Realistic Image Synthesis

- Lightcuts -

Philipp Slusallek
Karol Myszkowski
Gurprit Singh

Goals of Lightcuts

- **Efficient, accurate complex illumination**
- **In realistic and complex environments**

Environment map lighting & indirect
Time 111s

Textured area lights & indirect
Time 98s

(640x480, Anti-aliased, Glossy materials)

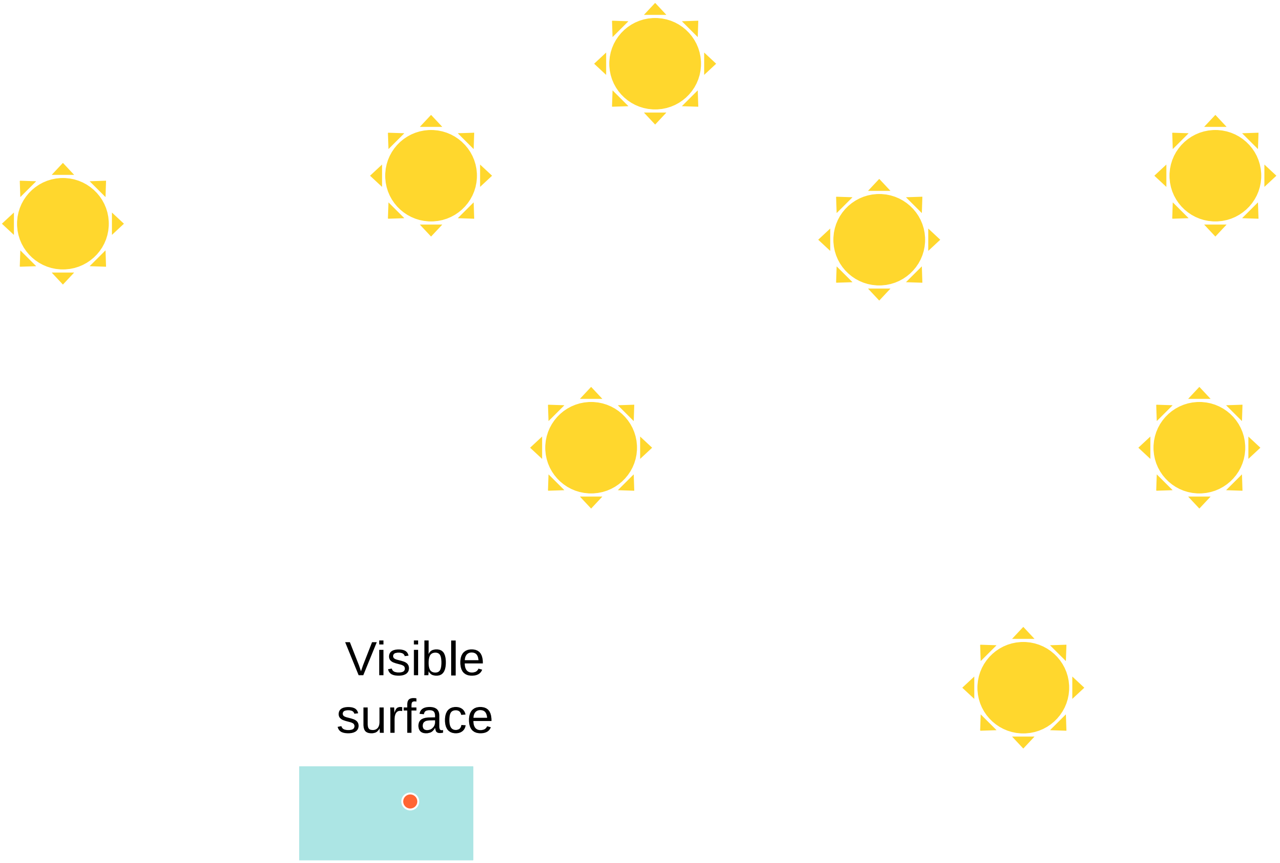
Motivation

- **Hierarchies in Global Illumination**
  - Only used in FE methods so far
  - Can greatly improved performance
    - Take advantage of $1/N^2$ power fall-off
    - Group together light from distant objects & handles it together
    - Can reduce computational complexity from $O(N^2)$ to $O(N)$

- **Question: How to use them in MC-style algorithms**
  - Key idea: Sample points generated from lights and from camera
  - Could group them hierarchically, if generated in advance
  - Would handle illumination of a group as one sample
  - Allows adaptive/progressive refinement
  - Key issues:
    - How to group: Must have criteria for grouping (e.g. by “similarity”)
    - When to refine: Must have an efficient “oracle”

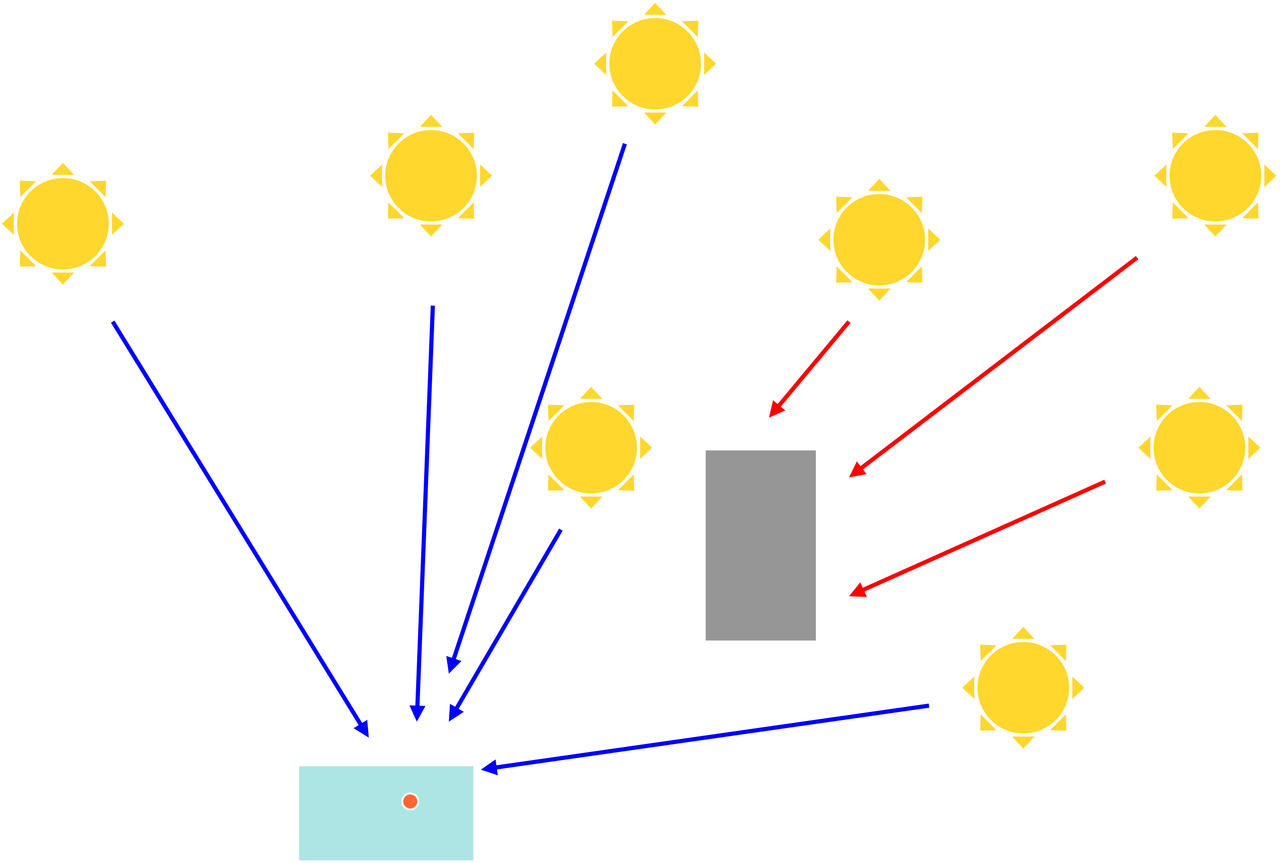
Lightcuts Problem

- **Many light samples**



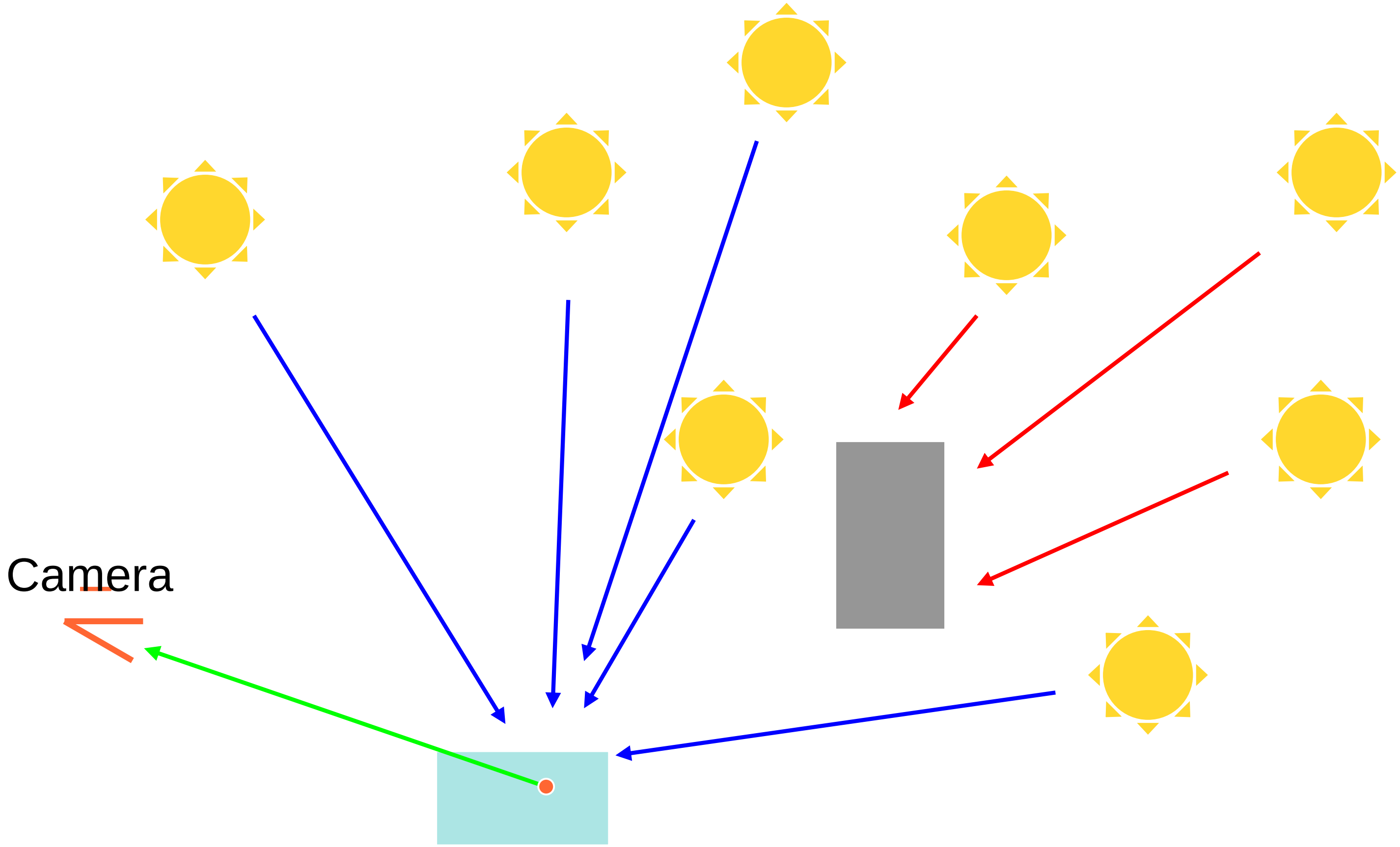
Lightcuts Problem

- **Complex visibility**



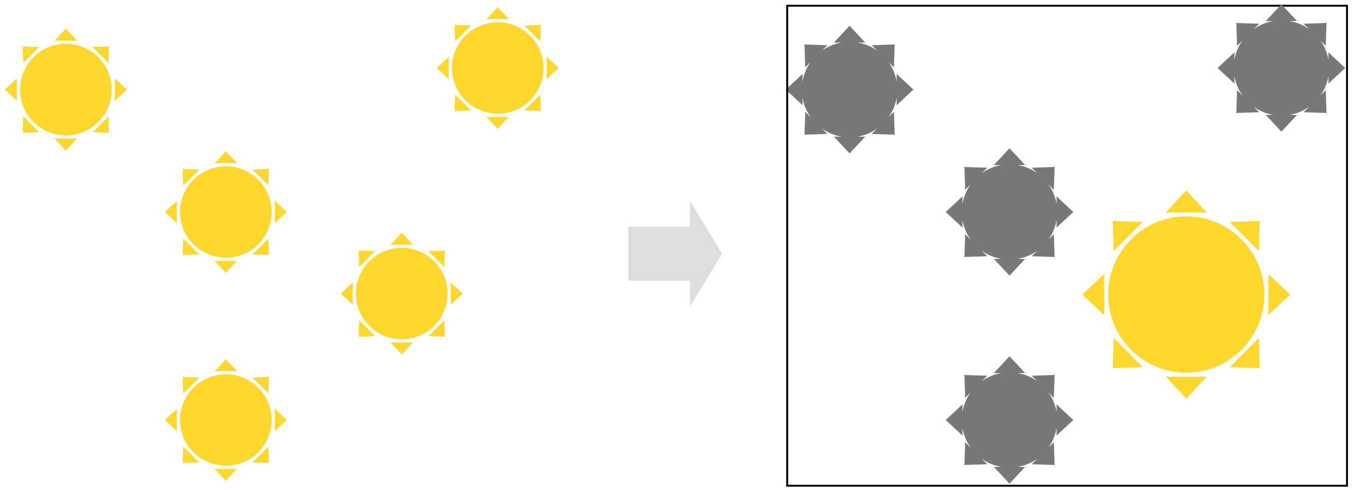
Lightcuts Problem

- **Material properties with complex reflection**



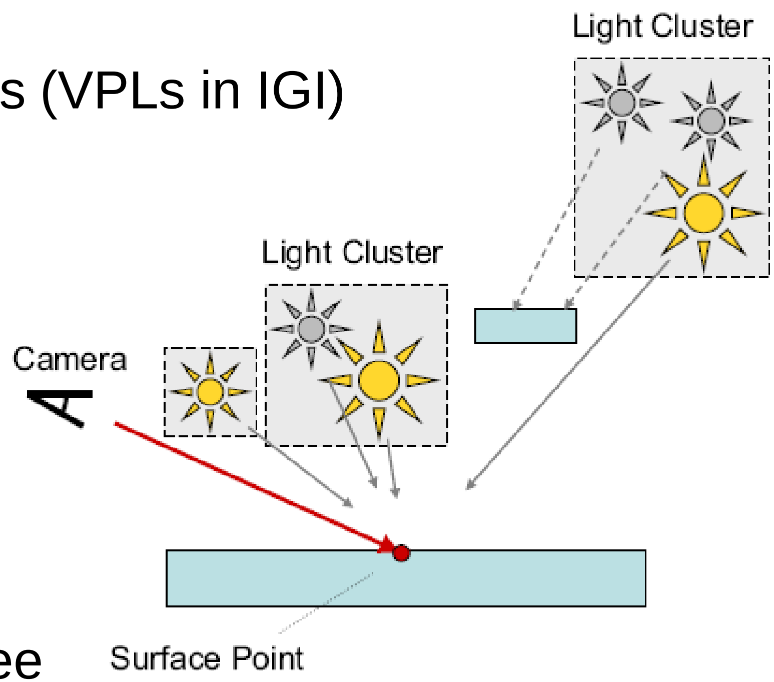
Key Concepts

- **Light Cluster**
  – Approximate many lights by a single brighter light (the representative light)



Clustering of Light Samples

- **Sources of (many) light samples**
  - Point lights
  - Sampled area lights
  - Sampled HDR environment lighting
  - Generated secondary lighting samples (VPLs in IGI)
- **General idea**
  - Group light samples into binary tree
  - Leafs are the input light samples
  - Inner nodes combine illumination from their children
    - Choose a representative location from among children
    - Combine and bound attributes
  - Illumination uses a ***cut*** through the tree
    - Adaptively combines far away lights into one
    - Samples the integral evenly given bounds on power contribution, solid angle, visibility, and angular falloff



Criteria for Clustering

- **Contribution from a cluster**
  - Given terms for material (M), geometry (G), visibility (V) and the intensity (I) of the (clustered) child light samples
  - Illumination from the cluster is then given as

$$L_C = \sum_{i \in C} M_i(x_i, \omega_o) G_i(x_i) V_i(x_i) I_i$$

- **Approximation**
  - However, this is too costly and is approximated as by a representative light sample j

$$\tilde{L}_C \approx M_j(x_j, \omega_o) G_j(x_j) V_j(x_j) \tilde{I}_j \qquad \tilde{I}_j = \sum_{i \in C} I_i$$

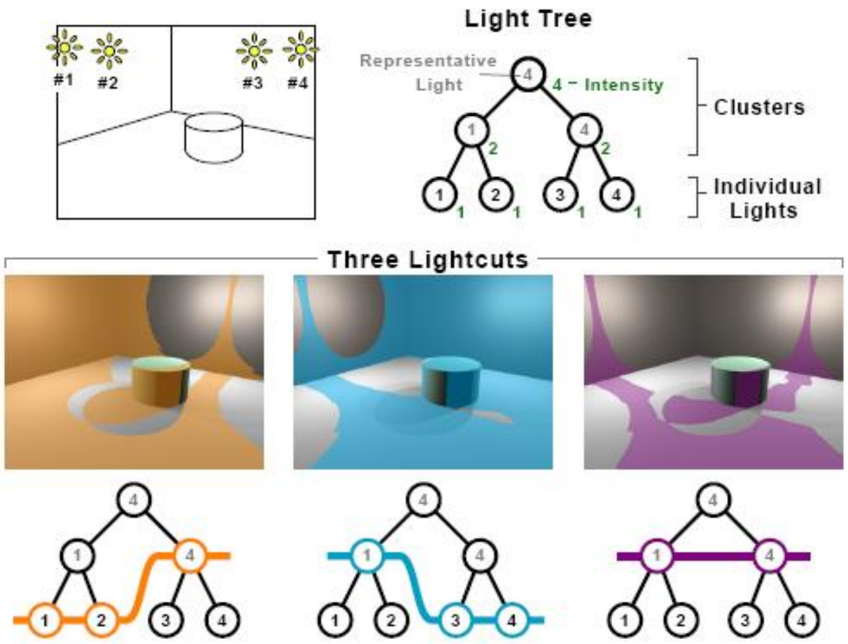
  - All properties are taken from representative, except light intensity
  - Create a full cluster up to a single root node
- **Issue**
  - Must have some way to bound the error of the approximation

Building the Light Tree

- Lights are split into types: Omni, oriented, and directional lights
  - Build a tree for each (but conceptually one big tree)
  - Directional lights are handled as point lights on a unit sphere
- Each cluster stores
  - Links to two children
  - Representative light (randomly chosen among children, ~ intensity)
  - Total intensity I_C (sum over all children)
  - Axis aligned bounding box
  - Oriented bounding cone (for oriented lights)
- Greedy bottom up build:
  - In each step create cluster that minimizes total cost
- Cost model: $I_C(\alpha_C^2 + c^2(1 - \cos\beta_C)^2)$
  - α_C : Diagonal length of bounding box
  - β_C: Half angle of bounding cone (of light directions)
  - c: Constant for relative scaling of spatial/directional data
    - Set to half the scenes Bbox for oriented lights, zero otherwise

Choosing a Cut

- **General Approach**
  - Set the cut to be the root node
  - Choose the node from the cut with worst error
  - Refine this node
    - Replacing it with its two children
  - Terminate if relative error is below 1%
    - Can be computed because we have approximated illumination due to existing cut
    - Criterion due to Weber's law
      - Relative perception
    - In the paper they use 2% without artifacts



Illumination Equation

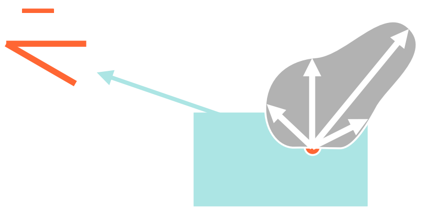
$$\text{result} = \sum_{\text{lights}} M_i \; G_i \; V_i \; I_i$$

Material term

Geometric term

Visibility term

Light intensity



Illumination Equation

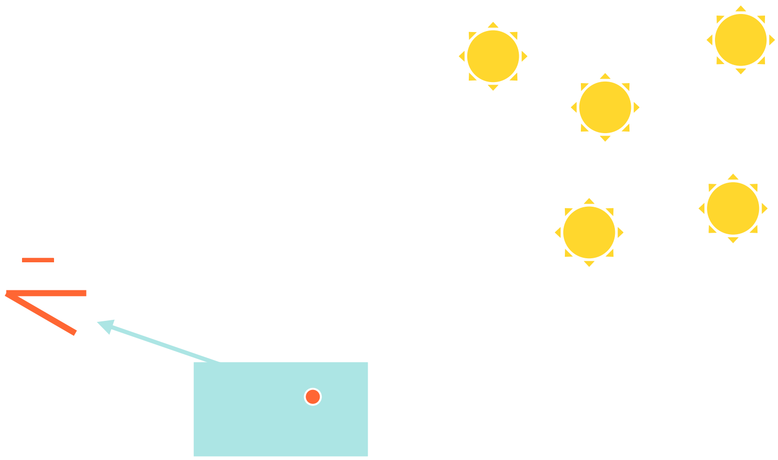
$$\text{result} = \sum_{\text{lights}} M_i \, G_i \, V_i \, I_i$$

Material term

Geometric term

Visibility term

Light intensity



Illumination Equation

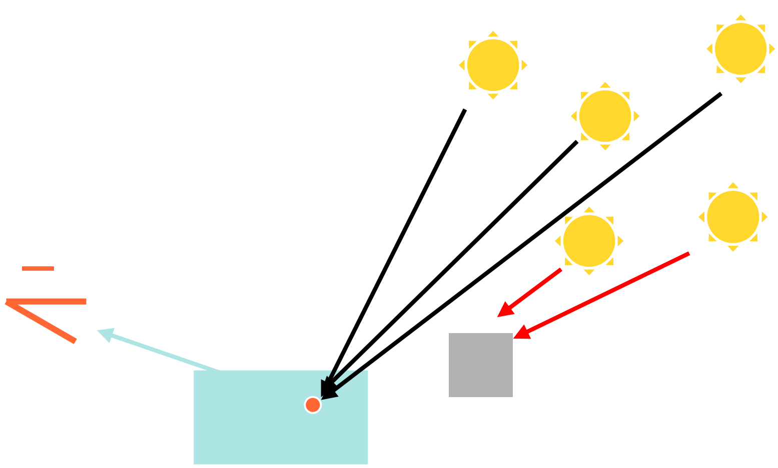
$$\text{result} = \sum_{\text{lights}} M_i \; G_i \; V_i \; I_i$$

Material term

Geometric term

Visibility term

Light intensity



Cluster Approximation

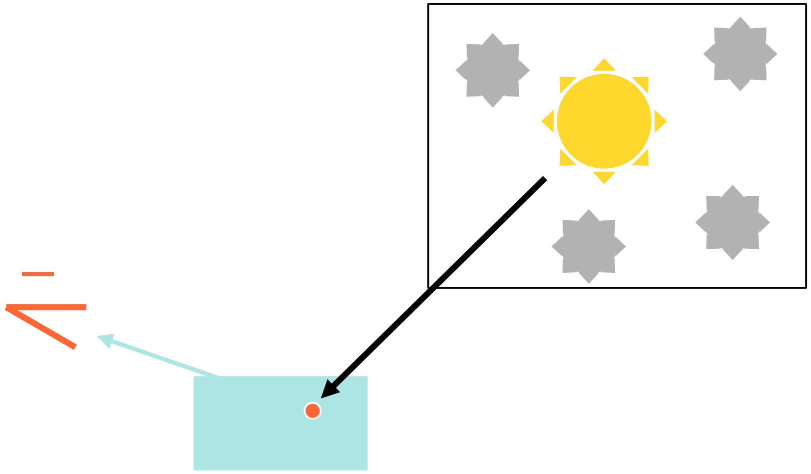
$$\text{result} = \sum_{\text{lights}} M_i \; G_i \; V_i \; I_i$$

Material term

Geometric term

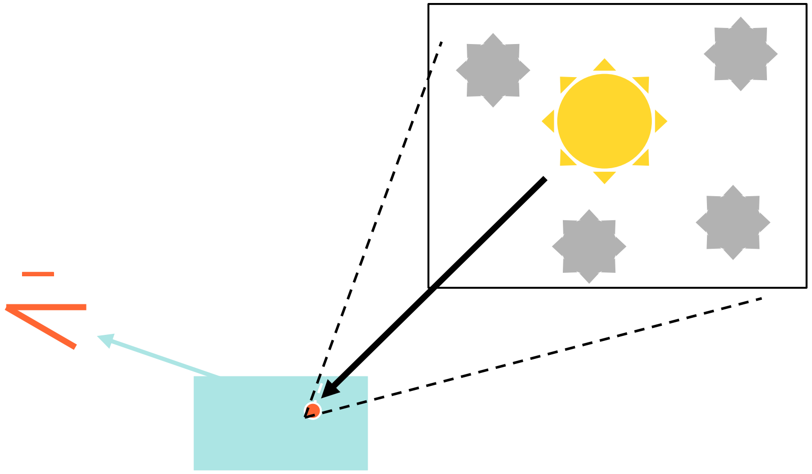
Visibility term

Light intensity



Cluster Error Bound

$$\text{error} \leq M_{\text{ub}} G_{\text{ub}} V_{\text{ub}} \sum_{\text{lights}} I_i$$



Bound each term

- Visibility <= 1 (trivial)
- Intensity is known
- Bound material and geometric terms using cluster bounding volume

Kitchen, 388K polygons, 4608 lights (72 area sources)

Kitchen, 388K polygons, 4608 lights (72 area sources)

Combined Illumination

Lightcuts 128s
4 608 Lights
(Area lights only)

Lightcuts 290s
59 672 Lights
(Area + Sun/sky + Indirect)

Combined Illumination

Lightcuts 128s
4 608 Lights
(Area lights only)
Avg. 259 shadow rays / pixel

Lightcuts 290s
59 672 Lights
(Area + Sun/sky + Indirect)
Avg. 478 shadow rays / pixel
(only 54 to area lights)

Extended Versions of Lightcuts

- **Reconstruction Cuts**
  - Operates in image space
  - Starts Lightcuts at coarse pixel grid
  - Interpolates either colors or lighting info, or resamples
  - Refines pixel grid where necessary (based on material, shadow info)

- **Multi-Dimensional Lightcuts**
  - Realizes that antialiasing, motion blur, etc. require many samples per pixel
    - Inefficient if Lightcut is recomputed for each of them
  - Instead build hierarchy of pixel samples and VPLs
    - Needs clever error bounds
  - Traverse simultaneously, subdividing either cut based on cost function