Sem vložte zadání Vaší práce.

České vysoké učení technické v Praze
Fakulta elektrotechnická
Katedra otevřené informatiky

Diplomová práce

Interaktivní sférické video ve virtuální realitě

Bc. Mikoláš Zuza

Vedoucí práce: Ing. David Sedláček Ph.D.

24. května 2017

Poděkování

Děkuji vedoucímu Ing. Davidovi Sedláčkovi Ph.D. za odborné vedení mé práce, Ing. Janu Buriánkovi za odborné konzultace a zapůjčení technického vybavení a své rodině za podporu během celého studia.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) 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 24. května 2017 .....

České vysoké učení technické v Praze
Fakulta elektrotechnická

Odkaz na tuto práci

Zuza, Mikoláš. *Interaktivní sférické video ve virtuální realitě.* Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta elektrotechnická, 2017.

Abstrakt

Práce se zabývá systémy pro záznam a přehrávání sférických videí a postupem tvorby interaktivního stereoskopického sférického obsahu pro HMD. Po provedení rešerše softwaru pro rekonstrukci geometrie scény a trajektorie kamery z videa, jsou popsané postupy otestované na zkušební sférické videosekvenci. Tato data jsou načtena do herního enginu Unity, kde jsou synchronně přehrávána spolu s videem. Výstupem práce je aplikace pro virtuální realitu, umožňující interagovat s objekty ve sférickém videu.

Klíčová slova Sférické video, rekonstrukce, virtuální realita, Unity.

Abstract

This thesis describes the process of recording and playing spherical videos and the process of creating interactive stereoscopic spherical content for HMD. After the research of software used for scene geometry and camera trajectory reconstruction from video the described processes are tested on a short spherical video. The reconstruction data are loaded into Unity game engine, where they're synchronously played alongside with the video. The result of this work is an application for virtual reality devices, which allows you to interact with

objects in the spherical video and proves the functionality of described procedures.

Keywords Spherical video, reconstruction, virtual reality, Unity

Obsah

Seznam obrázků

Úvod

Významný pokrok v oblasti virtuální a rozšířené reality vytvořil poptávku po obsahu, který je možný v ní zobrazovat. Pokud je obsah vytvořený v reálném světě, měl by nějakým způsobem zachycovat úhel záběru 360 stupňů tak, aby si uživatel sám mohl vybrat oblast zájmu. Poptávku po podobném obsahu vytvářejí i nová digitální planetária v České republice i ve světě. Přehrávání tohoto obsahu je převážně pasivní, uživatel pouze volí směr pohledu. Na druhou stranu pro virtuální realitu vznikají i interaktivní aplikace a hry. Ty se naopak odehrávají zcela v počítačem vykresleném prostředí.

Mezi plně virtuálním interaktivním světem a reálně zachyceném sférickém, ale pasivně zobrazeném obsahu, zatím neexistuje most. Propojení těchto dvou technologií a vytvoření tzv. rozšířené reality se začíná dařit v oblasti mobilních her s využitím externích senzorů hloubky a masivně paralelizovaném výpočtu rekonstrukce prostředí.

Tato práce se zabývá podobným propojením dvou prozatím odděleně vnímaných technologií a vytvořením interaktivního obsahu ve videu natočeném stereoskopicky všesměrově v reálném světě.

Kapitola 1

Sférická fotografie a video

Sférická fotografie je formou panoramatického snímku, ve kterém se snímají z jednoho místa všechny směry najednou. Všesměrový obraz je snímán buď sadou několika pevně spojených kamer nebo specializovanou kamerou s více objektivy. Výsledný snímek vzniká spojením jednotlivých obrazů v procesu zvaném stitching, který kamera buď provádí sama při natáčení nebo je nutné ho provést v softwaru pro tvorbu panoramat. Sférický snímek je možné pořídit i pomocí staršího systému s jednou kamerou a dvojicí zrcadel. Tento přístup se dnes používá spíše výjimečně, mezi jeho nevýhody patří nedostatečný úhel záběru, zkreslení obrazu zrcadly a nízké rozlišení.

1.1 Formát snímku

Snímek je uložen v ekvidistantní válcové projekci. Toto zobrazení promítá poledníky koule na rovnoběžné vertikální úsečky s konstantním rozestupem a rovnoběžky na horizontální úsečky s konstantním rozestupem. Zobrazení není ani plochojevná projekce ani konformní zobrazení, nezachovává tak úhly ani obsah ploch. Poměr stran snímku je 2:1. Zobrazení je prováděno podle předpisu:

$$x = \lambda \; cos(\phi_1)$$
$$y = \phi$$

Kde λ je zeměpisná délka, ϕ je zeměpisná šířka a ϕ_1 je skutečný rozsah šířky, který u sférických kamer bude určovat velikost slepého místa pod kamerou a přímo nad kamerou.

1.2 Přehrávání sférického videa

Při přehrávání není uživateli typicky snímek zobrazen přímo v ekvidistantní válcové projekci. Místo toho je aplikováno inverzní zobrazení a pomocí klasické perspektivní kamery je zobrazena pouze část sférického snímku. Při přehrávání na mobilních zařízeních se navíc pomocí vnitřních senzorů, jakým je například akcelerometr a gyroskop, přenáší orientace mobilu v prostoru na orientaci virtuální kamery. Na počítači je nejčastější ovládání kamery pomocí držení levého tlačítka myší a tažení v okně přehrávače. Další typ zařízení vhodný pro přehrávání sférického videa je HMD, jakým je například Oculus Rift nebo HTC Vive, které kromě vnitřních senzorů využívají k přesnému určení orientace a polohy kamery i externí laserové senzory a díky dvěma čočkám a displeji umístěném blízko k očím mají široké zorné pole (100 stupňů pro HTC Vive a 80 stupňů pro Oculus Rift).

1.3 Percepce rozlišení

Na rozdíl od klasického perspektivního snímku není percepce rozlišení a skutečné rozlišení snímku stejné. Uživatel se vždy dívá jen do části sférické projekce a má tak sníženou percepci rozlišení. Pro percepci odpovídající FullHD je nutné minimálně šestinásobné rozlišení snímku. Minimální rozlišení videa by proto mělo být alespoň 4096×2048, ideálně však 8192×4096, což odpovídá 33.5 Mpix, podobně jako rozlišení označené 8K. Pokud by měl být snímek stereoskopický, rozlišení respektive datový tok bude dvojnásobný[1].

Obrázek 1.1: Kamera Nokia OZO schopná natáčet v rozlišení 8K, převzato z [1]

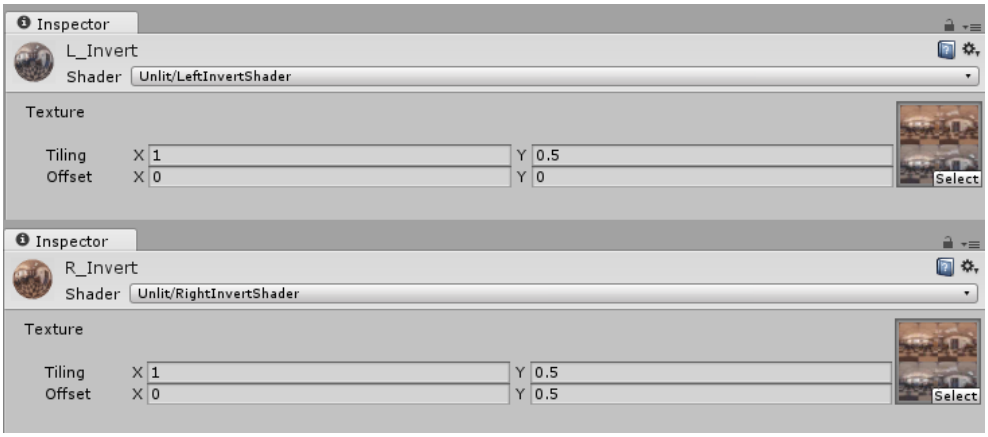
Kapitola 2

Zobrazení sférického obrazu v Unity

Pro počáteční testování bez přístupu k HMD typu HTC Vive byla zvolena VR helma VR BOX 2.0, která pro zobrazování využívá mobilní telefon se systémem Android, a knihovna Google VR SDK [4] umožňující kompilovat aplikace pro Android s podporou tohoto zařízení.

Obrázek 2.1: VR BOX 2.0 + mobilní telefon Mi2s

Systém zobrazování spočívá v mapování textury na sféru, ve středu sféry je umístěna kamera. Rozšíření podporující stereoskopické zobrazení do scény přidává druhou dvojici sféry a kamery, vzdálenost mezi kamerami odpovídá interokulární vzdálenosti v měřítku scény. Byl implementován shader, zobrazující pouze vnitřní stěny objektu. Dále byly vytvořeny 2 materiály, které ze stejné textury extrahují obraz pouze pro jedno, pravé či levé, oko. Pro extrakci samostatných obrazových dat pro pravé a levé oko ze společné textury bylo elegantně využito nastavení Tiling a Offset. Díky němu je možné velice



Obrázek 2.2: Nastavení tiling a offset textury v Unity

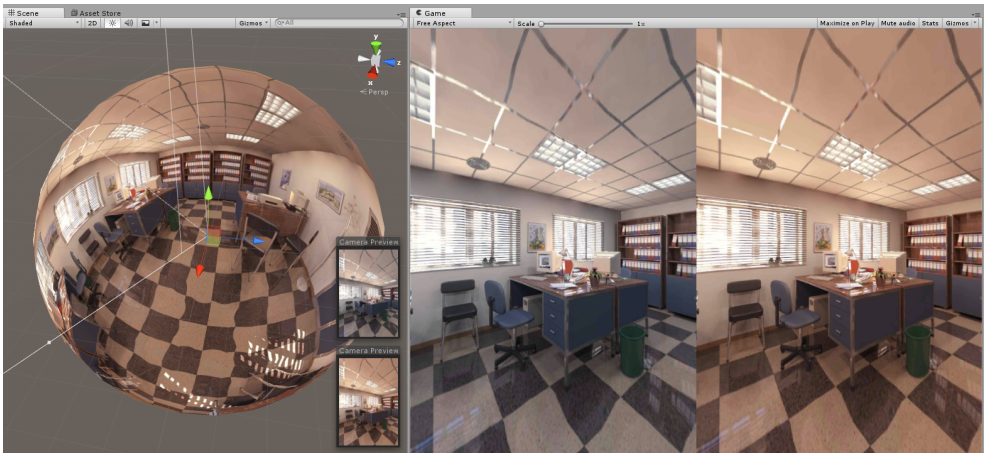
rychle změnit nastavení stereoskopického uspořádání obrazů z over-under na side-by-side, případně opravit prohozený levý a pravý kanál.

Pro testování byl vytvořen stereoskopický sférický snímek v rozložení over-under z původně monoskopického snímku, pro snazší ladění byl barevně odlišen kanál pro pravé a levé oko, díky tomu uživatel ihned pozná, který kanál sleduje.

Obrázek 2.3: Testovací stereo-sférický snímek

2.1 Proof of concept zobrazení stereoskopického sférického snímku

Sférám ve scéně byly přiřazeny odpovídající materiály a nastaveny culling vrstvy tak, aby pravá kamera ignorovala levou sféru a naopak. Do scény byla dále vložena krychle viditelná pro obě kamery. Při spuštění enginu je obraz na sférické textuře nerozeznatelný od krychle vytvořené v reálném čase, prototyp je tedy funkční.



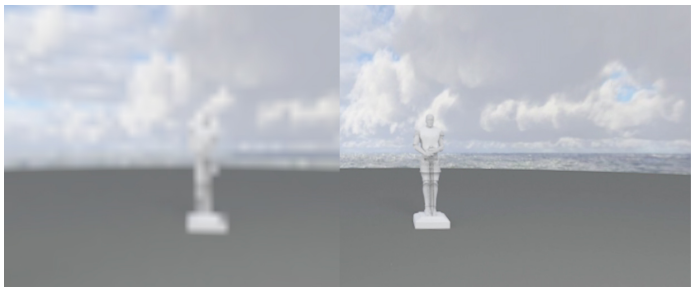
Obrázek 2.4: Náhled proof of concept uspořádání scény

Kapitola 3

Přehrávání sférického videa v Unity

Během psaní této práce se možnosti přehrávání videa v Unity [14] rychle měnily. Na Unity 2016 Keynote, které se konalo 1.11.2016, byla představena výrazně zlepšená podpora pro VR a to i v podobě úplně nového Video Playeru a VR nástrojů pro tvorbu scény s nasazenou virtuální helmou. Současný přehrávač videa do textury se ukázal být zcela nepoužitelný a to hlavně pro extrémní kompresi video souboru a nízkou obnovovací frekvenci.

Beta verze Unity 5.6 měla vyjít během měsíce listopadu 2016 (vyšla 13. prosince), nicméně nebylo možné na tuto aktualizaci čekat a zastavit práci na Diplomové práci. Proto byla implementována jednoduchá video textura. Video bylo nejprve exportováno jako sekvence obrázků ve formátu jpg, v metodě FixedUpdate herního objektu byla poté periodicky měněna textura za následující snímek. Tento přístup má několik nevýhod. Nejenže kvůli tomu data videa zabírají mnoho místa, protože neprobíhá žádná mezisnímková komprese, ale Unity neumí tak velké obrázky dostatečně rychle načítat. Video se přehrávalo s nízkou snímkovou frekvencí nebo musela být drasticky zmenšená velikost snímků a tedy výsledné rozlišení videa.



Obrázek 3.1: Srovnání rozlišení videa: vyměňování jednotlivých snímků (vlevo) nová Unity video textura (vpravo)

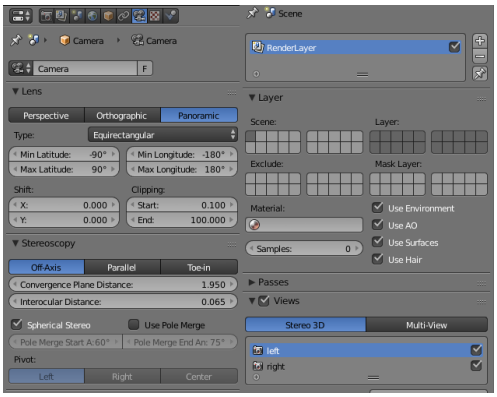
Poté co vyšla Beta 5.6 se přes chybějící dokumentaci nové funkcionality podařilo stylem pokus-omyl novou komponentu video textury zprovoznit. Video je možné pustit přímo ze souboru ve formátu mp4. Video po kompresi kodekem h264 je mnohem menší než ukládání jednotlivých snímků a navíc Unity s novou video texturou umí Material Override, ale i RenderToTexture, které stíhají bez problému video plynule přehrávat pro obě oči najednou. Pro extrakci jednotlivých obrazových kanálu pro pravé a levé oko byl použit stejný trik s nastavením tiling a offset textury jako v případě stereoskopického sférického snímku. Díky tomu stačí číst jediný video soubor ve formátu over-under nebo side-by-side. Ve vertex shaderu byl upraven výstup uv souřadnic, protože video se kvůli promítání na sféru přehrávalo zrcadlené horizontálně.

Kapitola 4

Tvorba sférického stereoskopického videa v Blenderu

Přestože cílem práce je propojit virtuální 3D objekty se sférickým videem natočeným v reálném světě, zejména pro prvotní testování bylo nutné vytvořit sférické video v programu pro 3D modelování. Pro tento účel byl zvolen Blender [13], obsahuje veškerou potřebnou funkcionalitu a je dostupný zdarma, dokonce je licencován jako Open-source.

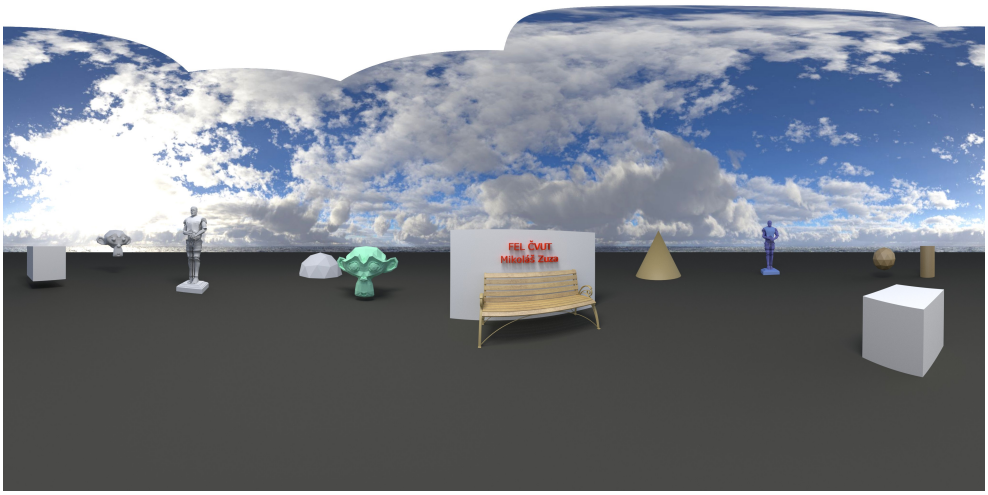
Pro vytvoření sférického stereoskopického snímku je nutné přepnout interní Blender render engine na renderer Cycles. Vhodné je povolit renderování na GPU, umožňuje-li to hardware. Na kartě Views je nutné zapnout Stereo 3D, samotná kamera se musí přepnout do módu Panoramic Equirectangular a nastavit úhlové rozsahy. Zároveň je potřeba zapnout Spherical Stereo, které dělá korektní rotaci dvojice kamer okolo středu jejich vzájemné spojnice, rotace okolo jejich vlastní osy by vedla k nesprávným výsledkům.



Obrázek 4.1: Nastavení sférické kamery v Blenderu

Při nastavování výstupu je nutné vybrat Stereo 3D konfiguraci. Na výběr je Top-Bottom, Side-by-Side nebo export do samostatných souborů pro pravé a levé oko. Pro potřeby trackování je optimální jako výstup zvolit dva soubory pro každý snímek (left, right). Video v konfiguraci Over-Under poté vytvořit spojením těchto snímků například v programu Adobe Premiere Pro.

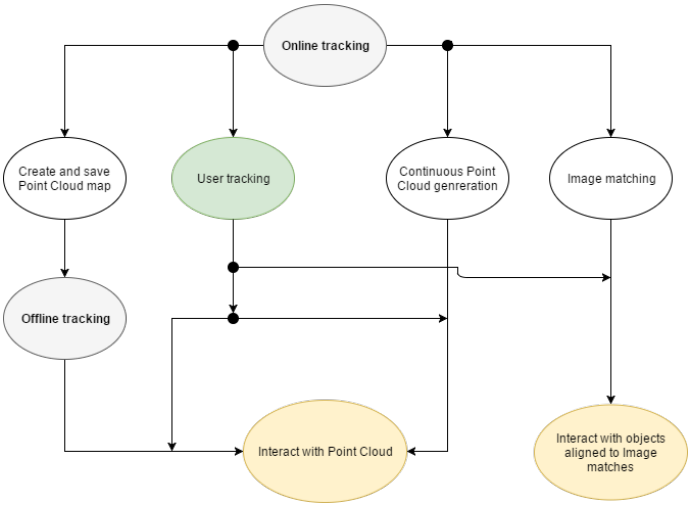
Obrázek 4.2: Nastavení exportu snímku v Blenderu



Obrázek 4.3: Snímek v ekvidistantní válcové projekci

Kapitola 5

Rešerše trackování sférického videa



Obrázek 5.1: Diagram možností trackování sférického videa

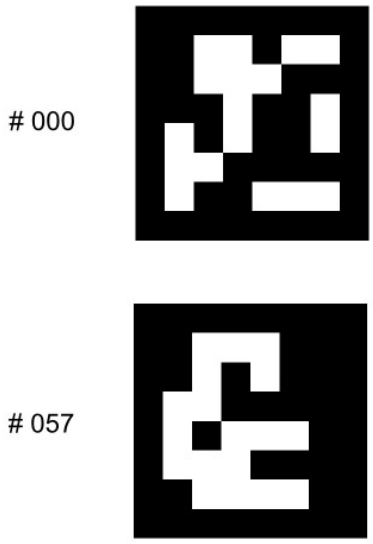
5.1 Online tracking

5.1.1 Image matching

Nejjednodušší způsob jak podle obrazových dat umístit do scény 3D objekt je pomocí rozpoznání předem definovaných obrazců. Tyto obrazce musí být kontrastní a nesymetrické. Každý obrazec má svůj identifikátor. Při zpracování snímku se hledá poloha těchto obrazců a na základě jejich vzájemné polohy a

natočení se v obrazu jednoznačně určí úhel natočení a vzdálenost kamery od obrazců[2].

Zásadní nevýhodou je nutnost umístit tyto obrázky do prostoru před pořízením videozáznamu. Navíc nesmí dojít k zákrytu žádné části obrazce. Problémem může být dokonce i to, když na obrazec dopadne hranice stínu vrženého nějakým objektem ve scéně. Obrazce jsou detekovatelné jen do určité vzdálenosti a pokud chceme tuto vzdálenost zvětšit, musíme zvětšit i velikost obrazců. Na druhou stranu je možné tracking provádět v reálném čase. Jednou z knihoven, která umisťuje objekty do scény na základě image matchingu, je Vuforia SDK.



Obrázek 5.2: Dva obrazce a jejich identifikátory pro image matching

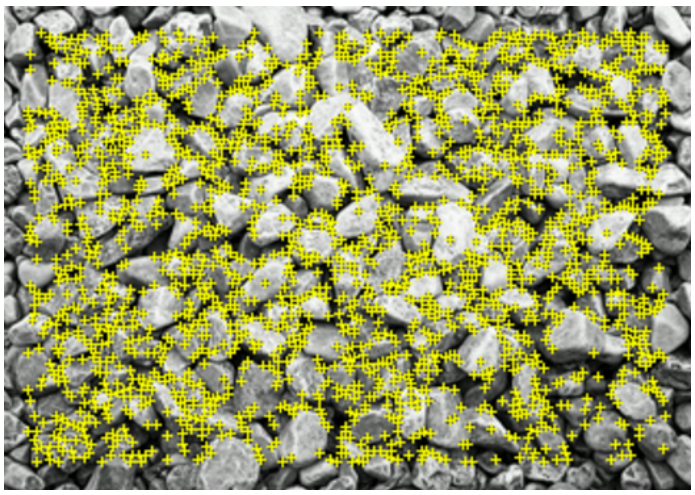
5.1.1.1 Vuforia SDK

Vuforia[3] je populární vývojová knihovna pro rozšířenou realitu, určená hlavně pro mobilní zařízení. Pro vývoj je navíc dostupná zdarma a to i ve formě Unity Asset package.

Do scény v Unity stačí vložit daný obrazec, který chceme sledovat, a jako potomka v hierarchickém uspořádání scény vložit objekt, který chceme na pozici obrazce umístit. Vzájemná poloha obrazce a 3D objektu bude při rekonstrukci zachována. Vuforia pro obrazový vstup v SDK očekává připojenou web kameru či obraz z kamery mobilního zařízení. Nicméně v kódu SDK stačí

přepsat řádek, který načítá obraz z web kamery, a nahradit ho vlastním výstupem Unity kamery do textury. Abychom ale mohli obrazce trackovat, je nutné se zaregistrovat a ve webové aplikaci vytvořit databázi obrazců pro tracking. Tuto databázi lze následně stáhnout a importovat do Unity. Při importu databáze však dojde k chybě. Odpověď vývojářů Vuforia radí použít Unity ve 32-bit verzi. Unity bohužel v listopadu 2016 ukončilo podporu 32-bit editoru a beta Unity 5.6, kterou je nutné použít pro přehrávání video textury, v této verzi není k dispozici.

Vuforia SDK tedy nebylo možné v této práci použít.



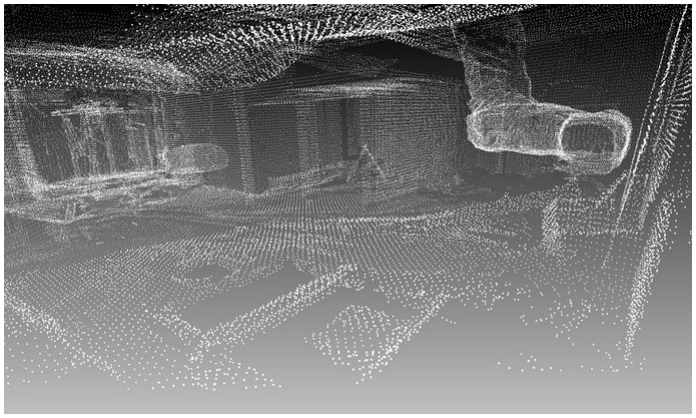
Obrázek 5.3: Obrazec pro určení polohy ve scéně a význačné body pro jeho analýzu

5.1.2 Generování point clouds

Na rozdíl od trackování pouze několika obrázků ve scéně se metoda používající Point clouds snaží rekonstruovat celý prostor, ve kterém se hráč pohybuje. Během rekonstrukce se vytváří velké množství trojrozměrných souřadnic bodů v prostoru pomocí hloubkového senzoru, dat z akcelerometru, gyroskopu a pomocí prostorového učení, které je schopné napojovat i nespojitá data[8]. Je pak pouze na nás, jak tyto body využijeme. Jednou z prvních knihoven, která umí point clouds generovat v reálném čase je Google Project Tango.

5.1.2.1 Google Project Tango

Knihovna, o které Google mluví již několik let a objevila se na mnohých živých prezentacích, působí téměř neuvěřitelně. I na mobilním zařízení je schopná rekonstruovat v reálném čase geometrii prostoru a následně například na ploše stolu zobrazovat interaktivní 3D objekty.



Obrázek 5.4: Point cloud reprezentující analýzu bytu, převzato z [5]

Obrázek 5.5: Optika Asus Zenfone AR, převzato z [6]

Podobně jako Vuforia je dostupná i jako Unity Asset Package. Do scény je potřeba vložit jednu kameru navíc, Tango Camera Prefab, která se stará o generování bodů. Dále několik dalších prefabů pro správu bodů a inicializaci knihovny. Při testování se však nedařilo SDK donutit generovat jakékoliv body. Po dlouhém testování a hledání vyšlo najevo, že oficiálně knihovna podporuje pouze 2 mobilní zařízení, Asus Zenfone AR a Lenovo Phab 2 Pro. Tato zařízení mají totiž kromě kamery ještě druhý obrazový snímač a dvojici infra senzorů. V kombinaci s IMU dokáže knihovna generovat point cloud, bez nich ale nedojde jen k snížení přesnosti, knihovna není schopná generovat jakékoliv souřadnice bodů. Pro účely této práce tedy knihovnu Tango[7] použít nelze.

5.2 Offline tracking

5.2.1 Kudan SDK

Kudan SDK[9] vytváří point cloud podobně jako Google Tango, data však vytváří v odděleném procesu jehož výstupem je množina bodů pro pozdější

použití. Při samotné realtime interakci uživatele tedy již žádné body nevznikají a pouze se určuje poloha k již existujícím bodům. Na rozdíl od knihovny Vuforia se nepodařilo nalézt část kódu, která načítá obraz z reálné kamery a podvrhnout jí vlastní texturou, do které by vykreslovala Unity kamera.

Po kontaktování vývojářů této knihovny bylo potvrzeno, že změna by se musela provést v samotném jádru knihovny, které je za normálních okolností přikládáno již zkompilované. Vývojáři navíc upozornili na fakt, že podobně jako knihovna Tango musí být na vstupu kromě obrazových dat i IMU a pokud by data z akcelerometru, gyroskopu a magnetometru nebyla přiložena na vstup, knihovna nebude fungovat. Nicméně všechny tyto senzory HTC Vive a další VR helmy mají. Je možné k nim získat přístup pomocí příkazové řádky ve složce SteamVR/tools:

```
lighthouse_console.exe /imu /dump
```

Do příkazové řádky se přesměruje výstup raw dat z akcelerometru a gyroskopu. Bohužel se opravdu přesměrují a Unity ztratí možnost k nim přistupovat, dokud se Unity nerestartuje a v ten moment se naopak odřízne přístup příkazové řádce. Využití této knihovny tedy také vytváří řadu překážek, které nelze snadno vyřešit.

Protože se nepodařilo použít ani jednu ze 3 největších a nejpoužívanějších knihoven pro Unity, byl pro trackování zvolen jiný postup využívající externí program k offline rekonstrukci scény i trajektorie kamery a zároveň vlastní kód pro zpracování těchto dat a načtení do Unity.

5.2.2 Voodoo Camera Tracker

Tento software pro nekomerční použití vznikl na Univerzitě Hannover. Voodoo Camera Tracker[10] odhaduje parametry kamery a rekonstruuje 3D scénu ze sekvence obrázků. Algoritmus rekonstrukce je plně automatický a jeho výsledek je možné exportovat do programů Autodesk Maya, Blender, 3D Studio Max a další. Při testování na krátkých sekvencích vracel program velmi kvalitní rekonstrukce trajektorie kamery a bodů ve scéně. Při delších sekvencích však algoritmus končil chybou, protože nebyl u některých snímků schopen s již zkalibrovanou kamerou vypočítat její polohu ve scéně. Na sekvencích delších než minuta chybou končil každý pokus o rekonstrukci a proto nebylo možné Voodoo Camera Tracker použít pro potřeby této práce.

5.2.3 Adobe After Effects

Adobe After Effects je software of firmy Adobe Systems pro tvorbu filmových efektů, animaci a kompozici. Součástí softwaru je 3D camera tracker, který analyzuje pohyb kontrastních skupin pixelů a vytváří jejich trojrozměrnou reprezentaci. Výsledkem rekonstrukce je zkalibrovaná kamera, klíčové snímky pro její polohu, orientaci a řídký point cloud. Body nevhodné pro tracking

(například části pohyblivých objektů ve scéně) je možné ručně smazat a zlepšit tak výslednou rekonstrukci.

After Effects neumí rekonstrukci exportovat. Při označení klíčových snímků v časové ose a stisknutí kombinace kláves CTRL+C pro kopírování se ale data ukládají do schránky v textové podobě. Zkopírováním všech klíčových snímků polohy a orientace kamery do textového souboru tak nahrazuje chybějící funkci exportu.

5.2.4 Visual SFM

Visual SFM[11] je aplikace s grafickým rozhraním pro Structure From Motion. Na vstupu aplikace načte sekvenci obrazů. Mezi jednotlivými snímky se poté vyhledávají podobné fragmenty obrazových dat. Pokud je program schopný rozpoznat dostatečný počet těchto podobných prvků, může vytvořit Řídkou rekonstrukci (Sparse reconstruction). Ta se snaží do 3D prostoru umístit polohy a orientace kamer, ze kterých byly snímky pořízeny, a zároveň přiřadit společným 2D prvkům 3D souřadnice.

Řídká rekonstrukce může trvat jednotky minut až několik hodin, to záleží na počtu, velikosti a podobnosti obrazových dat. Její výsledek nám umožňuje udělat si představu o tom jak dobře se rekonstrukce z dat podaří udělat, pokud spustíme druhou, externí metodu.

Pro rekonstrukci s hustou distribucí je potřeba stáhnout aplikaci CMVS profesora Yasutaka Furukawa[12]. Ta je proti řídké rekonstrukci časově mnohem náročnější. Vrací však mnohem víc bodů (statisíce až miliony) a většinou přesnější konfigurace kamer.

Kapitola 6

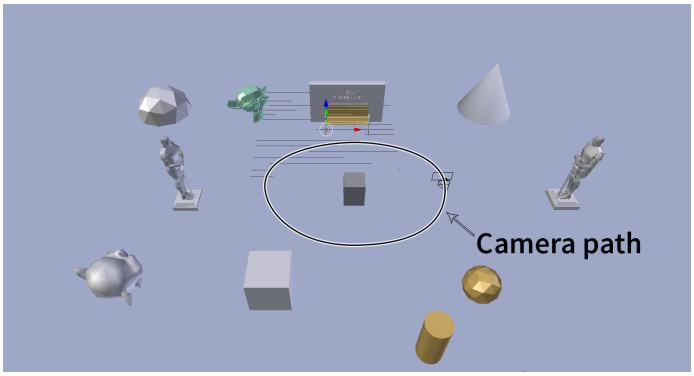
Trackování testovací scény s Visual SFM

Testovací sekvence má délku 10 vteřin, snímkovací frekvence je 30 snímků za vteřinu, celkem tedy 300 snímků. Každý snímek v sekvenci reálně odpovídá dvěma obrazům, jeden pro každé oko, a má rozlišení 4096×2048 pixelů. Trackování všech snímků by zabralo enormní množství času. Proto byly pro trackování smazány všechny snímky pro pravé oko a byla zmenšena snímkovací frekvence na polovinu smazáním všech obrazů se sudým pořadovým číslem. Zbylo tedy 150 snímků pouze pro levé oko. Není překvapením, že pokud byly programu Visual SFM na vstup vloženy neupravené snímky s úhlem záběru 360, nebyl schopen z jednotlivých obrazů zjistit ohniskovou vzdálenost čočky a výpočet selhal.

Obrázek 6.1: Středový výřez snímku

Na snímky byl aplikován středový výřez velikosti 1280×720 pixelů. Kamera se ve scéně pohybuje po eliptické dráze okolo krychle. Pokud proběhla rekonstrukce správně, trajektorie kamery by v rekonstrukci měla být jednoznačně poznat.

V řídké rekonstrukci jsou však kamery umístěny téměř na přímce. Po ořezu



Obrázek 6.2: Znázornění trajektorie kamery

byla proto ještě aplikována korekce objektivu, aby se křivky odpovídající rovným hranám v původní scéně narovnaly na přímky a jako vedlejší efekt byl zmenšen úhel záběru. To výsledek výrazně zlepšilo a kamery začaly mít větší rozptyl v ose x, nicméně stále byla trajektorie silně zkreslená. V husté rekonstrukci je navíc vidět, že Point cloud obsahuje hlavně body odpovídající Skyboxu, který vidí všechny kamery. Průměrně bylo na každém snímku rozpoznáno 2 500 společných prvků. V dokumentaci k Visual SFM je podobný případ popsán jako degenerativní, protože kamera hledí stále stejným směrem. Rekonstrukce je proto velmi vzdálená realitě.

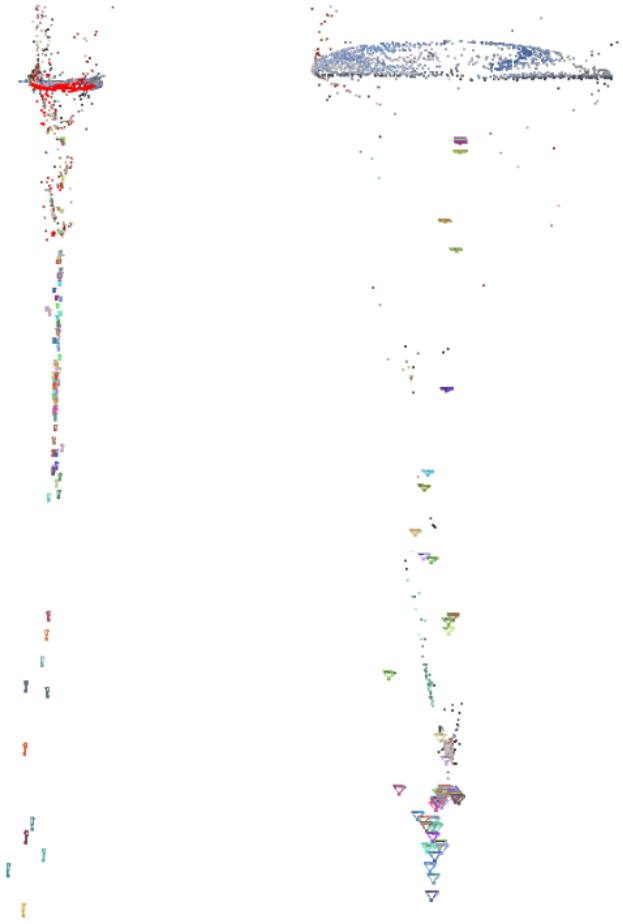
Protože tyto rekonstrukce jsou výpočetně a tedy i časově velmi náročné, před dalšími experimenty na rozsáhlých scénách bylo testování provedeno na krátkém videu natočeném na mobilní telefon. Jako testovací objekt byl použit kontrastní 3D vytištěný model sovy, který má jednoduchý základní tvar, ale i jemné detaily.

V prvním testu byl objekt umístěn na otočný podstavec a kamera pevně na stativ. Po každém snímku byl objekt mírně otočen. Po kompletním otočení okolo vlastní osy byl změněn úhel kamery a proces se opakoval.

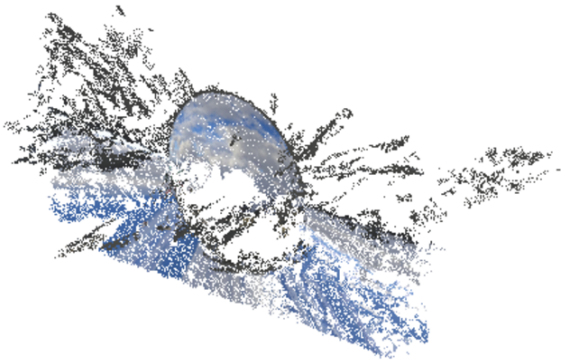
Při rekonstrukci se do 3D prostoru podařilo umístit pouze 2 kamery a rekonstrukce nevrátila takřka žádný výsledek. Problémem bylo, že všechna obrazová data pocházela z velmi malého rozsahu hloubky a poté pouze jednolitého pozadí. Přestože cílem rekonstrukce je hlavně červená sova a trajektorie kamery, různé předměty v pozadí a popředí mohou sloužit jako vodítko pro určení vzájemné polohy snímků. A čím větší je rozsah hloubky, ze které tyto shody pocházejí, tím větší informaci přináší do systému.

Pro druhý test byla proto sova umístěna na list čtverečkovaného papíru, pod který byly umístěny další dva barevné papíry a několik dalších předmětů do blízkého okolí. Dále bylo přidáno světlo jiné barevné teploty, které nasvítilo objekt pouze z jedné strany. Poté byla pořízena sada 37 fotografií z různých úhlů a vzdáleností v logickém pořadí tvořící souvislou dráhu.

Rekonstrukce však znovu selhala a nevrátila téměř žádná data. V tomto



Obrázek 6.3: Řídká rekonstrukce výřezu (vlevo) a rekonstrukce po korekci objektivu (vpravo)



Obrázek 6.4: Hustá rekonstrukce, většina bodů v rovině skyboxu

Obrázek 6.5: Testovací prostředí s otočným podstavcem

případě byl problém nedostatek dat, 37 fotografií v porovnání se zpracováním videa je velmi malé množství informace.

V dalším pokusu bylo místo sady obrázků pořízeno 4K video, ve kterém se kamera pomalu plynule pohybuje okolo snímaného objektu. Z videa byla poté vytvořena sekvence obrázků ve formátu JPEG se snímkovací frekvencí 5 snímků za vteřinu. Celkem bylo tak vytvořeno 350 snímků z původně 70 vteřin trvajícího videa. Všem snímkům byl navíc zvýšen kontrast a ostrost tak, aby byl zvětšen počet identifikovatelných společných prvků.

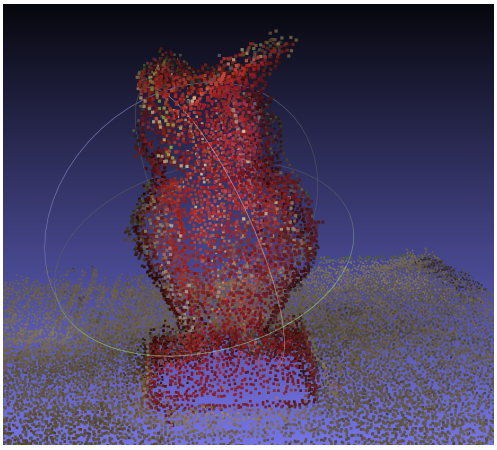
Ve Visual SFM byl místo standardního párování všech snímků vzájemně zvolen „Pairwise Matching -> Compute Sequence match“ a jako maximální vzdálenost mezi snímky pro párování byla vybrána hodnota 10. Výsledná rekonstrukce téměř perfektně odpovídá pohybu kamerou.

Protože řídká rekonstrukce věrně zachycovala trajektorii kamery i tvar sovy, byla provedena i hustá rekonstrukce. Ta obsahuje 719 600 bodů. Tvar sovy je v ní jednoznačně zachycen včetně detailů, dále je vidět rovina stolu a šum ve formě bodů ve volném prostoru.

Pro zpracování dat byl zvolen open-source program Meshlab. Jako první byl aplikován filtr Poisson-Disk Sampling, který odstranil většinu šumu a zároveň vytvořil rovnoměrně rozmístěné body. Počet bodů se tak zmenšil na 35 000.

Polygonální reprezentace bylo dosaženo pomocí výpočtu normál k 1 000 nejbližším sousedním bodům s ohledem na úhel kamery (bez něj je občas nutné normály otočit) a poté použit filtr Screened poisson surface reconstruction. V zadní části modelu bohužel nebyl dostatek dat a tak na místě vznikl útvar přemosťující díru v modelu. Ten byl ručně smazán. Pro potřeby 3D tisku či digitalizaci modelu do hry je výsledná kvalita modelu nedostatečná. Na druhou stranu pro potřeby této práce, kde bude sloužit maximálně pro výpočet kolizí a nebude uživateli přímo zobrazen, je více než dostačující.

Obrázek 6.6: Rekonstrukce trajektorie kamery a geometrie snímaného objektu



Obrázek 6.7: Model sovy po aplikaci Poisson-Disk Sampling

6.1 Import dat do Unity

Součástí výstupu Husté rekonstrukce je soubor „cameras_v2.txt“, který obsahuje informace o rozmístění a orientaci kamer ve scéně.

Důležité jsou řádky obsahující původní název souboru, vektor C pozice kamery a vektor R kvaternion rotace kamery. Pro import do Unity byl implementován skript TrackingParser, který zpracovává textová data a načítá je do datové struktury List<KeyValuePair<int,float[]». První hodnota v páru je

```
00000000.jpg
owl_img010.jpg
4208.00048828
1920 1080
-0.0695915701893 0.391269208658 0.21323189079
0.18715186504 0.404935318028 -0.0663519539448
-2.34833936348 0.406068680416 -0.321640937366
0.360127864333 -0.911006504239 0.157528854126 -0.124776252754
0.919233692374 -0.197147321386 0.340801063392
-0.376888943081 -0.690986324999 0.61684653348
0.113878944164 -0.69546877358 -0.709479052698
-0.253569091121
0 0 0

# Filename (of the undistorted image in visualize folder)
# Original filename

# Focal Length (of the undistorted image)
# 2-vec Principal Point (image center)
# 3-vec Translation T (as in P = K[R T])
# 3-vec Camera Position C (as in P = K[R -RC])
# 3-vec Axis Angle format of R
# 4-vec Quaternion format of R
# 3x3 Matrix format of R
# [Normalized radial distortion]
# 3-vec Lat/Lng/Alt from EXIF
```

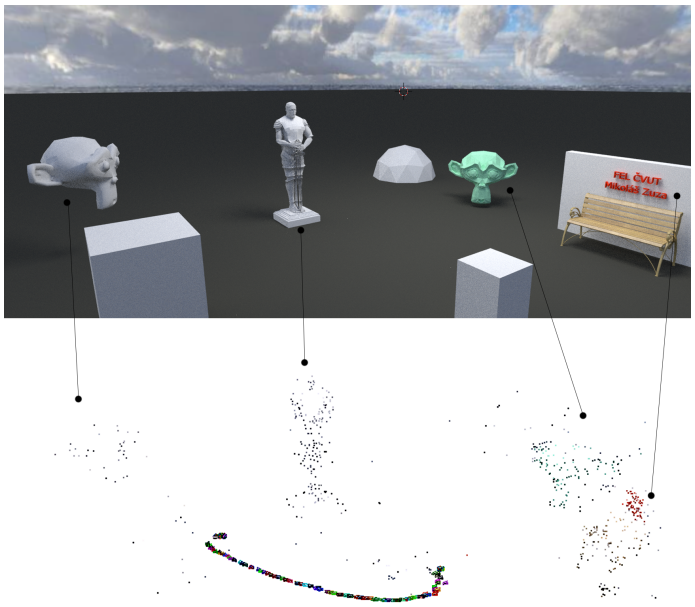
Obrázek 6.8: Záznam jedné kamery

číslo označující index snímku v původním videu. Tato hodnota se totiž nemusí shodovat s pořadovým číslem v rekonstrukci a bez korekce by snímky nebyly ve správném pořadí vzhledem k videu. Pomocí regulárního výrazu byl z názvu souboru extrahován jeho index. List byl pomocí lambda funkce seřazen právě podle tohoto indexu. Druhá položka v páru je pole floatů, má 7 prvků, 3 souřadnice polohy kamery a 4 hodnoty kvaternionu rotace.

Skript kromě načítání dat simuluje pohyb kamery ve scéně. Interpoluje mezi jednotlivými načtenými klíčovými snímky konfigurací kamer. Navíc se mu dá nastavit a za běhu měnit rychlost přehrávání a udělat korekci natočení okolo osy Z tak, aby rovina podlahy načtených dat odpovídala rovině XZ v Unity. V Unity je tak možné si přehrát celý proces natáčení objektu a to ideálně i s vloženým modelem vytvořeným v Meshlabu popsaným v předchozí části.

S nově získanými zkušenostmi byl změněn způsob, kterými se pořizují data pro trackování sférického videa. Místo výřezu byl obraz promítnut na

sféru pomocí již implementovaného přehrávače v Unity. Poté byl implementován skript, který umožňuje měnit orientaci kamery a natočen záběr, kde podobně jako u sovy je středový předmět obcházen kolem dokola. Výsledná rekonstrukce je jednoznačně nejlepší, jaká se doposud v této práci povedla. Jsou v ní rozpoznatelné jednotlivé objekty scény a trajektorie scény má tvar elipsy.



Obrázek 6.9: Hustá rekonstrukce sférického videa a korespondence v původním snímku

Kapitola 7

Natáčení sférického videa

7.1 Sférické kamery

Na rozdíl od klasických kamer mají sférické kamery typicky více než jednu optiku a čip, některé si vystačí jen s dvojicí, některé jich mohou mít i více než deset. Záběry jednotlivých kamer se kombinují do výsledného sférického obrazu. Více kamer umožňuje pořídit snímek ve větším rozlišení, zároveň však přibývá i počet švů v místě napojení dílčích snímků. Problematická je také synchronizace závěrek, pokud se nijak neřeší, mohou mít mezi sebou dílčí snímky posun až o polovinu snímkovací frekvence a výrazně se tak zhorší možnost snímky plynule napojit.

Pro účely této práce byly použity kamery Giroptic 360cam, Elmo QBiC PANORAMA X a Ricoh Theta, které zapůjčil Ing. Jan Buriánek. Všechny tři kamery jsou monoskopické podobně jako většina sférických kamer na trhu.

Giroptic 360cam

Kamera od firmy Giroptic má trojici optik v jednom kompaktním obalu a je voděodolná. Díky tomu, že jsou kamery součástí jediného zařízení, není nutné nijak řešit synchronizaci závěrek, provádí se automaticky interně. Kamera navíc i snímky v reálném čase skládá do sférické fotografie či videa a dokonce je možné video při natáčení sledovat přes aplikaci pro mobilní systém Android. Výsledné rozlišení sférického snímku je však poměrně malé, 2048×1024 při 30 snímcích za sekundu. Napojování je kvalitou průměrné a švy je tak možné na první pohled identifikovat.

Největší problém kamery je přeskakování snímků a trhané skoky vzniklé chybným skládáním snímků. Ve videu jsou hodně viditelné a objevují se občas i několikrát za sekundu, před trackováním videa z kamery Giroptic by bylo nutné ručně tyto snímky odstraňovat případně nějak automaticky filtrovat. Rozlišení je na sférické video navíc opravdu malé. Kamera je tak vhodná spíše pro amatérské natáčení, čemuž odpovídá i pořizovací cena 13 000 Kč.

Obrázek 7.1: Kamera Giroptic a detail napojení snímků tvořící šev

Elmo QBiC PANORAMA X

Sférický rig Elmo QBiC je tvořen čtveřicí širokoúhlých kamer QBiC MS-1 XP. Tyto kamery jsou připevněné na kovový rám se stativovým závitem. Protože se jedná o sestavu samostatných kamer, synchronizace neprobíhá interně. Je ale možná pomocí mobilní aplikace, která závěrky synchronizuje bezdrátově. Při testování se však ukázalo, že tato synchronizace nepracuje bezchybně a závěrky mají mezi sebou drobný časový posun. Navíc se synchronizuje opravdu jen relativní čas pořízení snímků a ne čas začátku nahrávání. Při vytváření sférického videa je tedy nutné ještě ručně začátky posunout. Rozlišení každé z kamer je 1920×1080 při 29 snímcích za sekundu. Výsledný sférický snímek tak má výrazně vyšší rozlišení než v případě kamery Giroptic.

Sférický snímek je ale nutné ručně skládat, kamery pouze ukládají jednotlivá videa na SD kartu. Kovový rám navíc není dokonale pevný a i minimální pohyb kamer výrazně ovlivňuje možnost snímky plynule napojit.

Obrázek 7.2: Kamera Elmo QBiC PANORAMA X

7.1.1 Ricoh Theta

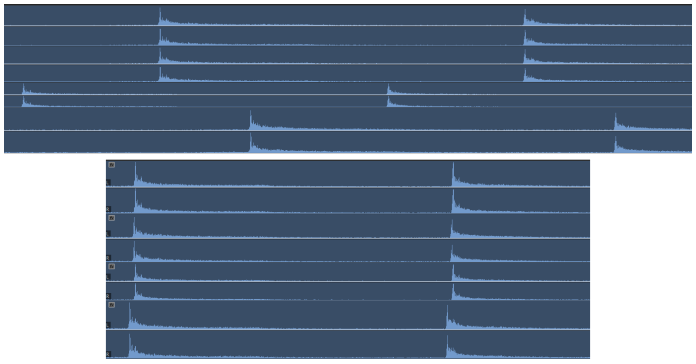
Ricoh Theta je kapesní sférická kamera využívající pouze dvě optiky. Synchronizace probíhá interně podobně jako skládání výsledné sférické fotografie nebo videa. Protože jsou použity pouze dvě optiky, ve snímku je pouze jeden šev oddělující dvě hemisféry. Sférické video má rozlišení 1920×960 při 30 snímcích za sekundu.

Obrázek 7.3: Kamera Ricoh Theta

7.2 Synchronizace

Kamera Elmo QBiC PANORAMA nesynchronizuje začátky videí, synchronizuje pouze čas, kdy se z čipu čtou data a vytvoří se tak snímek. Proto je nutné tuto synchronizaci ručně provést v postprocessingu. Možností jak nalézt společnou časovou značku je několik, mezi nejpopulárnější techniky patří synchronizace zvukem (filmová klapka), pohybem (prudké pohnutí konstrukce, na které jsou kamery připevněny) a světlem (zhasnutí světla v místnosti). Protože všechny kamery Elmo QBiC nahrávají zvuk, zvolil jsem synchronizaci časem. Po spuštění nahrávání všech kamer jsem hlasitě tlesknul, toto tlesknutí se výrazně zobrazí v grafu hlasitosti v čase. Jednotlivá videa pak stačí posunout tak, aby zvuková špička nastala ve stejný moment. Pokud by závěrky nebyly synchronní, nebude možné takový moment nalézt, protože jsme omezeni minimálním krokem posunu odvozeným ze snímkovací frekvence.

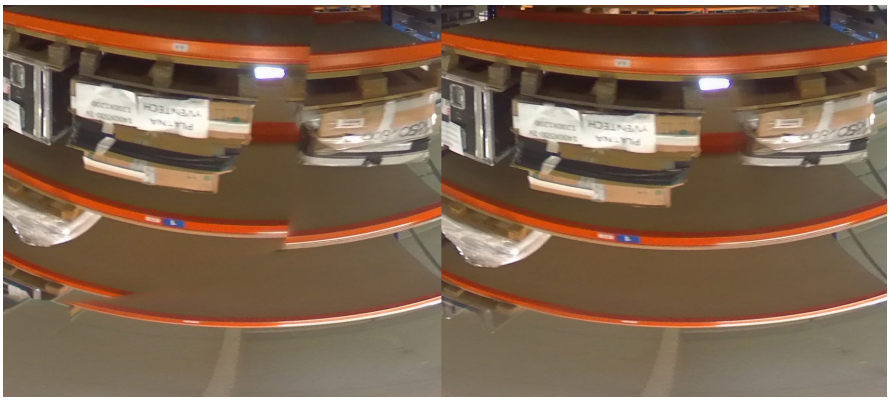
Přestože kamera Elmo QBiC synchronizaci závěrek provádí, není zarovnání dokonalé. Při testování byl maximální vzájemný posun času závěrky o jednu pětinu snímku. Při 30 snímcích za sekundu to odpovídá posunu o 0.0066 sekundy, což je dostatečně nízká hodnota, aby znatelně degradovala kvalitu videa.



Obrázek 7.4: Synchronizace podle zvukových stop. Před synchronizací (nahoře) a po synchronizaci (dole)

7.3 Stitching

Pokud kamera neumí sférické video automaticky vytvářet, je nutné ho vytvořit pomocí procesu nazývaného stitching z dílčích videí. PtGui je pokročilý program pro skládání panoramat. Jednotlivé snímky umí automaticky zarovnat, pro sférickou kameru je ale lepší použít šablonu od výrobce kamery. Protože kamery nejsou vždy v dokonale zarovnaných polohách, je vhodné šablonu upravit a zlepšit tak plynulé napojení snímků. Po úpravě šablony je možné vytvořit ze čtyř snímků jeden snímek sférický, zkontrolovat napojení a opakovat úpravy, dokud nejsme s napojením spokojeni.



Obrázek 7.5: Napojení snímků před (vlevo) a po (vpravo) korekci

Finální šablona bude použita pro vytvoření sférického videa. PtGui umí vytvářet pouze jednotlivé snímky, pomocí PtBatchBuilder je ale možné nagenerovat projekty pro každý snímek videa. Všechny snímky je nutné uložit do stejné složky a seřadit lexikograficky primárně podle čísla snímku a sekun-

dárně podle čísla kamery. Jedenáctý snímek druhé kamery se tedy může jmenovat například 00011_B. Po vygenerování projektů je vhodné je zpracovat na pozadí. Proces vytváření snímků může trvat i několik hodin v závislosti na jejich počtu. Z výsledných sférických snímků je poté možné vytvořit video ve formátu mp4, které lze importovat do Unity.

7.4 Korekce expozice

Při natáčení sférického videa je správná volba expozice netriviální. Pokud není scéna dokonale rovnoměrně nasvícená, některé kamery budou orientované směrem ke zdroji světla a kamery na druhé straně naopak od tohoto světla. Při zvolení stejné expozice na všech kamerách tak budou snímky z kamer ke světlu nadexponované, některé kamery správně exponované a kamery od světla podexponované. Při napojování dílčích snímků do sférického obrazu budou ale díky stejné expozici snímky barevně plynule navazovat. Lepší variantou je nechat každou kameru exponovat samostatně. Všechny dílčí snímky pak budou korektně exponované, při napojení bude ale viditelný skok v expozici na rozhraní snímků. Proto je nutné provést korekci expozice. Většina programů umí tuto korekci automaticky, přesto je často dobré ještě ručně expozici upravit například pomocí eliptických masek s rozmazanou hranicí.

Obrázek 7.6: Srovnání sférických snímků před (nahoře) a po (dole) korekci expozice

7.5 Porovnání obrazové kvality kamer

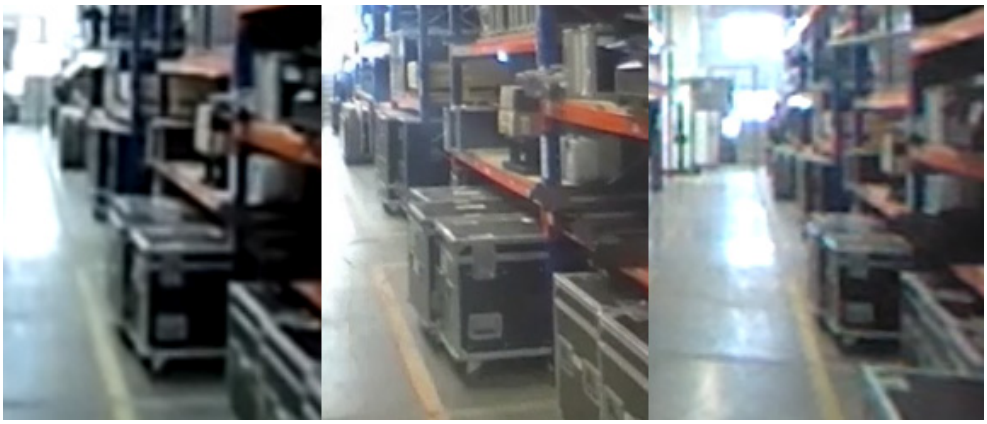
Pro srovnání možností kamer byla se všemi natočena totožná scéna. Pro plynulý pohyb byl připevněn stativ na pojízdnou krabici. Krabice byla tlačena uličkou tam a zpátky za spodní část tak, aby se operátor kamery maximálně schoval do slepého úhlu kamer, respektive aby byl co nejméně viditelný.

Obrázek 7.7: Pojízdný sférický camera rig

Nejhůře v testu dopadla kamera Giroptic. Video má malé rozlišení 2048 × 1024, vysokou kompresi, malý datový tok 14 Mb/s a viditelné švy. V kvalitě obrazu je na druhém místě kamera Ricoh Theta. Video má sice ještě menší rozlišení 1920×960, je ale velmi dobře napojené, má větší datový tok 17 Mb/s a obraz je viditelně kvalitnější. Nejlépe v testu dopadla kamera QBiC Panorama X. Video je sice nutné ručně skládat, výsledné rozlišení 3904 × 1952 má ale téměř čtyřikrát více pixelů než předchozí dvě kamery a datový tok 37 Mb/s.

Výsledky experimentu tak určily kameru QBiC Panorama X jako nejvhodnější pro další části této práce.

Obrázek 7.8: Srovnání sférických kamer, zleva Giroptic, QBiC Panorama X, Ricoh Theta



Obrázek 7.9: Srovnání sférických kamer v detailu, zleva Giroptic, QBiC Panorama X, Ricoh Theta

7.6 Natáčení sférického videa pro interaktivní aplikaci

7.6.1 Camera rig

Kamera QBiC Panorama X, která byla vybrána na základě testu obrazové kvality, se skládá ze čtyř kamer a kovového rámu. Na spodní straně tohoto rámu se nachází stativový závit, díky kterému je možné kameru připevnit na nejrůznější typy příslušenství. Při výběru tohoto příslušenství je důležitá zejména jejich velikost a tvar s ohledem na to, aby se co největší část schovala do slepého úhlu kamer, respektive aby byla vidět co nejméně. Zároveň je důležitá stabilita kamery a přenositelnost. Pro statický záběr je tedy ideální monopod, který bude v záběru téměř neviditelný, při pohybu kamery ale není dostatečně stabilní a kamera má jeho vrcholu nežádoucí volnost pohybu. Tri-

pod je stabilnější, pro minimalizaci viditelnosti v záběru je ale vhodné jeho nohy neotevřít na maximální rozpětí. Pro plynulý pohyb byl použit skateboard, který má dvě kuličková ložiska na každé kolo, díky kterým se dokáže pohybovat extrémně plynule. Pro zvětšení plochy skateboardu na něj byla umístěna krabice a na tuto krabici samotný tripod s kamerou. Drtivá většina tripodů obsahuje v horní části páku pro změnu náklonu směrem vpřed a vzad. Tato páka by byla viditelná v záběru, proto byla odstraněna a nahrazena stahovací páskou, která svým tlakem zabraňuje volné změně náklonu. Tento camera rig je vhodný pouze pro lokace s velmi rovným povrchem a tedy i minimem vibrací přenášených do stativu a kamery. Pokud je povrch hrubší, musí operátor kamery při jízdě na skateboardu držet stativ v ruce nad svou hlavou tak, aby minimalizoval svou viditelnost v záběru a svým tělem tlumit vibrace.

Obrázek 7.10: Pojízdný sférický camera rig

7.6.2 Volba lokace

Hlavním kritériem pro výběr lokace byl hladký povrch, který by umožnil natočit plynulé video s minimem otřesů. Dále byl výběr proveden s ohledem na rozmístění, tvar a počet volně stojících objektů, se kterými bude možné

interagovat, které budou usnadňovat tracking scény a zároveň budou respektovat minimální vzdálenost od kamery pro správné napojení sférického videa. Lokace by měla být dále vizuálně zajímavá a v neposlední řadě by měla reprezentovat České vysoké učení v Praze. Proto byly zvoleny lokace v kampusu ČVUT před Národní technickou knihovnou a v parku za NTK.

Obrázek 7.11: Prostor před Národní technickou knihovnou, převzato z [16]

7.6.3 Natočená videosekvence

Video se skládá ze dvou scén. První se odehrává v parku za Národní technickou knihovnou. Kamera se pohybuje ze středu tohoto parku směrem k ulici Studentská, během tohoto pohybu míjí několik stromů a laviček. Ve druhé scéně se kamera nachází před NTK a pohybuje se po trajektorii kopírující obvod knihovny. Během své cesty se několikrát zastaví. Od všech objektů byla dodržena minimální vzdálenost jednoho metru.

7.6.4 Střih a úpravy videa

Videa byla oříznuta a vyexportována jako sekvence obrázků v programu Adobe Premiere Pro. Pomocí PtGUI a PtBatchStitcher byla vytvořena sekvence sférických snímků. Tato sekvence byla opět importována do Adobe Premiere Pro, kde byla provedena korekce expozice a kontrastu. Mezi dvě scény byl vložen plynulý přechod přes černou barvu za účelem minimalizace nevolnosti při přehrávání pomocí HMD.

Kapitola 8

Tvorba interaktivní aplikace

8.1 Trackování reálného sférického videa

Na základě výsledků rešerše softwaru pro trackování videa bylo trackování provedeno pomocí programů Visual SFM a Adobe After Effects. Oba tyto programy očekávají na vstupu sekvenci snímku z perspektivní kamery. Během rekonstrukce navíc vzniká souřadný systém vzhledem k prvnímu snímku videa. Proto bylo video přehrané pomocí sférického přehrávače pro Unity implementovaném v této práci s kamerou v počátku souřadnicového systému a nulovou rotací. Záznam sférické projekce byl uložen jako perspektivní obraz a rozdělen na dvě části souvislého pohybu kamery podle přechodu mezi scénami. Poté byl použit pro tracking a byl použit jako vstup pro Visual SFM a Adobe After Effects.

Visual SFM úspěšně vytvořilo hustý point cloud, rekonstrukce trajektorie kamery měla bohužel příliš velkou chybu a obsahovala mezery, kde informace o trajektorii zcela chybí pro některé snímky.

8.1.1 Tracking s Adobe After Effects

Tracker Adobe After Effects umožňuje video stabilizovat, sledovat pohyb objektu ve scéně nebo rekonstruovat parametry, pohyb a orientaci kamery. Rekonstrukce kamery probíhá ve dvou krocích, nejprve je analyzován dvourozměrný pohyb kontrastních shluků pixelů. Poté se rekonstruuje samotná kamera. Zatímco první krok je nutné provést pouze jednou, výpočet parametrů kamery se opakuje při každé změně sledovaných bodů. Body je možné přidávat, důležitější je ale možnost body mazat. Je vhodné smazat všechny body detekované na nestatických objektech ve scéně. Po smazaní proběhne znovu rekonstrukce kamery a podle zobrazení průměrné chyby rekonstrukce můžeme sledovat jak se tracking zlepšil či zhoršil.

Výstupem rekonstrukce Adobe After Effects je množina klíčových snímků pozic kamery, orientace kamery a pozice trojrozměrných bodů. Označením a

Obrázek 8.1: Obrazovka rekonstrukce kamery v Adobe After Effects, tři body definující rovinu.

kopírováním klíčových bodů pozic kamer v časové ose se tato data uloží do schránky v textové podobě. Vložením do textového dokumentu můžeme tato data trvale uložit. Po hlavičce souboru následuje pro každý klíčový snímek jeden řádek s pořadovým číslem snímku a trojicí souřadnic. Stejným postupem je možné vytvořit soubor s informacemi o orientaci kamery. Data z jednotlivých souborů ke každé scéně je vhodné spojit do jednoho souboru.

```
Adobe After Effects 8.0 Keyframe Data

        Units Per Second        30
        Source Width    1920
        Source Height   1080
        Source Pixel Aspect Ratio       1
        Comp Pixel Aspect Ratio 1

Transform       Position
        Frame   X pixels        Y pixels        Z pixels
0       906.323 537.841 -815.135
1       906.333 537.841 -815.149
2       911.189 538.052 -821.466
3       915.405 538.22  -826.884
4       919.69  538.396 -832.364
```

Obrázek 8.2: Ukázka souboru s pozicemi rekonstruované kamery

8.2 Import dat

Pro import těchto dat do Unity byl implementován data parser. Souřadnicový systém Adobe After Effects má opačný směr osy Y proti souřadnicovému systému Unity, tento fakt je důležité při načítání zohlednit jak pro určení pozice, tak při tvorbě kvaternionů odpovídajících orientaci kamery.

Měřítko rekonstruované scény vychází ze vzdálenosti původně dvojrozměrných bodů v pixelech. Výsledné rozměry scény tak mohou přesahovat i hodnoty 1×10^6 a je nutné jejich rozsah zmenšit. Skript připnutý na hlavní kameře ve scéně interpoluje mezi pozicemi aktuálního a následujícího klíčového snímku. Snímkovací frekvence videa je 30 snímků za vteřinu, tomu odpovídá 30 řádků souřadnic pro každou vteřinu. Snímkovací frekvence pro HMD by měla být alespoň 90 snímků za vteřinu, mezi rekonstruovanými hodnotami proto budou vždy minimálně 3 interpolované. Stejným způsobem ale nelze aplikovat orientaci kamery, tu určuje uživatel myší nebo pohybem hlavy s HMD.

Během přehrávání videa je nutné provádět synchronizaci načítaných dat a aktuálního snímku videa, bez synchronizace by se důsledkem zahozených snímků při dočasném nedostatku výkonu postupně zvětšovala chyba zarovnání.

8.3 Inverzní rotace sférické projekce

Rotaci lze aplikovat pomocí rotační matice, aby bylo možné rotace jednoduše plynule interpolovat, je lepší použít kvaterniony. Protože orientaci kamery určuje uživatel, korekce natočení musela být implementována jiným způsobem, než přímým upravením transformace kamery. Během každého vykreslení se interpolují data rotací a jejich inverzí se otáčí nultý poledník sférické projekce. Díky tomu statické objekty ve scéně zůstávají statické i vzhledem k objektům ve sférickém videu.

Velmi užitečným vedlejším efektem tohoto přístupu je korekce natočení kamery vzhledem k prvnímu snímku. Během natáčení sférického videa vždy dojde k nežádoucí rotaci okolo svislé osy, která při přehrávání mění bod zájmu, aniž by uživatel aktivně změnil pohled. Tato rotace může zejména při přehrávání pomocí HMD narušovat orientaci v prostoru a v extrémních případech i způsobit kinetózu. Automatická rotace sférické projekce toto natočení kompenzuje a minimalizuje tak nechtěnou změnu pohledu.

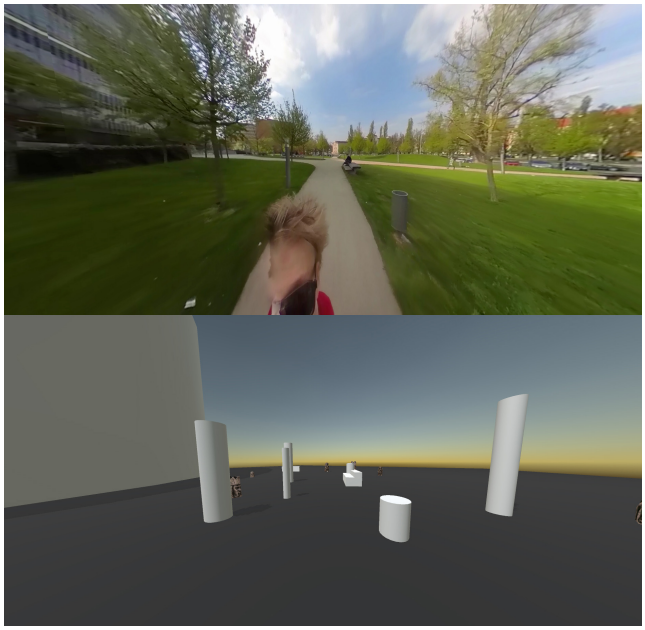
8.4 Tvorba kolizních těles

Adobe After Effects vytváří pouze řídký point cloud, který nelze jako celek z programu přímo exportovat. Je možné ale exportovat jednotlivé body, případně ze tří bodů vytvořit rovinu. Díky tomu je možné v Unity vytvořit rovinu odpovídající zemi ve videu, a zjistit polohu některých objektů ve scéně. Polohy

ostatních objektů, které nemají žádné referenční body v point cloudu, je možné zjistit ze vzdáleností v reálném světě k objektům, které reference mají. Pro výpočet kolizí je vhodné používat zjednodušené reprezentace objektů, proto byly stromy, lampy a koše nahrazeny válcem s kolizním tělesem tvaru kapsle, podobně jako lidé ve scéně a budova NTK v odpovídajícím měřítku. Lavičky byly nahrazeny kvádry.

8.5 Přepínání mezi scénami

Sférické video se skládá ze dvou videosekvencí, mezi kterými je přechod přes černou barvu. Každá videosekvence se musí trackovat zvlášť a výstupní data rekonstrukce mohou mít různé měřítko. Při přechodu mezi scénami proto implementovaný skript upravuje měřítko, parametry fyzikální simulace a aktivní objekty tak, aby měl uživatel stejnou zpětnou vazbu na akce nezávisle na aktuální scéně. Pro tyto přechody je nutné nastavit pořadová čísla snímku pro všechny střihy ve videu. Při přechodu jsou rovněž smazány všechny dočasné herní objekty (projektily).



Obrázek 8.3: Snímek ze sférického videa a kolizní tělesa v Unity

8.6 Herní obsah

Herní mechanika by měla co nejvíce přimět uživatele k interakci s objekty ve sférickém videu. Proto může hráč do scény střílet kovové kuličky. Do obou scén

byly rozmístěny objekty, pokud je hráč projektilem trefí, přičte se mu bod a objekt spadne na zem. V době kdy je objekt sražen a leží na zemi je nadále možné ho zasáhnout, hráč za to ale neobdrží žádný bod. Po několika sekundách se objekt znovu postaví, je možné ho opět zasáhnout a získat bod. Při zásahu těchto objektů se přehraje zvuková stopa oznamující úspěch. Projektily mají omezenou životnost, po 3 sekundách se začnou zmenšovat a krátce poté zmizí. Objekty jsou záměrně rozmístěny tak, aby v některých místech trajektorie kamery byl mezi hráčem a cíleným objektem nějaký další objekt z videa, v tomto případě strom, lavička nebo lampa. Při snaze získat bod se tak hráči může projektil od rekonstruovaných objektů z videa odrazit. Za zásah clonících objektů hráč žádný bod nezíská ani neztratí. Při zásahu stromu se ozve zvuk zásahu dřeva a odpovídající zvukovou odezvu mají i další objekty ve scéně. Objekty ve scéně jsou nasvíceny ze stejného směru jako objekty ve videu a odráží barvy z videa, díky tomu se zvětšuje pocit věrohodnosti objektů ve scéně. Ještě více by věrohodnost podpořily stíny objektů vržené na neviditelnou plochu, která by zobrazovala pouze tyto stíny. Tuto funkcionalitu se bohužel nepodařilo pro časovou náročnost implementovat.

8.6.1 Implementace

Ovládání se mění v závislosti na platformě, mířit lze myší, kterou se zároveň mění i pohled. Levým tlačítkem myši se ovládá střelba projektilů. Pokud uživatel používá HMD, ovládání je přirozenější a umožňuje nezávisle na sobě měnit pohled náklonem hlavy a mířit pomocí jednoho či dvou ovladačů HTC Vive. Při ovládání pomocí HTC Vive se střelba spouští zmáčknutím spouště ovladače. Pro implementaci skriptu ovladače byla použita knihovna OpenVR[15]. Tato knihovna obsahuje třídu SteamVR_TrackedController, která zajišťuje indexování ovladačů a základní čtení vstupu a polohy. Z této třídy dědí vlastní třída ovladače WandController. Neintuitivní je získání směru ovladače, ten není určen podle horní plochy ovladače, ale podle madla. Pro získání přirozeného směru ovladače je nutné získanou rotaci otočit o 45 stupňů okolo vektoru směřujícího vpravo v lokálním souřadnicovém systému ovladače.

Před startem videa a hry si hráč může prohlédnout objekty, za jejichž zásah získá bod. Start hry se neprovádí přímo kliknutím na tlačítko „Start“, místo toho musí hráč tlačítko zasáhnout projektilem, díky tomu se ihned seznámí s herní mechanikou a fyzikou projektilů. Po přehrání obou scén se hráči zobrazí dosažené skóre a aplikace znovu přejde do úvodního stavu.

Uživatelské rozhraní je minimalistické s ohledem na použití HMD, kde není vhodné používat dvojrozměrné prvky statické vzhledem ke kameře. Zobrazení aktuálního skóre je řešeno pomocí textu pozicovaném relativně k jednomu z ovladačů. Hráč tak nemusí přeostřovat na prvek blízko ke kameře a při míření na objekty ihned získává zpětnou vazbu o zásahu zvýšením počítadla. Pro snadnější míření ovladači HTC Vive byl přidán paprsek, který z ovladače vy-

```
public class WandController : SteamVR_TrackedController {

        public SteamVR_Controller.Device controller
        { get { return SteamVR_Controller.Input(
        (int) controllerIndex); } }
        public Vector3 velocity
        { get { return controller.velocity; } }
        public Vector3 angularVelocity
        { get { return controller.angularVelocity; } }

        protected LineRenderer lineRenderer;
        protected Vector3[] lineRendererVertices;

        bool isTriggerPressed = false;
        bool isTouchpadPressed = false;

        public PlayerController playerC;
        protected override void Start();
        protected override void Update();
        public override void
        OnTriggerClicked(ClickedEventArgs e);
        public override void
        OnTriggerUnclicked(ClickedEventArgs e);
        public float GetTriggerAxis();
}
```

Obrázek 8.4: Definice třídy WandController

Obrázek 8.5: Úvodní obrazovka aplikace

chází ve směru střelby. Paprsek mění barvu ze zelené na červenou v okamžiku,

kdy uživatel zmáčkne spoušť.

Obrázek 8.6: Počítadlo zásahu na ovladači HTC Vive

8.7 Testování

8.7.1 Anifilm

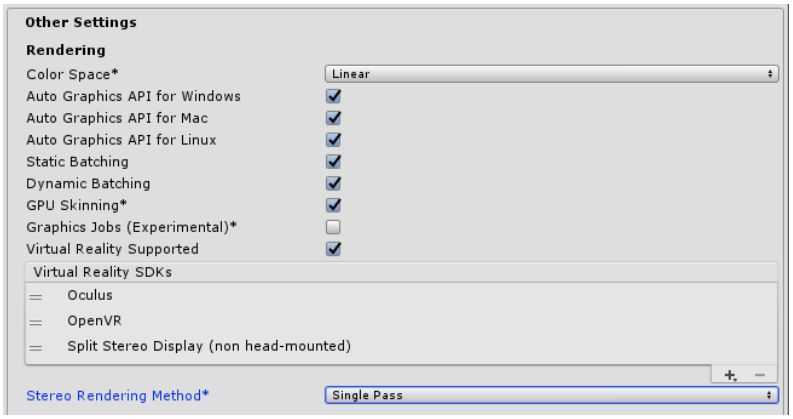
Testování probíhalo na Mezinárodním festivalu animovaných filmů v Třeboni na stánku katedry počítačové grafiky a interakce. Návštěvníci si zde mohli vyzkoušet interaktivní sférické video na zařízení HTC Vive. Všichni návštěvníci včetně dětí ihned pochopili ovládání aplikace a ani po opakovaném hraní nepociťovali nevolnost nebo jiné nepříjemné pocity.

Největším problémem při testování byla hardwarová chyba zařízení HTC Vive, kterému v některých situacích přestal zobrazovat displej, přestože ho měl uživatel na hlavě. Zhasínání pravděpodobně způsoboval probíjející audio konektor, na místě se problém bohužel nepodařilo odstranit. Dalším problémem při testování bylo občasné odlinkování knihovny OpenVR.dll během běhu aplikace, které způsobí pád programu. Problém se několikrát objevil i na ostatních testovaných aplikacích.

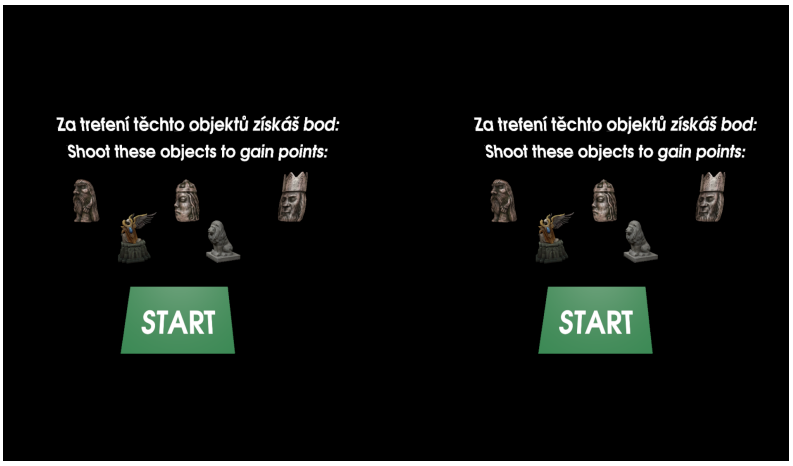
8.7.2 Stereoskopická projekční stěna

Další testování bylo provedeno na stereoskopické projekční stěně ve VR Labu katedry Počítačové grafiky a interakce na Karlově náměstí. Projektor Optoma stereoskopické projekční stěny očekává snímek ve formátu side-by-side, roztažený vertikálně tak, aby zabral celou obrazovku. Aby bylo v aplikaci možné takový snímek renderovat, je nutné přidat v nastavení přehrávače VR SDK pojmenované Split Stereo Display (non head-mounted) a umístit ho v seznamu pod SDK OpenVR. Díky tomu aplikace při spuštění nejdříve ověří, zda je připojen HMD. Pokud ano, bude obraz renderovat pro něj, pokud ne, zobrazí

se snímek side-by-side vhodný pro stereoskopickou projekční stěnu. Při spuštění aplikace je nutné zvolit zobrazení přes celou obrazovku, jinak projektor nedetekuje stereoskopický vstup.



Obrázek 8.7: Nastavení side-by-side stereoskopického zobrazení v Unity



Obrázek 8.8: Stereoskopické zobrazení aplikace side-by-side

Pro stereoskopický vjem musí mít uživatel nasazeny aktivní 3D brýle. Projektor pomocí infračerveného záření vysílá synchronizační časovač, podle kterého brýle střídavě zakrývají pravé a levé oko. Během testování byl ověřen vjem hloubky ze stereoskopického obrazu a byla upravena vzdálenost paralaxy pro menší zátěž očí.

Obrázek 8.9: aktivní 3D brýle Optoma

Závěr

Kamery pro záznam sférického videa se liší počtem objektivů, způsobem napojování obrazů, rozlišením a mnoha dalšími aspekty. Téměř žádné kamery nejsou schopné zaznamenávat sférické video stereoskopicky a monoskopické kamery mají až na nejnovější a nejdražší modely nedostatečné rozlišení.

V práci byl implementován přehrávač stereoskopického sférického videa v herním enginu Unity. Pro testování byla vytvořená krátká stereoskopická sférická videosekvence v programu Blender. Přehrávač podporuje HMD zařízení typu HTC Vive. Po srovnání dostupných sférických kamer byla s nejlepší kamerou pořízena krátká videosekvence.

V rešerši knihoven pro rozšířenou realitu byly popsány důvody proč jejich funkcionalita nestačí nebo nevyhovuje pro účely analýzy stereoskopického sférického videa. Na základě rešerše softwaru pro rekonstrukci geometrie scény a trajektorie kamery z videa byla scéna pomocí programu Visual SFM a Adobe After Effects rekonstruována. Pro oba programy byl implementován parser dat umožňující jejich načtení do herního enginu Unity.

Výstupem práce je aplikace, umožňující interagovat s objekty ve sférickém videu, díky synchronní transformaci kamery a natočení sférické projekce vzhledem k aktuálnímu snímku videa. Aplikaci je možné ovládat jak pomocí myši, tak pomocí ovladačů HTC Vive a přehrávat ji lze na běžném monitoru, uvnitř HMD i na stereoskopické projekční stěně.

Literatura

[1] J. Buriánek, Z. Rychetnik: *Internal meeting for 8KSVIT project* 12/2016, AV MEDIA

[2] ARToolKit: *How does ARToolKit work?*, [online][cit. 2017-19-5], Dostupné z: `http://www.hitl.washington.edu/artoolkit/documentation/userarwork.htm`

[3] Vuforia: *Eorld's most widely used augmented reality platform* [online][cit. 2016-20-11], Dostupné z: `https://vuforia.com/`.

[4] Google: *Google VR SDK*[online] [cit. 2016-10-2], Dostupné z: `https://developers.google.com/vr/unity/`

[5] Undergroundcity3d: *3D point cloud* [fotografie]. Dostupné z: `http://www.undergroundcity3d.com/uvod/forum/3d-point-cloud/`. Formát: 687x409

[6] Androidauthority: *Asus Zenphone AR* [fotografie]. Dostupné z: `http://www.androidauthority.com/asus-zenfone-ar-hands-on-tango-daydream-8gb-of-ram-oh-my-741242/`. Formát: 693x433

[7] Google: *Project Tango - Build apps that understand space and motion in high fidelity*, [online][cit. 2016-22-11], Dostupné z: `https://developers.google.com/tango/`.

[8] Trusted Reviews: *What is Google Tango? Google's AR tech explained*, [online][cit. 2017-10-01], Dostupné z: `http://www.trustedreviews.com/news/what-is-project-tango`.

[9] Kudan: *Kudan Computer Vision* [online][cit. 2016-15-10], Dostupné z: `https://www.kudan.eu/`.

[10] Viscoda: *Voodoo camera tracker*[online][cit. 2017-19-02] Dostupné z: http://www.viscoda.com/index.php/en/products/non-commercial/voodoo-camera-tracker#usage

[11] VisualSFM: *GUI application for 3D reconstruction using structure from motion*, [online][cit. 2016-15-2], Dostupné z: http://ccwu.me/vsfm/.

[12] Yasutaka Furukawa: *Clustering Views for Multi-view Stereo*[online][cit. 2016-19-02]. University of Washington, 2010, Dostupné z:https://www.di.ens.fr/cmvs/

[13] Blender Foundation: *Blender*[online] [cit. 2016-3-11], Dostupné z: http://www.blender.org/

[14] Unity Technologies: *Unity3d*[online] [cit. 2016-12-10], Dostupné z: http://unity3d.com/

[15] ValveSoftware: *OpenVR documentation*[online][cit. 2017-15-03] Dostupné z: https://github.com/ValveSoftware/openvr/wiki/API-Documentation

[16] Bydlení iDnes.cz: *NTK* [fotografie]. Dostupné z: http://1gr.cz/fotky/idnes/10/063/cl6/REZ3400a2_1.jpg. Formát: 630x416

[17] RoadToVR: *HTC Vive review* [fotografie]. Dostupné z: http://www.roadtovr.com/htc-vive-review-room-scale-vr-mesmerising-vr-especially-if-you-have-the-space-steamvr/. Formát: 1500x900

Příloha **A**

Seznam použitých zkratek

AVI Audio video interleave

CVMS Clustering views for multi-view stereo

ČVUT České vysoké učení technické

IMU Inertial measurement unit

JPG Joint Photographic (Experts) Group

MP4 MPEG 4, Moving Picture Experts Group

MPEG Moving picture experts group

NTK Národní technická knihovna

SDK Software development kit

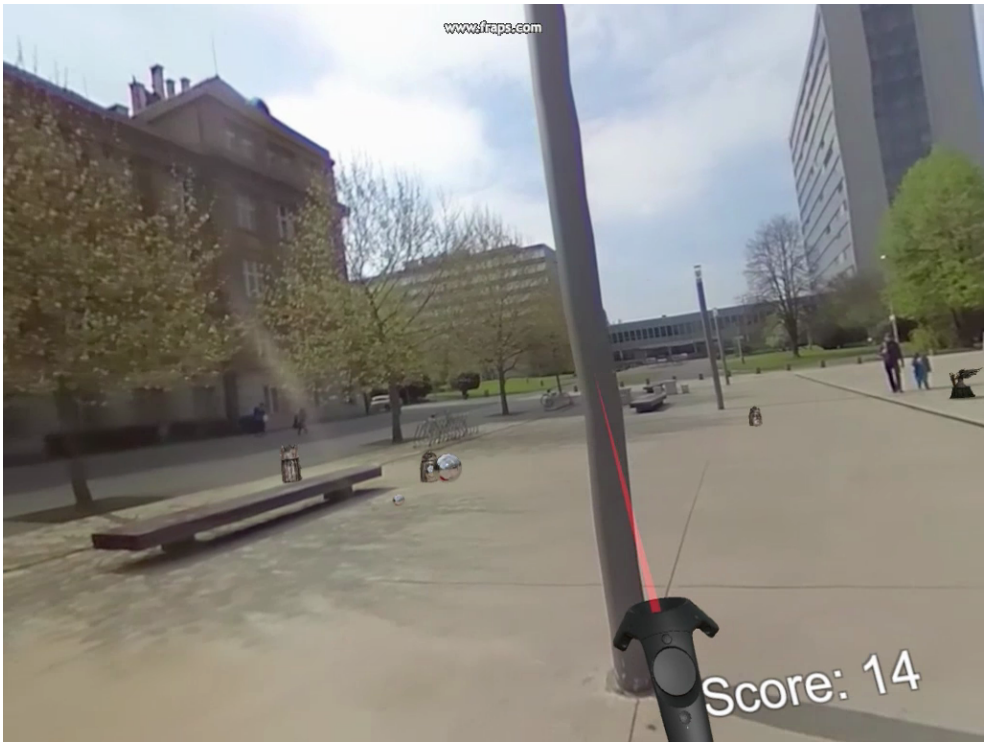
Příloha **B**

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `exe` ..... adresář se spustitelnou formou implementace
- `src`
  - `impl` ..... zdrojové kódy implementace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF

Příloha C

Ukázky interaktivního sférického videa



Obrázek C.1: Interakce před NTK a počítadlo zásahů

Obrázek C.2: Zásah lavičky

Obrázek C.3: Zásah osoby

Příloha D

Specifikace testovacích zařízení

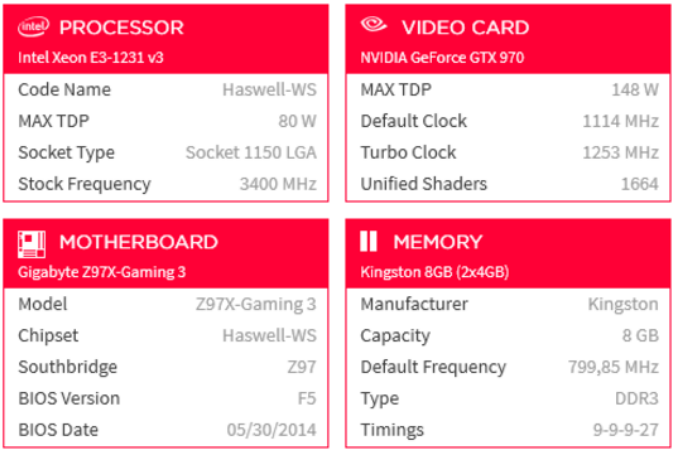
D.1 HTC Vive

Zařízení HTC Vive má OLED displej s rozlišením 2160 × 1200 s obnovovací frekvencí 90Hz. Úhel záběru je 110 stupňů. Maximální plocha pro snímání pohybu je 5x5 metrů a využívá akcelerometru, gyroskopu a laserovému systému trackování.

Obrázek D.1: HTC Vive

D.2 Stolní počítač

Počitač, na kterém byly prováděny rekonstrukce scény a vykreslování videí je postaven na procesoru Intel Xeon 1231v3 s 8 GB paměti, SSD diskem o kapacitě 256 GB a grafickou kartou nVidia gtx970. Stejný počítač byl použit pro renderování krátkého stereoskopického sférického filmu.



Obrázek D.2: Specifikace počítače