Západočeská univerzita v Plzni

Fakulta aplikovaných věd

Katedra informatiky a výpočetní techniky

Bakalářská práce

Subjektivní měření kvality trojúhelníkových sítí ve virtuální realitě

Plzeň, 2020 Jan Rach

Místo této strany bude zadání práce.

Prohlášení

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 12. července 2020 Jan Rach

Poděkování

Chtěl bych poděkovat panu Ing. Petru Vaněčkovi, Ph.D., vedoucímu mé práce za cenné rady při vypracování. Dále panu doc. Ing. Liborovi Vášovi, Ph.D., který mi práci zprostředkoval a poskytl původní program MeshTest. Na závěr chci poděkovat své rodině a přítelkyni za podporu při studiu.

Abstract

The goal of this thesis is to develop a program for subjective quality testing of triangle meshes in virtual reality with the possibility of interactions with the tested objects.

The first part sums up options for developing VR applications for devices that are present in the department.

The second part describes MeshTest application and contains basic information on triangle meshes and their subjective quality testing.

The third part deals with implementation of the software and methods that were used during the development.

In the end, a pilot study used to check functionality and useability of the application is discussed.

Abstrakt

Cílem této práce je vytvořit program pro subjektivní testování kvality trojúhelníkových sítí ve virtuální realitě s možností interakcí s testovanými objekty.

V první části jsou shrnuty možnosti tvorby VR aplikací pro zařízení, které jsou k dispozici na katedře.

Druhá část obsahuje popis původního programu MeshTest a také základní informace o trojúhelníkových sítích a jejich subjektivním testování kvality.

Ve třetí části je popsána implementace vytvořeného programu a postupy, které byly při jeho tvorbě použity.

V závěru je pojednáno o provedené pilotní studii, která slouží k ověření funkčnosti a použitelnosti programu.

Obsah

1. Úvod

Virtuální realita (VR) je technologie využívající počítačovou grafiku, která umožňuje uživateli ocitnout se v simulovaném prostředí, jenž může být naprosto odlišné od reálného světa [1]. To poskytuje tvůrcům obsahu obrovskou svobodu. V současné době je tato technologie na vzestupu a získává stále větší podíl na trhu [2].

Důležitou součástí trojrozměrné počítačové grafiky, a tedy i virtuální reality samotné, jsou takzvané polygonové (mnohoúhelníkové) meshe. Slovo *mesh* lze z angličtiny přeložit jako české *síť*. Účelem těchto mnohoúhelníkových sítí je reprezentovat v počítači objekt, který může být využit například při zobrazování simulovaného prostoru.

Mnohoúhelníkové sítě v počítačích mohou procházet různými procesy, které mají za úkol snížit jejich velikost nebo složitost [3]. Hodnocení jejich kvality po zpracování je obtížné, virtuální realita ale nabízí nebývalé možnosti, jak kvalitu zpracované sítě hodnotit subjektivně na základě zkoumání originálu a upravených verzí přímo ve virtuálním prostředí.

Cílem této bakalářské práce je seznámit se s příslušným hardwarem na *Západočeské univerzitě v Plzni* a také se subjektivním testováním trojúhelníkových sítí. Dále na základě získaných zkušeností vytvořit program pro testování těchto sítí ve virtuální realitě a otestovat jeho funkčnost a použitelnost.

2. Potřebná zařízení

Na *Katedře informatiky a výpočetní techniky Fakulty aplikovaných věd Západočeské univerzity v Plzni* je k dispozici laboratoř virtuální reality, která obsahuje množství specializovaného vybavení. To lze využít k tvorbě software pro testování kvality trojúhelníkových sítí ve VR.

Potřebnými zařízeními jsou zejména takzvané VR headsety [4, 6]. Jde o zařízení, které dokáže s pomocí dostatečně výkonného počítače s podporou 3D akcelerace simulovat virtuální prostředí okolo svého uživatele, který má tento headset nasazen na své hlavě. U velkého množství současných headsetů probíhá výpočet obrazu v samostatném připojeném počítači a výsledek je přenášen, buď přímo po kabelu nebo bezdrátově, do speciálních displejů v přední části headsetu. Druhou možností je vsadit dostatečně výkonnou výpočetní jednotku přímo do samotného zařízení.

Displeje jsou v zařízení zpravidla dva, jeden pro každé oko a jejich obraz je navzájem mírně posunutý. Tím je docíleno iluze prostorového vidění (*stereopse*). [5] Před samotným displejem se také nachází čočka, díky které se obraz jeví uživateli dostatečně vzdálený, aby na něj mohl zaostřit svůj zrak.

Pohyb uživatele ve virtuálním prostředí je možné realizovat více možnostmi. Balení headsetů často obsahují systém takzvaných základních stanic (*base stations*), které pomocí bezdrátových signálů (například infračervené záření) určují aktuální polohu zařízení v prostoru. Jiné výrobky mají na své přední straně umístěné kamery, které snímají reálný prostor v místnosti a pomocí počítačového rozpoznávání obrazu vyhodnocují okolí uživatele. Jde o modernější řešení, které využívá například zařízení *Oculus Quest* od společnosti *Oculus VR* [7].

Kromě přenosu reálného pohybu do virtuálního prostředí je většinou integrován také pohyb pomocí ovladačů, které uživatel drží ve svých rukou.

Tato zařízení obsahují různorodé ovládací prvky (joystick, směrová tlačítka, ...), která umožňují snadný pohyb virtuálním prostorem. Kromě motorických funkcí umožňují pomocí svých ovládacích prvků i množství jiných interakcí, které jsou kompletně v kompetenci vývojářů samotných aplikací.

V současné době je také možné snímat pohyb rukou pomocí speciálních rukavic nebo přímo vyhodnocovat jejich reálný obraz snímaný kamerou na přední straně headsetu. V současnosti jsou na trhu například zařízení od britské firmy *Ultraleap*.[8]

3. Zařízení pro VR v laboratoři

3.1 HTC Vive

Obrázek 2.1: HTC Vive, Zdroj:
https://www.whaat.cz/htc-vive-virtual-reality-headset-2x-motion-controller-and-2x-lighthouse

Zařízení od firmy *HTC* je v laboratoři dostupné v úpravě, která umožňuje jeho použití bez přenosového a napájecího kabelu díky bezdrátovému přenosu. Na vrchní straně je napojena vysílací stanice, která je prodávána jako samostatný prvek, není tedy obsažena v základním balení [9]. Baterie je umístěna za pasem uživatele a připojena pomocí krátkého kabelu. Díky tomuto řešení je značně zvýšena pohyblivost nositele headsetu [10].

Sledování polohy je řešeno pomocí dvou základních stanic rozmístěných po místnosti. Zařízení obsahuje i dva ovladače s ovládacími prvky pro pohyb pomocí joysticku a jiné interakce s virtuálním prostředím. *OLED* displeje headsetu mají rozlišení 2160x1200 obrazových bodů (1080x1200 pro každé oko) a jejich obnovovací frekvence je 90 Hz [11].

3.2 Valve Index

Obrázek 2.2: Valve Index, Zdroj:
https://sea.ign.com/valve-index/151256/review/valve-index-review

Valve Index je headset vydaný firmou *Valve* v roce 2019. Disponuje větším rozlišením oproti *HTC Vive*. Jeho *LCD* displeje mají rozlišení 2880x1600 (1440x1600 pro každé oko). Podporují obnovovací frekvenci o hodnotě 80, 90, 120 a 144 Hz.[12]

Příslušenstvím jsou speciální ergonomické ovladače nazvané "*Knuckles controllers*". Jsou navrženy s ohledem na anatomii ruky a mají tak umožnit uživateli snažší ovládání. Headset samotný je ale kompatibilní také s ovladači pro *HTC Vive*.

Obrázek 2.3: Knuckles controllers, Zdroj:
https://uploadvr.com/valve-reveals-news-knuckles-vr-controllers-with-improved-battery-and-more/

Sledování je vyřešeno pomocí dvou základních stanic.

3.3 Oculus Rift CV1

Obrázek 2.4: Oculus Rift CV1, Zdroj:
https://en.wikipedia.org/wiki/File:Oculus-Rift-CV1-Headset-Front.jpg

Z důvodu opatření proti šíření nového typu koronaviru v roce 2020 mi byl z laboratoře zapůjčen starší headset Oculus Rift CV1 od firmy Oculus VR pro práci v domácím prostředí.

Tento headset disponuje OLED displeji s rozlišením shodným s HTC Vive. Obnovovací frekvence je též 90 Hz. Součástí balení jsou také dvě základní stanice pro sledování určené k postavení na stůl a dva ovladače *Oculus Touch*[13].

Obrázek 2.5: Oculus Touch, Zdroj:
https://uk.webuy.com/product-detail/?id=svroculriftoucconta

4. Možnosti tvorby aplikací pro VR

4.1 Game engine obecně

Při tvorbě aplikace pro virtuální realitu je možné vytvořit od základů vlastní řešení nebo použít už hotový tzv. *game engine* [14] poskytovaný třetí stranou. První možnost je velmi časově i technicky náročná, proto se většinou jednotlivci a menší týmy rozhodnou využít už připravená řešení. To ušetří spoustu času, který může být věnován vývoji samotné aplikace. Nevýhodou je často závislost na třetí straně, povinnost respektovat dané licenční ujednání a někdy také nemožnost implementovat komplexnější nové vlastnosti.

Mezi nejpoužívanější enginy v současnosti patří *Unity* od *Unity Technologies* a *Unreal Engine 4* od *Epic Games* [15]. Existuje i velké množství dalších *enginů*, ale oproti dvou uvedeným disponují mezi studenty a nezávislými tvůrci mnohem menší uživatelskou základnou. Za zmínku stojí například *Lumberyard* [16] od společnosti *Amazon*, který je postaven na enginu *CryEngine* [17] od firmy *Crytek*.

Pro realizaci této bakalářské práce jsem vybral *engine Unity*. Dále budou zhodnoceny klady a zápory nástrojů *Unity* a *Unreal Engine 4*, včetně zdůvodnění mé volby.

4.2 Unity Engine

Engine *Unity* od firmy *Unity Technologies* patří mezi nejpopulárnější nástroje pro tvorbu interaktivních 2D a 3D aplikací na trhu [15]. Za jeho rozšířením stojí kromě uživatelské přívětivosti také výhodná licence pro menší studia. Ta umožňuje vydat svůj produkt komerčně bez nutnosti investovat předem peníze do placené licence [18].

Engine samotný je napsaný v *C++* a uživatelské skriptování samotných aplikací se provádí pomocí jazyka *C#*.

Software obsahuje velkou sadu nástrojů pro tvorbu vlastní aplikace. Kromě *editoru*, který je centrem celého vývoje, implementuje například také editor shaderů založený na spojování bloků (*Shader Graph*) [19], částicový systém pro simulaci částicových efektů a další [20].

Vývoj pro virtuální realitu je v *Unity enginu* uživatelsky přívětivý. Nástroj podporuje platformu *SteamVR* [21] implementující SDK *OpenVR* [22], dvě řešení pocházející od firmy *Valve*. Díky tomu je velmi jednoduché používat jak headset *HTC Vive*, tak *Valve Index* a *Oculus Rift* přímo v reálném čase při samotném vývoji. Přechod mezi jednotlivými VR zařízeními je z hlediska vývojáře vyvíjejícího v *Unity* s pomocí platformy *SteamVR* realizováno jednoduchým nastavením ovládání pro každý typ ovladače samostatně.

4.3 Unreal Engine 4

Unreal Engine 4 je čtvrtou generací *enginu* od společnosti *Epic Games*. Jde už o druhou verzi, která byla uvolněna pro použití širokou veřejností. Předchozí vydaná verze, nazvaná *Unreal Development Kit* [52] byla určena spíše pro tvorbu videoher s pohledem z první osoby. Současná verze už je pohodlně využitelná pro jakýkoliv typ interaktivních 2D a 3D aplikací včetně virtuální reality [23].

Skriptování je možné provádět buď přímo pomocí kódu v *C++*, nebo je možné využít speciální vizuální skriptovací nástroj nazvaný *Blueprints Visual Scripting* [24]. Kód je zde nahrazen spojováním jednotlivých bloků, které reprezentují proměnné a příkazy. Jde tedy o jakési zjednodušení klasického textového programování, která má umožnit tvorbu logiky aplikace i osobám bez větších programátorských znalostí.

Plugin pro SteamVR je k tomuto enginu taktéž dostupný [53].

Unreal Engine 4 implementuje obsáhlou sadu nástrojů, která je srovnatelná s *Unity enginem*. Nevýhodou je vyšší složitost práce se softwarem. Z toho důvodu je podle mého názoru tento engine vhodnější pro práci většího a zkušenějšího týmu.

4.4 Vlastní volba enginu pro bakalářskou práci

Engine *Unity* je možné považovat za snažší pro práci jednotlivce.. Také skriptování pomocí jazyka *C#* je pro mě obecně přístupnější, než možnosti, které nabízí *Unreal Engine*. Ačkoliv byl systém *Blueprints Visual Scripting* navržen z důvodu, aby i uživatelé bez zkušeností s programováním byli schopni vytvořit vlastní aplikační logiku, jde o poměrně komplexní systém, do jehož pochopení je nutné investovat značné množství času. Z těchto důvodů a také kvůli svým dosavadním zkušenostem jsem tedy zvolil engine *Unity* verze 2019.

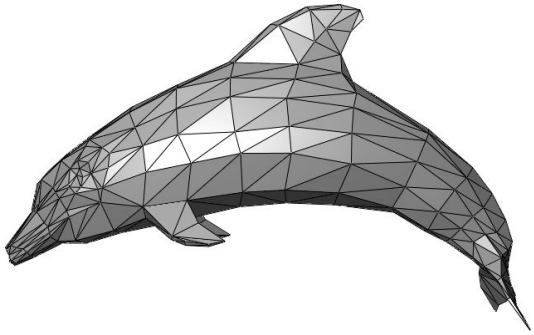
5. Rozbor software MeshTest

5.1 Software MeshTest

Software *MeshTest* je výsledkem práce na *Katedře informatiky a výpočetní techniky Fakulty aplikovaných věd ZČU*. Účelem je subjektivní testování kvality trojúhelníkových sítí, které prošly určitým algoritmem za účelem komprese nebo celkového snížení složitosti dané sítě. Program mi byl poskytnut pomocí sdíleného adresáře na *Gapps Zču* panem doc. Ing. Liborem Vášou, Ph.D., ale je k dispozici ke stažení i na webových stránkách *KIV ZČU* [25].

5.2 Polygonové sítě

Polygonová síť (*polygon mesh*) je virtuální konstrukce skládající se ze tří základních prvků: vrcholů (*vertices*), hran mezi nimi (*edges*) a ploch (*faces*), které tvoří mnohostěny [26]. Účelem je, jak už bylo řečeno, reprezentace trojrozměrných objektů v počítačích. Pokud jsou v množině všech ploch sítě zastoupeny pouze trojúhelníky, jde o trojúhelníkovou síť [27]. Obecně se v polygonové síti mohou kromě trojúhelníků vyskytovat i čtyřúhelníky nebo další jednoduché konvexní mnohoúhelníky [26].

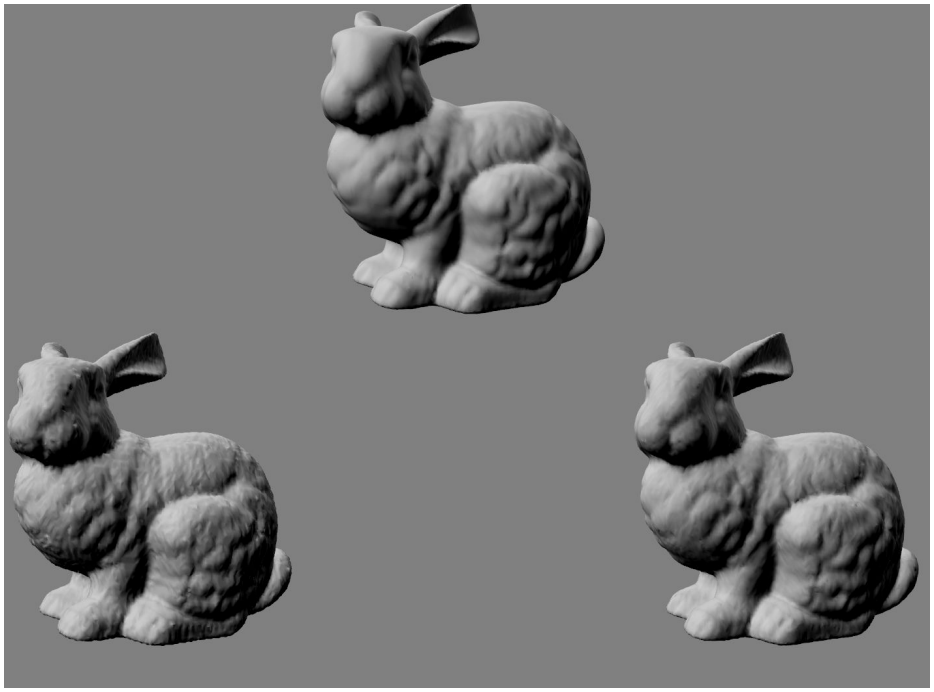


Obrázek 5.1: Trojúhelníková síť, Zdroj:
https://en.wikipedia.org/wiki/File:Dolphin_triangle_mesh.png

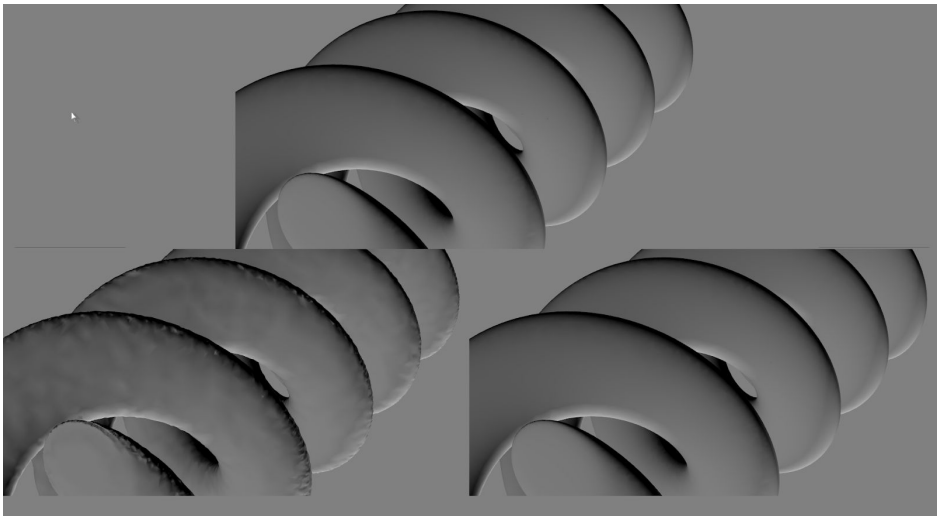
Tvorba těchto sítí probíhá například pomocí specializovaného software pro 3D modelování (*Blender* [28], *3Ds Max* [29], ...) nebo v poslední době často také využitím takzvané *fotogrammetrie* [30], kdy jsou pomocí kamery snímány reálné objekty z různých úhlů. Z fotografií je poté vygenerována polygonová síť reprezentující daný objekt. Výsledný produkt je často nutné dočistit a upravit ve zmíněných aplikacích pro 3D modelování.

Pro uložení trojúhelníkových sítí se využívá velké množství různých formátů souborů. Z těch nejčastějších jmenujme například formát *Filmbox* (*FBX*) [31] od firmy *Autodesk* nebo *OBJ* od *Wavefront Technologies* [32].

Čím větší množství jednotlivých stavebních elementů (*vertices, edges, faces*), daný mesh obsahuje, tím narůstá velikost příslušného souboru nebo velikost sítě v paměti počítače. Řešením je použití kompresních a zjednodušujících algoritmů, které sníží počet jednotlivých stavebních prvků a tím tedy i velikost souboru. Tímto procesem ale velmi rychle dochází ke snižování pozorované kvality samotné sítě. Cílem je tedy použít takový algoritmus (nebo jeho konfiguraci), který daný model zachová subjektivně co nejpodobnější originálu a stále dostatečně kvalitní při požadovaném snížení velikosti a složitosti sítě.



Obrázek 5.2: Poškozené trojúhelníkové sítě v programu MeshTest[25], nahoře originál, vespod poškozené varianty



Obrázek 5.3: Další poškozené trojúhelníkové sítě v programu MeshTest[25], nahoře originál, vespod poškozené varianty

Tento proces je v praxi velmi důležitý například při tvorbě a zobrazování rozsáhlých virtuálních 3D prostředí. Zde je vhodné mít komplexní geometrii uloženou ve více verzích podle úrovně detailů. Nejvíce detailní model, jeho největší LOD (*level of detail*), je použitý pokud se při vykreslování nachází blízko kamery. Naopak ta nejméně detailní varianta je využita, pokud se objekt vyskytuje ve velké vzdálenosti od kamery [33].

Model po kompresi je tedy možné a vhodné posuzovat subjektivně. Takovéto hodnocení je právě účelem programu MeshTest. Jde o desktopovou aplikaci pro *Windows*, která umožnila porovnávání komprimovaných a zkreslených 3D modelů s originálem a jejich hodnocení, respektive vybírání subjektivně podobnější komprimované sítě.

Jedním z úkolů této bakalářské práce je implementovat podobný software, ale tentokrát pro virtuální realitu.

5.3 Subjektivní hodnocení kvality trojúhelníkových sítí [34]*

Součástí zásad pro vypracování této bakalářské práce je seznámení se s metodami subjektivního testování kvality trojúhelníkových sítí. Informace pro tuto podkapitolu jsem čerpal zejména z článku *Perceptual Metrics for Static and Dynamic Triangle Meshes* [34]. Vzhledem k tomu, že jeden z autorů (pan doc. Ing. Libor Váša, Ph.D.) mi práci zprostředkoval, považuji ji za relevantní a pro mě nejlepší zdroj informací k tématu, které pro mě bylo kompletně nové. Tato kapitola tedy slouží jako shrnutí informací, které jsem díky seznámení se s touto problematikou získal.

Jak je uvedeno v daném článku, metody subjektivního hodnocení kvality pro statické *meshe* jsou velmi důležité například u již zmíněné komprese. Dají se rozdělit na metody založené na obrazu (anglicky *image-based*) a založené na modelu (anglicky *model-based*).

* Kapitola vychází z citovaného článku.

5.3.1 Image-based

Tyto metriky jsou založeny na lidském vnímání aplikovaném na statický 2D obraz hodnoceného *meshe*. Součástí mohou být i přímé uživatelské interakce s hodnoceným objektem (např. pohyb, rotace). Metriky mohou být využity například při snaze vybírat pro model vhodný LOD (viz. výše).

5.3.2 Model-based

Zde uživatel hodnotí přímo geometrii daného meshe, nesleduje tedy pouze jeho statický obraz jako u předchozího typu metrik. V článku jsou popsány například metriky založené na křivkách a nerovnostech povrchu nebo deformace oproti originálu.

5.3.3 Dynamické meshe

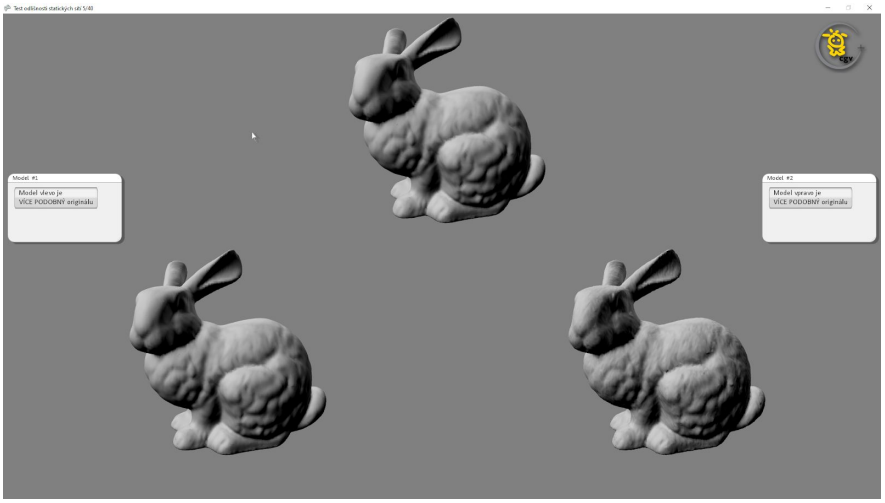
Kromě statických meshů může být samozřejmě hodnocena i kvalita dynamických (animovaných) meshů. Tím jsem se však v této bakalářské práci nezabýval.

5.4 Funkce původního programu MeshTest

Po spuštění programu je uživateli nejdříve nabídnuta krátká výuková sekvence, která kompletně popisuje ovládání aplikace a postup testování. Zahrnuty jsou dokonce i interaktivní ukázky manipulace s modely. Uživatel také musí zvolit své pohlaví. Po těchto náležitostech následuje samotné hodnocení trojúhelníkových sítí.

Obrázek 5.4: Úvod v programu MeshTest

Uživatel na obrazovce vidí trojici meshů. Uprostřed se nachází originál a po stranách jeho dvě poškozené verze. Každá z nich prošla určitým algoritmem zpracování (uživateli není poskytnuta informace kterým). Úkolem je zvolit ten model, který je podobnější originálu. Nejde tedy o volbu kvalitnějšího meshe, ale přímo o volbu věrnější kopie.



Obrázek 5.5: Testování v programu MeshTest

Mezi implementované interakce s objekty patří možnost rotovat s každým objektem, kdy zbylé dva tento rotační pohyb kopírují, zmenšování/zvětšování všech modelů a také jejich posun. Poměrně skrytá je možnost změnit pozici osvětlení.

Výsledky aktuálně proběhlého testování lze nalézt ve složce programu v souboru *data.xml*.

6. Návrh software MeshTest ve VR

Nabízí se tedy možnost vytvořit novou verzi programu *MeshTest*. Tentokrát ale v podobě interaktivní aplikace pro virtuální realitu. Uživatel procházející testem nebude hodnocený model sledovat pouze na obrazovce počítače, ale díky virtuální realitě objekt uvidí přímo před sebou a bude kolem něj moci libovolně přecházet, manipulovat s ním a porovnávat ho se vzorem. Limit pro pohyb zde tedy představuje hlavně reálné prostředí.

Jednotlivé modely jsou poskytnuty *Katedrou informatiky a výpočetní techniky* v *DAT* formátu. Postup jeho zpracování bude popsán dále.

Pro realizaci této aplikace pro virtuální realitu je nutné nejdříve vytvořit systém, který dokáže načíst proprietární formát, ve kterém jsou uloženy trojúhelníkové sítě pro hodnocení. *Unity* implicitně neumožňuje načítání 3D meshů z externích souborů při běhu aplikace, proto je nutné takovou funkcionalitu vytvořit. V *Unity API* (aplikační rozhraní) jsou ve třídě *Mesh* zastoupeny metody určené k tvorbě trojúhelníkových sítí v reálném čase [36]. Stačí tedy načíst ze souboru údaje o jednotlivých stavebních prvcích a postupně z nich sestavit žádaný model.

Metodika testování byla po konzultaci zvolena shodná jako v původním *MeshTestu*. Po spuštění a absolvování krátkého úvodu s popisem ovládání a volby pohlaví budou ve virtuálním prostoru před uživatelem zobrazeny tři trojrozměrné modely určené k hodnocení. Uprostřed se bude nacházet originál. Vlevo a vpravo budou stejně jako v původním *MeshTestu* dvě poškozené verze. Na konci testování budou výsledky vypsány do souboru.

Pro možnost kvalitního hodnocení modelů je nutné implementovat interakce uživatele. Za první z nich byla zvolena rotace objektu okolo určeného středu. Ta bude realizována pomocí "laserového" ukazatele směřujícího z ovladače. Uživatel tedy namíří ukazatel na objekt, kterým chce rotovat a pokud stiskne

určené interakční tlačítko, bude model rotovat okolo středu v závislosti na pohybu ovladačem. Současně se budou stejně otáčet i ostatní trojúhelníkové sítě, aby je uživatel mohl snadno porovnávat a hodnotit. Každý model zachová vhodný úhel vůči pozici ve středu scény.

Pro realizaci rotace bude nutné implementovat systém, který pro danou trojúhelníkovou síť určí vhodný bod, okolo kterého rotace objektu proběhne. Za něj bylo zvolené centrum takzvaného "bounding boxu [37]". Jde o střed kvádru, který přesně obaluje daný model.

Kromě rotace umožní aplikace i posun všech objektů od a k uživateli. Tato funkce byla zvolena jako náhrada zvětšování a zmenšování, které bylo zastoupeno v původním *MeshTestu*. S modely bude možné také manipulovat a posouvat je do jiných pozic pomocí ovladače a jeho “grip” tlačítka na držadle.

Vedle modelů budou ve virtuálním prostředí reprezentována i dvě tlačítka pro volbu výsledků testování. Uživatel tedy pomocí zmíněného ukazatele vybere, který ze dvou prezentovaných modelů je podobnější originálu, stejně jako v původním software.

Po samotné implementaci software bude provedena takzvaná pilotní studie, kdy bude určitý počet uživatelů procházet testováním. Kromě samotného porovnávání 3D modelů budou ale hodnotit i samotný software. Na základě odezvy budou poté provedeny potřebné změny, pokud budou potřeba.

Vzhledem k návaznosti na původní software jsem se rozhodl program pojmenovat MeshTestVR.

7. Implementace software

7.1 Úvod k implementaci

Implementace samotné logiky software je provedena v C# skriptech [38]. Za dva hlavní řídící objekty aplikace lze považovat instance tříd `SceneController` a `UIController`. Ty jsou obě dvě implementovány podle návrhového vzoru *singleton*. Jeho podstatou je, že instance dané třídy zde existuje pouze jedna [39].

Neuvádím zde všechny skripty, které byly při tvorbě programu vytvořeny. Kompletní zdrojový kód s komentáři je k dispozici na přiloženém DVD v příslušném *Unity* projektu.

7.2 SteamVR plugin

Pro podporu VR zařízení jsem využil *Unity SteamVR* plugin od firmy *Valve*, který je k dispozici ke stažení z *Unity Asset Store* [40], což je internetový portál obsahující velké množství obsahu pro okamžité použití přímo v *Unity*. Můžeme zde nalézt jak placený, tak i zdarma přístupný obsah. *SteamVR* plugin je bez poplatku přístupný na této adrese: https://assetstore.unity.com/packages/tools/integration/steamvr-plugin-32647.

Pro realizaci pohybu a ovládání aplikace jsem využil *prefab* [41] `Player` poskytovaný jako součást zmíněného pluginu.

7.3 Podporovaná zařízení

Program je vyvinut pro tři již zmíněné headsety v laboratoři: *HTC Vive*, *Valve Index* a *Oculus Rift CV1*. Podporovaným operačním systémem je *Windows 10*, který je na tamějších počítačích nainstalován.

7.4 Součásti programu

7.4.1 Třída SceneController

Tato třída je zodpovědná za hlavní akce probíhající v testovací scéně. Nalezneme zde například metody pro výběr a načtení modelů, které dále volají metody tříd, jenž obsahují podrobnější implementaci těchto funkcí. Načítání modelů je zahájeno spuštěním koprogramu (anglicky *coroutine* [42]) ze třídy `ModelLoader`.

7.4.2 Třída UIController

Jde o třídu, která má na starosti správu reakcí na uživatelské vstupy v *UI* aplikace. Při startu programu jsou zde přiřazeny jednotlivé reakce na události laserového ukazovátka ovladače. Odkazuje pak na samotnou instanci třídy `SceneController`, která obsahuje podrobnější implementaci příslušných reakcí.

7.4.3 Načítání meshů v reálném čase pomocí ModelLoader

Úkolem třídy `ModelLoader` je načítání modelů v reálném čase z poskytnutého binárního souboru. Algoritmus vychází z podobného v původním *MeshTestu* [25]. Pro tvorbu modelu při runtime aplikace je využita již zmíněná třída *Mesh* z *UnityAPI*, která umožňuje tvorbu a editaci meshů přímo ze skriptů. Soubor je čten pomocí třídy BinaryReader [43]. Jednotlivé byty jsou čteny jako celá čísla o velikosti 4 byty (Integer 32 bit) v následujícím pořadí:

načtení počtu trojúhelníků - 1x Int32 = 4 byty

cyklus pro načtení indexů vertexů jednotlivých trojúhelníků:
počet iterací cyklu = počet trojúhelníků

v každé iteraci se načtou tři indexy vertexů a umístí se do pole v odpovídajícím pořadí, tím se připraví data pro inicializaci odpovídajícího trojúhelníku - 3 x Int32 = 12 bytů

načtení počtu vertexů v meshi - 1 x Int32 = 4 byty

cyklus pro načtení souřadnic jednotlivých vertexů a normál:
počet iterací cyklu = počet vertexů

v každé iteraci se načtou dva trojrozměrné vektory, první obsahuje souřadnice daného vertexu, druhý odpovídající normály - 6 x Int32 = 24 bytů

Výsledkem jsou pole obsahující data jednotlivých vertexů, trojúhelníků a normál. Čtení každého souboru a zpracování jeho dat probíhá v samostatném vlákně, aby nedocházelo k nepříjemnému zamrzání programu.

Načtená data jsou přiřazeny jako atributy instanci třídy *Mesh*. Ta je přidána ke komponentě *MeshFilter* [44] jako atribut *mesh*. O samotné zobrazení se pak postará komponenta MeshRenderer [45].

7.4.4 Škálování meshů

Při posuzování jednotlivých meshů chceme, aby měly všechny vždy stejnou velikost bez ohledu na to, o který model se jedná. O to se stará metoda `ScaleModel` obsažená ve třídě `ModelLoader`. Princip algoritmu jsem čerpal z webu *Stack Overflow*[46]. Postup je následující: Pomocí *Unity API* se zjistí rozměry daného meshe (opět se využívá bounding box) [37, 54].

Vypočítá se trojrozměrný vektor, který by pro daný model odpovídal v Unity velikosti 1 a na ten se aplikuje metoda `Scale` [55], díky které získáme nový vektor odpovídající požadované velikosti, který můžeme přiřadit objektu.

7.4.5 Vycentrování pivotu

Aby bylo možné otáčet mesh okolo středu jeho *bounding boxu*, je nutné ho centrovat (zarovnat na střed) [47]. Byl zvolen postup, kdy se objektu obsahující komponentu *MeshRenderer* [45] daného meshe stanoví v hiearchii *Unity* scény rodič. Objekt s meshem se posune relativně oproti svému rodiči tak, aby pozice rodiče ležela v místě odpovídající požadované pozici pivotu. Poté už je možné objekt otáčet okolo rodiče pomocí metody *RotateAround* z *Unity API* [48].

7.4.6 Třída MoveObjectsFartherAndCloser

Slouží k pohybu s *meshy* směrem od centra scény a zpátky k němu. Směr je zvolen s ohledem na rotaci objektu, aby docházelo k oddálení a přiblížení objektu přesně po ose směřující od centra skrz daný objekt. Tím je docíleno toho, že při stisknutí příslušného tlačítka (viz. uživatelská příručka) se objekty začnou od uživatele stojícího v centru vzdalovat, aniž by docházelo ke změně úhlu, pod kterým jsou modely vidět. Akce tlačítka pro pohyb *meshů* k uživateli je analogická. Pozice objektů je změněna pomocí následujícího kódu:

```
for (int i = 0; i < objectsToMove.Length; i++)
            {
            if (moveObjectsCloser.state == true &&
            Vector3.Distance(objectsToMove[i].transform.position, center) <=
            distanceLimit)
                {
                    break;
                }
```

```
Vector3 direction = objectsToMove[i].transform.position -
center;
direction = direction.normalized;

objectsToMove[i].transform.position += (direction * sign *
movementSpeed * Time.deltaTime);
}
```

7.4.7 Třída ManipulateObjects

Manipulace s objekty v prostoru pravým ovladačem je implementována právě v této třídě. Při uchopení modelu pomocí tlačítka “grip” je uložena pozice ovladače pomocí metody `SaveReference()`. Při výpočtu dalšího snímku je volán následující kód:

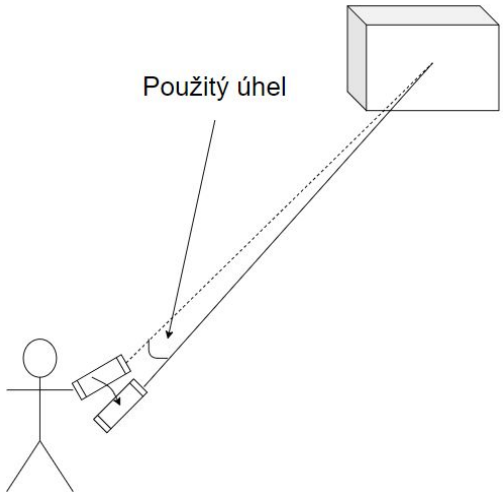
```
if (manipulationEnabled == true)
{
    Vector3 difference = this.transform.position - reference;
    SaveReference();
    manipulatedTransform.parent.position += difference;
}
```

Je spočítán rozdíl (`difference`) vektorů nové a staré pozice ovladače. Ten je aplikován na pozici manipulovaného objektu. Samozřejmě je opět pomocí metody `SaveReference()` uložena aktuální poloha ovladače pro výpočet posunu modelu u dalšího snímku.

7.4.8 Rotace objektů

Pro realizaci rotace objektů byl zvolen postup, kdy při výpočtu každého snímku, kdy je rotace aktivována, určíme rotaci abstraktních objektů, které by směřovali přímo ze středu modelu směrem k příslušné poloze VR ovladače. Spočítáme rozdíl rotací na všech jejich osách a výsledný zbytek aplikujeme na rotovaný model. Při vynásobení určitým násobkem docílíme přijatelné úhlové rychlosti rotace a také ji můžeme regulovat na základě vzdálenosti ovladače od

modelu, aby nedocházelo ke zvýšení rychlosti po přiblížení modelu a k opačnému jevu po jeho vzdálení.



Obrázek 7.1: Úhel použitý k určení rotace modelu

Obrázek 7.2: Modely rotují směrem k uživateli

7.5 Uživatelské rozhraní

7.5.1 Výběr možností

Pro výběr subjektivně věrnější varianty *meshe* uživatelem bylo zvoleno grafické uživatelské rozhraní umístěné přímo v simulovaném 3D prostoru aplikace. Má podobu dvou velkých modrých tlačítek vlevo a vpravo od testovaných *meshů*. Pro tvorbu bylo využito *Unity Canvas*, což je oblast, která slouží k umístění objektů uživatelského rozhraní. [49]. Tlačítka jsou realizována pomocí třídy Button [50] z *UnityEngine* a balíčku *TextMeshPro* [56].

7.5.2 Tutorial

Po spuštění programu je uživateli zobrazen krátký úvod, který ho seznámí s funkcemi aplikace a metodikou testování. Cílem je, aby se i uživatel, který je obecně virtuální reality neznalý, mohl zúčastnit testování a dokázal bez větších problémů program ovládat.

Podobně jako v původním *MeshTestu* jsem zvolil grafické uživatelské rozhraní zobrazující instruktážní text a po pilotní studii přidal i interaktivní ukázky, díky kterým má uživatel možnost si interakce s modely vyzkoušet ještě před samotným testováním. Text tohoto tutoriálu je založen na textu v původním programu [25], aby došlo ke srovnatelné edukaci testujícího uživatele.

Implementace logiky tutoriálu je provedena ve třídě `TutorialController`. Řídí ji zde jednoduchý čítač `instructionsIndex`.

7.6 Zpracování výsledků

Výsledky samotného testování jsou tisknuty do souboru ve formátu *XML* pomocí třídy `ResultsWriter`. Při konzultaci s vedoucím této bakalářské práce bylo dohodnuto, že jeho formát bude shodný s formátem výsledků z původního *Meshtestu* [25].

Formát tohoto XML souboru je následující:

```
<SubjectiveTask>
  <Task>
    <Original>./meshes\nissan\original.dat</Original>
    <Left>./meshes\nissan\v02.dat</Left>
    <Right>./meshes\nissan\v05.dat</Right>
    <Choice>Left</Choice>
  </Task>
  .
  .
  <Task>
    <Original>./meshes\jessi\original.dat</Original>
    <Left>./meshes\jessi\v05.dat</Left>
    <Right>./meshes\jessi\v02.dat</Right>
    <Choice>Left</Choice>
  </Task>
  <Sex>M</Sex>
  <Duration>12055,8</Duration>
  <Date>5 6 2020</Date>
</SubjectiveTask>
```

Jsou tedy uložena data o každém testovaném objektu, jeho poškozených variantách, uživatelské volbě podobnějšího modelu a na konci základní data o testování: tedy pohlaví příslušné osoby, uběhlý čas v milisekundách a datum.

Soubor je pojmenován názvem "*results"* doplněným o časové razítko v dvojciferném formátu:

"_den_měsíc_rok_hodina_minuta_vteřina.xml".

Důvodem je jednoznačné odlišení jednotlivých souborů s výsledky.

8. Pilotní studie

Účelem takzvané pilotní studie pro otestování programu je ověřit zejména použitelnost a uživatelskou přívětivost programu. Nejde tedy o vyhodnocení výsledků samotného subjektivního testování kvality trojúhelníkových sítí ve virtuální realitě. To bude možné provádět na *Katedře informatiky a výpočetní techniky* po kompletním uvolnění opatření proti šířené koronaviru. Testování by se mohli zůčastnit například studenti předmětů zaměřených na počítačovou grafiku.

8.1 Metodika studie

Pro provedení pilotní studie bylo zvoleno uživatelské testování spojené s vyplněním dotazníku. Ten je přiložen jako příloha bakalářské práce. Dotazník jsem vždy uživatele nechal podepsat nebo označit, abychom se k němu mohli vrátit při druhém kole studie po upravení programu. Kromě testovaného programu byl uživatelům předložen i původní *MeshTest* k porovnání.

Z důvodu opatření proti šíření nového typu koronaviru v roce 2020 [51] byla pilotní studie provedena v domácím prostředí na zapůjčeném zařízení *Oculus Rift* v omezenějším počtu testujících uživatelů.

8.2 Výsledky studie

Program MeshTestVR v pilotní studii postupně otestovalo 12 uživatelů. Zde se postupně shrnu odpovědi na otázky z dotazníku.

8.2.1 Celkový dojem z programu

U první otázky jsem obdržel velmi pozitivní ohlasy. Pro některé uživatele šlo o jedno z prvních setkání s virtuální realitou, dá se tedy říci, že byli ohromeni zobrazením virtuálního prostoru a možností se v něm volně pohybovat. Je důležité zdůraznit, že ani jeden z uživatelů, které jsem v domácím prostředí pro testování sehnal, nebyl odborník na virtuální realitu ani programátor, který by podle mého názoru dokázal být k aplikaci kritičtější.

8.2.2 Vysvětlení postupu testování kvality meshů

U této otázky se u uživatelů objevily k programu výhrady. Nelíbilo se jim, že v původním *MeshTestu* je na začátku zobrazen textový popis testování i s interaktivními ukázkami a v mé aplikaci tato funkčnost chybí. Rozhodl jsem se tedy tuto část do programu doplnit.

Dále se objevily stížnosti, že uživatelé netuší, kde jsou umístěna příslušná tlačítka na ovladačích a vyhovovalo by jim vidět přímo ve scéně nápovědu se schématem ovladače a popisem jednotlivých tlačítek.

8.2.3 Manipulace s objekty

Na manipulaci s objekty si nikdo v dotazníku nestěžoval. Uživatelům se líbilo, že mohou objekty libovolně otáčet, posouvat a manipulovat s nimy pomocí ovladačů.

8.2.4 Rozmístění objektů

Někteří uživatelé si v odpovědi na tuto otázku stěžovali, že jsou hodnocené objekty příliš velké. Zde je důležité zdůraznit, že při testování v domácím prostředí nebylo k dispozici tolik místa, jako je v laboratoři VR na *Západočeské univerzitě v Plzni*. Také jsem zde byl omezen dosahem základních stanic samotného zařízení *Oculus Rift*. Rozhodl jsem se tedy velikost objektů mírně snížit a posunout je blíže k sobě, abych dosáhl jistého kompromisu.

8.2.5 Osvětlení a pozadí scény

Ohledně osvětlení a pozadí scény jsem nezaznamenal negativní reakce. Uživatelé kladně hodnotili světla jako nerušivá. Šedé pozadí scény také obdrželo pozitivní hodnocení s odůvodněním, že nedráždí zrak. To považuji za poměrně podstatný bod, protože podle mých zkušeností si uživatelé, kteří nepoužívají virtuální realitu často, občas na bolest očí stěžují.

8.2.6 Kvalita uživatelského rozhraní

Zde jsem zaznamenal nejvíce kritické reakce. Uživatelům se grafika rozhraní příliš nelíbila. Podle jejich hodnocení jsou elementy tvořící pozadí jednotlivých oken ve scéně příliš rozmazané a nepříjemné na pohled. Dále se jednomu uživateli nezamlouvalo, že je některý text v programu psán bílým a jinde zase černým fontem. Jako řešení jsem zvolil možnost nakreslit vlastní grafiku uživatelského rozhraní a srovnání barvy fontů.

9. Úprava programu a ověření funkčnosti

9.1 Upravené části programu

Po provedení pilotní studie jsem se rozhodl upravit uživatelské rozhraní a nahradit původní grafické elementy *Unity* pro okna a tlačítka vlastními. K tvorbě jsem použil volně dostupný program *GIMP* [57]. Dále jsem mírně snížil velikost samotných modelů.

Na základě další připomínky jsem také doplnil jednoduchý manuál v podobě obrázku schémat jednotlivých ovladačů s popsanými funkcemi využívaných tlačítek. Ten jsem umístil přímo do tutoriálu v aplikaci, nad okno s textovým popisem jednotlivých kroků testování. Manuál je vložen taktéž do uživatelské příručky, kde jsou i uvedeny zdroje, které byly využity k jeho tvorbě.

Jako drobné vylepšení jsem také přidal popisky jednotlivých modelů, aby je šlo odlišit, pokud uživatel změní jejich pozici.

9.2 Metodika ověření

Pro zhodnocení úprav programu jsem pozval opět stejné uživatele, kteří prováděli první kolo studie a vyplňovali příslušný dotazník. Každé osobě jsem vždy jejich kopii vrátil a požádal ji o opětovné testování programu. Poté už jsem nevyžadoval vyplnění nového dotazníku, ale spíše vyjádření, zda byly body, ke kterým měl daný uživatel výhrady, uspokojivě upraveny.

9.3 Výsledky ověření

Uživatelé byli s provedenými změnami spokojeni. Kladně hodnotili zejména obrázkový manuál s popisem tlačítek a nové uživatelské rozhraní.

10. Závěr

Při vypracování této bakalářské práce jsem se seznámil s některými zařízeními a softwarovými řešeními pro tvorbu aplikací ve virtuální realitě. Dále jsem prostudoval základní metody subjektivního hodnocení kvality trojúhelníkových sítí.

Hlavní částí bylo navržení a realizace samotného software pro toto hodnocení *meshů* ve virtuální realitě. Ta byla realizována s pomocí softwaru *Unity* 2019 a jeho zdarma dostupného *SteamVR* pluginu. Aplikace bude dostupná na *Katedře informatiky a výpočetní techniky* pro provádění testování ve větším počtu osob. Funkčnost a použitelnost programu byla otestována pomocí takzvané pilotní studie.

Literatura

[1] Wikipedia. Virtual reality. [online] [Citace 25. 2. 2020] https://en.wikipedia.org/wiki/Virtual_reality

[2] Petrock, Victoria. Virtual and Augmented Reality Users 2019. [online] 27. března 2019. [Citace 25. 2. 2020] https://www.emarketer.com/content/virtual-and-augmented-reality-users-2019

[3] Váša, Libor - Skala Václav. *A Perception Correlated Comparison Method for Dynamic Meshes*. IEEE Trans Vis Comput Graph. 2011 Feb;17(2):220-30. doi: 10.1109/TVCG.2010.38. únor 2011 [Citace: 25. 2. 2020] Dostupné online: https://www.researchgate.net/publication/49675697_A_Perception_Correlated_Comparison_Method_for_Dynamic_Meshes

[4] Wikipedia, Virtual reality headset. [online] [Citace: 25. 2. 2020] https://en.wikipedia.org/wiki/Virtual_reality_headset

[5] Wikipedia, Stereopsis. [online] [Citace: 25. 2. 2020]

[6] Mehrfard, Arian - Fotouhi, Javad - Taylor, Giacomo - Forster, Tess - Navab, Nassir - Fuerst, Bernhard. *A Comparative Analysis of Virtual Reality Head-Mounted Display Systems*. Cornell University [online] 5. prosince 2019 [Citace: 25. 2. 2020]

[7] Oculus, Oculus Quest. [online] [Citace: 28. 2. 2020] https://www.oculus.com/quest/

[8] Ultraleap, Digital worlds that feel human. [online] [Citace: 28. 2. 2020] https://www.ultraleap.com/

[9] Vive, VIVE Wireless Adapter. [online] [Citace: 28. 2. 2020]
https://www.vive.com/sea/accessory/wireless-adapter/

[10] Vive, VIVE Power Bank.[online] [Citace: 28. 2. 2020]
https://www.vive.com/us/accessory/power-bank/

[11] Developer.Vive, VIVE Specs & User Guide. [online] [Citace: 28. 2. 2020]
https://developer.vive.com/resources/knowledgebase/vive-specs/

[12] Valve, Headset - Valve Index. [online] [Citace: 28. 2. 2020]
https://www.valvesoftware.com/index/headset

[13] Wikipedia, Oculus Rift CV1. [online] [Citace: 28. 2. 2020]
https://en.wikipedia.org/wiki/Oculus_Rift_CV1

[14] Wikipedia, Game engine. [online] [Citace: 28. 2. 2020]
https://en.wikipedia.org/wiki/Game_engine

[15] Christopoulou Eleftheria - Xinogalos, Stelios, *Overview and Comparative Analysis of Game Engines for Desktop and Mobile Devices*. Department of Applied Informatics, University of Macedonia, Greece. doi: 10.17083/ijsg.v4i4.194 [online] prosinec 2017 [Citace: 28. 2. 2020]
https://www.researchgate.net/publication/322027338_Overview_and_Comparative_Analysis_of_Game_Engines_for_Desktop_and_Mobile_Devices

[16] Amazon, Fully Customizable Game Engine - Amazon Lumberyard. [online] [Citace: 28. 2. 2020] https://aws.amazon.com/lumberyard/

[17] Cryengine, Cryengine. [online] [Citace: 5. 3. 2020]
https://www.cryengine.com/

[18] Store.Unity, Plans and pricing. [online] [Citace: 5. 3. 2020]
https://store.unity.com/

[19] Unity, Unity Shader Graph. [online] [Citace: 5. 3. 2020] https://unity.com/shader-graph

[20] Unity, Unity Real-Time Development Platform. [online] [Citace: 5. 3. 2020] https://unity.com/

[21] Steam, SteamVR. [online] [Citace: 5. 3. 2020] https://store.steampowered.com/app/250820/SteamVR/

[22] GitHub, ValveSoftware/openvr [online] [Citace: 5. 3. 2020] https://github.com/ValveSoftware/openvr

[23] UnrealEngine, The most powerful real-time 3D platform - Unreal Engine. [online] [Citace: 5. 3. 2020] https://www.unrealengine.com/

[24] Docs.UnrealEngine, Blueprints Visual Scripting. [online] [Citace: 5. 3. 2020] https://docs.unrealengine.com/en-US/Engine/Blueprints/index.html

[25] Váša, Libor. Software pro porovnávání modelů MeshTest, KIV ZČU. [online] 29. prosince 2013 [Citace: 5. 3. 2020] https://www.kiv.zcu.cz/cz/vyzkum/software/produkt-detail-cz.html?produkt_id=86

[26] Wikipedia, Polygon mesh. [online] [citace: 25. 2. 2020] https://en.wikipedia.org/wiki/Polygon_mesh

[27] Wikipedia, Triangle mesh. [online] [Citace: 5. 3. 2020] https://en.wikipedia.org/wiki/Triangle_mesh

[28] Blender, blender.org - Home of the Blender project. [online] [Citace: 15. 3. 2020] https://www.blender.org/

[29] Autodesk, 3ds Max. [online] [Citace: 15. 3. 2020] https://www.autodesk.com/products/3ds-max/

[30] Kraus, Karl. *Photogrammetry: Geometry from Images and Laser Scans*, de Gruyter; 2nd edition, November 2007, ISBN-13: 978-3110190076 [Citace: 25. 3. 2020]

[31] Autodesk, FBX. [online] [Citace: 25. 3. 2020] https://www.autodesk.com/products/fbx/overview

[32]Wikipedia, Wavefront .obj file. [online] [Citace: 25. 3. 2020] https://en.wikipedia.org/wiki/Wavefront_.obj_file

[33] Huebner - Robert, Reddy - Martin, Luebke - David, Varshney - Amitabh, Cohen - Jonathan D., Watson - Benjamin A.. *Level of Detail for 3D Graphic*, A volume in The Morgan Kaufmann Series in Computer Graphics, 2003, ISBN: 978-1-55860-838-2 [Citace: 28. 3. 2020]

[34] Corsini, M - Larabi, M. C. - Lavoué, G. - Petřík, O. - Váša, L. - Wang, K. *Perceptual Metrics for Static and Dynamic Triangle Meshes*. COMPUTER GRAPHICS forum, Volume 32 (2013), number 1 pp. 101–125, doi: 10.1111/cgf.12001, únor 2013, Dostupné online: https://www.researchgate.net/publication/258567938_Perceptual_Metrics_for_Static_and_Dynamic_Triangle_Meshes

[35] Wikipedia, Goraud shading. [online] [28. 3. 2020] https://en.wikipedia.org/wiki/Gouraud_shading

[36] Unity Documentation, Mesh. [online] [Citace: 5. 4. 2020] https://docs.unity3d.com/ScriptReference/Mesh.html

[37] Wikipedia, Bounding volume. [online] [Citace: 5. 4. 2020] https://en.wikipedia.org/wiki/Bounding_volume

[38] Unity Documentation, Creating and using scripts. [online] [Citace: 5. 4. 2020] https://docs.unity3d.com/Manual/CreatingAndUsingScripts.html

[39] Gamma, Erich - Helm, Richard - Johnson, Ralph - Vlissides, John. *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley Professional, October 1994, ISBN: 0201633612 [Citace: 5. 4. 2020]

[40] Unity, Assetstore. [online] [5. 4. 2020] https://assetstore.unity.com/

[41] Unity Documentation, Prefabs. [online] [5. 4. 2020] https://docs.unity3d.com/Manual/Prefabs.html

[42] Unity Documentation, Coroutines. [online] [5. 4. 2020] https://docs.unity3d.com/Manual/Coroutines.html

[43] Microsoft, BinaryReader Class. [online] [8. 4. 2020] https://docs.microsoft.com/en-us/dotnet/api/system.io.binaryreader?view=netframework-4.8

[44] Unity Documentation, Mesh Filter. [online] [8. 4. 2020] https://docs.unity3d.com/Manual/class-MeshFilter.html

[45] Unity Documentation, Mesh Renderer. [online] [8. 4. 2020] https://docs.unity3d.com/Manual/class-MeshRenderer.html

[46] Stack Overflow, Is it possible to define a 3d model/prefab size in unity?. [online] 27. dubna 2019 [Citace: 8. 4. 2020] https://stackoverflow.com/questions/55882745/is-it-possible-to-define-a-3d-model-prefab-size-in-unity

[47] Slovník cizích slov, centrování. [online] [Citace: 8. 4. 2020] https://slovnik-cizich-slov.abz.cz/web.php/slovo/centrovani

[48] Unity Documentation, Transform.RotateAround [online] [Citace: 8. 4. 2020] https://docs.unity3d.com/ScriptReference/Transform.RotateAround.html

[49] Unity Documentation, Canvas, [online] [Citace: 25. 4. 2020] https://docs.unity3d.com/Packages/com.unity.ugui@1.0/manual/UICanvas.html

[50] Unity Documentation, UI Button. [online] [Citace: 25. 4. 2020] https://docs.unity3d.com/2018.3/Documentation/ScriptReference/UI.Button.html

[51] ZČU, INFORMACE A OPATŘENÍ ZČU – COVID-19. [online] [Citace: 28. 5. 2020] https://www.zcu.cz/cs/Covid19-info/index.html

[52] Unreal Engine, Previous Versions, [online] [Citace: 2. 6. 2020] https://www.unrealengine.com/en-US/previous-versions

[53] Unreal Engine 4 Documentation, SteamVR Quick Start. [online] [Citace: 2. 6. 2020] https://docs.unrealengine.com/en-US/Platforms/VR/SteamVR/QuickStart/index.html

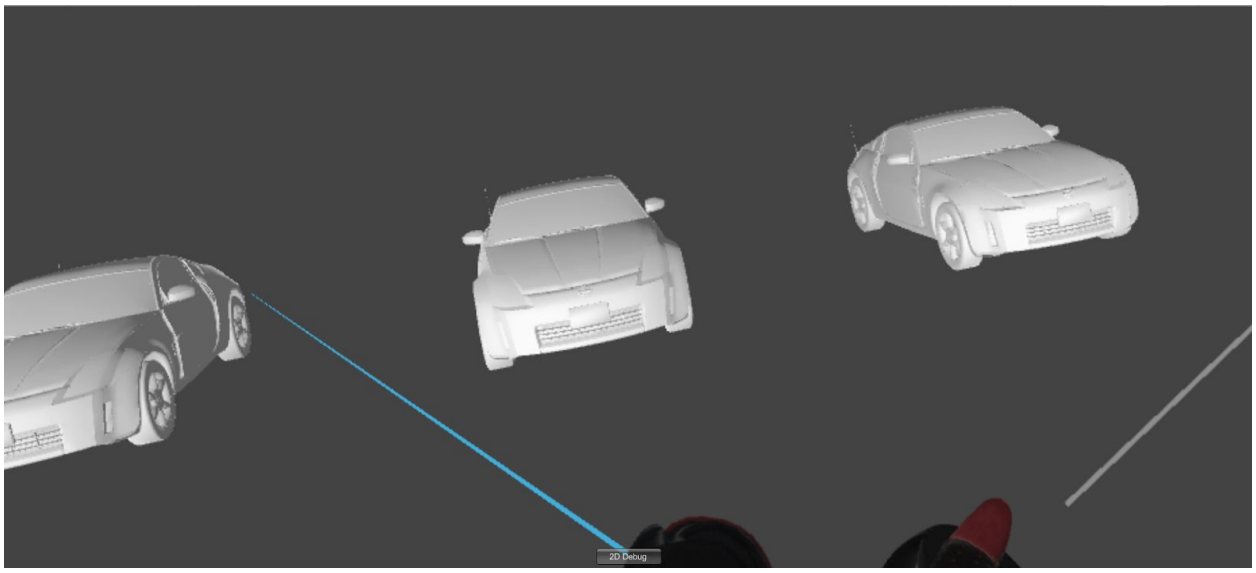
[54] Unity Documentation, Mesh.bounds. [online] [Citace: 5. 6. 2020] https://docs.unity3d.com/ScriptReference/Mesh-bounds.html

[55] Unity Documentation, Vector3.Scale. [online] [Citace: 5. 6. 2020] https://docs.unity3d.com/ScriptReference/Vector3.Scale.html

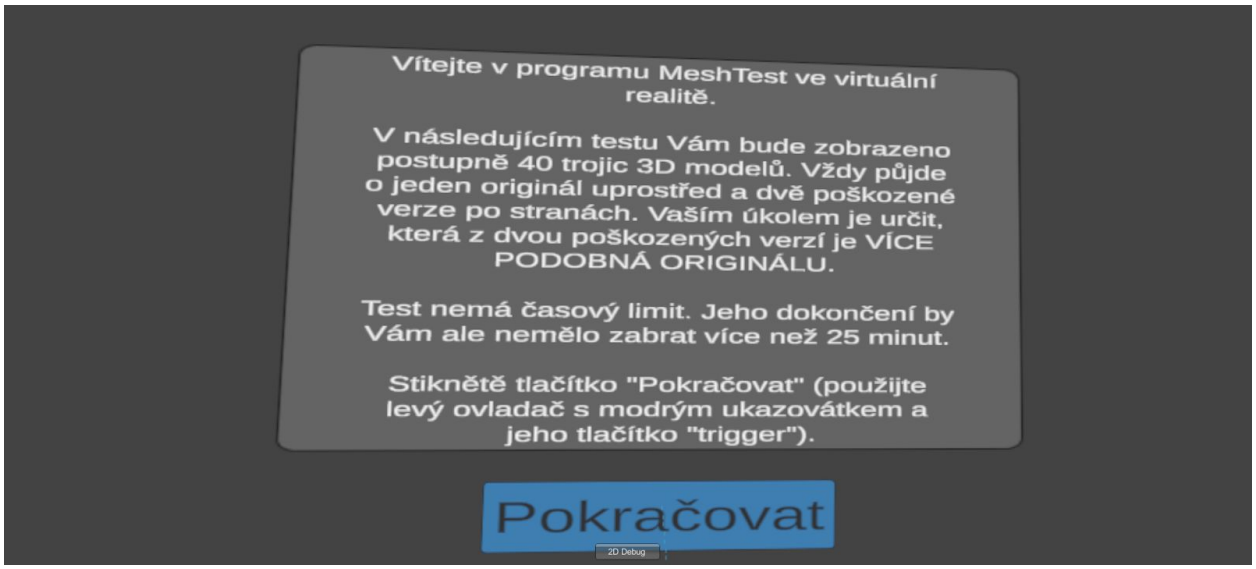
[56] Unity Documentation, TextMeshPRO. [online] [Citace: 8. 6. 2020] https://docs.unity3d.com/Manual/com.unity.textmeshpro.html

[57] GIMP, GNU Image Manipulation Program. [online] [Citace. 8. 6. 2020] https://www.gimp.org/

Příloha 1
Obrázky z verze před pilotní studií

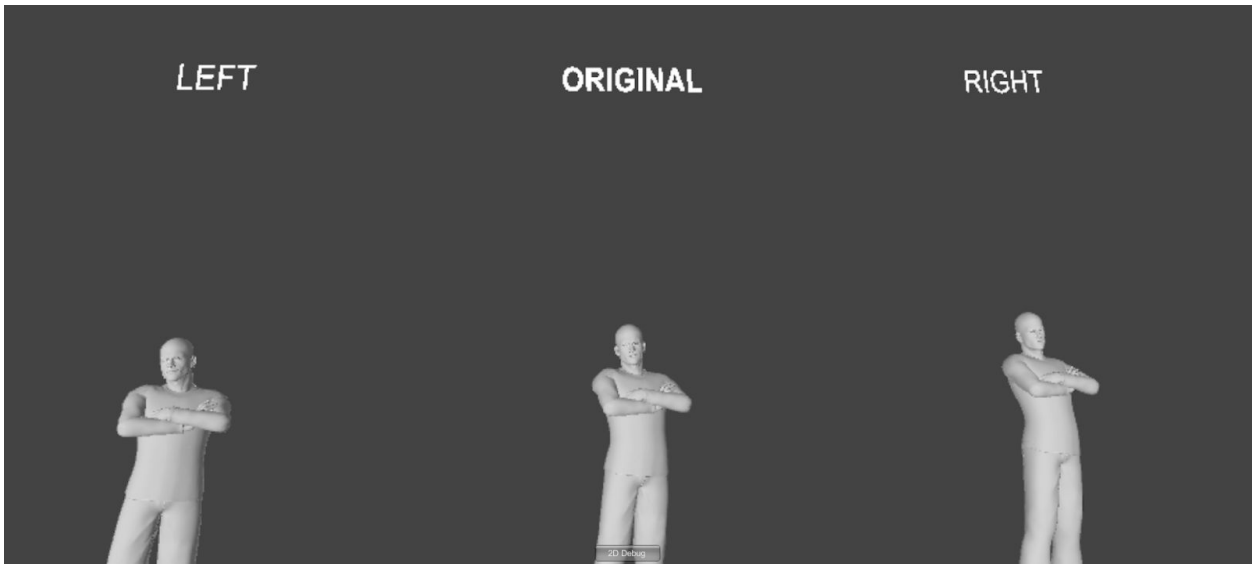


Obrázek P1.1: Testované objekty

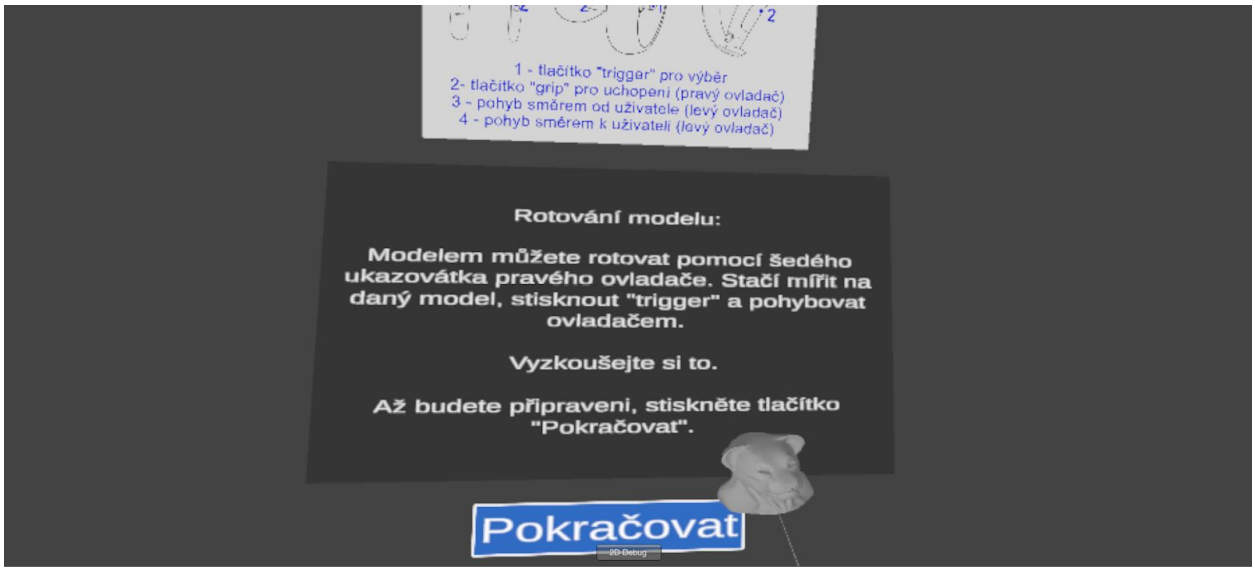


Obrázek P1.2: Tutorial

Příloha 2
Obrázky z verze po pilotní studii



Obrázek P2.1: Objekty byly mírně zmenšeny a byly přidány popisky



Obrázek P2.2: Byl přidán interaktivní tutorial s nápovědou

Příloha 3
Uživatelská příručka

Aplikaci MeshTestVR pro Windows 10 je umístěna na DVD, které je přílohou této bakalářské práce.

Podporovaným VR hardwarem jsou headsety HTC Vive, Valve Index a Oculus Rift CV1. Všechny jsou v současnosti (2020) k dispozici v laboratoři virtuální reality na FAV ZČU.

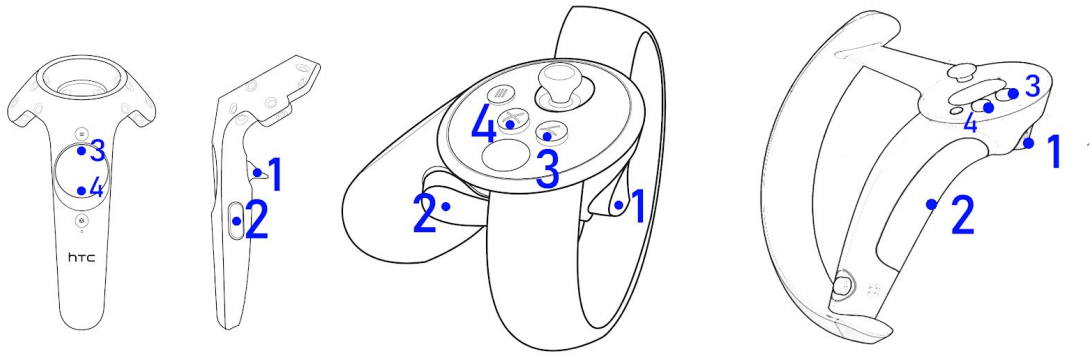
Je samozřejmě nutné splňovat minimální systémové požadavky určené výrobcem daného hardware.

Aplikace je vytvořena v enginu Unity, proto je ke spuštění nutné splňovat také požadavky stanovené výrobcem daného enginu:
https://docs.unity3d.com/Manual/system-requirements.html#player

Aplikace využívá platformu SteamVR od firmy Valve, proto je nutné si ji nejdříve nainstalovat do počítače. K tomu je nutné disponovat programem Steam a mít pro něj vytvořen uživatelský účet. Vše je možné postupně provést pomocí následující webové stránky:
https://store.steampowered.com/app/250820/SteamVR/

Poté už je možné spustit nejdříve platformu SteamVR a poté samotnou aplikaci.

Ovládání programu je vysvětleno v samotném tutoriálu. Přikládám ještě schéma s popsanými tlačítky pro ovladače podporovaných VR headsetů.



1 - tlačítko "trigger" pro výběr
2- tlačítko "grip" pro uchopení (pravý ovladač)
3 - pohyb směrem od uživatele (levý ovladač)
4 - pohyb směrem k uživateli (levý ovladač)
4 - reset pozice (pravý ovladač)

Pro uživatelskou příručku byly využity následující obrázky dostupné na webu:

Schéma ovladačů HTC Vive: https://imgur.com/gallery/5rTX05h

Schéma ovladačů Oculus Rift:
https://www.hiclipart.com/free-transparent-background-png-clipart-fhxlu

Schéma ovladačů Valve Knuckles controllers:
https://i.redd.it/7ccxyblfat031.png

Pro přidání dalších modelů v uvedeném proprietárním formátu je nutné umístit je do složky:
`./MeshTestVR\MeshTestVR_Data\StreamingAssets/meshes`
Je nutné dodržet vytvořenou strukturu. Vytvořte zde tedy složku pro příslušný model a umístěte do ní jeho original (pojmenujte ho `original`) a jeho poškozené varianty.""

Výsledky jsou po testování uloženy do adresáře:
`./MeshTestVR\MeshTestVR_Data`

Příloha 4
Dotazník k pilotní studii programu MeshTestVR

1) Ohodnoťte prosím Váš celkový dojem z programu.

2) Byl Vám postup testování vysvětlen v programu dostatečně?

3) Jak se Vám jevila manipulace s objekty. Byla dostatečně intuitivní?

4) Vyhovuje vám defaultní rozmístění testovaných objektů v prostoru scény?

5) Vyhovuje Vám osvětlení a pozadí scény?

6) Je uživatelské rozhraní nerušivé a dostatečně kvalitní?