EDAN30 Photorealistic Computer Graphics

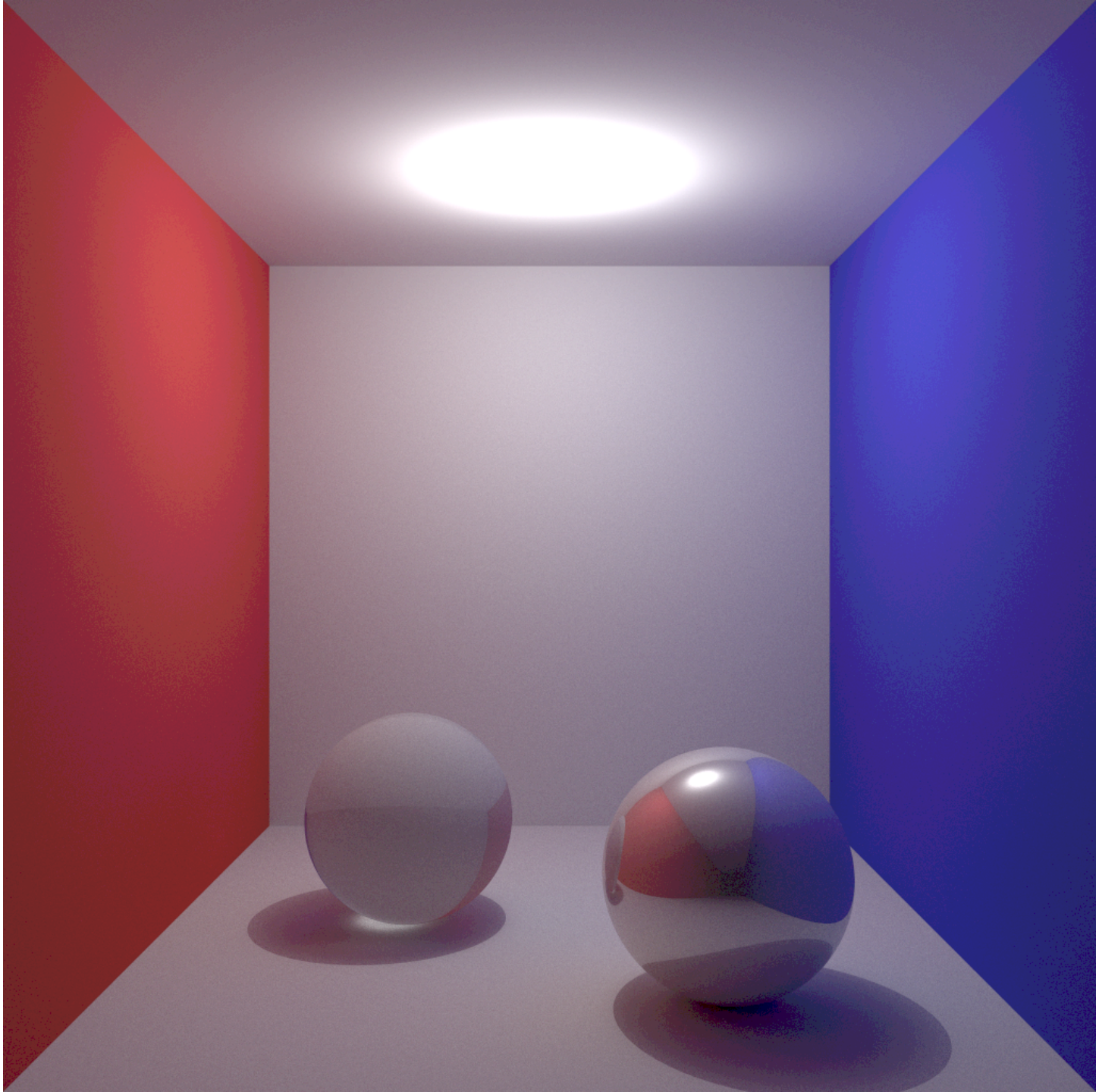
Seminar 4

Progressive Photon Mapping

Rasmus Barringer, PhD student (rasmus@cs.lth.se)

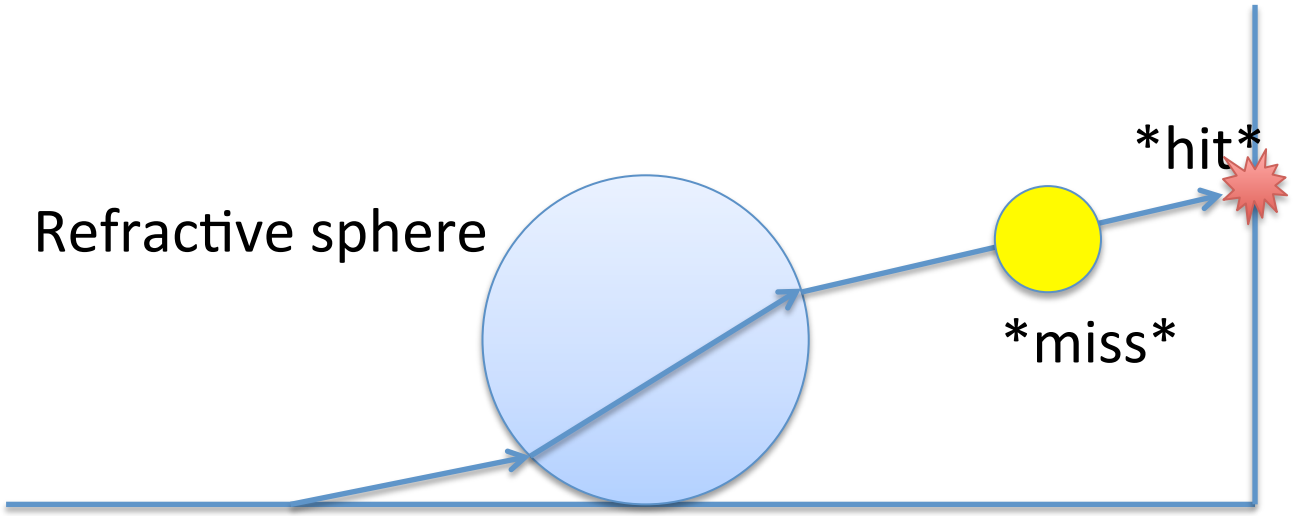
Motivation

- Capture all detail in caustics.



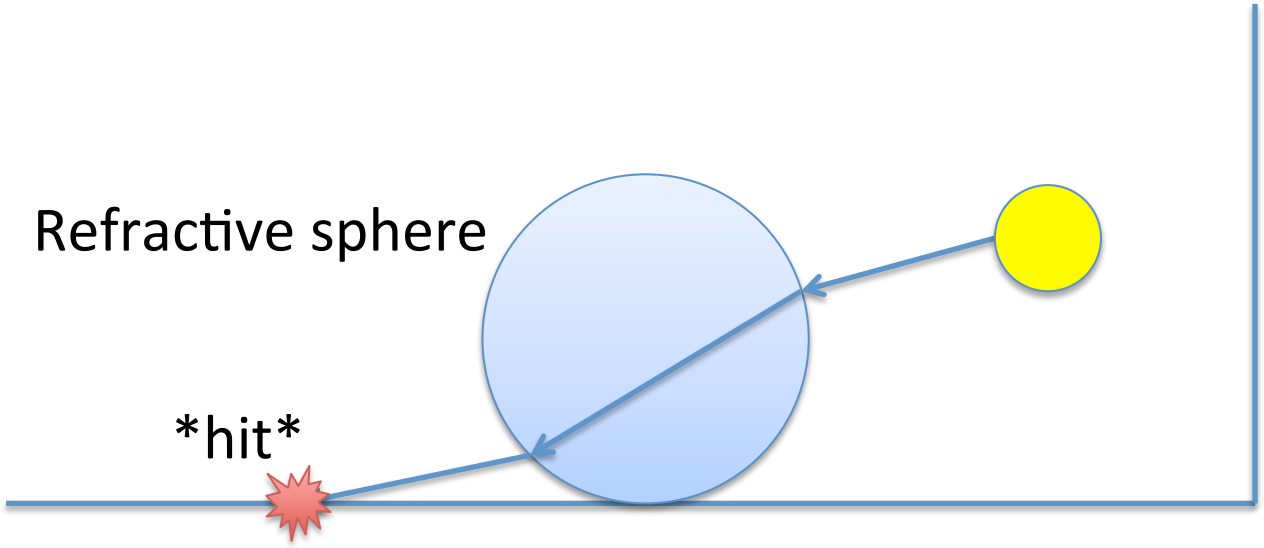
Incorrect caustics

- Cannot find direct light path.
- Only, indirect illumination contributes.



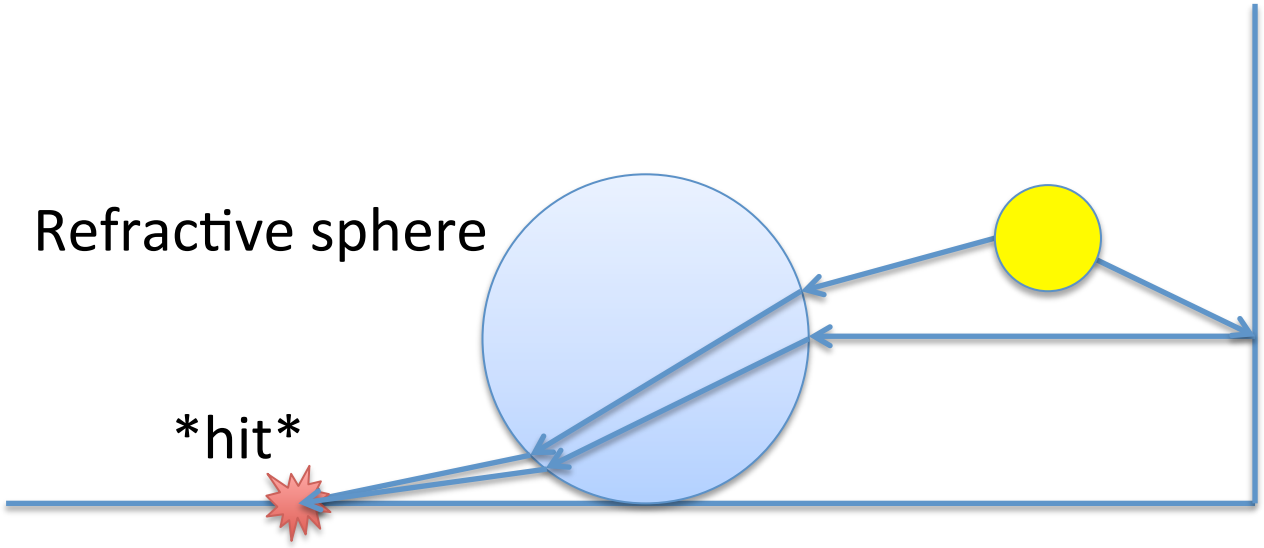
Progressive Photon Mapping

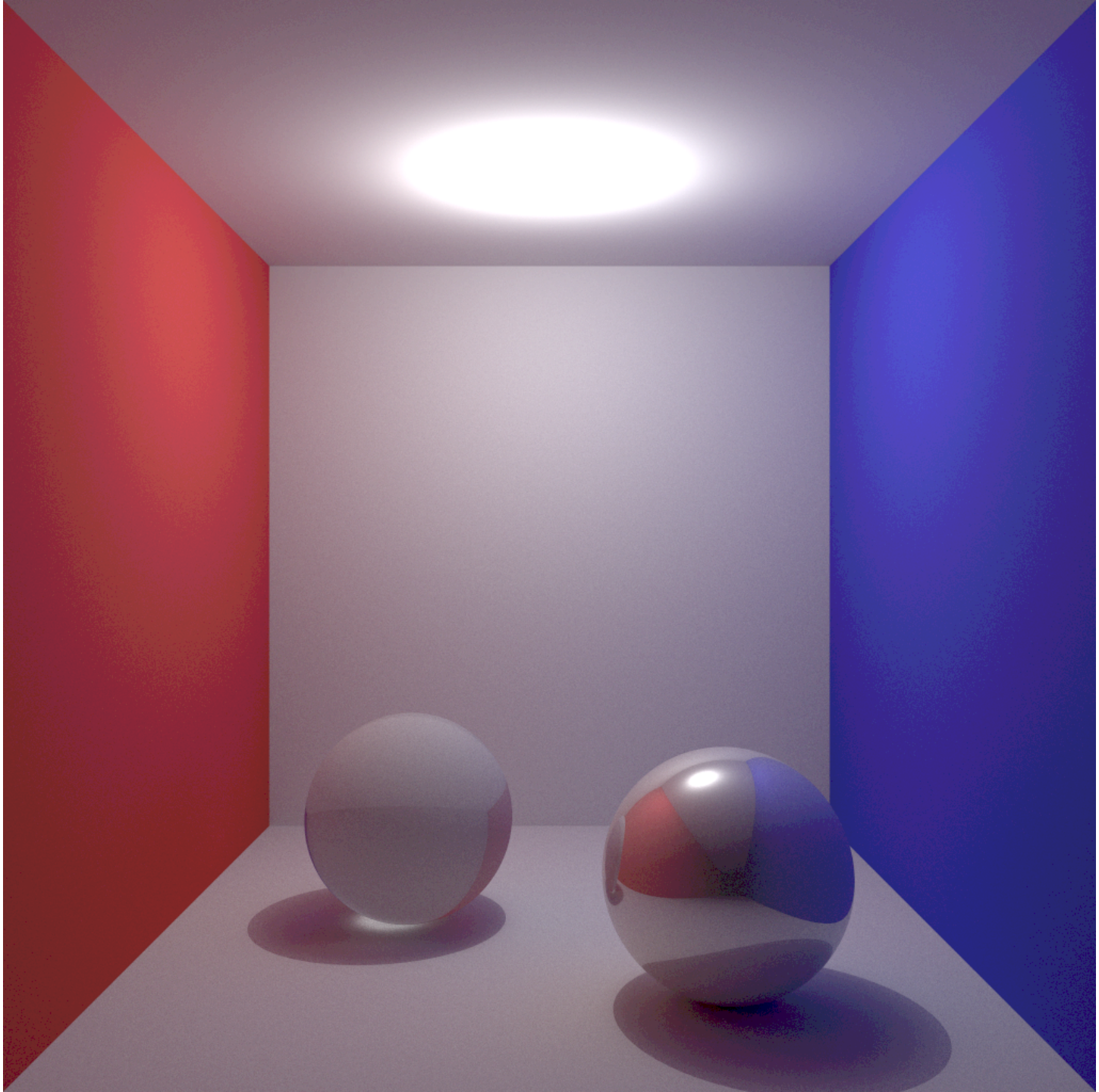
- Reversed direction.
- Trace rays (“photons”) from the light source to the ground behind the sphere.

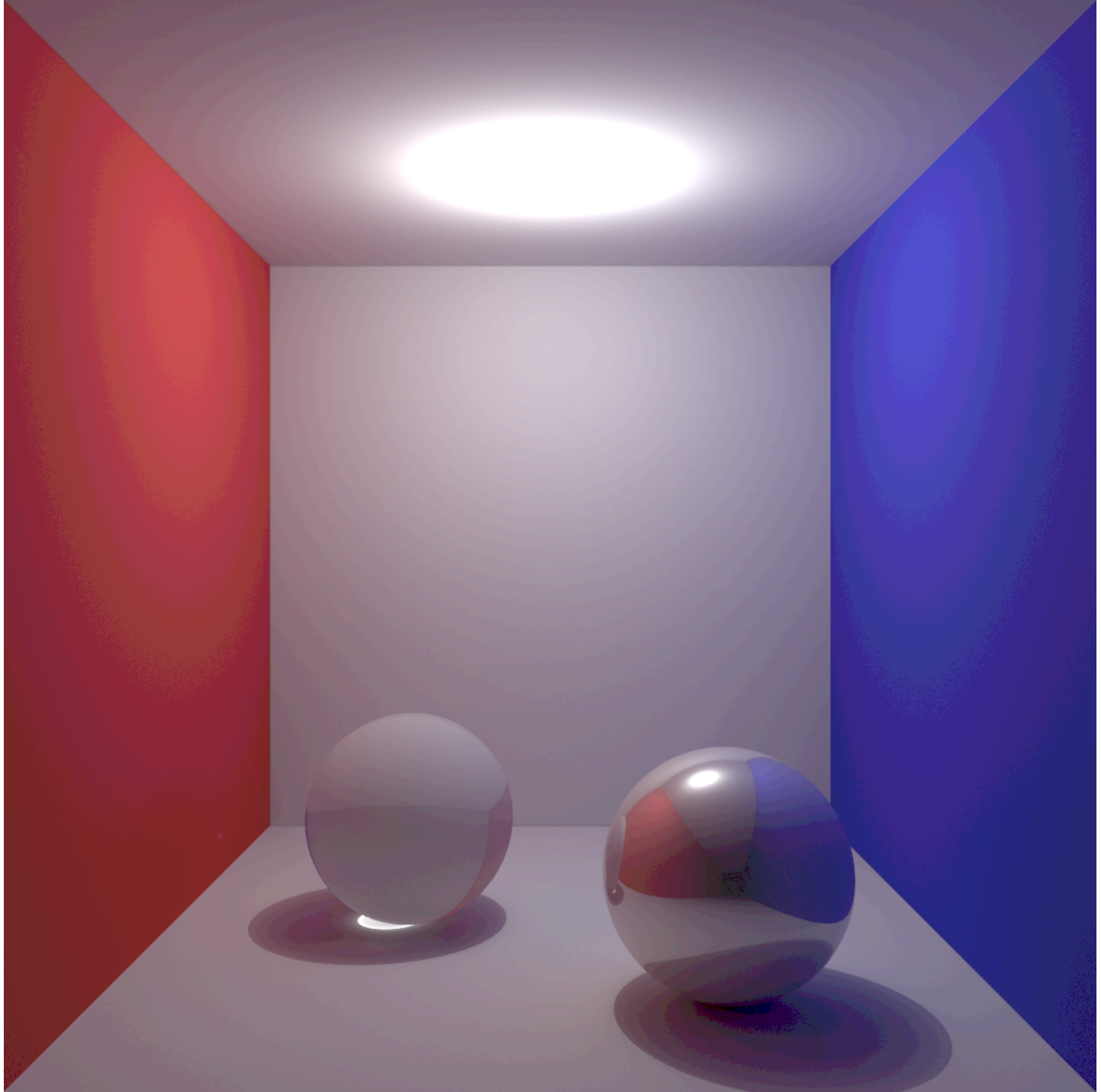


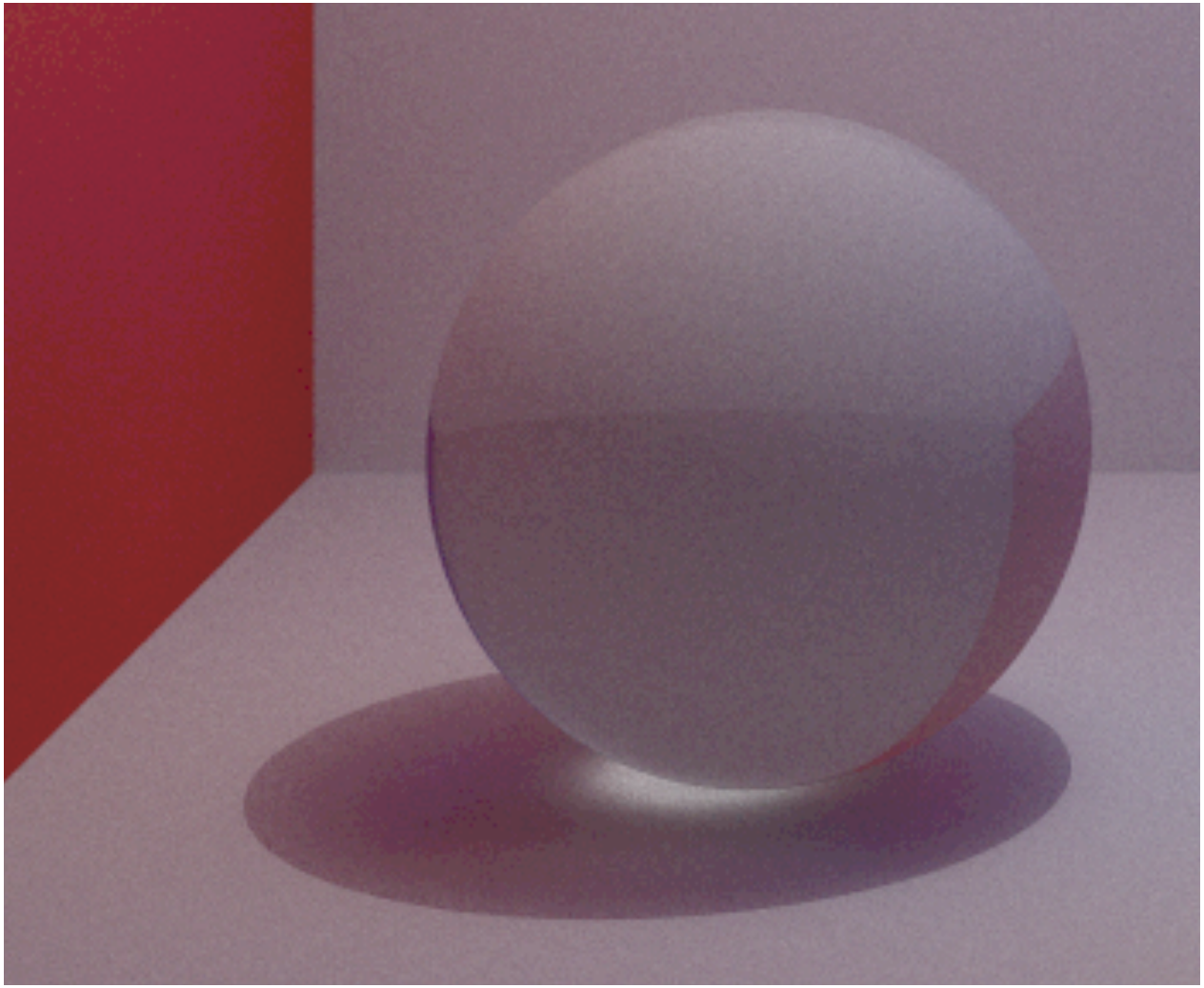
Progressive Photon Mapping

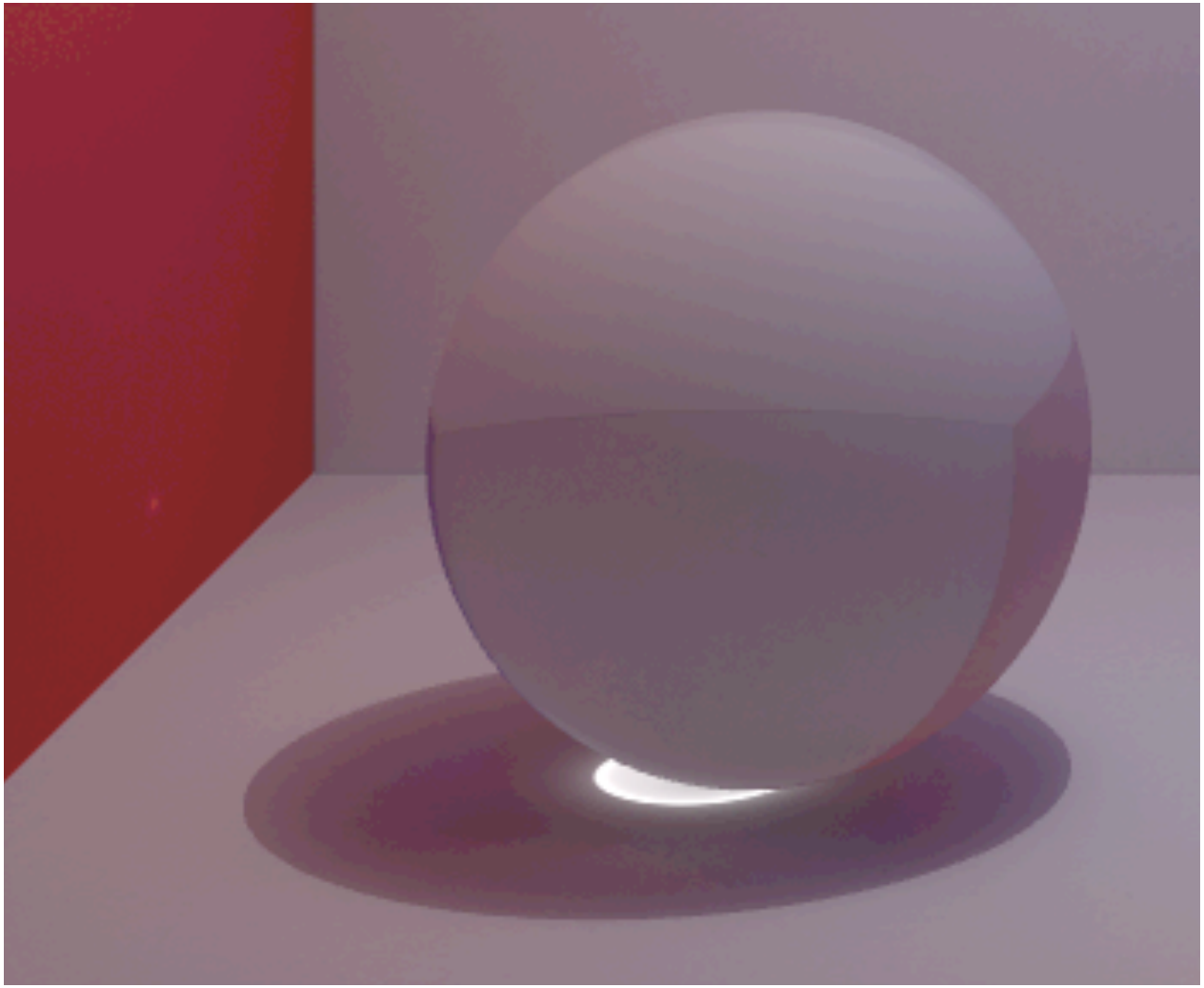
- Reversed direction.
- Trace rays (“photons”) from the light source to the ground behind the sphere.

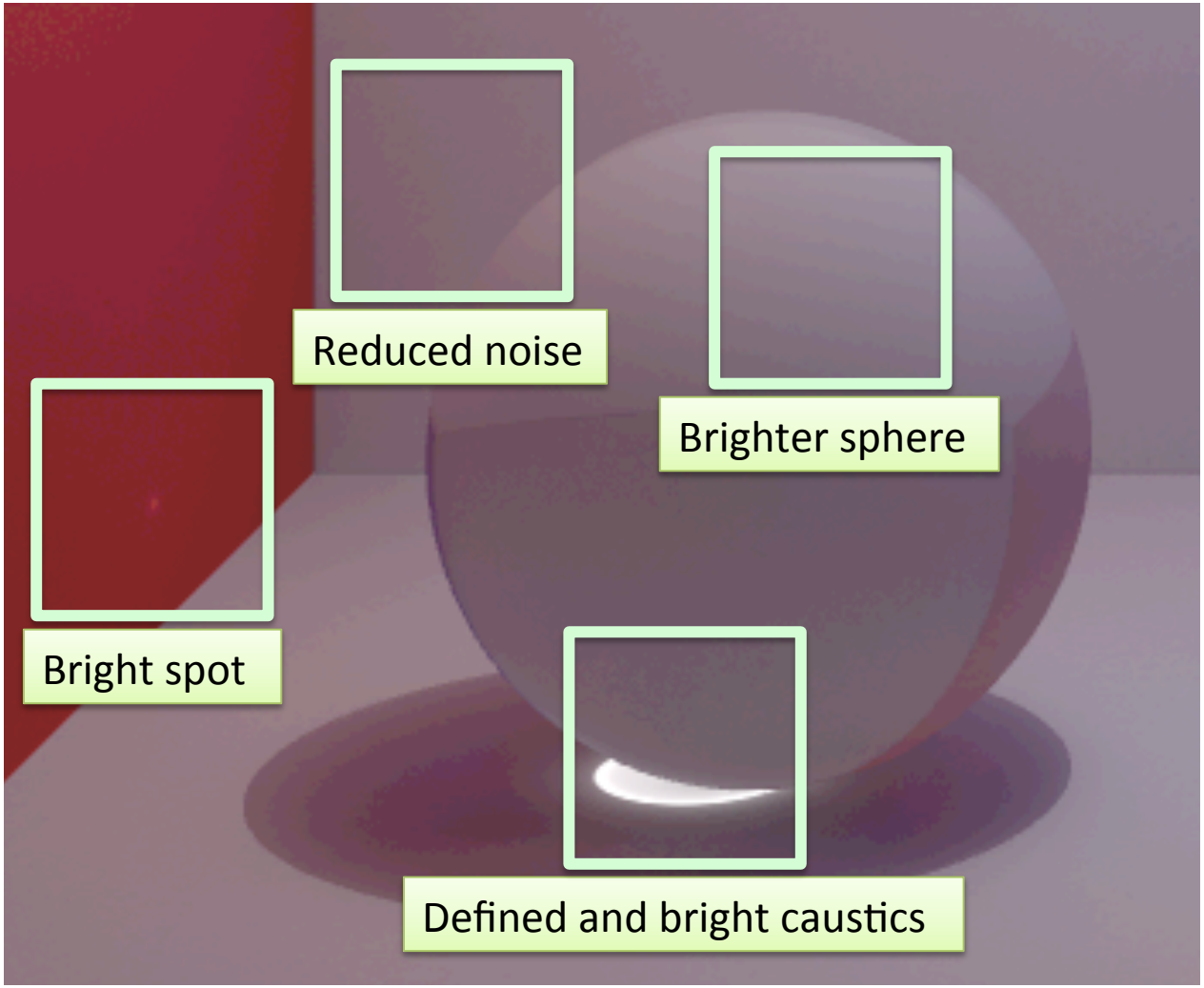








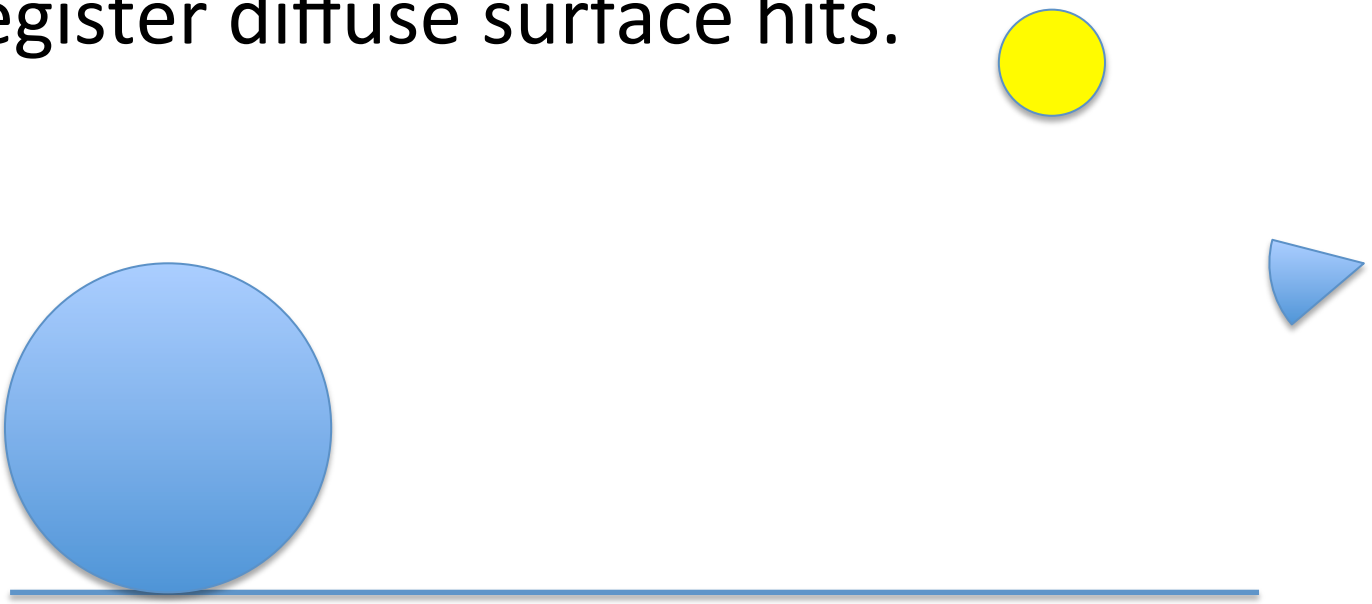



Reduced noise
Brighter sphere
Bright spot
Defined and bright caustics

PPM is multi-pass

1st pass:

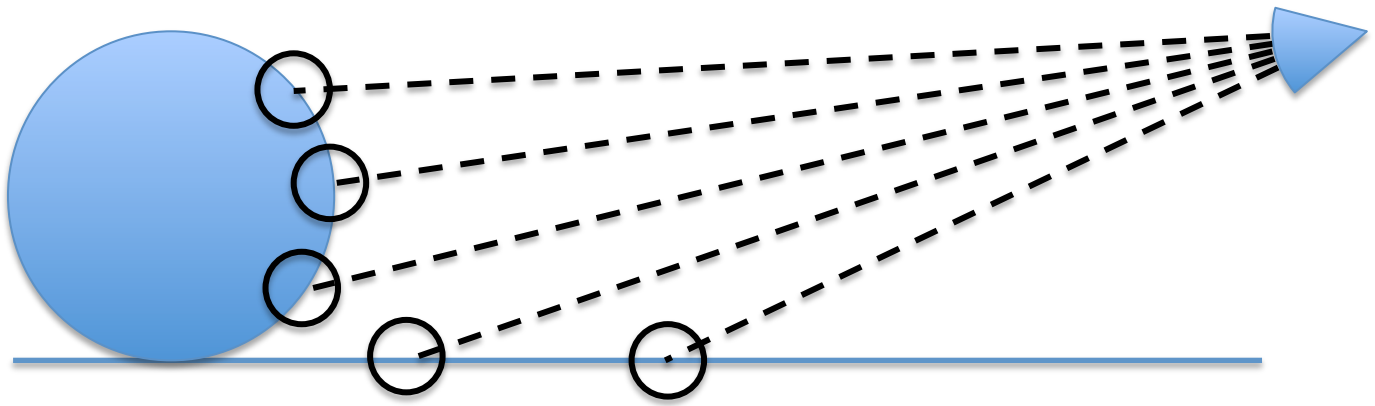
- Forward trace from eye.
- Register diffuse surface hits.



PPM is multi-pass

1st pass:

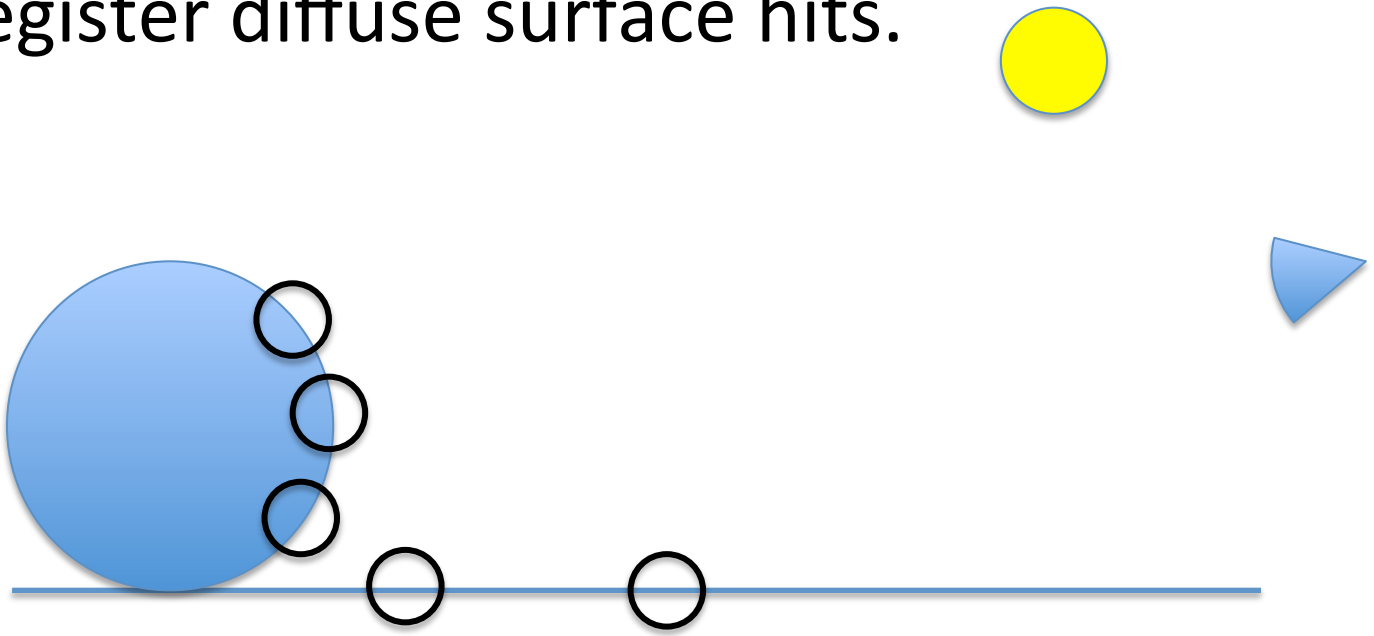
- Forward trace from eye.
- Register diffuse surface hits.



PPM is multi-pass

1st pass:

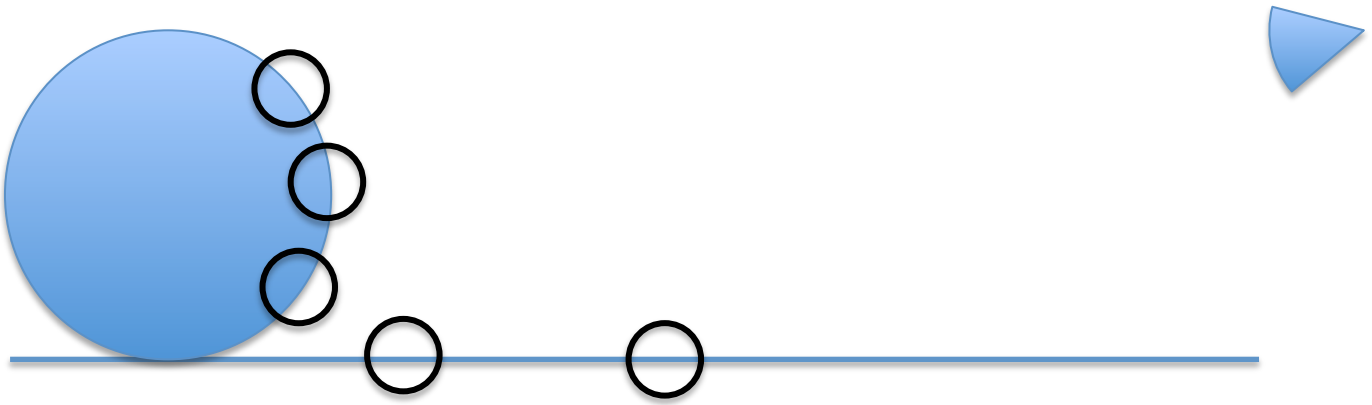
- Forward trace from eye.
- Register diffuse surface hits.



PPM is multi-pass

2nd pass:

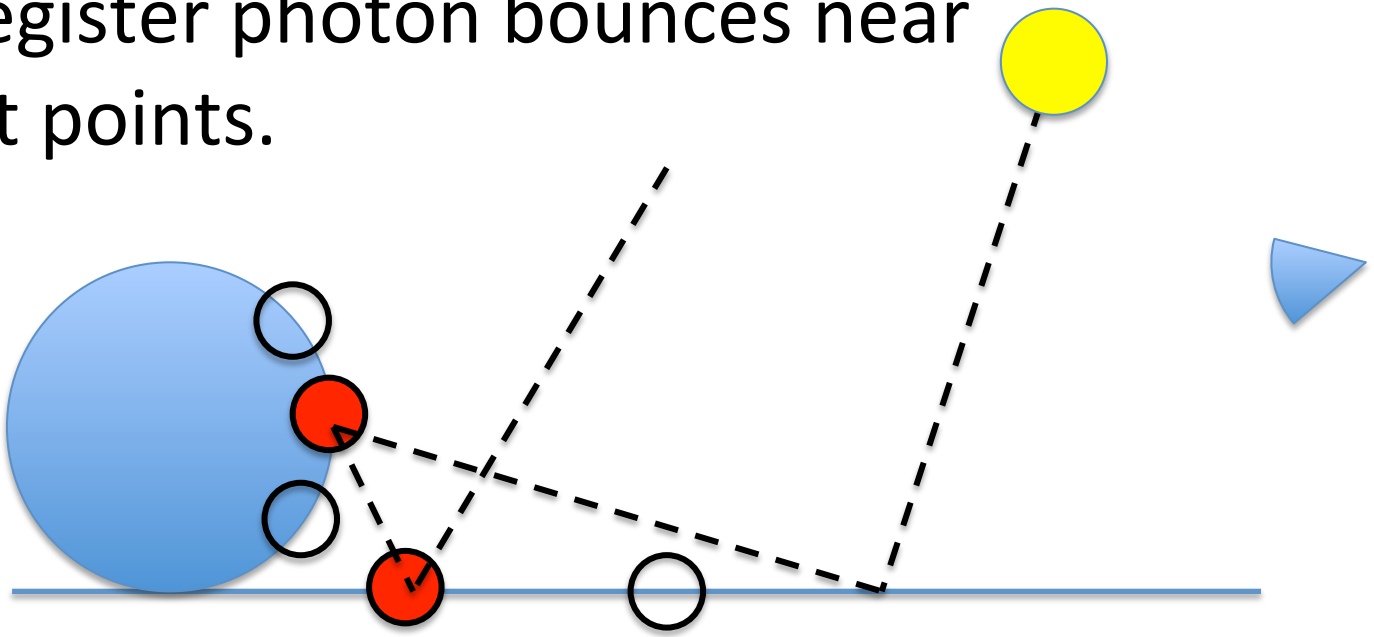
- Send photons from light source.
- Register photon bounces near hit points.



PPM is multi-pass

2nd pass:

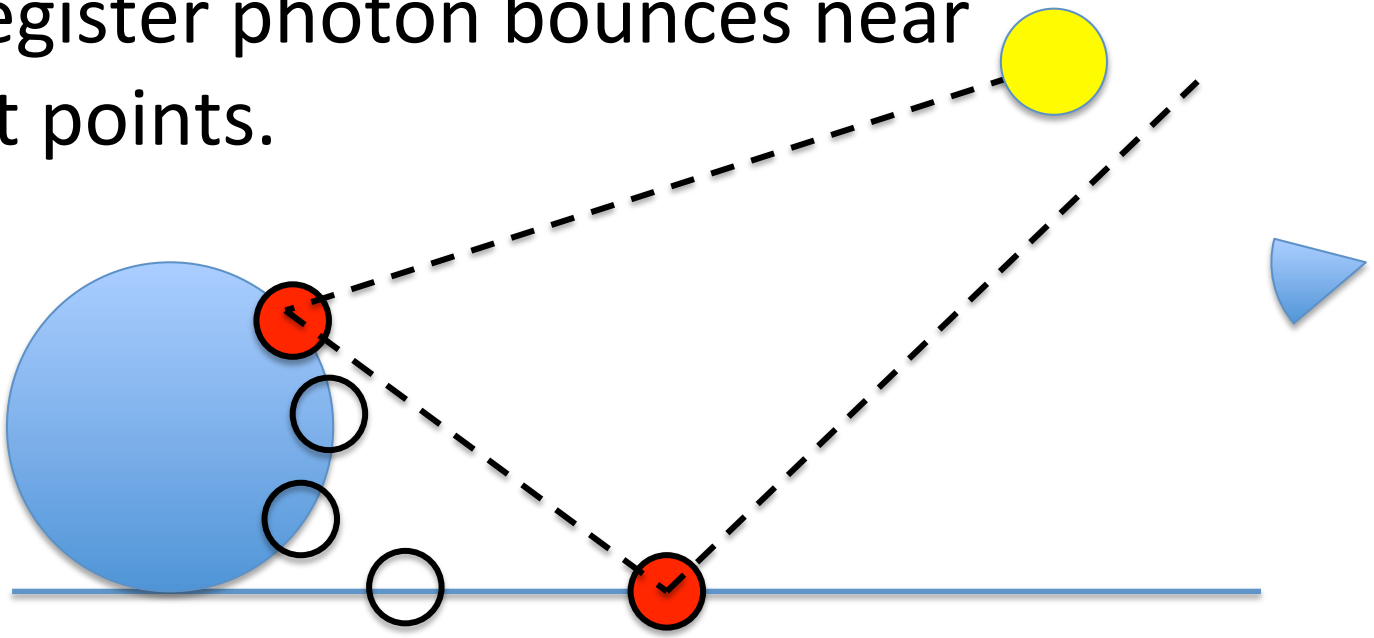
- Send photons from light source.
- Register photon bounces near hit points.



PPM is multi-pass

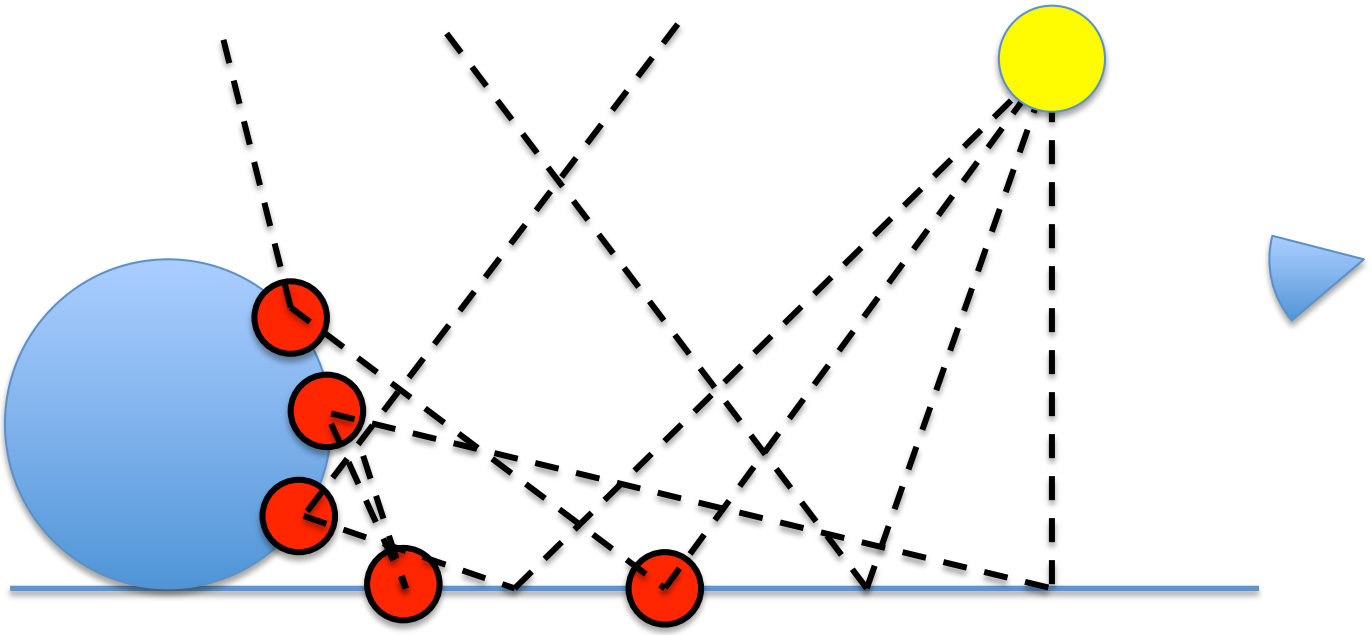
2nd pass:

- Send photons from light source.
- Register photon bounces near hit points.



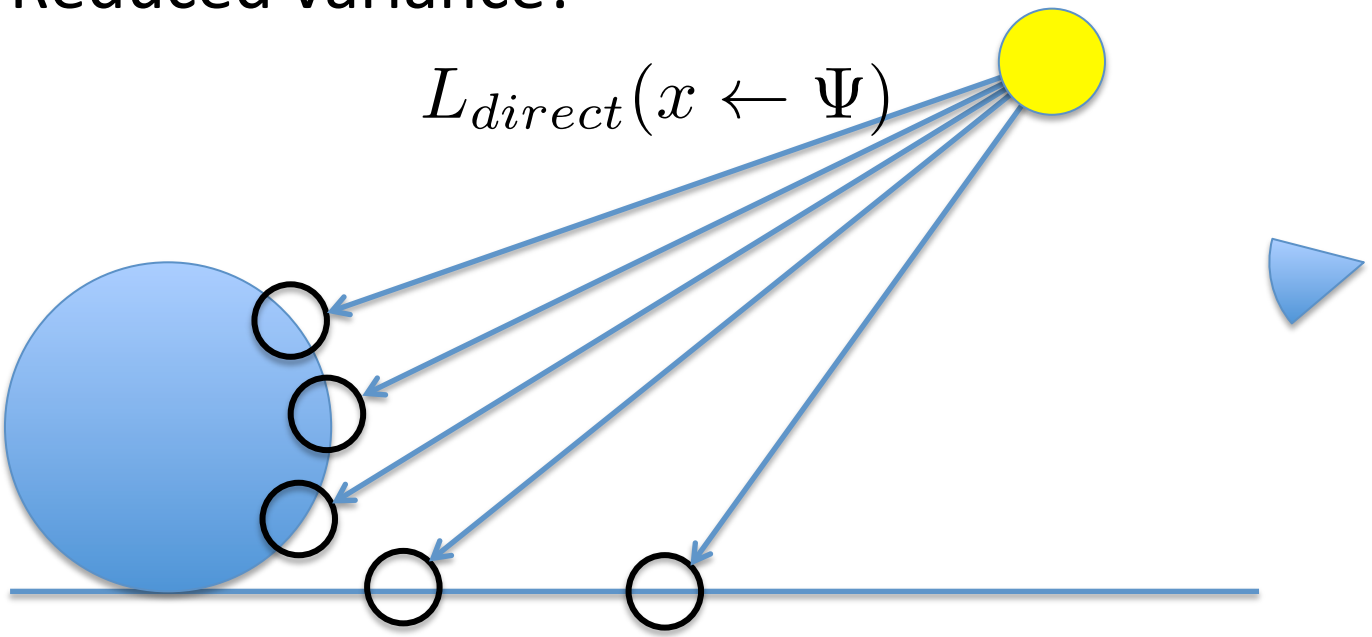
PPM is multi-pass

- Repeat 2nd pass multiple times.
- Continue until image quality is sufficient.



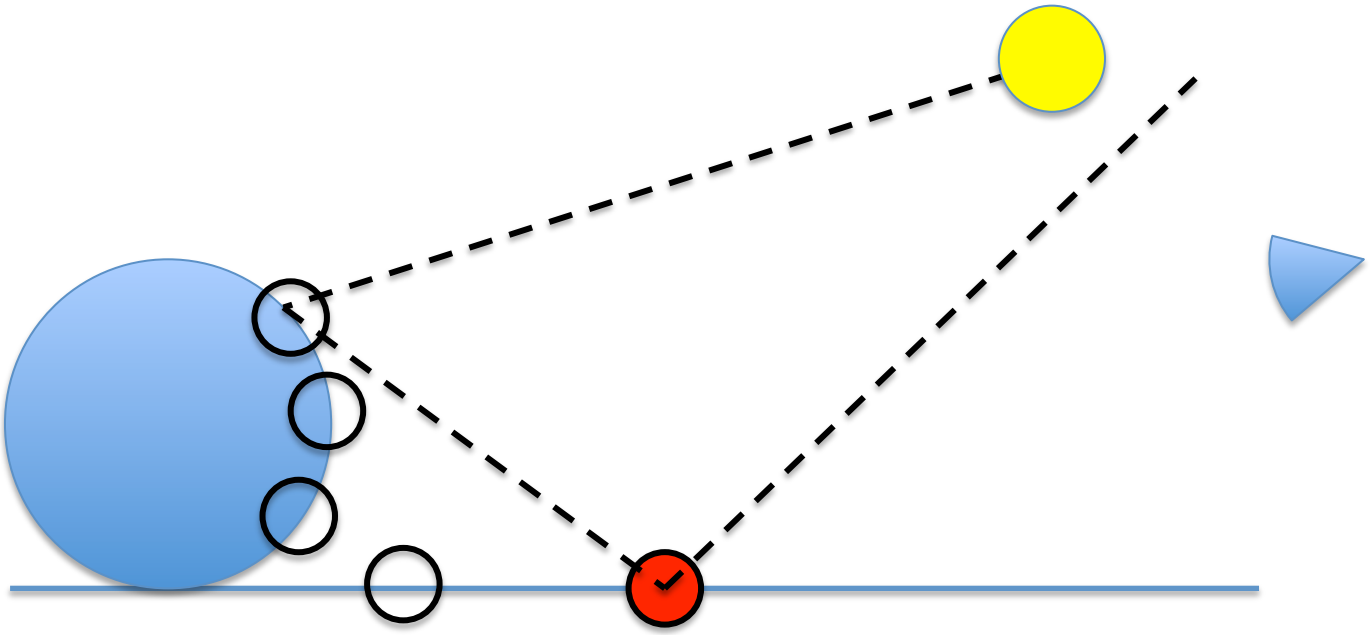
Direct illumination

- Calculate and store direct illumination at each hit point during 1st pass.
  - Reduced variance!



Direct illumination

- Need to alter 2nd pass.
  - Don't register the first bounce of a photon!



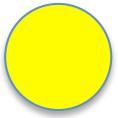
Photon

- Packet of energy.
- Photon power is described using flux:

$$\Phi_p$$

- Bounces around until hitting the background or terminated using Russian Roulette.

Light source



- Point light with intensity *I*.
- Emits photons in random directions.
- Photon flux:

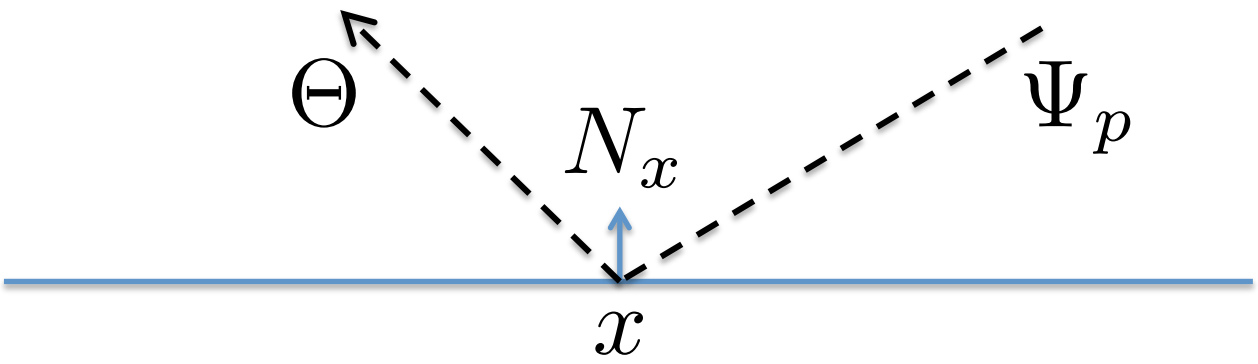
$$\Phi_p = 4\pi I$$

- Photon density automatically attenuates with distance^2.

Diffuse scatter

- When a photon hits a diffuse surface, it scatters in a random direction.
- The flux after a diffuse scatter is given by:

$$\frac{f_r(x, \Psi_p \leftrightarrow \Theta)\Phi_p(x \leftarrow \Psi_p)cos(N_x, \Theta)}{p(\Theta)}$$



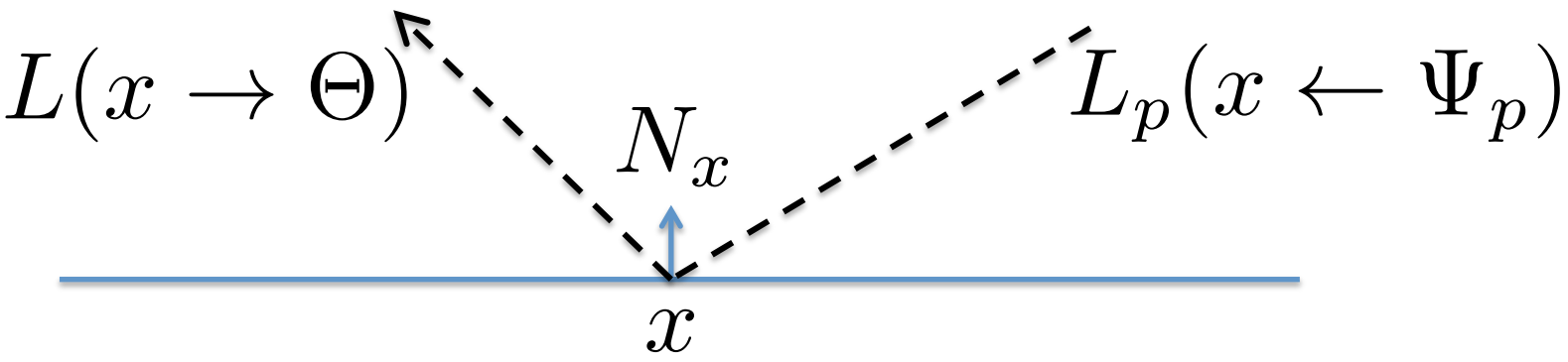
Diffuse scatter

?

$$\frac{f_r(x, \Psi_p \leftrightarrow \Theta)\Phi_p(x \leftarrow \Psi_p)cos(N_x, \Theta)}{p(\Theta)}$$

Diffuse scatter

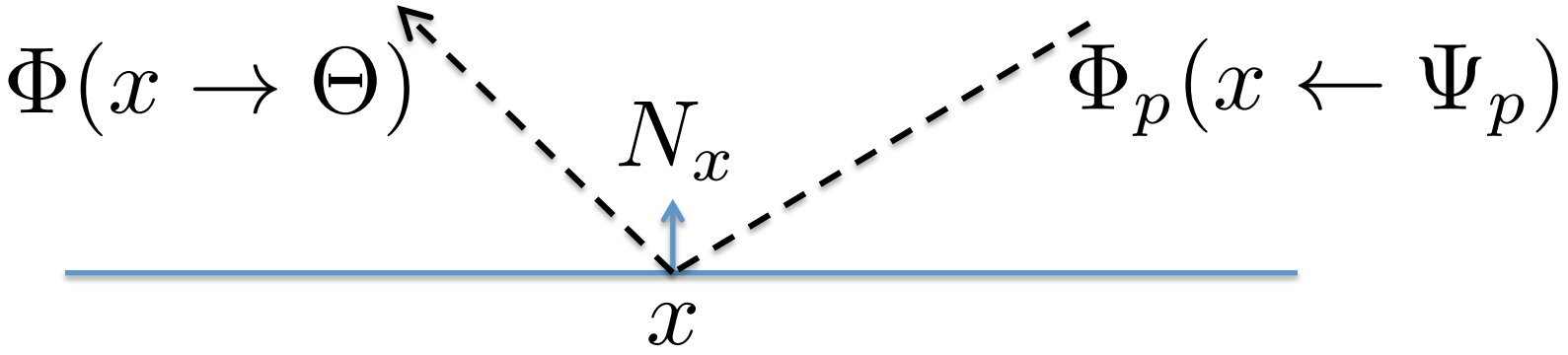
$$L(x \rightarrow \Theta) = f_r(x, \Psi_p \leftrightarrow \Theta) L_p(x \leftarrow \Psi_p) cos(N_x, \Psi_p)$$



Diffuse scatter

$$L = \frac{\mathrm{d}^2\Phi}{\mathrm{d}\omega \mathrm{d}A \cdot cos(N, \Psi)}$$

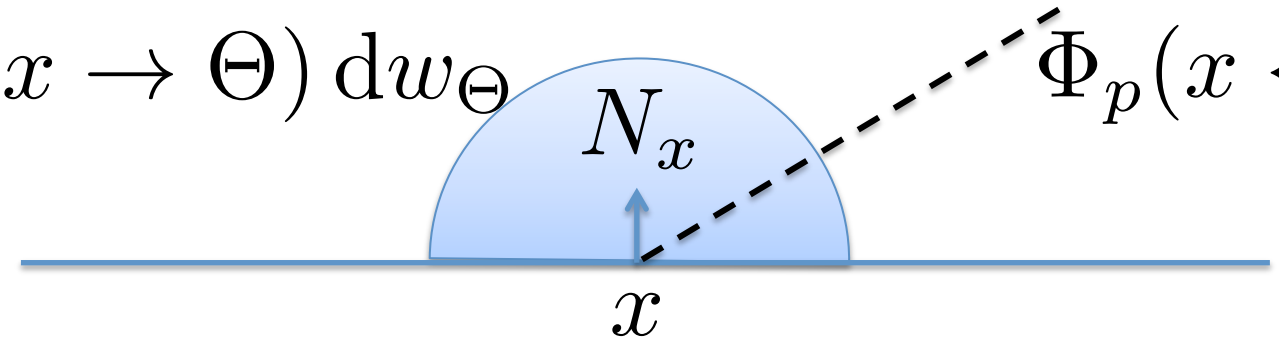
$$\Phi(x \rightarrow \Theta) = f_r(x, \Psi_p \leftrightarrow \Theta)\Phi_p(x \leftarrow \Psi_p)cos(N_x, \Theta)$$



Diffuse scatter

$$\int_{\Omega} \Phi(x \rightarrow \Theta)\, \mathrm{d}w_{\Theta} = \int_{\Omega} f_r(x, \Psi_p \leftrightarrow \Theta) \Phi_p(x \leftarrow \Psi_p) cos(N_x, \Theta)\, \mathrm{d}w_{\Theta}$$

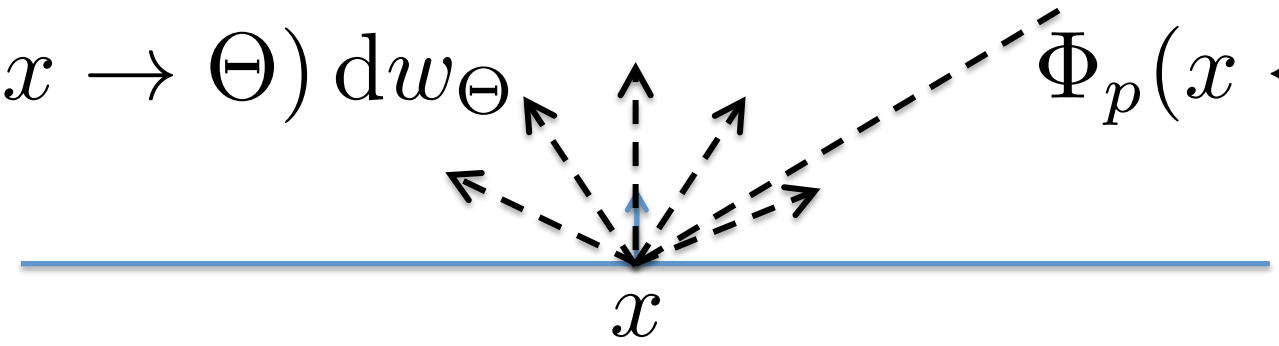
$$\int_{\Omega} \Phi(x \rightarrow \Theta)\, \mathrm{d}w_{\Theta}$$



Diffuse scatter

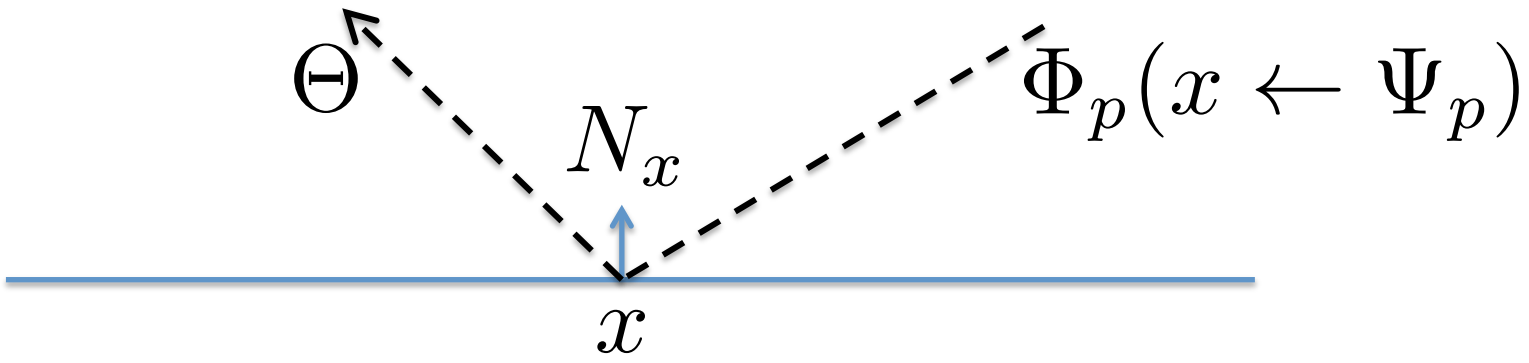
$$\int_{\Omega} \Phi(x \rightarrow \Theta)\, \mathrm{d}w_{\Theta} \approx \frac{1}{n} \sum_{i=1}^{n} \frac{f_r(x, \Psi_p \leftrightarrow \Theta_i)\Phi_p(x \leftarrow \Psi_p)cos(N_x, \Theta_i)}{p(\Theta_i)}$$

$$\int_{\Omega} \Phi(x \rightarrow \Theta)\, \mathrm{d}w_{\Theta}$$



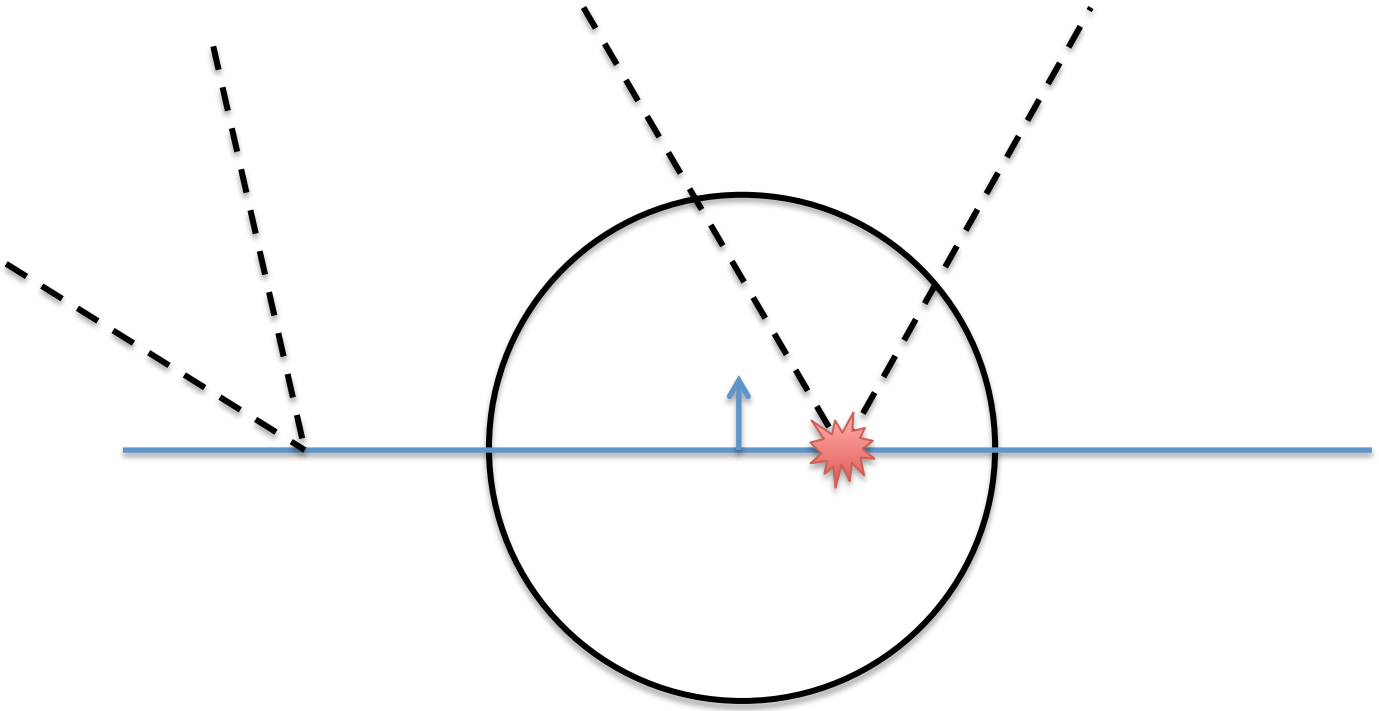
Diffuse scatter

$$\int_{\Omega} \Phi(x \rightarrow \Theta)\, \mathrm{d}w_{\Theta} \approx \frac{f_r(x, \Psi_p \leftrightarrow \Theta)\Phi_p(x \leftarrow \Psi_p)cos(N_x, \Theta)}{p(\Theta)}$$



Hit point

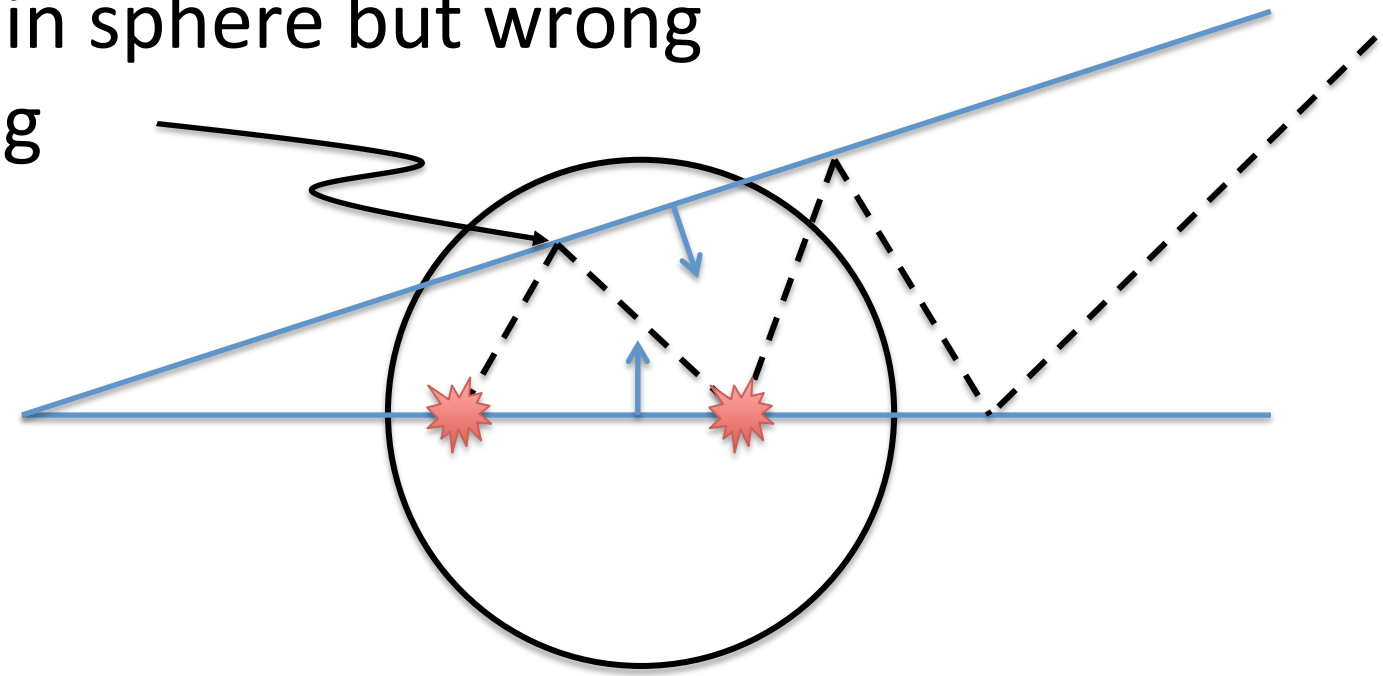
- Sphere with radius *r*.
- Collects photon bounces.



Hit point

- Ignore bounces at surfaces facing the normal of the hit point.

Within sphere but wrong facing



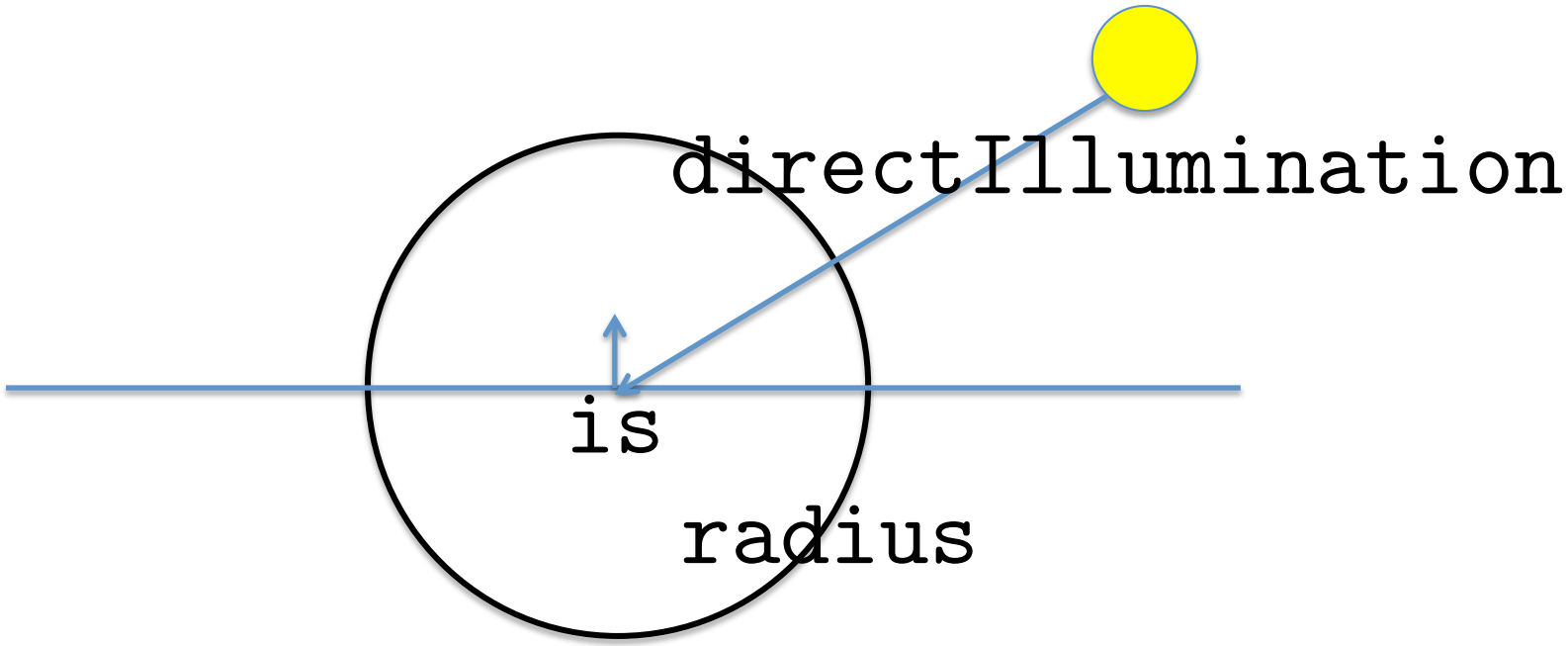
Hit point

```
struct Hitpoint {
    Intersection is;
    int pixelX, pixelY;
    float pixelWeight;
    float radius;
    Color directIllumination;
    float photonCount;
    int newPhotonCount;
    Color totalFlux;
};
```

Owner pixel

Photon statistics

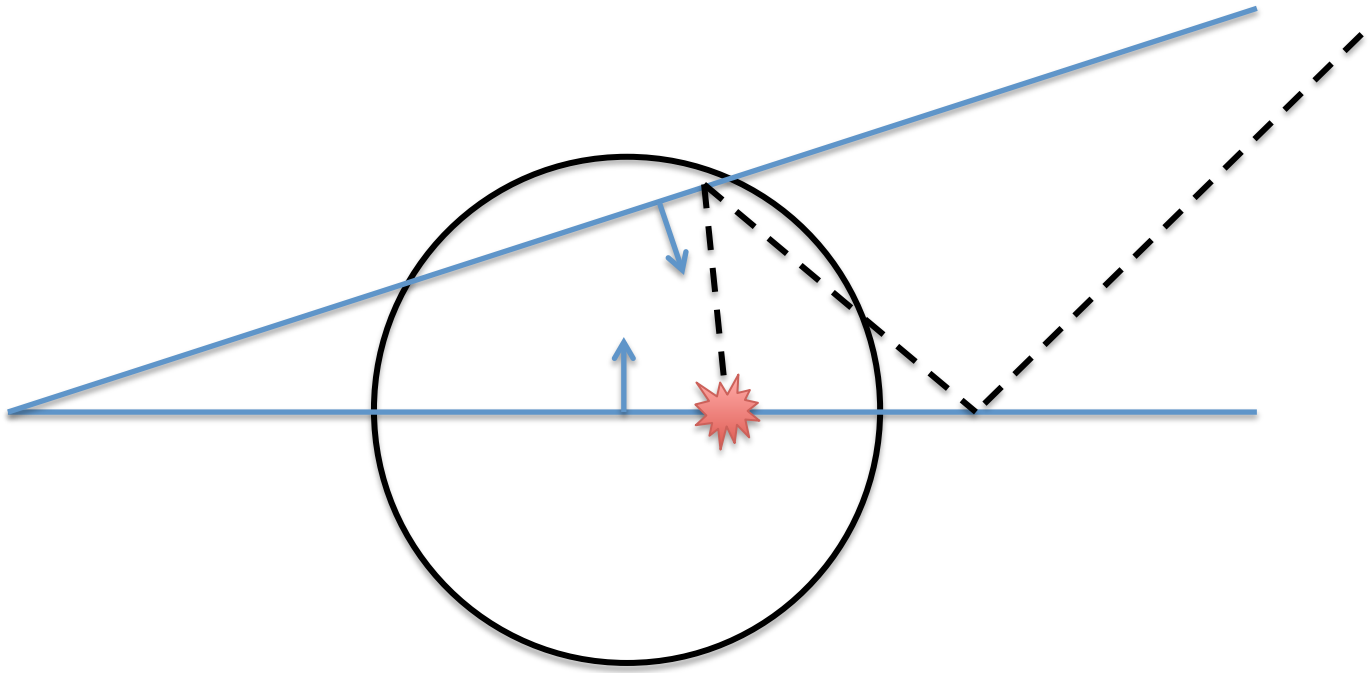
Hit point



Hit point

`hit.newPhotonCount += 1`

`hit.totalFlux +=` $\Phi_p \cdot$ `hit.is.mMaterial->evalBRDF (hit.is ,` $-\Psi_p$ `)`



Hit point

- A hit point contributes the following illumination to the “owner” pixel.

$$(\texttt{totalFlux} \,/\, (n_{total} \cdot \pi\, \texttt{radius}^2) + \texttt{directIllumination}) \cdot \texttt{pixelWeight}$$

Bias?

- Large sample spheres will result in bias!
  - Illumination is “blurred”.
- Solved by reducing the radius of each sample sphere after each photon iteration.
  - Amount of reduction depends on the number of photons received.

Bias?

$A = \texttt{hit.photonCount} + \texttt{hit.newPhotonCount}$

$B = \texttt{hit.photonCount} + \alpha \cdot \texttt{hit.newPhotonCount}$

$\texttt{hit.radius *=} \sqrt{\frac{B}{A}}$

$\texttt{hit.totalFlux *=} \frac{B}{A}$

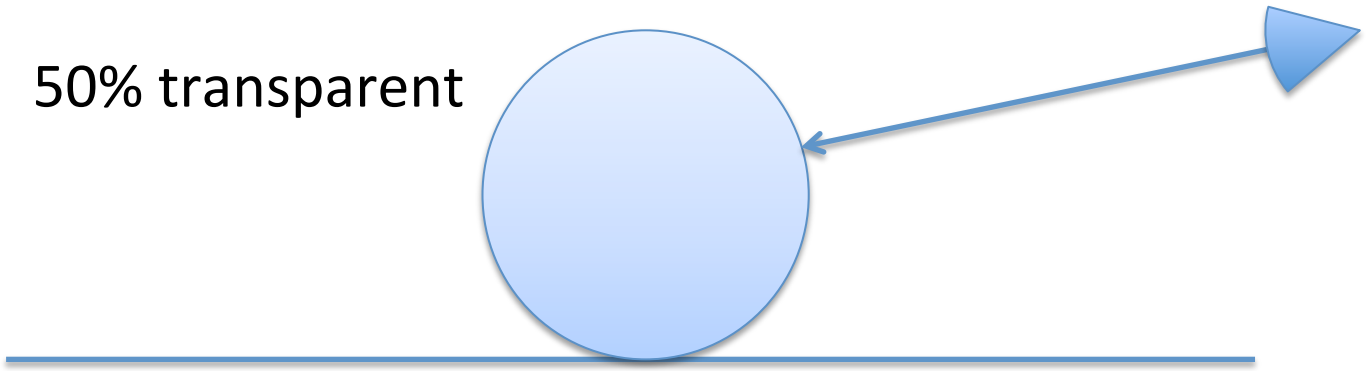
$\texttt{hit.photonCount} = B$

$\texttt{hit.newPhotonCount} = 0$

Reflection and refraction

1st pass:

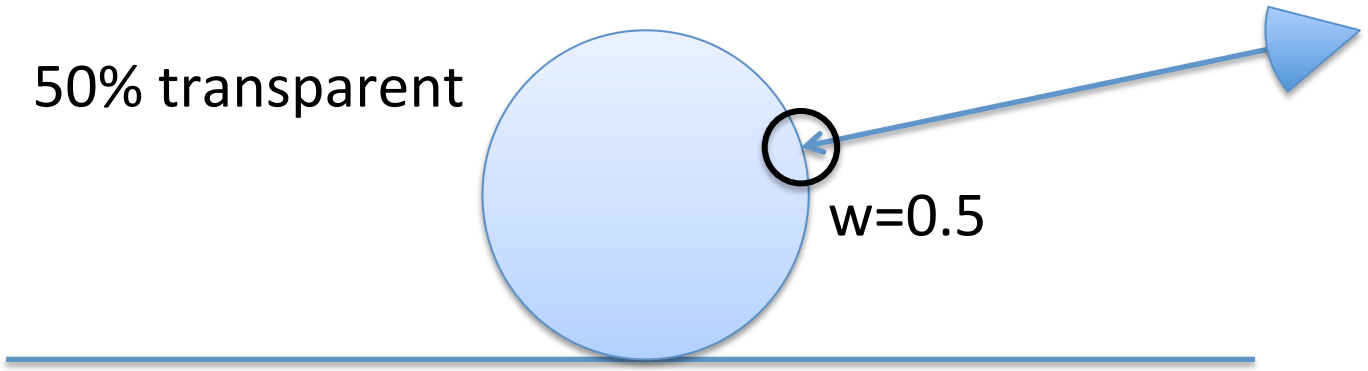
- Perform Whitted style reflection and refraction.
- Add hit points at diffuse surfaces.



Reflection and refraction

1st pass:

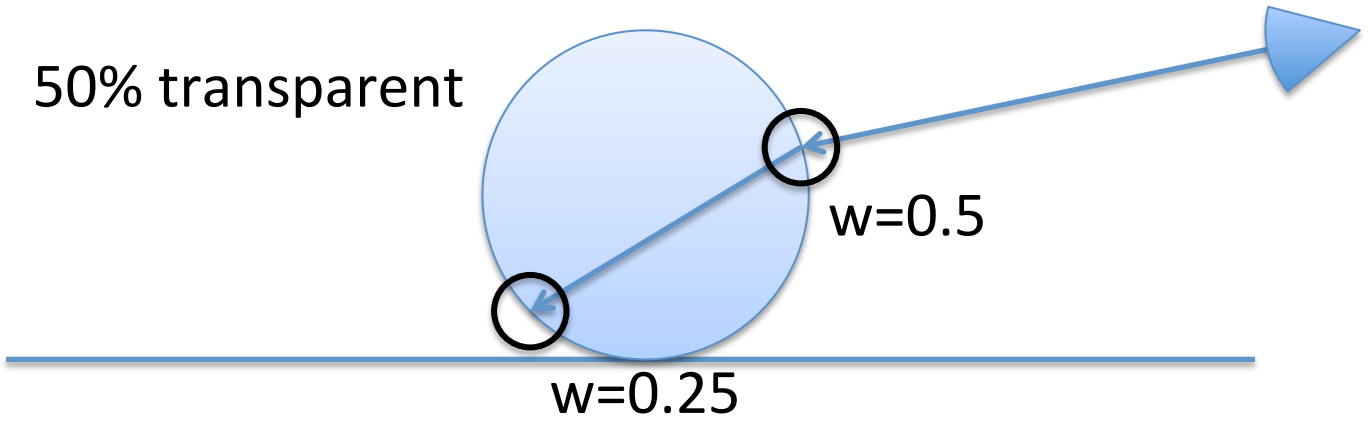
- Perform Whitted style reflection and refraction.
- Add hit points at diffuse surfaces.



Reflection and refraction

1st pass:

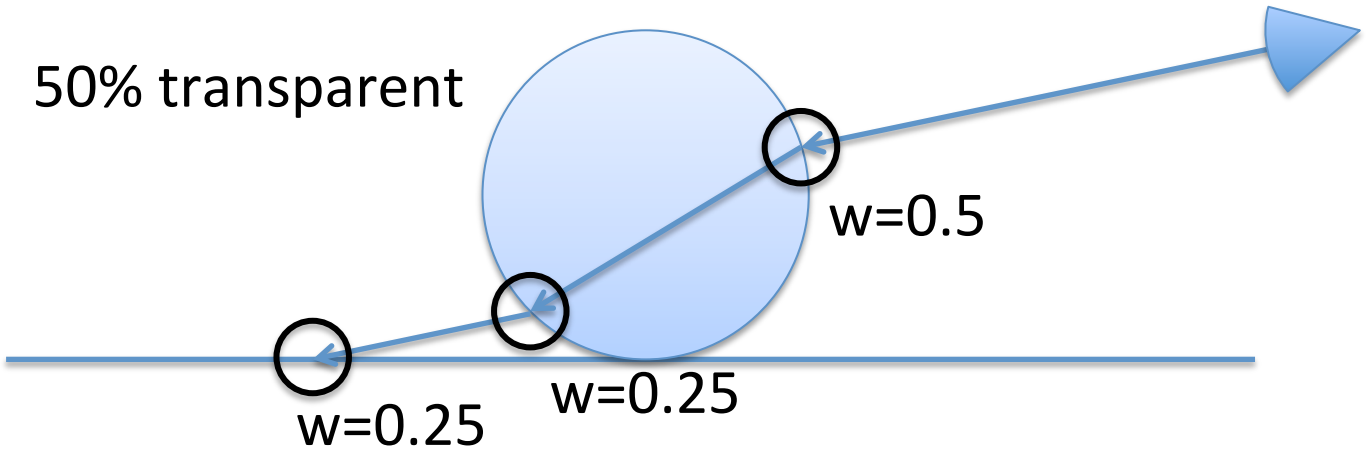
- Perform Whitted style reflection and refraction.
- Add hit points at diffuse surfaces.



Reflection and refraction

1st pass:

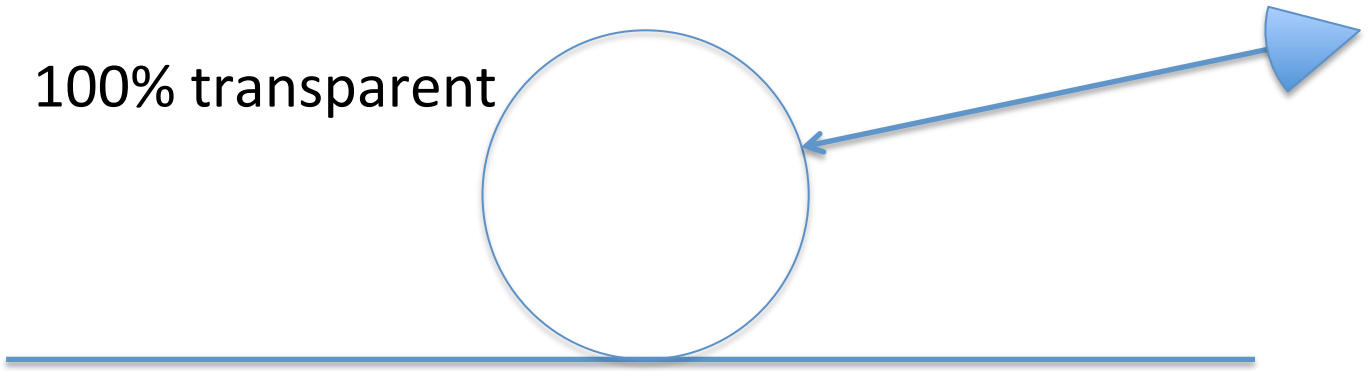
- Perform Whitted style reflection and refraction.
- Add hit points at diffuse surfaces.



Reflection and refraction

1st pass:

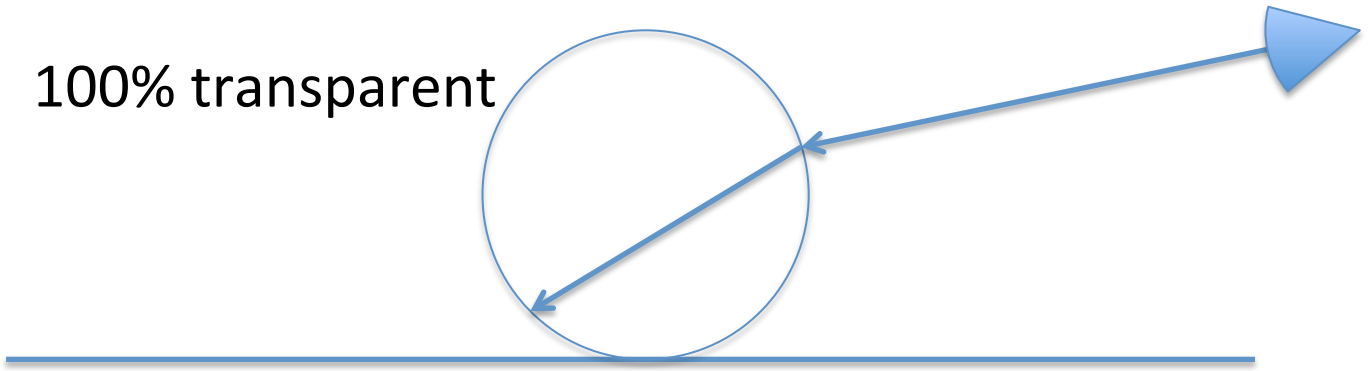
- Perform Whitted style reflection and refraction.
- Add hit points at diffuse surfaces.



Reflection and refraction

1st pass:

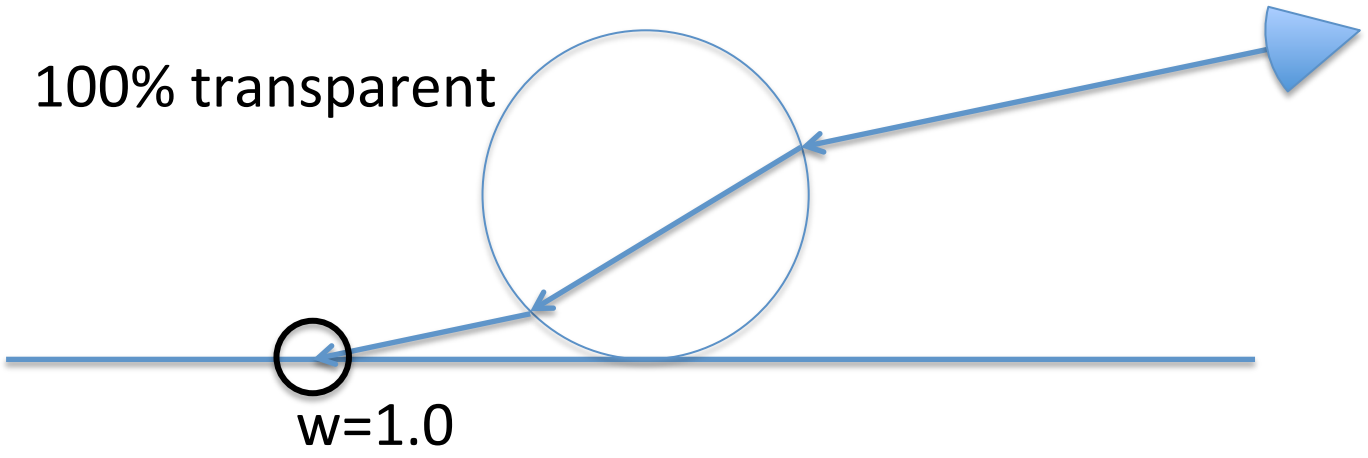
- Perform Whitted style reflection and refraction.
- Add hit points at diffuse surfaces.



Reflection and refraction

1st pass:

- Perform Whitted style reflection and refraction.
- Add hit points at diffuse surfaces.



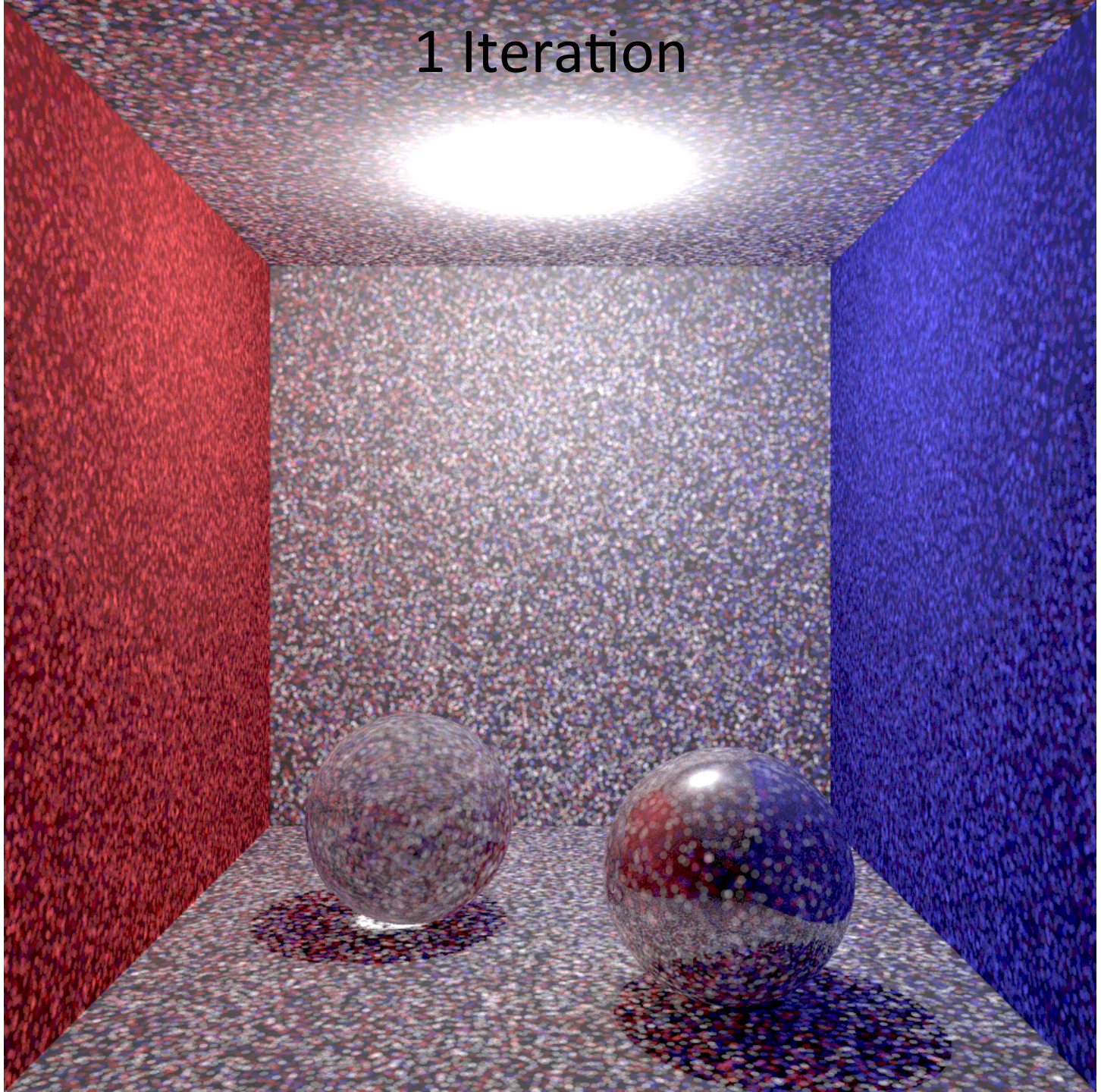
Reflection and refraction

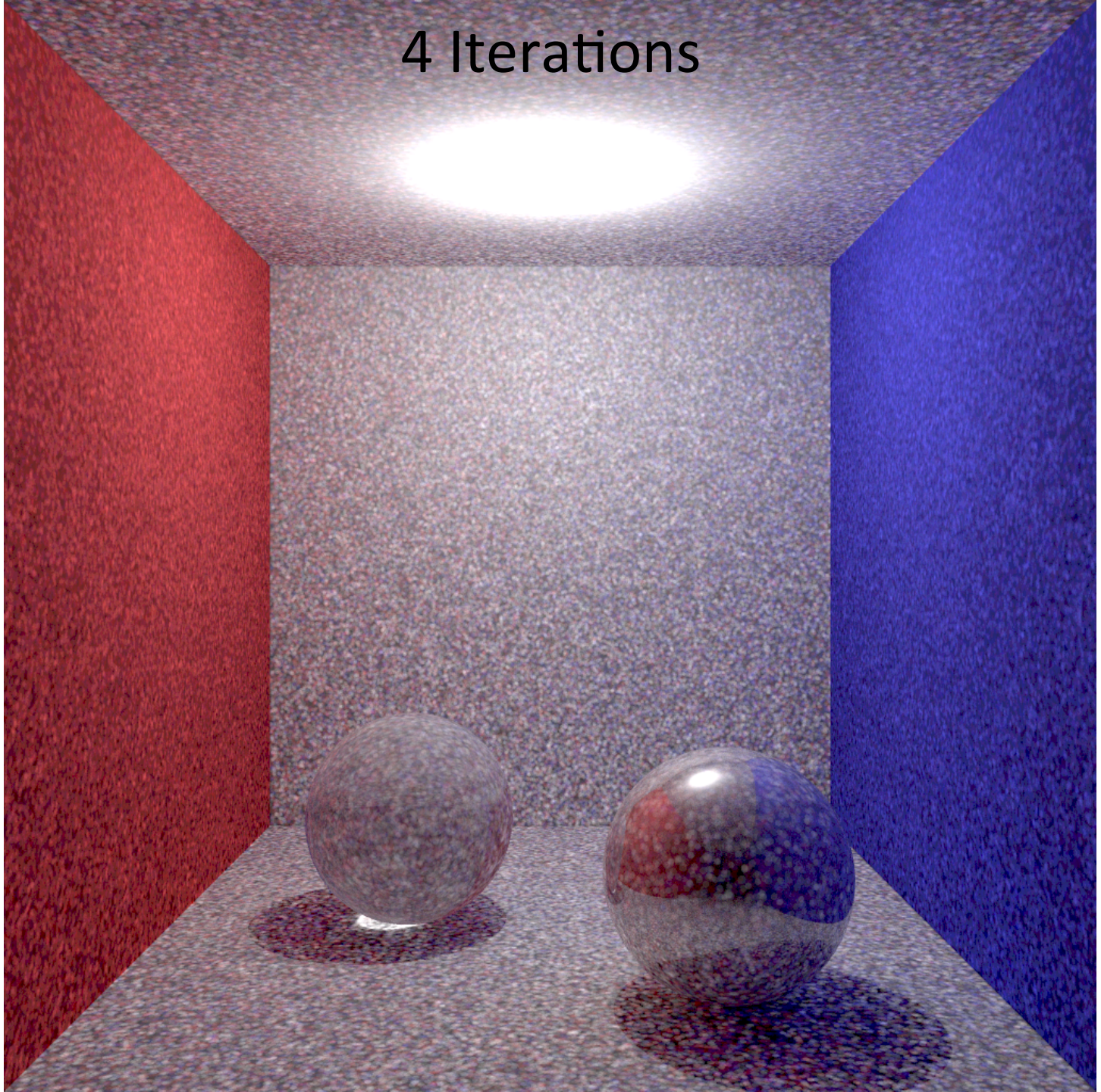
2nd pass:

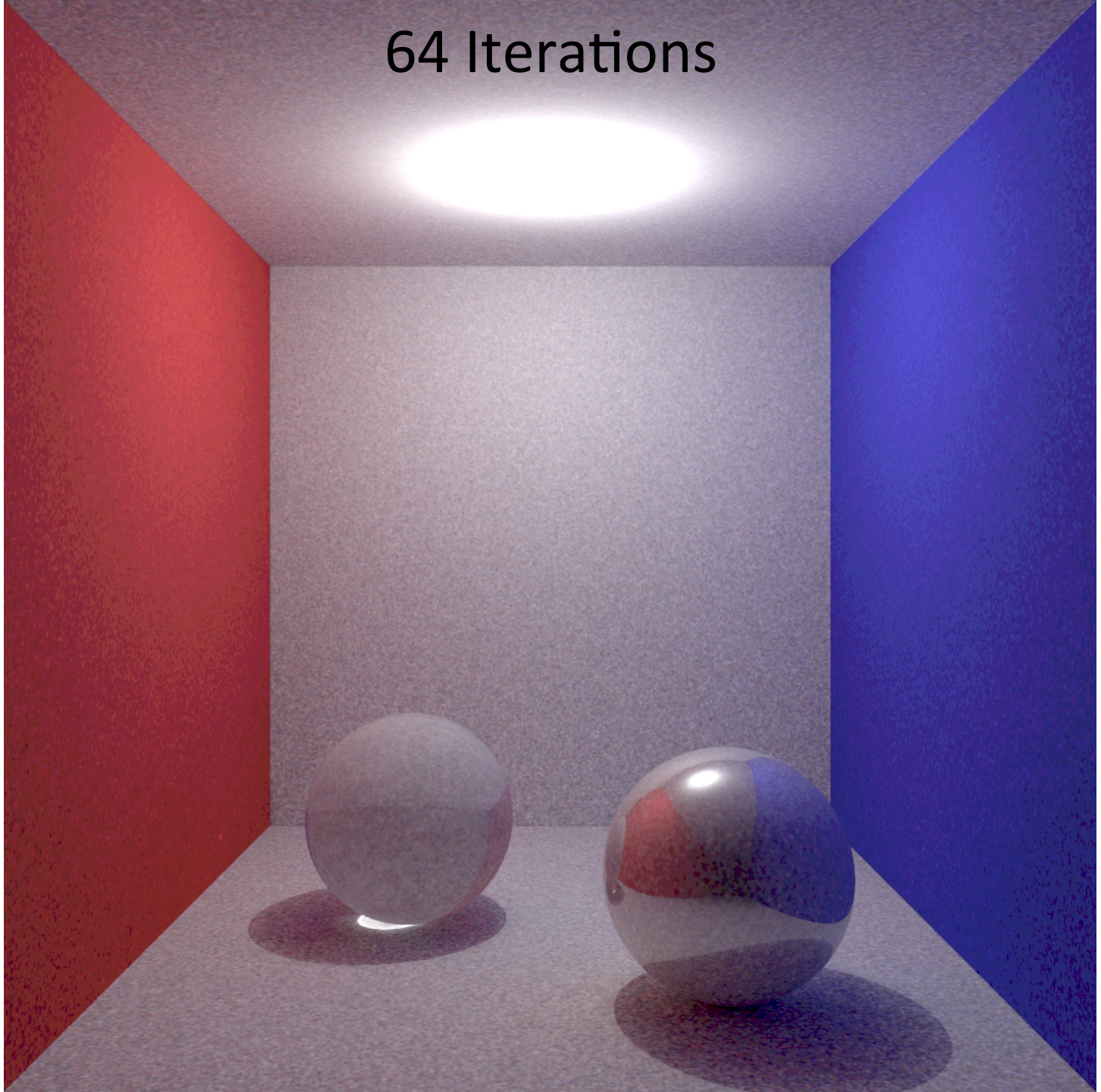
- Randomize the path of each photon.
- Use russian roulette to choose between:
  - Reflection.
  - Refraction.
  - Diffuse scatter.
  - Absorption.

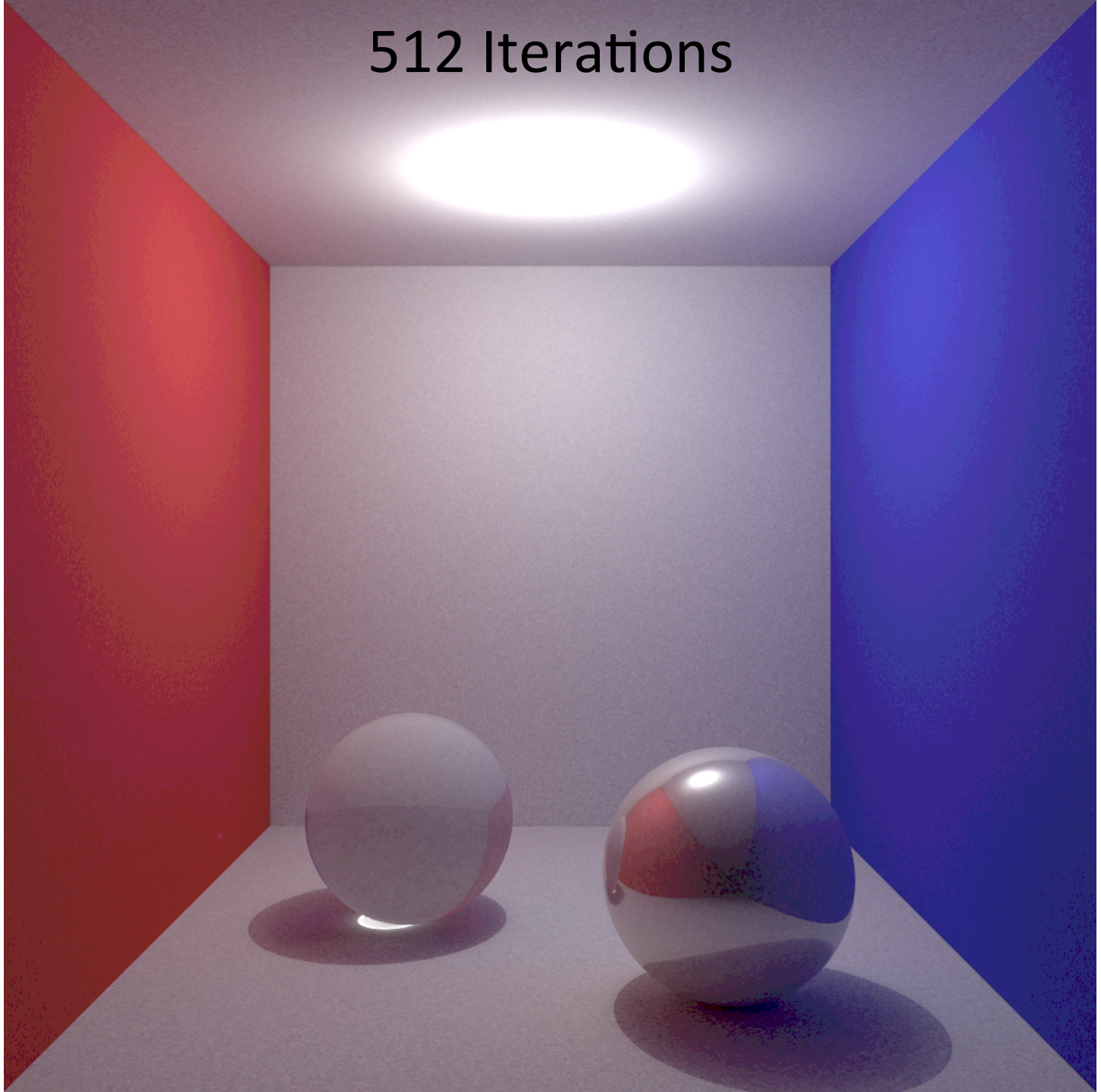
Example

- Radius reduction: $\alpha = 0.7$
- 100 000 photons/iteration

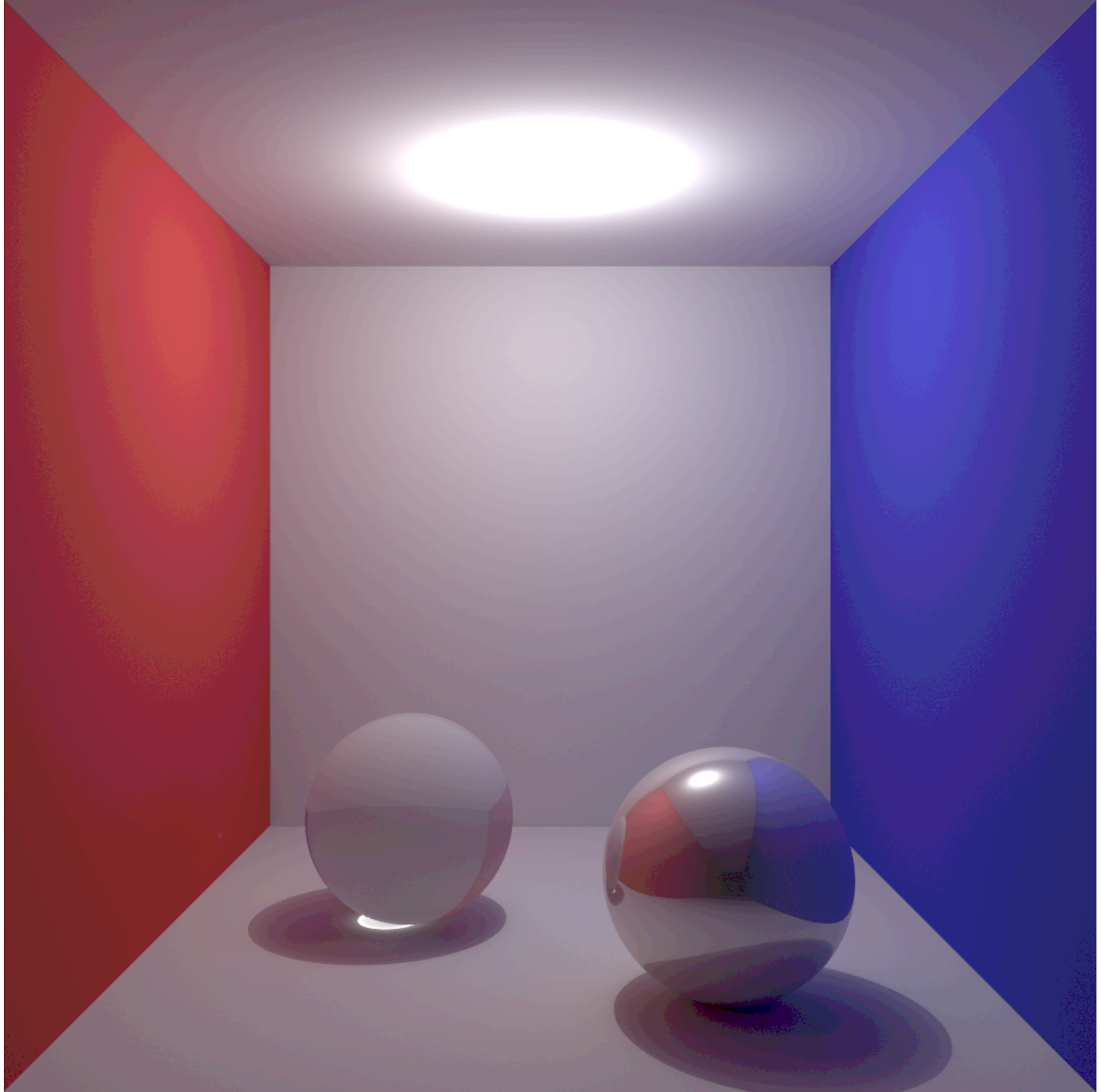

1 Iteration

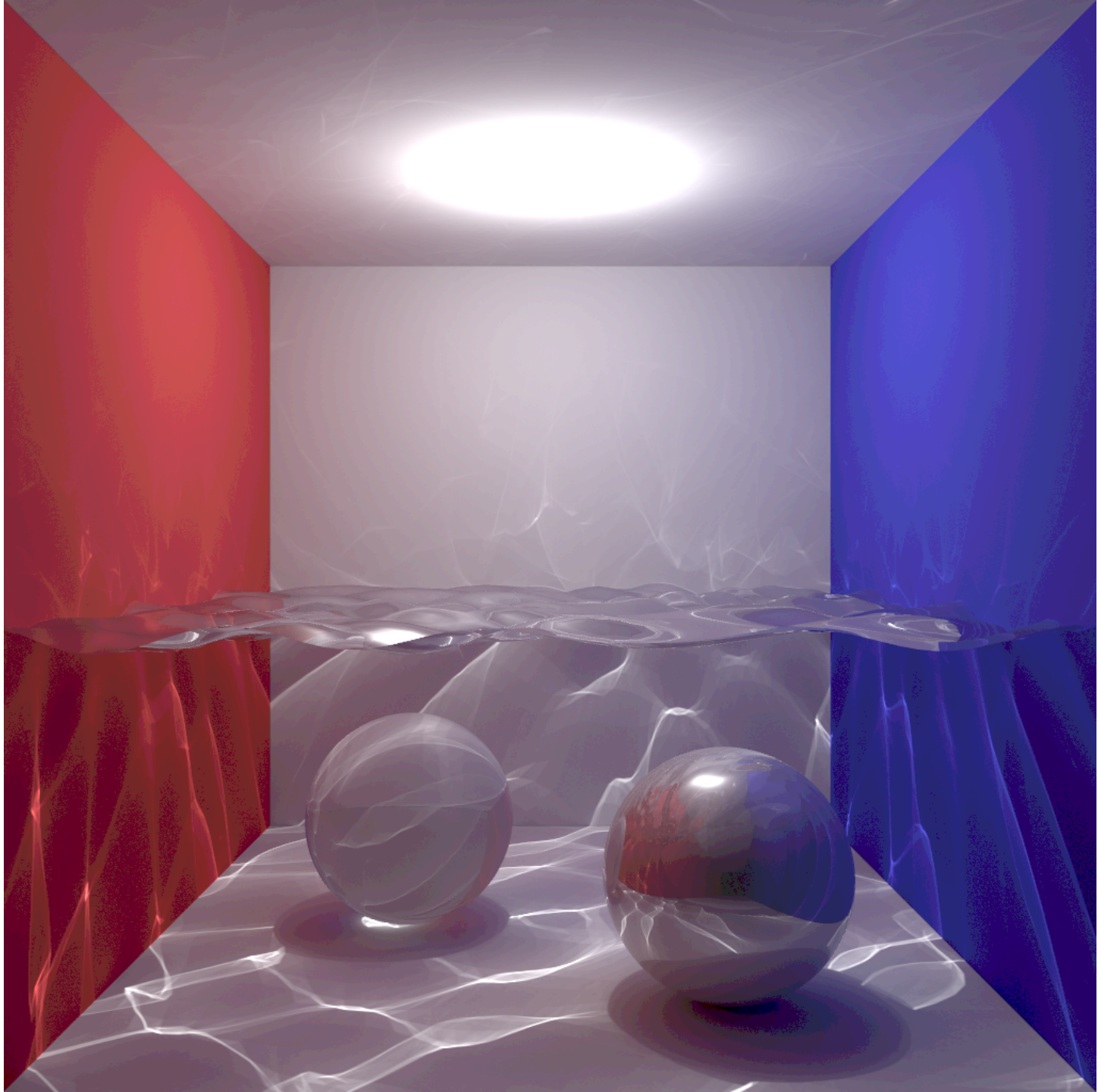

4 Iterations

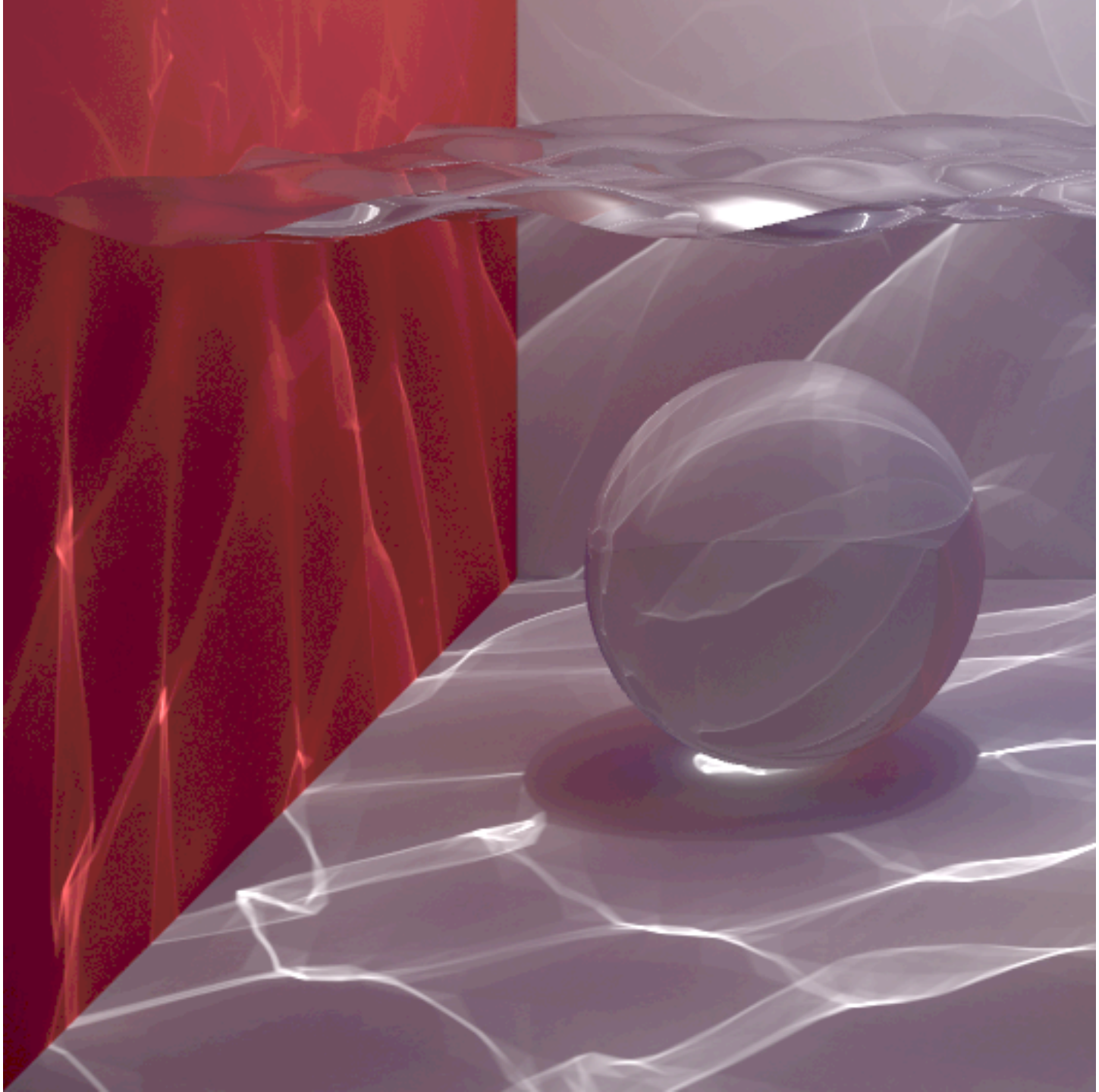

64 Iterations


512 Iterations

Water







The end

Progressive Photon Mapping

http://graphics.ucsd.edu/~henrik/papers/progressive_photon_mapping/