VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

FOTOREALISTICKÉ ZOBRAZOVÁNÍ

PHOTOREALISTIC RENDERING

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **MICHAL VLNAS**
AUTHOR

VEDOUCÍ PRÁCE — **Prof. Dr. Ing. PAVEL ZEMČÍK**
SUPERVISOR

BRNO 2018

Zadání bakalářské práce/19811/2017/xvlnas00

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačové grafiky a multimédií Akademický rok 2017/2018

Zadání bakalářské práce

Řešitel: **Vlnas Michal**
Obor: Informační technologie
Téma: **Fotorealistické zobrazování**
Photorealistic Rendering
Kategorie: Počítačová grafika

Pokyny:

1. Seznamte se s algoritmy realistického zobrazování 3D scén se zaměřením na metodu Path Tracing.
2. Vyberte vhodný algoritmus zobrazování a diskutujte možnosti implementace včetně využití již hotových modulů dostupných volně nebo na ÚPGM FIT.
3. Navrhněte vhodný postup implementace, případně modifikace, vybraného algoritmu a diskutujte dosažitelné vlastnosti.
4. Implementujte vybranou metodu a demonstrujte její funkčnost na vhodném příkladě.
5. Vyhodnoťte dosažené výsledky a vlastnosti algoritmu a též možnosti dalšího vývoje.

Literatura:

- Dle pokynů vedoucího

Pro udělení zápočtu za první semestr je požadováno:

- Body 1-3 zadání

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Zemčík Pavel, prof. Dr. Ing.**, UPGM FIT VUT
Datum zadání: 1. listopadu 2017
Datum odevzdání: 16. května 2018

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačové grafiky a multimédií
612 66 Brno, Božetěchova 2

L.S.

doc. Dr. Ing. Jan Černocký
vedoucí ústavu

Abstrakt

Tato práce popisuje návrh a implementaci metody řešení globálního osvětlení pomocí obousměrného sledování cesty s použitím techniky multiple importance sampling. V první části jsou uvedeny základní informace a postupy potřebné pro fotorealistické zobrazování a rovněž jsou zde představeny často používané metody v této problematice. Další část shrnuje současný stav v tomto odvětví. Poslední dvě části se věnují již konkrétně metodě sledování cesty, kde v první půlce je ukázáno matematické odvození a druhá část popisuje přímo implementaci s využitím CPU.

Abstract

This thesis describes proposal and implementation of a novel global illumination technique using bidirectional path tracing with multiple importance sampling. In the first part, basic informations and techniques are described to explain photorealistic graphic principles. Also, often used methods solving global illumination are shown. The following part summarizes current state of development. Last two parts are focused on mathematical derivation of the bidirectional path tracing and implementation itself using the CPU.

Klíčová slova

metoda sledování cesty, obousměrné sledování cesty, fotorealistické zobrazování, zobrazovací rovnice, multiple importance sampling

Keywords

pathtracing, bidirectional path tracing, photorealistic rendering, rendering equation, multiple importance sampling

Citace

VLNAS, Michal. *Fotorealistické zobrazování*. Brno, 2018. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Prof. Dr. Ing. Pavel Zemčík

Fotorealistické zobrazování

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Prof. Dr. Ing. Pavla Zemčíka. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Michal Vlnas
12. května 2018

Poděkování

Děkuji vedoucímu práce Prof. Dr. Ing. Pavlu Zemčíkovi za odborné vedení a velmi užitečné rady, které vedly k úspěšnému dokončení práce.

Obsah

Kapitola 1

Úvod

Tato práce popisuje základní informace ze speciálního odvětví počítačové grafiky, které se zaměřuje na zobrazování 3D objektu tak, aby výsledný obraz byl co nejvíce podobný reálné fotografii. Současně práce ukazuje odvození matematického aparátu, který je pro tuto práci potřebný a rovněž popisuje vytvoření počítačového programu, který tento problém řeší.

Již od počátku vývoje počítačové grafiky se nemalá skupina odborných pracovníků zaměřovala na co nejvíce realistické zobrazení počítačové grafiky. Postupem času vznikly různé způsoby a principy pro tzv. fotorealistické zobrazování počítačové scény. Ačkoliv většina metod je stále limitována výkonem procesoru, objevují se již algoritmy, které jsou kompletně použitelné na grafických kartách. I když stále většina efektivních metod není stále dostupná v akcelerované formě přes grafickou kartu, stále lze sledovat velké použití takových algoritmů.

V dnešní době lze pozorovat velký důraz na toto odvětví, ať ve filmovém, herním, či marketingovém průmyslu. Pokrok za posledních několik let je natolik velký, že dnešní velké firmy investují nemalé peníze do dalšího výzkumu a tvorby efektivního softwaru pro tyto účely. Což lze zpozorovat například ve filmovém průmyslu, kde se čím dál více používají techniky fotorealistického zobrazení ke tvorbě speciálních efektů, či přímo celých filmů. Ačkoliv fotorealistické metody stále nedosahují vykreslování v reálném čase, i přes masovou akceleraci pomocí grafické karty, jsou vhodné pro tzv. offline použití zejména v tomto odvětví průmyslu.

Cílem této práce je porozumění základních principů a metod pro zobrazení fotorealistické scény, rovněž návrh a implementace konkrétní metody.

První část práce popisuje základní principy počítačové grafiky, které slouží k fotorealistickému zobrazování. Je zde popsána reprezentace scény, projekce, radiometrické vlastnosti a další techniky. Rovněž jsou popsány konkrétní metody fotorealistického zobrazování (ray tracing, path tracing, radiozita, photon mapping). Důraz je kladen zejména na popis zobrazovací metody pomocí sledování cest a dalších náležitostí, potřebné pro pochopení této metody. Ve třetí kapitole je zhodnocen současný stav a vývoj tohoto odvětví a rovněž detailněji popsáno zadání a důvod zvolení této práce. Čtvrtá část zahrnuje formální matematické odvození a reprezentaci implementované metody. Poslední část popisuje přímo návrh a implementaci fotorealistického algoritmu. V závěru jsou představeny dosažené výsledky i výstupy z celé práce.

Kapitola 2

Fotorealistické zobrazování

Tato kapitola popisuje principy a techniky pro popis a zobrazení fotorealistické scény. Z počátku je zaměřena na základní definici scény, popis geometrie a povrchové reprezentace. Posléze jsou představeny potřebné radiometricko-optické vlastnosti a princip numerického aproximačního integrování. Dále se zaměřuje na konkrétní metody lokálního a globálního osvětlení. V neposlední řadě ukazuje již existující software pro řešení fotorealistických scén a této problematiky obecně. Tato kapitola si nedává za cíl být encyklopedickým přehledem, ale pouze ukázat základní poznatky potřebné v daném rozsahu práce.

2.1 Úvod a popis scény

Fotorealistické zobrazování je odvětvím počítačové grafiky, zaměřující se na zobrazování (angl. rendering). Jedná se o proces generování dvourozměrného obrázku ze třírozměrné scény. Toto odvětví se zaměřuje, oproti jiným metodám zobrazování, zejména na realističnost tak, aby výsledný obraz byl co nejvíce věrohodný skutečnosti. Zejména se zaměřuje na fyzikální simulace toku a přenosu světla ve scéně, realistické stíny, odlesky a jiné. Kvůli této vlastnosti většinou tyto metody nejsou schopny zobrazit scénu v reálném čase.

Dnešní metody zobrazování, které lze nalézt například v počítačových hrách, různých grafických aplikacích, apod., používají zjednodušené zobrazování právě s cílem dosáhnout zobrazení v reálném čase. Tím ovšem výsledný obraz ztrácí možné výhody fotorealističnosti, jako například přesné fyzikální stíny, odlesky, lomy světla a podobně.

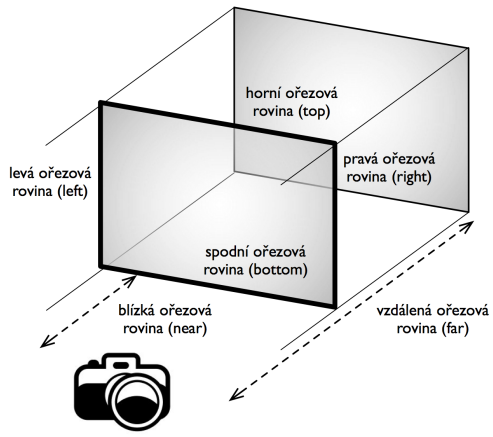
Reprezentace scény

Nejobvyklejší reprezentací tvaru tělesa je tzv. hraniční reprezentace [20], tzn. těleso je popsáno svými hranicemi - stěny, hrany, vrcholy.

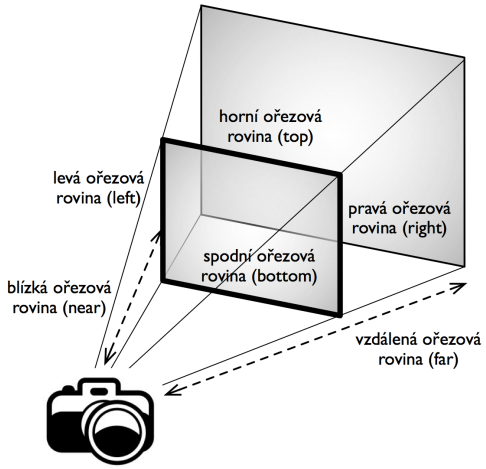
Konkrétní hranice jsou nejčastěji tvořeny trojúhelníkovou sítí [20]. Každý bod trojúhelníku poté nese informaci o své pozici a k tomu odpovídající normálu. Na základě tohoto lze v každém bodě vypočítat lokální osvětlovací model, tedy výslednou barvu, vztažený k materiálovým vlastnostem celého objektu. Detailnější popis lokálního osvětlovací modelu lze nalézt v kapitole 2.5.

2.2 Projekce

Projekce (někdy taktéž jako promítání) [20] slouží k transformaci trojrozměrné scény do dvourozměrného obrazu. Toho lze docílit vysíláním promítacích paprsků ze scény směrem k rovině obrazu. Lze rozlišit dva základní typy projekcí - *ortografická* a *perspektivní*.



Obrázek 2.1: Ortografická projekce [1]



Obrázek 2.2: Perspektivní projekce [2]

Ortografická

Ortografická (taktéž pravoúhlá) [20] projekce představuje pravoúhlý hranol ve scéně, kdy díky rovnoběžnosti všech promítacích paprsků zachovává všechny původní rozměry. Tento typ projekce je vhodný pro konstrukční programy typu CAD a podobně. Pro fotorealistické zobrazování vhodný není.

Perspektivní

Scéna je reprezentována jako komolý jehlan. Všechny promítací paprsky směřují do bodu projekce [20] (kamera) a procházejí přes rovinu obrazu (angl. image plane nebo near plane). Výsledný obraz zobrazuje objekty mezi blízkou a vzdálenou ořezovou rovinou. Čím blíže je objekt k blízké ořezové rovině, tím větší se zobrazí na výsledném obrazu.

2.3 Radiometrie

Jelikož fotorealistické zobrazení řeší fyzikální simulaci světla ve scéně, tedy řešení problému tzv. *globálního osvětlení*, je nutno definovat základní vlastnosti popisující chování světla. Z hlediska zjednodušení je v následujících vztazích uvažováno, že světlo se pohybuje po přímé linii, a to s nekonečnou rychlostí.

[1] Zdroj: https://is.mendelu.cz/eknihovna/opory/download.pl?objekt=23662
[2] Zdroj: https://is.mendelu.cz/eknihovna/opory/download.pl?objekt=23660

Zářivý tok

Zářivý tok, obvykle značený jako Φ_e, je tok energie procházející určitou plochu za jednotku času. Je měřen jednotkou *Watt*. Je definován vztahem [4]:

$$\Phi_e = \frac{\mathrm{d}E}{\mathrm{d}t} \tag{2.1}$$

kde E je energie vyzářená zdrojem za čas t.

Ozářenost a radiozita

Tyto dvě vlastnosti definují množství dopadajícího, resp. emitujícího a odraženého, zářivého toku $d\Phi_e$ na povrch tělesa dA. Dopadající zářivý tok na těleso se nazývá ozářenost a bývá označován jako E_e. Odcházející, tedy emitovaný a odražený, zářivý tok z povrchu je nazýván radiozita, značeno M_e. Jednotkou obou vlastností je $W \cdot m^{-2}$. Ozářenost a radiozita jsou definovány [4]:

$$E_e = \frac{\mathrm{d}\Phi_e}{\mathrm{d}A}, M_e = \frac{\mathrm{d}\Phi_e}{\mathrm{d}A} \tag{2.2}$$

Zářivost

Zářivost je měrná veličina zářivého toku $d\Phi_e$ v prostorovém úhlu $d\omega$. Její jednotka je $W{\cdot}sr^{-1}$[4].

$$I_e = \frac{\mathrm{d}\Phi_e}{\mathrm{d}\omega} \tag{2.3}$$

Zář

Zář je primární veličina k popisu světla v algoritmech globálního osvětlení. Je definována jako měrná veličina zářivosti plošného zdroje. Značí se L_e a její jednotka je $W{\cdot}m^{-2}{\cdot}sr^{-1}$. Důležitou vlastností záře je, že se nemění v závislosti na vzdálenosti. Lze ji vyjádřit vztahem [4]:

$$L_e(\mathbf{x}, \omega) = \frac{\mathrm{d}^2\Phi_e}{\mathrm{d}\omega \cdot dA \cdot cos(\theta)} \tag{2.4}$$

kde $L_e(\mathbf{x}, \omega)$ je zář v bodě $\mathbf{x}$ ve směru ω, Φ_e je zářivý tok zdroje, ω označuje prostorový úhel, A značí plochu povrchu a θ je úhel mezi normálou v bodě $\mathbf{x}$ a měřeným směrem.

Potenciál

V počítačové grafice je potenciálem myšleno množství zářivého toku tekoucího z bodu i do bodu j. Na rozdíl od záře nemá konkrétní jednotku, pohybuje se pouze v rozmezí 0 až 1 a určuje měřítko přenášeného světla. Základní formou je emitovaný potenciál W_e určující množství dopadajícího světla ze zdroje na pixel roviny obrazu [4]:

$$W_e(\mathbf{x}, \omega) = \begin{cases} 1 & \text{pro } (\mathbf{x}, \omega) \in S \\ 0 & \text{pro } (\mathbf{x}, \omega) \notin S \end{cases} \tag{2.5}$$

kde $(\mathbf{x}, \omega)$ definuje paprsek a S je množina všech paprsků dopadajících na pixel.

Další reprezentací potenciálu je tzv. *potenciálová rovnice* [4], která říká celkové množství emitovaného a odraženého světla, které dopadá na pixel v rovině obrazu.

$$W(\mathbf{x}, \omega_o) = W_e(\mathbf{x}, \omega_o) + \int_\Omega W(\mathbf{x}', \omega_i) \cdot f_r(\mathbf{x}, \omega_i \to \omega_o) \cdot \cos(\theta) \mathrm{d}\omega \tag{2.6}$$

kde $f_r(\mathbf{x}, \omega_i \to \omega_o)$ je tzv. BRDF (viz kapitola 2.5) a θ je úhel mezi směrem dopadu a normálou v bodě dopadu. Integrál je integrován přes všechny směry Ω.

2.4 Monte Carlo integrování

Pomocí metody Monte Carlo [13, 20] lze numerickým odhadem určit přibližnou hodnotu integrálu za použití náhodného vzorkování. Jiné algoritmy volí integrand podle určitých vzorů, zatímco Monte Carlo volí body zcela náhodně, s ohledem na zvolenou hustotu pravděpodobnosti.

Jednoduchou funkci $f : \Omega \to \mathbb{R}$ lze pomocí využití tzv. *primárního estimátoru* aproximovat jako:

$$I = \int_\Omega f(x) \mathrm{d}x = \frac{f(x)}{p(x)} \tag{2.7}$$

kde x je libovolná náhodná veličina s hustotou pravděpodobnosti $p(x)$.

Je ovšem patrné, že využíti pouze jednoho náhodného vzorku nezaručuje dostatečnou přesnost výsledku. Pro využití N nezávislých náhodných veličin se zavádí tzv. *sekundární estimátor*, pomocí něhož aproximace integrálu vypadá následovně:

$$I = \int_\Omega f(x) \mathrm{d}x = \frac{1}{N} \sum_{i=1}^{N} \frac{f(x_i)}{p(x_i)} \tag{2.8}$$

kde x_i je opět náhodná veličina s hustotou pravděpodobnosti $p(x_i)$.

Využitím sekundárního estimátoru, s dostatečným počtem náhodných vzorků, lze již dosáhnout důvěryhodných výsledků.

Pro efektivnější aproximaci integrálu je vhodné použít non-uniformní rozdělení pravděpodobnosti a zvolit takovou hustotu pravděpodobnosti $p(x)$, která nejvíce kopíruje možný rozptyl hodnot integrandu a dosáhnout tím efektu tzv. *vzorkování podle důležitostí*, angl. *importance sampling*.

2.5 Lokální osvětlovací modely

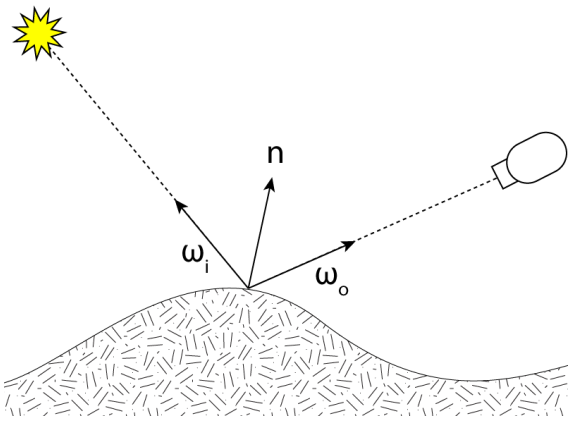
V této kapitole jsou popsány základní metody pro výpočet tzv. lokálního osvětlení, tedy výpočtu barvy v konkrétním bodě na povrchu tělesa. V první části je popsaná obousměrná distribuční funkce odrazu světla, a posléze jednotlivé empirické modely pro řešení lokálního osvětlení.

Obousměrná distribuční funkce odrazu světla

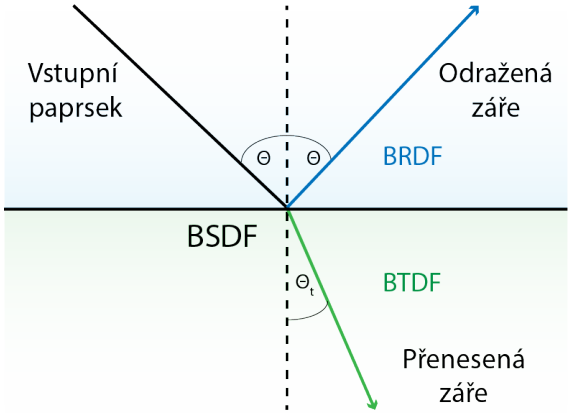
Obousměrná distribuční funkce odrazu světla pochází z anglického názvu *bidirectional reflectance distribution function*, zkráceně BRDF [14]. Tato funkce se používá pro vyjádření povrchových vlastností objektu, zejména odrazivých schopností. Poprvé byla odvozena a popsána Fredem Nicodemusem okolo roku 1965. Je vyjádřena vztahem:

$$f_r(\mathbf{x}, \omega_i \to \omega_o) = \frac{\mathrm{d}L_r(\omega_o)}{\mathrm{d}E(\omega_i)} = \frac{\mathrm{d}L_r(\omega_o)}{L_i(\omega_i) \cdot \cos(\theta_i)\mathrm{d}\omega_i} \tag{2.9}$$

kde se jedná o poměr odražené diferenciální záře L_r ve směru ω_o vůči ozáření povrchu $dE(\omega_i)$ ze směru ω_i, kde $L_i(\omega_i)$ reprezentuje zář dopadající ze směru ω_i na bod $\mathbf{x}$ a θ_i je úhel dopadající záře vůči normále.



Obrázek 2.3: Ilustrace znázorňující BRDF – ω_i je příchozí a ω_o je odchozí směr záře[3]



Obrázek 2.4: Ukázka nadstavby BSDF nad BRDF

BRDF má několik důležitých vlastností, které je nutno definovat pro pochopení dalších principů:

- **Linearita** – lineárnost vzhledem k záři lze odvodit přímo ze vzorce. V praxi to znamená, že příspěvky světla vyzářené z různých směrů lze mezi sebou sčítat.
- **Pozitivita** – hodnota BRDF je vždy nezáporná, platí tedy $f_r(\mathbf{x}, \omega_i \to \omega_0) \geq 0$
- **Helmholzova reciprocita** – toto je základní vlastnost každé fyzikálně korektní BRDF, vyplývá přímo ze zákona odrazu a znamená, že hodnota BRDF je stejná při záměně úhlu dopadu a odrazu, tedy platí $f_r(\mathbf{x}, \omega_i \to \omega_0) = f_r(\mathbf{x}, \omega_0 \to \omega_i)$
- **Zákon zachování energie** – poměr odraženého k příchozímu zářivému toku musí být menší nebo rovno 1, tzn. nelze reflektovat více světla než je emitováno ze zdroje.

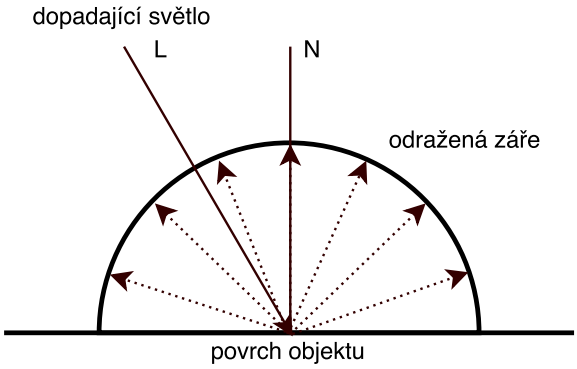
Pro korektnější simulaci fyzikálního chování povrchů lze zavést kromě BRDF ještě tzv. *obousměrnou distribuční funkci propustnosti*, zkráceně BTDF, která se chová podobně jako BRDF, ovšem na druhé straně povrchu. Současné použití BRDF a BTDF definuje tzv. BSDF (bidirectional scattering distribution function), kterou lze simulovat jak světlo odražené tak i přenášené, viz obrázek 2.4. Použití BSDF je vhodné pro popis složitějších materiálů, které umožňují propustnost světla, či ho přímo pohlcují. Takové materiály jsou vhodně popsány pomocí *Fresnelových rovnic* [4].

Lambertův osvětlovací model

Ideální reprezentací dokonale matného, tzv. *difúzního*, povrchu je model představený H. Lambertem [14]. V přírodě se tento model nevyskytuje z důvodu termodynamických vlastností,

[3] Zdroj: https://upload.wikimedia.org/wikipedia/commons/c/c3/BRDF_Diagram.png

avšak pro simulační účely je vhodný. Hlavní předností je, že při použití BRDF s Lambertovým modelem je záře stejná ve všech směrech pohledu z daného bodu a je úměrná k ozářenosti. Obecně platí, že čím blíže světlo dopadá normále povrchu, tím větší je množství odraženého světla. Tento jev je popsán tzv. *Lambertovým zákonem*. Ilustrace je znázorněna na Obrázku 2.5.



Obrázek 2.5: Ilustrace odrazivosti Lambertovského povrchu

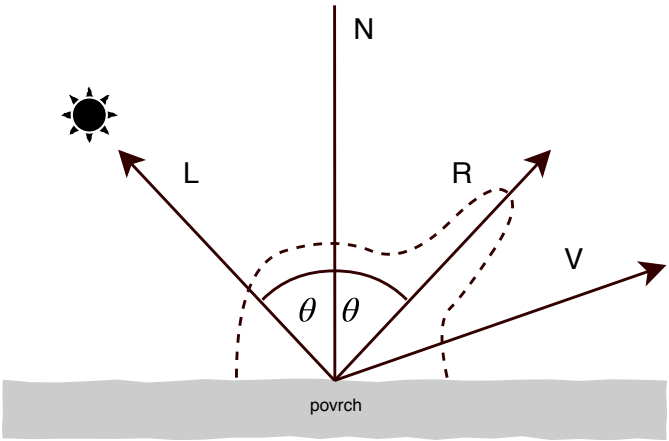
Výpočet výsledného odraženého světla lze popsat pomocí vztahu:

$$I = I_L C(\vec{L} \cdot \vec{N}) \tag{2.10}$$

kde I_L je barva dopadajícího světla, C je barva, $\vec{L}$ je vektor dopadajícího světla a $\vec{N}$ je normála.

Phongův osvětlovací model

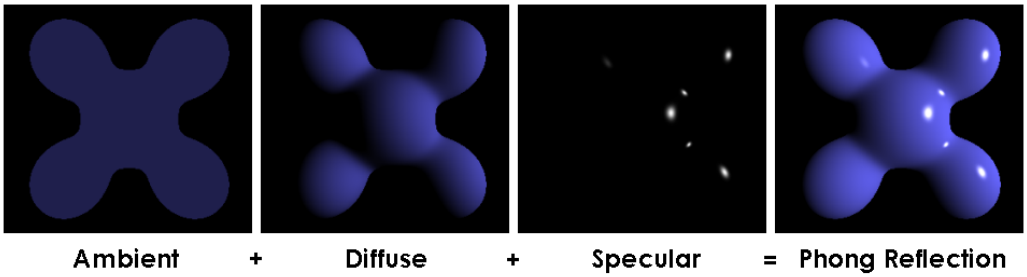
Phongův osvětlovací model [11] byl publikován roku 1973 vědcem Bui Thong Phongem v rámci jeho dizertační práce. Phong zde stanovuje, že pro tvorbu realistického vzhledu se světlo v daném bodě skládá ze tří složek – difúzní, zrcadlové (lesklé, spekulární) a okolní (ambientní).



Obrázek 2.6: Přehled použitých vektorů při výpočtu Phongova osvětlovacího modelu

Výpočet výsledného odraženého světla směrem k pozorovateli je dán součtem ambientní, difúzní a zrcadlové složky. Jednotlivé složky jsou ilustrovány na Obrázku 2.7.

$$I = I_A + I_D + I_S \tag{2.11}$$



Obrázek 2.7: Ilustrace složek a výsledného světla podle Phongova modelu [4]

Ambientní složka

Okolní složka světla je intenzita té části světla, které dopadá rovnoměrně ze všech směrů. Napodobuje sekundární odražené světlo vznikající odrazy od různých těles ve scéně. Jinými slovy se jedná o tzv. konstantní všesměrové světlo. Odraz okolního světla je dán vztahem:

$$I_A = I_a k_a \tag{2.12}$$

kde I_a je intenzita okolního světla ve scéně a $k_a \in (0, 1)$ je odrazivý koeficient materiálu objektu, určující schopnost tělesa odrážet okolní světlo.

Difúzní složka

Difúzní složka má totožné vlastnosti, které byly definovány již v kapitole 2.5.

$$I_D = I_L C(\vec{L} \cdot \vec{N}) \tag{2.13}$$

Zrcadlová složka

Zrcadlová složka, neboli lesklé světlo, udává intenzitu té části světla, která se od tělesa odráží ve směru vypočteném podle zákona odrazu světla. Výpočet je definován vztahem:

$$I_S = I_s k_s (\vec{V} \cdot \vec{R})^n \tag{2.14}$$

kde I_s určuje intenzitu odlesků, $k_s \in (0, 1)$ je odrazivý koeficient, který určuje míru zastoupení lesklého světla v celkovém odraženém světle, $\vec{V}$ je vektor směru k pozorovateli a $\vec{R}$ je odražené světlo od povrchu pomocí zákona odrazu. Lze jej určit vztahem:

$$R = 2(\vec{N} \cdot \vec{L})\vec{N} - \vec{L} \tag{2.15}$$

Phongův exponent $n \in (0, \infty)$ určuje míru ostrosti odrazu. Čím vyšší n, tím intenzivnější, ale menší zrcadlové odlesky.

[4] Zdroj: https://upload.wikimedia.org/wikipedia/commons/6/6b/Phong_components_version_4.png

Blinn-Phongův osvětlovací model

Blinn-Phongův model je modifikací již zmíněného Phongova modelu a byl publikován J. Blinnem [3]. Změna oproti Phongovu modelu je pouze ve výpočtu zrcadlové složky, kde namísto skalárního součinu $\vec{V} \cdot \vec{R}$ mezi vektorem pohledu od pozorovatele a vektoru odraženého paprsku, je použit skalární součin mezi normálou a půl vektorem mezi pozorovatelem a světelným zdrojem $\vec{N} \cdot \vec{H}$. Půl vektor je vyjádřen vztahem:

$$H = \frac{\vec{L} + \vec{V}}{||\vec{L} + \vec{V}||} \tag{2.16}$$

Obecně je Blinn-Phongův model rychlejší a efektivnější než Phongův a proto bývá často používán v grafických API jako např. *OpenGL* či *DirectX*.

2.6 Zobrazování

Scény v počítačové grafice lze zobrazovat mnoha způsoby. V dnešní době je nejrozšířenější zobrazování metodou *rasterizace trojúhelníků* [20]. Jelikož tuto metodu lze snadno akcelerovat pomocí hardwaru, je velmi hojně používána. Hlavní výhodou této metody je její rychlost. Ovšem na druhou stranu ale postrádá prvky fotorealistického zobrazování (i když některé modifikace metody zvládají přesvědčivé napodobení, ale stále nejsou fyzikálně korektní), z toho důvodu není vhodné se metodou dále zabývat.

Metody, které řeší fotorealistické vlastnosti, jsou algoritmy simulující fyzikální šíření světla scénou. Nazývají se metody pro řešení globálního osvětlení.

Tato kapitola se zaměřuje v první části na matematický popis řešení světla ve scéně, posléze na konkrétní metody implementující tento problém.

Zobrazovací rovnice

Zobrazovací rovnici (angl. rendering equation) publikoval v roce 1986 James T. Kajiya [8]. Rovnice popisuje přenos světla ve scéně. V původním znění je definována:

$$L(\mathbf{x}, \omega_0) = L_e(\mathbf{x}, \omega_0) + \int_\Omega L(\mathbf{x}', -\omega_i) f_r(\mathbf{x}, \omega_i, \omega_0) \cdot \cos(\theta_i) \mathrm{d}\omega_i \tag{2.17}$$

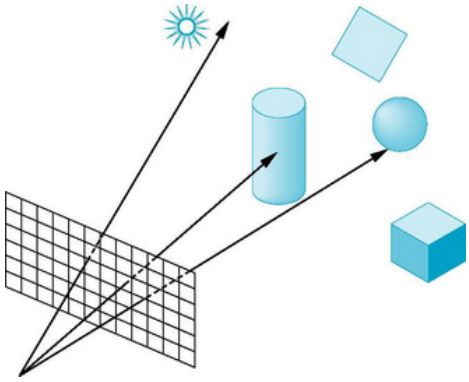
kde $L(\mathbf{x}, \omega_0)$ značí celkovou zář z bodu $\mathbf{x}$ ve směru ω_0, $L_e(\mathbf{x}, \omega_0)$ je emitovaná zář rovněž z bodu $\mathbf{x}$ ve směru ω_0, $L(\mathbf{x}', -\omega_i)$ označuje rekurzivní výpočet záře ze směru ω_i. Další složkou rovnice je koeficient BRDF $f_r(\mathbf{x}, \omega_i, \omega_0)$ násobený funkcí kosinus úhlu θ_i, který značí úhel mezi směrem ω_i a normálou povrchu. Integrál je řešený přes všechny směry Ω na hemisféře nad bodem $\mathbf{x}$.

Zobrazovací rovnici v úhlové formě není možné jednoduše vyřešit, proto se v praxi provádí množství úprav pro zjednodušení a aproximaci výsledku. Jedná se zejména o převod na *plošnou formu* a aproximace pomocí *metody Monte Carlo*. Tyto případy jsou podrobně rozebrány v kapitole 2.4 a 4.1.

Ray casting

Nejjednodušší zobrazovací metodou je tzv. vrhání paprsků, neboli ray casting [12]. Poprvé byla představena roku 1968 v publikaci od A. Appela [1].

Na rozdíl od reálného světa, kdy světlo postupuje od světelného zdroje, ray casting vysílá paprsky směrem od pozorovatele přes průmětnu do scény, viz obrázek 2.8. Výsledná



Obrázek 2.8: Ray casting – vystřelování paprsků[5]

barva bodu na průmětně odpovídá hodnotě povrchu prvního objektu, na který vyslaný paprsek narazil. Hodnota barvy průsečíků paprsku s objektem je určena na základě lokálního osvětlovacího modelu.

V některých literárních zdrojích lze nalézt označení metody jako *sledování paprsku* prvního řádu. Důvodem je, že ray casting vysílá pouze tzv. primární paprsky od pozorovatele, které končí při nárazu do nejbližšího objektu, ale neřeší dále možné odražené paprsky, tzv. sekundární paprsky. Z toho důvodu nelze touto metodou simulovat jakékoliv fotorealistické efekty.

Sledování paprsku

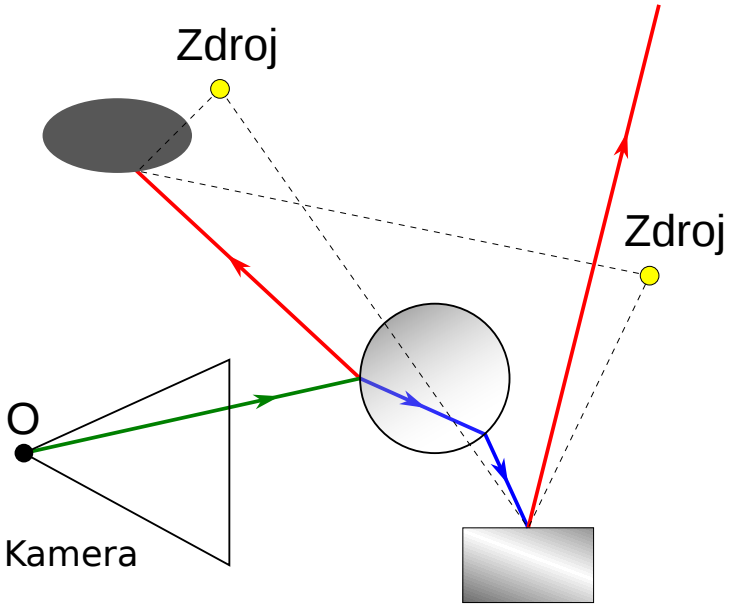
Sledování paprsku, angl. ray tracing, je první z metod, která řeší kompletní globální osvětlení. Poprvé byla tato metoda publikována T. Whittedem [19] roku 1979. Oproti metodě ray casting, ray tracing zavádí rekurzivní zpracování odražených paprsků.

Podobně jako u ray castingu, ray tracing vysílá paprsky od oka pozorovatele směrem do scény. Vyslané paprsky kolují scénou dokud nenarazí na objekt, oproti ray castingu zde ale nekončí. Jsou zavedeny tzv. sekundární paprsky, které lze rozdělit na tři druhy:

- **Stínové** – jsou vysílány z každého průsečíků paprsku se scénou, tedy s každým bodem dopadu, směrem ke světelnému zdroji. Slouží k určení, zda na průsečík přímo dopadají paprsky ze světelného zdroje, či zda je v zákrytu. Pokud stínový paprsek mezi průsečíkem a světlem nenarazí na překážku, je zář vysílaná světelným zdrojem brána v potaz při výpočtu difúzní a zrcadlové složky osvětlovacího modelu.
- **Odražené** – odražený paprsek je vyslán v případě, že bod dopadu má odrazivou vlastnost povrchu, ve směru podle zákona odrazu světla.
- **Refrakční** – podobně jako odražený paprsek, v případě, že povrch v bodě dopadu má vlastnost lámat, či propouštět světlo, refrakční paprsek je vyslán v příslušném směru lomu světla. K refrakci bývají nejčastěji používány Fresnelovy rovnice. Typickým refrakčním povrchem je sklo, voda či kov.

Celý algoritmus se po vyslání primárních paprsků rekurzivně zanořuje až do stanovené hloubky rekurze či dosažení eliminačního kritéria (např. ruská ruleta). Stínové paprsky jsou

[5] Zdroj: http://www.csci.csusb.edu/tongyu/courses/cs621/images/ray/ray-casting5.png
[6] Zdroj: https://upload.wikimedia.org/wikipedia/commons/c/c9/Recursive_raytracing.svg



Obrázek 2.9: Jednoduché sledování paprsku; primární paprsky – zelené, odražené – červené, refrakční – modré, stínové – čárkované[6]

vyslány v každém bodě, kdežto odražené a refrakční pouze v závislosti na typu povrchu. Každý další vyslaný sekundární paprsek, vyjma stínových, vysílá další sekundární paprsky.

Výsledná barva pixelu, skrz který prochází primární paprsek na rovině obrazu, se tedy určuje rekurzivně na základě příspěvků všech paprsků a jejich lokálních osvětlovacích modelů.

Sledování cesty

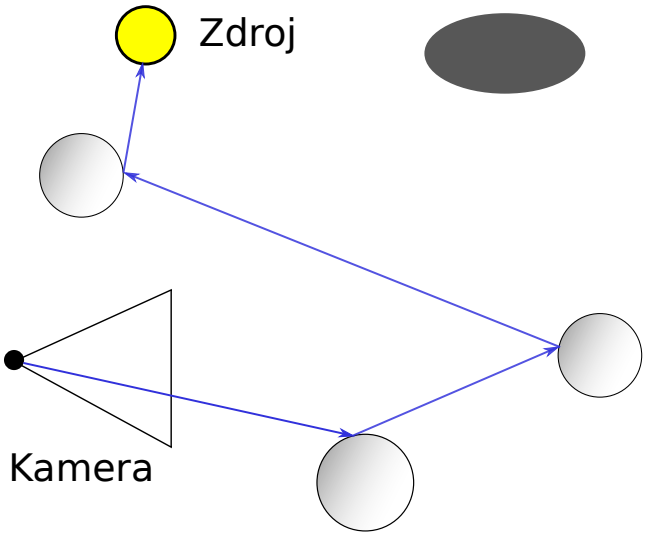
Další významnou metodou pro řešení globálního osvětlení je algoritmus sledování cest, angl. *path tracing*. Poprvé byla tato metoda představena společně se zobrazovací rovnicí v publikaci od James T. Kajiya [8] roku 1986.

Hlavním rozdílem této metody oproti ray tracingu je, že pro řešení zobrazovací rovnice využívá aproximace integrálu pomocí náhodné procházky metodou Monte Carlo. Základní forma path tracingu je matematicky popsána jako:

$$L(\mathbf{x}, \omega_o) = L_e(\mathbf{x}, \omega_0) + \frac{L(\mathbf{x}', -\omega_i) f_r(\mathbf{x}, \omega_i, \omega_0) \left| N_\mathbf{x} \cdot \omega_i \right|}{p(\omega_i)} \tag{2.18}$$

kde $p(\omega_i)$ je hustota pravděpodobnosti vzorkování směru ω_i.

Z rovnice je očividné, že ji lze řešit rekurzivním přístupem, kdy každý paprsek po zásahu ve scéně vygeneruje další paprsek z bodu kolize v náhodném směru na hemisféře nad průsečíkem. Tím se algoritmus zanořuje až do omezení hloubkou rekurze, případně do platnosti speciálního eliminačního kritéria, například typu *ruská ruleta* [16].



Obrázek 2.10: Ilustrace jednoduchého sledování cest

Na základě předchozího obrázku a rovnice 2.18 lze formálně definovat celý algoritmus pro výpočet barvy pixelu:

Algoritmus 1 Path tracing

```
1:  function PathTracing(Ray R)
2:      H ← Zjisti nejblížší průsečík paprsku ve scéně
3:      if H je prázdný nebo dosažena max. hloubka rekurze then
4:          return 0
5:      else
6:          N ← Vypočítej normálu v místě dopadu
7:          E ← Urči množství záře emitované zasaženým objektem
8:          R_N ← Vygeneruj nový paprsek
9:          p ← Urči pravděpodobnost vzorkování směru
10:         return E + PathTracing(R_N) · f_r(H.pos, R → R_N) · Dot(R_N, N) / p
```

Z Algoritmu 1 lze dedukovat, že není příliš efektivní. Zejména pro scény, které obsahují bodová světla je krajně nepoužitelný, protože pokud cestou paprsek neprotne světelný zdroj, celkový přírůstek je roven nule. Z tohoto, a mnoho dalších důvodů, se zavádí několik zásadních rozšíření:

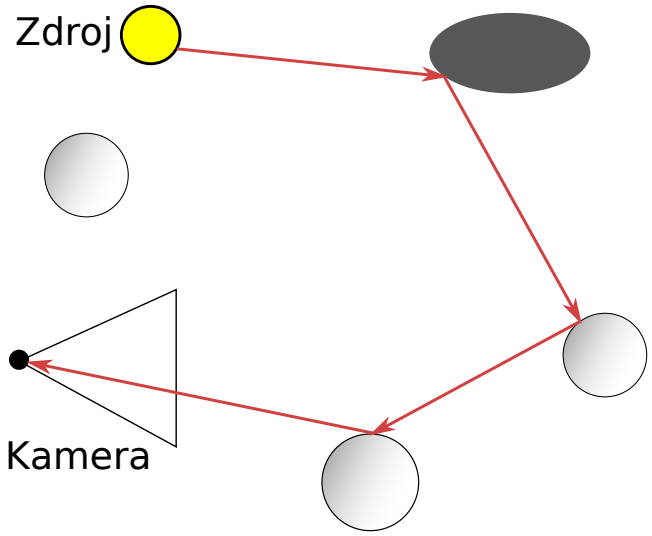
- **Přímé osvětlení** – zásadního zlepšení konvergence algoritmu lze dosáhnout vzorkováním světelných zdrojů a tím zavedení tzv. přímého osvětlení [15]. Během sledování paprsku se v každém bodě dopadu vyšle speciální stínový paprsek směrem k náhodnému bodu na světelném zdroji, který je, v případě, že není v zákrytu, nosičem přímého světla ze zdroje.
- **Vzorkování podle BRDF** – namísto uniformního generování směrů lze použití takovou hustotu pravděpodobnosti, která bude co nejvíce kopírovat rozptyl směrů podle BRDF. Například pro difúzní povrchy lze namísto hustoty $1/(2\pi)$ použít hustota $\cos(\theta)/\pi$, což zapříčiní, že paprsky jsou vrhány směrem pravděpodobnějšího odrazu.

- **Obousměrné sledování cesty** – používá kromě sledování cest od pozorovatele ke zdroji i cesty opačného směru, tedy od světelného zdroje k pozorovateli, viz metodu *sledování světla*. Tyto protichůdné cesty jsou mezi sebou propojovány, čímž lze dosáhnout maximálního světelného přírůstku. Tento princip byl poprvé představen E. Lafortunem [10]. Další výhodou tohoto rozšíření je možnost použití techniky *multiple importance sampling* [17], kde jsou jednotlivé cesty ohodnoceny váhou na základě vhodné heuristiky.

Použití výše zmíněných rozšíření sice algoritmus značně zkomplikuje, ovšem velmi redukuje počet zahozených paprsků z důvodů nulového přínosu.

Sledování světla

Opačným způsobem, oproti ray a path tracingu, je metoda sledování světla, angl. light tracing [5]. Na rozdíl od zmíněných předchozích metod, light tracing nevysílá primární paprsky od pozorovatele skrz rovinu obrazu, ale vzorkuje přímo světelné zdroje.



Obrázek 2.11: Ilustrace ideálního sledování světla, ačkoliv šance, že odražený paprsek zasáhne přímo kameru, je při použití jedno bodové kamery téměř nulová

Algoritmus pracuje s pevným počtem světelných paprsků, které jsou emitovány ze zdrojů. Po vytvoření vzorku na povrchu zdroje se vyšle paprsek směrem do kamery, pokud není zastaven kolizním objektem ve scéně a je v popředí kamery, docílí přímého osvětlení pixelu v rovině obrazu. Posléze je náhodným směrem do scény paprsek sledován, podobně jako v path tracingu. Při každém nárazu do objektu ve scéně je vyslán paprsek z průsečíku směrem do kamery. Pokud je cesta do kamery volná, protne rovinu obrazu a příslušnému pixelu se přiřadí světelný přírůstek. Jedná se pouze ale o sekundární osvětlení, největší přírůstek přináší přímý zásah paprsku do kamery, viz obrázek 2.11. Ovšem za použití bodové kamery (pinhole) je pravděpodobnost zásahu téměř nulová, což algoritmus značně omezuje a v základní formě bez kombinace s jinou metodou se téměř nepoužívá.

Základní rovnice light tracingu je odvozená z potenciálové rovnice 2.6. Zobrazuje množství vyzářeného toku z množiny S dopadajícího na pixel [5]:

$$\Phi(S) = \int_A \int_\Omega L_e(\mathbf{x}, \omega_o) \cdot W(\mathbf{x}, \omega_o) \cdot \cos(\theta)\, \mathrm{d}\omega\, \mathrm{d}A \tag{2.19}$$

kde $L_e(\mathbf{x}, \omega_o)$ určuje množství emitované záře z bodu $\mathbf{x}$ ve směru ω_o, $W(\mathbf{x}, \omega_o)$ je potenciál, rovněž v tomto bodu a směru, a θ je úhel mezi směrem ω_o a normálou v bodě $\mathbf{x}$.

Mapování fotonu

Mapování fotonu, angl. photon mapping [20], je oproti všem ostatním zmíněným metodám dvou-průchodový algoritmus globálního osvětlení. Metodu publikoval Henrik Wann Jensen [7] v roce 1996.

Tvorba fotonové mapy

V prvním průchodu algoritmu se tvoří tzv. fotonová mapa. Ze světelných zdrojů se vystřelují paprsky (fotony) směrem do scény. Podobně jako v jiných metodách, cesta fotonu po scéně je sledována, ovšem při každém nárazu se nepočítá osvětlovací model, ale úbytek zářivého toku Φ_e. Pravděpodobnost odrazu/lomu/pohlcení je dána materiálem. Směr odrazu či lomu a intenzita fotonu se vypočítává pomocí BSDF. Všechny parametry se ukládají do fotonové mapy. Pokud je foton již zcela pohlcen, trasování nepokračuje a jeho cesta končí.

Renderování

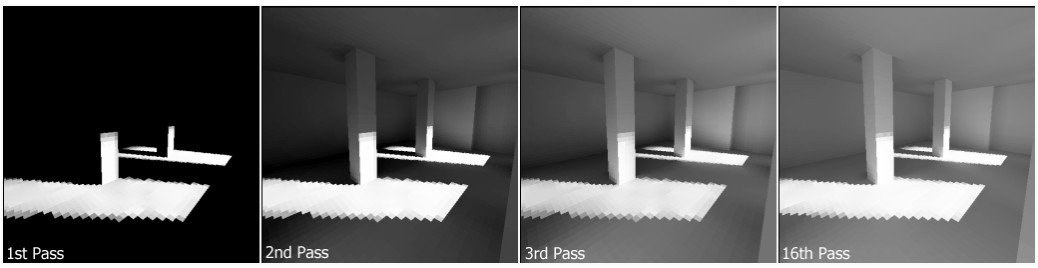
Po vytvoření fotonové mapy následuje fáze renderování. Podobně jako v jiných metodách, do každého pixelu v rovině obrazu je vyslán paprsek, který je trasován skrze scénu. Pro výpočet záře je v tomto případě použita zobrazovací rovnice. Z důvodu zvýšení efektivity se běžně rozděluje na 4 složky – přímé osvětlení, zrcadlové odrazy, kaustiky a měkké nepřímé osvětlení.

- **přímé osvětlení** – pro výpočet přímého osvětlení se používají stínové paprsky, které, podobně jako u ray tracingu či path tracingu, přenáší přímo světlo.
- **zrcadlové odrazy** – pro výpočet zrcadlového povrchu se velmi často využívají ray tracingové procedury.
- **kaustiky** – k tomuto výpočtu se přímo využívá fotonová mapa z prvního průchodu. Je důležité, aby mapa obsahovala dostatek fotonů, protože to je jediný možný zdroj informací o kaustikách ve scéně.
- **měkké nepřímé osvětlení** – opět je použita fotonová mapa, která ovšem už nepotřebuje být natolik veliká jako pro výpočet kaustik.

Výpočet záře z fotonové mapy nevyužívá všechny fotony v mapě, ale pouze N nejbližších. Tyto nejbližší fotony jsou v kouli, kolem vyhodnocovaného bodu, o ploše S. Posléze je množství světelného toku každého fotonu vyděleno touto plochou a vynásobeno koeficientem BRDF konkrétního fotonu. Celková suma těchto výsledků odpovídá množství záře na daném bodu ve směru paprsku, který ho zasáhl.

Radiozita

Radiozita [20] je jedna z mála metod, která je nezávislá na poloze a natočení kamery. Je založená na zákonu zachování energie, čímž je značně omezená pouze na uzavřené scény. Taktéž je limitována pouze na difúzní odraz světla, zrcadla nedokáže zpracovat.



Obrázek 2.12: Iterativní (progresivní) výpočet metodou radiozity. Počet iterací se zvyšuje zleva doprava.[7]

Pro samotný výpočet je využita zobrazovací rovnice v plošné formě s použitím BRDF. Před výpočtem je scéna rozdělena na velké množství malých plošek a současně jsou vypočítány faktory, které určují jak ploška ovlivňuje plošky kolem sebe, tzv. konfigurační faktory. Výpočet probíhá iterativně, lze tedy pozorovat průběžné výsledky.

Výsledkem algoritmu je množina obsahující hodnoty radiozity pro všechny plošky ve scéně. Pro zobrazení lze posléze využít například metodu ray tracing, která využije vypočítanou radiozitu a doplní zrcadlové odrazy, lomy světla a podobně.

2.7 Existující software

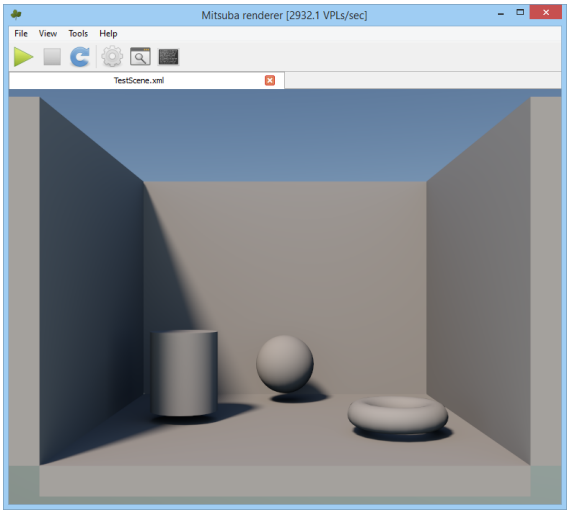
V dnešní době existuje řada aplikací pro vytváření fotorealistických obrazů, ačkoliv mnoho programů obsahuje kromě samotného renderingu i mnoho dalších věcí, jako například modelování scény, vytváření textur a podobně. V této kapitole je představen souhrn těch nejpodstatnějších.

Mitsuba Renderer

Velmi komplexním a obecným programem je Mitsuba Renderer. Jedná se o modulární aplikaci, která obsahuje velkou škálu různých integrátorů, které podporují víceméně všechny známé fotorealistické metody (ukázka výstupu na obrázku 2.13). Rovněž obsahuje mnoho modulů pro povrchové modely, textury, volumetrická data, světelné zdroje a podobně. Program je vytvářen formou open-source, tedy jsou přístupné všechny zdrojové kódy. Aplikace nabízí, jak textové prostředí, tak i grafické uživatelské rozhraní (viz obrázek 2.14), kde lze manipulovat se scénou a nahlížet do ní.

[7] Zdroj: https://upload.wikimedia.org/wikipedia/commons/f/f2/Radiosity_Progress.png

Obrázek 2.13: Obrázek vytvořený pomocí Mitsuba Renderer



Obrázek 2.14: Ukázka grafického uživatelského rozhraní programu Mitsuba Renderer

Obrázek 2.15: Obrázek vytvořený pomocí aplikace PBRT

Obrázek 2.16: Realistický výsledek pomocí programu V-Ray

PBRT

Druhá významná aplikace je PBRT, založená na knize o fyzikálním renderování. V této knize je popsán kompletní matematický aparát z této problematiky a rovněž velmi detailně implementace metody. Ačkoliv se zaměřuje spíše na metodu sledování cest a její významná rozšíření, lze pomocí tohoto programu dosáhnout velmi dobrých výsledků, jak se ostatně lze přesvědčit na obrázku 2.15. Opět jsou dostupné všechny zdrojové kódy.

Obrázek 2.17: Velmi přesvědčivý výsledek z programu Octane Render

V-Ray

V-Ray je asi nejpoužívanějším programem pro fotorealistické renderování. Je komerčním projektem, takže nelze nahlédnout do použitých algoritmů ani zdrojových kódu. Velkou výhodou je, že distribuce probíhá v rámci komponentů do modelovacích programů, jako např. Cinema4D, 3DS Max či Maya. Ačkoliv pořizovací hodnota je relativně vysoká, výsledky tomu zcela odpovídají. Výsledné obrázky lze pouze velmi složitě odlišit od skutečné fotografie, což lze zpozorovat např. na obrázku 2.16.

Octane Render

Octane Render je jeden z prvních komerčních programů, který přišel na trh s algoritmem pro fyzikálně založené renderování s plným využitím GPU. Jak již bylo řečeno, jedná se o komerční software vyvíjený firmou OTOY. Program nabízí široké možnosti zejména pro architekty a designéry. Kromě renderování celistvých scén, založených na pevné geometrii, Octane podporuje i volumetrické data, jako je např. mlha, kouř či oheň. Pořizovací cena je opět velmi vysoká, ovšem firma nabízí licence i pro studenty. Avšak výsledky opět ukazují velkou kvalitu, viz obrázek 2.17.

Maxwell

Posledním z řady velmi významných aplikací je Maxwell. Oproti ostatním zmíněným programům, Maxwell používá velmi specifický systém osvětlování, kde se snaží být co nejvíce realistický, jak jen to lze, oproti jiným programům, které využívají různé triky pro simulaci nekonečných světel a podobně. Na druhou stranu je relativně pomalejší než ostatní komerční produkty. Ačkoliv i přes jeho rychlost je velmi oblíbený mezi designéry a zejména architekty. Ilustrační výsledek lze vidět na obrázku 2.18.

Obrázek 2.18: Venkovní scéna vytvořená pomocí Maxwellu

Jak je již ostatně očividné, existuje velké množství softwaru pro fotorealistické zobrazování. Ovšem vzhledem ke složitosti fotorealizmu a časové náročnosti jsou komerční programy většinou velmi drahé. Naopak open-source projekty mnohdy nedosahují takové kvality, protože nemají dostatečné prostředky pro vývoj. Tato situace otevírá možnosti pro nové implementace vhodných metod za použití moderních technologií.

Kapitola 3

Zhodnocení současného stavu

V dnešní době existuje velké množství různých zobrazovacích metod. Lze je dělit podle zaměření, například pro herní průmysl se nepoužívají fotorealistické metody, naopak pro filmové efekty je fotorealizmus používám ve velkém množství.

Nejvíce je stále rozšířená ne-fotorealistická zobrazovací metoda rasterizací trojúhelníků, která se pomocí velkého množství bodových světel snaží co nejvíce přiblížit fotorealistickým efektům, ačkoliv toho ani zdaleka nedosahuje. Používá se zejména z důvodů své rychlosti, kdy je schopna vykreslovat v reálném čase. Z toho důvodu je právě používána zejména v herním průmyslu.

3.1 Hodnocení fotorealistických metod

Naopak z fotorealistických metod je nejčastěji používán ray tracing, zejména pro svoji implementační jednoduchost, což je ovšem kompenzováno značně limitujícími výslednými vlastnostmi. Další často užívanou metodou bývá path tracing, který v základní verzi není příliš efektivní, ale je otevřen velké rozšiřitelnosti, což je oproti jiným metodám značná výhoda.

	Rychlost	Kvalita	Dynamika	Rozšířit.	Implement.	GPU
Ray tracing	⋆⋆⋆	⋆⋆⋆	⋆⋆⋆	⋆	⋆⋆⋆⋆	⋆⋆⋆⋆
Path tracing	⋆⋆	⋆⋆⋆⋆	⋆⋆⋆	⋆⋆⋆⋆	⋆⋆⋆⋆	⋆⋆⋆
Bidir. path tr.	⋆⋆	⋆⋆⋆⋆⋆	⋆⋆⋆	⋆⋆⋆⋆	⋆⋆⋆	⋆⋆⋆
Photon map.	⋆⋆⋆	⋆⋆⋆	⋆⋆⋆	⋆⋆⋆	⋆⋆⋆	⋆⋆⋆
Radiozita	⋆⋆⋆⋆	⋆⋆⋆	⋆⋆⋆⋆	⋆⋆	⋆⋆⋆	⋆⋆⋆⋆

Tabulka 3.1: Srovnání jednotlivých metod

Výše zmíněná Tabulka 3.1 ukazuje jednoduché srovnání uvedených metod podle několika měřítek. Uvedené údaje jsou značně orientační a jsou závislé na konkrétních implementacích.

Rychlost Reprezentuje absolutní výkon konkrétní metody na stejném hardwaru.

Kvalita Představuje míru kvality, která je odvozena od množství šumu, podobnosti s předlohou a fyzikální korektnosti.

Dynamika Zhruba ohodnocuje, jak moc je metoda závislá na konkrétním úhlu pohledu, tedy jak je statická vs. dynamická. Zahrnuje i závislost na různých materiálech, světelných zdrojích a podobně.

Rozšiřitelnost Znázorňuje úroveň rozšiřitelnosti konkrétní metody, jak z hlediska možných rozšířeních algoritmu, tak i složitějších scén bez významné ztráty výkonu.

Implementace Ohodnocuje míru obtížnosti praktické implementace metody, nezahrnuje matematickou složitost rovnic, ale konkrétní složitost datových typů, podprogramů a potřebných knihoven.

GPU Poslední ohodnocení ukazuje míru použitelnosti metody na grafické kartě, což je zejména důležité pro budoucí vývoj vzhledem k rychlému zvyšování výkonu a paralelizace grafických karet.

Na základě předchozí tabulky lze říci, že fotorealistické algoritmy jsou velmi různorodé, nelze tedy přímo říci, která metoda je nejefektivnější či nejvýhodnější. Z hlediska kvality závisí zejména na tom, co je renderováno. Každá metoda má své výhody i nevýhody.

3.2 Technické parametry práce

Téma fotorealistické zobrazování jsem si zvolil zejména kvůli velkému zájmu o počítačovou grafiku. Dosavadní výuka byla zaměřená spíše na vykreslování rasterizací trojúhelníků, téma fotorealizmu bylo uvedeno pouze okrajově, ačkoliv bylo velmi zajímavé. Z toho důvodu jsem se chtěl zaměřit na metody řešení této problematiky. Při výběru konkrétního algoritmu rozhodoval nejvíce faktor možné výsledné kvality.

Na základě předchozích slov jsem k této práci zvolil metodu **sledování cest** (path tracing). Cílem práce je nastudovat potřebné informace, navrhnout algoritmus provádění této metody a v neposlední řadě implementace a vyhodnocení. Dalším cílem práce je implementovat rozšíření metody o obousměrné zpracování s podporou techniky multiple importance sampling.

Z hlediska implementačního charakteru jsou zahrnuty různé technické vlastnosti tak, aby splňovaly předpoklady zadání. Výčet vlastností implementace je následující:

1. Načítání scény pomocí jednoduchého XML formátu
2. Iterativní metoda sledování paprsku (path tracing)
3. Simulace povrchových vlastností pomocí BSDF s využitím Fresnelových rovnic
4. Rozšíření o obousměrné zpracování
5. Ohodnocení cest pomocí principu multiple importance sampling

Dále implementace zahrnuje mnoho vlastností, které jsou uvažovány jako podsystémy v předchozím výčtu, zejména jsou tím myšleny textury, světla či geometrické transformace.

Kapitola 4

Matematické odvození

V této kapitole je popsáno odvození všeho matematického aparátu použitého v implementaci. V první části je vysvětlen princip převodu zobrazovací rovnice na plošnou formu. Druhá část popisuje použití metody Monte Carlo pro aproximaci path tracingu a rovněž odvození estimátoru pro obousměrné zpracování. V poslední části je vysvětlen a ukázán princip multiple importance sampling.

4.1 Úprava zobrazovací rovnice

Zobrazovací rovnice (2.17) má v základní formě několik nedostatků. Zejména je velmi obtížné vyřešit integrál, který je integrovaný přes všechny směry hemisféry. Pro účely této práce je vhodnější převést rovnici do takového tvaru, který je integrovaný přes všechny body povrchu scény.

Převod zobrazovací rovnice na tří bodovou plošnou formu

Pro úspěšný převod na plošnou formu, je nutno rovnici upravit tak, aby byla nezávislá na směrech ω_o a ω_i. Z hlediska matematické korektnosti definujeme odchozí zář z bodu $\mathbf{x}'$ do bodu $\mathbf{x}$ jako:

$$L(\mathbf{x}' \to \mathbf{x}) = L(\mathbf{x}', \omega_o)$$

kde $\omega_0 = \overrightarrow{\mathbf{x}'\mathbf{x}}$, šipka označuje směr z bodu $\mathbf{x}'$ do $\mathbf{x}$. Podobnou nezávislost je nutno zavést i u BRDF koeficientu:

$$f_r(\mathbf{x}'' \to \mathbf{x}' \to \mathbf{x}) = f_r(\mathbf{x}', \omega_i, \omega_o) = f_r(\mathbf{x}', \overrightarrow{\mathbf{x}'\mathbf{x}''}, \overrightarrow{\mathbf{x}'\mathbf{x}}) \tag{4.1}$$

kde pro ω_o platí již definovaný směr a $\omega_i = \overrightarrow{\mathbf{x}'\mathbf{x}''}$. Poslední část nutná k dokončení transformace je vztah mezi úhlovým Ω a plošným A integrandem:

$$\mathrm{d}\omega = \frac{\mathrm{d}A|N_{\mathbf{x}''} \cdot \overrightarrow{\mathbf{x}''\mathbf{x}'}|}{\|\mathbf{x}'' - \mathbf{x}'\|^2}$$

Kompletní transformovaná rovnice integrovaná již vůči všem bodům povrchu, je definována:

$$L(x' \to x) = L_e(x' \to x) + \int_A L(x'' \to x') \cdot f_r(x'' \to x' \to x) \cdot G(x'' \leftrightarrow x')\mathrm{d}A_{x''} \tag{4.2}$$

kde pro zjednodušení rovnice je zaveden tzn. geometrický člen $G(x'' \leftrightarrow x')$, který lze definovat jako:

$$G(x'' \leftrightarrow x') = V(x'' \leftrightarrow x') \cdot \frac{|N_{x''} \cdot \overrightarrow{x''x'}| \cdot |N_{x'} \cdot \overrightarrow{x'x''}|}{\|x'' - x'\|^2}$$

Velmi podstatnou částí geometrické členu je funkce viditelnosti $V(x'' \leftrightarrow x')$, která nabývá hodnoty 1 v případě, že bod x' je přímo viditelný z bodu x'', tzn. lze mezi body vyslat paprsek tak, aniž by narazil do překážky, v opačném případě je funkce rovna 0.

Ideální přenosová rovnice

Předchozí rovnice (4.2) stále nemůže být přímo řešena, protože obsahuje na obou stranách stejnou proměnnou, řešení se nabízí pouze rekurzivním nahrazováním proměnné na pravé straně přímo pravou stranou, což vyústí v nekonečně složitý a dlouhý integrál, viz následující zjednodušený rozvoj:

$$\begin{aligned} L &= L_e(x) + \int_A L \,\mathrm{d}A = \\ &= L_e(x) + \int_A L_e(x) \,\mathrm{d}A + \int_A \int_A L \,\mathrm{d}A \,\mathrm{d}A = \\ &= L_e(x) + \int_A L_e(x) \,\mathrm{d}A + \int_A \int_A L_e(x) \,\mathrm{d}A\mathrm{d}A + \int_A \int_A \int_A L \,\mathrm{d}A \,\mathrm{d}A \,\mathrm{d}A = \dots \end{aligned}$$

Na základě práce E. Veache [16], lze rovnici (4.2) formulovat jako sumu přes sadu cest, kde cesta je definována jako $\overline{x} = \mathbf{x}_0\mathbf{x}_1 \dots \mathbf{x}_k$ pro $1 \leq k < \infty$ a kde $\forall\, \mathbf{x}_i \in A$.

$$\begin{aligned} L(x' \to x) = \sum_{k=0}^{\infty} \int_{A^{k+1}} & L_e(\mathbf{x}_0 \to \mathbf{x}_1) \cdot G(\mathbf{x}_0 \leftrightarrow \mathbf{x}_1) \cdot W_e(\mathbf{x}_{k-1} \to \mathbf{x}_k) \\ & \left(\prod_{i=1}^{k-1} f_r(\mathbf{x}_{i-1} \to \mathbf{x}_i \to \mathbf{x}_{i+1}) \cdot G(\mathbf{x}_i \leftrightarrow \mathbf{x}_{i+1}) \right) \mathrm{d}A(\mathbf{x}_0) \dots \mathrm{d}A(\mathbf{x}_k) \end{aligned} \tag{4.3}$$

Výhody plynoucí z předchozí rovnice jsou zejména přímost výpočtu, odstranění rekurze a možnost libovolného začátku cesty, což umožňuje zejména řešení principu obousměrného sledování cest. Je ovšem potřeba si uvědomit, že tato rovnice je ideální transportní rovnicí světla, kdy postupuje přímo od zdroje k pozorovateli.

Rozdělení přímé a nepřímé složky světla

Dalším, stále přetrvávajícím, problémem zobrazovací rovnice při využití v path tracingu je fakt, že v případě použití malých světelných zdrojů (například bodové zdroje), většina vzorkovaných cest nikdy zdroj nezasáhne a její celkový světelný přírůstek bude nulový. Tento nedostatek lze odstranit rozdělením přímého a nepřímého osvětlení [15] z odražené složky zobrazovací rovnice. Upravená odražená složka se nyní skládá z přímé a nepřímé části, viz následující rovnice (4.4).

$$\begin{aligned} L_r(\mathbf{x}, \omega_o) = & \int_A L_e(\mathbf{x}' \to \mathbf{x}) \cdot f_r(\mathbf{x}, \overrightarrow{\mathbf{x}'\mathbf{x}}, \omega_o) \cdot G(\mathbf{x}' \leftrightarrow \mathbf{x}) \,\mathrm{d}A_i \\ & + \int_\Omega L_r(\mathbf{x}', -\omega_i) f_r(\mathbf{x}, \omega_i, \omega_0) \cdot |N_x \cdot \omega_i| \mathrm{d}\omega_i \end{aligned} \tag{4.4}$$

kde první část rovnice odpovídá přímému osvětlení. Zbytek rovnice je zobrazen stále v úhlové formě pro jednodušší představu oddělení přímého světla.

4.2 Aplikace metody Monte Carlo

V předchozí části bylo ukázáno několik vylepšení zobrazovací rovnice. V této kapitole je zobrazen způsob řešení předchozích rovnic pomocí metody Monte Carlo.

Path tracing

Jak již bylo řečeno, rovnici (2.18) nelze řešit jiným způsobem, než-li rekurzí. Zavedením vztahu pro ideální přenos světla, oddělením přímého a nepřímého světla, a za pomoci Monte Carlo metody lze path tracing definovat jako:

$$L_p = \frac{1}{N} \sum_{i=0}^{N} \sum_{t=0}^{N_E} C_t \tag{4.5}$$

kde L_p označuje množství dopadající záře na daný pixel, N je počet vzorků pro výpočet integrálu pomocí metody Monte Carlo, N_E je maximální délka cesty a C_t je přínos jednotlivé cesty délky t.

Aproximovanou funkci přínosu cesty t lze pomocí ideální přenosové rovnice a přímého osvětlení definovat:

$$\begin{aligned} C_t = \frac{L_e(\mathbf{y}_0, \overrightarrow{\mathbf{y}_0\mathbf{x}_t})}{p(\mathbf{y}_0)} \cdot f_r(\mathbf{x}_t, \overrightarrow{\mathbf{x}_t\mathbf{y}_0}, \overrightarrow{\mathbf{x}_t\mathbf{x}_{t-1}}) \cdot G(\mathbf{x}_t \leftrightarrow \mathbf{y}_0) \cdot \\ \prod_{i=1}^{t-1} \frac{f_r(\mathbf{x}_i, \overrightarrow{\mathbf{x}_i\mathbf{x}_{i+1}}, \overrightarrow{\mathbf{x}_i\mathbf{x}_{i-1}}) \cdot |N_{\mathbf{x}_i} \cdot \overrightarrow{\mathbf{x}_i\mathbf{x}_{i+1}}|}{p(\overrightarrow{\mathbf{x}_i\mathbf{x}_{i+1}})} \end{aligned} \tag{4.6}$$

kde $L_e(\mathbf{y}_0, \overrightarrow{\mathbf{y}_0\mathbf{x}_t})$ určuje hodnotu přímého osvětlení z bodu $\mathbf{y}_0$ na světelném zdroji do bodu $\mathbf{x}_t$, podle hustoty pravděpodobnosti vzorkování bodu $\mathbf{y}_0$, násobeného koeficientem BSDF a geometrickým členem mezi těmito dvěma body. Zbylá část rovnice představuje cestu a působí jako nepřímé osvětlení, které je vypočítáno pomocí produktu z BSDF, násobené hodnotou kosinu úhlu θ, která je pro jednodušší výpočet uvažována jako hodnota skalárního součinu mezi normálou a směrem následujícího paprsku. Jednotlivé směry jsou vzorkovány podle hustoty pravděpodobnosti $p(\overrightarrow{\mathbf{x}_i\mathbf{x}_{i+1}})$.

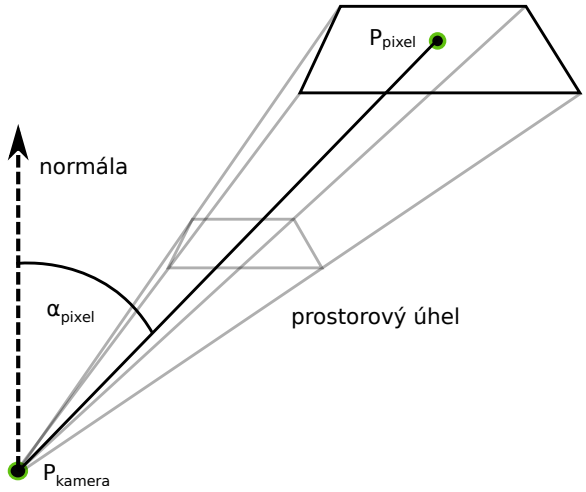
Předchozí rovnice dohromady fungují jako path tracing s přímým osvětlením, které je zajištěno vzorkováním světelného zdroje a přenosu světla na povrch pomocí stínového paprsku. Takto definovaný path tracing lze řešit jednoduchou iterativní formou.

Light tracing

Podobným způsobem, jako v předchozí části, lze odvodit i rovnice pro sledování světla. Je-li uvažováno, že potenciálovou rovnici 2.6 lze modifikovat stejným způsobem jako zobrazovací rovnici 2.17, tedy rozdělením přímé a nepřímé složky, lze za použití metody Monte Carlo definovat estimátor:

$$L_p = \frac{1}{N} \sum_{i=0}^{N} \sum_{t=0}^{N_L} C_s \cdot \frac{\|P_{pixel} - P_{kamera}\|^2}{A_{pixel} \cdot \cos(\alpha_{pixel})^2} \tag{4.7}$$

kde L_p je, podobně jako u path tracingu, množství dopadající záře na pixel, N_L určuje maximální délku cesty, C_s je přínos konkrétní cesty délky s. Zbylá část rovnice slouží k převodu diferenciálního toku na záři, kde P_{pixel} a P_{kamera} jsou body v prostoru, A_{pixel} je plocha pixelu a α_{pixel} je úhel mezi normálou v bodě kamery a směrem z kamery k bodu P_{pixel}. Intuitivní forma převodu mezi tokem a září je ukázána na obrázku 4.1.



Obrázek 4.1: Převod diferenciálního toku na zář – záře je vyjádřena jako tok na prostorový úhel a plochu. Vyznačený prostorový úhel je takový úhel, pod jakým je vidět pixel z bodu kamery. Vzhledem k použití bodové kamery, není uvažována diferenciální velikost plošky kamery při výpočtu převodu, viz rovnici 2.4.

Rovněž na základě předchozích rovnic lze definovat funkci přírůstku cesty pomocí ideální přenosové rovnice, podobně jako u path tracingu:

$$C_s = \frac{L_e(\mathbf{y}_0, \overrightarrow{\mathbf{y}_0\mathbf{y}_1})}{p(\mathbf{y}_0, \overrightarrow{\mathbf{y}_0\mathbf{y}_1})} \cdot f_r(\mathbf{y}_s, \overrightarrow{\mathbf{y}_s\mathbf{y}_{s-1}}, \overrightarrow{\mathbf{y}_s\mathbf{x}_0}) \cdot G(\mathbf{y}_s \leftrightarrow \mathbf{x}_0) \cdot \frac{W_e(\mathbf{x}_0, \overrightarrow{\mathbf{y}_t\mathbf{x}_0})}{p(\mathbf{x}_0)} \cdot \prod_{i=1}^{s-1} \frac{f_r(\mathbf{y}_i, \overrightarrow{\mathbf{y}_i\mathbf{y}_{i-1}}, \overrightarrow{\mathbf{y}_i\mathbf{y}_{i+1}}) \cdot |N_{\mathbf{y}_i} \cdot \overrightarrow{\mathbf{y}_i\mathbf{y}_{i+1}}|}{p(\overrightarrow{\mathbf{y}_i\mathbf{y}_{i+1}})} \tag{4.8}$$

kde první člen rovnice znázorňuje počáteční emitovanou záři z bodu $\mathbf{y}_0$ ve vzorkovaném směru podle dané hustoty pravděpodobnosti. Další tři členy rovnice představují přímo propojení bodu $\mathbf{y}_s$ do kamery. $W_e(\mathbf{x}_0, \overrightarrow{\mathbf{y}_t\mathbf{x}_0})$ je potenciál, který bude v případě použití dírkové kamery vždy roven 1, za splnění podmínky, že propojovaný bod se nachází v popředí kamery. Pravděpodobnost vzorkování bodu $\mathbf{x}_0$ bude rovněž vždycky rovna číslu 1, protože existuje pouze jediná možnost propojení s tímto bodem. Zbylou část rovnice tvoří produkt, který znázorňuje jednotlivé předchozí odrazy ve scéně. Skládá se ze součinu koeficientu BSDF a kosinem úhlu θ, kde pro jednodušší výpočet je využito skalárního součinu. Všechny generované směry odpovídají hustotě pravděpodobnosti $p(\overrightarrow{\mathbf{y}_i\mathbf{y}_{i+1}})$.

Pravděpodobnost

V obou zmíněných metodách (PT a LT) se velmi hojně vyskytují hustoty pravděpodobností, na základě kterých se vzorkují jednotlivé body a směry. Z toho důvodu je vhodné ukázat základní použité pravděpodobnosti při vzorkování. Souhrn pravděpodobností pro jednotlivá tělesa popisuje následující tabulka 4.1 [4].

Prostředí	Hodnota	Popis
koule	$\frac{1}{4\pi \cdot r^2}$	Pravděpodobnost vzorkování bodu na povrchu koule.
polokoule*	$\frac{1}{2\pi}$	Pravděpodobnost generování náhodného směru uniformě proporčně k prostorovému úhlu.
	$\frac{\cos\theta}{\pi}$	Vzorkování směru vzhledem k kosinově-váženému prostorovému úhlu
	$\frac{n+1}{2\pi}\cos^n\theta$	Zobecněný případ vážené pravděpodobnosti podle kosinu.
trojúhelník	$\frac{1}{A_{triangle}}$	Uniformní pravděpodobnost vzorkování bodu na povrchu, kde $A_{triangle}$ označuje plochu trojúhelníku

Tabulka 4.1: Tabulka pravděpodobností; * s poloměrem 1

4.3 Obousměrné sledování cest

Zavedením vztahů 4.6 a 4.8 lze nyní zapsat estimátor vyjadřující matematickou formulaci obousměrného sledování cesty [10, 16], angl. bidirectional path tracing, nyní již zkráceně pouze BDPT. Princip algoritmu BDPT je využití světelné a kamerové cesty, a jejich propojení, pro rychlejší konvergování scény. Celkový estimátor je definován:

$$L_p = \frac{1}{N}\sum_{i=0}^{N}\sum_{s=0}^{N_L}\sum_{t=0}^{N_E} \cdot w_{s,t} \cdot C_{s,t} \tag{4.9}$$

kde opět N je počet vzorků na pixel, N_L maximální délka světelné cesty, N_E maximální délka cesty od pozorovatele, $w_{s,t}$ je váhová funkce, viz kapitola 4.4, a $C_{s,t}$ je funkce přírůstku kombinované cesty.

Pro výpočet konkrétního přírůstku nelze využít jednotného vzorce, ale je potřeba brát v úvahu 4 konkrétní případy co mohou nastat.

$\mathbf{s = 0, t = 0}$ Nejjednodušší případ uvažuje přímou cestu ze světla do kamery, za předpokladu, že v cestě nestojí žádný kolizní předmět

$$C_{s,t} = L_e(\mathbf{y}_0, \overrightarrow{\mathbf{y}_0\mathbf{x}_0}) \cdot G(\mathbf{y}_0 \leftrightarrow \mathbf{x}_0) \tag{4.10}$$

$\mathbf{s > 0, t = 0}$ V případě, že cesta obsahuje pouze paprsky vystřelené ze světla, pak se jedná o čistý případ light tracingu z rovnice 4.8, kdy z jednotlivých průsečíků ve scéně jsou vyslány stínové paprsky do kamery, které přenáší sekundární světlo.

$\mathbf{s = 0, t > 0}$ V opačné situaci, kdy ve světelné cestě nejsou emitovány žádné paprsky, se jedná o klasický path tracing s přímým osvětlení ze světelného zdroje (s = 0 zahrnuje bod y_0 na povrchu světelného zdroje), který je definovaný v rovnici 4.6.

$\mathbf{s > 0, t > 0}$ Nejčastějším případem při použití BDPT je kombinovaná cesta, složená, jak ze světelné cesty, tak z cesty od pozorovatele, kromě přímého spojení do kamery při LT, přímého osvětlení při PT dochází navíc ke spojení jednotlivých bodů PT a LT

mezi sebou navzájem, viz následující rovnice 4.11.

$$C_{s,t} = \frac{L_e(\mathbf{y}_0, \overrightarrow{\mathbf{y}_0\mathbf{y}_1})}{p(\mathbf{y}_0, \overrightarrow{\mathbf{y}_0\mathbf{y}_1})} \cdot G(\mathbf{y}_s \leftrightarrow \mathbf{x}_t) \cdot f_r(\mathbf{x}_t, \overrightarrow{\mathbf{x}_t\mathbf{y}_s}, \overrightarrow{\mathbf{x}_t\mathbf{x}_{t-1}}) \cdot f_r(\mathbf{y}_s, \overrightarrow{\mathbf{y}_s\mathbf{y}_{s+1}}, \overrightarrow{\mathbf{y}_s\mathbf{x}_t}) \left(\prod_{i=1}^{t-1} \frac{f_r(\mathbf{x}_i, \overrightarrow{\mathbf{x}_i\mathbf{x}_{i+1}}, \overrightarrow{\mathbf{x}_i\mathbf{x}_{i-1}}) \left| N_{\mathbf{x}_i} \cdot \overrightarrow{\mathbf{x}_i\mathbf{x}_{i+1}} \right|}{p(\overrightarrow{\mathbf{x}_i\mathbf{x}_{i+1}})} \right) \left(\prod_{i=1}^{s-1} \frac{f_r(\mathbf{y}_i, \overrightarrow{\mathbf{y}_i\mathbf{y}_{i+1}}, \overrightarrow{\mathbf{y}_i\mathbf{y}_{i-1}}) \left| N_{\mathbf{y}_i} \cdot \overrightarrow{\mathbf{y}_i\mathbf{y}_{i+1}} \right|}{p(\overrightarrow{\mathbf{y}_i\mathbf{y}_{i+1}})} \right) \tag{4.11}$$

Řešení viditelnosti, v předchozí rovnici, mezi jednotlivými body je zahrnuto v geometrickém členu.

Zavedením obousměrného sledování cest lze získat vyšší světelný přírůstek ze všech cest, i za cenu vyšší algoritmické složitosti. Ovšem jak je dokázáno v práci E. Veache [16, 17], efektivnost BDPT lze výrazně posílit zavedením vhodného vzorkování dle důležitosti pro oba typy cest, viz následující podkapitola.

4.4 Multiple importance sampling

Algoritmus obousměrného sledování cest je sám o sobě velmi komplexní, což znamená, že kromě vyššího světelného přírůstku přináší i značný šum, způsobený současným vzorkováním BRDF a světelného zdroje. Pro redukci takto způsobeného šumu se používá, právě již v předchozí podkapitole zmíněná, váhová funkce.

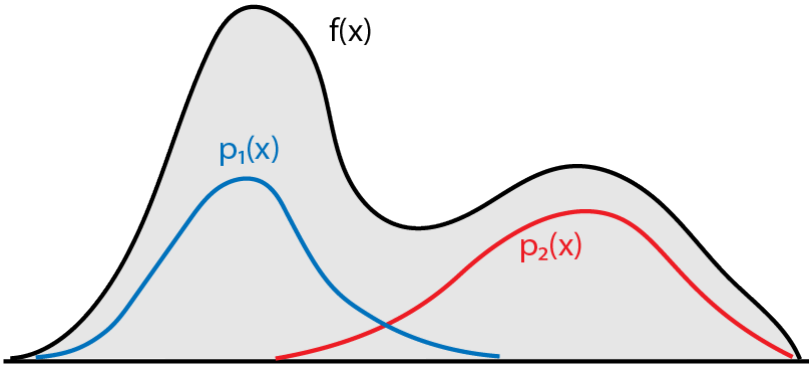
Možností jak definovat váhovou funkci je mnoho. Jeden z možných postupů je vyloučit takové světelné cesty, resp. cesty od pozorovatele, které v daném případě zvyšují množství šumu ve scéně. Tato varianta by značně redukovala šum, ale současně i rychlost konvergování scény, protože by došlo k zahození již plně vzorkovaných a vypočítaných cest.

Efektivnější způsob jak definovat váhovou funkci je použití principu multiple importance sampling [17]. V zjednodušeném pojetí se MIS snaží najít takovou kombinaci vzorkovacích technik, aby ve všech případech byla váha rozprostřena tak, aby se co nejvíce napodobila hodnota integrandu, viz obrázek 4.2.

Tohoto efektu lze dosáhnout použitím vhodné heuristiky, které definoval E. Veach a J. Guibas [17, 16]. Nejvhodnější je použití tzv. mocninné heuristiky (angl. power heuristic) s hodnotou exponentu $\beta = 2$. Základní forma mocninné heuristiky je podle E. Veache definována:

$$w_{s,t} = \frac{p_{s,t}^2}{\sum_{i=0}^{s+t} p_{i,s+t-i}^2} = \frac{1}{\sum_{i=0}^{s+t} (p_{i,s+t-i}/p_{s,t})^2} \tag{4.12}$$

kde p jsou pravděpodobnosti cest v dané konfiguraci s, t.



Obrázek 4.2: Multiple importance sampling: nejčastější případ; MIS je využito ke kombinaci obou vzorkovacích technik, odpovídajících jednotlivým hustotám pravděpodobnosti, do jedné, robustní, kombinované techniky

Výpočet pravděpodobnosti cest

Pro použití MIS je velmi důležitý výpočet pravděpodobností jednotlivých cest. V BDPT je tento počin značně komplikovaný, protože se sledují cesty z obou stran, je tedy nutné zvolit vhodné označení. Následující rovnice jsou založeny na práci I. Georgieva [6].

$$\overrightarrow{p_i}(\overline{x}) = \begin{cases} p_a(\overline{x}) & \text{pro } i = 0 \\ \overrightarrow{p_{\sigma,i}}(\overline{x}) \cdot \overrightarrow{g_i}(\overline{x}) & \text{jinak} \end{cases} \tag{4.13}$$

$$\overrightarrow{p_{\sigma,i}}(\overline{x}) = \begin{cases} p_\sigma(p_0 \to p_1) & \text{pro } i = 1 \\ p_\sigma(p_{i-2} \to p_{i-1} \to p_i) & \text{pro } i \geq 2 \end{cases} \tag{4.14}$$

$$\overrightarrow{g_i}(\overline{x}) = \frac{\cos\theta}{\|\mathbf{x}_i - \mathbf{x}_{i-1}\|} \tag{4.15}$$

Rovnice 4.13 představuje přímou hustotu pravděpodobnosti vzorkování části cesty $\overline{x}$, p_a je pravděpodobnost s ohledem na plochu, $p_{\sigma,i}$ je naopak pravděpodobnost s ohledem na prostorový úhel. Prostředek pro konverzi mezi plošnou a úhlovou pravděpodobností je označen jako $\overrightarrow{g_i}$.

Podobně jako předchozí pravděpodobnosti lze definovat ty, co jsou v opačném směru, pouze s tím rozdílem, že nebudou vyhodnocovány od začátku cesty ($i = 0$), ale od konce ($i = k$), rovněž je nutno změnit šipku vyznačující směr. Matematický produkt jednotlivých hustot pravděpodobností $\overrightarrow{p_i}(\overline{x})$, resp. $\overleftarrow{p_i}(\overline{x})$, určuje dohromady celkovou hustotu pravděpodobnosti vzorkování kompletní cesty $p_s(\overline{x})$, resp. $p_t(\overline{x})$. Celková hustota pravděpodobnosti pro spojení separovaných cest je $p_{s,t} = p_s(\overline{x}) \cdot p_t(\overline{x})$.

Úprava mocninné heuristiky

Aktuální forma mocninné heuristiky 4.12 umožňuje vyhodnocení váhy pouze na základě celé cesty, tzn. bylo by nutno celou cestu ukládat, včetně všech pravděpodobností. Podle Georgieva et al. [6] lze rovnici zjednodušit rozdělením na částečné váhy pro kameru a světlo.

$$
\begin{aligned}
w_{s,t} &= \frac{1}{\sum_{i=0}^{s-1}(p_{i,s+t-i}/p_{s,t})^2 + 1 + \sum_{i=s+1}^{s+t}(p_{i,s+t-i}/p_{s,t})^2} \\
&= \frac{1}{w_{light,s-1} + 1 + w_{camera,t-1}}
\end{aligned}
\tag{4.16}
$$

Vyhodnocení celé MIS váhy v rovnici 4.16 znamená výpočet světelné a kamerové váhy pro konkrétní konfiguraci. Tyto váhy ovšem mohou být vyhodnoceny progresivně během sledování obou cest (i je počítáno od začátku konkrétní cesty, tedy buď od senzoru kamery nebo bodu povrchu na světelném zdroji) [6]:

$$
\begin{aligned}
w_{camera,i} &= \frac{\overleftarrow{p_i}(x)}{\overrightarrow{p_i}(x)}(w_{camera,i-1} + 1) \\
w_{light,i} &= \frac{\overleftarrow{p_i}(x)}{\overrightarrow{p_i}(x)}(w_{light,i-1} + 1)
\end{aligned}
\tag{4.17}
$$

Ovšem z hlediska implementace stále nelze váhy 4.17 přímo progresivně počítat. Zejména z toho důvodu, že nelze vypočítat zpětnou pravděpodobnost, protože ještě není znám další bod zásahu, tzn. $\overleftarrow{p_i}(x)$ nemůže být jednoduše vyhodnocena. Rovněž $w_{camera,i-1}$ resp. $w_{light,i-1}$ nelze vypočítat protože je nutno znát PDF vzorkování z vertexu i do $i-1$, kterou ovšem lze odvodit pouze z vertexu $i+1$, protože opačná orientace příchozího vektoru do vertexu $i+1$ směr pro vyhodnocení BRDF v opačné orientaci a tedy pro určení pravděpodobnosti vzorkování do $i-1$. Aby se předešlo těmto problémům, Georgiev [6] zavedl substituci:

$$
\begin{aligned}
w_i &= \overleftarrow{p_i}(x)(vc_i + vcm_i) \\
vc_i &= \frac{1}{\overrightarrow{p_i}} \\
vcm_i &= \frac{\overleftarrow{g_{i-1}}}{\overrightarrow{p_i}}(vcm_{i-1} + \overleftarrow{p_{\sigma,i-2}}vc_{i-1})
\end{aligned}
\tag{4.18}
$$

kde právě zavedené substituce 4.18 se vyhýbají již zmíněným problematickým případům a umožňují přímou implementaci pomocí ukládání a aktualizování koeficientů vc_i a vcm_i.

Výchozí nastavení koeficientů

Pro úplnou formální korektnost je nutno definovat výchozí hodnoty pro zavedené koeficienty [6] v několika základních situacích.

Ze strany světla Výchozí nastavení pro cesty vedoucí ze světla závisí nejen na pravděpodobnosti vzorkování směru, ale rovněž i na velikosti plochy světla $p_0^{connect}$ a pravděpodobnosti vzorkování bodu p_0^{trace}.

$$
\begin{aligned}
vcm_1 &= \frac{p_0^{connect}}{p_0^{trace}} \cdot \frac{1}{\overrightarrow{p_1}} \\
vc_1 &= \frac{\overleftarrow{g_0}}{p_0^{trace} \cdot \overrightarrow{p_1}}
\end{aligned}
$$

Ze strany kamery Koeficienty pro výpočet MIS cesty směřující z kamery jsou relativně jednodušší na výpočet. Je nutno znát pouze celkový počet cest $n_{samples}$ a pravděpodobnost

vzorkování $\overleftarrow{p_1}$. Vzhledem k použití jedno bodové kamery není uvažována plocha kamery a pravděpodobnost vzorkování bodu.

$$vcm_1 = \frac{n_{samples}}{\overrightarrow{p_1}}$$
$$vc_1 = 0$$

Speciální případy

Každý typ světelného zdroje má jiné parametry, jinou velikost a jiné koeficienty pro výpočet MIS. Pozornost je potřeba ubírat zejména na delta světla, tzn. bodové světlo a jednobodové kuželovité světlo. Jelikož tyto světla se nenacházejí v běžném světě a nemají žádný fyzický povrch, tedy nemohou být zasaženy paprskem ve scéně, nelze vyhodnotit zpětnou pravděpodobnost $\overleftarrow{p_0}$. Z toho důvodu se zavádí speciální podmínka pro delta světla:

$$vc_i = 0$$

Druhým speciálním případem jsou nekonečná světla. Simulovat nekonečné světlo (např. slunce) není prakticky možné, protože to nedovoluje path integral framework [16], který dokáže pracovat pouze s konečným plošným rozměrem. Jak Georgiev ukázal ve své práci [6], lze si k tomuto ale pomoci předefinováním několika pravděpodobností. Lze změnit závislost $p_0^{connect}$, p_0^{trace} a $\overleftarrow{p_0}$ z plošné formy na úhlovou, díky čemuž lze uvažovat nekonečný směr. Zároveň je nutno nastavit $\overleftarrow{g_0} = 1$, protože konverzní faktor ztrácí smysl, pokud hustoty pravděpodobnosti jsou již v úhlové formě. Poslední úprava zahrnuje změnu $\overrightarrow{p_1}$, kterou je potřeba definovat jako:

$$\overrightarrow{p_1} = p(\hat{\mathbf{y}}) \cdot \cos\theta_{1\to 0}$$

kde $\hat{\mathbf{y}}$ označuje bod na rovině, která je kolmá ke směru ze světla $\overrightarrow{\mathbf{y}_0}$ do scény a je tečnou rovinou ke kouli kolem scény, čímž lze získat konkrétní konečný bod. Z toho důvodu $\cos\theta_{1\to 0}$ označuje úhel mezi normálou v bodě zásahu ve scéně $\mathbf{y}_1$ a směrovým vektorem $-\overrightarrow{\mathbf{y}_0}$.

Kapitola 5

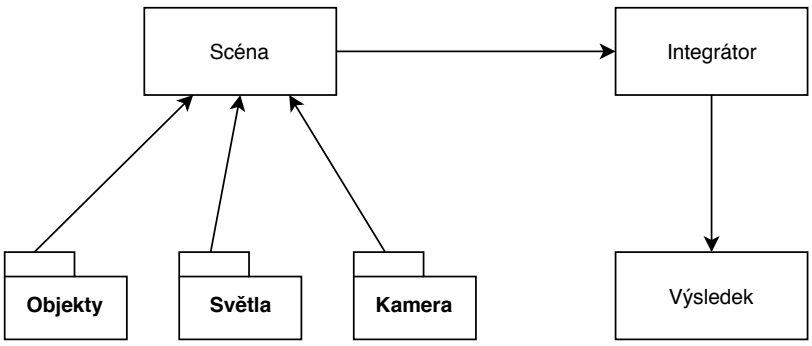
Popis implementace práce

Tato kapitola popisuje implementační část bakalářské práce. V první části je popsán obecný algoritmický návrh programu, posléze jsou podrobně rozebrány jednotlivé části.

Algoritmus je implementován čistě s využitím CPU, zejména kvůli složité matematické podobě. Pro akceleraci je využito vícevláknového principu, kde integrátor rovnoměrně dělí úkony mezi vhodný počet vláken. Součástí implementace je konfigurační soubor pro automatickou dokumentaci Doxygen.

5.1 Návrh

Při tvorbě návrhu byl kladen velký důraz na abstrakci a modulárnost celého programu tak, aby bylo možno jednotlivé komponenty přidávat a upravovat bez nutnosti jejich složitého zapojení do kódu. Základní schéma programu je tvořeno dvěma bloky – scéna a integrátor. Scéna obsahuje kontejner se všemi objekty a světly. Rovněž nastavuje a ovládá kameru. Oproti tomu integrátor je nezávislý modul, který scénu renderuje na základě nastavených parametrů – rozlišení, počet vzorků na pixel a podobně. Návrh uvažuje objekty jako abstraktní entitu, ze které se jednotlivé konkrétní objekty odvozují. Povrch objektu je tvořen rovněž abstraktním materiálem a splňuje vlastnosti zvolené BSDF. Základní model je ukázán na obrázku 5.1.



Obrázek 5.1: Blokové schéma návrhu základní činnosti programu

5.2 Reprezentace scény

Vzhledem k velkému rozšíření použití trojúhelníkových sítí je scéna rovněž založena na trojúhelníkové geometrii. Implementačně scénu tvoří abstraktní objekty (třída `Object`), které jsou konkretizovány na tři základní typy – koule, trojúhelník a mesh.

Implementace objektu typu koule je zejména kvůli jednoduchému analytickému popisu, čímž lze dosáhnout velmi přesné reprezentace, bez nutnosti použití velkého počtu trojúhelníků. Naopak trojúhelník je základním kamenem reprezentace libovolných tvarů ve scéně, kde pro jeho popis postačují tři body v prostoru, bez nutnosti implicitní specifikace normál. Posledním typem objektu je mesh, který představuje celou síť trojúhelníků a současně zapouzdřuje možné textury pro celý objekt, nikoliv pouze pro jediný trojúhelník (zejména z toho důvodu je reprezentován jako zvláštní typ objektu). Geometrie tohoto typu objektu je načítaná pomocí externí knihovny *Assimp* [9].

Načítání scény

Během spuštění programu lze scénu vybrat nebo načíst dvěma způsoby. První způsob je zvolením vstupního souboru jako programový argument. Tento soubor musí obsahovat správně zapsanou scénu pomocí jednoduchého vytvořeného XML formátu. Jednotlivé položky, kterými lze scénu definovat a nastavit jsou popsány v příloze A.

Druhý způsob spuštění programu je vybráním jedné ze zabudovaných scén. Program nabízí několik předdefinovaných ukázkových scén, které jsou různě složité a obsahují velmi rozdílné prvky, na čemž lze vyzkoušet jednotlivé funkce a náležitosti programu.

Světelné zdroje

Kromě základní geometrie, další nedílnou součástí scény jsou světelné zdroje. Z hlediska implementace jsou světla rovněž uvažovány jako abstraktní objekt – třída `Light`. Program implementuje většinu základních světel používaných v počítačové grafice: **bodové**, **plošné**, **směrové** (např. slunce), **světlo z pozadí** (environmentální mapa) a **kuželovité**.

Z hlediska implementace plošné světlo vyžaduje svázání s trojúhelníkem. Směrové světlo a světlo z pozadí jsou naopak mapovány na kouli kolem scény.

Zapouzdření scény a kamera

Geometrické objekty a světelné zdroje jsou zapouzdřeny uvnitř třídy `Scene`, která je napsána podle návrhového vzoru *singleton*, což znamená, že nelze renderovat více scén současně v jedné instanci programu. Tato třída mimo jiné obsahuje základní nastavení scény, jako je rozlišení obrazové roviny, počet vláken určených k renderování (workers) a počet vzorků na pixel.

Poslední část definice scény tvoří kamera (třída `Camera`). Pro inicializaci kamery je potřeba její výchozí pozice, směr pohledu a označení osy směřující nahoru. Rovněž lze definovat horizontální zorné pole kamery.

5.3 Popis povrchu pomocí BSDF

Velmi důležitým prostředkem pro fotorealistické efekty je právě povrchová reprezentace. Pro potřeby implementovaného programu je použita **bidirectional scattering distribution**

function (BSDF), která umožňuje simulovat například takové materiály, které propouštějí a lámou světlo.

Podobně jako u geometrických prvků, BSDF je implementování velmi abstraktně, zejména pro ulehčení pozdějšího nezávislého rozšíření o nové typy reprezentace. Tento abstraktní prvek představuje třída `BSDF`. Na základě této třídy jsou vytvářeny podtřídy, které implementují jednotlivé abstraktní metody potřebné pro vyhodnocení jednotlivých situací. Vzhledem k tomu, že ale dopředu nelze vytvořit BSDF pro každý bod povrchu konkrétní BSDF v závislosti na vlastnostech, je ke každému prvku (objektu) ve scéně přiřazena speciální materiálová třída (odvozená z třídy `Material`), která v případě, že algoritmus zasáhne konkrétní povrch, vytvoří instanci konkretizované třídy BSDF, kterou už v tomto případě lze zcela vyhodnotit.

Speciální případ povrchu je světelný zdroj. Z hlediska jednodušší implementace jsou světelné zdroje, které nemají δ-rozložení[1] pravděpodobnosti, uvažovány jako speciální typ povrchu, který obsahuje i vlastní implementaci BSDF.

Implementované BSDF

V programu je implementováno několik možných povrchových reprezentací. Každý druh BSDF byl vybrán kvůli jistým faktorům, které lze tímto otestovat, zejména dielektrický materiál umožňuje přesnou simulaci skleněných povrchů a kaustik.

Lambertovský model Základní reprezentace ideálního difúzního materiálu, tzv. Lambertovského, kde odrazivost je rovnoměrně stejná ve všech směrech (třída `DiffuseBSDF` a `DiffuseMaterial`).

Perfektní zrcadlo Základní simulace perfektního zrcadla, což znamená takový povrch, který odráží všechny paprsky přesně podle zákonu odrazu světla (třída `MirrorBSDF` a `MirrorMaterial`).

Lesklý povrch Kombinace předchozích dvou typů, kde se na základě pravděpodobnosti, vypočítané z odrazivosti jednotlivých složek, paprsek odráží buď podle zákona odrazu světla nebo náhodně dle Lamebrtovského modelu. Dohromady tento princip simuluje jednoduchý lesklý povrch (třída `GlossyBSDF` a `GlossyMaterial`).

Phongův model BSDF sestavená na základě Phongova osvětlovacího modelu, kdy přímo řeší difúzní a lesklou složku. Okolní složka (ambientní), je řešena přímo přes algoritmus path tracingu. Ačkoliv tento model je empirický a neodpovídá fyzikálním definicím, je vhodný pro použití a testování různých aspektů simulace (třída `PhongBSDF` a `PhongMaterial`).

Hladký dielektrický povrch Tato BSDF simuluje přechod mezi dvěma dielektrickými povrchy, kde první z nich je implicitně nastavený na vzduch a druhý lze definovat indexem refrakce (IOR). Mikroskopická struktura povrchu je uvažována jako perfektně hladká. Kromě indexu refrakce lze definovat i spekulární reflektanci a transmitanci, ačkoliv je doporučeno nechat ve výchozím nastavení, protože tato úprava porušuje fyzikální zákony `SmoothDielectricBSDF` a `SmoothDielectricMaterial`).

Konkrétní instance BSDF již nabízí přímo metody pro vyhodnocení hodnoty BSDF v daném bodě $\mathbf{x}$, vzorkování směru a určení pravděpodobnosti. Z pochopitelných důvodů je vyhodnocovací metoda validní ovšem pouze pro materiály s jinou než δ-distribucí.

[1] Typickým příkladem takového světla je bodové světlo, které lze zasáhnout pouze jediným směrem z daného bodu $\mathbf{x}$. Z toho důvodu je rozdělení pravděpodobnosti zásahu takového bodu tzv. Diracovo delta.

5.4 Algoritmus sledování cesty

Jak již bylo řečeno, použitým integrátorem v implementaci je tzv. **bidirectional path tracing** (BDPT). Kvůli relativně značné složitosti této metody je základní algoritmus rozdělen na části – zpracování světelné cesty, zpracování cesty od pozorovatele.

Hlavní programovou části algoritmu je třída `Bdpt`, která na základě nastavených parametrů rozdělí zpracování na vhodný počet BDPT integrátoru (`BdptTask`), kde každý běží na svém vlastním vlákně.

Algoritmus 2 Bidirectional path tracing - základní algoritmus

```
1:  function Task(Scene)
2:      definuj počet světelnych a kamerových cest k trasování
3:      for all světelné cesty do                          ▷ zpracování světelné cesty
4:          GenerateLightSample()
5:          while trasování do
6:              if nezasáhl objekt then
7:                  break
8:              BRDF ← inicializuj BSDF v bodě zásahu
9:              if BRDF není delta then
10:                 vertex ← naplň vertex podle bodu zásahu
11:                 ulož vertex do kontejneru světelných cest
12:                 JoinWithCamera()
13:             if cesta je příliš dlouhá then
14:                 break
15:             Sample() ← vzorkuj nový směr nebo ukonči cestu
16:         ulož výslednou délku cesty
17:     for all kamerové cesty do                          ▷ zpracování kamerové cesty
18:         GenerateEyeSample()
19:         color ← inicializuj na 0
20:         while trasování do
21:             if nezasáhl objekt then
22:                 break
23:             BRDF ← inicializuj BSDF v bodě zásahu
24:             if BRDF není delta then
25:                 color += DirectIllumination()
26:                 for all všechny odpovídající body světelné cesty do
27:                     color += ConnectVertices()
28:             if cesta je příliš dlouhá then
29:                 break
30:             Sample() ← vzorkuj nový směr nebo ukonči cestu
31:         ulož color do frame bufferu
```

Základní činnost jednotlivých `BdptTask` definuje Algoritmus 2. Vstupem algoritmu je scéna. Nejprve se určí počet cest k zpracování. Ten je určen na základě celkového počtu pixelů v rovině obrazu a počtu vláken. Každé vlákno dostane rovnoměrný podíl pixelů z roviny obrazu, ten posléze odpovídá počtu světelných a kamerových cest. V první fázi

algoritmus, tedy zpracování světelných cest, se vytvoří odpovídající množství cest, které jsou trasovány skrze scénu. Tyto cesty jsou vytvořeny vzorkováním světelných zdrojů. Pokud paprsek z dané cesty narazí během trasování do objektu, který nemá povrch s δ-distribucí, tak lze provést přímé spojení mezi kamerou a bodem zásahu, za předpokladu, že bod leží v rozsahu kamery a je viditelný. Rovněž se takový bod uloží do kontejneru, který je posléze použit v druhé části algoritmu. Následně je uložena celková délka světelné cesty bez bodů δ-distribucí.

V druhé části algoritmu se vysílají paprsky z kamery do scény. Podobně jako v první fázi, pokud paprsek zasáhne objekt s povrchem který není delta, je provedeno přímo osvětlení mezi aktuálním bodem zásahu a náhodným světelným zdrojem (pokud to lze provést – viditelnost). Rovněž se vytvoří propojení mezi přiřazenou světelnou cestou napříč mezi všemi body. Dále se vzorkuje další směr dokud není rozpoznán konec cesty pomocí maximální délky nebo kritériem ruské rulety. Po dokončení celé cesty je výsledný barva uložena do frame-bufferu.

Výše zmíněný Algoritmus 2 představuje pouze jeden průchod scénou, tzn. jeden vzorek na každý pixel. Pro kvalitní výsledek a redukci šumu je nutno celý proces mnohokrát opakovat a výsledek posléze vydělit zvoleným počtem vzorků na pixel.

Vzorkování cest

Důležitou částí implementace je vzorkování. Lze jej rozdělit na vzorkování povrchu světla, kamery a sekundárních paprsků během trasování.

Vzorkování světelného zdroje vybírá náhodně zvolené světlo, ze kterého se snaží emitovat paprsek, podle jeho vlastností (z konkrétní instance třídy světla). Výpočet pravděpodobností záleží na celkovém počtu světelných zdrojů a konkrétních zdrojích, např. velikost plochy u plošného světla, rozpětí úhlu u kuželovitého světla či velikost koule kolem scény u směrového světla (resp. zdroje z pozadí).

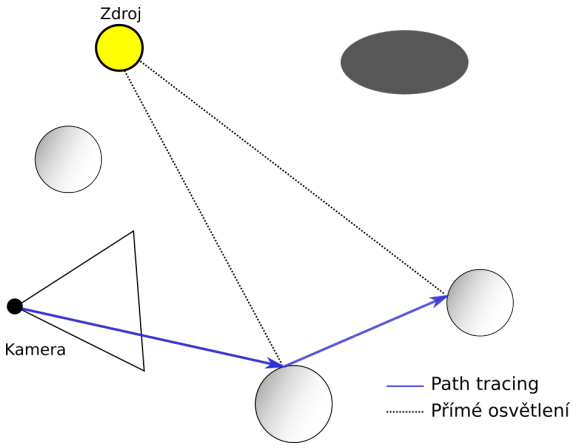
Na druhé straně, vysílání paprsků z kamery je podstatně jednodušší z toho důvodu, že v implementaci je použita tzv. dírková kamera (pinhole), která se skládá pouze z jednoho bodu, tzn. výchozí bod primárních paprsků je pořád stejný. Dále je vybrán 2D bod v rovině obrazu, který je převeden na 3D globální souřadnice, čímž lze určit směr vystřelení paprsku.

Třetí případ vzorkování je nejčastější. Během trasování může paprsek narazit do libovolných objektů ve scéně. Pro tuto situaci je použita vzorkovací metoda z BSDF, která určí nový směr a i pravděpodobnosti vzorkování této cesty. Současně jsou během tohoto procesu přepočítány koeficienty vc_i a vcm_i pro MIS a taktéž je určeno aktuální množství propouštěné záře. A v neposlední řadě je provedena tzv. **ruská ruleta**, která na základě největší složky záře a náhodně vygenerovaného vzorku určí, zda bude algoritmus pokračovat dále, či ukončí aktuální cestu.

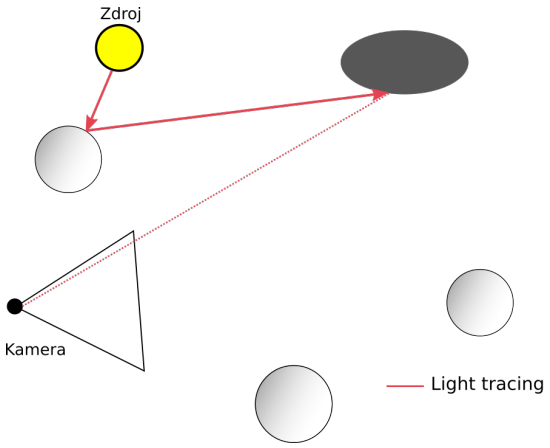
Přímé osvětlení

V základním algoritmu BDPT je zmíněno, že přímé osvětlení je počítáno pouze pro non-delta povrchy. Důvod je zejména takový, že např. při povrchu typu perfektní zrcadlo (což je nejtypičtější případ δ-distribuce) by se stejné množství světla odrazilo do scény takovým směrem, který není s největší pravděpodobností trasován, což by znamenalo zvýšení rozptylu a celkově nárůst šumu.

Z hlediska implementace je pro přímo osvětlení použita specifická metoda objektu náhodně zvoleného světla, která vystřelí foton směrem k vyhodnocovanému bodu. Pokud cestou foton nenarazí na překážku, tak BSDF vyhodnotí nový přírůstek záře, který je po-



Obrázek 5.2: Ilustrace přímého osvětlení v path tracingu



Obrázek 5.3: Přímé propojení bodu s kamerou v light tracingu

sléze škálován intenzitou vybraného světla a samozřejmě váhou MIS. Principiálně se tedy jedná pouze o stínový paprsek přenášející zář ze světelného zdroje, viz ilustraci 5.2.

Propojení s kameru

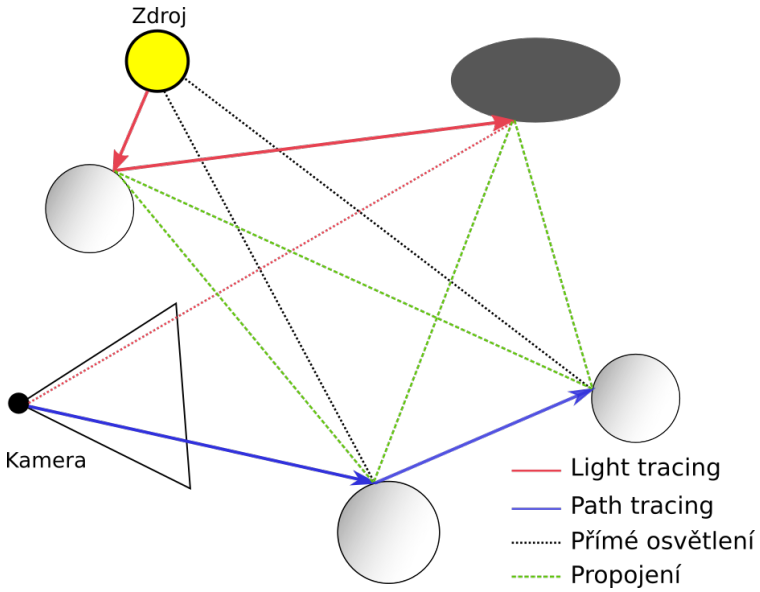
Propojení bodu s kamerou v light tracingu bylo již definováno v kapitole popisující samotný light tracing. Z implementačního pohledu se jedná o podobný princip stínového paprsku jako u přímého osvětlení, tzn. opět lze propojit pouze povrch, který nemá δ-distribuci. Pokud je daný bod z kamery přímo viditelný, tzn. není žádný kolizní objekt po cestě a je v hledáčku kamery, tak lze přímo přenést radianci daného bodu do kamery, viz obrázek 5.3. Celkový přírůstek záře je škálován váhou MIS, která je tvořena pouze váhou světelné cesty, a BSDF koeficientem.

Spojování cest

Implementace propojů mezi cestami je hlavní rys obousměrného sledování cest. Pro vytvoření spojení postačí vyslat paprsek v odpovídajícím směru mezi dvěma body světelné a kamerové cesty (viz obrázek 5.4). Matematicky se jedná přímo o výpočet obecného případu v rovnici 4.11. Je nutno vypočítat geometrický člen a vyhodnotit BSDF z obou cest ve směru propojovacího paprsku a rovněž zjistit, zda jsou body mezi sebou viditelné.

Zásah světelného zdroje

Z hlediska jednoduchosti není v Algoritmu 2 uveden případ přímého zásahu světelného zdroje, ačkoliv implementace tento problém přímo řeší. Pokud je zasažený povrch rozpoznán jako světelný (popsaný v kapitole o implementaci BSDF 5.3), tak je vypočítána přímo přenesená radiance ze zdroje do barvy pixelu. Tato zář je počítána pouze v závislosti na barvě a intenzitě zdroje. Váha MIS je vypočítána pouze pro kamerovou cestu, protože váha světla je v tomto případě rovna nule.



Obrázek 5.4: Kompletní BDPT zahrnující i spojování cest

Váhové funkce

Víceméně každá část algoritmu je ohodnocená váhou techniky MIS. Jelikož ale každá část vyžaduje jiný výpočet váhy, tato sekce shrnuje implementaci jednotlivých vah a vysvětluje jejich použití [6]. Z hlediska rychlosti a efektivnosti implementace je právě výhodnější použít progresivní vyhodnocování MIS, protože není nutno posléze ukládat celé cesty za účelem vyhodnocení celkové váhy, ale pouze jednu a vyhodnocovat váhy během trasování druhé, čímž lze ušetřit mnoho přístupů do paměti.

Propojení s kamerou Jak již bylo řečeno, propoj mezi cestou z LT a kamerou využívá pouze světelnou cestu, viz rovnici 5.1, kde n_{sample} je počet vzorků, v tomto případě celkový počet světelných cest. Vzhledem k použití jedno bodové kamery není uvažována pravděpodobnost zásahu bodu v kameře a celková plocha kamery.

$$\begin{aligned} w_{light,s-1} &= \frac{\overleftarrow{p_{s-1}}}{n_{samples}} \cdot (vcm_{s-1} + \overleftarrow{p_{\sigma,s-2}} \cdot vc_{s-1}) \\ w_{camera,0} &= 0 \end{aligned} \tag{5.1}$$

Zásah světelného zdroje Při zásahu světla je váha světelné cesty rovna nule, protože to je ideální případ transportu světla, kde není potřeba využívat světelnou cestu. Jelikož zásah do zdroje lze provést pouze u světel s fyzickou plochou, objevuje se výraz $p^{connect}$, který představuje pravděpodobnost vzhledem k velikosti plochy světla a p^{trace} představující pravděpodobnost zásahu, viz 5.2.

$$\begin{aligned} w_{light,s-1} &= 0 \\ w_{camera,t-1} &= p^{connect}_{t-1} \cdot vcm_{t-1} + p^{trace}_{t-1} \cdot \overleftarrow{p_{\sigma,t-2}} \cdot vc_{t-1} \end{aligned} \tag{5.2}$$

Přímé osvětlení Hlavní roli ve váhové funkci přímého osvětlení hraje velikost světelného zdroje, kde stejně jako u předchozího případu, $p^{connect}_0$ v rovnici 5.3 znamená pravděpodob-

nost vzhledem k ploše a p_0^{trace} pravděpodobnost zásahu.

$$\begin{aligned} w_{light,0} &= \frac{\overleftarrow{p_0}}{p_0^{connect}} \\ w_{camera,t-1} &= \frac{p_0^{trace}}{p_0^{connect}} \cdot \overleftarrow{p_{t-1}} \cdot (vcm_{t-1} + \overleftarrow{p_{\sigma,t-2}} \cdot vc_{t-1}) \end{aligned} \tag{5.3}$$

Spojování cest Pro nejčastější případ platí rovněž nejobecnější váhy, kde se již neuvažují pravděpodobnosti vzhledem k velikosti jednotlivých povrchů apod., jako v předchozích případech, ale pouze pravděpodobnosti vzorkování cest, viz 5.4.

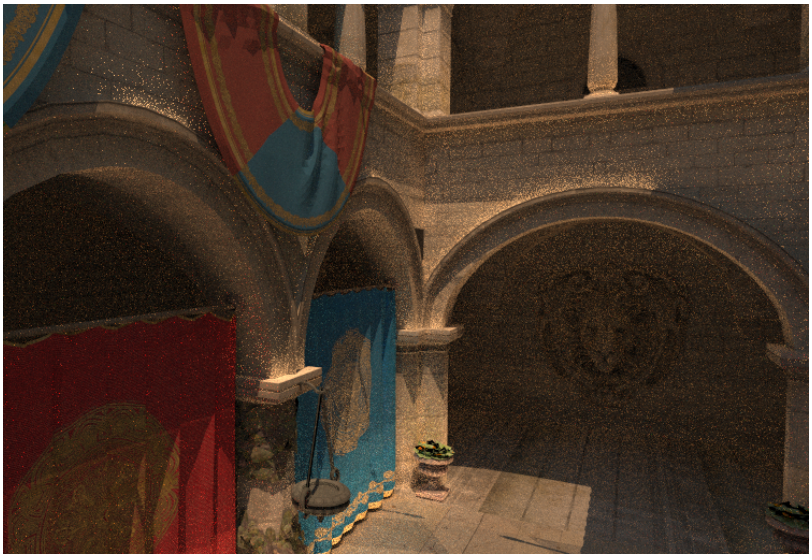
$$\begin{aligned} w_{light,s-1} &= \overleftarrow{p_{s-1}} \cdot (vcm_{s-1} + \overleftarrow{p_{\sigma,s-2}} \cdot vc_{s-1}) \\ w_{camera,t-1} &= \overleftarrow{p_{t-1}} \cdot (vcm_{t-1} + \overleftarrow{p_{\sigma,t-2}} \cdot vc_{t-1}) \end{aligned} \tag{5.4}$$

5.5 Další prvky implementace

Program implementuje několik dalších podpůrných prvků celého algoritmu. Zejména se jedná o akceleraci průchodu trojúhelníkovými sítěmi pomocí algoritmu kD-strom. Pro ohraničení jednotlivých buněk stromu je využit osově zarovnaný ohraničující rámeček (třída `AABBox`). Jelikož je ale tento algoritmus relativně komplikovaný a není v časových možnostech této práce ho implementovat, je použita již existující implementace prostřednictvím repozitáře *github*[2].

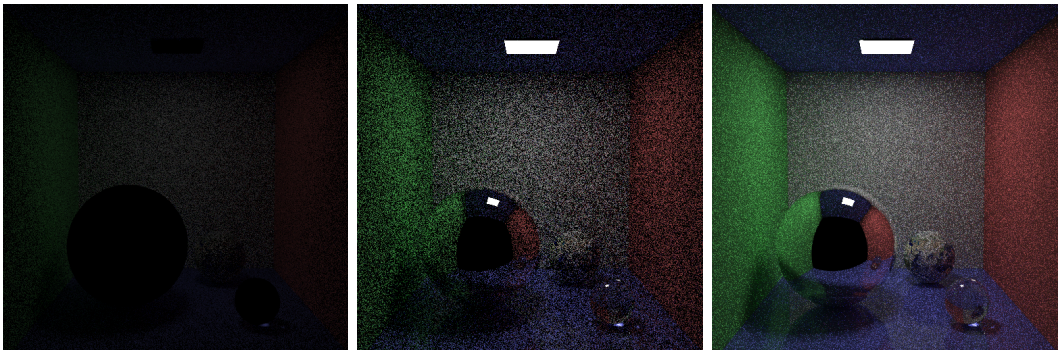
Implementovány jsou rovněž základní textury, které mohou byt namapovány, jak na trojúhelník, tak i na kouli. K určení hodnoty pixelu z UV souřadnic je využita bilineární interpolace (třída `ImageTexture`). Ukázka textur je ilustrována obrázkem 5.5.

V neposlední řadě, dalším důležitým prvkem programu je možnost používání základních 3D geometrických transformací pro jednodušší manipulaci zejména s velkými objekty. Implementovány jsou transformace: posunutí, škála a rotace (všechny lze nalézt v třídě `Transform`).



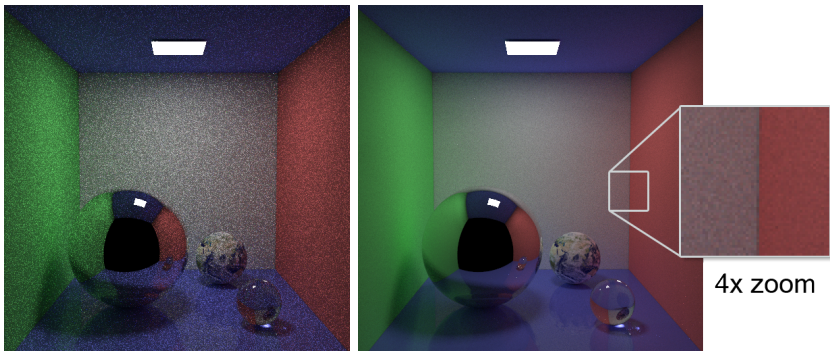
Obrázek 5.5: Sponza palác – ukázka renderování textur, 800 vzorků na pixel (~3,5 hodiny)

[2]Odkaz na konkrétní repozitář s autorskými právy lze nalézt v příslušném souboru uvnitř programové implementace.



(a) Light tracing (b) Path tracing (c) PT DI

Obrázek 5.6: Ukázka vykreslení jednoduché scény pomocí LT, PT a PT s přímým osvětlením. Všechny obrázky používají 60 vzorků na pixel.



(a) BDPT (b) BDPT MIS

Obrázek 5.7: Srovnání BDPT oproti BDPT s použitím multiple importance sampling. Oba obrázky obsahují 60 vzorků na pixel.

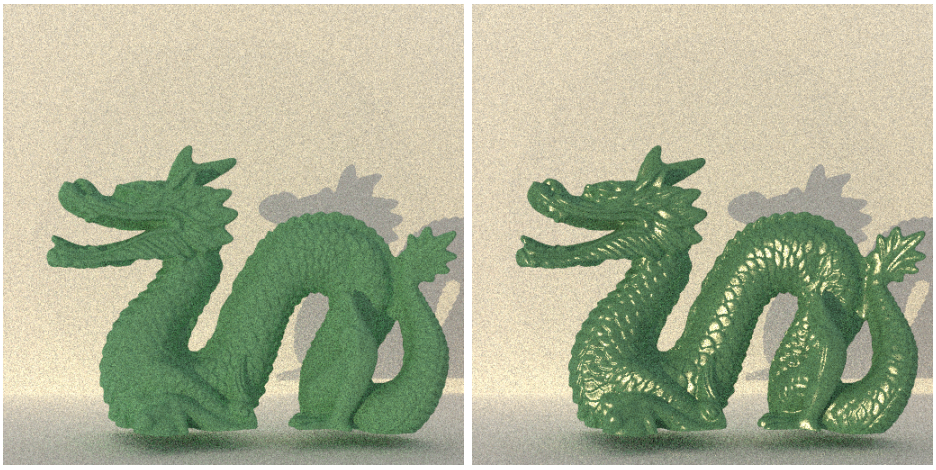
5.6 Testování a výsledky

Pro ověření správného výpočtu algoritmu bylo vytvořeno několik druhů testovacích sad. První část testování byla zaměřena na vytvoření výsledku z jednoduché scény a současně na porovnání jednotlivých metod, které obousměrné sledování cesty obsahuje jak podmnožinou v celém algoritmu.

Výsledek z jednoduché Cornell Box scény, která obsahuje tři analytické koule (perfektní zrcadlo, difúzní s texturou a sklo), lesklou podlahu a relativně velké prostorové světlo, za použití metody **light tracing**, lze vidět na obrázku 5.6a. Vzhledem k použití bodové kamery je jasné, že ve výsledném obraze bude pouze sekundární záře přenesená pomocí propojů s kamerou, což ale zabrání vykreslení povrchů s delta distribucí a světelných zdrojů.

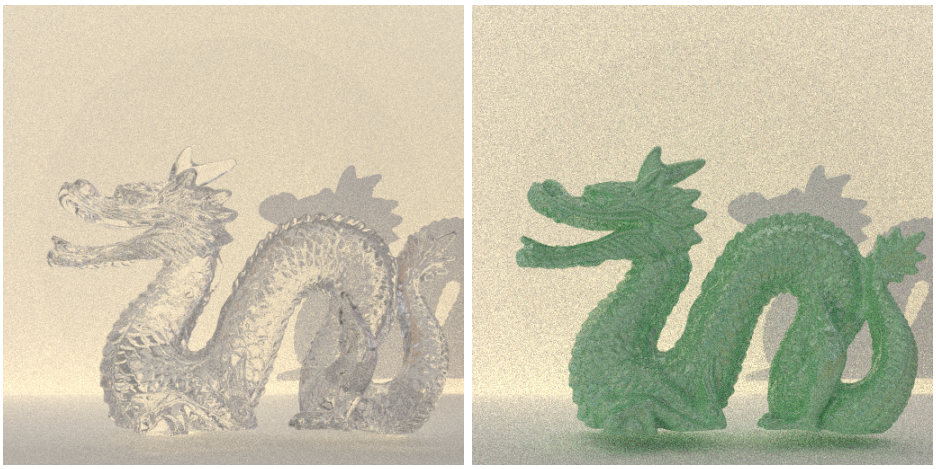
Při použití metody **path tracing** lze alespoň rozpoznat, jaké objekty se ve scéně nachází, viz 5.6b, ale množství šumu je, vzhledem k velikosti zdroje, stále enormní, což značí značnou neefektivitu této metody pro použití v takovémto druhu scény. Velkého zlepšení lze dosáhnout pomocí explicitního **přímého osvětlení**, což lze poznat na obrázku 5.6c, kde je šum už značně redukovaný.

Největší redukce šumu ovšem lze dosáhnout až za použití **BDPT**, viz obrázek 5.7a. Ačkoliv není tolik patrný rozdíl mezi předchozím obrázkem, je to zejména z důvodu rovno-



(a) Lambertův osvětlovací model (b) Phongův osvětlovací model

Obrázek 5.8: Srovnání Lambertova a Phongova modelu na relativně jednoduché scéně.



(a) Skleněný povrch (IOR 1.6) (b) Difúzně-spekulární povrch

Obrázek 5.9: Ukázka složitého skleněného a difúzně-spekulárního povrchu.

měrného rozložení vah. Při použití techniky MIS, která správně nastaví váhy pro jednotlivé vzorkovací techniky a značně tím sníží celkový rozptyl, lze vidět již relativně čistý výsledek na obrázku 5.7b.

Ačkoliv složitější metody vyžadují větší výpočetní čas, tak výrazně zvyšují kvalitu. Zejména při srovnání PT a BDPT může být jasně řečeno, že za stejný výpočetní čas lze pomocí BDPT dosáhnout mnohem lepší kvality, protože pro dosáhnutí stejného výsledků při použití BDPT stačí mnohem méně vzorků na pixel.

Vykreslování jednotlivých materiálů a složitějších scén

Druhá část testů se zaměřovala na vykreslování detailů jednotlivých materiálů. Současně bylo vyrenderováno několik komplexnějších scén, na kterých lze rozpoznat správné chování povrchů, zejména odlesků a kaustik. Na obrázku 5.8 lze vidět rozdíl při použití spekulární složky v Phongově modelu oproti čistě difúzní složce v Lambertovském modelu. Další

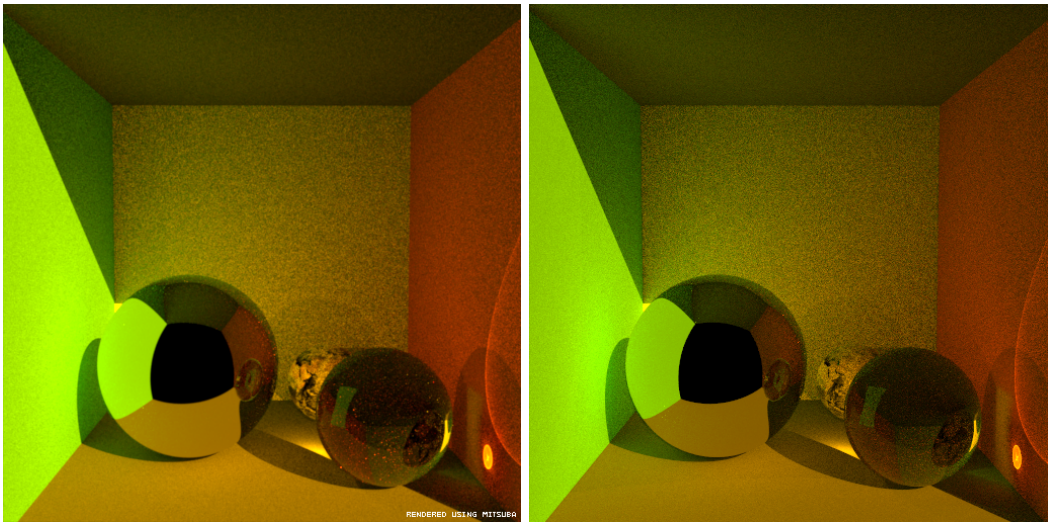
Obrázek 5.10: Složitá scéna obsahující refrakční povrchy diamantů, lesklou podlahu a složité textury.

ukázka, na obrázku 5.9, zobrazuje skleněný povrch, vzhledem k složitosti objektu se jedná o velmi komplexní výpočet refrakce (5.9a). Na posledním obrázku 5.9b je ukázán difúzně-spekulární povrch, který kombinuje vlastnosti difúzního Lambertovského modelu a zákonu odrazu světla, což dohromady vytváří kompletně lesklý povrch.

V další, již třetí, části testů, bylo vyzkoušeno vykreslování komplexních scén, které obsahují mnoho různorodých prvků, čímž se vyzkouší všechny vlastnosti implementace. Příkladem takové scény je obrázek 5.10, kde jsou obsaženy téměř všechny možné prvky – difúzní i spekulární odrazy, refrakce, kaustiky, několik druhů světelných zdrojů a jiné. Výsledný obraz obsahuje 6000 vzorků na pixel.

Porovnání s existujícím softwarem

Ačkoliv tato práce nepředstavuje novou metodu sama o sobě, ale pouze implementaci, bylo provedeno otestování navržené implementace s již existujícím moderním programem Mitsuba. Na následující stránce jsou ukázány výsledky z jedné scény, ale s různým nastavením kamery. První obrázek 5.11 ukazuje porovnání jednoduché cornell box scény, v tabulce 5.1 jsou pak shrnuty konkrétní čísla ukazující MSE (mean squared error, střední kvadratická odchylka), PSNR (peak to signal noise ratio, špičkový poměr signálu k šumu) a čas výpočtu. V tomto případě navržená implementace byla ve všech kriteriích lepší a výkonnější.



(a) Mitsuba (b) Navržená implementace

Obrázek 5.11: Cornell box scéna – oba obrázky vykresleny s 64 vzorky na pixel

Cornell box scéna

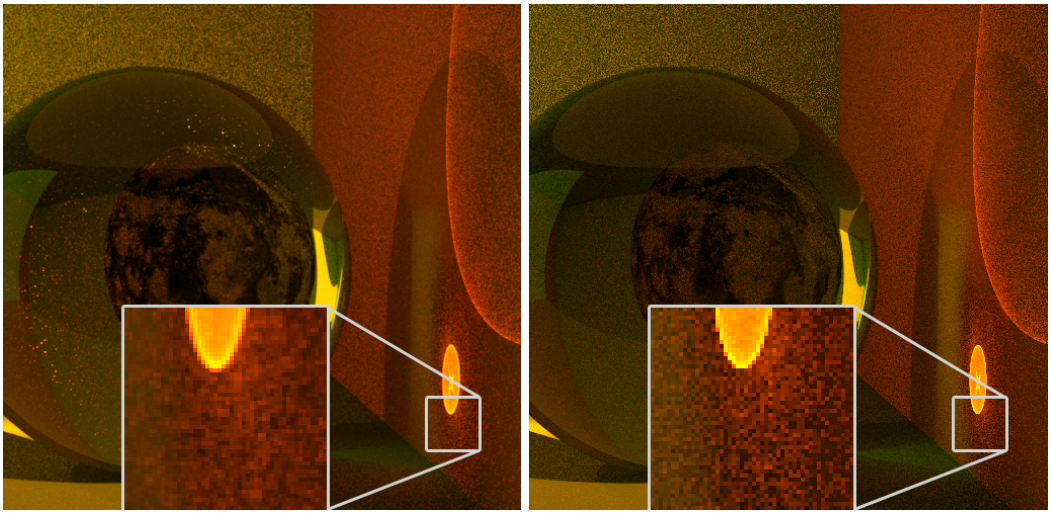
	MSE	PSNR	čas
Mitsuba	99.57	28.1 dB	32.4s
Implementace	64.15	30.1 dB	27.6s

Tabulka 5.1: Navržená implementace vs. Mitsuba – porovnání pomocí MSE a PSNR

Detail kaustiky

	MSE	PSNR	čas
Mitsuba	59.94	30.1 dB	58.1s
Implementace	80.27	29.5 dB	56.4s

Tabulka 5.2: Navržená implementace vs. Mitsuba – porovnání pomocí MSE a PSNR se zaměřením na kaustiky



(a) Mitsuba (b) Navržená implementace

Obrázek 5.12: Detail kaustiky – oba obrázky vykresleny se 128 vzorky na pixel

Druhý případ představuje stejnou scénu, ale s nastavením kamery tak, aby byla zaměřená na kaustiku způsobenou průchodem světla skrze skleněnou kouli, viz obrázek 5.12 a tabulka 5.2. Ačkoliv vizuálně se zdá obrázek z navržené implementace hezčí, podle metriky MSE se zdá být horší. Ovšem je nutno brát v potaz několik faktorů. Výsledný obraz z navržené implementace obsahuje mnoho šumu na zadní stěně scény, který má velmi vzdálené hodnoty od referenčního obrazu, naopak Mitsuba má spíše více šumu na skleněné kouli a méně na difúzní stěně. I takto relativně malý detail způsobí značnou akumulaci nepřesnosti při použití MSE. Z toho důvodu byla použita i metrika PSNR, která naopak říká, že oba obrázky jsou relativně podobně zašuměné oproti referenčnímu výstupu. Rovněž v tomto případě byl výpočetní čas téměř stejný jako u druhého rendereru, což je způsobeno zejména právě kaustikami, které jsou sice vypočítány trochu efektivněji, ale za to pomaleji než u Mitsuba.

Obrázek 5.13: Scéna zaměřená na vykreslování průhledných materiálů a kaustik (3000 vzorků na pixel, ~2.5h).

Další fáze testování byly spíše už zaměřené na vykreslování obrázků ve vysokém rozlišení zejména se složitou geometrií. Jedním z výsledků je obrázek 5.13 či 5.14, kde se nachází vícero objektů, které obsahují tisíce až milióny trojúhelníků.

Z hlediska funkčnosti je program schopný vykreslovat obrazy velmi složitých scén, což umožňuje nasazení v praxi, ačkoliv pro dosažení simulace naprosto reálných povrchů, je zapotřebí zpracovat i hrubé povrchy, které nejsou součástí implementace. Zejména z důvodu, že v přírodě se nenachází příliš mnoho jemných povrchů, ale převažují spíše hrubé. Pro simulaci hrubých povrchů lze využít tzv. microfacety s vhodnou distribucí (pro fyzikálně založené renderování se nejvíce využívá Beckmannova distribuce [2]). Rovněž by bylo nutno

Obrázek 5.14: Scéna obsahující velmi složité objekty i mimo kameru (2000 vzorků na pixel, ~6.5h).

aplikaci rozšířit o další fyzikálně založené typy BSDF, které se hojně využívají. Ovšem z hlediska funkčnosti se jedná pouze o konkrétní moduly, které lze přímo vložit do aplikace, bez nutnosti dalších úprav.

Kapitola 6

Závěr

Cílem této práce bylo navrhnout a implementovat fotorealistickou metodu sledování cest (path tracing), včetně možných zvolených rozšíření. Tento cíl byl splněn.

Prostudované materiály a náležitosti zobrazování 3D fotorealistických scén jsou popsány v první části práce. Rovněž jsou zde ukázány konkrétní metody a již existující řešení. V další části je shrnut současný stav a náležitosti zvolení metody path tracing, včetně diskuze možností implementace. Ve třetí části je ukázáno matematické odvození, které přímo definuje formální návrh výpočtů globálního osvětlení. V následující části je již popsán návrh a implementace definované metody, včetně demonstrace dosažených vlastností. Současně jsou ukázány dosažené výsledky na mnoha příkladech.

Výsledný algoritmus zvládá zpracování relativně složitých fotorealistických scén v definovaném formátu XML. Oproti často používanému modernímu programu Mitsuba, lze v mnoha případech dosáhnout přesnějších výsledků díky použití dvojité přesnosti a rovněž až o 15 % rychlejších výsledků, což ukazuje nespornou výhodu postupného vyhodnocování váhových funkcí během trasování, oproti většině publikovaných implementací, které provádí vyhodnocení právě až po dokončení sledování cesty. Část této práce byla rovněž publikována prostřednictvím studentské konference CESCG [18].

Prací v tomto odvětví počítačové grafiky jsem získal velmi cenné zkušenosti zejména v problematice implementace fotorealistických metod, ale rovněž i v řešení nekonečně rekurzivních rovnic pro simulování různých optických jevů, a problému s tím spojených.

Z hlediska možných rozšíření se nejvíce nabízí implementace dalších modulů BSDF, zejména pro hrubé, plastické či metalické povrchy. Druhým směrem, kde bych chtěl pokračovat, je možné využití GPU pro masovou akceleraci celého výpočtu. Ačkoliv tento algoritmus není úplně vhodný pro jednoduché zpracování pomocí GPU, dle mého názoru je to významná cesta pro výhled do budoucnosti. Další možností, kde by bylo možné zajistit celkové zlepšení algoritmu, je využití adaptivního vzorkování a speciálních filtrů pro redukci šumu.

Literatura

[1] Appel, A. Some Techniques for Shading Machine Renderings of Solids. In *Proceedings of the April 30–May 2, 1968, Spring Joint Computer Conference*. New York, NY, USA: ACM, 1968. S. 37–45. AFIPS '68 (Spring). Dostupné na: http://doi.acm.org/10.1145/1468075.1468082.

[2] Beckmann, P. a Spizzichino, A. *The scattering of electromagnetic waves from rough surfaces*. 1. vyd. Norwood: Artech House, 1987. ISBN 0-89006-238-2.

[3] Blinn, J. F. Models of Light Reflection for Computer Synthesized Pictures. *SIGGRAPH Comput. Graph.* červenec 1977, roč. 11, č. 2. S. 192–198. Dostupné na: http://doi.acm.org/10.1145/965141.563893. ISSN 0097-8930.

[4] Dutré, P. Global illumination compendium. *Computer Graphics, Department of Computer Science, Katholieke Universiteit Leuven* [online]. 2003. Dostupné na: https://people.cs.kuleuven.be/~philip.dutre/GI/.

[5] Dutré, P., Lafortune, E. P. a Willems, Y. D. Monte Carlo light tracing with direct computation of pixel intensities. In *3rd International Conference on Computational Graphics and Visualisation Techniques*. Alvor, Portugal: [b.n.], December 1993. S. 128–137.

[6] Georgiev, I., Křivánek, J., Davidovič, T. et al. Light Transport Simulation with Vertex Connection and Merging. *ACM Trans. Graph.* Listopad 2012, roč. 31, č. 6. S. 192:1–192:10. Dostupné na: http://doi.acm.org/10.1145/2366145.2366211. ISSN 0730-0301.

[7] Jensen, H. W. Global Illumination Using Photon Maps. *Proceedings of the Eurographics Workshop on Rendering Techniques '96* [online]. 1996 [cit. 2018-03-14]. S. 10. Dostupné na: http://dl.acm.org/citation.cfm?id=275458.275461. ISSN 3-211-82883-4.

[8] Kajiya, J. T. The Rendering Equation. In *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques*. New York, NY, USA: ACM, 1986. S. 143–150. SIGGRAPH '86. Dostupné na: http://doi.acm.org/10.1145/15922.15902. ISBN 0-89791-196-2.

[9] Kulling, K. *Assimp* [online]. 2006 [cit. 2018-04-30]. Dostupné na: http://www.assimp.org/.

[10] Lafortune, E. P. a Willems, Y. D. Bi-directional path tracing. In *Proceedings of Third International Conference on Computational Graphics and Visualization*

Techniques (Compugraphics '93). Alvor, Portugal: [b.n.], December 1993. S. 145–153. Dostupné na: http://graphics.cs.kuleuven.be/publications/BDPT/index.html.

[11] Phong, B. T. Illumination for Computer Generated Pictures. *Communications of the ACM*. červen 1975, roč. 18, č. 6. S. 311–317. Dostupné na: http://doi.acm.org.ezproxy.lib.vutbr.cz/10.1145/360825.360839. ISSN 0001-0782.

[12] Roth, S. D. Ray casting for modeling solids. *Computer Graphics and Image Processing*. 1982, roč. 18, č. 2. S. 109 – 144. Dostupné na: http://www.sciencedirect.com/science/article/pii/0146664X82901691. ISSN 0146-664X.

[13] Rémi, B. Monte Carlo methods. *EPJ Web of Conferences*. 2013, roč. 55, č. 02002. S. 21. Dostupné na: https://bit.ly/2qtajAT. ISSN 2100-014X.

[14] Shirley, P. a Morley, R. K. *Realistic Ray Tracing*. 2. vyd. Natick, MA, USA: A. K. Peters, Ltd., 2003. ISBN 1568811985.

[15] Shirley, P., Wang, C. a Zimmerman, K. Monte Carlo Techniques for Direct Lighting Calculations. *ACM Trans. Graph.* Leden 1996, roč. 15, č. 1. S. 1–36. Dostupné na: http://doi.acm.org/10.1145/226150.226151. ISSN 0730-0301.

[16] Veach, E. *Robust Monte Carlo Methods for Light Transport Simulation*. Stanford, CA, USA, 1998. Disertační práce. AAI9837162. ISBN 0-591-90780-1.

[17] Veach, E. a Guibas, L. J. Optimally Combining Sampling Techniques for Monte Carlo Rendering. In *Proceedings of the 22Nd Annual Conference on Computer Graphics and Interactive Techniques*. New York, NY, USA: ACM, 1995. S. 419–428. SIGGRAPH '95. Dostupné na: http://doi.acm.org/10.1145/218380.218498. ISBN 0-89791-701-4.

[18] Vlnas, M. Bidirectional Path Tracing. In *Proceedings of the 22nd Central European Seminar on Computer Graphics*. 1. vyd. Faoritenstraße 9-11/193, 1040 Vienna: TU Wien, Institute of Visual Computing & Human Centered Technology, Duben 2018. S. 45–52. Dostupné na: https://cescg.org/cescg_submission/bidirectional-path-tracing/.

[19] Whitted, T. An Improved Illumination Model for Shaded Display. *Commun. ACM*. červen 1980, roč. 23, č. 6. S. 343–349. Dostupné na: http://doi.acm.org/10.1145/358876.358882. ISSN 0001-0782.

[20] Žára, J., Felkel, P., Sochor, J. et al. *Moderní počítačová grafika*. 2. vyd. Brno: Computer Press, 2004. ISBN 80-251-0454-0.

Příloha A

XML formát pro popis scény

Celý popis XML scény je nutno zabalit do značky `<scene>... popis scény ...</scene>`, aby bylo možno z hlediska programové implementace rozpoznat jednotlivé elementy.

Kamera

Definice kamery je povinným prvkem v popisu scény. Základní atributy, které lze nastavit jsou popsány v tabulce A.1. Počátek, směr a osa směřující nahoru jsou povinné prvky. Ostatní atributy jsou volitelné, pokud nebudou nastaveny, použije se výchozí hodnota. Pro jednodušší orientaci jsou povinné atributy vyznačeny ve všech následujících tabulkách tučným písmem.

Tag	**Atribut**	**Hodnota**	**Popis**
camera	**origin**	vektor	Počáteční pozice kamery (XYZ)
	direction	vektor	Směr pohledu kamery (XYZ)
	up	vektor	Vektor určující osu směřující nahoru (XYZ)
	width	unsigned int	Šířka vykreslované scény (výchozí 512px)
	height	unsigned int	Výška vykreslované scény (výchozí 512px)
	fov	double	Horizontální zorné pole kamery (výchozí 60)

Tabulka A.1: XML popis scénové kamery

Ukázka:

```
<camera origin="-1.7, 0.1, -7" direction="-0.70, -0.3, 1.0" up="0, 1, 0"
width="512" height="512" fov="45.0" />

<camera origin="-1.7, 0.1, -7" direction="-0.70, -0.3, 1.0" up="0, 1, 0"/>
```

Geometrie

Základní geometrické prvky, které program podporuje lze popsat pomocí XML ve formátu uvedeném v tabulce A.2.

Typ	**Atribut**	**Hodnota**	**Popis**
sphere	**radius**	double	Poloměr koule
	x	double	Souřadnice na ose X (výchozí 0)
	y	double	Souřadnice osy Y (výchozí 0)
	z	double	Souřadnice na ose Z (výchozí 0)
triangle	**a**	vektor	Bod XYZ trojúhelníku
	b	vektor	
	c	vektor	
mesh	**src**	string	Cesta ke zdrojovému souboru objektu

Tabulka A.2: XML popis geometrických objektů

Ukázka:

```
<shape type="sphere" radius="2.3" x="1.0" y="-0.5" z="-4.0">
...
</shape>

<shape type="triangle" a="-0.3, 1.2, 4.7" b="-0.3, 3.4, 2.1" c="1.2, 3.4, 2.1">
...
</shape>

<shape type="mesh" src="cesta/ksouboru/objekt.obj">
...
</shape>
```

Každý geometrický prvek (kromě mesh) musí nutně mít přiřazený povrch, který definuje jeho vlastnosti pro vyhodnocení BSDF.

Pro objekt typu `mesh` lze nastavit pomocí BSDF pouze globální materiál pro celý objekt, nelze definovat rozdílné materiály pro různé kusy objektu. Naopak se očekává, že v adresáři kde se nachází odkaz na objekt se bude rovněž nacházet soubor popisující jeho povrch (např. soubor .mtl).

Povrch

Pro popis povrchu je určen tag `bsdf`. Definovat povrch je povoleno pouze uvnitř elementu popisujícího geometrii (např. trojúhelník). Lze vybrat libovolnou BSDF, která je implementována v programu, viz tabulku A.3.

Typ	**Atribut**	**Hodnota**	**Popis**
diffuse	**texture**	vektor/string	Difúzní textura
mirror	**texture**	vektor	Vektorová hodnota odrazu
glossy	**diffuse**	vektor/string	Difúzní textura
	specular	vektor	Vektorová hodnota odrazu
phong	**diffuse**	vektor/string	Difúzní textura
	specular	vektor	Textura zrcadlového odrazu
	exp	double	Hodnota phongova koeficientu
smoothdielectric	**texture**	vektor	Hodnota spekulární textury
	ior	double	Index refrakce

Tabulka A.3: XML popis pro definici BSDF

Ukázka:

```
<shape type="sphere" radius="2.3" x="1.0" y="-0.5" z="-4.0">
    <bsdf type="phong" diffuse="0.5, 0.3, 0.1"
          specular="0.5, 0.5, 0.5" exp="20"/>
</shape>

<shape type="sphere" radius="2.3" x="1.0" y="-0.5" z="-4.0">
    <bsdf type="diffuse" texture="textures/texture.png"/>
</shape>

<shape type="triangle" a="-0.3, 1.2, 4.7" b="-0.3, 3.4, 2.1" c="1.2, 3.4, 2.1">
    <bsdf type="smoothdielectric" texture="1,1,1" ior="1.6"/>
</shape>
```

Za zmínku stojí fakt, že při definici BSDF pro trojúhelník či mesh, není vhodné volit jinou texturu než vektorovou, protože nelze definovat přiřazení textur k jednotlivým vertexům. K tomu se běžně v praxi právě využívají implicitní souřadnice textur v geometrické definici objektu typu mesh, a případně mapování přes .mtl soubor.

Světelné zdroje

Světelný zdroj je povinné definovat pro každou scénu. Lze vybrat libovolný ze zdrojů v tabulce A.4. Všechny zdroje mají sdílené dva atributy – color a intensity. Oba tyto atributy je povinné definovat, protože určují primární charakteristiku světla. Pouze `environment` světlo nepotřebuje atribut `color`, protože hodnotu své radiance určuje na základě poskytnuté textury.

Typ	**Atribut**	**Hodnota**	**Popis**
area, point, sun, conical	**color**	vektor	Barva světla
	intensity	double	Intenzita
area	**a**	vektor	Souřadnice bodu
	b		
	c		
point	**position**	vektor	Pozice světla
sun	**direction**	vektor	Směr světla
environment	**texture**	vektor/string	Světelná textura
conical	**position**	vektor	Počátek světla
	direction	vektor	Světelný směr
	angle	double	Úhel rozevření kuželu

Tabulka A.4: XML popis pro definici světelných zdrojů

Ukázka:

```
<light type="area" color="1,1,1" intensity="20"
       a="-2, 19.8, 3" b="2, 19.8, 3" c="2, 19.8, 6" />

<light type="sun" color="0.5, 0.2, 0.0" intensity="8"
       direction="0.6, -0.7, 1.0" />

<light type="point" color="1,1,1" intensity="20" position="2, 19.8, 6" />

<light type="environment" intensity="1" texture="textures/kitchen_env.jpg" />

<light type="conical" color="1,1,1" intensity="20" position="0,10,0"
       direction="0,-1,0" angle="25.0" />
```

Transformace

V neposlední řadě lze na geometrické tvary typu `mesh` aplikovat základní 3D transformace, jako je posun, zvětšení/zmenšení či rotace. Transformace nelze použít na samotnou kouli či trojúhelník, protože ohraničující strukturu těchto elementů lze přímo nastavit, proto tedy postrádá smysl podporovat transformace pro tyto objekty. Popis formátu XML je ukázán v tabulce A.5 a rovněž v následující ukázce.

Typ	**Atribut**	**Hodnota**	**Popis**
translate	**value**	vektor	Vektor posunu
scale		double	Koeficient škálování
rotate_x		double	Rotace kolem osy (v radiánech)
rotate_y		double	
rotate_z		double	

Tabulka A.5: XML popis pro definici základních transformací

Ukázka:

```
<shape type="mesh" src="cesta/ksouboru/objekt.obj">
    <transformations>
        <transform type="scale" value="0.1" />
        <transform type="rotate_x" value="0.52" />
        <transform type="rotate_z" value="1.57" />
        <transform type="translate" value="0,5,0" />
    </transformations>
</shape>
```

Jelikož lze přiřadit více než jednu transformaci, je nutno všechny ohraničit značkou `transformations`. Důvod pro toto ohraničení je stejný jako u celého popisu scény, tzn. kvůli programové implementaci zpracování XML formátu.

Příloha B

Obsah přiloženého CD

Přiložené CD obsahuje všechny zdrojové kódy ve složce `src/`, již zkompilované binární soubory ve složce `build/` pod názvem `tracer`. Rovněž jsou přiloženy objekty použité během testování a v zabudovaných scénách. Dále ve složce `doc/` je automaticky generovaná dokumentace pomocí systému Doxygen. Další výstupy z programu, kromě těch co již jsou v práci, lze nalézt ve složce `build/results/`. V neposlední řadě je na CD několik dalších souborů, které jsou potřebné pro úspěšný překlad a sestavení a rovněž základní README s návodem k instalaci a potřebnými závislostmi.