The photon map extension for Radiance - Applications
Workflow, applications, caveats

Lars O. Grobe

9th Radiance Workshop, Freiburg 2010

22nd September 2010
Overview

1. Motivation

2. Workflow
   - Modelling considerations
   - New and extended commands

3. Caveats and pitfalls

4. Applications
   - LCP, prismatic glazing
   - Blinds
   - Pipes, ducts, collectors
   - Lenses
Radiance is generic and flexible enough to be applied to a wide range of simulation.

mkillum has been of great help modelling complex fenestration systems in Radiance.

Unfortunately, “Rendering with Radiance” page 580 (chapter 13, “Secondary light sources” clearly states:

“[…] 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. […]”
Extensions to Radiance

- low resolution sky models (e.g. Tregenza) allow to use glow sources and avoid some of the limitations
- rtcontrib, genbsdf, genskyvec all operate on rather low angular resolution
- perfectly useful for illuminance on work plane, annual calculations, ...
- but: what about glare, visual comfort, ...?

We still need a forward distribution for scenes where luminance variance is expected to be high...
Workflow overview

**Figure**: Schematic overview on a typical workflow involving photon mapping with Radiance.

**Figure**: Tools used to implement the tasks as described on the left.
the photon map allows to keep the workflow simple - no specialized generators needed

however we can optimize our model to achieve better results

there are still limitations in the photon map that we should at least be aware of...

So how to prepare a model to make it suitable for an extra forward pass?
Zoning

Generate separate photon maps for each separate space

- a map with fixed number of photons / constant memory requirements leads to a higher resolution
- required number of photons and gathering parameters depend on specific conditions in the given space
- re-use the photon map for spaces where no changes apply to, redistribute photons only for affected spaces
Extended sources are inefficient to handle in forward algorithms!

- you can often avoid extended sources by
- replacing transmissive components by sources:
  - apply skyglow to the window pane
  - use mkillum in a first step
- pointing mkpmap at the relevant ports where flux is entering the space:
  - photon ports
Figure: Extended source sky/ground: few photons that are emitted actually reach the interior space.
Figure: A surface in the window opening is marked as a photon port and acts as source for the interior space. All distributed photons reach the interior.
Curved surfaces

Radiance knows little about non-planar surfaces

- sphere, bubble
- cylinder, tube
- cone, cup

Everything else is tessellated!

But:
Many applications of the photon map are based on non-planar geometry - for planar faces we could use mirror and classic Radiance.
Curved surfaces II

Two approaches to model curved surfaces beyond the primitives:
- high-res faceted models (can be exported from CAD or by Radiance geometry generators)
- pseudo-CSG in Radiance...

CSG in Radiance?
- use mixfunc and vary between the surface material and “air”, mathematically switch status of surfaces
- can help avoiding high polygon counts
- do not use void as air material and avoid crashes...
Curved surfaces III

Building a lens from three geometry object

```plaintext
void mixfunc topMod

4 glassMat air "and(if(0.0125-sqrt(sq(Px)+sq(Py)),1,0),Nz)" rayinit.cal
0
0
0
topMod sphere topFace
0
0
4 0 0 -0.0725 0.075
```
Figure: Lenses modeled using spheres in Radiance. In total 16 geometric objects + light source.
mkpmap forward-distributes photons from the light source into the scene

```bash
mkpmap -ap <ptype> <pmapfile> <n> (...) -apo <pport> <octree>
```

- `ptype`: being one of:
  - `g`: global photons
  - `c`: caustic photons
  - `v`: volume photons
  - `p`: precomputed photons
- `pmapfile`: the file name of the stored photon map
- `n`: the estimated number of photons to emit
- `pport`: the name of a modifier marking photon port surfaces
- `octree`: the octree file of the scene

You can create more than one photon maps with one run of `mkpmap` - just append!
mkpmap and photon ports

mkpmap example

mkpmap -apg global.pmap 100000 -apc caustics.pmap 250000
-apo photonPortMat myscene.oct

- create global photon map
  - approx. 100,000 photons
  - save as global.pmap

- create caustic photon map
  - approx. 250,000 photons
  - save as caustics.pmap

- use all surfaces modified by photonPortMat as photon ports
mkpmap example

mkpmap -apg global.pmap 100000 -apc caustics.pmap 250000
-apo photonPortMat myscene.oct

- create global photon map
  - approx. 100,000 photons
  - save as global.pmap

- create caustic photon map
  - approx. 250,000 photons
  - save as caustics.pmap

- use all surfaces modified by photonPortMat as photon ports
pmapdump usage

pmapdump visualizes photon locations stored in photon maps as spheres

\[ \text{pmapdump} \ <\text{radius}\> \ <\text{pmapfile}_1\> \ (...) \ <\text{pmapfile}_N\> \ > \ <\text{radfile}\> \]

- **radius**: radius of spheres used to visualize recorded photon locations
- **pmapfile$_1$**: first photon map file
- **(...)**
- **pmapfile$_N$**: Nth photon map file
- **radfile**: scene file with photon locations shown as spheres

Photon types indicated by color: global caustic volume
pmapdump example

```
pmapdump .005 global.pmap caustics.pmap dump.rad
objview dump.rad

- create file dump.rad
- add one sphere of radius .005 at each photon location stored in global.pmap
- add one sphere of radius .005 at each photon location stored in caustic.pmap
```

oconv magic applies: overlapping geometry is evil - small spheres preferred...
Figure: Caustic photons marked by red spheres. Quick preview using objview.
Basic rpict/rtrace/rvu usage

```
rpict -ap<pctype> <pmapfile> <n> <options> <octree>
```

for each pmap-file to use:

- `ptype` photon type (g|c|v)
- `pmapfile` photon map file
- `n` amount of photons to consider (“gather”) for irradiance calculation at a point

and

- `options` remaining rpict/rtrace options as known
- `octree` octree file of the scene
rpict/rtrace/rvu ambient calculation

- **-ab -1** directly visualize recorded global photon locations
- **-ab 0** use photon density around point of interest to calculate irradiance
- **-ab 1 or more** calculate irradiance at point of interest by photon density over one ambient bounce

Lars O. Grobe

The photon map extension for Radiance - Applications
### Basic rpict/rtrace/rvu example

#### rpict example with global & caustic photons

```
rpict -ab 1 -apg g.pmap 10 -apc c.pmap 25
-vp 0 0 1.5 -vd 1 0 0 myscene.oct > myscene.unf
```

- **-ab 1** calculate diffuse indirect irradiance from global photons by one ambient bounce (does not affect other photon types)
- **-apg g.pmap 10** at each point where global photon density is to be assessed, gather nearest 10 photons from global.pmap
- **-apc c.pmap 25** at each point where caustic photon density is to be assessed, gather nearest 25 photons from caustic.pmap

**<second line>** classic rpict options, render a view at origin pointing along x+ to file myscene.unf
Irradiance calculations are possible only on geometry!

- Irradiance is calculated by photon gathering (counting nearest photons to a point)
- Photon locations are stored at photon path intersections with geometry only
- Photon distribution is outside rpict/rtrace/rvu - a sensor concept with adding an imaginary surface does not work
**void modifier applied to geometry**

- in classic Radiance, modifying an object with void switches it off
- with the photon map, this leads to a crash
- define an air material instead (using trans, dielectric, antimatter...)

**file handling**

- mkpmap does not overwrite existing files
- do not forget to remove stored photon maps after any relevant changes to the scene
- mkpmap is not directly supported by rad
Photon ports

- interreflection calculation does not take place outside photon ports
- light reflected onto the photon port by external surfaces is ignored
- possible workarounds are pregenerated illums that act as sources and are considered

*Figure:* Other than sources visible from the photon port, light reflected from outside objects is not accounted for.

**Surface orientation**

the photon port surface’s normal must points towards receiving side (mkillum behaves the other way around...)
What was not covered

- less common photon types:
  - **precomputed photons**: photon gathering happens with photon distributions, energy of given fraction of photons is stored in the photon map
  - **volume photons**: photon interaction with participating media ("mist" in Radiance)
  - **direct photons**: mostly for debugging...

- **bias compensation** to balance noise and blurring
- **advanced options** to control forward distribution
Figure: Laser cut panels have been included into double glazing to redirect light towards the ceiling. In a backwards raytracing algorithm, a huge amount of samples is necessary to consider the very different ray paths that neighboring rays follow.
LCP model

Figure: beams passing a laser cut panel (LCP) at 45, 30 and 0 degrees altitude.
Figure: Non-planar specular louvers are a typical redirecting system that can not be properly modeled using classical Radiance backwards raytracing at sunny sky conditions. The stochastic sampling of the ambient calculation would miss the sun in most cases.
Blinds model

- model was created using genblinds
- surfaces are assumed to be purely specular reflective

Figure: Beams passing two non-planar, specular louvers (blinds) at 45, 30 and 0 degrees altitude.
Comparison classic Radiance, pmap

Figure: Window with specular, non-planar blinds. Simulation using classic Radiance backwards raytracing.

(Images by Fraunhofer ISE)
Light pipes / ducts

- Systems that transport light by multiple internal reflections on specular surfaces.
- Maybe equipped with optimized collectors, sun tracking, ...
- Can not be efficiently modeled in classic Radiance, unless:
  - covered by some diffusing material on the top
  - under overcast sky conditions
  - at reduced angular resolution (e.g. Tregenza patch model)

Figure: Schematic section through a light pipe assembly.
To be considered:

- Ducts of circular and rectangular cross-sections
- Frontside mirrors, backside mirrors, TIR
- possibly additional mirror assemblies at entrance and exitance ports

**Figure:** Beams passing a light duct at 45, 30 and 0 degrees altitude.
- One exotic application for Radiance...
- In lighting: not-so-diffuse diffuser, such as lens plates?

**Figure**: Beams passing a spherical lens.
Questions

Questions? Answers? Ideas?