in each mipmap level. Furthermore, we keep the number of interpolation calculations low by not upsampling each mipmap level as a whole and combining it with the next higher resolution. Instead, we replace only the values of texels with a photon count below a certain threshold by upsampling them from the next higher mipmap level.

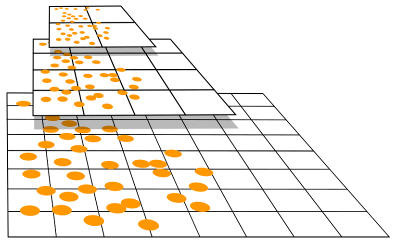


**Figure 4:** *2D pull-push. Higher resolutions exhibit bigger gaps, resulting in low-intensity texels that require upsampling [GGSC96].*

### 5.3. Upsampling

By upsamling a texel as shown in Algorithm 1 we obtain a new value to replace the photon count of the currently processed texel. Upsampling a texel with a too low photon count, regardless of the intensity differences between texels in this area, yields problems when using a lot of mipmap levels together with a high threshold. This combination results in several mipmap levels where a large amount or all texels are upsampled, as none of them satisfies the threshold. As a consequence, sharp edges get blurred. For example, bilinearly interpolating over a shadow border not only blurs the border, but also creates a small intensity leak from a light pixel into the shadow area. This intensity leak increases with each upsampled mipmap level. The intuitive solution to this problem is to upsample all texels that have too few photons, except texels that are directly on a border to a region with higher intensity. In order to exclude shadow borders from the upsampling pass, we first need to detect these "shadow borders".

**Data:** texel $P$, position $P_H$ of $P$ on next higher mipmap level $L_H$, nearest texels $H_i$ to $P_H$.
**Result:** new intensity value $f(P)$ for $P$.

**begin**
  $P_H \longleftarrow P$ on $L_H$;
  **for** $i \leftarrow 1$ **to** 4 **do**
    look up intensity values $f(H_i)$ near $P_H$ on $L_H$;
    `// for 4 nearest texel neighbors`
  **end**
  $f(P) \longleftarrow$ `Interpolate` $(f(H_i)|i = \{1,2,3,4\})$;
**end**

**Algorithm 1:** upsampling a texel

**Classification of Shadow Border Texels.** As shown in Algorithm 2 we consider an *unstable* texel as being located on a shadow border, based on the values of its neighboring texels. An unstable texel is upsampled either if it does not have any stable neighbors (and is therefore recognized as not being located on a shadow border) or if it is surrounded by a high number ($> 6$) of texels with high photon counts. In this case we consider it as isolated texel in a non-shadow region that has not been reached by enough photons yet to be classified as *stable* and therefore has to be upsampled. Texel values located on shadow borders are not interpolated. Done iteratively on every mipmap level, this technique can preserve shadow borders of dark shadows, as we avoid energy bleeding from bright to dark areas. However, borders might look pixelated due to the fact that the texels next to shadow borders are not upsampled on any mipmap level. To avoid this pixelation, we only do these checks for higher mipmap levels and upsample every texel that does not satisfy the threshold on level 1 and 0. The whole workflow of our multi-resolution image-filtering approach is shown in Figure 5. However, there are still cases where intensity leaks can occur near a shadow border and grow with each mipmap level, discussed in the next section.

**if** *texel P is unstable* **then**
  **foreach** *texel neighbor $P_i$ of P* **do**
    **if** *intensity value* $f(P_i) + f(P) \geq 2$ threshold **then**
      `// texel neighbor` $P_i$ `is stable`
      borderCounter ++;
    **end**
  **end**
  **if** borderCounter $== 0$ *or* borderCounter $> 6$ **then**
    `Upsample` $(P)$
  **end**
**end**

**Algorithm 2:** shadow border classification

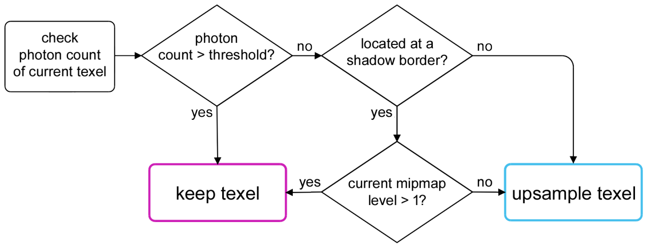



**Figure 5:** *Multi-resolution image-filtering workflow.*

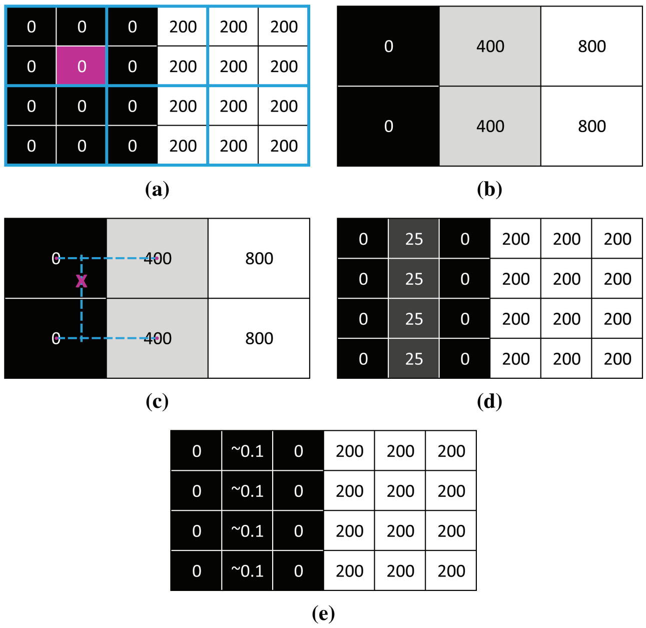



**Figure 6:** *Four texels framed in blue (a) are summed up to one texel on the next higher mipmap level (b). A bilinear interpolation (c) near a shadow border can produce intensity leaks (d). Bilateral interpolation produces significantly smaller intensity leaks (e).*

### 5.4. Intensity Leaks Near Shadow Borders

As depicted in Figure 6a, texels that are in shadow have a photon count of 0, while the illuminated texels next to the shadow border have a photon count of 200 each. The shadow texels next to the border are not upsampled (see Section 5.3), but all other texels with low photon count will be modified. In the given example, the texel highlighted in pink is processed: It does not satisfy our threshold and it is not directly located at a shadow border. According to the method we have proposed so far, it should thus be upsampled. Figure 6b shows the next higher mipmap level: Each texel is the sum of the corresponding four texels (framed in blue in Figure 6a) of the lower level. We now take the four texels of the higher mipmap level and do a bilinear interpolation as shown in Figure 6c. By bilinearly interpolating the values of the presented example, we get a resulting value of 100. Since each texel on the higher mipmap level in our image pyramid is a sum (not an average) of four texels of the lower level, we then divide the calculated value by four. As final result, we obtain a value of 25 for our currently processed texel, which represents an intensity leak from bright into darker areas.The resulting intensity leak for this whole region is shown in Figure 6d.

### 5.5. Bilateral Interpolation

In order to reduce the described intensity leaks, we implemented bilateral filtering using a Gaussian distribution as described by Tomasi and Manduchi [TM98] to interpolate texels. It consists of a combination of a domain filter using a closeness function based on Euclidian distance in the spatial domain, and a range filter, applying a photometric similarity function dependent on the intensity difference of texels. If we want to upsample texel $P = (x,y)$ and apply a bilateral interpolation, we first calculate the Euclidian distance of our target texel $P$ to all 4 texel centers $H_1 = (x_1,y_1)$, $H_2 = (x_2,y1)$, $H_3 = (x1,y2)$ and $H_4 = (x2,y2)$ in the proximity of $P$'s location on the next higher mipmap level. Then we use these distances $d_i$ to construct a Gaussian filter in the spatial domain as *closeness function* $c_i$ for each texel $H_i$:

$$c_i = e^{-\frac{1}{2}\left(\frac{d_i}{\sigma_d}\right)^2},$$

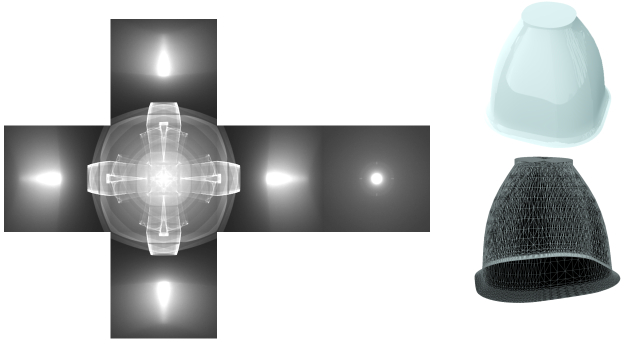
where $\sigma_d$ is the standard deviation of the Gaussian filter. Analogous to $c_i$, we now calculate the *similarity functions* $s_i$, by using the intensity difference of $P$ to a texel $H_i$ on the higher mipmap level. This calculation requires knowledge of the intensity value stored at $P$. This can be calculated when applying a bilateral filter to an image, but for our case the actual intensity value $f(P)$ is unknown. To solve this issue, we use the bilinearly interpolated intensity value $\tilde{f}(P)$ as approximate value for $f(P)$:

$$s_i = e^{-\frac{1}{2}\left(\frac{\delta(\tilde{f}(P),f(H_i))}{\sigma_r}\right)^2}.$$

We can now use the *closeness functions* and *similarity functions* for each texel $H_i$ as weighting factors for the final interpolation and normalize by the sum of weights. The factor 4 in the denominator accounts for the lack of normalization in our mipmap pyramid.

$$f(P) = \frac{\sum_{i=1}^{4} f(H_i)c_i s_i}{4\sum_{i=1}^{4} c_i s_i}$$

An example for bilateral interpolation, yielding a much smaller intensity leak than bilinear interpolation, is shown in Figure 6e.



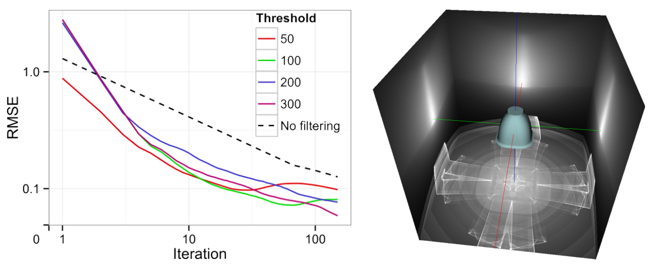
**(a)** *Reference image (cube map)* **(b)** *luminaire*

**Figure 7:** *Reference image with cube-map layout (a) and 3D model of the luminaire reflector in our test scene (b), intentionally skewed to feature complex interreflections and exhibit caustics.*

## 6. Results

To evaluate our results, we generated a reference image by casting 13 million photons in each iteration, and no image filtering was performed on these photon traced results. As reference image regarding image quality, we chose an iteration number of 5,000 (see Figure 7a), corresponding to almost 66 billion photons, obtained after about 12.5 hours of rendering time. The whole image has a resolution of 2048 × 3072, with $1024^2$ pixels for each cube-map face. We chose a bell-shaped luminaire, shown in Figure 7b, as test scene, and modified its material properties to make it reflective and translucent at the same time (similar to the material properties of opal glass or translucent glass). That way, photons are widely spread along the surrounding cube, thus creating a worst-case scenario for any photon-tracing algorithm. Other examples of test scenes can be seen in the accompanying additional material. The presented results were produced on an NVIDIA GeForce GTX 970 graphics card.

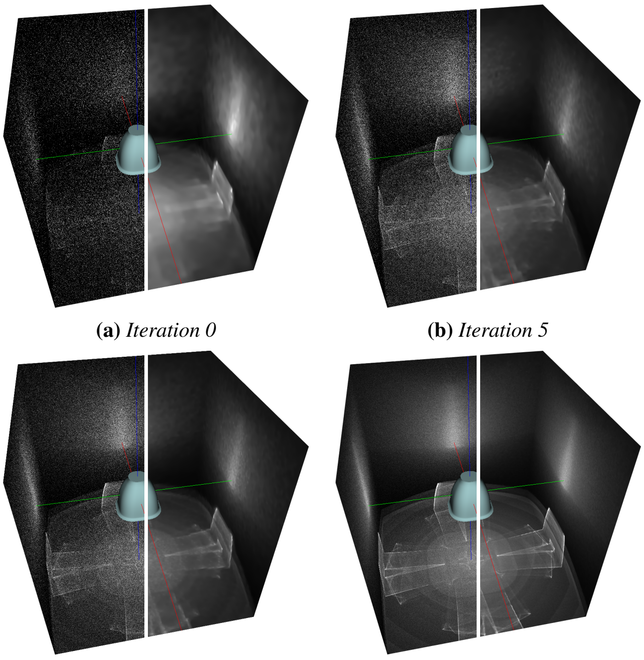
**Improved Image Quality of Filtered Intensity Maps.** As quality measure regarding the image quality of our results, we use the Root Mean Squared Error (RMSE). The intensity maps that are compared to each other store the total amount of photons per texel instead of the integral over all photons. Therefore, we scale each image by its mean, so that their values are in the same range, before we can calculate the RMSE between them. Figure 8a shows that the RMSE between the reference image and our intermediate results of subsequent iterations vastly decreases during the first few iterations, and does not change significantly afterwards. Depending on the chosen threshold, results differ over time. The impact of a certain threshold depends on the photon distribution in the intensity map, which is influenced by the luminaire material, the cube-map texture resolution and the number of photons that are cast. Because this photon distribution is hard to predict, our current implementation requires the user to manually adjust the threshold. Our experiments showed that during the first few iterations, a low threshold (e.g., 50) is more beneficial, while higher thresholds (e.g. 150-250) yield better results after more iterations. It can be seen that our image-filtered results have a significantly lower RMSE than a photon-traced image without applying an image-filtering method to it.




**(a)** *RMSE* **(b)** *Reference image*

**Figure 8:** *(a) RMSE of photon images without multi-resolution image filtering (dashed, rendered in 3.6 seconds per iteration) and our image-filtered intensity maps at different thresholds (solid colored, rendered in 3.3 seconds per iteration), measured against the reference image (b) (unfiltered, rendered in ~12.5 hours).*

**Interactive Visualization of Intermediate Results.** We obtained each of the image-filtered results (used in Figure 8a) in approximately 3.5 to 3.7 seconds, of which about 0.3 seconds were consumed by the multi-resolution image filtering. Since a key feature of our application is its interactivity, we also evaluated the results of a faster test run, configured to render one iteration per second. Figure 9 shows iterations 0, 5, 10, and 100 of the simulation for our test scene (Figure 7b) as displayed in the render view of our interactive tool. The left part of each subfigure shows the unfiltered intensity map and the right part shows the same intensity map after applying our multi-resolution image-filtering approach. Finally, Figure 8b shows our reference image in the 3D view. The side-by-side views show that our multi-resolution image-filtering approach not only enhances the objective error metric, but also significantly improves the perceived visual quality of the intermediate results.



**(a)** *Iteration 0* **(b)** *Iteration 5*

**(c)** *Iteration 10* **(d)** *Iteration 100*

**Figure 9:** *Intermediate results with a rendering time of ∼1 second and* 500 000 *rays per iteration and a threshold of 100, for a texture of size 2048 × 3072 texels (*$1024^2$ *per cube-map face). Left side of each figure: unfiltered intensity map. Right side: filtered preview.*

**3D Modeling and Simulation.** For the interactive development of luminaires, we created an editor that provides a multitude of parameters (for transformation of the geometry, modification of material properties, adaptation of rendering speed,...) and displays the progress of the simulation. The interactive visualization of intermediate results allows a luminaire engineer to see artifacts like unwanted caustics in the first few seconds of a simulation and makes it possible to immediately modify the 3D model of the luminaire in question. For early prototyping and fast previews, the resolution of the cube-map texture can be reduced (e.g., to $128^2$ per face) to speed up the simulation. If the same amount of photons are cast onto a smaller texture, the intermediate results are smoother than for high-resolution textures, and obtained quicker, since the time used for image filtering heavily depends on the texture resolution. The simulation time per iteration can also be significantly reduced by reducing the number of photons per iteration. For high-quality results that show every fine detail of shadows or caustics a higher texture resolution (e.g. $1024^2$) is needed. We refer to the accompanying video to observe the tool in practice.

**Novel Workflows.** The final result as well as the intermediate results of our approach can be imported and directly used in a light-planning software that supports photometry data in the form of cube-map textures. Spherical or any other data representation for other software could be easily generated as well. Early stages of a luminaire in development can be examined as part of a lighting concept in a 3D scene of a light-planning software as shown in Figure 10. This allows the designer to tailor luminaires to specific scenes or lighting concepts and creates a more flexible workflow by connecting luminaire development and lighting design. Consultations with industry experts showed that luminaire manufacturers appreciate the speed-up our work presents for luminaire development and the possibilities this novel workflow offers. However, introducing such a new workflow, which spans the complete development and design process, could require structural changes both in the team compilation and other established, connected workflows.

**Figure 10:** *Comparison of the lighting produced by two different luminaires with different material properties and emission profiles, created with our luminaire editor, and imported into the light-planning software* HILITE *[VRV].*

## 7. Conclusion and Future Work

We presented an interactive global-illumination algorithm that simulates the light distribution of a luminaire and produces meaningful intermediate results at interactive frame rates before it converges to a physically plausible solution. This allows us to provide an interactive visualization of the light distribution while it is being calculated. Our multi-resolution image-filtering approach significantly enhances the image quality of the intermediate simulation results when compared to non-filtered textures. Since these images are rendered at interactive frame rates, they can serve as previews of the light distribution in our luminaire editor. By integrating our approach into a tool that also provides modeling functionalities, we turned the very time-consuming iterative luminaire development into an interactive process. Since our editor is used during lighting design to directly edit the light distribution of luminaires, we provided a new way to combine two formerly decoupled workflows for luminaire development and lighting design. In future work, we plan to replace our user-defined threshold by an automatically calculated optimal value for the whole image or separate thresholds for different image regions that are adjusted in each iteration. Furthermore, importance sampling could be integrated by assigning an importance function as emission profile to an illuminant.

### Acknowledgments

This work was enabled by the Competence Centre VRVis. VRVis Forschungs-GmbH is funded by COMET – Competence Centers for Excellent Technologies (854174) by BMVIT, BMWFW, Styria, Styrian Business Promotion Agency – SFG and Vienna Business Agency – A Service offered by the City of Vienna. The COMET Programme is managed by FFG.

### References

[CLKL14] CHO H., LEE H., KANG H., LEE S.: Bilateral texture filtering. *ACM Trans. Graph. (TOG) 33*, 4 (2014), 128. 3

[DBMS02] DMITRIEV K., BRABEC S., MYSZKOWSKI K., SEIDEL H.-P.: Interactive global illumination using selective photon tracing. In *Proceedings of the 13th Eurographics Workshop on Rendering* (2002), EGRW '02, Eurographics Association, pp. 25–36. 2

[DIA] DIAL GMBH: DIALux. http://www.dial.de/dialux/. Accessed: 2017-02-13. 3

[GGSC96] GORTLER S. J., GRZESZCZUK R., SZELISKI R., COHEN M. F.: The lumigraph. In *23rd annual conference on Computer graphics and interactive techniques* (1996), ACM, pp. 43–54. 4

[GHUPS14] GARCIA HERNANDEZ R. J., URENA C., POCH J., SBERT M.: Overestimation and underestimation biases in photon mapping with non-constant kernels. *IEEE Trans. Vis. Comput. Graphics 20*, 10 (2014), 1441–1450. 2

[GKDS12] GEORGIEV I., KŘIVÁNEK J., DAVIDOVIČ T., SLUSALLEK P.: Light transport simulation with vertex connection and merging. *ACM Trans. Graph. 31*, 6 (Nov. 2012), 192:1–192:10. 2

[GTGB84] GORAL C. M., TORRANCE K. E., GREENBERG D. P., BATTAILE B.: Modeling the interaction of light between diffuse surfaces. In *11th Annual Conference on Computer Graphics and Interactive Techniques* (1984), SIGGRAPH '84, ACM, pp. 213–222. 3

[GWS04] GÜNTHER J., WALD I., SLUSALLEK P.: Realtime caustics using distributed photon mapping. In *Proceedings of the Fifteenth Eurographics Conference on Rendering Techniques* (2004), EGSR'04, Eurographics Association, pp. 111–121. 2

[HJ09] HACHISUKA T., JENSEN H. W.: Stochastic progressive photon mapping. In *ACM SIGGRAPH Asia 2009 Papers* (2009), SIGGRAPH Asia '09, pp. 141:1–141:8. 2

[HJ10] HACHISUKA T., JENSEN H. W.: Parallel progressive photon mapping on gpus. In *ACM SIGGRAPH ASIA 2010 Sketches* (2010), SA '10, pp. 54:1–54:1. 2

[HOJ08] HACHISUKA T., OGAKI S., JENSEN H. W.: Progressive photon mapping. *ACM Trans. Graph. (TOG) 27*, 5 (2008), 130. 2

[HST13] HE K., SUN J., TANG X.: Guided image filtering. *IEEE Trans. Pattern Anal. Mach. Intell. 35*, 6 (2013), 1397–1409. 2

[Ill] ILLUMINATING ENGINEERING SOCIETY: IES LM-63-1991. http://www.ies.org. Accessed: 2017-02-13. 3

[Jen96] JENSEN H. W.: Global illumination using photon maps. In *Rendering Techniques*. Springer, 1996, pp. 21–30. 2

[Jen01] JENSEN H. W.: *Realistic image synthesis using photon mapping*. AK Peters, Ltd., 2001. 4

[KD13] KAPLANYAN A. S., DACHSBACHER C.: Adaptive progressive photon mapping. *ACM Trans. Graph. 32*, 2 (Apr. 2013), 16:1–16:13. 2

[KK15] KNIEFACZ P., KROPATSCH W.: Smooth and iteratively restore: A simple and fast edge-preserving smoothing model. *arXiv preprint arXiv:1505.06702* (2015). 3

[KSKK10] KURT M., SZIRMAY-KALOS L., KŘIVÁNEK J.: An anisotropic brdf model for fitting and monte carlo rendering. *ACM SIGGRAPH Computer Graphics 44*, 1 (2010), 3. 4

[KW00] KELLER A., WALD I.: *Efficient importance sampling techniques for the photon map*. Universität Kaiserslautern. Fachbereich Informatik, 2000. 2

[LHH*13] LIN W.-C., HUANG T.-S., HO T.-C., CHEN Y.-T., CHUANG J.-H.: Interactive lighting design with hierarchical light representation. *Comput. Graph. Forum 32*, 4 (2013), 133–142. 1

[Lig] LIGHTING ANALYSTS, INC.: AGi32. http://www.agi32.com. Accessed: 2017-02-13. 3

[LTH*13] LUKSCH C., TOBLER R. F., HABEL R., SCHWÄRZLER M., WIMMER M.: Fast light-map computation with virtual polygon lights. In *ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (2013), pp. 87–94. 3

[LTM*14] LUKSCH C., TOBLER R. F., MÜHLBACHER T., SCHWÄRZLER M., WIMMER M.: Real-time rendering of glossy materials with regular sampling. *The Visual Computer 30*, 6-8 (June 2014), 717–727. 3

[LZ14] LIU X.-D., ZHENG C.-W.: Adaptive importance photon shooting technique. *Computers & Graphics 38* (2014), 158–166. 2

[MLM13] MARA M., LUEBKE D., MCGUIRE M.: Toward practical real-time photon mapping: Efficient gpu density estimation. In *ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (2013), pp. 71–78. 2

[NW09] NICHOLS G., WYMAN C.: Multiresolution splatting for indirect illumination. In *2009 symposium on Interactive 3D graphics and games* (2009), pp. 83–90. 4

[PSA*04] PETSCHNIGG G., SZELISKI R., AGRAWALA M., COHEN M., HOPPE H., TOYAMA K.: Digital photography with flash and no-flash image pairs. In *ACM Trans. Graph.* (2004), vol. 23, pp. 664–672. 2

[Rel] RELUX INFORMATIK AG: ReluxDesktop. http://www.relux.com. Accessed: 2017-02-13. 3

[SJ13a] SPENCER B., JONES M. W.: Photon parameterisation for robust relaxation constraints. In *Computer Graphics Forum* (2013), vol. 32, Wiley Online Library, pp. 83–92. 2

[SJ13b] SPENCER B., JONES M. W.: Progressive photon relaxation. *ACM Trans. Graph. Graphics (TOG) 32*, 1 (2013), 7. 2

[SOL*16] SORGER J., ORTNER T., LUKSCH C., SCHWÄRZLER M., GRÖLLER M. E., PIRINGER H.: Litevis: Integrated visualization for simulation-based decision support in lighting design. *IEEE Trans. Vis. Comput. Graphics 22*, 1 (Jan. 2016), 290–299. 1

[Sto] STOCKMAR, AXEL (LIGHT CONSULT INC., BERLIN): EULUMDAT. http://www.helios32.com/Eulumdat.htm. Accessed: 2017-02-13. 3

[SW14] SCHWARZ M., WONKA P.: Procedural design of exterior lighting for buildings with complex constraints. *ACM Trans. Graph. 33*, 5 (Sept. 2014), 166:1–166:16. 1

[TM98] TOMASI C., MANDUCHI R.: Bilateral filtering for gray and color images. In *Computer Vision, 1998. Sixth International Conference on* (1998), IEEE, pp. 839–846. 3, 6

[VRV] VRVIS RESEARCH CENTER: HILITE. http://www.vrvis.at/projects/hilite. Accessed: 2017-02-13. 1, 3, 7

[Wal98] WALTER B.: *Density Estimation Techniques for Global Illumination*. PhD thesis, Cornell University, 1998. 2

[War] WARD G.: Radiance Renderer. http://radsite.lbl.gov/radiance. Accessed: 2017-02-13. 3

[ZSXJ14] ZHANG Q., SHEN X., XU L., JIA J.: Rolling guidance filter. In *Computer Vision–ECCV 2014*. Springer, 2014, pp. 815–830. 3