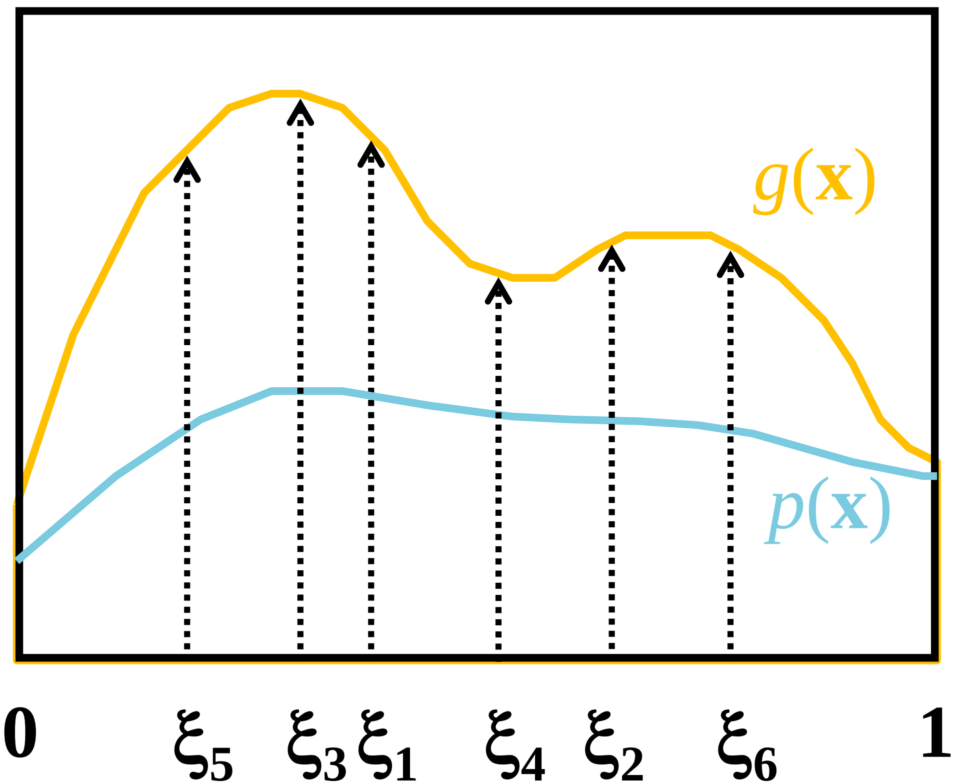
Computer graphics III – Monte Carlo integration II

Jaroslav Křivánek, MFF UK
Jaroslav.Krivanek@mff.cuni.cz

Monte Carlo integration

- General tool for estimating definite integrals



Integral:

$$I = \int g(\mathbf{x})\mathrm{d}\mathbf{x}$$

Monte Carlo estimate of I:

$$\langle I \rangle = \frac{1}{N}\sum_{k=1}^{N}\frac{g(\xi_k)}{p(\xi_k)}; \quad \xi_k \propto p(\mathbf{x})$$

Works "on average":

$$E[\langle I \rangle] = I$$

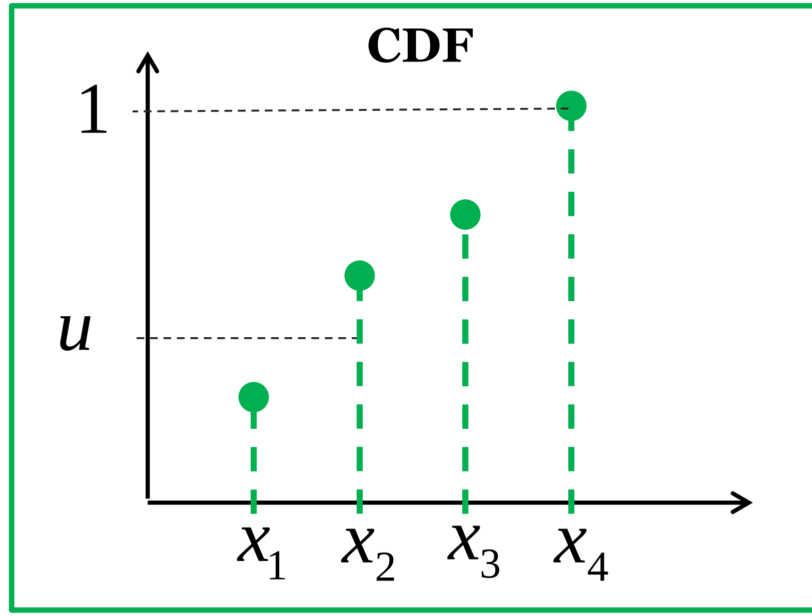
Generating samples from a distribution

PBRT 13.3

http://www.pbr-book.org/3ed-2018/Monte_Carlo_Integration/Sampling_Random_Variables.html#

Generating samples from a 1D discrete random variable

- Given a probability mass function $p(i)$, and the corresponding cdf $P(i)$



- Procedure
  1. Generate u from Uniform(0,1)
  2. Choose x_i for which

$$P(i-1) < u \leq P(i)$$

(we define $P(0) = 0$)

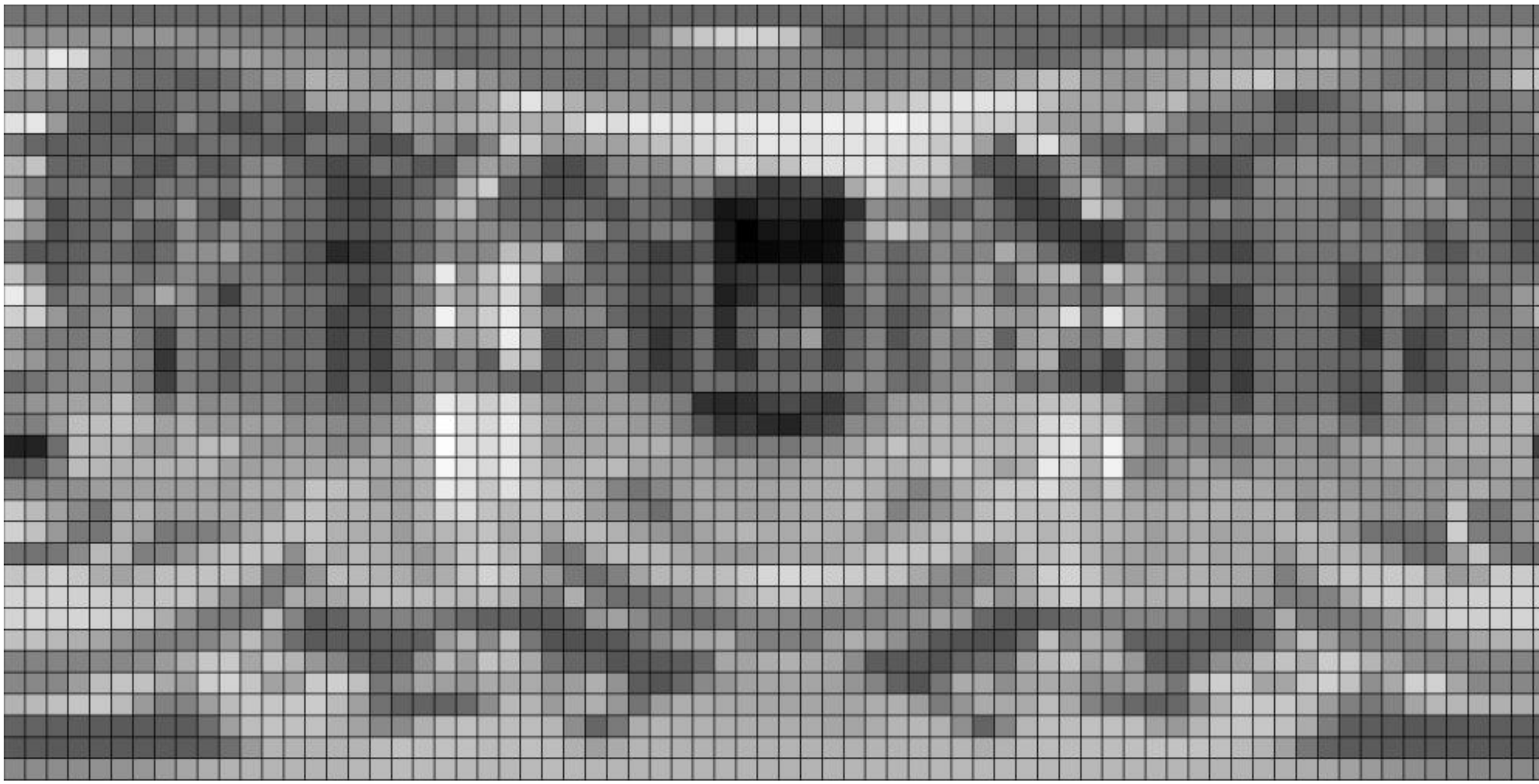
- The search is usually implemented by interval bisection

Generating samples from a 2D discrete random variable

- Given a probability mass function $p_{I,J}(i, j)$

- Option 1:
  - Interpret the 2D PMF as a 1D vector of probabilities
  - Generate samples as in the 1D case

Generating samples from a 2D discrete random variable



Generating samples from a 2D discrete random variable

- Option 2 (better)
  1. "Column" i_{sel} is sampled from the marginal distribution, given by a 1D marginal pmf

$$p_I(i) = \sum_{j=1}^{n_j} p_{I,J}(i, j)$$

  2. "Row" j_{sel} is sampled from the conditional distribution corresponding to the "column" i_{sel}

$$p_{J|I}(j \mid I = i_{\text{sel}}) = \frac{p_{I,J}(i_{\text{sel}}, j)}{p_I(i_{\text{sel}})}$$

Generating samples from a 1D continuous random variable

- Option 1: **Transformation method**

- Option 2: **Rejection sampling**

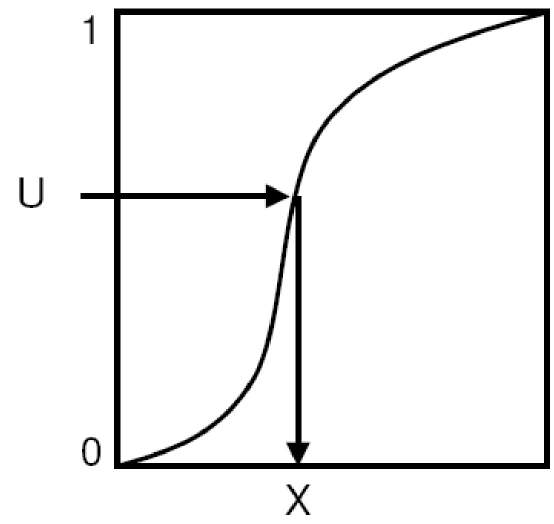
- Option 3: **Metropolis-Hastings sampling**
  - Separate lecture

Transformation method

- **<u>Theorem</u>** Consider a random variable *U* from the uniform distribution U (0, 1).
  Then the random variable *X*

  $$X = P^{-1}(U)$$

  has the distribution given by the **cdf** *P*.



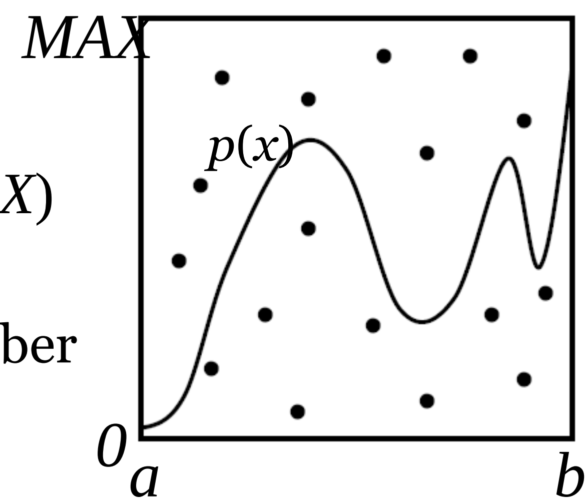
- To generate samples according to a given pdf *p*, we need to be able to:
  - calculate the cdf $P(x)$ from the pdf $p(x)$
  - calculate the inverse cdf $P^{-1}(u)$

  (analytically, on paper)

EXAMPLE DERIVATION FOR SAMPLING FROM UNIFORM(a,b) and EXP(a,b)

Rejection sampling in 1D

- **Algorithm**
  - Choose random u_1 from Uniform(a, b)
  - Choose random u_2 from Uniform(0, MAX)
  - Accept the sample if $p(u_1) > u_2$
    - Return u_1 as the generated random number
  - Repeat until a sample is accepted



- **<u>Theorem</u>** The accepted samples follow the distribution with the pdf $p(x)$.

- Efficiency = % of accepted samples
  - Area under the pdf graph / area of the bounding rectangle

Transformation method vs. Rejection sampling

- Transformation method: **Pros**
  - Almost always more efficient than rejection sampling (unless the transformation formula $x = P^{-1}(u)$ turns out extremely complex)
  - Constant time complexity. The number of random generator invocations is known upfront (important for SW architecture).
- Transformation method: **Cons**
  - May not be feasible (we may not be able to find the suitable form for $x = P^{-1}(u)$ analytically), but rejection sampling is always applicable as long as we can evaluate and bound the pdf (i.e. rejection sampling is more general)
- Smart rejection sampling can be very efficient (e.g. the Ziggurat method, see Wikipedia, https://en.wikipedia.org/wiki/Ziggurat_algorithm)

Sampling from a 2D continuous random variable

- Conceptually similar to the 2D discrete case
- Procedure
  - Given the joint density $p_{X,Y}(x, y) = \mathrm{p}_X(\mathrm{x})\, \mathrm{p}_{Y|X}(\mathrm{y} \mid \mathrm{x})$

1. Choose x_{sel} from the **marginal pdf**

$$p_X(x) = \int p_{X,Y}(x, y)\, \mathrm{d}y$$

2. Choose y_{sel} from the **conditional pdf**

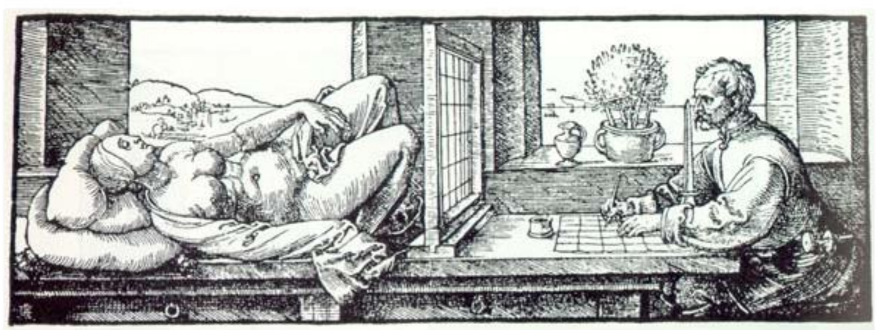
$$p_{Y|X}(y \mid X = x_{\mathrm{sel}}) = \frac{p_{X,Y}(x_{\mathrm{sel}}, y)}{p_X(x_{\mathrm{sel}})}$$

Transformation formulas for common cases in light transport

- P. Dutré: **Global Illumination Compendium**, http://people.cs.kuleuven.be/~philip.dutre/GI/

Global Illumination Compendium

The Concise Guide to Global Illumination Algorithms



Albrecht Duerer, *Underweysung der Messung mit dem Zirkel und Richtscheyt* (Nurenberg, 1525), Book 3, figure 67.

- PBRT, Section 13.6. http://www.pbr-book.org/3ed-2018/Monte_Carlo_Integration/2D_Sampling_with_Multidimensional_Transformations.html

Importance sampling from the physically-plausible Phong BRDF

- Ray hits a surface with a Phong BRDF. How do we generate a ray direction proportional to the BRDF lobe?

- Procedure

1. Choose the BRDF component (diffuse reflection, specular reflection, possibly refraction)

2. Sample direction from the selected component

3. Evaluate the total PDF and BRDF

Recap: Physically-plausible Phong BRDF

$$f_r^{\mathrm{Phong}}(\omega_{\mathrm{in}} \to \omega_{\mathrm{out}}) = \frac{\rho_{\mathrm{d}}}{\pi} + \frac{n+2}{2\pi}\rho_{\mathrm{s}} \max\{0, \cos\theta_{\mathrm{refl}}\}^n$$

- Where

$$\cos\theta_{\mathrm{refl}} = \omega_{\mathrm{out}} \cdot \omega_{\mathrm{refl}}$$

$$\omega_{\mathrm{refl}} = 2(\omega_{\mathrm{in}} \cdot \mathbf{n})\mathbf{n} - \omega_{\mathrm{in}}$$

- Energy conservation:

$$\rho_{\mathrm{d}} + \rho_{\mathrm{s}} \le 1$$

Selection of the BRDF component

```
float probDiffuse  = max(rhoD.r, rhoD.g, rhoD.b);
float prodSpecular = max(rhoS.r, rhoS.g, rhoS.b);
float normalization = 1.f / (probDiffuse + probSpecular);
// probability of choosing the diffuse component
probDiffuse *= normalization;
// probability of choosing the specular component
probSpecular *= normalization;

if ( uniformRand(0,1) <= probDiffuse )
   generatedDir = sampleDiffuse();
else
   generatedDir = sampleSpecular(incidentDir);

pdf = evalPdf(incidentDir, generatedDir,
              probDiffuse, probSpecular);
```

what is incDir and genDir in a path and light tracer

Sampling of the diffuse lobe

- Importance sampling with the density $p(\theta) = \cos(\theta) / \pi$
  - θ…angle between the surface normal and the generated ray
  - Generating the direction:

$$\varphi = 2\pi r_1 \qquad \theta = \operatorname{acos}(r_2)$$

$$x = \cos(2\pi r_1)\sqrt{1 - r_2^2}$$
$$y = \sin(2\pi r_1)\sqrt{1 - r_2^2}$$
$$z = r_2$$

    - r1, r2 … uniform random variates on <0,1)
- Reference: Dutre, Global illumination Compendium
- Derivation: Pharr & Humphreys, PBRT

where derived? (spe

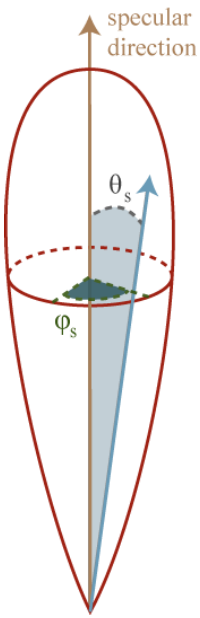
sampleDiffuse()

```
// generate spherical coordinates of the direction
const float r1 = uniformRand(0,1), r2 = uniformRand(0,1);
const float sinTheta = sqrt(1 – r2);
const float cosTheta = sqrt(r2);
const float phi      = 2.0*PI*r1;

// convert [theta, phi] to Cartesian coordinates
Vec3 dir (cos(phi)*sinTheta, sin(phi)*sinTheta, cosTheta);

return dir;
```

Here the generated direction is in the coordinate frame with the *z*-axis aligned to the surface normal (i.e. the local shading frame).

Sampling of the specular (glossy) component

- Importance sampling with the pdf $p(\theta_{\mathrm{refl}}) = (n+1)/(2\pi) \cos^n(\theta)$
  - θ_{refl} ...angle between the ideal mirror reflection of ω_{out} and the generated ray
  - Formulas for generating the direction:



$$\varphi = 2\pi r_1$$

$$\theta = \operatorname{acos}\left(r_2^{\frac{1}{n+1}} \right)$$

$$x = \cos(2\pi r_1)\sqrt{1 - r_2^{\frac{2}{n+1}}}$$

$$y = \sin(2\pi r_1)\sqrt{1 - r_2^{\frac{2}{n+1}}}$$

$$z = r_2^{\frac{1}{n+1}}$$

- r1, r2 … uniform random variates on <0,1)

sampleSpecular()

```
// build a lobe coordinate frame with ideal reflected direction = z-axis
Frame lobeFrame;
lobeFrame.setFromZ( reflectedDir(incidentDir, surfaceNormal) );

// generate direction in the lobe coordinate frame
//      use formulas form previous slide, n=Phong exponent
const Vec3 dirInLobeFrame = rndHemiCosN(n);

// transform dirInLobeFrame to local shading frame
const Vec3 dir = lobeFrame.toGlobal(dirInLobeFrame);

return dir;
```

evalPdf

```
float evalPdf(Dir incidentDir, Dir generatedDir,
              float probDiffuse, float probSpecular)
{
   return
     probDiffuse  * getDiffusePdf(generatedDir) +
     probSpecular * getSpecularPdf(incidentDir, generatedDir);
}
```

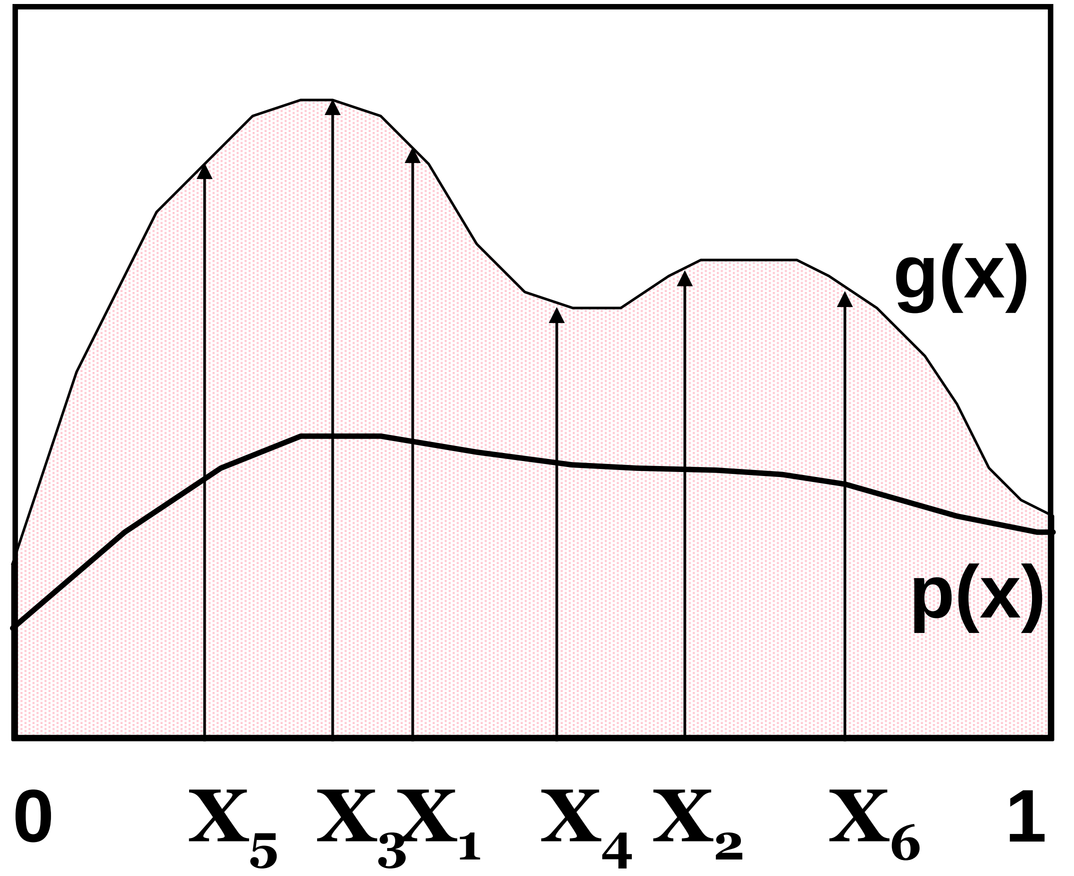
formulas from previous slides

Variance reduction methods for MC estimators

Variance reduction methods

- **Importance sampling**
  - The most commonly used method in light transport (most often we use BRDF-proportional importance sampling)
- **Control variates**
- **Improved sample distribution**
  - Stratification
  - quasi-Monte Carlo (QMC)

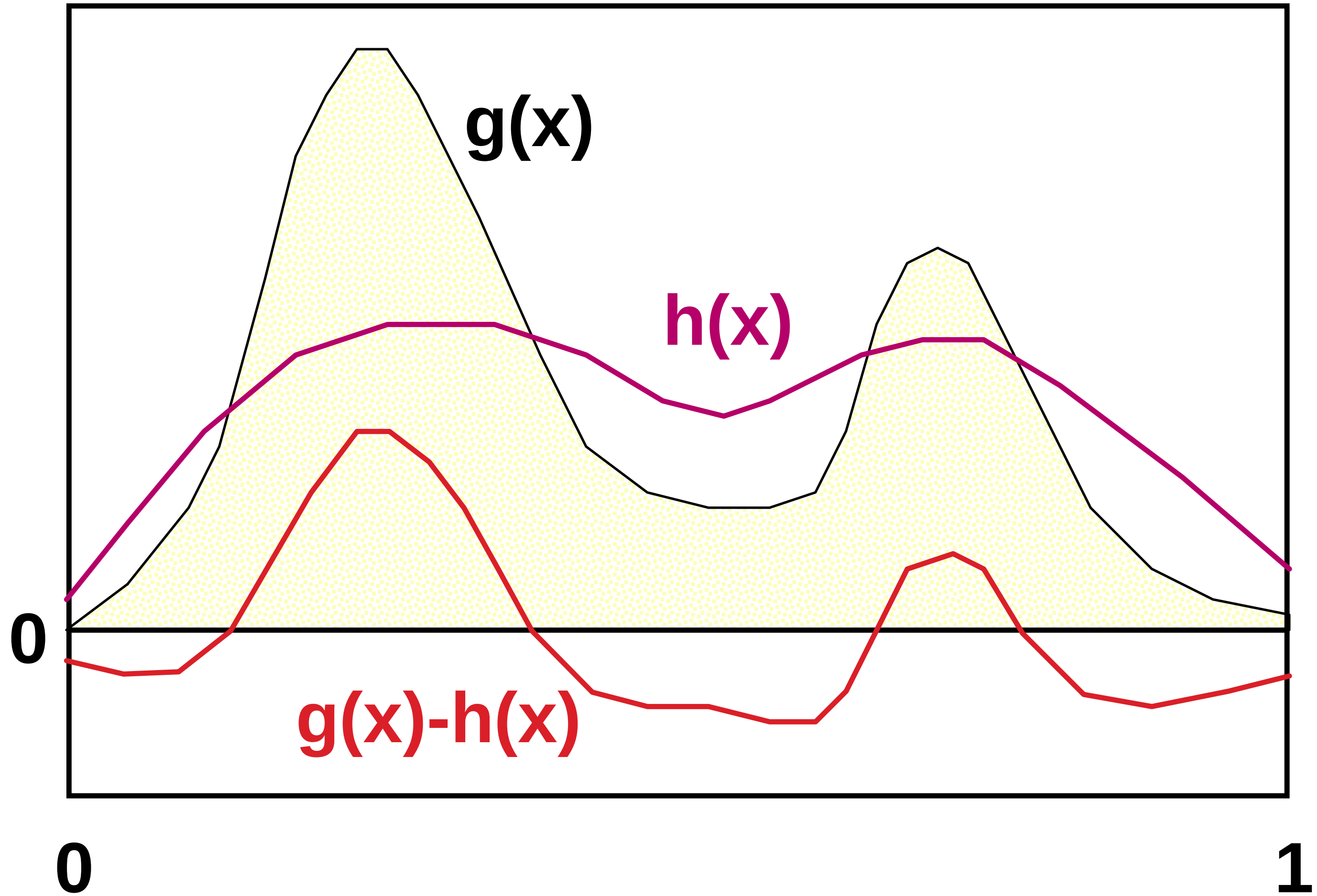
Importance sampling



Importance sampling

- Parts of the integration domain with high value of the integrand *g* are more important
  - Samples from these areas have higher impact on the result
- **Importance sampling** places samples preferentially to these areas
  - i.e. the **pdf** *p* is “similar” to the integrand *g*
- **Decreases variance** while keeping unbiasedness

Control variates



Control variates

Consider a function **h(x)**, that **approximates the integrand** and we can integrate it analytically:

$$I = \int g(\mathbf{x})\ \mathrm{d}\mathbf{x} = \underbrace{\int [g(\mathbf{x}) - h(\mathbf{x})]\ \mathrm{d}\mathbf{x}} + \underbrace{\int h(\mathbf{x})\ \mathrm{d}\mathbf{x}}$$

Numerical integration (MC)
Hopefully with less variance than integrating $g(\mathbf{x})$ directly.

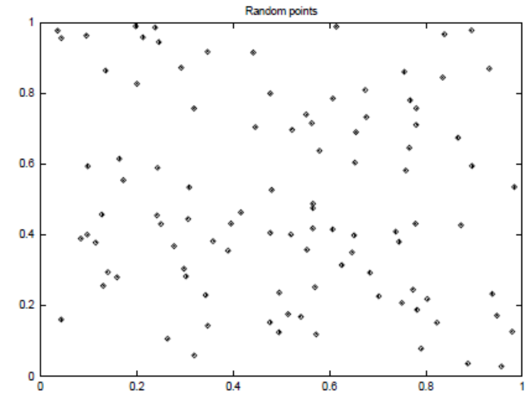
We can integrate analytically

Control variates vs. Importance sampling

- **Importance sampling**
  - Advantageous whenever the function, according to which we can generate samples, appears in the integrand as a **multiplicative factor** (e.g. BRDF in the reflection equation).
- **Control variates**
  - Better if the function that we can integrate analytically appears in the integrand as an **additive term**.
- This is why in light transport; we almost always use importance sampling and rarely control variates.

Better sample distribution

- Generating independent samples often leads to clustering of samples
  - Results in high estimator variance

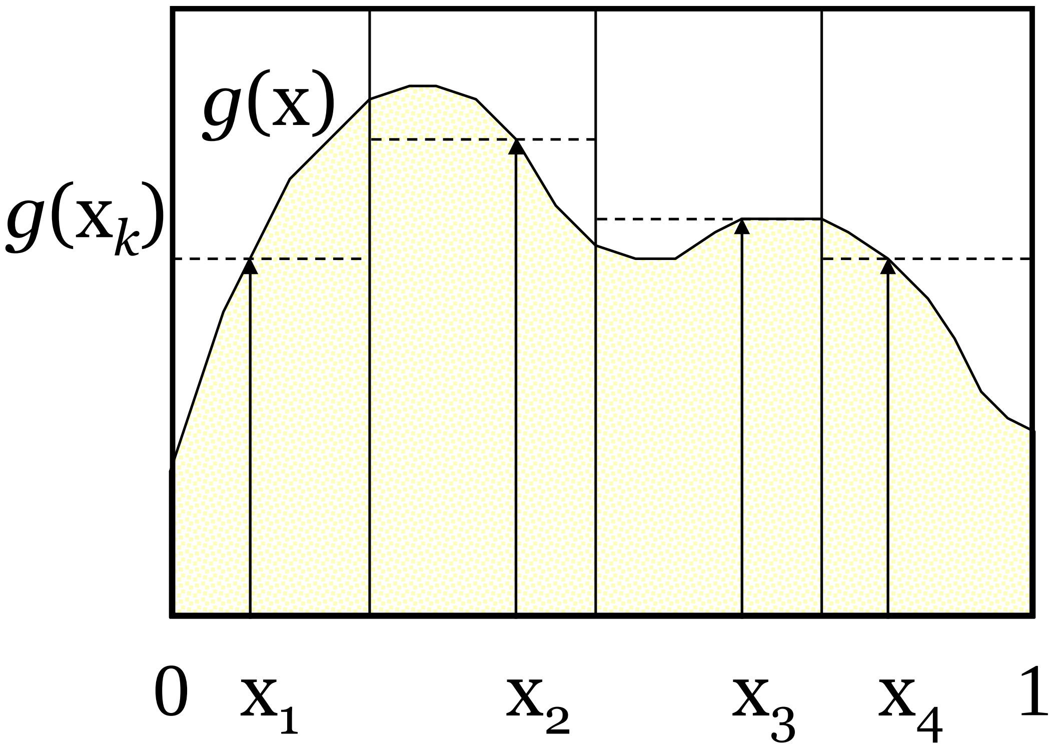


- Better sample distribution => better coverage of the integration domain by samples => lower variance

- Approaches
  - **Stratified sampling**
  - **quasi-Monte Carlo (QMC)**

Stratified sampling

- Sampling domain subdivided into disjoint areas that are sampled independently



Stratified sampling

Subdivision of the sampling domain Ω into N parts Ω_k:

$$I = \int_{\Omega} g(x)\,\mathrm{d}x = \sum_{k=1}^{N} \int_{\Omega_k} g(x)\,\mathrm{d}x = \sum_{k=1}^{N} I_k$$

Resulting estimator:

$$\hat{I}_{\text{strat}} = \frac{1}{N} \sum_{k=1}^{N} g(X_k), \qquad X_k \in \Omega_k$$

Stratified sampling

- Suppresses sample clustering
- Reduces estimator variance
  - Variance is provably less than or equal to the variance of a regular secondary estimator
- Very effective in low dimension
  - Effectiveness deteriorates for high-dimensional integrands

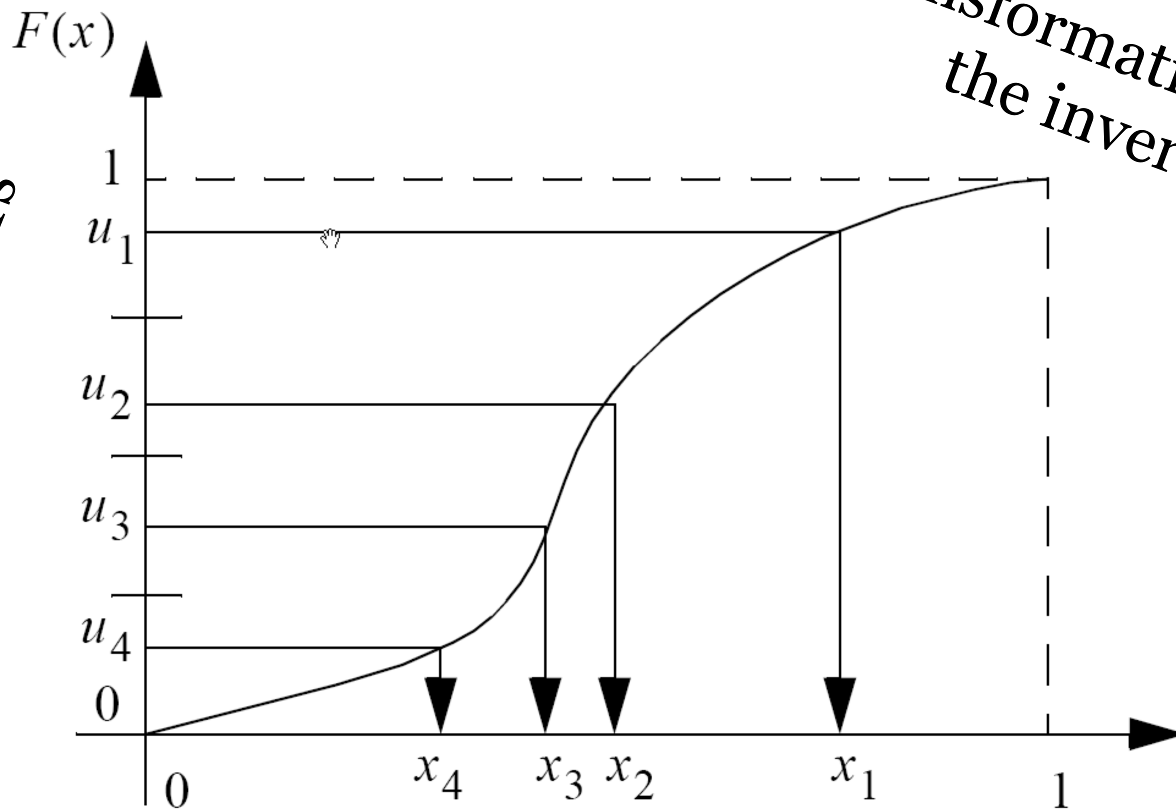
How to subdivide the interval?

- **Uniform** subdivision of the interval
  - Natural approach for a completely unknown integrand ***g***
- If we know at least roughly the shape of **the integrand *g***, we aim for a subdivision with the lowest possible variance on the sub-domains
- Subdivision of a ***d*-dimensional interval** leads to N^d samples
  - A better approach in high dimension is ***N*-rooks** sampling

Combination of stratified sampling and the transformation method

Transformation through the inverse cdf

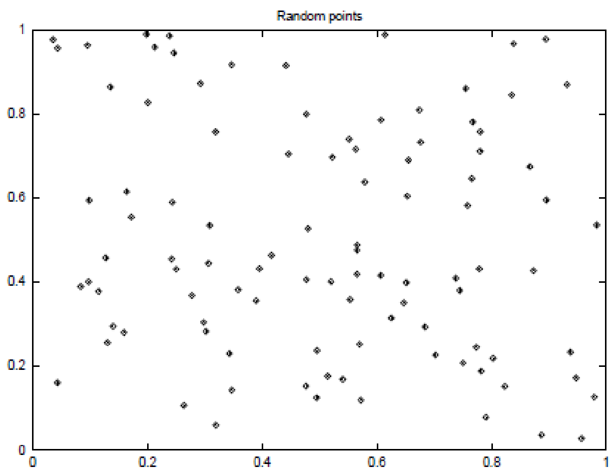
Stratification in the space of random numbers



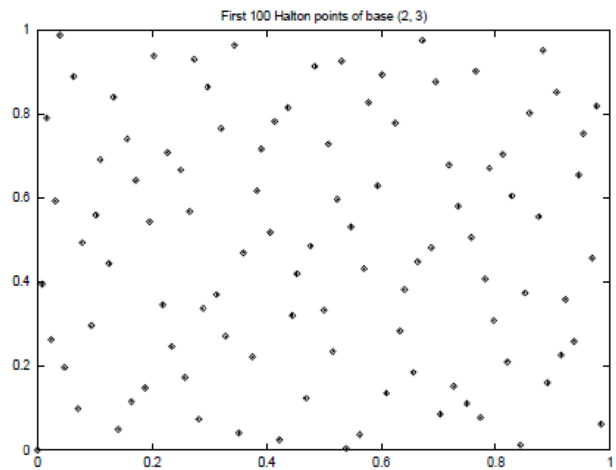
Quasi-Monte Carlo methods (QMC)

- Use of strictly deterministic sequences instead of (pseudo-)random numbers
- Pseudo-random numbers replaced by **low-discrepancy sequences**
- Everything works as in regular MC, but the underlying math is different (nothing is random so the math cannot be built on probability theory)

Discrepancy

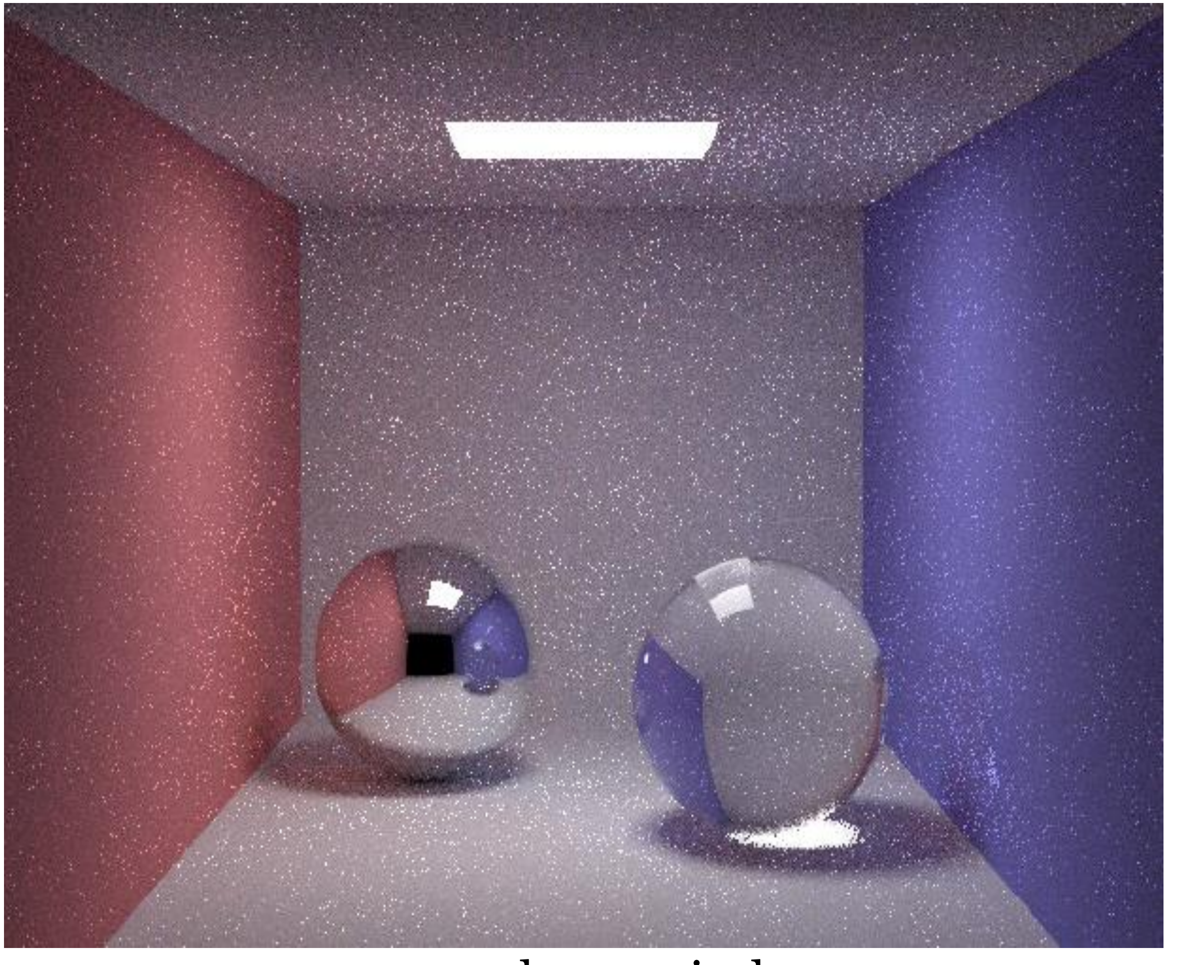


High Discrepancy
(clusters of points)



Low Discrepancy
(more uniform)

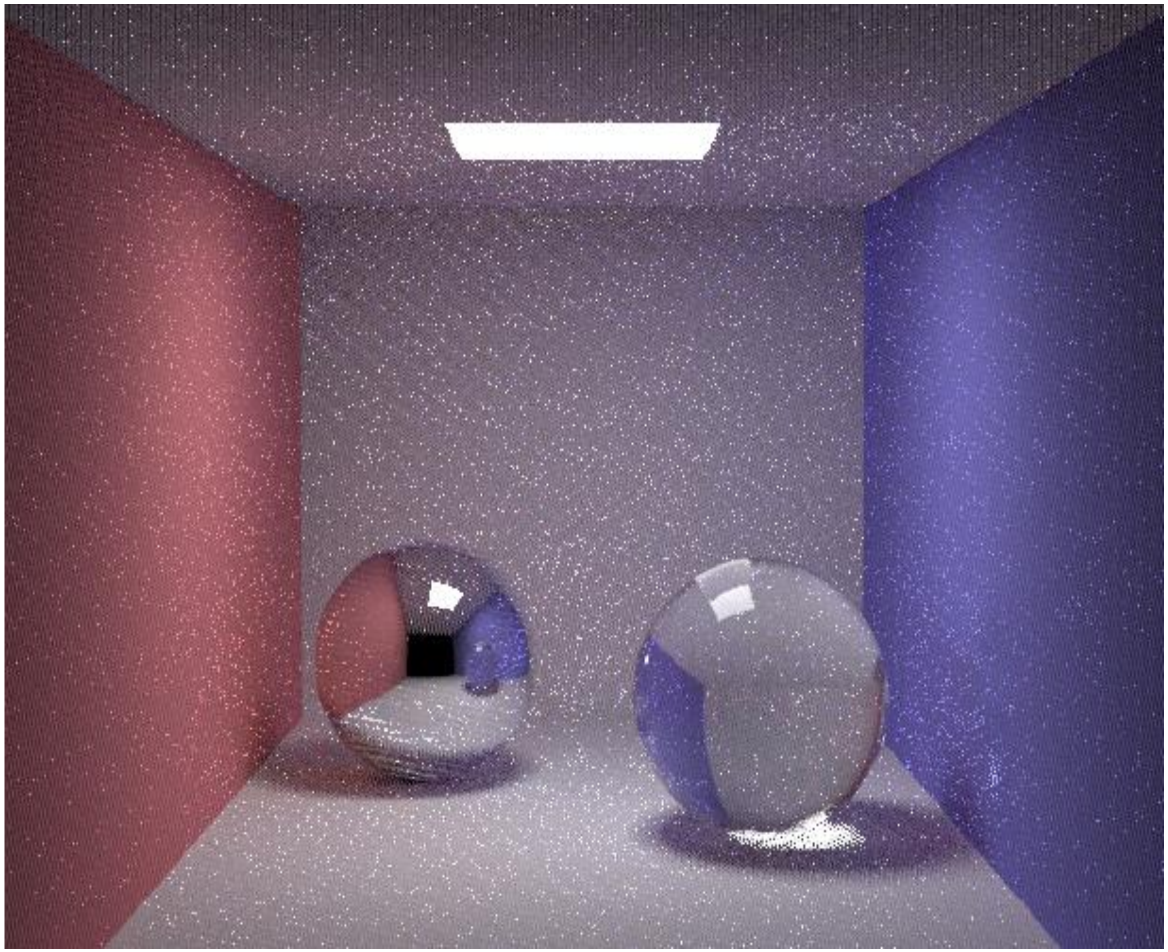
Stratified sampling



10 paths per pixel

Henrik Wann Jensen

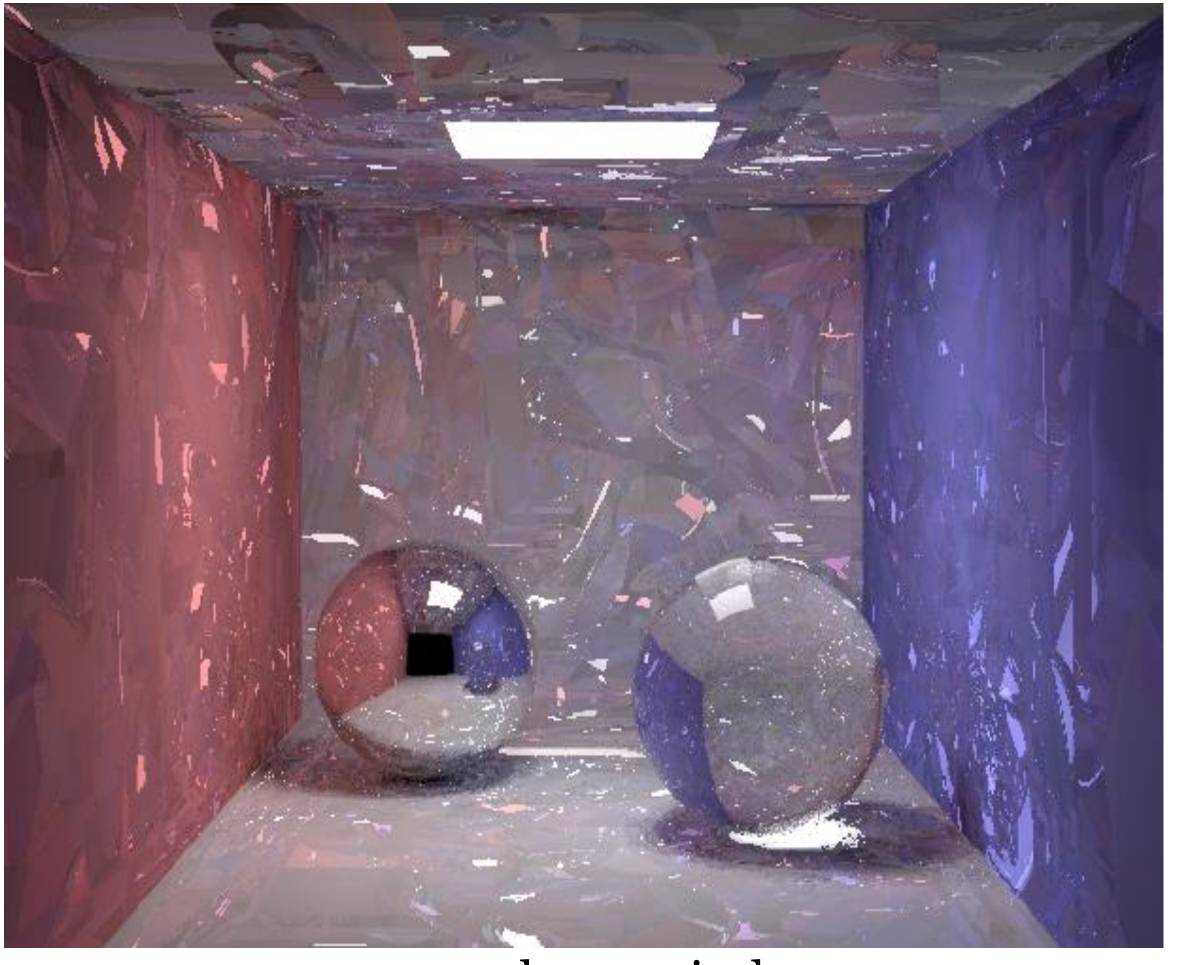
Quasi-Monte Carlo



10 paths per pixel

Henrik Wann Jensen

Same random sequence for all pixels



10 paths per pixel

Henrik Wann Jensen

Image-based lighting

Image-based lighting

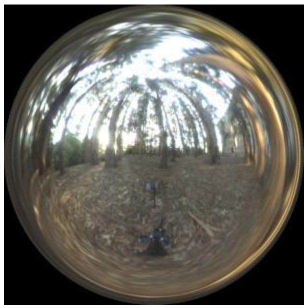
- Introduced by Paul Debevec (Siggraph 98)
- Routinely used for special effects in films & games

Environment mapping (a.k.a. image-based lighting, reflection mapping)

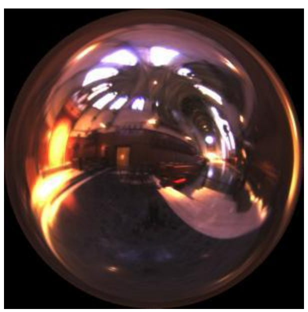
Miller and Hoffman, 1984
Later, Greene 86, Cabral et al, Debevec 97, …

Image-based lighting

- Illuminating CG objects using measurements of real light (=light probes)

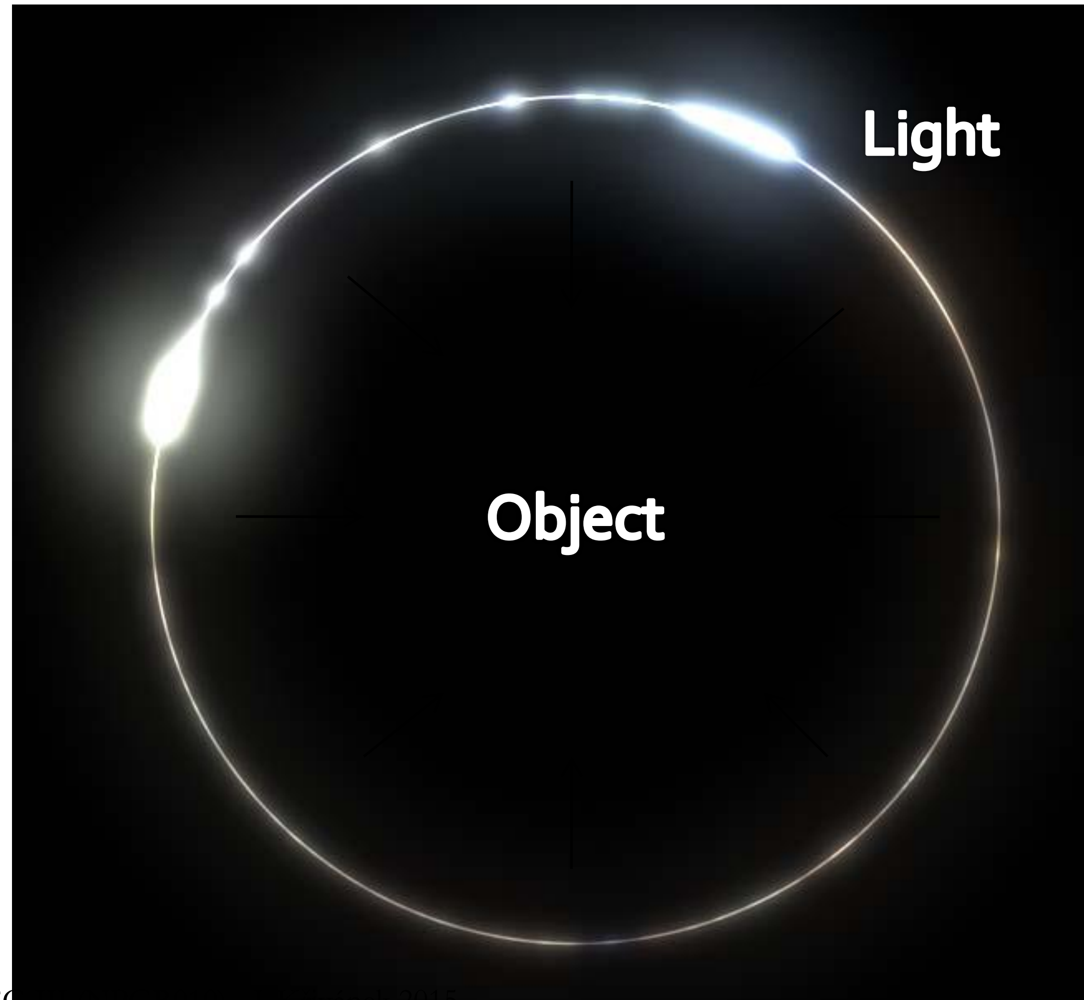


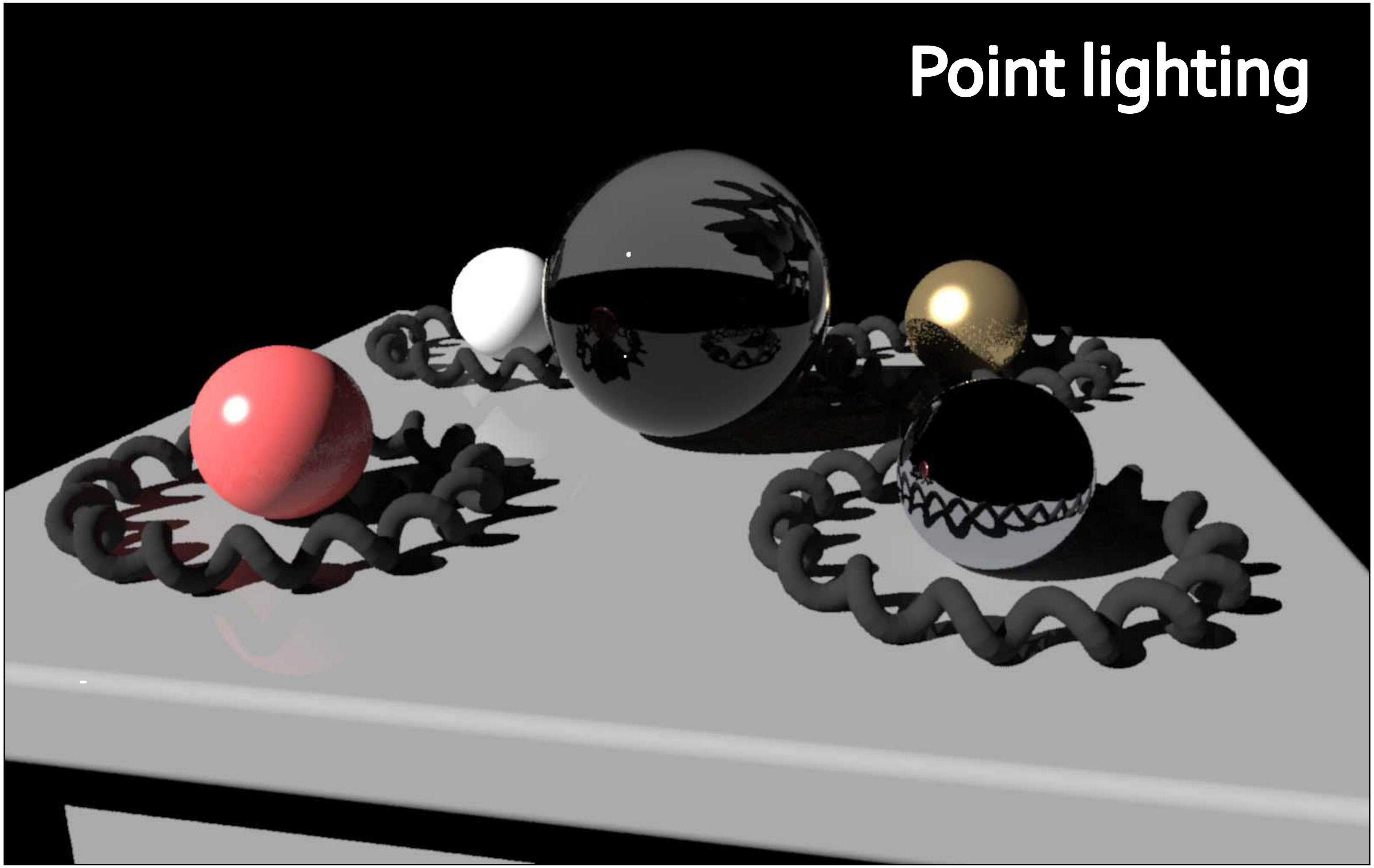
Eucaliptus grove

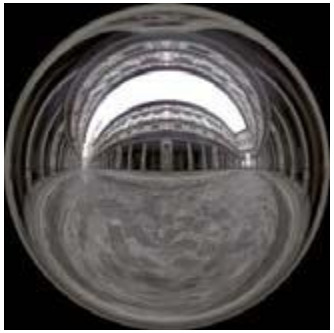


Grace cathedral

Uffizi gallery




Point lighting



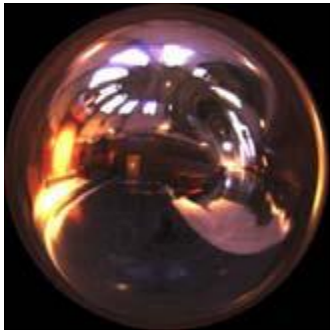
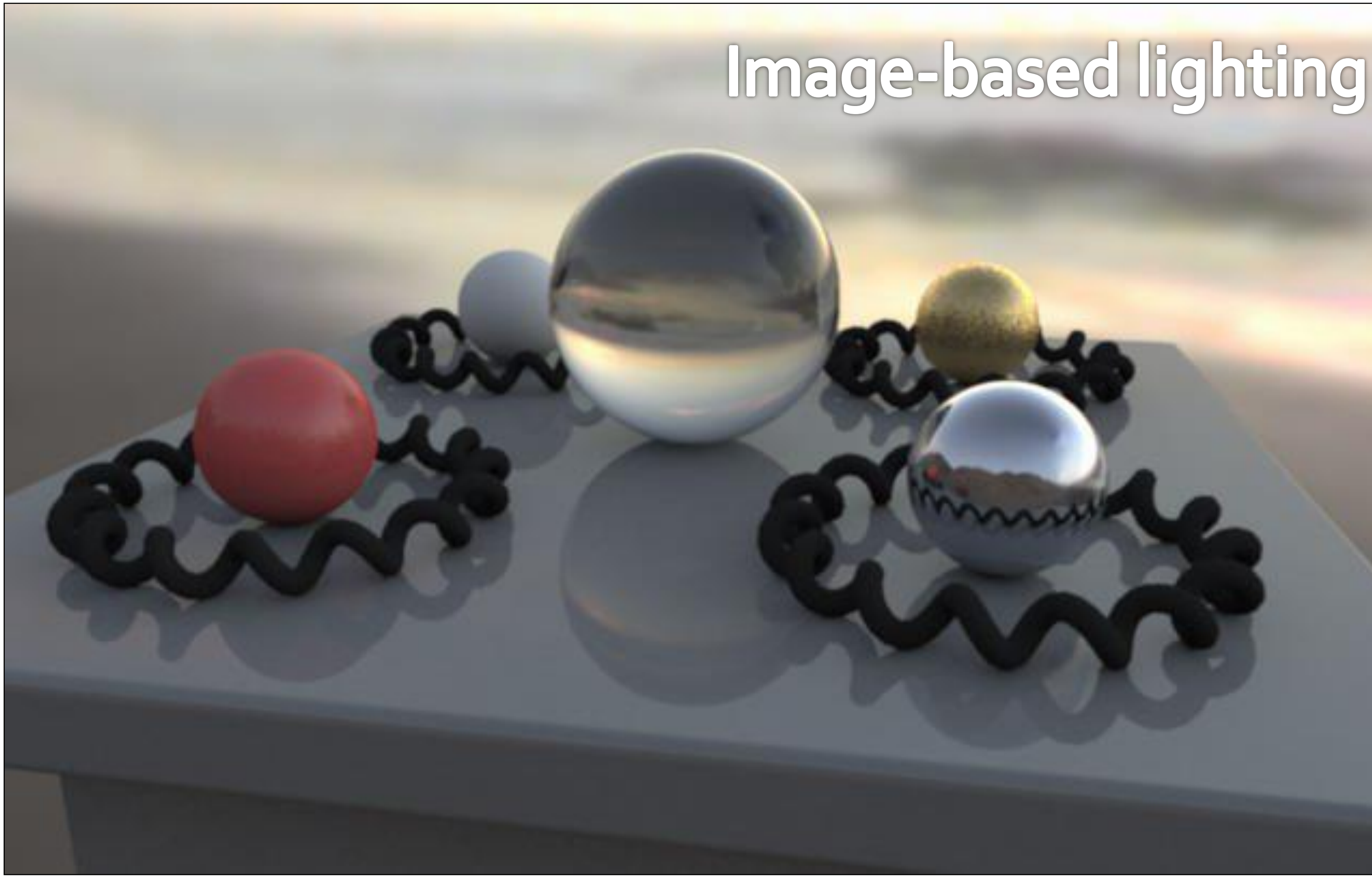


Image-based lighting



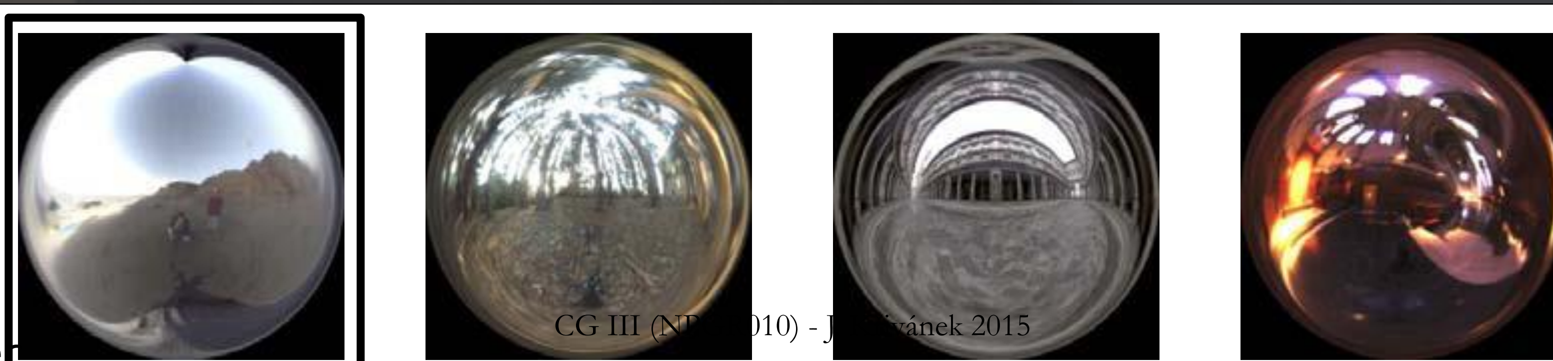
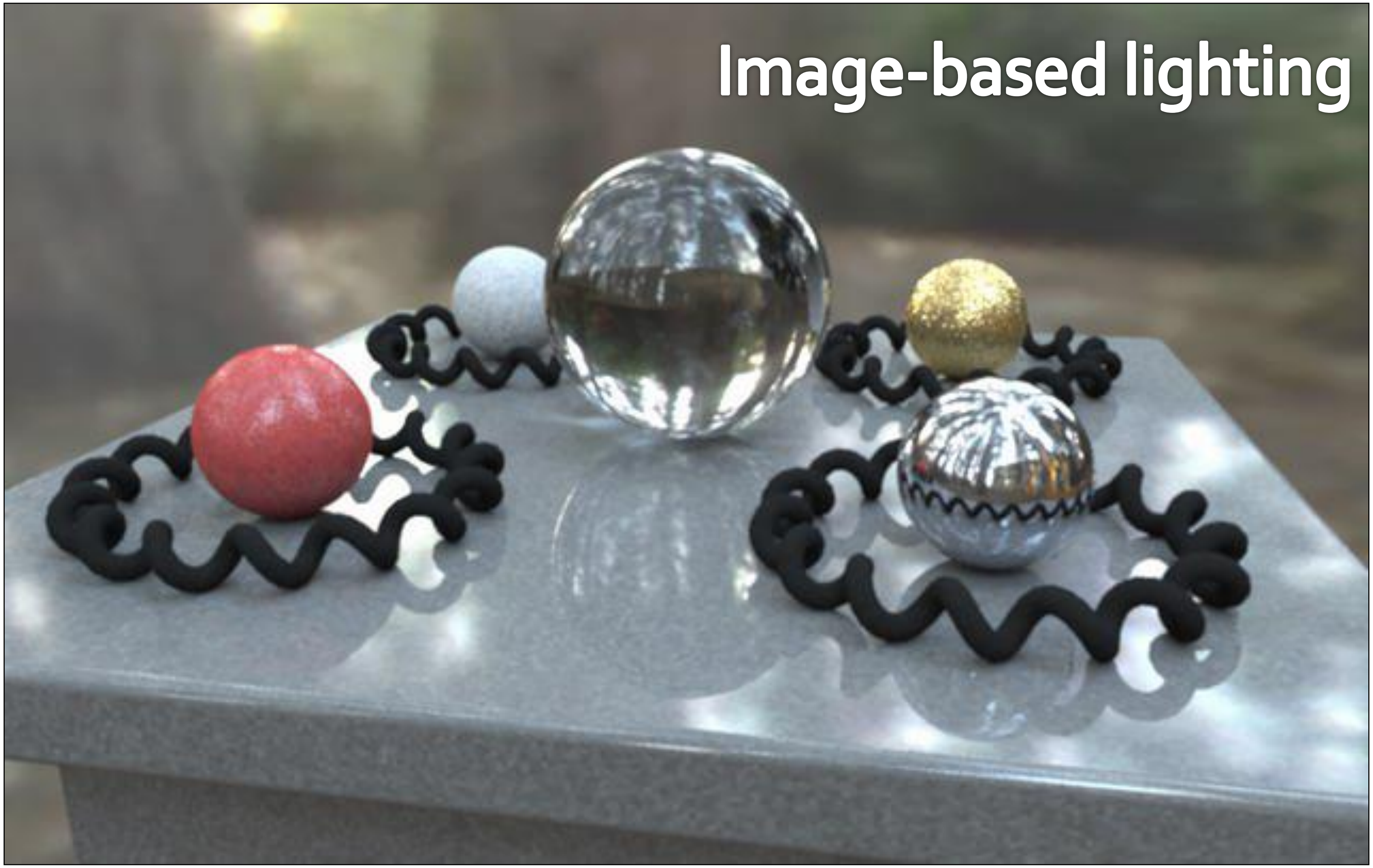
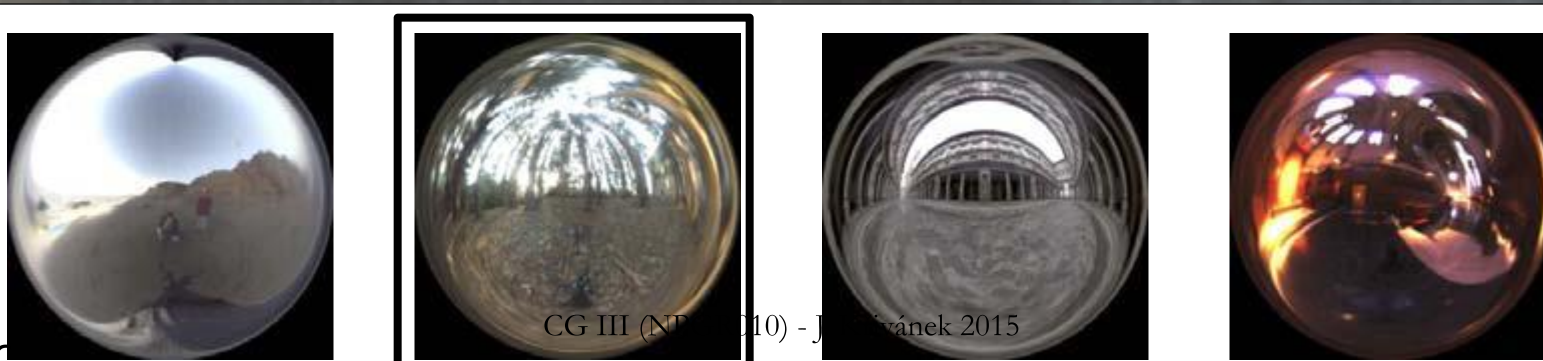


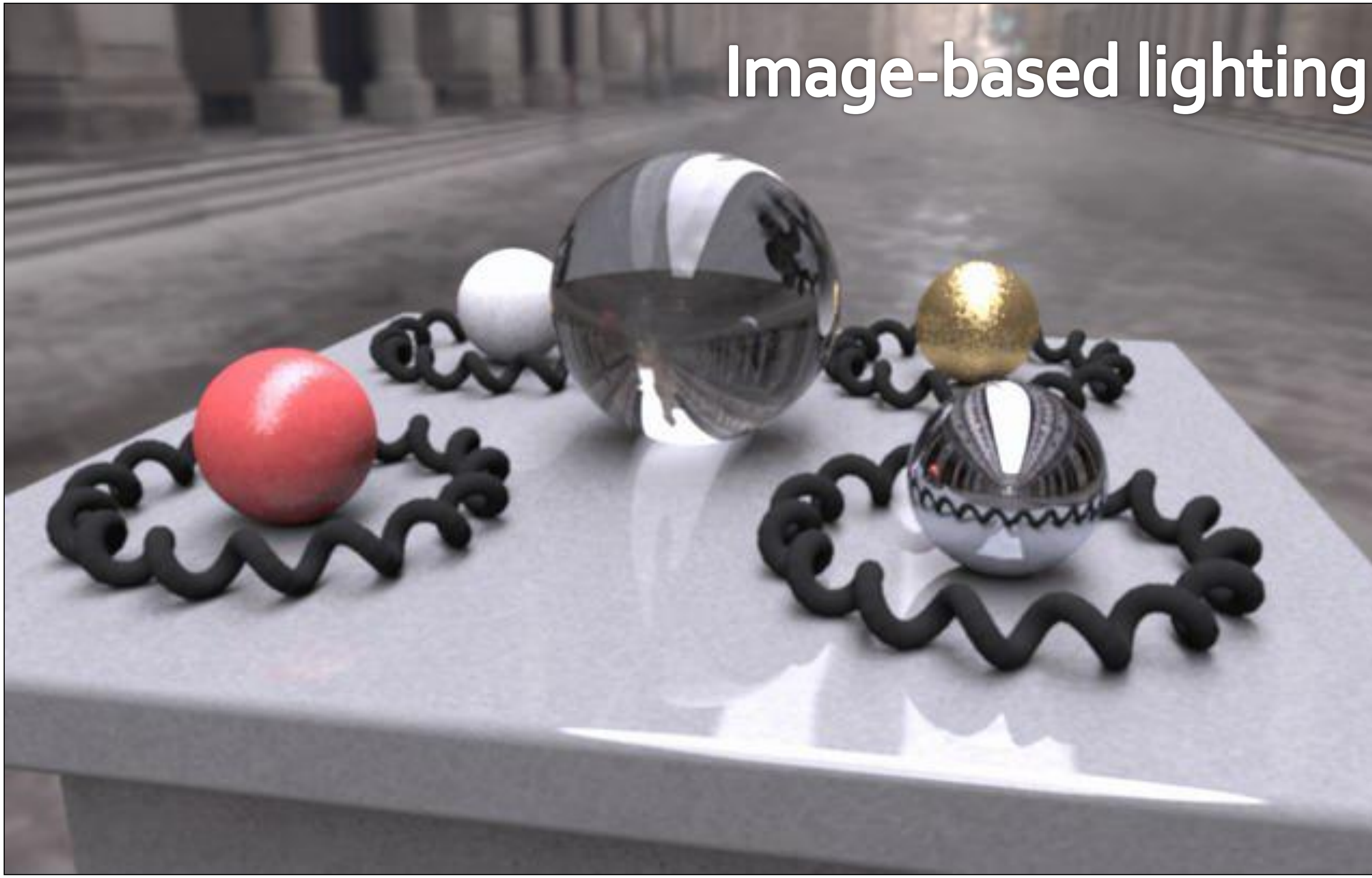
Image-based lighting

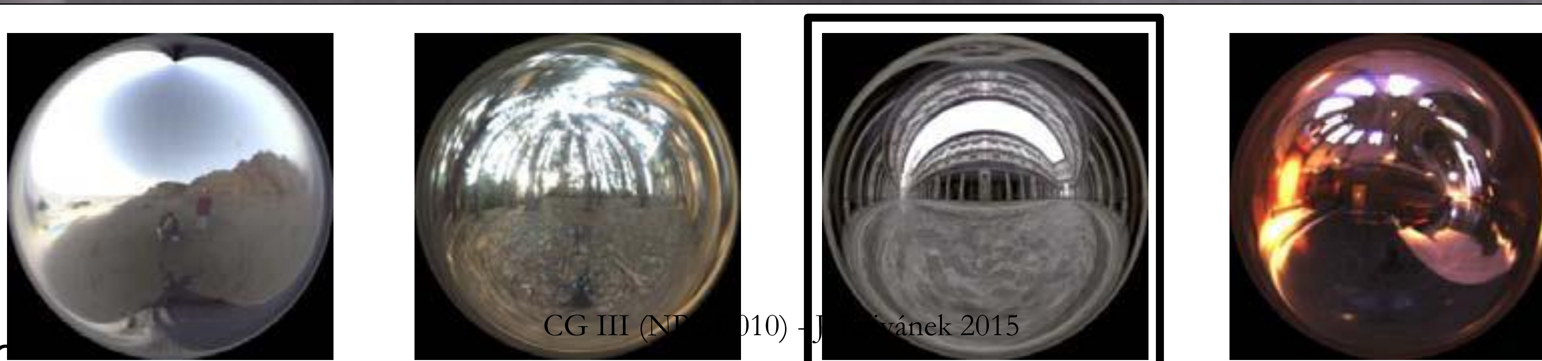




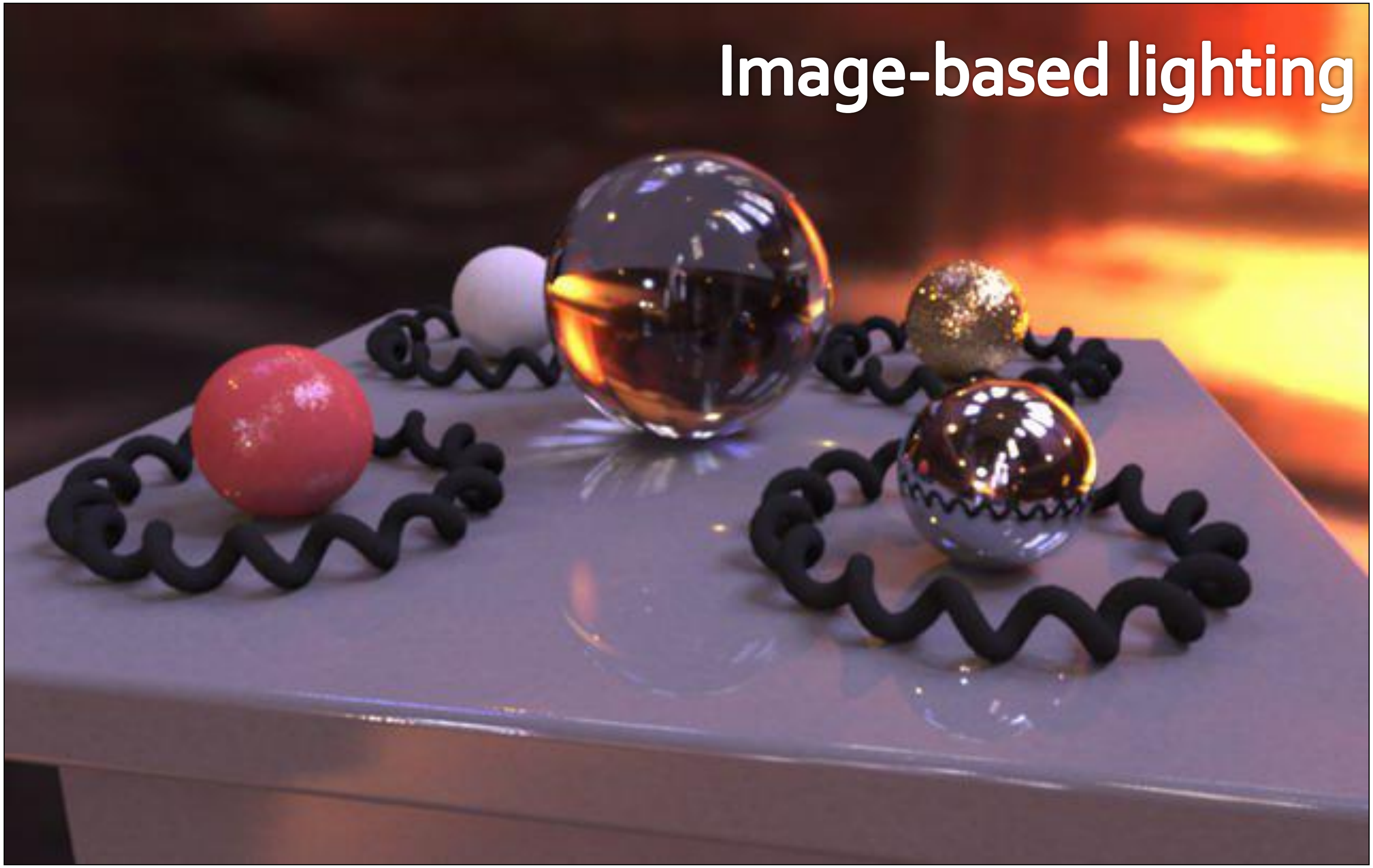
© Paul Debevec

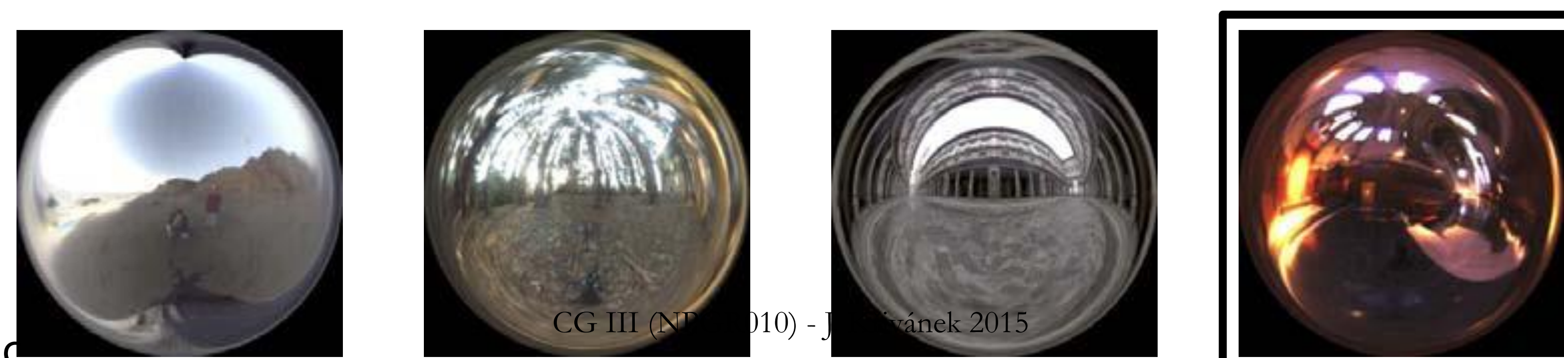
Image-based lighting





© Paul Debevec



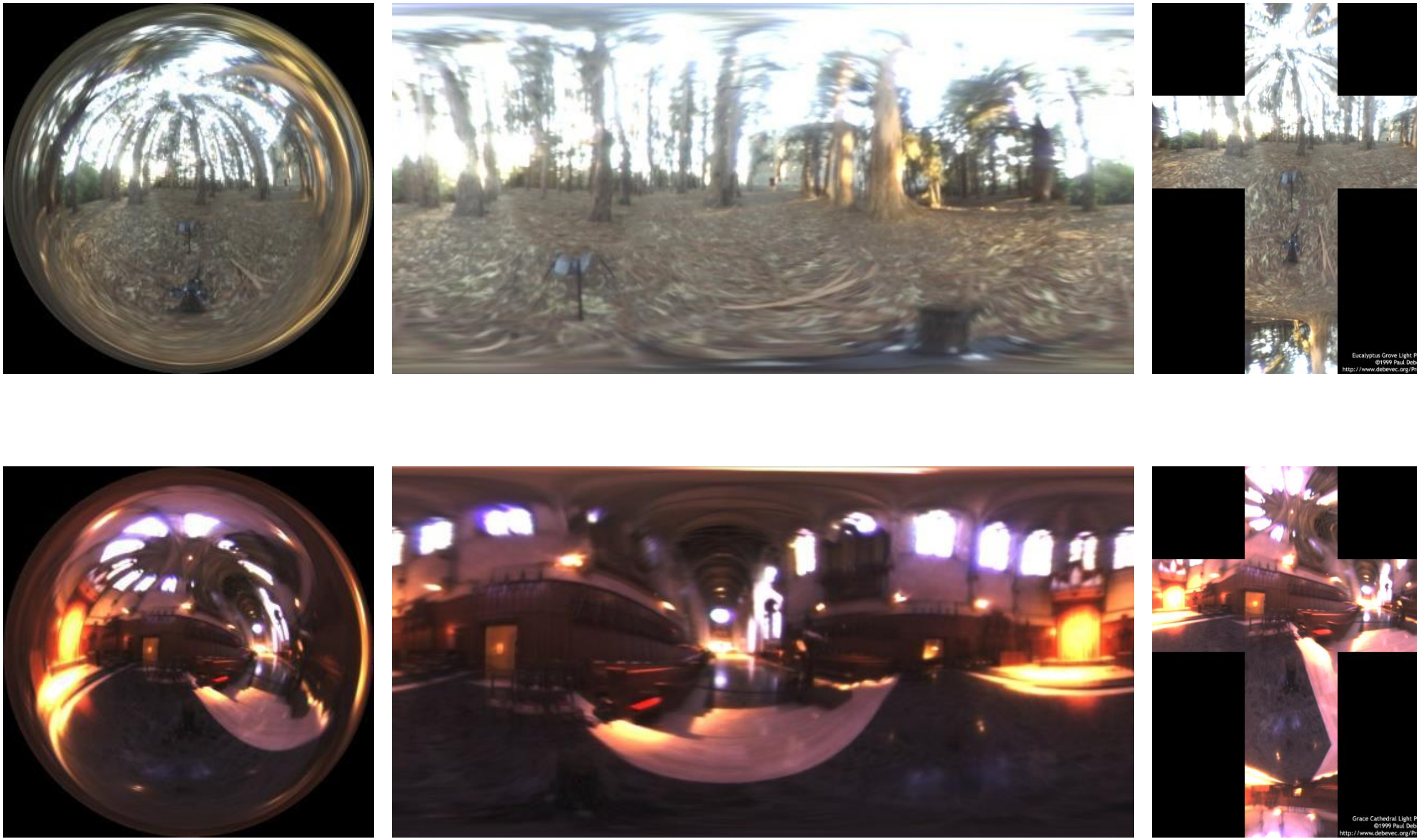


© Paul Debevec

Mapping

Eucaliptus grove

Grace cathedral

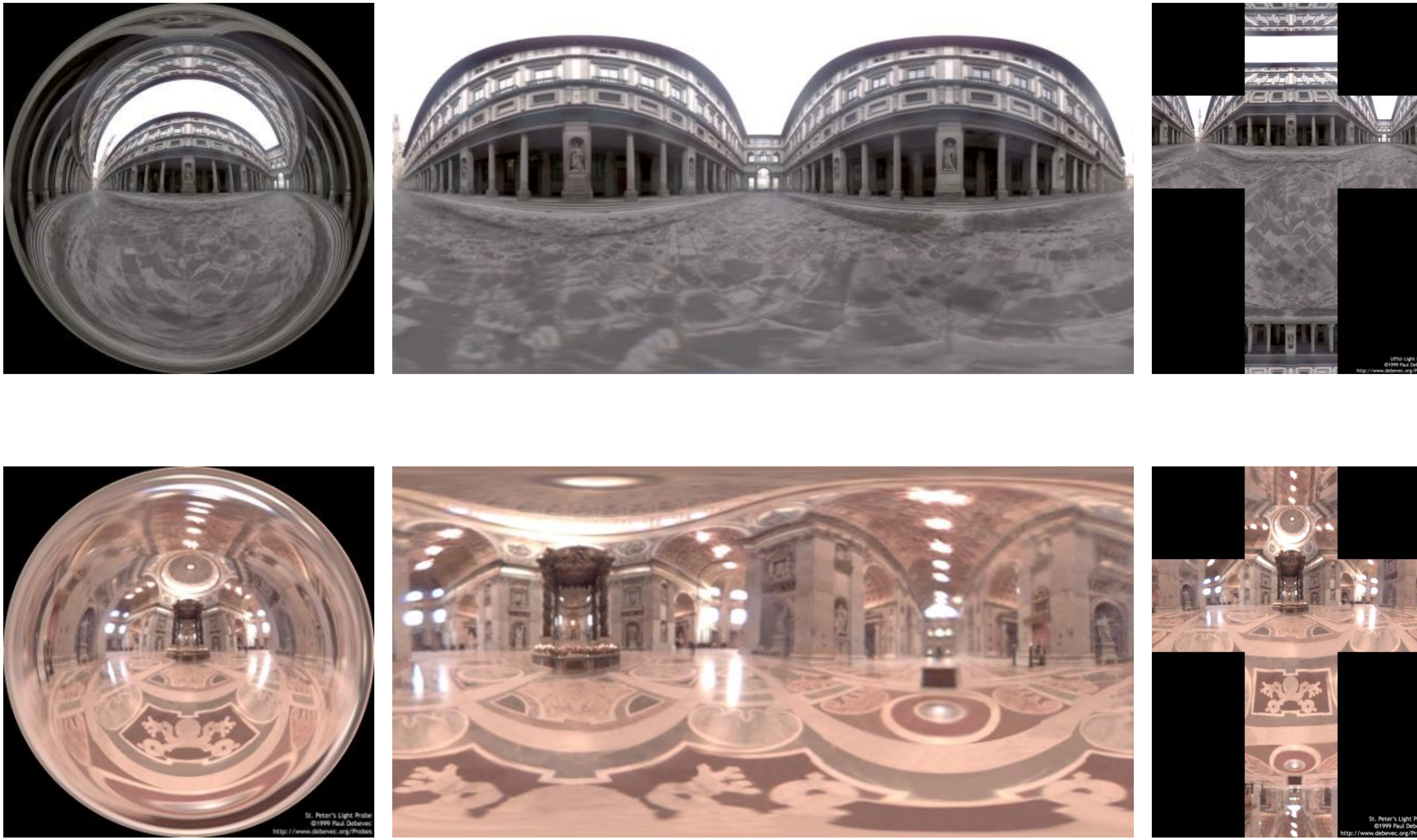


Debevec's spherical

"Latitude – longitude" (spherical coordinates)

Cube map

Mapping



Uffizi gallery

St. Peter's Cathedral

Debevec's spherical

"Latitude – longitude" (spherical coordinates)

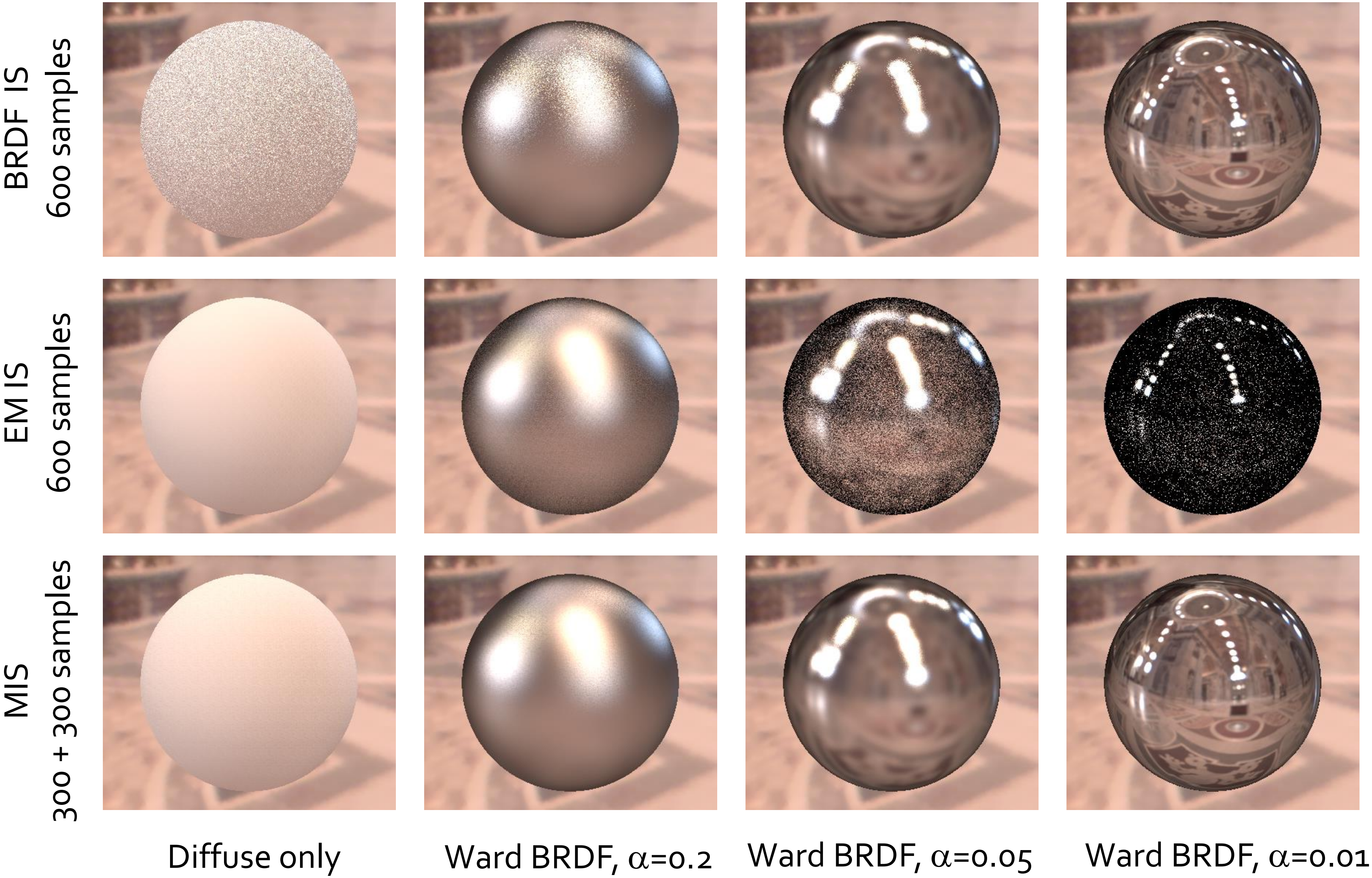
Cube map

Sampling strategies for image based lighting

- Technique (pdf) 1:
  BRDF importance sampling
  - Generate directions with a pdf proportional to the BRDF

- Technique (pdf) 2:
  Environment map importance sampling
  - Generate directions with a pdf proportional to $L(\omega)$ represented by the EM

Sampling strategies



Sampling according to the environment map luminance

- Luminance of the environment map defines the sampling pdf on the unit sphere

- For details, see PBRT, **13.6.7**
  http://www.pbr-book.org/3ed-2018/Monte_Carlo_Integration/2D_Sampling_with_Multidimensional_Transformations.html#Piecewise-Constant2DDistributions