Journal of Building Performance Simulation

ISSN: 1940-1493 (Print) 1940-1507 (Online) Journal homepage: http://www.tandfonline.com/loi/tbps20

An out-of-core photon mapping approach to daylight coefficients

R. Schregle, L.O. Grobe & S. Wittkopf

To cite this article: R. Schregle, L.O. Grobe & S. Wittkopf (2016): An out-of-core photon mapping approach to daylight coefficients, Journal of Building Performance Simulation, DOI: 10.1080/19401493.2016.1177116

To link to this article: http://dx.doi.org/10.1080/19401493.2016.1177116

Published online: 06 May 2016.

Journal of Building Performance Simulation, 2016
http://dx.doi.org/10.1080/19401493.2016.1177116

Taylor & Francis
Taylor & Francis Group

An out-of-core photon mapping approach to daylight coefficients

R. Schregle*, L.O. Grobe and S. Wittkopf

Lucerne University of Applied Sciences and Arts, Competence Centre Envelopes and Solar Energy, Technikumstrasse 21, CH-6048 Horw, Switzerland

(*Received 17 December 2015; accepted 7 April 2016*)

Climate-based daylight modelling (CBDM) is an effective means of assessing the performance of daylight redirecting components (DRCs) with highly directional scattering to determine their impact on daylight availability and visual comfort. Such a simulation imposes significant computational demands on commodity hardware as it requires high density luminance samples obtained by forward raytracing. We propose an out-of-core photon mapping method within the RADIANCE framework to compute high quality daylight coefficients as a basis for CBDM. The method is particularly suited to angularly selective DRCs exhibiting strong redirection in conjunction with non-uniform sky luminance distributions with high resolution subdivisions. Our implementation is a work in progress and currently accommodates up to 4.3G photons on disk, while optimizing the in-core memory footprint by loading only photons which actually contribute flux to sensor points. We also leverage the fact that photon paths are independent through parallelization.

Keywords: daylight simulation; out-of-core; paging; cache; photon map; Morton code

1. Introduction

The accurate simulation of daylit interiors is essential in assessing a building's daylight performance and energy saving potential. Climate-based daylight modelling (CBDM) with daylight coefficients accumulated over the duration of one year provides a realistic prediction of a building's performance since it accounts for on-site seasonal variations in the sky luminance distribution.

The daylight performance can be improved with daylight redirecting components (DRCs) which modulate the illuminance on a task plane, as well as increase visual comfort and reduce glare due to the luminance distribution within the field of view. DRCs bear characteristics which require precise simulation models for an accurate prediction of their impact on the building's performance; these include angular selectivity, a highly localized distribution of transmitted and reflected light giving rise to caustics and visible shadow patterns.

While a moderate spatial and temporal sampling density of photon paths is sufficient for the analysis of illuminance to assess Daylight Autonomy with DRCs (Heschong et al. 2013; McNeil and Lee 2013), a much higher sampling density is required to resolve non-uniformity and contrast in the luminance distribution (Johnsen, Dubois, and Grau 2006; McNeil 2011; Jakubiec and Reinhart 2012; Rockcastle and Andersen 2014). By extension, the accuracy of simulated luminance maps is highly dependent on the angular sampling density of the sky and the geometric resolution of the DRC model. This of course increases the complexity and computational demands of the simulation.

1.1. Daylight coefficients

Daylight coefficients (DCs) represent normalized contributions from discrete sky luminance zones, or *patches*, to illuminance sensor points (Tregenza and Waters 1983; Mardaljevic 1999). Standardized DC models for dynamic simulations have been proposed and validated (Bourgeois, Reinhart, and Ward 2008; Laouadi, Reinhart, and Bourgeois 2008). The DCs are accumulated and scaled with luminance distributions from a time-series in a postprocess, resulting in the predicted illuminance at the sensor points. This scaling can be repeated for multiple timesteps and their corresponding luminance distributions using the same DCs, provided the geometry is not altered beyond reorienting the building. The accuracy of the simulations obviously depends on the resolution of the underlying sampling used to compute the DCs; our proposed method aims to expedite and improve the quality of this sampling.

To accurately reflect local conditions, the sky patch luminances are derived from on-site illuminance measurements which are then mapped to corresponding patches of a sky model via a subdivision scheme (Tregenza 1987; Perez, Seals, and Michalsky 1993; Wienold and Sprenger 2013). High resolution sky subdivision schemes such as the Reinhart mapping with 2305 patches

*Corresponding author. Emails: roland.schregle@hslu.ch; roland.schregle@gmail.com

are particularly appropriate to reliably predict luminance for visual comfort analysis (Inanici 2013).

Raytracing techniques are expedient in computing DCs as they accurately model the propagation of light within an interior space, and how it is scattered by a DRC. The RADIANCE rendering and lighting simulation system's *rcontrib* tool performs this computation in a single aggregated raytracing pass for all sky patches (Ward et al. 2011), thus establishing an elegant and efficient workflow.

This functionality is now also supported by the RADIANCE photon mapping module via *contribution photons* (Schregle et al. 2015a) to accurately and efficiently compute DCs for highly specular DRCs using forward raytracing. The contribution photon map represents the distribution of normalized flux carried by photons emitted from the sky, which is then locally accumulated per sky patch at sensor points to obtain the DCs. When rendering luminance maps, the sensor points correspond to the primary ray hitpoints.

The primary advantages of photon mapping are twofold: its precomputed flux distribution is view independent; DCs may be recomputed for different luminance map viewpoints from the same photon map without additional forward raytracing. Furthermore, the photon map can resolve caustics (and therefore potential glare) from realistic solar sources subtending an angle $< 1^\circ$ (Schregle et al. 2015a), which are typically undersampled with backward raytracing. These advantages come less to bear when computing illuminance at sensor points, as their number is typically much lower than the number of rays cast in a luminance map.

1.2. *Current limitations of the* RADIANCE *photon map*

The accuracy of an annual simulation using photon mapping relies on three factors: the number of photons traced, the dimensions and complexity of the scene which affect the photon distribution, and the number of sky patches used for DC computation. This implies a very large number of photons to adequately resolve dense sky subdivisions such as Reinhart's, which challenges the memory resources of standard desktop PCs and even high end multicore workstations. A complex annual simulation can easily require a memory footprint of several tens of gigabytes. While a progressive solution using multiple smaller photon maps (Schregle, Grobe, and Wittkopf 2015b) would be conceivable, the advantages of a single bundled forward raytracing step would be mitigated.

The current implementation of the RADIANCE photon map, referred to in this paper as the *iC* photon map, uses a *k*-d tree space subdividing data structure (Bentley 1975) to store and locate photons at nearby sensor points. This data structure is entirely resident in main memory (i.e. *in-core*) for the duration of the simulation, even if only a subset of the photons actually contributes to the sensor points. This is not only wasteful, but can severely degrade performance when physical memory is exhausted and the virtual memory mechanism pages from disk.

1.3. *Advantages of out-of-core photon mapping*

A recently developed variant of the RADIANCE photon map addresses the above-mentioned issues and enables complex annual simulations on commodity hardware, which as of this writing constitutes a typical PC with 8GB of RAM and 4 physical CPU cores. This photon map is entirely resident on disk (i.e. *out-of-core*, or *ooC*) and partially loaded on demand into main memory using a custom caching scheme. Not only does this cater for very large photon maps within commodity memory limits, but also retrieves only those photons that actually contribute flux to sensor points when computing DCs.

These advantages come to bear in the context of CBDM with highly resolved sky models using photon mapping, as they enable handling small sources such as the sun – and the resulting localized flux distribution – at full resolution. This resolution can be leveraged since discretization occurs after the forward raytracing, thus different sky subdivisions can be applied without tracing additional rays.

We chose a sparse octree as underlying data structure due to its straightforward implementation, accelerated construction using a single pass over an ordered photon set, and lower node count compared to binary *k*-d trees. As our primary objective is to enable working with large photon maps within tight memory constraints, we disregard the acceleration effected by the different data structures during photon lookups when computing DCs. Our implementation benefits from multicore CPUs by parallelizing the forward raytracing step and, partially, the octree construction during the in-core photon sorting phase.

1.4. *Paper outline*

The rest of this paper is organized as follows: in Section 2 we cite previous work on which our ooC approach is based; in Section 3 we elaborate our methodology for computing DCs; Section 4 presents results obtained with our implementation for a case study with a DRC characterized by pronounced angular selectivity, including benchmarks comparing the ooC and iC photon map implementations; we conclude in Section 5 with an assessment and an outlook on future developments.

2. Previous work

Among the few validated photon mapping implementations specifically developed for daylight simulation are those included in the VELUX Daylight Visualizer (Foldbjerg et al. 2012) and RADIANCE (Schregle and Wienold 2004). The latter RADIANCE photon map has been recently enhanced to support DCs (Schregle 2015).

Like the RADIANCE framework it is based on, the software emphasizes portability across different platforms.

At the core of any photon mapping implementation lies a space subdividing data structure to efficiently store and retrieve photons near sensor points. Space subdividing data structures have a long history in computing and are fundamental to almost any application which requires managing multidimensional data volumes. Typical representatives of such data structures commonly used for 3D computer graphics include octrees (Meagher 1980), *k*-d trees (Bentley 1975) and bounding volume hierarchies (Goldsmith and Salmon 1987).

Searching within large space subdividing data structures commonly involves imposing an ordering on the data in a space filling curve which linearizes the multidimensional space such that each point corresponds to a unique index on the curve (Asano et al. 1997; Chen and Chang 2005; Haverkort 2011). Our approach uses an octree in conjunction with a *Z-curve*, which enumerates coordinates as a linear *Morton code* (Morton 1966). The efficiency of multidimensional search algorithms in octrees using Morton codes has been investigated in Connor and Kumar (2010) and Behley, Steinhage, and Cremers (2015).

There is a vast body of work on out-of-core data structures for many applications in computing involving large data volumes. In the computer graphics field, this mainly covers processing and rendering point-based and volumetric data using octrees (Kontkanen, Tabellion, and Overbeck 2011; Baert, Lagae, and Dutré 2013; Elseberg, Borrmann, and Nüchter 2013). More recent work focuses on applying out-of-core techniques to accommodate large computational models on GPUs (Zeidan, Nazmy, and Aref 2015).

An early development in applying out-of-core techniques specifically to photon mapping can be found in Christensen and Batali (2004), which uses an hierarchical *brick map* data structure for production rendering. An out-of-core photon mapping technique within large buildings is documented in Fradin, Meneveaux, and Horna (2005), which uses *portals* to track photon transport between adjacent rooms as their geometry is paged in-core. The portal technique has recently been adapted to a distributed out-of-core photon mapping system which runs on a commodity cluster (Günther and Grosch 2014).

3. Methodology

3.1. Overview

Figure 1 gives an overview of our ooC photon mapping workflow with DCs. It consists of the following consecutive steps:

Photon map generation to perform forward raytracing given the scene geometry and sky model, then build the ooC photon map on disk.

DC computation using nearest neighbour (NN) search and photon density estimates at the given sensor points to obtain the DCs for each sky patch.

Luminance scaling to scale the DCs with sky patch luminance distributions for selected timesteps, yielding the illuminance at each sensor point. This step is part of the standard DC workflow, and can be repeated without recomputing the DCs.

Each step is described in detail in the following sections.

3.2. Photon map generation

The contribution photon map is generated with RADIANCE's *mkpmap* tool and represents the indirect flux distribution in the scene, with relative contributions attributed to the individual light sources. To compute DCs, the sky light source has unit luminance, consequently the resulting photon flux is normalized. The ooC octree is built for the photon map and saved to disk to subsequently compute the DCs.

3.2.1. Forward raytracing

The forward raytracer emits photons from each source and simulates their redirection within the DRC and subsequent scattering in the room (see Figure 2). Photons' scattering directions are determined by Monte Carlo sampling of the material's *Bidirectional Scattering Distribution Function*, which quantifies the scattered light depending on the incident and exitant angles to the surface.

Each initial photon hitpoint results in a *photon primary* which contains a reference to its emitting light source as well as the incident direction; the latter is necessary to dynamically identify the emitting sky patch when computing DCs using a sky subdivision scheme (e.g. Tregenza or Reinhart). The photon map can then be reused to recompute DCs with different sky subdivisions.

Primaries are auxiliary to the photon map, and thus stored in a separate array. A primary is multiply referenced by all photons it spawns along its path. As a consequence, the number of primaries is typically much lower than the number of photons, and the former are therefore maintained in-core. We reduce the in-core memory consumption by encoding each primary direction vector as 4 bytes (compared to 12 when stored as floating point vector); this incurs a maximum discretization error of 0.0058°.

Photons are deposited as a result of multiple scattering events along a traced path. These paths are probabilistically terminated via Russian Roulette, depending on the material's reflectance or transmittance. Each photon is stored in an out-of-core unsorted heap file on disk along with its position, normalized flux, and its surface normal (to arbitrate backlighting and occlusion). In addition, the photon

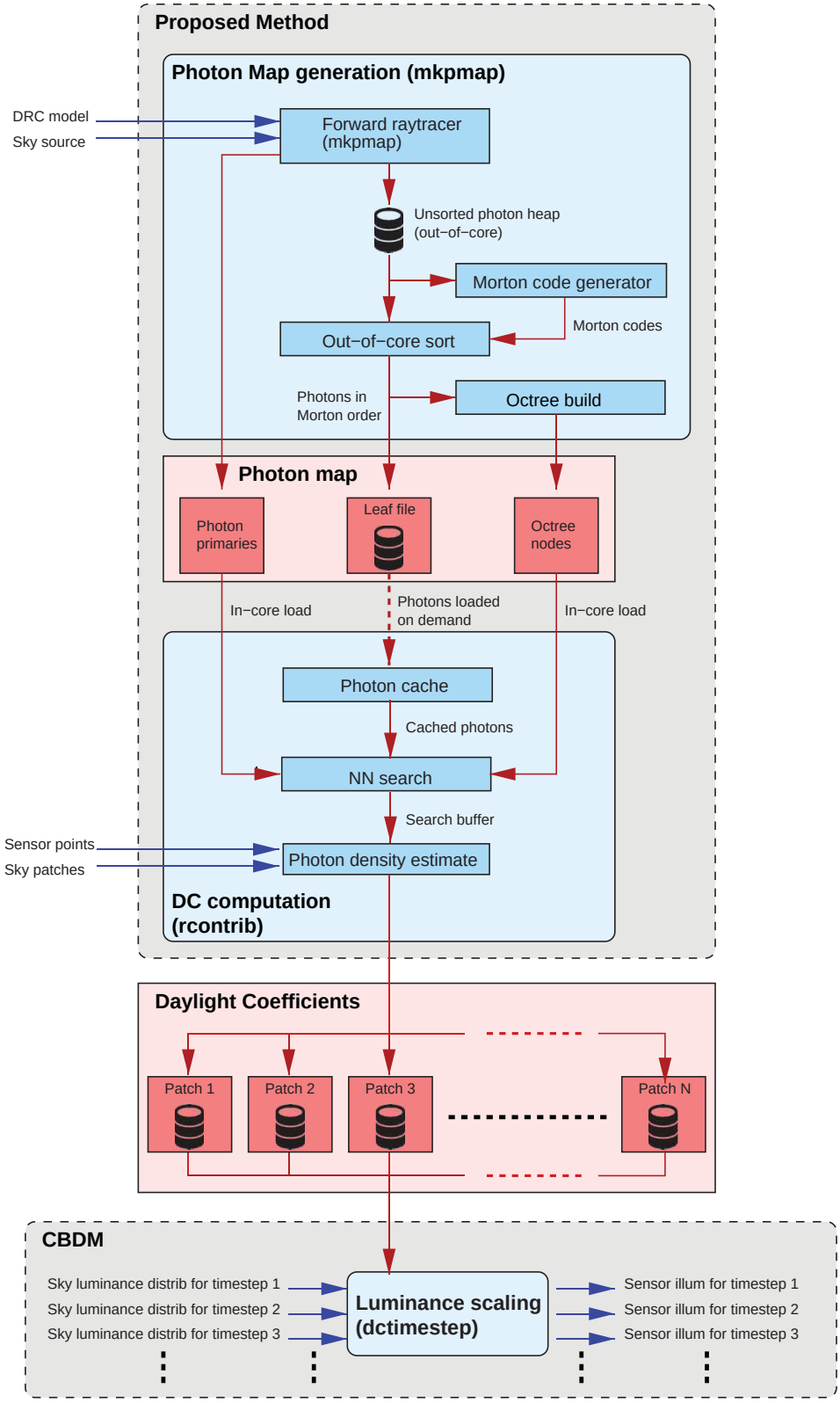

Figure 1. Overview of DC method with ooC photon mapping. The simulation is grouped into three stages: (a) photon map generation with *mkpmap* for a given DRC model and sky source with normalized luminance, (b) computation of DCs from the photon map with *rcontrib* for a set of sensor points and sky patches, and (c) scaling of the DCs with sky luminance distributions per timestep using *dctimestep* to obtain the temporally resolved sensor illuminance.

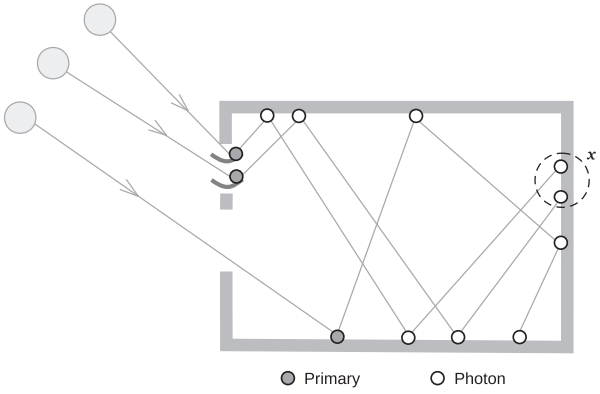

Figure 2. Contribution photon mapping with DRC (mounted in upper fenestration). Photons emitted from each light source are forward raytraced and scattered by the DRC and room geometry via Monte Carlo sampling. Photon primaries are deposited at initial hitpoints and refer to their emitting source, while subsequent hitpoints along the paths spawn photons which reference their corresponding primaries. DCs are evaluated at a sensor point $\boldsymbol{x}$ by locating a number of nearby photons and referencing their primaries to identify their emitting sources. Sky patches are resolved via the primary incident directions.

contains an index to its primary, which in turn identifies its emitting source and incident direction.

We leverage the inherent parallelism in the forward raytracing step due to the fact that photon paths are independent. The number of photons to trace is evenly distributed among multiple processes, which atomically write to the same heap file. I/O contention is reduced by locally buffering the photons generated by each process, and flushing these regularly. The buffers are randomly sized for each process to temporally decorrelate the flushing events.

3.2.2. *Out-of-core sort and octree leaf file construction*

Once forward raytracing is complete, the ooC octree is built. The octree is necessary to efficiently locate photons near sensor points to evaluate DCs.

In preparation to octree construction, the photons in the unsorted heap file are externally sorted according their *Morton codes* (see next section). External sorting algorithms for large data sets are a well studied area in computer science. We employ an external mergesort (Knuth 1998; Seyedafsari and Hasanzadeh 2010) to enumerate the photons in Morton order. This involves recursively subdividing the unordered heap file into smaller subblocks until their size falls below a threshold which can be quicksorted in-core and in parallel. The sorted blocks are written out to separate temporary files which are then merged into progressively larger blocks as recursion unwinds.

3.2.3. *Morton code generation*

The Morton code represents a linearization of the photons' coordinates within the scene geometry. It does so by mapping a photon's 3D Cartesian floating point coordinates $\boldsymbol{p} = [x, y, z]$ to a scalar integer $f_{M,i}(\boldsymbol{p})$, where i denotes the resolution of the code in number of bits per dimension. A $3i$-bit Morton code is thus defined as:

$$f_{M,i}(\boldsymbol{p}) : \mathbb{R}^3 \to \mathbb{N},$$
$$[x, y, z] \to (Z_{i-1}Y_{i-1}X_{i-1} \cdots Z_0Y_0X_0)_2, \quad (1)$$
$$[X, Y, Z] = \left\lfloor \frac{2^i - 1}{s}(\boldsymbol{p} - \boldsymbol{o}) \right\rfloor. \quad (2)$$

Here $\boldsymbol{p}$ is mapped to a vector $[X, Y, Z]$ of i-bit integers. This requires normalizing by the size s of the axis-aligned bounding box containing the entire photon set. Note that we subtract its origin $\boldsymbol{o}$, which can be arbitrary in a RADIANCE scene description. The bits comprising $[X, Y, Z]$ are then 3-way interleaved, denoted here as the binary representation $(Z_{i-1}Y_{i-1}X_{i-1} \cdots Z_0Y_0X_0)_2$. Note that this mapping incurs a discretization error proportional to $2^{-i}s$ per dimension.

Graphically, the Morton code corresponds to a Z-curve which recursively traverses all octants of an octree of up to i levels (see Figure 3 for the first two levels). Consequently this specifies the order in which photons are encountered at the octree leaves during a traversal, and the file containing the sorted photons becomes the octree's *leaf file*. Of course, depending on the photon distribution, only a subset of the 2^{3i} possible Morton codes will actually be assigned. Note that we do not explicitly store Morton codes with the photons, but generate these on the fly using an optimized bitmask method (Baert 2013) instead of bitwise iteration.

Our implementation currently uses 63-bit Morton codes with $i = 21$ bits per dimension, which provides sufficient resolution for the photon maps we have tested. Based on the leaf file, we generate the octree nodes to index the photons as described in the next section. The leaf file forms the bulk of the photon map's data volume and remains consistently out-of-core during DC computation.

3.2.4. *Octree node construction*

The octree nodes are generated from the leaf file using the efficient single pass algorithm described in Kontkanen, Tabellion, and Overbeck (2011). With the photons in Morton order, this algorithm obviates a top-down insertion from the octree root for each photon, which requires multiple traversals. Instead, the nodes are constructed bottom-up from the leaves, and visited only once.

The octree nodes and primaries are saved to a file which is loaded for the DC computation, and therefore separate from the leaf file. The octree nodes specifically serve to index the photons and are maintained in-core to facilitate photon retrieval.

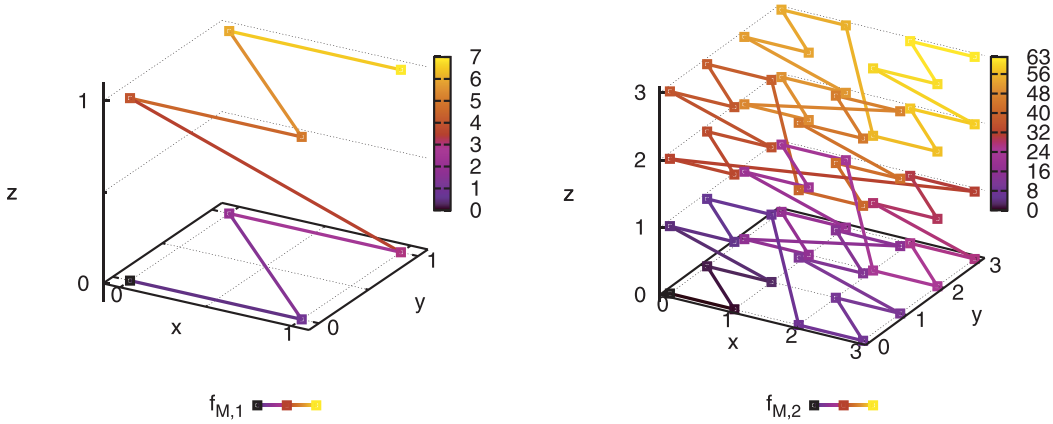

Figure 3. Space subdividing Z-curve at levels 1 (left) and 2 (right). At level i, the curve subdivides the space into 8^i suboctants. By traversing the suboctants in a fixed Z-order pattern, the curve linearizes their 3D Cartesian coordinates into discrete scalar indices. Each index corresponds to a suboctant's position along the length of the curve, and is referred to as its Morton code $f_{M,i}$.

3.3. *DC computation*

The DCs are evaluated for a set of n_s sensor points from a set of n_l sky patches using the normalized flux distribution in the previously computed photon map. The sky light source and mapping to enumerated patches is passed to RADIANCE's *rcontrib* tool. In principle, linking the photon paths to the sensor points results in a bidirectional raytracer, since the rays originate at the sensors and terminate at the sky patches (via the photon primaries); the raytracing solution is still consistent as the photon flux is invariant towards ray reversal.

3.3.1. *Photon cache*

Photons are loaded on demand from the leaf file when computing DCs. The goal of minimizing disk access and I/O latency is critical for the performance of any out-of-core implementation. Towards this end, we employ a customized photon cache which retains a subset of the photons in-core. A lookup table of in-core pages containing the photons is maintained using standard hashing techniques.

A constant load is maintained by evicting pages using an LRU (least recently used) page replacement policy (Tanenbaum and Bos 2014). The cache is optimized to bypass page table lookups and all bookkeeping if the same page is accessed in repetition; we observed this to be the case for over 90% of the photons accessed in our benchmarks. The size of the cache can be adjusted to the available physical memory.

When DC computation is parallelized, processes contend for simultaneous access to the octree leaf file, which would further aggravate I/O latency. For this reason, each process maintains a private cache instance.

3.3.2. *NN search*

To evaluate the DCs contributed by each sky patch to a sensor point $\boldsymbol{x}$, we locate the k nearest photons around $\boldsymbol{x}$ using an NN search in the octree (see Figure 2). The search algorithm consists of an octree traversal similar to that detailed in Behley, Steinhage, and Cremers (2015), which uses the in-core octree nodes to index the photons in the leaf file via the above-mentioned cache.

Photon indexing into the leaf file is implicit in our implementation. Each octree node contains a counter quantifying the number of photons contained in its subtree. This corresponds to the relative offset within the leaf file to the first photon contained in the node's next sibling subtree in Morton order. When traversing the octree, subtrees whose bounding boxes lie outside the search radius (corresponding to the distance of the furthest photon in the lookup) are pruned. By summing the photon counters of pruned subtrees until a leaf is reached, we obtain the absolute file position of the first photon within the leaf. With the 32-bit photon counters per node used in our current implementation, the octree can address a maximum of $2^{32} \approx 4.3$G photons.

The NN search passes the found photons for evaluation by the photon density estimate (see next section). With the iC photon map, these are returned as in-core references. In an out-of-core context, returning external references as indices into the leaf file is undesirable as this may incur additional disk activity if the corresponding pages are no longer cached when the photons are evaluated. Our implementation therefore redundantly stores the found photons in-core in a search buffer, whence they are immediately accessible without additional I/O.

3.3.3. *Contribution photon density estimate*

Once the k closest photons are found around a sensor point $\boldsymbol{x}$, its associated DCs are evaluated using a modified photon density estimate (Wann Jensen 2001).

Each photon is characterized by its normalized flux ϕ_p, its emitting sky patch l_p (identified via its primary's incident direction), and its hitpoint $\boldsymbol{x}_p$. The flux is accumulated

and tabulated according to l_p, and divided by the surface area containing the photons. This area is assumed to be planar within a radius r around $\boldsymbol{x}$, and that all patches contribute a roughly equal number of photons, hence the density estimate is approximate. Because the photon flux is normalized, the density estimate directly yields the DCs per contributing sky patch.

With the contribution density estimate, the coefficient c_l for sky patch l at sensor point $\boldsymbol{x}$ can be expressed as:

$$c_l(\boldsymbol{x}) \approx \sum_{i=1}^{k} K(\|\boldsymbol{x},\boldsymbol{x}_{p,i}\|)\frac{\phi_{p,i}}{\pi r^2} \quad \forall i : l_{p,i} = l, \quad \|\boldsymbol{x},\boldsymbol{x}_{p,i}\| \leq r, \tag{3}$$

where K is a normalized weighting function which acts as a rotationally symmetric filter based on a photon's distance to $\boldsymbol{x}$. The number of photons k contributing to the density estimate is the governing parameter and termed the *bandwidth*, as it defines the support of the filter K. It should be noted that density estimates can incur objectionable bias if the bandwidth is excessively high (Schregle 2003; Hernandez et al. 2014), resulting in visible blurring and a local underestimation of DCs in boundary regions.

3.4. Luminance scaling of DCs

As final step in the standard DC method, the DCs are scaled with on-site sky luminance distributions for individual timesteps (Mardaljevic 1999). In RADIANCE this is performed with the *dctimestep* tool.

The output of the DC computation can be represented as an $n_l \times n_s$ matrix $\mathbf{D}$ consisting of coefficients per sky patch contributed to each sensor point. In the standard DC formulation, this is scaled with an n_l-dimensional sky patch luminance vector $\boldsymbol{L}(t)$ to obtain the n_s-dimensional vector $\boldsymbol{E}(t)$ quantifying the illuminance for each sensor point at timestep t:

$$\boldsymbol{E}(t) = \mathbf{D} \cdot \boldsymbol{L}(t). \tag{4}$$

This scaling operation can be repeated for different timesteps t and corresponding luminance distributions $\boldsymbol{L}(t)$ without recomputing the DC matrix $\mathbf{D}$. We omit the solid angle per sky patch included in the original DC formulation as this geometric term is implicit in our approach.

During forward raytracing, photons are emitted from the exterior fenestration surface towards the interior, where they are scattered by the DRC. This requires the user to specify the external fenestration as a *photon port*. The photons' origins are randomly distributed over the fenestration area (which is discretized to reduce clustering), and their incident directions are similarly distributed over the sky's hemisphere. Note that these directions are not discretized into patches at this stage. A visibility test handles exterior obstructions as well as shadows cast by the building itself.

Photon incident directions are discretized when computing the DCs with the contribution photon density estimate (Equation (3)). The patch solid angle is implied here as it is proportional to the fraction of the k found photons that are mapped to each patch.

4. Results and discussion

We demonstrate the application of our method to a case study involving the retroreflecting DRC shown in Figure 4. The system is oriented in a forward and reversed configuration to effect retroreflection in the lower fenestration (ca. 1–1.75 m height) and redirection in the upper fenestration (ca. 1.75–2.75 m height) via an integrated lightshelf (Aydınlı et al. 2015; Köster 2015).

Our case study is conducted in a fictitious 6 × 6 m room with a CIE clear sky. The DRC is mounted within a large south-facing double glazing which spans the entire width of the room. The reflectances of the floor, walls, and ceiling were 20%, 50% and 70%, respectively. The results were obtained on an 8-core Intel Xeon E5-2660/2.6 GHz system with 8 GB RAM and a 6TB HDD.

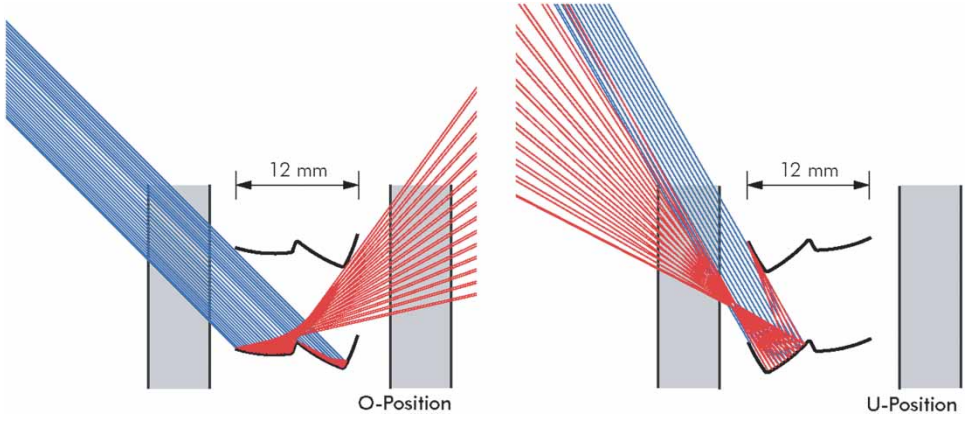

Figure 4. RETROLuxTherm lamella, designed and patented by Helmut Köster. The upper fenestration redirects light incident from left towards the ceiling (O-position, left), while the lower portion retroreflects (U-position, right); note the lamella is simply reversed for the latter. Reprinted with permission from Köster (2015).

4.1. Benchmarks

We conducted a series of benchmarks with the case study using a solar source (no sky) to compare the performance of the ooC photon map to the original iC implementation, as well as to assess the performance of the ooC cache and multiprocessing scalability.

4.1.1. Out-of-core vs. in-core photon map

The results of a benchmark comparing the performance of the ooC photon map with the original iC implementation are shown in Figure 5. The simulation parameters used for the benchmark are summarized in Appendix 1.

The left graph plots the number of photons against the time to generate the photon map using *mkpmap* with 8 parallel processes. While the ooC photon map clearly shows a strictly linear growth with the number of photons, the iC performance degrades significantly beyond 200M photons due to excessive paging. This can be partly attributed to incoherent access patterns while balancing the *k*-d tree to guarantee logarithmic search times. The iC photon map failed to generate more than 500M photons after exhausting the 16 GB swap space allocated on our test system. As the benchmark indicates a practical limit of 200M photons with iC, there is no benefit from using ooC for smaller photon maps.

The right graph in Figure 5 plots the number of photons against the time to render a luminance map using *rpict* (note that *rpict* lacks direct support for multiprocessing). Photon density estimates were performed at the primary ray hitpoints for each pixel, resulting in coherent access patterns and an almost 100% cache hit ratio with the ooC photon map.

It is evident from the graph that the ooC photon map benefits from the use of a cache, resulting in an almost 3-fold speedup. With suitable parameters (see values for the cache size and page size in Table A1), cached ooC outperforms iC by up to 30% at 500M photons, further supporting the effectiveness of the cache.

4.1.2. Out-of-core photon cache

Cache performance is critical within a small memory footprint, and the previous results with 8 GB RAM were obtained with a suitable cache page size. The graph in Figure 6 shows the effect of this parameter for cache sizes of 10k and 100M photons with *rpict*. The remaining parameters coincide with the previous benchmark, with the photon map size and lookup bandwidth fixed at 1G and 2800 photons, respectively.

The cache page size sets the granularity of the cache and is expressed as multiple of the bandwidth, thus adapting it to the data volume traversed during NN search, under the assumption that the majority of the inspected photons will then reside in the same page. This effect is evident in the graph, as performance rises with larger page sizes until 16 (corresponding to ca. 45,000 photons). Performance plateaus before dropping slightly again beyond this value with smaller caches due to the higher I/O latency incurred by loading larger pages.

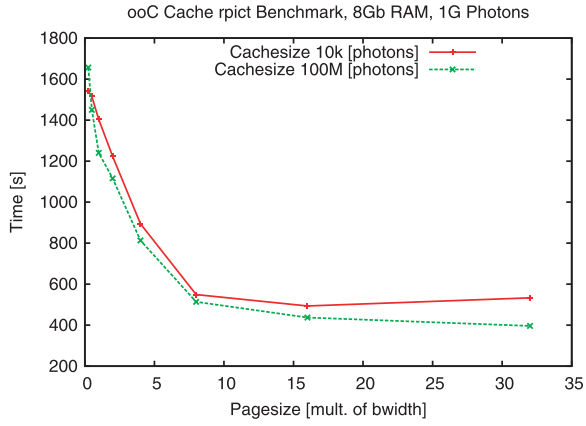

Figure 6. ooC photon cache performance vs. page size for lookups in a 1G photon map with *rpict*. Note the page size is specified as multiple of the lookup bandwidth, in this case 2800 photons.

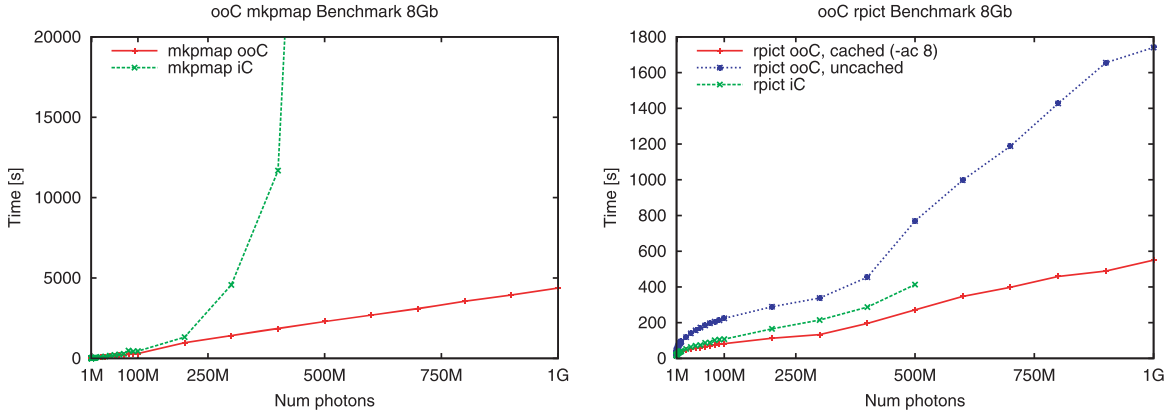

Figure 5. Benchmark results of forward raytracing with *mkpmap* (left) and rendering with *rpict* (right). Note that iC failed to generate more than 500M photons due to memory starvation.

The graph further suggests that the influence of the overall cache size, manifested in the similar curves, is infact quite marginal, and that this parameter can therefore be adapted to the available physical memory with minor penalty.

4.1.3. Out-of-core parallelization

In a further benchmark, we investigated the scalability of the ooC photon map as a function of parallel processes. The graph in Figure 7 plots performance as a function of photon map size and number of processes. Diminishing returns are evident during photon map generation with *mkpmap* (left graph) and rendering with *rtrace* (right graph, see Appendix 2 for parameters).

Sublinear scalability is expected for any parallelized application containing a serial component (Amdahl 1967), as is the case in the merging step during Morton order sorting performed by *mkpmap*. In addition, the probability of simultaneous cache misses during rendering, however rare, increases with the number of processes. This in turn aggravates the impact of contention among the

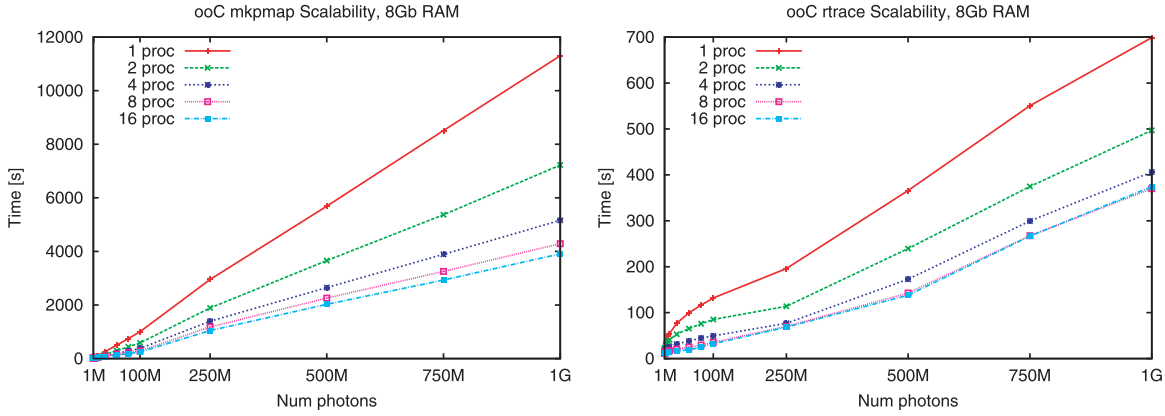

Figure 7. Parallel scalability of ooC photon map generation with *mkpmap* (left) and rendering with *rtrace* (right) as a function of photon map size and number of parallel processes.

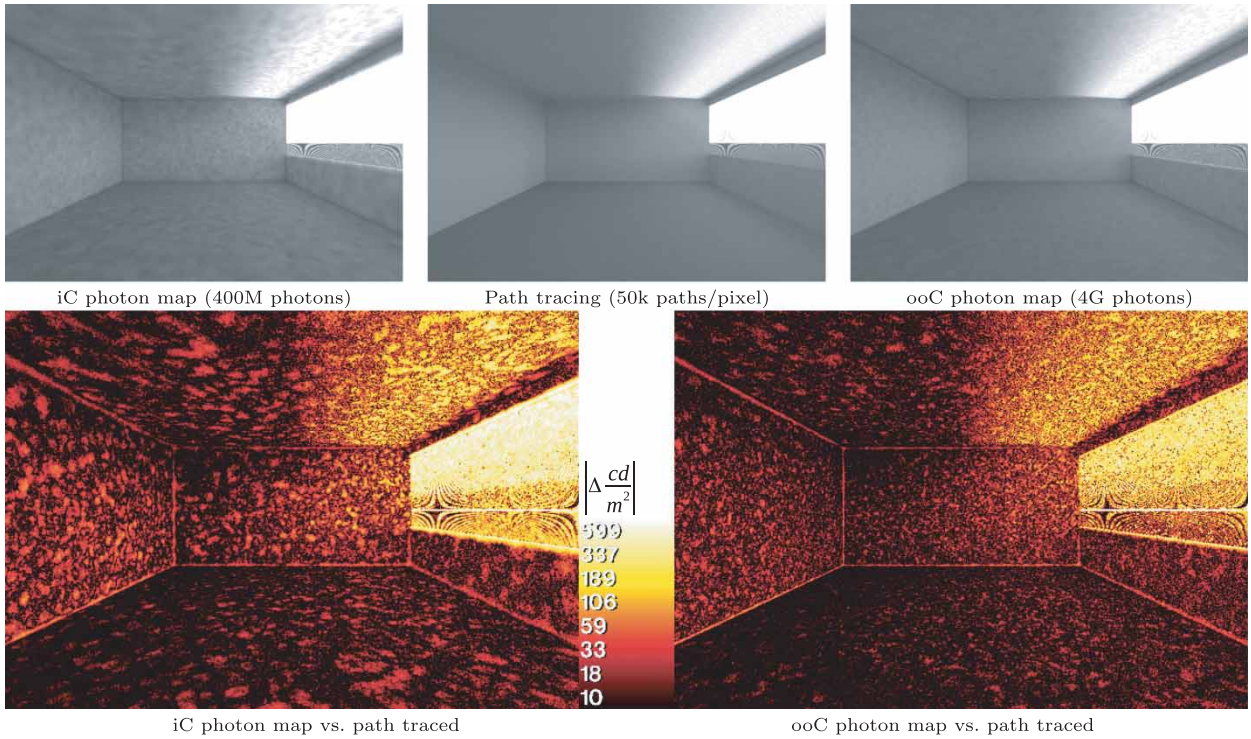

Figure 8. Luminance maps for June 1st, 12:30 pm in Lucerne from an annual simulation using 2305 Reinhart sky patches. The fenestration contains the DRC in Figure 4. Daylight Coefficients were generated with path-tracing using standard RADIANCE (upper centre), 400M iC photon map (upper left), and 4G ooC photon map (upper right). The absolute deviations between each photon map rendering and the reference are shown in the lower row.

processes when paging from the leaf file as a shared resource.

4.2. Annual simulation luminance maps

Figure 8 shows luminance maps rendered with 1200 × 835 pixels for an exemplary timestep from an annual simulation of our case study. The DCs were computed for a Reinhart sky with 2305 patches at the primary ray hitpoints from the photon density estimates. These were scaled by *dctimestep* with a patch luminance distribution generated with *gensky* for 12:30 pm on June 1st in Lucerne, Switzerland. In this timestep, the DRC retroreflects in the lower fenestration (hence no direct transmission), while the upper fenestration redirects towards the ceiling. Details on the parameters to generate the luminance maps can be found in Appendix 3.

In the upper row in Figure 8, we compare images generated with the iC and ooC photon maps to a path-traced reference generated with RADIANCE's standard *rcontrib* tool with 50,000 paths. The iC photon map contained 400M photons, corresponding to the limit imposed by the 8 GB RAM available, while the ooC photon map used 4G photons. The ooC photon map renders the luminance map with comparable quality to the reference in a fraction of the time. In particular, the caustics from redirection on the ceiling are similar, while those from the lower resolution iC photon map exhibit more obtrusive noise.

The lower row in Figure 8 shows falsecolour differences between each photon map rendering and the path traced reference. The images quantify the absolute deviation from the reference in cd/m^2, with RMS values of 175 and 91 cd/m^2 for the iC and ooC photon maps, respectively. The photon map renderings are characterized by low frequency noise typical for density estimation, which contrasts with the reference's higher frequency noise. Both photon mapped images also exhibit typical (albeit minor, due to a carefully chosen bandwidth) boundary bias at the polygon edges and corners, but otherwise agree well with the reference.

5. Conclusion and outlook

We have presented an out-of-core implementation of the RADIANCE photon mapping module to efficiently handle large photon maps to compute high quality DCs on commodity hardware in support of annual daylight simulations. The implementation supports multiprocessing, and its performance can be adjusted to the available physical memory via an in-core photon cache.

We demonstrated our method for a case study with a representative DRC characterized by retroreflection and redirection. The benchmarks conducted with this case study on a PC with 8 GB RAM reveal that iC photon map generation times rise exponentially due to excessive swapping, while ooC exhibits linear growth and even performs DC calculation up to 30% faster than iC.

The multiprocessing scalability is however limited by I/O contention and the portion of serialized code inherent in the RADIANCE system. This is to some extent dependent on the hardware and warrants further testing on other platforms. We also plan to test the performance with solid state disks (SSDs), although their effect is expected to be less pronounced with the photon cache.

This is a work in progress; viable future developments for the ooC photon map would include more extensive testing with different DRC types to study the impact of the photon distribution on the performance. The current capacity of 4.3G photons could be extended by using 64-bit photon counters in the octree nodes. Alternatively, a more compact solution with a two-level octree hierarchy could be used, whereby nodes refer to secondary octrees. Both strategies could then benefit from the larger physical memory of the next generation of commodity hardware, since this would accommodate a larger cache and thus improve performance.

Reciprocally, two strategies to reduce the photon count could be investigated. The first could limit octree node subdivision by merging photons where the density is unnecessarily high; such a heuristic would be based on the node's bounding box size. The second strategy could concentrate photon emission along sun paths defined a priori (assuming a fixed building location and orientation), and could then leverage the temporal coherence in the resulting flux distribution, for example, using motion vectors. In combination, these developments would further amplify the achievable complexity in an annual simulation with photon mapping.

The ooC photon mapping software will be made available as open source as part of the RADIANCE CVS repository.[1]

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This work was supported by the Swiss National Science Foundation as part of the project 'Simulation-based assessment of daylight redirecting components for energy savings in office buildings' under grant SNSF #147053.

Note

1. http://radiance-online.org/download-install/radiance-source-code

References

Amdahl, G. M. 1967. "Validity of the Single Processor Approach to Achieving Large Scale Computing Capabilities." AFIPS conference proceedings, Atlantic City, NJ, 483–485.

Asano, T., D. Ranjan, T. Roos, E. Welzl, and P. Widmayer. 1997. "Space-Filling Curves and Their Use in the Design of

Geometric Data Structures." *Theoretical Computer Science* 181 (1): 3–15.

Aydınlı, S., S. Gramm, H. Kaase, and H. Köster. 2015. "Einbindung tageslichttechnischer Messwerte in Planungsprogramme am Beispiel eines neuartigen Sonnenschutzsystems." *Bauphysik* 37 (5): 257–262.

Baert, Jeroen and Lagae Ares and Dutré Philip. 2014 "Out-of-Core Construction of Sparse Voxel Octrees". *Computer Graphics Forum* 33 (6): 220–227.

Baert, J., A. Lagae, and P. Dutré. 2013. "Out-of-Core Construction of Sparse Voxel Octrees." Proceedings of the 5th high-performance graphics conference, 27–32. HPG '13, Anaheim, California New York, NY: ACM.

Behley, J., V. Steinhage, and A. B. Cremers. 2015. "Efficient Radius Neighbor Search in Three-Dimensional Point Clouds." IEEE international conference on robotics and automation, ICRA 2015, 3625–3630. Seattle, WA, 26–30 May, 2015.

Bentley, J. L. 1975. "Multidimensional Binary Search Trees Used for Associative Searching." *Communications of the ACM* 18 (9): 509–517.

Bourgeois, D., C. F. Reinhart, and G. Ward. 2008. "Standard Daylight Coefficient Model for Dynamic Daylighting Simulations." *Building Research & Information* 36: 68–82.

Chen, H.-L., and Y.-I. Chang. 2005. "Neighbor-Finding based on Space-Filling Curves." *Information Systems* 30 (3): 205–226.

Christensen, P. H., and D. Batali. 2004. "An Irradiance Atlas for Global Illumination in Complex Production Scenes." Eurographics symposium on rendering (2004), 133–141, Eurographics Association.

Connor, M., and P. Kumar. 2010. "Fast Construction of *k*-Nearest Neighbour Graphs for Point Clouds." *IEEE Transactions on Visualisation and Computer Graphics* 16 (4): 599–608.

Elseberg, J., D. Borrmann, and A. Nüchter. 2013. "One Billion Points in the Cloud – an Octree for Efficient Processing of 3D Laser Scans." *ISPRS Journal of Photogrammetry and Remote Sensing* 76: 76–88.

Foldbjerg, Peter, Thorbjørn Færing Asmussen, Nicolas Roy, Per Sahlin, Lars Ålenius, Henrik Wann Jensen, and Claus Jensen. 2012. "Daylight Visualizer and Energy & Indoor Climate Visualizer, a Suite of Simulation Tools for Residential Buildings." Proceedings of the building simulation and optimization conference BSO12, Loughborough, September.

Fradin, D., D. Meneveaux, and S. Horna. 2005. "Out of Core Photon-Mapping for Large Buildings." In *Eurographics Symposium on Rendering (2005)*, edited by K. Bala and P. Dutré. The Eurographics Association. https://pdfs.semanticscholar.org/a58c/67e0b93d4b06e4c3a79fef1bc7c41656ffcb.pdf.

Goldsmith, J., and J. Salmon. 1987. "Automatic Creation of Object Hierarchies for Ray Tracing." *IEEE Computer Graphics and Applications* 7 (5): 14–20.

Günther, T., and T. Grosch. 2014. "Distributed Out-of-Core Stochastic Progressive Photon Mapping." *Computer Graphics Forum* 33 (6): 154–166.

Haverkort, H. J. 2011. "An Inventory of Three-Dimensional Hilbert Space-Filling Curves." *CoRR*, abs/1109.2323.

Hernandez, R. J. G., C. Urena, J. Poch, and M. Sbert. 2014. "Overestimation and Underestimation Biases in Photon Mapping with Non-Constant Kernels." *IEEE Transactions on Visualization and Computer Graphics* 20 (10): 1–1.

Heschong, Lisa, Van Den Wymelenberg, Keven Marilyne Andersen, N. Digert, L. Fernandes, A. Keller, J. Loveland, H. McKay, and R. Mistrick. 2013. *Approved Method: IES Spatial Daylight Autonomy (sDA) and Annual Sunlight Exposure (ASE)*. New York: Illuminating Engineering Society of North America. ISBN: 978-0-87995-272-3.

Inanici, M. 2013. "Dynamic Daylighting Simulations from Static High Dynamic Range Imagery Using Extrapolation and Daylight Coefficient Methodologies." Proceedings of 13th conference of international building performance simulation association, Chambéry, France, 3392–3399.

Jakubiec, J. A., and C. F. Reinhart. 2012. "The 'Adaptive Zone' – a Concept for Assessing Discomfort Glare Throughout Daylit Spaces." *Lighting Research and Technology* 44 (2): 149–170.

Johnsen, K., M. C. Dubois, and K. Grau. 2006. *Assessment of Daylight Quality in Simple Rooms: Impact of Three Window Configurations on Daylight Conditions, Phase 2*. Hørsholm, Denmark: Danish Building Research Institute.

Knuth, D. E. 1998. *The Art of Computer Programming, Volume 3: Sorting and Searching*. Redwood City, CA: Addison Wesley Longman.

Kontkanen, J., E. Tabellion, and R. S. Overbeck. 2011. "Coherent Out-of-Core Point-Based Global Illumination." *Computer Graphics Forum* 30 (4): 1353–1360.

Köster, H. 2015. *Daylight Modulation*. Frankfurt: WITAG-Verlag. ISBN: 978-3-00-048400-1.

Laouadi, A., C. Reinhart, and D. Bourgeois. 2008. "Efficient Calculation of Daylight Coefficients for Rooms with Dissimilar Complex Fenestration Systems." *Journal of Building Performance Simulation* 1 (1): 3–15.

Mardaljevic, J. 1999. "Daylight Simulation: Validation, Sky Models and Daylight Coefficients." PhD diss., De Montfort University, Leicester.

McNeil, A. 2011. "On the Sensitivity of Daylight Simulations to the Resolution of the Hemispherical Basis Used to Define Bidirectional Scattering Distribution Functions." Technical report DOE/LBNL FY11, Lawrence Berkeley National Laboratory.

McNeil, A., and E. S. Lee. 2013. "A Validation of the Radiance Three-Phase Simulation Method for Modeling Annual Daylight Performance of Optically-Complex Fenestration Systems." *Journal of Building Performance Simulation* 6 (1): 24–37.

Meagher, D. 1980. "Octree Encoding: A New Technique for the Representation, Manipulation and Display of Arbitrary 3-D Objects by Computer." Technical report IPL-TR-80-111, Image Processing Laboratory, Renssalaer Polytechnic Institute.

Morton, G. 1966. "A Computer Oriented Geodetic Data Base; and a New Technique for File Sequencing." Technical report, IBM Ltd., Ottawa, Canada.

Perez, R., R. Seals, and J. Michalsky. 1993. "All-Weather Model for Sky Luminance Distribution – Preliminary Configuration and Validation." *Solar Energy* 50 (3): 235–245.

Rockcastle, S., and M. Andersen. 2014. "Measuring the Dynamics of Contrast & Daylight Variability in Architecture: A Proof-of-Concept Methodology." *Building and Environment* 81: 320–333.

Schregle, R. 2003. "Bias Compensation for Photon Maps." *Computer Graphics Forum* 22 (4): 729–742.

Schregle, R. 2015. "Development and Integration of the RADIANCE Photon Map Extension." Technical report, Lucerne, Switzerland: Lucerne University of Applied Sciences and Arts.

Schregle, R., C. Bauer, L. O. Grobe, and S. Wittkopf. 2015a. "EvalDRC: A Tool for Annual Characterisation of Daylight Redirecting Components with Photon Mapping." Proceedings CISBAT 2015, Lausanne, Switzerland.

Schregle, R., L. O. Grobe, and S. Wittkopf. 2015b. "Progressive Photon Mapping for Daylight Redirecting Components." *Solar Energy* 114: 327–336.
Schregle, R., and J. Wienold. 2004. "Physical Validation of Global Illumination Methods: Measurement and Error Analysis." *Computer Graphics Forum* 23 (4): 761–781.
Seyedafsari, M. H., and I. Hasanzadeh. 2010. "Optimal External Merge Sorting Algorithm with Smart Block Merging." *International Journal of Computer, Electrical, Automation, Control and Information Engineering* 4 (2): 237–240.
Tanenbaum, A. S., and H. Bos. 2014. *Modern Operating Systems*. 4th ed. Upper Saddle River, NJ: Prentice Hall Press.
Tregenza, P. R. 1987. "Subdivision of the Sky Hemisphere for Luminance Measurements." *Lighting Research & Technology* 19 (1): 13–14.
Tregenza, P. R., and I. M. Waters. 1983. "Daylight Coefficients." *Lighting Research & Technology* 15 (2): 65–71.
Wann Jensen, H. 2001. *Realistic Image Synthesis Using Photon Mapping*. Natick, MA: A. K. Peters.
Ward, Greg, Richard Mistrick, Eleanor S. Lee, Andrew McNeil, and Jacob Jonsson. 2011. "Simulating the Daylight Performance of Complex Fenestration Systems Using Bidirectional Scattering Distribution Functions within Radiance." Technical report LBNL-4414E, Lawrence Berkeley National Laboratory.
Wienold, J., and W. Sprenger. 2013. "Gendaylit 2.3." Presented at the 12th international RADIANCE workshop, Golden, CO.
Zeidan, M., T. Nazmy, and M. Aref. 2015. "GPU-based Out-of-Core HLBVH Construction." In *Proceedings Eurographics Symposium on Rendering*, edited by J. Lehtinen and D. Nowrouzezahrai. Darmstadt, Germany: The Eurographics Association.

Appendix 1. Out-of-core vs. in-core benchmark parameters

Table A1 lists the *mkpmap* and *rpict* parameters used for the benchmarks in Section 4.1.1. The photon density estimate bandwidth k was proportional to the number of photons to limit bias (ranging from $k = 9$ at 10^4 photons, to $k \approx 2800$ at 10^9 photons). The ooC cache size (− aC) was similarly adjusted in the range $[100 \cdots 10^7]$ photons, organized into pages of $8k$ photons (− ac), thus adapting the cache granularity to the data volume traversed during the NN search. By setting the ambient bounces (− ab) to −1 we visualize photons directly at the primary rays when rendering luminance maps, thus obtaining coherent photon map lookups.

Table A1. Benchmark simulation parameters.

Command	Parameter	Description	Value
mkpmap	− ap *pmap* N_p	Photon map file *pmap*, target photon count N_p	$N_p = [10^4 \cdots 10^9]$
rpict	− ap *pmap* k	Photon map file *pmap*, lookup bandwidth k (in photons)	$k = \lfloor\sqrt{0.008 N_p}\rfloor = [9 \cdots 2828]$
	− x, − y	Resolution in pixels	800, 556
	− ab	Ambient bounces	−1
	− aC	Num cached photons	$0.01 N_p = [100 \cdots 10^7]$
	− ac	Cache page size (in photons) as multiple of k	8

Appendix 2. Out-of-core parallelization benchmark parameters

The multiprocessing scalability benchmark for a 1G photon map in Section 4.1.3 was performed with *rtrace*, since *rpict* lacks native multiprocessing support. Parallel rendering was achieved with the −*n* option and passing primary (view) rays to *rtrace* with *vwrays*:

```
vwrays -vf viewfile -x 800 -y 800 | rtrace -x 800 -y 556 -ld- -fac -ov -n nproc -ac 8
-aC cachesize -ap pmap k -ab -1 octree
```

See Table A1 for cache size and photon lookup bandwidth (k) values.

Appendix 3. Annual simulation luminance map parameters

The DCs for the reference luminance map in Section 4.2 were obtained by generating 50,000 primary rays per pixel with *vwrays* and path tracing these with *rcontrib* (by spawning a single ambient ray with − ad 1):

```
vwrays -vf viewfile -x 1200 -y 1200 -pj 0.1 -ff -c 50000 |
rcontrib -x 1200 -y 835 -ld- -ffc -n 20 -c 50000 -ab 8 -ad 1 -e MF:4 -f reinhart.cal
-b rbin -bn Nrbins -fo -o reinhart-%04. hdr -m skyglow octree
```

Here *rcontrib* bins the DCs from the sky source `skyglow` according to a Reinhart MF:4 sky subdivision (2305 patches) and saves these in RADIANCE HDR picture format to separate files `reinhart-`*nnnn*`.hdr` for each patch number *nnnn*. The DCs for the iC and ooC photon maps where obtained in similar fashion with photon lookup bandwidths according to Table A1.

A sample luminance map was then obtained from the DCs for June 1st at 12: 30 pm in Lucerne, Switzerland (approximate latitude 47°N, longitude 8°E) as follows:

```
gendaylit 6 1 12.5 -m -30 -G 838 157 -a 47 -o -8 | genskyvec | dctimestep
reinhart-%04.hdr
```

Here *gendaylit* outputs a RADIANCE scene description of the sky with the appropriate luminance distribution. This is then mapped to the 2305 sky patches (the default) with *genskyvec* to obtain the sky luminance vector. This vector is finally passed to *dctimestep*, which scales the DCs accordingly.