



**VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ**  
BRNO UNIVERSITY OF TECHNOLOGY



**FAKULTA INFORMAČNÍCH TECHNOLOGIÍ**  
**ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ**  
FACULTY OF INFORMATION TECHNOLOGY  
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

## **FOTOREALISTICKÉ ZOBRAZOVÁNÍ**

PHOTOREALISTIC RENDERING

### **BAKALÁŘSKÁ PRÁCE**

BACHELOR'S THESIS

### **AUTOR PRÁCE**

AUTHOR

PAVEL MELCER

### **VEDOUCÍ PRÁCE**

SUPERVISOR

Prof. Dr. Ing PAVEL ZEMČÍK

BRNO 2016

**Vysoké učení technické v Brně - Fakulta informačních technologií**

Ústav počítačové grafiky a multimédií

Akademický rok 2015/2016

**Zadání bakalářské práce**

Řešitel: **Melcer Pavel**

Obor: Informační technologie

Téma: **Fotorealistické zobrazování**  
**Photorealistic Rendering**

Kategorie: Počítačová grafika

**Pokyny:**

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

**Literatura:**

- Dle pokynů vedoucího

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).

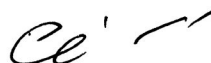
Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo 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 2015

Datum odevzdání: 18. května 2016

**VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ**  
Fakulta informačních technologií  
Ústav počítačové grafiky a multimédií  
602 00 Brno, Božetěchova 2



---

doc. Dr. Ing. Jan Černocký  
vedoucí ústavu

## Abstrakt

Tato bakalářská práce se zabývá problematikou realistického zobrazování 3D scén, kde jsou popsány postupy vedoucí k fotorealistickému obrazu. Nejprve jsou uvedeny způsoby definice předmětů, jak je scéna vykreslena na displej, typy světelných zdrojů a povrchové charakteristiky materiálů. V práci se dále nachází popis lokálního osvětlovacího modelu, optických jevů a metod používaných k realistickému zobrazení (raytraycing, pathtraycing, radiozita a photon mapping). Další část zmiňuje důvod výběru téma a popis implementace metody raytraycing a photon mapping.

## Abstract

This bachelor thesis deals with the realistic rendering of 3D scenes, which describes the procedures leading to the photorealistic image. First are mentioned ways to definition objects, how the scene is rendered on screen, the types of light source and surface characteristics of materials. The work is also a description of the local shading model, optical phenomena and the methods used to realistically display (raytraycing, pathtraycing, radiosity and photon mapping). Next part mentions the reason for choosing the topic and description of implementation methods raytraycing and photon mapping.

## Klíčová slova

sledování fotonu, sledování paprsku, sledování cesty, radiozita, projekce, dvousměrná odrazová distribuční funkce, zobrazovací rovnice, phongův osvětlovací model, realistický obraz

## Keywords

photon mapping, raytraycing, pathtraycing, radiosity, projections, bidirectional reflectance distribution function, rendering equation, phong reflection model, realistic image

## Citace

MELCER, Pavel. *Fotorealistické zobrazování*. Brno, 2016. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Zemčík Pavel.

# Fotorealistické zobrazování

## Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Prof. Dr. Ing. Pavla Zemčíka. Další informace mi poskytl pan Ing. Tomáš Lysek. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....

Pavel Melcer  
18. května 2016

## Poděkování

Děkuji vedoucímu práce Prof. Dr. Ing. Pavlovi Zemčíkovi a Ing. Tomáši Lyskovi za odborné rady a materiály, které vedly k pochopení principů počítačové grafiky.

© Pavel Melcer, 2016.

*Tato práce vznikla jako školní dílo na Vysokém učení technickém v Brně, Fakultě informačních technologií. Práce je chráněna autorským zákonem a její užití bez udělení oprávnění autorem je nezákonné, s výjimkou zákonem definovaných případů.*

# Obsah

<b>1 Úvod</b>	<b>2</b>
<b>2 Zobrazení fotorealistických scén</b>	<b>3</b>
2.1 Scéna . . . . .	3
2.2 Průsečík paprsku s polygonem . . . . .	5
2.3 Projekce scény . . . . .	6
2.4 Osvětlení scény . . . . .	6
2.5 Dvousměrná odrazová distribuční funkce . . . . .	9
2.6 Osvětlovací modely . . . . .	10
2.7 Existující programy . . . . .	12
<b>3 Globální zobrazovací metody</b>	<b>13</b>
3.1 Optické jevy . . . . .	13
3.2 Zobrazovací rovnice . . . . .	15
3.3 Sledování paprsku . . . . .	16
3.4 Sledování cesty . . . . .	18
3.5 Radiozita . . . . .	18
3.6 Sledování fotonů . . . . .	19
<b>4 Důvod a plán práce</b>	<b>22</b>
<b>5 Popis práce</b>	<b>23</b>
5.1 Blokové a funkční schéma programu . . . . .	23
5.2 Konfigurace programu . . . . .	24
5.3 Vytvoření scény . . . . .	24
5.4 Načítání scény . . . . .	24
5.5 Promítací průmětna . . . . .	25
5.6 Sledování paprsků . . . . .	26
5.7 Progresivní sledování fotonů . . . . .	29
<b>6 Závěr</b>	<b>32</b>
<b>Literatura</b>	<b>33</b>
<b>Přílohy</b>	<b>35</b>
Seznam příloh . . . . .	36
<b>A Obsah CD</b>	<b>37</b>

# Kapitola 1

## Úvod

Během rozvoje počítačů v 60. letech docházelo k navyšování výpočetního výkonu a ke vzniku nového odvětví informatiky, dnes nazývaného počítačová grafika. Vývoj zobrazení grafiky pomocí počítačů mělo velký přínos, který je dnes vidět ve všech aspektech současné doby. Především je prokázáno, že člověk získává nejvíce informací pomocí vizuálního dojmu. Dále počítačová grafika zjednodušuje tvůrčí práci v mnoha odvětvích lidských činností. Největší význam má v architektuře, strojírenství, filmovém a herním průmyslu.

Fotorealistické zobrazení je již od počátku hlavním cílem počítačové grafiky. V průběhu let docházelo k významným objevům, které vždy posouvali realističnost. Fyzikální simulace šíření světla je náročná na výpočetní výkon. A proto až s příchodem výkonných grafických karet v posledních letech nastává největší rozvoj. Například je výrazný rozdíl v grafice mezi hrami vytvořených pro konzole starých 10 let (Playstation 2 nebo Xbox360) a novinkami pro konzole (Playstation 3 nebo Xbox one). Je patrné, že s rostoucím výkonem bude vždy docházet k výraznému posunu v jejich realističnosti.

Hlavním cílem mé bakalářské práce je seznámit se s tvorbou 3D scény a základními principy počítačové grafiky. Moje práce je primárně zaměřena na algoritmy k vykreslení scén, které působí realistickým dojmem.

Následující text je členěn do čtyř kapitol. V kapitole 2 a 3 budou popsány nastudované metody a principy. Nejprve jsou zmíněny způsoby definice těles a scén, poté principy snímání scény vedoucí k vykreslení na obrazovku. Další část druhé kapitoly je věnována světelným zdrojům a povrchovým vlastnostem předmětů. Ve třetí kapitole jsou uvedeny optické jevy a popsány globální zobrazovací metody simulující šíření světla.

V kapitole 4 jsou uvedeny důvody k výběru tématu této bakalářské práce, její průběh a následný plán realizace. Kapitola 5 je věnována návrhu a popisu implementace aplikace.

V závěru jsou zmíněny přínosy, které mě práce přinesla.

## Kapitola 2

# Zobrazení fotorealistických scén

Od 60. let 20 století se programátoři ve spojených státech amerických snaží za pomoci počítačů zobrazit svět kolem nás. Největší přínos v nově vznikajícím odvětví přinesla univerzita v Utahu, kde roku 1968 David Evans založil projekt pro rozvoj a zkoumání počítačové grafiky. Během několika let univerzita dosáhla několika objevů, které jsou i dnes používány jako základní principy. Na univerzitě také vznikla slavná konvice z Utahu, která je do současnosti využívána jako testovací model.



Obrázek 2.1: Konvice z Utahu[18].

### 2.1 Scéna

Pokud chtějí například architekti ukázat vyprojektovaný dům nebo strojaři zobrazit navrženou součástku je nutné definovat svět kolem nás v digitální podobě. K abstrakci světa slouží 3D scény, které mohou být například malý fotoateliér s plátnem nebo rozsáhlé scény reprezentující modely měst.

Scéna obsahuje definici tvaru a povrchové vlastnosti těles, jejich umístění vzhledem k ostatním objektům, charakteristiku a pozici světelných zdrojů nebo kamer. V následující části budou popsány způsoby popisu geometrie objektů. Na závěr podkapitoly je zmíněn souborový formát k uložení definice scény, který je podporován existujícími programy.

## Konstruktivní geometrie - CSG

Objekt je definován pomocí stromu složeného z 3D primitiv (válec, kvádr či koule) a transformacemi nad nimi. CSG je uplatněno k modelování předmětů ve strojírenském inženýrství nebo architektury - CAD/CAM.

## Šablonování

Předmět je vytvořen pohybem 2D profilu po křivce. Šablonováním jde jednoduše a rychle vytvořit například sklenici.

## Dekompoziční modely

Voxel je elementární objemová jednotka (krychle nebo hranol), která je uložena v 3D poli, oktalovém stromu nebo kombinací obou struktur. Složením voxelů vzniká objekt znázorněný na obr. 2.2. Takto vytvořený model nese informaci o vnitřní struktuře těles. Nevýhodou jsou vyšší nároky na paměť. Využití je především v medicíně k reprezentaci výsledků z CT nebo magnetické rezonance.



Obrázek 2.2: Dekompoziční model[13].

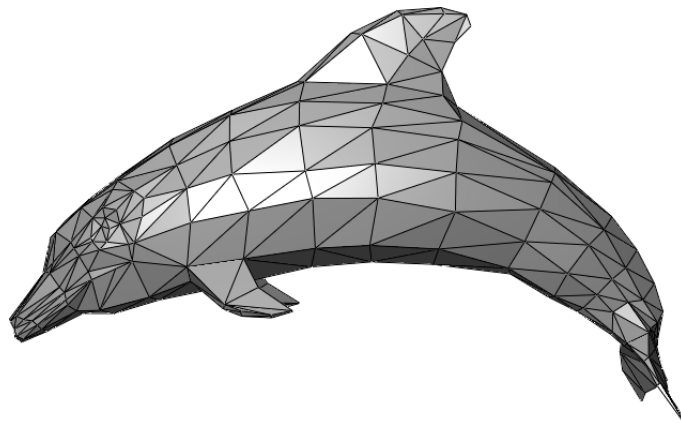
## Polygonální modely

Objekt je jednoznačně definován pomocí vrcholů, hran a stěn, které se nazývají polygony. Může se jednat o trojúhelníky nebo čtyřúhelníky. Dnešní grafické karty mají zabudovanou hardwarovou podporu pro tento typ reprezentace. Výhodou jsou malé nároky na paměť, ale někdy za cenu ztráty detailnější informace o vyhlazení tvaru objektů. Definice objektu pomocí polygonů je v současnosti nejpoužívanější způsob.

## Collada

Anglicky **C**ollaborative **D**esign **A**ctivity je otevřený souborový formát, který je podmnoužinou jazyka XML. Byl definovaný firmou Sony Computer Entertainment pro Playstation





Obrázek 2.3: Model delfína složeného z polygonů[17].

3. Slouží k uložení scény a modelů, které jsou reprezentovány jako trojúhelníková síť. Je podporovaný v grafických programech Maya, 3DS Max, Blender nebo Cinema4D.

## 2.2 Průsečík paprsku s polygonem

Jak bylo zmíněno výše v kapitole 2.1, obvykle se v počítačové grafice používá model sestaven ze sítě mnoha trojúhelníků. Během simulace je nejčastější operací nalezení průsečíku vyslaného paprsku s polygony. Paprsek je popsán rovnicí 2.1, kde  $o$  je počáteční místo odkud byl vyslán paprsek,  $\vec{d}$  normalizovaný směr a  $t$  určuje vzdálenost od počátku.

Trojúhelník je definován třemi body A, B a C reprezentující rovinu, kterou lze popsat barycentrickými souřadnicemi v rovnici 2.2. Po spojení a úpravě rovnic 2.1 a 2.2 získáváme vztah 2.3 pro nalezení průsečíku. Aby ležel průsečík uvnitř trojúhelníku musí platit podmínky 2.4.

$$p(t) = o + t\vec{d} \quad (2.1)$$

$$p(\beta, \gamma) = A + \beta(B - A) + \gamma(C - A) \quad (2.2)$$

$$o + t\vec{d} = A + \beta(B - A) + \gamma(C - A) \quad (2.3)$$

$$\begin{aligned} 0 &\leq t \leq t_{max} \\ \beta + \gamma &< 1 \\ 0 &< \beta \\ 0 &< \gamma \end{aligned} \quad (2.4)$$

K určení vzdálenosti  $t$  a výpočtu  $\beta$  a  $\gamma$ , které slouží k interpolaci normál nebo k získání souřadnice vzorku textury, je potřeba rovnici 2.3 rozepsat pro všechny dimenze na výslednou soustavu rovnic 2.5, která je pomocí cramerova pravidla vypočtena.

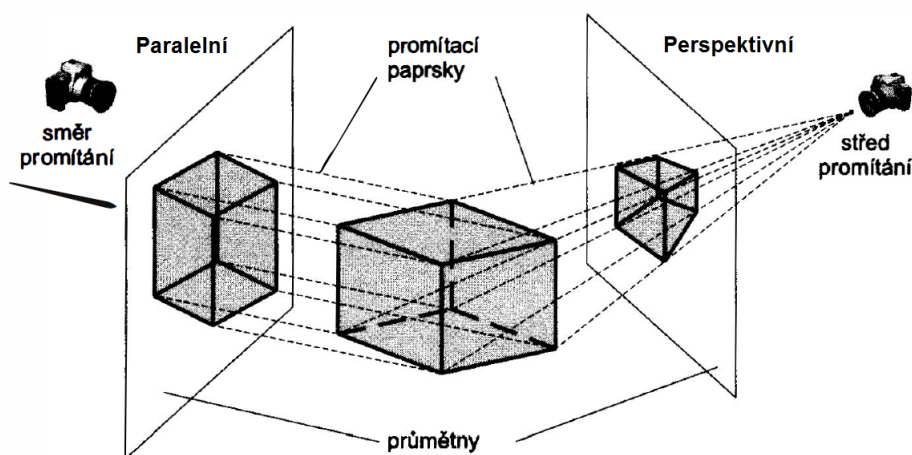
$$\begin{aligned}
o_x + td_x &= A_x + \beta(B_x - A_x) + \gamma(C_x - A_x) \\
o_y + td_y &= A_y + \beta(B_y - A_y) + \gamma(C_y - A_y) \\
o_z + td_z &= A_z + \beta(B_z - A_z) + \gamma(C_z - A_z)
\end{aligned}
\tag{2.5}$$

## 2.3 Projekce scény

Současné displeje zobrazují pouze dvou dimenzionální prostor, ale scéna je definovaná ve třech rozměrech. K vykreslení na obrazovku je tedy nutné scénu transformovat pomocí operace zvané promítání neboli renderování. Během promítání dochází ke ztrátě prostorové informace a ke zkreslení skutečného tvaru objektu, proto se podle potřeby využívají různé způsoby promítání. Studium promítacích metod se zabývá deskriptivní geometrie.

Princip znázorněný na obrázku 2.4 je vždy stejný. Scéna je snímána pomocí kamery, které mají definovaný směr pohledu a pozici ve scéně. Od kamery jsou vysílány promítací paprsky procházející přes plochu v prostoru zvanou průmětna. Paprsky dopadají na tělesa ve scéně, kde je nalezen průsečík. V tomto bodě je vyhodnocena barva, která je zpětně promítnuta na průmětnu čímž vzniká dvou dimenzionální obraz.

Nejpoužívanějšími promítacími metodami v počítačové grafice je paralelní a perspektivní projekce. Rozdíl spočívá ve směru paprsků. V případě paralelní projekce mají všechny paprsky stejný směr, zatímco perspektivní paprsky vycházejí pouze z jednoho bodu s různými směry. Paralelní promítání je vhodné v oblastech, kde je potřeba zachovávat přesné proporce a úhly například ve strojírenství.

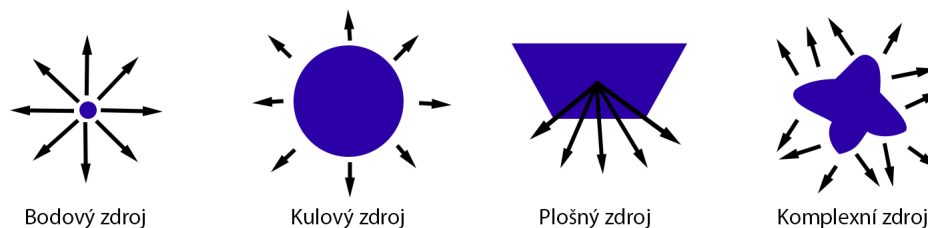


Obrázek 2.4: Proces renderování[2].

## 2.4 Osvětlení scény

V následující části jsou popsány charakteristiky různých typů světél vyobrazených na obrázku 2.5 a jaké vlastnosti mají konkrétní povrchy materiálů. Dále je uvedena formální notace transportu světla v prostoru a jeho odrazy. V závěru části je uvedena funkce BRDF popisující odrazové vlastnosti a lokální osvětlovací model těles.

## Typy zdroje světla



Obrázek 2.5: Typy světelných zdrojů.

### Bodový zdroj

Jedná se o nejjednodušší typ světla. Je jednoduchý na implementaci, ale v reálném světě těžko k nalezení. Světelné fotony vznikají v jednom bodě a šíří se v prostoru ve všech směrech rovnoměrně.

### Kulový zdroj

Je abstrakcí slunce nebo žárovky, kde fotony vznikají na povrchu koule s daným poloměrem. Jsou šířeny stejně jako u bodového zdroje ve všech směrech rovnoměrně. Výhodou je možnost vytvoření měkkých stínů.

### Plošný zdroj

Modeluje okna nebo plošná světla. Světlo vniká po celé ploše mnohoúhelníku a je šířeno do polokoule obklopující plochu. Vytváří také měkké stíny.

### Obecný zdroj

Jedná se o nejsložitější typ, kdy tvar je popsán geometrií a vyzařovací funkcí pro každý bod.

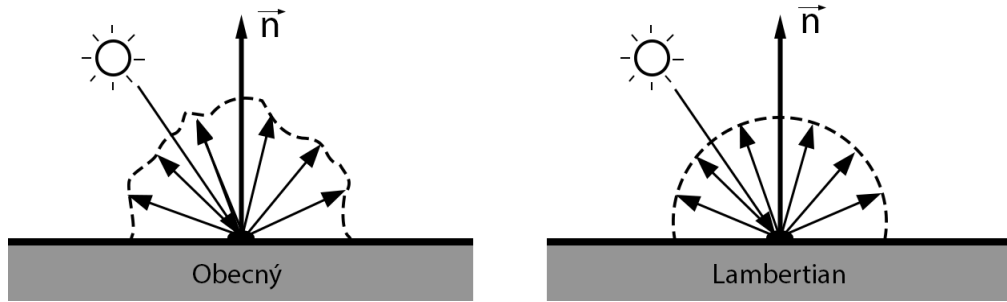
## Povrchy materiálů a jejich odrazy

### Difúzní povrch

Simuluje matné předměty jako plast, papír nebo zdi. Energie se po dopadu na jejich povrch z větší části pohltí. Pro difúzní odraz platí, že vstupní radianci rozptýluje rovnoměrně do všech směrů. Přináší informaci o barvě povrchu. Ideální difúzní povrch se nazývá Lambertovský povrch a je matematickou abstrakcí.

### Zrcadlové materiály

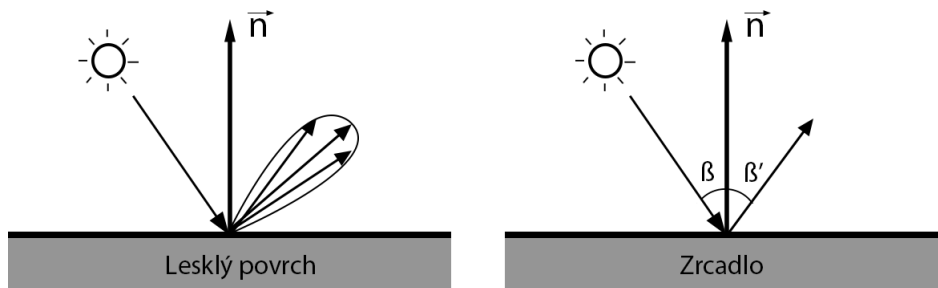
Po dopadu na hladký povrch například kov, dochází pouze k odrazu světla. Odražený směr je vypočten podle zákona odrazu. Na obrázku 2.7 je znázorněn odražený směr. V případě



Obrázek 2.6: Difuzní povrch materiálu.

ideálního odrazu docházející u zrcadel platí vztah 2.9, kde  $\vec{\omega}_{in}$  je směr příchozího paprsku a  $\vec{n}$  je normála v povrchu v bodě dopadu.

$$\vec{\omega}_{out} = 2(\vec{\omega}_{in} \cdot \vec{n})\vec{n} - \vec{\omega}_{in} \quad (2.6)$$



Obrázek 2.7: Zrcadlový povrch materiálu.

## Průhledné materiály

V materiálech typu sklo nebo voda dochází k lomu na rozhraní dvou prostředí s různou optickou hustotou. Paprsek je po dopadu rozdělen na dvě části odražený a lomený. Úhel lomu se vypočítává ze Snellova zákona lomu 2.7, kde  $\eta_t$  a  $\eta_i$  udává index lomu prostředí.

$$\frac{\sin \theta_i}{\sin \theta_t} = \frac{\eta_t}{\eta_i} \quad (2.7)$$

## Transport světla

Paul Heckbert [4] v roce 1990 formuloval notaci popisující transport a odraz světla využívající symboly a regulární výrazy

### Symboly

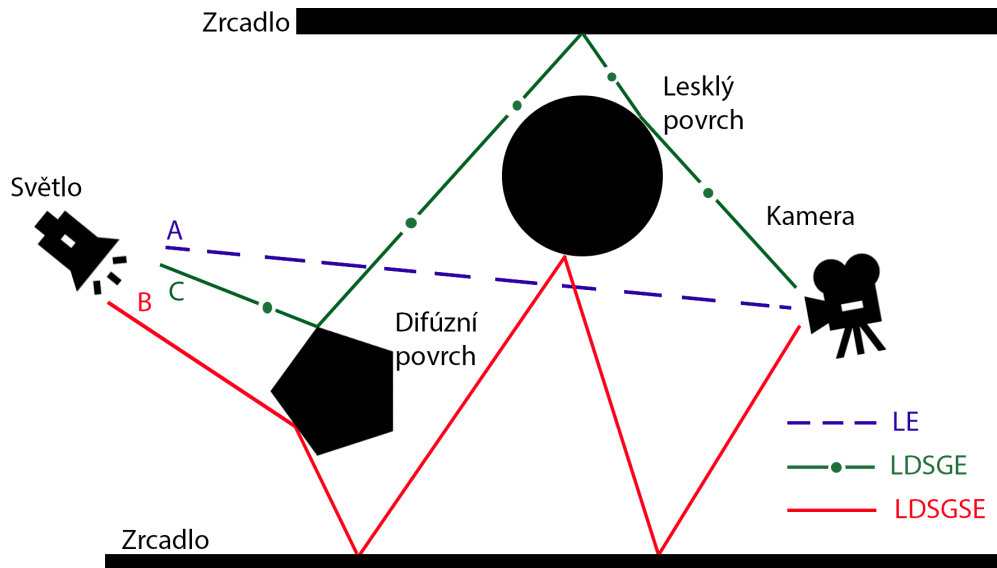
- L – světelný zdroj (light)
- D – odraz od difúzního povrchu (diffuse)
- S – odraz od zrcadlového povrchu (specular)

- G – odraz od lesklého povrchu (glossy)
- E – dopad paprsku do oka nebo kamery (eye)

### Regulární výrazy

- (k)? - žádný nebo jeden odraz
- (k)+ jeden nebo více odrazů
- (k)\* žádný nebo více odrazů
- (k|q) cesta k nebo q

Na obrázku 2.8 je znázorněno několik transportů světla. Paprsek A směřuje přímo ze světla do kamery, má tedy označení LE. Paprsek C jednou zasáhne difúzní objekt a lesklý povrch, zároveň i dvakrát dojde ke srážce se zrcadlem. Výsledné označení je LDSGSE.



Obrázek 2.8: Heckbertova notace transportu.

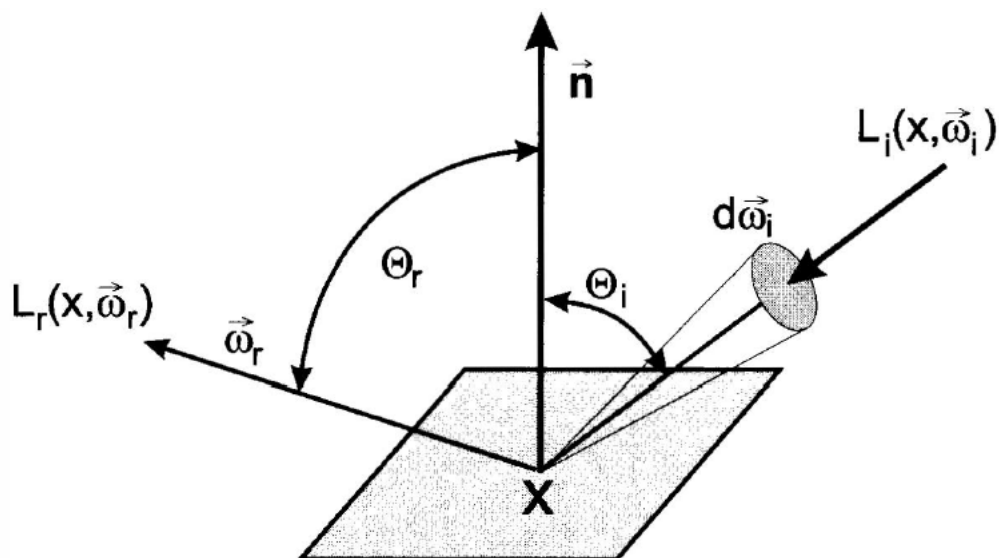
## 2.5 Dvousměrná odrazová distribuční funkce

Zkráceně BRDF z anglického jazyka Bidirectional Reflectance Distribution Function popsal Fred Nicodemus [9]. Funkce má označení  $f_r(x, \vec{\omega}_r, \vec{\omega}_i)$  a charakterizuje odrazové vlastnosti materiálu v určitém bodě označeném  $x$ . Světlo dopadá ze směru  $\vec{\omega}_i$  a odráží ve směru  $\vec{\omega}_r$ . BRDF definuje poměr 2.8 odražené radiance v bodě  $x$  označené jako  $dL_r(x, \vec{\omega}_r)$  ke vstupní diferenciální radianci  $dL_i(x, \vec{\omega}_i)$  promítnuté na kolmou plochu. Čím víc je dopad kolmější tím je přijatý výkon větší.

$$f_r(x, \vec{\omega}_r, \vec{\omega}_i) = \frac{dL_r(x, \vec{\omega}_r)}{dL_i(x, \vec{\omega}_i)(\vec{\omega}_i \cdot \vec{n})d\vec{\omega}_i} \quad (2.8)$$

## Vlastnosti BRDF

- Helmholtzův princip reciprocity - hodnota BRDF zůstává stejná i po záměně směru dopadu a odrazu tedy  $f_r(x, \vec{\omega}_r, \vec{\omega}_i) = f_r(x, \vec{\omega}_i, \vec{\omega}_r)$ . Umožňuje použití dvousměrových globálních metod.
- Pozitivita BRDF -  $f_r(x, \vec{\omega}_r, \vec{\omega}_i) \geq 0$
- Anizotropie - Obecná vlastnost materiálu a proto i BRDF je anizotropní. Jedná se o vlastnost, kdy odraz světla nezáleží jen na vstupním, výstupním směru a bodu  $x$ , ale také na natočení povrchu kolem normálového vektoru k povrchu. Například kompaktní disk, pokud je z nějakého směru nasvícen, po otočení kolem normálového vektoru se mění barva odraženého světla. Správně by funkce měla mít tvar  $f_r(x, \vec{\omega}_r, \phi, \vec{\omega}_i)$  kde  $\phi$  určuje úhel natočení bodu. Pro zobrazování se ovšem tato vlastnost neuvažuje.
- Linearita - hodnota BRDF pro vstupní úhel  $\vec{\omega}_i$  nezávisí na hodnotách BRDF pro jiné vstupní úhly. Paprsek ze směru  $\vec{\omega}_i$  je tedy vyzářen bez ohledu na ostatní úhly. Zajišťuje funkci lokálních osvětlovacích modelů.



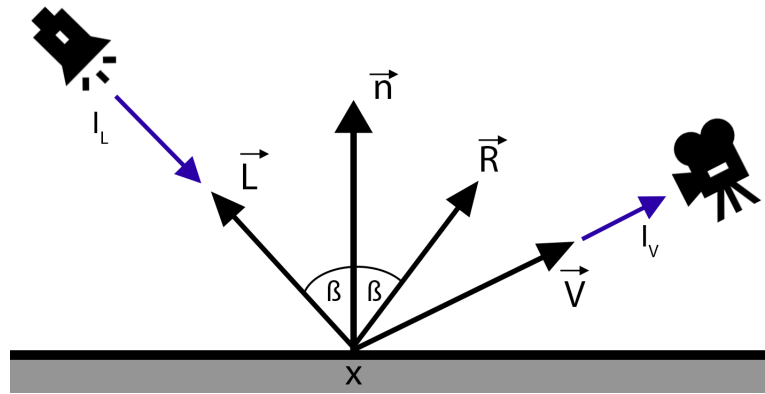
Obrázek 2.9: BRDF[2].

## 2.6 Osvětlovací modely

### Phongův osvětlovací model

Představen vědcem Bui-Tuong Phongem [11] roku 1977 na univerzitě v Utahu. Pomocí empirického zkoumání definoval osvětlovací model používaný pro výpočet odraženého světla. Na obrázku 2.10 jsou znázorněny vektory podílející se na výpočtu modelu.

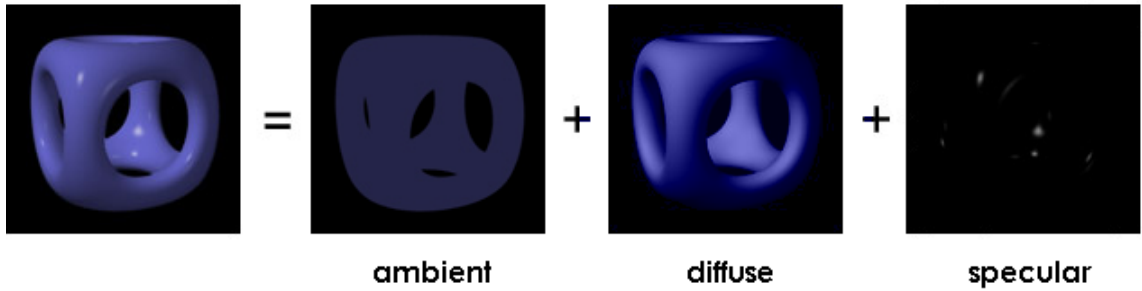
- $L$  – od bodu  $X$  ke světelnému zdroji
- $N$  – normálový vektor plochy v bodě  $X$



Obrázek 2.10: Vektory ve phongovu modelu.

- $R$  – ideální směru zrcadlového odrazu, je symetrický s vektorem  $L$  podle normály výpočet pomocí vztahu 2.9.
- $V$  – vektor pohledu, od bodu  $X$  ke kameře

Phong rozlišil tři druhy odrazu světla od povrchu na ambientní, difuzní a zrcadlový, které po složení vytváří výsledek. Na obr. 2.11 jsou tyto složky zobrazeny jednotlivě a pohromadě jako výsledný obraz.



Obrázek 2.11: Skládání výsledného osvětlení.[15]

### Zrcadlová složka

Ve světle je vyjádřena vtahem 2.9, ze kterého vyplývá, že čím jsou vektory  $\vec{R}$  a  $\vec{V}$  blíží, je odraz světlejší. Pokud je skalární součin  $\vec{V} \cdot \vec{R} < 0$  je hodnota  $I_S$  rovna 0. V rovnici  $I_L$  představuje barvu dopadajícího paprsku, koeficient odrazu  $r_s \in [0, 1]$  určuje míru zastoupení zrcadlové složky v celkově odraženém světle. Vektory  $\vec{V}$  a  $\vec{R}$  jsou popsány výše. Pro modifikaci ostroty se skalární součin vektorů umocní Phongovým exponentem  $h$  nabývajících hodnot v rozmezí  $[1, \infty]$ . S vyšší hodnotou exponentu bude lesklá plocha menší a intenzivnější.

$$I_S = I_L r_s (\vec{V} \cdot \vec{R})^h \quad (2.9)$$

### Difuzní složka

Pro určení zastoupení difúzní složky v odraženém světle platí vztah 2.10. Kde  $I_L$  je barva dopadajícího světla a stejně jako u zrcadlové složky intenzita roste čím je směr dopadu blíže normále, princip je založen na základě Lambertova zákonu. V případě  $\vec{L} \cdot \vec{N} < 0$  je difuzní složka nulová, Koeficient  $r_d$  udává zastoupení difuzní složky.

$$I_D = I_L r_d (\vec{L} \cdot \vec{N}) \quad (2.10)$$

### Ambientní složka

Okolní rozptýlené světlo přicházející ze všech směrů vzniklo mnohonásobnými odrazy od ostatních těles a rozptylem zapříčiněný molekulami vzduchu. Složka je vyjádřena dle vztahu 2.11, kde  $I_a$  reprezentuje množství okolního světla, v empirických modelech je konstantní pro celou scénu a koeficient  $r_a$  má podobnou vlastnost jako  $r_d$  u difuzní složky.

$$I_A = I_a r_a \quad (2.11)$$

Výsledné světlo  $I_V$  se získá pomocí součtu 2.12.

$$I_V = I_S + I_D + I_A \quad (2.12)$$

### Blinn-Phongův osvětlovací model

Zrcadlovou složku lze vypočítat pomocí zjednodušeného vztahu 2.13, který navrhl Blinn. Princip spočívá v zamezení složitějšího výpočtu odraženého paprsku. Místo něj se získá půlvektor  $\vec{H}$ . V rovnici 2.14 se vyskytují již známé vektory. Pro vyšší rychlost je tento model používán v OpenGL a DirectX.

$$I_S = I_L r_s (\vec{H} \cdot \vec{N})^h_B \quad (2.13)$$

$$\vec{H} = \frac{\vec{L} + \vec{V}}{\|\vec{L} + \vec{V}\|} \quad (2.14)$$

## 2.7 Existující programy

### Cinema4D

Jedná se o komerční program k vytváření 3D grafiky, který vyvíjí německá společnost Maxon Computer. Program je oblíbený především kvůli jednoduchému ovládání a přívětivému rozhraní. Umožňuje polygonální modelování, texturování, animace a realistické zobrazení, které za pomoci externích doplňků působí vysoce realistickým dojmem. První verze vyšla v roce 1993.

### Blender

Program je vydáván pod licencí GNU GPL. Oblíbený je hlavně pro multiplatformnost a bezplatné použití i ke komerčnímu využití. Doplnění funkčnosti lze pomocí Python skriptů. První verze byla vydána roku 1995.



## Kapitola 3

# Globální zobrazovací metody

V této kapitole jsou popsány metody a principy realistického zobrazení scén. Nejprve jsou uvedeny v kap 3.1 optické jevy, které nastávají při kontaktu se světelným paprskem nebo ostatními předměty v okolí. Dále je popsán základní matematický aparát 3.2 pro popis odrazu světla po dopadu paprsku v určitém bodě, který je pomocí metod aproximován. Existují dvě základní skupiny metod, které se dělí dle závislosti na směru pohledu. Všechny globální metody se snaží nalézt řešení zobrazovací rovnice.

### 3.1 Optické jevy

#### Přímé osvětlení – (angl. direct illumination)

Reprezentuje lokální osvětlovací model popsáný v kapitole 2.6, který simuluje výpočet barvy objektu po jediném odrazu světla. Dle notace 2.4 lze dráhu světla zapsat pomocí LDE, LSE nebo LGE.

#### Stín

Člověk za pomoci stínu intuitivně chápe umístění objektů ve scéně a pomáhá nám vnímat objekt ve třech dimenzích. Charakteristika stínu je dána vlastnostmi a typem světelného zdroje. Pro bodový zdroj vznikají ostré stíny, které oproti měkkým stínům nemají oblast s postupným nárůstem šedi.

#### Zrcadlový odraz

Hladké materiály s vysokou odrazivostí vytváří odlesky světla nebo ostatních předmětů ve scéně na svém povrchu. Trajektorie je  $L(S+|G+)*E$

#### Difuzní nepřímý odraz

Tento odraz patří mezi nejnáročnější jevy. Pro výpočet je potřeba delší čas a vyšší výpočetní výkon. Jev je fyzikálním nahrazením ambientní složky ve Phongově modelu. Projevuje se v reálném světě například jako pomalá změna intenzity odrazu světla v rozích místností.

### Difuzní přenos barvy – (angl. color bleeding)

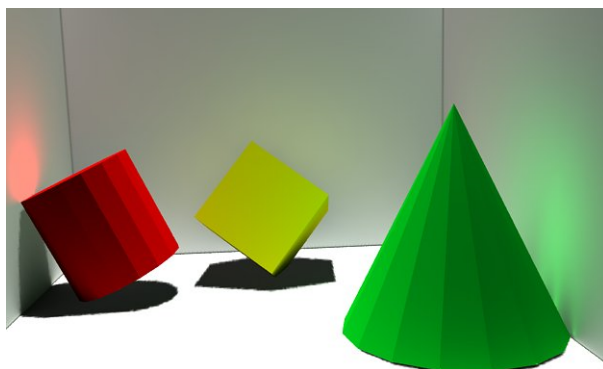
Jev vzniká důsledkem difuzního odrazu světla vznikající v těsné blízkosti dvou difúzních povrchů různé barvy. První objekt emituje svou barvu na druhý a opačně. Efekt je na obrázku 3.1, kde na bílých zdech lze vidět barevné odrazy od předmětů stojících poblíž.

### Kaustika

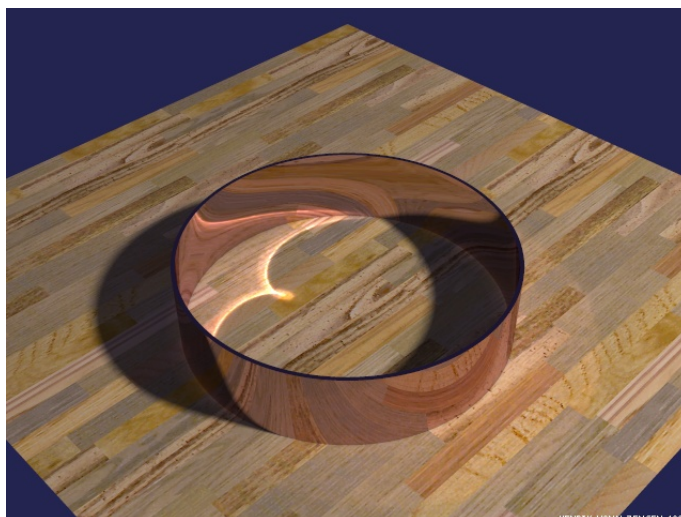
Lidově se jevu říká „prasátka“ na obrázku 3.2, který vzniká vysokou koncentrací paprsků poblíž jednoho místa, které se odráží od zrcadlového nebo průhledného materiálu.

### Opticky aktivní média – (angl. participating media)

Zobrazovací rovnice popsaná v další části nezahrnuje transport světla aktivním médiem, simulace je tedy obtížná. Efekt nastává po dopadu paprsků na částice prachu, vody, nebo dýmu. Pro běžnou simulaci se tento jev neuvažuje, i když v posledních letech se začíná zvyšovat zájem.



Obrázek 3.1: Difuzní míchání barev[12].



Obrázek 3.2: Kaustika.[6]

## 3.2 Zobrazovací rovnice

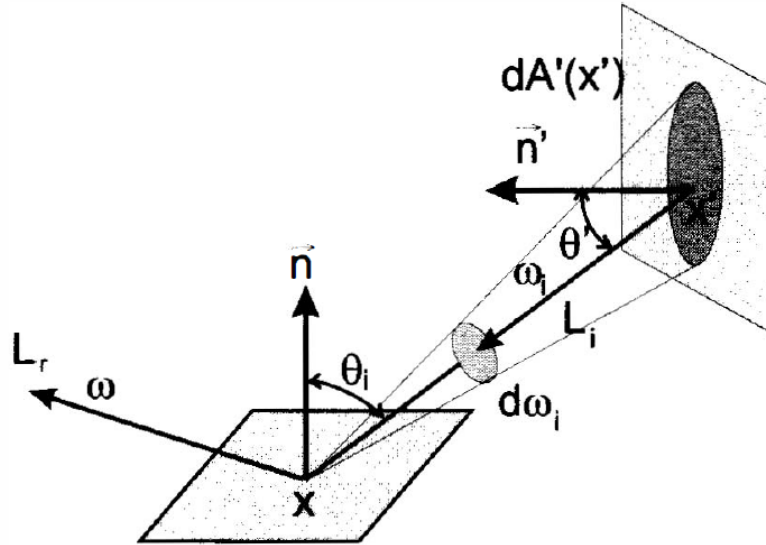
Rovnice 3.1 byla formulována Jamesem Kajiyou roku 1986 v článku [8]. Rovnice matematicky popisuje zobrazování scény. Jejím vyřešením je vypočtena vycházející radiance  $L_o$  z každého bodu  $x$  na povrchu objektů v jakémkoliv směru  $\vec{\omega}$ . Podmínkou pro platnost rovnice je zákon o zachování energie, kdy radiance opouštějící bod  $x$  musí být odrazena nebo absorbována, proto rovnice nepopisuje opticky aktivní prostředí.

$$L_o(x, \vec{\omega}) = L_e(x, \vec{\omega}) + \int_{\Omega} f(x, \vec{\omega}, \vec{\omega}_i) L_o(x, -\vec{\omega}_i) \cos \Theta d\vec{\omega}_i \quad (3.1)$$

kde:

- $L_e(x, \vec{\omega})$  je radiance vyzářená z bodu  $x$  je-li bod  $x$  světelný zdroj.
- integrál představuje integraci přes polokouli  $\Omega$
- $f(x, \vec{\omega}, \vec{\omega}_i)$  je dvou směrová odrazová distribuční funkce BRDF
- $L_o(x, -\vec{\omega}_i)$  je příchozí radiance do bodu  $x$  ze směru  $-\vec{\omega}_i$
- $\cos \theta$  je úhel sevřený směrem  $\vec{\omega}_i$  a normálovým vektorem povrchu  $\vec{n}$  v bodě  $x$

Původní rovnice patří mezi Fredholmovy rovnice druhého řádu a komplikací pro výpočet je, že se radiance vyskytuje na levé i pravé straně uvnitř integrálu. Proto je výhodnější formulace udávající odraženou radianci jako integrál přes přispívající plochy. Na základě principu znázorněného na obrázku 3.3, lze rovnici modifikovat na rovnici 3.2.



Obrázek 3.3: Zobrazovací rovnice přes přispívající plochy.[2]

$$L_o(x, \vec{\omega}) = L_e(x, \vec{\omega}) + \int_S f(x, x' \rightarrow x, \vec{\omega}) L_i(x', \rightarrow x) V(x, x') G(x, x') dA' \quad (3.2)$$

kde:

- $L_e(x, \vec{\omega})$  je radiance vyzářená z bodu  $x$  je-li bod  $x$  světelný zdroj.

- integrál představuje integraci přes přispívající plochy  $S$  představující odraženou radianci v bodě  $x$  a směru  $\vec{\omega}$
- $x' \rightarrow x$  je aproximace vektoru  $\vec{\omega}_i$  z bodu  $x'$  do bodu  $x$
- $V(x, x')$  tzv. visibility term nabývající hodnot [3.3](#).

$$V(x, x') = \begin{cases} 1 & \text{pokud je bod } x \text{ viditelný z bodu } x' \\ 0 & \text{není viditelný} \end{cases} \quad (3.3)$$

- $G(x, x')$  tzv geometry term v sobě zahrnuje koeficienty vyjadřující promítnutí radiance po vyzaření z bodu  $x'$  a po dopadu do bodu  $x$ . Term má tvar [3.4](#).

$$G(x, x') = \frac{(\vec{\omega} \cdot \vec{n}')(\vec{\omega}' \cdot \vec{n})}{\|x' - x\|^2} \quad (3.4)$$

Upravená rovnice udává, že pro výpočet radiance odcházející ve směru  $\vec{\omega}$  z bodu  $x$  na ploše, musíme vzít v úvahu všechny ostatní plochy ve scéně, které jsou viditelné z tohoto bodu. Odcházející radiance je vynásobena dvou směrovou distribuční funkcí BRDF a získaným geometrickým členem. Integrál přes všechny plochy ve scéně, spolu s vlastní radiancí udává vyzařenou radianci daným směrem.

### 3.3 Sledování paprsku

Metoda sledování paprsku (angl. raytracing) patří do třídy globálních metod závislých na směru pohledu, který začíná od pozorovatele s trajektorií  $L(D|G)?S^*E$ . Představená byla v roce 1980 Turnerem Whittedem [\[14\]](#).

Je založená na jednodušší metodě vrhání paprsku (angl. raycasting), která byla představená již v roce 1968 Arturem Appelem [\[1\]](#). Nevýhodou starší metody bylo zanedbání interakce mezi tělesy, jelikož pro získání barvy pixelu je vyslán paprsek od pozorovatele a v místě nejbližšího dopadu je brán za výsledek pouze výpočet lokálního osvětlení. Metodě se proto říká sledování paprsku prvního řádu.

Whitted raycasting modifikoval a přidal ke sledování paprsky, které prochází nebo se zrcadlově odrazí od tělesa. Lze zobrazit na povrchu zrcadlové obrazy jiných těles a vržené stíny. Při výpočtu výsledné barvy je rekurzivně volaná funkce [3.1](#) se vstupním parametrem hloubky rekurze roven 0. Rekurse je ukončena dokud tento parametr není větší než povolená maximální hloubka.

```
sledujPaprsek(paprsek, hloubkaRekurze)
{
    if najdiPrusecik(Ray)
    {
        barva = LokalniOsvetleni(prusecik)
        if hloubkaRekurze <= MaxHloubka
        {
            barva += SledujPaprsek(odrazenyPaprsek, Deep + 1)
            barva += SledujPaprsek(lomenyPaprsek, Deep + 1)
        }
        return barva;
    }
}
```

```

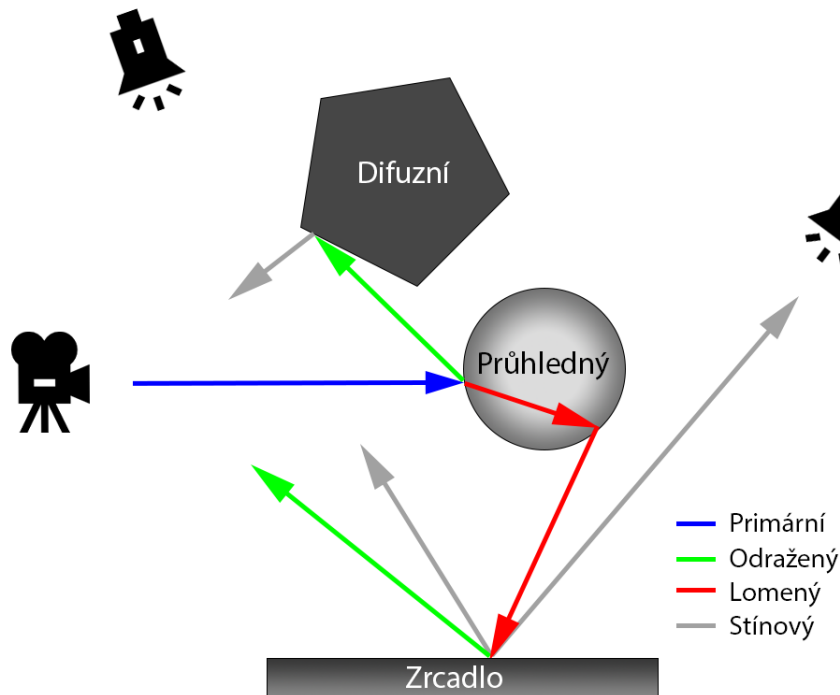
    }
    else
        return Barva_pozadi
}

```

Algoritmus 3.1: Rekurzivní funkce pro sledování paprsků

Pro paprsky putující scénou se zavedlo následující označení:

- Primární paprsek (primary ray) je vyslán od kamery procházející průmětnou.
- Sekundární paprsek (secondary ray) vzniká při dopadu primárního nebo sekundárního paprsku. Po dopadu jsou v průsečíku vytvořeny dva paprsky zrcadlově odražený nebo lomený. Sekundárních paprsků je během simulace nejvíce.
- Stínový paprsek (shadow ray) nachází vržené stíny. Je vyslán z bodu dopadu ke světelnému zdroji. Pokud nenarazí na jiné těleso mezi tímto bodem a světlem, tak se světelný zdroj podílí na osvětlení ve výpočtu lokálního osvětlovacího modelu. Z bodu je vyslán stejný počet stínících paprsků jako je počet světél.



Obrázek 3.4: Typy paprsků.

Phongova rovnice 2.12 pro výpočet výsledné barvy je pro sledování paprsků modifikována na rovnici 3.5, kde  $I_S$ ,  $I_D$  a  $I_A$  je již vysvětleno dříve.  $I_R$  udává intenzitu odraženého paprsku a  $I_T$  lomeného paprsku. Intenzity jsou poníženy o koeficient daného odrazu.

$$I_V = I_S + I_D + I_A + I_R + I_T \quad (3.5)$$

$$I_R = r_s I_R \quad (3.6)$$

Metoda sledování paprsků dokáže simulovat jevy jako vržené stíny, odražené nebo lomené objekty. Nedokáže ovšem zobrazit kaustiky a nepřímé osvětlení, protože nebere vůbec v úvahu paprsky odražené od difuzních těles.

### 3.4 Sledování cesty

Angl. Path Tracing stejně jako raytracing sleduje paprsky od pozorovatele. Metoda byla popsána poprvé společně se zobrazovací rovnicí [8]. Výhodou oproti raytracingu je kompletní vyřešení zobrazovací rovnice 3.1, kde do výpočtu jsou zahrnuty nepřímé difuzní odrazy a kaustiky. Notace transportu je tedy  $L(S|D)+E$ . Metoda je schopna simulovat různé světelné zdroje.

K získání barvy jednoho pixelu je vysláno několik paprsků, kde ze všech vzorků je výsledná hodnota zprůměrována. Při nalezení průsečíku paprsku s tělesem je stejně jako u sledování paprsků vypočten lokální osvětlovací model v daném bodě, rozdílný je přístup k odrazům.

Raytracing bere v úvahu pouze zrcadlové odrazy, ale pathtracing vypočítává náhodný odraz v každém bodě. Díky tomu je v simulaci sledováno mnoho paprsků a důsledkem jsou vyšší nároky na čas a výpočetní výkon. Při snížení počtu paprsků pro vzorkování vzniká šum.

### 3.5 Radiozita

Metoda sledování paprsků popsaná v části 3.3 kvalitně simuluje zrcadlové odrazy nebo průhledné objekty, ale na fyzikální simulaci reálného světa nestačí. Nelze simulovat jiné typy světla než bodový zdroj a tedy pouze ostré stíny. Nepřímé difuzní osvětlení využívá konstantní ambientní složku z Phongova modelu.

K dosažení věrné simulace šíření světla přispěli vědci Goralová, Torrance, Greenberg a další v polovině osmdesátých let. Využili poznatků z oblasti výpočtu tepelného záření. Radiozita vychází ze zákona o zachování energie. Existuje předpoklad přenosu světla mezi objekty v energeticky uzavřené scéně, kde přenos není ovlivněn prostředím. Objekty jsou neprůhledné a dochází pouze k difuzním odrazům.

Jelikož pro difuzní odrazy nezáleží na pozici pozorovatele aplikuje radiozitivní rovnice 3.7 speciální přístup pro řešení zobrazovací rovnice 3.2.

$$B(x) = E(x) + \rho(x) \int_S B(x') G(x, x') dx' \quad (3.7)$$

kde:

- Radiozita  $B(x)$  je světelný tok vyzářený v bodě  $x$ .  $B(x) \geq 0$
- $\rho(x)$  je difuzní odrazivost v bodě  $x$ .  $0 \leq \rho(x) \leq 1$
- $B(x)$  zahrnuje geometrické informace dvojic povrchů
- $S$  je množina všech ploch ve scéně.

### 3.6 Sledování fotonů

Metoda sledování fotonů (anglicky Photon mapping) byla publikována v roce 1995 Henrikem Wann Jensenem [7]. Snahou bylo vytvořit nový mechanismus k výpočtu zobrazovací rovnice, který bude schopen zobrazit rozsáhlou scénu a všechny optické jevy. Radiozita věrně simuluje pouze difuzní odrazy a protože scéna musí být rozdělena na menší plošky je výpočet větších scén náročný. Pathtracing věrně simuluje všechny jevy za cenu velmi vysokých nároků na výkon.

Jensen definoval nový přístup k zobrazování, kdy rozdělil výpočet na dvě fáze. V prvním průchodu jsou emitovány fotony od zdroje světla, které jsou ve scéně sledovány. Po dopadu na objekt je foton uložen do fotonové mapy. Fotonová mapa zanedbává geometrickou reprezentaci scény díky tomu umožňuje zobrazování složitých scén nebo procedurálně definovaných objektů. Ve druhém průchodu probíhá klasický raytracing s využitím mapy k získání příchozí radiance v daném bodě.

#### Tvorba fotonové mapy

Fotony nesou informaci o energii světelného zdroje. Energie jednoho fotonu je vypočítána dle rovnice 3.8, kde  $P$  je energie a  $N$  celkový počet generovaných fotonů.

Po dopadu fotonu na difuzní povrch jsou do fotonové mapy uloženy údaje o směru dopadu a úbytku světelného toku  $\Delta\Phi_p(x, \vec{\omega}_p)$

$$P_{foton} = \frac{P_{zdroj}}{N} \quad (3.8)$$

#### Generování fotonu

Světelné zdroje emitují fotony, které jsou vystřeleny do prostoru. Směr fotonu je daný typem světla popsaných v části 2.4.

Algoritmus pro generování směru fotonů z difuzního bodového zdroje využívá metody Monte Carlo. Náhodně jsou generovány čísla pro všechny osy a pokud výsledný vektor není uvnitř jednotkové kružnice dochází k opětovnému generování směru.

```
generujFoton()  
{  
    do  
    {  
        // x,y,z je smer v prostoru na jednotlivych osach  
        x = nahodne cislo mezi <-1,1>  
        y = nahodne cislo mezi <-1,1>  
        z = nahodne cislo mezi <-1,1>  
    } while ( x * x + y * y + z * z > 1 )  
}
```

Algoritmus 3.2: Generování směru fotonu

#### Více zdrojů světla

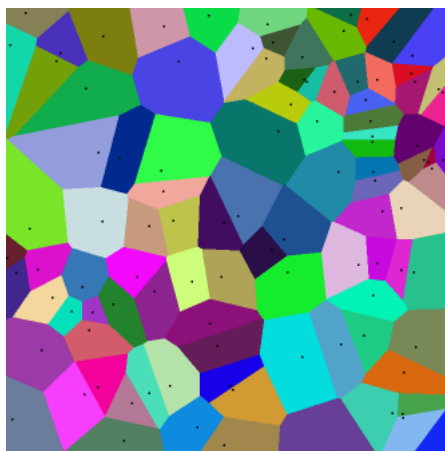
Pokud scéna obsahuje více světla, jsou fotony emitovány a sledovány od každého z nich. Je-li některé světlo jasnější než jiné generuje se větší počet fotonů.

## Datová struktura fotonové mapy

Během prvního průchodu jsou fotony postupně ukládány do pole nebo seznamu. Ve druhém průchodu se neustále vyhledává  $N$  nejbližších fotonů k získání světelného toku v bodě. Prohledávání v poli je nevyhovující a dochází tedy k transformaci na jinou datovou strukturu.

## Voronův diagram

Struktura je vhodná pro rychlé vyhledávání nejbližších fotonů, které jsou vzájemně propojeny se svými sousedy. Start vyhledávání probíhá na náhodném prvku a poté jsou rekurzivně hledáni sousedi. Časová složitost pro je  $O(k \log n)$  kde  $k$  je počet požadovaných sousedů. Problémem jsou vysoké nároky na paměť  $O(n^2)$ . Při vysokém počtu fotonů je struktura nevhodná.



Obrázek 3.5: Voronův diagram[16].

## KD strom

Jedná se o prostorovou datovou strukturu, která je rozšířením klasického binárního stromu o více dimenzí. Každý uzel obsahuje ukazatele na své levé a pravé podstromy, které štěpí rovinu na dvě části. Rychlost vyhledání jednoho fotonu je v nejlepším případě  $O(\log n)$  a v nejhorším  $O(n)$ . Složitost vyhledání  $k$  nejbližších bodů je  $O(k + \log n)$ . Strom lze sestavit propojením pomocí ukazatelů a nebo lze využít principu, kdy prvek na pozici  $i$  obsahuje svůj levý podstrom na pozici  $2 * i$  a pravý podstrom na pozici  $2 * i + 1$ . Druhým principem lze ušetřit až 40% paměti, protože nejsou ukládány ukazatele.

## Typy map

- Globální - Reprezentuje difuzní přenos mezi objekty
- Kaustická - Slouží ke zvýšení ostroty kaustik. Jsou uloženy fotony, které dopadly alespoň jednou na zrcadlový nebo průhledný povrch.
- Objemová (volume) - fotony pocházející od nepřímého osvětlení aktivním médiem například svíčka nebo průchod skrz mlhu.



## Renderování scény

Po sestavení fotonové mapy dochází ke sledování paprsku od pozorovatele. Princip je stejný jako u raytracingu s rozdílným způsobem výpočtu barvy v případě paprsku na difuzní povrch. V daném bodě je nutné vypočítat radianci z fotonové mapy. Odražená radiance  $L_r$  v bodě  $x$  se vypočítá pomocí rovnice 3.9, kde  $L_i$  lze zaměnit rovnicí 3.10 vyjadřující vztah mezi radiancí a světelným tokem. Po úpravě vznikne rovnice 3.11.

$$L_r(x, \vec{\omega}) = \int_{\Omega} f(x, \vec{\omega}, \vec{\omega}_i) L_i(x, \vec{\omega}_i) \cos \Theta d\vec{\omega}_i \quad (3.9)$$

$$L_i(x, \vec{\omega}_i) = \frac{d^2 \Phi_i(x, \vec{\omega}_i)}{\cos \Theta d\vec{\omega}_i dA} \quad (3.10)$$

$$L_r(x, \vec{\omega}) = \int_{\Omega} f(x, \vec{\omega}, \vec{\omega}_i) \frac{d^2 \Phi_i(x, \vec{\omega}_i)}{dA} \quad (3.11)$$

Po převedení integrálu na sumu vznikne rovnice 3.12.

$$L_r(x, \vec{\omega}) = \int_{\Omega} f(x, \vec{\omega}, \vec{\omega}_i) \frac{d^2 \Phi_i(x, \vec{\omega}_i)}{dA} \approx \sum_{p=1}^N f(x, \vec{\omega}, \vec{\omega}_i) \frac{\Delta \Phi_p(x, \vec{\omega}_p)}{\Delta A} \quad (3.12)$$

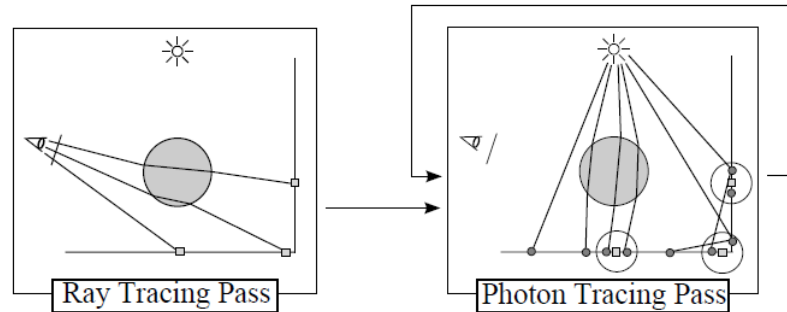
Hodnotu  $\Delta A$  lze aproximovat na výslednou rovnici 3.13.

$$L_r(x, \vec{\omega}) \approx \frac{1}{\pi r^2} \sum_{p=1}^N f(x, \vec{\omega}, \vec{\omega}_i) \Delta \Phi_p(x, \vec{\omega}_p) \quad (3.13)$$

## Progresivní photon mapping

Photon mapping představený Henrikem Jensenem poskytuje velmi dobré výsledky. Úskalím metody je, že k dobrému výsledku je nutné vygenerovat velmi mnoho fotonů a fotonová mapa zabere dost paměti. V roce 2008 Hachisuk [3] klasický photon mapping modifikoval na iterativní metodu. Uvedl, že lze generovat fotonové mapy iterativně s menším počtem fotonů a světelný tok akumulovat.

Zaměnil pořadí průchodů, které je znázorněno na obrázku 3.6. Nejprve je spuštěn klasický raytracing, ve kterém se po dopadu na difuzní povrch uloží do struktury zvané hitpoint informace potřebné ke syntéze výsledného obrazu.



Obrázek 3.6: Fáze progresivního foton mappingu[3].

## Kapitola 4

# Důvod a plán práce

### Důvod práce

Během studia na střední škole elektrotechnické jsem nemohl pracovat v plném rozsahu s informačními technologiemi. Získával jsem pouze základy s elektronikou a programováním mikrokontroléru, a proto jsem se dále rozhodl plně věnovat studiu informatiku na VUT. Rozhodoval jsem se, zda se věnovat vytváření informačních systémů, rozvoje umělé inteligence nebo věnovat se sítím jako jejich správce. Během studia na VUT fakulty informatiky jsem získal přehled o všech částech informatiky, které v současné době využívám ve svém dalším rozvoji.

Počítačová grafika mě ale velmi zaujala díky předmětu základy počítačové grafiky, kde jsem získal i mou nejlepší známku. Pro téma bakalářské práce jsem si vybral velmi zajímavou práci: "Fotorealistické zobrazování". Téma pro mě nebylo zdaleka lehké, jelikož s programováním v této oblasti jsem neměl potřebné zkušenosti, ale přeci jen jsem si chtěl velmi vyzkoušet vytvořit vlastní renderer.

### Plán práce

- Nastudování zobrazovacích algoritmů
- Namodelování objektů v externím programu
- Načtení scény a její zpracování
- Implementace metody sledování paprsku
- Implementace progresivní metody sledování fotonu

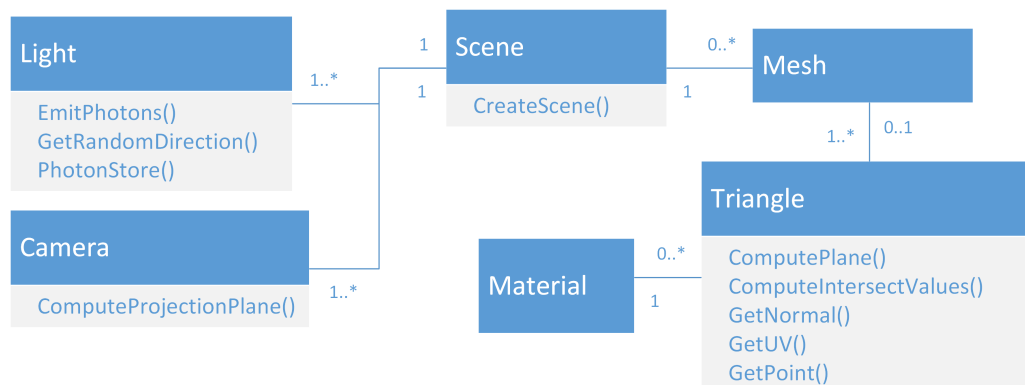
## Kapitola 5

# Popis práce

V kapitole je popsána praktická část bakalářské práce. Je uvedeno objektové schéma jednotlivých funkčních částí a postup implementace jedné z fotorealistických metod.

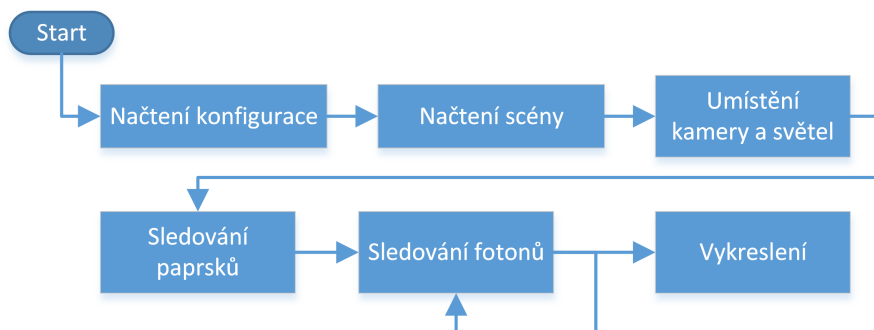
### 5.1 Blokové a funkční schéma programu

Následující schéma zobrazuje základní objekty, které se podílejí na reprezentaci scény a těles. Uvedené metody u tříd jsou zmíněny dále v textu.



Obrázek 5.1: Objektová struktura scény.

Diagram 5.2 popisuje procesy, které jsou po startu aplikace prováděny.



Obrázek 5.2: Diagram běhu programu.

## 5.2 Konfigurace programu

Na začátku běhu programu se inicializuje třída `Config` implementovaná dle návrhového vzoru jedináček. Třída obsahuje vlastnosti zobrazovacích metod, nastavení cesty k souboru s definicí scény nebo rozlišení výsledného obrazu. Výchozí parametry lze modifikovat pomocí souboru `konfigurace.cfg`.

## 5.3 Vytvoření scény

Modelování geometrie těles a jejich rozmístění v prostoru probíhalo v grafickém programu Cinema4D. Program byl zvolen vzhledem ke zkušenostem. Po umístění předmětů na své pozice, byly dodány světelné zdroje a tělesům nastaveny textury, barvy a parametry odrazu nebo lomu.

Během testování exportu scény do formátu Collada popsaného v části 2.1, byl zjištěn problém s exportem kamery. Nedocházelo ke správnému nastavení parametrů kamery `aspect_ratio` a `xfov`. Scéna po načtení v programu byla zdeformovaná. Řešením problému je využití druhého grafického editoru Blender 2.7, ve kterém je scéna naimportována. Po importu je potřeba opravit materiálové vlastnosti objektů, umístit kameru a bod určující směr pohledu kamery. Napojení kamery s bodem pohledu je řešeno pojmenováním bodu složením z `Název-kamery` a přípony `_Target`. Nyní po exportu je scéna připravena k načtení v programu.

## 5.4 Načítání scény

Načtení scény probíhá po inicializaci konfiguračních dat. V konstruktoru třídy `Scene` se nastaví pozadí a výchozí index lomu paprsku při simulaci. Načtení samotné scény probíhá voláním metody `ParseSceneFile` s parametrem cesty ke Collada souboru.

Soubor je analyzován pomocí open source knihovny Assimp, která obsahuje parser ke zpracování souborů s uloženou scénou. Kromě Collady knihovna podporuje formáty Autodesk, 3ds Max, Wavefront object a další. Zpracovaná data jsou ukládána do struktur. Struktura `aiScene` je hlavní struktura reprezentující scénu.

Povrchové vlastnosti těles jsou převáděny z assimp struktury `aiMaterial` do třídy `Material` metodou `LoadMaterial`. Načtení barev a většiny parametrů probíhá korektně, ale index odrazivosti je nutné opravit. Je-li materiál kompletně difuzní v souboru se neobjevuje parametr `<reflectivity>`. Při analýze je očekávána hodnota 0, ale assimp vrací 1. Nastává problém se specifikováním materiálu s kompletním odrazem. Řešením je ruční úprava parametru v souboru snížením hodnoty z 1 o nastavenou mez. Při zpracování je načtená hodnota invertována a dle kódu 5.1 upravena. Mez je nastavena na 5.00e-005.

```
if (reflectiveFactor > 0)
{
    if (reflectiveFactor < bound)
    {
        reflectiveFactor = 0;
    }
    reflectiveFactor = 1.0 - reflectiveFactor;
}
```

Algoritmus 5.1: Oprava hodnoty odrazivosti

## Kolekce objektů

V rámci programu jsou některé objekty využívány ve všech funkčních částech, a proto statická třída `ObjectContainer` obsahuje reference na ně.

## 5.5 Promítací průmětna

Jakmile je scéna načtena metodou `ComputeProjectionPlane` jsou nastaveny parametry promítání. Při zobrazování je použita perspektivní projekce, která je zmíněná v kapitole 2.3. Z parametrů kamery je potřeba vypočítat promítací průmětnu, přes kterou prochází vysílané paprsky. Průmětna je definovaná body `topLeft`, `topRight`, `bottomLeft` a `bottomRight`. K jejich výpočtu je nejprve nutné získat směr pohledu kamery pomocí rovnice 5.1, kde *Target* je pozice bodu pojmenovaného *Název-kamery* *\_Target* a *Origin* je pozice kamery.

$$\vec{Direction} = Target - Origin \quad (5.1)$$

Po získání směru pohledu je vypočítán rovnicí 5.2 střed promítací průmětny, kde *near* je parametr kamery udávající vzdálenost průmětny od pozice kamery.

$$Center = Origin + \vec{Direction} * near \quad (5.2)$$

Pro získání vektorů *right* a *up* definujících spolu s bodem *Center* rovinu platí následující vztah, kde  $\vec{X} = (0, 0, -1)$

$$\begin{aligned} \vec{Right} &= \vec{Direction} \times \vec{X} \\ \vec{Up} &= \vec{Right} \times \vec{Direction} \end{aligned} \quad (5.3)$$

V dalším kroku je potřeba vypočítat výšku *h* a šířku *w* průmětny pomocí zorného pole *fov* a *aspect* ratia.

$$\begin{aligned} h &= 2 \tan \frac{fov}{2} near \\ w &= h * aspect \end{aligned} \quad (5.4)$$

Z výpočtu výšky a šířky je možné získat výsledné vektory potřebné k získání rohů průmětny.

$$\begin{aligned} \vec{width} &= \vec{right} * w * 0.5 \\ \vec{height} &= \vec{up} * h * 0.5 \end{aligned} \quad (5.5)$$

Body určující obdélník průmětny jsou získány ze vztahu:

$$\begin{aligned} topLeft &= Center - \vec{Width} + \vec{Height} \\ topRight &= Center + \vec{Width} + \vec{Height} \\ bottomLeft &= Center - \vec{Width} - \vec{Height} \\ bottomRight &= Center + \vec{Width} - \vec{Height} \end{aligned} \quad (5.6)$$

## 5.6 Sledování paprsků

Třída `RayTracer` implementuje první fázi realistického zobrazení. Běh raytracingu je spuštěn metodou `Run`.

### Vyslání paprsku

Pro rovnoměrné vyslání paprsků přes průmětnu je nutné určit horizontální a vertikální krok, který je připočítán ke směru předchozího paprsku. Nejprve je nutné vypočítat poměr mezi pixely pro šířku a výšku výsledného obrazu.

$$\begin{aligned} dx &= \frac{1}{sirka} \\ dy &= \frac{1}{vyska} \end{aligned} \tag{5.7}$$

Výsledné kroky jsou získány vynásobením poměrů s vektory  $\vec{TB}$  směřující shora dolů a  $\vec{LR}$  směřující zleva doprava. Kde  $\vec{TB} = bottomLeft - topLeft$  a  $\vec{LR} = topRight - topLeft$

$$\begin{aligned} horizontalStep &= \vec{LR} \cdot dx \\ verticalStep &= \vec{TB} \cdot dy \end{aligned} \tag{5.8}$$

Následující algoritmus za pomoci vypočtených hodnot rovnoměrně vysílá paprsky přes průmětnu.

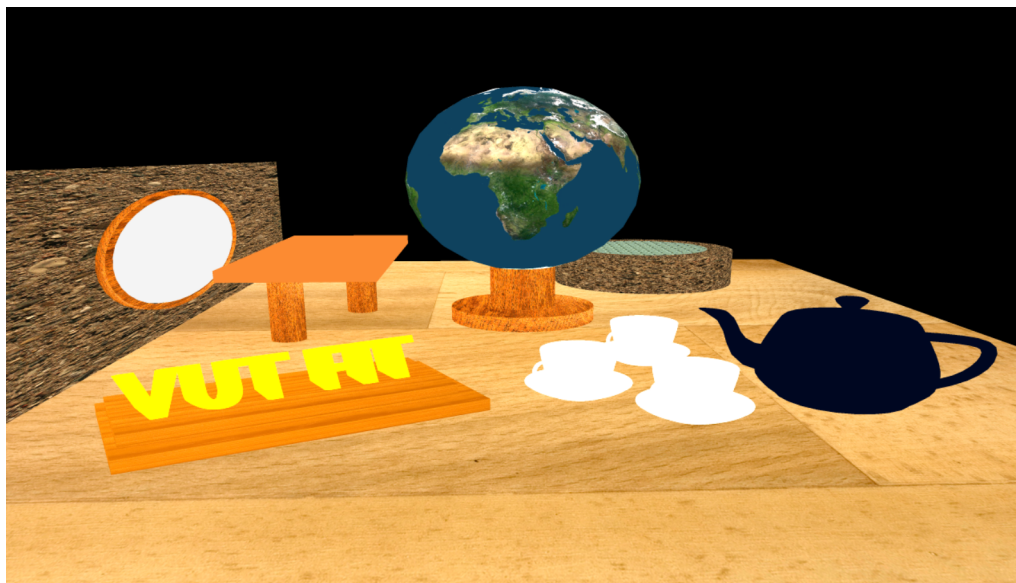
```
for (unsigned int y = 0; y < height; y++) {
    horizontal = vertical;
    for (unsigned int x = 0; x < width; x++) {
        rayDirection = horizontal - cameraPosition;
        color = TraceRay(rayDirection);
        putPixel(x, y, color);
        horizontal = horizontal + horizontalStep;
    }
    vertical = vertical + verticalStep;
}
```

Algoritmus 5.2: Vysílání paprsků průmětnou

### Výpočet barvy pixelu

Po získání aktuálního směru je paprsek vyslán metodou `TraceRayRun` do prostoru. Funkcí `FindIntersect` je vyhledáván průsečík paprsku s nejbližším objektem. V případě nenalezení průsečíku je vrácena barva pozadí, opačně objekt `Intersection` obsahuje informaci o vzdálenosti průsečíku od počátku paprsku a barycentrické hodnoty, které jsou vysvětleny v části 2.2. Z těchto hodnot je pomocí interpolace zjištěna pozice průsečíku v polygonu, normála a textura v daném bodě.

Pokud zanoření rekurze je menší než nastavená mez parametrem `ray-recursive-steps`, dochází k výpočtu barvy průsečíku metodou `ComputeColor`.

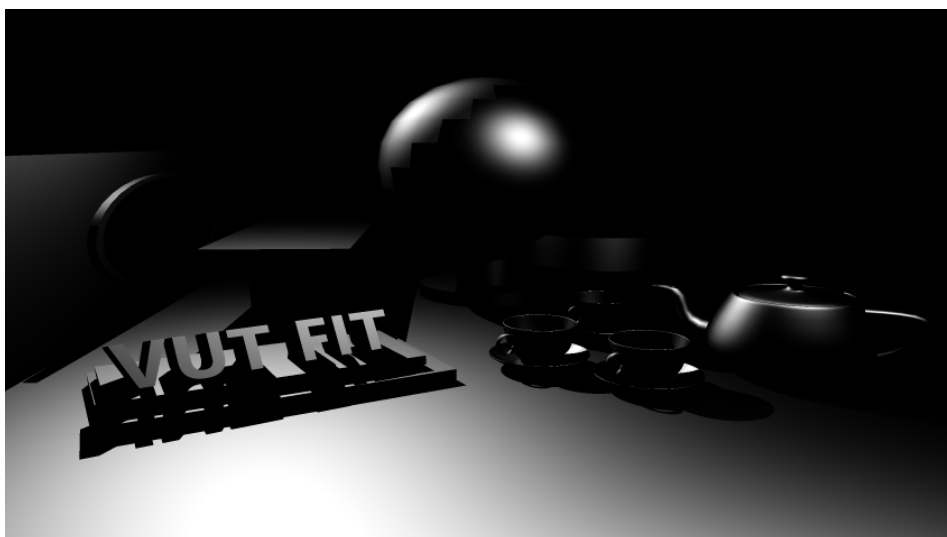


Obrázek 5.3: Difuzní složka.

Nejprve je vyhodnocena difuzní složka materiálu bez stínů. Obsahuje-li materiál texturu je nejprve nutné interpolací barycentrických hodnot vypočítat souřadnice  $u$  a  $v$  k získání hodnoty textury v daném bodě.

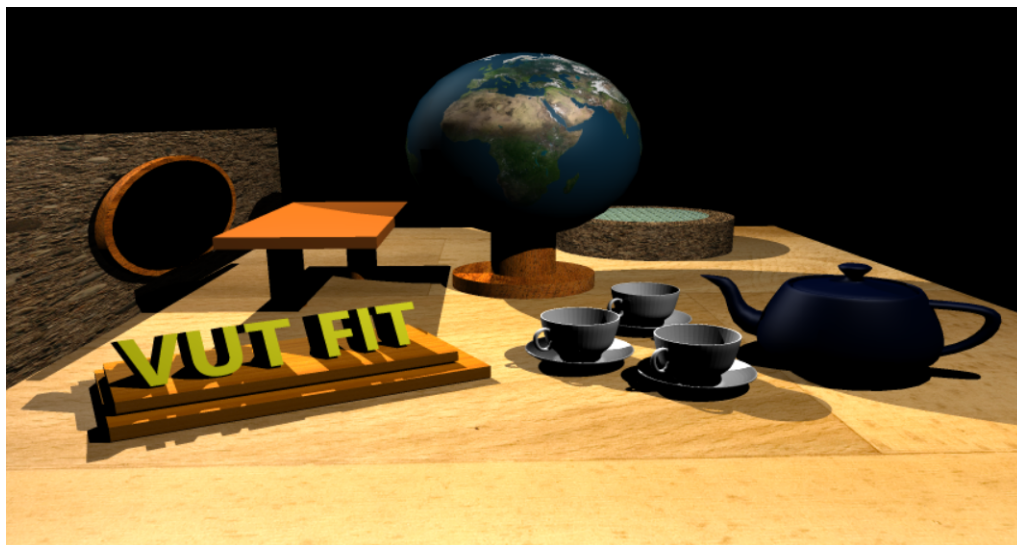
Pomocí difuzní složka vnímáme barvu tělesa, ale k docílení dojmu trojrozměrného prostoru nám pomáhají stíny. K tomu je potřeba vypočítat lokální osvětlení pomocí Phongova modelu 2.6. Stíny a spekulární odrazy jsou vypočteny metodou `CalculateDiffuseAndSpecular` ve, které jsou z průsečíku ke všem světelným zdrojům vyslány stínové paprsky. Nenarazí-li paprsek na jiné těleso během trajektorie podílí se dané světlo na osvětlení.

Difuzní složka je vynásobena s intenzitou přicházejícího světla, která je vypočtena dle rovnice 2.10. Spekulární část zobrazená na obrázku 5.4 se vypočítá dle rovnice 2.9.



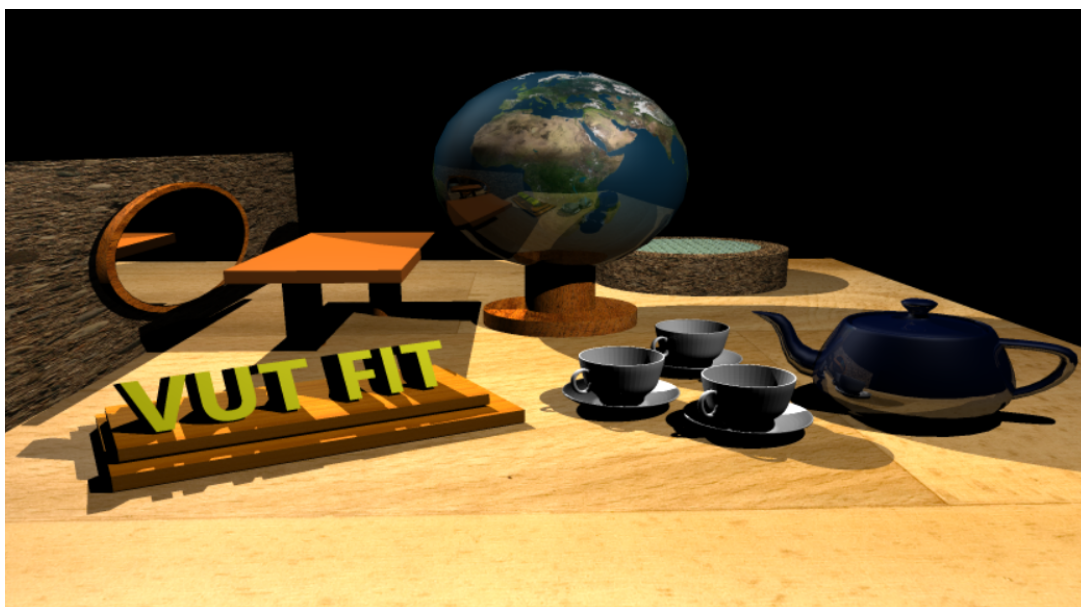
Obrázek 5.4: Spekulární složka.

Součtem všech částí je Phongův osvětlovací model vypočítán. Nyní obraz 5.5 obsahuje stíny a odlesky světla.



Obrázek 5.5: Lokální osvětlení

Raytracing rekurzivně vyhodnocuje odrazivé a průhledné plochy. Výsledný obraz 5.6 je součtem lokálního osvětlení a barvy vrácené z rekurzivního volání metody `TraceRay` s navýšeným parametrem hloubky rekurze.



Obrázek 5.6: Výsledek metody sledování paprsků.



## Uložení hitpointů

Výstupem je seznam hitpointů, které umožňují syntézu obrazu po dokončení následující fáze sledování fotonů. V této fázi jsou při interakci s difuzními předměty ukládány hitpointy, které obsahují souřadnice zkoumaného pixelu v průmětně, směr paprsku, aktuální hodnotu vypočtené barvy, pozici a normálu průsečíku.

## 5.7 Progresivní sledování fotonů

Po dokončení sledování paprsků a uložení hitpointů nastává druhá fáze realistického zobrazení. Ve druhém průchodu jsou sledovány fotony od světelných zdrojů komponentou **PhotonTrace**. Implementovaná je progresivní varianta, která probíhá dokud počet dokončených iterací není roven maximálního počtu iterací nastaveným parametrem **progressive-iterations**. V každém iteračním kroku jsou emitovány fotony ze světelných zdrojů, které reprezentuje třída **Light** obsahující světelnou charakteristiku a umístění v prostoru.

Metodou **EmitPhotons** je zahájeno generování fotonů. Nejprve probíhá simulace nepřímého osvětlení, kde výkon světla lze nastavit parametrem **light-indirect-power** v souboru **Konfigurace.cfg**. Energie jednoho fotonu je získána vztahem 3.8 a vynásobením barvou světla.

K získání náhodného směru je použita metoda **GetRandomDirection**, která implementuje algoritmus 3.2.

Generování probíhá dokud počet paprsků, které zasáhly scénu není roven požadovanému počtu fotonů.

Samotné sledování fotonů je implementováno v metodě **PhotonTrace**. Podobně jako u sledování paprsků dochází k zastavení rekurze v momentě, kdy hloubka rekurze překročí stanovenou mez definovanou parametrem **light-recursive-steps**.

Po dopadu na těleso je foton odražen v difuzním směru, zrcadlovém nebo je pohlcen. K rozhodnutí jakou akci zvolit je použit princip Ruské rulety.

Nejprve je nutné vypočítat vážený průměr všech barevných složek pro každý odraz.

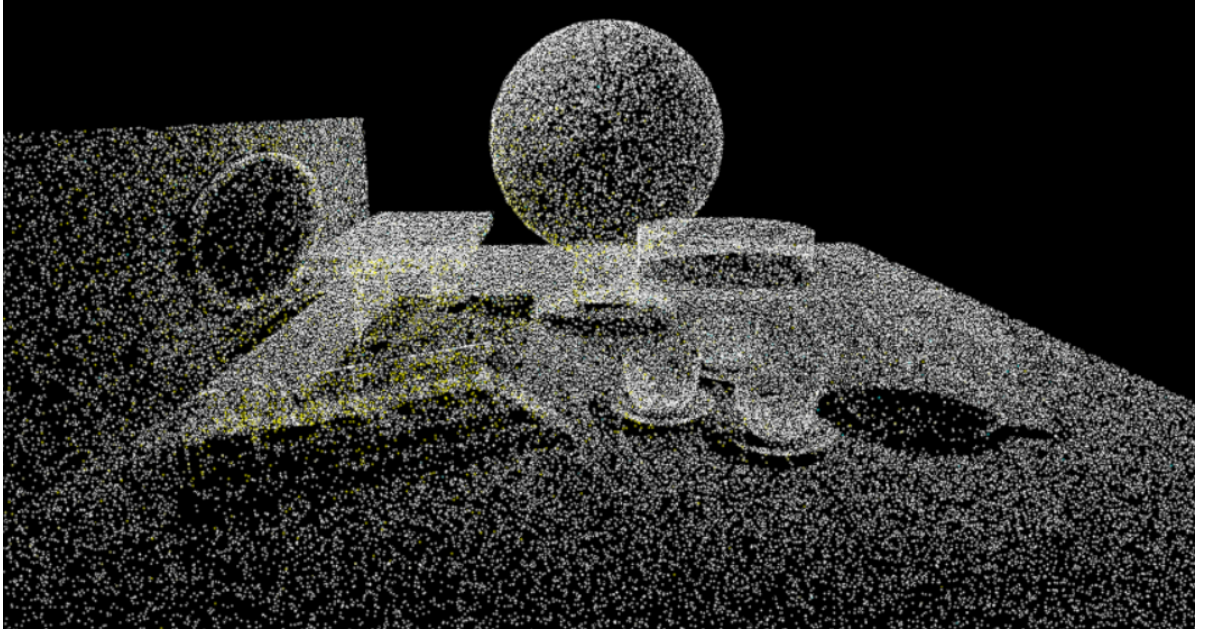
$$\rho_{avg} = \frac{\rho_{red} + \rho_{green} + \rho_{blue}}{3} \quad (5.9)$$

Na základě průměrů platí vztahy určující typ odrazu, kde  $X$  je náhodné číslo s rovnoměrným rozložením od 0 do 1.

$$\begin{aligned} 0 \leq X \leq \rho_{d,avg} &\rightarrow \text{difuzní odraz} \\ \rho_{d,avg} \leq X \leq \rho_{d,avg} + \rho_{s,avg} &\rightarrow \text{zrcadlový odraz} \\ \rho_{d,avg} + \rho_{s,avg} \leq X \leq 1 &\rightarrow \text{foton je pohlcen} \end{aligned} \quad (5.10)$$

Intenzita odraženého fotonu je snížena dle následujícího vztahu, kde  $\Phi_i$  je příchozí světelný tok a  $\rho$  barva tělesa.

$$\Phi_{out} = \Phi_i \rho / \rho_{avg} \quad (5.11)$$



Obrázek 5.7: Globální fotonová mapa.

Při difuzním odrazu nebo pohlcení dochází k uložení fotonu do fotonové mapy, která je implementována podle Henrika Jensena. Na obrázku 5.7 je fotonová mapa vytvořena emitováním 100 000 fotonů.

Jakmile jsou všechny fotony vygenerovány je fotonová mapa vybalancovaná k dosažení optimální složitosti vyhledávání nejbližších fotonů v okolí hitpointu. Z nalezených fotonů je získán světelný tok, který je akumulován do pomocné proměnné `accuReflectedFlux` ve struktuře hitpointu.

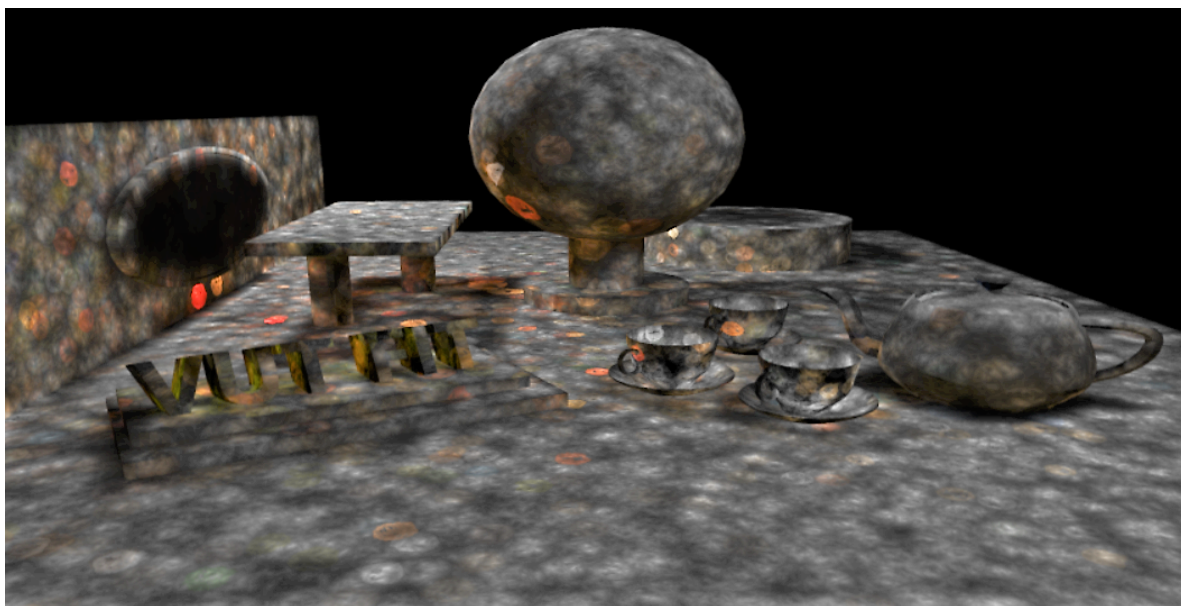
### Vykreslení scény

Po poslední iteraci je z akumulovaného světelného toku  $\tau(x, \vec{\omega})$  vypočtena radiance pomocí rovnice 5.12, kde  $N$  je celkový počet vygenerovaných fotonů. K získání barvy zkoumaného bodu je radiance vynásobena s barvou uloženou v hitpointu.

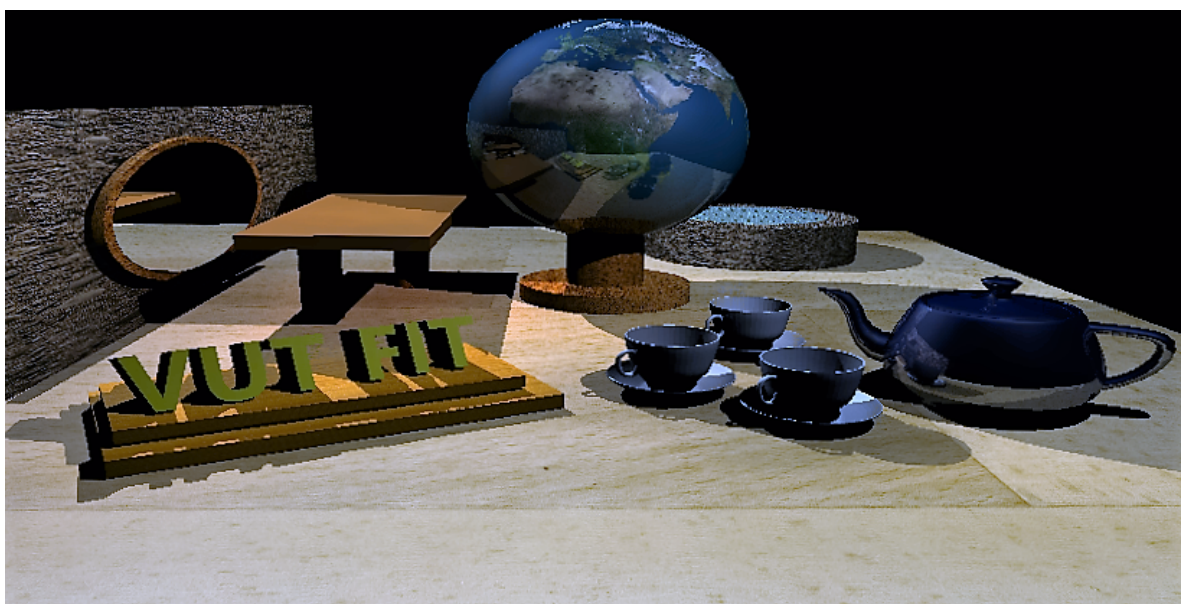
Hodnota radiance scény je zobrazena na obrázku 5.8.

$$L(x, \vec{\omega}) = \frac{1}{\pi R(x)^2} \frac{\tau(x, \vec{\omega})}{N} \quad (5.12)$$

Výsledná barva jednoho hitpointu je přičtena k aktuální barvě pixelu na pozicích v průmětně, kde původně prošel primární paprsek. Finální výsledek syntézy hitpointů je na obrázku 5.9.



Obrázek 5.8: Získaná radiance



Obrázek 5.9: Výstup z foton mapingu.

## Kapitola 6

# Závěr

Cílem této bakalářské práce bylo seznámit se a získat praktické zkušenosti v počítačové grafice, zejména v oblasti tvorby digitálních scén a jejich realistickém zobrazení. Cílem bylo naprogramovat a vyzkoušet konkrétní metodu, která se využívá k fotorealistickému vykreslení.

Na začátku této práce byly nastudovány obecné principy vytváření digitálních těles a jejich vyobrazení na displeji počítače. Následně byly získány informace o fyzikálním chování světla v prostoru a možnosti současných metod pro fotorealistické zobrazení. Jedna z metod byla navržena a naprogramována. Výsledek byl vyzkoušen na testovací scéně, která je zobrazena v rámci kapitoly týkající se popisu této práce.

Tato práce pro mě znamená velký přínos. V současné době se věnuji vývoji informačních systémů. O počítačovou grafiku se intenzivněji zajímám posledním rokem a to i díky této bakalářské práci, která mě posunula k vývoji počítačových her a virtuální scénografie.

Během bakalářského studia jsem se většinou setkával s praxí v oboru spojeným s vývojem informačních systémů a webů. Po dokončení práce si představuji, že bych se ucházel o pozici programátora ve firmě zabývající se počítačovou grafikou.

Vytvořený program bych chtěl dále rozvíjet v optimalizaci. Zajímavé by mohlo být aplikaci převést z jedno vláknové na více vláknovou. Zejména na moderních počítačových grafikách. Převedením výpočtů na grafickou kartu by znamenalo výrazné zrychlení vykreslování, jelikož v jednom čase lze vypočítat několikanásobně více vzorků. Dalším krokem bych chtěl program změnit z konzolové aplikace na okenní s grafickým uživatelským rozhraním. S tím souvisí potřeba rozdělení zdrojových kódů na více projektů, kde jeden z nich by obsahoval pouze zobrazovací jádro.

# Literatura

- [1] Appel, A.: Some techniques for shading machine renderings of solids. *Proceedings of the April 30–May 2, 1968*: s. 37–48, ISSN 00010782, doi:10.1145/1468075.1468082.  
URL <http://portal.acm.org/citation.cfm?doid=1468075.1468082>
- [2] Bedřich, B.; Sochor, J.; Felkel, P.; aj.: *Moderní počítačová grafika*. COMPUTER PRESS, 2004, ISBN 80-251-0454-0.
- [3] Hachisuka, T.; Ogaki, S.; Jensen, H. W.: Progressive photon mapping. *ACM Trans. Graph.*, ročník 27, č. 5, 12 2008: s. 130:1–130:8.
- [4] Heckbert, P. S.: Adaptive radiosity textures for bidirectional ray tracing. *ACM SIGGRAPH Computer Graphics*, ročník 24, č. 4, 8 1990: s. 145–154, ISSN 00978930, doi:10.1145/97880.97895, [Online].  
URL <http://portal.acm.org/citation.cfm?doid=97880.97895>
- [5] Jensen, H. W.: *Realistic Image Synthesis Using Photon Mapping*. Natick, MA, USA: A. K.Peters, Ltd., 2001, ISBN 1-56881-147-0.
- [6] Jensen, H. W.: Caustics. 2002, [Online; navštíveno 12.4.2016].  
URL <http://graphics.ucsd.edu/~henrik/images/caustics.html>
- [7] Jensen, H. W.; Christensen, N. J.: Efficiently rendering shadows using the photon map. *Edugraphics + Compugraphics Proceedings*, 1995: s. 285–291.
- [8] Kajiya, J. T.: The rendering equation. *ACM SIGGRAPH Computer Graphics*, ročník 20, č. 4, 1986: s. 143–150, ISSN 00978930, doi:10.1145/15886.15902.  
URL <http://portal.acm.org/citation.cfm?doid=15886.15902>
- [9] Nicodemus, F. E.: Directional Reflectance and Emissivity of an Opaque Surface. *Applied Optics*, ročník 4, č. 7, 1965: s. 767–775, ISSN 0003-6935, doi:10.1364/AO.4.000767.  
URL <https://www.osapublishing.org/abstract.cfm?URI=ao-4-7-767>
- [10] Peter, S.; Keith, M. R.: *Realistic ray tracing. 2nd ed.* Natick, MA, USA: A. K.Peters, Ltd., 2003, ISBN 1-56881-198-5.
- [11] Phong, B. T.: Illumination for computer generated pictures. *Communications of the ACM*, ročník 18, č. 6, 1977: s. 311–317, ISSN 00010782, doi:10.1145/360825.360839.  
URL <http://portal.acm.org/citation.cfm?doid=360825.360839>
- [12] Rosisqy, C.: Color Bleeding. 2010, [Online; navštíveno 11.4.2016].  
URL <http://furnituredesignhome.blogspot.cz/2010/05/color-bleeding.html>

- [13] SZZ: Transformace, reprezentace a zobrazení 3D objekt. 2016, [Online; navštíveno 6.3.2016].  
URL [http://szz.g6.cz/doku.php?id=temata:12-3d\\_objekty:main](http://szz.g6.cz/doku.php?id=temata:12-3d_objekty:main)
- [14] Whitted, T.: An improved illumination model for shaded display. *Communications of the ACM*, ročník 23, č. 6, 1980: s. 343–349, ISSN 00010782, doi:10.1145/358876.358882.  
URL <http://portal.acm.org/citation.cfm?doid=358876.358882>
- [15] Wikipedia: Phong reflection model. 2006, [Online; navštíveno 17.3.2016].  
URL [https://en.wikipedia.org/w/index.php?title=Phong\\_reflection\\_model&oldid=67375642](https://en.wikipedia.org/w/index.php?title=Phong_reflection_model&oldid=67375642)
- [16] Wikipedia: Voroného diagram. 2014, [Online; navštíveno 2.5.2016].  
URL [https://cs.wikipedia.org/wiki/Voron%C3%A9ho\\_diagram](https://cs.wikipedia.org/wiki/Voron%C3%A9ho_diagram)
- [17] Wikipedia: Polygon mesh. 2016, [Online; navštíveno 6.3.2016].  
URL [https://en.wikipedia.org/wiki/Polygon\\_mesh](https://en.wikipedia.org/wiki/Polygon_mesh)
- [18] Wikiwand: Utah teapot. 2016, [Online; navštíveno 4.3.2016].  
URL [http://www.wikiwand.com/en/Utah\\_teapot](http://www.wikiwand.com/en/Utah_teapot)

# Přílohy

## Seznam příloh

### A Obsah CD

37



## Příloha A

### Obsah CD

- zdrojové soubory programu
- spustitelný program
- testovací scény
- textury a obrázky
- konfigurační soubor
- soubor s popisem ovládání programu