Advanced Topics

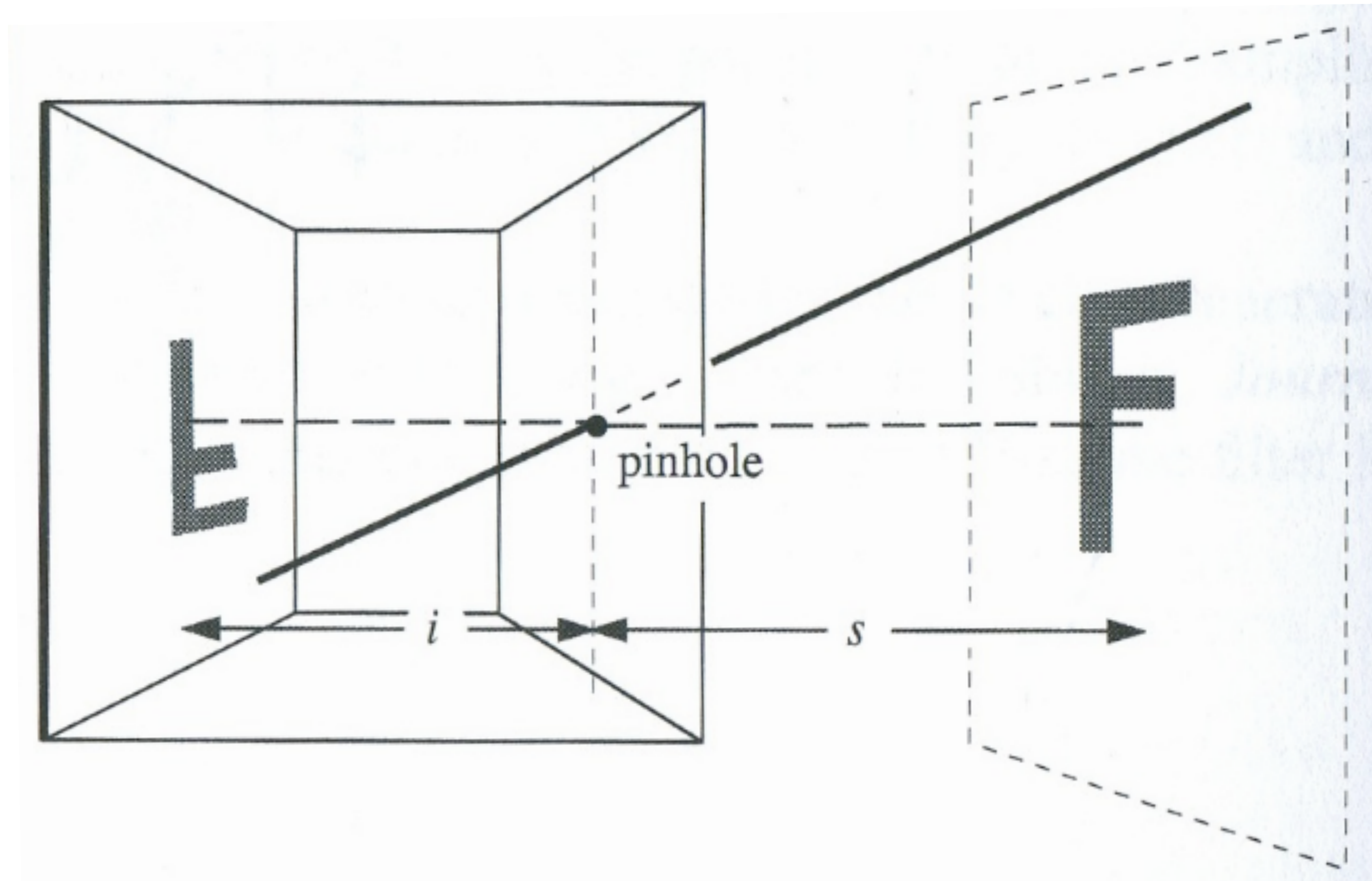
Michael Doggett
Department of Computer Science
Lund university

http://ompf.org/forum/viewtopic.php?f=10&t=1492&start=100#p19847
SPPM : two surface glass shapes created using Carlo H. Séquin's Scherk-Collins Sculpture Generator ... let it cook overnight (~4 hours @ 1440x1080)."

Outline

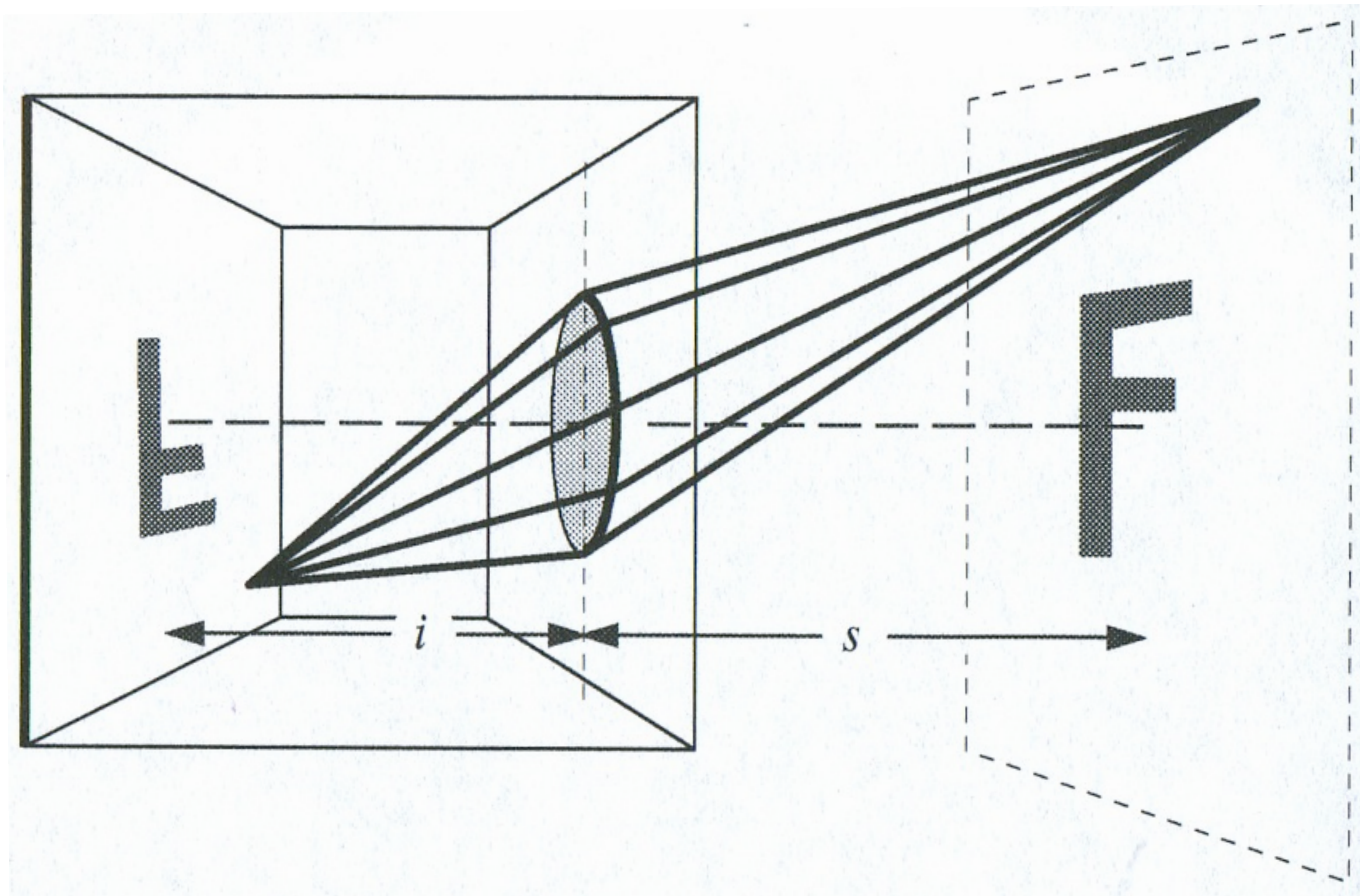
- Depth of Field and Motion Blur
- Texture Mapping
- Participating Media
- Elective Assignment

Pin-hole Camera



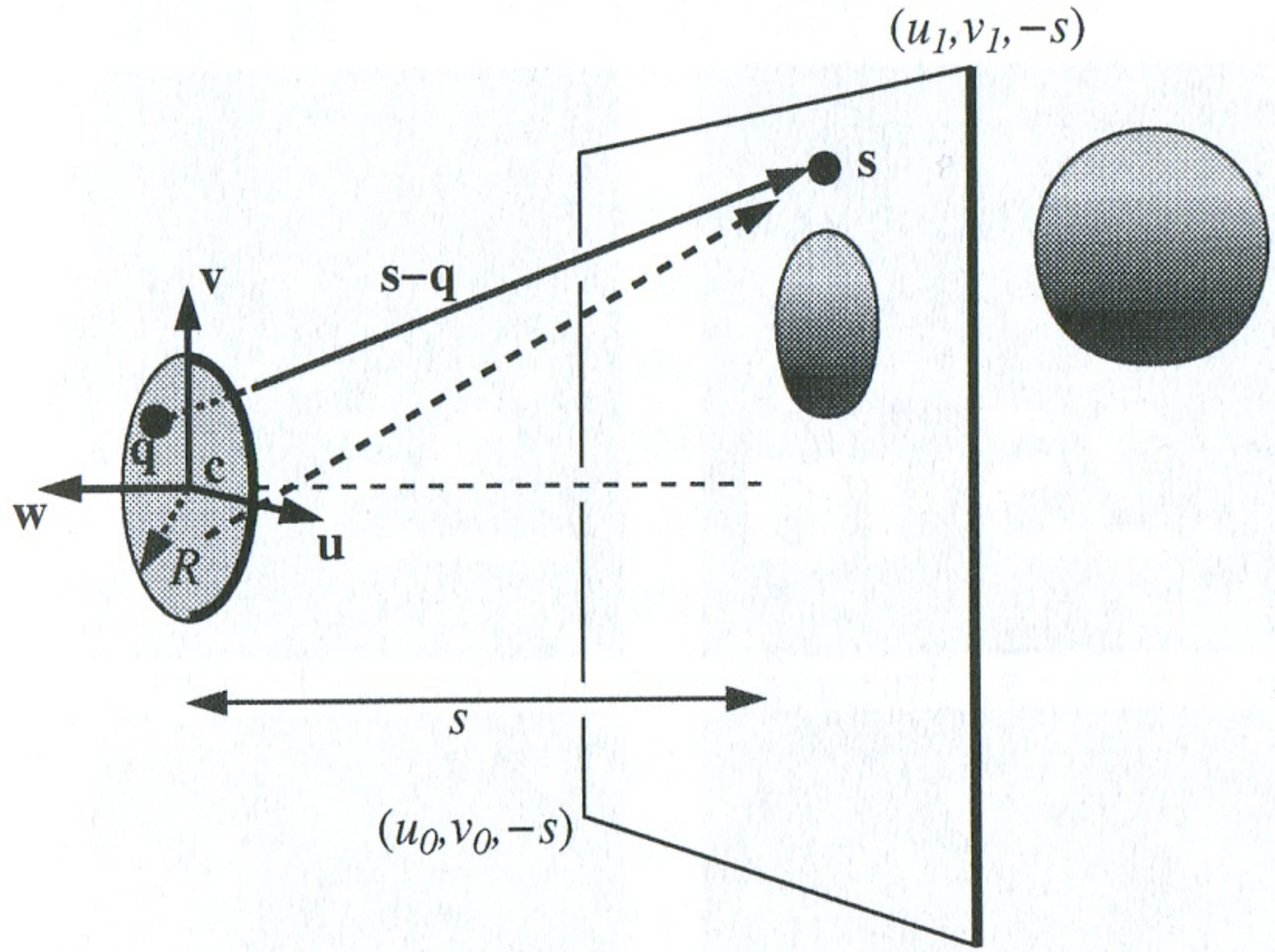
- Doesn't get enough light
- Need a bigger hole and a longer exposure time

Thin-lens camera



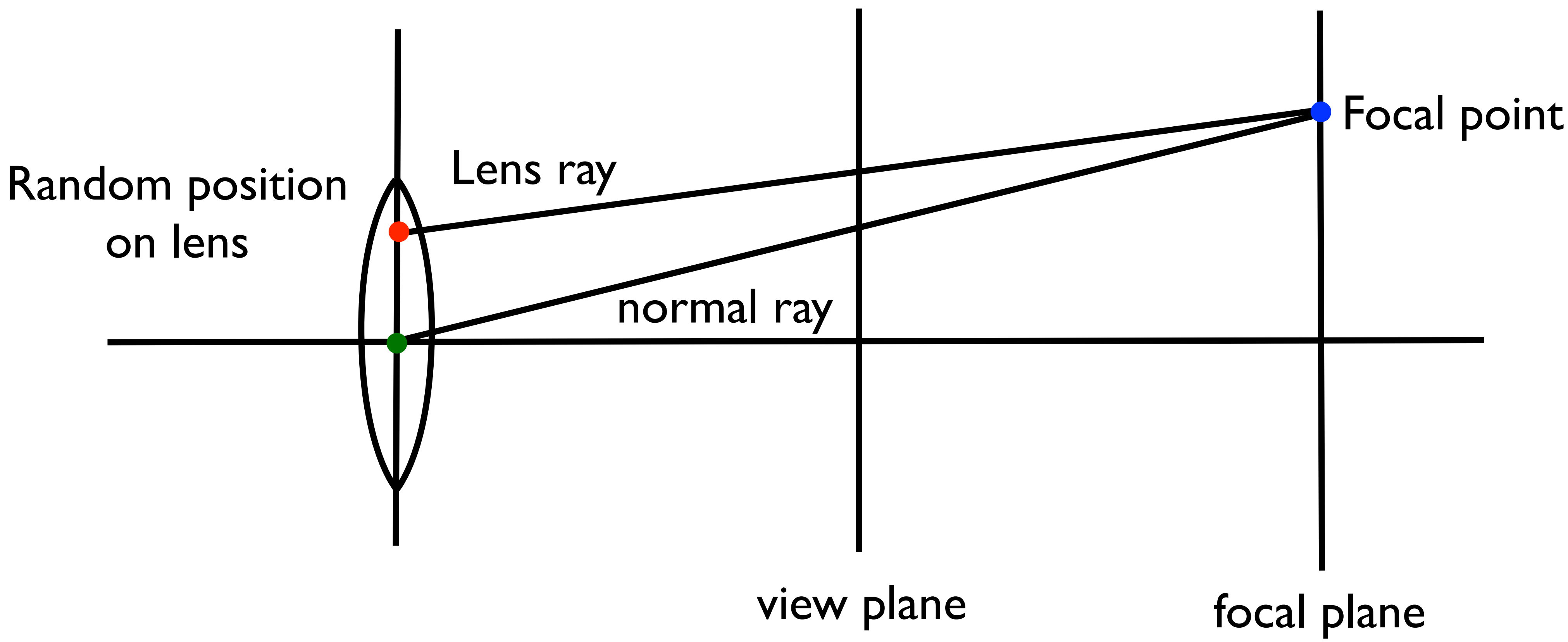
- Need a lens
- Make a disc for the lens

Thin-lens camera



- Take random samples on the lens for ray start
- Typically use *u,v* coordinates to represent position on lens
- Objects not on the focal plane (*s*) will be blurred

Setting up DOF rays



- Trace normal ray to focal plane to find focal point
- Trace a new lens ray from random position on lens to focal point

Depth of field

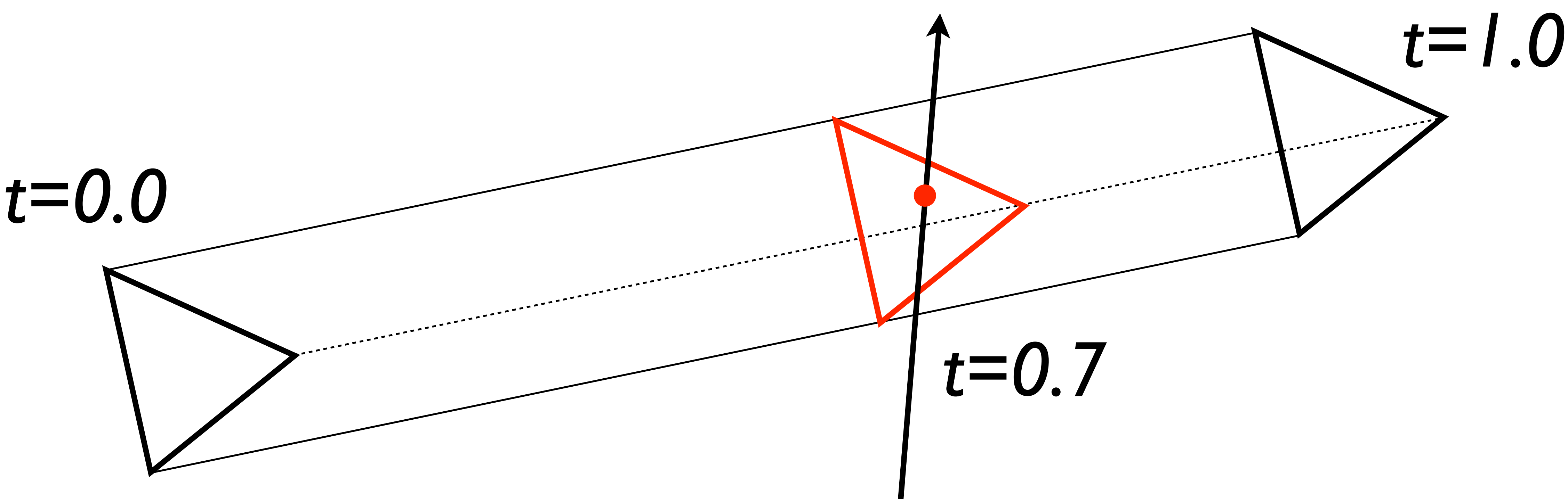
from "Decoupled Sampling for Graphics Pipelines", Jonathan Ragan-Kelley, Jaakko Lehtinen, Jiawen Chen, Michael Doggett and Fredo Durand, ACM Transactions on Graphics 30(3) (To Appear at SIGGRAPH 2011)

Motion Blur

from "Decoupled Sampling for Graphics Pipelines", Jonathan Ragan-Kelley, Jaakko Lehtinen, Jiawen Chen, Michael Doggett and Fredo Durand, ACM Transactions on Graphics 30(3) (To Appear at SIGGRAPH 2011)

Motion Blur

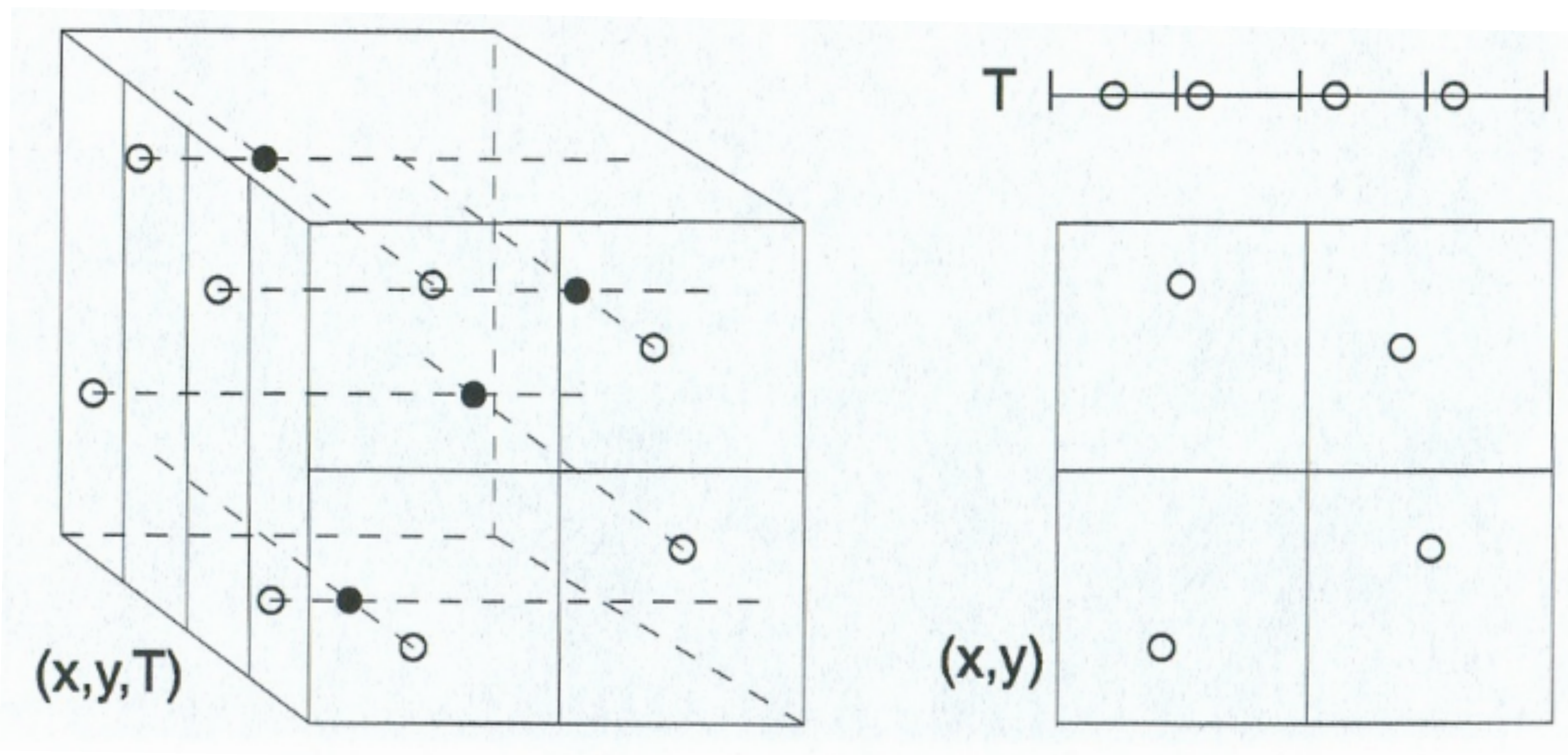
- Store position at shutter open and shutter close
- Associate a random time (t) with each ray

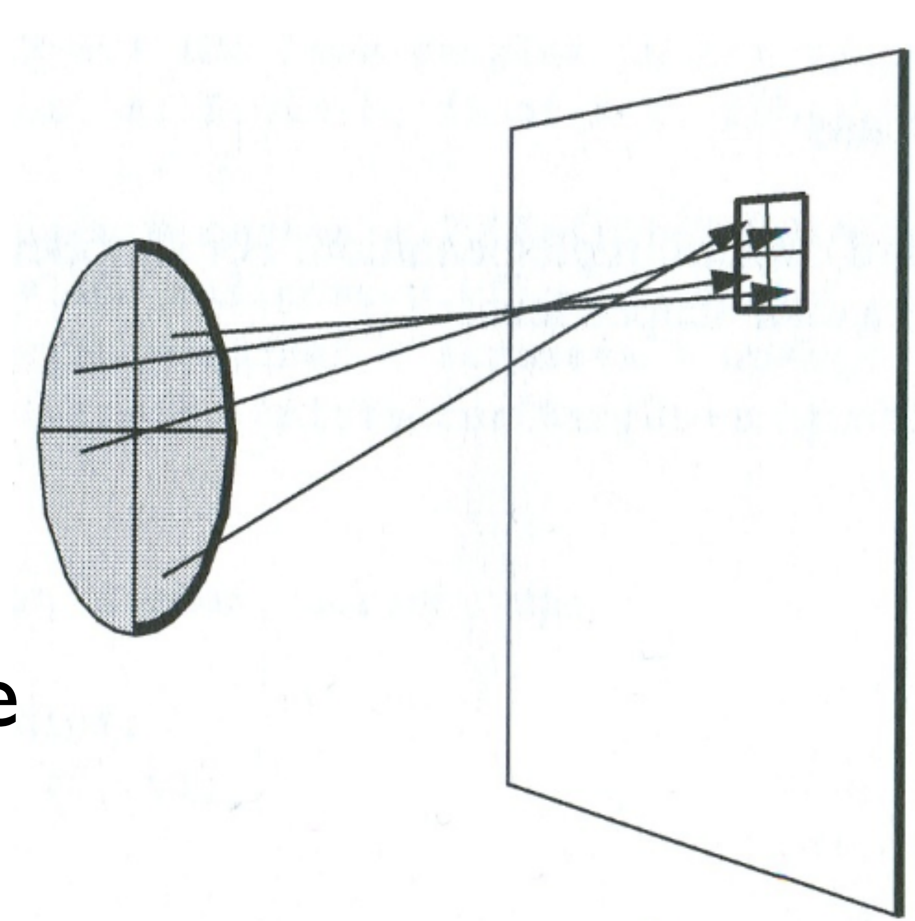


Multidimensional Sampling

- Sample in pixel, lens, and time
  - 5 dimensions (x, y, u, v, t)
- How to generate samples?
  - Random per dimension? - bad
  - Jittered, 32 samples 2x2x2x2x2
    - only 2x2 samples for spatial antialiasing

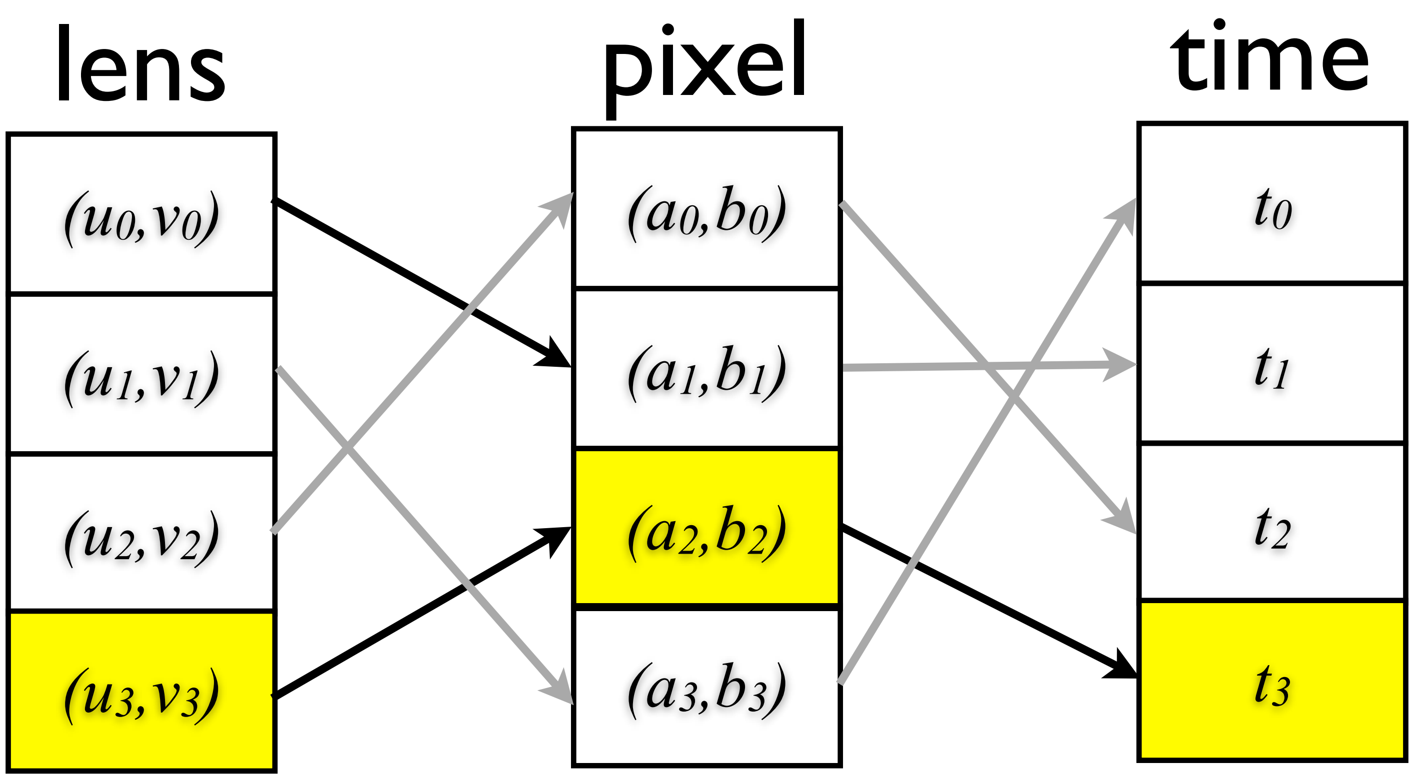
Multidimensional Sampling - Randomly pair jittered samples





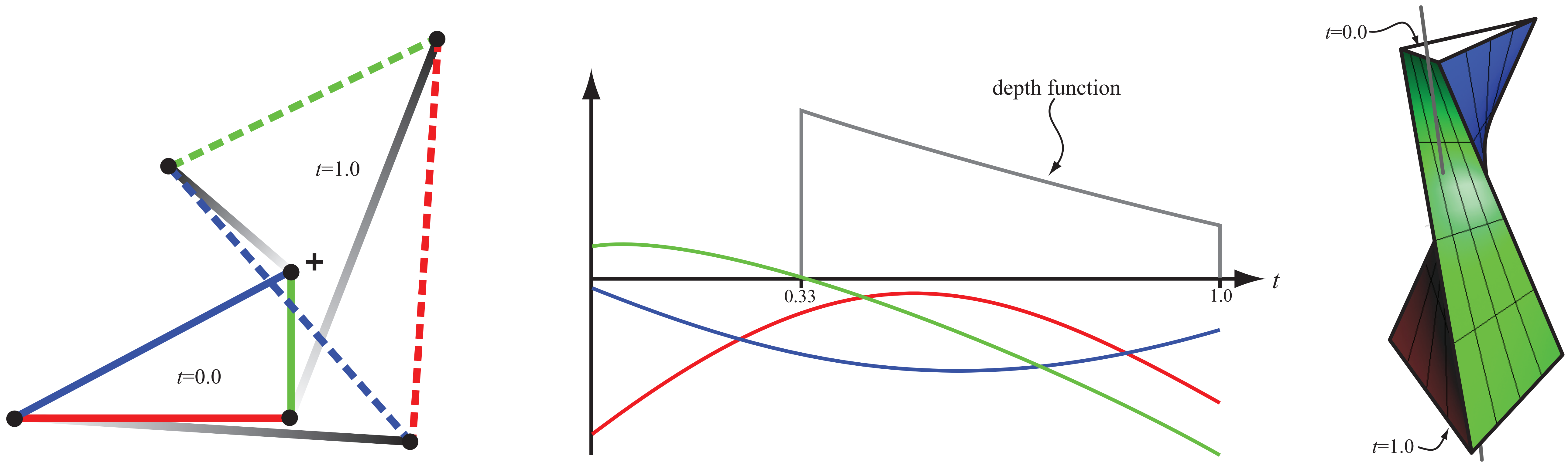
- Four samples over space and time
- Jitter in x,y, jitter in t and group
- Do the same thing for lens samples and pixels

Multidimensional Sampling - Randomly pair jittered samples



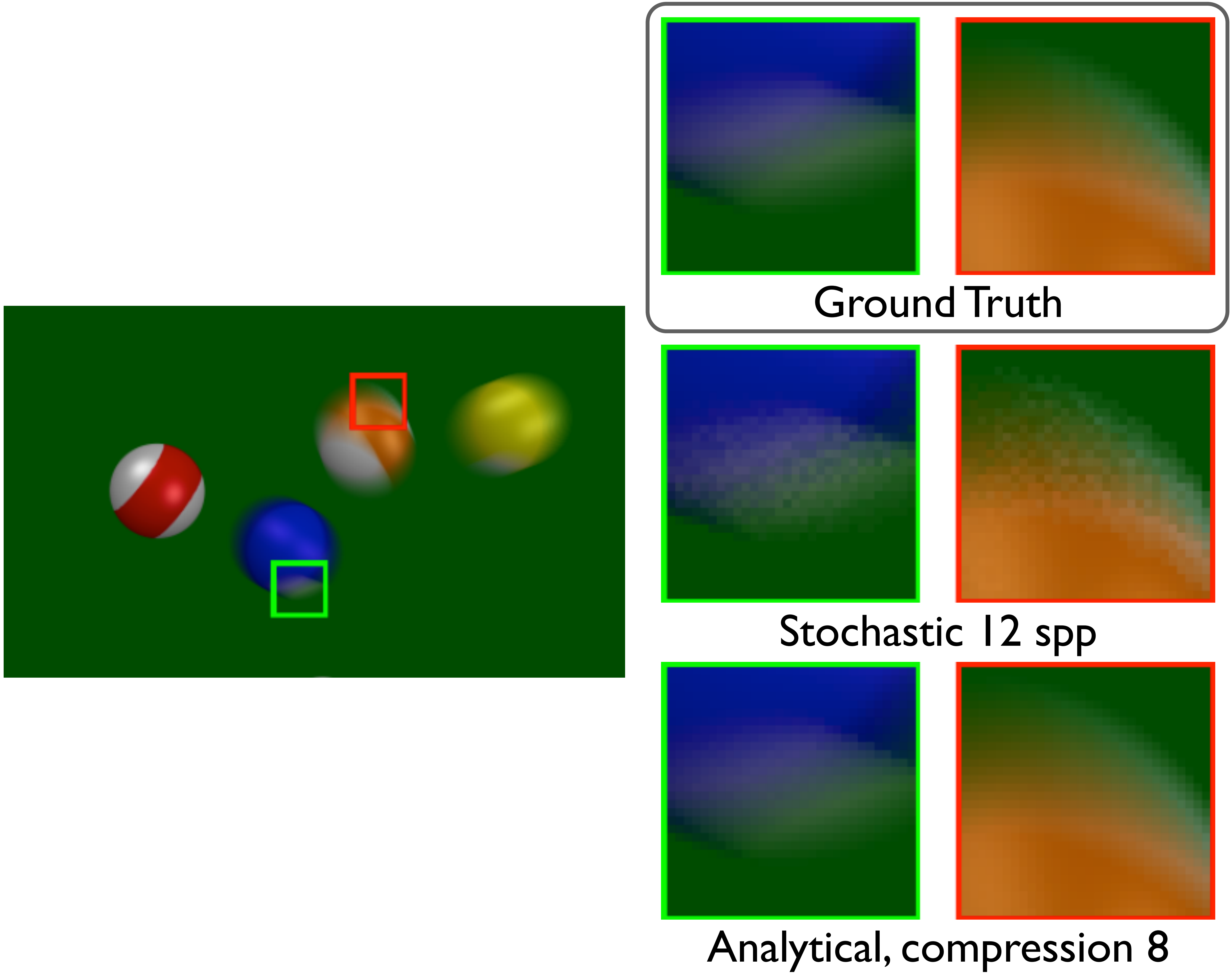
- Generate good samples for each
- Randomly group up over lens, space, and time
- Interested? Read “Distributed ray tracing”, Rob Cook, Thomas Porter, Loren Carpenter, SIGGRAPH 1984

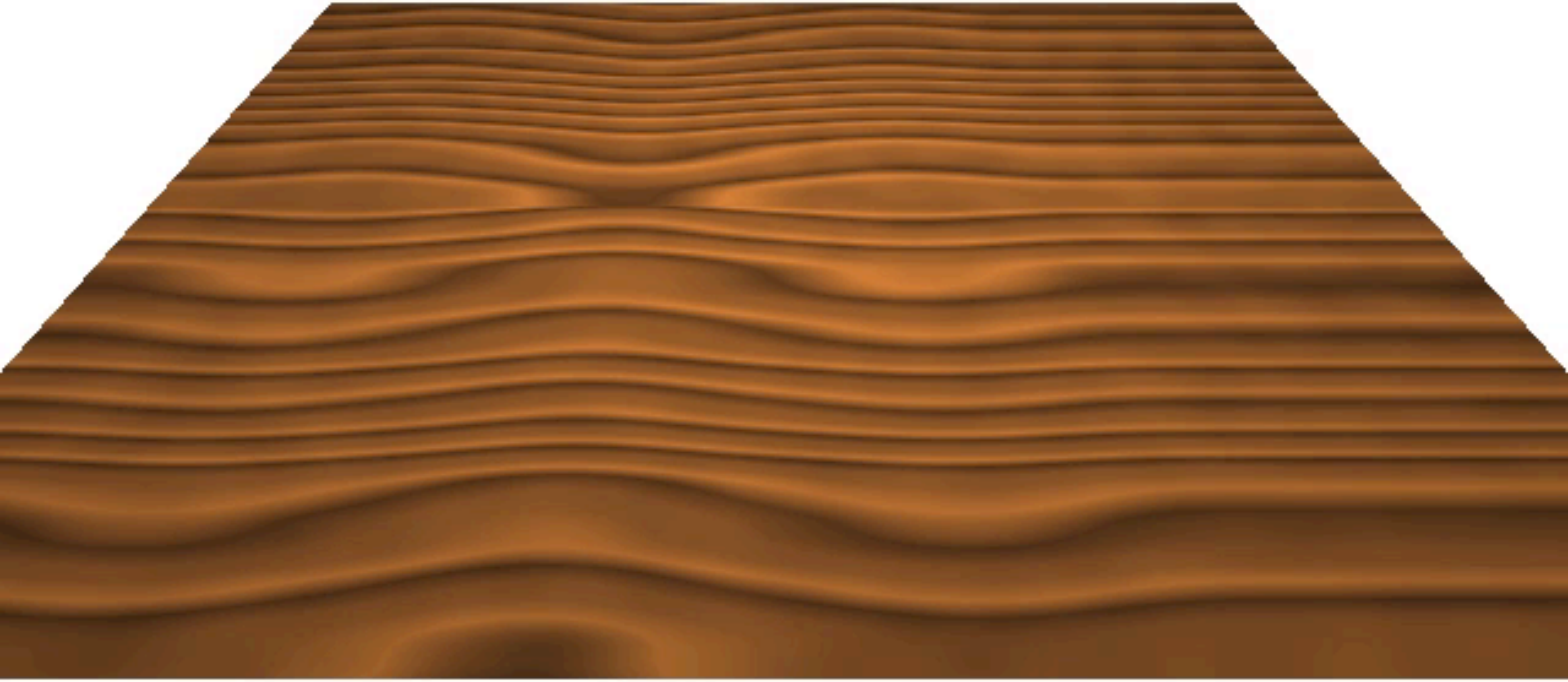
Analytical Motion Blur



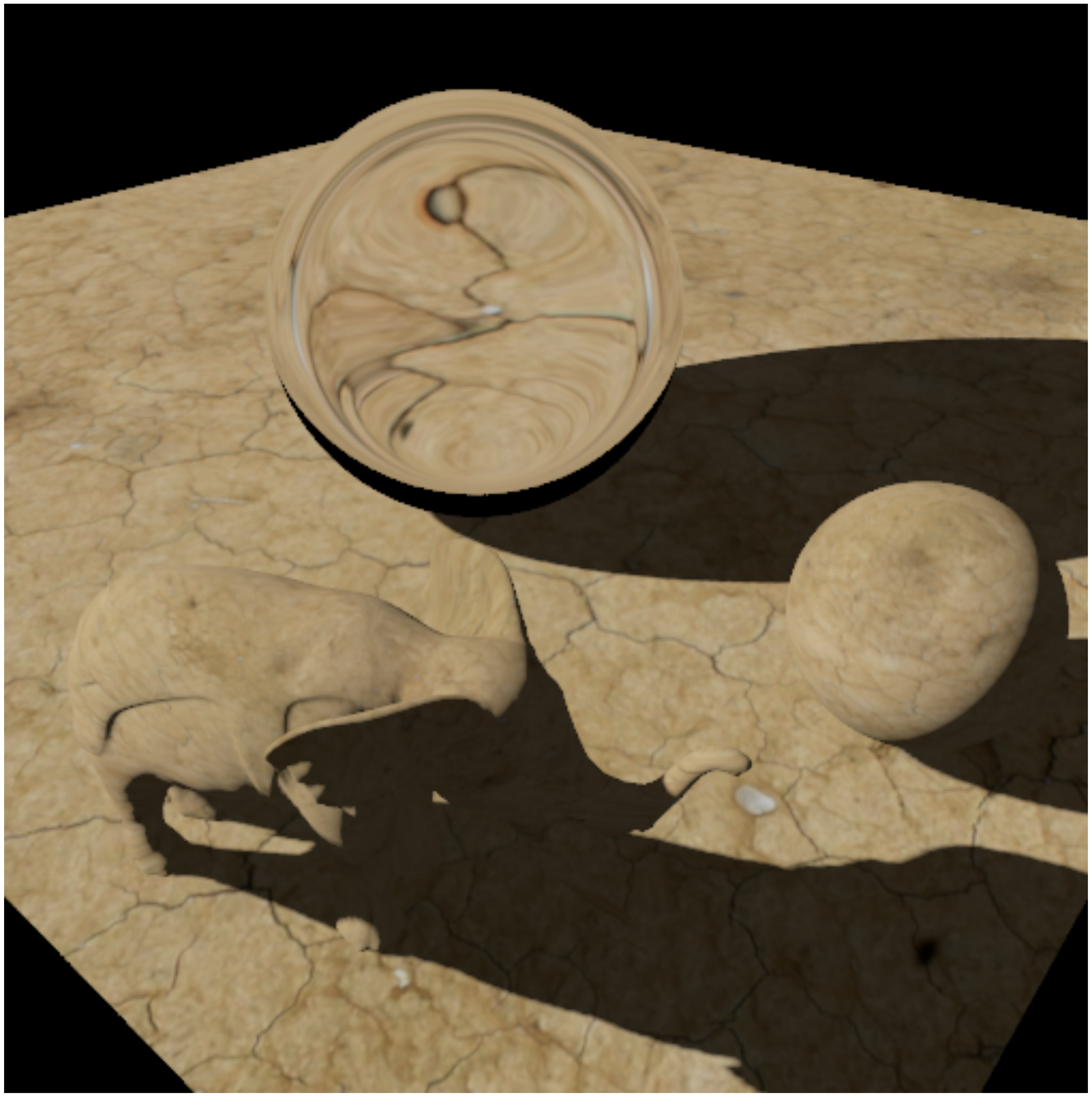
- Sampled motion blur always results in some noise
- Compute visibility analytically

from "Analytical Motion Blur Rasterization with Compression", Carl Johan Gribel, Michael Doggett, and Tomas Akenine-Möller, High-Performance Graphics 2010


Ground Truth
Stochastic 12 spp
Analytical, compression 8



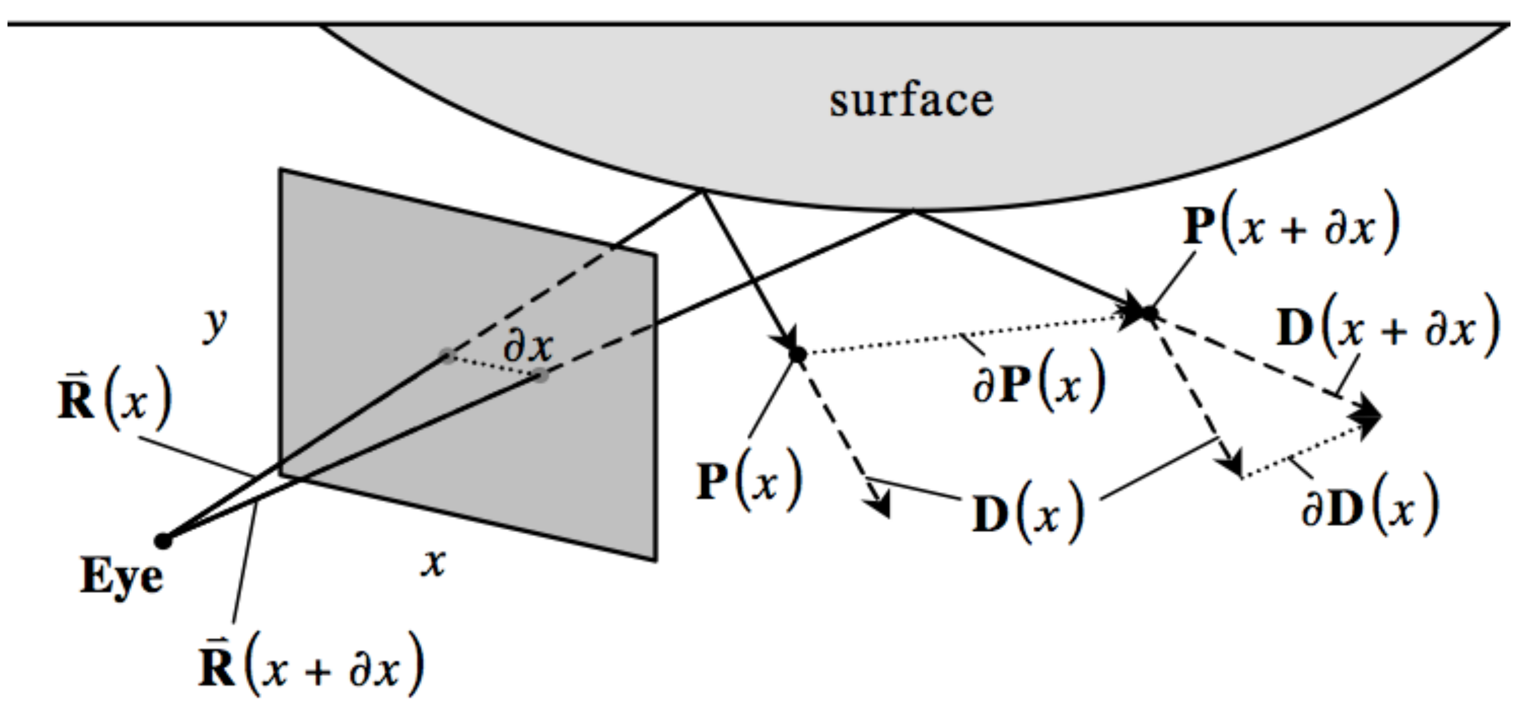
Texture Mapping



- Adds surface colors
- Enables normal mapping
- Could use Sponza scene from EDAN35 High Performance CG!

Texture Mapping

- We need 'ray differentials'
  - Derivatives of the ray with respect to the image plane
  - For full details see, "Tracing Ray Differentials", Homan Igehy, SIGGRAPH 99
- Calculate starting differential values by differentiating the ray origin and direction for x and y at the camera



Texture Mapping

- Propagate the differentials correctly at intersections for reflection and refraction
  - Add code to `Intersection::calculatePositionDifferential` and `Intersection::getReflection/RefractionRay`
- For intersectable
  - Calculate the texture coordinate differential given the hit position and the hit position differential `Intersectable::calculateTextureDifferential`
  - For reflection and refraction propagation
    - Calculate the normal differential given the hit position and the hit position differential `Intersectable::calculateNormalDifferential`

Texture Mapping

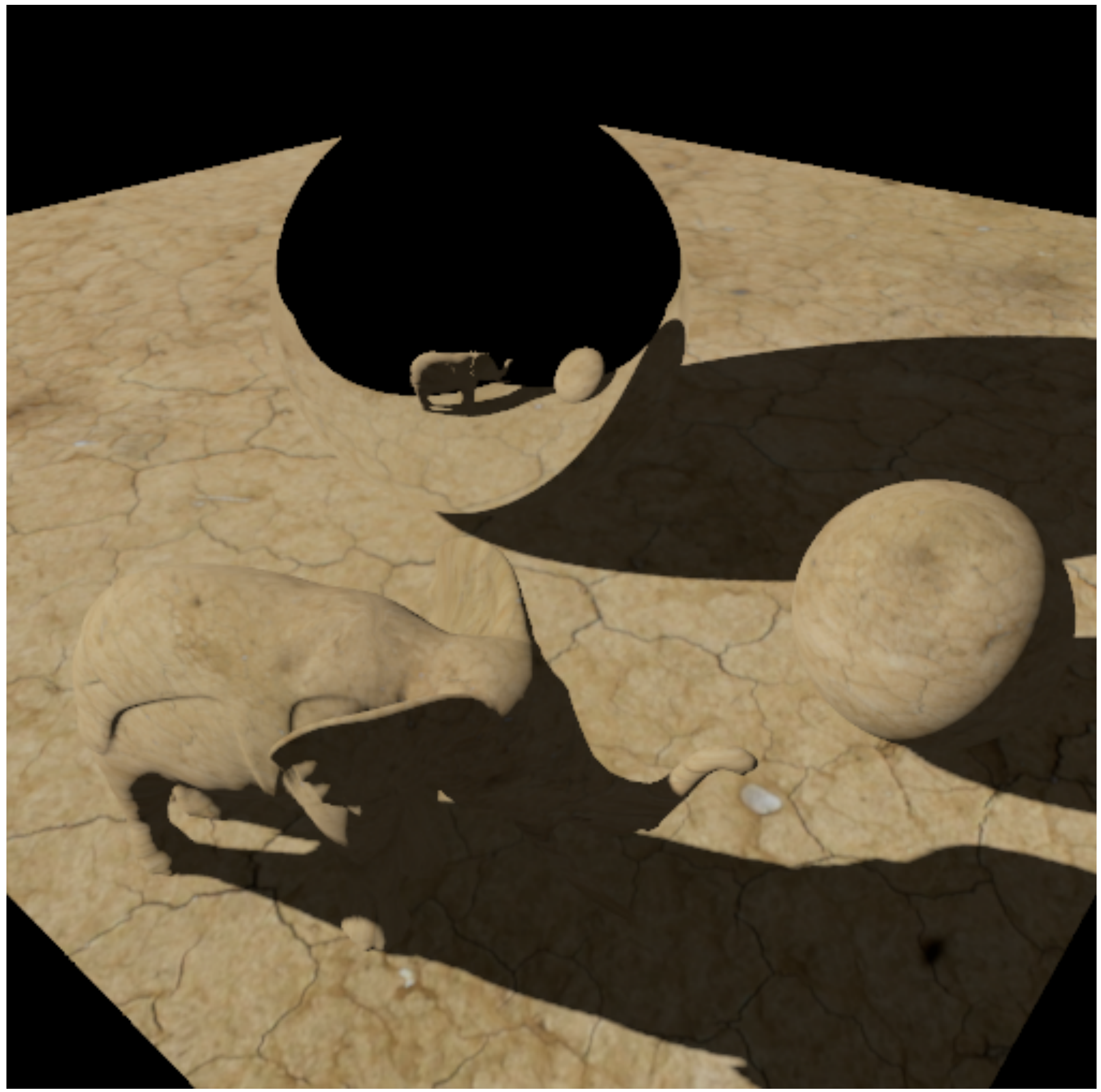
- Example of a BRDF using texture differentials

```
Color evalBRDF(const Intersection& is, const Vector3D& L) {
    Ray::Differential dp = is.calculatePositionDifferential();

    UV duvdx = is.mObject->calculateTextureDifferential(is.mPosition, dp.dx);
    UV duvdy = is.mObject->calculateTextureDifferential(is.mPosition, dp.dy);

    return Diffuse::getBRDF(is, L) * texture->getAnisotropic(is.mTexture.u,
is.mTexture.v, duvdx, duvdy);
}
```

Texture Mapping



Rendering Participating Media using Photon Mapping

"Efficient Simulation of Light Transport in Scenes with Participating Media using Photon Maps",
Henrik Wann Jensen and Per H. Christensen, ACM SIGGRAPH 1998

Participating Media

- Radiance (L) is reduced due to scattering and absorption
  - Absorption coefficient σ_a
  - Scattering coefficient σ_s
- Radiance is increased due to in-scattering

from Chapter 10, "Realistic Image Synthesis Using Photon Mapping", Henrik Wann Jensen, 2001

Volume Rendering Equation

- Integrate over a length s
- L = emitted + in-scattered + other-end

$$
\begin{aligned}
L(x,\vec{\omega}) \;=\; & \int_0^s e^{-\tau(x,x')}\sigma_a(x')L_e(x')\,dx' + \\
& \int_0^s e^{-\tau(x,x')}\sigma_s(x')\int_{\Omega_{4\pi}} p(x',\vec{\omega}',\vec{\omega})L_i(x',\vec{\omega}')\,d\vec{\omega}'\,dx' + \\
& e^{-\tau(x,x+s\vec{\omega})}L(x+s\vec{\omega},\vec{\omega})
\end{aligned}
$$

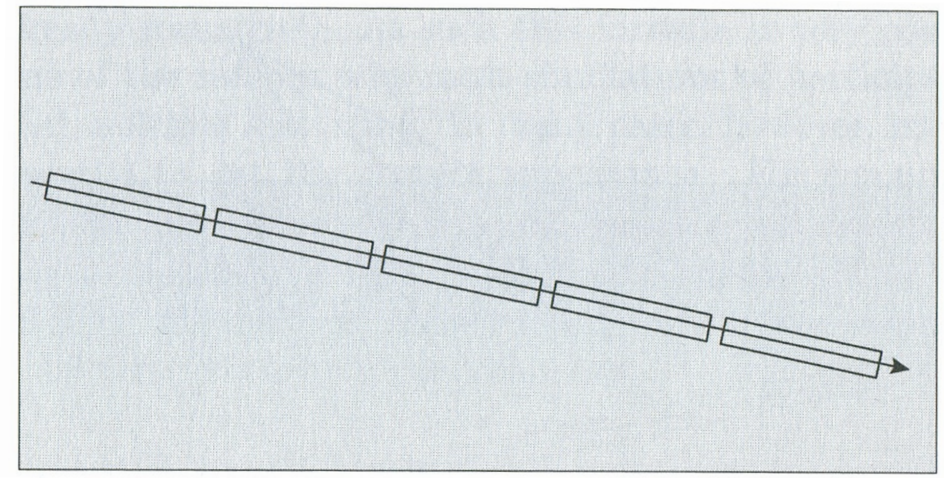
emitted light

in-scattering

out-scattering (from scattering and absorption)

- p is the phase function
  - similar to BRDF
  - integrate over sphere

Ray Marching Algorithm



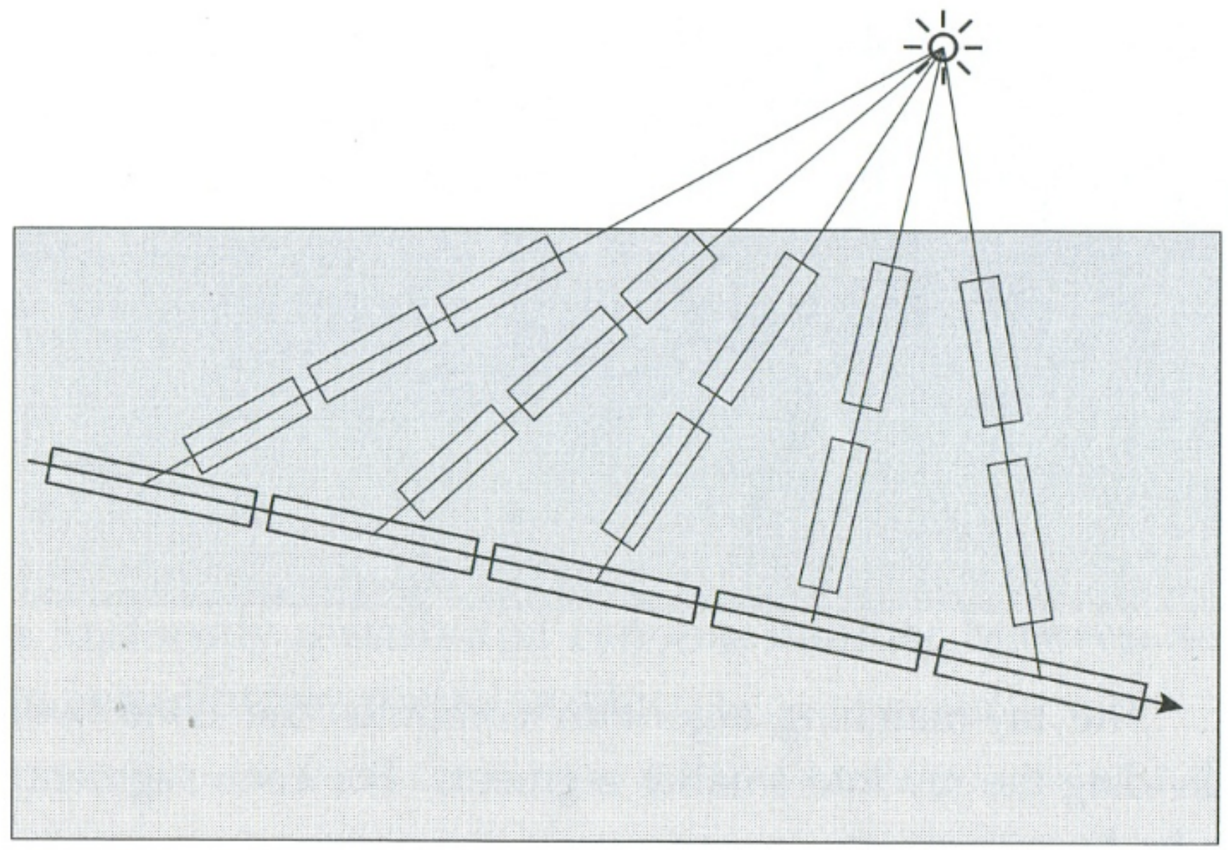
- Solve for segments, Dx
  - assume constant
    - incoming light
    - medium properties

$$L_{n+1}(x,\vec{\omega}) = \sum_{l}^{N} L_l(x,\vec{\omega}_l')p(x,\vec{\omega}_l',\vec{\omega})\sigma_s(x)\Delta x \ + \ e^{-\sigma_t \Delta x} L_n(x+\Delta x,\vec{\omega})$$

N light source contribution

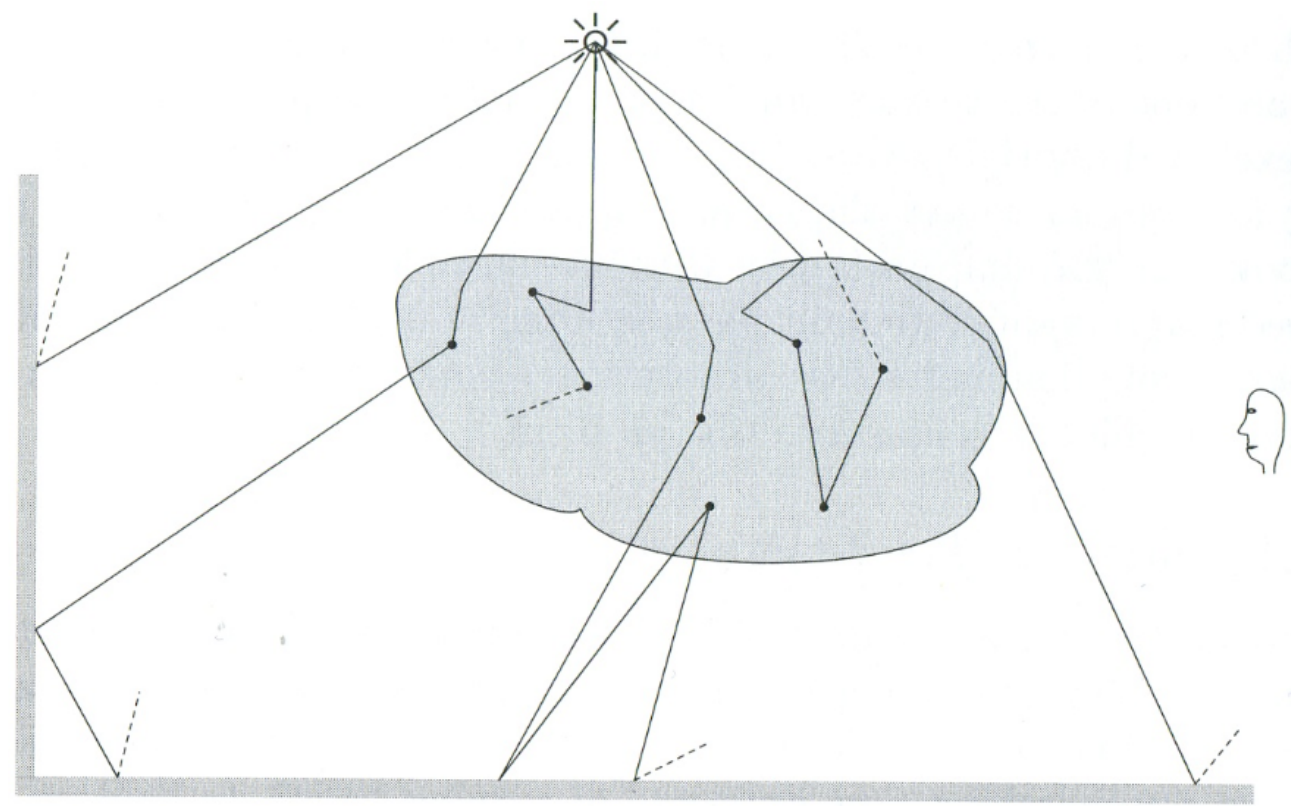
previous segment

Ray-marching



- Must do a ray-marching integration for each shadow ray if the medium is non-homogeneous

Photon tracing in Participating Media



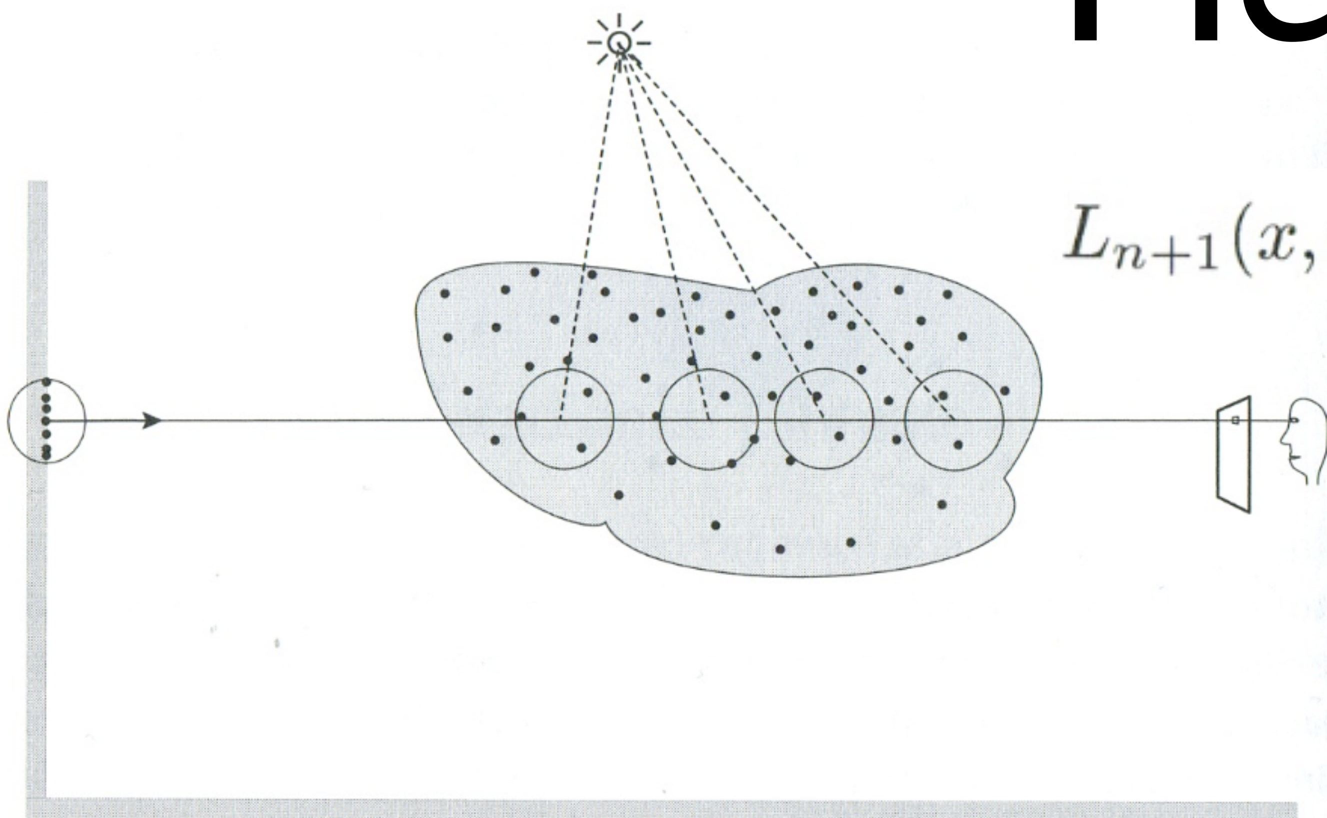
- Photons either scatter or absorbed
- Photons are stored in a **volume photon map**
- Only store photons that have been scattered once

Volume Radiance Estimate

$$
\begin{aligned}
(\vec{\omega} \cdot \nabla) L_o(x, \vec{\omega}) &= \int_{\Omega_{4\pi}} p(x, \vec{\omega}', \vec{\omega}) \frac{d^2\Phi(x, \vec{\omega})}{dV} \\
&\approx \sum_{p=1}^{n} f(x, \vec{\omega}'_p, \vec{\omega}) \frac{\Delta\Phi_p(x, \vec{\omega}'_p)}{\Delta V} \\
&\approx \sum_{p=1}^{n} f(x, \vec{\omega}'_p, \vec{\omega}) \frac{\Delta\Phi_p(x, \vec{\omega}'_p)}{\frac{4}{3}\pi r^3}
\end{aligned}
$$

- Gather photons in a sphere to estimate out-scattered radiance at a given point

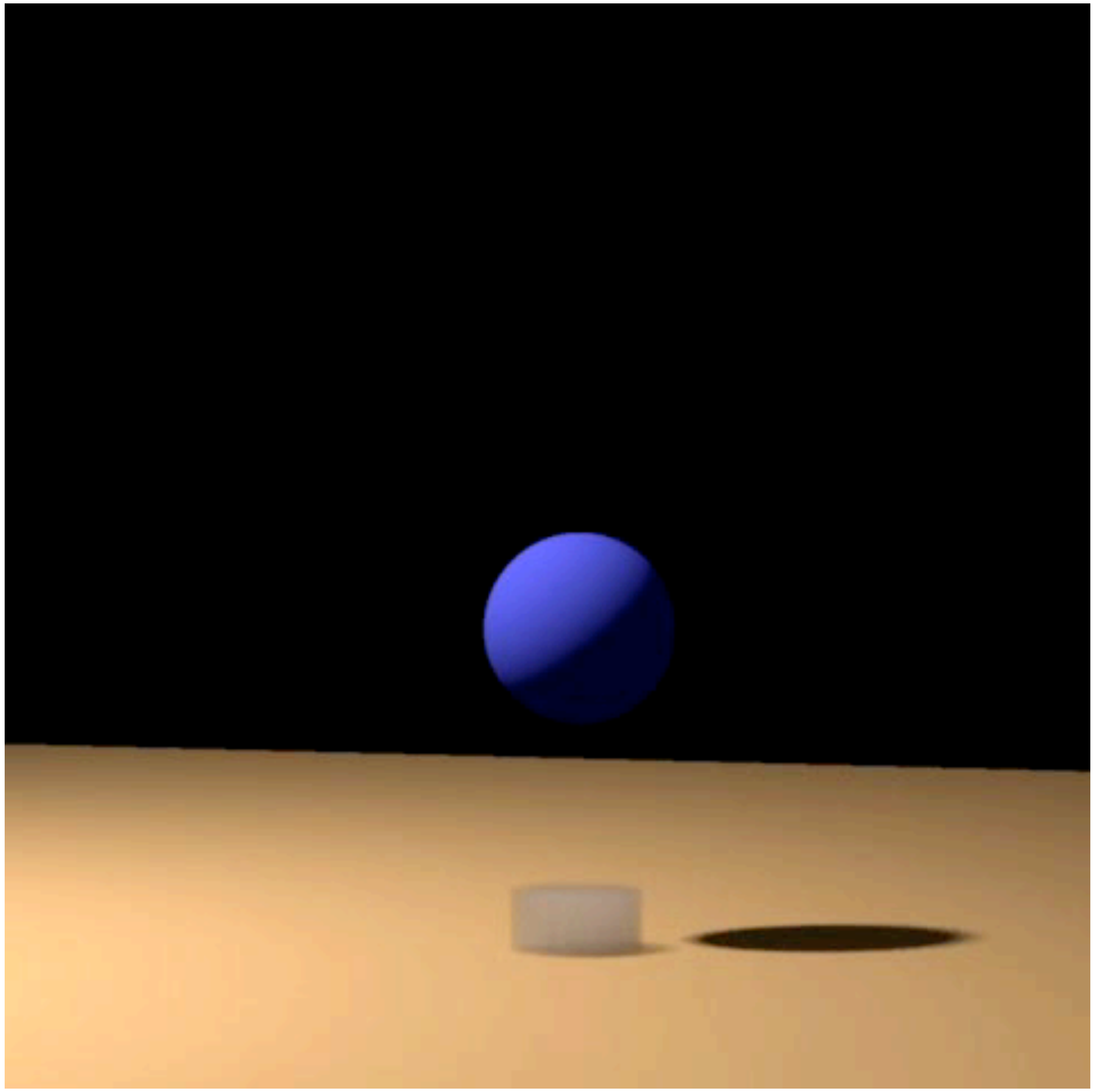
Rendering Participating Media



$$L_{n+1}(x,\vec{\omega}) = \sum_{l}^{N} L_l(x,\vec{\omega}'_l)p(x,\vec{\omega}'_l,\vec{\omega})\sigma_s(x)\Delta x + \left\{\sum_{p=1}^{n} f(x,\vec{\omega}'_p,\vec{\omega})\,\frac{\Delta\Phi_p(x,\vec{\omega}'_p)}{\frac{4}{3}\pi r^3}\right\}\Delta x + e^{-\sigma_t(x)\Delta x}L_n(x+\Delta x,\vec{\omega})\ .$$

- Direct light (single scattering) + volume radiance estimate (multiple scattering)
- Use similar technique for subsurface scattering

Smoke?



- “Visual Simulation of Smoke”, Ron Fedkiw, Jos Stam, Henrik Wann Jensen, ACM SIGGRAPH 2001
- “A Simple Fluid Solver based on the FFT”, Jos Stam, Journal of Graphics Tools 2001
- GPU Gems: Chapter 38. Fast Fluid Dynamics Simulation on the GPU, Mark Harris

Elective Assignment

- Realistic image of an object or scene
- Image, Images (movie)
  - no bigger than 1024x768
- Title and 200 words describing what, how and why it's great!
- I'll show your image, you describe what's special about it to the class.
- Due : Thursday 12pm, 22nd May (day before final lecture)

Elective Assignment

- A chance to show your creativity
- Might want to have new objects?
  - Look on web for free 3D obj models
  - Try blender/3D Max and save as obj format
- Have a look at Stanford University Rendering Competition for more ideas
  - https://graphics.stanford.edu/wikis/cs348b-08/FinalProject
  - But be careful, you only have a week!

Next

- Friday - Lab 4
- Next Monday 12th
  - Guest speaker!
- In 2 weeks, Friday 23nd May
  - Assignment 5 Due : 12pm, Thursday 22nd May
  - Rendering presentation and Course Summary