Global Illumination

2017 Fall

National Cheng Kung University

Instructor: Shih-Chin Weng 翁士欽 (Style.Me)

What is Global Illumination?

Photo by Abigail Keenan

Photo by Kaushik Panchal

Stages of Photorealistic Rendering

- Measurement and acquisition of scene data
  - Reflectance (BRDF), sub-surface scattering (BSSRDF), appearance (BTF), etc.
- **Light transport simulation (today's focus!)**
  - Ray tracing, photon-mapping, radiosity, etc.
- Visual display
  - Tone mapping

The State of Rendering

Ray tracing is everywhere in VFX & animation industry!


Walt Disney
Animation Studios
Hyperion Renderer


weta
DIGITAL
Manuka Renderer

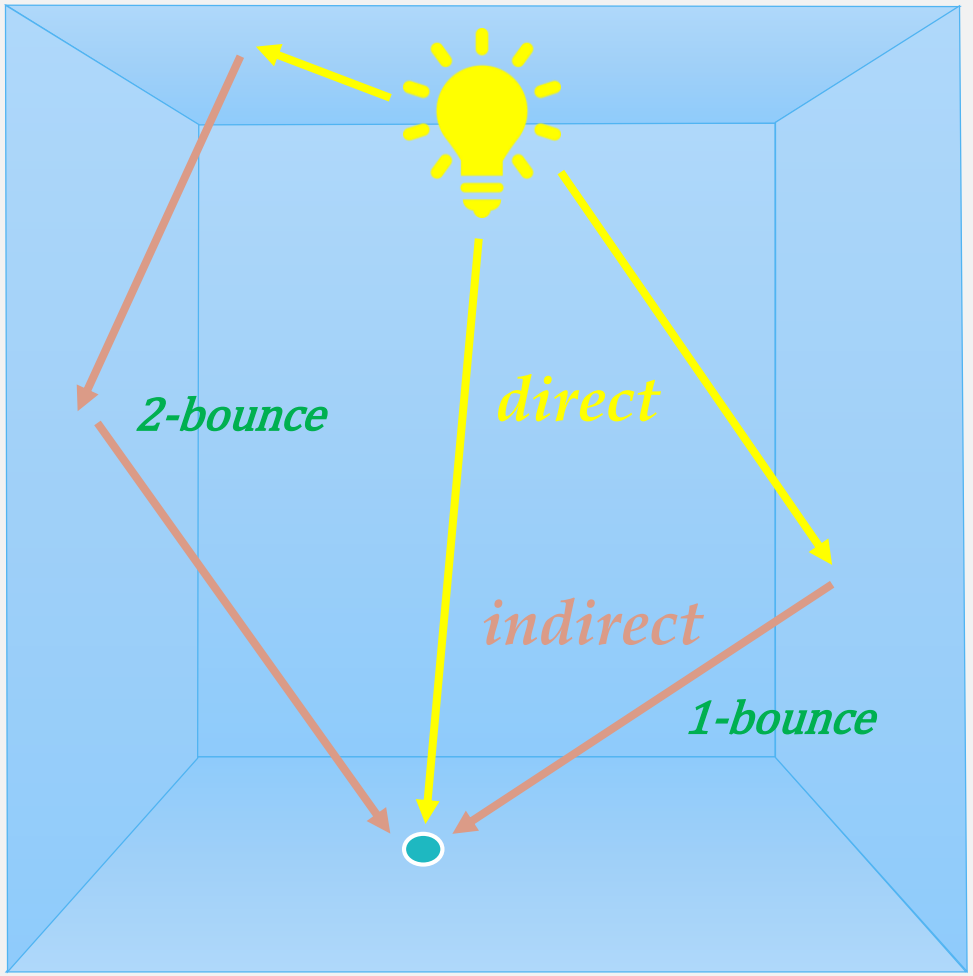

Disney · PIXAR
RenderMan


SOLIDANGLE
arnold renderer

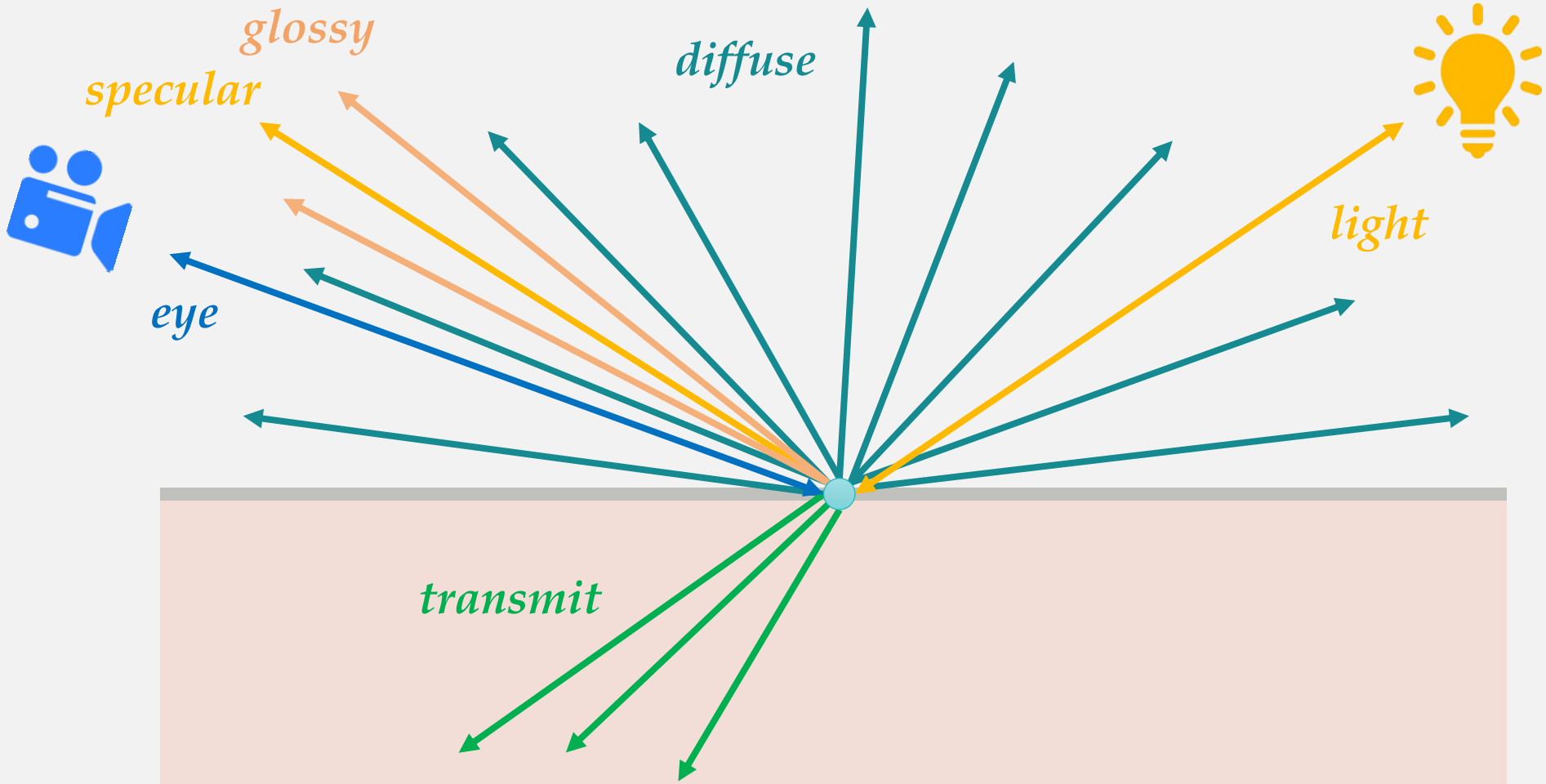

KONG
SKULL ISLAND
MAKING OF
HYBRIDE
A UBISOFT Division

Basic Concepts

Where Does Light Come From?

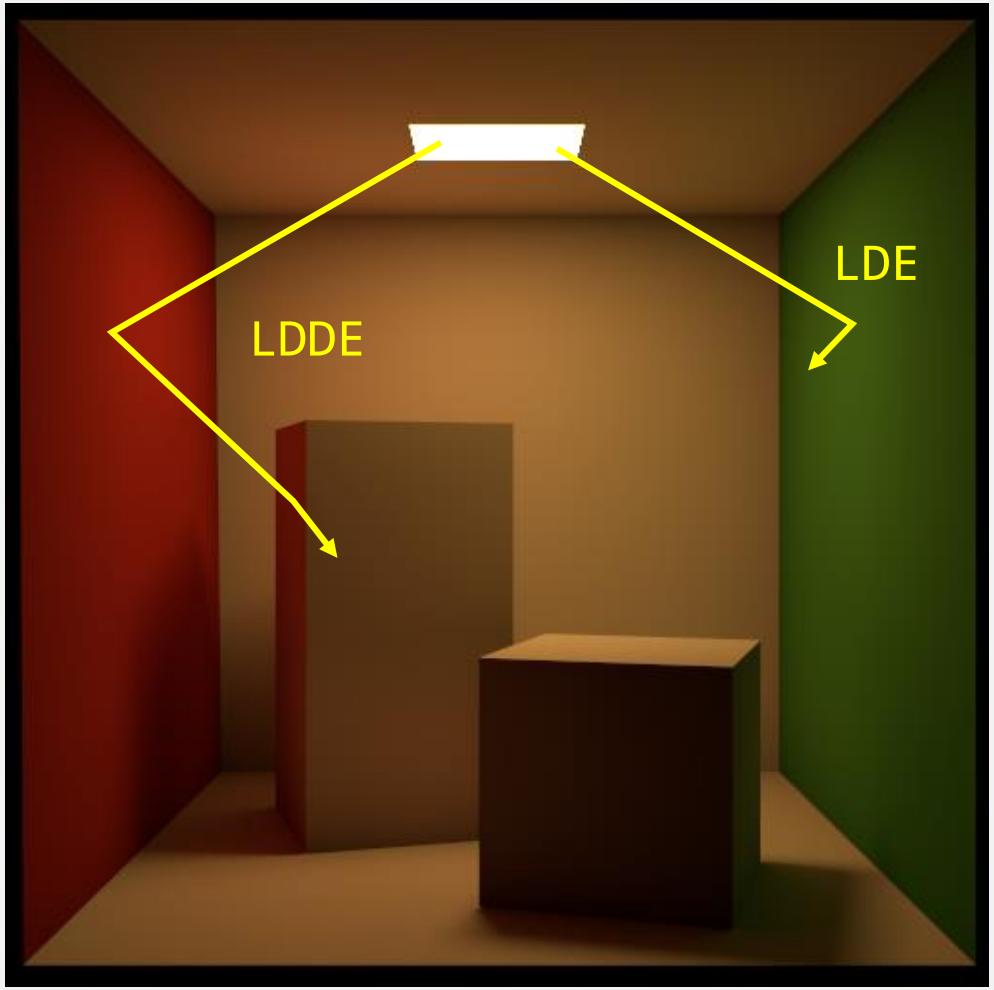


Light Path Expression

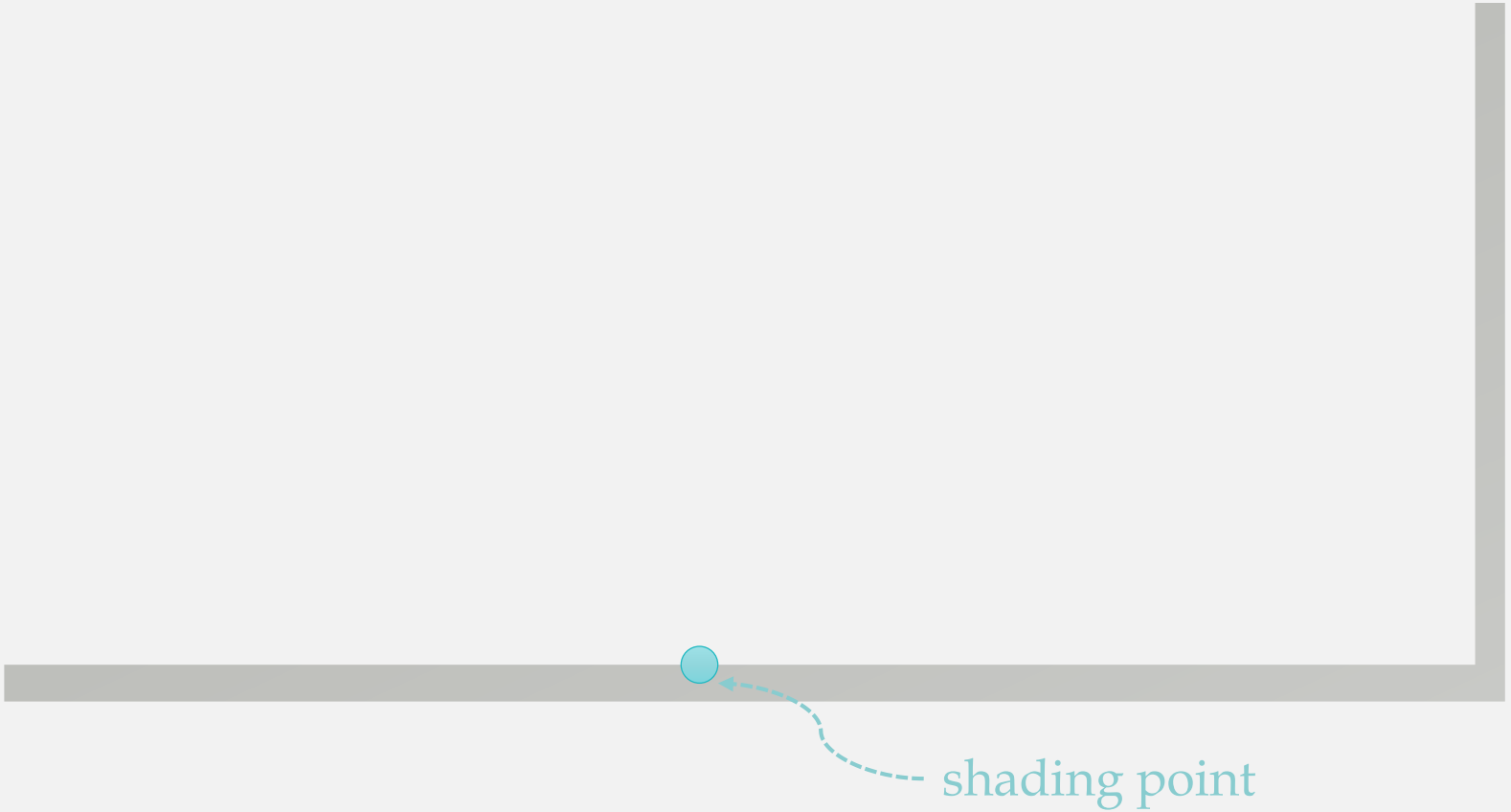



```
L: light
D: diffuse
S: specular
G: glossy
T: transmit
E: eye
```

Light Path Expression (Cont.)

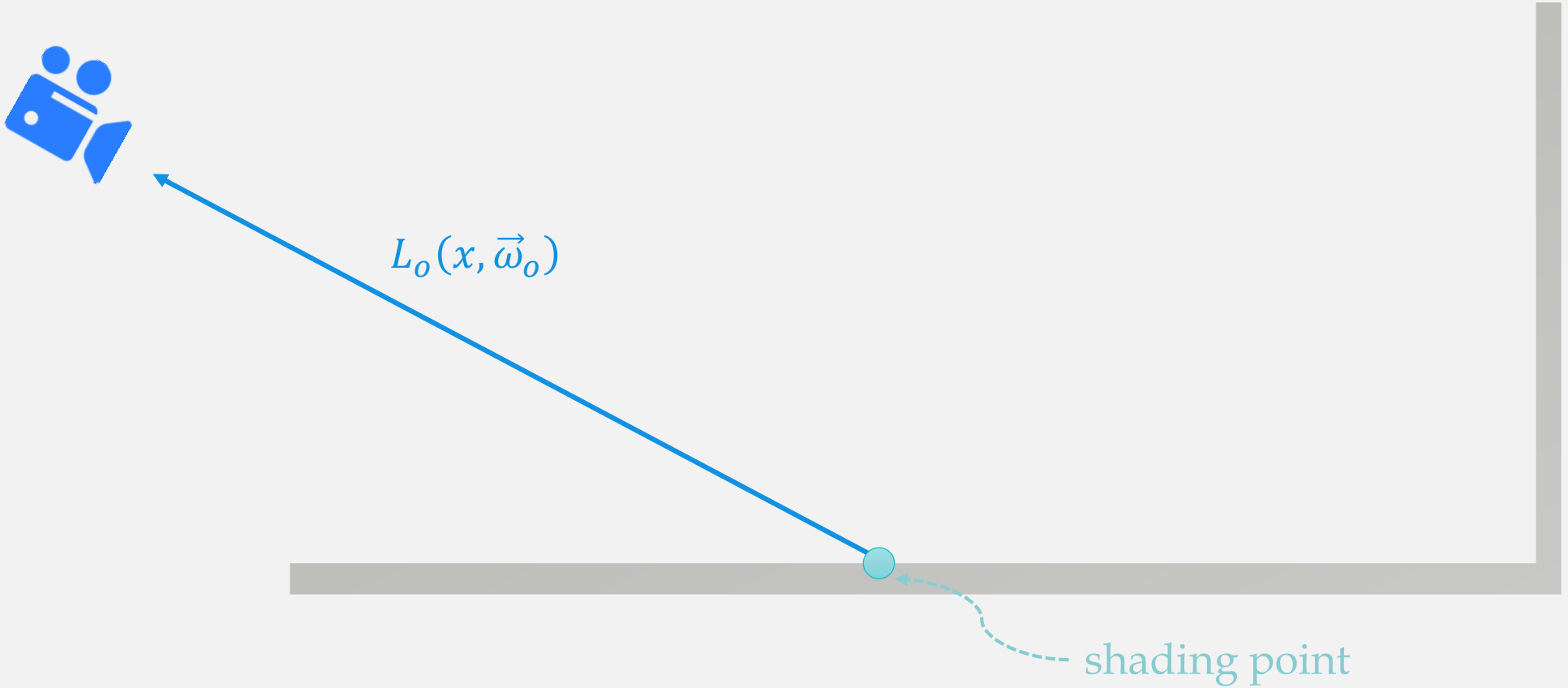


Recursive Ray Tracing



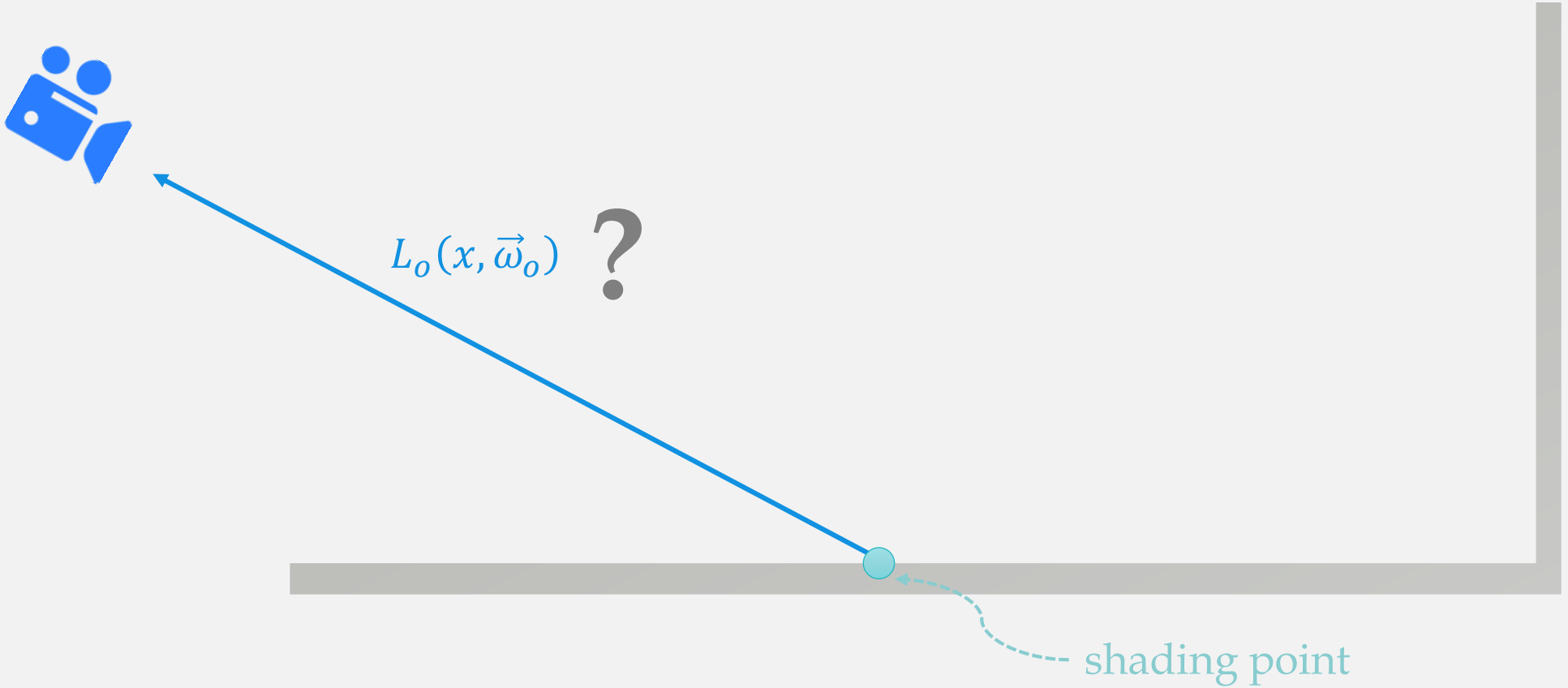
[Inspired by Křivánek, '10]

Recursive Ray Tracing



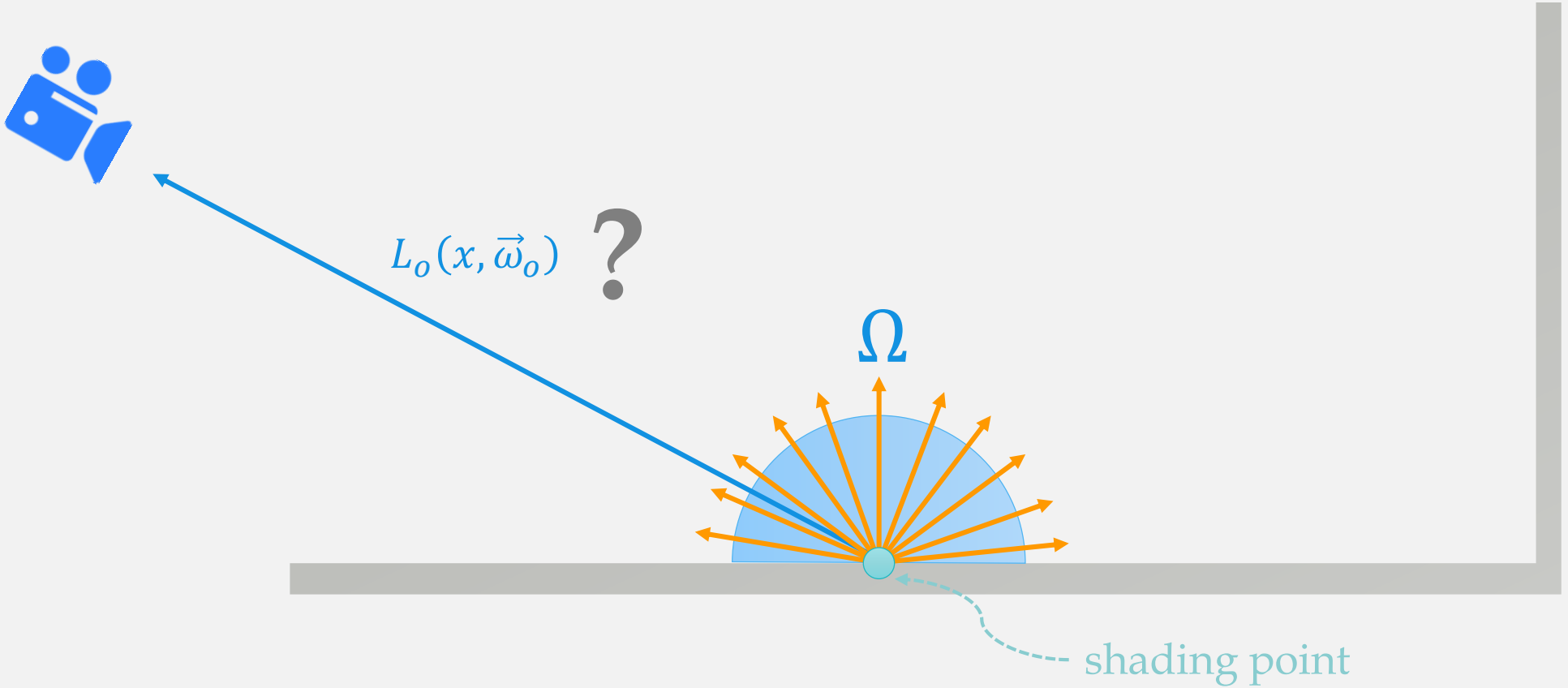
[Inspired by Křivánek, '10]

Recursive Ray Tracing



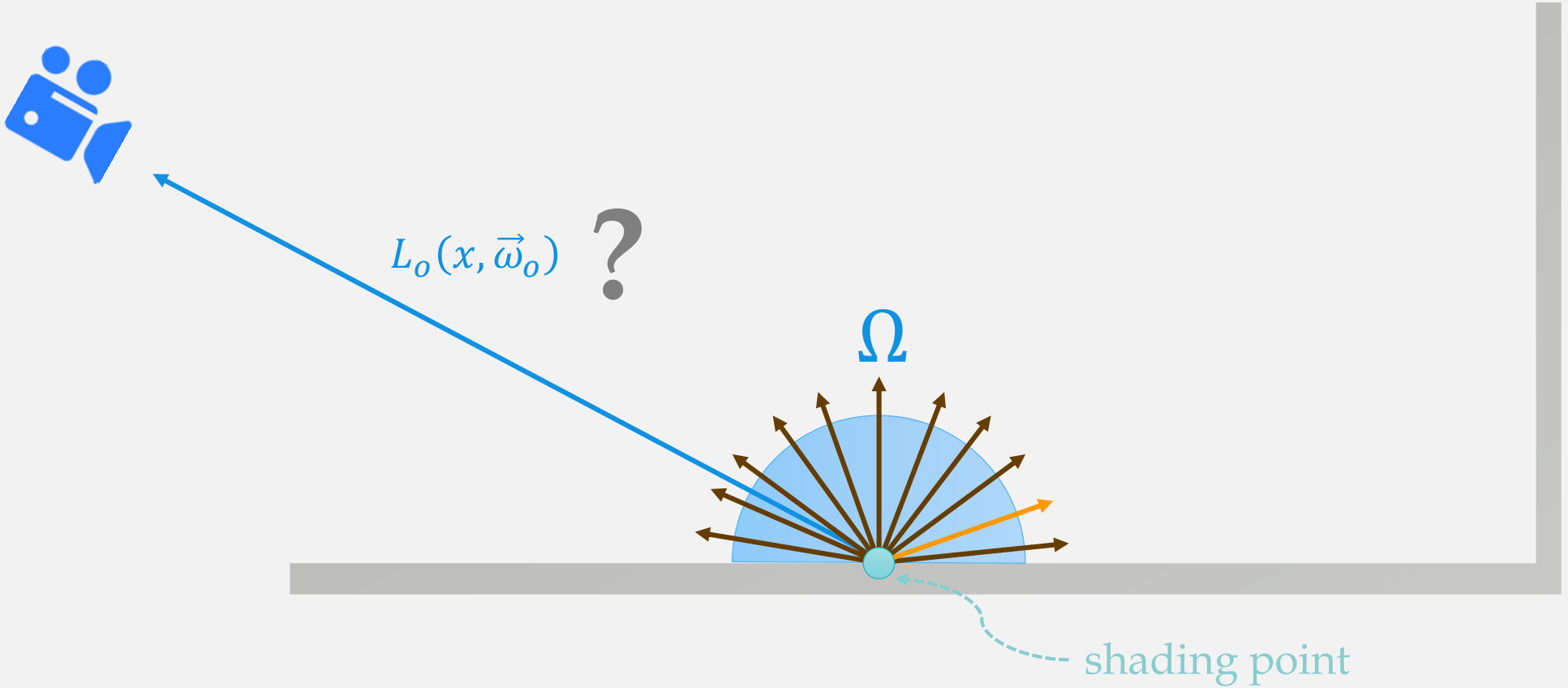
[Inspired by Křivánek, '10]

Recursive Ray Tracing



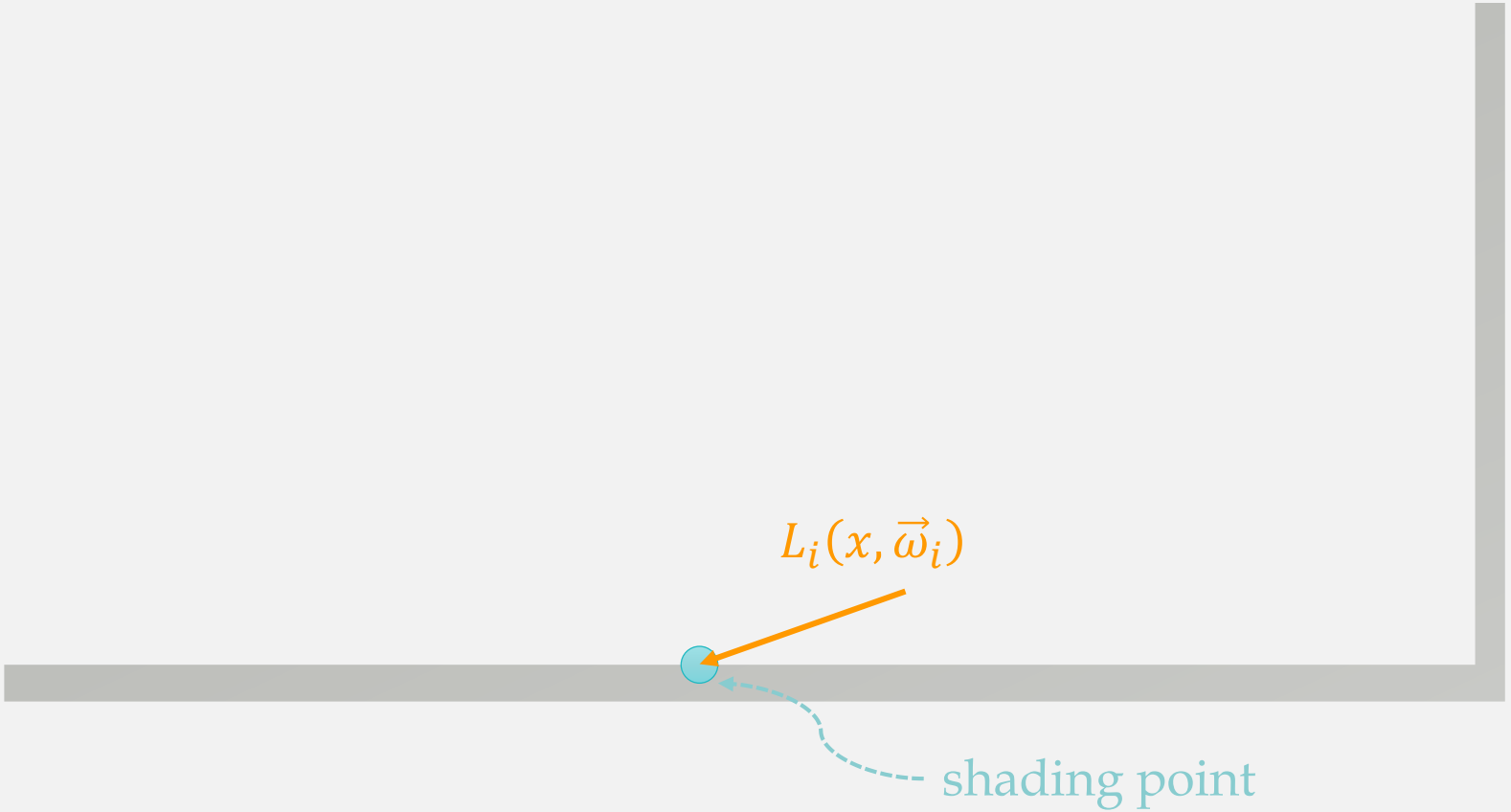
[Inspired by Křivánek, '10]

Recursive Ray Tracing



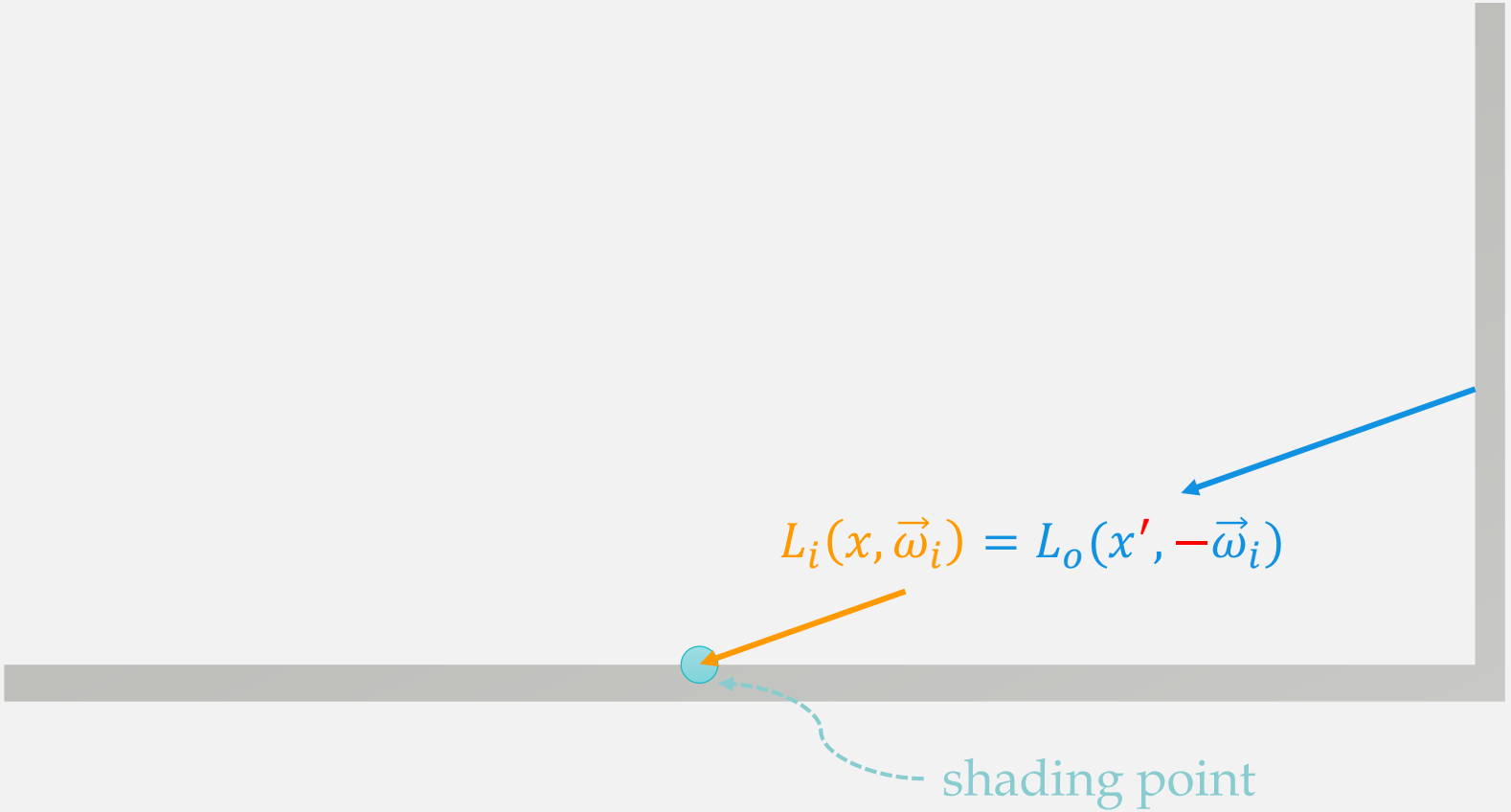
[Inspired by Křivánek, '10]

Recursive Ray Tracing



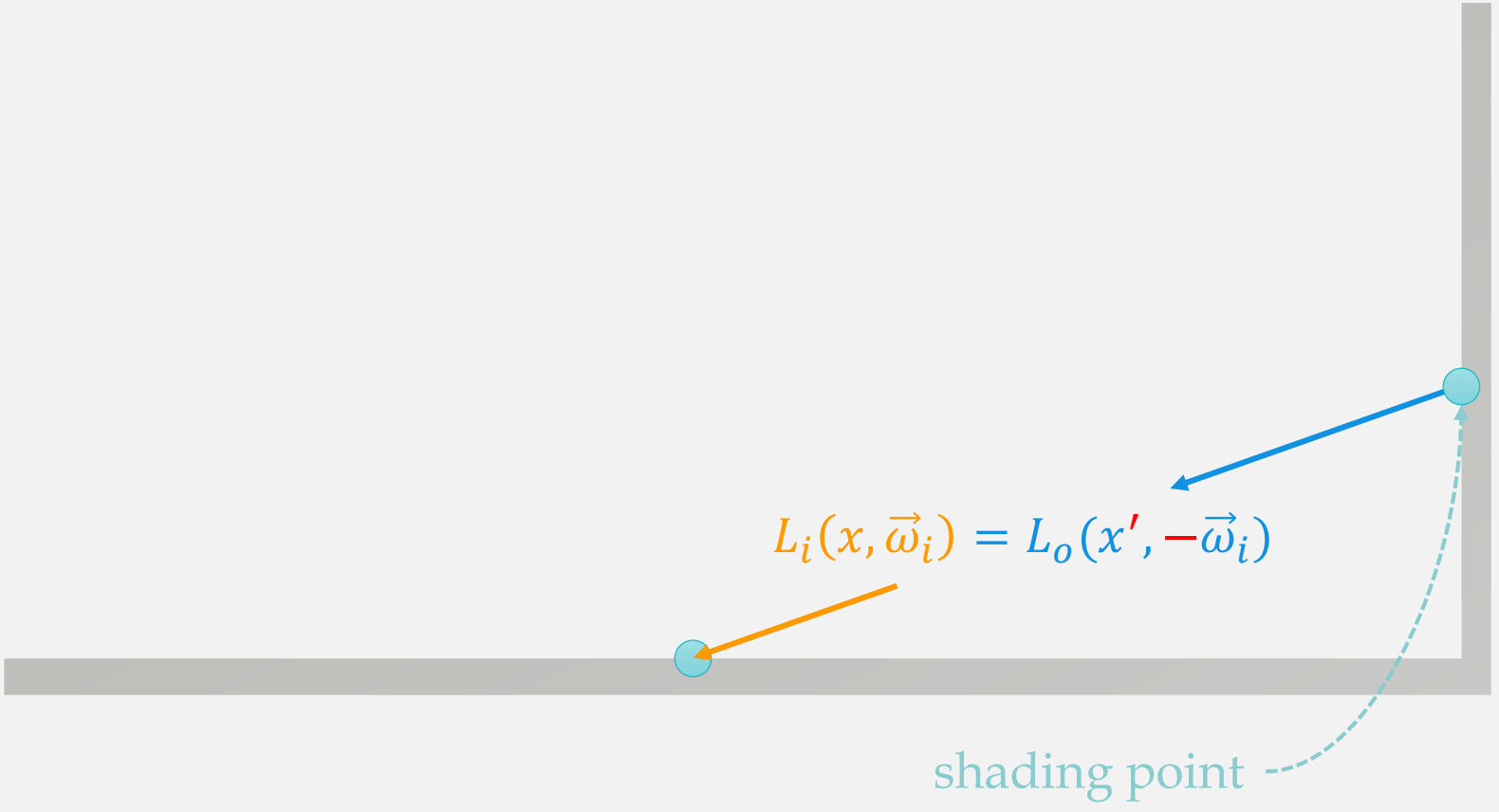
[Inspired by Křivánek, '10]

Recursive Ray Tracing



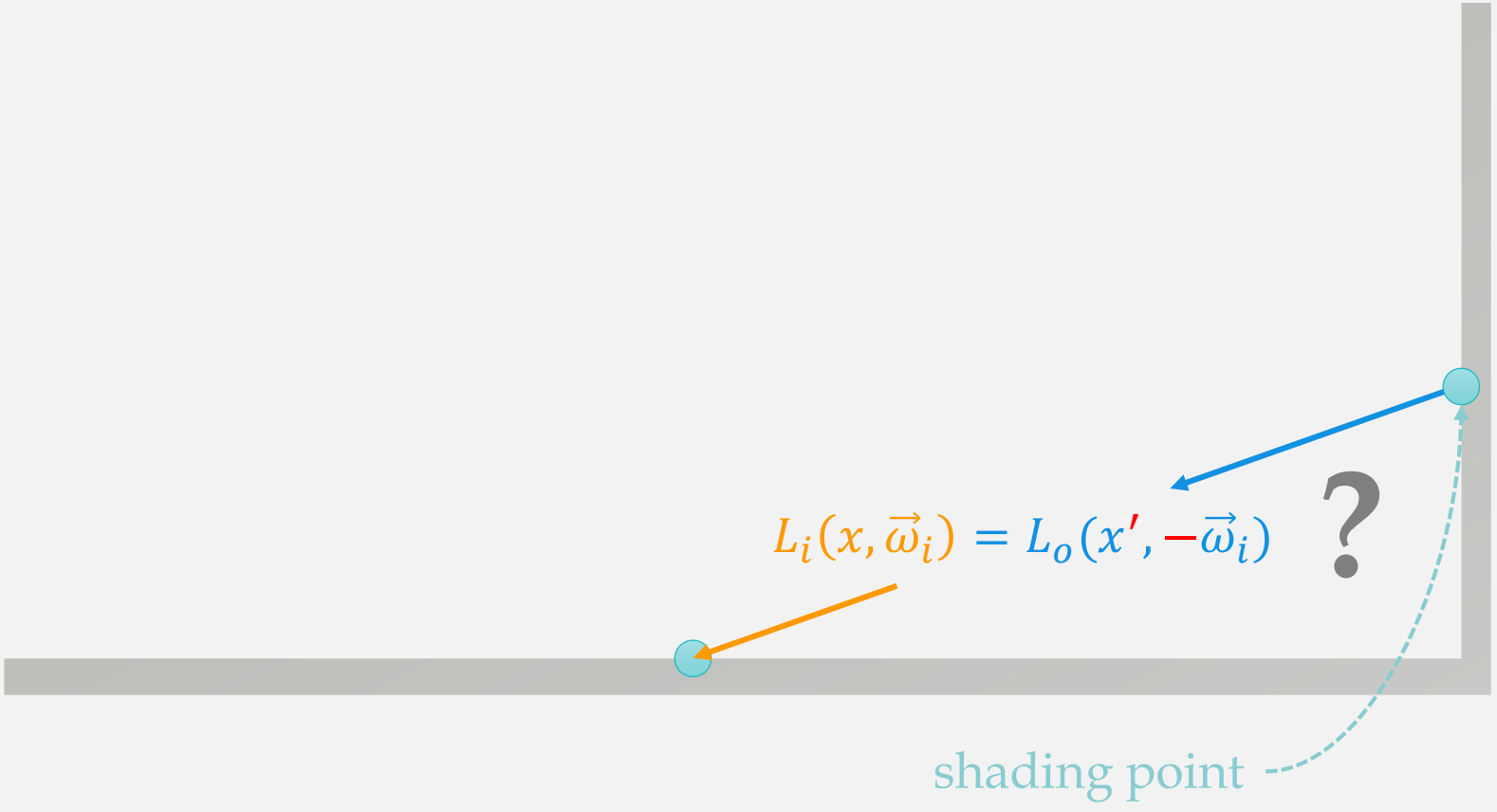
[Inspired by Křivánek, '10]

Recursive Ray Tracing



[Inspired by Křivánek, '10]

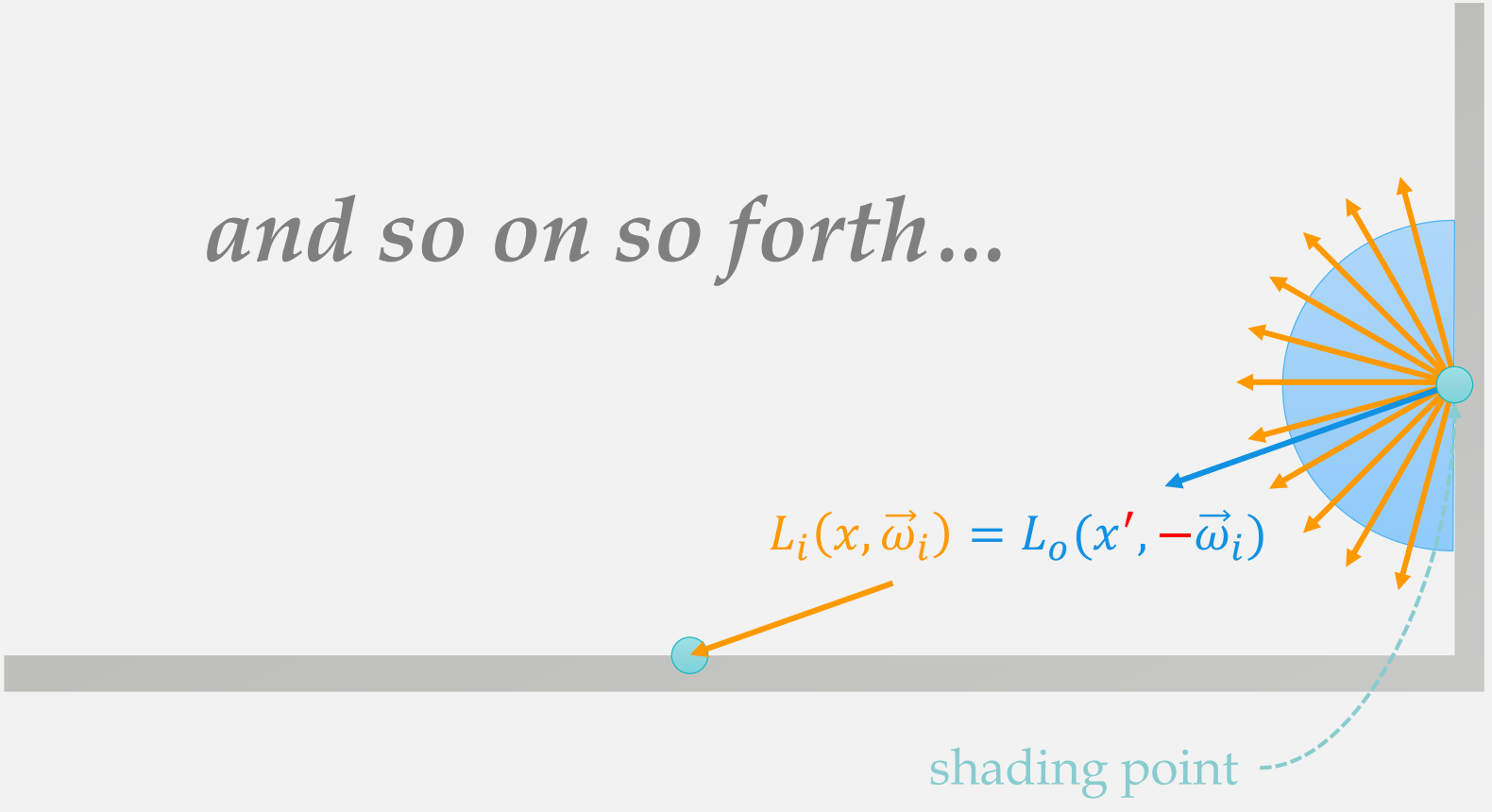
Recursive Ray Tracing



[Inspired by Křivánek, '10]

Recursive Ray Tracing

and so on so forth…



[Inspired by Křivánek, '10]


bounce 1

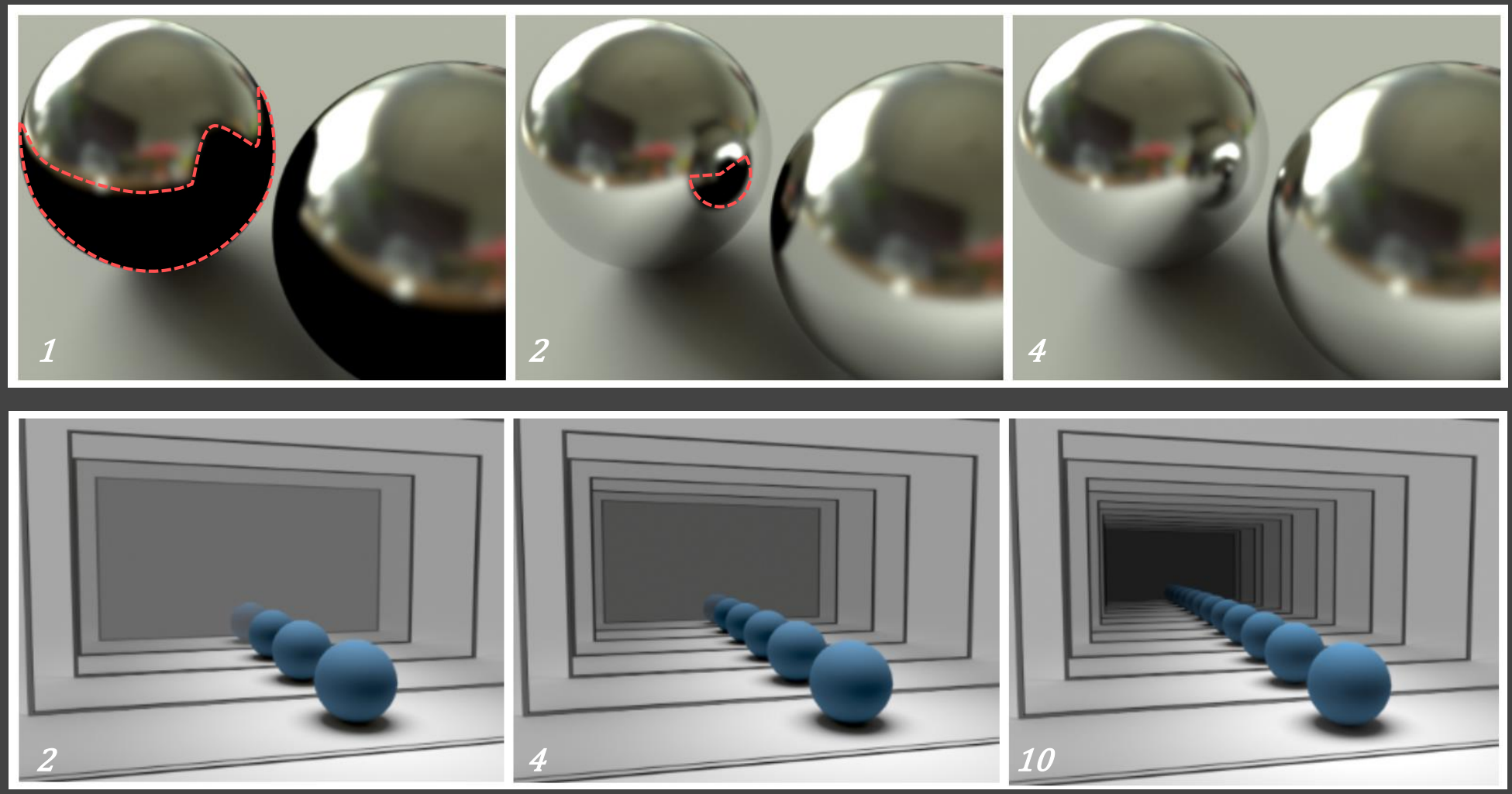

bounce 2


bounce 4


bounce 8

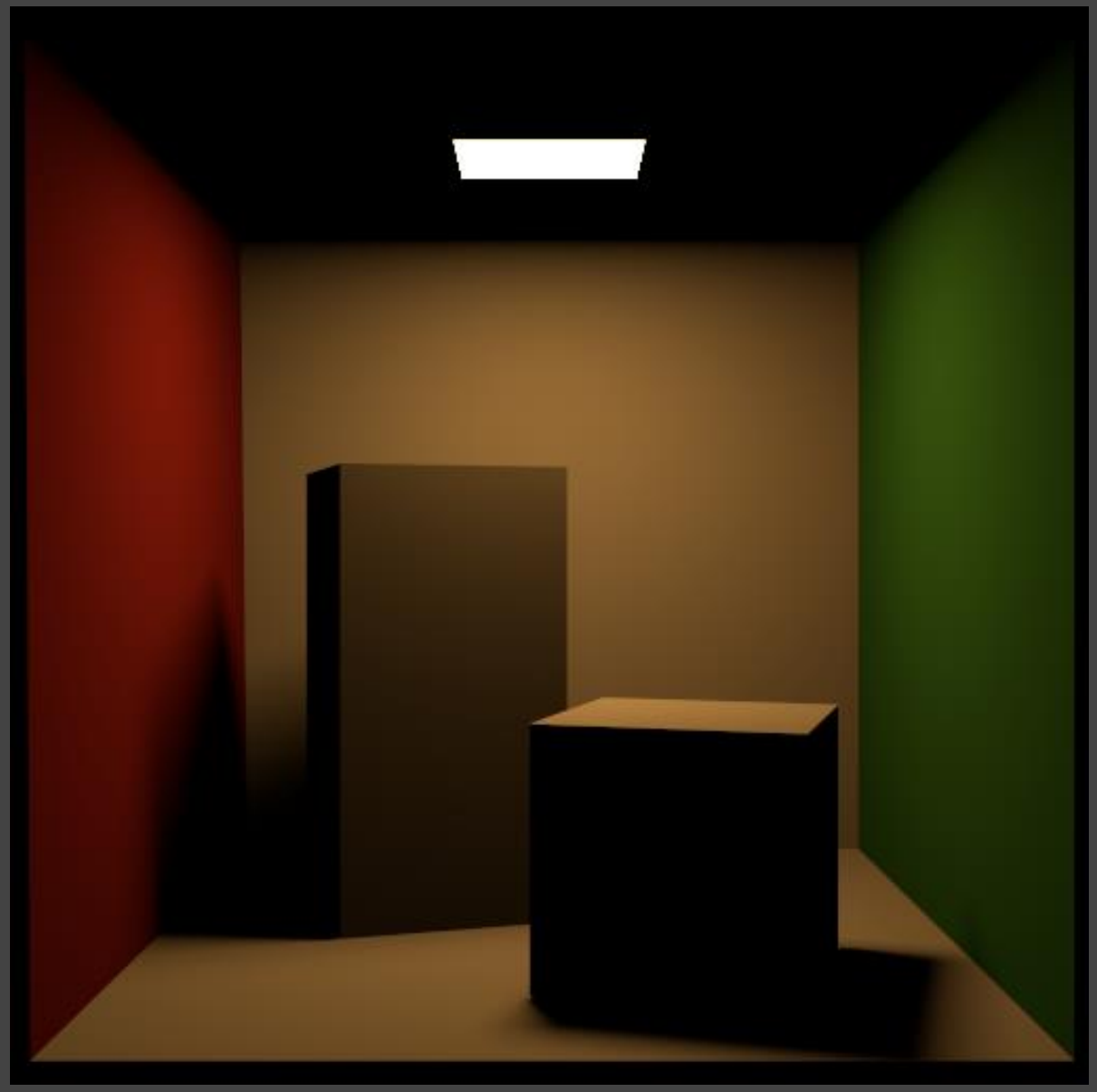

bounce 16

Ray Depth Control in Arnold Renderer

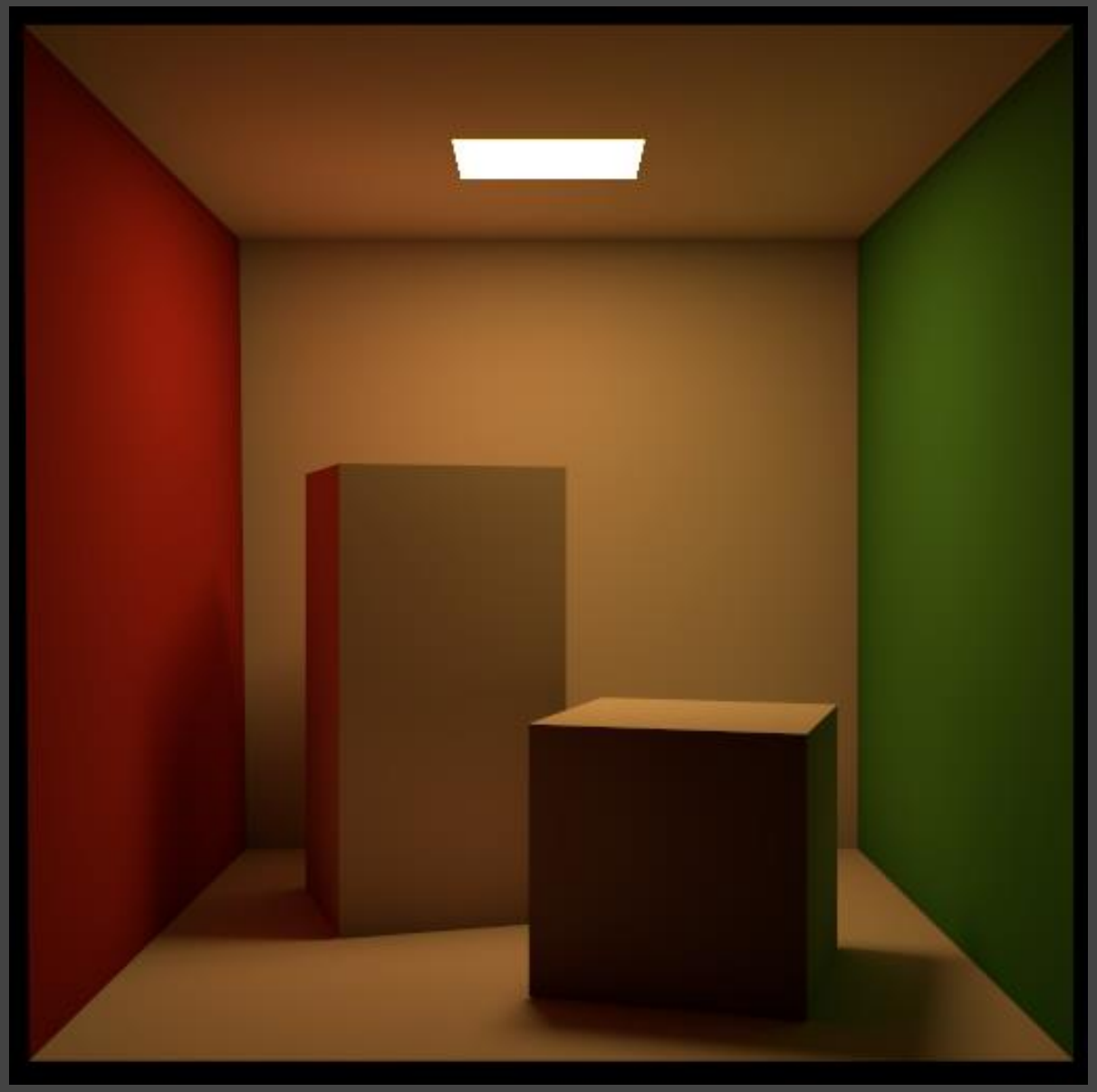


https://support.solidangle.com/display/AFMUG/Ray+Depth

Spot the Difference!



Direct Illumination

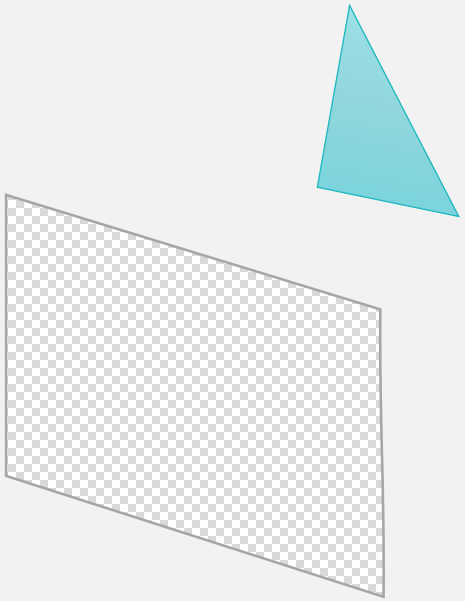


Global Illumination (Direct + Indirect)

Primary Visibility

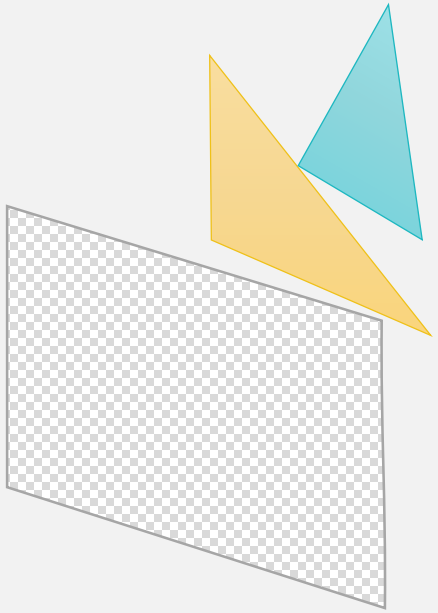
Rasterization

- Surface to eye
- Visibility via depth buffer



Ray Tracing

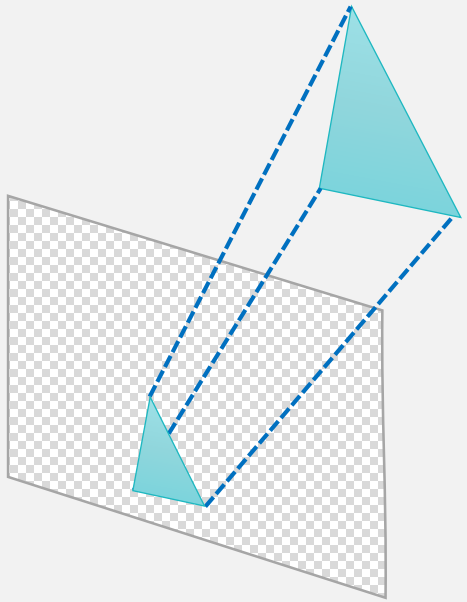
- Eye to surface
- Visibility via ray casting



Primary Visibility

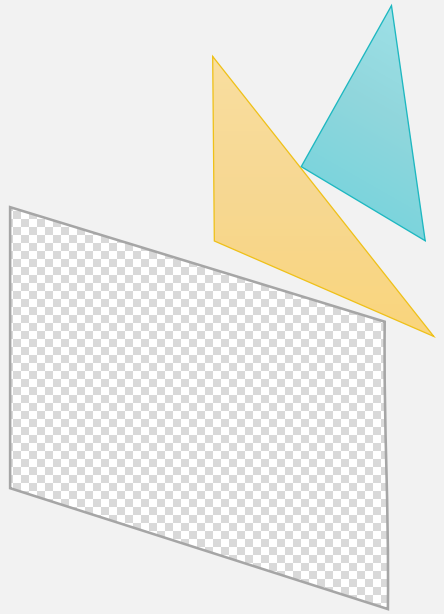
Rasterization

- Surface to eye
- Visibility via depth buffer



Ray Tracing

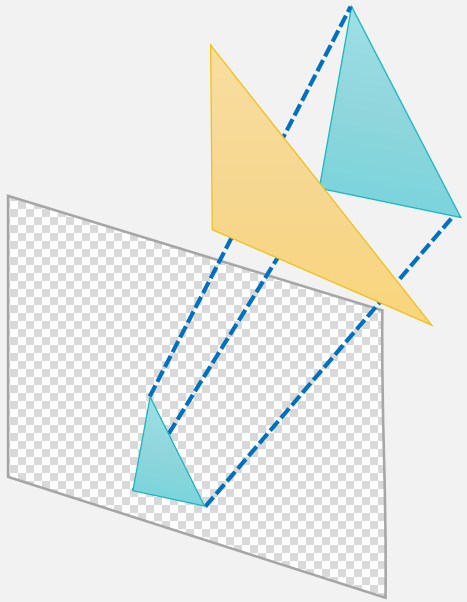
- Eye to surface
- Visibility via ray casting



Primary Visibility

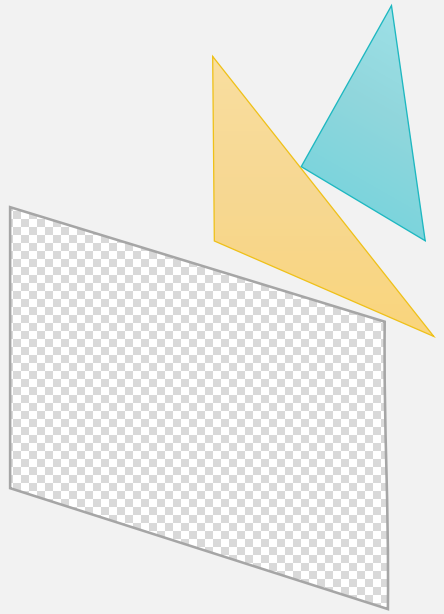
Rasterization

- Surface to eye
- Visibility via depth buffer



Ray Tracing

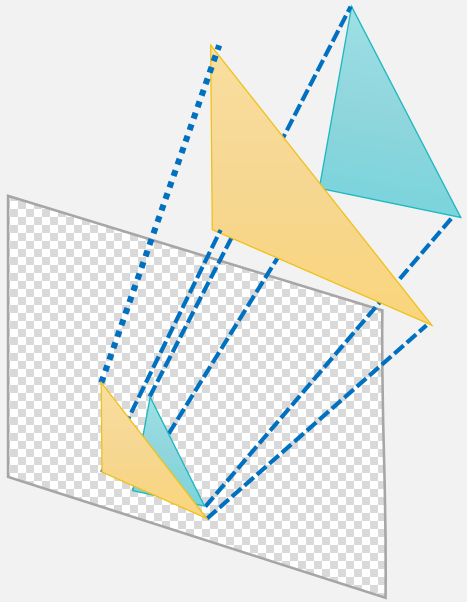
- Eye to surface
- Visibility via ray casting



Primary Visibility

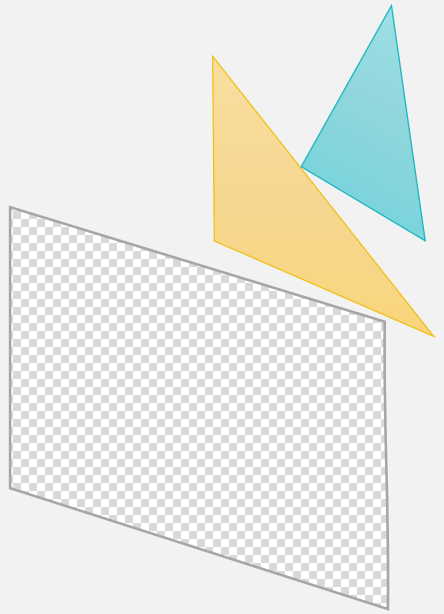
Rasterization

- Surface to eye
- Visibility via depth buffer



Ray Tracing

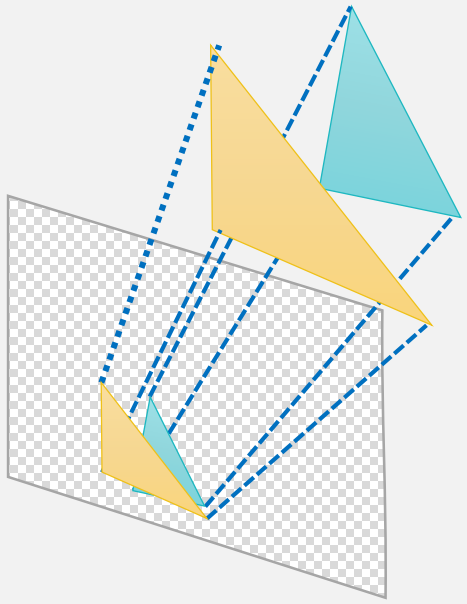
- Eye to surface
- Visibility via ray casting



Primary Visibility

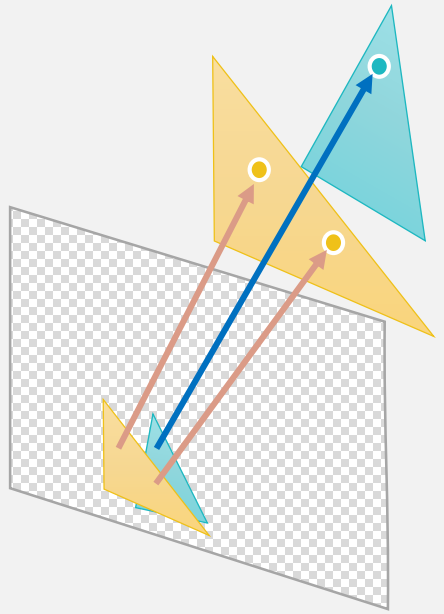
Rasterization

- Surface to eye
- Visibility via depth buffer



Ray Tracing

- Eye to surface
- Visibility via ray casting



Ray-Casting

- Find the nearest intersection from a ray
- Computed with different geometry representations
  - Explicit
    - Triangular meshes
    - Bezier curves for hair/fur
  - Implicit
    - Volume data (voxels)
    - Point cloud

Acceleration Structures for Ray-Casting

Ray-Casting Computation

```
for each ray in each pixel:
   for each geometry primitive in the scene:
      if intersect(ray, primitive):
         return closest point
```

Ray-Casting Computation

```
for each ray in each pixel:
    for each geometry primitive in the scene:
        if intersect(ray, primitive):
            return closest point
```

Spatial Coherence

- Geometry primitives only occupy a small portion of the ambient space
- Primitives can be ordered by their spatial locations
- A location in space is associated with a limited number of primitives

But, how should we do it?

Divide & Conquer!

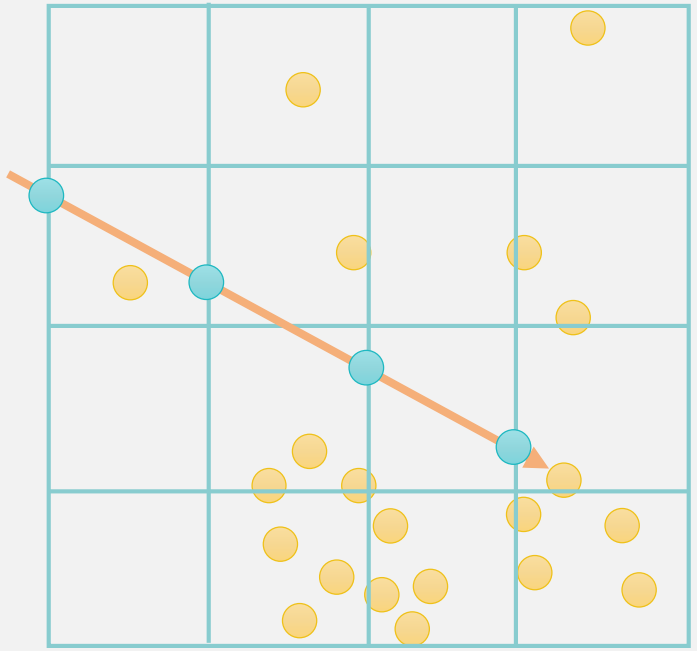
Photo by Amy Treasure

Acceleration Structures

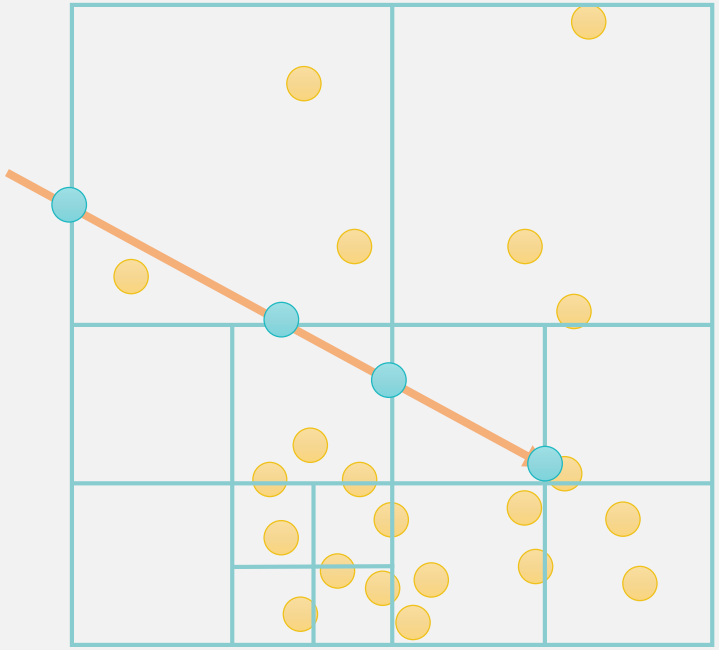
- Uniform grids
- Quadtree/Octree
- k-D tree
- BSP (Binary Space Partitioning) tree
- Bounding Volume Hierarchy (BVH)

Spatial Partition

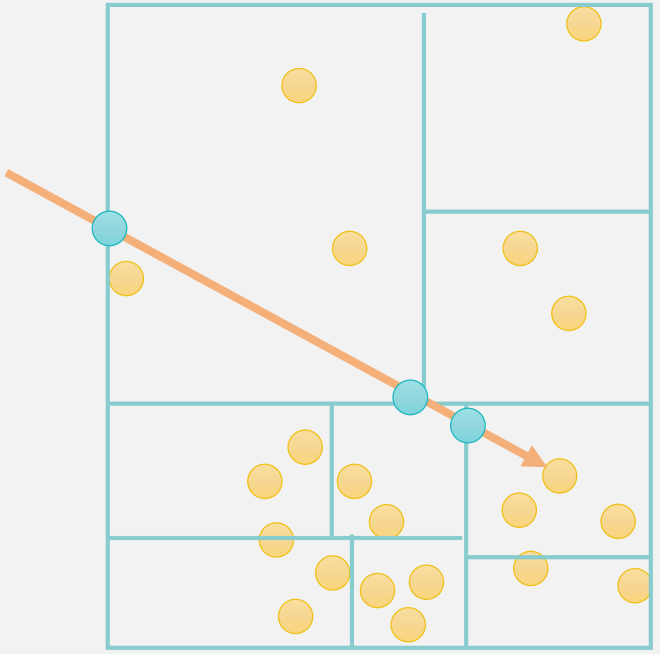
Uniform Grids



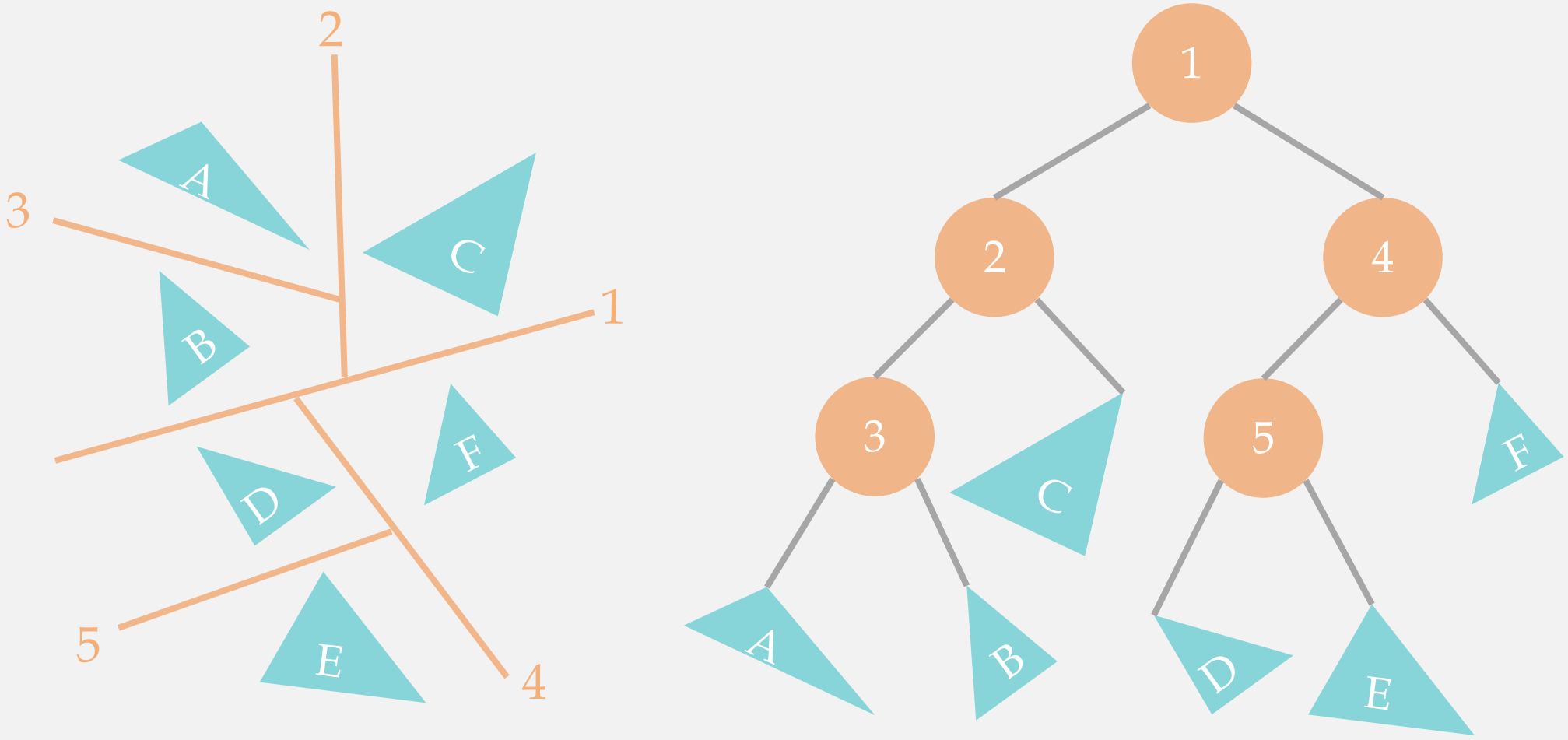
Quad Tree



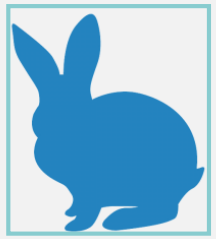
k-D Tree



Binary Space Partitioning Tree

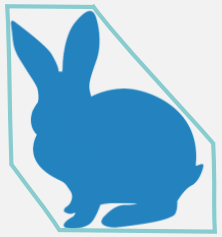


Types of Boundary Volumes



AABB

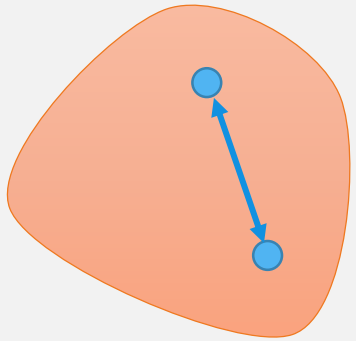
sphere



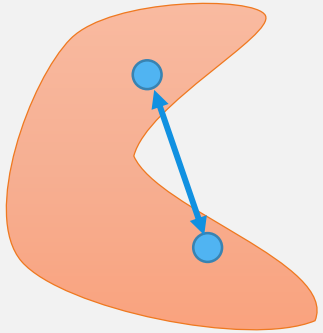
k-DOP

OBB

convex hull

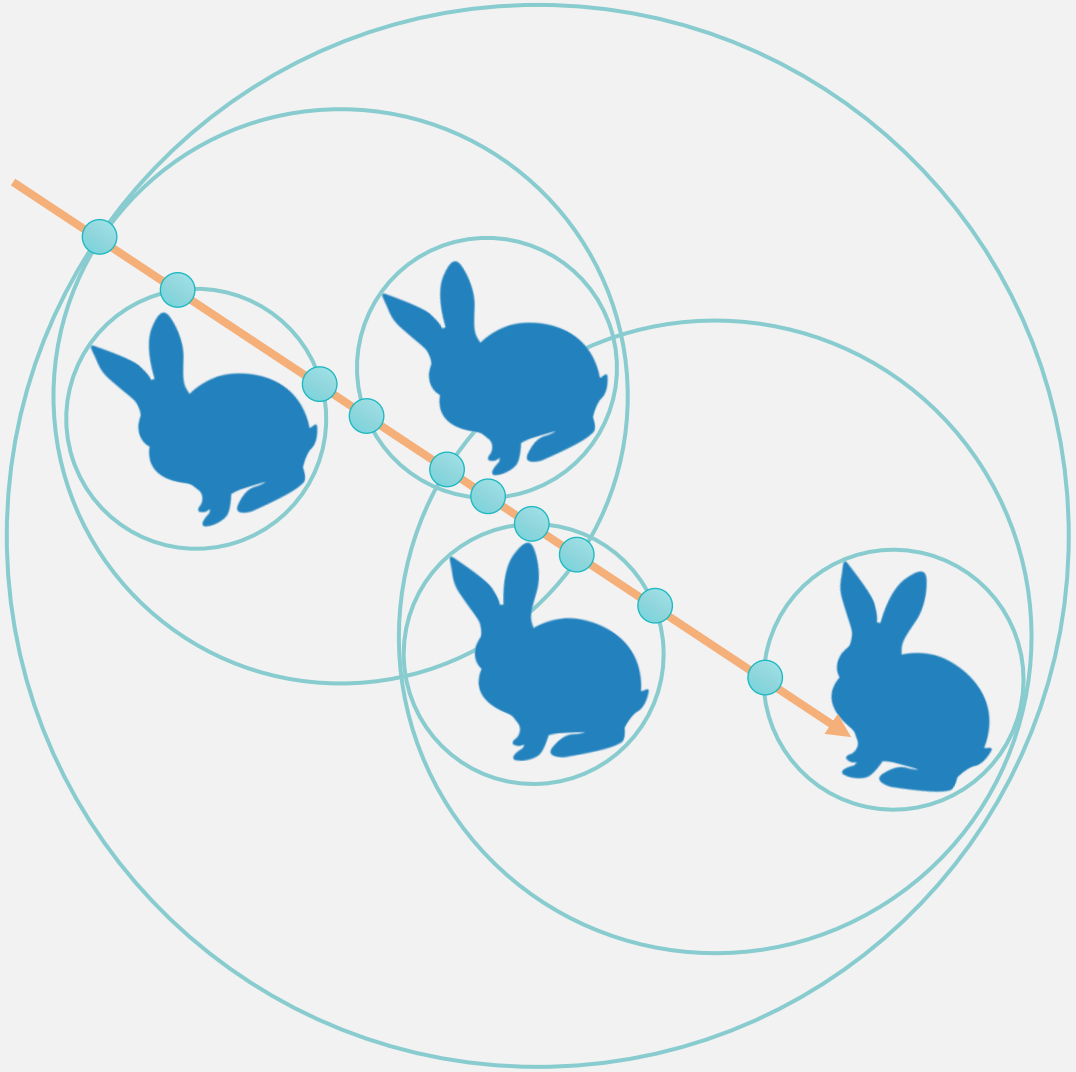


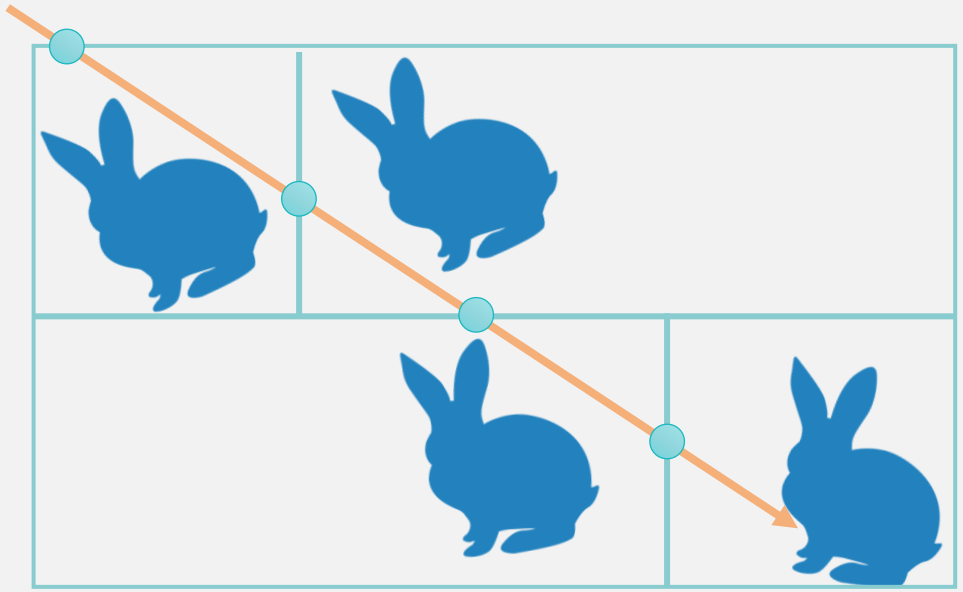
convex



concave

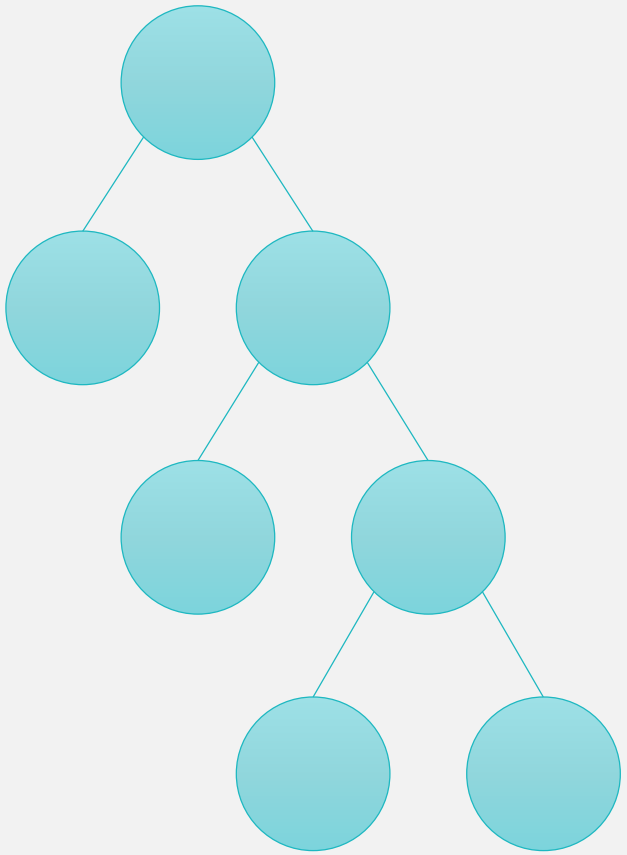
Hierarchy Traversal





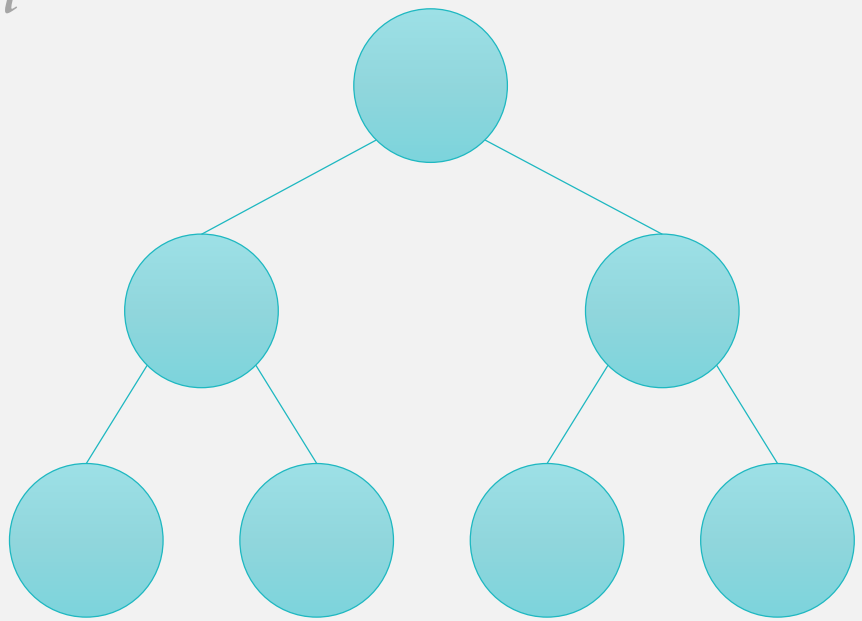
Balance = Query Performance

Imbalanced

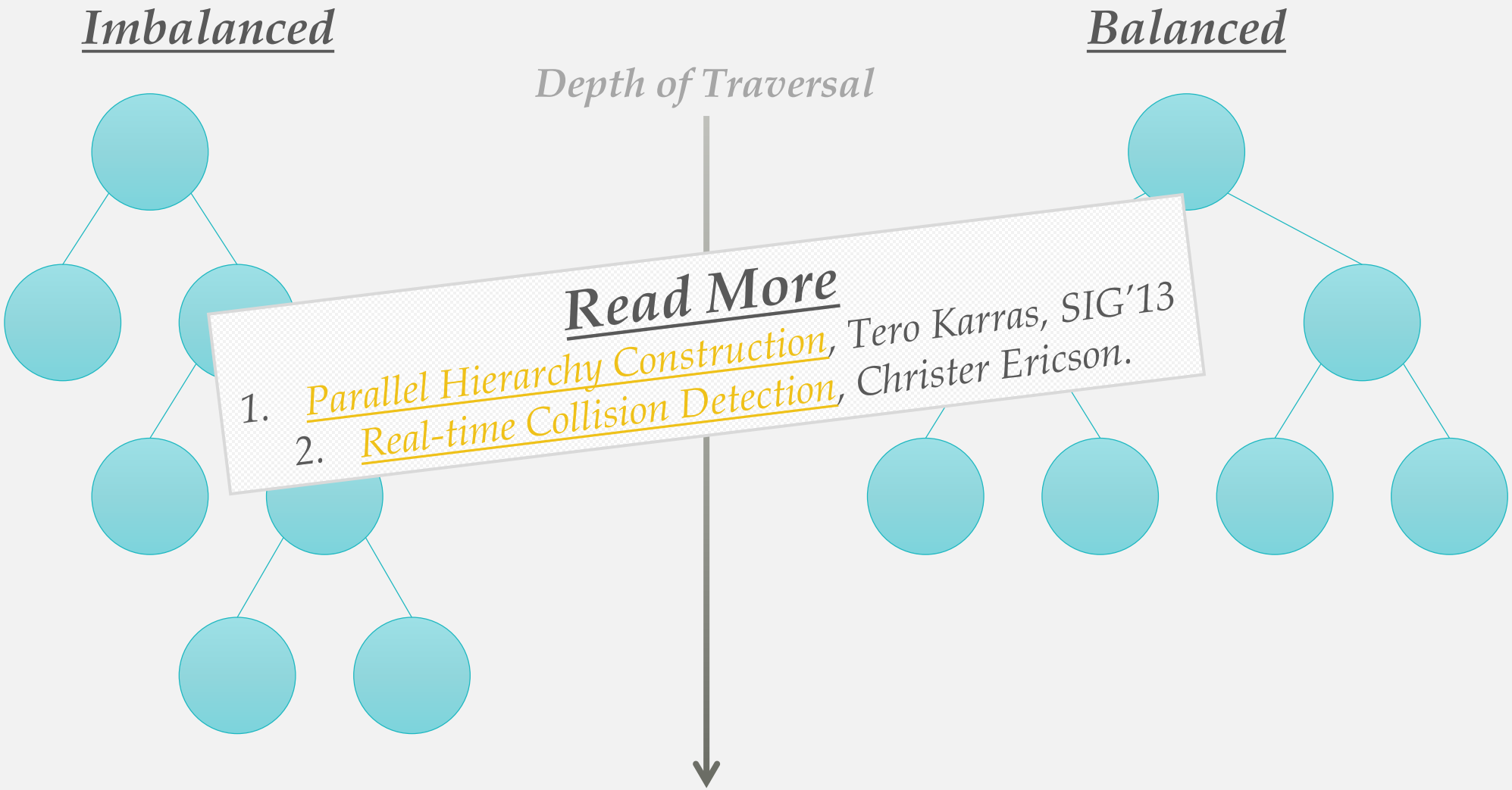


Depth of Traversal

Balanced



Balance = Query Performance



Object/Object Intersection

Last changed: April 20, 2017

This page gives a grid of intersection routines for various popular objects, pointing to resources in books and on the web. The most comprehensive books on the subject are *Geometric Tools for Computer Graphics* (GTCG) and *Real-Time Collision Detection* (RTCD); the former is all-encompassing, the latter more approachable and focused. A newer book focused in large part on object/object intersection tests is the *Game Physics Cookbook* (GPC), with code - see its giant grid for what intersections it covers. For a unified static and dynamic object intersection and distance library (non-commercial use only, though), see the TGS collision system.

Guide to source abbreviations:

- **3DG** - *3D Games: Real-time Rendering and Software Technology*, Alan Watt and Fabio Policarpo, Addison-Wesley, 2001.
- **GPC** - *Game Physics Cookbook*, by Gabor Szauer, Packt Publishing, March 2017, with code
- **GPG** - *Game Programming Gems*, ed. Mark DeLoura, Charles River Media, 2000.
- **GTCG** - *Geometric Tools for Computer Graphics*, Philip J. Schneider and David H. Eberly, Morgan Kaufmann Publishers, 2002. Good, comprehensive book on this topic.
- **Gems** - The *Graphics Gems* series. See the book's website for individual book links and code.
- **GTweb** - Geometric Tools, Dave Eberly's online computer graphics related software repository. His book *3D Game Engine Design* also covers these, in a readable format, as well as many other object/object intersection tests.
- **IRT** - *An Introduction to Ray Tracing*, ed. Andrew Glassner, Academic Press, 1989.
- **JCGT** - *The Journal of Computer Graphics Techniques*.
- **jgt** - *journal of graphics tools*. A partial code repository is available.
- **RTCD** - *Real-Time Collision Detection*, by Christer Ericson, Morgan Kaufmann Publishers, 2004.
- **RTR** - *Real-Time Rendering, Third Edition*, by Tomas Möller, Eric Haines, and Naty Hoffman, A.K. Peters Ltd., 2008.
- **RTR2** - *Real-Time Rendering, Second Edition*, by Tomas Akenine-Möller and Eric Haines, A.K. Peters Ltd., 2002.
- **SG** - Simple Geometry library, Steve Baker's vector, matrix, and quaternion manipulation library.
- **TGS** - Teikitu Gaming System Collision, Andrew Aye's object/object intersection/distance and sweep/penetration software (non-commercial use only).
- **TVCG** - IEEE Transactions on Visualization and Computer Graphics.

Individual article references follow after the table.

Static Object Intersections

Entries are listed from oldest to newest, so often the *last* entry is the best. This table covers objects not moving; see the next section for dynamic objects.

	ray	plane	sphere	cylinder	cone	triangle	AABB	OBB	frustum	polyhedron	
ray	Gems p.304; SG; TGS; RTCD p.198; SoftSurfer; RTR2 p.618; RTR3 p.781	IRT p.50,88; SG; GTCG p.482; TGS; RTCD p.175; SoftSurfer (more); GPC	IRT p.39,91; Gems p.388; Held jgt 2(4); GTweb; 3DG p.16; GTCG p.501; TGS; RTCD p.127,177; RTR2 p.568; RTR3 p.738; GPC	IRT p.91; Gems IV p.356; Held jgt 2(4); GTweb; GTCG p.507; TGS; RTCD p.194	IRT p.91; Gems V p.227; Held jgt 2(4); GTweb doc; GTCG p.512	Möller- Trumbore jgt 2(1): code (mirror), paper draft; IRT p.53,102; Gems IV p.24; Held jgt 2(4); GTweb; 3DG p.17; Möller (mirror); GTCG p.485; TGS; RTCD p.153,184; Löfstedt jgt 10(2): code, paper draft Chirkov jgt 10(3): code; Lagae jgt 10(4): code, paper draft; SoftSurfer; RTR2 p.578; RTR3 p.746; Havel TVCG June 2009;	IRT p.65,104; Gems p.395; Smits; 3DG p.20; Terdiman (optimized Woo); Schroeder; GTCG p.626; TGS; RTCD p.179; Mahovsky jgt 9(1); Williams jgt 10(1) (code); Eisemann jgt 12(4) (code); RTR2 p.572; RTR3 p.742; Shirley 2016; GPC	(IRT p.104; Gems II p.247); GTweb; Gomez; GTCG p.630; TGS; RTCD p.179; RTR2 p.572; RTR3 p.743; GPC	(IRT p.104; Gems II p.247)	IRT p.104; Gems II p.247; GTCG p.493; Platis jgt 8(4); RTCD p.198; SoftSurfer	**ray**

Practical Issues

- Construction costs in space and time
  - Use *float* for scalar data instead of double
  - Pointers are costly in x64 platform
    - Might point to incontiguous memory location in heap
    - To save storage, try using *int32 or short* for indexing
  - Geometry compression?
    - Unit vector quantization
    - Store local position in float, only use double for their transform matrix
- ***Parallelism and locality*** are key factors for parallel processing

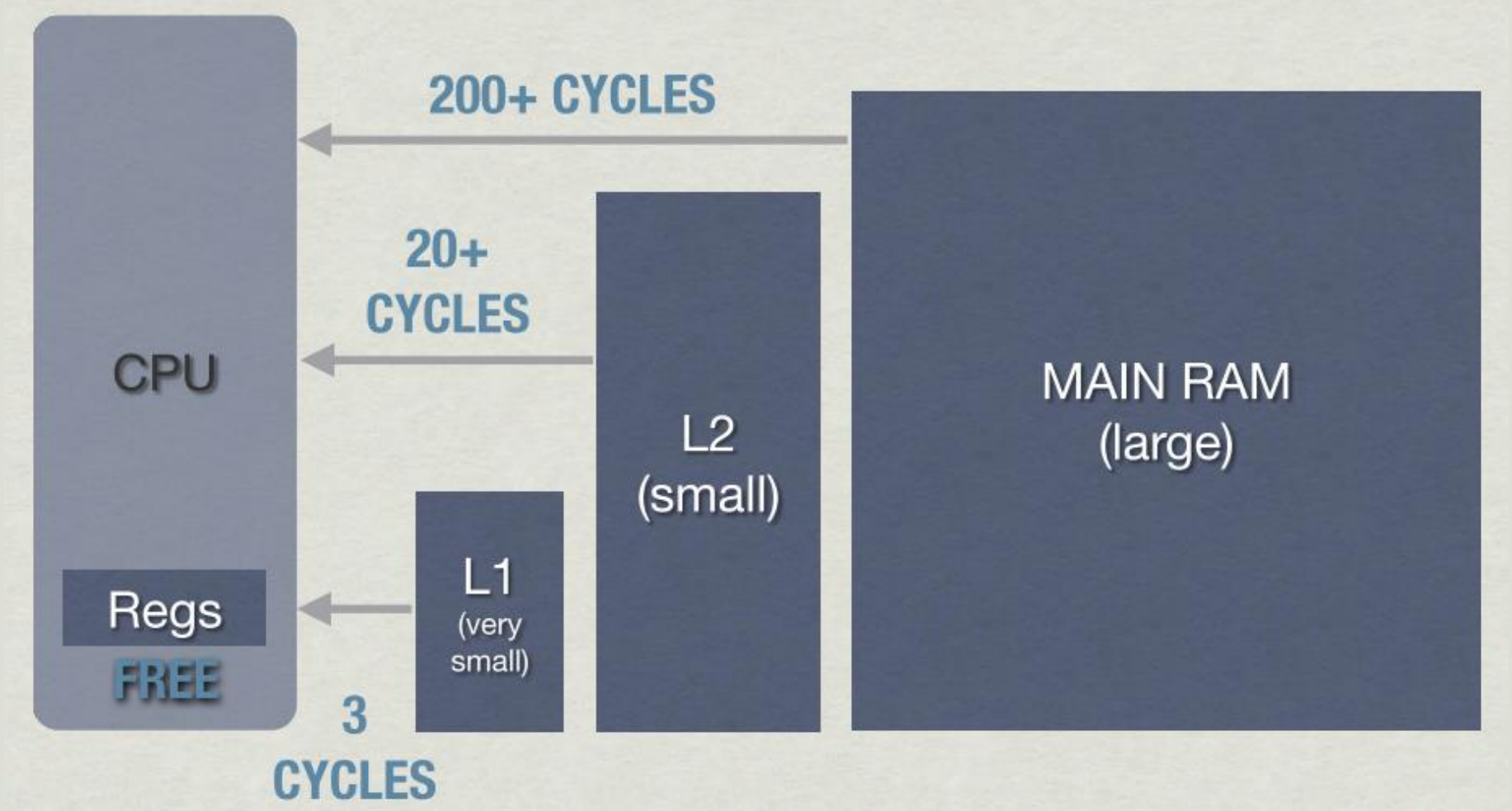
Practical Issues

- Construction costs in space and time
  - Use *float* for scalar data instead of double
  - Pointers are costly in x64 platform
    - Might point to incontiguous memory location in heap
    - To save storage, try using *int32*
  - C
  - M, only use double for their transform matrix
- ***Parallelism and locality*** are key factors for parallel processing

Read More

1. *A Survey of Efficient Representations*, Zina H. Cigolle et al., JCGT'14.
2. *Geometry Compression*, Michael Deering, Computer Graphics '95.

Memory Caching



Dogged Determination: Technology and Process at Naughty Dog Inc., Jason Gregory. SINFO XXI 2014 Keynote

Cache Line

- The fixed size data block transferred between memory and cache
- Might take hundreds of clocks to move around
- *False sharing*
  - Different threads access elements which reside in the same cache line

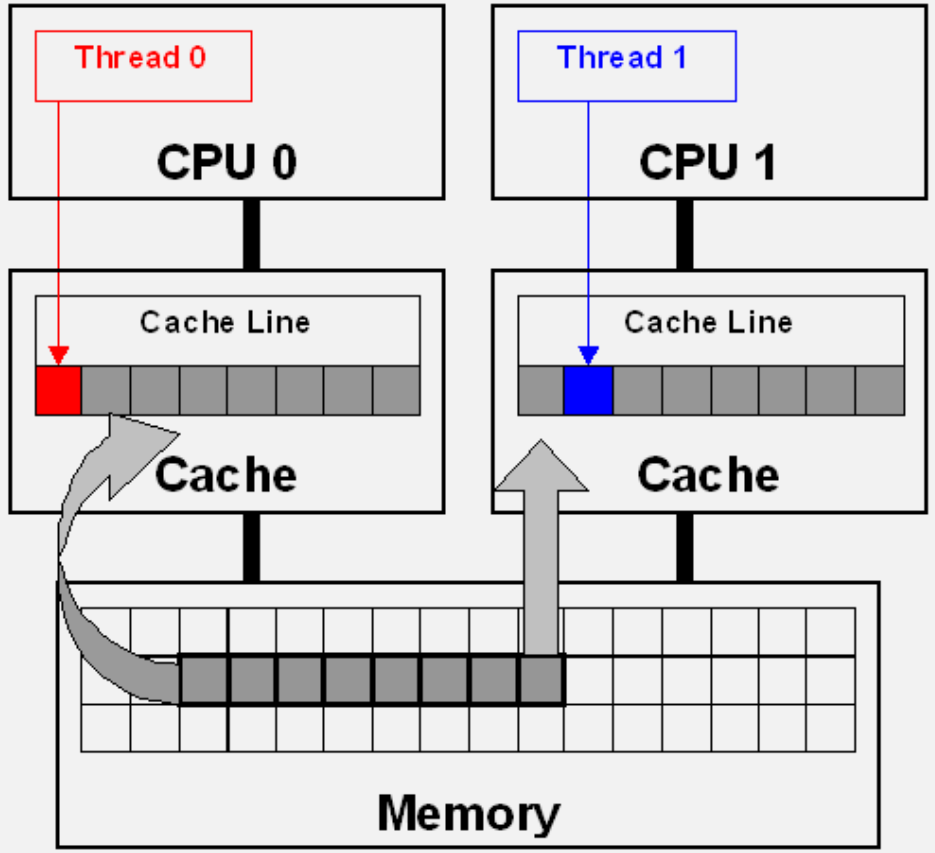
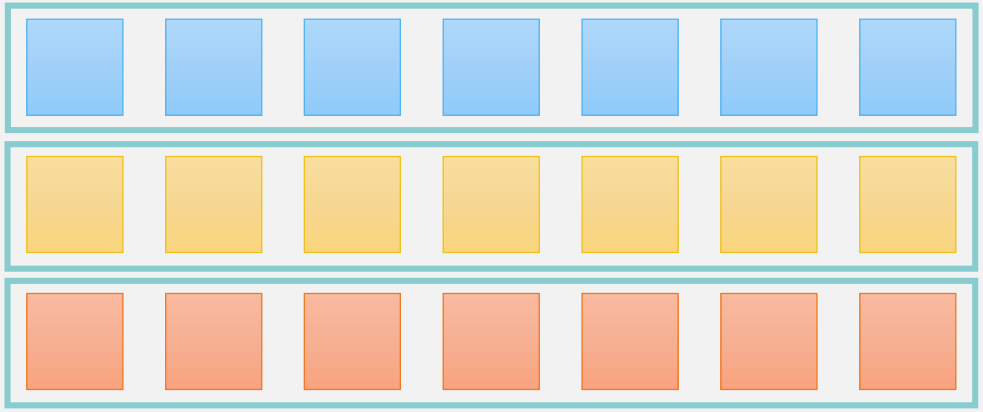


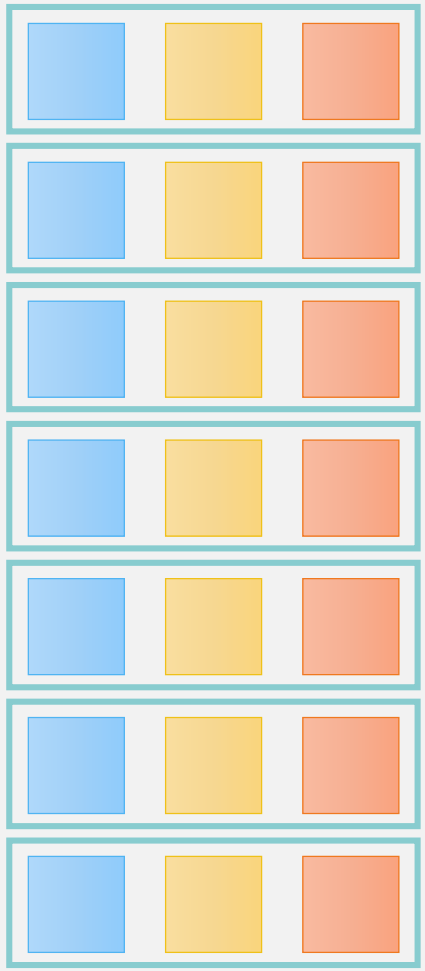
figure from https://software.intel.com

SOA vs. AOS



Structures of Arrays (SOA)

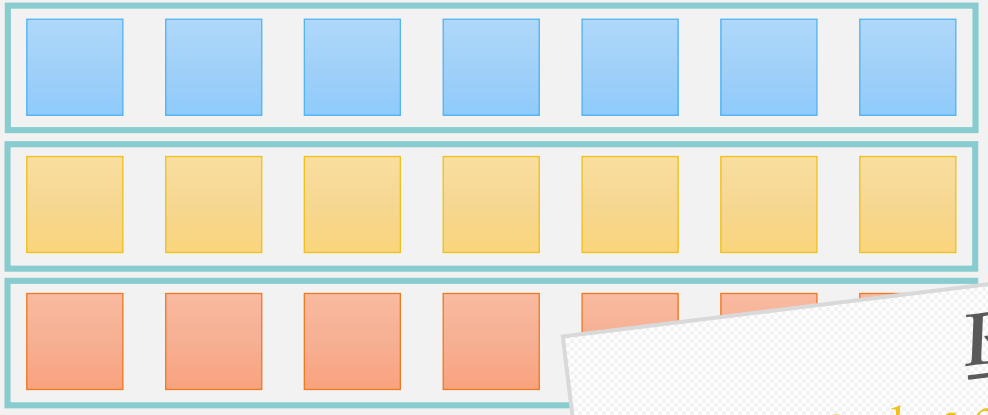
- Easily aligned cache boundaries
- Easier to utilize SIMD
- Chance for hardware prefetching



Array of Structures (AOS)

- Intuitively match the object abstraction
- Might cause cache alignment problems
- Hard to vectorized

SOA vs. AOS



Structures of Arrays

- Easily aligned cache boundaries
- Easier to utilize SIMD
- Chance for hardware prefetching

Array of Structures (AOS)

- Intuitively match the object
  straction
  ght cause cache
  gnment problems
  Hard to vectorized

Read More

1. *CPU Caches and Why You Care*, Scott Meyers.
2. *Cache Aware Components*, Randy Gaul.

YMMV

Object-Oriented or Data-Oriented Design?

- Abstraction is good for modeling
  - But over-abstraction is harmful for performance
- Memory access pattern is crucial for parallel processing
  - *Structure of Arrays* (SOA) vs. *Array of Structures* (AOS)
  - Hot/cold splitting
- 80/20 principle
  - Optimizing after profiling!!
  - Don't optimizing the insignificant parts

YMMV

Object-Oriented or Data-Oriented Design?

- Abstraction is good for modeling
  - But over-abstraction is harmful for performance
- Memory access pattern is crucial for parallel processing
  - *Structure of Arrays* (SOA) vs. *Array of St*
  - Hot/cold
- 80/20 pr
  - Optimiz
  - Don't optimizing the insignificant parts

Read More

1. *Data Oriented Design*, Richard Fabian.
2. *Data-Oriented Design and C++*, Mike Acton, CppCon'14.

Profiling!!

Light Transport Techniques

Recap

Radiometry

Radiant flux $\Phi = \frac{\mathrm{dQ}}{\mathrm{dt}}$ (J/sec)

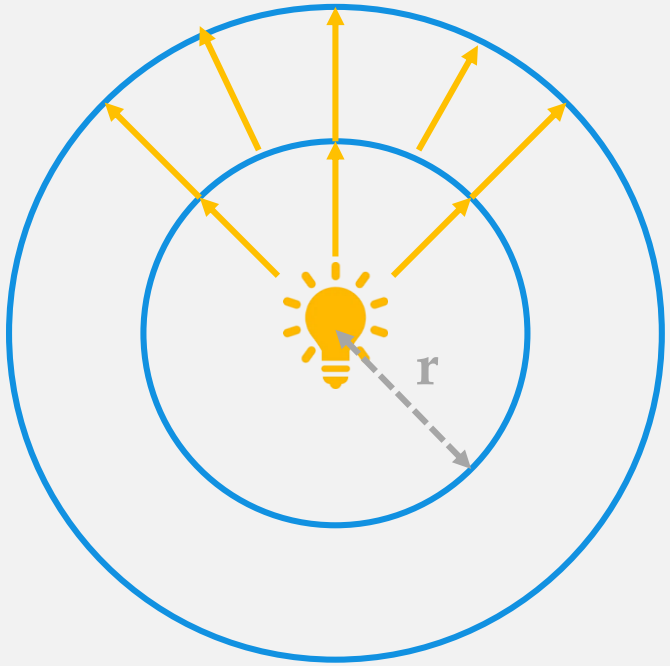
The total amount of energy passing through a region of surface per unit time

Irradiance $E = \frac{d\Phi}{dA}$

Pre area incoming flux at a surface

Radiant Exitance or *Radiosity* $M = B = \frac{d\Phi}{dA}$

$$E = \frac{\Phi}{4\pi r^2}$$



the total amount Φ measured at inner and outer sphere is the same (equals to the radiant flux of the point light)

Recap

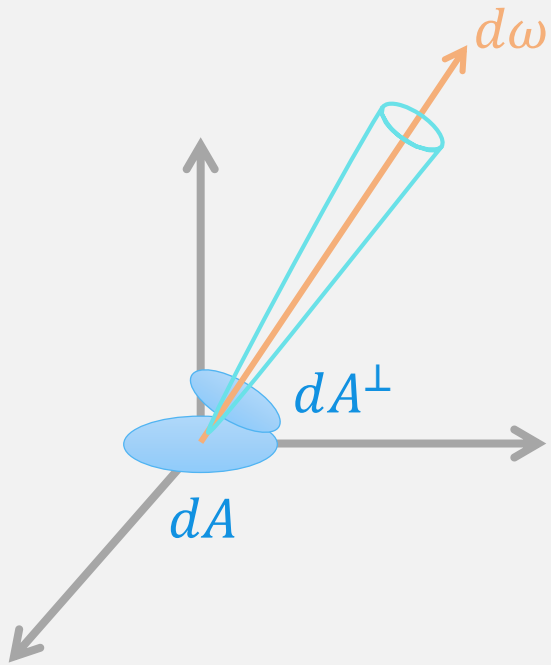
Radiance

$$L = \frac{d^2\Phi}{d\omega dA^{\perp}} = \frac{d^2\Phi}{d\omega dA\cos\theta}$$

flux

solid angle

projected area



$$L\cos\theta = \frac{d^2\Phi}{dAd\omega} = \frac{dE}{d\omega} \Rightarrow L\cos\theta\, d\omega = dE$$

Recap

Albedo

- Hemispherical-directional reflectance (aka. black sky albedo)

$$\rho_{hd}(\omega_o) = \int_{\Omega} f_r(\omega_o, \omega_i)|\cos\theta_i| d\omega_i$$

- Hemispherical-hemispherical reflectance (aka. white sky albedo)
  - The ratio of total reflected energy to total incident energy

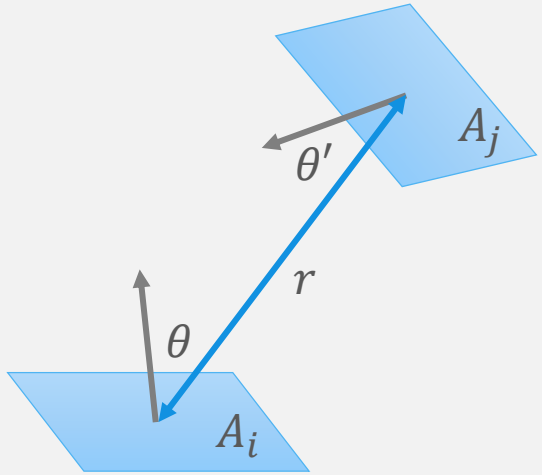
$$\rho_{hh} = \frac{1}{\pi}\int_{\Omega}\int_{\Omega} f_r(\omega_o, \omega_i)|\cos\theta_o \cos\theta_i| d\omega_o d\omega_i$$

Light Transport Techniques

- Radiosity
- Ray Tracing
  - Path tracing
  - Light tracing
  - Bidirectional path tracing
- Photon Mapping
- Irradiance Caching
- Point-Based Global Illumination (PBGI)
- VCM (Vertex Connection & Merging)

Radiosity

- Originated from heat transfer in physics
- Only account for diffuse-diffuse interactions
  - Assume opaque Lambertian surface
  - Divide scene into patches
  - Update radiosity patch by patch
  - Radiosity is constant within any patch
  - View-independent calculation



- Form factor F_{ji} the fraction of energy from A_j to A_i

$$A_iB_i = A_iE_i + \rho_i \sum_{j=1}^{n} F_{ji}A_jB_j \qquad A_jF_{ji} = \int_{x\in P_i}\int_{y\in P_j} \frac{\cos\theta\cos\theta'}{\pi r^2} v(x,y)dydx = A_iF_{ij}$$

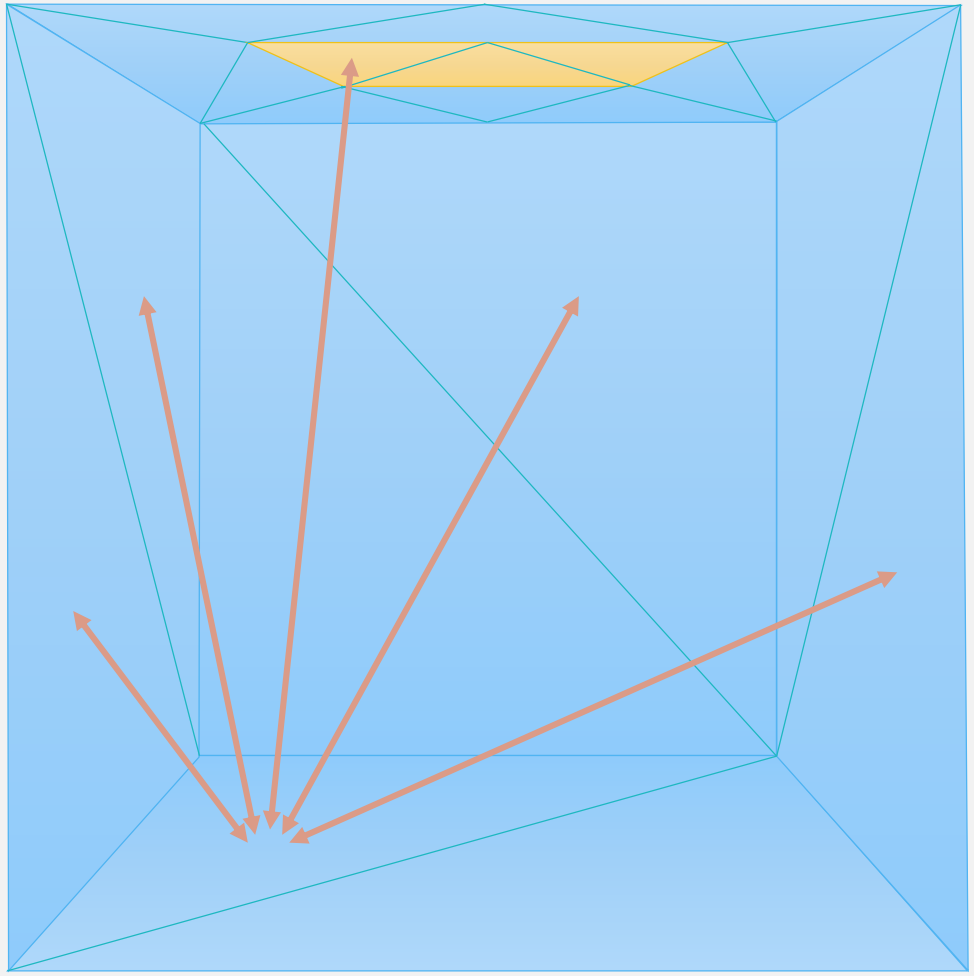
$$B_i = E_i + \rho_i \sum_{j=1}^{n} \left(\frac{F_{ji}A_j}{A_i}\right) B_j = E_i + \rho_i \sum_{j=1}^{n} F_{ij}B_j$$

Area: A
Radiosity: B
Radiant exitance: E
Reflectance: ρ
Form factor: F_{ij}
Visibility: $v(x,y)$

Radiosity (Cont.)

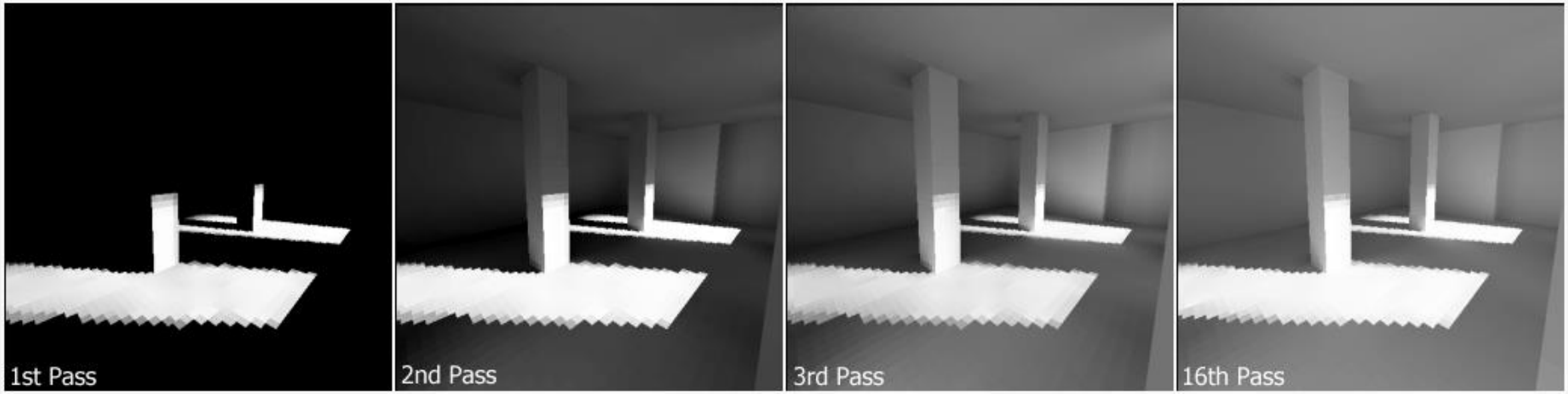
$$B_i - \rho_i \sum_{j=1}^{n} F_{ij} B_j = E_i$$

$$\begin{pmatrix} 1-\rho_1 F_{11} & -\rho_1 F_{12} & \cdots & -\rho_1 F_{1n} \\ -\rho_2 F_{21} & 1-\rho_2 F_{22} & \cdots & -\rho_2 F_{2n} \\ \vdots & \vdots & & \vdots \\ -\rho_n F_{n1} & -\rho_n F_{n2} & \cdots & 1-\rho_n F_{nn} \end{pmatrix} \begin{pmatrix} B_1 \\ B_2 \\ \vdots \\ B_n \end{pmatrix} = \begin{pmatrix} E \\ E_2 \\ \vdots \\ E_n \end{pmatrix}$$



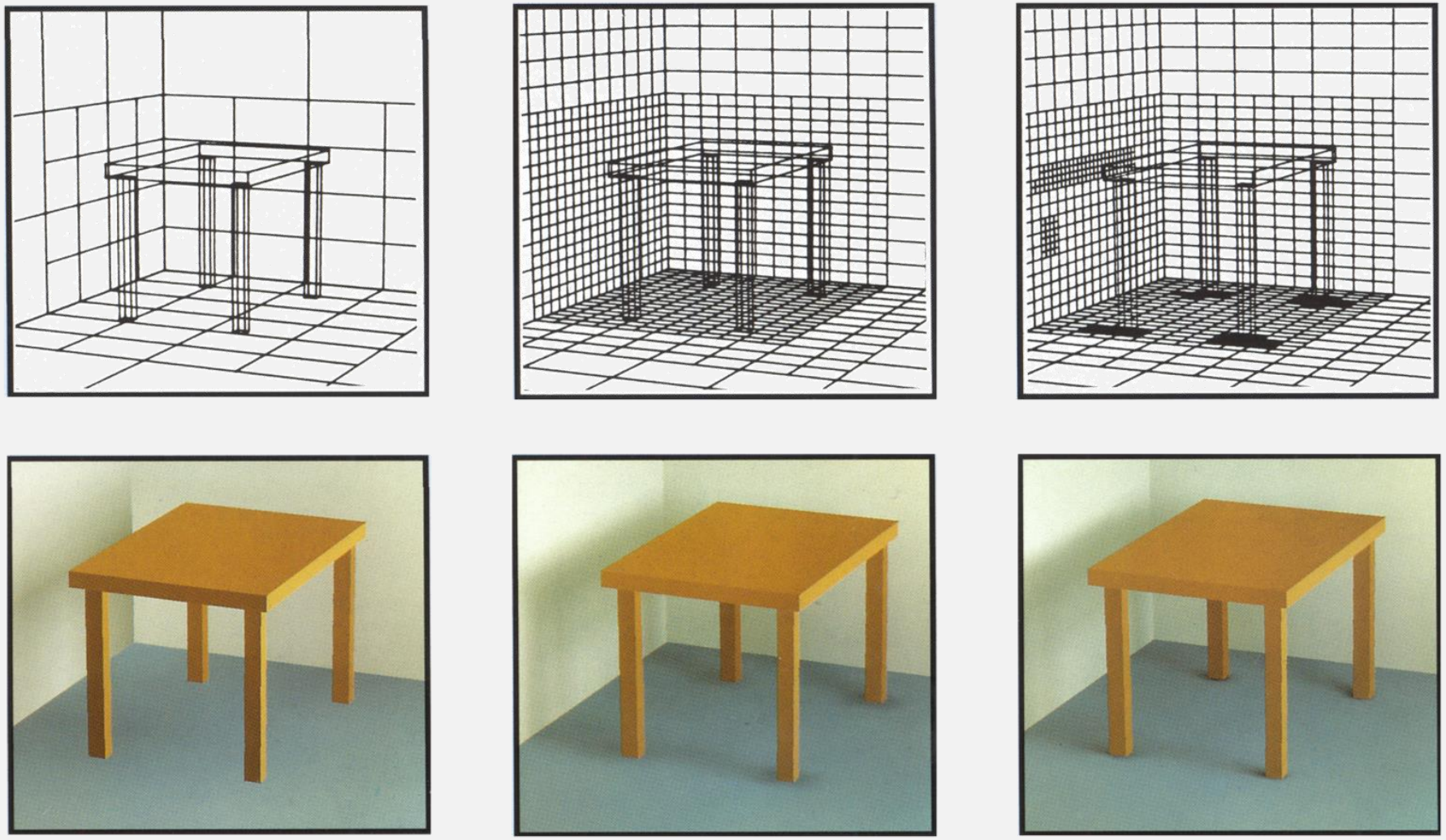
Radiosity (Cont.)

- The computation time is dominated by the number of patches
- The render quality is controlled by patch size and shape
- The mesh is getting complex for capturing shadow boundaries

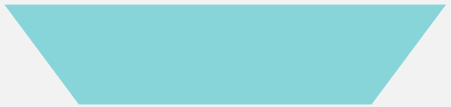


https://en.wikipedia.org/wiki/Radiosity_(computer_graphics)#/media/File:Radiosity_Progress.png

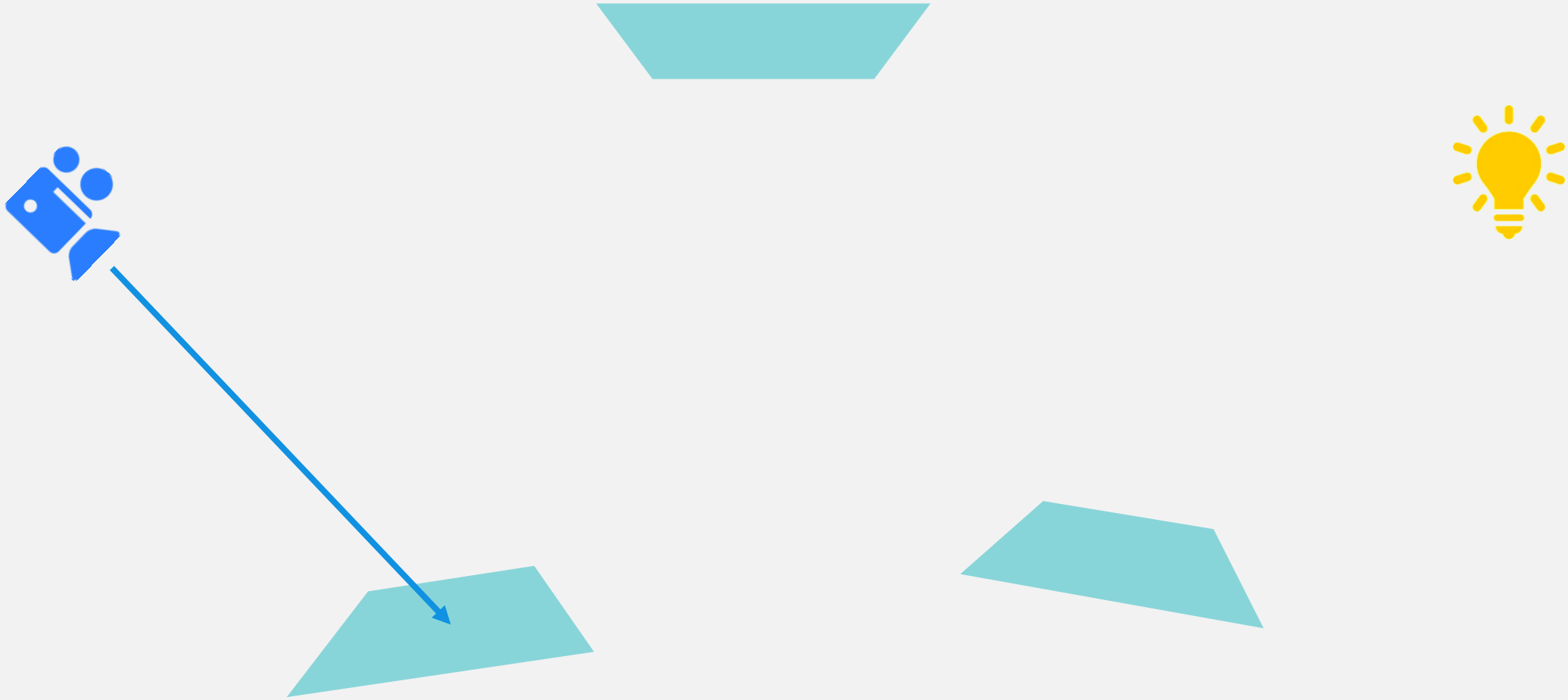
Radiosity – Adaptive Mesh Subdivision



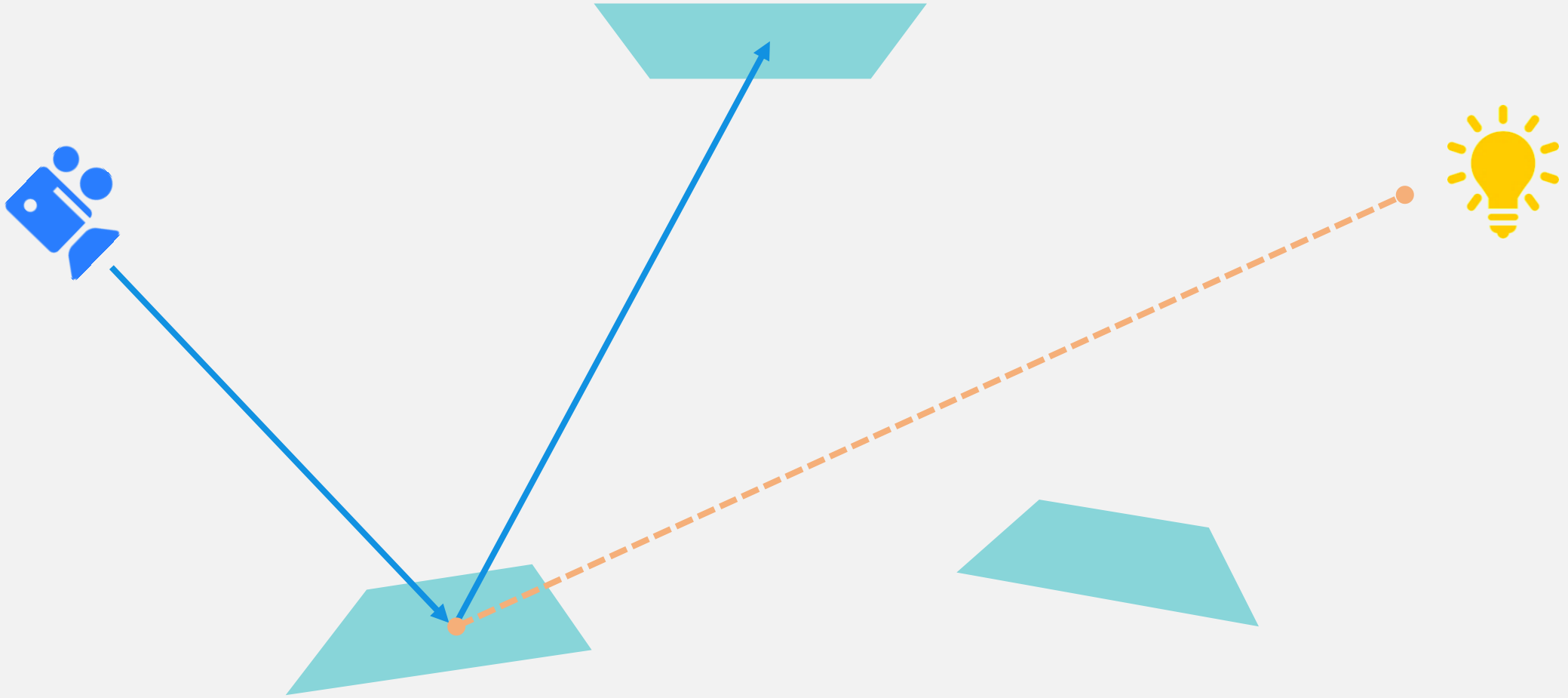
Path Tracing



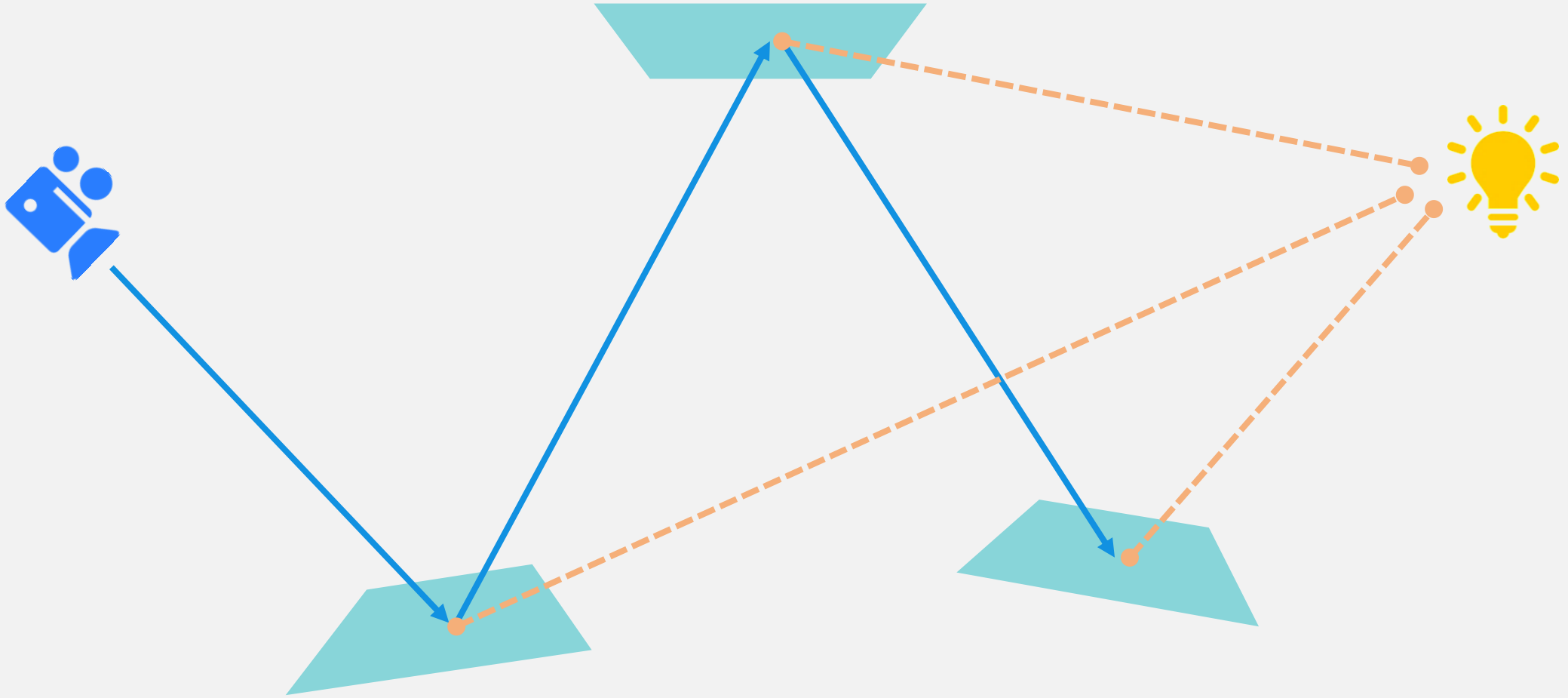
Path Tracing



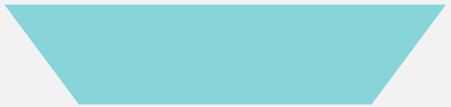
Path Tracing



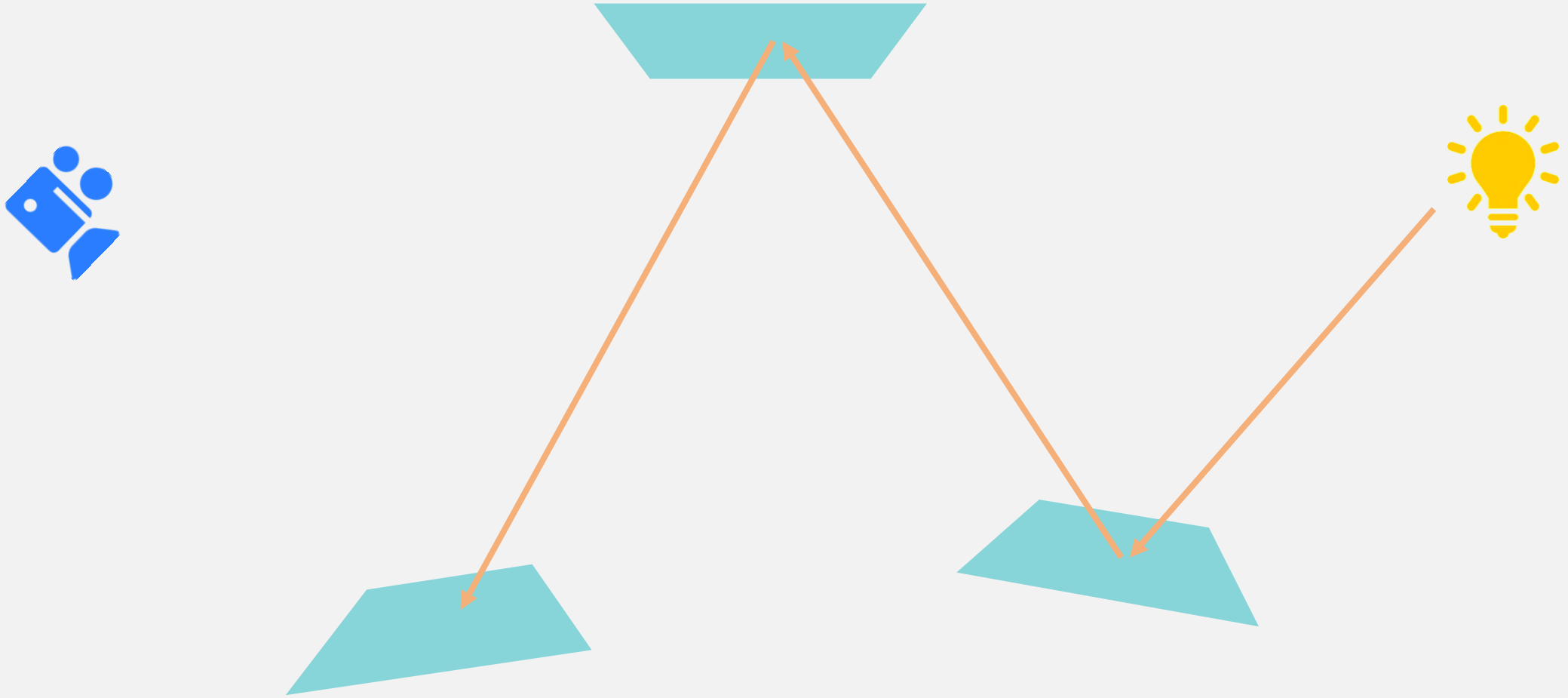
Path Tracing



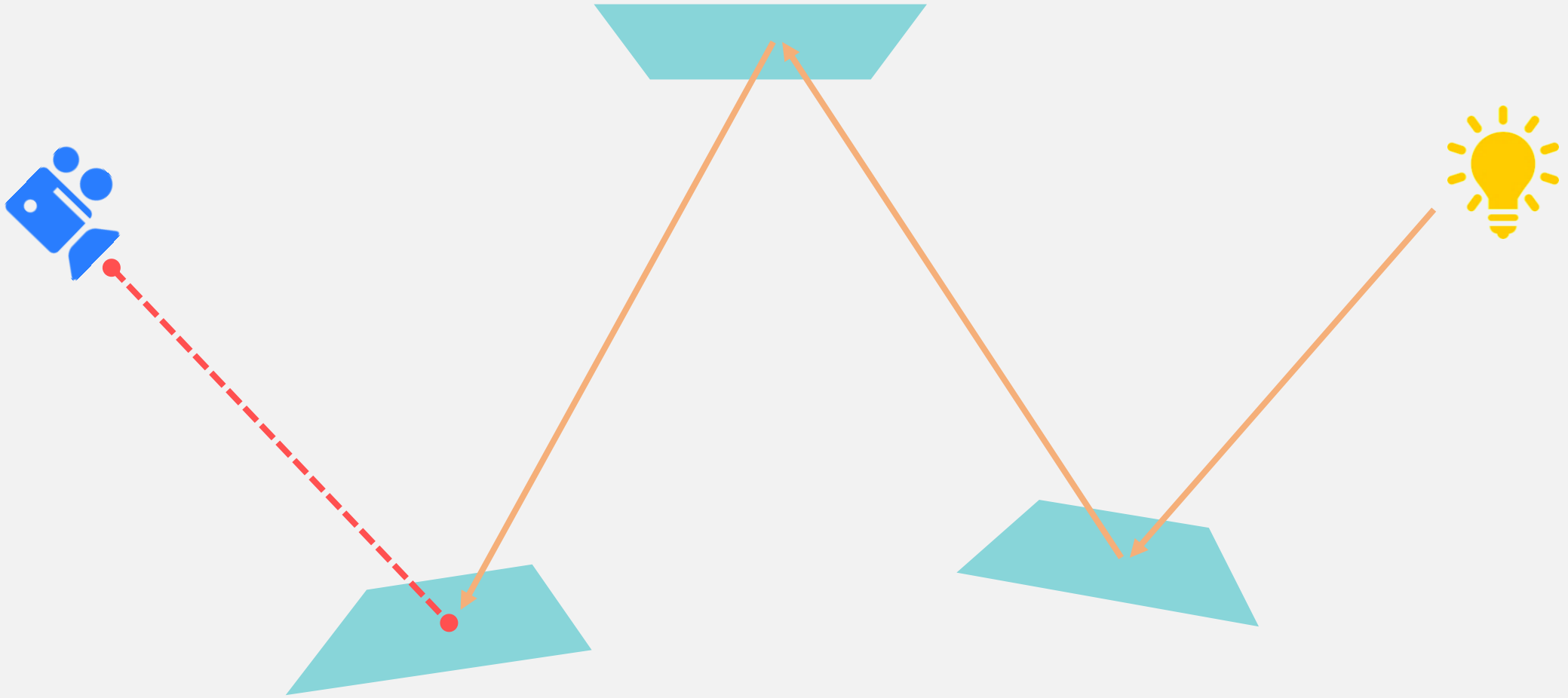
Light Tracing



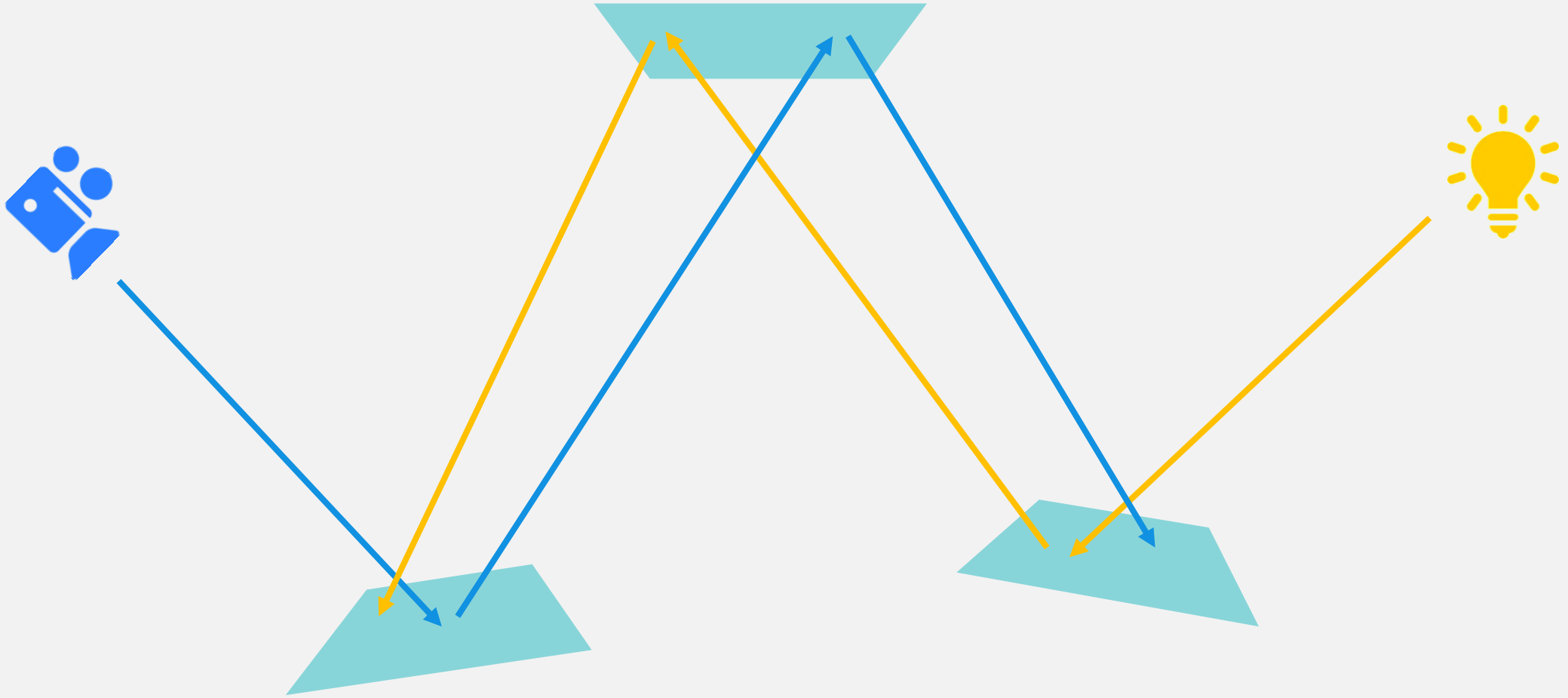
Light Tracing



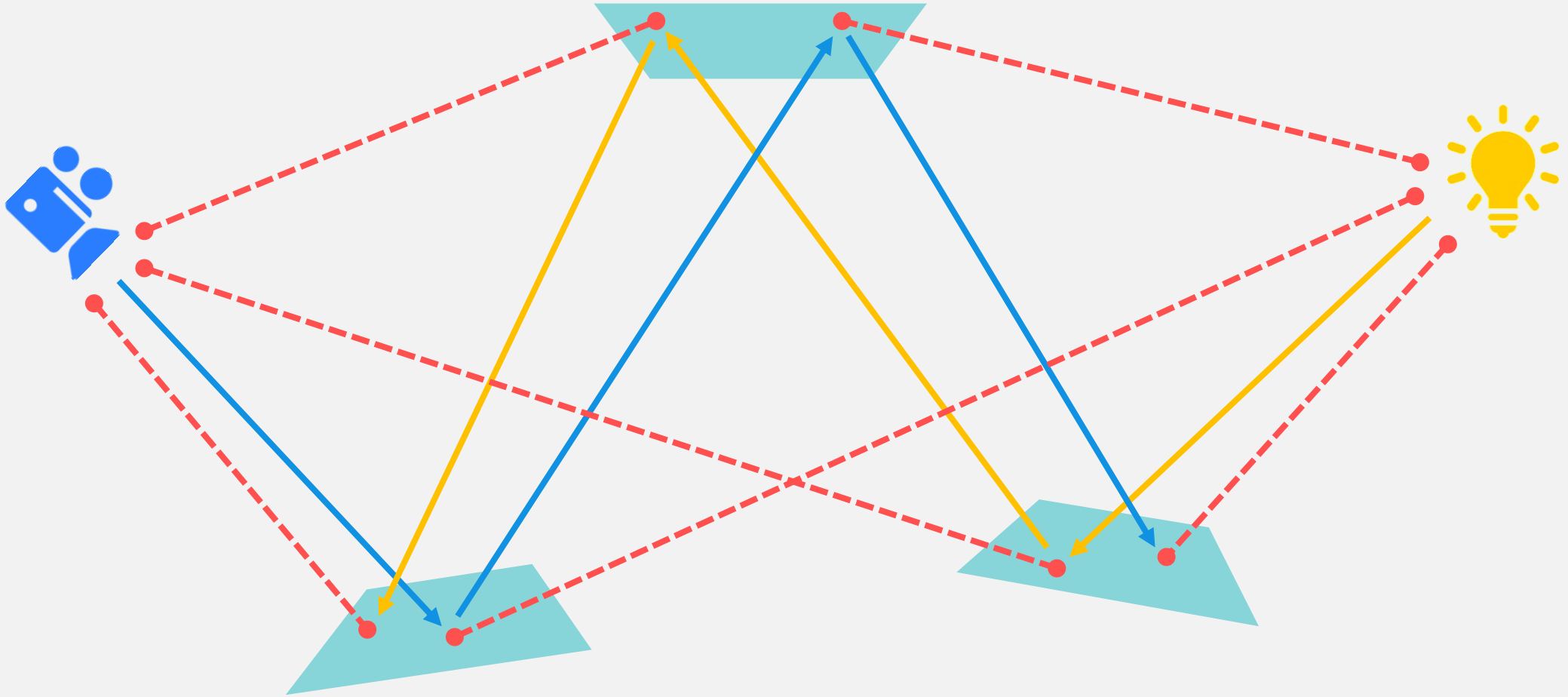
Light Tracing

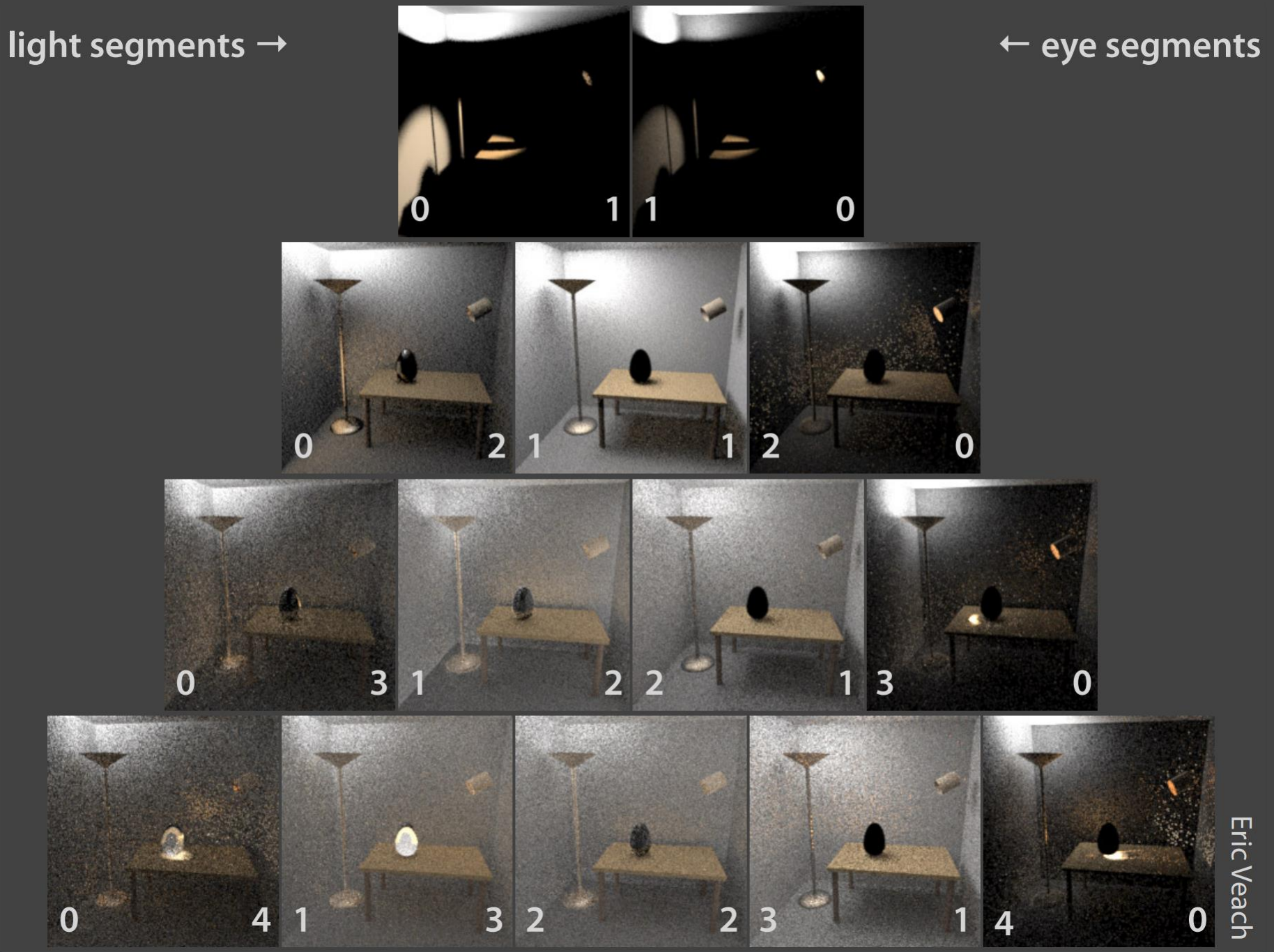


Bidirectional Path Tracing

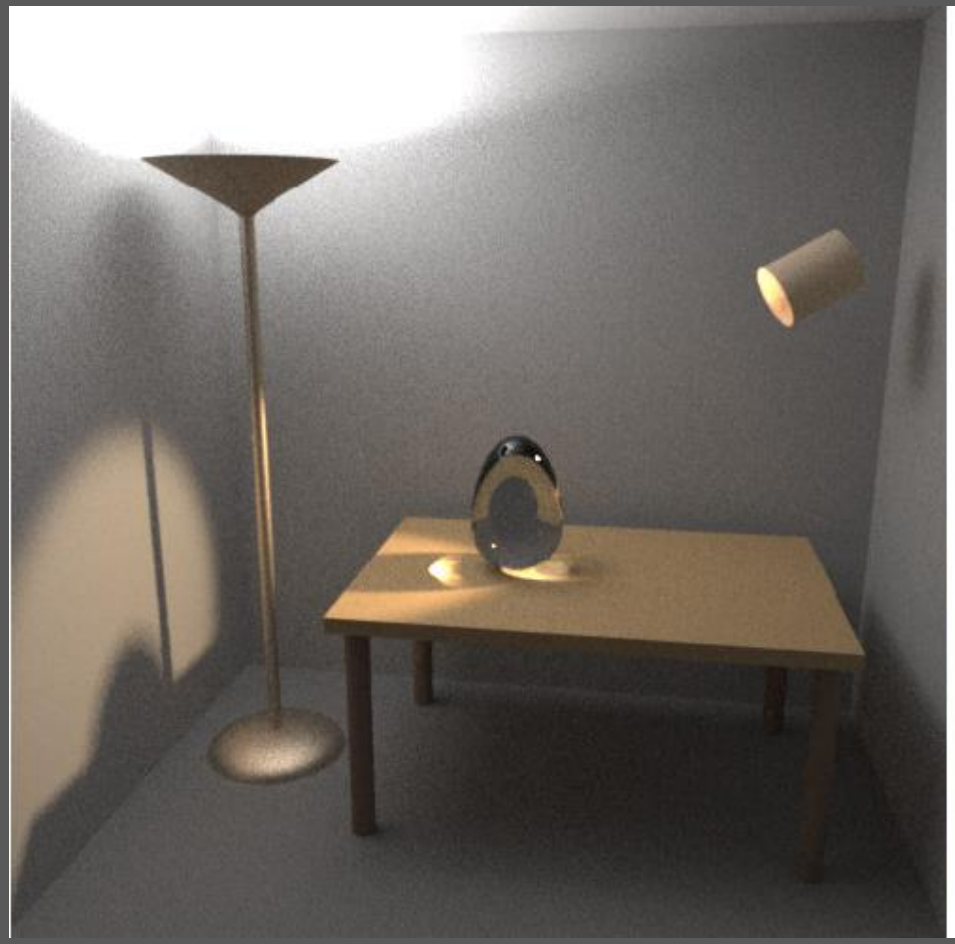


Bidirectional Path Tracing




light segments →
← eye segments
0 1
1 0
0 2
1 1
2 0
0 3
1 2
2 1
3 0
0 4
1 3
2 2
3 1
4 0
Eric Veach

Bidirectional Path Tracing (Cont.)

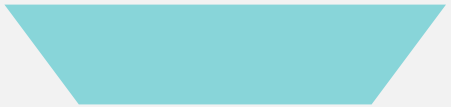


Bidirectional Path Tracing, 25 spp

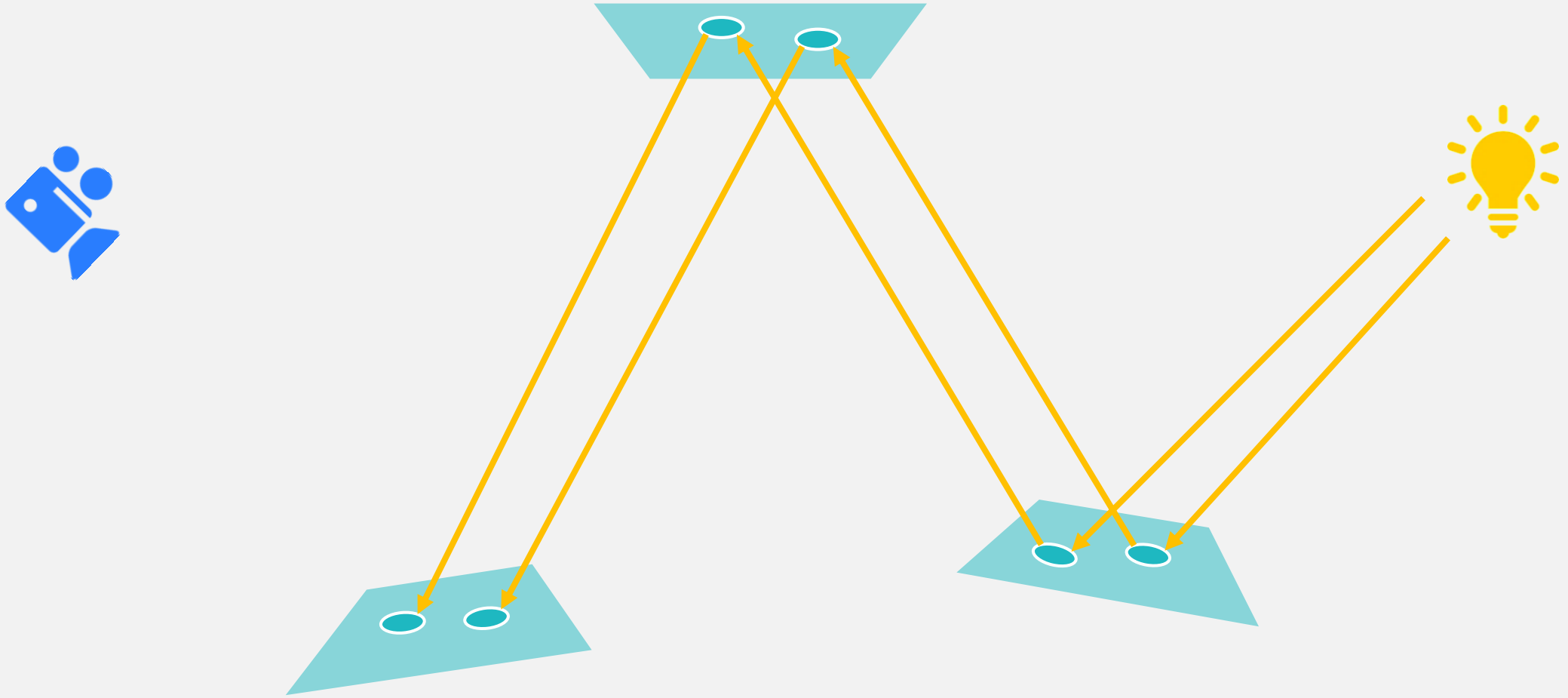
Path Tracing, 56 spp

[Veach, 04]

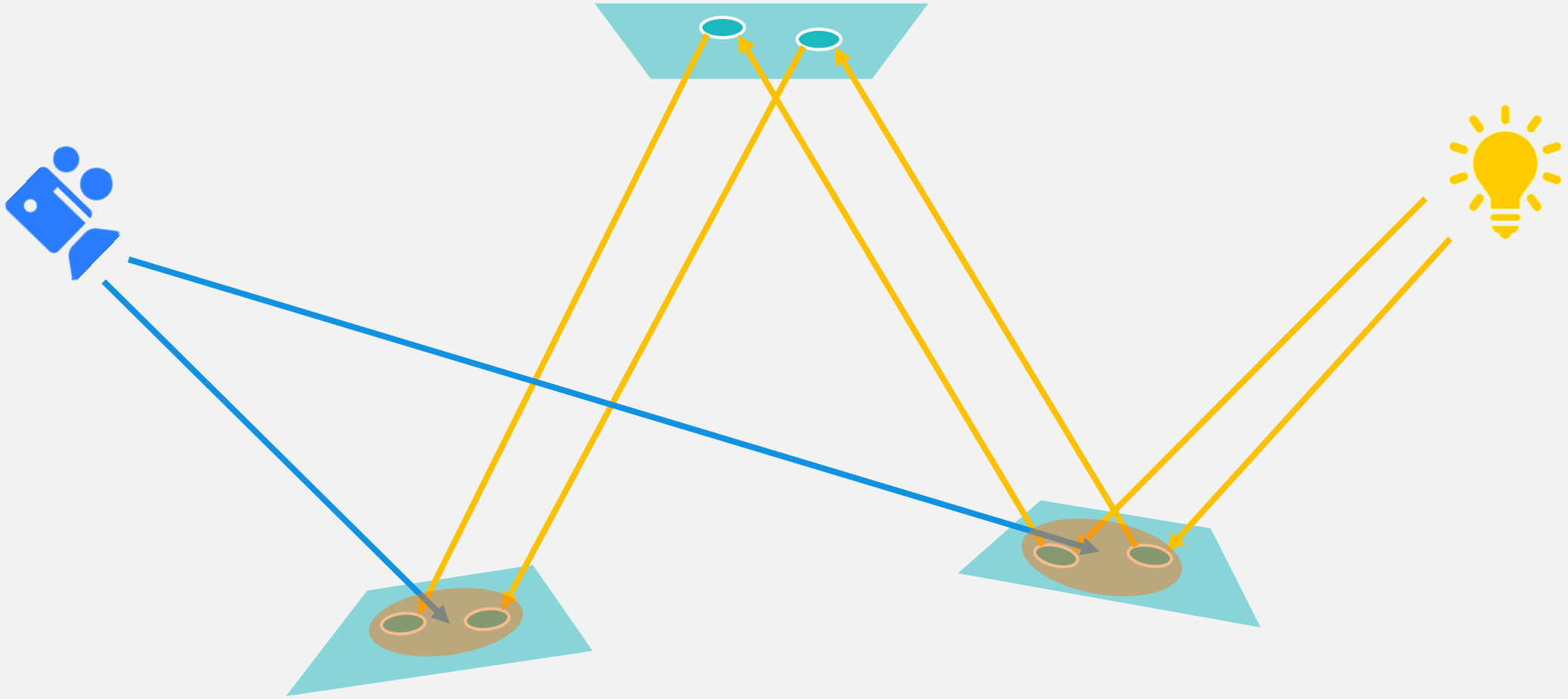
Photon Mapping

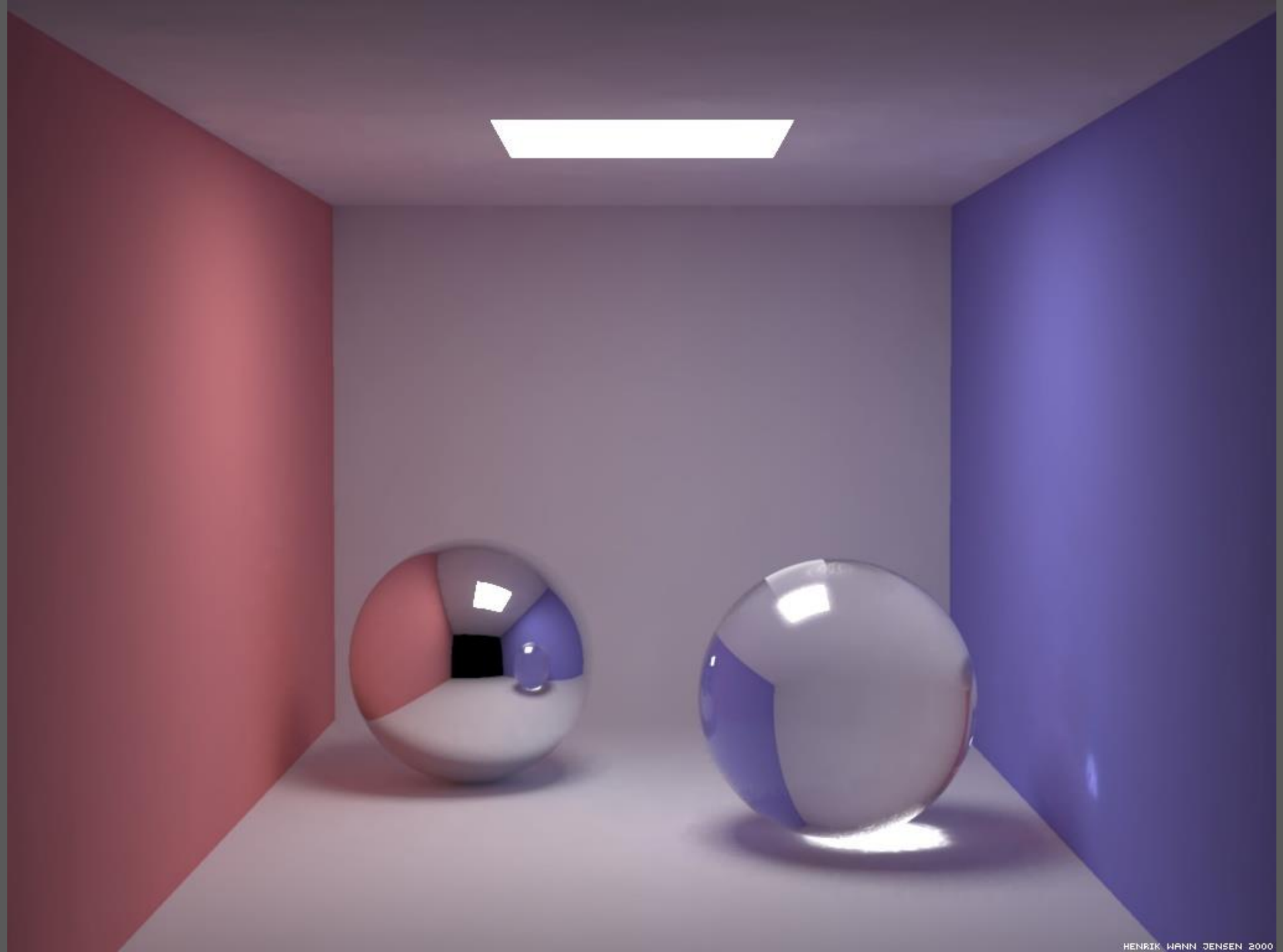


Photon Mapping



Photon Mapping




HENRIK WANN JENSEN 2000

Specular-Diffuse-Specular Caustic Path

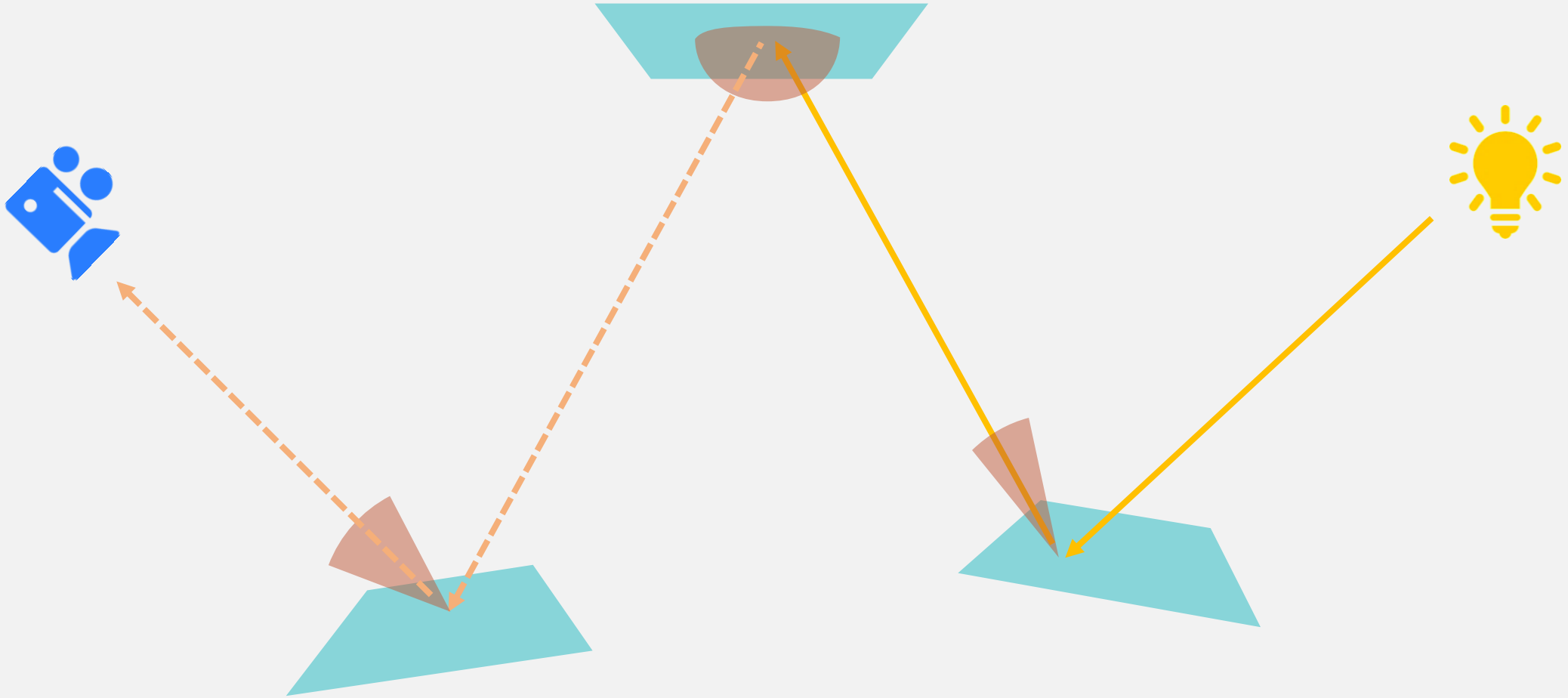
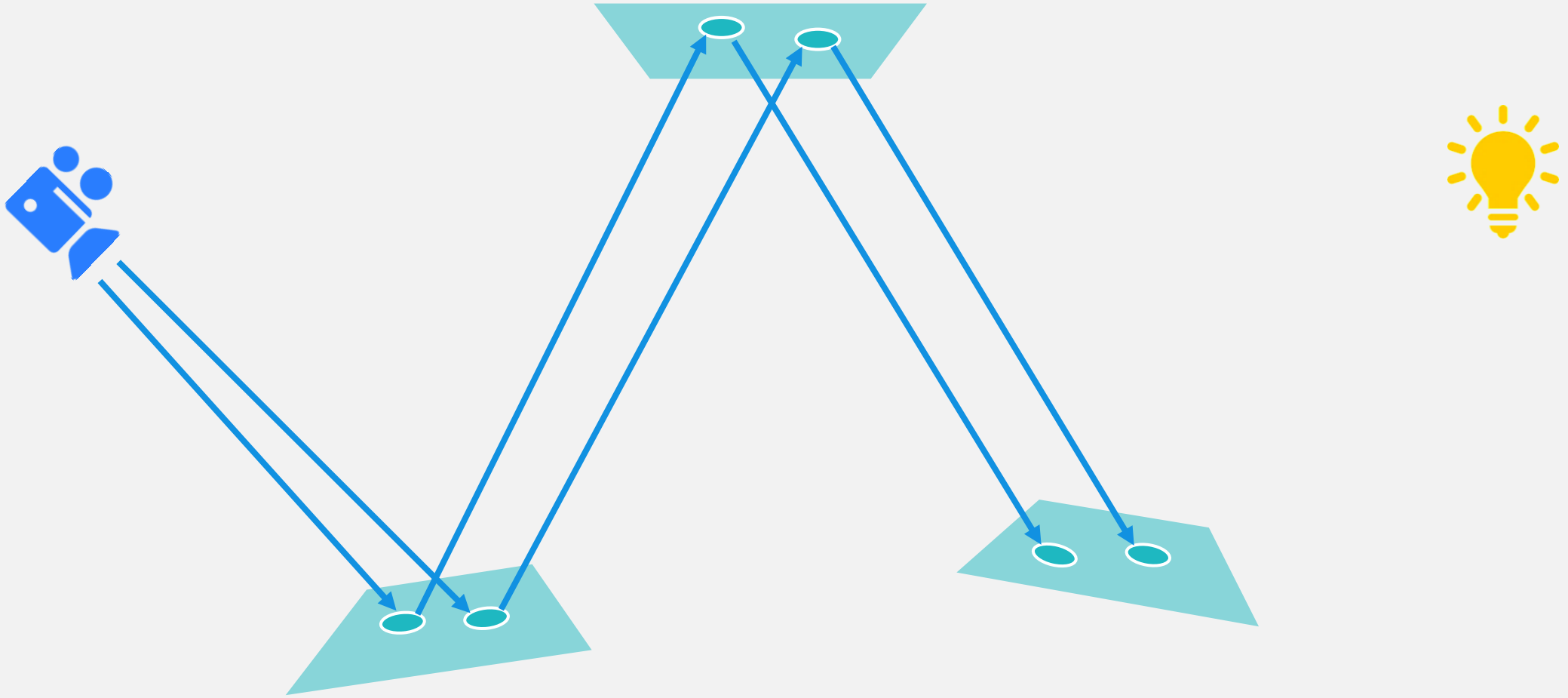


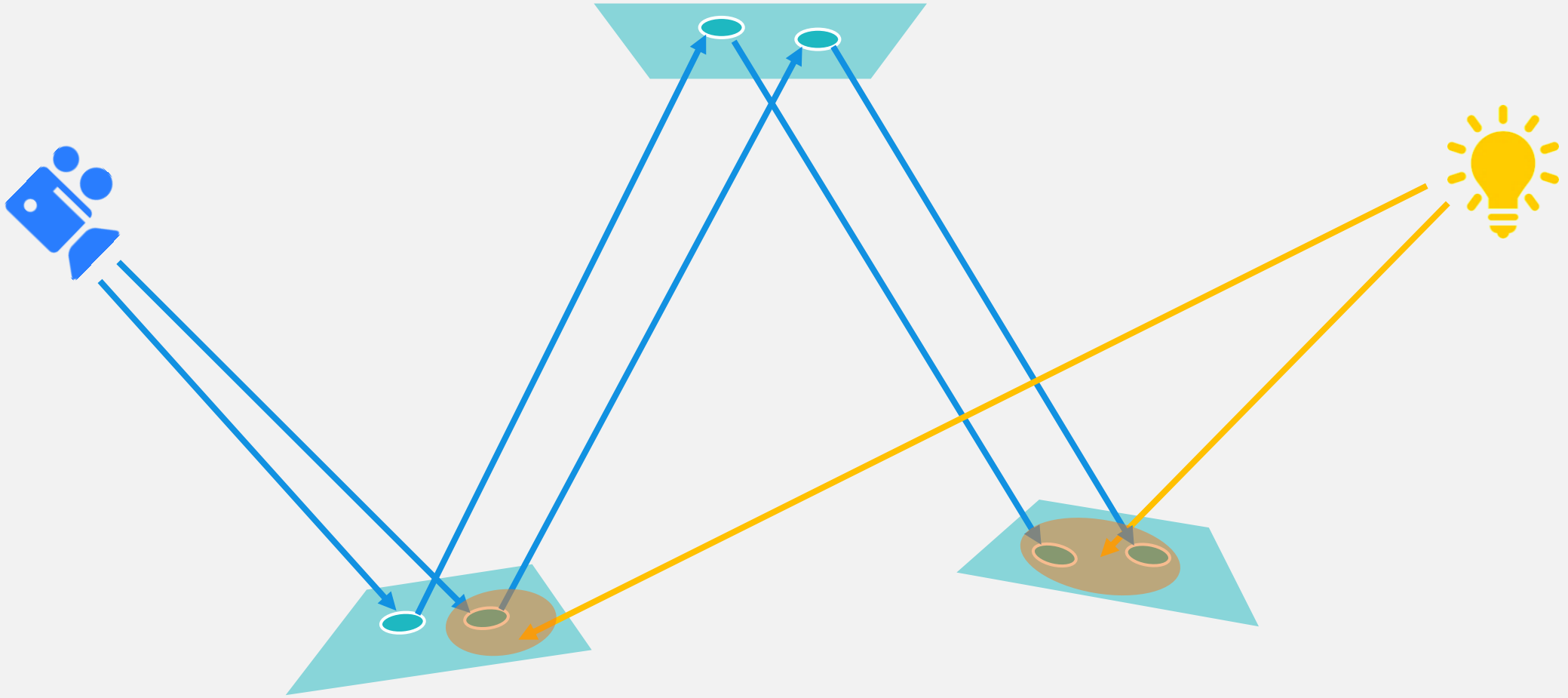
Photo by Raphaël Biscaldi

Progressive Photon Mapping

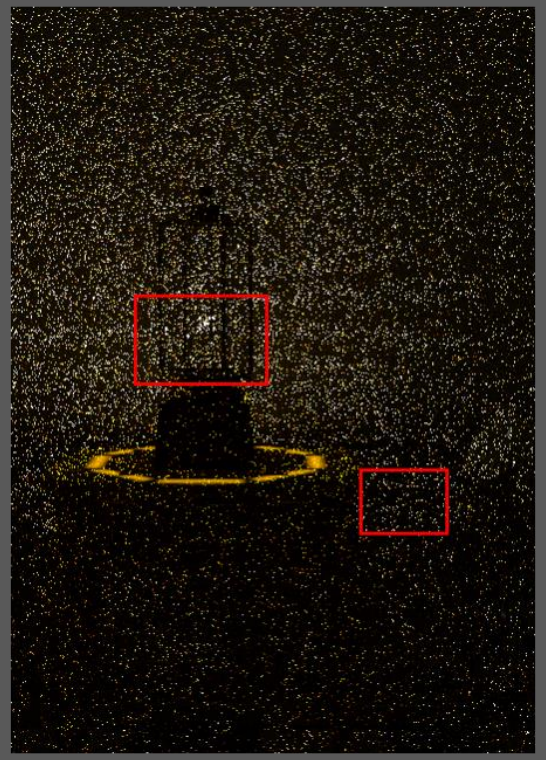
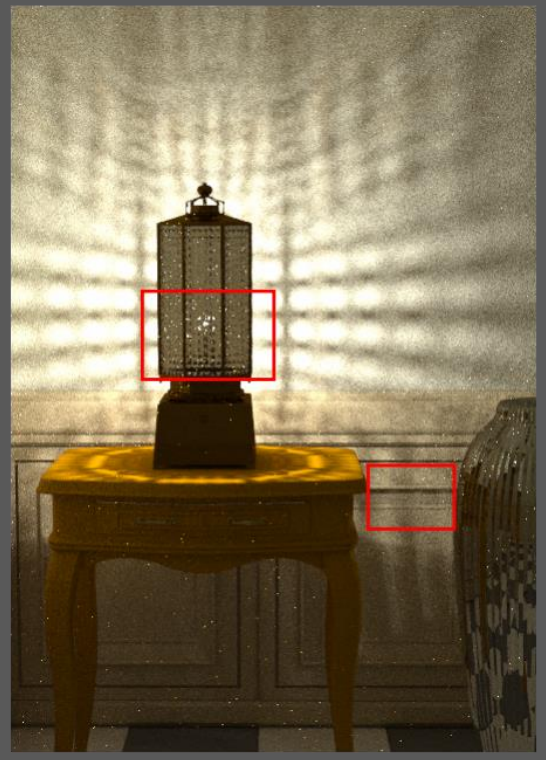
Progressive Photon Mapping

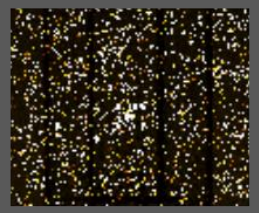
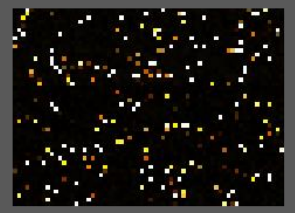
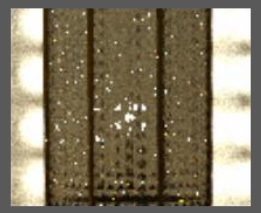
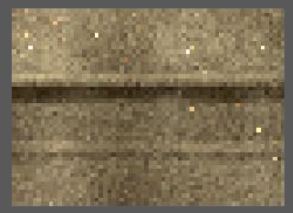
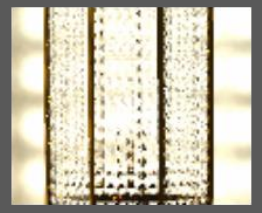


Progressive Photon Mapping



Comparison

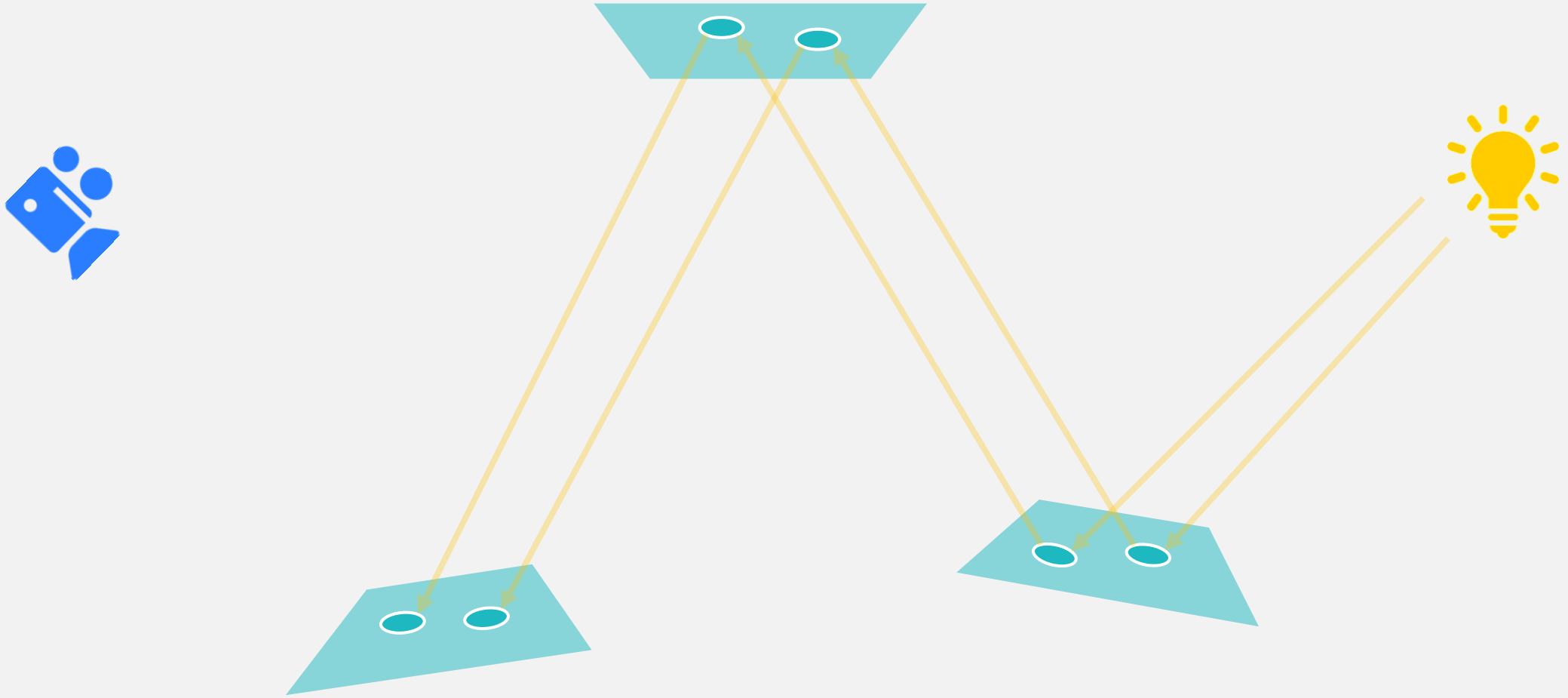

Path Tracing

Bidirectional Path Tracing

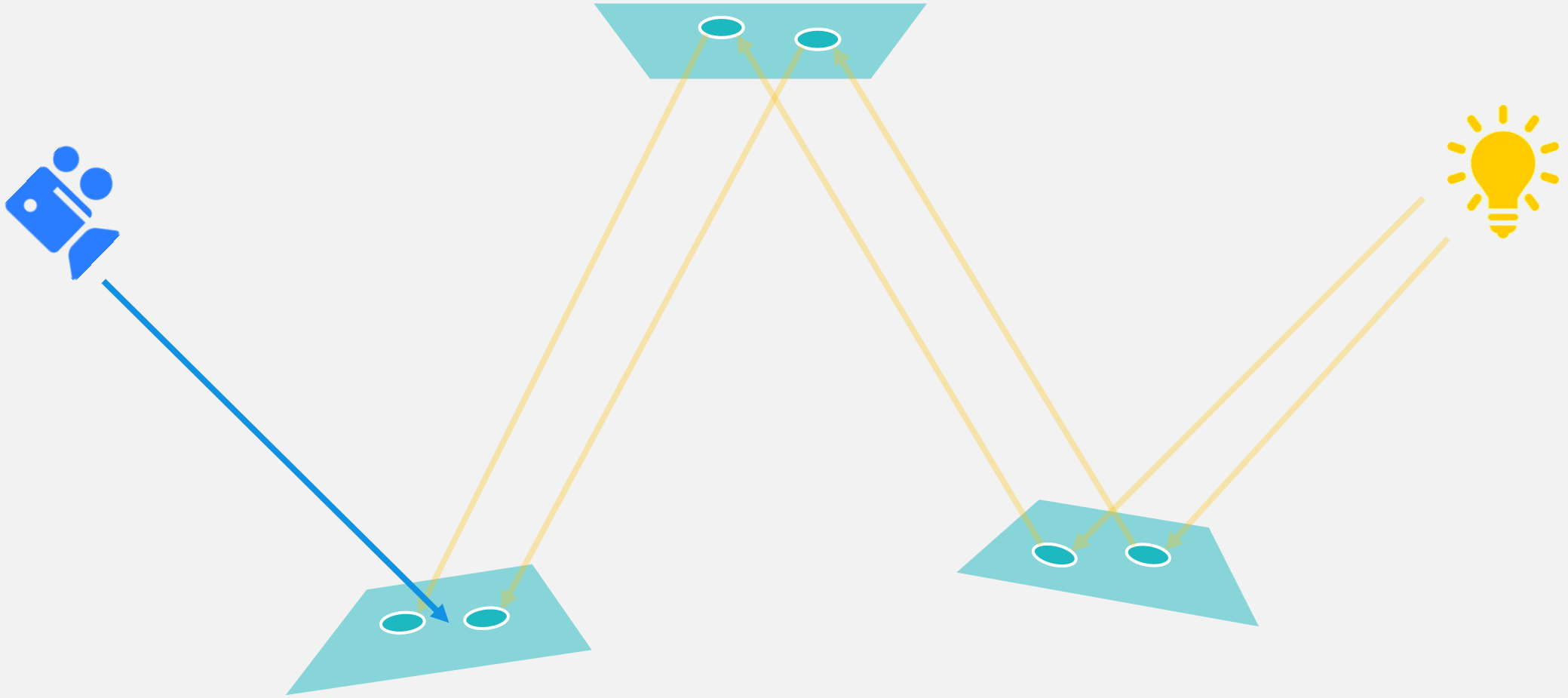
Photon Mapping

Progressive Photon Mapping

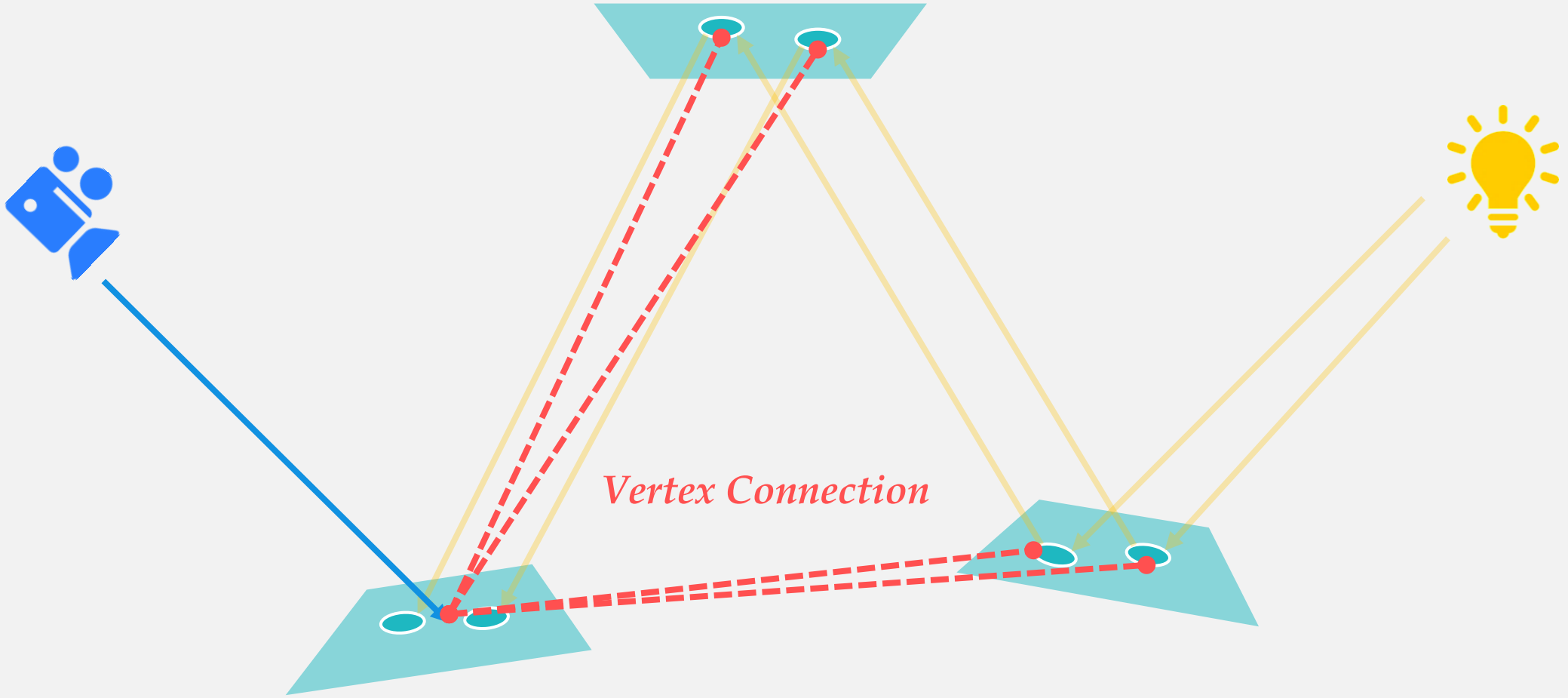
Vertex Connection & Merging (VCM=BT+PM)



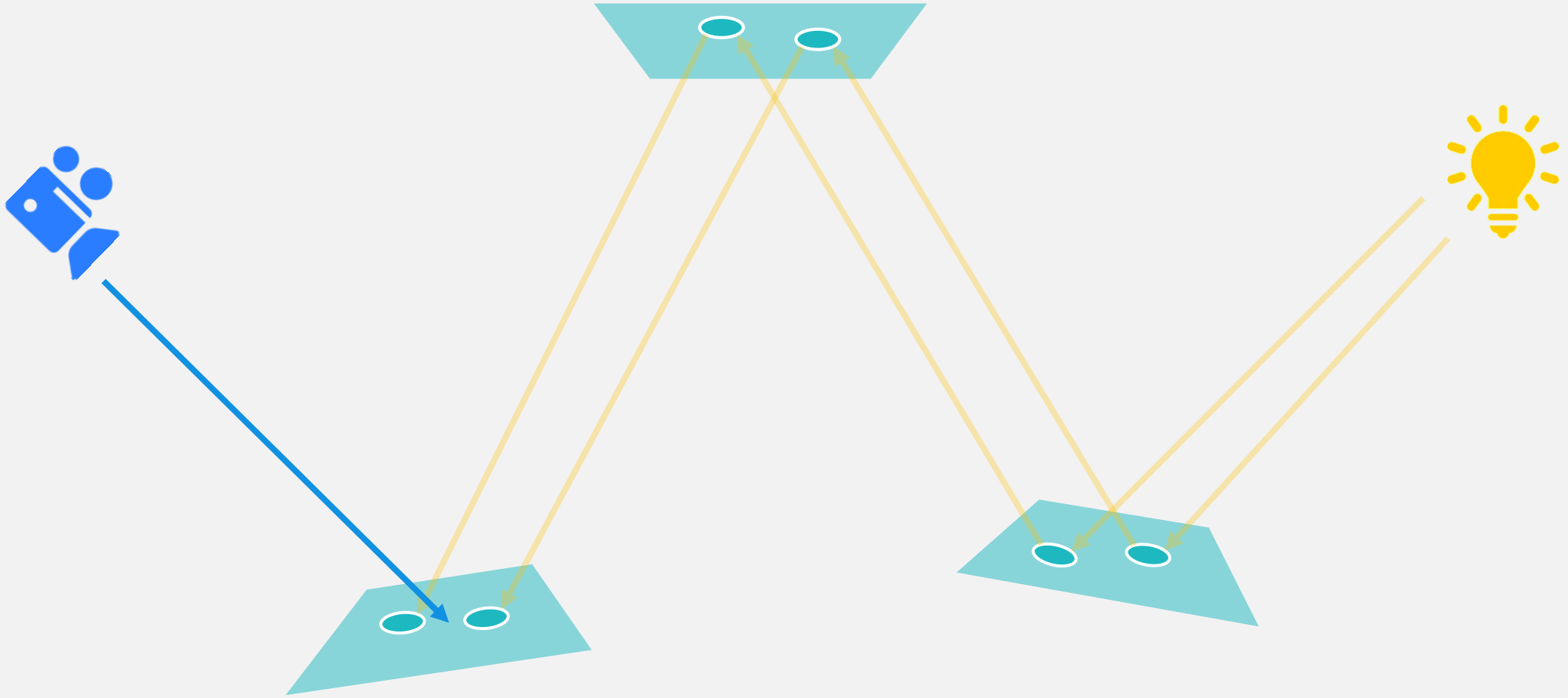
Vertex Connection & Merging (VCM=BT+PM)



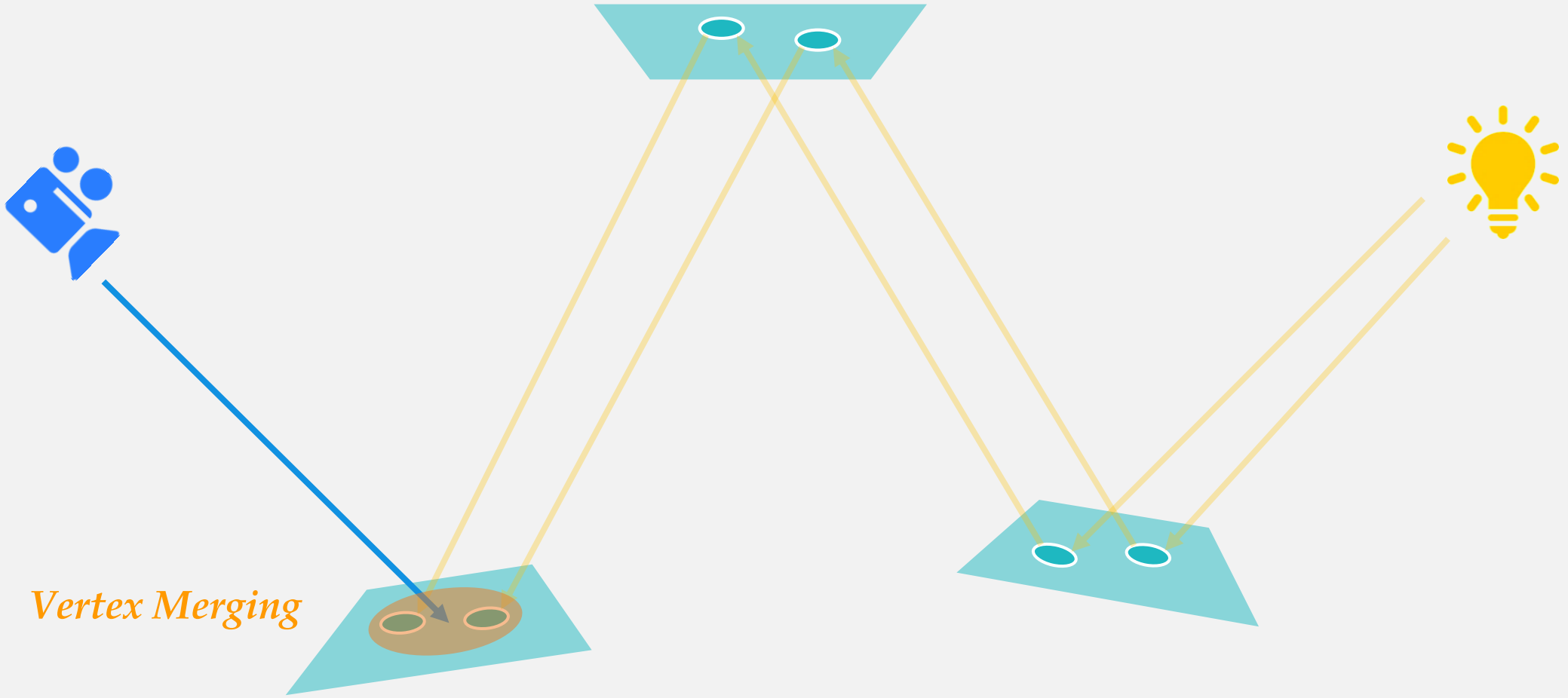
Vertex Connection & Merging (VCM=BT+PM)



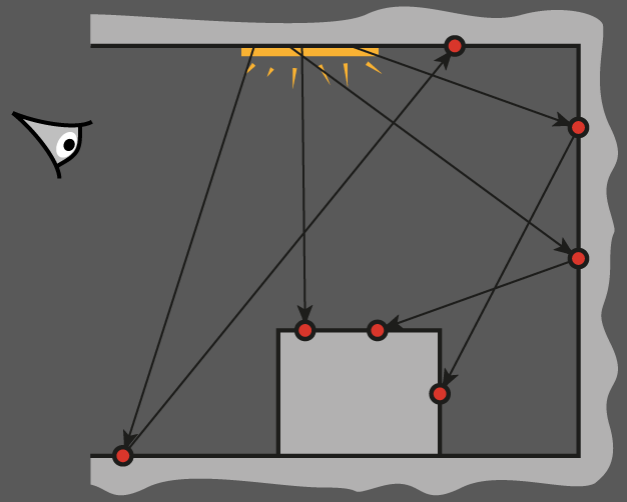
Vertex Connection & Merging (VCM=BT+PM)



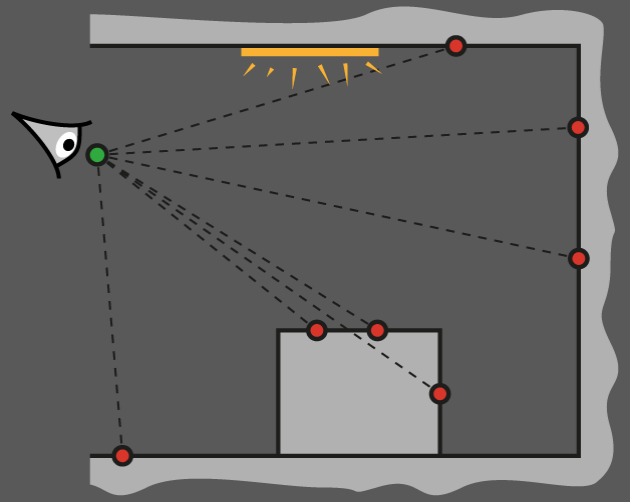
Vertex Connection & Merging (VCM=BT+PM)



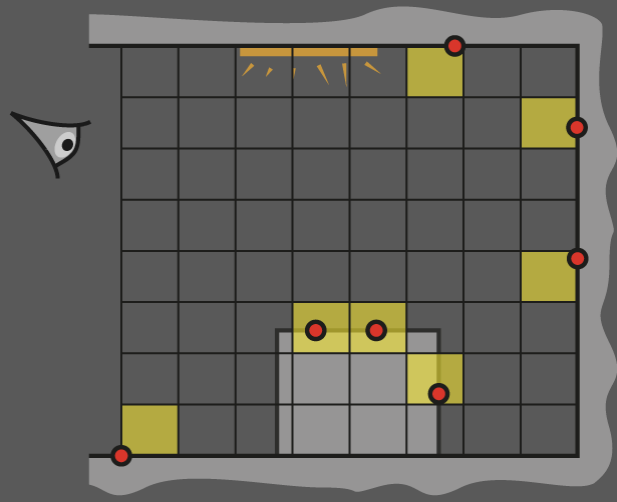
Stage 1: Light sub-path sampling



a) Trace sub-paths

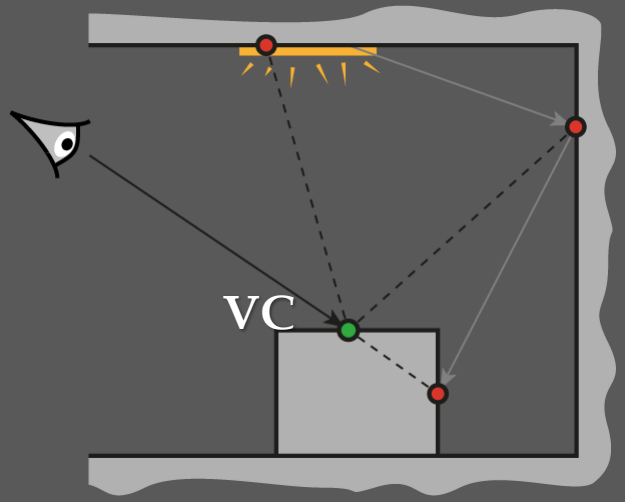


b) Connect to eye

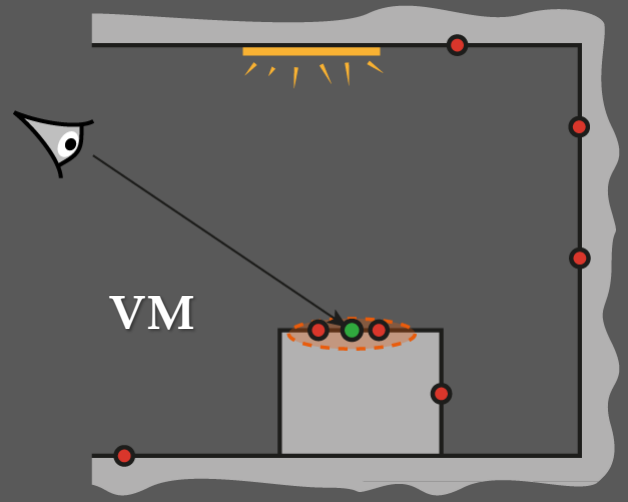


c) Build search structure

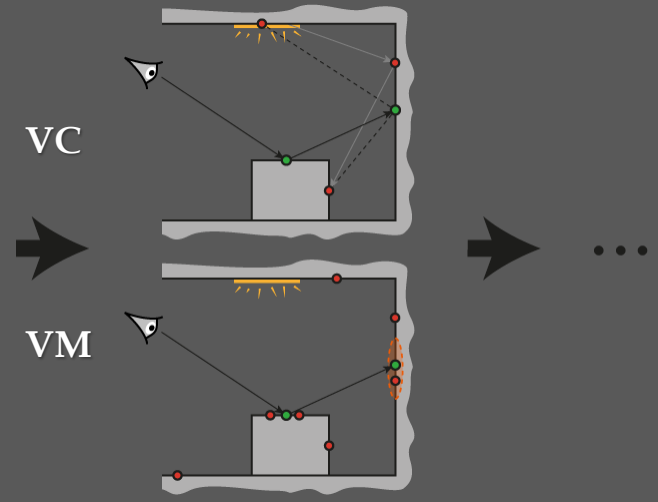
Stage 2: Eye sub-path sampling



a) Vertex connection



b) Vertex merging



c) Continue sub-path

Path Tracing

Bidirectional Path Tracing

Progressive Photon Mapping

VCM

Path Tracing

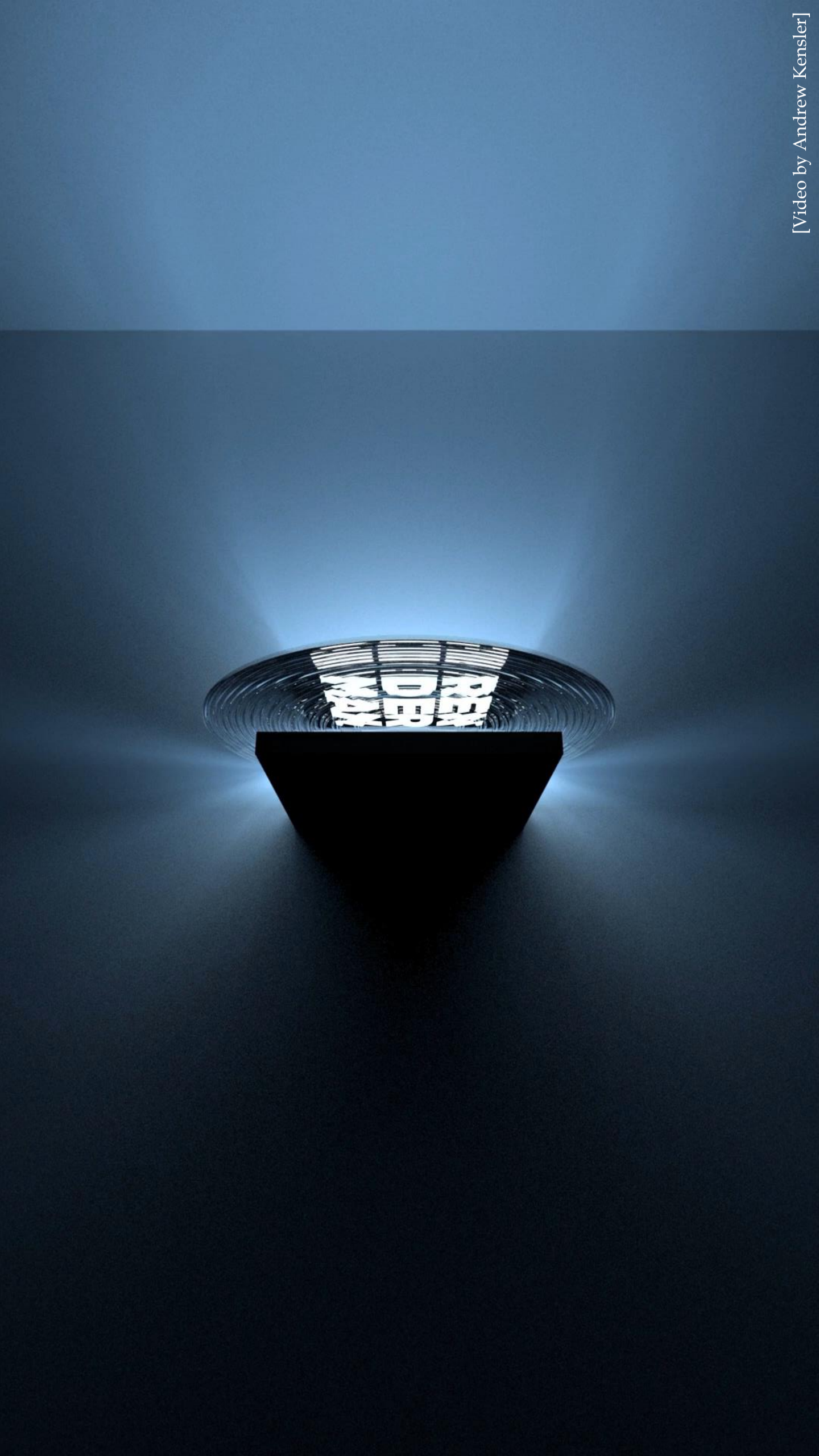
Bidirectional Path Tracing

Progressive Photon Mapping

VCM

Read More

1. *Global Illumination Across Industries*, SIG'10
2. *Recent Advances in Light Transport Simulation: Theory & Practice*, SIG'13

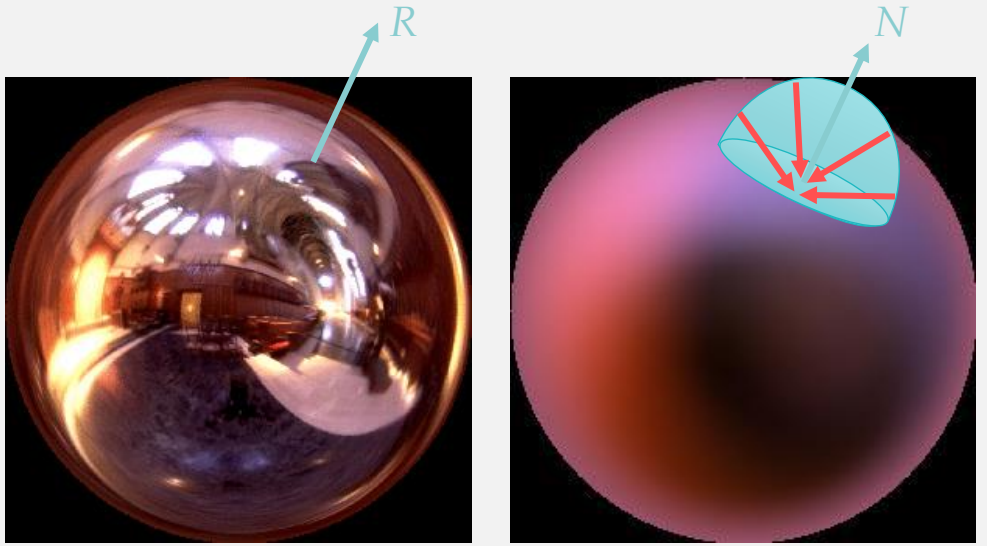

[Video by Andrew Kensler]

Irradiance Environment Mapping

- Given surface normal, return diffuse reflection (irradiance)
- Irradiance is usually blurred (low-frequency signals)
  - Since it's computed by applying wide filter to original environment map

$$E(n) = \sum_i L_i \cdot \max(l \cdot n, 0)$$

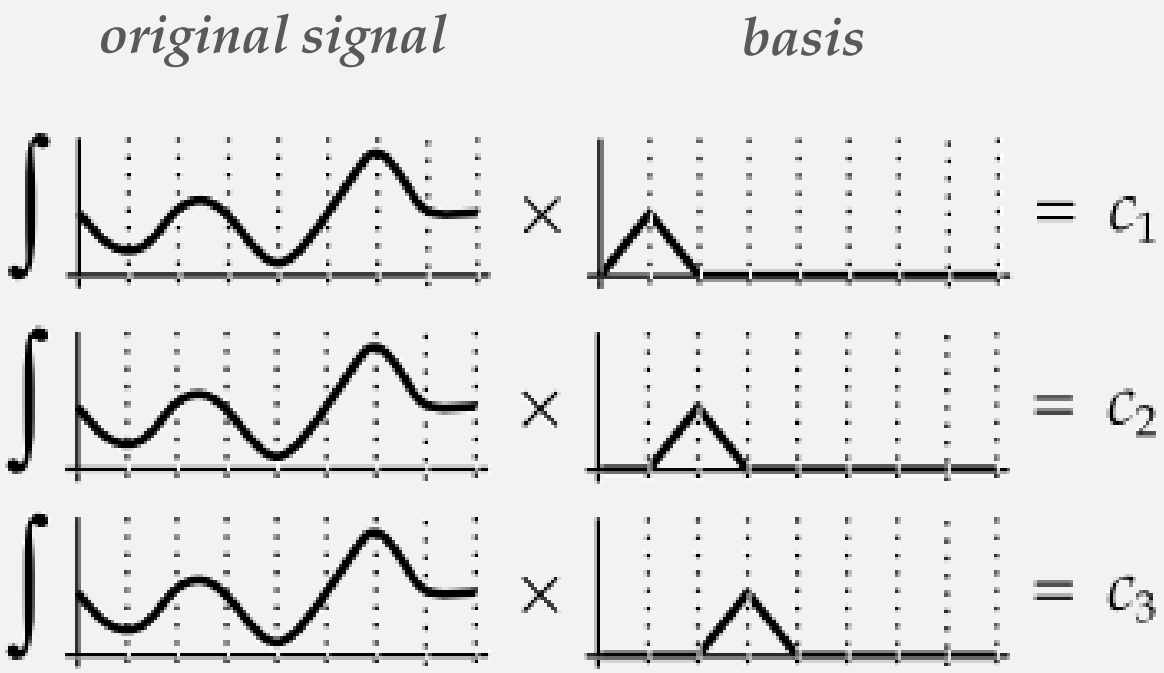
- Assumptions
  - Lambertian surface
  - Distance illumination
  - No shadowing



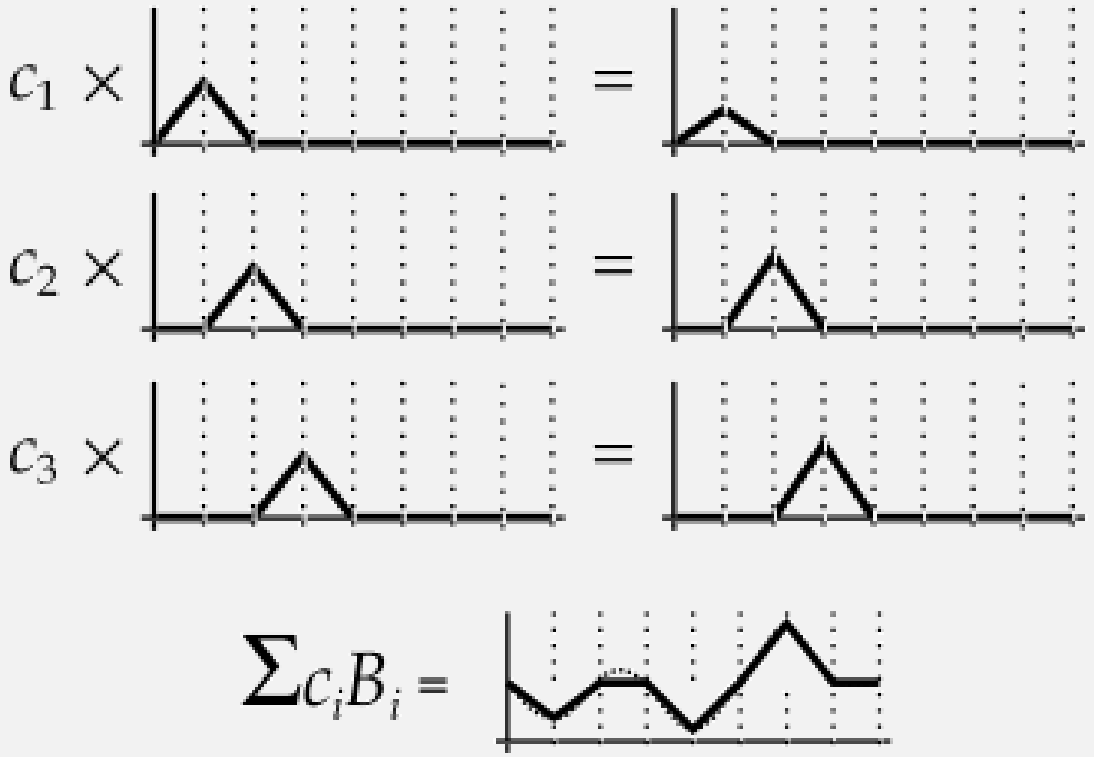
incident radiance irradiance env map

Decomposition & Reconstruction

Projection



Reconstruction



[Green, '03]

Spherical Harmonics (SHs)

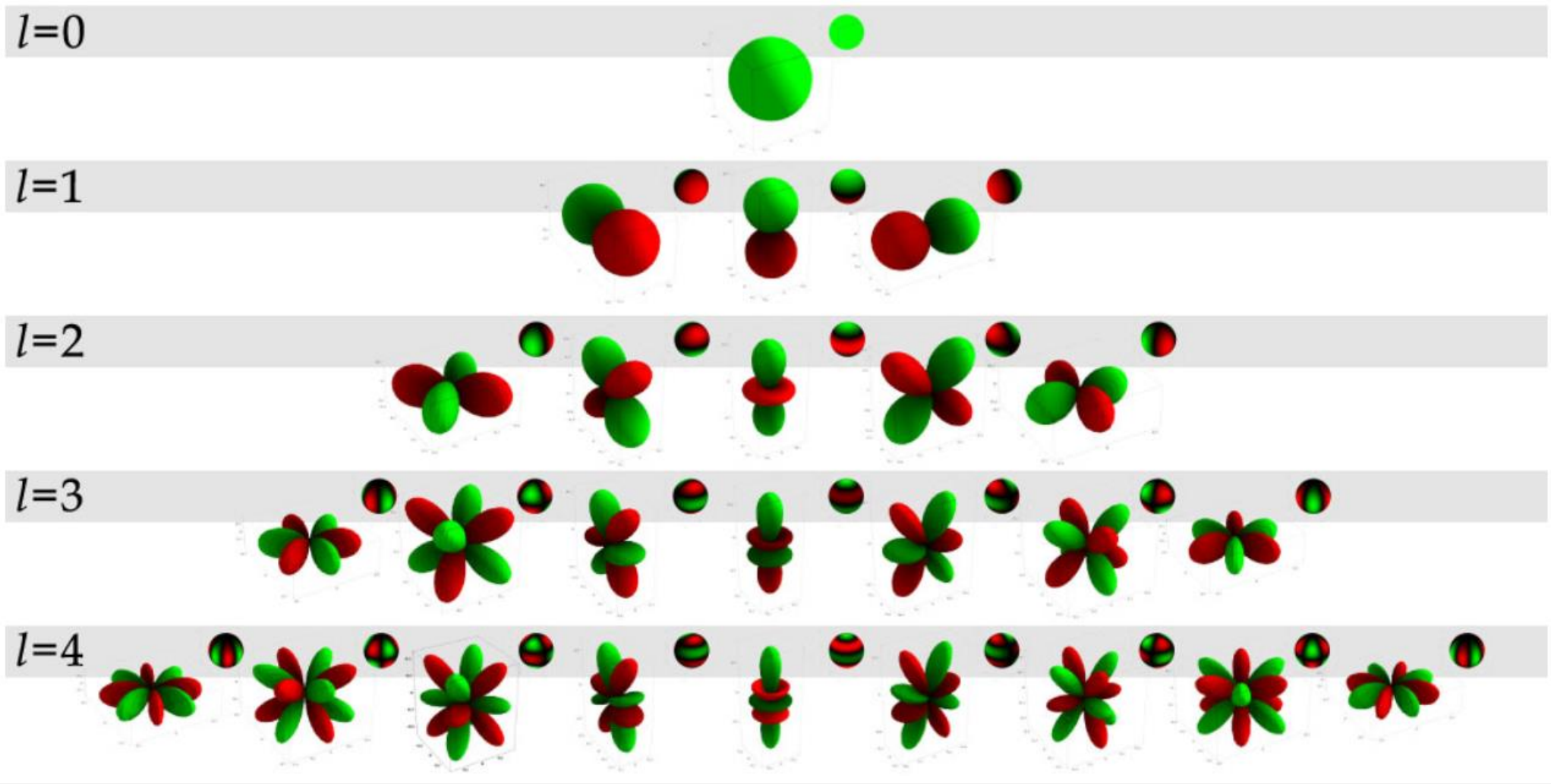
- The orthogonal basis functions over the sphere
- Represented in spherical coordinates by the recursive function

$$y_l^m(\theta,\phi) = \begin{cases} \sqrt{2}K_l^m \cos(m\phi)\, P_l^m(\cos\theta), m > 0 \\ \sqrt{2}K_l^m \sin(-m\phi) P_l^{-m}(\cos\theta), m < 0 \\ K_l^0 P_l^0(\cos\theta), m = 0 \end{cases}$$

  - Where l is the band and m is the index within the band
- Two great properties:
  - Rotationally invariant: $\tilde{g}(s) = \tilde{f}(R(s)) = R(\tilde{f}(s))$ where $g(s) = R(f(s))$
  - The integral of the function's products is the same as the dot product of their coefficients

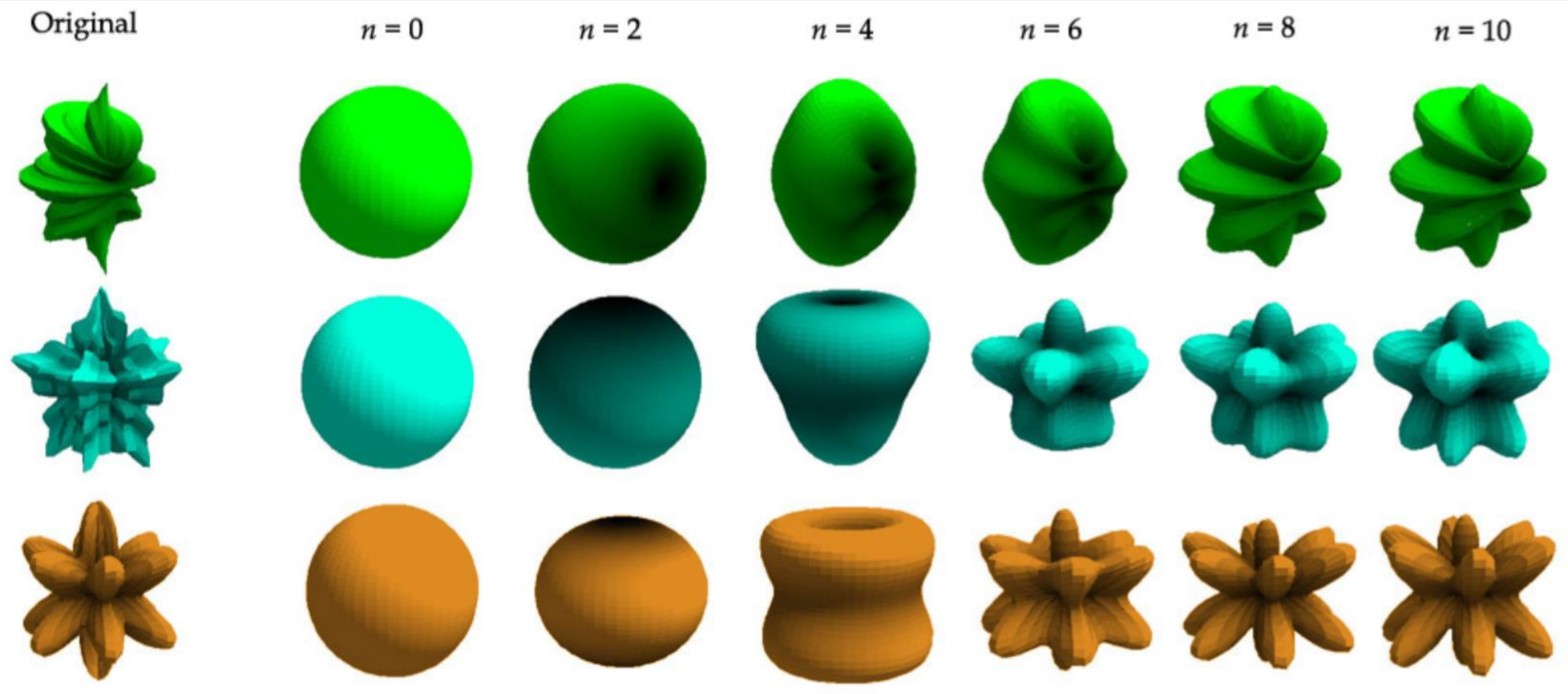
$$\int_S \tilde{L}(s)\tilde{t}(s)ds = \sum_{i=0}^{n^2} L_i t_i$$

Spherical Harmonics (Cont.)



[Green, '03]

Spherical Harmonics Representations



[Green, '03]

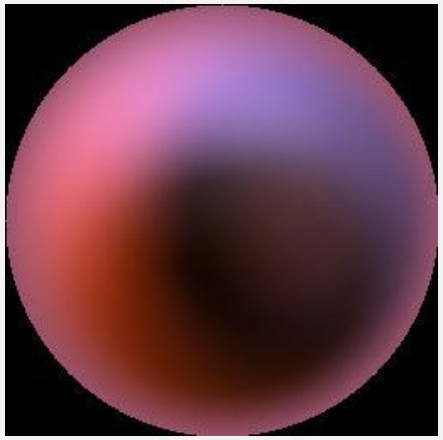
Spherical Harmonics Representations

Read More

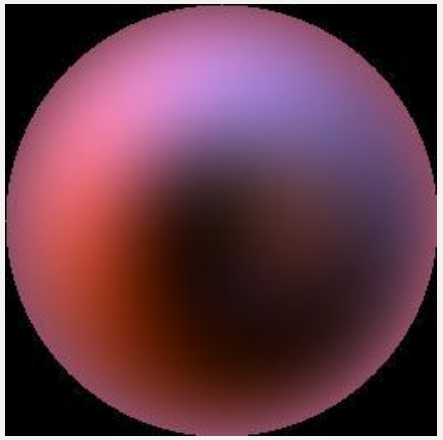
1. *Spherical Harmonics Lighting: The Gritty Details*, Robin Green.
2. *Stupid Spherical Harmonics (SH) Tricks*, Peter-Pike Sloan. 2010

[Green, '03]

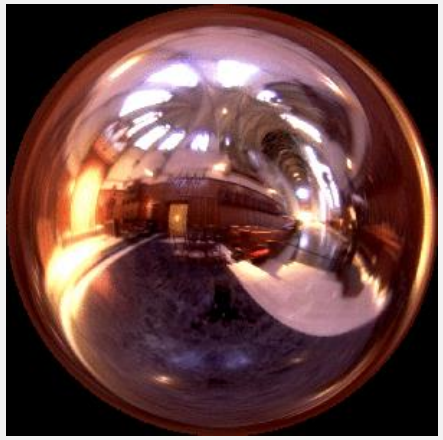
Irradiance Environment Map with SHs



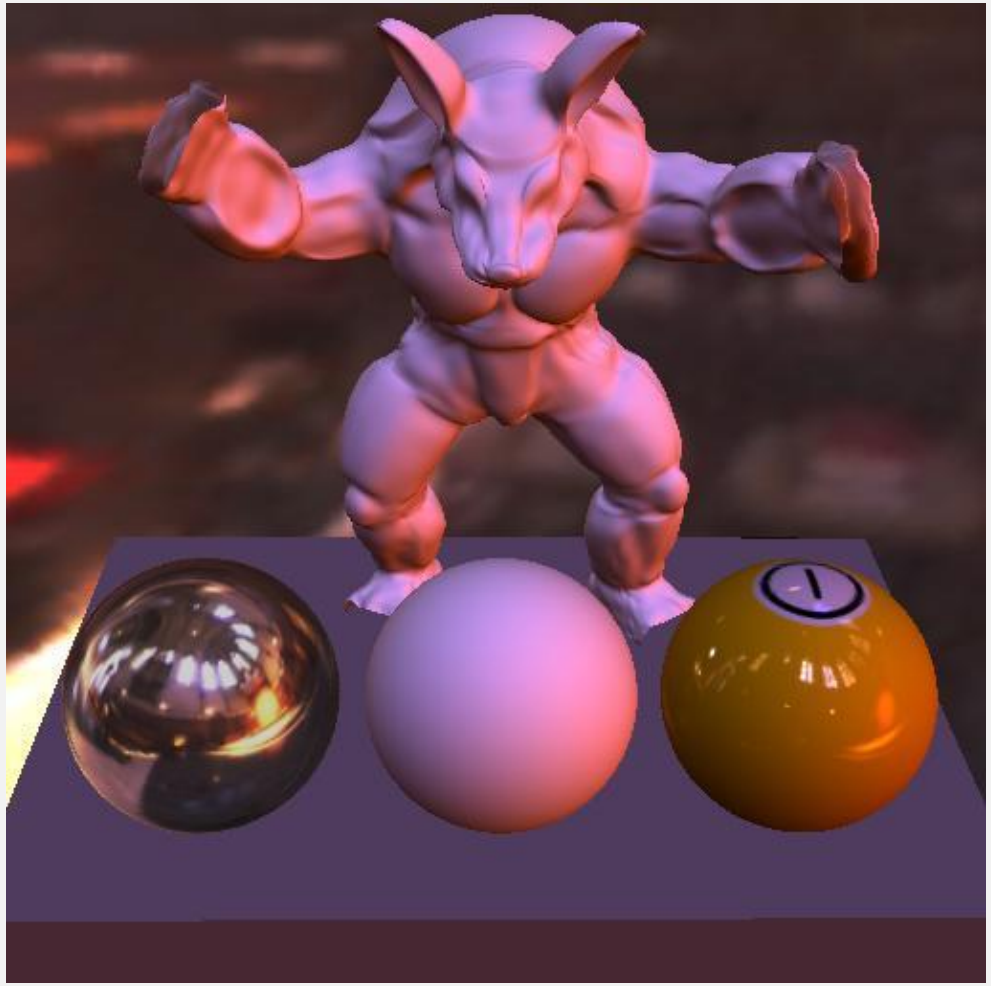
accumulation



SH, 9 coeffs

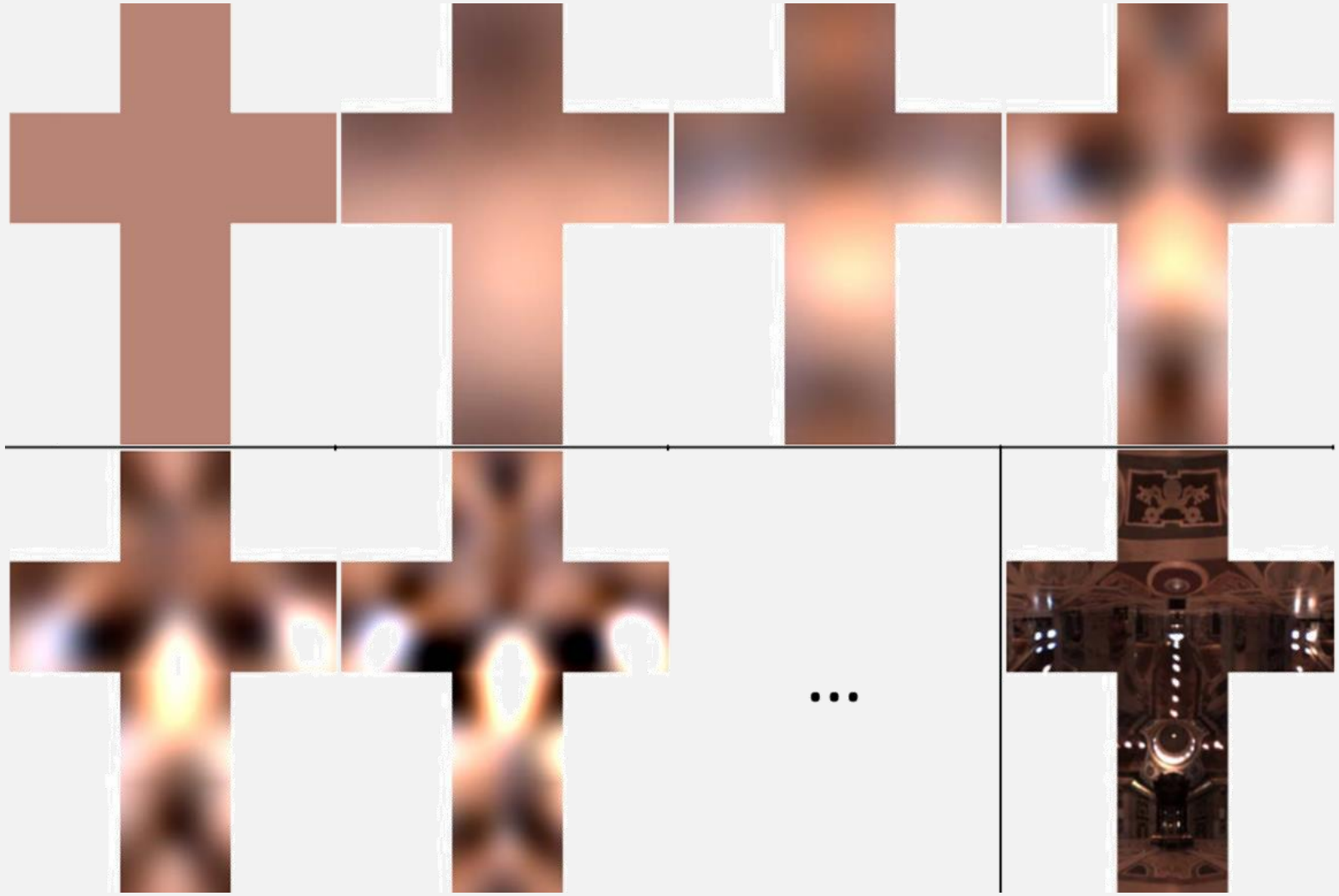


< 3% RMS error in average with 9 SH coefficients [Basri Jacobs 01]

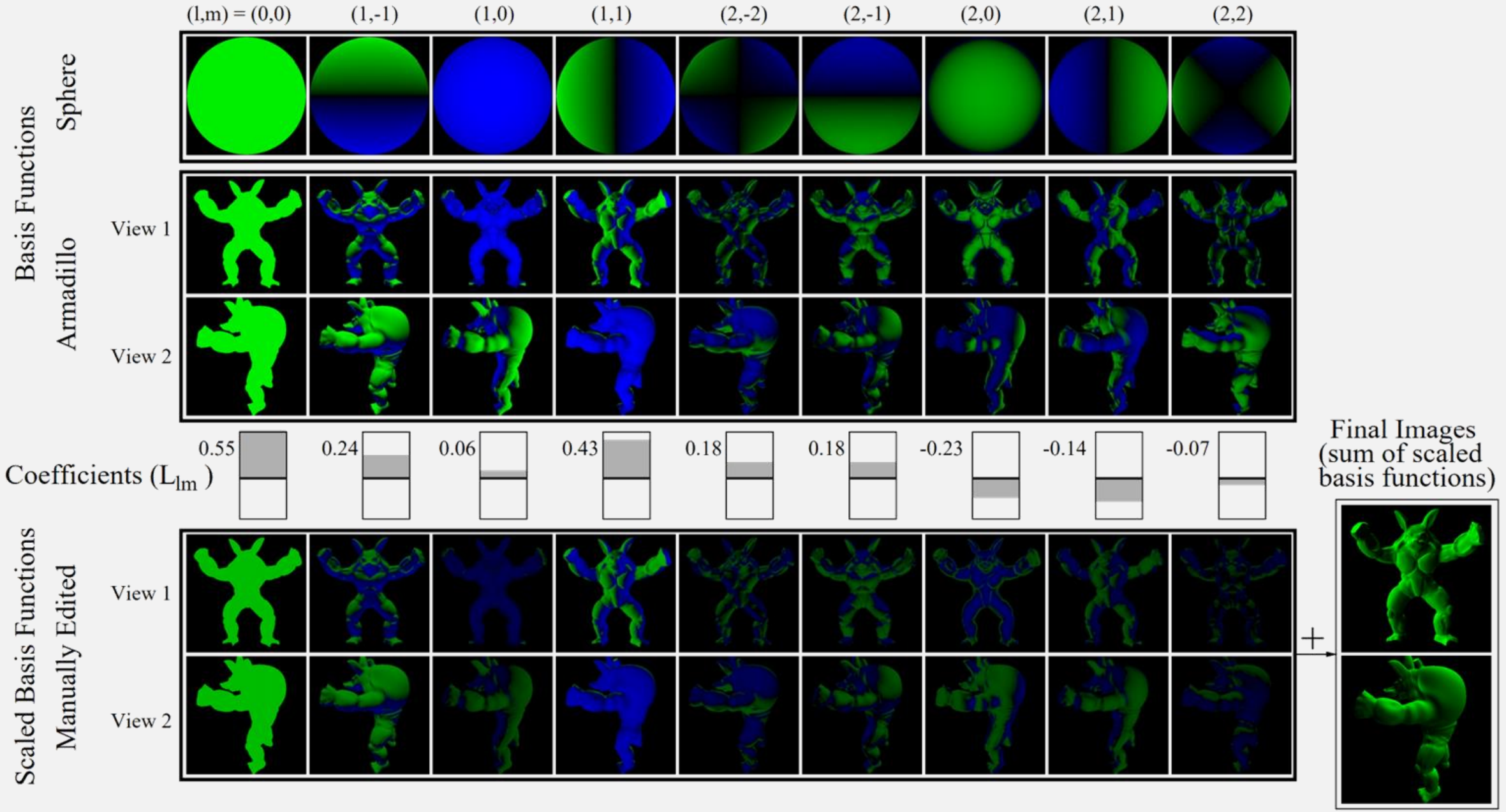


[Ramamoorthi and Hanrahan '01]

Environment Map Reconstruction with SHs



[Sloan, '08]

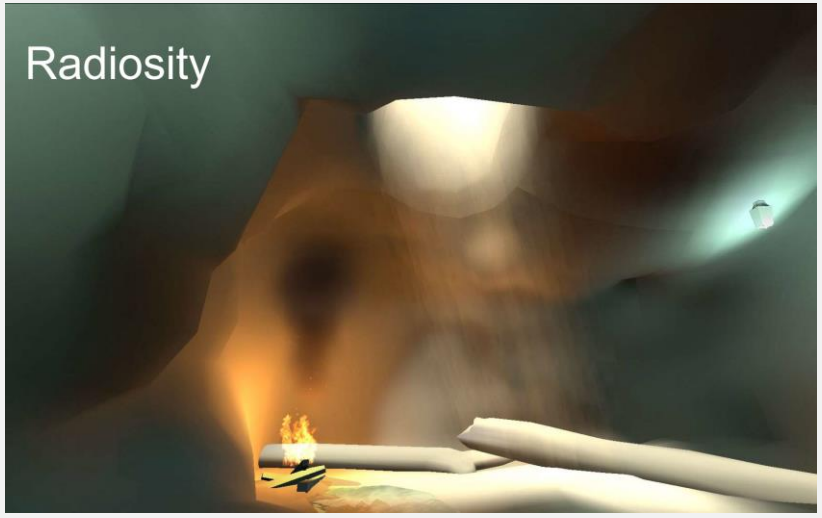


[Ramamoorthi and Hanrahan, '01]

Lightmap

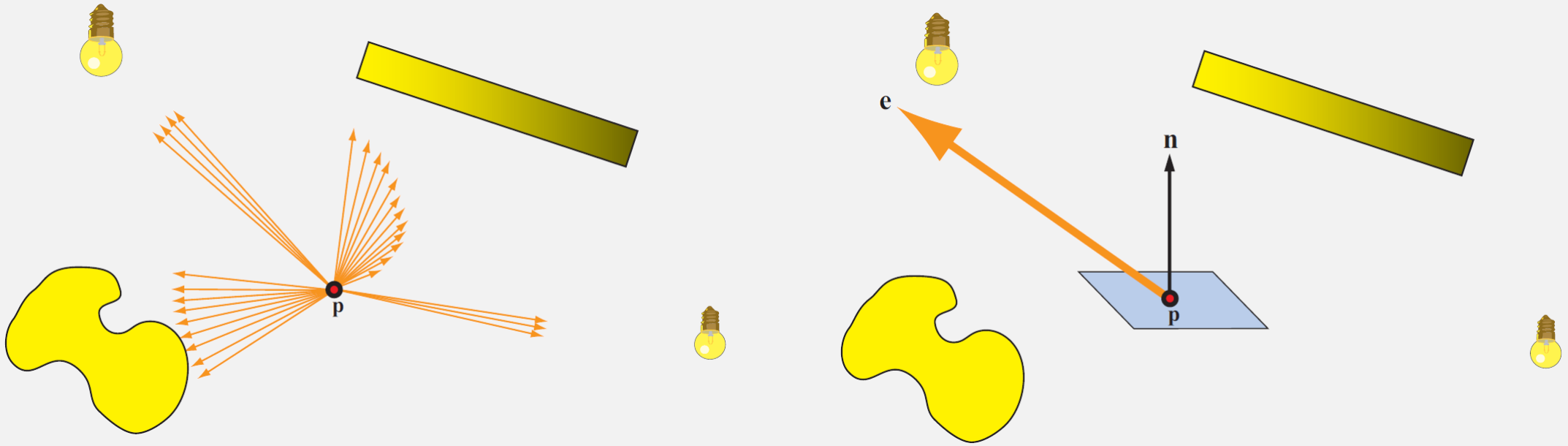
- For Lambertian surface, its reflectance $B = \rho E$ is
  - proportional to the incoming irradiance
  - equal in all outgoing directions
- Lighting computation involves accumulating radiance, and this accumulation is linear
- Thus we could precompute irradiance for each object
  - store in texture or vertex color
- The fetched irradiance is usually multiplied with diffuse/albedo maps in separate textures

Lightmap (Cont.)



http://www.decew.net/OSS/References/D3DTutorial10_Half-Life2_Shading.pdf

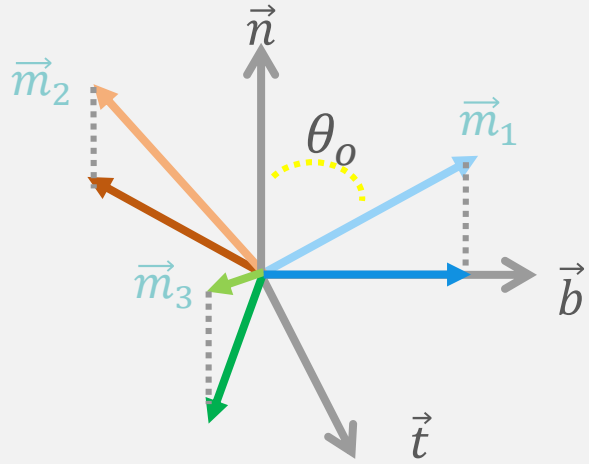
Vector Irradiance



$$\bar{E}(x,\vec{n}) = E(x,-\vec{n}) - E(x,\vec{n}) = e \cdot \vec{n}$$

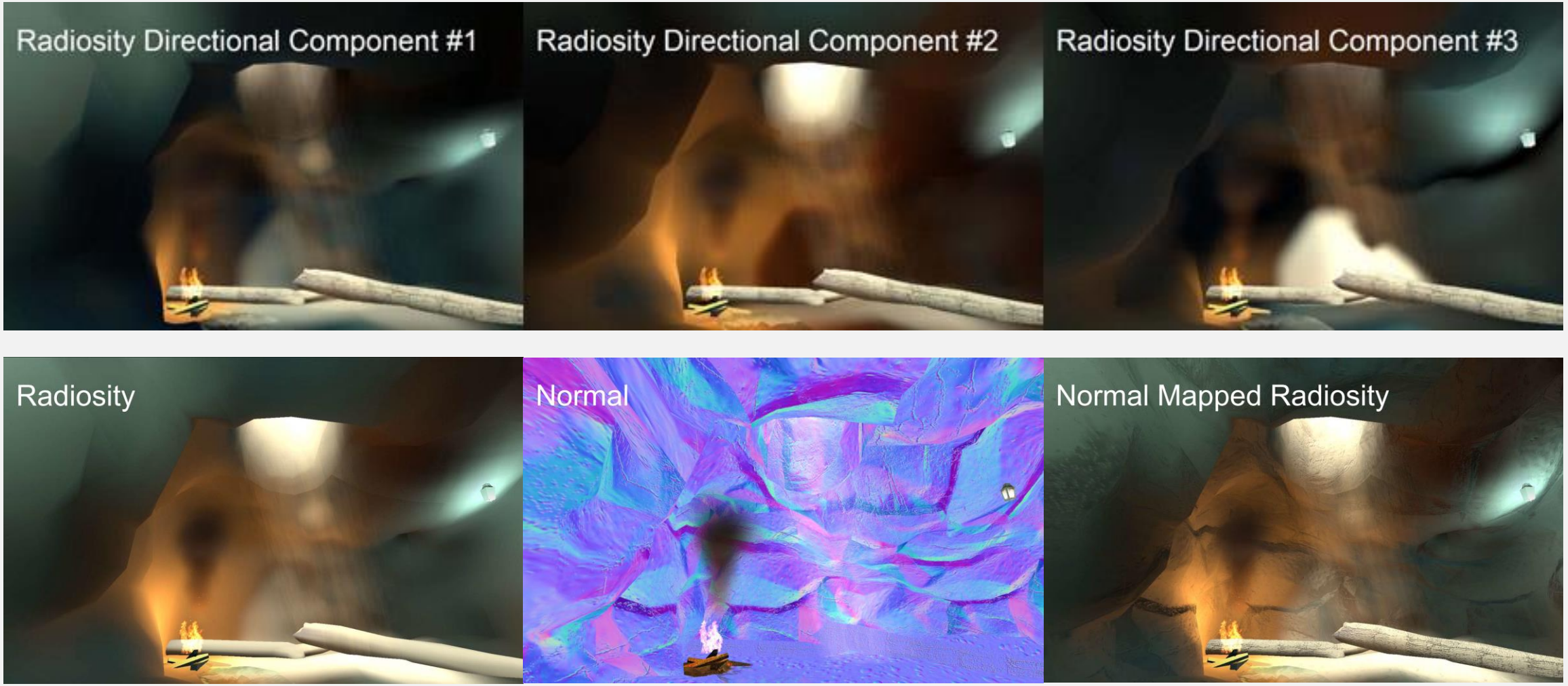
Radiosity Normal Mapping [Half-Life 2, 2004]

- How to work with high-frequency normal maps
- Encode how irradiance changes with the normal
- Precompute vector irradiance along three basis vectors in tangent space
- At each shading point
  - Transform shading normal to local frame
  - Sample three irradiance colors and blending those values with projected coefficients of base vector



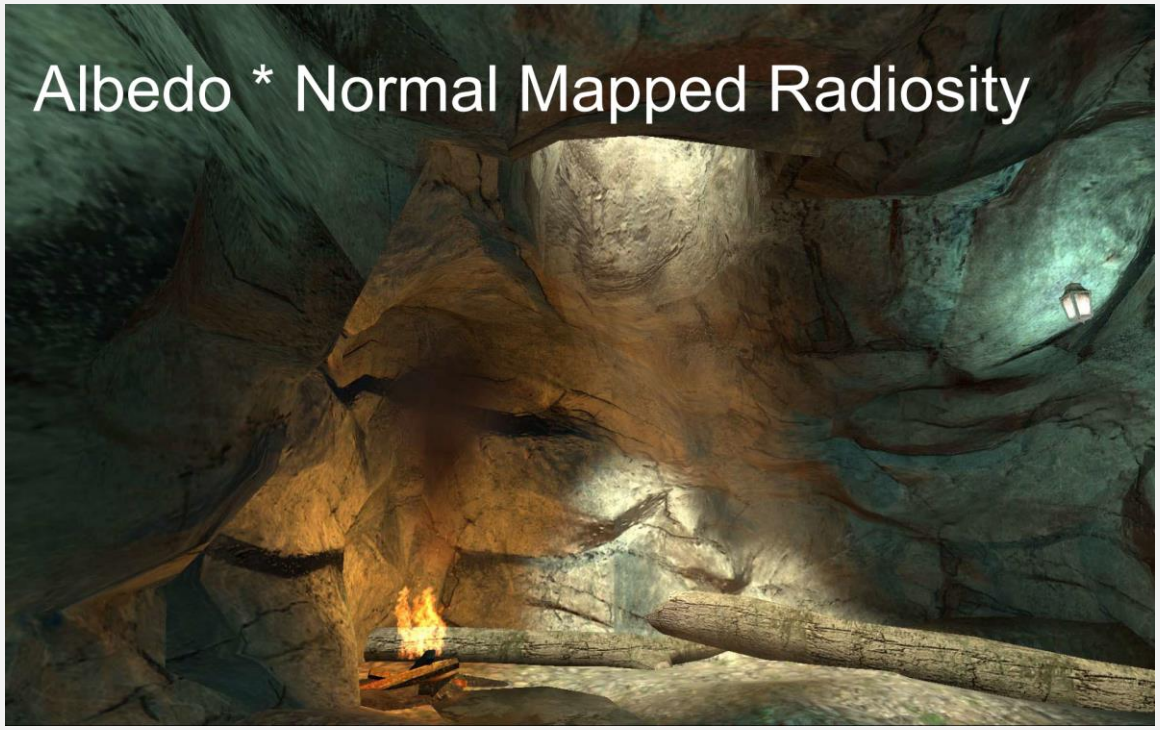

```
lightmapColor[0] * dot(m1, normal) +
lightmapColor[1] * dot(m2, normal) +
lightmapColor[2] * dot(m3, normal)
```

Radiosity Normal Mapping (Cont.)



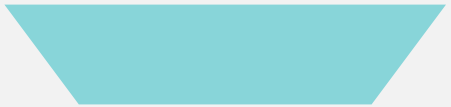
http://www.decew.net/OSS/References/D3DTutorial10_Half-Life2_Shading.pdf

Radiosity Normal Mapping (Cont.)

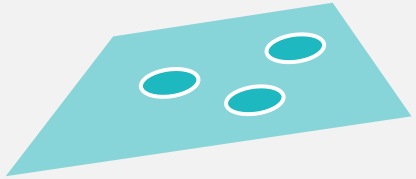


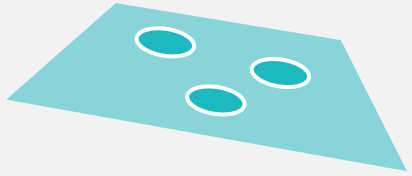
http://www.decew.net/OSS/References/D3DTutorial10_Half-Life2_Shading.pdf

Irradiance Caching

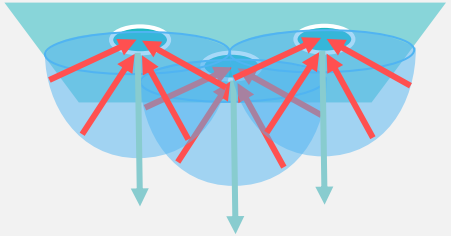


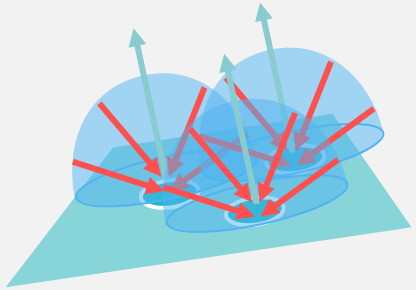
Irradiance Caching

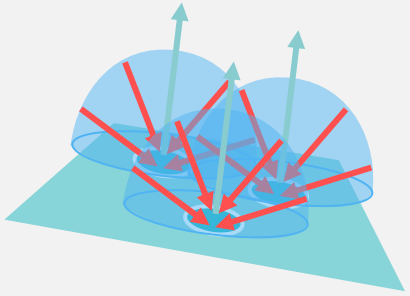




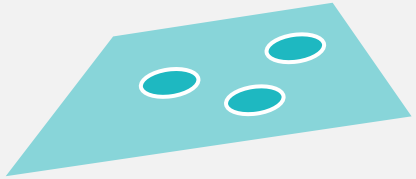
Irradiance Caching

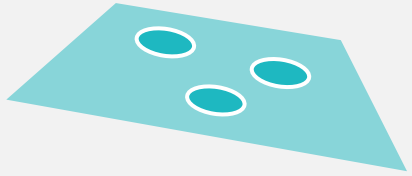




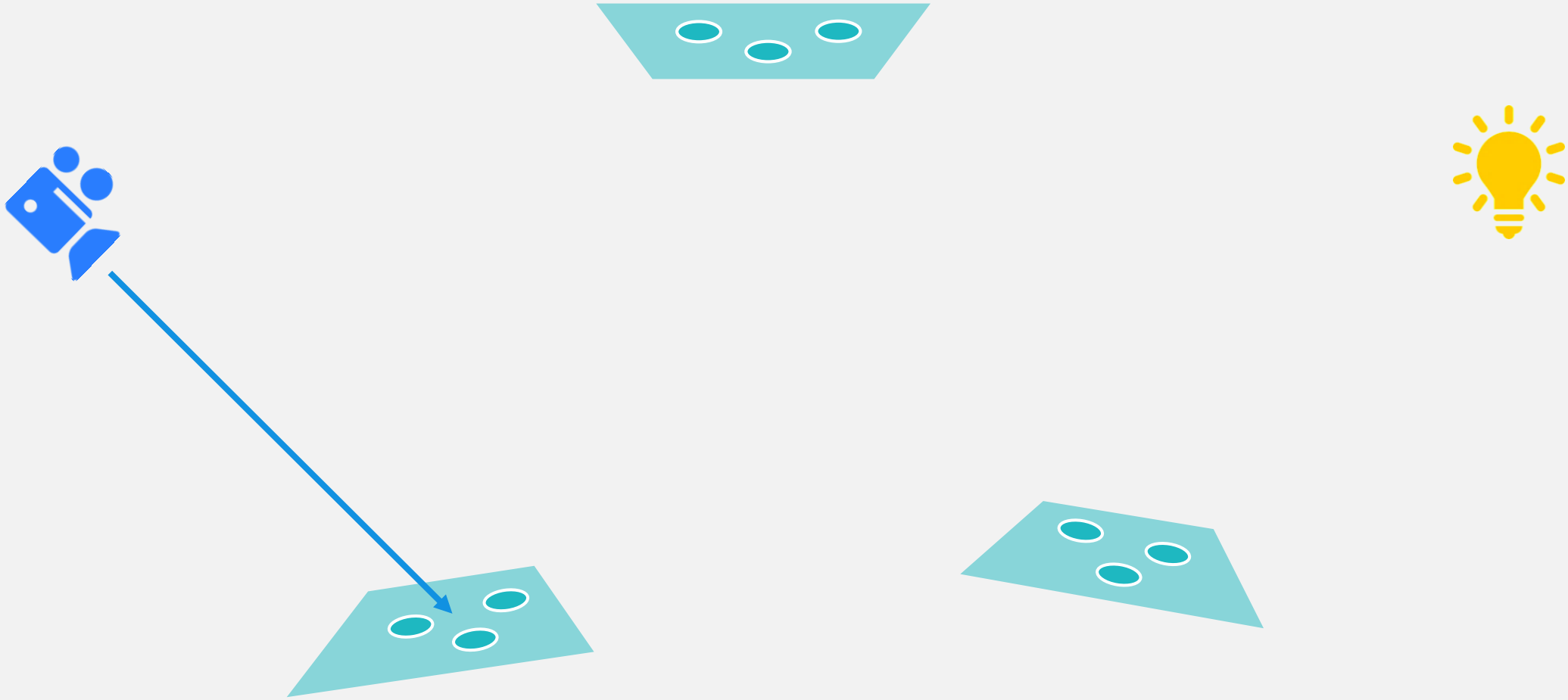


Irradiance Caching

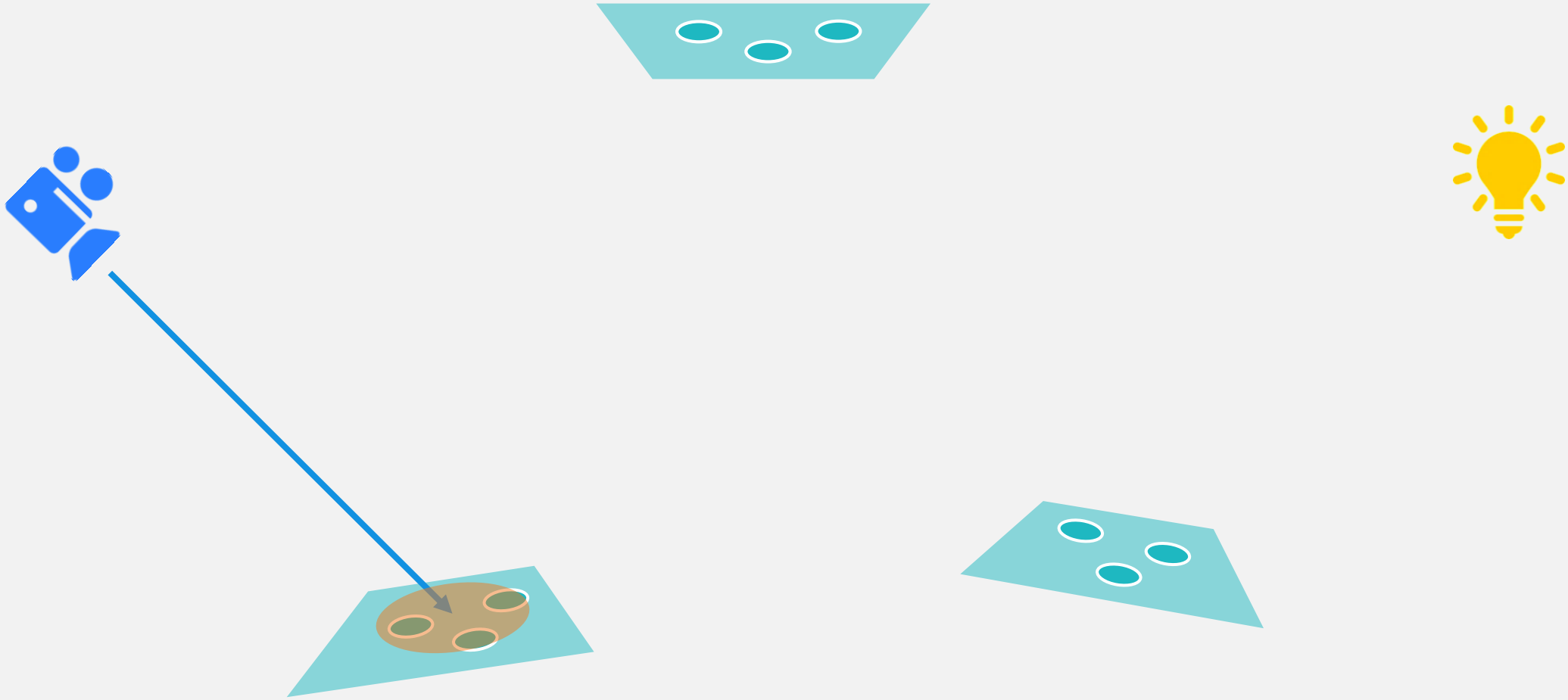




Irradiance Caching



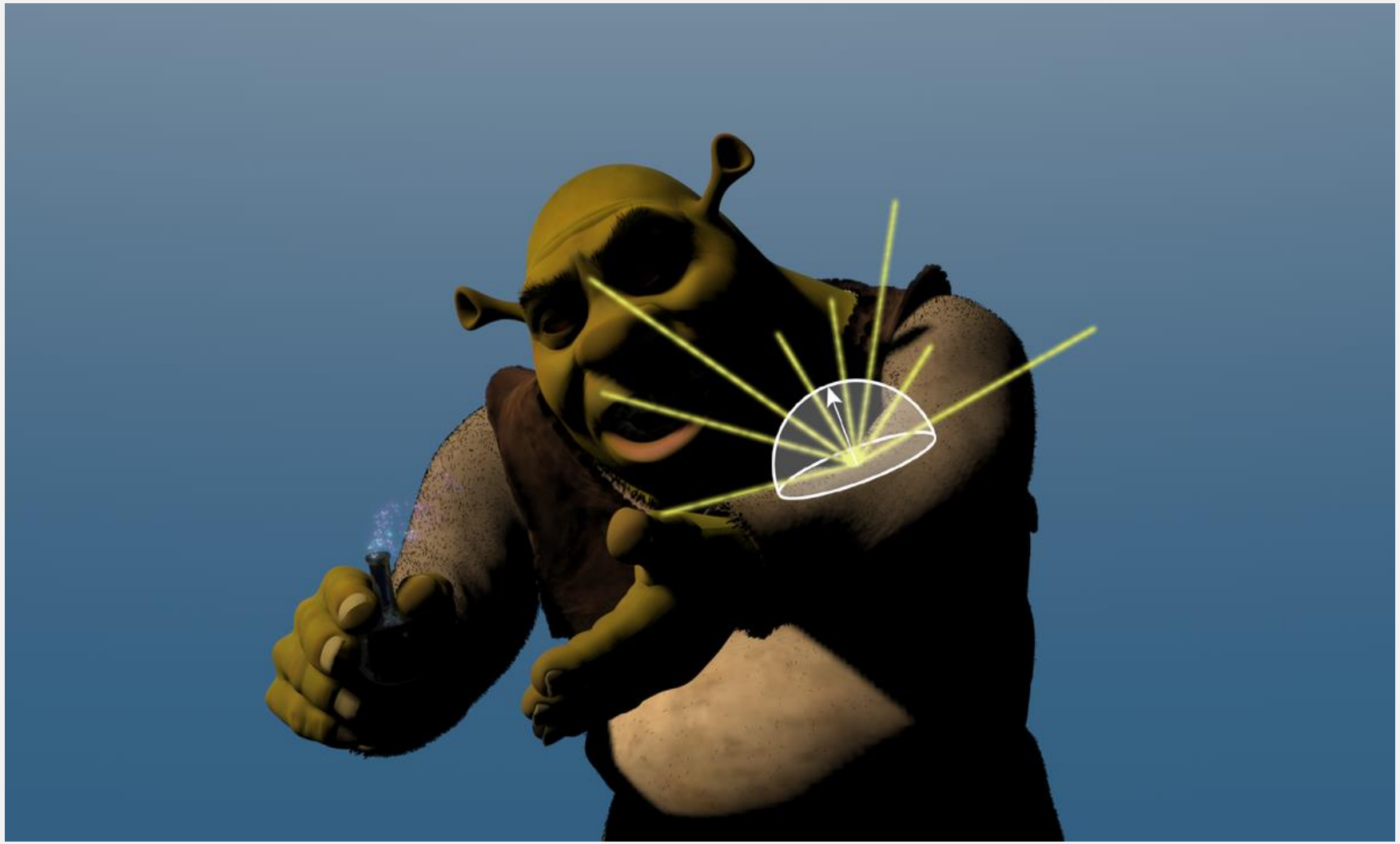
Irradiance Caching



Irradiance Caching – Spread Final Gather Points

[Tabellion '10]

Irradiance Caching – Compute Irradiance



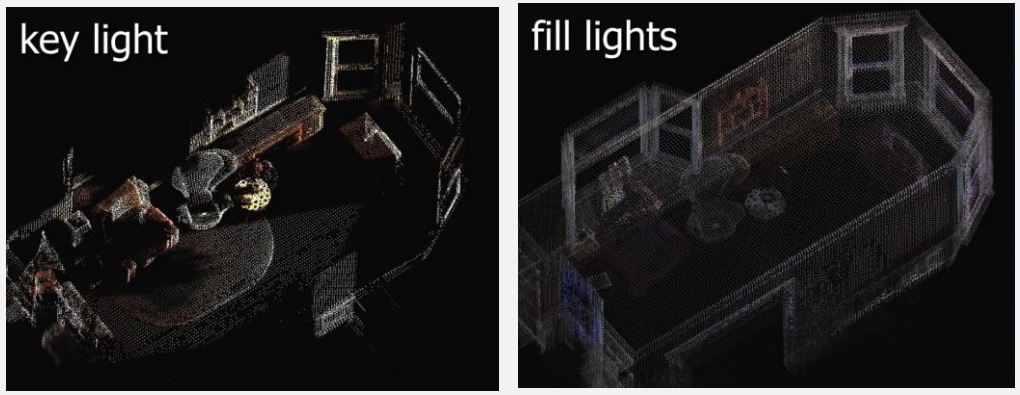
[Tabellion '10]

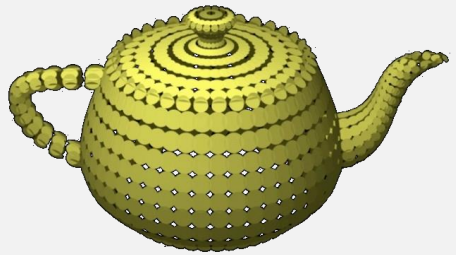
Irradiance Caching – Shading with Nearby Samples

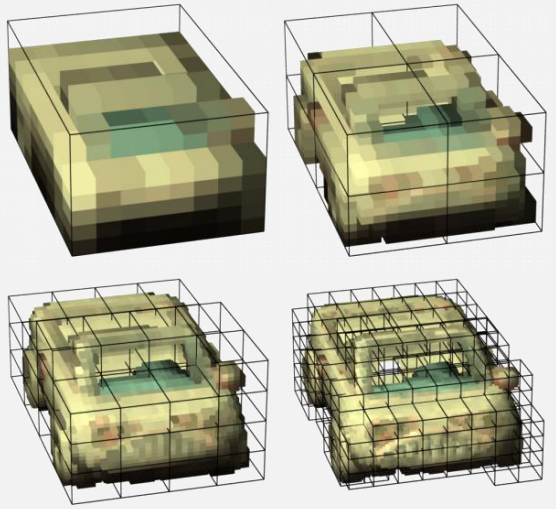
[Tabellion '10]

Point-Based Global Illumination

1. Generate point cloud of directly illuminated surface colors
   - Store position, normal and color for each point
2. Organize points into octree (brick map); larger points and spherical harmonics
3. Compute indirect diffuse/glossy illumination at each shading point







[Christensen and Batali '04]


Direct Illumination
https://graphics.pixar.com/library/PointBasedColorBleeding/SlidesFromAnnecy09.pdf


Direct & Indirect Illumination
https://graphics.pixar.com/library/PointBasedColorBleeding/SlidesFromAnnecy09.pdf

PBGI Foliage AO Example

[Tabellion '10]

Analyze the light paths in a real scene…