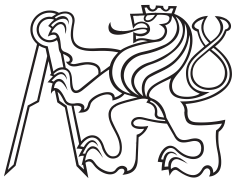
Bakalářská práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická
Katedra počítačové grafiky a interakce

Sledování paprsku v reálném čase v Unity

Bohumil Bednář

Vedoucí: doc. Ing. Jiří Bittner, Ph.D.
Obor: Počítačové hry a grafika
Studijní program: Otevřená informatika
Květen 2021

Poděkování

Děkuji docentu Bittnerovi, za poskytování konzultací po dobu tvorby této bakalářské práce.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně, a že jsem uvedl veškerou použitou literaturu.

V Praze, 21. května 2021

Abstrakt

Práce se zabývá zmapováním možností sledování paprsku v reálném čase v Unity a chováním této funkce v herním nasazení.

Klíčová slova: Počítačová grafika, Sledování parsku, Unity

Vedoucí: doc. Ing. Jiří Bittner, Ph.D.

Abstract

Thesis explores possibilites of real-time ray tracing in Unity and its application in computer games.

Keywords: Computer Graphics, Ray Tracing, Unity

Title translation: Real-Time Ray Tracing in Unity

Obsah

Obrázky

Tabulky

ctuthesis t1606152353

Kapitola 1

Úvod

Sada funkcí sledování paprsku v reálném čase neboli real-time ray tracingu byla přidána do Unity v roce 2019 ve verzi 2019.3. Jde o stále relativně novou sadu funkcí. Tato sada funkcí poskytuje dva nové způsoby vykreslování alternativní k běžně používané rasterizaci, konkrétně jde o ray tracing a path tracing. Dále také tato sada poskytuje množství grafických efektů počítaných formou ray tracingu. Tato práce se zabývá otestováním této sady funkcí z hlediska grafické kvality a výkonu. Testy jsou prováděny jak na scénách snažících se přiblížit realitě tak na jednoduchý scénách sloužících k otestování konkrétní funkcionality.

1.1 Terminologie

Termín sledování paprsku v reálném čase je v práci používán i v anglické verzi real-time ray tracing, případně je použita zkratka RTRT. Názvy položek v menu, nastavení, a dalších prvků Unity editoru jsou ponechány v anglickém jazyce, taktéž jsou v případě některých odborných termínů použity anglické názvy.

1.2 Struktura práce

V kapitole Základy vykreslování jsou prezentovány základní informace, ze kterých vycházejí vykreslovací metody testované v této práci. Druhá kapitola Metody vykreslování obsahuje informace vysvětlující způsob fungovaní využitých metod vykreslovaní a vytváření grafických efektů. Třetí kapitola Prostředí HDRP a rendering v Unity obsahuje informace o fungování použitém rendering pipeline v Unity a jeho prvcích a parametrech týkajících se tématu práce. Čtvrtá kapitola Testovací aplikace a prostředí obsahuje popis aplikace vytvořené pro testování, základní informace o testování a informace o použitých sestavách hardwaru. Pátá kapitola Testování obsahuje jednotlivé testy, jejich výsledky a zhodnocení.

Kapitola 2

Základy vykreslování

2.1 Rendering

Rendering neboli vykreslování je proces vytváření 2d obrazu ze scény. Tento proces je prováděn algoritmem, případně sadou algoritmů, přičemž tyto algoritmy jsou částí nebo celým řešením takzvané vykreslovací rovnice(rendering equation), která popisuje pohyb světla po scéně:

$$L_0(x, \omega_0, \lambda, t) = L_e(x, \omega_0, \lambda, t) + \int_\Omega f_r(x, \omega_i, \omega_0, \lambda, t) L_i(x, \omega_i, \lambda, t)(\omega_i n) d\omega_i$$

kde:
$L_0(x, \omega_0, \lambda, t)$ je spektrální zář vlnové délky světla λ odchozího směrem ω_0 v čase t z bodu x
x je poloha v prostoru
ω_0 je směr odchozího světla
λ je konkrétní vlnová délka světla
t je čas
$L_e(x, \omega_0, \lambda, t)$je emitovaná spektrální zář
Ω je jednotková hemisféra se středem n obsahující všechny možné ω_i
$f_r(x, \omega_i, \omega_0, \lambda, t)$ je obousměrná distibuční funkce odrazu světla vlnové délky λ ze směru ω_i do směru ω_0 v čase t v konkrétním bodě x
ω_i je opak směru příchozího světla
$L_i(x, \omega_i, \lambda, t)(\omega_i n)$ je spektrální zář vlnové délky světla λ odražená ze směru ω_i v čase t do bodu x

n je normála plochy v bodě x
$\omega_i n$ je oslabující faktor odchozí intenzity záření
[Kaj86]

Existuje mnoho způsobů jak tohoto dosáhnou tato práce s však zabývá pouze čtyřmi z nich a to konkrétně rasterizací, ray castingem, ray tracingem a path tracingem. Důvodem tohoto je, že Unity používá za standardní situace jako vykreslovací metodu rasterizaci a zároveň ray tracing a path tracing jsou součástí nové sady funkcí, přičemž ray casting je jejich ideový předchůdce a některé principy fungování zůstávají podobné. Rendering lze rozdělit na dvě kategorie real-time rendering a non real-time rendering. Non real-time rendering se využívá pro vykreslování neinteraktivních scén jako jsou například filmy, videa nebo cutscény a tudíž je daleko méně omezen dobou vykreslování, tudíž může využívat výpočetně náročné metody bez nutnosti zjednodušování. Ray tracing a path tracing se původně používali tímto způsobem, z důvodu nedostupnosti dostatečně výkonného hardwaru. Real-time rendering je využíván pro vykreslování interaktivních scén například v počítačových hrách. Cílem real-time renderingu je vykreslit scénu v co nejlepší kvalitě za přijatelný čas. Ideální je, aby byla doba vykreslování taková, že FPS budou blízká obnovovací frekvenci monitoru, zároveň by neměla v nejhorším FPS klesnout po hranici 20, aby se udržela alespoň částečná iluze plynulosti animací a pohybu. Z tohoto důvodu se pro real-time rendering využívají buď metody s relativně nízkou výpočetní náročností jako je třeba rasterizace nebo se využívají různým způsobem výpočetně zjednodušené náročnější metody. Snížení výpočetní náročnosti složitější metod lze provést několika způsoby. Používaná řešení jsou například použití náhodného vzorkování a následná interpolace části výsledků, omezení maximálního počtu odrazů. Z pohledu implementace algoritmu je velmi zásadní pro jakou platformu je algoritmus implementován. Standardně používané platformy jsou CPU a GPU případně specializovaný hardware. Tato práce se zabývá testováním funkčnosti Unity real-time ray tracingu a rasterizace, které jsou implementovány pro běh primárně na GPU, přičemž některé fáze přípravného charakteru provádí CPU.

2.2 Rendering pipeline

Pipeline je sada po sobě jdoucích kroků sériově si předávajících své výstupní data, která jsou následně použita jako vstupní data následujícího kroku nebo finální výsledek. Rendering pipeline nebo také grafická pipeline je konceptuální model, který popisuje proces vykreslování 3d scény na 2d obrazovku. Dělí se na 3 části: Aplikační část obsahující aktualizaci scény, detekci kolizí. Geometrickou část obsahující proces konverze dat scény(objektů, světel, jejich

souřadnic vzhledem ke kameře,..) do formátu využívaného následujícím krokem. Ve vykreslovací části je na data scény aplikována vykreslovací metoda a jí používaný shading, osvětlovací metoda a grafické efekty.

Tento konceptuální model je pak realizován ve formě sady algoritmů, které provádějí tyto kroky a řídí přechody mezi nimi. Každý rendering pipeline má také nadefinovaný vlastní způsob jakým přijímá vstupní data(objekty a jejich materiály a textury, světla, skybox)a také má specifickou sadu shaderů, kterou používá. Implementace jednotlivých kroků muže být na základě mnoha různých faktorů včetně použité vykreslovací metody mezi jednotlivými rendering pipeline zcela odlišná, takže není reálně možné provést detailnější a zároveň obecně aplikovatelný popis jednotlivých kroků. Rozdělení pipeline na jednotlivé kroky má také mnoho různých možností provedení. Z těchto důvodů je v této práci použit pro ilustraci funkčnosti příklad používající rasterizaci jako vykreslovací metodu. [She]

2.3 Shading a Osvětlovací metody

Shading je způsob vizualizace hloubky ve scéně. Toho je dosaženo použitím různě tmavých odstínů na povrchu objektu v závislosti na materiálu povrchu, světlech ve scéně a použité osvětlovací metodě. Osvětlovací metody jsou součástí procesu renderingu v počítačové grafice. Někdy jsou také nazývány osvětlovací modely. Simulují zjednodušeným způsobem chování osvětlení ve scéně ze vstupních dat, kterými jsou rozložení scény, geometrie a materiály objektů ve scéně a světelné zdroje. V závislosti na konkrétní osvětlovací metodě mohou existovat i další parametry ovlivňující výsledek jako jsou specifické parametry metody, částicové efekty, volumetrické efekty nebo jiné typy objektů ve scéně či jejích parametrů. V některých případech je nelze oddělit co je vykreslovací a co osvětlovací metoda, protože některé metody jako například ray casting řeší obojí funkčnost zároveň.[ea19] Cílem osvětlovacích metod je zvýšit realističnost a grafickou kvalitu scény přidáním světelných efektů. Nejjednodušší osvětlovací metody pouze simulují shading, komplikovanější metody také mohou simulovat zastínění světelného zdroje jiným objektem, stíny, odraz, ohyb a lom světla, či další složitější efekty. Existuje více různých dělení osvětlovacích metod do kategorií. Osvětlovací metody lze rozdělit na lokální a globální. Lokální osvětlovací metody počítají pouze přímé osvětlení, nepřímé osvětlení je v nich nahrazeno ambientní složkou světla, díky čemuž jsou všechny povrchy, které nejsou přímo osvětleny, osvětleny uniformně. Globální osvětlovací metody místo toho simulují odrazy světla od objektů, čímž vytvářejí podstatně realističtější výsledek obzvláště při použití stínů. Další možné rozdělení dělí osvětlovací metody na fyzikálně věrohodné, které se snaží zjednodušeně simulovat fyzikální procesy a empirické, které ignorují

fyzikální procesy a snaží se vytvořit uvěřitelný výsledek jiným způsobem.

2.4 Shader

Shader je typ počítačového programu běžně používaný v počítačové grafice. Standardně je využíván pro shading, grafické efekty a post-processing, implementace rendering pipeline je také z výrazné části tvořena shadery. Většina shaderů je programována pro běh na GPU místo CPU, díky čemuž shadery mají k dispozici omezenou sadu instrukcí. GPU má na běžných PC řádově více jader než CPU, to umožňuje paralelní běh velkého množství vláken, což je nutné pro dostatečně rychlý běh shadingu. Shadery se dělí na různé typy podle toho jaká vstupní data využívají a jaká jsou jejich výstupní data. Mezi typy shaderů patří například vertex shader a fragment shader. Shadery využívané v Unity jsou většinou implementovány v c-sharp za využití knihoven Unity.[ea19]

Kapitola 3

Metody vykreslování

3.1 Rasterizace

Rasterizace je metoda konverze vektorové grafiky na rastrový obrázek, který je následně možné vykreslit. Jedná se o velmi často používanou techniku vykreslování 3d modelů, výhodou rasterizace oproti například. RT je její rychlost a relativně nízká výpočetní náročnost. Rasterizace je pouze proces mapování geometrie scény na pixely, barva pixelů je vypočítávána osvětlovacím modelem. Rasterizace funguje tak, že nejprve rozloží geometrii objektů na trojúhelníky. A následně provede projekci těchto trojúhelníků na 2D obraz podle určitých pravidel, která závisí na konkrétní implementaci rasterizace. Zásadní výhodou rasterizace je její poměrně nízká náročnost na výpočetní výkon a její rychlost. Další výhodou je, že není zároveň osvětlovací metodou a je s ní možné použít mnoho různých osvětlovacích metod. Rasterizace je méně přesná než ray tracing a ztrácí část své efektivity pokud jsou tělesa ve scéně reprezentovány jinak než polygonálně.[Pin88]

3.2 Ray casting

Ray casting je vykreslovací metoda. Funguje na principu, že je z oka kamery vržen skrz každý bod obrazové roviny paprsek jako polopřímka. Následně je pro každý paprsek zjištěn bod průniku s objektem ve scéně. Následně je z

materiálů nalezeného bodu objektu a světel ve scéně zjištěna barva pixelu. Toho je dosaženo, za použití vybrané osvětlovací metody, přičemž zjišťování která světla na pixel působí je učiněno na základě orientace plochy které je bod součástí za principu, pokud je plocha orientována směrem ke zdroji světla tak na ni toto světlo dopadá a není blokováno nebo zastíněno.[App68] Ray casting je výrazně rychlejší než ray tracing z důvodu, že z bodu průniku nejsou vysílány již žádné další paprsky. Díky tomu jde o relativně rychlou a jednoduchou metodu vykreslovaní. Na rozdíl od ray tracingu je výsledný obraz výrazně méně realistický. Ray casting nepodporuje použití globálního osvětlený a bez výrazné nástavby nesouvisející s ray castingem není schopen vykreslovat stíny, odrazy, lom a mnohé další světelné efekty. Oproti rasterizaci lépe pracuje s nepolygonálními objekty. Jako všechny ostatní metody používající paprsky trpí problémem výrazného růstu výpočetní náročnosti s růstem velikosti rozlišení vykreslovaného obrazu.

3.3 Ray tracing

Ray tracing je vykreslovací a zároveň osvětlovací metoda, stejně jako ray casting funguje na principu, že je z oka kamery vržen skrz každý bod obrazové roviny paprsek jako polopřímka a následně je pro každý z nich zjištěn bod průniku s objektem ve scéně. Následně jsou ke všem světlům ve scéně vrženy další paprsky s nimiž je dále nakládáno v závislosti na nastaveném minimálním a maximálním povoleném počtu odrazů a s čím se jako první střetnou. Paprsky které směřují skrz objekt ze kterého vycházejí dále nepokračují pokud není objekt z transparentního materiálu. Paprsky které se střetnou se zdrojem světla jsou vyhodnoceny v závislosti na minimálním počtu povolených odrazů, pokud je parsek alespoň n-tý odraz a minimální počet odrazů je n, tak je vyhodnocen, naopak pokud je paprsek méně než n-tý odraz je zahozen a není s ním dále zpracováván. Pokud se paprsek střetně s jiným objektem než je zdroj světla a pokud jde odražený paprsek s menším počtem odrazů než je maximální povolený počet tak jsou z bodu střetu opět vyslány paprsky ke všem zdrojům světla a ty se řídí opět podle stejných pravidel. [Whi80] Vyhodnocení paprsků se provádí tak, že se pro zjištění jak je objekt osvětlen používají paprsky, které nakonec dorazily do zdroje světla a splnili minimální povolený počet odrazů. Světlo z těchto paprsků je shromážděno a projevují se do jeho výsledné barvy a intenzity i materiály po cestě zasažených objektů. Paprsky které nakonec zasáhly objekt ve scéně se používají k vypočítání stínů.[SP99] Ray tracing má v závislosti na povoleném počtu odrazů tři možnosti chování buď pokud je minimální povolený počet odrazů 1 a maximální také 1 tak ray tracing vypočítává pouze přímé osvětlení. Pokud je minimální povolený počet odrazů 2 a maximální také 2 ray tracing počítá pouze nepřímé osvětlení odražené od 1 následujícího objektu, případně pokud je maximální počet

v této situaci větší tak počítá od více následujících objektů, k podobné situaci dojde i pokud je zvětšen minimální počet odrazů. Minimální počet povolených odrazů musí byt menší nebo roven maximálnímu jinak nebude ray tracing fungovat. Pokud je minimální počet povolených odrazů 1 a maximální větší než 1, ray tracing počítá jak přímé tak nepřímé světlo. Ray tracingu existuje více různých verzí, které se liší například jak je nakonec vypočítáváno osvětlení objektů, či jsou nějak zjednodušeny, aby byla metoda zrychlena nebo rozšířeny, aby metoda vykreslovala i další efekty. Základní metoda ray tracingu je schopna vykreslovat stíny a simulovat odraz, rozptyl a lom světla. Ray tracing je velmi náročná metoda na výpočetní výkon a dlouho nebylo možné použít ho pro real-time rendering, je to možné až v nedávné době díky existenci nových výkonných grafických karet. Výpočetní náročnost ray tracingu je výrazně závislá na maximálním povoleném počtu odrazů, počtu světel ve scéně a počtu odrazivých a transparentních objektů ve scéně. Ray tracing produkuje velmi realistický obraz, avšak cenou za to je jak relativně komplikovaný algoritmus tak jeho vysoká výpočetní náročnost, díky tomu že pracuje na podobném principu jako ray casting je také schopen lépe než rasterizace pracovat s nepolygonálními objekty.[Kaj86] Ray tracing relativně přesně modeluje fyzikální proces šíření světla, díky čemuž relativně přesně modeluje globální osvětlení, světelné a shadingové efekty, přičemž také simuluje stíny, odraz a lom světla i na komplikovanějších objektech v komplexnějších prostředí. Ray tracing také automaticky kombinuje shadingové efekty z více různých objektů korektním způsobem.[IW03]

3.4 Path tracing

Path tracing je modifikovaná verze ray tracingu. Na rozdíl od ray tracingu kde se při každém odrazu paprsky větví, path tracing odráží z každého odrazu pouze jeden náhodným směrem vržený paprsek, čímž nakonec po dosažení maximálního možného počtu odrazů vznikne jedna cesta. Pro každý bod obrazové roviny je vytvořeno větší množství těchto náhodně generovaných cest, přičemž čím větší je počet vypočítávaných cest, tím realističtější je výsledný obraz. Standardně používaný path tracing má metodu vytváření cest postavenou na metodě Monte carlo. PT integruje přes osvětlení přicházející do bodu na povrchu objektu, následné je výsledek zredukován pomocí funkce odrazivosti, a vyslán ke zpracování. Tento proces je opakován pro každý pixel výsledného obrazu. Na rozdíl od RT obsahuje výpočet efektů jako je hloubka pole, měkké stíny, nepřímé osvětlení, ambient occlusion.[Kaj86]

3.5 Stíny a stínové mapy

Stíny jsou ztmavené oblasti do kterých je přístup světla blokován neprůhledným objektem. Stíny jsou v počítačové grafice do scény přidávány procesem zvaným mapovaní stínů(shadow mapping) nebo pomocí některé z vykreslovacích metod založených na využití paprsků(ray tracing, path tracing). Tento proce zjišťuje, zda je pixel viditelný ze zdroje světla. Proces vytváření stínů lze standardně rozdělit na dva kroky, které jsou vytváření stínových map a a aplikace stínů na scénu. V případě, že je k vykreslování scény využívána rasterizace tvorba stínových map nejprve je vykreslena scéna z pohledu světla, toto je provedeno v závislosti na typu světla. Pro světla vycházející z jednoho bodu je pohled vykreslení perspektivní projekce o šířce a úhlu odpovídající parametrům světla. Pro směrová světla je využita ortografická projekce. Z tohoto vykreslení je extrahován a uložen Z-buffer(neboli depth buffer), který obsahuje informace o hloubce, přičemž informaci o barvě a texturách není nutné vypočítávat, tím vznikne hloubková mapa, která se většinou ukládá do grafické paměti jako textura. Hloubková mapa musí být při změně pozice světla nebo relevantních objektů ve scéně aktualizována a pro každé světlo je nutné vytvořit samostatnou mapu. Velmi často se při vykreslování těchto map pracuje pouze s omezenou sadou objektů relevantních pro konkrétní světlo. Takto vzniká stínová mapa, přičemž některá řešení vytvářejí z těchto map stínový atlas.[Wil78] V případě použití ray tracingu lze stíny rovnou vypočítat během vykreslení scény. Toto funguje na principu prodloužení paprsku jdoucích ke světlu na opačnou stranu. Ray casting a path tracing používají podobné řešení jako ray tracing.

3.6 Ambient Occlusion

Je globální osvětlovací metoda, která vypočítává jak je konkrétní bod scény vystaven ambientnímu osvětlení. Výsledkem této metody je však pouze aproximace zastínění, standardní varianta této metody zcela ignoruje ze kterého směru světlo přichází, modernější metody jsou však již schopné informaci o směru osvětlení částečně zahrnout do výsledky metody.[TR09] Toho je dosaženo pomocí výpočtu dostupnosti příslušného bodu. Existuje velké množství algoritmů řešících ambient occlusion. Výsledkem této metody je mapa zastínění ambientního osvětlení. Existuje velké množství různých způsobů výpočtu této metody, HDRP používá 2 z nich a to konkrétně screen space ambient occlusion a ray traced ambient occlusion.

3.7 Screen space

Je označení pro prostor v současném záběru kamery. Tento prostor je velmi často využíván jako omezení, na kterém jsou vypočítávány různé grafické efekty. V případě Unity jsou významnou částí protějšků ray tracingových funkcí, funkce fungující na principu screen space. Toto se týká konkrétně screen space ambient occlusion a screen space reflection.

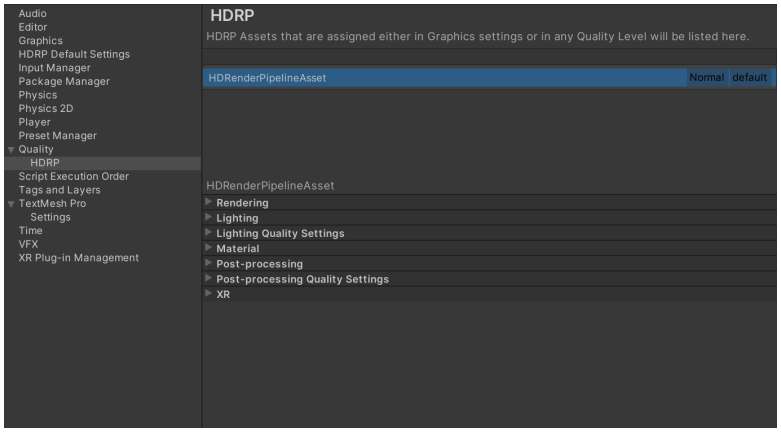
Kapitola 4

Prostředí HDRP a rendering v Unity

4.1 HDRP Pipeline

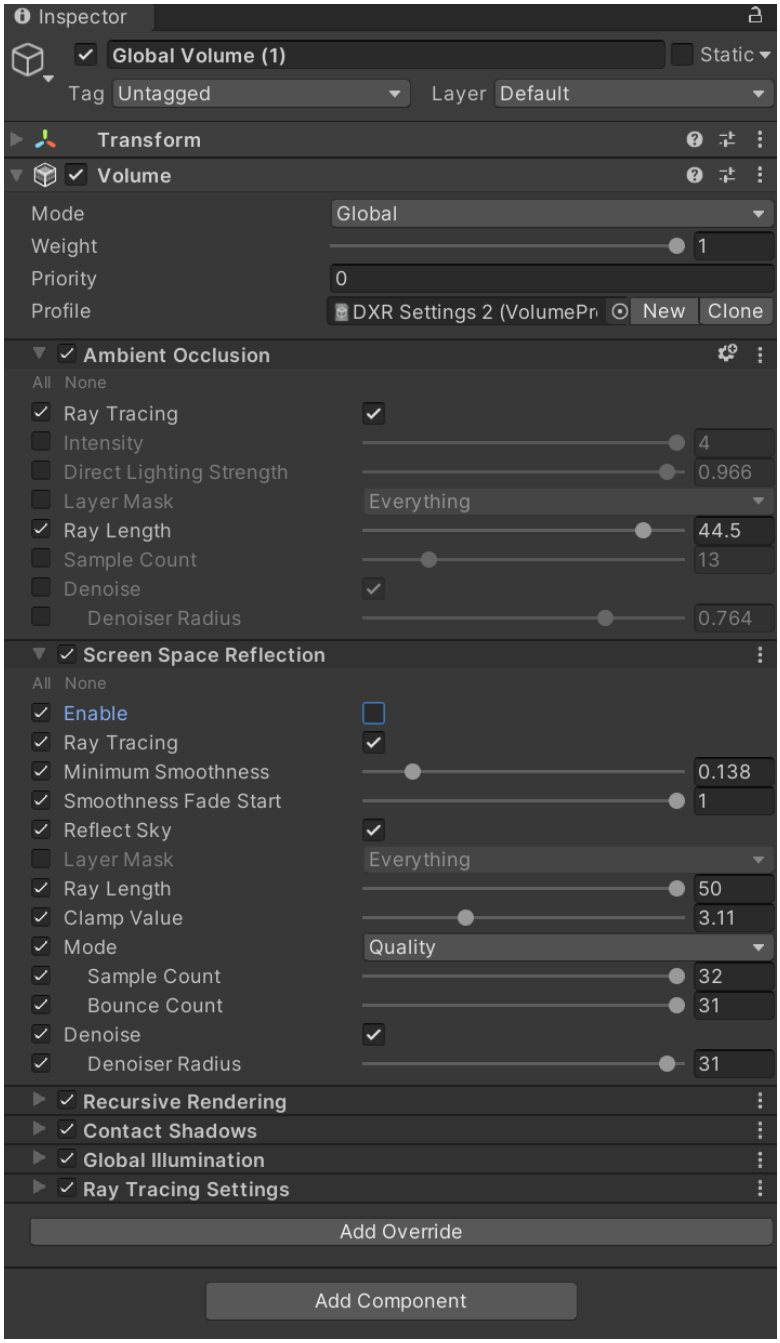
HDRP(High definition render pipeline) je vykreslující pipeline sloužící k vytváření projektů ve vyšší grafické kvalitě než vestavěný vykreslovací pipeline Unity. Nejzásadnějším rozdílem mezi HDRP a vestavěným vykreslovacím pipeline je, že HDRP využívá jinou sadu shaderů. Materiály v Unity mají různé definice v závislosti na tom jaký shader používají. V této práci je primárně využíván materiál postavený na shaderu HDRP/lit. Tento materiál je HDRP verze standardního Unity materiálu s rozšířenou funkčností. Tento materiál má tři klíčové parametry, které mají zásadní vliv na chování materiálu. První z těchto parametrů je Surface Type, což je parametr určující zda se jedná o průhledný či neprůhledný materiál. Druhým zásadním parametrem je Rendering Pass, který určuje zda je či není povrch vykreslován pomocí ray tracingu, každý materiál vykreslovaný pomocí ray tracingu se chová jako lesklý takže ve scéně vykreslované ray tracingem je při současné funkčnosti nutné nechat pro materiály, na než nemají být vrhány odrazy nechat tento parametr ve výchozím nastavení. Třetím zásadním parametrem je Material Type, jenž určuje o jaký typ materiálu se jedná, pomocí tohoto parametru se vytvářejí například iridiscenční materiály. Vlastnosti povrch materiálu jsou z většiny udávány sekcí Surface Inputs, ve které jsou klíčové tři parametry, Base Map udávající barvu povrchu, metallic a smoothnes udávající odrazivé vlastnosti povrchu. Ostatní sekce vlastností materiálu obsahují nastavení různých dalších vlastností materiálu a jsou dostupné v závislosti na základních vlastnostech materiálu. V HDRP assetu, který je dostupný v položce Project settings menu edit, jsou obsaženy základní nastavení týkající renderingu,

osvětlení, postprocessingových efektů a některých dalších funkcí. Mezi těmito nastaveními se vyskytuje povolení využívání ray tracingu jako takového a také omezení vykreslování různých grafických efektů podle jejich maximální vzdálenosti či počtu pro vykreslení a také rozlišení některých grafických efektů jako například odrazy na rovině a prvků jako například skybox.



Obrázek 4.1: HDRP Asset

HDRP využívá pro aplikování velké části funkcí souvisejících s ray tracingem Global volume. V global volume jsou obsaženy nastavení ray tracingu a grafických efektů pro konkrétní scénu. Mezi tyto nastevení se řadí například zda je konkrétní grafický efekt vykreslován pomocí ray tracingu a také různé parametry jako je například maximální počet odrazů paprsků či jejich délka.[Tec]



Obrázek 4.2: HDRP Volume

4.2 Objekty v HDRP

HDRP používá modularní implementaci herních objektů(GameObjects). GameObject je složen prvků objektu a podobjektů, přičemž každý GameObject obsahuje souřadnice, rotaci a měřítko. Ostatní komponenty objektu jsou volitelné. Tyto další komponenty lze rozdělit na několik skupin. Skupina způsobující, že má objekt vykreslitelný vzhled obsahuje například mesh, mesh renderer a materiál, případně animaci či grafický efekt. Další skupinou je skupina pomocných prvků pro vykreslování do které lze zařadit světla, kameru, různé typy probů a další objekty podobného typu. Mezi významnější skupiny se také řadí komponenty využívané pro běh fyziky, mezi které se řadí primárně různé typy colliderů, panty a podobné. Dále také existuje skupina UI obsahující tlačítka, textová pole a další prvky většinou sloužící k vytváření různých menu pro aplikaci. Mezi významnější skupiny se také řadí skupina prvků zajišťujících detekci a obsluhu událostí do které patři například event system, evet trigger a prvky přijímající různé typy vstupu. Posledním velmi významným komponentem jsou skripty, které umožňují pokročilejší řízení objektů ve scéně.[Tec]

4.3 Základní požadavky pro využití RTRT v Unity

4.3.1 Hardwarové požadavky

Unity RTRT běží pouze na novějších grafických kartách značky NVIDIA, přičemž pouze nejnovější generace grafických karet poskytuje hardwarovou akceleraci RTRT. Hardwarová akcelerace sledování paprsku je k dispozici na následujících GPU:

- • NVIDIA GeForce RTX 3060, RTX 3060Ti, RTX 3070, RTX 3080, RTX 3090
- • NVIDIA GeForce RTX 2060, RTX 2080 Super, RTX 2070, RTX 2070 Super, RTX 2080, RTX 2080 Super, RTX 2080 Ti NVIDIA TITAN RTX
- • NVIDIA Quadro RTX 3000 (pouze pro notebook), RTX 4000, RTX 5000, RTX 6000, RTX 8000

NVIDIA také poskytuje ray tracing fallback pro některé grafické karty předchozí generace:

- NVIDIA GeForce GTX
- Turingova generace: GTX 1650, GTX 1660 Super, GTX 1660 Ti
- Pascal generace: GTX 1060, GTX 1070, GTX 1080, GTX 1080 Ti
- NVIDIA TITAN V
- NVIDIA Quadro: P4000, P5000, P6000, V100

Použití grafické karty bez hardwarové akcelerace má však zásadní dopad na rychlost vykreslování.

4.3.2 Softwarové požadavky

Unity real-time ray vyžaduje ke svému běhu DirectX12, z tohoto důvodu běží v současnosti pouze na operačním systému Windows s nainstalovaným DirectX12. Pro použití Unity Real-Time Ray Tracing je nutné použít Unity verze 2019.3 nebo novější a zároveň musí být použit HDRP projekt, přičemž použitá verze HDRP musí být alespoň HDRP 7.1.

4.4 Rendering v HDRP

Unity HDRP standardně využívá k vykreslování scény rasterizaci. Ray tracing umožňuje nahrazení rasterizace pro mesh rendering a také obsahuje sadu grafických efektů umožňujících nahradit mnohé screen space a stínovací efekty. Tyto ray tracingové efekty nejsou mají přístup i k datům mimo prostor obrazovky. HDRP obsahuje náhled ray tracingu od verze 2019.3.

HDRP ray tracing má v Unity 2019.4 následující limitace:

- Nepodporuje skinning, blend shape, alembic, vertexové animace.

- Nepodporuje tesselaci
- Nepodporuje per pixel displacement (parallax occlusion mapping, height map, depth offset)
- Nepodporuje VFX a Terén
- Nepodporuje některé HDRP Materiály, mezi které patří Hair, StackLit, Eye a AxF
- Neobsahuje korektní culling stínů. Využívá frustum culling jako náhradu.

[Tec]

4.4.1 Ambient occlusion v HDRP

Ambient occlusion má v HDRP implementovány 2 různé verze. První verzí je screen space ambient occlusion a druhou je ray traced ambient occlusion. V HDRP je pro výpočet ambient occlusion standardně používána metoda screen space ambient occlusion, která využívá k výpočtu pouze data vyskytující se ve screen space. Toto zjednodušení do určité míry snižuje přesnost této implementace. Ray tracingová alternativa ke screen space ambient occlusion je ray traced ambient occlusion, jejímž hlavním rozdílem je, že může využívat vyskytující se mimo sreen space, což v některých případech umožňuje přesnější výpočet. Tato funkce aproximuje tlumení ambientního osvětlení zastíněním. Obě implementace se na scénu aplikují pomocí Ambient Occlusion Volume Override.[Tec]

4.4.2 Kontaktní stíny v HDRP

Cotact shadows je grafický efekt, který řeší přesnější výpočet menších detailů stínů na povrchu objektů. HDRP obsahuje dvě verze této funkce, první verze HDRP contact shadows pracuje ve screen space, druhá verze ray traced contact shadows pracuje na principu ray tracingu díky čemuž může využívat data mimo prostor obrazovky. Obě implementace se na scénu aplikují pomocí Volume Override.[Tec]

4.4.3 Globální osvětlení v HDRP

Globální osvětlení je v HDRP řešeno pomocí tří různých řešení, kterými jsou lightmap, které jsou založeny na podobném principu jako stínové mapy a nemají real-time verzi, lightprobes fungujících podobně jako reflection probes a Ray Traced Global Illumination řešící globální osvětlení za využití ray tracingu.[Tec]

4.4.4 Odrazy světla v HDRP

Reflection probe

Fungují jako kamera, která snímá obraz okolí ve všech směrech z budu, který je následně promítán na lesklé povrchy nacházející se v oblasti vlivu konkrétního reflection probu. Obraz v reflection probu je uložen jako cubemap a rozlišení tohoto typu probu se nastavuje pro všechny reflection proby v projektu zároveň takže není možné pokud je u některého ze specifických probů požadována větší kvalita tento problém nijak řešit. Reflection proby jsou extrémně náročné na operační paměť z důvodu, že jsou v ní neustále uloženy cubemapy ze všech probů na scéně, což je velkým problémem při použití rozsáhlých scén. Reflection proby se dělí na baked a realtime, přičemž baked reflection proby odrážejí pouze objekty, které jsou označeny jako statické z pohledu odrazů a zcela ignorují dynamické objekty z tohoto pohledu je nelze považovat za konkurenta funkce Ray traced reflections, která odráží všechny objekty. [Tec] Real-time reflection proby odrážejí všechny objekty, takže jsou legitimním konkurentem Ray traced reflections, tento typ probů má tři možná nastavení jak je cubemap aktualizován real-time, on demand a on enable.Real-time mód neustále snímá okolí bez ohledu na to zda je probe a jeho pole působnosti uvnitř zorného pole a je potřeba ho vypočítávat. Toto vede k extrémnímu růstu náročnosti reflection probů na grafickou kartu, což opět vede k velkým problémům s využitím reflection probů v rozsáhlých scénách. Zbylé dva módy aktualizace naopak vyžadují řízení probu pomocí skriptu, což snižuje nároky na grafickou kartu oproti real-time aktualizaci v případě, že nejsou takto využívány všechny proby na scéně.

Planar reflection probe

Funguje velmi podobně jako reflection probe pouze s rozdílem, že snímají obraz okolí z roviny místo jednoho bodu a neexistuje baked planar reflection probe. Obraz z nich je ukládán ve formě 2d textury. Na rozdíl od reflection probes je možné nastavit rozlišení každého planar reflection probe zvlášť což umožňuje regulovat jejich rozlišení v závislosti na jak velkou plochu odráží a důležitosti konkrétního odrazu.[Tec]

Screen space reflection

Screen space reflection je funkce, která vypočítává odrazy objektů viditelných v zorném poli. Tato metoda vytváří vysoce kvalitní odrazy avšak odrážený objekt musí být také obsažen v zorném poli.

Ray Traced reflections

Ray Traced reflections je ray tracingová varianta screen space reflection, která pracuje i s daty mimo screen space díky čemuž dokáže vypočítávat odrazy kompletně a nepotřebuje pro kompletní pokrytí reflection proby.

4.4.5 Síny v HDRP

Stíny jsou v HDRP řešeny buď za pomocí stínových map nebo využitím ray traced shadows, které vytvářejí stíny za pomocí ray tracingu. Ray Traced Shadows fungují pouze pro směrová, bodová a oblastní světla.[Tec]

4.4.6 Light Cluster v HDRP

HDRP používá strukturu k uložení sady světel, které ovlivňují každou oblast. Při rasterizaci HDRP používá strukturu dlaždic pro neprůhledné objekty a strukturu clusterů pro průhledné objekty. Hlavní rozdíl mezi těmito dvěma

strukturami a strukturami používanými pro sledování paprsku spočívá v tom, že tato struktura není založena na zorném poli kamery (frustum). Pro sledování paprsků vytváří HDRP mřížku zarovnanou s osami, která v každé buňce ukládá seznam světel, který se má načíst, pokud v této buňce dojde k průniku. Pomocí Light Cluster Volume Override lze změnit, jak HDRP vytváří tuto strukturu.[Tec]

4.4.7 Path tracing v HDRP

Unity HDRP obsahuje fungující implementaci path tracingu, která je dalším alternativním způsobem vykreslování scén.

4.4.8 Recursive rendering v HDRP

Tato funkce umožňuje alternativní způsob mesh renderingu, tím že nahrazuje rendering pipeline pro GameObjects konstruovaných jako mesh. GameObjects, které používají tento režim vykreslování, vrhají refrakční a odrazové paprsky rekurzivně. To znamená, že když paprsek zasáhne povrch, odrazí se nebo se lomí a pokračuje v zasažení dalších povrchů, počet těchto odrazů či lomů je možné určit. Hladkost materiálu neovlivňuje způsob, jakým se paprsek odráží nebo láme, což činí tento režim vykreslování užitečným pro vykreslování vícevrstvých průhledných GameObjects.[Tec]

4.4.9 Základní nastavení

Před použitím funkcí sledování paprsku v projektu HDRP je nutné nastavit projekt HDRP pro podporu sledování paprsků. HDRP podporuje sledování paprsku pouze pomocí rozhraní DirectX 12 API, takže sledování paprsku funguje pouze v editoru Unity Editor nebo Windows Unity Player, když se vykreslují s DirectX 12. Je třeba změnit výchozí grafické API projektu HDRP z DirectX 11 na DirectX 12 .

Existují dva způsoby, jak toho dosáhnout:

- Použitím Průvodce Render Pipeline

- Ručním nastavením

Průvodce Render Pipeline umožňuje provézt základní nastavení sledování paprsku v HDRP projektu velmi jednoduchým způsobem.

1. Průvodce Render Pipeline je možné otevřít v Window > Render Pipeline vybráním HD Render Pipeline Wizard.
2. Následně je nutné vybrat záložku HDRP + DXR.
3. Pak už jen stačí kliknout na tlačítko Fix All

Následně je ještě nutné správně nastavit kamery. Toho lze dosáhnout buď změnou výchozího nastavení kamer nebo změnou nastavení konkrétní kamery.

Výchozí nastavení lze změnit následujícím způsobem:

1. Otevřením okna Project Settings v menu: Edit > Project Settings a následně vybráním záložky HDRP Default Settings.
2. Následně je nutné vybrat v rozbalovací nabídce Camera, poté rozbalit sekci Rendering a následně je v sekci potřeba zapnout Ray Tracing.

Nastavení Ray Tracing pro specifickou kameru:

1. Nejprve je nutné vybrat konkrétní kameru ve scéně.
2. Následně je v sekci General, nutné povolit Custom Frame Settings.
3. Tímto je umožněno v sekci Rendering section, povolení Ray Tracingu pro tuto konkrétní kameru.

Kapitola 5

Testovací aplikace a prostředí

5.1 Testovací prostředí a metodika testování

Testování chování real-time ray tracingu a s ním souvisejících funkcí je děleno do několika částí. Tyto části jsou děleny jednak podle způsobu provedení testování tak cílových informací z tohoto testování. Podle způsobu provedení testování je v této práci testování děleno na testování provedeno samostatně a uživatelské testy. Podle cílových výsledků se testování dělí kvalitativní testování, hledání defektů a testování výkonu a plynulosti běhu.

5.2 O aplikaci

Aplikace se skládá z hlavního menu a 9 testovacích scén, které jsou zaměřeny na různé aspekty chování RTRT v Unity. Aplikace byla původně postavená na HDRP verzi 7.3.1 později byla však konvertována do verze 7.5.2. Aplikace byla původně vytvářena v Unity 2019.3, ale později bylo z důvodu vydání verze 2019.4, která není na rozdíl od 2019.3 beta verzí, ale jedná se o verzi s dlouhodobou podporou. RTRT obsažený ve verzích 2019.3 má některé zásadní rozdíly od RTRT obsaženého v 2019.4, ale tyto rozdíly nejsou v práci zpracovány z důvodu, že Unity 2019.3 je již v současnosti zastaralé.

Aplikace je ovládána kombinací klávesnice a myši. Ve většině scén se

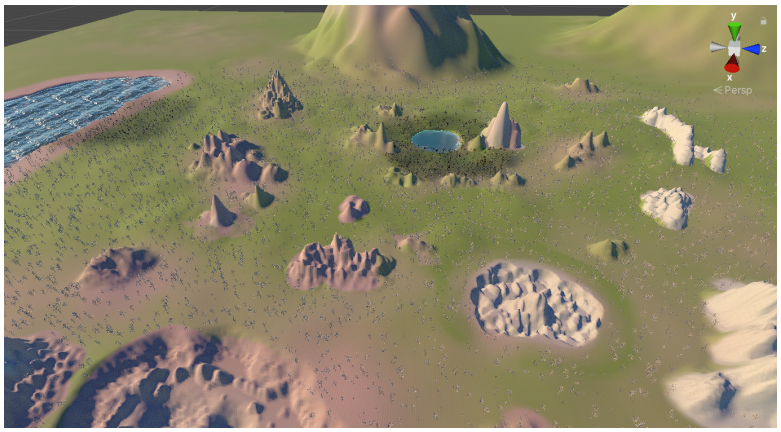
vyskytuje avatar hráče v podobě jednoduchého robota ve tvaru kapsle. Tento avatar slouží jako pohyblivá kamera a má schopnost jak chůze, letu a také rozhlížení se ve stylu FPS her, což umožňuje prohlédnout si chování RTRT z mnoha různých úhlů.

RTRT je v aplikaci možné vypnout a zapnout pouze z hlavního menu. Důvodem tohoto, je že po přepnutí je z důvodu způsobu technického řešení RTRT v Unity při přepnutí nutné znovu načíst scénu.

5.3 Testovací scény

5.3.1 Scéna 1

Scéna 1 je exteriérová scéna zobrazující zalesněnou krajinu s komplexním terénem, scéna je také z většiny pokrytá stromy umístěnými pomocí mass place trees. Pro testování klíčové lokace sa nacházejí u dvou vodních ploch na scéně. Kolem menší z vodních ploch a u jedné strany větší vodní plochy se u každé z ploch nacházejí 4 skupiny stromů umístěných jako game objecty. Standardně je aktivní pouze jedna ze 4 skupin stromů další tři skupiny jsou posunuté kopie první skupiny a lze je postupně aktivovat a deaktivovat pomocí kláves k a j. Z pohledu testování vlastností ray tracingu se tato scéna primárně zaměřuje na demonstraci současné implementace ray tracingu v Unity pro využití v rozsáhlých exteriérových scénách, stíny stromů a rozdíly ve výkonu vzhledem k počtu aktivovaných skupin stromů.



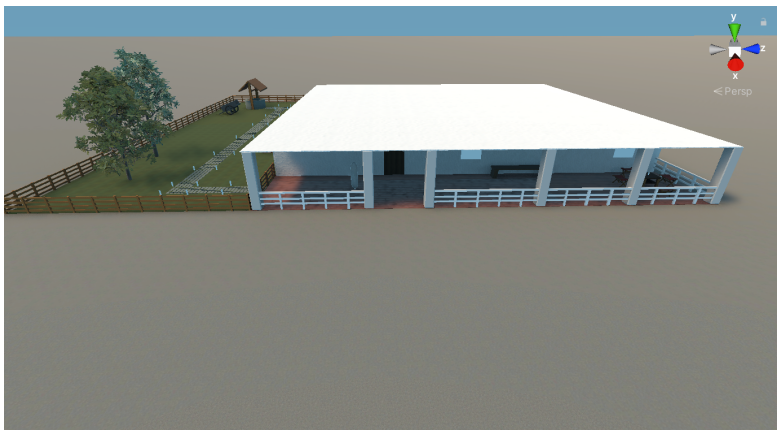
Obrázek 5.1: Scéna 1 RTRT

Klíčové prvky scény:

- Stromy - Globální iluminace osvětlí jinak neosvětlené části, jako spodní strany větví, atd...
- Terén - I když není podporovaný ray tracingem, tak je na něm dobře efekt viditelný globální iluminace.

5.3.2 Scéna 2

Scéna 2 obsahuje dům se zahradou. Zahrada je oplocená a vyskytují se na ní stromy, studna a různé další objekty. Dům se skládá z terasy, která obsahuje stolek se židlemi, kuchyně obsahují kamna, jídelní stůl se židlemi, zrcadlovou sín se stěnami pokrytými zrcadly a zrcadlovými sloupy ve tvaru kvádru s čtvercovou podstavou a obývací pokoj obsahující různé typy nábytku. Z pohledu testování se tato scéna primárně zaměřuje na chování osvětlení a odrazů v interiéru a průchod světla okny do budovy.



Obrázek 5.2: Scéna 2 RTRT

Klíčové prvky:

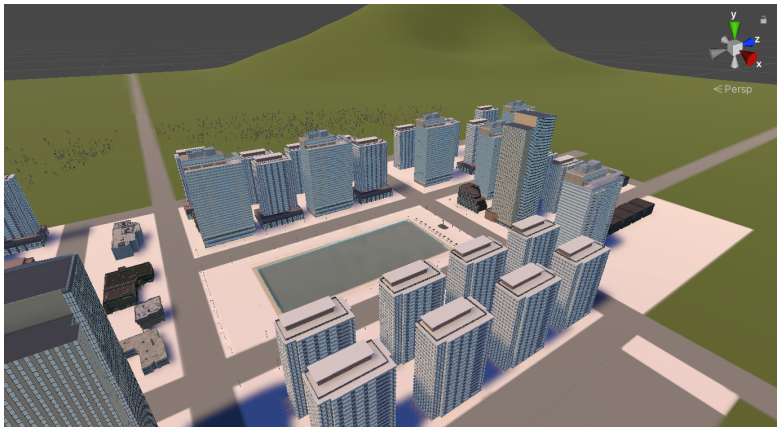
- Okna -Průhlednost a odrazy okolí, ztráta kvality díky šumu.
- Zrcadla - Odrazy a vysoký šum.
- Lampy - Real Time ray tracing stínů.

Rtrt materiály:

- zrcadlo: rendering pass ray tracing, maximální parametry metallic a smoothness, base map bílá barva.
- sklo: rendering pass ray tracing, maximální parametr smoothness, nízký metallic, base map bílá barva, alpha 26.

5.3.3 Scéna 3

Scéna 3 je zjednodušená část moderního města s krajinou okolo. Ve středu města se nachází vodní nádrž a kolem ní se nacházejí různé převážně výškové budovy. Testování scény se zaměřuje na chování osvětlení a stínů v městské scéně a také chování odrazů ve venkovním městském prostředí.



Obrázek 5.3: Scéna 3 RTRT

Klíčové prvky scény:

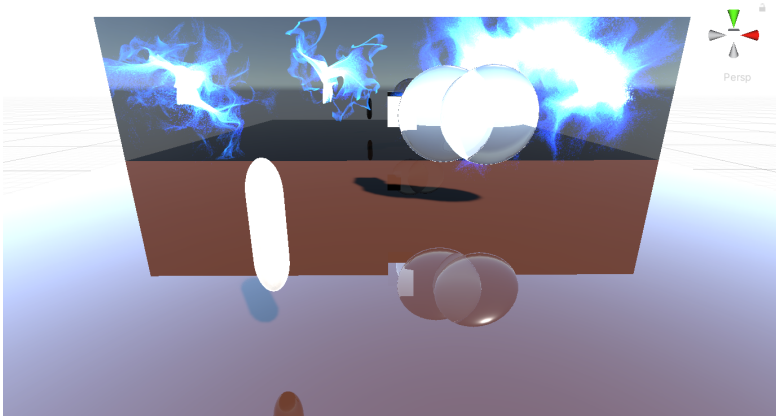
- Mrakodrap skleněný - sledovaní odrazu okolí a obzoru - viditelné problémy rtrt – neodráží se terén
- Budova s venkovní posezením - odraz ve skleněném přízemí budovy - výrazný rozdíl mezi rasterizací a rtrt
- Vodní plocha - Je tvořena materiálem, který má vlastní vertex shader. - Vertex shadery nejsou ray tracingem podporované, a proto není v odrazech viditelná.

Rtrt materiály

- sklo: rendering pass ray tracing, vysoké parametry metallic a smoothness, base map šedá barva.

5.3.4 Scéna 4

Scénu 4 tvoří dvě plochy, tři particle efekty, krychle a deformovaná koule. Tato scéna ukazuje rozdíly v chování odrazu a lomu světla mezi ray tracingem a reflection proby.



Obrázek 5.4: Scéna 4 RTRT

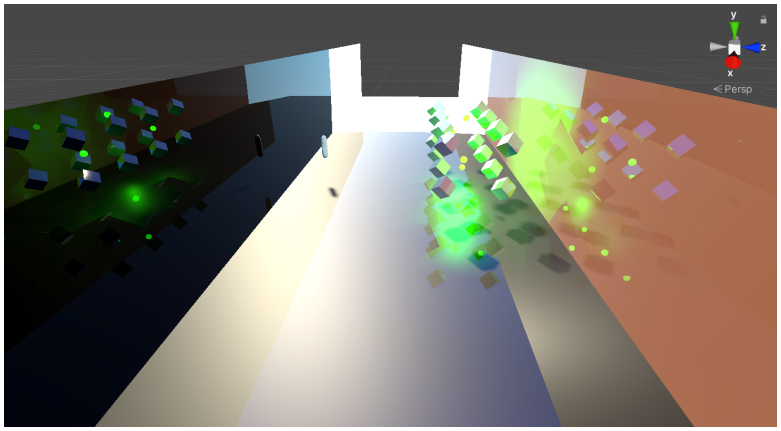
Klíčové prvky scény:

- Particle efekty - Ukazují že nejsou podporovány ray tracingem.
- Plochy - Neodrážejí particle efekty.
- Koule - Zakrývá particle efekt a také deformuje obraz krychle.

5.3.5 Scéna 5

Scénu tvoří čtyři plochy, šestnáct krychlí a čtyři koule s bodovými světly. Krychle a světla se pohybují. Tato scéna se z pohledu testování soustředí

na chování odrazů a osvětlení mezi pohybujícími se světly a objekty a jejich působením na plochy umístěné poblíž nich.



Obrázek 5.5: Scéna 5 RTRT

Klíčové prvky:

- Krychle - slouží jako ukázka RTRT na pohybujícím se objektu.
- Koule - Obíhají okolo svého bodu

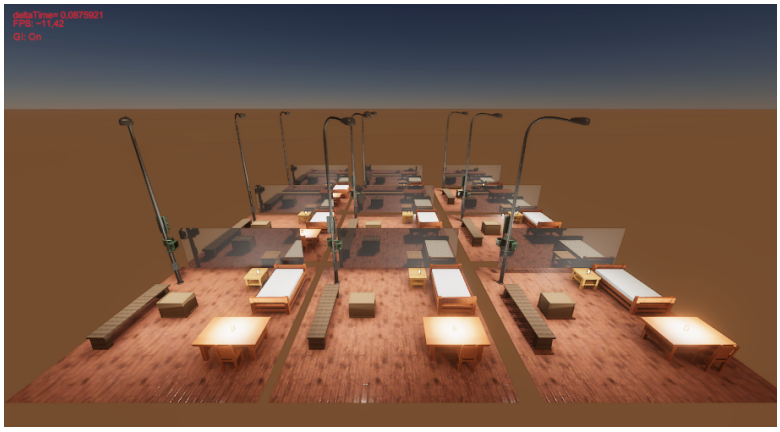
RTRT materiály

- materiál krychlí: má bílou barvu, maximální parametr metallic a skoro maximální smoothness.
- materiál koulí je průhledný, má minimální parametr metallic a velmi nízký smoothness, má zelenou barvu a alfu 183, je emisivní, s intenzitou 100Nits.

5.3.6 Scéna 6 až 9

Scéna 6 se skládá z jednoho čtverce podlahy, na kterém je umístěno několik kusů nábytku a jiných assetů použitých v ostatních scénách a jednoho zrcadla. Všechny assety mají rendering pass nastavený na ray tracing. Scéna 7 se skládá ze čtyř těchto čtverců, scéna 8 z devíti a scéna 9 z šestnácti. Cílem

těchto scén je zjistit náročnost RTRT v závislosti na počtu renderovaných objektů při vysokém nastavení všech overridů.



Obrázek 5.6: Scéna 8 rasterizace

5.4 Testovací prostředí

Pro testování byly využity dvě sestavy hardwaru. Sestava 1 obsahuje grafickou kartu NVIDIA GeForce GTX 1080 která nemá přímou hardwarovou podporu takže není dostatečně výkonná na komplexnější testy. Z tohoto důvodu byla sestava 1 použita pouze pro testy výkonu z fixních kamer ve scéně. Všechny ostatní testy byly provedeny za použití sestavy 2.
Testovací sestava 1:
AMD Ryzen 7 1700X
8 jader procesoru
16 vláken
Base Clock 3.4GHz
Max Boost Clock 3.8GHz
L1 Cache 768KB
L2 Cache 4MB
[AMDa]
NVIDIA GeForce GTX 1080
Architektura GPU - Pascal
Snímkový zásobník - 8 GB GDDR5X
Rychlost paměti - 10 Gbps
Boost Clock - 1733 MHz
[Cora]
16GB RAM

Testovací sestava 2:

AMD Ryzen 7 3700X
8 jader procesoru
16 vláken
Base Clock 3.6GHz
Max Boost Clock 4.4GHz
L1 Cache 512KB
L2 Cache 4MB
L3 Cache 32MB
[AMDb]

NVIDIA GeForce RTX 3070
Architektura GPU - Ampere
Snímkový zásobník - 8 GB GDDR5X
Rychlost paměti - GDDR6
Boost Clock - 1.73 GHz
[Corb]

32GB RAM

Kapitola 6

Testy

Testování ray tracingu v této práci lze rozdělit na dvě různé fáze testování. První fáze zkoumá grafickou kvalitu a defekty objevené jak při použití rasterizace tak ray tracingových metod. Druhá fáze se zaměřuje na měření výkonu a plynulosti vykreslování v závislosti na aktivaci konkrétních funkcí ray tracingu a rasterizace.

6.1 Testování výkonnosti

Tato část testování se zaměřuje na porovnávání různých verzí scén z pohledu GPU time a FPS.

6.1.1 Test výkonu na testovací sestavě 1

Na testovací sestavě 1 byl proveden pouze test výkonnosti na méně komplexních scénách 4 až 9. Měření proběhlo na kameře 1 pro všechny tyto scény kromě scény 5 u které byly využity všechny 3 statické kamery. Cílem tohoto testy bylo zjistit, zda je Unity real-time ray tracing použitelný na starším hardwaru. Tento test byl proveden v menším rozlišení než následující testy, takže není možné číselně porovnávat jeho výsledky s ostatními testy.

Testovací sestava 1								
	Scéna 4	Scéna 5.1	Scéna 5.2	Scéna 5.3	Scéna 6	Scéna 7	Scéna 8	Scéna 9
Rasterizace	16,74	6,85	6,13	7,88	11,85	19,85	26,18	25,41
RTRT	41,39	69,92	74,21	55,65	20,26	41,03	67,84	93,67
RTRT+GI	65,13	111,76	117,15	94,93	32,97	79,42	153,39	313,32

Tabulka 6.1: Scéna 4-9 výsledky, Hodnoty výsledků testů udávají GPU time v ms

Zhodnocení výsledků testu

Tento test ukázal, že i na relativně jednoduchých scénách jako jsou scény 4 až 6, je pokles výkonu aplikace při použití real-time ray tracingu na grafických kartě NVIDIA GeForce GTX 1080 příliš vysoký pro reálné použití v praxi. Toto dokazuje, že na grafických kartách stejné generace jako je použitá karta nelze Unity real-time ray tracing v praxi použít, protože NVIDIA GeForce GTX 1080 patří z pohledu výkonu ve své generaci grafických karet výrazně nad průměr.

6.1.2 Test výkonu na testovací sestavě 2 Scény 4 a 5

Tento test se soustředí na měření výkonu na scénách 4 a 5. Testovací výsledky byly získány z jedné statické kamery v každé ze scén. Cílem testu je porovnat z pohledu výkonu ray tracingové a rasterizační řešení scén 4 a 5, přičemž scéna 4 se soustředí na odraz a lom světla ve statické scéně, scéna 5 se soustředí na odraz světla v dynamické scéně. Rasterizační verze scény 4 obsahuje jeden reflection probe a jeden planar reflection probe a scény 5 obsahuje pět reflection probů. Všechny reflection proby jsou nastaveny na snímání v reálném čase. Testovaná nastavení scén jsou rasterizace, ray tracing a obě tyto varianty se zapnutou funkcí global illumination.

Test výkonu na testovací sestavě 2 Scény 4 a 5		
	Scéna 4	Scéna 5
Rasterizace	26,14	19,61
Rasterizace+GI	30,15	17,18
RTRT	17,08	9,94
RTRT+GI	19,24	13,07

Tabulka 6.2: Scéna 4 a 5 výsledky, Hodnoty výsledků testů udávají GPU time v ms

Zhodnocení výsledků testu

Tento test ukázal, že výpočet odrazů a lomu v takto jednoduchých scénách je z pohledu výkonu efektivnější provádět za využití ray tracingu.

6.1.3 Test výkonu na testovací sestavě 2 Scény 6 až 9

Tento test se soustředí na měření výkonu na scénách 6 až 9. Testovací výsledky byly získány z jedné statické kamery v každé ze scén. Každá řada na scéně standardně obsahuje jeden planar reflection probe, tento testovací scénář používá dvě různé varianty od každé scény, varianta -Z odebírá vliv reflection probů z výsledků, protože jsou v ní deaktivována zrcadla ve scéně. Testovaná nastavení scén jsou rasterizace, ray tracing a obě tyto varianty se zapnutou funkcí global illumination.

Test výkonu na testovací sestavě 2 Scény 6 až 9				
	Scéna 6	Scéna 7	Scéna 8	Scéna 9
Rasterizace+GI-Z	13,03	22,60	36,55	60,89
Rasterizace+GI	17,80	39,70	87,59	100
Rasterizace-Z	13,23	22,51	40,03	60,27
Rasterizace	17,71	38,21	81,62	100
RTRT+GI-Z	14,59	25,56	40,97	60,69
RTRT+GI	14,59	26,20	41,96	62,65
RTRT-Z	14,20	25,51	40,74	59,52
RTRT	14,20	26,13	41,63	61,87

Tabulka 6.3: Scéna 6-9 výsledky, Hodnoty výsledků testů udávají GPU time v ms

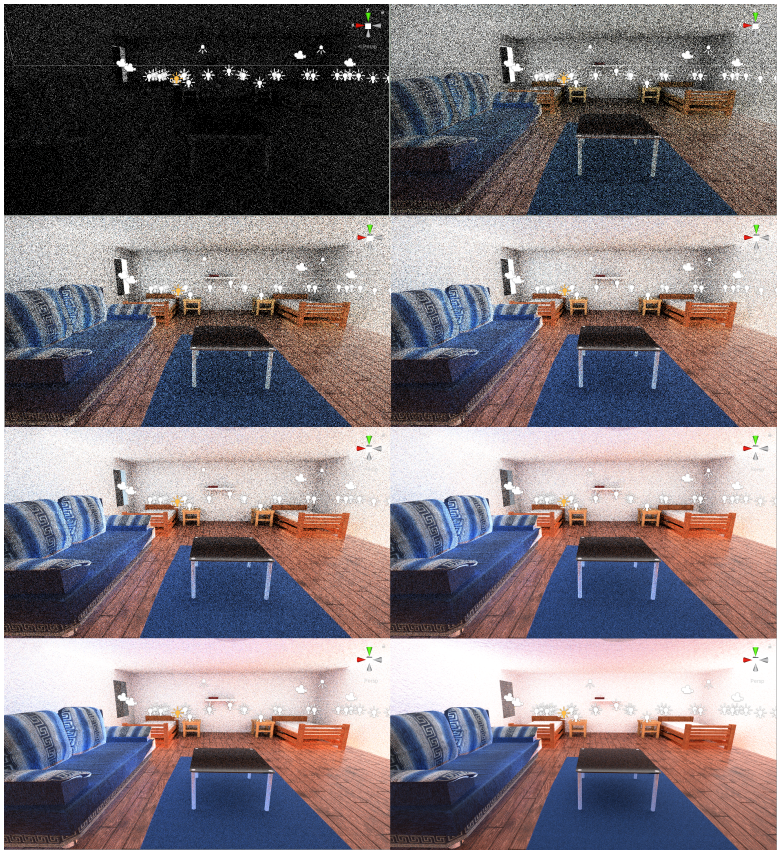
Zhodnocení výsledků testu

Výsledky tohoto testu ukazují, že pokud se ve scéně nevyskytují zrcadla je pokles výkonu přibližně stejný pro rasterizaci a ray tracing. Další informace získaná z tohoto testu je, že rychlost vykreslování ray tracingové scény je v obou případech jak se zrcadle tak bez přibližně stejná. Tento test taká dokazuje, že využití planar reflection probů má zásadní vliv na výkon a ray traced reflection jsou na testovací sestavě 2 výrazně efektivnější.

6.2 Testování funkčnosti

6.2.1 Test funkčnosti path tracingu na sestavě 2

Bylo opakovaně prováděno vykreslení obývacího pokoje ve scéně 2 v editoru, přičemž byl mezi jednotlivými vykresleními měněn maximální počet vzorků. Byly vykresleny verze pro 1 vzorek a následně pro 8 až 512, přičemž každý snímek z této sady měl dvojnásobný měněn maximální počet vzorků. Dále byl také řádově sledován čas jednotlivých vykreslení.



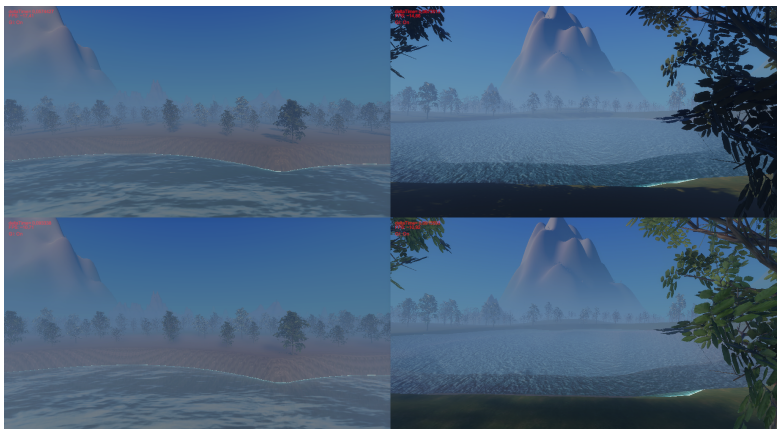
Obrázek 6.1: Path tracing - scéna 2, 1 až 512 vzorků postupně po řádcích zleva do prava

Zhodnocení výsledků testu

Veškeré snímky kromě snímku s maximálním počtem vzorků 512 vykazují příliš vysokou hladinu šumu pro využití v aplikaci uživatelem. Vykreslení všech snímků s maximálním počtem vzorků více jak 100 trvá čas v řádu sekund, přičemž pro relativní plynulost aplikace je nutné minimálně vykreslit přibližně 20 snímků za sekundu což v kombinaci se hladinou šumu vyskytující se ve snímcích ukazuje, že path tracing nelze v tuto chvíli v Unity uplatnit jinak než pro vykreslení scény vývojářem v editoru či tvorbu cutscén.

6.2.2 Scéna 1 stíny stromů a mlha

Tento testovací scénář ukazuje chování stínů stromů v zamlžené krajině. V tomto případě ke vidět, že stíny vykreslené pomocí RTRT jsou v zamlženém prostředí realističtější.



Obrázek 6.2: Scéna 1, dva různé pohledy, horní snímky rasterizace, dolní snímky RTRT

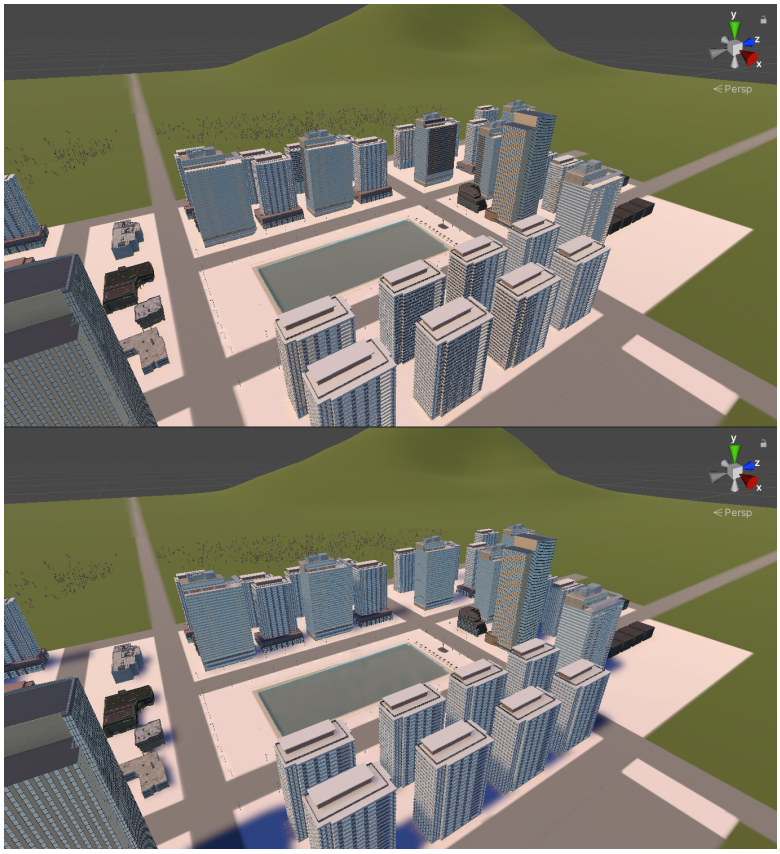
6.2.3 Scéna 3 odraz v oknech budovy

Tento testovací scénář porovnává odraz posezení v oknech budovy. Na tomto obrázku je vidět, že neschopnost RTRT odrážet terén je zásadním problémem.

Obrázek 6.3: Scéna 3 budova s posezením, horní snímek rasterizace, dolní snímek RTRT

6.2.4 Scéna 3 pohled z výšky

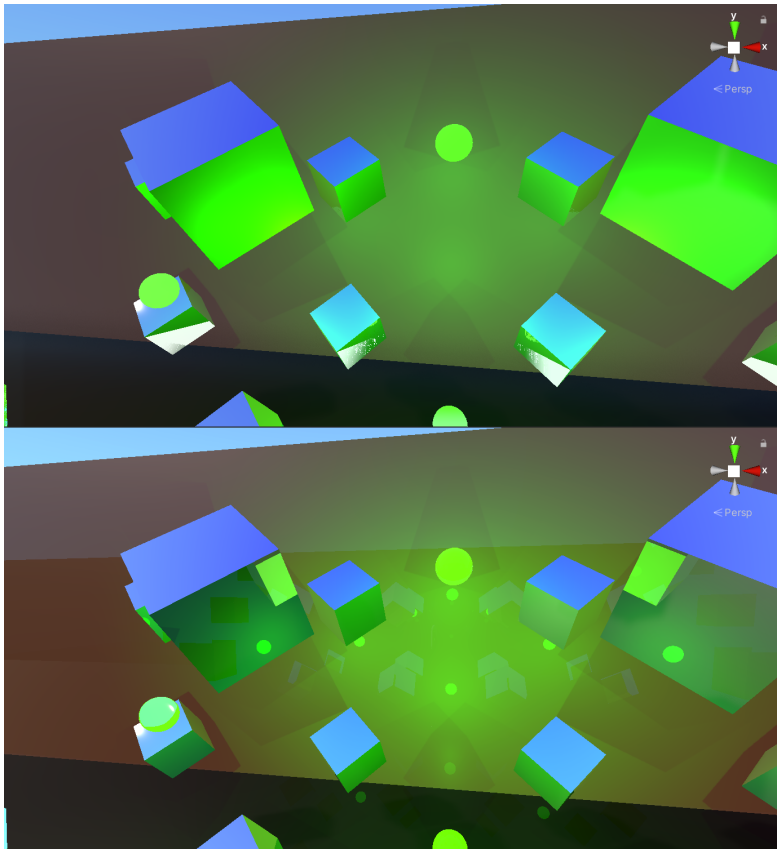
Tento testovací scénář ukazuje, že stíny vrhané RTRT jsou viditelné z větší vzdálenosti než alternativa.



Obrázek 6.4: Scéna 3 pohled z výšky, horní snímek rasterizace, dolní snímek RTRT

6.2.5 Scéna 5 odraz a stíny

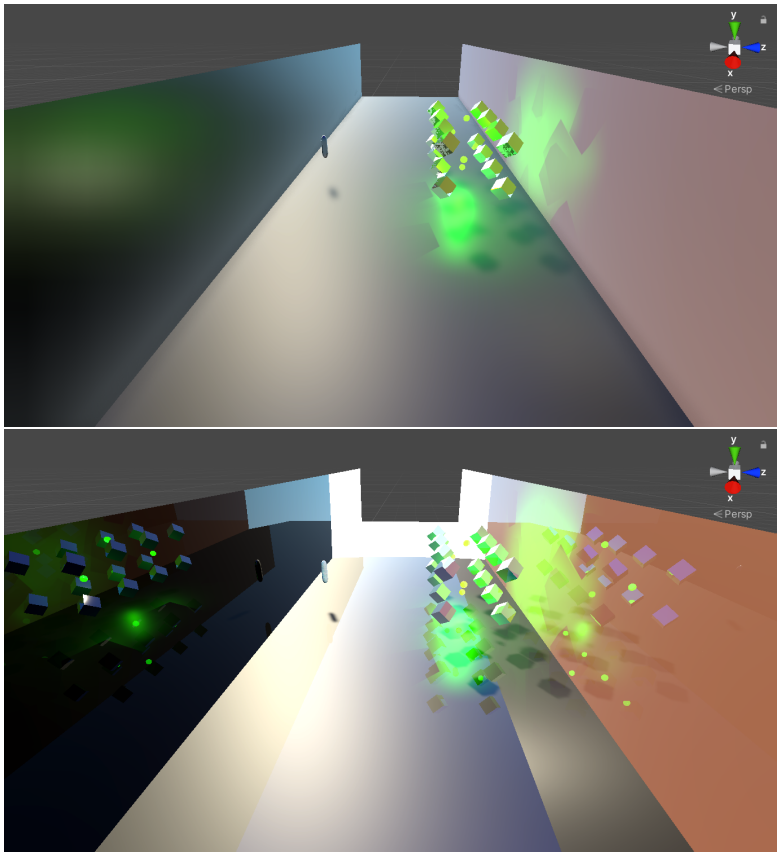
Tento testovací scénář porovnává odrazy a stíny na krychlích a pozadí. Na snímcích je vidět, že kvalita obou variant stínů je velmi podobné avšak kvalita odrazů RTRT verze je zásadně lepší a realističtější.



Obrázek 6.5: Scéna 5 detail - horní snímek rasterizace, dolní snímek RTRT

6.2.6 Scéna 5 celkový vzhled

Tento testovací scénář ukazuje že RTRT je schopno vytvářet podstatně přesnější odrazy a reflection proby jsou velmi limitovány svou kvalitou v případě většího využití v nestatické scéně.

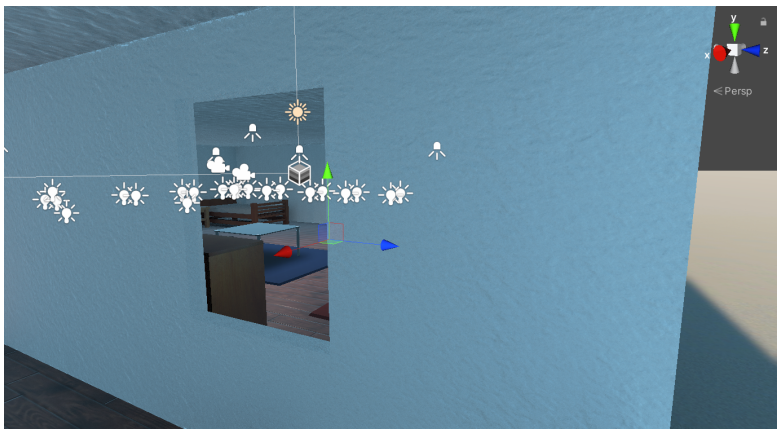


Obrázek 6.6: Scéna 5 pohled z boku

6.3 Grafické defekty

6.3.1 Scéna 2 RT - nevykreslené okno

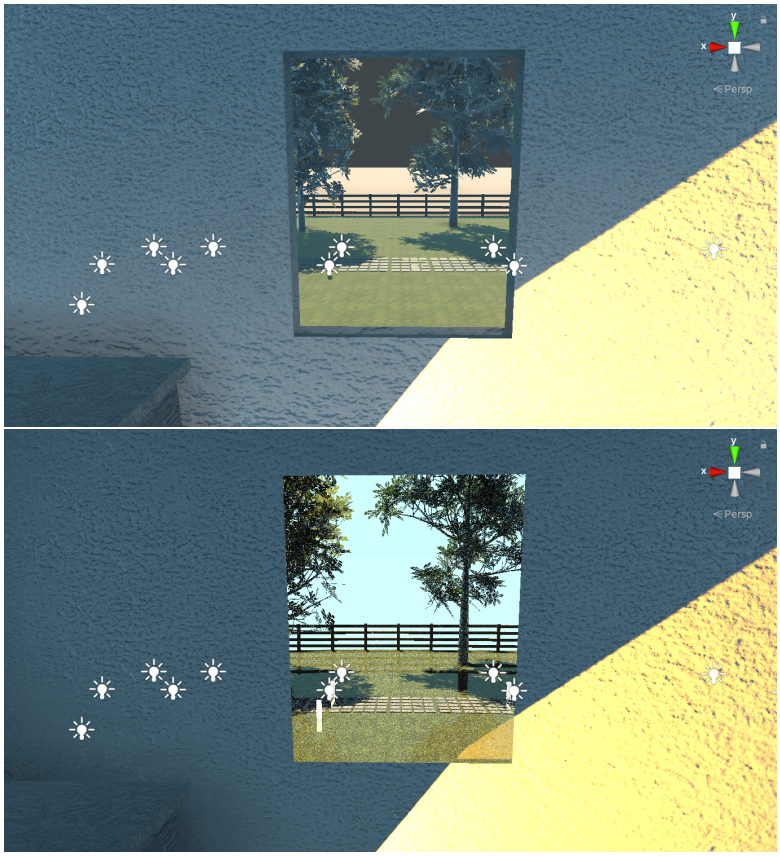
Tento grafický defekt se vyskytuje ve scéně 2 v případě, že je deaktivovaný recursive rendering, díky čemuž v tomto případě dochází k tomu, že okno není vykresleno.



Obrázek 6.7: Scéna 2 vypnutý recursive rendering

6.3.2 Scéna 2 RT - chybný lom světla procházejícího oknem

Kvádrové okno na druhém z obrázků(RTRT) využívá refrakční model box. Tento model a stejně tak i model thin jsou pro ray tracing špatně nadefinované a chovají se přibližně stejně jako model sphere, který slouží k modelování průchodu světla objekty kulovitého tvaru a čočkami. Toto způsobuje kulovitou deformaci objektů nacházejících se za oknem. První z obrázků ukazuje vykreslení stejného okna pomocí rasterizace.



Obrázek 6.8: Scéna 2 pohled z okna: rasterizace, RTRT

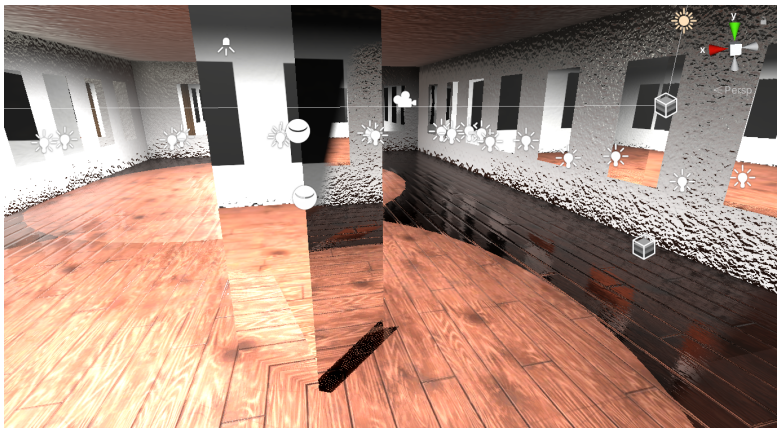
6.3.3 Scéna 2 RA - průchod světla skrze dveře a zdi

Při použití rasterizace bez globálního osvětlení dochází při některých kombinací různých faktorů(úhel světla, kvalita stínové mapy, rozmístění ostatních objektů ve scéně) k případům, že část světla projde do jiného prostoru zcela obklopeného ostatními objekty. V tomto konkrétním případě se v zakryté místnosti objevuje prstenec světla odpovídající okrajové části stropní lampou vyzařovaného kuželu.

Obrázek 6.9: Scéna 2

6.3.4 Scéna 2 RA - konflikt reflection probe a screen space reflection

Na zrcadlových sloupech ve scéně 2 vykreslené rasterizací dochází při pohledu z některých úhlů ke konfliktu mezi reflection probe a screen space reflection, výsledkem čehož je nesouvislý odraz na sloupu a v některých případech i další grafické defekty.

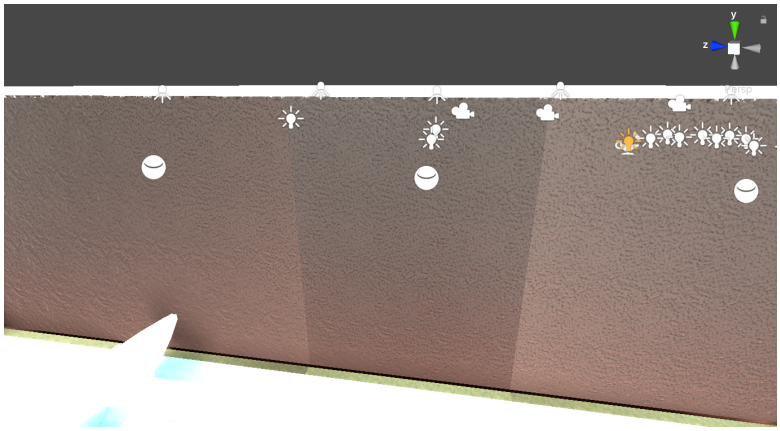


Obrázek 6.10: Scéna 2 chybný odraz

6.3.5 Scéna 2 RA - reflection probe

Reflection probe může v některých případech ovlivňovat objekty v jeho oblasti působení špatně předvídatelným způsobem. V tomto konkrétním případě,

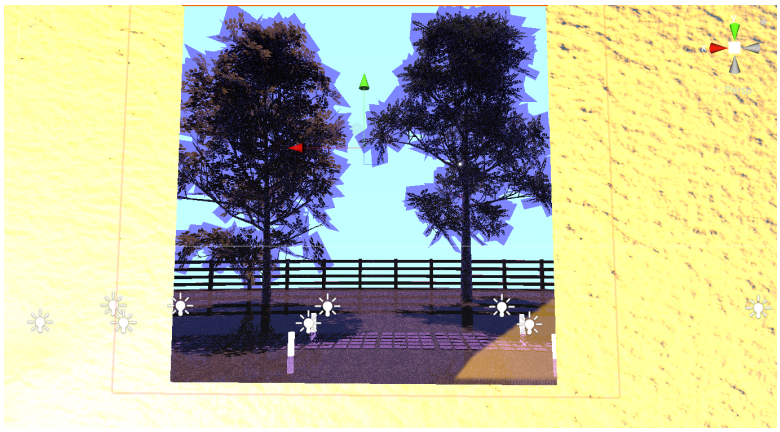
přestože mají všechny proby uvnitř v místnosti mají zcela shodné vlastnosti kromě pozice podél zdi, tak prostřední z nich ovlivňuje stěnu místnosti z venkovní strany.



Obrázek 6.11: Scéna 2 chyba reflection probe

6.3.6 Scéna 2 RT - transmitance color

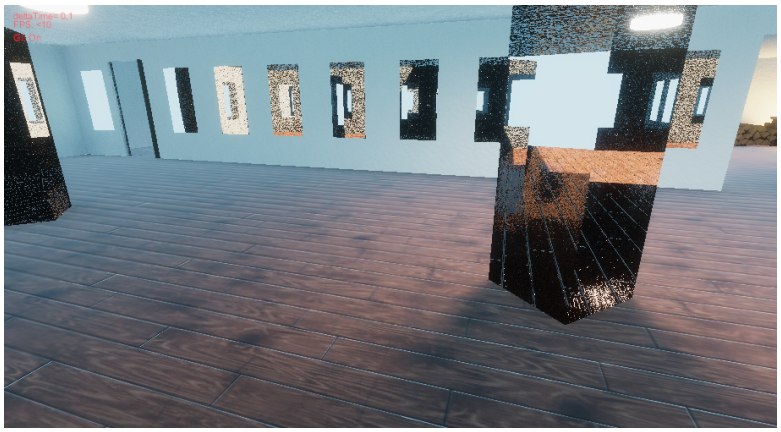
V případě, že je transmitance color nastavená jinak než, aby neměnila barvy objektů za průhledný objektem a zároveň se za objektem vyskytuje objekt pokrytý poloprůhledný materiálem, například listy stromu vytvořené jako průhledný objekt pokrytý materiálem s zcela průhledným základem a částečně pokrytým texturou, dojde k obarvení celého objektu zakrytého průhledným objektem včetně průhledné části. Okno v této verzi scény má nastavenou transmitance color tak, že objekty viděné skrze okno jsou zabarvené. Chybou v tomto případě je, že pohled skrze okno barví i zcela průhledné části objektů za oknem, konkrétně objekty obsahující listy stromů.



Obrázek 6.12: Scéna 2 RTRT pohled skrz barevné okno

6.3.7 Scéna 2 RT - problém s vícenásobným odrazem

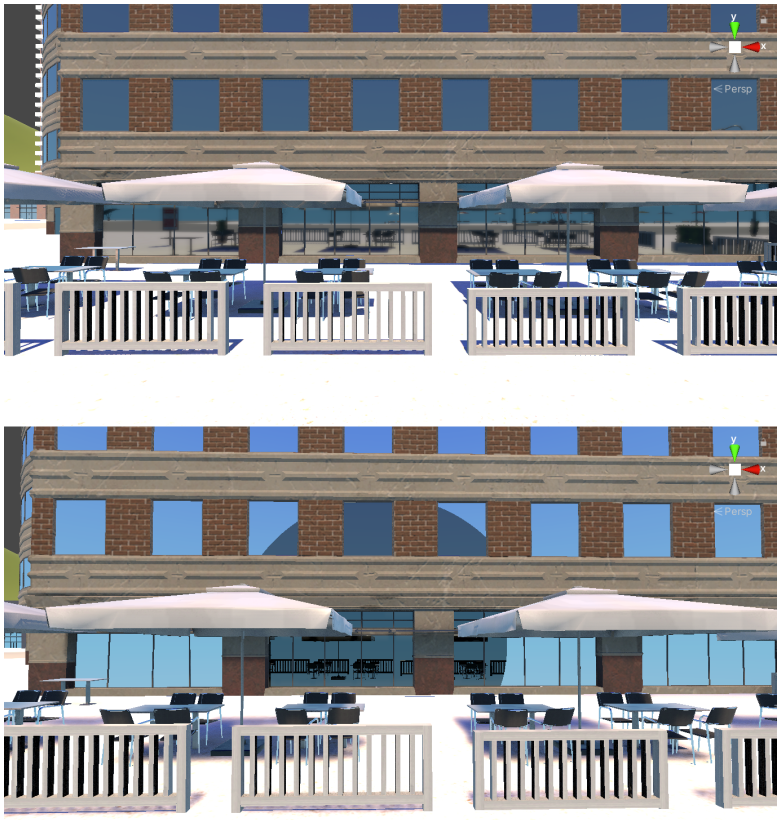
V tomto případě dochází na zrcadlovém sloupu v místnosti ke špatné kombinaci vícenásobných odrazů mezi jednotlivými zrcadly.



Obrázek 6.13: Scéna 2 RTRT

6.3.8 Scéna 3 RT - odraz v oknech z větší vzdálenosti

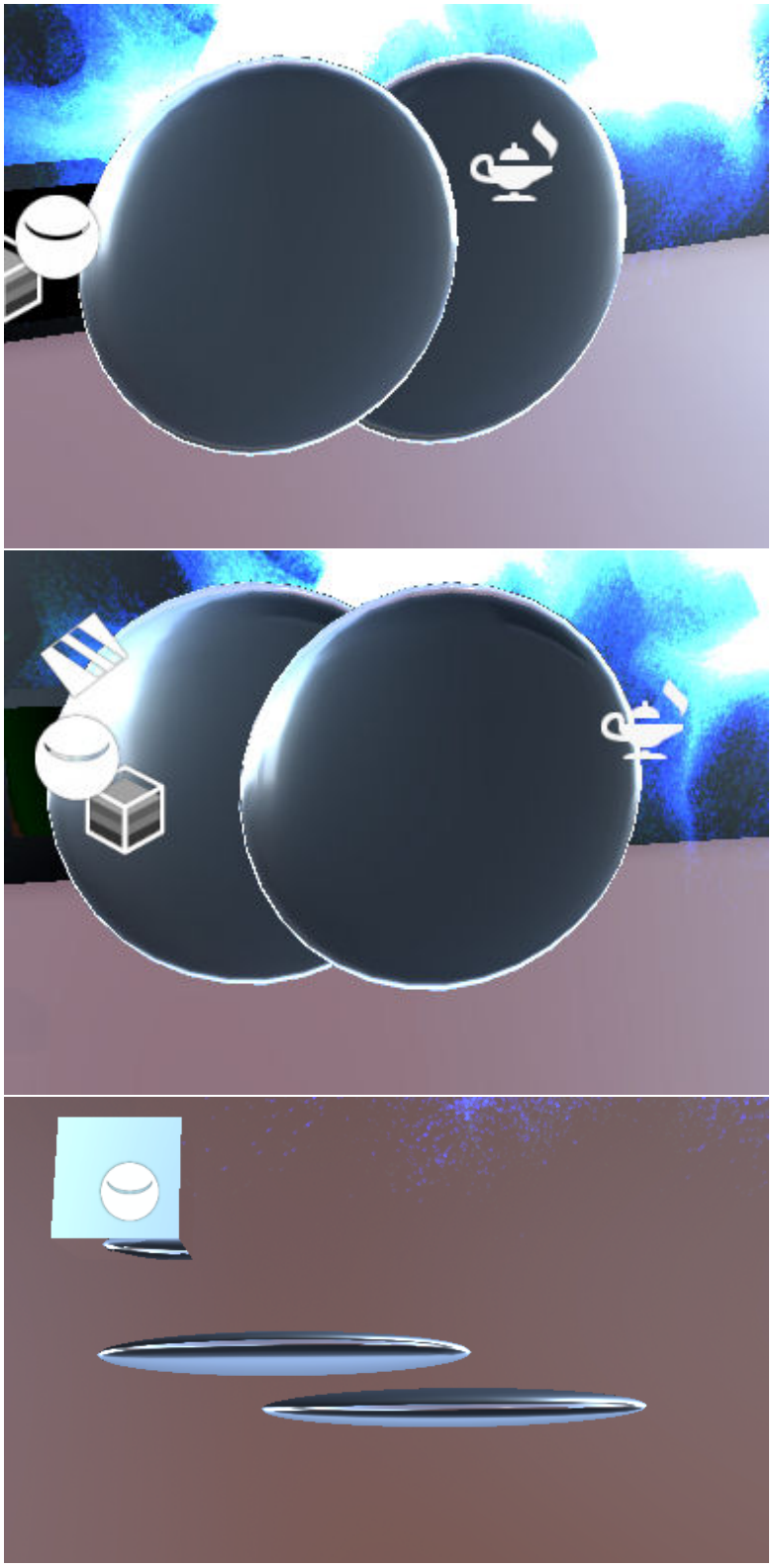
Ray tracingový odraz se na budově vykresluje pouze do určité vzdálenosti od kamery, což má za následek vznik kruhu s odrazem na budově.



Obrázek 6.14: Scéna 3 budova s posezením z větší vzdálenosti, horní obrázek rasterizace, dolní obrázek ray tracing

6.3.9 Scéna 4 RA

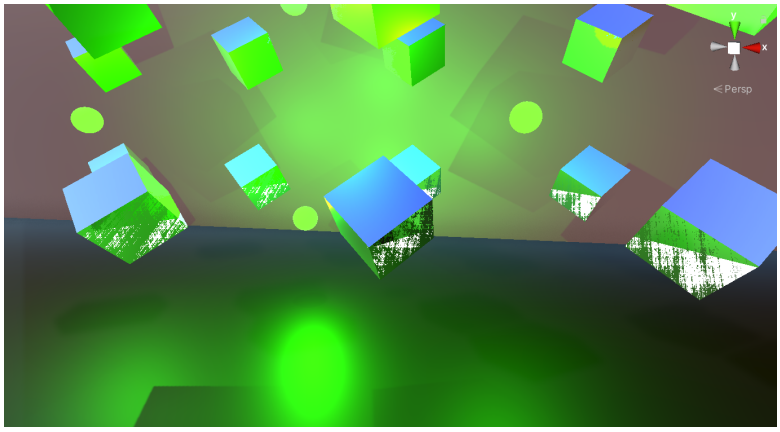
Objekty jsou na prvním obrázku(rasterizace) vykresleny v opačném pořadí než jak jsou umístěny ve scéně. V tomto případě jde o špatnou detekci, který z objektů je blíže ke kameře.



Obrázek 6.15: Scéna 4: rasterizace, RTRT, RTRT pohled zhora

6.3.10 Scéna 5 RA

Na dolních krychlích dochází ke konfliktu reflection probů a screen space reflection. Rasterizace neumí v některých případech správně detekovat, který objekt je blíže ke kameře. K této chybě nejčastěji dochází pokud jsou dva relativně tenké objekty z pohledu hloubky vzhledem ke kameře umístěny blízko za sebou v částečném zákrytu. Tato pak v některých úhlech pohledu způsobí, že je objekt, který je částečně zakrytý druhým objektem vzhledem ke kameře vykreslen před objektem, který ho zakrývá.



Obrázek 6.16: Scéna 5 rasterizace

6.4 Uživatelské testy

Dotazník pro uživatelské testování aplikace obsahuje 17 otázek týkajících se scén 1 až 5 a prostor pro vyjádření se k příslušným scénám. Testy se soustředí čistě na hodnocení vzhledové stránky, primárním důvodem tohoto modelu uživatelských testů je relativně malá dostupnost různých sestav kompatibilního hardwaru. Uživatelské testy byly provedeny třemi různými uživateli, přičemž test prvním a druhým uživatelem byl proveden na testovací sestavě 2 a test třetím uživatelem přes vzdálené sdílení obrazovky na testovácí sestavě číslo 2.

Real time ray tracing - uživatelské testování

Tento dokument je dotazník pro účely uživatelského testování aplikace vytvořené v Unity, která je součástí bakalářské práce. Dotazník se primárně zaměřuje na vzhledovou stránku hodnocení ray tracingu v Unity a s ním souvisejících funkcí.
Otázky jsou vždy přiřazeny ke konkrétní scéně v aplikaci. K jejich zodpovězení je ideální volný průchod aplikací ve scénách 2 a 3, u zbylých scén jsou dostačující fixní kamery.
Hodnotící stupnice je definována jako porovnávání ray tracingové verze scén oproti rasterizační(vypnutý ray tracing).
Jednotlivé scény jsou dostupné z hlavního menu testovací aplikace. Pro přechod mezi ray tracing verzí a rasterizační je vždy nutné vrátit se do hlavního menu a přepnout na alternativní verzi. Toto je způsobeno způsobem implementace ray tracingu v Unity.

Hodnocení(RTRT verze oproti rasterizaci):
Kvalita:
1/výrazně horší 2/horší 3/přibližně stejná 4/lepší 5/výrazně lepší

Testovací sestava
Procesor:
Grafická karta:
Paměť:

Scéna 1

1.1. Porovnejte jak které stíny stromů u jezírka se vám líbí více

1.V. Prostor pro vyjádření se ke scéně 1

Scéna 2

2.1. Jak hodnotíte rtrt verzi zahrady oproti standardní verzi.

2.2. Porovnejte scénu z pohledu kvality osvětlení

2.3. Porovnejte scénu z pohledu kvality stínů

2.4. Porovnejte scénu z pohledu kvality odrazů

2.5. Porovnejte scénu z pohledu celkového dojmu

2.6. Porovnejte která zrcadlová síň se vám zdá realističtější?

2.7. Porovnejte která kuchyň je lépe osvětlená.

2.8. Porovnejte která verze zahrady má lepší stíny?

2.V. Prostor pro vyjádření se ke scéně 2

Scéna 3

3.1. Porovnejte scénu z hlediska kvality stínů

3.2. Porovnejte kvalitu odrazů v oknech budovy s posezením

3.3. Porovnejte kvalitu odrazů mezi mrakodrapy

3.V. Prostor pro vyjádření se ke scéně 3

Scéna 4

4.1. Porovnejte scénu z pohledu kvality odrazů na pozadí a krychli

4.2. Porovnejte scénu z pohledu kvality lomu světla v čočkách

4.V. Prostor pro vyjádření se ke scéně 4

Scéna 5

5.1. Porovnejte scénu z pohledu kvality stínů

5.2. Porovnejte scénu z pohledu kvality odrazů na pozadí

5.3. Porovnejte scénu z pohledu kvality odrazů mezi krychlemi

5.V. Prostor pro vyjádření se ke scéně 5

Následující seznam obsahuje odpovědi uživatelů na jednotlivé otázky, volné otázky nebyly žádným z uživatelů zodpovězeny.

- 1.1. 1, 3, 4
- 2.1. 4, 3, 3
- 2.2. 5, 5, 5
- 2.3. 4, 4, 4
- 2.4. 4, 4, 5
- 2.5. 4, 4, 3
- 2.6. 4, 4, 5
- 2.7. 5, 4, 5
- 2.8. 4, 3, 4
- 3.1. 4, 3, 4
- 3.2. 1, 2, 3
- 3.3. 3, 4, 4
- 4.1. 4, 4, 5
- 4.2. 5, 5, 5
- 5.1. 3, 3, 4
- 5.2. 5, 5, 5
- 5.3. 5, 4, 5

Zhodnocení výsledků testu

Z celkového pohledu vyplývá, že uživatelé, kteří aplikaci testovali, hodnotí Unity real time ray tracing jako spíše přínosný až na některé konkrétní případy. Mezi tyto případy patří stíny stromů ve scéně 1 jejichž hodnocení je sporné a kvalita odrazu v oknech budovy s posezením, kde je hodnocení spíše negativní, což je pravděpodobně v tomto případě způsobeno tím, že v RTRT verzi scény neodráží tato budova terén.

6.5 Souhrné výsledky

Tato sekce obsahuje výsledky dosežené shrnutím informací z více různých testů a dalších informací získaných během práce na demonstrační aplikaci.

6.5.1 stíny

Terén je do určité míry schopen detekovat zda se k jeho části dostane světlo nebo je zastíněno jinou částí terénu, díky čemuž je terén natočený proti světlu zastínění i když jsou vypnuté stíny. Mapy stínů mají bez využití screen space shadows rozmazané okraje a vyskytují se v nich defekty, přičemž screen space shadows pouze zpřesňují okraje stínů. Ray traced shadows při rychlém pohybu kamery mohou být krátkou chvíli vidět v některých případech na předchozím umístění stínů před pohybem kamery, okraje těchto stínů většinou zrní. RT stíny nemají defekty které se projevují u map stínů a SSS, nedochází však u nich k zastínění terénem.

6.5.2 Metody vyuzívané rasterizací vs RTRT

Ray tracing má oproti metodám rasterizace jak výhody tak nevýhody. Statické metody rasterizace jako jsou lightmaps a baked reflection probes neumějí reagovat na dynamické scény. Rasterizace z tohoto důvodu nemá k dispozici kvalitní real-time řešení globálního osvětlení Dynamické prvky jako jsou například reflection probes a podobně, pokrývají pouze menší část grafických efektů, jako jsou například odrazy světla. Většinou se jedná o zjednodušené řešení, takže neposkytují zcela realistický výsledek ale pouze jeho zdání. Na menších scénách jsou efektivnější než rt, avšak na rozsáhlejších scéna u nich dochází k mnoha problémům. Nejzásadnějším problémem je neschopnost těchto prvků zjistit, zda jsou relevantní pro současně vykreslovaný snímek. Toto vede k nutnosti, buď manuální režie těchto prvků pomocí skriptů nebo růstu náročnosti vykreslení scény vzhledem k míře využití těchto prvků. Screen space metody poskytují v současné chvíli lepší grafickou kvalitu při nižší spotřebě výkonu, neumožňují však pracovat s gameobjecty vyskytujícími se mimo záběr kamery a to znemožňuje odstranění některých grafických artefaktů jako je třeba světlo prosvítající skrz zdi. RTRT metody nevytváří většinu grafických artefaktů, které jsou vytvářeny předchozími metodami. Mají relativně nízký růst náročnosti výpočtu vzhledem k velikosti scény, avšak růst náročnosti vykreslení naopak většinou roste výrazně s počtem

světel na scéně. Velkou limitací je v současnosti neschopnost práce s terénem, což se týká nejenom terénu samotného, ale také objektů na něj umístěných pomocí nástrojů pro editaci terénu. Toto se dotýká zejména vegetace, protože unity není korektně schopné pracovat s příliš velkým množstvím gameojects a pokud je každý strom, květina a trs trávy jeden gameobject, dochází k výrazným dopadům na výkon. V některých interiérových scénách může docházet k problémům, obzvláště pokud se na nich vyskytuje velké množství odrazivých objektů, v těchto případech dochází k velkému výskytu zrnění. Dalším problémem jsou stíny komplikovaných objektů s mnoha tenkými částmi jako jsou například stromy, u těchto objektů dochází ke vzniku výrazně tenčích stínů něž jaké by měly být. Zásadním problémem je také špatně definované chování lomu světla pro nekulové objekty. RTRT je nevhodné použít pro exteriérové scény typu lesa, protože stromy umístěné pomocí funkce place trees jsou chápány jako součást terénu, takže jsou rtrt zcela ignorovány což se výrazně projeví pokud jsou zapnuty raytraced shadows tak stromy nebudou mít stíny. Tuto situaci není možné nijak vyřešit protože jiné způsoby vytvoření lesa, buď vytvoří extrémně velké množství game objectů, což má velmi špatný vliv na výkon aplikace nebo naopak menší množství extrémně komplikovaných objektů(část lesa jako jeden objekt), což naopak způsobí problémy s například collidery, interakcí a položení takovýchto objektů na nerovný terén je téměř nemožné. Problémy s výkonem aplikace jsou z části způsobeny Unity HDRP enginem, který z neznámého důvodu neumí plně využít hardwarovou sestavu

Kapitola 7

Závěr

Scény vykreslené pomocí Unity real time ray tracingu vykazují celkové zlepšení oproti scénám vykresleným rasterizací s několika výjimkami. Při využití hardwarové sestavy obsahující grafickou kartu s přímou hardwarovou podporou ray tracingu jsou navíc mnohé z ray tracingových efektů méně náročné na výkon než jejich alternativy. Přesto však má testovaná implementace ray tracingu zásadní nedostatky, které omezují její možnosti využití. Mezi nejzásadnější z těchto problémů patří neschopnost RTRT vykreslovat stíny a odrazy terénu a vizuálních efektů, některé defekty objevující se u odrazu a lomu světla a také řada dalších drobnějších problémů. Argumentem napomáhajícím ray tracingu je fakt, že i u alternativ k RTRT efektům se vyskytují grafické defekty. Z těchto důvodů je možné Unity real time ray tracing doporučit pro vykreslování interiérových scén neobsahujících výše uvedené limitace naopak pro exteriérové scény, které bývají z významné části založené na terénu, je současná implementace ray tracingu v Unity zcela nevhodná.

Příloha A

Návod

Hlavní menu

- Vypnout program: *
- Zapnout a nebo vypnout RT: +
- Spouštění scén = číslo scény
- Rozlišení 1920x1080: m
- Rozlišení 1280x720: n

Scéna 1

- Aktivace/deaktivace skupin stromů: O/P

Scény 1 a 2

- Změna času (rotace směrového světla): K(+10) a L(-10)

Scény 1-5

- Pohyb vpřed: W
- Pohyb vzad: S
- Pohyb doleva: A
- Pohyb doprava: D
- Zrychlení pohybu (10x): Left Shift
- Přepínání letového módu: F
- Let nahoru: Space
- Let dolů: Left Ctrl
- Statické kamery: 1, 2, 3
- Kamera hráče: 0

Všechny RTRT scény

- Přepínání globální iluminace: G

Všechny Scény

- Zapnutí/vypnutí kurzoru: i
- Návrat do Menu: Esc
- Rozlišení 1920x1080: m
- Rozlišení 1280x720: n

Příloha B

Literatura

[AMDa] Inc Advanced Micro Devices, *Amd ryzen 7 1700x processor product information.*

[AMDb] ———, *Amd ryzen 7 3700x processor product information.*

[App68] Arthur Appel, *Some techniques for shading machine rendering of solids*, AFIPS 1968 Spring Joint Comptr. Conf, 1968.

[Cora] NVIDIA Corporation, *Nvidia geforce gtx 1080 product information.*

[Corb] ———, *Nvidia geforce rtx 3070 product information.*

[ea19] Haines et al., *Ray tracing gems*, Apress, 2019.

[IW03] Jörg Schmittler Carsten Benthin Philipp Slusallek Ingo Wald, Timothy J.Purcell, *Realtime ray tracing and its use for interactive global illumination*, In Eurographics State of the Art Reports, 2003.

[Kaj86] James T. Kajiya, *The rendering equation*, Computer Graphics, 1986, pp. 143–150.

[Pin88] Juan Pineda, *A parallel algorithm for polygon rasterization*, In Proceedings of Siggraph '88, 1988, pp. 17–20.

[She] Han-Wei Shen, *The opengl rendering pipeline.*

[SP99] Peter-Pike J. Sloan Peter Shirley Brian Smits Charles Hansen Steven Parker, William Martin, *Interactive ray tracing*, In Symposium on interactive 3D graphics, 1999, pp. 119–126.

[Tec] Unity Technologies, *Hdrp 7.3 package documetation.*

[TR09] Hans-Peter Seidel Tobias Ritschel, Thorsten Grosch, *Approximating dynamic global illumination in image space*, IN PROC. I3D, 2009, p. 75–82.

[Whi80] Turner Whitted, *An improved illumination model for shaded display*, Communications of the ACM **23** (1980), 343–349.

[Wil78] Lance Williams, *Casting curved shadows on curved surfaces*, In Computer Graphics (SIGGRAPH '78 Proceedings, 1978, pp. 270–274.

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Bednář** Jméno: **Bohumil** Osobní číslo: **437309**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačové grafiky a interakce**

Studijní program: **Otevřená informatika**

Studijní obor: **Počítačové hry a grafika**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Sledování paprsku v réalném čase v Unity

Název bakalářské práce anglicky:

Real-Time Ray Tracing in Unity

Pokyny pro vypracování:

Zmapujte dostupná rozhraní pro zobrazování pomocí metody sledování paprsků v reálném čase. Nastudujte a popište možnosti efektů dosažitelných pomocí metod sledování paprsku v reálném čase dostupných v Unity.
Vytvořte technologickou demonstrační aplikaci využívající sledování paprsků v Unity a proveďte důkladné testy kvality a rychlosti zobrazování. Součástí demonstrační aplikace bude hratelné demo, jehož cílem je předvést schopnosti a omezení sledování paprsku v realném čase v herním nasazení.
Vyhodnoťte limity implementace z hlediska velikosti scény, možnosti jejich dynamických změn a složitosti zobrazovaných efektů. Proveďte základní uživatelský test vytvořeného dema, který vyhodnotí subjektivní vnímání efektů simulovaných sledováním paprsků ve srovnání se standardním zobrazovacím řetězcem.

Seznam doporučené literatury:

[1] Tomas Akenine-Moller et al. Real-Time Rendering (4th edition). CRC Press, 2018.
[2] Haines et al. Ray Tracing Gems, Apress, 2019.
[3] Real-Time Ray Tracing in Unity. https://unity.com/ray-tracing

Jméno a pracoviště vedoucí(ho) bakalářské práce:

doc. Ing. Jiří Bittner, Ph.D., Katedra počítačové grafiky a interakce

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **11.02.2020** Termín odevzdání bakalářské práce: _____

Platnost zadání bakalářské práce: **30.09.2021**

doc. Ing. Jiří Bittner, Ph.D.		prof. Mgr. Petr Páta, Ph.D.
podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

. _____ _____

Datum převzetí zadání Podpis studenta