Scalable many-light methods

Jaroslav Křivánek
Charles University, Prague

SEVENTH FRAMEWORK PROGRAMME

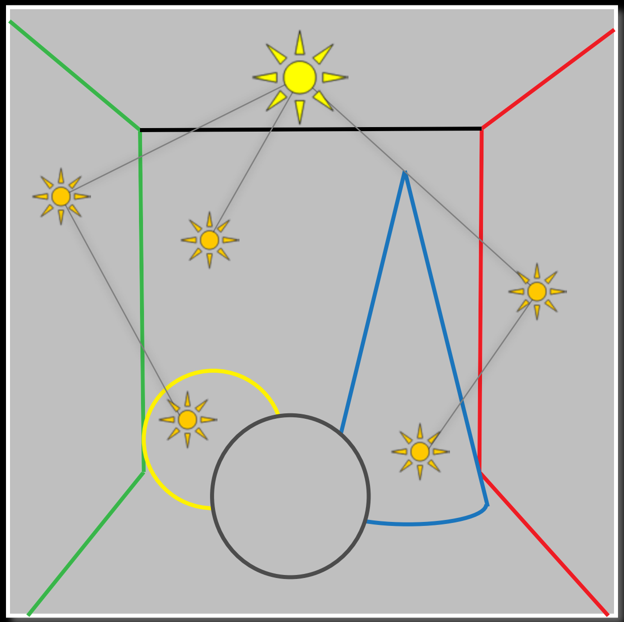
MARIE CURIE ACTIONS

Instant radiosity

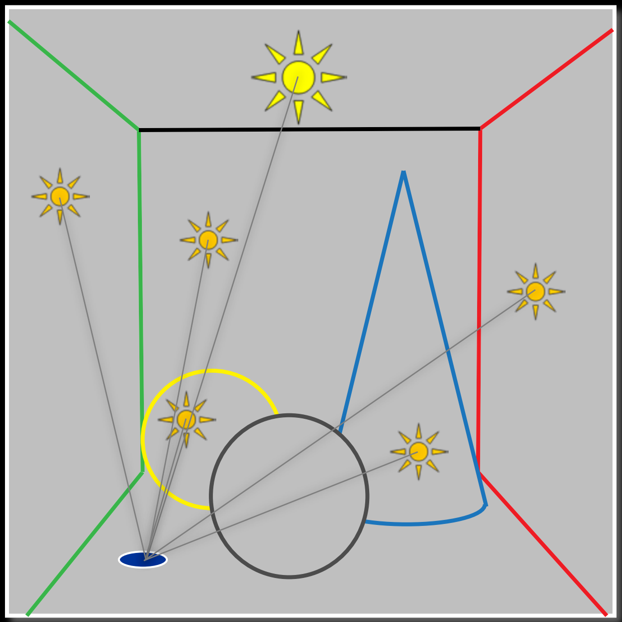
- Approximate indirect illumination by

  Virtual Point Lights (VPLs)

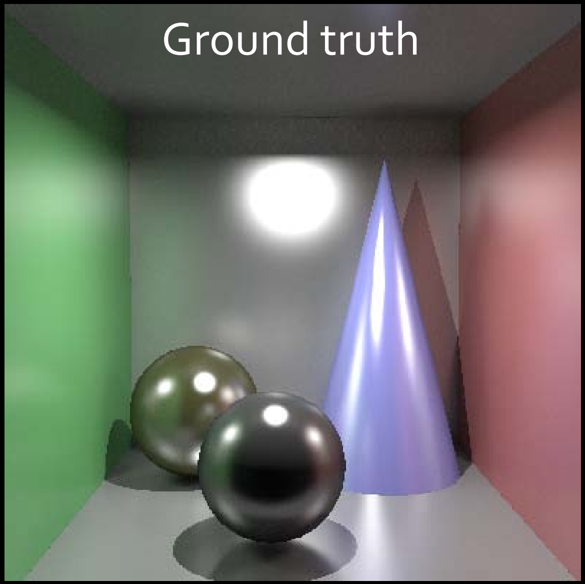
1. Generate VPLs

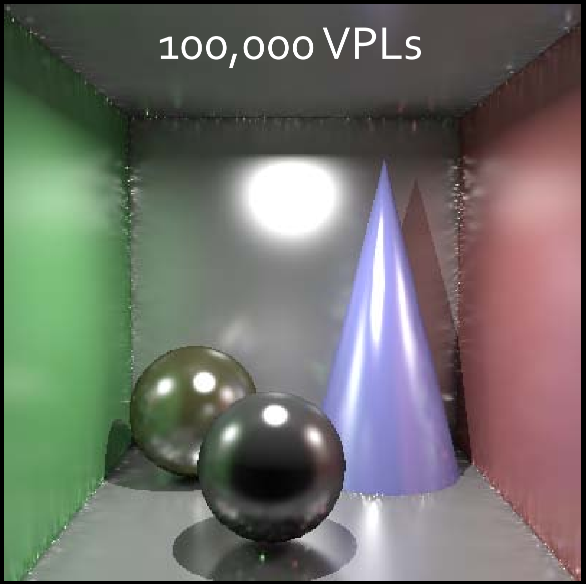


2. Render with VPLs



Instant radiosity with glossy surfaces





- Large number of VPLs required
  - True even for diffuse scenes
  - Scalability issues

Scalable many-light methods

1. Generate many, many VPLs
2. Use only the most relevant VPLs for rendering

- Choosing the right VPLs
  - Per-pixel basis
    - Lightcuts [Walter et al 05/06]
  - Per-image basis
    - Matrix Row Column Sampling [Hašan et al. 07]

Scalability with many lights

Approach #1:

Lightcuts &
Multi-dimensional lightcuts

Walter et al., SIGGRAPH 2005/06

Slides courtesy Bruce Walter:

http://www.graphics.cornell.edu/~bjw/papers.html

Lightcuts: A Scalable Approach to Illumination

Bruce Walter, Sebastian Fernandez,
Adam Arbree, Mike Donikian,
Kavita Bala, Donald Greenberg

Program of Computer Graphics, Cornell University

SIGGRAPH2005

Lightcuts

- Efficient, accurate complex illumination

Environment map lighting & indirect
Time 111s

Textured area lights & indirect
Time 98s

(640x480, Anti-aliased, Glossy materials)

Scalable

- Scalable solution for many point lights
  - Thousands to millions
  - Sub-linear cost

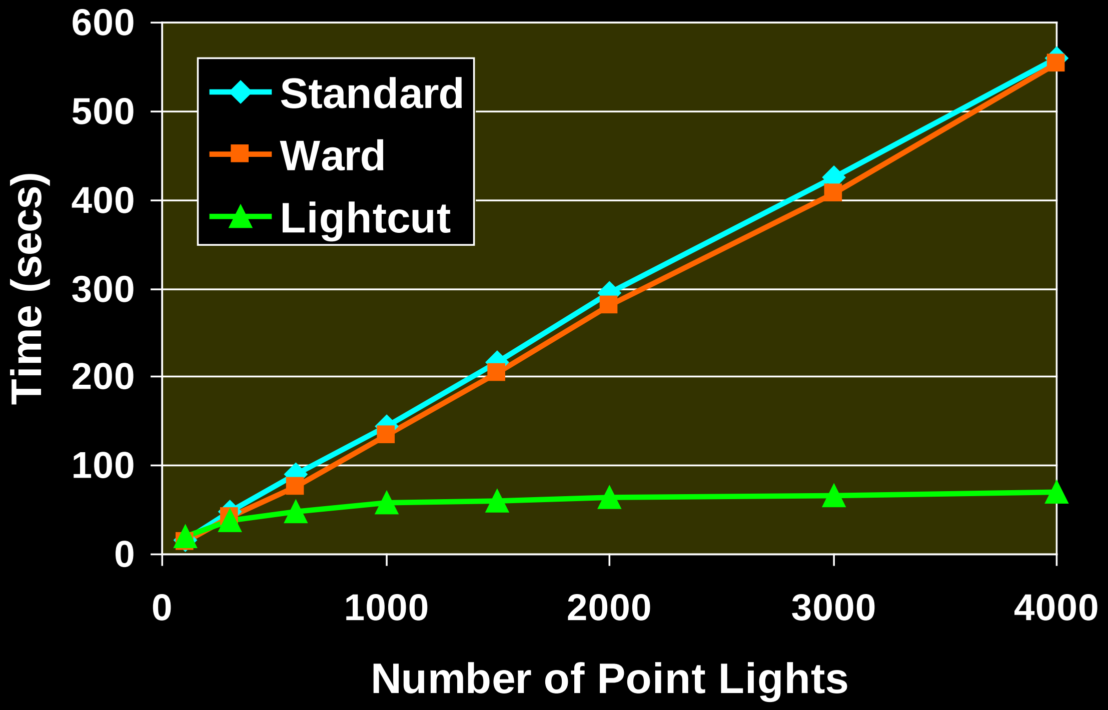


Tableau Scene

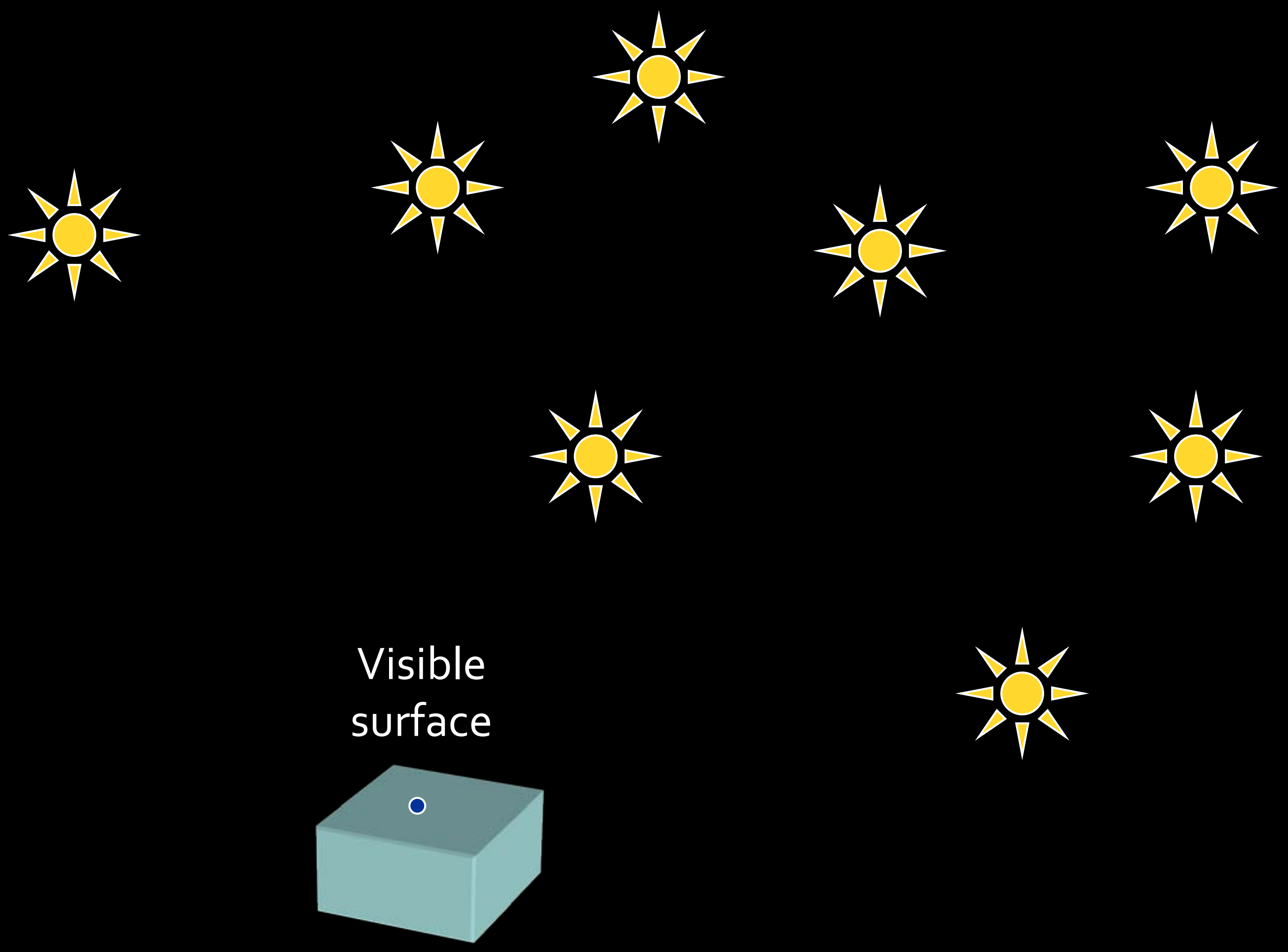
Complex Lighting

- Simulate complex illumination using point lights
  - Area lights
  - HDR environment ma
  - Sun & sky light
  - Indirect illumination

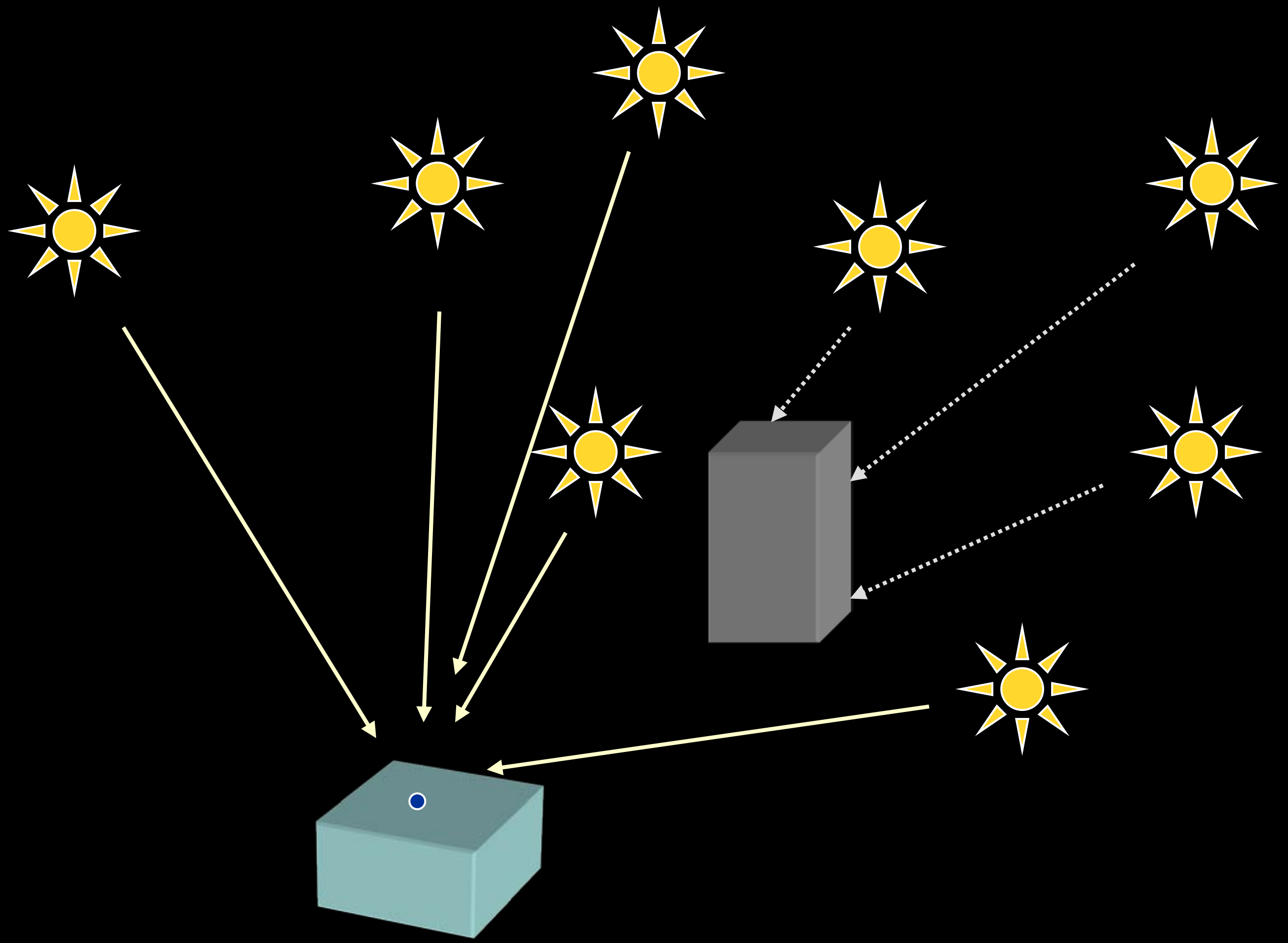
- Unifies illumination
  - Enables tradeoffs between components

Area lights + Sun/sky + Indirect

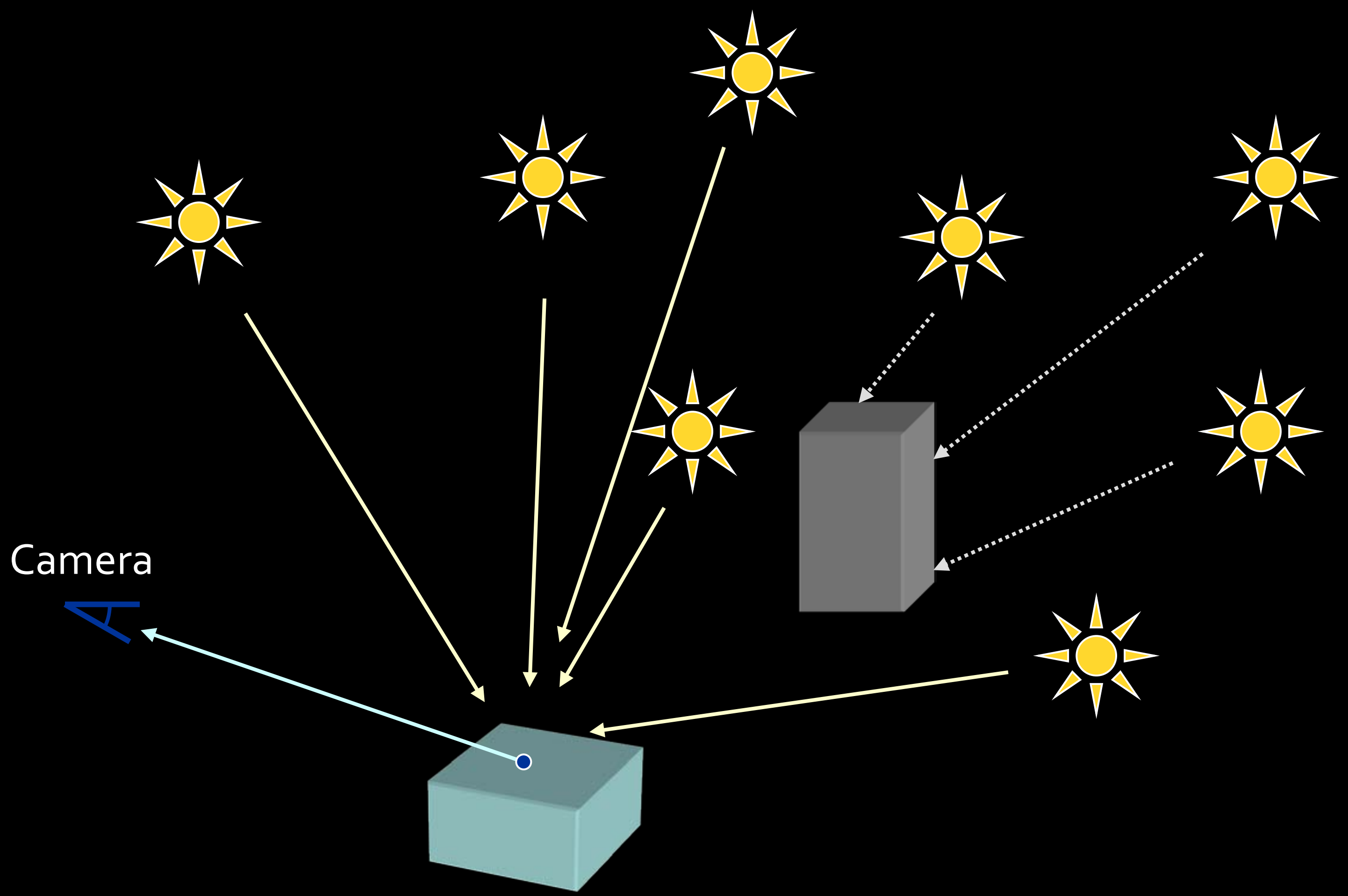
Lightcuts Problem



Lightcuts Problem

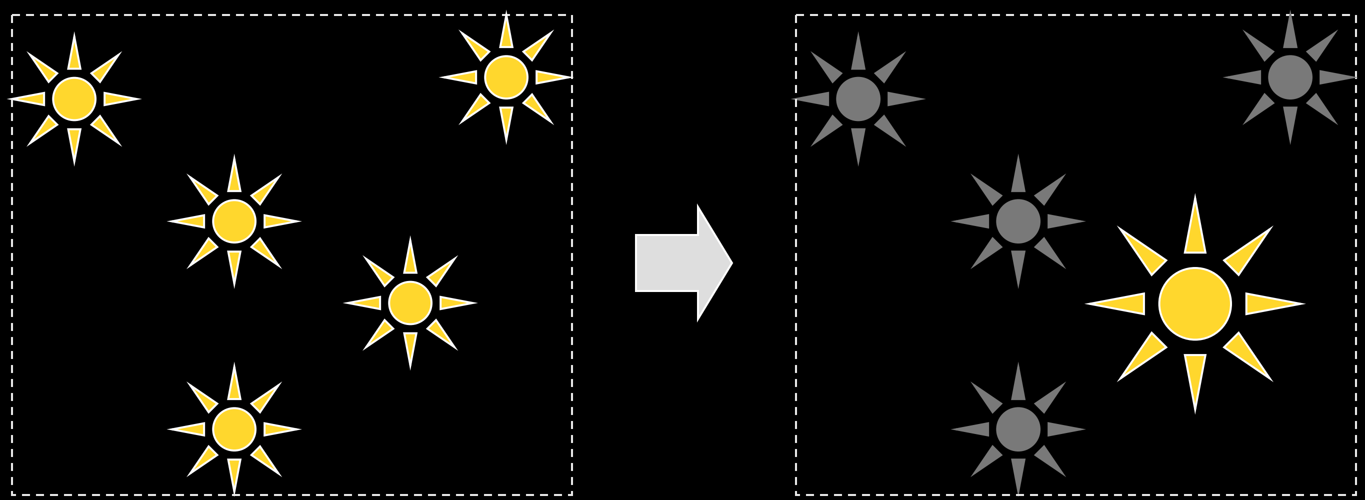


Lightcuts Problem



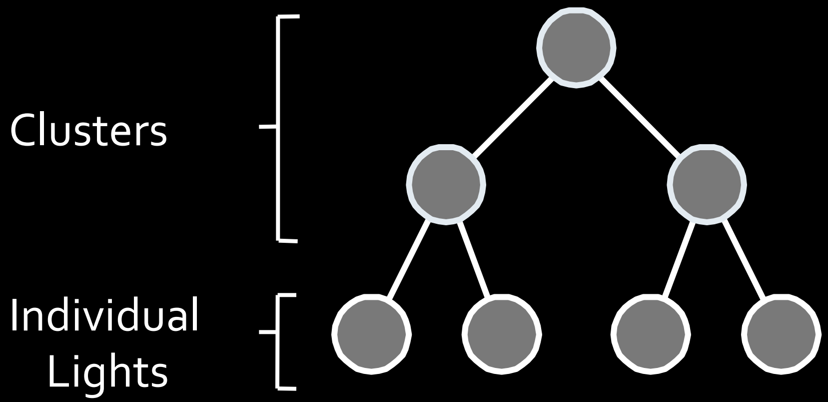
Key Concepts

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



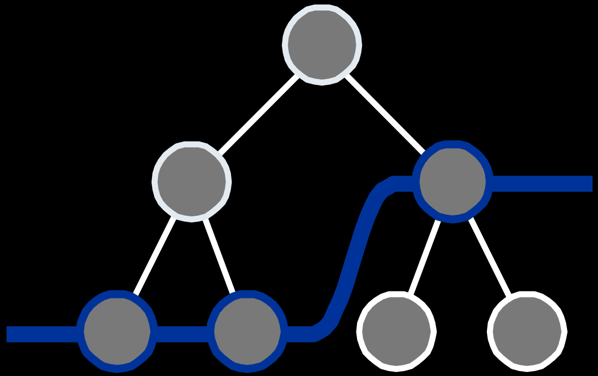
Key Concepts

- Light Cluster
- Light Tree
  - Binary tree of lights and clusters

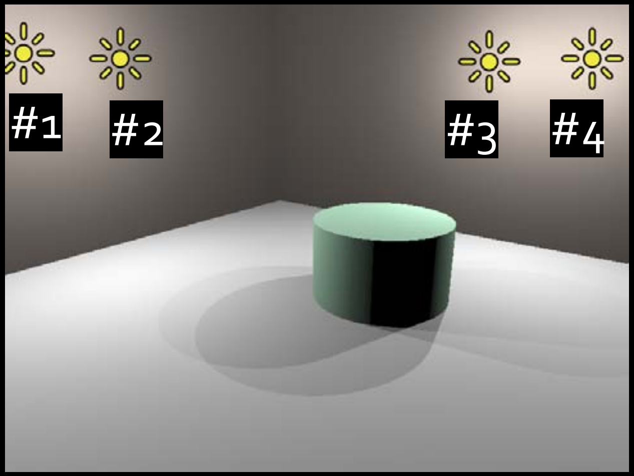


Key Concepts

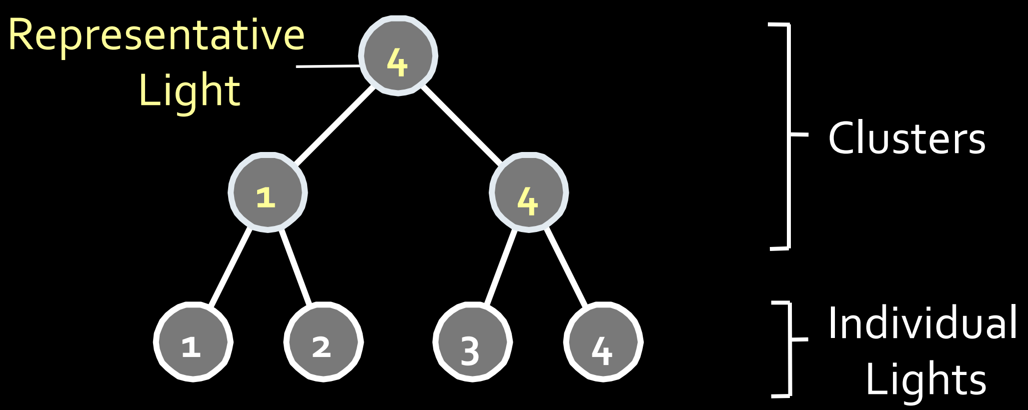
- Light Cluster
- Light Tree
- A Cut
  - A set of nodes that partitions the lights into clusters



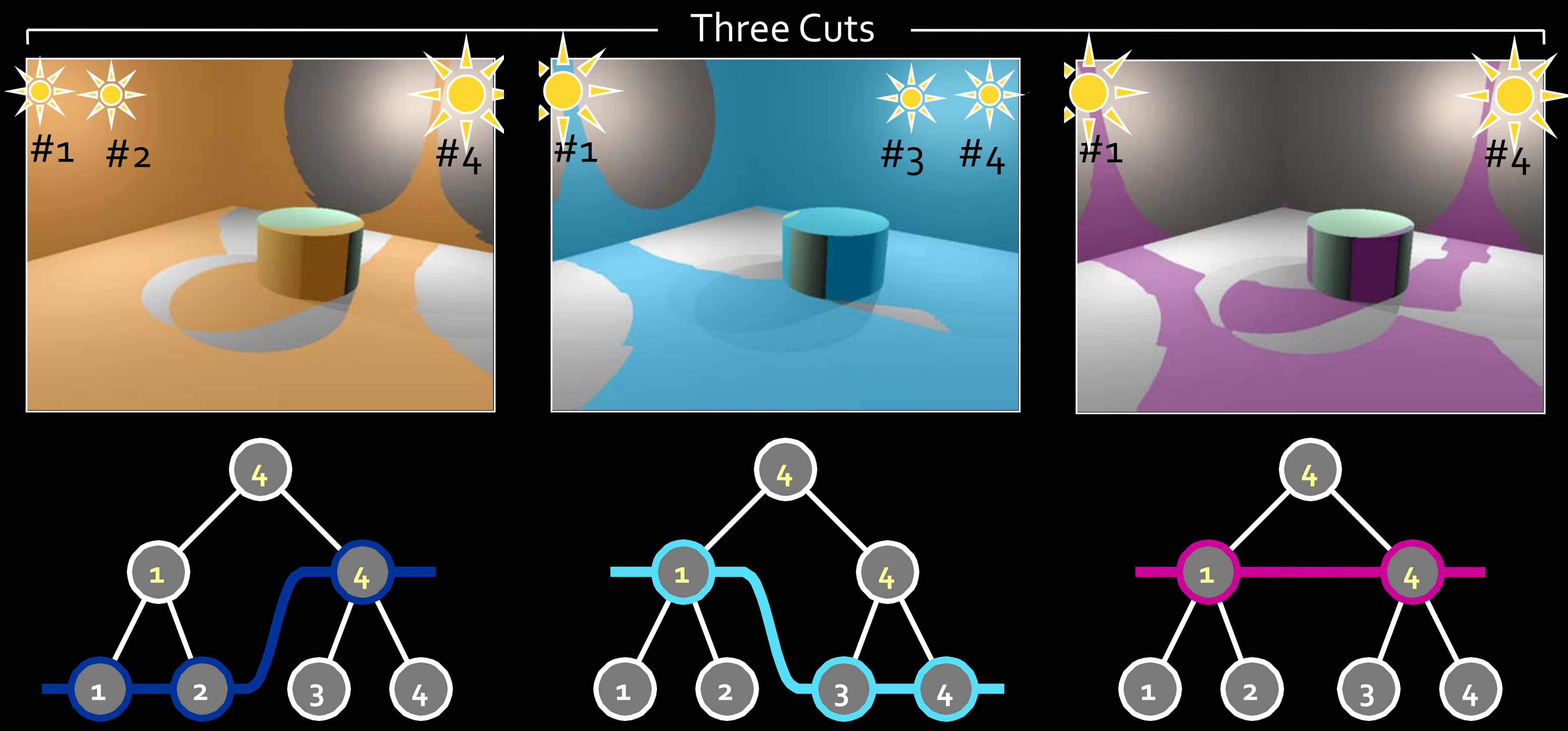
Simple Example



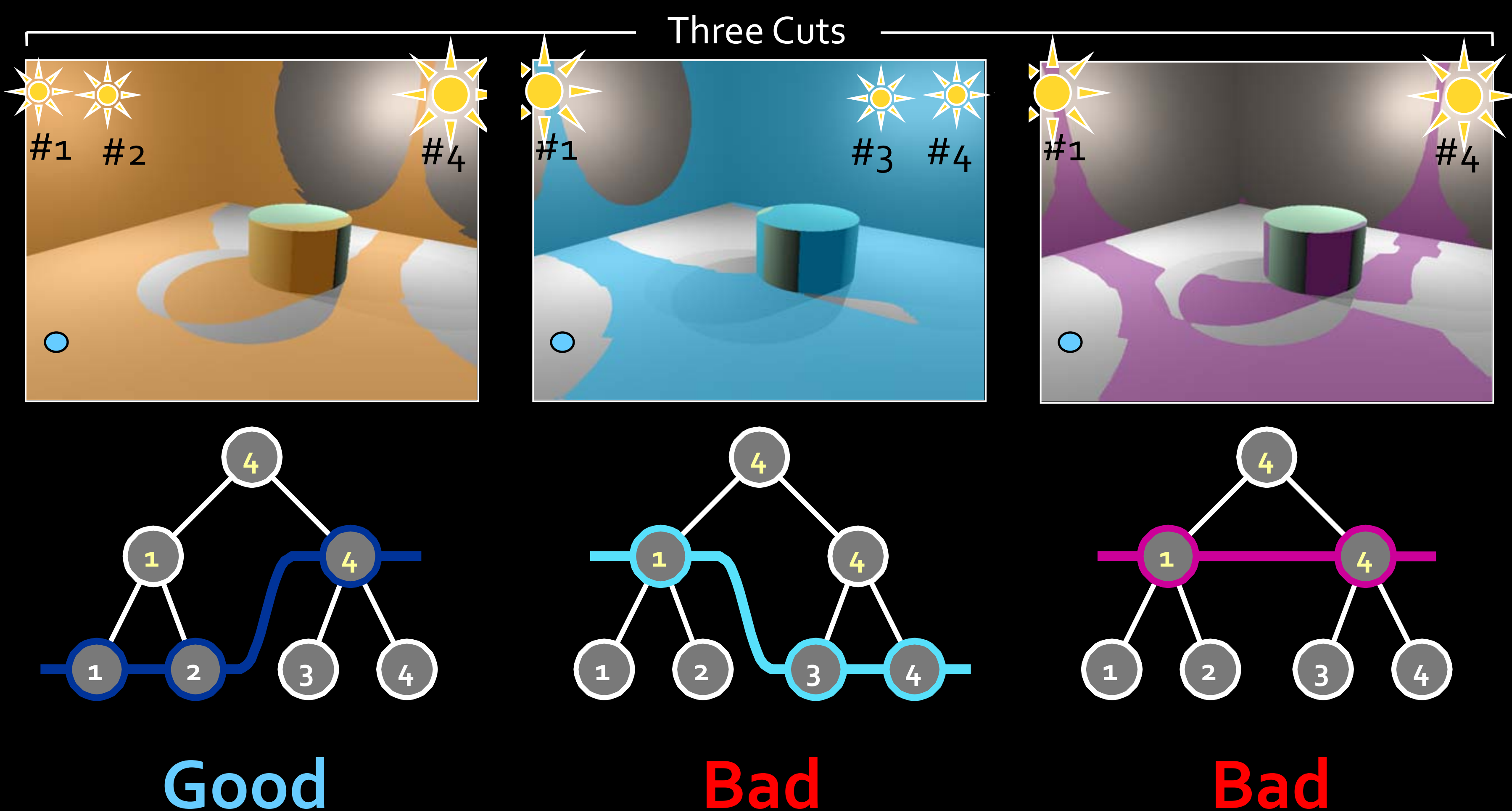
Light Tree



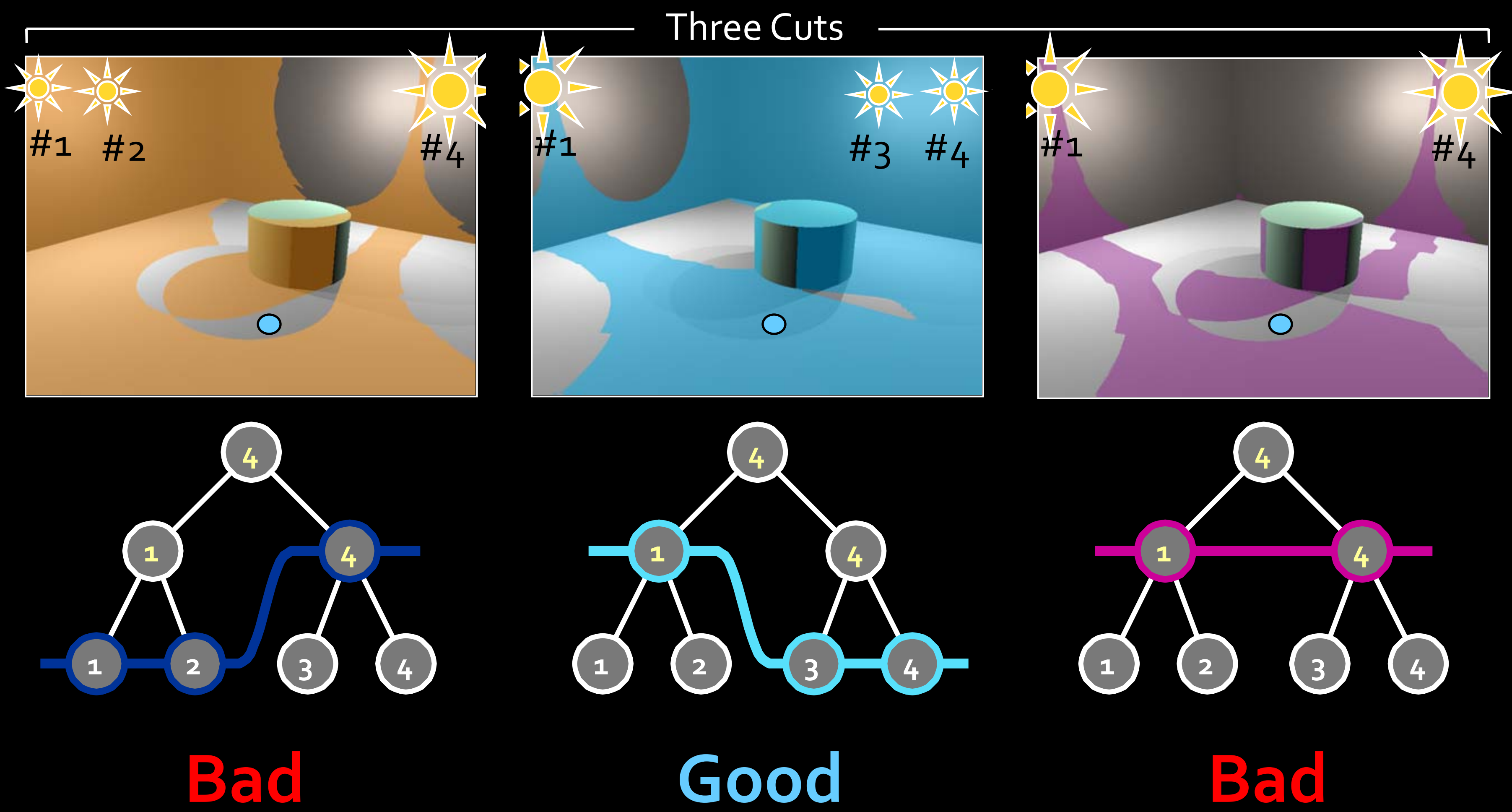
Three Example Cuts



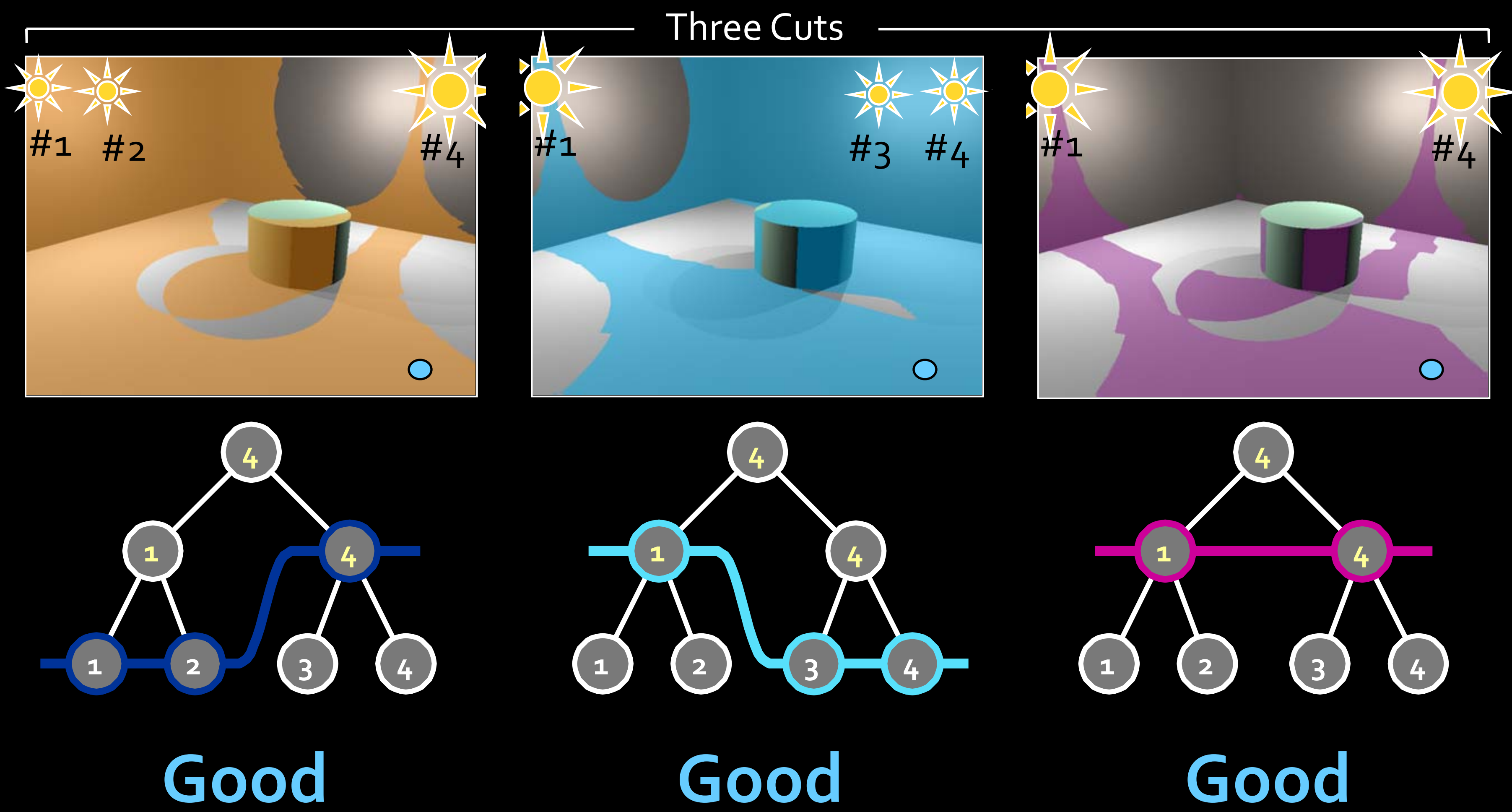
Three Example Cuts



Three Example Cuts



Three Example Cuts

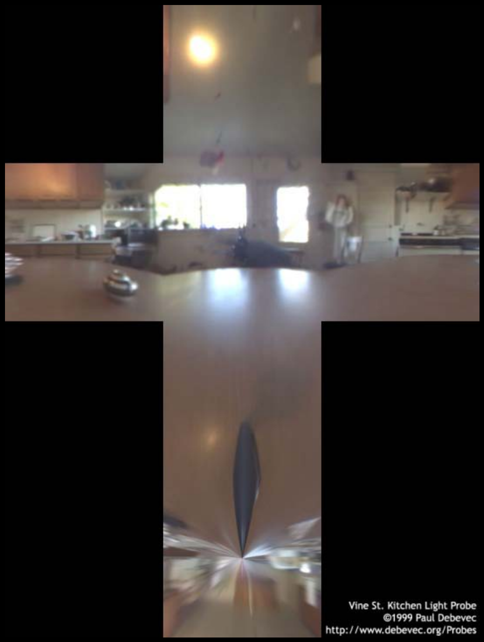


Algorithm Overview

- Pre-process
  - Convert illumination to point lights
  - Build light tree
- For each eye ray
  - Choose a cut to approximate the illumination

Convert Illumination

- HDR environment map
  - Apply captured light to scene
  - Convert to directional point lights using [Agarwal et al. 2003]



- Indirect Illumination
  - Convert indirect to direct illumination using Instant Radiosity [Keller 97]
    - Caveats: no caustics, clamping, etc.
  - More lights = more indirect detail

Algorithm Overview

- Pre-process
  - Convert illumination to point lights
  - Build light tree
- For each eye ray
  - Choose a cut to approximate the local illumination
    - Cost vs. accuracy
    - Avoid visible transition artifacts

Perceptual Metric

- Weber's Law
  - Contrast visibility threshold is fixed percentage of signal
  - Used 2% in our results
- Ensure each cluster's error < visibility threshold
  - Transitions will not be visible
  - Used to select cut

Illumination Equation

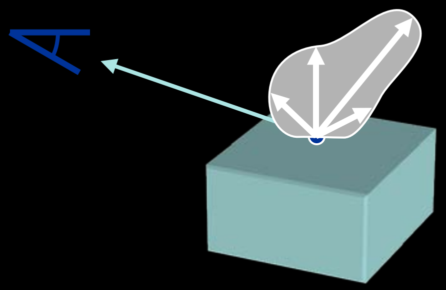
$$\text{result} = \sum_{\text{lights}} M_i \; G_i \; V_i \; I_i$$

Material term

Geometric term

Visibility term

Light intensity



Currently support diffuse, phong, and Ward

Illumination Equation

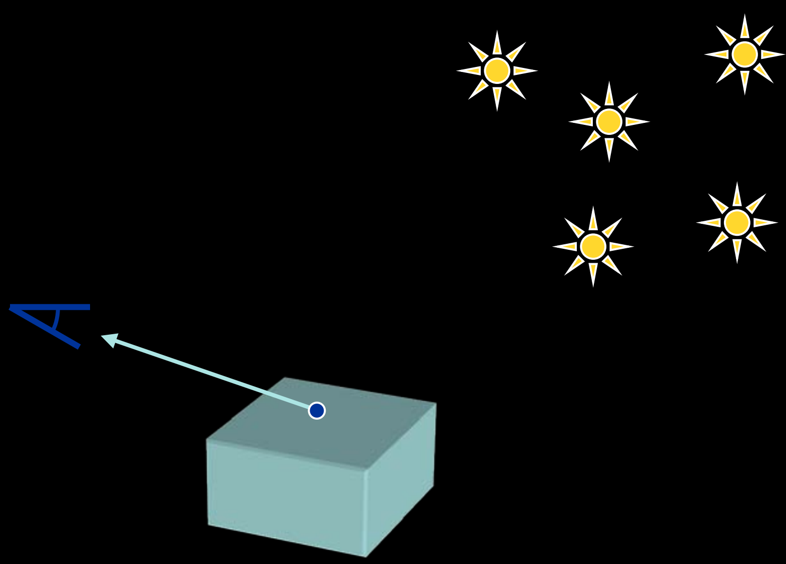
$$\text{result} = \sum_{\text{lights}} M_i \; G_i \; V_i \; I_i$$

Material term

Geometric term

Visibility term

Light intensity



Illumination Equation

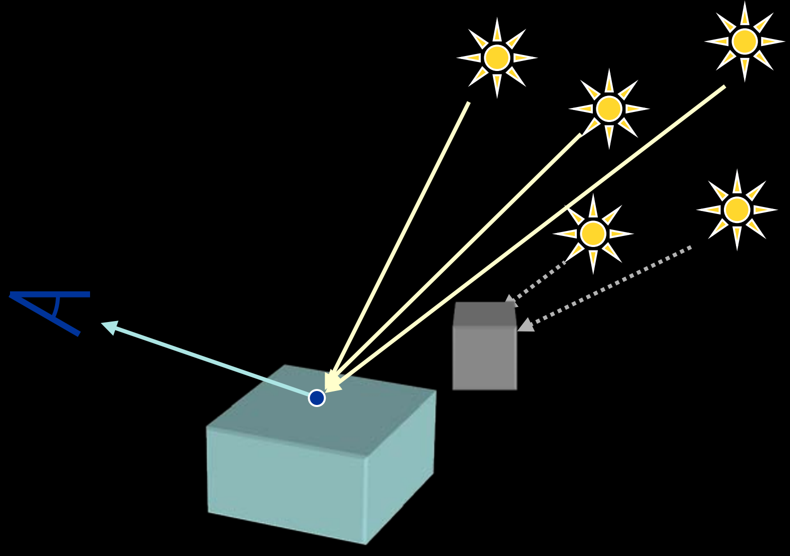
$$\text{result} = \sum_{\text{lights}} M_i \; G_i \; V_i \; I_i$$

Material term

Geometric term

Visibility term

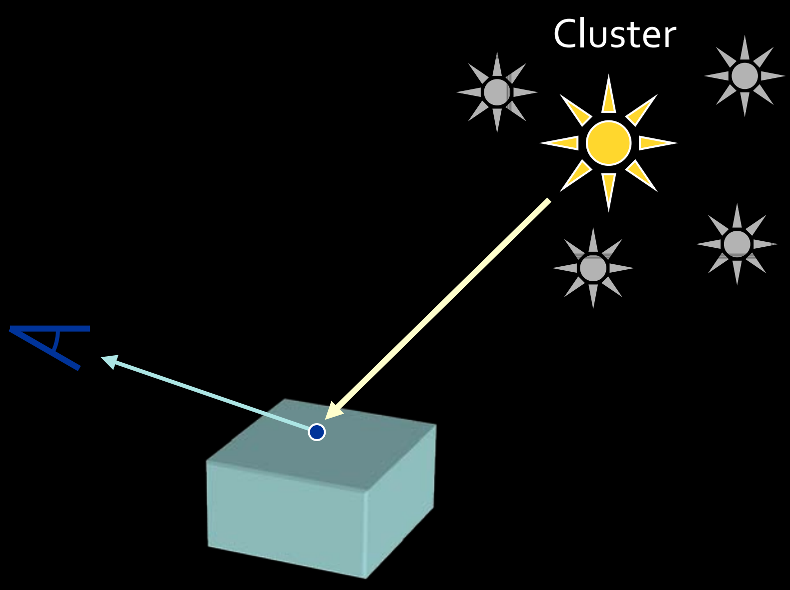
Light intensity



Cluster Approximation

$$\text{result} \approx M_j \; G_j \; V_j \sum_{\text{lights}} I_i$$

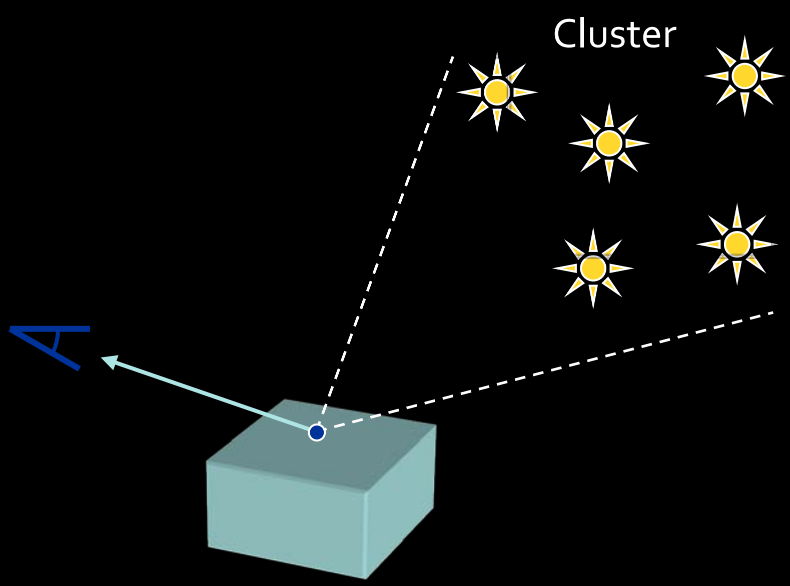
j is the representative light



Cluster Error Bound

$$\text{error} \leq M_{\text{ub}}\, G_{\text{ub}}\, V_{\text{ub}} \sum_{\text{lights}} I_{\text{i}}$$

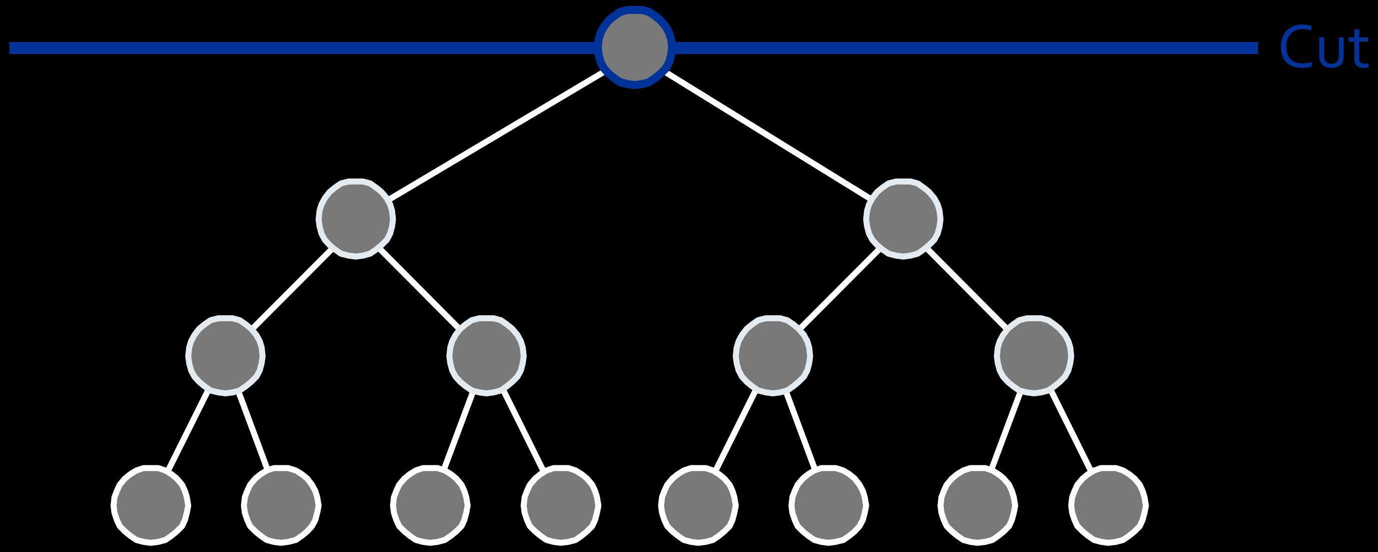
- Bound each term
  - Visibility <= 1 (trivial)
  - Intensity is known
  - Bound material and geometric terms using cluster bounding volume



ub == upper bound

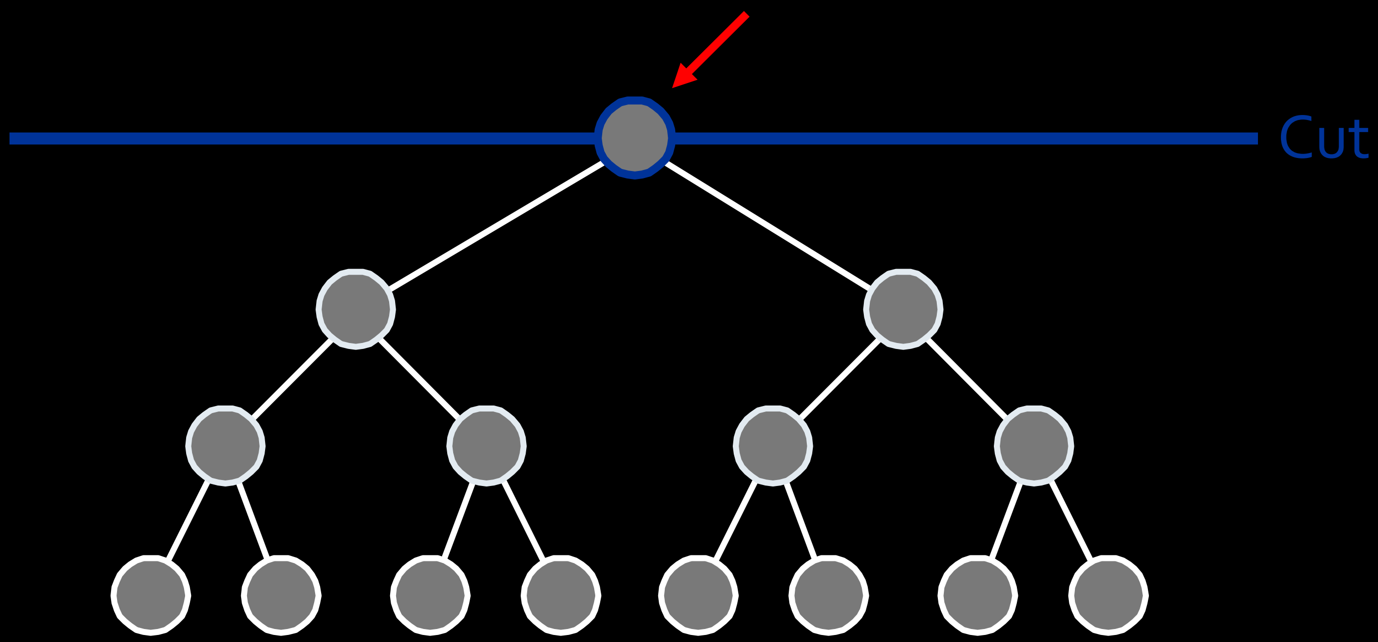
Cut Selection Algorithm

- Start with coarse cut (eg, root node)



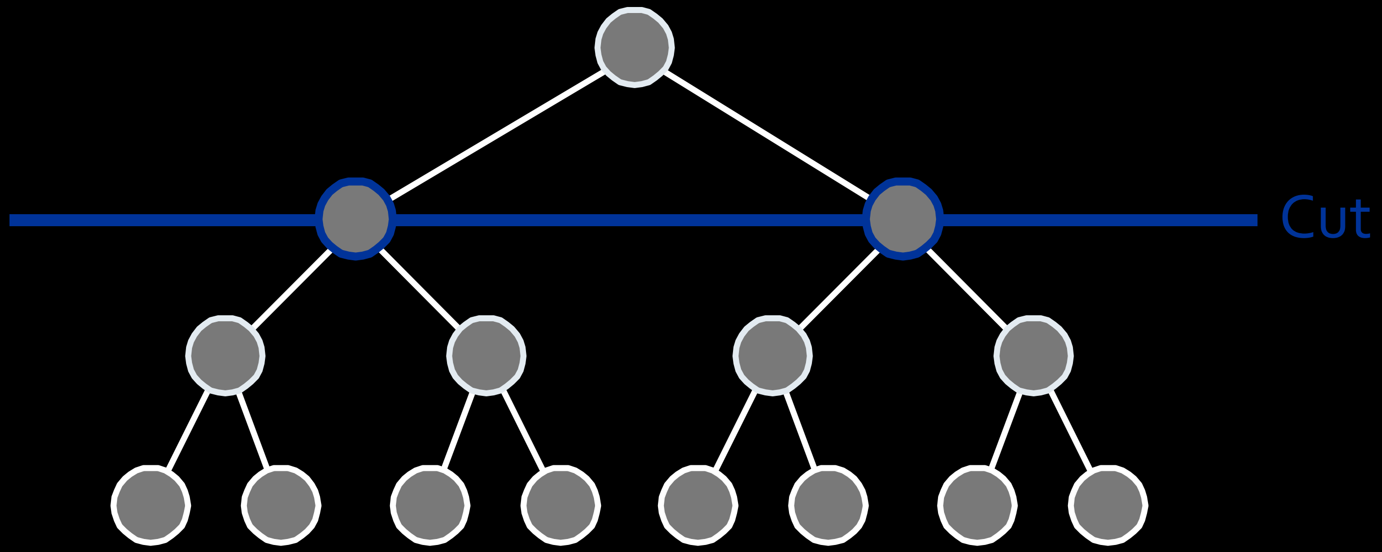
Cut Selection Algorithm

- Select cluster with largest error bound

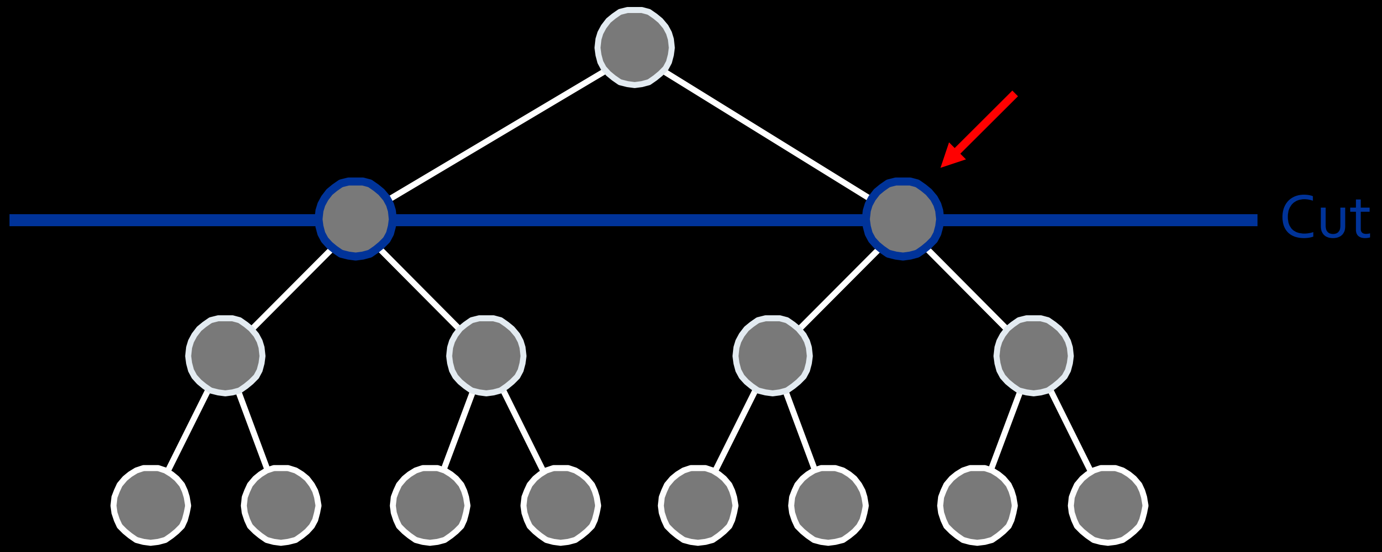


Cut Selection Algorithm

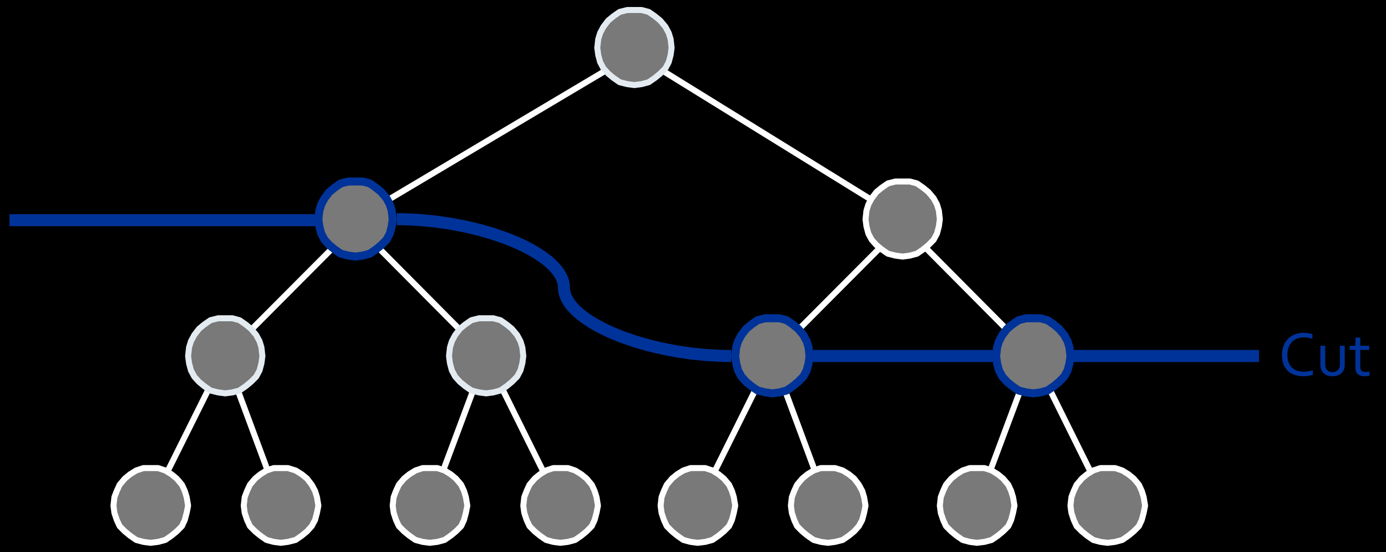
- Refine if error bound > 2% of total



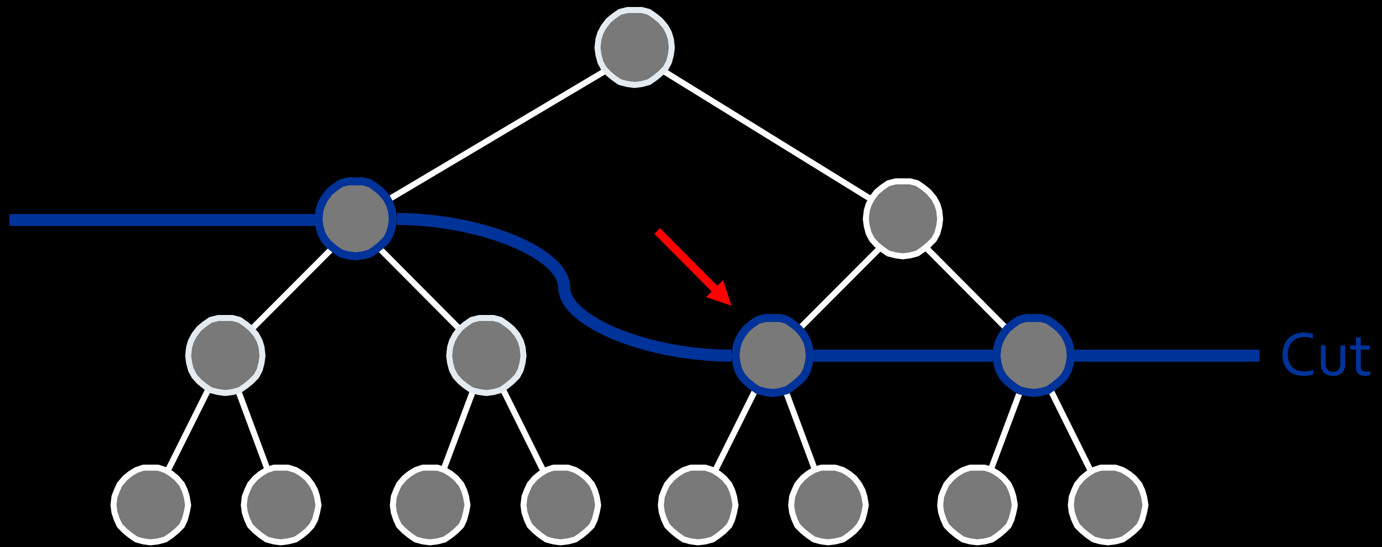
Cut Selection Algorithm



Cut Selection Algorithm

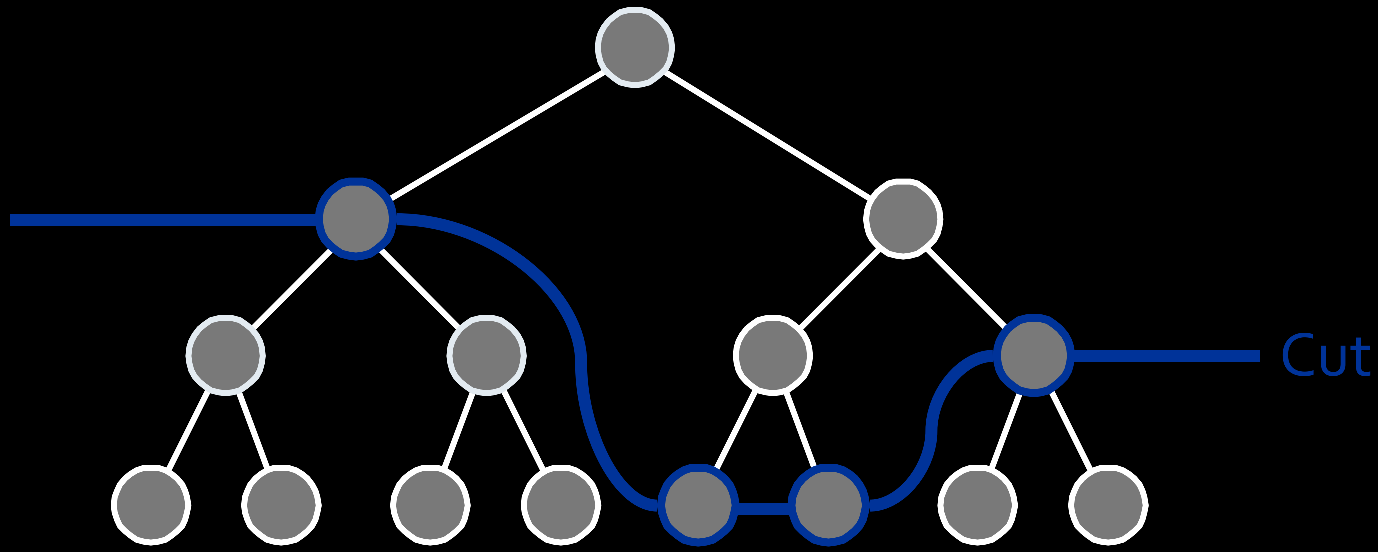


Cut Selection Algorithm



Cut Selection Algorithm

- Repeat until cut obeys 2% threshold



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

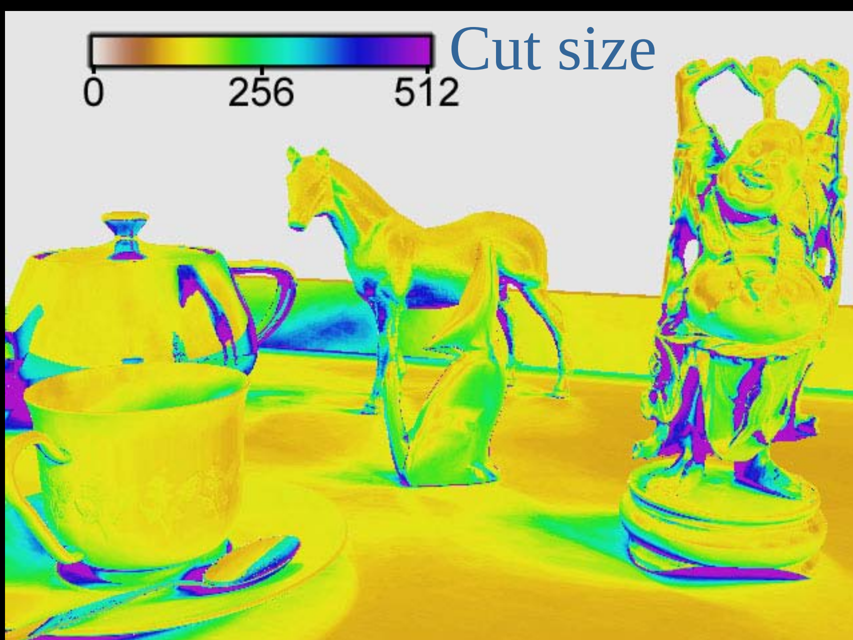
4 608 Lights
(Area lights only)

Avg. 259 shadow rays / pixel

Lightcuts 290s

59 672 Lights
(Area + Sun/sky + Indirect)

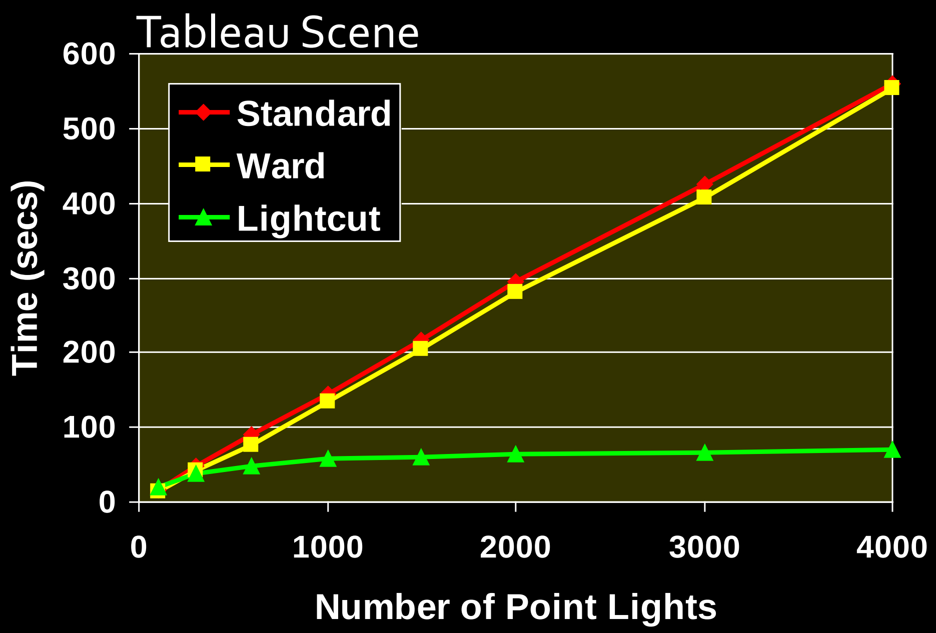
Avg. 478 shadow rays / pixel
(only 54 to area lights)

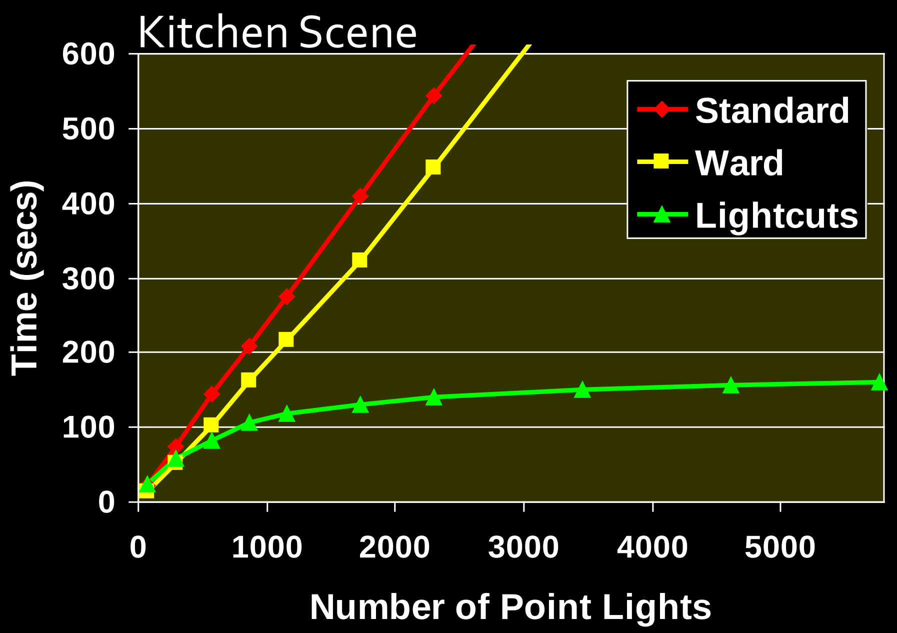


Error x 16

Scalable

- Scalable solution for many point lights
  - Thousands to millions
  - Sub-linear cost





Lightcuts

Bigscreen Model | Cut Size (False Color)

- Problem: Large cuts in dark areas

Lightcuts Recap

- Unified illumination handling
- Scalable solution for many lights
  - Locally adaptive representation (the cut)
- Analytic cluster error bounds
  - Most important lights always sampled
- Perceptual visibility metric
- Problems
  - Large cuts in dark regions
  - Need tight upper bounds for BRDFs

Multidimensional Lightcuts

Bruce Walter
Adam Arbree
Kavita Bala
Donald P. Greenberg

Program of Computer Graphics, Cornell University

Problem

- Simulate complex, expensive phenomena
  - Complex illumination
  - Anti-aliasing
  - Motion blur
  - Participating media
  - Depth of field

$$\text{Pixel} = \int_{\text{Time}} \int_{\text{Pixel Area}} \int_{\text{Lights}} L(\mathbf{x}, \omega) \ldots$$

Problem

- Simulate complex, expensive phenomena
  - Complex illumination
  - Anti-aliasing
  - Motion blur
  - Participating media
  - Depth of field

$$\text{Pixel} = \underset{\text{Volume}}{\int} \underset{\text{Time}}{\int} \underset{\substack{\text{Pixel}\\\text{Area}}}{\int} \underset{\text{Lights}}{\int} L(\mathbf{x},\omega)\ldots$$

Problem

- Simulate complex, expensive phenomena
  - Complex illumination
  - Anti-aliasing
  - Motion blur
  - Participating media
  - Depth of field

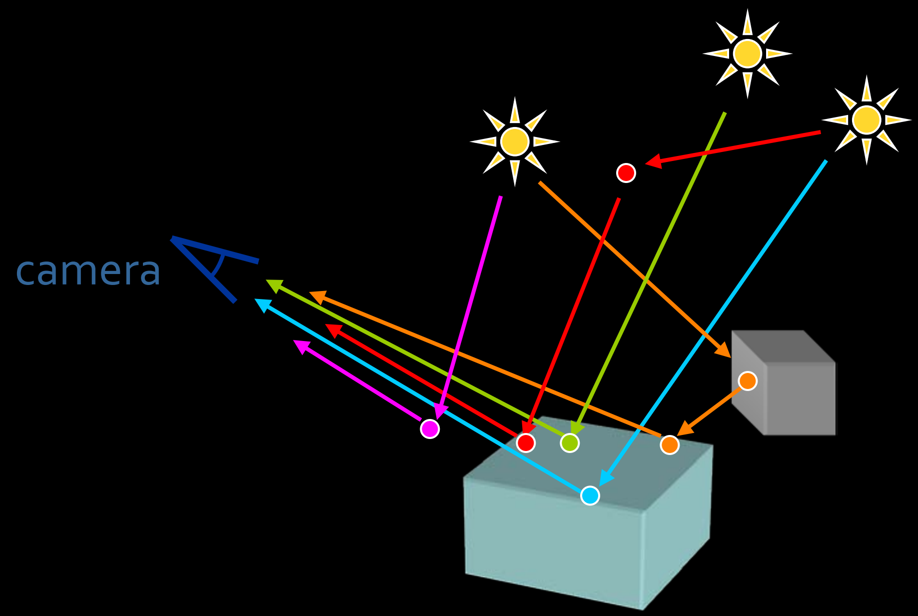
$$\text{Pixel} = \int_{\text{Aperture}} \int_{\text{Volume}} \int_{\text{Time}} \int_{\text{Pixel Area}} \int_{\text{Lights}} L(\mathbf{x},\omega)\ldots$$

Problem

- Complex integrals over multiple dimensions

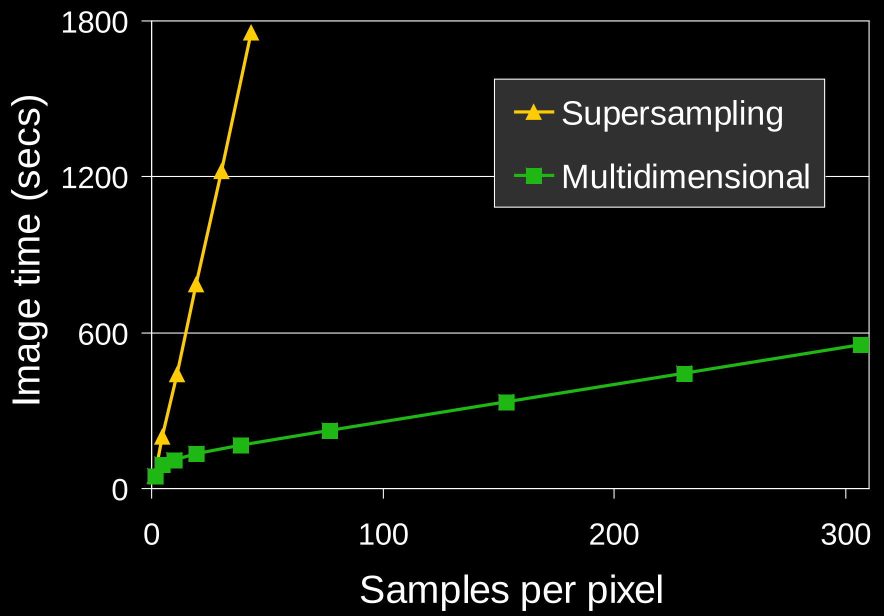
$$\text{Pixel} = \underset{\text{Aperture}}{\int} \; \underset{\text{Volume}}{\int} \; \underset{\text{Time}}{\int} \; \underset{\substack{\text{Pixel}\\\text{Area}}}{\int} \; \underset{\text{Lights}}{\int} L(\mathbf{x},\omega)\ldots$$

  - Requires many samples



Multidimensional Lightcuts

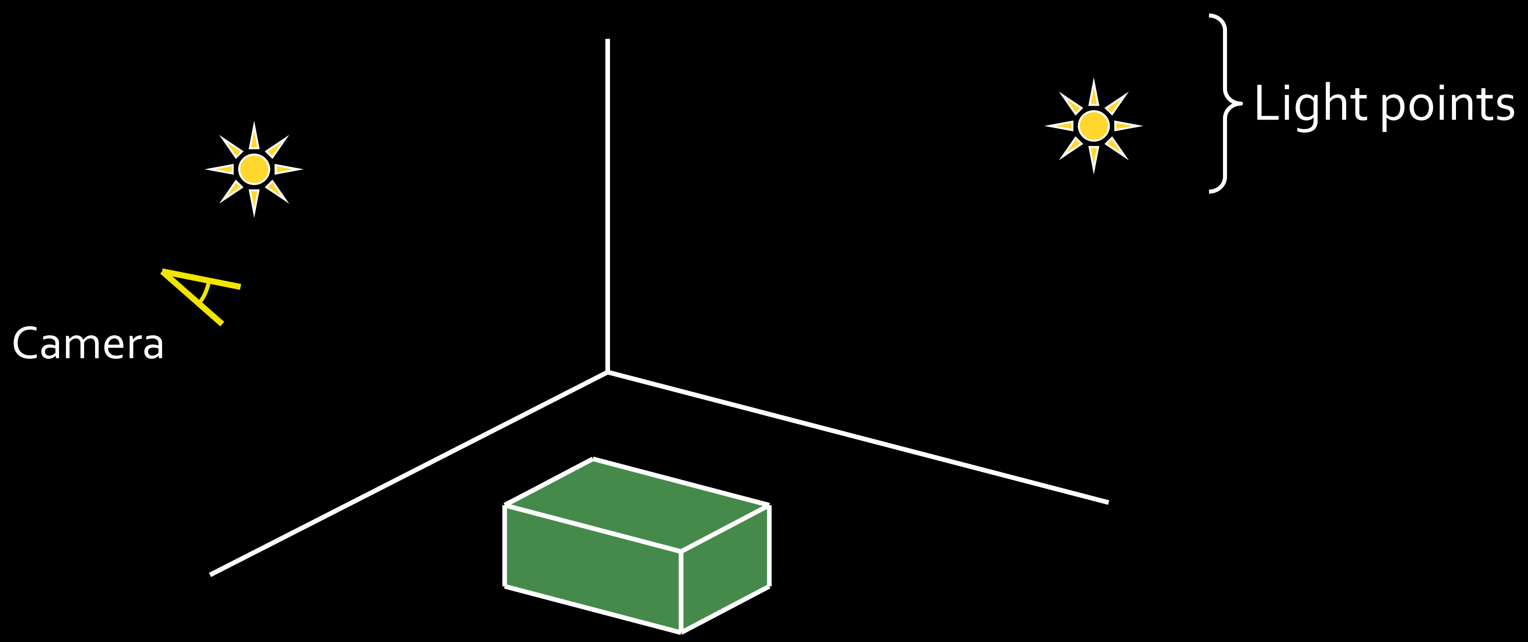
- Solves all integrals simultaneously
- Accurate
- Scalable




Direct only (relative cost 1x)
Direct+Indirect (1.3x)
Direct+Indirect+Volume (1.8x)
Direct+Indirect+Volume+Motion (2.2x)

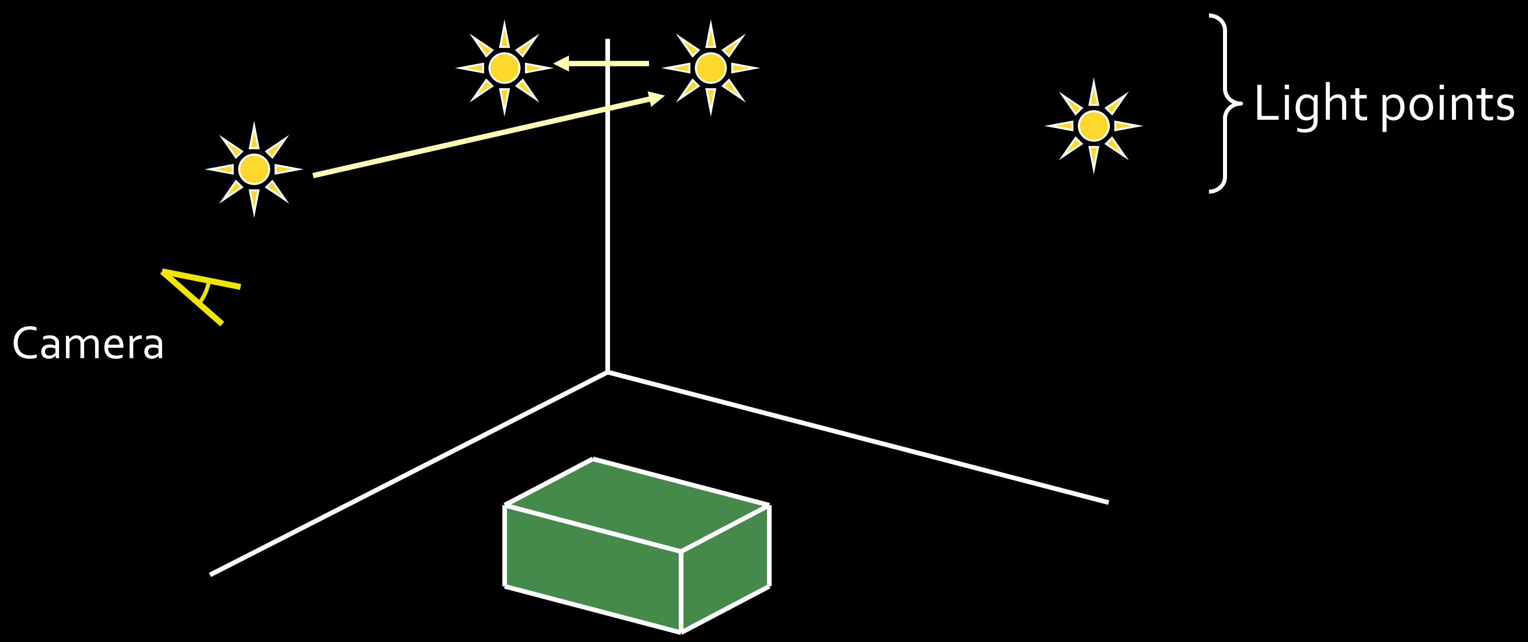
Point Sets

- Discretize full integral into 2 point sets
  - Light points ($\mathbf{L}$)
  - Gather points ($\mathbf{G}$)



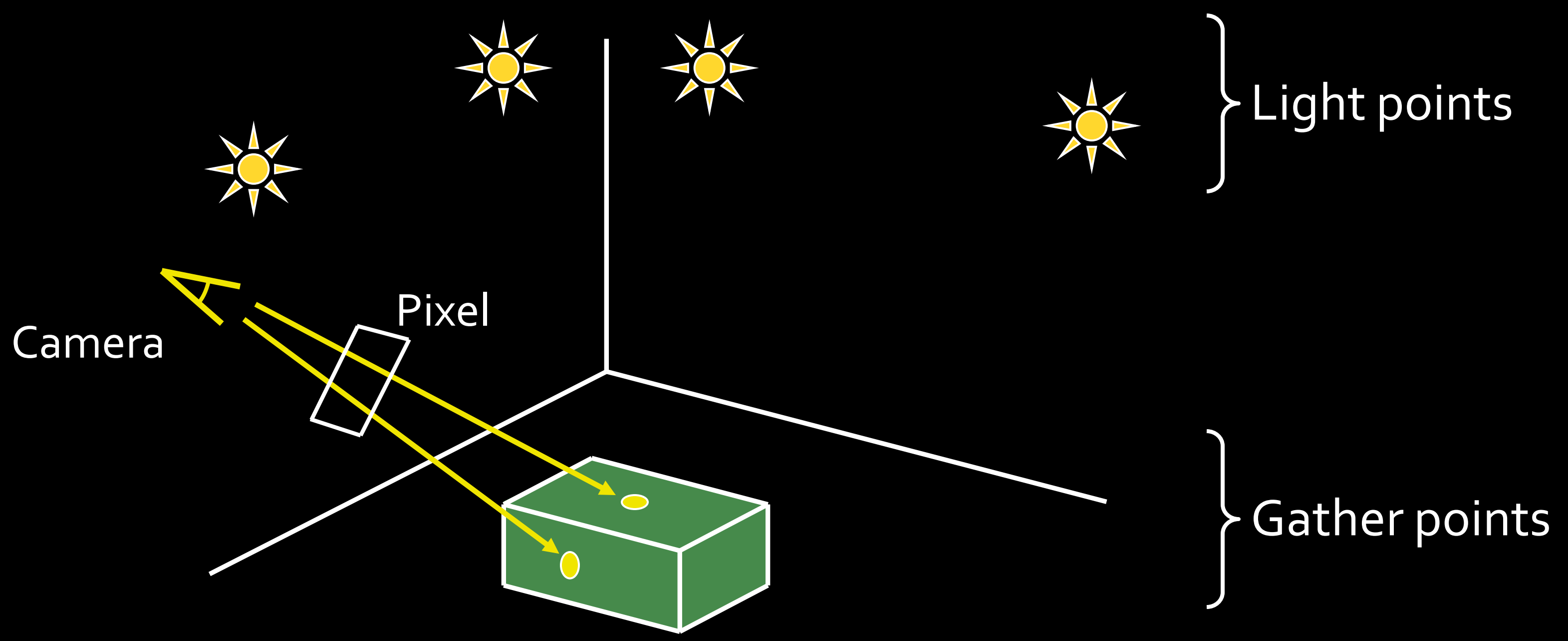
Point Sets

- Discretize full integral into 2 point sets
  - Light points ($\mathbf{L}$)
  - Gather points ($\mathbf{G}$)



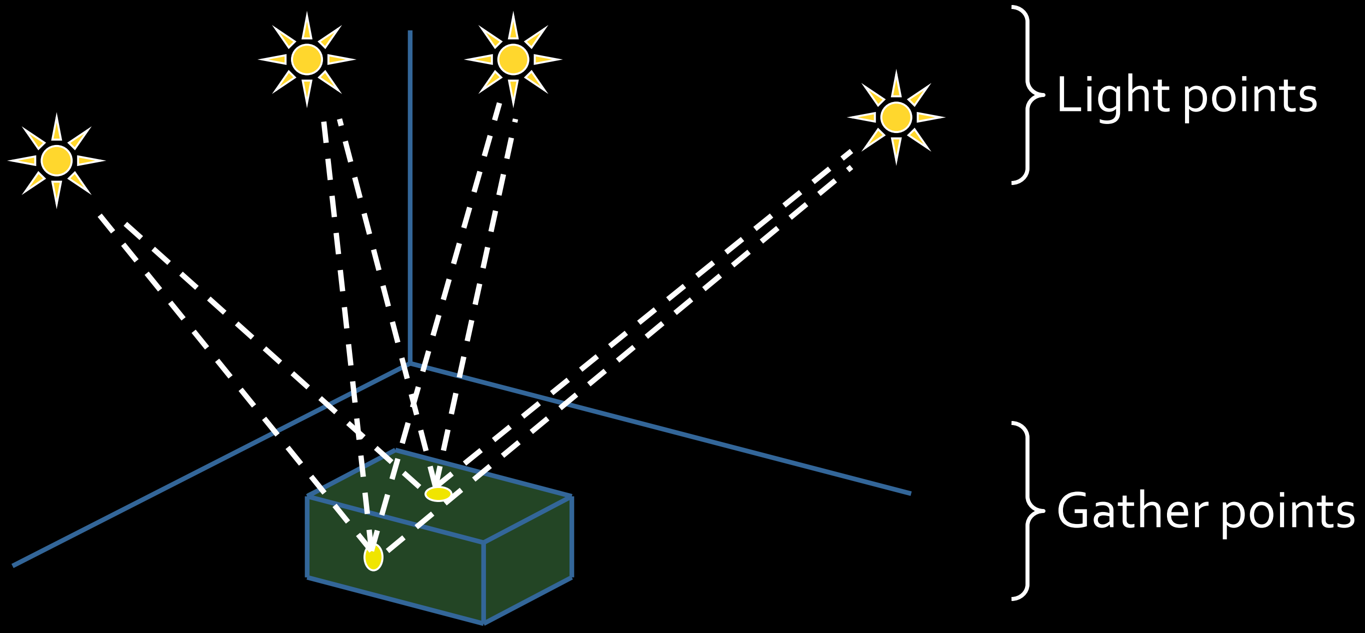
Point Sets

- Discretize full integral into 2 point sets
  - Light points ($\mathbf{L}$)
  - Gather points ($\mathbf{G}$)



Point Sets

- Discretize full integral into 2 point sets
  - Light points (**L**)
  - Gather points (**G**)



Discrete Equation

- Sum over all pairs of gather and light points
  - Can be billions of pairs per pixel

$$\text{Pixel} = \sum_{(j,i) \in \mathbf{G} \times \mathbf{L}} S_j \, M_{ji} \, G_{ji} \, V_{ji} \, I_i$$

Gather strength

Material term

Geometric term

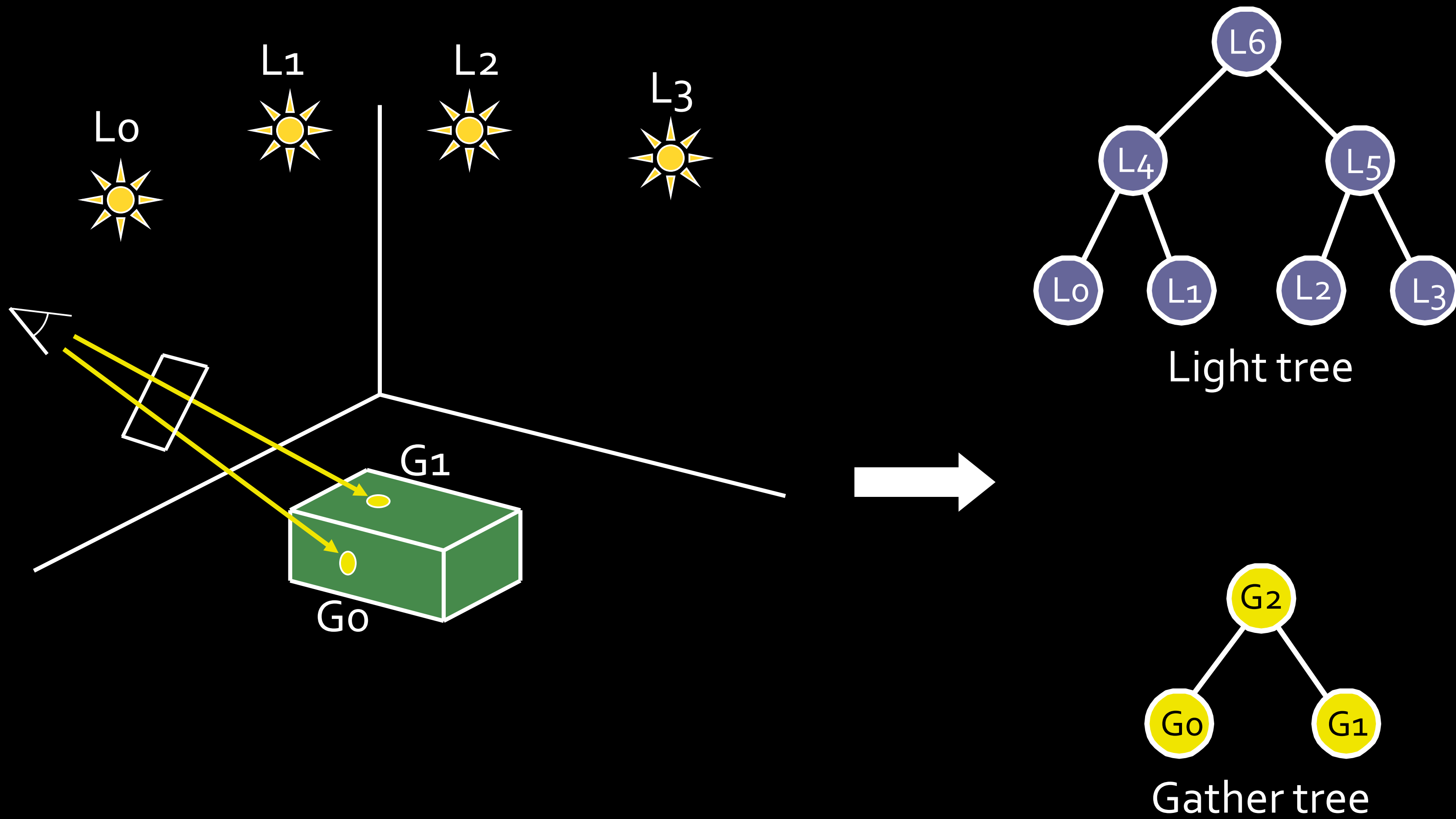
Visibility term

Light intensity

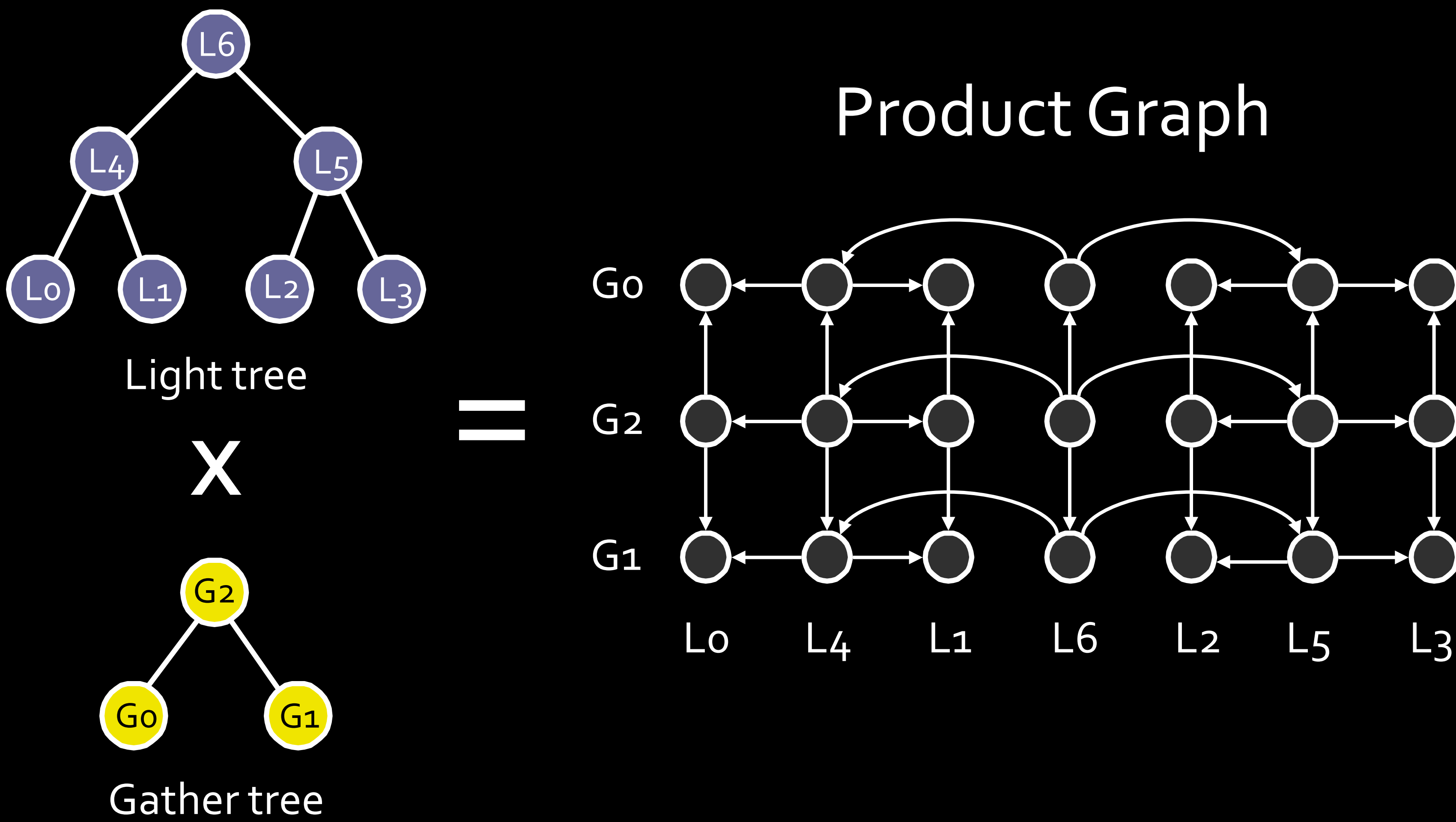
Product Graph

- Explicit hierarchy would be too expensive
  - Up to billions of pairs per pixel
- Use implicit hierarchy
  - Cartesian product of two trees (gather & light)

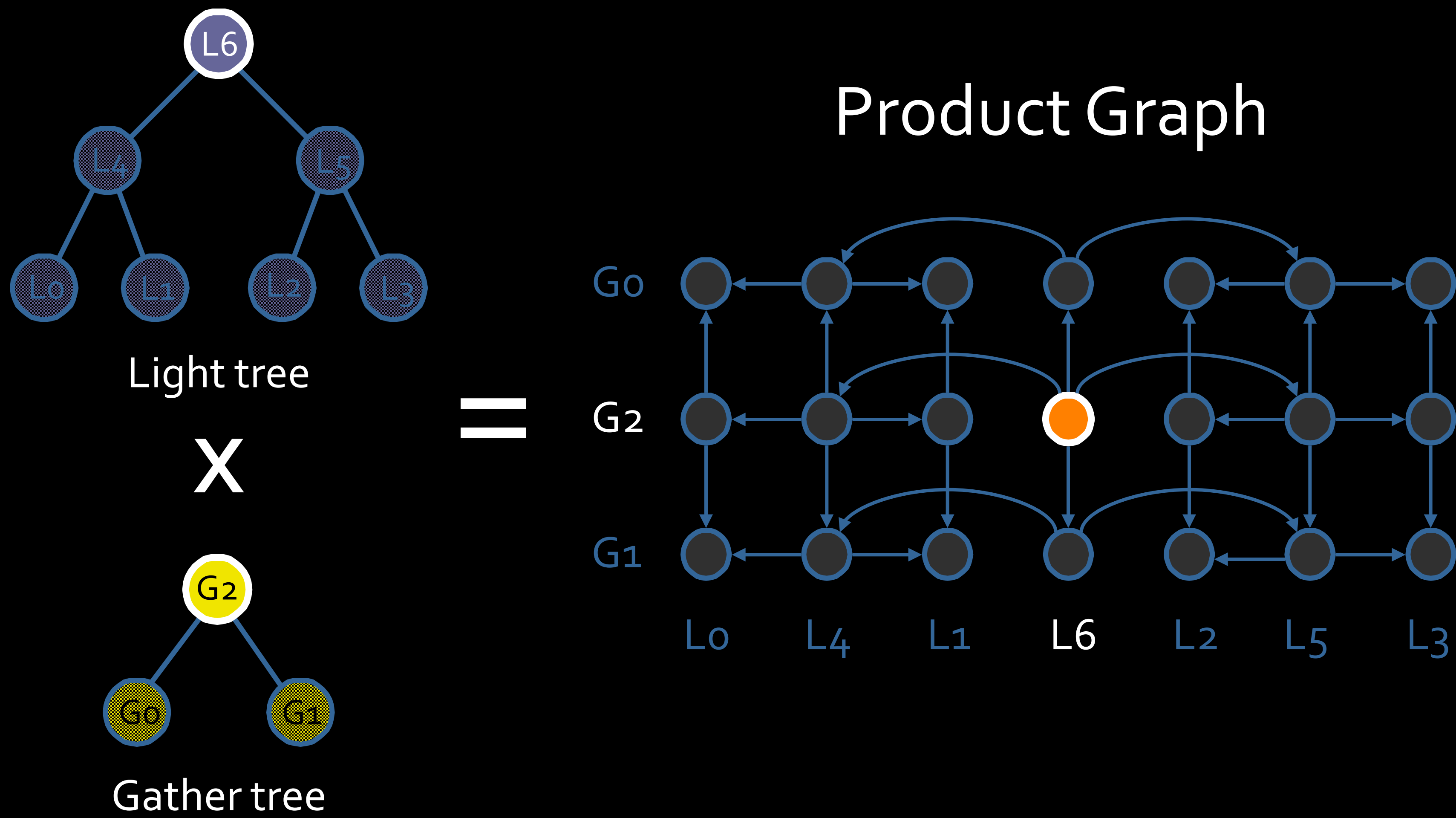
Product Graph



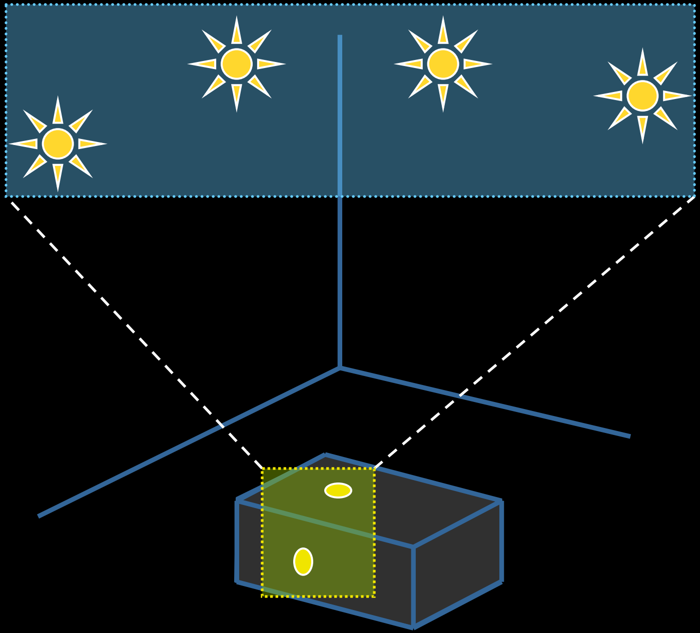
Product Graph

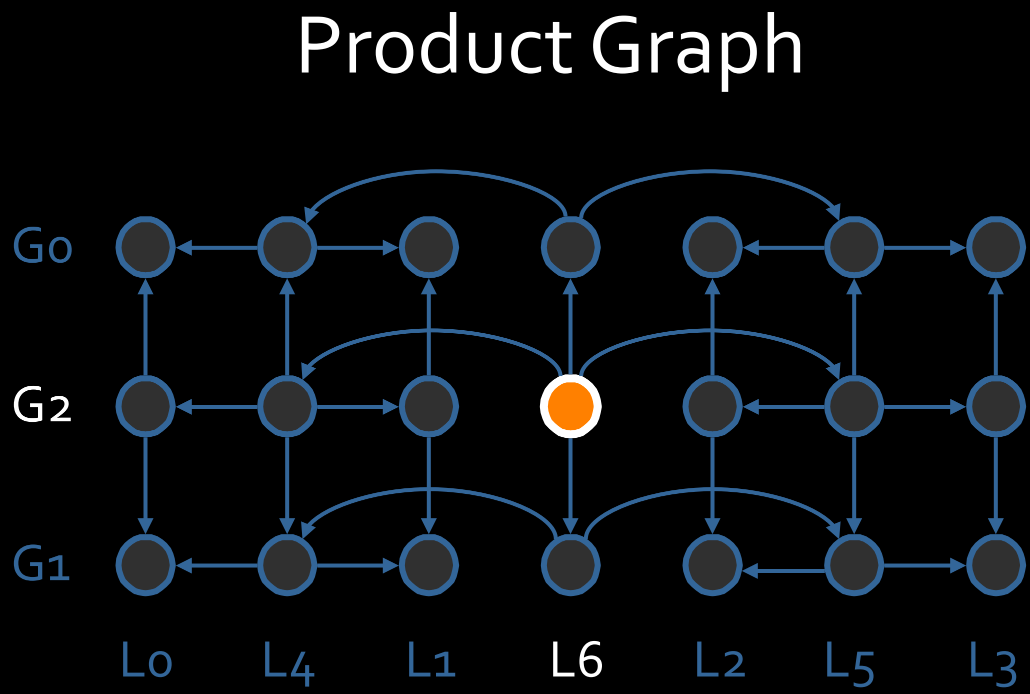


Product Graph

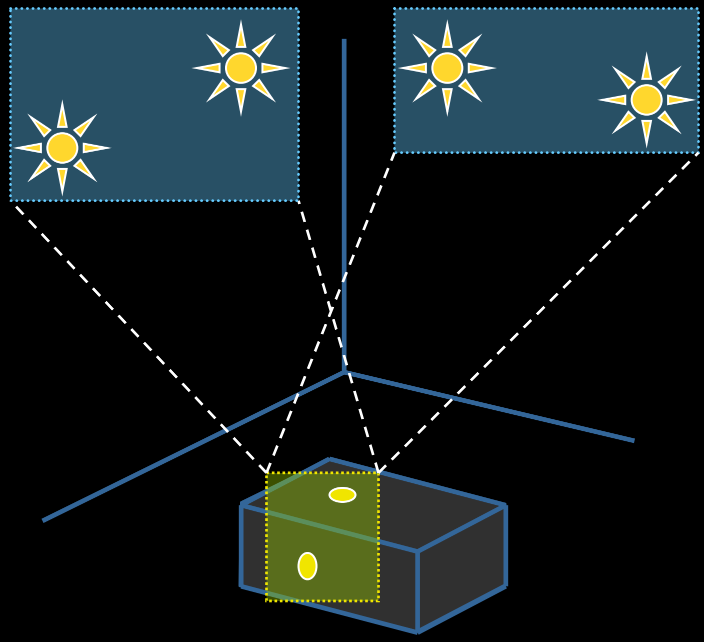


Product Graph

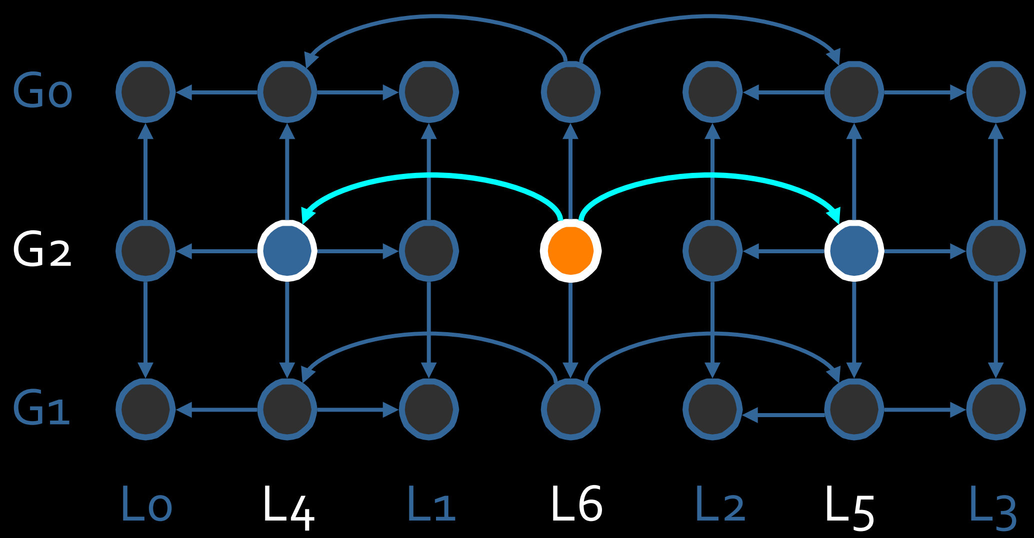




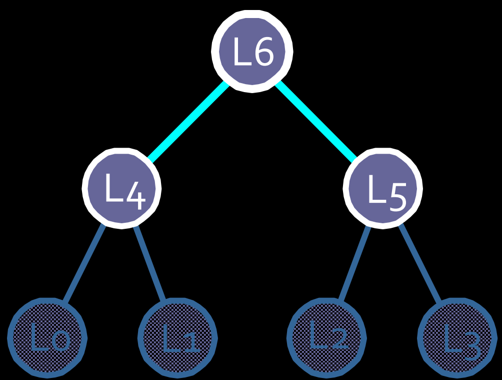
Product Graph



Product Graph

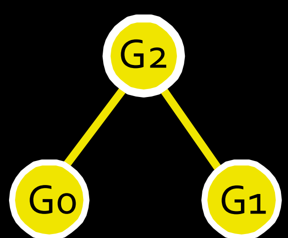


Product Graph



Light tree

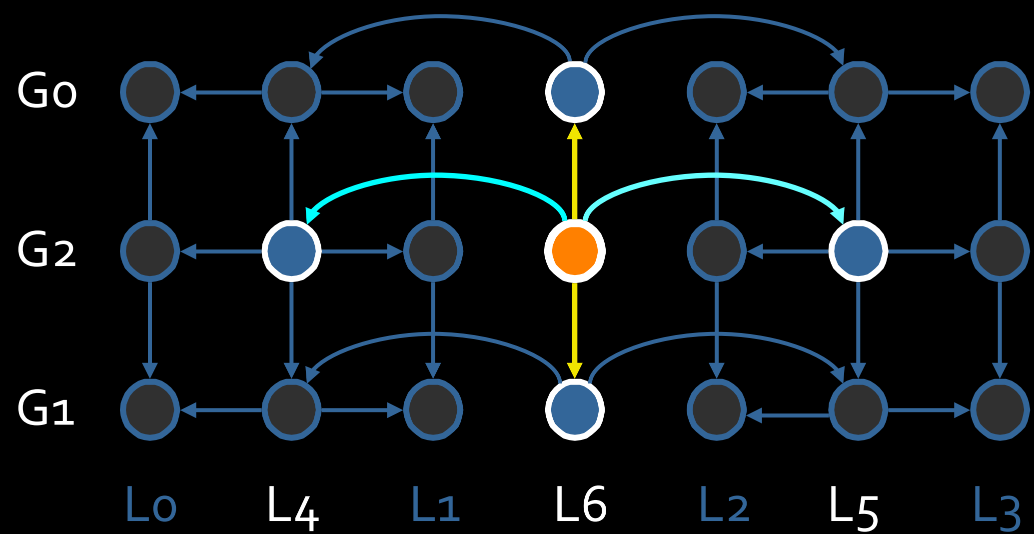
x



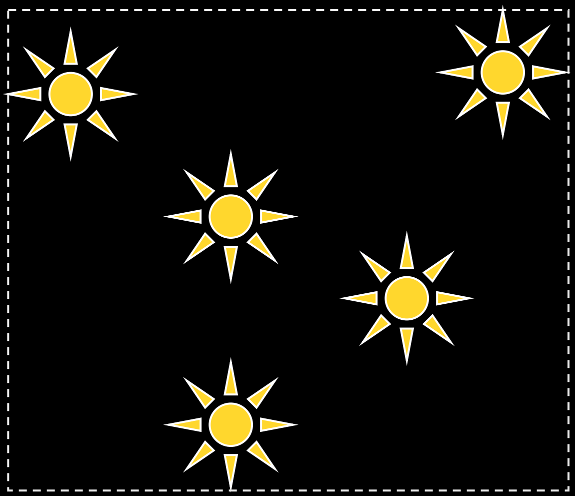
Gather tree

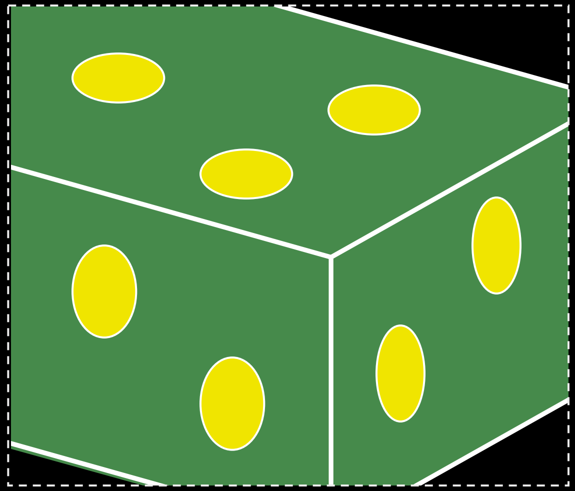
=

Product Graph

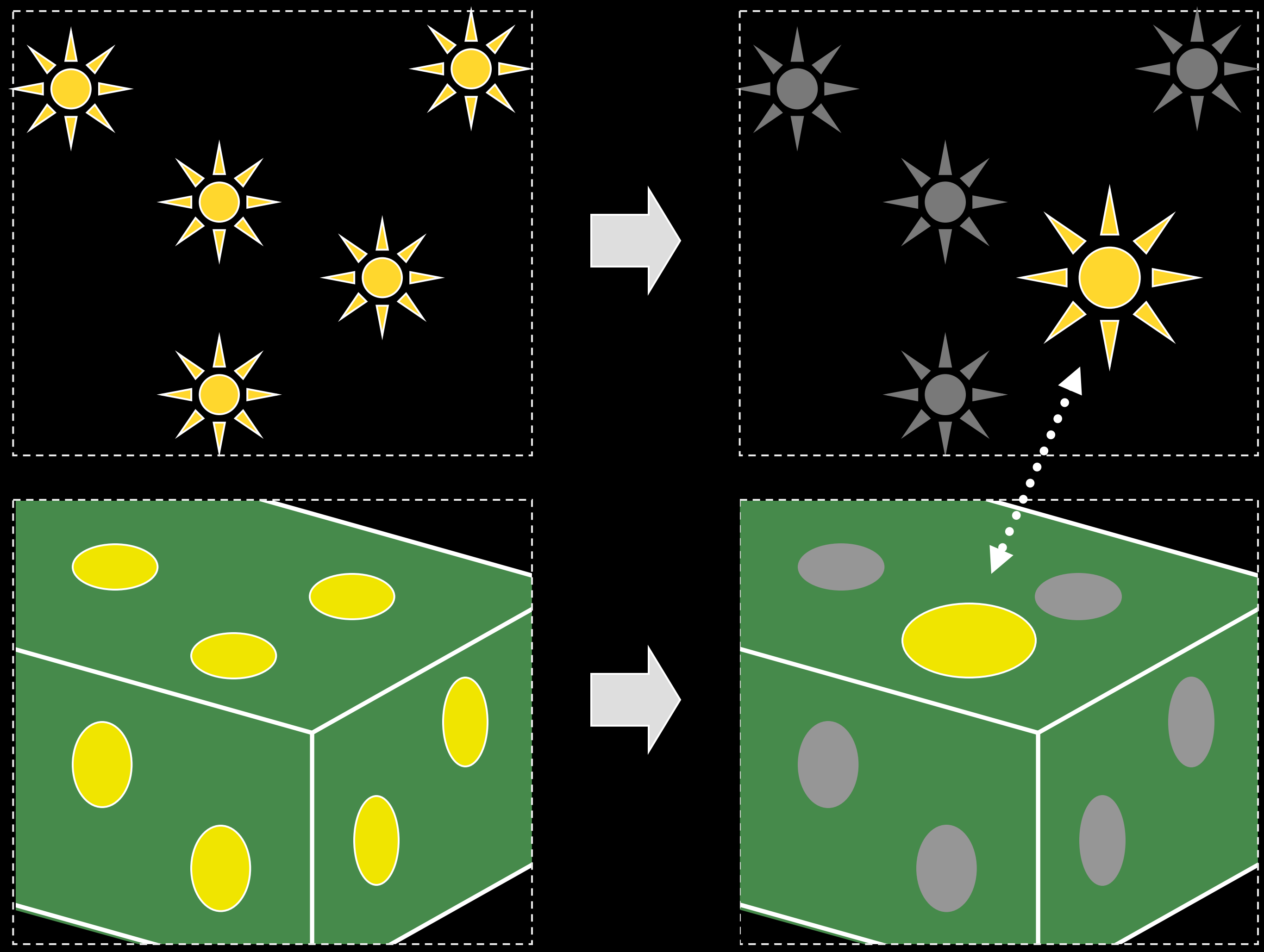


Cluster Representatives



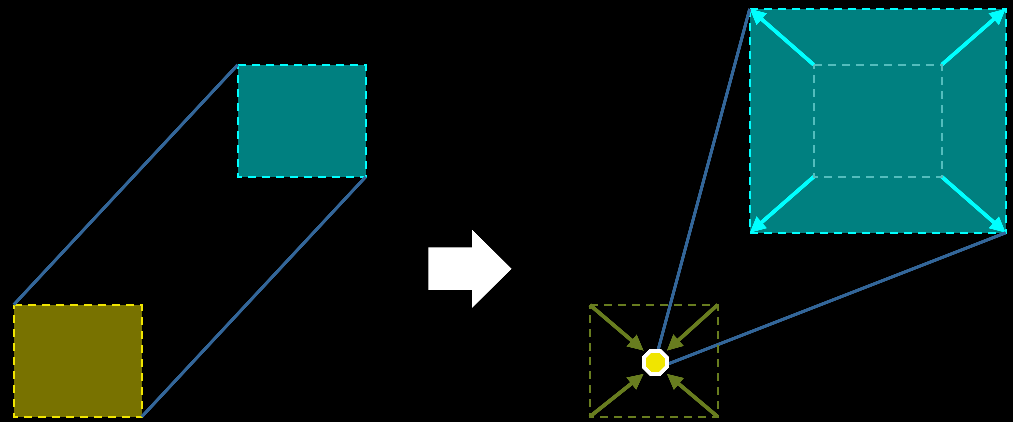


Cluster Representatives

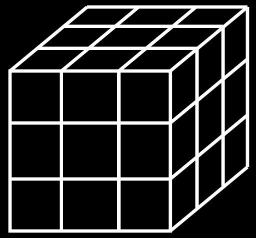


Error Bounds

- Collapse cluster-cluster interactions to point-cluster
  - Minkowski sums
  - Reuse bounds from Lightcuts



- Compute maximum over multiple BRDFs
  - Rasterize into cube-maps



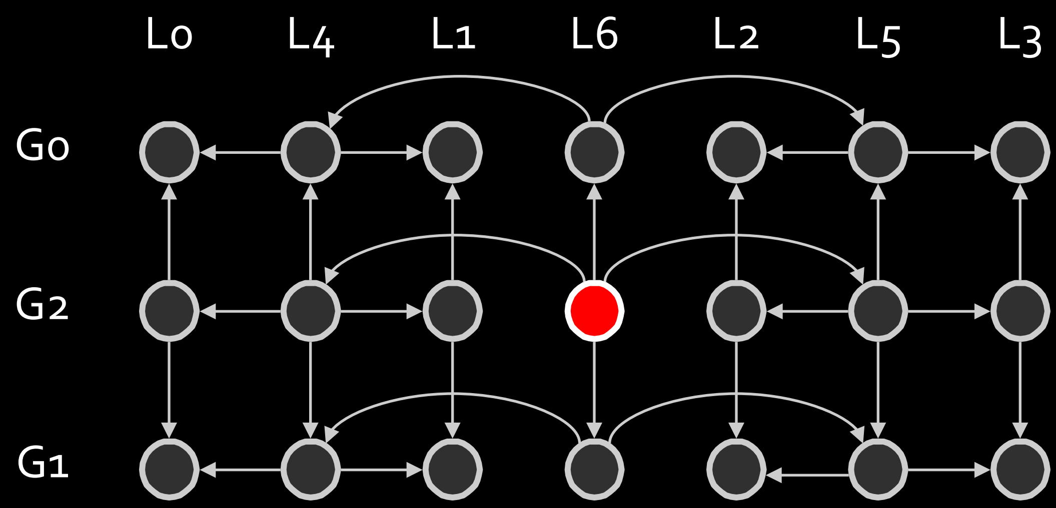
- More details in the paper

Algorithm Summary

- Once per image
  - Create lights and light tree
- For each pixel
  - Create gather points and gather tree for pixel
  - Adaptively refine clusters in product graph until all cluster errors < perceptual metric

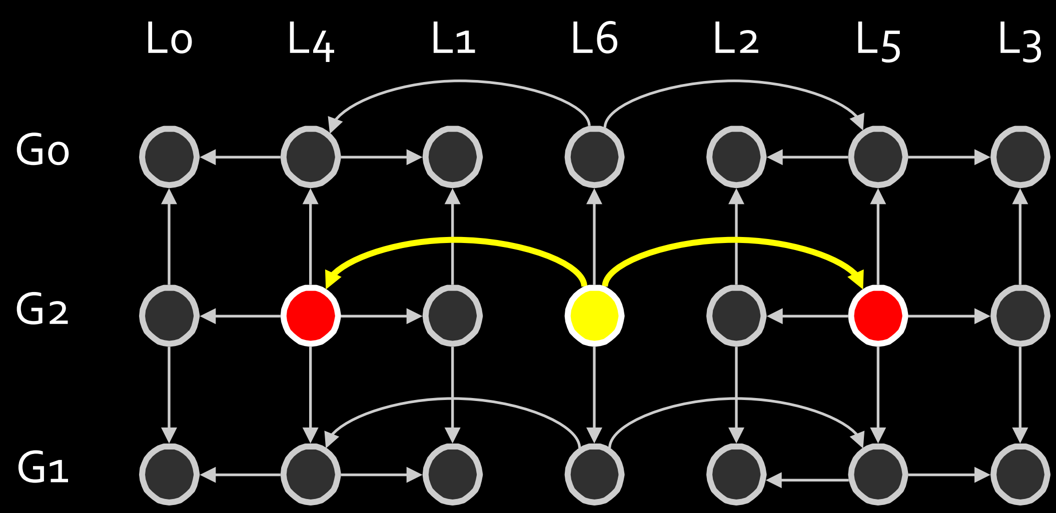
Scalability

- Start with a coarse cut
  - Eg, source node of product graph



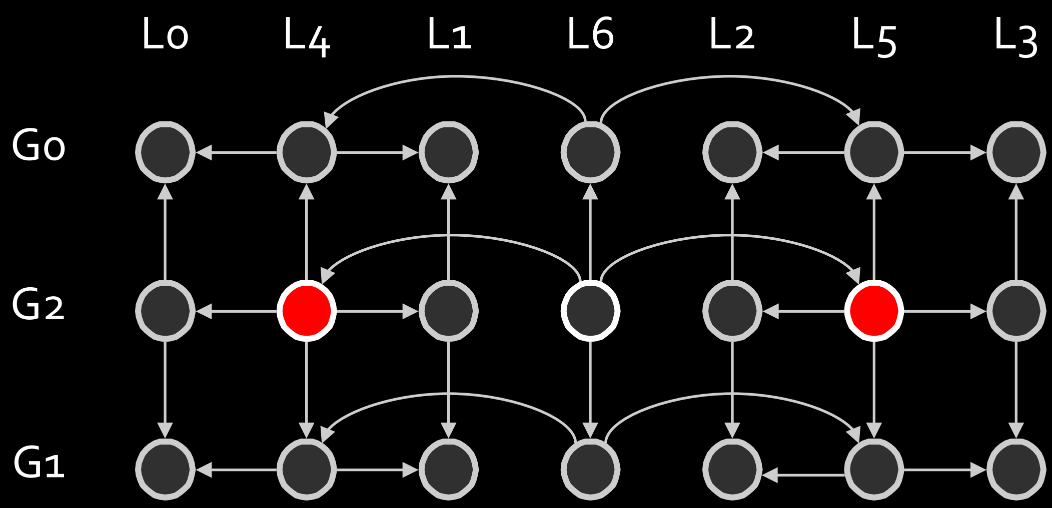
Scalability

- Choose node with largest error bound & refine
  - In gather or light tree



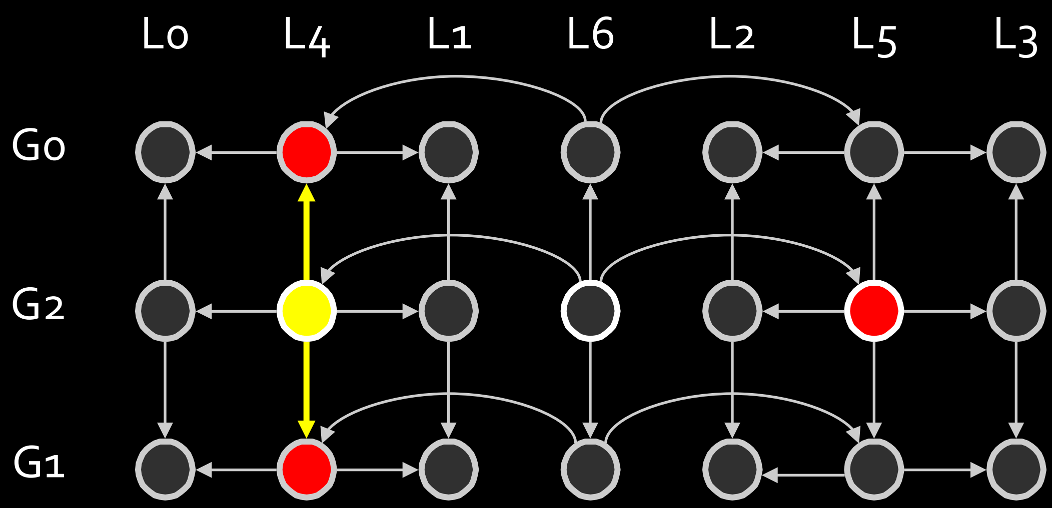
Scalability

- Choose node with largest error bound & refine
  - In gather or light tree



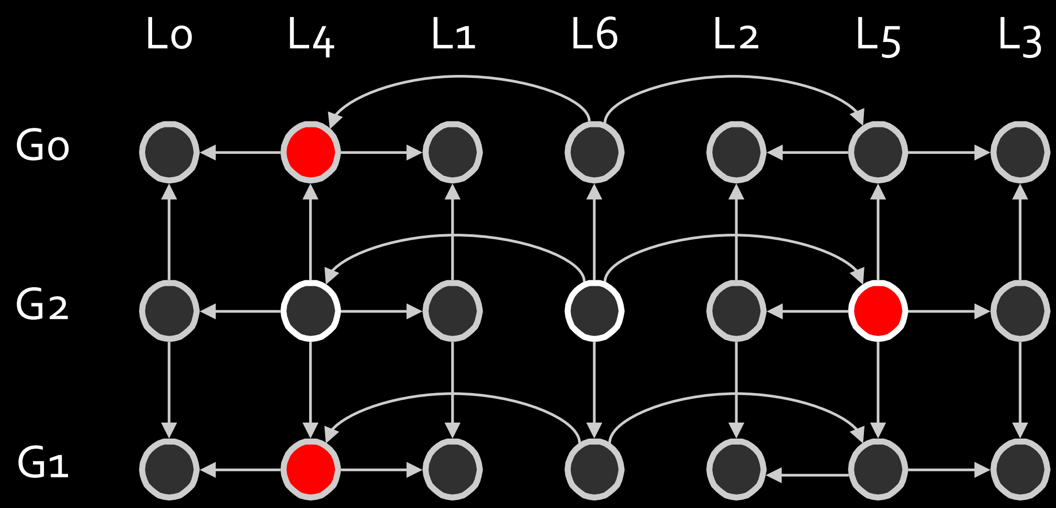
Scalability

- Repeat process



Algorithm summary

- Until all clusters errors < perceptual metric
  - 2% of pixel value (Weber's law)



Results

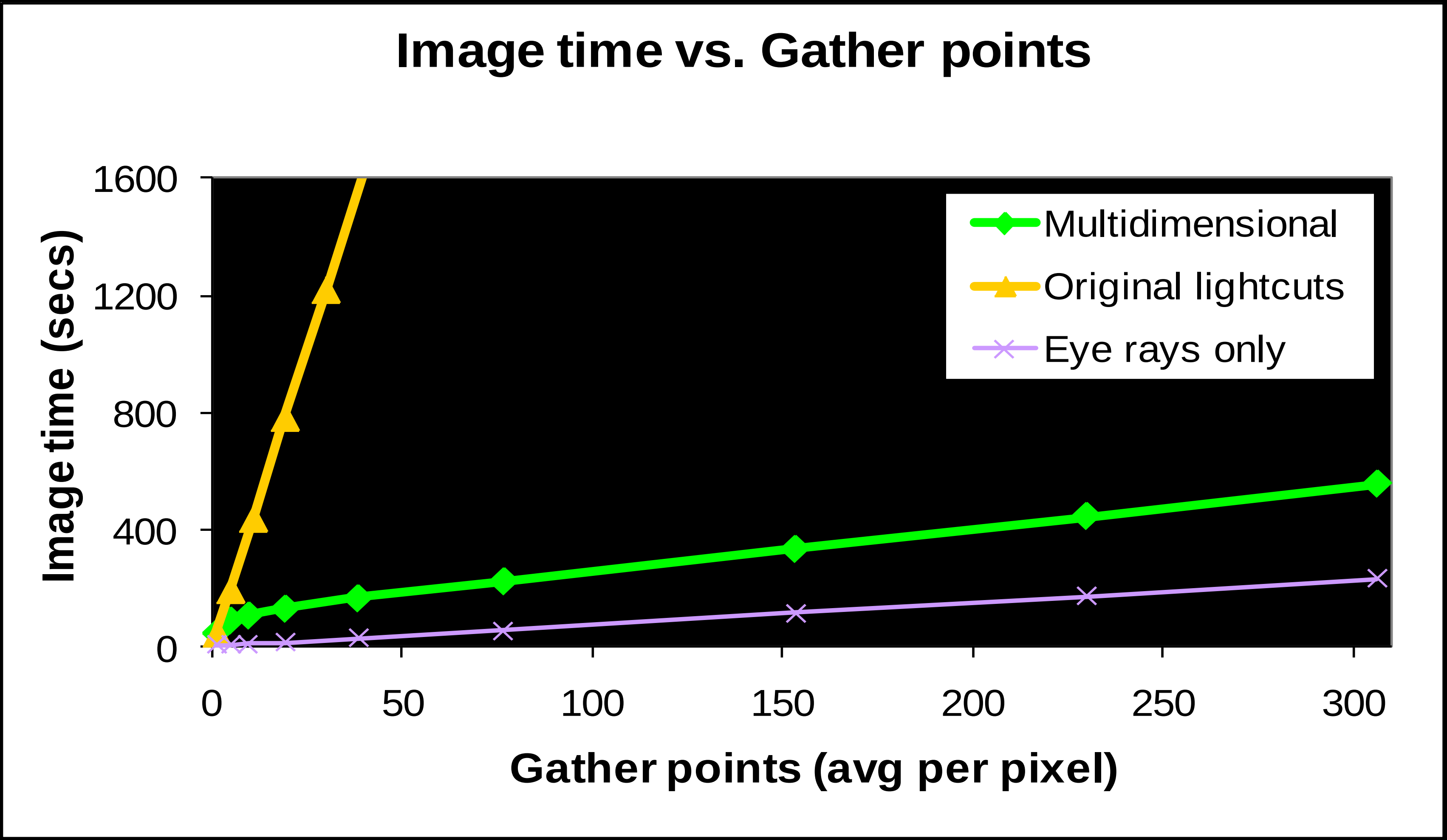
- Limitations
  - Some types of paths not included
    - Eg, caustics
  - Prototype only supports diffuse, Phong, and Ward materials and isotropic media

Roulette

7,047,430 Pairs per pixel   Time 590 secs

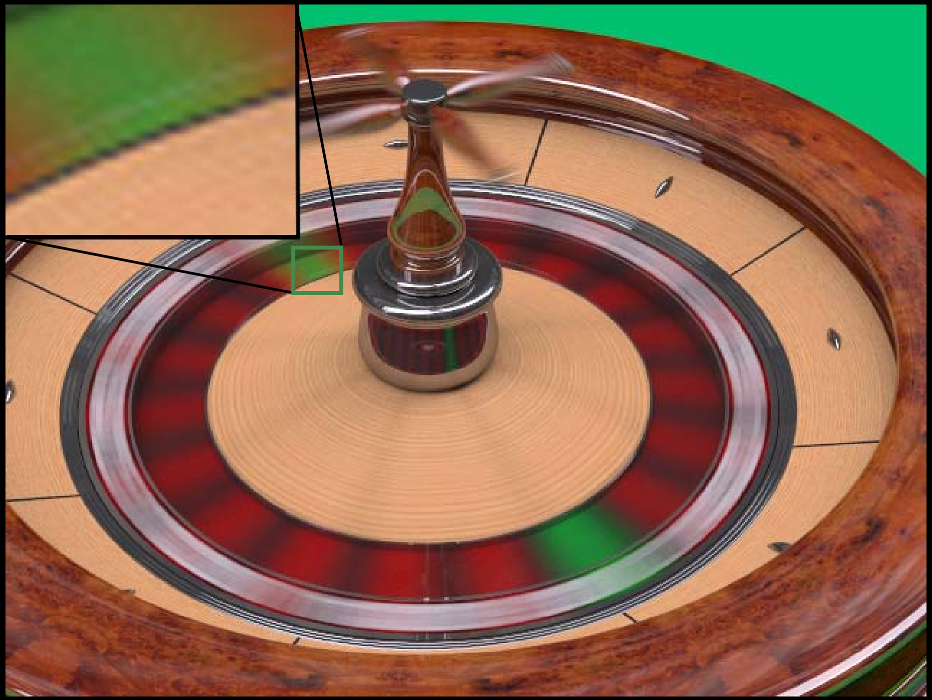
Avg cut size 174 (0.002%)

Scalability



Metropolis Comparison

Zoomed insets



Our result
Time 9.8min

Metropolis
Time 148min (15x)
Visible noise
5% brighter (caustics etc.)

Kitchen

5,518,900 Pairs per pixel   Time 705 secs
Avg cut size 936 (0.017%)

180 Gather points X 13,000 Lights = 234,000 Pairs per pixel

Avg cut size 447 (0.19%)

114,149,280 Pairs per pixel   Avg cut size 821

Time 1740 secs

Scalability with many lights

Approach #2:

Matrix Row-Column sampling

Hašan et al., SIGGRAPH 2007

Slides courtesy Miloš Hašan:

http://www.cs.cornell.edu/~mhasan/

Improving Scalability and Performance

Brute force:

10 min

13 min

20 min

↓ ↓ ↓

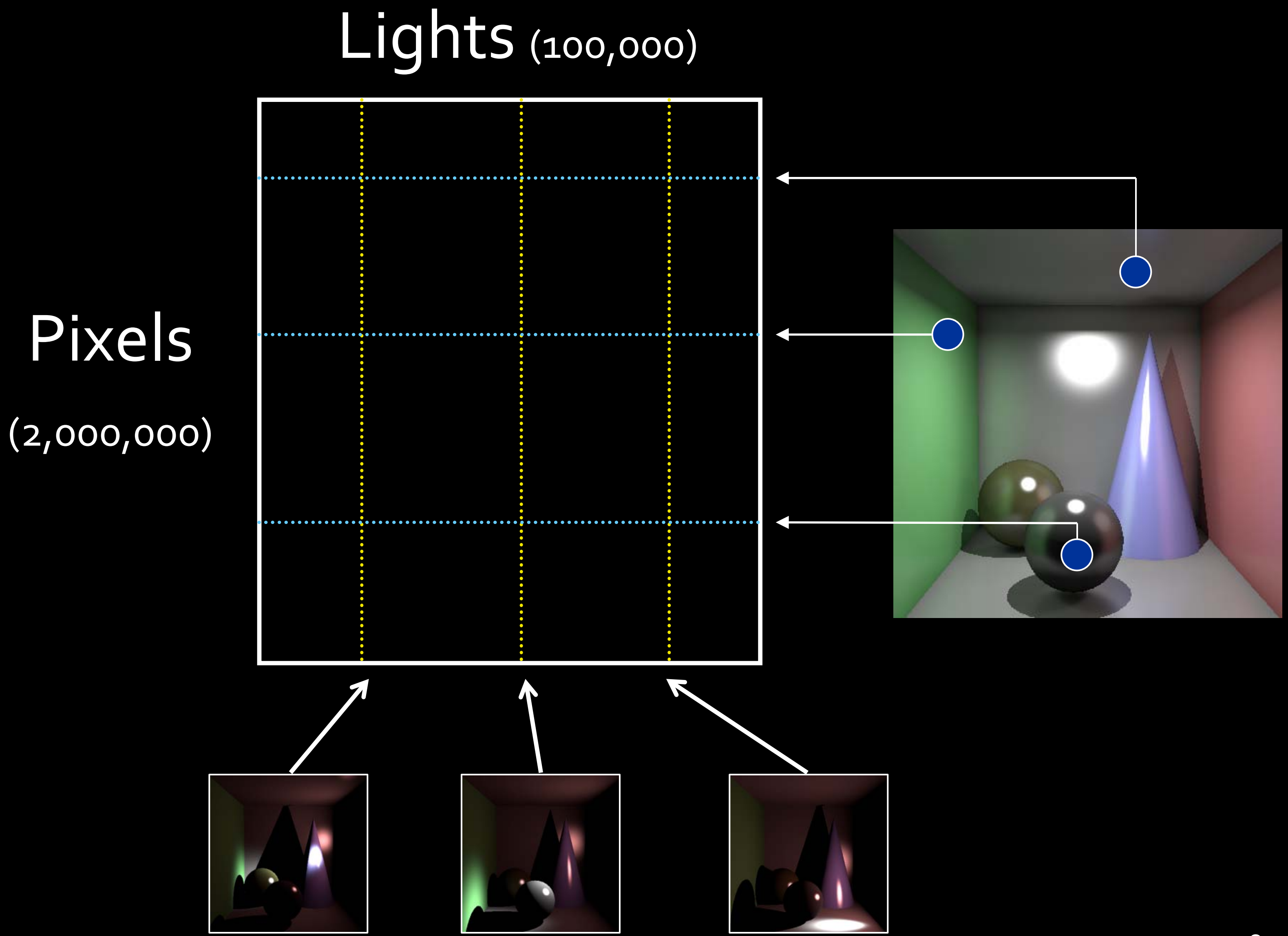
Our result:

3.8 sec

13.5 sec

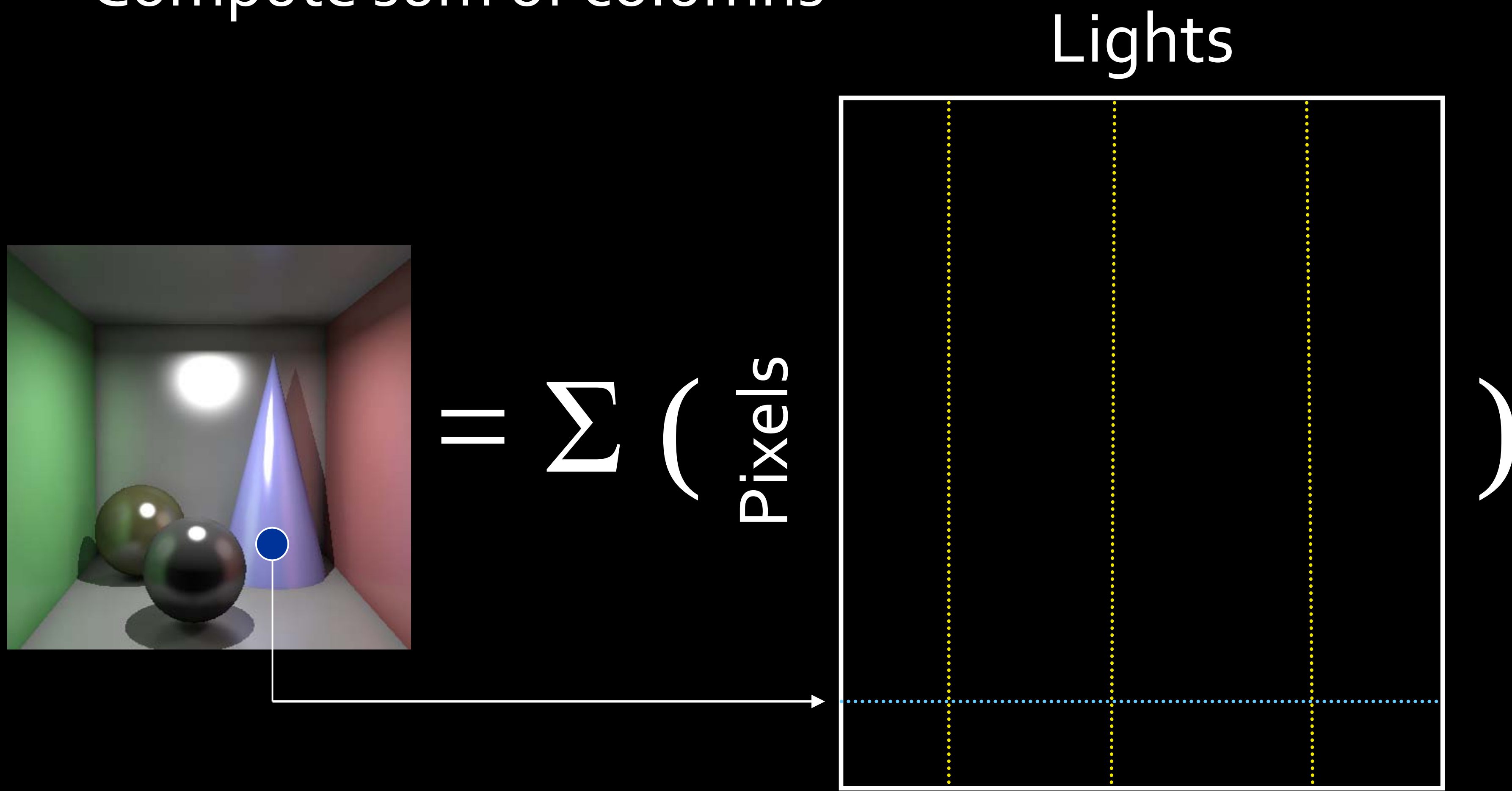
16.9 sec

A Matrix Interpretation



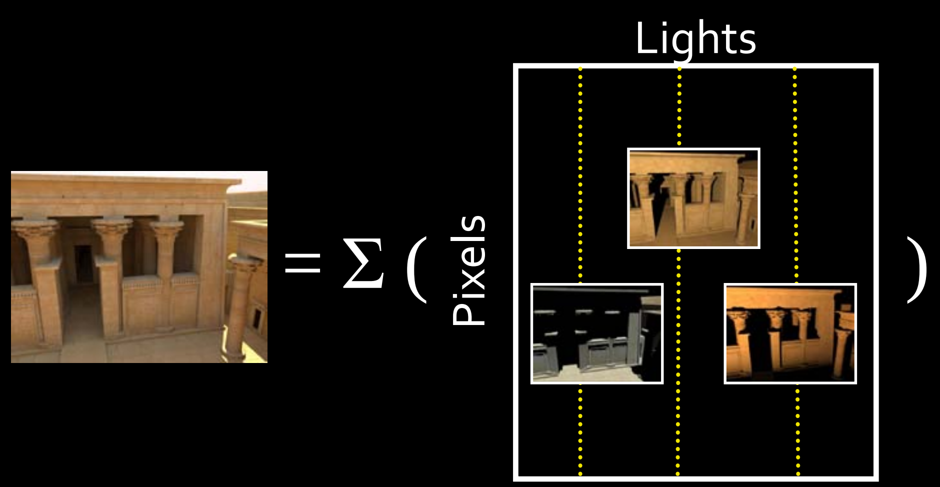
Problem Statement

- Compute sum of columns

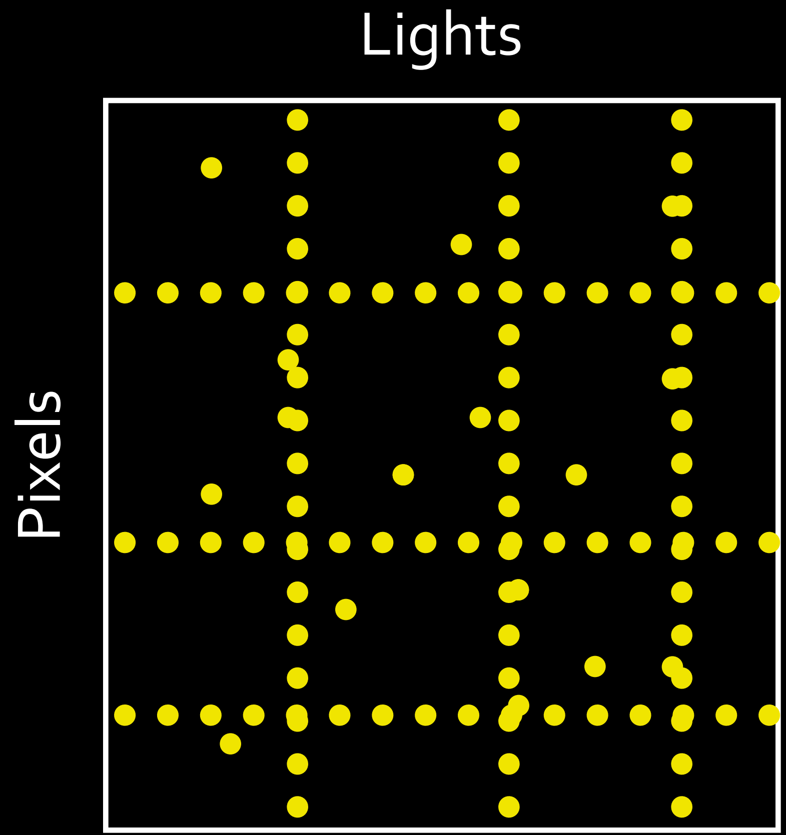


Low-Rank Assumption

- Column space is (close to) low-dimensional



Ray-tracing vs Shadow Mapping

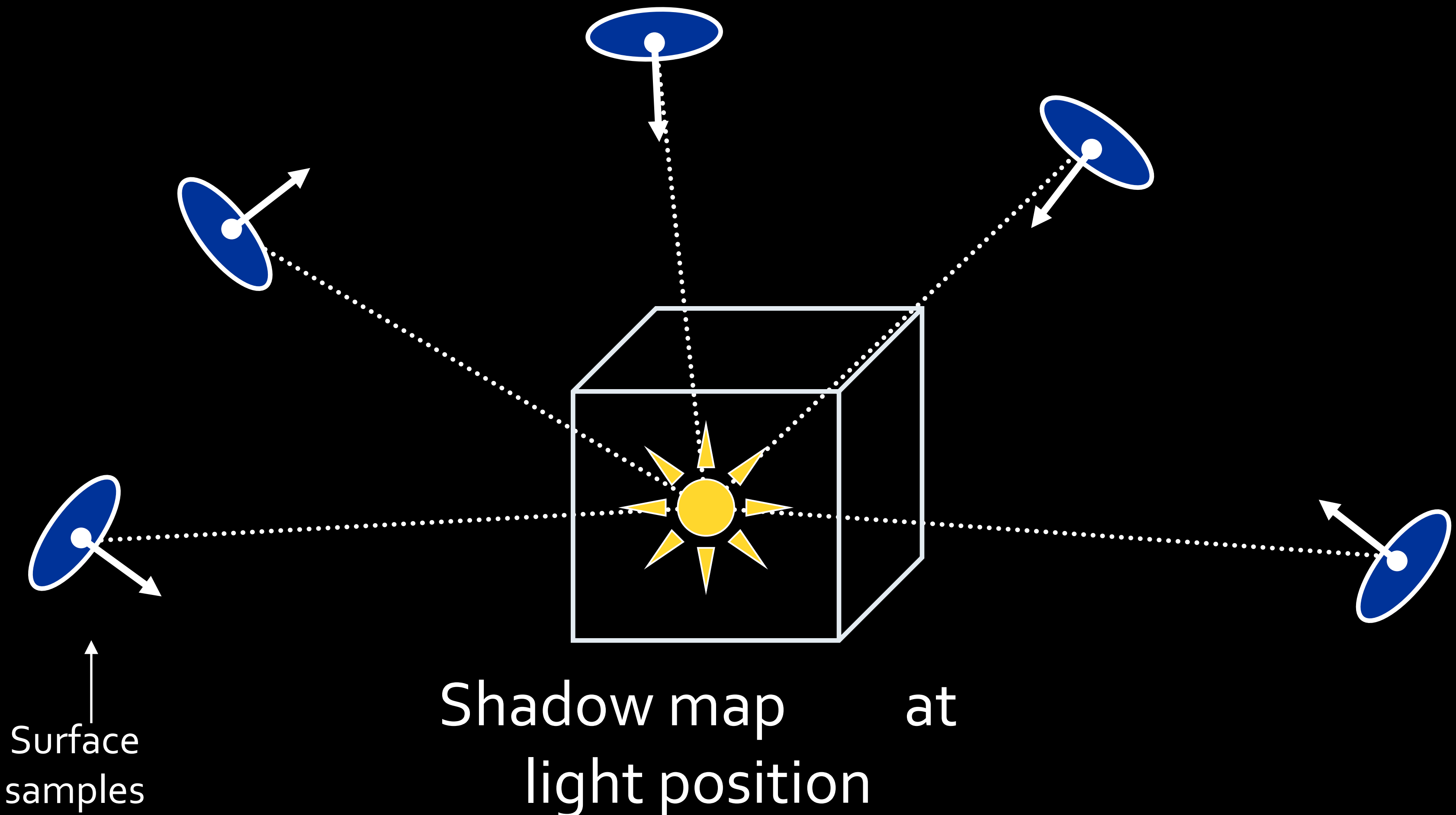


Point-to-many-points visibility: Shadow-mapping

Point-to-point visibility: Ray-tracing

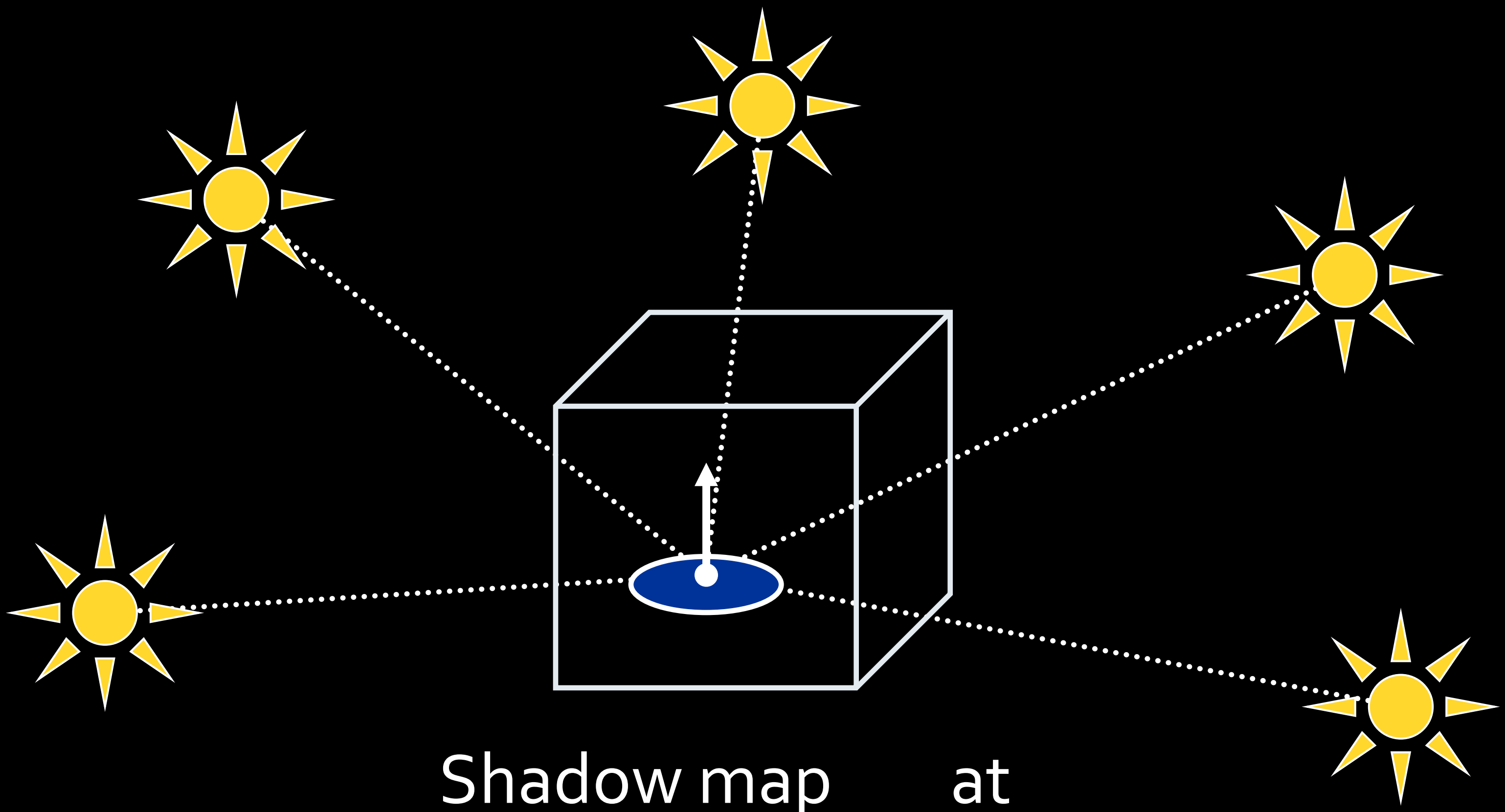
Computing Column Visibility

- Regular Shadow Mapping



Row-Column Duality

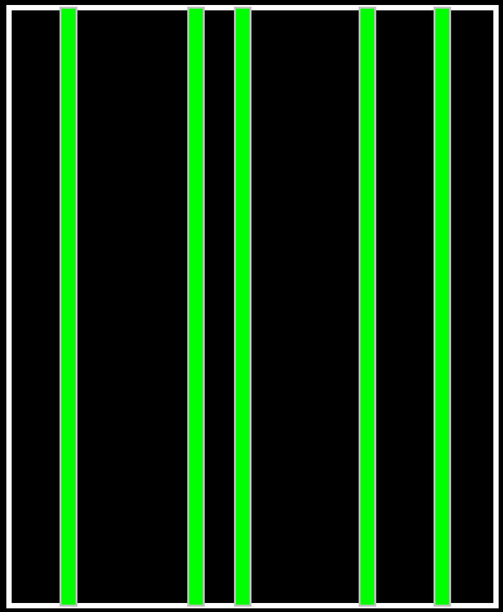
- Rows: Also Shadow Mapping!



Shadow map at
sample position

Image as a Weighted Column Sum

- The following is possible:



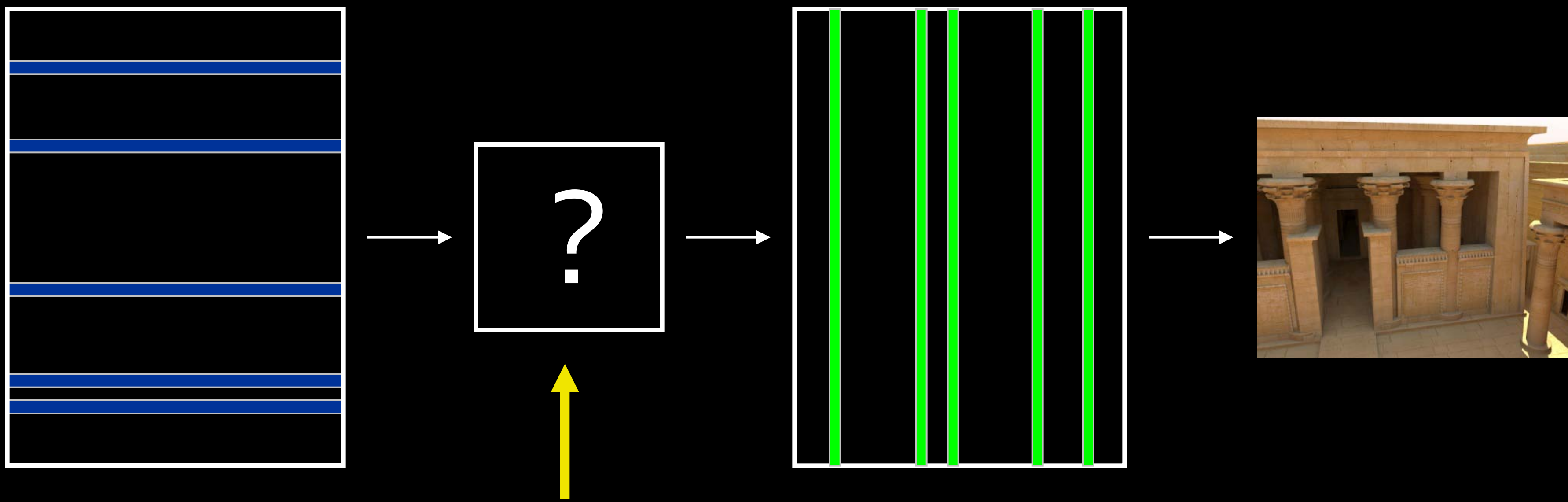
Compute small subset of columns

→

compute weighted sum

- Use rows to choose a good set of columns!

The Row-Column Sampling Idea



compute rows

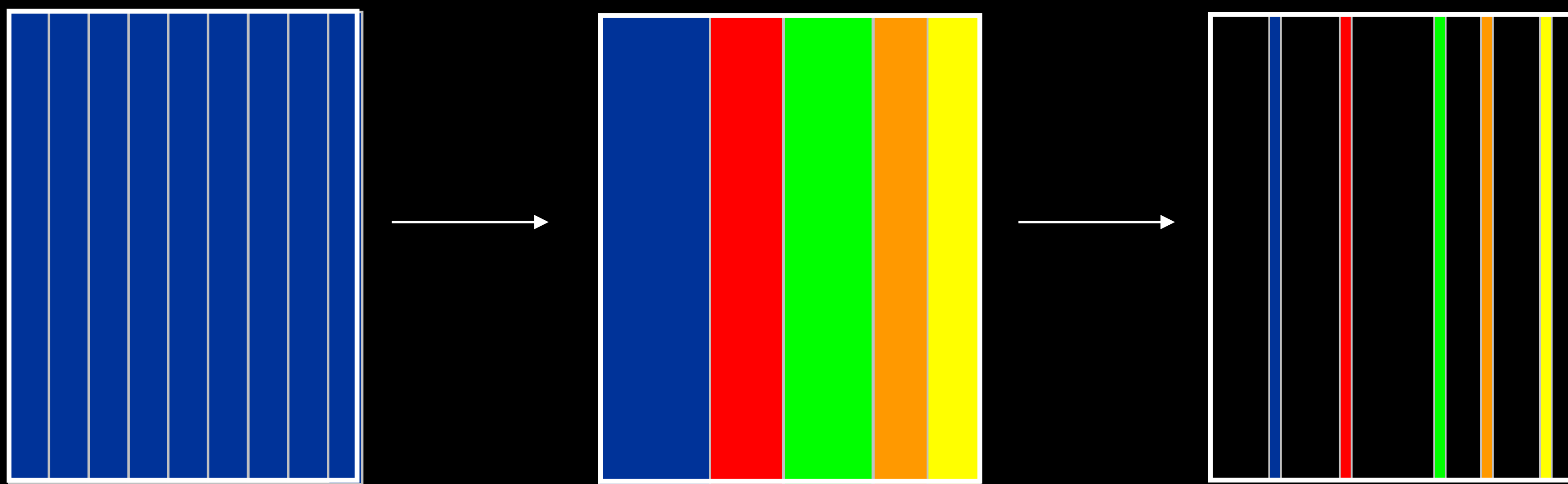
choose columns and weights

how to choose columns and weights?

compute columns

weighted sum

Clustering Approach

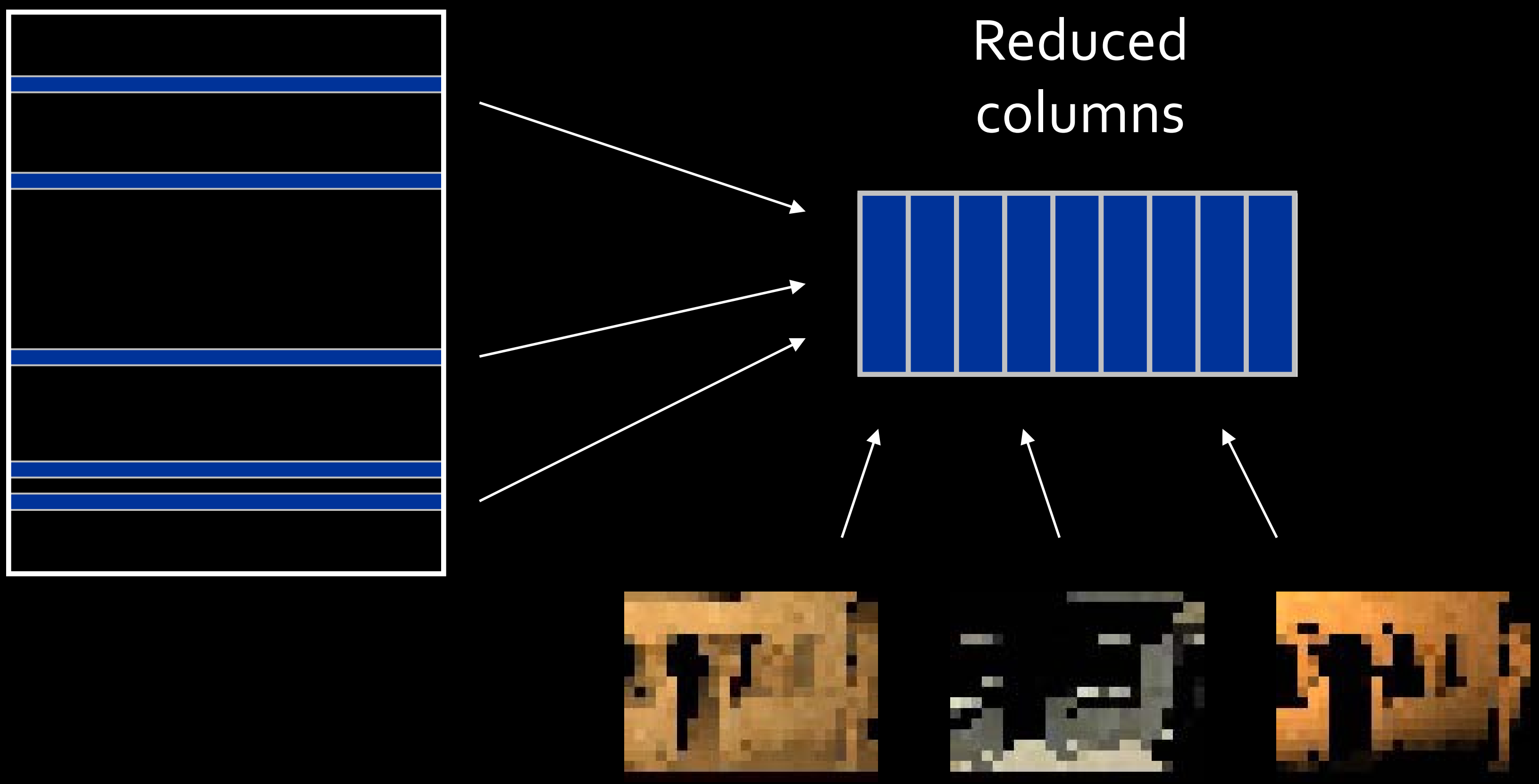


Columns

Clustering

Choose representative columns

Reduced Matrix

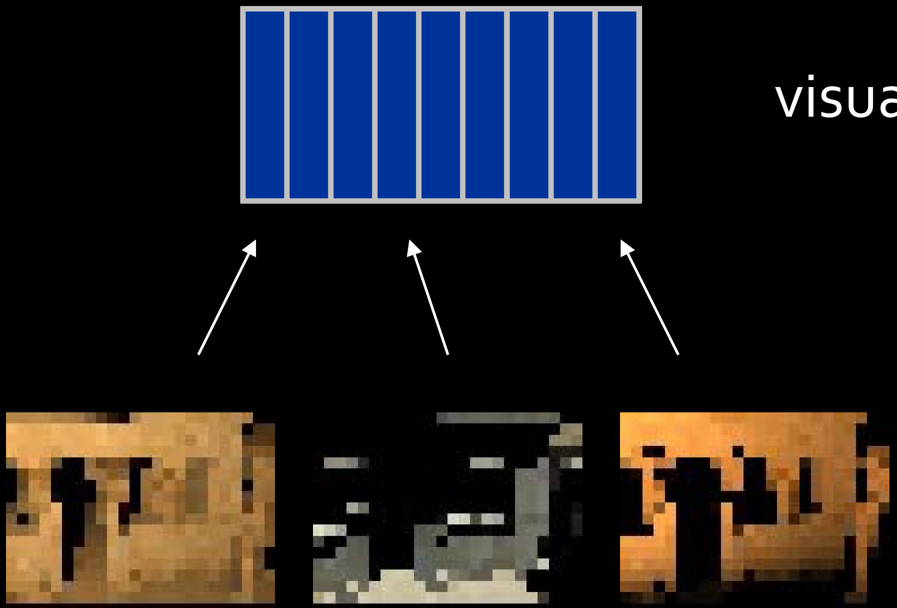


Weights and Information Vectors

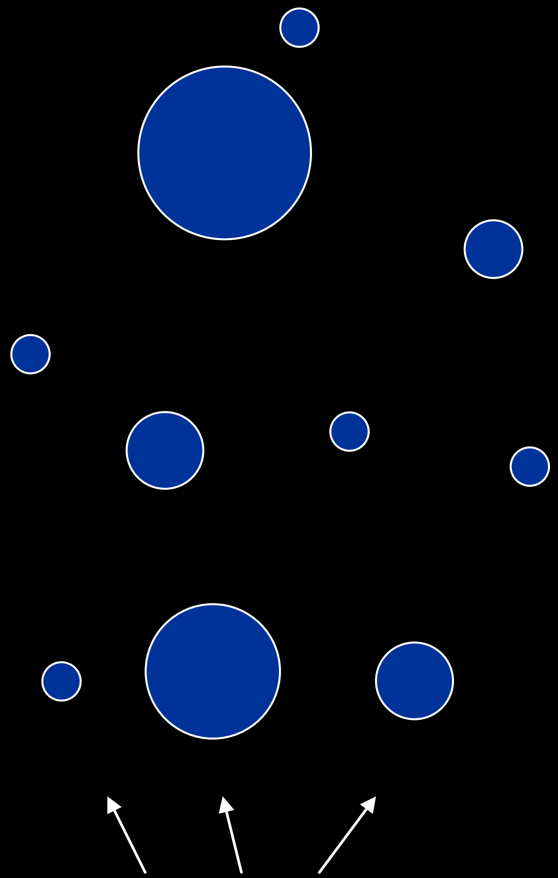
- Weights w_i
  - Norms of reduced columns
  - Represent the “energy” of the light
- Information vectors x_i
  - Normalized reduced columns
  - Represent the “kind” of light’s contribution

Visualizing the Reduced Columns

Reduced columns: vectors in high-dimensional space



visualize as …



radius = weight

position = information vector

Monte Carlo Estimator

- Algorithm:
  1. Cluster reduced columns
  2. Choose a representative in each cluster, with probability proportional to weight
  3. Approximate other columns in cluster by (scaled) representative
- This is a Monte Carlo estimator
- Which clustering minimizes its variance?

The Clustering Objective

- Minimize: $$\underbrace{\sum_{p=1,\dots,k} cost(C_p)}_{\text{total cost of all clusters}}$$

- where: $$cost(C) = \sum_{i,j \in C} w_i \, w_j \, \|\mathbf{x}_i - \mathbf{x}_j\|^2$$

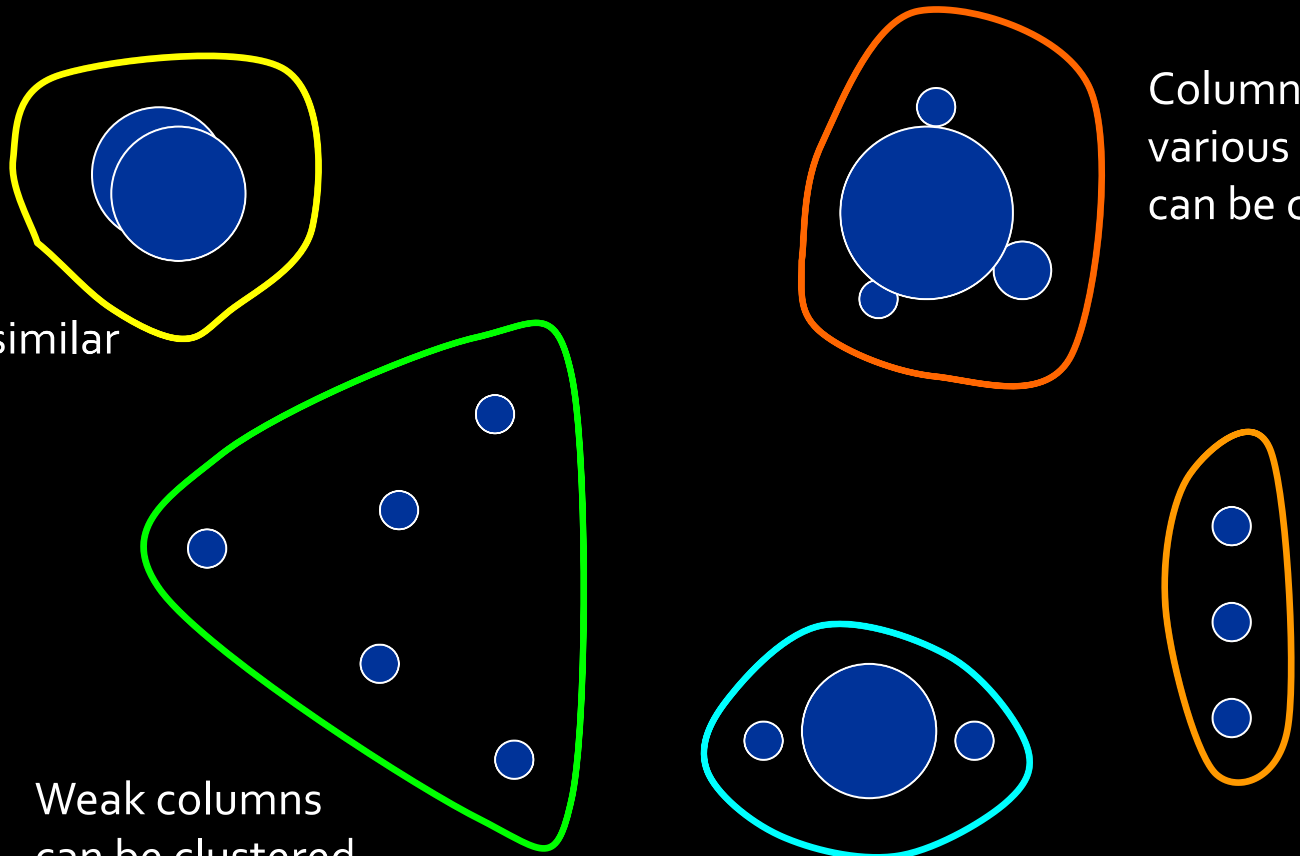
cost of a cluster

sum over all pairs in it

weights

squared distance between information vectors

Clustering Illustration



Columns with various intensities can be clustered

Strong but similar columns

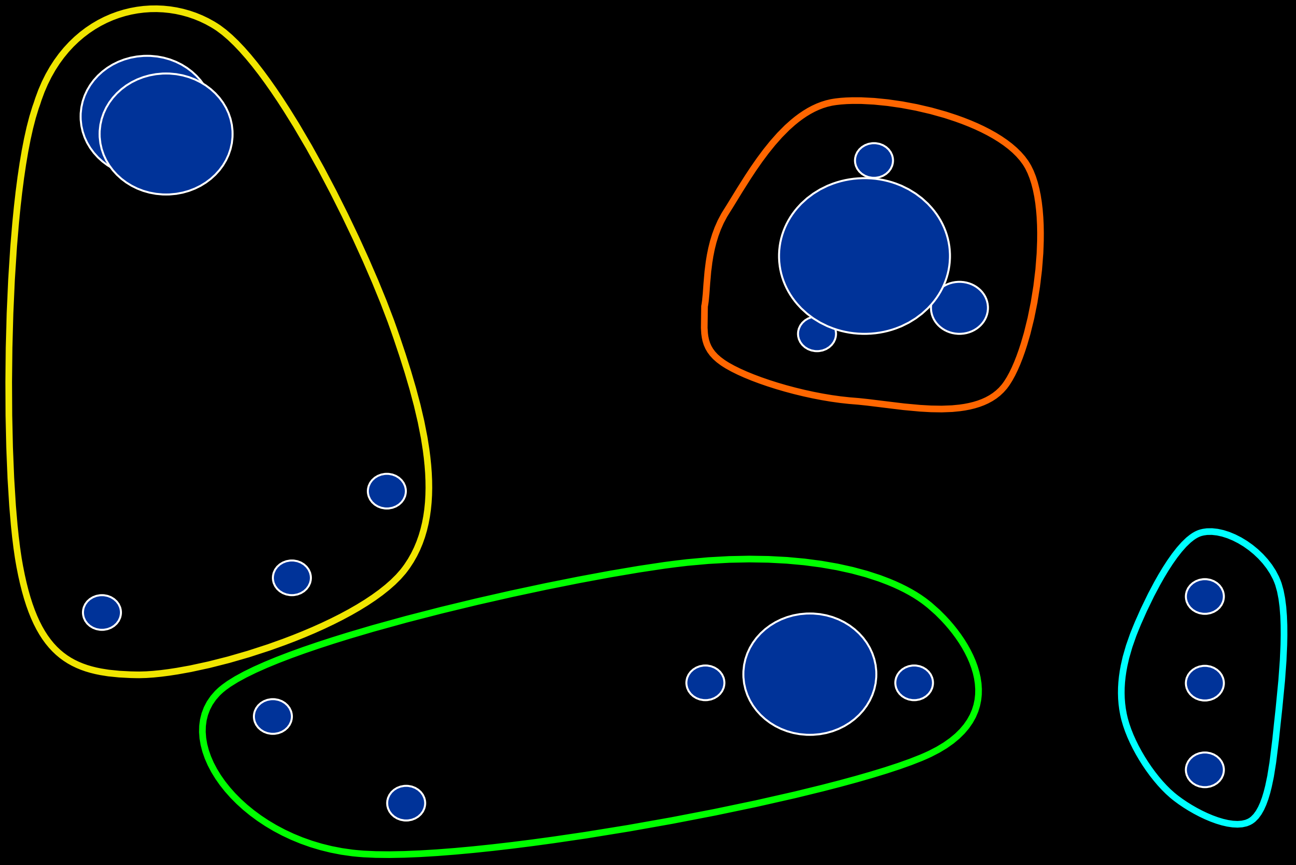
Weak columns can be clustered more easily

$$cost(C) = \sum_{i,j \in C} w_i \; w_j \; \|\mathbf{x}_i - \mathbf{x}_j\|^2$$

How to minimize?

- Problem is NP-hard
- Not much previous research
- Should handle large input:
  - 100,000 points
  - 1000 clusters
- We introduce 2 heuristics:
  - Random sampling
  - Divide & conquer

Clustering by Random Sampling

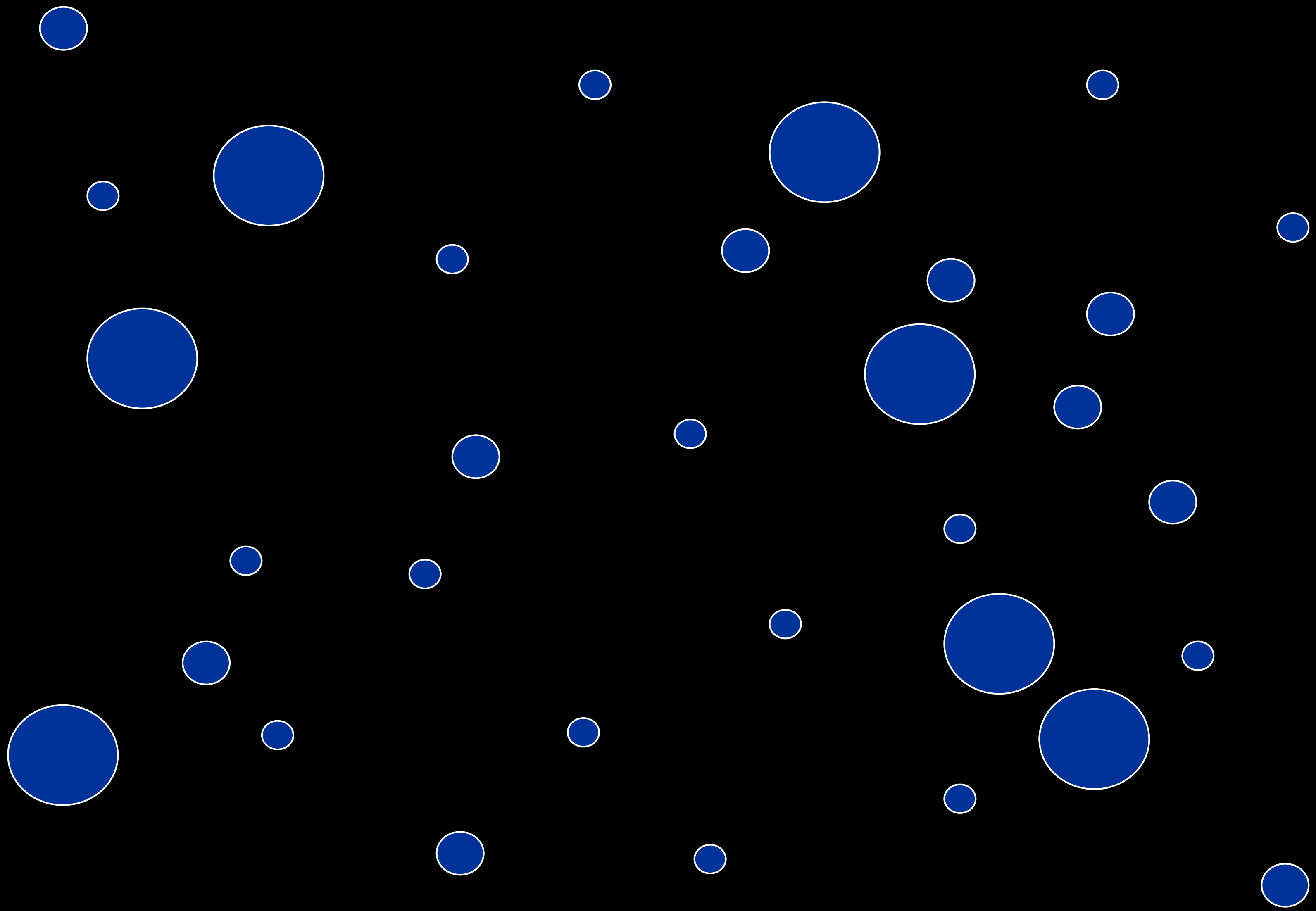


- Very fast (use optimized BLAS)
- Some clusters might be too small / large

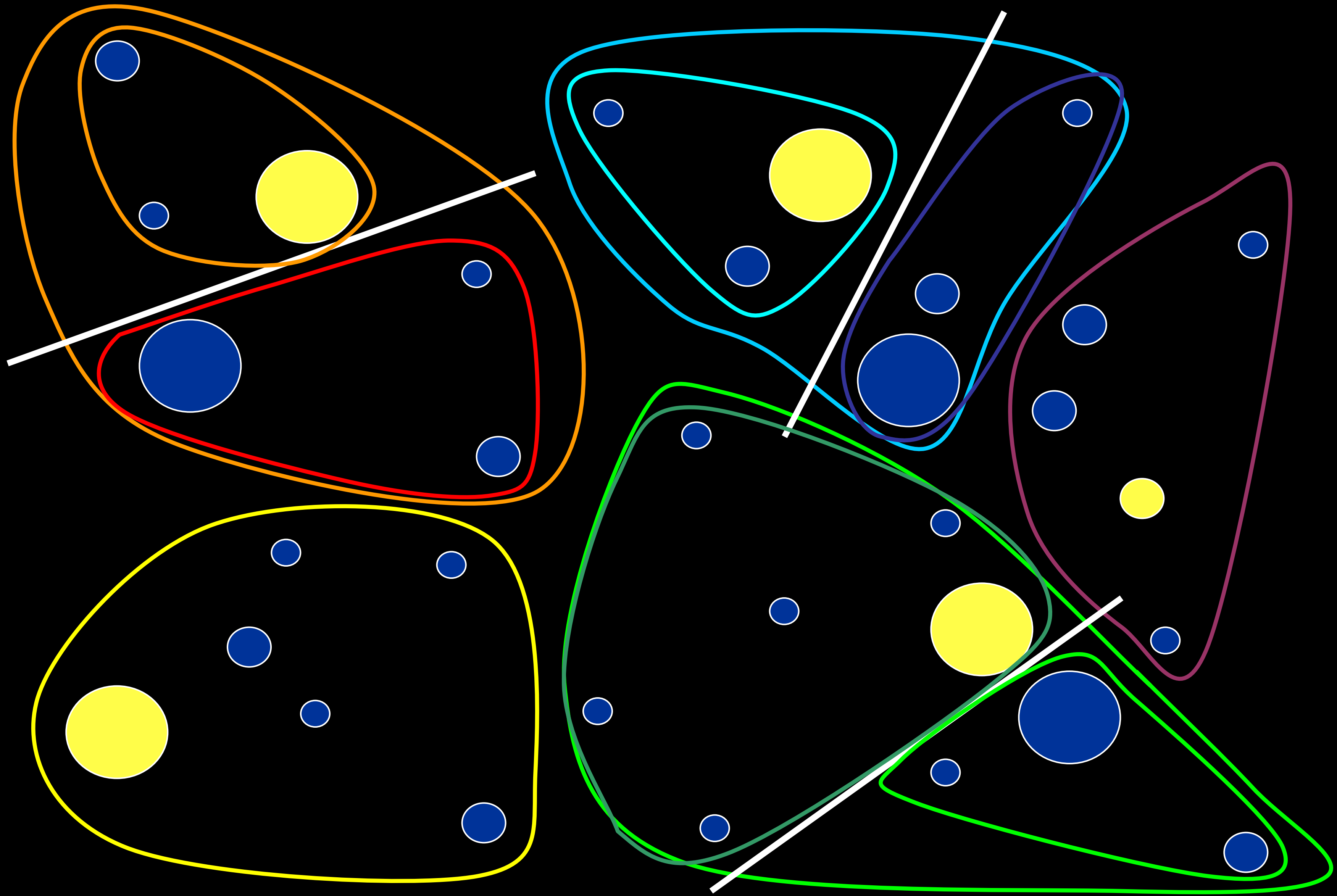
Clustering by Divide & Conquer

- Splitting small clusters is fast
- Splitting large clusters is slow

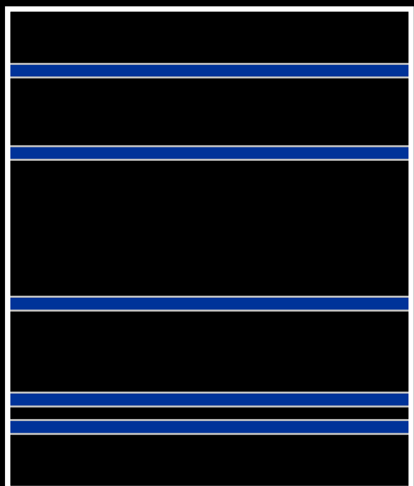
Combined Clustering Algorithm



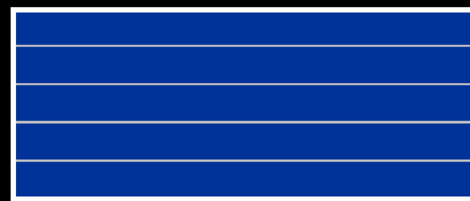
Combined Clustering Algorithm



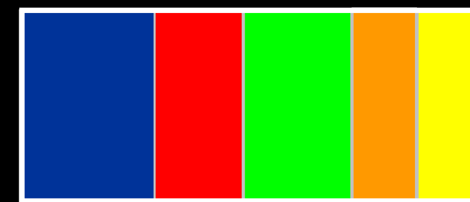
Full Algorithm



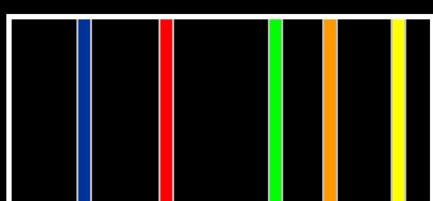
Compute rows (GPU)



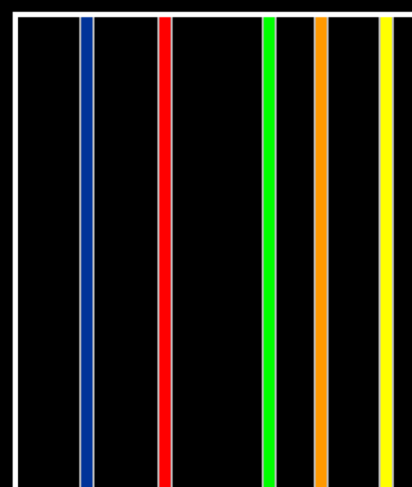
Assemble rows into reduced matrix



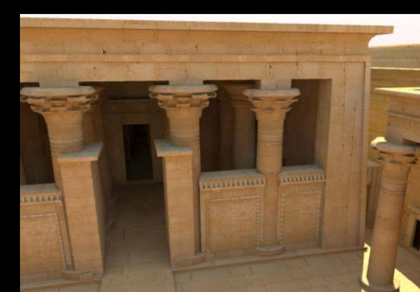
Cluster reduced columns



Choose representatives



Compute columns (GPU)



Weighted sum

Example: Temple

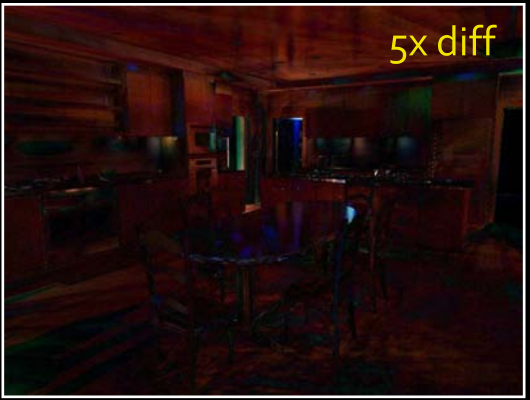
- 2.1m polygons
- Mostly indirect & sky illumination
- Indirect shadows

Our result: 16.9 sec (300 rows + 900 columns)

Reference: 20 min (using all 100k lights)

Example: Kitchen

- 388k polygons
- Mostly indirect illumination
- Glossy surfaces
- Indirect shadows

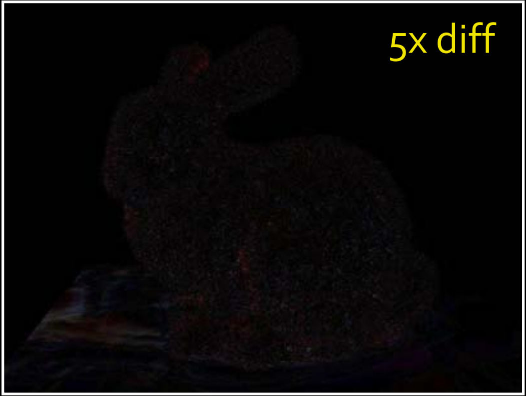


Our result: 13.5 sec (432 rows + 864 columns)

Reference: 13 min (using all 100k lights)

Example: Bunny

- 869k polygons
- Incoherent geometry
- High-frequency lighting
- Kajiya-Kay hair shader



Our result: 3.8 sec (100 rows + 200 columns)

Reference: 10 min (using all 100k lights)