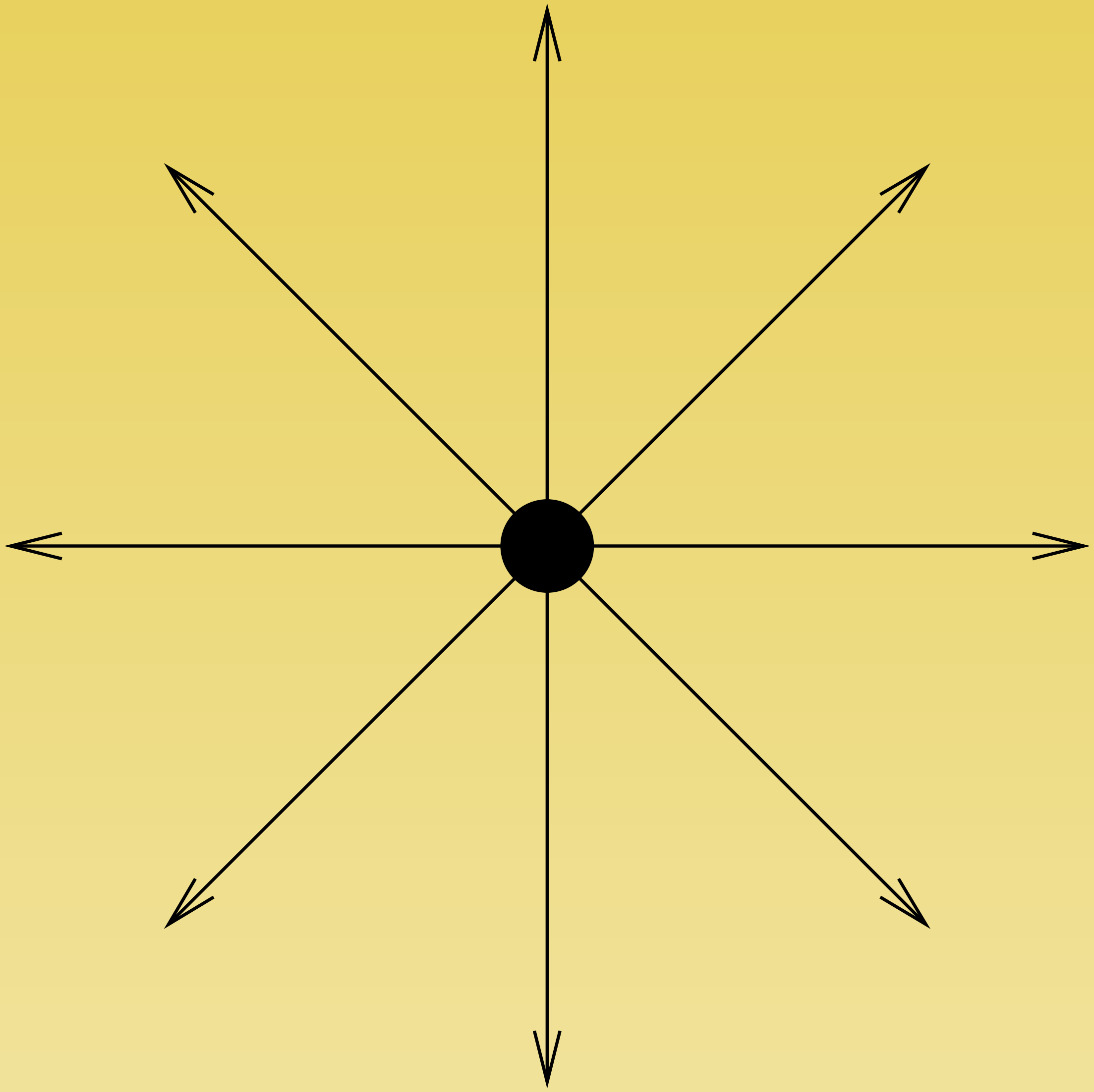
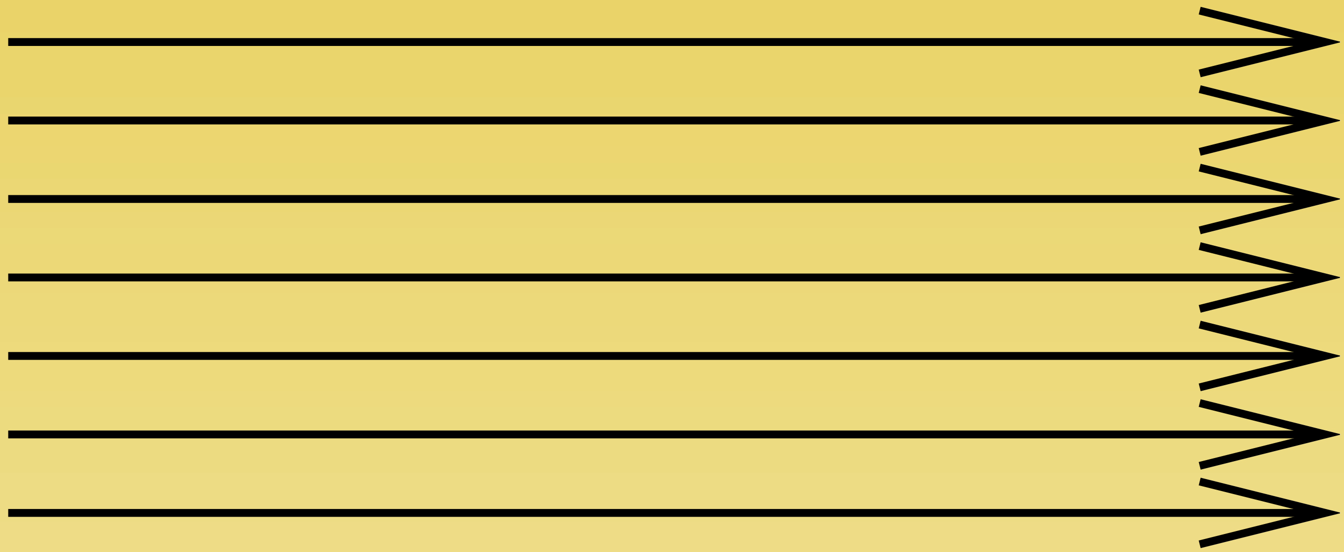
- Next week - Non-photorealistic Rendering
- Following week - animation tutorial
- By then you need some work in progress.

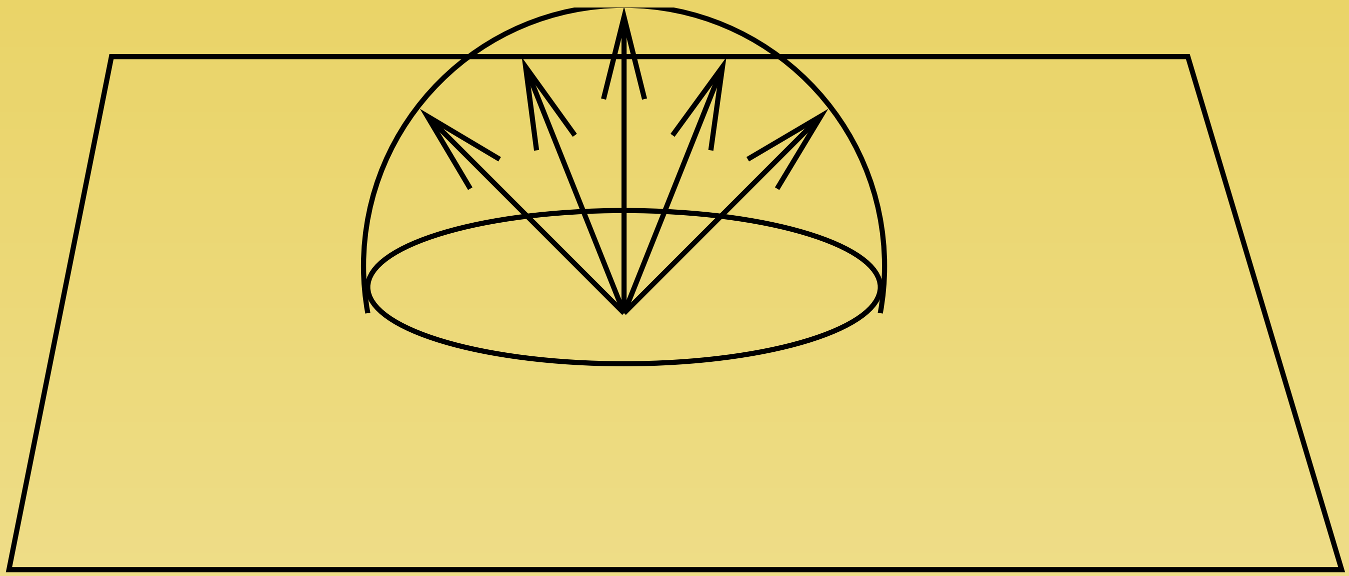
Point Light Source



Directional Source



Square Light Source



Photon Emission

The power (”wattage”) of a light is distributed to its emitted photons, n_e.

Each photon has the power:

$$P_photon = \frac{P_light}{n_e}$$

This gives the photons the same flux, don’t want to waste time on low power photons.

Photon Emission Example

```
EmitPhotonsFromDiffusePtLight() {
  NumPhotons = 0;
  while (NumPhotons <= TotalPhotons) {
     Dir = <rand, rand, rand>
     Start = PtLight.location;
     TraceRay(Start, Dir);  // Trace the photon
     NumPhotons++;
  }
  scale power of stored photons with 1/NumPhotons;
}
```

Photon Emission

- Multiple Lights
  - ⋆ Emit photons from each source.
  - ⋆ Emit more photons from brighter lights, (power of all photons should be even)
- Sparse geometry. Use projection maps. (don't waste photons that don't hit)
- Trace a photon (similar to a ray).
  - ⋆ Photons propagate flux.

Tracing Photons

When photon hits surface: reflect, transmit, absorb.
Probabilistically by Russian roulette (random death).
This allows all photons to have the same power.

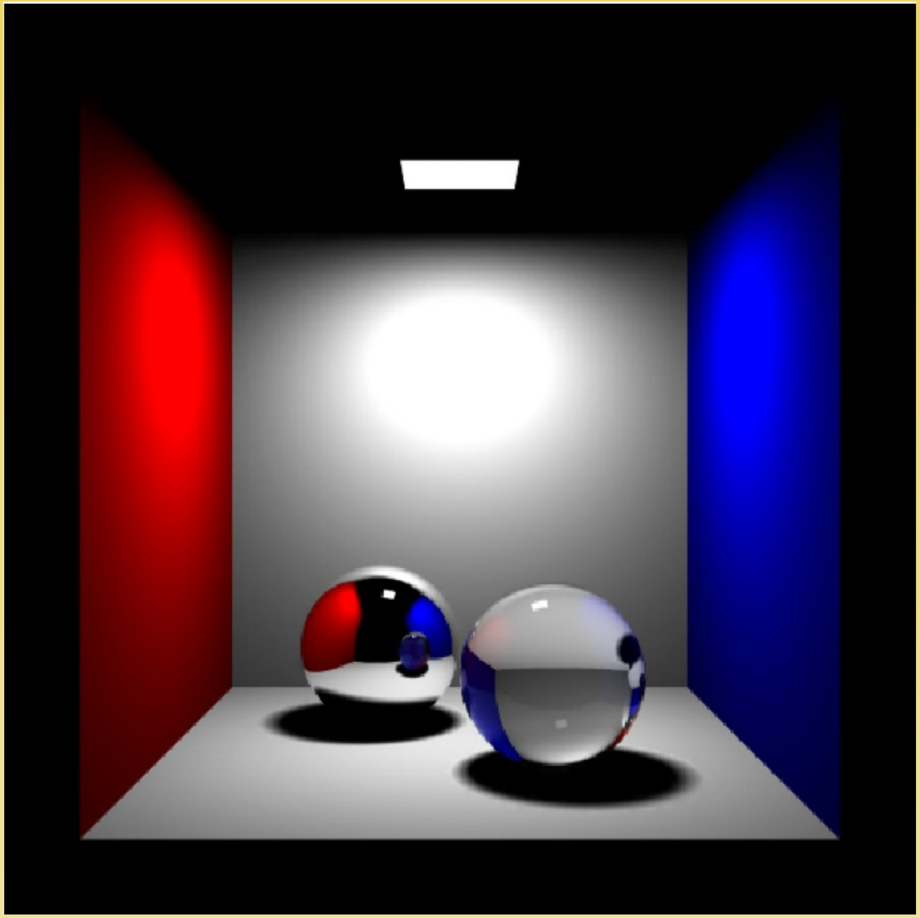
Diffuse coefficient d,
specular coefficent s,

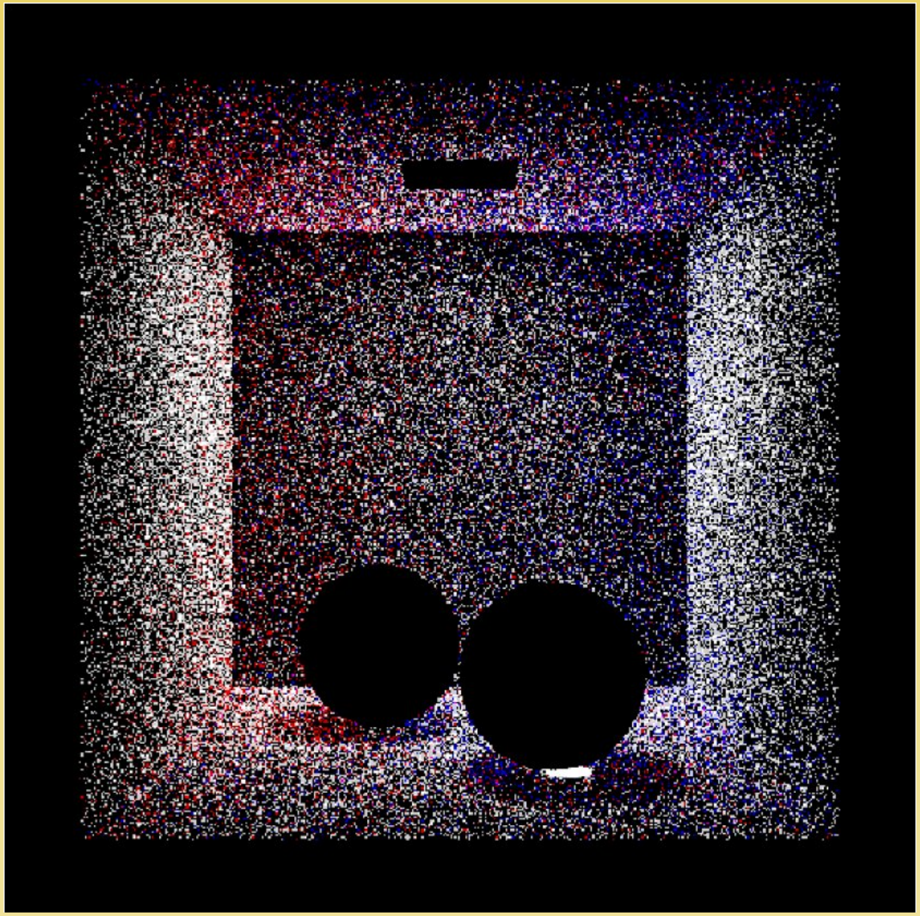
$$d + s \leq 1$$

$$\xi = rand, \qquad \xi \in [0, 1]$$

$\xi \in [0, d]$	diffuse reflection
$\xi \in (d, s + d]$	specular reflection
$\xi \in (s + d, 1]$	absorption

Cornell Box Example





Storing the Photons

- Store where photons hits non-specular surfaces.
- May be more than one per photon.
- Store position, photon power, incident direction.
- Use volume map if participating media

```
struct Photon {
 float x,y,z;    // position
 char p[4];      // power, packed
 char phi, theta;// incident direction
 short flag;     // flag used in kdtree
}
```

Photon Maps

- Caustic photon map: Photons with $>= 1$ specular reflections. $LS + D$ (should be more dense)
- Global photon map: all diffuse surfaces $L\{S|D|V\}*D$
- Volume photon map: participating medium $L\{S|D|V\}+V$
- Build with separate photon tracing steps.

Storing the photon map

- Build a kd-tree.
- Multi-dimensional binary search tree.
- Each node divides space along one axis, left branch on one side and right branch on the other.
- $O(N)$ worst-case search, $O(logN)$ average.
- Light in a scene can be very coherent so the tree tends to get skewed.
- The balanced kd-tree can be searched in $O(logN)$
- Worthwhile to balance the tree.

Estimating Radiance

$$L_r(x,\omega) = \int_{\Omega_r} f_r(x,\omega',\omega) L_i(x,\omega') |n_x.\omega'| d\omega_i'$$

where L_r - reflected radiance at x in direction ω. Ω_r is the hemisphere of incoming directions, f_r is the BRDF at x and L_i is the incoming radiance.

We can get the flux from the photon map and

$$L_i(x,\omega') = \frac{d^2\phi_i(x,\omega')}{\cos\theta_i d\omega_i' dA_i}$$

relates the flux to radiance.

Estimating Radiance

$$L_r(x,\omega) = \int_{\Omega_r} f_r(x,\omega',\omega) L_i(x,\omega') |n_x.\omega'| d\omega'_i$$

$$L_i(x,\omega') = \frac{d^2\phi_i(x,\omega')}{\cos\theta_i d\omega'_i dA_i}$$

$$L_r(x,\omega) = \int_{\Omega_r} f_r(x,\omega',\omega) \frac{d^2\phi_i(x,\omega')}{\cos\theta_i d\omega'_i dA_i} |n_x.\omega'| d\omega'_i$$

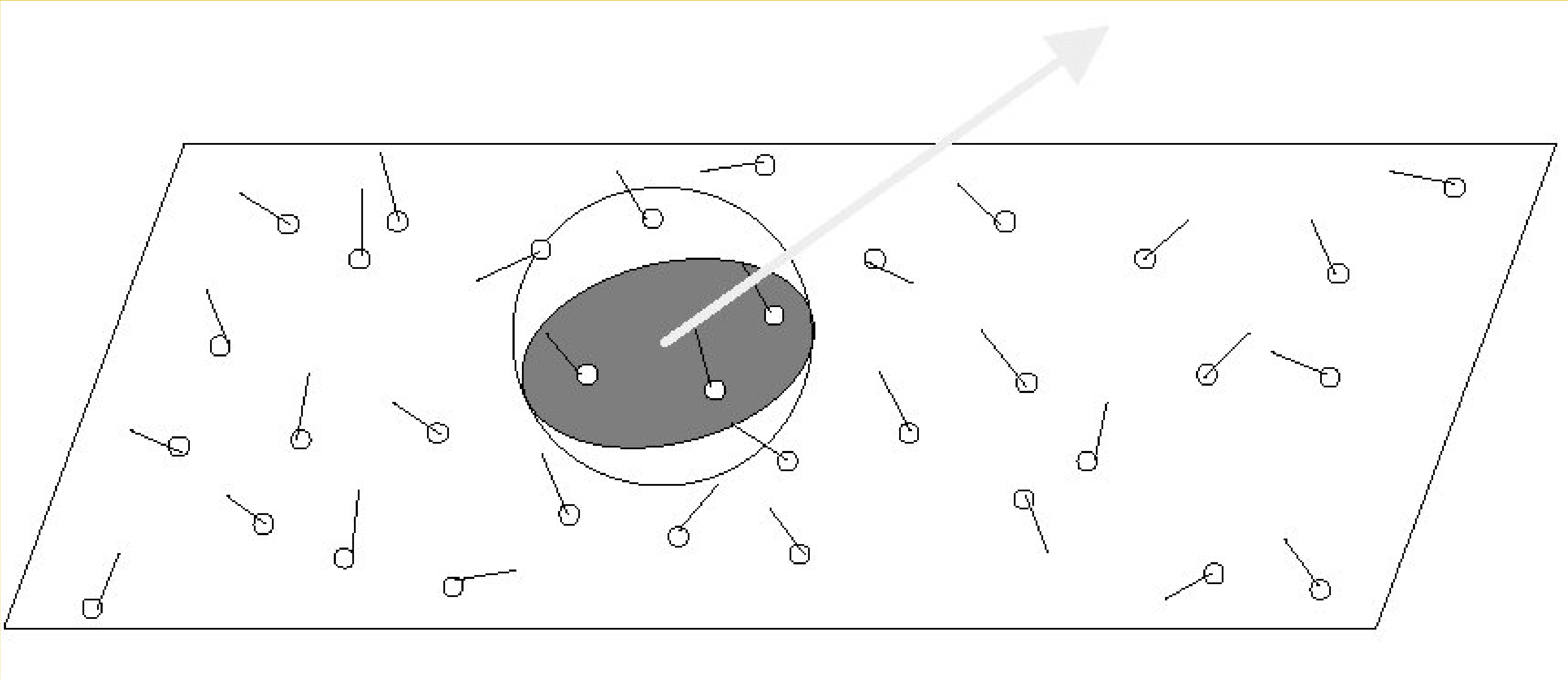
$$= \int_{\Omega_r} f_r(x,\omega',\omega) \frac{d^2\phi_i(x,\omega')}{dA_i}$$

Incoming Flux

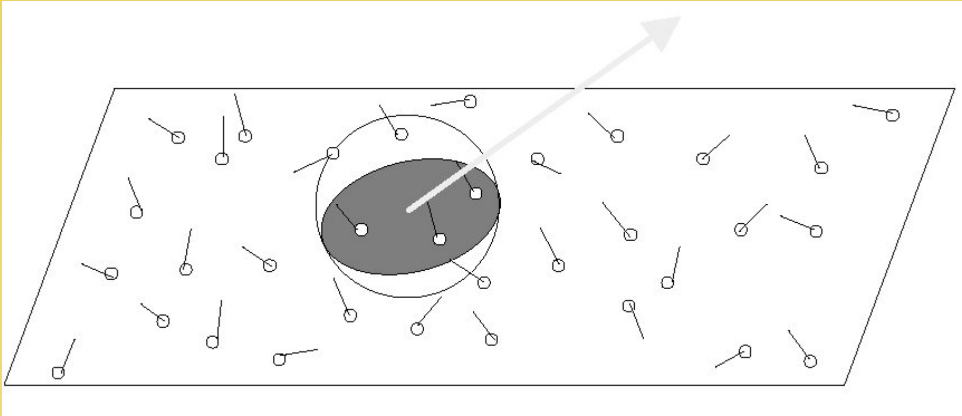
- The incoming flux ϕ_i can be estimated with the photon map.
- Locate the n photons with the shortest distance to x.
- Each photon p has the power $\Delta\phi_p(\omega_p)$
- Assume the photon intersects the surface as x gives

$$L_r(x,\omega) \approx \sum_{p=1}^{n} f_r(x,\omega_p,\omega)\frac{\Delta\phi_p(x,\omega_p)}{\Delta A_i}$$

Radiance Estimate



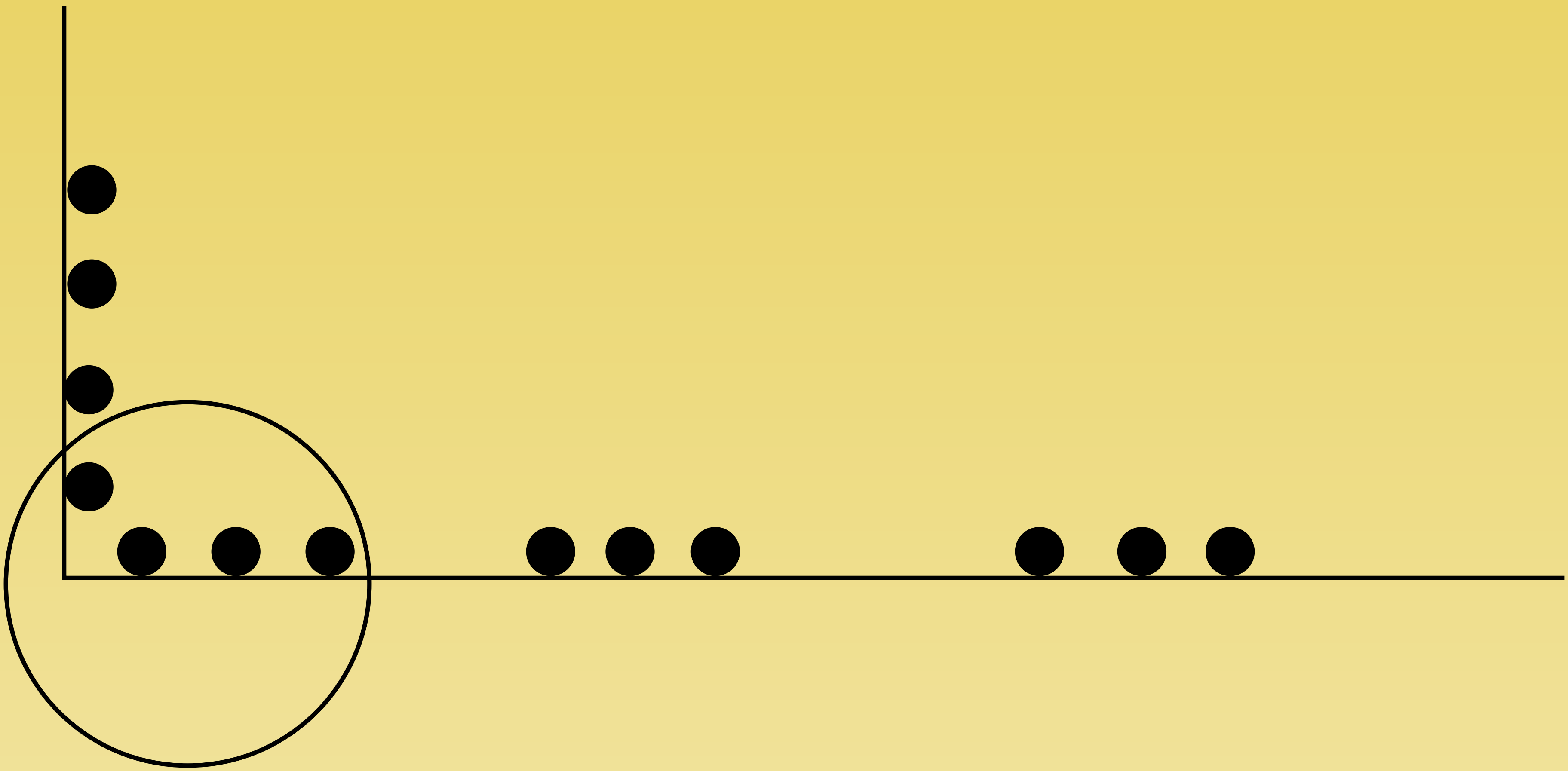
Radiance Estimate



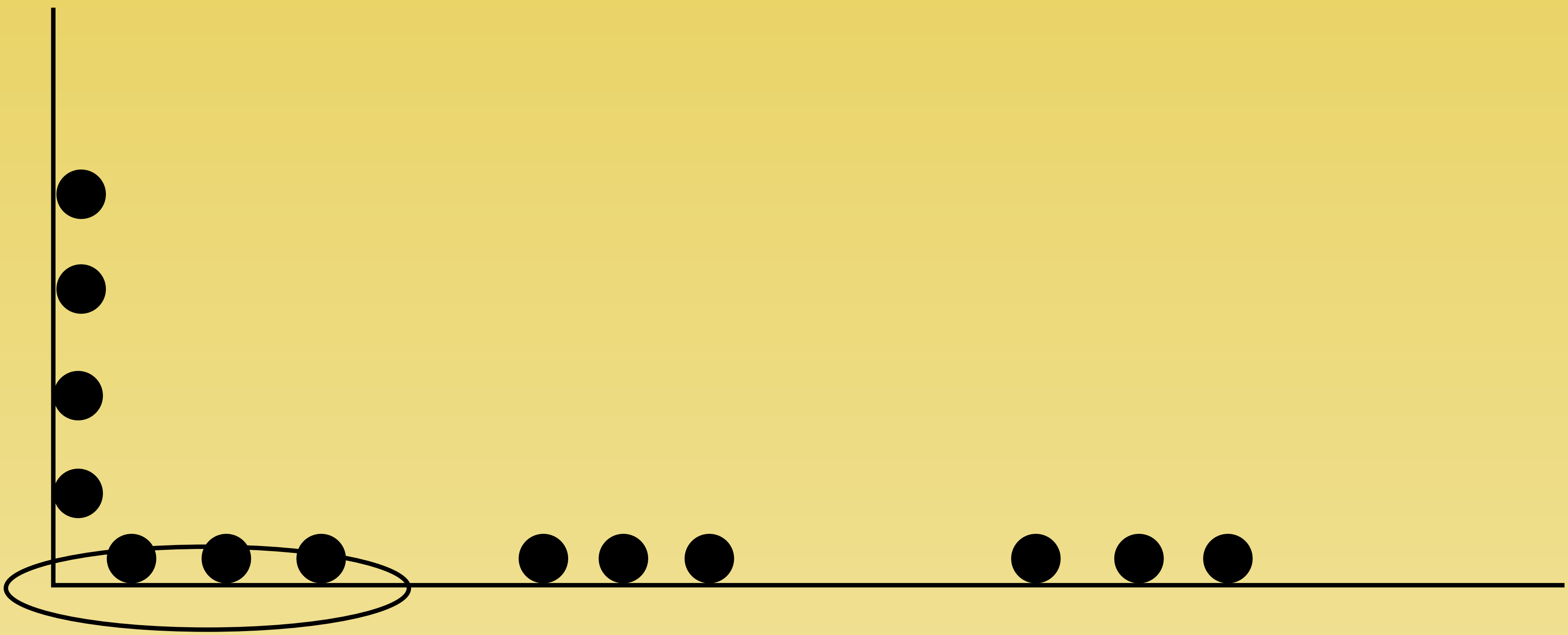
Project sphere onto surface, so the area $\Delta A = \pi r^2$

$$L_r(x,\omega) \approx \frac{1}{\pi r^2} \sum_{p=1}^{n} f_r(x,\omega_p,\omega)\Delta\phi_p(x,\omega_p)$$

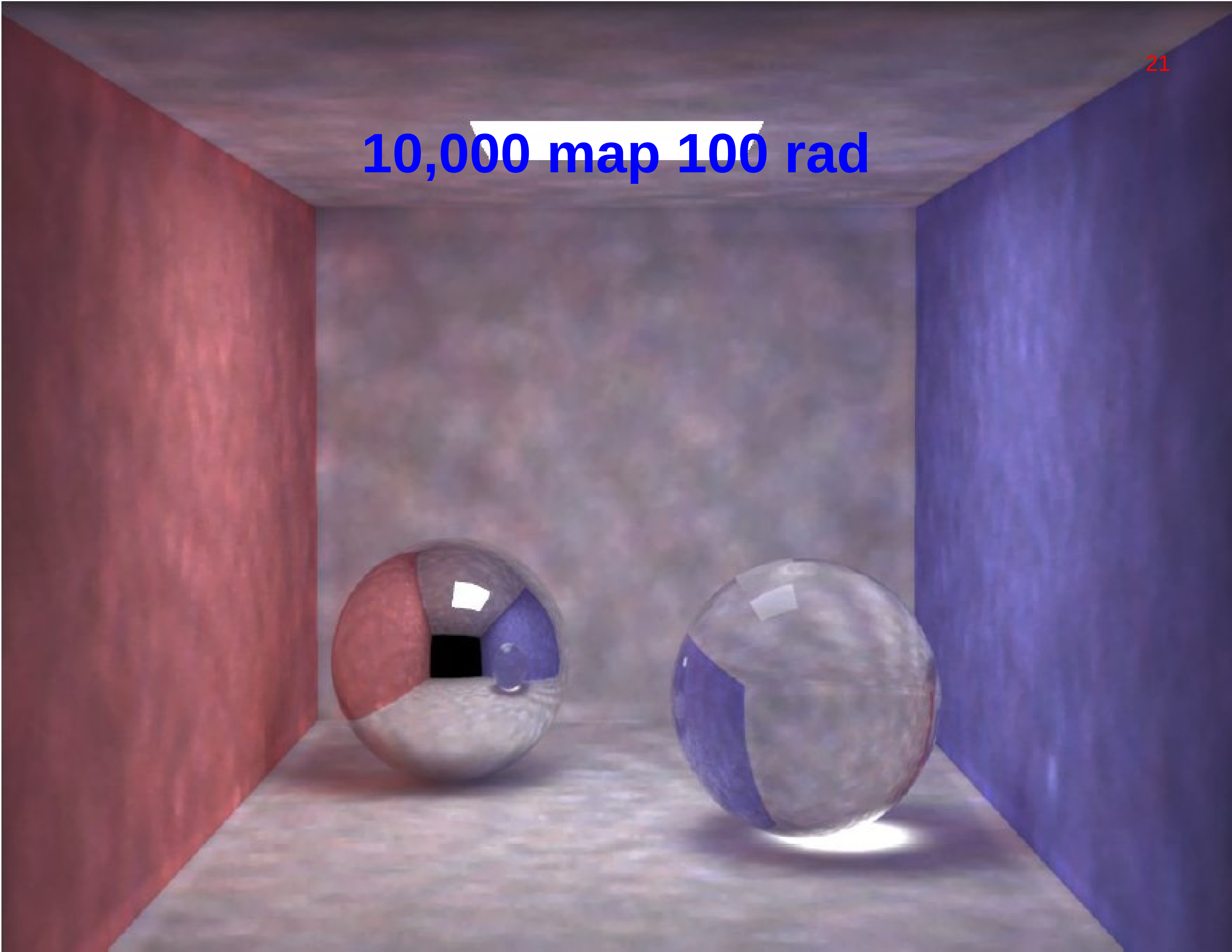
Radiance Estimate Problems



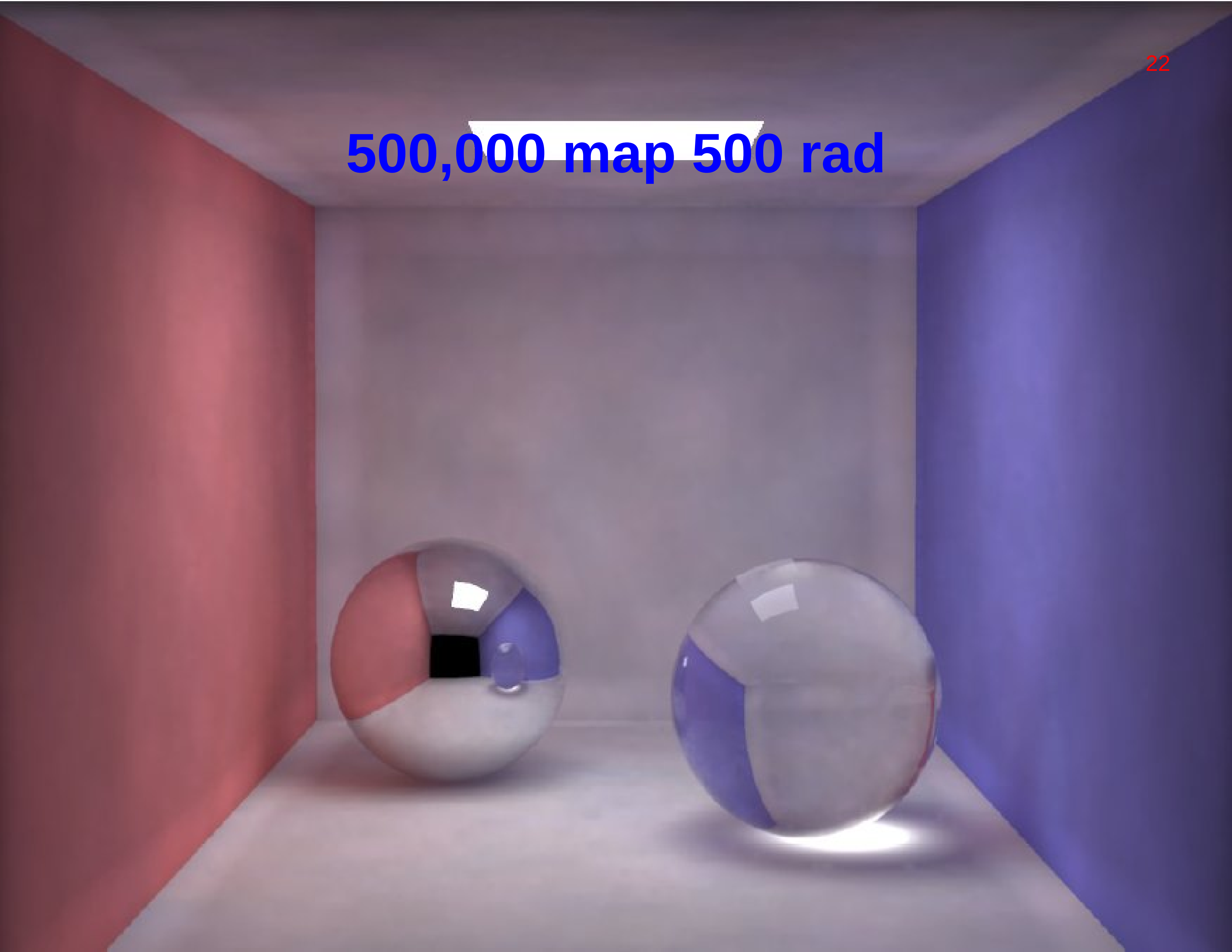
Radiance Estimate Problems



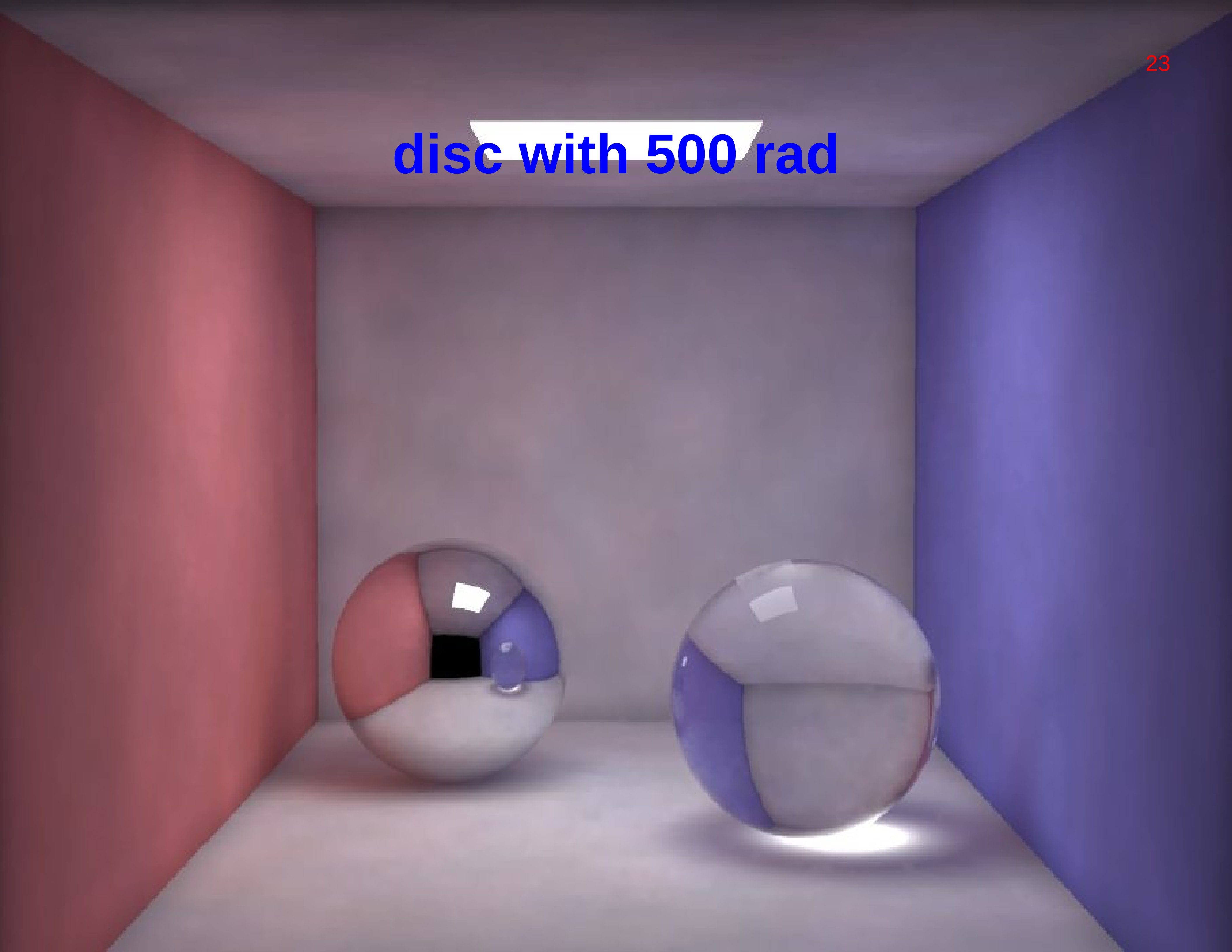
10,000 map 100 rad



500,000 map 500 rad



disc with 500 rad



Filtering

- The sphere is fast.
- Disk is more accurate.
- ΔA depends on volume.
- If ΔA is wrong or not enough photons in either the map or the estimate or blending occurs, there will be errors.
- Fix with a 2D filter weighting photons based on distance.
- Cone filter or Gaussian filter.

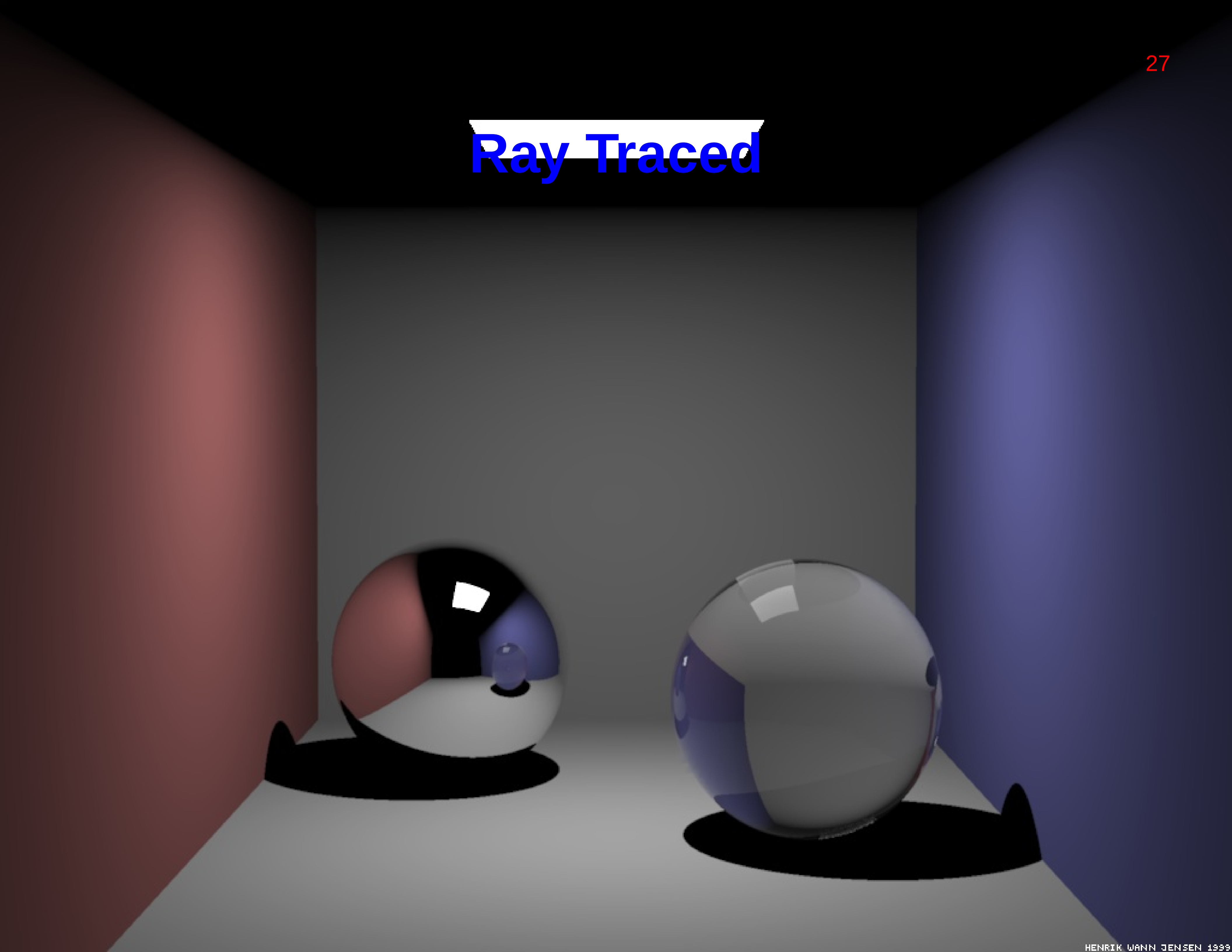
Rendering

- Incoming radiance = Direct illumination (from lights) + caustics (indirect illumination from light sources via specular interactions) + indirect illumination (diffusely reflected)
- Direct illumination done with ray tracing as are shadows.
- Specular and glossy reflections done with normal ray tracing (monte carlo actually)
- Shadow rays can be done with shadow photons or shadow maps for speed.

Rendering

- Incoming radiance = Direct illumination (from lights) + caustics (idirect illumination from light sources via specular interactions) + indirect illumination (diffusely reflected)
- Direct illumination done with ray tracing as are shadows.
- Specular and glossy reflections done with normal ray tracing (monte carlo actually)
- Shadow rays can be done with shadow photons or shadow maps for speed.

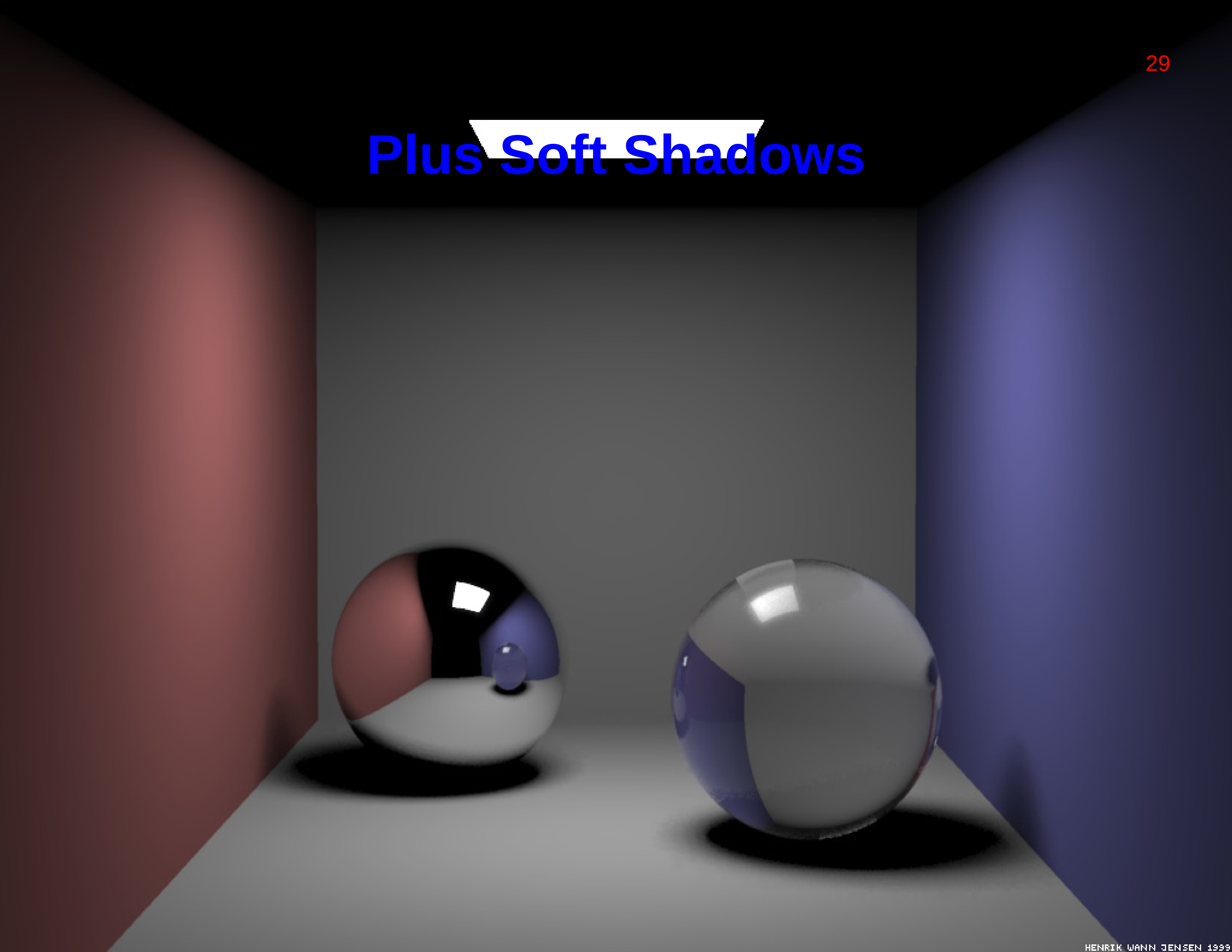
Ray Traced



Rendering

- Incoming radiance = Direct illummination (from lights) + caustics (idirect illummination from light sources via specular interactions) + indirect illumination (diffusely reflected)
- Direct illumination done with ray tracing as are shadows.
- Specular and glossy reflections done with normal ray tracing (monte carlo actually)
- Shadow rays can be done with shadow photons or shadow maps for speed.

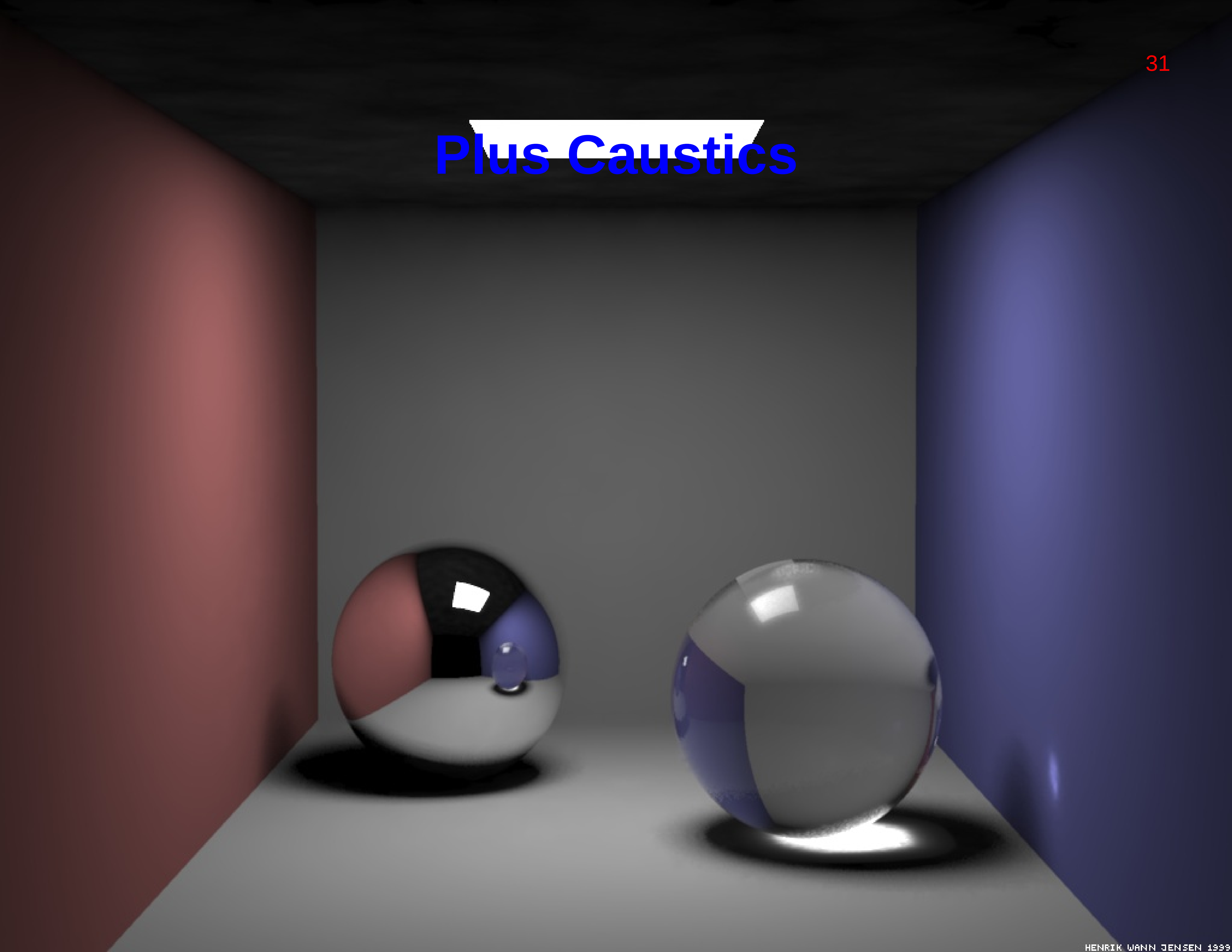
Plus Soft Shadows



Rendering

- Caustics are found by sampling the caustic photon map. High number of photons in the caustic map are required for sharp caustics. The global photon map can give an approximation.
- Mulitple diffuse reflections. Radiosity or the global photon map which is much faster.
- Participating media can be done with the volume map.

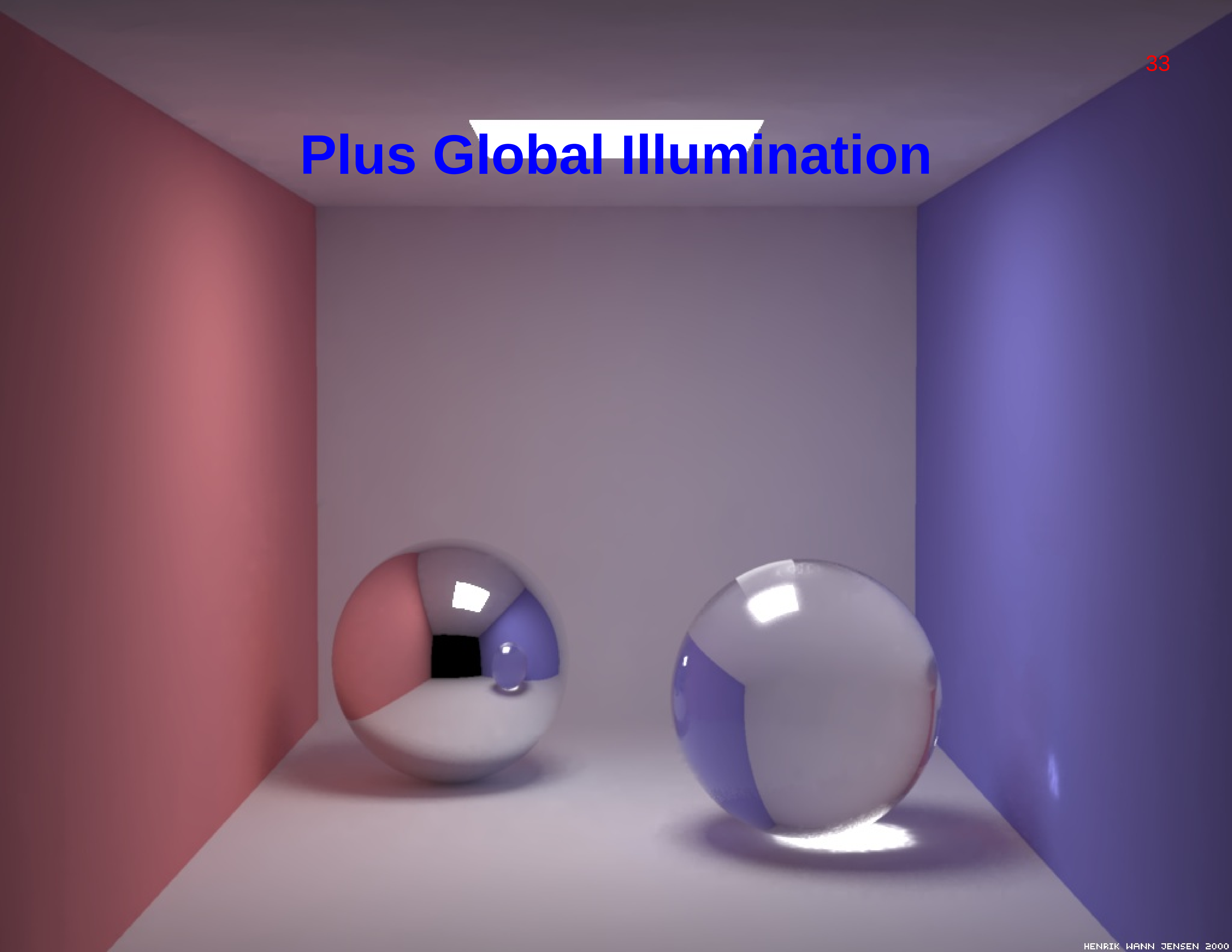
Plus Caustics



Rendering

- Caustics are found by sampling the caustic photon map. High number of photons in the caustic map are required for sharp caustics. The global photon map can give an approximation.
- Mulitple diffuse reflections. Radiosity or the global photon map which is much faster.
- Participating media can be done with the volume map.

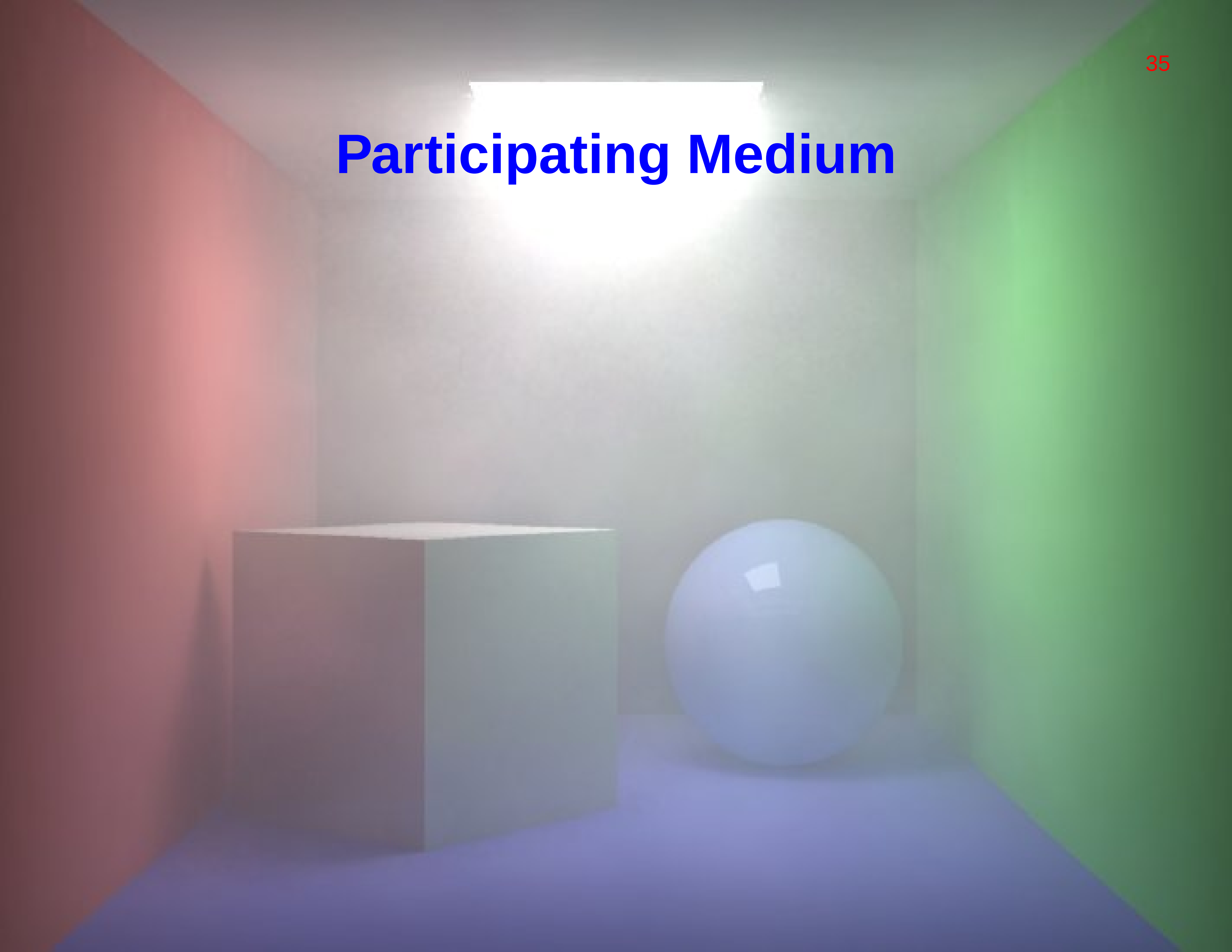
Plus Global Illumination



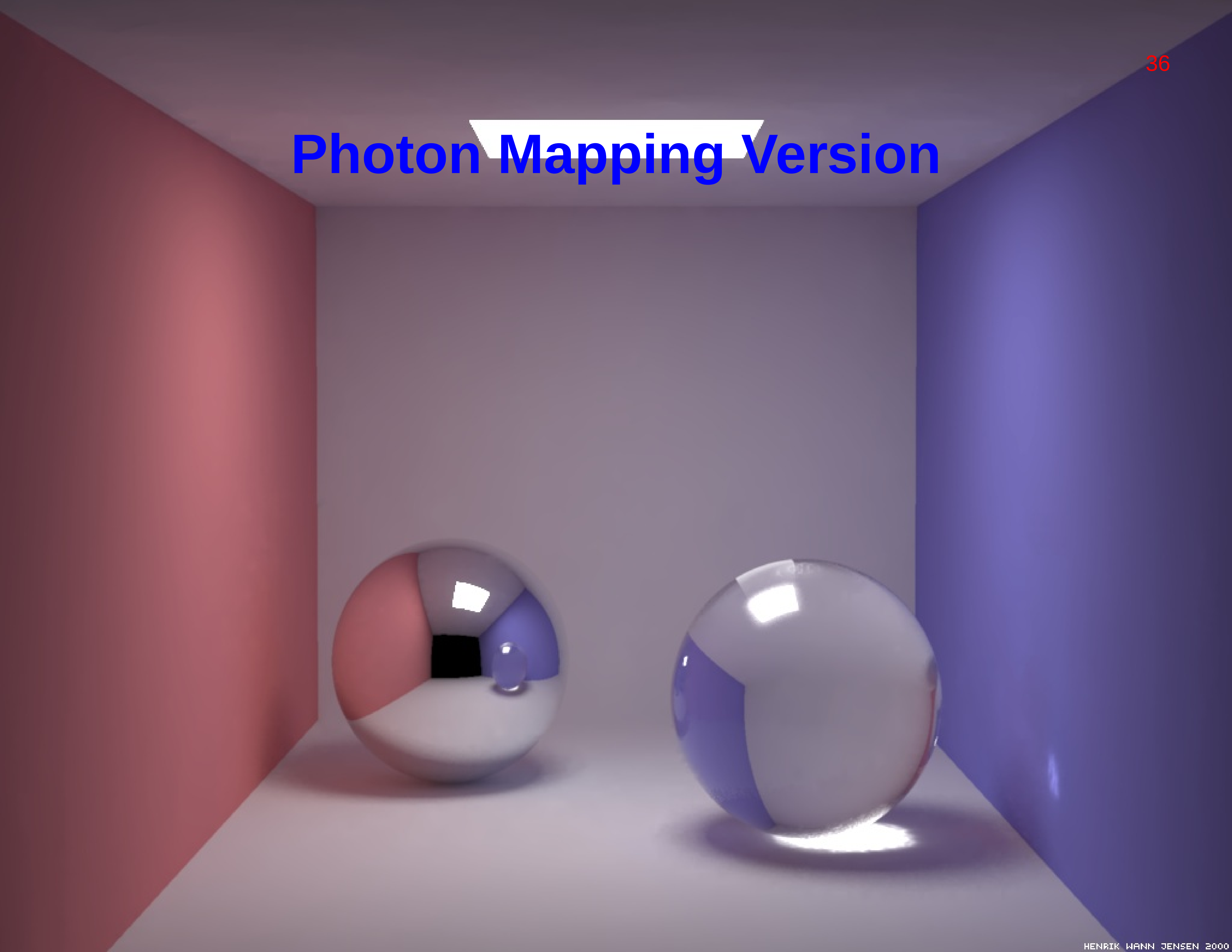
Rendering

- Specular and glossy reflections done with normal ray tracing (monte carlo actually)
- Caustics are found by sampling the caustic photon map. High number of photons in the caustic map are required for sharp caustics. The global photon map can give an approximation.
- Mulitple diffuse reflections. Radiosity, or the global photon map which is much faster.
- Participating media can be done with the volume map.

Participating Medium

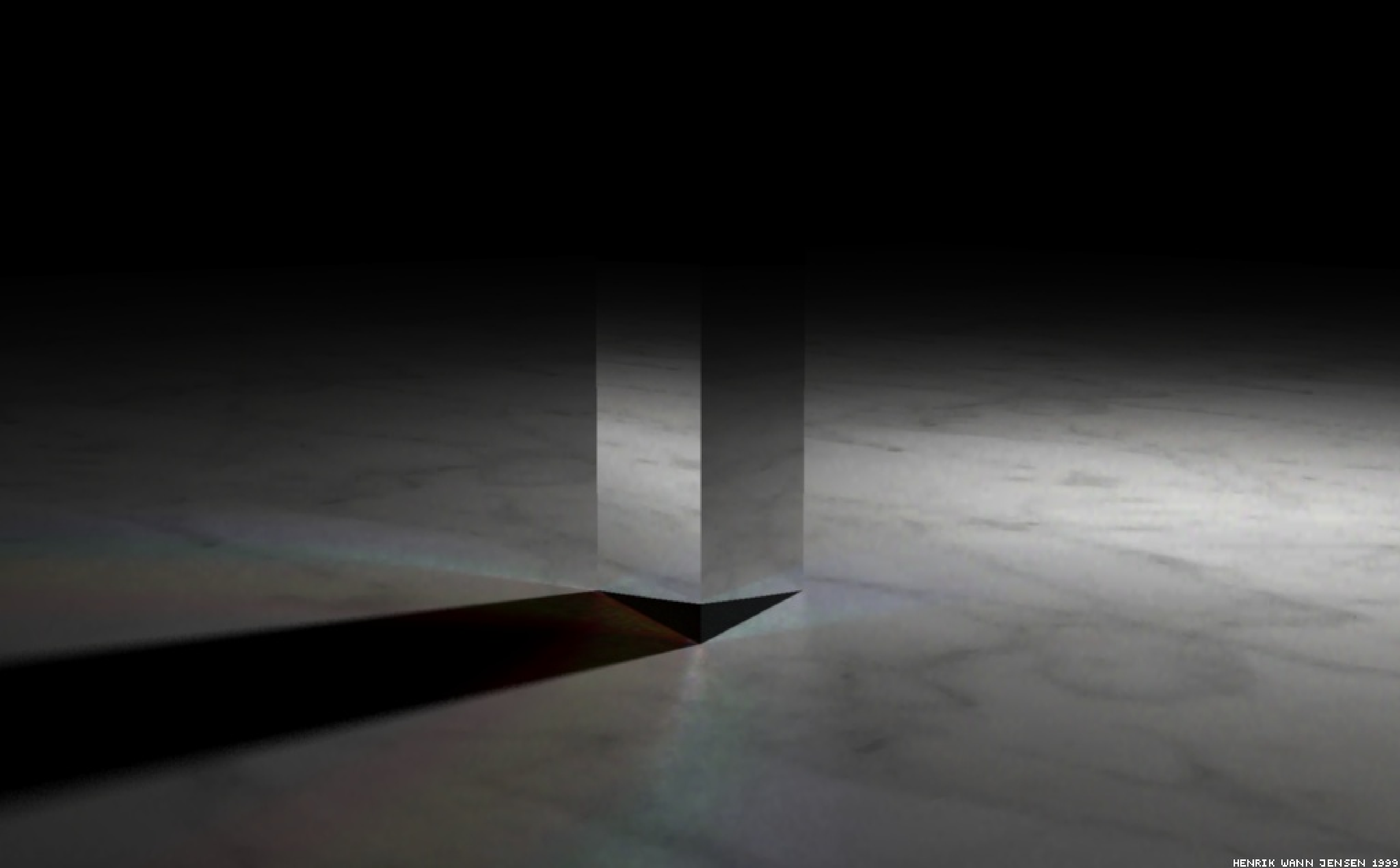


Photon Mapping Version

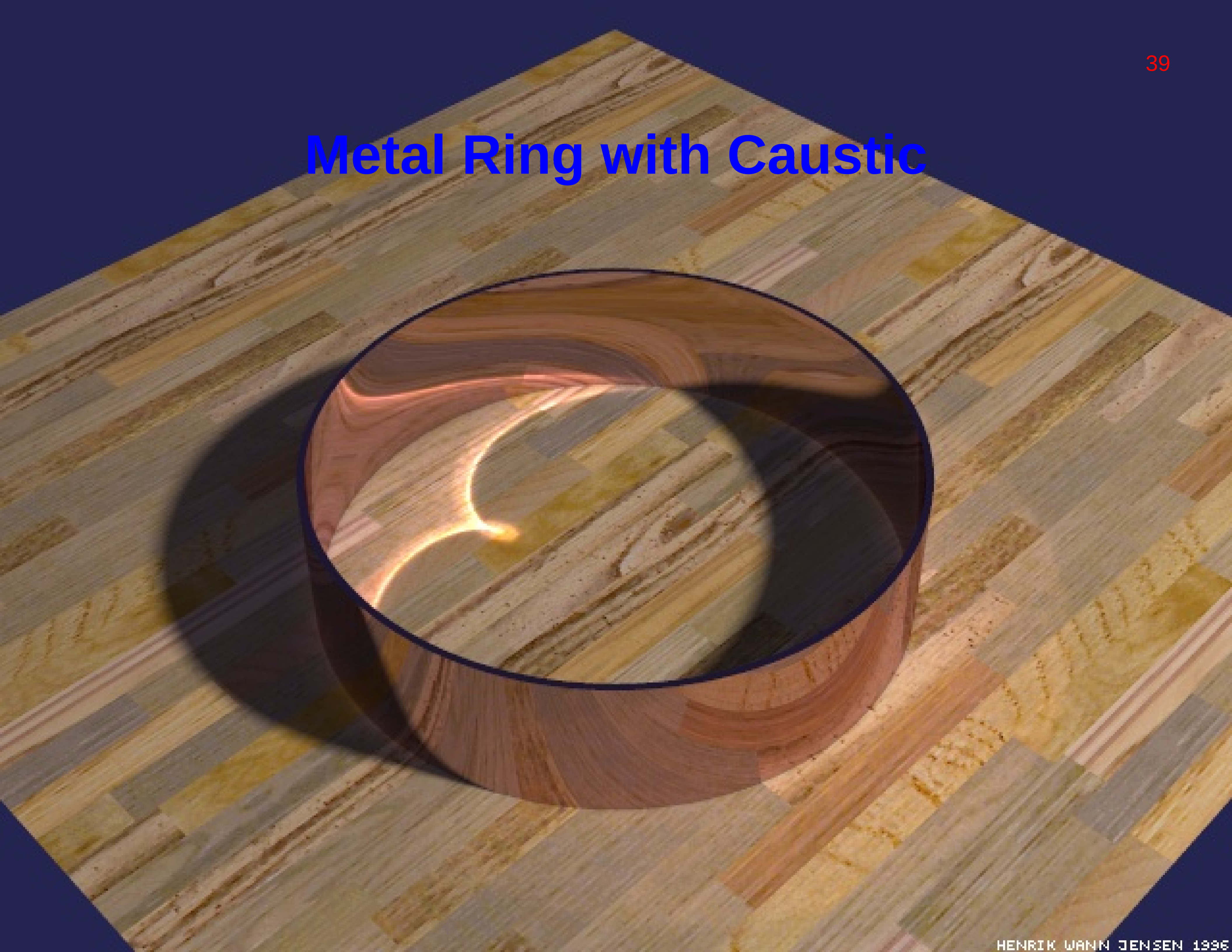


Path Tracing Version

Prism with dispersion



Metal Ring with Caustic



Results