The RADIANCE Photon Map Extension User Manual

Roland Schregle (roland.schregle@{hslu.ch, gmail.com})
CC Building Envelopes
Lucerne University of Applied Sciences and Arts

Revision 1.34
March 23, 2021

Contents

1 What's New?

Development of the RADIANCE photon map continues within a new project hosted by the Tokyo University of Science under the guidance of Prof. Nozomu Yoshizawa. The following features have been added:

- Photon port orientation options; ports may now be reversed to emit from their backfaces, or even from both sides. The default behaviour of (forward) emission towards the interior as defined by the port's normal (per ***mkillum*** convention) is retained.
- ***pmapdump*** text dump option; photon positions and their flux may now be dumped as plain text for input into interactive point cloud processors and viewers.

2 Introduction and Motivation

The RADIANCE rendering system [Ward1994] is a very versatile and powerful tool for lighting simulation, and one of the few physically based renderers with available source code. However, developments in light redirecting materials have presented new challenges in their simulation; their specular nature makes them difficult to simulate with RADIANCE, often resulting in an undersampled specular component and therefore excessive noise. Extreme sampling parameters yield a marginal improvement at the expense of high computational load.

Particularly the phenomenon of caustics is undersampled with RADIANCE. Caustics are bright, iridescent highlights on diffuse surfaces caused by specular reflection or refraction. Figure 1 shows an example of a caustic from a retroreflecting lamella, which is a typical redirecting component encountered in daylight simulation. Quoting chapter 13 of "Rendering with RADIANCE", [Ward1998],

> "Other cases involving curved, specular reflectors pose similar difficulties for ***mkillum***, and the only long-term solution seems to be the creation of a forward raytracing module for computing these kinds of illums."

The problem is actually more fundamental due to RADIANCE's backward raytracing methodology, and not restricted to ***mkillum*** alone.

Figure 1: Photograph of caustic from retroreflecting lamella (patented by Helmut Köster).

RADIANCE's difficulties with highly directional materials motivated the development of the forward raytracing module mentioned above as a supplementary algorithm. The photon map [Jens1996, Jens1997, Jens2000] was chosen for a number of reasons:

- Apart from handling indirect diffuse illumination, it handles caustics efficiently and has become the algorithm of choice for this purpose.
- It is straightforward to integrate into existing ray tracing environments.
- It is decoupled from the scene geometry and thus can handle arbitrarily complex objects. This also enables using the photon map to simulate indirect illumination in volumes such as participating media [Jens1998, Schr2019].
- It represents a view independent solution to the indirect illumination and can thus be reused for multiperspective renderings with unchanged geometry.

The photon map is based on a (light) particle transport simulation, which lends its name. Each photon interacts with the objects it strikes and is either reflected, transmitted, or absorbed, depending on the characteristics of the material. A Monte Carlo sampling method is used to generate the reflected or transmitted directions of a photon if it survives. Eventually a particle is terminated either by absorption or leakage (if it leaves the scene), and a new photon is emitted.

3 Overview of the Implementation

The photon map algorithm comprises two steps:

Photon distribution (forward pass): as a preprocess, photons are emitted from all light sources and probabilistically reradiated or absorbed upon striking a surface or passing through a volume, depending on its properties. The result is a view independent representation of the indirect illumination (figure 2).

Photon gathering (backward pass): during the actual rendering, the indirect illumination for a point on a surface or within a volume is determined by finding a number of nearest photons to the point. The irradiance is then proportional to the photon density (figure 3), hence the process is known as *density estimation* in the statistics literature. The number of photons gathered defines the area covered by the density estimate, and will be referred to as its *bandwidth* for the rest of this document.

Note that the combination of a forward and backward pass comprises a bidirectional raytracer, thus all possible light transport paths are accounted for.

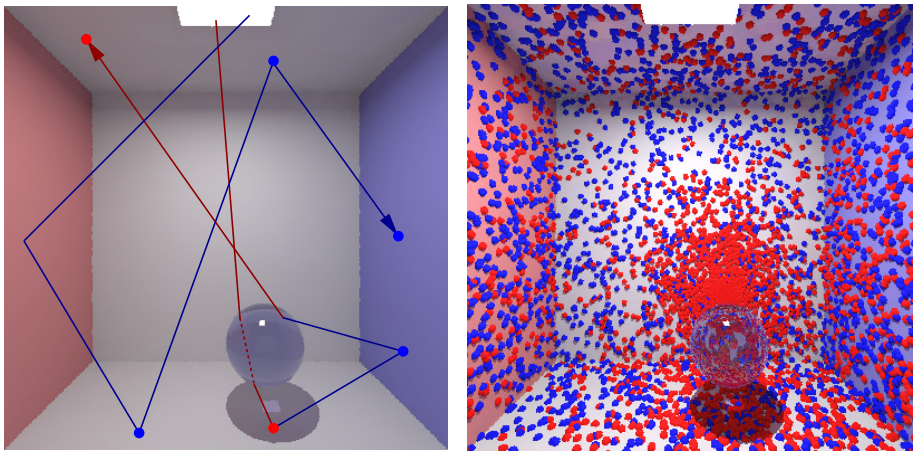

Figure 2: Global photon distribution in the Cornell box. Left: global and caustic photon paths during forward pass. Dots indicate stored photons, where blue represents global photons, and red represents caustic photons. Right: photon distribution after completion of forward pass (visualised with ***pmapdump***). Note the concentration of caustic photons on the back wall reflected from the glass sphere.

The distribution step is performed by the ***mkpmap*** utility. It is responsible for building the photon map and saving it to a file for subsequent reuse by ***rpict***[1] for

[1] Unless noted otherwise, ***rpict*** also implies its siblings ***rtrace*** and ***rvu***, which also support photon mapping.

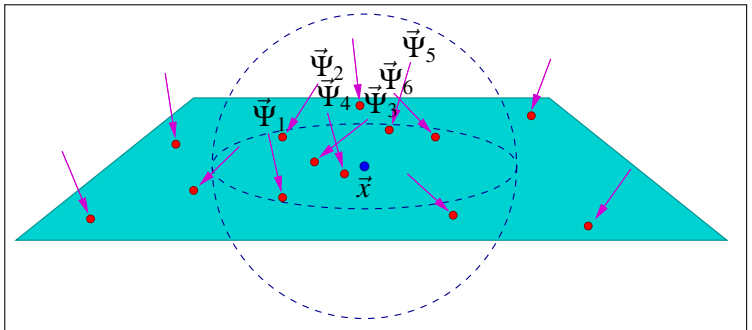

Figure 3: Gathering photons via nearest neighbour search. The number of gathered photons defines the *bandwidth*, or intercepted area, of the density estimate use to evaluate the irradiance.

the actual rendering and photon gathering. The general photon mapping workflow can therefore be summarised as:

```
/* Prepare model... */
/* Thinkety-think... (mull over parameters) */

mkpmap [-apg globalpmap nphotons] \
       [-apc causticpmap nphotons] \
       [-apv volumepmap nphotons] ... \
       [options] octree

/* Clickety-click... pokety-poke... */

rpict | rtrace | rvu [-ap globalpmap [bwidth]] \
                     [-ap causticpmap [bwidth]] \
                     [-ap volumepmap [bwidth]] ... \
                     [options] octree
```

Doan' panic, parameters (highlighted in italics) will be explained in the following sections.

Six photon types are available, which are stored in separate photon maps and visualised differently:

Global photons account for all indirect (ambient) illumination incident on surfaces with a diffuse component.

Caustic photons account for all indirect specular to diffuse transfers, i.e. caustics. These transport paths are also accounted for by global photons, but not visualised directly like caustics.

Volume photons account for indirect illumination within participating media using the *mist* primitive. See section 3.1.1 for details.

Direct photons account for the direct component from light sources and are intended to serve as a debugging option. These should be rarely used in practice.

Precomputed global photons store irradiance from precomputed density estimates from a number of nearby photons. This is an optimised variant of global photons and assumes a low indirect irradiance gradient. See section 3.2.1 for details.

Contribution photons account for contributions from light sources via references to the emitting source, which may also be normalised as coefficients. This is a variant of global photons intended for climate based modelling in conjunction with ***rcontrib***. See section 6 for details.

The stock RADIANCE functionality (referred to here as RADIANCE CLASSIC™) is preserved in ***rpict*** when photon mapping mode is disabled. When rendering with the photon map, the ambient calculation is augmented by a photon map density estimate, but the direct and indirect specular components are still delivered by RADIANCE CLASSIC™.

3.1 Supported Materials

Material types in RADIANCE must provide support for probabilistically generating outgoing directions for incident photons using Monte Carlo methods. The following material primitives support photon mapping:

Analytical Gaussian: *plastic, metal, trans, plastic2, metal2, trans2*

Refracting/Reflecting: *glass, dielectric, interface, mirror*

Data-driven *bsdf*

Mixtures: *mixfunc, mixdata*

Patterns: *colorfunc, brightfunc, colordata, brightdata, colorpict*

Textures: *texfunc, texdata*

Volumes: *mist, antimatter*

Materials with scattering defined procedurally or through tabulated data are only partially supported. The *plasfunc, metfunc, transfunc, plasdata, metdata*, and *transdata* materials currently only scatter photons diffusely, and will not produce caustics. The *brtdfunc* material only produces caustics via ideal (mirror) specular reflection and transmission. These materials have been largely superseded by the more general *bsdf* material, which uses a highly optimised internal data representation and efficient sampling, and is preferred for realistic scattering behaviour.

3.1.1 Participating Media

The RADIANCE photon map supports indirect inscattering in participating media defined with the *mist* primitive. Participating media are somewhat unwieldy in RADIANCE; in order for volume photons to function correctly with materials which modify the coefficient of extinction and scattering albedo (i.e. *dielectric* and *interface*), it is necessary to surround the light source(s) and viewpoint with a *mist* boundary (e.g. *bubble*), unless one intends to restrict volume photons to a small section of the scene.

Furthermore, in order to obtain volume caustics from light passing through an object (see Figure 17 for such an example), it must also be surrounded by a mist boundary. Alternatively, the mist parameters can be set globally on the command line, however this requires setting the source and viewpoint boundaries' extinction to zero, otherwise extinction is accumulated.

Furthermore, ***rpict*** only accounts for inscattering along rays of finite length, i.e. which intersect a local object. The scene therefore requires peripheral geometry to delineate a finite expanse of *mist*, e.g. a boundary of this material for the entire scene. All this just to make life more interesting.

That being said, volume photon mapping opens up interesting applications beyond rendering caustics in fog or smoke, for example the simulation of light propagation in architectural spaces [Schr2019], as shown in Figure 11.

3.1.2 Light Sources

Light sources require additional code for photon emission. Currently supported primitives are *light, spotlight, glow*, and *source.* To ensure accurate distribution, light source geometries are partitioned to a user specified resolution and photons emitted from each partition.

The behaviour of a few emitter types differs from RADIANCE CLASSIC™. In the case of the *source* primitive there is no geometry to partition; instead, the scene cube faces are partitioned and act as emitters, unless photon *ports* are used (see section 3.2.4). Furthermore, the *glow* primitive acts as regular light source within its maximum radius for shadow testing, in which case its contribution is accounted for by the photon map, otherwise it is handled by RADIANCE CLASSIC™'s ambient calculation. Virtual sources are disabled with photon mapping, since caustic photons account for this functionality.

As in RADIANCE CLASSIC™, light sources may be modified by function files. Functions may access the following variables (see also *rayinit.cal*) during photon emission, which are defined in terms of the photon origin on the light source:

Dx, Dy, Dz : the photon's direction, pointing towards its origin on the light source

Nx, Ny, Nz : the surface normal on the light source at the photon origin

Px, Py, Pz : the emitted photon's origin on the light source

Rdot : the dot product of the surface normal and the photon's direction at its origin.

3.2 Optimisations

A number of optimisations have been incorporated into the implementation to either improve performance or accuracy. The user must be aware of the fact that most of these optimisations incur trade-offs; increasing performance will inevitably compromise accuracy, and vice versa. Such is the harsh reality of Monte Carlo.

3.2.1 Precomputed Global Photons

Per Christensen's precomputed photon mapping approach [Chri2000] has been integrated into ***mkpmap*** and is available via the **-app** option to accelerate the subsequent global photon gathering step in ***rpict***.

With this optimisation, global photon irradiance is precomputed for a fraction of the photon positions and stored with the photons, while the rest are discarded. This reduces the overhead of global photon lookups during the gathering phase; only the single closest global photon is retrieved, and its precomputed irradiance can be used directly (figure 4). The disadvantage of this optimisation is potential local bias in scenes with nonuniform indirect illumination.

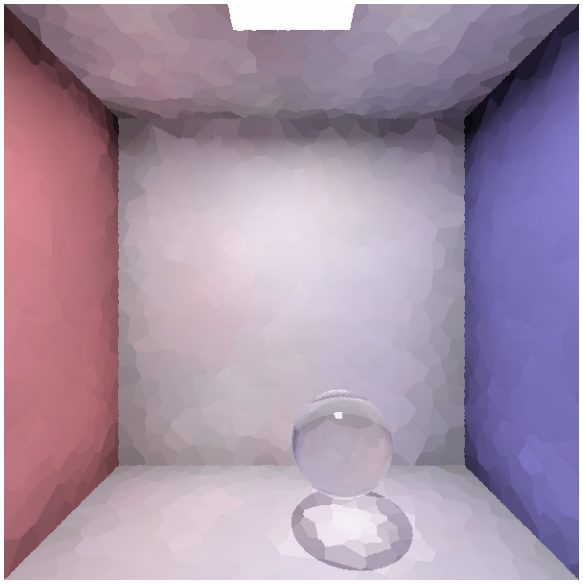

Figure 4: Directly visualised precomputed global photon irradiance. Note the caustic from the glass sphere.

3.2.2 Bias Compensation

The density estimate bandwidth affords the user control over the quality and accuracy of the rendering by trading off between noise and bias. Setting the bandwidth too low may result in excessive noise, while setting it too high may result in excessive blurring and local bias in the solution.

To counter this bias/noise tradeoff, a *bias compensation* [Schr2003] was developed and incorporated into the density estimate as an option. Its purpose is to adapt the bandwidth within a user specified lower and upper bound.

Bias compensation employs a binary search for an optimal bandwidth based on the likelihood that deviations from a running average irradiance are due to noise or bias. This results in substantially improved contours in nonuniform indirect illumination (particularly caustics) with a slight increase in noise in these regions (figure 5).

The algorithm also compensates for boundary bias (irradiance falloff at polygon edges) and chromatic bias. However, this sophistication comes at a price, as it does impact performance noticeably.

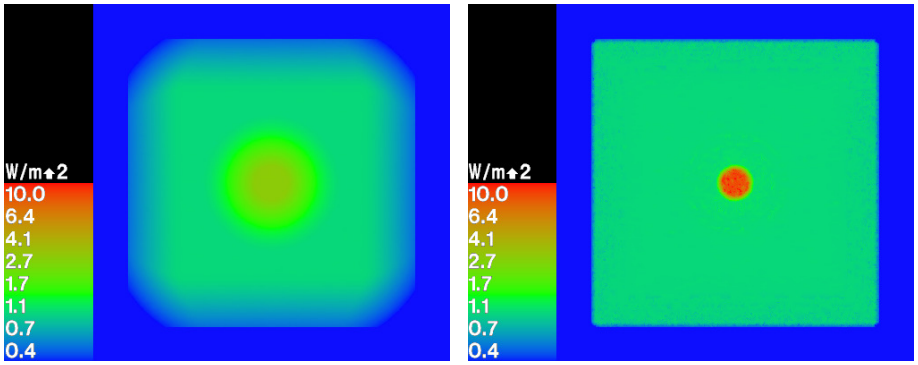

Figure 5: Left: falsecolour rendering of severely biased caustic with a bandwidth of 5000 photons. The caustic should have an irradiance of 10 W/m^2. Right: falsecolour rendering of the same caustic with bias compensation using a bandwidth of 50–5000 photons.

3.2.3 Automatic Maximum Search Radius

The photon lookup algorithm uses a maximum search radius to identify the nearest neighbours. This radius is progressively reduced during the search until the required number of photons (as defined by the density estimate bandwidth) remain. Photons beyond the maximum search radius are disregarded.

An appropriately sized maximum search radius will eliminate many distant photons from the search, thus greatly accelerating the process. Setting it too small, however, will result in a "short" lookup, with fewer photons found than requested

with the bandwidth. While this does not invalidate the resulting density estimate, a very low number of photons will render it inaccurate due to excessive noise.

An optimal maximum search radius is difficult to predict as it changes dynamically with the local photon density. The maximum search radius is therefore estimated from the average photon distance to the centre of gravity of the photon distribution. This adapts the maximum search radius to the average photon density and accounts for non-uniform concentrations of photons, as is often the case with caustics.

In situations where the photon distribution is grossly skewed, the user can manually override this automation in ***rpict*** by specifying a fixed maximum search radius with the **-am** option.

3.2.4 Photon Ports

Emitting photons from the *source* primitive presents a challenge; there is no local geometry associated with these directional sources. While the default behaviour is to emit photons from the scene cube faces, this can become highly inefficient for sources with large solid angles. These are infact situations for which a forward raytracer is fundamentally inappropriate. RADIANCE CLASSIC™'s backward raytracer fares better here, since sending a ray to a large source is trivial. With the forward raytracer, the majority of emitted photons will be lost and won't contribute to the photon map.

To aggravate the situation, directional sources are often used to render interior daylit spaces. With the viewer positioned inside, the few photons that actually strike the geometry must also pass through windows or skylights before they can even contribute visibly to the photon map. Needless to say, this amounts to abysmal performance.

A simple workaround to the problem is the concept of *photon ports*. The ports are basically apertures through which photons can enter the space containing the viewpoint. These can be windows, skylights, but also invisible polygons acting as boundaries within the scene (figure 6). The primary drawback to this scheme is that it requires user intervention. The user must define the port geometry and specify the objects by their modifiers using the **-apo** option to ***mkpmap***. Adroit placement of ports can dramatically reduce the photon loss factor, resulting in substantially shorter distribution times with ***mkpmap***.

All objects in the scene description using the specified port modifiers act as ports during photon distribution. Photons are then emitted directly from the ports rather than from the scene cube faces. The incident flux from directional sources is evaluated at each port and the sources are checked for occlusion. However, no interreflection calculation is made; a port differs from an *illum* in this respect. Photons are scattered by the port upon emission as if they had passed through it.

By default, *port objects are defined with their surface normals pointing inwards* as per ***mkillum*** convention. This behaviour can be overridden with if the port normals happen to point outwards; in this case, the port orientation can be reversed.

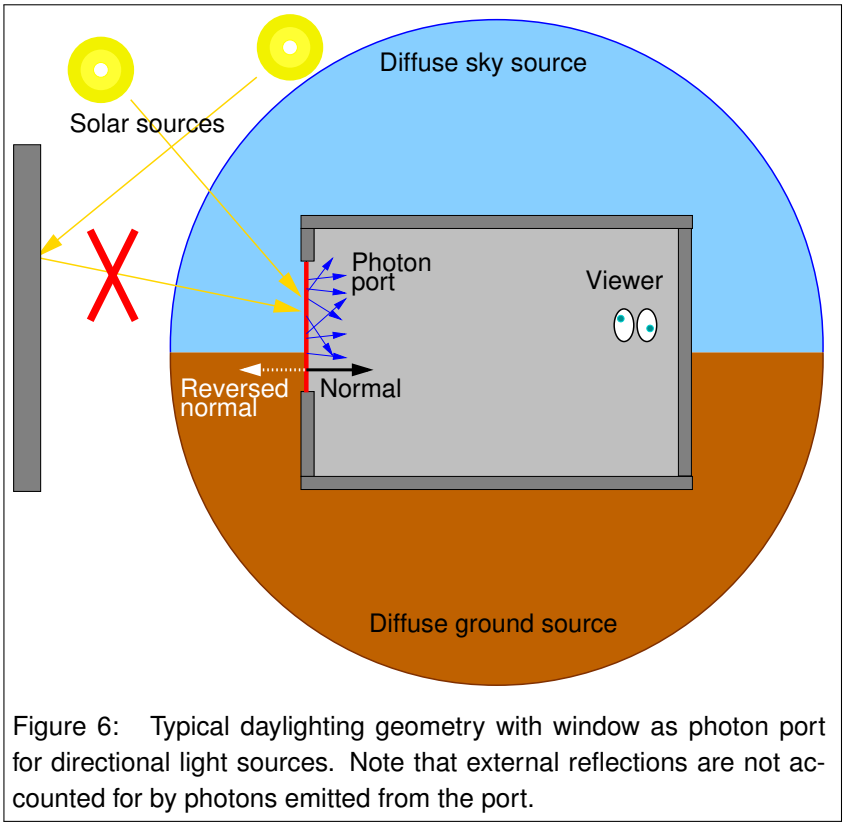

Figure 6: Typical daylighting geometry with window as photon port for directional light sources. Note that external reflections are not accounted for by photons emitted from the port.

In addition, ports can also be configured to emit bidirectionally, i.e. from either side. A possible use case is using a *mist* volume boundary as port; per definition, the photon will only scatter if it enters the medium, i.e. if its normal points outwards. In this case, the port needs to be reoriented to face inwards.

In some cases ports may not coincide with any scene geometry. These situations call for invisible ports. Photons should be emitted from them, but they should not be affected by scattering through the port, and the port should not be visible during rendering. One simple solution is to define photon ports using the *antimatter* modifier, with *void* as its only string argument.

Photon ports are the only optimisation in the implementation which do not compromise accuracy for performance, assuming correct placement on behalf of the user. A more general approach using Markov Chain techniques to automatically find viable photon paths without user intervention may materialise one day. Maybe.

Or not. Probably not.

3.2.5 Regions of Interest (ROIs)

The photon map extension supports the specification of one or more regions of interest, or ROIs, in which to exclusively store photons. This is

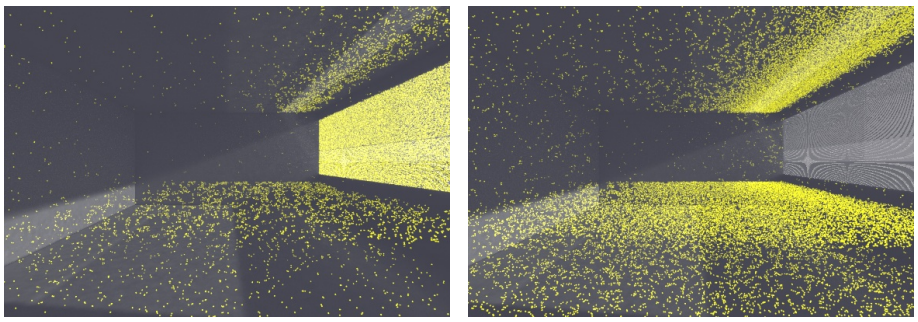

Figure 7: Photon distributions (visualised with ***pmapdump***) within an interior fitted with a redirecting system. Without constraint, the majority of photons are stored in the fenestration due to interreflection (left). By defining a region of interest corresponding to the room's extent with the **-api** option, ***mkpmap*** only deposits photons in the interior, resulting in an improved distribution and rendering quality (right). This example also uses an invisible *antimatter* sensor surface (defined via the **-aps** option) to deposit photons at work plane height. This is noticeably underpopulated without ROI, thus impacting illuminance measurements on it.

done by supplying appropriate 3D coordinates defining bounding boxes as $(x_{min}, y_{min}, z_{min}, x_{max}, y_{max}, z_{max})$ to ***mkpmap*** via mutiple instances of the **-api** option.

Photons are then only stored if their hitpoints lie within one of the defined ROIs, in which case it will be added to the photon map. The photon path and the resulting contribution to the lighting solution remains unaffected; path segments entering a ROI will deposit photons, while those leaving a ROI will not. Note that the number of stored photons remains approximately the same, but the photon map generation can take considerably longer, depending on the volume and location of the ROIs relative to the photon trajectories.

The primary purpose of defining a ROI is to optimise the photon distribution by concentrating photons where they contribute significantly, and disregarding parts of the geometry which are out of view or insignificant. A typical example is an interior space with a fenestration, with many photons being deposited within the latter, thus reducing the number of useful photons in the interior. The ROI would then correspond to the extent of the interior and exlude the fenestration. An example of this is shown in figure 7.

ROIs are an advances option withinherent pitfalls, as excluding photons from the scene can lead to bias. It is assumed that the user can estimate the bias that results from omitting the indirect illumination contributed by the excluded photons outside the ROI. For example, using a ROI corresponding to the abovementioned interior to estimate glare from the fenestration would clearly lead to underestimation as interreflections within the fenestration are omitted.

4 Generating photon maps with *mkpmap*

The ***mkpmap*** utility accepts a RADIANCE octree as input and performs a forward raytracing pass to generate a photon map of each specified type for the scene. The photons are stored in a space subdividing data structure [Bent1975] which facilitates nearest neighbour queries required for the photon lookup step. The data structure is written to a portable binary file for subsequent use by ***rpict***.

4.1 Distribution Algorithm

The parametrisation of the distribution step is primarily intended to be user friendly; the user specifies the required number of photons stored in each map. These are treated as "targets" which the photon distribution code tries to attain, since there is no way of knowing the outcome of the distribution a priori.

The distribution algorithm makes an educated guess at the number of photons it must emit in order to satisfy the targets. This is based on the results of a distribution prepass in which a small number of photons (a fraction of the minimum specified target) is emitted and distributed. Since the distribution process scales linearly, the remaining number of photons to emit can be predicted and distributed in the main pass. The actual number of photons generated after the distribution is complete will approximate the targets.

4.2 Distribution Failure

The distribution step may bog down and terminate after a user specified number of prepass attempts if any of the specified photon maps are still empty. This is usually an indication of a scene anomaly or an incompatible photon map specification. This is intended as a convenience for the user; without it, ***mkpmap*** would end up in an infinite loop. It is nevertheless the user's responsibility to specify photon map types compatible with the materials and to check the validity of the scene.
The following situations can cause ***mkpmap*** to fail:

- pathological scenes in which none of the light sources see any geometry
- specifying a global photon map for a scene containing no materials with diffuse component
- specifying a caustic photon map for a scene containing no materials with diffuse or specular component
- specifying a volume photon map for a scene lacking the *mist* primitive.

4.3 Runaway Photons

Photons may become trapped inside *dielectric* or *interface* materials, often due to infinite total reflection. These "runaway" photons are terminated after a user defined

bounce limit is exceeded, and a warning is issued under the assumption that this is an anomaly of the scene geometry.

4.4 Progress Report

If progress reports are enabled, ***mkpmap*** will report the following during distribution:

```
Ne emitted, Ng global, Nc caustic, Nv volume, Nd direct,
p% after t hours
```

where:

- N_e is the total number of emitted photons
- N_g, N_c, N_v, and N_d is the number of global, caustic, volume, and direct photons stored, respectively
- p is the percentage completed
- t is the time elapsed.

When generating a precomputed global photon map, a report will also be issued subsequent to the distribution step, indicating the number of precomputed photons:

```
Np precomputed, p% after t hours
```

4.5 Command line arguments for *mkpmap*

The general synopsis of the ***mkpmap*** utility is:

```
mkpmap -apg|-apc|-apv|-apd|-app|-apC \
        pmapfile nphotons [precompbwidth] ... \
        [options] octree
```

The options are as follows:

-apg *pmapfile nphotons*
: Generate a global photon map containing approximately *nphotons* photons, and output to *pmapfile*. This accounts for all indirect illumination, from both specular and diffuse scattering, on surfaces with a diffuse component. This is the most general type of photon map and replaces the ambient calculation in ***rpict***.

-apc *pmapfile nphotons*
Generate a separate caustic photon map containing approximately *nphotons* photons, and output to *pmapfile*. This is a subset of the global photon map intended for direct visualisation at primary rays. This accounts for all indirect illumination on diffuse surfaces from specular scattering, which usually exhibits a large gradient and requires a higher resolution than the global photon map, typically containing the tenfold number of photons.

-apv *pmapfile nphotons*
Generate a volume photon map containing approximately *nphotons* photons, and output to *pmapfile*. These account for indirect inscattering in participating media such as *mist* and complement the direct inscattering computed by ***rpict***. See also the **-me**, **-ma** and **-mg** options below.

-apd *pmapfile nphotons*
Generate a direct photon map containing approximately *nphotons* photons, and output to *pmapfile*. This only accounts for direct illumination on diffuse surfaces, and is intended for debugging and validation of photon emission from the light sources, as the quality is too low for actual rendering.

Direct photon maps are useful to determine suitable settings for the **-ds** and **-dp** options when using light sources with complex modifiers, as they provide a visual check for the resolution of the probability density function used for photon emission. Furthermore, it can serve as a validation tool for comparison with RADIANCE's direct component.

-apC *pmapfile nphotons*
Generate a light source contribution photon map containing approximately *nphotons* photons, and output to *pmapfile*. This may then be used by ***rcontrib*** to compute light source contributions. When used with ***rpict***, contribution photon maps behave as regular global photon maps and yield cumulative contributions from all light sources.

With this option, ***mkpmap*** uses a modified photon distribution algorithm that ensures all light sources contribute approximately the same number of photons. Each photon indexes a primary hitpoint, incident direction, and emitting light source which can be used to bin contributions per light source and direction.

Note ***mkpmap*** cannot generate a contribution photon map in combination with others in a single run, as it uses a different distribution algorithm. Other photon maps specified on the command line will be ignored.

See section 6 for details on using photon maps with ***rcontrib***.

-app *pmapfile nphotons precompbwidth*
Generate a precomputed global photon map containing a fraction of *nphotons* photons (specified with the **-apP** option, see below), and output to *pmapfile*.

This is a special case of the global photon map where the irradiance is evaluated with a bandwidth of *precompbwidth* photons for a fraction of the photon positions, and stored as photon flux; the remaining photons are discarded as their contributions have been accounted for. This obviates the explicit irradiance evaluation by ***rpict***, thus effecting a speedup at the expense of accuracy. The resulting error is tolerable if the indirect illumination has a low gradient, which is usually the case with diffuse illumination.

See section 3.2.1 for details on using precomputed global photons.

-apD *predistrib*

Photon predistribution factor; this is the fraction of *nphotons* which are emitted in a distribution prepass in order to estimate the remaining number of photons to emit in the main pass to approximately yield a photon map of size *nphotons*.

Setting this too high may yield more than *nphotons* in the initial pass with highly reflective geometry. Note that this value may exceed 1, which may be useful if the resulting photon map size greatly deviates from *nphotons* with a very low average reflectance.

-api x_{min} y_{min} z_{min} x_{max} y_{max} z_{max}

Defines a region of interest within which to store photons exclusively; photons will only be stored within the volume bounded by the given minimum and maximum coordinates. Multiple instances of this option may be specified with cumulative effect to define compound regions of interest.

This is useful for constraining photons to only the relevant regions of a scene, but may increase the photon distribution time. Beware that this option may lead to biased results. See section 3.2.5 for details and caveats on using ROIs.

-apm *maxbounce*

This option is synonymous with **-lr** for backwards compatibility, and may be removed in future releases.

-apM *maxprepass*

Maximum number of iterations of the distribution prepass before terminating if some photon maps are still empty. This option is rarely needed as an aborted prepass indicates an anomaly in the geometry or an incompatibility with the specified photon map types.

-apo[*+|-|0*] *portmod*

Specifies a modifier *portmod* to act as a photon port. All objects using this modifier will emit photons directly in lieu of any distant light sources defined with the *source* material. This greatly accelerates photon distribution in scenes where photons have to enter a space which separates them from the emitting light source via an aperture (e.g. fenestration, skylight) acting as a port.

In a typical daylight simulation scenario, a fenestration acts as a port to admit photons into an interior after emission from sky and solar sources. See section 3.2.4 for details on using photon ports. Multiple instances of this option may be specified.

By default, ports are oriented to emit in the halfspace defined by their associated surface normal. This can be overridden by specifying a trivalent suffix as follows:

+: forward emission; this is equivalent to the abovementioned default behaviour,

-: backward emission; the port is reversed and photons are emitted into the halfspace facing away from the surface normal,

***0*:** bidirectional emission; photons are emitted from both sides of the port.

Typical situations that call for a reversed photon port include, for example:

- Using fenestrations as ports that were (for whatever reason) defined with outward facing normals, without having to revise the model,
- Using a *mist* primitive as a port, since this requires outward facing normals in order to register the photons as having entered the volume.
- Reorienting a port associated with a *bsdf* modifier, since inverting its normal would also reorient the BSDF and alter its behaviour.

Other oddball scenarios are conceivable. If in doubt, specify a bidirectional port orientation for a slight performance penalty, as photon emission is attempted from both sides. For well-defined port geometry with inward-facing normals, just use the default; doan' mess with da normalz.

Photon port geometry is discretised according to the **-dp** and **-ds** options. These parameters aid in resolving spatially and directionally varying illuminance received by the port from distant light sources, e.g due to partial occlusion or when using climate-based sky models.

-apO[*+*|*-*|*0*] *portfile*
Read photon port modifiers from the file *portfile*, separated by whitespace, as a more convenient alternative to multiple instances of **-apo**.

-apP *precomp*
Photon precomputation factor; this is the fraction of *nphotons* global photons to precompute in the range $]0, 1]$ when using the **-app** option. The remaining photons are discarded. See section 3.2.1 for details on using precomputed global photons.

-apr *seed*
Seed for the random number generator. This is useful for generating different photon distributions for the same octree and photon map size, notably in progressive applications.

-aps *sensmod*

Specifies a modifier *sensmod* defined as *antimatter* material to act as a virtual (i.e. invisible) sensor surface. Photons will be deposited on all surfaces using this modifier, just like regular materials, but will then be transferred through the surface without undergoing scattering; the surface therefore does not affect the light transport and simply acts as an invisible photon receiver.

This is useful when photon irradiance is to be evaluated at points which do not lie on regular geometry, e.g. at workplane height with ***rtrace***'s **-I** option. Without this workaround, photons would be collected from parallel but distant planes, leading to underestimation.

Note that photons are only deposited when incident from the front side of the sensor surface, i.e. when entering the *antimatter*, thus the surface normal is relevant. ***mkpmap*** reports an error if the specified modifier is not an *antimatter* material.

See figure 3.2.5 for an example of a sensor surface in action.

-apS *sensfile*

Read virtual sensor surface modifiers from the file *sensfile*, separated by whitespace, as a more convenient alternative to multiple instances of **-aps**.

-ae *excmod*

Add *excmod* to the ambient exclude list, so that it will be ignored by the photon map. Objects having *excmod* as their modifier will not have photons deposited on them. Multiple modifiers may be given, each as separate instances of this option.

-aE *excfile*

Same as **-ae**, except modifiers to be excluded are read from *excfile*, separated by whitespace.

-ai *incmod*

Add *incmod* to the ambient include list, so that it will contribute to the photon map. Only objects having *incmod* as their modifier will have photons deposited on them. Multiple modifiersmay be given, each as separate instances of this option. Note that the ambient include and exclude options are mutually exclusive.

-aI *incfile*

Same as **-ai**, except modifiers to be included are read from *incfile*, separated by whitespace.

-bv[+|-]

Toggles backface visibility; enabling this causes photons to be stored and possibly scattered if they strike the back of a surface, otherwise they are unconditionally absorbed and discarded.

-dp *sampleres*

Angular resolution for sampling the spatial emission distribution of a modified light source or photon port (e.g. via *brightfunc*), in samples per steradian. This determines the accuracy of the numerical integration of the flux emitted by the source or port, and may need to be increased with complex distributions in combination with caustics.

-ds *partsize*

Light source partition size ratio; this determines the spatial resolution for photon emission from local light sources (or photon ports in case of distant sources). A light source or photon port is spatially partitioned to distribute the photon emission over its surface. This parameter specifies the ratio of the size (per dimension) of each partition to the scene cube extent, and may need to be increased for modified light sources with high spatial variance, or for partially occluded photon ports.

-e *diagfile*

Redirect diagnostics and progress reports to *diagfile* instead of the console.

-fo[*+*|*-*]

Toggles overwriting of output files. By default, ***mkpmap*** will not overwrite an already existing photon map file. This is to prevent inadvertently destroying the results of potentially lengthy photon mapping runs.

-ld *maxdist*

Limits cumulative distance travelled by a photon along its path to *maxdist*. Photon hits within this distance will be stored, and the photon is terminated once its path length exceeds this limit.

This is useful for setting radial regions of interest around emitting/reflecting geometry, but may increase the photon map generation times and yield biased results in regions beyond *maxdist*. Discretion is advised.

-lr *maxbounce*

Maximum number of bounces (scattering events) for a photon path before being considered “runaway” and terminated. Photons paths are normally terminated via Russian Roulette, depending on their albedo. With unrealistically high albedos, this is not guaranteed, and this option imposes a hard limit to avoid an infinite loop.

-ma *ralb galb balb*

Set the global scattering albedo for participating media in conjunction with the **-apv** option. See the ***rpict*** documentation for details.

-me *rext gext bext*

Set the global extinction coefficient for participating media in conjunction with the **-apv** option. See the ***rpict*** documentation for details.

-mg *gecc*
: Set the global scattering eccentricity for participating media in conjunction with the **-apv** option. See the ***rpict*** documentation for details.

-n *nproc*
: Use *nproc* processes for parallel photon distribution. There is no benefit in specifying more than the number of physical CPU cores available (so doan' even try). This option is currently not available on M$ Wind0z platforms.[2]

  So there. Tuff luck.

-t *sec*
: Output a progress report every *sec* seconds. See section 4.4.

At least one of the **-apg**, **-apc**, **-apv**, **-apd**, or **-app** options must be specified. These can be combined subject to compatibility with the materials used in the scene. The **-apC** option, in contrast, uses a different distribution algorithm and cannot be combined with the aforementioned; these will simply be ignored when generating a contribution photon map.

4.6 Examples

The following command generates a global photon map *bonzo.gpm* and a caustic photon map *bonzo.cpm* containing approximately 10000 and 100000 photons, respectively, with a progress report being issued every 5 seconds:

```
mkpmap -apg bonzo.gpm 10k -apc bonzo.cpm 100k -t 5 \
   bonzo.oct
```

Generate a global photon map containing 80000 photons, then precompute the diffuse irradiance for 1/4 of these with a bandwidth of 40 photons:

```
mkpmap -app bonzo-precomp.gpm 80k 40 -apP 0.25 bonzo.oct
```

Generate 1 million global photons by emitting them from external light sources of type *source* into a reference room via a fenestration with modifier *glazingMat* acting as photon port, with inward-facing normal:

```
mkpmap -apg refRoom.gpm 1m -apo glazingMat refRoom.oct
```

Generate a contribution photon map containing 10 million photons to bin light source contributions with ***rcontrib***:

```
mkpmap -apC bonzo-contrib.gpm 10m bonzo.oct
```

[2] Wind0z doesn't support RADIANCE's (admittedly legacy) UNIX-style *fork()* parallel programming paradigm. The photon map extension grudgingly conforms to this convention, although POSIX threads are now clearly the preferred, portable approach.

5 Rendering Photon Maps with *rpict*

The modified ***rpict*** rendering tool takes the photon map generated by ***mkpmap*** and performs photon lookups and density estimation. Without photon map functionality enabled, it behaves like RADIANCE CLASSIC™.

5.1 Caustic Photon Visualisation

Caustic photons are visualised at points seen directly by the observer, or via specular reflections. Virtual light sources (normally enabled with the *mirror* material) are disabled with the photon map, as the resulting caustics are already accounted for.

Note that irradiance calculations with ***rtrace*** using caustic photons are surface bound; irradiance for points which do not lie on surfaces will therefore be biased. As a workaround, ***mkpmap*** offers the **-aps** and **-apS** options to specify an invisible receiver surface with *antimatter* material for storing photons in the same plane as the sensor points. See section 4 for details.

5.2 Global Photon Visualisation

Visualisation with a global photon map implies an ambient calculation, thus ***rpict***'s **-ab** option defaults to 1; this differs from the standard RADIANCE CLASSIC™behaviour, where the ambient calculation must be explicitly enabled.

Global photon irradiance is visualised via an intermediate ambient bounce (commonly referred to as *final gather* in the literature [Chri2000], see figure 8). This reduces noise in the density estimates from the ambient daughter rays, and the resulting ambient irradiance is cached and interpolated as in RADIANCE CLASSIC™, unless explicitly disabled with the **-aa 0** option.

If the global photon irradiance was precomputed by ***mkpmap*** with the **-app** option, the ambient daughter rays return the precomputed irradiance from the closest global photons. This improves performance noticeably at the expense of accuracy if the indirect irradiance changes rapidly, which is rarely the case with diffuse illumination.

Specifying a negative number of ambient bounces **-ab** enables a debugging mode which does not spawn any ambient rays, but directly visualises the global photon irradiance at the primary ray. This provides a very rough but fast preview of the global illumination solution contained in the photon map. Figure 4 is an example of a such a preview for a precomputed photon map.

5.3 Progress Report

Bias compensation statistics are provided in the progress report with the standard **-t** option. The statistics for each photon type subject to bias compensation are reported as follows:

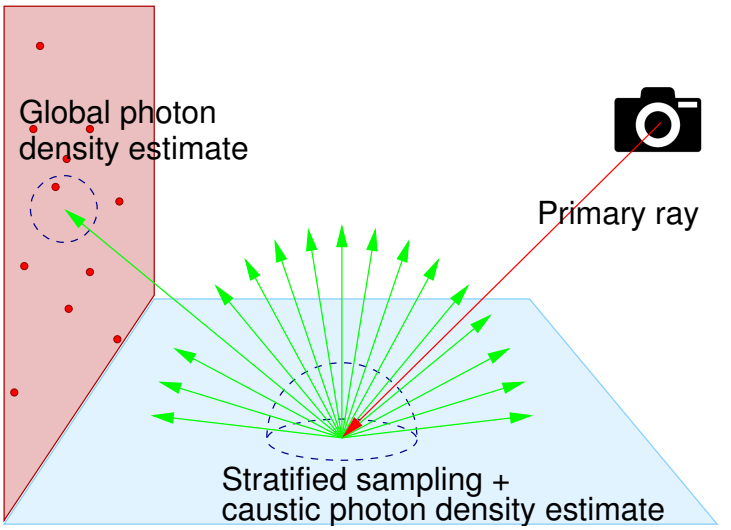

Figure 8: Caustic photons are visualised at the primary ray, while an intermediate stratified sampling step is used between the primary ray and the global photon density estimate (*final gather*).

$$N_{min}/N_{max}/N_{avg}\ \ ptype\ ,\ \ (\varepsilon_{min}/\varepsilon_{max}/\varepsilon_{rms}\ \ \mathrm{error})$$

where:

- N_{min}, N_{max} and N_{avg} is the minimum, maximum, and average number of photons used per density estimate, respectively
- $ptype$ is one of *global*, *caustic*, *volume*, or *direct*.
- ε_{min}, ε_{max}, and ε_{rms} is the minimum, maximum, and RMS (root mean square) estimated relative errors in the resulting irradiance, respectively.

Since bias cannot be determined without knowing the actual illumination, there is no reliable way of separating it from noise. As such, the error estimates are approximate and serve as a rough guide to judge the quality of the bias compensation.

5.4 Command Line Arguments for *rpict*

The photon mapping backward pass is disabled by default and must be explicitly enabled using the **-ap** option. The different photon map types may be renderered independently or in combination. Note that rendering only the global or caustic photon map does not take all light transport paths into consideration at the primary hitpoint. Since modifying the lights or geometry invalidates the photon map, a error is triggered if the octree is newer than the photon map, referring to the latter as “stale”.

In general, a physically valid simulation (assuming specular and diffuse materials) with the photon map is obtained with the following basic set of parameters:

```
mkpmap -apg globalpmap nphotons -apc causticpmap nphotons octree

rpict -ap globalpmap bwidth -ap causticpmap bwidth -ab 1 \
      -vf view octree
```

As with RADIANCE CLASSIC™, the **-ad**, **-ar**, and **-aa** parameters may need to be refined in some cases to improve the rendering quality and accuracy. Some parameters are adopted from RADIANCE CLASSIC™ due to their assumed familiarity to the user, and have similar semantics in the context of photon mapping. The ***rpict*** photon mapping options are as follows:

-ab *nbounces*

This value defaults to 1 in photon mapping mode (see **-ap** below), implying that global photon irradiance is always computed via one ambient bounce; this behaviour applies to any positive number of ambient bounces, regardless of the actual value specified.

A negative value enables a preview mode that directly visualises the irradiance from the global photon map without any ambient bounces. This is intended for debugging and checking the irradiance distribution, as the quality of the resulting renderings is generally poor with sparsely distributed photons. If the global photons are precomputed, this produces a Voronoi diagram.

As with RADIANCE CLASSIC™, setting this to 0 disables the ambient component, but caustic photon irradiance is still visualised at the primary ray.

-ap *pmapfile* [$bwidth_1$ [$bwidth_2$]]

Enables photon mapping mode. Loads a photon map generated with ***mkpmap*** from *file*, and evaluates the indirect irradiance depending on the photon type (automagically detected) using density estimates with a bandwidth of $bwidth_1$ photons, or the default bandwidth if none is specified (a warning will be issued in this case).

Global photon irradiance is evaluated as part of the ambient calculation (see section 5.2), caustic photon irradiance is evaluated at primary rays (see section 5.1), and indirect inscattering in mist is accounted for by volume photons. Contribution photons are treated as global photons.

Additionally specifying $bwidth_2$ enables bias compensation for the density estimates with a minimum and maximum bandwidth of $bwidth_1$ and $bwidth_2$, respectively. See section 3.2.2 for details on how bias compensation works.

Global photon irradiance may be optionally precomputed by ***mkpmap*** (see section 3.2.1, in which case the bandwidth, if specified, is ignored, as the nearest photon is invariably looked up.

Using direct photons replaces the direct calculation with density estimates for debugging and validation of photon emission.

-am *maxdist*

Maximum search radius for photon map lookups. Without this option, an initial maximum search radius is estimated for each photon map from the average photon distance to the dis- tribution's centre of gravity. It is then adapted to the photon density in subsequent lookups.

This option imposes a global fixed maximum search radius for all photon maps, thus defeating the automatic adaptation. It is useful when multiple warnings about short photon lookups are issued.

Note that this option does not conflict with the bandwidth specified with the **-ap** option; the number of photons found will not exceed the latter, but may be lower if the maximum search radius contains fewer photons, thus resulting in short lookups. Setting this radius too large, on the other hand, may degrade performance.

This setting applies to *all* photon maps specified with **-ap**.

-ac *pagesize* [3]

Set the photon cache page size. The photon cache reduces disk I/O incurred by on-demand loading (paging) of photons, and thus increases performance. This is expressed as a (float) multiple of the density estimate bandwidth specified with **-ap** under the assumption that photon lookups are local to a cache page.

Cache performance is sensitive to this parameter: larger pagesizes will reduce the paging frequency at the expense of higher latency when paging does occur. Sensible values are in the range 4 (default) to 16.

-aC *cachesize* [3]

Set the total number of photons cached when using out-of-core photon mapping, taking into account the pagesize specified by **-ac**. Note that this is approximate as the number of cache pages is rounded to the nearest prime. This allows adapting the cache to the available physical memory. In conjunction with the **-n** option, this is the cache size per parallel process.

Cache performance is less sensitive to this parameter, and reasonable performance can obtained with as few as 10k photons. The default is 1M. This option recognises multiplier suffixes (k = 10^3, M = 10^6), both in upper and lower case.

[3] These options are only available with out-of-core photon mapping. Check if they appear in the output of ***rpict*** **-help** to verify that your software build supports out-of-core photon mapping, as this is a compile-time option.

5.5 Examples

To render ambient illuminance with a global photon map *global.pm* via one ambient bounce, and from a caustic photon map *caustic.pm*, using a bandwidth of 50 photons for each:

```
rpict -ab 1 -ap global.pm 50 -ap caustic.pm 50 \
     -vf scene.vf scene.oct > scene.hdr
```

6 Photon Mapping with Light Source Contributions

The RADIANCE photon map has been adapted to facilitate the computation of light source contributions for annual and climate-based daylight simulation [Schr2015]. In this workflow, the scene supplied to ***mkpmap*** contains multiple light sources representing different time-dependent positions (e.g. solar elevations depending on season and time of day) of one source, which are handled as aggregate for efficient simulation. The ***rcontrib*** tool supplied with RADIANCE can then determine the irradiance for each light source for a given sensor position in the scene, thus efficiently accounting for each source's contribution. In practice, this is done by "binning" the resulting irradiance into separate output files for each light source. See the ***rcontrib*** man page for details.

In support of light source contributions, ***mkpmap*** uses a modified photon distribution algorithm via the **-apC** option that ensures each light source contributes approximately the same number of photons, irrespective of its emitted flux. To compensate and maintain energy balance, the flux of each photon is adjusted accordingly, which can raise the variance of the resulting density estimates.

The generated photon map contains a list of primary hitpoints, incident directions, and emitting light sources, which are indexed by each photon which is spawned along the primary ray's path. This information is used to identify the emitting light source for each photon map, as well as apply an additional binning function to the incident direction via the **-b** option as documented in the ***rcontrib*** man page, e.g. to identify contributions from individual sky patches. An example of binned photon contributions is shown in Figure 9 for patches from a Tregenza sky.

Unlike ***rcontrib***, the photon map only supports contributions from light sources, not arbitrary modifiers. Ray coefficients are supported via the **-V-** option, which amounts to normalising the emission of each light source for *a posteriori* scaling without rebuilding the photon map. Using fewer photons than there are light sources for photon density estimates results in omitted contributions, thus ***rcontrib*** clamps the bandwidth accordingly and issues a warning.

A typical invocation of the RADIANCE photon map with light source contributions consists of the following:

```
mkpmap -apC contribpmap nphotons octree

rcontrib -ap causticpmap bwidth -I -h -V -fo \
         -o %s-contrib.dat -M sourcemodfile octree < sensorfile
```

where *sourcemodfile* is a file containing the list of light source modifiers to bin, and *sensorfile* is a file containing a list of sensor positions as input for the irradiance calculation. The contributions are output to separate files $*$-*contrib.dat* prefixed by each light source modifier.

An example of a visual analysis of a redirecting component for a half year using contribution photons is shown in figure 10. The performance of the system

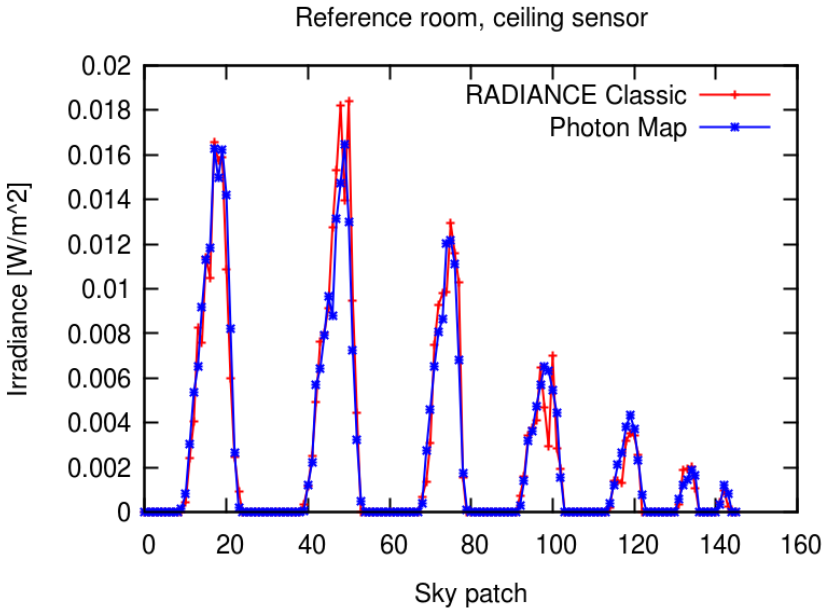

Figure 9: Comparison of photon contributions from 145 Tregenza sky patches (blue) to the indirect irradiance obtained with RADIANCE CLASSIC™ (red) on an interior ceiling sensor via a window facing south.

(consisting of retroreflecting blinds) can be assessed subject to seasonal daylight availability. Redirection towards the ceiling is evident particularly from January to April. Transmission, on the other hand, occurs at low solar angles until March, before transitioning into retroreflection at high solar angles.

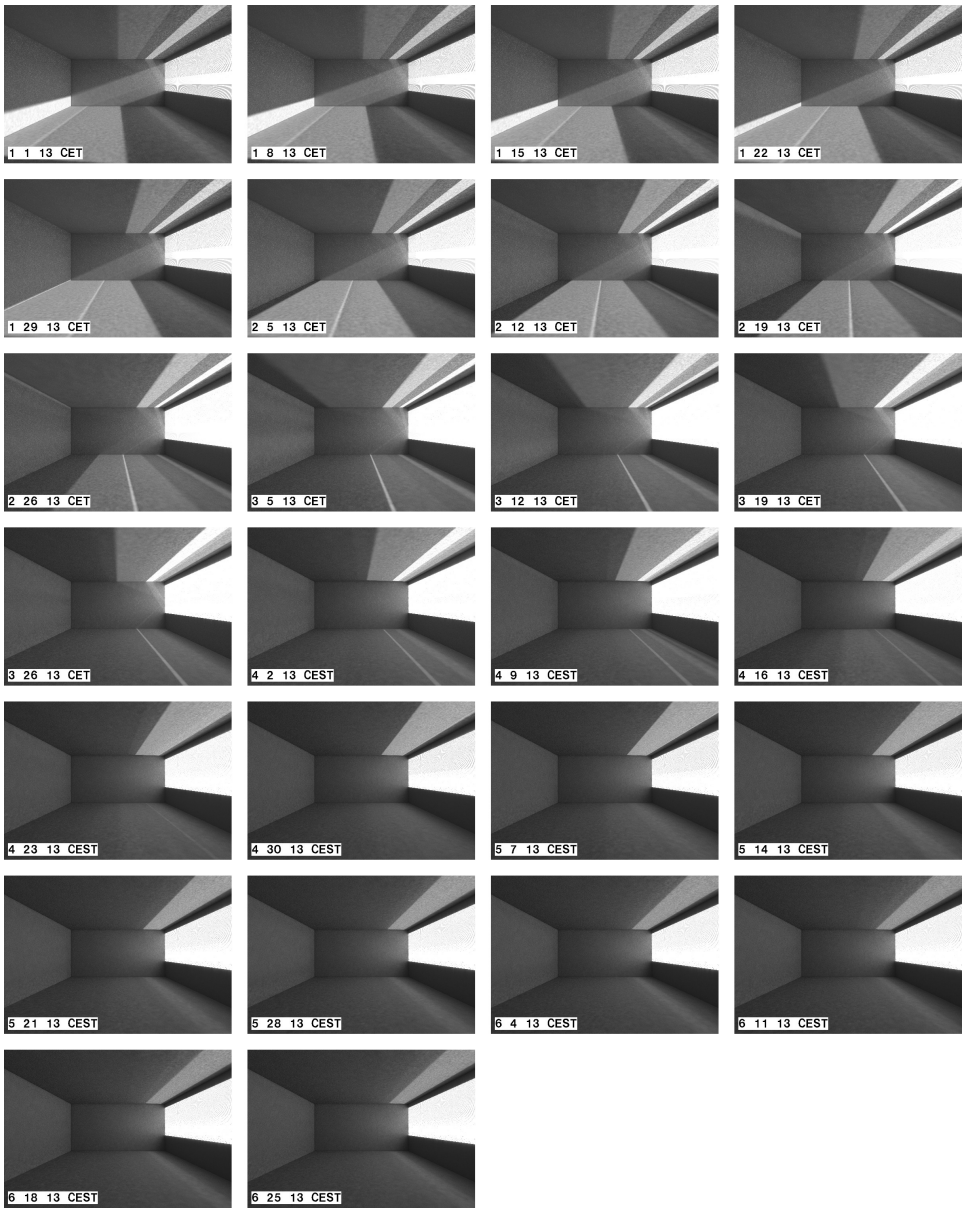

Figure 10: Retroreflecting blinds at 26 solar angles for a half year (January to June, in weekly intervals) rendered with contribution photon map and ***rcontrib***. In each instance the time of day is 1:00pm. Redirection and transmission dominate until March, while retroreflection dominates in the following months.

7 Utilities

7.1 Querying the Photon Map with *getinfo*

The ***getinfo*** tool supplied with RADIANCE can extract the information about a photon map from the header of the corresponding file. The output has the following format:

```
#?RADIANCE
cmdline
N_p photons @ [φ_r, φ_g, φ_b] avg watts
FORMAT=fmt
VERSION=ver
```

where

- $cmdline$ is the ***mkpmap*** command line used to build the photon map, including all arguments,
- N_p is the number of photons in the photon map,
- $[\phi_r, \phi_g, \phi_b]$ is the photon flux averaged over the RGB colour channels,
- fmt identifies the photon map type, which is one of
  - *Radiance_Global_Photon_Map*,
  - *Radiance_PreComp_Photon_Map*,
  - *Radiance_Caustic_Photon_Map*,
  - *Radiance_Volume_Photon_Map*,
  - *Radiance_Direct_Photon_Map* or
  - *Radiance_Contrib_Photon_Map*
- ver is a file format version number to check build options and output file compatibility.

As with other file types, ***getinfo*** can query multiple photon maps in one command line.

7.2 Visual Examination of Photon Maps with *pmapdump*

7.2.1 Overview

Pmapdump is a minimalist tool to visualise the photon distribution in one or more photon map files. The utility can be used for illustrative purposes to understand the photon map algorithm, but more importantly, it can be used to debug scenes in which the location of photons is unclear. Examples of ***pmapdump*** output can be seen in Figure 2 (right), Figure 7 and Figure 11.

Pmapdump takes one or more photon map files generated with ***mkpmap*** as input and, by default, sends a RADIANCE scene description of their photon distributions to the standard output. Photons are represented as spheres of material type *glow*. These can be visualised with e.g. ***objview***, ***rpict***, or ***rvu*** to assess the location and local density of photons in relation to the scene geometry. No additional light sources are necessary, as the spheres representing the photons are self-luminous.

Alternatively, photons can also be output as an ASCII point list, where each line contains a photon's position and colour. This point list can be imported in a 3D point cloud processor/viewer to interactively explore the photon map.

An arbitrary number of photon maps can be specified on the command line and the respective photon type is determined automagically. Per default, the different photon types are visualised as colour coded spheres according to the following default schema:

Blue: global photon

Cyan: precomputed global photon

Red: caustic photon

Green: volume photon

Magenta: direct photon

Yellow (Yellow): contribution photon

These colours can be overridden for individual photon maps with the **-c** option (see below). Alternatively, photons can be individually coloured according to their actual RGB flux with the **-f** option (see below); while this makes it difficult to discern photon types, it can be used to quantitatively analyse colour bleeding effects, for example.

7.2.2 Options

The synopsis for ***pmapdump*** is as follows:

```
pmapdump [-a] [-n num1] [-r radscale1] [-f] \
         [-c rcol1, gcol1, bcol1] pmap1 \
         [-a] [-n num2] [-r radscale2] [-f] \
         [-c rcol2, gcol2, bcol2] pmap2 \
         ...
```

Options are effective for the photon map file immediately following on the command line, and are reset to their defaults after completion of each dump. As such they must be set individually for each photon map. ***Pmapdump*** takes the following options:

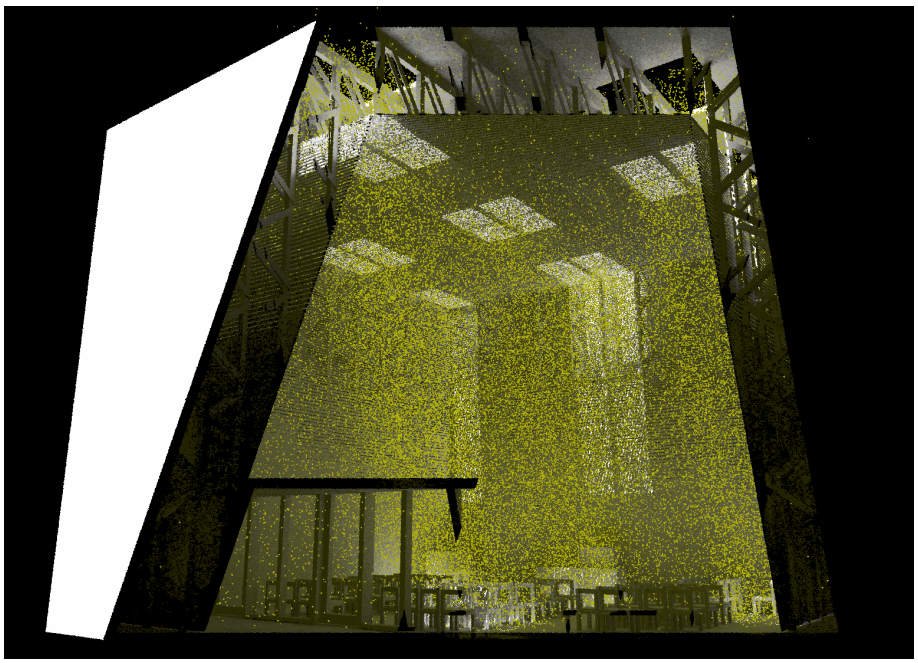

Figure 11: Cutaway view of volume photons entering a modernist church through skylights, rendered with default ***pmapdump*** output (yellow spheres) merged with the original scene geometry. Model courtesy of Prof. Nozomu Yoshizawa, Tokyo University of Science.

-a[*+*|-]

Boolean switch to output photons as a point list in ASCII (text) format instead of a RADIANCE scene. Each output line consists of 6 tab-separated floating point values: the X, Y, Z coordinates of the photon's position, and the R, G, B colour channels of its flux. These values. notably the flux, are expressed in scientific notation if necessary to accommodate their high dynamic range.

As ***pmapdump*** groups its options per photon map, this option must be specified per photon map for consistent output. This prevents erroneously dumping RADIANCE scene descriptions along with point lists, which will fail to load in the 3D point cloud processor/viewer.

-c *rcol gcol bcol*

Specifies a custom sphere/point colour for the next photon map. The colour is specified as an RGB triplet, with each component in the range (0..1]. Without this option, the default colour for the corresponding photon type is used. This option is mutually exclusive with **-f**.

-f[*+*|-]

Boolean switch to colour each sphere/point according to the corresponding photon's RGB flux instead of a constant colour. The flux is adjusted for the

fraction of dumped photons to maintain the total flux contained in the dumped photon map. Note that no exposure is applied, and as such resulting colours can span several orders of magnitude and may require tone mapping with ***pcond*** for visualisation. This option is mutually exclusive with **-c**.

-n *num*
: Specifies the number of spheres or points to dump for the next photon map. The dump is performed by random sampling with *num* as target count, hence the number actually output will be approximate. *Num* may be suffixed by a case-insensitive multiplier for convenience, where $k = 10^3$ and $m = 10^6$, although the latter may lead to problems when processing the output geometry with ***oconv***. The default number is 10k.

-r *radscale*
: Specifies a relative scale factor *radscale* for the sphere radius. The sphere radius is determined automatically from an estimated average distance between spheres so as to reduce clustering, assuming a uniform distribution. In cases where the distribution is substantially nonuniform (e.g. highly localised caustics) the radius can be manually corrected with this option. The default value is 1.0. This option is ignored for point list output in conjuction with **-a**.

7.2.3 Notes

Note that the RADIANCE scene output may contain many overlapping spheres in areas with high photon density, particularly in caustics. This results in inefficient and slow octree generation with ***oconv***. Generally this can be improved by reducing *num* and/or *radscale*.

7.2.4 Examples

The following example visualises the distribution of global and caustic photons superimposed on the scene geometry with 5000 pale red and 10000 pale blue spheres, respectively:

```
pmapdump -n 5k -c 1 0.4 0.4 global.pm \
         -n 10k -c 0.4 0.4 1 caustic.pm | \
oconv - scene.rad > scene_pm.oct

rvu scene_pm.oct
```

Note the use of a pipe to directly create *scene_pmdump.oct* from the ***pmapdump*** output.

In contrast, this example visualises the caustic photon distribution superimposed on the scene geometry with 10000 spheres coloured according to the photons' respective RGB flux:

```
pmapdump -n 10k -f caustic.pm | \
oconv - scene.rad > scene_pm.oct
```

RADIANCE scene dumps may also be viewed on their own by simply piping the output of ***pmapdump*** directly into ***objview*** (using the default number of spheres in this example):

```
pmapdump zombo.pm | objview
```

Instead of a RADIANCE scene, the following example dumps photons as a (really long) point list to an ASCII file for import into a 3D point cloud processor/viewer, as shown in Figure 12.

```
pmapdump -a -f -n 100k lotsa.pm > lotsa_pointz.xyz
```

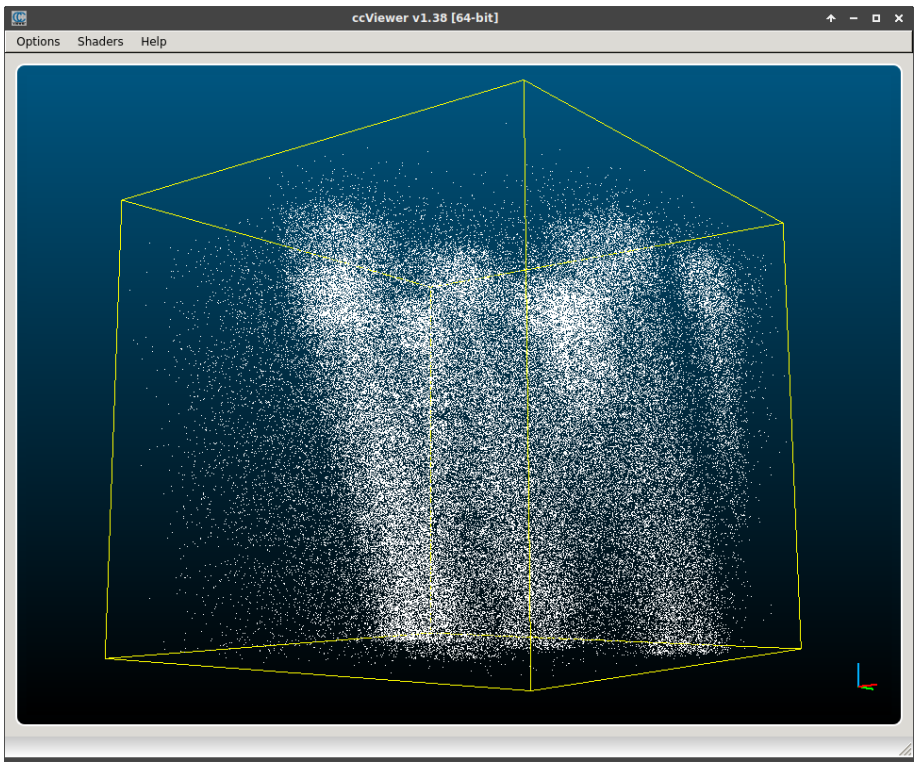

Figure 12: ASCII point list imported into an interactive point cloud processor/viewer (in this case ***ccViewer***, part of the excellent open-source *CloudCompare* toolsuite [CC2020]). The point list was exported by ***pmapdump*** with the **-a** option from the volume map rendered in Figure 11.

8 Acknowledgements

The following institutions contributed to the development of the RADIANCE photon map extension:

Fraunhofer Institute for Solar Energy Systems (ISE) supported by the German Research Foundation (DFG LU204/10-2, *Fassadenintegrierte Regelsysteme* (FARESYS)),

Lucerne University of Applied Sciences and Arts (HSLU) supported by the Swiss National Science Foundation (SNSF 147053, *Simulation based assessment of daylight redirecting components for energy savings in office buildings*),

Tokyo University of Science supported by the JSPS Grants-in-Aid for Scientific Research Programme (KAKENHI JP19KK0115, *A new expression of three-dimensional light flow in architectural spaces and its visual interpretation in the context of human-centric lighting design*).

The Developer(s) would also like to thank the following individuals for their invaluable contributions:

Greg Ward for his support in integrating the code, and his tireless efforts in keeping RADIANCE and its community alive,

Dr. Peter Apian-Bennewitz for initiating development of the photon map at Fraunhofer ISE,

Prof. Stephen Wittkopf and Dr. Lars Grobe for re-igniting development within the scope of the *Daylight redirecting components* project at HSLU,

Prof. Nozomu Yoshizawa and his team for initiating the project *Three-dimensional light flow* at Tokyo University of Science and applying photon mapping in their experiments,

Rob Gugliemetti for his huge help in getting this thing compiled under Windoze and providing nightly builds for that platform, 'cos The Developer(s) doan' do Windoze,

Carsten Bauer (Capt. B) and Dr. David Geisler-Moroder for testing the tool with REALLIFE™ scenes, reporting bugs and providing valuable feedback.

Finally, a big thanks to *anyone* out there nuts enough to try the extension out!

References

[Bent1975] Jon Louis Bentley: “Multidimensional Binary Search Trees Used for Associative Searching”. *CACM* 18(9), 1975.

[CC2020] CloudCompare 3D point cloud and mesh processing software open source project, 2020. Available at http://www.cloudcompare.org.

[Chri2000] Per Christensen: “Faster Photon Map Global Illumination”. *Journal of Graphics Tools* 4(3), April 2000.

[Jens1996] Henrik Wann Jensen: “Global Illumination using Photon Maps”. *Rendering Techniques ’96*, Eds. X. Pueyo and P. Schröder, Springer-Verlag, 1996.

[Jens1997] Henrik Wann Jensen: “Rendering Caustics on Non-Lambertian Surfaces”. *Computer Graphics Forum*, 16(1), March 1997.

[Jens1998] Henrik Wann Jensen and Per H. Christensen: “Efficient Simulation of Light Transport in Scenes with Participating Media using Photon Maps”. *Proceedings SIGGRAPH ’98*, July 1998.

[Jens2000] Henrik Wann Jensen and Niels Jørgen Christensen: “A Practical Guide to Global Illumination using Photon Maps”. *SIGGRAPH 2000 Course Notes 8*, July 2000.

[Ward1994] Gregory J. Ward: “The RADIANCE Lighting Simulation and Rendering System”. *Proceedings SIGGRAPH ’94*, July 1994.

[Ward1998] Greg Ward Larson and Rob Shakespeare: *Rendering with* RADIANCE. Morgan Kaufmann, San Francisco, 1998.

[Schr2003] Roland Schregle: “Bias Compensation for Photon Maps”. *Computer Graphics Forum* 22(4), December 2003.

[Schr2015] Roland Schregle, Carsten Bauer, Lars O. Grobe and Stephen Wittkopf: “EvalDRC: A tool for annual characterisation of daylight redirecting components with photon mapping”. *Proceedings CISBAT 2015*, Lausanne, Switzerland.

[Schr2016] Roland Schregle, Lars O. Grobe and Stephen Wittkopf: “An Out-of-Core Photon Mapping Approach to Daylight Coefficients”. *Journal of Building Performance Simulation*, 9(6), 2016.

[Schr2019] Roland Schregle, Nozomu Yoshizawa, Toshihide Okamoto, Ken Komazawa: “An Ongoing Visual Assessment of Three-dimensional Light Flow Expressed Through Volume Photon Mapping”. Presented at the *18th International RADIANCE Workshop*, New York City, New York, USA, August 2019.

Appendix

A Sample Renderings

A.1 Caustics

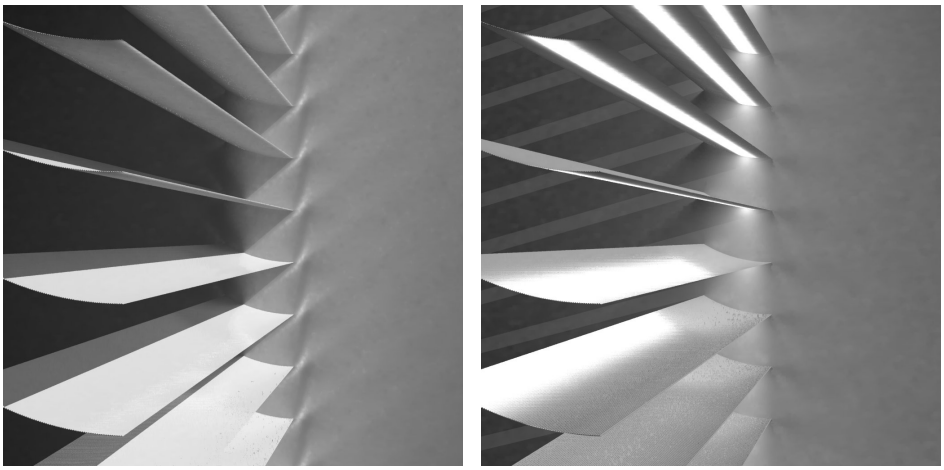

Figure 13: Retroreflecting blinds patented by Helmut Köster, rendered with RADIANCE photon map using 600k global and 6M caustic photons with a bandwidth of 100 and 100–2000 (bias compensated), respectively. At 40° incidence (left), sunlight is retroreflected as evidenced by the caustics on the wall, while skylight is scattered into the interior off the diffuse lamellae undersides. At 20° incidence (right), the inclination requires adjustment as sunlight is now redirected towards the lamellae undersides rather than being retroreflected.

Figure 14: Cross-sectional view of a lightpipe rendered with 500k global and 1M caustic photons. The photons are emitted via ports defined for the window, pipe aperture, and external heliostat (not visible).

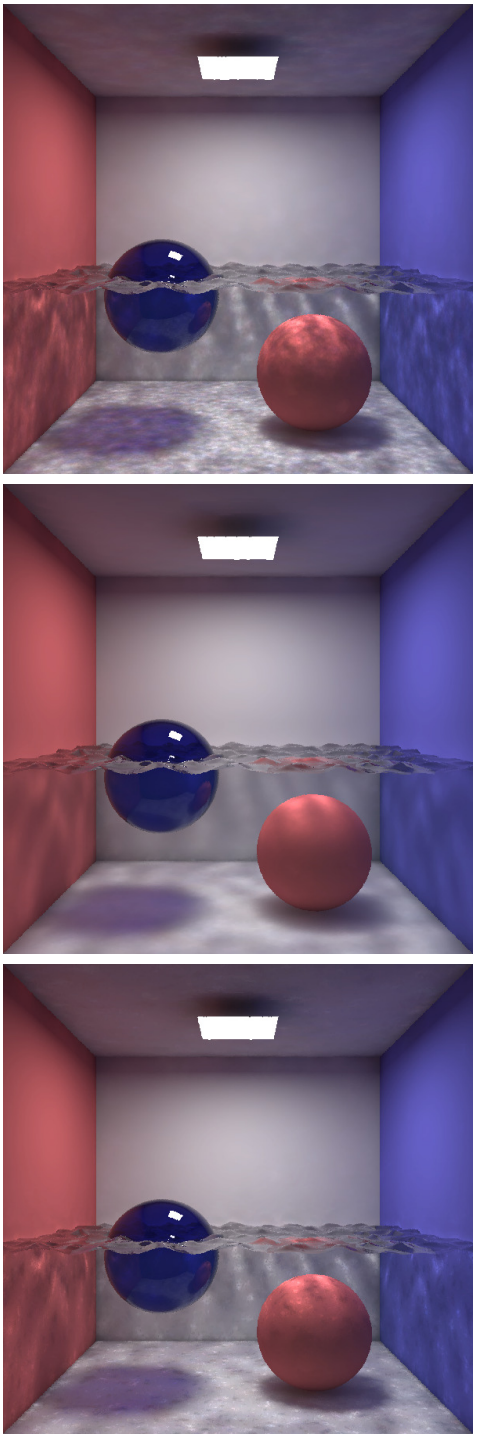

Figure 15: Flooded Cornell box rendered with caustic photon map using fixed bandwidths of 50 (top) and 500 (centre) photons, and a dynamic bandwidth of 50–500 photons with bias compensation (bottom).

A.2 Volume Caustics

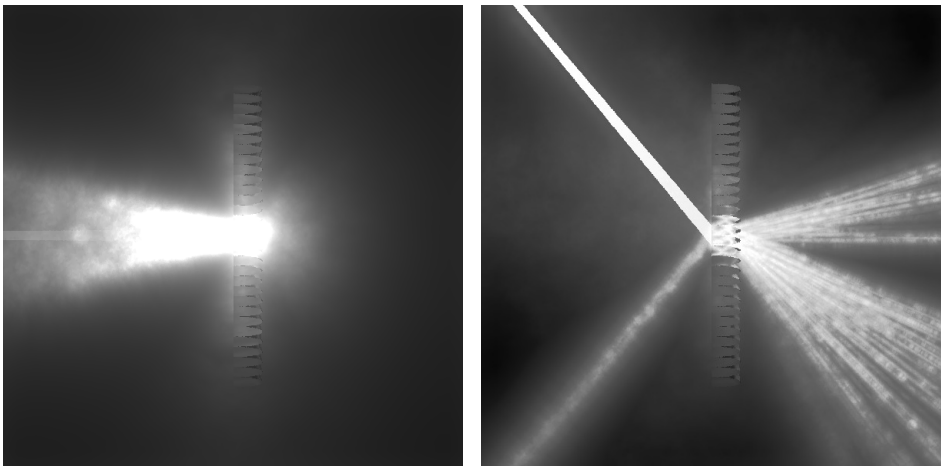

Figure 16: Compound parabolic concentrator with opaque apertures rendered with volume photon map in a *mist* environment. At normal incidence (left), the incident beam is retroreflected, whereas at 50° incidence (right) it is transmitted and redirected in a characteristic fan-out.

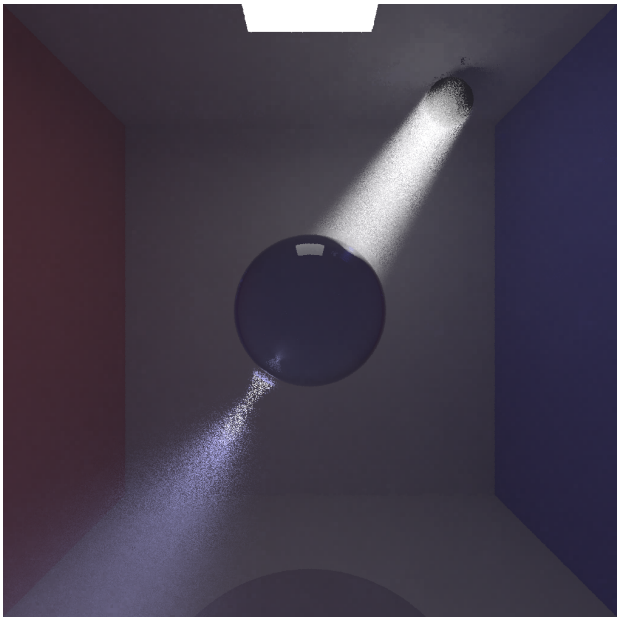

Figure 17: Funky volume caustics in a *mist* environment from refracted volume photons as they pass through a *dielectric* sphere. Note the blue cast imposed on the refracted photons.

B Parameter Ranges

The following table lists typical ranges for various parameters used with ***mkpmap*** and ***rpict***. It is primarily aimed at novices, who are no doubt bewildered by the plethora of options. The ranges are recommendations based on the developer's experience, and should not be interpreted as fixed constraints. In some situations it is not unlikely to use settings far beyond the typical range.

mkpmap parameter	Typical range
nphotons	100k–100M (100k–1G with out-of-core option)
-apD	0.1–1.0
-apm	1000–5000
-apP	0.25–0.5
-dp	1000–100000
-ds	0.0001–0.01
rpict parameter	Typical range
$bwidth_1$	50–100 (50–1000 with out-of-core option)
$bwidth_2$	200–500 (200–5000 with out-of-core option)
-ab	1 (default via ambient bounce), -1 (quick&dirty direct visualisation)
-ac	4.0–16.0
-am	100-1000

C Bugs and Weirdness

- The spotlight focal point is ignored by the photon map; it coincides with the light source surface. This corresponds to specifying a direction vector of infinitesimal length in RADIANCE CLASSIC™.
- ***mkpmap***'s distribution algorithm can exceed the specified number of photons with scenes having high reflectance or when excessively high settings of **-apD** are used.
- Caustic photon irradiance will be biased when passing ***rtrace*** points which do not lie on a surface, unless a virtual receiver surface is defined with *antimatter* and specified with ***mkpmap***'s **-aps** or **-apS** option, as detailed in section 4. This also applies when directly visualising global photons with **-ab 1**. Technically this is not a bug, but a fundamental constraint of the photon mapping algorithm itself.

D *mkpmap* Troubleshooting

Problem	Cause	Solution
mkpmap stores too many photons	Too many emitted photons in distribution prepass	Reduce **-apD**
Number of photons stored differs substantially from specification	Too few photons emitted in distribution prepass; prediction for main pass inaccurate	Increase **-apD**
mkpmap complains about runaway photons	Photons stuck in infinite scattering loop, possibly pathological geometry or materials	Check geometry and materials; increase **-apm**
mkpmap is slow with *source* primitive	*source* emits from scene cube faces	Define photon ports
mkpmap bogs down in light source flux integration	Excessively high probability density function resolution; too many source partitions	Decrease **-dp**; increase **-ds**
mkpmap bails out after too many distribution prepasses	photon ports oriented outward; using caustic photons with nonspecular geometry; using volume photons without *mist*; pathological geometry or nonabsorbing materials; light sources see no geometry	Check port normal(s) or reverse their orientation(s) with **-apo-**; check scene and specify compatible photon types

E *rpict/rtrace* Troubleshooting

Problem	Cause	Solution
Excessive noise in photon irradiance	Bandwidth too low	Increase bandwidth
Excessive blurring in photon irradiance	Bandwidth too high	Decrease bandwidth or increase size of photon map
Excessive noise and blurring in different regions	Bandwidth is fixed	Apply bias compensation for dynamic bandwidth
Severe irradiance falloff at polygon edges	Boundary bias due to high bandwidth	Reduce bandwidth or apply bias compensation
Splotchy ambient illumination from global photons	Excessively noisy density estimates; too few ambient samples; irradiance caching too inaccurate	Check density estimate with **-ab** -1, and increase bandwidth if necessary; increase **-ad** if illumination has high variance or contains caustics; increase **-ar** and/or decrease **-aa**
Direct photon irradiance from modified light source deviates significantly	Probability density function cannot adequately resolve emission distribution; too few source partitions	Increase **-dp**; if distribution varies over surface, decrease **-ds**
Caustics are underestimated for measurement points passed to ***rtrace*** which do not lie on scene geometry	No photons in measurement plane; photons are retrieved from distant coplanar objects, resulting in oversized radius	Define virtual receiver surface with *antimatter* material, and specify its modifier with ***mkpmap***'s **-aps** or **-apS** option when building the photon map
rpict/rtrace complains about short photon lookups	Maximum photon search radius too small	Increase **-am**
Everything is slow	User expects too much	Twiddle thumbs