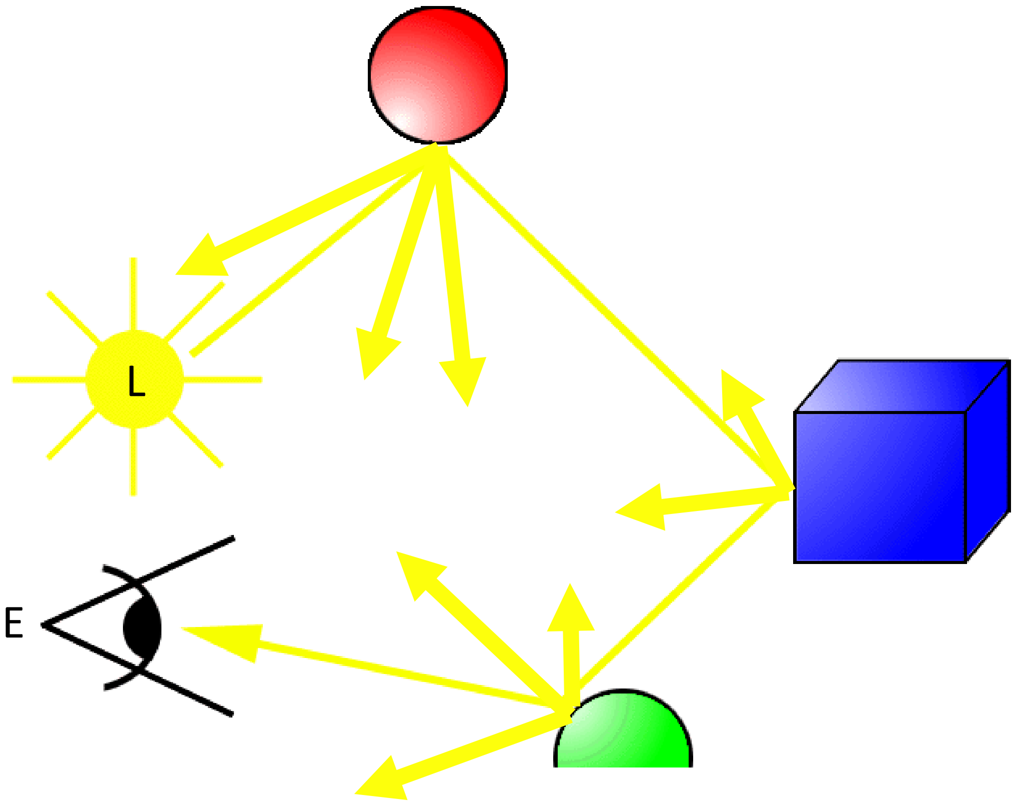
Path Tracing II

Computer Graphics
Fall Semester 2025 FHNW

S. Felix

University of Applied Sciences and Arts Northwestern Switzerland
School of Computer Science

Light Paths

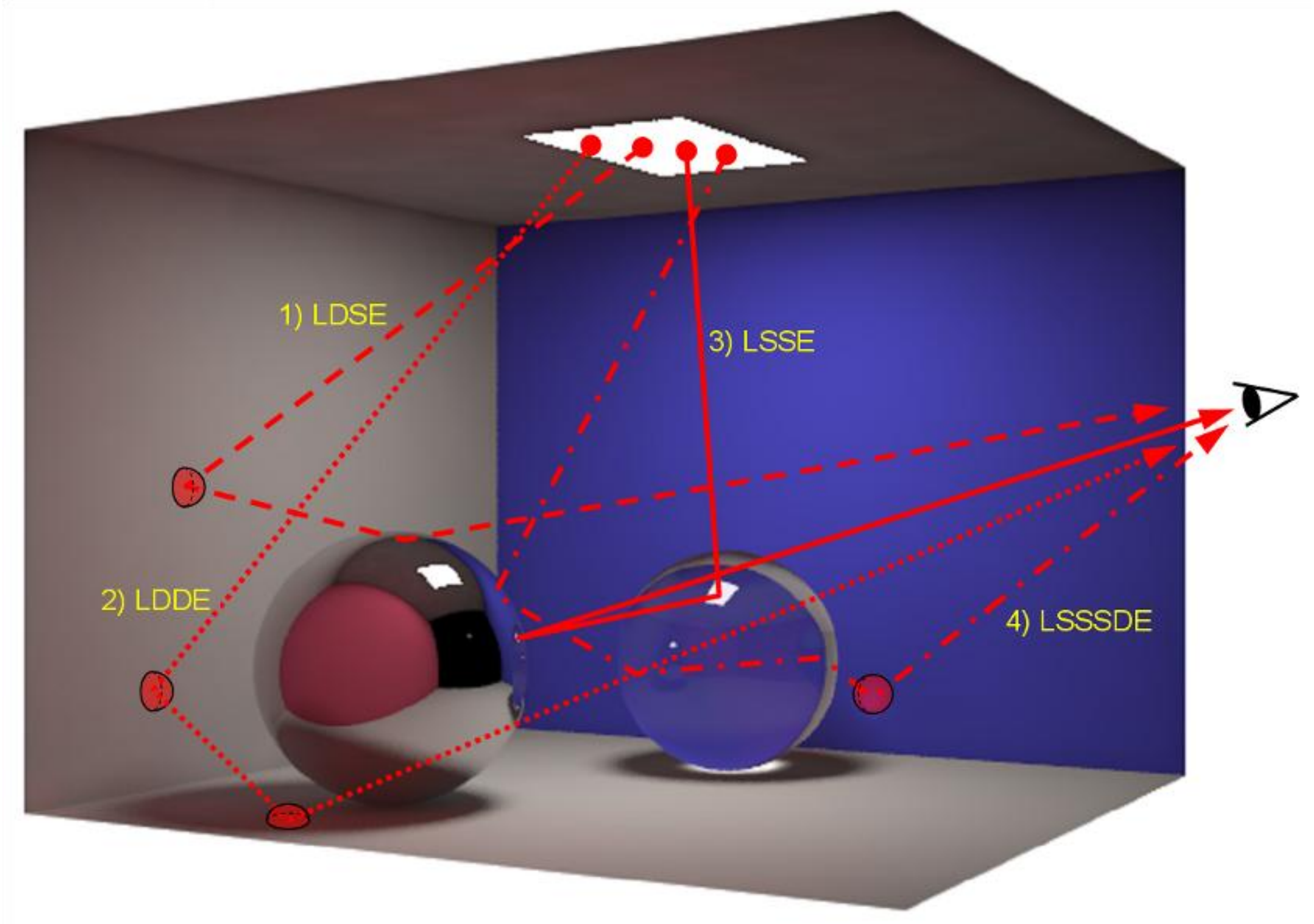


Light Source

Diffuse Reflection

Specular Reflection

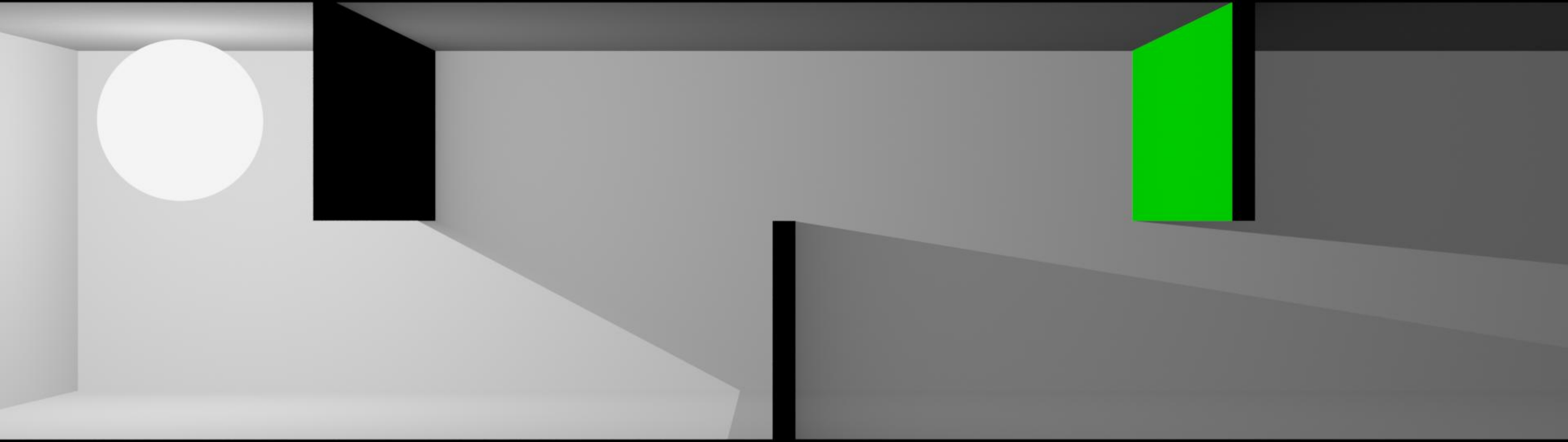
Eye

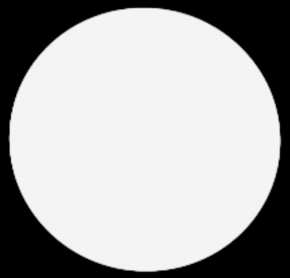


There are many kinds of light paths: **L(D|S)*E**

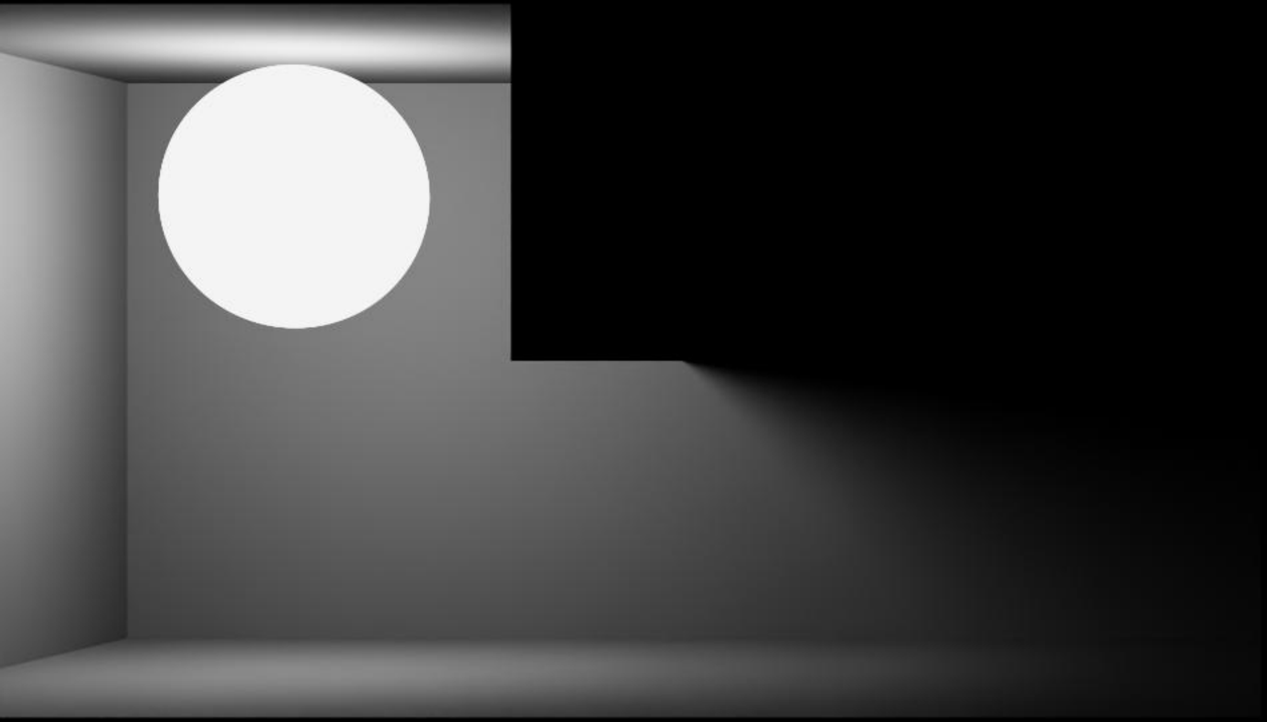
2) Floor illuminated by diffuse reflection of wall

4) Wall illuminated by light passing through sphere

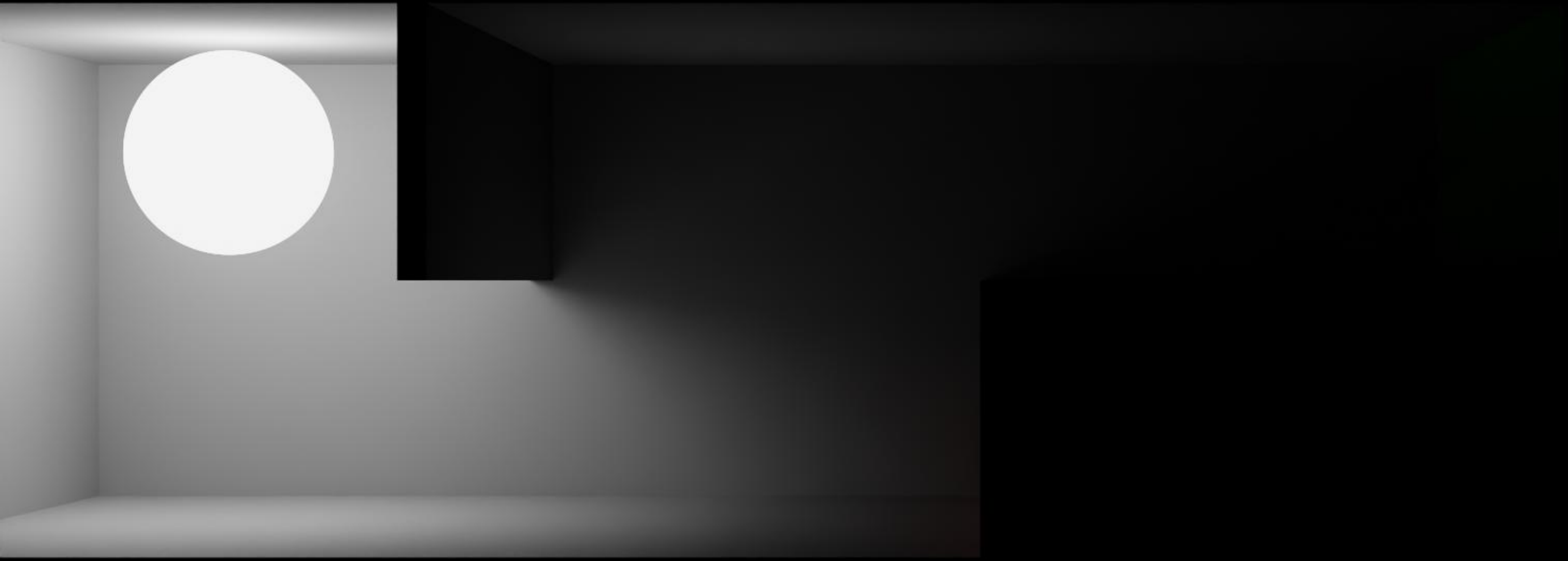




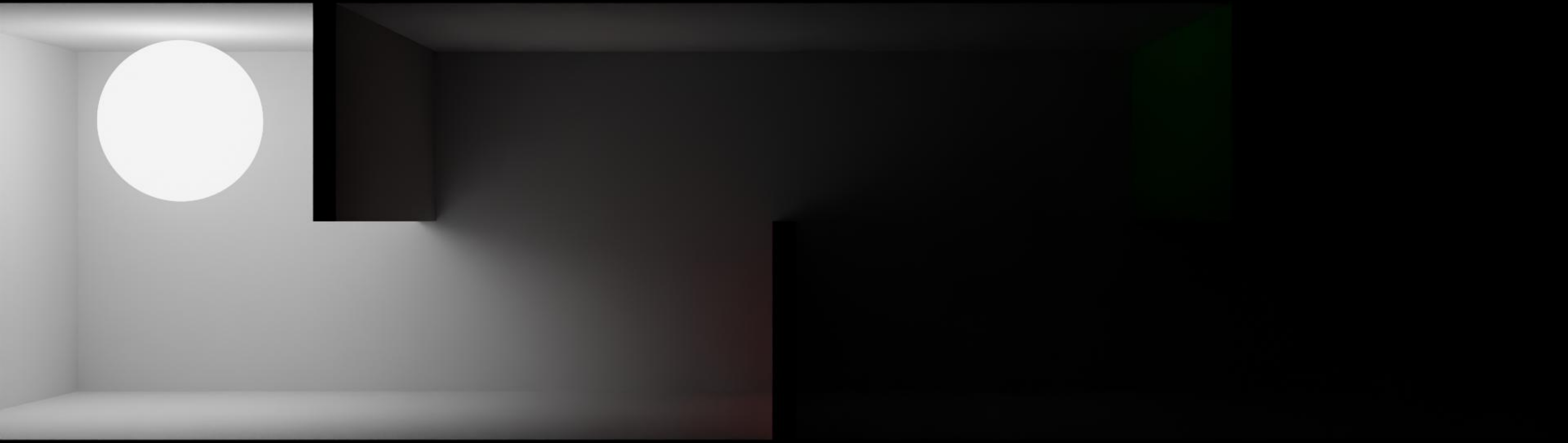
LE



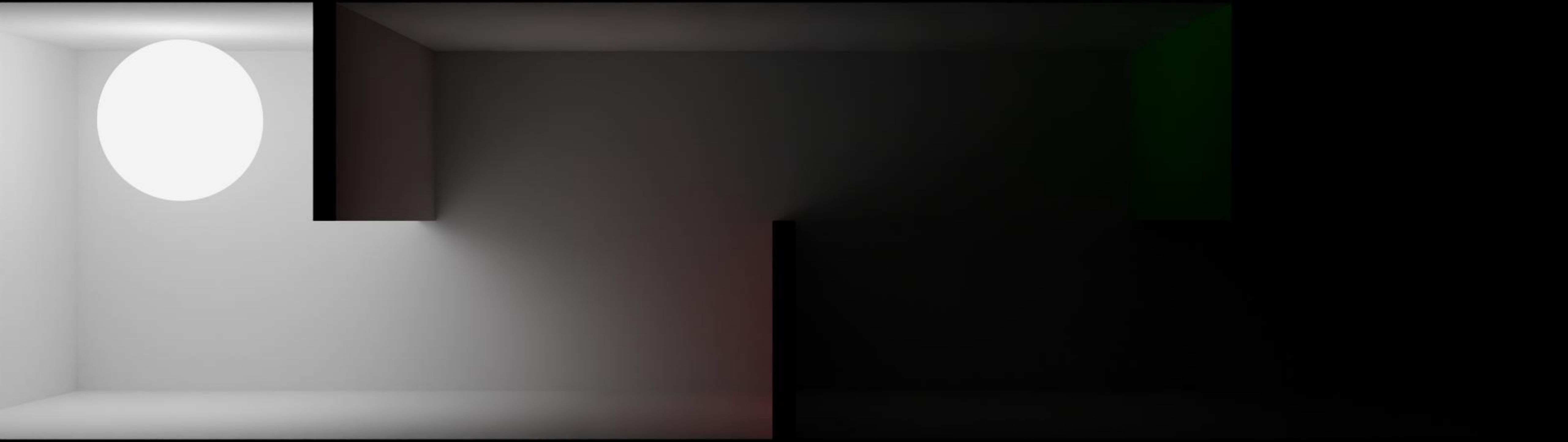
LD?E



LD{,2}E

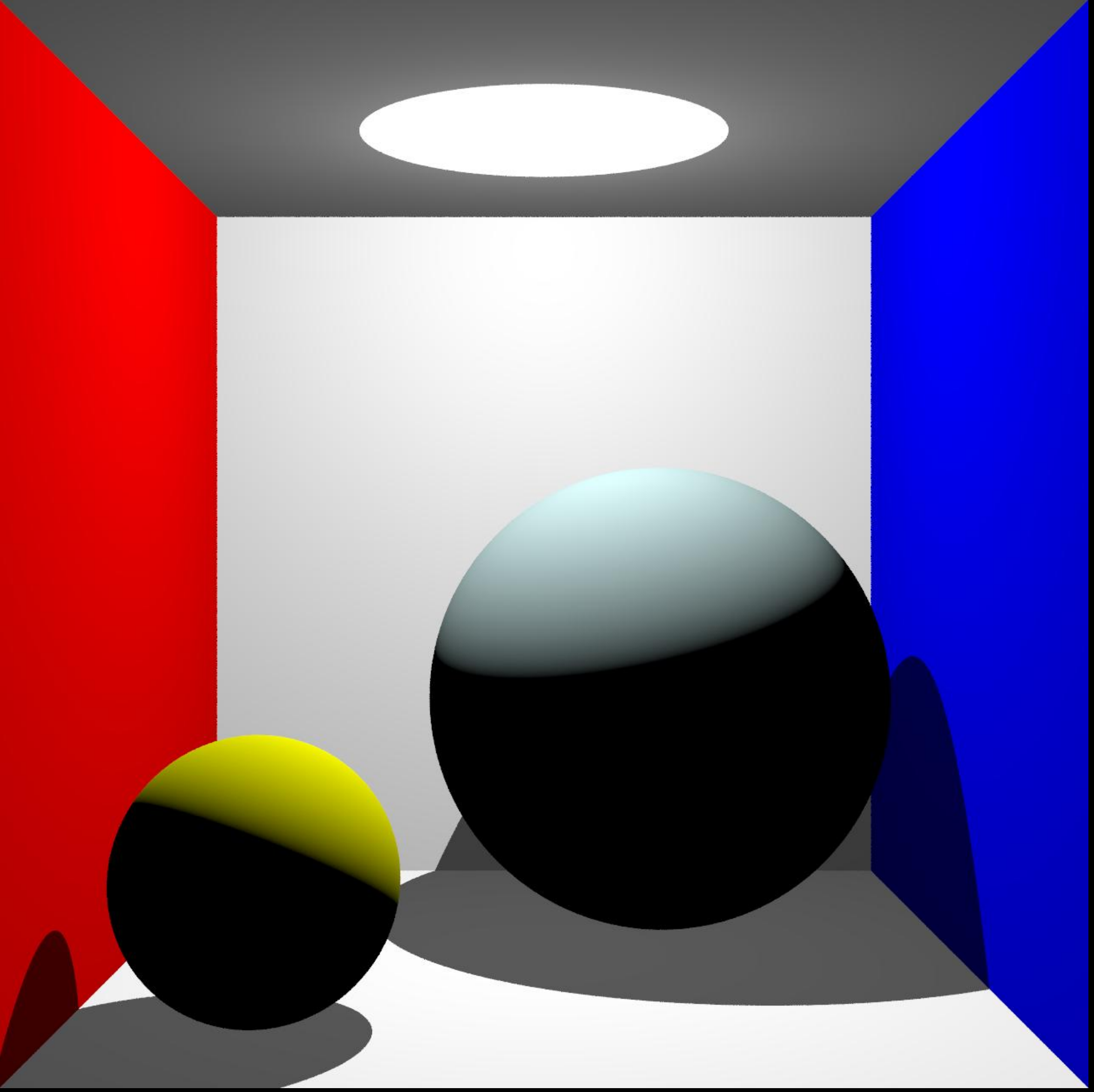


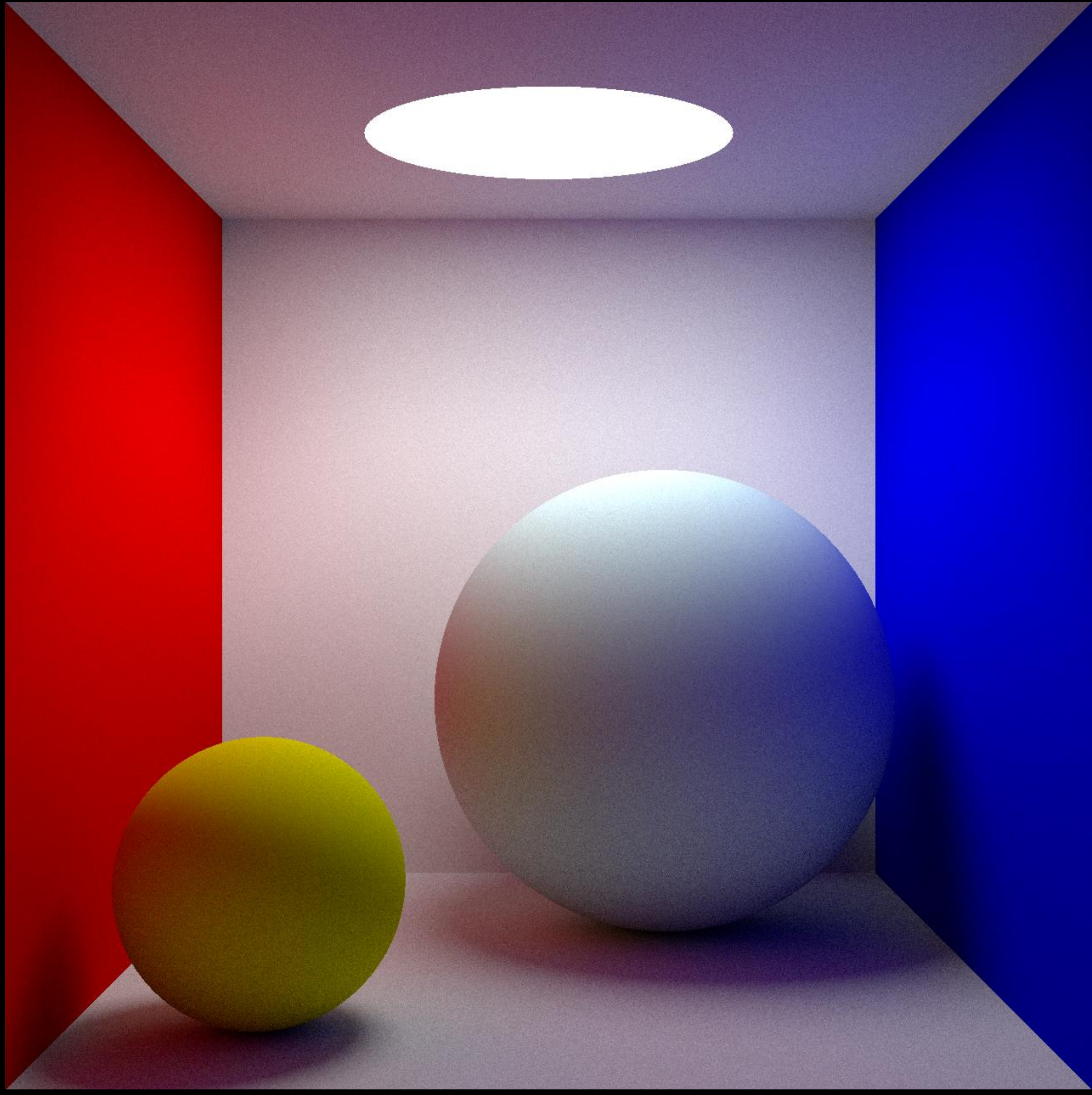
LD{,3}E

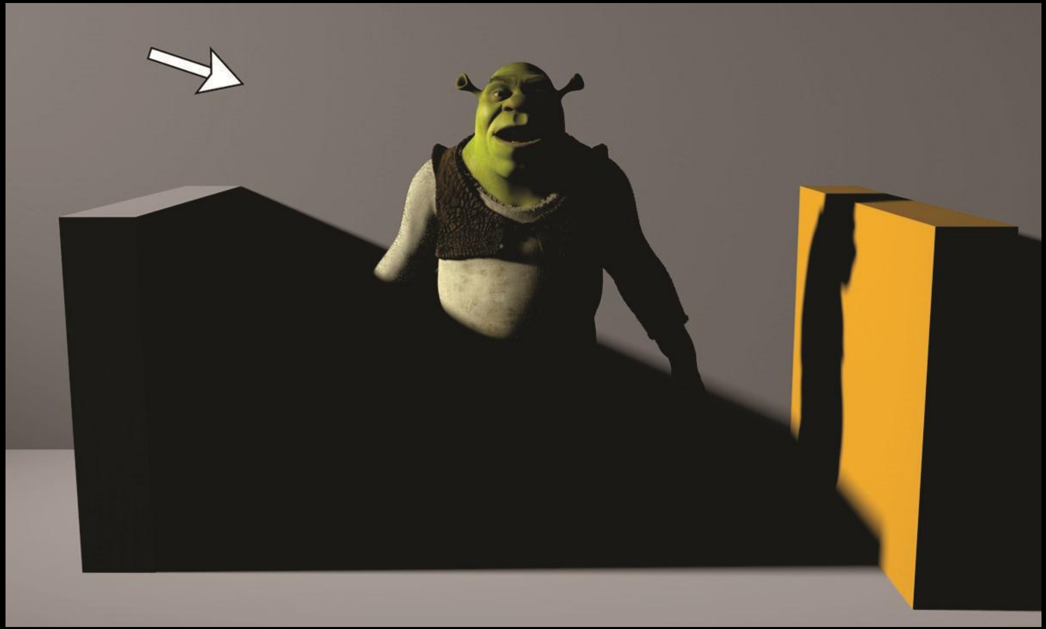


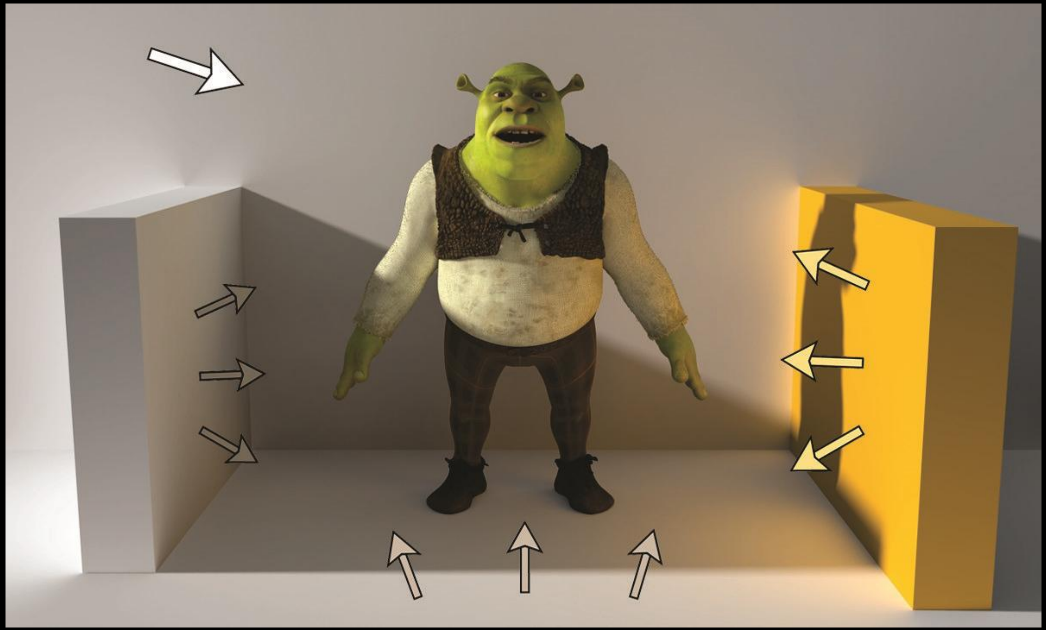
LD{,4}E

(Global Illumination)
LD*E


LD?E


LD*E (Global Illumination)



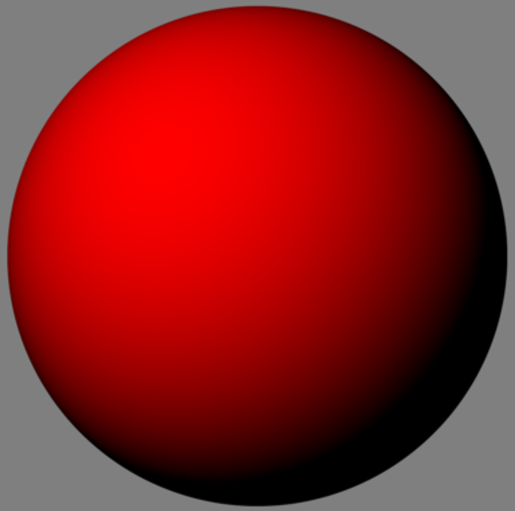


Global Illumination is being used in movies since 2004 (Shrek II)

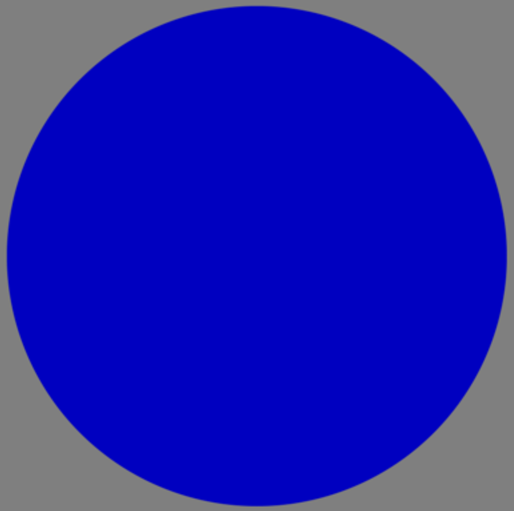
Accelerated with techniques like Importance Sampling, Manifold Exploration, Path Guiding, Irradiance Caching, Spatiotemporal Reservoir Sampling, Point-based GI, Radiosity, Photon Mapping, Screen-Space GI, Path Guiding, Reflective Shadow Maps and many others.

$$\begin{bmatrix}1\\1\\1\end{bmatrix} \qquad \begin{bmatrix}1\\0\\0\end{bmatrix} \qquad \begin{bmatrix}0\\0\\0.2\end{bmatrix} \qquad \begin{bmatrix}1\\0\\0.2\end{bmatrix}$$

$Light \odot$



$\odot \dots +$



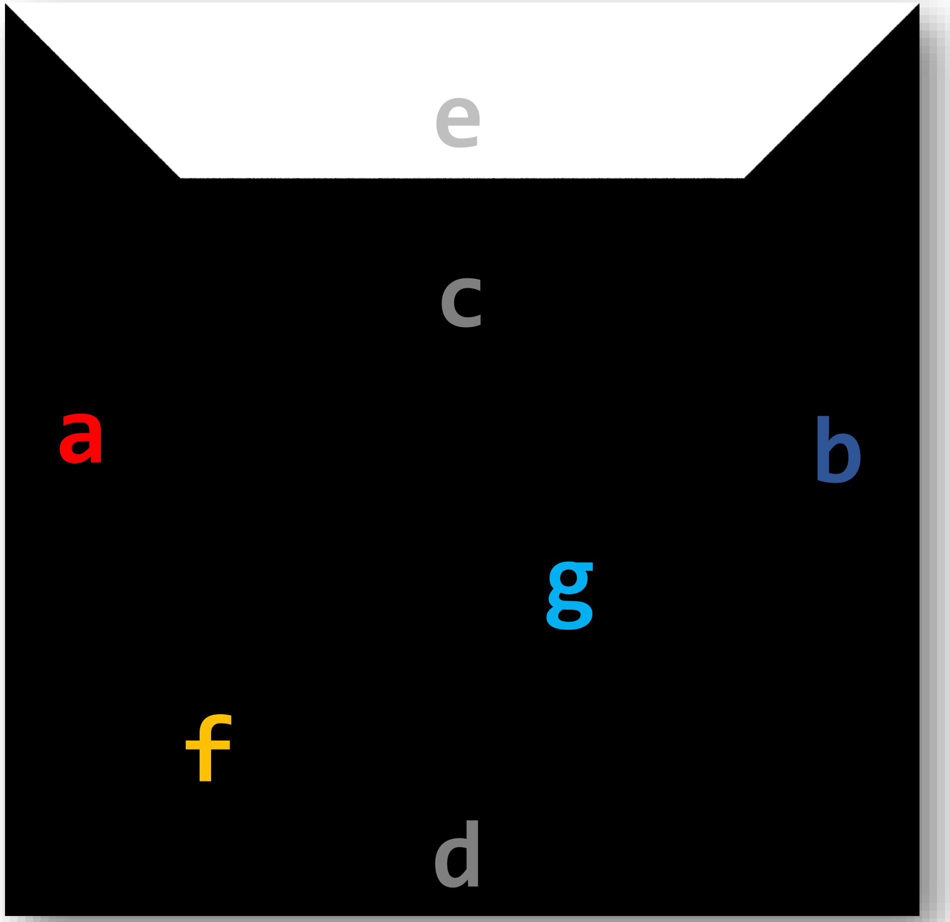
$=$

"Diffuse reflectance"
or "Lambert reflectance"
or "material color"
or "Albedo"

"Emission"
or "Luminance"

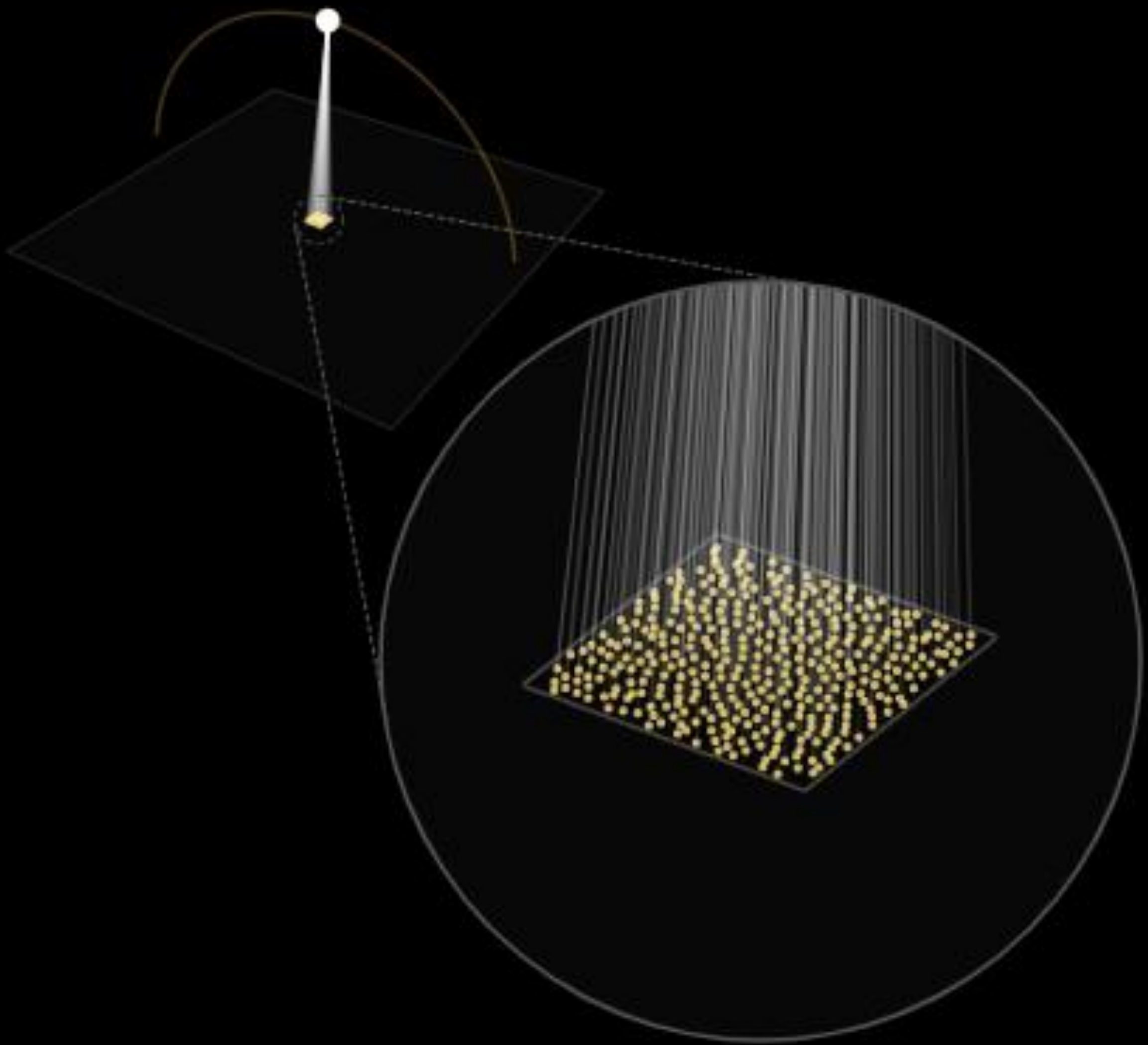
Combined result

Introducing Light Sources

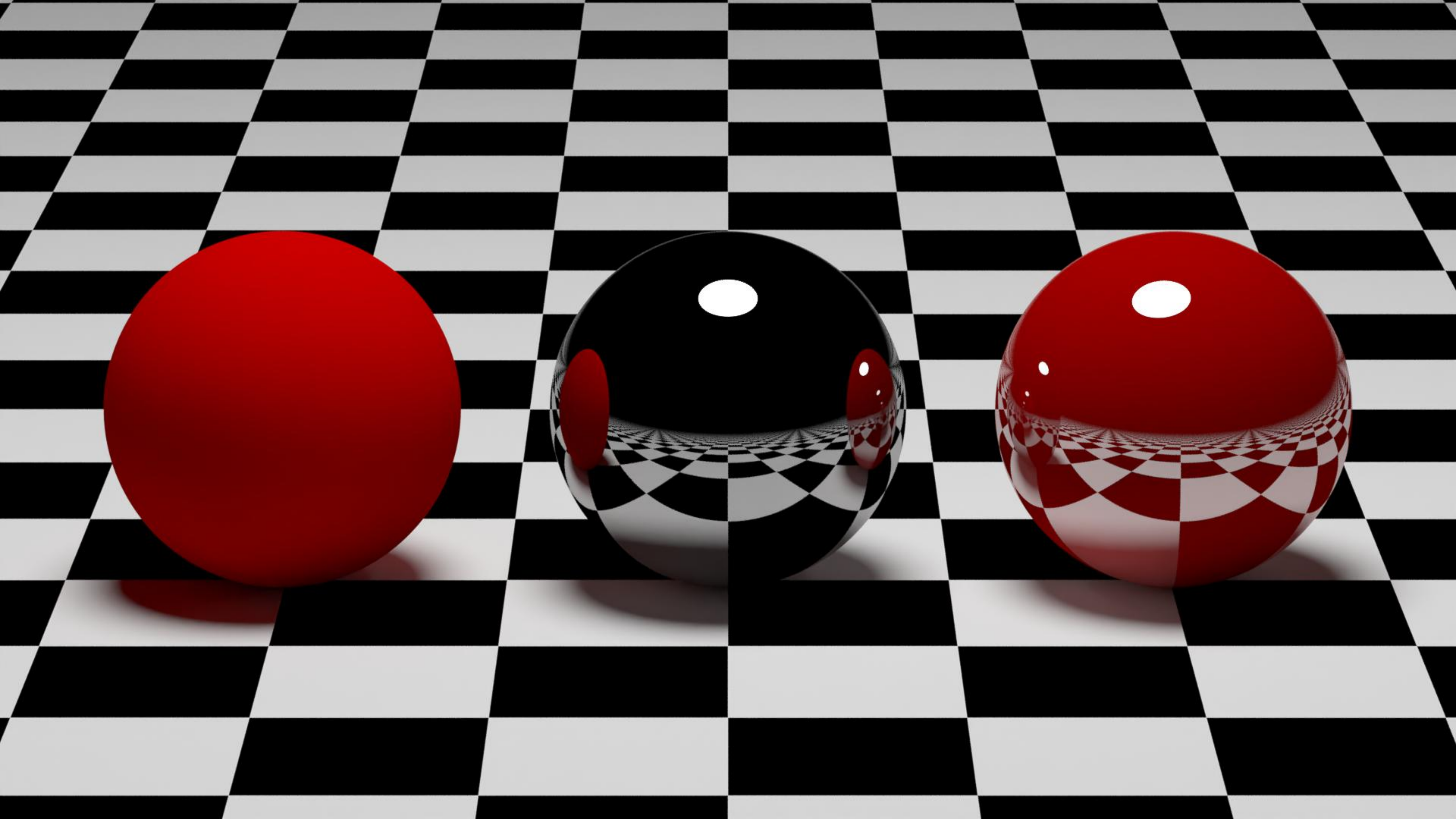



```
Scene
{
  Sphere(…, Diffuse=      Red, Emission=      Black) a
  Sphere(…, Diffuse=     Blue, Emission=      Black) b
  Sphere(…, Diffuse=     Gray, Emission=      Black) c
  Sphere(…, Diffuse=     Gray, Emission=      Black) d
  Sphere(…, Diffuse=    White, Emission= 2 * White) e
  Sphere(…, Diffuse=   Yellow, Emission=      Black) f
  Sphere(…, Diffuse=LightCyan, Emission=      Black) g
}
```

Why $\overrightarrow{\omega_i} \cdot \vec{n}$?



Animation: Bartosz Ciechanowski (https://ciechanow.ski/lights-and-shadows/)

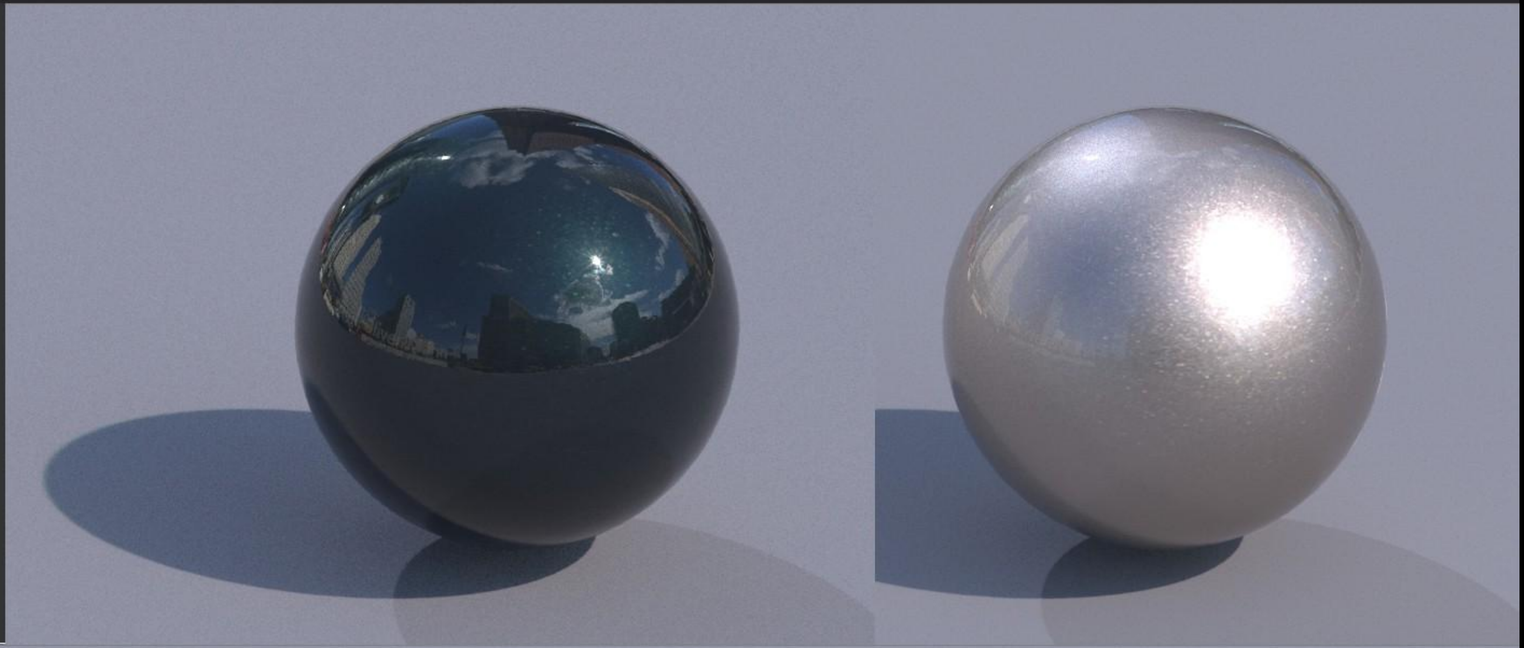


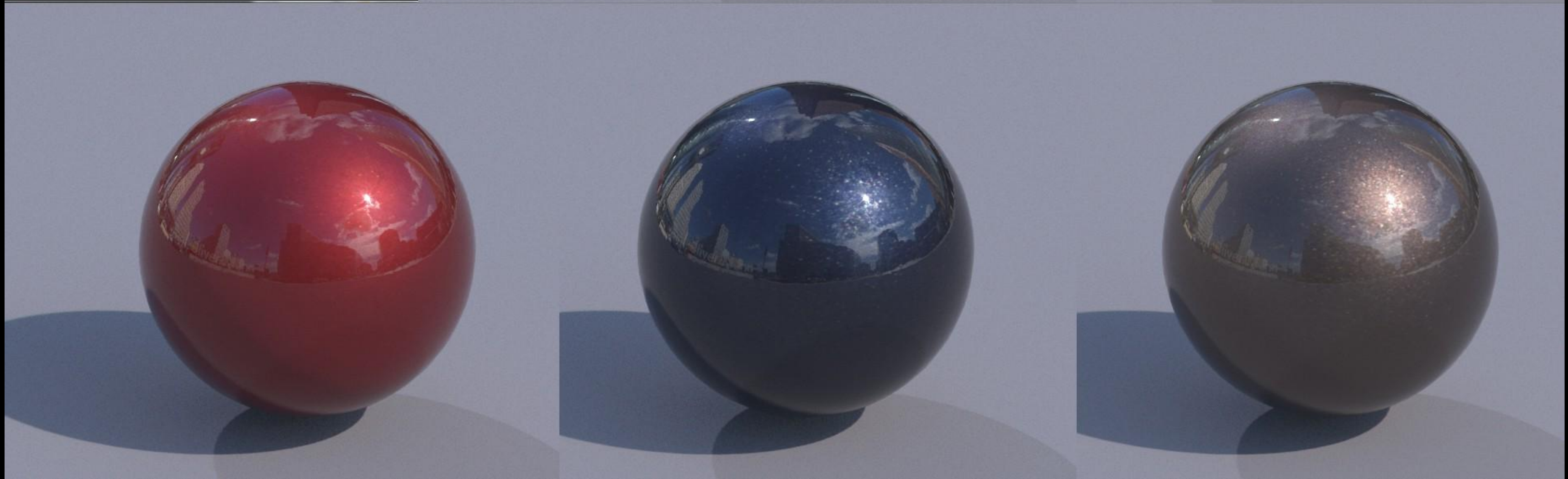
teamup

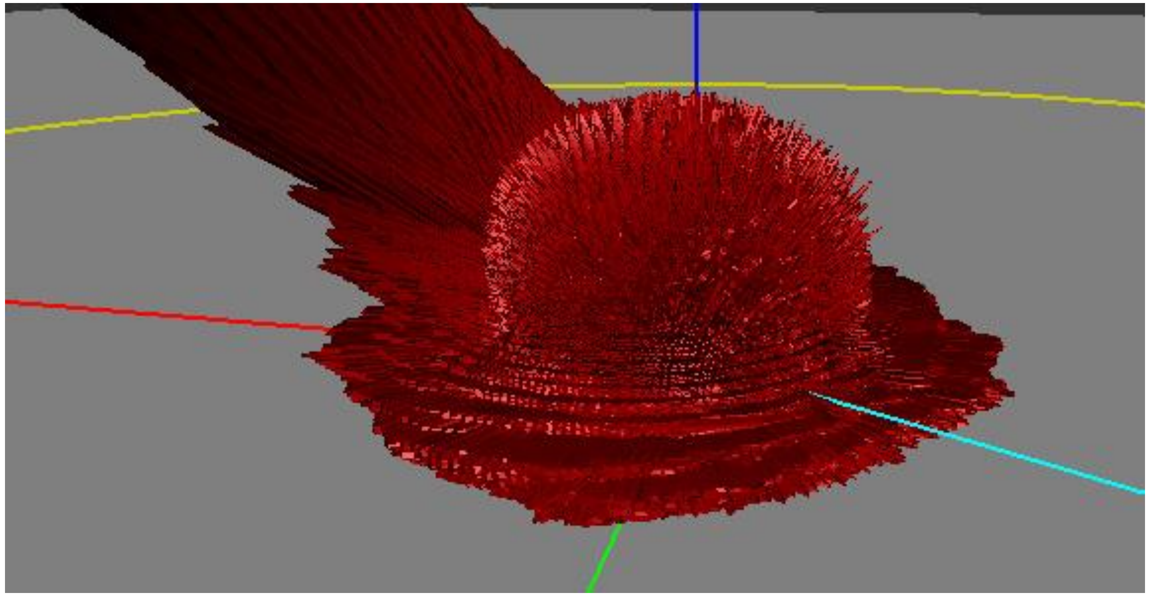
Rendered using Multi-Optics

Measured Carpaints.

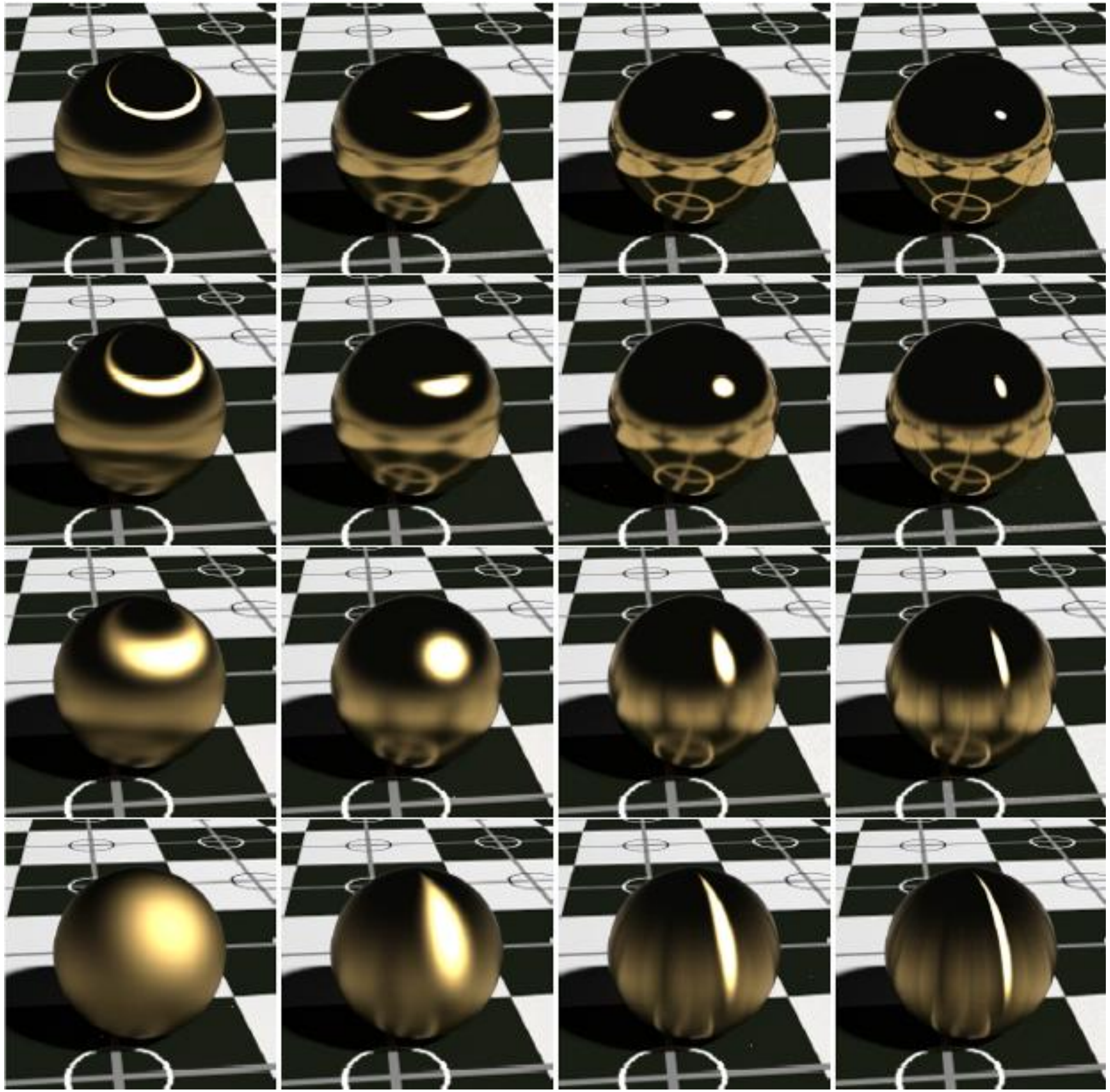
Editable Flakes and Colors.







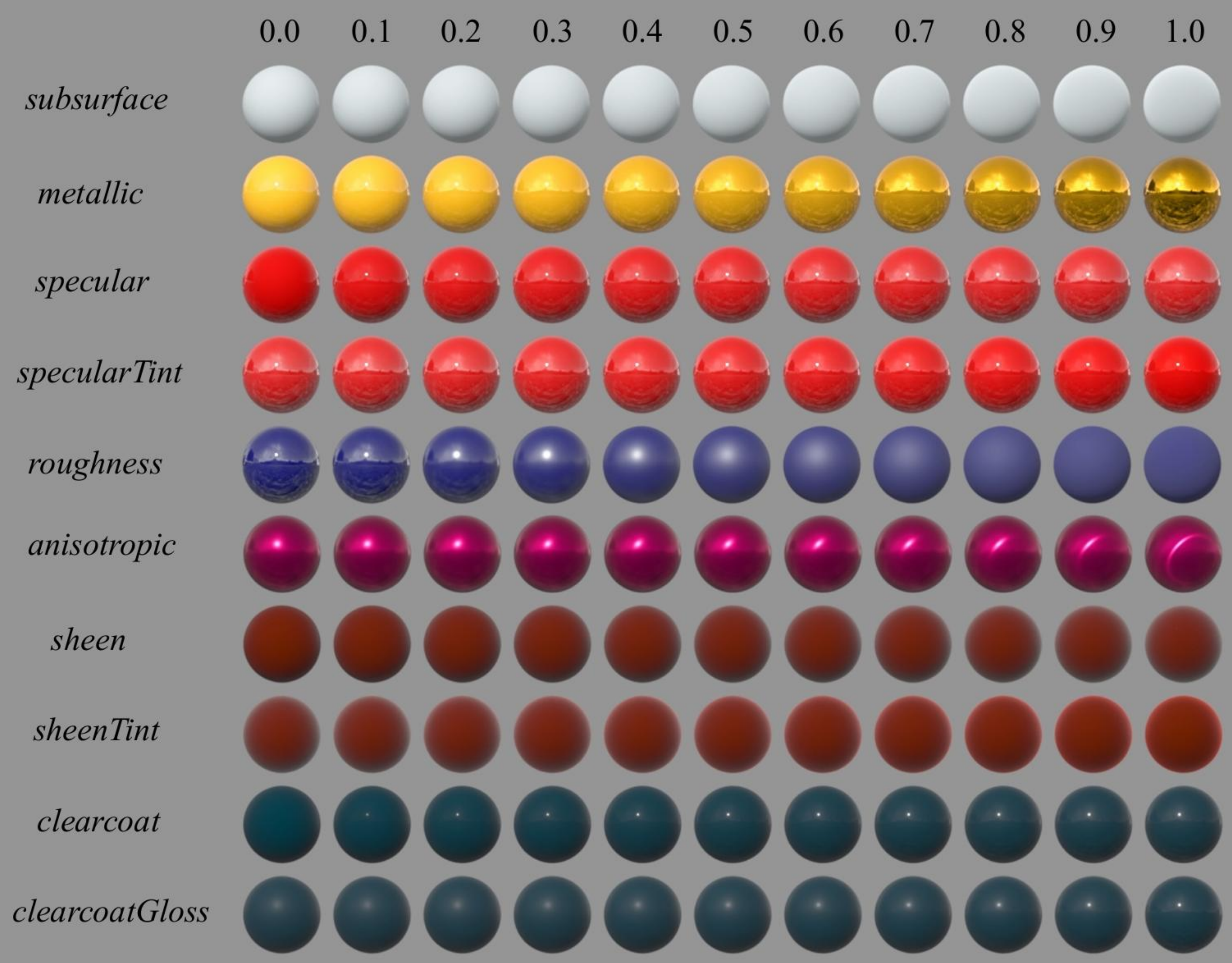
Nylon BRDF, Disney Research



An Anisotropic Phong BRDF Model, Ashikhmin et al.

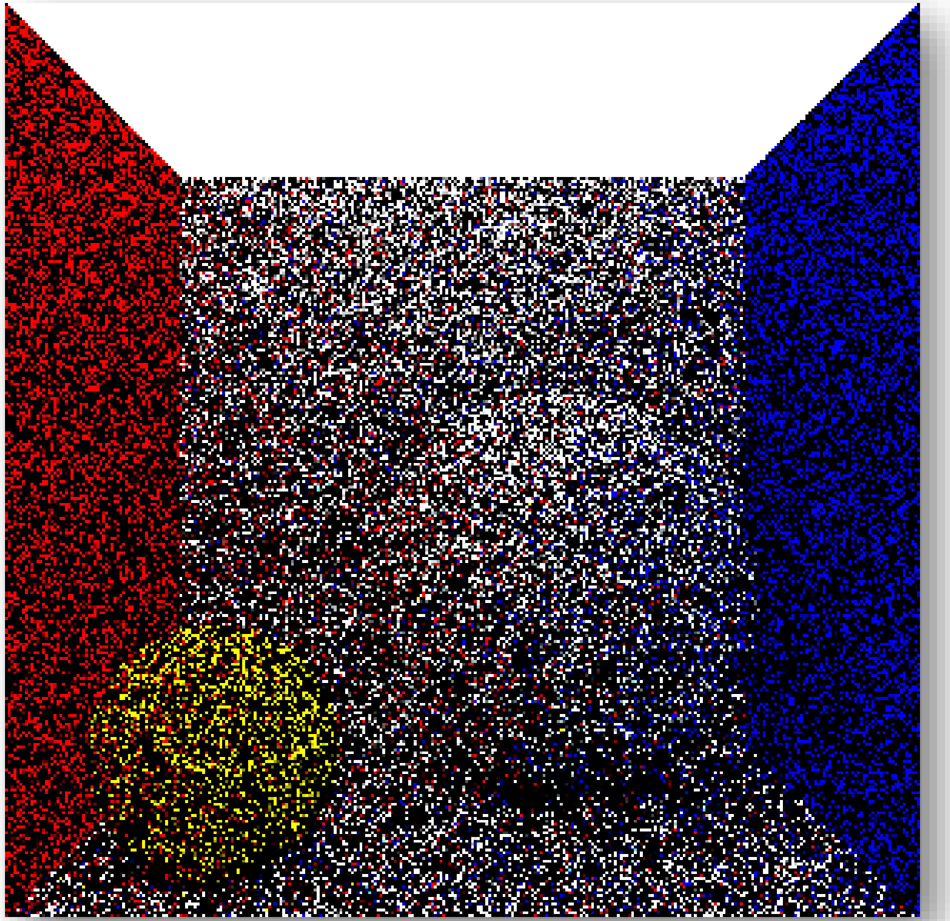
BRDFs describe **arbitrary** materials.

BRDFs may depend on more parameters than just ω_i and ω_o: Location, Wavelengths, Time, …



The [Disney BRDF](#) (Burley et al.) describes every material with 10 parameters.

Monte Carlo Path Tracing

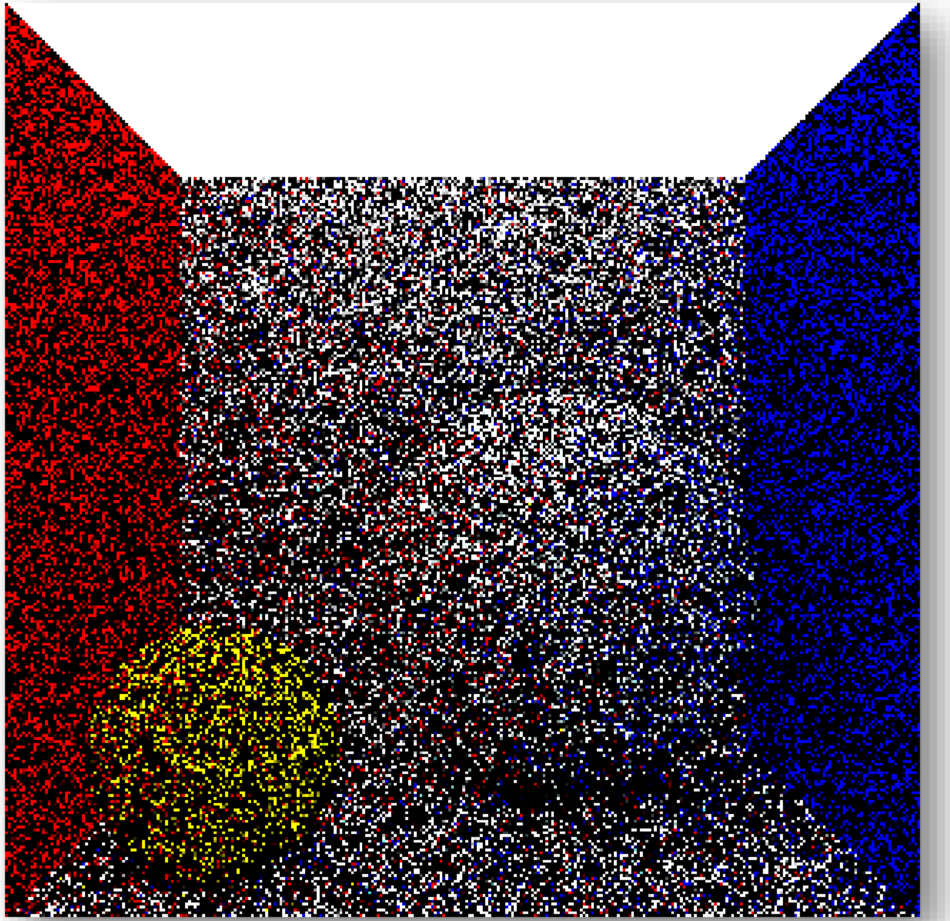


```
Vec3 ComputeColor(Vec3 O, Vec3 d⃗)
{
    hp ≔ Closest hitpoint of ray(O, d⃗) in scene



    Sample random direction ω_r ∈ Ω

    return hp_emission + 2π (ω_r · n) BRDF(d, ω_r) ⊙ ComputeColor(hp, ω_r)
}
```

$$hp_{emission} + 2\pi\left(\widehat{\vec{\omega_r}}\cdot\widehat{\vec{n}}\right)\mathrm{BRDF}(\widehat{\vec{d}},\widehat{\vec{\omega}}_r)\odot \mathrm{ComputeColor}(hp,\vec{\omega_r})$$

```
Vec3 BRDF(ω_I, ω_O)
{
    return Diffuse Color of hit object · 1/π
}
```

Monte Carlo Path Tracing



```
Vec3 ComputeColor(Vec3 O, Vec3 d)
{
    hp := Closest hitpoint of ray(O,d) in scene
    if x~U[0,1] < p
        return hp_emission

    Sample random direction ω_r ∈ Ω
    return hp_emission + 2π/(1−p) (ω_r · n) BRDF(d, ω_r) ⊙ ComputeColor(hp, ω_r)
}

Vec3 BRDF(ω_I, ω_O)
{
    return Diffuse Color of hit object · 1/π
}
```

$$\texttt{Vec3 ComputeColor(Vec3 O, Vec3 } \vec{d}\texttt{)}$$

$$hp \coloneqq \text{Closest hitpoint of } ray(O,\vec{d}) \text{ in scene}$$

$$\text{if } x \sim \mathcal{U}_{[0,1]} < p \quad \text{return } hp_{emission}$$

$$\text{Sample random direction } \widehat{\overrightarrow{\omega_r}} \in \Omega$$

$$\text{return } hp_{emission} + \frac{2\pi}{1-p}\left(\widehat{\overrightarrow{\omega_r}} \cdot \widehat{\vec{n}}\right) \mathrm{BRDF}(\widehat{\vec{d}}, \widehat{\vec{\omega}}_r) \odot \mathrm{ComputeColor}(hp, \overrightarrow{\omega_r})$$

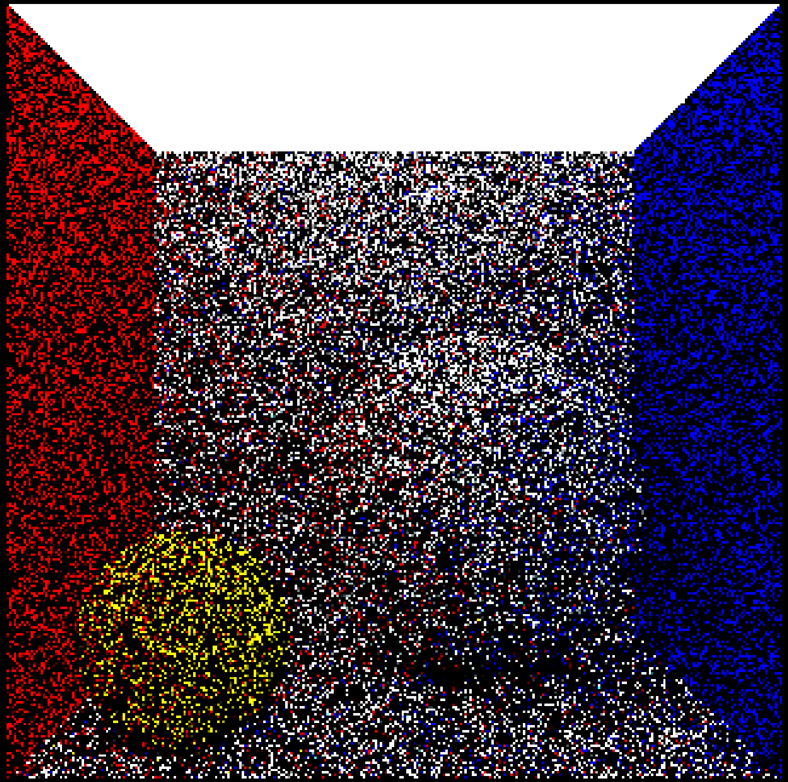
$$\texttt{Vec3 BRDF}(\widehat{\vec{\omega}}_I, \widehat{\vec{\omega}}_O)$$

$$\text{return Diffuse Color of hit object} \cdot \frac{1}{\pi}$$

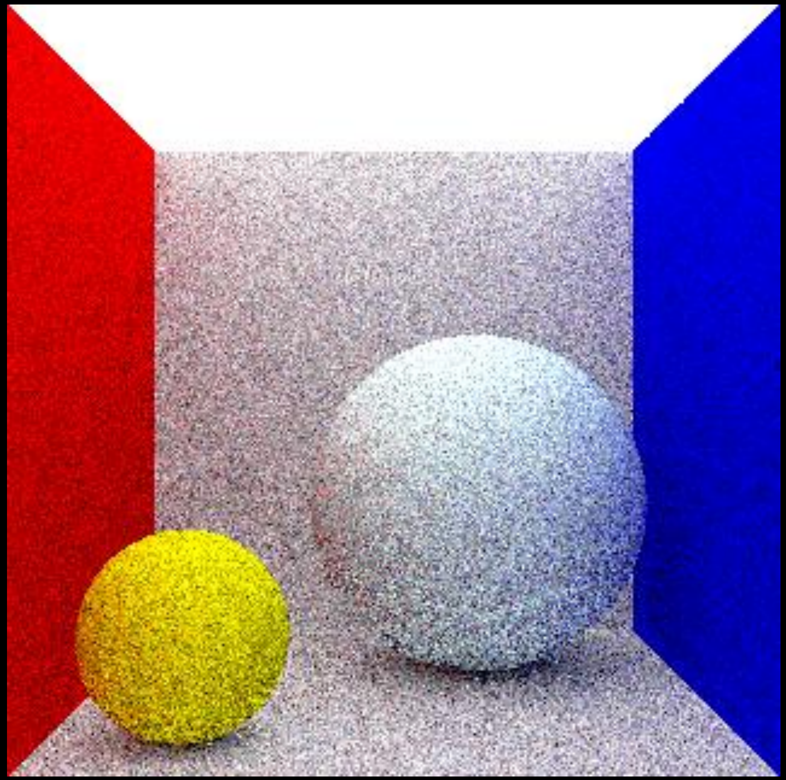
e.g. $0.05 < p < 0.5$

Monte Carlo Path Tracing

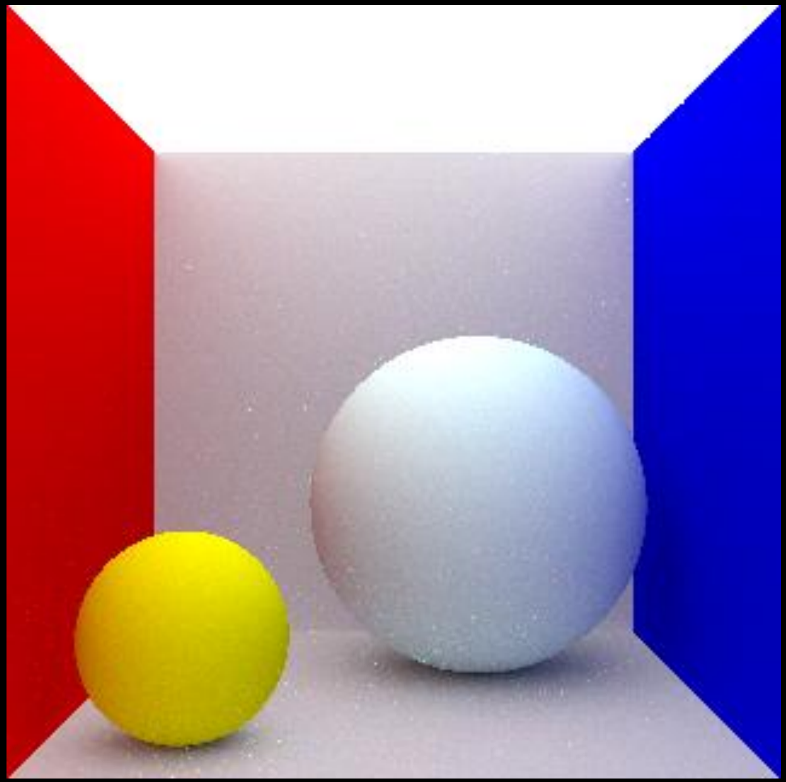
1 Ray/Pixel

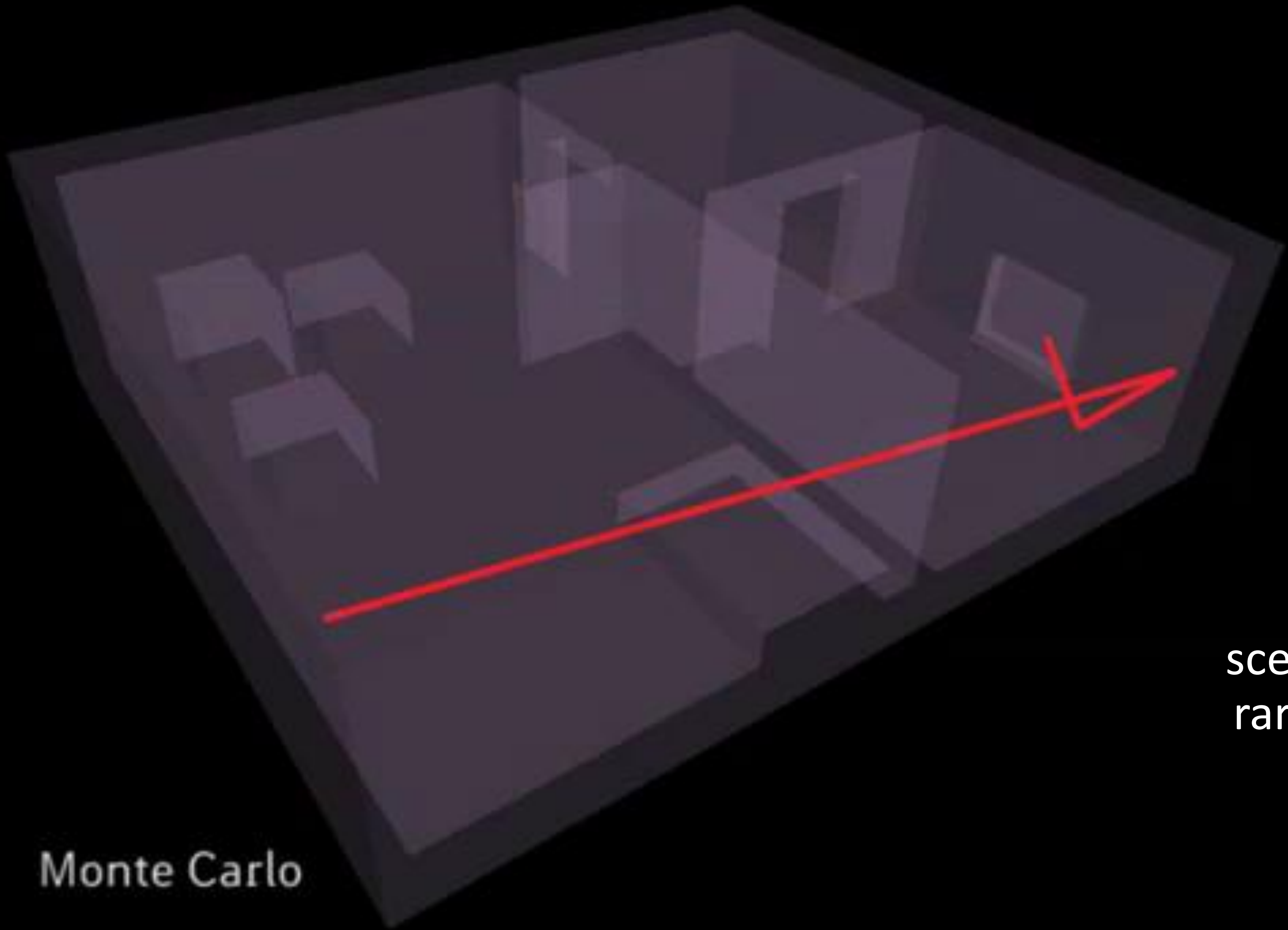


Averaging 32 Rays/Pixel



Averaging 4096 Rays/Pixel

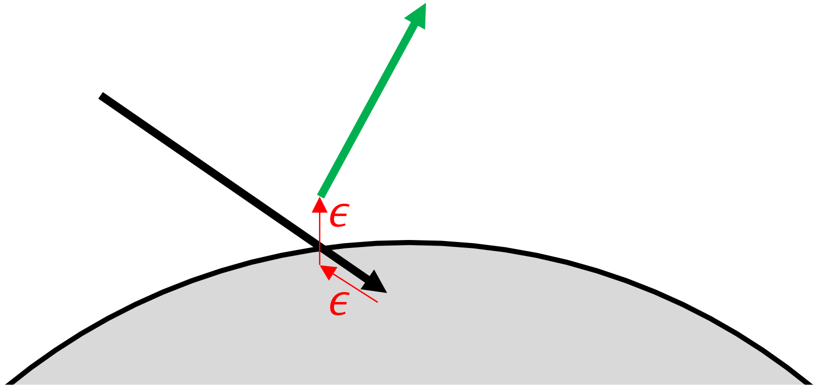




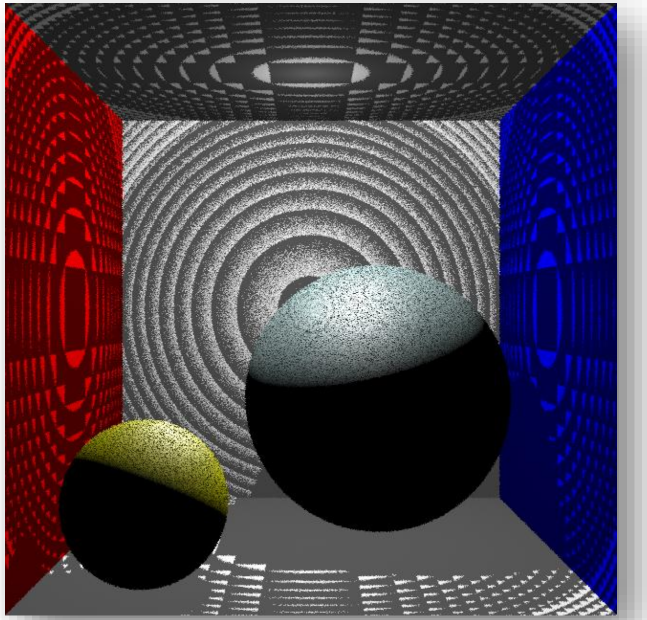
In challenging scenes, camera rays rarely find a way to a light source.

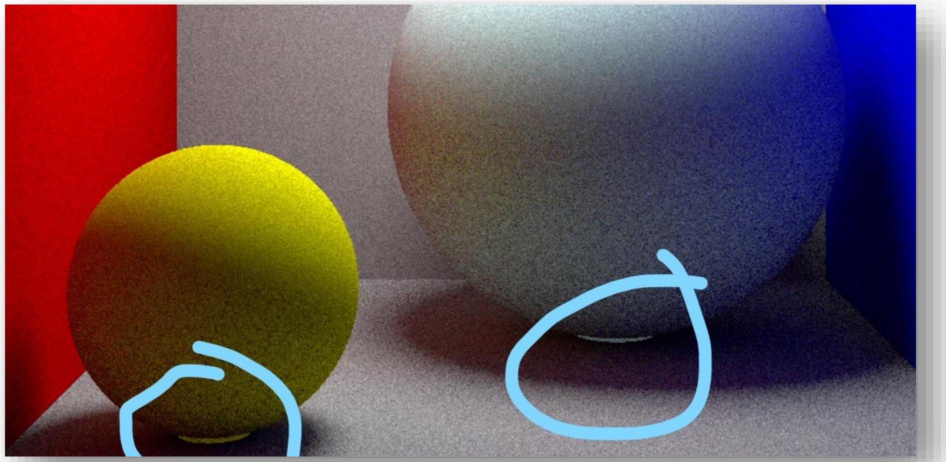
Working with Floating Point

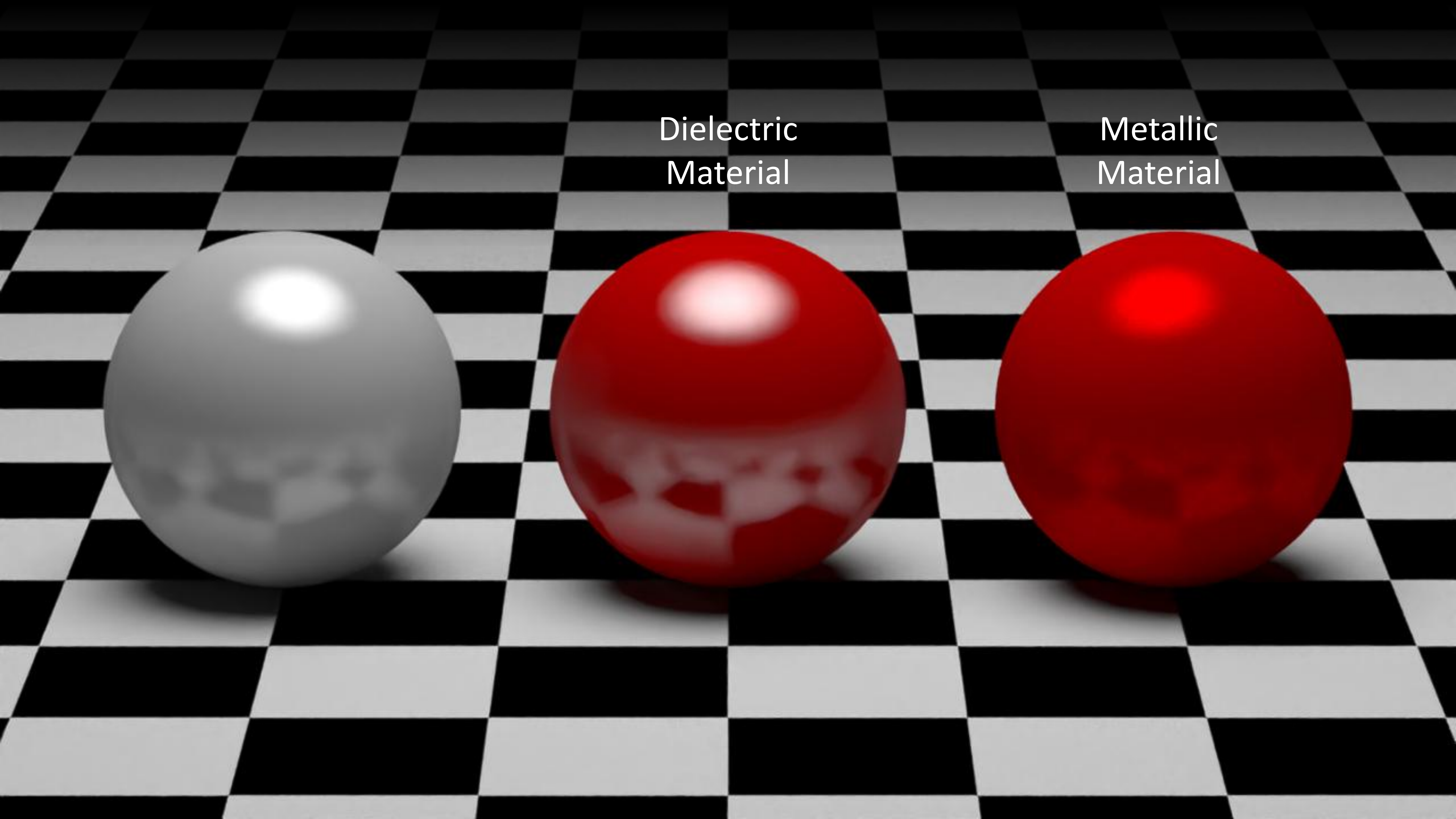
Limited accuracy means that intersections end up **inside** the object in 50% of the cases.



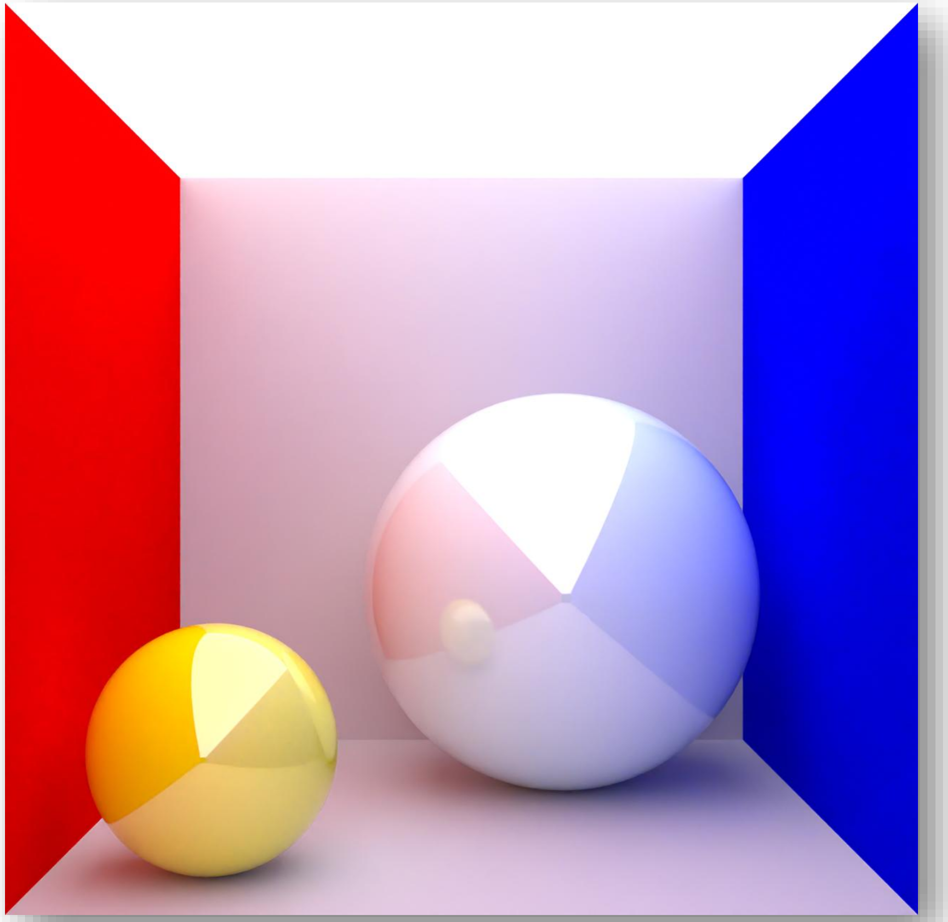
Start reflected rays **outside** the object, by *nudging* the starting point backwards and/or outwards.






Dielectric
Material
Metallic
Material

Reflective Materials



```
Color BRDF(…)
{
    d_r = Compute reflection of d on n
    if ω_r = d_r
        return Diffuse Color · 1/π + μ(Specular Color of object)
    else
        return Diffuse Color of object · 1/π
}
```

$\hat{\vec{d}}_r$ = Compute reflection of $\hat{\vec{d}}$ on $\hat{\vec{n}}$

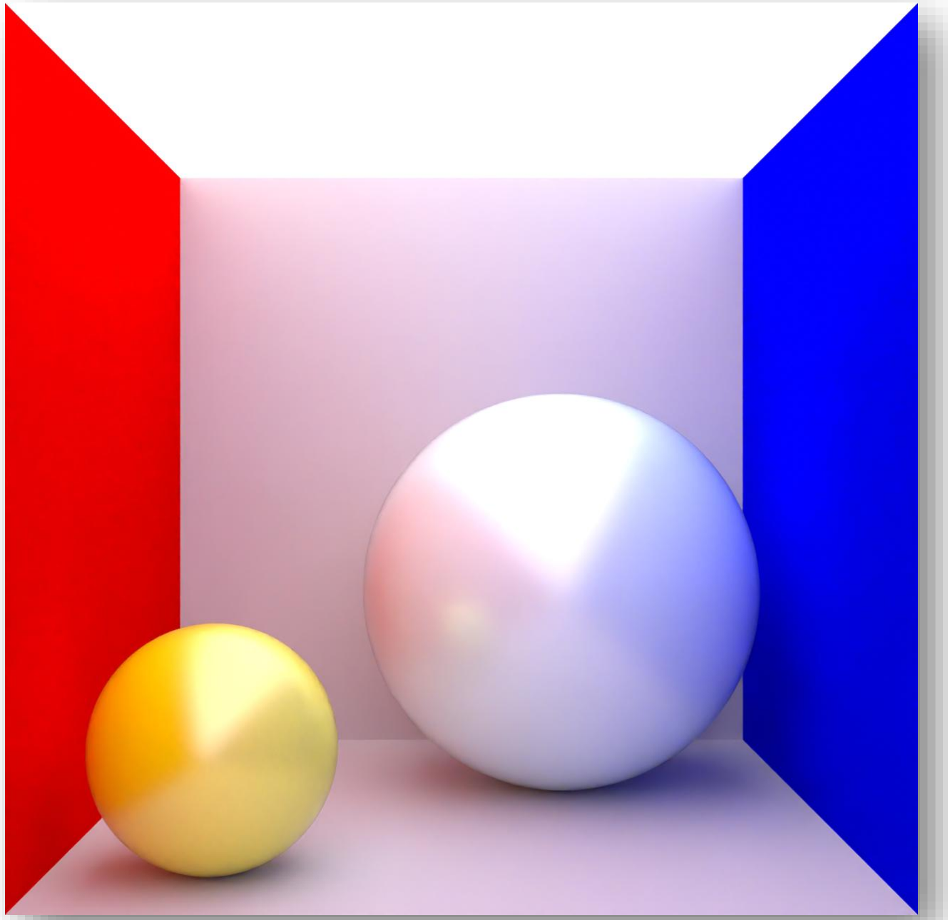
if $\widehat{\vec{\omega}_r} = \hat{\vec{d}}_r$

return Diffuse Color $\cdot \frac{1}{\pi}$ + μ(Specular Color of object)

else

return Diffuse Color of object $\cdot \frac{1}{\pi}$

Reflective Materials



```
Color BRDF(…)
{
    d_r = Compute reflection of d on n
    if ω_r · d_r > 1 − ε
        return Diffuse Color · 1/π + μ(Specular Color of object)
    else
        return Diffuse Color of object · 1/π
}
```

Where: $\widehat{\vec{d}}_r$ = Compute reflection of $\widehat{\vec{d}}$ on $\widehat{\vec{n}}$; if $\widehat{\vec{\omega}_r} \cdot \widehat{\vec{d}}_r > 1 - \epsilon$ return Diffuse Color $\cdot \frac{1}{\pi}$ + μ(Specular Color of object) else return Diffuse Color of object $\cdot \frac{1}{\pi}$

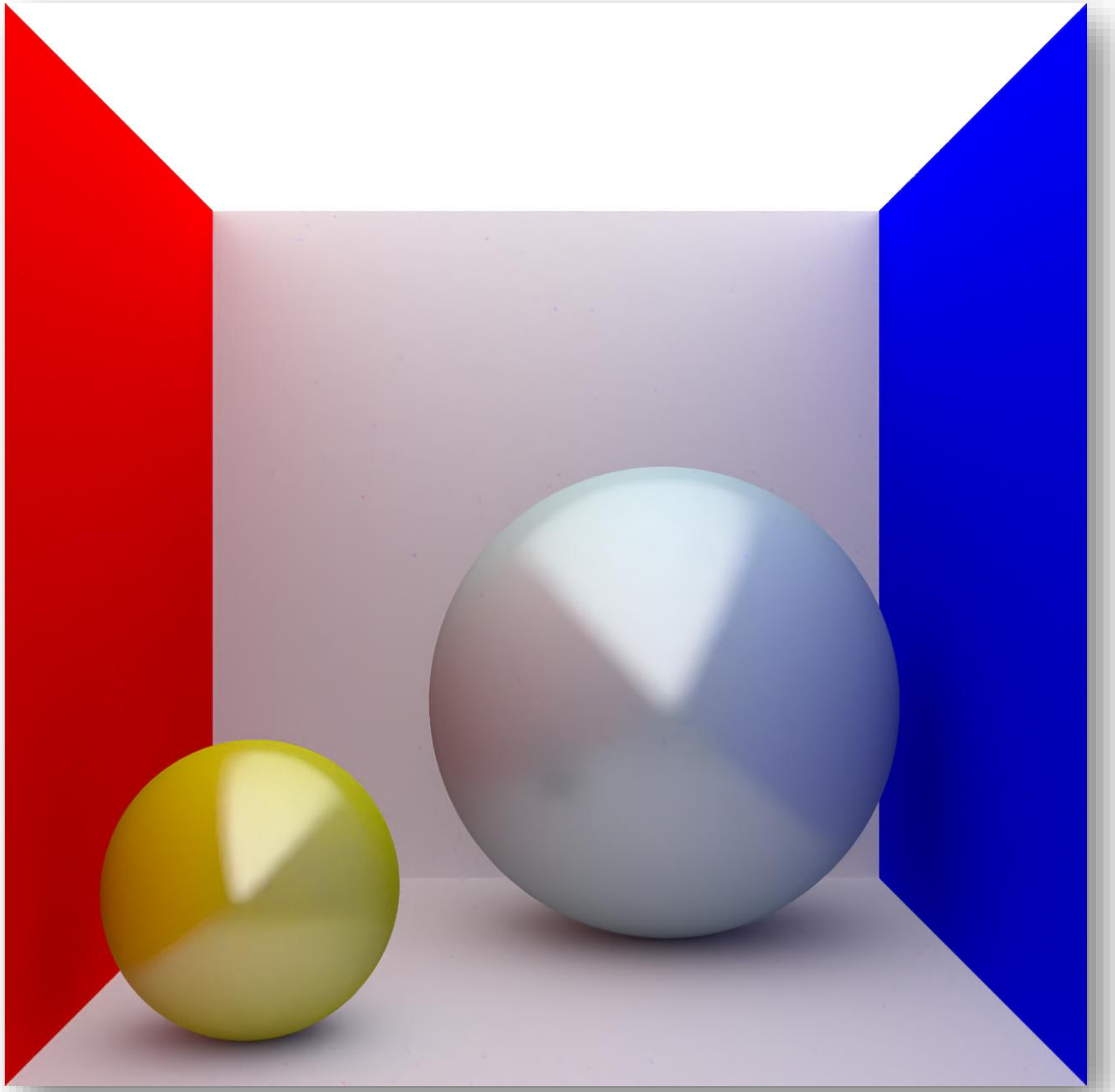
e.g. $\epsilon = 0.01, \mu = 10$

This Week's Lab

Extend the path tracer with:

- **Emissive** materials
- **Multiple rays** per pixel
- **Reflections** on a subset of objects

1. Comment out the code from last week
2. Take screenshots of the results



The first 11:11 of https://www.youtube.com/watch?v=gsZiJeaMO48
summarizes and visualizes a lot of today's lecture