PG1 - Semestrální práce

Tislický Jan

Květen 2022

1 Úvod

Cílem semestrální práce pro předmět NI-PG1 bylo vytvoření Ray Tracer nebo Path Traceru. Z těchto dvou byla nakonec vybrána první možnost. Kromě rendereru byla také stanovena podmínka výběru akcelerační struktury, která urychlí samotný výpočet. Mimo to bylo možné libovolně rozšířit tento základní koncept o jakýkoliv další prvek. V této práci bylo jediné přidané rozšíření Fresnelova rovnice. Tu je možné si zjednodušeně představit, jako poměr pro smíchání výsledků odrazu a lomu. Pro srovnání využití tohoto aspektu a bez něj, je možné se podívat na poslední dva obrázky 6. Drtivá většina implementace byla provedena dle připraveného textu předmětu. Pouze některé drobnosti, jako např. Fresnel nebo některé části akcelerační struktury, byly získány z různých webů nebo elektronických knih či článků.

2 Struktura projektu

Celý projekt na gitu je rozdělen do snad trochu logického uspořádání. Hlavním souborem, jak se dá čekat, je main.cpp. Hlavním výtvarníkem v tomto projektu je pak raytracer.h (.cpp), který obsahuje veškeré potřebné metody pro syntézu obrazů. Tedy sledování paprsků, volání výpočtu průsečíků, vytváření/ukládání výsledného obrázku, přidávání různých jednotek jako jsou objekty, pohledy (kamery), načítání zadaného souboru, začátek stavby akcelerační struktury a výpočet osvětlení)

Poslední zmíněný soubor pak využívá celou řadu dalších zdrojových souborů k dosažení syntézy obrazu. Jako primitiva je možné nalézt trojúhelníky nebo sféry, které mají své metody na výpočet průsečíků, normál, nastavení různých konstant nebo povrchů. triangle.h (.cpp) neobsahuje oproti sféře výpočet průniku s paprskem. V prvních několika verzích tomu tak bylo, nicméně později kvůli komplikaci závislosti jednotlivých zdrojových souborů na sobě byl tento výpočet přesunut k paprsku.

Soubor camera.h (.cpp) obsahuje vše potřebné pro správné nastavení pohledu. Kamera samotná musí být také schopná správně vygenerovat paprsky na základě zadaných souřadnic a tuto funkcionalitu také má. Mimo to je možné také získat informativní výpisy k dané kameře.

Světelné zdroje je možné nalézt v souboru light.h (.cpp). Zde je možné nalézt očekávané položky, navíc je tu ukazatel na trojúhelník, ke kterému dané světlo patří a počet vzorků, které budou vzorkovány na daném plošném zdroji světla. S plošným zdrojem světla pak také souvisí vzorkování na daném trojúhelníku nebo výpočet vah jednotlivých světel.

Se světelnými zdroji také souvisí světelné paprsky, ty jsou k nalezení v souboru ray.h (.cpp). Kromě základních komponent obsahuje paprsek i souřadnice zásahu nějakého z trojúhelníku, který si zároveň drží opět pomocí ukazatele. Dále si také pamatuje zda je uvnitř nebo vně nějakého objektu. Jako metody pak má paprsek odraz nebo lom paprsku a průsečíky s trojúhelníkem nebo obálkou.

Samotné obálky je možné nalézt v souboru BoundingBox.h (.cpp). Obálky jsou jednoduše definovány pomocí maximálního bodu a minimálního bodu a s nimi je zde zanesen i střed. Tyto části jsou dopočítávány a měněny na základě toho, s čím se pracuje. Z metod je možné zde nalézt rozšíření obálky s bodem nebo jinou obálkou případně získání nejdelší osy dané obálky. Tato část se může hodit, pokud po-

dle originálně zadané osy není snadné získat vhodné rozdělení.

Obálky se pak využívají zejména v akcelerační struktuře. Pro tento projekt bylo vybráno základní BVH s rozdělením bodů podle středu. BVH je stromová struktura, která ke svému fungování potřebuje další struktury. Těmi jsou jednotlivé uzly, ty obsahují hlavně obálku, osu dělení a pak ukazatel na pravý nebo levý uzel případně trojúhelník. BVH pak obsahuje ukazatel na kořen a seznam listů. Nad těmi je zavolána výstavba dané struktury. Tato struktura také obsahuje metodu na průchod celého stromu a nalezení průsečíku.

Jedním z posledních souborů v projektu, které zatím nebyly zmíněny je vec3.h (.cpp). Ten obsahuje definici trojrozměrného vektoru, který je využit na mnoha místech v tomto projektu. Tato struktura byl poskytnuta v kostře projektu a byla mírně poupravena, dle potřeb.

Hlavičkový soubor conf.h pak obsahuje několik málo konstant, které je možné zakomentovat a získat tak trochu jiné chování (jedno vlákno vs více vláken). Využití akcelerační struktury nebo velikosti výsledného obrazu.

Jako doporučený nahrávač obj souborů byl použit tiny_obj_loader.h (.cc). Ten umožňuje snadno a rychle získat potřebné informace o zadaném obj souboru a tím značně ulehčit další postup práce.

3 Implementace

Během implementace nenastaly žádné závažnější problémy než špatná interpretace zdrojového textu. Případně doplňkových zdrojů. Jediný trochu závažnější problém byl interpolace normál v trojúhelníku. První část tohoto problému plynula ze špatného uvědomění jak je možné interpolovat normálu s pomocí barycentrických souřadnic a druhá část ze špatného uložení normál pro jednotlivé trojúhelníky.

Frustrující problémem byla implementace datové struktury, u které byl největší problém implementace průsečíku obálky a paprsku. Z počátku se průsečíky počítaly jen někde a nakonec po změně výpočtu průsečíku byl tento problém odstraněn.

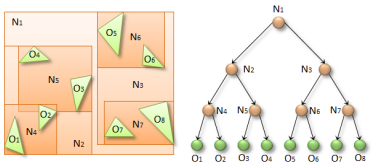

Figure 1: BVH ilustrativní obrázek.

4 Akcelerační struktura

Jak již bylo zmíněno jako akcelerační struktura bylo vybráno BVH. Tato datová struktura je založená na osově zarovnaných obálkách primitiv. Ta jsou pak poskládána do větších celků, které také získávají obálku, ty jsou velikostně závislá na obálkách jednotlivých primitiv.Celá tato struktura je pak reprezentována jako binární strom. Pokud se podíváme na scénu, pak můžeme říct, že je obalena jednou velkou obálkou. Ta se pak rozdělí dle různých strategií na dvě menší ve vybrané ose, opět je možné mít více přístupů. Následně se opět dělí na menší a menší obálky až se dojde k primitivům (trojúhelníkům).

Dělící osa se střídá pomocí metody Round Robin, až na případ kdy je vše vlevo nebo vpravo, v takovém případě se vezme nejdelší osa a proběhne pokus o vhodné rozdělení. Rozdělení pak probíhá pomocí jednoduchého určení středového bodu a rozdělení na levou a pravou část, případně opravy rozdělení.

V tomto projektu byla zvolena rekurzivní výstavba stromu BVH. Ve dvou nejjednodušších případech se rovnou vrátí (jedno primitivum) nebo se vytvoří jeden vnitřní uzel a na něj se naváží dva listy. Tento vnitřní uzel se pak vrátí. V případě, že je primitiv více než dvě, se aplikuje rozdělení dle vybrané osy na levou a pravou část. Toto rozdělení proběhne při jednom průchodu daného rozsahu primitiv a jejich prohazování. V případě, že jsou všechna primitiva na jedné nebo druhé straně, tak je nutné udělat úpravu rozdělení. V takovém případě, dojde k pokusu jakkoliv rozdělit primitiva a to dle nejdelší osy jejich obálky. Toto rozdělení se pak zanese do stromu a rekurzivně se pokračuje dokud není dělení ukončeno.

Traverzace této struktury je opět rekurzivní. Nejdříve se paprsek otestuje vůči dané obálce, pak se

zjistí zda má cenu jít dále vzhledem k již nalezenému primitivu. Pokud má cenu jít dále pak se zjistí zda se jedná o primitivum. pokud ano tak se pouze provede test na průnik paprsku s daným primitivem. Jinak se sestoupí do nižších vrstev struktury.

5 Výsledky Měření

Tato sekce obsahuje naměřené výsledky. V tabulce 1 jsou uveden naměřené hodnoty pro tři scény s a bez akcelerační strukturou.

6 Výstupy

Tato část obsahuje pouze výstupy pro jednotlivé části projektu. Výstupy jsou brány tak jak šly postupně s tvorbou tohoto projektu.

Scéna	Stavba [μ s]	Výpočet [s]	# paprsků	# průsečíků	# trav. uzlů	Prům. průsečíků / paprsek	Prům. traverzace / paprsek	doba traverzace [μ s]
CornellBox-Original	~ 8.005	1.126	36808012	355247675	1379861618	~9.651	~37.488	~5.844
CornellBox-Original	X	~0.809	36808012	1251472408	X	~34.000	X	X
CornellBox-Sphere	~615.658	~7.853	43378131	1222786556	12542558635	~28.189	~289.144	~47.965
CornellBox-Sphere	X	~45.394	43378131	94911350628	X	~2188.000	X	X
CornellBox-Sphere-f	~612.193	~10.834	45762422	1266743348	13109732892	~27.680863	~286.473	~46.448
CornellBox-Sphere-f	X	~44.391	45762422	100128179336	X	~2188.000	X	X

Table 1: Tabulka s naměřenými hodnotami raytraceru s a bez akcelerační struktury. Měření byla provedena pouze s 32 vzorky na světlo a s multithread nastavením na procesoru Ryzen 9 3950x, 16 jader, 32 vláken, 3500MHz až 4700MHz

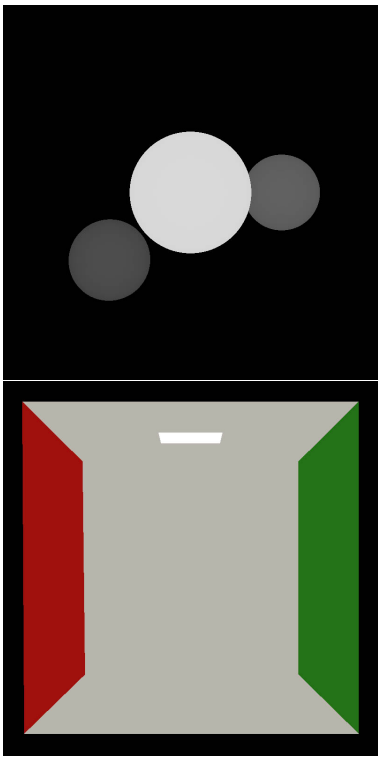

Figure 2: První tři kroky implementace. Hloubkový test, přiřazení barev.

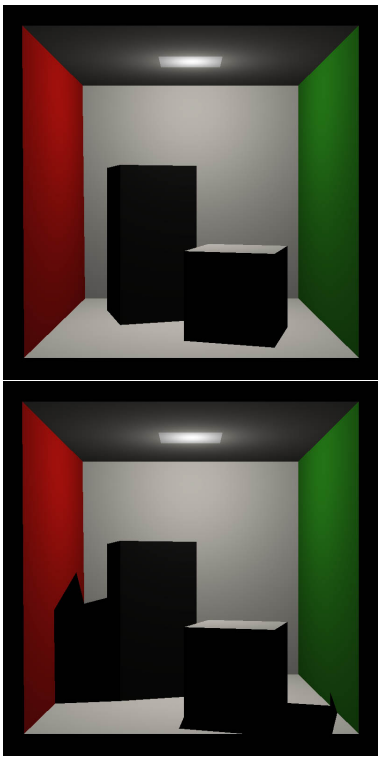

Figure 3: Výsledek po přidání světel a s výpočtem barvy a následně se stíny.

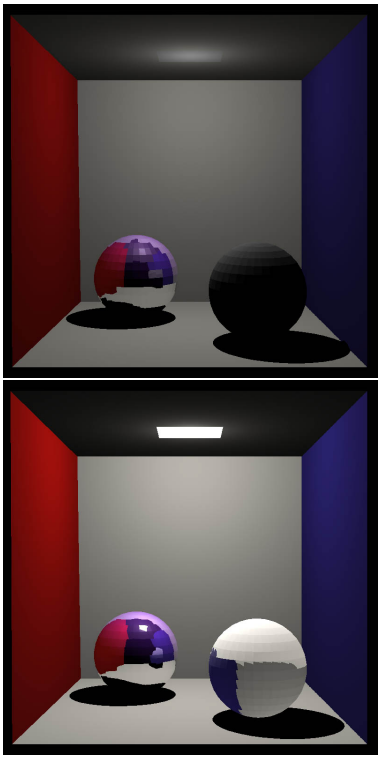

Figure 4: Postup práce s odraženými a lomenými paprsky. Zde je vidět několik vad, kde nejzávažnější byla interpolace normál a správné nastavení normál při načítání.

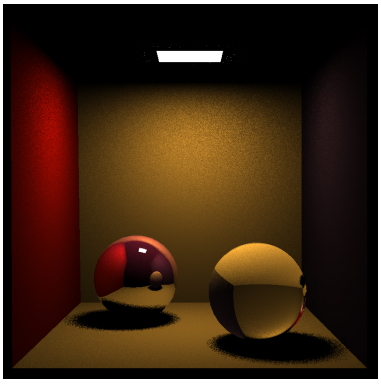

Figure 5: Výsledný zrnitý obraz po opravách a s plošným světlem.

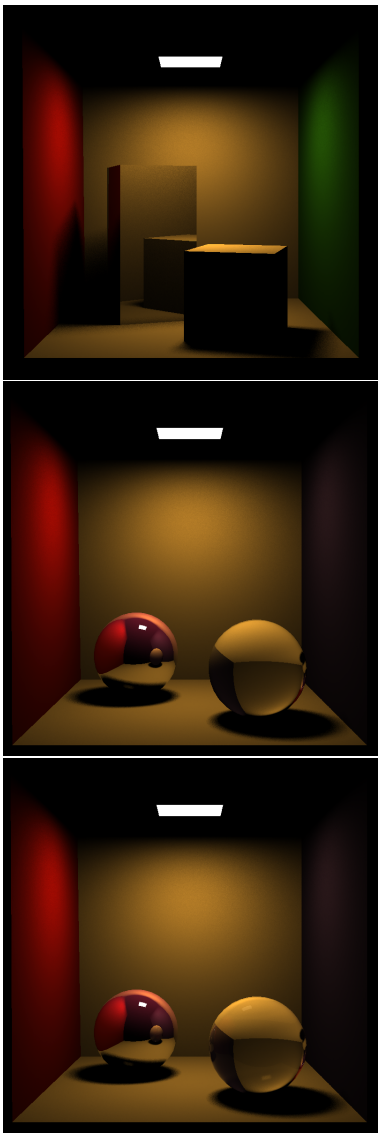

Figure 6: Závěrečné rendery s 32 vzorky na světlo. Poslední dva obrázky se liší v použité scéně. Předposlední nemá přidaný efekt Fresnel a poslední má. To může být vidět jako drobné odrazy na povrchu pravé koule.