The Radiance \textit{rtcontrib} Program

Greg Ward
Anyhere Software
Basic Idea

- Normally, details about what sources contributed to a pixel are thrown away.
- Saving this information allows different source contributions to be substituted.
- More generally, the flux transfer between a point and any other point may be quantified.
Quantifying 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)
A Simple Example

Eye

- $\omega_{\text{TW}} = 1.0$

Src

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

Diffuse=40%
Specular=0

Diffuse=25%
Specular=3%
Problem: Daughter Rays
Diffuse Interreflections
Central Concept
Understand This First!

• Rays of interest are fed as input, as for \texttt{rtrace}
  • view rays for image, illuminance points, etc.

• Output is determined by “catch surfaces”
  • defined via \textit{modifier names} and \textit{bin expressions}

• this part is new, and can be a bit tricky
rtcontrib: Gathers Rays

- Gather contributions & coefficients and sum them up logically
- Different applications require different sums:
  - Daylight coefficients sum at sky patches
  - Luminaire model may sum at lamp surface
- rtcontrib provides general mechanism, while rtrace handles actual ray-tracing
General Operation

- Looks like `rtrace` command, similar options
- Options tell `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 -V+ -o part_%s.pic -m fluor1 -m fluor2 \
-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

Modifier options:
- **-o** *ospec* output specification
- **-b** *binv* bin number

Modifier specification:
- **-m** *mod* modifier name
- **-M** *file* modifier list from file

Used by **-b**

May contain ‘!’, ‘%d’ or ‘%s’

Integer expression, or ‘0’ to disable
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` 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` 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`.

```sh
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` model.oct
```

The `-ffc` option is an `rtrace` option telling `rtcontrib` to expect single-precision floats on input and produce RGBE colors on output.
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
Example with Captured Sky

- Parthenon model created by Paul Debevec & Co. at USC’s ICT from 100’s of site laser scans
- Los Angeles sky captured by Jessi Stumpfel on the one cloudy day she could find
- `rtcontrib` run plus `pccomb` sums for 690 animation frames done in 6 hours on G5 Quad
Parthenon Rendering Method

- Run `rtcontrib` to compute each Tregenza sky patch DC contribution as a partial image
  - Takes the time of a single MC rendering
- Use low-resolution version of captured sky to compute Tregenza patch radiances
- Use radiance as coefficients in `pcomb` command to combine partial images
Actual Commands

rtcontrib -x 2048 -y 1361 -ffc -ab 2 -ad 1024 -f tregenza.cal -b tbin -o p%d.hdr -m white -n 4 parth.oct < rays.flt

Then, for each frame time:

mksky.csh $time | oconv - > capsky.oct

rtrace -h -w -dv- capsky.oct < tregsamp.dat | total -64 -m > pval.dat

pcomb -h `rcalc -o '-s ${$1} p${recno-1}.hdr' pval.dat` > f$time.hdr
Example pcomb Command

```bash
```
Using **dctimestep**

```
dctimestep p%d.hdr pval.dat > f10-38.hdr
```

*Much faster!*

This didn’t exist when the animation was done.
Computing Caustics (1)

Send rays from “sun”

Accumulate results on 5 submerged pool surfaces
Computing Caustics (2)

vwrays -pj .6 -vf pool_par.vf -x 500 -y 500 -ff \n  | rtcontrib -w -ffc -c 0 \n      -f pool_coords.cal -o %s_caustics.pic \n      -b floor_bin -bn 7500 -x 50 -y 150 -m floor \n      -b s_wall_bin -bn 1500 -x 50 -y 30 -m s_wall \n      -b n_wall_bin -bn 1500 -x 50 -y 30 -m n_wall \n      -b e_wall_bin -bn 4500 -x 150 -y 30 -m e_wall \n      -b w_wall_bin -bn 4500 -x 150 -y 30 -m w_wall \n      catchscene.oct

{ south wall bins }
SWallXres : 50;
SWallZres : 30;
SWallWidth : 5 {meters};
SWallOrigX : -2.5;
SWallHeight : 3 {meters};
SWallOrigZ : -3.08;
s_wall_bin = floor(SWallXres*(Px - SWallOrigX)/SWallWidth) +
  floor(SWallZres*(1 - (Pz - SWallOrigZ)/SWallHeight))*SWallXres;
void colorpict s_wall_pat
7 red green blue s_wall_caustics.pic . (Px--2.5)/3 (Pz--3.08)/3
0
0

s_wall_pat glow s_wall
0
0
4 5 5 5 0
“Forward Ray-Tracing”

- The caustics example shows one way \texttt{rtcontrib} can act like a photon-mapper.

- It would be nice to extend this to more general situations.

- \texttt{genBSDF} does this for surfaces, what about light pipes, etc?

- Likely future work.
The Care and Feeding of \texttt{rtcontrib}

- If you use \texttt{-b}, must use \texttt{-bn} as well
- Careful where your place \texttt{-m} and \texttt{-M} options
- Tracking absolute flux values using \texttt{-c 0} requires careful distribution of incident rays
- Remember that contributions are triplet values
- Learn the difference between \texttt{-V-} and \texttt{-V+}
- Options must be appropriate to pure MC
What Is Pure Monte Carlo?

- No indirect cache (-aa 0)
- Uncorrelated sampling (-u+)
- Russian roulette termination (-lr ≤0)
- Other recommended settings:
  - -as 0 -dt 0 -dj 0.9 -sj 1
  - -lw setting very important: 1/#paths
No Ambient Cache?

- `rtcontrib` needs to know at any point what contribution will ultimately be made.
- Ambient values are stored and reused later in an untraceable way.
- `Daysim` avoids this issue by storing daylight coefficients in each ambient value.
- `rtcontrib` more general and memory-efficient.
Multiprocessing in rtcontrib

- Supported with \texttt{-n} option
- This has been greatly sped up and improved in the 4.2 replacement command, \texttt{rcontrib}
- No calls to \texttt{rtrace} and little overhead
- Using \texttt{-c} option further improves scalability
Since many components are often desired, the number of open files may be a constraint.

- Raise max. open descriptors/process:
  - `ulimit -n 1024`, or
  - `limit descriptors 1024`

- This may still be inadequate in some cases....
Recovery Options

• If output is to binary file(s) via \(-o\) option, &...

• Exact command is re-issued with \(-r\) option,

• Then \texttt{rtcontrib} attempts to append data

• So make sure old process is dead, first!

• If \(-c\ 0\) option is in effect, sums are updated

  • very useful for progressive calculations