Recent Code Changes and Additions, Including the *Radiance* Occluder Cache

Greg Ward, Anyhere Software

Is There a New Release?

- The HEAD release is the currently recommended version of *Radiance*
- We need to make an official 3.6 release, which is long overdue
- We're waiting for a stable Windows compile with SCONS
- Is incomplete Windows version acceptable?
  - Missing would be rvu, rholo, ximage, etc.

What Is New in HEAD

- **ra_tiff** supports for 16-bit and IEEE float
- **rad -N** option for multiprocessing
- Increased default rendering parameters
- **oconv** made more robust to tiny details
- **rcalc** float/double support and “passive mode” for inline file editing
- Created **ra_bmp** image converter
- Created occluder cache (more later)

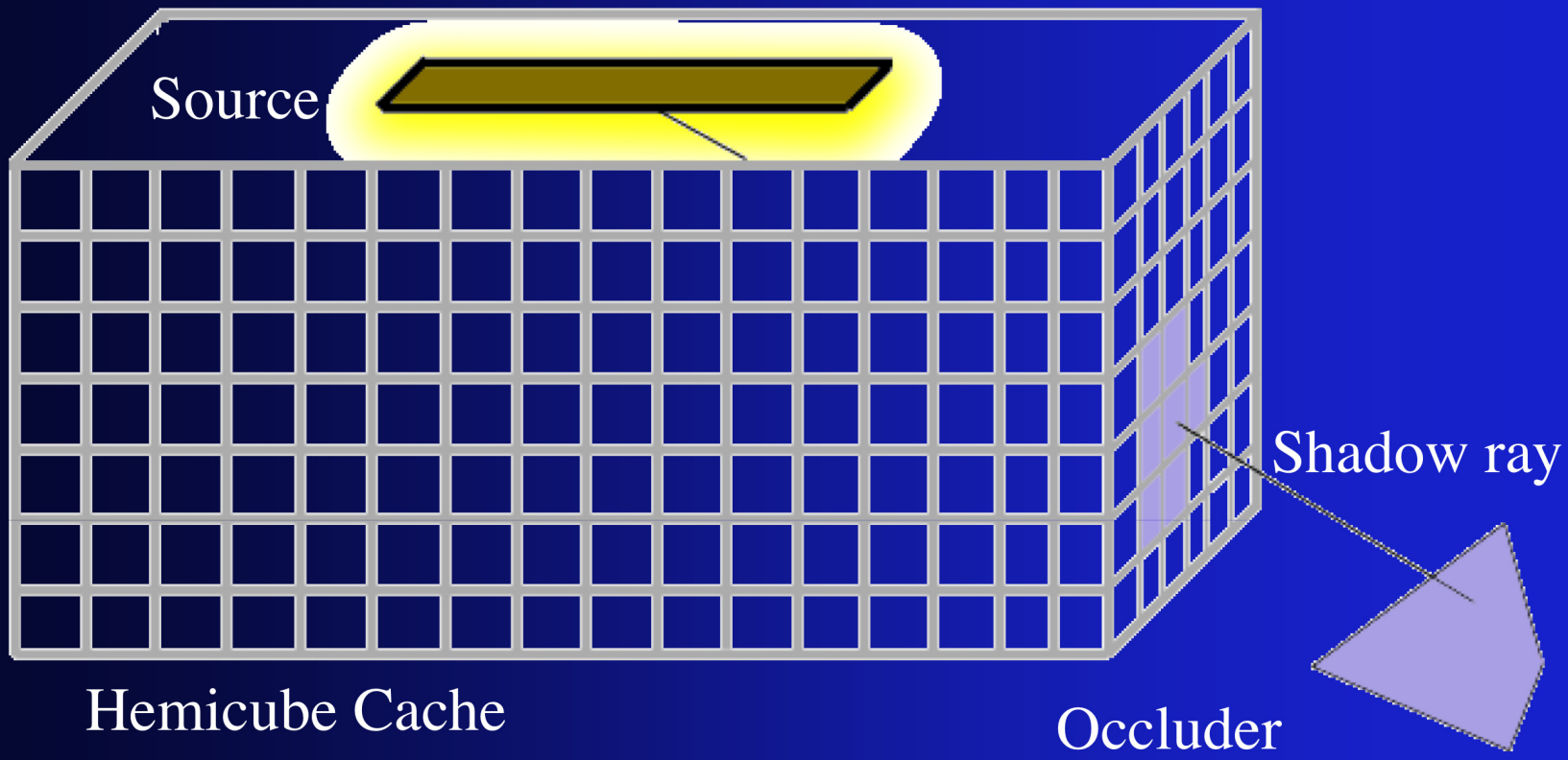
What's New (cont'd)

- Renamed conflicting programs
  - **rview -> rvu** (with link back to original)
  - **calc -> icalc**
  - **lam -> rlam**
  - **gencat -> gencatenary**
- **mkillum -n** option for multiprocessing
- Improved normalization of standard Gaussian reflectance model

Radiance Occluder Cache

- Basic Idea:
  - Record last light blocker for each source
  - Test it for shadowing before others
- Refined Idea:
  - Subdivide directions for each light source
  - Initialize cache with blockers nearest source
  - Preemptive check in selective shadow testing

Example Occluder Cache



Why Does It Work?

- An occlusion anywhere along a ray's length is sufficient -- does not need to be first
- Small occluders in cache get replaced by bigger ones, which have more longevity
- Preemptive testing saves time by avoiding potential contribution calculations
- Setting **-dt 0** no longer a "killer"

Cache Considerations

- Spherical sources use cube r.t. hemicube
- Distant sources use rectangle sized by global bounding volume (octree)
- Each cache is initialized from light source
  - Improves odds for large, nearby occluders
  - Traces about 2000 rays/source at startup
- No associated command-line options

When Doesn't It Work?

- When occluder is not an opaque surface
  - Disabled for translucent materials
  - Disabled for *mesh* and *instance* objects
- When source is subdivided for adaptive shadow testing
  - Destroys cache coherency
- For virtual sources and spotlights
  - Narrow angles wasteful of cache space

Anecdotal Performance

- I have seen 20-50% speedup with scenes containing hundreds of light sources
- Big changes comparing **-dt 0** before/after
  - Valuable for high-accuracy calculations
- Most of the speedup comes from avoiding potential contribution calculations
  - Actual shadow testing not that much faster
- Perform your own comparisons
  - Switch off with -DSHADCACHE=0

Next Steps

- Investigating methods for automatic illum generation based on efficiency considerations
- Coding assistance needed with Windows porting of *Radiance* GUI programs
  - A few volunteers already -- assignments?