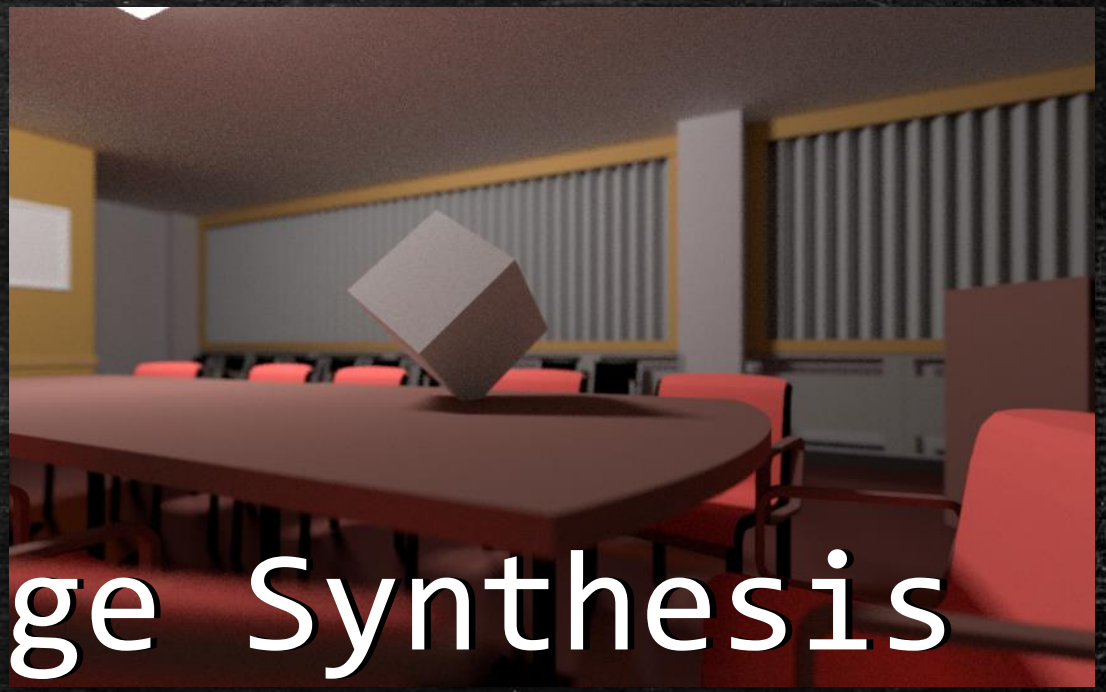
Realistic Image Synthesis



Dr. J. Bikker

lecture 5

Today:

- Improved Sampling
- Path Tracing Recap
- Explicit light sampling
- Importance Sampling
- Resampled Importance Sampling
- Multiple Importance Sampling

Path Tracing & Sampling

Basic idea:

A 100 watt light bulb emits ~10^{20} photons per second.
Simulating these, and recording which ones hit the camera film, yields a correct image.

Optimization 1:
We still get a correct image when we reverse photon paths.
(Now we only need those paths that actually connect the camera to a light source)

Optimization 2:
10^{20} is overkill, we get away with far less (luckily).

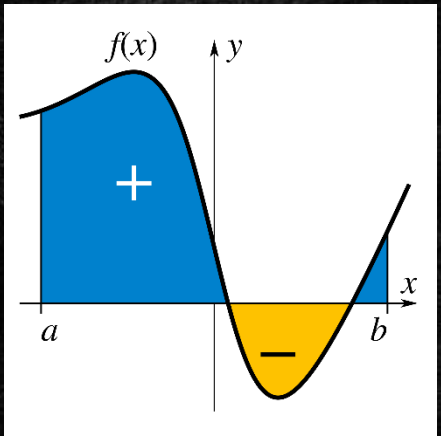
Optimization 3:
Better sampling.

Improving Sampling

Sampling:

In statistics:

"selection of a subset of individuals from within a statistical population to estimate characteristics of the whole population." (Wikipedia)

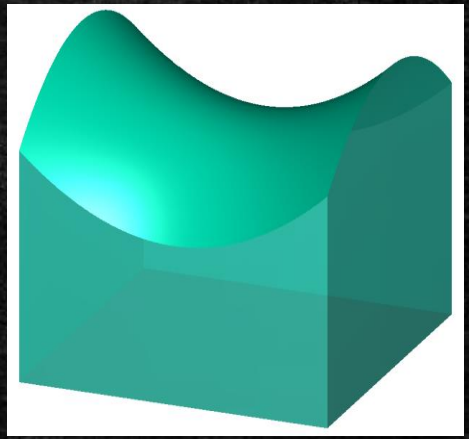


In graphics:

Evaluating a function f(x) for a uniform random x.

Or, in case the integral is multi-dimensional:

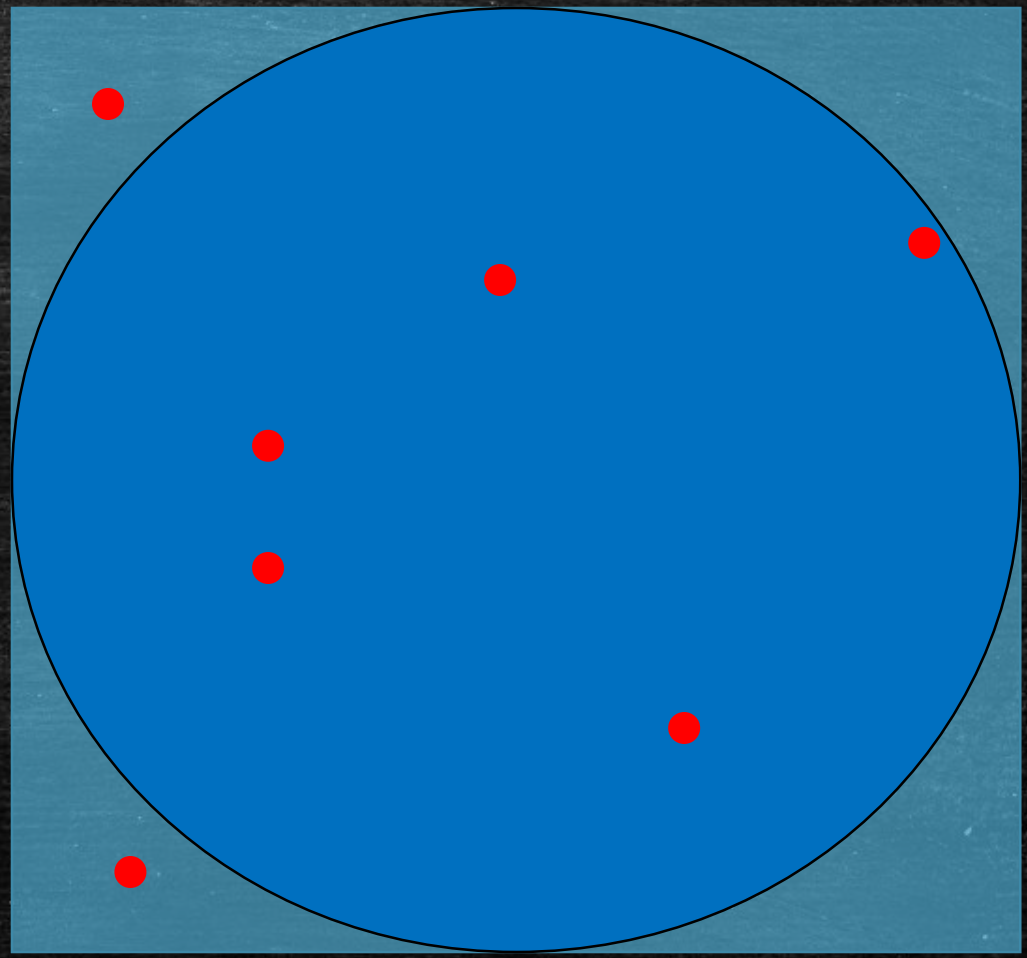
Evaluating a function for a set of random parameters.



Improving Sampling

$\pi \approx 3.141593$

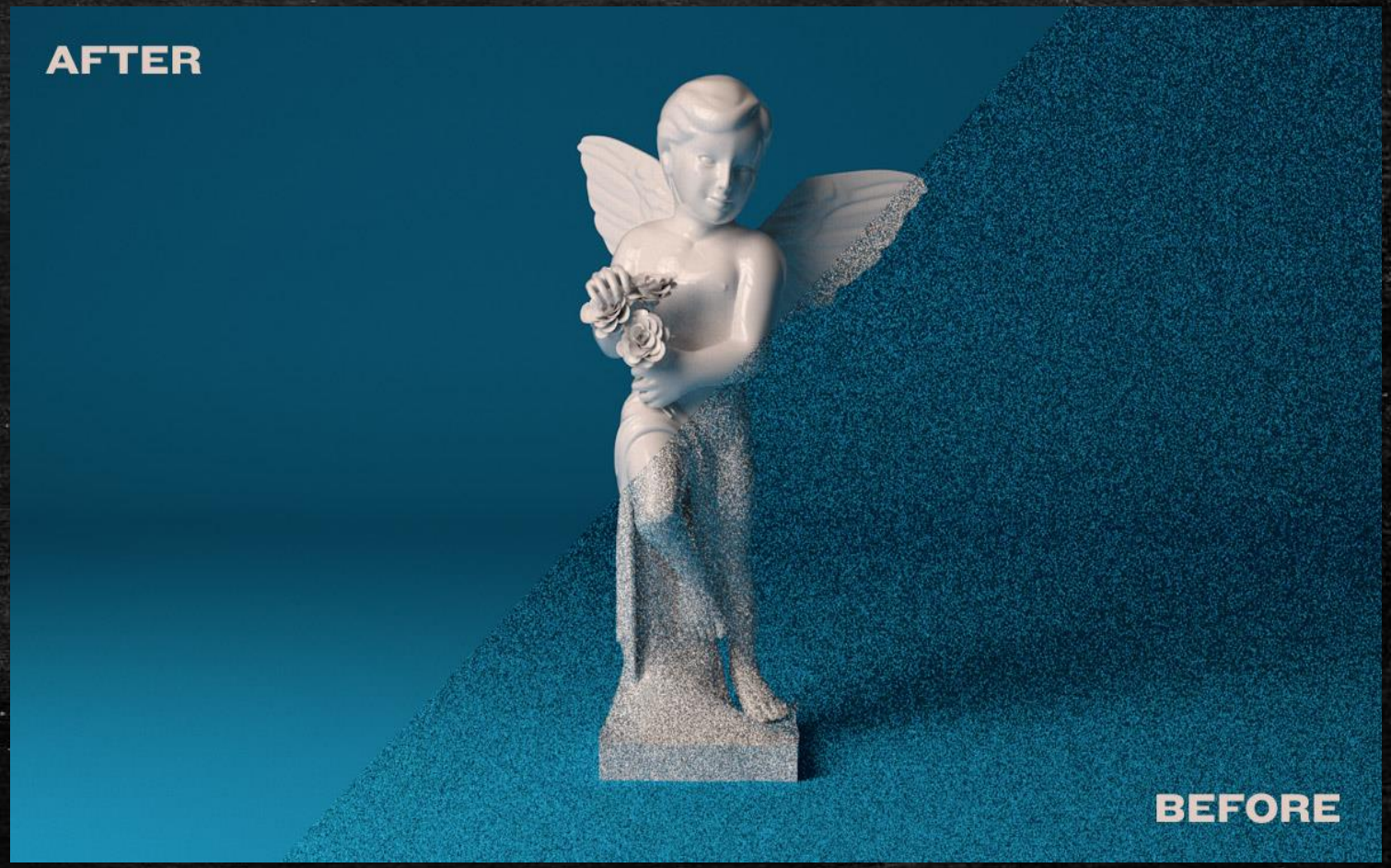
$= 4 * P_{inside} / P_{total}$



Improving Sampling

Sampling for graphics:

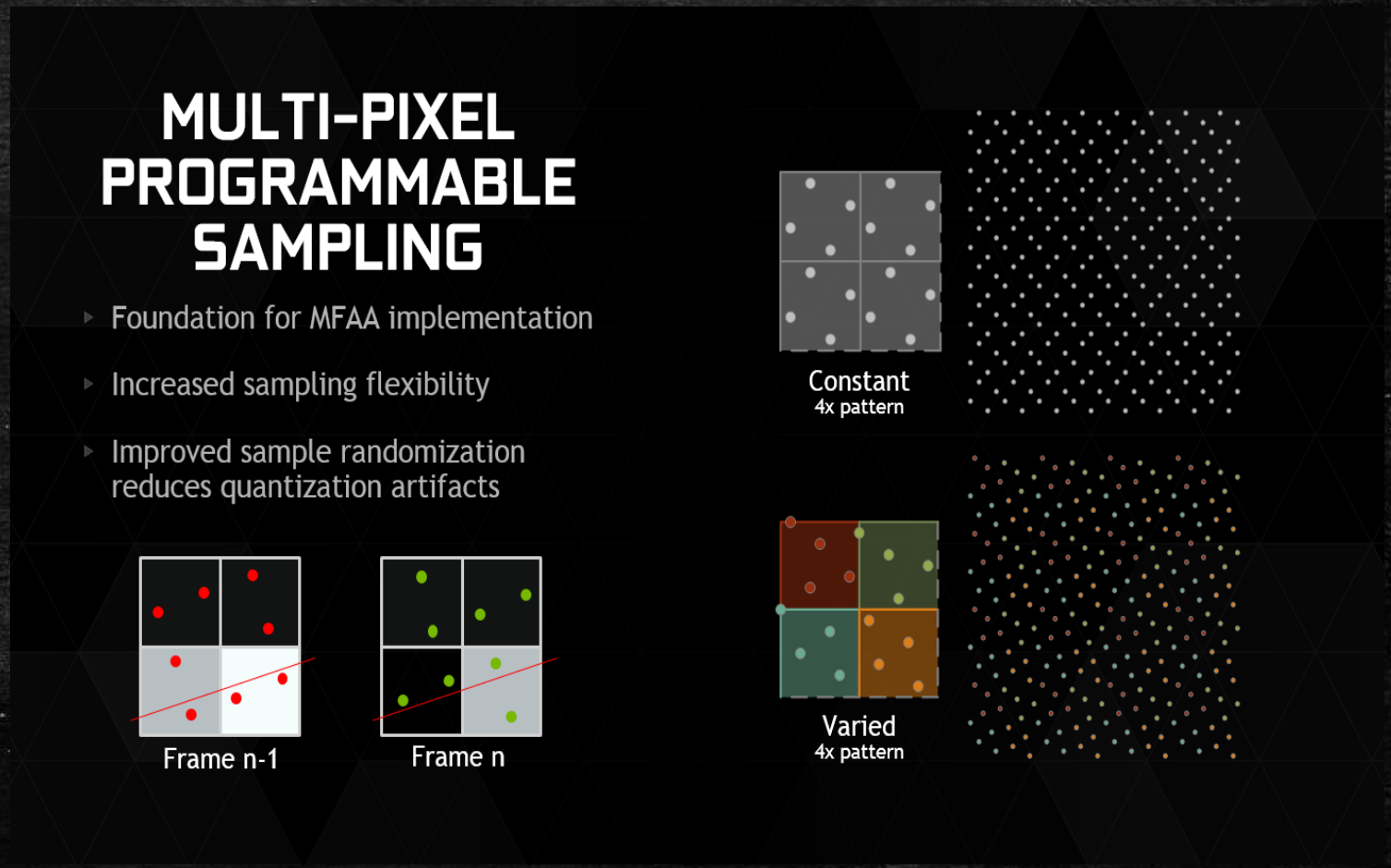
1. Area lights
2. Anti-aliasing
3. “Skylight”



Improving Sampling

Sampling for graphics:

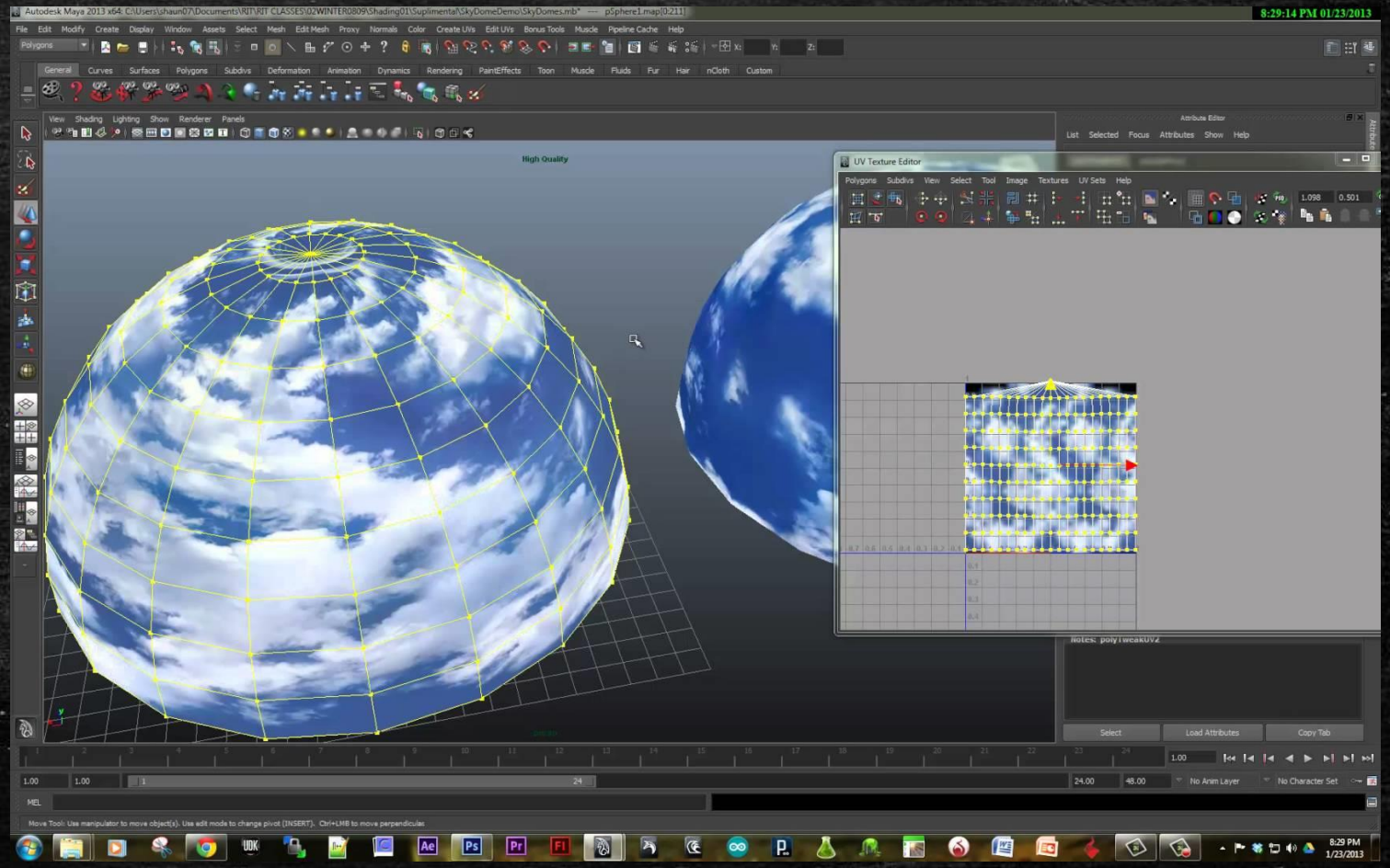
1. Area lights
2. Anti-aliasing
3. "Skylight"



Improving Sampling

Sampling for graphics:

1. Area lights
2. Anti-aliasing
3. “Skylight”
4. Motion blur
5. Depth of field
6. Dispersion
7. …



Improving Sampling

So, *sampling is everywhere*.

In fact, path tracing is a sampling process:

$$L_o(\mathbf{x}, \mathbf{w}) = L_e(\mathbf{x}, \mathbf{w}) + \int_\Omega f_r(\mathbf{x}, \mathbf{w}', \mathbf{w}) L_i(\mathbf{x}, \mathbf{w}')(-\mathbf{w}' \cdot \mathbf{n}) d\mathbf{w}'$$

We will evaluate this recursive integral by averaging multiple random samples.
One sample in this context becomes:

A random path from the camera through a pixel to a light source.

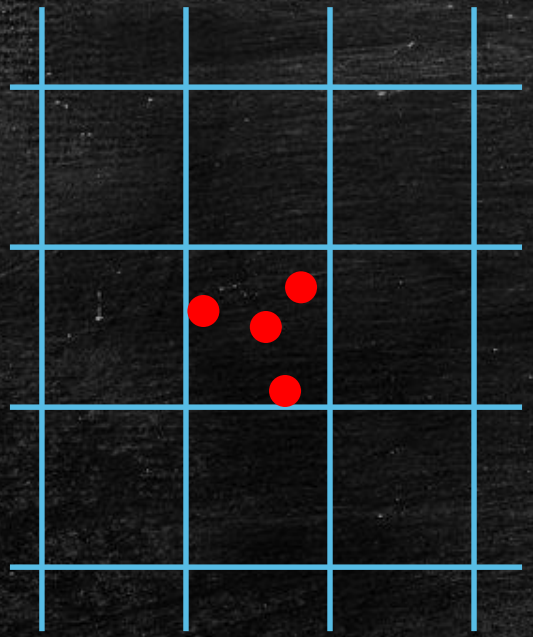
Improving Sampling

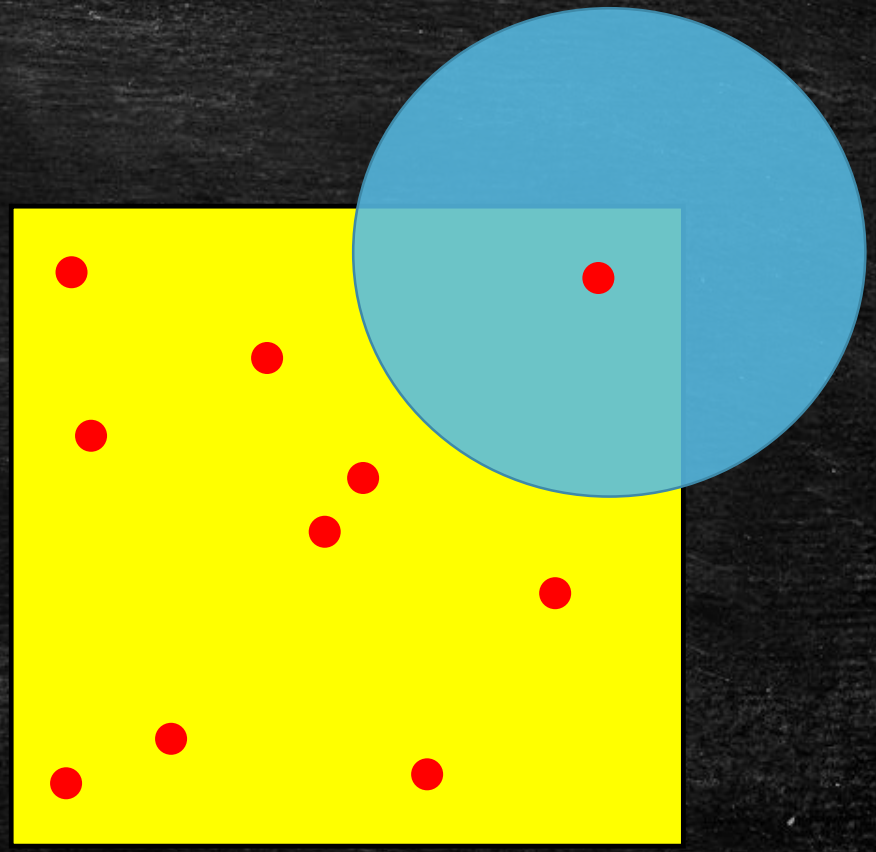
Question: can we do better than random sampling?
Answer: Yes we can.

Approaches:

1. Uniform sampling
2. Importance sampling

Improving Sampling



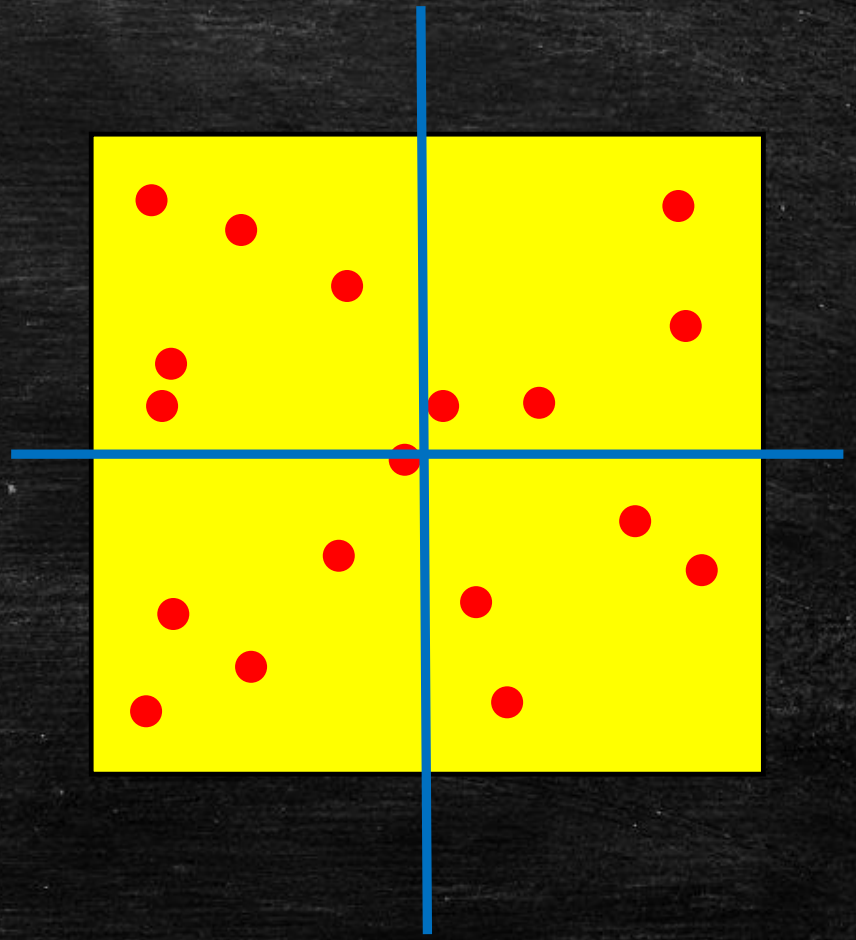


Improving Sampling

Stratification

Improving uniformity of the samples

Disadvantage: when drawing N samples, N must be a multiple of 4.



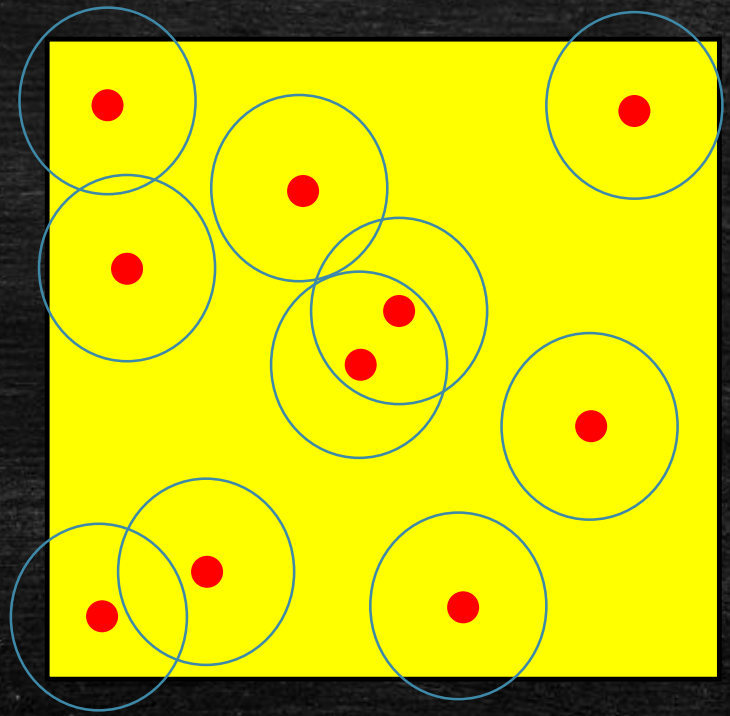
Improving Sampling

Poisson Disc

Maximizing uniformity

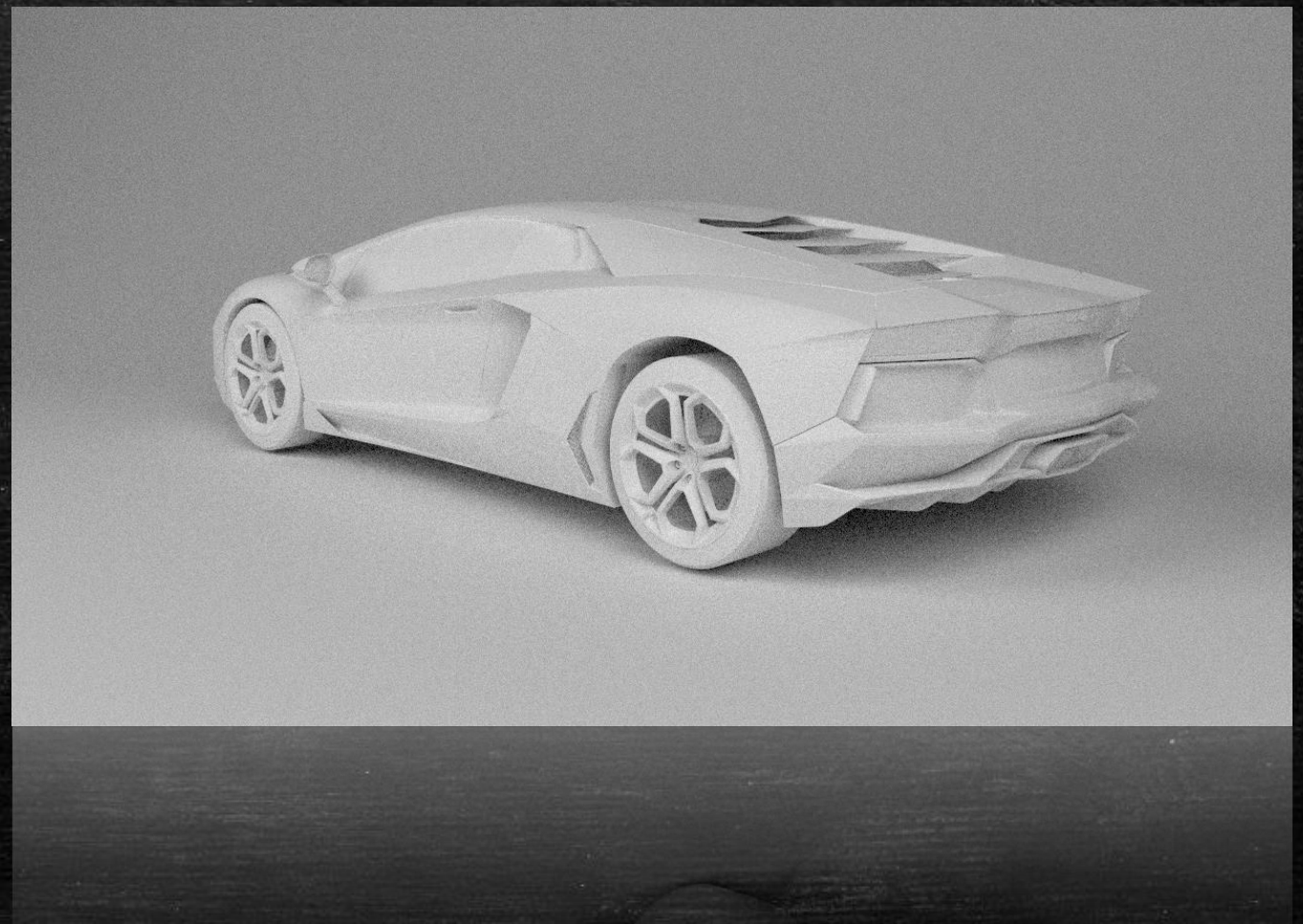
Start with random points,
make points repel each other,
wait until system stabilizes.

(Note: much better approaches exist)



Improving Sampling

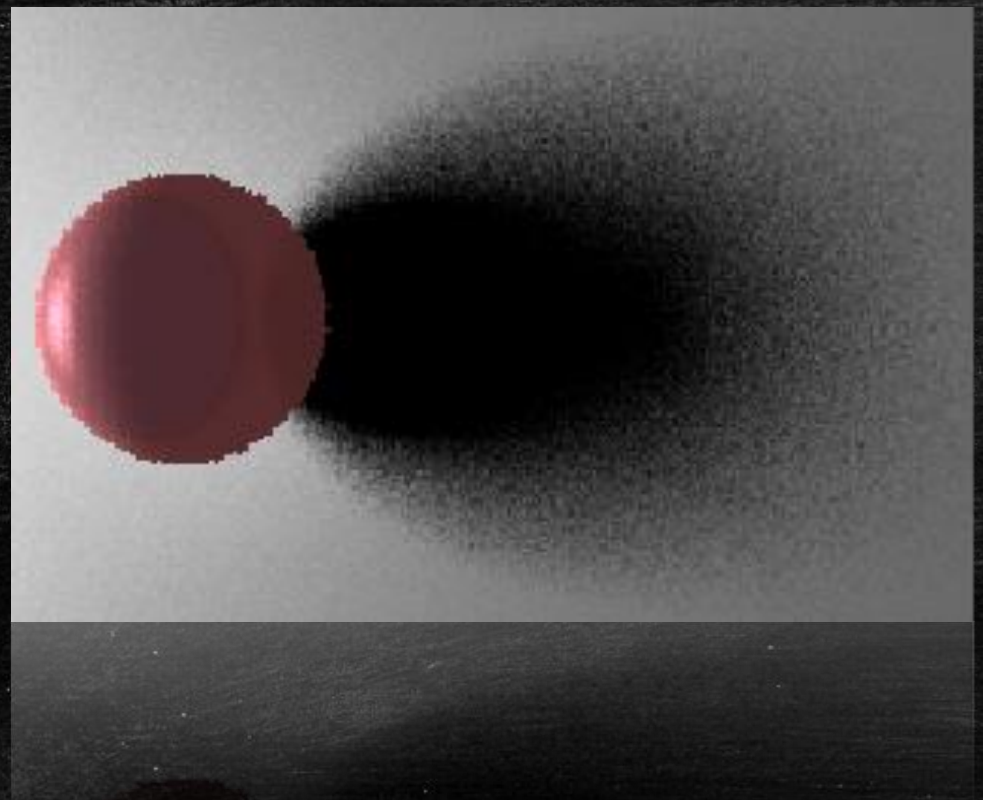
Worth the extra effort?

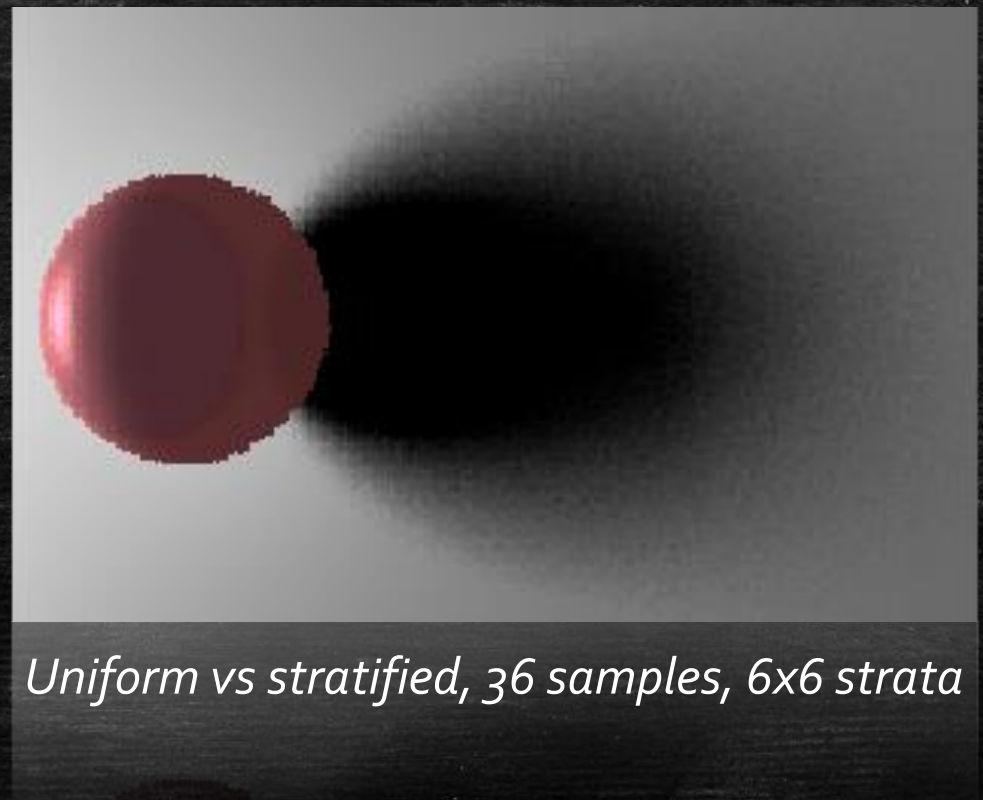



Uniform

Improving Sampling

Worth the extra effort?





Uniform vs stratified, 36 samples, 6x6 strata

Improving Sampling

Further reading:

Quasi-Monte Carlo Rendering with Adaptive Sampling,
http://www.kki.yamanashi.ac.jp/~ohbuchi/online_pubs/eg96_html/eg96.htm

Poisson Disk sampling,
http://devmag.org.za/2009/05/03/poisson-disk-sampling

PCF shadows using Poisson Disk,
https://electronicmeteor.wordpress.com/2013/02/05/poisson-disc-shadow-sampling-ridiculously-easy-and-good-looking-too

Today:

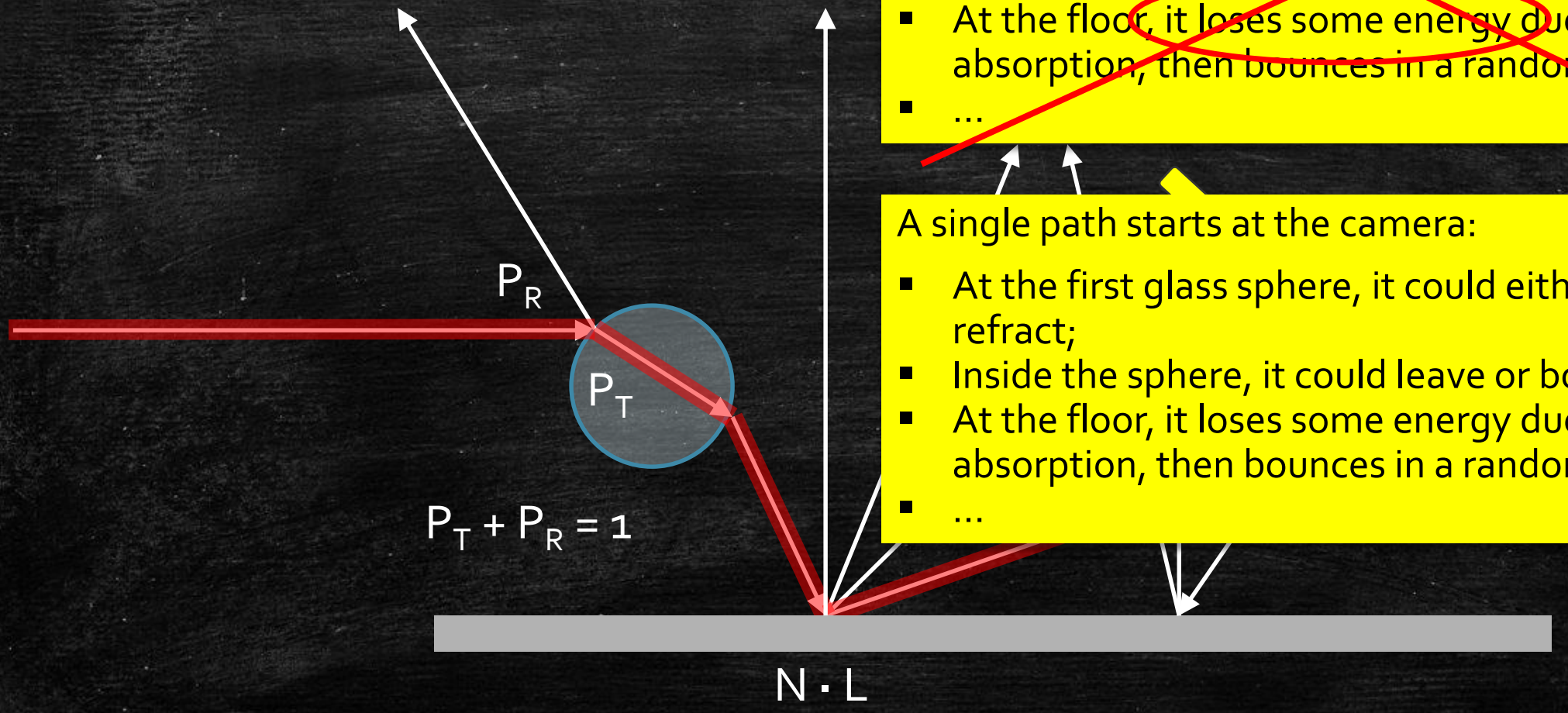
- Improved Sampling
- **Path Tracing Recap**
- Explicit Light Sampling
- Importance Sampling
- Resampled Importance Sampling
- Multiple Importance Sampling

Path Tracing Recap

Basic idea:

A 100 watt light bulb emits ~10^{20} photons per second.
Simulating these, and recording which ones hit the camera film, yields a correct image.

Path Tracing



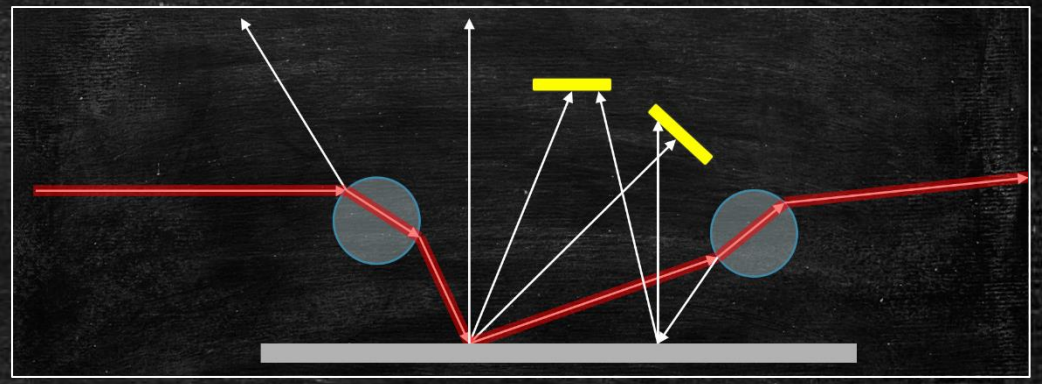
A single photon leaves the camera:

- At the first glass sphere, it could either reflect or refract;
- Inside the sphere, it could leave or bounce back in;
- At the floor, it loses some energy due to absorption, then bounces in a random direction;
- …

A single path starts at the camera:

- At the first glass sphere, it could either reflect or refract;
- Inside the sphere, it could leave or bounce back in;
- At the floor, it loses some energy due to absorption, then bounces in a random direction;
- …

Path Tracing



Constructing one random path backwards:
One random *sample*.

For each path a number of uniform random variables is used:

- Position on the film (2 values; for depth of field);
- Position on the pixel (2 values; for anti-aliasing);
- Reflect or refract (1 value);
- Reflect or refract (1 value);
- Direction of diffuse bounce (2 values);
- …

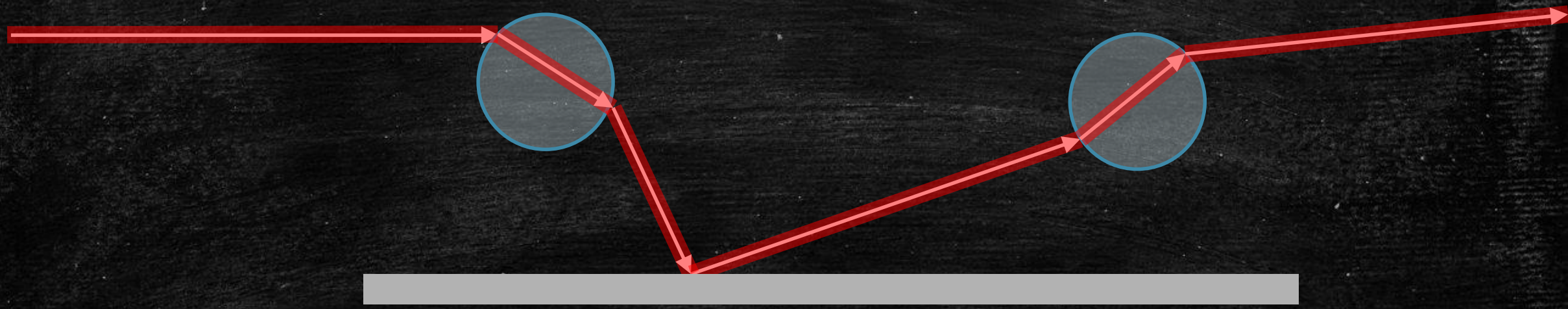
This process is referred to as a *random walk*.

Path Tracing

normalization; $\int N \cdot D$ (over hemisphere) = 0.5.

Basic algorithm:

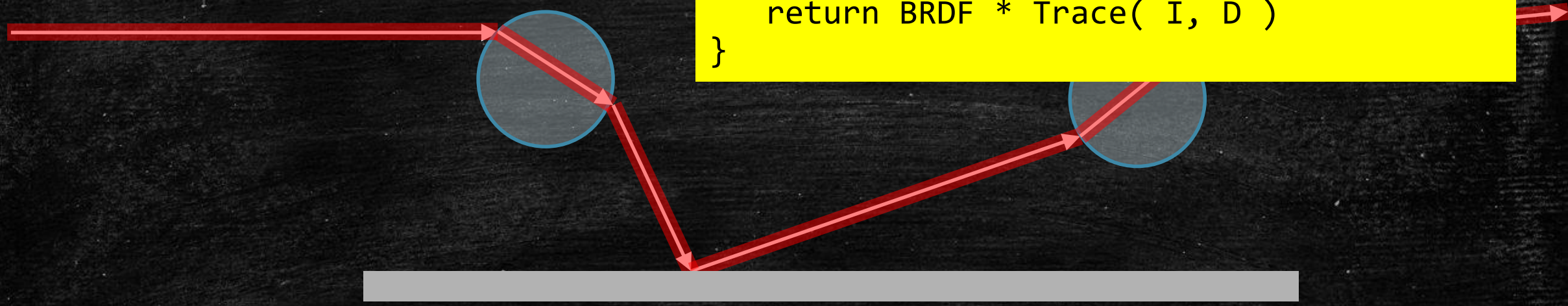
```
color3f Trace( vec3 O, vec3 D )
{
    I,N,mat = Intersect( O, D )
    if (mat.IsLight()) return mat.emissive
    D = RandomReflection( N )
    BRDF = 2 * mat.color * dot( N, D )
    return BRDF * Trace( I, D )
}
```



Path Tracing

Basic algorithm:

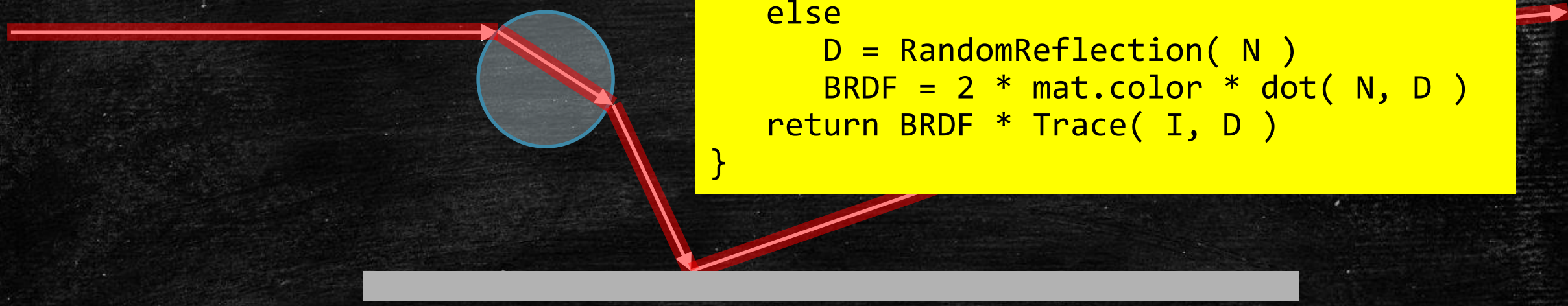
```
color3f Trace( vec3 O, vec3 D )
{
   I,N,mat = Intersect( O, D )
   if (mat.IsLight()) return mat.emissive
   if (mat.IsMirror())
      D = Reflection( D, N )
      BRDF = 1 * mat.color
   else
      D = RandomReflection( N )
      BRDF = 2 * mat.color * dot( N, D )
   return BRDF * Trace( I, D )
}
```



Path Tracing

Basic algorithm:

```
color3f Trace( vec3 O, vec3 D )
{
   I,N,mat = Intersect( O, D )
   if (mat.IsLight()) return mat.emissive
   if (mat.IsMirror())
      D = Reflection( D, N )
      BRDF = 1 * mat.color
   else if (mat.IsDielectric())
      D = ReflectOrRefract( D, N, mat )
      BRDF = 1 * mat.color
   else
      D = RandomReflection( N )
      BRDF = 2 * mat.color * dot( N, D )
   return BRDF * Trace( I, D )
}
```



Path Tracing

Quick question:
Does this produce photo-realistic image?

Yes!

But:
Only for scenes with the following materials:

- Pure diffuse
- Pure mirrors
- Pure dielectrics.

And:
It's very inefficient (why?).

Basic algorithm:

```
color3f Trace( vec3 O, vec3 D )
{
   I,N,mat = Intersect( O, D )
   if (mat.IsLight()) return mat.emissive
   if (mat.IsMirror())
      D = Reflection( D, N )
      BRDF = 1 * mat.color
   else if (mat.IsDielectric())
      D = ReflectOrRefract( D, N, mat )
      BRDF = 1 * mat.color
   else
      D = RandomReflection( N )
      BRDF = 2 * mat.color * dot( N, D )
   return BRDF * Trace( I, D )
}
```

Today:

- Improved Sampling
- Path Tracing Recap
- **Explicit Light Sampling**
- Importance Sampling
- Resampled Importance Sampling
- Multiple Importance Sampling

Explicit Light Sampling

$$L_o(\mathbf{x}, \mathbf{w}) = L_e(\mathbf{x}, \mathbf{w}) + \int_\Omega f_r(\mathbf{x}, \mathbf{w}', \mathbf{w}) L_i(\mathbf{x}, \mathbf{w}') (-\mathbf{w}' \cdot \mathbf{n}) d\mathbf{w}'$$

$L_i(x,w')$: light arriving at x from direction w'.

In direction w', we find a point y. Point y is reflecting light in direction –w', consisting of:

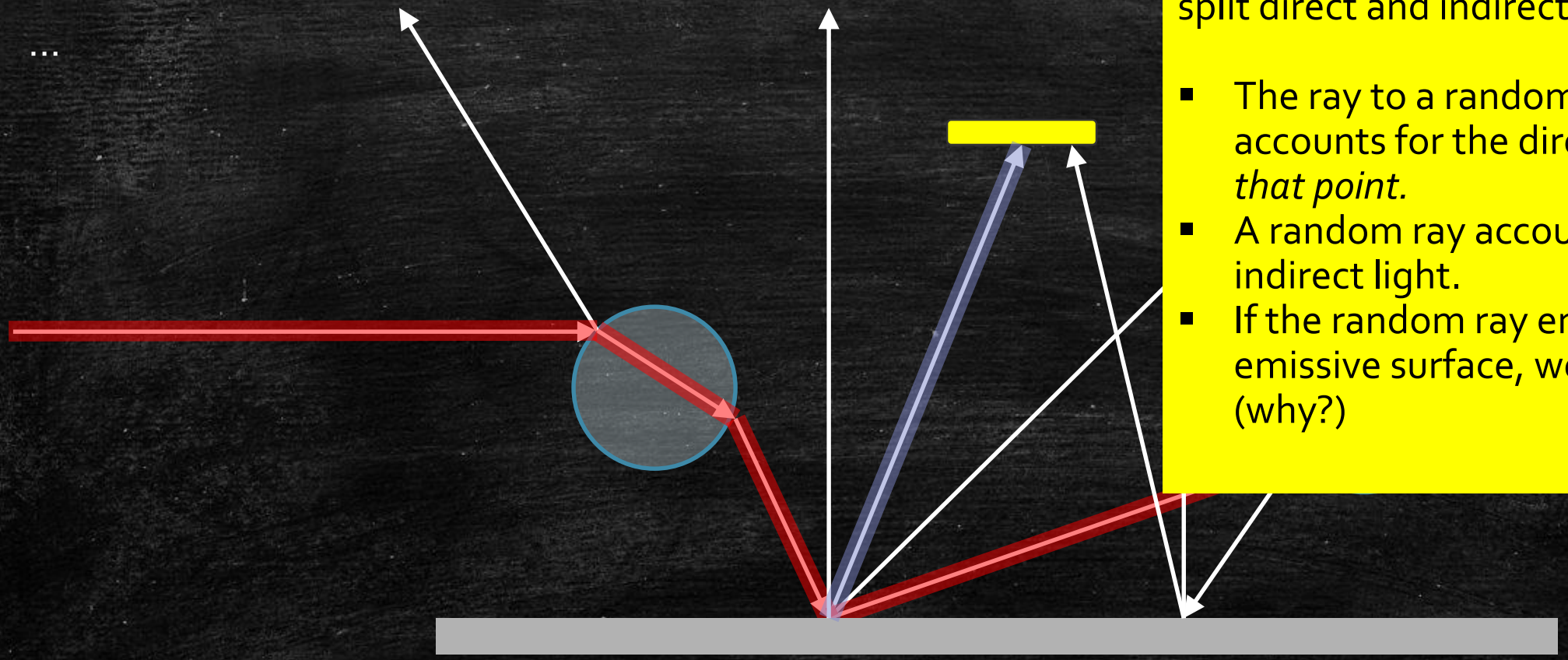
$L_e(y, -w')$ ← *direct light from y (if y is emissive)*

$F_r(y, w'', -w') L_i(y, -w') (-w'' \cdot n) d w''$ ← *indirect light, reflected by y*

We can evaluate direct and indirect light independently.

Explicit Light Sampling

…



At each diffuse surface:

- *Create a path to a random light source.*

This yields the correct result, if we split direct and indirect light:

- The ray to a random light source accounts for the direct light *at that point.*
- A random ray accounts for indirect light.
- If the random ray encounters an emissive surface, we discard it. (why?)

Explicit Light Sampling

Greatly increases the chance of actually reaching a light in a decent number of steps.

Depends on separation of direct and indirect light.

Direct light:

$$L_{direct} = Emission * 1/r^2 * N \cdot L$$

Account for the fact that we pick one random light:

$$L_{direct} = Emission * 1/r^2 * N \cdot L * N_{lights}$$

Today:

- Improved Sampling
- Path Tracing Recap
- Explicit Light Sampling
- **Importance Sampling**
- Resampled Importance Sampling
- Multiple Importance Sampling

Importance Sampling

Experiment:

We throw 2 dice. The value of the first one is always multiplied by 6, the other one is used as-is. The maximum value is thus 36 + 6 = 42; the minimum value is 7.

Question:

What is the average value of one roll?

Importance Sampling

Experiment:

Our scene contains 2 lights. One is huge and bright, the other one is small and faint. At each diffuse surface, we select a random light source to estimate direct illumination at that point.

Do we get a correct image if we sample both lights with equal probability?

(the answer is yes, but sampling the large light with a greater probability is more efficient)

Importance Sampling

Experiment:

When doing a diffuse bounce, rays along the normal are scaled by 1, while rays nearly parallel to the plane are scaled by almost 0. Should we produce all directions with equal probability?

Importance Sampling

Importance sampling: *Favoring paths that matter most.*

This requires careful scaling of path transport:

TWO LIGHTS: When we sample only one, we scale its contribution by 1/P, where P is the probability of selecting that one light.

Example:

Light 1 has a 25% probability of being picked, light 2 75%.
If we pick light 1, we scale its contribution by 1/0.25 = 4.
If we pick light 2, we scale its contribution by 1/0.75 ≈ 1.3333.

Verify: 250 * 4 + 750 * 1.3333 = 1000 + 1000 = 2000.

Importance Sampling a BRDF

Rather than picking a random reflection, we can also pick a random reflection with a *cosine weighted distribution*. Rays in this distribution have a greater probability of being generated in the direction of the normal.

(but, there's always a small chance that we produce a ray that is nearly parallel to the plane).

The distribution of probabilities is referred to as PDF: *Probability Density Function*.

A PDF is optimal if it is proportional to the function being sampled.

What is the PDF for the dice? And for the two light sources?

Importance Sampling a

```
vec3 diffuseReflectionCosWeighted( -
{
   float r = sqrt( r1 ), theta = 2 *
   return vec3(r * cos( theta ), r *
}
```

Basic algorithm:

```
color3f Trace( vec3 O, vec3 D )
{
   I,N,mat = Intersect( O, D )
   if (mat.IsLight()) return mat.emissive
   D = RandomReflectionCosWeighted( N )
   PDF = dot( N, D ) * (1 / (2 * PI))
   BRDF = 2 * mat.color * dot( N, D )
   return (BRDF / PDF) * Trace( I, D )
}
```

This PDF matches the diffuse BRDF exactly: the probability of picking outgoing direction L is N · L.
Note that the contribution of a ray produced by this method must be scaled by 1/P.

Importance Sampling a BRDF

Today:

- Improved Sampling
- Path Tracing Recap
- Explicit Light Sampling
- Importance Sampling
- **Resampled Importance Sampling**
- Multiple Importance Sampling

RIS[1]

Importance sampling for lights:
Favoring the brighter ones

What determines 'brightness'?

1. Lamp material brightness
2. Lamp area
3. Distance to the camera
4. Angle
5. Occlusion

But:

- What if your lights are far apart?
- What if a bright light is occluded?
- What if we have many lights?

1. Importance Resampling for Global Illumination, Talbot et al.

RIS – Many lights

Picking a random light from a large set:

For each light, calculate potential contribution
Then, randomly pick a light with a probability proportional to this potential contribution.

Problem:
For many lights, this is a lot of work.

Solution:
Precalculate potential contribution as brightness * area.
(note: this does not take into account distance from the camera, nor orientation)

RIS – Many lights

Solution:
Precalculate potential contribution as brightness * area.

RIS: use this coarse estimate to select N lights.
For each of these N lights, calculate potential contribution more accurately:

- Distance to camera
- Angle relative to camera

From these N lights, randomly select one based on the accurate estimate.

THE END

next week: advanced path tracing