Overview

… follows the structure of the STAR

- Introduction & Welcome (Carsten)
- Many-Light Rendering Concepts (Jan)
  - Basic Idea
  - Improved Virtual Lights Generation
  - Lighting with Virtual Lights
- Really Many Lights: Scalability (Carsten)
- Interactive and Real-Time Rendering (Carsten)
- Conclusions, Outlook, Q&A (Jan & Carsten)

the slides of this part are based on Bruce Walter's and Milos Hasan's presentations

Overview: Scalability

Scalable Solutions for (Really) Many Lights

- VPL usage is more expensive than generation
- Lightcuts
  - illumination at a single receiver: Lightcuts
    "Lightcuts: a Scalable Approach to Illumination" by Walter, Fernandez, Arbree, Bala, Donikian, Greenberg, SIGGRAPH 2005
  - illumination over a pixel: Multidimensional Lightcuts
    "Multidimensional Lightcuts" by Walter, Arbree, Bala, Greenberg, SIGGRAPH 2006
- Matrix Row-Column Sampling
  „Matrix row-column sampling for the many-light problem“ by Hasan, Pellacini, Bala, SIGGRAPH 2007
- important: these methods use virtual POINT lights only

Scalability

Why Many Lights?

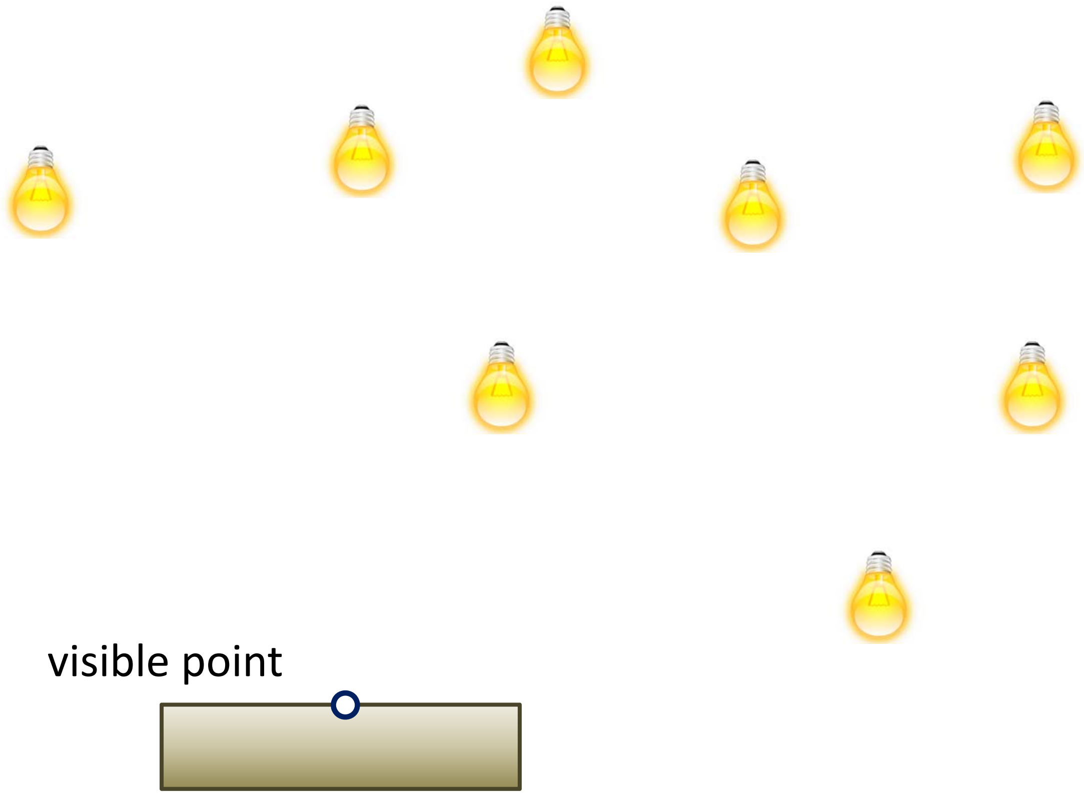
- simulate complex illumination using point lights
  - area lights
  - HDR environment maps
  - sun and sky light
  - indirect illumination
- more lights → more accurate
  - … and more expensive
  - naive cost: linear in lights
- goal: sub-linear cost per light

area lights + sun/sky + indirect

Lightcuts

Setting / Problem

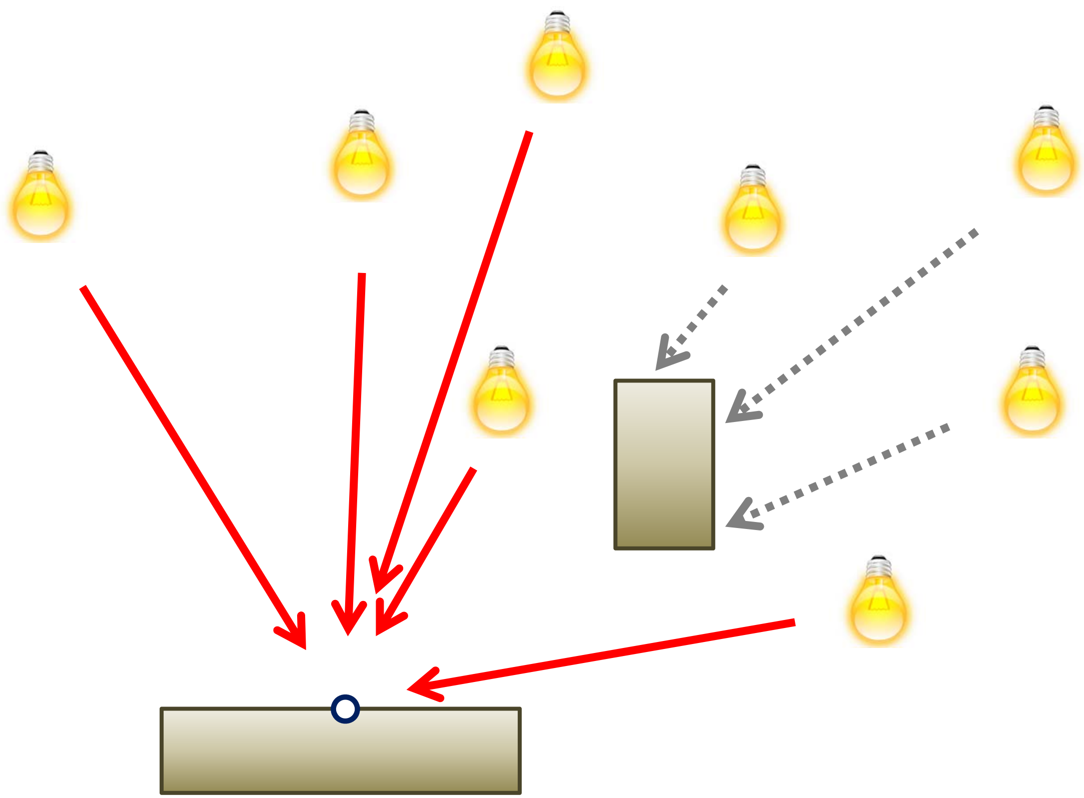
- many lights, a surface point to be lit



Lightcuts

Setting / Problem

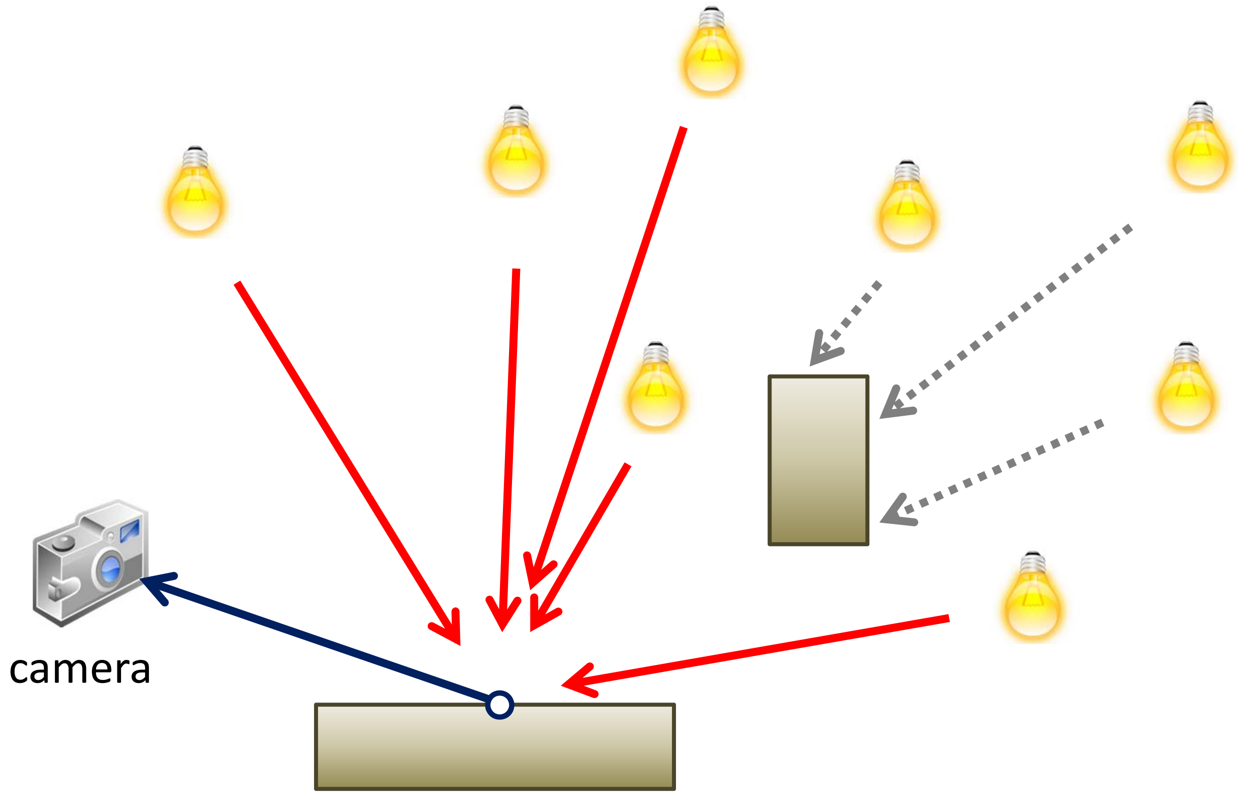
- many lights, a surface point to be lit



Lightcuts

Setting / Problem

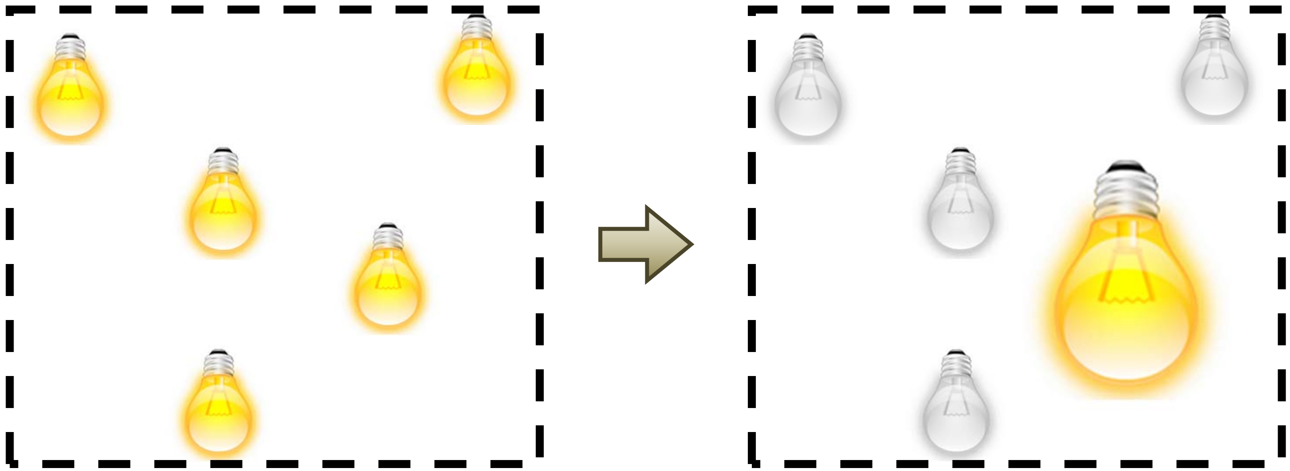
- many lights, a surface point to be lit



Lightcuts

Key Concepts

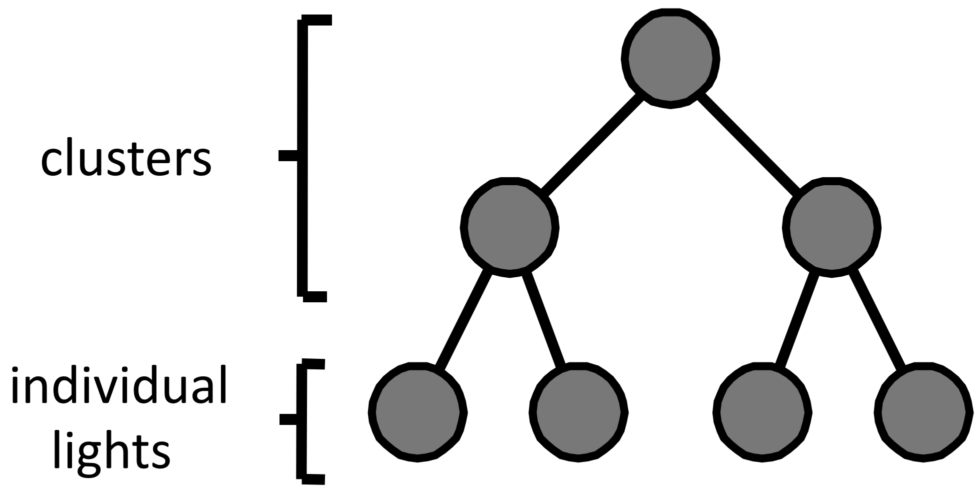
- light cluster
- light tree
- a cut: set of nodes that partitions the lights into clusters



Lightcuts

Key Concepts

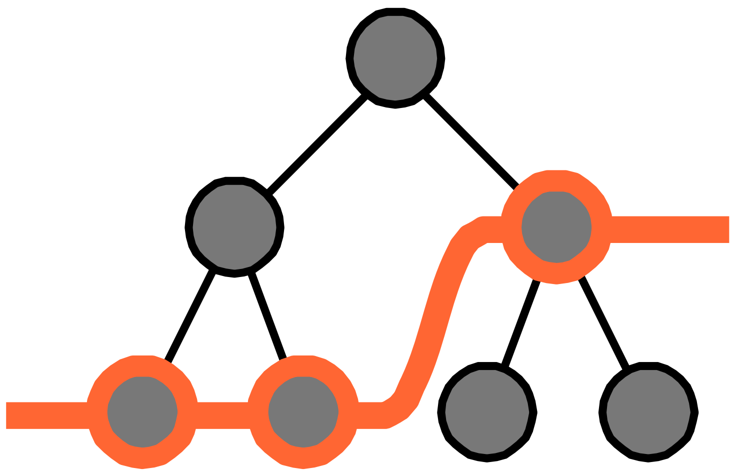
- light cluster
- light tree
- a cut: set of nodes that partitions the lights into clusters



Lightcuts

Key Concepts

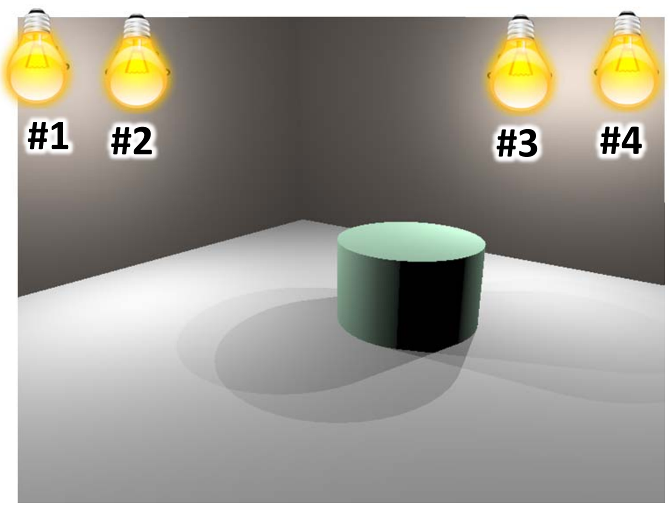
- light cluster
- light tree
- a cut: set of nodes that partitions the lights into clusters

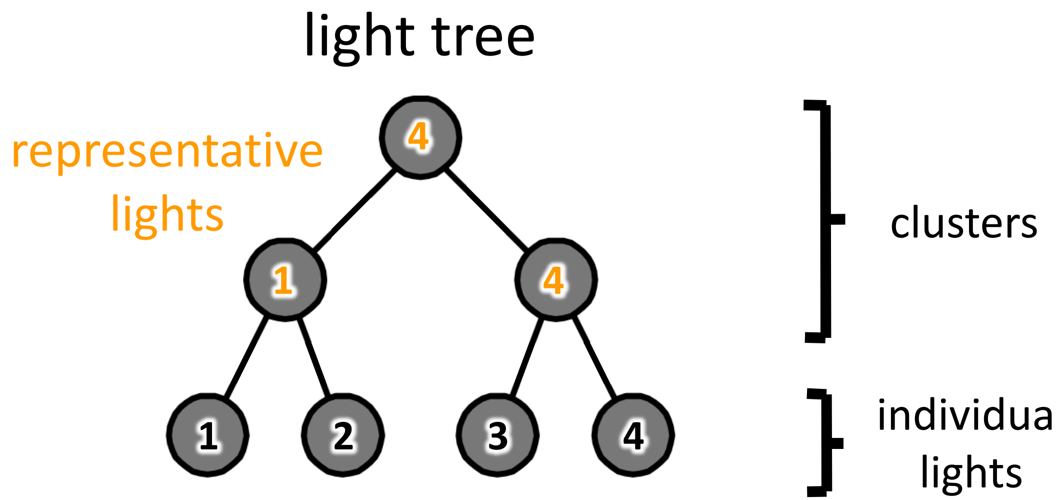


Lightcuts

Simple Example

- 4 individual lights, 3 clusters

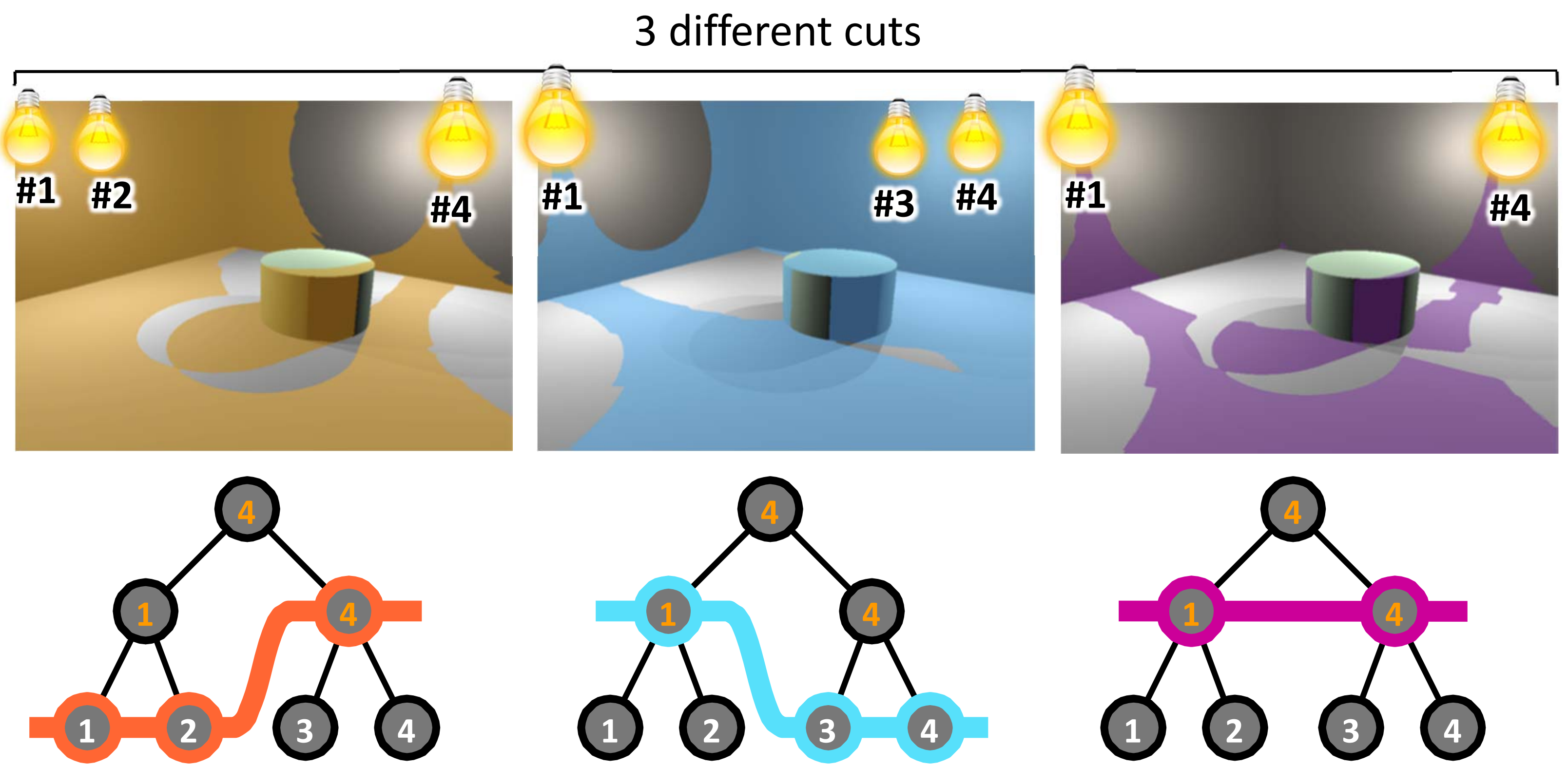




Lightcuts

Simple Example

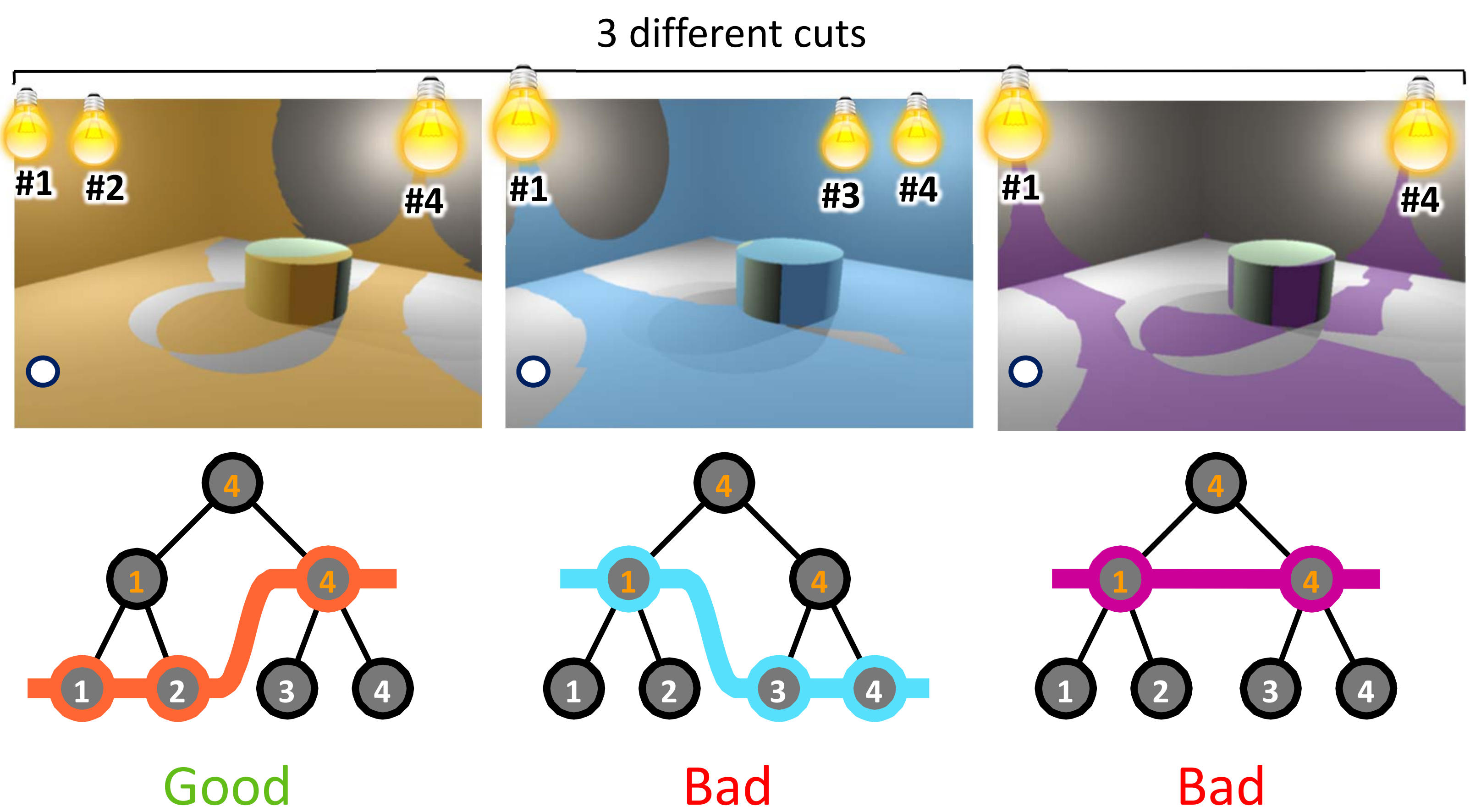
- 4 individual lights, 3 clusters



Lightcuts

Simple Example

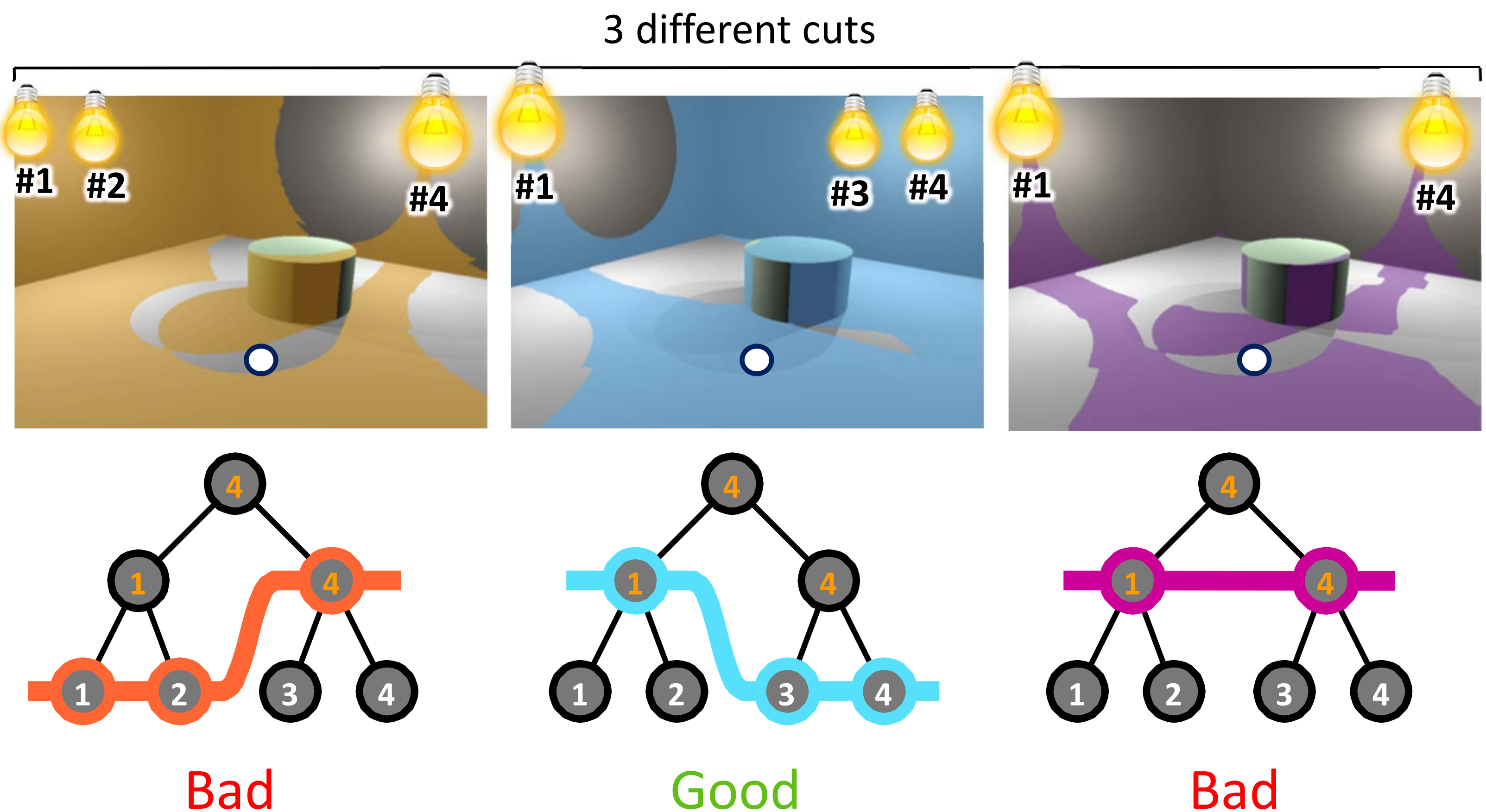
- 4 individual lights, 3 clusters



Lightcuts

Simple Example

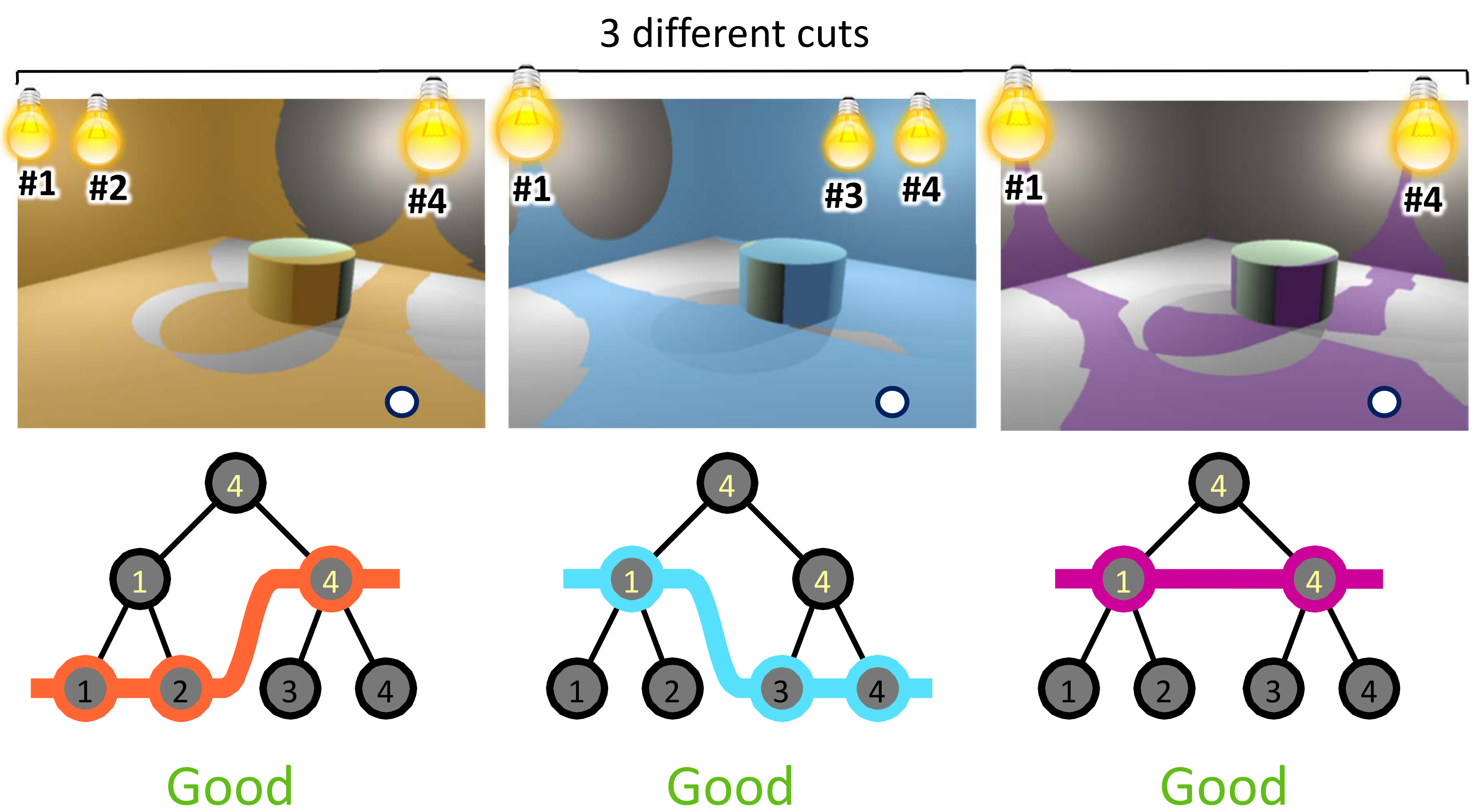
- 4 individual lights, 3 clusters



Lightcuts

Simple Example

- 4 individual lights, 3 clusters



Lightcuts

Algorithm Overview

- pre-process
  - convert illumination to point lights
  - build light tree
- for each visible point
  - choose a cut to approximate the local illumination
    - bound maximum error of cluster approximation
    - refine cluster if error bound is too large

Lightcuts

Perceptual Metric

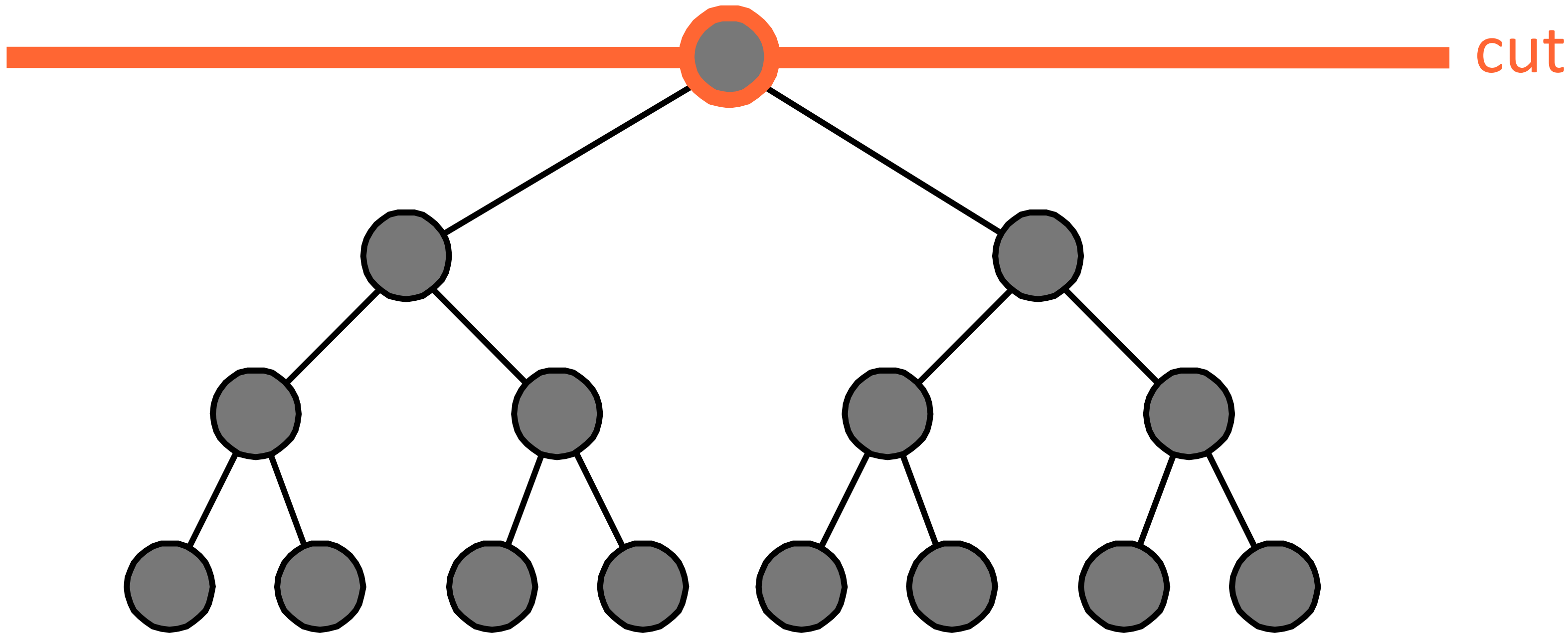
- Weber's Law
  - contrast visibility threshold is fixed percentage of signal
  - used 2% in Lightcuts

- ensure each cluster's error < visibility threshold
  - transitions will not be visible
  - used to select cut

Lightcuts

Cut-Selection Algorithm

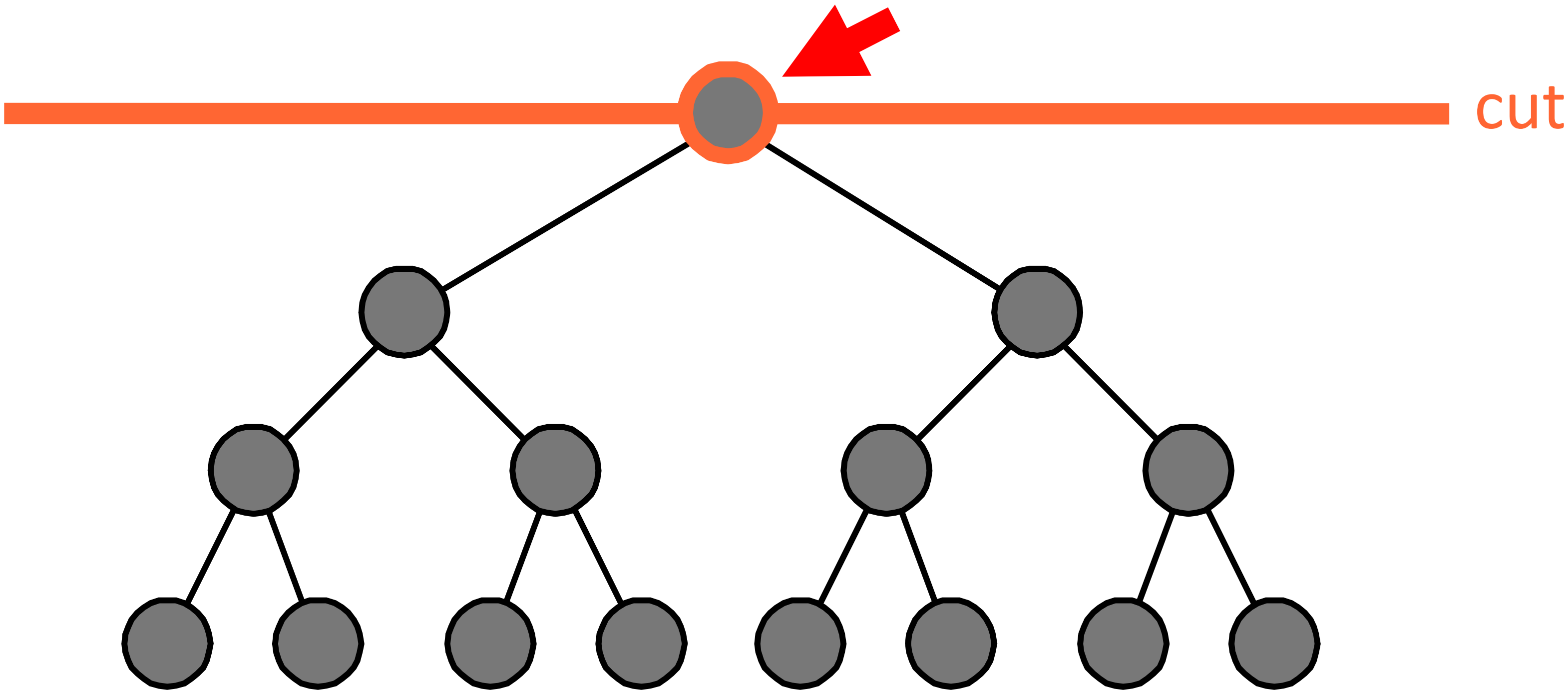
- start with coarse cut (root-node)



Lightcuts

Cut-Selection Algorithm

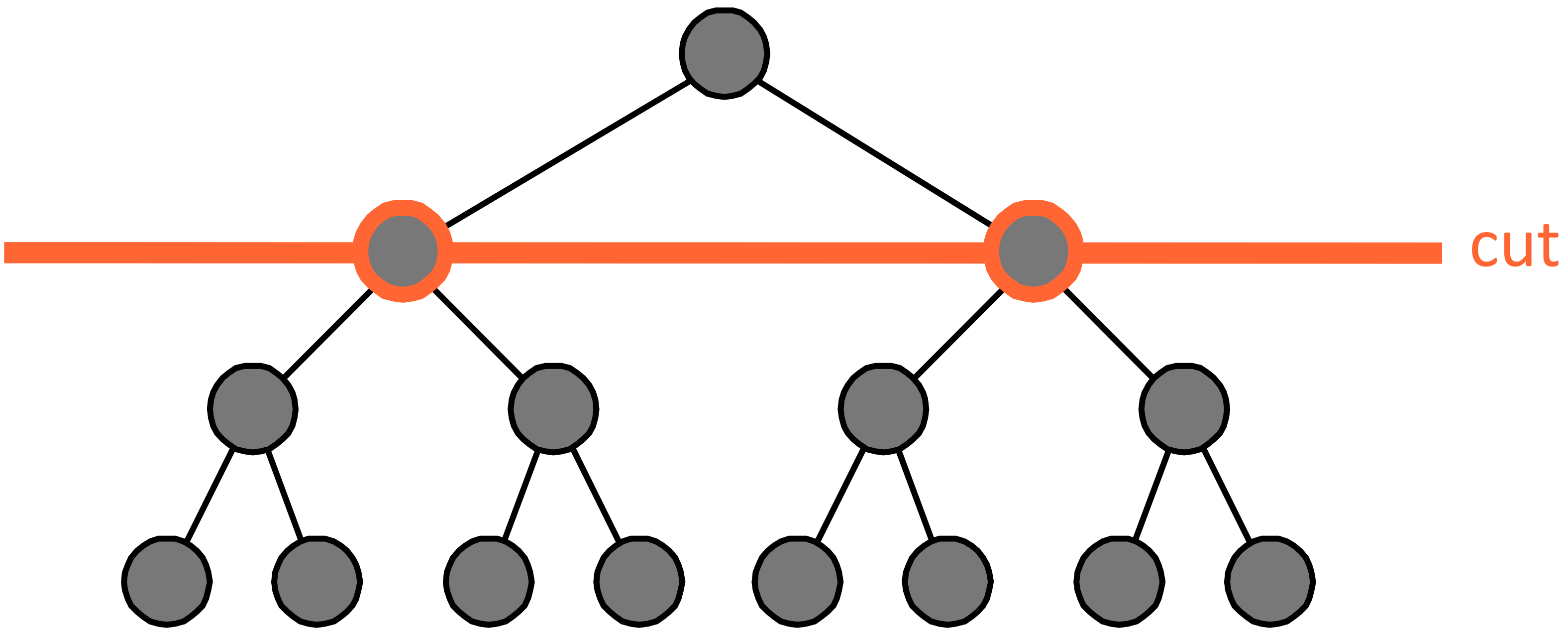
- select cluster with largest error-bound



Lightcuts

Cut-Selection Algorithm

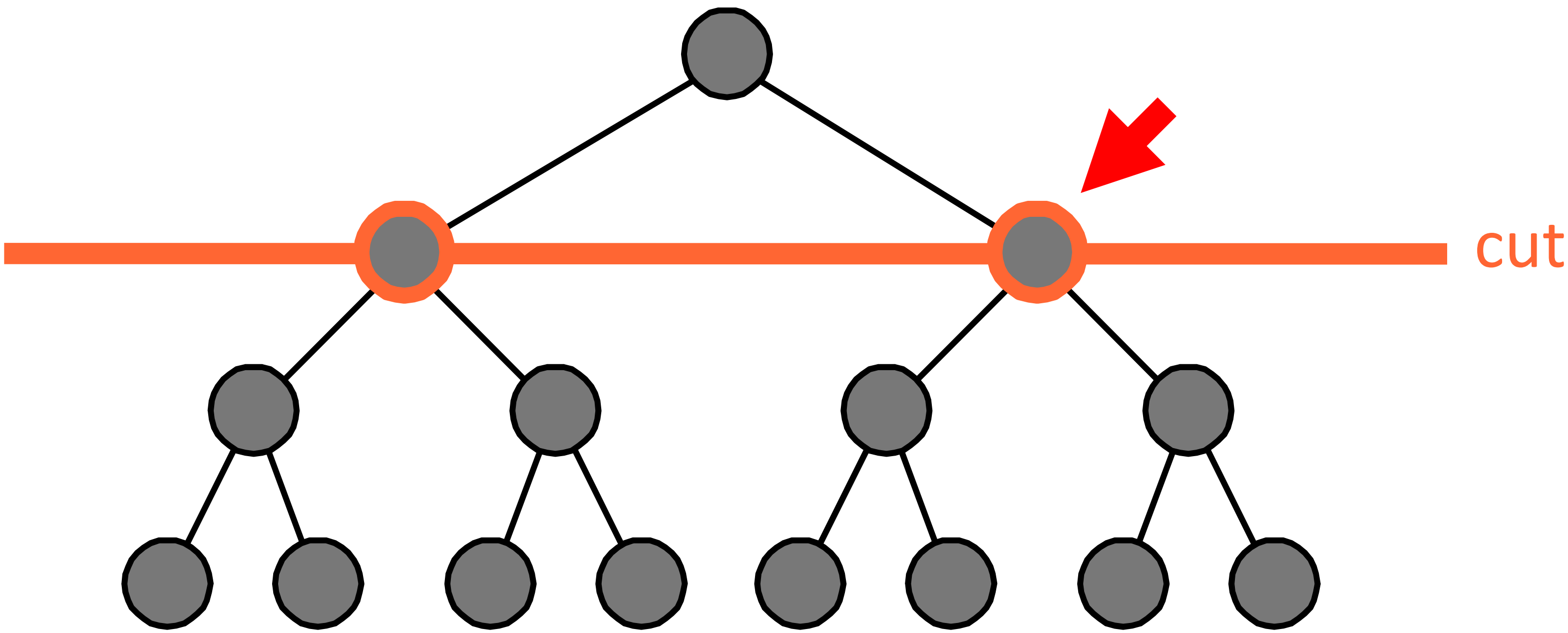
- refine if error bound > 2% of total
- see [Walter et al. 2005] how to compute cluster estimate and error bound (remember: this also includes BRDFs)



Lightcuts

Cut-Selection Algorithm

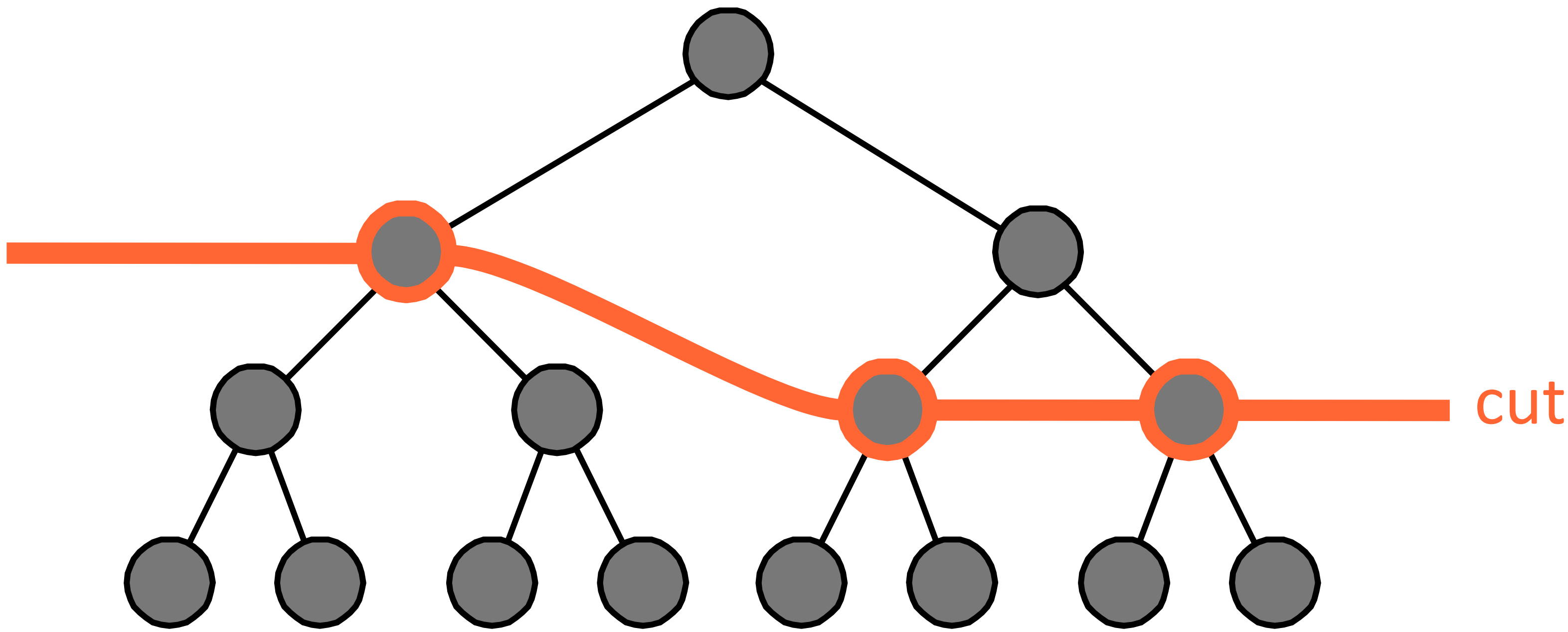
- (again) select cluster with largest error-bound



Lightcuts

Cut-Selection Algorithm

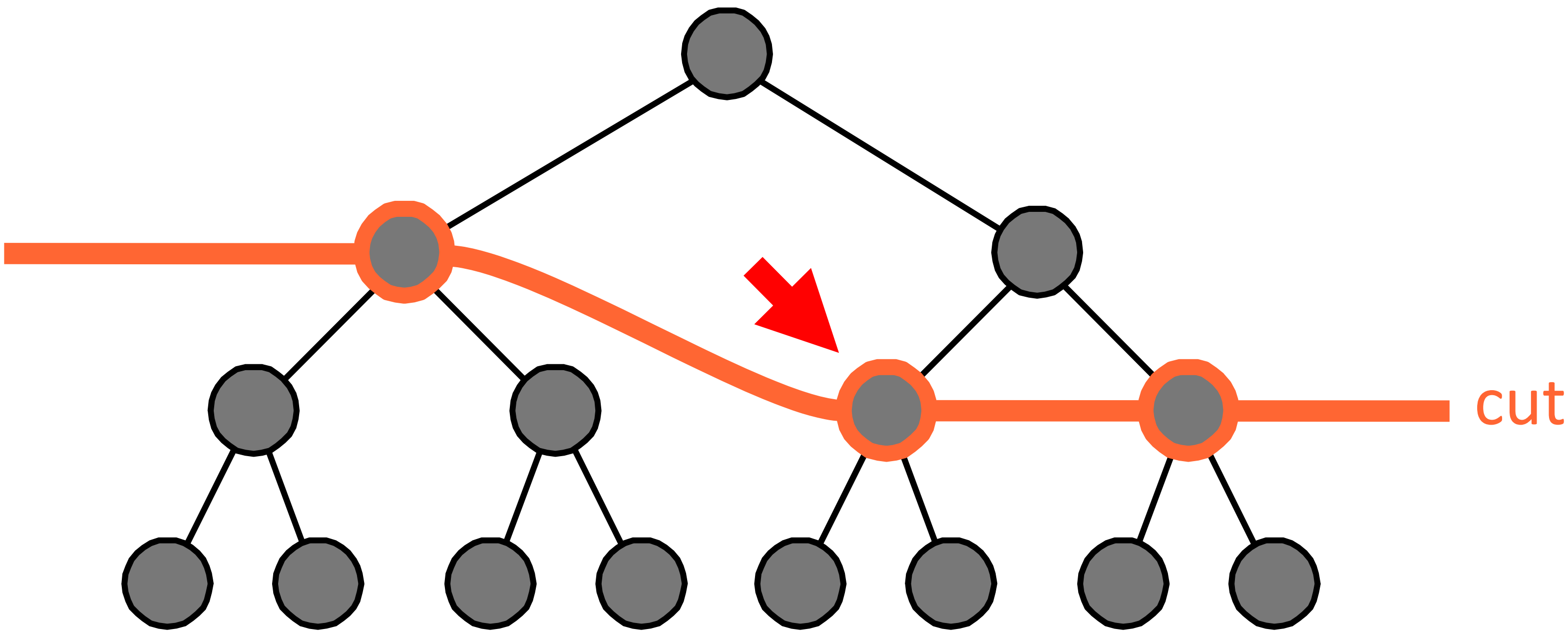
- … and refine if its error bound is above threshold …



Lightcuts

Cut-Selection Algorithm

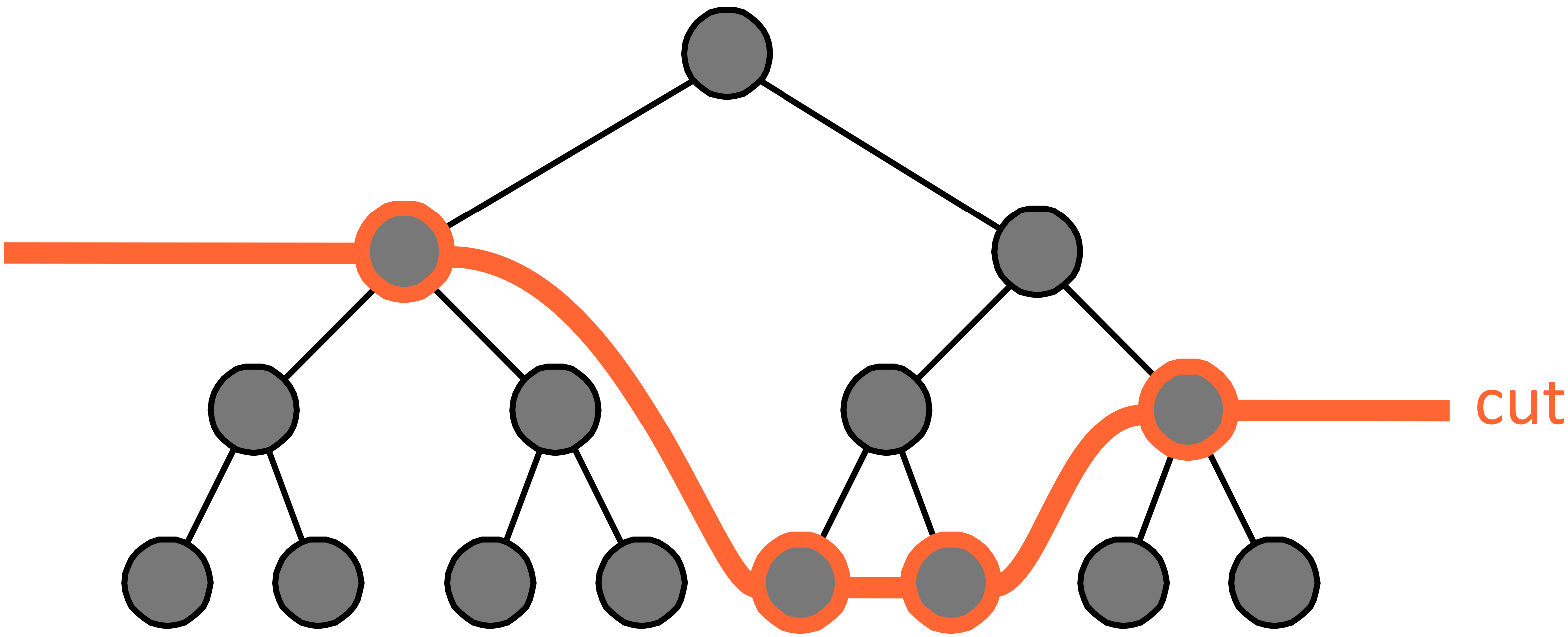
- … and so on …



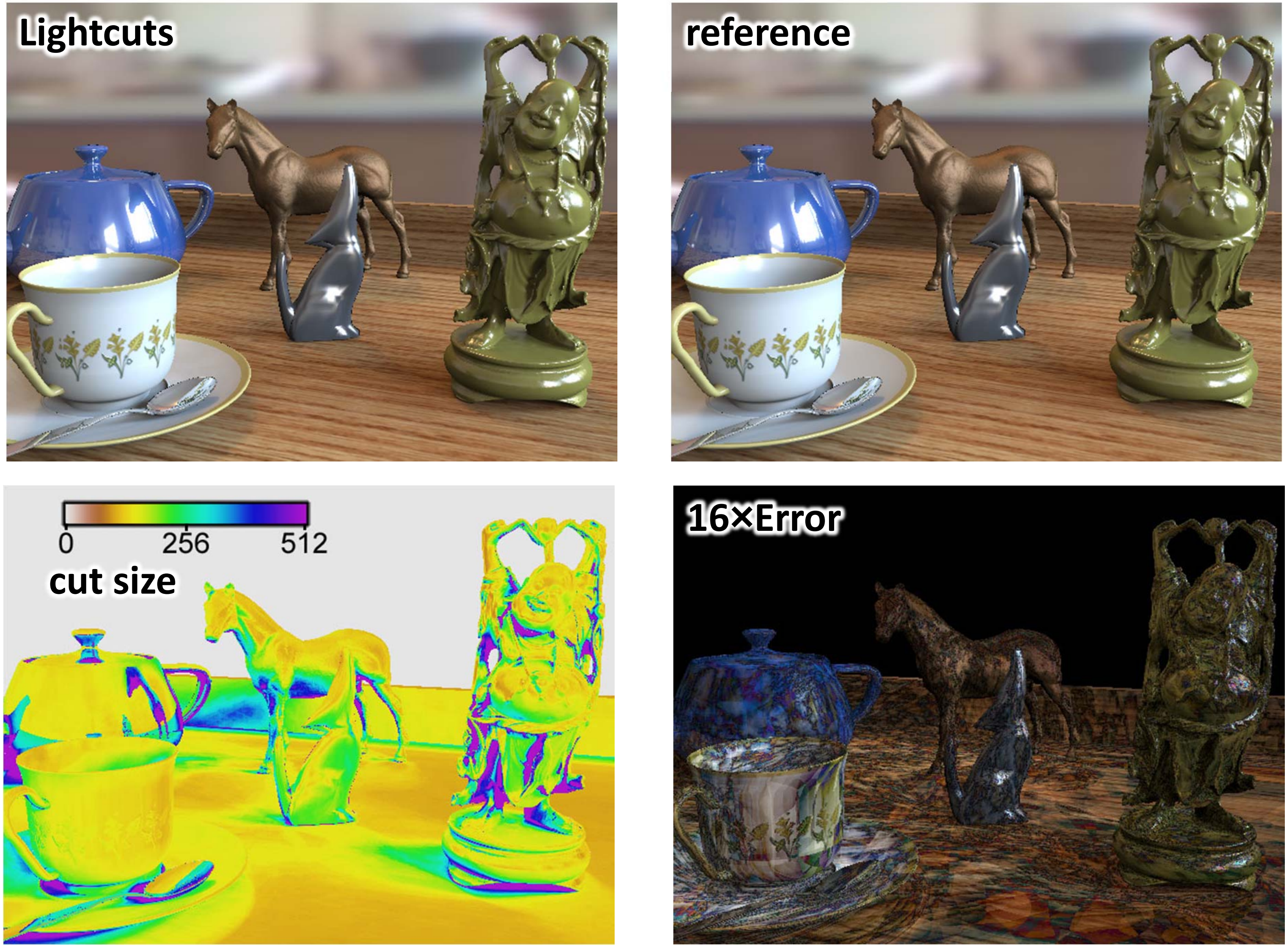
Lightcuts

Cut-Selection Algorithm

- … repeat until the entire cut obeys 2% threshold



Lightcuts - Results



Tableau, 630K polygons, 13 000 lights, (env map + indirect)

Lightcuts

Summary

- 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

- … anything else?

Multidimensional Lightcuts

A pixel is more than a point…

- motion blur

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

Multidimensional Lightcuts

A pixel is more than a point…

- motion blur
- participating media

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

Multidimensional Lightcuts

A pixel is more than a point…

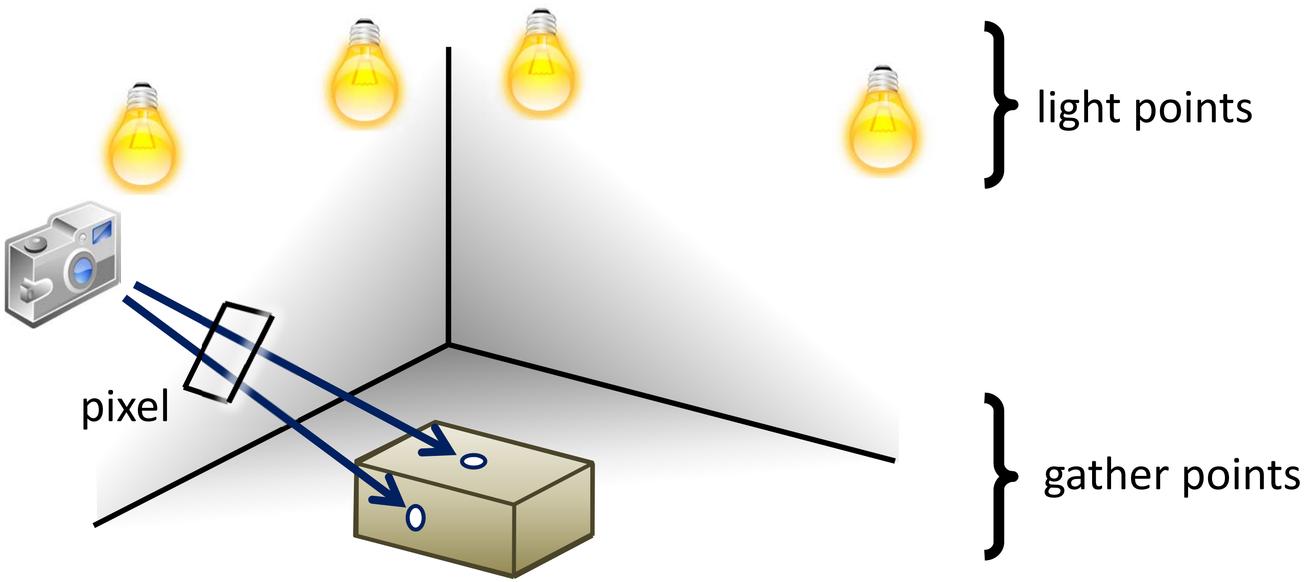
- motion blur
- participating media
- depth of field

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

Multidimensional Lightcuts

Concept

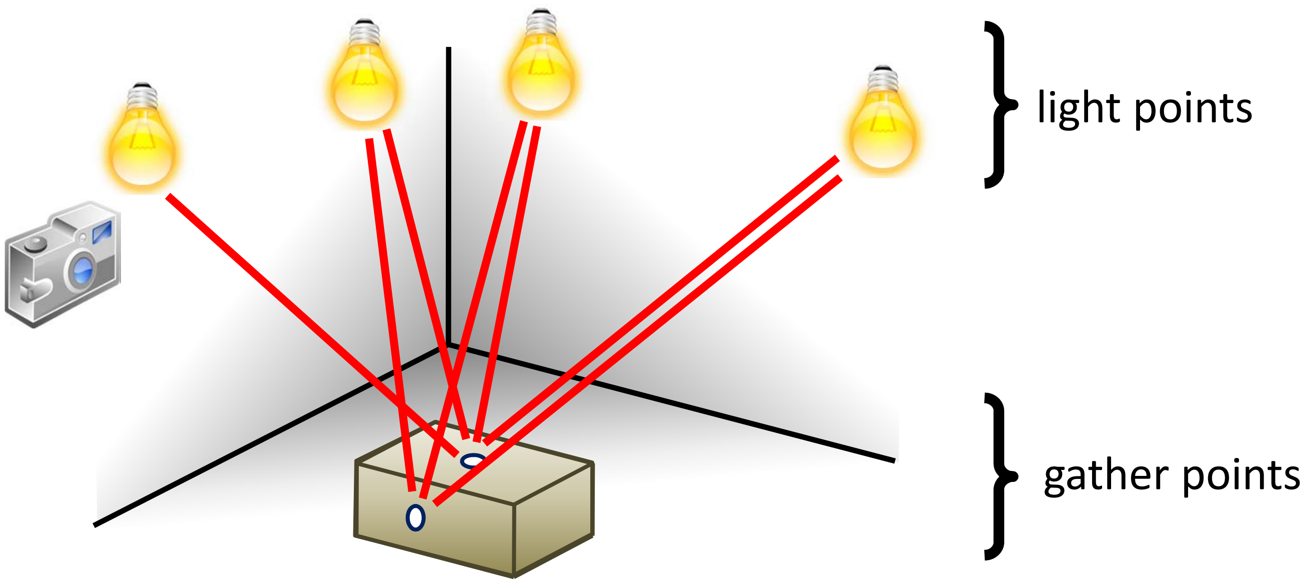
- discretize full integral into 2 point sets
  - light points (L)
  - gather points (G)



Multidimensional Lightcuts

Concept

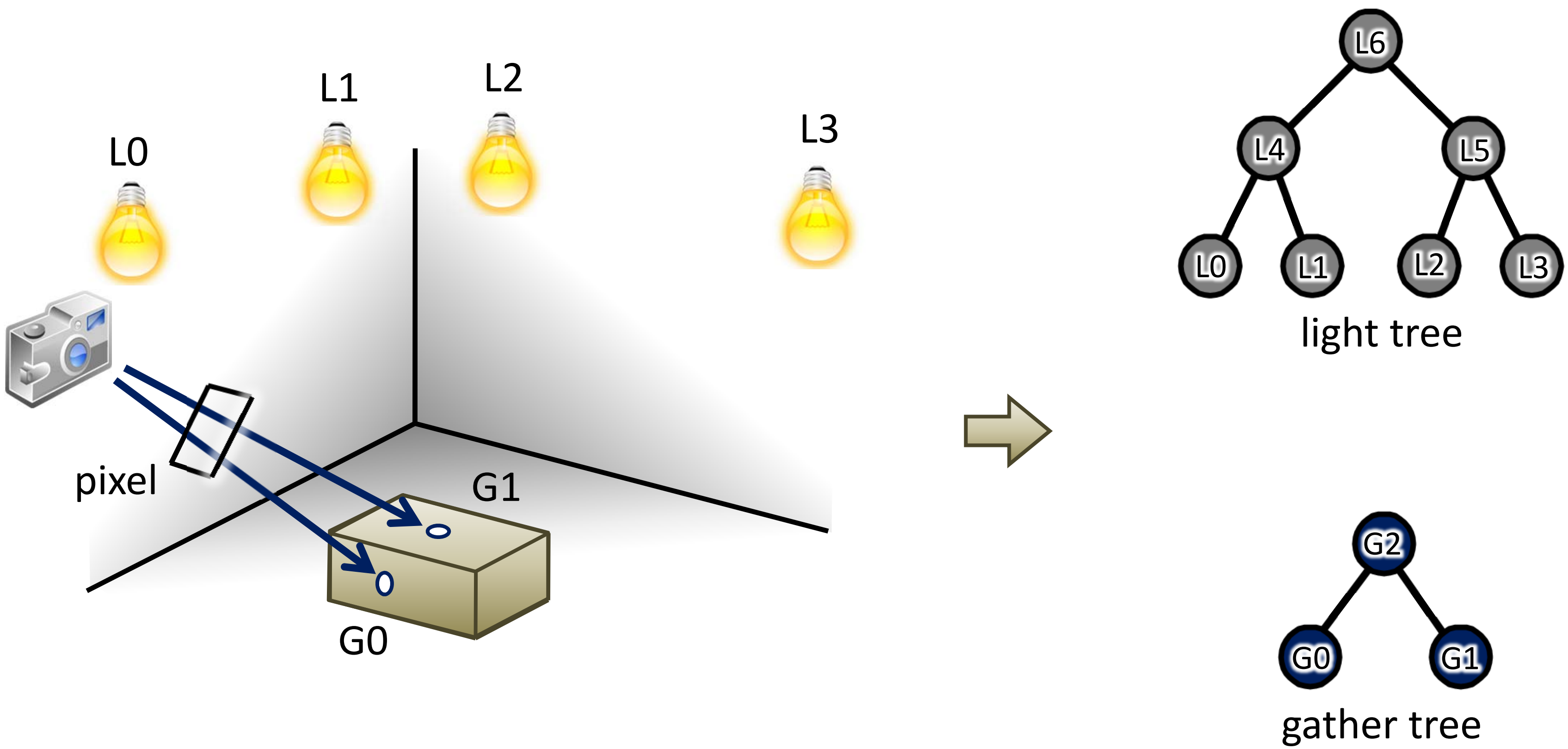
- discretize full integral into 2 point sets
  - light points (L)
  - gather points (G)



Multidimensional Lightcuts

Concept

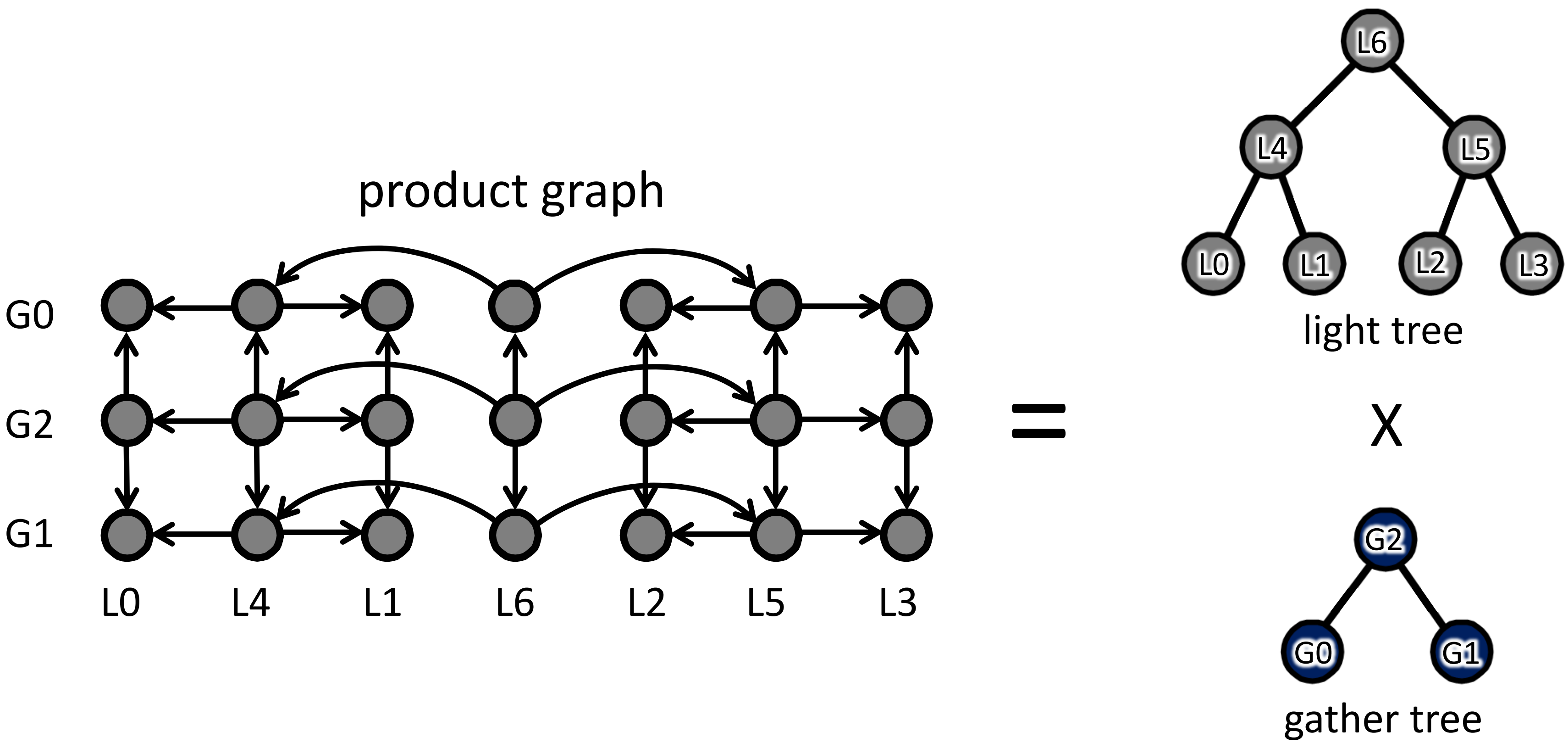
- cluster light and gather points into 2 trees



Multidimensional Lightcuts

Concept

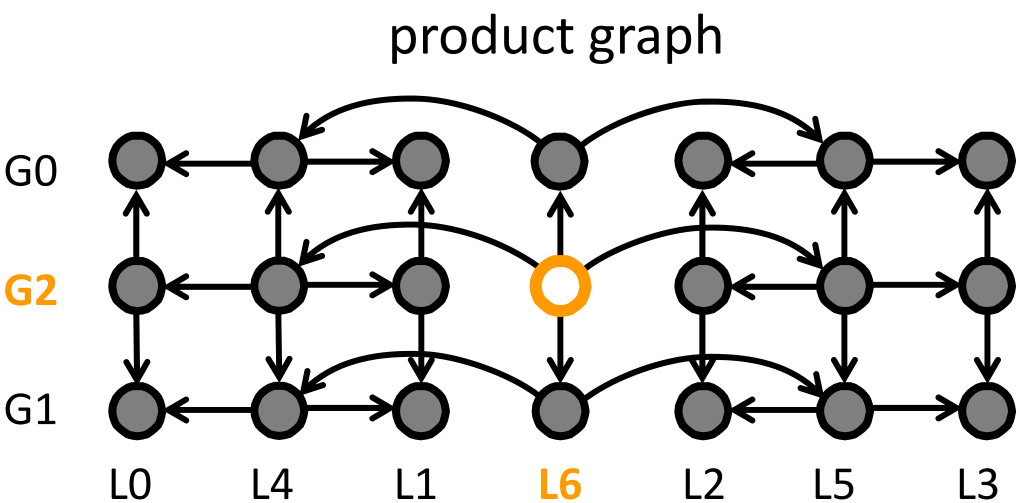
- product graph: hierarchy over the set of all gather-light pairs (never stored explicitly)

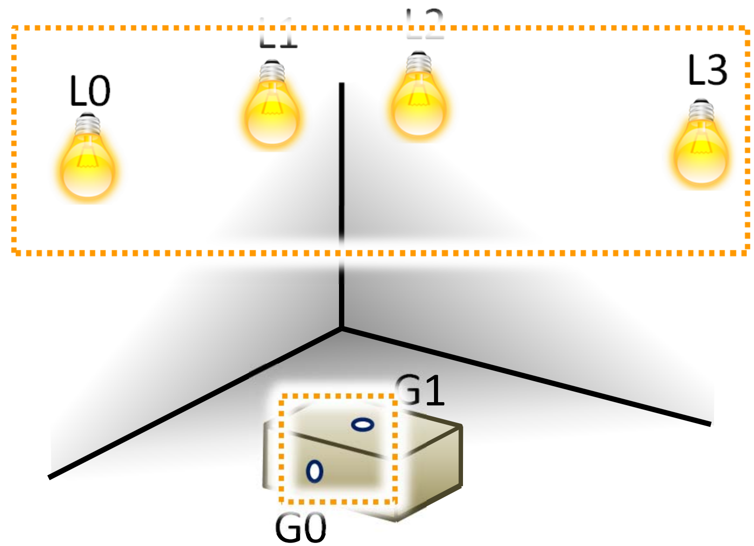


Multidimensional Lightcuts

Concept

- product graph: hierarchy over the set of all gather-light pairs (never stored explicitly)

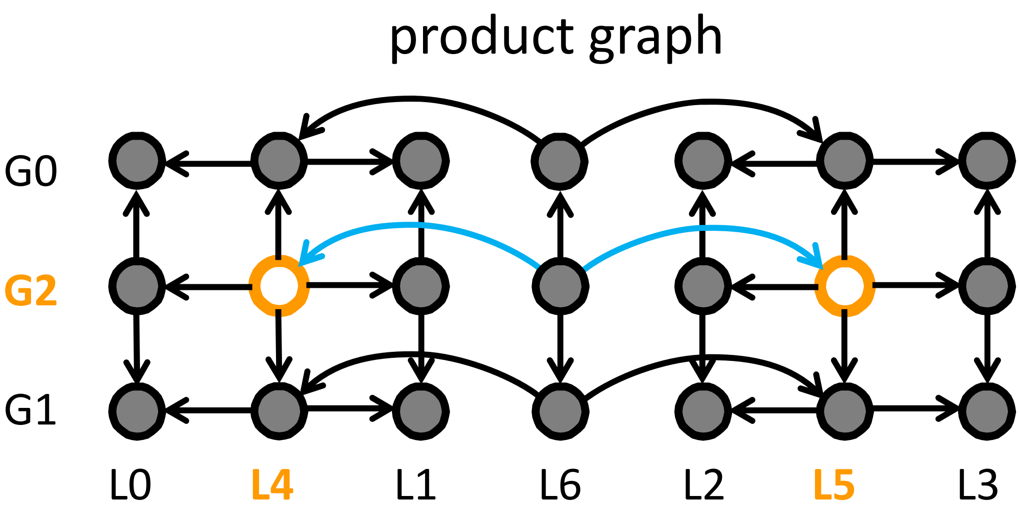


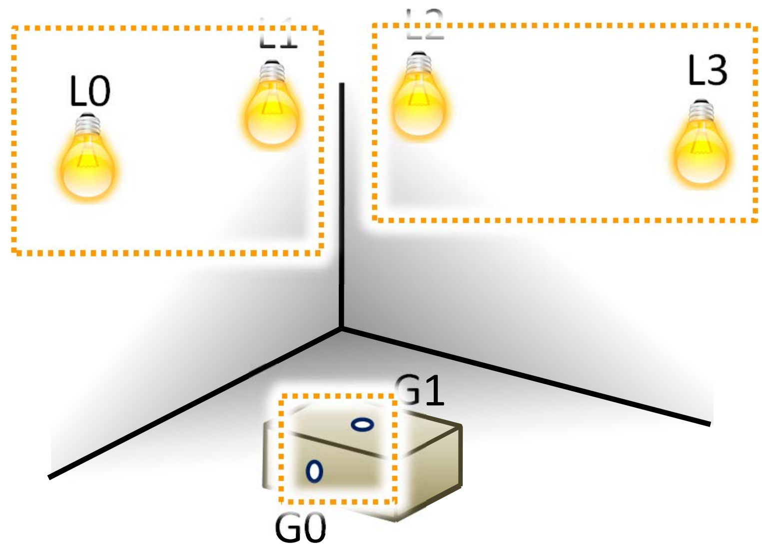


Multidimensional Lightcuts

Concept

- product graph: hierarchy over the set of all gather-light pairs (never stored explicitly)

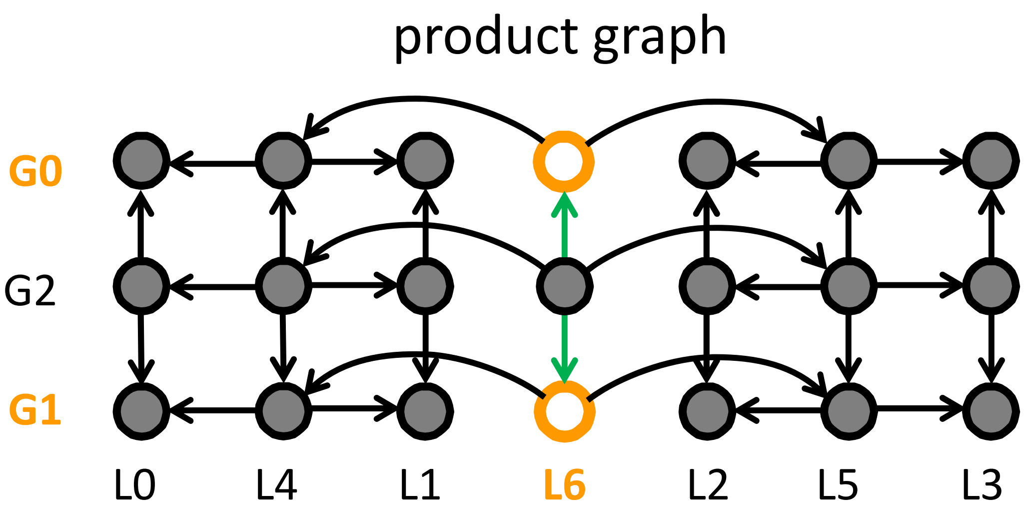


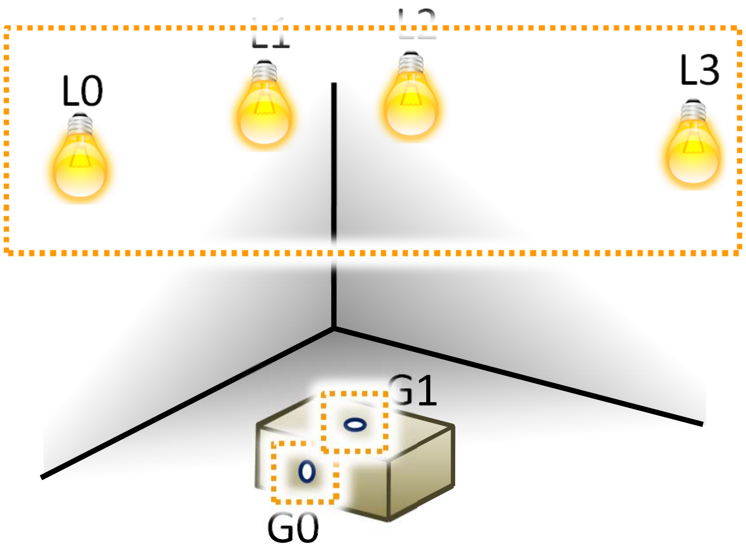


Multidimensional Lightcuts

Concept

- product graph: hierarchy over the set of all gather-light pairs (never stored explicitly)





Multidimensional Lightcuts

Algorithm Overview

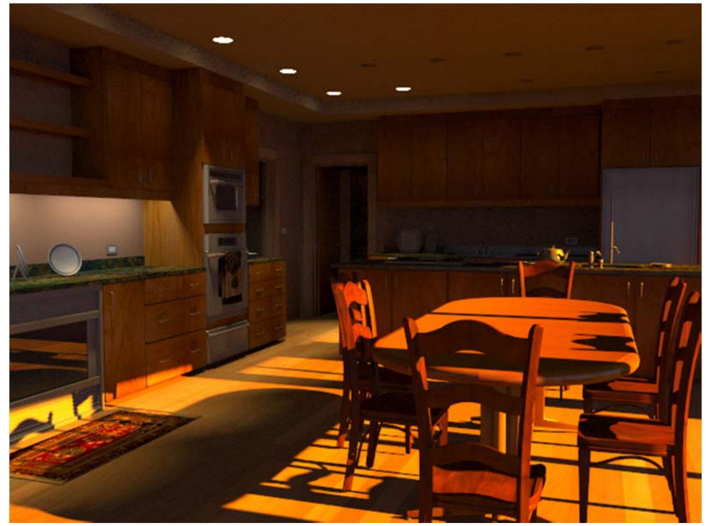
- once per image
  - create lights and light tree
- for each pixel
  - create gather points and gather tree for pixel
  - adaptively refine clusters in product graph until all cluster errors < perceptual metric (please see the paper for details)

Multidimensional Lightcuts

Results

Direct only (relative cost 1x)

Direct+Indirect (1.3x)



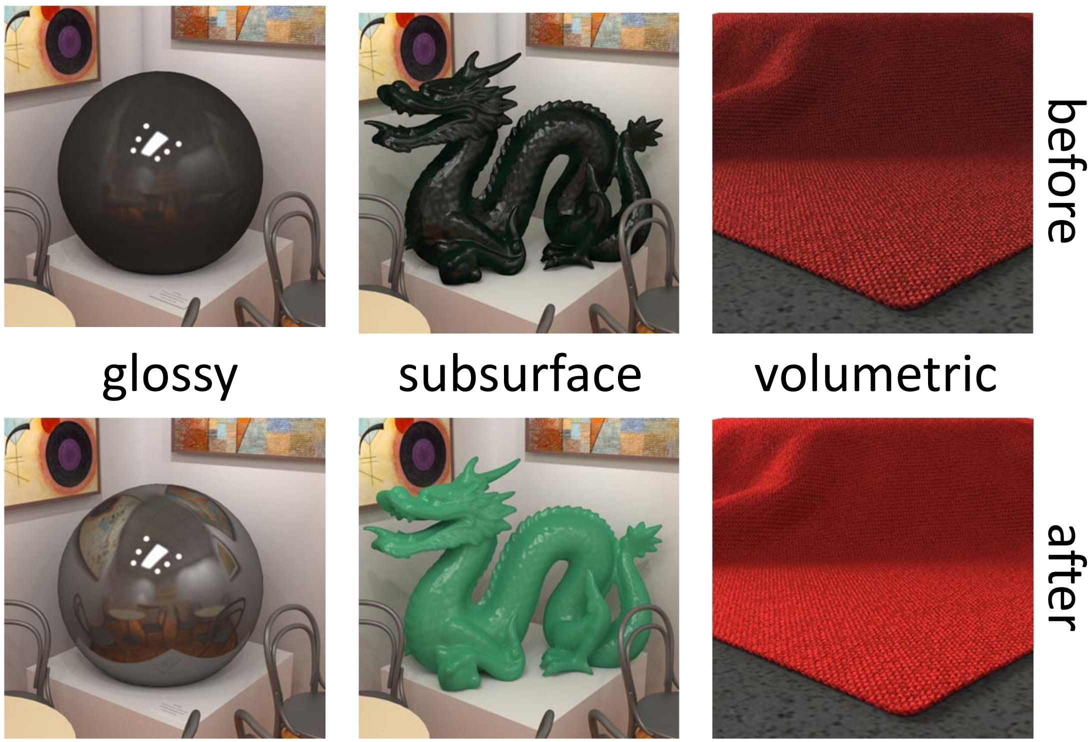
Direct+Indirect+Volume (1.8x)

Direct+Indirect+Volume+Motion (2.2x)

Bidirectional Lightcuts

Bidirectional Lightcuts

- handles more effects including glossy reflections, subsurface, short-range indirect illumination
- bidirectional formulation and a set of weighting strategies to reduce the bias in VPL-based rendering



Bidirectional Lightcuts, Walter et al., SIGGRAPH 2012

Scalability

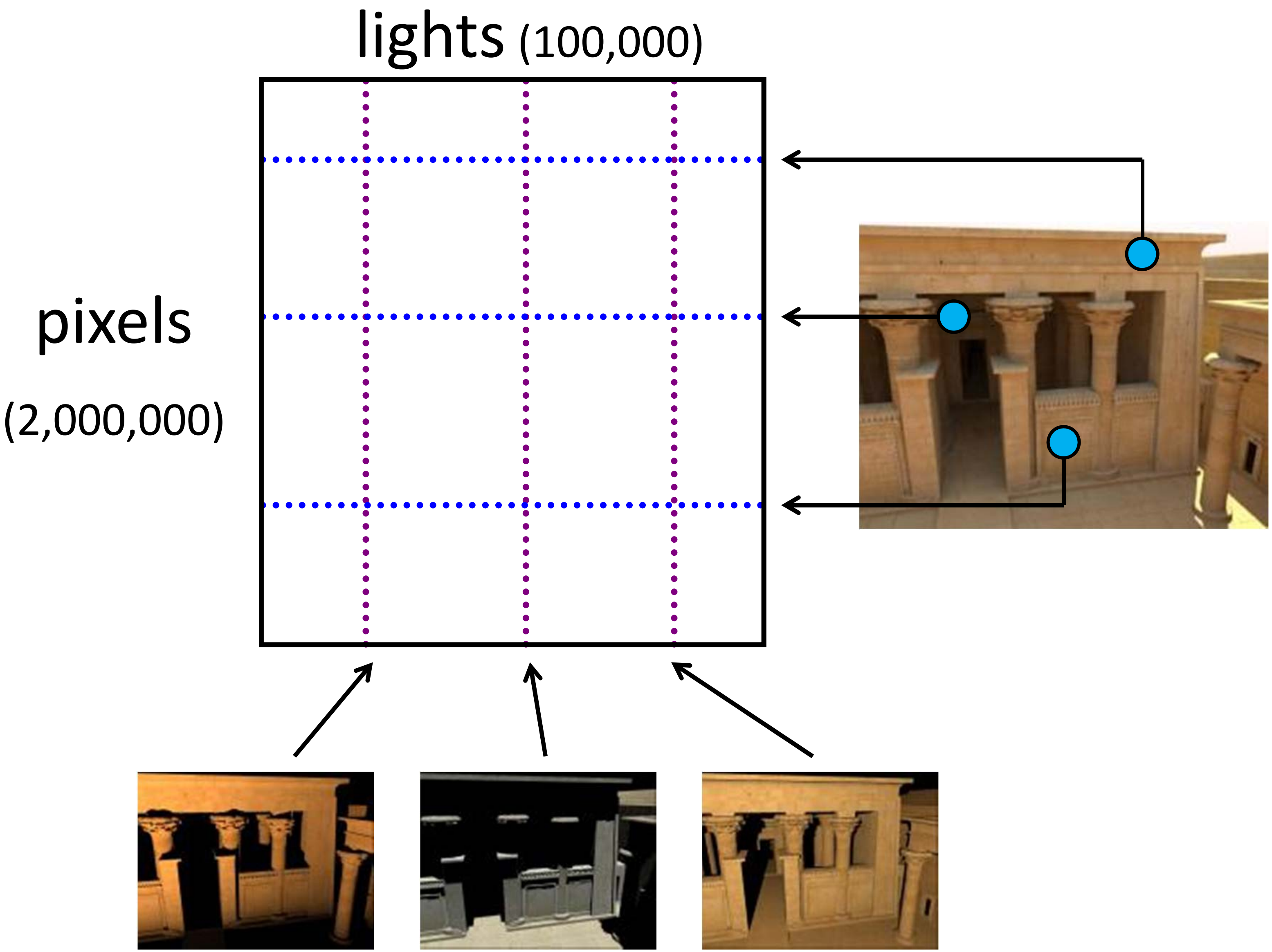
Alternatives to Lightcuts

- Matrix Row-Column Sampling [Hasan et al. 2007]
- Visibility Clustering

- potential advantages
  - shadow mapping instead of ray tracing
  - simpler to implement
  - no bounds on BRDFs required
  - faster in occluded environments

Matrix Row-Column Sampling

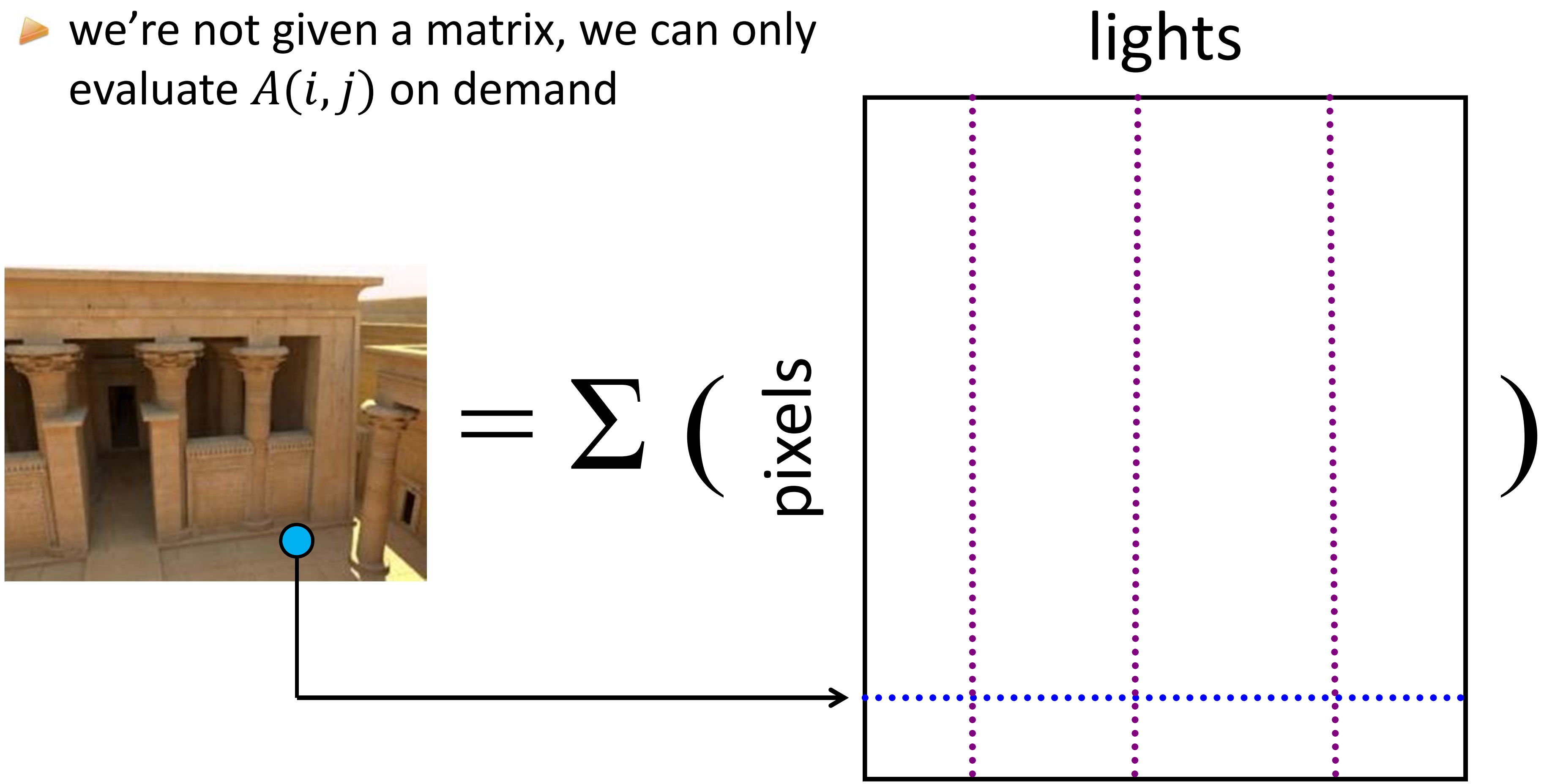
Matrix Interpretation



Matrix Row-Column Sampling

Problem Statement

- pixel colors: compute sum of columns
- we're not given a matrix, we can only evaluate $A(i, j)$ on demand

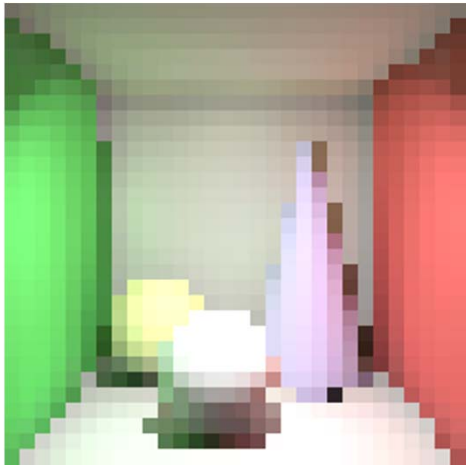


Matrix Row-Column Sampling

Problem Statement

- … such matrices are highly structured

643 lights



A simple scene

30 x 30 image

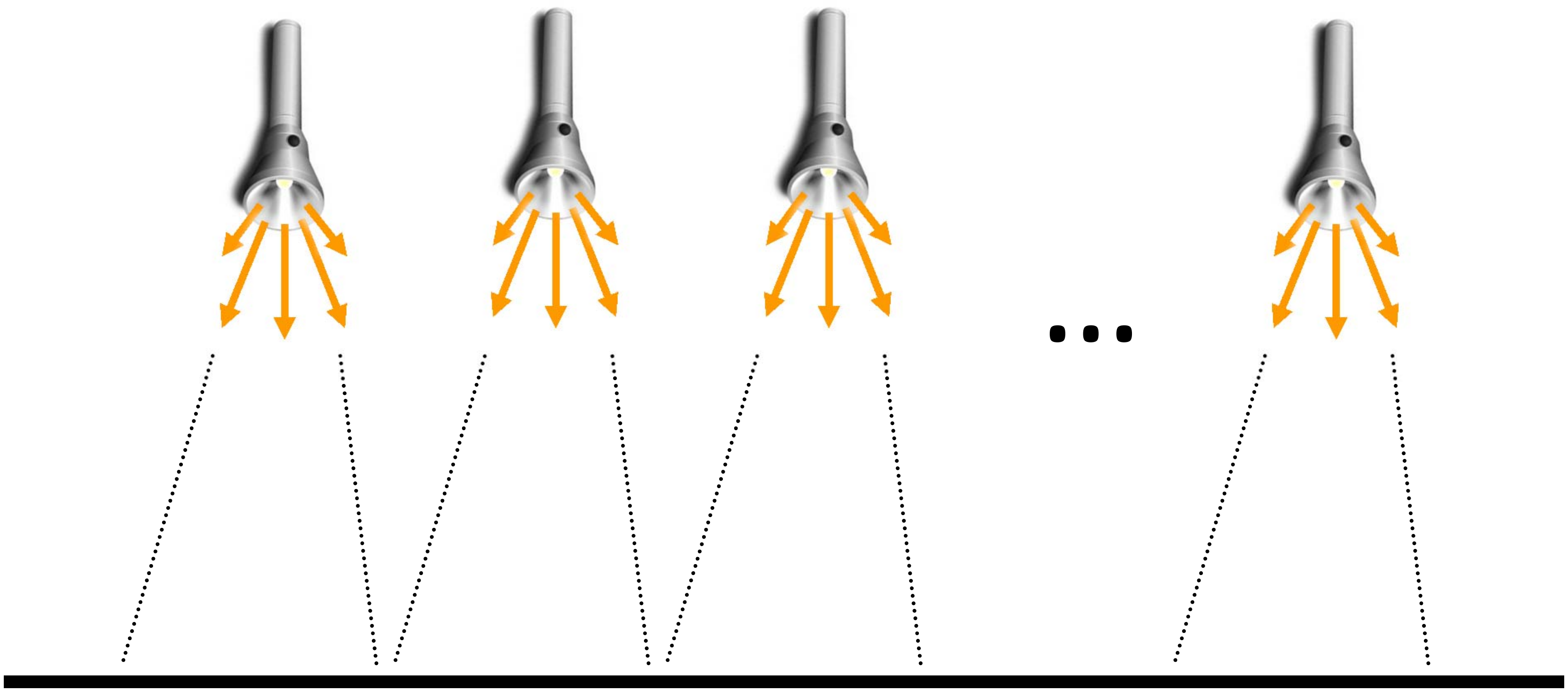
900 pixels

The matrix

Matrix Row-Column Sampling

Low Rank Assumption Violation

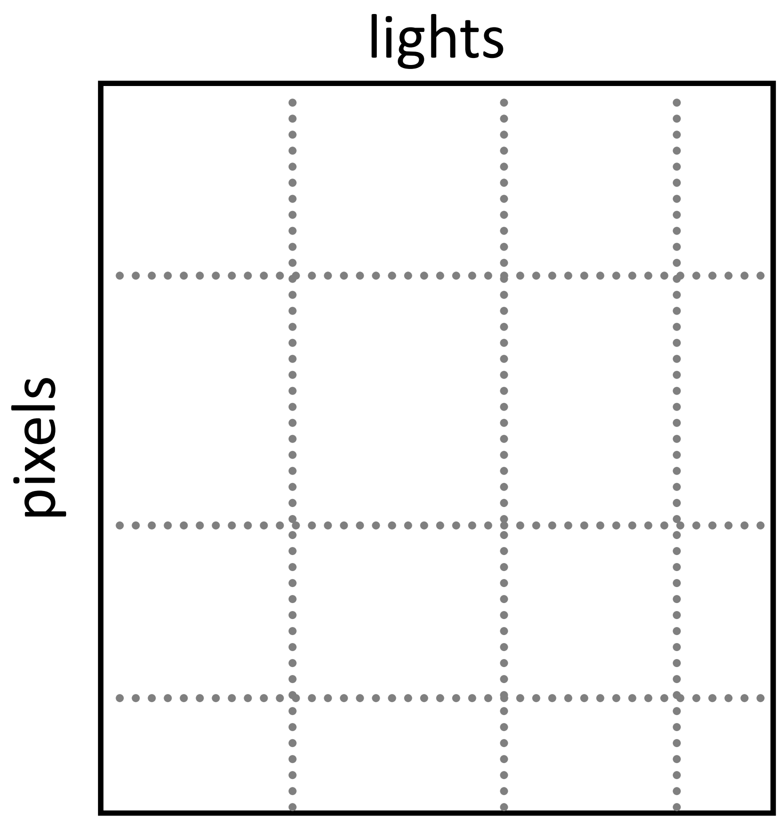
- bad case: lights with very local contribution



Matrix Row-Column Sampling

Matrix Interpretation

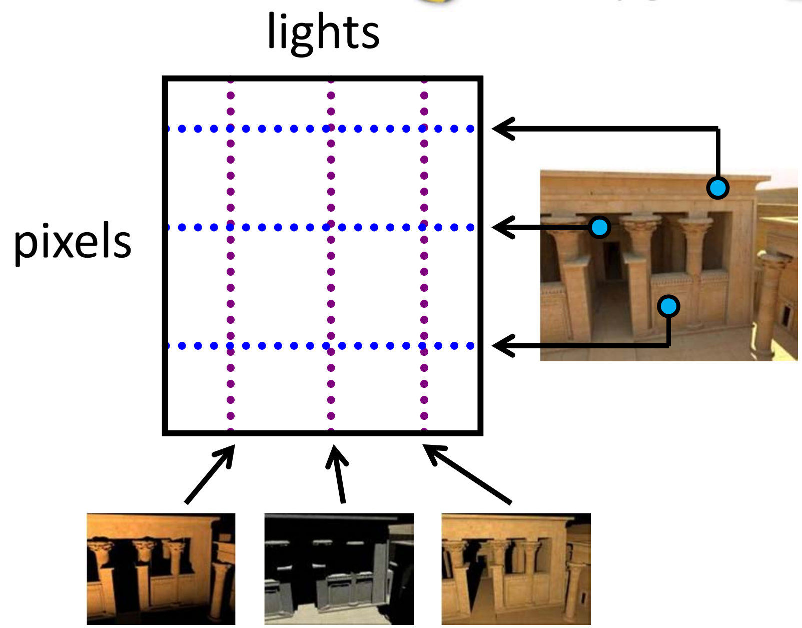
- sample a subset of matrix elements
- sampling patterns do matter
  - point-to-point visibility: raytracing
  - point-to-many-points visibility: shadow mapping

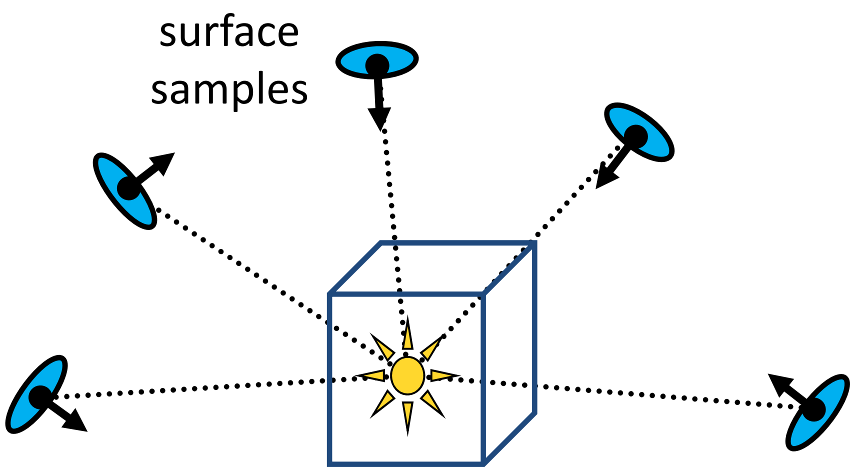


Matrix Row-Column Sampling

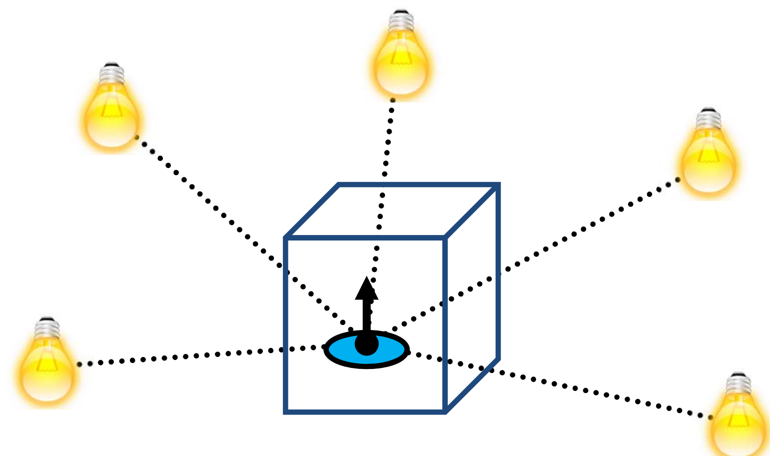
Row-Column Shadow Duality

- columns: regular shadow mapping
- rows: also shadow mapping





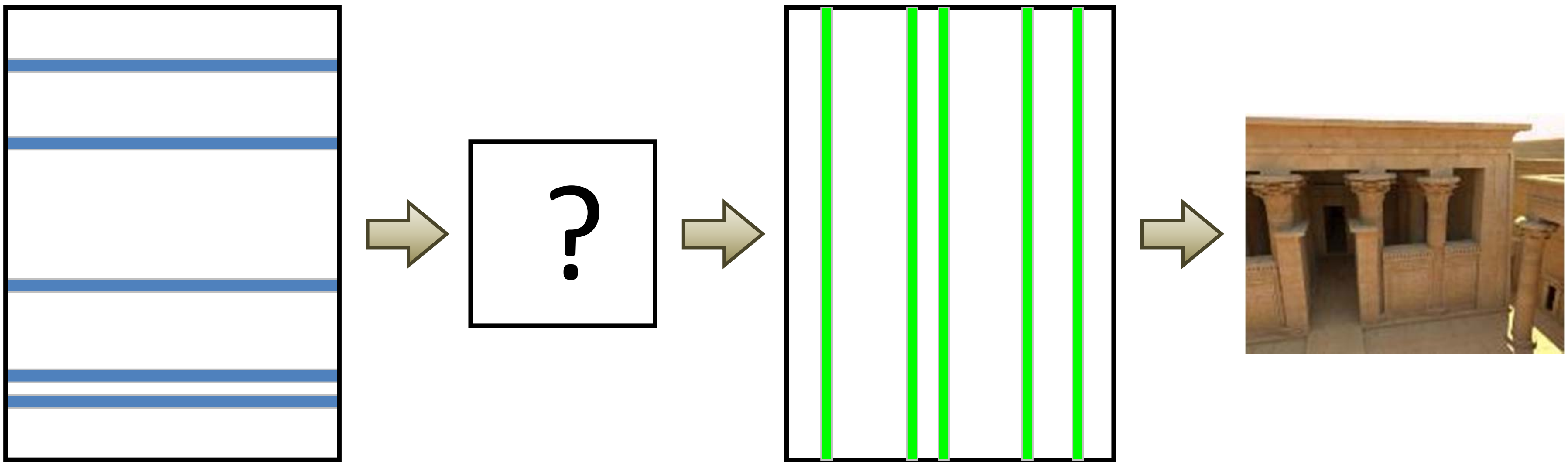
shadow map at
light position



shadow map at
sample position

Matrix Row-Column Sampling

Exploration and Exploitation



compute rows (explore)

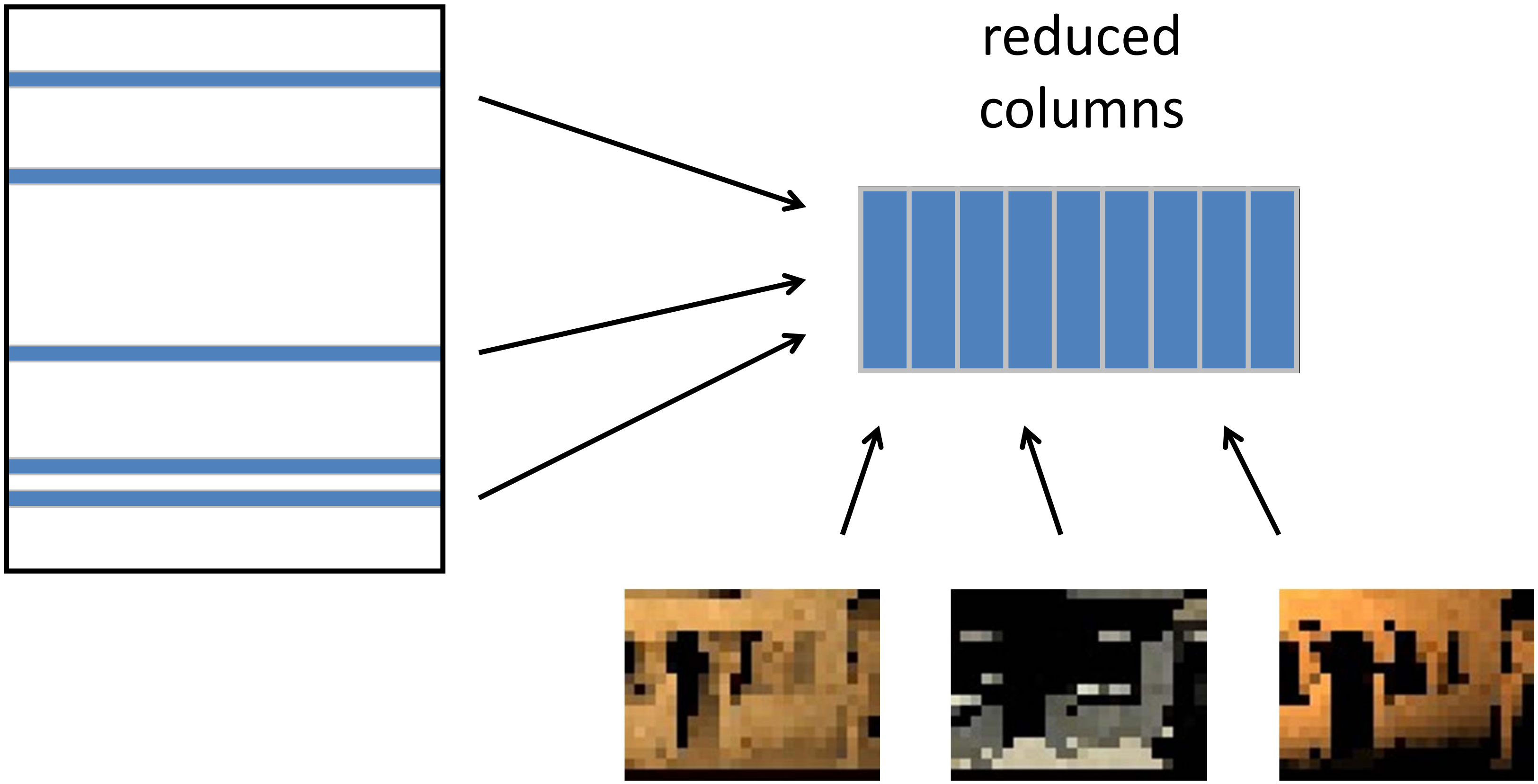
how to choose columns and weights?

compute columns (exploit)

weighted sum

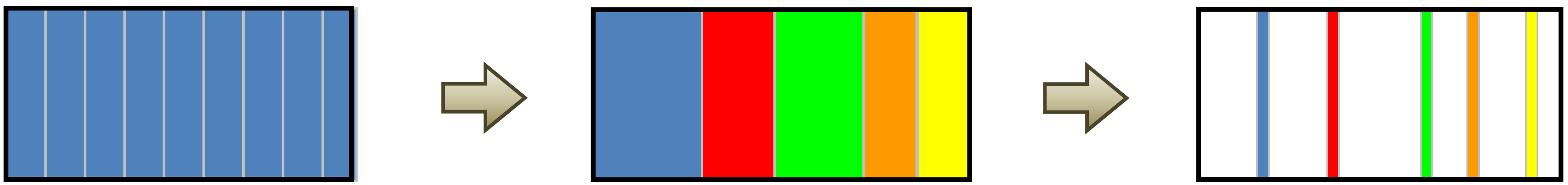
Matrix Row-Column Sampling

Reduced Matrix



Matrix Row-Column Sampling

Clustering Approach



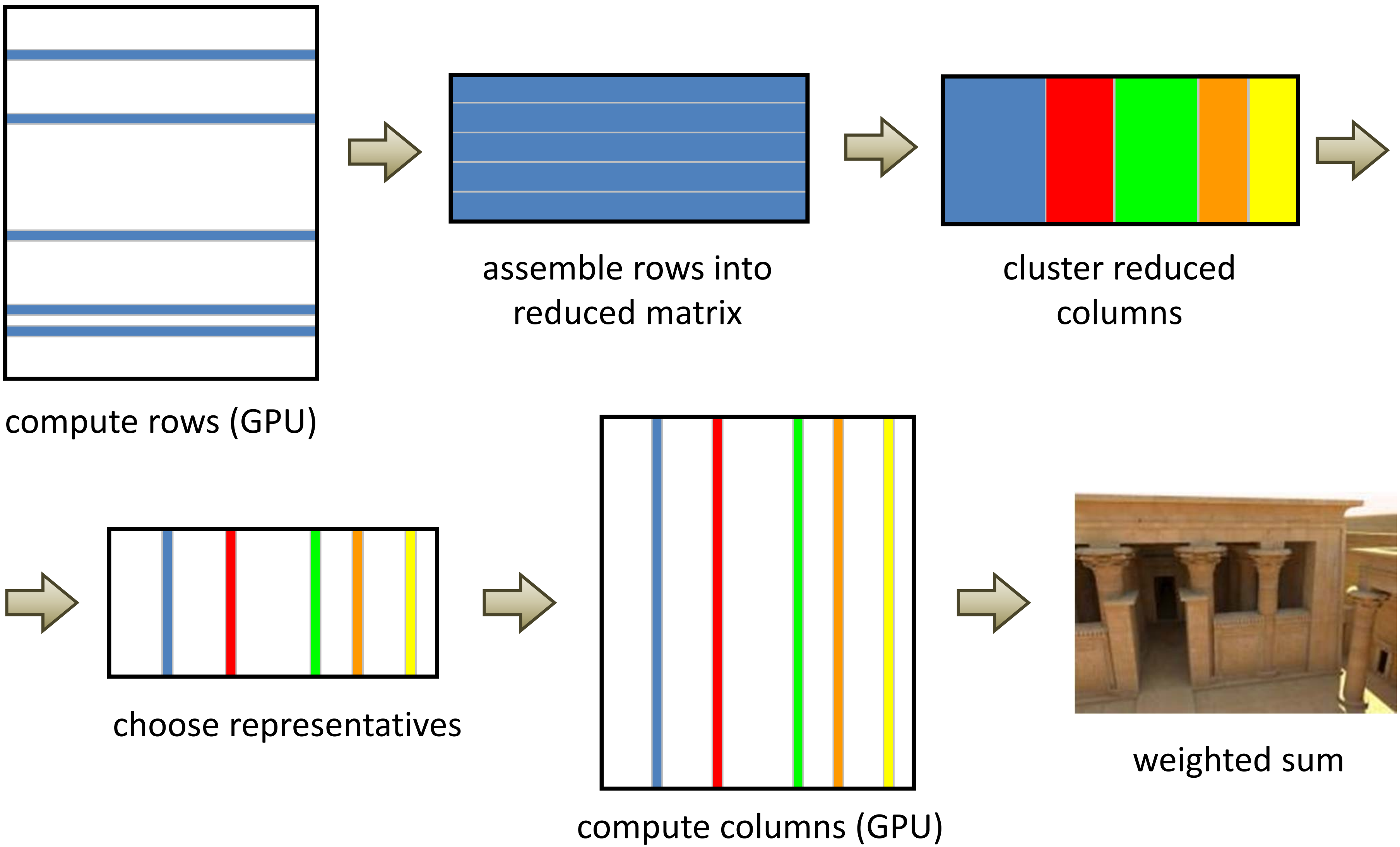
reduced columns

choose *k* clusters

choose representative columns

Matrix Row-Column Sampling

Algorithm Overview



Matrix Row-Column Sampling

Results: Temple

- 2.1m polygons
- mostly indirect and sky illumination
- indirect shadows

MRCS: 16.9 sec
(300 rows + 900 columns)

Reference: 20 min
(using all 100k lights)

Matrix Row-Column Sampling

Results: Trees and Bunny

- complex incoherent geometry
- low rank, not low frequency

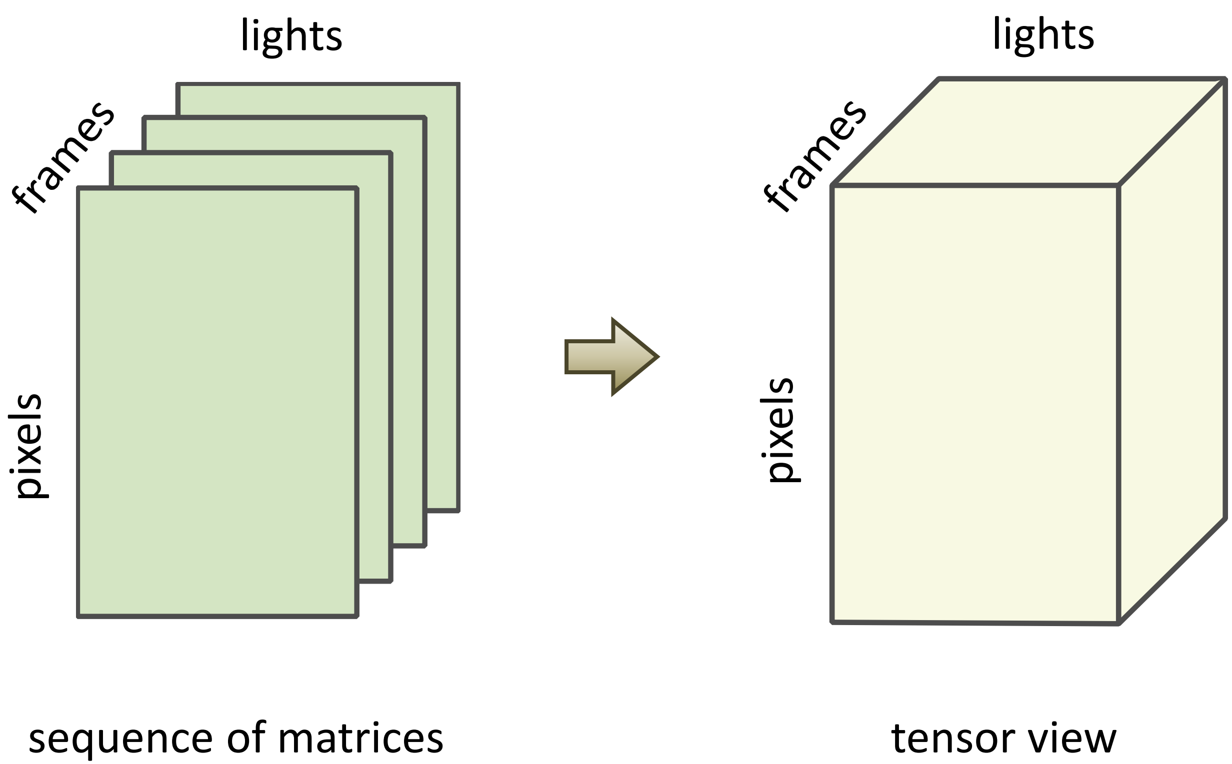
MRCS: 2.9 sec
(100 rows + 200 columns)

MRCS: 3.8 sec
(100 rows + 200 columns)

Matrix Row-Column Sampling

Tensor Clustering for Animated Scenes

- sequence of matrices (one per frame) can be seen as one **large** tensor

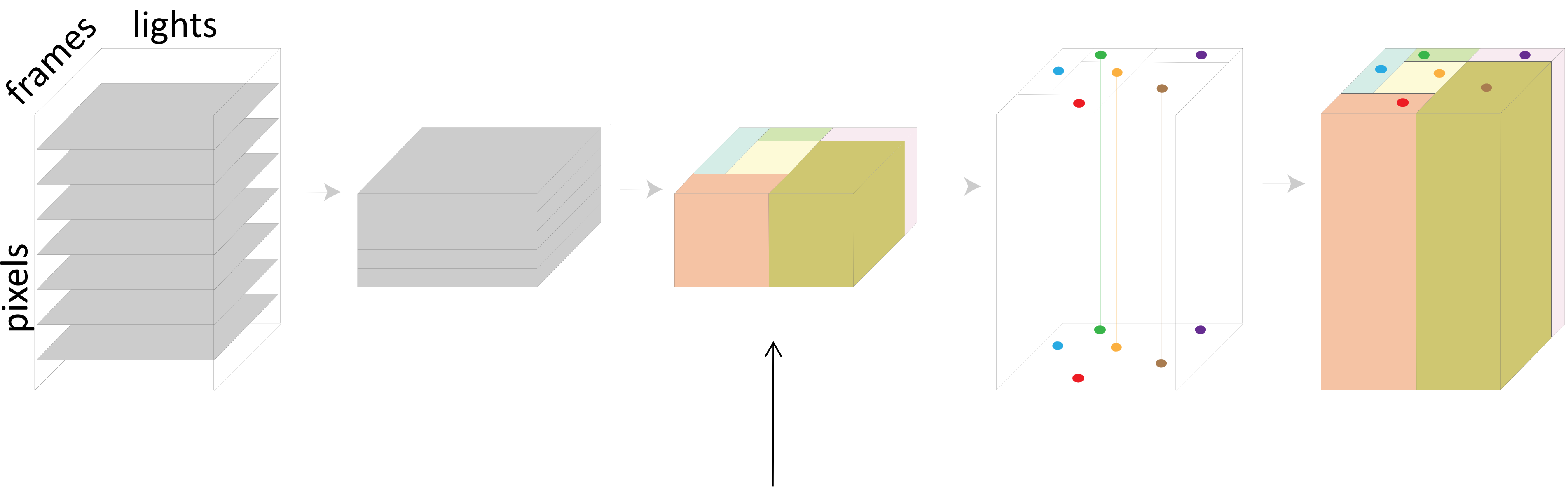


Matrix Row-Column Sampling

Tensor Clustering for Animated Scenes

- no details here!

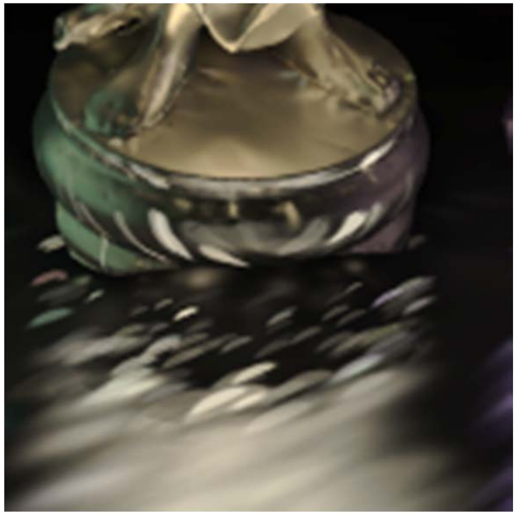
Sample slices | Reduced tensor | Cluster reduced columns | Compute representatives | Reconstruct full tensor



Rectangular clustering

More Clustering Strategies

- LightSlice [Ou and Pellacini 2011]
  - compute initial clustering
  - refine it differently in different “slices”
  - use neighboring slices to get more rows

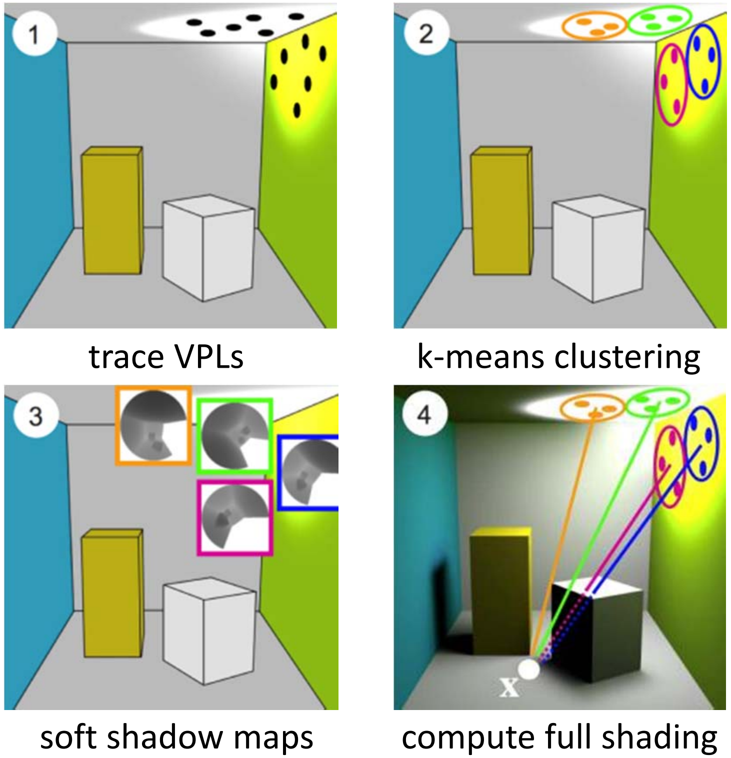


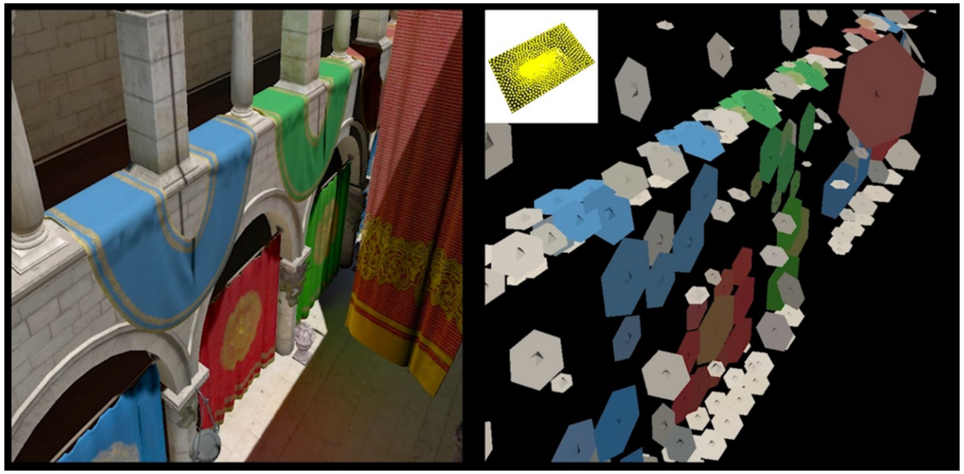
- Visibility Clustering [Davidovič et al. 2010] (already in Jan’s part)
  - separate shading from visibility
  - for global lights:
    - cluster visibility
    - shade from more VPLs

More Clustering Strategies

- Clustered Visibility [Dong et al. 2009]
  - cluster VPLs
  - use soft shadow mapping
  - shade from all VPLs

- RSM Clustering [Prutkin et al. 2012]
  - bidirectional importance
  - temporally stable clustering
  - compute virtual disc lights





Overview

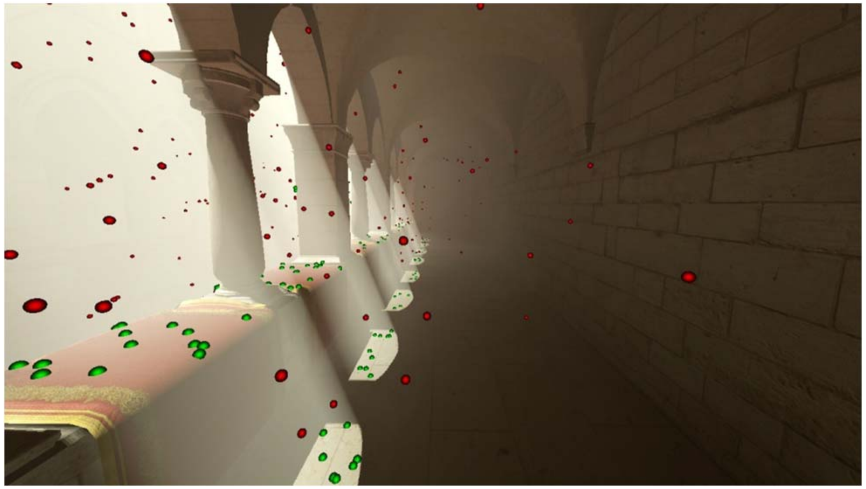
… follows the structure of the STAR

- **Introduction & Welcome** (Carsten)
- **Many-Light Rendering Concepts** (Jan)
  - **Basic Idea**
  - **Improved Virtual Lights Generation**
  - **Lighting with Virtual Lights**
- **Really Many Lights: Scalability** (Carsten)
- **Interactive and Real-Time Rendering** (Carsten)
- **Conclusions, Outlook, Q&A** (Jan & Carsten)

Real-time Many-light Rendering

Outline

- main difference to offline-methods is visibility computation
  - rasterization instead of raycasting
  - VPL generation
  - lighting and shadowing from VPLs

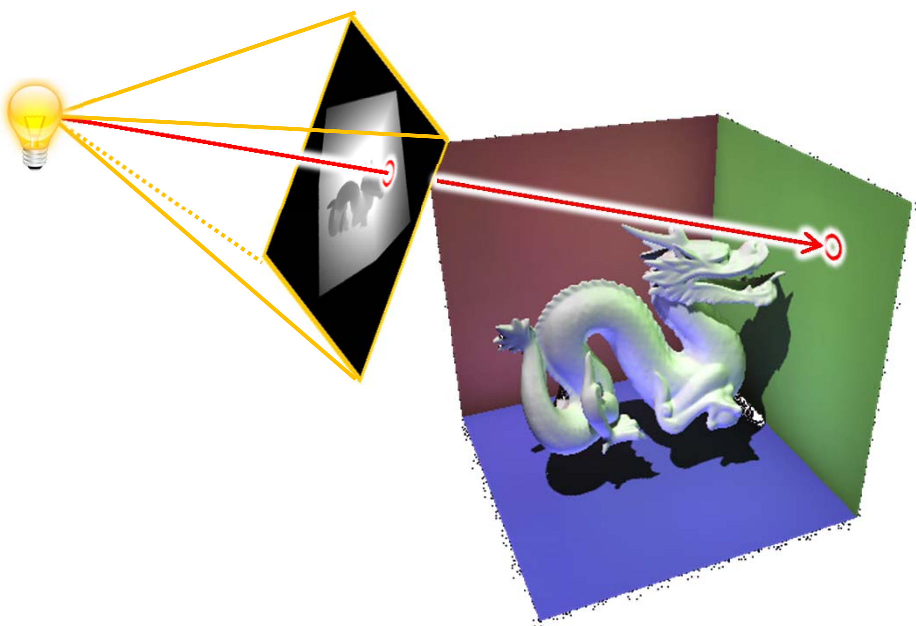


Real-time Many-light Rendering

Visibility Computation for VPL Generation

- real-time rendering ↔ mostly diffuse scenes ↔ relatively few VPLs (~10^3)
- if acceleration structure available use ray casting

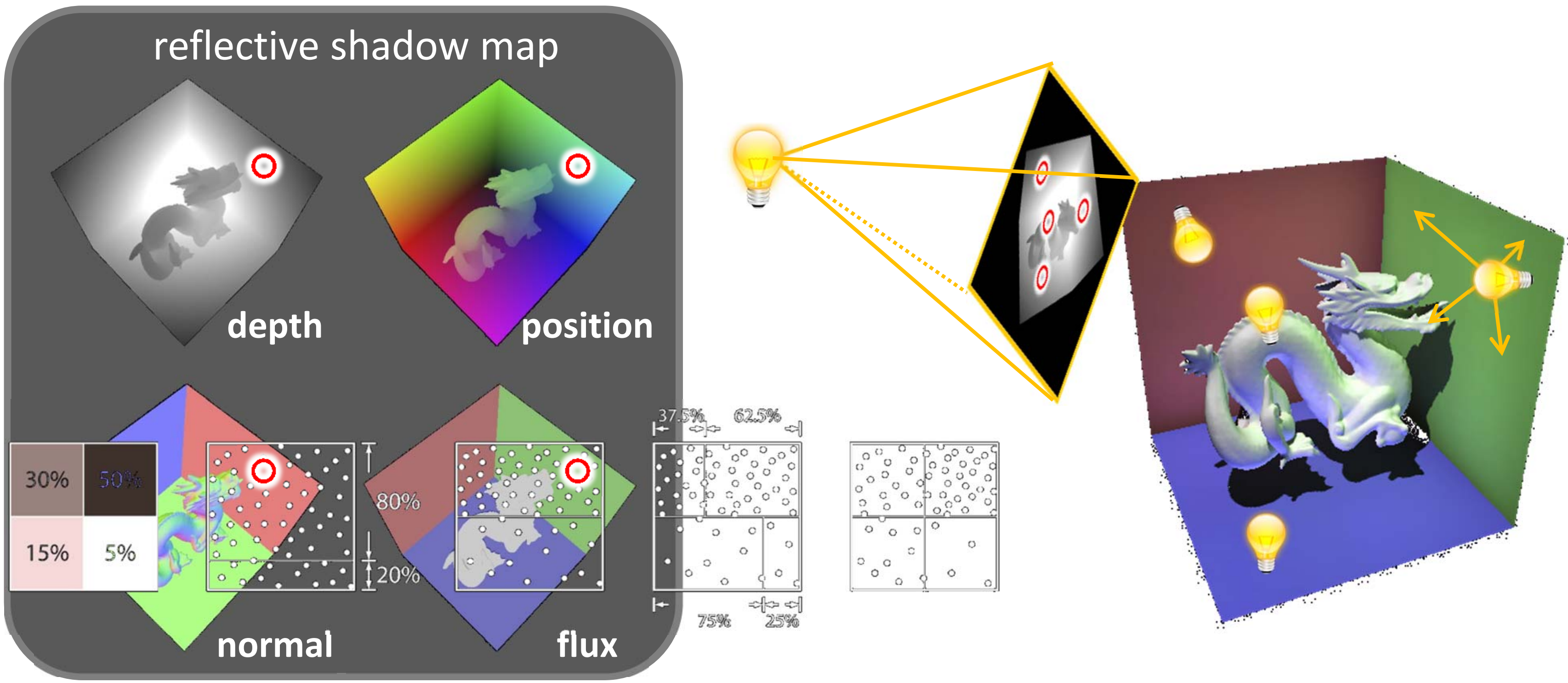
- VPL generation with rasterization
  - render scene from light
  - observation: visible surfaces = first intersection of light path



Real-time Many-light Rendering

VPL Generation with Rasterization

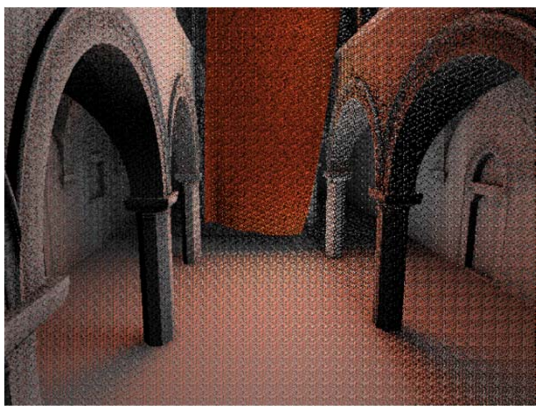
- render scene from light into reflective shadow map [DS05]: all information available for creating VPLs and continuing paths
  - single bounce indirect illumination by directly sampling the RSM
  - importance sampling can easily be added [DS06][REH*11]
- proceed recursively by rendering another RSM



Rendering with VPLs

Lighting and Shadowing

- many lights can be handled with deferred shading
  - interleaved sampling (problem: detailed normals/geometry) [Seg06]
  - hierarchical shading [NW10]
  - accumulate and filter incident light [SW09]
  - clustered deferred and forward shading [OBA12]

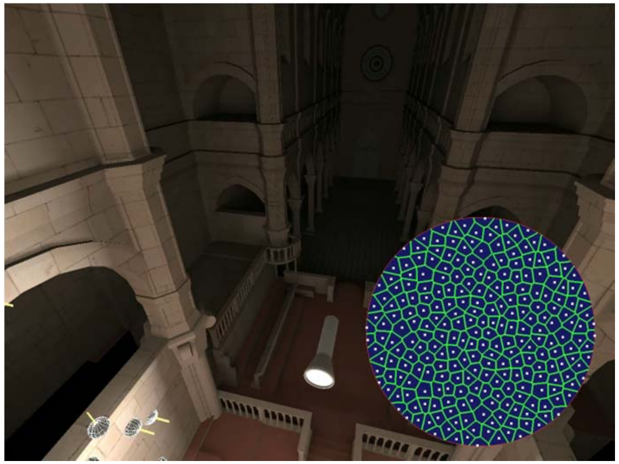


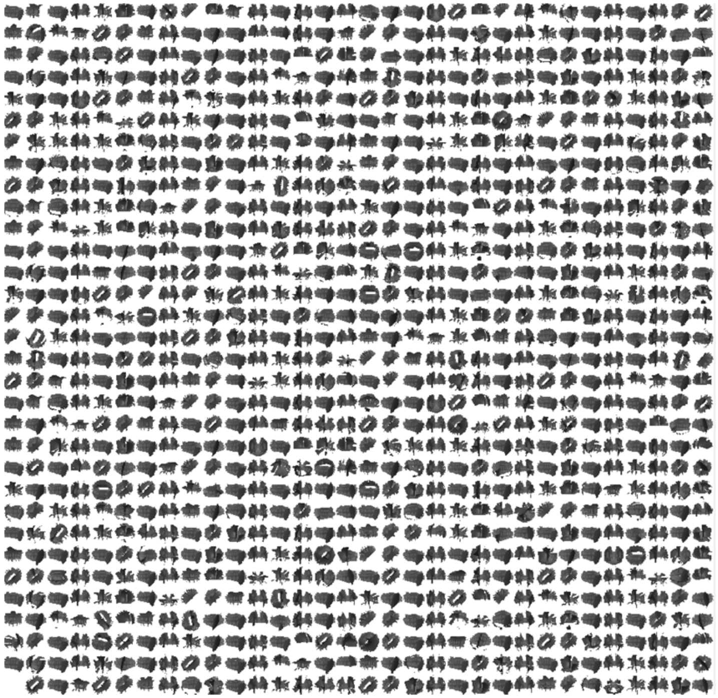

- **bottleneck: shadow computation**

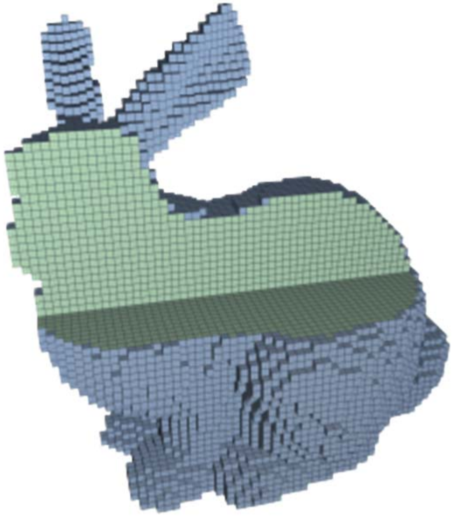
Rendering with VPLs

Shadow Computation

- …is the real bottleneck with instant radiosity / many lights methods
  - exploit temporal coherency [LSKLA07]
  - sampled visibility
    - voxelization, e.g. [SS10]
    - faster shadow maps



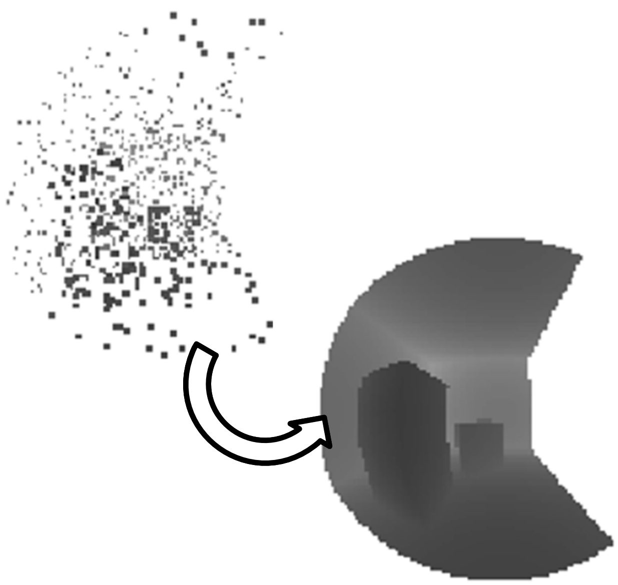


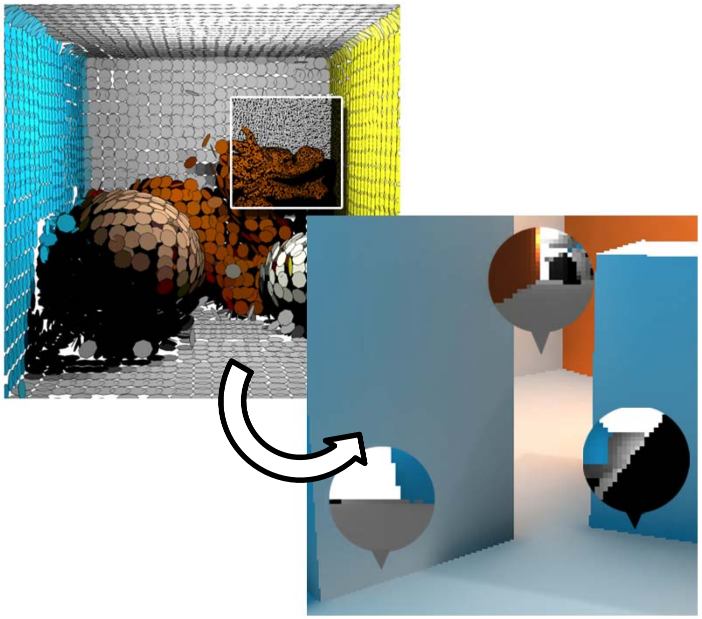


Shadow Mapping for VPLs

Problem Setting

- need **many shadow maps** of **low/moderate resolution**
- rendering the scene many times (transformation, …) is costly
  - what we need is level-of-detail rendering
  - point representations are well-suited for fast, approximate renderings
  - two approaches: simple LOD with no connectivity and water-tight rendering with point hierarchy

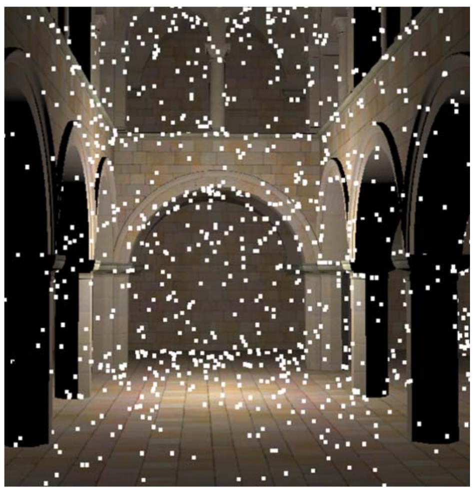


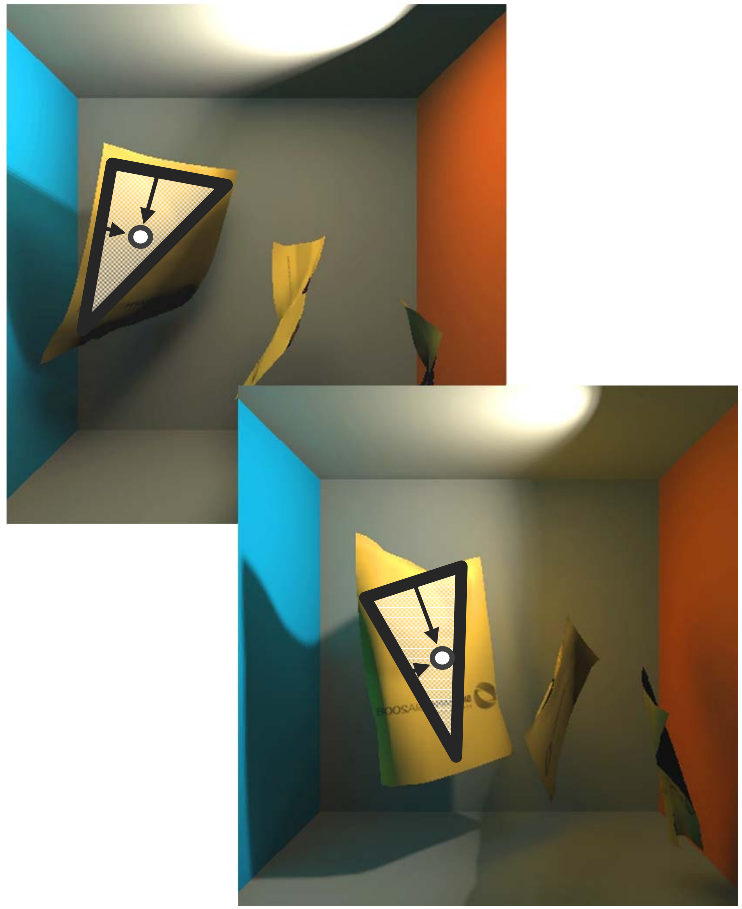


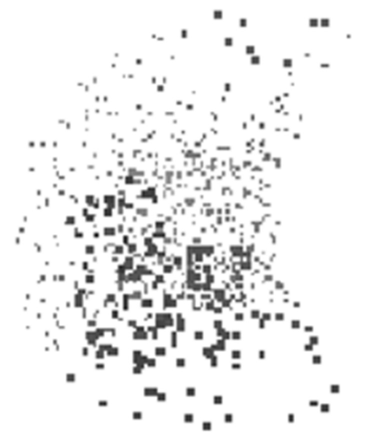
Shadow Mapping for VPLs

Imperfect Shadow Maps

- create random sets of point samples (triangle ID + barycentric coords)
- 4k to 16k points per “shadow map” (global parameter)







Shadow Mapping for VPLs

Imperfect Shadow Maps

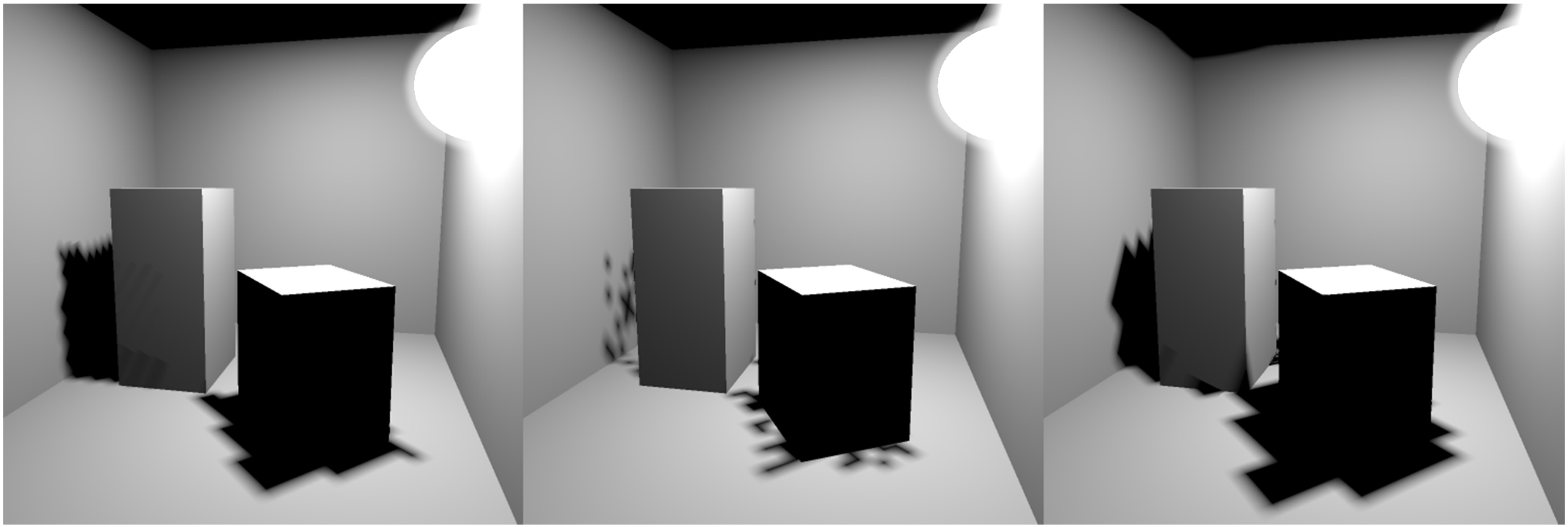
- 4k to 16k points per “shadow map” (global parameter)
- heuristic to reconstruct the surfaces from point samples

without pull-push | **with** pull-push | triangle rasterization

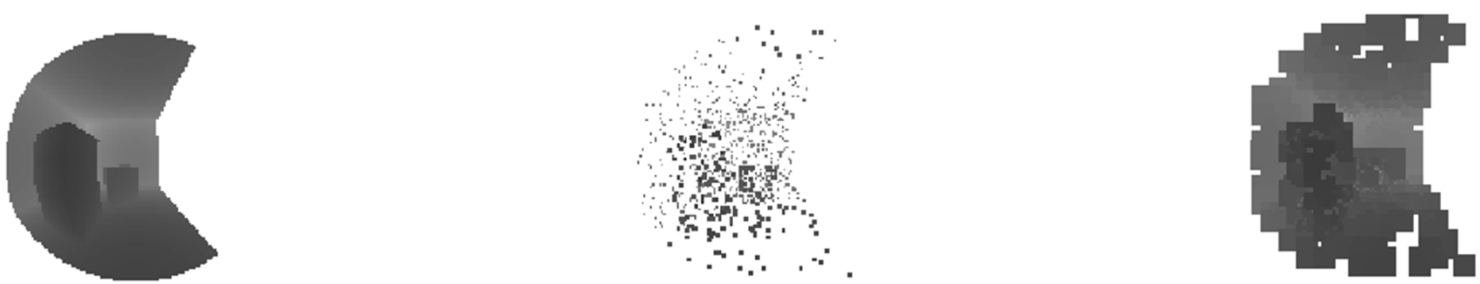
Shadow Mapping for VPLs

Imperfect Shadow Maps

- comparison of shadow maps for a single point light



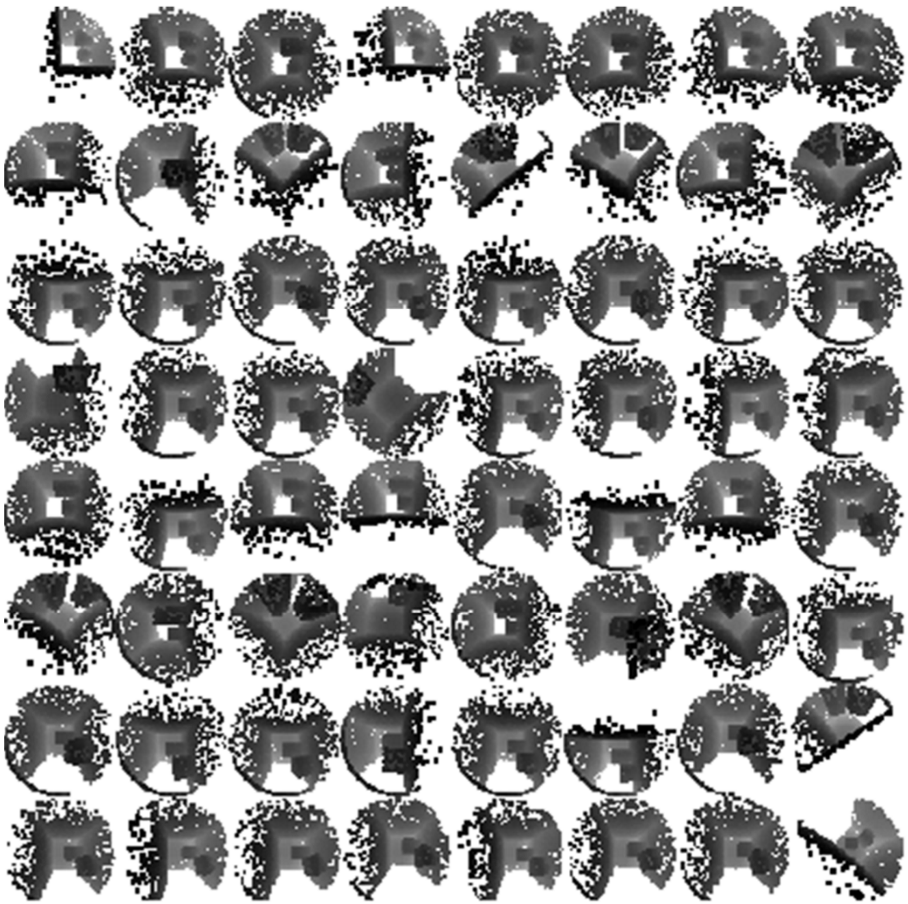
triangle rasterization | **without** pull-push | **with** pull-push



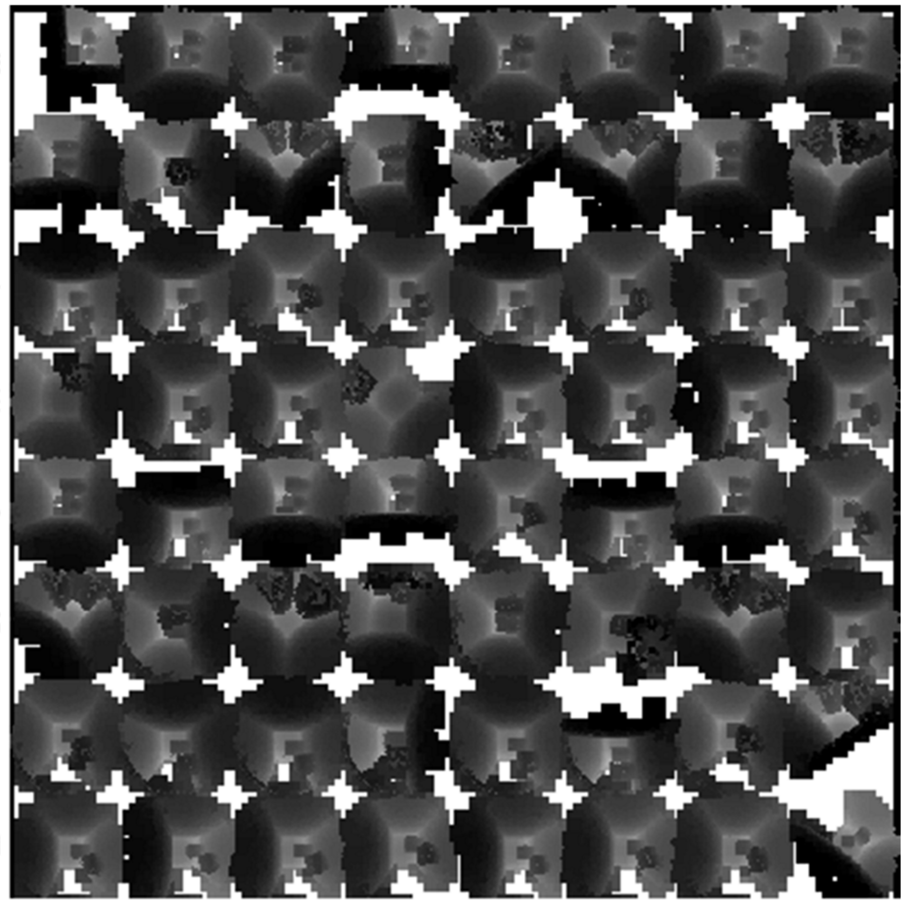
Shadow Mapping for VPLs

Imperfect Shadow Maps

- pull-push in image-space: parallel for thousands of shadow maps



without pull-push

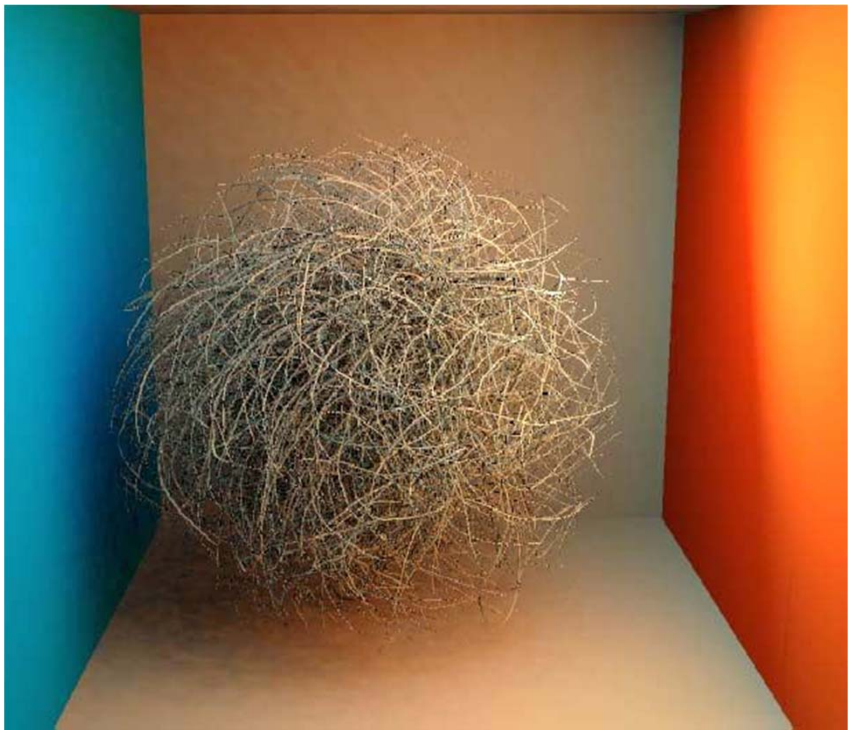


with pull-push

Shadow Mapping for VPLs

Imperfect Shadow Maps

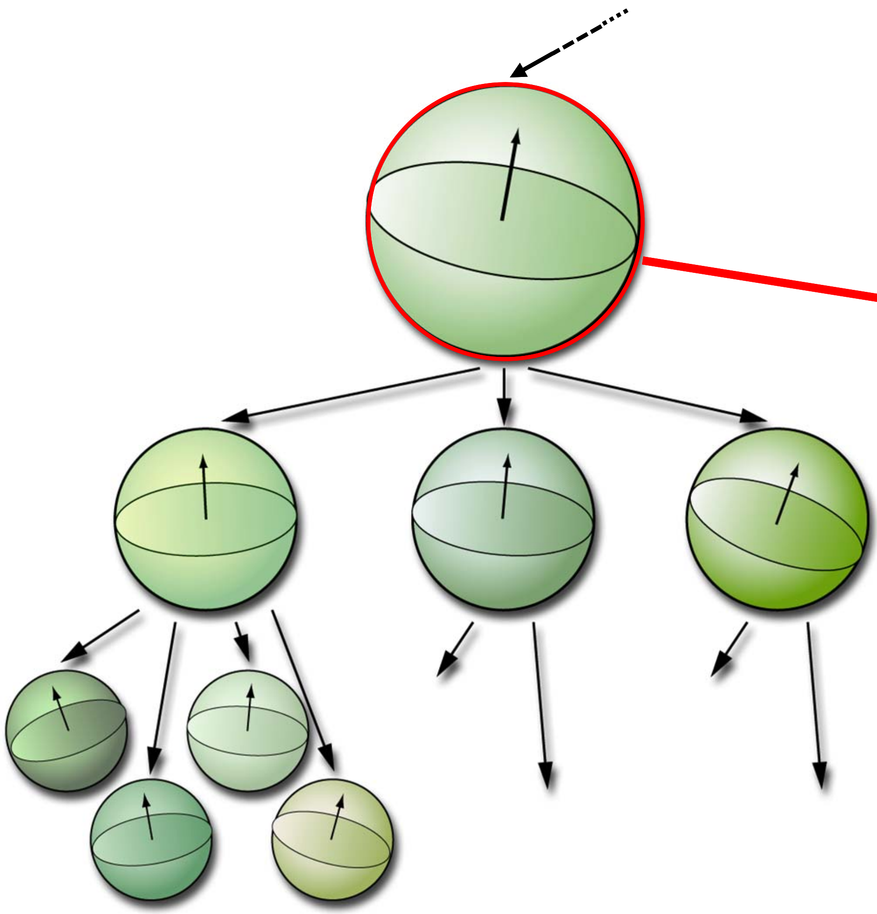
- … can render thousands of shadow maps in 100ms
- … work because errors average out
- … require playing with parameters

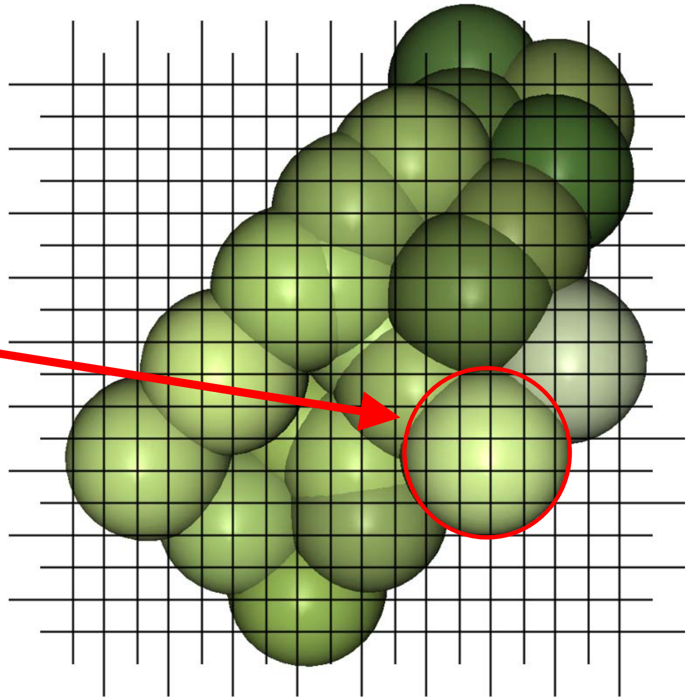


Shadow Mapping for VPLs

High-Quality Point-based Rendering

- create random points on surfaces and create hierarchy
- idea of Qsplat: traverse hierarchy until projected size of point primitive is small enough





Shadow Mapping for VPLs

High-Quality Point-based Rendering

- create random points on surfaces and create hierarchy
- idea of Qsplat: traverse hierarchy until projected size of point primitive is small enough

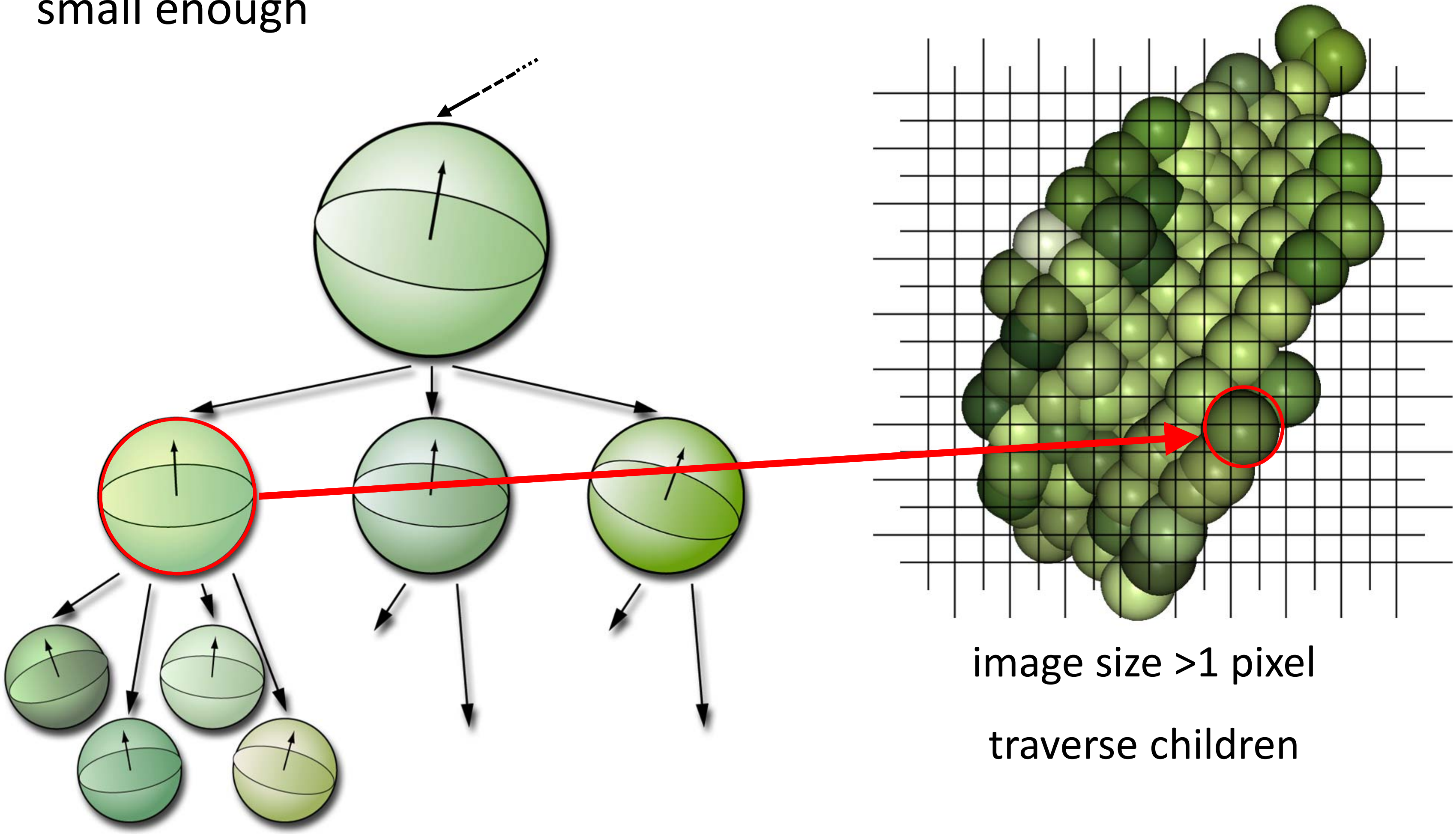


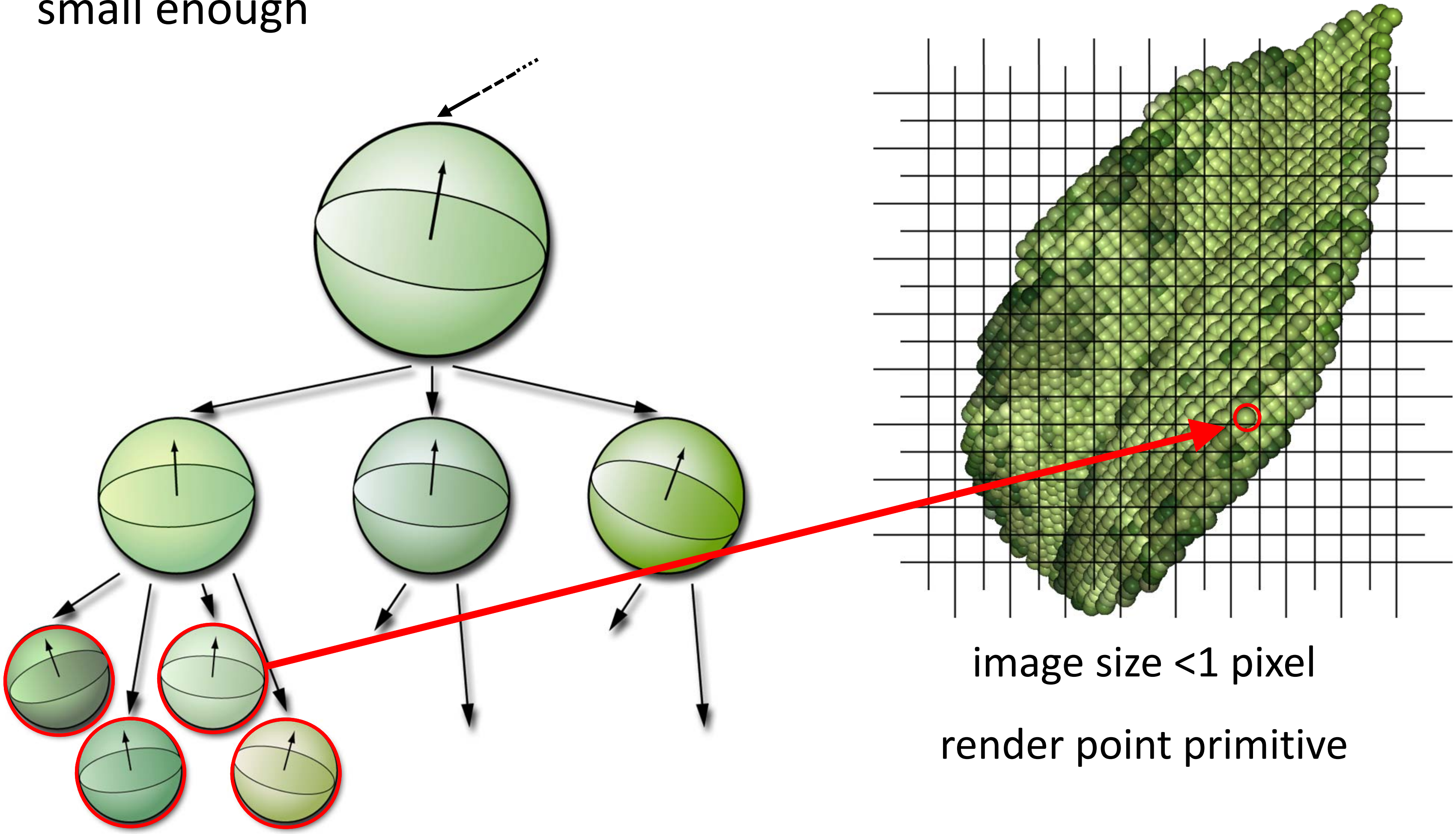
image size >1 pixel

traverse children

Shadow Mapping for VPLs

High-Quality Point-based Rendering

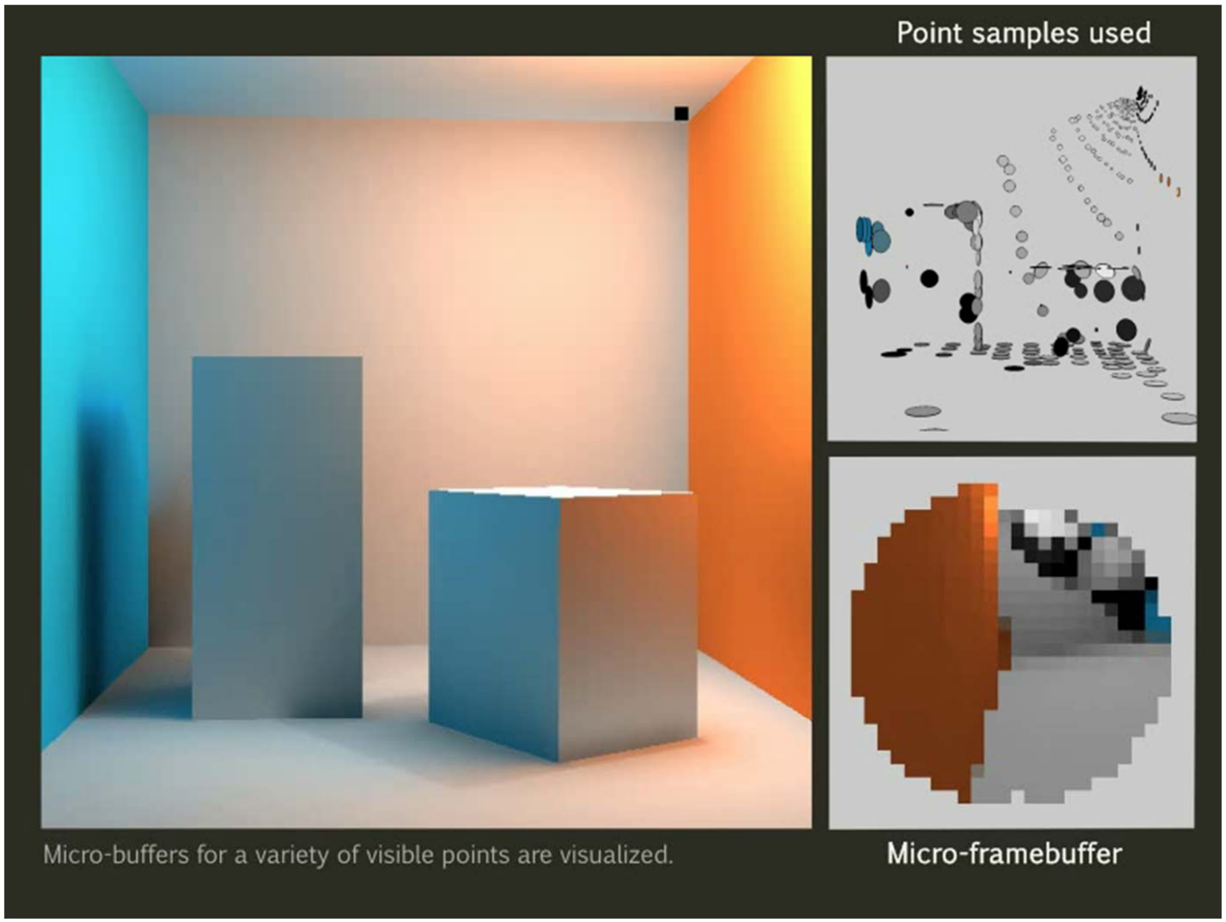
- create random points on surfaces and create hierarchy
- idea of Qsplat: traverse hierarchy until projected size of point primitive is small enough



Shadow Mapping for VPLs

Micro-Rendering

- renders accurate environment maps / depth buffers from point hierarchy
- actually developed for final gathering, using CUDA/OpenCL
- can be used to create (R)SMs (in 2009: ~16k in 100 ms, each 24^2 pixels)



Shadow Mapping for VPLs

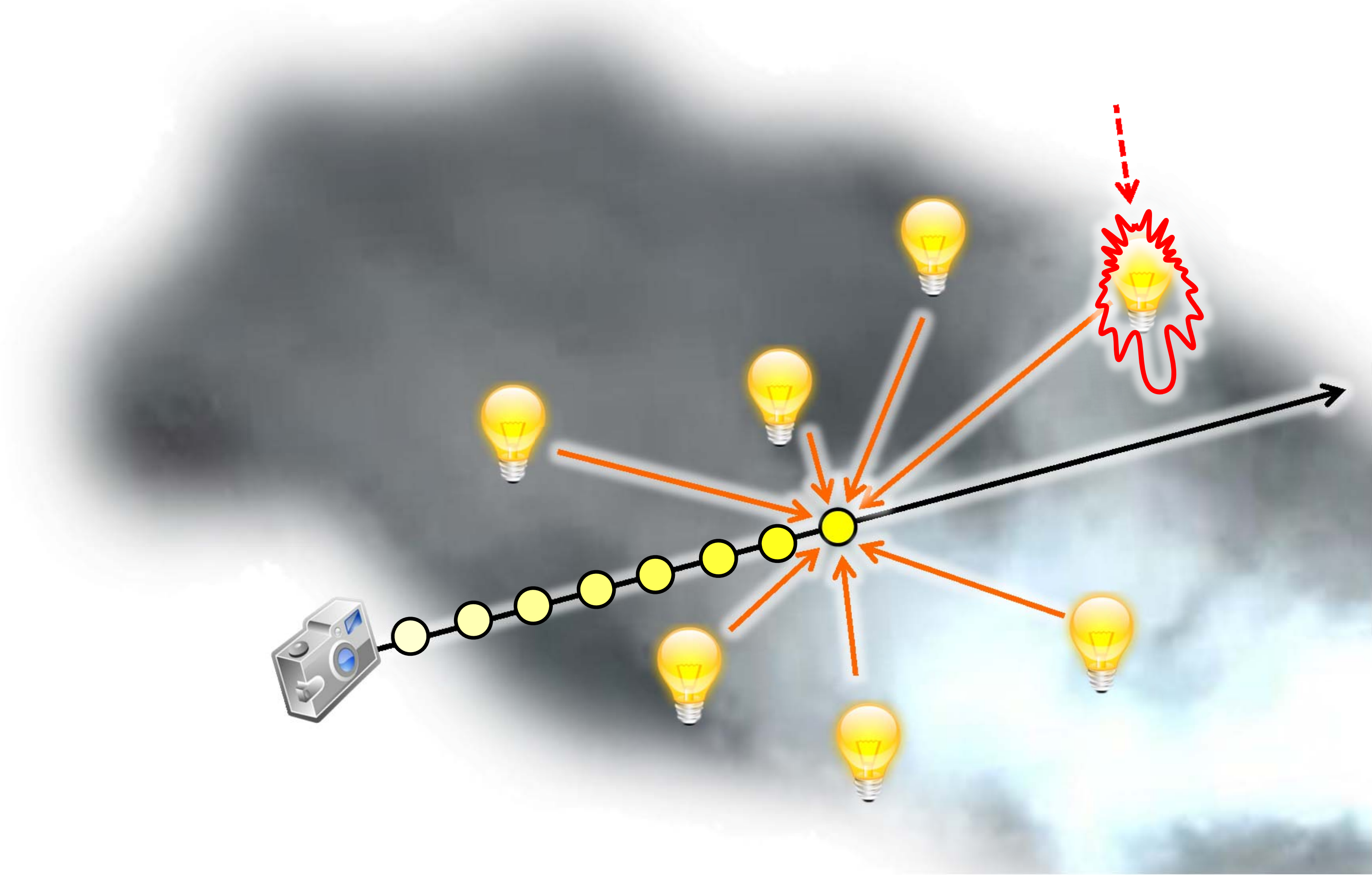
ManyLODs [Holländer, PhD Thesis]

- fine-grained LOD selection for many views based on BVH
- incremental and lazy update schemes to many-view problem

Rendering Strategies for Participating Media

Light Transport in Participating Media

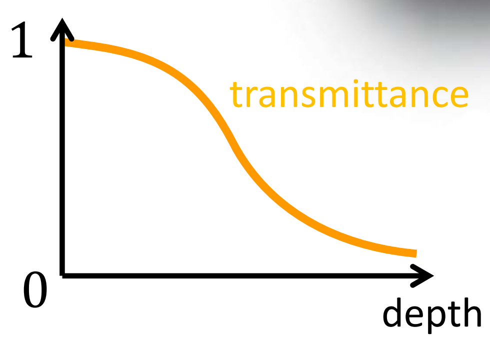
- direct light from surface VPLs and
- single-scattering from media VPLs (emit according to phase function)
- VPLs also generated at scattering events in media (Jan' part)

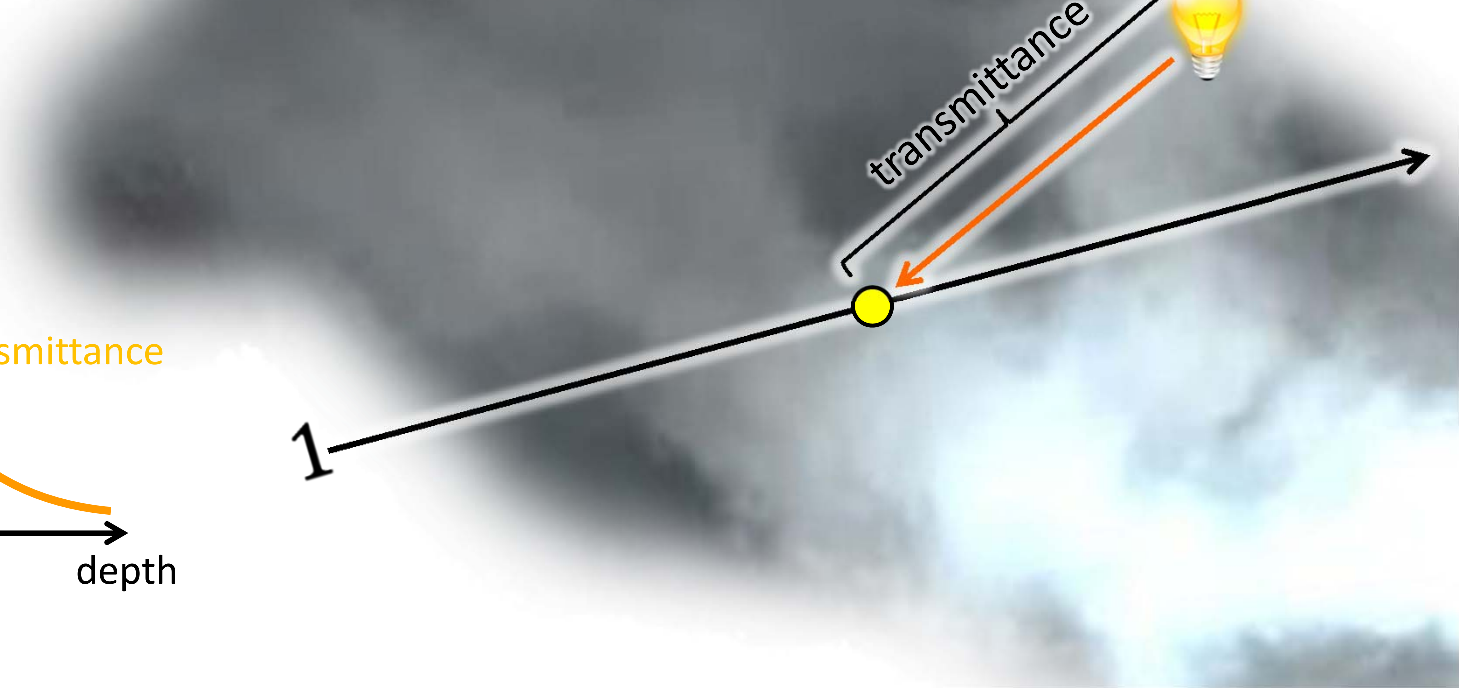


Participating Media with Many-Lights

Visibility and Transmittance

- homogeneous media:
  - standard shadow map per VPL (compute transmittance)
- heterogeneous media:
  - shadow map plus ray marching or
  - deep shadow maps [LV00] or
  - adaptive volumetric SM [SVLL10]

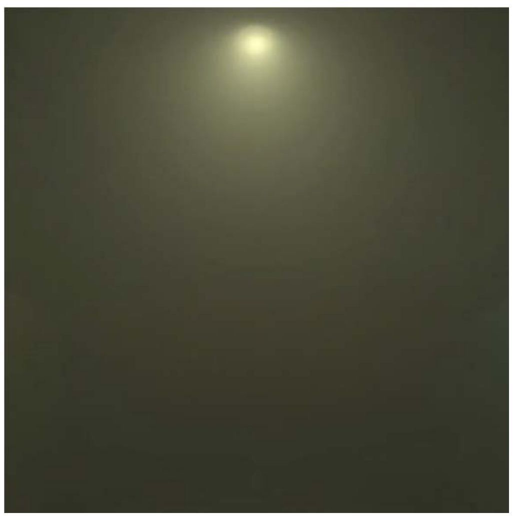




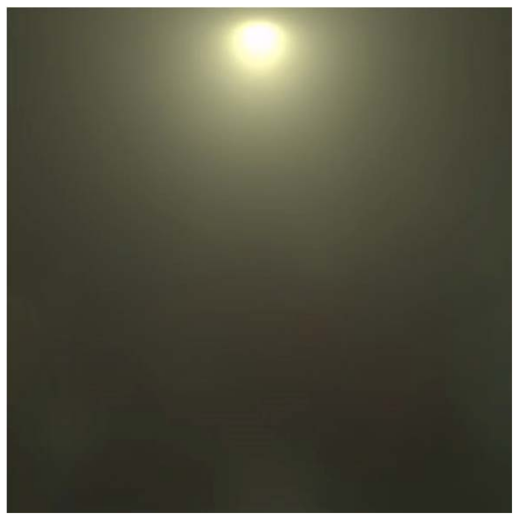
Real-time Many-light Rendering

Conclusions

- many-lights methods work quite well in real-time
  - bias compensation is feasible for surfaces and media
  - glossiness for surfaces ↔ anisotropic phase functions for media
  - for mostly diffuse scenes, for scenes with moderate anisotropic media



isotropic



moderate anisotropic

strong anisotropic