The Radiance rtcontrib Program

Greg Ward
Anyhere Software
Quantify Contributions

Q: How much came from each light bank?
Background

- Core *Radiance* rendering routines recursively evaluate radiance, hence the name.
- Potentially useful information about where light originates is lost during this process.
- Prior to version 3.7, there were two solutions:
  - Repeat rendering for each source (costly).
  - Switch to *Daysim* (daylight coefficients only).
Method

• New member in RAY structure for storing current ray coefficient (3 floats for RGB)

• Minor change to evaluation ordering in Radiance rendering routines

• Function for multiplying ray coefficients back to the root of the tree (i.e., the PRIMARY ray)

• Improvement to -aa 0 speed & accuracy

• New ‘T’ and ‘W’ options for rtrace -o
### Example Code Change

#### Diff for /ray/src/rt/normal.c between version 2.49 and 2.50

<table>
<thead>
<tr>
<th>Line 258</th>
<th>Line 258</th>
</tr>
</thead>
<tbody>
<tr>
<td>/* transmitted ray */</td>
<td>/* transmitted ray */</td>
</tr>
<tr>
<td>if (((nd.specf&amp;&amp;((SP_TRANISP_PURE</td>
<td></td>
</tr>
<tr>
<td>RAY lr;</td>
<td>RAY lr;</td>
</tr>
<tr>
<td>if (rayorigin(&amp;lr, r, TRANS, nd.tspec) == 0) {</td>
<td>copycolor(lr.rcol, nd.mcolor); /* modified by color */</td>
</tr>
<tr>
<td>VCOPY(lr.ddir, nd.pdir);</td>
<td>VCOPY(lr.ddir, nd.pdir);</td>
</tr>
<tr>
<td>rayvalue(&amp;lr);</td>
<td>rayvalue(&amp;lr);</td>
</tr>
<tr>
<td>scalecolor(lr.rcol, nd.tspec);</td>
<td>multicolor(lr.rcol, lr.rcol);</td>
</tr>
<tr>
<td>multicolor(lr.rcol, nd.mcolor); /* modified by color */</td>
<td>multicolor(lr.rcol, lr.rcol);</td>
</tr>
<tr>
<td>addcolor(r-&gt;rcol, lr.rcol); transtest *= bright(lr.rcol);</td>
<td>addcolor(r-&gt;rcol, lr.rcol); transtest *= bright(lr.rcol);</td>
</tr>
<tr>
<td>transdist = r-&gt;rot + lr.rt;</td>
<td>transdist = r-&gt;rot + lr.rt;</td>
</tr>
</tbody>
</table>
Contribution Coefficients

- A “contribution coefficient” is the fraction of a ray’s return value that will ultimately apply.
  - This is closely related, but not equal, to the “ray weight” reported by `-otw`.
  - ‘T’ option for `rtrace -o` traces to light sources.
  - The ‘W’ option reports contribution coefficient.
A Simple Example

Diffuse = 25%
Specular = 3%

Diffuse = 40%
Specular = 0%

Eye

Src

- $\sigma_{TW} = 0.25 \Omega_1$
- $\sigma_{TW} = 0.03 \times 0.40 \Omega_2$
- $\sigma_{TW} = 0.03$

$\sigma_{TW} = 1.0$
Problem: Daughter Rays
Diffuse Interreflections
Solution: Gather Rays

• Need a general method to gather contribution coefficients and sum them together

• Different applications require different sums:
  • Daylight coefficients sum at sky patches
  • Luminaire model may sum at lamp surface

• How do we do it all?
Enter `rtcontrib`

- Manage the calculation of ray contribution coefficients by `rtrace`
- Gather contributions for use in linear light combinations (e.g., daylight coefficients)
- Facilitate analysis of optical systems such as light pipes and luminaires
- Provide flexible output (ASCII and binary data as well as *Radiance* pictures)
General Operation

• User specifies **rtrace** options and octree

• User tells **rtcontrib** where to collect values
  
  • required modifier name(s)
  
  • optional bin number based on ray direction and intersection point

• Output sent to one or more files or commands
  
  • specified by modifier name and bin number
Lighting Example

vwrays -ff -x 1024 -y 1024 -vf model.vp \
| rtcontrib -o part_%s.pic -m fluor1 -m fluor2 \n-ffc `vwrays -d -x 1024 -y 1024 -vf model.vp` -u+ model.oct
rtcontrib Options

General options:

- \( n \) \( N \) start \( N \) rttrace processes
- \( r \) recover previously aborted calculation
- \( e \) expr compile definitions string
- \( f \) source compile definitions file

} Used by \( -b \)

Modifier options:

- \( o \) ospec output specification
- \( b \) binv bin number

May contain ‘!’ and ‘%d’ or ‘%s’
Integer expression, or ‘0’ to disable

Modifier specification:

- \( m \) mod modifier name
- \( M \) file modifier list from file
Lighting Example Dissection

vwrays -ff -x 1024 -y 1024 -vf model.vp \ | rtcontrib -o part_%s.pic -m fluor1 -m fluor2 \ -ffc `vwrays -d -x 1024 -y 1024 -vf model.vp` -u+ model.oct
**Lighting Example Dissection**

**vwrays** provides primary ray origins and directions (in floating point) for pictures to be generated by rtcontrib

```
vwrays -ff -x 1024 -y 1024 -vf model.vp \ | rtcontrib -o part_%s.pic -m fluor1 -m fluor2 \ -ffc `vwrays -d -x 1024 -y 1024 -vf model.vp` -u+ model.oct
```

Second invocation reports actual resolution (\(-x 1024 -y 690\))
Lighting Example Dissection

Specifies output files and associated modifiers, creating `part_fluor1.pic` and `part_fluor2.pic`.

```
vwrays -ff -x 1024 -y 1024 -vf model.vp \ | rtcontrib -o part_%s.pic -m fluor1 -m fluor2 \ -ffc `vwrays -d -x 1024 -y 1024 -vf model.vp` -u+ model.oct
```

The `-ffc` option is an `rtrace` option telling `rtcontrib` to expect single-precision floats on input and produce RGBE colors on output. The `-u+` option specifies pure Monte Carlo sampling.
Discrete solar positions at each patch location improve rendering of sharp shadow features. Actual sun position is interpolated from four closest suns.

From John Mardaljevic’s thesis.
Daylight Coefficient Example

Blinds: up, top, down
@ 10° increments
Optional overhang

Upper & lower glass:
42 separate runs
of 146 sky patches
& 145 solar patches
One hemispherical fisheye view
Example Contributions (1)

Sky patch 045 from lower glass
no overhang
no blinds

Sun patch 045 from lower glass
no overhang
no blinds
Example Contributions (2)

Sky patch 045 from lower glass with overhang blinds @ 20°

Sun patch 045 from lower glass with overhang blinds @ 20°
Dec 28
Overhang
10 am
blinds down
@ 10°
• I hope to be working in the future
• Apply \texttt{rtcontrib} to optical problems
  • BTDF simulations
  • Luminaires, light pipes, etc.
• Integrate with energy simulation tools