# Raytracing
## a další globální zobrazovací metody

Pavel Strachota

FJFI ČVUT v Praze

11. května 2015

# Obsah

# Obsah

# Motivace

obsah přednášky až dosud: kroky vedoucí k přímému zobrazení jednotlivých objektů na průmětnu $\Longrightarrow$ vhodné pro zobrazování v reálném čase

---

- definice scény, triangulace
- **promítání** - maticová transformace
- **řešení viditelnosti** - obrazový a objektový přístup, více či méně univerzální algoritmy, *z*-buffer, přímý výpočet **stínů**
- **osvětlování** a metody **stínování** - aplikace osvětlovacího modelu na polygonální síť
- mapování textur (bude potřeba i pro globální metody), perspektivně korektní mapování **přímo v průmětně**

---

**nyní**: globální zobrazovací metody, snaha o fotorealistické zobrazování $\Longrightarrow$ barvu každého pixelu ovlivňují obecně všechny objekty ve scéně

# Raytracing - úvod

- **raytracing (sledování paprsku)** - klasická globální vykreslovací metoda
- vychází z geometrické optiky, jednoduchý algoritmus, jednoduchá implementace
- **ne**ní založena na stochastickém fyzikálním osvětlovacím modelu jako další metody
- ve vhodné modifikaci je základní součástí komplexních algoritmů pro fotorealistické zobrazování, které kombinují více metod pro zachycení co nejširší škály efektů.

Software:
POV-Ray, Mental Ray, V-Ray, **YafaRay** (dříve YafRay)

---

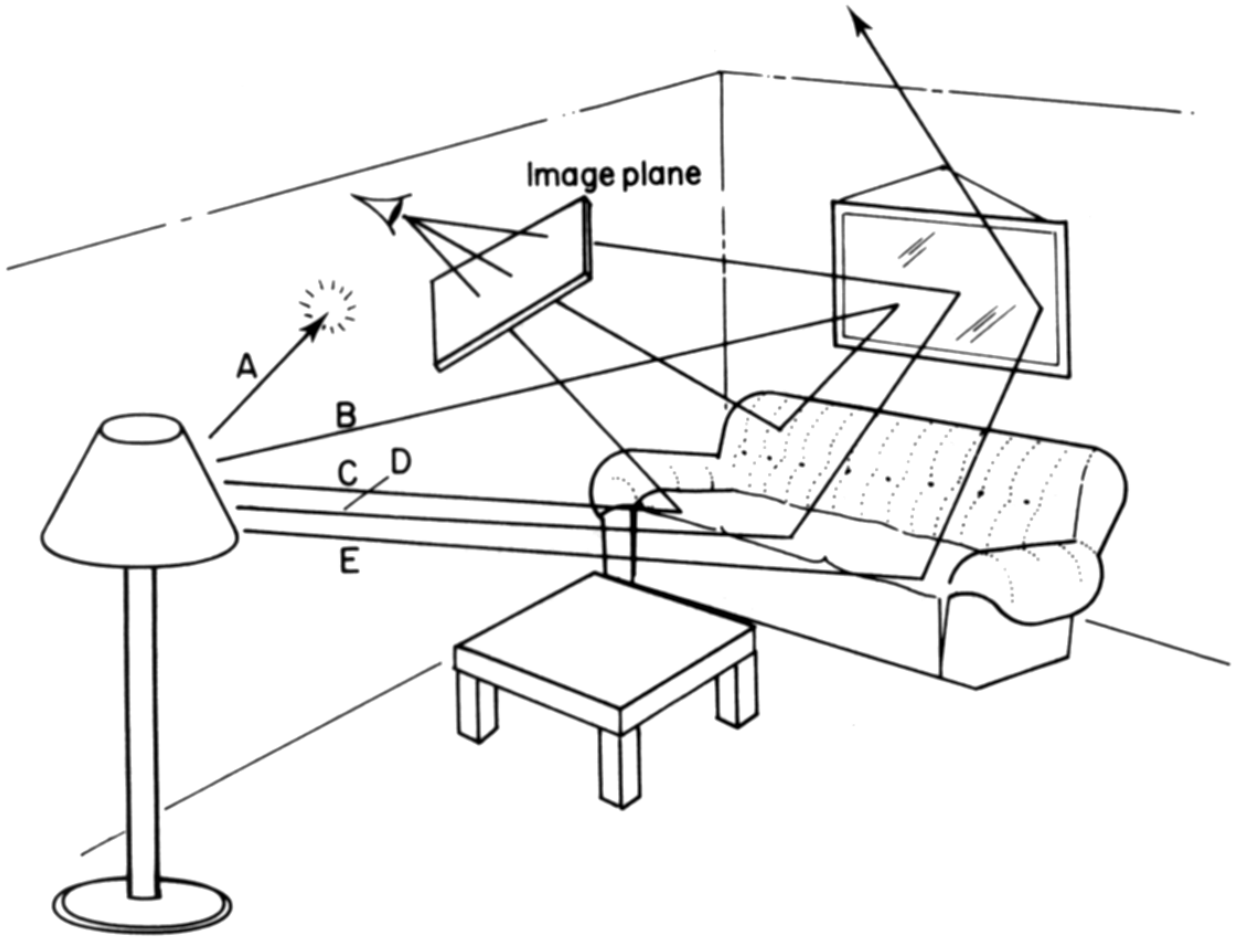
- výchozí situace pro všechny metody: máme definovanou scénu a pohled kamery - tj. umístění průmětny a definice středu, resp. směru promítání

# Princip raytracingu

dva přístupy k výpočtu šíření světla pomocí geometrické optiky

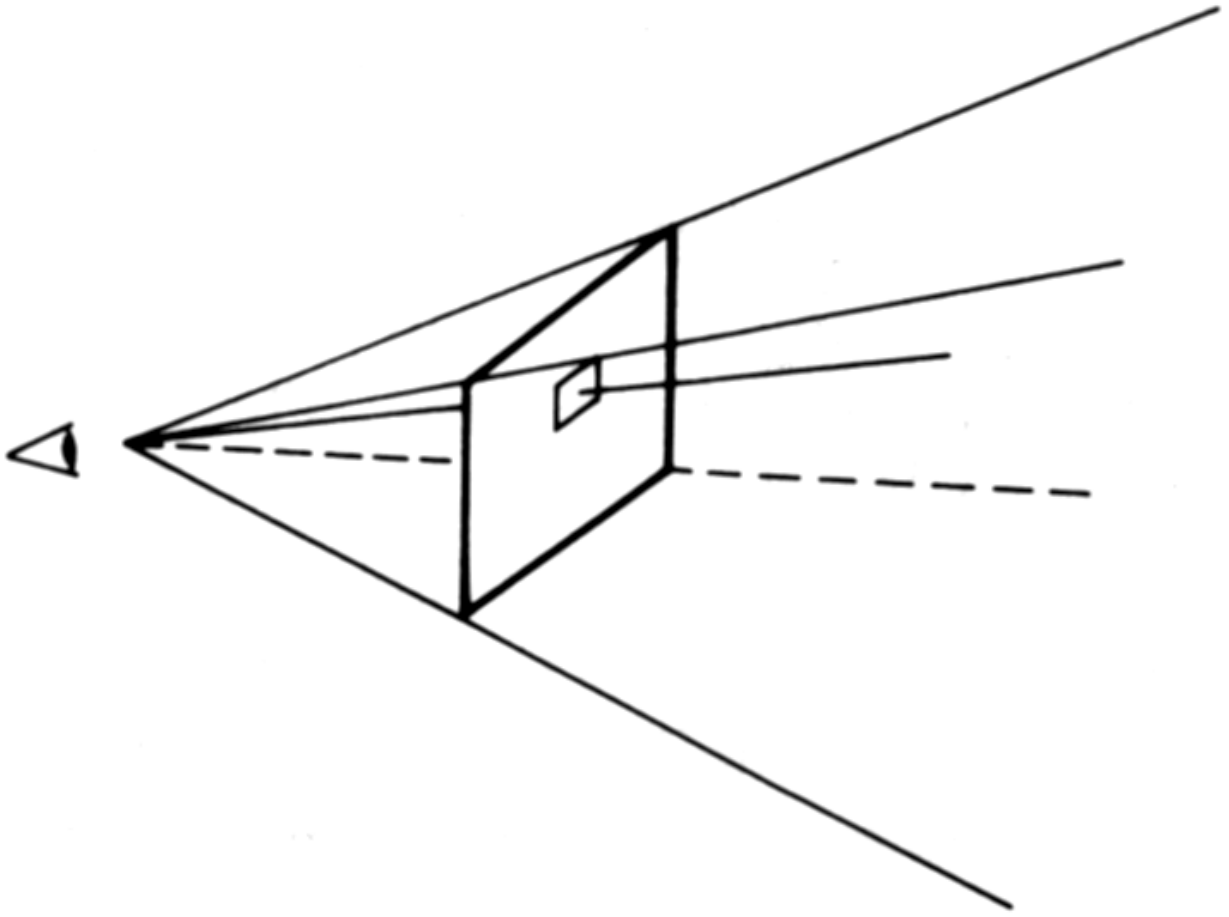
1. ***sledování fotonů*** - **ze světelného zdroje** vyšleme foton (směr je závislý na typu zdroje) a sledujeme, na jaké těleso dopadne, kam se odrazí, lomí, absorbuje.
   - jestliže se nakonec odrazí do průmětny, poznamenáme si jeho barvu v příslušném pixelu
   - problém - většina fotonů letí mimo průmětnu (přesto lze využít - viz *fotonové mapy*)
2. ***sledování paprsku*** **směrem od pozorovatele** - každým pixelem průmětny vedeme paprsek směrem od pozorovatele (středu promítání)
   - ve skutečnosti se světlo šíří obráceně, ale problém lze přeformulovat:
     místo „který foton se trefí do průmětny?“ se ptáme „**jakou barvu bude mít paprsek, který přilétá k pozorovateli z daného směru (daným pixelem)**?“
   - barva paprsku vznikne složením barvy mnoha „fotonů“ (viz dále)

# Znázornění šíření světla pomocí paprsků




Paprsky ve scéně

# Znázornění šíření světla pomocí paprsků



**Raytracing** - vyslání paprsku pixelem v průmětně

# Šíření světla v základní verzi algoritmu

- sledujeme, jak vznikne paprsek, který k pozorovateli přichází daným pixelem průmětny
- najdeme nejbližší povrch, který paprsek protíná, a od nějž tedy přichází
- v závislosti na vlastnostech povrchu světlo paprsku tvoří:

1. ambientní osvětlení objektu
2. světlo vzniklé **přímým difuzním odrazem (bodových) světelných zdrojů od povrchu**
3. světlo vzniklé **přímým zrcadlovým odrazem světelných zdrojů**
4. **zrcadlový odraz** paprsku přicházejícího od **jiného objektu** v odpovídajícím směru
5. světlo lomené na povrchu (průhledného) objektu

- Pro (1), (2), (3) máme Phongův osvětlovací model, vznik paprsků (4) a (5) dále **rekurzivně** sledujeme

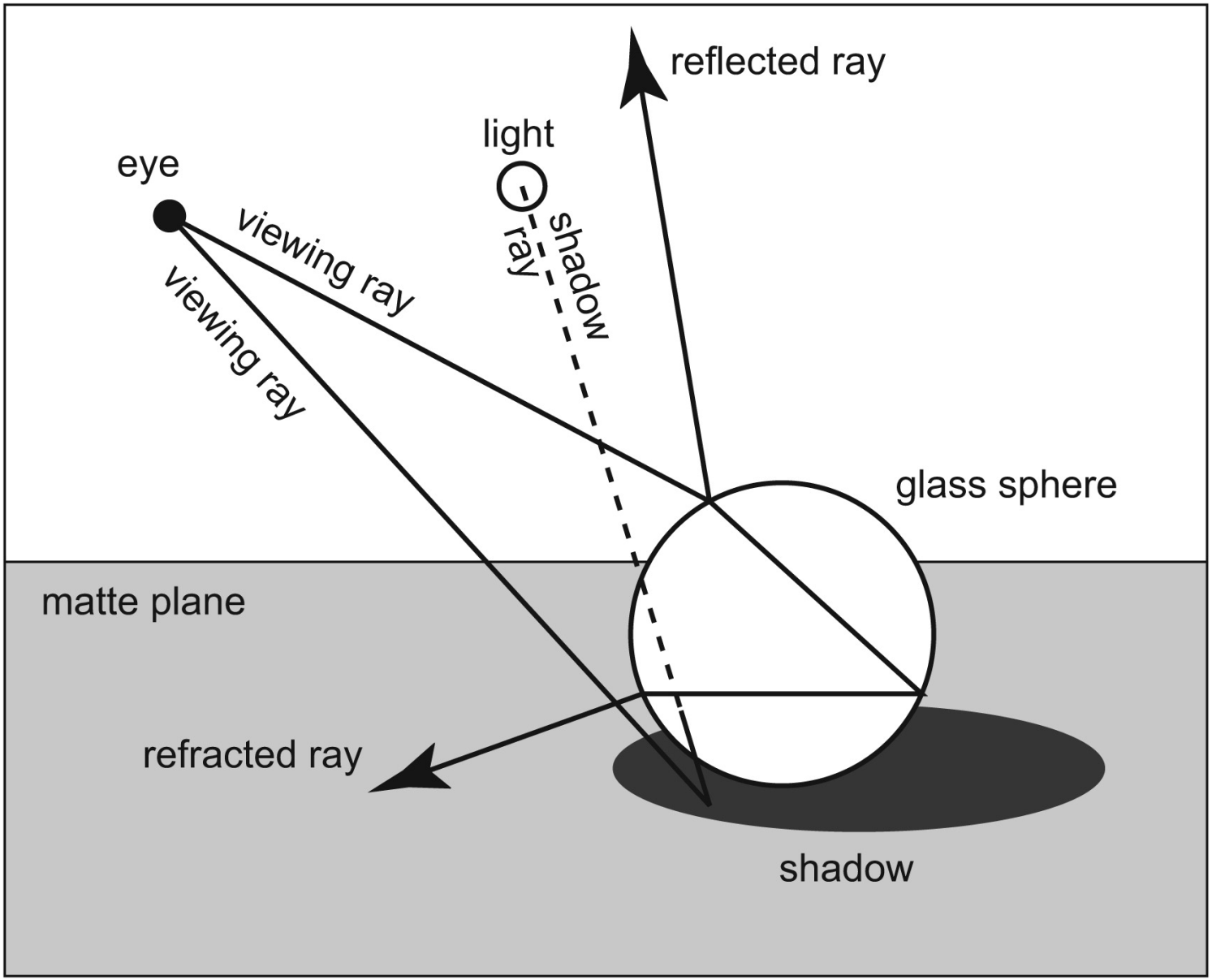
# Terminologie pro typy paprsků

- **obrazový (primární) paprsek** (*viewing ray*) - paprsek, o jehož barvu se zajímáme (který zpočátku vyšleme z místa pozorovatele)

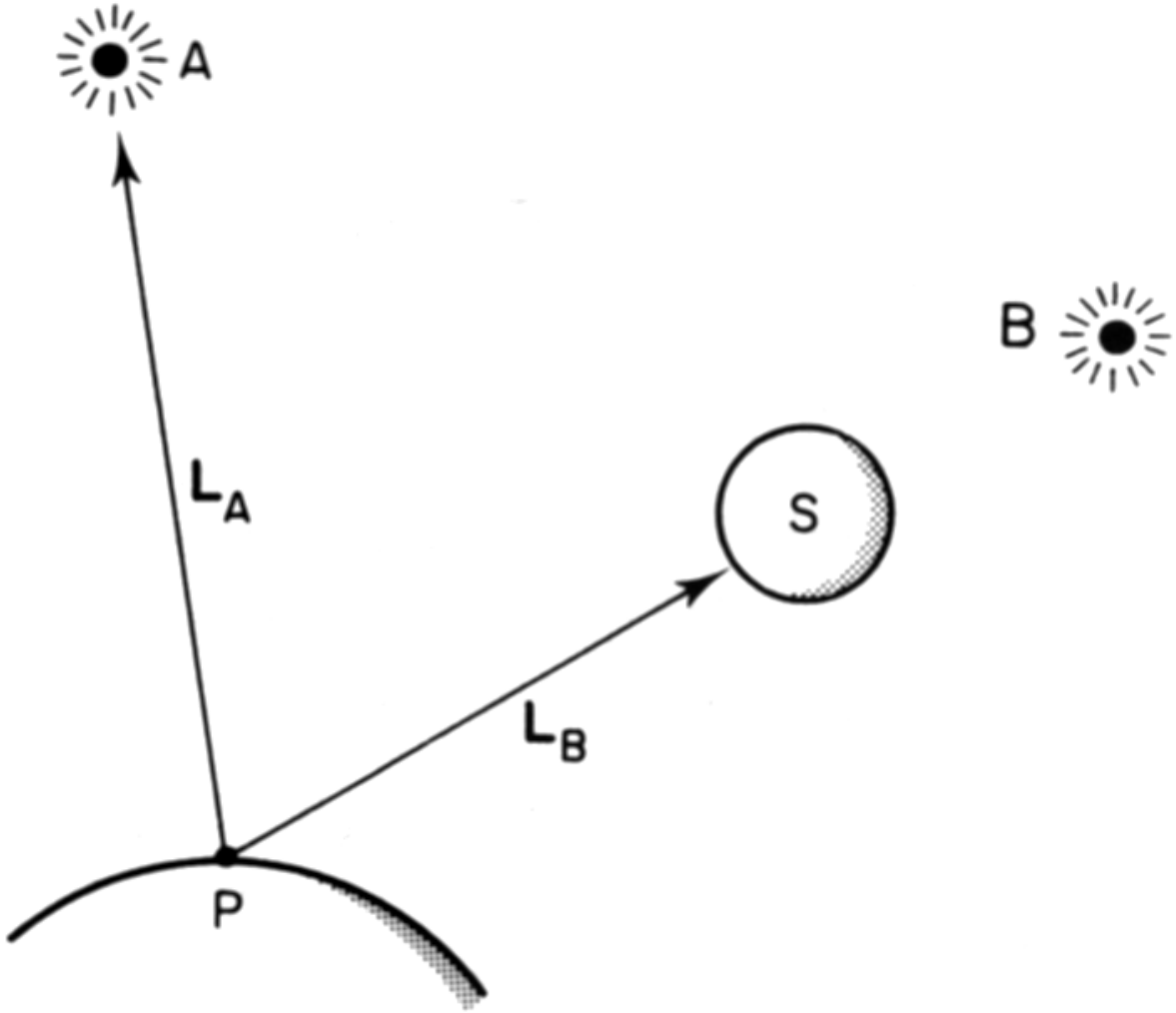
Sekundární paprsky:

- **stínový paprsek** (*shadow ray*) - vyslaný z místa dopadu ke každému z bodových světelných zdrojů - jestliže v cestě nestojí jiný objekt, resp. povrch, daný bod je světelným zdrojem osvětlen $\Longrightarrow$ stínové paprsky jsou zodpovědné za zobrazení stínu
- **odražený paprsek** (*reflected ray*) - paprsek ve směru, odkud musí přicházet světlo, aby se odrazilo k pozrovateli (tj. do směru obrazového paprsku)
- **lomený paprsek** (*refracted ray)* - paprsek ve směru, odkud musí světlo procházet povrchem objektu (rozhraním opticky aktivních prostředí), aby došlo k lomu světla k pozorovateli

# Typy paprsků - schéma



# Stínové paprsky

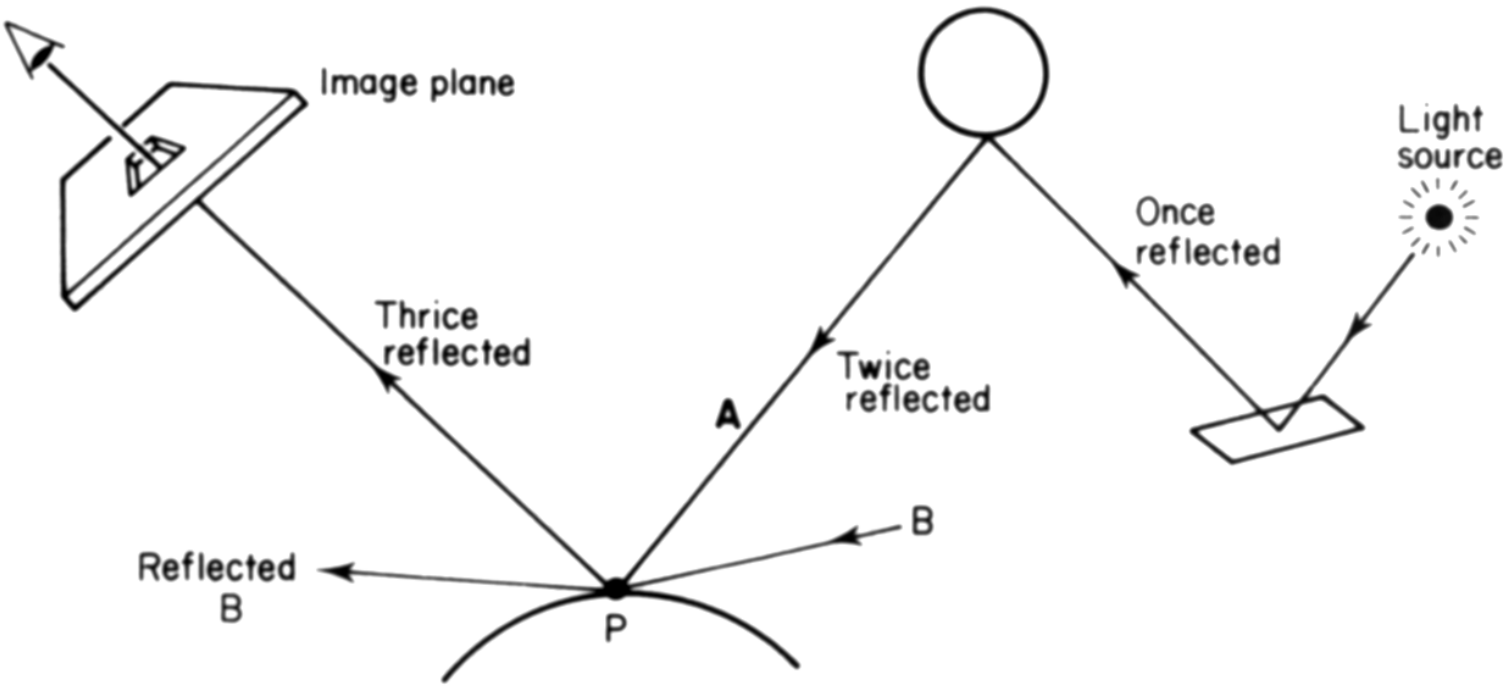



- nezajímáme se o to, který průsečík je nejbližší, ale pouze o to, zda průsečík existuje

# Odražené paprsky

- viz *Osvětlování a stínování*: odraz paprsku $\boldsymbol{v}$ (směrem od pozorovatele) od povrchu s normálou $\boldsymbol{n}$ vypočítáme jako

$$\boldsymbol{r} = 2(\boldsymbol{n} \cdot \boldsymbol{v})\,\boldsymbol{n} - \boldsymbol{v}$$

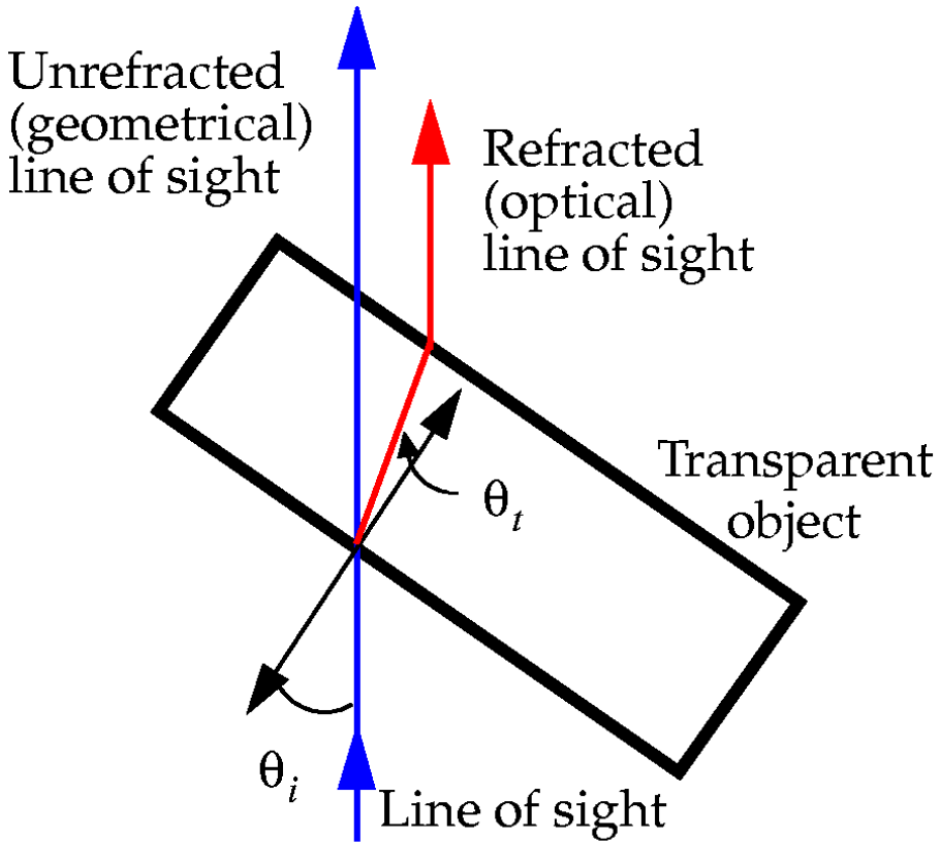


# Lomené paprsky 1/2

- Snellův* zákon lomu
- založen na principu nejkratšího času průchodu paprsku prostředím
- platí

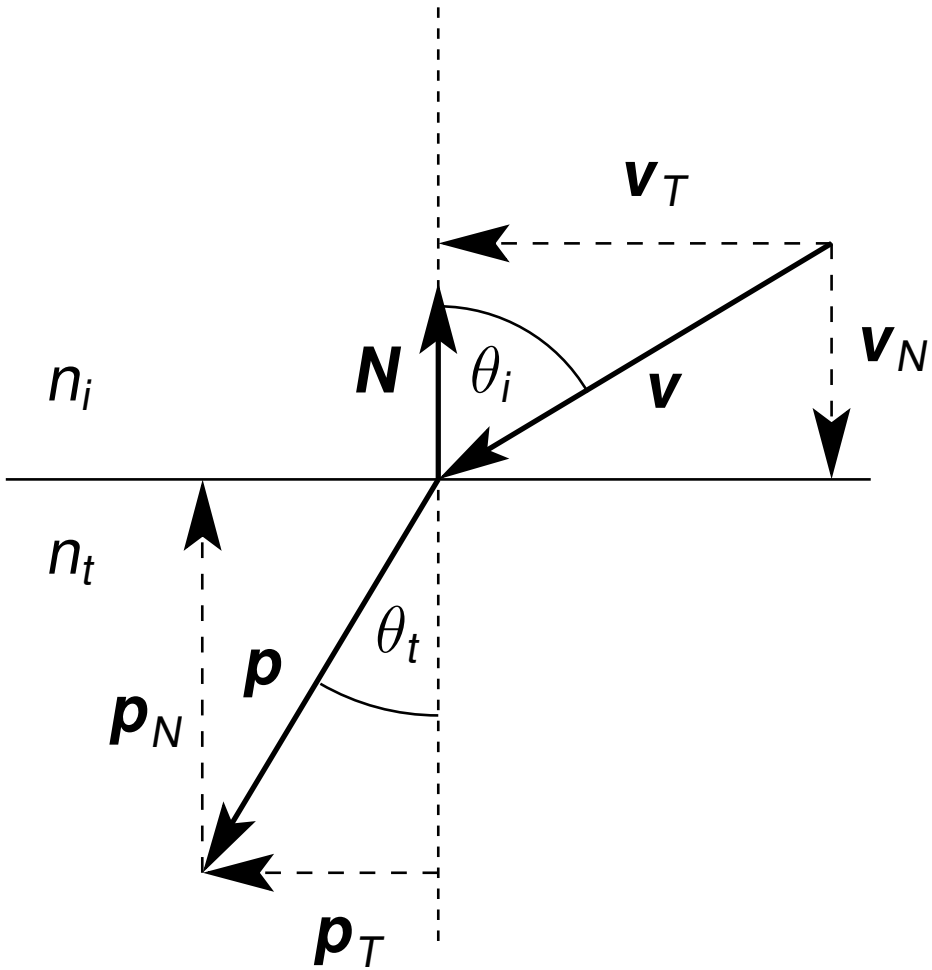
$$\frac{\sin\theta_i}{\sin\theta_t} = \frac{v_i}{v_t} = \frac{n_t}{n_i},$$

kde $n_i, n_t$ jsou indexy lomu dvou prostředí na rozhraní a $v_i, v_t$ jsou rychlosti světla v daných prostředích




*) *Snell - holandský matematik Willebrord Snellius*

# Lomené paprsky 2/2




předpokládáme $|\boldsymbol{N}| = 1$:

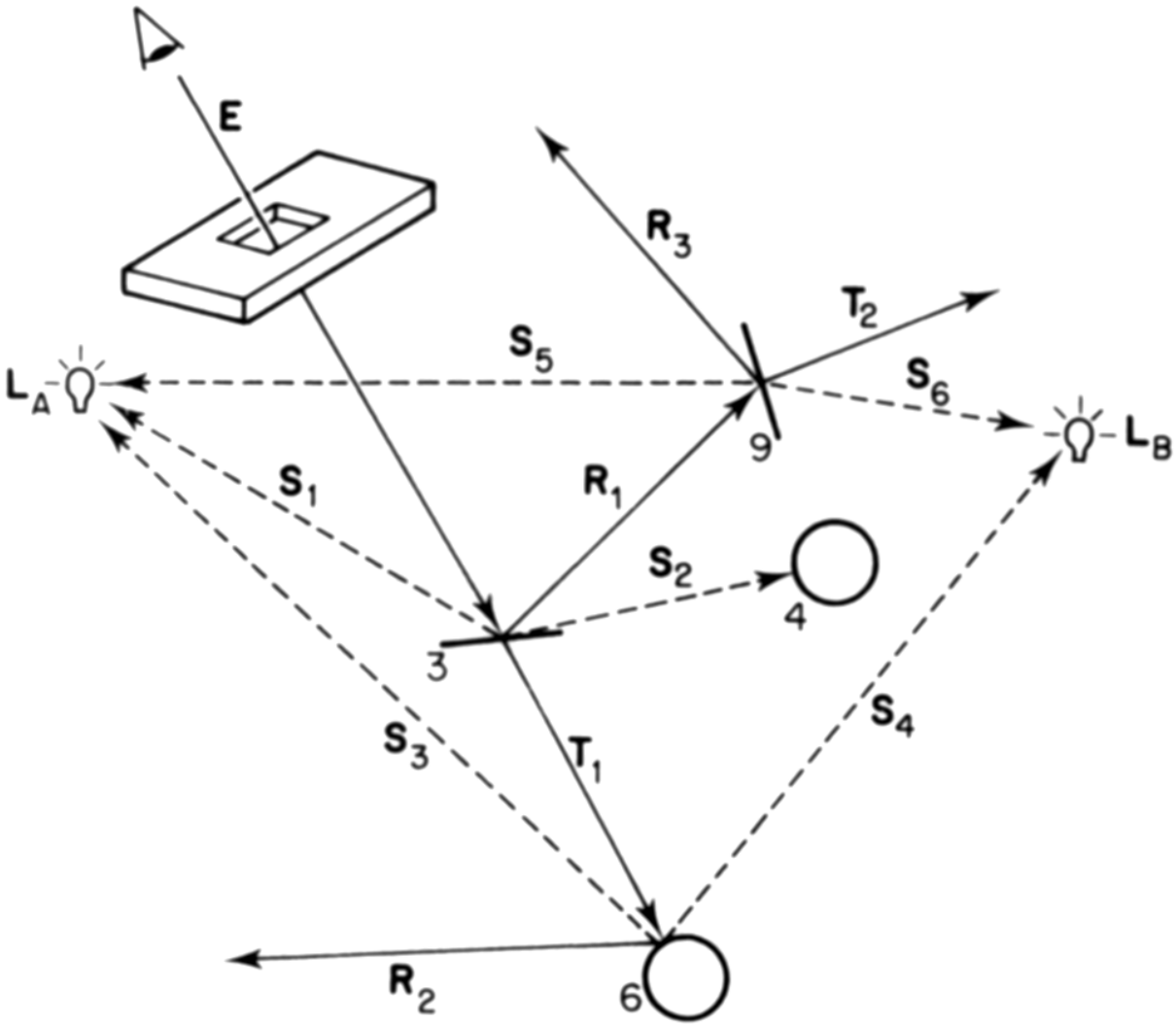
- $\boldsymbol{v}_N = (\boldsymbol{v} \cdot \boldsymbol{N})\boldsymbol{N}$
- $\boldsymbol{v}_T = \boldsymbol{v} - \boldsymbol{v}_N$
  - a platí $|\boldsymbol{v}_T| = |\boldsymbol{v}| \sin\theta_i$
- $\boldsymbol{p}_T = k\boldsymbol{v}_T$ kde $k = \frac{\sin\theta_t}{\sin\theta_i} = \frac{n_i}{n_t}$
  - takže pak $|\boldsymbol{p}_T| = |\boldsymbol{v}| \sin\theta_t$
- $\boldsymbol{p}_N = \sqrt{|\boldsymbol{v}|^2 - |\boldsymbol{p}_T|^2}\,\boldsymbol{N}$
- $\boldsymbol{p} = \boldsymbol{p}_T - \boldsymbol{p}_N$
  - takže $|\boldsymbol{p}| = |\boldsymbol{v}|$

# Rekurzivní sledování paprsku

- v bodě průsečíku obrazového paprsku s povrchem vypočítáme stínové praprsky, odražený a lomený paprsek
- zjistíme, které stínové paprsky (směrem ke zdrojům světla) neprochází dalšími objekty („volné“ paprsky)
- pro světelné zdroje příslušné „volným“ stínovým paprskům vyhodnotíme Phongův osvětlovací model
- k výsledné barvě přičteme barvu odražených a lomených paprsků, násobenou koeficientem $< 1$
- tyto barvy zjistíme **rekurzivním sledováním** obou paprsků - stávají se **obrazovými paprsky** a aplikujeme na ně stejnou proceduru jako na původní paprsek od pozorovatele
- **zastavení rekurze**:
  - paprsek opustí scénu
  - příspěvek barvy je již příliš malý
  - pevná hloubka rekurze

# Rekurzivní sledování paprsku

## Příklad



# Nejjednodušší rekurzivní raytracing - pseudokód

```
RGB TraceRay(Ray v, int r_depth) {
 FinalColor={0,0,0};
 Isect=FindNearestIntersection(v);
 if(Isect==INFINITY) return(BackgroundColor);
 foreach L in LightSources {
  if(ShadowRay(Isect,L)==FREE)
   FinalColor += PhongIllumination(Observer,Isect,L);
 }
 if(r_depth<MAX_RECURSION_DEPTH) FinalColor +=
   M_S(Isect)*TraceRay(ReflectedRay(v,Isect),r_depth+1)
  +M_T(Isect)*TraceRay(RefractedRay(v,Isect),r_depth+1);
 return(FinalColor);
}
```

- `M_S` ... zrcadlová barva materiálu $m_S$ (viz *osvětlování*)
- `M_T` ... barva průhledného materiálu (*transparency*)

# Obsah

# Nalezení průsečíku paprsku s objektem

- paprsek: přímka parametrizována jako $\boldsymbol{S} + t\boldsymbol{v}$
- algoritmy pro nalezení průniku přímky a jednoduchých povrchů:
  - koule
  - rovina, trojúhelník, polygon
  - bikubické povrchy, NURBS, ...
- např. objekty reprezentované pomocí **CSG** (viz *modelování pevných těles*)- snadné vykreslení pomocí raytracingu:
  - nelze nalézt jen průsečík s nejbližším primitivním tělesem, ale:

1. vypočítáme všechny průsečíky paprsku s primitivními tělesy v reprezentaci objektu
   $\Longrightarrow$ vzniknou intervaly parametru $t$ (úsečky na paprsku), kde je paprsek uvnitř jednotlivých primititv
2. s intervaly provedeme stejné operace, jaké jsou uloženy ve vrcholech CSG stromu $(\cup, \cap, -)$
3. začátek nejbližšího intervalu udává průsečík se skutečným tělesem

# Vliv konečné přesnosti 1/2

- průsečík $\boldsymbol{P}$ nalezen pomocí hodnoty parametru $t$ v parametrickém zápisu přímky: $\boldsymbol{P} = \boldsymbol{S} + t_{\boldsymbol{P}}\boldsymbol{v}$
- vlivem konečné přesnosti počítače neleží průsečík přesně na povrchu
  $\Longrightarrow$ při sledování stínového/odraženého/lomeného paprsku můžeme detekovat falešný průsečík se stejným povrchem v jeho bezprostřední blízkosti.
  Např. po parametrickém vyjádření odraženého paprsku ve směru $\boldsymbol{r}$ najdeme průsečík

$$\boldsymbol{P'} = \boldsymbol{P} + t_{\boldsymbol{P'}}\boldsymbol{r},\ t \neq 0,\ \text{ale}\ t \approx 0.$$

# Vliv konečné přesnosti 2/2

## Řešení

1. vyloučit stejné těleso ze zpracování sekundárních paprsků - jen pro stínové a odražené paprsky, navíc nutný předpoklad konvexnosti tělesa
2. posunout nový výchozí bod $\boldsymbol{P}$ (původní průsečík) ve směru normály ven z objektu (pro stínové a odražené paprsky), resp. dovnitř (pro lomené)

# Metody urychlení algoritmu

- rychlejší výpočet průsečíků
  - rychlejší výpočet jednotlivého průsečíku - obaly objektů, efektivnější rutiny pro výpočet průsečíku
  - méně výpočtů průsečíků - hierarchie ohraničujících objemů (obalů), rozdělení prostoru (uniformní či neuniformní), směrové techniky
- méně paprsků
  - adaptivní vzorkování podle složitosti scény
  - zobecněné paprsky - válec, kužel
- paralelizace (snadná), vektorizace

# Obaly objektů a jejich hierarchie

- **jednoduché obaly** objektů - například koule obsahující jeden celý objekt (Whitted)
  - pokud paprsek neprotne ani kouli, není nutné testovat průsečík s objektem
  - pro paprsek blízko objektu (protne kouli) sice provádíme 2 testy, ale takových dvojic paprsek-objekt je obvykle málo $\Longrightarrow$ v průměru urychlení o konstantní faktor
  - nevýhoda: stále zůstává test všech $n$ objektů (složitost $O(n)$)
- **hierarchické** obaly - strom obalů ve formě rovnoběžnostěnů (Rubin, Whitted)
  - uzel stromu představuje obal, který v sobě obsahuje všechny obaly v potomcích
  - pokud paprsek neprotne kořen, není třeba testovat jeho potomky atd.
  - problém: sestavit hierarchii efektivně $\Longrightarrow$ snaha o složitost $O(\log n)$

# Paměť překážek (*light buffer*)

- jedna z metod založených na tzv. *směrové krychli*
- bodový světelný zdroj obklopen krychlí, jejíž povrch je případně rozdělen uniformně nebo adaptivně (např. *quadtree*) na menší čtverce
- ke každému čtverci přiřazen seznam objektů, pro které existuje paprsek od zdroje skrz daný čtverec, který je protíná (preprocessing)

---

- v průběhu raytracingu snadno určíme, kterým čtvercem prochází stínový paprsek
- získáme příslušný seznam objektů.
- testujeme průsečík stínového paprsku jen s objekty v seznamu. Ostatní objekty nemůže stínový paprsek protnout, protože je neprotne **žádný** paprsek od zdroje skrz daný čtverec.

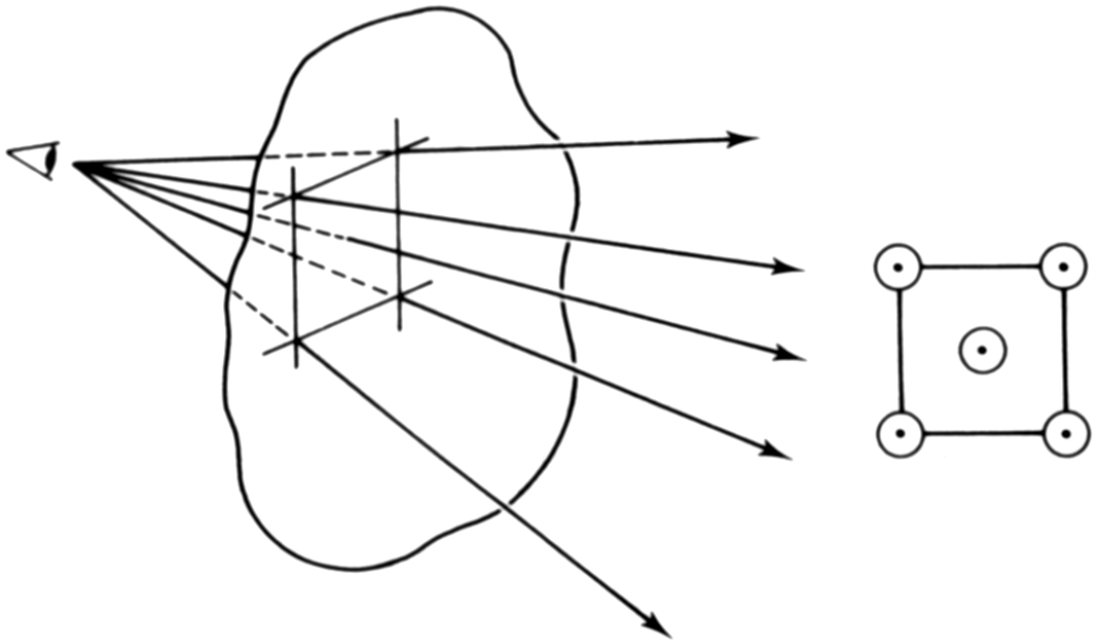
# Obsah

# Nedostatky základní verze algoritmu

- **aliasing**
- umožňuje pouze **bodové zdroje** světla
  - generuje pouze **ostré stíny**
  - neumí **kaustiky** (*caustics*) - obrazce vzniklé promítáním světla, prostupujícího od světelného zdroje průhledným objektem, na povrch jiného objektu
- rekurze nezahrnuje **difuzní odraz** světla od objektů do scény - stále máme ambientní složku osvětlení
- neumí zpracovat **průsvitnost** (*translucency*), pouze čirou průhlednost
- chybí iluze **hloubky ostrosti** (*depth of field*)

# Antialiasing - adaptivní nadvzorkování

- vyšleme každým pixelem 5 paprsků, jeden středem a 4 jeho rohy
- jestliže se výsledné barvy paprsků příliš neliší, pixel obarvíme jejich průměrem
- v opačném případě rozdělíme pixel do 4 subpixelů a na každý opakujeme stejný postup



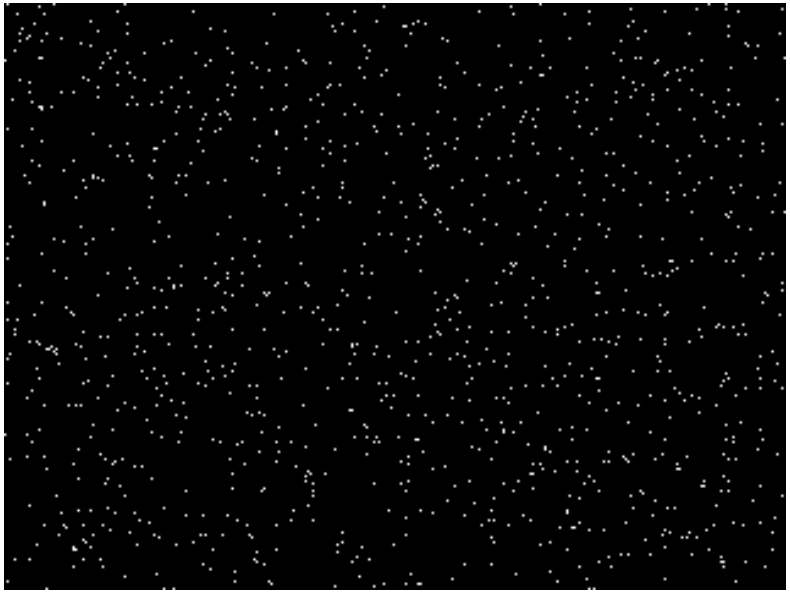
# Adaptivní nadvzorkování

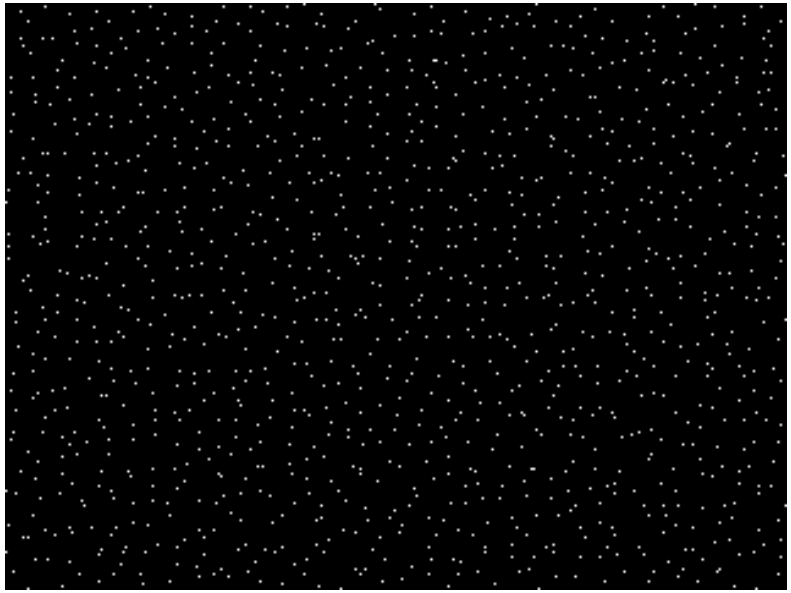
## Vlastnosti

- snadný postup, dobře odstraní aliasing
- zásah tělesa s malým průmětem však není jistý ani při 5 vyslaných paprscích (těleso mineme a stále všechny paprsky budou mít podobnou barvu)
- nadvzorkování je lepší udělat **nepravidelně**

# Stochastické vzorkování

- nadvzorkování
- výběr vzorků
  - zcela náhodně
  - v každém pixelu stejný počet vzorků, náhodně posunutých v rámci pixelu vůči pravidelné přížce - tzv. **roztřesení** (*jittering*)
- efektivně odstraní aliasing - přesněji: aliasing je ve formě šumu, k němuž je lidské vnímání lhostejné





# Stochastické vzorkování

## Roztřesení - náhodné rozdělení

- aproximujeme zákon řídkých jevů (Poissonovo rozdělení)

Zákon řídkých jevů:
právě jeden vzorek v intervalu $(t, t+h)$ s pravděpodobností $\lambda h + o(h)$, více než jeden s pravděpodobností $o(h)$ nezávisle na $t$ ($\lim_{h\to 0} o(h)/h = 0$). Potom pravděpodobnost, že v intervalu $(0, t)$ máme $k$ vzorků, je

$$P(X_t = k) = \frac{(\lambda t)^k}{k!} e^{-\lambda t}$$

- vzorky jsou dostatečně daleko od sebe a zároveň ne příliš daleko

# Distribuovaný raytracing 1/8

- nadvzorkování lze využít i v průběhu rekurzivního sledování
- generujeme svazek primárních, odražených, stínových a lomených paprsků kolem původního přesného paprsku
- jejich směry mají náhodné rozdělení dané **distribuční funkcí**, která je specifická pro daný povrch
- tím stochasticky (metodou Monte Carlo) aproximujeme integrál celkové intenzity světla $I$ odrážející se od povrchu pod prostorovým úhlem $(\phi_r, \theta_r)$

$$I(\phi_r, \theta_r) = \int_{\phi_i} \int_{\theta_i} L(\phi_i, \theta_i)\, R(\phi_i, \theta_i, \phi_r, \theta_r)\, \mathrm{d}\phi_i \mathrm{d}\theta_i,$$

kde $R$ je funkce odrazivosti a $L$ je funkce osvětlení (dopadající intenzity světla v závislosti na úhlu)

# Distribuovaný raytracing 2/8

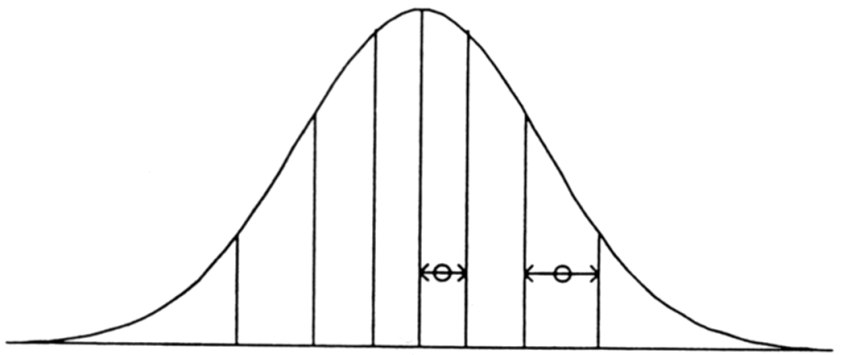
## Implementace vzorkování - vážení vzorků

- vzorky chceme vážit tak, aby např. příspěvek odražených paprsků odpovídal funkci odrazivosti $R(\phi_i, \theta_i, \phi_r, \theta_r)$

2 možnosti

1. vzorkovat strejnoměrně, poté vážit hodnotou $R$ (nebo obecně jiného rozdělení v závislosti na případu)
2. generovat směry paprsků s pravděpodobností odpovídající „důležitosti“ - např. hodnotě $R$ (*importance sampling*)

- výsledné barvy už jen zprůměrovat
- rozdělení def. oboru „filtru“ (např. funkce $R$) na oblasti $\Omega_i$ o stejné ploše (integrálu $\int_{\Omega_i} R$), 1 vzorek na každou oblast



# Distribuovaný raytracing 3/8

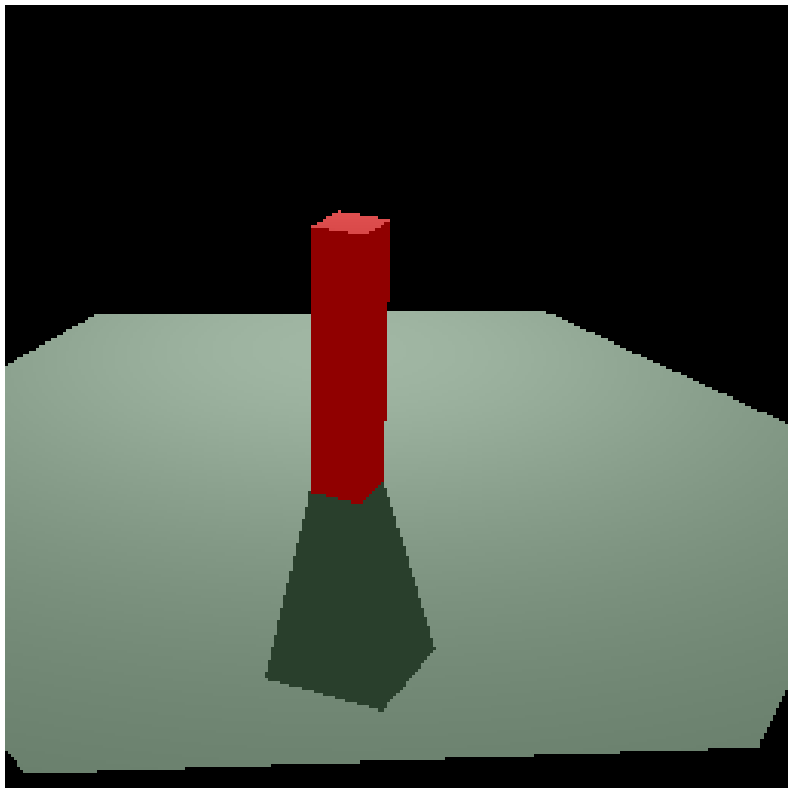
## Efekty dosažitelné distribuovaným nadvzorkováním

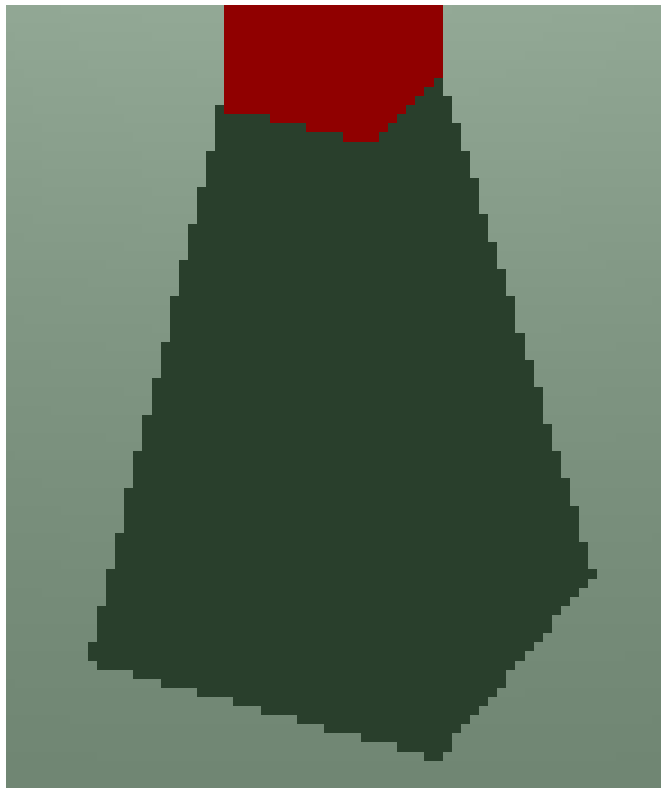
- odpadají zjednodušující předpoklady, vedoucí k základnímu algoritmu raytracingu
  - např. pokud $L$ je $\delta$-funkce, pak máme bodové zdroje světla a z integrálu je konečná suma
  - pokud $R$ je $\delta$-funkce, tak máme jen dokonalé zrcadlové odrazy
- umožňuje simulovat další efekty:
  - plošné zdroje světla, neostré stíny
  - průsvitnost
  - neostré odrazy - různé úrovně lesku (*gloss*)
  - konečná hloubka ostrosti
  - rozmazání pohybem
- importance sampling + jittering $\Longrightarrow$ antialiasing

# Distribuovaný raytracing 4/8

## Neostré stíny

- distribuce stínových paprsků v prostorovém úhlu plošného světelného zdroje



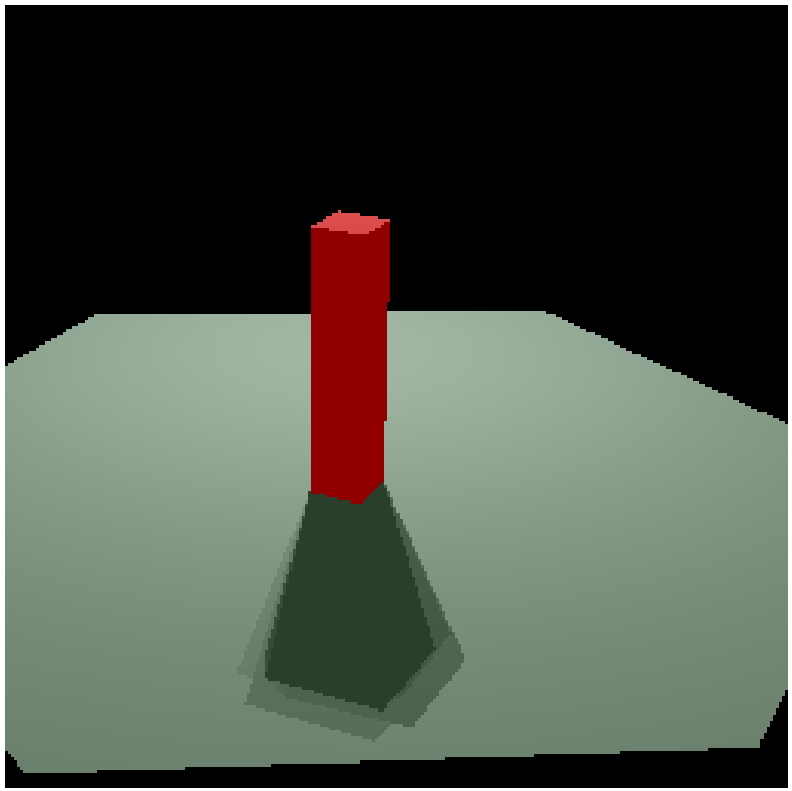


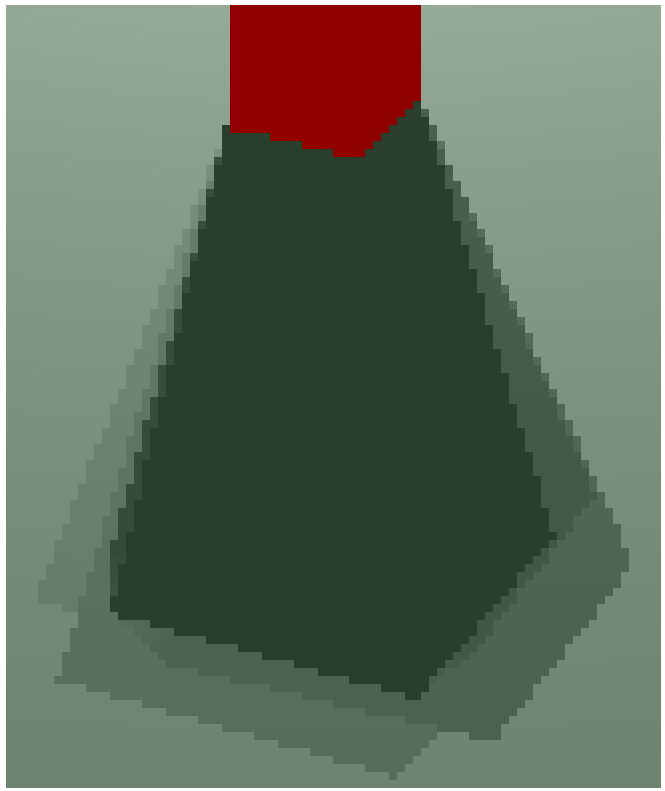
Tradiční základní raytracing

# Distribuovaný raytracing 4/8

## Neostré stíny

- distribuce stínových paprsků v prostorovém úhlu plošného světelného zdroje



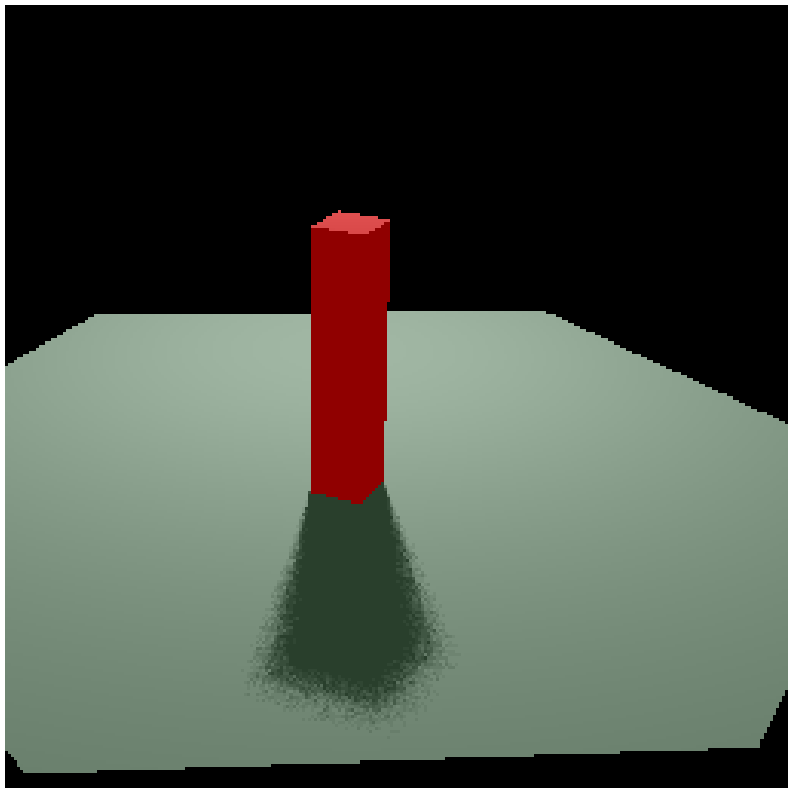


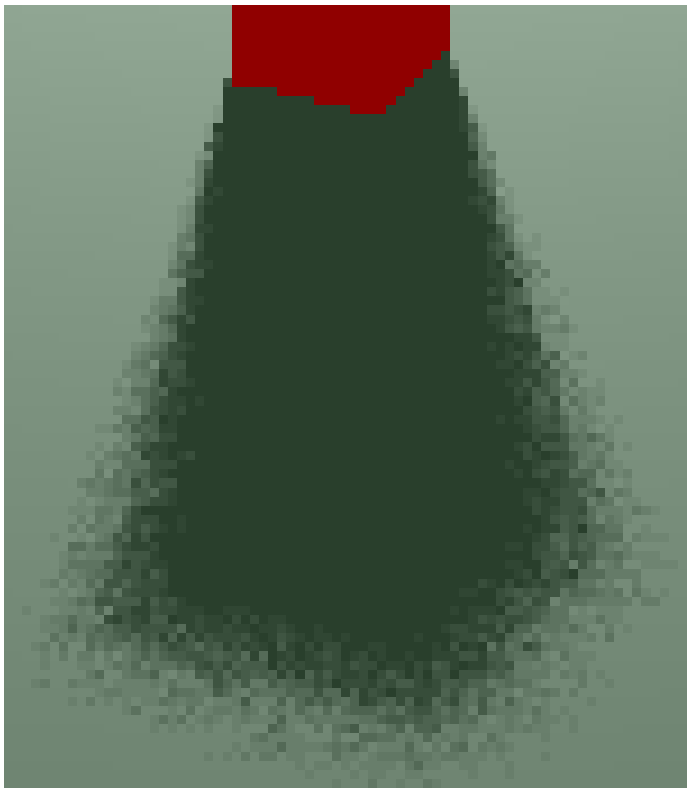
Uniformní vzorkování

# Distribuovaný raytracing 4/8

## Neostré stíny

- distribuce stínových paprsků v prostorovém úhlu plošného světelného zdroje



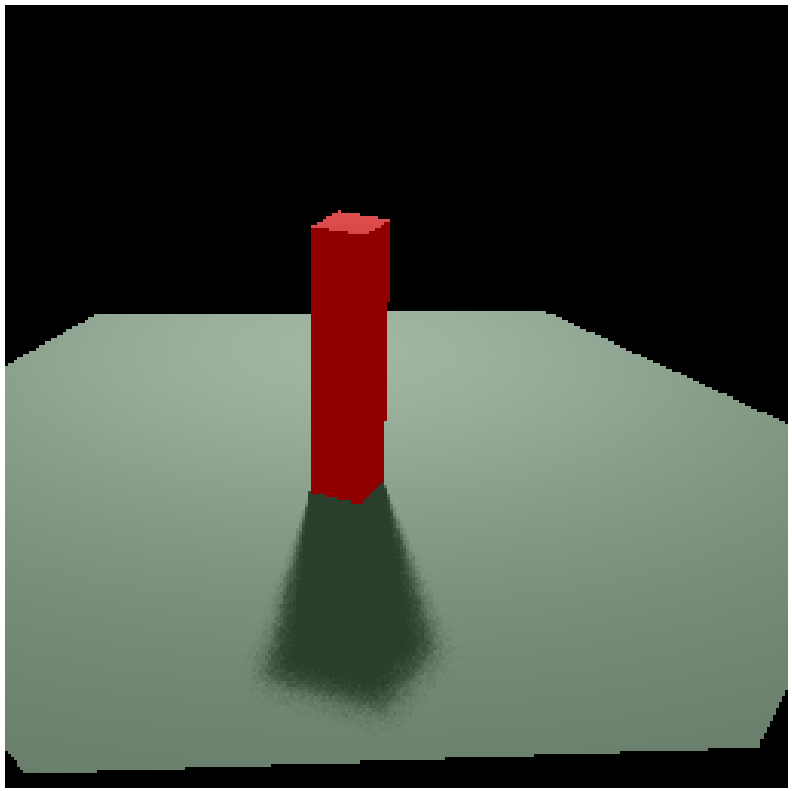


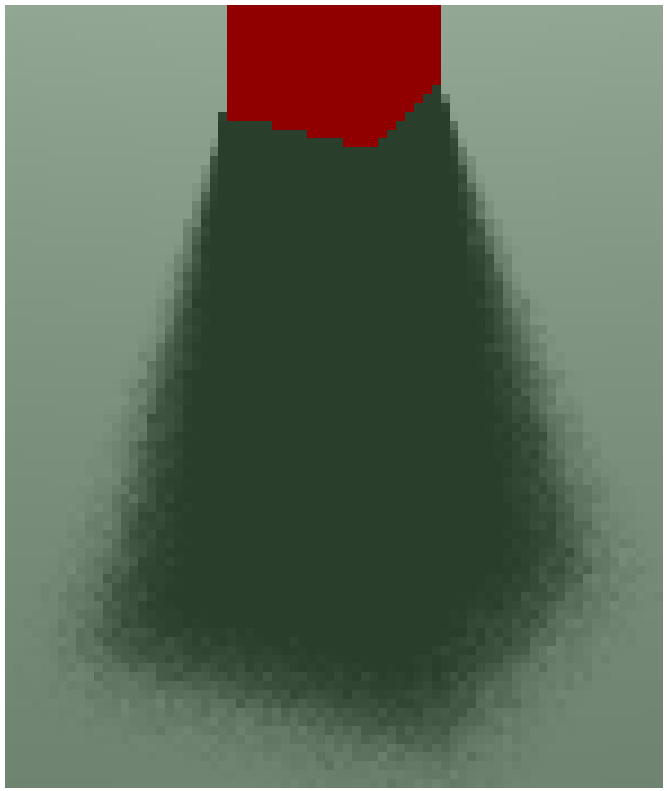
DR 10 paprsků

# Distribuovaný raytracing 4/8

## Neostré stíny

- distribuce stínových paprsků v prostorovém úhlu plošného světelného zdroje



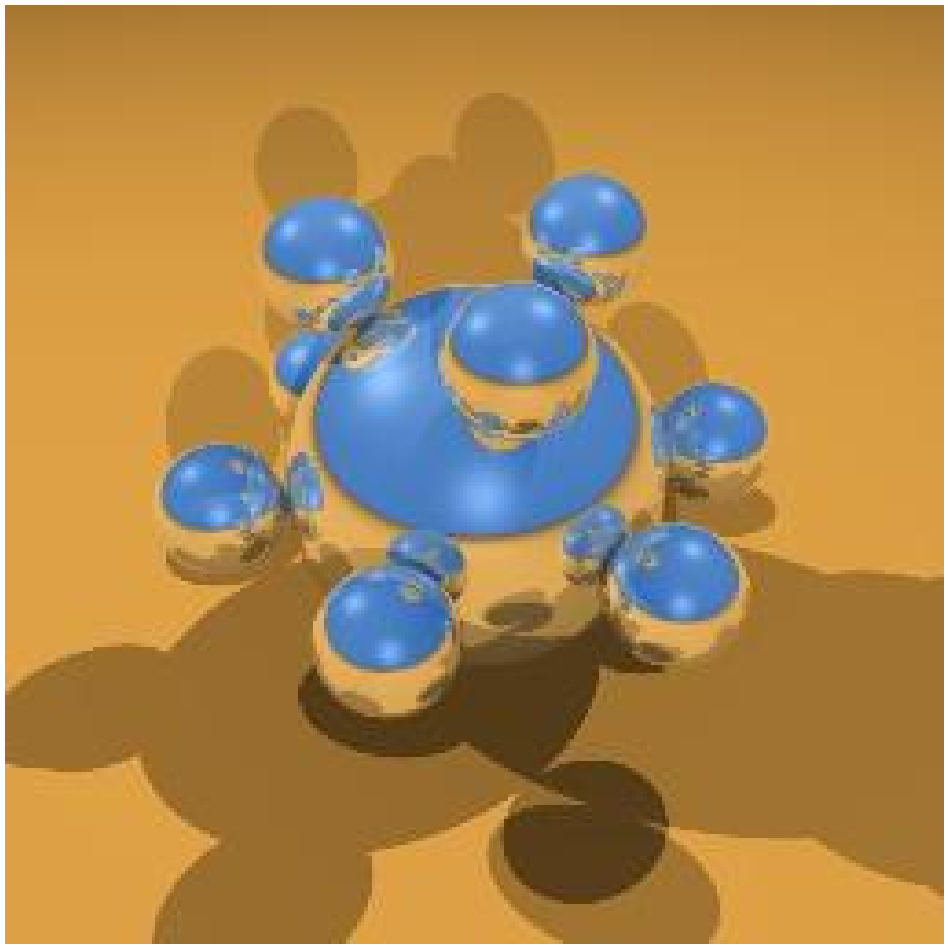


DR 50 paprsků

# Distribuovaný raytracing 4/8

## Neostré stíny

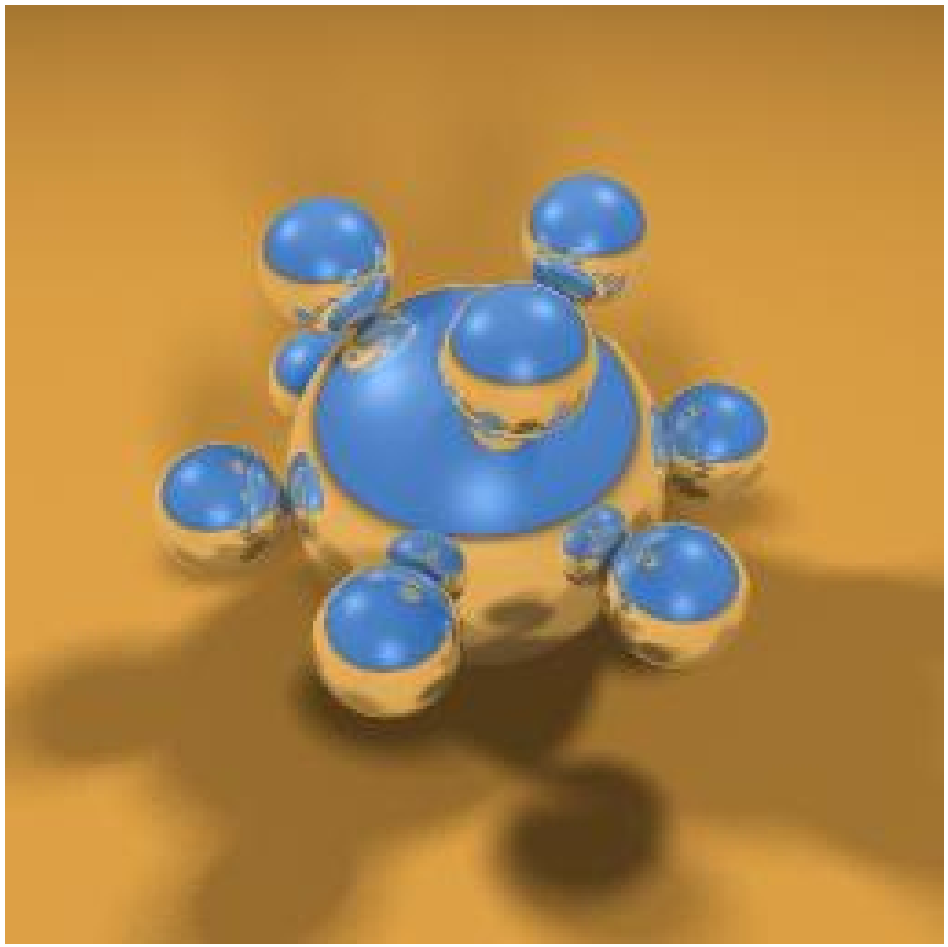
- distribuce stínových paprsků v prostorovém úhlu plošného světelného zdroje



# Distribuovaný raytracing 4/8

## Neostré stíny

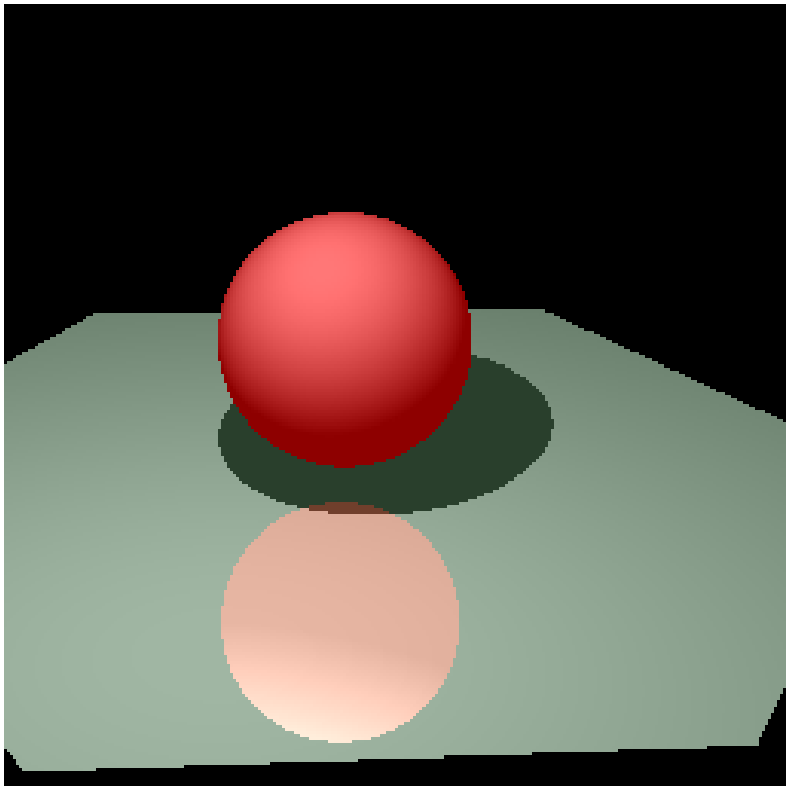
- distribuce stínových paprsků v prostorovém úhlu plošného světelného zdroje

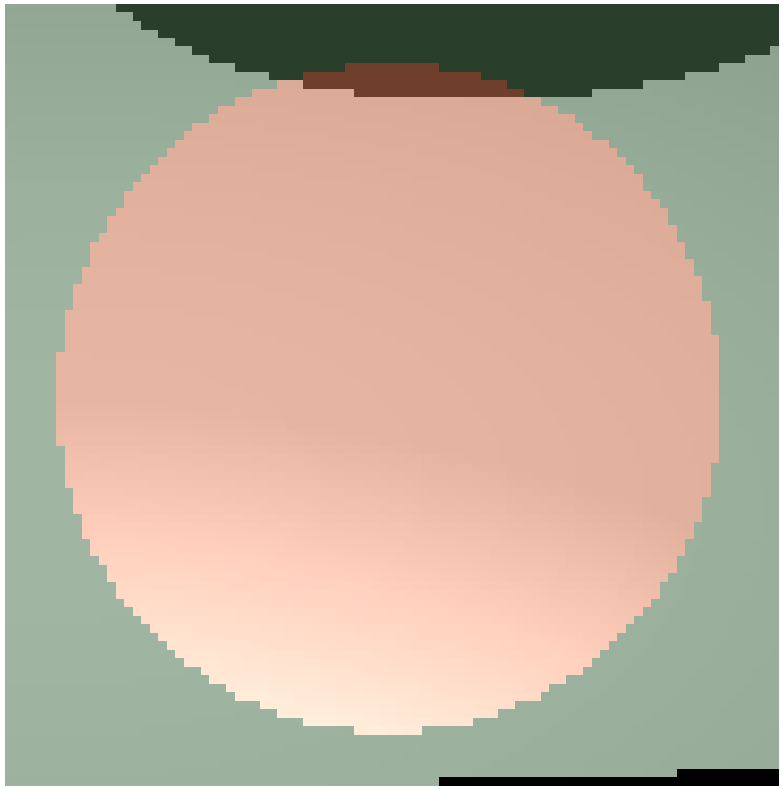


# Distribuovaný raytracing 5/8

## Neostré odrazy

- distribuce odražených paprsků kolem úhlu (dokonalého) odrazu



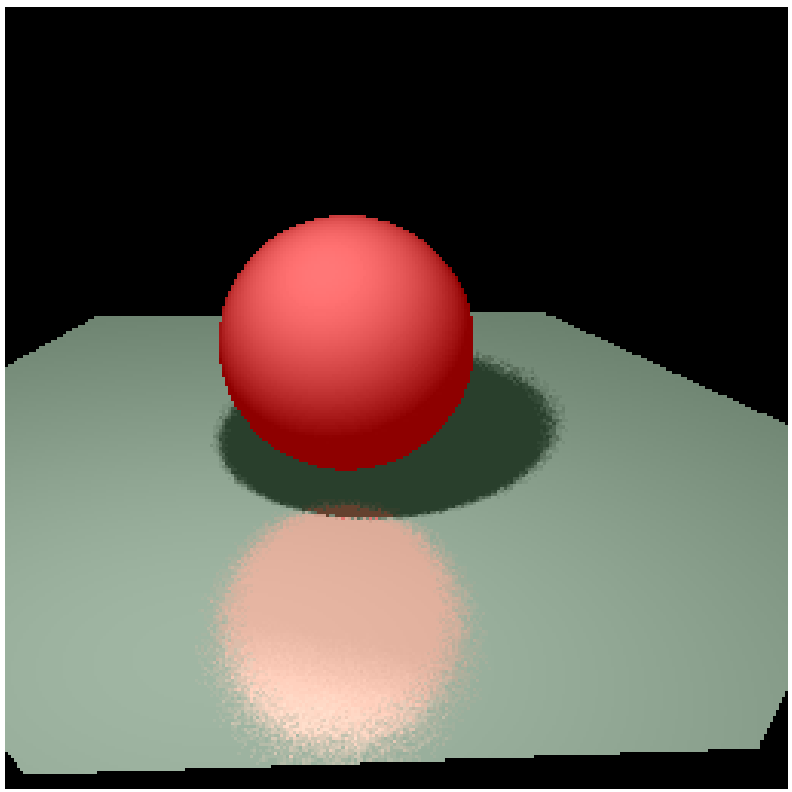


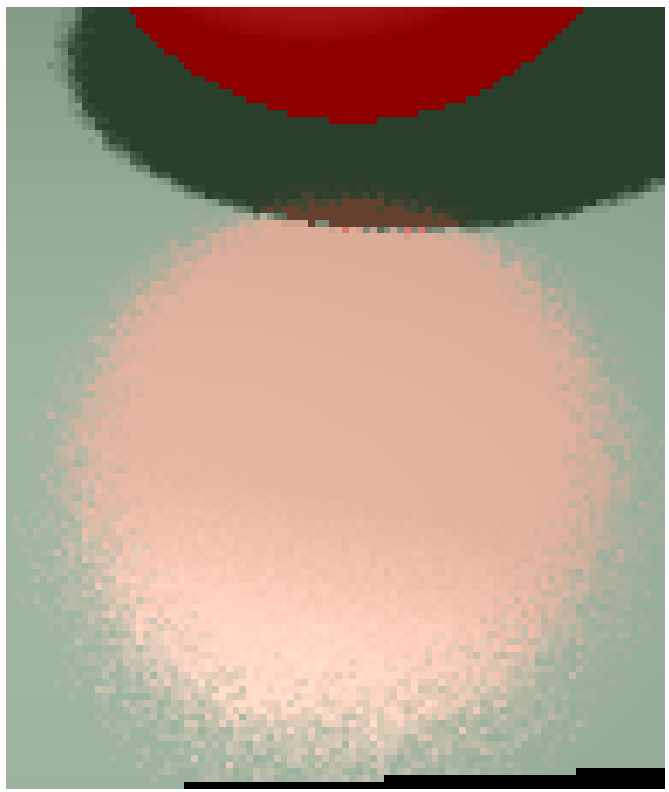
Tradiční základní raytracing

# Distribuovaný raytracing 5/8

## Neostré odrazy

- distribuce odražených paprsků kolem úhlu (dokonalého) odrazu



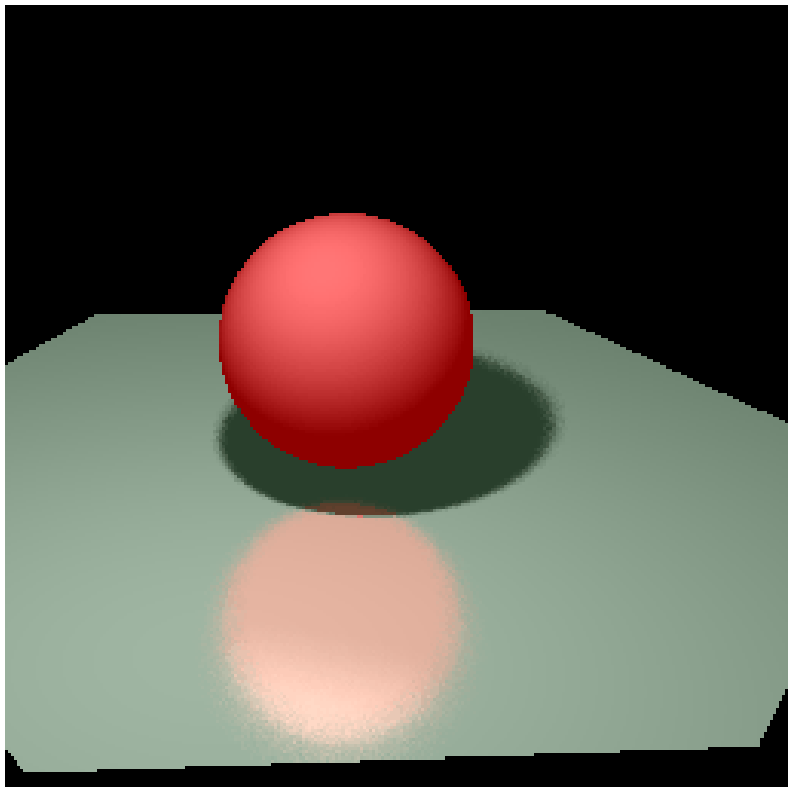


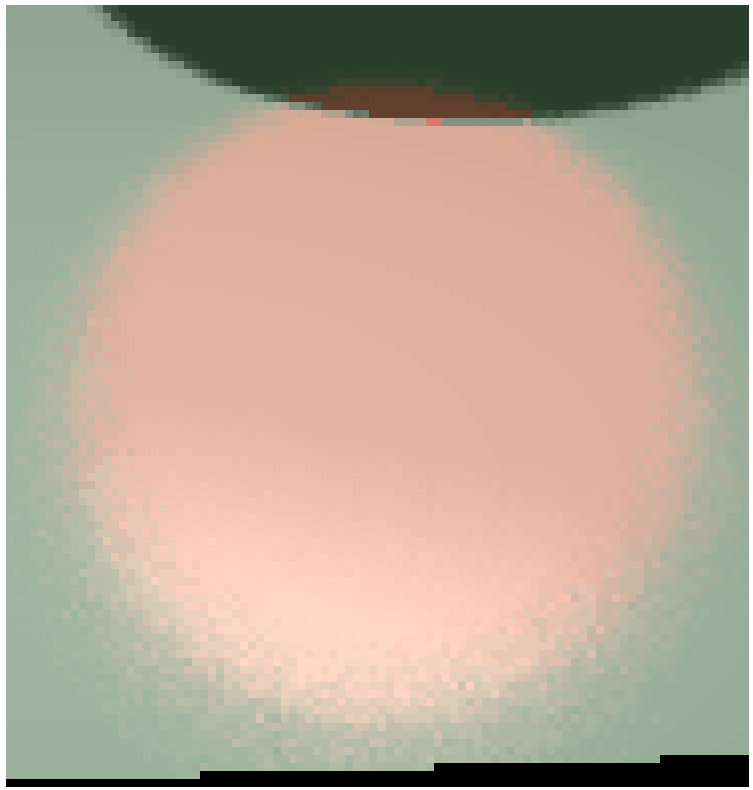
DR 10 paprsků

# Distribuovaný raytracing 5/8

## Neostré odrazy

- distribuce odražených paprsků kolem úhlu (dokonalého) odrazu



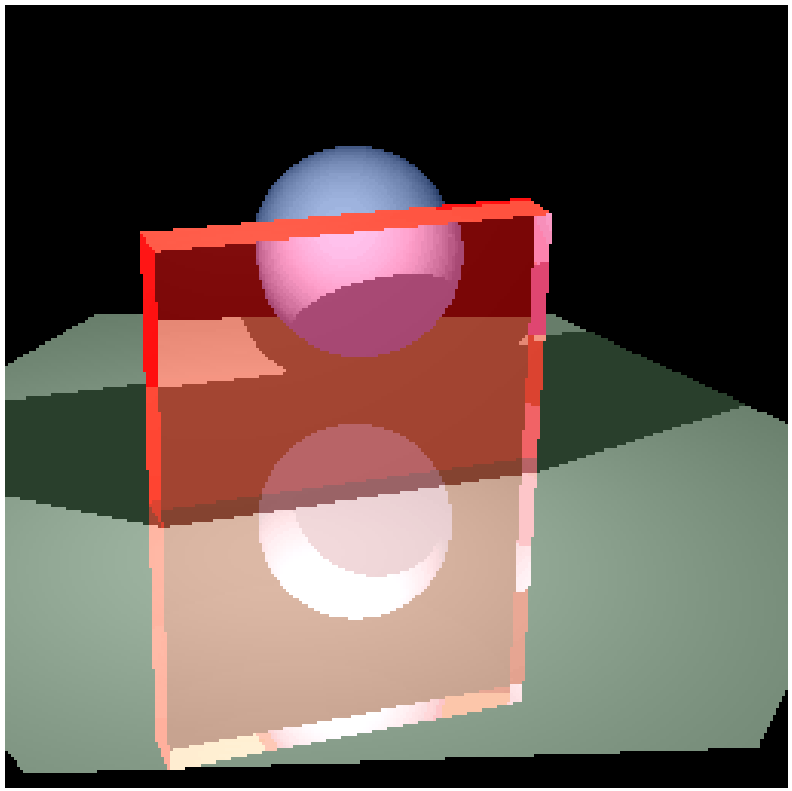


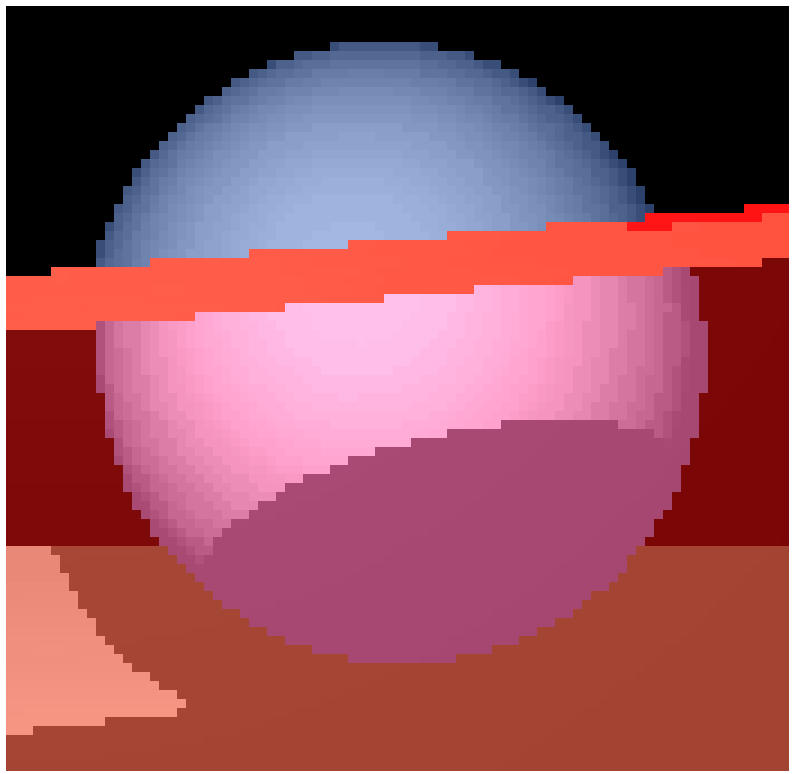
DR 50 paprsků

# Distribuovaný raytracing 6/8

## Průsvitnost

- distribuce lomených paprsků kolem hlavního směru úhlu lomeného paprsku - funkce „propustnosti“ místo odrazivosti $R$



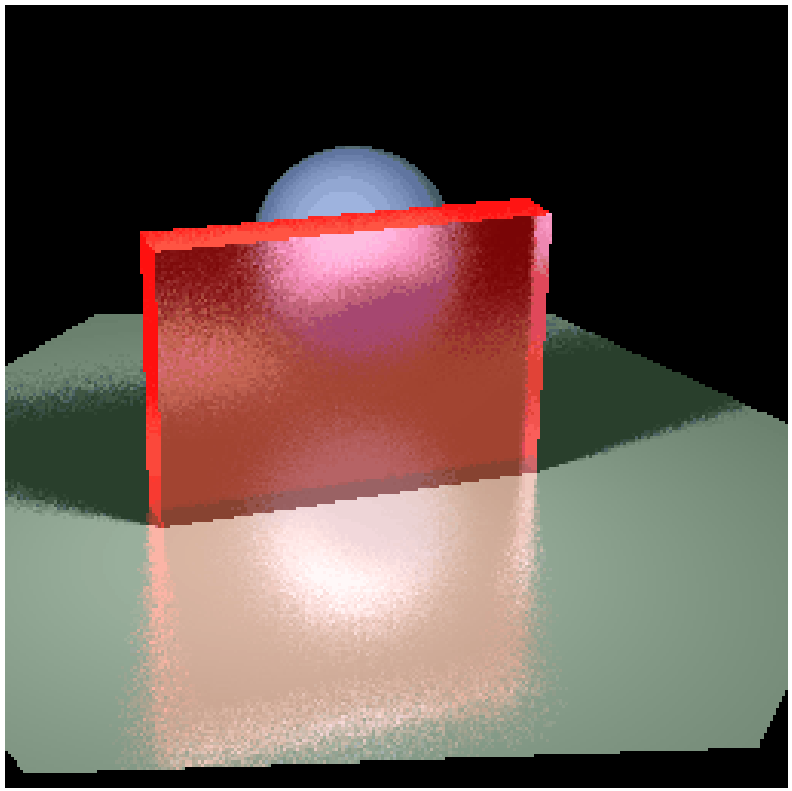


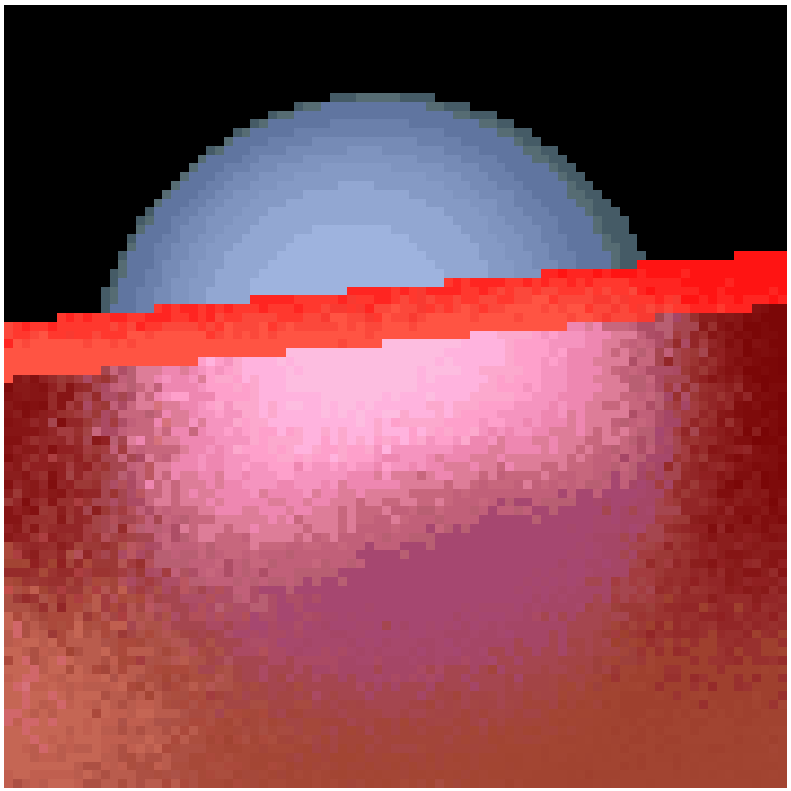
Tradiční základní raytracing

# Distribuovaný raytracing 6/8

## Průsvitnost

- distribuce lomených paprsků kolem hlavního směru úhlu lomeného paprsku - funkce „propustnosti“ místo odrazivosti $R$



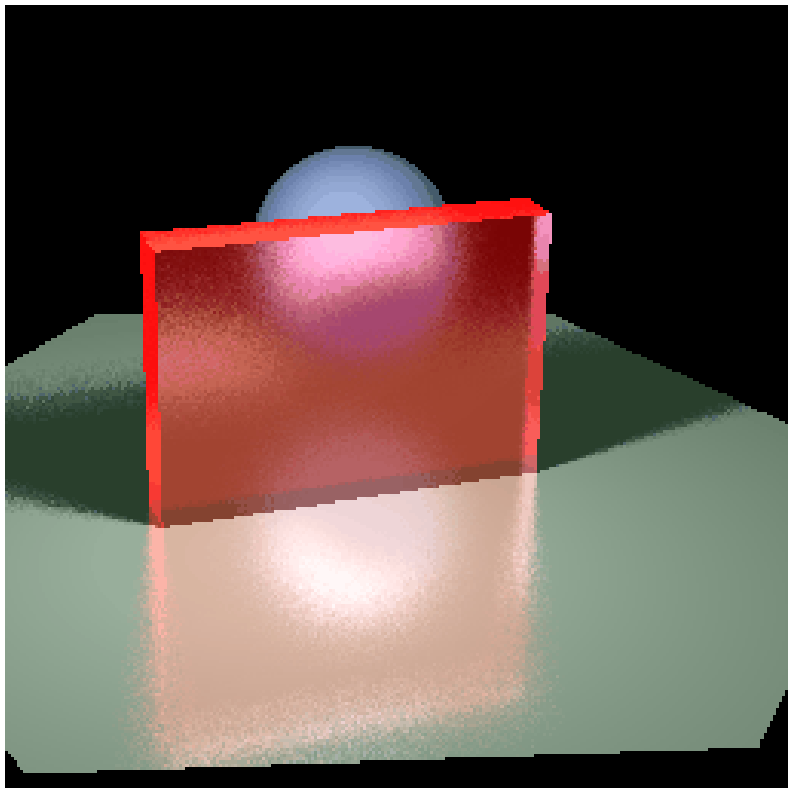


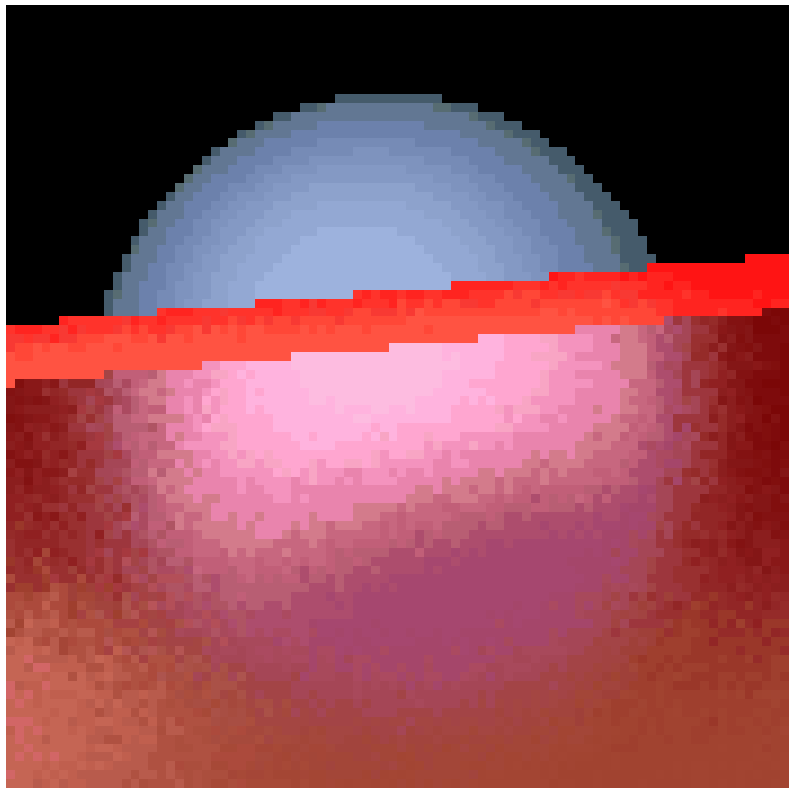
DR 10 paprsků

# Distribuovaný raytracing 6/8

## Průsvitnost

- distribuce lomených paprsků kolem hlavního směru úhlu lomeného paprsku - funkce „propustnosti“ místo odrazivosti $R$





DR 20 paprsků

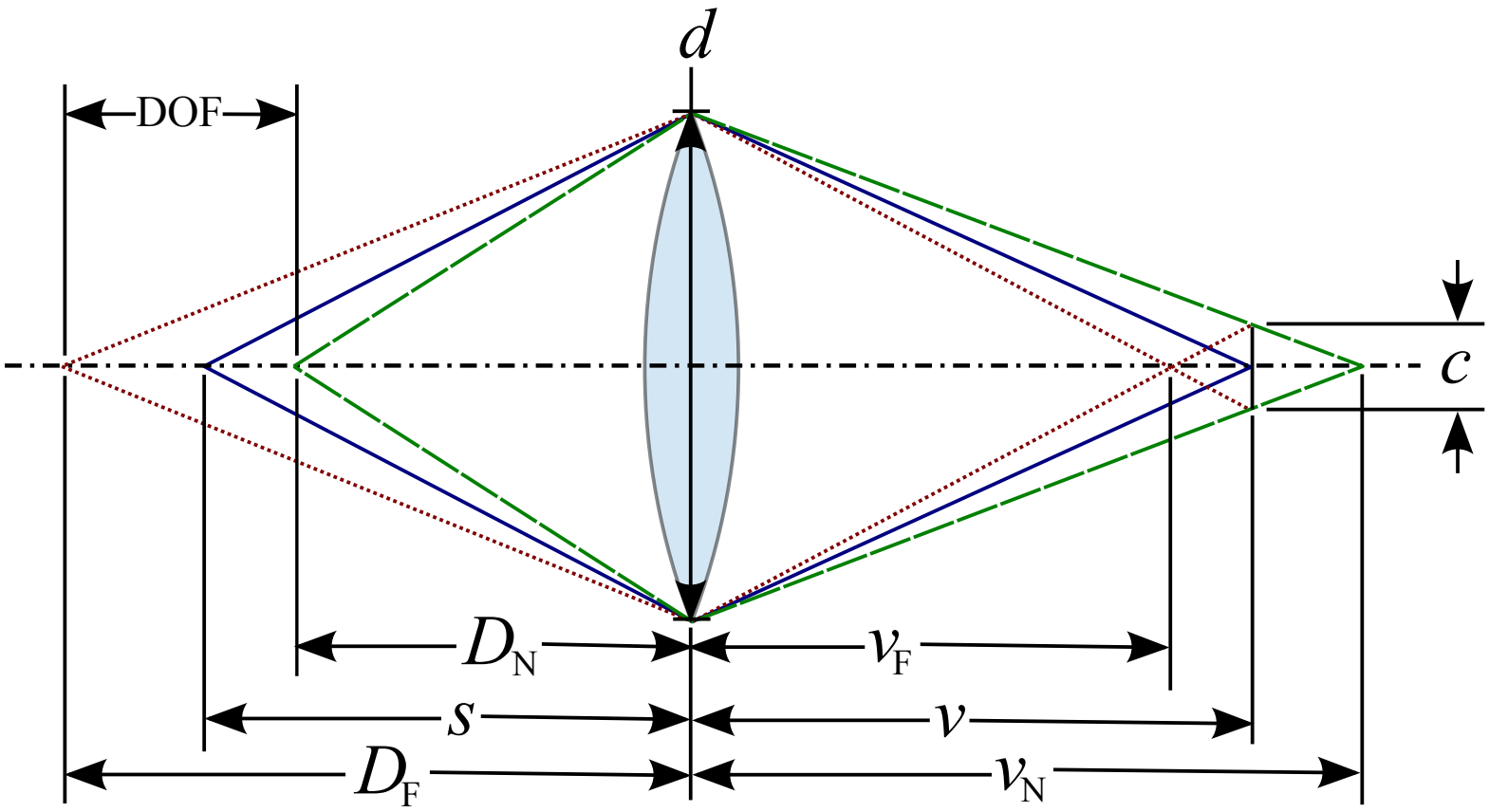
# Distribuovaný raytracing 7/8

## Hloubka ostrosti - úvod

- fotografie - všechny objekty ostré, jen jestliže fotíme nekonečně malou dírkou
- ve skutečnosti jsou ostré jen objekty v určité vzdálenosti od průmětny - rovina ostrosti rovnoběžná s průmětnou
- konečný průřez čočky (clona, *aperture*) $\Longrightarrow$ body ve scéně se zobrazí jako kruhy (*circle of confusion*)
- **hloubka ostrosti** (*depth of field*) = rozsah vzdáleností, kde jsou tyto kruhy menší než pixel na průmětně (resp. kde jsou stále pozorovatelné jako body)
- pokud se chceme přiblížit dojmu fotografie, implementujeme i hloubku ostrosti

# Distribuovaný raytracing 7/8

## Hloubka ostrosti - schéma



# Distribuovaný raytracing 7/8

## Hloubka ostrosti - implementace

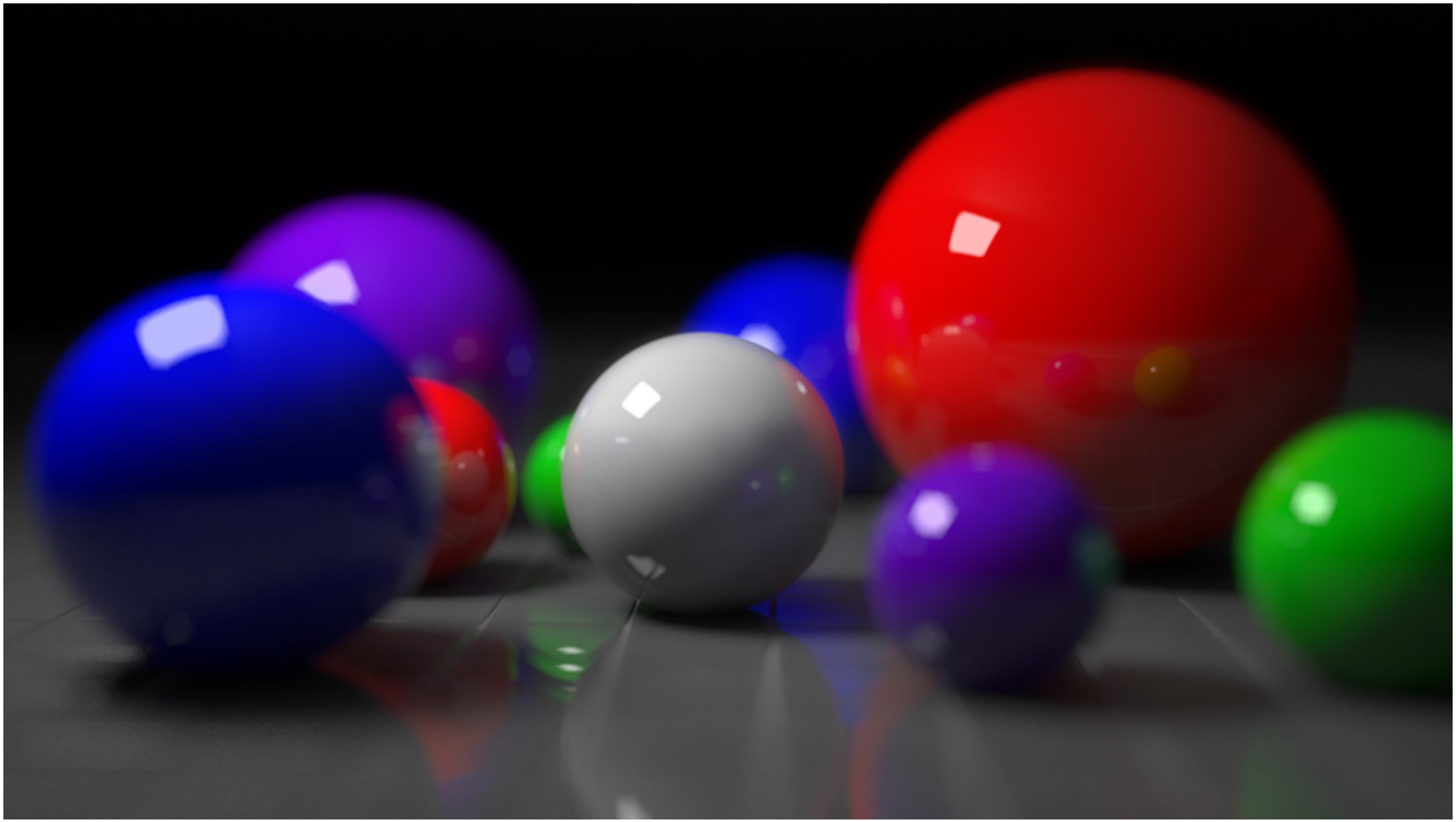
- máme zadánu vzdálenost roviny ostrosti a velikost clony $d$
- vyšleme primární obrazový paprsek z pixelu $\boldsymbol{P}$, jeho průsečík s rovinou ostrosti označíme $\boldsymbol{Q}$
- vyšleme další paprsky náhodně rozmístěné v kruhu o velikosti $d$ se středem v $\boldsymbol{P}$ tak, aby procházely bodem $\boldsymbol{Q}$
- čím větší $d$, tím menší hloubka ostrosti

# Distribuovaný raytracing 7/8

## Hloubka ostrosti - příklady

# Distribuovaný raytracing 7/8

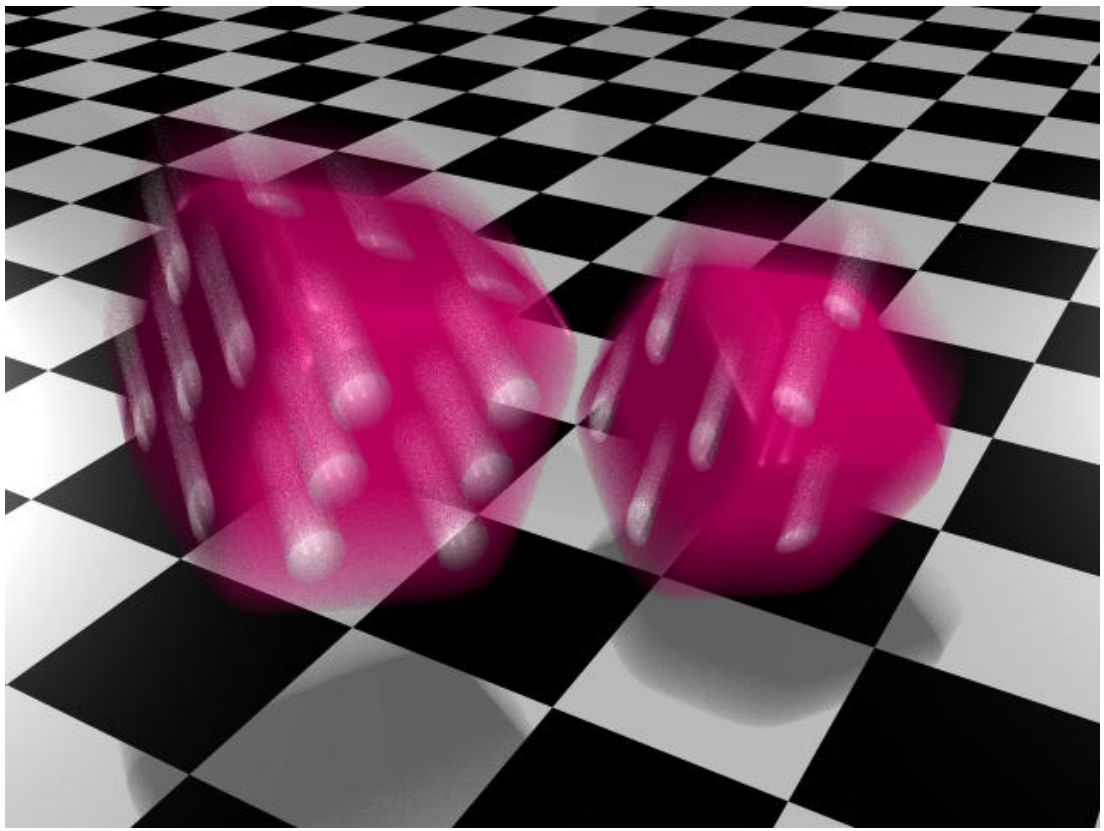
## Hloubka ostrosti - příklady



# Distribuovaný raytracing 8/8

## Rozmazání pohybem

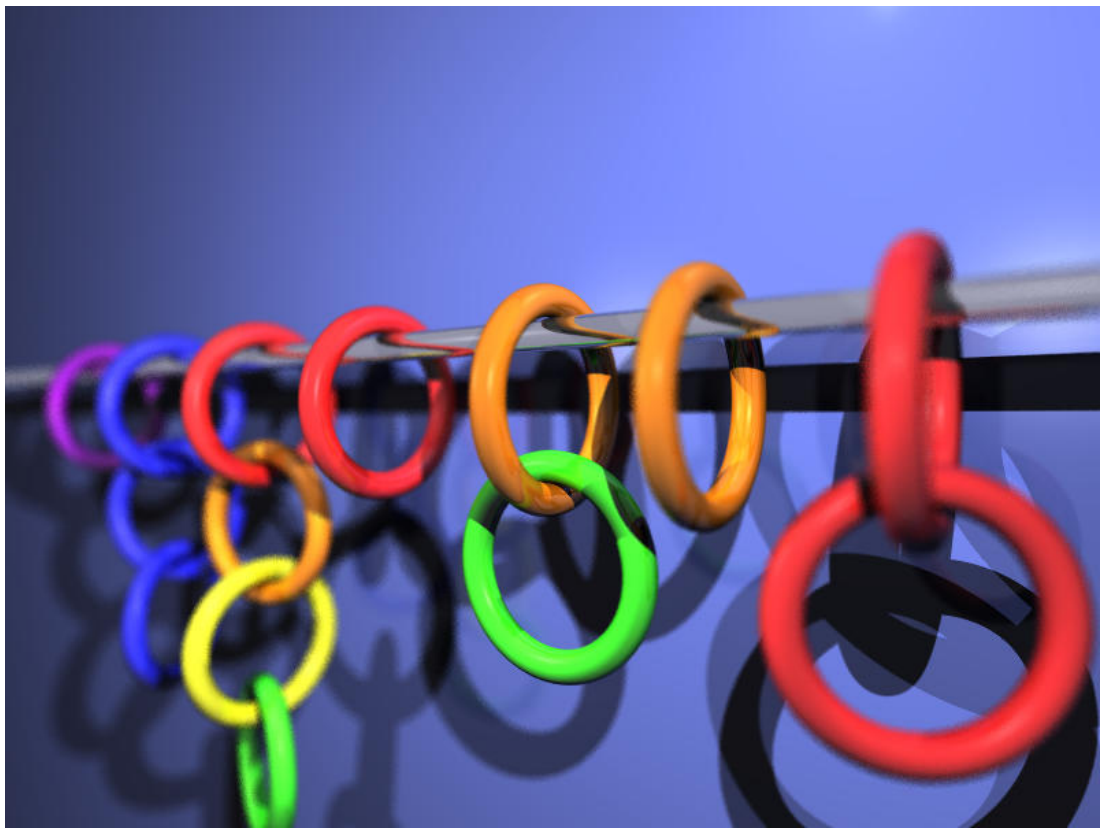
- polohy objektů jsou funkce času
- stochastické vzorkování paprsků nejen v prostoru, ale i v čase



# Distribuovaný raytracing 8/8

## Rozmazání pohybem

- polohy objektů jsou funkce času
- stochastické vzorkování paprsků nejen v prostoru, ale i v čase



# Obsah

# Metoda fotonových map

- **dvoufázová metoda**

1. vyzařování fotonů ze světelných zdrojů podle rozložení zářivého toku daného zdroje
   - fotony se mohou ve scéně odrazit, lomit, absorbovat
   - při interakci fotonu s objektem (kromě zrcadlového odrazu) je jeho příspěvek uložen do trojrozměrné mřížky - *fotonové mapy*
   - pro každý foton se ukládá: pozice, zářivý tok, příchozí směr
   - pro fotony tvořící **kaustiky** je vytvořena speciální fotonová mapa s vyšším rozlišením
2. sledováním paprsku se vykresluje scéna
   - při dopadu paprsku na objekt se vyhodnotí osvětlovací model
   - informace o osvětlení poskytují nejbližší buňky fotonové mapy

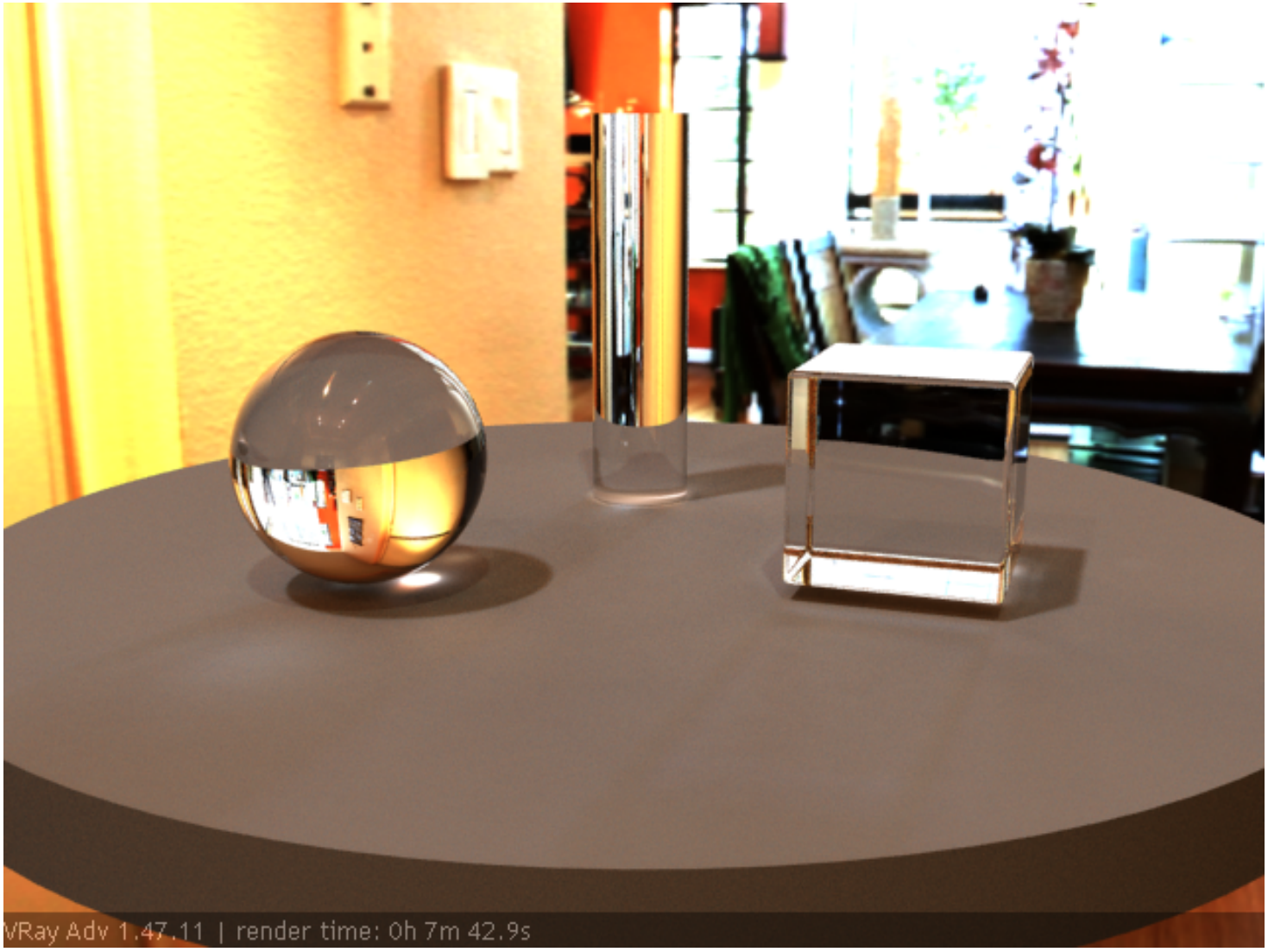
# Metoda fotonových map

## Vlastnosti

- lze simulovat všechny druhy přímého a nepřímého osvětlení, difúzních a dokonale i nedokonale zrcadlových odrazů
- nezávisí na geometrii scény
- rychlá metoda, snadná paralelizace
- pro složité scény paměťově náročná
- běžná součást renderovacího software založeného na raytracingu

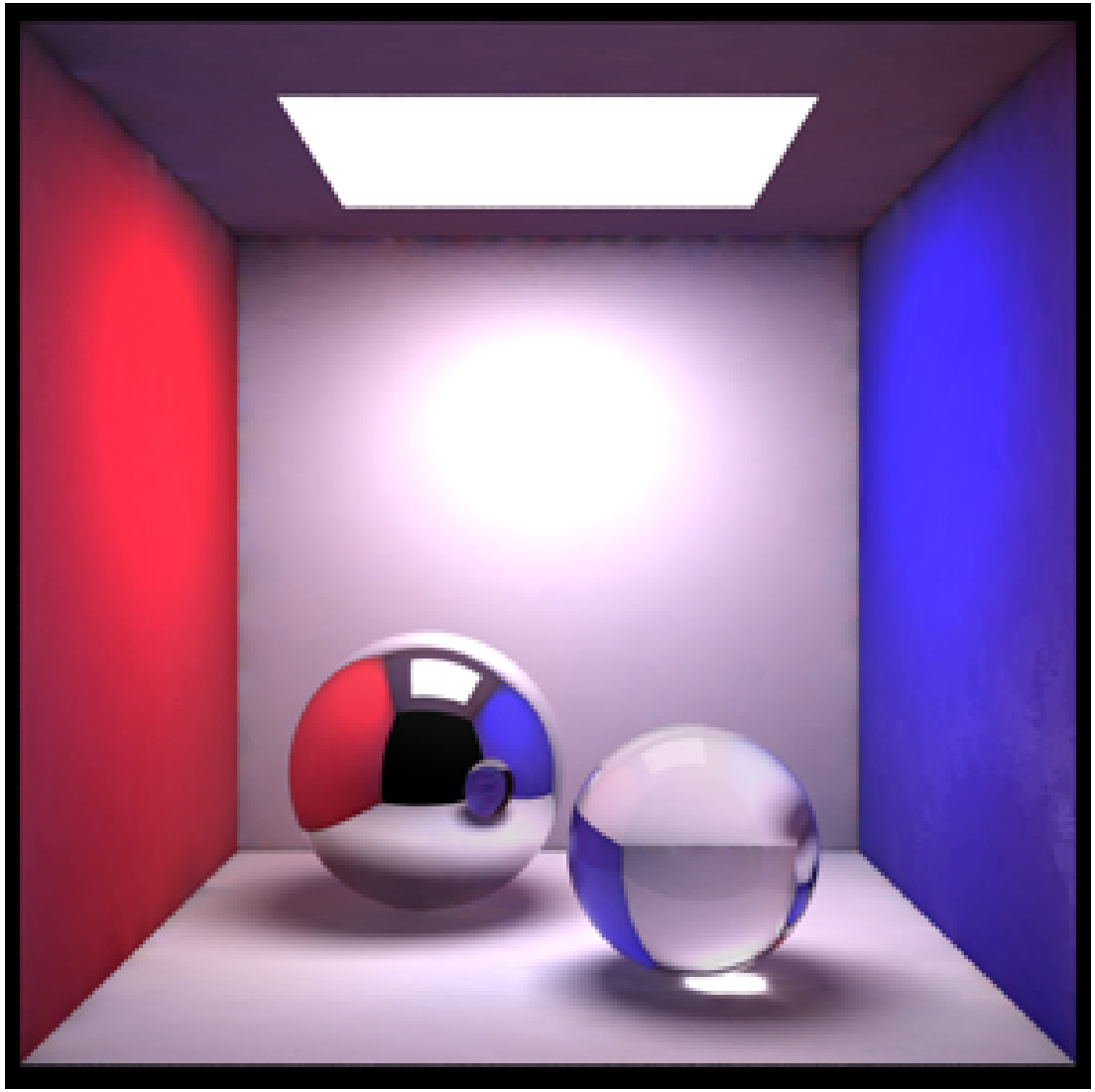
# Metoda fotonových map

## Příklady



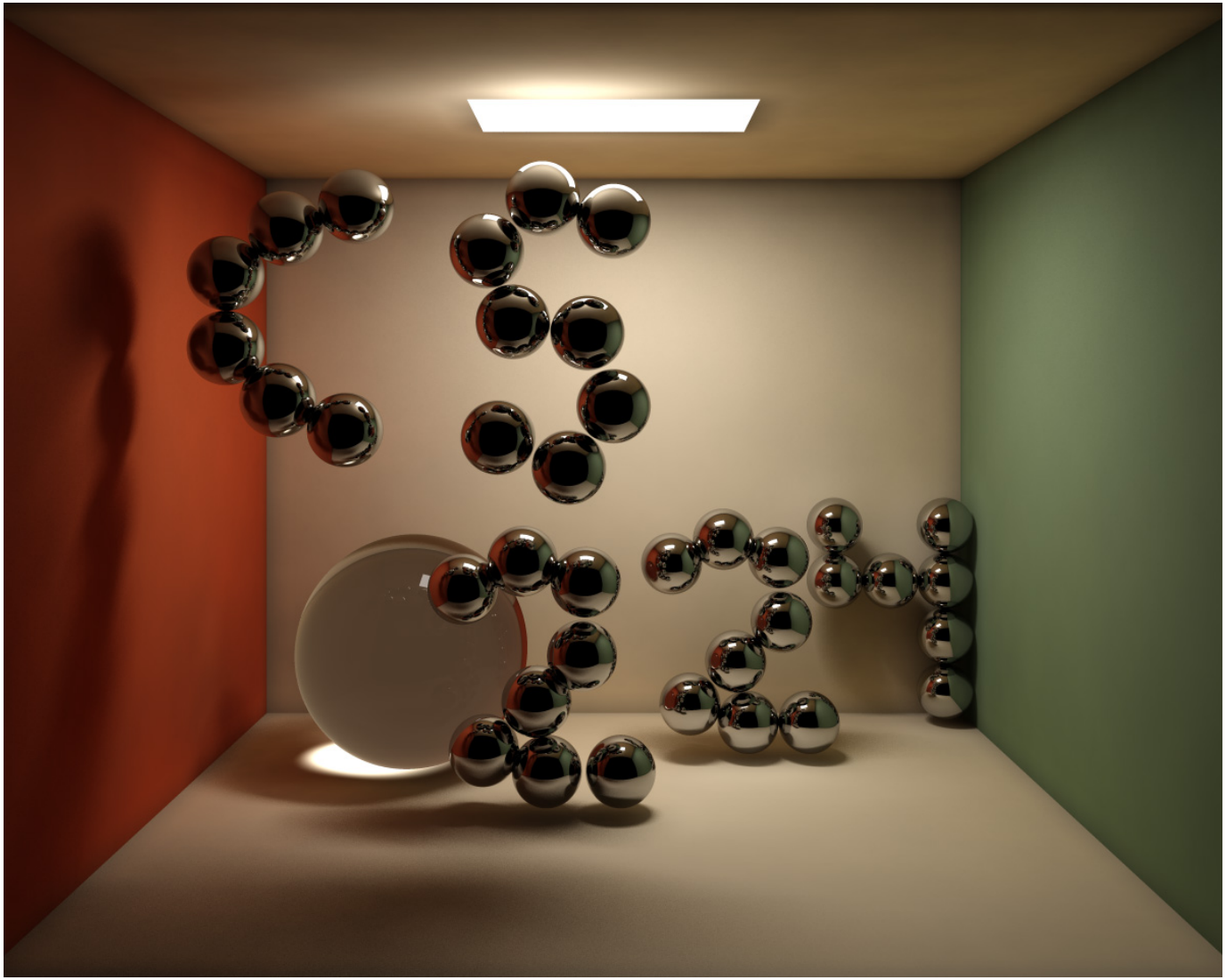
# Metoda fotonových map

## Příklady



# Metoda fotonových map

## Příklady

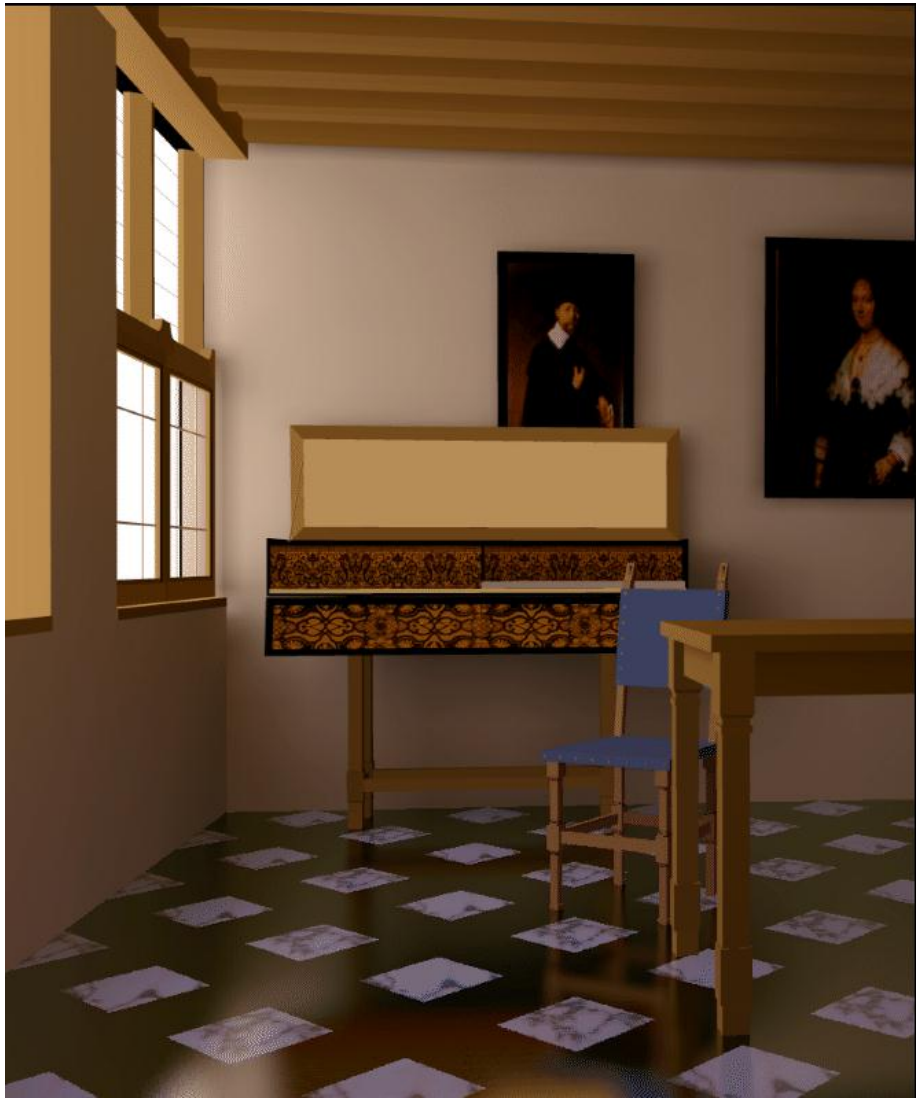


# Metoda radiozity

- řešení zjednodušené zobrazovací rovnice
- **zobrazovací rovnice** - integrální rovnice popisující v každém bodě energetickou bilanci světla (absorbované, vyzářené, odražené)
- rozdělení odrazu světla od povrchu v závislosti na *úhlu dopadu* a *úhlu pohledu* vyjádřeno **distribuční funkcí** (**BRDF** - *Bidirectional Reflectance Distribution Function*)
- zjednodušení: uvažujeme pouze difuzní odrazy - tzv. *radiozitní rovnice*
  $\Longrightarrow$ model plně popisuje difuzní šíření světla ve scéně
- numerické řešení - scéna pokryta sítí, metoda konečných prvků

# Metoda radiozity

## Příklady



# Metoda radiozity

## Příklady

# Metoda radiozity

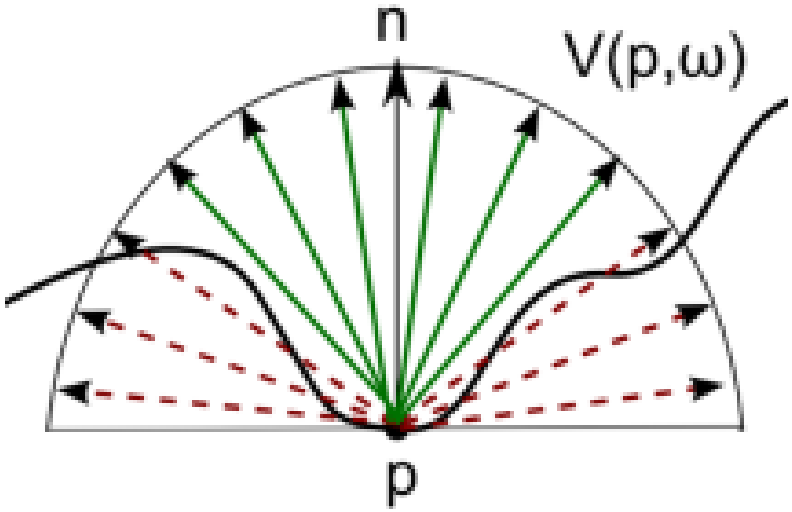
## Příklady

# Ambient occlusion

- pouze simuluje globální osvětlovací model
- vypočítává, v jaké míře na povrch objektu dopadá ambientní osvětlení
- lze předpočítat a výsledek uložit do textury
- vzorkování integrálu

$$AO(\boldsymbol{P},\boldsymbol{n}) = \frac{1}{\pi}\int_{\omega} \boldsymbol{V}(\boldsymbol{p},\omega)\cdot\boldsymbol{n}\mathrm{d}\omega$$

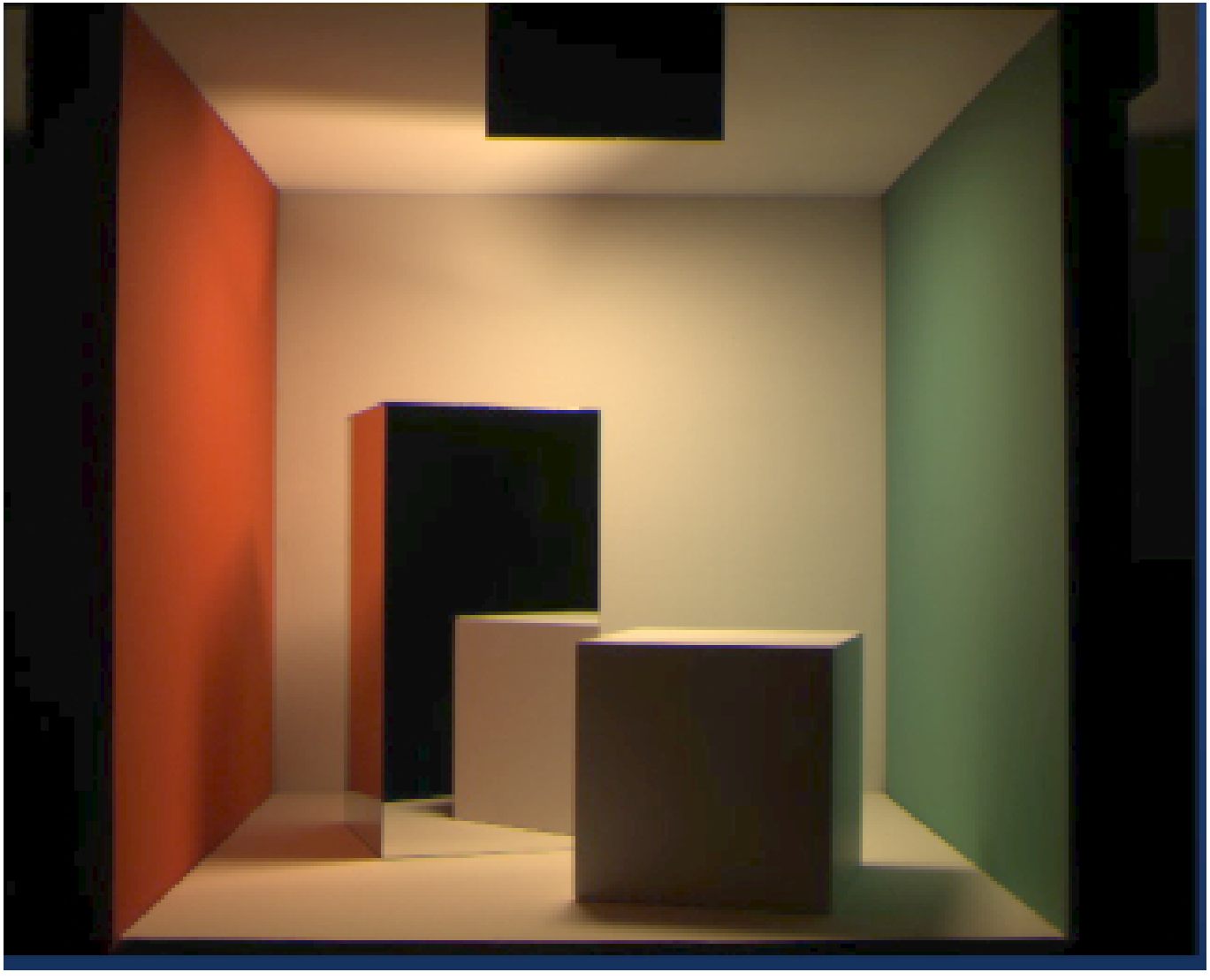
- každý paprsek, který neprotne žádný objekt, přispěje k ambientnímu osvětlení



# Cornellova krabice

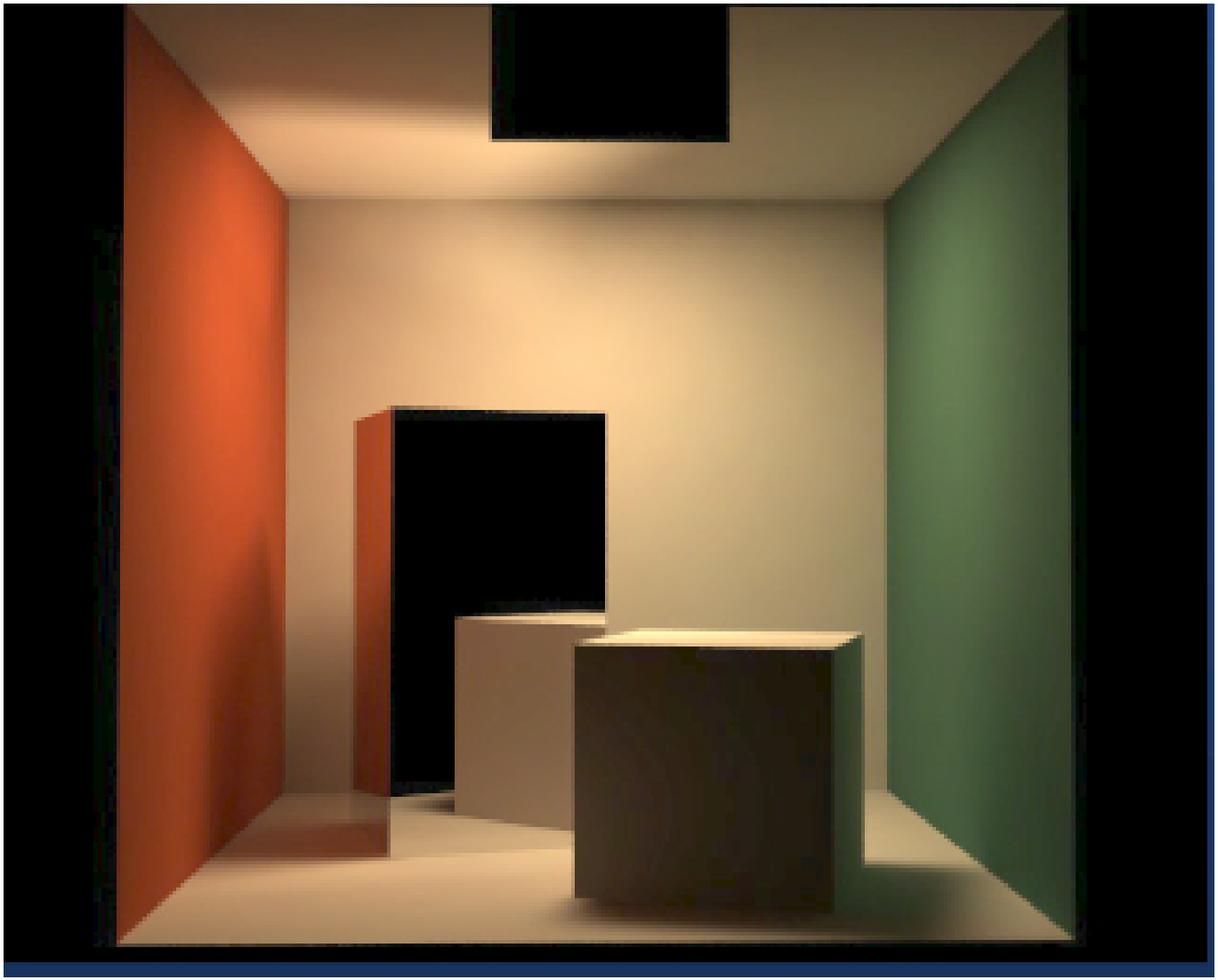
- *Cornell box*
- precizně vyrobená a přesně nasvětlená krabice na Cornell University v Ithace, NY, USA
- referenční scéna pro algoritmy fotorealistického zobrazování
- její parametry (rozměry, barvy, materiály) jsou přesně známy, a proto ji lze přesně modelovat

# Rendering Cornellovy krabice



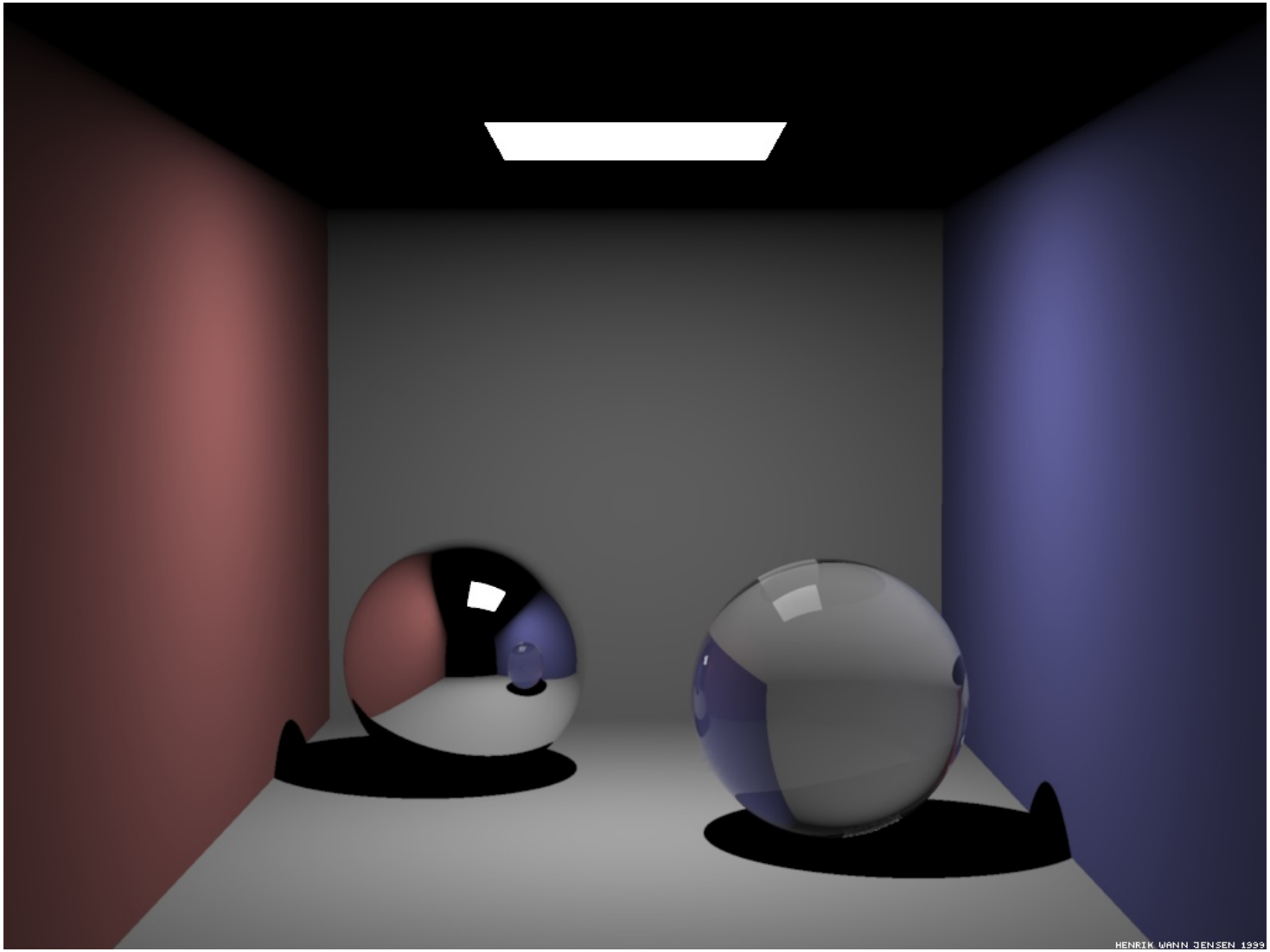
Skutečná Cornellova krabice

# Rendering Cornellovy krabice



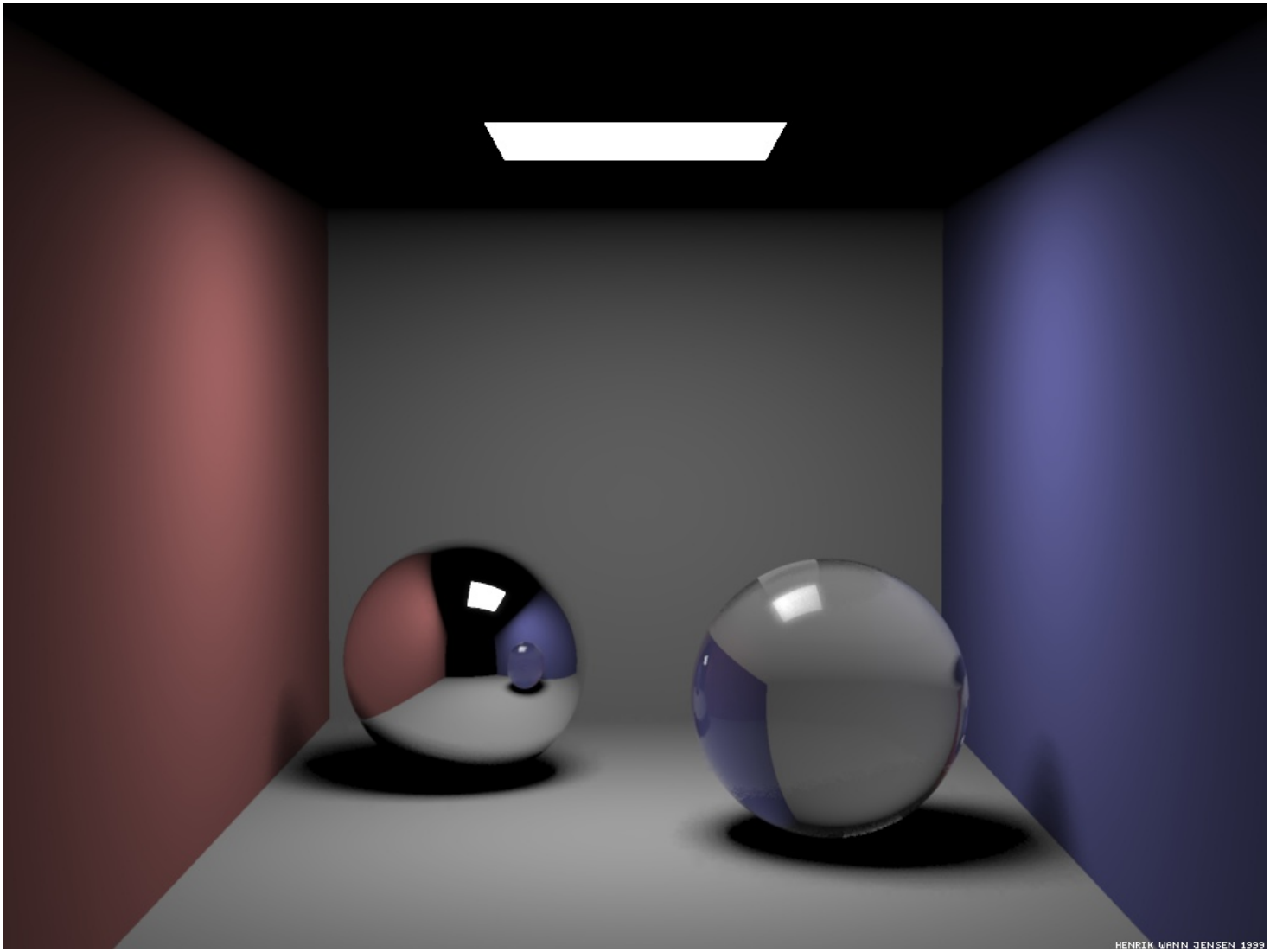
Simulace - radiozita

# Raytracing Cornellovy krabice



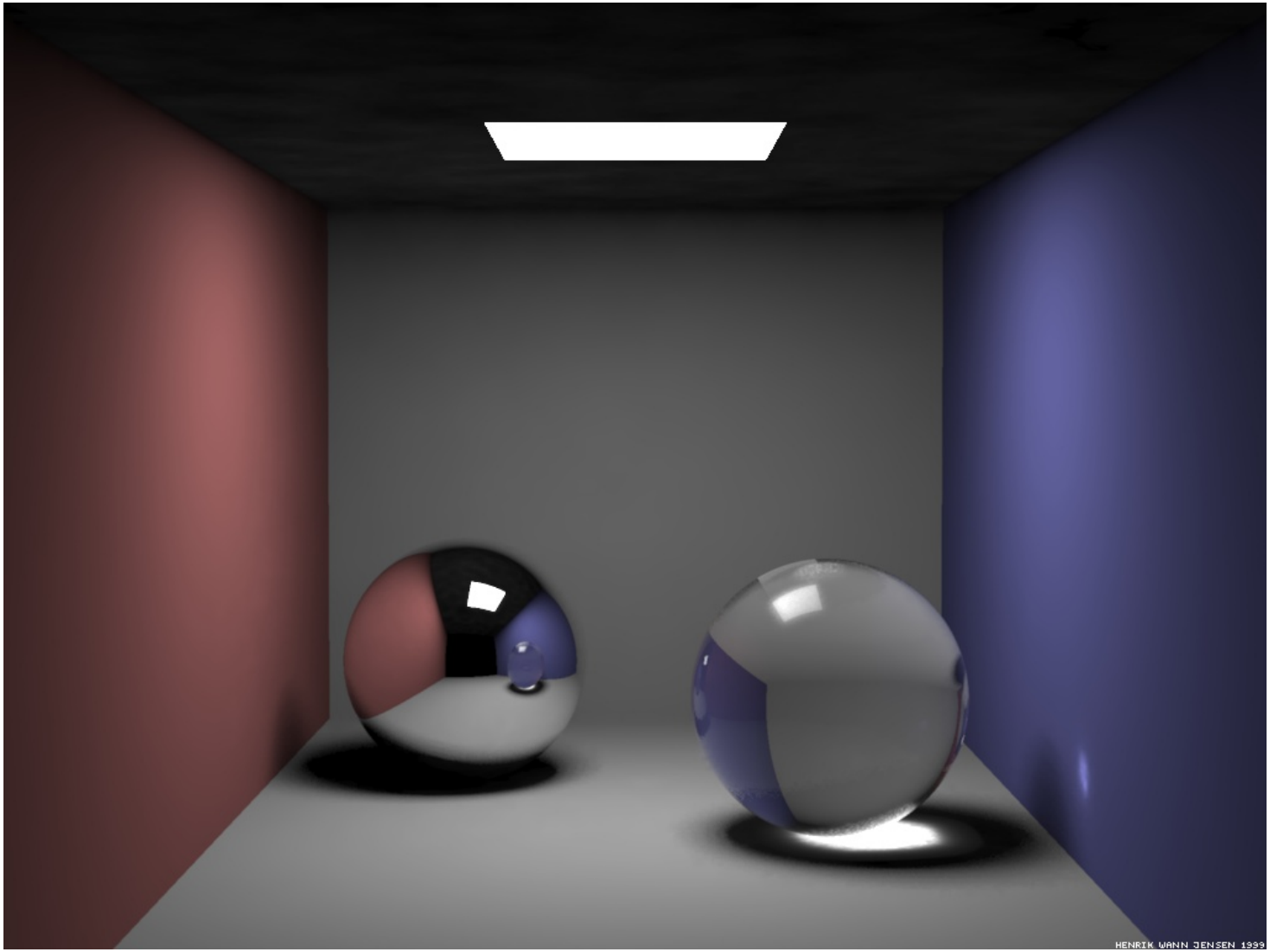

Raytracing

# Raytracing Cornellovy krabice



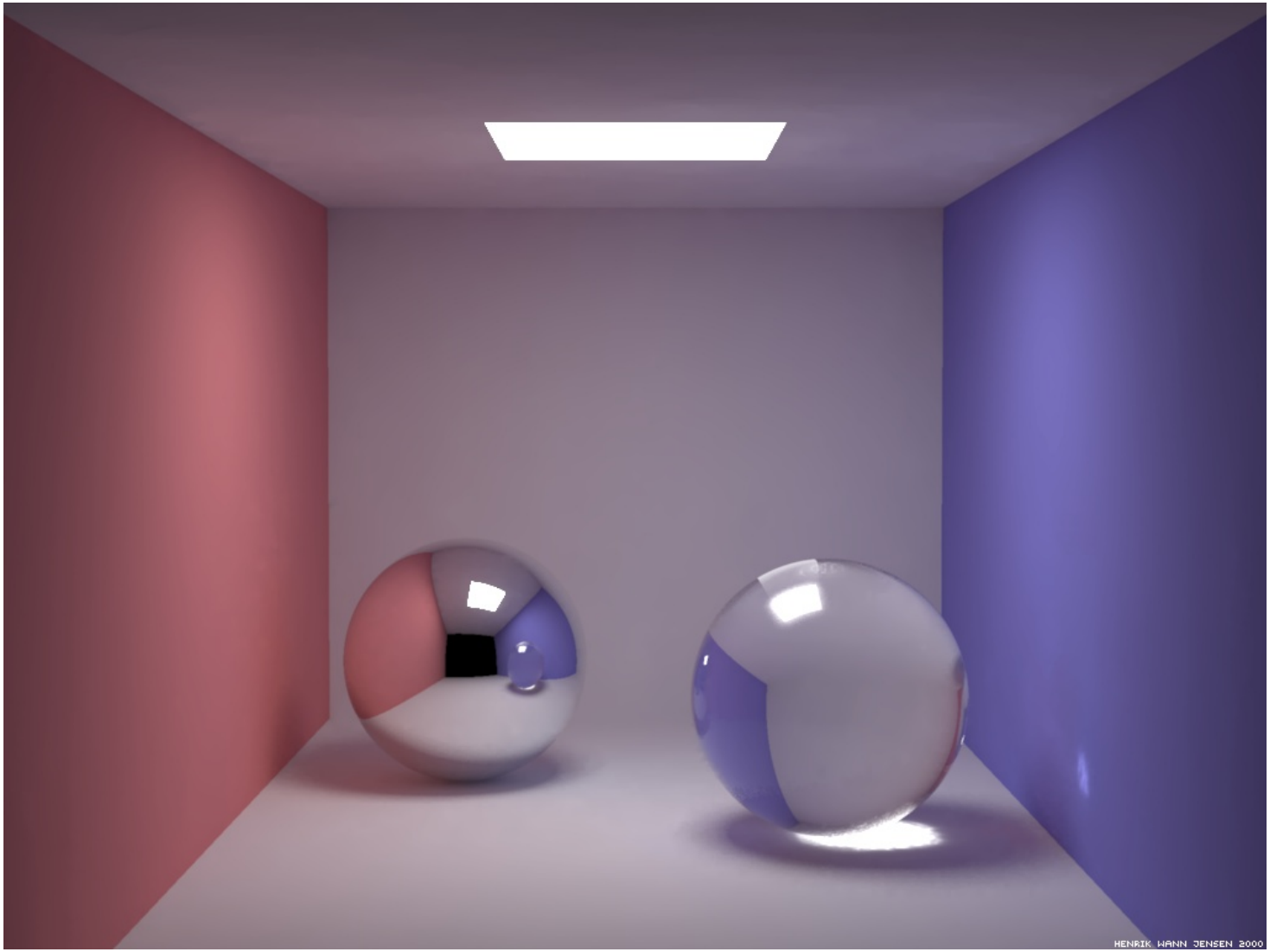

Raytracing + měkké stíny

# Raytracing Cornellovy krabice



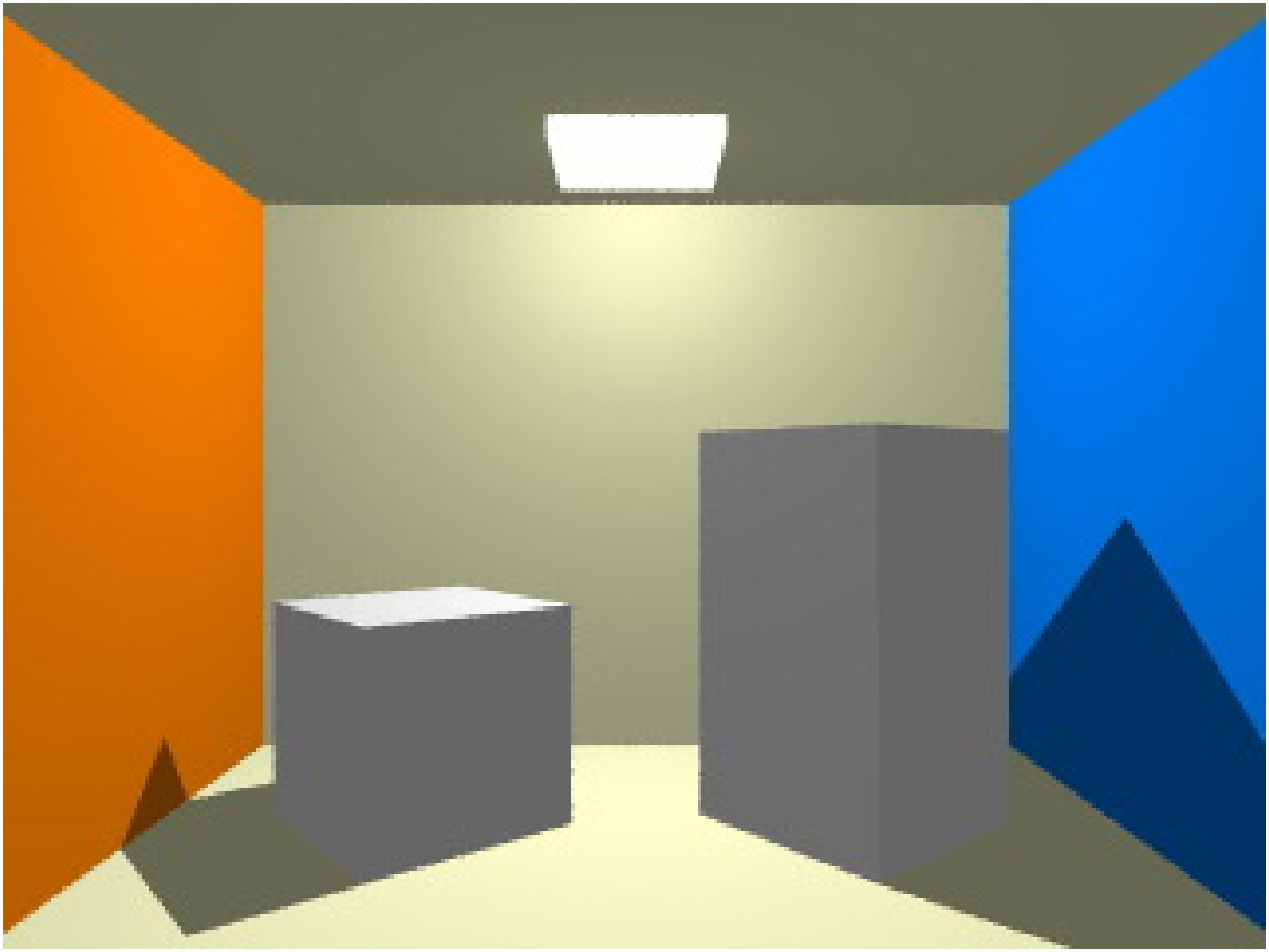

Raytracing + měkké stíny + kaustiky

# Raytracing Cornellovy krabice



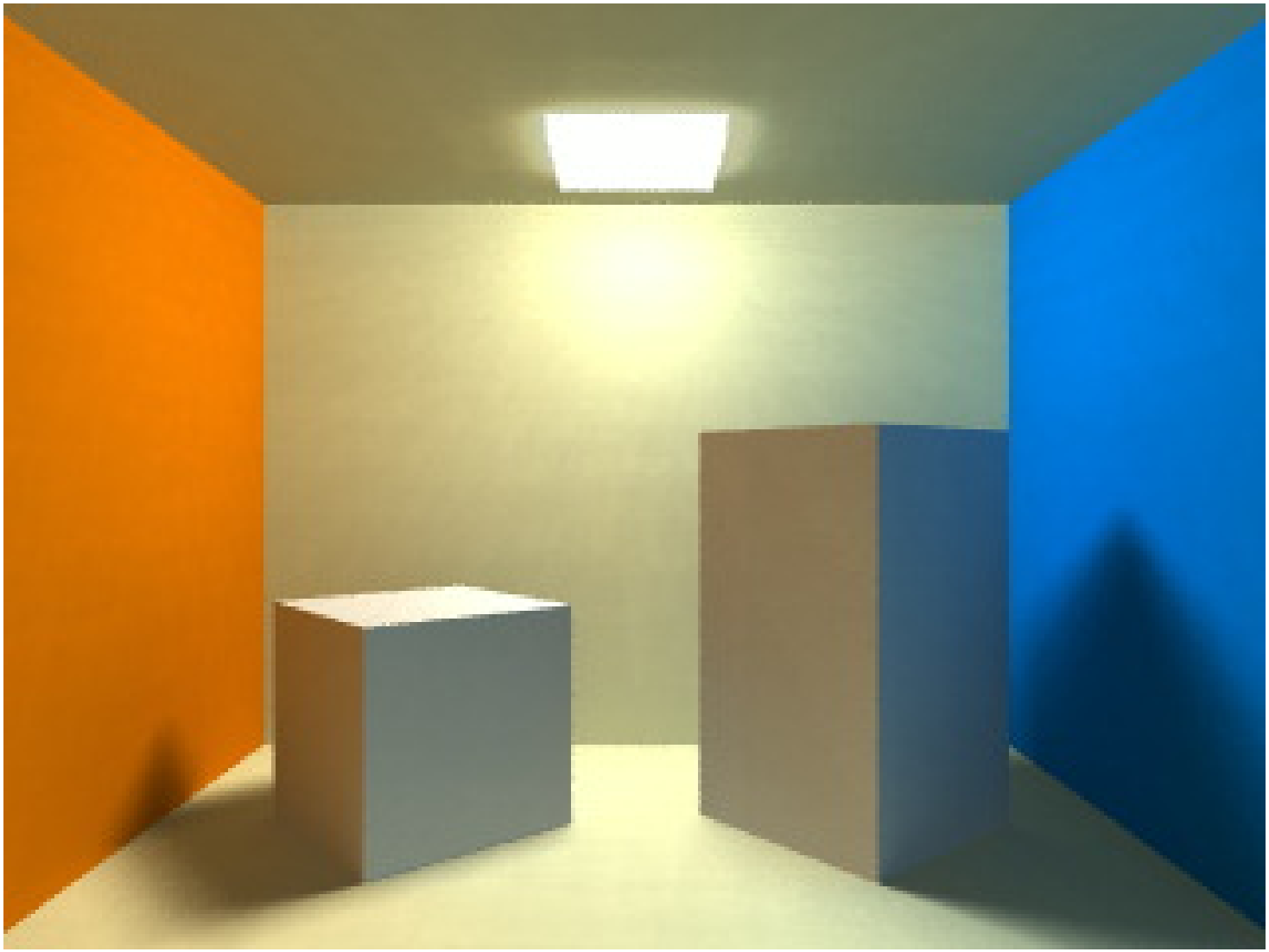

Raytracing + měkké stíny + kaustiky + globální osvětlení

# Globální osvětlení v Cornellově krabici



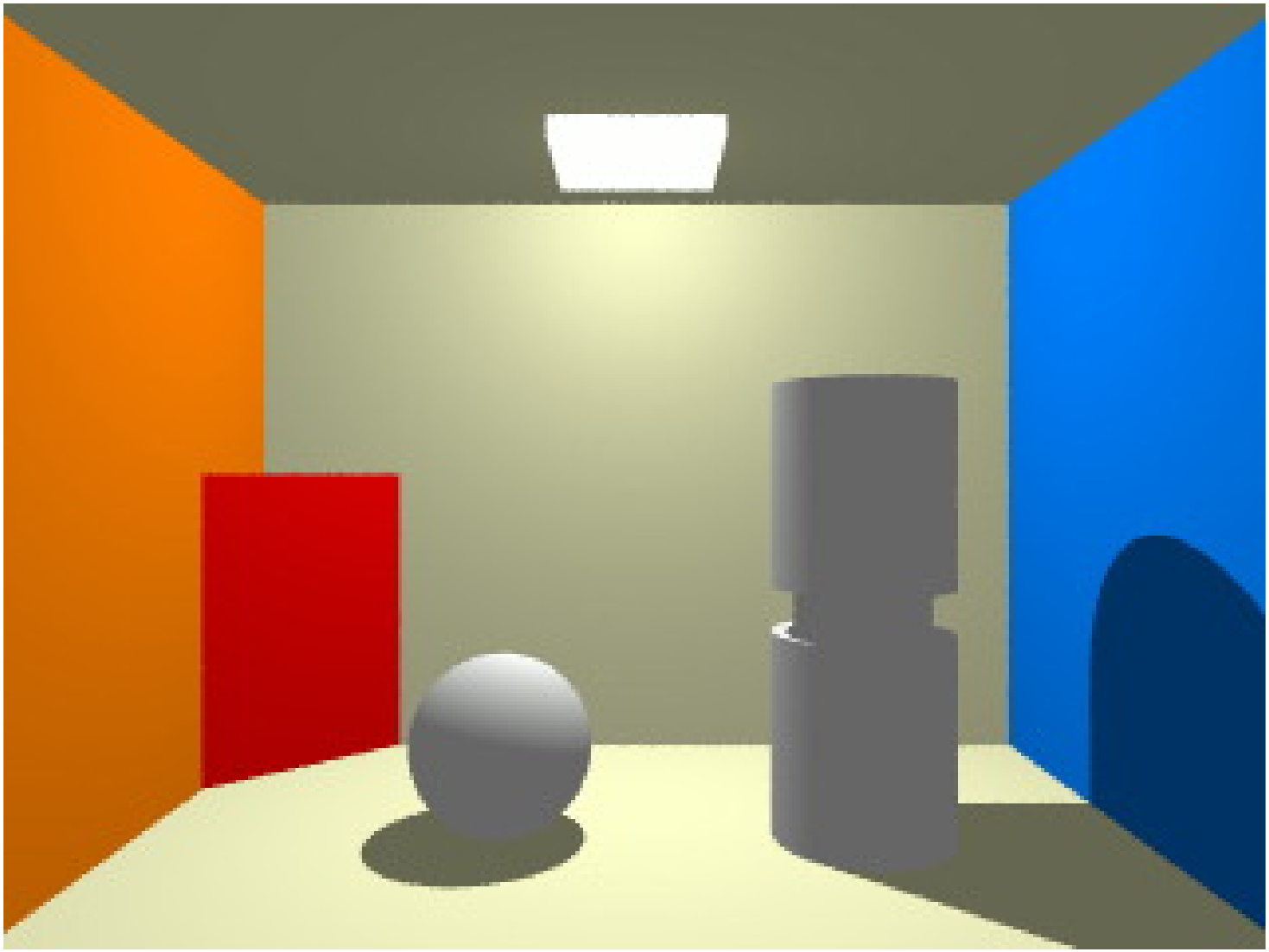
Scéna 1 bez G.O.

# Globální osvětlení v Cornellově krabici



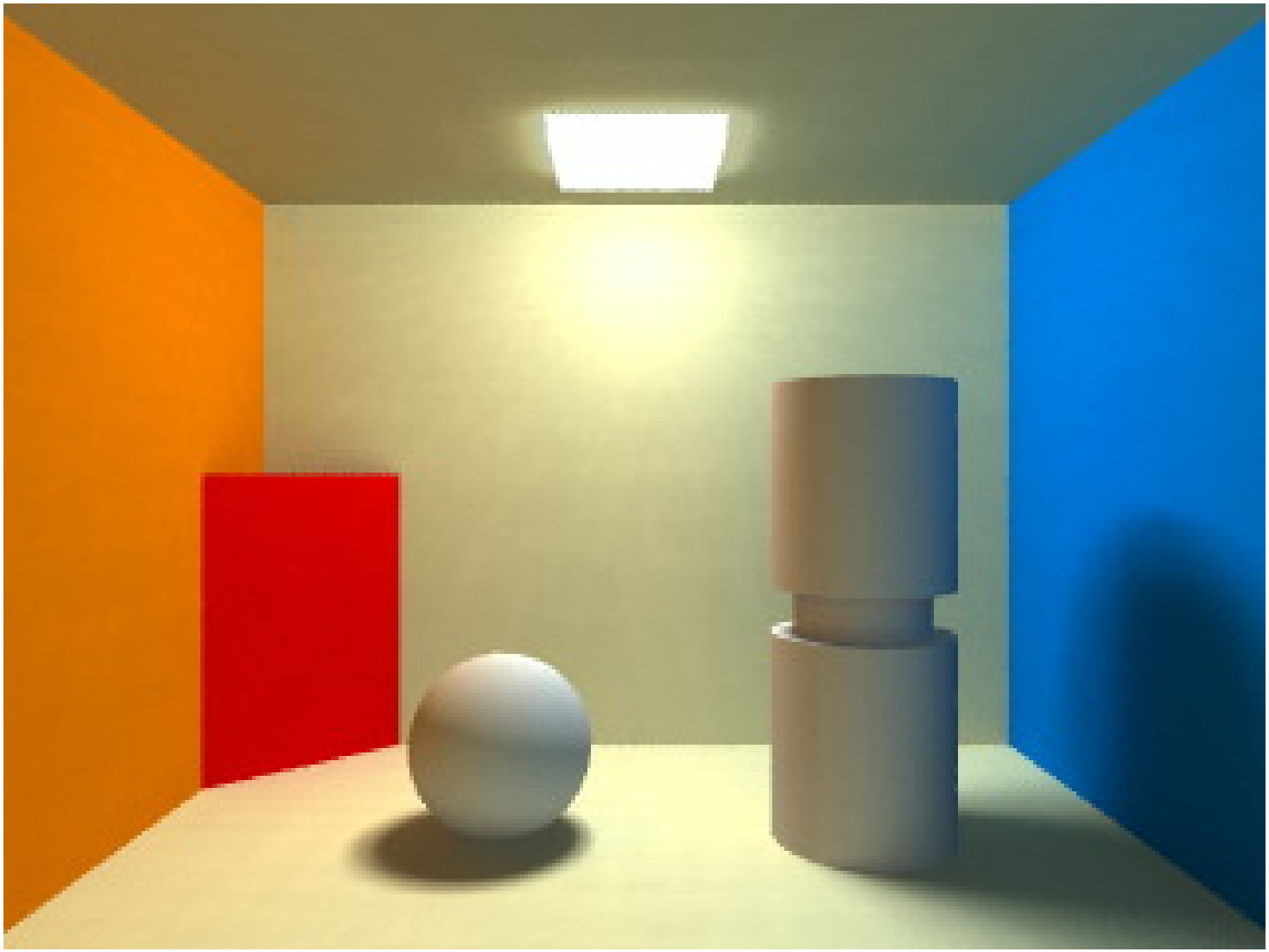
Scéna 1 s G.O.

# Globální osvětlení v Cornellově krabici



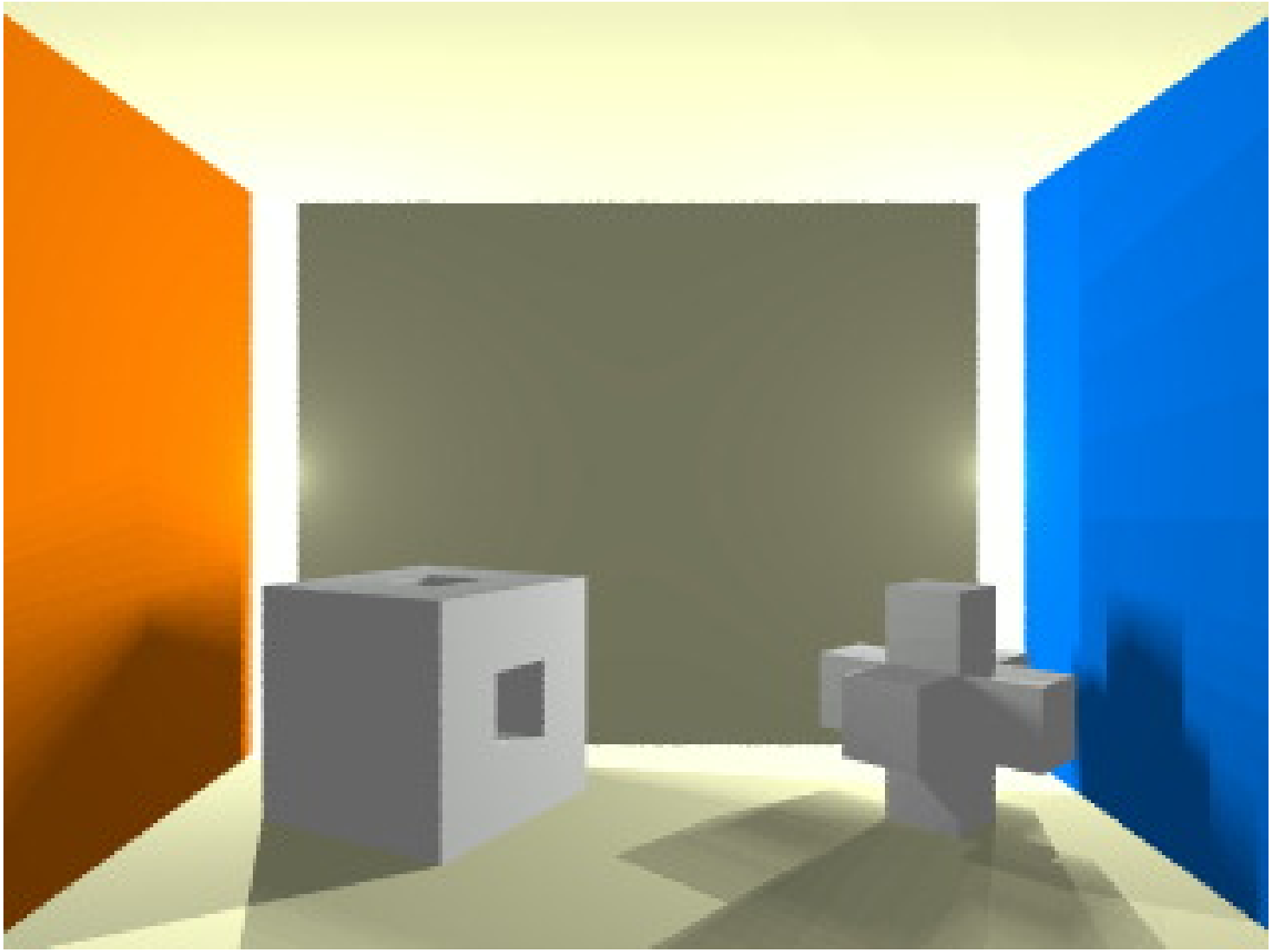
Scéna 2 bez G.O.

# Globální osvětlení v Cornellově krabici



Scéna 2 s G.O.

# Globální osvětlení v Cornellově krabici

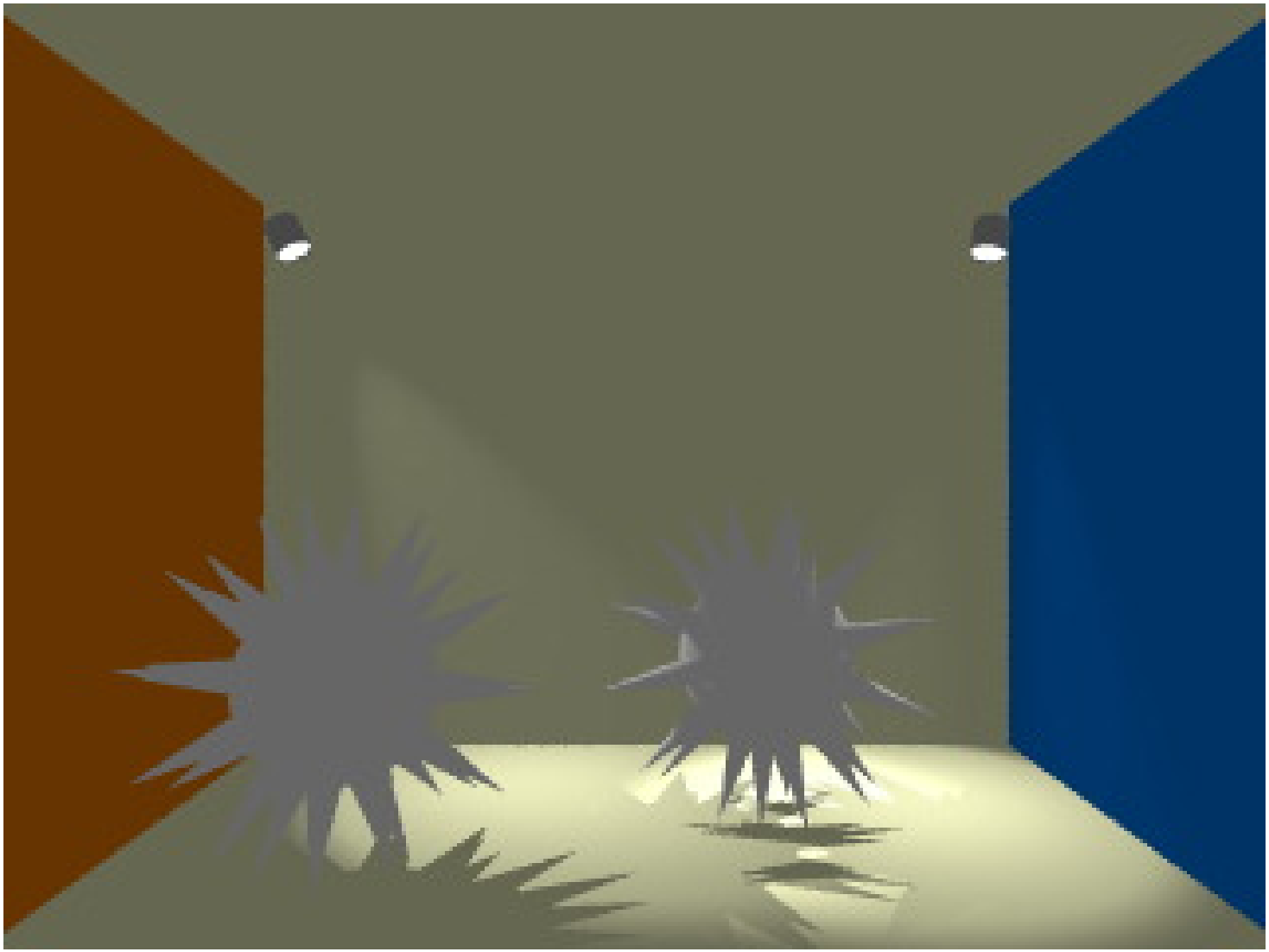


Scéna 3 bez G.O.

# Globální osvětlení v Cornellově krabici

Scéna 3 s G.O.

# Globální osvětlení v Cornellově krabici

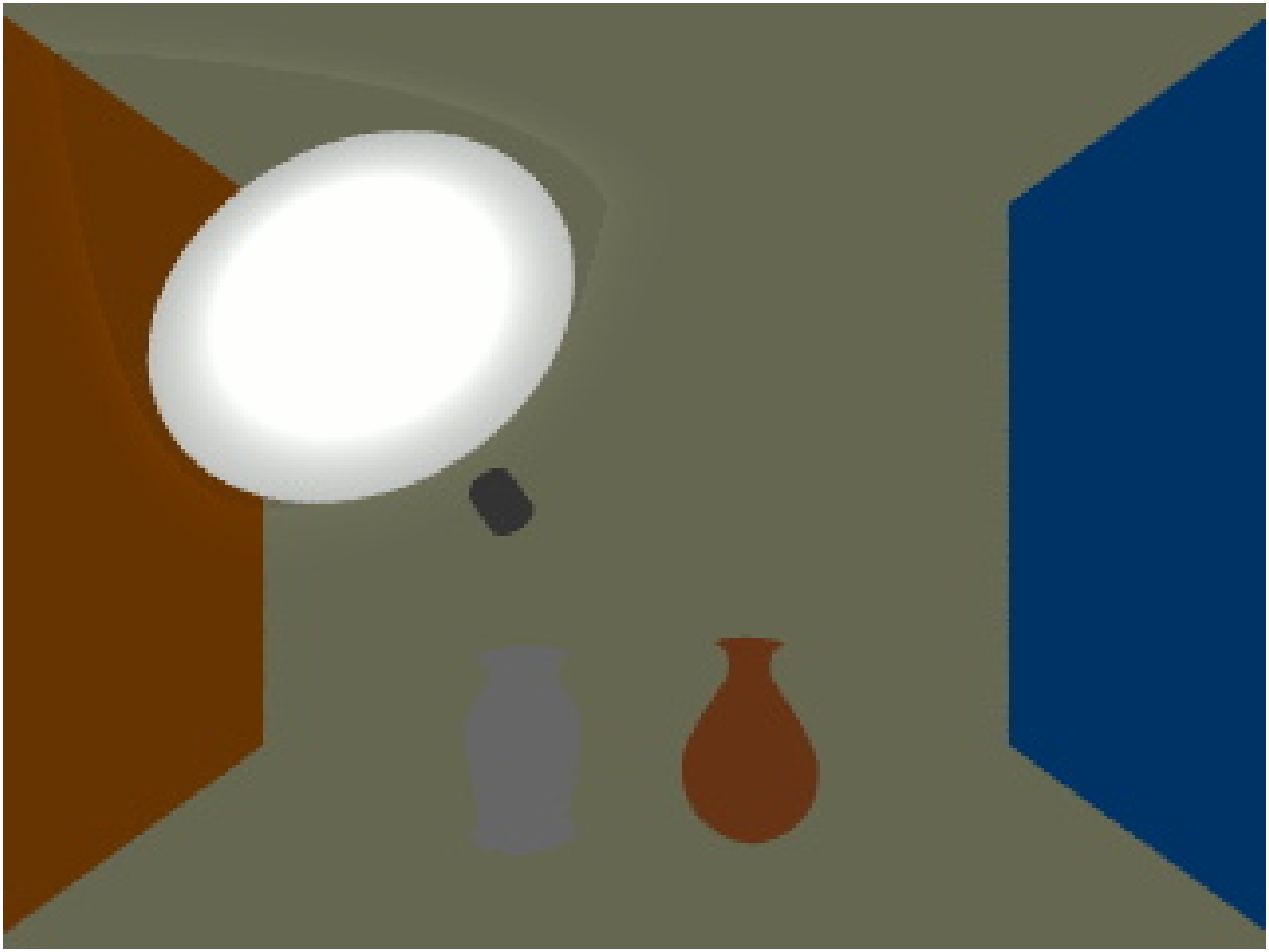


Scéna 4 bez G.O.

# Globální osvětlení v Cornellově krabici

Scéna 4 s G.O.

# Globální osvětlení - další scény

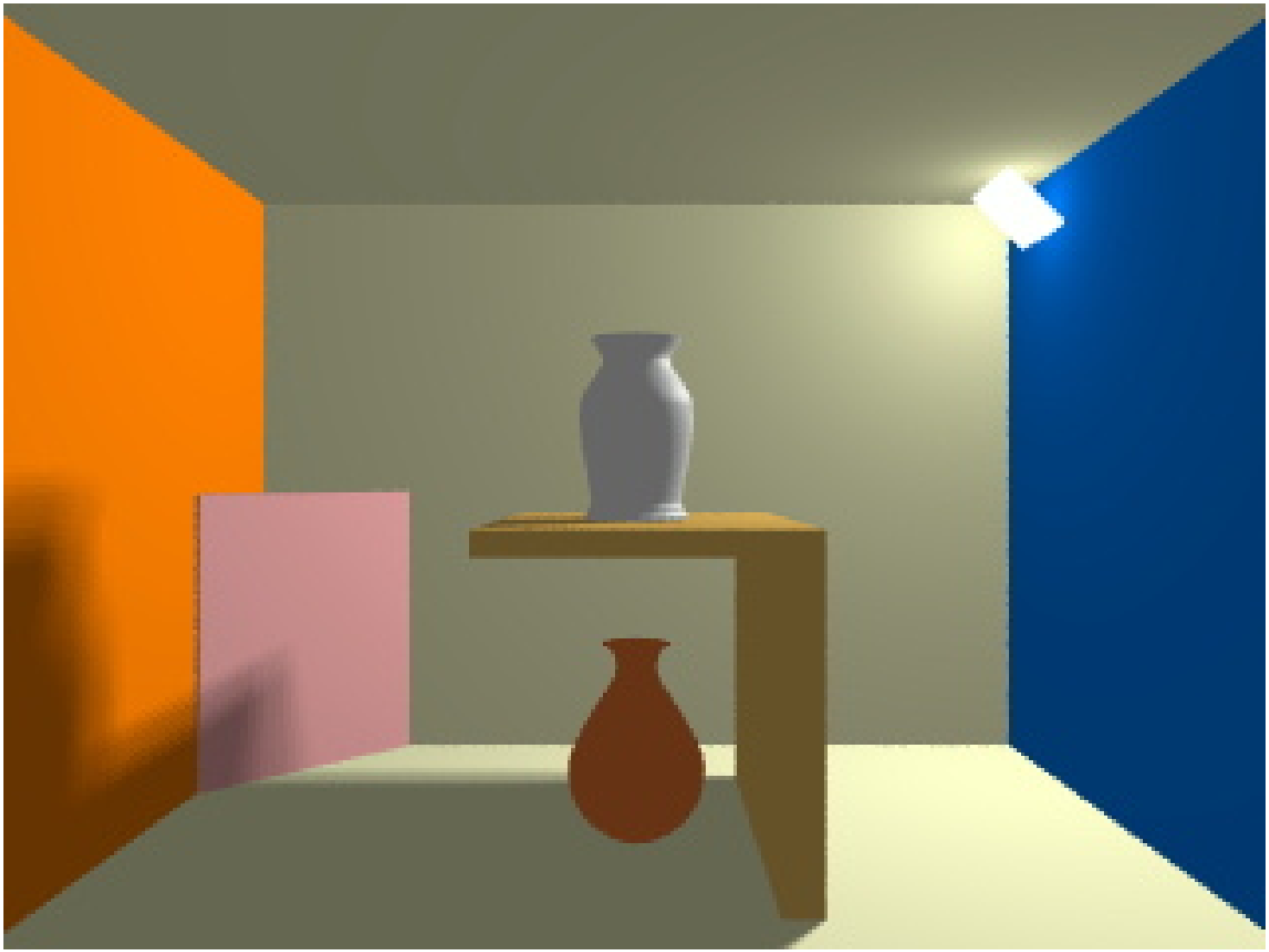


Scéna 1 bez G.O.

# Globální osvětlení - další scény

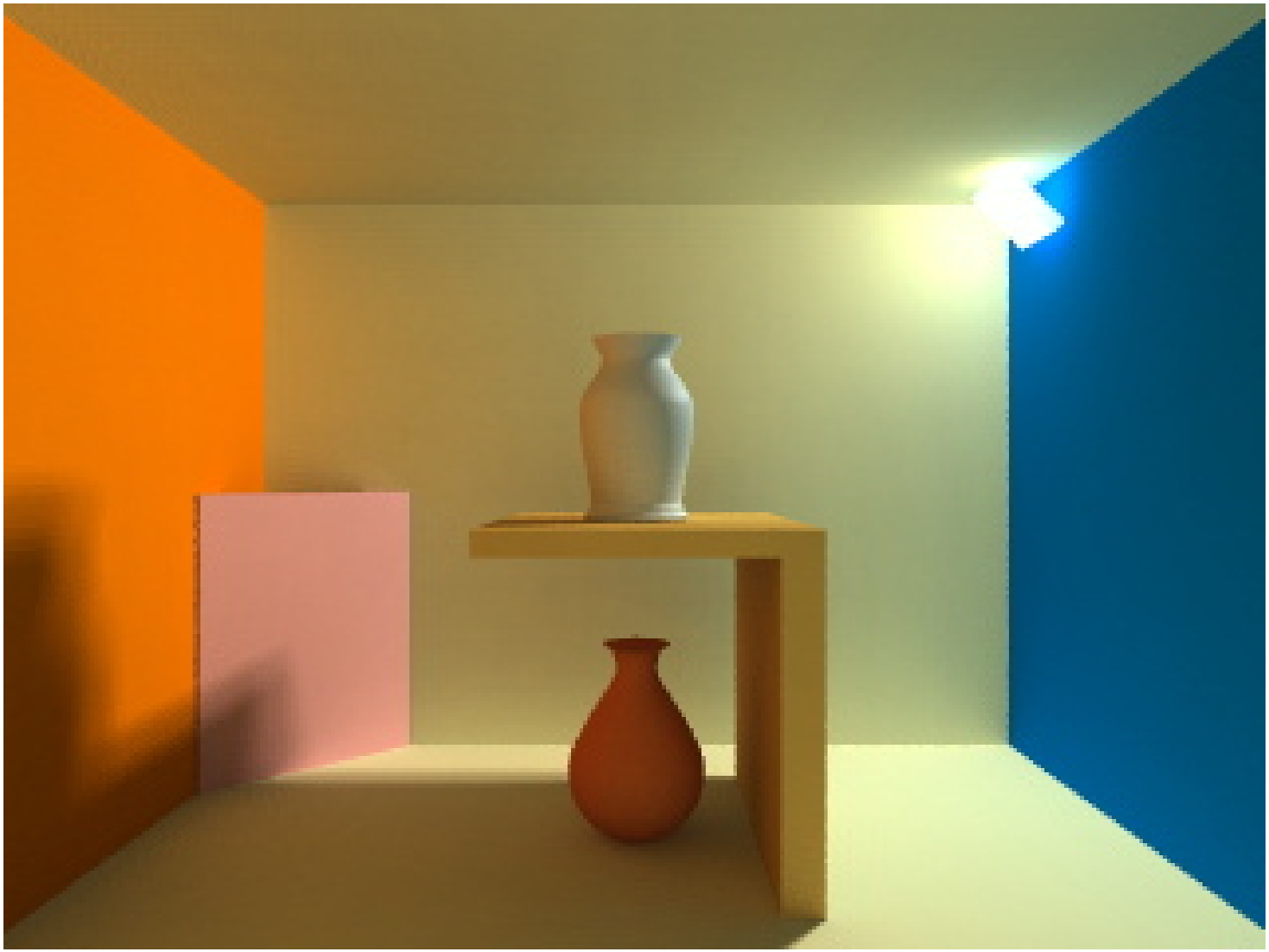
Scéna 1 s G.O.

# Globální osvětlení - další scény



Scéna 2 bez G.O.

# Globální osvětlení - další scény



Scéna 2 s G.O.

# Globální osvětlení - další scény

Scéna 3 bez G.O.

# Globální osvětlení - další scény

Scéna 3 s G.O.

# Obsah

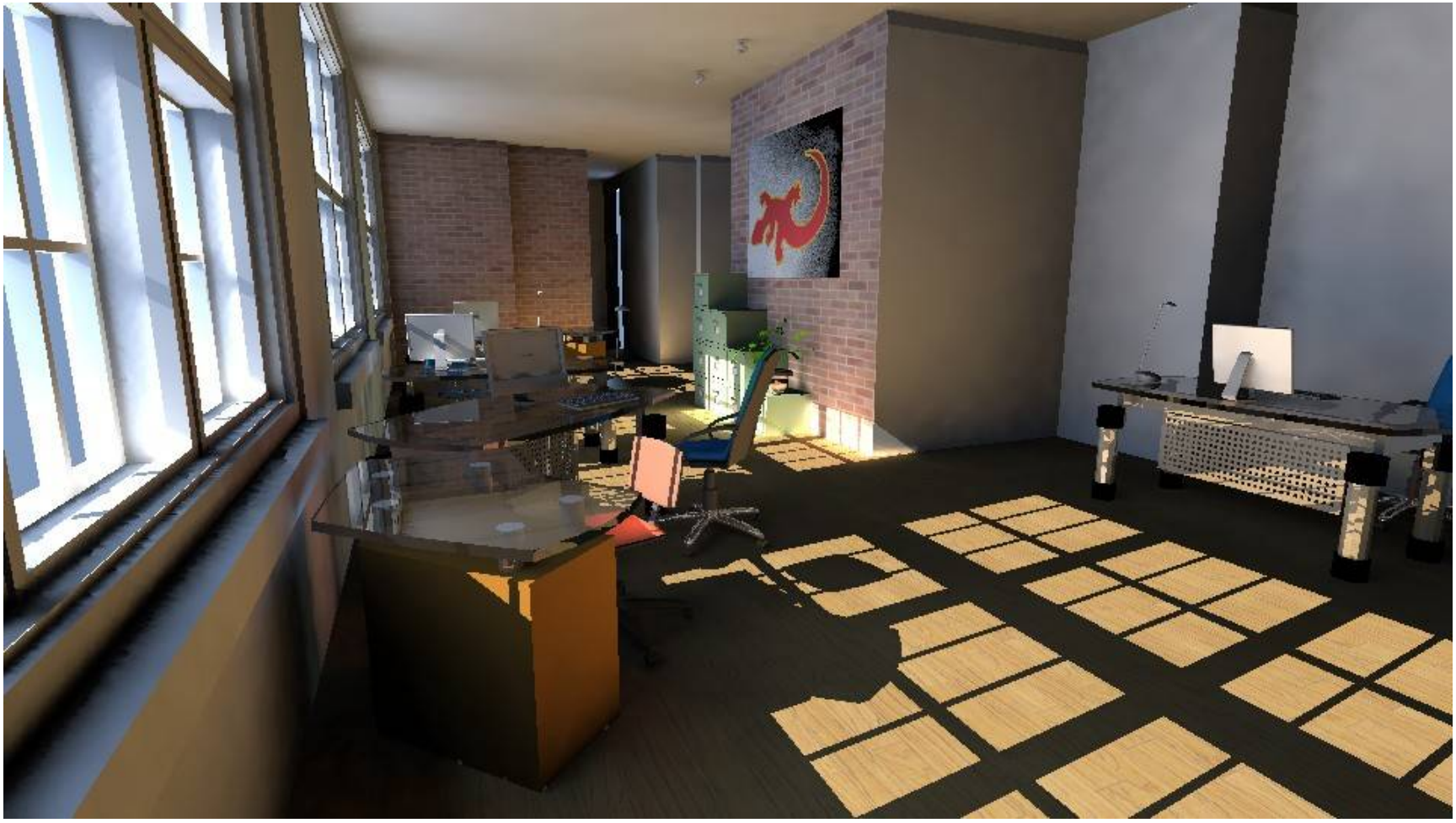
# Galerie obrázků (YafaRay, POV-Ray)

# Galerie obrázků (YafaRay, POV-Ray)

# Galerie obrázků (YafaRay, POV-Ray)

# Galerie obrázků (YafaRay, POV-Ray)

# Galerie obrázků (YafaRay, POV-Ray)



# Galerie obrázků (YafaRay, POV-Ray)

# Galerie obrázků (YafaRay, POV-Ray)

# Galerie obrázků (YafaRay, POV-Ray)

# Galerie obrázků (YafaRay, POV-Ray)

# Galerie obrázků (YafaRay, POV-Ray)

# Galerie obrázků (YafaRay, POV-Ray)

# Literatura


- Žára, Beneš, Sochor, Felkel: *Moderní počítačová grafika*. Computer Press, 2005.
- A. Glassner: *An introduction to ray tracing*. Morgan Kaufmann, 1989. ISBN 0-12-286160-4.
- T. Whitted: *An improved illumination model for shaded display*. Comm. ACM, 23(6), (1980), 343-349.
- C. Kolb, D. Mitchell, P. Hanrahan: *A realistic camera model for computer graphics*. Proc. 22nd Conference on Computer graphics and in teractive techniques, (1995), 317-324.
- M. Cohen, J. Wallace: *Radiosity and realistic image synthesis*. Morgan Kaufmann, 1993. ISBN 0-12-178270-0.