CS 87/187, Spring 2016

Rendering Algorithms

Participating Media II

Prof. Wojciech Jarosz

wojciech.k.jarosz@dartmouth.edu

(with slide improvements by Jan Novák)

Dartmouth

Today's Menu

- Fast(er) multiple scattering
  - volumetric photon mapping
  - photon beams
  - virtual point lights
  - virtual ray/beam lights

Lots of algorithms

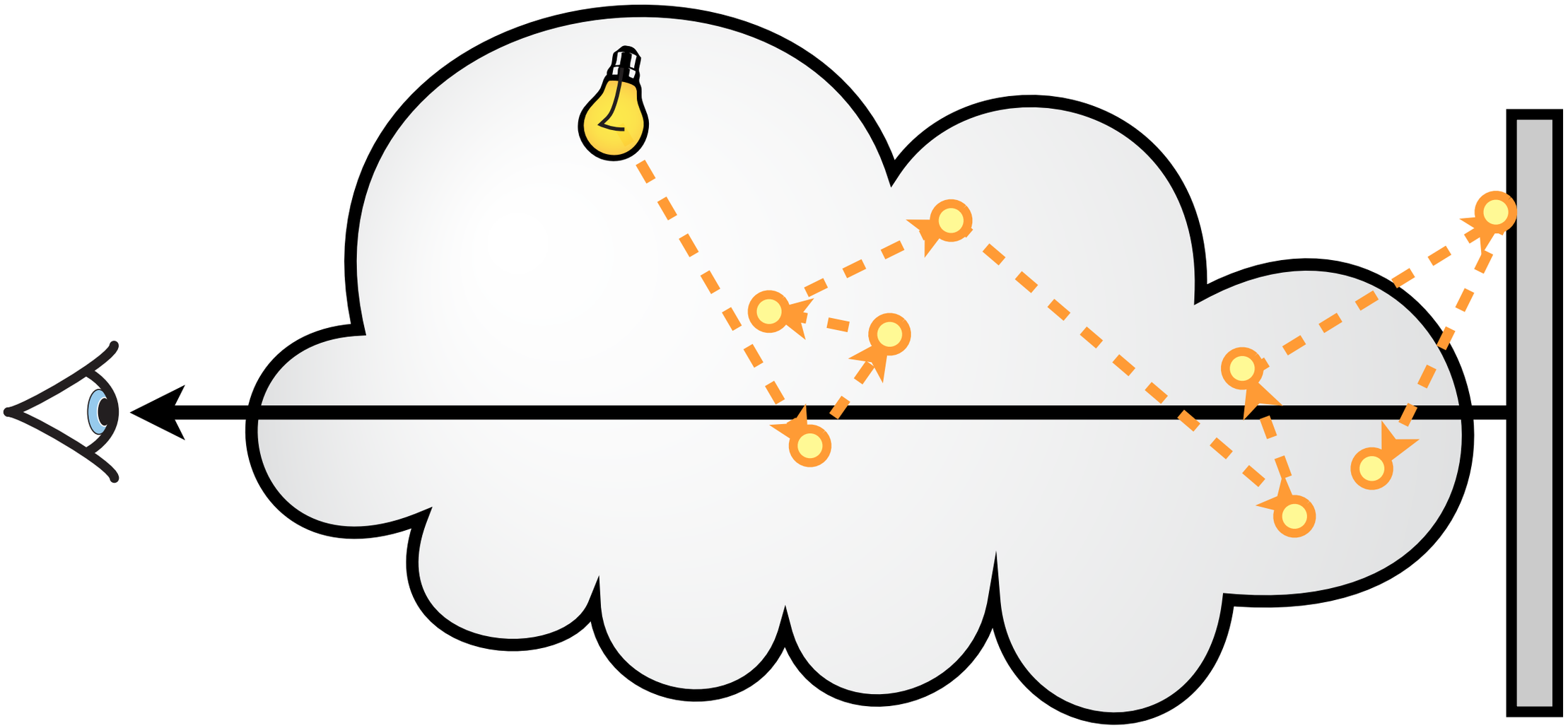
- radiosity
  - the zonal method [Rushmeier and Torrence 87]
- (bidirectional) path tracing
  - volumetric path tracing [Veach and Guibas 94; Lafortune and Willems 96]
  - joint importance sampling [Georgiev et al. 13]
- photon mapping
  - volumetric photon mapping [Jensen and Christensen 98]
  - beam radiance estimate [Jarosz et al. 08]
  - (progressive) photon beams [Jarosz et al. 11a,b]
- (ir)radiance caching
  - volumetric radiance caching [Jarosz et al. 08]
- many-light rendering
  - multidimentional lightcuts [Walter et al. 06]
  - virtual point lights [Engelhardt et al. 2012]
  - virtual ray lights [Novák et al. 2012a]
  - virtual beam lights [Novák et al. 2012b]
- "all" in one
  - unified theory of points, beams, and paths in volumes [Křivánek et al 2014]

Volumetric Photon Mapping

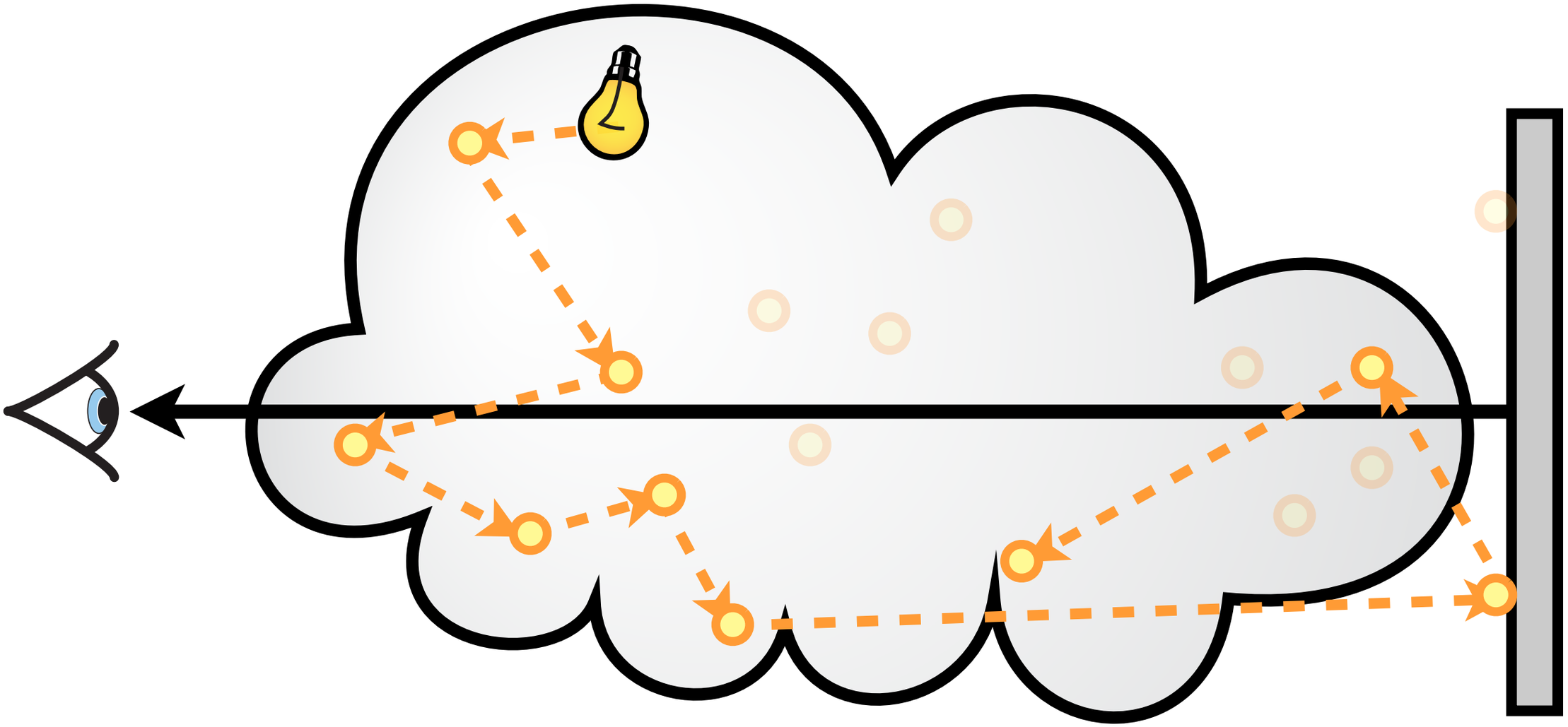
Volumetric Photon Mapping

- Two-pass algorithm:
  - Photon tracing
    - Simulate scattering of photons
  - Rendering
    - Reuse photons to estimate multiple scattering

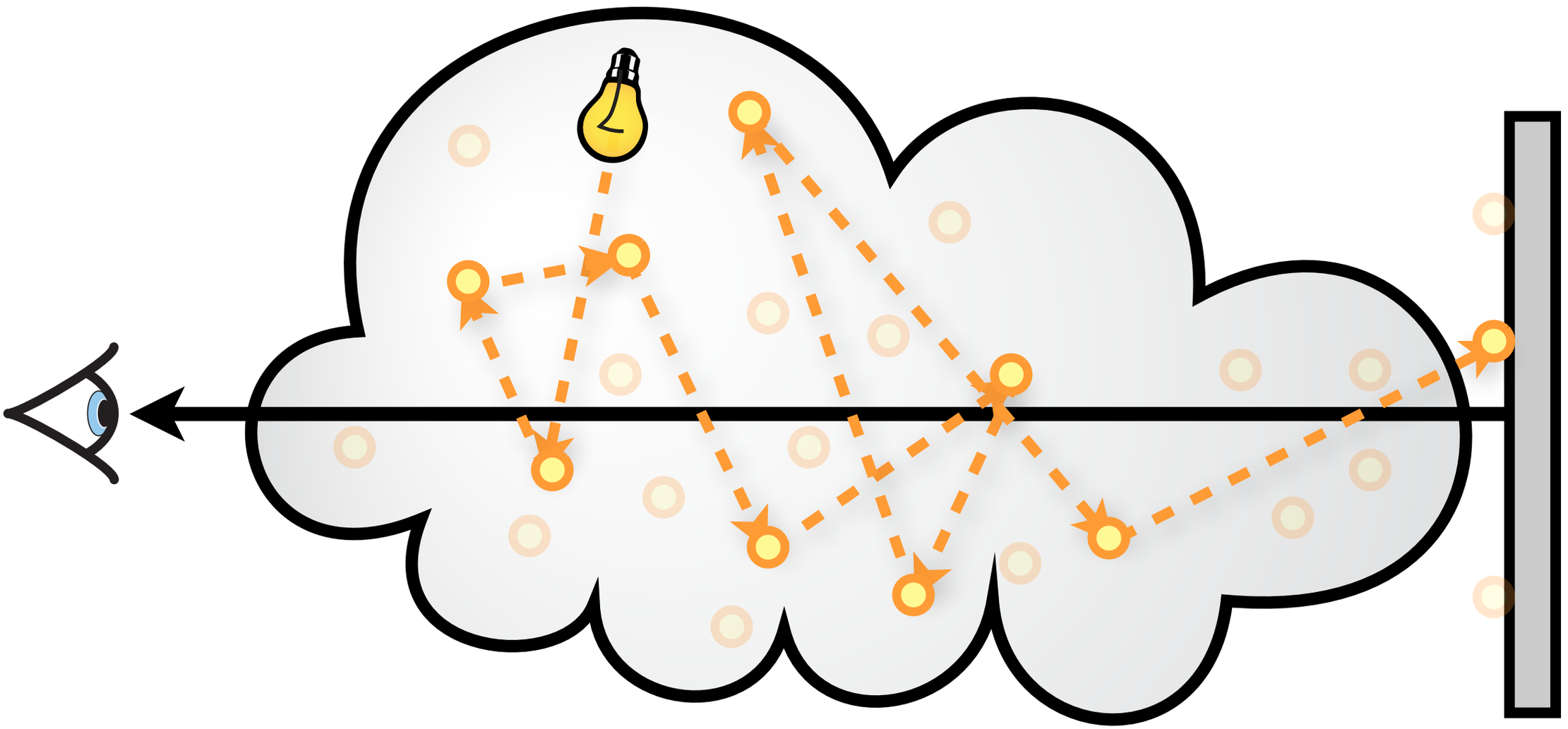
Photon Tracing in Participating Media



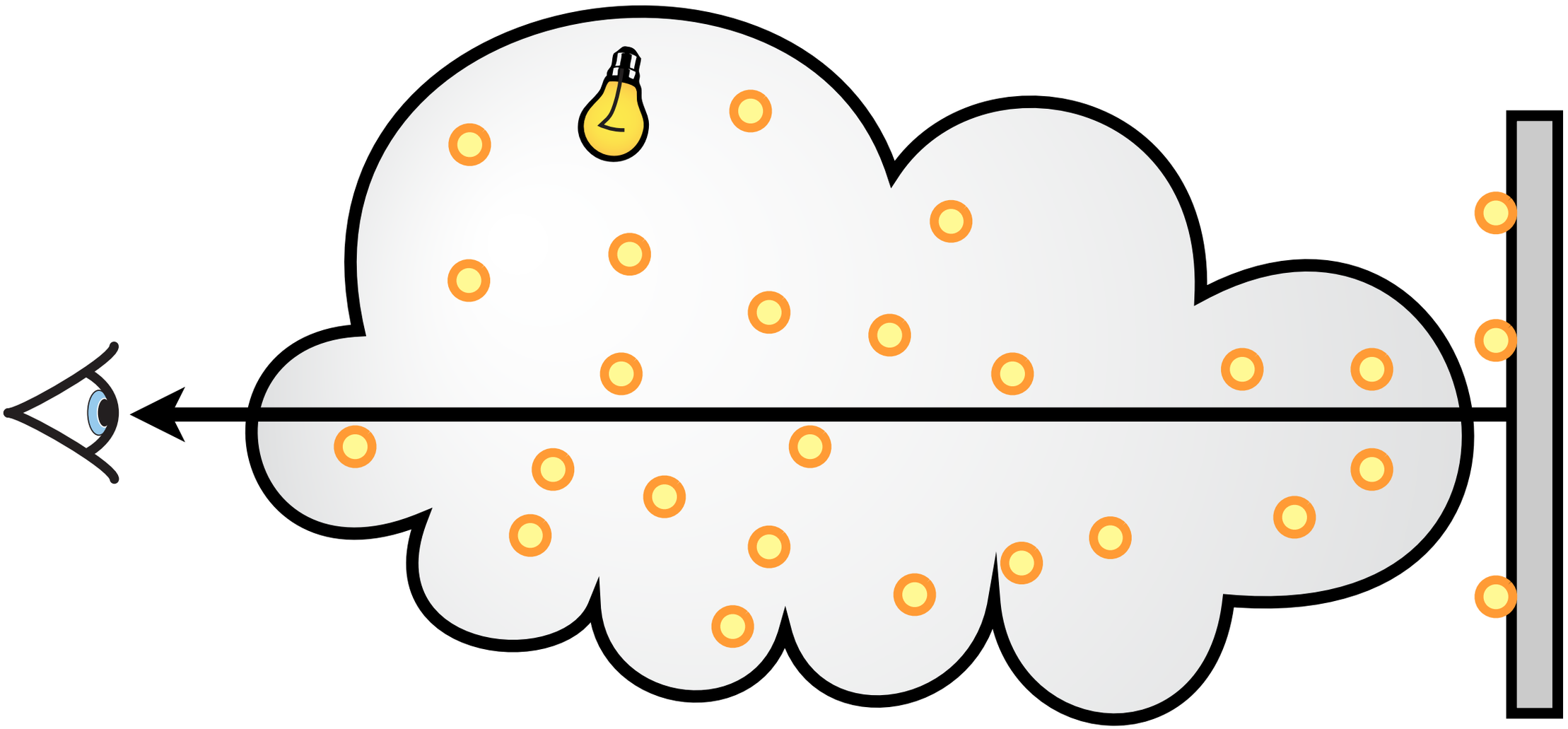
Photon Tracing in Participating Media



Photon Tracing in Participating Media

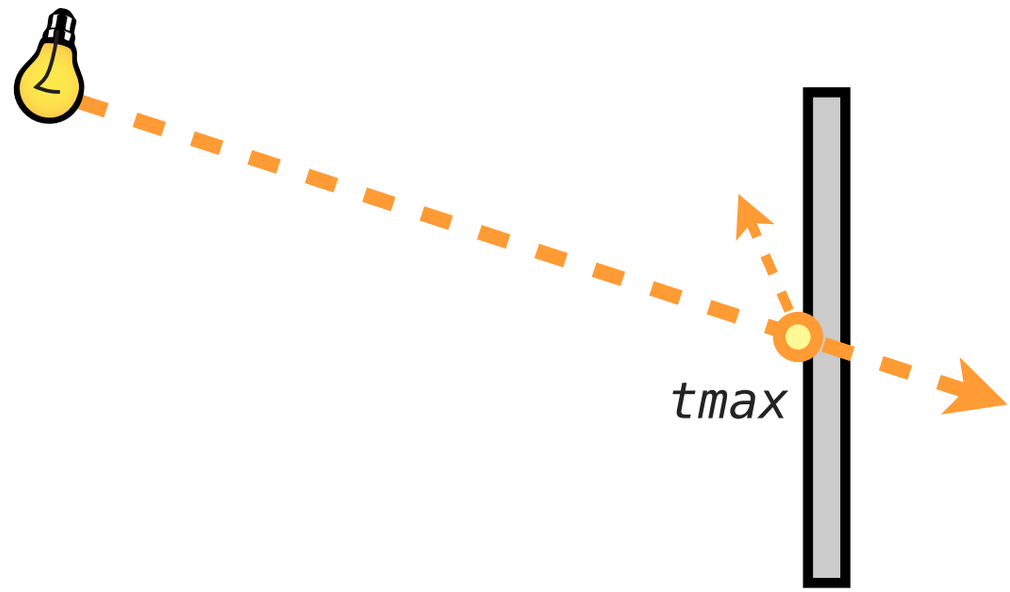


Photon Tracing in Participating Media



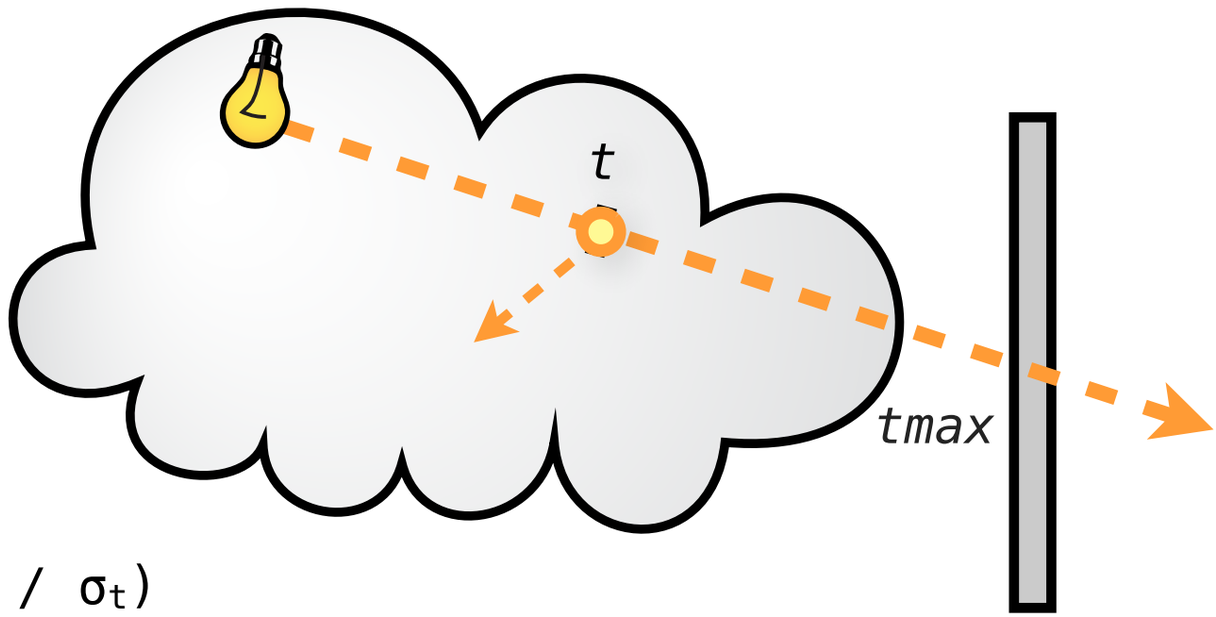
Basic Photon Tracer

```
void vPT(x, ω, Φ)
    tmax = nearestSurfaceHit(x, ω)
    x += tmax * ω // propagate photon
    storeSurfacePhoton(x, ω, Φ)
    (ωi, pdfi) = sampleBRDF(x, ω)
    return vPT(x, ωi, Φ * BRDF(x,ω,ωi) / pdfi)
```



Basic Volumetric Photon Tracer

```
void vPT(x, ω, Φ)
    tmax = nearestSurfaceHit(x, ω)
    t = freeFlightDistance(x, ω)
    if (t < tmax)     // media scattering
        x += t * ω    // propagate photon
        storeVolumePhoton(x, ω, Φ)
        return vPT(x, samplePF(), Φ * σs / σt)
    else              // surface scattering
        x += tmax * ω // propagate photon
        storeSurfacePhoton(x, ω, Φ)
        (ωi, pdfi) = sampleBRDF(x, ω)
        return vPT(x, ωi, Φ * BRDF(x,ω,ωi) / pdfi)
```

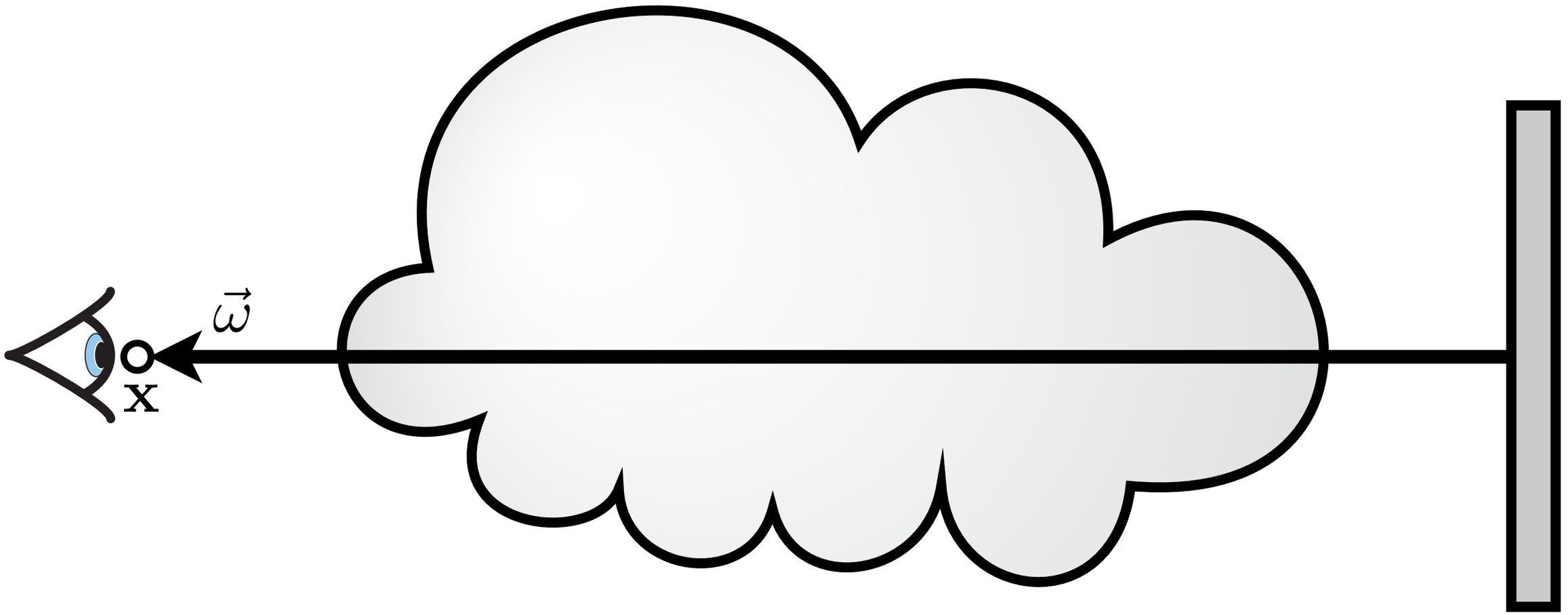


Two-pass Algorithm

- Photon tracing
  - Simulate scattering of photons
- Rendering
  - Reuse photons to estimate multiple scattering

Volume Rendering Equation

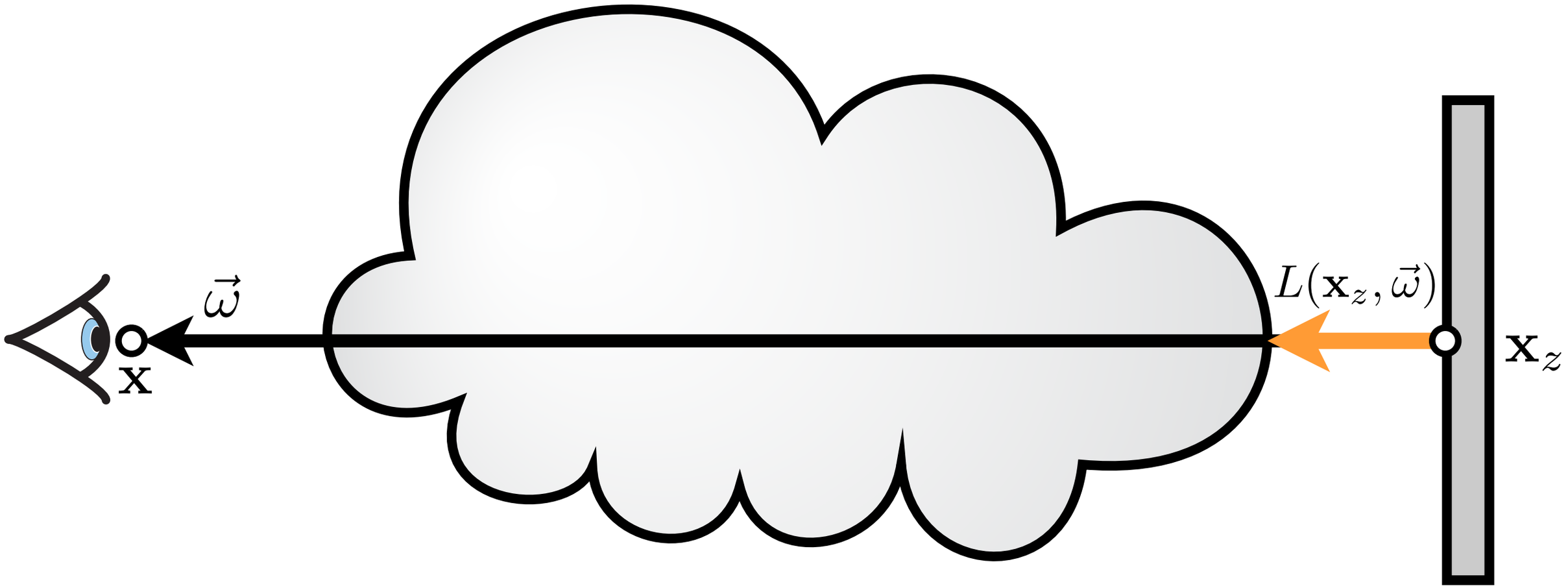
*without emission



$$L(\mathbf{x}, \vec{\omega}) = \int_0^z T_r(\mathbf{x}, \mathbf{x}_t)\sigma_s(\mathbf{x}_t)L_s(\mathbf{x}_t, \vec{\omega})dt \; + T_r(\mathbf{x}, \mathbf{x}_z)L(\mathbf{x}_z, \vec{\omega})$$

Volume Rendering Equation

*without emission

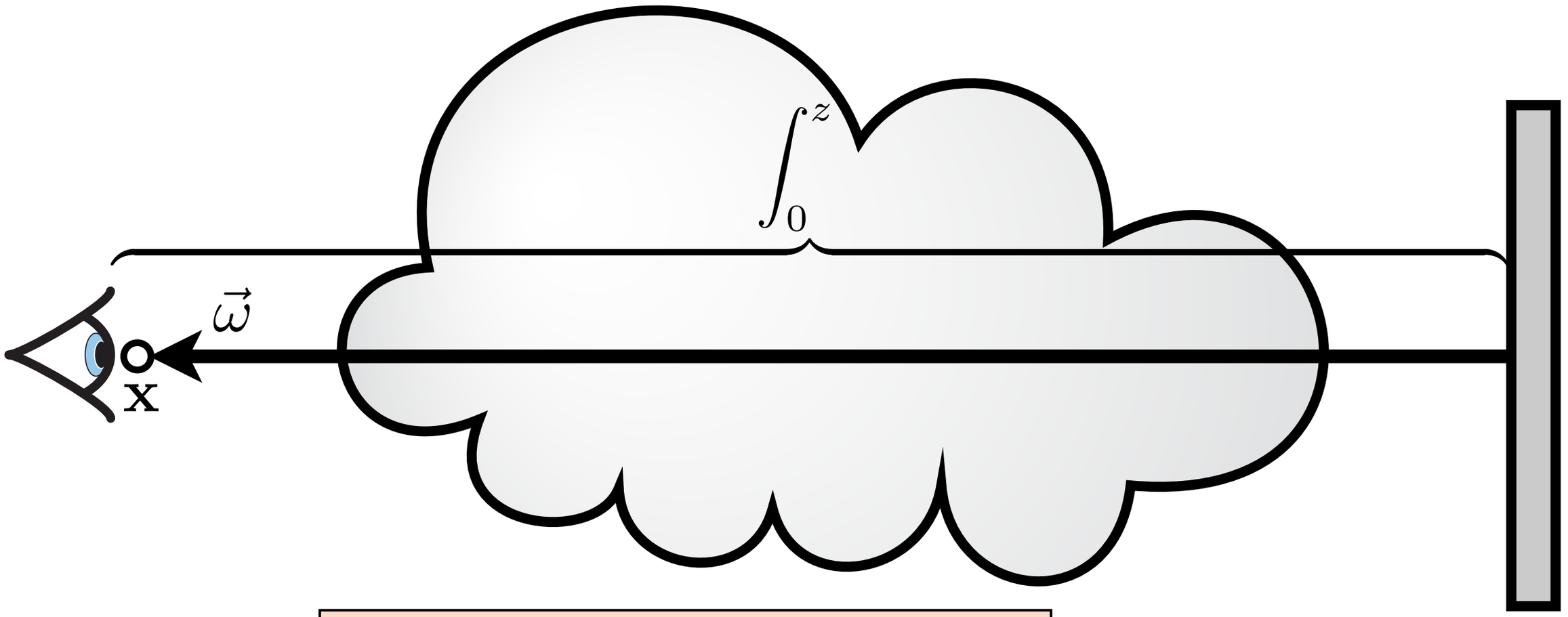


$$L(\mathbf{x}, \vec{\omega}) = \int_0^z T_r(\mathbf{x}, \mathbf{x}_t)\sigma_s(\mathbf{x}_t)L_s(\mathbf{x}_t, \vec{\omega})dt + \boxed{T_r(\mathbf{x}, \mathbf{x}_z)L(\mathbf{x}_z, \vec{\omega})}$$

Reduced surface radiance

Volume Rendering Equation

*without emission

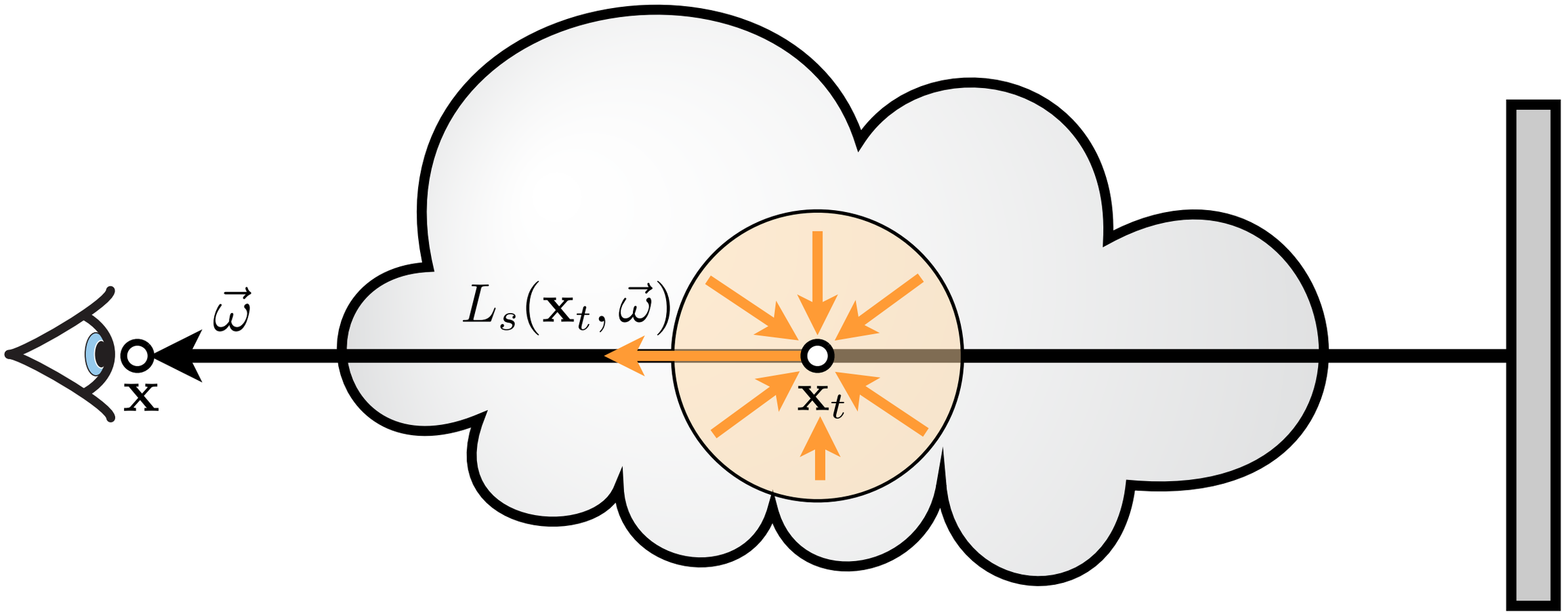


$$L(\mathbf{x},\vec{\omega}) = \int_0^z T_r(\mathbf{x},\mathbf{x}_t)\sigma_s(\mathbf{x}_t)L_s(\mathbf{x}_t,\vec{\omega})dt + T_r(\mathbf{x},\mathbf{x}_z)L(\mathbf{x}_z,\vec{\omega})$$

Accumulated in-scattered radiance

Volume Rendering Equation

*without emission



$$L(\mathbf{x}, \vec{\omega}) = \int_0^z T_r(\mathbf{x}, \mathbf{x}_t)\sigma_s(\mathbf{x}_t)\boxed{L_s(\mathbf{x}_t, \vec{\omega})}dt \; + T_r(\mathbf{x}, \mathbf{x}_z)L(\mathbf{x}_z, \vec{\omega})$$

$$\boxed{L_s(\mathbf{x}_t, \vec{\omega})} = \int_{S^2} f_p(\mathbf{x}, \vec{\omega}', \vec{\omega})L_i(\mathbf{x}_t, \vec{\omega}')\, d\vec{\omega}'$$

Volume Rendering Equation

*without emission

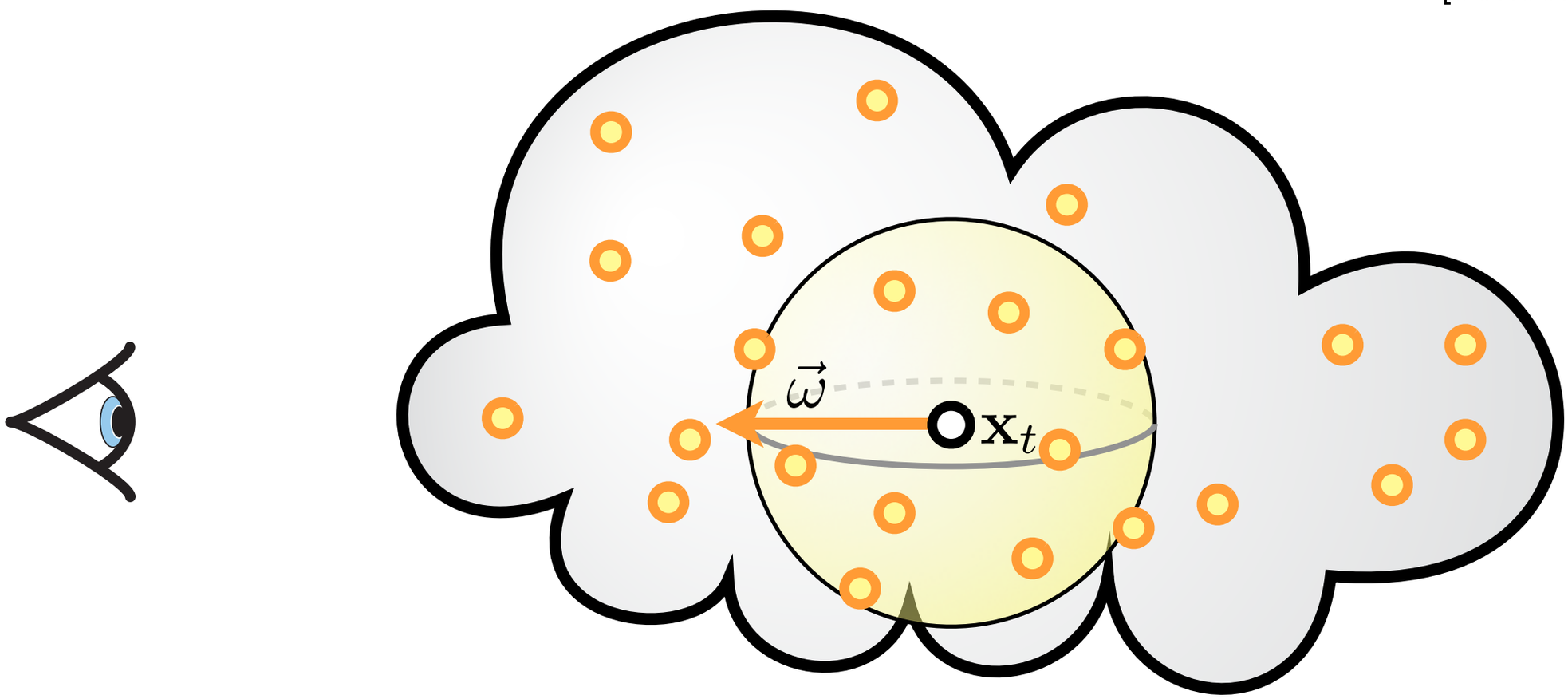
- Two approaches:
  - Volumetric radiance estimate
    - Ray-marching + 3D volume radiance estimate [Jensen and Christensen 98]
  - Beam Radiance Estimate
    - [Jarosz et al. 08]

$$L(\mathbf{x},\vec{\omega}) = \int_0^z T_r(\mathbf{x},\mathbf{x}_t)\sigma_s(\mathbf{x}_t)L_s(\mathbf{x}_t,\vec{\omega})dt \; + T_r(\mathbf{x},\mathbf{x}_z)L(\mathbf{x}_z,\vec{\omega})$$

$$L_s(\mathbf{x}_t,\vec{\omega}) = \int_{S^2} f_p(\mathbf{x},\vec{\omega}',\vec{\omega})L_i(\mathbf{x}_t,\vec{\omega}')\,d\vec{\omega}'$$

Volumetric Radiance Estimate

[Jensen & Christensen 98]

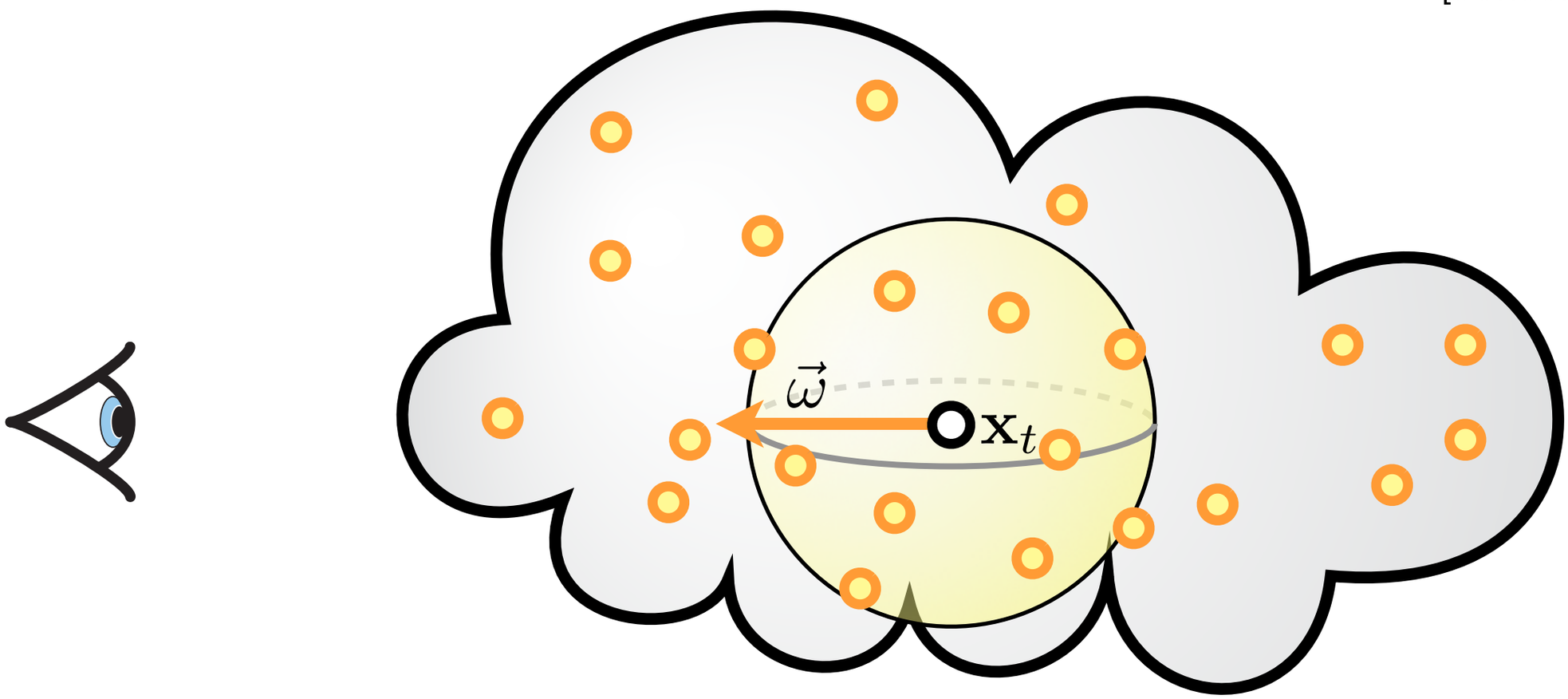


$$L(\mathbf{x}, \vec{\omega}) = \int_0^z T_r(\mathbf{x}, \mathbf{x}_t)\sigma_s(\mathbf{x}_t)\boxed{L_s(\mathbf{x}_t, \vec{\omega})}dt \; + T_r(\mathbf{x}, \mathbf{x}_z)L(\mathbf{x}_z, \vec{\omega})$$

$$\boxed{L_s(\mathbf{x}_t, \vec{\omega})} = \int_{S^2} f_p(\mathbf{x}, \vec{\omega}', \vec{\omega})L_i(\mathbf{x}_t, \vec{\omega}')\, d\vec{\omega}'$$

Volumetric Radiance Estimate

[Jensen & Christensen 98]

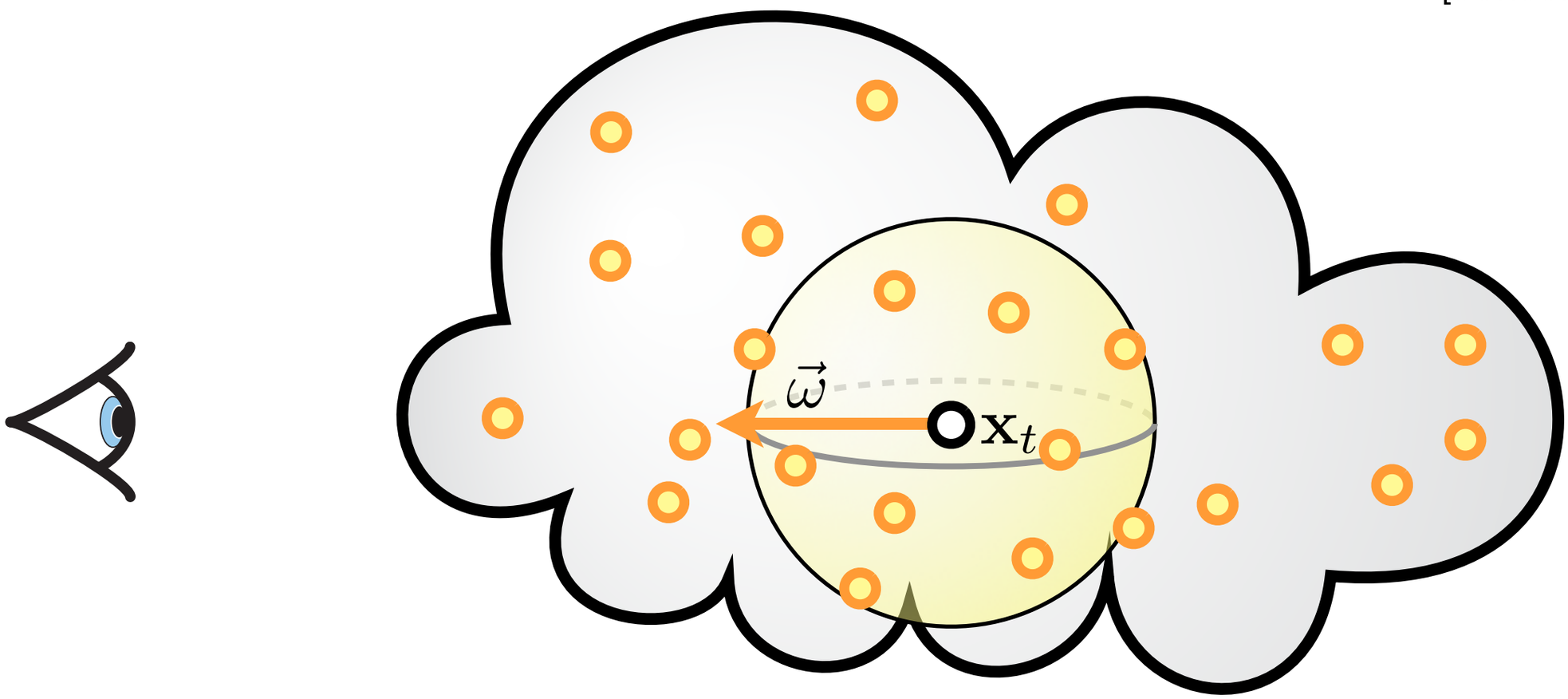


$$L(\mathbf{x}, \vec{\omega}) = \int_0^z T_r(\mathbf{x}, \mathbf{x}_t)\sigma_s(\mathbf{x}_t)\boxed{L_s(\mathbf{x}_t, \vec{\omega})}dt \; + T_r(\mathbf{x}, \mathbf{x}_z)L(\mathbf{x}_z, \vec{\omega})$$

$$\boxed{L_s(\mathbf{x}_t, \vec{\omega})} = \int_{S^2} f_p(\mathbf{x}, \vec{\omega}', \vec{\omega})L_i(\mathbf{x}_t, \vec{\omega}')\, d\vec{\omega}'$$

Volumetric Radiance Estimate

[Jensen & Christensen 98]

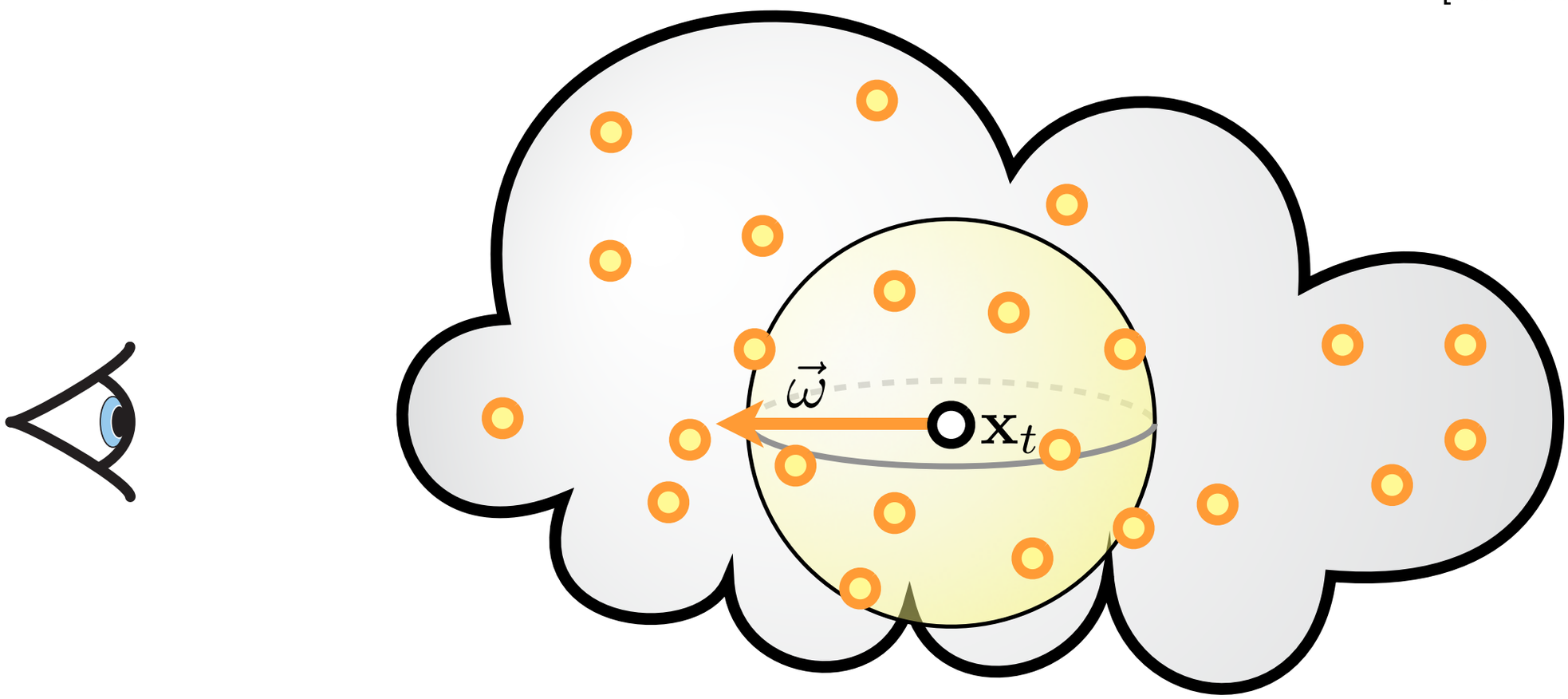


$$L(\mathbf{x}, \vec{\omega}) = \int_0^z T_r(\mathbf{x}, \mathbf{x}_t)\sigma_s(\mathbf{x}_t)\boxed{L_s(\mathbf{x}_t, \vec{\omega})}dt + T_r(\mathbf{x}, \mathbf{x}_z)L(\mathbf{x}_z, \vec{\omega})$$

$$\boxed{L_s(\mathbf{x}_t, \vec{\omega})} \approx \sum_{i=1}^{k} f_p(\mathbf{x}_t, \vec{\omega}_i, \vec{\omega}) \frac{\Phi_i(\mathbf{x}_t, \vec{\omega}_i)}{\mathcal{V}(\mathbf{x}_t)}$$

Volumetric Radiance Estimate

[Jensen & Christensen 98]

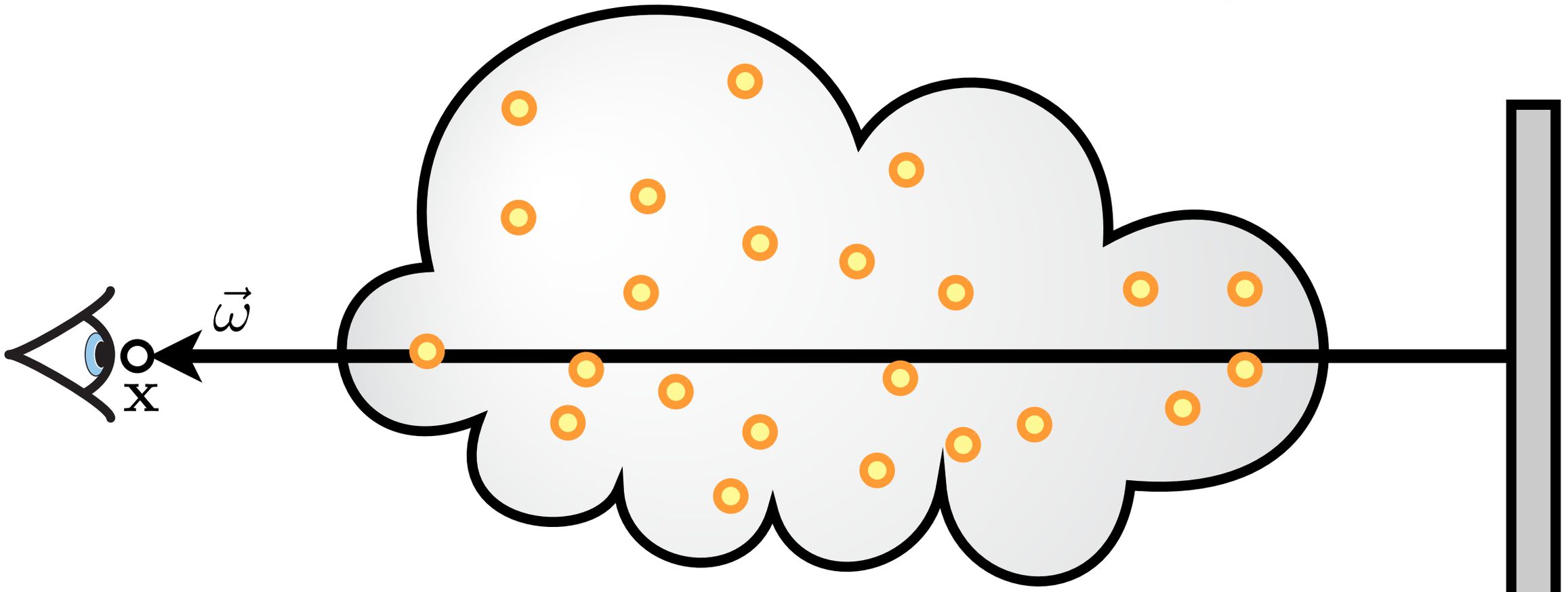


$$L(\mathbf{x}, \vec{\omega}) = \int_0^z T_r(\mathbf{x}, \mathbf{x}_t)\sigma_s(\mathbf{x}_t)\boxed{L_s(\mathbf{x}_t, \vec{\omega})}dt \; + T_r(\mathbf{x}, \mathbf{x}_z)L(\mathbf{x}_z, \vec{\omega})$$

$$\boxed{L_s(\mathbf{x}_t, \vec{\omega})} \approx \sum_{i=1}^{k} f_p(\mathbf{x}_t, \vec{\omega}_i, \vec{\omega}) \frac{\Phi_i(\mathbf{x}_t, \vec{\omega}_i)}{\frac{4}{3}\pi r(\mathbf{x}_t)^3}$$

Volumetric Radiance Estimate

[Jensen & Christensen 98]

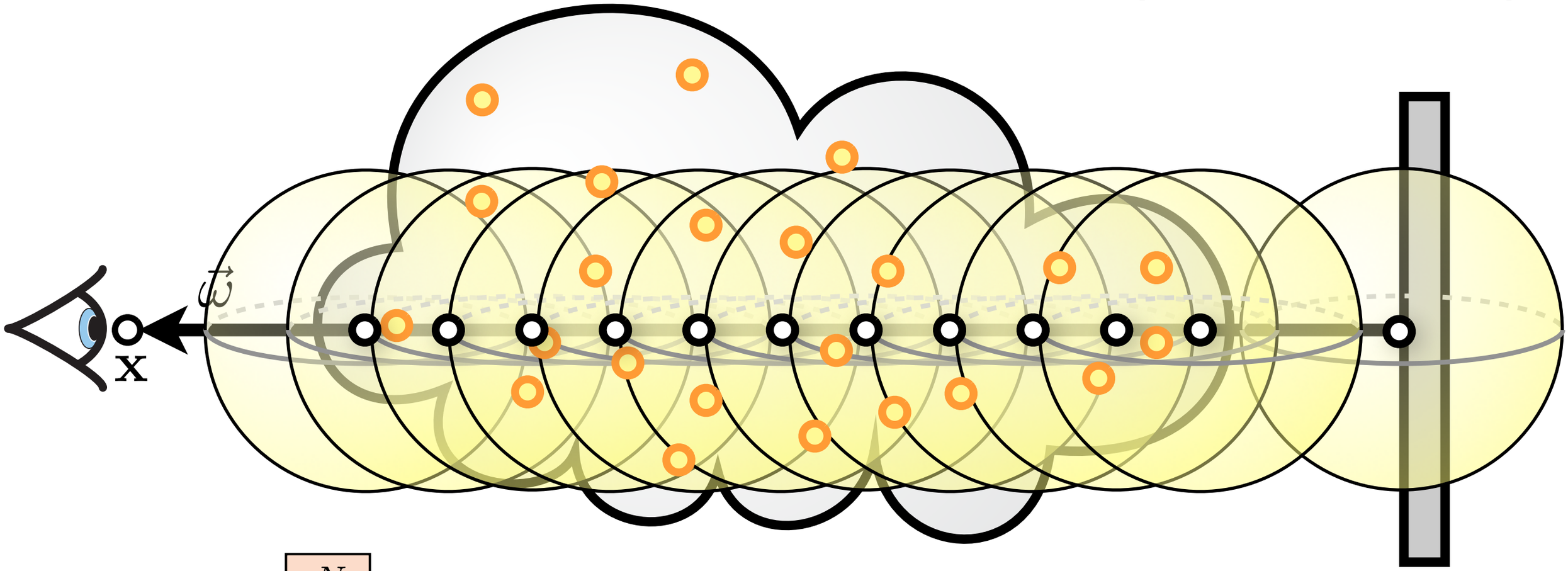


$$L(\mathbf{x},\vec{\omega}) = \int_0^z T_r(\mathbf{x},\mathbf{x}_t)\sigma_s(\mathbf{x}_t)L_s(\mathbf{x}_t,\vec{\omega})dt + T_r(\mathbf{x},\mathbf{x}_z)L(\mathbf{x}_z,\vec{\omega})$$

Approximate/compute using ray-marching

Ray-marching

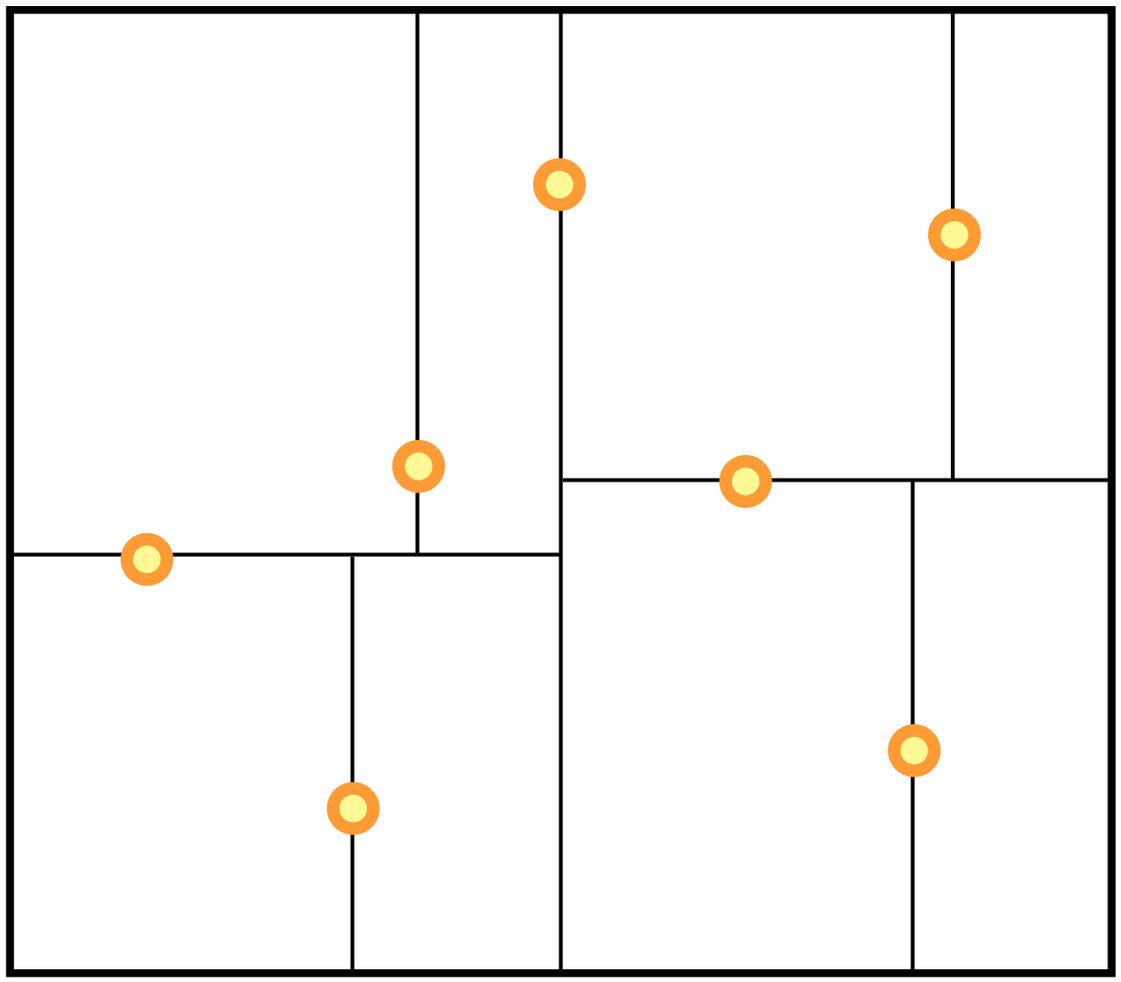
[Jensen & Christensen 98]



$$L(\mathbf{x},\vec{\omega}) = \boxed{\sum_{i=1}^{N}} T_r(\mathbf{x},\mathbf{x}_{t,i})\sigma_s(\mathbf{x}_{t,i})L_s(\mathbf{x}_{t,i},\vec{\omega})\boxed{\Delta t} + T_r(\mathbf{x},\mathbf{x}_z)L(\mathbf{x}_z,\vec{\omega})$$

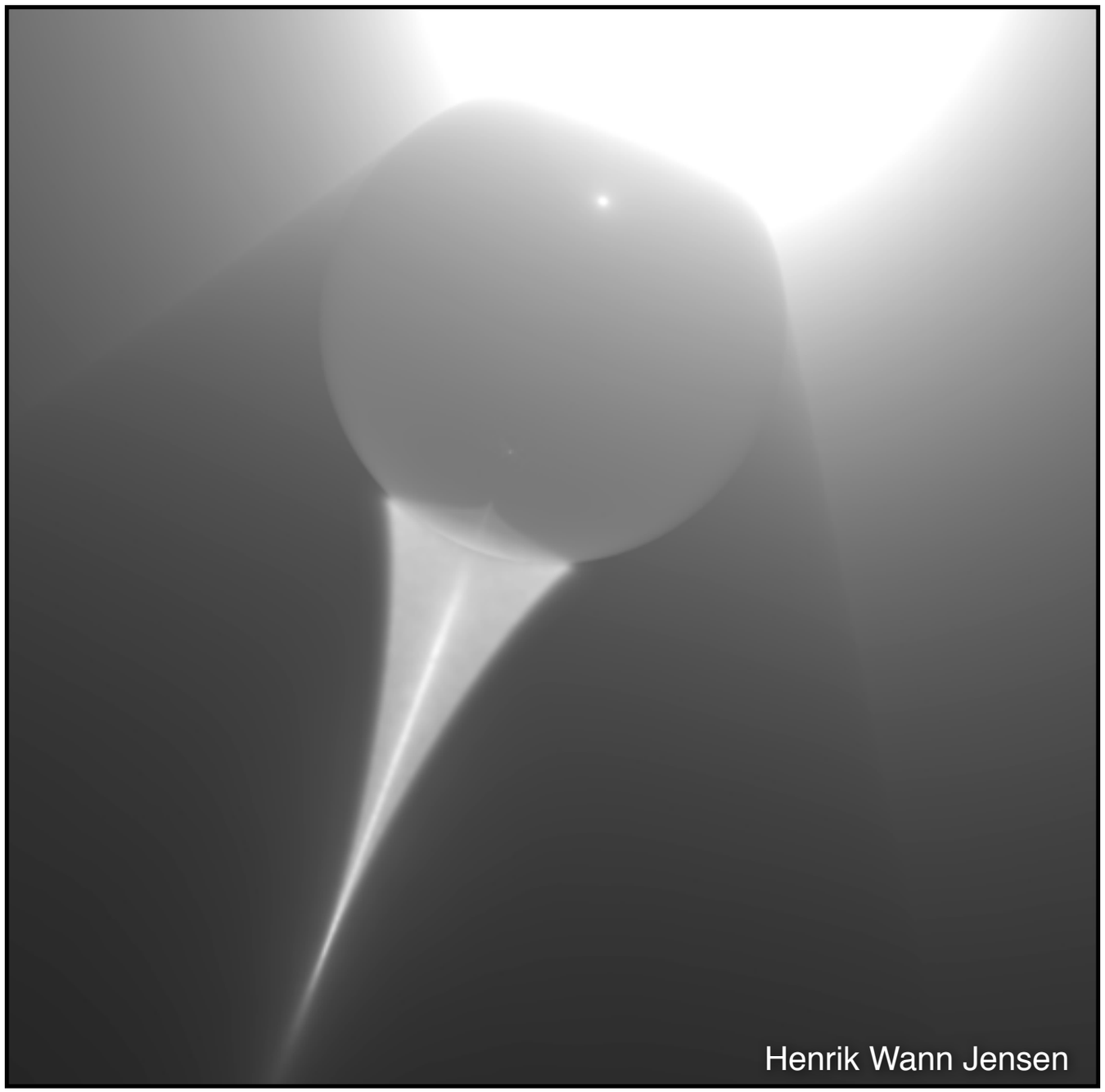
Approximate/compute using ray-marching

Efficient Range Searching



Construct a balanced kD-tree for the photons.

A Volume Caustic

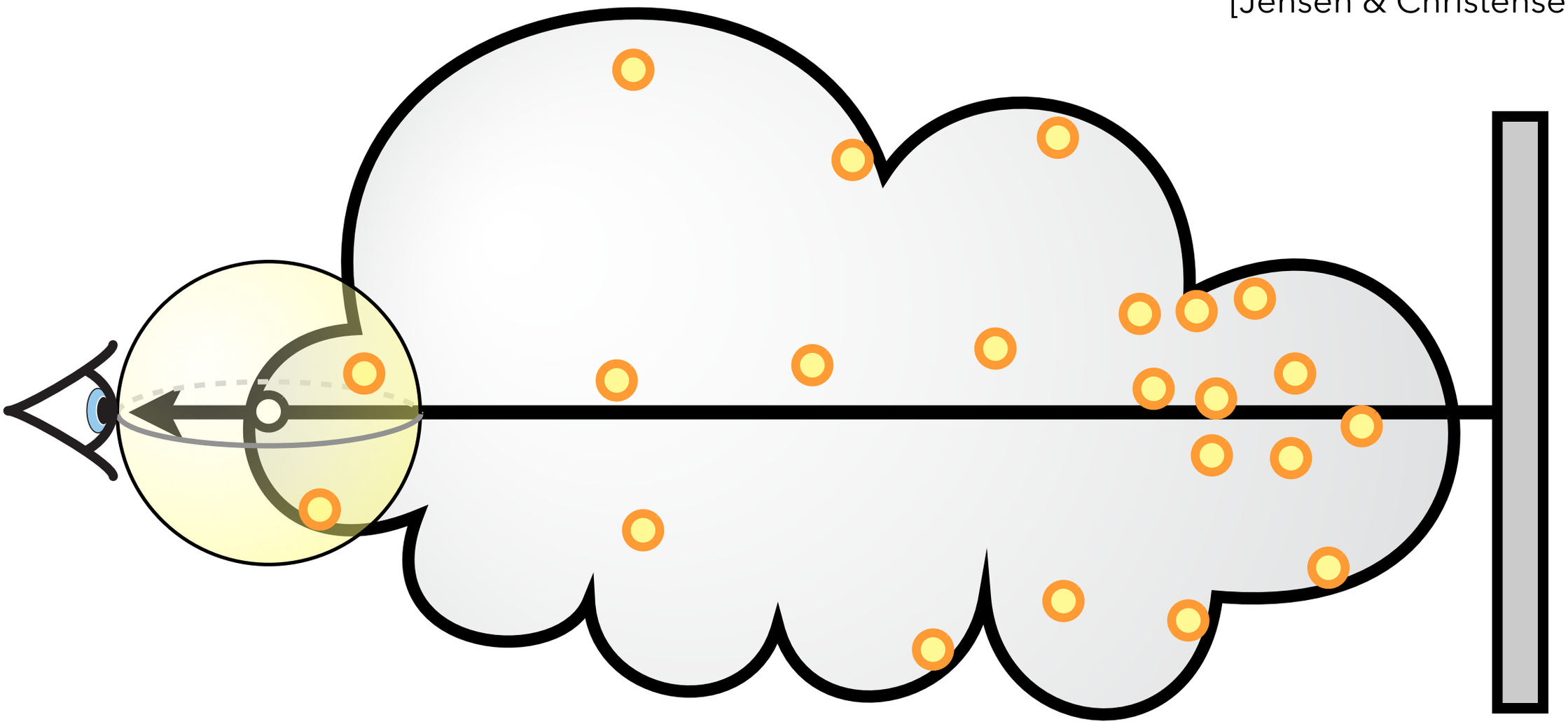


500,000 photons. 1 minute

Subsurface Scattering

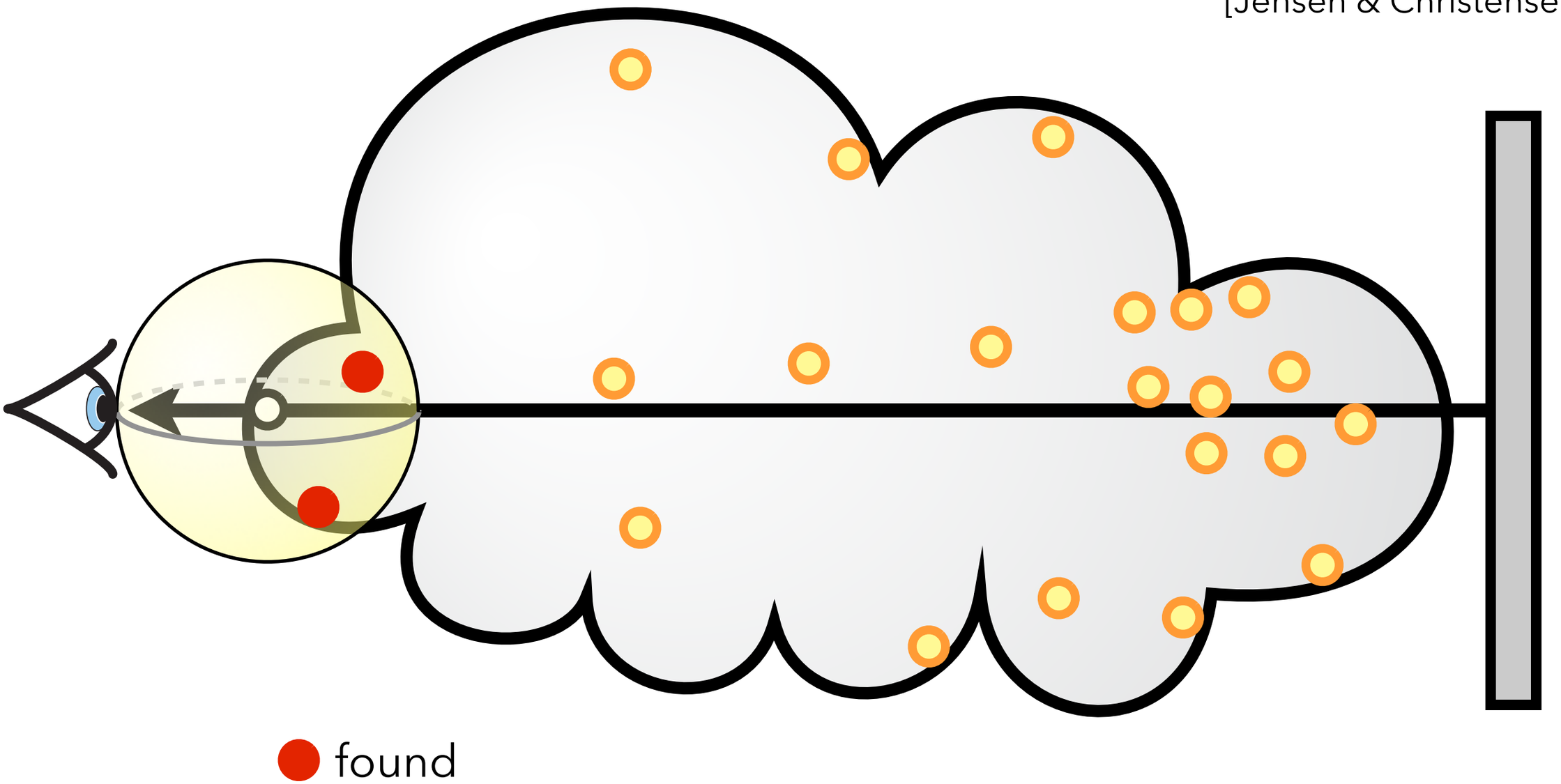
Volumetric Photon Mapping

[Jensen & Christensen 98]



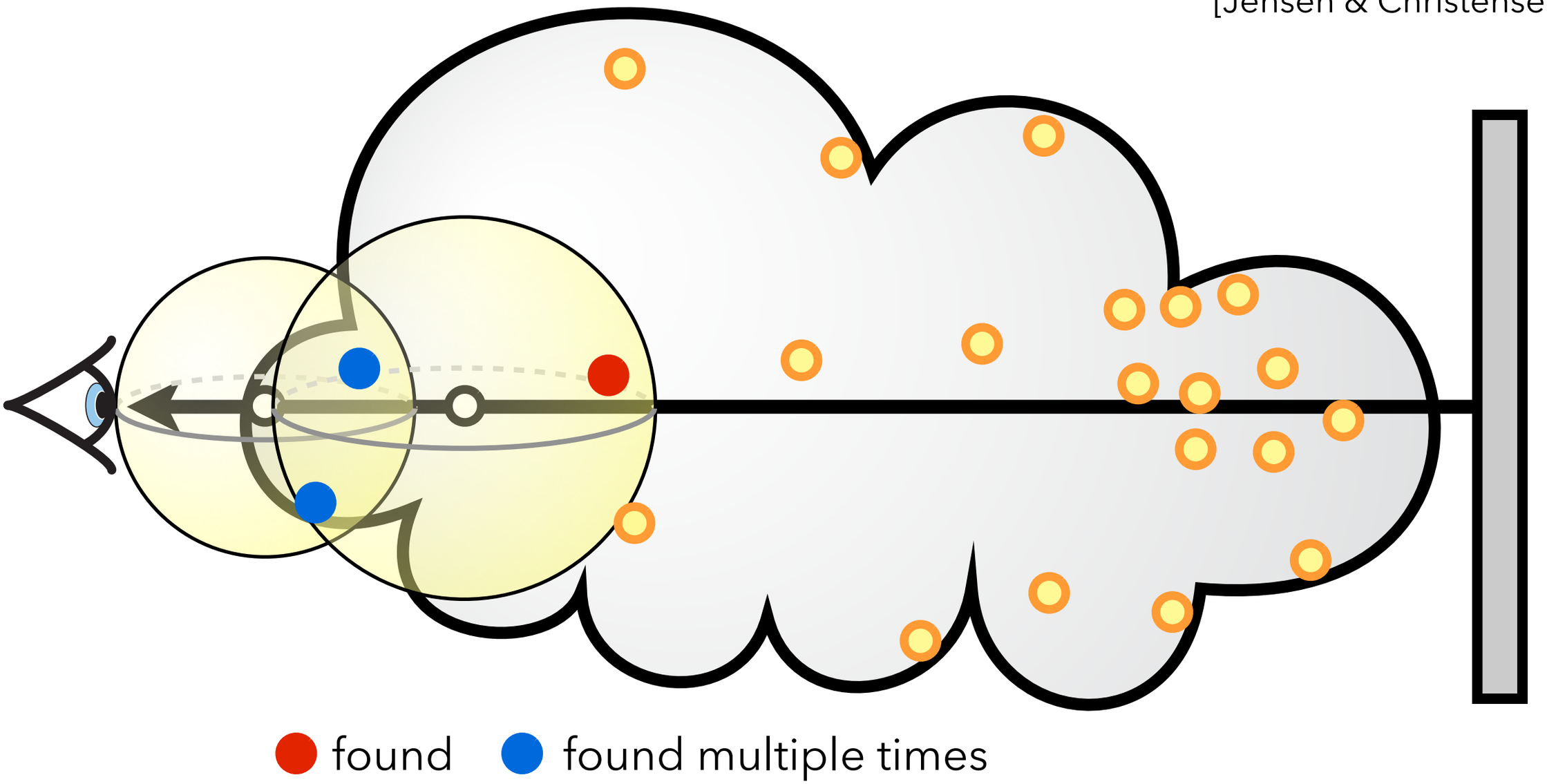
Volumetric Photon Mapping

[Jensen & Christensen 98]



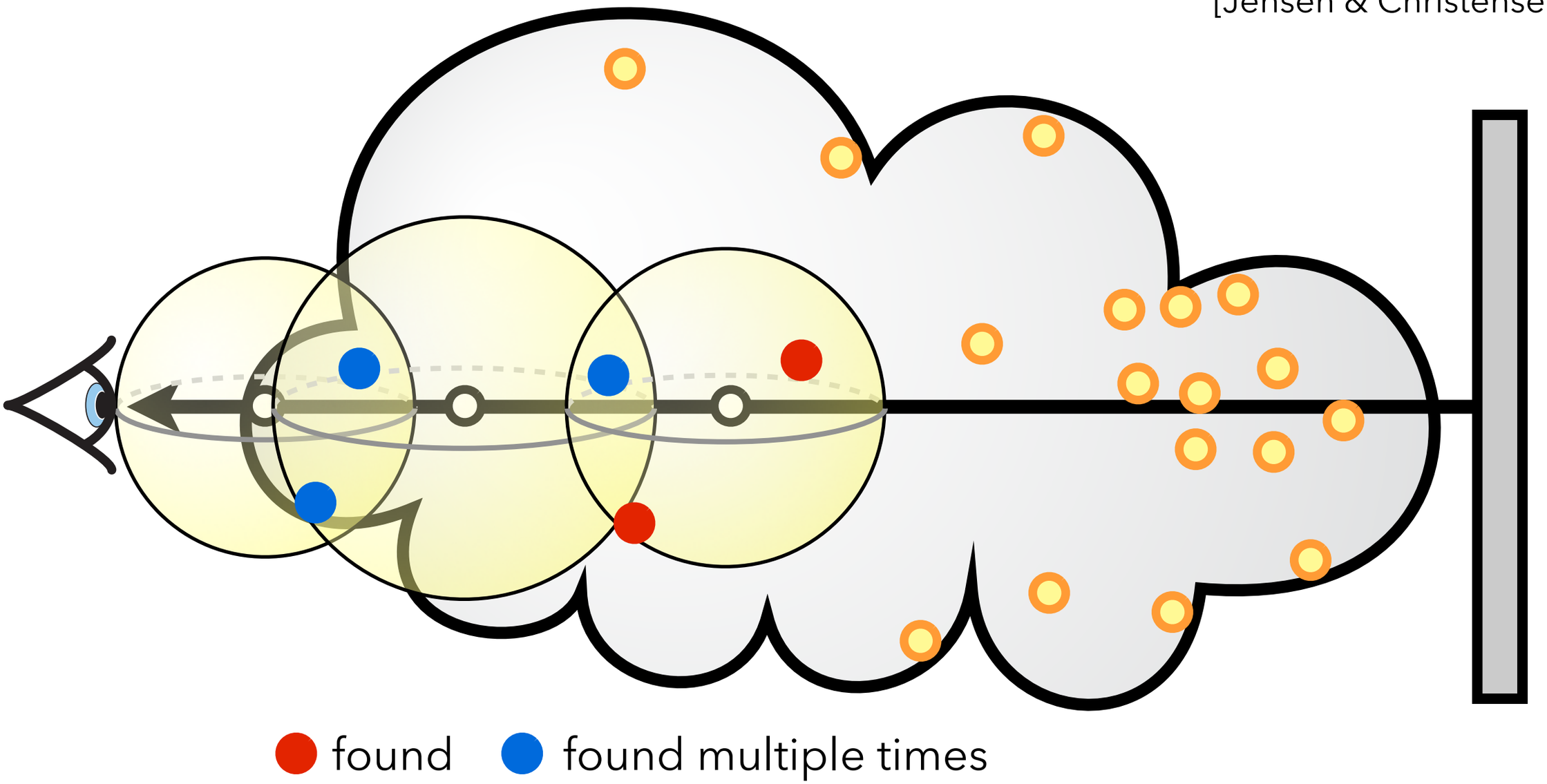
Volumetric Photon Mapping

[Jensen & Christensen 98]



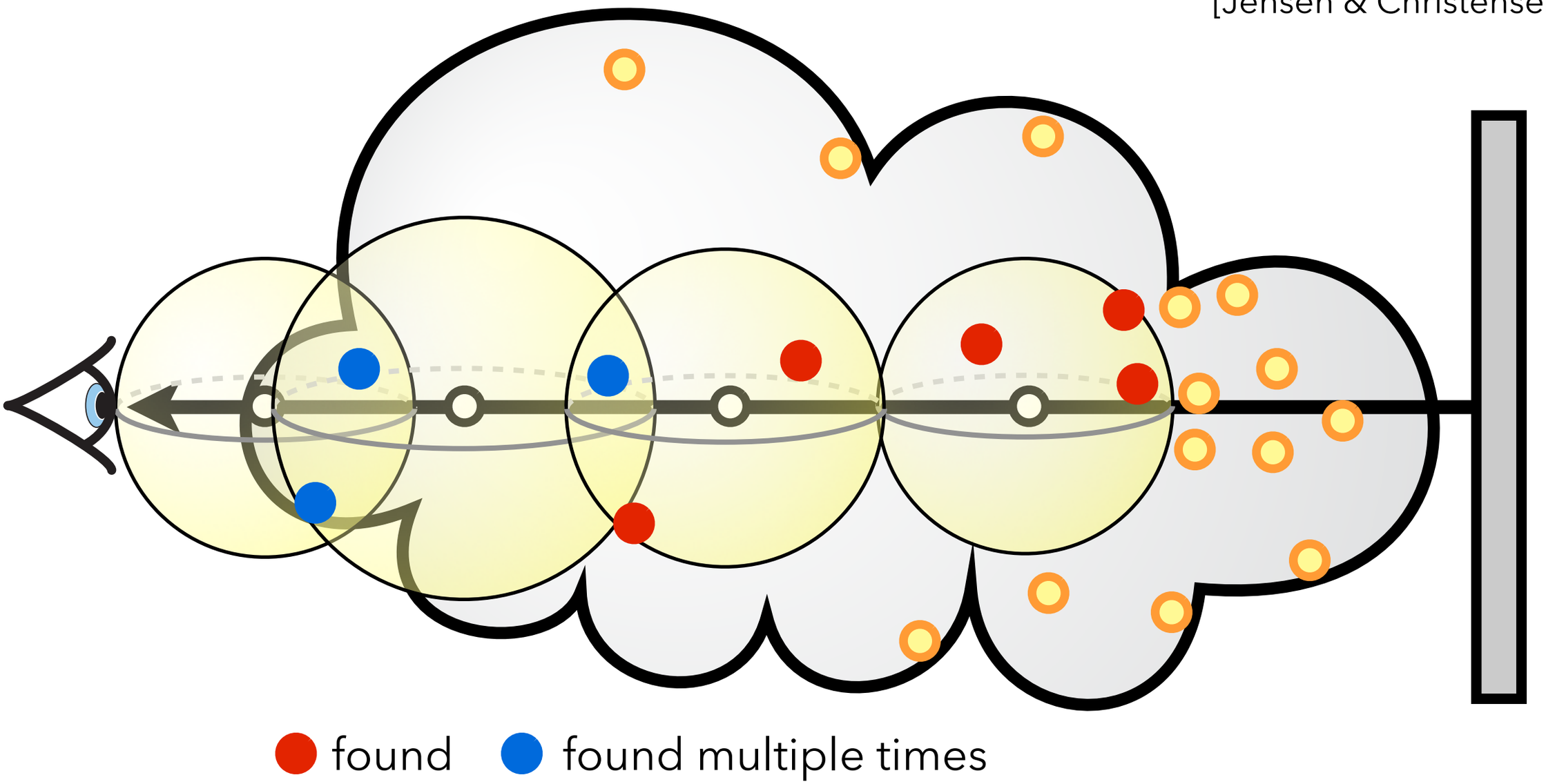
Volumetric Photon Mapping

[Jensen & Christensen 98]



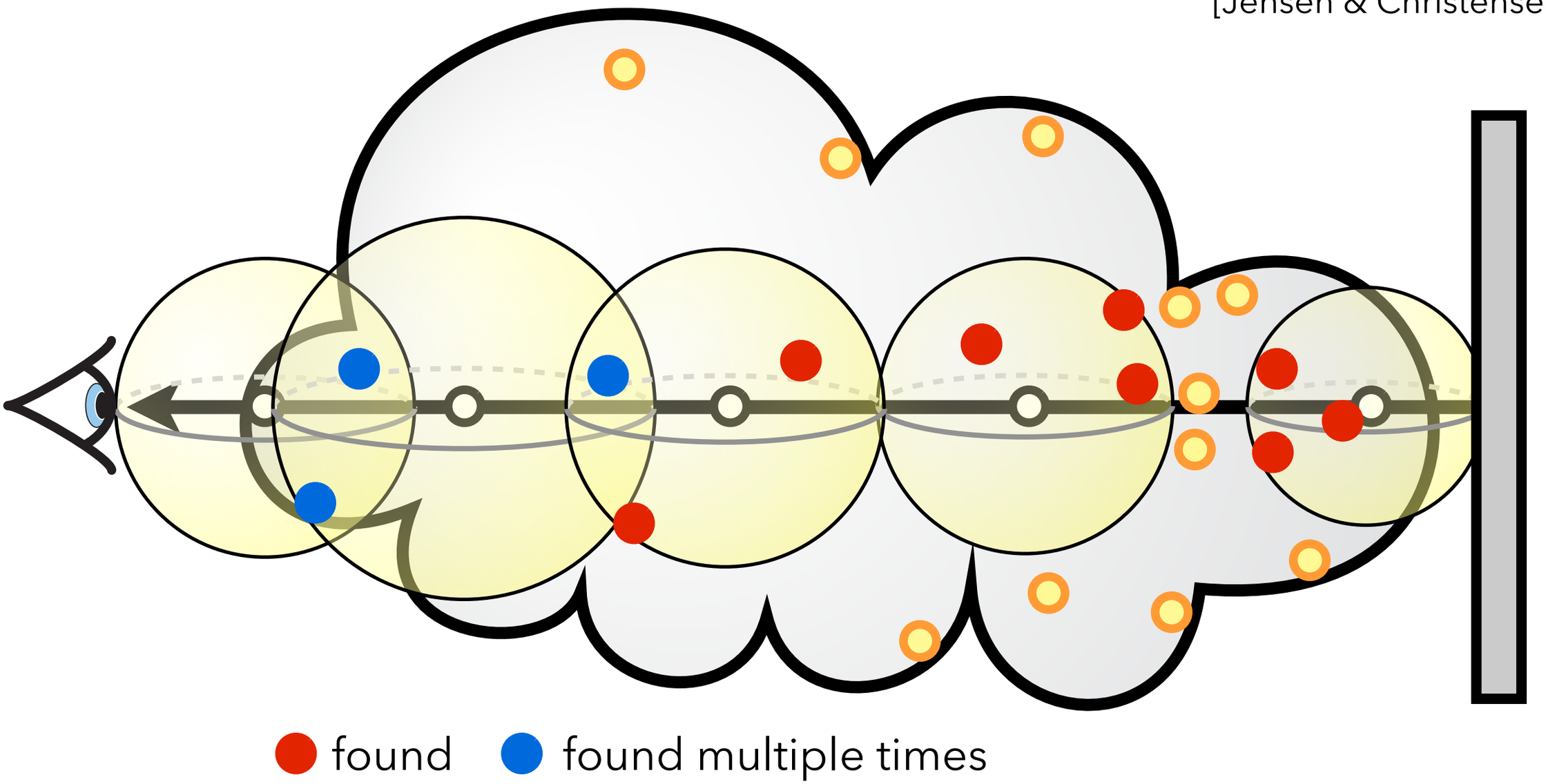
Volumetric Photon Mapping

[Jensen & Christensen 98]



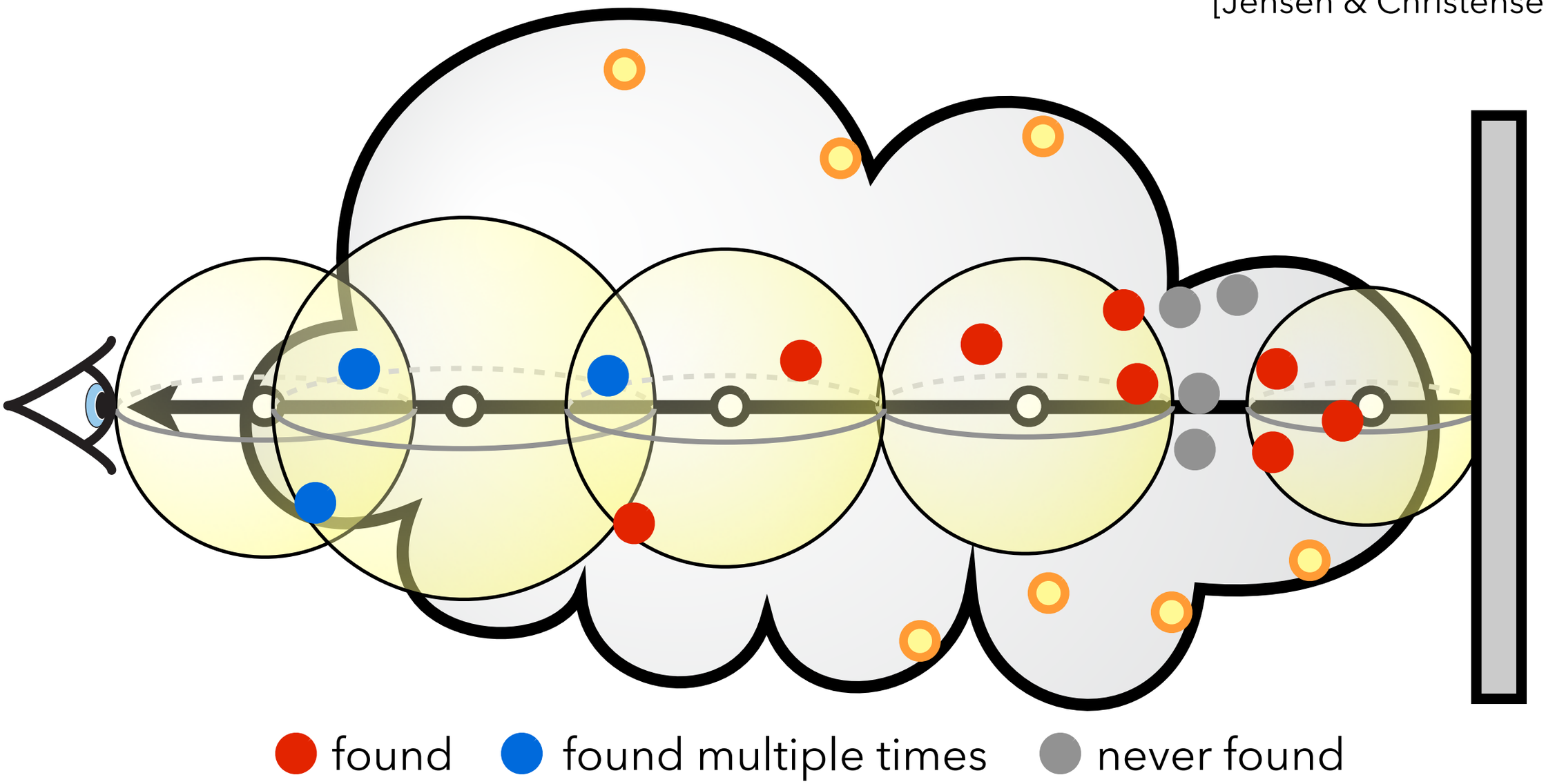
Volumetric Photon Mapping

[Jensen & Christensen 98]



Volumetric Photon Mapping

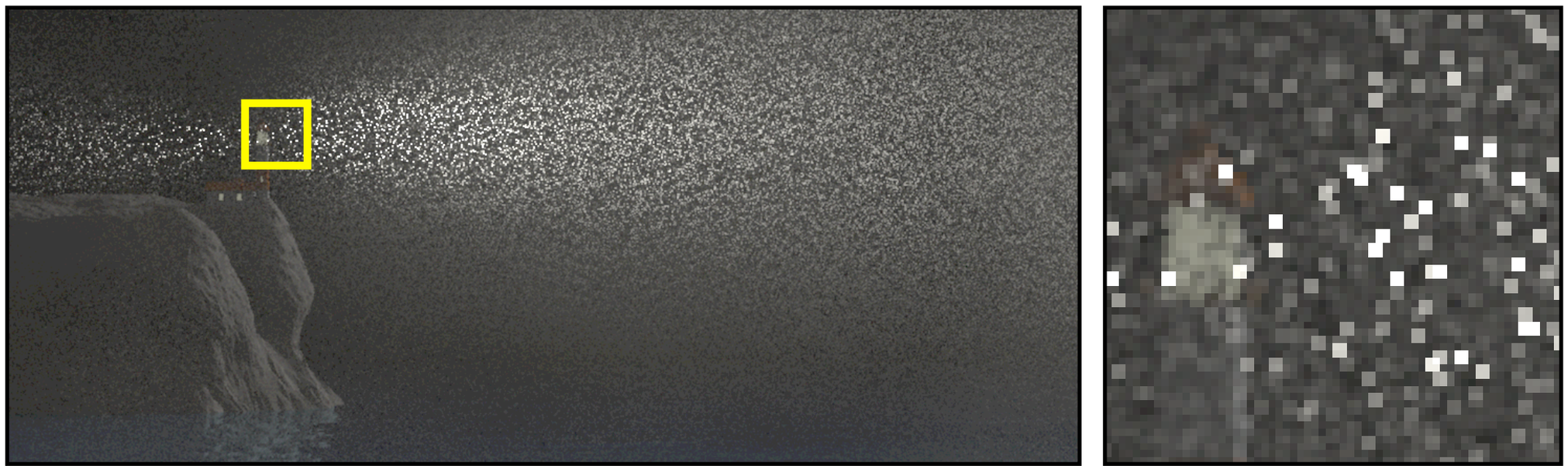
[Jensen & Christensen 98]

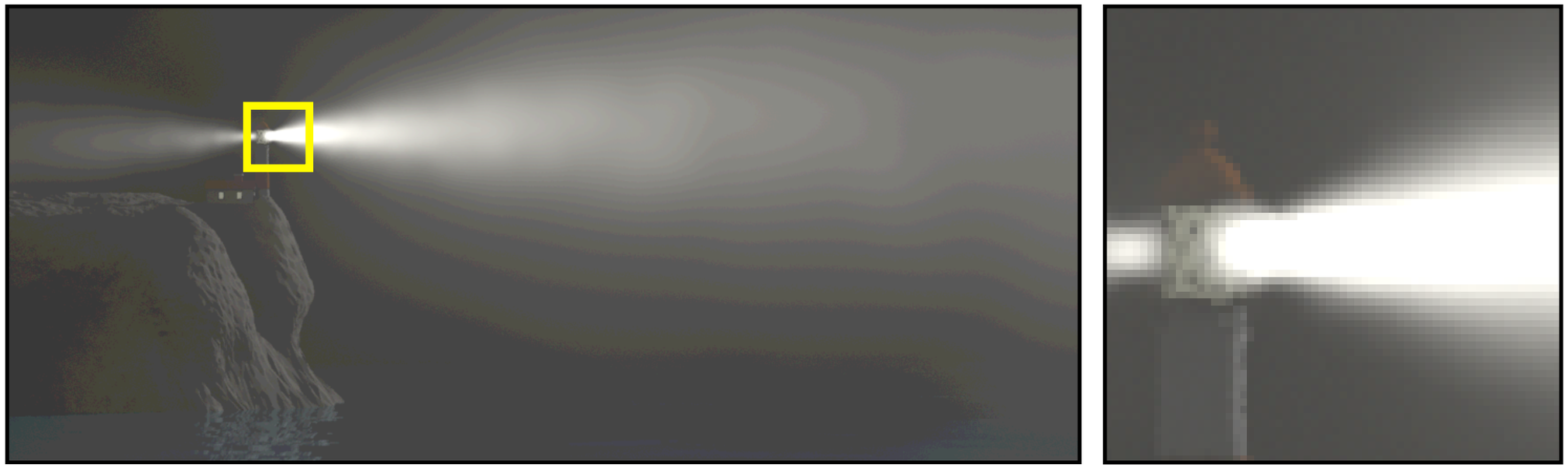


Drawbacks

[Jensen & Christensen 98]

Large Step-size

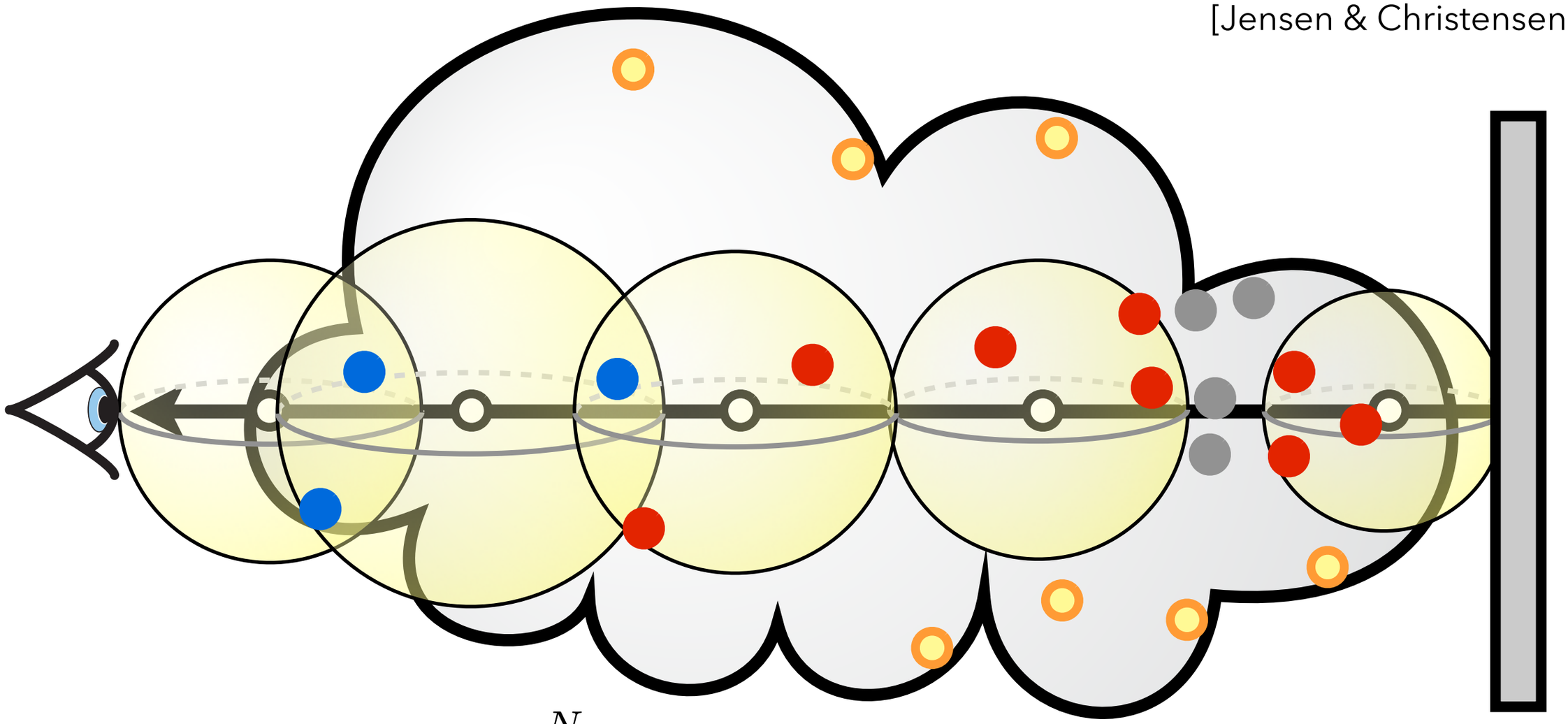




Very Small Step-size

Volumetric Photon Mapping

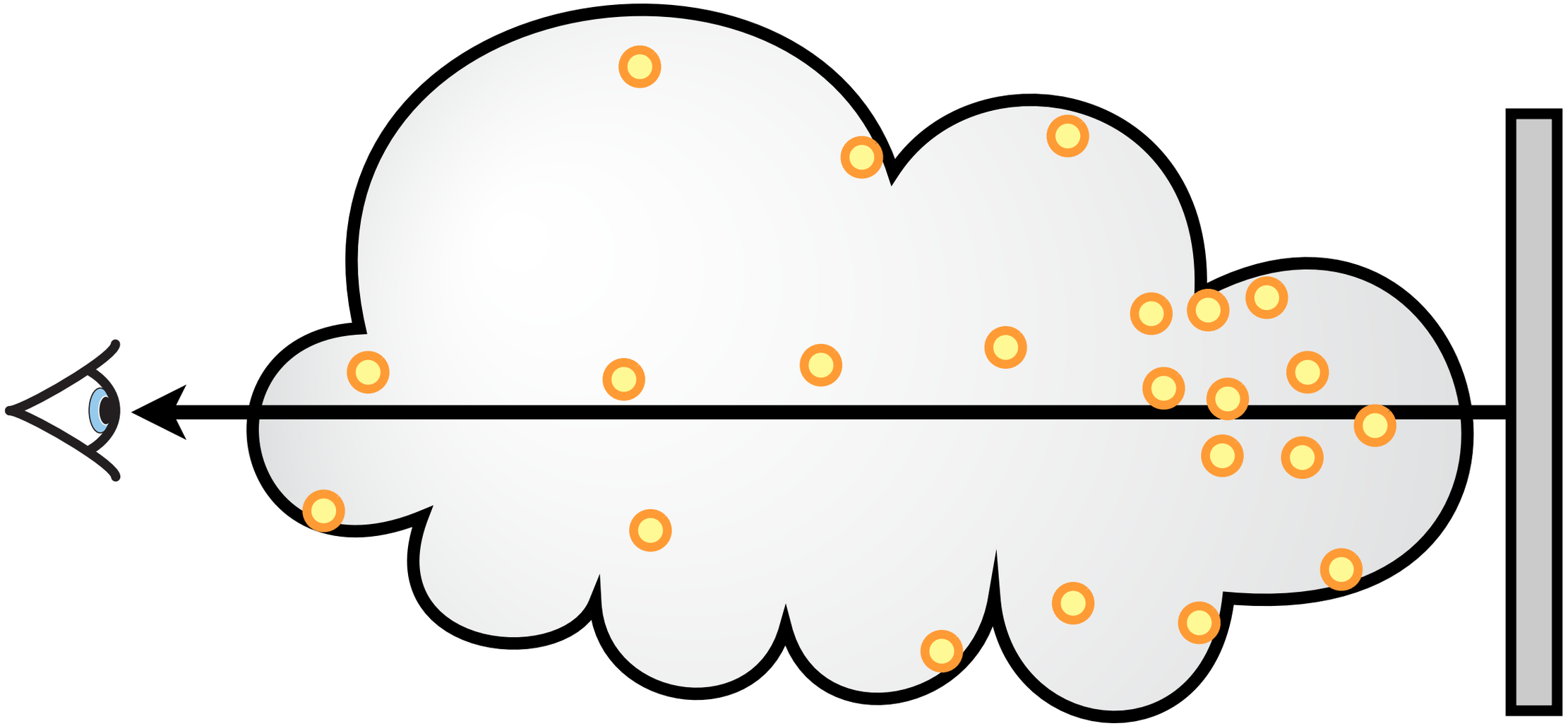
[Jensen & Christensen 98]



$$L_m(\mathbf{x}, \vec{\omega}) = \sum_{i=1}^{N} T_r(\mathbf{x}, \mathbf{x}_{t,i}) \sigma_s(\mathbf{x}_{t,i}) L_s(\mathbf{x}_{t,i}, \vec{\omega}) \Delta t$$

$$L_s(\mathbf{x}_t, \vec{\omega}) \approx \sum_{i=1}^{k} f_p(\mathbf{x}_t, \vec{\omega}_i, \vec{\omega}) \frac{\Phi_i(\mathbf{x}_t, \vec{\omega}_i)}{\frac{4}{3}\pi r(\mathbf{x}_t)^3}$$

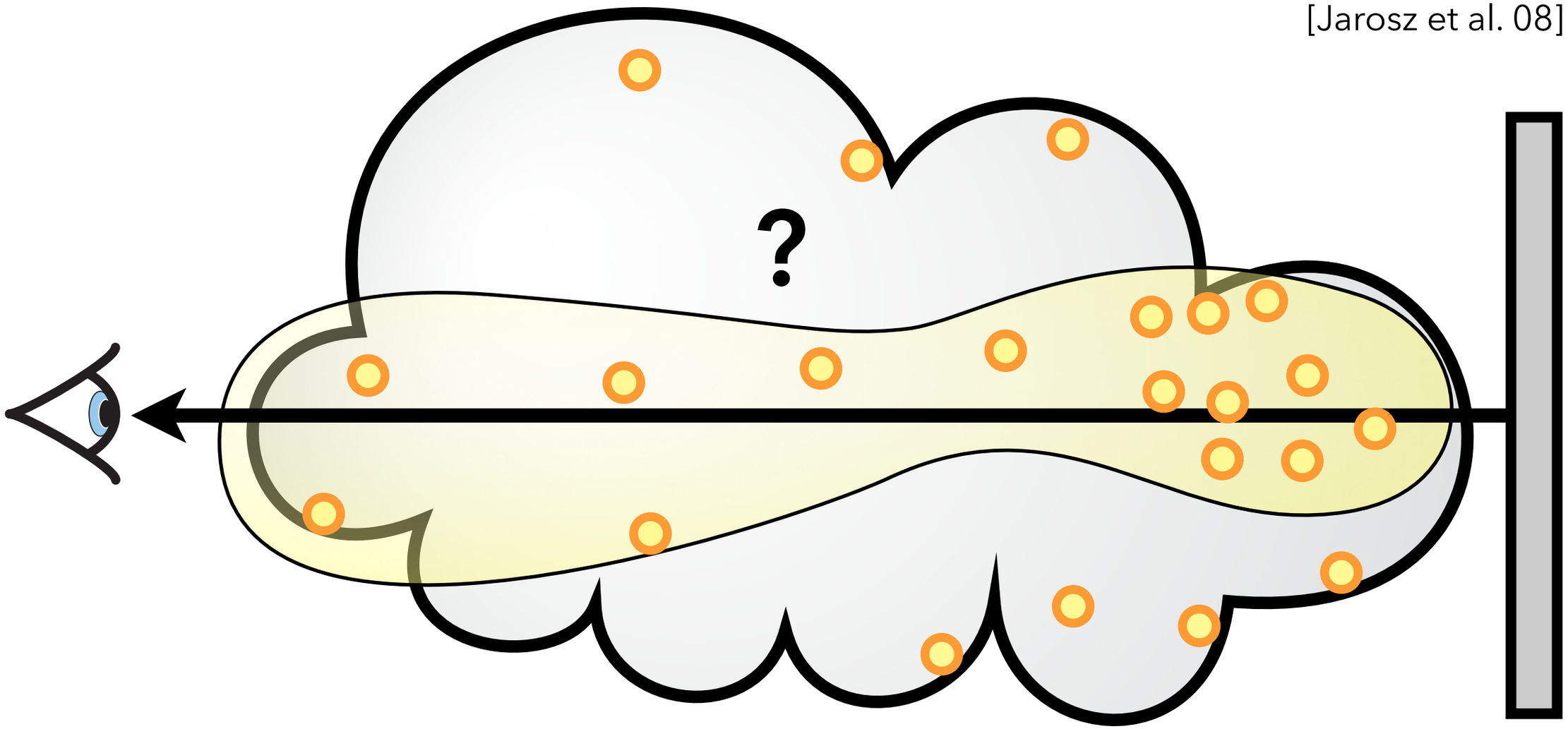
Rethinking the problem...



$$L_m(\mathbf{x}, \vec{\omega}) = \int_0^z T_r(\mathbf{x}, \mathbf{x}_t)\sigma_s(\mathbf{x}_t)L_s(\mathbf{x}_t, \vec{\omega})dt$$

Find all photons at once...

[Jarosz et al. 08]

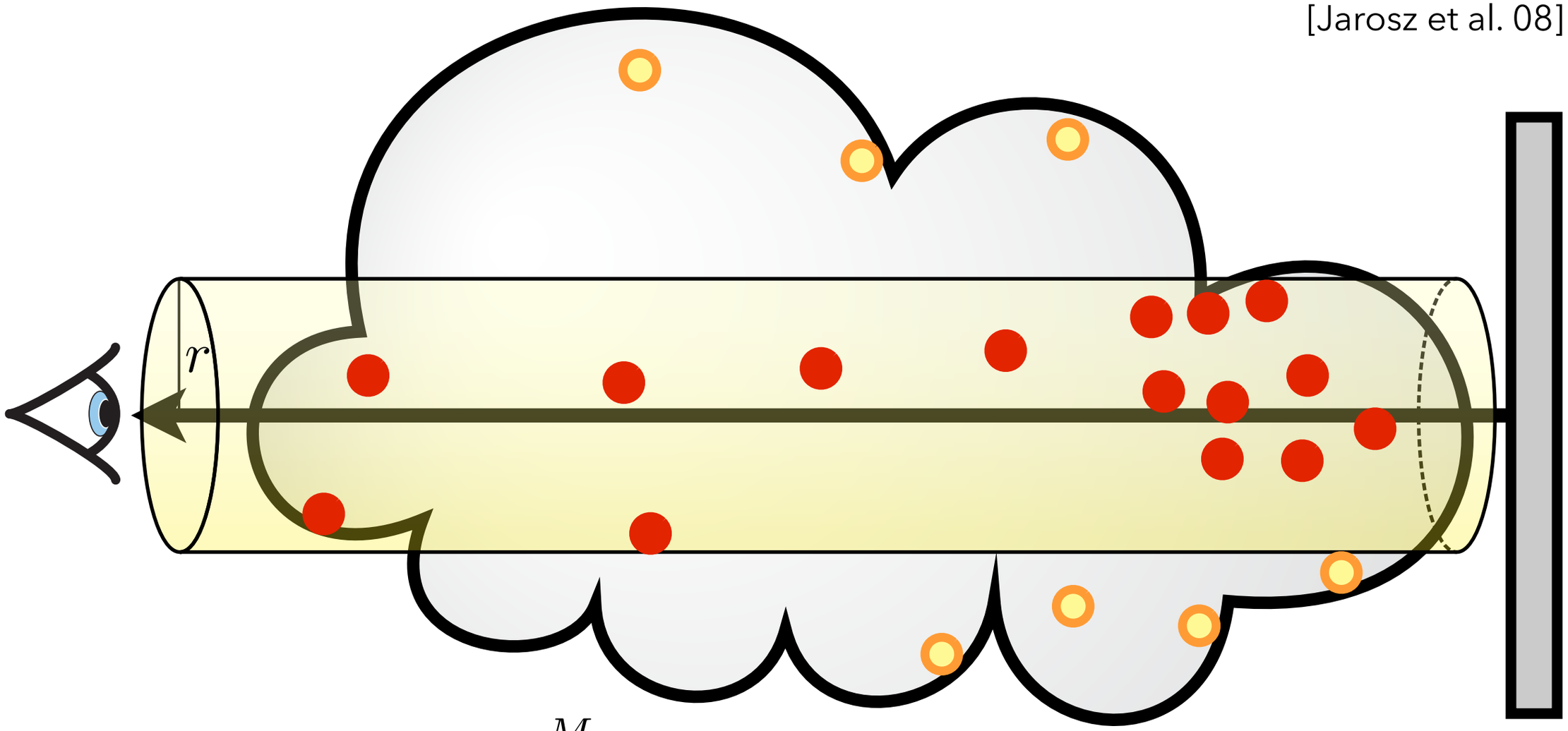


$$L_m(\mathbf{x}, \vec{\omega}) = \int_0^z T_r(\mathbf{x}, \mathbf{x}_t)\sigma_s(\mathbf{x}_t)L_s(\mathbf{x}_t, \vec{\omega})dt$$

How to find the photons?

Beam Radiance Estimate

[Jarosz et al. 08]



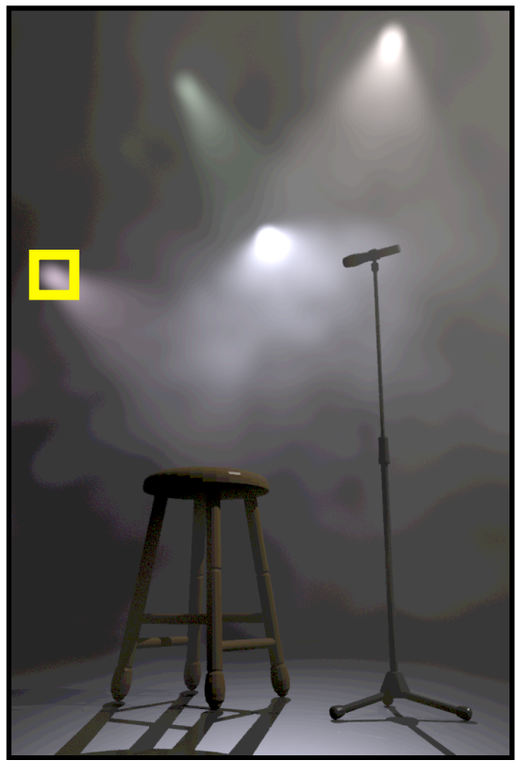
$$L_m(\mathbf{x}, \vec{\omega}) \approx \sum_{i=1}^{M} T_r(\mathbf{x}, \mathbf{x}_i)\sigma_s(\mathbf{x}_i)f_p(\mathbf{x}_i, \vec{\omega}_i, \vec{\omega})\frac{\Phi_i}{\pi r^2}$$

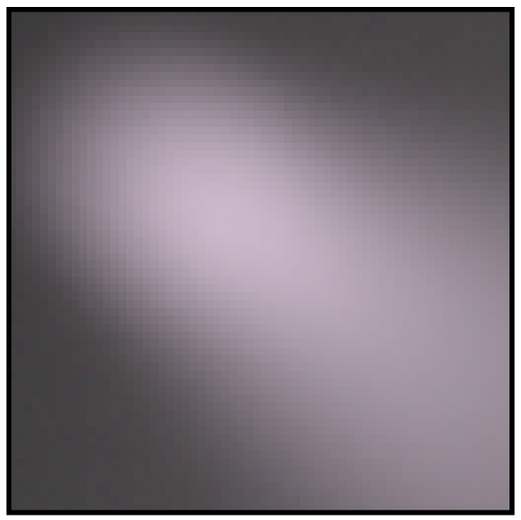
Fixed Radius Cylinder

$\mathbf{x}_i$: orthogonal projection of i^{th} photon onto the ray

Fixed Radius Comparison

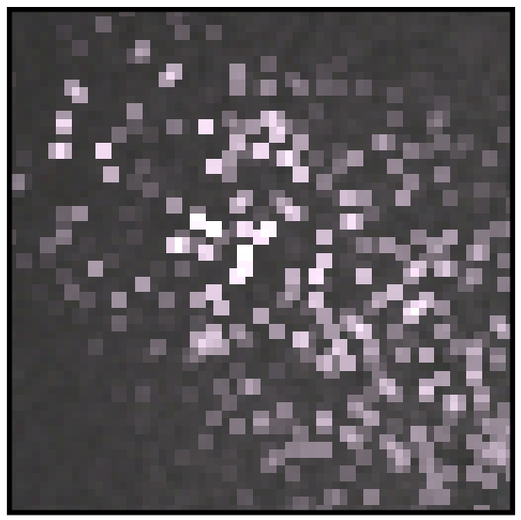
Trad. Estimate





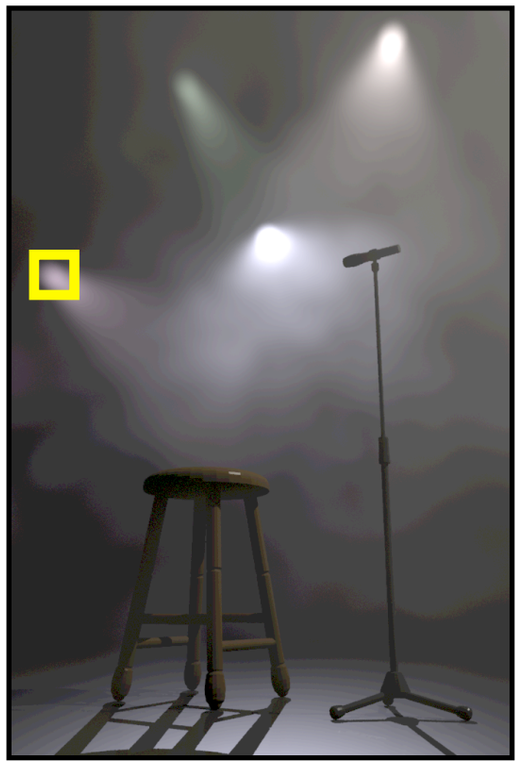
(∞)

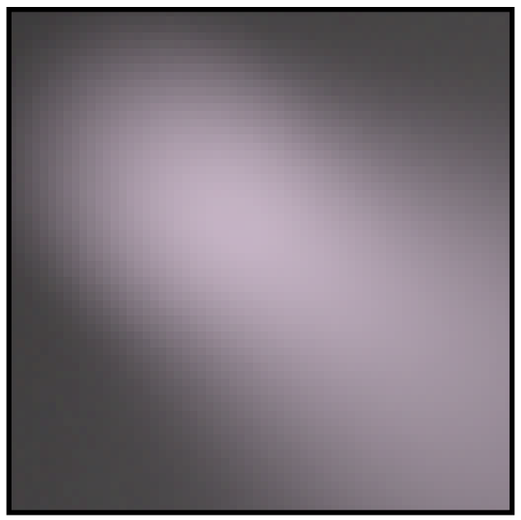
Trad. Estimate



(4:21)

Beam Estimate

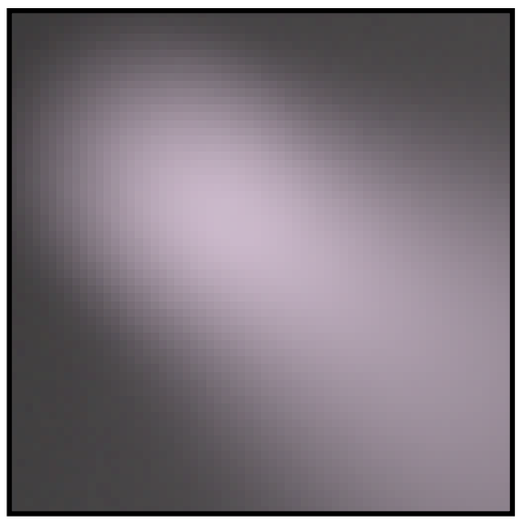




(4:15)

Volumetric Photon Mapping

Fixed Radius

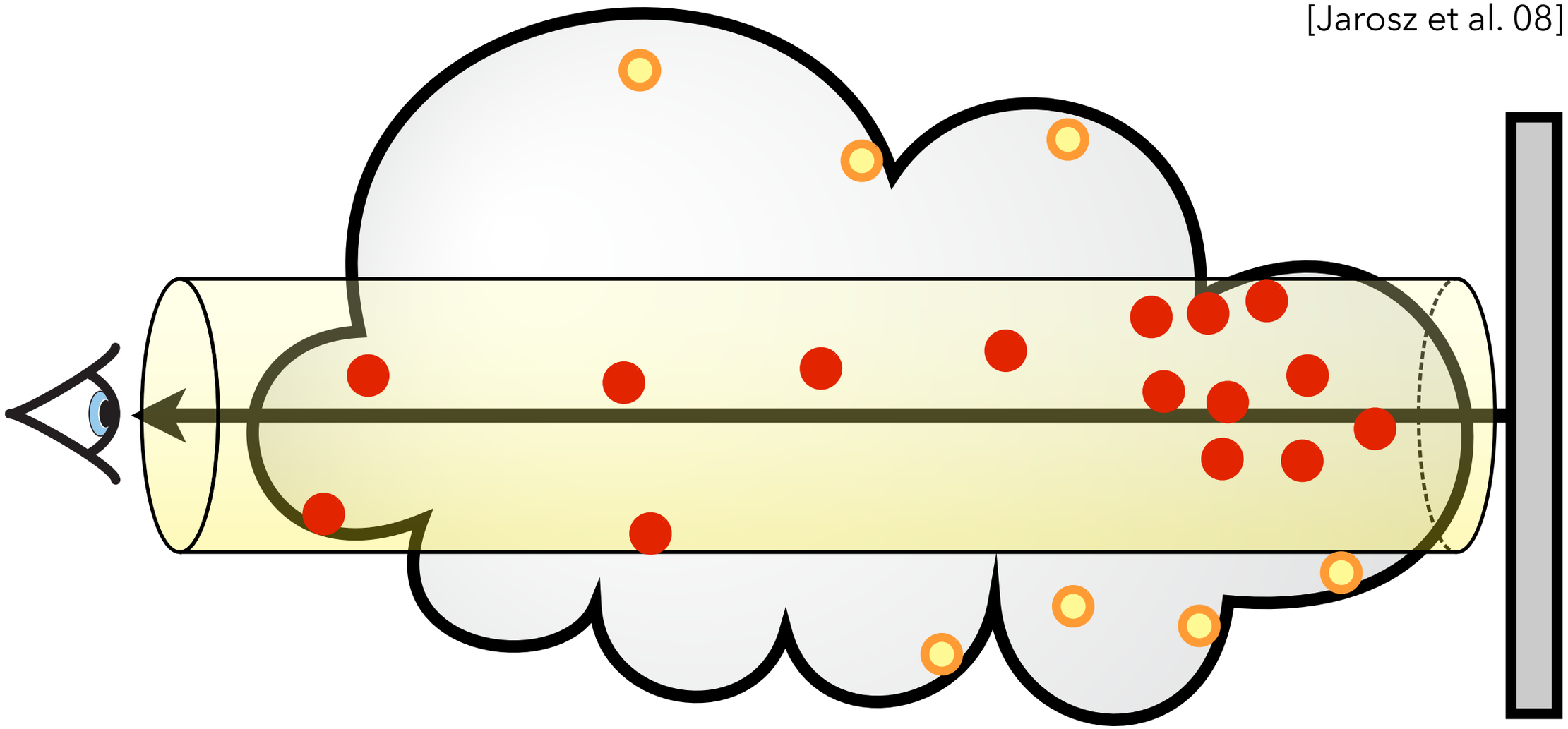


Nearest Neighbor

(Defining the kernel support by finding k nearest photons)

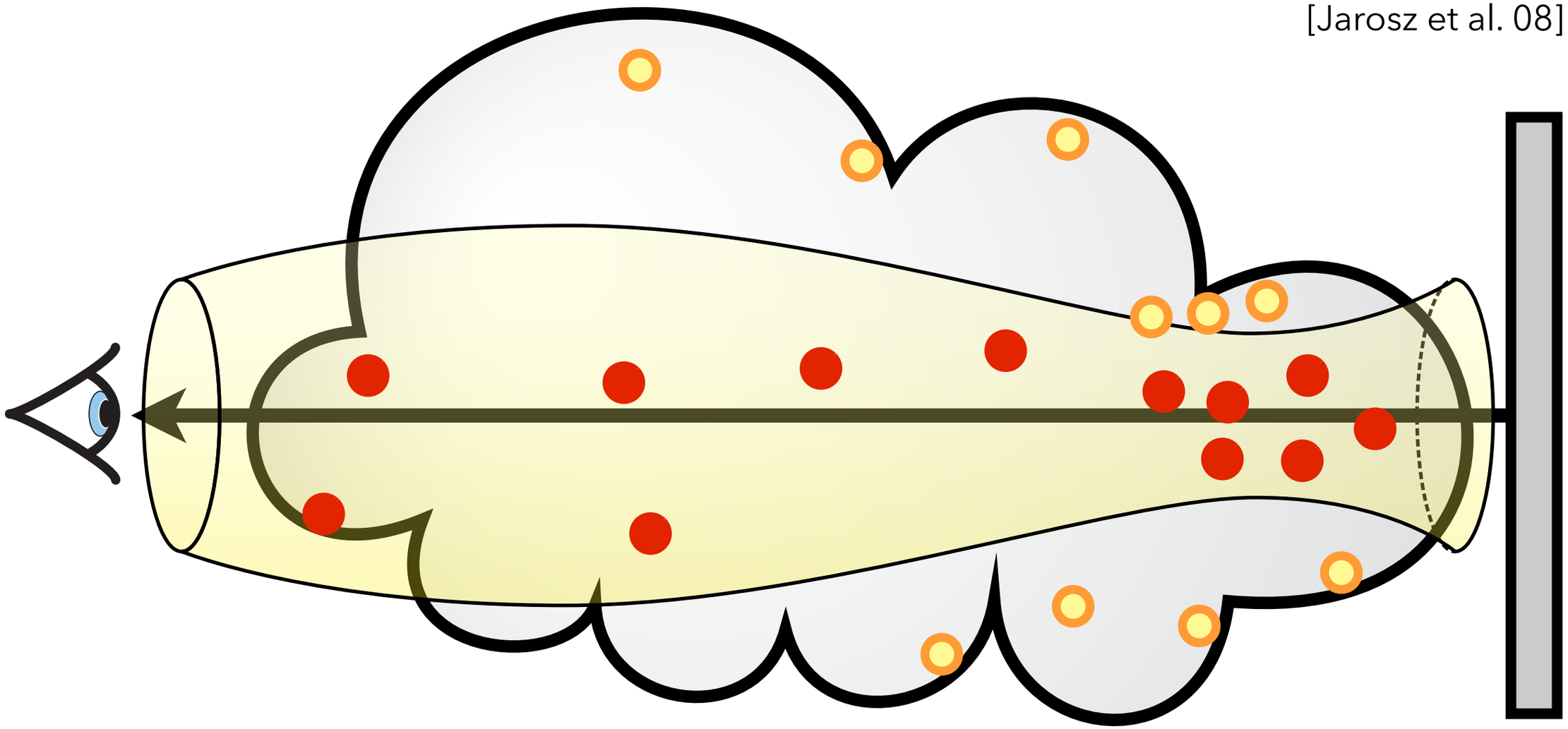
Fixed Radius

[Jarosz et al. 08]



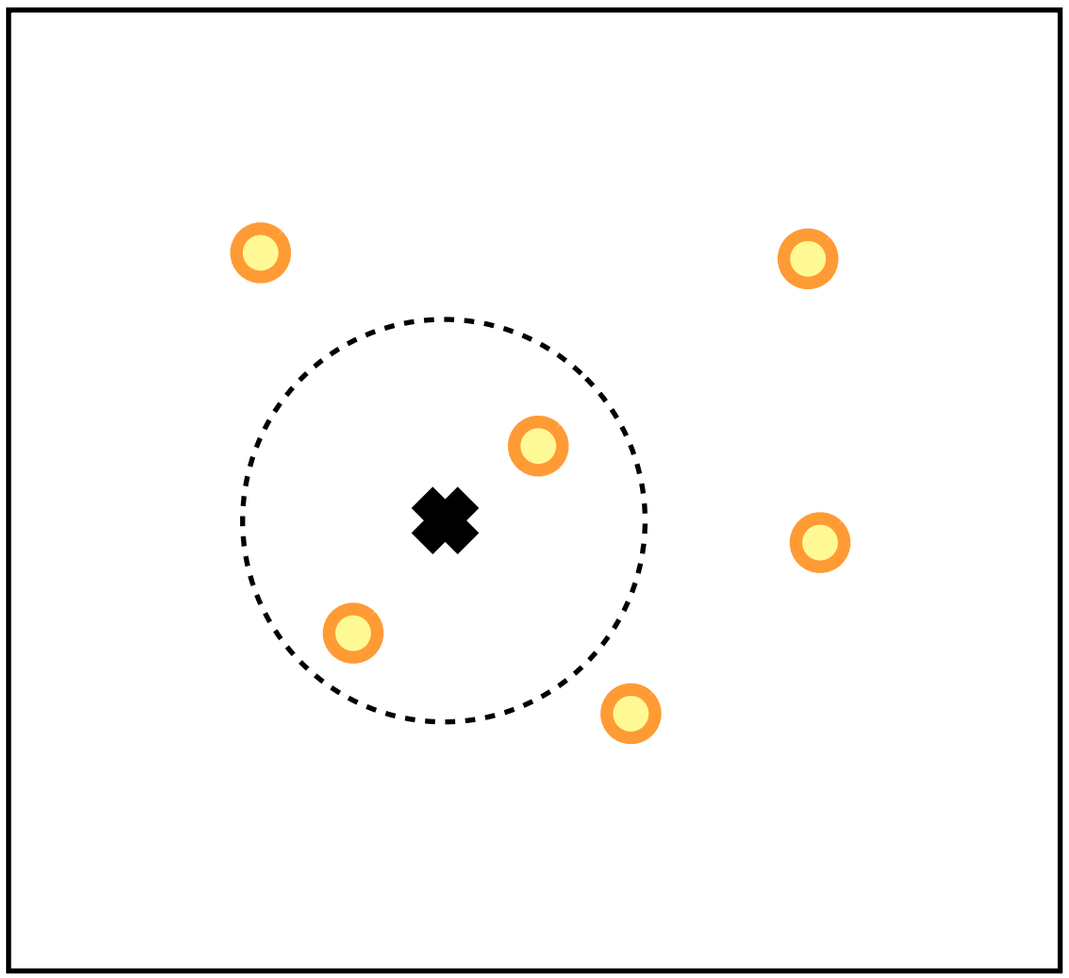
Varying Radius

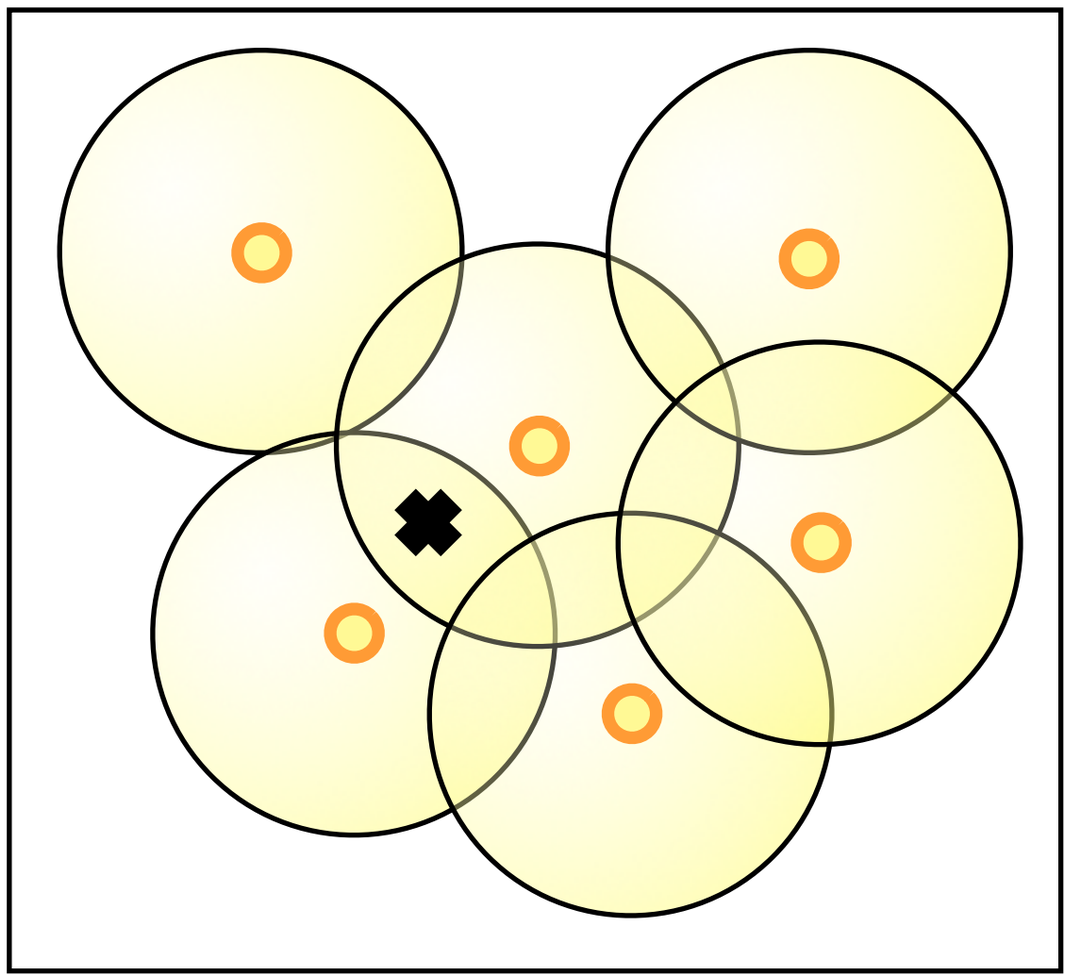
[Jarosz et al. 08]



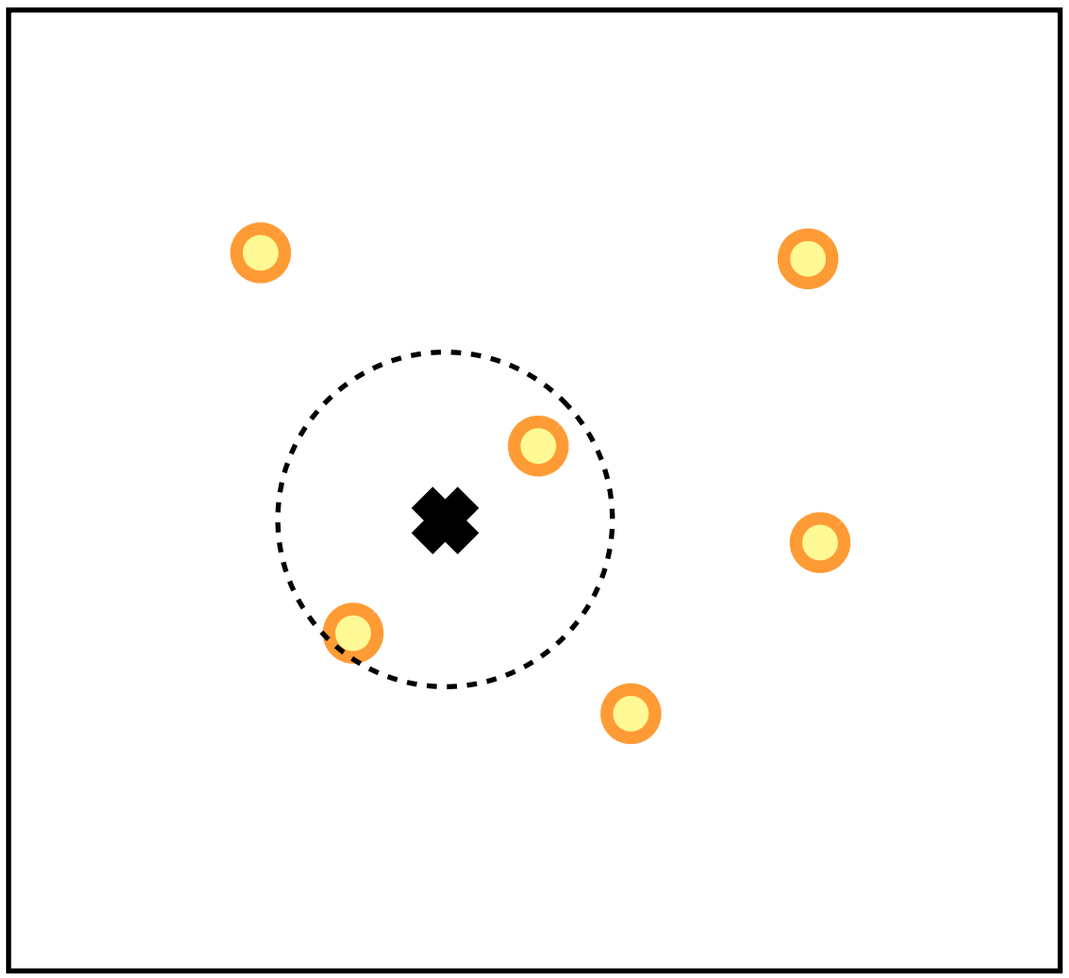
How to implement this efficiently?

Primal vs Dual

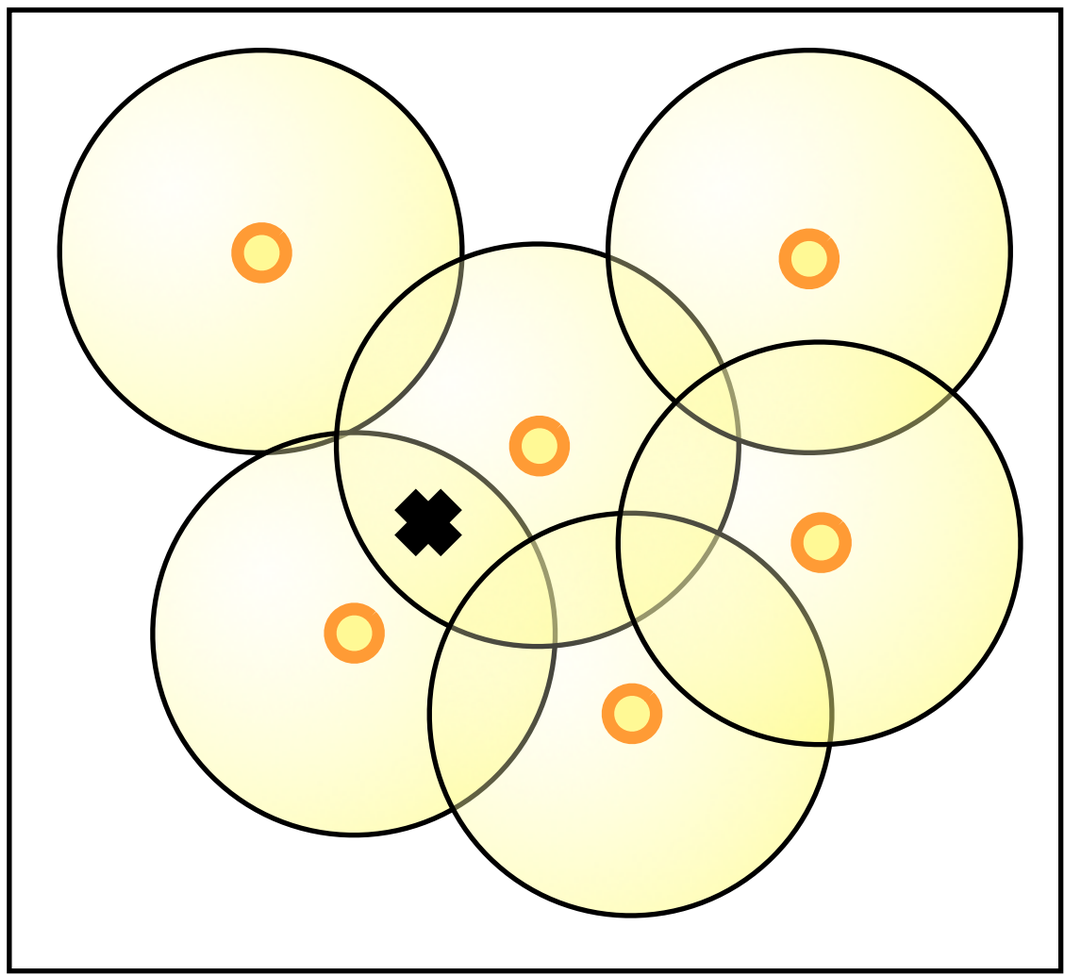




Primal vs Dual

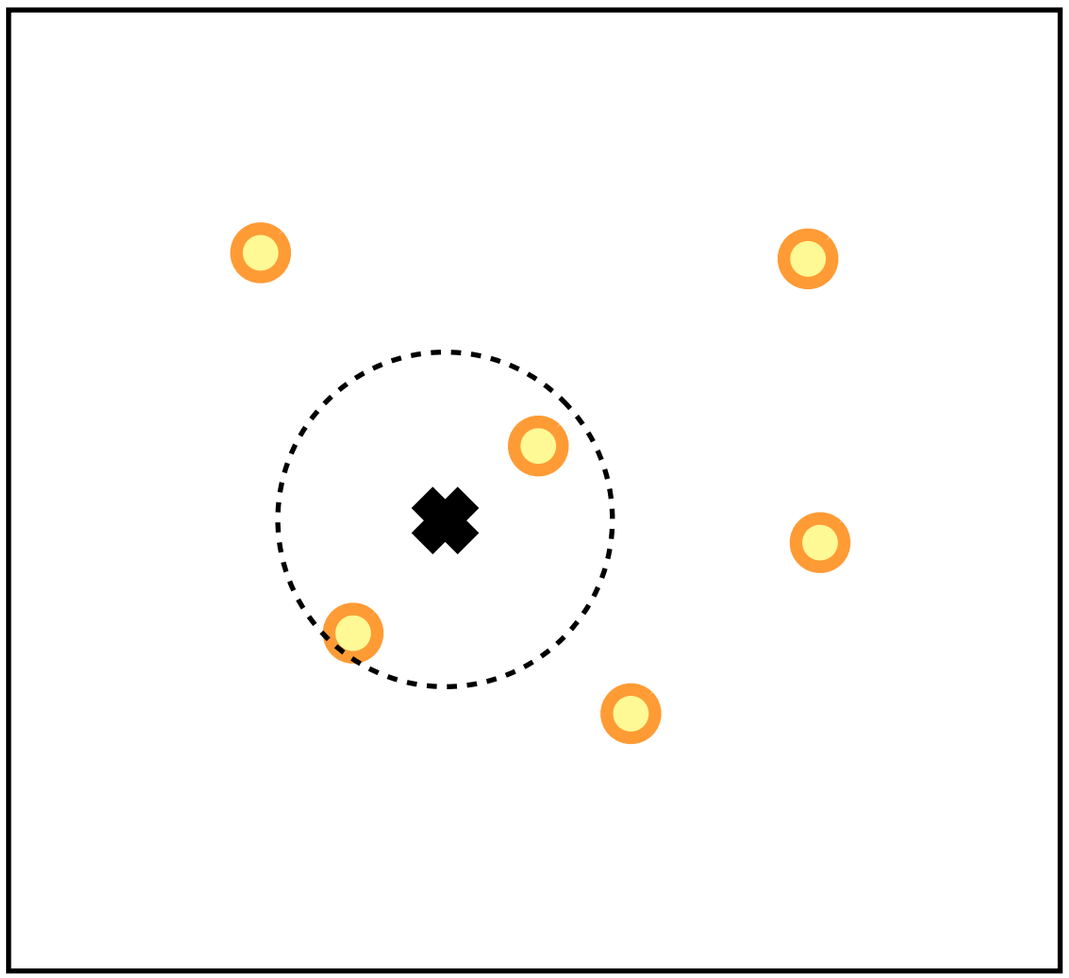


k-nearest neighbor

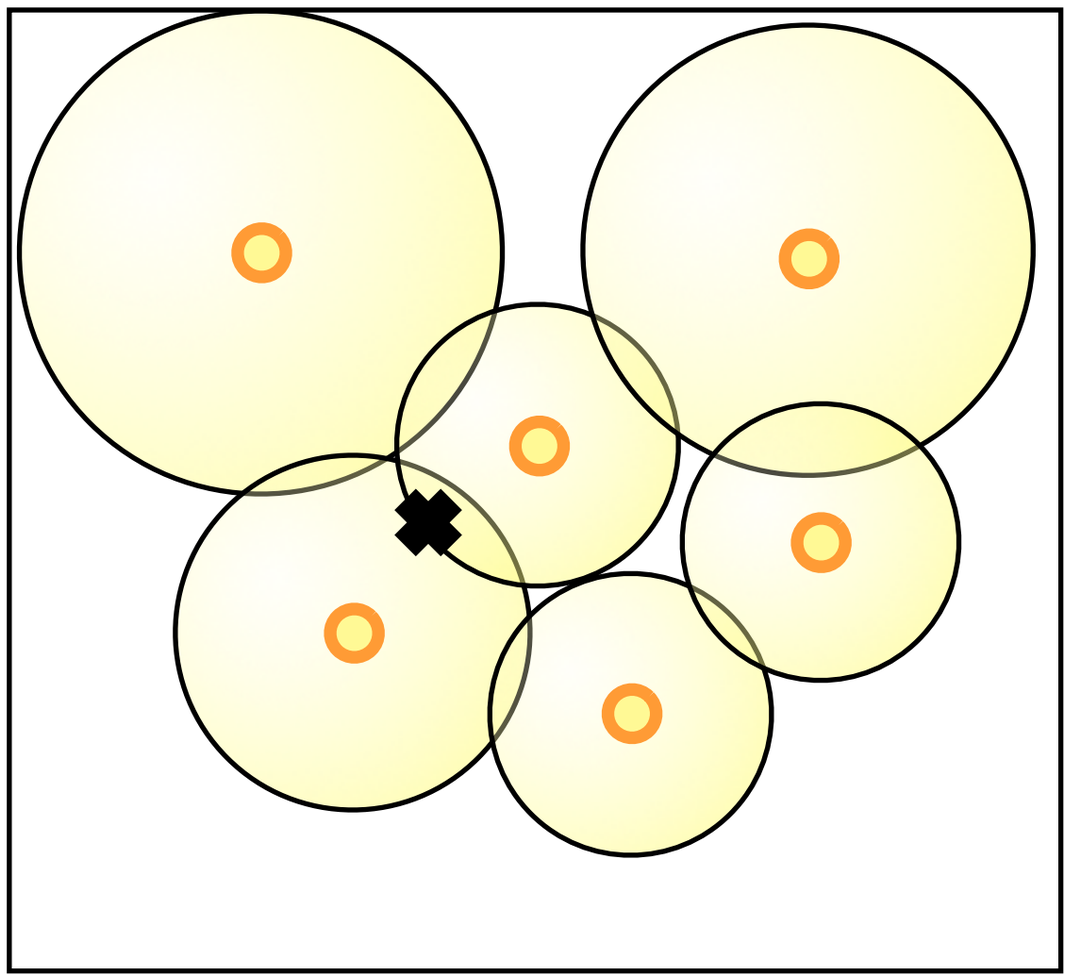


allow kernel radius to vary: adaptive kernel method

Primal vs Dual



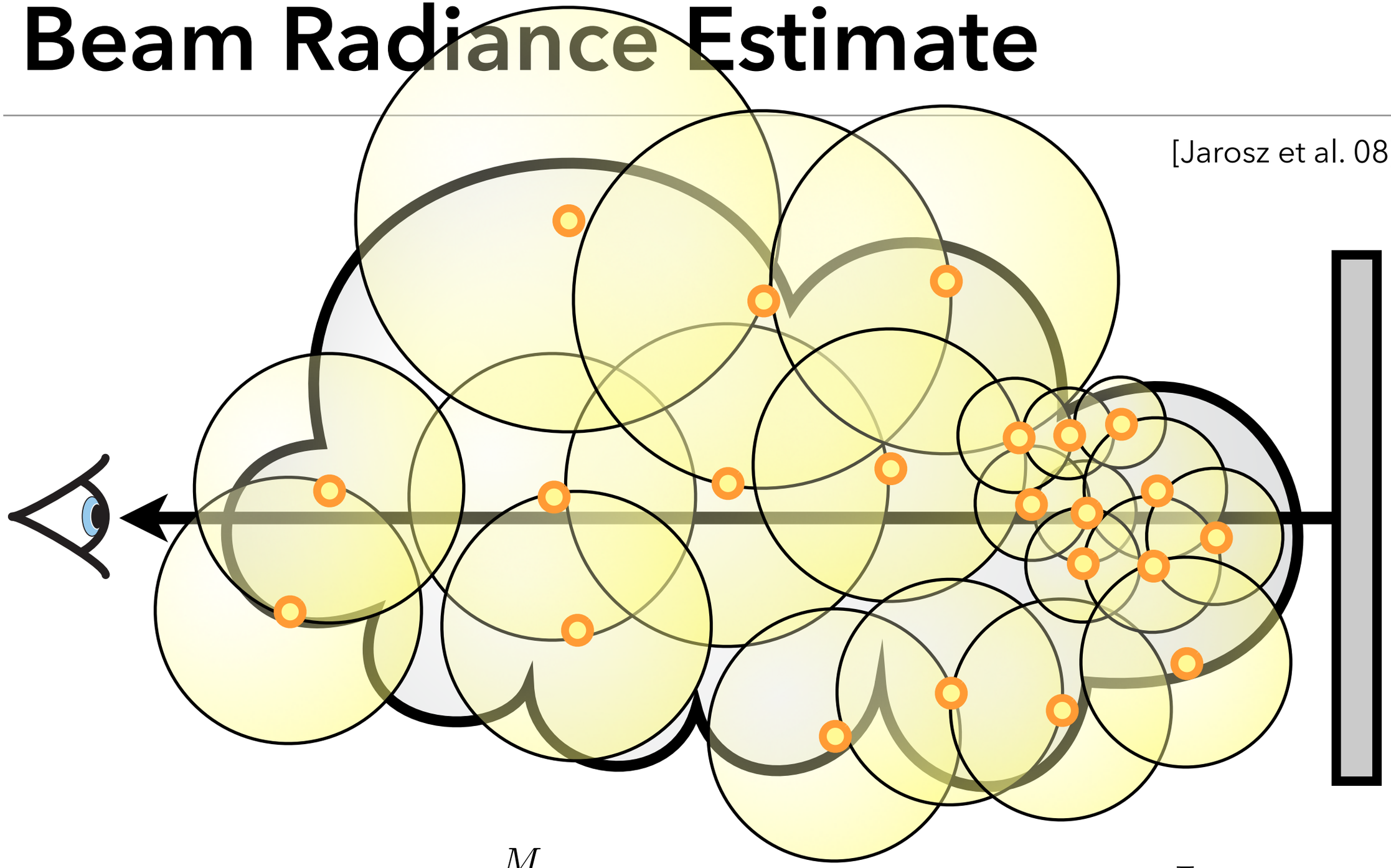
k-nearest neighbor



allow kernel radius to vary: adaptive kernel method

Beam Radiance Estimate

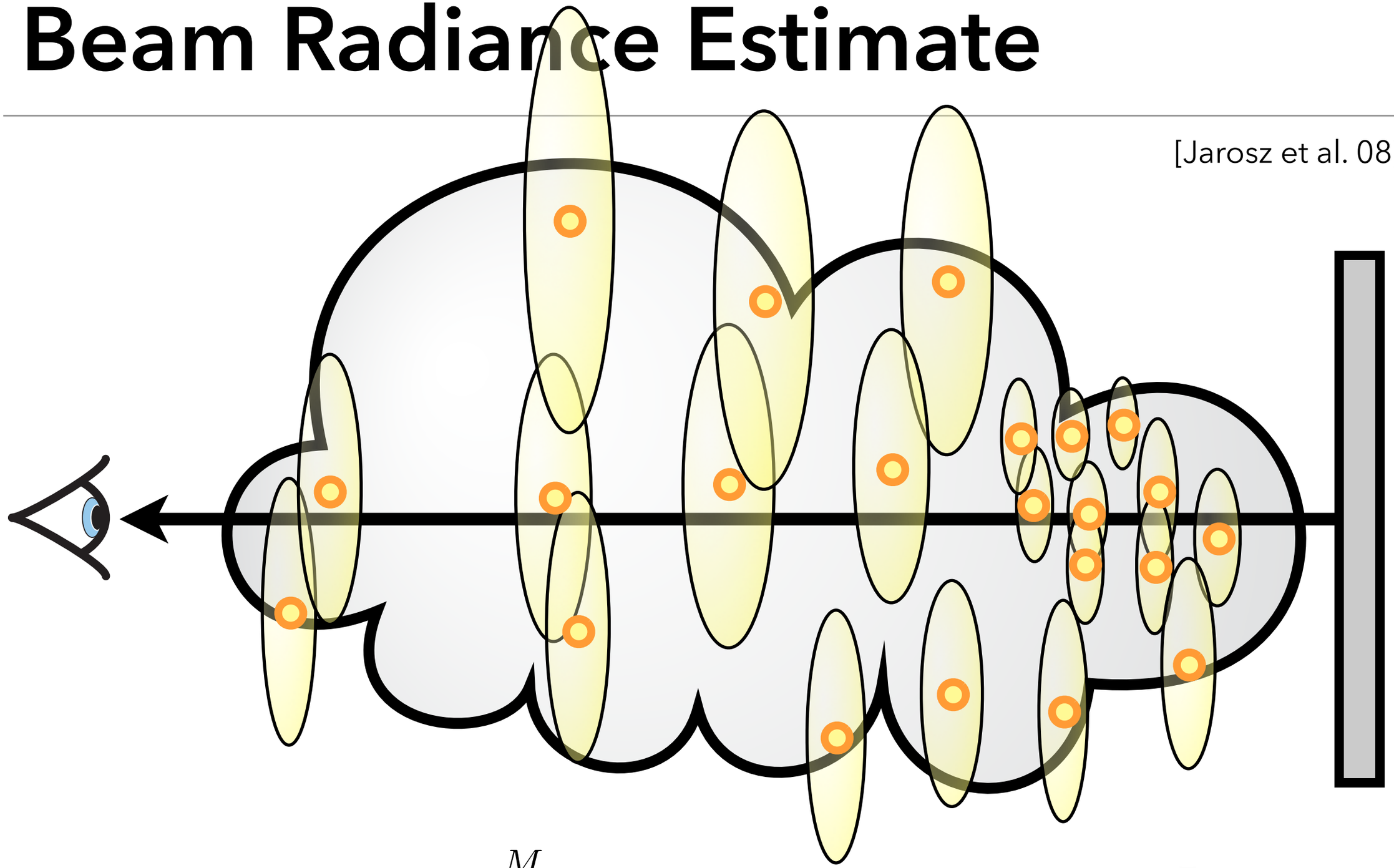
[Jarosz et al. 08]



$$L_m(\mathbf{x}, \vec{\omega}) \approx \sum_{i=1}^{M} T_r(\mathbf{x}, \mathbf{x}_i) \sigma_s(\mathbf{x}_i) f_p(\mathbf{x}_i, \vec{\omega}_i, \vec{\omega}) \frac{\Phi_i}{\pi r_i^2}$$

Beam Radiance Estimate

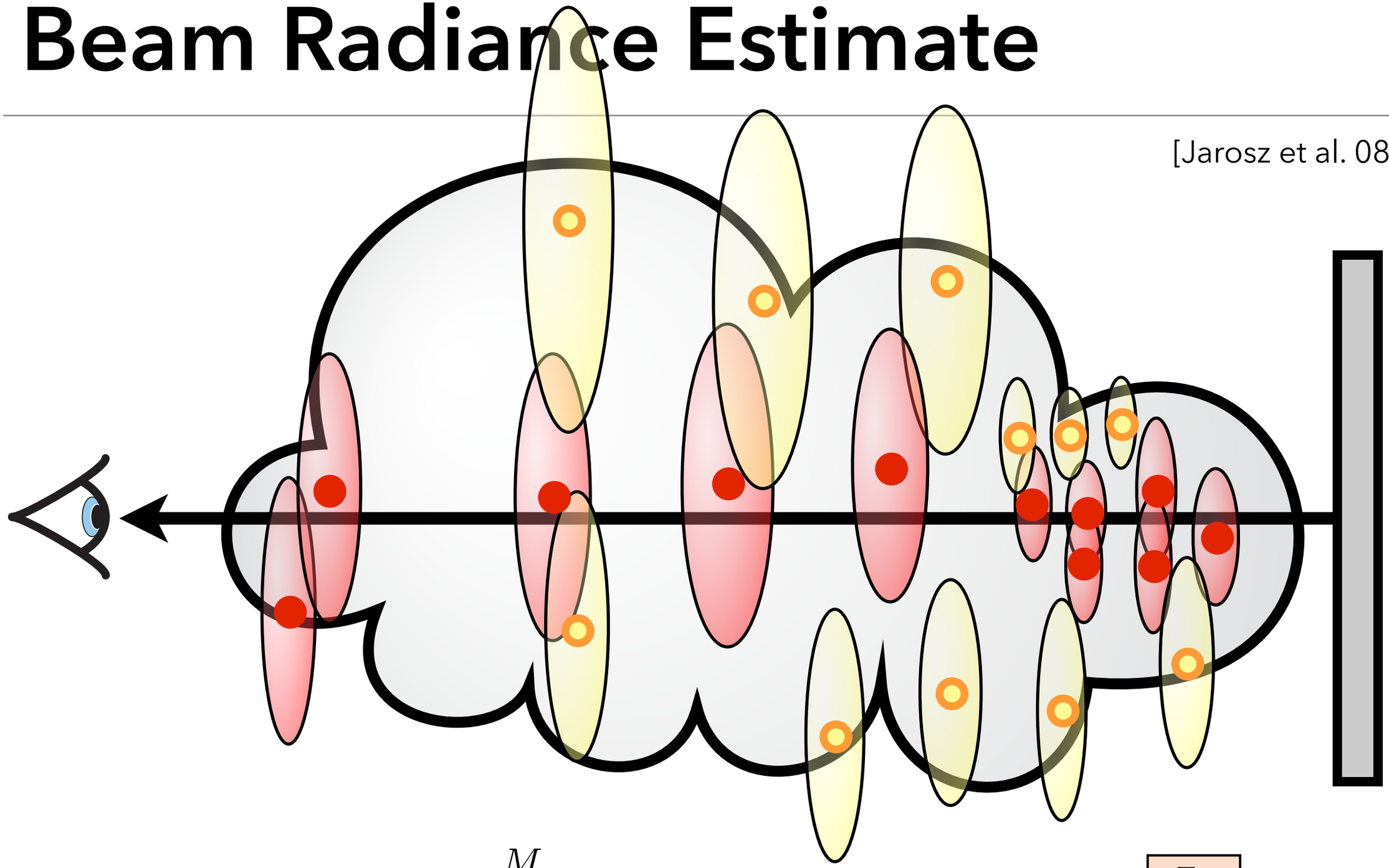
[Jarosz et al. 08]



$$L_m(\mathbf{x}, \vec{\omega}) \approx \sum_{i=1}^{M} T_r(\mathbf{x}, \mathbf{x}_i) \sigma_s(\mathbf{x}_i) f_p(\mathbf{x}_i, \vec{\omega}_i, \vec{\omega}) \frac{\Phi_i}{\pi r_i^2}$$

Beam Radiance Estimate

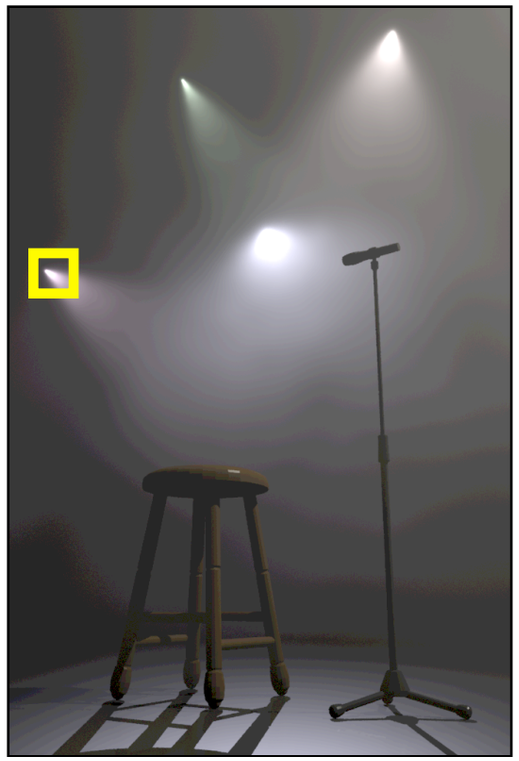
[Jarosz et al. 08]



$$L_m(\mathbf{x}, \vec{\omega}) \approx \sum_{i=1}^{M} T_r(\mathbf{x}, \mathbf{x}_i) \sigma_s(\mathbf{x}_i) f_p(\mathbf{x}_i, \vec{\omega}_i, \vec{\omega}) \boxed{\frac{\Phi_i}{\pi r_i^2}}$$

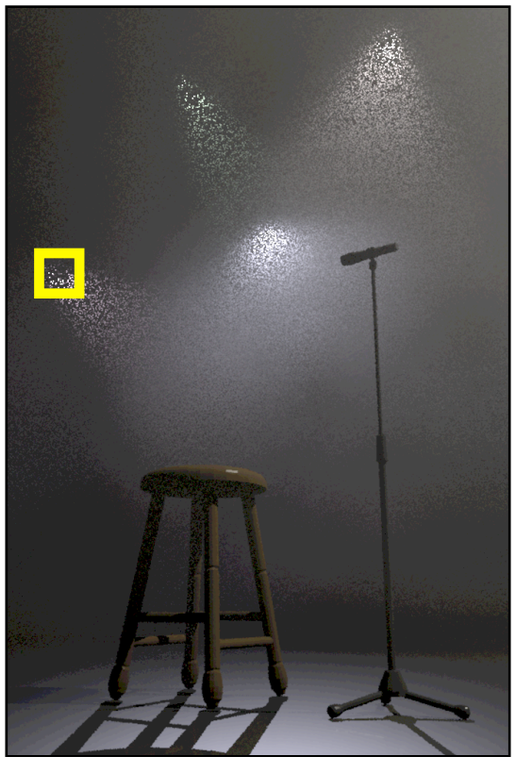
Adaptive Radius Comparison

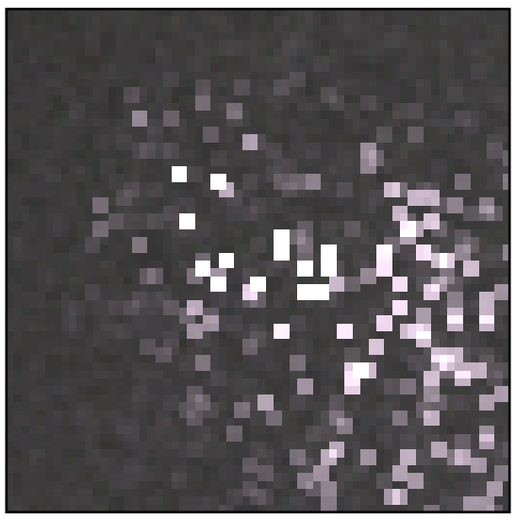
Trad. Estimate



(∞)

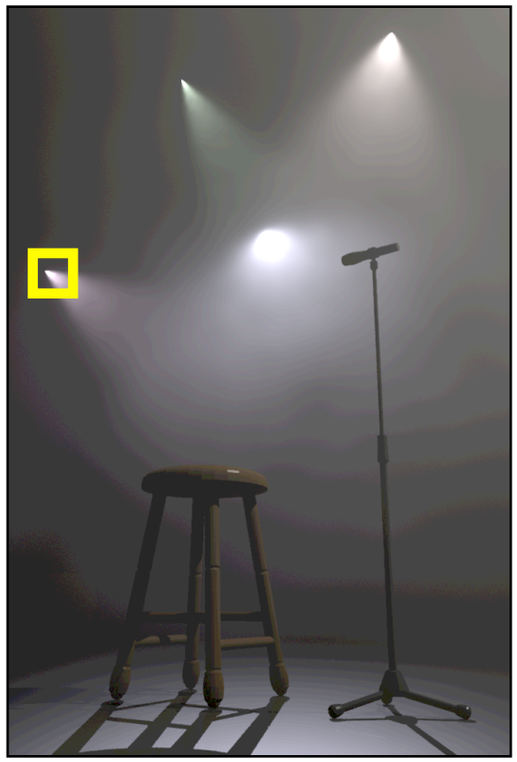
Trad. Estimate

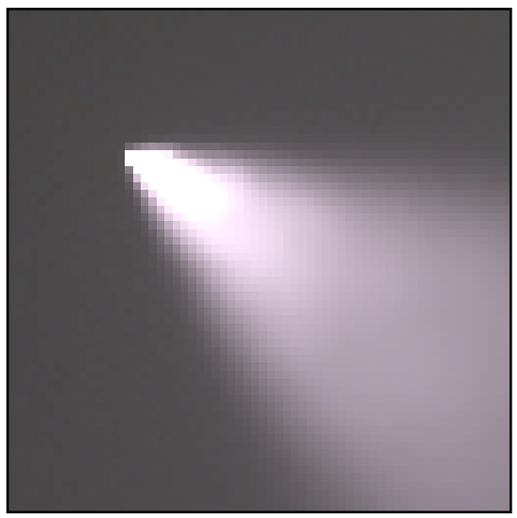




(6:38)

Beam Estimate





(6:22)

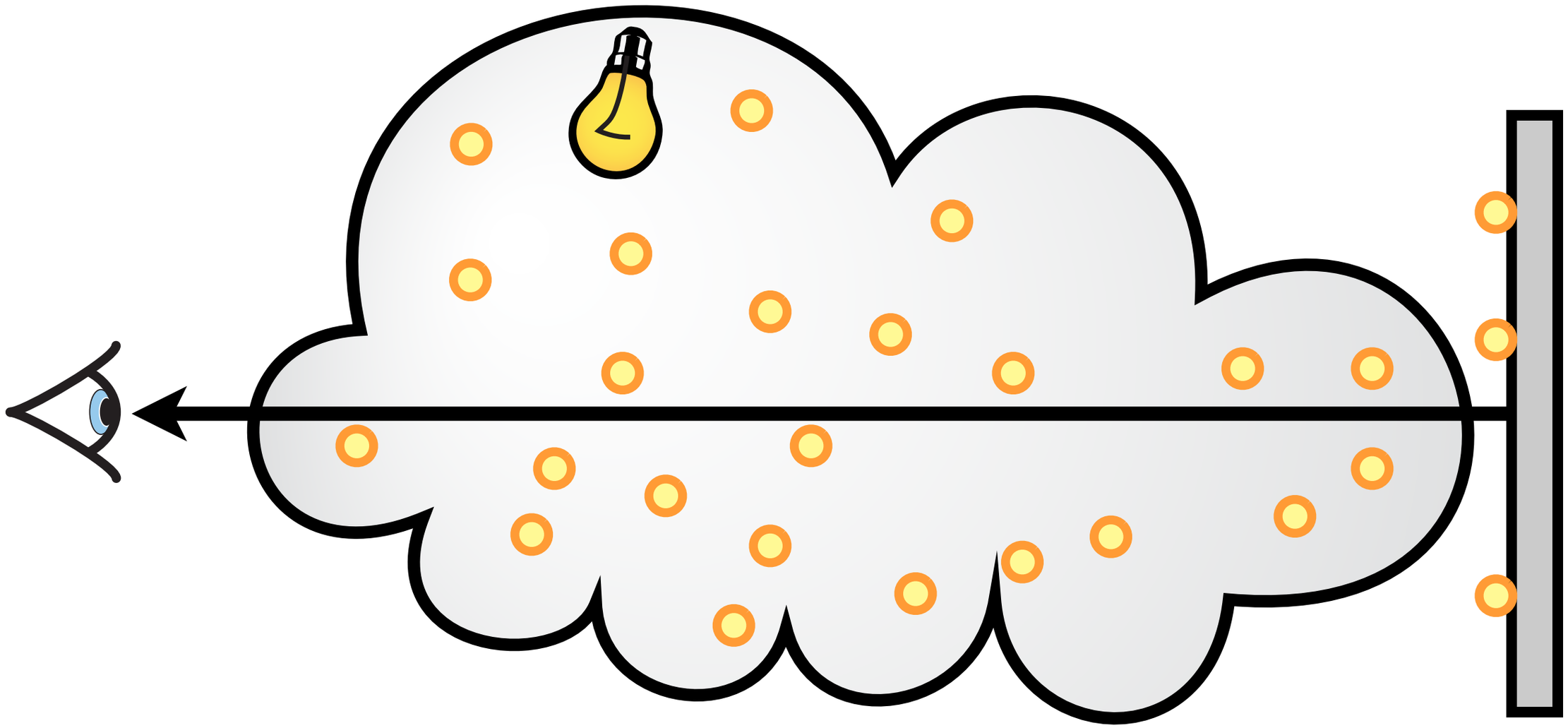
Algorithm

Same as Regular Photon Mapping

1. Shoot photons from light sources
2. Construct a balanced kD-tree for the photons
3. Assign a radius for each photon (photon-discs)
4. Create acceleration structure over photon-discs
5. Render:
   - For each ray through the medium, accumulate all photon-discs that intersect ray

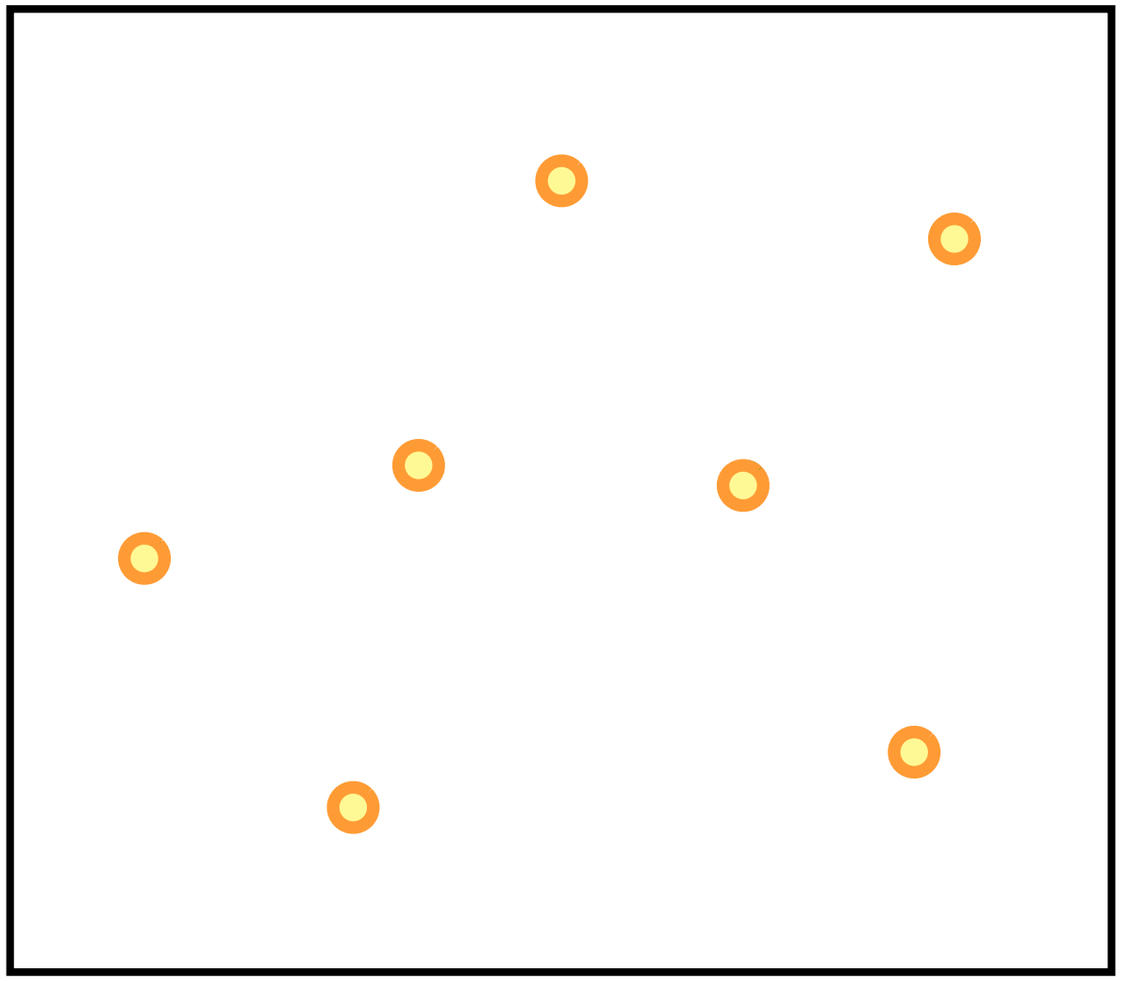
Beam Density Estimation

Algorithm



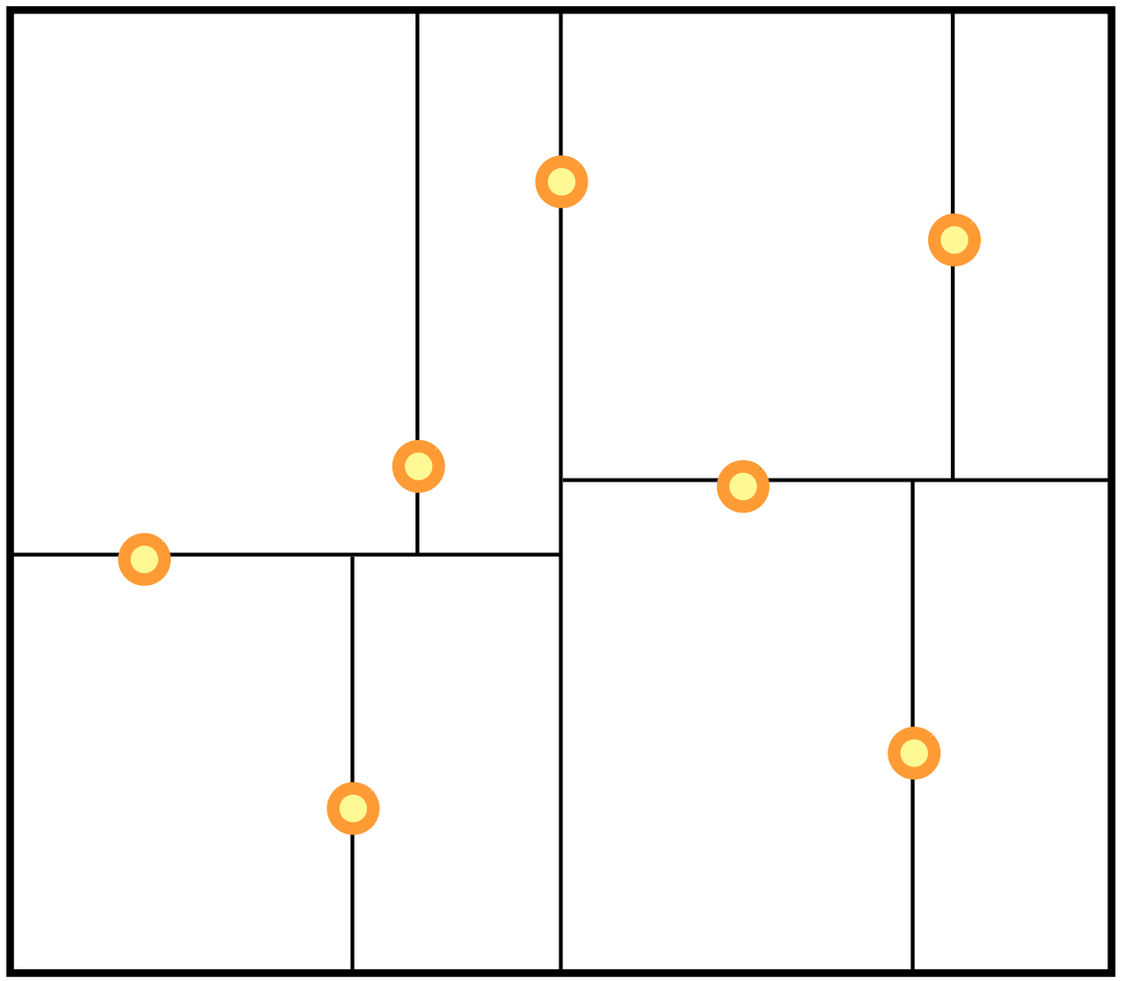
1) Shoot photons from light sources.

Algorithm



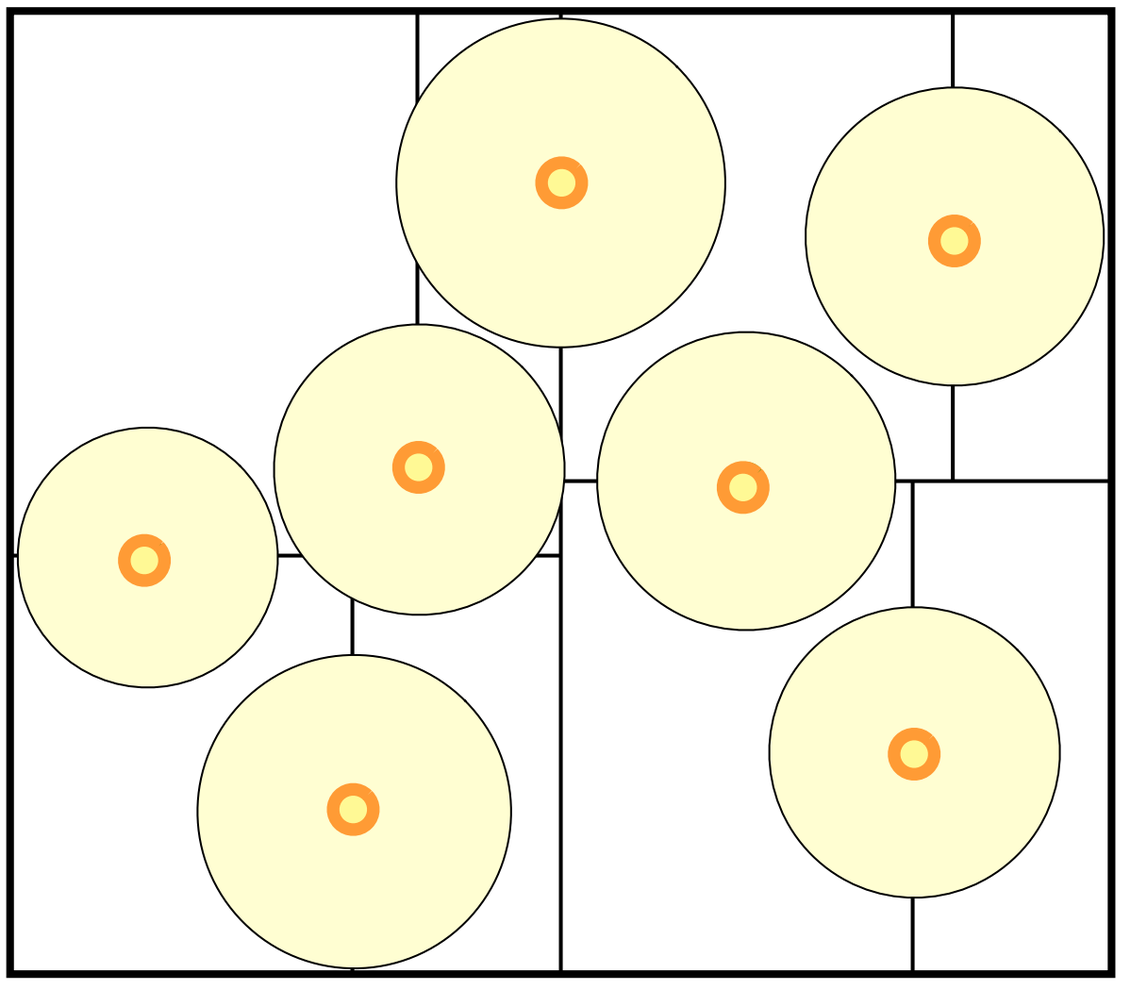
2) Construct a balanced kD-tree for the photons.

Algorithm



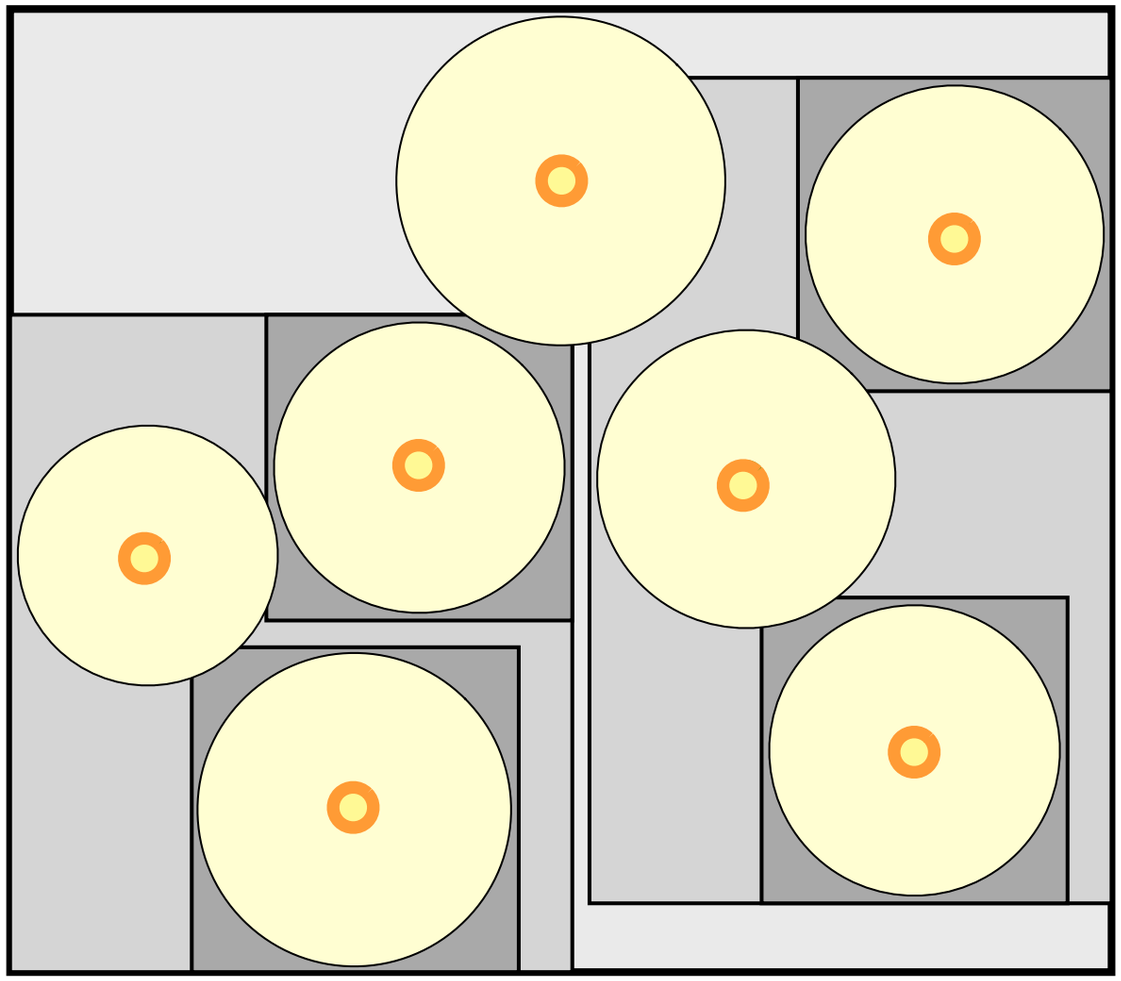
2) Construct a balanced kD-tree for the photons.

Algorithm



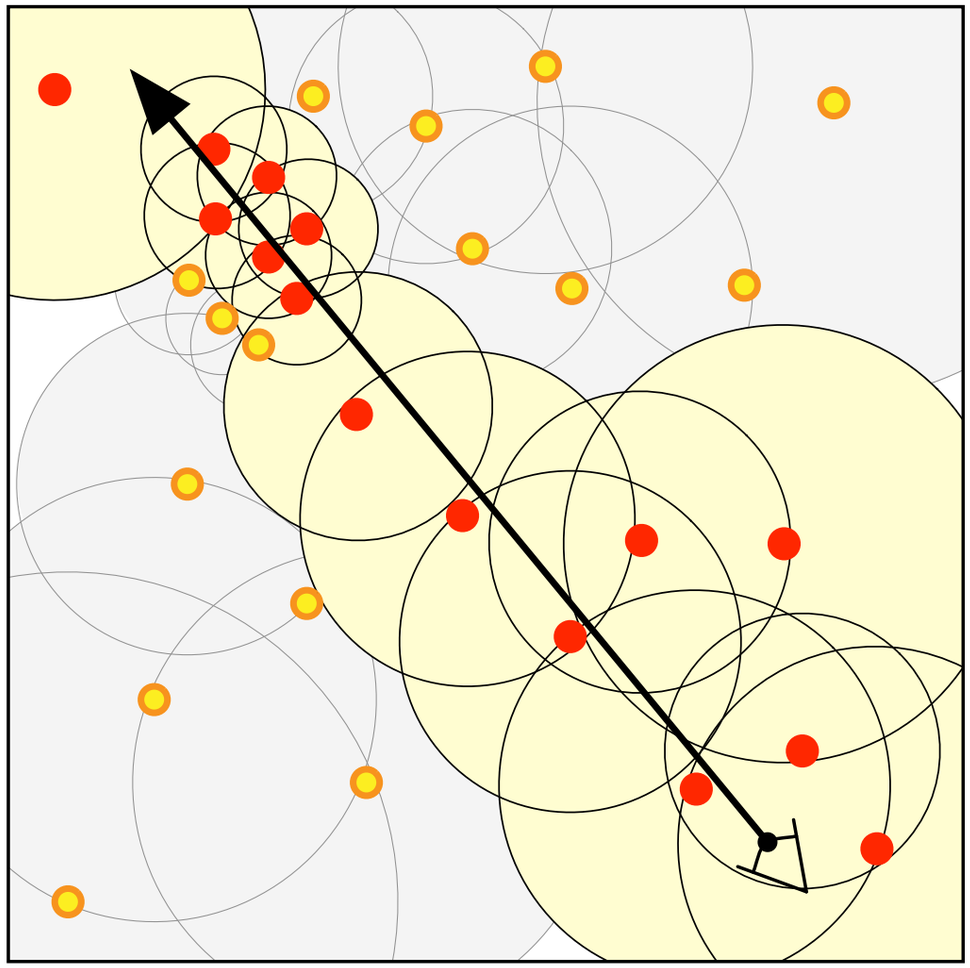
3) Assign a radius for each photon computed from the local density of photons

Algorithm



4) Create a bounding-box hierarchy over photon discs/spheres

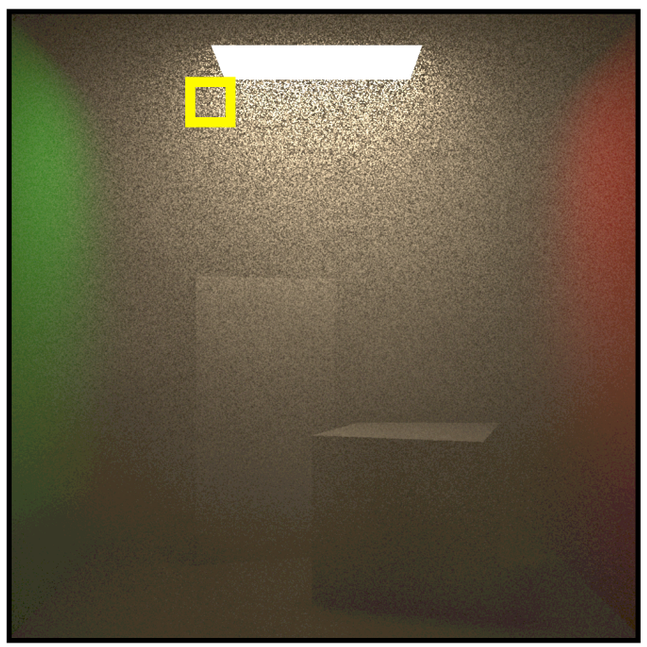
Algorithm

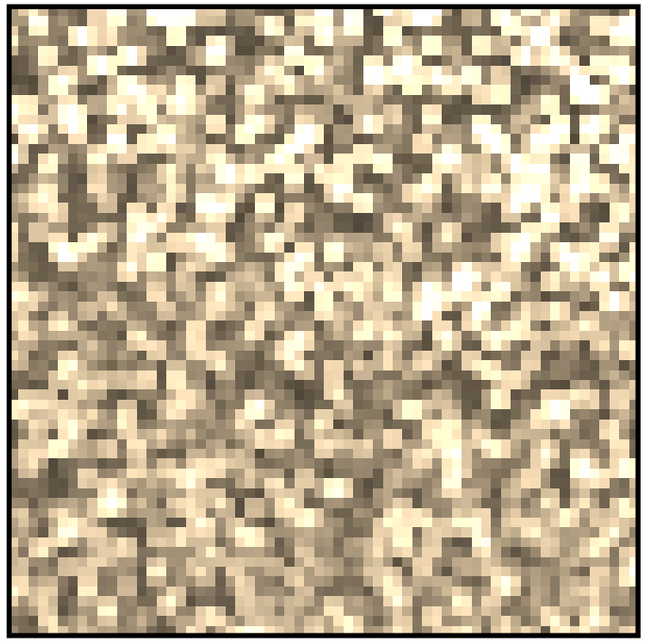


5) Render: For each ray through the medium, accumulate all photon discs that intersect ray.

Smoky Cornell Box

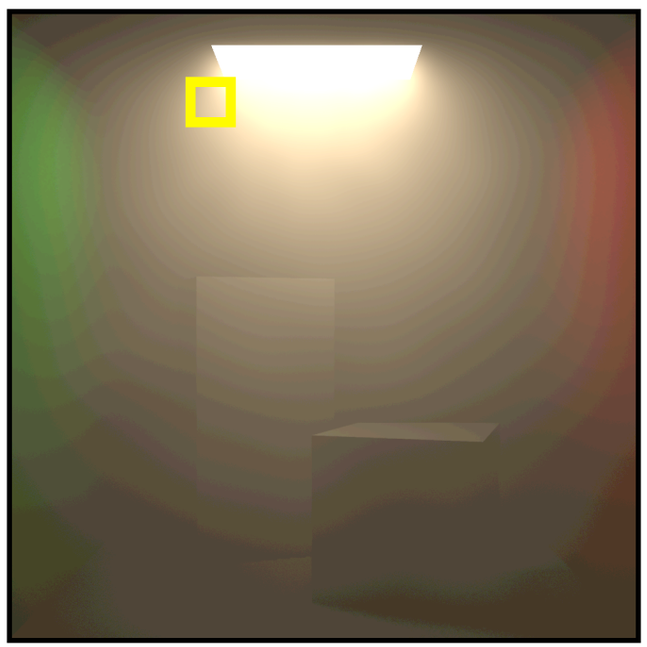
Trad. Estimate

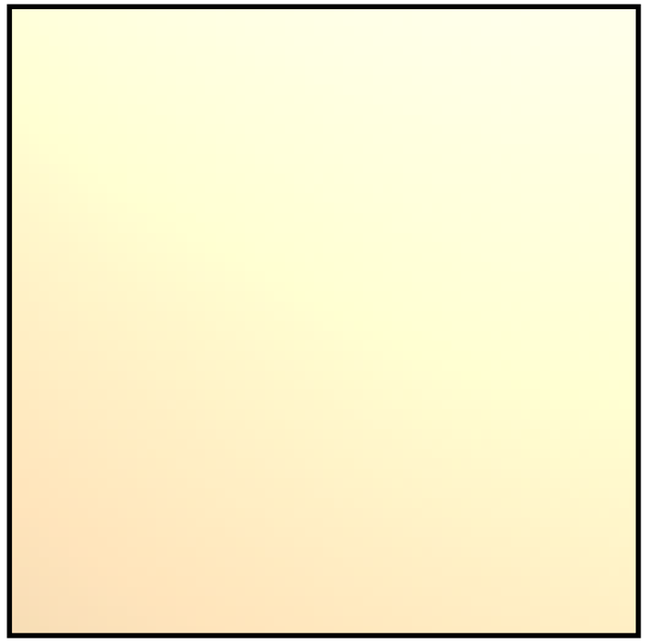




(4:03)

Beam Estimate



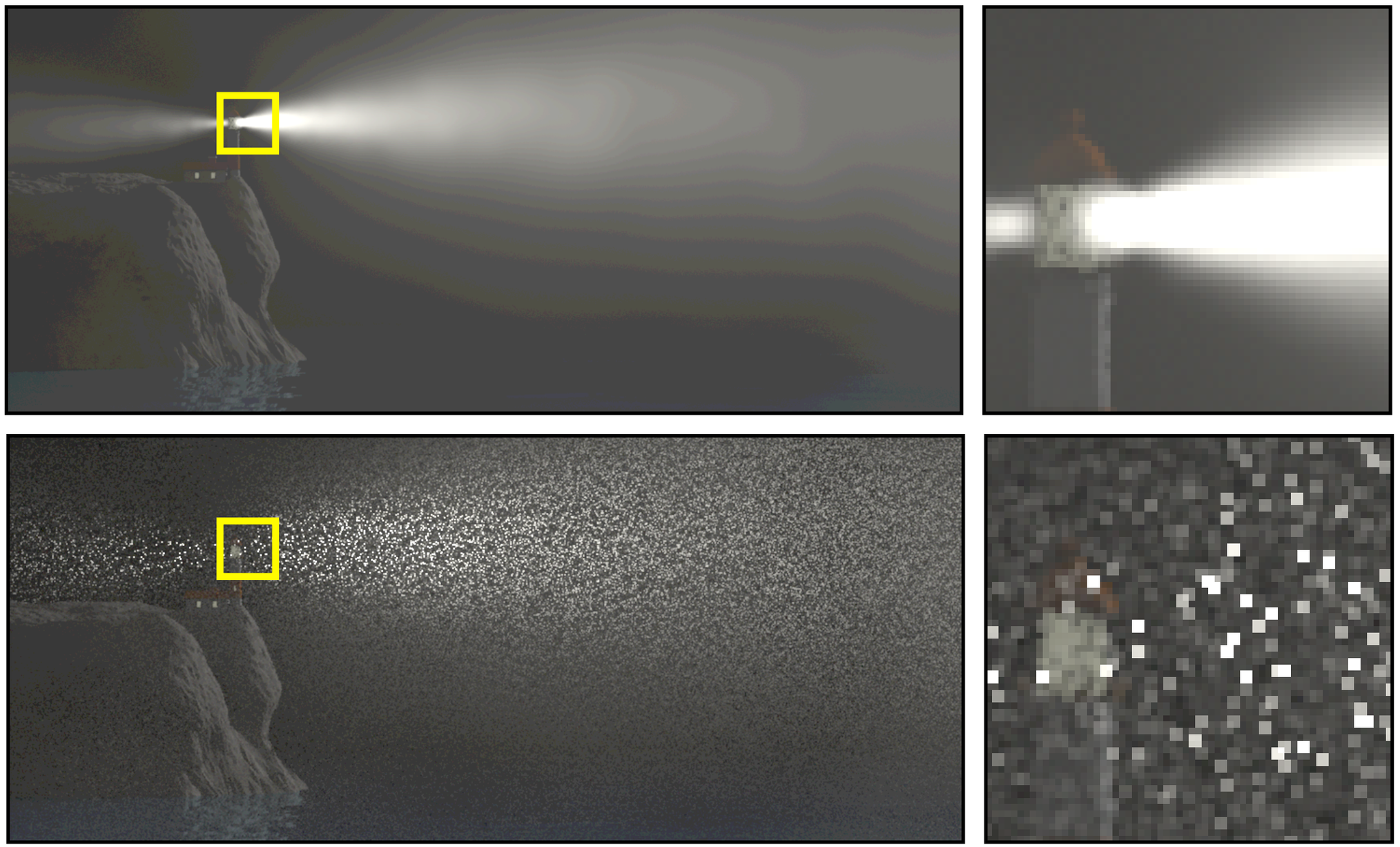


(3:35)

Lighthouse

Beam Estimate

(1:05)



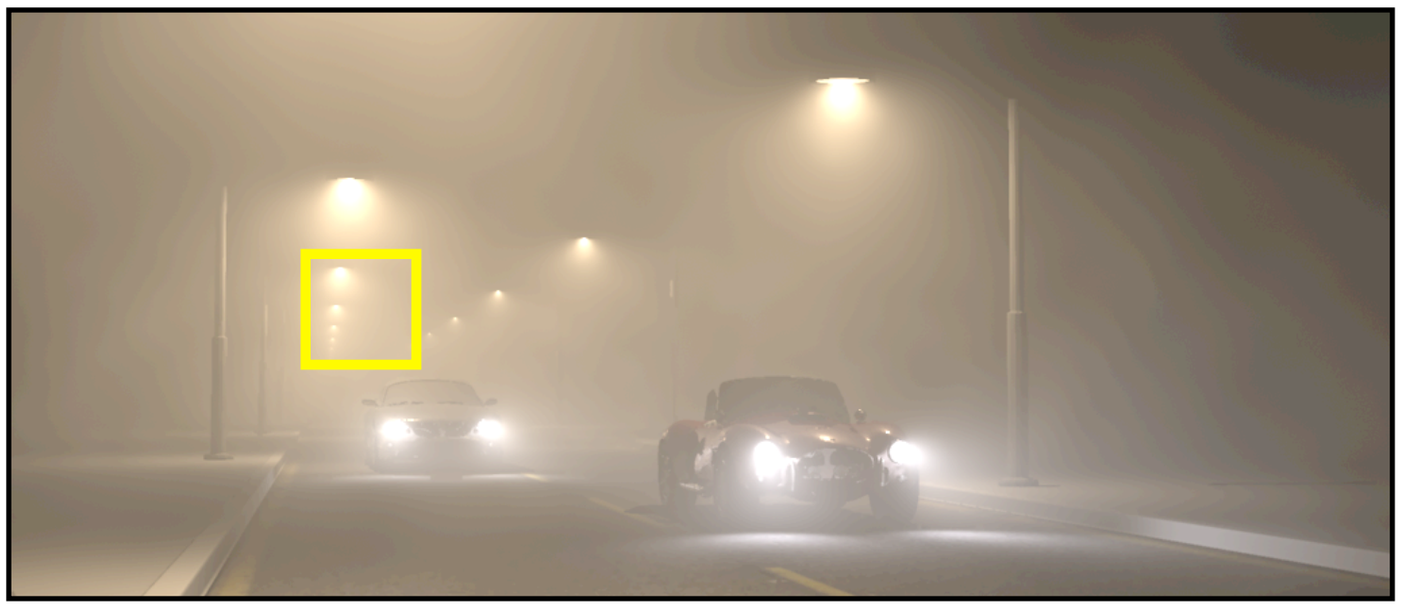
Traditional Estimate

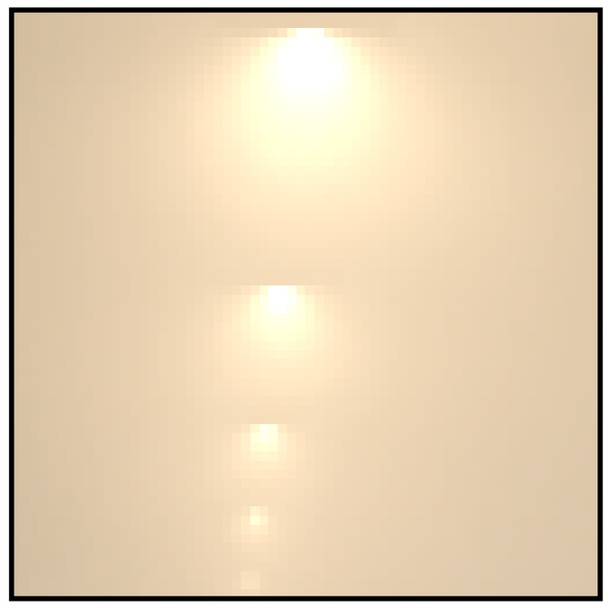
(1:12)

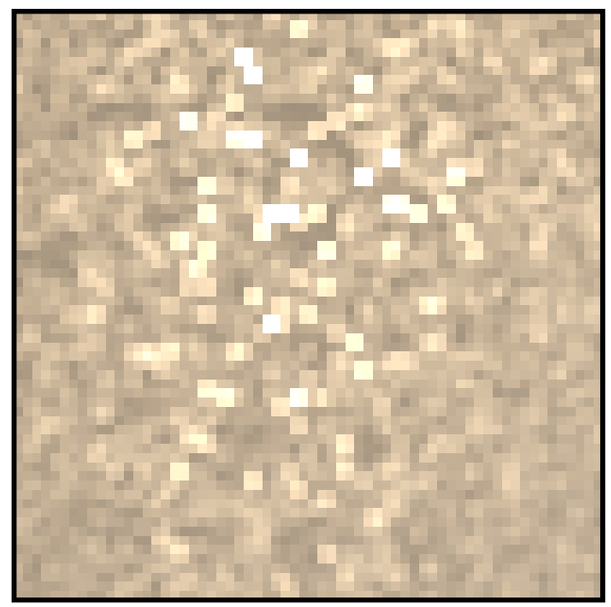
Cars on Foggy Street

Beam Estimate

(1:53)







Traditional Estimate

(2:02)

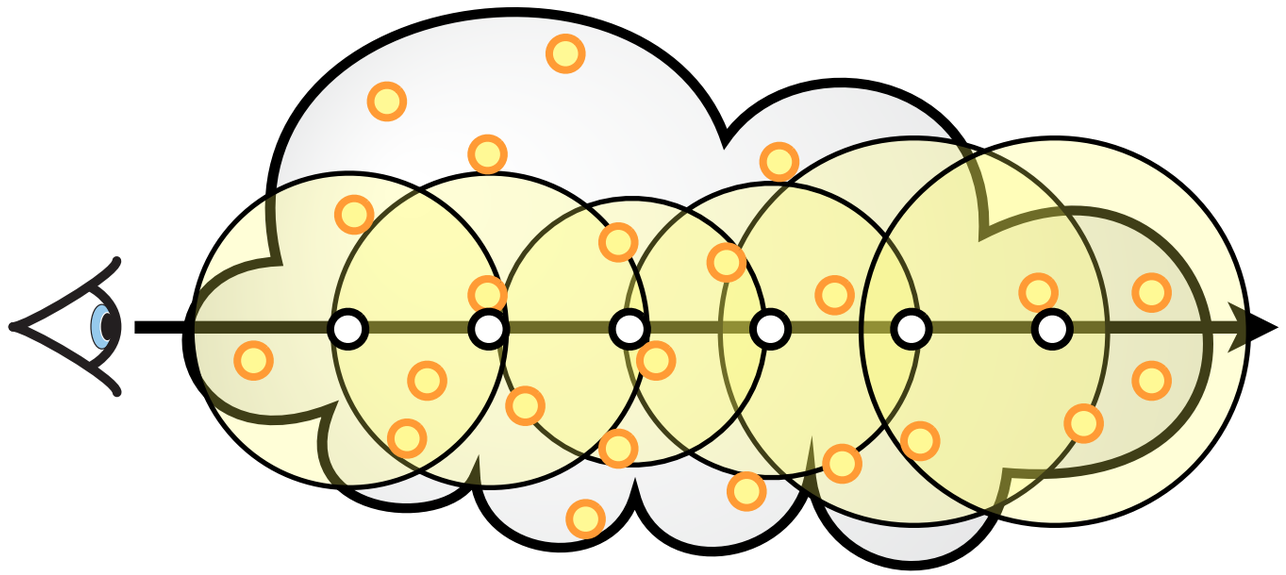
Questions?

Visual Break

So Far...

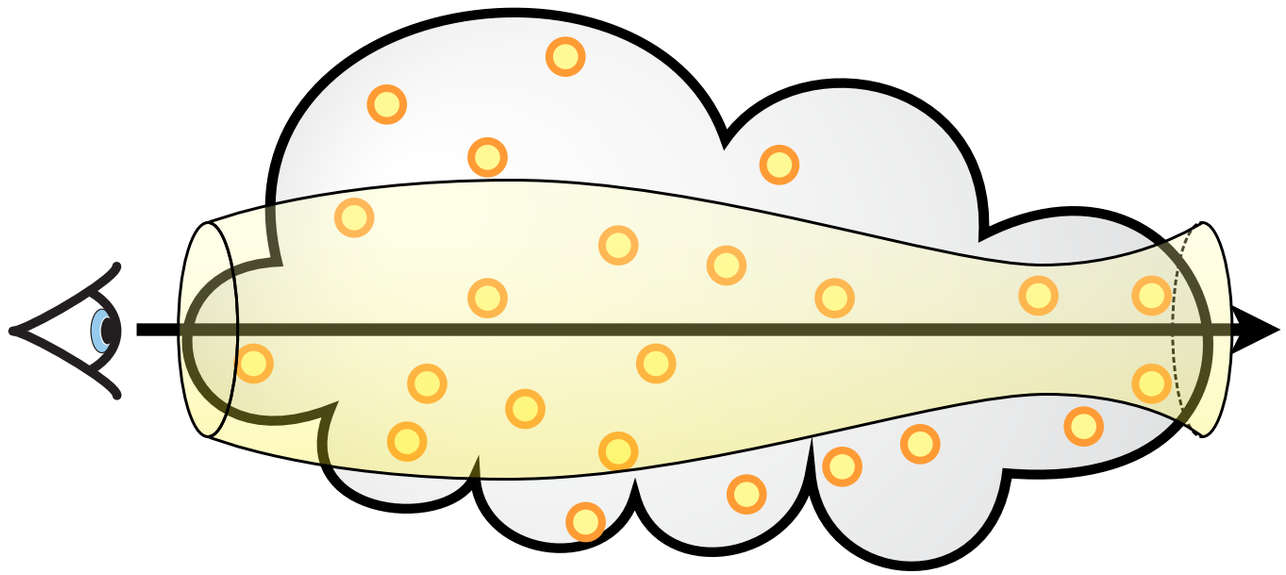
- Volumetric Photon Mapping (**VPM**)
  [Jensen & Christensen 98]

Query	x	Data	Blur
Point	x	Point	(3D)



- The Beam Radiance Estimate (**BRE**)
  [Jarosz et al. 08]

Query	x	Data	Blur
Beam	x	Point	**(2D)**



Other possibilities

Query	x	Data	Blur
Point	x	Point	(3D)
Beam	x	Point	(2D)

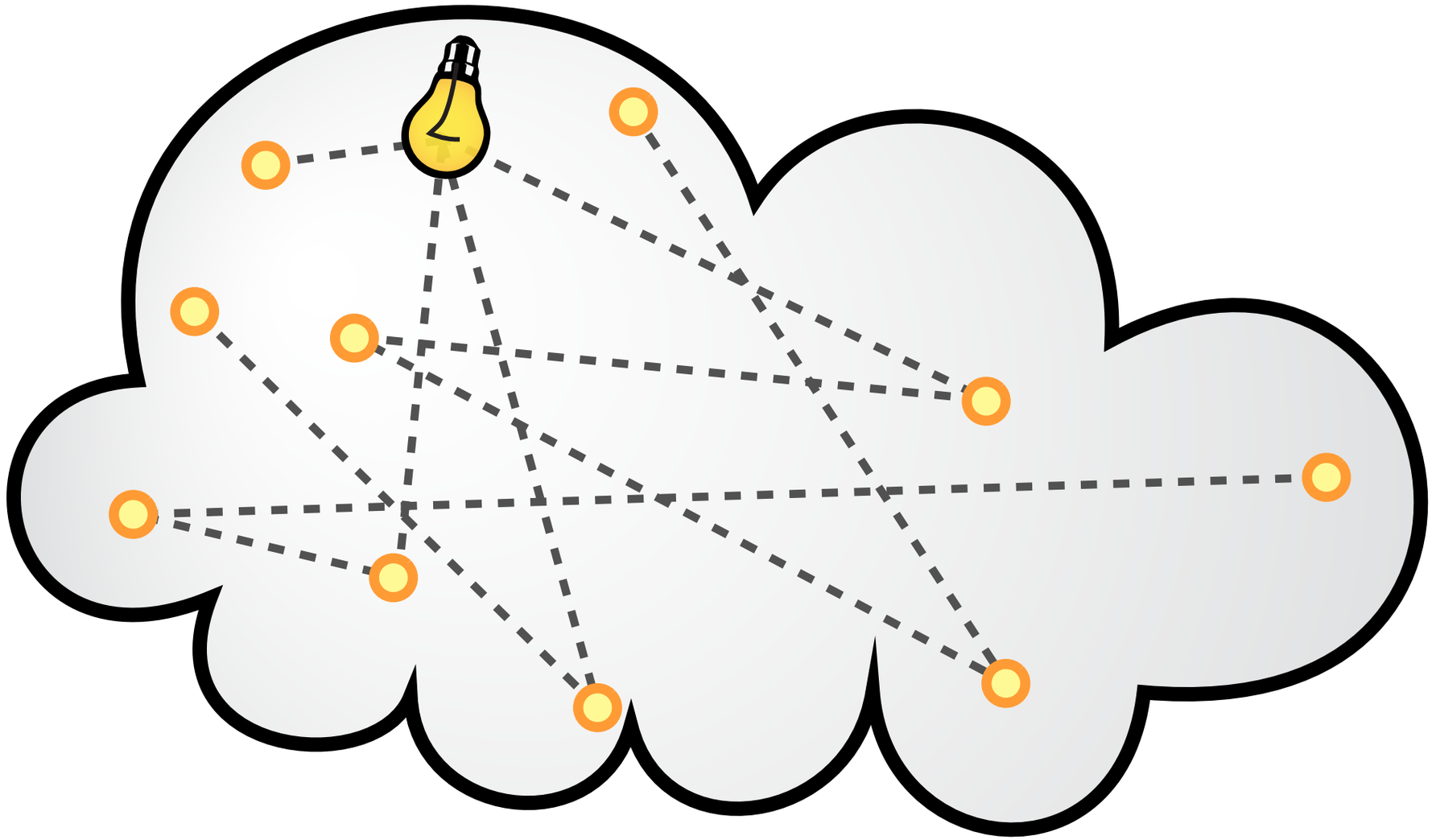
Other possibilities

[Jarosz et al. 11]

Query	x	Data	Blur
Point	x	Point	(3D)
Beam	x	Point	(2D)
Beam	**x**	**Point**	**(3D)**
Point	**x**	**Beam**	**(3D)**
Point	**x**	**Beam**	**(2D)**
Beam	**x**	**Beam**	**(3D)**
Beam	**x**	**Beam**	**$(2D)_1$**
Beam	**x**	**Beam**	**$(2D)_2$**
Beam	**x**	**Beam**	**(1D)**

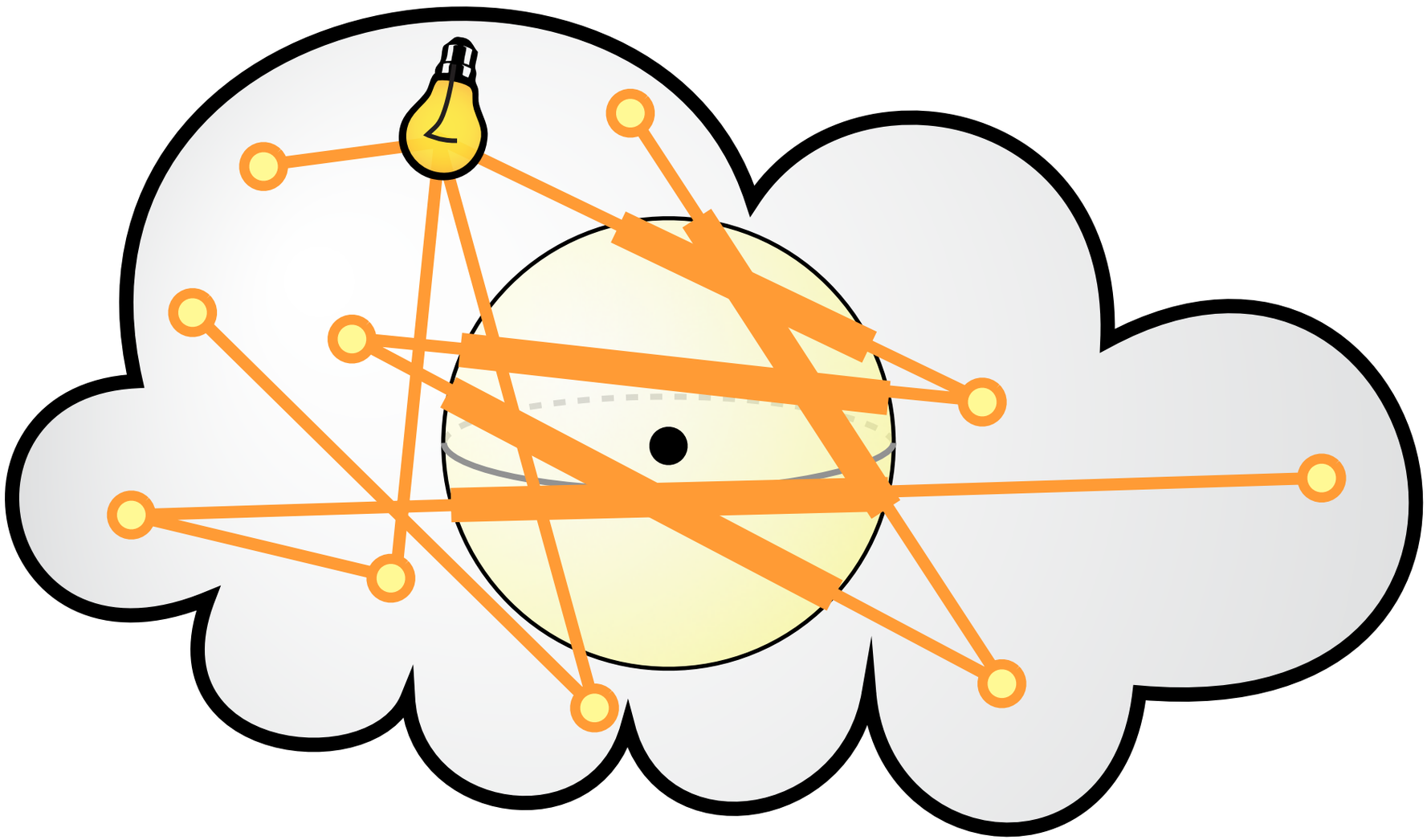
Volumetric Photon Mapping

[Jarosz et al. 11]



Volumetric Photon Mapping

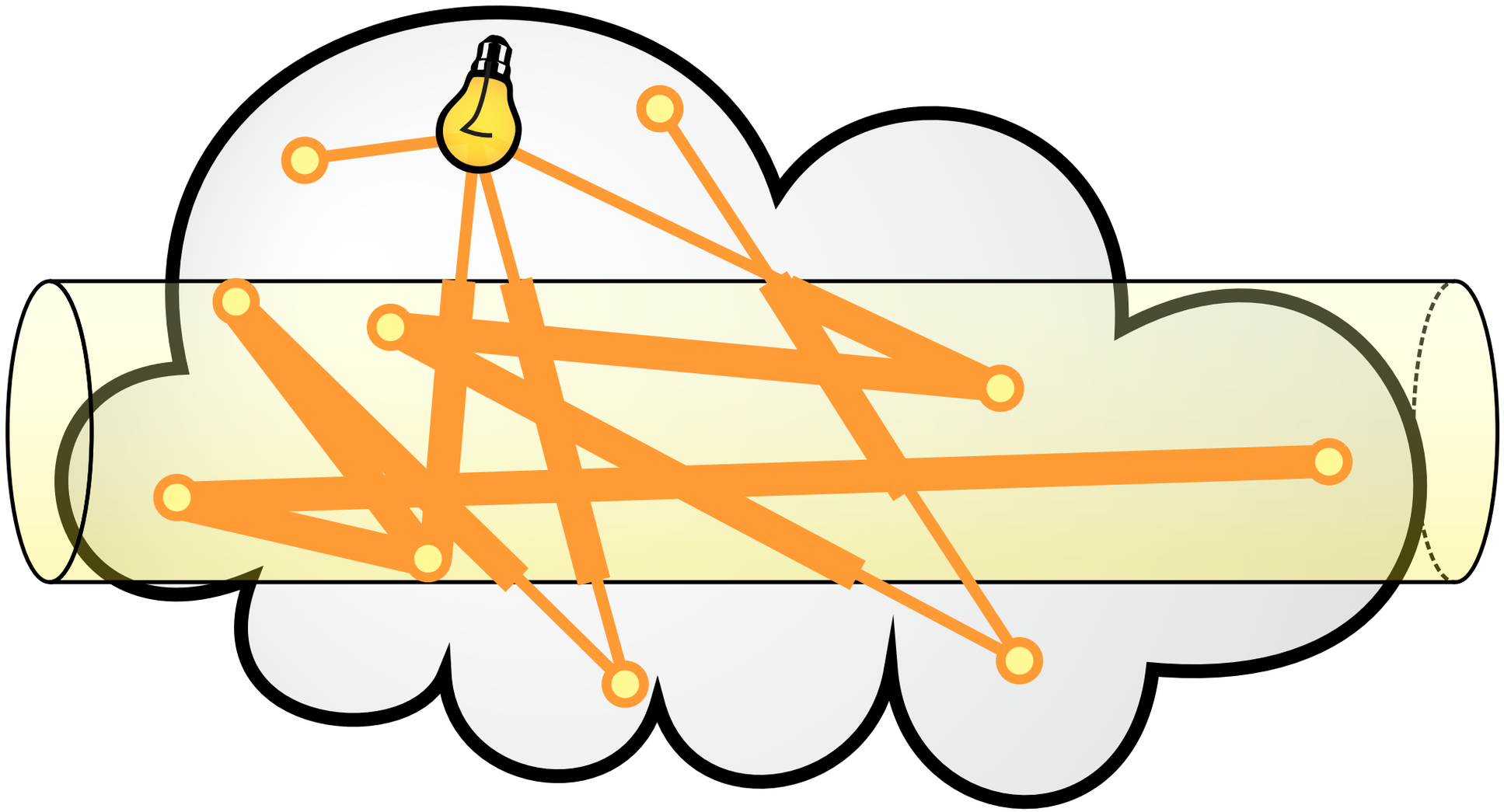
[Jarosz et al. 11]



Photon Beams x Point Query

Volumetric Photon Mapping

[Jarosz et al. 11]

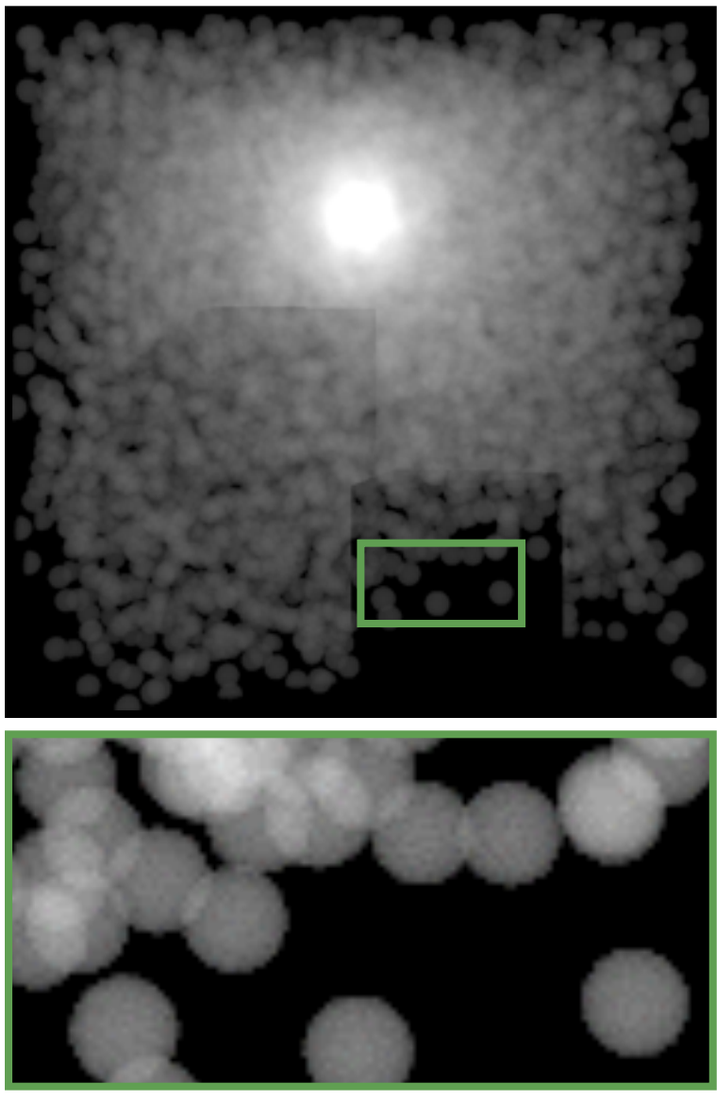


Photon Beams x Beam Query

Photon Points vs. Photon Beams

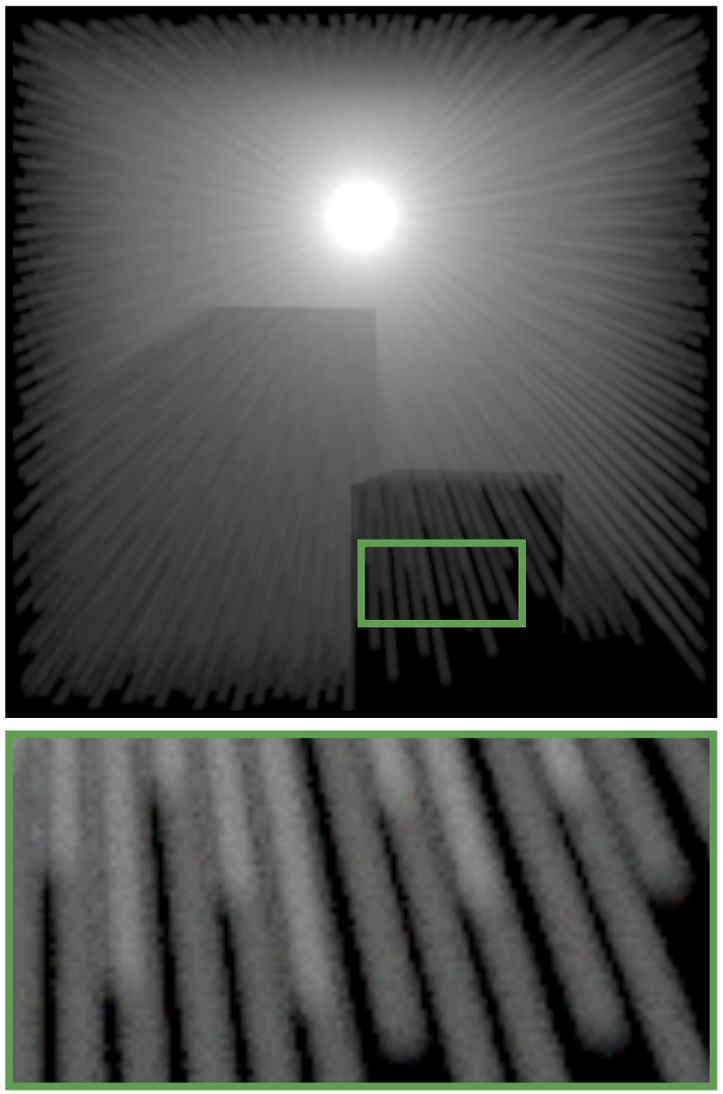
[Jarosz et al. 11]

100k Photon Points

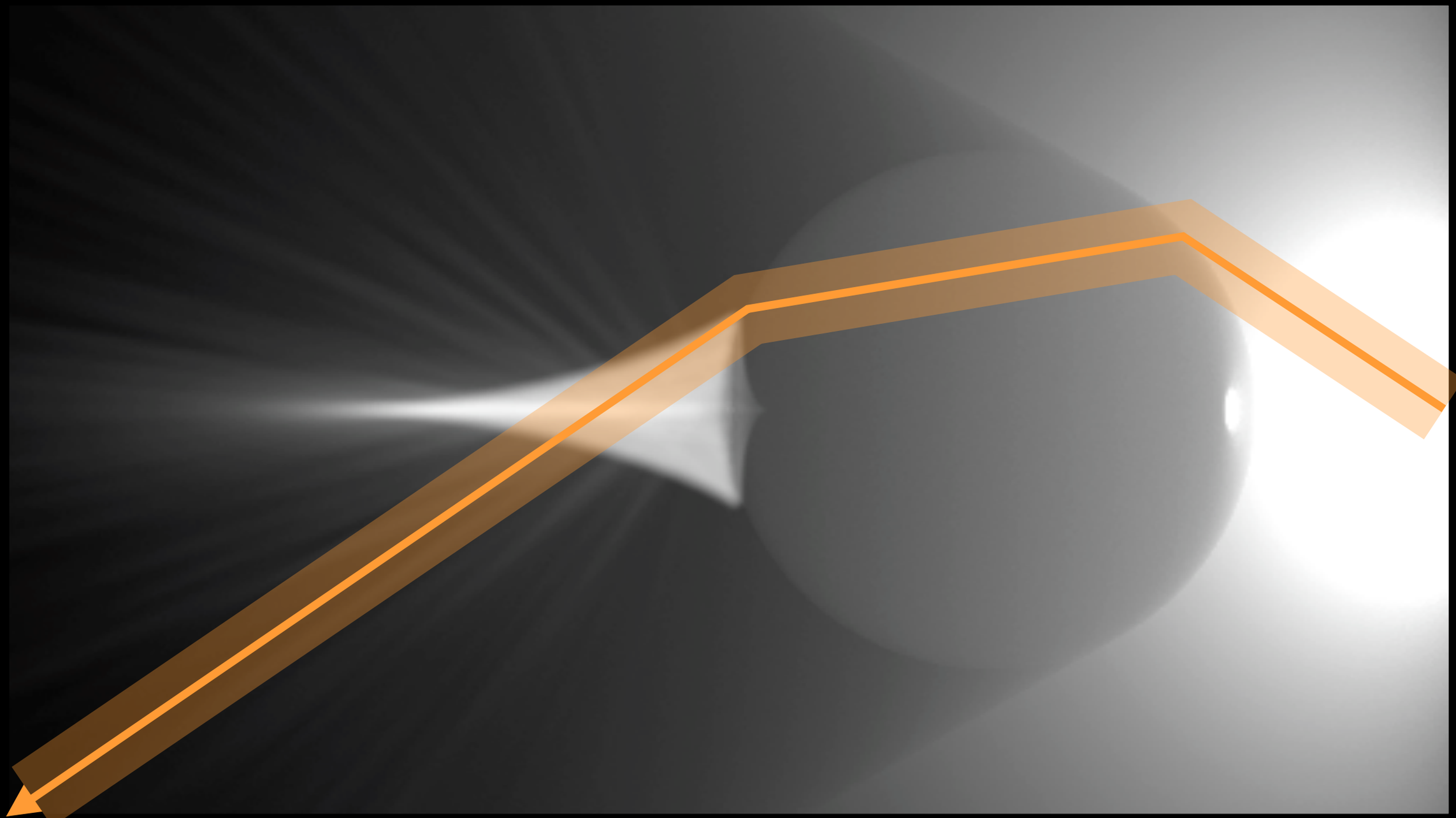


Ground Truth

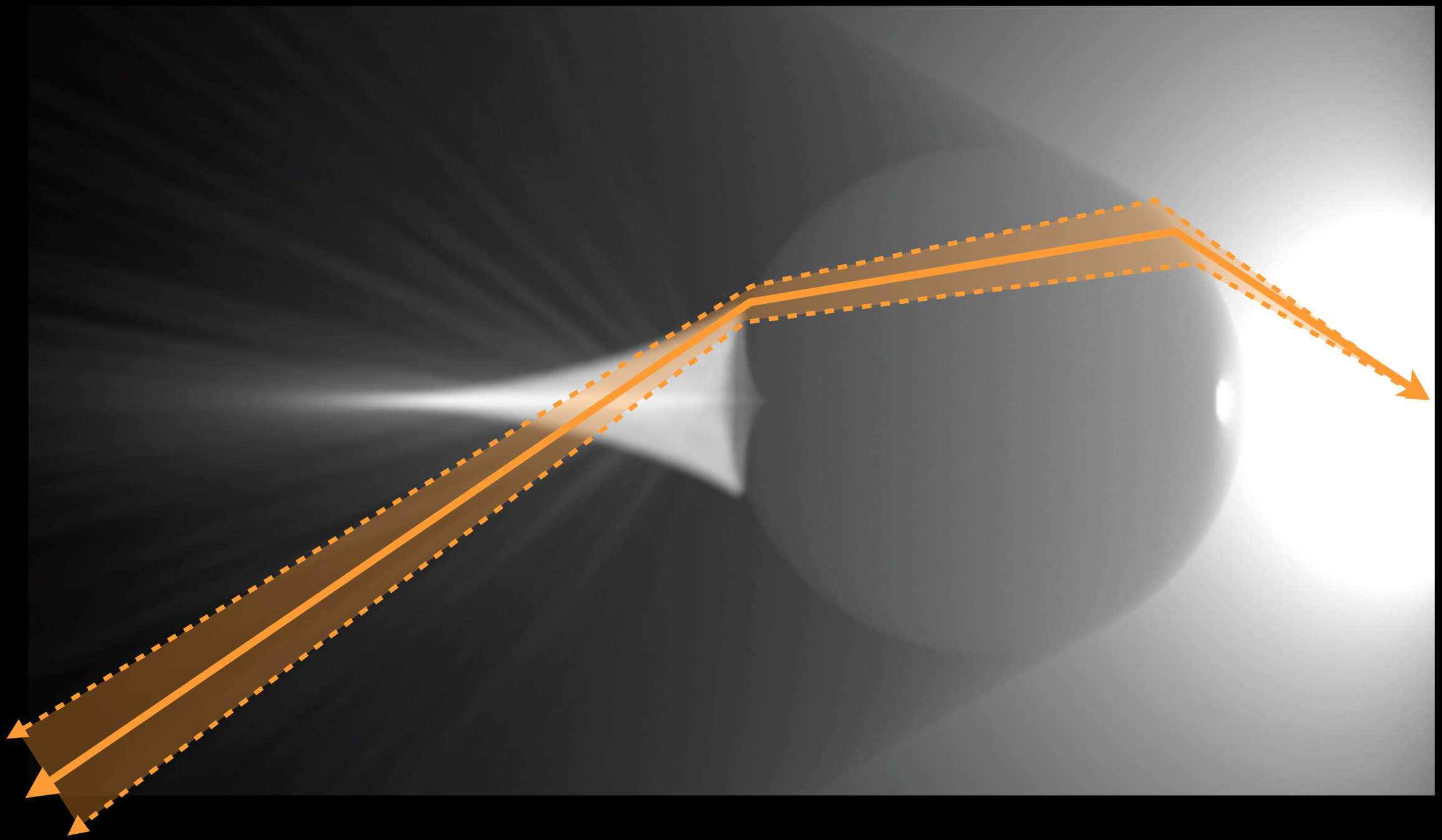
5k Photon Beams



Fixed-width Beams

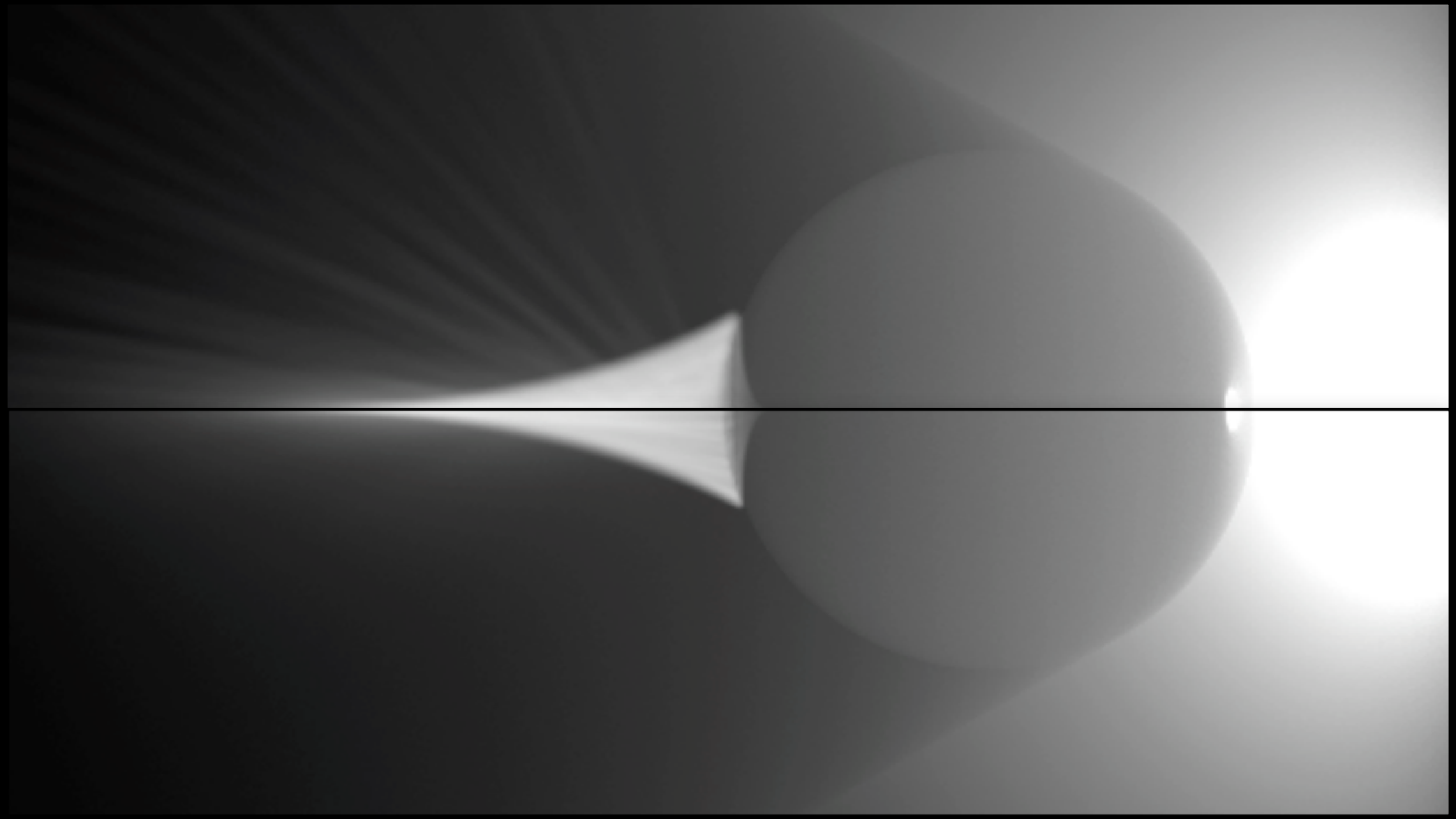


Fixed-width Beams



Photon Differentials
[Igehy 99, Schjøth et al. 07]

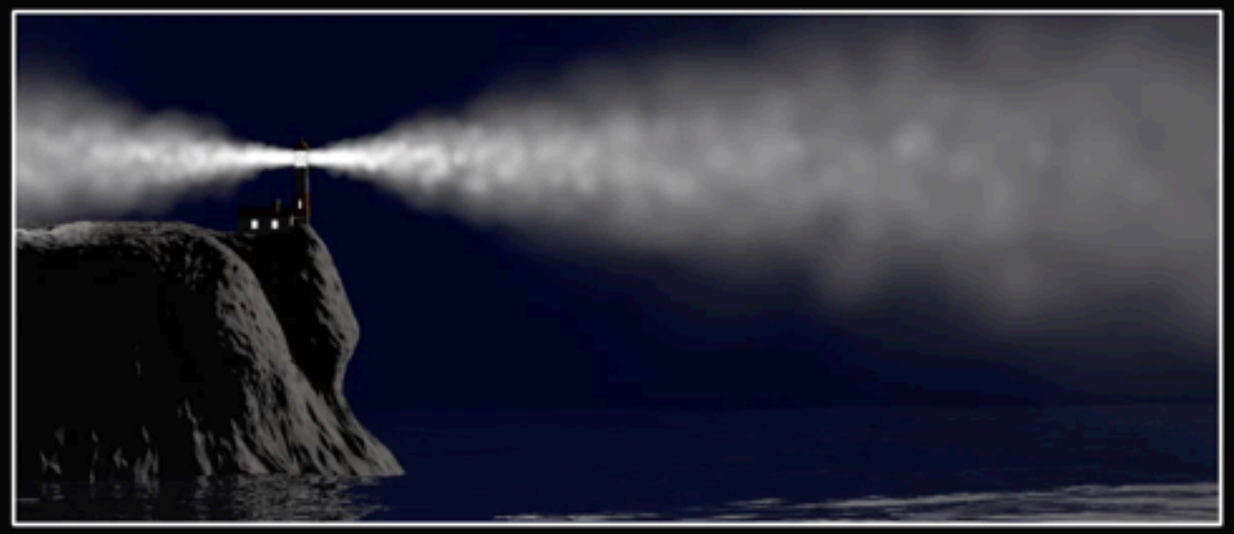
Fixed-width Beams



Adaptive-width Beams

Lighthouse

Photon Points



10K Photon **Points**
~ 31 seconds/frame

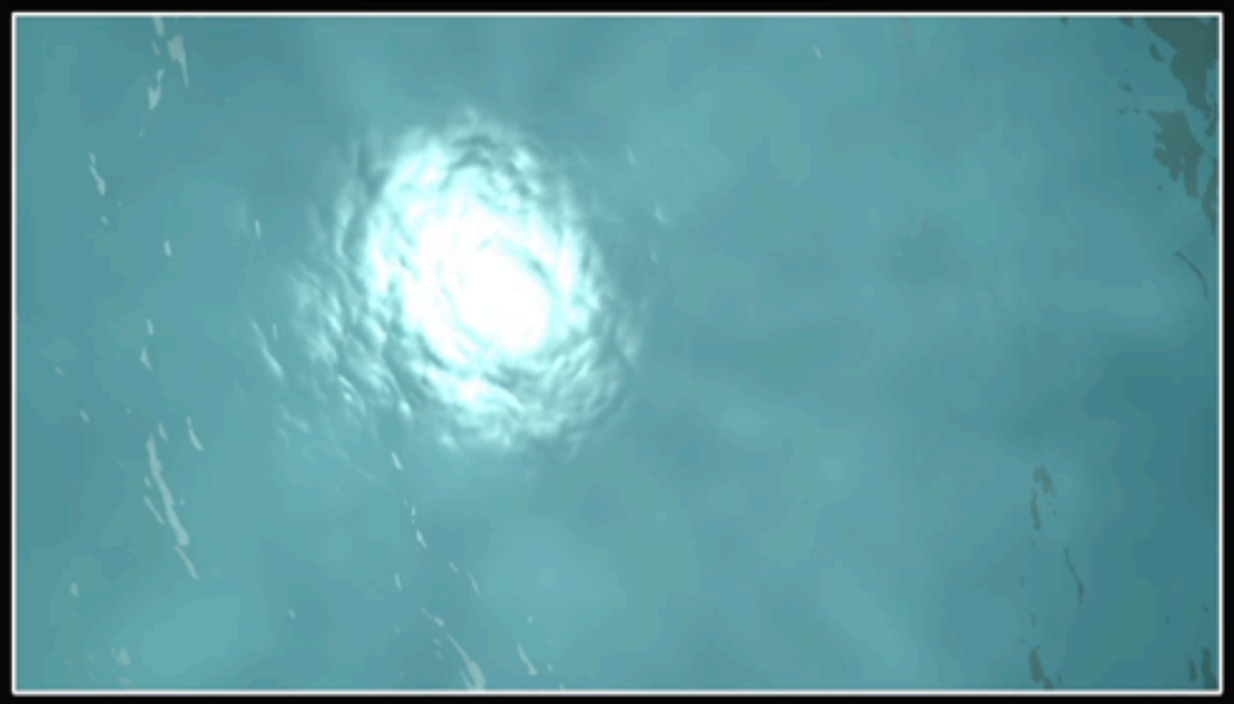
Photon Beams

700 Photon **Beams**
~ 25 seconds/frame

Roughly Equal Time

Underwater Sun Beams

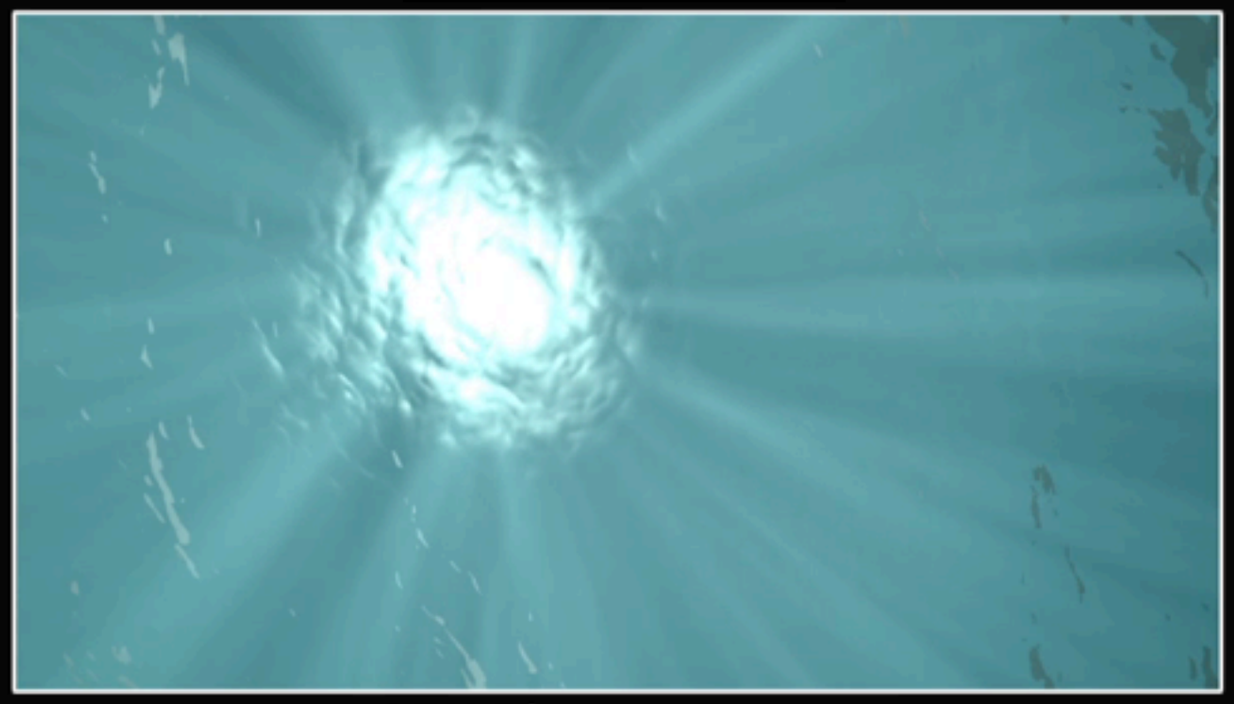
Photon Points



100K Photon **Points**
~ 204 seconds/frame

Roughly Equal Time

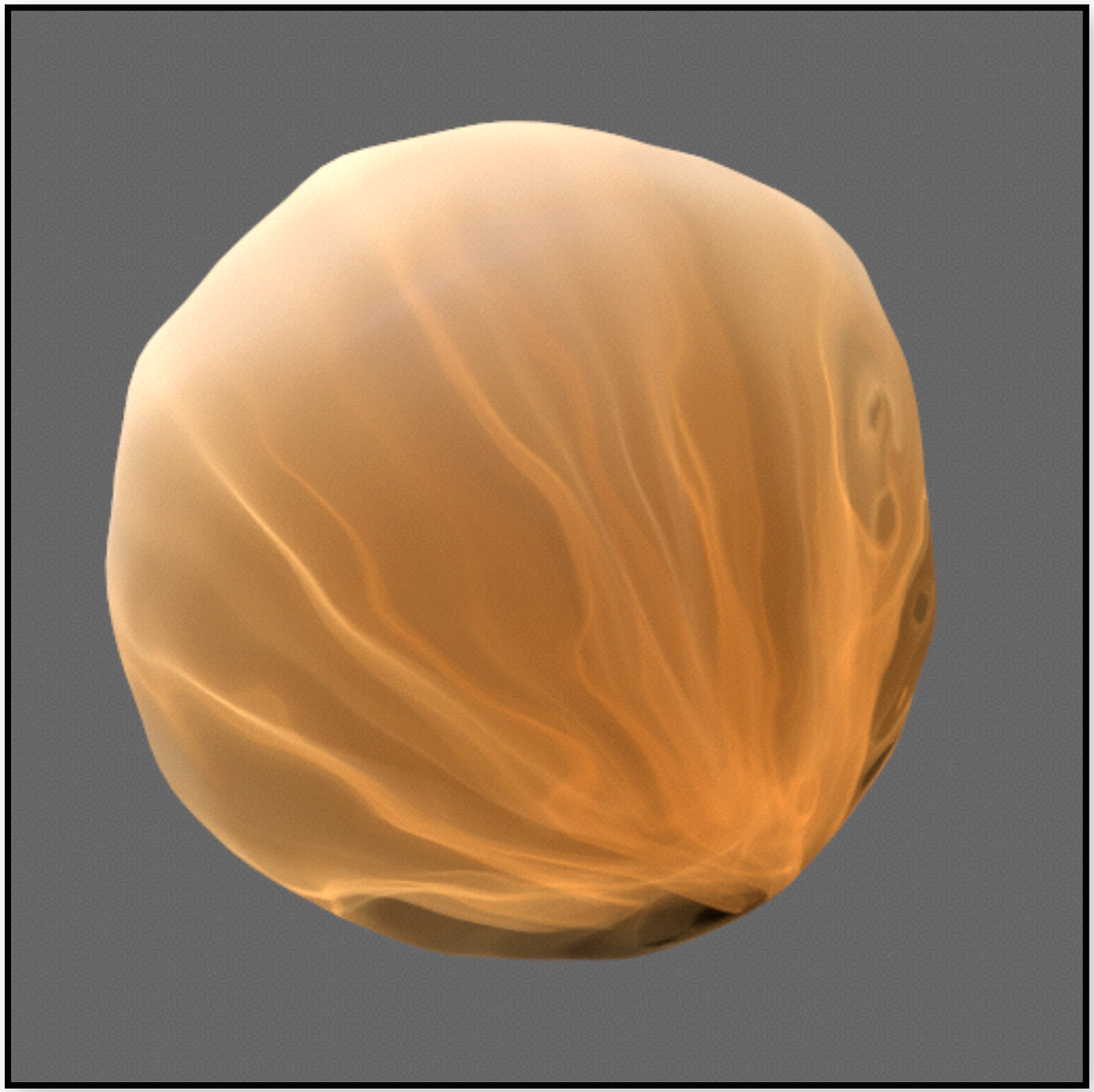
Photon Beams



25K Photon **Beams**
~ 200 seconds/frame

Bumpy Sphere

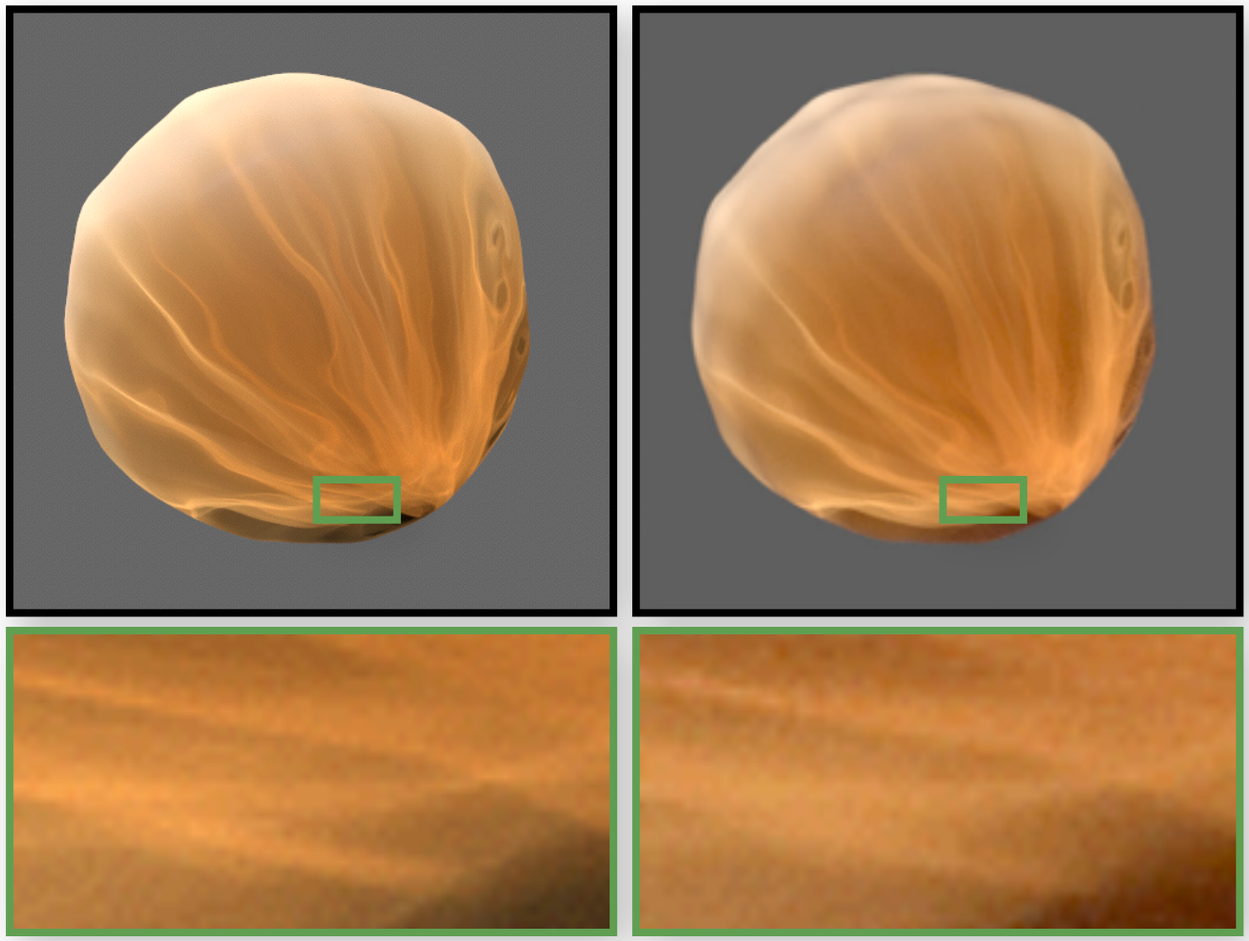
scene courtesy of Bruce Walter



Bumpy Sphere

scene courtesy of Bruce Walter

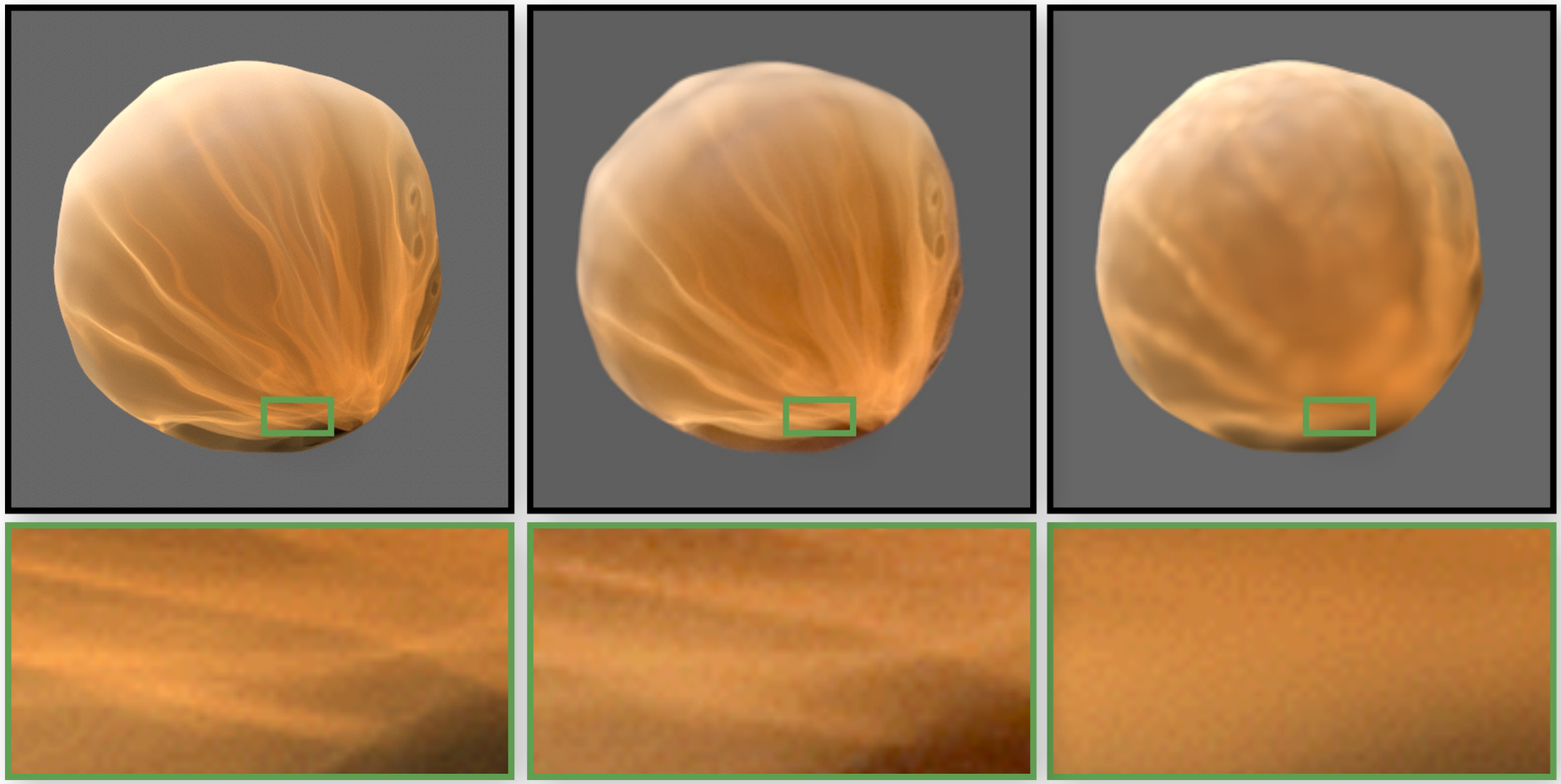
Ground Truth **[Walter et al. 09]**



Bumpy Sphere

scene courtesy of Bruce Walter

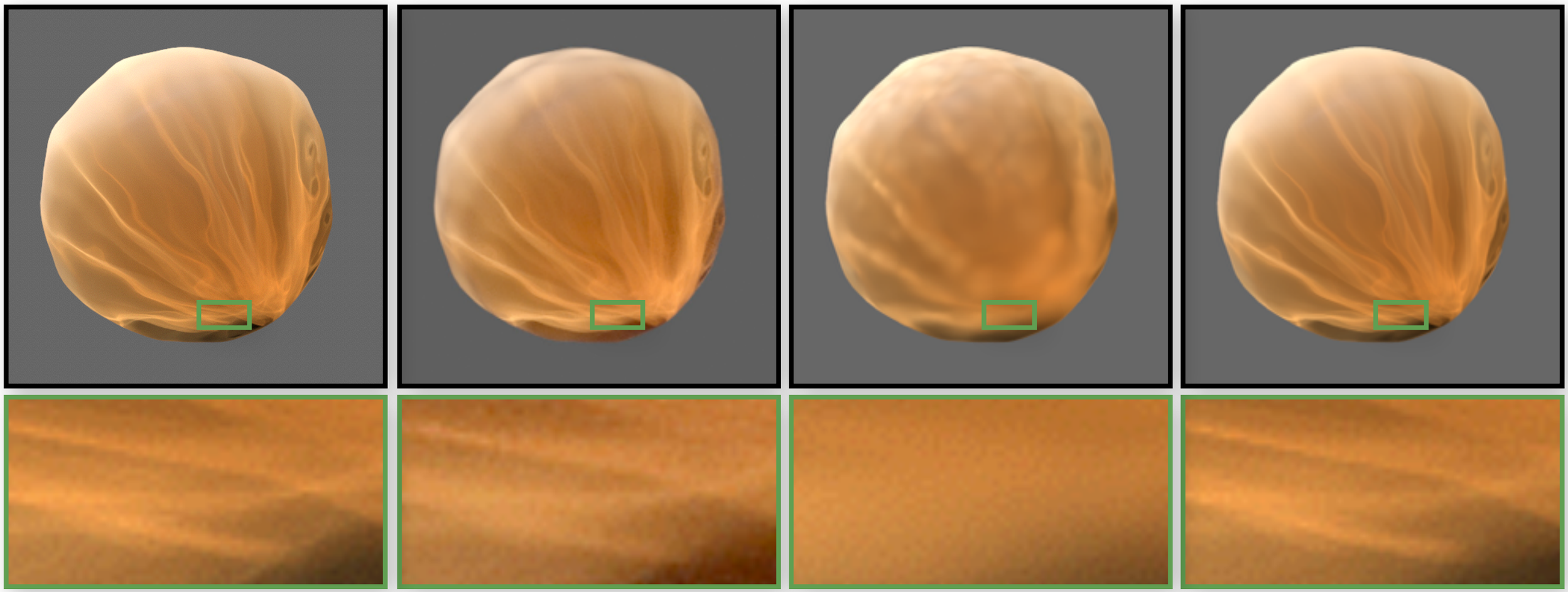
Ground Truth **[Walter et al. 09]** **90k Photon Points**



Bumpy Sphere

scene courtesy of Bruce Walter

Ground Truth **[Walter et al. 09]** **90k Photon Points** **90k Photon Beams**

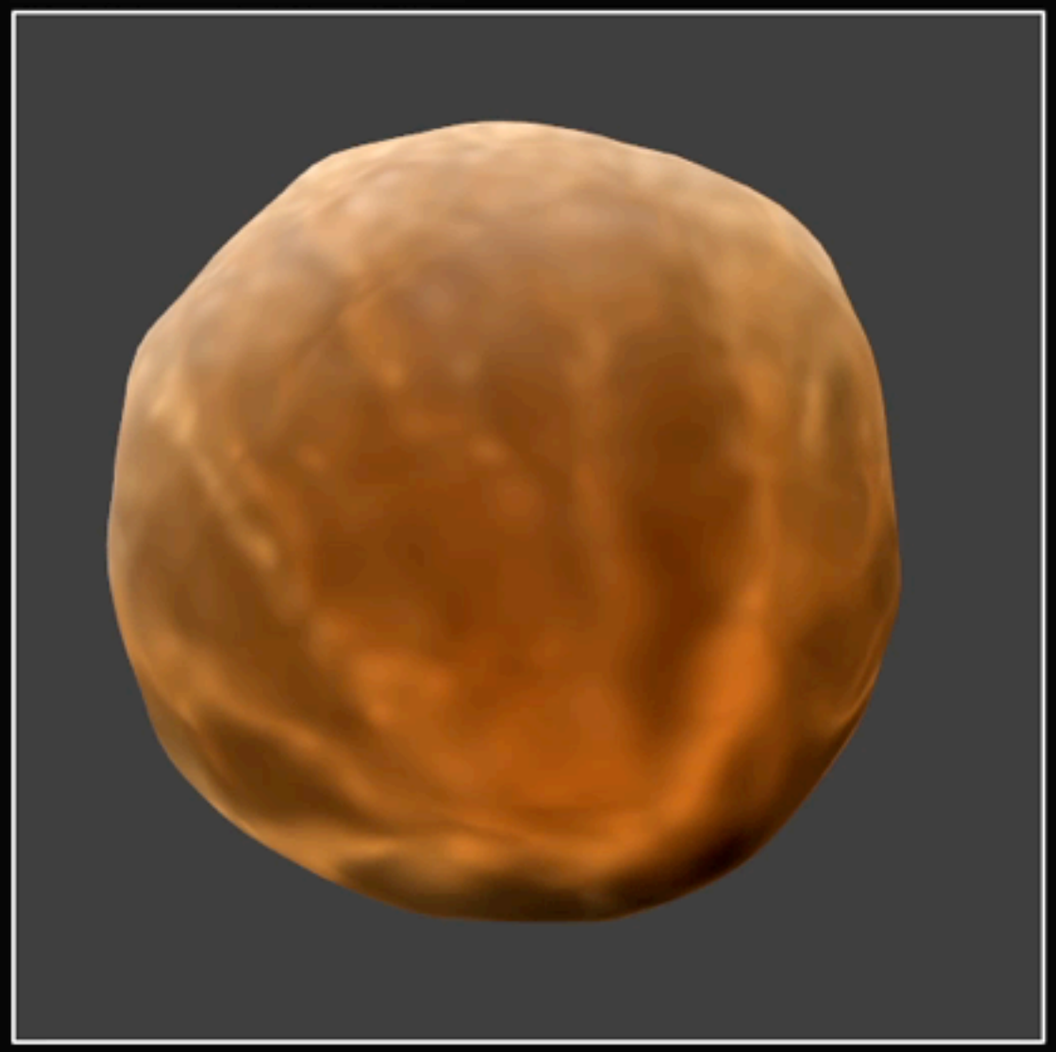


Bumpy Sphere

Rendered at 512x512 with up to 16 samples/pixel

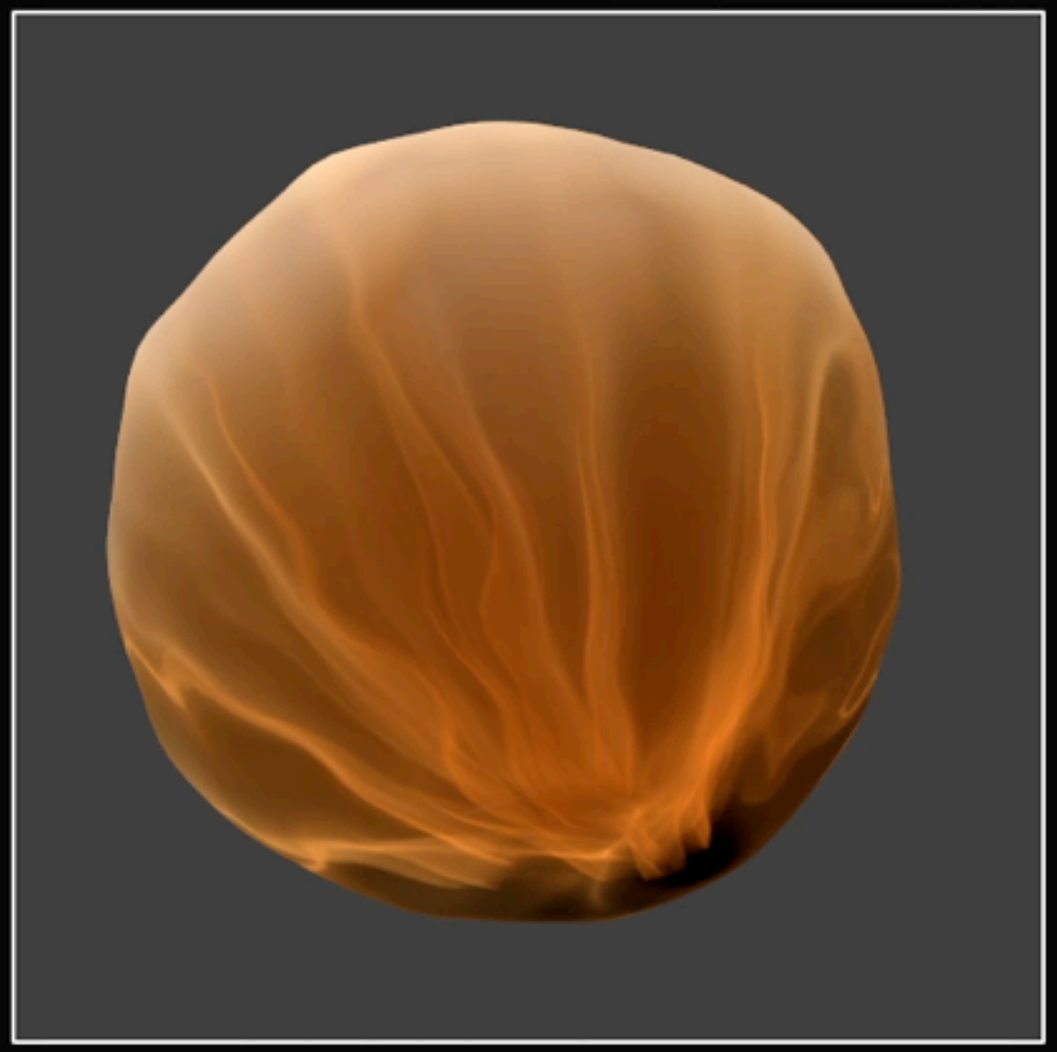
Equal Photon Count

Photon Points



90K Photon **Points**
~ 40 seconds/frame

Photon Beams



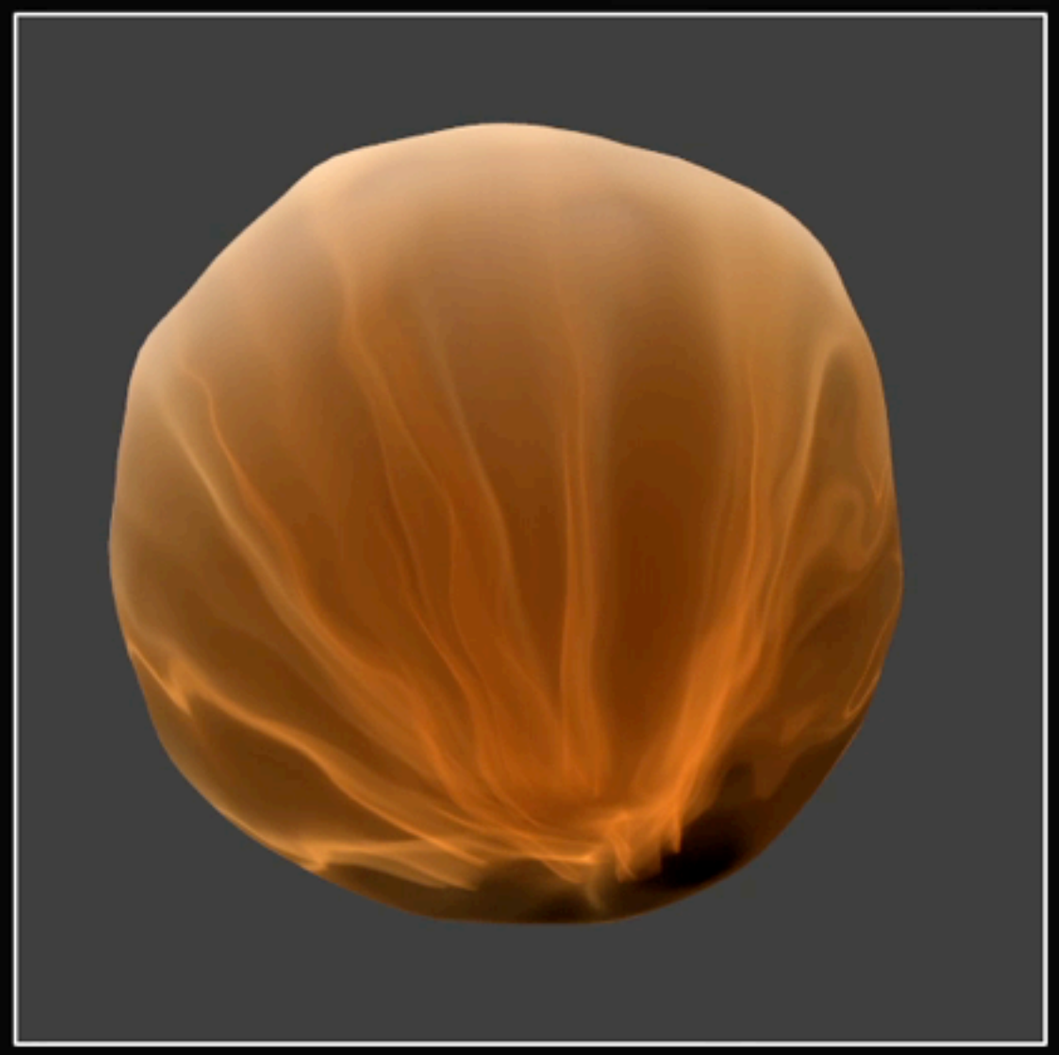
90K Photon **Beams**
~ 103 seconds/frame

Equal Render Time

Photon Points

1.3M Photon **Points**
~ 101 seconds/frame

Photon Beams



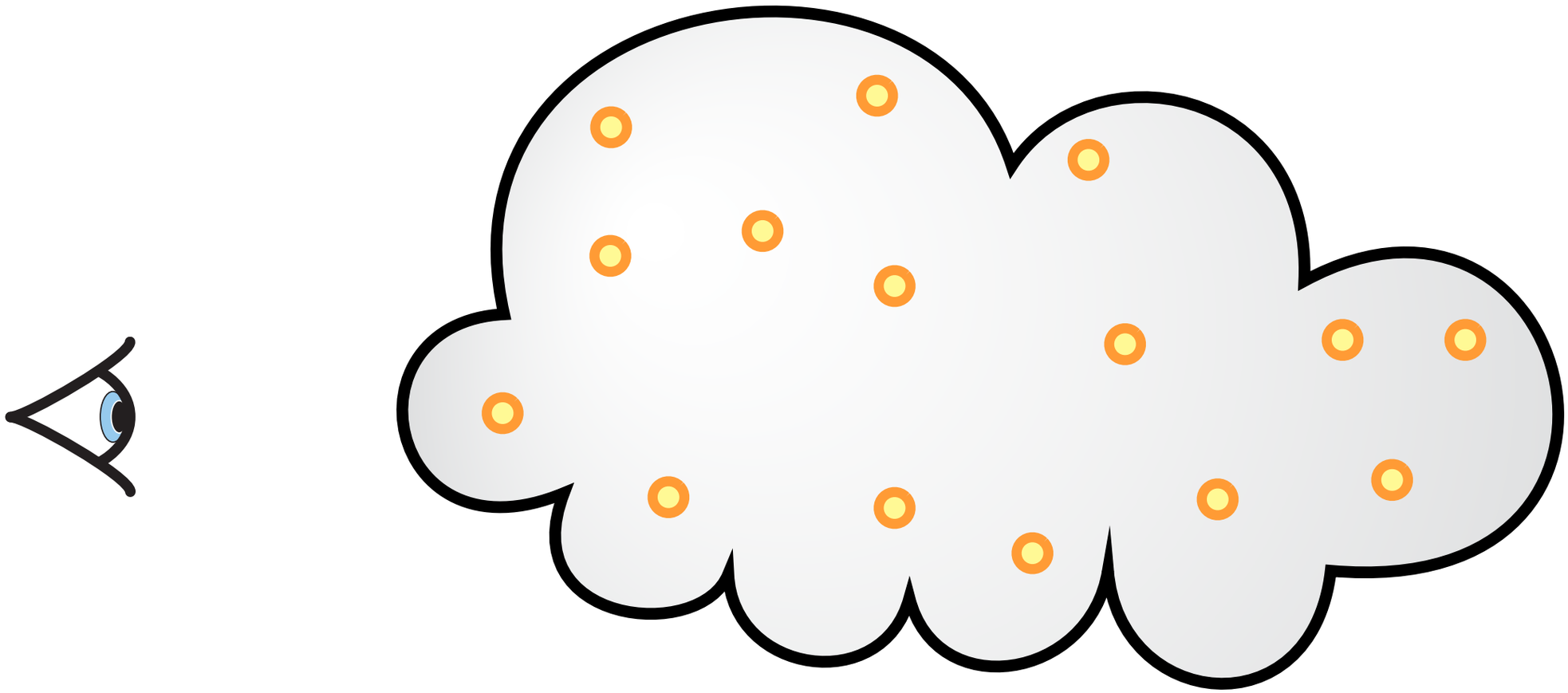
90K Photon **Beams**
~ 103 seconds/frame

Photon Beams

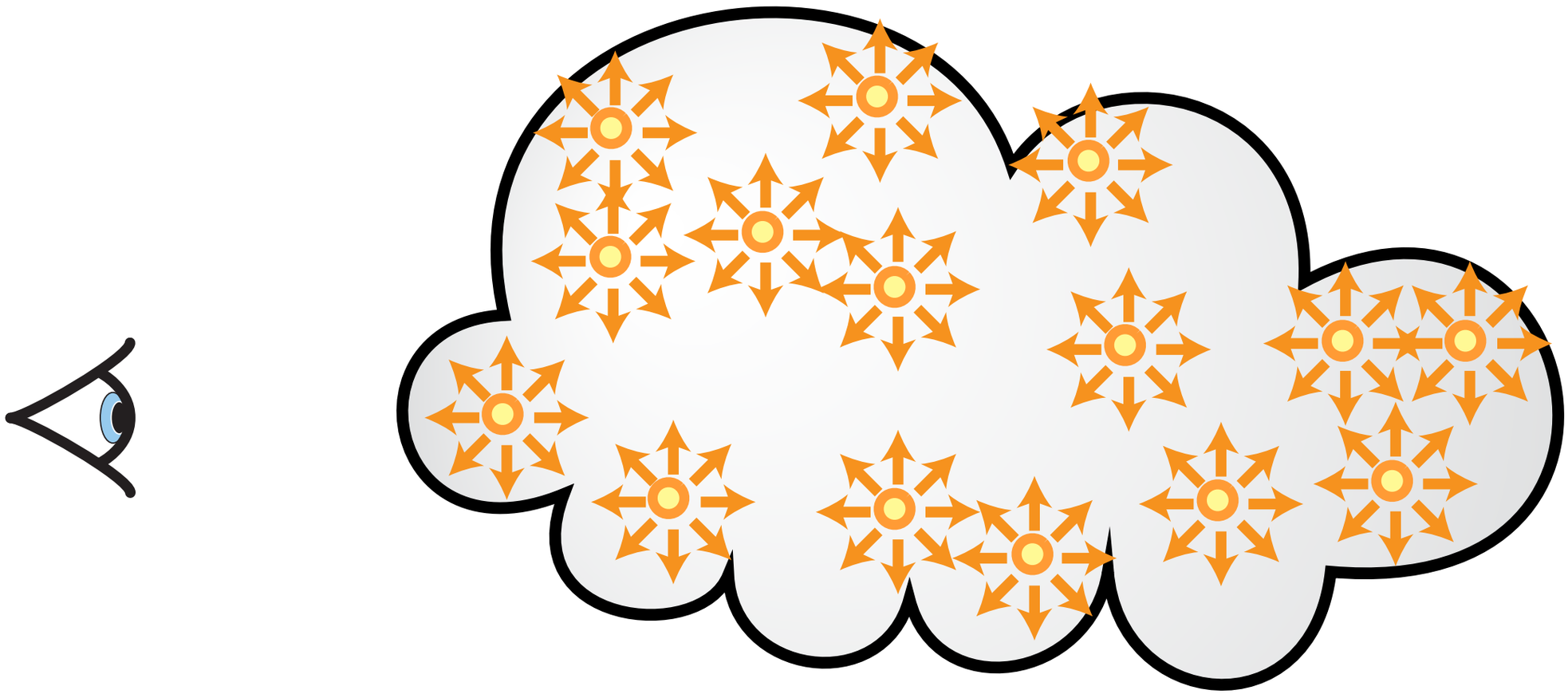
- Advantages:
  - "Up-res'ing" # of photon along paths
  - Lower-dimensional kernels reduce bias
  - Less variance
  - GPU-friendly (rasterize beams into framebuffer!)
- Difficulties:
  - Need to intersect each ray with all photon beams (expensive!)

Volumetric Many-light Rendering

Volumetric VPLs



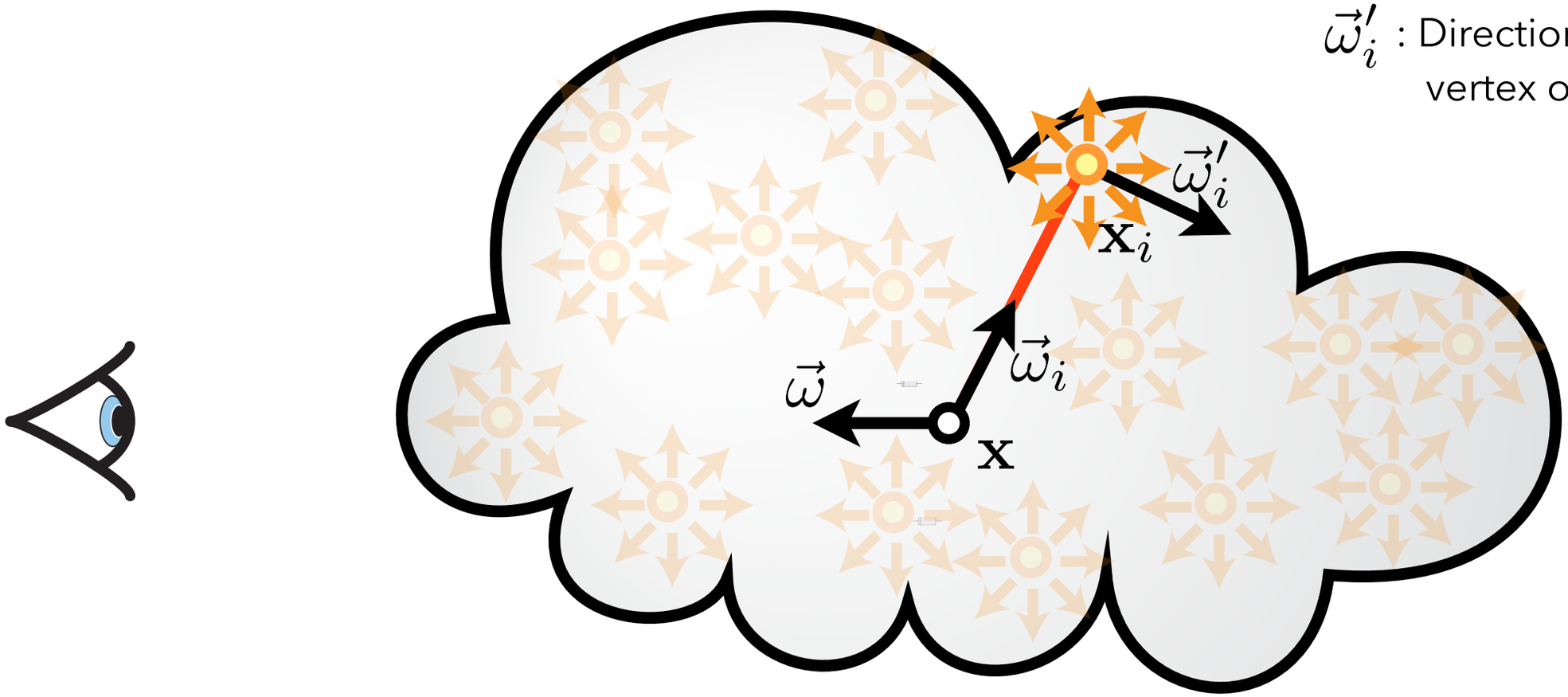
Volumetric VPLs



Turn photons into volumetric virtual point lights

Volumetric VPLs

$\vec{\omega}_i'$: Direction towards previous vertex on the photon path



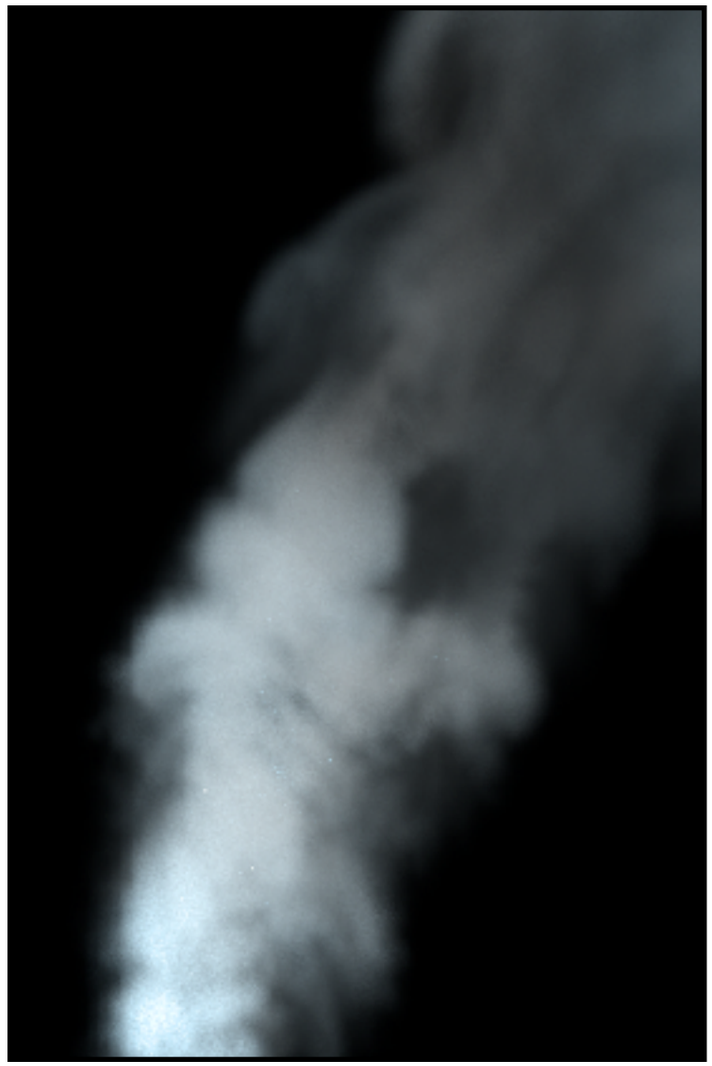
$$L_s(\mathbf{x}, \vec{\omega}) \approx \sum_{i=1}^{N} f_p(\mathbf{x}, \vec{\omega}_i, \vec{\omega}) T_r(\mathbf{x}, \mathbf{x}_i) G(\mathbf{x}, \mathbf{x}_i) f_p(\mathbf{x}_i, \vec{\omega}_i', -\vec{\omega}_i) \Phi_i$$

$$G(\mathbf{x}, \mathbf{x}_i) = V(\mathbf{x}, \mathbf{x}_i) \frac{1}{\|\mathbf{x} - \mathbf{x}_i\|^2}$$

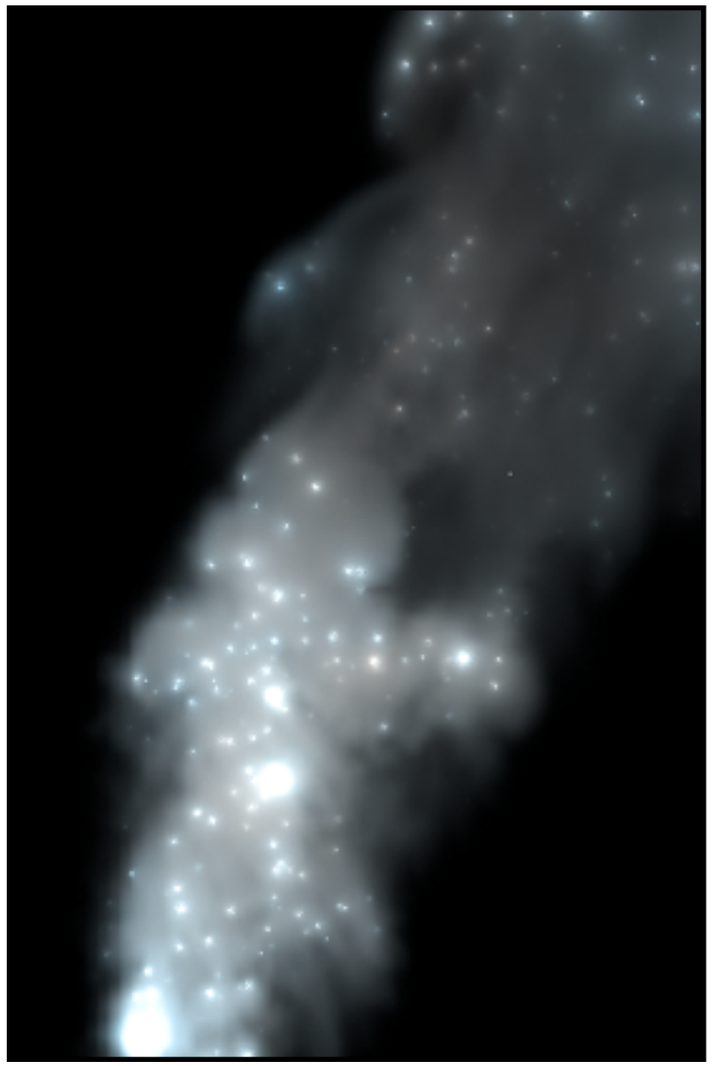
There used to be two cosines here, but they disappeared because we are in a volume!

Volumetric VPLs

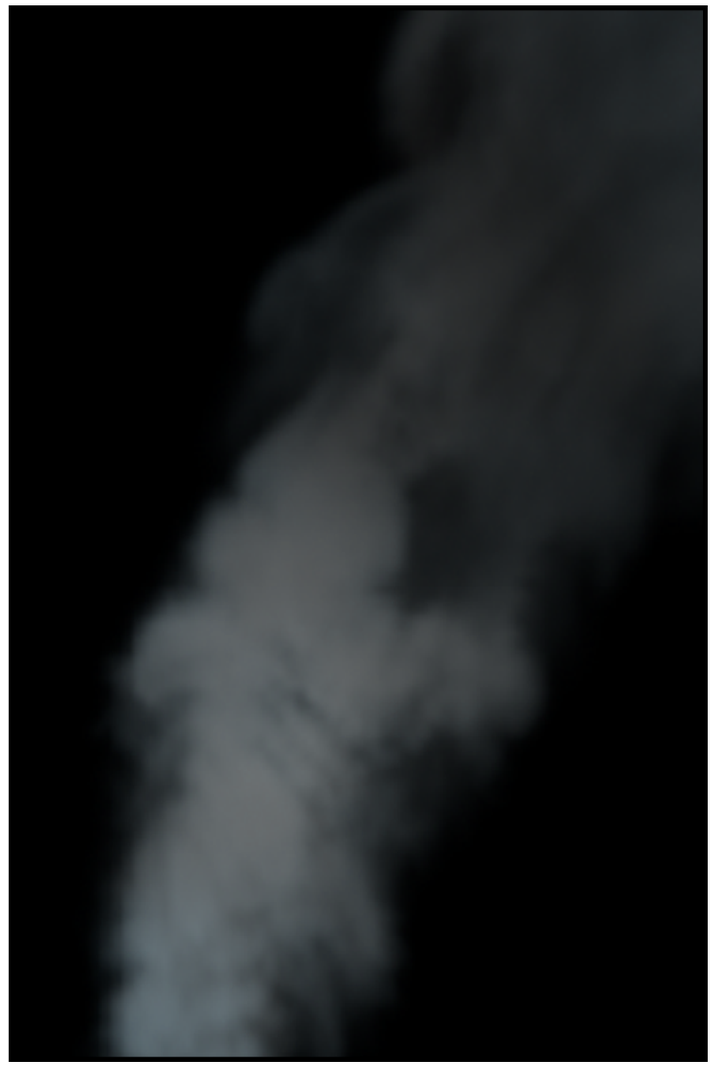
Reference



Unbounded G-term



Bounded G-term



Splotches

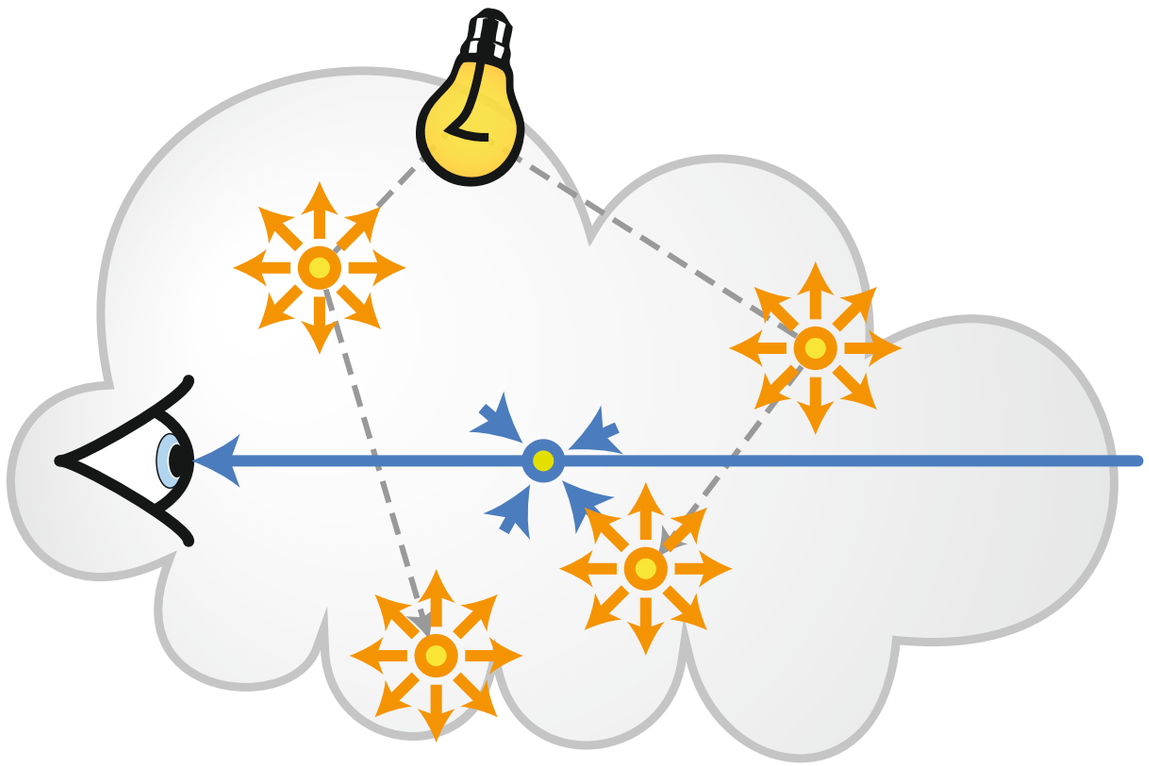
Energy loss

Can we reduce the splotches differently?

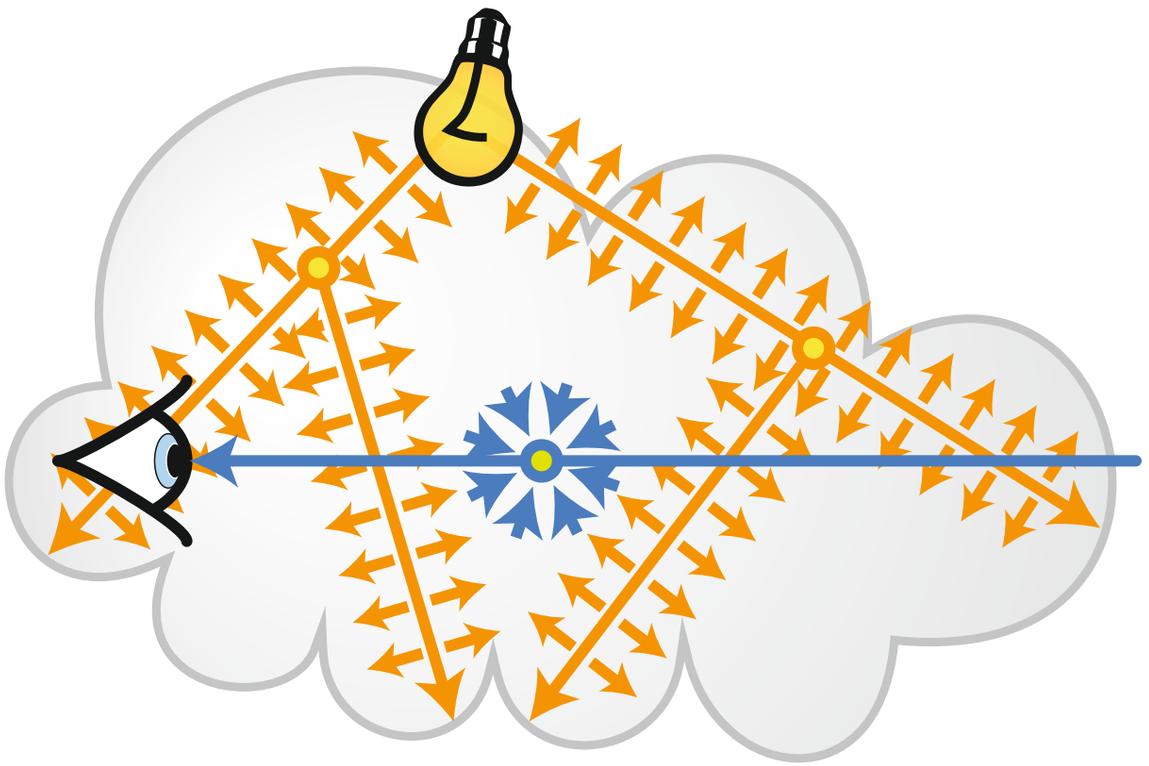
Virtual Ray Lights

[Novák et al. 12a]

- "Spread" energy along segments of photon path
  - Similar concept as in photon beams



Virtual Point Lights

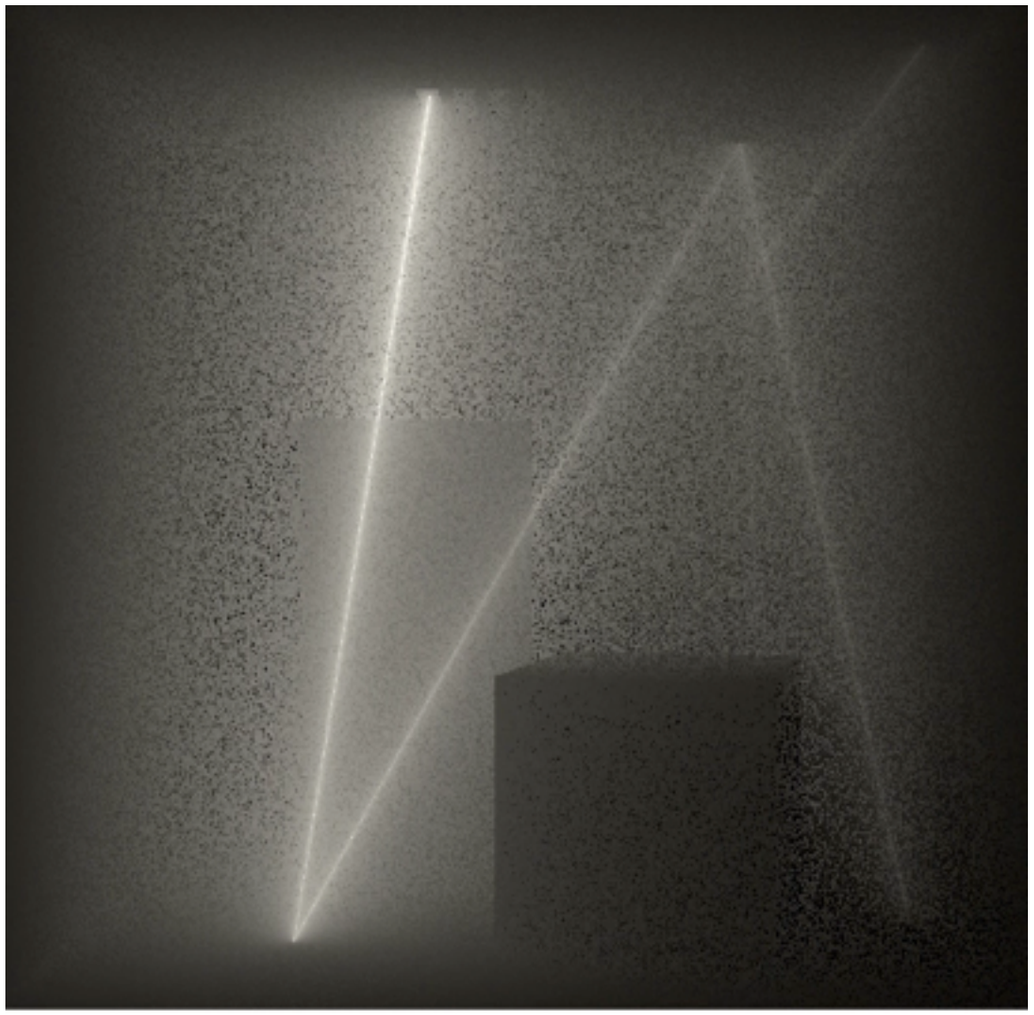


Virtual Ray Lights

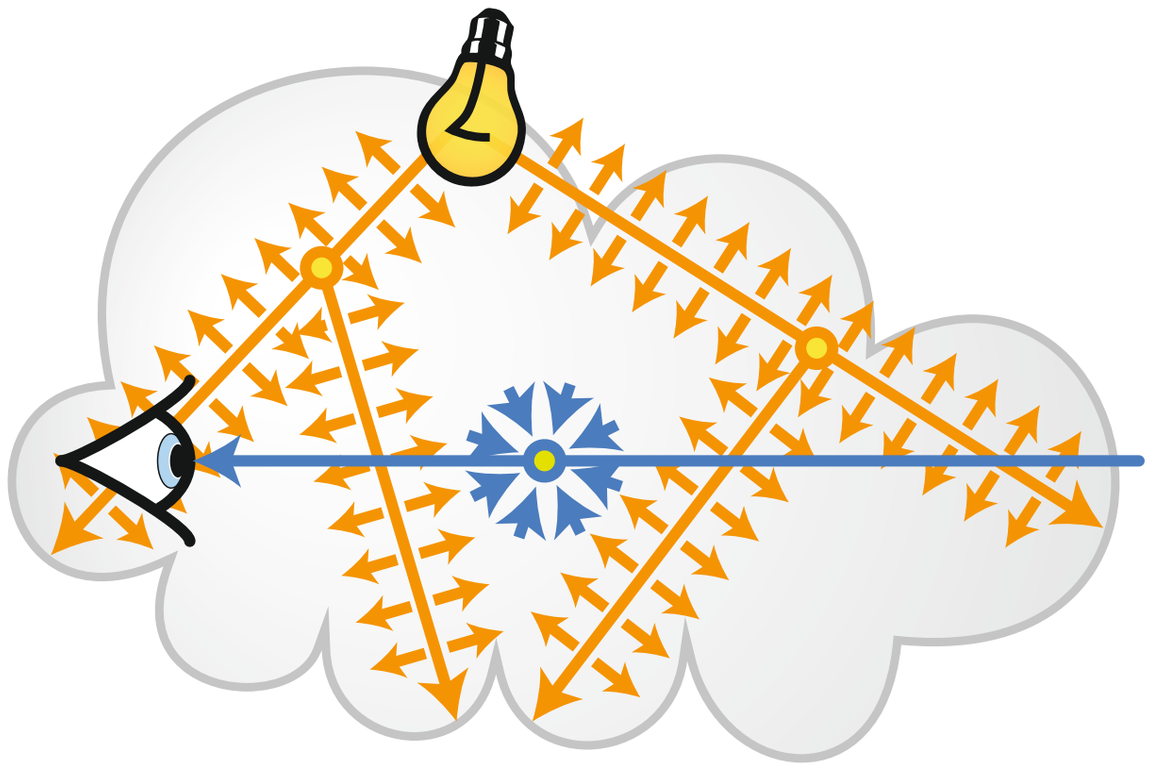
Virtual Ray Lights

[Novák et al. 12a]

- "Spread" energy along segments of photon path
  - Similar concept as in photon beams



6 VRLs

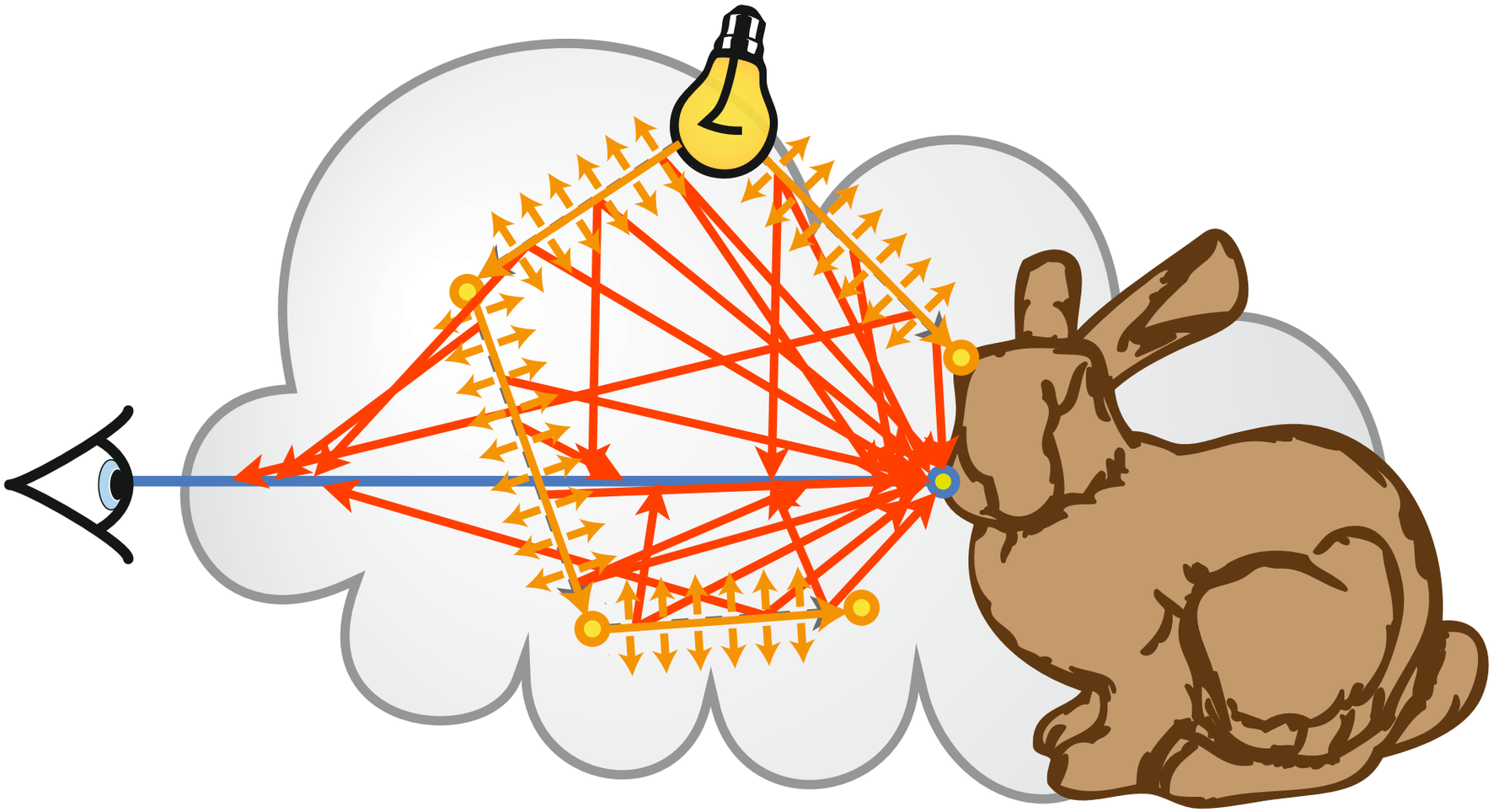


Virtual Ray Lights

Virtual Ray Lights

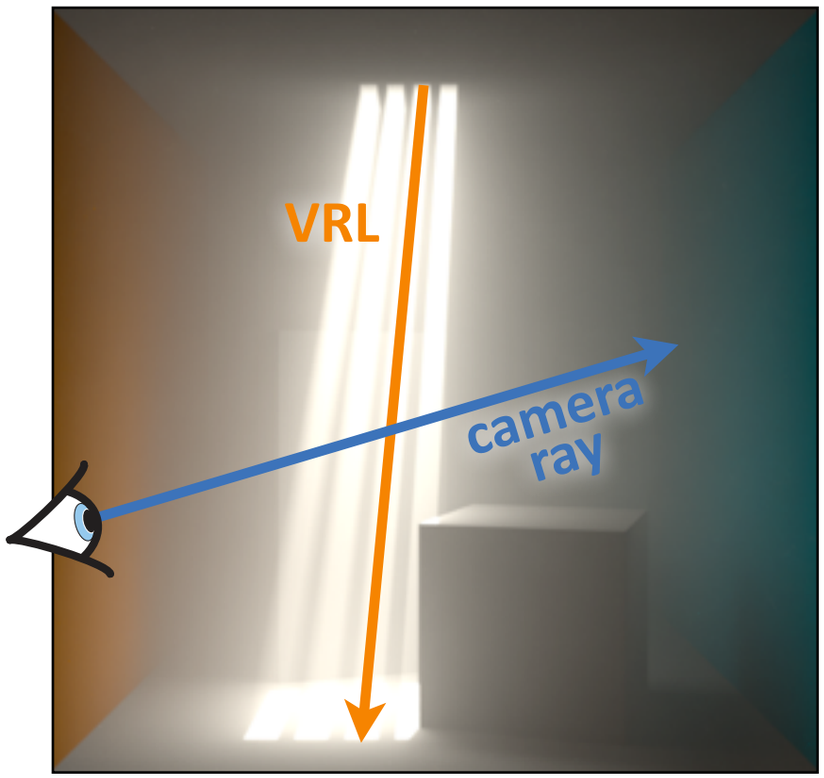
[Novák et al. 12a]

- "Spread" energy along segments of photon path
  - Similar concept as in photon beams



Virtual Ray Lights

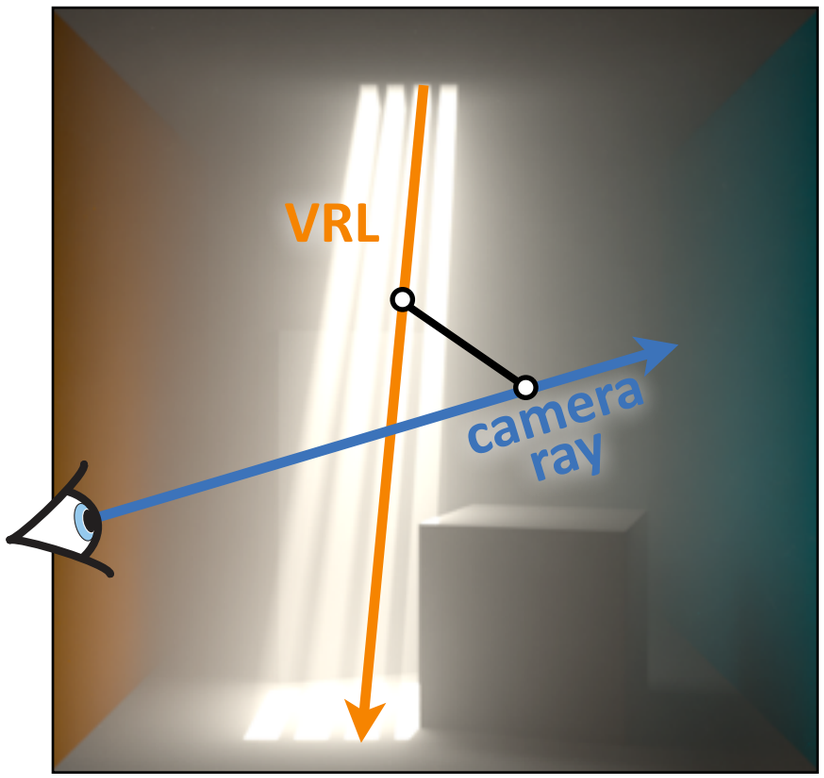
[Novák et al. 12a]

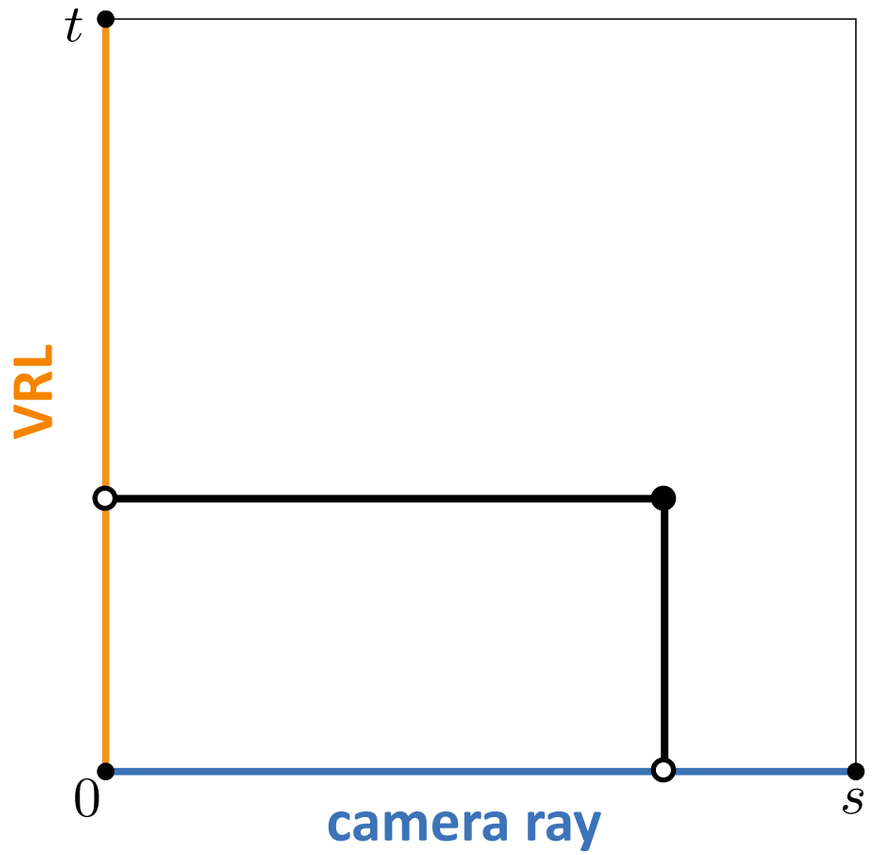


Virtual Ray Lights

[Novák et al. 12a]

$$L = \Phi \int_0^s \int_0^t \frac{f_p(\theta_u) f_p(\theta_v)\, \sigma_s(u) \sigma_s(v)\, T(u) T(v) T(w)\, V}{d(u,v)^2} dv du$$

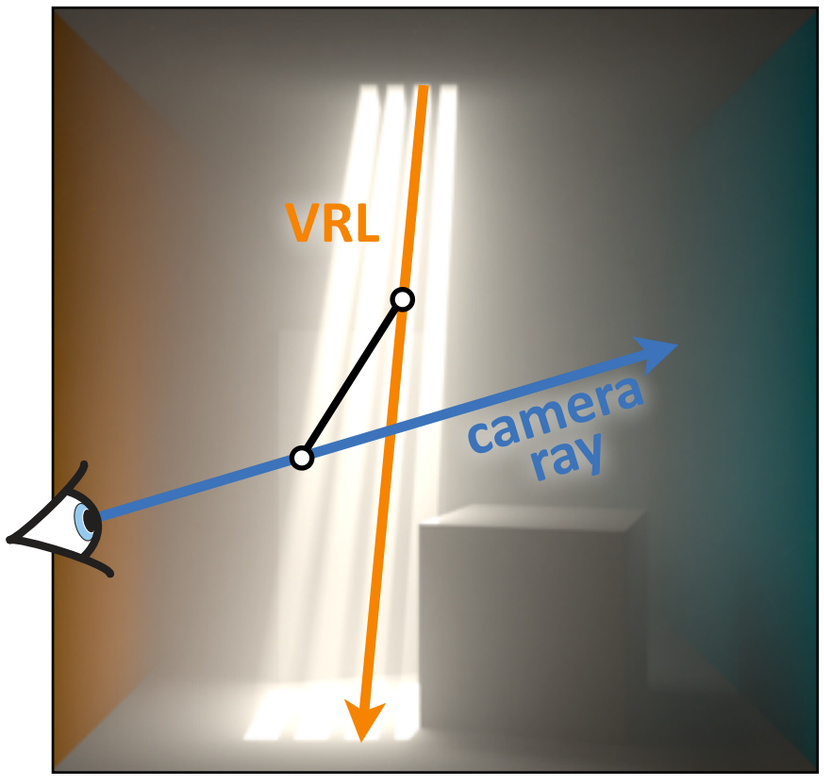


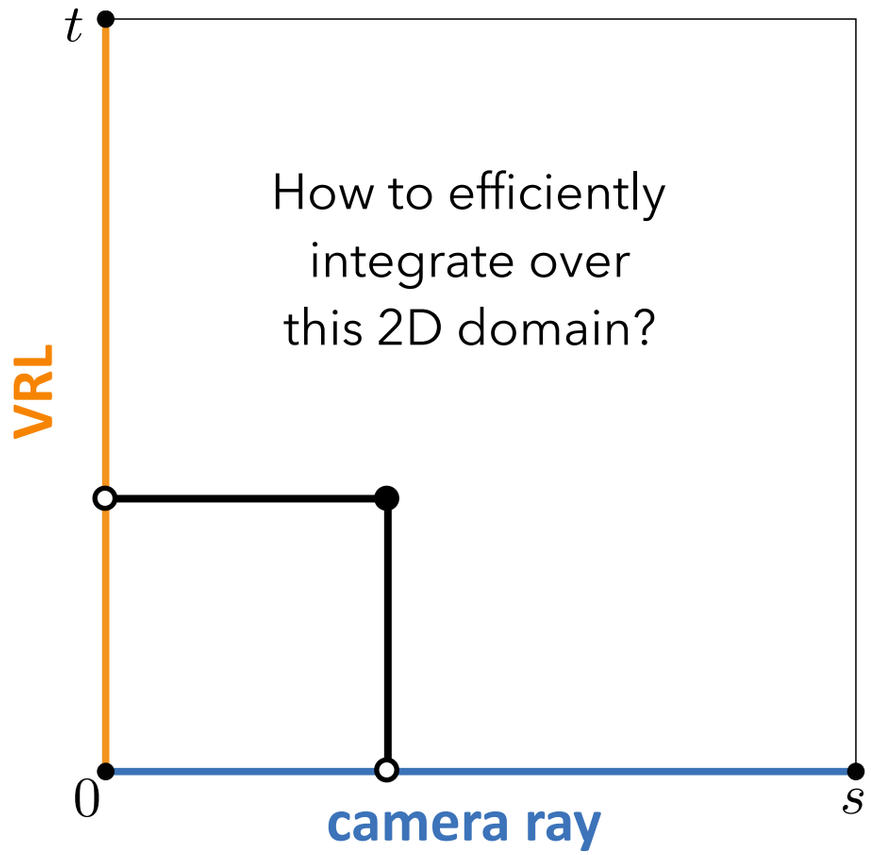


Virtual Ray Lights

[Novák et al. 12a]

$$L = \Phi \int_0^s \int_0^t \frac{f_p(\theta_u) f_p(\theta_v)\, \sigma_s(u) \sigma_s(v)\, T(u) T(v) T(w)\, V}{d(u,v)^2} dv du$$

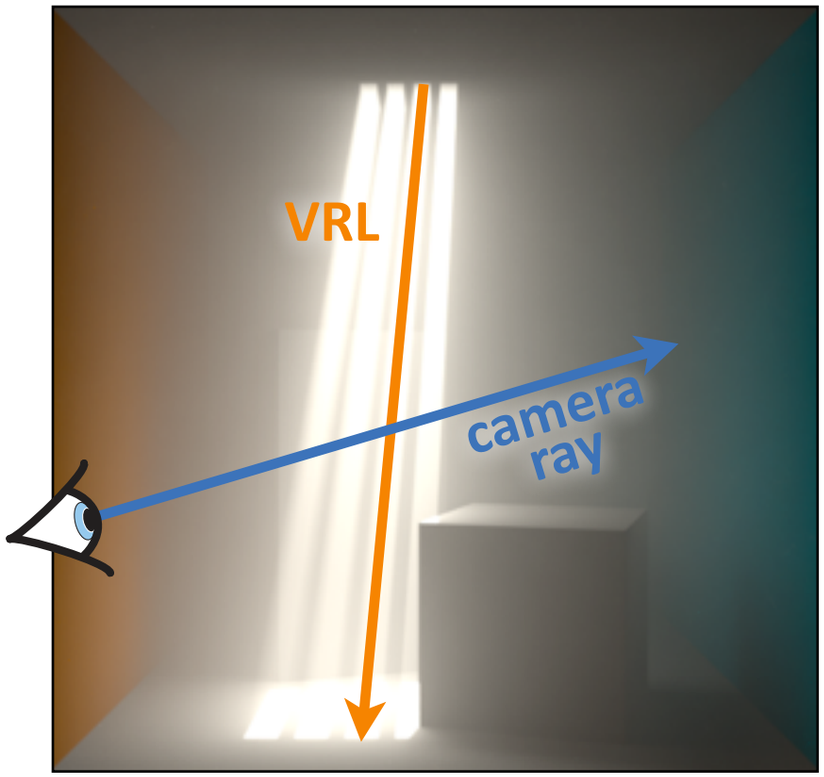




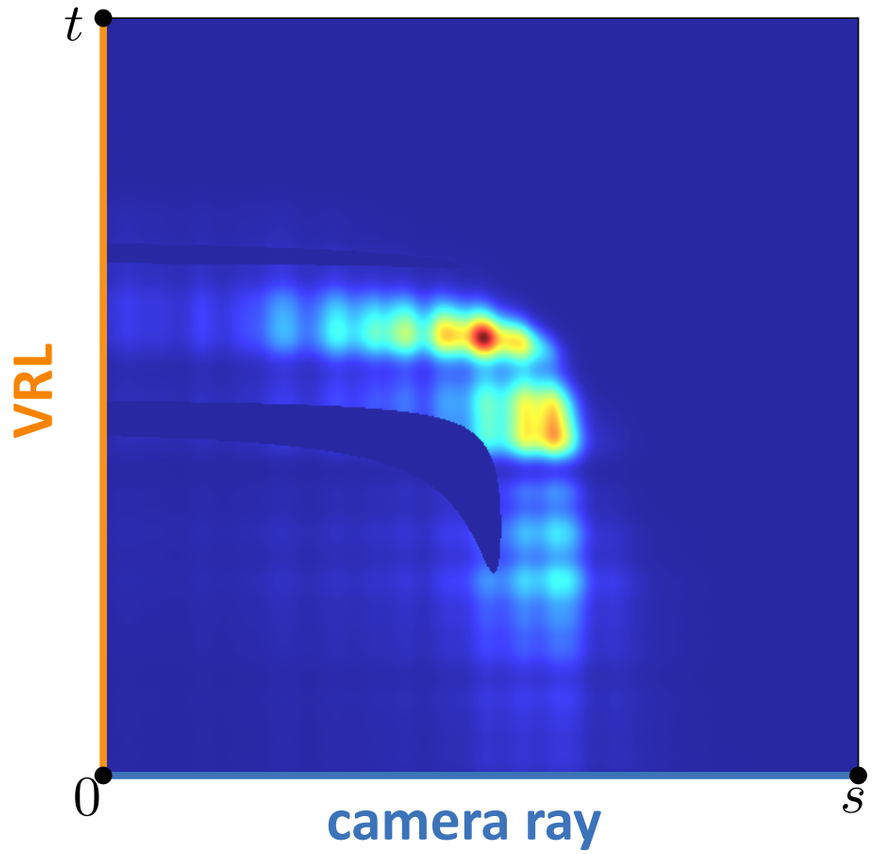
Virtual Ray Lights

[Novák et al. 12a]

$$L = \Phi \int_0^s \int_0^t \frac{f_p(\theta_u) f_p(\theta_v)\, \sigma_s(u) \sigma_s(v)\, T(u) T(v) T(w)\, V}{d(u,v)^2} dv du$$



Visualization of the integrand



Virtual Ray Lights

[Novák et al. 12a]

$$L = \Phi \int_0^s \int_0^t \frac{f_p(\theta_u) f_p(\theta_v)\, \sigma_s(u) \sigma_s(v)\, T(u) T(v) T(w)\, V}{d(u,v)^2} dv du$$

- Importance sampling:
  - Combine equiangular sampling with importance sampling of the two phase functions
  - Semi-analytic procedure, greatly reduces noise

Comparison

Fruit Juice

homogeneous

anisotropic (HG g = 0.55)

512x512

Comparison

Surface illumination

(Photon Mapping)

Single scattering

(Photon Beams)

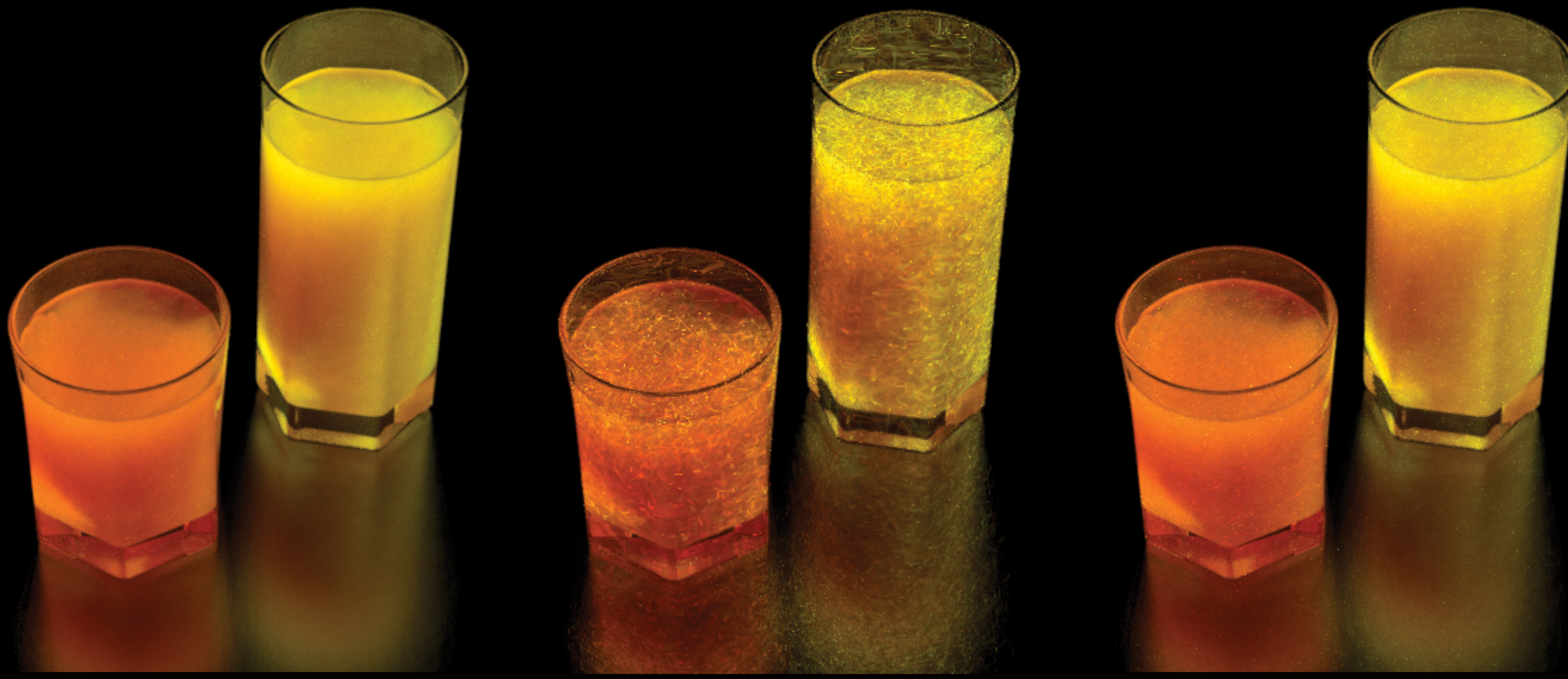
Multiple scattering

Multiple Scattering Only

Virtual Ray Lights

Progressive Photon Beams

Virtual Point Lights



Temporal Stability

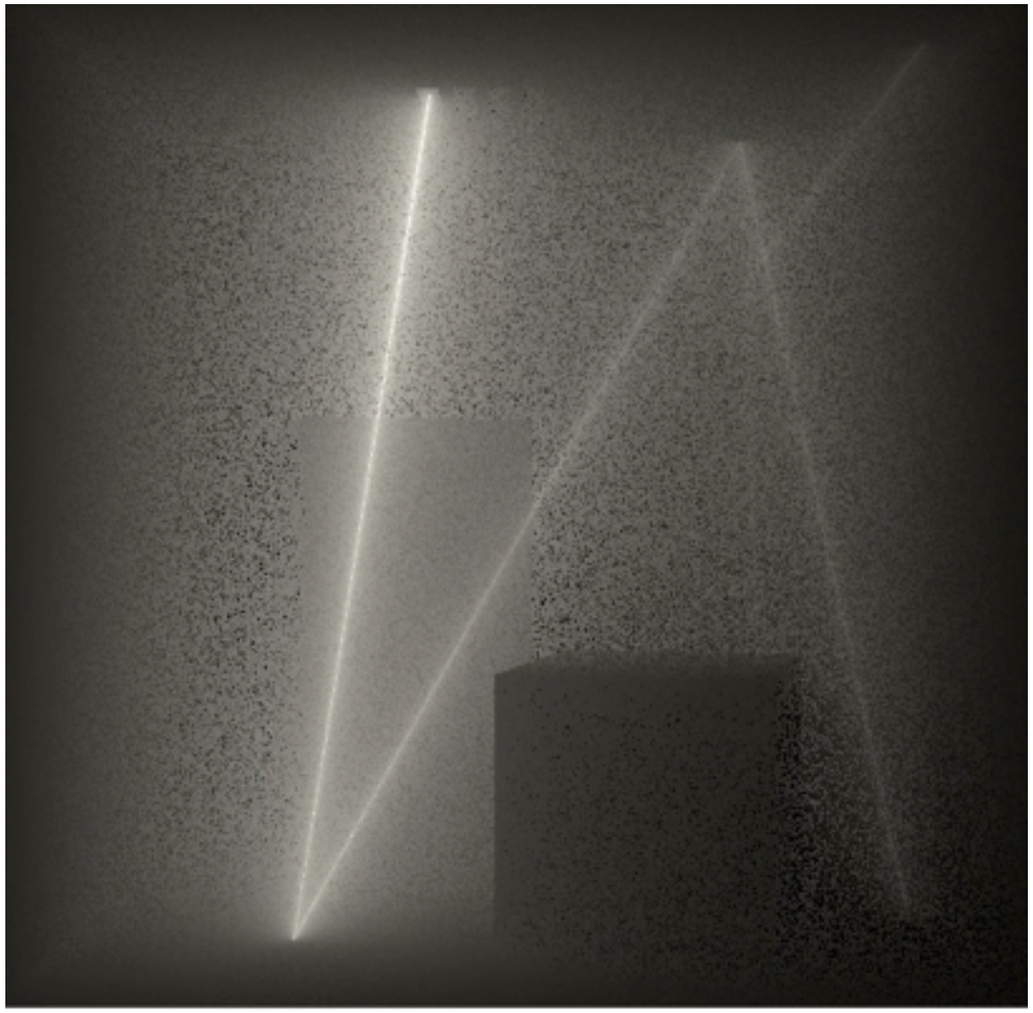
Virtual Ray Lights

1 minute/frame

Virtual Point Lights

1 minute/frame

Can we remove the singularity completely?

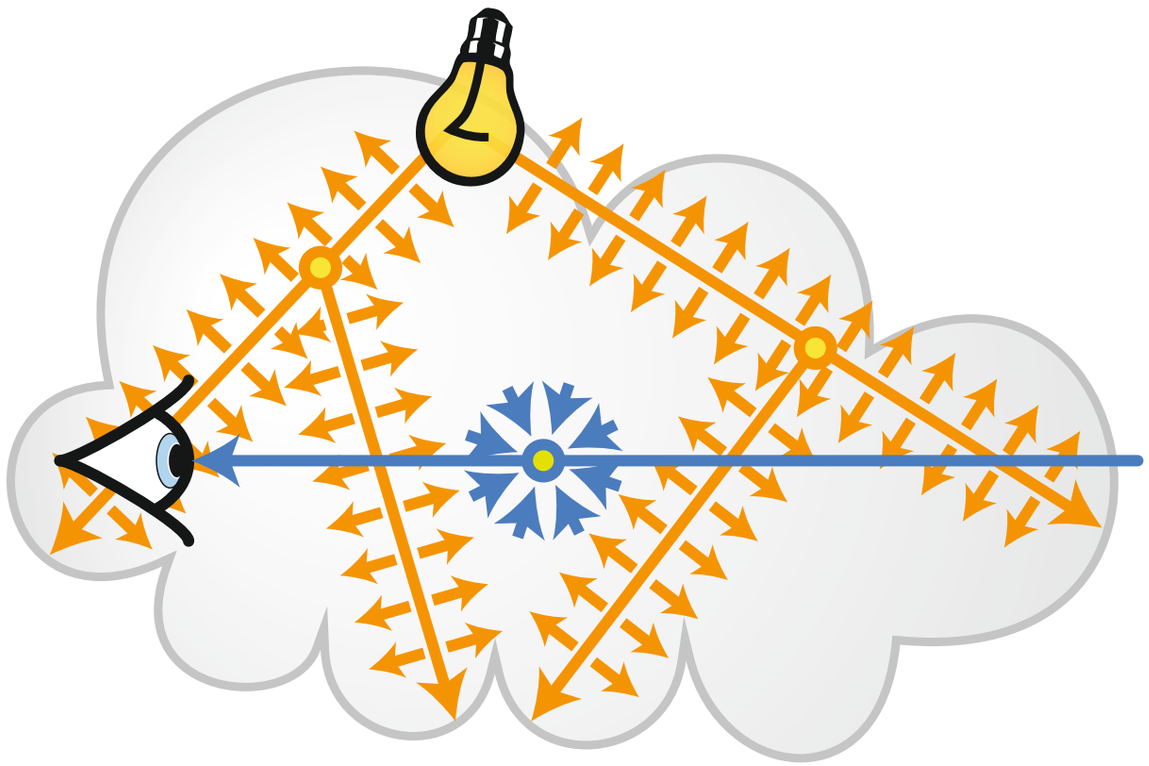


6 VRLs

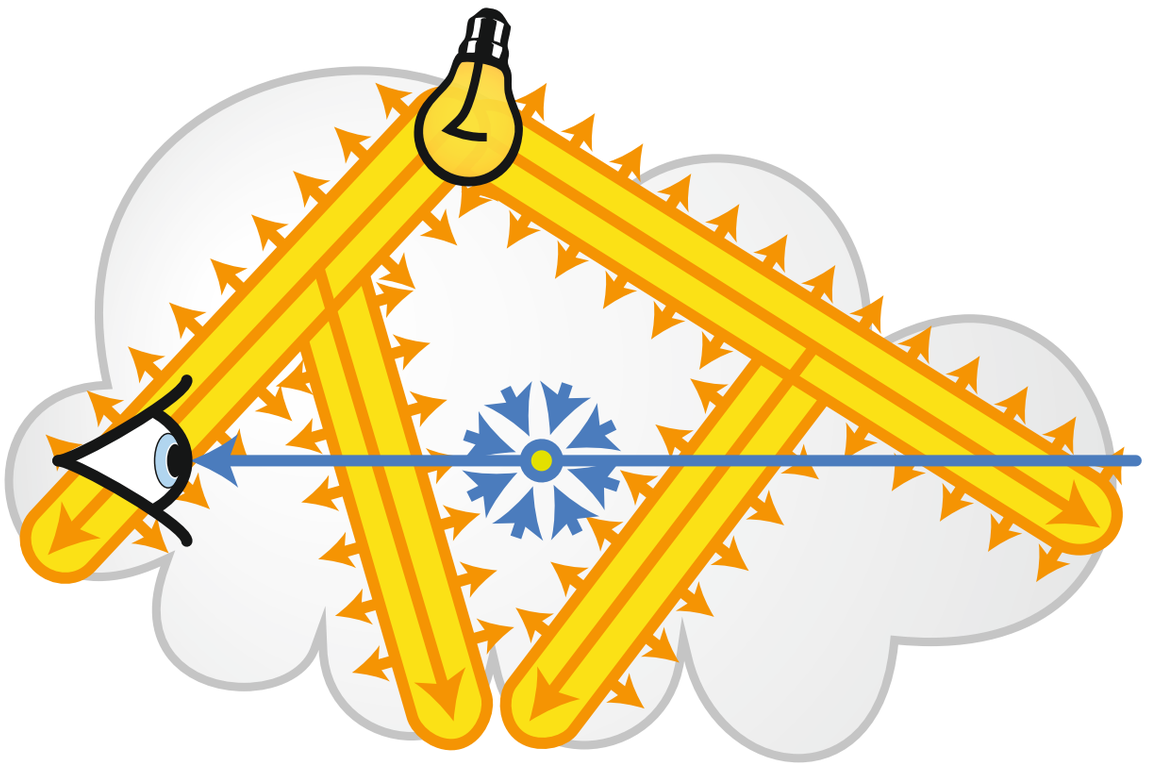
Virtual Beam Lights

[Novák et al. 12b]

- "Spread" energy spatially
  - Finite-thickness beam instead of infinitesimal ray
  - Similar concept to virtual spherical lights



Virtual Ray Lights

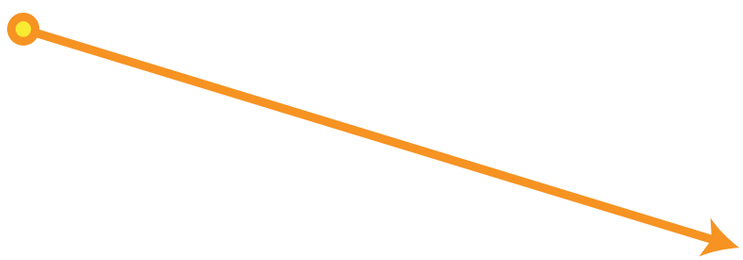


Virtual Beam Lights

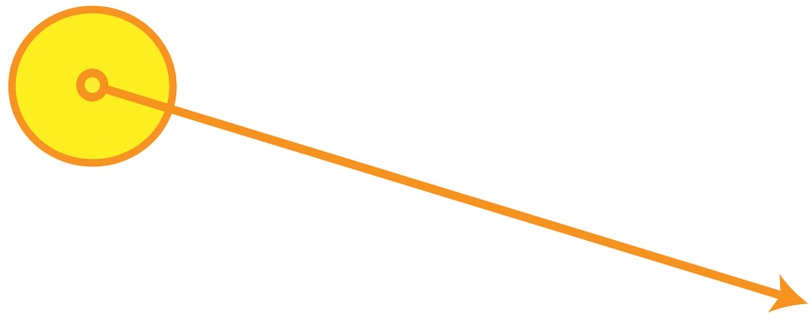
Ray Light vs Beam Light

[Novák et al. 12b]

Virtual Ray Light



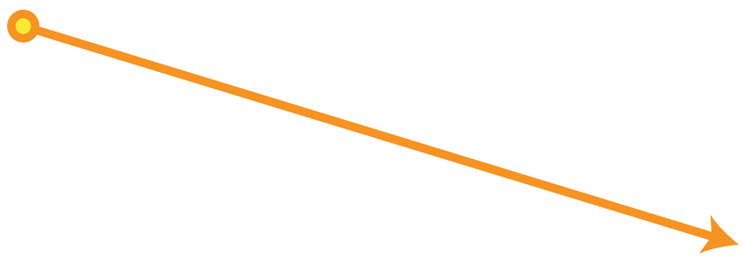
Virtual Beam Light



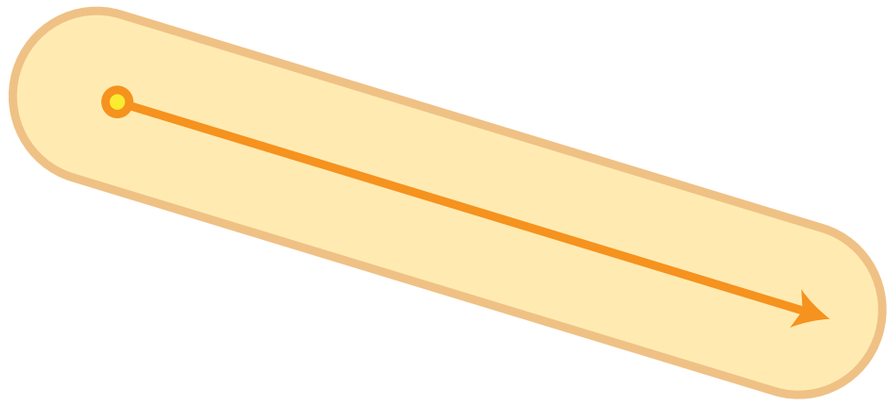
Ray Light vs Beam Light

[Novák et al. 12b]

Virtual Ray Light



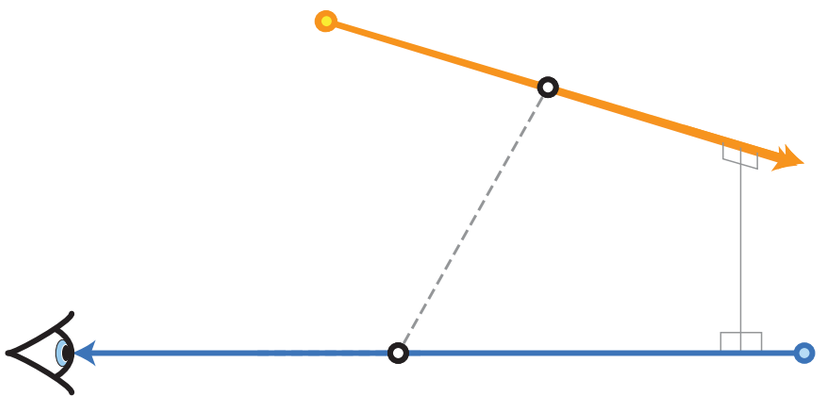
Virtual Beam Light



Ray Light vs Beam Light

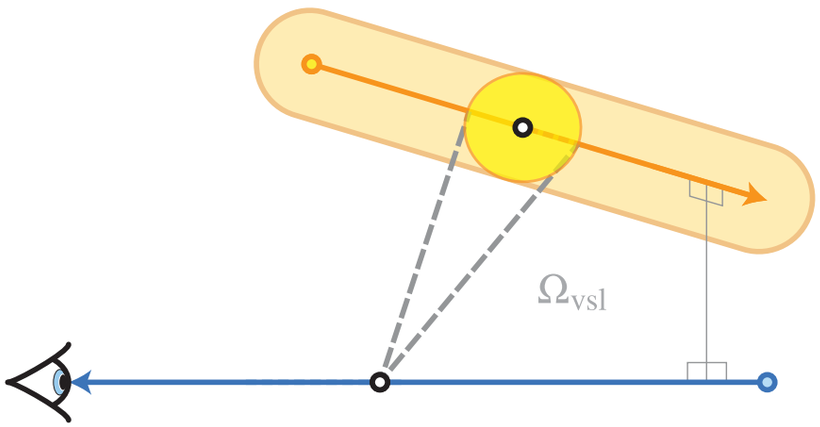
[Novák et al. 12b]

Virtual Ray Light



$$L = \Phi \int_0^s \int_0^t \frac{f_p(\theta_u) f_p(\theta_v)\, \sigma_s(u) \sigma_s(v)\, T(u) T(v) T(w)}{d(u,v)^2} dv du$$

Virtual Beam Light

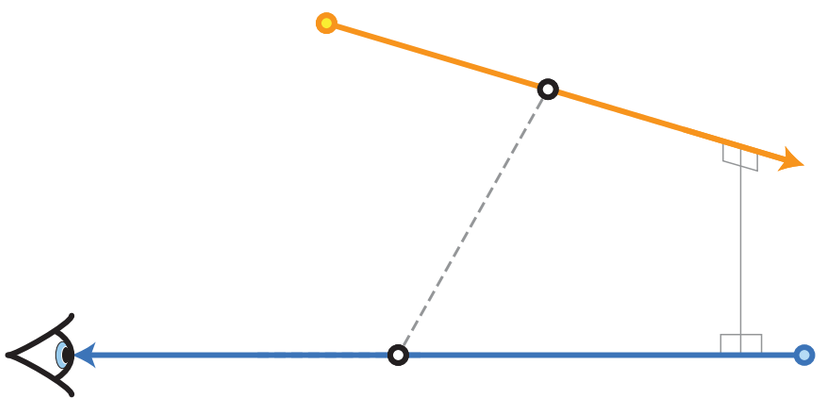


$$L = \frac{\Phi}{\pi r^2} \iint \sigma_s(u) T(u) \int_{\Omega_{\text{VSL}}} f_p(\theta_u) f_p(\theta_v) \sigma_s(v) T(w) T(v)\, \mathrm{d}\omega\ dv du$$

Ray Light vs Beam Light

[Novák et al. 12b]

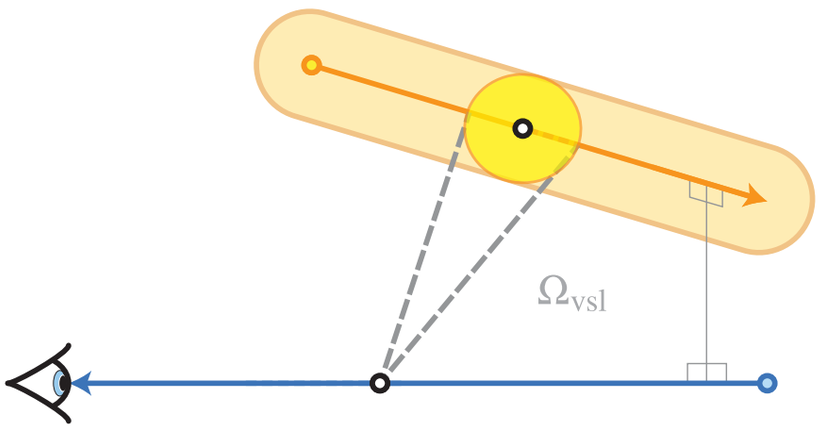
Virtual Ray Light



$$L = \Phi \int_0^s \int_0^t \frac{f_p(\theta_u) f_p(\theta_v)\, \sigma_s(u) \sigma_s(v)\, T(u) T(v) T(w)}{d(u,v)^2} dv du$$

We are avoiding the singularity!

Virtual Beam Light



$$L = \frac{\Phi}{\pi r^2} \iint \sigma_s(u) T(u) \int_{\Omega_{\text{VSL}}} f_p(\theta_u) f_p(\theta_v) \sigma_s(v) T(w) T(v)\, \mathrm{d}\omega\, dv du$$

Cars Scene

homogeneous

isotropic

1280x720

Media-to-Media Transport

VRLs

0.0K VRLs

VBLs

0.0K VBLs

16 seconds

Buddha Scene

homogeneous

anisotropic (HG g= 0.7)

720x720

Media-to-Media Transport

VRLs

0.3K VRLs

VBLs

0.3K VBLs

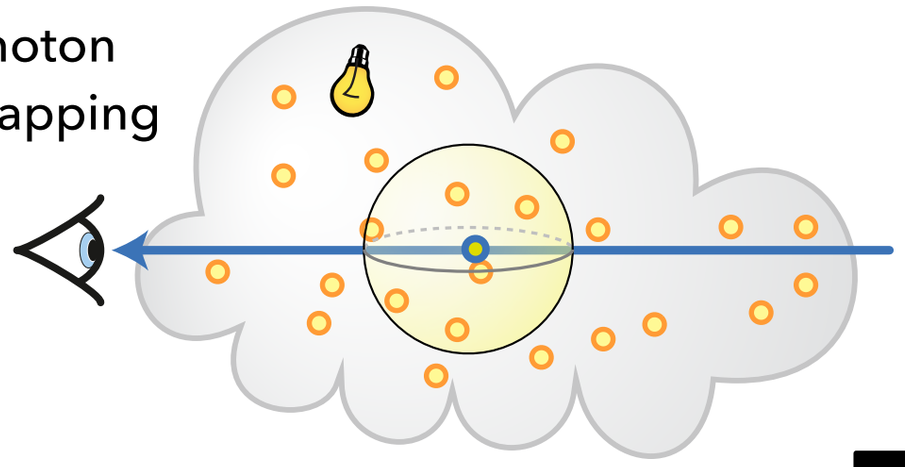
26 seconds

Volumetric Many-Light Rendering

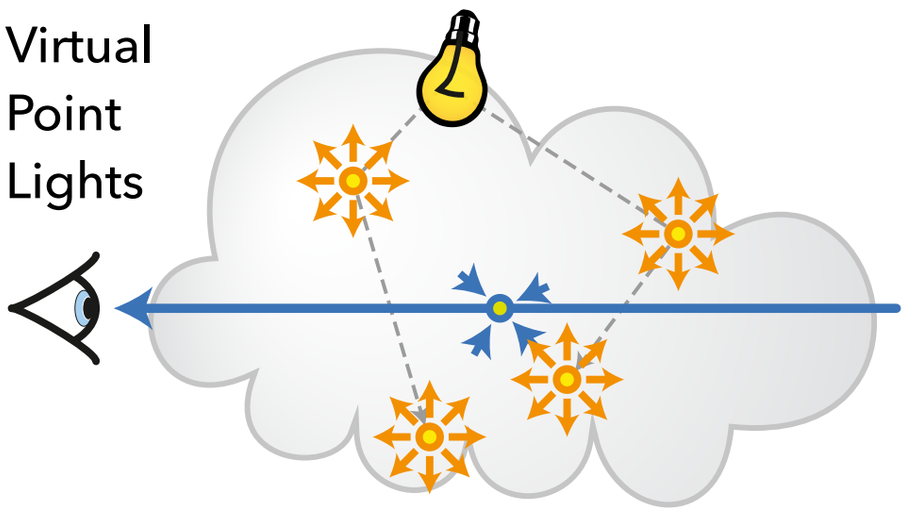
- Volumetric Virtual Point Lights
  - Trivial extension of surface VPLs
  - Splotches more severe than on surfaces
- Virtual Ray Lights
  - Singularity is reduced by spreading along rays
- Virtual Beam Lights
  - Singularity is avoided at the cost of bias
  - Shrink radius too as in progressive photon beams

Care about efficiency?

Volumetric Photon Mapping

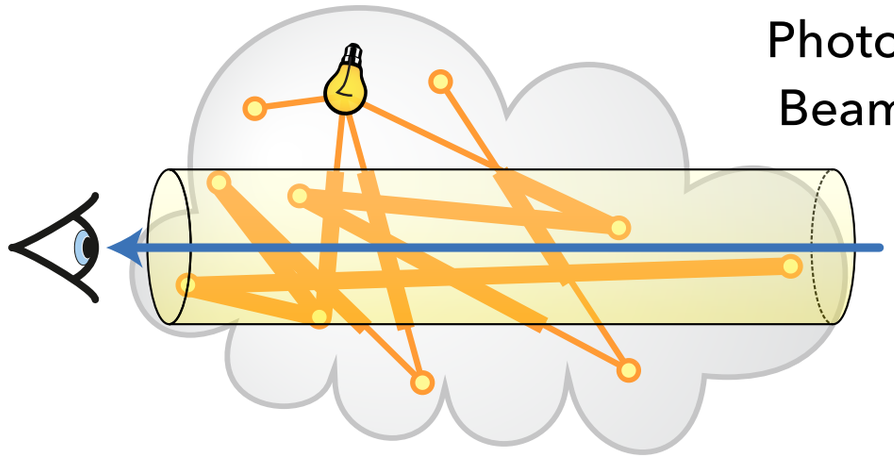


Virtual Point Lights

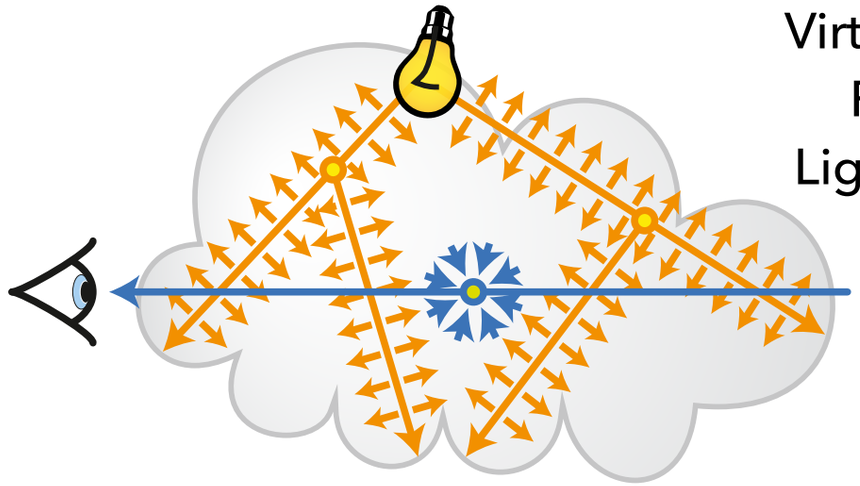


Use entire segments!

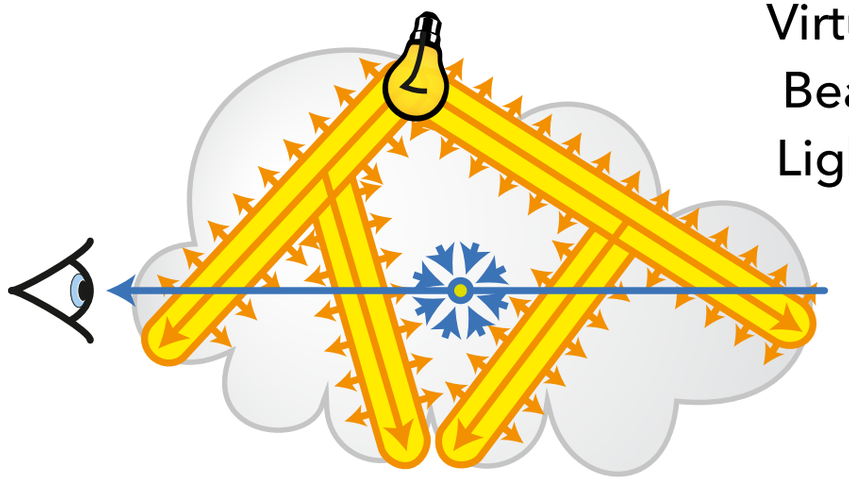
Photon Beams



Virtual Ray Lights



Virtual Beam Lights



Next time