A4M39RSO

Sledování cest (Path tracing)

Vlastimil Havran
ČVUT v Praze – CTU Prague
Verze 2014

Rendering = integrování

- Antialiasing
  - Integrál přes plochu pixelu
- Osvětlení plošným zdrojem
  - Integrál přes plochu světla
- Nepřímé osvětlení na plochách
  - Integrál přes hemisféru příchozích směrů
- Hloubka ostrosti
  - Integrál přes čočku konečné velikosti
- Rozostření pohybem
  - Integrál přes čas otevření uzávěrky

Rendering = integrování

- Všechny uvedené problémy lze řešit podobným algoritmem
  1. Vrhni mnoho náhodných paprsků
  2. Zprůměruj výsledek

- Paprsky jsou vzorky – jde o aplikaci MC integrování

- Poprvé v grafice: Cook 1984: Distributed ray tracing
  - Česky „Trasování (sledování) rozprostřených paprsků“
  - Také „Stochastic ray tracing“
- Později „Path Tracing” (1986), jeden paprsek po odrazu

Antialiasing = integrování

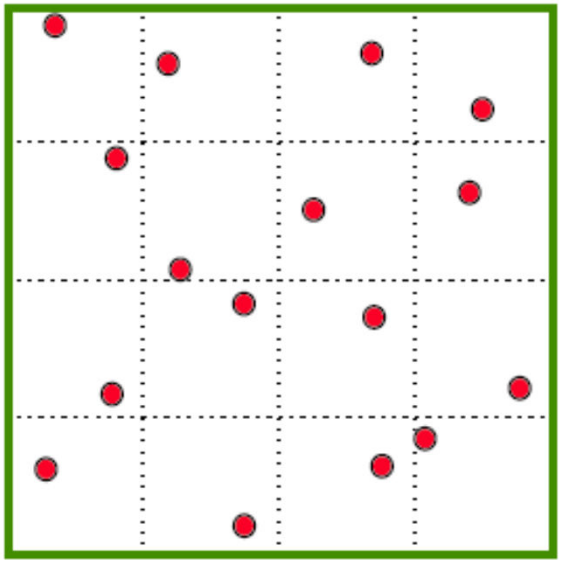
- Průměrná radiance přes plochu pixelu
  - tj . integrál radiance přes plochu pixelu dělený plochou pixelu
- Implementace MC integrováním:

```
for k = 1 to N do
   vyber náhodně pozici (x_k,y_k) uvnitř pixelu
   vrhni primární paprsek skrz (x_k,y_k) a spočti zář L(x_k,y_k)
   C += A * L(x_k,y_k) * w(x_k,y_k)   /* A je zde kvůli normalizaci */
C /= N * A
```

- C je výsledná barva pixelu
- *A* je plocha pixelu (lze ji vykrátit – z algoritmu zmizí)
- *w* je pre-filtr
  - Často w=1, tzv. box filtr
  - Lepší výsledky dává např. filtr ve tvaru Gaussiánu
  - Kvalitní filtry jsou větší než 1 pixel
    - vzorek z jednoho pixelu přispívá i k sousedním pixelům
    - složitější algoritmus

Antialiasing = integrování

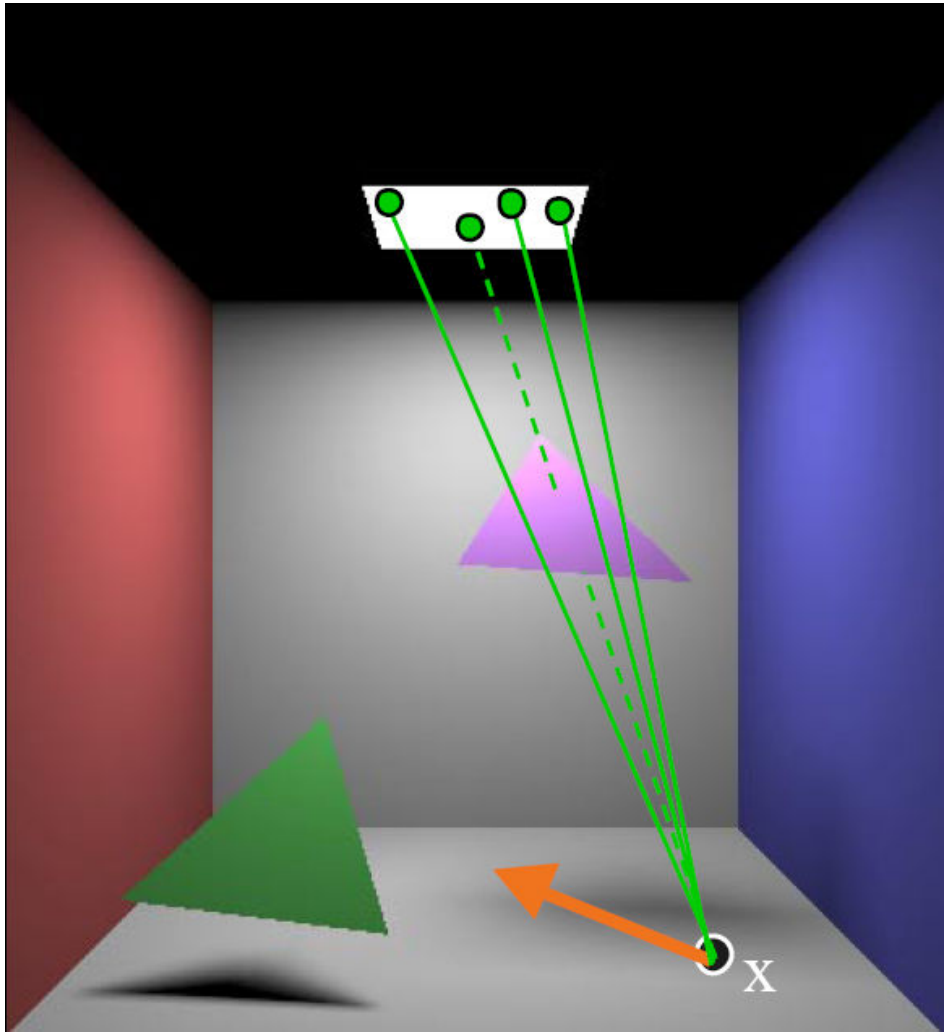
- Snížení variance
  - aplikace MC techniky „výběr na podintervalech“ (**stratified sampling**)
  - v kontextu vzorkování roviny obrazu též nazýváno **jittering**



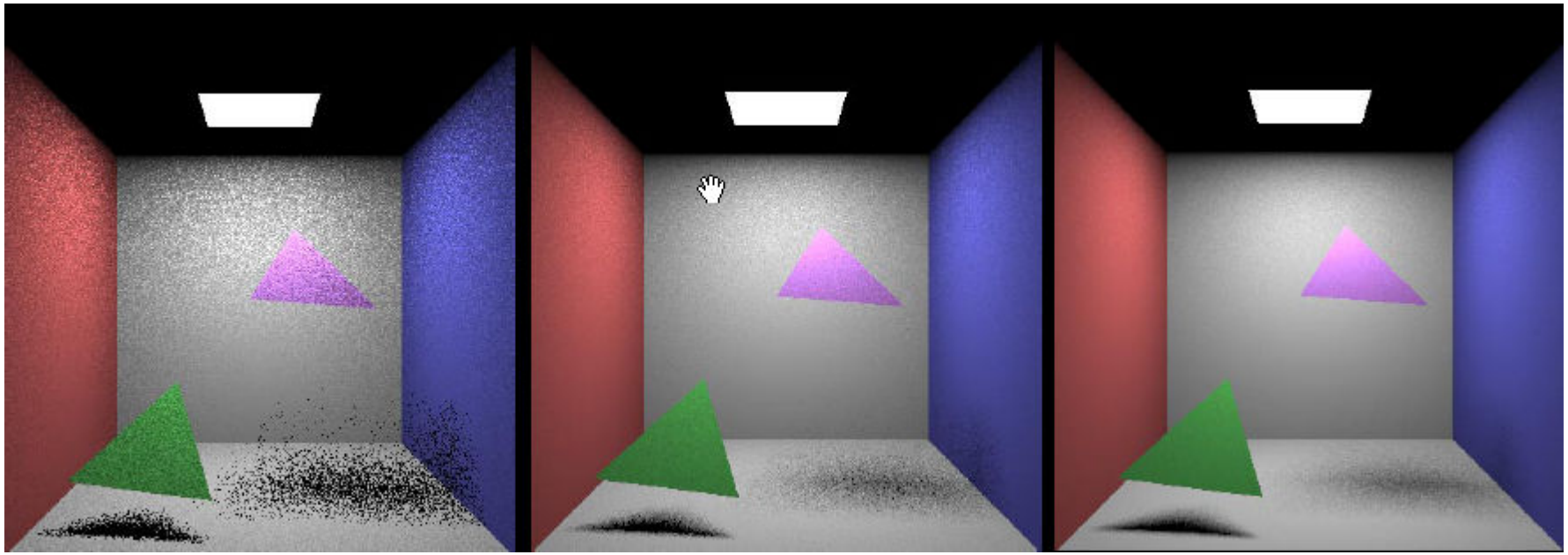
Plošné zdroje světla = integrování

- Osvětlení plošným zdrojem světla
  - Integrál přes plochu světla
- Úloha se převádí na MC integrování:
  - Rozmísti body po povrchu světla
  - Spočti viditelnost a vzdálenost k bodu osvětlení
  - Spoči průměr hodnot záře (angl. radiance)
- Snížení variance
  - výběr na podintervalech (stratified sampling) na povrchu světla
- Pozor! Pro každý osvětlovaný pixel vyberu na světelném zdroji jiné náhodné vzorky, jinak vzniká systematická a viditelná nízkofrekvenční chyba!

Plošné zdroje světla = integrování



Plošné zdroje světla = integrování

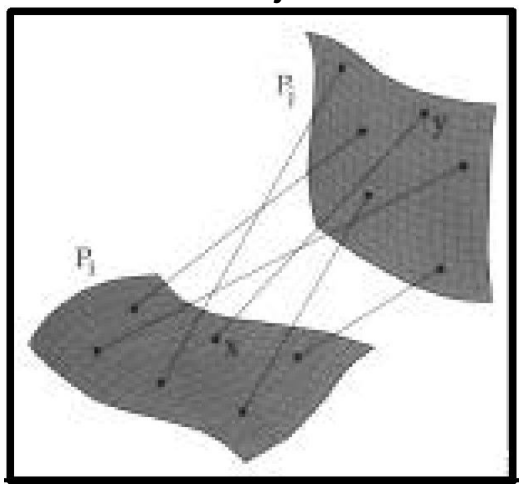


1 vzorek na pixel 9 vzorků na pixel 36 vzorků na pixel

Výpočet form faktorů pro radiozitu = integrování

- Form faktor
  - kolik procent energie vyzářené z plochy A_i dopadne na A_j
- Analytický tvar

$$F_{ij} = \frac{1}{A_i} \int_{A_i} \int_{A_j} \frac{\cos\theta_x \cos\theta_y}{\pi r^2} V(\mathbf{x},\mathbf{y}) dA_j dA_i$$



- Implementace MC integrováním:

```
for k = 1 to N do
   Vyber náhodný bod x_k na A_i a bod y_k na A_j
   Spočti viditelnost V(x_k,,y_k)        /* vržení stínového paprsku */
   if (V(x_k,,y_k)==1)
      F_ij += A_i A_j cos q_x cos q_y / p r^2  /* násobení A_i A_j kvůli *
F_ij /= N * A_i                                /* normalizaci */
```

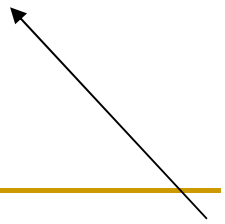
Rovnice odrazu = integrování

- "Kolik světla je odraženo do směru ω_o, je-li bod **x** osvětlen ze všech příchozích směrů?"
- Integrál přes všechny příchozí směry:

$$L_o(\mathbf{x},\omega_o) = \int_{\Omega} L_i(\mathbf{x},\omega_i) f_r(\omega_i,\mathbf{x},\omega_o)\cos\theta_i d\omega_i$$

- Implementace MC integrováním

```
outgoing_radiance(x, n, wo) {
   Color Lo=(0,0,0);
   for i=1 to N {
      vygeneruj náhodný směr na hemisféře s p(wi)
      Lo += trace_ray(x,wi)*BRDF(wo,x,wi)*dot(n,wi)/p(wi)
   }
   return Lo / N;
}
```



rekurze

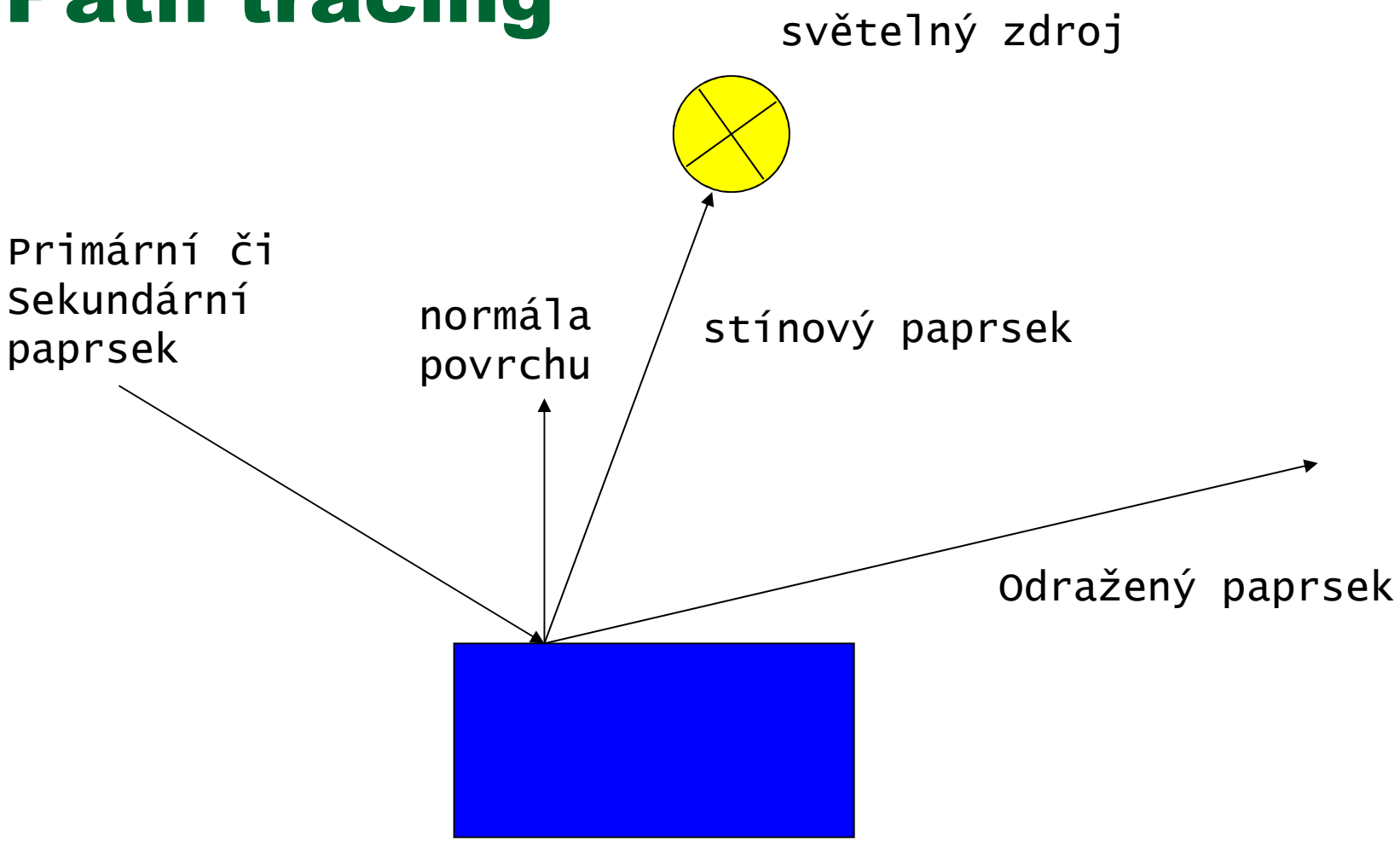
Rovnice odrazu = integrování

- Snížení variance
  - Výběr na podintervalech (stratified sampling)
    - rozdělení hemisféry na buňky, v každé buňce jeden paprsek
  - Podstatný výběr (importance sampling)
    - hodně paprsků ve směru laloku BRDF
    - tj. hustota pravděpodobnosti pro vzorkování = normalizovaná BRDF
    - dělení hustotou často vykrátí BRDF ve vnitřním cyklu
    - Bude popsáno detailně pro Phongovu BRDF

Path tracing

- Rekurze – obrovské množství paprsků na hlubších úrovních
- Neefektivní
- Path tracing – česky Trasování cest, anglický název je obvyklejší.
  - vždy vybere pouze jeden sekundární paprsek
    - nejdříve se vybere způsob interakce (ideální lom, ideální odraz, difúzní odraz, …)
    - pak se použije importance sampling podle vybrané interakce
  - přímé osvětlení – vyber náhodně jeden vzorek na jednom zdroji světla pro každý odraz.
- Pokud odražený paprsek zasáhne zdroj světla, příspěvek emitovaný ze světla se nezapočítá.

Path tracing

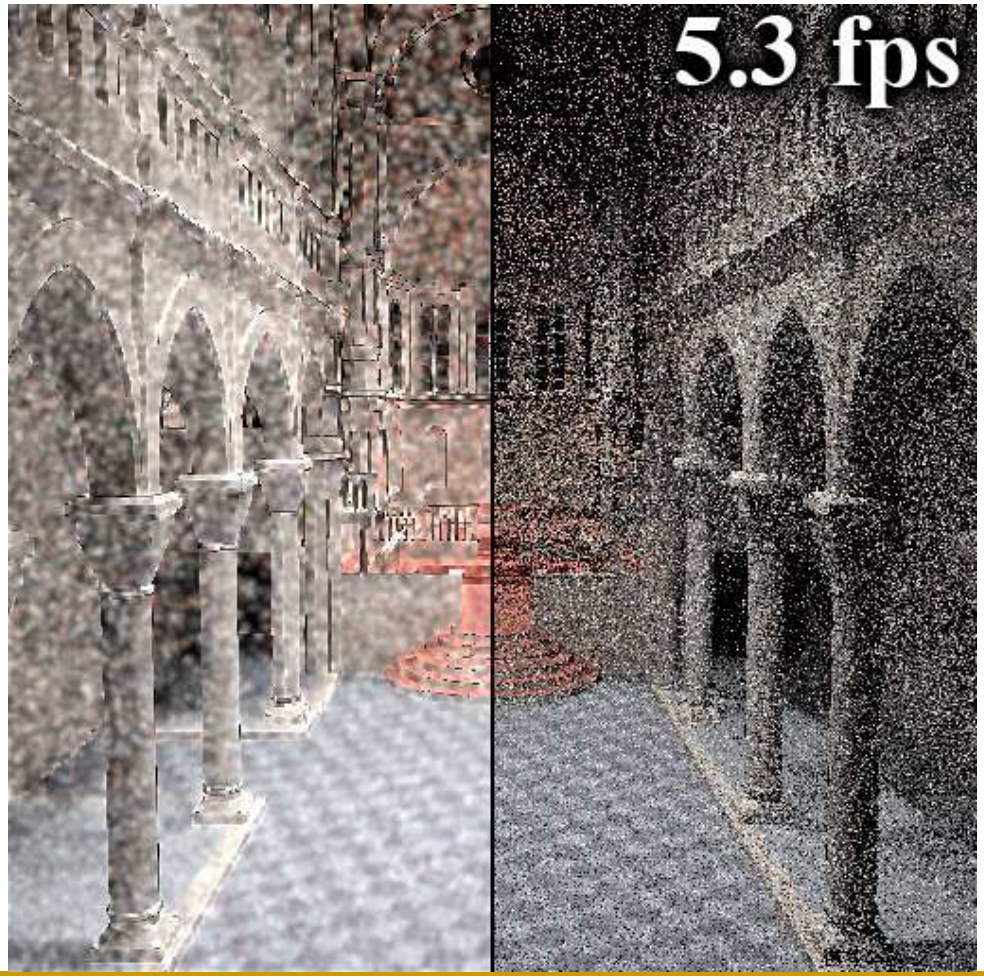


Path tracing

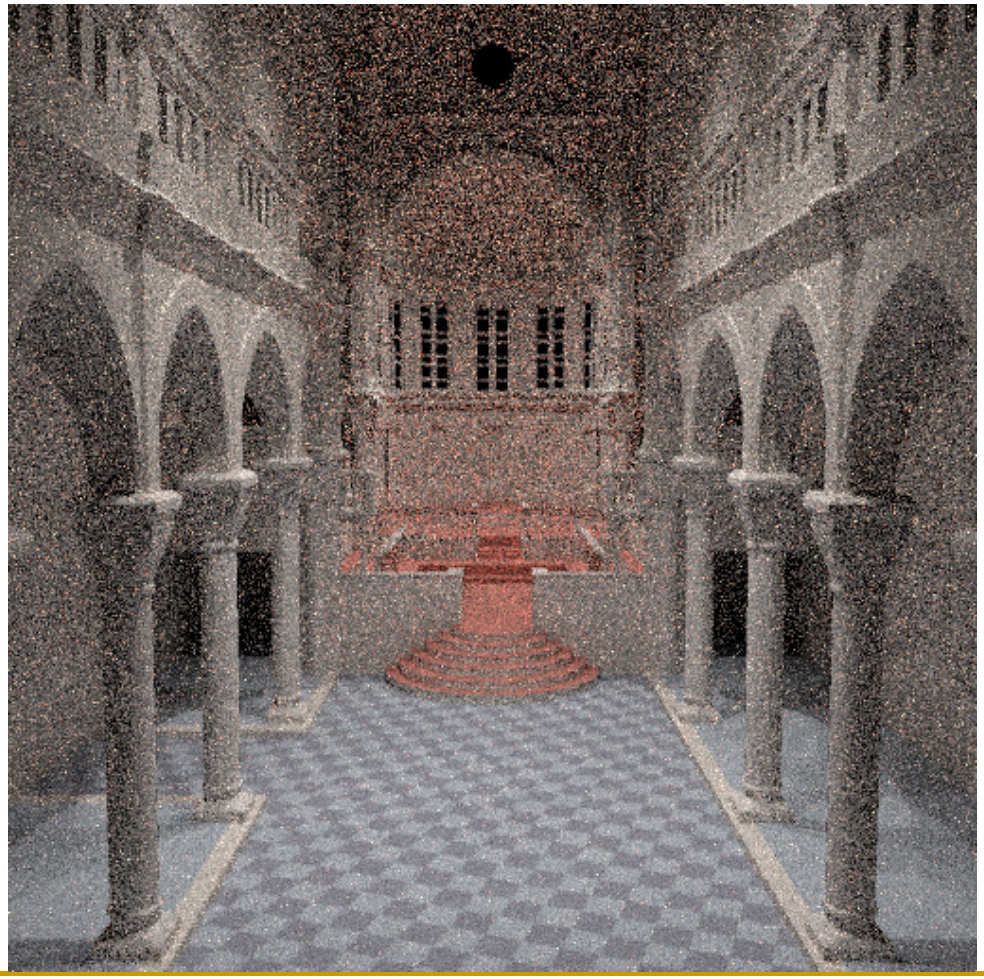
- Trasuj tisíce cest přes každý pixel a zprůměruj výsledek
- Výhoda: žádná exploze počtu paprsků kvůli rekurzi
- Používá se jako **referenční algoritmus** pro jiné – trvá dlouho, ale je jednoduchý na implementaci a poskytuje korektní výsledek s větším nebo menším šumem.
- Pro obrázek bez viditelného šumu není neobvyklé, že pro pixely je nutné vrhnout 10^5 až 10^6 cest.
- GPU implementace existují, například článek: *Path Regeneration for Interactive Path Tracing*, 2010/2011, Novák, Havran, Dachsbacher.

  http://dcgi.felk.cvut.cz/home/havran/ARTICLES/eg2010_pt.pdf + implementace diplomové práce pana Nováka.

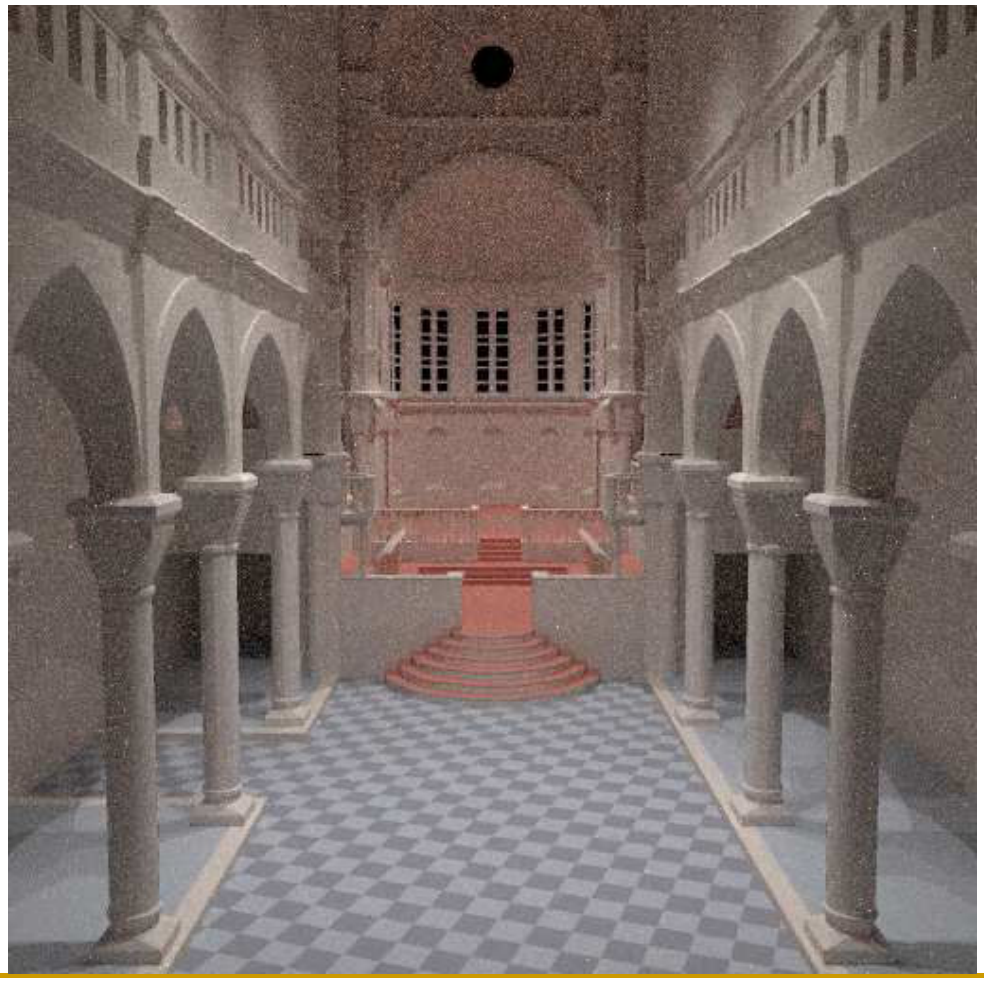
Path Tracing – 0,2 sekundy + filtrování (levá strana)



Path Tracing - 1 sekund na GPU.



Path Tracing - 10 sekund na GPU.



Importance sampling Phongovy BRDF

Importance sampling Phongovy BRDF

- Paprsek dopadne na plochu s Phongovou BRDF. Jak vygenerovat sekundární paprsek pro nepřímého osvětlení?
- Klasický ray tracing
  - 1 paprsek pro ideální zrcadlový odraz
  - 1 paprsek pro ideální zrcadlový lom
- Path tracing
  - Bere v úvahu nejenom ideální zrcadlový odraz/lom
  - Vrhá pouze 1 sekundární paprsek – je třeba zvolit interakci
  - Algoritmus:
    1. vyber interakci (absorbce / difůzní odraz / lesklý odraz / lom)
    2. pokud nevybrána absorbce, vzorkuj vybranou interakci

Fyzikálně věrohodná Phongova BRDF

$$f_r(\omega_i, x, \omega_o) = \frac{\rho_d}{\pi} + \frac{n+2}{2\pi}\rho_s \cos^n \theta_r$$

- Kde:

$$\cos\theta_r = \omega_o \cdot \omega_r$$

$$\omega_r = 2(\omega_i \cdot \mathbf{n})\mathbf{n} - \omega_i$$

- Zachování energie: $\rho_d + \rho_s \leq 1$

- Funkce důležitosti je BRDF * cos(θ)

Výběr interakce

```
pd = max(rho_d.r, rho_d.g, rho_d.b); // probability of d
ps = max(rho_s.r, rho_s.g, rho_s.b); // probability of s
xi = rand(0,max(1,pd+ps)); // in case pd+ps>1
if (xi < rd)
   L += 1/pd * SampleDiffuse();     // 1/rd = 1/prob
else if (xi < rd+rs)
   L += 1/ps * SampleSpecular();    // 1/rs = 1/prob
// else L += 0;  (russian roulette - absorption)
```

Vzorkování difúzního odrazu

- Importance sampling s hustotou $p(\theta) = \cos(\theta) / \pi$
  - θ…úhel mezi normálou a vygenerovaným sekundárním paprskem
  - Generování směru:

$$\varphi = 2\pi r_1$$

$$\theta = \operatorname{acos}(\sqrt{r_2})$$

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

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

$$z = \sqrt{r_2}$$

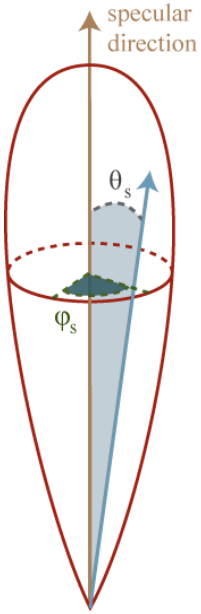
    - r1, r2 … uniformní na <0,1>
  - Zdroj: Dutre, Global illumination Compendium (on-line)
  - Odvození: Pharr, Physically Based Rendering Toolkit.
  - A taky: Eric Lafortune and Y.D.Willems: Using the Modified Phong BRDF for Physically Based Rendering, report 1994

SampleDiffuse()

```
// build the local coordinate frame with N = z-axis
Vector3D U = ArbitraryNormal(N);// U is perpendicular to the normal N
Vector3D V = CrossProd(N,U);    // orthonormal base with N and U

// generate direction in the local coordinate frame
float r1 = rand(0,1), r2 = rand(0,1);
float sin_theta = sqrt(1 - r1);
float cos_theta = sqrt(r1);
float phi       = 2.0*M_PI*r2;
float prob      = cos_theta/M_PI;
Vector3D dir (cos(phi)*sin_theta, sin(phi)*sin_theta, cos_theta);

// transform to global coord frame
Vector3D gdir = dir.x * U + dir.y * V + dir.z * Z

// trace ray
Color Li = TraceRay(x,gdir);
return Li * rho_d / prob;
                    // importance sampling -> division by prob
```

Vzorkování lesklého odrazu

- Importance sampling s hustotou $p(\theta) = (n+1)/(2\pi)\cos^n(\theta)$
  - θ…úhel mezi ideálně zrcadlově odraženým ω_o a vygenerovaným sekundárním paprskem
  - Generování směru:



$$\varphi = 2\pi r_1$$

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

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

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

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

- r1, r2 … uniformní na <0,1>

SampleSpecular()

```
// build the local coordinate frame with R = z-axis
Vector3D R = 2*dot(N,wi)*N - wi; // ideal reflected dir
Vector3D U = ArbitraryNormal(R);     // U is perpendicular to R
Vector3D V = CrossProd(R,U);         // orthonormal base with R and U

// generate direction
float prob = hemisph_cosn(dir, n); // formulas form prev slide
Vector3D gdir = dir.x * U + dir.y * V + dir.z * Z

// reject if under the horizon
float cos_theta_i = dot(N, gdir);
if(cos_theta_i<=0)  return Color(0,0,0);

// trace ray
Color Li = TraceRay(x,gdir);
// evaluate the BRDF
Color fr = rho_s * (n+2)/(M_PI*2) * pow(dir.z, n); //dir.z=cos_theta_r
return Li * fr * cos_theta_i / prob;   // see the reflection equation
```

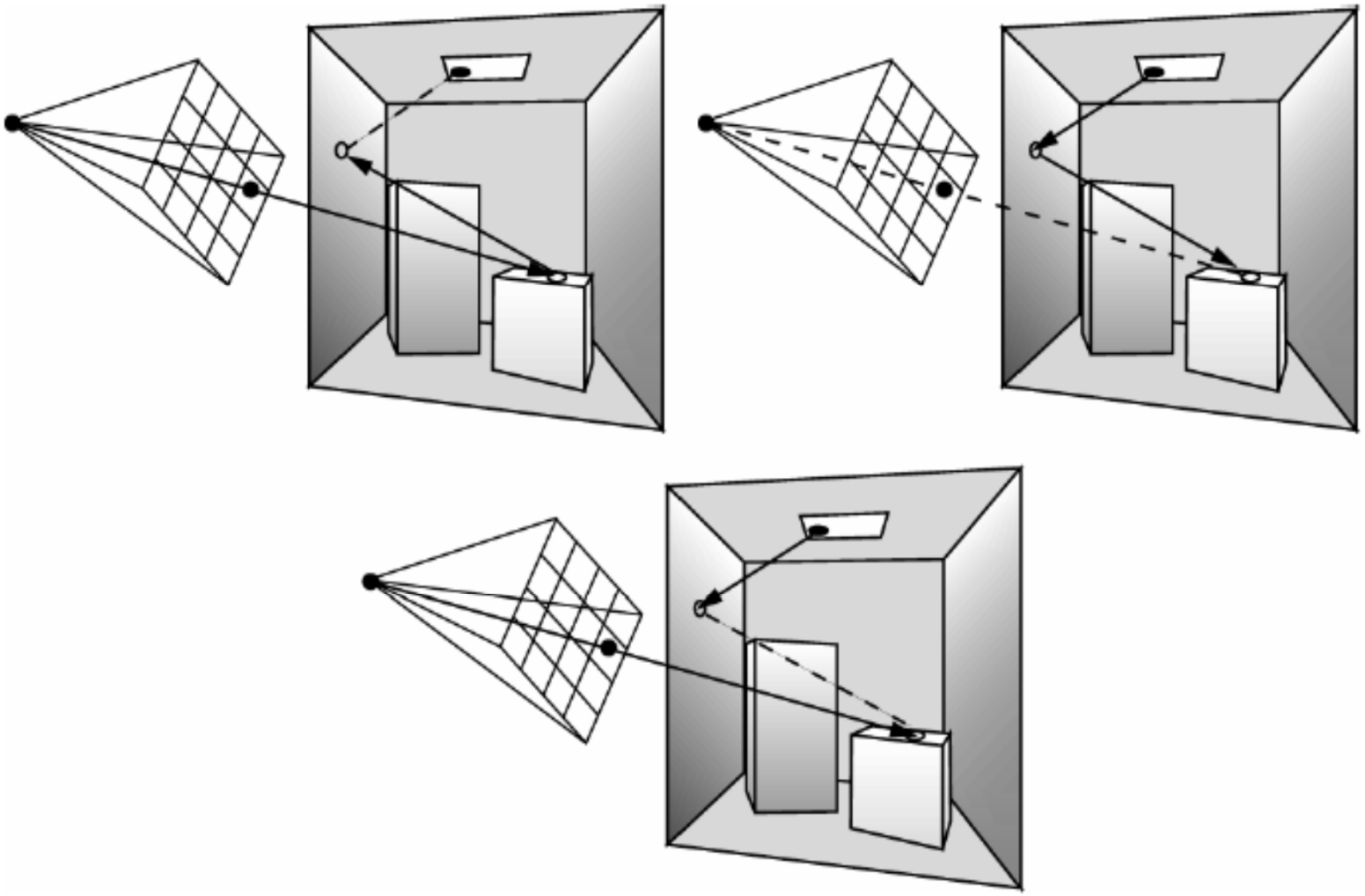
Bidirectional path tracing

2 varianty metody

1) Eric Lafortune and Y.D. Willems: Bidirectional Path Tracing, 1993

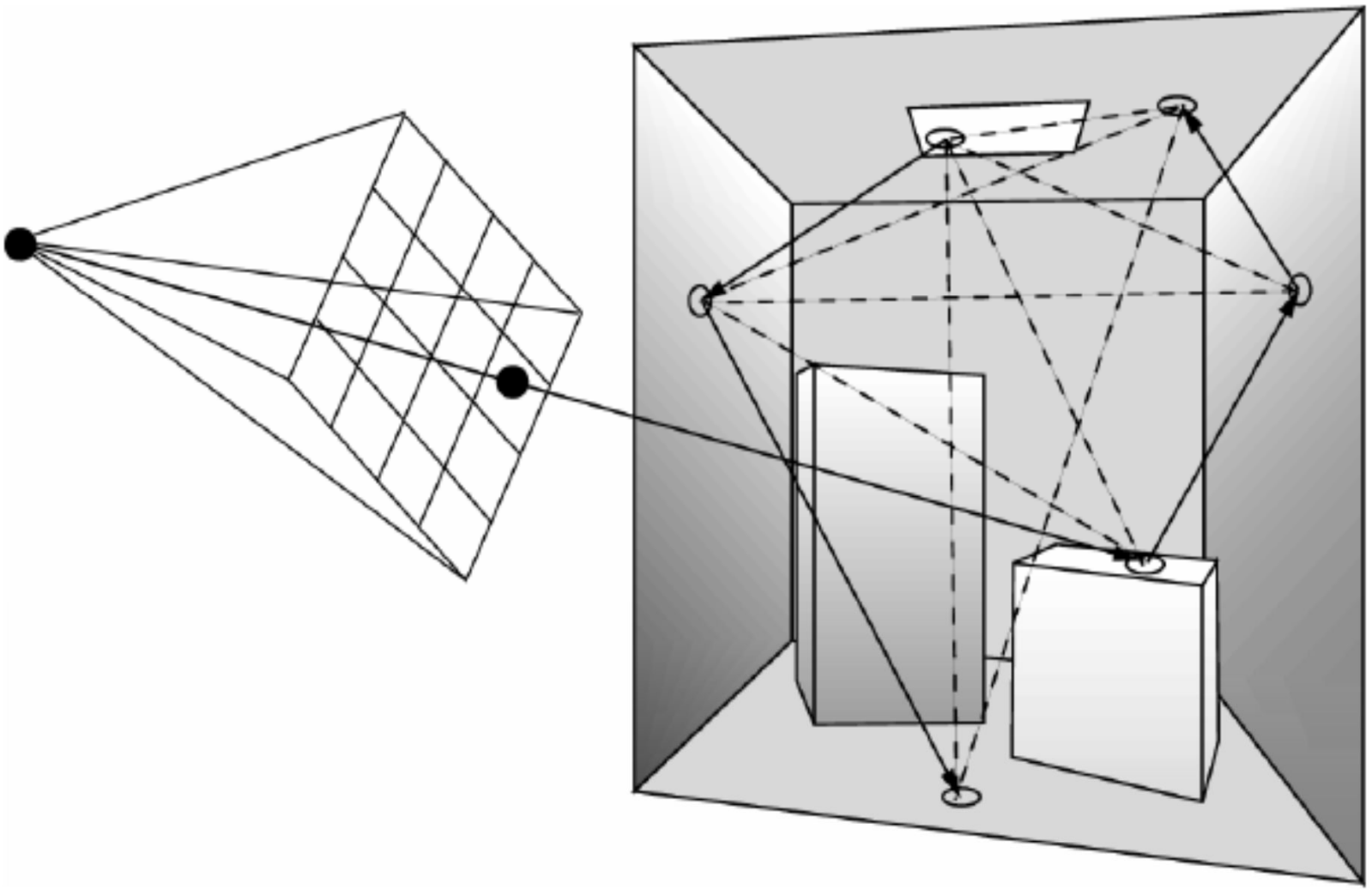
2) Eric Veach and L.J. Guibas: Bidirectional Estimators for Light Transport, 1994

Bi-directional path tracing



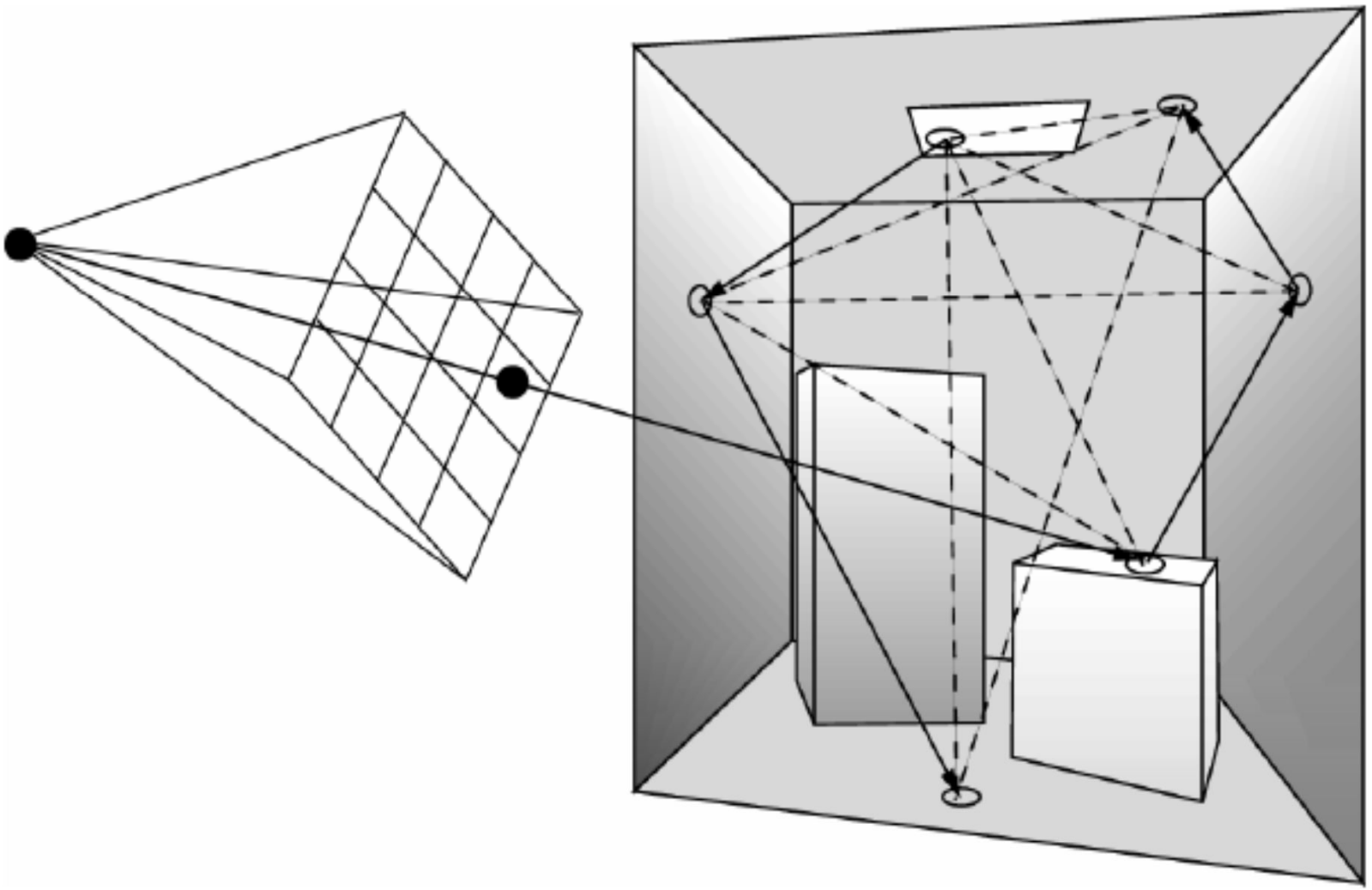
[Dutré, Bekaert, Bala]

Bi-directional path tracing



[Dutré, Bekaert, Bala]

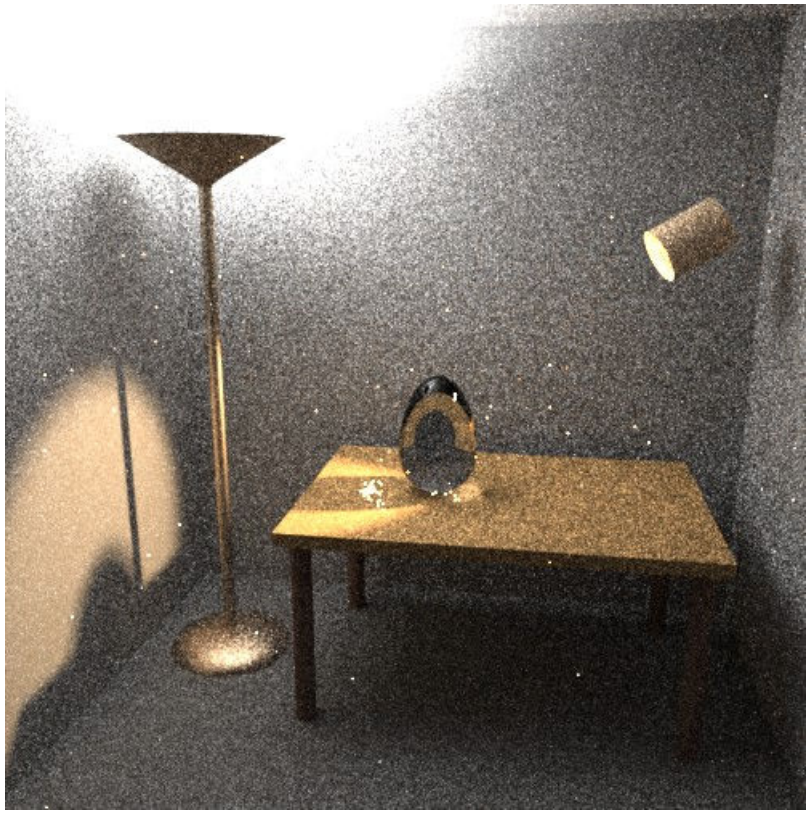
Bi-directional path tracing



[Dutré, Bekaert, Bala]

Comparison

Bidirectional path tracing



Path tracing

From Veach and Guibas, 1995

Path Pyramid

From Veach and Guibas