Computer Graphics

- Volume Rendering -

Pascal Grittmann

Philipp Slusallek

Using pictures from: **Monte Carlo Methods for Physically Based Volume Rendering**; SIGGRAPH 2018 Course; Jan Novák, Iliyan Georgiev, Johannes Hanika, Jaroslav Křivánek, Wojciech Jarosz

Overview

- So far:
  - Light interactions with surfaces
  - Assume vacuum in and around objects
- This lecture:
  - Participating media
  - How to represent volumetric data
  - How to compute volumetric lighting effects
  - How to implement a very basic volume renderer

Fog & clouds

Steve Lacey

Underwater

source: dailypictures.info

Surface or volume?

source: Flickr

source: Studio Lernert & Sander

Avatar. Copyright © 2009 20th Century Fox

Arrival. Copyright © 2016 Paramount Pictures

Big Hero 6. Copyright © 2014 Walt Disney Enterprises, Inc.

Mortal Engines. Copyright © 2018 Universal Studios

Fundamentals

Volumetric Effects

- Light interacts not only with surfaces but everywhere inside!
- Volumes scatter, emit, or absorb light

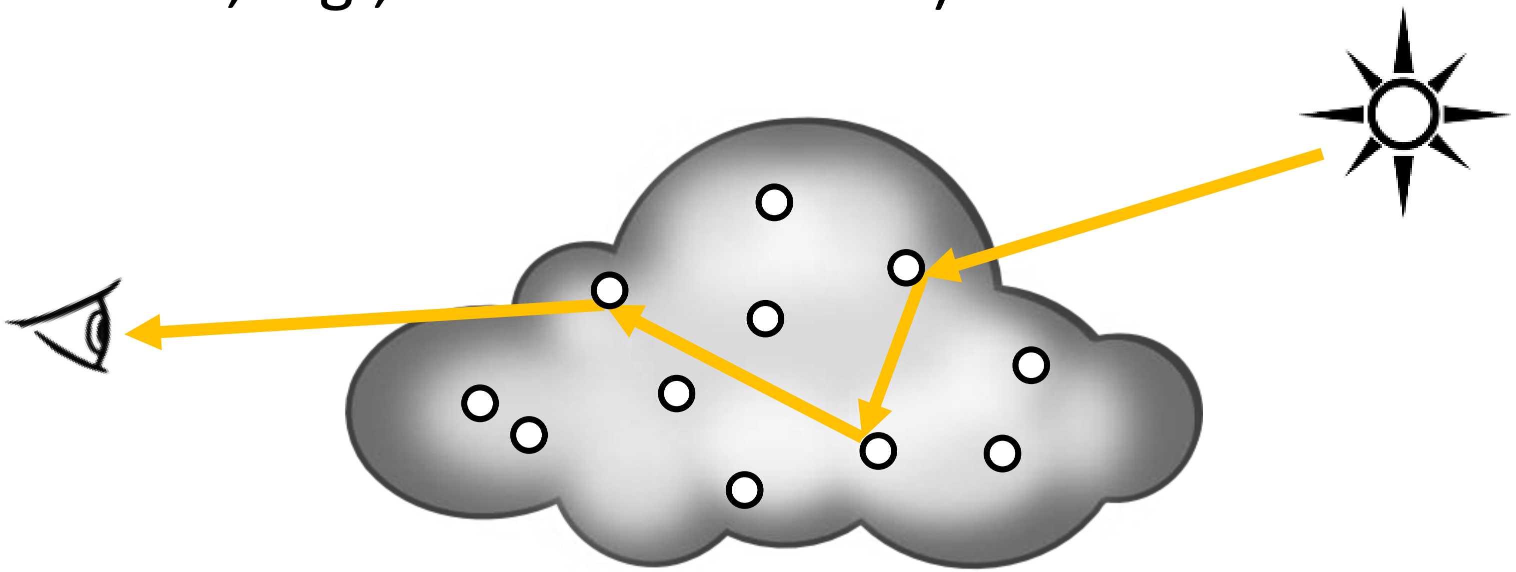
http://coclouds.com

http://wikipedia.org

http://commons.wikimedia.org

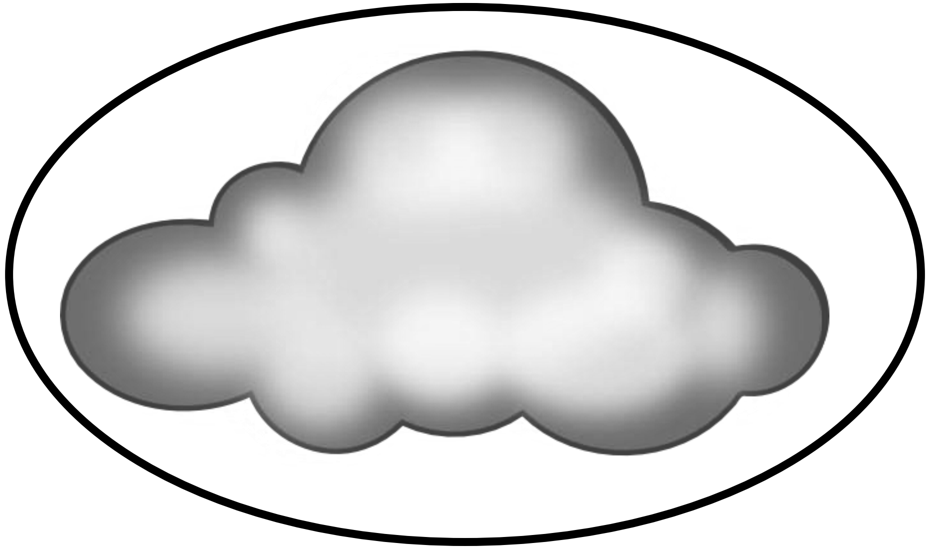
Approximation: Model Particle Density

- Modeling individual particles of a volume is, of course, not practical
- Instead, represent statistically using the average density
- (Same idea as, e.g., microfacet BSDFs)



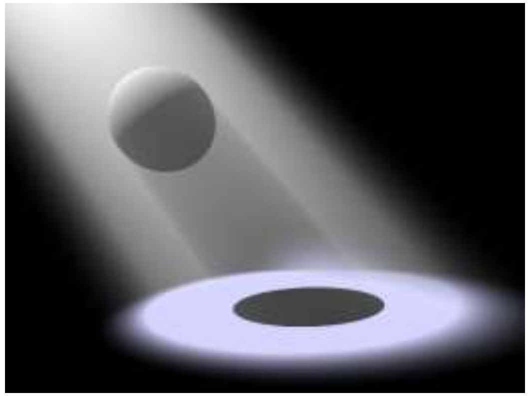
Volume Representation

- Many possibilities (particles, voxel octrees, procedural,…)
- A common approach: Scene objects can “contain” a volume



Volume Representation

- Homogeneous:
  - Constant density
  - Constant absorption, scattering, emission,
  - Constant phase function (later)



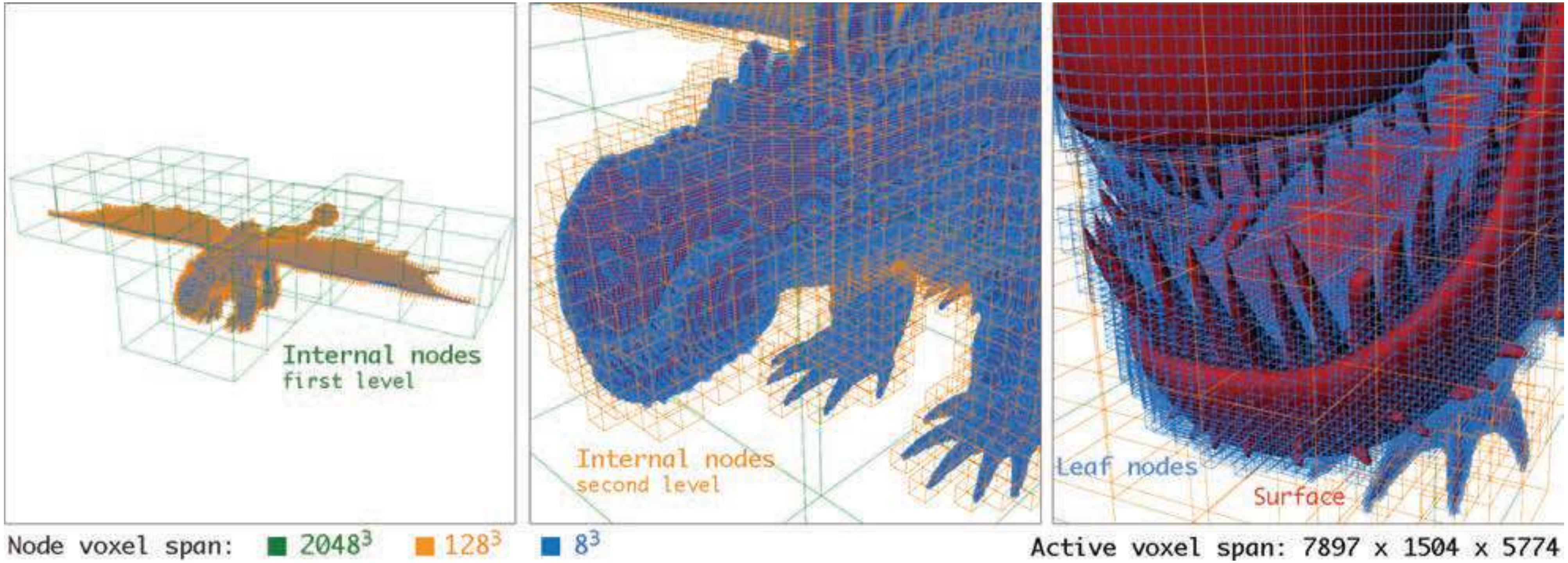
- Heterogeneous:
  - Coefficients and/or phase function vary across the volume
  - Can be represented using **3D textures**
  - (e.g., voxel grid, procedural)

http://wikipedia.org

Representation Example: OpenVDB

[K. Museth, 2013]

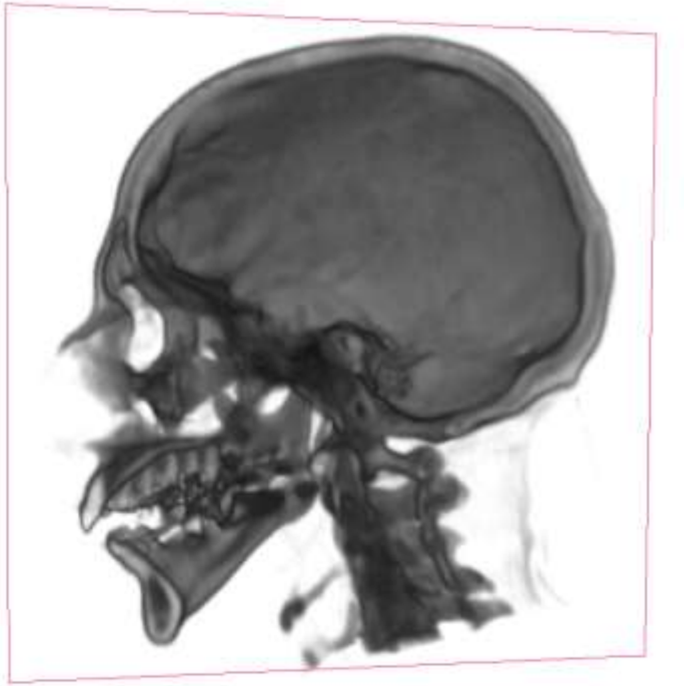
- Open source library
- Manages volume data
- Discretized, sparse, hierarchical grid



Data Acquisition Examples

- Real-world measurements via tomography
- Simulation, e.g.,
  - Fluids,
  - Fire and smoke,
  - Fog

https://docs.blender.org

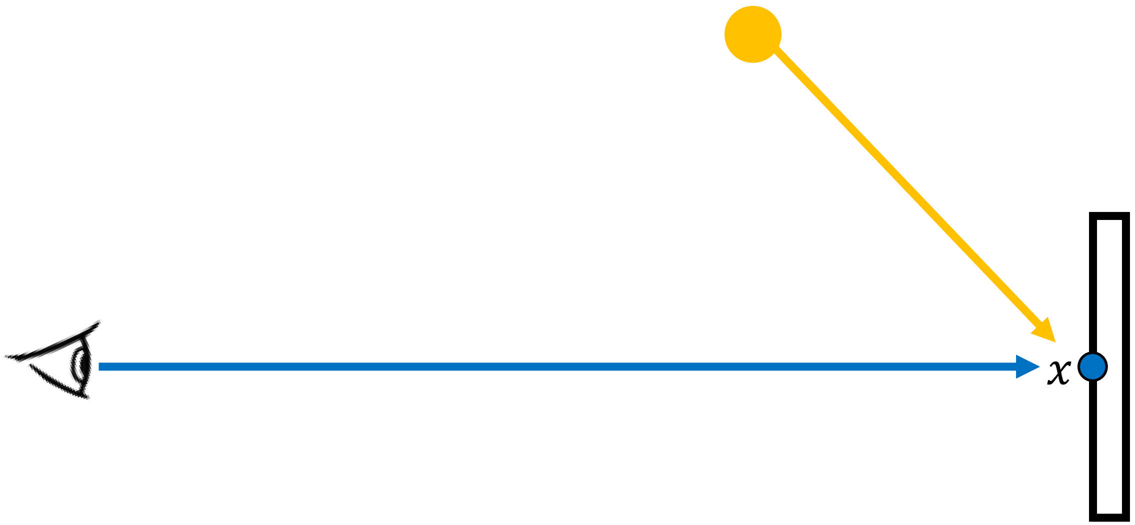


Simulating Volumes

Mathematical Formulation of Volumetric Light Transport

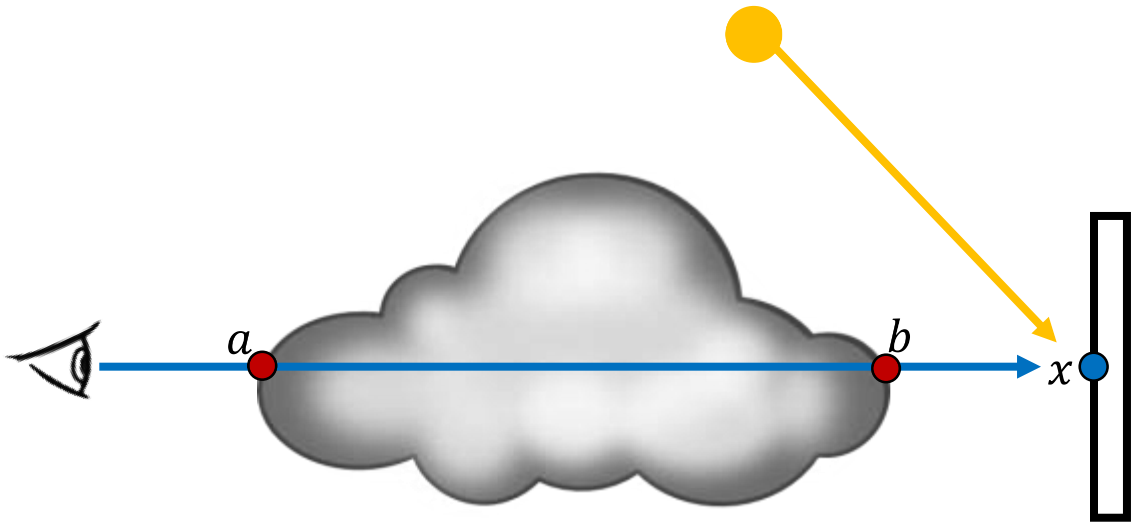
So far: Assume Vacuum

- Compute $L_o(x, \omega_o)$ using the rendering equation



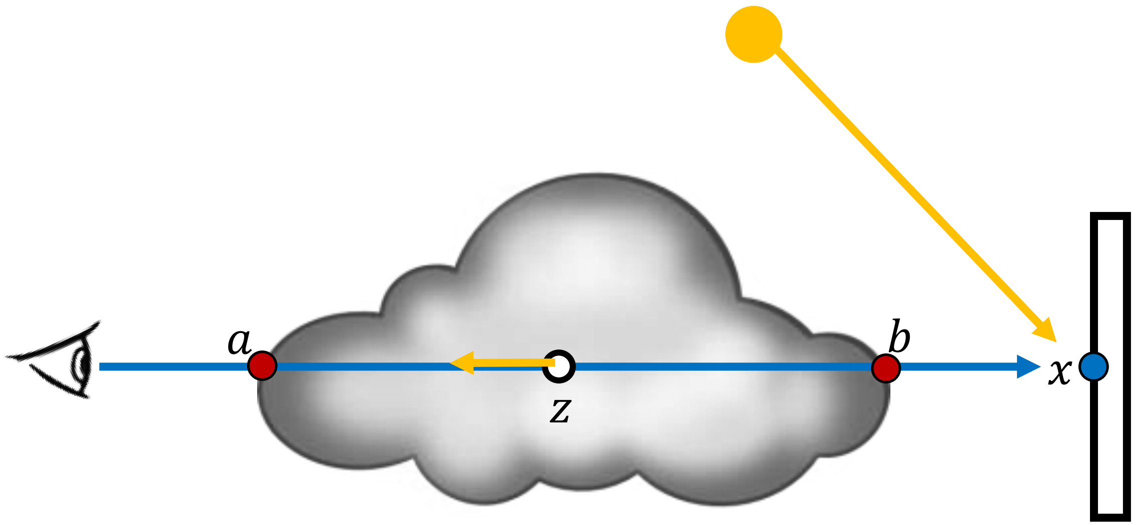
Volume Absorbs and Scatters Light

- Compute $L_o(x, \omega_o)$ using the rendering equation
- Only a fraction $T(a, b)L_o(x, \omega_o)$ arrives at the eye



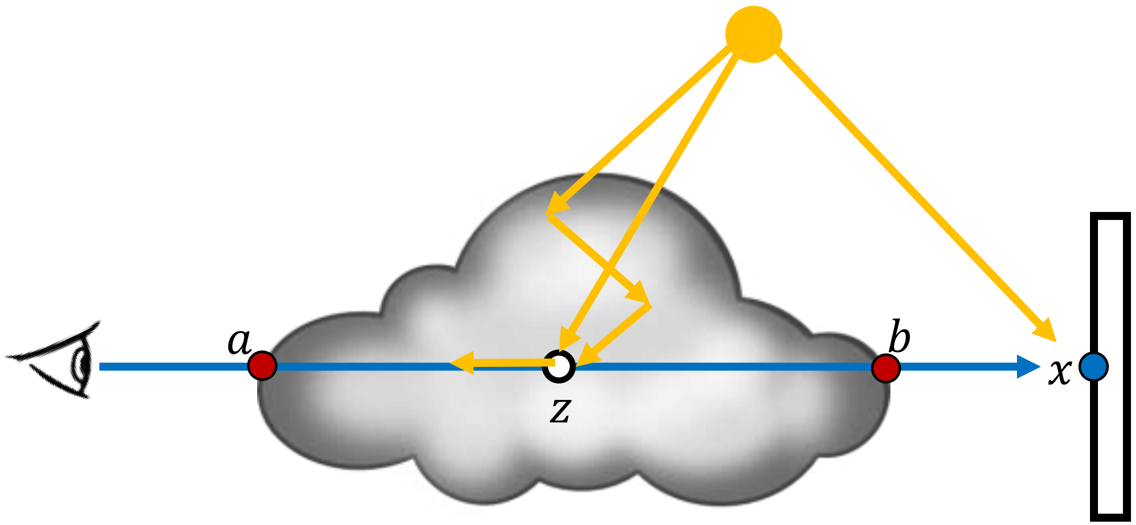
Volume Emits Light

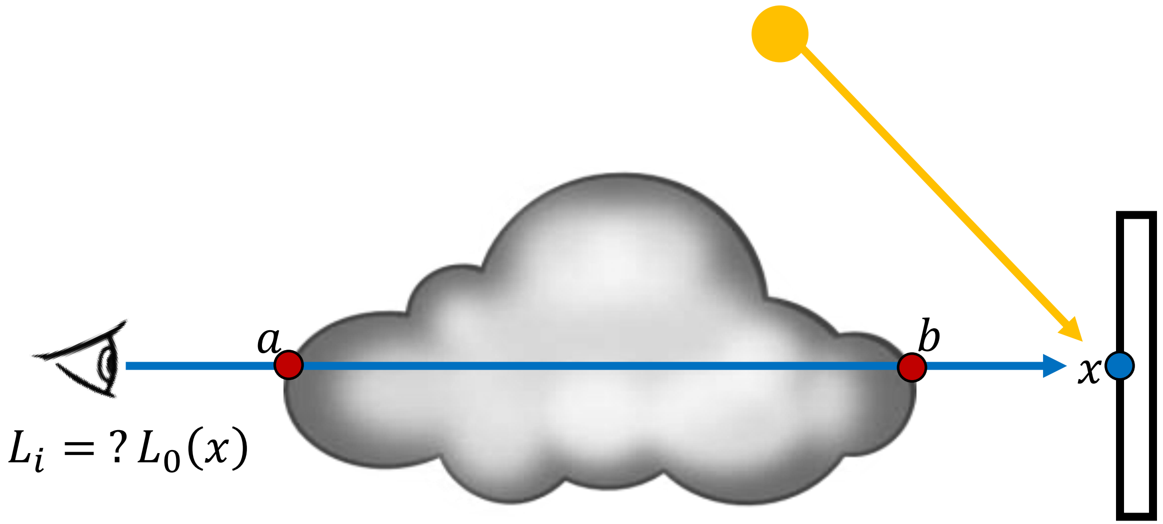
- Compute $L_o(x, \omega_o)$ using the rendering equation
- Only a fraction $T(a, b)L_o(x, \omega_o)$ arrives at the eye
- Every point z between a and b might emit light



Volume Scatters Light

- Compute $L_o(x, \omega_o)$ using the rendering equation
- Only a fraction $T(a, b)L_o(x, \omega_o)$ arrives at the eye
- Every point z between a and b might emit light
- Every point z might be illuminated through the volume





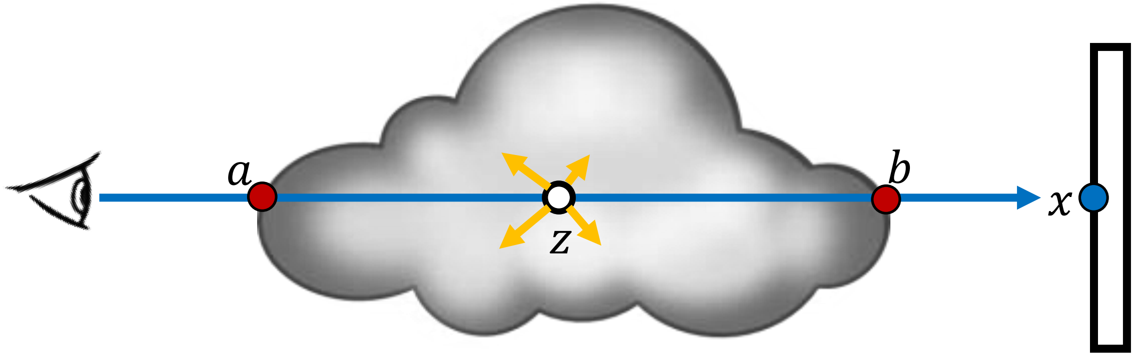
http://commons.wikimedia.org

Attenuation

Computing Absorption and Out-Scattering

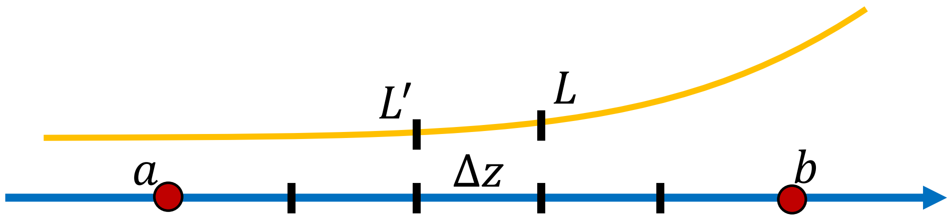
Attenuation = Absorption + Out-Scattering

- Every point in the volume might absorb light or scatter it in other directions
- Modeled by absorption and scattering densities: $\mu_a(z)$ and $\mu_s(z)$
- Might depend on position, direction, time, wavelength,…
- For simplicity: we assume only positional dependence



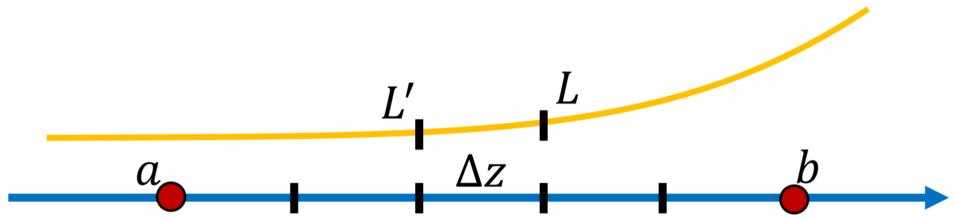
Computing Absorption – Intuition

- Consider a small segment Δz
- Along that segment, radiance is reduced from L to L'
- $L' = L - \mu_a \, L \, \Delta z$
- Where μ_a is the percentage of radiance that is absorbed (per unit distance)



Computing Absorption – Intuition

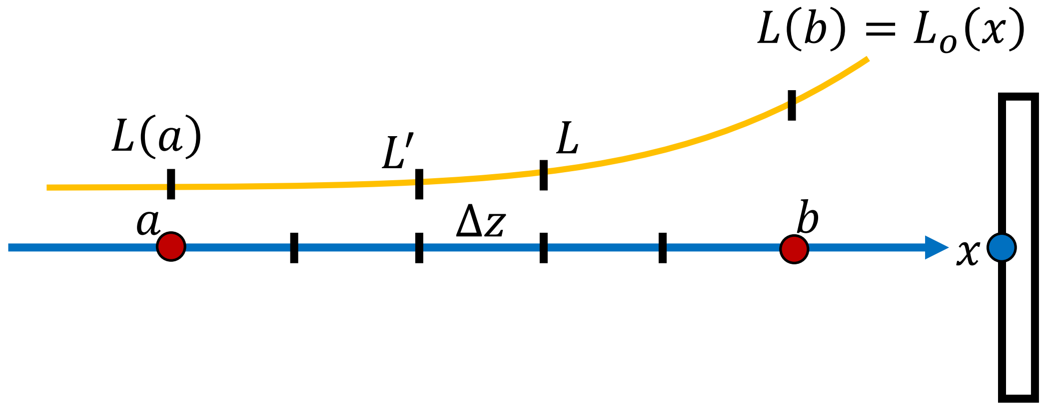
- Consider a small segment Δz
- Along that segment, radiance is reduced from L to L'
- $L' = L - \mu_a \, L \, \Delta z$
- Where μ_a is the percentage of radiance that is absorbed (per unit distance)



So the absorbed radiance is: $\Delta L = L' - L = -\mu_a \, L \, \Delta z$

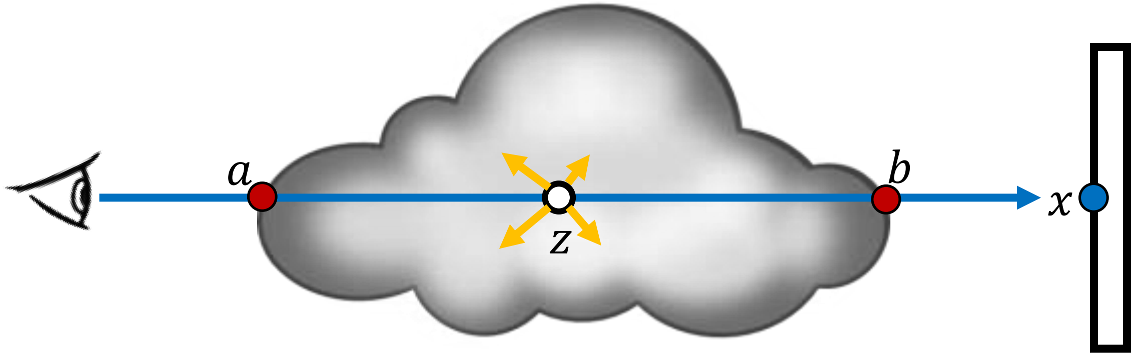
Computing Absorption – Exponential Decay

- $\Delta L = -\mu_a \, L \, \Delta z$
- For infinitely small Δz, this becomes
- $dL = -\mu_a \, L \, dz$
- A differential equation that models exponential decay!
- Solution: $L(a) = L_o(x)\, e^{-\int_b^a \mu_a(t)\, dt}$



Computing Out-Scattering

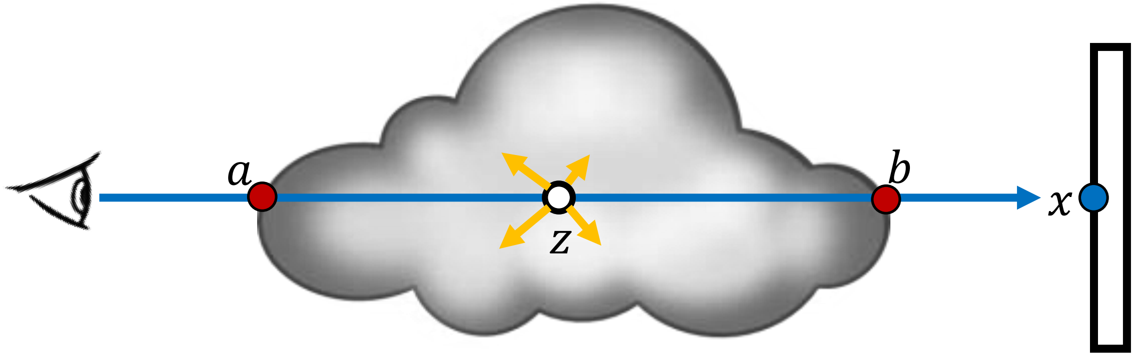
- Same as absorption, only different factor!
- $\mu_s(z)$: percentage of light scattered at point z
- $L(a) = L_o(x)\, e^{-\int_b^a \mu_s(t)\, dt}$



Computing Attenuation

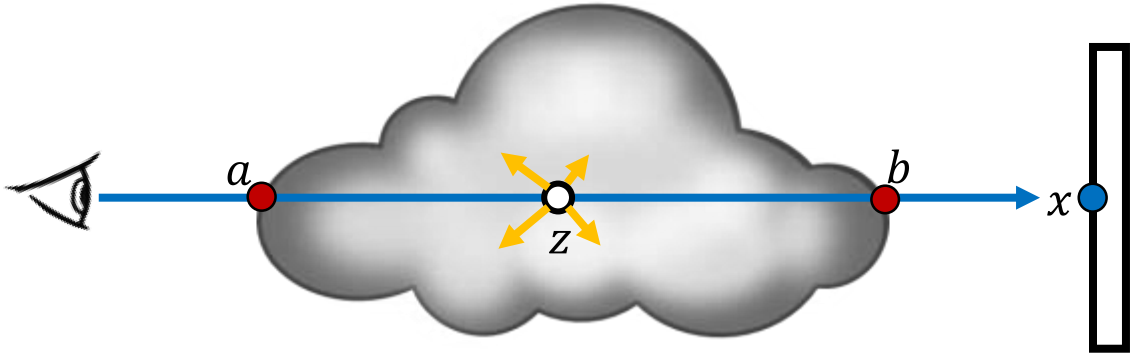
- Fraction of light that is neither absorbed nor out-scattered
- $\mu_t = \mu_a + \mu_s$
- $L(a) = L_o(x)\ e^{-\int_b^a (\mu_a(t)+\mu_s(t))\ dt}$

- Attenuation: $T(a,b) = e^{-\int_b^a \mu_t(t)\ dt}$



Computing Attenuation – Homogeneous

- Simple case: constant density / attenuation
  - $\mu_t(z) = \mu_t \quad \forall z$
- $T(a,b) = e^{-\int_b^a \mu_t(t)\,dt} = e^{-(a-b)\mu_t}$

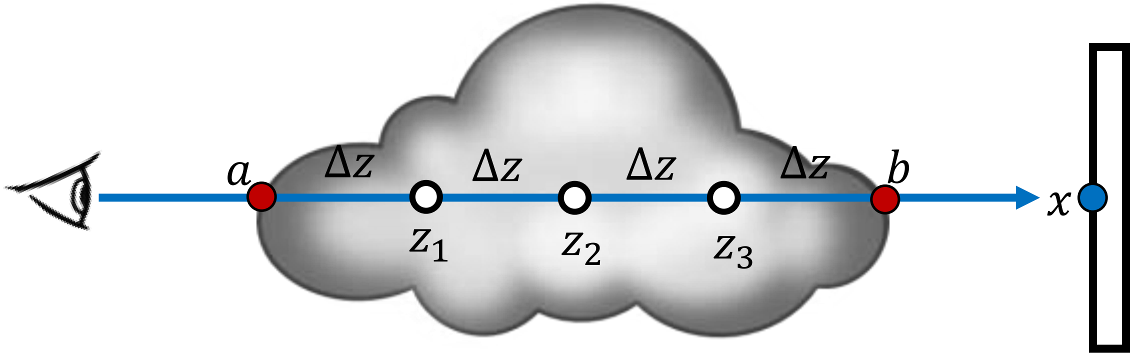


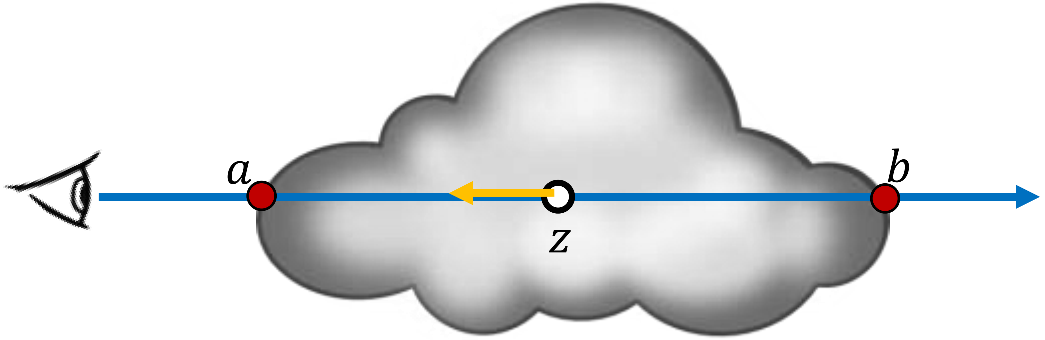
Estimating Attenuation

- We need to solve another integral:
  - $T(a,b) = e^{-\int_b^a \mu_t(t)\,dt}$
- Many solutions, e.g., Monte Carlo integration (next semester)
- Simple solution: Quadrature

Estimating Attenuation – Ray Marching

- We need to solve another integral:
  - $T(a,b) = e^{-\int_b^a \mu_t(t)\,dt}$
- Simple solution: Quadrature
- Ray marching: evaluate at discrete positions (fixed stepsize Δz)
- $\int_b^a \mu_t(t)\,dt \approx \sum_i \mu_t(z_i)\,\Delta z$





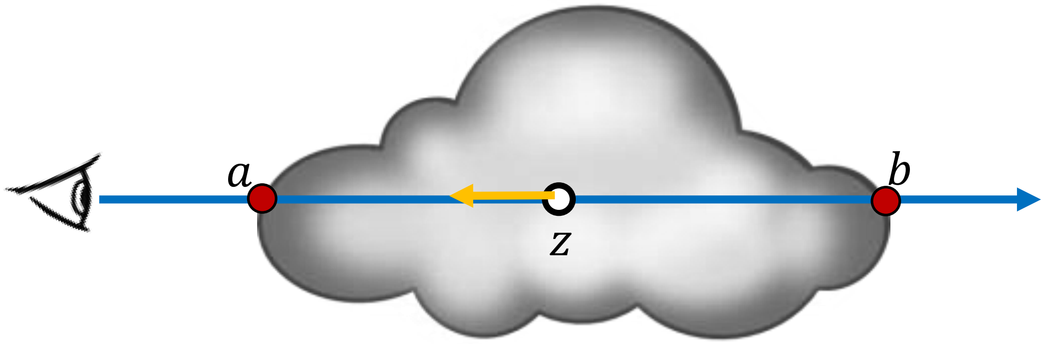
Emission

Explosions!

http://wikipedia.org

Every Point Might Emit Light

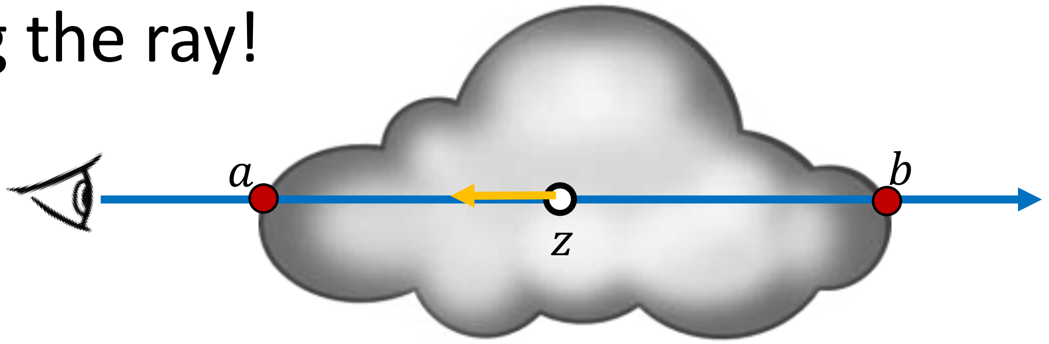
- Assume z emits $L_e(z)$ towards a
- Some of that light might be absorbed or out-scattered: It is attenuated
- $L(a) = L_e(z)\, T(z, a)$



Every Point Might Emit Light

- Assume z emits $L_e(z)$ towards a
- Some of that light might be absorbed or out-scattered: It is attenuated
- $L(a) = L_e(z)\, T(z, a)$

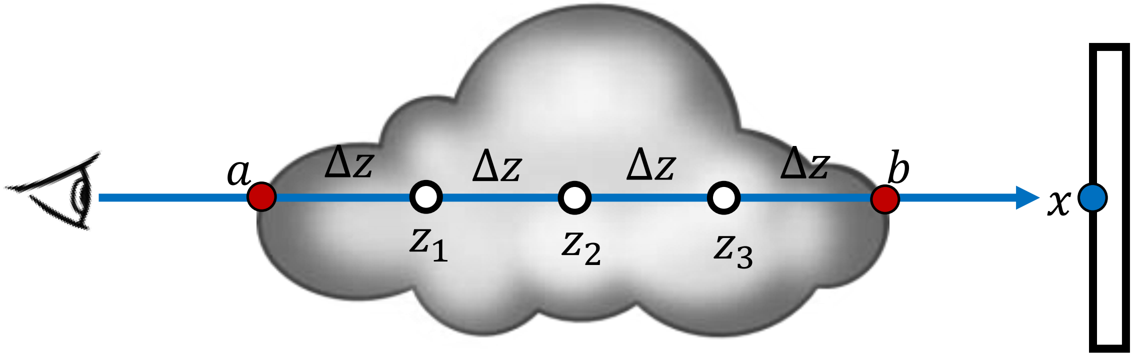
- Happens at every point along the ray!
- $L(a) = \int_a^b L_e(z)\, T(z, a)\, dz$



- Another integral…

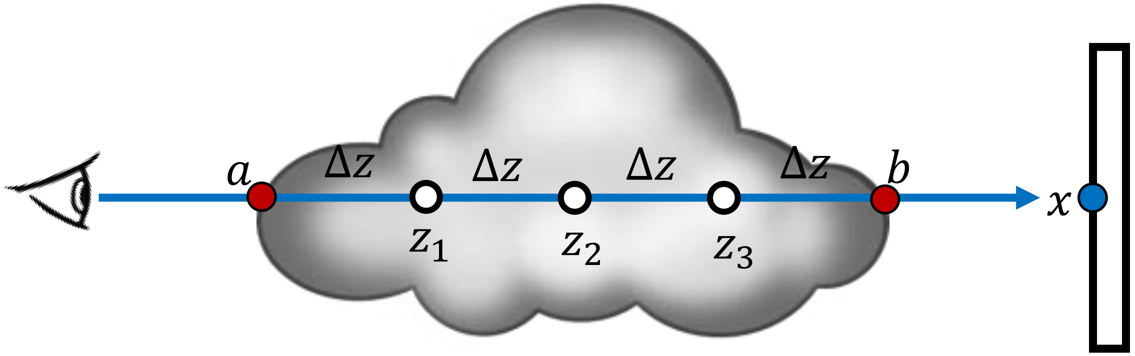
Ray Marching for Emission

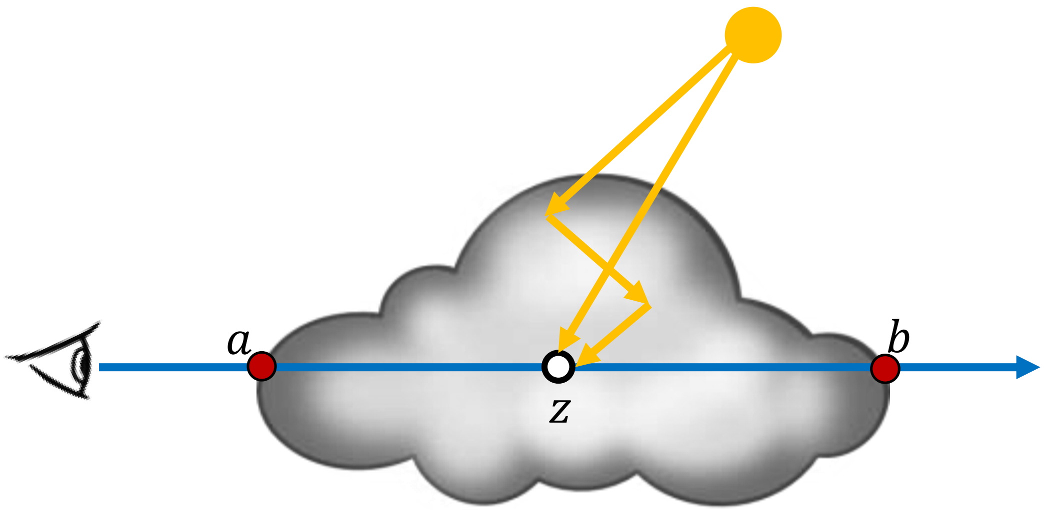
- Same as before: integrate via quadrature
- $\int_a^b L_e(z)\, T(z,a)\, dz \approx \sum_i L_e(z_i)\, T(z_i,a)\, \Delta z$
- Attenuation $T(z_i, a)$ estimated as before



Ray Marching for Emission

- Same as before: integrate via quadrature
- $\int_a^b L_e(z)\, T(z,a)\, dz \approx \sum_i L_e(z_i)\, T(z_i,a)\, \Delta z$
- Attenuation $T(z_i, a)$ estimated as before
- Attenuation can be incrementally updated:
  - $T(z_i, a) = T(z_{i-1}, a)\, T(z_i, z_{i-1})$
  - (because it is an exponential function)



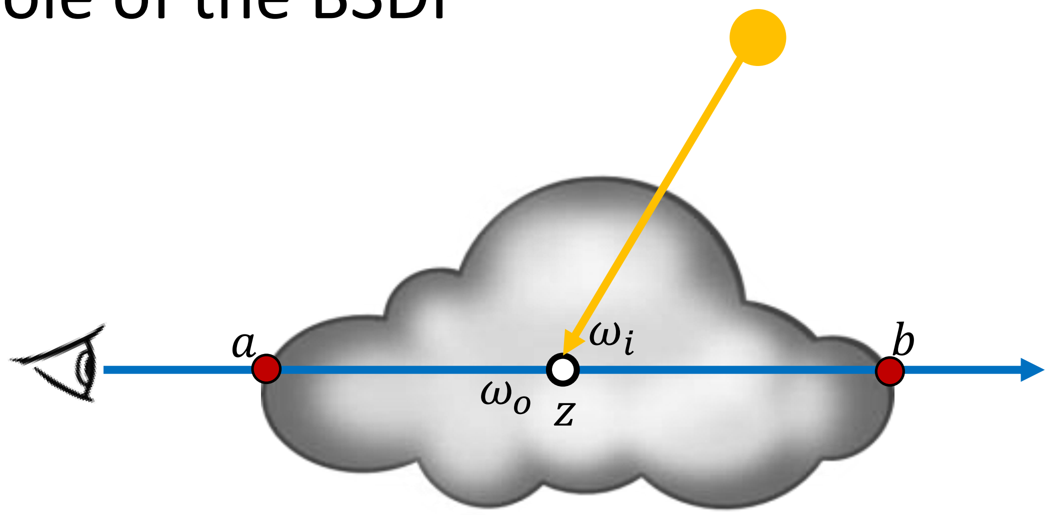


In-Scattering

Accounting for Reflections Inside the Volume

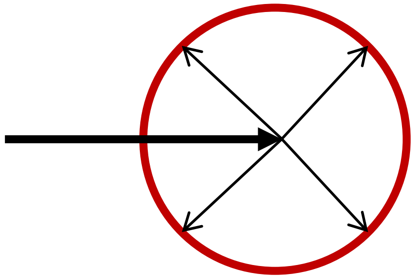
Direct Illumination

- Account for the (attenuated) direct illumination at every point z
- Similar to the rendering equation:
- $L_o(z, \omega_o) = \int_\Omega L_i(x, \omega_i)\, f_p(\omega_i, \omega_o)\, d\omega_i$
- Integration over the whole sphere Ω
- The **phase function** f_p takes on the role of the BSDF



Phase Functions

- $L_o(z, \omega_o) = \int_\Omega L_i(x, \omega_i)\, \boldsymbol{f_p(\omega_i, \omega_o)}\, d\omega_i$
- Describe what fraction of light is reflected from ω_i to ω_o
- Similar to BSDF for surface scattering
- Simplest example: isotropic phase function
  - $f_p(\omega_i, \omega_o) = \frac{1}{4\pi}$
  - (energy conservation: $\int_\Omega \frac{1}{4\pi} d\omega = 1$)



Phase Functions: Henyey-Greenstein

- Widely used
- Easy to fit to measured data

- $f_p(\omega_i, \omega_o) = \frac{1}{4\pi} \frac{1-g^2}{(1+g^2+2g\cos(\omega_i,\omega_o))^{\frac{3}{2}}}$

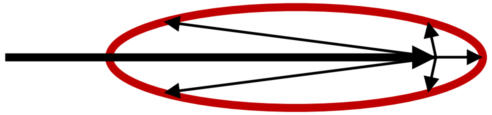
- g: asymmetry (scalar)
- $\cos(\omega_i, \omega_o)$: cosine of the angle formed by ω_i and ω_o

Henyey-Greenstein: Asymmetry Parameter

- $g = 0$: isotropic
- Negative g: back scattering
- Positive g: forward scattering

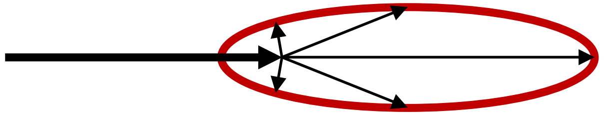
Back Scattering

http://commons.wikimedia.org



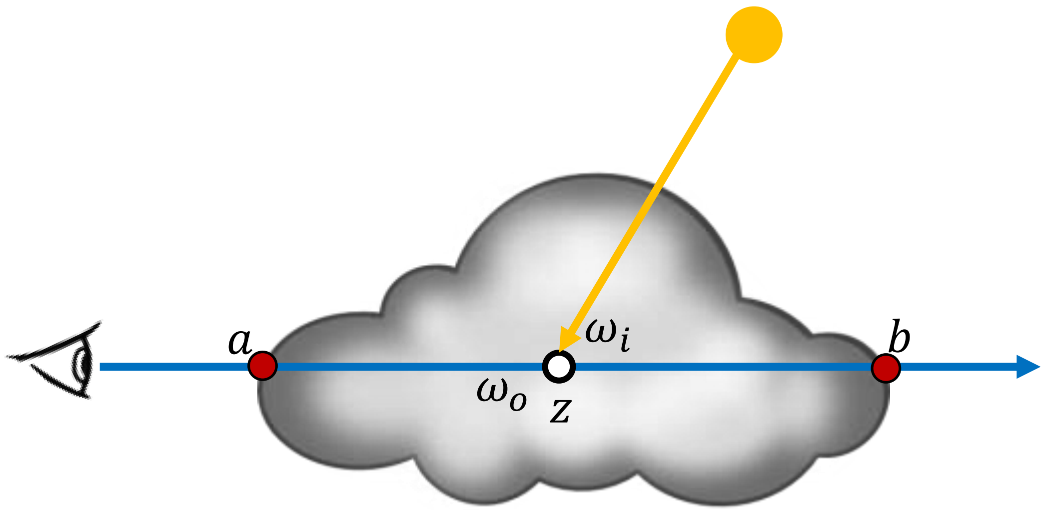
Forward Scattering

http://coclouds.com



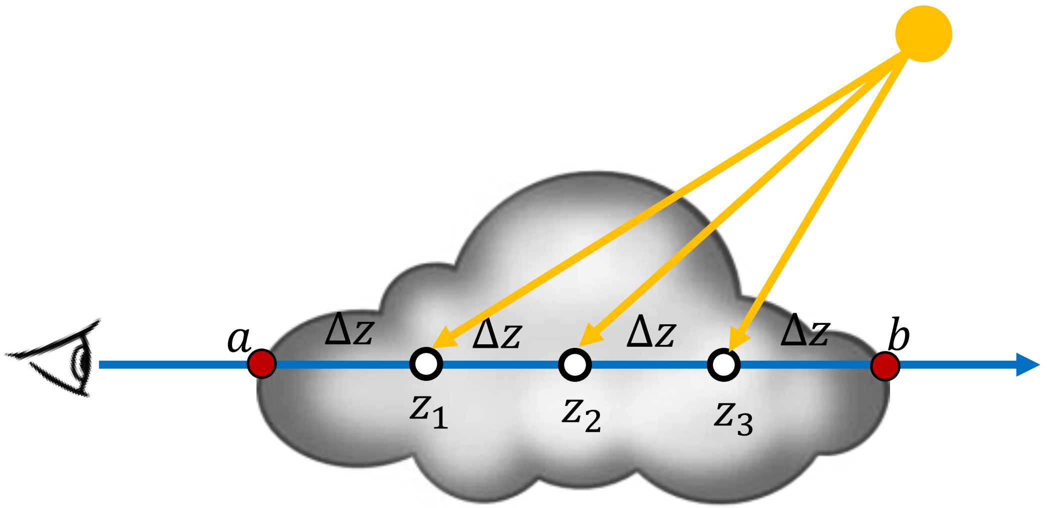
How to Estimate Volume Direct Illumination

- Reflected radiance at a point z: $L_o(z, \omega_o) = \int_\Omega L_i(x, \omega_i)\, f_p(\omega_i, \omega_o)\, d\omega_i$
- In our framework:
  - Sum over all point lights (as for surfaces)
  - Trace shadow ray (as for surfaces)
  - Estimate attenuation along the shadow ray (as for surfaces)



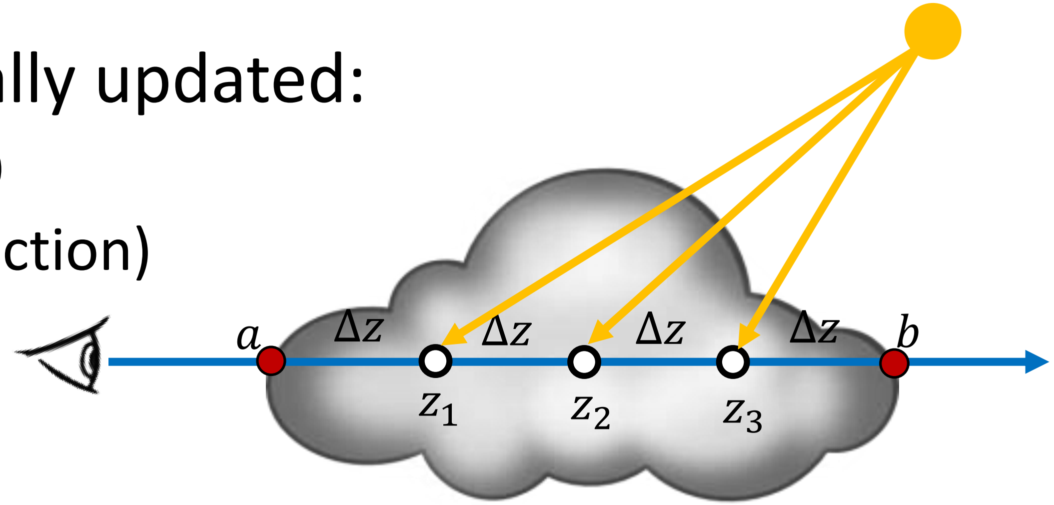
Ray Marching to Compute In-Scattering

- Same as for emission
- Goal: estimate the integral $\int_a^b T(z,a)\,\mu_s(z)\,L_i(z)\,f_p\,dz$
- Quadrature:
  - $\int_a^b T(z,a)\,\mu_s(z)\,L_i(z)\,f_p\,dz \approx \sum_i T(z_i,a)\,\mu_s(z)\,L_i(z_i)\,f_p\,\Delta z$



Ray Marching to Compute In-Scattering

- Same as for emission
- Goal: estimate the integral $\int_a^b T(z,a)\, L_i(z)\, f_p\, dz$
- Quadrature:
  - $\int_a^b T(z,a)\, L_i(z)\, f_p\, dz \approx \sum_i T(z_i,a)\, L_i(z_i)\, f_p\, \Delta z$
- Attenuation $T(z_i,a)$ estimated as before
- Attenuation can be incrementally updated:
  - $T(z_i,a) = T(z_{i-1},a)\, T(z_i,z_{i-1})$
  - (because it is an exponential function)

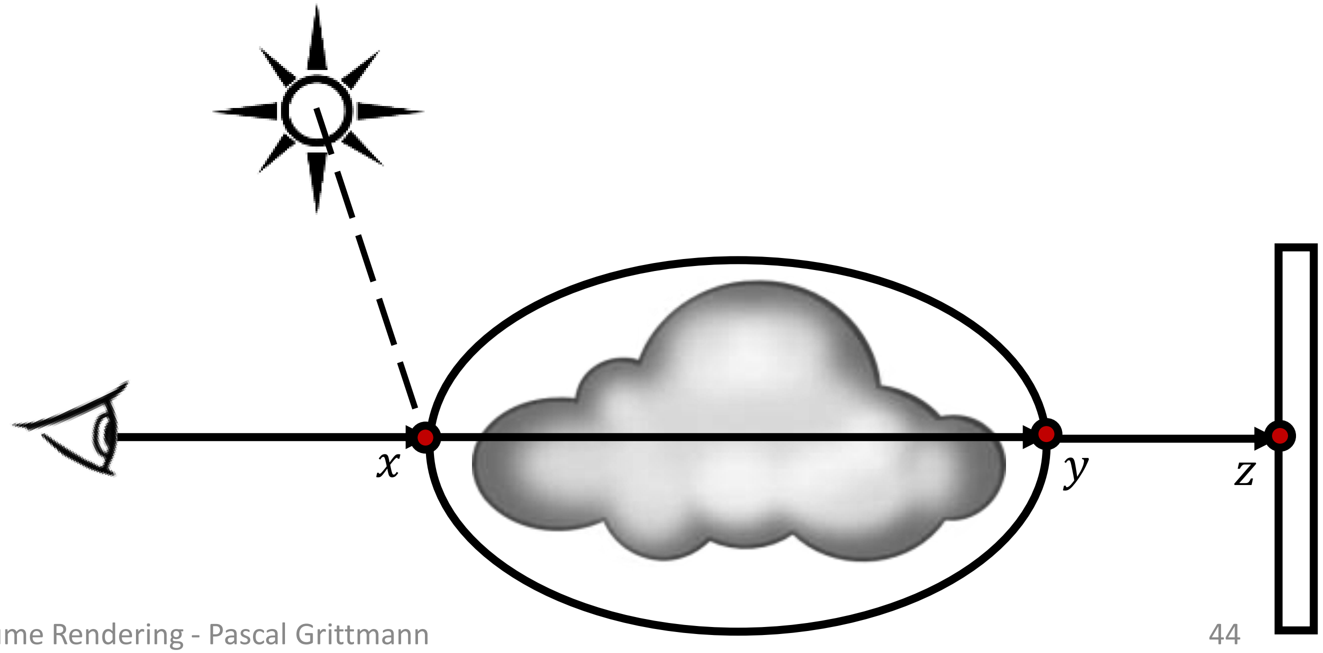


Putting it all Together

A Simple Volume Integrator

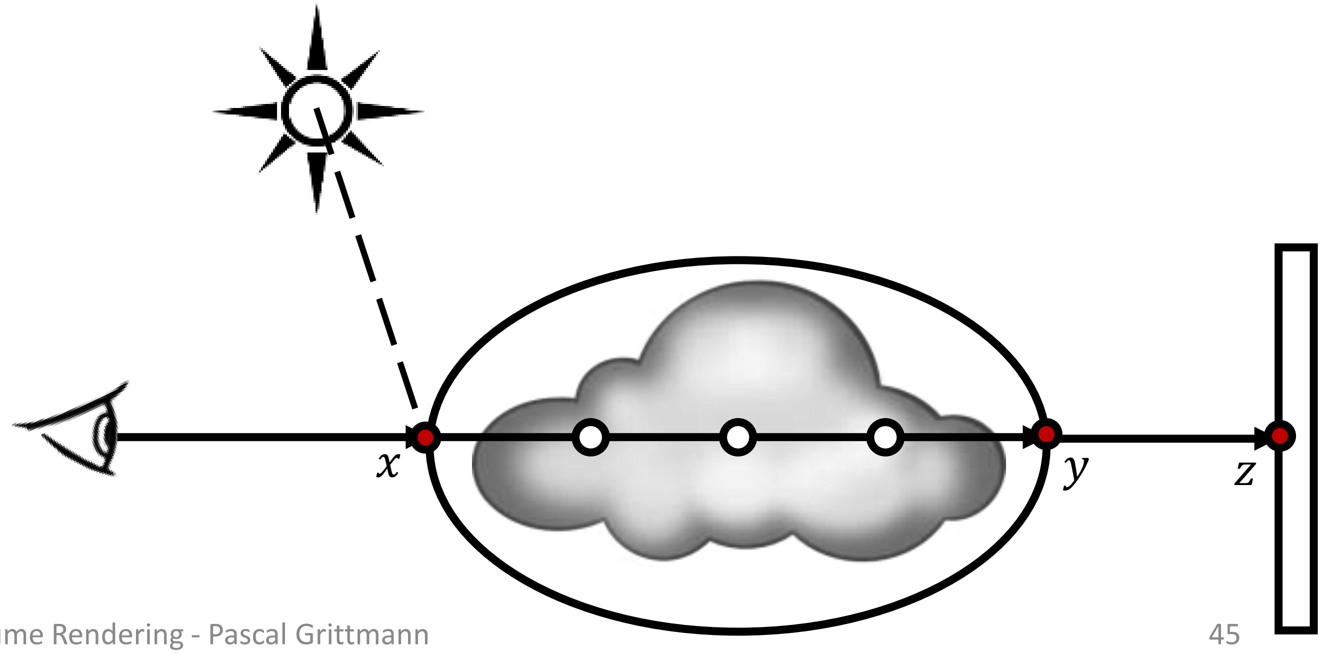
A Simple Volume Integrator

- Estimate direct illumination at x (as before)
- If volume: continue straight ahead until no volume (yields intersections y, z)



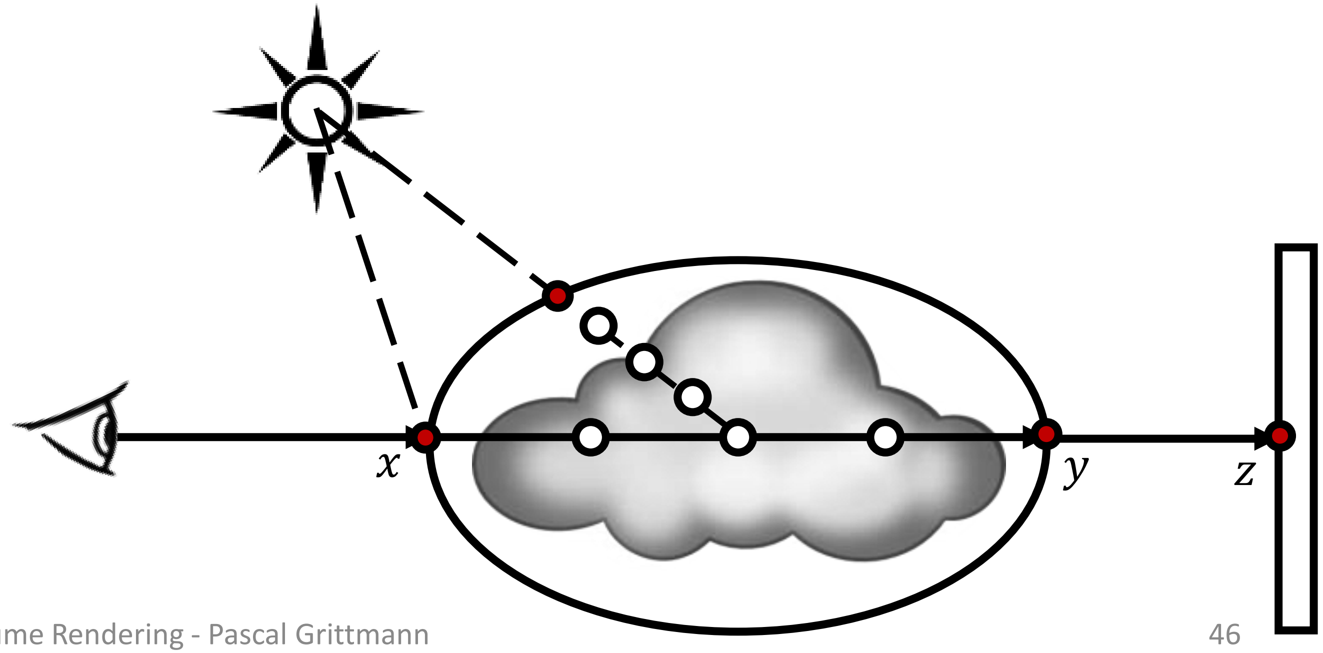
A Simple Volume Integrator

- Estimate direct illumination at x (as before)
- If volume: continue straight ahead until no volume (yields intersections y, z)
- Ray marching to estimate attenuation, emission,



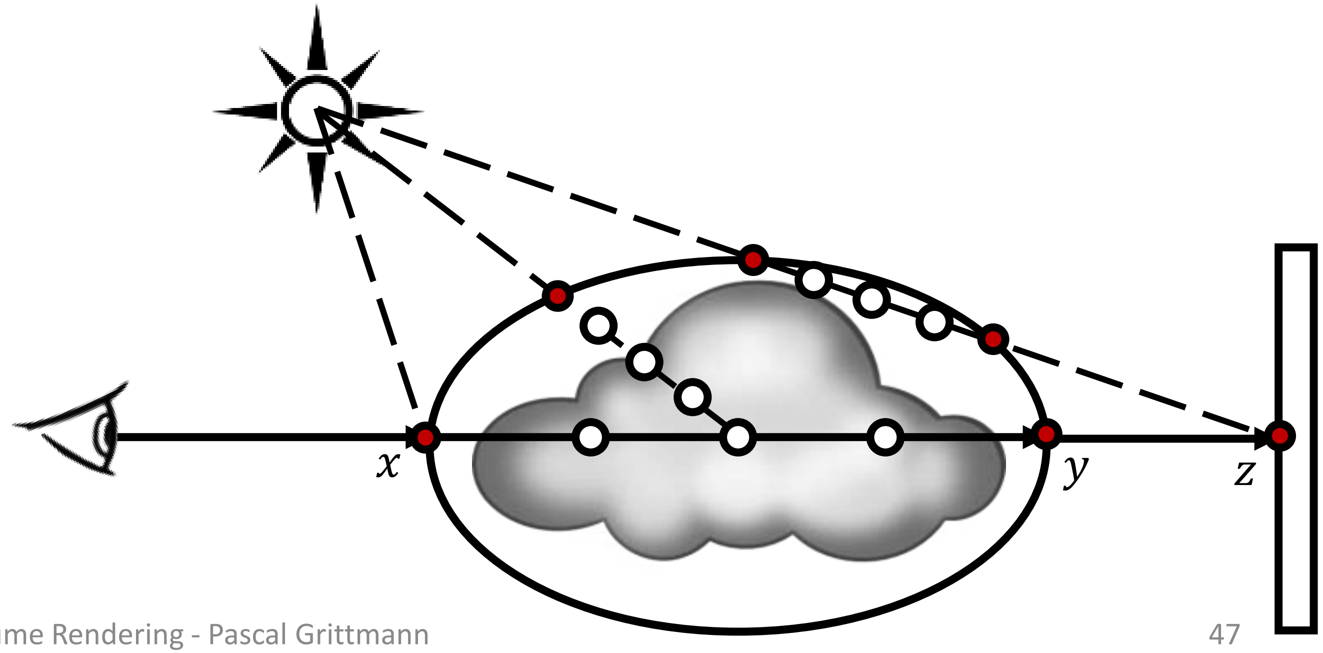
A Simple Volume Integrator

- Estimate direct illumination at x (as before)
- If volume: continue straight ahead until no volume (yields intersections y, z)
- Ray marching to estimate attenuation, emission, and in-scattering
  - Shadow rays to the lights + ray marching to compute attenuation



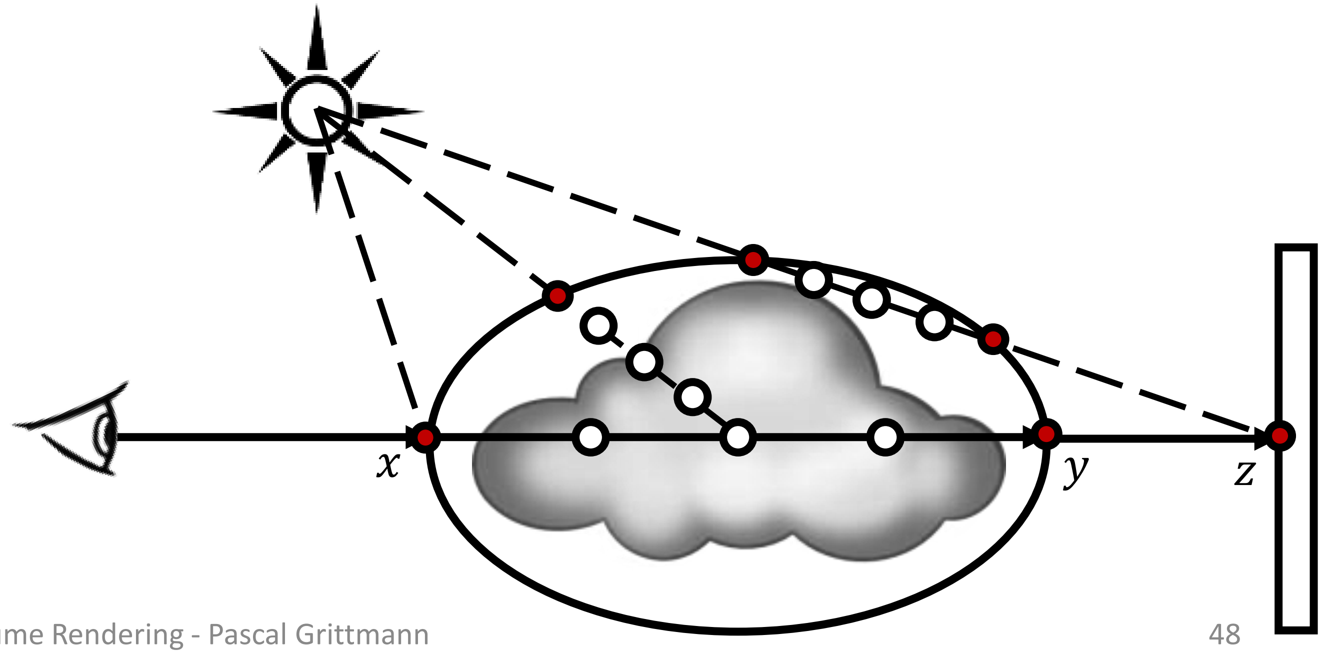
A Simple Volume Integrator

- Estimate direct illumination at x (as before)
- If volume: continue straight ahead until no volume (yields intersections y, z)
- Ray marching to estimate attenuation, emission, and in-scattering
  - Shadow rays to the lights + ray marching to compute attenuation
- Compute illumination at z (as before)



A Simple Volume Integrator

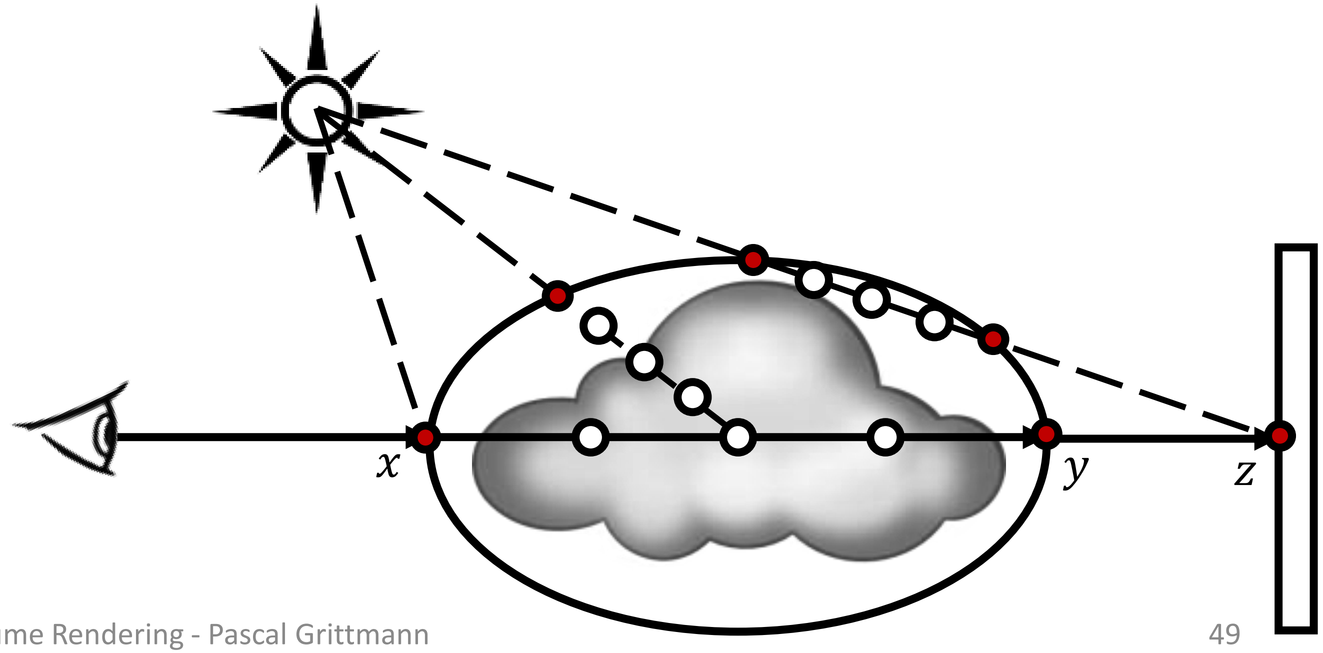
- Estimate direct illumination at x (as before)
- If volume: continue straight ahead until no volume (yields intersections y, z)
- Ray marching to estimate attenuation, emission, and in-scattering
  - Shadow rays to the lights + ray marching to compute attenuation
- Compute illumination at z (as before)
- Add together:
  - Attenuated illumination from z
  - Volumetric emission along $\overline{xy}$
  - In-scattering along $\overline{xy}$
  - Direct illumination at x



A Simple Volume Integrator

- Estimate direct illumination at x (as before)
- If volume: continue straight ahead until no volume (yields intersections y, z)
- Ray marching to estimate attenuation, emission, and in-scattering
  - Shadow rays to the lights + ray marching to compute attenuation
- Compute illumination at z (as before)
- Add together:
  - Attenuated illumination from z
  - Volumetric emission along $\overline{xy}$
  - In-scattering along $\overline{xy}$
  - Direct illumination at x

Multiply by BTDF! (applies to: Attenuated illumination from z; Volumetric emission along $\overline{xy}$; In-scattering along $\overline{xy}$)



References

- [K. Museth, 2013] “VDB: High-Resolution Sparse Volumes With Dynamic Topology”. ACM Transactions on Graphics, Volume 32, Issue 3, Pages 27:1-27:22, June 2013