

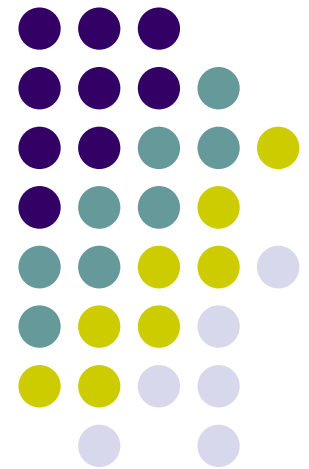
Computer Graphics

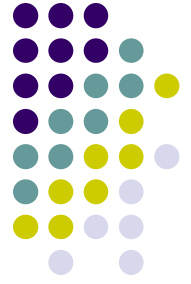
CS 543 – Lecture 12 (Part 2)

Advances in Graphics

Prof Emmanuel Agu

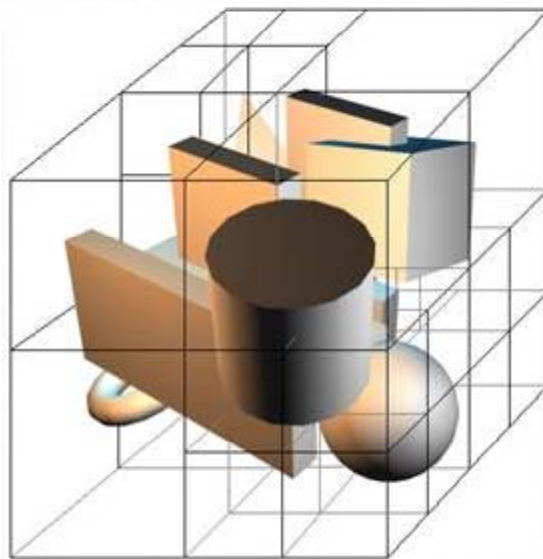
*Computer Science Dept.
Worcester Polytechnic Institute (WPI)*



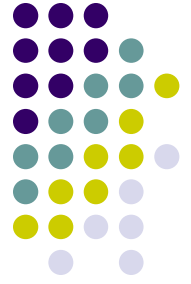


Accelerating Ray Tracing

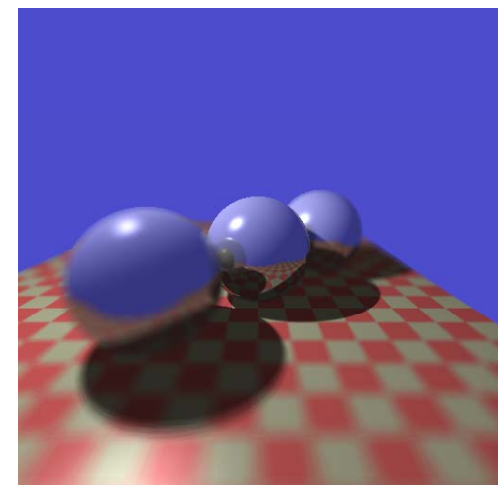
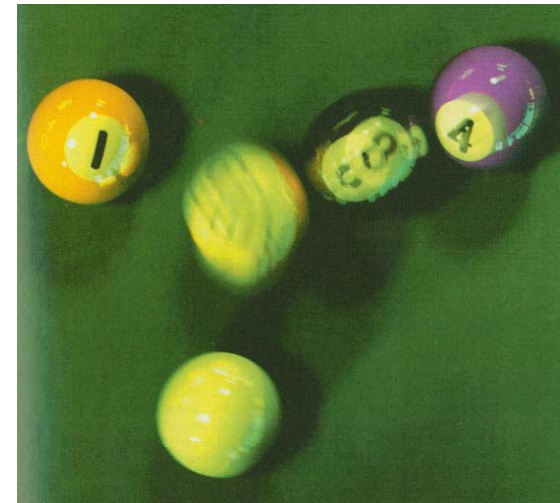
- To accelerate ray tracing, place grid over scene
- Test cells recursively
- Acceleration structures: BSP trees, kd trees, etc

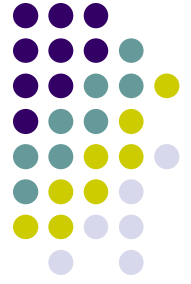


Making Ray Tracing Look Real



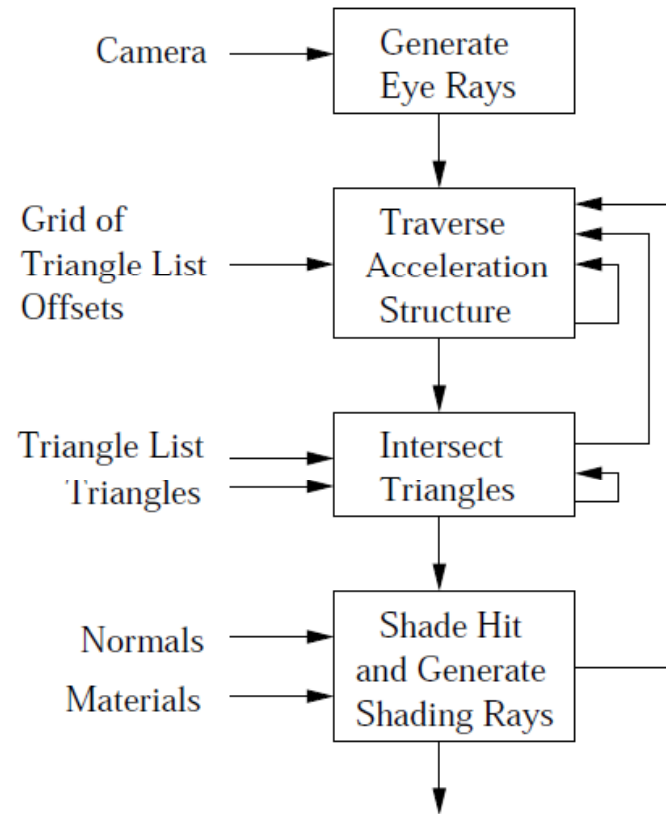
- **Antialiasing**
 - Cast multiple rays from eye through same point in each pixel
- **Motion blur**
 - Each of these rays intersects the scene at a different time
 - Reconstruction filter controls shutter speed, length
- **Depth of Field**
 - Simulate camera better
 - f-stop
 - focus
- **Other effects** (soft shadow, glossy, etc)

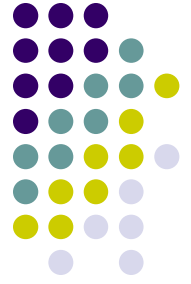




Real Time Ray Tracing

- Multi-pass rendering: Ray tracer using 4 shaders



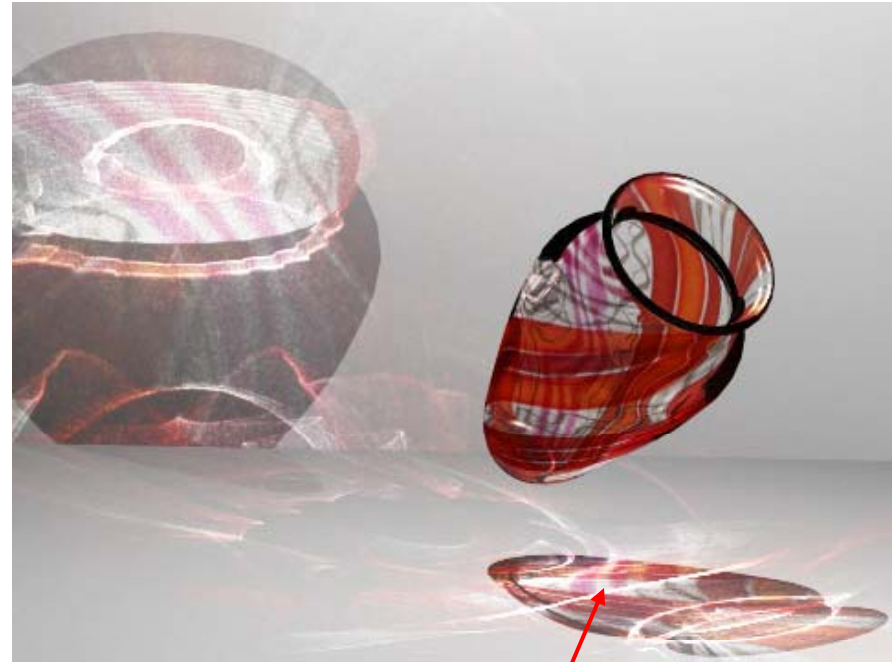
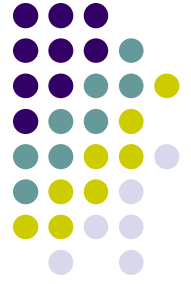


Real Time Ray Tracing

- Nvidia Optix ray tracer
- Needs high end Nvidia graphics card
- SDK is available on their website
- <http://developer.nvidia.com/object/optix-home.html>

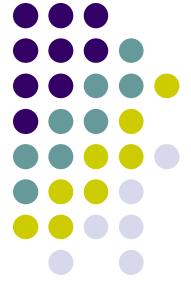


Photon mapping examples



Caustics

Images: courtesy of Stanford rendering contest

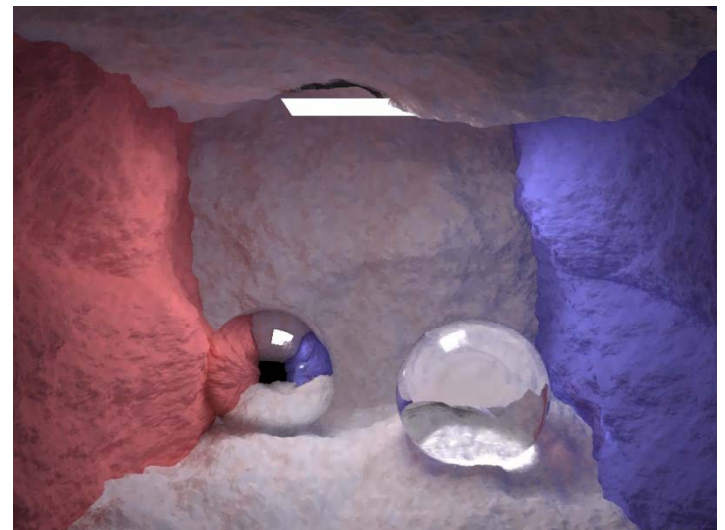


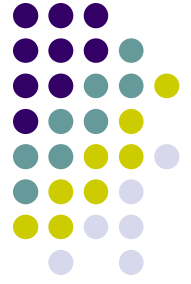
Photon Mapping

- Simulates the transport of individual photons (Jensen '95-'96)

Two pass algorithm

- Pass 1 - **Photon tracing**
 - Emit photons from lights
 - Trace photons through scene.
 - Store photons in kd-tree (photon maps)
- Pass 2 - **Rendering**
 - Render scene using information in the photon maps to estimate:
 - Reflected radiance at surfaces
 - Scattered radiance from volumes and translucent materials.
- Good for effects ray tracing can't:
 - Caustics
 - Light through volumes (smoke, water, marble, clouds)





Photon Tracing

Photon scattering

- Emitted photons are probabilistically scattered through the scene and are eventually absorbed.
- Photon hits surface: can be reflected, refracted, or absorbed
- Photon hits volume: can be scattered or absorbed.

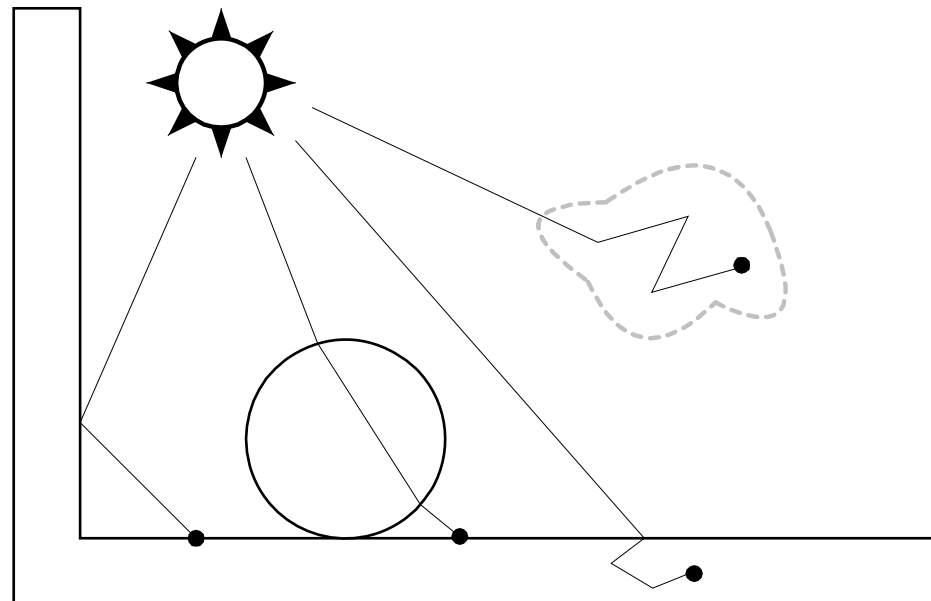
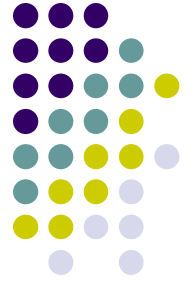
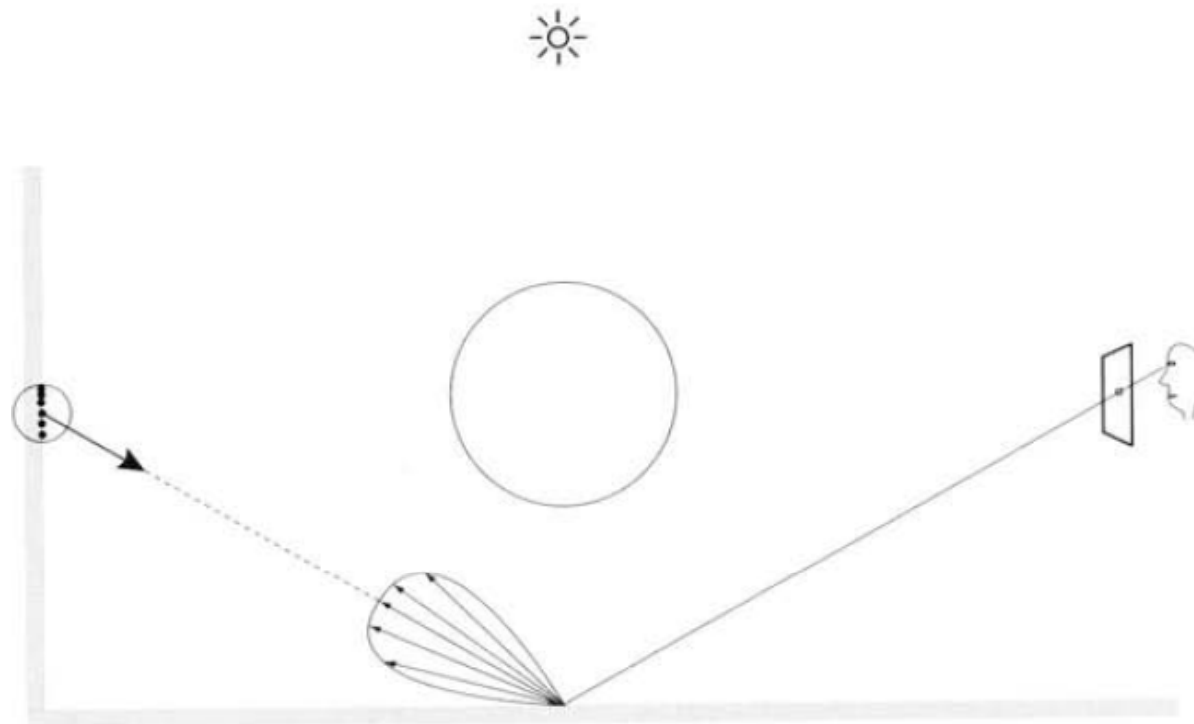


Illustration is based on figures from Jensen[1].

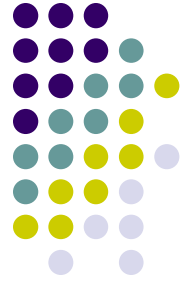
Photon mapping: Pass 2 - Rendering



- Indirect diffuse lighting: Use ray tracing
- Indirect light, volumes, caustics: estimate illumination using photon map

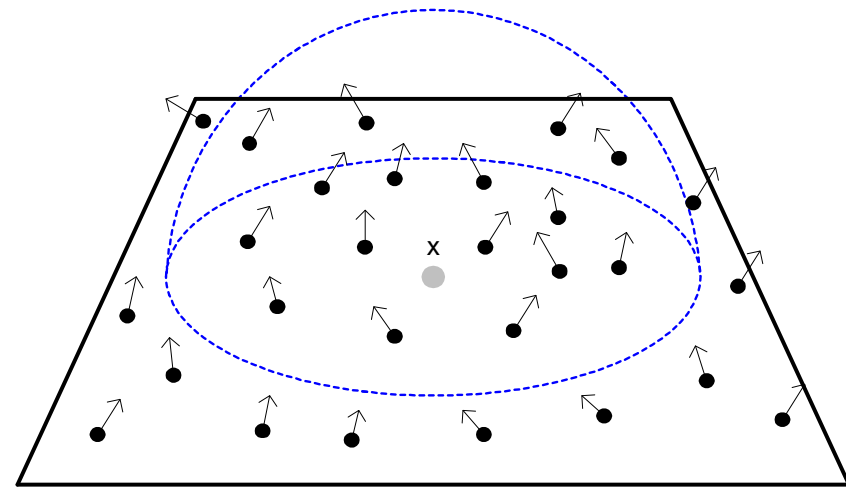
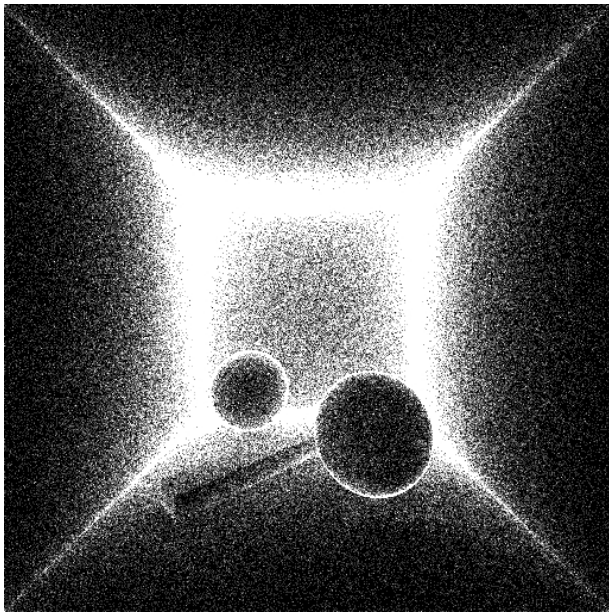


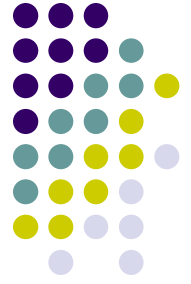
Photon Tracing



Pass 2 - Rendering

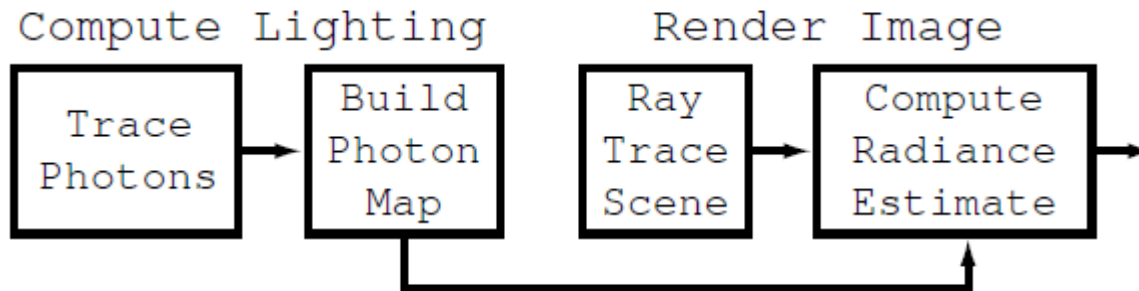
- Imagine ray tracing a hitpoint x
- Information from photon maps used to estimate radiance from x
- Radius of circle required to encountering N photons gives radiance estimate at x

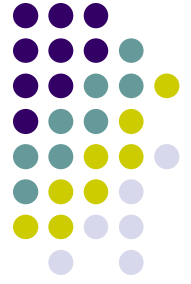




Real Time Photon mapping

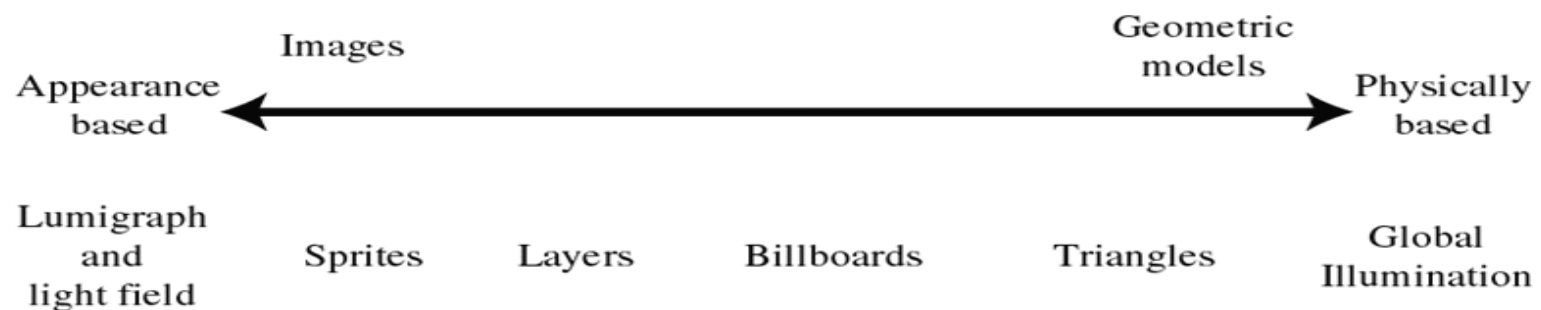
- Similar idea to real-time ray tracing.
- Photon mapping as multi-pass shading

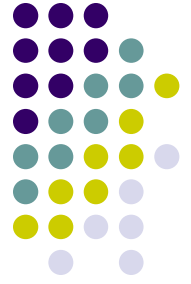




Real-Time Rendering Techniques

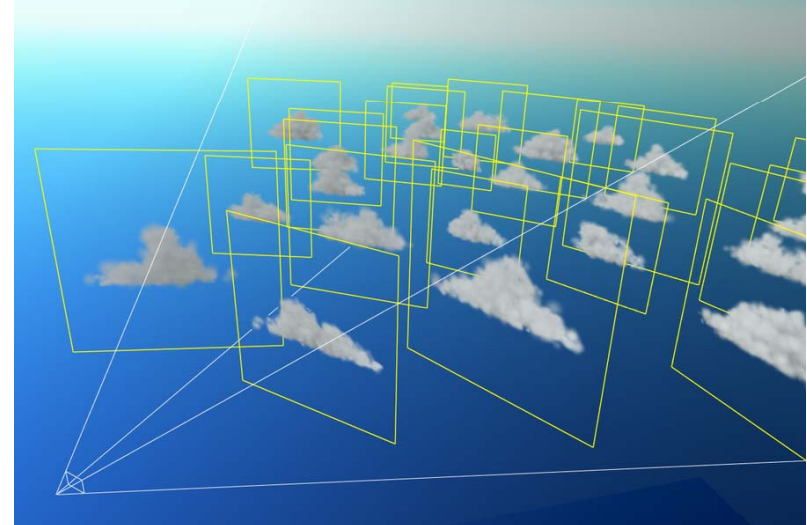
- Applications: game engines, virtual reality, simulators, etc
- Algorithms must run at min 30 FPS
- Polygonal techniques: OpenGL, DirectX
- Shaders: Pixel/vertex shading
- Level of detail management (simplification, tessellation)
- Texturing to improve RT performance
- Point-based rendering
- BRDF factorization, SH lighting
- Image-based rendering: Spectrum of IBR techniques



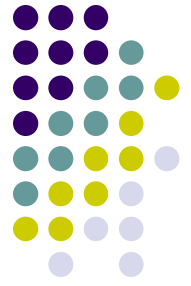


Billboards

- IBR: pre-render geometry onto images/textures
- Rendering at runtime involves simple lookups, fast
- Similar technique used for crowds in NFL madden football

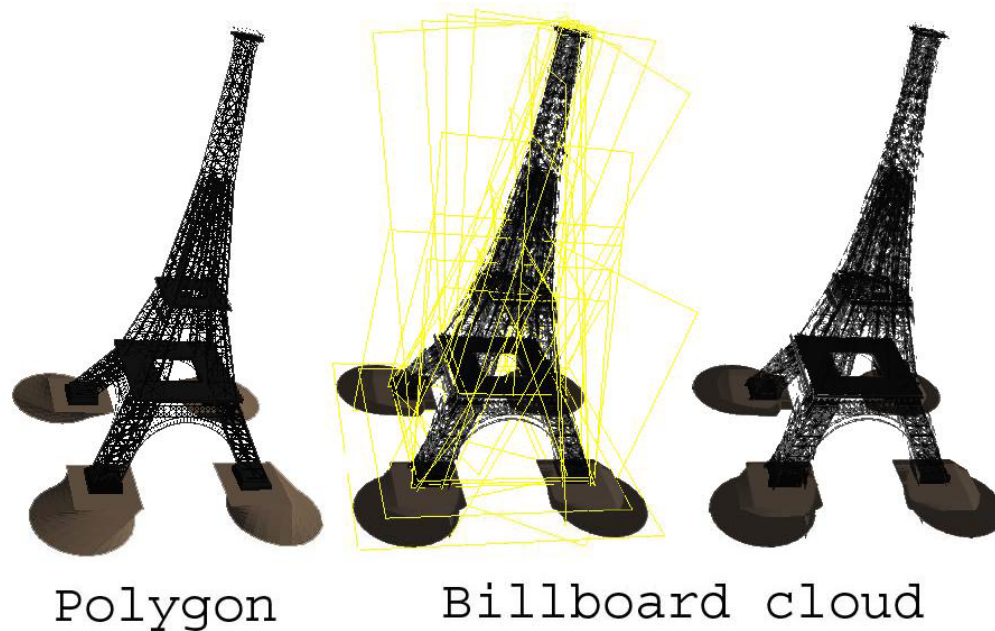


Real time cloud rendering, Mark J. Harris



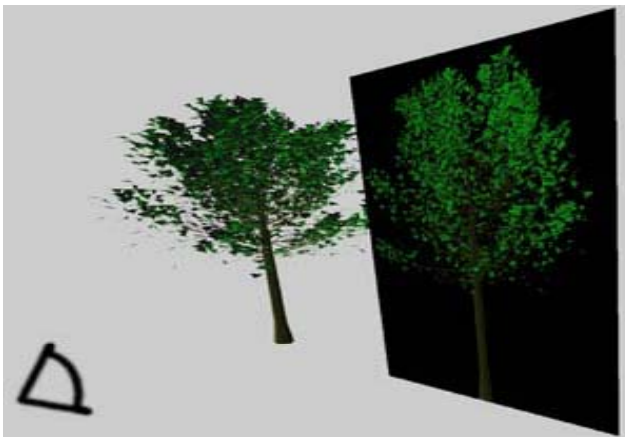
Billboard Clouds

- **Billboard Clouds**, Decoret, Durand *et al* [SIGGRAPH'03]
- Render complex mesh onto cloud of billboards
- Billboard inclined at different viewpoints



Imposters

- Similar to billboards



Impostors Made Easy – William Damon, Intel



No Impostors

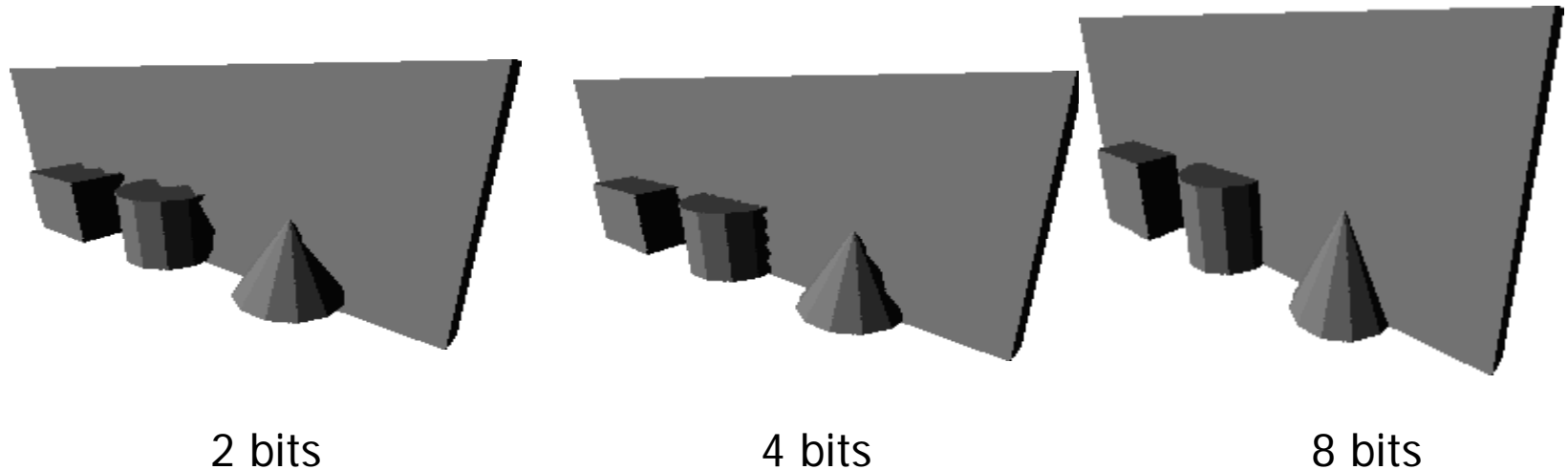


With Impostors

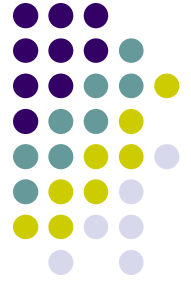


Depth Sprite aka Nailboard

- Give depth to image !
- $RGB\Delta$ - Δ (transparency) is depth parameter
- Set Δ based on depth of actual geometry
- Accuracy varies with no. of bits to represent Δ



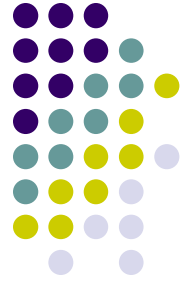
<http://zeus.gup.uni-linz.ac.at/~gs/research/nailbord/>



IBR: Pros and Cons

- Pros
 - Simplifies computation of complex scenes
 - Rendering cost independent of scene complexity
- Cons
 - Static scene geometry
 - Fixed lighting
 - Fixed look-from or look-at point

Recent Trends in Games

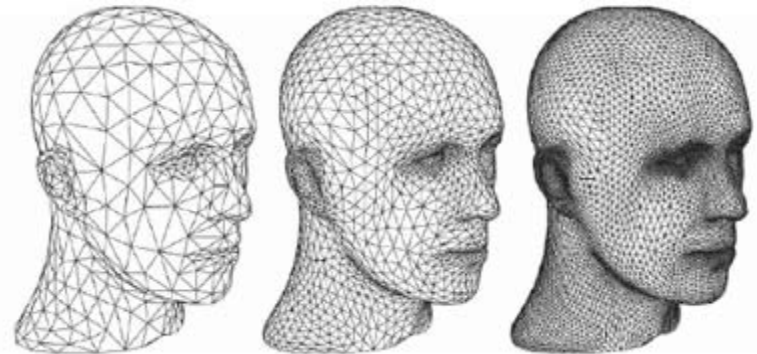


1. Real Time LoD Management
2. Capture rendering data
3. Pre-computation to speed up run-time
4. Screen Space GI techniques
5. Real Time Global Illumination
6. Hardware-accelerated physics engines



Trend 1: Real-Time LoD Management

- Geometry shader unit, can generate new vertices, primitives from original set
- Tessellation and simplification algorithms on GPU
- Real-time change LoD in game



[Zorin and Schröder, 2000]



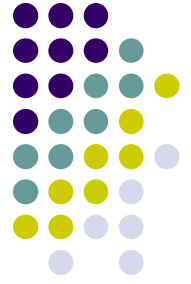
Trend 2: Capture Rendering Data

- Old way: use equations to model:
 - Object geometry, lighting (Phong), animation, etc
- New way: capture parameters from real world
- Example: motion in most sports games (e.g. NBA 2K live) is captured.
 - How? Put sensors on actors
 - Actors play game
 - Capture their motion into database
 - Player motion plays back database entries



Courtesy: Madden NFL game

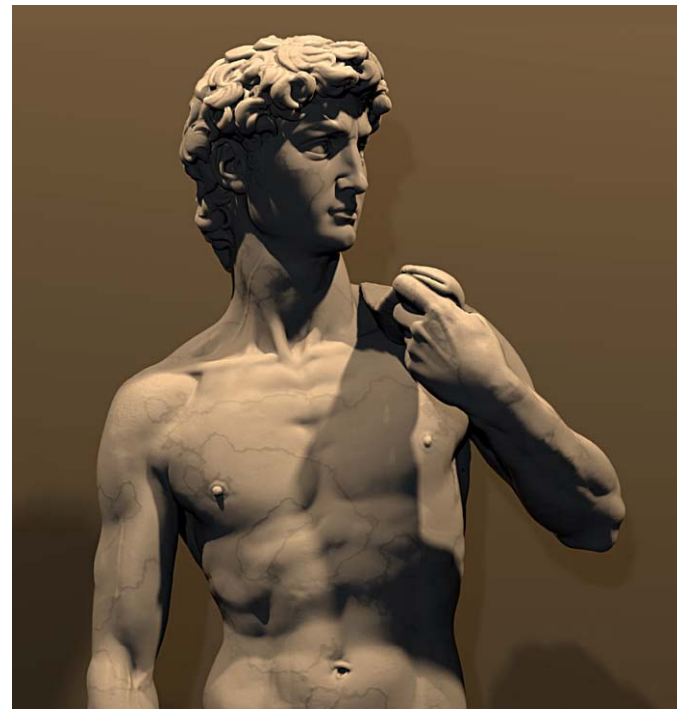
Geometry Capture: 3D Scanning



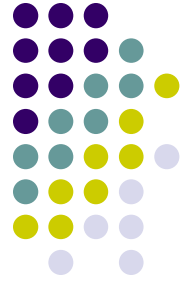
- **Capturing geometry trend:** Precise 3D scanning (Stanford, IBM, etc) produce very large polygonal models

Model: David

Largest dataset Size: 2 billion polygons, 7000 color images!!



Courtesy: Stanford Michael Angelo 3D scanning project



How is capture done?

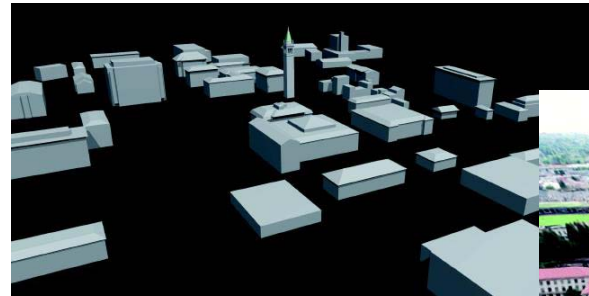
- **Capture:**
 - Digitize real object geometry and materials
 - Use cameras, computer vision techniques to capture rendering data
 - Put data in database, many people can re-use
- **Question:** What is computer vision?



Exactly What Can We Capture?



1. Appearance (volume, scattering, transparency, translucency, etc)



2. Geometry

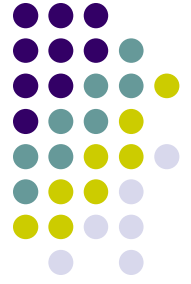
3. Reflectance & Illumination



4. Motion

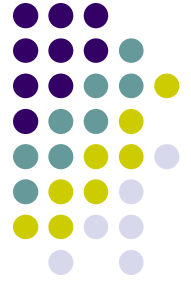


Light Probes: Capturing light



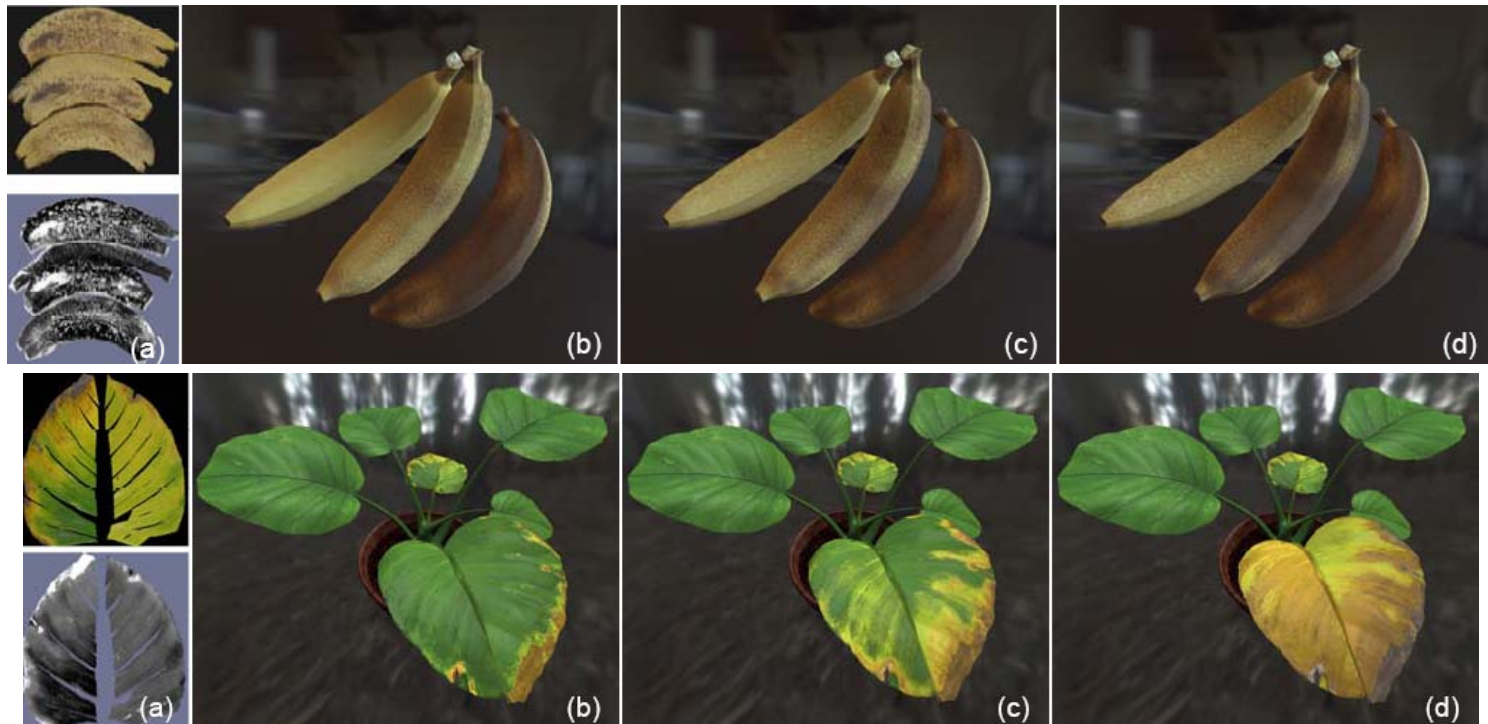
Amazing graphics, High Dynamic Range?



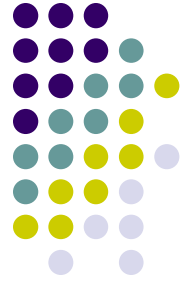


Capture Material Reflectance (BRDF)

- **BRDF:** How different materials reflect light
- **Examples:** cloth, wood, velvet, etc
- Time varying?: how reflectance changes over time
- TV examples: weathering, ripening fruits, rust, etc

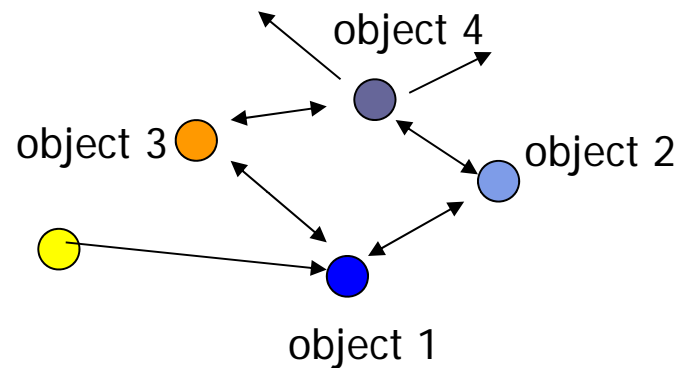


Why effort to capture?



- **Big question:** If we can capture real world parameters, is this really **computer graphics**?

Trend 3: Pre-computation to speed up run-time



- Pre-compute **lighting**
 - Lights objects mostly static
 - Use GPU to pre-compute approximate lighting solutions
 - Speeds up run-time
- Pre-computed **Occlusion**
- Pre-computed **Radiance Transfer** (reflections)
 - Use spherical harmonics

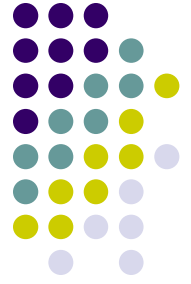
Pre-computed Global Illumination



HALO 3

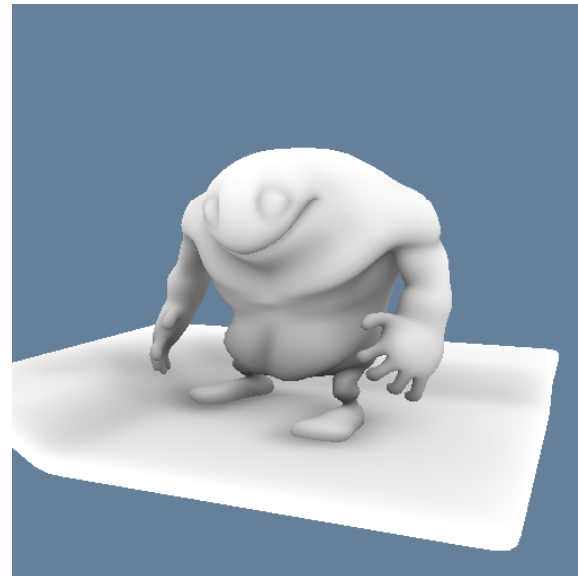
BUNGIE

SIGGRAPH2009



Pre-Compute Occlusion

- Ambient occlusion
 - Each rendered point receives hemisphere of light
 - Estimate fraction of hemisphere above point that is blocked
 - Render ambient term as fraction of occlusion

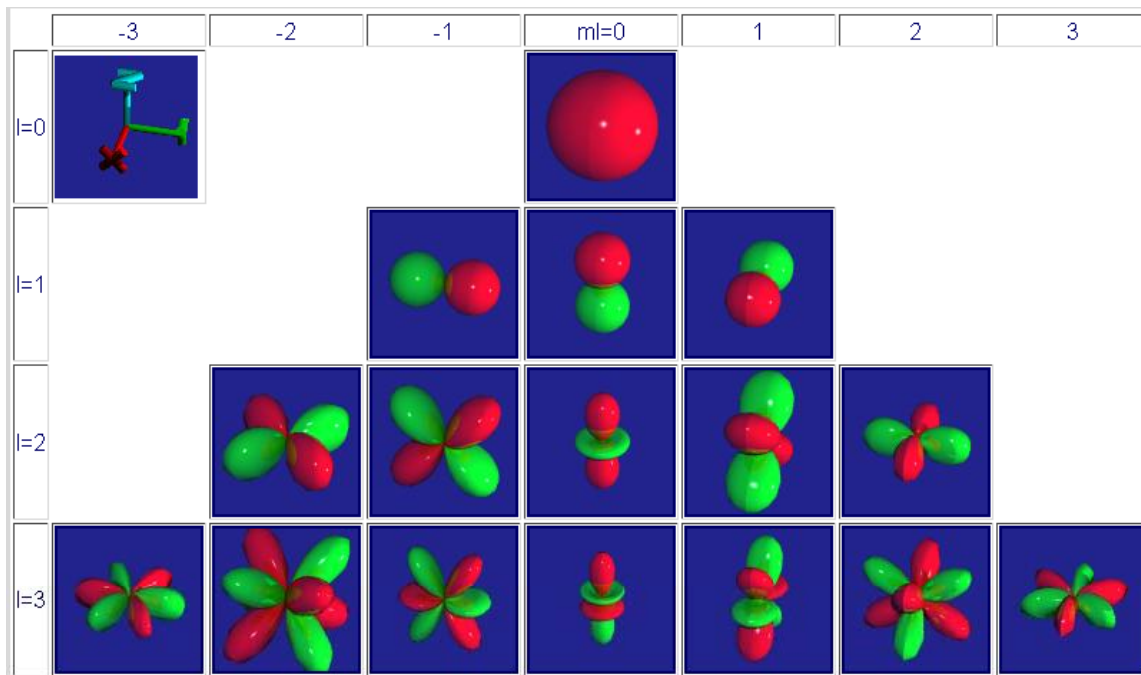


Courtesy Nvidia
SDK 10



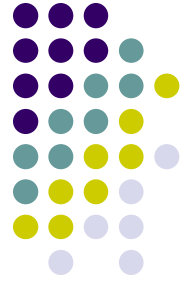
Precomputed Radiance Transfer

- Factorize and precompute light and material as **Spherical Harmonics**
- Run-time: Light reflection is dot product at run time (Fast)



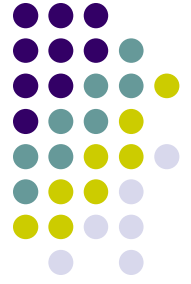
Sponza Atrium: Courtesy Marko Dabrovic

Trend 4: Real time Global Illumination



- What's the difference?
 - Pre-compute means lookup at run-time
 - Approximate representations (e.g Spherical Harmonics)
 - Fast, but not always accurate
- Real Time Global Illumination: state-of-the art
 - Calculate complex GI equations at run-time
 - Use GPU, hardware

Real Time Global Illumination



- Ray tracing enables global illumination
- Instead of billboards, imposters, images use physically-based appearance models
- Very cool effects:
 - Shadows
 - Ambient Occlusion
 - Reflections
 - Transmittance
 - Refractions
 - Caustics
 - Global subsurface scattering
 - What does it look like?

Real-time Lighting in Games



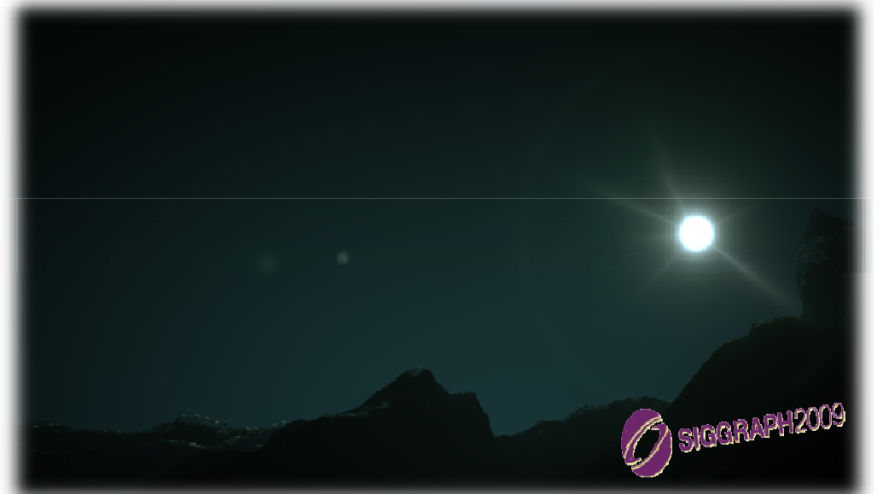
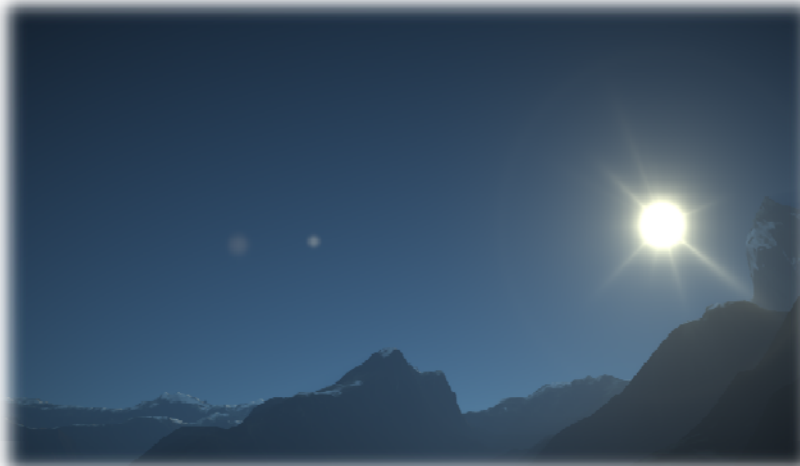
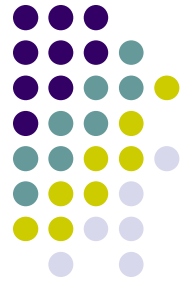
Sky and Atmosphere: HALO 3 Previous Model

- Used in Halo 3
- [PSS99][PreethamHoffman03]
- Offline pre-computed sky texture
- Real-time scattering
- Single scattering only
- Viewable from ground only

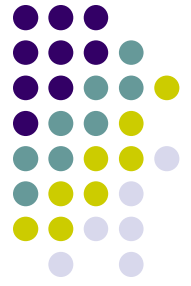
Current Model

- [BrunetonNeyret2008]
- Single and multiple scattering
- Pre-computation on the GPU
- Viewable from space
- Light shafts

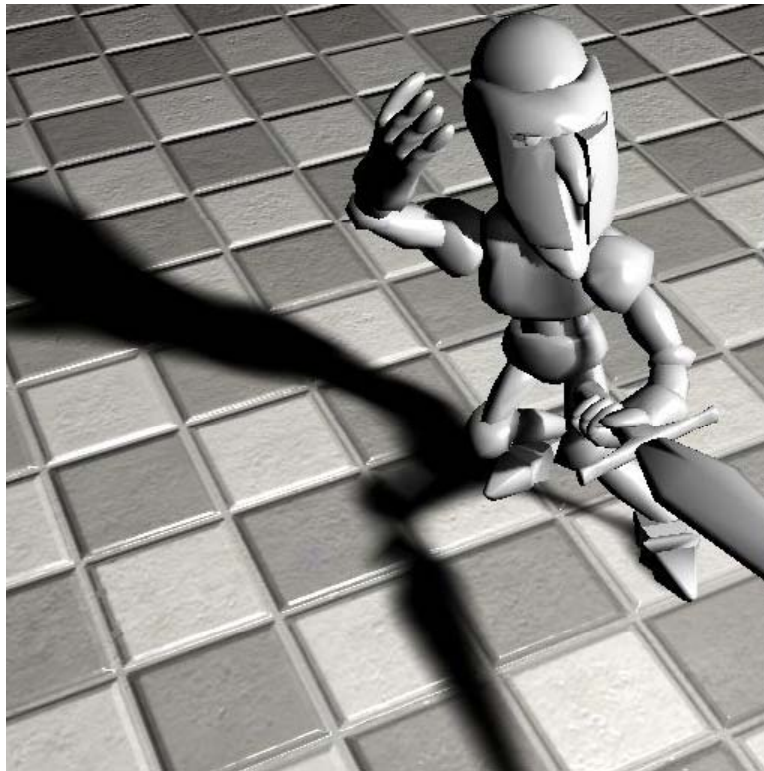
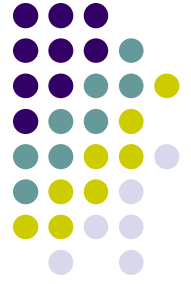
Different Atmospheres



Time Of Day



Shadows

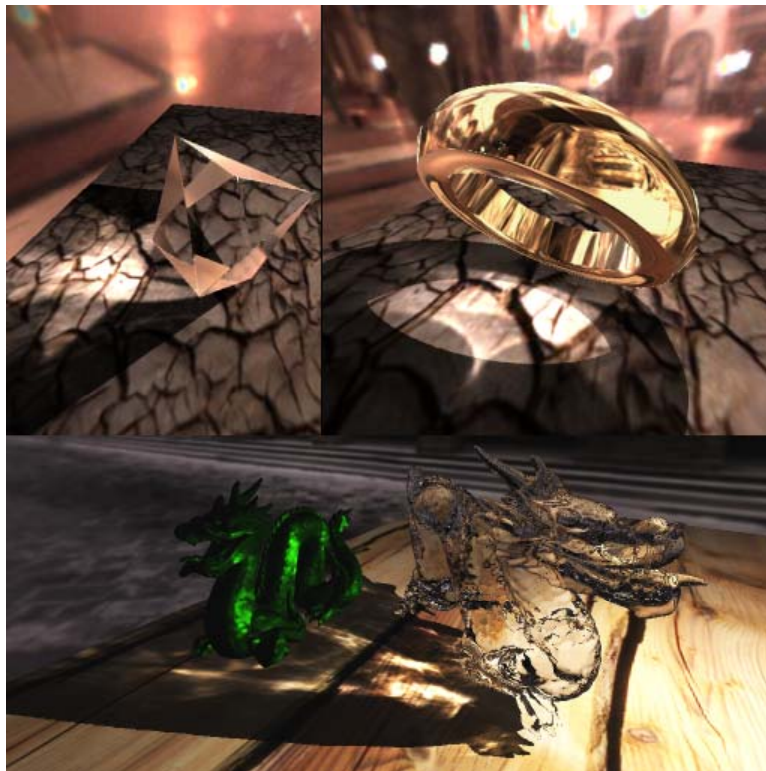
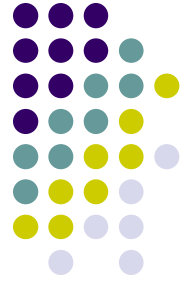


Variance shadow mapping
Courtesy Nvidia SDK 10



Courtesy Hellgate:London,
flagship studios inc

Caustics and Refraction

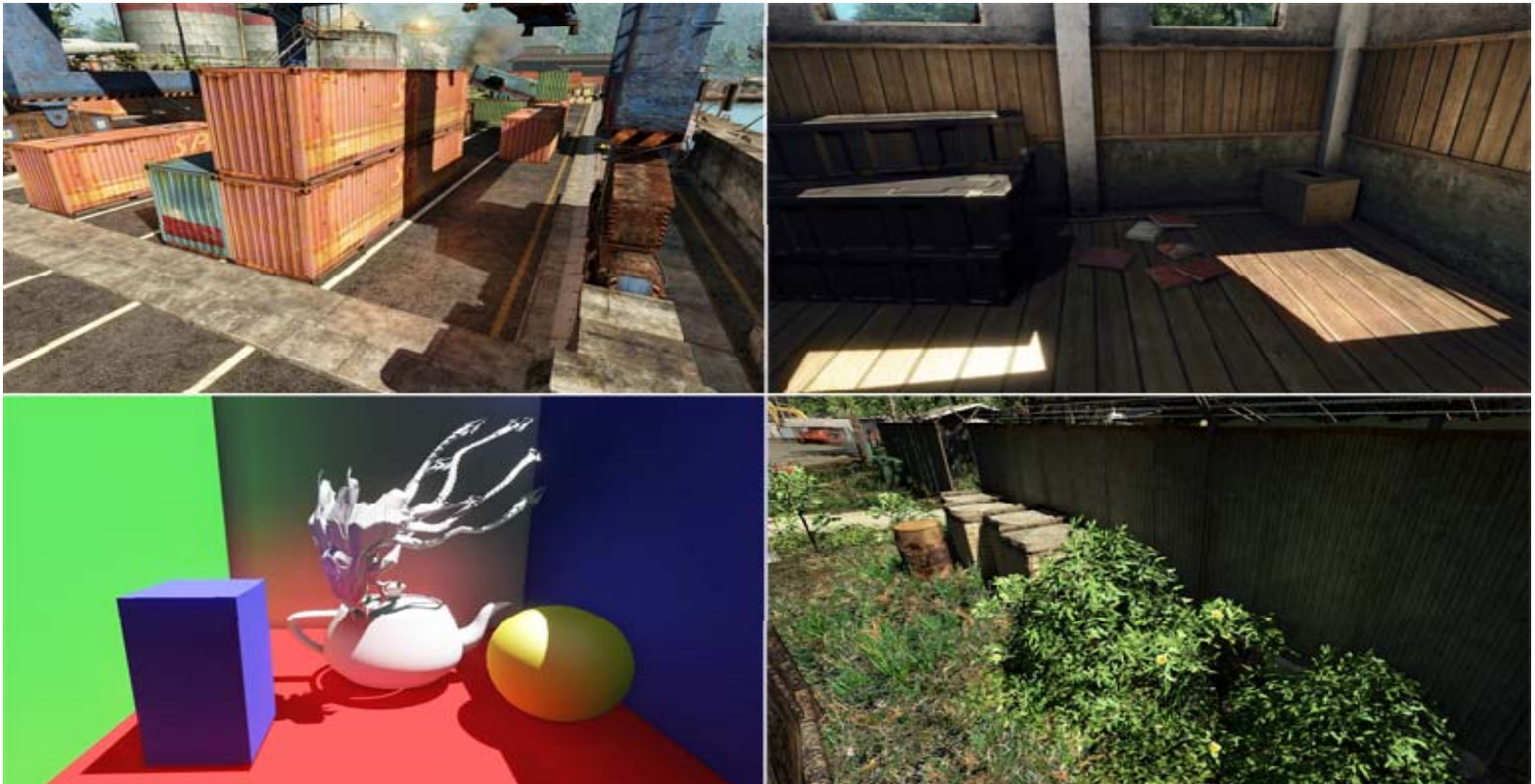


Courtesy Chris Wyman, Univ Iowa

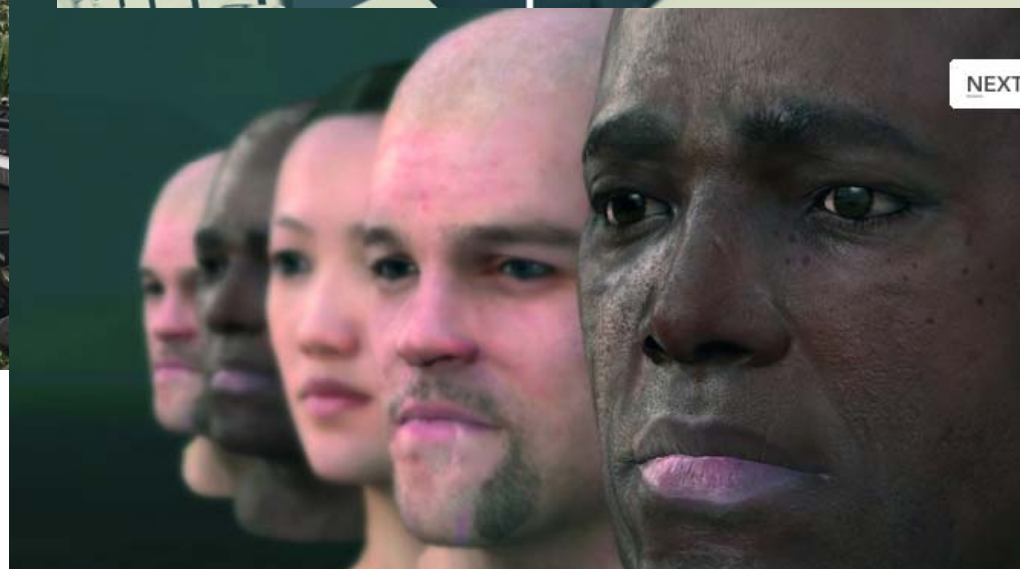
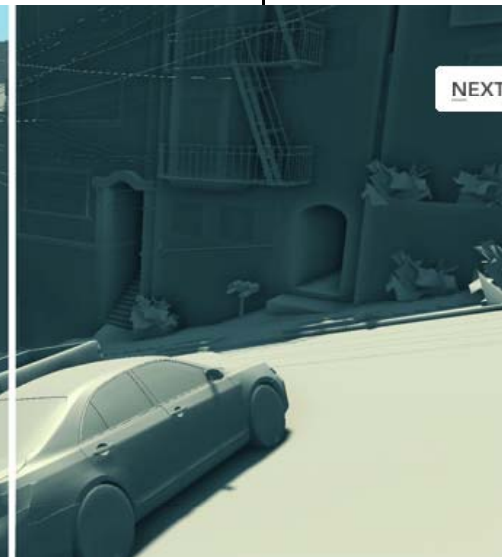
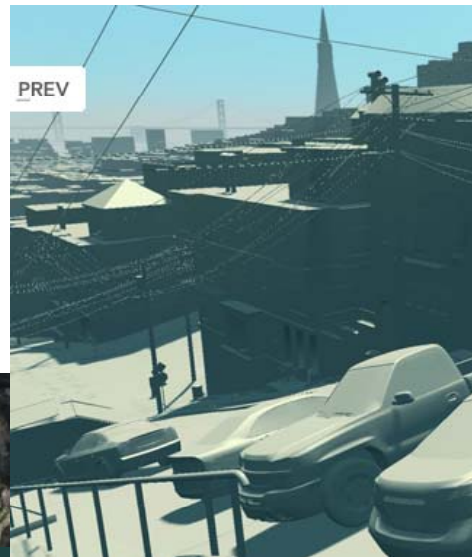
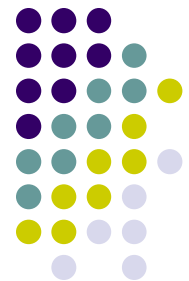
CryEngine 3:GI with Light Propagation Volumes



- State-of-the-art game engine
- Real-time simulation of massive, indirect physically-based lighting

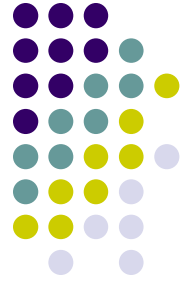


Crytek Crisis Engine Screenshots

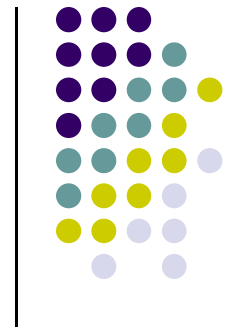


Demo

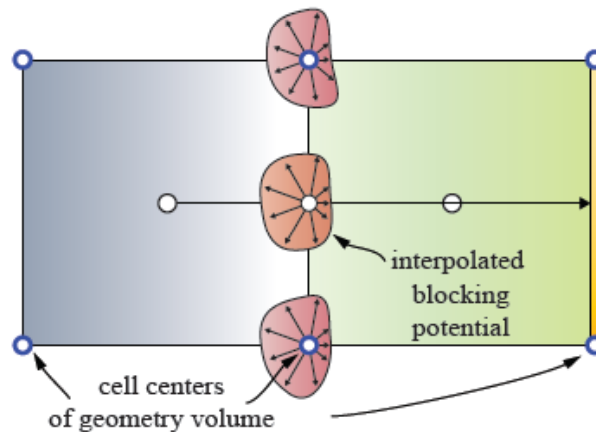
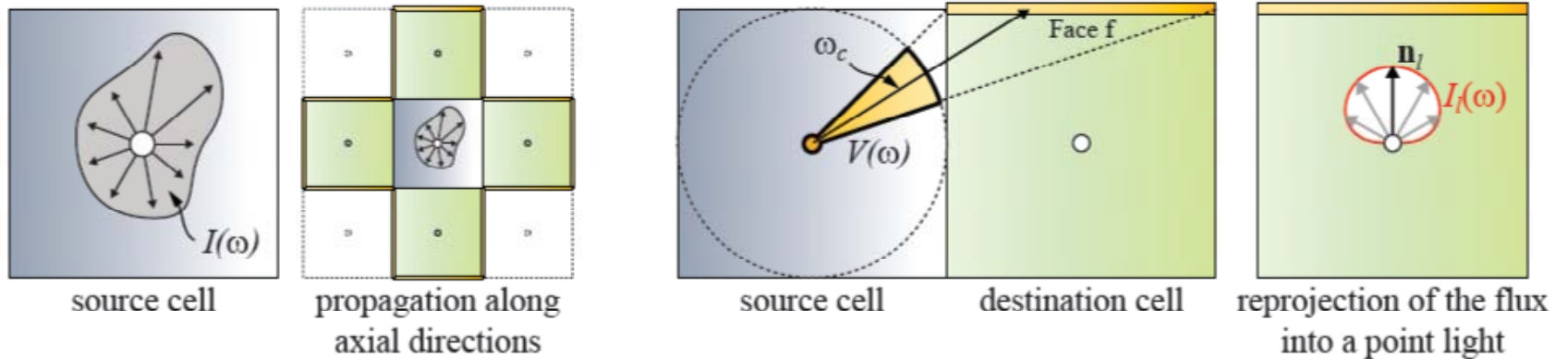
- Light Propagation Volumes [Demo](#)



LPV Idea



Main idea: represent light propagation as Virtual Point Lights (VPL)
Re-project VPL into adjacent cells

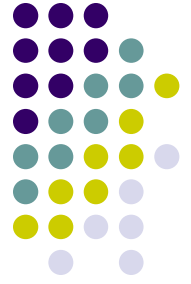


Trend 5: Screen-Space GI Techniques



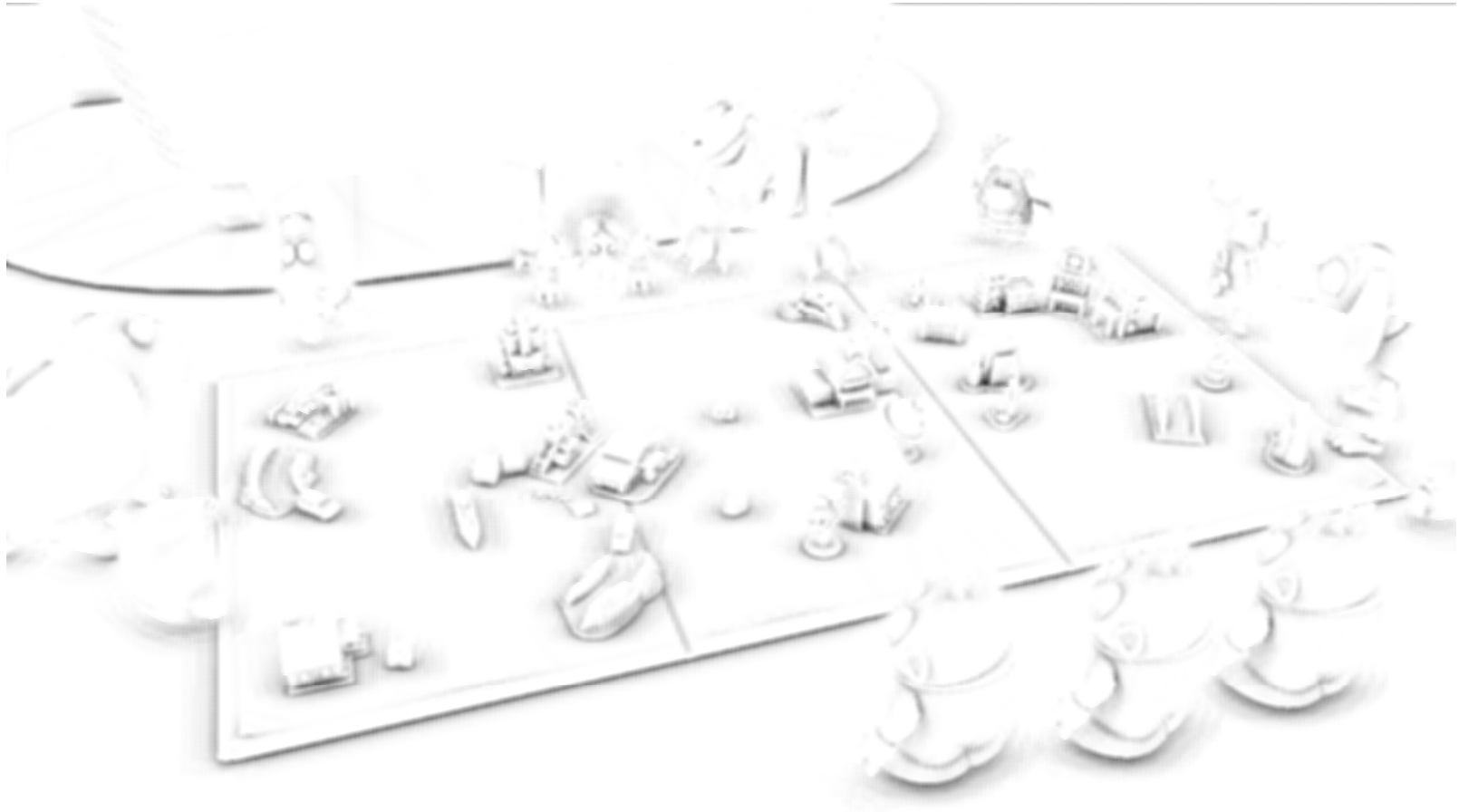
- Toy Story 3: Screen space Ambient Occlusion

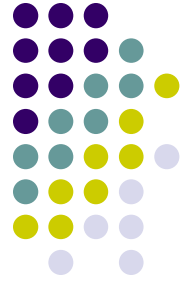




SSAO in Toy story 3

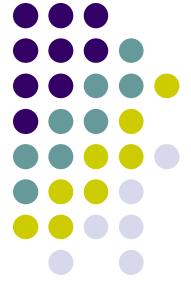
- Viewing just the ambient term of shading





Trend 6: Physics Engines on GPU

- Nvidia Physx engine
- SDK: developer.nvidia.com/object/physx_features.html
 - Complex rigid body object physics system
 - Advanced character control
 - Ray-cast and articulated vehicle dynamics
 - Multi-threaded/Multi-platform/PPU Enabled
 - Volumetric fluid creation and simulation
 - Cloth and clothing authoring and playback
 - Soft Bodies
 - Volumetric Force Field Simulation
 - Vegetation



References

- Pat Hanrahan, CS 348B, Spring 2005 class slides
- Yung-Yu Chuang, Image Synthesis, class slides, National Taiwan University, Fall 2005
- Kutulakos K, CSC 2530H: Visual Modeling, course slides
- UIUC CS 319, Advanced Computer Graphics Course slides
- <http://www.siggraph.org/education/materials/HyperGraph/raytrace/rtrace0.htm>
- Akenine Moller et al, Real-Time Rendering, 3rd edition
- Advances in Real-Time Rendering in 3D graphics and games, SIGGRAPH course notes 2009
- Anton Kaplanyan and Carsten Dachbacher, Cascaded light propagation volumes for real-time indirect illumination, in Proc. Si3D 2010
- Hao Chen and Natalya Tatarchuk, Lighting Research at Bungie, Advances in Real-Time Rendering in 3D Graphics and Games SIGGRAPH 2009 Course notes