Photon Mapping

Thanks to Henrik Wann Jensen, UCSD

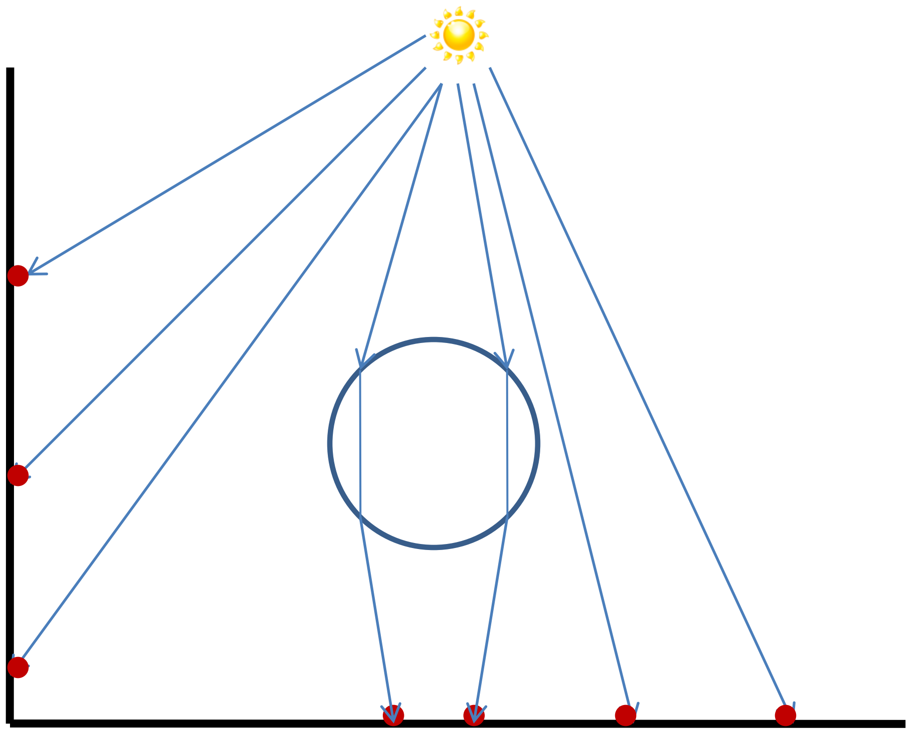
Photon mapping

A two-pass method

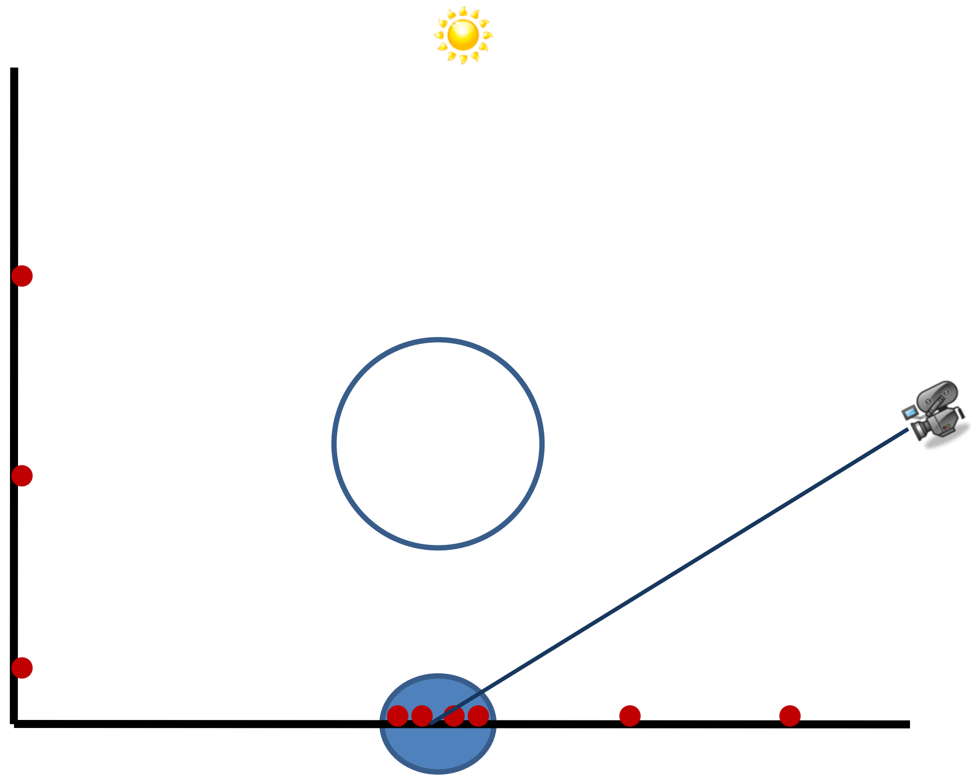
Pass 1: Build the photon map (photon tracing)

Pass 2: Render the image using the photon map

Building the photon map: Photon tracing



Rendering using the photon map



What is a photon?

- Flux (power) - not radiance!
- Collection of physical photons
  - A fraction of the light source power
  - Several wavelengths combined into one entity

Photon emission

Given Φ Watt lightbulb.
Emit N photons.
Each photon has the power $\frac{\Phi}{N}$ Watt.

- Photon power depends on the number of emitted photons. Not on the number of photons in the photon map.

Diffuse point light

Generate random direction
Emit photon in that direction

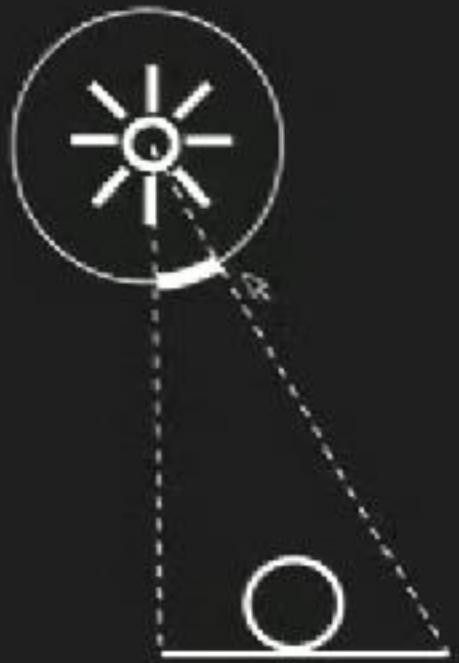
```
// Find random direction
do {
  x = 2.0*random()-1.0;
  y = 2.0*random()-1.0;
  z = 2.0*random()-1.0;
} while ( (x*x + y*y + z*z) > 1.0 );
```

Example: Diffuse square light

- Generate random position p on square
- Generate diffuse direction d
- Emit photon from p in direction d

```
// Generate diffuse direction
u = random();
v = 2*π*random();
d = vector( cos(v)√u, sin(v)√u, √(1-u) );
```

Projection maps



Surface interactions

The photon is

- Stored (at diffuse surfaces) and
- Absorbed (A) or
- Reflected (R) or
- Transmitted (T)

$$A + R + T = 1.0$$

Storing the photon

```
struct photon {
  float x,y,z;          // position
  char p[4];            // power packed as 4 bytes
  char phi,theta;       // incident direction
  short flag;           // flag used for kd-tree
}
```

Memory overhead: 20 bytes/photon.

Photon scattering

The simple way:

Given incoming photon with power Φ_p

Reflect photon with the power $R * \Phi_p$

Transmit photon with the power $T * \Phi_p$

- Risk: Too many low-powered photons - wastefull!
- When do we stop (systematic bias)?
- Photons with similar power is a good thing.

Russian Roulette

- Statistical technique
- Known from Monte Carlo particle physics
- Introduced to graphics by Arvo and Kirk in 1990

Terminate un-important photons and still get the correct result.

Russian Roulette Example

Surface reflectance: $R = 0.5$
Incoming photon: $\Phi_p = 2$ W

```
r = random();
if ( r < 0.5 )
  reflect photon with power 2 W
else
  photon is absorbed
```

Reflect 100 photons with power 2 Watt instead of 200 photons with power 1 Watt.

Russian Roulette Example 2

Surface reflectance: R = 0.2
Surface transmittance: T = 0.3
Incoming photon: Φ_p = 2 W

```
r = random();
if ( r < 0.2 )
  reflect photon with power 2 W
else if ( r < 0.5 )
  transmit photon with power 2 W
else
  photon is absorbed
```

Sampling a BRDF

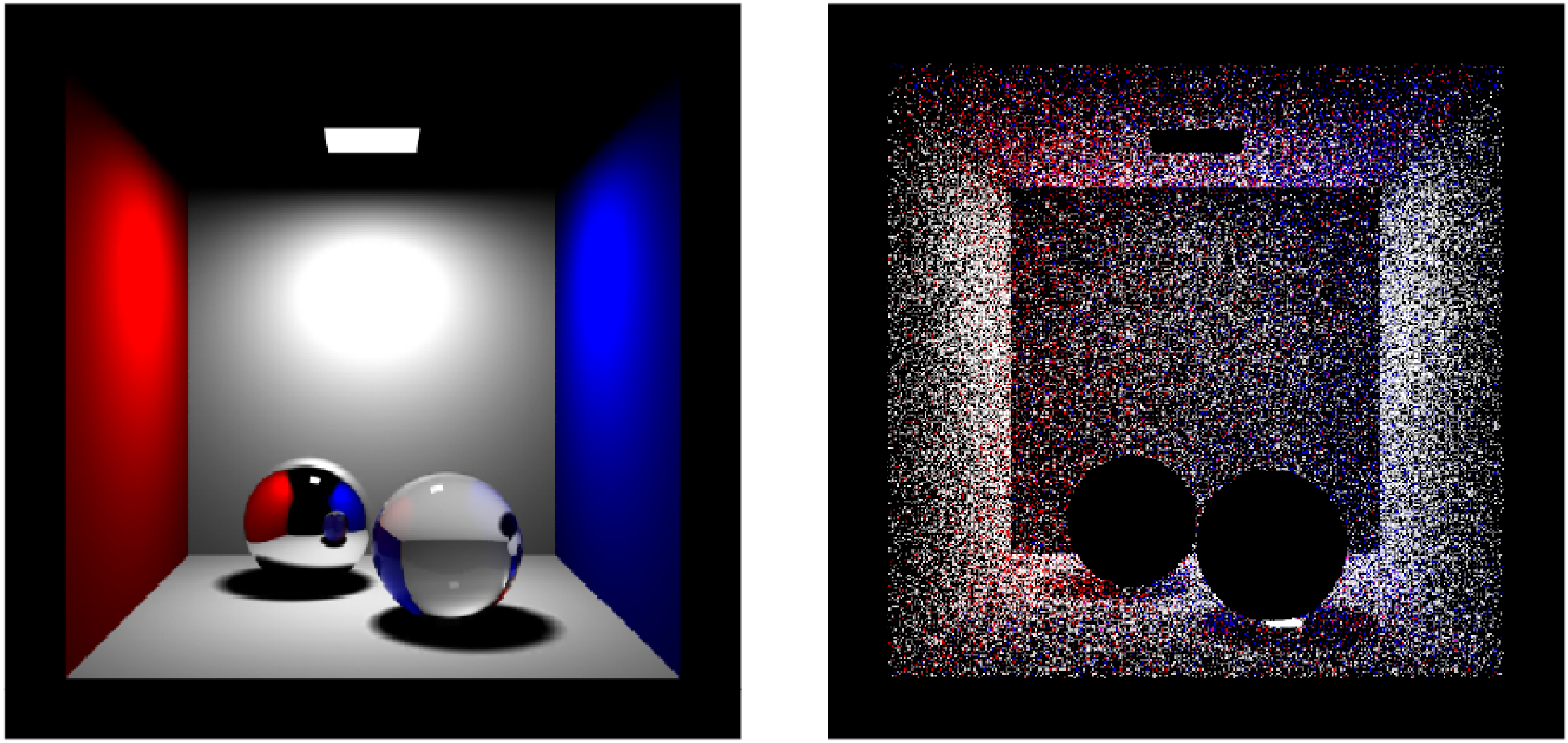
$$f_r(x, \vec{\omega}_i, \vec{\omega}_o) = w_1 \cdot f_{r,d} + w_2 \cdot f_{r,s}$$

```
r = random()·(w1 + w2);
if ( r < w1 )
  reflect diffuse photon
else
  reflect specular
```

Photon tracing

Overview:

```
While (we want more photons) {
  Emit a photon
  while (photon hits a surface) {
   Store photon
   Use Russian Roulette to scatter photon
  }
}
Build balanced kd-tree
```



(a) (b)

Figure 4.4: “Cornell box” with glass and chrome spheres: (a) ray traced image (direct illumination and specular reflection and transmission), (b) the photons in the corresponding photon map.

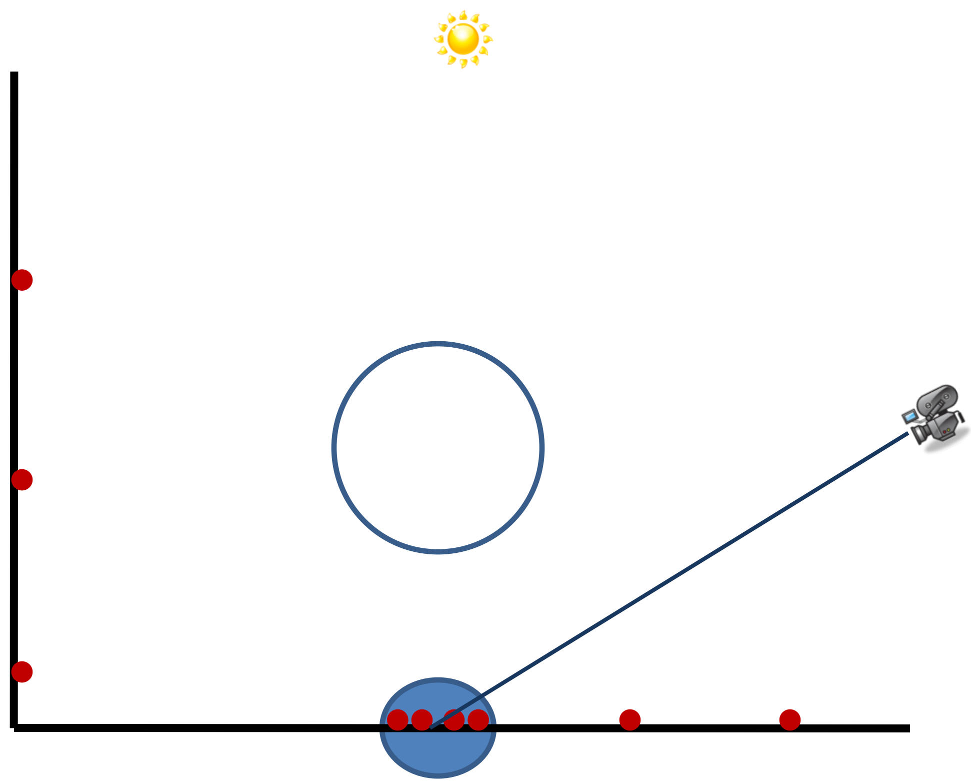
Photon mapping

A two-pass method

Pass 1: Build the photon map (photon tracing)

Pass 2: Render the image using the photon map

Rendering using the photon map



Rendering

We want a Radiance value, L, per pixel.

The photon map stores flux/power.

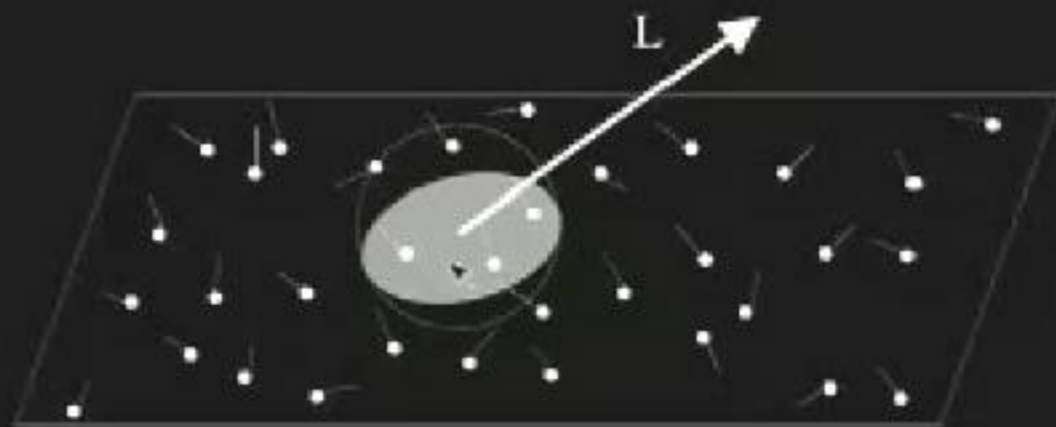
Radiance is the differential flux per differential solid angle per differential cross-sectional area:

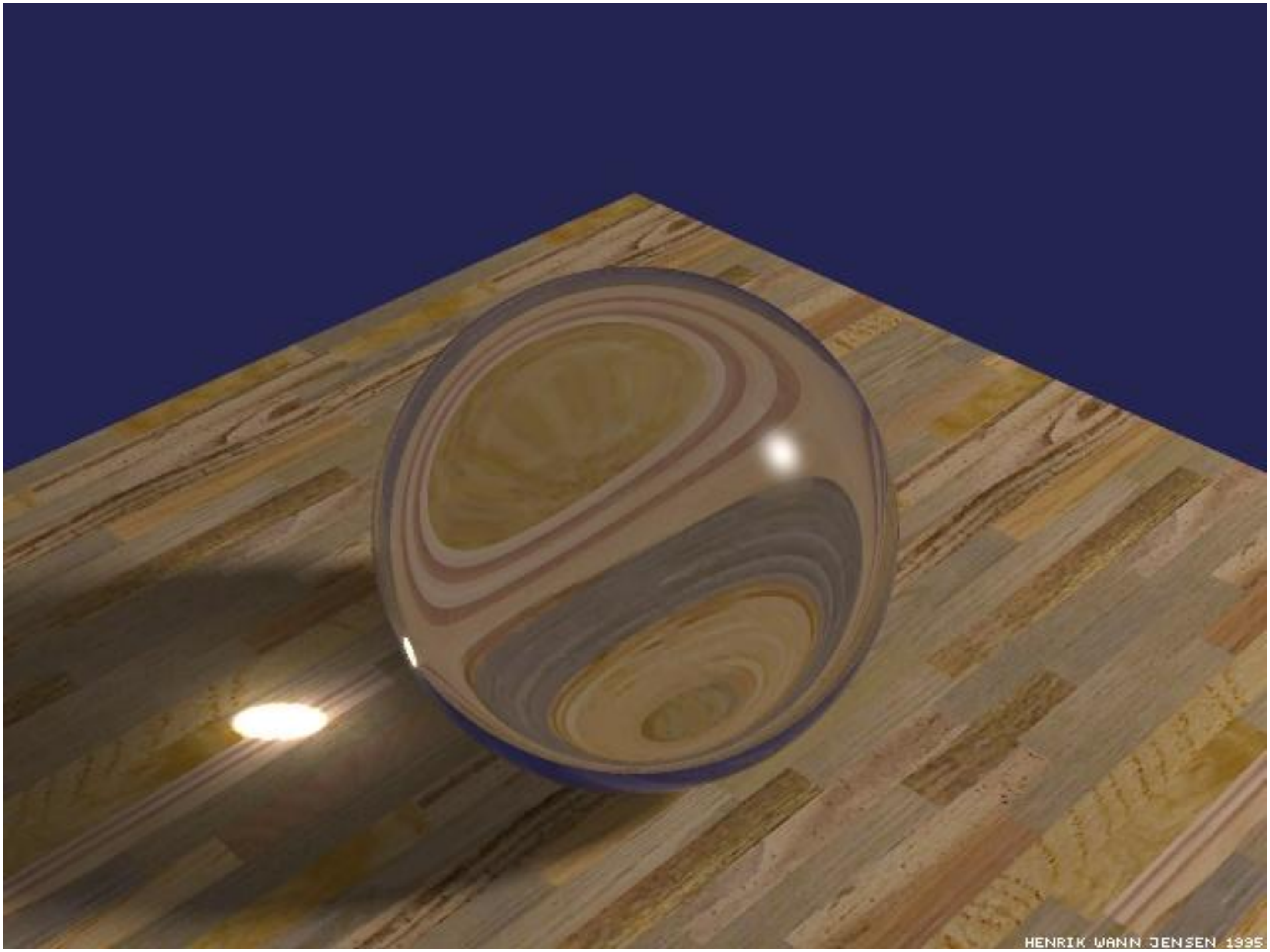
$$L(x, \vec{\omega}) = \frac{d\Phi^2(x, \vec{\omega})}{d\omega \, \cos\theta \, dA}$$

Radiance estimate

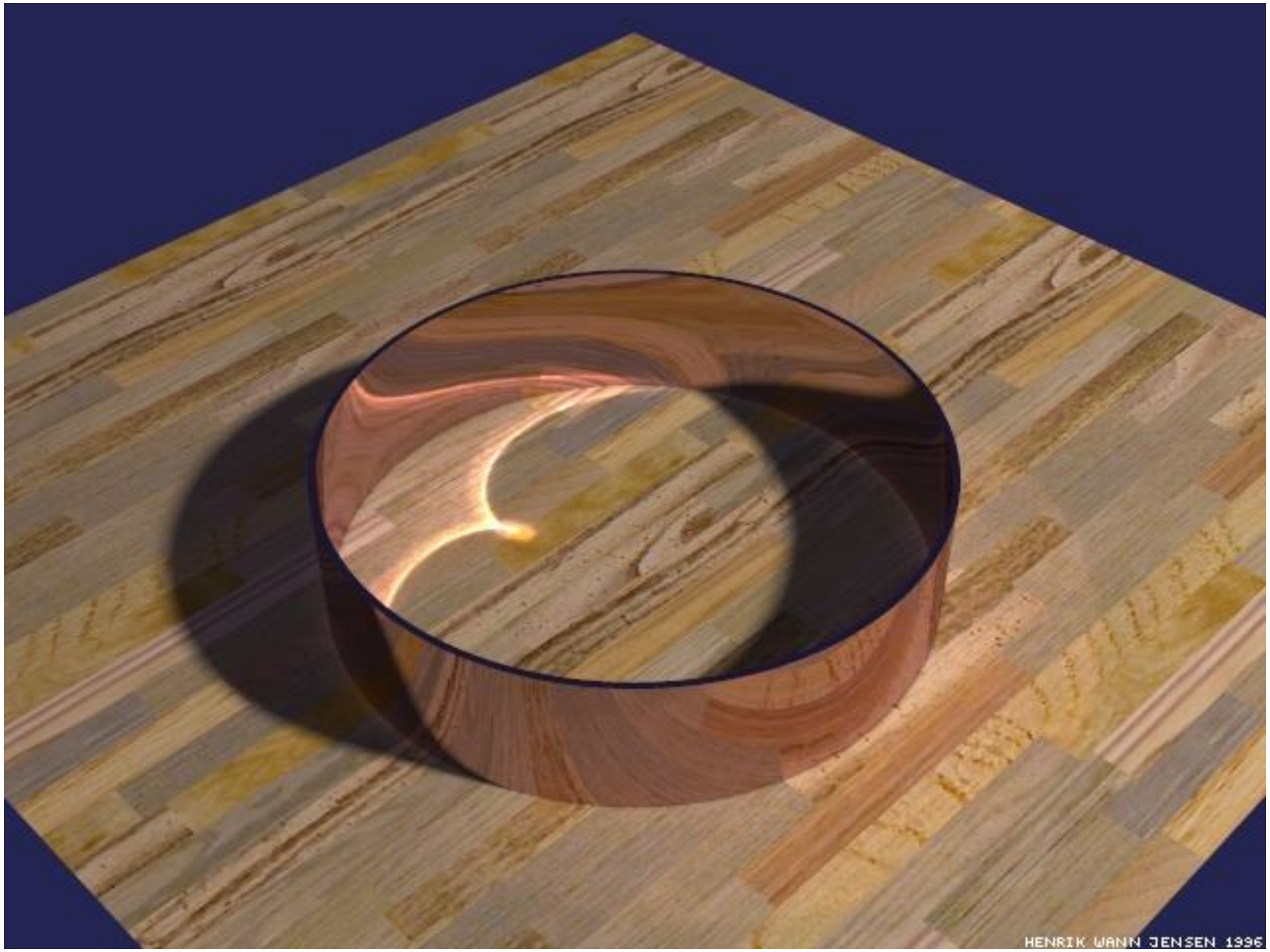
$$
\begin{aligned}
L(x,\vec{\omega}) &= \int_\Omega f_r(x,\vec{\omega}',\vec{\omega}) L'(x,\vec{\omega}') \cos\theta' \, d\omega' \\
&= \int_\Omega f_r(x,\vec{\omega}',\vec{\omega}) \frac{d\Phi'^2(x,\vec{\omega}')}{d\omega' \cos\theta' \, dA} \cos\theta' d\omega' \\
&= \int_\Omega f_r(x,\vec{\omega}',\vec{\omega}) \frac{d\Phi'^2(x,\vec{\omega}')}{dA} \\
&\approx \sum_{p=1}^{n} f_r(x,\vec{\omega}'_p,\vec{\omega}) \frac{\Delta\Phi_p(x,\vec{\omega}'_p)}{\Delta A}
\end{aligned}
$$

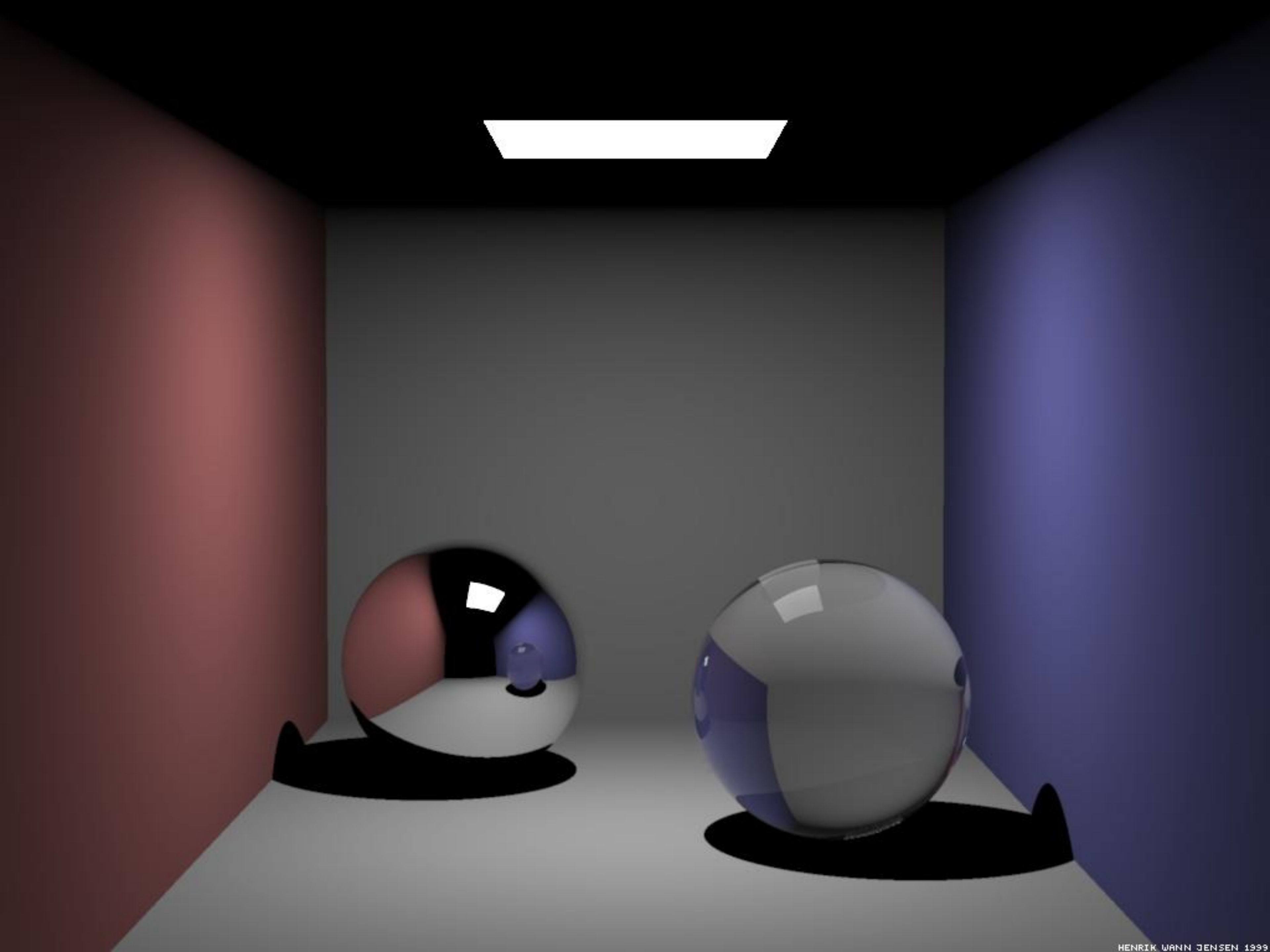
Radiance estimate

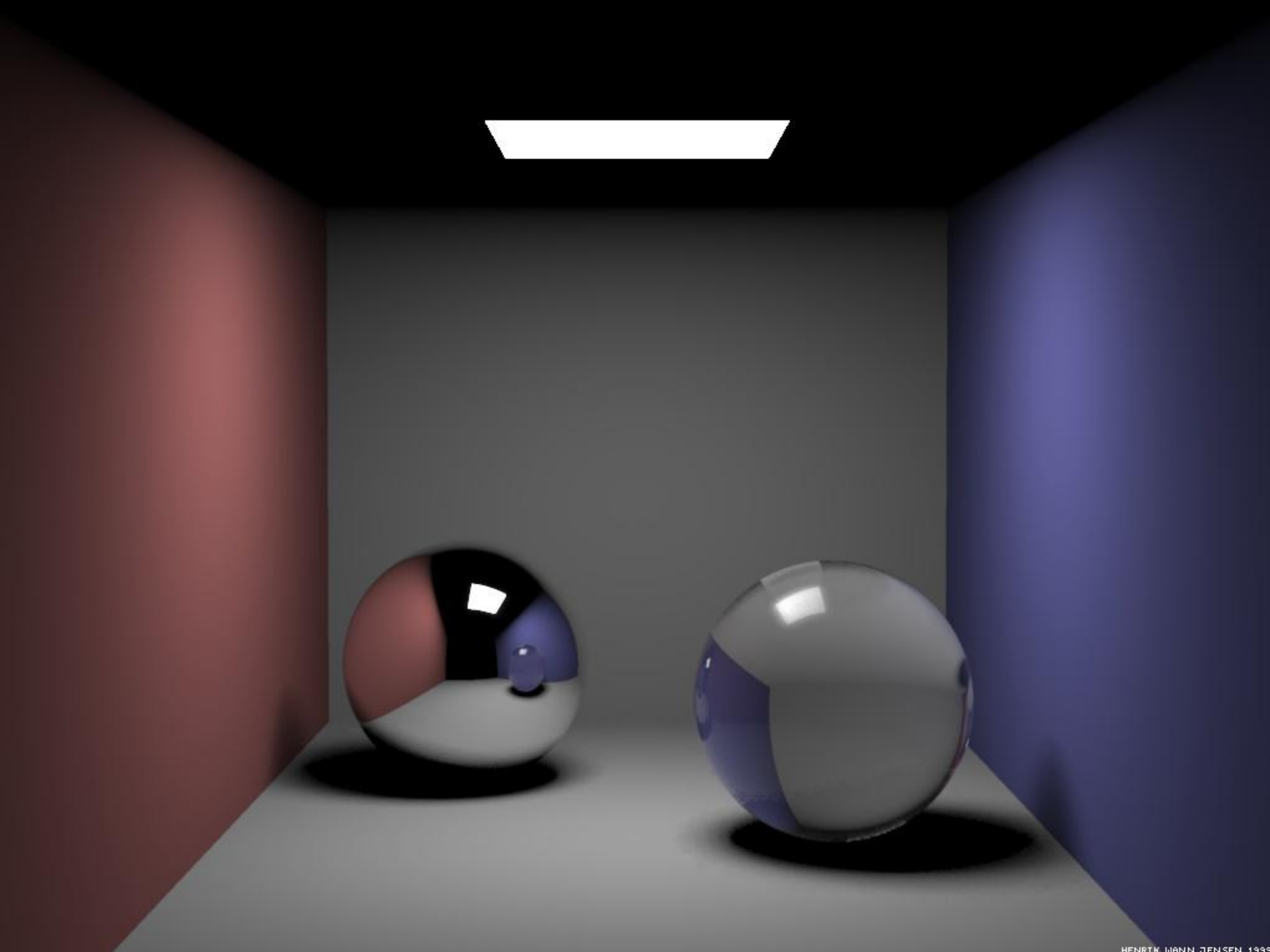


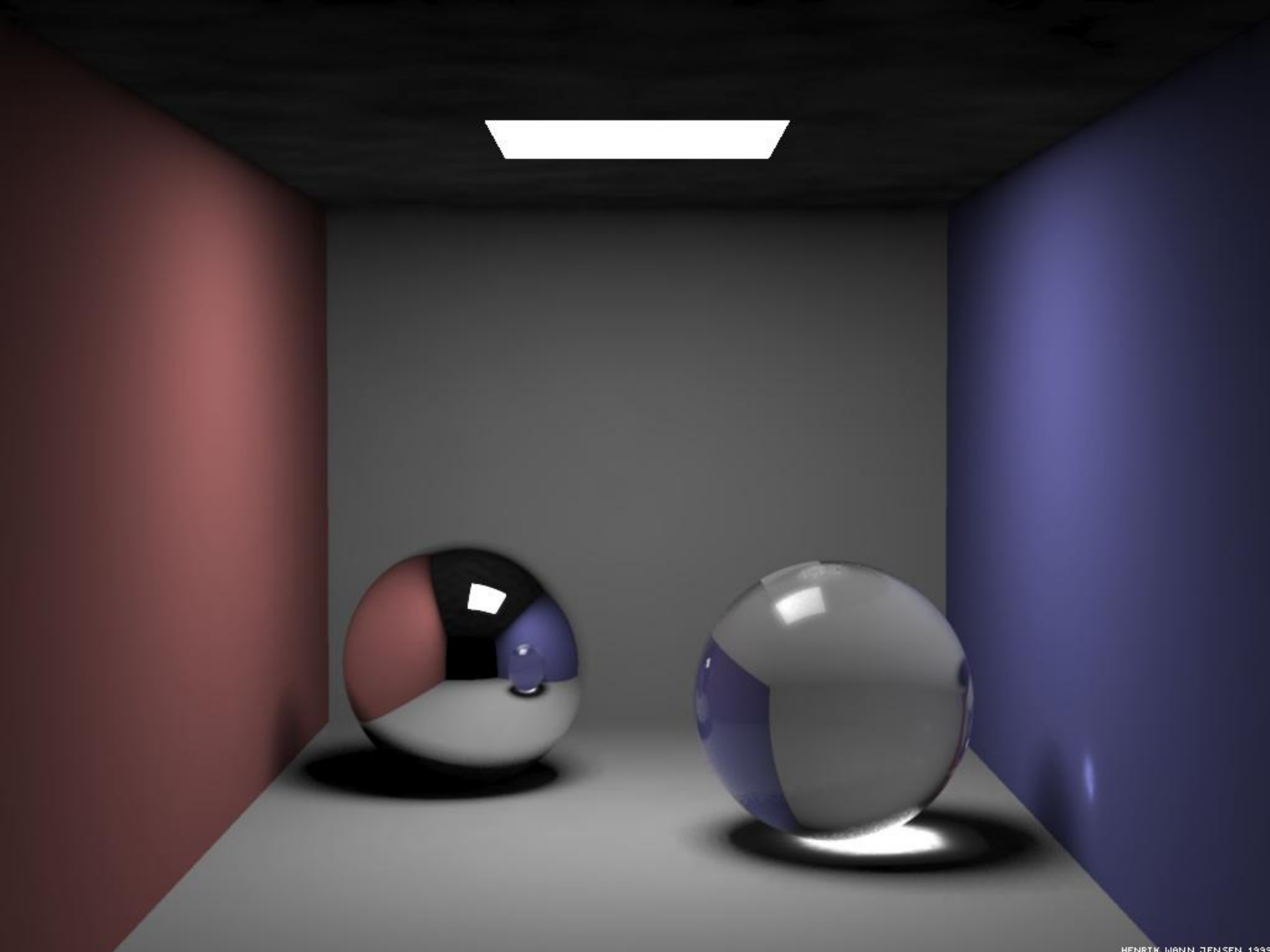


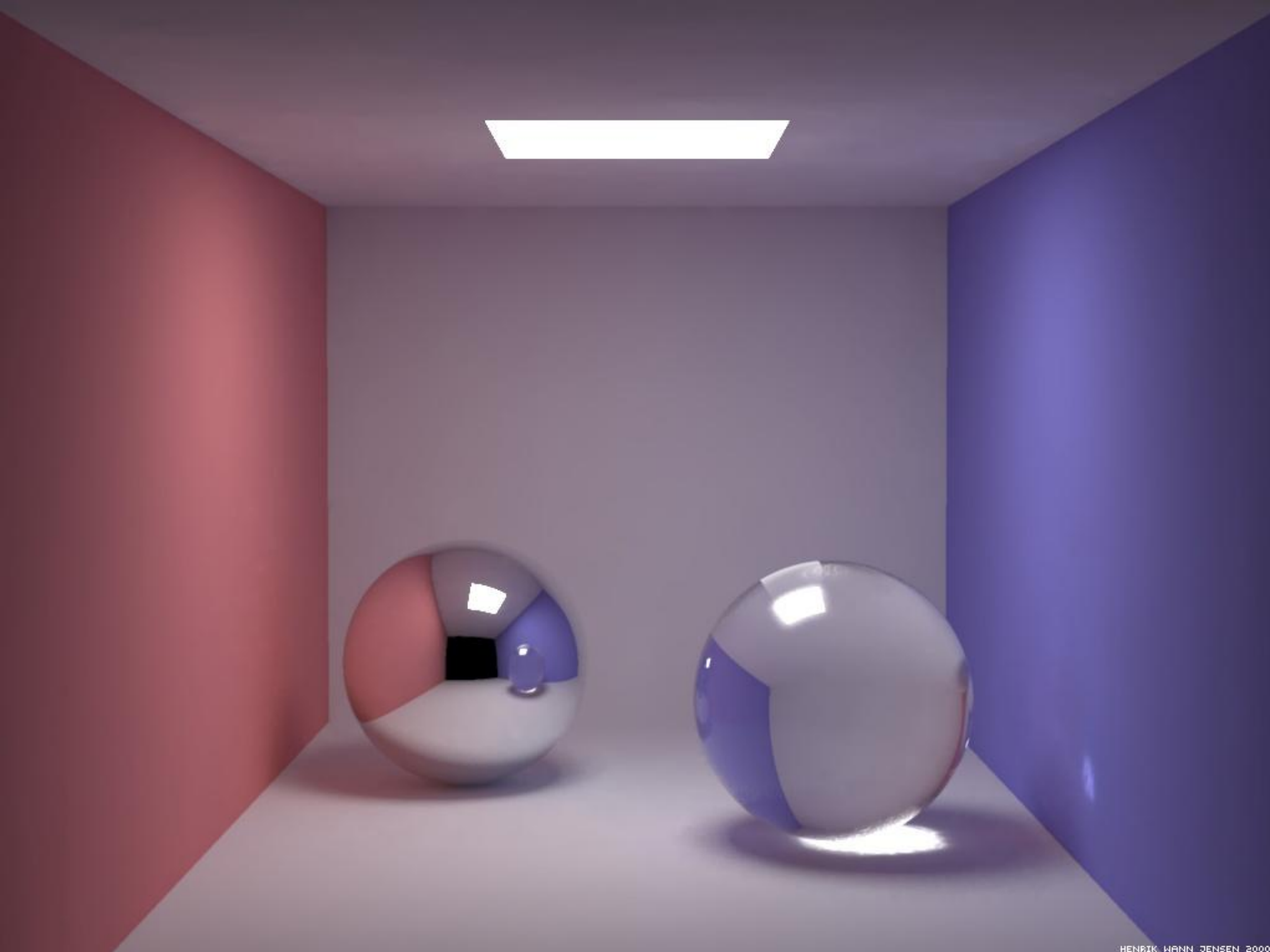
30000 photons / 50 photons in radiance estimate


HENRIK WANN JENSEN 1996


HENRIK WANN JENSEN 1999


HENRIK WANN JENSEN 1999


HENRIK WANN JENSEN 1999


HENRIK WANN JENSEN 2000

Adding water --- more caustics

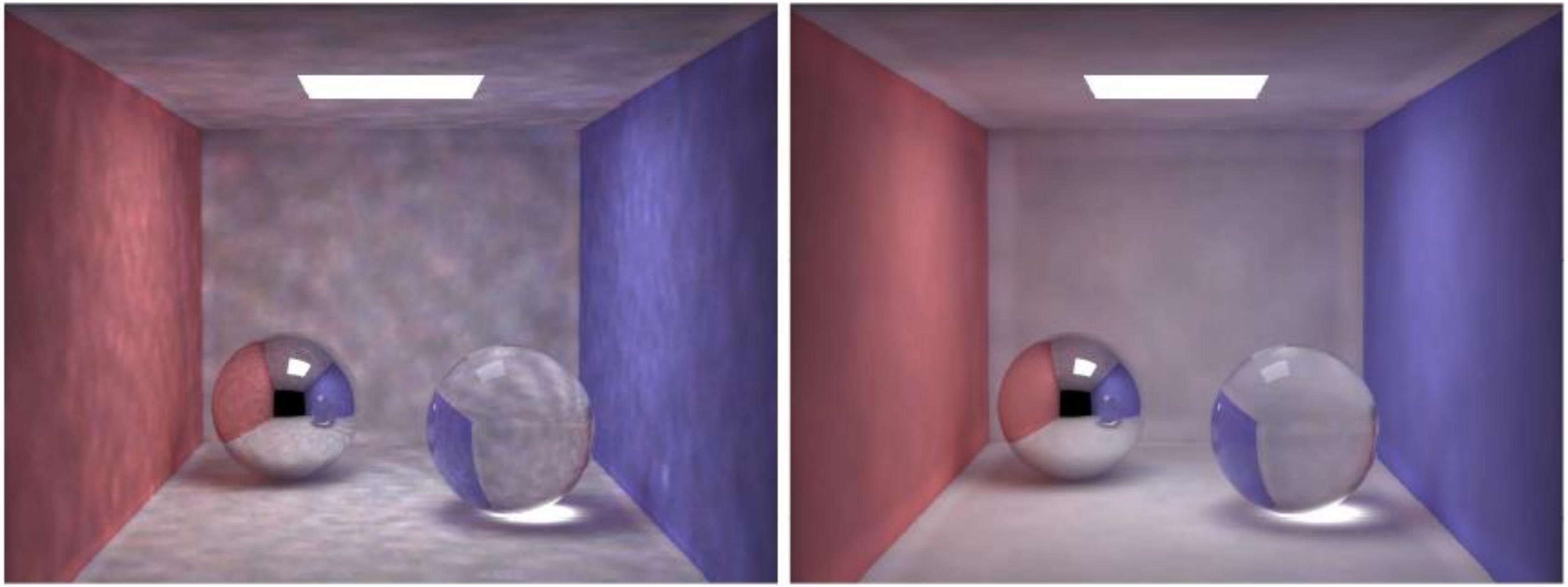
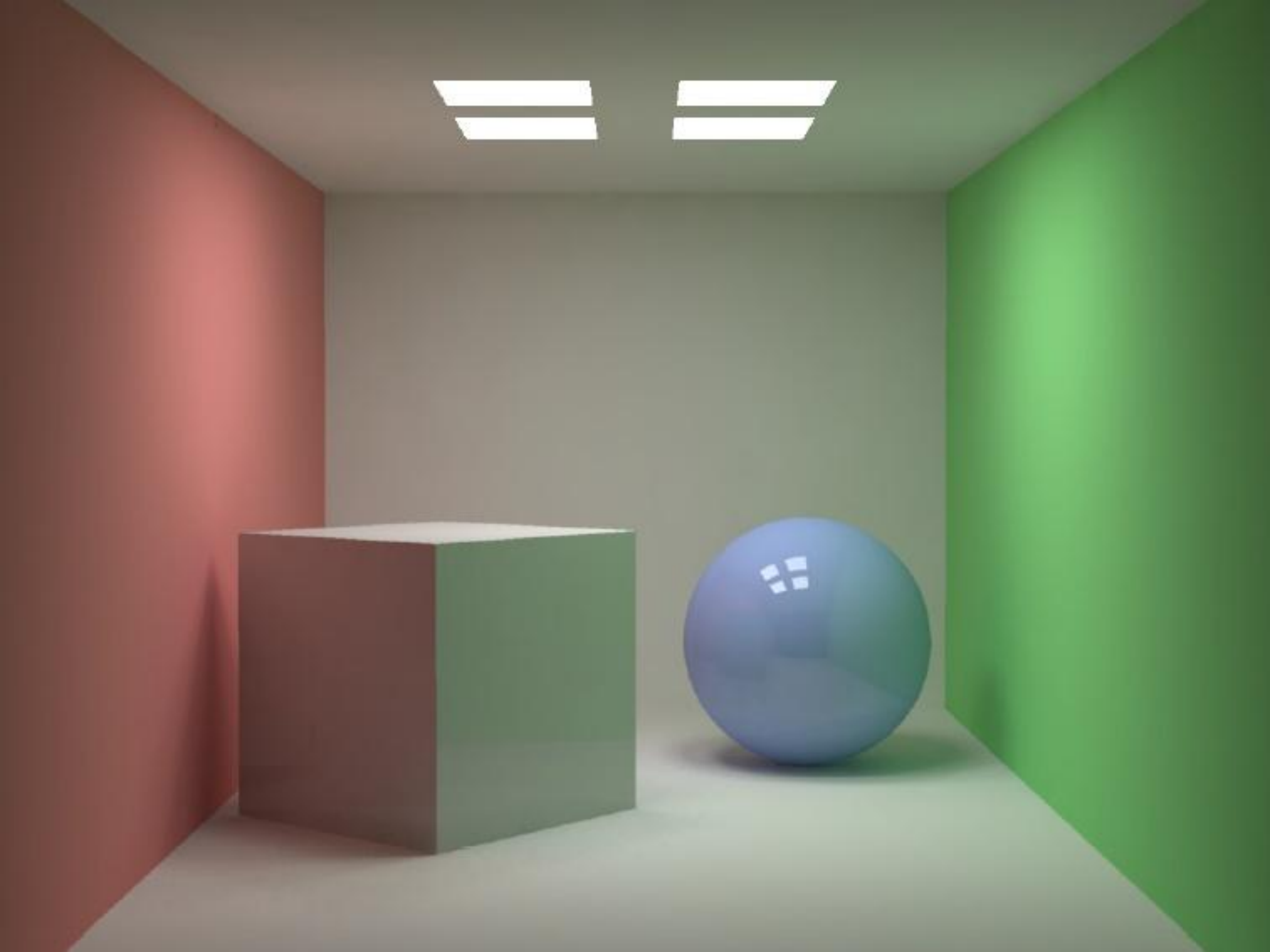


Figure 4.20: Global photon map radiance estimates visualized directly using 100 photons (left) and 500 photons (right) in the radiance estimate.




HENRIK WANN JENSEN 1996


RENDERED WITH DALI - HENRIK WANN JENSEN 2000