Multiresolution Radiosity Caching for Global Illumination in Movies

Per Christensen George Harker
Jonathan Shade Brenden Schubert
Dana Batali

Pixar Animation Studios

SIGGRAPH 2012

PIXAR

Background

- Global illumination is used more and more in CG movie production
  - faster lighting design, more realistic lighting
- Existing methods: distribution ray tracing, path tracing, point-based GI
- Present new caching strategy to make distribution ray tracing more efficient

Motivation

- The bottleneck in ray-traced global illumination is not raw ray tracing speed, but evaluating shaders:
  - displacement, light source, surface shaders
  - texture map lookups, procedural texture gen, shadows, brdf eval, shader overhead, external plug-ins, ...
- Let’s try to reuse shader results where possible

Outline

- Related work
- Multiresolution radiosity cache
- Shading system interface
- Global illumination examples
- Other applications: shadows, subsurface

Related work: distrib. ray tracing

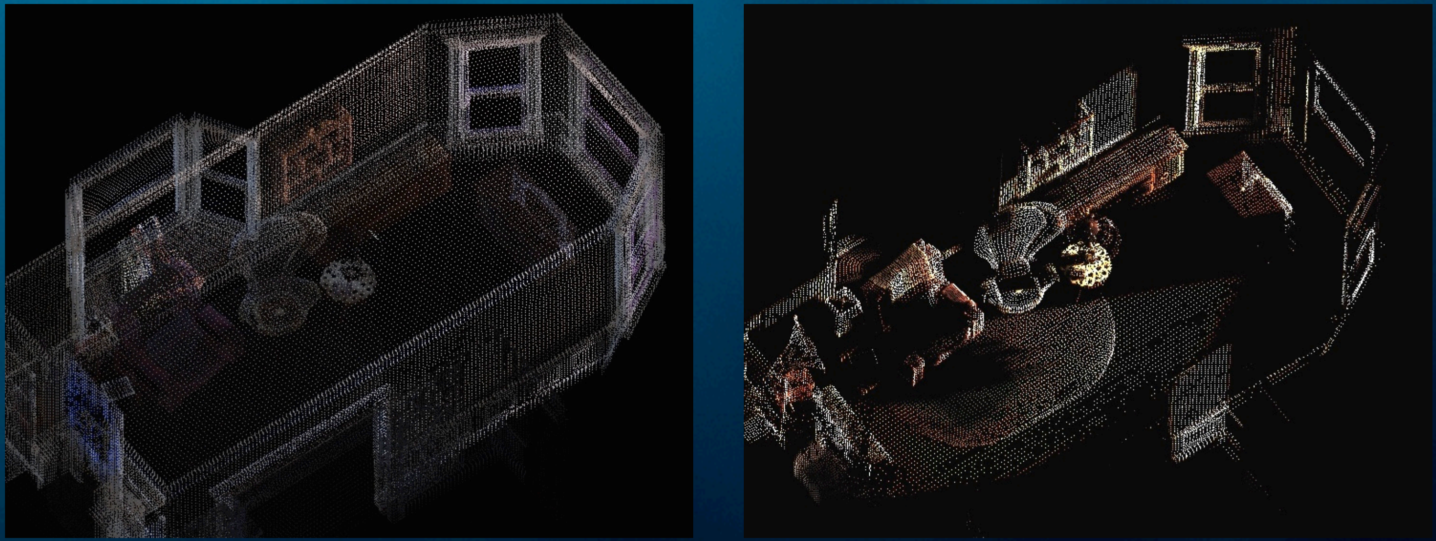
- Store direct illum in textures; distribution ray tracing: 1-bounce GI
- Pros: few shader evals
- Cons: many texture files, multipass, 1-bounce
- Used on ‘Shrek 2’ and other movies [Tabellion04]

Related work: path tracing

- Shoot many rays/pixel; only 1 ray deeper
- Pros: simple algorithm, very general, single pass, fast interactive feedback
- Cons: many shader evals, noisy, slow convergence
- Used on ‘Monster House’ and other movies [Fajardo10]

Related work: point-based GI

- Store point cloud(s) of direct illum



- Compute indir illum by rasterizing points and clusters [Christensen08,Ritschel09]

Related work: point-based GI

- Pros: fast, noise-free
- Cons: file I/O, multipass, not ideal for interactive
- Used at many studios, >40 movies

- Part of Pixar pipeline: `Up’, `Toy Story 3’, ...

PIXAR

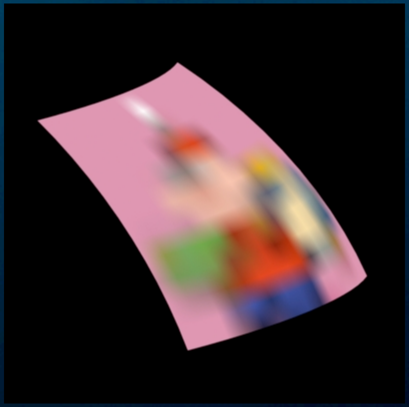
Multiresolution radiosity caching

- Best properties:
  - single pass: interactive
  - few shader evals (caching)
  - multi-bounce GI
  - fixed cache size
  - no file I/O

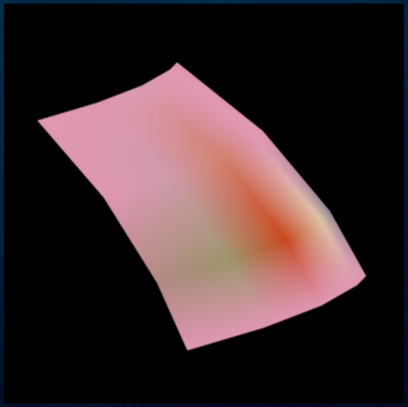
PIXAR

Radiosity cache structure and data

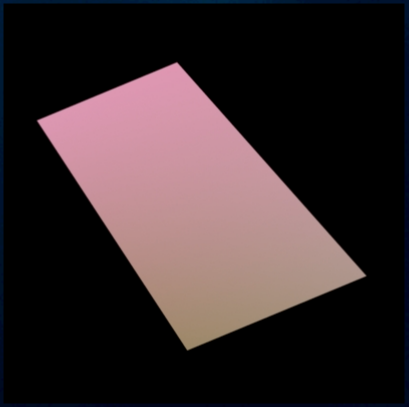
- Radiosity of surface patch at 3 resolutions:
  - – fine: every REYES shading point
  - – medium: every 4th shading point
  - – coarse: 4 corners



fine: 15x12



medium: 5x4



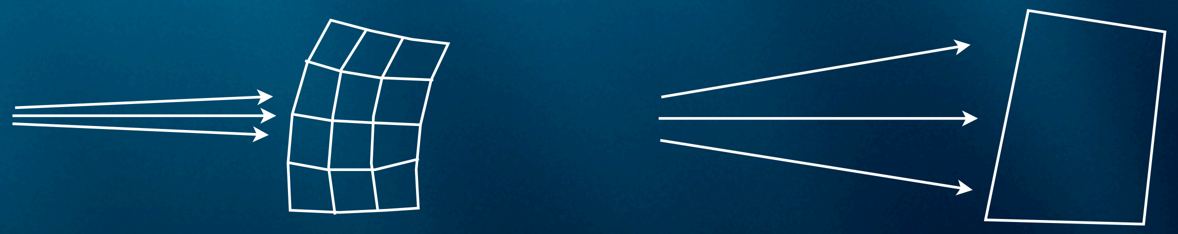
coarse: 2x2

Radiosity cache structure and data

- Cache entry identifier:
  - patch number
  - diffuse ray depth (for multi-bounce)
  - motion segment (for motion blur)
  - surface side
  - timestamp (for interactive applications)

Radiosity cache lookups

- Use ray differentials [Igehy99] to select cache resolution



- Similar to multires tessellation cache

PIXAR

Shading system interface

- The renderer needs to call shaders for information. Previous shader methods:
  - displacement
  - opacity
  - prelighting (calc textures etc)
  - lighting
  - postlighting (e.g. non-linear mapping)

PIXAR

Shading system interface

- New shader method:
  - diffuselighting
- Computes view-independent part of lighting: diffusely reflected light from surface (radiosity)
- Result stored in radiosity cache

Rendering

- Run shader at REYES shading points
- Shoot diffuse rays to sample indirect diffuse illum (“color bleeding”)
- At ray hit points: use cached radiosity or run diffuselighting and cache result
- Radiosity is computed on demand and reused many times

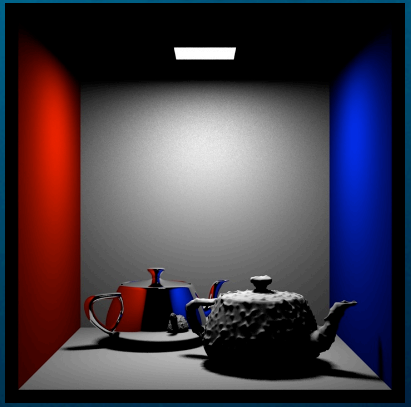
Global illum results: Cornell boxes

- Images at 1k resolution
- Times for 8-core PowerMac
- 8 GB memory
- Default radiosity cache size: 100 MB

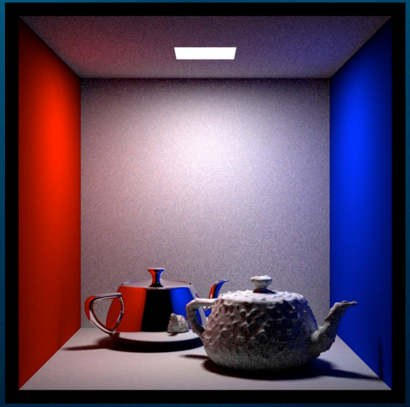
Global illum results: preview quality

- 4 shadow + 4 diffuse rays / shading point

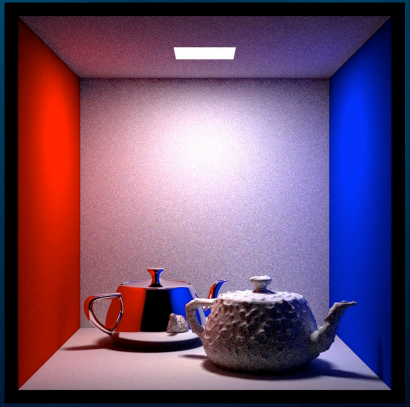
0 bounce



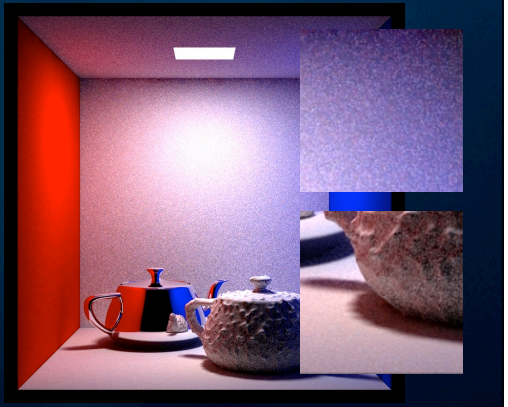
1 bounce



2 bounces



3 bounces



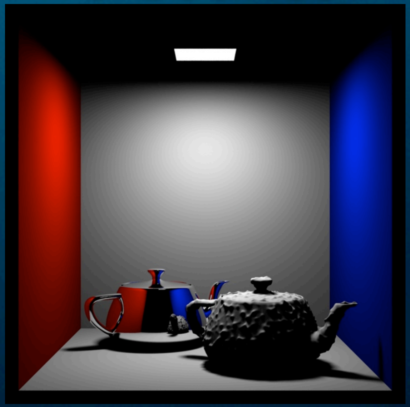
w/o:	4 sec	38 sec	2.3 min	7.6 min
cache:	4 sec	10 sec	11 sec	13 sec
speedup:	-	3.8x	13x	35x

PIXAR

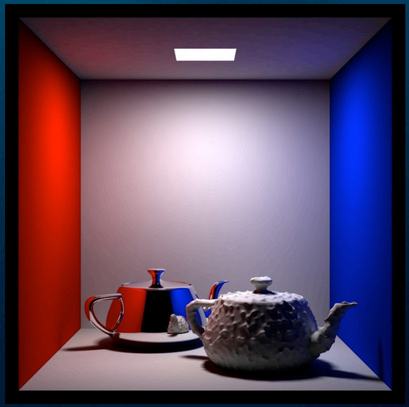
Global illum results: final quality

- 64 shadow + <1024 diff rays / shading pt.
- Irradiance interpolation [Ward92]

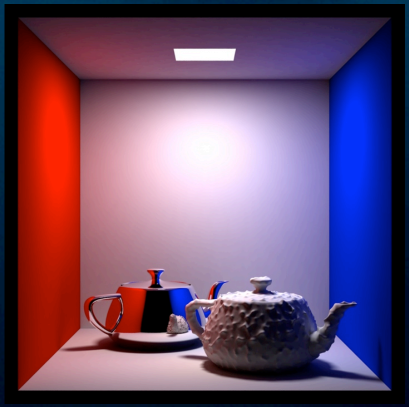
0 bounce



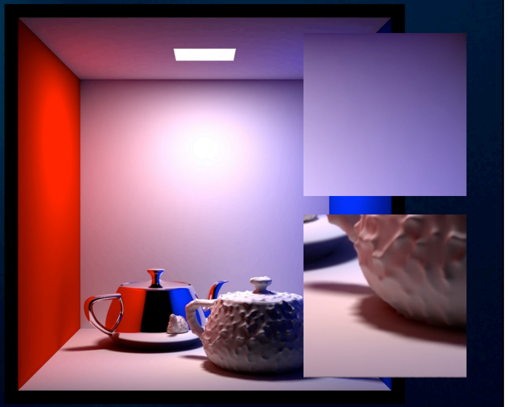
1 bounce



2 bounces



3 bounces



w/o:	22 sec	13.2 min	2.3 hours	>17 hours?
cache:	22 sec	64 sec	99 sec	127 sec
speedup:	-	12x	84x	?x

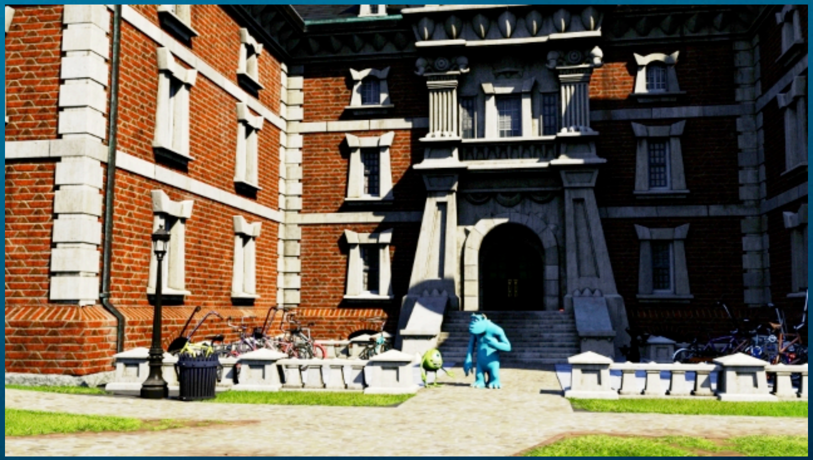
Global illum results

- Cornell boxes are simple
- Even higher speedups for:
  - – more complex shaders
  - – more light sources
- More realistic results: Monsters U.

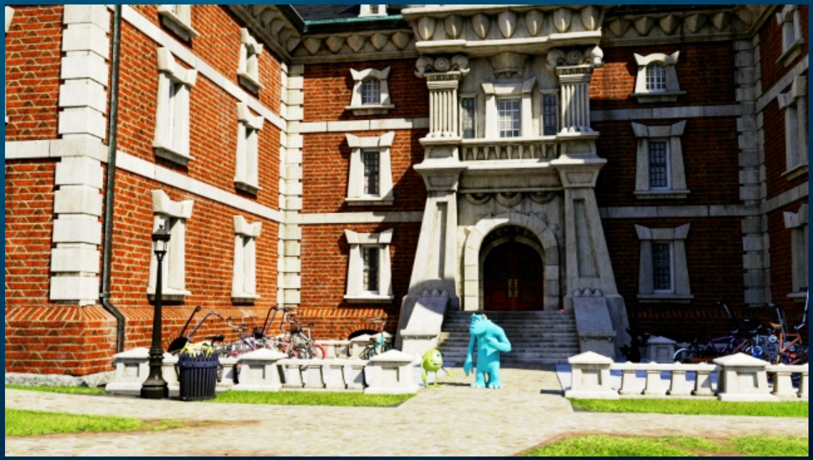
Global illum results: Monsters U.

- Images at 1920x1080 resolution
- Times for 4 cores
- 12 GB memory
- Radiosity cache size 256 MB

Global illum results: Monsters U.

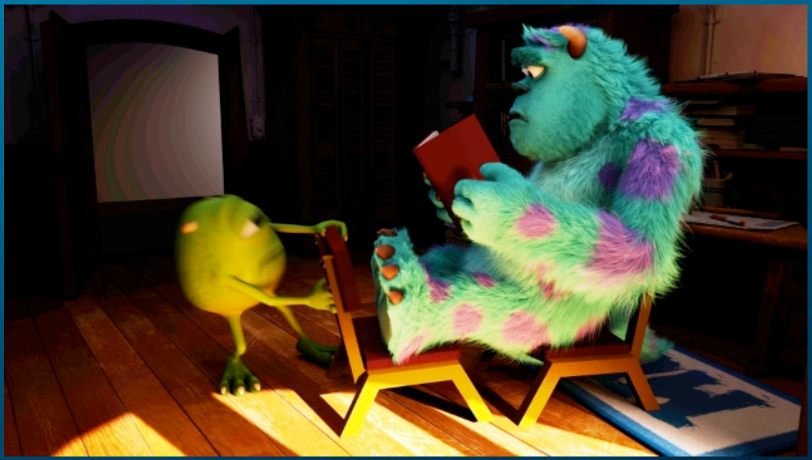


direct: 25 min
(2 lights)

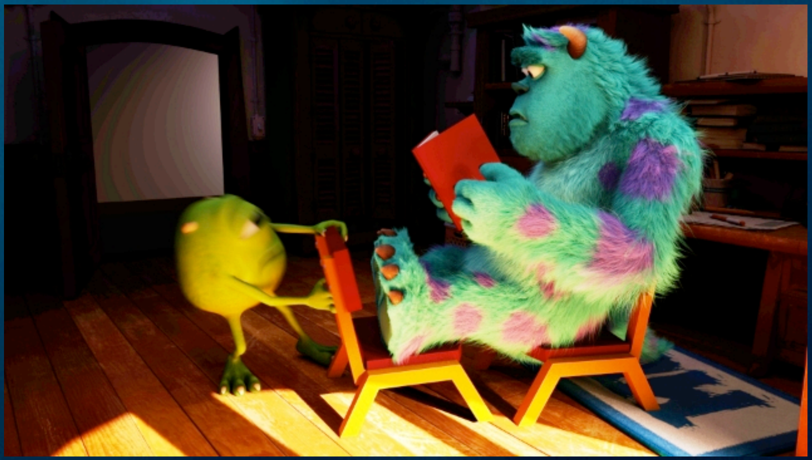


global: 1.5 hour
33x slower w/o cache

Global illum results: Monsters U.



direct: 2.3 hours
(20 lights)



global: 4 hours
41x slower w/o cache

Global illum results: Monsters U.

- Speed for different radiosity cache sizes:
  - 0 MB : 102 hours
  - 16 MB : 36 hours
  - 64 MB : 1.3 hour
  - 256 MB : 1.1 hour
  - 1 GB : 1.1 hour

PIXAR

Global illum results: Monsters U.

- GI reduced #lights from 100s to 10-20
- Radiosity caching chosen over point-based GI because of interactivity (lighting turn-around)
- Master lighting: 4 weeks -> 2 weeks

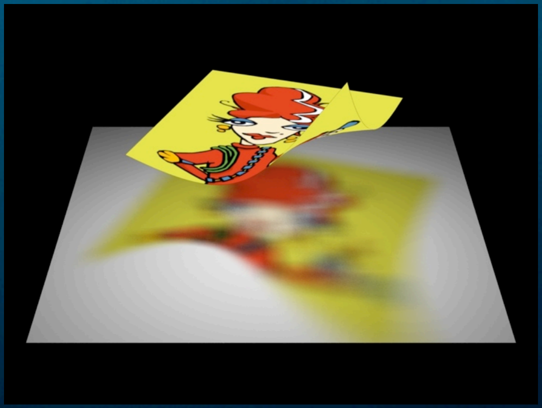
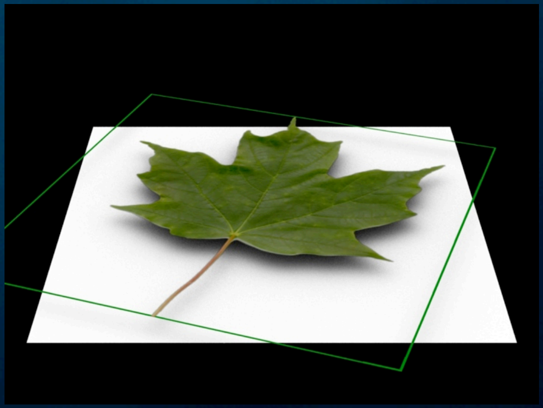
PIXAR

Other applications of caching

- We developed method to accelerate GI, but can be applied to other parts of shading pipeline
- Whereever there are view-independent results that can be reused

Other applications: opacity

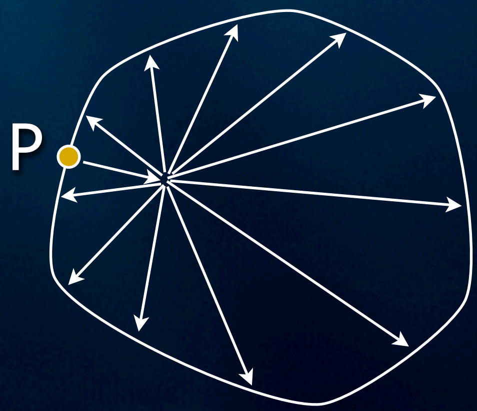
- Computed by shaders: shadows, ambient occlusion, volume extinction
- Speedups: ~2-4x

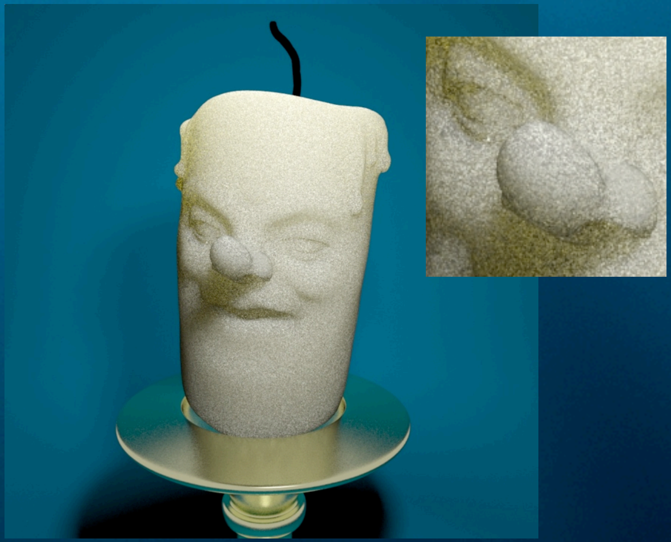
PIXAR

Other applications: subsurf scatter

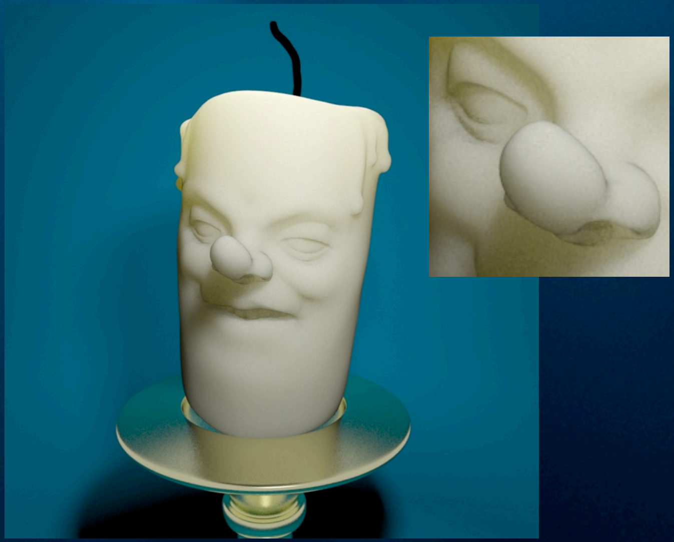
- Ray-traced subsurface scattering
- Trace rays to sample surf geom + illum
- Cache irradiance -- incident illum on surface



Other applications: subsurf scatter



preview: 19 sec
3.4x speedup



final: 3.2 min
17x speedup

Other applications: subsurf scatter

- Used in production, for example:
  - – Snow White and the Huntsman (skin, wings)
  - – The Dark Knight Rises (bats)

More info

- graphics.pixar.com/library/RadiosityCaching :
- Pixar tech memo #12-06
- Videos of interactive GI and subsurface scattering

Conclusion

- Reduces bottleneck: shader evals (and shadow rays)
- Suitable for interactive & final rendering
- Speedups > 30 in production scenes
- In PRMan
- Used in production at several studios (including Pixar)

Acknowledgements

- RenderMan/Seattle team
  - Katrin Bratland, Julian Fong, Stephen Friedman, Chris Harvey, Ian Hsieh, Andrew Kensler, Charlie Kilpatrick, David Laur, Brian Saunders, Brian Savery, Chris Scoville, Brian Smits, Adam Wood-Gaines, Wayne Wooten
- Pixar Emeryville
  - Bill Reeves, Jean-Claude Kalache, Christophe Hery, Ryusuke Villemin, Sanjay Bakshi, Guido Quaroni, Tony DeRose, ...

PIXAR

Thanks!

Questions?