Počítačová grafika III – Path tracing

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

Zobrazovací rovnice – Rendering equation

$$L(\mathbf{x}, \omega_o) = L_e(\mathbf{x}, \omega_o) + \int_{H(\mathbf{x})} L(\mathrm{r}(\mathbf{x}, \omega_i), -\omega_i) \cdot f_r(\mathbf{x}, \omega_i \rightarrow \omega_o) \cdot \cos\theta_i \, \mathrm{d}\omega_i$$

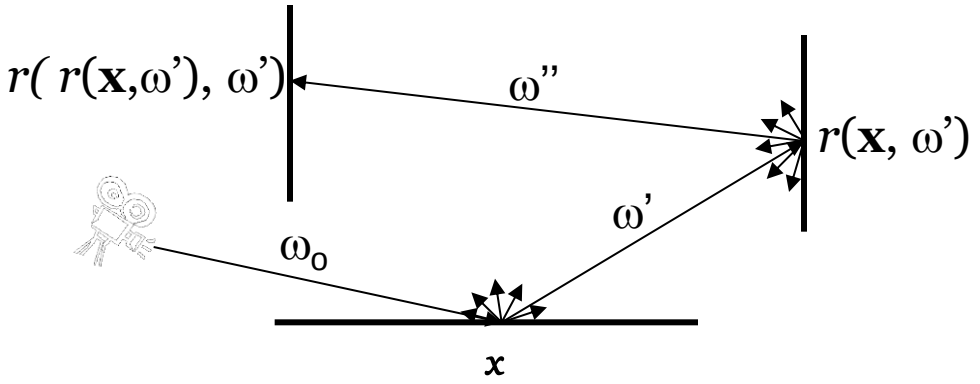
- Popis ustáleného stavu = **energetické rovnováhy** ve scéně.
- **Rendering** = výpočet $L(\mathbf{x},\omega_o)$ pro místa viditelná přes pixely.

Rekurzivní interpretace

- Úhlová formulace ZR

$$L(\mathbf{x}, \omega_o) = L_e(\mathbf{x}, \omega_o) + \int_{H(\mathbf{x})} L(\mathrm{r}(\mathbf{x}, \omega'), -\omega') \cdot f_r(\mathbf{x}, \omega' \rightarrow \omega_o) \cdot \cos\theta' \, \mathrm{d}\omega'$$

- Pro výpočet $L(\mathbf{x}, \omega_o)$ potřebuji spočítat $L(\mathrm{r}(\mathbf{x}, \omega'), -\omega')$ pro všechny směry ω' okolo bodu $\mathbf{x}$.
- Pro výpočet každého $L(\mathrm{r}(\mathbf{x}, \omega'), -\omega')$ potřebuji spočítat $L(\mathrm{r}(\mathrm{r}(\mathbf{x}, \omega'), -\omega''), -\omega')$ pro všechny směry ω'' okolo bodu $\mathrm{r}(\mathbf{x}, \omega')$
- Atd… => rekurze



Sledování cest (Path tracing, Kajiya86)

- Pouze jeden sekundární paprsek
  1. Náhodný výběr interakce (ideální lom, difúzní odraz, …)
  2. Importance sampling podle vybrané interakce
  - Výhoda: žádná exploze počtu paprsků kvůli rekurzi

- Přímé osvětlení
  - Doufej, že náhodně vygenerovaný paprsek trefí zdroj, anebo
  - Vyber náhodně jeden vzorek na jednom zdroji světla
  - Nejlépe: kombinuj předchozí přístupy pomocí MIS

- Trasuj stovky cest přes každý pixel a zprůměruj výsledek

Path tracing, rekurzivní formulace

getLi (x, ω):

y = traceRay(**x**, ω)

return

Le(**y**, −ω) + // emitted radiance

Lr (**y**, −ω) // reflected radiance

Lr(x, ω):

ω′ = genUniformHemisphereRandomDir(**n**(**x**))

return 2π * brdf(x, ω, ω′) * dot(**n**(**x**), ω′) * rayRadianceEst(x, ω′)

Sledování cest od kamery

```
renderImage()
{
  for all pixels
  {
    Color pixelCol = (0,0,0);
    for k = 1 to N
    {
      wk := náhodný směr skrz k-tý pixel
      pixelCol += getLi(camPos,wk)
    }
    return Lo / N
  }
}
```

Path Tracing – Implicitní osvětlení

```
getLi(x, w)
{
    Color thrput = (1,1,1)
    Color accum  = (0,0,0)
    while(1)
    {
        hit = NearestIntersect(x, w)
        if no intersection
                return accum + thrput * bgRadiance(x, w)
        if isOnLightSource(hit)
                accum += thrput * Le(hit.pos, -w)
        ρ = reflectance(hit.pos, -w)
        if rand() < ρ // russian roulette – survive (reflect)
                wi := SampleDir(hit)
                thrput *= fr(hit.pos, wi, -w) * dot(hit.n, wi) / (ρ*pdf(wi))
                x  := hit.pos
                w  := wi
        else // absorb
            break;
    }
    return accum;
}
```

Ukončení rekurze – Ruská ruleta

- Pokračuj v rekurzi s pravděpodobností q
- Uprav váhu faktorem 1 / q

$$Z = \begin{cases} Y / q & \text{pokud } \xi < q \\ 0 & \text{jinak} \end{cases}$$

$$E[Z] = \frac{E[Y]}{q} \cdot q + 0 \cdot \frac{1}{q-1} = E[Y]$$

Výběr náhodného směru – Importance Sampling

```
getLi(x, w)
{
    Color thrput = (1,1,1)
    Color accum  = (0,0,0)
    while(1)
    {
        hit = NearestIntersect(x, w)
        if no intersection
            return accum + thrput * bgRadiance(x, w)
        if isOnLightSource(hit)
            accum += thrput * Le(hit.pos, -w)
        ρ = reflectance(hit.pos, -w)
        if rand() < ρ // russian roulette - survive (reflect)
            wi := SampleDir(hit)
            thrput *= fr(hit.pos, wi, -w) * dot(hit.n, wi) / (ρ * pdf(wi))
            x  := hit.pos
            w  := wi
        else // absorb
            break;
    }
    return accum;
}
```

Výběr náhodného směru – Importance Sampling

- Obyčejně vzorkujeme s hustotou „co nejpodobnější“ součinu

$$f_r(\omega_i, \omega_o) \cos \theta_i$$

  - Ideálně bychom chtěli vzorkovat podle

$$L_i(\omega_i) f_r(\omega_i, \omega_o) \cos \theta_i,$$

    ale to neumíme, protože neznáme L_i

- Co když bude hustota přesně úměrná $f_r(\omega_i, \omega_o) \cos \theta_i$?

„Ideální“ BRDF Importance Sampling

$$p(\omega_\mathrm{i}) \propto f_r(\omega_\mathrm{i} \rightarrow \omega_\mathrm{o}) \cdot \cos\theta_\mathrm{i}$$

- Normalizace (integrál pdf musí být = 1)

$$p(\omega_\mathrm{i}) = \frac{f_r(\omega_\mathrm{i} \rightarrow \omega_\mathrm{o}) \cdot \cos\theta_\mathrm{i}}{\int_{H(\mathbf{x})} f_r(\omega_\mathrm{i} \rightarrow \omega_\mathrm{o}) \cdot \cos\theta_\mathrm{i} \, \mathrm{d}\omega_\mathrm{i}}$$

odrazivost ρ

„Ideální“ BRDF IS v Path Traceru

- Obecná hustota (pdf)

```
...
thrput *= fr(.) * dot(.) / ( ρ * p(wi) )
```

- „Ideální“ BRDF importance sampling

$$p(\omega_i) = f_r(\omega_i \rightarrow \omega_o) \cdot \cos\theta_i \;/\; \rho$$

```
...
thrput *= 1
```

Pravděpodobnost přežití cesty

```
getLi(x, w)
{
    Color thrput = (1,1,1)
    Color accum  = (0,0,0)
    while(1)
    {
        hit = NearestIntersect(x, w)
        if no intersection
                return accum + thrput * bgRadiance(x, w)
        if isOnLightSource(hit)
                accum += thrput * Le(hit.pos, -w)
        ρ = reflectance(hit.pos, -w)
        if rand() < ρ // russian roulette - survive (reflect)
                wi := SampleDir(hit)
                thrput *= fr(hit.pos, wi, -w) * dot(hit.n, wi) / (ρ * p(wi))
                x  := hit.pos
                w  := wi
        else // absorb
            break;
    }
    return accum;
}
```

Pravděpodobnost přežití cesty

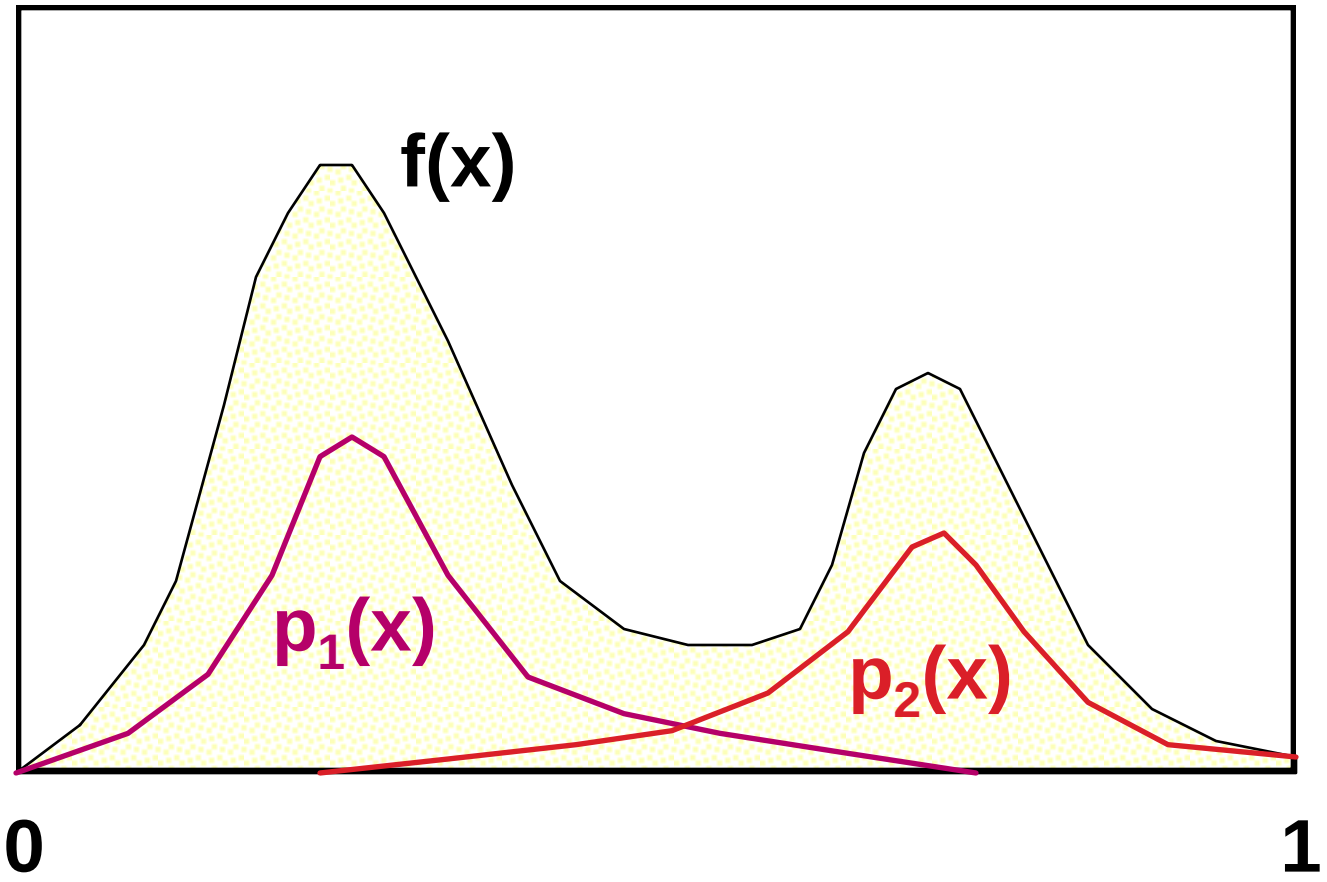
- Použití odrazivosti ρ jako p-nosti přežití dává smysl
  - Pokud plocha odráží jen 30% energie, pokračujeme pouze s 30% pravděpodobností.

- Co když neumím spočítat ρ ?
- Alternativa
  1. Nejdříve vygeneruj náhodný směr podle $p(\omega_i)$
  2. $$q_{\text{survival}} = \min\left\{1,\ \frac{f_r(\omega_i \rightarrow \omega_o)\cos\theta_i}{p(\omega_i)}\right\}$$
  - Pro „ideální“ BRDF IS stejné jako původní metoda

Výpočet přímého osvětlení pomocí MIS v path traceru

Multiple Importance Sampling

(Veach & Guibas, 95)



Vyrovnaná heuristika (Balance heurist.)

- Výsledný estimátor (po dosazení vah)

$$F = \sum_{i=1}^{n} \sum_{j=1}^{n_i} \frac{f(X_{i,j})}{\sum_k n_k p_k(X_{i,j})}$$

  - příspěvek vzorku nezávisí na tom, ze které byl pořízen techniky (tj. pdf)

Použití MIS v path traceru

- Pro každý vrchol cesty generované z kamery:
  - Generování explicitního stínového paprsku pro techniku p_2 (vzorkování plochy zdroje)
  - Sekundární paprsek pro techniku p_1 (vzorkování zdroje)
    - Sdílený pro výpočet **přímého** i **nepřímého** osvětlení
    - Pouze na přímé osvětlení se aplikuje MIS váha (nepřímé osvětlení se připočte celé)
  - Při výpočtu MIS vah je potřeba vzít v úvahu pravděpodobnost ukončení cesty (ruská ruleta)

Více zdrojů světla

- Možnost 1:
  - Stínový paprsek pro náhodný bod na každém zdroji světla
- Možnost 2 (často lepší):
  - Náhodný výběr zdroje (s p-ností podle výkonu)
  - Stínový paprsek k náhodně vybranému bodu na vybraném zdroji
- Pozor: Pravděpodobnost výběru zdroje ovlivňuje hustoty (a tedy i váhy) v MIS