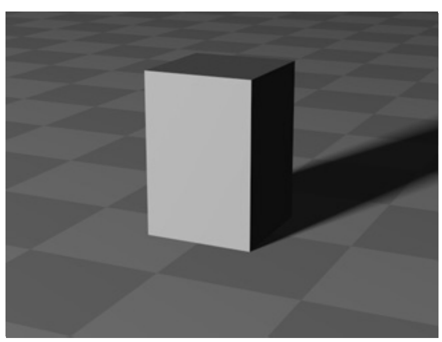
Lecture 14: Many Lights

**CS 6620, Spring 2009**

**Kavita Bala**
Computer Science
Cornell University

## Why Shadows?

- Crucial for spatial and depth perception



## Shadows

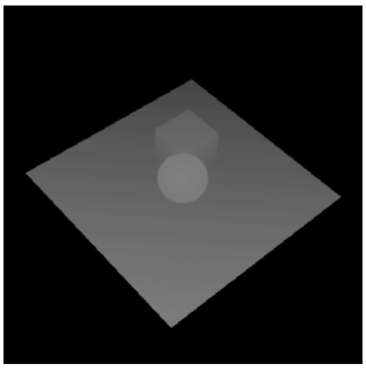
Methods for fast shadows:

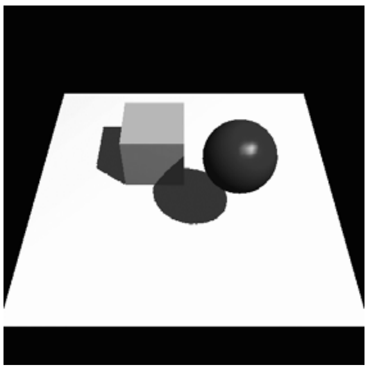
- Shadow Maps
- Shadow Volumes



## Shadow Maps

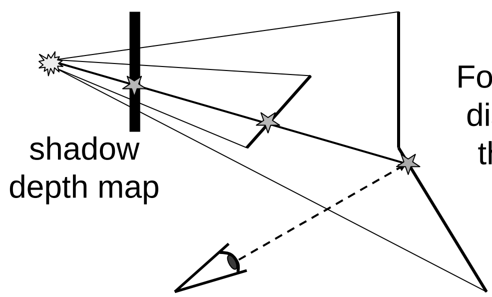
- Introduced by Lance Williams (SIGGRAPH 1978)
- Render scene from light's view
  - – black is far, white is close





## Using the Shadow Map

- When scene is viewed, check viewed location in light's shadow buffer
  - If point's depth is greater than shadow depth, object is in shadow




For each pixel, compare distance to light with the depth stored in the shadow map



## Shadow Mapping: Pass 1

- Depth testing from light's point-of-view
  - Two pass algorithm
- First, render depth buffer from light's point-of-view
  - Result is a "depth map" or "shadow map"
  - A 2D function indicating the depth of the closest pixels to the light
  - This depth map is used in the second pass

## Shadow Mapping: 2nd pass

- Second, render scene from the eye's point-of-view

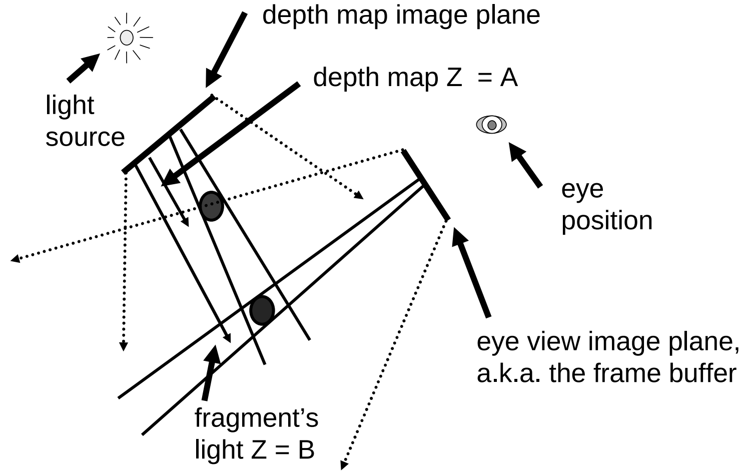


## Shadow Mapping: Comparison

- For each rasterized fragment
- Two values
  - A = Z value from depth map at fragment's light XY position
  - B = Z value of fragment's XYZ light position
- If (B > A),
  - There must be something closer to the light than the fragment
  - So, fragment is shadowed
- If A and B are approximately equal, the fragment is lit

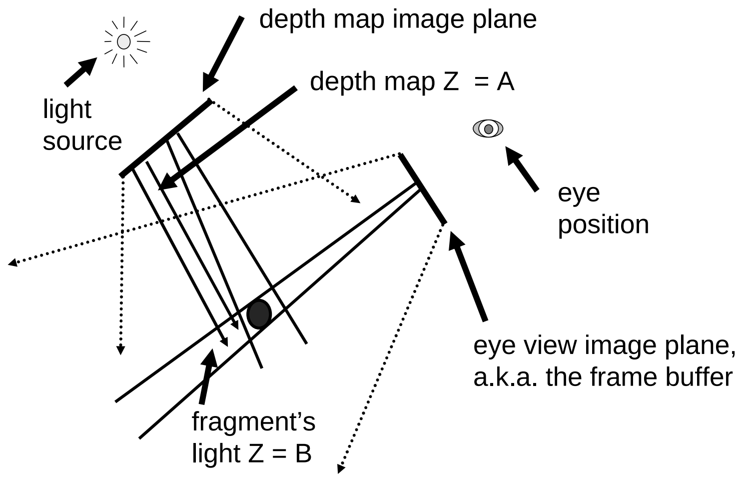
# Example: Shadowed

The A < B shadowed fragment case

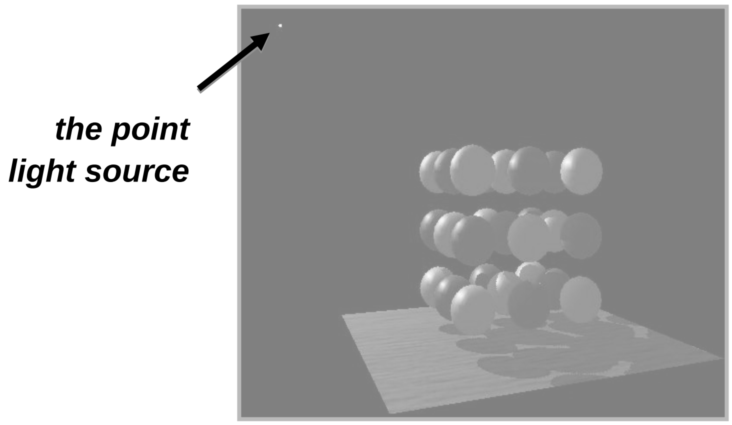





# Example: Visible

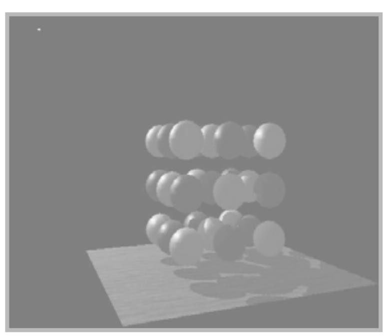


# Example

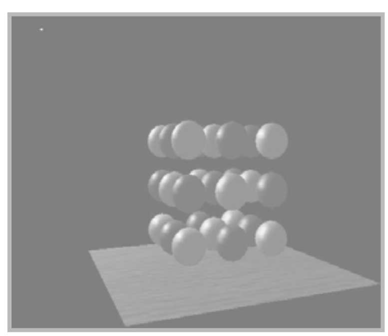





# Example



*with shadows*



*without shadows*

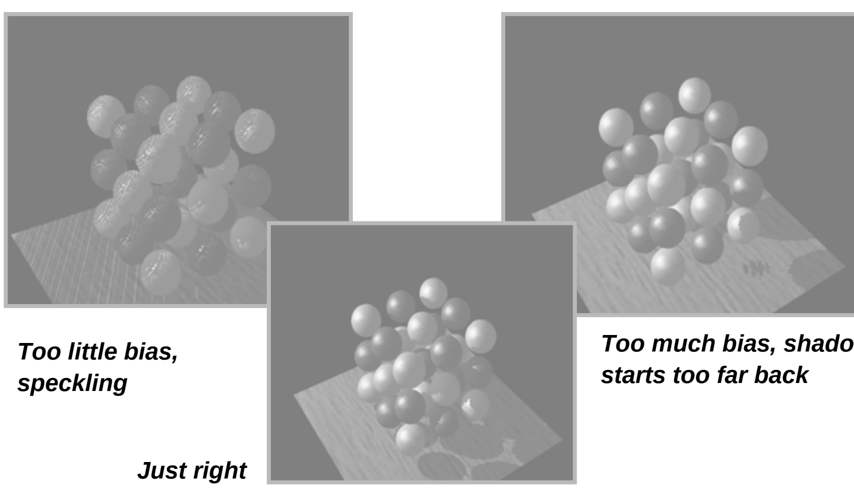
# Shadow Map Issues

- Get speckling. Why?
- Use triangle Ids? Object ids?
  - Meshes?
- Bias: b
  - If (p.z < SM(x,y)+b) p in shadow
    - Where (x,y) is the position in SM to which the point p projects
  - If b is large, could get light where it is shadowed
  - If b is small, could get speckling



# Bias Issues

- How much polygon offset bias depends



***Too little bias, speckling***

***Just right***

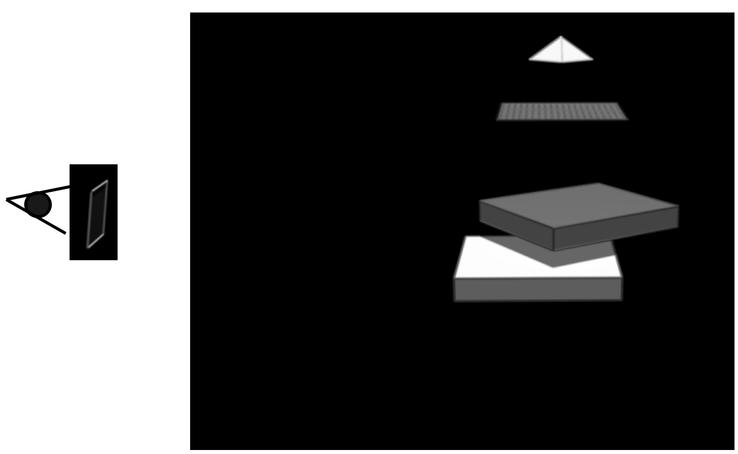
***Too much bias, shadow starts too far back***

## Shadow Map Issues

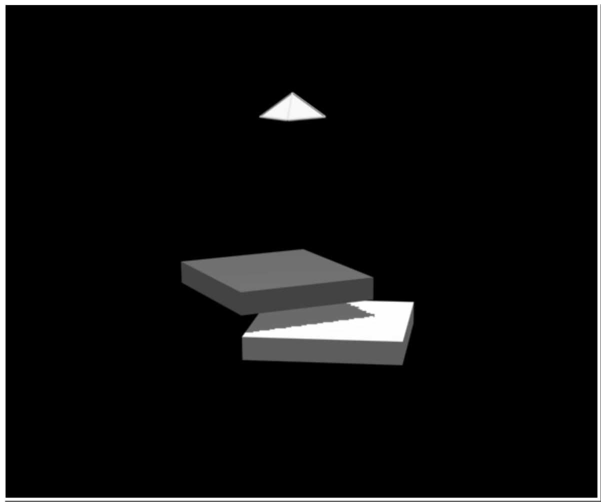
- Can only cast shadows over a frustum
  - Use 6 (like a cube map)



## Aliasing (Distant)

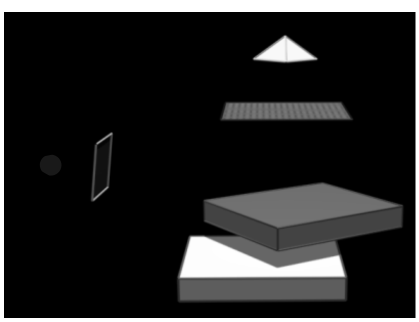


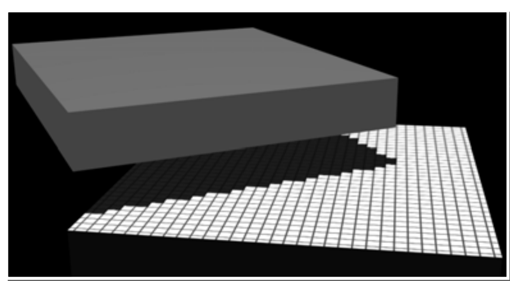
## Aliasing in Eye View (Distant)



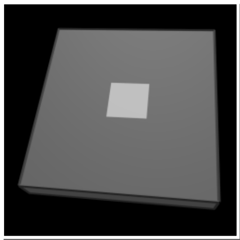


## Aliasing (Close)

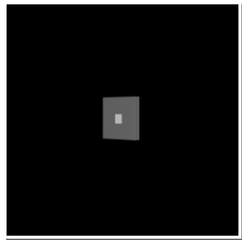




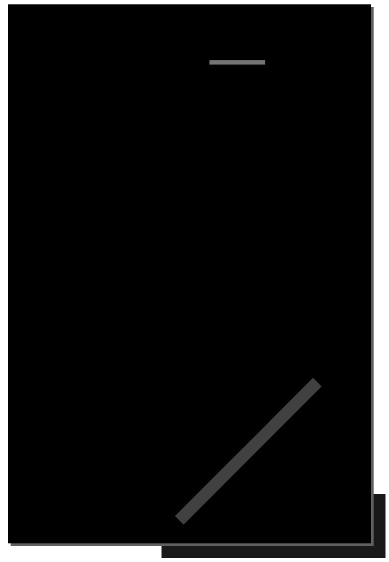
# Why does Aliasing arise?



Eye View



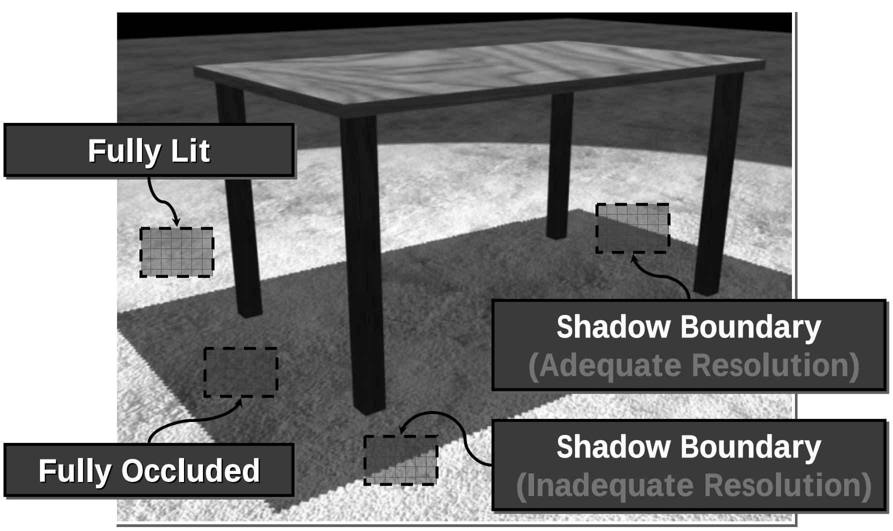
Shadow Map



Eye View Projected Area != Shadow Map Projected Area



# Where does aliasing occur?



## Handle Aliasing

- Adaptive Shadow Maps
  – Fernando, Fernandez, Bala, Greenberg [SIG01]
- Perspective Shadow Maps
  – Stamminger, Drettakis
- Many, many variants

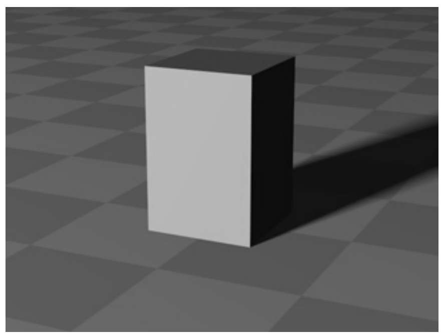


## Implementation Issues

- Shadow maps are dominating
- Hardware support for them
- Remember, hardware does perspective-correct texturing
- Shadowmaps use projective texturing

## Soft Shadows

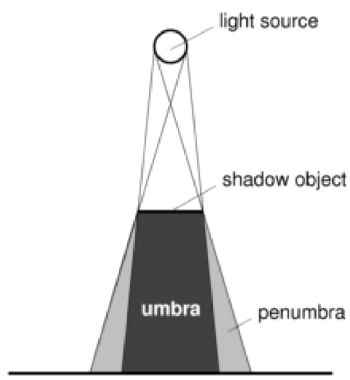
- Soft shadows appear more natural
  - Hard to get soft shadows in hardware
- MANY systems that try to approximate soft shadows





## Soft Shadows

- Soft shadows appear more natural
  - Hard to get soft shadows in hardware



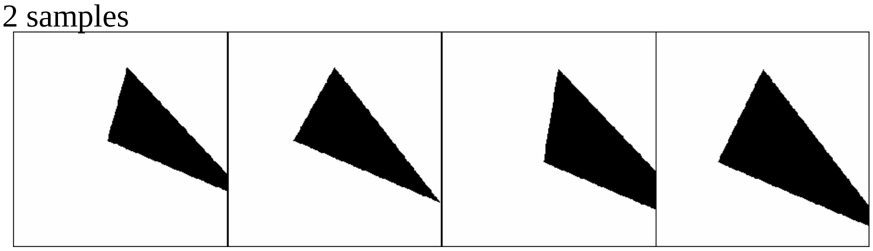
# Heckbert and Herf

- Use accumulation buffer
- Render shadows from multiple point lights over the area light
- Accumulate shadows

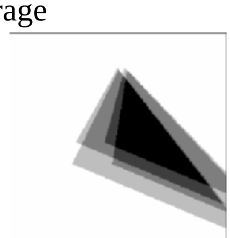


# Soft Shadows: Heckbert/Herf

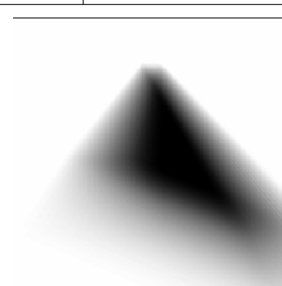
2 x 2 samples



average



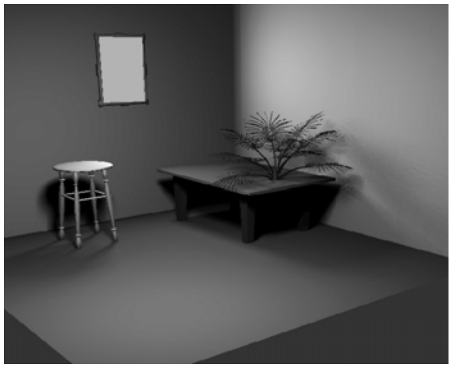
16 x 16 samples



Images courtesy of Michael Herf and Paul Heckbert

## Heckbert/Herf Soft Shadows

- Advantage: gives true penumbra
- Limitations: overlapping shadows are unconvincing unless a lot of passes are made



Images courtesy of Michael Herf and Paul Heckbert



## Summary

- Shadow maps
  - User projective texturing: requires hardware support/shaders
  - Pros: simple, fast
  - Cons: Aliasing, Bias, Hard shadows
  - One shadow map per light
  - Render scene twice per frame
    - If static, can reuse

- Soft shadows
  - Use accumulation buffers

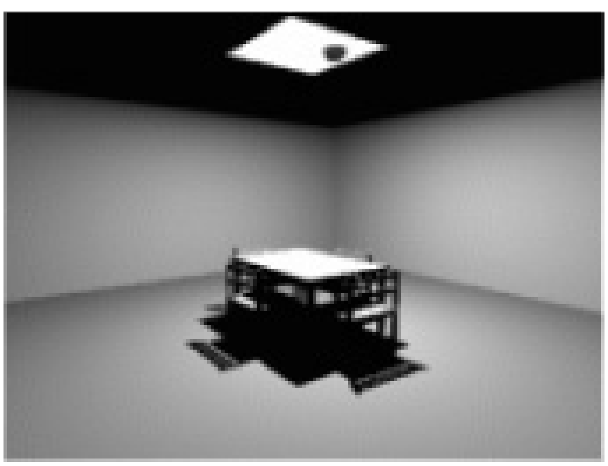
## Instant radiosity

- Interesting hack:
  - Bounce ‘photons’ from light sources
    - Same as first pass from photon mapping
  - Have *point lights* at bounces!
  - Do cheap rendering using
    - Fast ray tracing or graphics hardware (using SM)



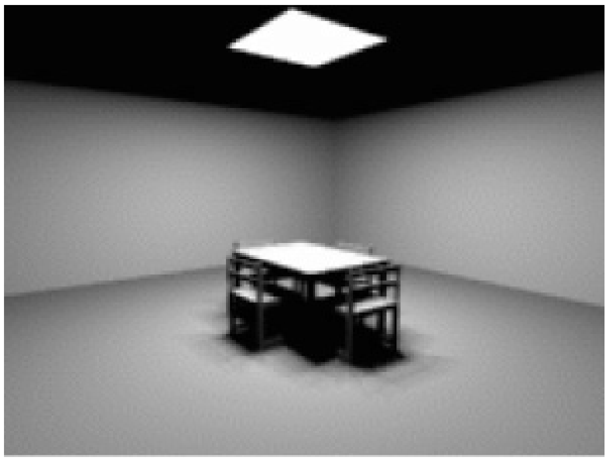
## Instant radiosity demo

- Point lights on the light source



# Instant radiosity demo

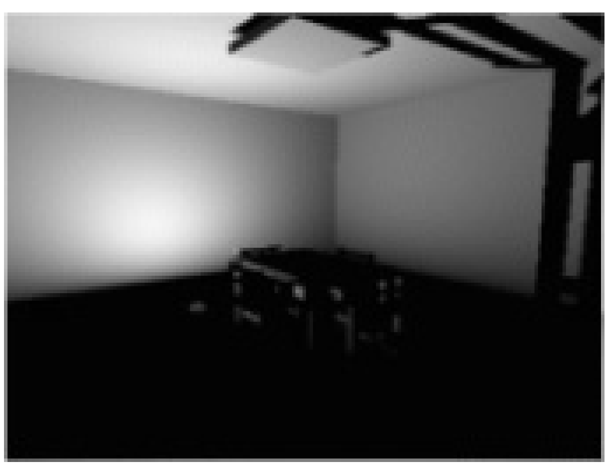
- Direct illumination



Average of previous 10 renders

# Instant radiosity demo

- Point lights one ‘bounce’ into scene



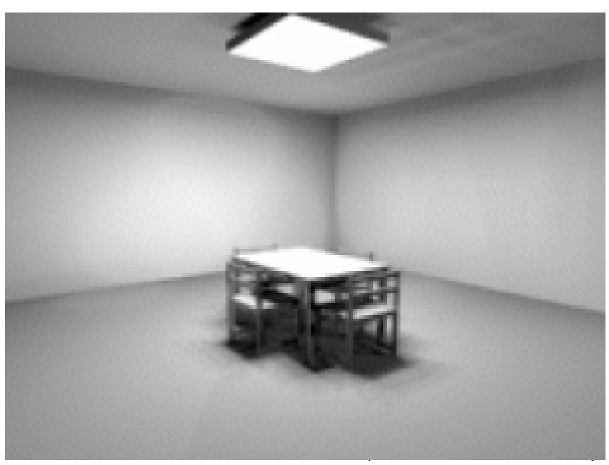
## Instant radiosity demo

- Radiosity

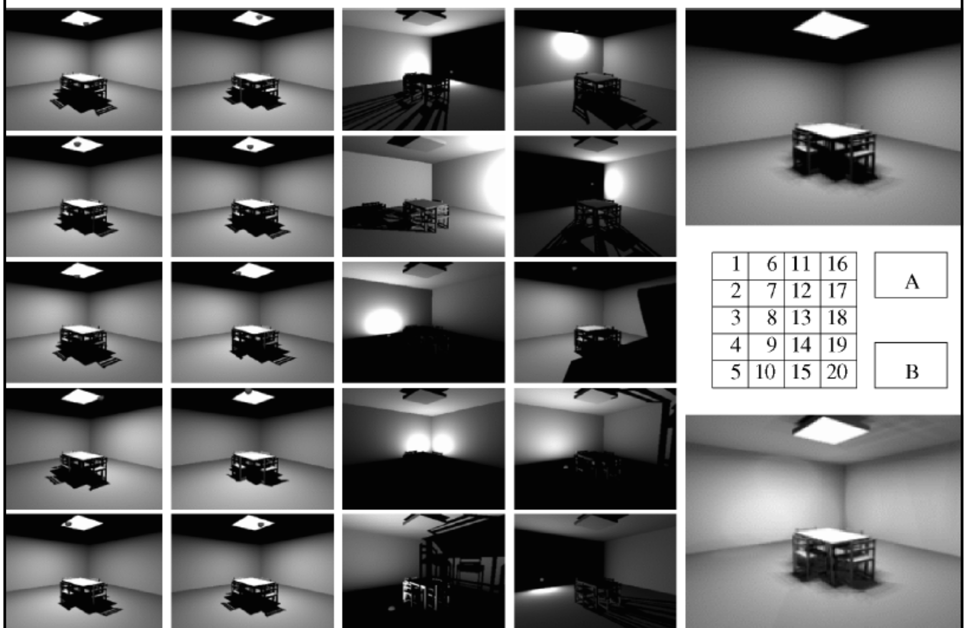
Average of previous 10 renders

## Instant radiosity demo

- Result

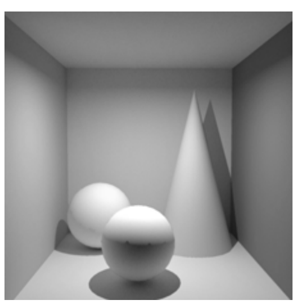


Instant radiosity

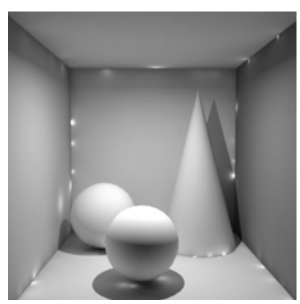





# The Problem: Diffuse Scene

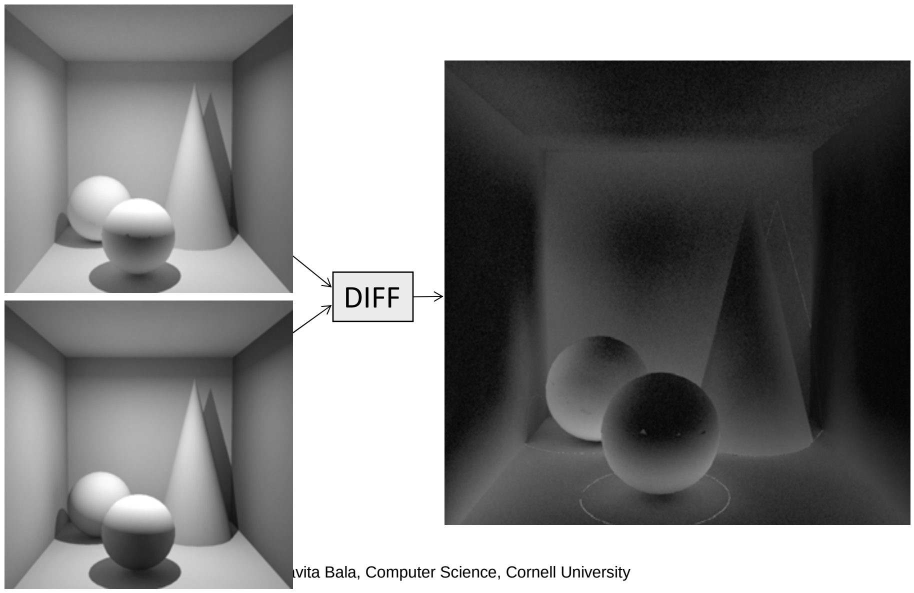


Path tracer
272 seconds
(Core 2 Duo, 2 threads)



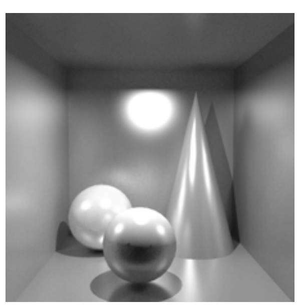
**no clamping**

# How Much is Missing?

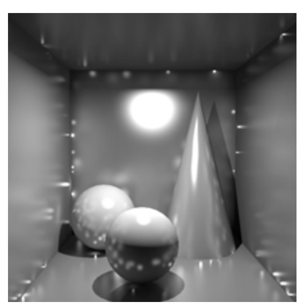





# The Bigger Problem: Glossy Scene

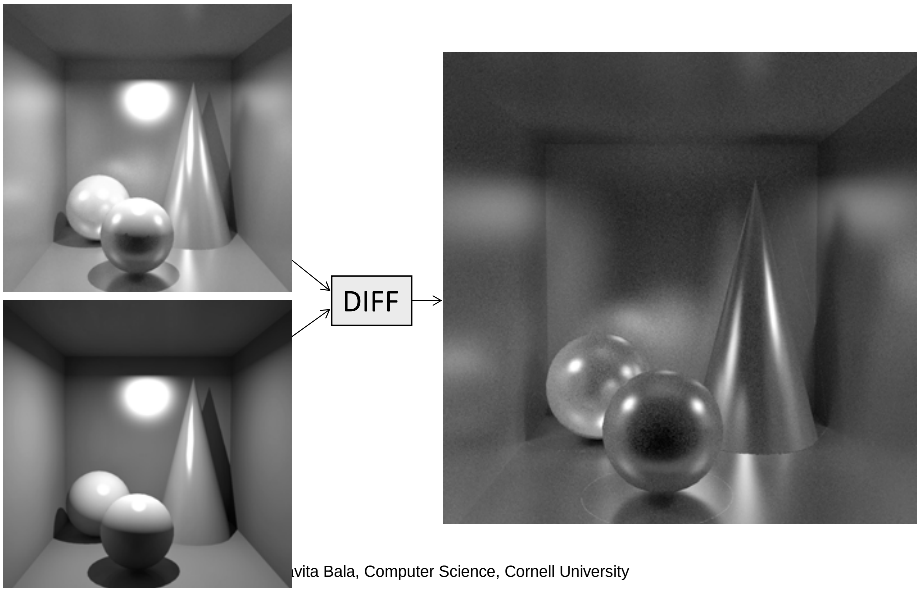


Path tracer
569 seconds
(Core 2 Duo, 2 threads)



**no clamping**

# How Much is Missing?






# Advantage

- Point lights are placed randomly
- But the same set of lights for each pixel
  – Still Monte Carlo, but samples are *positively correlated*
- This means less noise
  – But maybe banding

- Disadvantage
  – Clamping!

## Many Lights

- Most techniques work for a single light source
- Many light sources
  - For environment maps
  - For indirect illumination
- Treat it is a single integration domain
  - Importance sample lights
  - Importance sampling (with visibility) still hard problem



## Research on many lights

- Ward '91
- Shirley, Wang, Zimmerman '94
- Fernandez, Bala, Greenberg '02
  - Donikian, Fernandez.. '06
- Lightcuts '05

## Shirley, Wang, Zimmerman ‘94

- Try to avoid linear cost of evaluating lights
- Separate lights into
  – Set of important lights (a small set)
  – Set of “dim” lights (large set)
- Construct pdf using:
  – all important lights
  – 1 out of all the dim lights
- Importance sample these lights



## Shirley, Wang, Zimmerman ‘94

- Region of influence for important lights
  – Octree cells in region of influence have light in important set
- However, the partitioning into important and dim sets remains hard
- Also, still are not taking visibility into account