TECHNICKÁ UNIVERZITA V KOŠICIACH
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

KOLABORÁCIA VO VIRTUÁLNEJ REALITE: VSTUPNO-VÝSTUPNÁ ČASŤ PRE MS HOLOLENS

Diplomová práca

2020 **Kristína Matiková, Bc.**

TECHNICKÁ UNIVERZITA V KOŠICIACH
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

KOLABORÁCIA VO VIRTUÁLNEJ REALITE: VSTUPNO-VÝSTUPNÁ ČASŤ PRE MS HOLOLENS

Diplomová práca

Študijný program:	Informatika
Študijný odbor:	9.2.1 Informatika
Školiace pracovisko:	Katedra počítačov a informatiky (KPI)
Školiteľ:	Ing. Štefan Korečko, PhD.
Konzultant:	Ing. Marián Hudák

2020 Košice **Kristína Matiková, Bc.**

Abstrakt

Táto diplomová práca sa venuje vstupno-výstupnej časti kolaboratívneho virtuálneho prostredia pre zariadenie zmiešanej reality Microsoft HoloLens. Je súčasťou projektu Global - Collaborative Virtual Environment, na ktorom spolupracujú aj ďalší riešitelia záverečných prác Technickej Univerzity v Košiciach v laboratóriu inteligentných rozhraní komunikačných a informačných systémov. Z pohľadu analýzy sa zapodieva zariadením Microsoft HoloLens a jeho technickými možnosťami v tejto oblasti. Rovnako sa analyzujú aj už existujúce aplikácie určené na spoluprácu vo virtuálnej realite a ich vlastnosti a spôsob využitia. Na základe poznatkov získaných z tejto analýzy sa navrhne a implementuje klientska časť kolaboratívneho virtuálneho prostredia s využitím framework-u A-Frame. Pre účely experimentálneho testovania sa vytvoria ďalšie jednoduché a potrebné prostredia. V závere práce sa vyhodnotia výsledky z tohto testovania a tým sa zhodnotí celkové použitie vytvoreného prostredia.

Kľúčové slová

Virtuálna realita, kolaborácia, Microsoft HoloLens, A-Frame

Abstract

This master thesis is devoted to input-output part of collaborative virtual environment for Microsoft HoloLens mixed reality device. Thesis is part of Global - Collaborative Virtual Environment project, in which other researchers at the Technical University of Košice cooperate in the laboratory of intelligent interfaces of communication and information systems. From the analytics point of view, it is concerned with Microsoft HoloLens device and its technical capabilities in this area. Moreover, existing applications for virtual reality collaboration, their properties and practical usage are also analyzed. The client part of the collaborative virtual environment will be designed and implemented using the A-Frame framework based on the knowledge gained from the analysis. Furthermore, other simple but necessary environments are created for experimental testing. In conclusion, the results of experimental testing are rated and thus the overall usability of the implemented environment is evaluated.

Key words

Virtual reality, collaboration, Microsoft HoloLens, A-Frame

57691

TECHNICKÁ UNIVERZITA V KOŠICIACH

FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Katedra počítačov a informatiky

ZADANIE DIPLOMOVEJ PRÁCE

Študijný odbor: **Informatika**
Študijný program: **Informatika**

Názov práce:

Kolaborácia vo virtuálnej realite: vstupno-výstupná časť pre MS HoloLens

Collaboration in Virtual Reality: Input-Output Component for MS HoloLens

Študent: **Bc. Kristína Matiková**
Školiteľ: **Ing. Štefan Korečko, PhD.**
Školiace pracovisko: **Katedra počítačov a informatiky**
Konzultant práce: **Ing. Marián Hudák**
Pracovisko konzultanta: **Katedra počítačov a informatiky**

Pokyny na vypracovanie diplomovej práce:

1. Oboznámiť sa so zariadením Microsoft HoloLens a analyzovať možnosti a existujúce prístupy jeho využitia v kolaboratívnej virtuálnej realite (KVR).
2. Na základe analýzy navrhnúť a implementovať klientskú časť systému KVR, určenú pre Microsoft HoloLens a to s použitím webových technológií.
3. Pri návrhu a implementácii spolupracovať s riešiteľmi záverečných prác, pracujúcich na ďalších častiach systému KVR.
4. Experimentálne overiť použiteľnosť implementovaného riešenia a porovnať ho s podobnými existujúcimi riešeniami.
5. Vypracovať dokumentáciu podľa pokynov vedúceho práce.

Jazyk, v ktorom sa práca vypracuje: slovenský
Termín pre odovzdanie práce: 04.05.2020
Dátum zadania diplomovej práce: 31.10.2019

prof. Ing. Liberios Vokorokos, PhD.
dekan fakulty

Čestné vyhlásenie

Vyhlasujem, že som celú diplomovú prácu vypracoval/a samostatne s použitím uvedenej odbornej literatúry.

Košice, 04. mája 2020 .....

vlastnoručný podpis

Poďakovanie

Týmto by som sa chcela vrelo poďakovať vedúcemu práce Ing. Štefanovi Korečkovi, PhD. a konzultantovi práce Ing. Mariánovi Hudákovi za ich odbornú pomoc pri riešení tejto práce formou konzultácii a užitočných rád. Rovnako im ďakujem za umožnenie prístupu do laboratória LIRKIS a zapožičanie zariadenia Microsoft HoloLens, ktoré bolo nevyhnutnou súčasťou tejto práce.

Zároveň sa chcem poďakovať svojím blízkym a priateľom, ktorý ma vždy motivovali a poskytovali mi morálnu podporu pri riešení problematiky diplomovej práce.

Obsah

Zoznam obrázkov

Zoznam tabuliek

Zoznam symbolov a skratiek

API Application Programming Interface predstavuje rozhranie pre programovanie aplikácii obsahujúce už naprogramované funkcie, triedy alebo iné programy

CGI Computer-Generated Imagery je štandard na vytváranie obrázkov generovaných počítačom

FPS Frame Per Second označuje počet snímok zobrazených za jednu sekundu

G-CVE Global – Collaborative Virtual Environtment je názov vyvíjaného prostredia určeného pre vzdialenú kolaboráciu v laboratóriu LIRKIS

KVR Kolaboratívna Virtuálna Realita

LIRKIS Laboratórium Inteligentných Rozhraní Komunikačných a Informačných Systémov je umiestené na Technickej Univerzite v Košiciach a venuje sa výskumu počítačovej grafiky a virtuálnej reality

MS Označuje skratku pre firmu Microsoft, ktorá stojí za zostrojením zariadenia HoloLens

Úvod

Existuje už mnoho aplikácii pre virtuálnu realitu, ktoré sa vyvíjali za rôznym účelom použitia, či už pre jedného alebo viacerých používateľov. Rovnako sú rozlíšené aj typom virtuálnej reality a typom cieľového zariadenia, pre ktoré sú vytvárané. V kontexte tejto diplomovej práce je dôležité vedieť, že virtuálna realita má viacero technológii odlišujúcich sa od seba spôsobom zobrazovania virtuálneho a reálneho sveta pre používateľov. Táto práca je práve venovaná jednému poddruhu, konkrétne zmiešanej realite, keďže cieľové zariadenie Microsoft HoloLens prvej generácie je primárne určené na zobrazovanie tohto typu virtuálnej reality. Zmiešaná realita vzniká spojením skutočného sveta s virtuálnymi objektami, ktoré interagujú so skutočným svetom v reálnom čase. Úlohou práce je teda spracovanie vstupno-výstupných údajov pre toto zariadenie za účelom grafického zobrazovania virtuálnych objektov a tiež spolupráce so vzdialenými klientami vo virtuálnom kooperatívnom prostredí G-CVE.

Táto diplomová práca je súčasťou väčšieho projektu s názvom Kolaborácia vo virtuálnej realite, na ktorom spolupracujú aj iný študenti Technickej Univerzity v Košiciach. Títo študenti sú riešiteľmi diplomových prác patriacich do tohto projektu s názvami Kolaborácia vo virtuálnej realite: vstupno-výstupná časť pre inteligentné telefóny [27] a Kolaborácia vo virtuálnej realite: komunikačná a riadiaca časť [28]. Očakávaným produktom našich diplomových prác je virtuálne kooperatívne prostredie schopné podporovať pripojenie klientov s rôznymi zariadeniami, konkrétne inteligentnými telefónmi a zariadením MS HoloLens. Toto prostredie bude spracovávať vstupné aj výstupné údaje pripojených zariadení a zabezpečovať serverovú komunikáciu medzi klientami, čo zabezpečí možnosť kolaborácie vzdialených klientov. Medzi tieto údaje patrí pozícia a rotácia klientov prenášaná na ich avatarov, vstupné hodnoty z ovládačov a údaje o interakcii s objektami. Okrem iného tento projekt poskytne možnosť iným študentom podieľať sa na rozširovaní tohto virtuálneho prostredia s názvom LIRKIS G-CVE [30] o nové funkcionality. Medzi tieto funkcionality môže patriť podpora ďalších zariadení virtuálnej reality alebo zahrnutie interaktívnych prvkov pre účely kolaborácie. Rovnako im bude umožnená grafická úprava virtuálnej scény pre potreby klientov. V súvislosti s tu spracovanou časťou práce bude možné vylepšenie projektu pre potreby novšej verzie zariadenia HoloLens 2.

Táto diplomová práca je rozčlenená podľa obsahu na štyri hlavné časti. V úvodnej kapitole sa analyzuje problematika kolaborácie v spojení s virtuálnou a zmiešanou realitou. To zahŕňa objasnenie technických možností zariadenia MS HoloLens z pohľadu priestorového mapovania, vstupných rozhraní a možností vývoja aplikácie pre túto platformu. Čo sa týka využitého jazyka a softvérového rámca (framework-u) na vývoj prostredia G-CVE, Javascript a A-Frame, je v krátkosti

zhrnutá ich podpora pre dané zariadenia ako aj pre iné zariadenia virtuálnej reality. Okrem iného je objasnená A-Frame architektúra a princíp na ktorom funguje. Ako posledné v tejto časti sú spomenuté už existujúce aplikácie určené na vzdialenú virtuálnu spoluprácu. Druhá a tretia kapitola sú venované návrhu a implementácii klientskej časti systému G-CVE, ale aj testovacích prostredí, ktoré boli využité na experimenty a vyhodnotenie práce. V rámci klientskej časti sa objasní konštrukcia prostredia a avatara používateľa, ktorý odráža pohyb klienta v scéne. Okrem toho sa opíšu spôsoby interakcie klienta so systémom. Potom sa táto časť bude venovať testovacím prostrediam, ktoré zahŕňajú implementáciu potrebného nového spôsobu ladenia systému. V tejto práci boli vytvorené tri testovacie prostredia a to konkrétne na kontrolu funkcionality systému strážcu, na porovnanie vzdialenosti medzi reálnym a virtuálnym svetom prostredníctvom pohybu používateľa a prostredie na kontrolu a porovnanie rôznych spôsobov interakcie. Záver práce je venovaný experimentálnemu testovaniu s využitím spomínaných prostredí. Účelom tohto testovania bolo objektívne vyhodnotenie práce na základe výsledkov zistených z meraní a opísaných v tejto kapitole.

1. Formulácia úlohy a cieľ práce

Cieľ tejto diplomovej práce je zameraný na celkové oboznámenie sa so zariadením Microsoft HoloLens a na analyzovanie možností tohto zariadenia vo vzťahu s kolaboratívnou virtuálnou realitou. Zistí sa existencia prístupov využitia tohto zariadenia v systéme KVR. Na základe poznatkov nadobudnutých v analytickej časti práce sa navrhne a implementuje klientska časť systému kolaborácie vo VR, špecificky určená pre zariadenie MS HoloLens. Implementuje sa za pomoci webových technológii v spolupráci s riešiteľmi ďalších záverečných prác, pracujúcich na iných častiach systému KVR. Rovnako je úlohou experimentálne overiť použiteľnosť implementovaného riešenia klientskej časti v praxi, s využitím objektívnych metód testovania. Pre vykonanie experimentov bude potrebná spolupráca so vzdialenými klientami. Na základe výsledkov sa vypracuje objektívne vyhodnotenie častí systému tak ako aj jeho celku. Na záver sa vypracuje podrobná dokumentácia pre možnosť budúceho nasadenia a vylepšenia klientskej časti systému inými záujemcami o rozšírenie tejto diplomovej práce

2. Kolaboračné prostredia v zmiešanej realite

Systém zmiešanej reality, ktorý spája reálne a virtuálne objekty do jedného prostredia, sa v súčasnosti rýchlo rozvíja rôznymi smermi z hľadiska využitia. Jedným z týchto smerov je aj spolupráca viacerých používateľov v takomto hybridnom prostredí, napriek tomu, že výskum aplikácií virtuálnej a zmiešanej reality sa zameriava zväčša na individuálne skúsenosti [20]. V závislosti od typu používaného zariadenia pre zobrazovanie virtuálneho prostredia, cez použitú technológiu na vývoj samotného softvéru až po nasadenie softvéru v praxi môžu byť výsledky výskumu rôzne. Preto pred samotným vývojom je podstatné preskúmať a analyzovať všetky závažné faktory ovplyvňujúce skúsenosť používateľa s kolaboráciou v zlúčenej realite. Prvým faktorom je zariadenie pre zobrazovanie zmiešanej reality a jeho dispozície z technického hľadiska. Na začiatku kapitoly sa teda analyzuje zobrazovacie zariadenie Microsoft HoloLens a jeho vstupno-výstupné technológie. Z pohľadu kolaborácie je dôležité aby zariadenie dokázalo presvedčiť zmysly človeka o reálnosti scény a poskytnúť spôsob interakcie používateľa s vizualizovaným prostredím. Rovnako relevantné je analyzovať podporu zariadenia pre rôzne druhy aplikácii, hlavne pre typ vyvíjanej aplikácie. Druhým faktorom je technológia využitá práve na tvorbu prostredia pre kolaboráciu, a preto sa v druhej časti kapitoly rozoberie použitý webový framework A-Frame, určený na vývoj webových aplikácií pre virtuálnu realitu a založený na otvorenej špecifikácii WebVR. Objasní sa princíp fungovania aplikácií vytvorených týmto frameworkom a architektúra, na ktorej sú stavané. V rámci kolaborácie sa analyzuje podpora využitia vytvoreného prostredia primárne zariadením HoloLens, ale stručne rozoberie aj podporu inými typmi zariadení. V závere kapitoly sú uvedené existujúce systémy kolaborácie vo virtuálnej realite, ktoré majú potenciál využitia v rôznych odvetviach, napríklad školstve alebo zdravotníctve. V závislosti od typu systému prinášajú tieto aplikácie potenciál uľahčiť ľudom vzdialenú komunikáciu a zlepšiť celkovú kvalitu života. Okrem toho sa analyzujú vlastnosti prostredí určených na kolaboráciu, pre lepšie porozumenie očakávaní používateľov od vyvíjanej aplikácie.

2.1. Technické schopnosti zariadenia Microsoft HoloLens

Pred vývojom aplikácie zmiešanej reality je dôležité porozumieť hlavným konceptom a faktorom ktoré vplývajú na dizajn a kvalitu aplikácie. Napríklad objekty väčšie ako holografický rámec zariadenia, nebudú používateľovi zobrazené v celku, a preto je dôležité zvážiť škálovanie objektov, alebo samotná aplikácia nebude spustiteľná na určitom zariadení, pretože nemusí podporovať zvolený druh aplikácie. Týchto faktorov sa pri vývoji vyskytuje veľa, a teda je dôležité preskúmať a porozumieť možnostiam, ktorými samotné zariadenie disponuje, aby sa predišlo komplikáciám počas vývoja. Nasledujúca kapitola sa teda bude venovať technickým vlastnostiam zariadenia Microsoft HoloLens, ktoré vychádzajú z oficiálnej Microsoft dokumentácie [1]. Medzi najhlavnejšie

faktory, ktoré budú konkretizované ďalej, patrí mapovanie reálneho priestoru, vstupno-výstupná časť zariadenia a podporovaný typ aplikácií virtuálnej reality. Priestorové mapovanie môže napomôcť k celkovej skúsenosti používateľa so zlúčenou realitou vytváraním detailnej reprezentácie skutočného povrchu snímaného hĺbkovými kamerami. Aplikácia tieto údaje z mapovania môže využiť na rozmiestnenie objektov v scéne tak, aby interagovali s reálnou miestnosťou. To presvedčí optické zmysly používateľa o reálnosti scény. Rovnako je dôležitá aj samotná interakcia osoby s prostredím. Tá by mala byť čo najviac prirodzená a napodobňovať interakciu v reálnom svete, preto je podstatné preskúmať vstupno-výstupné technické vybavenie zariadenia. Správnym dizajnom a implementáciou vstupov sa výrazne zlepší spolupráca používateľov v zlúčenej realite. Okrem toho typ vyvíjanej aplikácie úzko súvisí s možnosťou kolaborácie, keďže sa predpokladá využívanie rôznych zariadení virtuálnej reality, nie len zariadenia MS HoloLens. Pri vývoji je nevyhnutné zvážiť využitie aplikácie aj na iných platformách ako na cieľovej platforme a preto je potrebné porovnanie podporovaných aplikácií. Adekvátnou kombináciou spomínaných faktorov sa dosiahne čo najlepší výsledok práce.

2.1.1. Charakteristika technických možností

Microsoft HoloLens (Obr. 1) je plne samostatný holografický systém bežiaci na operačnom systéme Windows 10, ktorý využíva modernú optiku a senzory na zobrazovanie trojrozmerných hologramov do skutočného sveta [2]. Na rozdiel od ostatných zariadení virtuálnej reality si nevyžaduje pripojenie na počítač, ovládače alebo pripojenie na zdroj energie. Priehľadné holografické šošovky umožňujú používateľom vidieť reálny svet, v ktorom sú umiestnené virtuálne objekty zobrazené priehľadnými displejmi. Realistické zobrazenie objektov v priestore je zabezpečené environmentálnymi a hĺbkovými kamerami slúžiacimi na mapovanie reálneho priestoru. Okrem toho Microsoft HoloLens disponuje mikrofónom na hlasové ovládanie a vbudovanými reproduktormi pre zvukovú odozvu systému. Ako nezávislý systém zmiešanej reality, pozostáva HoloLens z niekoľkých funkčných komponentov, ktoré na základe operačného mechanizmu môžeme deliť na:

- Odhad polohy hlavy
- Rekonštrukcia reálneho prostredia
- Spracovanie virtuálneho prostredia
- Vnímanie používateľa
- Ovládanie používateľom

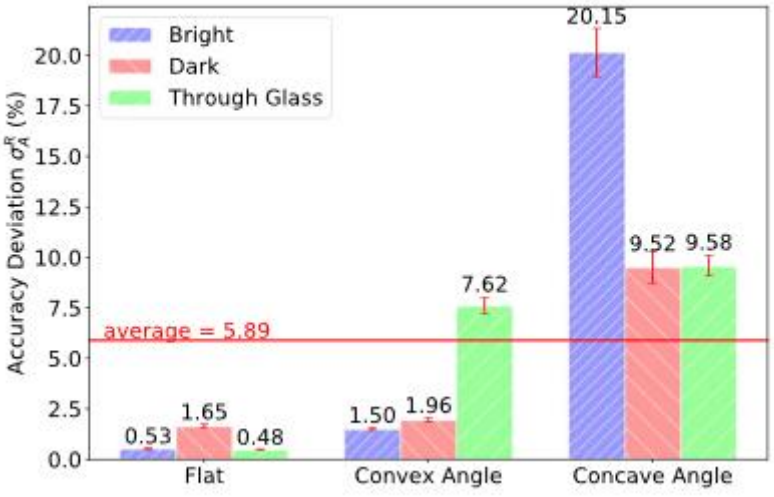
Obr. 1 Microsoft Hololens prvej generácie

2.1.2. Priestorové mapovanie v zmiešanej realite

Technológia priestorového mapovania umožňuje umiestňovať virtuálne objekty na reálny povrch. Ukotvenie hologramov do reálneho priestoru na základe informácií z mapovania povrchu pomáha presvedčiť používateľa o ich fyzickej prítomnosti. Takto umiestnené objekty majú fixnú pozíciu aj pri pohybe používateľa, ktorý si môže hologramy prezrieť z rôznych uhlov a vzdialeností. Microsoft HoloLens je schopný pomocou priestorového mapovania vytvoriť detailnú virtuálnu reprezentáciu povrchu reálneho sveta. Skenovanie priestoru je náročný proces na pamäť a preto je potrebné predísť problému s výkonom zariadenia, ako je napríklad vynechávanie snímkov obrazovky. Preto boli vyvinuté dve metódy skenovania povrchov, aktívne skenovanie a mapovanie s využitím vyrovnávacej pamäte. Aktívne priestorové mapovanie môže byť rovnako výhodné ako škodlivé, a preto vývojár aplikácií musí dôsledne zvážiť, čo je výhodnejšie a podporí lepší zážitok s virtuálnej reality. V prípade priestorového mapovania s využitím vyrovnávacej pamäte, zariadenie spravidla urobí snímku priestoru, uloží si ju do cache pamäte a následne používa tieto údaje počas celého používania aplikácie. Kontinuálnym mapovaním prostredia je zariadenie nútené aktívne skenovať prostredie, čím sa rýchlejšie zahrieva a to negatívne vplýva na procesor. Aby toho zariadenie bolo schopné potrebuje mať aj vysoké náklady na systém. Naopak skenovanie s využitím cache pamäte je v tomto smere lepšie, keďže jeho využitím dochádza k zlepšeniu výkonu zariadenia a jeho procesoru. Jeho nevýhodou je ale potenciálne veľké množstvo dát, ktoré musia byť uložené v pamäti aplikácie alebo zariadenia. Z hľadiska implementácie je ľahšie využiť cache mapovanie pretože nie je prerušené zmenami v priestore a údajoch o ňom. Avšak aplikácie, ktoré zahŕňajú pohyb reálnych objektov alebo ľudí, si vyžadujú aktívne skenovanie, pretože informácie o pohybe objektov nie je možné získať z údajov vyrovnávacej pamäte.

Ako sa uvádza v článku o vyhodnotení zariadenia MS HoloLens prostredníctvom montážnej aplikácie v rozšírenej realite [3], HoloLens na priestorové mapovanie využíva hĺbkové kamery, ktoré

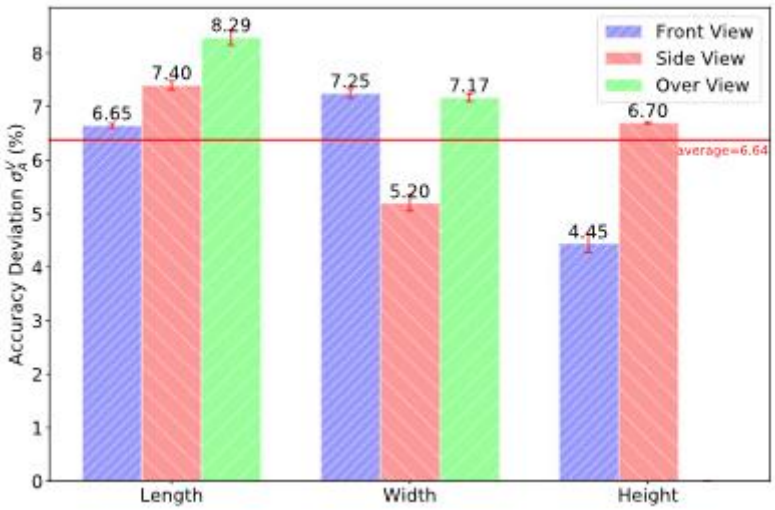
mapujú povrch miestnosti a vytvárajú z namapovaných údajov sieť. Pre väčšinu aplikácií je táto technológia postačujúca na vytvorenie interakcie hologramov a reálneho sveta. Avšak aplikácie, ktoré si vyžadujú precízne umiestnenie hologramov do reálneho sveta, ako napríklad montážne aplikácie si s touto technológiou nevystačia. Dôvodom je nedostatočne detailná namapovaná sieť údajov. V prípade potreby, sa technológia priestorového mapovania môže nahradiť technológiou založenou na rozpoznávaní značiek, ktoré môžu detailne definovať označené lokácie. Rovnako aplikácie vyžadujúce dáta priestorového mapovania, sú vysoko závislé na zariadení, ktoré tieto dáta automaticky získava pri používaní. Kvalita takýchto dát závisí od času, ktorý bol využitý na skenovanie a od oblasti, ktorá je skenovaná. Všeobecne platí že skenované dáta budú kvalitnejšie, ak zariadenie bude mať viac času na ich spracovanie. Ďalším dôležitým elementom je závislosť od osvetlenia a komplexnosti povrchov v miestnosti, ako je dokázané experimentom na rekonštrukciu skutočného prostredia využitím Microsoft HoloLensu [4]. V tomto experimente sa porovnávalo reálne prostredie s jeho rekonštruovaným 3D modelom, ktoré bolo ovplyvnené rôznymi faktormi (svetelné podmienky, textúra povrchu objektov, vzdialenosť zariadenia od skutočného objektu, tvar objektov). Výsledky prvého experimentu sú zobrazené na obrázku nižšie (Obr. 2), kde v grafe na osi y je číselne označená odchýlka presnosti mapovania a na osi x typ meraného povrchu. Pre každý typ povrchu (hladký, s konvexnými uhlami, s konkávnymi uhlami) je v grafe zapísaný výsledok odchýlky mapovania pri jasnom svetle označený modrou, v tmavom prostredí označený červenou a cez sklenený povrch označený zelenou.



Obr. 2 Porovnanie presnosti mapovania v rôznych podmienkach [4]

Výsledok meraní jasne naznačuje, že lepšie výsledky rekonštrukcie prostredia sú dosiahnuté za jasného svetla a s hladkými povrchmi, v porovnaní s tmavým prostredím a nerovnými povrchmi. Druhý experiment sa zameral na ukotvenie hologramov v reálnom prostredí a meranie medzery, poprípade prekrývania medzi hologramom a cieľovým povrchom. V tomto experimente sa využila

aplikácia, ktorá pripevnila virtuálnu kocku presných rozmerov na skenovaný povrch reálnej kocky s rovnakými rozmermi. Účastníci projektu skenovali povrch z rôznych uhlov a následne umiestnili kocku na reálny povrch. Pri každom uhle mapovania sa zaznačila vzdialenosť hologramu od cieľového povrchu, ktorá bola ovplyvnená len algoritmom priestorového mapovania. Výsledky tohto experimentu sú zaznamenané na obrázku nižšie (Obr. 3). V grafe na osi y je zaznačená odchýlka presnosti pre vizualizáciu hologramu v priestore a na osi x meraná odchýlka v rámci dĺžky, šírky a výšky virtuálnej kocky. Modrou farbou je označené meranie z čelného pohľadu, červenou z bočného pohľadu a zelenou z pohľadu zhora.



Obr. 3 Porovnanie presnosti ukotvenia hologramu z rôznych uhlov [4]

Na základe týchto výsledkov zaznamenaných v grafe sa určilo, že HoloLens je schopný vizualizovať hologramy so štandardnou odchýlkou 0,25 cm, čo je pre ľudské oko ťažko rozoznateľné.

2.1.3. Rozhrania pre vstup

Zariadenie Microsoft HoloLens disponuje troma hlavnými rozhraniami pre vstup, ktoré výrazne zlepšujú vnímanie zmiešanej reality používateľom:

- Spracovanie pohľadu
- Rozoznávanie gest
- Rozoznávanie hlasu

Na základe informácií z týchto vstupných rozhraní sú používateľovi prostredníctvom zariadenia poskytnuté potrebné vizuálne ale aj zvukové informácie. Tieto informácie môžu byť ďalej používateľom ovládané a to pomocou rozpoznávania hlasových príkazov, gest rúk a pohľadu používateľa určeného na základe polohy a rotácie hlavy.

Holografický rámec predstavuje spôsob pre vyvolanie interakcií medzi používateľom a virtuálnym svetom. Je preto dôležité vyhodnotiť kam upriamuje používateľ pozornosť a ako vyvoláva interakciu

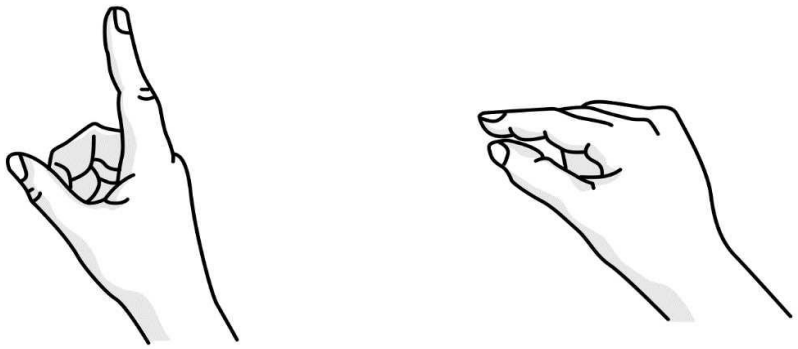
so systémom. Spracovanie pohľadu je najdôležitejšia interakcia zariadenia HoloLens, keďže spárovaním s gestami alebo hlasom dokáže používateľ ovplyvňovať virtuálny svet okolo seba. Ak by aplikácia podporovala manipuláciu s viacerými objektami, práve údaje zo spracovania pohľadu by systému určilo, s ktorým objektom používateľ chce naviazať interakciu. Gaze alebo pohľad, je preto primárnou formou vstupu v prostredí zmiešanej reality. Tak ako v reálnom svete, človek sa pozrie na virtuálny objekt, s ktorým chce pracovať a následne vyvolá interakciu pomocou jedného z ďalších vstupných rozhraní. Pohľad posiela informácie zariadeniu o smere, ktorým sa používateľ pozerá, čo umožňuje určiť používateľov úmysel vo virtuálnom prostredí. Okrem určovania pozornosti používateľa v scéne, sa gaze využíva na zameranie a rozpoznávanie gest, čo umožňuje interakciu používateľ – virtuálny svet. Ak sa využije sledovanie pohľadu spolu s mriežkou priestorového mapovania, používateľovi sa naskytne možnosť pridávať nové hologramy do scény ich umiestnením na povrch v reálnom svete. Rovnako je dôležité, že zariadenia pre virtuálnu realitu môžu disponovať dvoma spôsobmi zisťovania pohľadu, a to na základe:

- polohy hlavy (head gaze)
- polohy zreníc (eye gaze)

Čo sa týka konkrétne zariadenia HoloLens, jeho staršia verzia HoloLens prvej generácie disponuje len pohľadom určeným polohou hlavy, zatiaľ čo HoloLens 2 podľa oficiálnej Microsoft dokumentácie [1] podporuje oba možnosti. Head gaze je vektor umiestnený medzi očami používateľa a určovaný pomocou polohy a rotácie jeho hlavy. Ak sa používateľ rozhliada po miestnosti, tento vektor sa pohybuje súčasne s jeho hlavou a tak určuje miesto na ktoré sa pozerá. Polohu hlavy používateľa možno určiť z polohy a orientácie systému HoloLens odhadnuté pomocou algoritmu inerciálnej meracej jednotky a algoritmu iteračného najbližšieho bodu. Podľa Zhanga, úspešnosť týchto algoritmov výrazne ovplyvní presnosť sledovania polohy hlavy systémom [5]. Reálne prostredie je snímané pomocou hĺbkových kamier a environmentálnych kamier, ktoré spolu s využitím algoritmu KinectFusion dokážu zrekonštruovať reálne prostredie na trojrozmerný objekt [6]. Avšak nevýhodou určovania pohľadu len na základe orientácie hlavy je to, že človek nie vždy pozerá na miesto, ktorým má natočenú hlavu. Aby bola interakcia so zmiešanou realitou úspešná je človek nútený neustále pozerať pred seba a svoju pozornosť z hologramu na iný meniť len pohybom hlavy. Väčšina aplikácií virtuálnej reality využíva spolu s technológiou head gaze aj vizuálnu indikáciu toho, kam sa človek pozerá, teda kurzor. To poskytuje istotu pre používateľa pri interakcii s objektami. V prípade využitia technológie eye gaze sa využitie kurzora neodporúča, pretože to môže spôsobiť odvrátenie pozornosti zo scény na neustále pohybujúci sa kurzor. Interakcia s využitím sledovania pohybu oka je rýchlejšia na rozdiel od sledovania polohy a orientácie hlavy. Ako uvádza Zhiwei a Qiang [7], monitorovanie pohybu oka v reálnom čase

umožňuje systému zmeniť scénu na základe priestorových a časových charakteristík pohybov očí. Napríklad údaje získané zo sledovania pohybu oka môže systém využiť na odvodenie informácií, o ktoré má používateľ záujem. Následne systém podnikne kroky ako je zaostrenie alebo priblíženie oblasti, na ktorú používateľ upiera pozornosť. Najbežnejším prístupom k odhadu smeru zraku je podľa článku o interakcii s využitím sledovania pohybu oka, technológia založená na relatívnej polohe medzi zornicou a rohovkou [8]. Okrem toho Zhiwei a Qiang taktiež tvrdia, že samotné sledovanie pohybu oka nestačí. Celkový smer pohľadu človeka je určený dvoma faktormi: orientácia tváre a orientácia očí. Orientácia tváre predstavuje globálny smer pohľadu zatiaľ čo orientácia očí lokálny smer. Spolu teda určujú finálny pohľad osoby.

Podpora gest rúk zariadením Microsoft HoloLens poskytuje používateľovi okamžitú interakciu s obsahom scény a bez využitia ďalšieho príslušenstva akým je napríklad gamepad. HoloLens zameriava gestá rúk, ktoré sú pre neho viditeľné, sledovaním ich polohy pomocou mechanizmu head gaze s využitím hĺbkových a environmentálnych kamier. Dokáže rozpoznať dva stavy rúk a to ich stav pripravenosti a stav stlačenia (Obr. 4).



Obr. 4 Zľava stav pripravenosti, sprava stav stlačenia [1]

Ak sa ruky nachádzajú mimo rámca, kurzor na HoloLense má tvar bodky, zatiaľ čo pri rozoznaní rúk v rámci zmení svoj tvar na kruh. Takáto odozva systému napomáha správnej interakcii používateľa so systémom. Rovnako, ak sú ruky v iných pozíciách HoloLens ich ignoruje, ale ak zariadenie rozpozná gesto, systém vykoná príslušnú operáciu. Pre každú ruku nachádzajúcu sa v rámci, dokáže HoloLens poskytnúť informácie o jej polohe a smerovom vektore. Kombinácia technológie sledovania pohľadu v spojení s vykonaním operácie danej systémom sa nazýva gaze-and-commit interakcia. Alternatívou k tejto interakcii je point-and-commit, ktorá zariadením HoloLens nie je podporovaná, pretože si vyžaduje použitie ovládačov na pohyb.

Keďže zariadenie Hololens je obmedzené z hľadiska technológie a jednotného návrhu interakcie používateľa so systémom, disponuje len obmedzeným počtom gest. Tento počet možných interakcií so zariadením HoloLens by mohol byť navýšený napríklad zahrnutím externého produktu

Microsoft Kinect ako je uvedené v článku o prirodzenej a intuitívnej interakcii s hologramami [9]. Zahrnutím ďalšieho senzoru by sa zvýšil počet použiteľných gest a tak sa zlepšila prirodzenosť interakcie s virtuálnym obsahom. Avšak v súčasnosti HoloLens rozpoznáva len dve hlavné gestá nazývané Air tap a Bloom. Gesto Air tap je ekvivalentom kliknutia myšou. Pre jeho správne použitie používateľ musí zdvihnúť ruku vo zvislej polohe na úroveň hlavy, namieriť pohľad na prvok v scéne a klepnúť palcom a ukazovákom. Druhým základným gestom je Bloom. Podobne ako klávesa Windows, gesto Bloom je špeciálna akcia HoloLens systému a slúži len na návrat do ponuky Štart. Používateľ realizuje toto gesto zdvihnutím ruky dlaňou hore a spojenými prstami. Následne len roztvorí ruku. Okrem spomínaných hlavných gest HoloLens reaguje aj na zložené gestá. Tieto zložené gestá začínajú použitím gesta Air Tap a udržaním ruky v tejto pozícii. Z tejto pozície sa odvíja navigačné a manipulačné gesto hýbaním ruky pozdĺž súradnicových osí v trojrozmernom priestore. Odchýlky v každom smere sú detegované zariadením a sú v rozsahu -1 až 1 [10]. Využitím týchto údajov dokáže systém prepočítavať polohu hologramov, s ktorými používateľ manipuluje a tým zabezpečuje prirodzené presúvanie objektov v zlúčenej realite.

Posledným hlavným vstupným rozhraním slúžiacim na interakciu so systémom je rozpoznávanie hlasových príkazov. Pri používaní hlasových povelov môže byť potrebné využiť zameriavací mechanizmus, napríklad kurzor. To sa vzťahuje hlavne na povel Select, ktorý je alternatívou gesta Air Tap. Niektoré príkazy si ale zameriavací mechanizmus nevyžadujú, napríklad hlasové pokyny pre virtuálneho asistenta Microsoft Cortana. V Microsoft dokumentácii sú uvedené všetky špecifické príkazy pre HoloLens, ktoré dokáže Cortana spracovať. Príkladný je povel Go to Start ktorý plní rovnakú funkciu ako gesto Bloom. Okrem toho HoloLens má zabudovanú funkciu „pozri, povedz to“ ktorá pridáva písomné označenia všetkým tlačidlám v používateľskom rozhraní, ktoré sa dajú ovládať hlasovým povelom. Používateľom to výrazne uľahčí prácu so zariadením pretože uvidia všetky príkazy, ktoré môžu nad daným tlačidlom vysloviť. Podľa Hon-Andersona [11] medzi najdôležitejšie faktory pri rozpoznávaní reči patrí rýchlosť a presnosť. Používatelia očakávajú od systému aby interpretoval ich hlas čo najpresnejšie a poskytoval spätnú väzbu v reálnom čase tak, že používateľ rýchlo zistí akú reakciu spustil ich hlasový príkaz. V prípade HoloLensu sa po zaznamenaní hlasového príkazu nad kurzorom zobrazí spätná väzba, ktorá obsahuje názov príkazu, ktoré zariadenie rozoznalo.

Okrem už spomínaných troch hlavných vstupov zariadenia HoloLens je možné zariadenie spárovať prostredníctvom technológie Bluetooth s ovládačom nazývaným HoloLens Clicker. Tento ovládač disponuje len jedným tlačidlom slúžiacim na výber, posúvanie, presúvanie a zmenu veľkosti hologramov namiesto použitia gest alebo hlasových príkazov. Keďže zariadenie Hololens má

k dispozícii Bluetooth technológiu je možné spárovať ho aj s inými vstupnými Bluetooth zariadeniami, napríklad myšou alebo klávesnicou.

2.1.4. Možnosti vývoja aplikácii

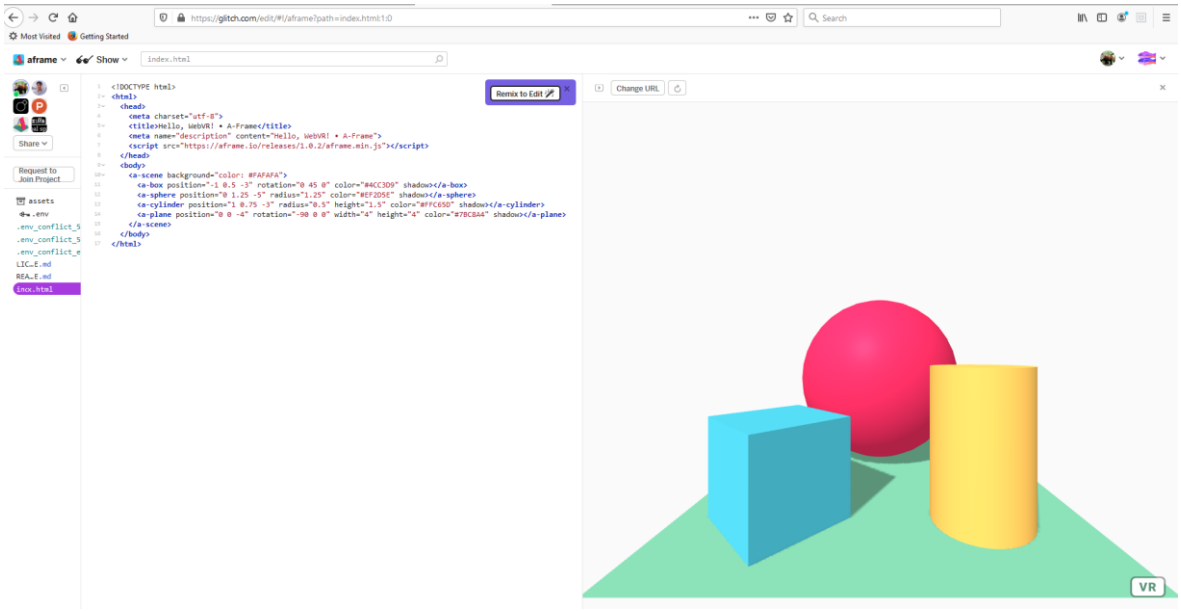
Existuje viacero metód vývoja medzi ktorými si vývojár aplikácií pre HoloLens môže zvoliť. Každý proces návrhu si vyžaduje prístup k problémom v širšom zmysle a ich opakované riešenie tak, aby vyhovovalo koncovým používateľom. Napriek tomu, že sa táto práca venuje vývoju kolaboračnej aplikácie v rámci A-Frame (bližšie popísanom v kapitole 2.2.), je dôležité spomenúť aj ostatné možnosti vývoja aplikácie pre HoloLens. Jednou z nich je aj vývojová platforma Unity, ktorá ako sa uvádza v jej oficiálnej dokumentácii [12], okrem zariadenia HoloLens poskytuje vstavanú podporu pre množstvo iných zariadení virtuálnej reality. Pomocou Unity je možné vytvoriť kolaboračnú aplikáciu s možnosťou podpory prepínania medzi viacerými zariadeniami, čo rieši problém kolaborácie používateľov s rôznymi zariadeniami na virtuálnu realitu. Nevýhodou je neustále exportovanie a nasadenie vytvoreného projektu na reálne zariadenie alebo HoloLens emulátor, za účelom testovania validity aplikácie. To môže výrazne navýšiť čas potrebný na vývoj takejto aplikácie.

Ďalšou alternatívou je vývoj s použitím technológie WebVR, otvorenej špecifikácie, ktorá umožňuje zažiť virtuálnu realitu cez webový prehliadač. Podľa Microsoft dokumentácie a oficiálnej stránky WebVR [13], je táto technológia na zariadení HoloLens dostupná prostredníctvom webového prehliadača Microsoft Edge. Webová aplikácia vytvorená týmto spôsobom prezentuje obsah vo virtuálnej realite pomocou knižnice WebGL, ak dostupné VR zariadenie podporuje režim virtuálnej prezentácie obsahu stránky. Webový framework A-Frame poskytuje systém pre vývoj trojrozmerných prostredí podobný systému pre tvorbu webových stránok a je postavený práve na technológii WebVR. Poskytuje preddefinované komponenty ako sú modely, geometrické útvary alebo animácie. Veľkou výhodou takýchto aplikácií je ich vysoká dostupnosť pre rôzne zariadenia virtuálnej reality.

2.2. Aframe

Kapitola sa bude venovať webovému frameworku pre vytváranie prostredí virtuálnej reality nazývaného A-Frame a bude vychádzať z oficiálnej dokumentácie tohto frameworku [14]. Jeho základom je jadro založené na princípe entita-komponent, ktoré poskytuje deklaratívnu, rozšíriteľnú a skladateľnú štruktúru. Výhodou je možnosť vývoja z obyčajného HTML súboru bez nutnosti inštalácie vývojového prostredia alebo exportovania a nasadenia na zariadenie. A-Frame poskytuj možnosť vytvárať aplikácie s využitím online editora kódu Glitch (Obr. 5), ktorý okamžite nasadzuje vyvíjanú webovú aplikáciu na svoj server. To výrazne uľahčuje proces vývoja a testovania

validity aplikácie. V prvej časti tejto kapitoly je opísaná podpora A-Frame pre platformy, virtuálne zariadenia a webové prehliadače, z hľadiska eventuálnej kolaborácie používateľov s rôznymi zariadeniami. Čím bude podpora A-Framu väčšia, tým menej kritických problémov sa vyskytne pri vývoji. V ďalšej časti sa ozrejmí architektúra entita-komponent-systém, využívaná najmä na vývoj hier a založená na princípe dedičnosti a hierarchie. Vysvetlí sa jej koncept , výhody a využitie pre tvorbu webovej aplikácie pre virtuálnu realitu. Keďže A-Frame je postavený na HTML, v poslednej časti tejto kapitoly sa opíše ovládanie virtuálnej scény a entít pomocou jazyka Javascript a DOM API, ktoré sa využívajú pri bežnom vývoji webových aplikácií. Okrem toho sa upresní zapuzdrenie Javascript kódu do A-Frame komponentov z dôvodu modularizácii kódu a prehľadnejšej logiky a správania aplikácie.



Obr. 5 Online editor Glitch, zľava kód aplikácie, sprava vizualizácia aplikácie

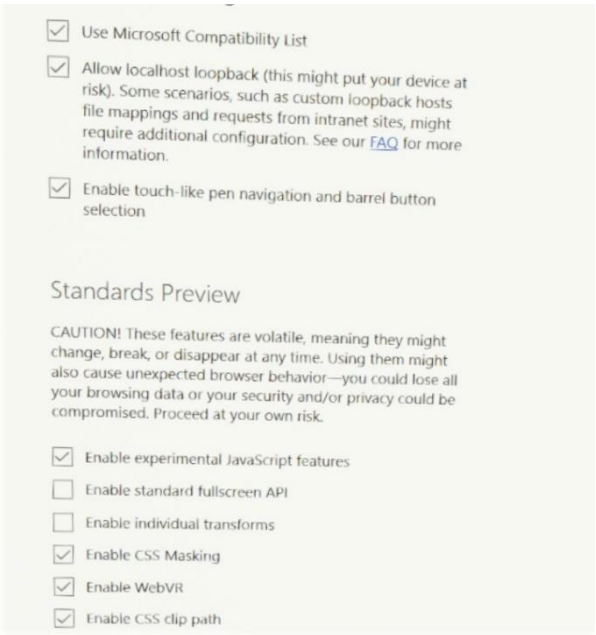
2.2.1. Podpora pre VR zariadenia a webové prehliadače

Z hľadiska kolaborácie vo virtuálnej realite je potrebná čo najlepšia podpora pre rôzne VR zariadenia. V rámci frameworku A-Frame je navyše závažná aj podpora WebVR webovými prehliadačmi. Podľa oficiálnej A-Frame dokumentácie je v súčasnosti podporovaný zariadeniami a webovými prehliadačmi uvedenými v Tab. 1. Rovnako podporuje aj najmodernejšie mobilné prehliadače, ktoré nemajú WebVR podporu prostredníctvom metódy WebVR pollyfill. Medzi tieto prehliadače patrí Safari a Firefox pre operačný systém iOS a Chrome pre operačný systém Android. Napriek tomu, že zariadenie HoloLens sa v tejto dokumentácii nespomína, podpora pre webový prehliadač Microsoft Edge umožňuje spúšťať webové aplikácie pre virtuálnu realitu aj na tomto zariadení.

VR headset	Webový prehliadač
HTC Vive	Supermedium
Oculus Rift	Firefox
Oculus Quest	Oculus Browser
Oculus Go	Samsung Internet
Google Daydream	Microsoft Edge
Samsung GearVR	Chrome
Vive Focus	Exokit

Tab. 1 Zoznam podporovaných zariadení a prehliadačov

Napriek tomu, že zariadenie HoloLens sa v tejto dokumentácii nespomína, podpora pre webový prehliadač Microsoft Edge umožňuje spúšťať webové aplikácie pre virtuálnu realitu aj na tomto zariadení. Pre správne fungovanie A-Frame aplikácií na zariadení HoloLens je potrebné v prehliadači v časti about:flags povoliť experimentálne funkcie Enable experimental Javascript features a funkciu Enable WebVR (Obr. 6).



Obr. 6 Nastavenie experimentálnej funkcie WebVR na zariadení HoloLens

2.2.2. Architektúra entita-komponent-systém

Entita-komponent-systém je architektonický vzor využívaný webovým frameworkom A-Frame. Táto architektúra dodržiava zásadu kompozícia pred dedičnosťou, čo umožňuje väčšiu flexibilitu pri vytváraní virtuálnych objektov. Zloženie pred dedičnosťou je v objektovo orientovanom programovaní princíp ktorý sa snaží dosiahnuť polymorfné správanie a opakované použitie kódu v rámci zloženia tried, namiesto ich dedičnosti od rodičovskej triedy [15]. Využitím tejto zásady sa odstránia problémy so širokou hierarchiou dedenia, ktorá je častokrát ťažko pochopiteľná.

Namiesto toho sa využíva princíp kde každý objekt vo virtuálnej scéne predstavuje entitu pozostávajúcu z jedného alebo viacerých komponentov definujúcich správanie a funkcionalitu danej entity. Podľa terminológie Martina [16], je táto architektúra definovaná takto:

- Entita – objekt na všeobecné účely, označuje každý objekt vo virtuálnej scéne ako samostatnú položku
- Komponent – údaje o jednom aspekte objektu a o jeho interakcii s virtuálnym svetom
- Systém – beží nepretržite a vykonáva akcie nad každou entitou, ktorá má komponent s rovnakým aspektom ako daný systém

Táto definícia je všeobecná pre tento architektonický vzor a preto sa v A-Frame dokumentácii uvádza špecifická definícia, kde:

- Entita – je základom všetkých objektov v scéne a bez pripojených komponentov nič nevytvára ani nevykresľuje, podobne ako je prázdny v HTML
- Komponent – je opakovane použiteľný modul alebo dátový kontajner na poskytovanie vzhľadu, správania alebo funkčnosti entít s možnosťou miešania, porovnávania a konfigurácie komponentov
- Systém – poskytuje globálnu správu a služby pre triedy komponentov a je využiteľný na separáciu logiky a dát

V prípade vytvorenia reprezentácie používateľa vo virtuálnom svete by entita predstavovala samotného používateľa zasadeného do scény a pozostávala by z komponentov pozícia, rotácia a avatar hráča. Každý pripojený komponent by mal svoje vlastnosti, napríklad avatar by mal vlastnosť farba a identifikačné číslo. Úpravou vlastností komponentov je možné vytvárať rôzne typy hráčov a ich možností v rámci systému. Napríklad pridaním ďalších komponentov k entite hráča by v kolaboračnom virtuálnom prostredí bolo možné rozlíšiť používateľov s možnosťou manipulácie a interakcie s inými entitami a používateľmi, ktorý majú len rolu pozorovateľa.

Medzi výhody architektúry entita-komponent-systém patrí väčšia flexibilita pri definovaní objektov miešaním alebo využívaním existujúcich komponentov. To odstraňuje prípadné problémy so zložitými reťazcami dedičných tried a s ich komplexnou, prepojenou funkcionalitou. Rovnako podporuje vytváranie čistého dizajnu kódu, ktorý je ľahko čitateľný a porozumiteľný a to prostredníctvom enkapsulácie, modularizácie a možnosti opätovného použitia kódu. Táto architektúra je najviac využívaná a osvedčená pri vývoji komplexných aplikácií virtuálnej reality. Okrem toho samostatné funkčné komponenty môžu byť zdieľané v rámci A-Frame komunity, čo uľahčuje komunite rozvoj tohto webového frameworku.

2.2.3. Využívanie jazyku JavaScript v rámci komponentov

A-Frame je vo svojej podstate HTML s programovou časťou a so špecifickými tagmi pre vytváranie entít, a preto je možné tieto entity v scéne ovládať pomocou jazyka JavaScript alebo využitím DOM API, ktoré sa využívajú pri vývoji bežných webových aplikácií. Pre ich využívanie framework A-Frame zaviedol predpísané zapuzdrenie takého to kódu do komponentov, čo zabezpečuje vykonanie tohto kódu v správnom čase. Podľa oficiálnej dokumentácie predpis zapuzdrenia kódu, alebo aj registrovania komponentu pozostáva z jedinečného názvu komponentu a jeho definície. Samotná definícia obsahuje objekt schému, ktorá definuje a popisuje vlastnosti komponentu, a metódy spracovania životného cyklu:

- Init – zavolá sa len raz, keď sa komponent inicializuje
- Update – zavolá sa, keď sa komponent inicializuje a vždy, keď sa zmení niektorá z jeho vlastností
- Tick – zavolá sa pri každom vykresľovaní scény
- Remove – zavolá sa, keď je komponent odstránený z entity, alebo keď je entita odpojená zo scény
- Pause – zavolá sa, keď je komponent odstránený zo scény a keď je zo scény odstránené alebo pozastavené dynamické správanie
- Play – zavolá sa, keď sa komponent inicializuje a keď sa do scény pridá dynamické správanie

V závislosti od vykonania kódu v životnom cykle aplikácie sa vloží kód v jazyku JavaScript do jednej z týchto metód. Základným príkladom je JavaScript funkcia na výpis hodnôt do konzoly prehliadača console.log. Pre jej využitie je potrebné registrovať nový komponent pred zavolaním HTML kódu pre vytvorenie scény, vložiť túto funkciu do jednej z metód spracovania životného cyklu aplikácie a následne zavolať názov komponentu ako názov nejakého HTML atribútu v rámci A-Frame scény. Týmto spôsobom je možné využiť JavaScript na pridávanie, upravovanie a odstraňovanie entít a ich komponentov. Zásluhou jazyka Javascript a DOM existuje možnosť jednoduchej komunikácie medzi entitami a komponentami prostredníctvom eventov. Event signalizuje aplikácii, že sa odohrala udalosť na ktorú čakala [17]. Najdôležitejším typom eventov pre interaktívne webové aplikácie je zachytávanie signálov z rôznych ovládačov, vďaka čomu aplikácia dokáže reagovať na požiadavky používateľa o interakciu so scénou. V súlade s dokumentáciou Gamepad API [18], je toto najjednoduchší a konzistentný spôsob prístupu k signálom z ovládačov. Gamepad API pozostáva z troch rozhraní, a to rozhranie reprezentujúce pripojený ovládač, rozhranie reprezentujúce tlačidlo na ovládači a rozhranie zastupujúce spustené eventy súvisiace s ovládačom. Ďalej obsahuje eventy, ktoré sa spustia pri pripojení alebo odpojení ovládača a špeciálnu funkciu, ktorá vráti reťazec objektov pre každý pripojený ovládač. Teoreticky sa to dá využiť na vytvorenie A-Frame

komponentu, ktorý upozorní používateľa na to či je ovládač pripojený alebo odpojený. Navyše sa v komponente môže napísať funkcia umožňujúca používateľovi pohyb a interakciu s virtuálnym svetom po stlačení určitého tlačidla na ovládači, čo bude odsledované spomínanými eventami. Takáto funkcionalita dokáže používateľom výrazne zlepšiť zážitok z kolaborácie vo virtuálnej realite.

2.3. Kolaborácia vo virtuálnej realite

V súčasnosti existuje obrovské množstvo aplikácií pre jedného používateľa so širokým spektrom využitia, avšak ešte väčší potenciál má využitie zmiešanej reality pre aplikácie založené na spolupráci používateľov. Moderná technológia na spoluprácu vo virtuálnom prostredí má svoje nedostatky hlavne čo sa týka interakcie s reálnym obsahom v priestore. Používatelia pri osobnej kolaborácii, v rámci zdieľaného fyzického pracovného priestoru, využívajú čo najjasnejšie spôsoby vzájomnej komunikácie, napríklad reč, gestá alebo iné neverbálne spôsoby komunikácie. V porovnaní s imerzívnymi virtuálnymi prostrediam, zmiešaná realita umožňuje používateľom prezerať si a využívať reálne objekty a čo je dôležitejšie, účastníci dokážu vidieť navzájom svoje výrazy tváre a reč tela, čo redukuje požiadavky na grafické vykresľovanie. Takáto kolaborácia môže byť pre účastníkov intuitívna a priaznivá, pretože podporuje prirodzenú ľudskú komunikáciu. Naopak pri vzdialenej kolaborácii, kde účastníci nezdieľajú rovnaký fyzicky priestor, môže dôjsť k nedostatku prirodzenosti komunikácie. Na vyriešenie tohto problému je potrebné v aplikácii zabezpečiť čo najvernejšie virtuálne reprezentácie vzdialených účastníkov, ktoré by boli schopné prirodzenej komunikácie. To si vyžaduje vyriešiť prenos informácií o reálnom pohybe a pohľade používateľa do virtuálnej podoby, aby ostatný účastníci dokázali odhadnúť čo sa daný jedinec chystá urobiť a vedeli správne reagovať na tento podnet. Navyše to značne navyšuje požiadavky na grafiku. Táto kapitola sa teda bude venovať kľúčovým vlastnostiam prostredí vhodných na kolaboráciu aj s ich odôvodnením a rizikami, ktoré prináša ich absencia. Okrem toho rozoberie možné využitia trojrozmerných interaktívnych scén za účelom kolaborácie v rôznych odvetviach. Z pohľadu edukácie poukazuje napríklad na využívanie kolaboračných prostredí zamerané na výučbu budúcich lekárov alebo na zaúčanie montáži komplikovaných strojov. Rovnako analyzuje aj využitie virtuálnej spolupráce v rámci pracovných konferencií a porovná výhody takéhoto riešenia.

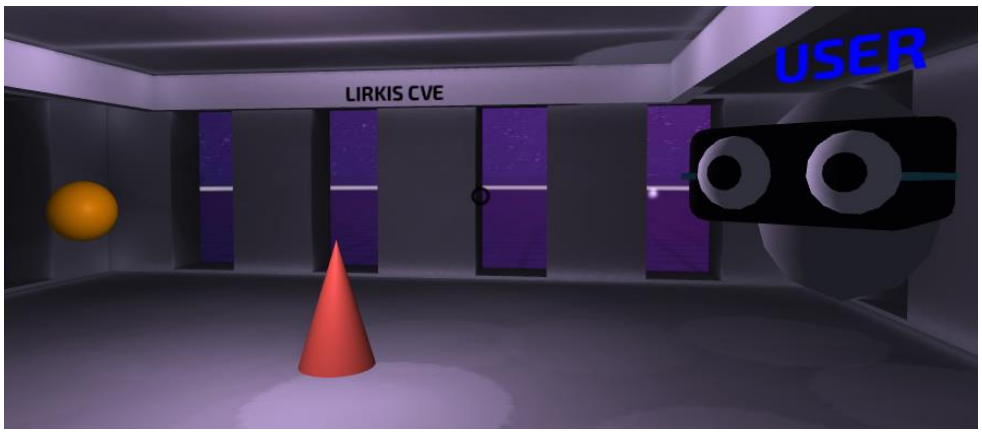
2.3.1. Vlastnosti kolaboračných prostredí

Ako spomína Billinghurst a Kato vo svojom článku o kolaborácií v rozšírenej realite [19], existuje päť kľúčových vlastností prostredí pre určených na spoluprácu používateľov:

- Virtualita
- Augmentácia
- Spolupráca

- Nezávislosť
- Individualita

Virtualita je vlastnosť fyzicky neexistujúcich, virtuálnych objektov, ktoré pôsobia reálne. V kolaboračnom virtuálnom prostredí táto vlastnosť predstavuje možnosť preskúmania virtuálnych objektov všetkými používateľmi. Na rozdiel od virtuality, augmentácia predstavuje skutočné objekty vo fyzickom svete, ktoré sú pomocou virtuálnej reality rozšírené o virtuálne anotácie. Obohatenie fyzických objektov takýmto spôsobom sa obvykle uskutočňuje v reálnom čase a poskytuje tak používateľom dodatočné informácie k reálnemu objektu. Ďalšou kľúčovou vlastnosťou je spolupráca, ktorej základom je dispozícia používateľa vidieť všetkých účastníkov v kolaboračnom prostredí a spolupracovať s nimi prirodzeným spôsobom. V prípade kolaborácie v zdieľanom fyzickom priestore je prirodzená komunikácia ľahšie dosiahnuteľná, pretože účastníci navzájom vidia reč tela a výraz tváre ostatných. Vo virtuálnom vzdialenom prostredí je oveľa náročnejšie dosiahnuť požadovanú úroveň komunikácie z hľadiska implementácie. Takéto prostredia musia mať zabezpečené nahradenie používateľa jeho virtuálnou reprezentáciou, inak nazývanou avatar. Pre správnu spoluprácu vo virtuálnej realite je strategické aby avatar kopíroval pohyby používateľa a zabezpečil interakciu so scénou. Na obrázku (Obr. 7) sa nachádza jednoduché vzdialené virtuálne prostredie s dvoma používateľmi. V tomto prostredí je možné odsledovať len polohu používateľovho avatara a smer ktorým sa pozerá.



Obr. 7 Virtuálne kolaboračné prostredie s avatarom účastníka

Štvrtou vlastnosťou je nezávislosť. Tá reprezentuje schopnosť každého používateľa ovládať svojho vlastného avatara v scéne a tým sa pozerať na virtuálne objekty z rôznej perspektívy. Ak by takúto schopnosť účastníci pri kolaborácii nemali, znížilo by to ich úspešnosť pri riešení spoločných problémov a dosiahnutí cieľov kolaborácie. Poslednou kľúčovou vlastnosťou je odlišnosť v zobrazených údajov pre každého používateľa, individualita. Je podstatná hlavne v aplikáciách virtuálnej reality, ktoré si vyžadujú aby boli používateľom zobrazené individuálne údaje. Takouto aplikáciou je napríklad konferencia spolupracovníkov, kde jeden má rolu moderátora a ostatní sú

len pozorovatelia. Moderátorovi by sa zobrazili okrem spoločných údajov aj individuálne, ktorými by mohli byť napríklad tlačidlá na posúvanie medzi snímkami prezentácie. Pozorovateľom by sa tieto tlačidlá nezobrazovali z dôvodu neúmyselného narušenia prezentácie. Všetky spomínané kľúčové vlastnosti je rovnako možné aplikovať aj na zmiešanú realitu, ako je uvedené v ďalšom článku o kolaborácii v zmiešanej realite od Billinghursta a Hirokazua [20].

2.3.2. Využitie virtuálnej kolaborácie

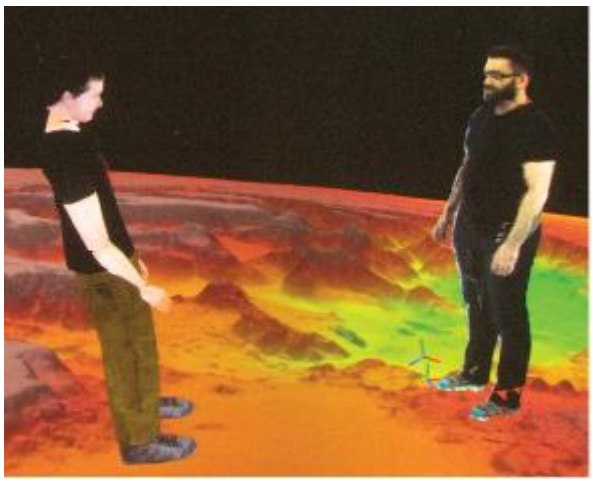
Virtuálna kolaborácia umožňuje rozvíjať edukáciu v rôznych smeroch a odvetviach, zabezpečením bezpečného simulovaného prostredia. Hansenova tvrdí [21], že virtuálne prostredia takýmto spôsobom poskytnú dobrý pedagogický potenciál všetkým študentom zvýšením ich produktivity a znížením úzkosti z náročných procesov edukácie. Rovnako má virtuálna kolaborácia potenciál vo všetkých odvetviach, či už sa to týka zdravotníctva alebo výskumu vesmíru, a jej zahrnutím v praxi dokáže zlepšiť kvalitu života mnohým ľuďom. Vhodným príkladom z odvetvia zdravotnej starostlivosti je globálny nedostatok odborných neurochirurgov a prostriedkov na ich spoluprácu pri edukácii alebo reálnych zákrokoch. Technológia, ktorá by zabezpečila vzdialenú kolaboráciu a poradenstvo skúseného chirurga s miestnym chirurgom v reálnom čase, má preto vysoký potenciál. Takúto technológiu, nazývanú virtuálna interaktívna prítomnosť a rozšírená realita, opisuje aj Davis a jeho spoluautori [22]. Je to nástroj, ktorý využíva iPad na bezdrôtové pripojenie na internet, prostredníctvom ktorého je umožnená kolaborácia na veľké vzdialenosti. Vzdialeným používateľom sa zobrazia dôležité snímky alebo videá z chirurgickej operácie v reálnom čase a tak dokážu poskytnúť odbornú verbálnu a vizuálnu pomoc miestnemu chirurgovi. Vysoké rozlíšenie obrazu a jeho bezdrôtový prenos za sterilných podmienok odborníkom zabezpečí zlepšenie procedurálnej bezpečnosti a účinnosti. Nedostatok odborníkov má negatívny vplyv na pozornosť venovanú študentom a na celkovú kvalitu výučby nie len v zdravotníctve. Tento problém je najviac viditeľný v praktickej výučbe študentov kde je potrebný osobný prístup a dohľad nad výučbou. Preto sa čoraz častejšie hľadajú efektívne alternatívne spôsoby bezpečnej výučby. Martín-Gutiérrez a spol. [23] opisujú takýto spôsob výučby prostredníctvom aplikácie, slúžiacej ako asistent, ktorý postupne prevedie študentov inštrukciami na manuálne splnenie úlohy zadanej pedagógom v laboratóriu. Tento asistent zobrazuje animácie trojrozmerných modelov umiestnených na hlavných paneloch pracoviska elektro-technického laboratória (Obr. 8).

Obr. 8 Elektro-technické laboratórium a príprava na plnenie virtuálnej úlohy [23]

Animácie ukazujú študentom ako správne pripojiť vodiče a umiestniť komponenty ako napríklad cievky, rotor alebo magnety, na vytváranie inštalácií a konfigurácií rôznych elektrických strojov. Pri tomto procese je možná kolaborácia študentov navzájom alebo spolupráca s odborným asistentom za účelom zvýšenia kvality výučby a odhodlania študentov v edukácii s prípadným zlepšením výkonu študenta. V súvislosti s bežným kancelárskym pracovným prostredím boli vyvinuté nápady na kolaboráciu vzdialených pracovníkov prostredníctvom konferencie. S týmito nápadmi prišli Billinghurst a Hirokazu. Prvotné rozhrania boli vyvíjané s cieľom preskúmať ako by bolo možné vzdialene spolupracovať a navzájom sledovať smer pohľadu a reč tela ostatných účastníkov virtuálnej konferencie. To dosiahli premietaním skutočného pracovného priestoru a samotných používateľov ostatným zúčastneným (Obr. 9). Používatelia mali možnosť rozložiť si ostatných projekcie v rámci svojho skutočného kancelárskeho priestoru, čo im poskytlo možnosť potenciálne viesť konferenciu na rôznych miestach, nie len v kancelárii. Výsledky týchto návrhov predstavujú víziu lepších virtuálnych konferencií so vzdialenými používateľmi a s možnosťou kolaborácie na firemných projektoch.

Obr. 9 Vzdialená kolaborácia s video avatarmi účastníkov [19]

Fairchild a spol. zasa popisuje vo svojom článku [24] systém zmiešanej reality, ktorý je výsledkom integrácie technológie telepresencie a vyvíjanej aplikácie s cieľom zlepšenia spolupráce pri výskume vesmíru. Tento systém podporuje neverbálnu komunikáciu, vrátane pohľadu očí a výraze tváre, prostredníctvom kombinácie imerzívnej projekčnej technológie a free viewpoint videa. Strategická je interpretácia týchto vlastností systému v reálnom čase simulácie vesmíru. Vyvíjanou aplikáciou je konkrétne simulátor povrchu Marsu s podporou interaktívnych geologických nástrojov, slúžiacou na kolaboráciu medzi vzdialenými účastníkmi v imerzívnom virtuálnom prostredí pri skúmaní povrchu Marsu (Obr. 10). V článku sa uvádza, že každý účastník spúšťa inštanciu simulátora s lokálne uloženými údajmi o Marse. Interakcia používateľa so systémom je potom synchronizovaná technológiou manažér kolaborácie. V predvolenej konfigurácii aplikácie sa na reprezentáciu vzdialených používateľov využíva štandardný avatar vytvorený technológiou CGI. Avšak simulátor Marsu rozšírili neskôr o podporu trojrozmerných video avatarov. To si vyžadovalo v systéme zahrnúť komunikačný modul na príjem trojrozmernej siete a video streamu tak ako aj modul vykresľovania na vizualizáciu trojrozmerného avatara v simulácii povrchu Marsu. Tento prístup by bolo možné implementovať v mnohých ďalších oblastiach, ktoré si vyžadujú vzdialenú spoluprácu ľudí pri diskusii o priestorových modeloch na ktoré je potrebné pozerať sa z rôznych uhlov, najmä ak sú súčasťou konverzácie emócie účastníkov. Spoločné rozhodovanie a riadenie pohotovostných služieb v prípade katastrofy je dobrým príkladom situácie, keď je priestorový kontext a ľudské emócie potrebné konzultovať spolu. Ďalším vhodným príkladom môže byť zdravotníctvo a potreba diaľkovej terapie s dôležitosťou porozumenia neverbálnych spôsobov komunikácie počas liečby. Využitie takýchto aplikácii v praxi môže spájať ľudí znížením potreby cestovania na veľké vzdialenosti a tak celkovo zlepšiť kvalitu života.



Obr. 10 Kolaborácia pri skúmaní povrchu Marsu s využitím telepresencie [24]

3. Návrh a implementácia klientskej časti systému G-CVE

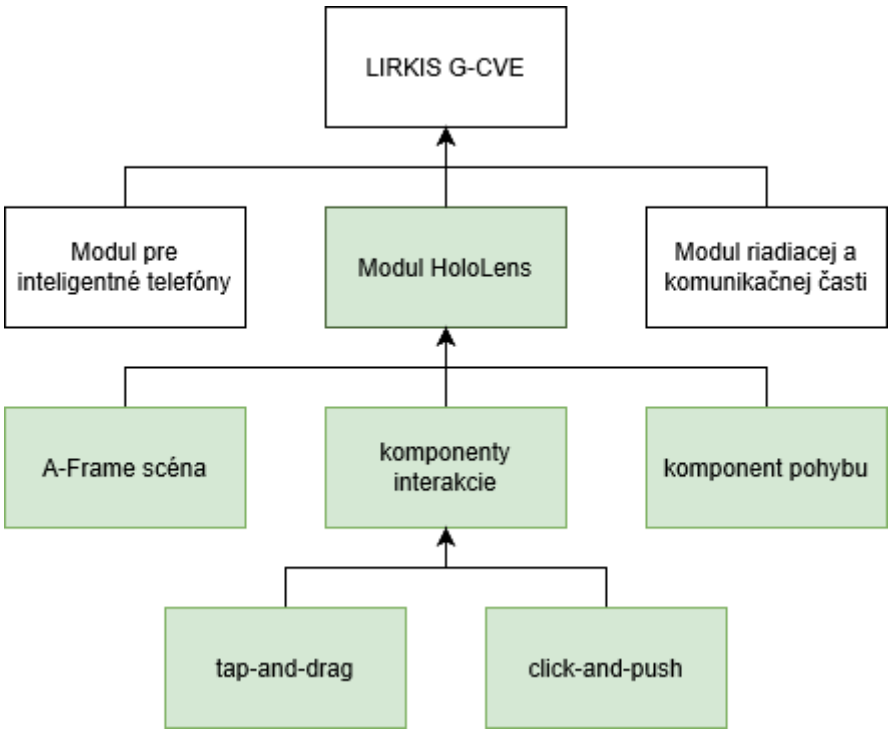
Jedna z kľúčových výziev v technologickom napredovaní zmiešanej reality sa týka navrhovania kolaboratívnych skúseností. Systémy virtuálnej reality celkom nepodporujú tímovú spoluprácu, pretože majú tendenciu zameriavať sa na individuálne skúsenosti používateľa a neuľahčujú prirodzenú komunikáciu používateľov. Prostredia zmiešanej reality poskytujú lepšie riešenia pre spoluprácu, avšak pri vytváraní jednoduchej hladkej komunikácie medzi hardvérom a softvérom môže dôjsť k vzniku veľkých prekážok [25]. V tejto kapitole sa rozoberú kľúčové výzvy pri návrhu diplomového projektu s primárnym zameraním na riešenie technických výziev. Hlavnou výzvou dizajnu bolo vytvoriť prostredie pre tímovú spoluprácu viacerých účastníkov so schopnosťou vzájomnej interakcie. Správanie účastníkov by malo byť čo najviac prirodzené v rámci pohybu po scéne a interakcie s objektami. Na zostavenie takejto aplikácie bolo potrebné navrhnúť integrované, dynamické a nákladovo efektívne riešenie pre sledovanie pozície a rotácie každého používateľa a pre sledovanie vstupov zo všetkých ovládačov kvôli interakcii so scénou.

V úvode kapitoly je opísané spoločné demo systému G-CVE, ktoré bolo vyvíjané v spolupráci s tímom riešiteľov systému na kolaboráciu vo virtuálnej realite. Toto demo bude opísané hlavne z pohľadu používateľa so zariadením HoloLens. Druhá podkapitola sa venuje návrhu a implementácii vizuálnej reprezentácie používateľov so zariadením Microsoft HoloLens. Avatar sa rozoberie z hľadiska jeho zloženia a funkcionality. V rámci zloženia sa opíše z akých častí avatar pozostáva, aký je ich vzájomný vzťah, teda hierarchia a vzťah vzhľadom k scéne. Opíšu sa aj vlastnosti týchto častí, ktoré ovplyvňujú vzhľad avatara, napríklad materiál, veľkosť a farba. Okrem toho sa podkapitola 3.2 venuje aj funkcionalite avatarov, ktoré komponenty sú potrebné na zabezpečenie pohybu a rotácie avatara v scéne, a ktoré pre lepšiu skúsenosť z kolaborácie pre samotného používateľa. Posledná podkapitola obsahuje návrh a implementáciu dvoch komponentov určených na interakciu s objektami v scéne. Prvým komponentom je zdrojový kód napodobujúci hlavnú charakteristickú interakciu zariadenia Hololens. Tou je využívanie zloženého gesta Air Tap, teda navigačného a manipulačného gesta na pohybovanie s hologramami vo svojom okolí. Podstatou druhého komponentu interakcie je využívanie raycastera a priesečníka s objektami na ich naviazanie na kurzor a následné uvoľnenie vykonané pri určitej udalosti. Touto udalosťou bude stlačenie tlačidla na ovládači HoloLens Clicker.

3.1. Kolaboratívne prostredie G-CVE

Cieľom spoločnej práce bolo navrhnúť a implementovať virtuálne globálne prostredie vhodné pre kolaboráciu viacerých používateľov s rôznymi zariadeniami. Toto prostredie podporuje prihlásenie klienta troma spôsobmi, a to ako admin, používateľ alebo z pohľadu robota. Po prihlásení sa

používateľ ocitne v scéne obsahujúcej model miestnosti, v ktorom sa používatelia pohybujú. Rovnako sa v scéne môže pohybovať aj robot, v prípade ak je nejaký klient prihlásený týmto spôsobom. Na Obr. 11 sú zobrazené moduly, ktoré boli vyvíjané s cieľom umožniť používateľovi plne využiť prostredie LIRKIS G-CVE.



Obr. 11 Moduly systému LIRKIS G-CVE

V rámci tejto diplomovej práce je riešený len modul klientskej časti pre zariadenie Microsoft HoloLens, ktorý umožňuje klientovi prihlásiť sa do scény len ako používateľ. Súčasťou tohto modulu sú tri hlavné zložky:

- html súbor, ktorý obsahuje A-Frame scénu – *hololens.html*
- javascript súbor s implementovanou interakciou – *hololens-interaction.js*
    - komponent interakcie - *tap-and-drag*
    - Komponent interakcie - *click-and-push*
- javascript súbor s implementovaným pohybom avatara – *hololens-postition.js*

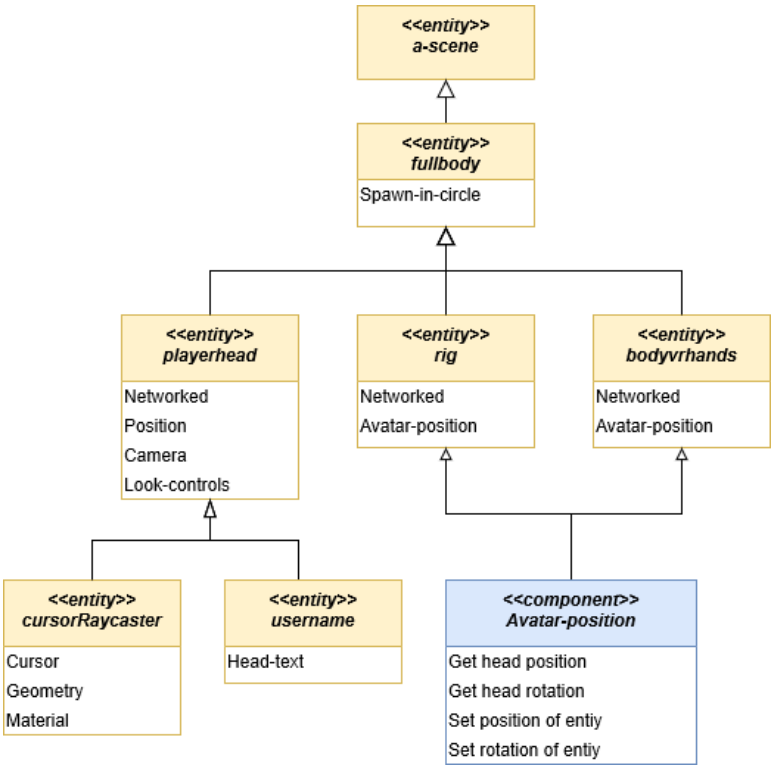
Súbor s názvom *hololens.html* obsahuje implementáciu scény a všetkých potrebných prvkov pre správne zobrazenie prostredia G-CVE pre zariadenie HoloLens. Tento súbor sa líši od hlavného súboru *index.html* a html súboru pre používateľov s mobilnými zariadeniami, hlavne importovanými skriptami, využitou A-Frame verziou a štruktúrou avatara. Pri spustení tohto prostredia je používateľ presmerovaný z hlavného html súboru na stránku určenú pre zariadenie HoloLens pomocou kontroly využívaného prehliadača. V prípade tohto zariadenia to bude stále

prehliadač Microsoft Edge. Pre vylúčenie ostatných zariadení s týmto prehliadačom sa ešte využije dialógové okno, ktoré sa spýta klienta či používa zariadenie HoloLens. Po presmerovaní sa začne využívať A-Frame verzia 0.9.0. ktorá sa doposiaľ ukázala ako najstabilnejšia pre naše zariadenie. Rovnako sa bude využívať verzia avatara prispôsobená používateľovi so zariadením Hololens. Návrh a implementácia avatara, ako aj riešenie pohybu a rotácie sú uvedené v nasledujúcej kapitole 3.2. Okrem toho sa v našom html súbore nachádzajú všetky environmentálne objekty z ktorých scéna pozostáva a objekty určené na interakciu s používateľom. Práve na tieto objekty sú naviazané komponenty implementované v Javascript súbore s názvom *hololens-interaction.js*. Súčasťou interakcie je aj implementovaná fyzika. Preto je veľmi potrebné mať v scéne podlahu cez ktorú sa objekty s fyzikou neprepadnú. Navrhnuté druhy interakcie sú bližšie opísané v kapitole 3.3.

3.2. Avatar používateľa

Používateľ si uvedomuje prítomnosť svojho tela v zmiešanej realite, avšak pre kolaboratívne prostredie je dôležitou súčasťou aj grafická reprezentácia pre ostatných používateľov. Pre zvýšenie používateľského zážitku je potrebné zabezpečiť zobrazovanie polohy, rotácie alebo dokonca aj akcie rúk používateľov. Existuje niekoľko spôsobov akými dokážeme navrhnúť a reprezentovať používateľa vo virtuálnej realite. V kapitole 2.3.2 je spomenuté využitie niektorých typov avatarov v špecifickom prostredí, avšak pre vyvíjaný systém sa zvolila jednoduchá reprezentácia avatara pomocou trojrozmerných geometrických útvarov. Samotný návrh jednoduchej verzie avatara je zložený z troch základných častí reprezentovaných geometrickými útvarmi: hlava, telo a ruky.

Je dôležité zvýrazniť fakt, že používateľ so zariadením HoloLens si vidí svoje vlastné telo a ruky, a tým pádom aj akcie, ktoré s nimi vykonáva. Preto je potrebné zvážiť odstránenie rúk a tela avatara pre takéhoto používateľa. Avšak pre ostatných používateľov v kolaboračnom prostredí je vhodné zobraziť celého avatara, bez ohľadu na zariadenie s ktorým sa do prostredia pripojil. Z tohto dôvodu je avatar navrhnutý zo spomínaných troch častí. Na Obr. 12 môžeme vidieť hierarchiu html entít vytvárajúcich avatara, ich zloženie a najdôležitejšie komponenty ovplyvňujúce rôzne faktory, hlavne funkcionalitu avatara v scéne.



Obr. 12 Diagram hierarchie html entít avatara

Prvým základným objektom je html entita predstavujúca celého avatara s identifikátorom **fullbody** a s komponentom *spawn-in-circle*. Tento komponent môže, ale nemusí byť v scéne zahnutý. Zabezpečuje objavenie sa pripojených avatarov na náhodnom mieste v rámci kruhu, ktorého rádius je vstupnou vlastnosťou tohto komponentu. Dôvodom je zamedzenie objavovania sa používateľov na rovnakom mieste, teda ich prekrývanie sa pri vstupe do scény. Táto funkcionalita sa dá obmedziť nastavením nulového rádiusu, čo bude znamenať, že používatelia sa zjavia na rovnakom mieste. Potomkami tejto prvej rodičovskej html entity sú spomínané tri časti avatara, teda hlava **playerhead**, telo **rig** a ruky **bodyvrhands**. Najdôležitejšou časťou je hlava, pretože zariadenie Microsoft HoloLens získava údaje z reálneho pohybu a orientácie používateľovej hlavy, a teda tieto údaje je vhodné preniesť aj na hlavu avatara. Pre dosiahnutie lepšieho výkonu sa na tvorbu avatara využil A-Frame systém na správu prostriedkov (materiály, obrázky, trojrozmerné objekty...), ktoré je možné vopred načítať a uložiť do vyrovnávacej pamäte. Je označený tagom *<a-assets>.* V rámci tohto systému bola vytvorená aj šablóna pre avatara, v ktorej sú špecifikované zdroje trojrozmerných objektov pre každú časť tela, ich hierarchia, škálovateľnosť, pozícia, rotácia, farba, nepriehľadnosť a dokonca aj jednoznačný identifikátor. Táto šablóna je označená tagom *<a-template>.*

```
<a-entity
        id="playerhead"
        networked="template:#avatar-template; attachTemplateToLocal:false;"
        camera
        position="0 1.6 0"
        look-controls >
        <a-entity
                cursor="fuse: true; fuseTimeout: 500"
                position="0 0 -1"
                geometry="primitive: ring; radiusInner: 0.01; radiusOuter: 0.018"
                material="color: white; shader: flat"
        > </a-entity>
        <a-text class="username" visible="true" head-text></a-text>
</a-entity>
```

Obr. 13 Implementácia hlavy avatara

Entita predstavujúca používateľovu hlavu má okrem komponentu na nastavenie pozície ďalšie tri hlavné komponenty: *networked, camera* a *look-controls* (Obr. 13). Komponent *networked*, ktorý funguje na princípe synchronizácie entít a ich komponentov s pripojenými používateľmi, je dôležitý pre funkčnú kolaboráciu. Pre jeho správne fungovanie musí samotná scéna obsahovať komponent sieťovej scény. Keďže html entita hlavy je jedným z objektov, ktorých pozíciu a rotáciu chceme synchronizovať so všetkými účastníkmi, tak obsahuje práve komponent s dvoma parametrami. Prvý parameter je *template*, s hodnotou jednoznačného identifikátora odkazujúceho na šablónu hlavy. Druhým parametrom je *attachTemplateToLocal*. Tento parameter dokáže pripojiť šablónu pre lokálneho používateľa, čo je v našom prípade nežiaduce, pretože nechceme aby si používateľ videl hlavu vlastného avatara. Z tohto dôvodu má parameter priradenú hodnotu false. Ďalším komponentom objektu hlavy je kamera, ktorá nemá nastavené žiadne parametre. Je pre systém relevantná, pretože definuje perspektívu sledovania scény používateľom. Pozícia objektu na ktorom je tento komponent naviazaný je zhruba vo výške očí. Pre umožnenie rotácie a pohybu kamery sa jej komponent spáruje s ovládacím komponentom *look-controls.* Je určený primárne na rotáciu entity v prípade ak je zmenená orientácia používateľovho zariadenia. Čo sa týka zmeny pozície, tento komponent obsahuje vlastnosť *hmdEnabled*, ktorá má predvolenú hodnotu true. To značí, že sa bude meniť okrem rotácie aj pozícia entity, s pozíciou používaného zariadenia MS HoloLens. Avšak funkcionalita tohto komponentu sa spustí až po zapnutí virtuálneho imerzívneho módu prehliadača. Okrem týchto častí má html entita hlavy aj dvoch potomkov: html entita kurzora **cursorRaycaster** a používateľského mena **username**. Html entita kurzor je viditeľná pre každého používateľa zvlášť. Obsahuje tri vedľajšie komponenty (*position, geometry, material*) a jeden

hlavný (*cursor*). Komponent *cursor* je založený na komponente *raycaster* a poskytuje pre používateľa možnosť interakcie pohľadom. Okrem toho dokáže odpočúvať udalosti a zachytiť priesečník s najbližšou viditeľnou entitou. Pomocou komponentu *position* je mu priradená pozícia jednej jednotky vzdialenosti pred používateľovou hlavou, aby bol dobre viditeľný. Prsteňový tvar, rádius a farba kurzora sú nastavené cez komponenty *geometry* a *material*. Ďalším potomkom html entity hlavy je text, ktorý obsahuje meno používateľa a je zobrazený len ostatným používateľom. Obsahuje komponent na získavanie a nastavovanie mena. Slúži na rozpoznanie účastníkov scény. Celkovo vyzerá implementácia avatarovej hlavy pred implementáciou interakcie ako je zobrazená na Obr. 13. S implementovanou interakciou sa čiastočne pozmení kurzor avatara, čo je opísané v kapitole 3.3.

```
<a-entity id="bodyvrhands"
networked="template:#avatar-bodyhands;attachTemplateToLocal:false;"
avatar-position ></a-entity>
<a-entity id="rig"
networked="template:#avatar-body;attachTemplateToLocal:false;"
avatar-position ></a-entity>
```

Obr. 14 Implementácia rúk a tela avatara

Ďalšími dvoma časťami avatara sú entity predstavujúce telo a ruky. Obe html entity majú rovnaké komponenty *networked* a *avatar-position.* Rovnako ako pri hlave, komponent *networked* má parameter *template*, s hodnotou jedinečného identifikátora referujúceho na šablónu (pre každú časť tela je to iná šablóna) a parameter *attachTemplateToLocal*, ktorý je taktiež nastavený ako false. Najzaujímavejšou časťou týchto entít je komponent *avatar-position* (Obr. 15), ktorý získava údaje o pohybe a orientácii používateľovej hlavy. Následne využíva tieto údaje aj na nastavenie pozície a rotácie rúk a tela. Pozícia v rámci trojrozmerného priestoru je podľa údajov z hlavy menená rovnako len po osiach x a z. Z pozície na osi y sa odrátava určitá hodnota aby bolo telo vo správnej výške v závislosti od hlavy. Čo sa týka otáčania tela a rúk, využijú sa len súradnice rotácie okolo osi y, pretože telo avatara by sa nemalo otáčať iným smerom.

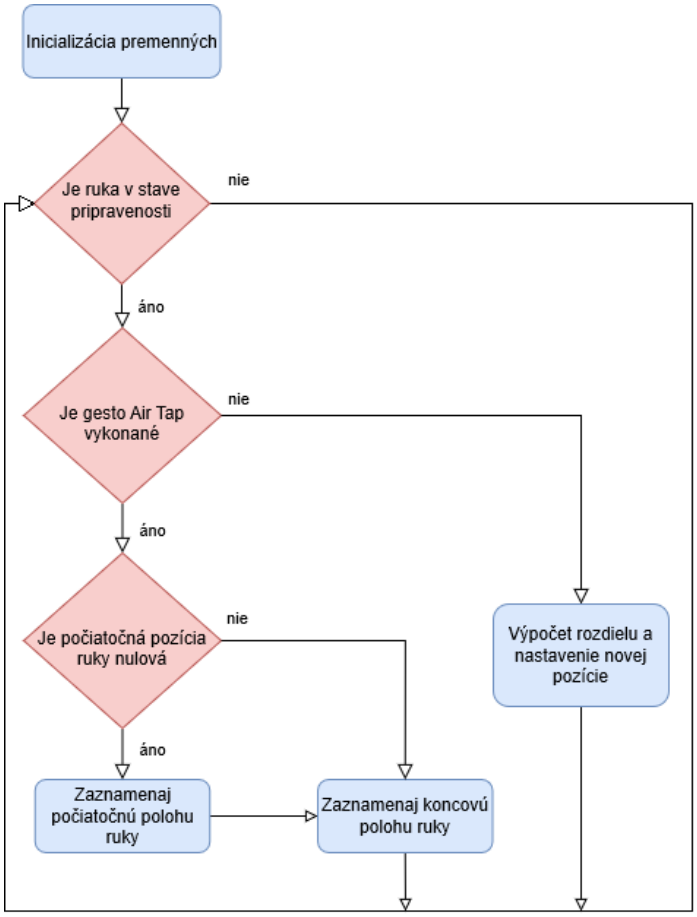
```
tick: function() {
    rot = player_head.getAttribute("rotation");
    pos = player_head.getAttribute("position");
    el.setAttribute("position", { x: pos.x, y: pos.y - 1.6, z: pos.z });
    el.setAttribute("rotation", { y: rot.y });
}
```

Obr. 15 Implementácia komponentu avatar-position

3.3. Spôsoby interakcie

Interakcia pohľadom je najjednoduchšou a primárnou formou manipulácie s objektami v prostredí zmiešanej reality. Podobne ako v reálnom svete, tak aj vo virtuálnom človek pozerá na objekt s ktorým chce pracovať. Využitím toho faktu je možné vytvoriť jednoduchú odozvu systému vyvolanú pohľadom používateľa. Na to je však potrebné zistiť, na ktorý virtuálny objekt sa používateľ pozerá. Dokumentácia A-Frame obsahuje potrebný komponent, ktorý s využitím metódy raycasting poskytuje informácie o pretínaní pomyselnej čiary, vedenej od entity istým smerom, s inými entitami. Avšak len na základe detegovaných údajov o priesečníkoch ešte nevieme zistiť, na čo sa používateľ pozerá, pretože si nemusí byť vedomý toho, ako raycasting funguje. Preto A-Frame poskytuje ďalší komponent, viac prirodzenejší pre človeka, nazývaný kurzor. Je vytvorený na základe raycaster komponentu a poskytuje systému informácie o interakcii pohľadom s inými entitami. Kurzor môže byť ukotvený v strede obrazovky a tým používateľovi jasne dávať najavo, že systém bude reagovať na interakciu pohľadom, ak sa bude pozerať na entitu práve cez viditeľný kurzor. Pre ešte lepšiu spätnú väzbu pre používateľa je možné navrhnúť a pridať animácie na entitu kurzor, ktoré budú meniť jeho vzhľad v závislosti od udalosti, ktorú vyvolá.

Charakteristickou súčasťou zariadenia Microsoft HoloLens je jeho interakcia s hologramami pomocou snímania gest rúk. Je to rýchly a efektívny spôsob práce so zmiešanou realitou bez využitia ďalšieho príslušenstva. Z tohto dôvodu bol navrhnutý prvý skript na interakciu, ktorý dokáže spracovať údaje zo snímania používateľových rúk a umožní tak interakciu s využitím gest. Časť návrhu a implementácie tohto skriptu, týkajúca sa spôsobu získavania údajov z rúk, ktoré sa správajú ako ovládač, je uvedená v kapitole o testovaní interakcie 4.4. Princípom tohto komponentu s názvom tap-and-drag je využívanie a sledovanie polohy rúk v trojrozmernom priestore, sledovanie odohrania gesta Air Tap (stlačenie tlačidla) a nastavovanie novej polohy objektu na základe jednoduchých výpočtov. Grafické znázornenie jednotlivých krokov tohto skriptu je stvárnené na Obr. 16 pomocou vývojového diagramu. Na jeho základe vysvetlíme jednotlivé časti a procesy potrebné na funkčnosť komponentu tap-and-drag.



Obr. 16 Vývojový diagram interakcie s využitím gesta Air Tap

Prvým symbolom diagramu je čiastkový krok s popisom *Inicializácia premenných*. Týmto krokom sa inicializujú dve premenné používané ďalej v kóde a nastaví sa im hodnota null. Prvá premenná *handStartPosition* označuje polohu ruky v momente stlačenia tlačidla, teda vykonania gesta Air Tap. Druhá premenná označuje polohu používateľovej ruky po uvoľnení tohto gesta. Všetky nasledujúce čiastkové kroky sa vykonávajú v slučke. Prvým symbolom slučky je podmienka s popisom *Je ruka v stave pripravenosti*. Tento stav je bližšie opísaný v kapitole 2.1.3. Keďže sa ruka správa ako ovládač, tento stav sa dá odkontrolovať metódou *gamepad.connected*, ktorá vráti hodnotu true, ak zariadenie zachytí prítomnosť alebo gesto pripravenosti ruky. V diagrame môžeme vidieť že ak ovládač nie je pripojený, nevykoná sa žiaden čiastkový kód. V prípade ak je pripojený, vykoná sa ďalšia podmienka s popisom *Je gesto Air Tap vykonané.* Táto podmienka kontroluje vykonanie tohto gesta pomocou metódy *gamepad.buttons[0].pressed*, ktorá vráti hodnotu true ak má používateľ stlačený ukazovák a palec. Pokiaľ je táto podmienka vyhodnotená ako pravdivá, vývojový diagram sa dostane do ďalšej podmienky. Avšak v opačnom prípade sa vykoná čiastkový proces s popisom *Výpočet rozdielu a nastavenia novej pozície,* ktorého logika bude opísaná neskôr. V tomto bode je dôležitá nasledujúca podmienka s popisom *Je počiatočná pozícia ruky nullová.* Tá preveruje hodnotu premennej *handStartPosition* či je rovná hodnote null. Ak áno, tak sa vykoná

kód na zistenie pozície ruky v priestore a uloženie tejto hodnoty do premennej označujúcej počiatočnú pozíciu. Následne sa zistí a uloží hodnota pozície ruky do premennej pre koncovú pozíciu *handEndPosition*. V prípade že premenná *handStartPosition* má už zaznamenanú pozíciu ruky, zaznamenáva sa táto pozícia len do premennej *handEndPosition*. Celé kontrolovanie a značenie polohy ruky prebieha len ak je vykonávané gesto Air Tap. Po uvoľnení gesta sa vykoná posledný čiastkový kód v diagrame. Ten kontroluje opäť hodnotu premennej pre začiatočnú pozíciu ruky a vykoná výpočty len ak je v nej uložená pozícia ruky. Pre každú súradnicu trojrozmerného priestoru sa vypočíta rozdiel medzi koncovou a počiatočnou pozíciou ruky. Ako ďalšie sa získajú súradnice objektu na ktorý sa používateľ pozerá a k ním sa priráta vypočítaný rozdiel pre každú súradnicu zvlášť. Po presunutí objektu na novú pozíciu sa premenným pre uchovávanie pozícii ruky nastaví hodnota null. Po vykonaní tohto kódu sa opäť kontroluje stav pripravenosti ruky.

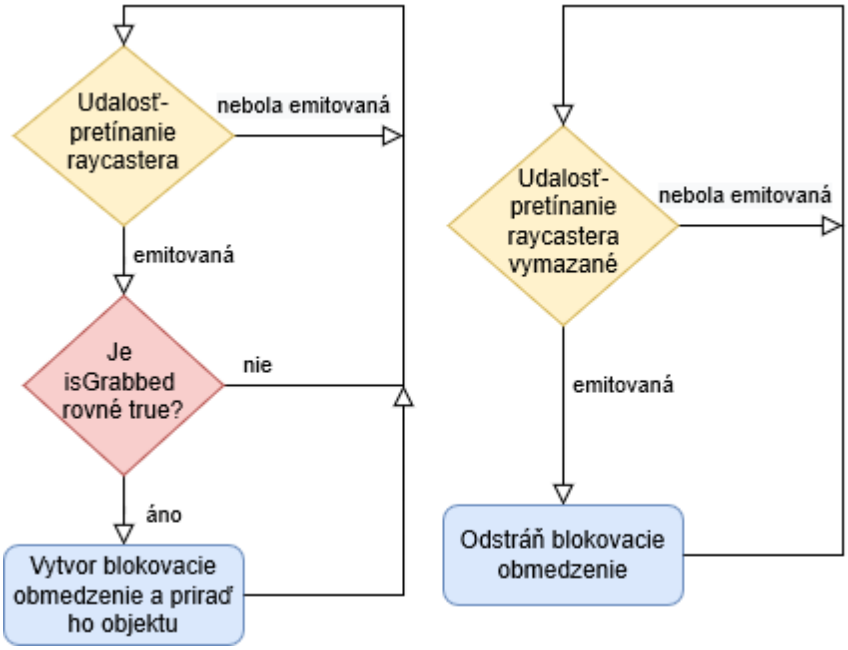
Podobne ako prvý komponent určený na interakciu, aj komponent s názvom click-and-push využíva sledovanie udalosti pretínania raycastera s objektom a udalosti zrušenia tohto pretínania. Rozdielom je však vykonanie kódu navyše pri spustení týchto udalostí a implementácia kurzora avatara (Obr. 17).

```
<a-entity
        cursor
        raycaster = "objects: .clickable; showLine: true; far: 500"
        visible = "false"
        position = "0 0 0"
></a-entity>
<a-entity
        body = "type: static; mass: 5; shape: sphere; sphereRadius: 0.02;"
        visible = "true"
        position = "0 0 -1"
        geometry = "primitive: ring; radiusInner: 0.008; radiusOuter: 0.01"
></a-entity>
```

Obr. 17 Zmena implementácie kurzora

Dôležitou zmenou je html entita kurzor, ktorej pribudol komponent *raycaster* na určenie priesečníka všetkých objektov s triedou *clickable* do vzdialenosti 500 jednotiek. Okrem toho sa pôvodný kurzor zneviditeľnil a posunul na inú pozíciu. Pridala sa nová entita, graficky reprezentujúca kurzor na jeho pôvodné miesto. Na ňu sa naviazal komponent body, ktorý priradil statickosť, tvar a hmotu entite v rámci fyziky. Táto entita slúži na vytváranie obmedzení, spomínaných nižšie v tejto kapitole. Vývojové diagramy, zobrazujúce postup vykonávania

rozdielneho skriptu tejto interakcie môžeme vidieť na Obr. 18. Dôležitými časťami tohto skriptu je premenná *isGrabbed* a blokovacie obmedzenie.



Obr. 18 Vývojové diagramy udalostí, zľava pretínanie raycastera, sprava vymazanie pretínania

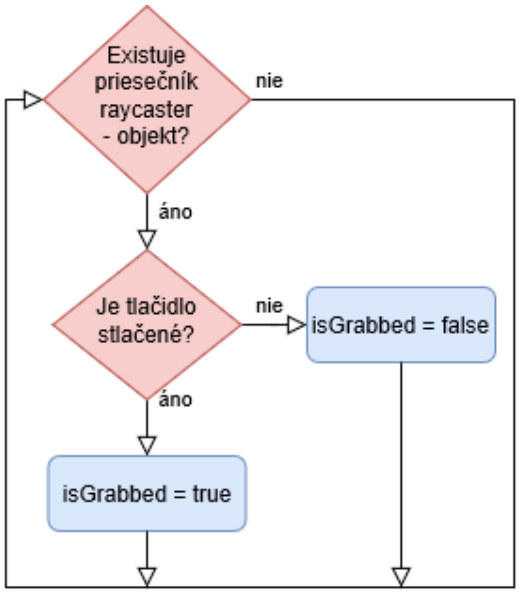
Pri implementácii tohto komponentu bola využitá knižnica *cannon.js*. Slúži na jednoduchú trojrozmernú fyziku pre aplikácie webových prehliadačov. Konkrétne sa z tejto knižnice využila funkcionalita na tvorbu fyzických obmedzení. Tie sú vytvárané a mazané na základe spustenia spomínaných udalostí. Implementácia tohto skriptu je zobrazené na Obr. 19.

```
if (isGrabbed == true) {
  this.constraint = new CANNON.LockConstraint(this.el.body, parent.body, {
          maxForce: 5000});
  this.system.addConstraint(this.constraint);
}
...
this.system.removeConstraint(this.constraint);
```

Obr. 19 Implementácia vytvárania, priradzovania a rušenia obmedzení

Pri spustení prvej udalosti, ktorá sleduje či sa používateľ pozerá na objekt sa vykoná kód, obsahujúci podmienku kontrolujúcu hodnotu premennej *isGrabbed*. Ak je jej hodnota rovná true, vytvorí sa nové fyzické blokovacie obmedzenie. Toto obmedzenie, zapísané kódom *CANNON.LockConstraint(this.el.body, parent.body, {maxForce: 5000})*, odstráni všetky stupne slobody medzi dvoma telami *this.el.body* (objekt na ktorý sa používateľ pozerá) a *parent.body*

(kurzor). Následne metóda *this.system.addConstraint(this.constraint)* pridá toto obmedzenie do sveta. Pre používateľa sa to javí ako prilepenie objektu na kurzor a hýbanie s objektom smerom, ktorým sa používateľ otočí. Pri spustení druhej udalosti, keď sa používateľ nepozerá na interaktívny objekt, sa vykoná metóda *this.system.removeConstraint(this.constraint)* na odstránenie obmedzenia medzi objektom a kurzorom zo scény. Potom sa už len implementovala logika, ktorá vytvárala tieto obmedzenia len v prípade ak tlačidlo na ovládači bolo stlačené. Vývojový diagram tejto logiky je zobrazený na Obr. 20. Ak sa používateľ pozerá na daný objekt, a tlačidlo je stlačené, nastaví sa premennej *isGrabbed* hodnota true. V prípade ak sa na objekt pozerá, ale tlačidlo nie je stlačené, nastaví sa tejto premennej hodnota false. To zabezpečí aby sa obmedzenia vytvárali len vtedy, ak si to používateľ vyžaduje.



Obr. 20 Vývojový diagram interakcie s využitím fyzických obmedzení

4. Návrh a implementácia testovacích prostredí

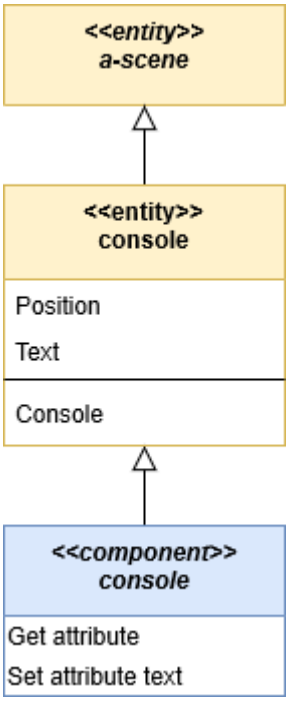
Súčasťou návrhu systému pre kolaboráciu vo virtuálnom prostredí s využitím zariadenia Microsoft HoloLens bolo nevyhnutné vytvorenie niekoľko testovacích prostredí určených na overenie konkrétnej funkcionality systému a následne jeho vyhodnotenie. Neoddeliteľnou súčasťou pri vývoji počítačových programov je aj ladenie, pretože tento metodický postup pomáha nachádzať a znižovať počet chýb v programe. Tieto chyby môžu byť logického alebo syntaktického pôvodu a často spôsobujú chybné správanie programu bez akejkoľvek indikácie, kde sa problém nachádza.

V tejto kapitole sa opíšu najhlavnejšie funkcie kolaboračného prostredia, ktoré bolo potrebné vyskúšať na cvičných scénach. Pre lepšie testovanie bola vytvorená náhrada webovej konzoly, ktorá poskytla možnosť získania presnejších výsledkov. Na základe výsledkov z testovania hlavných funkcionalít sa postupne vylepšoval celkový vyvíjaný systém. Medzi tieto cvičné prostredia patrí aj scéna s implementovaným strážcom, ktorý predstavuje pre používateľa ohraničenie scény v ktorej sa smie pohybovať. Vo virtuálnom svete to slúži primárne na ochranu používateľa, aby nevyšiel z bezpečného priestoru a nezranil sa v reálnom prostredí. Avšak v prostredí zmiešanej reality, ktoré nám HoloLens poskytuje takýto strážca neslúži ako zábrana, pretože používateľ vidí reálny svet a dokáže sa vyhýbať nebezpečenstvu. V tomto prostredí predstavuje strážca skôr ohraničenie veľkosti scény. Ďalším testovacím prostredím je jednoduchá scéna určená na meranie vzdialeností v reálnom a virtuálnom svete pri pohybe používateľa. Keďže používateľ, ktorý používa Microsoft HoloLens sa bude voľne pohybovať v reálnom svete, je dôležité aby sa jeho pohyb vo virtuálnom svete odrážal čo najvernejšie. Na základe meraní z tohto testovacieho prostredia je možné navrhnúť autentickejší pohyb v kolaboratívnom prostredí. Súčasťou testovacích prostredí je aj ďalšia scéna určená primárne na skúšanie interakcie s prvkami scény, či už s využitím pohľadu, gest rúk alebo HoloLens Clicker-om. Touto scénou je možné dosiahnuť prirodzenejšiu interakciu pre používateľa a tak zlepšiť jeho zážitok zo spolupráce v zmiešanej realite.

4.1. Spôsob ladenia systému

Ladenie webových aplikácií môže zahŕňať využitie externých testovacích aplikácií, alebo jednoduchší spôsob ladenia pomocou developerských nástrojov vbudovaných do webového prehliadača. Medzi tieto nástroje patrí konzola alebo debugger. Rozdielom je, že zatiaľ čo konzola umožňuje len vypísať nadobudnuté hodnoty kódu písaného v jazyku Javascript, debugger dokáže takýto kód aj pozastaviť vo vopred určených miestach. Problém ale nastáva pri použití týchto nástrojov cez zariadenie Microsoft HoloLens. Napriek tomu, že disponuje webovým prehliadačom Microsoft Edge, ktorý ladiace nástroje obyčajne má, nebolo ich na tomto zariadení možné aktivovať. Tento problém výrazne skomplikoval vývoj celej aplikácie, a preto bolo potrebné navrhnúť

jednoduchú náhradu za konzolu. Javascript metóda console.log(), ktorá vypíše správu do konzoly a je najčastejšie využívaná na testovanie je v navrhovanej náhrade nevyužiteľná, avšak princíp na akom funguje je aj tak možné využiť. Správa, ktorá by sa za normálnych okolností vypísala do konzoly webového prehliadača, ostane nezmenená. Zmeniť je potrebné len metódu a miesto výpisu. Najlepším spôsobom je výpis správ s nadobudnutými hodnotami priamo v scéne, takže je potrebné vytvoriť nejakú entitu, na ktorú sa budú tieto správy vypisovať. Táto entita by mala mať minimálne komponent text, ktorý by bolo možné meniť počas behu aplikácie a komponent, ktorý by určil jej tvar.



Obr. 21 Diagram hierarchie html entít pre konzolu

Podľa dokumentácie A-Frame, existuje presne taký jednoduchý element *<a-entity>* s komponentom pre zobrazovanie textu v scéne. Na Obr. 21 môžeme vidieť hierarchiu hlavných html entít a komponentov potrebných pre návrh konzoly. Prvou html entitou je samotná scéna, ktorá je nevyhnutná pre renderovanie všetkých jej potomkov. Ďalšiu html entitu predstavuje objekt konzoly **console**, ktorý obsahuje tri základné komponenty (Obr. 22). Prvým je pozícia, teda umiestenie konzoly v rámci scény. Druhým je text, ktorý bude menený a vypisovaný do scény podľa potrieb. Tento komponent má rôzne nastaviteľné vlastnosti, ale v tomto prípade najdôležitejšími sú vlastnosti farba *color* a hodnota *value*. Farba zabezpečí dostatočnú viditeľnosť textu uloženého ako hodnota komponentu text.

```
<a-entity
   position="0 2 -7"
   text="color: white; value: THIS IS CONSOLE; width: 6; align: left;"
   console>
</a-entity>
```

Obr. 22 Implementácia konzoly využitím elementu a-entity

Tretím komponentom je konzola, ktorá zabezpečuje získavanie potrebných atribútov a ich výpis na objekt v scéne. Pomocou tohto komponentu sa výrazne uľahčí ladenie a vývoj systému pre kolaboráciu. Spôsob registrácie nového A-Frame komponentu s názvom console môžeme vidieť na Obr. 23. V rámci definície tohto komponentu je možné definovať metódy, ktoré spracovávajú životný cyklus. V komponente konzoly sa nachádzajú hlavne základné metódy *init()*, ktorá sa zavolá raz pri renderovaní scény a *update(),* ktorá aktualizuje zmeny entít. Jednou z týchto zmien je aj nastavenie požadovanej hodnoty komponentu text. Táto hodnota môže byť v závislosti od potreby statická vo forme jednoduchého textového reťazca, dynamická vo forme premennej obsahujúcej atribút entity, alebo ich kombináciou pre lepšiu orientáciu a porozumenie výpisov konzoly.

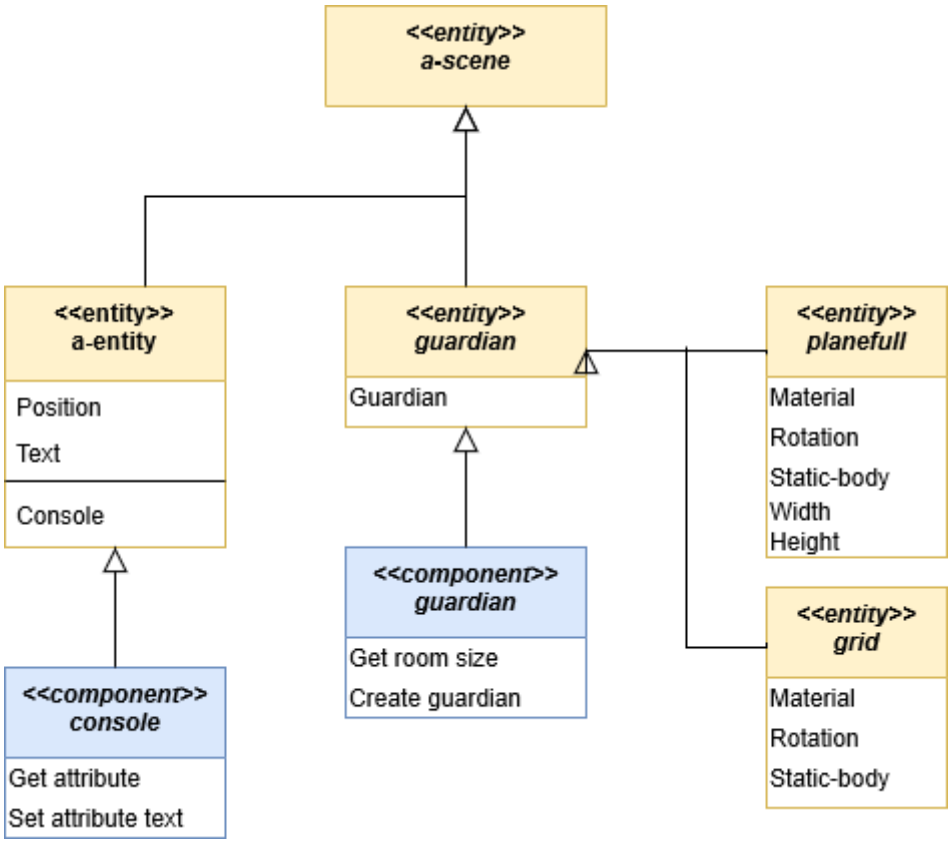
```
AFRAME.registerComponent("console", {
  init: function() {
    el = this.el;
  },
  update: function() {
    el.setAttribute("text", "value", "...");
  }
});
```

Obr. 23 Zjednodušená implementácia komponentu konzoly

Spojením týchto možností dokážeme vytvoriť v scéne konzolu, na ktorú sa budú počas celého behu aplikácie vypisovať požadované hodnoty a správy. S touto vytvorenou konzolou je ladenie vyvíjanej aplikácie oveľa jednoduchšie a navyše zabezpečí lepšiu funkcionalitu systému s menším počtom chybných procesov.

4.2. Testovacie prostredie pre komponent strážcu

Podobne ako spoločnosť Oculus [26] ohraničuje svoje herné prostredie pomocou vizuálnych hraníc, aj táto testovacia scéna bola inšpirovaná systémom strážcu. Oculus umožňuje používateľom vytvoriť si vlastnú hranicu ľubovoľných tvarov a rozmerov alebo stacionárnu hranicu v tvare kruhu, ktorá je počas simulácie virtuálnej reality neviditeľná. Objaví sa len v prípade ak sa používateľ dostane do jej tesnej blízkosti, aby ho to upozornilo na prípadné nebezpečenstvo. Okrem toho je možné naprogramovať dodatočnú funkcionalitu v prípade ak používateľ prekročí túto hranicu, napríklad pozastavenie scény alebo odhlásenie používateľa z virtuálneho prostredia. Aj keď je tento systém strážcu od spoločnosti Oculus veľmi dobre navrhnutý, pre testovacie prostredie bola vytvorená jeho podobná zjednodušená verzia.



Obr. 24 Diagram hierarchie html entít pre strážcu

Ak zoberieme v úvahu len tradičný dizajn miestností pozostávajúcich zo štyroch na seba kolmých stien, môžeme jednoducho navrhnúť strážcu s rovnakým dizajnom ako vidíme na Obr. 24 vyššie. Základnom je pri každej testovacej scéne html entita *<a-scene>*, ktorá je rodičom všetkých ďalších entít. Pre účely ladenia tohto prostredia je zahrnutá aj konzola spomínaná v predošlej kapitole 4.1. Ďalším objektom tejto scény je jednoduchá html entita s komponentom *guardian* a potomkami, ktorý predstavujú dvojrozmerné plochy *<a-plane>* pre podlahu **planefull** a steny strážcu **grid**. Hlavným rozdielom medzi rovinou podlahy a rovinami stien je komponent veľkosti plochy, ktorý

steny nemajú nastavený. Dôvodom je potreba existencie podlahy určitej veľkosti aj v prípade ak používateľ nezadá na vstupe dĺžku a šírku miestnosti. Okrem toho má každá plocha komponent určujúci jej rotáciu a materiál v scéne. Pri vytváraní stien strážcu bolo obzvlášť dôležité zohľadniť materiál, ktorý sa na nich nanáša, konkrétne priehľadnosť a farbu materiálu. Steny ohraničujúce scénu by mali byť dostatočne viditeľné, ale zároveň by nemali odpútať pozornosť používateľa od ostatných prvkov scény. Preto bola zvolená stena reprezentovaná mriežkou, cez ktorú je možné vidieť a súčasne je relatívne dobre rozpoznateľná. Je vytvorená pomocou obrázka mriežky vo formáte png, ktorý umožní v scéne ignorovať priehľadné pixely obrázka. Zdroj obrázka mriežky je v kóde poznačený jednoznačným identifikátorom a zapísaný pomocou elementu *<img>,* ktorý musí byť vnorený v elemente *<a-assets>*. Následne sa objektu roviny nastaví komponent materiál s atribútom zdroj (Obr. 25), ktorého hodnota bude jednoznačný identifikátor obrázka. Okrem toho je potrebné nastaviť aj atribút transparentnosti na hodnotu *false* a atribút prahu pre transparentnosť na hodnotu *0,5*. To zabezpečí zachovanie priehľadných pixelov zdrojového png obrázka.

```
<a-entity guardian>
   <a-plane
      material="src: #grid; transparent: false; alphaTest: 0.5;"
      rotation="0 180 0"
      static-body>
   </a-plane>
...
</a-entity>
```

Obr. 25 Implementácia plochy strážcu so špecifickým materiálom

Napriek tomu, že je strážca pozorovateľný z vnútornej strany, toto riešenie zamedzuje jeho viditeľnosti, ak sa používateľ nachádza z vonkajšej strany ohraničeného priestoru. Príčinou je vložený png obrázok, ktorý je len dvojrozmerný a teda viditeľný len z prednej strany. Takže ak sa používateľ vyskytne mimo ohraničeného priestoru, neuvidí jeho hranice, ale uvidí zvyšok scény a tým pádom sa bude môcť vrátiť naspäť.

```
var height = prompt("Choose a height of scene", height);
var width = prompt("Choose a width of scene", width);
```

Obr. 26 Implementácia získavania rozmerov scény od používateľa

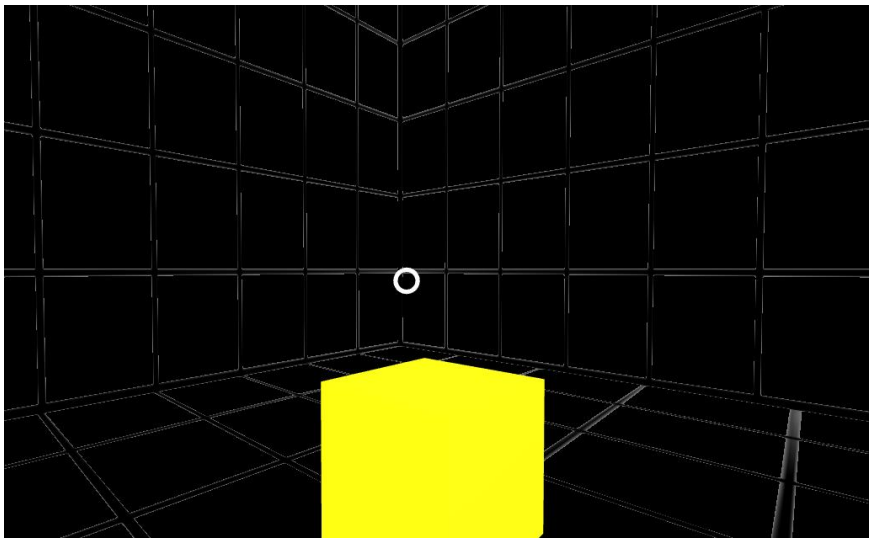
Rovnako podstatné z hľadiska veľkosti virtuálneho priestoru je poskytnutie možnosti používateľovi na výber, respektíve zadanie rozmerov testovaného strážcu. Aby užívateľ zadal veľkosť hraníc pred vygenerovaním scény, bol využitý jazyk Javascript, konkrétne metóda *prompt*, ktorá zobrazí okno s výzvou pred načítaním stránky (Obr. 26). To umožní zadať na vstupe šírku a dĺžku scény, ktorých hodnota sa po potvrdení používateľom uloží do premenných *height* a *width,* a využije na vytvorenie strážcu v jeho okolí. Pre správne generovanie hraníc by sa používateľ mal postaviť do stredu miestnosti a zadať rozmery reálneho priestoru, v ktorom sa môže pohybovať. Umiestenie používateľa je dôležité kvôli návrhu strážcu, ktorý počíta s umiestnením avatara v strede scény, a tým pádom prepočíta súradnice, na ktoré má byť umiestnená hranica podľa vstupnej dĺžky a šírky. Prepočet súradníc a nastavenie pozície pre všetky štyri hranice je zobrazená na Obr. 27 nižšie. Rovnako môžeme vidieť na obrázku, že v prípade ak používateľ nezadal veľkosť miestnosti na vstupe, steny strážcu sa odstránia zo scény.

```
if ((height != null) & (width != null)) {
      back.setAttribute("width", width);
      back.setAttribute("height", 4);
...
      back.setAttribute("position", { y: 2, z: -(height / 2) });
      front.setAttribute("position", { y: 2, z: height / 2 });
      right.setAttribute("position", { y: 2, x: -(width / 2) });
      left.setAttribute("position", { y: 2, x: width / 2 });
} else {
      back.parentNode.removeChild(back);
...
}
```

Obr. 27 Implementácia nastavenia pozície hraníc strážcu

Súčasťou strážcu môže byť aj zavedenie fyziky do scény, na kontrolu kolízie avatara s hranicou. Príkladom využitia kontroly kolízie je zastavenie avatara používateľa pred opustením scény, aj keď sa používateľ v reálnom svete presunul za ohraničený priestor. Na realizáciu takého to návrhu je potrebné integrovať fyziku do systému pomocou A-Frame komponentu *physics* priradeného k elementu *<a-scene>.* Za predpokladu, že vytvorené scéna ešte neobsahuje objekt predstavujúci zem alebo podlahu, je nevyhnutné pridať ho, aby sa predišlo nekonečnému padaniu avatara používateľa, ktorý bude mať priradený komponent kinematického tela *kinematic-body*. Objektu predstavujúcemu zem alebo a všetkým štyrom plochám hraničnej mriežky sa pridá komponent

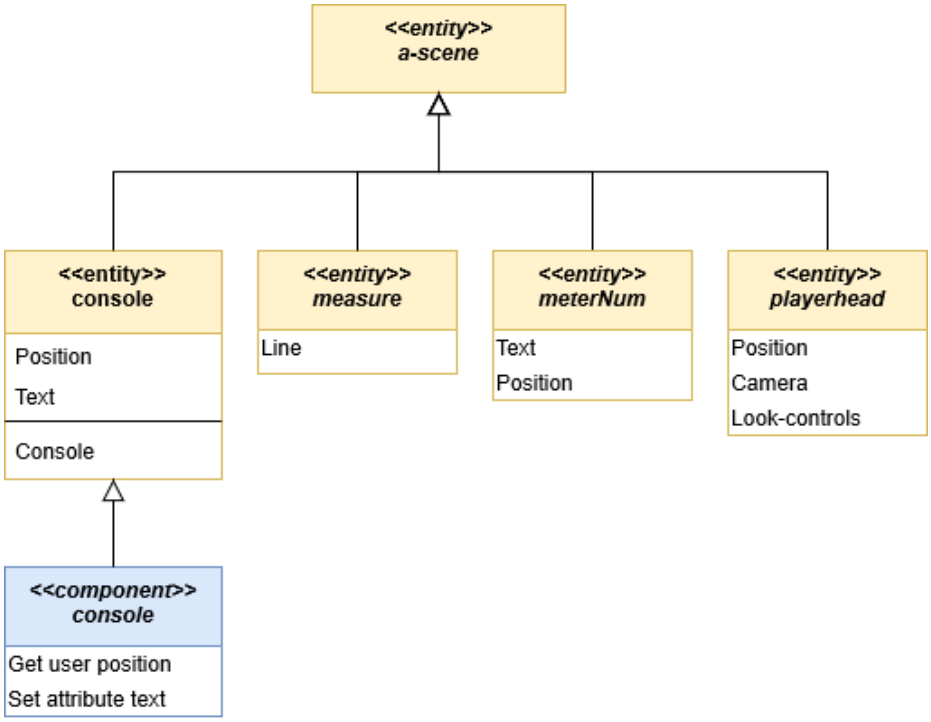
statického tela *static-body*. To zabráni avatarovi hýbať sa za tieto hranice a spadnúť mimo scény, zatiaľ čo reálny používateľ môže opustiť virtuálne ohraničený priestor.



Obr. 28 Vizuálna reprezentácia scény so systémom strážcu

4.3. Testovacie prostredie pre kontrolu vzdialenosti

Tak, ako je uvedené v experimentoch z predchádzajúcej kapitoly 2.1.2, je vhodné v aplikácii otestovať virtuálnu vzdialenosť v porovnaní s reálnou. Pre vylepšenie používateľskej skúsenosti zo systému je vhodné otestovať pohyb v scéne v závislosti na pohybe v reálnom svete, aby sa eliminovali prípadné odlišnosti. Prostredie, ktoré by toto dokázalo otestovať musí byť navrhnuté čo najjednoduchšie a najpresnejšie, napríklad ako to môžeme vidieť na Obr. 29.



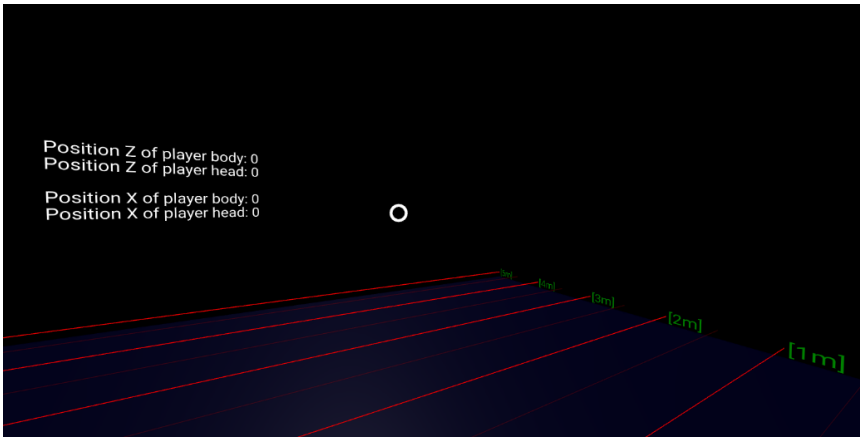
Obr. 29 Diagram hierarchie entít scény pre testovanie vzdialenosti

Základom tejto scény sú štyri objekty: konzola **console**, mierka **measure**, číslovanie mierky **meterNum** a používateľ **playerhead**. Konzola je používaná na získavanie používateľovej polohy v scéne a výpis týchto hodnôt do scény a je umiestnená oproti používateľovi vo vzdialenosti 5 jednotiek. Druhým objektom je jednoduchá entita reprezentujúca čiary mierky vytvorené pomocou viacerých komponentov *line* (Obr. 30), ktoré vykreslia čiaru so zadanou začiatočnou *start* a koncovou *end* pozíciou. Každá čiara, ktorá označovala nárast o celú jednotku bola sýtej červenej farby a každá čiara, ktorá označovala polovicu jednotky mala zníženú nepriehľadnosť. Čiary označujúce nárast o polovicu boli len pomocné pre meranie pohybu, keďže pre používateľa je prirodzenejšie spraviť viacero menších krokov ako jeden dlhý. Prvá čiara začína na počiatočnej pozícii používateľa a posledná na pozícii konzoly.

```
<a-entity
  line__1= "start: 5, 0.1, 0; end: -5 0.1 0; color: red"
  line__1.5= "start: 5, 0.1, -0.5; end: -5 0.1 -0.5; color: red; opacity: 0.2"
  ....>
</a-entity>
```

Obr. 30 Implementácia mierky pre testovanie vzdialenosti

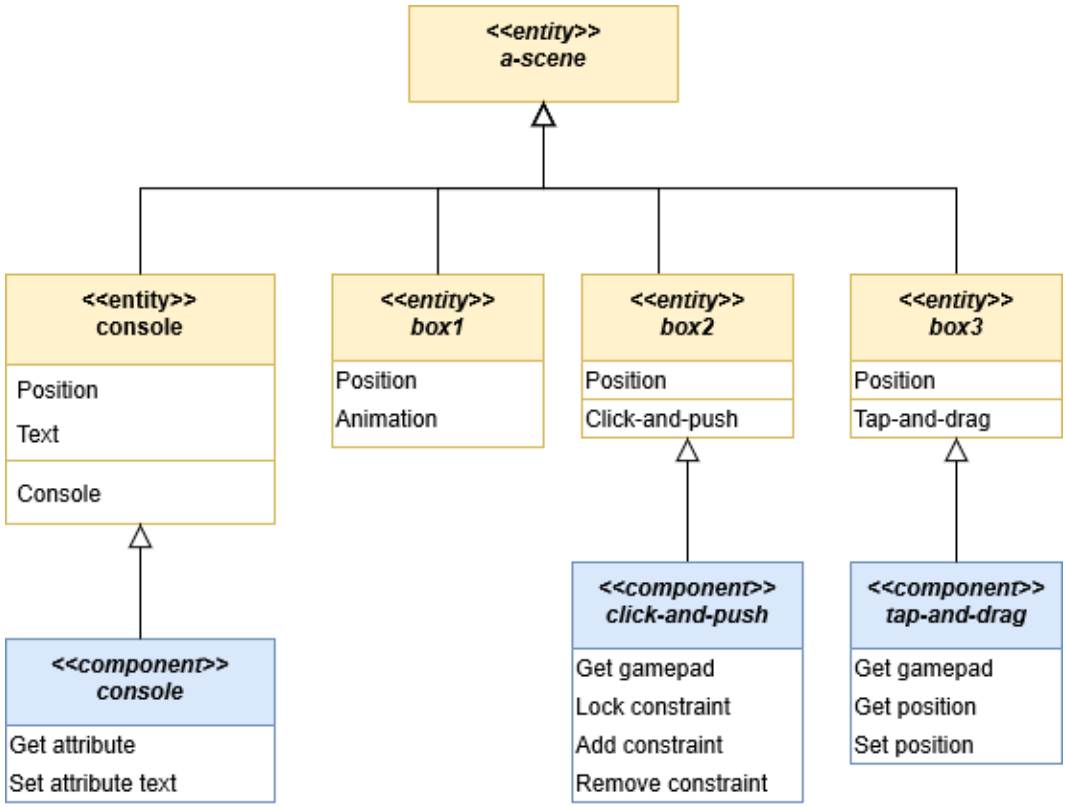
Ďalším objektom v scéne sú jednoduché entity s komponentami text a pozícia. Slúžia na označenie čiar mierky pre lepšiu orientáciu používateľa pri testovaní. Označené sú len čiary s nárastom o celú jednotku a to výraznou zelenou farbou. Posledným dôležitým objektom je používateľ, s komponentom pozícia pre jeho zasadenie do scény. Okrem toho má komponent *camera,* ktorý definuje z ktorej perspektívy sa používateľ pozerá a komponent *look-controls,* ktorý umožňuje vstupnému zariadeniu manipulovať s pozíciou a rotáciou kamery. Viac o týchto komponentoch sa nachádza v kapitole 3.2. Takto navrhnutá a implementovaná scéna (Obr. 31) umožní testovanie vzdialenosti v scéne a reálnej vzdialenosti pri pohybe používateľa. V ideálnom prípade sa táto vzdialenosť mení rovnako vo virtuálnom aj v reálnom svete. V opačnom prípade sa údaje z testovania využijú na doladenie pohybu a zníženie rozdielov vo vzdialenosti.



Obr. 31 Vizuálna reprezentácia prostredia pre testovanie pohybu

4.4. Testovacie prostredie pre kontrolu interakcie

Najdôležitejšou súčasťou akéhokoľvek virtuálneho systému je možnosť interakcie so scénou. To platí aj pre kolaboráciu v zmiešanej realite, a preto je potrebné navrhnúť testovacie prostredie pre ladenie interakcie používateľa so scénou. Keďže pre vyvíjanú aplikáciu boli navrhnuté tri spôsoby interakcie, konkrétne pohľadom, ovládačom a gestami rúk, navrhnutá scéna bude obsahovať prvky pre testovanie všetkých troch spôsobov.



Obr. 32 Diagram hierarchie scény pre testovanie interakcie

Súčasťou tejto scény, ako môžeme vidieť na Obr. 32 sú štyri hlavné html entity, objekty. Okrem týchto objektov by mala byť súčasťou scény aj plocha predstavujúca podlahu. Prvým objektom je ako aj pri ostatných scénach konzola slúžiaca na výpis rôznych hodnôt, v tomto prípade hlavne na výpis údajov získaných z ovládačov. Tieto údaje zahŕňajú stav pripojenia a odpojenia, prítomnosť tlačidiel a stav ich stlačenia, pozícia a orientácia v trojrozmernom priestore. Pre overenie funkčnosti metódy raycasting a komponentu kurzor (spomínaných v kapitole 3.3), a do scény umiestni prvý objekt kocky s nastavenou animáciou rotácie pri priamom pohľade na tento objekt. Použitý komponent *animation* (Obr. 33) umožňuje animovať objekty a meniť hodnoty ich komponentov a hodnoty vlastností komponentov. V tomto prípade sa postupne mení vlastnosť komponentu *rotation* na cieľovú hodnotu určenú vlastnosťou *to.* Na spustenie animácie je využitá vlastnosť *startEvents,* na jej pozastavenie *pauseEvents* a na jej pokračovanie *resumeEvents.* Tieto vlastnosti majú priradenú udalosť, pri ktorej sa spustia, konkrétne to je pohľad na daný objekt

kocky. Okrem toho je animácia nastavená na nekonečné opakovanie pomocou vlastnosti *loop*. Kocku bude možné obísť zo všetkých strán, aby bolo možné otestovať presnosť a funkčnosť interakcie pohľadom z rôznych uhlov a vzdialeností.

```
animation__rotate = "property: rotation;
                     startEvents: mouseenter;
                     pauseEvents: mouseleave;
                     resumeEvents: mouseenter;
                     to: 0 360 0;
                     loop: true; "
```

Obr. 33 Implementácia animácie rotácie objektu

Pre overenie funkčnosti ovládača HoloLens Clicker sa v scéne nachádza ďalší objekt kocky, ktorý reaguje na stlačenie tlačidla ovládača zmenou svojej polohy. Tomuto objektu je priradený komponent *click-and-push*, ktorý zabezpečuje interakciu pri priamom pozeraní sa používateľa na daný objekt, vytváraním, pridávaním a odoberaním fyzických obmedzení na objekte. Ako prvé je potrebné získať vstupné údaje z ovládača. Pomocou udalosti *gamepadconnected,* uvedenej v dokumentácii Gamepad API, zachytíme stav pripojenia ovládača. Ak táto udalosť zachytí pripojenie ovládača je možné využiť metódu *navigator.getGamepads()*, pomocou ktorej získame zoznam všetkých pripojených ovládačov. Z tohto zoznamu je možné zistiť jedinečný index pre každý pripojený ovládač, teda pre HoloLens Clicker je to index 6. Tento index je dôležitý pre volanie konkrétneho ovládača v kóde. Po zistení indexu ovládača a jeho priradeniu k premennej *clicker* (Obr. 34), bolo možné overiť prítomnosť tlačidla na ovládači pomocou Gamepad API metódy *gamepad.buttons*, ktorá vráti pole objektov predstavujúcich tlačidlá ovládača. Stav stlačenia je získavaný využitím metódy *gamepad.buttons[0].pressed*, ktorá vráti hodnotu *true* v prípade ak je tlačidlo na pozícii 0 stlačené. Overením týchto údajov výpisom na konzolu bolo možné implementovať interakciu používateľa s objektom po kliknutí na tlačidlo ovládača. Táto interakcia je konkrétnejšie opísaná v predchádzajúcej kapitole 3.3.

```
window.addEventListener("gamepadconnected", function() {
      var gamepads = navigator.getGamepads
        ? navigator.getGamepads()
        : navigator.webkitGetGamepads
        ? navigator.webkitGetGamepads
        : [];
      clicker = gamepads[6];
    });
```

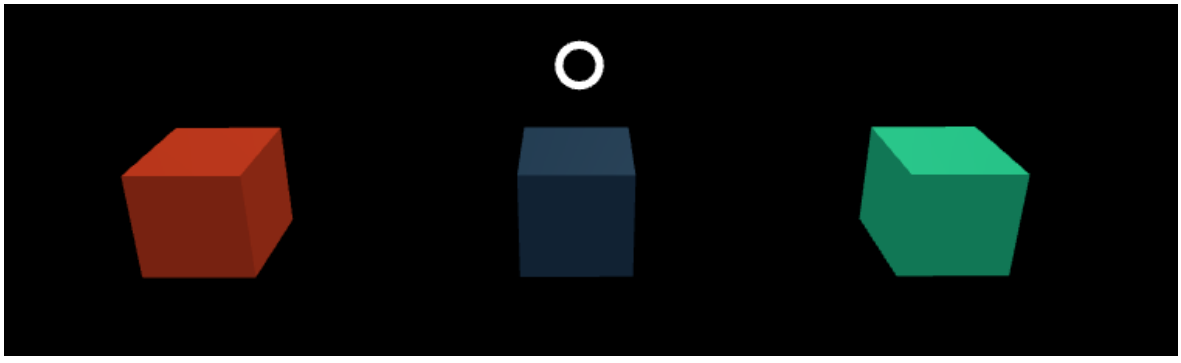
Obr. 34 Implementácia získania vstupov ovládača HoloLens Clicker

Štvrtou entitou pre používateľov zariadenia HoloLens je objekt kocky, určený na testovanie zmeny polohy objektu s využitím gest rúk. Pri implementácii získavania zoznamu všetkých pripojených ovládačov bolo zistené, že ruky sa správajú pre zariadenie ako ovládače, a preto je z nich možné získať stav pripojenia a stlačenia tlačidla (Air tap). Implementácia bude teda vyzerať rovnako ako implementácia ovládača Clicker s rozdielom jedinečného indexu, ktorý je pre ruky číslo 4. Ako ďalšie bolo potrebné overiť schopnosť získavať pozíciu a orientáciu rúk ako ovládačov a následne otestovať rozsah v akom sa tieto hodnoty pohybujú. Znova sa na výpis týchto informácii využila konzola určená na debuggovanie. V dokumentácii Gamepad API sa uvádza, že pre získanie pozície sa využíva experimentálna metóda *gamepad.pose.position*, ktorá vráti pozíciu ovládača ako 3D vektor (Obr. 35). Pre získanie orientácie ovládača je to experimentálna metóda *gamepad.pose.orientation*, ktorá, ak je to možné, vráti orientáciu ovládača ako hodnotu kvarternión.

```
var hand = gamepads[4];
var gpPosX = hand.pose.position[0];
var gpPosY = hand.pose.position[1];
var gpPosZ = hand.pose.position[2];
```

Obr. 35 Implementácia získavania pozície rúk

Po zistení možnosti získavania pozície rúk sa vytvoril komponent *tap-and-drag*, v ktorom je implementovaná funkcionalita hýbania objektom pomocou gest rúk. Princípom je zachytenie udalosti pripojenia ruky ako ovládača, čakanie na vykonanie gesta Air Tap, ktoré sa správa ako stlačenie tlačidla, získanie počiatočnej polohy ruky a získanie polohy ruky pri uvoľnení gesta. Na základe rozdielov medzi počiatočnou a koncovou pozíciou ruky sa prepočítajú nové súradnice objektu kocky a nastavia sa mu pomocou už spomínanej metódy *setAttribute()*. Výsledkom návrhu a implementácie týchto troch entít s príslušnou funkcionalitou je scéna (Obr. 36), ktorá je vhodná na otestovanie rôznych spôsobov interakcie používateľa s objektami a na presnosť a odozvu tejto interakcie.



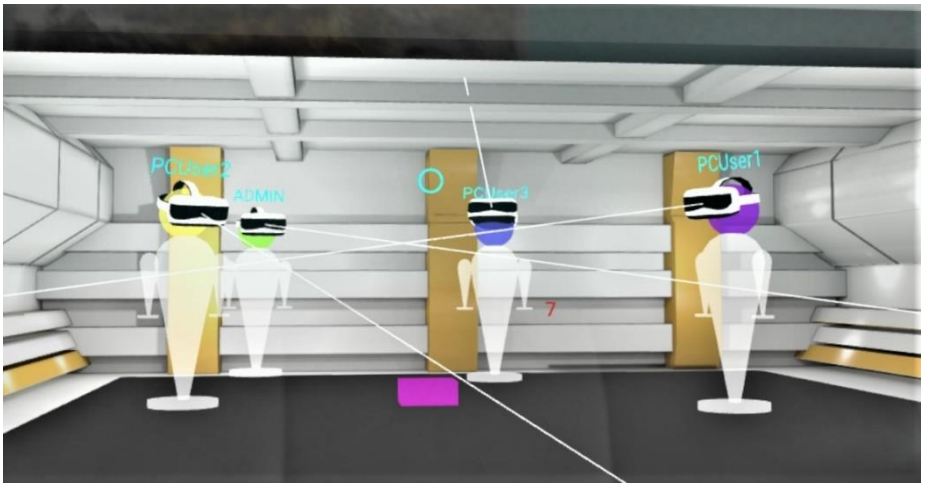
Obr. 36 Vizuálna reprezentácia prostredia pre testovanie interakcie

5. Vyhodnotenie prostredia G-CVE

Experimentálne testovanie a vyhodnotenie vstupno-výstupnej časti vo virtuálnom prostredí G-CVE bolo realizované so zariadením MS HoloLens prvej generácie. Pri testovaní sa spolupracovalo s lokálnymi ale aj vzdialenými klientami, ktorí mali poznatky o virtuálnej realite a zariadení MS HoloLens na rôznej úrovni. V prípade potreby bola klientom poskytnutá inštruktáž pre správne vykonanie úloh testovania. V rámci experimentov vykonávaných pre vyhodnotenie prostredia sme sa hlavne sústredili na schopnosť zariadenia spracovávať vstupné a výstupne údaje, konkrétne prijímanie údajov zo servera a ich vykresľovanie zariadením. Dôraz sa teda dával na snímkovú frekvenciu v rôznych situáciách a prípadné systémové oneskorenie pri vykonávaní úloh. Okrem toho sa testovanie sústredilo aj na jednotlivé navrhnuté komponenty, pri ktorých sme sa sústredili hlavne na ich využiteľnosť v prostredí G-CVE. Objektívne sa vyhodnotili výhody a nevýhody komponentov na základe rôznych situácii a vykonávaných úloh.

5.1. Vyhodnotenie kolaborácie v prostredí G-CVE

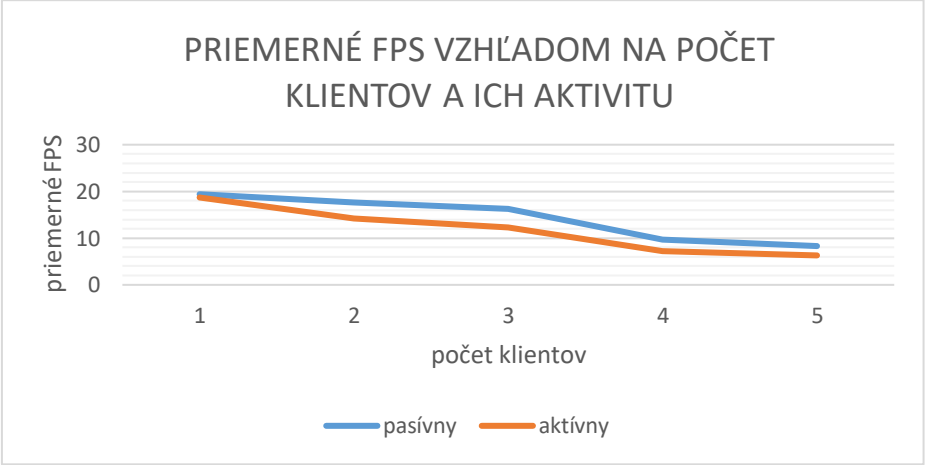
Prvou súčasťou vyhodnotenia kolaboratívneho prostredia G-CVE bolo niekoľko pokusov odmerať priemerný počet snímok za sekundu počas určitej doby a za určitých podmienok. Dôležitým faktorom tohto experimentu bol počet hráčov a vplyv tohto počtu na plynulosť scény. Okrem toho bola dôležitá aj aktivita používateľov, keďže klient, ktorý sa aktívne hýbe po virtuálnom prostredí je väčšou záťažou pre systém ako pasívny klient stojaci na mieste. Keďže nebolo možné zohnať viacero používateľov so zariadením Microsoft HoloLens, pre účely testovania sa klienti pripájali do prostredia G-CVE pomocou svojich počítačov a mobilných zariadení z rôznych vzdialených miest (Obr. 37).



Obr. 37 Klienti pripojení v prostredí G-CVE

Experiment pozostával z postupného prihlasovania sa piatich používateľov, ktorí mali na povel vykonávať dve úlohy po dobu desiatich sekúnd. Prvou úlohou bolo nehybné státie v zornom poli používateľa so zariadením Microsoft HoloLens. Druhou bolo aktívne hýbanie sa, taktiež v zornom

poli klienta s HoloLens-om. Počas desiatich sekúnd, pri piatich pokusoch sa pri každej úlohe zaznamenával priemerný počet snímok za sekundu. Rozdielom pri týchto pokusoch bol počet pripojených používateľov. Výsledky experimentu sa zaznamenali do tabuľky, z ktorej vznikol graf (Obr. 38). V spodnej časti grafu sa nachádza počet prihlásených používateľov v scéne a v jeho ľavej časti priemerný počet snímok za sekundu, nameraný počas desiatich sekúnd. V grafe sú zaznamenané dva aktivity testovacích subjektov. Modrou farbou je zaznačené pasívne státie a oranžovou aktívne hýbanie sa po scéne. Ako môžeme vidieť v grafe, priemerné FPS klesá s pribúdajúcim počtom prihlásených klientov. Rozdielom medzi aktívnym a pasívnym správaním klientov je len v krivke priemerného FPS, ktorá pri aktívnom hýbaní sa používateľov klesá rýchlejšie. Najvyšší nameraný počet snímok za sekundu bol pri jednom prihlásenom pasívnom používateľovi, ktorým bol samotný klient s HoloLens-om. Najnižší priemer bol nameraný pri aktívnom pohybe piatich používateľov. Najvýraznejší pokles FPS nastal po pripojení štvrtého klienta, nezáležiac na aktivite. Celkovo sa snímková frekvencia znižuje s narastajúcim počtom klientov, čo spôsobuje nepríjemné a neplynulé zobrazovanie scény.



Obr. 38 Priemerné FPS pri aktivite rôzneho počtu používateľov

Pri realizovaní prvého experimentu so vzdialenými používateľmi sa prišlo na problém s oneskorením systému pre zariadenie Microsoft HoloLens. Pri testovaní sa ukázalo, že po zadaní úlohy a hlasovom potvrdení používateľov o jej vykonaní cez iné médium, vznikol časový posun systému spôsobujúci neskorší pohyb avatarov pri vykonávaní úlohy. Namerané oneskorenie bolo v rozmedzí od 5 až po 25 sekúnd, pričom narastalo s časom stráveným v scéne. Pre odstránenie tohto časového posunu bolo potrebné odhlásenie sa zo scény, obnovenie webovej stránky a opätovné prihlásenie sa do prostredia G-CVE. Po tomto zistení sa overovalo aj opačné oneskorenie, sledovaním pohybu reálneho používateľa a jeho avatara so zariadením HoloLens. Avšak v tomto prípade sa výrazné oneskorenie neprejavilo a avatar sa pre ostatných klientov hýbal v zhruba rovnakom čase ako jeho používateľ. Okrem toho sa časový posun testoval aj s používateľmi pripojenými cez rovnakú lokálnu sieť. Podobne ako pri vzdialených klientoch, aj v tomto prípade

používateľ s HoloLensom vnímal oneskorené pohyby avatarov, pričom nameraný ping z danej siete na server mal priemernú hodnotu 120ms.

Ako posledné sa testovalo prostredie G-CVE z hľadiska používateľského zážitku so zariadením Microsoft HoloLens. Vzhľadom na situáciu bolo možné zahrnúť do tohto experimentu len dvoch ľudí. Obaja zúčastnení boli vo veku 50 rokov a nemali predchádzajúce skúsenosti s virtuálnou a ani zmiešanou realitou. Keďže ich poznatky o danej téme boli minimálne, pred testovaním boli v krátkosti oboznámení o dôležitých faktoroch v oblasti virtuálnej reality. Rovnako im bola poskytnutá inštruktáž na používanie zariadenia Microsoft HoloLens, ktorá zahŕňala informácie o zmiešanej realite a možnostiach využitia daného zariadenia v porovnaní s inými zariadeniami virtuálnej reality. Po takomto oboznámení zúčastnených o základných informáciách boli pripravení na splnenie troch jednoduchých úloh: prihlásenie sa do systému G-CVE, pohyb po virtuálnej scéne a interakcia s objektami. Vzhľadom na ich skúsenosti im bol poskytnutý dostatočný čas na vykonanie týchto úloh, ktorý sa do výsledku z testovania nezarátal. Po ukončení úloh im boli položené jednoduché otázky týkajúce sa celkového dojmu z G-CVE systému ale aj jednotlivých častí, ako napríklad náročnosť interakcie s objektami. Vyhodnotením získaných odpovedí sme zistili, že celkový dojem zo scény G-CVE bol v podstate pozitívny, avšak oneskorenie pohybu avatarov ostatných používateľov nevplývalo na testovacie subjekty kladne. Okrem toho, používatelia nemali výrazný problém pohybovať a orientovať sa v scéne, zatiaľ čo dávali pozor aj na svoje fyzické okolie. Problém mali s veľkosťou miestnosti, ktorá nebola dostatočne veľká na testovanie tejto scény, a teda nemali možnosť neobmedzeného pohybu. Čo sa týka interakcie s objektami, najväčší problém mali s ovládaním komponentu s využitím fyziky a ovládača HoloLens Clicker. Ovládanie druhého komponentu im prišlo viac prirodzené a ľahšie na naučenie sa. Myšlienka kolaborácie vzdialených klientov v prostredí virtuálnej reality sa subjektom zdala perspektívna.

5.2. Vyhodnotenie jednotlivých komponentov

Súčasťou návrhu práce bolo aj vytvorenie testovacích prostredí za účelom vykonania experimentov potrebných na vyhodnotenie práce. Vytvorili sa tri prostredia, ktoré mali vyhodnotiť systém strážcu, mierku virtuálnych objektov v porovnaní s reálnym svetom testovanú pohybom používateľa a prostredie na vyhodnotenie interakcie so systémom. V tejto kapitole sa priblížia detaily experimentov a podmienky za ktorých boli vykonávané. Na základe výsledkov experimentovania sa uvedú grafy, z ktorých sa odvodí objektívne vyhodnotenie komponentov, ktoré zahrnie ich klady ale aj zápory. Táto kapitola sa nebude venovať testovaniu systému G-CVE ako celku, ale testovaniu jeho jednotlivých častí a komponentov.

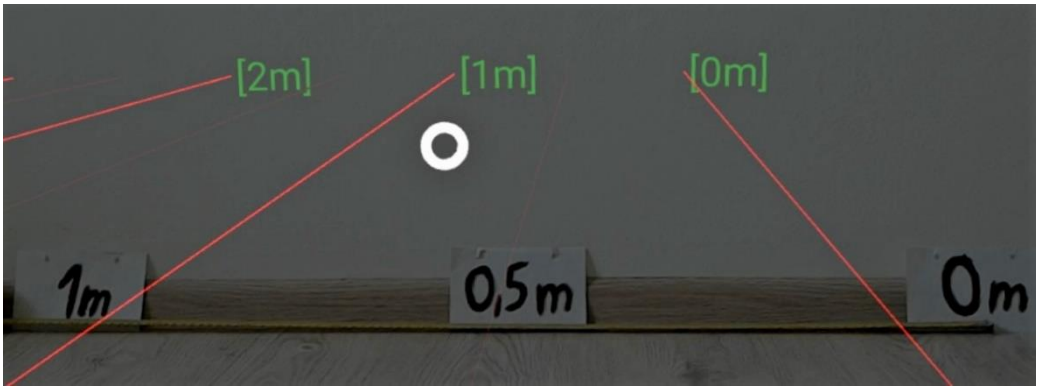
5.2.1. Vyhodnotenie strážcu

V rámci vyhodnotenia testovacej scény s komponentom strážcu bolo potrebné otestovať viacero faktorov. Prvým z nich bolo osvetlenie miestnosti. Ďalšími faktormi bola farba mriežky strážcu a jeho umiestnenie po načítaní scény. V praxi sa vyskúšali všetky kombinácie faktorov, teda svetlá izba s výraznou alebo s tmavou mriežkou a tmavá izba s výraznou aj s tmavou mriežkou. V oboch izbách s rôznym osvetlením bola výrazná mriežka dobre viditeľná, avšak príliš pútala pozornosť používateľa, ktorý sa nevedel sústrediť na dôležitejšie prvky scény. Toto zistenie vylučuje využitie výraznej mriežky strážcu v scénach, keďže sa používateľ nedokáže sústrediť na správne veci. Ďalšou kombináciou bola tmavá mriežka strážcu a dobre osvetlená miestnosť. Všeobecne tmavšie virtuálne objekty pozorované cez zariadenie Microsoft HoloLens sú menšou záťažou na ľudské oči, a preto bola táto kombinácia najlepšou. Avšak napriek dobrej kombinácii bola mriežka neustále v zornom poli používateľa a teda mohla pôsobiť rozptyľujúco. Poslednou kombináciou bola tmavá mriežka a tmavá, zle osvetlená miestnosť. V tomto prípade bola mriežka strážcu slabo viditeľná a neplnila svoj účel. Navyše samotne zle osvetlená miestnosť zvyšovala riziko narazenia používateľa do reálneho objektu, čomu sme sa snažili predísť. Čo sa týkalo umiestenia strážcu, pri dodržaní podmienok umiestenia používateľa v strede miestnosti a zadaní správnych rozmerov, umiestenie bolo takmer ako očakávané. Celkovo systém strážcu nie je vhodný v tomto stave na používanie v kolaboratívnych prostrediach a preto ani nebol využitý v systéme G-CVE. Okrem toho dôvod jeho vynechania vo vyvíjanom prostredí bol taký, že samotná scéna obsahovala steny, ktoré ju ohraničovali. Navyše používateľ dokázal vnímať aj svoje fyzické okolie ak bola miestnosť dostatočne osvetlená.

5.2.2. Vyhodnotenie vzdialenosti pri pohybe

Jedným z dôležitých prvkov zmiešanej reality je aj odpovedajúca vzdialenosť medzi virtuálnou scénou a reálnym svetom. Je potrebné scénu navrhnúť tak, aby bola zhruba rovnakej veľkosti ako by sa v realite očakávalo, a aby pohyb človeka v reálnom priestore odpovedal aj pohybu jeho avatara vo virtuálnom priestore. Na vykonanie tohto experimentu bolo využité prostredie opísané v kapitole 4.2. Okrem toho bolo potrebné v reálnom prostredí vytvoriť na zemi viditeľnú mierku, ktorá by označovala každý meter a pol meter v realite. Pri testovaní sa vykonalo desať pokusov, pričom v každom pokuse bolo vykonaný pohyb vpred päť krát. Na začiatku každého pokusu sa scéna reštartovala a používateľ sa postavil na vopred určené miesto vedľa mierky v reálnom prostredí. Z odchýlok nameraných pri každom pohybe sa vytvoril a zaznamenal priemer. Na základe vypočítaného priemeru, ktorý sa pohyboval v rozmedzí -0,15 až +0,14, môžeme usúdiť, že jedna jednotka v scéne predstavuje jeden meter v realite. Napriek tomu, že toto meranie nie je úplne presné, je tento predpoklad oprávnený, čo môžeme vidieť aj na Obr. 39 z testovania. Vďaka tomuto

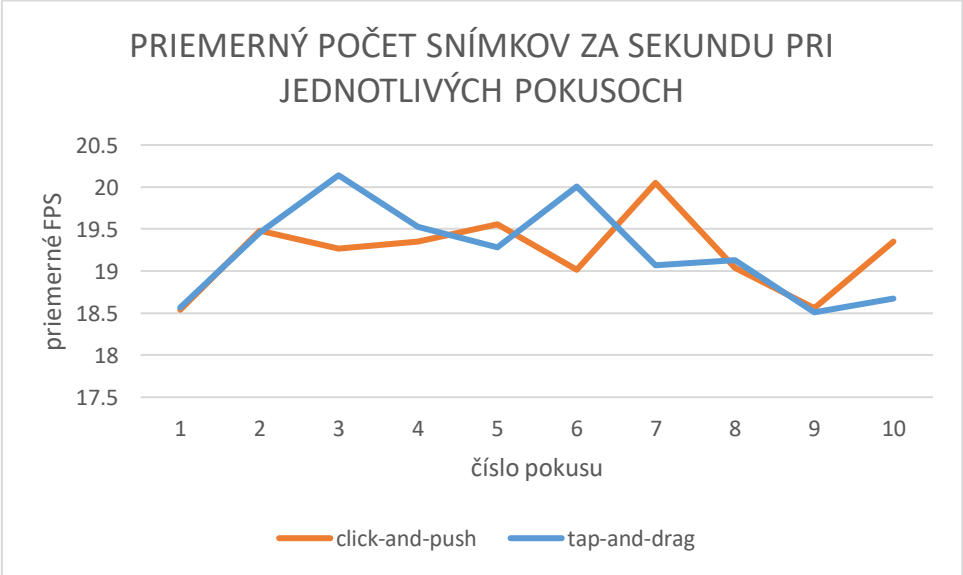
zisteniu nebolo potrebné upravovať vzdialenosti a škálovať finálnu verziu systému G-CVE. Používateľ tak má možnosť vidieť virtuálnu scénu v jej reálnej veľkosti.



Obr. 39 Porovnanie reálnej a virtuálnej mierky

5.2.3. Vyhodnotenie interakcie

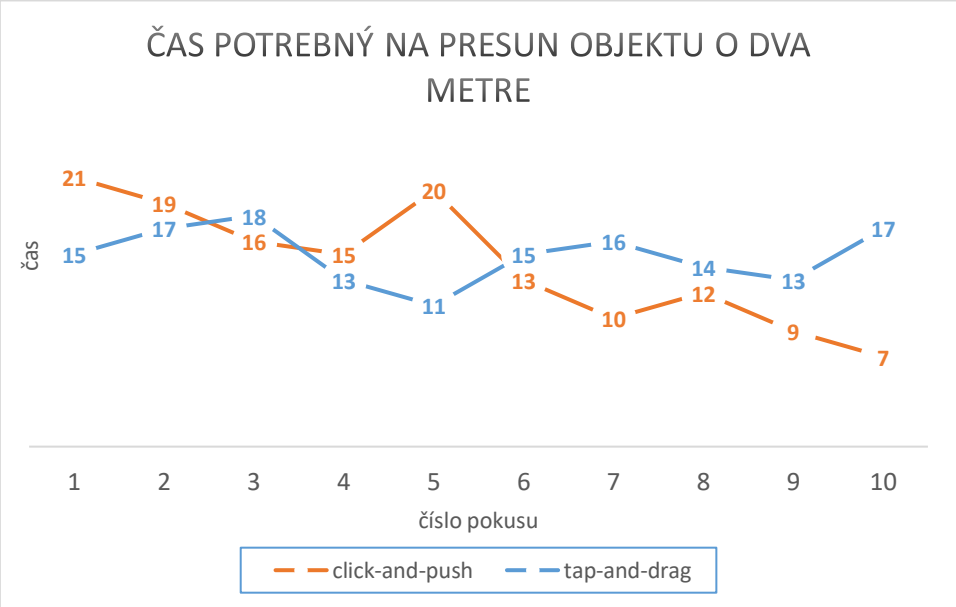
Súčasťou vyhodnotenia dvoch komponentov určených na interakciu, tap-and-drag a click-and-push, bola aj séria experimentov zahrňujúca sledovanie priemerného počtu snímok za sekundu, a sledovanie času potrebného na presunutie objektu do určitej vzdialenosti. Využitá na to bola testovacia scéna z kapitoly 4.4. Táto scéna bola vytvorená s čo najmenším počtom objektov a teda aj celkovým počtom polygónov, aby zbytočne nevznikali nepriaznivé faktory ovplyvňujúce výsledky testovania. Prvým experimentom bolo teda sledovanie snímok za sekundu, ďalej už len FPS. Cieľom bolo porovnať FPS pri používaní oboch komponentov a zistiť, či spôsob ich implementácie ovplyvňuje plynulosť scény. Úlohou tohto experimentu bolo používať komponent po dobu desiatich sekúnd s tým že sa zaznamenával počet snímok za sekundu. Z každého pokusu sa vyrátal priemerný počet FPS a zaznamenal sa do tabuľky. Pokusov bolo desať pre každý z komponentov.



Obr. 40 Priemerné FPS pri testovaní interakcie

Na základe tabuľky s výsledkami sa vytvoril graf, ktorý môžeme vidieť na Obr. 40. Oranžovou farbou je zaznamenaný priemerný počet FPS pri používaní komponentu click-and-push a modrou farbou pri používaní tap-and-drag. V spodnej časti grafu je zaznačené číslo pokusu (1 až 10) a v ľavej časti priemerný počet FPS (17,5 až 20,5). Z grafu môžeme vidieť, že sa v jednoduchej scéne pohybovalo priemerné FPS pri interakcii s objektami v rozmedzí 18 až 20. Tieto hodnoty nie sú alarmujúce, keďže sa takmer nijako nelíšia od priemerného FPS pri obyčajnom pohybe touto scénou. Zo zaznamenaných údajov môžeme usúdiť že implementácia oboch komponentov výrazne nezaťažuje systém.

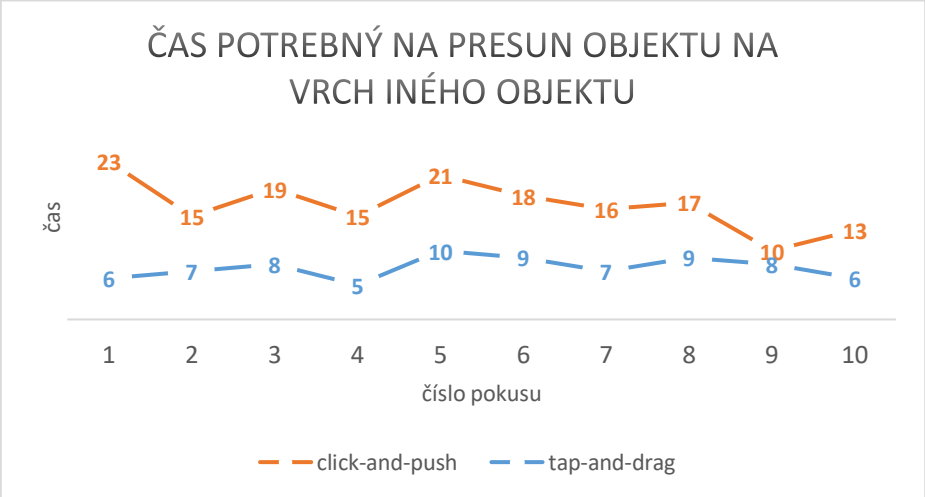
Druhý experiment bol zameraný na porovnanie použiteľnosti týchto komponentov. Opäť sa využila testovacia scéna opísaná v kapitole 4.4. Úlohou tohto experimentu bolo presunutie objektu na určité miesto. Keďže použitá scéna neobsahuje žiaden vhodný referenčný bod, využil sa na to objekt v reálnom svete. Tým bola čiara vyznačená na zemi vo vzdialenosti dva metre od virtuálnych objektov, za ktorú ich bolo potrebné presunúť. Následne sa vykonalo desať pokusov s dvoma objektami, na ktorých boli naviazané testované komponenty. Pri každom pokuse a každom objekte sa zaznamenal čas potrebný na presunutie daného objektu za referenčnú čiaru. Tieto výsledky boli zaznamenané do tabuľky, z ktorej sa následne vytvoril graf zobrazený na Obr. 41. Na tomto grafe sú opäť výsledky z testovania komponentu click-and-push zaznačené oranžovou farbou a výsledky z komponentu tap-and-drag zasa modrou farbou. V spodnej časti sa nachádza označenie čísla pokusu (1 až 10). Vo vrcholoch kriviek grafu sa nachádzajú hodnoty reprezentujúce čas v sekundách potrebný na presun objektu (7 až 21 sekúnd).



Obr. 41 Graf času potrebného na vykonanie horizontálnej interakcie

Z výsledkov tohto experimentu tvaru krivky v grafe vieme povedať, že pokusy s komponentom tap-and-drag trvali zhruba rovnaký čas. Rozdiel medzi najrýchlejším a najpomalším pokusom bol len 7 sekúnd. Priemer z desiatich pokusov bol 14,9 sekúnd. Pokusy s komponentom click-and-push dokázali, že čas na vykonanie tej istej akcie klesá s narastajúcim počtom pokusov, teda sa zlepšuje ak si používateľ zvykne na jeho fungovanie. Rozdiel medzi najrýchlejším a najpomalším časom tohto komponentu bol 14 sekúnd. Priemer z desiatich pokusov bol 14,2 sekúnd. Zo zaznamenaných údajov je možné usúdiť, že komponent tap-and-drag je ľahší na používanie, avšak o trocha pomalší ako click-and-push na vzdialenosť 2 metre.

Tretí experiment bol takmer rovnaký ako druhý. Rozdielom bolo to, že objekt sa nepresúval na vzdialenosť dva metre, ale bolo potrebné ho presunúť na vrch ďalšieho objektu. Rovnako sa meral čas potrebný na zvládnutie tejto úlohy v desiatich pokusoch pre oba komponenty. Objekt, na vrch ktorého mali byť presunuté, bol rovnakej veľkosti a tvaru ako presúvané objekty. Výsledky experimentu sú zaznamenané v grafe (Obr. 42). Opäť je čas potrebný na vykonanie úlohy s komponentom click-and-push označený oranžovou a s komponentom tap-and-drag modrou farbou. V spodnej časti sa nachádza číslo pokusu. Z tohto grafu môžeme vidieť že priemerný čas vykonávania úlohy tretieho experimentu bol v prípade click-and-push 16,7 sekúnd a v prípade tap-and-drag komponentu bol priemerný čas 7,5 sekúnd. Z týchto údajov je jasné, že túto úlohu bolo ťažšie vykonať s komponentom click-and-push, ktorý má zavedenú fyziku. Dôvodom bolo neustále padanie presúvaného objektu z plochy, na ktorú mal byť presunutý. Okrem toho bolo ťažké zdvihnúť tento objekt zo správneho uhla. Naopak využitie komponentu tap-and-drag bolo na túto úlohu ľahšie. Problémom bolo však to, že nemal zavedenú fyziku, a teda sa často prekrýval s objektom na ktorý mal byť presunutý.



Obr. 42 Graf času potrebného na vykonanie vertikálnej interakcie

Celkovým výsledkom tohto experimentálneho testovania interakcie bolo odhalenie plusov a mínusov oboch navrhnutých komponentov, ich možnosti využitia a najlepší spôsob prevedenia interakcie. Pre lepší prehľad sú zistené údaje porovnané v tabuľke Tab. 2. Napriek výhodám a nevýhodám týchto komponentov sú oba využiteľné a vhodné na interakciu používateľa s objektami.

	tap-and-drag	**click-and-push**
Prítomnosť fyziky	neprítomná	prítomná, objekty sú dynamické
Uchytenie objektu	jednoduché	vyžaduje si cvik a presnosť
Posun očakávaným smerom	vždy sa pohne v smere pohybu ruky	pri uchytení objektu zo zlej strany sa môže pohnúť neočakávaným smerom
Vzdialenosť posunu	vhodný na menšie vzdialenosti, posun je pomalý a limitovaný pohybom ruky	vhodný na väčšie vzdialenosti vďaka rýchlosti posunu a možnosti hýbania sa aj s uchyteným objektom
Presnosť posunu	presnejší posun vďaka možnosti posunu o malú vzdialenosť	dynamické objekty je ťažšie presunúť na očakávané miesto
Smer posunu	bezproblémové hýbanie všetkými smermi	problém s hýbaním objektu smerom k používateľovi
Plynulosť posunu	objekt sa objaví na novej pozícii až po vykonaní gesta	objekt sa hýbe počas celej interakcie s používateľovým kurzorom
Intuitívnosť interakcie	podobná prirodzenej interakcii s hologramami	vyžaduje si natrénovanie tejto interakcie

Tab. 2 Porovnanie komponentov interakcie

Záver

Vzdialená kooperácia je pomerne často využívaným spôsobom na spoluprácu v rôznych odvetviach a na rôzne účely. Cieľom našich diplomových prác bolo vytvorenie virtuálneho prostredia s možnosťou podpory rôznych zariadení za účelom umožnenia interaktívnej vzdialenej kolaborácie. Keďže existuje mnoho zariadení na zobrazovanie virtuálnej reality, vznikla idea vytvoriť prostredie podporujúce viacero zariadení a to pomocou vhodnej webovej technológie. Táto idea vznikla z dôvodu perspektívy virtuálnej reality do budúcnosti a potreby v dnešnej dobe spolupracovať s klientami na diaľku.

Naše kolaboratívne virtuálne prostredie sme sa snažili vytvoriť čo najjednoduchšie na implementáciu. Vytvorené komponenty sú dynamické, teda je možné vložiť ich do akejkoľvek scény, v prípade ak sa dodržia požiadavky opísané v systémovej príručke. Naše prostredie G-CVE aktuálne podporuje desktop-y, niektoré inteligentné telefóny a zariadenie Microsoft HoloLens. Experimentálnym testovaním uvedením v tejto diplomovej práci je však preukázané, že podpora a využívanie zariadenia MS HoloLens pre zobrazovanie tohto prostredia nie je ideálne. Toto testovanie zahŕňalo kontrolu počtu zobrazených snímok za sekundu, časové oneskorenie systému ale aj používateľský dojem. Z hľadiska dojmu klientov z prostredia G-CVE boli ohlasy podľa dotazníka viac pozitívne, hlavne čo sa týkalo pohybu a interakcie. Avšak problémom podľa meraní je priemerný počet nameraných FPS, ktorý nepresiahol hodnotu 20. Pri zložitejších scénach s väčším počtom polygónov táto hodnota mierne klesala. Okrem toho počet FPS negatívne menil aj pri narastajúcom počte prihlásených klientov, ktorý boli v zornom poli používateľa s HoloLensom. V tomto prípade hodnota FPS rapídne klesala až na neprijateľné hodnoty, čo spôsobovalo neplynulý beh scény. Tieto hodnoty môžu byť ale obmedzované výkonom daného zariadenia. Čo sa týka komponentov zabezpečujúcich interakciu, pri experimentoch sa ukázalo, že majú svoje silné aj slabé stránky z hľadiska fyziky, precíznosti a vzdialenosti potrebnej na presun objektu. Zatiaľ čo jeden komponent je lepší na precízny posun o malú vzdialenosť bez zahrnutia fyziky, druhý je vhodný na rýchlejší pohyb o veľkú vzdialenosť s využitím fyziky.

V dôsledku je teda podpora pre zariadenie zmiešanej reality Microsoft HoloLens prvej generácie dostačujúca, ale nie ideálna. Avšak porovnaní s podobnou multiplatformovou webovou aplikáciou Mozilla Hubs, ktorá taktiež slúži na kolaboráciu vo virtuálnom prostredí [29], je výsledok tejto diplomovej práce lepší o podporu vstupov zo zariadenia MS HoloLens. Zatiaľ čo Mozilla Hubs poskytuje len prezenčnú formu pre toto zariadenie, klientska časť tejto práce dokáže spracovať údaje o polohe zariadenia, gestikulačné vstupy a vstupy z ovládača HoloLens Clicker, ktoré následne využíva na pohyb po scéne a interakciu používateľa s objektami v tejto scéne.

Zoznam použitej literatúry

[1]. Microsoft dokumentácia [online]. [cit. 2020-1-2]. Dostupné z: https://docs.microsoft.com/en-us/windows/mixed-reality/

[2]. Microsoft HoloLens [online]. [cit. 2020-1-2]. Dostupné z: https://www.microsoft.com/en-us/hololens/hardware

[3]. Evans, Gabriel, a spol. "Evaluating the Microsoft HoloLens through an augmented reality assembly application." *Degraded Environments: Sensing, Processing, and Display 2017*. Vol. 10197. International Society for Optics and Photonics, 2017. Dostupné z: https://lib.dr.iastate.edu/cgi/viewcontent.cgi?article=1178&context=me_conf

[4]. Liu, Yang, a spol. "Technical evaluation of HoloLens for multimedia: a first look." *IEEE MultiMedia* 25.4 (2018): 8-18. Dostupné z: https://filedn.com/lgg0zBdMexjkOsGGt8ihIAf/publication/2018%20IEEE%20Multimedia.pdf

[5]. Zhang, Zhengyou. “Microsoft kinect sensor and its effect,”IEEE Multimedia, vol. 19, no. 2, pp. 4–10, Feb. 2012. Dostupné z: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6190806

[6]. Zhang, Longyu, Haiwei Dong, a Abdulmotaleb El Saddik. "From 3D sensing to printing: A survey." *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 12.2 (2016): 27. Dostupné z: https://www.researchgate.net/profile/Longyu_Zhang2/publication/283664754_From_3D_Sensing_to_Printing_A_Survey/links/569481cb08ae425c68964871/From-3D-Sensing-to-Printing-A-Survey.pdf

[7]. Zhu Zhiwei, Qiang Ji. „Eye and gaze tracking for interactive graphics display.“ Machine Vision and Applications (2004). Dostupné z: https://link.springer.com/content/pdf/10.1007/s00138-004-0139-4.pdf

[8]. Hutchinson TE, White K, Worthy JR, Martin N, Kelly C, LisaR, Frey A „Human-computer interaction using eye-gazeinput.!“ IEEE Transactions on Systems Man and Cybernetics (1989). Dostupné z: https://www.researchgate.net/profile/Preston_White/publication/3113649_Human-computer_interaction_using_eye-gaze_input_IEEE_Trans_Syst_Man_Cybern/links/0c960533d61d05115a000000.pdf

[9]. Funk, Markus, Mareike Kritzler, and Florian Michahelles. "HoloLens is more than air tap: Natural and intuitive interaction with holograms." *Proceedings of the Seventh*

International Conference on the Internet of Things. ACM, 2017. Dostupné na internete: http://www.makufunk.de/wp-content/papercite-data/pdf/funk2017hololens.pdf

[10]. Hou, Weiting. "Augmented Reality Museum Visiting Application based on the Microsoft HoloLens." *Journal of Physics: Conference Series*. Vol. 1237. No. 5. IOP Publishing, 2019. Dostupné na internete: https://iopscience.iop.org/article/10.1088/1742-6596/1237/5/052018/pdf

[11]. Hon-Anderson, Eric. "Real-time voice recognition on a handheld device." U.S. Patent No. 9,009,033. 14 Apr. 2015. Dostupné na internete: https://patentimages.storage.googleapis.com/8a/33/5b/e63f635c3875ed/US9009033.pdf

[12]. Dokumentácia Unity [online]. [cit. 2020-1-2]. Dostupné na internete: https://docs.unity3d.com/560/Documentation/Manual/VRDevices.html

[13]. Dokumentácia WebVR [online]. [cit. 2020-1-2]. Dostupné na internete: https://webvr.info/developers/

[14]. Dokumentácia A-Frame [online]. [cit. 2020-1-2]. Dostupné na internete: https://aframe.io/docs/

[15]. Gamma, Erich; Helm, Richard; Johnson, Ralph; Vlissides, John (1994). Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley. p. 20. ISBN 0-201-63361-2

[16]. Martin, Adam. "Entity Systems are the Future of MMOG Development Part 2". Retrieved 25 December 2013.

[17]. Dokumentácia Javascript [online]. [cit. 2020-1-2]. Dostupné na internete: https://javascript.info/introduction-browser-events

[18]. Dokumentácia Gamepad API [online]. [cit. 2020-1-2]. Dostupné na internete: https://developer.mozilla.org/en-US/docs/Web/API/Gamepad_API

[19]. Lukosch, Stephan, a spol. Collaboration in augmented reality. *Computer Supported Cooperative Work (CSCW)*, 2015, 24.6: 515-525. Dostupné na internete: https://blog.roziqbahtiar.com/wp-content/uploads/2013/09/2002-CACM-CollabAR.pdf

[20]. Billinghurst, Mark and Hirokazu Kato (1999). Collaborative Mixed Reality. In *Proceedings of the First International Symposium on Mixed Reality (ISMR'99). Mixed Reality – Merging Real and Virtual Worlds, Yokohama, Japan, 19–21 March 1999*, Berlin, Germany: Springer Verlag, pp. 261–284. . Dostupné na internete:

http://www.perchristiansson.com/edu/aau/education/reports/1999_billinghurst_mixed_reality.pdf

[21]. Hansen, Margaret M. „Versatile, immersive, creative and dynamic virtual 3-D healthcare learning environments: review of the literature.“ Journal of medical Internet research 10.3 (2008).

[22]. Davis, Matthew Christopher, et al. „Virtual interactive presence in global surgical education: international collaboration through augmented reality.“ *World neurosurgery* 86 (2016): 103-111.

[23]. Martín-Gutiérrez, Jorge, a spol. "Augmented reality to promote collaborative and autonomous learning in higher education." *Computers in human behavior* 51 (2015): 752-761.

[24]. Fairchild, Allen J., et al. A mixed reality telepresence system for collaborative space operation. *IEEE Transactions on Circuits and Systems for Video Technology*, 2016, 27.4: 814-827. Dostupné na internete: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7490357

[25]. Peters, Erwin, et al. Design for collaboration in mixed reality: Technical challenges and solutions. In: *2016 8th International Conference on Games and Virtual Worlds for Serious Applications (VS-GAMES)*. IEEE, 2016. p. 1-7. Dostupné na internete: https://www.researchgate.net/profile/Igor_Mayer/publication/308034642_Design_for_Collaboration_in_Mixed_Reality_Technical_Challenges_and_Solutions/links/5c814514299bf1268d448861/Design-for-Collaboration-in-Mixed-Reality-Technical-Challenges-and-Solutions.pdf

[26]. Dokumentácia Oculus Guardian System [online]. [cit. 2020-1-2]. Dostupné na internete: https://developer.oculus.com/documentation/native/pc/dg-guardian-system/

[27]. Balluch Tomáš. „Kolaborácia vo virtuálnej realite: vstupno-výstupná časť pre inteligentné telefóny“. V tlači. Diplomová práca. Technická univerzita v Košiciach, máj 2020

[28]. Ivan Michal. „Kolaborácia vo virtuálnej realite: komunikačná a riadiaca časť“. V tlači. Diplomová práca. Technická univerzita v Košiciach, máj 2020

[29]. Mozilla Hubs dokumentácia [online]. [cit. 2020-4-3]. Dostupné z: https://hubs.mozilla.com/docs/

[30]. Marián Hudák, Štefan Korečko, Branislav Sobota: Enhancing Team Interaction and Cross-platform Access in Web-based Collaborative Virtual Environments, In: Proceedings of 2019 IEEE 15th International Scientific Conference on Informatics, IEEE, 2019, pp. 160-164

Prílohy

Príloha A: DVD médium – diplomová práca v elektronickej podobe, prílohy v elektronickej podobe

Príloha B: Používateľská príručka

Príloha C: Systémová príručka