FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	Věnná města českých královen II. – úprava textur
Student:	Jan Tislický
Vedoucí:	Ing. Radek Richtr, Ph.D.
Studijní program:	Informatika
Studijní obor:	Webové a softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce letního semestru 2020/21

Pokyny pro vypracování

Projekt věnných měst českých královen má za cíl vytvoření historicky věrného, virtuálního modelu těchto měst napříč časovou osou od 14. století do současnosti.
Pro zvýšení věrohodnosti virtuálního modelu pro použití ve virtuální realitě budou textury modelu upravovány v závislosti na proměnlivých podmínkách okolí (např. sníh, námraza, poškození bahnem atp.).

1) Proveďte rešerši alespoň čtyř vhodných možností úpravy textur, zaměřte se na fyzikální věrohodnost, stávající praktiky, použitelnost v projektu VMČK a state-of-the-art články.
2) Analyzujte jejich použití a využití vzhledem k jejich budoucímu použití v projektu věnných měst (výkon, přenos, mobilní zařízení).
3) Navrhněte a následně implementujte jako zásuvné moduly do Blenderu alespoň dva z vámi analyzovaných vlivů.
4) Aplikujte změny textur na reálné modely, otestujte a diskutujte jejich kvalitu a věrohodnost.

Seznam odborné literatury

Dodá vedoucí práce.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

doc. RNDr. Ing. Marcel Jiřina, Ph.D.
děkan

V Praze dne 24. září 2019

Poděkování

Chtěl bych zde poděkovat svému vedoucímu Ing. Radku Richtrovi, PhD., za veškerou podporu během studia i psaní této bakalářské práce. Jeho vedení a konstruktivní komentáře k celé práci byly velkým přínosem a posouvaly práci vpřed velkými skoky. Dále bych také rád poděkoval Ing. Elišce Šestákové za konzultace a návrhy na zlepšení některých částí této práce a také za konzultace ohledně dalších částí studia na FIT ČVUT v Praze. Za jazykovou kontrolu práce patří můj dík Mgr. Haně Davídkové, která mi pomohla práci opravit tak, aby byla správně po jazykové stránce. Závěrem bych chtěl poděkovat všem mým spolužákům a rodině za ohromnou podporu v začátcích, během a hlavně ke konci mého studia.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo"), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 6. ledna 2020 .....

Odkaz na tuto práci

Tislický, Jan. *Věnná města českých královen II. – úprava textur*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2020.

Abstrakt

Tato práce řeší několik přírodních fenoménů, které mohou působit na budovy v běžném životě. Tyto efekty jsou převedeny do zásuvného modulu pro 3D grafický program Blender. Tento modul následně vytváří textury pro využití ve virtuální realitě. Tyto textury jsou vytvářeny z materiálu Cycles. Výsledný zásuvný modul využívá například skalární součin či několik matematických nebo grafických metod. Modul je schopný vytvářet několik efektů počasí a z nich následně vytvořit textury pro jeden celý rok. Kromě zásuvného modulu zde lze nalézt popis zadávacího skriptu, který umožňuje automatizovat vytváření textur případně nastavovat celé prostředí Blenderu pro další práci.

Klíčová slova zapékání textur, variabilní úprava textur, Blender, zásuvný modul, blednutí barev, zarůstání, roční období, déšť, Cycles, 3D grafika

Abstract

This thesis focuses on a few nature phenomena that acts on buildings in everyday life. These effects are converted into a plug-in for 3D graphical tool Blendner. This plug-in can create textures for virtual reality usage. Textures created by this plug-in are based on Cycles materials. Final version of the plug-in uses for example dot product, several mathematical or graphical methods. Capabalities of this plug-in are to create few weather effects and from these create texture sets for a whole year. Besides this, plug-in has it's own assign script, which enables automatization of texture creating, or adjust interface of Blender for other work with this plug-in.

Keywords texture baking, variable texture alteration, Blender, plug-in, color fading, overgrowing, season, rain, Cycles, 3D graphics

Obsah

Seznam obrázků

Úvod

Projekt věnných měst českých královen má za cíl vytvoření historicky věrného, virtuálního modelu těchto měst napříč časovou osou od 14. století do současnosti. Pro zvýšení věrohodnosti virtuálního modelu pro použití ve virtuální realitě budou textury modelu upravovány v závislosti na proměnlivých podmínkách okolí.

Jak již bylo řečeno pro zvýšení věrohodnosti virtuálního modelu ve virtuální realitě budou vytvářeny textury, které obsahují různé vlivy počasí. Tedy předmětem zkoumání a následně řešení je vytvoření výpočetního modelu pro napodobení přírodních vlivů. A tento model následně využít v zásuvném modulu pro program Blender.

Motivací pro výběr tohoto téma byl zájem přiučit se něco nového z oblasti texturování modelů. Pak také vytvoření si představy jaké jsou vůbec možnosti práce s 3D grafikou co se týká osvětlení i vlivů počasí. U osvětlení pak zjistit limitace a možnosti globálních osvětlovacích modelů.

Hlavním cílem této práce je najít optimální řešení různých přirozených jevů, které působí na stavby v reálném životě. Ty následně implementovat a provést uživatelské testování. Výsledný modul bude simulovat podmnožinu všech analyzovaných vlivů. Vedlejším cílem je pak i návrh a vytvoření jmenné konvence pro soubory, se kterými bude pracovat zásuvný modul a také vytvoření zadávacího skriptu. Ten pak bude umožňovat částečnou nebo úplnou automatizaci vytváření textur.

Práce je strukturována do tří hlavních částí. V první je rozebráno pět různých efektů, které je možné uplatňovat na modely. Jedná se o efekty blednutí barev 2, zarůstání 4, osvětlení 3, roční období 6 a dešť 5. Druhá část obsahuje návrh jak je možné vytvořit jednotlivé efekty a na co je potřeba dávat pozor. Zde je opět každému efektu věnována samostatná kapitola, následně je zavedena jmenná konvence, se kterou bude zásuvný modul pracovat. Kromě toho je zde také ukázána struktura tříd jak zásuvného modulu, tak zadávacího skriptu. Ve třetí části je popsána implementace celého zásuvného modulu.

Po implementaci následuje shrnutí výsledků uživatelského testování. Poslední zmíněná kapitola také obsahuje ukázky výsledných textur nasazených na příslušný model.

Práce související s touto jsou Věnná města českých královen I. – úprava textur, kde se kolegové Michal Zajíc a Denisa Sůvová zabývali podobnou problematikou s rozdílnými výsledky ve částech, které měli podobné. I tato práce se zabývá některými společnými jevy, ale přidává možnost automatizace celého procesu tvorby textur.

Kapitola 1

Cíl práce

Hlavním cílem této práce je vytvoření plně automatizovatelného zásuvného modulu, který usnadní práci na projektu věnných měst českých královen (dále jen projekt). Zásuvný modul bude vytvářet několik efektů, které se mohou běžně vyskytnout. Celý zásuvný modul bude vyvinut v jazyce Python s pomocí knihovny pro Blender API a dalších potřebných knihoven. Modul bude přístupný po instalaci a umožní úpravy různých modelů i mimo projekt. Modulu bude upravovat textury tak, aby působily, jako by byly vystaveny určitému vlivu počasí. Při mapování jednotlivých efektů budou přidány i poznatky pro možnost použití pro projekt.

Vedlejším cílem je návrh a implementace zadávacího skriptu, který bude umožňovat snadnější přístup. Tento cíl je motivován možností automatizace tvorby textur pomocí příkazové řádky. S pomocí tohoto sktiptu bude možné spouštět zásuvný modul v neinteraktivní podobě dle parametrů.

Kapitola 2

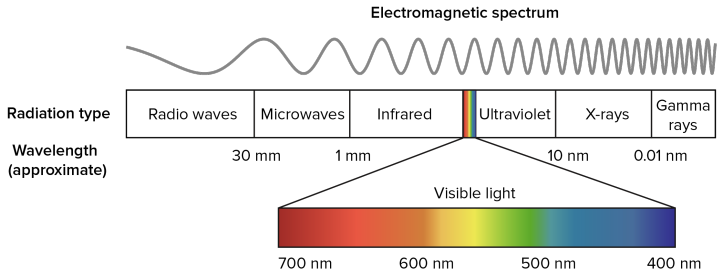
Blednutí barev

Blednutí barev je přirozený jev, který se v přírodě objevuje. Tato skutečnost jen není dostatečně dobře vidět z důvodu regenerace živých organismů, rostlin atd. Ovšem pokud se podíváme na syntetické látky, barvy a podobné materiály, které vyrábí a barví člověk, tak tento jev nalezneme.

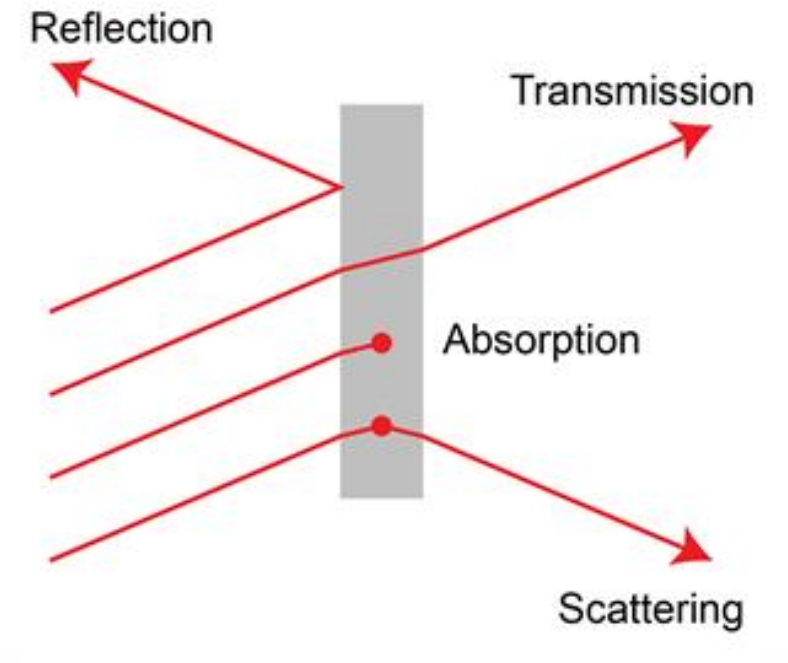
Z fyzikálního [2, 3] hlediska se světlo skládá z mnoha složek, pro tuto problematiku se nejvíce hodí zmínit ultra fialovou složku obvykle značenou krátce UV, která se pohybuje v rozmezí 100 nm až 320 nm, ještě je vhodné zmínit viditelnou část 390 nm až 700 nm.

Viditelné spektrum udává barvu předmětu, na který je pozorován, dle pohlcených, odražených a lomených částí spektra z paprsku, který dopadl na nějaký předmět. Zde lze rovnou říci, že některé barvy jsou více náchylné na blednutí a ztrácí svou barevnost rychleji než jiné. Je to právě složka UV, která má zásadní vliv na samotné blednutí barev. Více je uvedeno v následující části, která tuto otázku řeší z chemického hlediska.

Z chemického [5, 6] hlediska je blednutí barev poměrně jednoduchý proces. Barvy a pigmenty pohlcují určité části světla, které na ně dopadá, část se odráží a část se přenáší, jak již bylo řečeno, po pohlcení fotonu se mohou



Obrázek 2.1: Rozvržení světelného spektra [1]



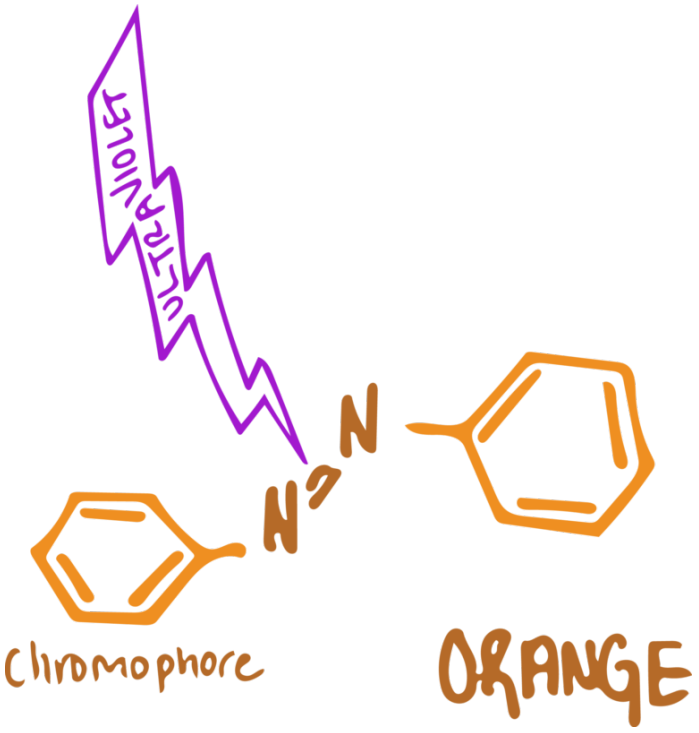
Obrázek 2.2: Reakce světla na povrch hmoty [4]

elektrony excitovat do vyšších energetických stavů. Ve většině případů dojde k vybití tohoto náboje, je tu ale také možnost, že díky zvýšené energetické hodnotě dojde k chemické reakci. Kvůli této reakci dojde ke zrušení kovalentní vazby nebo k jiné nevratné reakci s jinou molekulou. Toto změní strukturu molekuly, zejména její absorpční vlastnosti, a tím pádem může začít pohlcovat jiné části viditelného spektra, jinými slovy, může změnit barvu. Zde během zmíněné absorpce přichází v úvahu zmíněné UV záření. To může způsobit rozpad některých vazeb, a tedy i nezvratné reakce. Kromě blednutí barev může v těchto chemických reakcích dojít k zežloutnutí některých materiálů. Toto je nejvíce vidět na papírových novinách, či starých knihách.

2.1 Dřívější zpracování problematiky

První práce [8], která se více zabývala touto problematikou, je z oblasti konzervace. Giles a McKay [9, 10] studovali mechanismy a faktory, které ovlivňují blednutí pigmentů a barev. Jejich zjištění bylo, že materiály, které mají rovnoměrnou strukturu barvy, vyblednou mnohem rychleji než ty, které mají barvu se shlukující strukturou.

Johnston-Feller et al. [11] studovali rychlost vyblednutí tenkých filmů s několika pigmenty. Tyto pigmenty vystavili různým úrovním osvětlení. Tento ex-



Obrázek 2.3: Chemická reakce oranžové barvy za přítomnosti UV záření [7]

periment přinesl důležitý poznatek, blednutí barev se řídí kinematikou prvního řádu, což je rychlost ztráty pigmentu za čas s ohledem na koncentraci.

Jiný druh zkoumání provedli Shi et al. [12] a Shi a Lu [13]. Zde bylo využito interpolace spline křivek[1] v barevném prostoru RGB. Tento způsob byl vytvořen za účelem vizualizace uměleckých děl, která už prošla procesem blednutí. Jako vstup je požadován počáteční a konečný stav. Tato technika však měla několik úskalí, např. výsledek nebyl vždy vizuálně uspokojivý, nedokázal správně vypočítat výsledek pro delší vystavení materiálu světlu. Navíc tato technika nebyla navržena pro sledování vývoje blednutí pod hypotetickými podmínkami.

Další zpracování provedl Berns et al. [14]. Zde byla vytvořena technika pro digitální restaurování maleb. Jako počáteční hodnota byl vzat degradovaný pigment a jeho nedegradovaný předchůdce. Tyto dva vzorky byly použity pro simulaci různých stavů daného pigmentu. K této simulaci byla použita teorie Kubelka-Munk [15].

[1]Aproximace křivky, která může být definována spojitou množinou souřadnic bodů, kterými má procházet.

2.2 Správné řešení

V této sekci bude popsáno, jak je možné správně řešit blednutí barev z fyzikálního hlediska. Naprosto korektní model by musel umět vzít v potaz chemické složení materiálu, jeho hustotu v různých částech, intenzitu světla, vlnovou délku dopadajícího světla, teplotu, vlhkost a mnoho dalšího. Výpočetní model uvedený v článku o barevných změnách vyvolaných působením světla [8] je schopen pokrýt větší část z požadovaných proměnných. Výpočetní model uveden v tomto článku zde bude rozebrán v následujících sekcích. Většina z prací zmíněných v článku [8], kapitola dva, je založena na Kubelka-Munk teorii [15], která je zjednodušeně popsána v článku [15]. Na této teorii je také vystavěn výpočetní model v článku [8]. Další sekce se zabývají řešením z článku [8], které je rozděleno do teoretické a praktické části. Nutno podotknout, že rozebírané řešení není jediné možné.

2.2.1 Popis teoretické části frameworku

Při řešení tohoto problému v té nejjednodušší formě lze problematiku fotorealistické vizualizace materiálu rozdělit na dva případy. V prvním případě je materiál, který je zkoumán, homogenní. Ve druhém více komplikovaném případě se jedná o tzv. kompozitní materiál.

Homogenní deska S použitím následujícího značení E^d je hustota toku světla směrem dolů a E^u je opět hustota toku světla směrem nahoru. Dále μ označuje koeficient útlumu, μ^a označuje koeficient absorpce a μ^s je koeficient rozptylu. Označení dz indikuje hloubku v ose Z.

Poté, co světelný tok urazí vzdálenost dz, je tok zeslaben o absorbci $\mu^a E^d dz$ a rozptyl $\mu^s E^d dz$, zároveň je však mírně zesílen díky odchodu světla z materiálu $\mu^s E^u dz$. Z předchozího vyplývají diferenciální rovnice

$$E_z^d = -(\mu^a + \mu^s)E^d + \mu^s E^u \ , \tag{2.1}$$

$$-E_z^u = -(\mu^a + \mu^s)E^u + \mu^s E^d \ , \tag{2.2}$$

kde E_z^d a E_z^e označují parciální derivace E^d a E^u vzhledem k hloubce z.

K docílení efektu světelné expozice je nutné vzít v úvahu dobu, po kterou se daný předmět nacházel na světle. Zde to znamená použít čas jako parametr pro rovnice 2.1 a 2.2 ve všech komponentách. Diferenciální rovnice poté musí být odvozeny tak, aby popisovaly, jak se $\mu^a(z,t)$ a $\mu^s(z,t)$ mění s dobou vystavení světlu. Výsledný systém diferenciálních rovnic 2.1 a 2.2 spojený s rovnicemi popsanými v předchozí větě popisuje efekt světelné expozice. Takový systém musí být vyřešen a k tomu je zapotřebí použít různé matematické metody.

Světelný tok v materiálu je definován jako

$$F(\vec{r}) = \int_{S^2} L(\vec{r}, \omega) d\omega \ , \tag{2.3}$$

Obrázek 2.4: V této ukázce jsou různá stádia zbarvení novin. Malý obrázek vlevo nahoře ukazuje původní zbarvení. Vpravo je vidět barevná degradace novinového papíru. Velký obrázek vespod ukazuje noviny po otevření. Získáno z [16]

kde $L(\vec{r}, \omega)$ označuje osvětlení na pozici $\vec{r}$ ve směru $\omega = (\theta, \phi)$. Hustota absorbované energie v určitém bodě materiálu je výsledek toku $\Phi = Fdt$ a koeficientu absorpce μ^a. Dále se definuje β jako objem hmoty, která spotřebuje dané množství energie. Změna koncentrace objemu f barvy je tedy

$$df = -\beta \mu^a F dt \ . \tag{2.4}$$

S použitím rovnic 2.1, 2.2 a 2.4 vznikne následující systém diferenciálních rovnic

$$\begin{aligned} E_z^d &= -(\mu^a + \mu^s)E^d + \mu^s E^u \ , & (2.5)\\ -E_z^u &= -(\mu^a + \mu^s)E^u + \mu^s E^d \ , & (2.6)\\ f_t &= -\beta \mu^a F \ . & (2.7) \end{aligned}$$

Z předchozích rovnic vyplývají určité počáteční podmínky

$$\begin{aligned} \mu^a(z,0) &= \mu_0^a \ , & (2.8)\\ \mu^s(z,0) &= \mu_0^s \ , & (2.9)\\ E^d(0,t) &= E_0 \ , & (2.10)\\ E^u(z_{max},t) &= \rho^* E^d(z_{max},t) \ . & (2.11) \end{aligned}$$

Zde μ_0^a a μ_0^s jsou počáteční absorpční a rozptylové koeficienty materiálu, E_0 je ozáření přicházející shora, z_{max} je hloubka materiálu a ρ^* je odrazivost sloučeniny, která vznikla vystavením světlu.

Kompozitní materiál Za předpokladu, že materiál není homogenní, respektive jeho struktura zbarvení je výsledek různých pigmentů či barev, je potřeba některé předchozí koeficienty upravit. Koeficienty jednotlivých částí jsou označeny jako μ_j^a respektive μ_j^s. Celkový absorpční a rozptylový koeficient je tedy

$$\begin{aligned} \mu^a &= \sum_j \mu_j^a \ , & (2.12)\\ \mu^s &= \sum_j \mu_j^s \ . & (2.13) \end{aligned}$$

Koeficienty kompozitního materiálu jsou použity na zjištění změn ve světelném toku s hloubkou z. Každá složka materiálu, myšleno barevná složka, snižuje objem chemických sloučenin. Pro každé barvivo je tato změna úměrná energii, kterou dané barvivo absorbovalo. A tedy platí $df_j = -\beta_j \mu_j^a F dt$ pro $j = 1, \ldots, m$.

Pokud má nějaký materiál koeficient absorpce $\mu^a = 0$, pak takový materiál nebude měnit barvu. Toto je konzistentní se zákonem zachování energie. Pokud materiál, který jednou částí absorbuje, ale nerozptyluje, a další částí, která neabsorbuje, ale rozptyluje, pak takový materiál nebude měnit barvu se změnou v čase.

Spektrální kontext Pokud se světlo použije v širším pohledu, pak je potřeba použít i jeho spektrální složení[2]. Pro spektrální kontext platí to samé, co je popsáno výše. Objem chemických sloučenin je snížen úměrně k absorbované energii přes celé spektrum. V takovém případě se koeficient μ_j^a stane závislý na vlnové délce. Světelný tok je nahrazen spektrálním světelným tokem F_λ. Objemová rychlost blednutí β_j je úměrná počtu fotonů, které jsou absorbovány, a proto je lineárně úměrná vlnové délce světla [17]. Aby vznikl efekt blednutí barev, musí mít foton dostatek energie k vyvolání excitace. Proto musí být vlnová délka menší než maximum λ_{max}. Z předchozího vyplývá následující rovnice.

$$df_j = -dt \int_0^{\lambda_{max}} \beta_j \mu_j^a F_\lambda d\lambda \tag{2.14}$$

Barevné produkty z chemických reakcí V některých případech se může stát, že chemické látky vzniklé z absorbování energie nemusí být bezbarvé. Jako příklad se dá uvést papír, který žloutne po vystavení světlu. Tento jev je způsoben různými chemickými látkami, které vznikají při štěpení barviva, a musí se s ním počítat.

2.2.2 Diskretizace teoretického řešení

Pro vyřešení soustav rovnic 2.6–2.11 bude vhodné využít lineární algebry a s její pomocí daný problém vyřešit. Pro tyto účely je nutné nejdříve diskretizovat hloubku z a čas t. Jakmile je tento krok hotov, lze poté řešit problém absorpce v jednotlivých vrstvách, kde jsou koeficienty μ^a a μ^s konstantní.

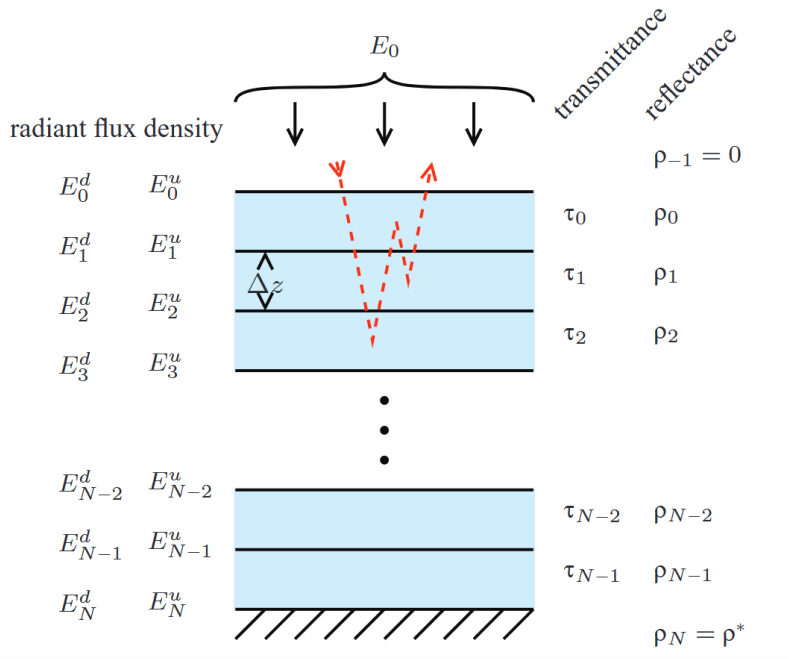
Hloubková diskretizace Pro vyřešení hustoty světelného toku E^d a E^u v rozumném čase je potřeba diskretizovat hloubku z do N vrstev, kde budou koeficienty μ^a a μ^s konstantní. Za těchto předpokladů lze zjistit odrazivost a přenos v každé vrstvě. Toho lze dosáhnout díky Kubelkově formuli [18] pro konečné vrstvy.

$$\rho_i = \frac{1}{a_i + b_i \coth b_i \mu_i^s \Delta z} , \tag{2.15}$$

$$\tau_i = \frac{b_i}{a_i \sinh b_i \mu_i^s \Delta z + b_i \cosh b_i \mu_i^s \Delta z} , \tag{2.16}$$

kde $a_i = \frac{(\mu_i^s + \mu_i^a)}{\mu_i^s}$ a $b_i = \sqrt{a_i^2 - 1}$. Jelikož každá vrstva má uniformní rozptylové a absorpční koeficienty, je odrazivost a přenos každé vrstvy stejný shora dolů i zdola nahoru. V tomto výpočetním modelu se nezohledňuje další dělení v jednotlivých vrstvách, tím je myšleno odrážení či rozptyl od okolních vrstev. Odrazivost v nekonečnu $\rho_{-1} = 0$ nerozptylující vrchní vrstvy a $\rho_N = \rho^*$ jako odrazivost poslední vrstvy.

[2]Spektrálním složením je myšleno složení vlnových délek daného paprsku.



Obrázek 2.5: Graf ukazující hloubkovou diskretizaci simulovaného média dle dvouproudého modelu. Tloušťka každé vrstvy je označena jako Δz. Získáno z [8]

Pro hodnoty E_i^d a E_i^u platí že $E_i^d = E^d(i\Delta z)$, $E_i^u = E^u(i\Delta z)$, kde se hodnoty i pohybují v rozmezí $0 \leq i \leq N$. Tyto hodnoty vyjadřují odrazivost a přenos ve vrstvách následovně

$$E_0^d = E_0 \ , \tag{2.17}$$
$$E_i^u = \rho_i E_i^d + \tau_i E_{i+1}^u, 0 \leq i < N \ , \tag{2.18}$$
$$E_i^d = \tau_{i-1} E_{i_1}^d + \rho_{i-1} E_i^u, 0 < i \leq N \ , \tag{2.19}$$
$$E_N^u = \rho_N E_N^d \ . \tag{2.20}$$

Tyto rovnice se dají převést na matice

$$\mathbf{E} = \mathbf{M}\mathbf{E} + \mathbf{E_0}. \tag{2.21}$$

Zde $\mathbf{E}$ a $\mathbf{E_0}$ jsou $2(N+1)$ dimenzionální vektory

$$\mathbf{E} = (E_0^u, E_0^d, \ldots, E_N^u, E_N^d)^T \ , \tag{2.22}$$
$$\mathbf{E_0} = (0, E_0, 0, \ldots, 0)^T \ . \tag{2.23}$$

M je blok tridiagonální matice o rozměrech $2(N+1) \times 2(N+1)$

$$\mathbf{M} = \begin{pmatrix} \mathbf{R_0} & \mathbf{T_0^u} & & 0 \\ \mathbf{T_0^d} & \ddots & \ddots & \\ & \ddots & \ddots & T_{N-1}^u \\ 0 & & T_{N-1}^d & R_N \end{pmatrix} \tag{2.24}$$

s bloky 2×2.

$$\mathbf{R}_i = \begin{pmatrix} 0 & \rho_i \\ \rho_{i-1} & 0 \end{pmatrix}, 0 \leq i \leq N \ , \tag{2.25}$$

$$\mathbf{T}_i^u = \begin{pmatrix} \tau_i & 0 \\ 0 & 0 \end{pmatrix}, 0 \leq i < N \ , \tag{2.26}$$

$$\mathbf{T}_i^d = \begin{pmatrix} 0 & 0 \\ 0 & \tau_i \end{pmatrix}, 0 \leq i < N \ . \tag{2.27}$$

Bloková tridia1gonální struktura matice umožňuje vyřešit tento problém v čase $\mathcal{O}(N)$. Při řešení systému lineárních rovnic se časová složitost rapidně změní na $\mathcal{O}(N^3)$.

Časová diskretizace Čas je zde řešen pomocí rovnice 2.14, kde světelný tok je konstantní v každé vrstvě. Absorpční koeficienty jsou poté aktualizovány dle

$$f(t + \Delta t) = f(t) \exp\left(-\Delta t \int_0^\infty \beta \mu^a F_\lambda d\lambda\right). \tag{2.28}$$

F_λ jako spektrální světelný tok je počítána pro fixní množství vlnových délek. Absorpční koeficient μ^a je vzorkován na stejné množině vlnových délek. Výsledek je numericky zakomponován, a tím pádem může být absorpční koeficient přepočítán jako

$$\mu^a(t + \Delta t) = \frac{f(t + \Delta t)}{f(t)} \mu^a(t) \ . \tag{2.29}$$

2.3 Používané řešení v CG

Tento efekt není běžně řešenou problematikou, jelikož se většina simulací (zejména počítačové hry) odehrává v krátkých časových intervalech. Případně je nechán prostor pro odpověď typu „Barvy vypadají nověji, jelikož v nedávné době došlo k novému natření povrchu.“ Hlavním uplatněním v CG je lepší porozumění danému prostředí. Například stáří předmětu nebo novin.

Když bude potřeba dodat nějaké renderované scéně další rozměr realističnosti, pak je tento efekt nezbytný, pokud člověk nechce ukazovat návrh

Obrázek 2.6: Vlevo je vidět původní zbarvení obrazu, následuje ukázka po ztrátě barevnosti některých pigmentů obrazu. Poslední obrázek je detailní pohled na degradovaný obrázek. Získáno z [8]

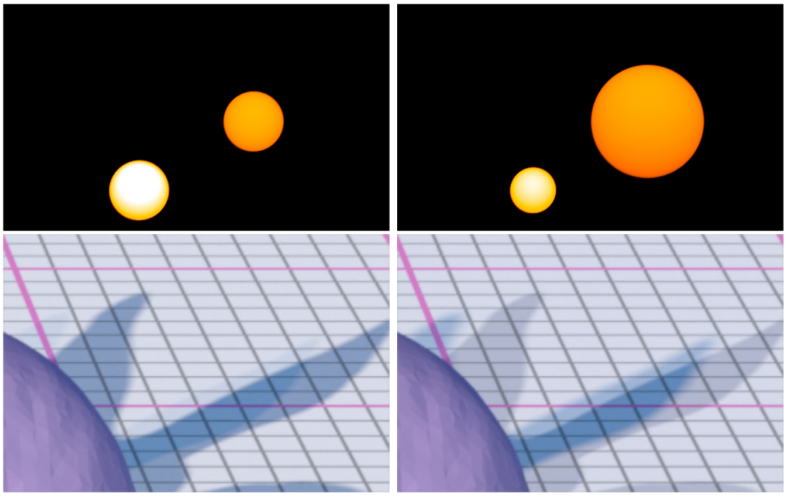
nově vypadající kuchyně, obývacího pokoje atp. V případě, že je potřeba sledovat vývoj vzhledu nějaké místnosti nebo objektu, může být tento efekt vcelku užitečný na ukázání stáří dané místnosti. Taková informace říká mnohem více o sledovaném místě, o tom, kdo zde žil nebo pracoval, kde mohl odpočívat.

2.4 Vhodné řešení pro projekt Věnných měst

V projektu Věnných měst je tento efekt vhodný zejména při dotváření vzhledu historických budov. Po vytvoření veškerých textur, lze postup zmíněný v této kapitole použít pro dotvoření realistického vzhledu. Vždy se nepovede vyladit odstíny perfektně již při tvorbě, proto je občas potřeba použít podobné výpočetní systémy pro vytvoření příslušného efektu. V sekci 7.1 je popsáno navrhované řešení pro projekt Věnných měst. Tento postup je v mnoha ohledech zjednodušen a upraven tak, aby se dal, co nejsnáze použít. Tato část má využití zejména při vytváření vzhledu napříč několika stoletími. Se znalostí skladby materiálu původní budovy a jejího vzhledu je možné ji vytvořit virtuálně a následně tímto postupem simulovat barevnou degradaci. Pak stačí jednotlivé výsledky nastavit jako referenční body a interpolovat mezi nimi.

2.5 Binární sluneční soustava, mimozemské scény

Wilkie v článku [16] uvádí možné využití více sluncí nebo mimozemských scén pro různé výzkumy nebo simulátory. Jak je zřejmé, možnosti lidstva jsou, vzhledem k současnému vesmírnému programu, značně omezené. Navzdory tomu existují různé instituce, které se touto problematikou zabývají (např. NASA). Je možné, že pod vlivem jiné vlnové délky záření z příslušných hvězd vznikly jiné materiály, které pohlcují jiná spektra, než jak tomu je na naší planetě, toto je předmětem diskuse [16]. Za předpokladu, že například kovy budou na jiné planetě stejného typu, jako je například železo, hliník, platina, atp., lze v laboratorních podmínkách zjistit jejich chování a následně sestavit správný model vzhledu pro dané osvětlení, tyto modely však nemusí být



Obrázek 2.7: Ukázka stínu v binární soustavě. Získáno z [16]

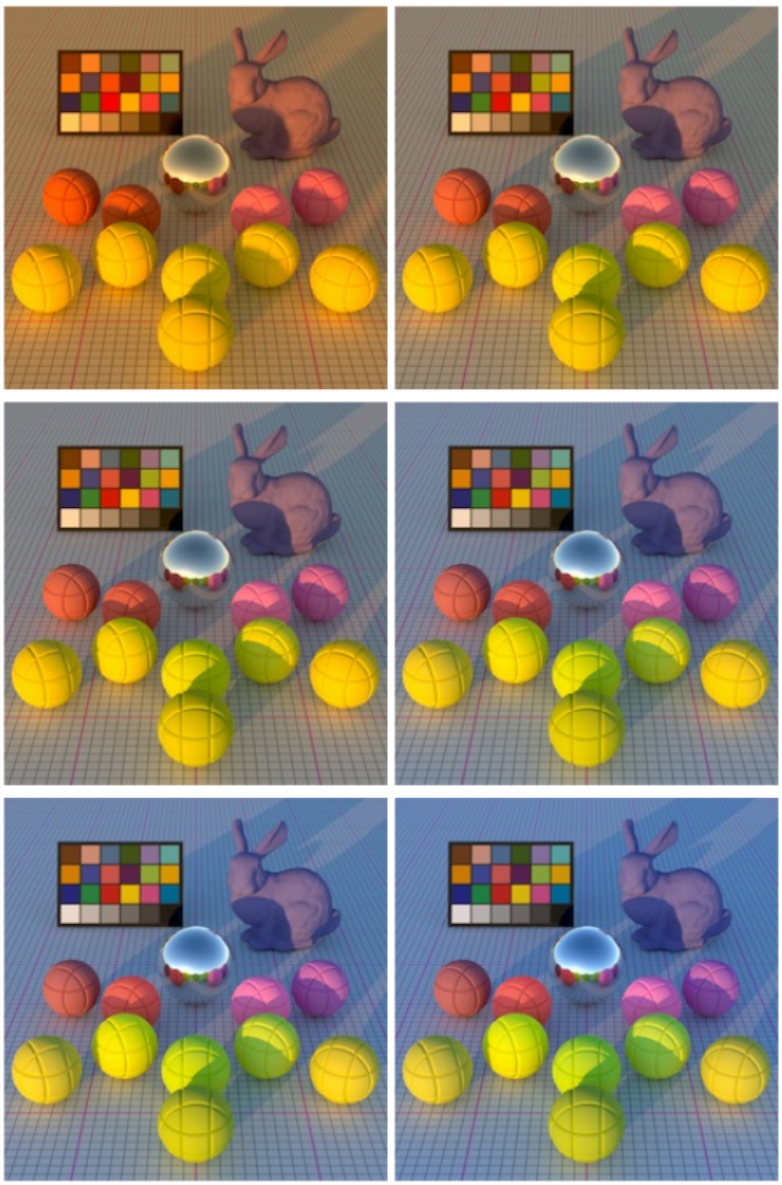
zcela použitelné v počítačové grafice (CG). Hlavním problémem by mohl být výpočetní výkon.

V mnohých ohledech se pro renderování a obecně v CG na jiných planetách ve vesmíru používá model osvětlení podobný tomu, který je známý v pozemském prostředí. Ten nemusí být vždy přesný, jak již bylo zmíněno, buď kvůli materiálům na povrchu (chování), nebo kvůli složení atmosféry dané planety a podobným faktorům. Bez přesného popisu vlastností oblohy, kterou by scéna z planety měla, by bylo velmi obtížné stanovit správná chování a vytvořit přesvědčující výsledný efekt.

V CG, přesněji při renderování, existují tři základní cesty jak vytvořit realistický efekt oblohy.

1. Simulace přenosu světla atmosférou
2. High dynamic range (HDR) snímky prostředí naší planety
3. Analytické modely osvětlení

První možnost, jak již vyplývá z názvu, bude nevhodná, jelikož bude velmi časově náročná (a v době vytváření této práce poměrně těžce realizovatelná). Druhá možnost poskytuje stále velmi omezené možnosti, s touto možností se nedá dostatečně dobře předvídat chování jiných planet, je tedy vhodná pouze pro utváření scén v našich podmínkách. Třetí možnost také není obecně použitelná na jiné planety. Většina modelů zahrnuje mnoho předpočítaných dat pro rozptyl světla a podobné efekty atmosféry. To pomáhá při vyhýbání se hlavnímu problému, kterým je výpočetní čas. Předpočítané hodnoty, které jsou zmíněny výše, jsou specifické pro atmosféry zemského typu (a slunce



Obrázek 2.8: Osvětlení při různých teplotách hvězdy. Získáno z [16]

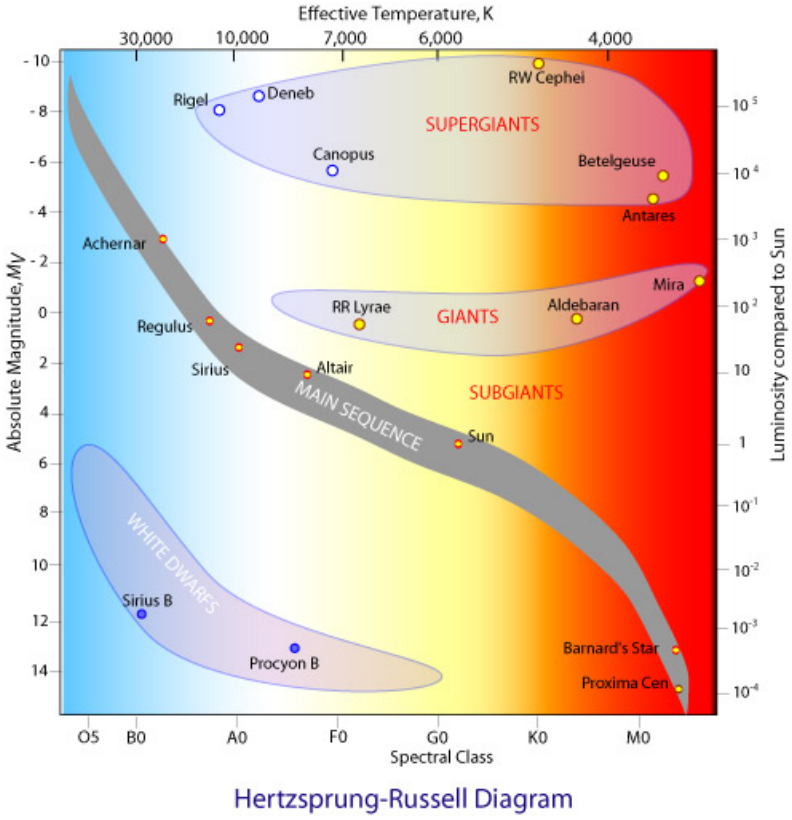
podobné tomu našemu), ale tyto hodnoty jsou neužitečné pro vytváření scén z jiného než zemského prostředí.

Je zde jedna výjimka, dva modely počítají sluneční osvětlení pro každou vlnovou délku, a proto může být použito pro vzhled oblohy na planetách zemského typu pod jiným sluncem, ale k ničemu jinému už ne. Jedná se o modely Preetham [19] a Hošek [20].

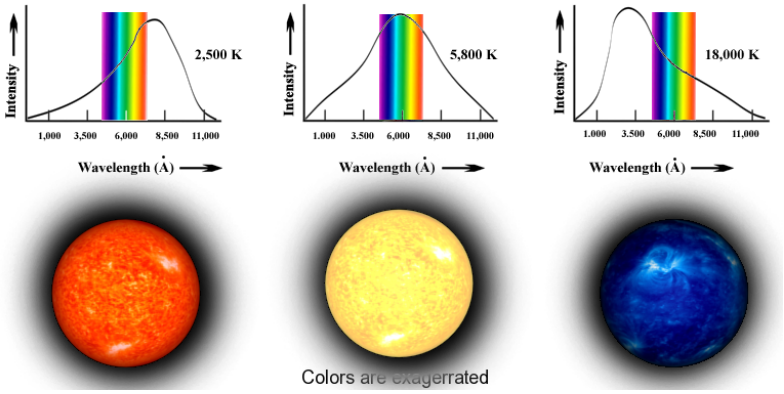
Pro představu je vhodné vzít počítačovou hru nebo simulátor, ve kterém se objevuje planeta jiná než naše, v tom případě je zapotřebí rozmyslet si spoustu věcí. Z pohledu osvětlení má vliv, kromě zmíněné barvy, a tedy i Kelvinovské[3] teploty daného slunce, počet sluncí v daném systému. Je možné, že se bude hodit soustava, která má dvě slunce. Každé slunce může být jiného typu, a tím pádem vyzařuje jiné světlo, což pak ovlivňuje příspěvky jednotlivých sluncí na povrchu. I jedno slunce může mít jinou Kelvinovskou teplotu než naše slunce a to také významně ovlivní materiály na těchto planetách.

Tato část se hlavně zabývá blednutím barev předmětů, které jsou vystavené určitému světlu po určitou dobu. Ačkoli pro projekt Věnných měst tato část nemá valný efekt, je zde uvedena čistě ze zájmu o tuto problematiku. Jak již bylo zmíněno, barvy blednou po vystavení světlu po určitou dobu. Stejný princip se dá použít i pro tuto možnost. Pokud je více než jedno slunce v soustavě, pak se musí brát ohled na každé zvlášť a jejich vliv na přítomné materiály, které mají ve své blízkosti. Tyto materiály přijímají sluneční světlo a struktura jejich barev se postupně mění. Pokud se vezme v úvahu binární sluneční systém. Pak se v takovém případě budou dělit příspěvky dle určitého poměru, který lze v CG volit, a tím se i ovlivní blednutí barev předmětů/materiálů.

[3]Kelvinova teplotní stupnice je využívána zejména v astronomii, s její pomocí lze vyjádřit velké teplotní rozdíly, není záporná a její nula znamená absolutní nulu. Teplota hvězd a jiných těles ve vesmíru se určuje podle této stupnice. Teplota nám udává zbarvení hvězdy a tím nám i říká, jaké spektrum zhruba daná hvězda vyzařuje.



Obrázek 2.9: Diagram ukazující vztahy různých veličin pro hvězdy [21]



Obrázek 2.10: Barevného rozložení různých druhů hvězd [22]

Kapitola 3

Denní doba

Denní doba je v moderních simulátorech a počítačových hrách řešena dynamicky. Světla se pohybují po předem určených drahách a s nimi se mění i vzhled prostředí[4]. U světel se také mění další podstatné faktory jako například barva nebo intenzita. Důležitou roli zde hraje také odražené světlo a umělé osvětlení. Každý model má svou texturu na kterou dopadá světlo, to následně částečně alteruje vzhled.

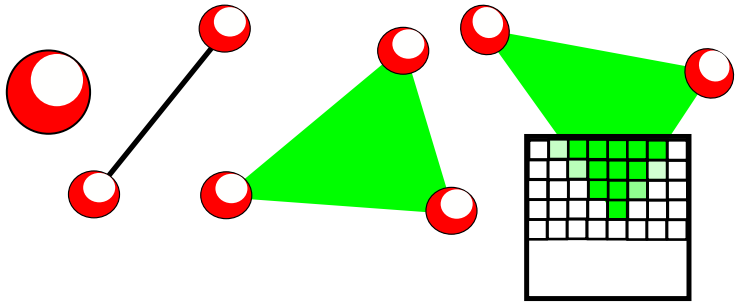
Barva modelu, kromě základní barvy textury, je tedy ovlivněna mnoha faktory, jako příklad lze uvést počet světel, jejich barevné spektrum, vzdálenost od zdroje. Každý světelný zdroj přispívá nějakým způsobem k barevnému vzhledu textury, a tedy i modelu.

V dalších sekcích bude nastíněné správné řešení v CG. Tedy výpočetní modely, používané funkce a jejich popis. Dále popis, jak se v momentální době správné řešení upravuje nebo neupravuje pro potřeby simulací v závislosti na výpočetní síle.

3.1 Světlo ve fyzice a světlo v počítačové grafice

Světlo [23] jako fyzikální jev bylo v průběhu let chápáno různě. Nejdříve bylo považováno za částici nazývanou foton. Toto odpovídalo prvním náhledům na hmotu. Později se začalo ukazovat, že tento náhled není ani zdaleka dostačující a přesný. Další teorie popisuje světlo jako elektromagnetické vlnění, které je vnímatelné lidským okem. Posun k této teorii se ukázal být korektní, ale stále vyvstávaly otázky, na které tato teorie nedokázala odpovědět. Později se začíná uvažovat o tzv. duální povaze světla, což znamená, že světlo je za určitých podmínek vnímáno jako elektromagnetická vlna a za určitých jako částice. Nicméně i tato teorie nebyla korektní. Zatím nejpřesnější a také nejkomplexnější teorie ohledně světla se nazývá kvantová elektrodynamika (an-

[4]Myšleno skybox, volně přeloženo vzhled vzdáleného obzoru.



Obrázek 3.1: Obrazové ukázání pojmů zleva vrchol, hrana, ploška a fragment

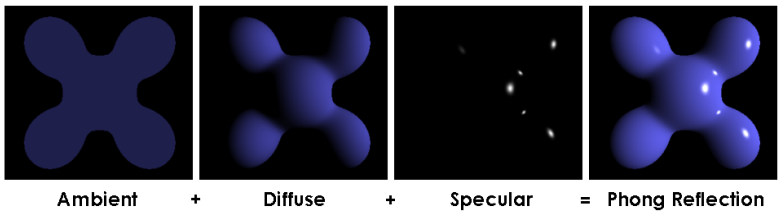
glická zkratka QED). Tato teorie spojuje myšlenky klasického elektromagnetismu, kvantové mechaniky a speciální teorie relativity.

Světlo v počítačové grafice V CG se světlo obvykle nesimuluje přesně pomocí fyzikálních modelů, využívá se pouze aproximace tohoto chování. Jak již bylo zmíněno, světlo může být v CG pojato různě a ne všechny modely výpočtu osvětlení potřebují znát např. energii daného fotonu či jiné fyzikální zákonitosti ohledně světla. Proto je pohled na světlo v mnoha případech zjednodušen na barvu a vektor směru daného fotonu, nebo ještě přesněji vektor směřující od zdroje světla k bodu, který má být zobrazen.

V reálném světě se ze světelného zdroje vysílá nepřeberné množství fotonů, které působí na okolí. Pokud se v simulacích využije stejné skutečnosti, pak by renderování jednoho snímku trvalo velmi dlouhou dobu. Důvod je prostý, pravděpodobnost, že se paprsek vyslaný ze světelného zdroje setká s nějakým tělesem a následně se dostane do oka pozorovatele je velmi malá. Z tohoto faktu vyvstane otázka, jak tedy zjistit osvětlení objektů. Řešení je jednoduché. Místo toho, aby se vysílaly paprsky ze zdroje, tak se budou do scény vysílat z oka. Tam kde se protne paprsek s objektem, se zjistí dle použitého stínovacího a osvětlovacího modelu, co je potřeba. Pokud se paprsek nesetká s ničím, pak se použije barva pozadí. Co je stínovací a osvětlovací model, bude rozebráno v následující sekci.

V CG se běžně operuje s výrazy jako fragment, vrchol, face (ploška) nebo normála. Fragment reprezentuje část rasterizovaného objektu, který je spjatý s pixely. Vrchol je jeden bod v 3D prostoru o souřadnicích x, y a z. Jednotlivé vrcholy jsou spojeny hranami a ze tří vrcholů vzniká ploška. Ta se dá také popsat jako nejmenší rovná plocha. Na každou takovou plochu pak připadá normála, která je vždy kolmá na plošku.

Další velmi časté pojmy jsou různé druhy odrazů například difúzní, spe-



Obrázek 3.2: Phongův osvětlovací model a jeho jednotlivé složky [26]

kulární a glossy. Difúzní odraz je rovnoměrně rozptýlené světlo okolo místa dopadu. Spekulární odraz je jinak označován jako zrcadlový odraz a je to ideálně odražené světlo. Poslední glossy odraz způsobuje lesklé části modelu, jedná se o odraz podobný zrcadlovému, jen nenastal dokonalý odraz světla.

3.2 Osvětlovací a stínovací modely

V CG jsou dva rozdílné pojmy pro zjištění působení světla na objekty. Jedná se o osvětlovací a stínovací modely. Nejjednodušší popis rozdílu těchto dvou pojmů je následující: stínování jsou různé metody jak zjistit barvu objektu, osvětlení říká, jak vypočítat barvu objektu v závislosti na parametrech daného objektu [24].

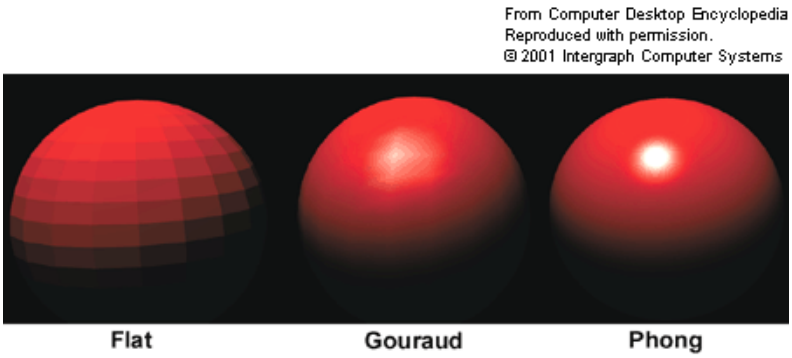
3.3 Stínovací modely

Tyto modely udávají, jak jsou vypočítány odstíny barev pro pixel modelu na základě barvy v každém vrcholu modelu. A také kde přesně se má výpočet aplikovat. Existují tři základní modely výpočtu stínování, přesněji Phongův, Gouraudův a ploškový [25].

Phongovo stínování Phongovo stínování je vylepšení pro získání lepších spekulárních odrazů. Hlavní myšlenka je interpolace normál, které jsou ve vrcholech[5], barva je pak vypočítána pro jednotlivé fragmenty, kde se zohledňuje spočtená normála. Jedná se o vizuálně uspokojivý přístup při zobrazení kulovitého tvaru, silně difúzních povrchů nebo plastů.

Gouraudovo stínování Gouraodovo stínování je způsob vyvinutý zejména pro hladší přechody barev na kulatých nebo hladkých objektech. Hlavní myšlenkou Gouraudova stínování jsou rozdílné normály ve vrcholech. Barva se spočítá v těchto vrcholech a následně proběhne interpolace barvy v plošce

[5]Jedná se o průměr hodnot normál z plošek, které obsahují daný vrchol.



Obrázek 3.3: Porovnání jednotlivých druhů stínování [27]

mezi vrcholy. Výpočet barvy přes vrcholy je více efektivní, protože je méně vrcholů než fragmentů[6].

Gouraudovo stínování není vhodné pro materiály se spekulární složkou, jelikož tento typ stínování zvýrazňuje strukturu, která pak může vytvářet rušivý dojem. Implementace tohoto typu je rychlá a snadná, ale neposkytuje výrazné odrazy rovnoměrně po objektu, pouze v jeho vrcholech.

Ploškové stínování Model ploškového stínování je z předchozích nejjednodušší a nejefektivnější, ale má své nedostatky. Barva je spočtena pro každou plošku zvlášť. Každá z těchto plošek má normálu a ta udá stínování (odstín) dané plošky. Tímto přístupem je možné získat známý low-poly[7] vzhled modelů, na které jsou zkoumány. V mnoha případech je toto však nežádoucí a používá se jiný přístup.

3.4 Osvětlovací modely

Jedná se o techniky, které říkají, jak na základě vlastností scény, objektů a pozorovatele vypočítat výslednou barvu. Osvětlovací modely rozdělujeme do tří skupin. Případně je možné tyto modely dělit na fotorealistické a nefotorealistické.

První je fyzikálně správná, tato skupina pracuje s mnoha proměnnými, jako je například interakce s materiálem nebo vliv prostředí na světlo. Bohužel jsou tyto modely výpočetně extrémně náročné, ne-li nemožné. Proto se využívají jen v ojedinělých případech.

Druhá skupina se nazývá empirická. Tyto modely se snaží co nejvíce aproximovat a zachovat fyzikální chování a zároveň být výpočetně co nejméně

[6]Fragment je množství informace pro vykreslení v jednom pixelu 3.1

[7]low-poly objekt, je takový objekt který má málo geometrie, tedy hrany jednotlivých ploch jsou viditelné

náročné. V mnoha případech tyto modely skutečně stačí a vytváří dostatečně dobré výsledky.

Třetí skupina může být označena jako speciální modely. Tyto modely se využívají pro různé účely a mohou vypadat různě. Jako příklad lze uvést Cel-shading, který simuluje komiksový styl.

Mezi fotorealistické osvětlovací modely patří například Lambertův, Oren-Nayer, Minnaert, Cook-Torrence, Ward anisotropic nebo Subsurface Scattering. Další nefotorealistický model je například Gooch shading.

Phongův osvětlovací model Jedná se o silně empirický, rychlý a dost často vizuálně dostačující osvětlovací model. Skládá se ze tří komponent, což jsou ambientní, spekulární a difúzní odraz.

Phongův osvětlovací model je jednoduše vyjádřen následující rovnicí

$$I_V = I_a + I_s + I_d \ . \tag{3.1}$$

Zde I_V je výsledná intenzita v nějakém určitém bodě. I_a je intenzita ambientní složky, I_s je intenzita spekulární složky a I_d je intenzita difúzní složky. Jednotlivé části I_a, I_s a I_d získáme vyřešením následujících rovnic

$$I_a = I_A r_a \ , \tag{3.2}$$
$$I_s = I_L r_s (\vec{v}\vec{r})^h \ , \tag{3.3}$$
$$I_d = I_L r_d (\vec{l}\vec{n}) \ . \tag{3.4}$$

Vektor $\vec{r}$ lze vypočítat jako $\vec{r} = 2(\vec{l}\vec{n})\vec{n} - \vec{l}$.

Pro více zdrojů světla je potřeba tuto skutečnost zohlednit a započítat reakci na každé světlo zvlášť

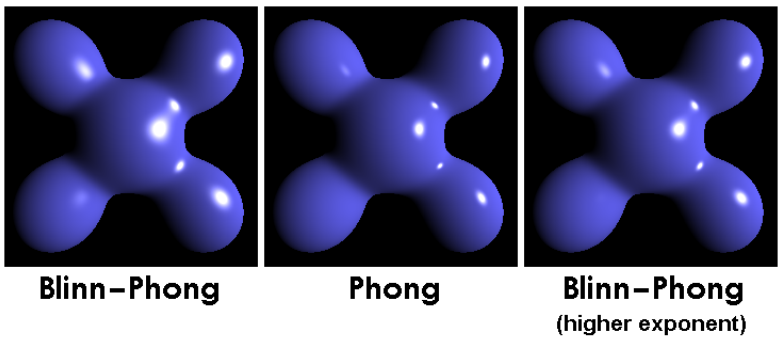
$$I_V = I_A r_S + \sum_{k=1}^{M} I_{L_k} (r_s (\vec{v}\vec{r}_k)^h + r_d (\vec{l}_k \vec{n})) \ . \tag{3.5}$$

Blinn-Phongův osvětlovací model Jak již název napovídá, tento model je nějakým způsobem spojen s Phongovým osvětlovacím modelem. Ve skutečnosti se jedná o zjednodušení klasického Phongova modelu. Tento model je opět o malý kousek blíž realitě, využívá se zde tzv. půl vektoru. Půl vektor je vektor, který je jednotkový a je přesně v půlce úhlu, který svírají dva jiné vektory.

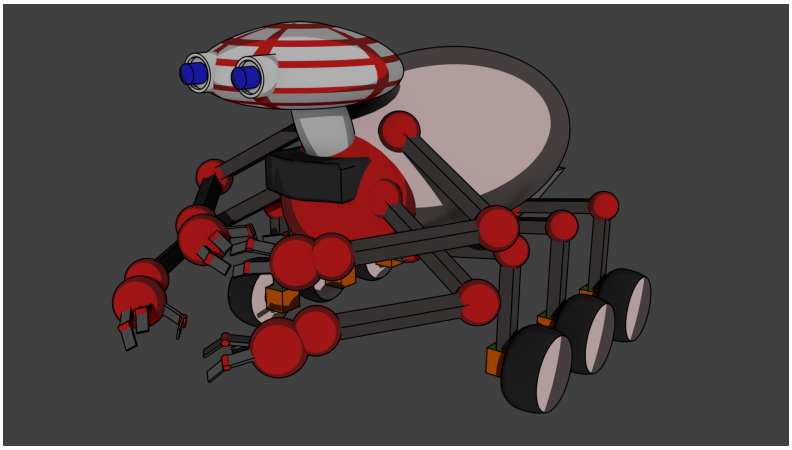
Jediná úprava, kterou je potřeba zavést, je při výpočtu spekulární složky. Zde se využije výše zmíněný půl vektor

$$I_s = I_L r_s (\vec{h}\vec{n})_{h_B} \ . \tag{3.6}$$

Důsledek využití půl vektoru je vidět hlavně ve spekulárních odrazech. Koeficient h je v tomto modelu čtyřikrát větší než ve Phongově modelu pro přibližně stejně velké odlesky. I samotné odlesky se mohou chovat jinak, z některých úhlů mohou vypadat jako ovál.



Obrázek 3.4: Blinn-Phongův osvětlovací model v porovnání s normálním Phongovým osvětlením [28]

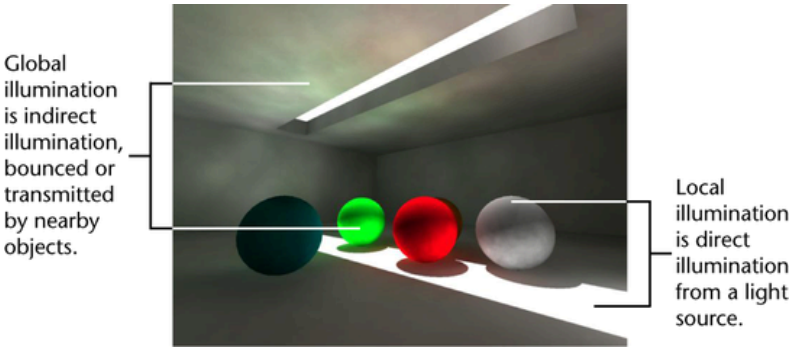


Obrázek 3.5: Ukázka Cel-Shadingu

Cel-Shading Jak již bylo řečeno, tento osvětlovací model pomáhá simulovat komiksový vzhled. Jedná se o nefotorealistický styl renderování. Hlavní využití je ve filmovém průmyslu (animované pohádky) nebo herním průmyslu. Nejčastěji je cílem zobrazení 3D grafiky jako 2D grafiky, ale jsou známé i výjimky jako například počítačová herní série Borderlands. Typický poznávací znak je černá linka na okrajích objektů, linka není striktně na okrajích, ale i všude, kde je nějaká hrana či roh.

3.5 Správné řešení osvětlení

Aby bylo dosaženo co nejvíce uspokojivých výsledků, je potřeba vzít globální osvětlení. Jedná se o skupinu osvětlovacích modelů, které jsou velmi výpočetně



Obrázek 3.6: Rozdíl přímého a nepřímého osvětlení [29]

složité. Stále se jedná o aproximaci, ale ta je fyzikálně uspokojivá. Osvětlení jednotlivých objektů bere v potaz i ostatní objekty, paprsky světla, které se odráží nebo lámou a prochází skrz objekty. Všechny tyto paprsky se využijí k výpočtu osvětlení bodů v simulovaném prostředí. Tímto stylem vznikají barevné nádechy okolních předmětů.

Naproti tomuto osvětlení je tzv. lokální osvětlení. V těchto případech se započítává pouze světlo, které přichází ze světelného zdroje. Nedochází k odrážení ani lomení paprsků. Jelikož se odražené paprsky nezapočítávají, nevznikají barevné nádechy.

První možnost je to, čeho se většina vývojářů snaží dosáhnout, ale ne vždy je to možné. Globální osvětlovací modely jsou velmi komplexní na výpočet a je k nim potřeba přidávat mnoho informací, které se musí dopočítávat v závislosti na aktuálních pozicích pohledů a světel. Během roku 2018 byla představena nová grafická karta společnosti NVIDIA s názvem RTX, celá řada má označení RTX 2000. Tato sada grafických karet už umožňuje využívat real-time[8] Ray tracing[9]. Co je Ray tracing bude vysvětleno níže.

Všechny následující metody počítají globální osvětlovací model. Většina těchto metod funguje tak, že do daného pohledu vystřelují jednotlivé fotony a svým způsobem využívají informaci o odrazu, zlomu nebo absorbci fotonů. Množství fotonů může být velké, platí, že čím více fotonů vystřelíme do scény, tím přesnější vzhled dostaneme.

3.5.1 Ray tracing

Ray tracing [30, 31, 32] je technika, která sleduje nepřeberné množství paprsků světla skrz model. Základní algoritmus sleduje jednotlivé paprsky zpětně od

[8]Použití dané metody v reálném čase.

[9]Z minulosti jsou známy příklady, kdy se některá z metod globálního osvětlení použila, ale samotná hra zobrazovala s 30 FPS což znamená Frames Per Second. Volně přeloženo obrazovky za vteřinu. Jinak řečeno kolikrát se obrazovka vykreslí za jednu vteřinu.

Obrázek 3.7: Porovnání renderu s a bez Ray tracingu [33]

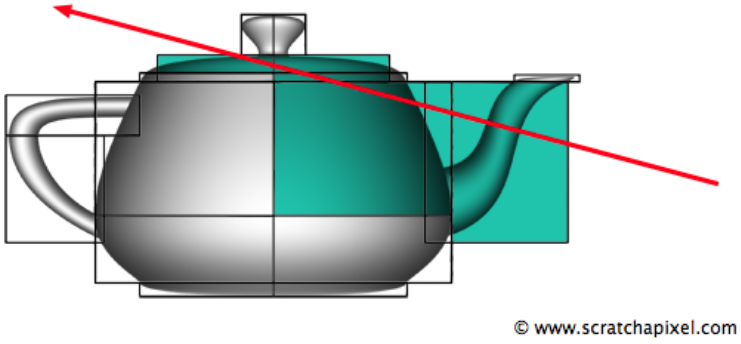
pozorovatele ke světelnému zdroji a následně simuluje interakci s virtuálními objekty. Tento přístup je schopný vytvořit fotorealistické rendery. Věrohodnost může být až tak vysoká, že si člověk může takový render splést se skutečnou scénou.

Jak již bylo zmíněno, algoritmus Ray tracingu sleduje paprsky z oka pozorovatele ke zdroji světla, oko je zde kamera, která sleduje nějaký objekt ve virtuálním 3D světě. Z kamery jsou vystřelovány jednotlivé fotony a sleduje se jejich dráha plus interakce s objekty. Základní implementace by využívala jeden paprsek pro pixel. Zjistilo by se, jaký objekt je tímto paprskem zasažen, a poté by se vypočítalo osvětlení ze zdrojů, které se k danému tvaru dostanou, případně jak se dál paprsek odráží. Takový odraz může generovat další paprsky, které se musí započítat k výslednému osvětlení.

Pro zjištění množství světla, které náleží jednomu pixelu, musí algoritmus Ray tracingu obdržet důležité informace pro výpočet. Mezi tyto informací patří například jak daleko je světelný zdroj, jak je silný, je potřeba znát úhel odrazu relativně k úhlu ke světelnému zdroji, než proběhne samotný výpočet nebo jak silný bude odražený paprsek. Tento postup se následně opakuje pro každý další světelný zdroj a také pro nepřímé světlo, které je získáno z odrazů od ostatních objektů ve scéně. Průhledné a poloprůhledné materiály jako voda nebo sklo lámou paprsky a přidávají další informace do renderingu. Vše musí mít nějaký limit v počtu odrazů, aby se zamezilo sledování do nekonečna.

Aby se ušetřilo co nejvíce výpočetního času, je potřeba přidat nějakou strukturu, která ukáže, zda má cenu pokračovat ve sledování daného paprsku. Pro Ray tracing je možné využít datové struktury - například BVH Traversal[10]. Dělení prostoru je komplexní proces, který má jednoduchou myšlenku, a to optimalizovat výpočet kolizí paprsku a modelu. Ve scéně, která má stovky objektů a každý objekt má miliony trojúhelníků, by bylo velmi obtížné zjistit, na jaké objekty paprsek dopadá a jak ovlivňuje daný model. Tento vyhledávací

[10]Bounding Volume Hierarchy Traversal



Obrázek 3.8: Ukázka práce BVH [35]

problém by hrubou silou trval velmi dlouho, BVH urychlí tento proces tím, že vytvoří strom objektů, kde každý objekt je obalen boxem. Pro detailnější vysvětlení BVH a ukázku implementace viz [34].

Samotný ray tracing nepokrývá veškeré fyzikální zákonitosti, které normálně vnímá zdravé lidské oko. S tímto algoritmem lze dosáhnout dobrých stínů, odrazů a lomů světla a přímého osvětlení. Ale důležité efekty, které udávají věrohodnost dané scény jako například difúzní materiál, nepřímé osvětlení a kaustiky, musí být dopočítány jinými metodami.

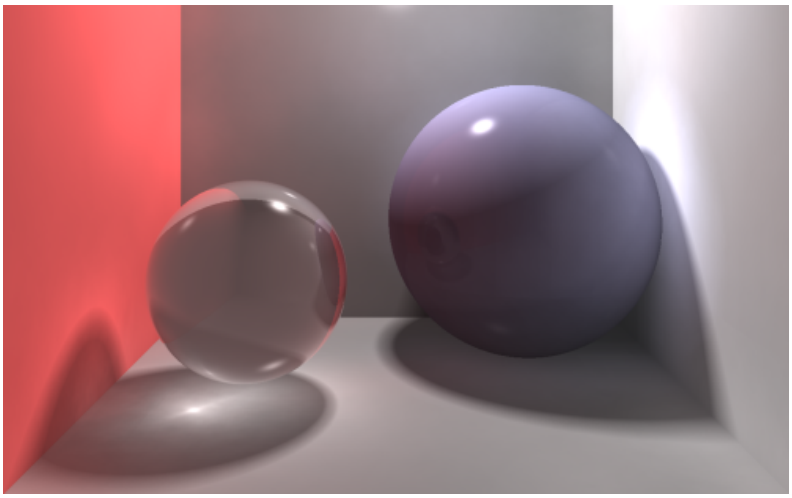
Jak již bylo zmíněno výše, ray tracing je v momentální době podporován grafickou kartou společnosti Nvidia. Některé počítačové hry tuto možnost již podporují, ale k plnému nasazení této technologie je ještě dlouhá cesta. Nicméně je to velký pokrok v technologii a metodách vykreslování virtuálních scén. Na stránkách [36] lze nalézt několik videoklipů, které znázorňují možnosti nových karet a jak takové efekty vypadají.

3.5.2 Fotonové mapy

Fotonové mapy [38, 39] jsou dvou fázový algoritmus, který vyvinul Henrik Wann Jensen [31], tento algoritmus přibližně řeší zobrazovací rovnici (rendering equation [40]). Tato metoda je efektivnější alternativa k metodě Monte Carlo [41, 42] ray tracing.

Fotonové mapování rozděluje řešení osvětlení od geometrie, kde toto řešení je uloženo ve speciální struktuře, která se označuje fotonové mapy. Tato technika je flexibilní, jelikož zobrazovací rovnice se může řešit různými způsoby a následně je výsledek upraven pomocí vytvořených map. Tato metoda může být i rozšířena pro výpočet kaustik a podpovrchového rozptýlení světla. Tento algoritmus je označován za dvou průchodový, jelikož nejdříve počítá fotonové mapy a následně je začleňuje do výsledku renderu.

Obrázek 3.9: Ukázka ray tracingu v praxi [37]



Obrázek 3.10: Ukázka výsledku metody fotonových map se zaměřením na kaustiky [43]

První průchod Tato část má dva základní kroky, prvním krokem je emise fotonů. Jednotlivé fotony jsou generovány ve světelných zdrojích a následně „vystřelovány“ do scény. Množství fotonů, které se získá ze zdrojů světla, je různé a závisí zejména na intenzitě daného zdroje. To znamená, že pokud je zdroj velmi jasný, pak takový zdroj vyprodukuje více fotonů než zdroj, který jen matně září. Směr daného fotonu je náhodně vybrán, přičemž vždy záleží na typu daného světelného zdroje. Emise fotonů lze měnit v závislosti na scéně.

Druhý krok je dělení fotonů, kde v závislosti na tom, co se s fotonem stane, se tato skutečnost zaznamená do fotonové mapy. Fotony jsou pohlcovány, lámou se nebo se odrážejí v závislosti na povrchu. Při tomto procesu mohou vznikat některé nepodstatné fotony, těch se lze zbavit pomocí techniky ruské rulety. Ilustračním příkladem může být vystřelení 1000 fotonů do scény, kde je objekt s pravděpodobností odrazu 0,5. Pak bude buď 1000 fotonů s poloviční energií, nebo 500 fotonů s počáteční energií, přičemž druhá možnost dá velmi podobný výsledek jako ta první. Z toho je také patrné, že dojde k uvolnění paměti a klesne paměťová i časová složitost. Je zjevné, že se jedná o modelovou situaci a že ve skutečnosti je potřeba započítat i jiné faktory povrchu objektu a prostředí.

Pro každý foton je potřeba uchovat informace o tom, kde se srazil s nějakým objektem, o jeho intenzitě (RGB složky), směrech a další důležité informace.

Druhý průchod Druhý průchod je samotné renderování a úpravy v závislosti na informacích ve fotonové mapě. Dojde k aproximaci zobrazovací funkce. Následně se využije klasický ray tracing, ”vystřelování”paprsků z kamery do scény, když paprsek zasáhne povrch, tak se posbírají informace o osvětlení z N nejbližších fotonů, které leží ve sférickém okolí bodu dopadu, z prvního průchodu. Tato informace je přidána k informaci o osvětlení získané z ray tracingu.

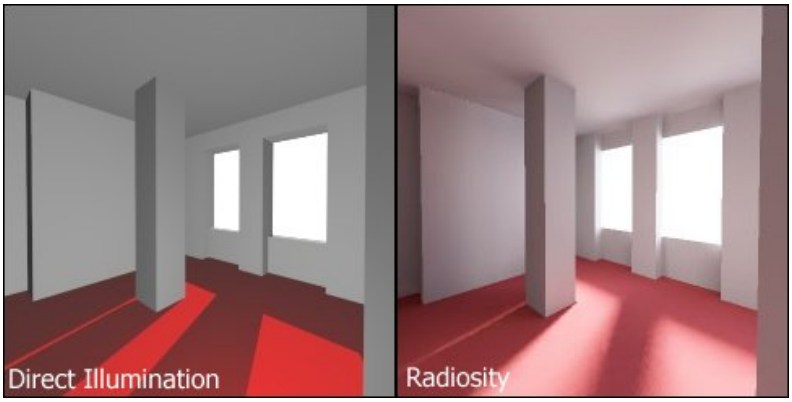
Tato metoda zvládá jak difúzní, tak spekulární materiály. Zvládá vyřešit kaustiky a mnohé další. Její velká nevýhoda je opět výpočetní čas, zejména nutnost dvojího průchodu.

3.5.3 Radiosita

Radiosita [31, 45, 46] je alternativní technika globálního osvětlení, opět je používána k aproximaci komplexního šíření světla na povrchu objektů. S touto technikou je možné dosáhnout fotorealistických obrazů. Radiosita využívá inženýrské metody nazývané „FEM“[11], kde je problém osvětlení celku rozdělen na menší problémy osvětlení jednotlivých částí. Na tyto malé plošky lze pak využít algebraické metody.

[11] Finite element method [47]

Obrázek 3.11: Kaustiky spočítané metodou fotonových map [44]



Obrázek 3.12: Srovnání osvětlení při použití Radiosity [48]

Obrázek 3.13: Ukázka radiosity v interiéru [49]

Distribuce světla se získá řešením lineárních rovnic pro světelnou výměnu mezi všemi ploškami. Tato metoda byla původně schopna zpracovat pouze difúzní povrchy, kde byl každý odraz konstantní a nezávislý na směru. Tento jednoduchý model byl později rozšířen a umožnil výpočet komplexnějších povrchů, ale stále je problém se spekulárními materiály, které nejsou řešeny správně. Radiosita jako taková je zcela pohledově nezávislý algoritmus globálního osvětlení, ale je výpočetně i paměťově náročný. Tato skutečnost byla později rozšířena o pohledově závislý výpočet a další techniky, které pomohly snížit časovou složitost algoritmu radiosity. Cenou za nižší čas bylo snížení přesnosti osvětlení pro vzdálené nebo nevýznamné plošky ve finálním renderu.

Tento model je výhodný pro osvětlení jednoduchých modelů s difúzními materiály, ale velmi náročný pro složité modely a modely s jiným než difúzním materiálem. Důvod pro vysokou výpočetní cenu je ten, že tento algoritmus počítá hodnoty pro každou plošku modelu. Navíc se jedná o řešení na uzavřeném modelu, mozaikovou reprezentaci skutečné geometrie. Samotná reprezentace může být velmi nepřesná, pokud se model nesestaví správně. Výsledkem toho jsou pak neostré stíny, které jsou velký problém u radiosity. Zde budou mít ostré stíny tendenci k rozmazání. Tento problém má také řešení, ale po jeho aplikaci opět vzroste časová složitost. Z předchozího je vidět, že pro použití v real-time simulátorech není tak úplně praktické.

3.6 Používané metody v počítačové grafice

V dnešní době záleží na povaze projektu, na kterém se pracuje [50]. Pokud je potřeba docílit realistické hry nebo simulace, pak se budou volit jiné modely než při tvorbě animací nebo podobných problematik. Různé druhy game engine[12] podporují různé modely osvětlení, a proto závisí také na prostředí, ve kterém probíhá vývoj.

Nejčastěji skloňovaný termín je BRDF[13]. Nad touto funkcí stojí další její vylepšení, která berou v úvahu další fyzikální složky materiálů. Speciální případ BRDF je BTF, která je postavená na stejném principu, jen využívá textur.

K BRDF a jejím variacím se využívají různé osvětlovací modely a jiné techniky aproximace pro správné zobrazení. Nemusí se zvolit jen jeden model, lze kombinovat několik modelů pro výpočty různých částí osvětlení a využít příslušné výsledky. Dle zvolených modelů se dostávají příslušné výsledky. Jako osvětlovací modely je možné volit jak globální, tak lokální osvětlovací modely a tím dostat příslušné výsledky. Každá volba má svou cenu, některé modely potřebují příslušné mapy navíc, některé zase jen výpočetní čas navíc. Ve dnešní době je snaha prosazovat real-time metodu globalního osvětlení, a to je Ray tracing. Cena je ve většině případů větší výpočetní čas, případně větší paměťové nároky.

3.6.1 BRDF

Většina světla, které lidské oko vnímá, je odražené a tím jsou vnímány i barevné nádechy od různých materiálů. Barvu objektu specifikuje několik faktorů, jako například spektrální charakteristika světla, které dopadá na daný materiál, vlastnosti povrchu a zejména které vlnové délky daný materiál odráží.

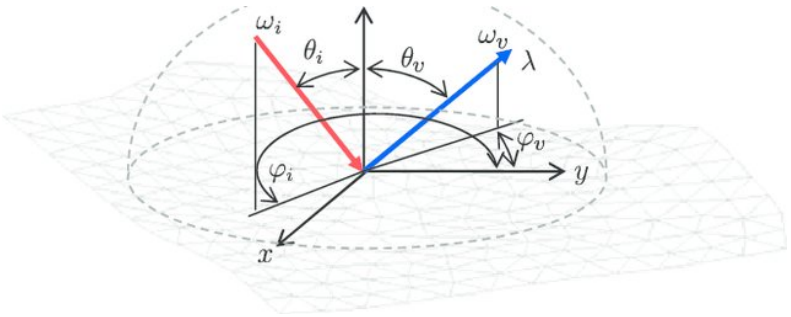
BRDF [25, 52] popisuje matematicky odrazové vlastnosti materiálu v nějakém bodě. Světlo dorazí do tohoto bodu z určitého směru ω_i a odráží se ve směru ω_r. BRDF se značí $f_r(x, \omega_r, \omega_i)$ a definuje poměr odraženého světla ku vstupnímu světlu promítanému na kolmou plochu. Odražené osvětlení v bodě x se značí $dL_r(x, \omega_r)$ a vstupní osvětlení $dL_i(x, \omega_i)$, celá rovnice pro BRDF je

$$f_r(x, \omega_r, \omega_i) = \frac{dL_r(x, \omega_r)}{dL_i(x, \omega_i)(\omega_i n)d\omega_i} \tag{3.7}$$

Jinými slovy říká, kolik procent světla se odrazí do daného směru. Pokud je BRDF pro všechny dvojice úhlů, pak jsou odrazové vlastnosti tohoto materiálu zcela popsány. Pro získání co nejvíce věrohodného výsledku je potřeba získat co nejvíce vzorků pro daný materiál.

[12] Game engine v doslovném překladu herní motor je software, ve kterém se daná hra nebo simulace vyvíjí.

[13] Bidirectional Reflectance Distribution Function – obousměrná odrazová funkce, její fungování bude vysvětleno v příslušné kapitole.



Obrázek 3.14: Ukázka BRDF [51]

Čím více se provede měření, tím lepší výsledky budou produkovány. Nejmenší možné množství vzorků, které stačí pro věrohodné výsledky, je 81×81. V tomto počtu lze interpolovat výsledky a získat potřebný výsledek. BRDF jako takové má určité vlastnosti, ty jsou uvedeny níže.

Linearita BRDF BRDF je lineární vzhledem k osvětlení, což znamená, že výsledné odražené světlo může být výsledek několika nezávislých zdrojů světla z různých zdrojů.

Helmholzova reciprocita Helmholzova reciprocita říká, že BRDF je symetrická funkce. To znamená, že BRDF v daném bodě zůstane stejná i při otočení směru dopadu a odrazu.

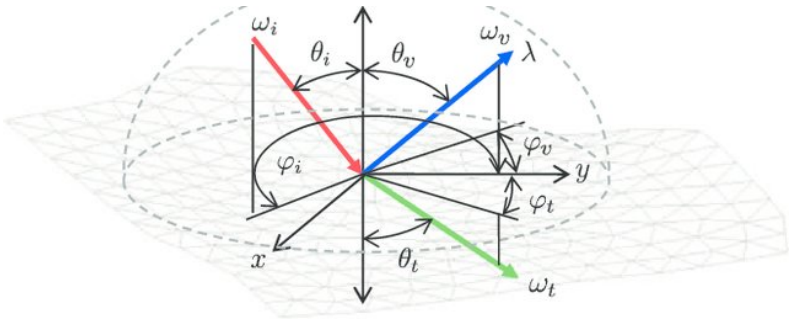
$$f_r(x, \omega_r, \omega_i) = f_r(x, \omega_i, \omega_r) \tag{3.8}$$

Toto platí pouze pro odraz, nikoli lom světla. Jedná se o jednu z hlavních vlastností dvousměrných globálních osvětlovacích technik.

Zákon zachování energie Pro diferenciální plošku platí, že poměr diferenciálního toku odraženého od této plošky vůči toku, který na ni dopadl, bude vždy menší než jedna. Jednoduše řečeno, odrazem se nemůže vytvořit energie.

$$\begin{aligned} \frac{d\Phi_r}{d\Phi_i} &= \frac{\int_{\Omega_r} L_r(\omega_r \cos\theta_r d\omega_r)}{\int_{\Omega_i} L_i(\omega_i)\cos\theta_i d\omega_i} = \\ &= \frac{\int_{\Omega_r}\int_{\Omega i} f_r(x,\omega_i,\omega_r) L_i(\omega_i)\cos\theta_i d\omega_i \cos\theta_r d\omega_r}{\int_{\Omega_i} L_i(\omega_i)\cos\theta_i d\omega_i} \leq 1 \end{aligned} \tag{3.9}$$

(An)izotropie Izotropní BRDF je invariantní k otočení kolem normály. Jinak řečeno závisí na natočení povrchu vůči směru světla. Obecná BRDF



Obrázek 3.15: Ukázka BSDF [51]

je čtyřrozměrná funkce, kde dva parametry slouží jako příchozí a dva jako odchozí směry. Pro izotropní BRDF je absolutní azimut příchozího a odchozího směru nezajímavý proti relativnímu rozdílu mezi azimutem příchozího a odchozího směru. Díky této skutečnosti je možné BRDF zavést jako trojrozměrnou funkci.

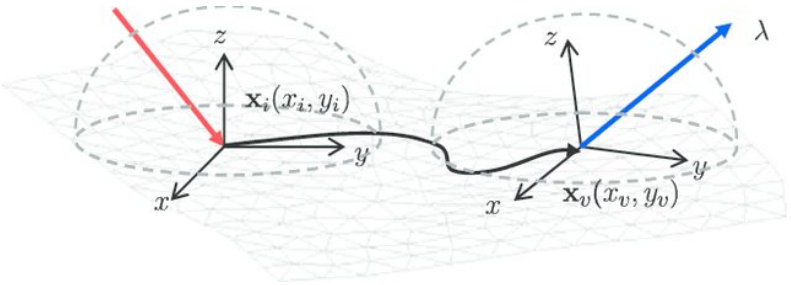
$$\begin{aligned} f_r(\theta_i, \phi_i, \theta_o, \phi_o) &= f_r(\theta_i, \phi_i + \phi, \theta_o, \phi_o + \phi) = \\ &= f_r(\theta_i, \theta_o, \phi_o - \phi_i) \ . \end{aligned} \tag{3.10}$$

3.6.2 BSDF

BSDF je variace klasického BRDF, které je rozšířeno o další podobnou funkci, která se označuje jako BTDF. Tato funkce řeší přenos světla v materiálu, na který dopadá světlo a případně kde opustí materiál. Je zjevné, že budou potřebná další data ať už naměřená, nebo vypočítaná. Přidání dalších prvků pro správné zjištění zvýší výpočetní a paměťovou zátěž. Na druhou stranu dojde k pozitivnímu posunu ve věrohodnosti produkovaného obrazu.

Myšlenka BSDF je následující. Světlo, které dopadne na nějaký materiál, se dělí na spekulární odraz, lom a pohlcení. Cílem BTDF je vyřešit část, která se zalomí, a dát její výsledek, případně nějak upravit následující výpočty. BRDF se postará o správné řešení vyobrazení v bodě dopadu.

Samotné BSDF stačí na průhledné a poloprůhledné materiály, ale pokud se renderují organické povrchy, nemusí to být dostatečné. U takových materiálů světlo dopadne na povrch, následně se může zalomit do materiálu a opět ho opustit. Místo, kde světlo opustí materiál, nemusí být to samé, jako bylo místo dopadu. Tento jev se označuje jako Subsurface Scattering [53].



Obrázek 3.16: Ukázka BSSRDF [51]

3.6.3 BSSRDF

Jak již bylo zmíněno, BSDF nezvládne veškeré efekty, které jsou potřeba pro fotorealistické renderování. Dalším krokem je tedy BSSRDF [54, 55], které přidává navíc práci s lomem světla v materiálu. Samotné BRDF pouze aproximuje fungování BSSRDF.

Jak již bylo řečeno, u BSDF se světlo může lomit do materiálu a pokračovat dál. Každý materiál má nějaký objem, skrz který lomené světlo cestuje, a je potřebné samotný BSDF model doplnit o další zákonitosti. BRDF i BSDF předpokládají, že se světlo odrazí v tom samém bodě, což není pravda. Světlo může v materiálu cestovat, dál se lámat a opouštět materiál v různých místech s různou intenzitou. BRDF i BSDF počítají pouze průměr paprsků, které by opustily materiál a dle toho dále počítá.

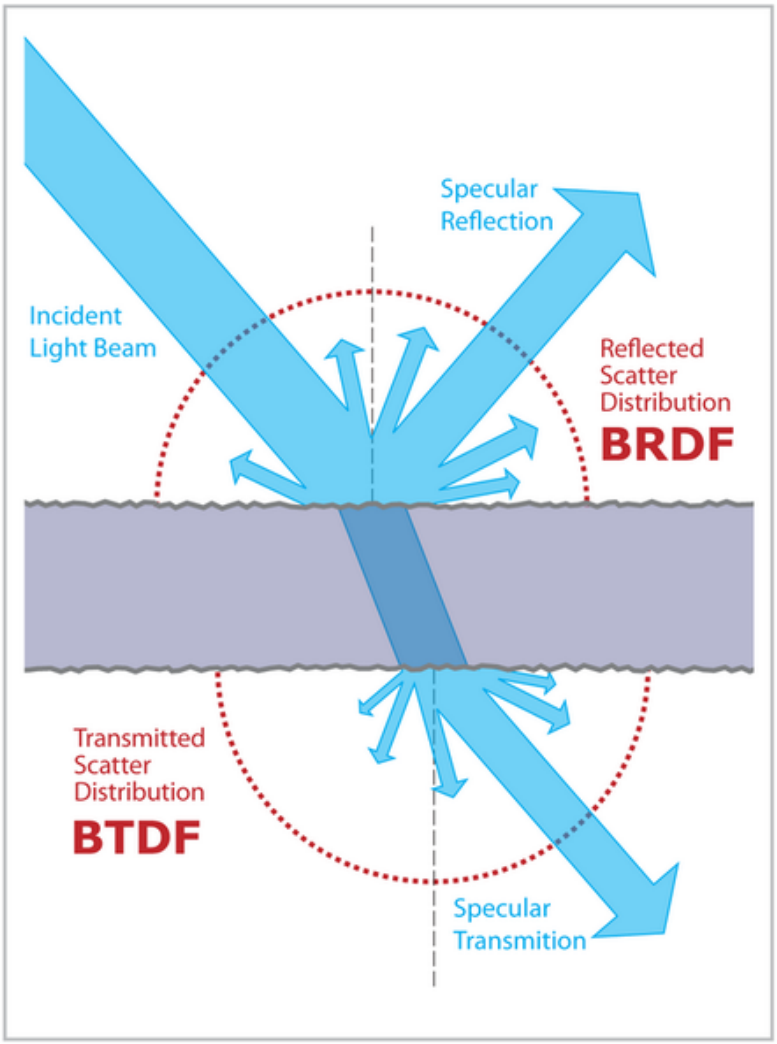
Protože se zahrnují veškeré paprsky, které mají opustit materiál, opět vzroste časová a paměťová složitost. To je opět kompenzováno zlepšením výsledků, které jsou trochu víc fotorealistické.

3.6.4 BTF

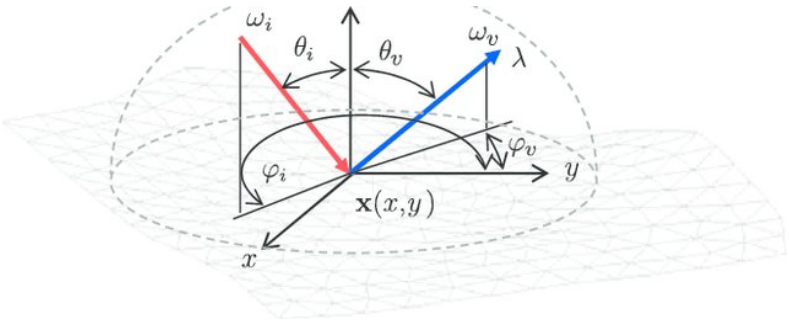
U modelů v CG nejsou důležité jen materiály, ze kterých je vytvořen daný objekt, ale také struktura povrchu, jaké má daný model normály, textury a mnoho dalšího. Je jasné, že s pouhým materiálem nelze získat požadovaný vzhled. Hlavní rozdíl mezi BRDF a BTF [57] je v tom, s čím pracují. BRDF, jak již zaznělo, pracuje pouze s materiálem a jeho chováním. BTF na druhou stranu pracuje s texturou, tedy přidává práci potřebnou pro správné použití textury.

BTF umožňuje naměřit mnohem více než povrch. Díky ní je možné reprodukovat i samo zastínění bez potřebné geometrie navíc. Ovšem to není zadarmo, jak se dá čekat, opět vzroste nárok na paměť. V tomto případě je to ovšem až do řádů GB[14].

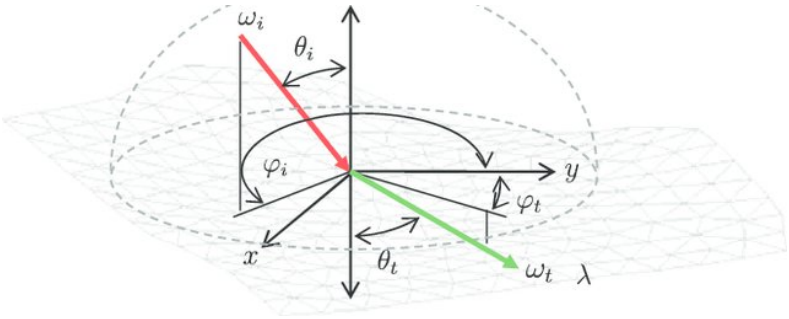
[14]Pro představu textura o velikosti 512×512 bude mít celkový datový objem $81 \times 81 \times 512 \times 512 \times 3 = 5{,}1$ GB. Pro texturu velikosti 4096 je tato velikost 330 GB



Obrázek 3.17: Porovnání BRDF a BTDF [56]



Obrázek 3.18: Ukázka BTF [51]



Obrázek 3.19: Ukázka BTDF [51]

Jen pro zajímavost jsou zde uvedeny počty parametrů některých funkcí typu BxDF. Pro základní BRDF je potřeba mít čtyři reálné proměnné, BSSRDF je osmi dimenzionální funkce a například BTF je šesti dimenzionální. [58]

3.7 Vhodné řešení pro projekt Věnných měst

Metody popsány pro osvětlení během dne jsou různé. Různé game engine využívají různé metody a mají různé možnosti. Nejlepší možností by bylo využít globální osvětlovací modely a dynamická světla, což bude jen velmi těžce proveditelné pro veškerá cílová zařízení. Pro zjednodušení simulace cyklu dne je možné využít sady textur, které se budou postupně měnit na daném modelu a tím se vytvoří příslušný efekt, tyto textury je možné vytvářet automaticky už v modelovacím programu. V závislosti na metodách podporovaných modelovacím programem budou výsledné textury kvalitní. Pokud se využije nástroje Blender, pak se kvalita výsledku bude pohybovat spíše k těm kvalitnějším. Tento nástroj totiž využívá hlavně ray tracing a tím pádem dostává adekvátní výsledky vzhledem k této metodě. Jedním z vhodných řešení by

tedy bylo vytvořit sady textur, které simulují denní dobu. Tím by bylo možné ušetřit velké množství výpočetního času. Také je to vhodný postup pro rychlejší práci s rozšířenou realitou.

Kapitola 4

Zarůstání budov

Další z přirozených jevů, který se objevuje stejně jako blednutí barev v průběhu času, je zarůstání [59] budov flórou, například břečťanem. Tento jev je schopen dodat scéně, která je pozorována, další rozměr realističnosti. Umožní tím budově trochu „zestárnout“. Nejlepším příkladem jsou rozpadlé hrady a zámky, o které nikdo nepečuje. To samozřejmě není jediný vhodný příklad, další může být strom porostlý lišejníkem nebo jinými popínavými rostlinami.

Tento jev se dá zkoumat z různých pohledů, např. fyzikálně, chemicky nebo biochemicky. Každý pohled přináší nové poznatky a nová kritéria pro přesný popis v CG. Například z pohledu fyziky je potřeba podívat se na gravitaci, přilnavost povrchů dané rostliny, rychlosti růstu atd. Z čeho se chemicky přesně skládá podklad u rostliny, jaké má rostlina možnosti na přilnutí k povrchu, jak na daný povrch reaguje atd. Zejména pak celkové složení dané rostliny.

Asi nejrozsáhlejší a nejvíce ovlivňující pohled na tuto problematiku má biologie. Z ní vychází mnoho podstatných informací, které pomáhají v CG správně umístit a nechat vyrůstat příslušné rostliny. Z tohoto pohledu je nejpodstatnější typický biom dané rostliny, přeci jen není možné nasadit rostlinu typickou pro tropický prales na polární čepičku planety. Dále je podstatné umístění, některé popínavé rostliny jsou fotofobní, rostou na neosvětlené straně stromů nebo budov. Další podstatný prvek z biologie rostlin jsou potřebné podmínky pro život. Jak vůbec daná rostlina roste, jestli se popíná co nejvýše s minimálním rozptylem, nebo jestli se rozrůstá do všech stran stejně. Dalším faktorem by mohl být typický tvar dané rostliny, může mít tvar připomínající písmeno S nebo I atd.

Mnoho z výše zmíněných věcí jsou podstatné detaily, které by neměly být opomenuty. Jejich hlavní účel je přidání věrohodnosti scénám, ve kterých se rostliny vyskytují. V mnoha případech se ve hrách či simulátorech dá nalézt zásadní chyba v ohledu kde, co a jak roste nebo neroste.

Obrázek 4.1: Ukázka typu zarůstání, které by mohlo být požadovaným výsledkem [60]

Obrázek 4.2: Jiný typ požadovaného výsledku [61]

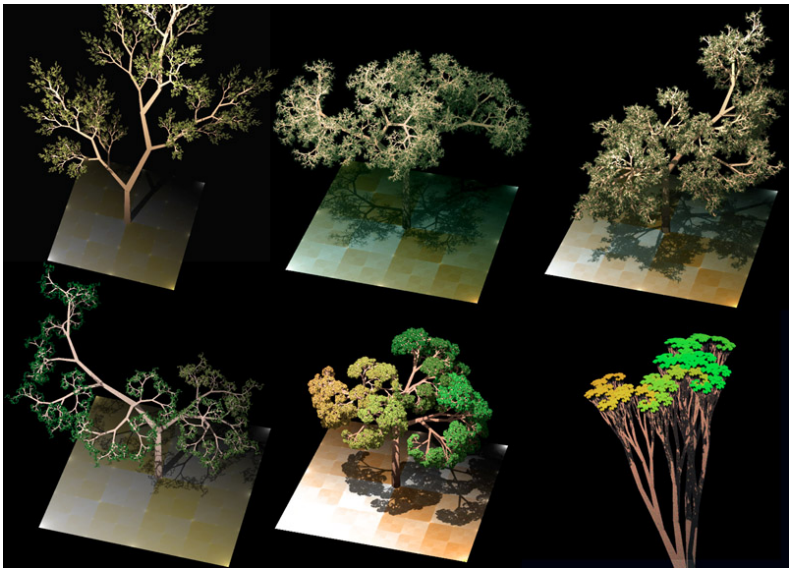
4.1 Přírodovědecky správné řešení

Rostliny jsou „modelovány“ a „deformovány“ v mnoha ohledech, jak již bylo zmíněno. Z toho také vyplývá, jaké všechny parametry se musí zohledňovat při vytváření fyzikálně správného [59, 62] modelu růstu popínavé flóry. Také je důležité, že v přírodě není vše zcela symetrické a nezachovává stejné rozestupy, občas jsou rostliny vyrostlé trochu podivně a i tato skutečnost se musí brát v potaz.

Mezi základní parametry u každé rostliny patří její druh a typické vlastnosti daného rodu rostlin. Tyto parametry by určily spoustu další parametrů, například jaký typ stonku daná rostlina má, jaký biom je pro ni nejlepší, kde přesně roste, jak se uchytává na svém povrchu atd. Dalšími parametry by mohla být tloušťka nově vyrostlé části rostliny a s jakou rychlostí jí vzrůstá obvod. Jak reaguje na povrch, se kterým se setkává, jinak řečeno jestli je jako víno, které se postupně chytá pomocí malých spirál, nebo má přísavky jako břečťan.

Dalším parametrem by mohl být úhel mezi dvěma větvemi dané rostliny za předpokladu, že má větve. Druh listu je také podstatný, protože podle něj se musí upravit rychlost růstu další části rostliny. V úvahu je třeba také brát vnější vlivy, jako je gravitace, vystavení světlu a jeho vliv na růst, vítr. Materiál podkladu je také velmi důležitý, ne všechny materiály umožňují rostlinám stejné podmínky k růstu.

Další podstatná část je roční období, ve kterém se daná rostlina sleduje. Některé přes zimu vůbec nejsou vidět, některé jsou pokryty sněhem. Na podzim většina rostlin změní barvu a postupně se stáhne do kořenů a čeká na



Obrázek 4.3: Ukázka možných výstupů při použití L-systemu [63]

jaro, kdy opět vyroste a snaží se přežít další část roku.

Jak je vidět, je mnoho potřebných parametrů a ne u všech musí nutně platit, že se dají začlenit do výpočetního procesu. Existuje však velké množství výpočetních modelů této problematiky, které jsou schopny podchytit větší množství zmíněných parametrů. Jak je také vidět při zahrnutí naprosto všech parametrů pro fyzikálně správný růst, je složitost výpočtů vyšší, než když zahrneme pouze některé podstatné parametry. A pokud se navíc přidá ještě generování detailní geometrie, pak složitost vzroste ještě víc. Většina postupů, které se touto problematikou zabývají, využívají jen některé části a podle toho také vydávají příslušné výsledky. V některých rostliny správně využívají svého okolí a rostou přes něj, v jiných se nezvládají dostat do vyšších částí modelů (nezvládnou přilnout k povrchu), nebo mají zvláštní tvar, který z hlediska biologie není možný nebo jen těžko dosažitelný.

4.2 Používané řešení v CG

V této sekci budou ukázány některá zpracování této problematiky, jejich pozitiva i negativa dle článků o L-systemu a virtuálních plazivých rostlinách [59, 62]. Dále zde bude zmíněn software, který pomáhá ve filmovém i herním průmyslu. A také vhodnost pro projekt Věnných měst.

Obrázek 4.4: Ukázka práce L-systemu. Jedná se o fraktální travnatý porost [65]

4.2.1 L-system

L-system [59, 64], známý také jako Lindenmayer system, je varianta formální gramatiky, která popisuje růst rostlin. Jako každá gramatika se i L-system skládá z neterminálních a terminálních množin, pravidel a počátečního symbolu. Jedná se o formální gramatiku. Tento systém byl vytvořen a postupně vyvíjen maďarským teoretickým botanikem a biologem Aristidem Lindenmayerem v roce 1968 na univerzitě v Utrechtu. První použití tohoto systému bylo pro simulace buněčného růstu. Později se také využil k popisu růstu rostlin a i jiných organismů. Kromě využití v biologii se také dá využít při vytváření fraktálů.

V průběhu let vývoje hardwaru se vědcům zalíbil L-systém a využili ho k vizualizaci růstu rostlin. Tato skutečnost přilákala mnoho dalších biologů, kteří se zajímali o vizualizaci růstu pro své simulace. Postupem času se ukázalo, že čistě biologický pohled na tento problém nestačí, a byli přizváni další vědci na vylepšení daného modelu. Zde vznikl model FSPM jako odnož L-systemu.

L-system je v jistém smyslu univerzální, dá se jím generovat všechno možné. Ovšem při tvorbě různých scén nebo i rostlin se nedají použít zcela stejné parametry, protože by to narušovalo věrohodnost dané scény. Proto byly vyvinuty další specifikace L-systému. Jedním z nich je L+C modeling language, jedná se parametrizovaný L-system.

Mnoho komerčně používaného softwaru využívá právě L-system nebo jeho derivát. Mezi hlavní zástupce patří X-frog nebo SpeedTree. Tyto nástroje umožňují realistické generování vegetace pro 3D prostředí. SpeedTree byl použit v mnoha známých projektech, jako je např. Avatar nebo World of Tanks.

Rekurzivní povaha L-systemu má některé nepříjemné dopady na gene-

rování vegetace. Například vůbec neřeší kolize samotných částí, které generuje. To platí, pokud mluvíme o původním systému, který vytvořil Arisid Lindenmayer. Pokud přihlédneme k dalším vylepšením a úpravám, které byly vytvořeny v průběhu let, tyto nedokonalosti se postupně podařilo zmírnit nebo odstranit. Některé tyto systémy byly vytvořeny tak, aby následovaly biomechaniku a další jevy, jako je třeba gravitace [59].

Proměnné : X F
Konstanty : + - [ ]
Počáteční symbol : X
Pravidla : $X \rightarrow F + [[X] - X] - F[-FX] + X$
$F \rightarrow FF$
Úhel : 25°

Algoritmus 1: Ukázka gramatiky pro získání fraktální rostliny 4.5, +/- znamená změnu úhlu do prava/leva o zadaný úhel. F znamená pohyb vpřed, X je pouze vyčkávací bod, závorky [ ] znamenají uložení a následné navrácení hodnot směru a úhlu.

Catherina Alena Jirasek [67] provedla simulace s ohledem na gravitaci a dosáhla velmi dobré výsledku, pokud daná rostlina volně rostla a nemusel se brát ohled na okolní prostředí. Kromě této simulace se jí podařila i simulace, ze které vyšel biomechanicky správný výsledek růstu do tvaru S. Tento výsledek je v souladu s výsledkem práce Fouriera et. al [68].

4.2.2 Metoda Thomase Lufta

Metoda sestavená Thomasem Luftem [69] začíná v jednom bodě, který je nazván kořenem. Následně bere v úvahu různé přírodní a okolní jevy, které mají vliv na růst. Hlavním bodem je směr růstu, váha předchozích částí, které mohou mít za následek změnu směru růstu, ale také náhodná ovlivnění, přilnavé síly k dalším objektům, up-vector, který imituje sledování světla rostlinou, a také gravitace. Jak se Luft zmiňuje, tento přístup nemá sloužit k biologickým simulacím, ale jako jednoduchý přístup k vytvoření složité a přesvědčivé vegetace, která se přizpůsobí existující scéně. Pokud by bral ohled i na biomechanické principy vegetace, pak by se model stal komplexnějším.

4.2.3 Particle systems – AMAP

Částicové systémy [62] se dají využít k různým věcem, jedna z nich je právě růst flóry. První popis použití tohoto systému je popsán Reffyem v článku [71] a dále rozveden Benešem a Jaegerem v článcích [72, 73]. AMAP je biologicky řízený systém "inteligentních"částic. Každá část simulované květiny má

Obrázek 4.5: Ukázka práce L-systemu. Výstup z gramatiky 1. Získáno z [66]

určité chování. Toto chování záleží na vnitřním stavu a vnějších podmínkách. Například pupen obsahující apikální meristém[15], může pokračovat ve svém růstu, pouze pokud je dostatečně zásoben ze země a CO_2 z přidružených listů. Ty produkují tento materiál, jen pokud jsou vystaveny světlu. Kořen může získávat vodu z půdy, jen pokud je přítomna atd. Původně byl AMAP jméno pro model v CG, ale v dnešní době je to komerčně dostupný produkt buď jako samotný program, nebo zásuvný modul do programu MAYA.

4.2.4 Speedtree

Speedtree [75, 76] je soubor programátorských a modelovacích nástrojů vytvořený Interactive Data Visualization, Inc. za účelem pomoci s vývojem her a vytváření filmů v přírodních prostředích. Tyto nástroje jsou schopny vytvářet animace pro listy, architektury a mnoho dalších věcí, které jsou vhodné pro zmíněná odvětví. To vše v reálném čase pro použití ve hrách nebo různých simulacích, které vyžadují rychlé renderování. Jako příklad využití tohoto sou-

[15] Apikální meristém jinak vzrostný vrchol je dělivé pletivo na vrcholku rostliny [74]

Obrázek 4.6: Ukázka výstupu generátoru Thomase Lufta [70]

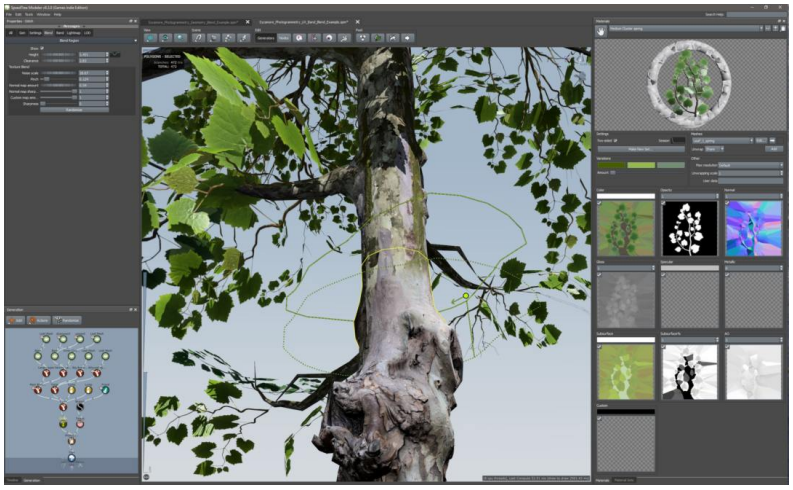
boru nástrojů lze uvést Star Trek Into Darkness, Avatar, Life of Pi a mnoho dalších.

4.3 Vhodné řešení pro projekt Věnných měst

V projektu Věnných měst by bylo vhodné docílit co nejkratšího zobrazovacího času a co nejmenší komplexnosti, proto dynamicky rostoucí vegetace není vhodná volba. Hlavním důvodem je potřebná geometrie navíc a tedy větší datový tok přes síť, což by nemuselo být dostačující. Kromě dynamické vegetace není vhodný ani statický model rostlin, které rostou po budovách, stále se jedná o další geometrii. Případně existují i další možnosti, ale ty jsou hodně specifické v tom, jaký typ rostliny je potřeba. Nebo jakým způsobem vegetaci vytváří. Jako další možnost by bylo vhodné uvést Ivy generátor od Thomase Lufta [69].

Pokud by bylo možné používat neomezené množství geometrie, tak by stačilo dle historických fotografií vytvořit přibližné pokrytí budovy příslušnou rostlinou. Následně dle parametrů L-systemu nebo jiného programu, přizpůsobit fázi růstu podle referencí. Celý takto vytvořený model následně stačí uložit a začít využívat.

V době tvorby této práce však nelze odesílat neomezená množství dat v rozumném čase. Dalším důvodem je, že mobilní zařízení by nemusela mít dostatečnou paměťovou kapacitu pro uložení všech potřebných modelů. Hlavním



Obrázek 4.7: Ukázka výstupu programu speedtree [77]

problémem by tedy byla složitost modelu. S každou další rostlinou, každou další fází růstu by nabývala velikost geometrie, a tím by narůstal i čas potřebný pro přenos modelů a vykreslení. Toto je velký problém pokud se začne uvažovat nad rozšířenou realitou a částečně i virtuální realitou.

Asi nejlepším krok by bylo opět se uchýlit k texturám, na původní texturu lze pustit L-system upravený pro textury a výsledky uložit. Tím se zajistí dostatečná rychlost i při slabém připojení nebo zobrazovacím prostředí.

Tento navrhovaný postup lze uplatnit, při využití dalších externích nástrojů pro vytváření výstupů z gramatik. Případně použít zásuvný modul pro tvorbu vegetace. Zde by ovšem byl problém správného nastavení kořene rostliny a dalších částí. Jiný postup, který by také mohl být předmětem zkoumání, je vygenerovat rostlinu dle všech kritérií, a následně tuto rostlinu promítnout na texturu budovy, kterou rostlina obrůstá. Tento přístup zachová fyzikální i biomechanické aspekty a bude schopen dodržet historický kontext. Největší úskalí nastává při pokusu o promítnutí jednoho objektu na jiný, respektive na jeho texturu. Nástroj Blender v tomto ohledu nevychází příliš vstříc a je potřeba hodně práce okolo. Oba předchozí přístupy mají své plusy i minusy, ale hlavním problémem zůstává potřeba složitého nastavování a hledání optimálního vzhledu dle reference. I tato část by se dala zautomatizovat, ale složitost řešení narůstá.

Při využití možností nástroje Blender, je možné docílit vizuálně zajímavého výsledku při vynechání některých částí původní textury. Místo ní lze přidat zelenou barvu, která simuluje rostliny. Tento postup je rychlý a jednoduchý, ale postrádá korektnost biomechaniky a ostatních oborů. Tento postup je nastíněn a ukázán v sekci 7.2.

Kapitola 5

Déšť

Déšť představuje přírodní jev, při kterém padá voda z oblohy v podobě kapek. Ta se dále kumuluje a vytváří odlišný vzhled krajiny. Tento jev není až tak zajímavý v reálném životě, ale v CG už jde o vcelku záludný problém jak správně nastavit jednotlivé části, výpočty atd. Možná vyvstává otázka, co je na tom tak těžkého. Déšť jsou jednotlivé kapky vody, ty dopadají na zem a následně nějakým způsobem ovlivní povrch, se kterým se setkají. Mohou se vsáknout, zůstat na povrchu, shromažďovat se v kalužích či nádobách. Při dopadu jednotlivých kapek na hladinu kaluží nebo vodní plochy vznikají různé jevy, které mění dynamiku hladiny. Jak je vidět, není to až tak jednoduché napodobit pro CG.

Při tvorbě simulace, ve které je obsažen déšť, je potřeba brát ohled na změnu vlastností materiálů, na které působí. Déšť je jedním z efektů, které nejvíce ovlivňují změnu materiálu. Jako další významné vlivy počasí lze uvést námrazu, pokrytí sněhem, změny po zásahu bleskem, kroupami atd. Jedním z největších oříšků pro CG je zobrazení duhy (lomu světla ve vodě).

Pro projekt Věnných měst, kde se zobrazují různé modely budov i v závislosti na aktuálním počasí, je možnost změnit vzhled i po této stránce vcelku rozumný krok. Následující sekce ukazují některá možná řešení.

5.1 Fyzikálně správné řešení

Pro fyzikálně správné řešení [79] pro jakýkoliv simulátor by bylo nutné vytvoření částicového systému, jeho správné nastavení a následné spuštění na dané modely. Na tyto modely by dopadaly jednotlivé kapky a ty následně ovlivňovaly celkový vzhled modelu. Při kontaktu jednotlivých kapek vznikají různé jevy („cákance“, vlnění hladiny, . . .). Pro správné simulování dynamiky hladiny by bylo zapotřebí generovat mesh na povrchu modelů, pokud je to možné[16]. Takové modely by musely mít velmi jemnou strukturu meshe, což

[16]Kolmé zdi nebudou zadržovat vodu tak, jako různé kapsy nebo vodorovné podlahy.

Obrázek 5.1: Snímek obrazovky s deštěm ze hry Batman Arkham Knight [78]

ale povede ke složitějším modelům, a tím i k pomalejšímu vykreslování. Tato část je zmíněna hlavně z důvodu zaplňování drobných pórů a děr v modelech. Momentální výpočetní síly grafických karet toto bohužel neumožňují.

Další fyzikálně správné řešení jsou soustavy rovnic, které popisují tuto problematiku. Jeden z možných způsobů je popsán v článku o fyzikálně založeném výpočtu proudění na povrchu těles [80]. Vlastnosti objektů, které jsou ovlivněny vodou, se rozdělí do jednotlivých aspektů a využívají se tam, kde mají smysl (voda se například nedostane do povrchu železa). Proudění neprobíhá pouze na povrchu těles, ale také v jejich struktuře. Jak je vidět například u látky, která je zcela nasáknuta vodou, zde voda proudí ve směru gravitace. Dále je důležitá členitost povrchu, vlastnosti podloží (sání / odpuzování vody) atd. Například pro podpovrchový tok vody platí následující rovnice:

$$\begin{aligned} &\frac{\partial}{\partial x}(K_{xx}k_{rw}\frac{\partial h_G}{\partial x}) + \frac{\partial}{\partial y}(K_{yy}k_{rw}\frac{\partial h_G}{\partial y}) + \\ &+\frac{\partial}{\partial z}(K_{zz}k_{rw}\frac{\partial h_G}{\partial z}) - W + q_{go} + q_{gc} = \\ &= \phi\frac{\partial S_w}{\partial t} + S_w S_s\frac{\partial h_G}{\partial t} \end{aligned} \tag{5.1}$$

kde x, y a z jsou kartézské souřadnice, K_{xx}, K_{yy} a K_{zz} značí hydraulickou vodivost s respektem k osám x, y a z. k_{rw} je relativní propustnost, která je funkcí nasycení vody, jak je uvedeno v relativní křivce propustnosti, h_G je hydrodynamický podpovrchový systém. W je objemový průtok skrz jednotku objemu. q_{go} tok na jednotku objemu podloží z dvourozměrné podzemní domény průtoku, q_{gc} tok na jednotku objemu podloží z jednorozměrného kanálu nebo povrchové vodní domény. ϕ je úroveň pórovitosti povrchu, S_w je stupeň nasycení vody,

který je určen retenční vlhkostí křivky v závislosti na hydrodynamice. S_s je možnost specifického sání porézního materiálu a t je čas.

Další podstatnou částí pro fyzikální korektnost by bylo sání materiálů, ne všechny materiály nasávají vodu stejně. Také by záviselo na velikostech jednotlivých hladin kaluží. Po skončení deště je jeho vliv vidět a postupně se ztrácí. Dle fyzikálních zákonů platí, že rychlost odpařování má vztah k velikosti hladiny, okolní teplotě a k dalším faktorům.

Zatím se mluvilo pouze o efektech, které jsou spojené s různými tělesy (modely), zemí atd. Ale podstatnou roli při dešti také hrají jiné faktory. Nejvýznamnější jsou vítr, množství vody, která dopadá na zem, a délka deště samotného. U těchto faktorů se různí jejich viditelnost, například vliv větru je vidět zejména na budovách a obecně suchých místech, např. pod stromy, u autobusové zastávky atp., kdy bude budova vypadat jinak za deště s větrem ze severozápadu a jinak s větrem z jihovýchodu. Množství vody, která dopadne na povrch země, je zejména viditelné na rychlosti vytváření kaluží nebo rychlosti namočení oblečení. Délka deště se pozná hlavně podle velikosti kaluží, kde podstatnou roli hraje množství napršené vody, nebo promáčení půdy. Z předchozího vznikají různé kombinace, které mohou dávat podobné výsledky, nebo naprosto jiné.

Zde je vidět část všech aspektů, které mají vliv na materiály. A je jasné, že v momentální době se této úrovně realističnosti dá dosáhnout jen velmi těžce. Během deště vzniká mnoho efektů, mezi které patří například duha, různé stříkance na okolí, „cákance“ v rámci kaluže nebo vlnění hladiny. Kromě těchto základních jevů se také mění vlastnosti materiálů, které se setkají s vodní hladinou. Může docházet k nasání vody, jejímu odpuzování atd., tím následně vznikají další efekty v okolí.

5.2 Používané modely výpočtu deště v CG

Vzhledem k výpočetní síle bylo potřeba upravit, jakým způsobem [81, 82, 83] se samotný déšť vytváří v CG. K vytvoření jednotlivých kapek se využívají částicové systémy, které produkují předem dané množství částic. Tyto částice simulují jednotlivé kapky a ty zanikají po určitém čase nebo po kolizi s modely. Modelům je přiřazena další textura, která obsahuje informace o odrazivosti daného materiálu. V závislosti na délce působení deště se pak daný efekt zvyšuje až na své maximum a toto maximum platí po dobu trvání deště. Po jeho skončení se následně vliv odrazivé složky začne snižovat až na nulu.

Tento způsob je jeden z mnoha, s jejichž pomocí se v momentální době vytváří efekt deště u textur, které mu jsou vystaveny. Samozřejmě jsou zde i další části, ale předchozí postup byla pouze kostra postupu. Jedná se o snadnější přístup k této problematice a je snáze pochopitelný i následně ovlivnitelný pro obyčejného uživatele.

Obrázek 5.2: Snímek obrazovky ze hry Watch_Dogs s pohledem na zpracování deště [84]

To, jak přesně se déšť vytvoří, závisí na volbě nástrojů, které použijeme. Jiný pracovní postup bude pro Unreal Engine, jiný pro Unity, Substance Designer, Blender atd. Ve videotutoriálu Unreal Engigne 4 – déšť a co k němu patří [85] je nastíněn jeden z možných postupů. Je zde také vysvětleno, co k této problematice patří. Výsledný postup práce se může lišit i podle kombinací nástrojů, které se použijí.

Výše zmíněné nástroje obsahují také Substance designer. Jedná se o placený nástroj pro tvorbu 3D, který umí kromě jiného exportovat různá nastavení a další potřebné věci. Co je podstatné, tato exportovaná nastavení se dají importovat rovnou do různých enginů a dále s nimi pracovat.

U používaných postupů si lze všimnout různých nedokonalostí. Zářným příkladem je hlína či kůra stromu bez porostu. V mnohých hrách nebo simulátorech je vidět, že se po dešti například pole leskne jako železný materiál na slunci nebo pouštní pláň stejně jako zmíněné pole. Toto je způsobeno tím, že kromě barvy není další informace, jaké vlastnosti daný materiál má. Další vcelku viditelná nedokonalost je, že odlesk se na jedné textuře zvyšuje rovnoměrně v celé textuře a ne postupně, podle úrovně „namočení“ daného materiálu. Tento jev, je ale viditelný až při důkladném sledování dění ve scéně, s vhodnou změnou v odrazivosti je to nepostřehnutelné. Toto je vidět ke konci videa u výše zmíněného tutoriálu [85].

5.3 Vhodné řešení pro projekt Věnných měst

Nejlepší volbou pro projekt Věnných měst by bylo vzít některou z realistických metod a použít je. Výpočty potřebné ke správnému určení všech možností jsou

obsáhlé a složité. Kromě toho by bylo potřeba znát velké množství informací navíc například struktura a vlastnosti materiálů budov. Pro silné stroje by to nemusel být tak velký problém, ale pro mobilní zařízení už ano. Tím pádem je jasné, že bude potřebné zjednodušení. Přidávání další geometrie (zde particles system[17]) by také uškodilo jak výpočetnímu času na zobrazení, tak množství dat, která by se musela odesílat/udržovat v paměti. Proto je rozumným krokem přidat další sadu textur, které budou tento jev simulovat.

Bylo by možné využít spekulární mapy na odrazy a texturu „mokrého“ modelu uložit zvlášť. Ve výsledku by se tyto informace spojily a vytvořily výsledný vzhled mokrého povrchu. Pokud nebude možné využít spekulární mapy, pak bude vhodné využít jen zbarvení. Jednou z posledních možností, které zbývají, je nástroj Blender a materiály Cycles. S jejich pomocí se vytvoří další potřebné textury. Tyto textury budou zbarvené, jako by na ně působil déšť, a mohou na sobě mít slabý odlesk tam, kde to bude dávat smysl. Vytvoří se několik těchto textur podle toho, jak moc „mokré“ dané modely mají být.

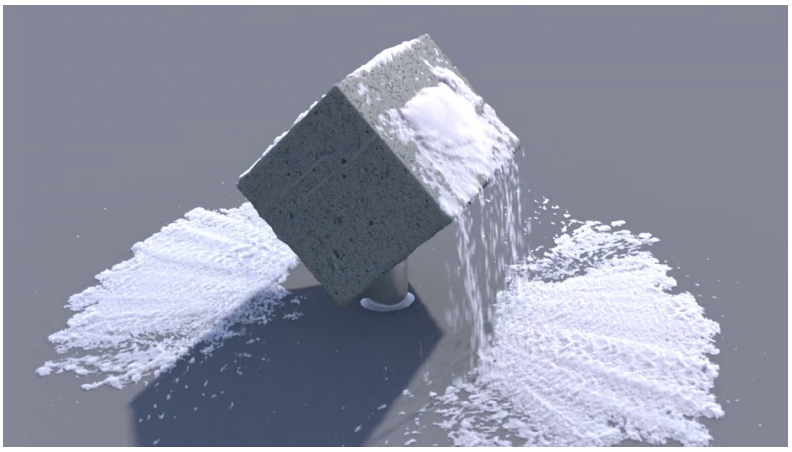
[17] Částicový systém, který by byl vhodně nastaven pro simulaci deště.

Kapitola 6

Roční období

Vzhled budov, aut atp. nemusí být ovlivněn pouze typickými přírodními jevy, jako je déšť nebo sníh. Vzhled okolních předmětů je do značné míry také ovlivněn tím, ve které roční době je daný objekt pozorován. Například se budova bude nějak jevit v teplejších odstínech v průběhu jara a celé léto. Na podzim bude tento teplý nádech postupně mizet velmi podobně, jako se na jaře objevil, a bude přecházet do studených odstínů, tento jev je způsoben pohybem planety vůči slunci. To má za následek jiný sklon určitého místa na zemi vůči paprskům slunce. Tyto paprsky musí urazit určitou vzdálenost navíc v atmosféře, ve které interagují s jejími prvky. Tím pádem ztratí část své energie a také jim trvá déle, než dopadnou na příslušné místo.

Jedním z faktorů síly paprsků ze slunce je jejich sklon k zemi. Jinak řečeno kam přesně na zemi dopadají paprsky (zeměpisná šířka a délka). Zde je také vidět rozdíl např. na rovníku a na zeměpisné poloze České republiky, počet



Obrázek 6.1: Ukázka duální povahy sněhu [86]

Obrázek 6.2: Ukázka laviny, která je vypočtená pomocí matematického modelu [86]

ročních období je rozdílný (4 vs. 2), jiná je teplota, vlhkost atd.

6.1 Fyzikálně správné řešení

Pro fyzikálně správné řešení problematiky osvětlení v průběhu roku by bylo zapotřebí naměřit příslušné úhly dopadů slunečních paprsků. Tyto hodnoty se dají využít pouze pro velmi úzké území. Dále by bylo vhodné naměřit společně s úhly i spektra paprsků. Spektrum by se využilo pro barvu zdroje světla ve venkovních prostorech a pro světlo, které proniká do budov pomocí oken. V úvahu by se muselo brát i světlo odražené od okolí a to celý problém může výrazně ztížit.

V případě zimy by se dalo použít podobné řešení s částicemi jako při dešti nebo fyzikálně založené výpočty. Druhý postup není až tak jednoduchý vzhledem k povaze sněhu. Sníh jako takový je pevná hmota (sněhové vločky), mezi kterou je vzduch. Pokud se začne uvažovat o lavinách a dalších jevech spojených se sněhem, pak se i sníh začne chovat jako kapalina. Tento problém již má řešení pomocí Material point method [87, 88]. Také by bylo potřeba stanovit směr větru pro přesné zasněžení modelů a mnoho dalších jevů, které se běžně vyskytují. Pro simulace sněhu vzniklo mnoho studií a jejich použití se liší.

6.2 Používané řešení v CG

Tato problematika se týká zejména simulátorů a počítačových her. Počítačové hry tento problém řeší [89, 90] buď po svém, nebo vůbec. První možnost se

Obrázek 6.3: Konzolový titul God of War, který je zasazen do severské mytologie má také ukázkově zpracované zimní prostředí [91]

může lišit podle autorů, může jít o změnu osvětlení, živější textury objektů a tak podobně. S druhou možností se lze setkat častěji, většina her a simulátorů se odehrává v malém časovém horizontu, než aby se tento problém řešil. V momentální době převládá spíše druhý způsob, časová osa není příliš velká, a tím pádem není nutné řešit jednotlivá roční období. Případně se řeší jen některé, tak jako je to u některých herních titulů z žánru tahových strategií (Endless Legend, Medival: Total War). U těchto her se mění roční období pouze na léto nebo zimu.

Pokud má nějaký projekt dlouhou časovou osu a je potřeba tento problém řešit, pak existuje nespočet cest jak docílit dostatečně věrohodného napodobení ročních období. Pro jednotlivá roční období platí různé specifikace. Na jaře příroda začíná ožívat, a tím pádem dochází k živějšímu zabarvení okolí, sluneční paprsky také získávají na síle, a proto jsou přítomny teplejší odstíny. Zde může být použita základní textura, s přidaným parametrem živosti objektu. Tento parametr v průběhu jara bude narůstat, až se dostane na své maximum. Kromě této skutečnosti by bylo vhodné zahrnout i přítomnost hmyzu a podobných živočichů v přírodě, kteří se projevují hlavně na jaře, to samé platí i o flóře. V létě jsou všude živé barvy, a tím pádem se využívá textura tak, jak byla vytvořena. Případně se na ní projevují některé jevy, jako je zarůstání atd. Na podzim postupně živost opět klesá a odstín paprsků se posouvá do studenějších odstínů.

Zima je kapitola sama pro sebe, sluneční paprsky jsou ve studených odstínech a stejně tak veškeré modely, které jsou v nějaké scéně nebo simulátoru.

Proto se v tomto ročním období musí brát ohled i na sníh, který je hlavním znakem. Opět vyvstává otázka, jak sníh reprezentovat: jako další mesh, jako texturu, jinak? Každý způsob může mít své pro i proti. V některých titulech se využívá původní mesh, která se pomocí shaderů upraví a využije jako sníh. Tímto způsobem lze i docílit efektu hlubokého sněhu (Assassin's Creed III [92]). Jinde stačí posunout postavu o několik centimetrů níže a využít normálových či displace map, do kterých se přidá informace o deformaci. Cest k docílení efektu sněhu je mnoho a dost se různí v možnostech použití.

6.3 Vhodné řešení pro projekt Věnných měst

V projektu Věnných měst by bylo vhodné nevyužívat klasického osvětlení, ale jen textur, proto je změna v barvě zdroje světla nevhodný koncept. Nicméně samotné textury jsou dostatečně silný nástroj. Jak již bylo zmíněno, hlavní rozdíl je v osvětlení, proto lze vzít originální textury a ty upravit podle příslušného ročního období. To by znamenalo nejméně čtyři textury, kde každá reprezentuje jedno roční období.

Co se týká sněhu, jeho reprezentace proběhne opět změnou textury na bílou tam, kde to má smysl. A pokud to bude jen trochu možné, i změnou normálových map. Sníh postupně dorovnává nerovnosti stejně tak jako voda. Tím pádem se díry a prohlubně na povrchu modelu zaplní a nebudou vidět. Generování další meshe je těžko použitelné vzhledem k technologii.

Obrázek 6.4: Budovatelská strategie zasazená do věčné zimy Frostpunk. Zde je vidět nádherné zpracování mechanik sněhu [93]

Kapitola 7

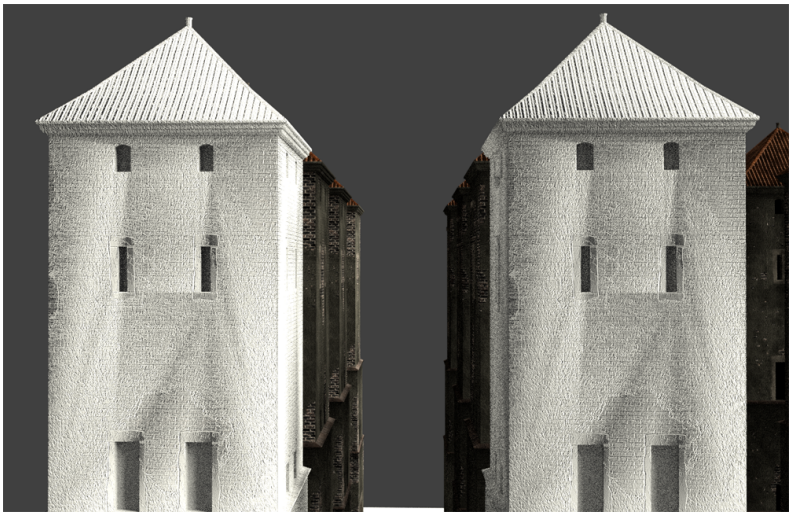
Návrh

Tato kapitola obsahuje návrh řešení zásuvného modulu do programu Blender. Budou zde navrženy jednotlivé části modulu, zejména to, jak se bude dosahovat různých výsledků pro dané jevy. Pro tuto práci bylo vybráno pět efektů a to blednutí barev(viz sekce 7.1), zarůstání budov(viz sekce 7.2), denní doba(viz sekce 7.3), roční období(viz sekce 7.4) a déšť(viz sekce 7.5). Každému efektu je věnována samostatná sekce. Velmi časté budou odkazy na jazyk Python, jelikož se zásuvné moduly do programu Blender píší zejména v tomto jazyce.

7.1 Blednutí barev

Blednutí barev, jak napovídá sekce 7.1, je vcelku obsáhlá a složitá problematika. Nicméně metoda výpočtu změn v barvě popsaná v článku [8] ji dokáže velmi dobře nasimulovat. Tato metoda ovšem počítá s velkým množstvím informací, které se v některých případech dají dohledat jen těžce. Bylo by potřeba pro každou budovu, která se bude zobrazovat, zjistit například stavební materiál, ze kterého se skládá, jakou dobu už daná budova stojí nebo zda nedošlo k opravám. Veškeré tyto informace mohou narušit hloubkovou strukturu původního obezdění a změnit tak vlastnosti povrchu, a tím pádem by se změnil i hloubkový výpočet zmíněné metody. Samotné výpočty z numerického přiblížení metody obsahují zejména maticový počet. Ten je nativně podporovaný v jazyce Python, který využívá Blender. Vzhledem k přítomnosti i dalších částí výpočtu, jako například časové údaje, by bylo vhodné přidat i další moduly jako například SciPy, NumPy nebo jiné podobné moduly. Pro zrychlení výpočtů by také bylo vhodné využít Cython či jiné verze Pythonu. Hlavní výhodou celé této metody je její fyzikální přesnost.

Pokud by byla tato metoda použita, pak by výsledky byly z fyzikálního i chemického hlediska téměř správné, vždy může nastat nějaká chyba či odchylka. Jedním z vážnějších problémů by mohla být potřeba velkého množství nastavení pro různé budovy. Z těchto důvodů, by bylo vhodné tuto metodu zjednodušit. Zde by bylo možné vytvořit pomyslnou databázi s různými ukáz-



Obrázek 7.1: Porovnání metod prolnutí vrstev *screen* a *lighten*

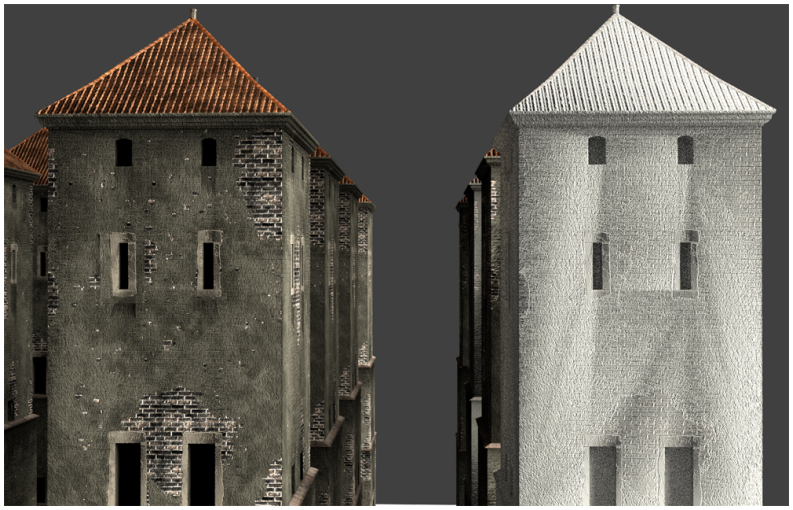
kovými hodnotami a z nich si vybrat pouze ty, které jsou vizuálně uspokojivé. To ovšem bude stále trvat vcelku dlouhou dobu. Proto je vhodné další zjednodušení. Další cestou je využít různé módy míchání barev z 2D grafiky. Tento postup nevyžaduje žádné další informace o daném modelu, je jednodušší na pochopení i vytvoření. Jediné, co je potřeba, je textura příslušící danému modelu a nastavení určitých hodnot. Hlavní výhodou tohoto přístupu je jeho rychlost. Odebráním informací o materiálu, ze kterého se budova skládá, dojde k nepřesnému výpočtu barvy po dlouhodobém vystavení světlu. Jinak řečeno, ne všechny materiály dopadnou tak, jak by tomu bylo ve skutečnosti. Nicméně pro stavby by tento přístup měl být vhodný a dostačující.

Textury je možné v Blenderu míchat různými způsoby, tak jak tomu je v editorech pro fotografie. V editoru materiálu lze použitou texturu pozměnit dle nastavení příslušného nodu, který je označen jako *mixRGB*. Pro zesvětlení barev se využívají zejména čtyři hlavní metody výpočtu. Každá z nich využívá dvě a více vrstev, které se smíchají pomocí definovaného výpočtu.

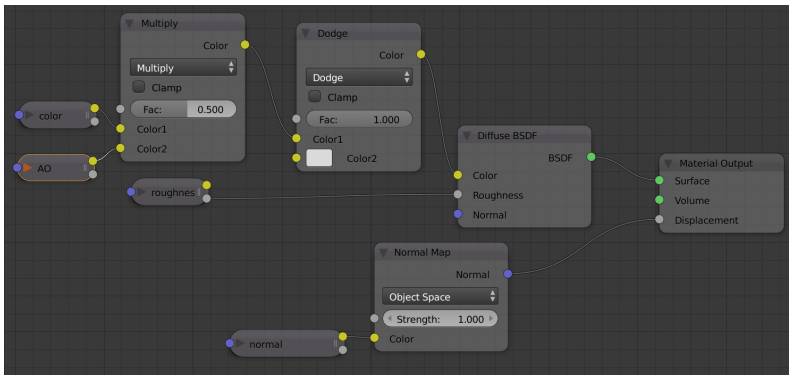
Screen vezme dvě vrstvy které jsou nad sebou, invertuje jejich barevné hodnoty, ty následně mezi sebou pronásobí a vydělí. Nakonec dané hodnoty invertuje zpět.

Lighten porovná odpovídající pixely přední vrstvy a podkladové vrstvy, následně použije větší hodnotu jako výstup.

Dodge každý pixel v pozadí se vynásobí hodnotou 256 a následně se výsledná hodnota vydělí inverzí pixelu z vrstvy popředí zvětšeného o jedna.



Obrázek 7.2: Porovnání metod prolnutí vrstev *dodge* a *add*



Obrázek 7.3: Výsledná kompozice nodů pro blednutí barev

Add jednoduše sečte pixely dvou vrstev a případně oříznou hodnoty, tak aby maximální hodnota nebyla větší než je maximum pro pixel.

Tyto efekty lze v programu Blender lehce realizovat, jelikož jsou podporovány přímo v nastavení nodů[18]. Jednotlivé metody mají i své matematické vyjádření, ta jsou převzata z odkazu Blenderu na oficiální stránky Gimpu [94]. Použité proměnné jsou f pro popředí a b jako pozadí. Pro metodu *screen* platí

[18] Anglické slovíčko node/nodes je v komunitě okolo Blenderu vcelku časté a využívá se i v češtině. Označuje jednotlivé části při kompozici materiálů. Ačkoliv by bylo vhodné využít český ekvivalent, pro přesnou představu uživatelů Blenderu je ponechán novotvar tohoto slova.

$$f(f,b) = 255 - \frac{(255-f)\cdot(255-b)}{255} \ . \tag{7.1}$$

Metoda *lighten* využívá následující předpis

$$f(f,b) = \max(f,b) \ . \tag{7.2}$$

Nejlépe vypadající metodou je *dodge*, který lze vypočítat následujícím způsobem

$$f(f,b) = \frac{256\cdot b}{((255-f)+1)} \ , \tag{7.3}$$

metoda *add* může být vyjádřena takto

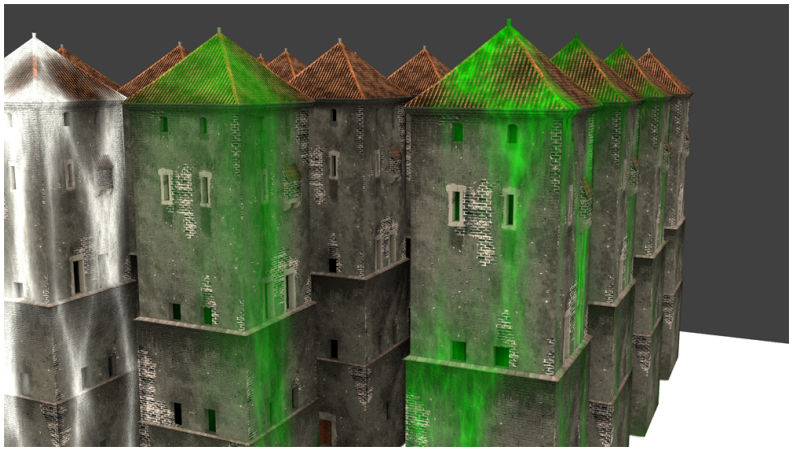
$$f(f,b) = \min((f+b),255) \ . \tag{7.4}$$

Tento efekt se nebude realizovat, ale je zde ukázán výsledek s různými hodnotami a v porovnání s ostatními zmíněnými metodami. Kromě výsledků je zde ukázáno i nastavení nodu pro dosaženi daného efektu a pseudokód daného nastavení. Z těchto jednoduchých alternativ se vizuálně nejlépe jeví *dodge* s nastavenou téměř bílou barvou podkladu a rozumným mícháním barev.

Pro několik algoritmů, které se v této práci objeví, je vhodné zavést značení jednotlivých částí. Veškeré **textury** budou označeny tučně, *proměnné* italikou, VEKTOR velkými písmeny a Materiál modelu bude vždy napsán celými slovy a každé slovo začíná velkým písmenem.

Vstup: Texture, **AOmap**, **NormalMap**, **Roughness**
Výstup: FadedMaterial
for *each pixel in* ***Texture*** *and each AOpixel in* ***AOmap*** **do**
  colorTO ← zkombinuj hodnoty *pixel* a *AOpixel*
  pixel ← vypočti novou barvu z hodnoty *colorTO* a hodnoty 0,7
end
DT ← **FT** zkombinuj s **Roughness** mapou textury
FadedMaterial ← z **DT** a **Normalmap** poskládej výslednou zesvětlenou barvu
return FadedMaterial;

Algoritmus 2: Pseudokód reprezentující funkcionalitu z obrázku 7.3, proměnná **AOmap** označuje Ambient Occlusion map a hrubost povrchu je zde reprezentování pomocí **Roughness**. Hodnota 0,7 byla empiricky zjištěna jako vhodná pro zesvětlení celé textury. **DT** označuje mezikrok pro poskládání **Roughness** a zesvětlené textury, zde označenou jako **FT**. Ukázka výsledné textury po této úpravě je v kapitole 9.



Obrázek 7.4: Vygenerované pokrytí rostlinou, jasně zelená barva je zvolena pro viditelnost vygenerované části

7.2 Zarůstání budov

V sekci 4.3 byly naznačeny různé možnosti řešení této problematiky. Jako poslední zmíněná je možnost použití vestavěných možností nástroje Blender. Ten nabízí několik procedurálně generovaných textur, které je možné využít různými způsoby. Tyto textury lze různě spojit a vytvořit rozmanité efekty, případně je maskovat.

Každý z těchto generátorů má své možnosti a uplatnění. Mezi procedurálně generované textury, které nabízí Blender, patří:

Brick Texture jak již název napovídá, tak se jedná o procedurální generování textur, které připomínají cihlové zdi.

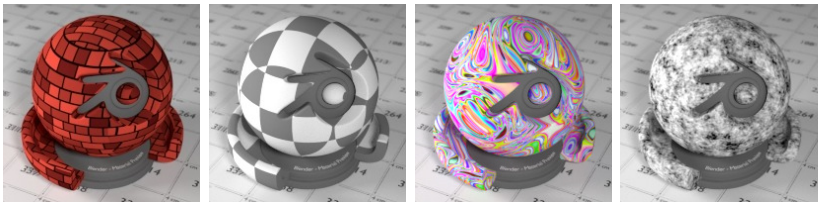
Checker Texture Zde je taktéž snadné představit si nějaký možný výsledek. Jak název napovídá jedná se o šachovnicové pole.

Magic Texture Vytváří „halucinogenní“ barevnou texturu, ukázka je na obrázku 7.5.

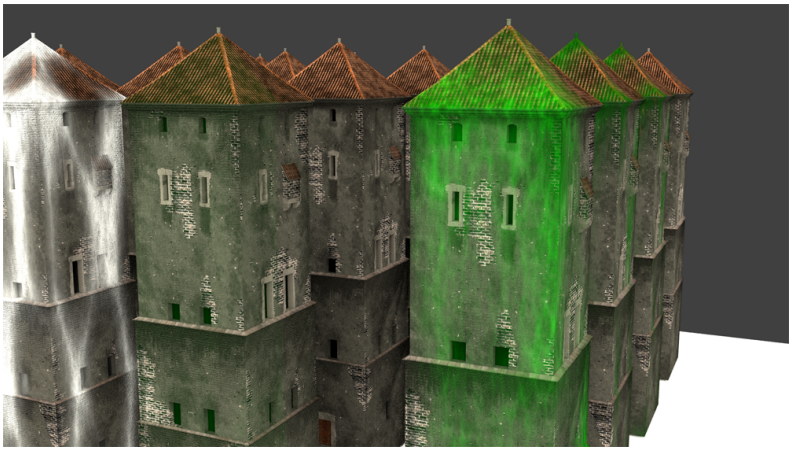
Musgrave Texture Jeden z typů fraktálních šumů, který je založen na Perlinovském šumu. Větší představu poskytne obrázek 7.7.

Noise Texture Vytváří šum, který lze využít pro různé účely jako například vytvoření hrubšího povrchu.

Voroni Texture Vychází z matematického postupu dekompozice prostoru. Jinak se také nazývá *Voronoi tessellation* nebo *Dirichlet tessellation*.



Obrázek 7.5: Příklady různých druhů procedurálních textur v základním nastavení. jedná se z leva o textury *Brick* [95], *Checker* [96], *Magic* [97] a *Musgrave* [98]

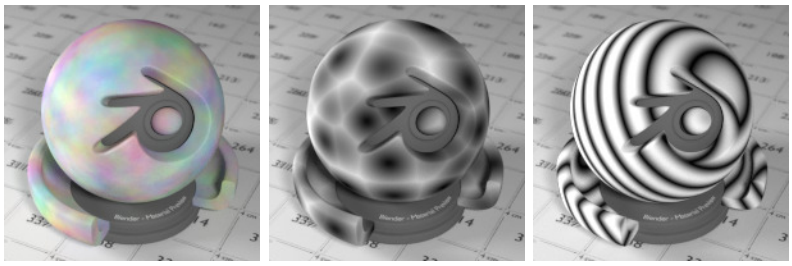


Obrázek 7.6: Ukázka vygenerovaného pokrytí rostlinou s jinými parametry

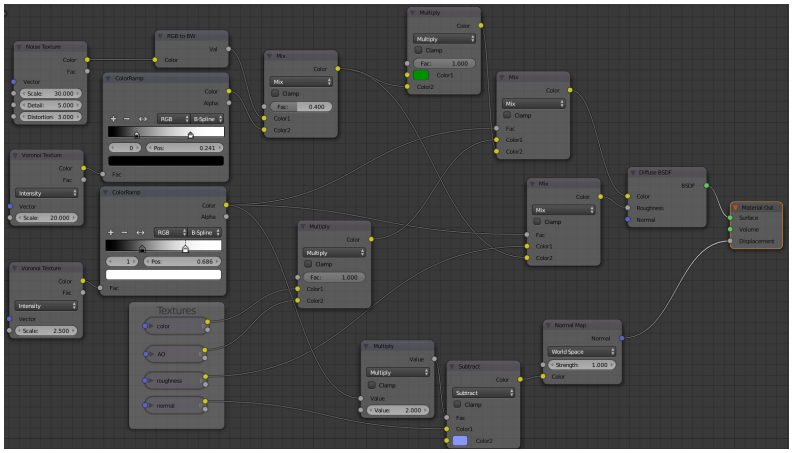
Wave Texture Tato možnost umožňuje vytvořit vlny v závislosti na svých parametrech. Ty mohou být uniformní nebo různě rozvrstvené.

Z výše zmíněných procedurálních textur je možné vybrat libovolnou skupinu. Je ale jasné, že první dvě se pravděpodobně budou těžce komponovat do výsledku. Vizuálně zajímavý výsledek podává kombinace nodů *Voronoi Texture* a *Nois Texture*. Výsledky jsou viditelné na obrázcích 7.4 a 7.6.

Obrázky 7.4 a 7.6 ukazuje možné nastavení k získání napodobení zarostlé budovy. Tento návrh nezvládne napodobit rostliny ani správné zarůstání, nicméně vytváří rychlý a zajímavý výsledek. Celé nastavení nodů lze upravit tak aby bylo trochu lépe ovladatelné. Hlavním nedostatkem tohoto postupu je nemožnost nastavení různých rostlin.



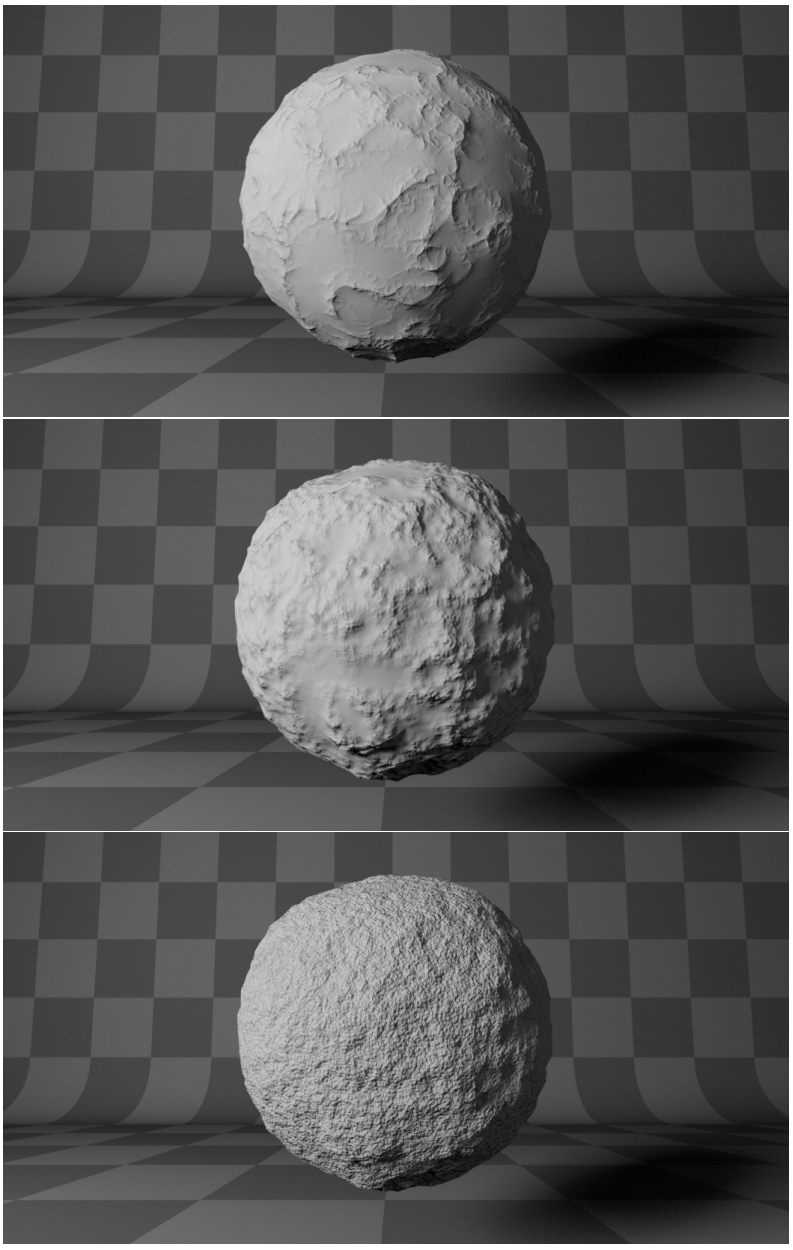
Obrázek 7.7: Příklady výše zmíněných druhů procedurálních textur v základním nastavení. jedná se z leva o textury *Noise* [99], *Voronoi* [100] a *Wave* [101]



Obrázek 7.8: Výsledná kompozice nodů

Vstup: Texture, **NormalMap**, **AOmap**
Výstup: OGtex
plCov ← pomocí procedurálních textur vytvoř pokrytí budovy
plM ← pomocí procedurálních textur vytvoř masku porostu
mat ← ze vstupních textur poskládej základní vzhled modelu
nC ← namixuj novou barvu modelu s pomocí **plCov** a **mat** s faktorem **plM**
Nnormal ← uprav normály modelu pomocí **NoramlMap** a **plM**
OGtex ← složením vytvoř výslednou texturu z **nC** a **Nnormal**
return **OGtex**

Algoritmus 3: Pseudokód reprezentující funkcionalitu z obrázku 7.8



Obrázek 7.9: Porovnání různých nastavení procedurální textury *musgrave*. Metody zhora *riged* [102], *hybrid* [103] a *fbm* [104]

7.3 Denní doba

Denní doba je faktorem, který není běžné komplexně rozebírat. Světelné zdroje v simulátorech a hrách jsou dynamické a tím pádem vznikají veškeré efekty stínů a světla automaticky. Pokud je ovšem potřeba docílit trochu většího výkonu, pak veškeré postupy používané v CG jsou dost pomalé. BRDF a další metody jsou navíc datově velmi náročné a v případě rozšířené reality by to mohl být velký problém. Pokud by se využily tyto techniky, pak by mohl nastat další problém a to rychlost výpočtu správné části v jedné BRDF materiál na mobilních zařízeních. Z těchto důvodů se bude volit jednoduchá textura daného modelu. Naprosto ideální stav by byl, pokud by veškerá zařízení dokázala počítat globální osvětlovací modely a pojmout veškeré modely daného města. V takovém případě by bylo pouze nutné získat data o vlnové délce světla v příslušném historickém okamžiku. Toto je ovšem ideální případ, který se zřejmě nedá zajistit v době vytváření této práce. Proto se využije zjednodušený model a to jedna textura s barvou, která, mimo jiné, obsahuje i informace o osvětlení.

Stále zde nebylo nic řečeno o problematice světel. Textura jako taková zvládne udržet spoustu informací, jak již bylo řečeno, a tedy i informace o osvětlení. Model s texturou obsahující tuto informaci umožňuje šetřit výkon, jelikož ho stačí pouze zobrazit a není potřeba složitě dopočítávat například stíny nebo vzájemné zastínění. Model s takovou texturou bude sám o sobě vidět i v tmavé scéně.

Z těchto důvodů bude zásuvný modul vytvářet několik množin textur, které budou reprezentovat různá časová období během jednoho dne i jednoho roku. Navrhovaný postup je takový, že se nastaví světla na příslušný měsíc společně s odstínem zdroje světla. Následně se spustí 24x zapékání textury a po každém zapečení se příslušná textura uloží s určitým formátem jména[19]. Přepočítají se některé hodnoty osvětlení a pokračuje se další hodinou. Po dokončení jednoho dne se nastaví další měsíc v roce a příslušné hodnoty osvětlení a spustí se další zapékání.

Jakým způsobem se bude otáčet se zdroji, vypočítávat odstín slunce nebo vypočítávat úhel slunce nad obzorem bude ukázáno v následujících sekcích. Dále je zmíněna možnost nočního osvětlení měsícem a nastíněna problematika s tím spojená. Jako poslední zde bude zmíněno umělé osvětlení.

7.3.1 Výpočet úhlu slunce nad obzorem

Při simulování denní doby je vhodné zhruba vědět, jak vysoko nad obzorem se slunce může nacházet. Tato skutečnost přidává další rozměr realističnosti v simulovaném prostředí. Takový úhel lze lehce spočítat, jediné, co je potřeba je trocha znalostí ze zeměpisu. Postačí dva jednoduché údaje zeměpisná šířka místa, kde má být spočítán tento úhel a zeměpisná šířka bodu, kam dopadají

[19] Volba jména je popsána v sekci 7.8.1

sluneční paprsky kolmo. Takto lze pro každý den i měsíc spočítat, jak vysoko vyjde slunce nad obzor. Pro zjednodušení se zde bude počítat pouze se třemi významnými daty a to rovnodennosti, letní a zimní slunovrat. Jednotlivé dny lze spočítat postupně dle následujících rovnic

$$equinox = 90^\circ - latitude \ , \tag{7.5}$$

$$soltice_s = 90^\circ + 23{,}5^\circ - latitude \ , \tag{7.6}$$

$$soltice_w = 90^\circ - 23{,}5^\circ - latitude \ . \tag{7.7}$$

Proměnná *latitude* bude údaj ve stupních, který říká zeměpisnou šířku místa pro výpočet. Číselný údaj 90° říká, že tento výpočet provádíme pro pravé poledne, kdy je slunce nevýše na obloze. Druhá číselná hodnota 23,5° označuje obratníky. Při letním slunovratu se tento údaj přičítá a při zimním naopak odečítá. Rovnice pro slunovrat jsou brány specificky pro severní polokouli, pro jižní stačí pouhá záměna znamének u hodnot obratníků. Zmíněné rovnice lze dohledat například v [105]. S výslednými údaji se více pracuje v dalších sekcích.

7.3.2 Otáčení zdrojů světla

Nejlepším řešením by bylo veškeré osvětlení řešit pomocí dynamických světel a globálního osvětlovacího modelu, to je ovšem v dnešní době trochu problém. Mobilní zařízení nemají takový výkon a i pro virtuální realitu by to mohl být problém. Velmi dobrým přístupem bude mít veškeré údaje o poloze slunce nad obzorem v příslušné historické době. Tyto údaje následně stačí číst, případně průměrovat a nastavovat potřebné části programu. Množství dat bude úměrné množství měřených dat. Také by záleželo, kolik by se vytvářelo textur z těchto hodnot. Existují nástroje, které umožní zjistit příslušné úhly až k roku 1900, to by ovšem pro projekt Věnných měst nestačilo. Proto se spíše zvolí výpočet, který nebude tak přesný jako skutečně naměřené hodnoty, ale bude dostačující.

Je potřeba také zmínit očividné a to, že úhly se v průběhu roku mění v závislosti na aktuálním měsíci. S naměřenými hodnotami se tato skutečnost sama nastavuje. Při výpočtu to bude potřeba zohlednit. Pro každý měsíc je jiný výchozí bod[20] při pravém poledni. Výchozí hodnoty lze dopočítat a proto zde dojde k dalšímu zjednodušení. Vycházet se bude pouze ze tří hodnot a mezi nimi se bude interpolovat. Tyto hodnoty jsou význačné dny v roce, jak již bylo zmíněno v sekci 7.3.1. Interpolace proběhne dle rovnice 7.8. Proměnné rat_{mv}, rat_{me} a rat_{ms} udávají poměr využití úhlu v daném měsíci, např.: 2,1,0. Zbývající proměnné $angle_w$, $angle_e$ a $angle_s$ značí úhly vypočtené v sekci 7.3.1 ve významné dny v roce. Rovnice vypadá následovně

$$angle = \frac{rat_{mw}}{3} \cdot angle_w + \frac{rat_{me}}{3} \cdot angle_e + \frac{rat_{ms}}{3} \cdot angle_s \ . \tag{7.8}$$

[20] Úhel slunce nad obzorem v pravé poledne.

Proměnná *angle* udává úhel v pravé poledne pro zadaný měsíc. Ten se následně využije jako výchozí bod a pomocí otáčení směru zdroje světla se vytvoří potřebný úhel pro danou hodinu. Samotné otáčení bude probíhat ve dvou osách a to v ose X a Z. Tato kombinace umožňuje simulovat pohyb po obloze, poslední dva kroky které je potřeba učinit jsou, nalézt jak rychle se má pohybovat zdroj světla v těchto osách. Na obě tyto rychlosti stačí jednoduchý výpočet, který následuje

$$step_x = \frac{(180° - (90° - angle))}{monthDay} , \tag{7.9}$$

$$step_z = \frac{360°}{hours} . \tag{7.10}$$

Číselné hodnoty 360° a 180° představují úhly, které je potřeba rozdělit. Hodnota ve jmenovateli *monthDay* je počet hodin, které je potřeba přibližně rozdělit. Proměnná *hours* udává počet hodin v jednom dni a *angel* úhel v pravé poledne v příslušném měsíci.

Při otáčení zdroji světla v závislosti na hodině je potřeba zjistit, jak se zdroje světla chovají v příslušném programu. Pak stačí provést příslušnou korekci vypočteného úhlu osvětlení vůči použitému nástroji a zajistit vše potřebné pro vytvoření textur.

7.3.3 Odstín slunce během dne

Odstín slunce lze pomocí různých přístrojů[21] změřit. To bohužel platí pouze pro současnost, z historického hlediska jsme odkázáni na odhad vývoje samotného slunce. Kromě úhlů, o kterých byla sekce 7.3.2, bude také vhodné rovnou načítat i odstín slunce v jednotlivých měřeních. Tím pádem opět vzroste datový objem potřebný pro přesné výpočty. Pro zjednodušení se vyjde z faktu, že přes léto jsou odstíny spíše do žluté a přes zimu spíše do bílé potažmo s lehkým nádechem modré. Tím vzniká i pocit studených a teplých barev. Ráno a večer jsou odstíny spíše dočervena a v poledne je odstín různý v závislosti na ročním období. Samotný výpočet je pak stejný jako v rovnici 7.8. Změna nastane v záměně úhlu za odstín v daném ročním období. Upravená rovnice vypadá takto

$$tone = \frac{rat_{mw}}{3} \cdot tone_w + \frac{rat_{me}}{3} \cdot tone_e + \frac{rat_{ms}}{3} \cdot tone_s . \tag{7.11}$$

Výsledný *tone* je jediné číslo, které se vloží do barvy zdroje světla ve scéně na pozici pro červenou a zelenou barvu. Tím vznikne různé tónování výsledné textury. V průběhu dne se pak odečítá nebo přičítá malá konstanta, která zajistí, že na začátku dne a k jeho konci je odstín spíše červený.

[21] Tyto přístroje se nazývají spektrofometry

7.3.4 Měsíční svit

Aby vzniklo skutečně ideální osvětlení nějakého modelu, pak je potřeba promyslet i osvětlení během noci. Kromě lamp zmíněních v sekci 7.3.5 je potřeba připočítat možnost přítomnosti měsíce. Měsíc sám o sobě nevydává žádné záření, pouze záření odráží. Odražené světlo má jinou vlnovou délku, protože v důsledku odrazu dojde k pohlcení určitých vlnových délek. Které vlnové délky se pohltí a které odrazí závisí na materiálu, který je přítomný na Měsíci. Tato skutečnost přidává další potřebné údaje k výpočtu osvětlení, tentokrát ovšem během noci.

Nutno podotknout, že měsíc není stacionární a má své pohybové cykly. Pohybuje se na oběžné dráze kolem země, která se také sama o sobě mění. Dále pak má měsíc různé fáze, zatmění atd. Pro historickou přesnost je potřeba veškeré tyto údaje posbírat a zavést. To by mohl být velký problém v některých případech. Z těchto několika bodů je vidět, že se výpočetní model značně zkomplikuje. Proto se s měsícem v této práci nepočítá.

7.3.5 Umělé osvětlení

Umělé osvětlení jako jsou lampy, baterky nebo pochodně jsou pouze rozšiřující prvek. Opět ideální bude znát polohu všech lamp z historické doby a podle polohy je přidat do scény. Tato světla pak přispějí k celkovému vzhledu v zapečené textuře. Kromě polohy lamp bude potřeba znát mnoho dalších parametrů. Jako příklad postačí různé chemické sloučeniny, které mají po vznícení různý efekt na plamen. Tím pádem světlo z lamp může být různé barvy. Proto se v této práci tato světla nebudou uvažovat. Přidání takových světel není těžké realizovat, stačilo by mírné rozšíření zadávacího skriptu a samotného zásuvného modulu.

7.3.6 Zatažený den

V průběhu dne se může vyskytnou jakékoliv počasí. Tím pádem se může obloha zcela zatáhnout a žádné přímé osvětlení není, pouze zhruba rovnoměrně odražené světlo. Tento efekt lze získat vcelku jednoduše pomocí několika osvětlení typu hemi. Jejich význam pro osvětlení je popsán v sekci 7.6. Při simulaci počasí pod mrakem lze hlavní světelný zdroj nastavit na nulu a příslušná ambientní světla nastavovat dle hodiny tak, jako když se slunce pohybuje nad mraky.

7.4 Roční období

Běžný fenomén jako jsou roční období není běžně řešenou problematikou. A pokud ano pak to většinou není kontinuální záležitost tak jako z reálného

života. Pro dokonalé napodobení ročních období bude zapotřebí spousta dat navíc, jak již bylo řečeno v kapitole 6.

Léto a zima jsou jednoduší části roku, hlavní viditelný rozdíl je sníh a holé nebo zarostlé stromy. Další podstatný faktor který byl již zmíněn v sekci 6.2 je odstín slunečních paprsků. Ten bude v zimních měsících do studena, tedy bílá, potažmo bílá s jemným nádechem modré. Zde by se hodilo mít správné hodnoty naměřené a následně je měnit podle měsíce u zdroje světla. Toto bylo již zmíněno v sekci 7.3, kde bylo nastíněno počítání osvětlení v jednom dni se zmíněným rozšířením na celý rok. Zde by bylo potřeba provést měření v průběhu celého roku. Tedy z každého dne v roce vznikne množina dat a v té se budou podle určitého klíče vybírat hodnoty. Věnná města zobrazují historickou podobu budov a tedy by bylo potřeba mít naměřené hodnoty v historickém kontextu. Takové měření lze jen těžko provést. Pro podzim a jaro je třeba mít více dat. Kromě odstínu osvětlení je třeba brát v potaz také okolní prostředí, které může být tvořeno například stromy, keři či jiným porostem. Ten na podzim shazuje listí a tím pádem se mění stíny. Kromě nich se také ve vzduchu objevují padající listy, které mohou uvíznout na budovách a tím také trochu změnit vzhled. Oproti tomu na jaře, kdy příroda opět ožívá, se mění pouze stíny a opět odstín paprsků.

Aby nemusela vznikat další geometrie, která by ztěžovala výpočty, bude minimem změna odstínu světla a případný sníh v zimních měsících. Sníh bude řešen pouze zbarvením příslušných částí bílou barvou. Vzhledem k množství potřebných dat, kdy některé by bylo téměř nemožné získat, bude vhodné vytvořit nějaké zjednodušení. Zde se nabízí všechna data zprůměrovat v každém měsíci a z nich dále vycházet. Nebo pro každý měsíc vybrat přibližně stejný den a z jednoho měření vytvořit potřebná data. Další možností je vytvořit jednoduchou interpolaci mezi několika hodnotami a tím zajistit rozlišení jednotlivých ročních období.

Tak jako se v sekci 7.3 určovaly tři úhly v období slunovratů a letní/zimní rovnodennosti, tak i zde se určí výchozí hodnoty. Ty byly empiricky zjištěny a jejich hodnoty jsou 0,6 pro zimu, pro jaro podzim 0,8 a pro léto 1,0. V kapitole 8 budou ukázány výsledky pro několik různých hodnot pro porovnání s těmito hodnotami. Interpolace mezi hodnotami je popsána v sekci 7.3.3. Kde se z výše zmíněných hodnot dopočítá odstín zdroje světla pro zadaný měsíc.

7.4.1 Sníh

Sníh je v mnoha ohledech unikátní a špatně se napodobuje kvůli svému chování, které je popsáno v sekci 6.1. Pro jednotlivé modely bude nejlepší nechat je zasněžit a následně bude možné využít různě zasněžené modely. Tímto postupem lze simulovat další skutečnost a to vrstvení sněhu. Jak již bylo řečeno, tento postup vytváří velké nároky na vykreslovací zařízení. Proto se využije bílé barvy, která se použije na specifické části. Opět pro docílení krátkého zobrazovacího času se využije pouze textur s úpravou specifických částí. Takto

upravená textura se následně uloží a je připravena k použití. Kromě barevné textury by bylo vhodné začlenit také spekulární mapu, ta se ale pro zjednodušení neuvažuje. Také bylo vhodné upravit normálové mapy modelu, ale stejně jako spekulární mapa se tato možnost neuvažuje.

Výpočet, který je zde prezentován, je částečně převzat ze semetrální práce J. Kravce[22]. Převzat byl pouze výpočet faktoru a následně mírně rozšířen s jiným výpočtem směru zasněžení, případně směru deště. Faktor bílé barvy místo barvy textury je následující

$$fac = \frac{\frac{dot1}{0,6} - (Snow_{level} - Snow_{softness})}{\max((Snow_{softness} \cdot 1{,}2), 0{,}001)} \quad . \tag{7.12}$$

Proměnné $Snow_{level}$ a $Snow_{softness}$ jsou dvě čísla na vstupu. $dot1$ označuje výsledek skalárního součinu normály textury a směru. Tento směr určuje odkud by teoreticky mohl vát vítr. Nicméně pro zmírnění efektu sněhu je třeba předchozí rovnici trochu upravit. Správné použití směru je popsáno v následující sekci.

7.4.2 Přidání směru k výpočtu

V ideálním případě budou známé meteorologické údaje z období, pro které se modely vytváří. Při znalosti dat se dá model ještě doplnit o směr, odkud vane vítr a tím pádem, kde bude model více zasněžený. Při znalosti směru větru a normál modelu budovy lze upravit model výpočtu, který bude přibarvovat bílou barvou pouze ty části, které jsou vystaveny příslušnému směru. Úprava rovnice 7.12 je následující

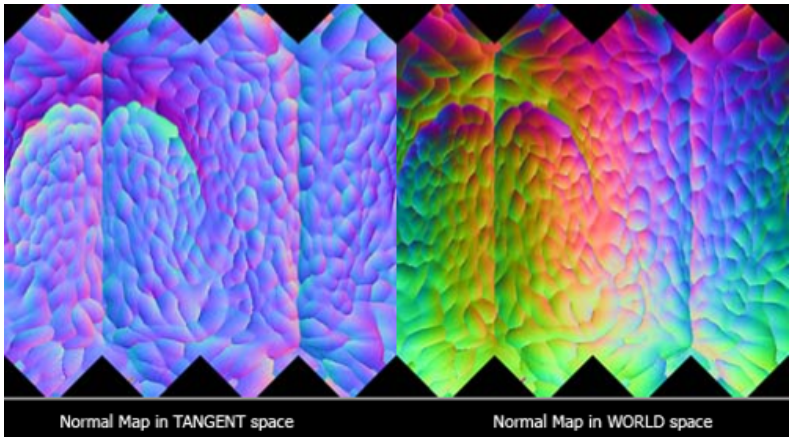
$$fac = \frac{\left(\frac{dot1}{0,6} - dot2\right) - (Snow_{level} - Snow_{softness})}{\max((Snow_{softness} \cdot 1{,}2), 0{,}001)} \quad . \tag{7.13}$$

Přidaná proměnná $dot2$ označuje výsledek skalárního součinu normál modelu a směru odkud vane vítr. Tím vznikne maska, která umožní zmírnit efekt sněhu na odvrácených stranách modelu. Tento model výpočtu je ovšem potřebný pouze v případě, že se použije nevhodný prostor normálových souřadnic. Při využití normál v prostoru objektu se z proměnné $dot1$ dostane pouze vzhled jako by příslušná plocha čelila vlivu. Pro získání výsledku bez potřeby přepočtu je vhodnější využít normály zakódované ve světových souřadnicích.

7.5 Déšť

Podobně jako v sekci 7.4 i zde bude potřeba velké množství dat o prostředí, aby bylo možné přiblížit se realisticky věrohodnému výsledku, velkým množstvím

[22]Semestrální práce z předmětu BI-PGA, téma bylo tak jako zde simulovat různé efekty počasí.



Obrázek 7.10: Normálová mapa v globálních a lokálních souřadnicích

dat jsou myšleny například meteorologické informace. Zde se jedná převážně o meteorologická data, jako příklad lze uvést intenzitu deště nebo sílu větru. Kromě údajů o počasí je potřeba přesná znalost použitého stavebního materiálu. Každý materiál má jiné sací vlastnosti nebo jakým způsobem bude vlhnout, další možná potřebná data byla uvedena v sekci 5.3.

Vzhledem k podobnosti deště a sněhu lze postupovat velmi podobně. Pro lepší výpočet bude vhodné přidat jako vstup i textury daného modelu. Zde bude hlavní změna v používaných barvách a stylu použití. Texturu obsahující údaje o barvách modelu lze různě změnit opět pomocí režimů prolnutí 2D obrázků. Jedním z těchto režimů je *overlay*. Ten lze, tak jako předchozí metody, vyjádřit pomocí výpočtu, který vypadá následovně

$$f(f,b) = \begin{cases} 2 \cdot f \cdot b, & \text{pro } b \leq 0,5 \\ 1 - 2 \cdot (1-f) \cdot (1-b), & \text{pro } b > 0,5 \ . \end{cases} \tag{7.14}$$

Tento režim se použije na texturu modelu a jako pozadí se zvolí tmavá barva. Podle této hodnoty bude nebo nebude viditelný efekt navlhlé zdi. Empiricky zjištěná hodnota, která dává vizuálně uspokojivé výsledky, je přibližně 0,09. Ukázka výsledné textury při použití barvy pozadí s touto hodnotou bude v kapitole 9. Hodnoty *pixel* a *normal* v následující ukázce jsou sobě odpovídající pozice ve dvou texturách.

Výpočet faktoru pro použití ztmavené textury, nebo původní textury je stejný jako v rovnici 7.12. Tato hodnota může být následně ještě upravena v případě nevyhovujícího výsledku. Zmíněnou úpravou je myšleno vynásobení nějakou hodnotou, která bude pozorovateli připadat rozumná. Výsledkem této části budou stejně jako v případě sněhu pouze jinak zbarvené původní textury. Použití změn v normálových mapách nebo přidaní spekulárních map se neuvažuje.

Vstup: Texture, **NormalMap**, *RS*, *RL*, DIR
Výstup: RainedTexture
fac ← spočítej *fac* z hodnot *RS*, *RL*, 1,2 a 0,001
for *each pixel in* ***Texture*** *and each normal in* ***NormalMap*** **do**
  dot1 ← spočítej skalární součin DIR a *normal*
  overT ← spočítej ztmavenou texturu z *pixel*
  Ntex ← ulož na příslušnou pozici v **Ntex** výsledek smíchání *fac*, *pixel* a **overT**
end
return **Ntex**

Algoritmus 4: Algoritmus pro získání zapršené textury, *RS* označuje jak jemně má být déšť zpracován, *RL* pak značí úroveň deště na dané textuře a DIR pak směr odkud prší.

7.6 Simulace ambientního osvětlení

Při osvětlování jednoho modelu může vzniknout jeden drobný problém, za předpokladu využití složitějších metod osvětlení jako je například Ray Tracing. Strany které nejsou vystaveny zdroji světla přímo nemusí být vůbec viditelné. Jinak řečeno jediná přítomná barva bude černá. To by mohl být problém zejména při vytváření osvětlení v jednotlivých hodinách jednoho dne a největší problém přijde pro noční osvětlení bez lamp a měsíce.

Jedno z možných řešení může být přidat různé objekty simulující okolní zástavbu nebo přímo modely s materiály, které stojí v okolí. Dalším řešením je přidat do scény další druh osvětlení a ten nastavit rozumně tak, aby i v noci byla barva modelu alespoň částečně vidět. Tento postup je opět nejjednodušší a bude použit. Druh osvětlení, které je na to vhodné je typ hemi. Tento druh osvětlení se přidá na osy X a Y v kladném i záporném směru. Dohromady se takto docílí viditelnosti modelu i v nočních hodinách. Jak již bylo zmíněno v sekci 7.3.6, takto rozvržená světla lze využít také k simulaci zataženého počasí.

7.7 Zadávací skript

Celý zásuvný modul by bylo vhodné z části automatizovat případně plně automatizovat. Představa je taková, že by byl přítomen zadávací skript, který by vzal určité parametry a postupně v závislosti na nastavení vytvoří zadané množiny dat. Díky možnosti spouštění různých skriptů přímo z příkazové řádky je automatizace možná. Bude vhodné aby skript uměl vytvořit příslušnou scénu, importovat potřebné modely, nastavit jejich materiály a následně předat parametry a zahájit vytváření textur.

Skript bude pracovat z aktuálního umístění, kde se předpokládá přítomnost všech potřebných souborů. Minimum těchto souborů se skládá ze souboru

daného modelu ve formátu FBX[23] a čtyři textury (základní barva, normálová mapa, roughness a ambient occlusion[24]). Pro jednoduché načítání a importování byla založena jednoduchá jmenná konvence. Model uložený ve formátu FBX bude muset být označen následujícím způsobem $filename$_LODX.FBX. Část $filename$ bude jeden z požadovaných vstupů pro skript, stejně tak LOD[25]X které udává o jakou úroveň detailu se jedná. Veškeré textury, které budou připravené pro dané modely, musí mít jména v následujícím formátu, který je jen rozšířením předchozího způsobu, $filename$_LODX_$type$.png. Nová část *type* označuje jednu ze čtyř zmíněných textur. Následuje ukázka očekávaného pojmenování souborů.

```
Kropatschka_LOD1.FBX
Kropatschka_LOD1_BaseColor.png
Kropatschka_LOD1_Normal.png
Kropatschka_LOD1_AO.png
Kropatschka_LOD1_Roughness.png
```

7.7.1 Možnosti skriptu

Který výstup je požadovaný by mělo být jasné rovnou z příkazové řádky. Proto bude mít skript několik možností, jak ho spustit. Následuje seznam navrhovaných přepínačů s krátkým popisem jejich činnosti.

-m, - -model povinná část skriptu, která udává jméno načítaného modelu, jedná se co case sensitive hledání souboru s modelem.

-ts, - -textureSize pro vytvoření cílového obrazu je potřeba znát velikost textury, tu se zásuvný modul dozví z tohoto přepínače.

-#L, - -#LOD poslední povinný prvek skriptu, který se využívá při načítaní jednotlivých úrovní detailů a jejich počet.

-s, - -sun nepovinný přepínač umožňující změnit sílu hlavního zdroje světla.

-H, - -Hemi podobně jako předchozí přepínač i tento mění sílu zdroje světla, zde je ovšem rozdíl jakému světlu se mění síla. Jedná se o pomocné světlo vydávající rozptýlené osvětlení.

-a, - -all tento přepínač umožní postupně vytvořit veškeré kombinace všech efektů.

-sl, - -snowLevel pro jednoduché nastavení úrovně zasněžení je zde připraven tento přepínač, ten bude mít výchozí hodnotu. Tím pádem lze použít i samotný přepínač.

[23] Proprietární formát souboru s 3D modelem

[24] Zastínění okolím, metoda pro výpočet zastínění modelů při výpočtu lokálního osvětlení

[25] LOD označuje Level of detail, opět jako u slova nody je to používaná zkratka a proto zde není překládána.

-r, - -rain podobně jako předchozí přepínač, jen nastavuje úroveň deště. Opět lze použít s nebo bez další hodnoty.

-c, - -clouds pouze říká zda bude zataženo či nikoliv.

Zadávací skript bude možné využít například následujícím způsobem.

```
>>> blender --python assigns_script.py --
            -m Kropatschka -ts 1024 -#L 2 -c
```

Spuštění výše uvedeného příkazu nastaví celou scénu, tedy importuje model Kropatschka (*-m Kropatschka*) a všechny textury, které k tomuto modelu přísluší. Přepínač *-#L 2* říká, že se budou importovat dvě úrovně detailů, jinak řečeno dva modely a osm textur. *-ts 1024* nastaví velikost výsledné textury na 1024×1024, ta se nejdříve vytvoří interně pro Blender a po vytvoření výsledku se uloží. Poslední přepínač je *-c*, ten říká, že se bude vytvářet sada textur, kde není přímé sluneční světlo. První řádek říká jak se má Blender spustit, tak to zadaný první řádek říká, že se spustí na pozadí - *-backgroud* a také, že se spustí externí python skript - *-python assigns_script.py*. Dvě pomlčky nakonci značí konec argumentů pro spuštění Blenderu z příkazové řádky. Spuštění Blenderu na pozadí nechá pouze proběhnout veškeré operace a následně se vypne. Pokud se tato možnost nepoužije, pak vše proběhne velmi podobně, jen po skončení práce zadávacího skriptu a zásuvného modulu zůstane Blender otevřený.

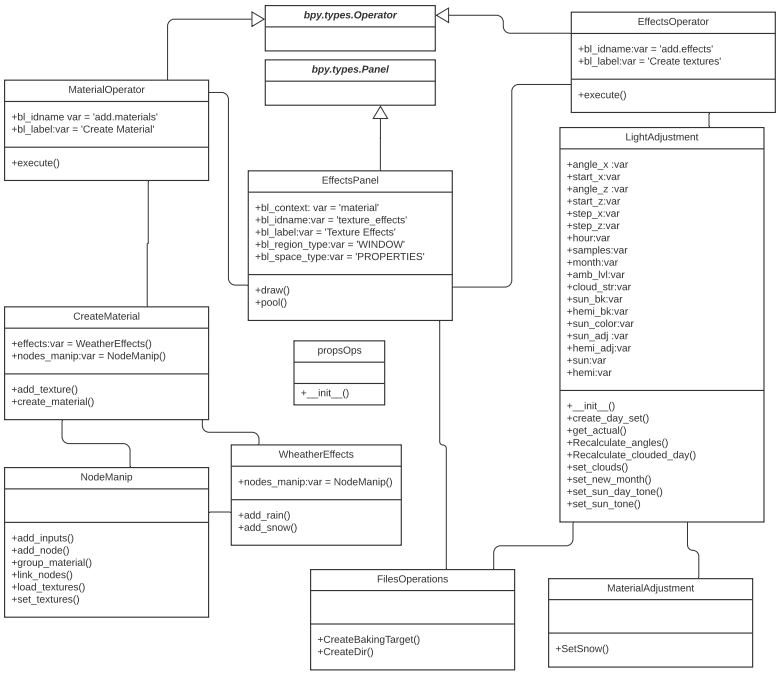
7.8 Návrh zásuvného modulu

Zásuvný modul musí obsahovat minimálně dvě třídy jimiž jsou XXXOperator a XXXPanel. Tyto dvě třídy zajišťují rozhraní a případný spustitelný operátor. V této práci se bude jednat o třídy EffectsOperator, MaterialOperaor a EffectsPanel

EffectsPanel vystavuje k dispozici GUI při interaktivním režimu Blenderu. Obsahuje zejména metodu pro vykreslení GUI a kontrolu přístupnosti. Blender 2.79b má celkem tři režim, Blender render, Cycles render a Blender game. Pro co nejlepší využití síly Blenderu se vyžaduje režim Cycles, který podporuje jednoduchou tvorbu materiálů. Tato třída dědí své vlastnosti ze stuktury, která je označena jako *PANEL*. Tato třída umožní udržet informace o UI elementech.

EffectsOperator a MaterialOperator zastupuje třídu pro vytvoření spustitelné procedury/metody. Hlavní metodou je *execute*, která říká, co se bude dít po zmáčknutí tlačítka, které je umístěné v GUI. EffectsOperator dědí ze třídy *OPERATOR*, která drží informace o spustitelné operaci či registraci.

Práci MaterialOperator bude možné snadno upravit ve smyslu které úrovně detailů budou potřebovat které textury. Prvotní návrh je nechat všechny textury pro nejdetailnější model. Pro zbytek bude stačit pouze barevná textura a normálová mapa. Toto nastavení by mělo jít snadno změnit doplněním



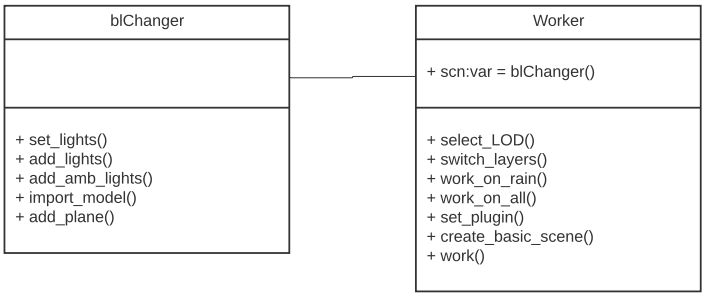
Obrázek 7.11: Class diagram zásuvného modulu

určitých míst kódu. Tato část kódu označuje výčet úrovní detailů, které mohou mít více než dvě textury. Ukázka je zahrnuta v kapitole 8.5.6.

Předchozí třídy jsou vzájemně provázány. Kromě předchozích tříd zde budou další významné třídy, a to FilesOperations a LightAdjustment, NodeManip, WeatherEffects a CreateMaterial. Třída NodeManip bude obsahovat metody na tvorbu a úpravu skupin nodů, jedná se hlavně o přidávání a linkování. WeatherEffects pak využívá NodeManip k tvorbě materiálů simulujících déšť a sníh. CreateMaterial bude vytvářet příslušný materiál a do něj přidá efekty počasí či texturovou skupinu, zajišťující základní materiálový náhled. FilesOperations bude zajišťovat práci se soubory a LightAdjustment bude pracovat se světly. Navíc bude poslední jmenovaná třída vytvářet samotné textury. Dále modul obsahuje několik podpůrných tříd.

7.8.1 Návrh jmenné konvence výstupu

Pro jedna složka textur pro denní dobu bez jakýchkoliv úprav vyjde přibližně na 288 textur. V takovém množství se bude orientovat jen velmi těžce. Proto bude vhodné zavést jmennou konvenci pro tyto soubory. Vyjde se z pojmenování, které je v sekci 7.7 a bude rozšířeno o potřebné informace. Navrhovaný vzor je následující $filename$_LODX_$type$_$date$.png. Nová část *date* značí o jaký měsíc a jakou hodinu se jedná. Pro zlepšení představy může výsledné



Obrázek 7.12: Class diagram zadávacího skriptu

jméno vypadat například následujícím způsobem.

```
Kropatschka_LOD1_BaseColor_M2_H15.png
```

Podobné pojmenování ponesou i složky, které budou obsahovat příslušné textury. Vzor pro pojmenování složky je velmi podobný jako pro textury $filename\_\mathrm{LOD}X\_mat$. Následuje ukázka pojmenování složky pro model *Kropatschka*, úroveň detailů jedna a označení *daytime*.

```
Kropatschka_LOD1_daytime
```

Zde část *mat* označuje množinu textur, kterou daná složka obsahuje. Možná označení pro *mat* jsou například: daytime(sun)clouds, LR, NR a HR. Jednotlivé části pak vyjadřují denní dobu, zatažené počasí a použité množství deště.

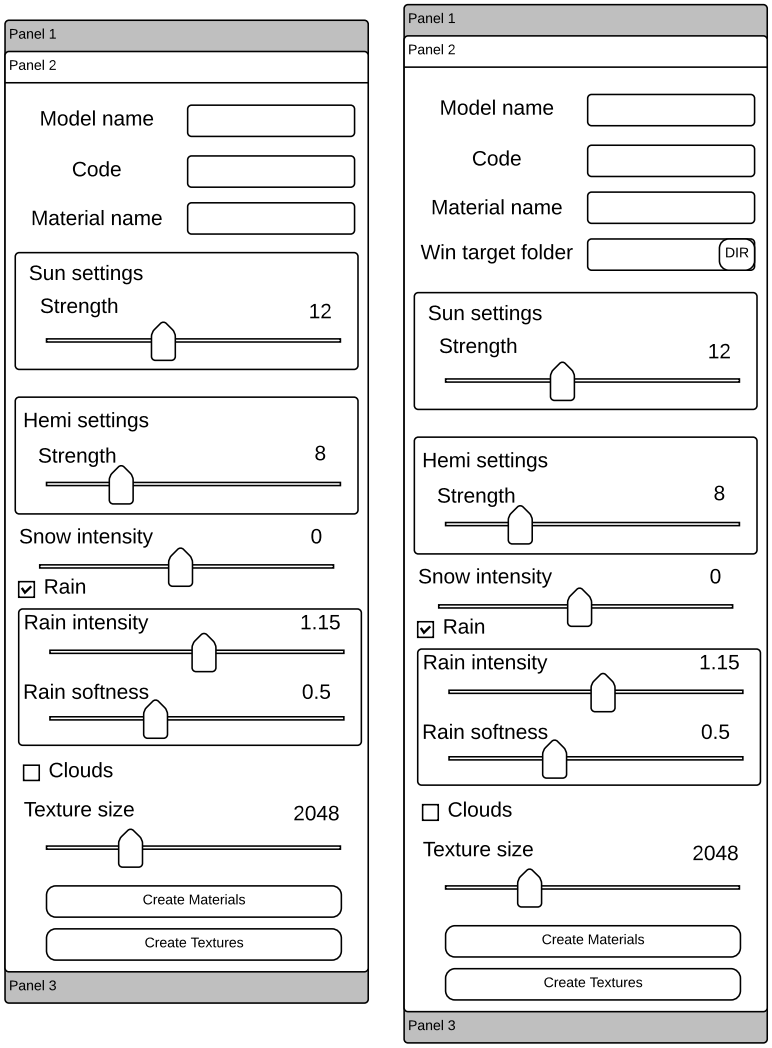
7.9 Návrh zadávacího skriptu

Zadávací skript je oproštěn od nutných náležitostí zásuvného modulu. Tedy návrh spadá zcela pod představivost tvůrce skriptu. Zadávací skript se bude skládat z následujících tříd blChanger a Worker.

Každá z těchto tříd bude mít na starosti malou část vytvoření potřebné scény a následné spouštění tvorby textur. Třída blChanger nastavuje celé prostředí Blenderu tak, jak je potřeba pro správné fungování zásuvného modulu. Worker zajišťuje zpracování všech požadavků a využití možností zásuvného modulu k dosažení zadaného výsledku.

7.10 Návrh rozhraní zásuvného modulu

Samotný zásuvný modul bude možné ovládat i přímo z Blenderu. Zde bude vhodné umístit nějaké GUI, přes které bude možné ovlivňovat různé části modulu. Zejména pak jméno modelu, jak se má označit ukládaná textura,



Obrázek 7.13: Návrh rozhraní pro Linuxové distribuce a Windows

jaká úroveň detailů má být použita, síla zdrojů světla atd. Tento panel bude pracovat zejména s materiálem a proto bude vhodné ho umístit k nastavení materiálu. Rozhraní bude předávat podobné informace jako jsou uvedené u zadávacího skriptu.

Vzhledem k různým přístupům k systémovým souborům bude potřeba i různě pracovat s ukládáním souborů. Pro systémy Linuxové báze stačí Blender spustit z příslušné složky. Ta následně slouží jako cíl pro ukládání souborů. V systému Windows je spouštění z určitého umístění trochu obtížnější, proto je lepší přidat do GUI další položku tak jak napovídá obrázek 7.13.

Při návrhu rozhraní je potřeba se pozastavit nad pojmenováním jednot-

livých vstupů. Z návrhu rozhraní je vidět jak budou pojmenovány jednotlivé složky. Zde byla snaha udělat názvy co nejjednodušší a zároveň výstižné. Pro označení *Code* se velmi těžce vymýšlel ekvivalent. Význam tohoto vstupu je jaké kódové označení má mít výsledná textura nebo složka, jak je naznačeno v sekci 7.8.1. Obsah boxů *Sun* a *Hemi* je přímo z Blenderu a odtud také vychází pojmenování položek.

Kapitola 8

Realizace

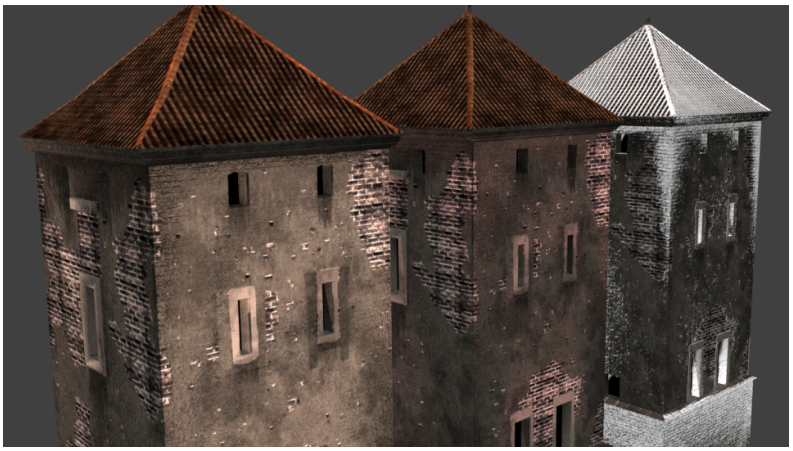
Z pěti zkoumaných a navržených efektů budou pro tuto práci zpracovány tři. Ty jsou spjaty s denní dobou, deštěm a ročním obdobím. V této kapitole bude nastíněna realizace a některé problémy, které bylo potřeba řešit v průběhu implementace.

8.1 Denní doba

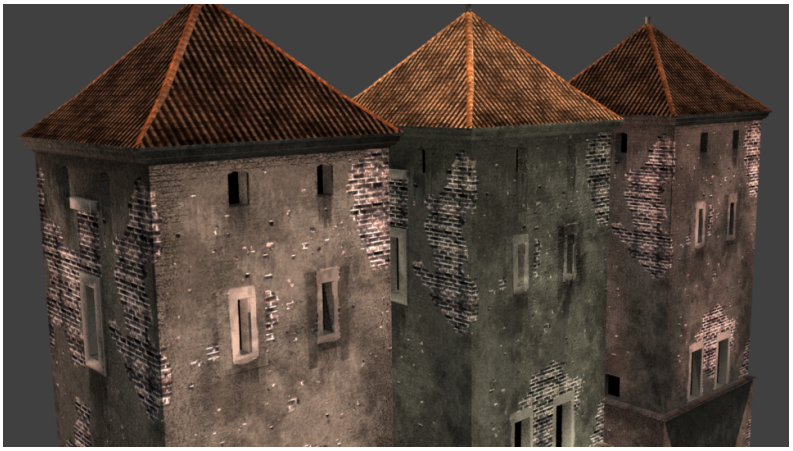
Výpočet úhlu slunce nad obzorem by bylo možné provádět při prvních fázích zásuvného modulu, ale kvůli nutnosti mít neustále zadanou správnou zeměpisnou polohu, bylo toto zjednodušeno. Modul má v sobě uložené tři předpočítané úhly, které následně využívá pro stanovení polohy slunce nad obzorem. Tyto úhly jsou navíc pro potřeby programu Blender přepočítány. Po přidání světla je v Blenderu vše nastaveno na nulu, to znamená, že světlo je v počátku souřadného systému, velikosti jsou výchozí a rotace okolo jednotlivých os jsou nula stupňů. Světlo v takovéto výchozí pozici má osvětlení ve směru záporné osy Z. Proto jsou úhly z výpočtu 7.5 částečně upraveny a to tak, že je každý odečten od 90°. Takto upravený úhel nad obzorem je už správně nastaven.

Rovnice 7.8 ukazuje, jak by bylo možné spočítat úhel slunce nad obzorem pro zadaný měsíc. Tato rovnice se ideálně hodí pro výpočty v realizovaném zásuvném modulu. Pro vytváření přibližně hodinových rozestupů u osvětlení lze opět využít navrhované rovnice 7.9 a 7.10. Zde bylo nutné rovnici 7.9 mírně upravit a místo *monthDay* použít číslo 12. Verze s proměnnou poskytovala velmi zvláštní výsledky. Později se ukázalo, že i část v čitateli potřebuje menší změnu a proto proměnná *angle* je pouze jeden úhel a to pro léto. Ve výsledku je $step_x$ statický a podává rozumné výsledky v průběhu celého roku.

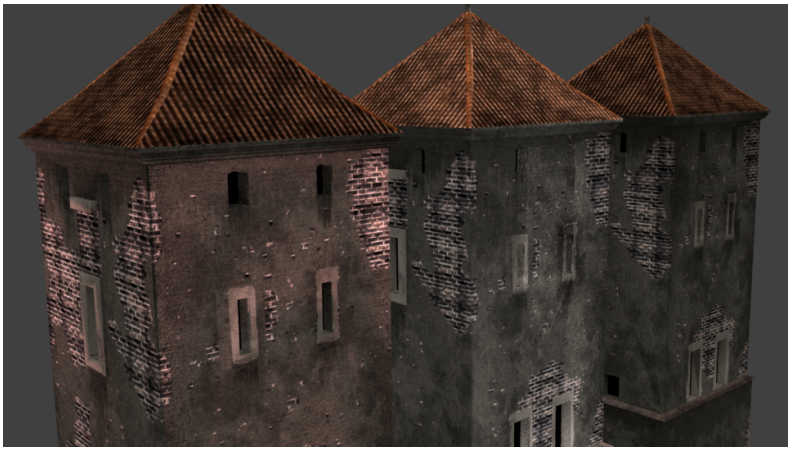
Podobný výpočet jako pro úhly slunce nad obzorem byl navrhován i pro výpočet odstínu slunce. Tento výpočet je reprezentován rovnicí 7.11. Opět není potřeba jakékoliv úpravy, výsledky které podává tento výpočet jsou uspokojivé.



Obrázek 8.1: Srovnání denní doby pro červen, září a prosinec v sedm hodin ráno



Obrázek 8.2: Srovnání denní doby pro červen v sedm, dvanáct a dvacet hodin



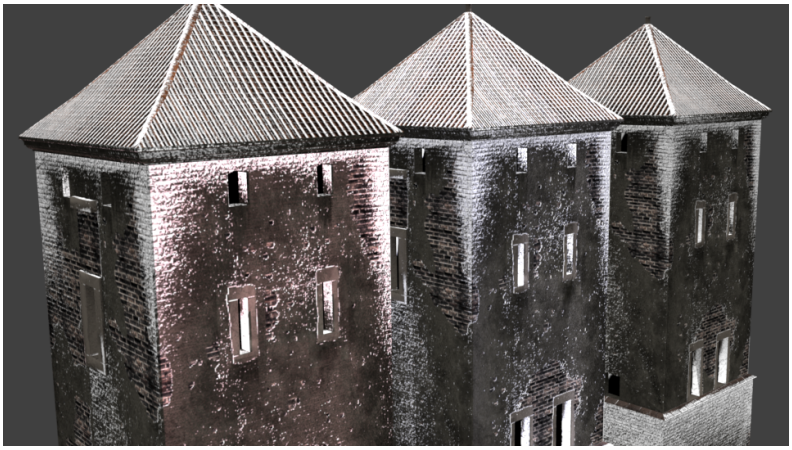
Obrázek 8.3: Srovnání denní doby pro říjen v sedm, dvanáct a dvacet hodin

Blender má několik druhů světel. Mezi nimi se nachází směrové jinak označené jako slunce a ve verzi 2.79b je pod označením hemi další typ směrového osvětlení, které je mírně rozptýlené. Pokud by se vzal pouze sluneční zdroj, pak ve většině nastavení budou osvětleny pouze strany modelu, které jsou světlu vystaveny. Zbývající stěny budou ve stínu, který bude téměř absolutně černý. Z tohoto důvodu je potřeba přidat další zdroj a to typ hemi, který zajistí osvětlení i pro ostatní stěny. Toto ovšem platí pouze pokud je ve scéně minimum modelů. V ostatních případech by se dalo přidaně osvětlení typu hemi vyřadit. Použité typy světel jsou tedy hemi a slunce. S oběma se pohybuje stejným způsobem, který byl popsán v sekci 7.3.

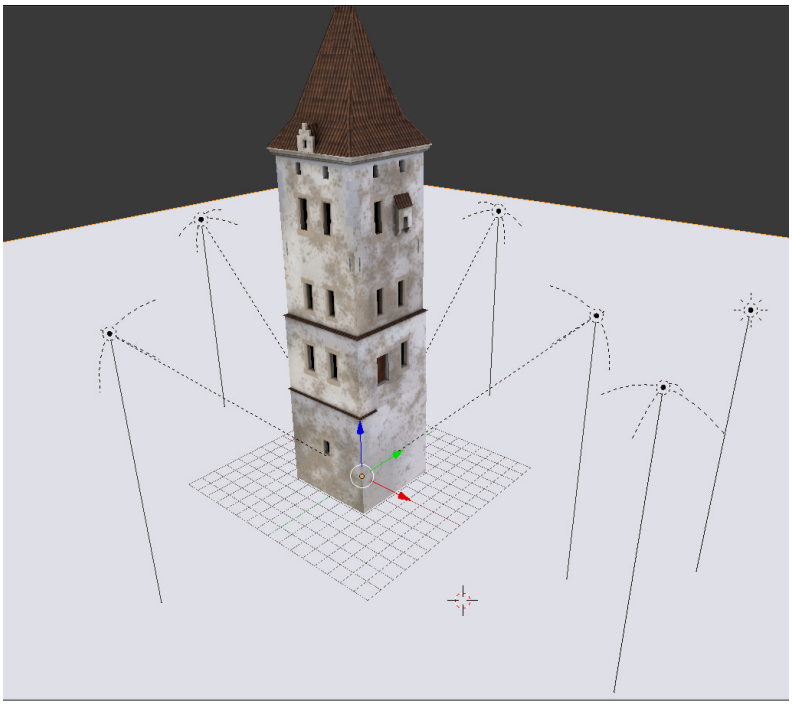
Další použitou částí pro zásuvný modul je simulace zataženého dne. Návrh na jednoduchou úpravu stylu osvětlení je použit. Tento postup je velmi jednoduchý. Hlavní osvětlení je vypnuto a jediné, co se mění, je ambientní osvětlení modelu. To je v poledne nastaveno na nejvyšší hodnotu, zde 8,0. V průběhu dne se tato hodnota zmenšuje/zvětšuje dle potřeby.

Sekce 7.6 naznačuje možné rozmístění světel ve scéně. Toto rozmístění je použité i pro výslednou práci zadávacího skriptu i zásuvného modulu. Každé ze světel označených jako „ambHemi“ je rozmístěno podél os X a Y v kladném i záporném směru a všechna směřují do počátku systému souřadnic.

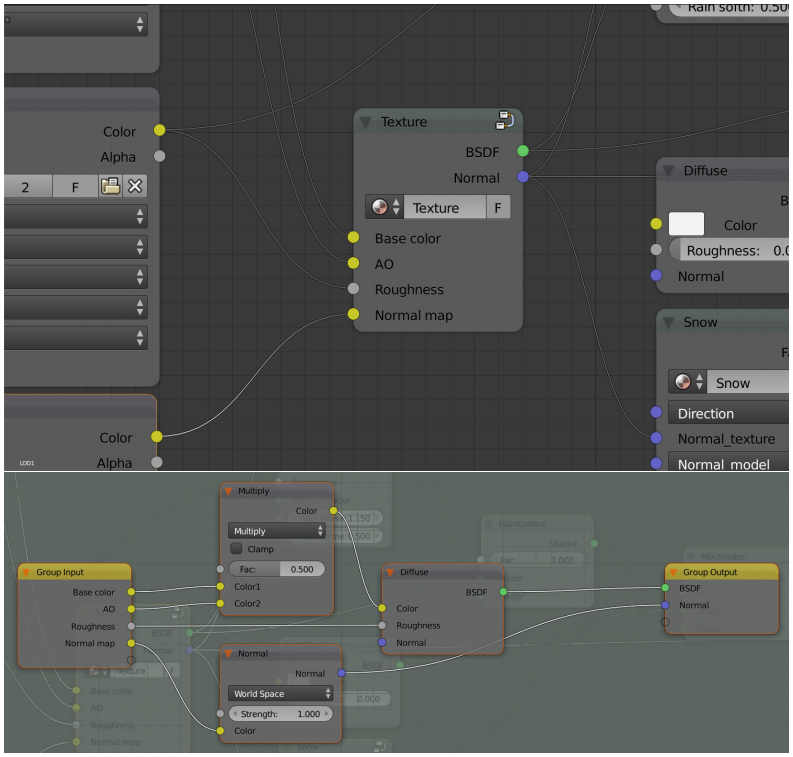
Veškeré ovládání světel, metody pro změny hodnot, výpočty a další různé potřebné metody či proměnné jsou uloženy ve třídě LightAdjusment. Tato třída zajišťuje vytvoření jednoho dne dle nastavených hodnot. Pro každý měsíc se přenastaví hodnoty a následně se znovu spustí.



Obrázek 8.4: Srovnání denní doby pro prosinec v sedm, dvanáct a patnáct hodin



Obrázek 8.5: Rozvržení světel pro rovnoměrné osvícení



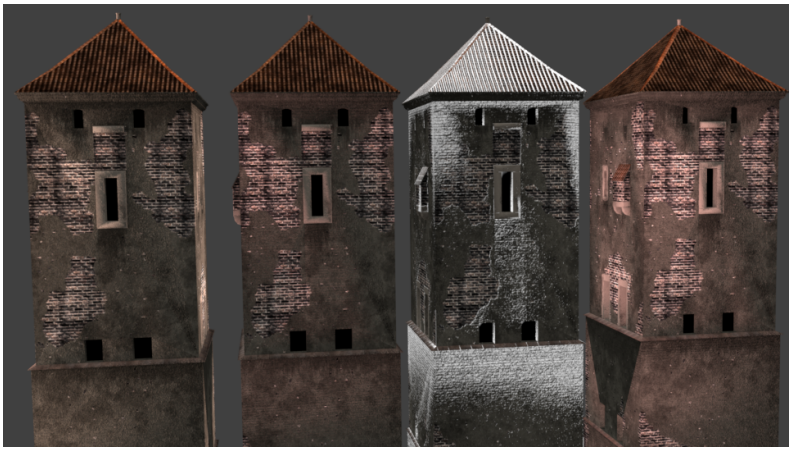
Obrázek 8.6: Skupina nodu pro vytvořeni základního náhledu na model nahoře, dole obsah této skupiny

8.2 Roční období

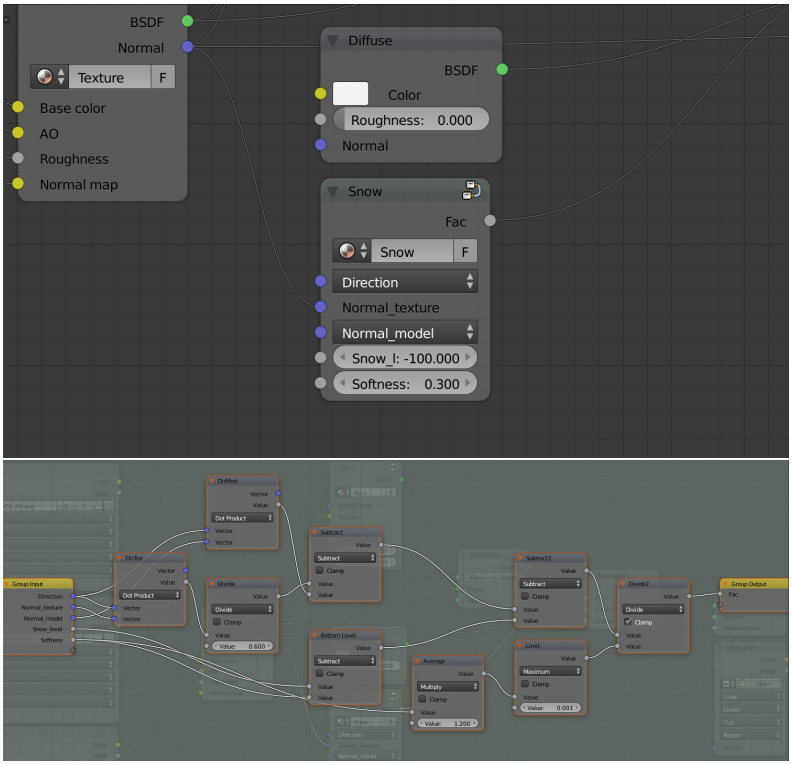
Odstín osvětlení modelů je již vyřešen v sekci 8.1. Tím pádem zbývá vyřešit přidání sněhu a případně směru větru pro věrohodnější výsledky. Výpočet uvedený v sekci 7.12 je možné využít, Blender a materiály Cycles umožňují vytvořit tento výpočet. Jak již bylo zmíněno v sekci 7.4.2 tento výpočet by stačil při použití normál ve světových souřadnicích. Pokud se použijí modelové souřadnice pak je potřeba využít upraveného výpočtu 7.13, který směr do určité míry simuluje.

Výpočet faktoru zasněžení je sloučen do skupiny[26], která má vstupy a výstupy. Její přesná ukázka je na obrázku 8.8.

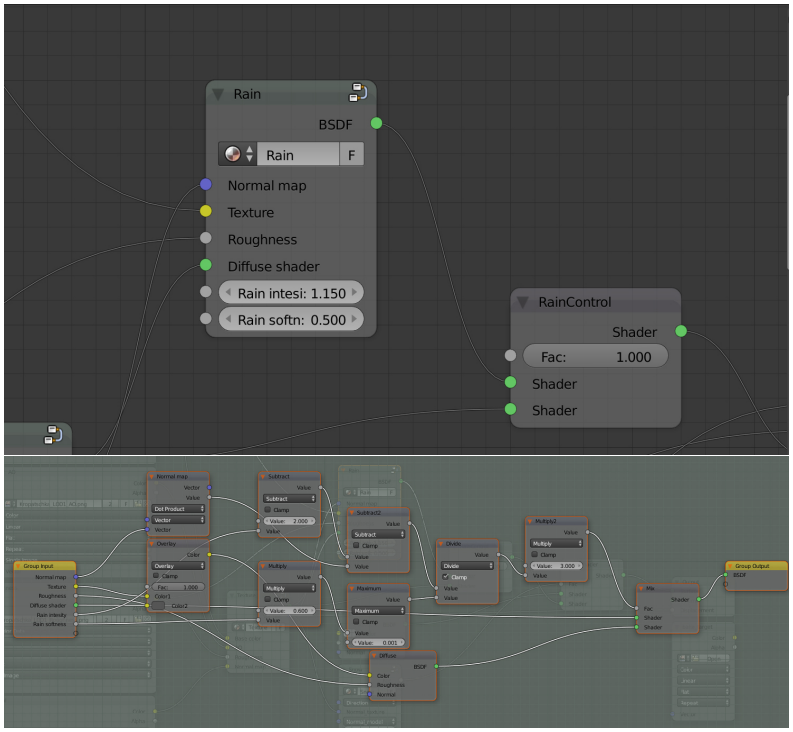
[26]Node groups – jedná se o zmenšení obsáhlých nastavení materiálů do menších a kompaktnějších celků.



Obrázek 8.7: Porovnání výsledných textur v různých ročních obdobích v průběhu roku, za každé roční období je zvolen jeden měsíc a je vždy vzata sedmá hodina ranní. Z leva červen, září, prosinec a březen



Obrázek 8.8: Skupina nodu pro vytvořeni efektu sněhu nahoře, dole obsah této skupiny



Obrázek 8.9: Skupina nodu pro vytvořeni vytvoření efektu deště nahoře, dole obsah této skupiny

8.3 Déšť

Navrhovaný postup ze sekce 7.5 je použit v celé své šíři. Původní textura se vezme a použije s možností prolnutí 2D obrazů v režimu *overlay*. Používaná rovnice 7.12 je převedena do výpočtu pomocí Cycles a využita pro výpočet faktoru, tak jak je vidět na obrázku 8.9. Pro úplnost je zde uveden i pseudo kód, který je převoditelný na nody. Nastavení ukázané na obrázku 8.9 vytváří výsledky viditelné na obrázcích v sekci 9. Ve stejné sekci bude také ukázáno porovnání různých hodnot

Vstup: Texture, **NormalMap**, *RS*, textitRL, DIR, **DC**
Výstup: RainedTexture
fac, **RT** ← spocitej dle algoritmu 4
RainedMaterial ← s pomocí *fac*, **RT** a **DC** spočítej nový material s deštěm return RainedMaterial;

Algoritmus 5: Přepis nodu z obrazku 8.9, **DC** označuje původní barvu textury, DIR směr odkud vane vítr, **RT** označuje zapršenou texturu

8.4 Zadávací skript

Celý návrh zadávacího skriptu byl převzat a vytvořen. Skript počítá se jmennou konvencí, která byla zavedena v sekci 7.7. To platí jak pro předávané modely tak pro textury s nimi svázané. Navrhované možnosti skriptu byly vytvořeny a po zjištění další možnosti použití rozšířeny o možnost **-n, - -none**. Tato malá změna byla zakomponována do zadávacího skriptu. Nově přidaný přepínač pouze nastaví celou scénu a importuje potřebné modely. Dále už je vše v rukách toho, kdo chce pracovat s příslušným modelem. Jednotlivé třídy a jejich metody jsou rozebrány v následujících sekcích.

```
>>> blender --python assigns_script.py -- -m Kropatschka
      -ts 2048 -#L 1 -n
```

8.4.1 blChanger

Pro nastavení celého prostředí Blendru slouží třída blChanger. Obsahuje podstatné metody, které umožňují přizpůsobit scénu potřebám zásuvného modulu. Mezi tyto metody pátří *set_lights*, která dle svých parametrů nastaví hlavní světelný zdroj (slunce a hemi) na sílu dle parametrů. Další metodou je *add_amb_lights*, ta přidá navrhovaný počet světel typu hemi a rozmístí je na příslušná místa. Kromě toho také nastaví jejich směr do počátku souřadného systému. Metoda *add_light* přidá do scény světelný zdroj dle svých parametrů a provede příslušné transformace, které jsou určeny ostatními parametry. Důležitou metodou je *import_model*, která dle svých argumentů importuje do scény zadaný model s příslušným počtem úrovní detailů. Dále zavolá některé metod ostatních tříd a vytvoří příslušný materiál. Poslední metodou této třídy je *add_plane*. Tato metoda přidá do scény poslední prvek a to plochu, na které importované modely budou stát.

8.4.2 Worker

Tato třída, jak napovídá její název, pracuje na všech dílčích úkonech pro vytvoření množin textur. Pro dosažení zadaného výsledku využívá metody dříve jmenované třídy a několik vlastních. Hlavní metodou této třídy je *work*. Ta vyčistí výchozí scénu, vybere první vrstvu jako výchozí a vytvoří základní rozložení scény, tu také nastaví. Následně dle argumentů z příkazové řádky začne spouštět zásuvný modul. K dosažení těchto částí využívá další metody, mezi které patří *create_basic_scene*, ta na základě svých parametrů vytvoří základní scénu a nastaví některé podstatné části. Metoda *set_plugin* nastaví dle parametrů výchozí hodnoty pro celý plugin. Mezi pracovní metody patří *work_on_all*, ta postupně vytváří jednotlivé textury pro každou z kombinací efektů. Další metodou je *work_on_rain*. Ta zpracovává požadavek na vytvoření deštivého dne, který se řídí předanými parametry. Metoda *switch_layers* pouze

```
def import_model(self, nlods, name):
        bpy.ops.object.select_all(action='DESELECT')
        for i in range(nlods):
            file = name + '_LOD' + str(i + 1)
            bpy.ops.import_scene.fbx(filepath= './' + file +
            ↪  '.FBX')
            bpy.context.scene.objects.active =
            ↪  bpy.context.selected_objects[0]
            bpy.context.object.name = name + '_LOD' + str(i + 1)
            bpy.data.scenes['Scene'].code = 'LOD' + str(i + 1)
            bpy.ops.add.materials()
            bpy.context.object.layers[i + 1] = True
            for i in range(i+1):
                bpy.context.object.layers[i] = False
```

Listing 1: Metoda třídy blChanger, která zajišťuje import modelu na základě jmenné konvence, která byla zavedena v sekci 7.8.1

přepíná aktivní vrstvy, ze kterých se budou vytvářet textury. Poslední metodou této třídy je *select_LOD*, která dle jména a čísla, které jsou předány jako parametry, vybere správný model ke zpracování.

8.5 Zásuvný modul

Celý zásuvný modul je napsán dle návrhu. Obsahuje tedy několik hlavních tříd a pak také pár menších tříd. Pro každou ze tříd bude uvedena její význam a metody, které využívá. Kromě všech tříd obsahuje zásuvný modul několik funkcí, které jsou dobré pro vytváření dalšího obsahu, nebo úkony potřebné k přidání zásuvného modulu. Nakonec také obsahuje podstatné globální proměnné, které pomáhají při různých výpočtech.

8.5.1 NodeManip

Tato třída zajišťuje základní operace s nody. Kromě nodů se zde také pracuje se samotnými skupinami a přidávání nebo nastavování obrázků jak textur. Při vytváření skupin je také potřeba přidat jejich vstupy, tato funkcionalita je také obsažena v této třídě. Tyto metody jsou jmenovitě *add_node*, *link_nodes*, *add_inputs*, *group_material*, *load_textures* a *set_textures*.

8.5.2 WeatherEffects

WeatherEffects obsahuje dvě metody, ty využívají NodeManip k sestavení potřebných struktur na dva hlavní efekty. První metoda *add_rain* přidá do

```
    def create_basic_scene(self, curScn, args):
        print('Configuring scene')
        #configure used resources
        curScn.render.engine = 'CYCLES'
        curScn.cycles.device = 'GPU'
        print('Inserting necessary objects')
        #add plane as bedding for model
        self.scn.add_plane(30)
        #add lights for whole scene
        self.scn.add_light(args.sunStr, [20,5,15],'SUN', 0.5)
        self.scn.add_light(args.hemiStr, [20,-5,15],'HEMI',
        ↪  shadow=False)
        self.scn.add_amb_lights()
        print('Importing model {}'.format(args.modelName))
        #import model
        self.scn.import_model(args.numLODS, args.modelName)
```

Listing 2: Metoda třídy Worker, která vytvoří základní scénu a nastaví potřebné části

materiálu, který je předán parametrem, skupinu s označením *Rain*, pokud tato skupina neexistuje, pak ji vytvoří. Obrázek 8.6 ukazuje jak bude vytvořená skupina vypadat.

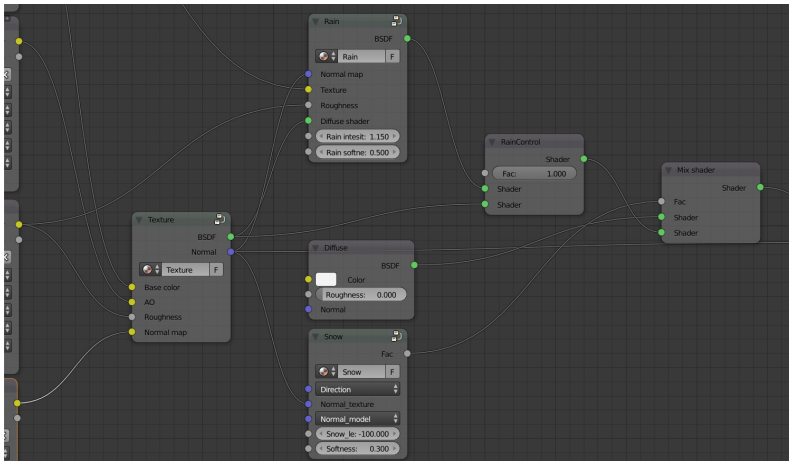
Podobně metoda *add_snow* přidá do materiálu, skupinu označenou jako *Snow*. Opět platí, že pokud neexistuje, pak ji vytvoří. Ukázka vytvořené skupiny je na obrázku 8.8.

8.5.3 CreateMaterial

CreateMaterial znovu vytváří materiály příslušným modelům. Kromě toho jim také přidává zjednodušenou skupinu nodů na základní vzhled. Využívá k tomu dříve jmenované třídy ze kterých poskládá materiál.

Mezi metody této třídy patří *add_texture* a *create_material*. První jmenovaná metoda vytváří, podobně jako *add_rain*, skupinu nodů, ty slouží ke zjednodušení zapojení všech textur. Obrázek 8.6 ukazuje obsah této skpiny.

Druhá metoda vytváří materiál jako celek. K tomu využívá dříve zmíněné metody *add_XXX*. Dále přidá další podstatné části tohoto materiálu jak je vidět na obrázku 8.10. Mezi tyto části patří shader simulující sníh, faktorové ovládání deště/sněhu atd.



Obrázek 8.10: Kompozice výsledného obrázku

8.5.4 EffectsPanel

Jedna z potřebných tříd, pokud má být vykreslováno GUI. Aby tato třída měla požadované vlastnosti, tak musí dědit z Blender API (bpy) ze specifické třídy. Touto třídou je *bpy.types.Panel*. Dále obsahuje dvě metody. První je označena jako *pool*, tato metoda hlídá zda má být příslušný panel vykreslen do Blenderu. Druhou metodou je *draw*, která do zadaného kontextu vykreslí požadovaný panel. Ten může obsahovat různé prvky GUI, které jsou definovány jako *bpy.props.XXXProperty*. Tyto prvky mohou být různého druhu například posuvníky, zaškrtávátka, barvy. Zde jsou využity zejména posuvníky, „zaškrtávátka“ a řetězce.

8.5.5 EffectsOperator

Další velmi podstatná třída, která obsahuje spuštění samotného vytváření textur. Obsahuje jednu metodu a to *execute*. Po vykreslení tohoto operátoru, se v GUI zobrazí tlačítko, na které je možné kliknout a tím spustit provádění zadané sekvence. Tato třída opět musí dědit z Blender API, děděná třída je *bpy.types.Operator*.

8.5.6 MaterialOperator

Tato třída je velmi podobná třídě EffectsOperator. Hlavním rozdílem je zde funkcionalita v metodě *execute*. V předchozím případě se postupně volalo vytváření textur. Zde se jen nastaví materiál dle zadaného parametru v GUI.

```
def draw(self, context):
        # layout for everything
        layout = self.layout
        #add properties to layout - string properties
        layout.prop(context.scene, 'model')
        layout.prop(context.scene, 'code')
        layout.prop(context.scene, 'mat_name')
        # for Windows platform add one more part
        if platform.system() == 'Windows':
            layout.prop(context.scene ,'w_path')
        # create bordered parts for light setting
        sun_box = Create_box_lamps(layout, 'Sun', 'Sun
        ↪  Settings')
        hemi_box = Create_box_lamps(layout, 'Hemi', 'Hemi
        ↪  Settings')
        # properties for numeric values
        layout.prop(context.scene, 'snow_level')
        ...
```

Listing 3: Metoda draw třídy EffectsPanel, tato metoda vykresluje GUI do rozhranní Blenderu, dle hlavičky celého modulu bude tento panel v záložce pro materiály

8.5.7 FilesOperations

FilesOperations pomáhá vytvářet složky a cíle pro zapékání textur. Má tedy dvě metody *create_dir* a *create_baking_target*. První metoda vezme svůj argument a pokusí se vytvořit cílovou složku. Jak již bylo zmíněno v návrhu, je třeba brát ohled na jinou práci se složkami v různých systémech. Druhá metoda pouze přidá obrázek, který bude vnitřně Blenderu sloužit jako cíl zapečení požadované textury.

8.5.8 LightAdjustment

Třída LightAdjustment má na starosti práci s nastavováním všech podstatných částí okolo světel, jejich tónování a další. Mezi její metody patří *set_clouds*, která nastaví osvětlení pro zatažený den. *Recalculate_clouded_day* je využívána ke změně intenzity osvětlení v průběhu dne, pokud je zvoleno zatažené počasí. Podobně *Recalculate_angles* mění osvětlení, v tomto případě však otáčí s hlavním zdrojem světla. Pomocná metoda *get_actual* umožňuje získat aktuální údaje pro úhly a tónování hlavního zdroje světla. Parametrem této metody je soubor hodnot, ze kterých se má spočítat aktuálně potřebná hodnota. Metody *set_sun_tone* a *set_sun_day_tone* nastavují správné odstíny zdroje světla

```
    def set_new_month(self):
        material = MaterialAdjustment()
        print('\n_____________________________________')
        print('Next calculated month will be ' +
        ↪  str(self.month))
        # set correct start angles for sun
        self.angle_x = self.start_x =
        ↪  self.get_actual(sun_angles)
        self.sun.rotation_euler[0] = self.hemi.rotation_euler[0]
        ↪  = self.start_x * pi/180
        self.sun.rotation_euler[2] = self.hemi.rotation_euler[2]
        ↪  = 0 * pi/180
        # check if snow is needed
        if self.month > 11 or self.month < 3:
            material.SetSnow(bpy.context.scene.snow_level)
        else:
            material.SetSnow(-100.0)
        ...
```

Listing 4: Metoda třídy LightAdjustment, která nastavuje potřebné hodnoty pro nový měsíc

pro jeden den pomocí první metody. Druhá interpoluje mezi teplejšími a studenějšími odstíny pro jeden den. Výpočet potřebného úhlu pro osvětlení provádí metoda *set_new_month*, která z hodnot uvedených v příslušné proměnné získá aktuální nastavení a následně ho nastaví hlavnímu zdroji světla. Poslední metodou této třídy je *create_day_set*. Ta využívá dříve uvedených tříd a vytváří tak jednotlivé množiny textur. Ty ukládá do složek dle svých parametrů.

8.5.9 MaterialAdjustment

MaterialAdjustment je třída připravená pro možné rozšíření. Zatím obsahuje metodu pro nastavení skupiny *Snow*. Metoda, která tuto manipulaci zajišťuje, je označena *SetSnow*. Zde je možné přidat další metody upravující například zbarvení sněhu, hrubosti nebo další skupiny nodů.

8.5.10 PropsOps

Jednotlivé prvky GUI jsou uloženy přímo ve scéně a tato třída inicializuje veškeré potřebné prvky zásuvného modulu. Obsahuje pouze metodu *__init__*. Při deklaraci proměnné typu PropsOps se vyvolá tato metoda a vytvoří ve scéně všechny potřebné položky pro GUI.

8.6 Doprovodné metody a globalní proměnné

Kromě výše zmíněných tříd obsahuje zásuvný modul dvě další metody. Tyto metody vytváří v GUI ohraničené boxy do kterých je možné vkládat nebo napojovat další části Blenderu. Názvy těchto metod jsou *Create_box* a *Create_light_box*. Verze pro světla přijímá jako argumenty jméno daného boxu a jméno světla, pro které je potřeba vytvořit box. Druhá verze vyhledá v zadaném materiálu příslušný node a také pojmenuje box. Pro obě verze platí, že jejich obsah je nějaké nastavení hledaného nodu.

Globální proměnné, které využívá zásuvný modul jsou následující

months_ratio určuje poměr různých hodnot, které je nutné počítat,

sun_stop udává pro každý měsíc hodinu, kdy se má světelný zdroj zastavit v otáčení a nastavit na nulu v síle osvětlení,

sun_angles jsou úhly v jednotlivých významných dnech, tyto úhly jsou již upraveny dle potřeb Blenderu,

sun_tone tato proměnná určuje zbarvení zdroje světla ve významných dnech roku,

sun_min_day_tone podobně jako předchozí určuje zbarvení zdroje světla s tím rozdílem, že zde se jedná o nejnižší možnou hodnotu.

Pouze první dvě proměnné mají prvky složené z více hodnot, tak jak to umožňuje konstrukce slovníku a list v Pythonu. Pro proměnné, které mají jen tři hodnoty, platí že jsou interpolovaný dle poměru hodnot uvedených v proměnné *months_ratio*.

Kapitola 9

Testování

Tato kapitola obsahuje ukázky volených hodnot pro různé části zásuvného modulu a výsledky uživatelského testování. Pro porovnání jsou v sekci 9.2 ukázány další hodnoty, metody atp. Obě části, které jsou výsledkem této práce, byli podrobeny uživatelskému testování. V sekci 9.1 jsou shrnuty výsledky s možnými opravami, které vyšly jako kritická místa z testování.

9.1 Uživatelské testování

Testování bylo provedeno na základě několika scénářů, které jsou obsaženy na přiloženém médiu pod názvem *user_testing_scenerios.pdf*. Předmětem testování bylo zejména ovládání zásuvného modulu a zadávacího skriptu. Vizuální spokojenost byla testována na upraveném zásuvném modulu, který vytvářel menší množství textur pro každou možnost. Tato úprava byla použita, protože i na silnějších počítačích by výpočet velkého množství velkých textur mohl trvat nepřiměřeně dlouho. Testování bylo z časových důvodů provedeno s malým množstvím testerů.

Testování ukázalo, že u jedné části GUI není zcela zřejmé, co znamená. Tato skutečnost byla jedním z očekávaných výstupů. Položky v GUI označené jako *Model name*, *Code* a *Material name*. Tyto tři části vytváří dohromady jména složek, do kterých se budou ukládat výsledné textury a také jak se budou jmenovat výsledné textury. zde by bylo vhodné vymyslet přesnější označení, nebo doplnit o ukázky, případně živé generování jména.

Dalším matoucím prvkem bylo ovládání intenzity deště, která se ovládá obráceně, než testeři očekávali. Hodnoty od jedné k nule dávají vcelku zajímavé výsledky pro model Kropatschka v úrovni detailů dva (případně jedna u původního zdroje). V momentě kdy se zadá hodnota od zhruba dvou dál, tak je materiál zcela čistý. Podobně tomu je v případě sněhu. Zde je výchozí hodnota nula, což může být mírně matoucí. Zde naštěstí platí, že čím vyšší hodnota, tím více sněhu. Oba tyto problém je možné vyřešit například tím, že se přidá přepočet na hodnotu potřebnou pro fungování celého materiálu. Jinak řečeno

takový výpočet, aby hodnota vystavená do GUI byla při základním nastavení jedna a od ní níže budou efekty ustávat. Naopak výše bude efekt nabývat vyšší intenzity. K obou těmto hodnotám patří i nastavení jemnosti efektu. Tato položka byla také jen s těží pochopitelná. Zde by stálo za úvahu ji úplně vyškrtnout nebo dodat pořádné vysvětlení.

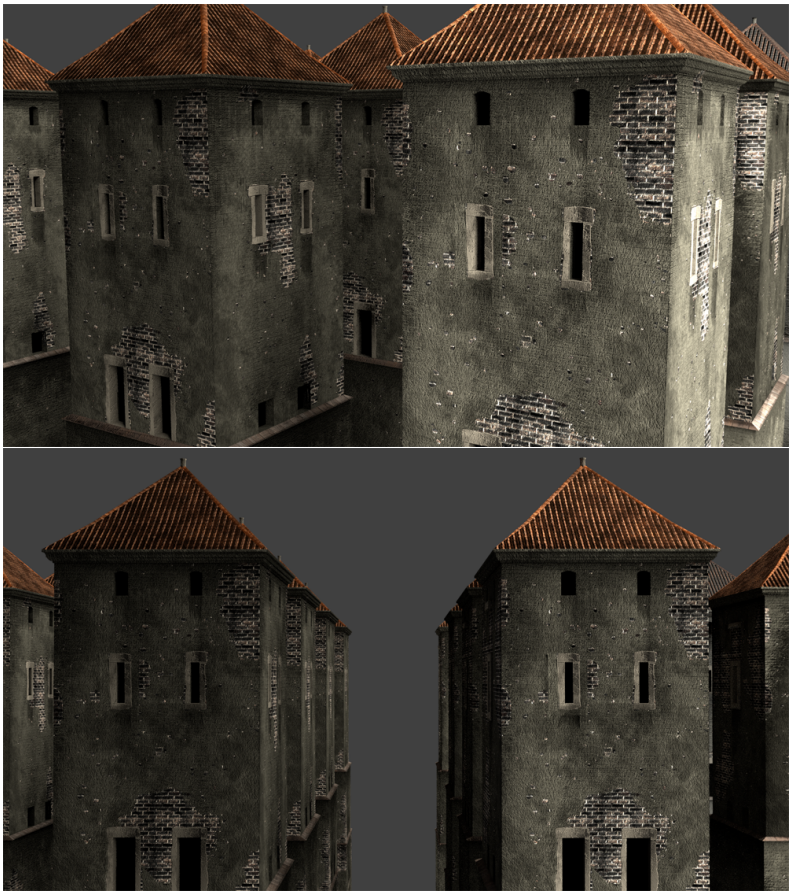
Kromě chyb v GUI bylo výsledkem testování také zjištění některých implementačních chyb, které byly následně odstraněny. Mezi tyto chyby patří špatné vytváření jmen pro cílové složky nebo špatná konverze hodnot zadaných ze zadávacího skriptu.

Testeři také uvedli, že by bylo vhodné mít některé další možnosti k dispozici. Za bolestivé místo byla označena doba trvání vytvoření celého roku v momentě, kdy se chce uživatel pouze podívat jak by při daných hodnotách vypadal příslušný model. Zde by bylo vhodné přidat možnost vytvoření vzorku pro jednotlivé efekty a ty ihned ukázat uživateli. Předchozí platí jak pro GUI tak pro zadávací skript. Další velmi dobrou připomínkou testerů bylo přidání možnosti ovládat výchozí hodinu i měsíc, případně kolik přesně měsíců se má počítat nebo jaké. Poslední podstatnou připomínkou bylo umožnění vidět výsledek při úpravě intenzity sněhu. Všechny tyto body jsou dobré pro rozšíření v navazujících pracích.

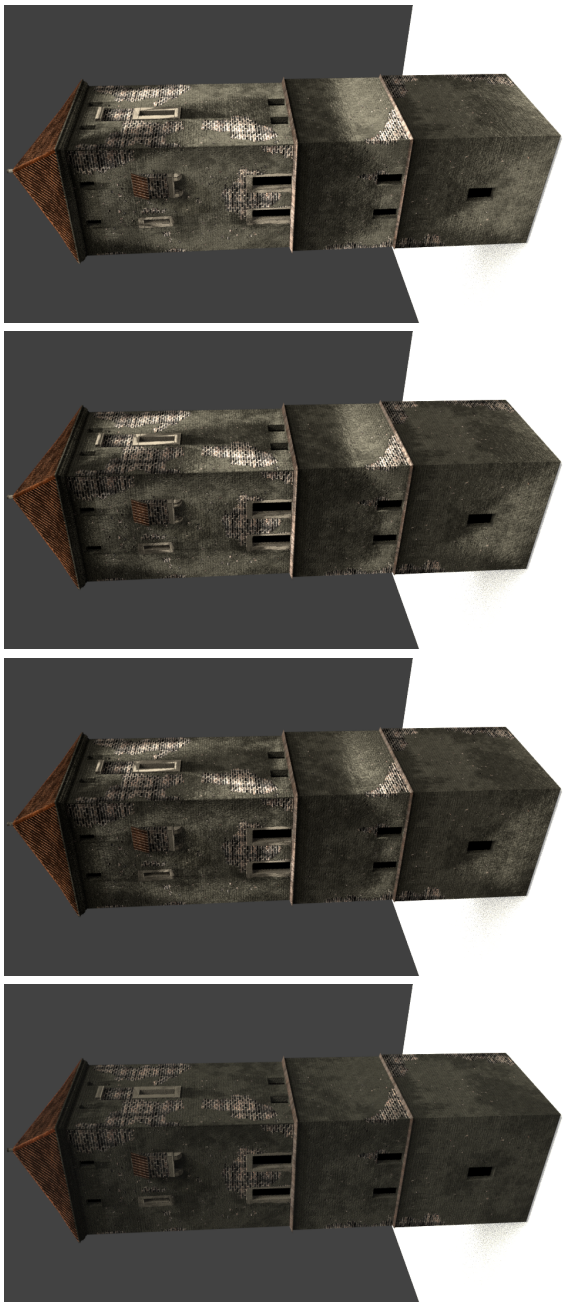
9.2 Ukázky volených hodnot pro porovnání

V této sekci jsou ukázky různých hodnot pro různé části zásuvného modulu. Jedná se pouze o části, které jsou vystavené do GUI v Blenderu. Postupně se lze podívat na různé hodnoty pro efekt blednutí, deště a následně sněhu. Také zde lze nalézt ukázku simulace zataženého počasí.

Obrázek 9.1: Různé hodnoty pro blednutí textury, první má nastavenou barvu pozadí při míchání barev na 0,0. Druhá ukázka využívá hodnoty 0,35. Jak je vidět zde není rozdíl až tak viditelný

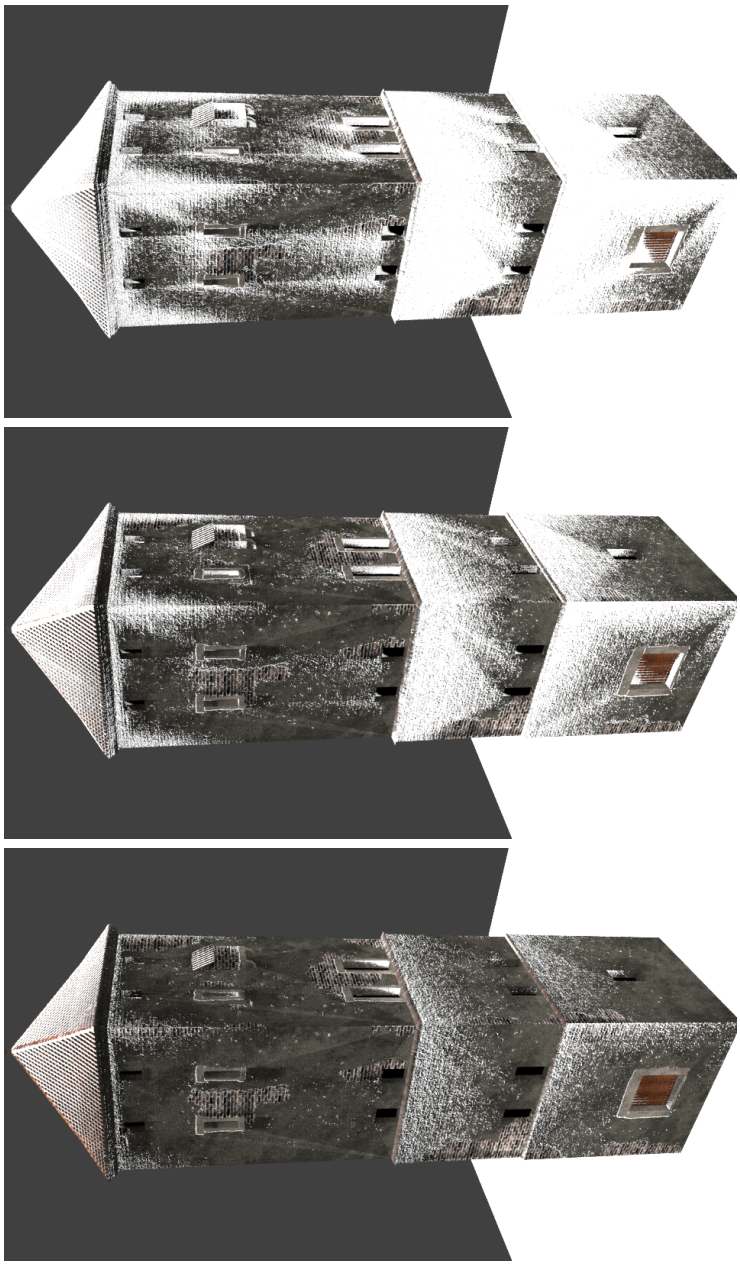


Obrázek 9.2: Doporučená hodnota pro model na renderu je 0,7, jak je vidět podává vizuálně uspokojivý výsledek. Poslední render ukazuje porovnání předchozí hodnoty s normální texturou.

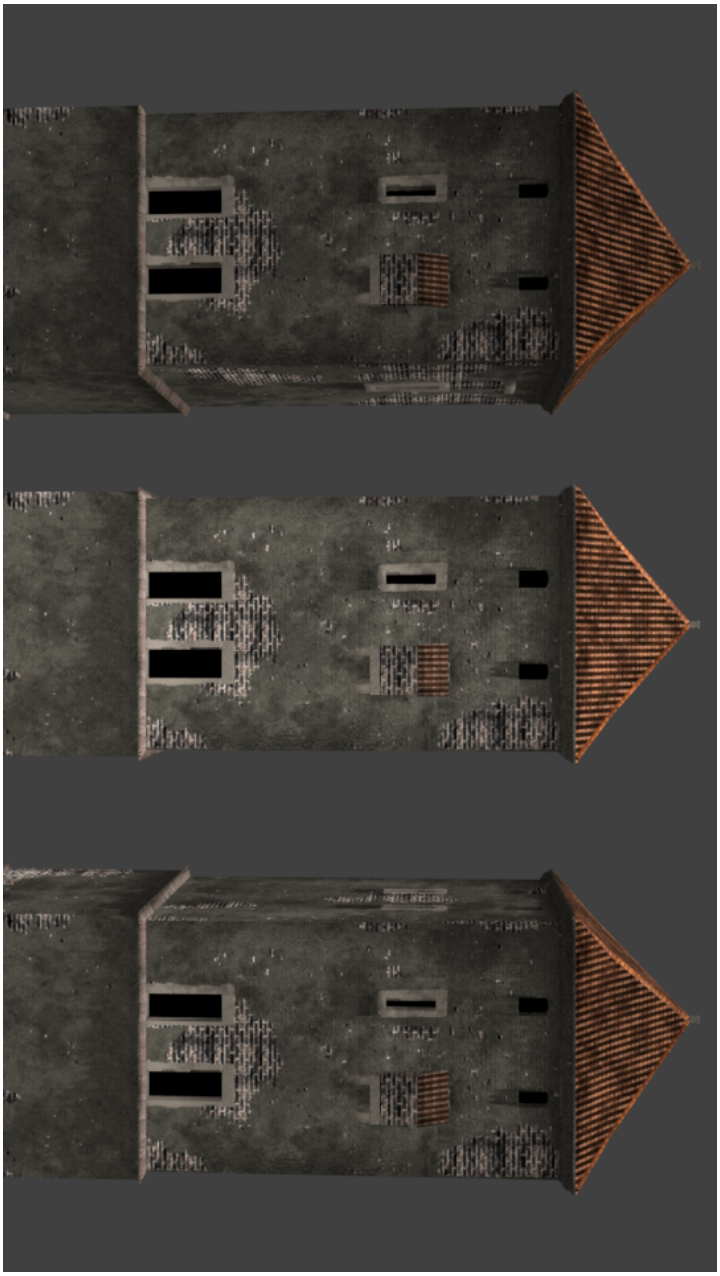


Obrázek 9.3: Ukázka různých hodnot pro intenzitu deště. Hodnoty jsou zleva 0,8, 0,92, 1,0 a 1,15

Obrázek 9.4: Porovnání různých hodnot pozadí při míchání vrstev. Hodnoty jsou zleva 0,0, 0,1, 0,2 a 0,4



Obrázek 9.5: Zasněžený model, hodnoty intensity zasněžení jsou zleva −1, 0 a 1



Obrázek 9.6: Srovnání osvětlení modelu při zataženém počasí, pro červen v sedm, dvanáct a šestnáct hodin

Závěr

Cílem této práce bylo vytvořit zásuvný modul do programu Blender, který by byl schopen generovat textury s vybranými vlivy počasí a průběhu dne. Sekundárním cílem bylo vytvoření zadávacího skriptu, s jehož pomocí je možné automatizovat celý generovací proces. Oba tyto cíle se ve výsledku podařilo splnit po realizační stránce, po vzhledové stránce výsledků je ještě možné vylepšení jako například přidání odlesků (spekulární mapy) nebo doplnění zajímavějších zpracování jednotlivých vlivů. V závěrečných fázích práce vyvstal další cíl, kterým bylo vytvoření uceleného textu pro studenty, kteří by chtěly navázat na tuto práci. Celá tato práce byla takto psána již od začátku bez většího záměru.

V této práci lze nalézt analýzu některých metod v počítačové grafice pro vytváření různých efektů počasí a průběhu dne. Následně proběhl návrh jednotlivých efektů za pomoci nástroje Blender a jeho vestavěných možností přesněji řečeno Cycles material editoru. Kromě jednotlivých efektů zde byl také navržen celý zásuvný modul a zadávací skript. Kromě návrhu zde byla obsažena také možná implementace užšího výběru efektů. Celý obsah byl zakončen testováním, které obsahuje shrnutí výstupů z uživatelského testování a ukázka různých výstupů výsledného modulu.

Výsledky zásuvného modulu jsou závislé na zadaných hodnotách a metodě výpočtu. Použité metody v této práci dávají zajímavé výsledky vzhledem k provedeným zjednodušením. Zajímavý výsledek podává samotné krokování dne v různých hodinách.

Výsledek této práce by mohl najít uplatnění při zjednodušení vytváření textur a její automatizaci. Také umožňuje naprosté odstínění uživatele od nutnosti porozumění dané problematice. Výstup zásuvného modulu umožňuje zrychlit zobrazovací čas v simulovaném prostředí. Vylepšení pro tento modul jsou různé například zlepšení výpočtů náležitostí jednotlivých dnů, nebo lepší vzhled efektů sněhu. Případně další která jsou zmíněna na konci sekce 9. Kromě výše zmíněných má tato práce také uplatnění jako výchozí bod pro tvorbu zlepšení nebo dodělání zkoumaných efektů.

Bibliografie

2. *EverydayMysteries* [online] [cit. 2018-10-27]. Dostupné z: http://www.loc.gov/rr/scitech/mysteries/colors.html.

3. ZAHBAZ. Why does sunlight cause colors to fade?(physics stack exchange). In: *PhysicsStackExchange* [online]. 2015 [cit. 2018-10-27]. Dostupné z: https://physics.stackexchange.com/questions/178549/why-does-sunlight-cause-colors-to-fade.

5. ZAHBAZ. Why does sunlight cause colors to fade?(chemistry stack exchange). In: *ChemistryStackExchange* [online]. 2015 [cit. 2018-10-27]. Dostupné z: https://chemistry.stackexchange.com/questions/29445/why-does-sunlight-cause-colors-to-fade.

6. BERNARD, James P. *Why Do Dyes Fade? A Look at Low Lightfast Dyes* [online]. 2016 [cit. 2018-10-27]. Dostupné z: http://www.fsw.cc/dyes-fade-look-low-lightfast-dyes/.

8. KIMMEL, Bradley W.; BARANOSKI, Gladimir V. G.; CHEN, T. F.; YIM, Daniel; MIRANDA, Erik. Spectral Appearance Changes Induced by Light Exposure. *ACM Trans. Graph.* 2013, roč. 32, č. 1, s. 10:1–10:13. ISSN 0730-0301. Dostupné z DOI: 10.1145/2421636.2421646.

9. GILES, C.; MCKAY, R. The lightfastness of dyes: A review. *Textile Res. J.* 1963, roč. 33, č. 7, s. 528–575.

10. GILES, C. The fading of colouring matters. *J. Appl. Chemi.* 1965, roč. 15, s. 541–550.

11. JOHNSTON-FELLER, R.; FELLER, R.; BAILIE, C.; CURRAN, M. The kinetics of fading: Opaque paint films pigmented with alizarin lake and titanium dioxide. *J. Amer. Inst. Conserv.* 1984, roč. 23, č. 2, s. 114–129.

12. SHI, Xifan; LU, Dongming; LIU, Jianming. Color Changing and Fading Simulation for Frescoes Based on Empirical Knowledge from Artists. In: *Proceedings of the 7th Pacific Rim Conference on Advances in Multimedia Information Processing*. Hangzhou, China: Springer-Verlag, 2006, s. 861–869. PCM'06. ISBN 978-3-540-48766-1. Dostupné z DOI: 10.1007/11922162_98.

13. SHI, Xifan; LU, Dongming. Colorimetric and Chemical Modeling Based Aging Simulation of Dunhuang Murals. In: *Proceedings of the The Fifth International Conference on Computer and Information Technology*. Washington, DC, USA: IEEE Computer Society, 2005, s. 570–574. CIT '05. ISBN 0-7695-2432-X. Dostupné z DOI: 10.1109/CIT.2005.85.

14. IMAI, F.; ROSEN, M.; BERNS, R. Multi-Spectral imaging of a Van Gogh's self-portrait at the National Gallery of Art, Washington, D.C. *In Proceedings of the IS&T PICS Conference. IS&T*. 2001, s. 185–189.

15. PRODUCTS, Harrick Scientific. *What is Kubleka-Munk?* [online] [cit. 2018-10-28]. Dostupné z: http://mmrc.caltech.edu/FTIR/Literature/Diff%5C%20Refectance/Kubelka-Munk.pdf.

16. WILKIE, Alexander; HOŠEK, Lukas. Predicting Sky Dome Appearance on Earth-like Extrasolar Worlds. In: *Proceedings of the 29th Spring Conference on Computer Graphics*. Smolenice, Slovakia: ACM, 2013, 145:145–145:152. SCCG '13. ISBN 978-1-4503-2480-9. Dostupné z DOI: 10.1145/2508244.2508263.

17. PEARCE, E. Photodegradation, photo-oxidation and photostabilization of polymers, B. Ranby and J. F. Rabek, Wiley-Interscience, New York, 1975, 573 pp. $47.50. *Journal of Polymer Science: Polymer Letters Edition*. 1975, roč. 13, č. 10, s. 621–622. Dostupné z DOI: 10.1002/pol.1975.130131008.

18. KUBELKA, Paul. New Contributions to the Optics of Intensely Light-Scattering Materials. Part I. *J. Opt. Soc. Am.* 1948, roč. 38, č. 5, s. 448–457. Dostupné z DOI: 10.1364/JOSA.38.000448.

19. PREETHAM, A. J.; SHIRLEY, Peter; SMITS, Brian. A Practical Analytic Model for Daylight. In: *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 1999, s. 91–100. SIGGRAPH '99. ISBN 0-201-48560-5. Dostupné z DOI: 10.1145/311535.311545.

20. HOSEK, L.; WILKIE, A. An Analytic Model for FullSpectral Sky-dome Radiance. *CM Trans. Graph.* 2012, roč. 31, s. 4.

23. STARK, Glenn. *Light physics* [online] [cit. 2018-12-10]. Dostupné z: https://www.britannica.com/science/light.

24. RAIMOND TUNNEL Jaanus Jaggo, Margus Luik. *Shading and Lighting* [online] [cit. 2018-12-12]. Dostupné z: https://cglearn.eu/pub/computer-graphics/shading-and-lighting%5C#.

25. RICHTR, Radek. *Programování grafických aplikací, osvětlovací modely* [online]. 2017 [cit. 2018-12-26]. Dostupné z: https://edux.fit.cvut.cz/courses/BI-PGA/_media/lectures/17pga_p7.pdf [Soubor přístupný po přihlášení do sítě ČVUT].

30. HAYWARD, Andrew. *What is ray tracing? Everything you need to know about the next big graphical leap.* [online] [cit. 2018-12-19]. Dostupné z: https://www.techradar.com/news/ray-tracing.

31. JENSEN, H. *Realistic Image Synthesis Using Photon Mapping.* New York: A K Peters/CRC Press, 2001. Dostupné také z: https://www.taylorfrancis.com/books/9781439863633.

32. WALTON, Jarred. *What is ray tracing, and how does Nvidia's GeForce RTX handle the technology?* [online] [cit. 2018-12-19]. Dostupné z: https://www.pcgamer.com/what-is-ray-tracing/.

34. SCRATCHAPIXEL 2.0. *Introduction to Acceleration Structures BVH* [online] [cit. 2018-12-19]. Dostupné z: https://www.scratchapixel.com/lessons/advanced-rendering/introduction-acceleration-structure/bounding-volume.

36. SMITH, Matthew S. *What is ray tracing, and how will it change games?* [online] [cit. 2018-12-19]. Dostupné z: https://www.digitaltrends.com/computing/what-is-ray-tracing/.

38. YU, Tin-Tin; LOWTHER, John; SHENE, Ching-Kuang. Photon Mapping Made Easy. In: *Proceedings of the 36th SIGCSE Technical Symposium on Computer Science Education.* St. Louis, Missouri, USA: ACM, 2005, s. 201–205. SIGCSE '05. ISBN 1-58113-997-7. Dostupné z DOI: 10.1145/1047344.1047418.

39. WATERS, Zack. *Photon Mapping* [online] [cit. 2018-12-17]. Dostupné z: https://web.cs.wpi.edu/~emmanuel/courses/cs563/write_ups/zackw/photon_mapping/PhotonMapping.html.

40. RORY. *Lighting: The Rendering Equation* [online]. 2008 [cit. 2018-12-17]. Dostupné z: http://www.rorydriscoll.com/2008/08/24/lighting-the-rendering-equation/.

41. METROPOLIS, Nicholas; ULAM, Stanislaw. The monte carlo method. *Journal of the American statistical association.* 1949, roč. 44, č. 247, s. 335–341.

42. BURGIN, Mark S.; ADES, Maurice J. Monte Carlo Methods and Super-recursive Algorithms. In: *Proceedings of the 2009 Spring Simulation Multiconference*. San Diego, California: Society for Computer Simulation International, 2009, 140:1–140:6. SpringSim '09. Dostupné také z: http://dl.acm.org/citation.cfm?id=1639809.1655368.

45. HOPE, Computer. *Radiosity* [online]. 2018 [cit. 2018-12-19]. Dostupné z: https://www.computerhope.com/jargon/r/radiosity.htm.

46. SKWIGLY. *Computer Concepts – What is Radiosity?* [online]. 2004 [cit. 2018-12-19]. Dostupné z: http://www.skwigly.co.uk/computer-concepts-what-is-radiosity/.

47. REDDY, J.N.; REDDY, J.N. *An Introduction to the Finite Element Method*. McGraw-Hill, 2006. McGraw-Hill series in mechanical engineering. ISBN 9780071267618. Dostupné také z: https://books.google.cz/books?id=bamCPwAACAAJ.

50. U/JOEBAF. *What lighting models are most popular in games?* [online] [cit. 2018-12-25]. Dostupné z: https://www.reddit.com/r/gamedev/comments/1clmlm/what_lighting_models_are_most_popular_in_games/.

52. BÁRTOVÁ, Kristina. *Odraz světla, BRDF* [online] [cit. 2018-12-25]. Dostupné z: https://cgg.mff.cuni.cz/~jaroslav/teaching/2012-NPGR010/notes/NPGR010-2012%20-%2003%20-%20poznamky.pdf.

53. PLURALSIGHT. *Understanding Subsurface Scattering - Capturing the Appearance of Translucent Materials* [online] [cit. 2018-12-26]. Dostupné z: https://www.pluralsight.com/blog/film-games/understanding-subsurface-scattering-capturing-appearance-translucent-materials.

54. HENRIK WANN JENSEN Stephen R. Marschner, Marc Levoy; HANRAHAN, Pat. *A Practical Model for Subsurface Light Transport* [Proceedings of SIGGRAPH']. 2001 [cit. 2018-12-26]. Dostupné z: http://jbit.net/~sparky/bssrdf.pdf.

55. CONTRIBUTORS, Wikipedia. *Bidirectional scattering distribution function* [online]. 2017 [cit. 2018-12-26]. Dostupné z: https://en.wikipedia.org/wiki/Bidirectional_scattering_distribution_function.

57. RICHTR, Radek. *Multimediální grafické aplikace, realistické metody texturování* [online]. 2017 [cit. 2018-12-26]. Dostupné z: https://edux.fit.cvut.cz/courses/BI-PGA/_media/lectures/17pga_p8.pdf [Soubor přístupný po přihlášení do sítě ČVUT].

58. CONTRIBUTORS, Wikipedia. *Bidirectional reflectance distribution function* [online]. 2017 [cit. 2018-12-26]. Dostupné z: https://en.wikipedia.org/wiki/Bidirectional_reflectance_distribution_function.

59. KNUTZEN, Johan. *Generating Climbing Plants Using L-Systems* [online] [cit. 2019-02-28]. Dostupné z: `http://www.cse.chalmers.se/~uffe/xjobb/climbingplants.pdf`.

62. BENEŠ, Bedrich; MILLÁN, Erik Uriel. Virtual Climbing Plants Competing for Space. In: *CA '02: Proceedings of the Computer Animation* [online]. Washington, DC, USA: IEEE Computer Society, 2002, s. 33 [cit. 2019-02-24]. ISBN 0-7695-1594-0. Dostupné z: `http://hpcg.purdue.edu/bbenes/papers/Benes02CA.pdf`.

64. CONTRIBUTORS, Wikipedia. *L-system* [online]. 2018 [cit. 2019-02-24]. Dostupné z: `https://en.wikipedia.org/wiki/L-system`.

67. JIRASEK, Catherine Alena. A biomechanical model of branch shape inplants expressed using l-systems. *Master's thesis, University of Calgary*. 2000. Dostupné také z: `https://prism.ucalgary.ca/bitstream/handle/1880/39999/55218Jirasek.pdf?sequence=1&isAllowed=y`.

68. MERIEM FOURNIER Henri Bailleres, Bernard Chanson. Tree biomechanics: growth, cumulative prestresses, and reorientations. *Biomimetics* [online]. 1994, roč. 2, s. 229–251 [cit. 2019-02-24].

69. LUFT, Thomas. *An Ivy generator* [online] [cit. 2019-02-24]. Dostupné z: `http://graphics.uni-konstanz.de/~luft/ivy_generator/`.

71. REFFYE, P. de; EDELIN, C.; FRACON, J.; JAEGER, M.; PUECH, C. Plants Models Faithful to Botanical Structure and Development. In: *InProceedings of SIGGRAPH'88 of Annual Conference Series* [online]. 1988, s. 151–158 [cit. 2018-02-24].

72. BENEŠ, B. *Visual Simulation of Plant Developmentwith Respect to Influence of Light* [online]. Springer Computer Science, pages 125-136. Springer-Verlag Wien NewYork: InComputer Animation a Simulation'97, 1997 [cit. 2018-02-24].

73. LECOUSTRE, R.; REFFYE, P. de; JAEGER, M.; DI-NOUARD, P. Controlling the Architectural Geometry of Plant's Growth - Application to the Begonia Genus. *InComputer Animation'92* [online]. 1992, s. 199–214 [cit. 2018-02-24].

74. CONTRIBUTORS, Wikipedia. *Vzrostný vrchol* [online]. 2018 [cit. 2019-02-24]. Dostupné z: `https://en.wikipedia.org/wiki/Vzrostn%C3%BD_vrchol`.

75. DATA VISUALIZATION, Inc. nteractive. *SpeedTree* [software]. 2017 [cit. 2019-02-24]. Dostupné z: `https://store.speedtree.com/`.

76. CONTRIBUTORS, Wikipedia. *SpeedTree* [online]. 2018 [cit. 2019-02-24]. Dostupné z: `https://en.wikipedia.org/wiki/SpeedTree`.

79. BLENDERGURU. *How to Create Realistic Rain* [online]. 2012 [cit. 2019-01-28]. Dostupné z: https://www.blenderguru.com/tutorials/how-to-create-realistic-rain.

80. PANDAY, Sorab; HUYAKORN, Peter S. A fully coupled physically-based spatially-distributed model for evaluating surface/subsurface flow. *Advances in Water Resources* [online]. 2004, roč. 27, č. 4, s. 361–382 [cit. 2019-01-28]. ISSN 0309-1708. Dostupné z DOI: https://doi.org/10.1016/j.advwatres.2004.02.016. A Tribute to George F. Pinder.

81. TOKAREV, Kirill. *Real-Time Rain Material Within Unreal Engine* [online] [cit. 2019-01-28]. Dostupné z: https://80.lv/articles/setting-up-rain-materials-in-substance-designer/.

82. INATOR. *Is it any Weather/water/snow and wind system in UE4?* [online]. 2014 [cit. 2019-01-28]. Dostupné z: https://forums.unrealengine.com/development-discussion/content-creation/12819-is-it-any-weather-water-snow-and-wind-system-in-ue4?27320-Is-it-any-Weather-water-snow-and-wind-system-in-UE4=&highlight=weather.

83. U/22VORTEX22. *Achieving Photorealistic and Realistic behavior rain systems* [online]. 2018 [cit. 2019-01-28]. Dostupné z: https://www.reddit.com/r/unrealengine/comments/7okaln/achieving_photorealistic_and_realistic_behavior/.

85. NOBODY. Unreal Engine 4 Tutorial [German/Deutsch] - Regen und was dazugehört - #004. In: *YouTube* [online]. 2014 [cit. 2019-01-28]. Dostupné z: https://www.youtube.com/watch?v=D_6VKS2nNMU.

87. CONTRIBUTORS, Wikipedia. *Material point method* [online]. 2018 [cit. 2019-02-24]. Dostupné z: https://en.wikipedia.org/wiki/Material_point_method.

88. *The Snow Graphics in 'Frozen' Can Predict the Mechanics of Real Avalanches* [online] [cit. 2019-02-24]. Dostupné z: https://medium.com/penn-engineering/the-snow-graphics-in-frozen-can-predict-the-mechanics-of-real-avalanches-57a453b3752c.

89. NISHITA, Tomoyuki; IWASAKI, Hiroshi; DOBASHI, Yoshinori; NAKAMAE, Eihachiro. A Modeling and Rendering Method for Snow by Using Metaballs. *Computer Graphics Forum*. 1997, roč. 16, č. 3, s. C357–C364. Dostupné z DOI: 10.1111/1467-8659.00173.

90. FEARING, Paul. Computer Modelling of Fallen Snow. In: *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques* [online]. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 2000, s. 37–46 [cit. 2019-02-24]. SIGGRAPH '00. ISBN 1-58113-208-5. Dostupné z DOI: 10.1145/344779.344809.

92. NAIMA. *Assassin creed III deep snow tech ...* [online]. 2012 [cit. 2019-02-24]. Dostupné z: https://polycount.com/discussion/108145/assassin-creed-iii-deep-snow-tech.

94. GIMP ORG. *Layer Modes* [online] [cit. 2019-07-24]. Dostupné z: https://docs.gimp.org/en/gimp-concepts-layer-modes.html.

105. ANONYM459623(OTÁZKA). *Slunce nad obzorem* [online] [cit. 2019-09-19]. Dostupné z: https://www.ontola.com/cs/ondi/seozuy/vyska-slunce-nad-obzorem-vypocet.

Seznam použitých obrázků

1. *Light and photosynthetic pigments* [online obrázek] [cit. 2019-11-17]. Dostupné z: https://www.khanacademy.org/science/biology/photosynthesis-in-plants/the-light-dependent-reactions-of-photosynthesis/a/light-and-photosynthetic-pigments.

4. *Light Matter Interaction* [online obrázek] [cit. 2019-11-17]. Dostupné z: https://www.renishaw.fr/fr/a-basic-overview-of-raman-spectroscopy--25805.

7. *Fade look dyes* [online obrázek] [cit. 2019-11-17]. Dostupné z: http://www.fsw.cc/dyes-fade-look-low-lightfast-dyes/.

21. *Hertz-Sprung diagram* [online obrázek] [cit. 2019-11-16]. Dostupné z: https://www.atnf.csiro.au/outreach/education/senior/cosmicengine/stars_hrdiagram.html.

22. *Star colors* [online obrázek] [cit. 2019-11-16]. Dostupné z: https://docs.kde.org/trunk5/it/extragear-edu/kstars/ai-colorandtemp.html.

26. *Phong components* [online obrázek] [cit. 2019-11-17]. Dostupné z: https://en.wikipedia.org/wiki/Phong_shading%5C#/media/File:Phong_components_version_4.png.

27. *Shading comparation* [online obrázek] [cit. 2019-11-17]. Dostupné z: https://www.giantbomb.com/gouraud-shading/3015-4864/images/.

28. *Blinn-Phong comparation* [online obrázek] [cit. 2019-11-17]. Dostupné z: https://en.wikipedia.org/wiki/Blinn%5C%E2%5C%80%5C%93Phong_reflection_model%5C#/media/File:Blinn_phong_comparison.png.

29. *Maya comparation global/local illumination* [online obrázek] [cit. 2019-11-17]. Dostupné z: https://knowledge.autodesk.com/support/maya-lt/learn-explore/caas/CloudHelp/cloudhelp/2015/ENU/MayaLT/files/BoL-Indirect-global-vs-direct-illumination-htm.html.

33. *RTX Bright memory* [online obrázek] [cit. 2019-11-17]. Dostupné z: https://wccftech.com/bright-memory-the-action-game-made-by-a-single-developer-is-getting-nvidia-rtx-ray-tracing-soon/.

35. *BVH example* [online obrázek] [cit. 2019-11-17]. Dostupné z: https://www.scratchapixel.com/lessons/advanced-rendering/introduction-acceleration-structure/bounding-volume.

37. *RTX Nvidia example* [online obrázek] [cit. 2019-11-17]. Dostupné z: https://www.ign.com/articles/2019/10/10/what-is-ray-tracing-and-should-you-care.

43. *PhotonMap Caustic* [online obrázek] [cit. 2019-11-17]. Dostupné z: https://mattisb0rgen.wordpress.com/2009/05/23/photon-mapping-and-caustics/.

44. *PhotonMap Caustic - glass* [online obrázek] [cit. 2019-11-17]. Dostupné z: https://en.wikipedia.org/wiki/Photon_mapping%5C#/media/File:Glas-1000-enery.jpg.

48. *Radiosity interior* [online obrázek] [cit. 2019-11-17]. Dostupné z: https://en.wikipedia.org/wiki/Radiosity_(computer_graphics)%5C#/media/File:Radiosity_Comparison.jpg.

49. *Radiosity interior another example* [online obrázek] [cit. 2019-11-17]. Dostupné z: http://www.formz.com/manuals/renderzonerendering/!SSL!/WebHelp/8_1_radiosity_solution_using_sunlight.html.

51. HAINDL, Michal; FILIP, Jiří. *Visual Texture: Accurate Material Appearance Measurement, Representation and Modeling*. 2013. ISBN 9781447149019. Dostupné z DOI: 10.1007/978-1-4471-4902-6.

56. *Comparation of BRDF and BTDF* [online obrázek] [cit. 2019-11-17]. Dostupné z: https://en.wikipedia.org/wiki/Bidirectional_scattering_distribution_function#/media/File:BSDF05_800.png.

60. *Reallife overgrown building* [online obrázek] [cit. 2019-11-16]. Dostupné z: https://imgur.com/gallery/SXFojWm.

61. *Reallife overgrown building* [online obrázek] [cit. 2019-11-16]. Dostupné z: https://imgur.com/gallery/o4PTnPV.

63. *L-system output* [online obrázek] [cit. 2019-11-16]. Dostupné z: https://en.wikipedia.org/wiki/File:Dragon_trees.jpg.

65. *L-system output* [online obrázek] [cit. 2019-11-16]. Dostupné z: https://en.wikipedia.org/wiki/File:Fractal_weeds.jpg.

66. *Fractal plant* [online obrázek] [cit. 2019-11-16]. Dostupné z: https://en.wikipedia.org/wiki/File:Fractal-plant.svg.

70. *Ivy generator output* [online obrázek] [cit. 2019-11-16]. Dostupné z: http://graphics.uni-konstanz.de/~luft/ivy_generator/images/example_3.jpg.

77. *SpeedTree* [online obrázek] [cit. 2019-11-16]. Dostupné z: https://store.speedtree.com/blog/.

78. *Batman Arkham Knight rain detail* [online obrázek] [cit. 2019-11-17]. Dostupné z: https://i.ytimg.com/vi/Vxq53NmfVxM/maxresdefault.jpg.

84. *Watch Dogs rain detail* [online obrázek] [cit. 2019-11-17]. Dostupné z: https://i.ytimg.com/vi/W9yJ-_3Kmnw/maxresdefault.jpg.

86. *Snow simulation screenshot* [online obrázek] [cit. 2019-11-17]. Dostupné z: https://medium.com/penn-engineering/the-snow-graphics-in-frozen-can-predict-the-mechanics-of-real-avalanches-57a453b3752c.

91. *God of War screenshot* [online obrázek] [cit. 2019-11-17]. Dostupné z: https://www.zing.cz/novinky/96034942/v-god-of-war-ceka-na-hrace-tajemstvi-ktere-jeste-nikdo-neobjevil/.

93. *Frostpunk screenshot* [online obrázek] [cit. 2019-11-17]. Dostupné z: https://www.kotaku.com.au/2017/10/frostpunkis-a-city-builder-where-you-can-eat-the-dead/.

95. *Blender example Brick* [online obrázek] [cit. 2019-11-21]. Dostupné z: https://docs.blender.org/manual/en/latest/render/shader_nodes/textures/brick.html.

96. *Blender example Checker* [online obrázek] [cit. 2019-11-21]. Dostupné z: https://docs.blender.org/manual/en/latest/render/shader_nodes/textures/checker.html.

97. *Blender example magic* [online obrázek] [cit. 2019-11-21]. Dostupné z: https://docs.blender.org/manual/en/latest/render/shader_nodes/textures/magic.html.

98. *Blender example musgrave* [online obrázek] [cit. 2019-11-21]. Dostupné z: https://docs.blender.org/manual/en/latest/render/shader_nodes/textures/musgrave.html.

99. *Blender example voronoi* [online obrázek] [cit. 2019-11-21]. Dostupné z: https://docs.blender.org/manual/en/latest/render/shader_nodes/textures/voronoi.html.

100. *Blender example noise* [online obrázek] [cit. 2019-11-21]. Dostupné z: https://docs.blender.org/manual/en/latest/render/shader_nodes/textures/noise.html.

101. *Blender example wave* [online obrázek] [cit. 2019-11-21]. Dostupné z: https://docs.blender.org/manual/en/latest/render/shader_nodes/textures/wave.html.

102. *Blender example musgrave* [online obrázek] [cit. 2019-11-21]. Dostupné z: https://docs.blender.org/manual/en/latest/_images/render_shader-nodes_textures_musgrave_example-type-ridged.jpg.

103. *Blender example musgrave* [online obrázek] [cit. 2019-11-21]. Dostupné z: https://docs.blender.org/manual/en/latest/_images/render_shader-nodes_textures_musgrave_example-type-fbm.jpg.

104. *Blender example musgrave* [online obrázek] [cit. 2019-11-21]. Dostupné z: https://docs.blender.org/manual/en/latest/_images/render_shader-nodes_textures_musgrave_example-type-hybrid.jpg.

Příloha **A**

Seznam použitých zkratek

AMAP Atelier de Modélistation de l'Architecture des Plantes, dílna pro modelování architektury rostlin

BRDF Bidirectional Reflectance Distribution Function, Obousměrná distribuční funkce odrazu

BTF Bidirectional Texture Function, Obousměrná texturovací funkce

BSSDF Bidirectional Scattering-Surface Distribution Function, Obousměrná distribuční funkce povrchového rozptylu

BSDF Bidirectional Scattering Distribution Function, Obousměrná distribuční funkce rozptylu

CG Computer Graphics

FEM Finit element method, metoda konečných prvků

GUI Graphical user interface

LOD Level Of Detail

UV Ultraviolet light, ultrafialové světlo

Příloha **B**

Obsah přiloženého disku

- `readme.txt` ..... stručný popis obsahu disku.
- `src`
  - `impl` ..... zdrojové kódy implementace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `tislijan.pdf` ..... text práce ve formátu PDF
  - `user_manual.pdf` ..... uživatelská příručka k zásuvnému modulu
  - `testing.pdf` ..... scénáře uživatelského testování