Illumination by Randomized Integration

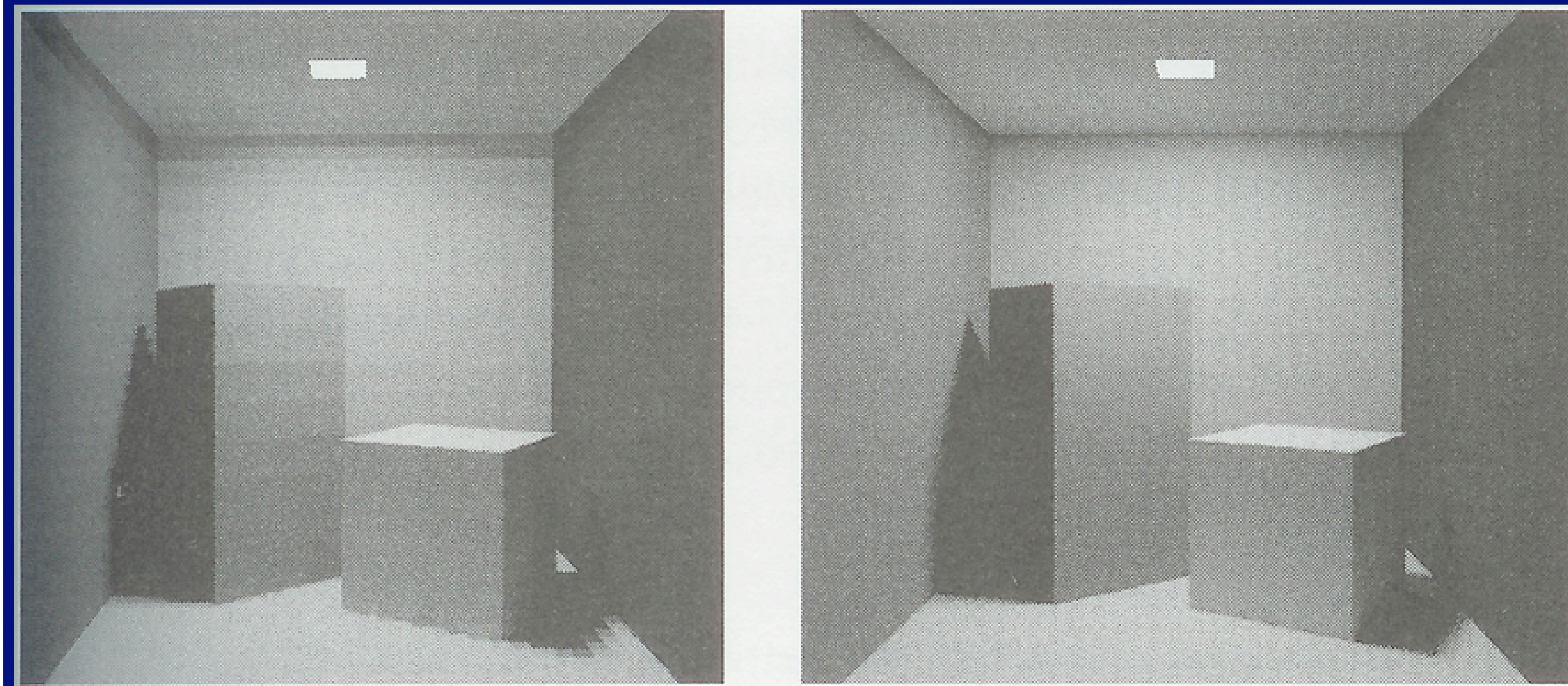
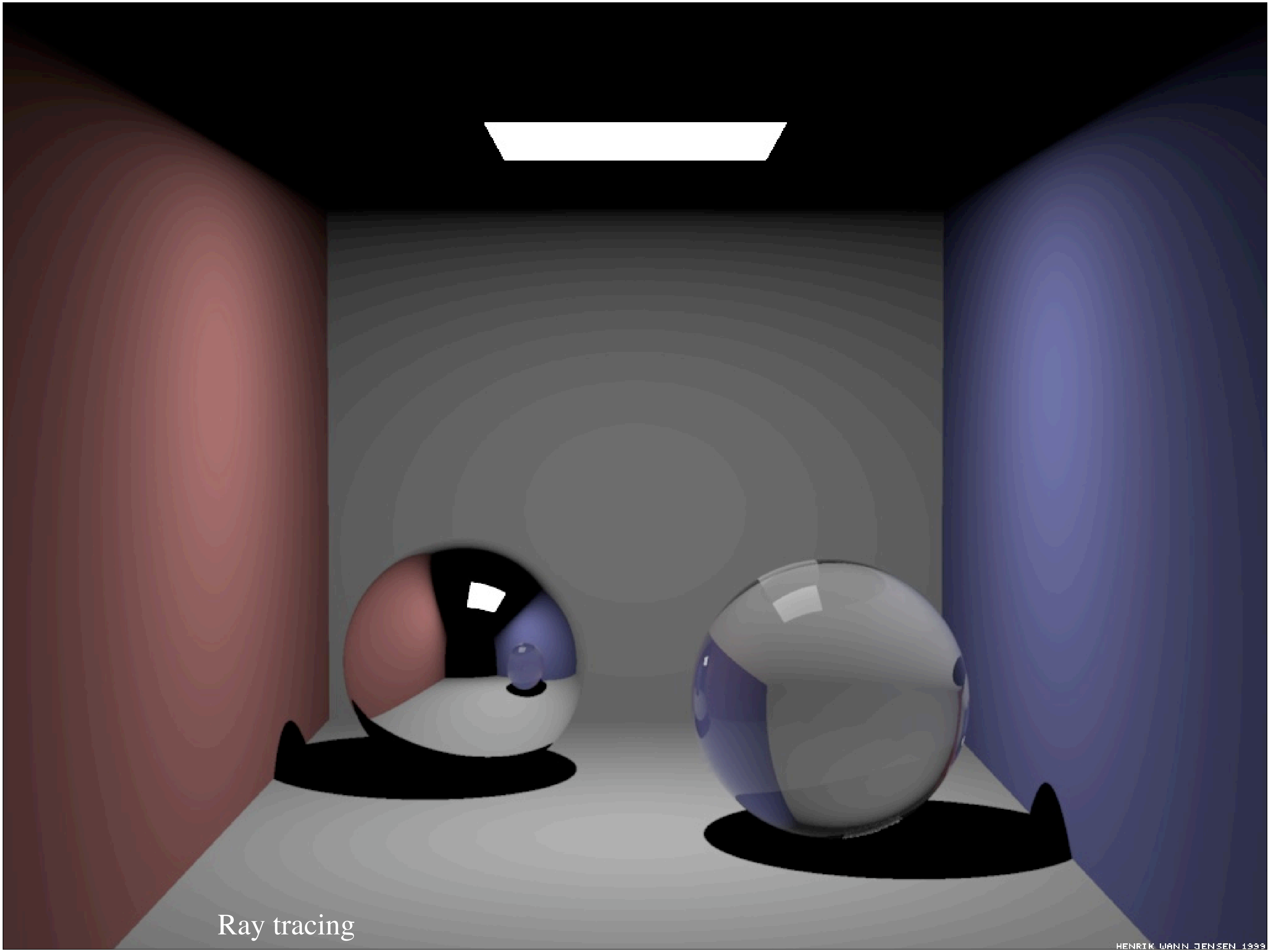
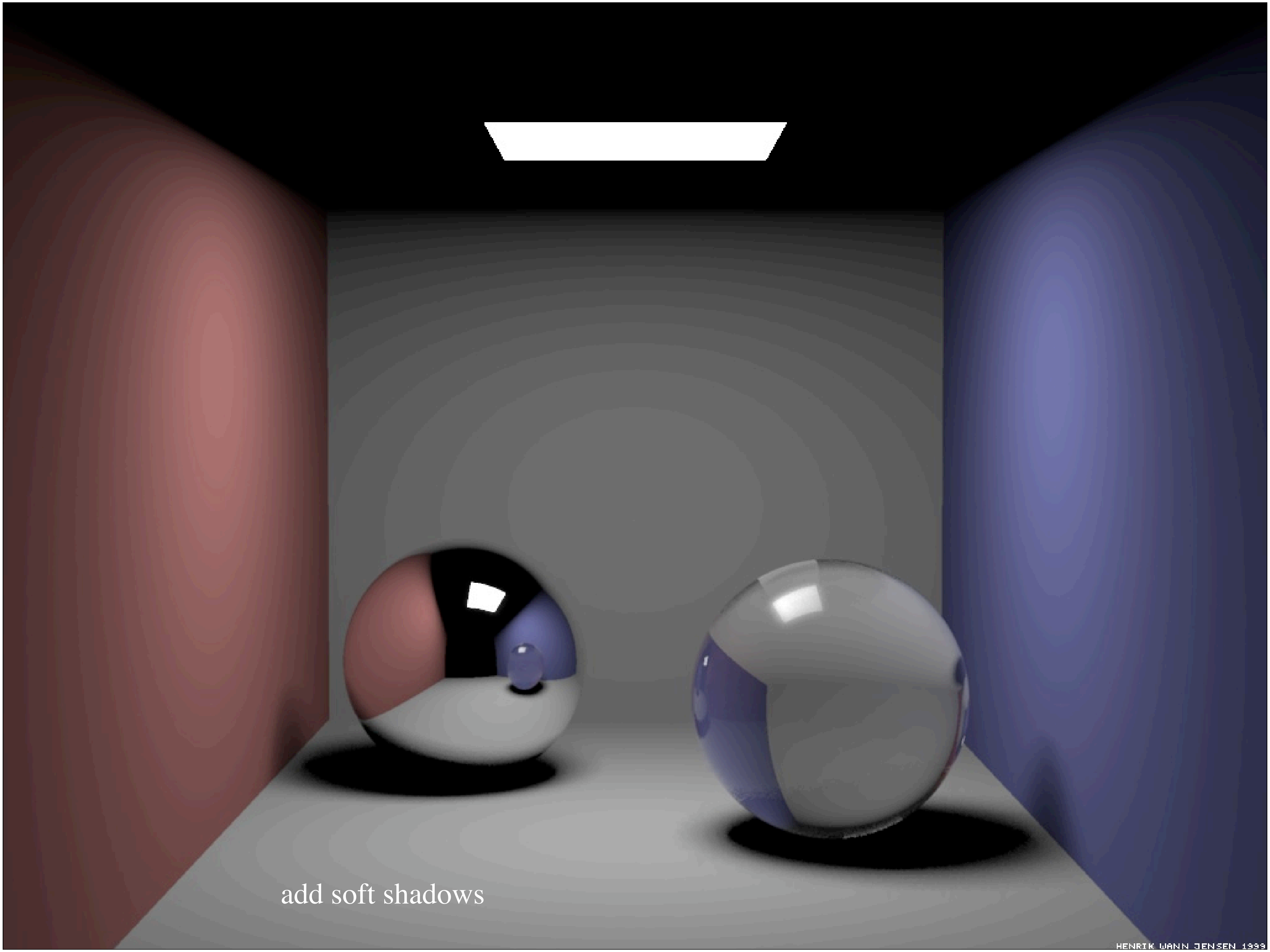


Figure from Dutre, Bekaert, Bala 03; rendered by Suykens-De Laet



Ray tracing


add soft shadows
HENRIK WANN JENSEN 1999

Paths

- Recall we could do LDS*E with eye ray tracing
  - we can do LD*E with finite element radiosity methods
    - actually, can also do LD*S*E (how?)
- But there are many more paths
  - and surfaces might not be only S or D, too
- Model paths explicitly

Path tracing

- Paths start at eye
- At diffuse surface, choose ongoing direction uniformly at random across hemisphere
- Value of path is accumulated values of reflectance along path times Exitance at end

- But
  - where does a path stop?
    - when albedo is zero (many luminaires)

Issues

- Severe variance problems as described
  - or very very slow for nice solutions
- What about surfaces that aren’t diffuse?
  - and illumination at different wavelengths?

Russian roulette

- Increase probability of absorbtion of low weight paths, and reweight the paths
  - notice that ρ helps here a lot -
  - we could change the probability of absorbtion

Path tracing

- Path tracing
  - Path starts at eye
  - At a diffuse surface with albedo ρ, path is
    - continued with probability ρ
    - absorbed with probability 1- ρ
  - Value of path
    - E(x) if it arrives at luminaire
    - 0 otherwise
  - Direction along which path continues is uniform over exit hemisphere

This is a diffuse surface formulation.

Particle tracing

- Particle starts at source
- At a diffuse surface with albedo ρ, path is
  - continued with probability ρ
  - absorbed with probability 1- ρ
- In either case, it deposits power in a texel at that point
  - particle with power ϕ arriving at texel with area A_t
    - deposits power $\frac{\phi}{A_t}$
    -
- Direction along which path continues is uniform over exit hemisphere

This is a diffuse surface formulation.

Global illumination

- Account for all light transfer

$$L(\mathbf{x}, \mathbf{x} \rightarrow \mathbf{y}) = L_e(\mathbf{x}, \mathbf{x} \rightarrow \mathbf{y}) + \int \text{radiance due to irradiance} d\omega$$

- And this gets us

$$L(\mathbf{x}, \mathbf{x} \rightarrow \mathbf{y}) = L_e(\mathbf{x}, \mathbf{x} \rightarrow \mathbf{y}) + \int \rho_{bd}(\mathbf{x} \rightarrow \mathbf{y}, \mathbf{u} \rightarrow \mathbf{x}) L(\mathbf{x}, \mathbf{u} \rightarrow \mathbf{x}) \cos\theta d\omega$$


RENDERED USING DALI - HENRIK WANN JENSEN 2000

Path tracing for general surfaces

- Path tracing
  - Path starts at eye
  - At a surface, path is
    - continued with probability α
    - absorbed with probability 1- α
  - Value of path
    - $L_e(\mathbf{x}_n, \mathbf{x}_n \rightarrow \mathbf{x}_{n-1})$ if it arrives at luminaire
    - 0 otherwise
  - Direction along which path continues
    - a draw from P(ω)
  - Weight path segment by

$$\frac{\rho_{bd}(\mathbf{x}_{n-1} \rightarrow \mathbf{x}_n, \mathbf{x}_n \rightarrow \mathbf{x}_{n+1}) \cos\theta}{P(\omega)\alpha}$$

    - and accumulate these weights

This will do anything,
but with very serious variance problems

Variance problems

- Paths may not find the light often
  - this could be fixed by clever choice of P to heavily emphasize directions toward the source
- Caustics will be poorly rendered, because the path to the source is obscure

Bidirectional path tracing

- Start paths at both eye and light and join them
- Notice:
  - a pair of eye-light paths generates many possible transfer paths
  - we can use each of these, if we compute weights correctly to get integral estimate right

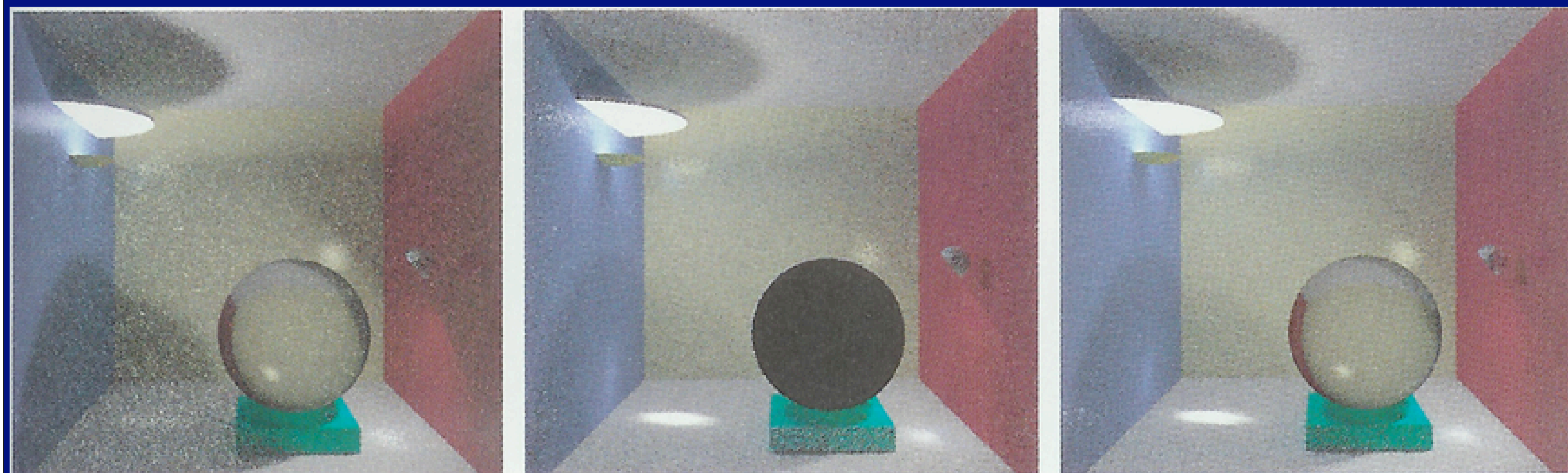


Figure from Dutre, Bekaert, Bala 03; rendered by Suykens-De Laet

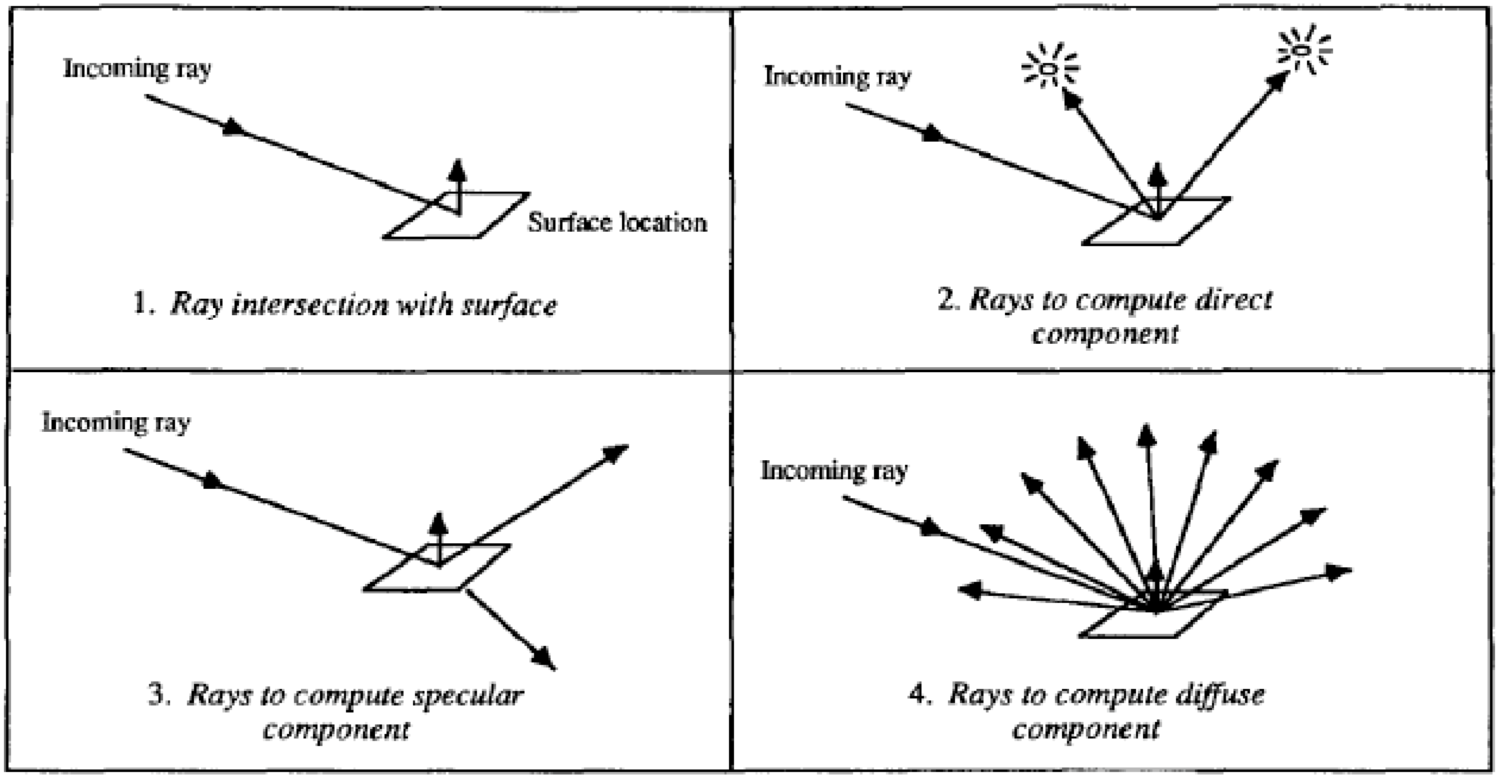


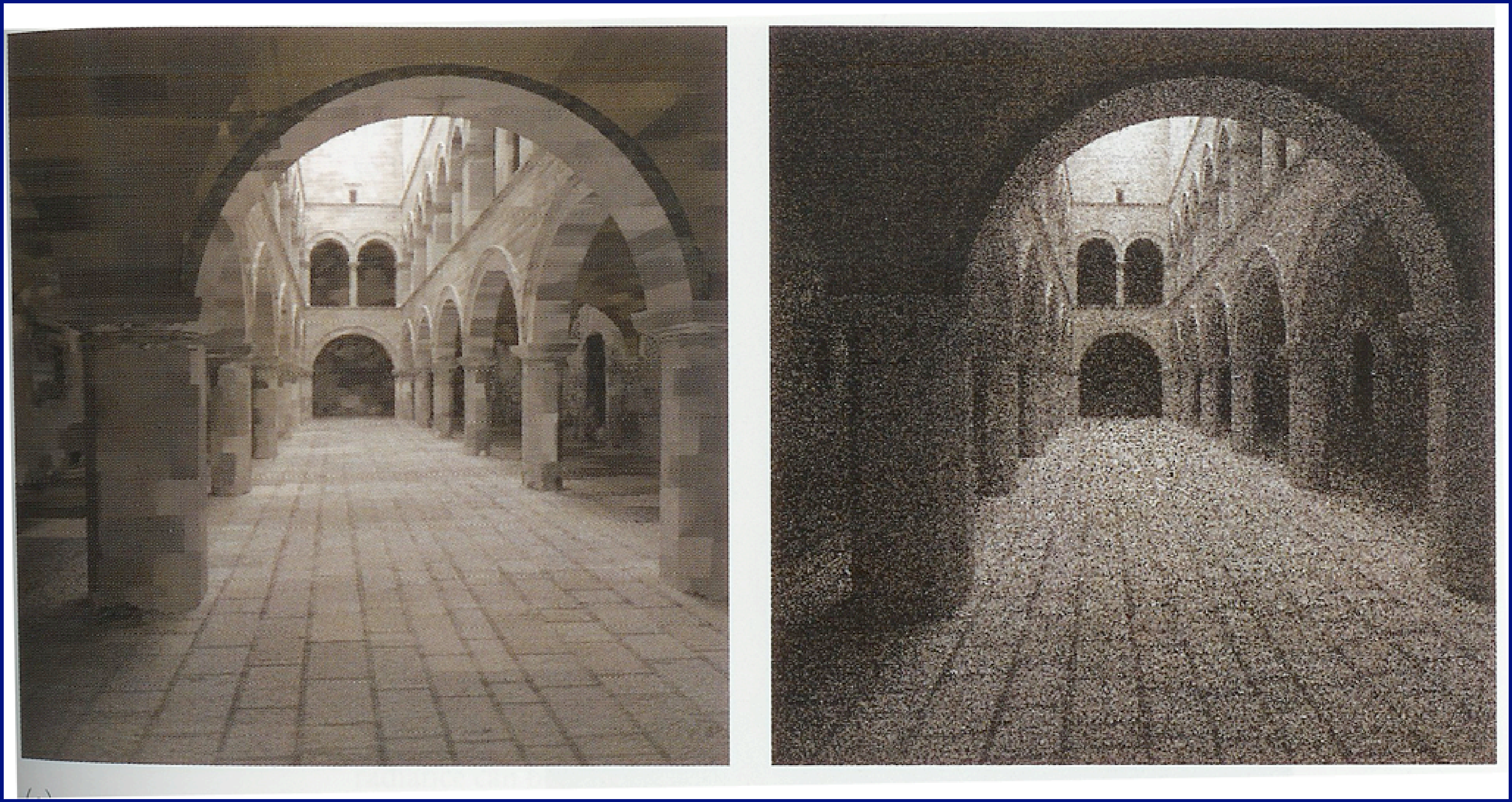
Figure 1: The four steps of ray tracing.

Figure from Ward et al, “A Ray-tracing solution for diffuse interreflection”, 1988

Irradiance caching

- The indirect term varies slowly over space
  - cache and interpolate
- Cache by
  - storing irradiance samples in octree with normal
- Interpolate by
  - obtaining all samples with error smaller than α
    - error is:
      - (distance term)+(normal term)
    - not enough samples?
      - generate new ones and cache them
  - forming weighted sum using extrapolated illumination values

Powerful standard method in
almost every modern rendering system



Irradiance cache vs path tracing, from Pharr + Humphreys

Cache sample locations from Pharr + Humphreys

Irradiance caching: samples

- Obtaining samples:
  - evaluate irradiance at sample point by:
    - direct term:
      - sample each source directly, as before
    - indirect term:
      - sample non-source directions with probability $P(\omega)$
      - form estimate

$$\frac{1}{N}\sum_j \frac{L_i(\mathbf{x}, \omega_j)\cos\theta_j}{P(\omega_j)}$$

      - Notice that the incoming radiance might be computed from the cache, if there are samples

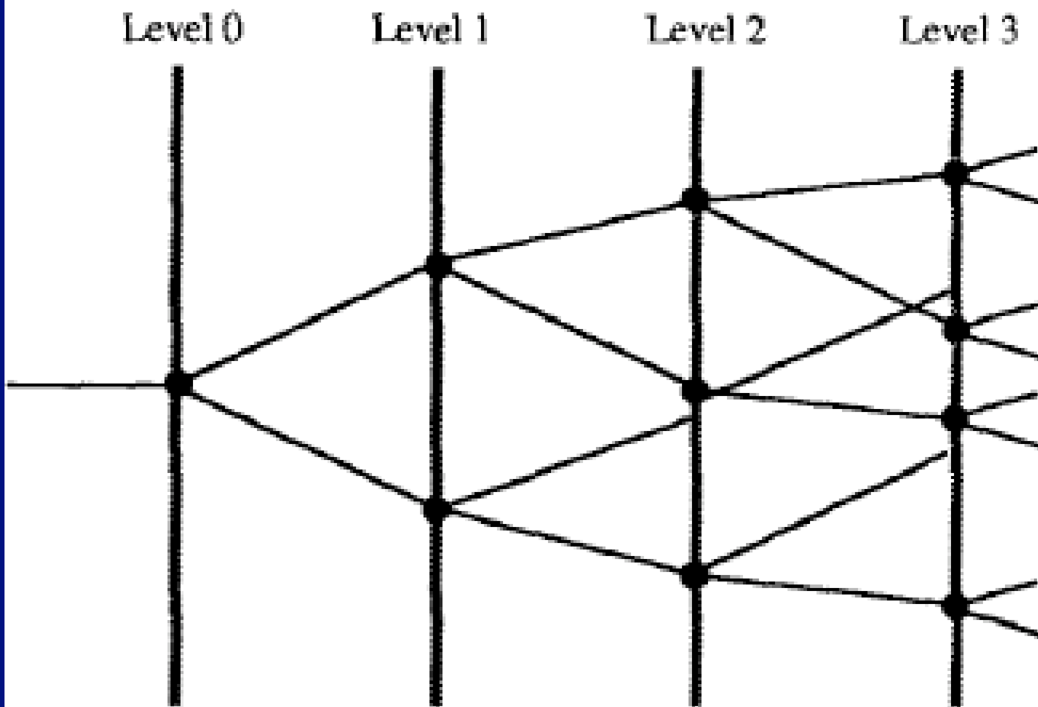


Figure 7: The lines represent rays, and the points represent primary evaluations. The rays that reuse computed values do not propagate.

Note:
Russian roulette prevents the tree getting out of hand
Fairly quickly, the cache fills up

Figure from Ward et al, "A Ray-tracing solution for diffuse interreflection", 1988

Irradiance caching: reconstruction

- Query octree for possible samples
  - Do not want to use:
    - samples that are too far away
      - this is a function of how samples were obtained
    - samples with a bad normal
    - samples that lie closer to the eye than current point
- Reconstruct by
  - weighted sum of samples
  - interpolation process can use:
    - distances
    - gradients

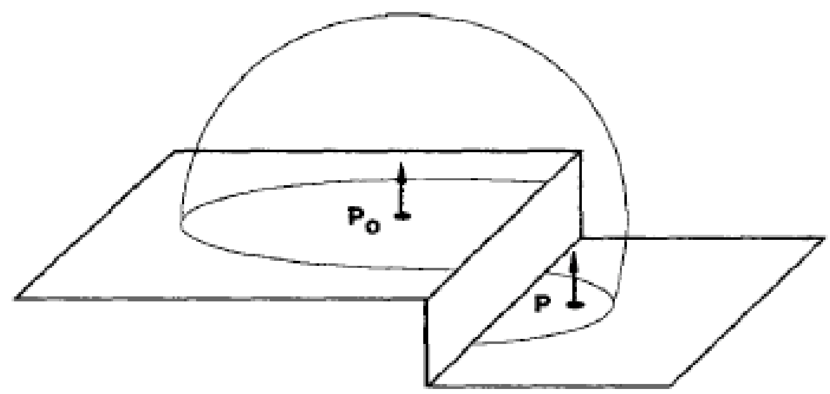
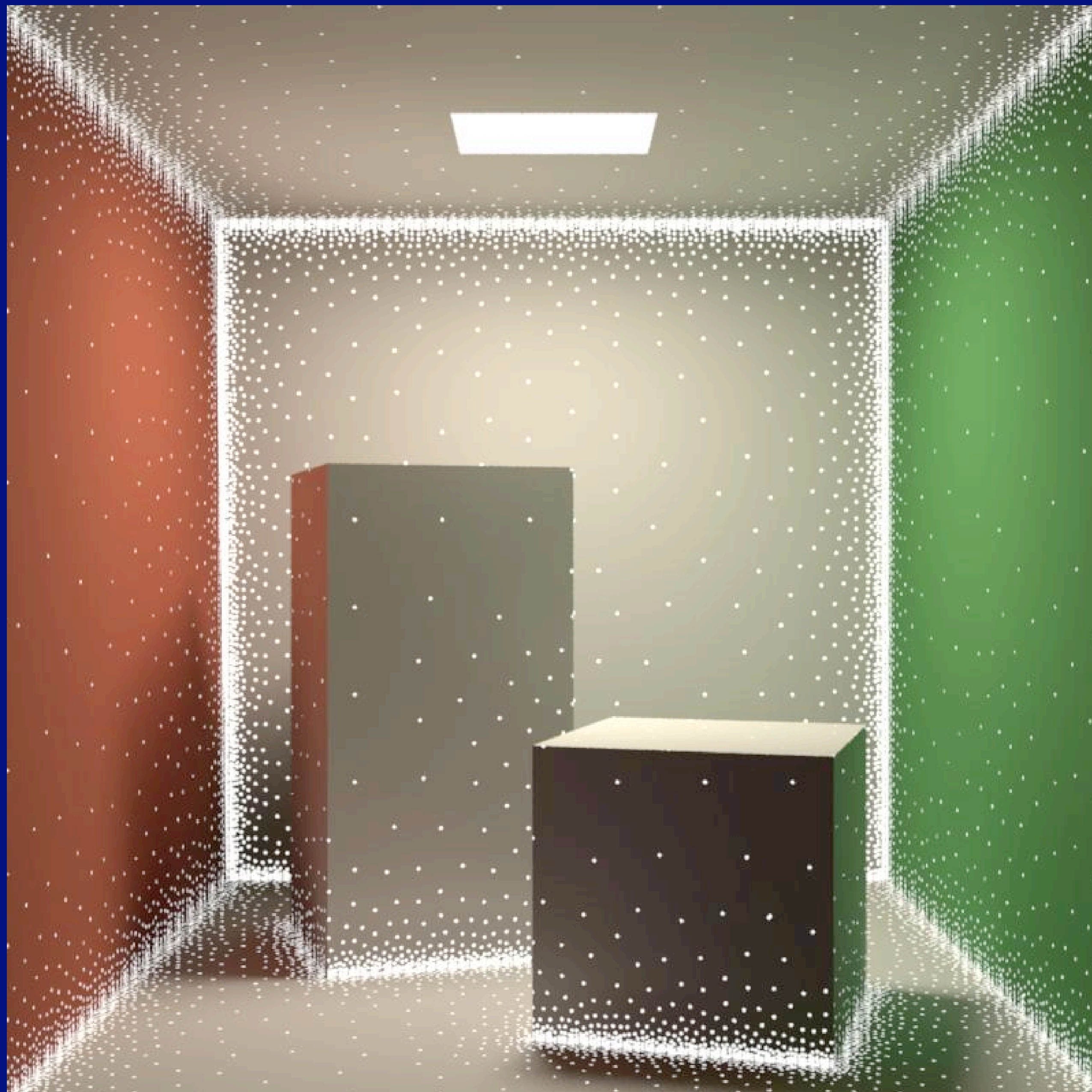


Figure 4: $\vec{P}_0$ sees few close-by surfaces, so its estimated error at $\vec{P}$ is small. But $\vec{P}$ is shadowed by the surface under $\vec{P}_0$, and the true illuminance is different.

Figure from Ward et al, "A Ray-tracing solution for diffuse interreflection", 1988



Typical sample locations for
irradiance cache

Photon maps

- Drop the requirement of an unbiased estimate of illumination
  - accept some bias for better variance properties
- Propagate photons from source, cache when they arrive at surfaces
- Interpolate illumination value by averaging over k-nearest neighbours
- Caustic variance
  - use two classes of photon: sample specular, refractive directions separately
- How many photons?
  - keep trying till it looks good

Photon propagation

- Photons carry Power
  - scale photons from source by number emitted
- reflected
  - diffuse
    - store in map when it arrives, propagate
      - prob proportional to cos theta
      - power scaled by albedo
        - or use russian roulette
  - specular
    - do not store in map, propagate
      - along specular direction
      - power scaled by reflectivity
        - or use russian roulette
  - arbitrary BRDF
    - inportance sample outgoing direction

Photon propagation

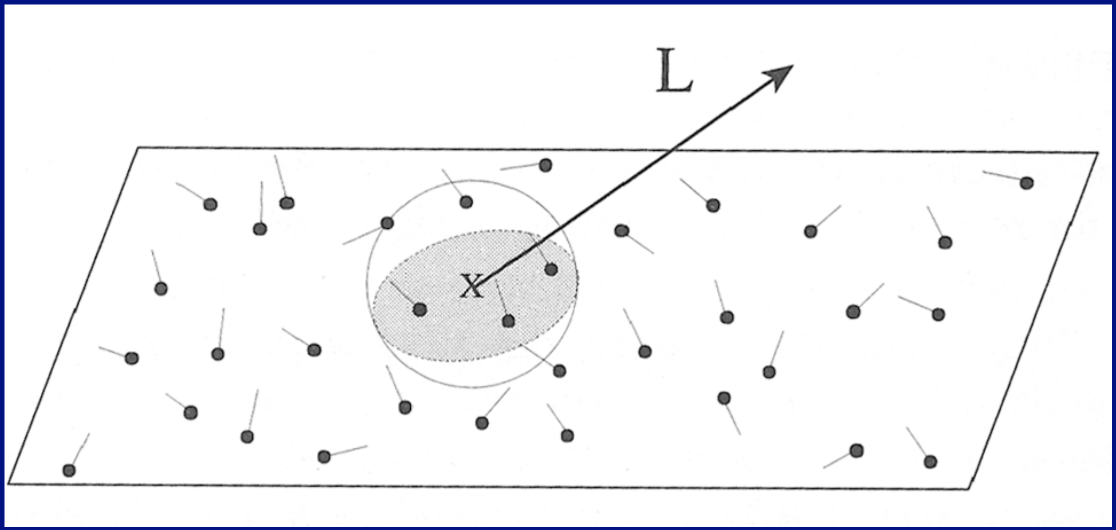
- When a photon arrives at complex surface, multiple photons could be generated
  - eg specular + diffuse
    - russian roulette to decide whether
      - specular
        - reflected/absorbed
      - diffuse
        - reflected/absorbed
- Photons are stored at diffuse (non-specular!) surfaces only
- Stored as:
  - Power, Location, Normal

Photon storage and querying

- Store in k-d tree
  - to look up r closest photons
  - tree represents free space close to surfaces

Evaluating Radiance

- Reflected radiance is: $$L_r(\mathbf{x},\omega) = \int_\Omega \rho_{bd}(\omega,\omega_i) L_i(\mathbf{x},\omega_i)\cos\theta d\omega$$
- Each photon carries known power, in known direction
  - assume the relevant photons all arrive at x
  - each contributes radiance (power/dA)
  - assume surface is flat around x, build a circle
    - photons in this circle contribute
    - area is known



$$L_r(\mathbf{x},\omega)\frac{1}{\pi r^2}\sum \rho_{bd}(\omega,\omega_j)P_j(\mathbf{x},\omega_j)$$

Figure from Jensen’s book

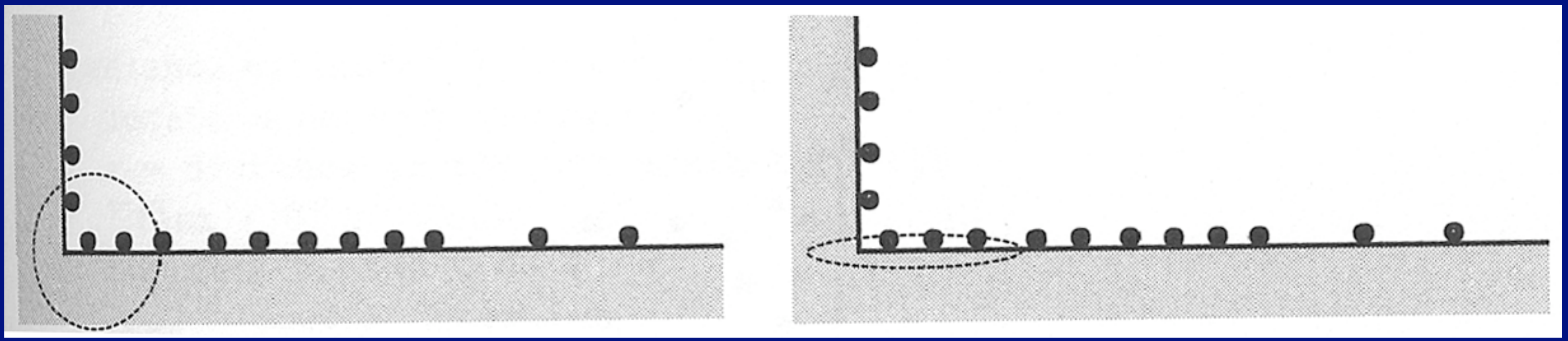
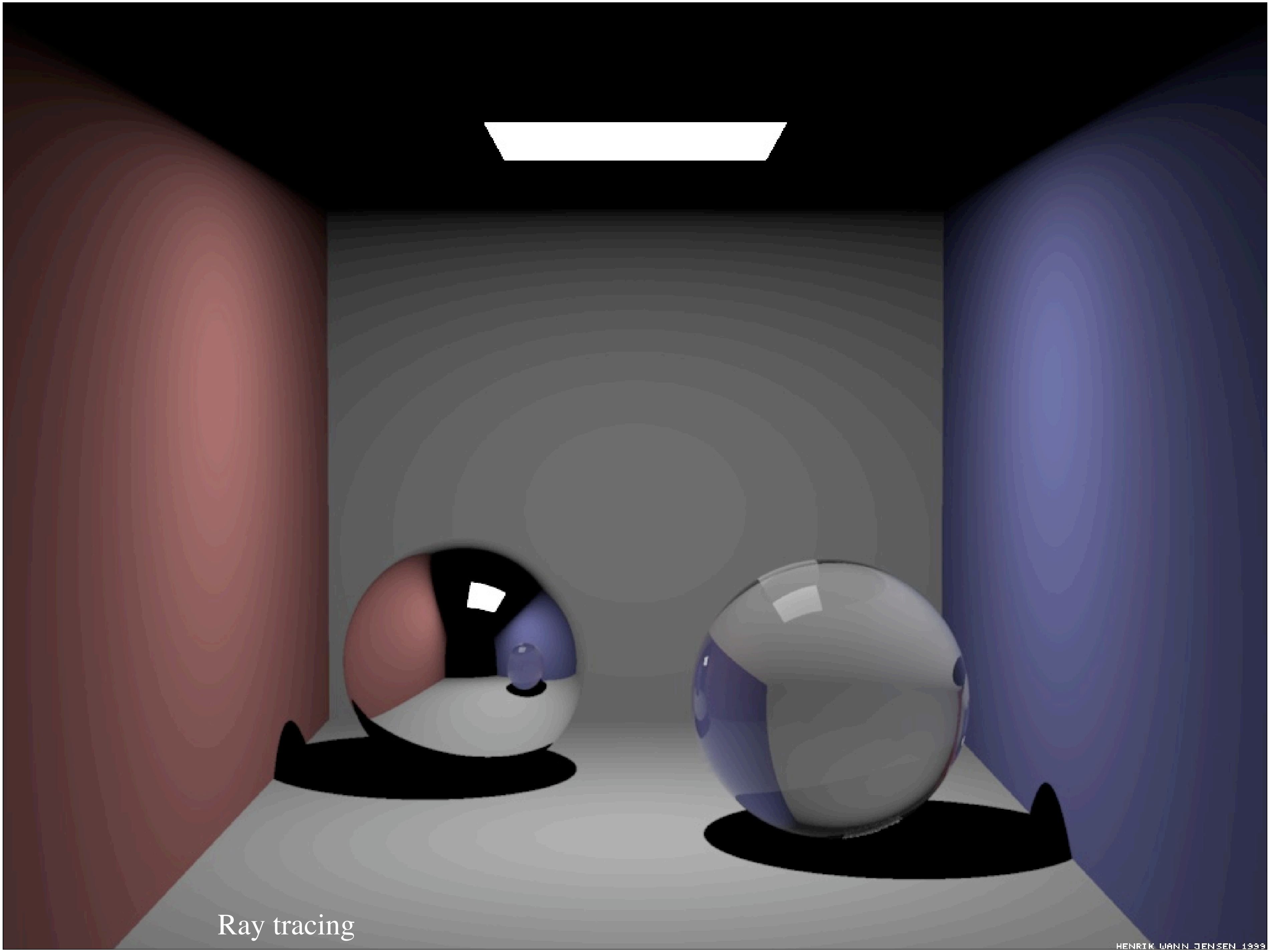


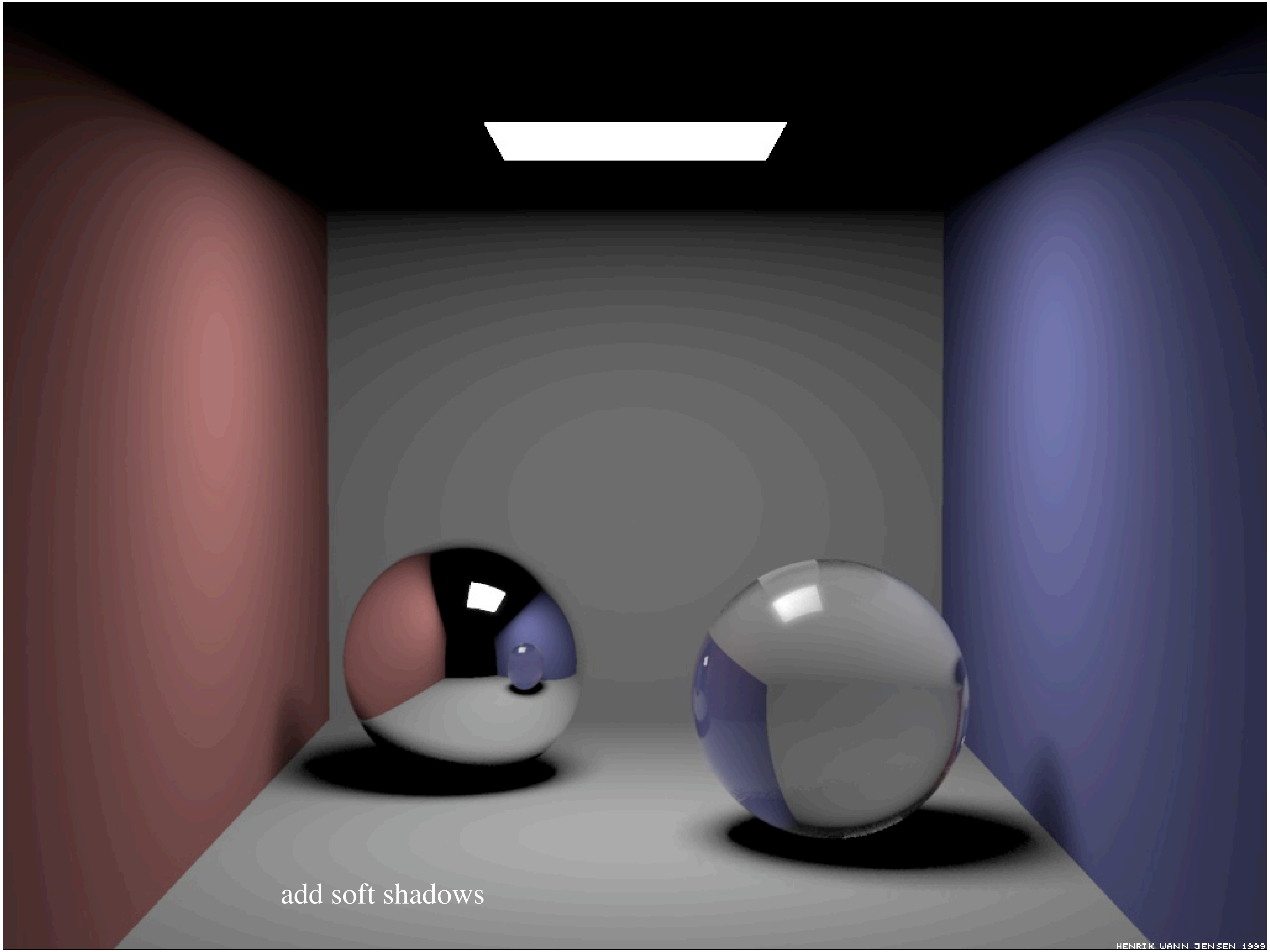
Figure from Jensen's book

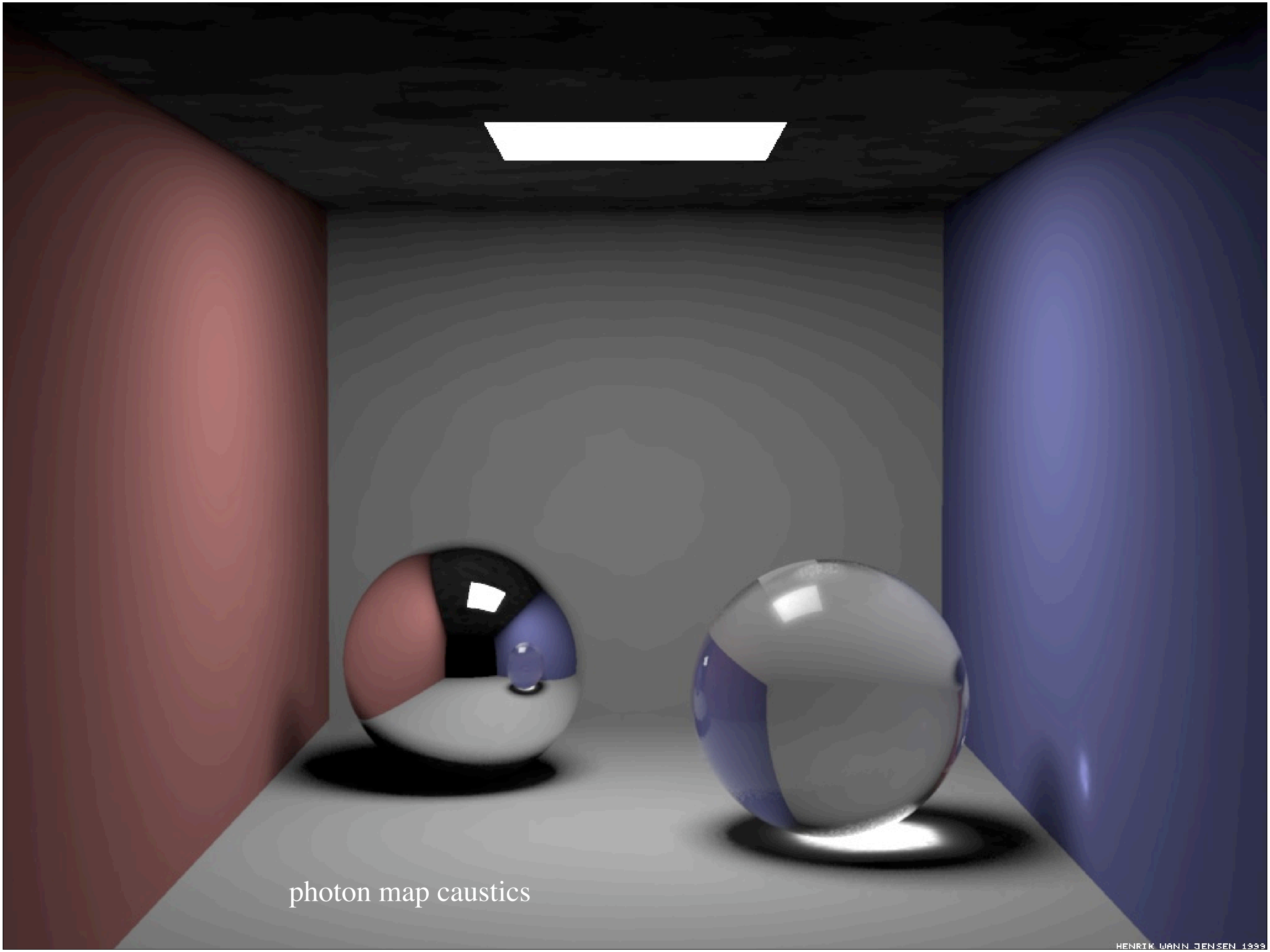
A two pass renderer

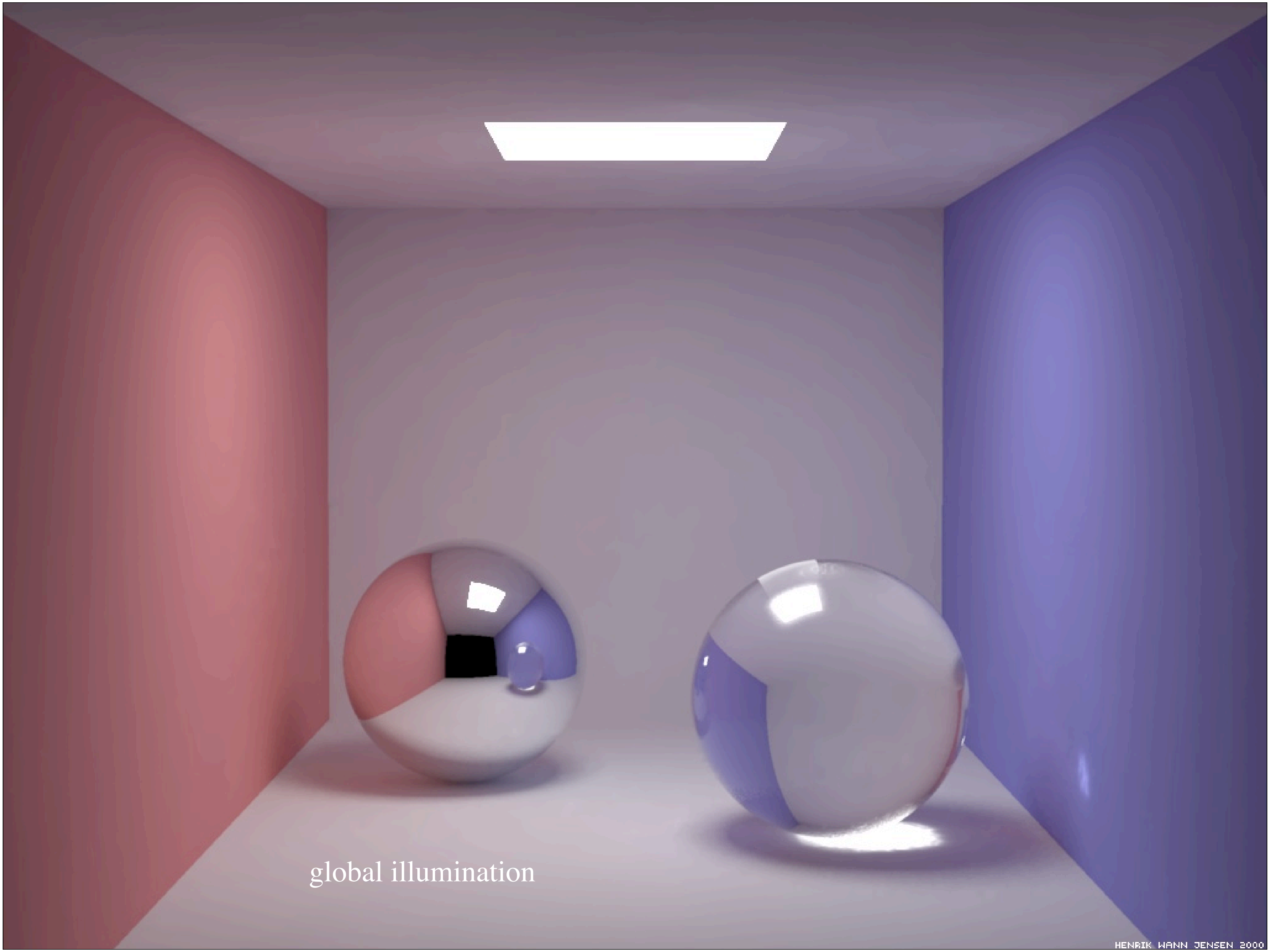
- Propagate photons
  - two classes
    - caustic photons toward specular/glossy, refractive objects
      - large numbers
      - caustic map
    - global illumination photons toward diffuse objects
      - small numbers
- Gather
  - render using
    - direct term by area source sampling
    - specular term by ray-tracing
    - caustic term by direct query to photon map
    - global illumination term by gathering the photon map

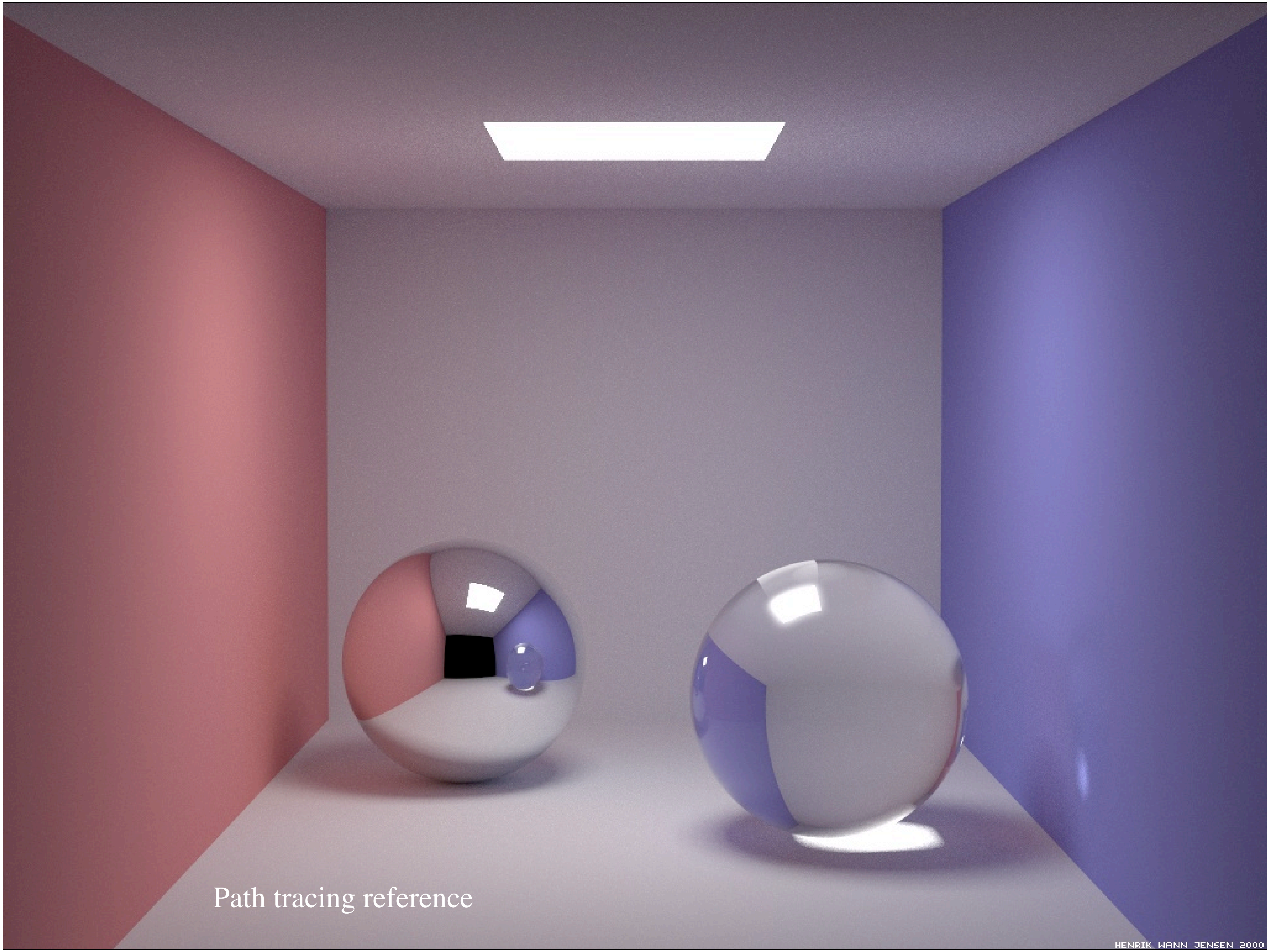

RENDERED WITH DALI - HENRIK WANN JENSEN 2000


Ray tracing
HENRIK WANN JENSEN 1999


add soft shadows
HENRIK WANN JENSEN 1999


photon map caustics
HENRIK WANN JENSEN 1999


global illumination
HENRIK WANN JENSEN 2000


Path tracing reference
HENRIK WANN JENSEN 2000