Point-Based Global Illumination for Movie Production

Per Christensen

Pixar Animation Studios

SIGGRAPH 2010 Course

PIXAR

Overview

- Point-based global illumination
  - generating direct illumination point cloud
  - rendering GI using point cloud
- Examples of use in movies
- Variations and extensions
- What's next?

PIXAR

Related work

- Method is inspired by Bunnell's point-based GPU method
- Related to clustering radiosity and point-based subsurface scattering

PIXAR

Point-based global illumination

- Fast, low memory, no noise
- Handles complex geometry (including dense polygon meshes, hair, leaves, displacement), many light sources, complex surface shaders, ...
- Movie-production friendly
- Part of Pixar's RenderMan renderer

PIXAR

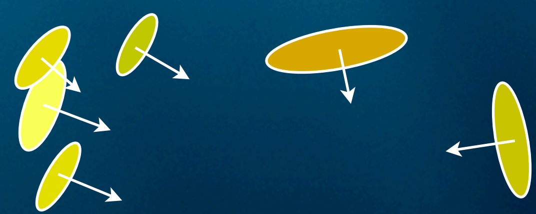
Point-based global illumination

- Three steps:
- Generate point cloud of directly illuminated surface colors (radiosity)
- Organize points into octree; larger points and spherical harmonics
- Render: compute diffuse/glossy global illumination at each shading point

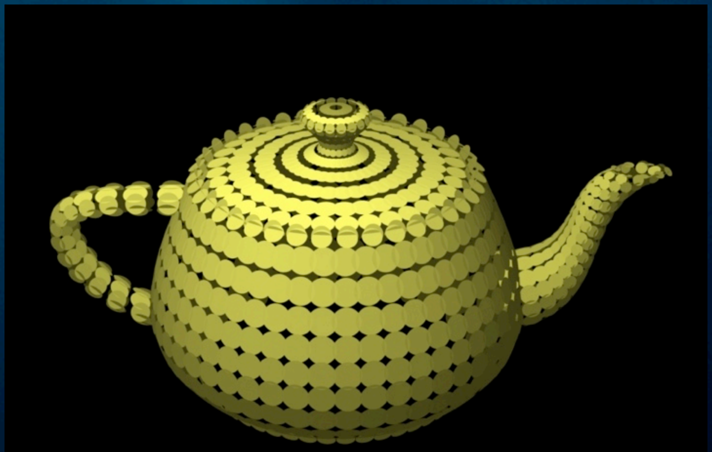
PIXAR

A point cloud

- Each point: position, normal, radius, color = a colored disk



point cloud



point cloud

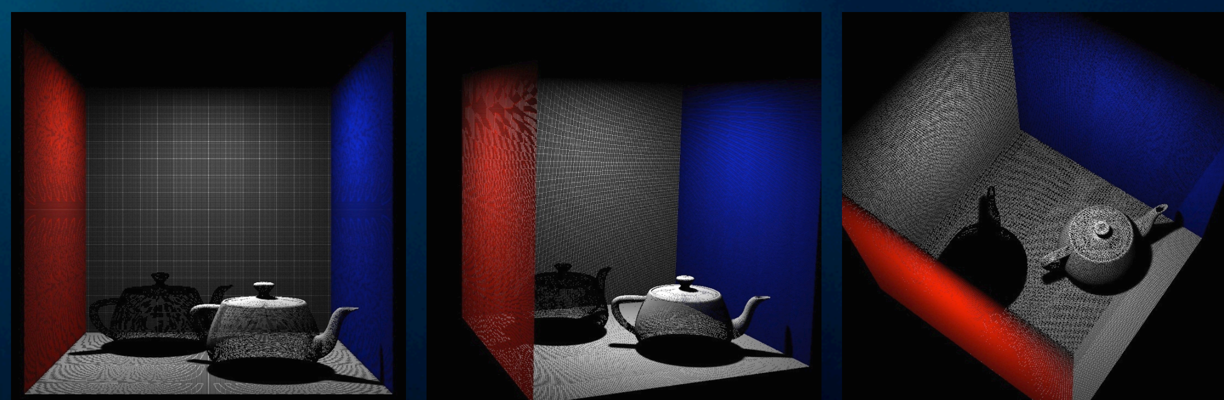
- Terminology: “point” or “disk” or “surfel”?

PIXAR

Generate point cloud

- Render direct illumination image
- Generate point cloud file at same time

rendered image

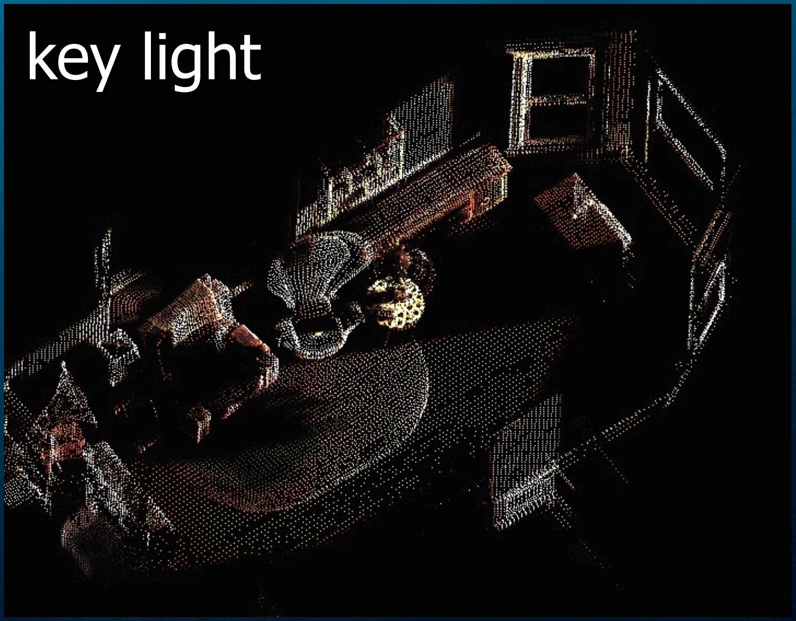


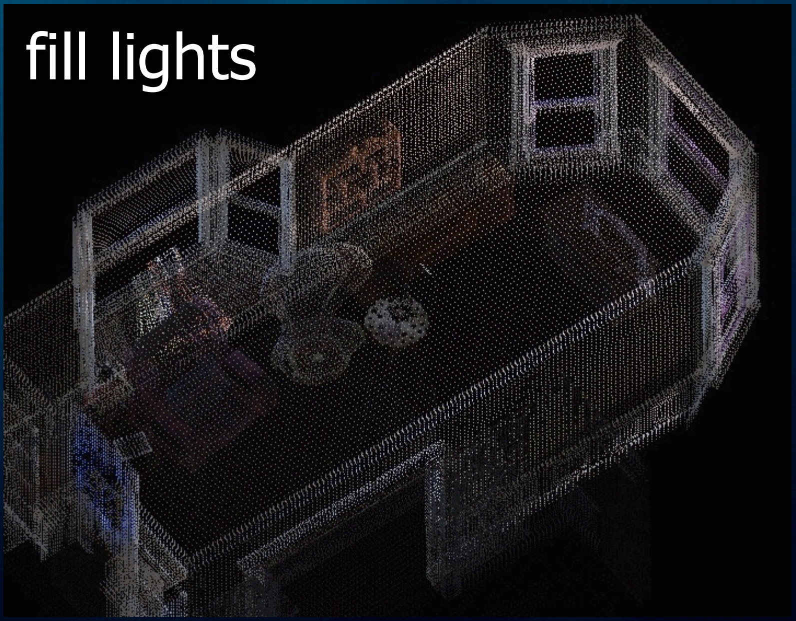
point cloud, 560K points (various views)

PIXAR

Generate point cloud

- Point cloud files from “Up”





PIXAR

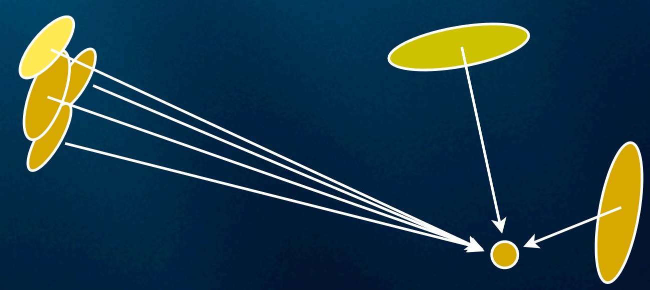
Organize points into octree

- Organize points into octree
- Each cluster of points is represented by a larger point or a spherical harmonic representation of directional light distribution

PIXAR

Compute global illum at a point

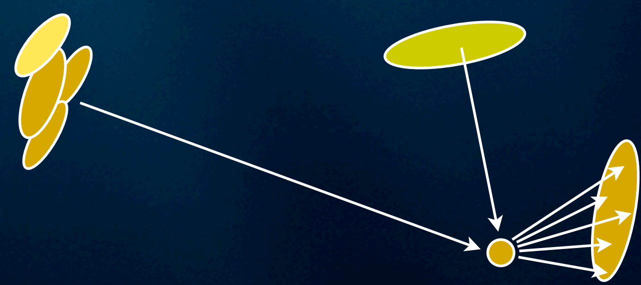
- Basic idea: add up color from all other points!



PIXAR

Compute global illum at a point

- For efficiency: use cluster of points for distant points
- For higher accuracy: ray trace close points



PIXAR

Compute global illum at a point

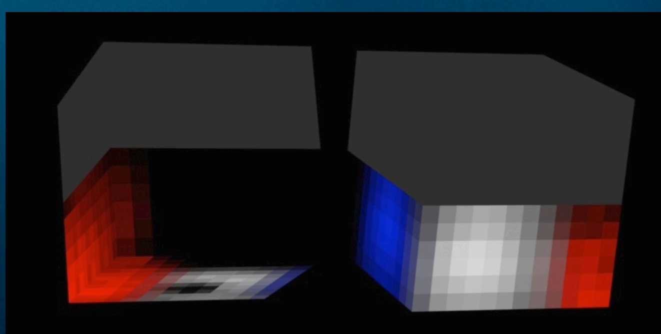
- Problem: if all points are added up, even points "hidden" behind other points will contribute

PIXAR

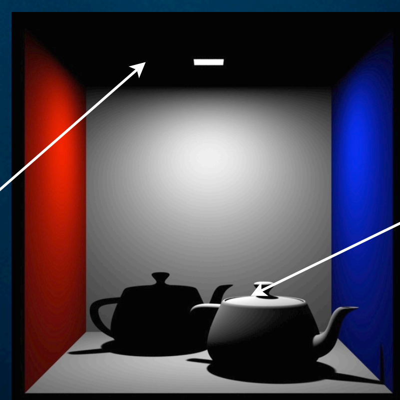
Compute global illum at a point

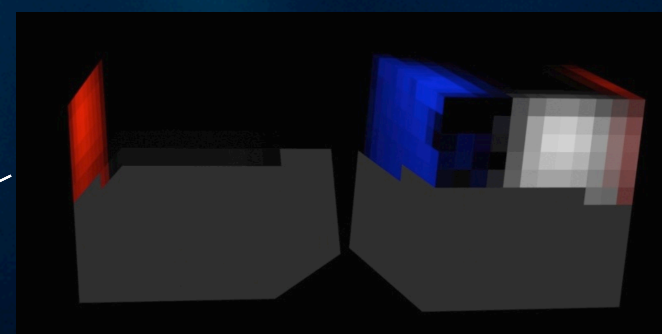
- Solution: rasterize colors contributing to a point -- world “as seen” by that point

- Raster cube examples:



point on ceiling





point on teapot lid

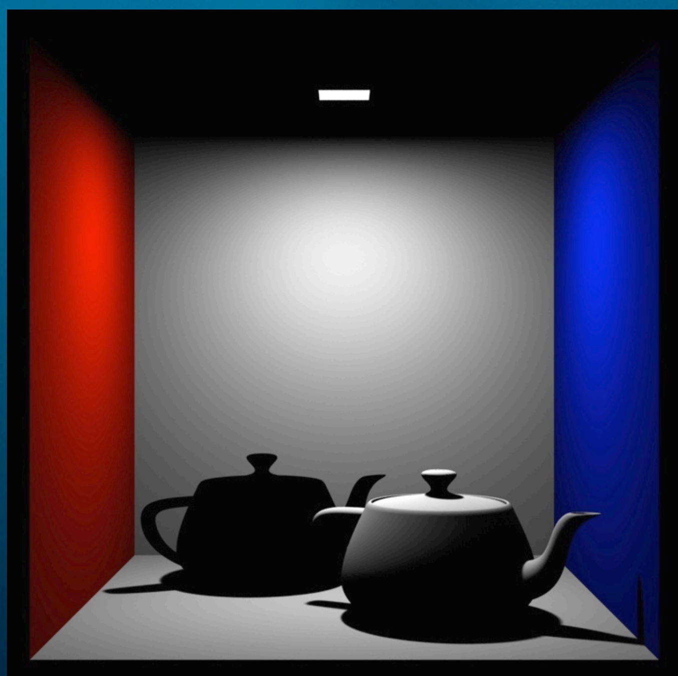
PIXAR

Compute global illum at a point

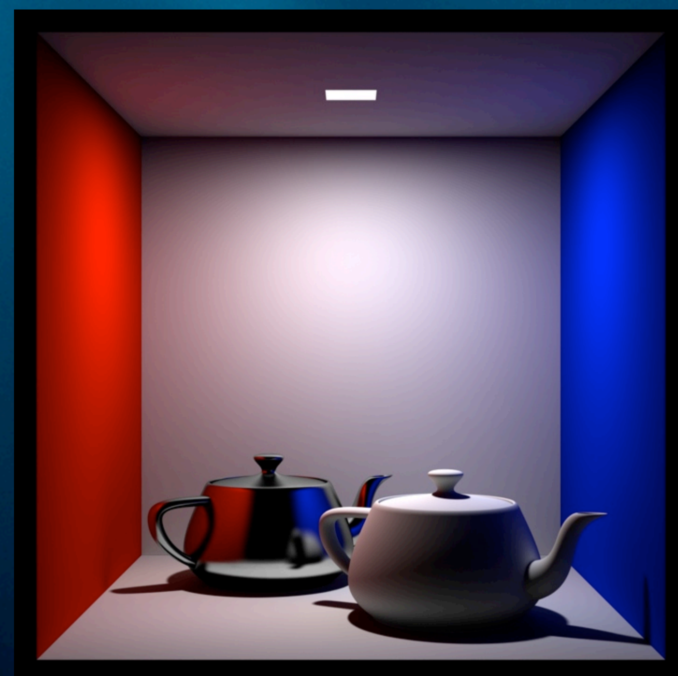
- Multiply all raster pixel colors by reflectance function (BRDF); add
- Result is diffuse / glossy reflection at point

PIXAR

Global illumination result



direct illum (9 sec)



direct illum + diffuse GI + glossy GI (21 sec)

PIXAR

Use in movies

- Implemented in Pixar's RenderMan
- Integrated into lighting pipeline at ILM, Pixar, Disney, DNeg, MPC, ...

PIXAR

Use in movies

- Pirates of the Caribbean 2 & 3, Eragon, Surf's Up, Spiderman 3, Harry Potter 5 & 6, Chronicles of Narnia, Fred Claus, Beowulf, Spiderwick Chronicles, Ironman 1 & 2, Indiana Jones, 10,000 BC, Batman: Dark Knight, Quantum of Solace, Cloverfield, Doomsday, Hellboy 2, Inkheart, Wall-E, Bolt, Star Trek, Terminator 4, The Boat that Rocked, Fast & Furious 4, Angels and Demons, Night at the Museum, Up, Transformers 2, 2012, Sherlock Holmes, Percy Jackson, The Green Zone, Prince of Persia, Toy Story 3, ...

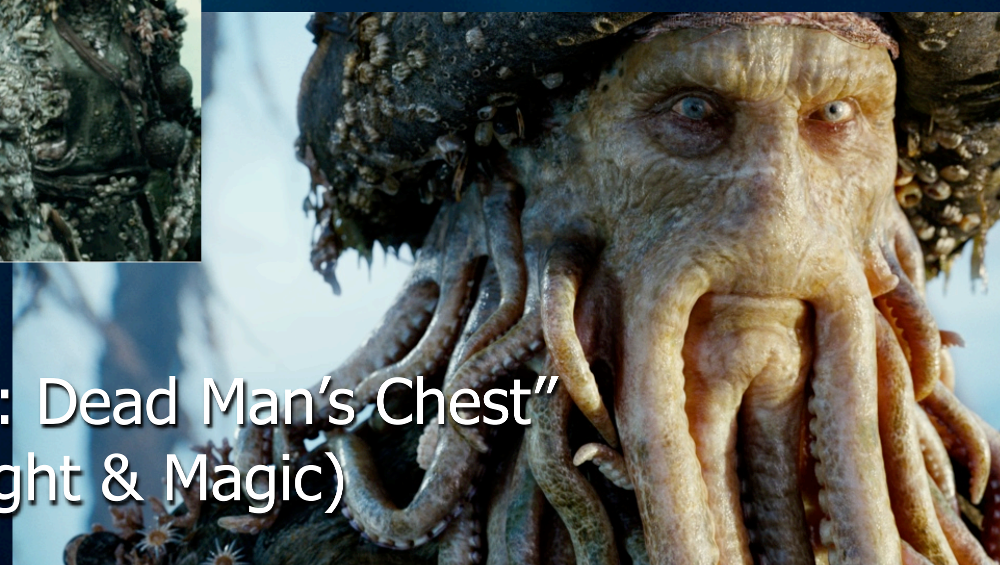
PIXAR

Sony: "Surf's Up" ambient occlusion

"Surf's Up" test (Courtesy of Rene Limberger, Sony)

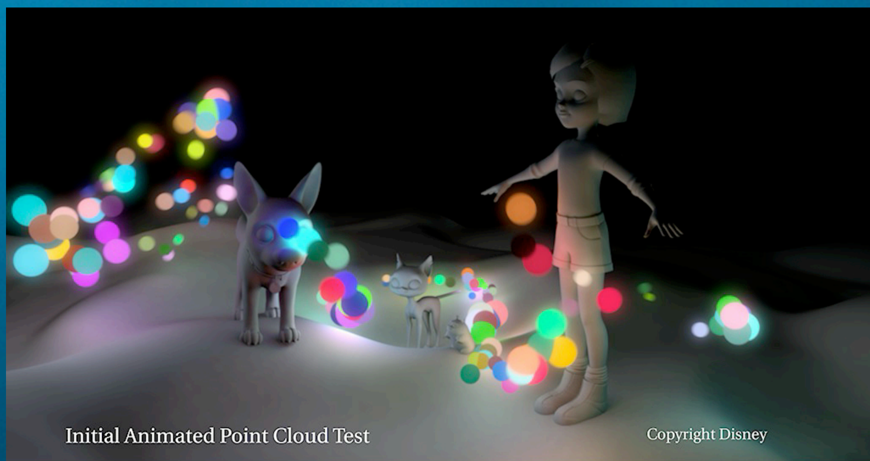
PIXAR

ILM: Davy Jones



"Pirates of the Caribbean: Dead Man's Chest"
(Courtesy of Industrial Light & Magic)

Disney: special effects on "Bolt"



(Courtesy of Dale Mayeda, Disney)

PIXAR

“Up” example without global illum

PIXAR

"Up" example with global illum

PIXAR

"Up" example without global illum

PIXAR

"Up" example with global illum

PIXAR

“Toy Story 3” examples

PIXAR

“Toy Story 3” examples

PIXAR

"Toy Story 3" examples

PIXAR

"Toy Story 3" examples

PIXAR

"Toy Story 3" examples

PIXAR

"Toy Story 3" examples

PIXAR

Variations and extensions

- Area light sources
- Environment illumination
- Multiple light bounces
- Final gather for photon maps
- Ambient/directional/reflection occlusion
- Volumes

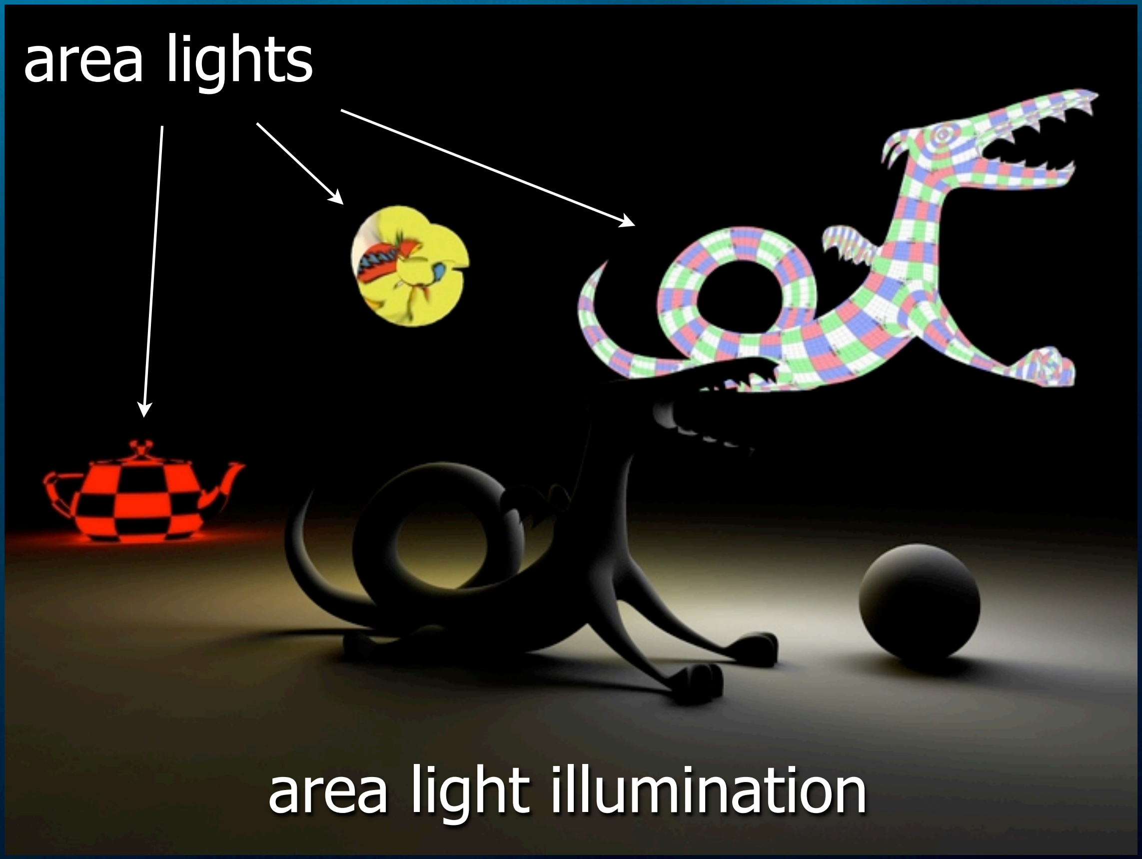
PIXAR

Area light sources + soft shadows

- Treat area light sources the same as surfaces: generate point cloud with color data
- Light sources can have arbitrary shape and colors
- Also write (black) points for shadow-casting objects

PIXAR

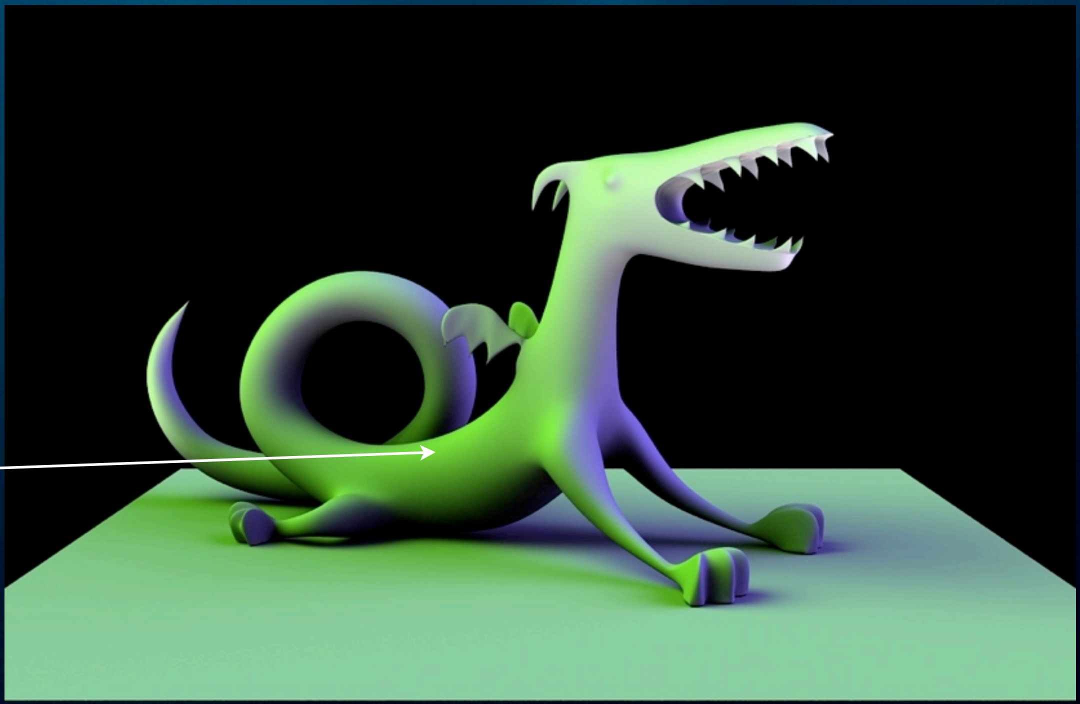
Area light sources + soft shadows



PIXAR

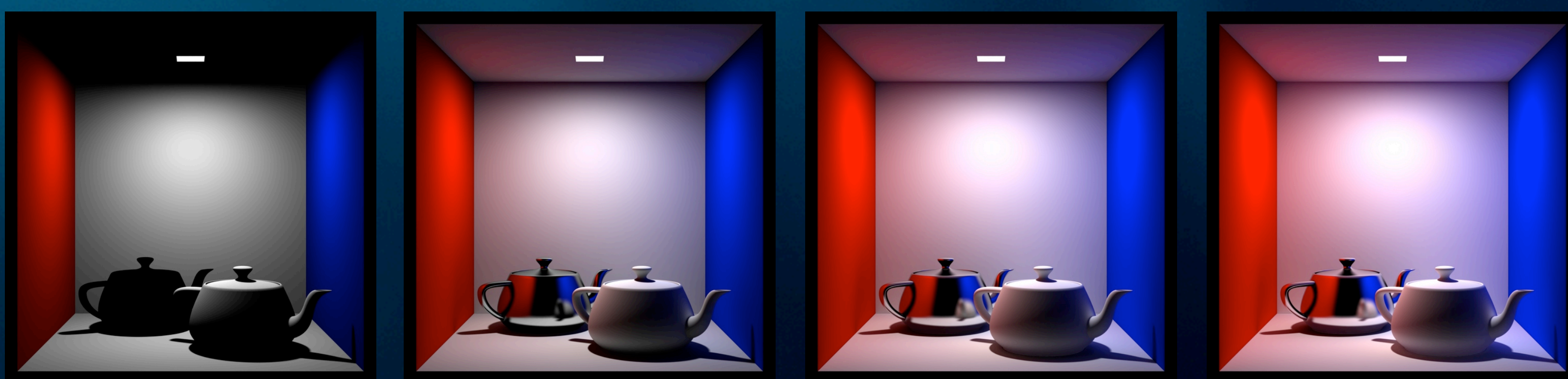
Environment illumination -- IBL

- Use environment color for raster pixels not covered by points



Multiple light bounces

- Run the algorithm n times
- (For efficiency: first n-1 times can be computed at fewer points)



$n = 0$ $n = 1$ $n = 2$ $n = 3$

PIXAR

Final gather for photon mapping

- Final gather step is usually done with ray tracing; slowest part of photon mapping
- Use point-based method instead

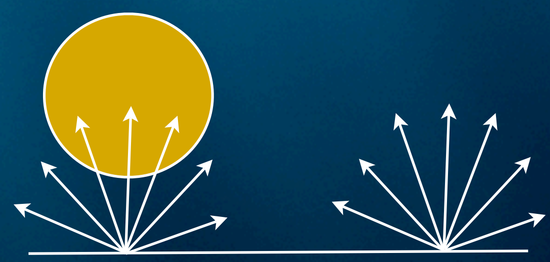
PIXAR

Final gather for photon mapping

PIXAR

Special case: Ambient occlusion

- Fraction of hemisphere above a point that's covered



- Similar to shadows on overcast day
- Values between 0 and 1

PIXAR

Ambient occlusion

- Generate point cloud with only position, normal, radius (no colors)

PIXAR

Ambient occlusion

PIXAR

Ambient occlusion (and reflections)

PIXAR

NEW: Image-based relighting

- In addition to ambient occlusion, also compute directional visibility: spherical harmonic coeffs. at each point
- Compute SH coeffs for environment map
- (Re-)rendering is just multiplying SH coefficients -- 9 or 25 mults/point. Fast!

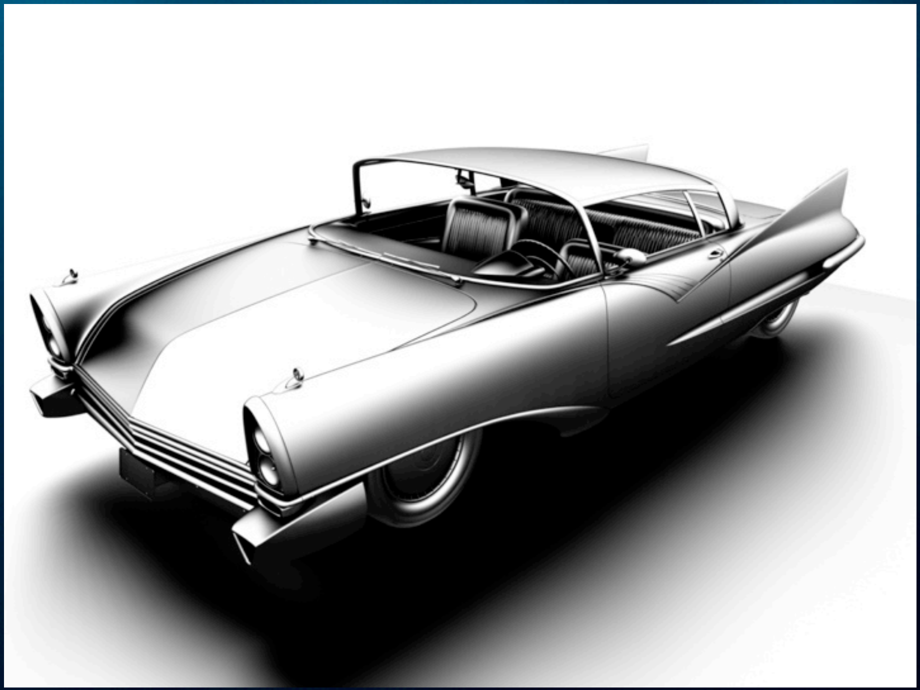
PIXAR

NEW: Image-based relighting

IXAR

Special case: reflection occlusion

- As ambient occlusion, but narrow cone of directions (around reflection direction)



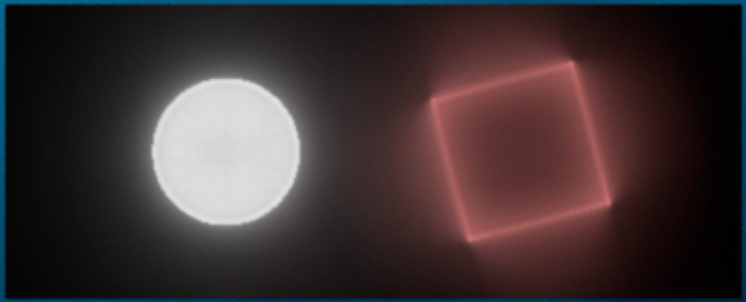
PIXAR

Global illumination in volumes

- Points don’t have normals: spheres, not disks
- Illumination from all directions: entire raster cube
- surface ↔ volume
- volume ↔ volume

PIXAR

Global illumination in volumes



surface to volume

volume to volume

PIXAR

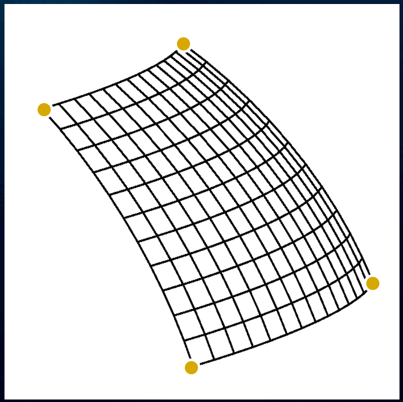
Optimization: interpolation

- If the color bleeding varies only a little in an area (<2%), we simply interpolate it
- Technique known from ray tracing ("irradiance cache")

PIXAR

Optimization: interpolation

- Compute color bleeding at the 4 corners of surface patch
- Is the difference between 4 values small?
  - yes: interpolate on patch
  - no: split patch in 2; recurse



surface patch

PIXAR

Parallel computation

- Global illumination at each point is independent
- Ideal for parallel execution
- Observed speedups:
  - 4 cores: ~3.6
  - 8 cores: ~6.6

PIXAR

More information

- M. Bunnell, “Dynamic ambient occlusion and indirect lighting”, GPU Gems 2
- P. Christensen, “Point-based approximate color bleeding”, Pixar tech memo #08-01
- T. Ritschel et al, “Micro-rendering for scalable, parallel final gathering”, SIGGRAPH Asia 2009

PIXAR

Summary

- Point-based diffuse and glossy global illumination is fast and can handle complex production scenes
- Also works for area lights, env. map illumination, multiple bounces, ambient occlusion, reflection occlusion, volumes
- In Pixar's RenderMan
- Widely used in production

PIXAR

What's next?

- "Up" and "Toy Story 3": 1-bounce PBGI was used in addition to all the traditional lights
- Next:
  - reduce number of traditional lights?
  - multiple bounces?

PIXAR

What's next?

- Implementation improvements:
  - improved accuracy in rasterization?
  - baking micropolygon grids?
  - GPU implementation?

PIXAR

Acknowledgments

- RenderMan team: Dana Batali, ...
- Mike Bunnell, Rene Limberger, Christophe Hery
- Pixar: Max P, P Sumo, JC, Stefan, Guido, ...
- Dale Mayeda (Disney), Philippe Leprince (DNeg), Anders Langlands (MPC), ...

Thanks!

PIXAR

Questions?

PIXAR