FAKULTA APLIKOVANÝCH VĚD
ZÁPADOČESKÉ UNIVERZITY
V PLZNI

KATEDRA INFORMATIKY
A VÝPOČETNÍ TECHNIKY

Bakalářská práce

Editor záznamů pro výuku ve virtuální realitě

Jan Purkart

PLZEŇ

2024

FAKULTA APLIKOVANÝCH VĚD
ZÁPADOČESKÉ UNIVERZITY
V PLZNI

KATEDRA INFORMATIKY
A VÝPOČETNÍ TECHNIKY

Bakalářská práce

Editor záznamů pro výuku ve virtuální realitě

Jan Purkart

Vedoucí práce
Ing. Petr Vaněček, Ph.D.

PLZEŇ 2024

Citace v seznamu literatury:
PURKART, Jan. *Editor záznamů pro výuku ve virtuální realitě*. Plzeň, 2024. Bakalářská práce. Západočeská univerzita v Plzni, Fakulta aplikovaných věd, Katedra informatiky a výpočetní techniky. Vedoucí práce Ing. Petr Vaněček, Ph.D.

ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta aplikovaných věd
Akademický rok: 2023/2024

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Jan PURKART**
Osobní číslo:	**A21B0250P**
Studijní program:	**B0613A140015 Informatika a výpočetní technika**
Specializace:	**Informatika**
Téma práce:	**Editor záznamů pro výuku ve virtuální realitě**
Zadávající katedra:	**Katedra informatiky a výpočetní techniky**

Zásady pro vypracování

1. Seznamte se se softwarem pro podporu výuky ve virtuální realitě vyvíjeným na KIV a s technologiemi použitými pro jeho vývoj.
2. Prozkoumejte aktuální možnosti použitého enginu Unity případně jiných nástrojů pro editování časových os záznamů.
3. Navrhněte vhodný způsob, jak zaznamenaná data editovat (střih, spojování, časová synchronizace zvuku a záznamu).
4. Proveďte potřebné změny na straně stávajícího software a následně implementujte navržené řešení a začleňte jej.
5. Řešení důkladně otestujte a zdokumentujte.

Rozsah bakalářské práce: **doporuč. 30 s. původního textu**
Rozsah grafických prací: **dle potřeby**
Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam doporučené literatury:

Dodá vedoucí bakalářské práce.

Vedoucí bakalářské práce: **Ing. Petr Vaněček, Ph.D.**
Katedra informatiky a výpočetní techniky

Datum zadání bakalářské práce: **2. října 2023**
Termín odevzdání bakalářské práce: **2. května 2024**

L.S.

Doc. Ing. Miloš Železný, Ph.D.
děkan

Doc. Ing. Přemysl Brada, MSc., Ph.D.
vedoucí katedry

V Plzni dne 25. října 2023

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů. Tato práce nebyla využita k získání jiného nebo stejného akademického titulu.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Západočeská univerzita v Plzni má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona.

v Nýřanech dne 23. února 2024

. . . . . .
Jan Purkart

V textu jsou použity názvy produktů, technologií, služeb, aplikací, společností apod., které mohou být ochrannými známkami nebo registrovanými ochrannými známkami příslušných vlastníků.

Abstrakt

Tato bakalářská práce se zabývá zaznamenáváním, editováním a přehráváním 3D záznamu z aplikace virtuální třídy a jeho zpětného přehrání. Už dnes se v českém školním systému využívá technologií virtuální reality k výuce. Tato práce má za úkol zlepšit prostředí výuky ve virtuální realitě, přepracovat aplikaci virtuální třídy, navrhnout a implementovat editor záznamu z virtuální reality.

Abstract

This bachelors thesis focuses on capturing, editing and finally playing the 3D recording of teaching in virtual classroom. Today, universities are already leveraging virtual reality technologies for education. This thesis aims to enhance the learning environment within virtual reality, revamp the virtual classroom application, and design and implement a virtual reality recording editor.

Klíčová slova

VR • Edukace ve VR • Editor 3D záznamu • Unity

Poděkování

Rád bych poděkoval především svému vedoucímu, Ing. Petru Vaněčkovi, Ph.D., který mi pomohl tuto práci dovést k úspěšnému konci a zároveň doc. Ing. Liboru Vášovi, Ph.D. za konzultace a pomoc při stanovení cílů.

Dále bych chtěl poděkovat své rodině a blízkým za podporu a trpělivost během celého studia.

V neposlední řadě si poděkování zaslouží také studenti, na jejichž práci navazuji: Ing. Hácha Filip, Vítek Por, Hejman Jakub a Lukáš Varga

Jan Purkart,
autor práce

Obsah

Úvod 1

Vývoj moderních technologií v oblasti vzdělávání nabízí inovativní přístupy k výuce a interakci studentů s učivem. Jedním z takových příkladů je aplikace virtuální třídy, která poskytuje prostředí pro výuku pomocí virtuální reality. Za pomoci zařízení pro virtuální realitu je pak možné předávat učivo žákům i v případech, kdy není kontaktní výuka možná. Tato práce se zabývá přepracováním existující aplikace virtuální třídy vyvinuté v prostředí Unity na Fakultě aplikovaných věd Západočeské univerzity v Plzni[1] a vytvořením editoru záznamu z 3D prostoru.

Jde o konstrukci nových základů dle poznámek a zpětné vazby, které vzešly z testování a používání předchozí verze virtuální třídy. Bakalářská práce zahrnuje analýzu stávajícího stavu v tomto odvětví, návrhy a architektury obou vznikajících aplikací a implementaci nových funkcí, nástrojů. Výsledkem by měl být funkční systém, ve kterém se záznam výuky a její průběh vykoná v virtuální třídě, následně se záznam vyedituje v aplikaci editoru a nakonec zpětně dosadí do virtuální třídy pro pozdější přehrání.

Dalším cílem je optimalizace výkonu aplikací a zlepšení uživatelského rozhraní pro plynulejší a efektivnější práci. Tímto způsobem se zvyšuje přitažlivost a užitečnost aplikací pro vyučující i studenty, což přispívá k lepšímu využití moderních technologií ve vzdělávání. Aplikace je teoreticky možné využívat i v jiných oborech. Například pro matematické účely či fyzikální simulace a jejich zobrazování v prostoru.

Po vytvoření obou aplikací je také nutné definovat omezení a možná rozšíření pro budoucí práci na tomto projektu. Tato práce má také za úkol otestovat obě aplikace a vytvořit obsah, který jejich použití více zpřístupní.

[1] Dále zkráceně Západočeská univerzita nebo ZČU

Seznámení s technologiemi

2

Myšlenka edukace ve VR vznikla již v roce 1992 [Hel92]. Dnes existuje více směrů, ve kterých lze virtuální realitu využít k výuce. Základními principy je interaktivita a vizuální zpětná vazba. Těchto principů využívá například aplikace virtuální třídy [HVV21], která byla vytvořena na ZČU v roce 2020. Aplikace je implementována v prostředí pro vývoj her a její síťová vrstva je implementována přes externí C# server. Cílové zařízení pro aplikaci původní virtuální třídy byla Oculus 1 a Oculus 2, která byla v době vývoje aplikací veřejně dostupná.

2.1 XR, VR, AR

Vzhledem k tématu práce je také nutné definovat a vysvětlit pojmy XR, VR a AR. XR[1] je zkratka pro "Rozšířenou realitu"a označuje technologie, které spojují reálný svět s virtuálním světem. XR zahrnuje virtuální realitu (VR), rozšířenou realitu (AR). VR je interaktivní počítačové prostředí, které simuluje fyzickou přítomnost uživatele v imaginárním světě. Uživatelé mohou prozkoumávat a interagovat s prostředím pomocí speciálních zařízení, jako jsou VR headsety. AR je technologie, která umožňuje přidávat digitální obsah do reálného světa. Uživatelé mohou vidět reálný svět skrze zařízení, jako jsou chytré telefony nebo AR brýle a do tohoto světa se mohou přidávat virtuální objekty nebo informace.

2.2 VR zařízení

Na trhu se vyskytuje mnoho zařízení pro virtuální realitu. Aktuálně jsou ke dni 3.2. nejpoužívanějšími zařízeními na trhu v Česku:

- Meta Quest 2 - starší model od firmy Meta Platforms, Inc., OS:Android
- Meta Quest 3 - novější model od firmy Meta Platforms, Inc., OS:Android

[1] XR = Extended Reality, VR = Virtual Reality, AR = Augmented Reality

- Playstation VR2 - VR zařízení od firmy Sony Corporation, nutnost připojení k Playstation 4 nebo 5
- HTC Vive PRO 2 - zařízení od firmy HTC Corporation, dražší produkt slibující vyšší kvalitu

Nutno zmínit, že ve Spojených státech amerických čerstvě vyšlo zařízení Apple Vision Pro, avšak v Čechách ještě není komerčně dostupné. Jako další zařízení bychom pak měli zmínit Valve Index, Odyssey+ nebo Pico Neo 3. Z tohoto seznamu byly vybrány jako cílové zařízení Meta Quest 2 a Meta Quest 3, které běží na operačním systému Android a lze je jednoduše připojit k SteamVR a k desktop aplikacím.

2.3 PCVR

PCVR znamená "PC Virtual Reality"a označuje virtuální realitu, která je provozována přes počítač. Uživatelé se připojují k VR brýlím k výkonnému počítači, který generuje a renderuje obsah virtuální reality. To umožňuje více výpočetního výkonu a pokročilejší grafiku než u mobilní VR. PCVR také často umožňuje využití pokročilých ovladačů a senzorů pro sledování pohybu, což přispívá k interaktivitě a realismu ve virtuálním prostoru.

2.4 Možná prostředí pro vývoj aplikací

V úvahu jako základ pro aplikace přichází několik možností herních enginů.

2.4.1 Godot

Jednoduché, rozšiřitelné open-source prostředí [God18], které však neobsahuje velkou podporu pro VR aplikace. Jako programovací jazyky využívá C++, C# a vlastní GDScript. Lze v něm tvořit jak 2D tak 3D scény.

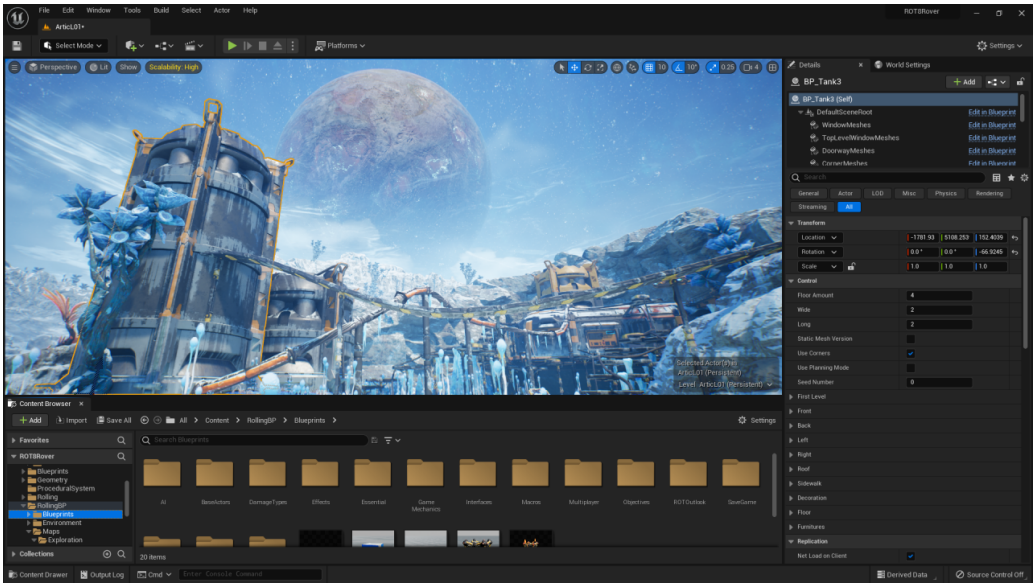
2.4.2 Unreal

Prostředí s vysokou vizuální kvalitou a podporou pro VR, ale na druhou stranu působí těžkopádně a není vhodné pro malé týmy [Unr23]. Je velice efektivní pro 3D scény, simulace či vytváření 3D filmů.

2.4.3 Unity

Podporuje oproti konkurenci VR aplikace pomocí mnoha knihoven[Uni30]. Je vhodnější pro menší týmy či jednotlivce. Má také velkou výhodu v možnostech exportu do více platform.

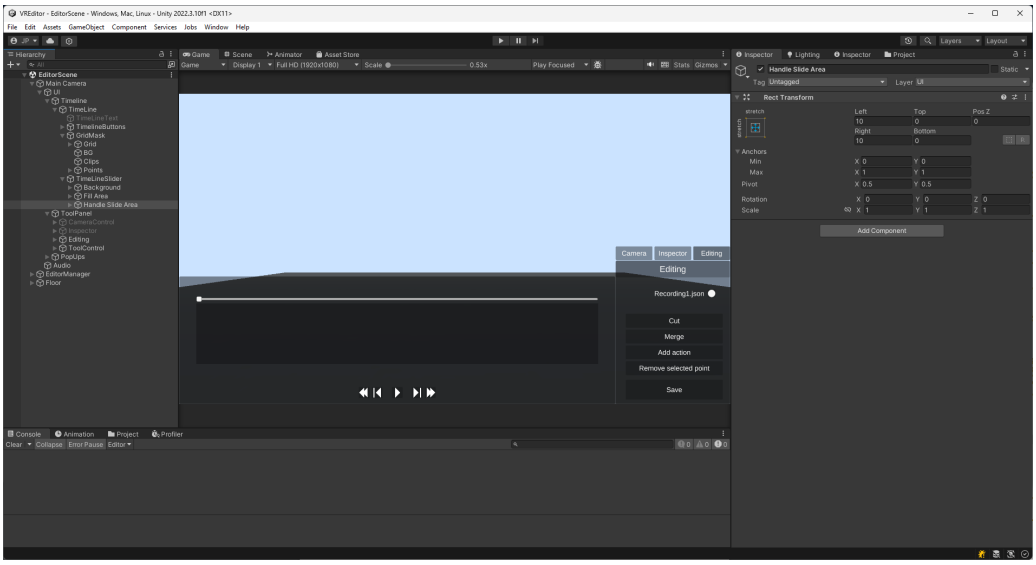
Obrázek 2.1: Ukázka enginu Godot [Val12]



Obrázek 2.2: Ukázka enginu Unreal [Unr23]

2.5 Unity

Unity je herní eninge, který je široce využíván při vývoji her, simulací, virtuální reality a dalších interaktivních aplikací. Je vyvinut společností Unity Technologies a poskytuje prostředí pro tvorbu multimediálních aplikací napříč různými platformami včetně počítačů, mobilních zařízení a herních konzolí. Díky své široké podpoře platforem umožňuje Unity tvůrcům vytvářet aplikace pro různé operační



Obrázek 2.3: Ukázka enginu Unity (editor)

systémy a zařízení, což zahrnuje počítače s Windows, macOS a Linuxem, mobilní telefony a tablety s Androidem a iOS, herní konzole jako PlayStation, Xbox a Nintendo Switch, a dokonce i rozšířenou realitu (AR) a virtuální realitu (VR) zařízení. Zvolení Unity jako prostředí pro vývoj obou aplikací přispívá k jednoduchosti implementace budoucích rozšíření.

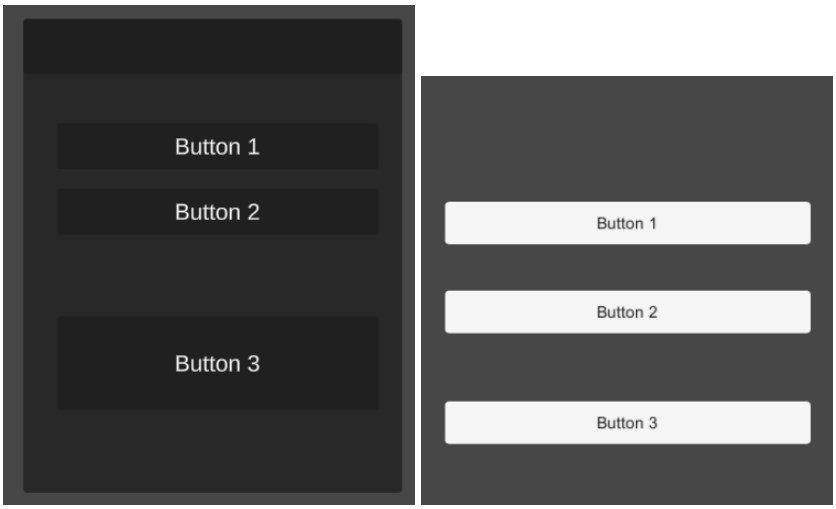
2.5.1 3D scéna

V Unity je 3D scéna základním konceptem pro vytváření virtuálních prostředí, ve kterých se odehrává interaktivní hra nebo simulace. Scéna je místo aplikace, které obsahuje veškeré vizuální a interaktivní prvky, jako jsou objekty, uživatelé, osvětlení, kamery a další.

2.5.2 Unity UI

V Unity se uživatelské rozhraní[2] dá vytvářet za pomocí grafických prvků, tlačítek a posuvníků. Při vývoji aplikací byla vyvinuta nová uživatelská rozhraní, která mají naprogramovaná vlastní chování, avšak vychází z těchto interaktivních primitiv. Kvůli zastaralému návrhu původního UI od Unity bylo navrhnuto vytvoření jednotného stylu z dostupných primitiv, které Unity nabízí. Porovnání můžeme vidět na obrázku 2.4, kde na levé straně je nově stylizované UI a na pravé straně se nachází základní verze od Unity.

[2] dále UR = uživatelské rozhraní / UI = user interface



Obrázek 2.4: Porovnání vlastního a základního Unity UI

2.5.3 URP – Universal Render Pipeline

Universal Render Pipeline (URP) je grafickým vykreslovacím systémem. Optimalizuje výkon díky moderním technikám se zachováním dostatečných detailů. Nabízí přednastavená nastavení pro rychlou tvorbu a umožňuje vytvářet vlastní shadery pro specifické efekty. Podporuje multi-platformní nasazení a snadnou přenositelnost mezi zařízeními. Alternativou k URP je pak HDRP, který se soustředí na vysokou kvalitu a velké množství detailů ve scéně. Dalším rozdílem je pak přístup k vývoji například pro osvětlení, stíny nebo reflexe. Nevýhodou je větší složitost nastavení a vyšší minimální požadavky výsledné aplikace. URP bude využito jak při vývoji editoru, tak nové virtuální třídy.

2.5.4 Playerprefs systém

Tento systém slouží pro lokální ukládání a načítání informací z aplikace do zařízení. Používá se pro dlouhodobé nastavení preferencí uživatele. Použití v praxi můžeme vidět na tomto kódu:

Zdrojový kód 2.1: Příklad využití PlayerPrefs systému

```
1 public void
2 PlayerPrefsExample(NetworkObjectToSend objectTD)
3 {
4   if (PlayerPrefs.GetFloat("key") >= 1)
5   {
6     PlayerPrefs.SetInt("unlocked",1);
7   }
8 }
```

2.5.5 Programování v Unity

Programování v Unity probíhá v jazyce C#, kde se zároveň využívá předem připravených struktur [Uni30]. Unity má například své vlastní základní třídy a komponenty, které ulehčují vývoj z pohledu programování. Například dále často zmiňovaná třída `Vector3` obsahuje základní složky: `float` x, `float` y, `float` z a související výpočetní funkce ze světa 3D grafiky. Jednou z komponent je například `LineRenderer`, který postupně zobrazuje úsečky mezi seznamem bodů. Třída `Color` reprezentuje barvu v Unity, hodnoty kanálů pro `rgba (red, green, blue, alpha)` jsou v rozmezí 0 až 1.

Unity také obsahuje své vlastní metody. Nejdůležitějšími metodami jsou:

- Awake() - Je volána jednou, když objekt vytvoří instanci skriptu nebo se objekt stane aktivním. Často se používá k inicializaci proměnných nebo k nastavení počátečních hodnot.
- Start() - Je volána jednou, když je objekt aktivován a je připraven na spuštění, těsně po volání Awake(). Je často využívána pro nastavení scény při inicializaci a pro konfigurace objektů, které jsou závislé na jiných objektech nebo skriptech ve scéně.
- Update() - Je volána jednou za každý snímek a je vhodná pro zpracování vstupů, pohybu a jiných činností, které se mají provádět každý snímek.
- FixedUpdate() - Je volána před každým fyzikálním výpočtem. Je vhodná pro zpracování fyzikálních operací nebo pohybu, protože se provádí s konstantní frekvencí nezávisle na rychlosti snímání.
- OnCollisionEnter() - Je volána, když se tento Collider dotkne jiného Collideru. Je používána pro detekci kolizí a interakcí mezi herními objekty. Existuje i nekolizní alternativa pro Trigger Collider komponentu.

2.5.6 Netcode

Unity knihovna pro síťovou vrstvu aplikací v programovacím jazyce C# [Mic22]. K dispozici v Unity projektech od verze 2021. Má možnost přímého napojení na Unity Gaming Services (UGS), avšak tato možnost je spíše pro aplikace s větším počtem uživatelů, a tak není v aplikaci virtuální třídy využita. Umožňuje vytvářet spojení typu Host-Client, serializovat a deserializovat očekávaná data, posílat RPC volání. Aplikace virtuální reality využívá verze 1.6.0. V budoucí verzi 1.7. je plánováno přidat podporu TCP, což umožní vytvořit efektivní přenos pro trojúhelníkové sítě, či přenos objemnějších dat přes síťovou vrstvu.

Metody, které používají Netcode, využívají takzvané atributy metody, ty jsou definované ve hranatých závorkách nad metodou. Například takto:

Zdrojový kód 2.2: Příklad ServerRPC metody

```
1  [ServerRpc]
2  public void
3  DespawnObjectServerRpc(NetworkObjectToSend objectTD)
4  {
5    NetworkObject netObject = objectTD.networkObject;
6    if (netObject == null)
7    {
8      return;
9    }
10
11   spawnHistory.Remove(netObject.gameObject);
12   netObject.Despawn(true);
13   Destroy(netObject.gameObject);
14 }
```

Netcode se dá snadno napojit na další běžné služby, jako například externí hostování serveru a analytické balíčky. Práce v Unity pak vypadá tak, že se do scény musí vložit objekt s komponentou `NetworkManager`, která se stará o detailní nastavení síťové vrstvy. Před exportováním aplikace se musí přímo v Unity projektu nastavit objekty, které budou vytvářeny při připojení a se kterými se bude interagovat.

Podobné aplikace 3

Tato kapitola se týká průzkumu a také porovnání podobných aplikací pro virtuální třídu a v druhé sekci pro editor 3D záznamu.

3.1 VR huby

VR huby jsou platformy pro virtuální realitu, které umožňují uživatelům interakci a komunikaci ve virtuálním prostředí prostřednictvím avatarů.

3.1.1 VR Chat

Tato aplikace se stala významným hráčem v oblasti sociálních VR prostředí, kde uživatelé mohou prozkoumávat různé virtuální světy, setkávat se s ostatními uživateli a komunikovat pomocí hlasu, textových zpráv a gest. Její první verze vyšla již v roce 2014 a byla tedy jednou z prvních na trhu. Aplikace je vyvíjena v Unity.

Obrázek 3.1: Ukázka aplikace VR Chat

3.1.2 Horizon Worlds

Horizon Worlds je alternativní platforma pro virtuální realitu vyvinutá společností Meta (dříve Facebook) a vydaná v roce 2019.

Obrázek 3.2: Ukázka Horizon Worlds

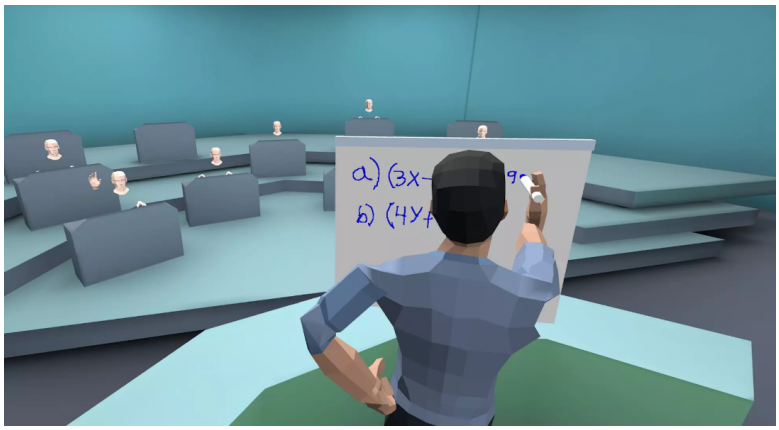
3.1.3 VR Classroom template

Na obchodě pro platformu Unity se také vyskytuje šablona projektu pro virtuální třídu. Tento projekt běží na síťové vrstvě Photon Pun 2[1]. Poskytuje hlasovou komunikaci přímo v aplikaci, kreslení na tabuli a přehrávání prezentace. Neposkytuje však vkládání objektů či prostorové kreslení. I když má podobný cíl jako projekt virtuální třídy na ZČU, není zcela stejný. Tato šablona by se mohla hodit jako dobrý základ pro výuku ve VR, nevyužívá ale plně možností trojrozměrného prostoru. Tato šablona stojí 60$.

3.2 Editory

Při hledání aplikací podobných vyvíjenému editoru nebyly nalezeny shody. Aplikace se nejblíže podobá právě systémům ve vývojovém prostředí Unity nebo Unreal. Proto zde budou popsány tyto příklady.

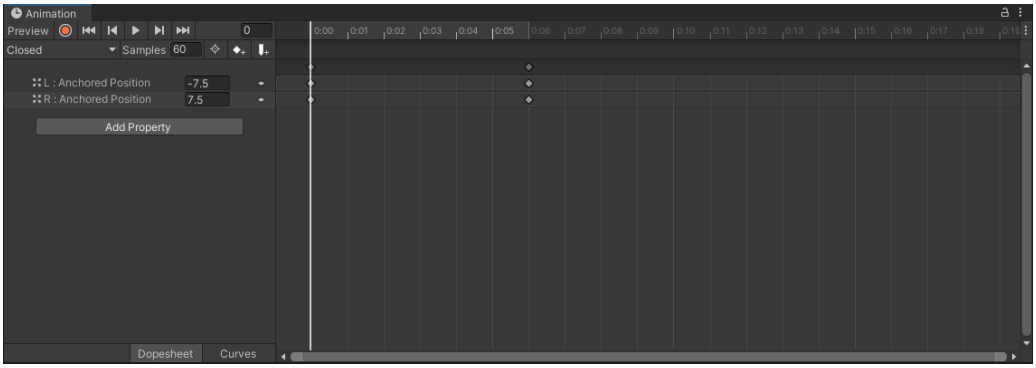
[1] https://assetstore.unity.com/packages/tools/network/pun-2-free-119922 [Uni05]



Obrázek 3.3: Ukázka VR Classroom template

3.2.1 Unity Animator

Animátor je hlavní inspirací pro časovou osu v návrhu editoru. Skládá se ze dvou částí. První určuje chování mezi různými animacemi, jejich přechody podle parametrů či jejich rychlost přehrávání. Druhá pak slouží k editaci a ručnímu vytvoření akcí, které vytváří jednu z animací použitých daným objektem.



Obrázek 3.4: Ukázka animátoru Unity

3.2.2 Unity Timeline

Unity Timeline je složitější verze animačního nástroje připravovaná především pro neinteraktivní části hry, ve které se uživatel soustředí především na připravenou animaci. Kromě základní funkčnosti animátoru obsahuje také možnosti lepší organizace do složek, přehrávání zvuků a zaměřování kamery.

Návrh 4

4.1 Návrh úprav ze strany stávajícího softwaru

Stávající software virtuální třídy vznikl v Unity verzi 2019.4.24f1. Tato verze je zastaralá a nelze v ní využívat nové možnosti pro vývoj, které novější verze nabízí. Proto vznikl návrh předělat od základu VRC1 a vytvořit její novou verzi. Funkce aplikace byly implementovány jen ty, které byly ověřené při používání první verze jako praktické a užitečné. Hlavní myšlenkou je vytvořit novější základy, na kterých bude možnost v budoucnu stavět a které lze rozvíjet. Toto rozhodnutí vyplynulo ze společné debaty v roce 2023. Hlavními důvody pro přepracování aplikace do nové verze Unity byly udržitelnost, čistota kódu, využití knihovny Netcode a možnost napojit záznamový systém pro akce.

4.2 Návrh editoru

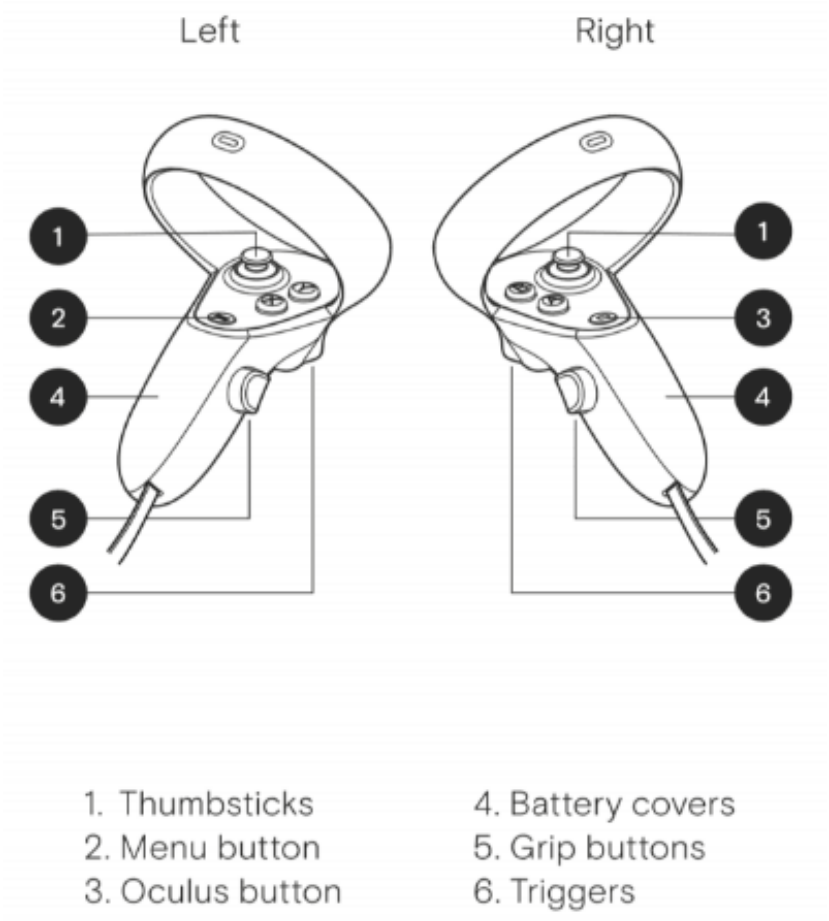
Aplikace editoru záznamu z virtuální reality by měla obsahovat možnosti vkládání a exportování dat společně s funkcemi k jejich manipulaci. Celá aplikace bude probíhat v jedné Unity scéně. Obrazovka bude rozdělena na několik částí:

- Inspektor akce
- Časová osa s akcemi
- Inspektor kamery
- Zobrazení 3D scény

4.2.1 Návrh uživatelského ovládání

Jediná možnost, jak může uživatel ovládat aplikaci, bude přes ovladače VR zařízení. Tyto ovladače, jak můžeme vidět z obrázku 4.1, mají k dispozici jeden joystick [1], dvě tlačítka [X,Y / A,B], jedno grip [5] tlačítko a jedno trigger tlačítko [6]. Na ovladači

se také vyskytuje tlačítko pro menu a funkce Oculus, to však není k dispozici pro vstup z aplikací. Uživatel bude muset mít přístup k mnoha nástrojům, proto bude nutno vymyslet systém, který mezi nimi bude umět přepínat. Mezi základní nástroje bude patřit vytváření objektů, nástroj pro změnu parametrů tvorby, pohyb po 3D scéně, ukazatele a ukazovátko. Vzhledem k možnosti budoucího rozvoje nástrojů bude nejlepší rozdělit výběr nástrojů do několika segmentů. Prvním bude vybírání nástroje, druhým tvorba / mazání, třetím pak signalizace.



Obrázek 4.1: Ukázka Oculus ovladačů

4.2.2 Návrh práce se soubory

Vstup dat do aplikace bude jen skrze daný JSON formát. Na práci s daty by se měl použít souborový formulář, který vrátí cestu k složce s daty.

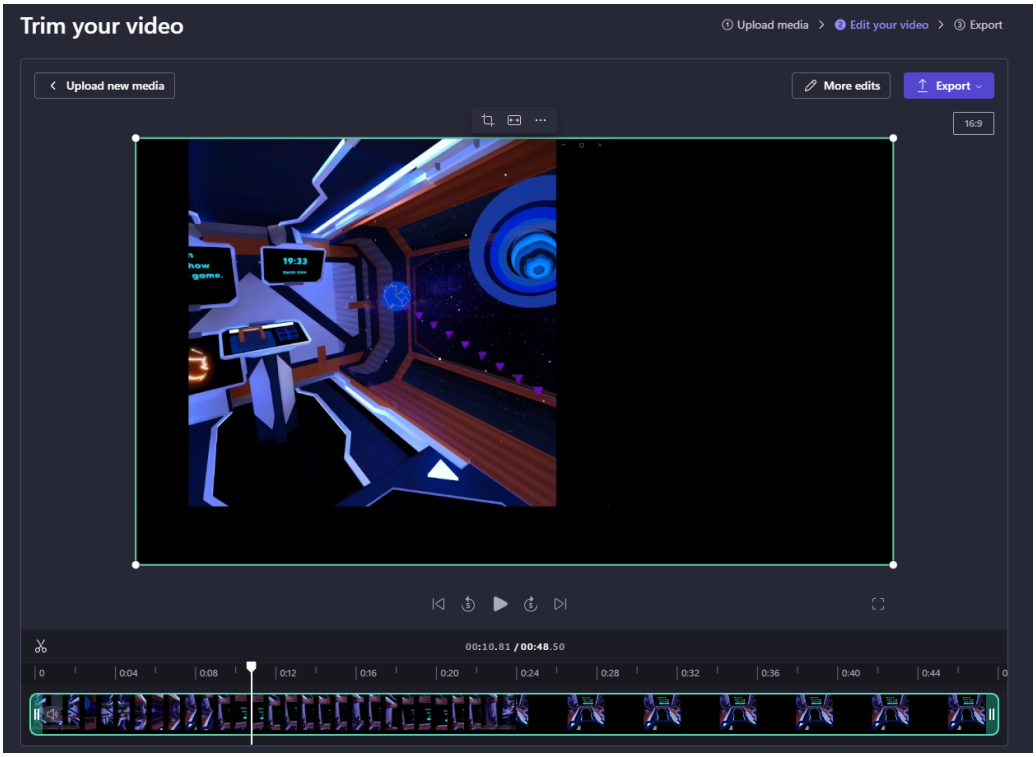
Složka by měla obsahovat soubory pro načtení, tj.:

- .json soubor s akcemi
- .mp3 soubory, které se v záznamu využijí

Očekávané velikosti jsou pro JSON soubor v rámci Kb/Mb, pro .mp3 soubory jsou to jednotky až desítky Mb. Aplikace by také měla obsahovat nějakou formu automatického ukládání, pro případ že uživatele potká během editace vyrušení.

4.2.3 Návrh časové osy

Časová osa je nástroj pro přesun v časově orientovaném záznamu. Existuje několik různých implementací, ze kterých je většina určena pro editování 2D videa. Například u ukázky z editoru ClipChamp (4.2) můžeme vidět i zobrazení snímků pro daný časový úsek. Druhým příkladem je Animační časová osa v Unity, ta obsahuje jednotlivé snímky jako sloty, které mohou obsahovat akce. K této časové ose má záznam blíž a proto bude hlavní inspirací pro vlastní nástroj.



Obrázek 4.2: Ukázka aplikace Microsoft ClipChamp

4.2.4 Návrh panelu nástrojů

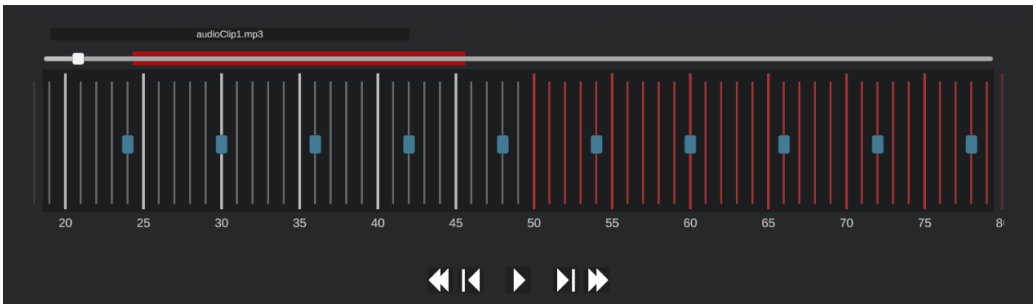
Tento panel by měl obsahovat tři nástroje. Prvním z nich je nástroj pro ovládání kamery. Kameru lze zde buď ručně nastavit, navrátit do původní pozice, či odemknout a zamknout pohyb. Druhým nástrojem je inspektor akcí, který zobrazuje informace o aktuálně zvoleném bodu akce. Třetím nástrojem je pak nástroj pro editaci záznamu, tento nástroj umožní vytvořit nový bod akce, vybraný smazat, uložit celý záznam nebo načíst nový.

4.2.5 Možnosti editace

Při načtení záznamu se načte soubor a vytvoří se kopie složky pro export, která se pak dále upravuje. Editaci prozatím zařizují 4 funkce. Střih, změna, odebrání a přidání bodu záznamu. Střih je funkce, která má dva časové parametry *start* a *konec*. Při střihu by se měli odstranit všechny body akcí mezi těmito dvěma parametry a každá následující akce se posune o vzorec proměnné *posun*.

$$posun = konec - start$$

Časové okno střihu by se pak vizuálně mělo promítnou i do časové osy. Vložení nového bodu akce by mělo zapojit za pomoci formuláře novou akci. Změna vybraného bodu by pak měla probíhat podobně. Tento formulář bude mít větší počet parametrů, aby mohl přesně definovat datovou strukturu. Všechny akce by měli být ihned vidět jak na uživatelském rozhraní v aplikaci, tak později v exportovaném .json souboru.



Obrázek 4.3: Ukázka střihu

Úprava ze strany stávajícího softwaru

5

Vzhledem ke starší verzi základů stávající aplikace virtuální třídy se nabízeli dvě možnosti. A to právě migrace či přepracování aplikace do novější verze Unity. Migrace však měla mnoho překážek jako například kompatibilitu vykreslovacího sytému a zároveň by stále zabrala kolem desítek hodin, vzhledem k přehlednosti projektu. Na základě domluvy s vedoucím bakalářské práce a klientem Doc. Ing. Liborem Vášou, Ph.D. tak byla vytvořena nová aplikace od základů až po základní funkcionalitu. Tato aplikace zachovává účel původní aplikace, tedy je to cílená VR aplikace pro výuku, jejíž základní funkčnost činí, že se více uživatelů může připojit do edukačního virtuálního prostoru. Aplikace byla otestována během období domácí výuky (Covid 19) v roce 2020 přímo výukou na FAV [Poó22] u předmětu KIV/ZPOS, což umožnilo získat více zpětné vazby pro vývoj nové verze.

5.1 VRC2 úvod

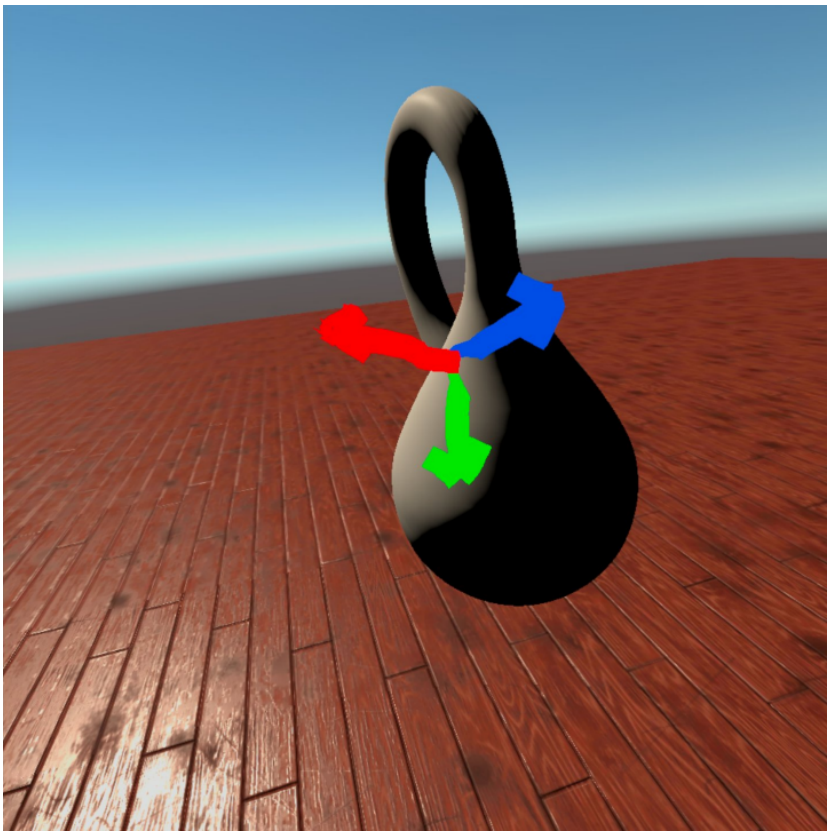
Aplikace bude podobně jako její předchůdce vyvíjena v Unity. Původní aplikace byla ve verzi (2019.4.24f1), VRC2 je ve vyvinuta ve verzi (2022.3.10f1). Díky novější verzi můžeme využívat nové knihovny pro VR a hlavně síťovou knihovnu Netcode [2.5.6]

5.2 Cílová platforma

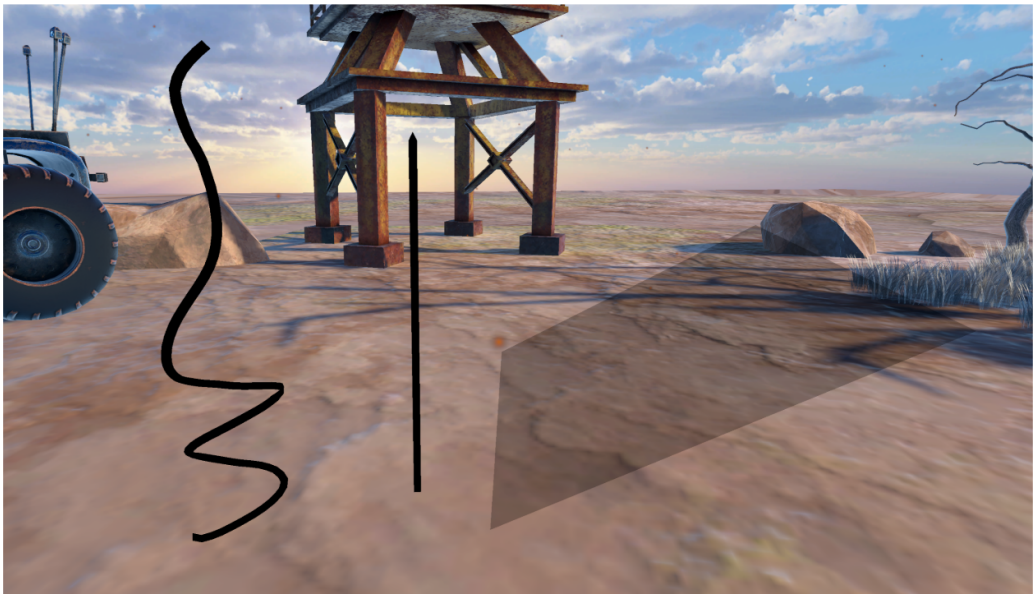
Cílovou platformou je PCVR a do budoucna také android zařízení Oculus 2, Oculus 3. Aplikace je však stále zpětně kompatibilní pro Oculus 1. Zatímco pro zařízení Oculus je nutné exportovat android .apk instalační soubor, pro PCVR je zapotřebí vytvořit .exe soubor.

5.3 Základní funkčnost VRC2

Lobby je virtuální prostor, kde se uživatelé setkávají. Hostování je proces, kdy jeden uživatel vytváří a spravuje hru, na kterou se ostatní uživatelé připojují. Uživatelé



Obrázek 5.1: Ukázka původní virtuální třídy (VRC1)



Obrázek 5.2: Ukázka primitiv [Poó22]

mohou vytvářet a hostovat lobby nebo se k němu připojit. Uživatelé mohou do prostoru kreslit, vkládat plochy a úsečky různých velikostí a barev, vkládat trojúhelníkové sítě. Uživatelé mohou změnit svou výšku. Uživatelé mohou zaznamenávat aktivity do .json souboru, který se pak dále může editovat v editoru VR záznamů. Všechny tyto akce jsou promítnuty i na síťové vrstvě.

5.4 Ovládání aplikace

Pravý ovladač slouží k teleportaci, rotaci a k signalizačním funkcím. Teleportaci probíhá pomocí přetáhnutí páčky joysticku ovladače směrem od uživatele. Rotace pak přetáhnutím směrem k uživateli. Uživatel může také zobrazit informační symboly. Tyto symboly slouží učiteli jako nonverbální zpětná vazba. Tyto informační symboly jsou dva. Vykřičník se zapne pomocí [A]. Otazník se zapne pomocí [B]. Paralelně k signalizačním funkcím může uživatel využívat jednoho ze tří nástrojů. Tyto nástroje se ovládají pomocí levého ovladače. Mezi nástroji lze přepínat pomocí promáčknutí joysticku na levém ovladači. První je pro tvorbu objektů, ve kterém lze vytvářet typy definované v 6.3, zkráceně jde o krychle, malování, úsečky a plochy. Vytváření objektů má 3 části. První částí je vytvoření, druhou modifikace, třetí potvrzení. Pro kreslení je modifikací přidávání bodů a při potvrzení se musí vypočítat trojúhelníková síť z bodů. Pro zbytek je modifikací myšleno změna pozice, rotace a škály objektu. Tato modifikace probíhá podle pohybu a rotace ovladače. Aplikace nejdříve vytvoří objekt z prefabu, který je předem definovaný, má tedy nutné komponenty. Pro malování se pohybem ruky body posílají přes HostRPC, které je rozešle všem klientům přes síťové ClientRPC, kteří je zasadí do komponenty Line Renderer a zobrazí je v aplikaci. Funkci ukazovátka lze zapnout pomocí tlačítka [Trigger]. Při zmáčknutí tlačítka se zobrazí raycast paprsek od pozice ovladače do pozice prvního nárazu či maximální vzdálenosti, nastavitelné v Unity projektu.

5.5 Architektura VRC2

5.5.1 Rozložení aplikace

Aplikace se skládá ze sedmi větších částí: nastavení projektu, struktury scén, ovládání uživatele, síťové vrstvy, herních nástrojů, přehrávání a nahrávání záznamu, nástrojové panely.

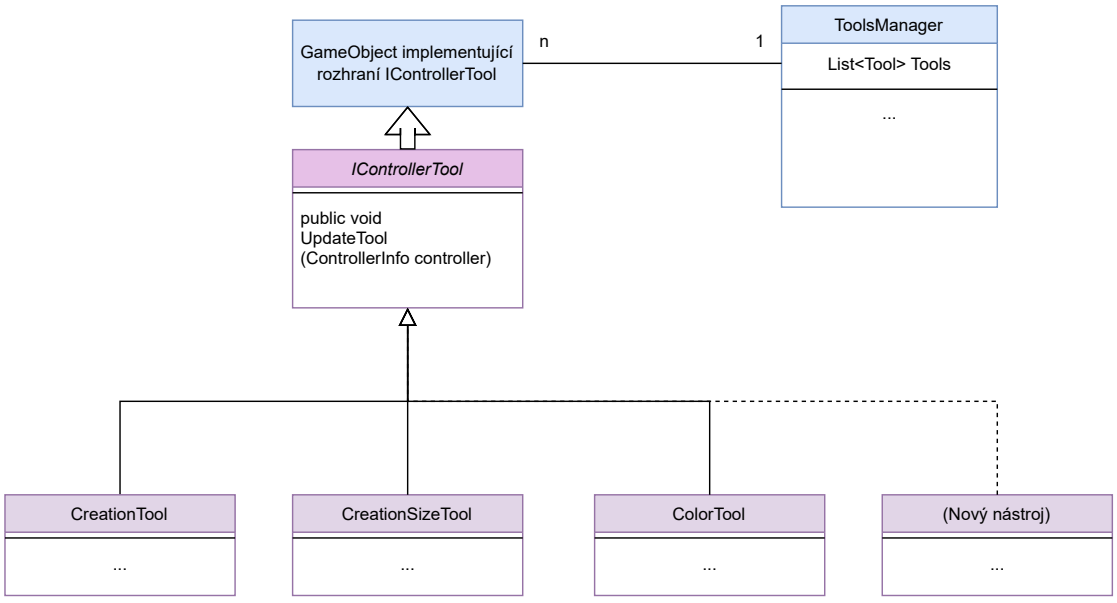
5.5.2 Nástroje a funkce

Nástroj pro nastavení velikosti zobrazuje aktuální možnosti objektů s aktuální nastavenou velikostí a základní černou barvou. Pomocí přetáhnutí joysticku doleva a doprava může uživatel měnit velikost. Velikost je omezená v projektu minimem a

maximem. Nástroj pro tvorbu objektů umožňuje pomocí přetáhnutí joysticku ovladače vybrat typ objektu, který chce uživatel vytvořit. Pak pomocí stisknutí tlačítka [X] začne objekt vytvářet a pomocí tlačítka [X] dokončí vytváření objektu a objekt položí. Nástroj také nabízí možnost mazat uživatelem vytvořené objekty. Toto chování je založeno na kolizním systému. Kolem ovladače je vytvořen kolizní objekt s určitou vzdáleností, který při zmáčknutí tlačítka [Y] detekuje, jestli jsou některé vytvořené objekty dostatečně blízko a popřípadě je odstraní. Jak lokálně tak u ostatních klientů po síti. Nástroj pro vybrání barvy uživateli zobrazuje kulatý panel s odstínem[1] a saturací, sloupec ukazující světlost a k tomu malý kulatý objekt, na kterém se ukazuje aktuální barva.

5.5.3 Diagram nástrojů

Diagram ukazuje, jaké nástroje implementují rozhraní IControllerTool a vazbu mezi tímto rozhraním přímo v správci nástrojů uvnitř hlavní scény nové virtuální třídy. Přerušovaná čára zde značí blok pro nový nástroj. Jednotlivé implementace rozhraní mají desítky vlastností a metod. Diagram ukazuje jen obecné vazby.



[1] HSV model: odstín = hue, saturace = saturation, světlost = value

5.5.4 Implementace nástrojů

Nástroje byly vyvinuty tak, aby se systém dal snadno rozšiřovat. Nutnou podmínkou je implementování rozhraní, které obsahuje metodu pro aktualizování nástroje[2]. Každý snímek nástroj dostane jako vstup stav ovladače a je přímo na implementaci daného nástroje, jak s ním naloží. Potom se nový skript nástroje nasadí na objekt a ideálně uloží do hierarchie pod objekt levého ovladače, kde je i zbytek nástrojů. Jako poslední je nutno vložit nově vytvořený objekt do listu nástrojů v `ToolsController.cs`. Toto vložení je ideální udělat přes inspektor v Unity a vložit objekt obsahující skript pro nástroj. Kód pro načtení nástrojů pak vypadá takto:

Zdrojový kód 5.1: Příklad přebírání nástroje do listu v `ToolsController.cs`

```
1  void Awake()
2  {
3    ...
4    tools = new List<ControllerTool>();
5    foreach (GameObject gameObjectWithTool in
     toolsGameObjects)
6    {
7      ControllerTool tool = gameObjectWithTool.GetComponent
     <ControllerTool>();
8      if (tool != null)
9      {
10       tools.Add(tool);
11       Debug.Log("Added tool:" + gameObjectWithTool.name);
12     }
13   }
14
15   ShowTool();
16   ...
17 }
18
```

Obnovení nástrojů probíhá v Unity metodě `void Update`, která je volána při vykreslení každého snímku. Jako parametr pro toto obnovení vstupuje odkaz na ovladač, ze kterého lze uvnitř nástroje načíst vstupy. Například implementace rozhraní pro nástroj určující velikost vypadá takto:

[2]Aktualizování probíhá každý snímek, tedy cca 90krát za vteřinu.

Zdrojový kód 5.2: Příklad implementace - nástroj pro velikost

```
void UpdateTool(InputDevice controller)
{
  controller.TryGetFeatureValue(UnityEngine.XR.CommonUsages
   .primary2DAxis, out Vector2 axisValue);
  float oldSize = size;
  size += stepIncrease * axisValue.x;
  bool changed = false;
  if (oldSize != size)
  {
    textField.text = (Mathf.Round(size * 10) / 10f).
   ToString();
    PlayerPrefs.SetFloat("Size", size);
    changed = true;
  }

  if (size < min)
  {
    size = min;
  }
  if (size > max)
  {
    size = max;
  }

  if (changed)
  {
    UpdateShowcaseObjects();
    UpdateBar();
  }
}
```

5.5.5 Přehrávání a nahrávání záznamu

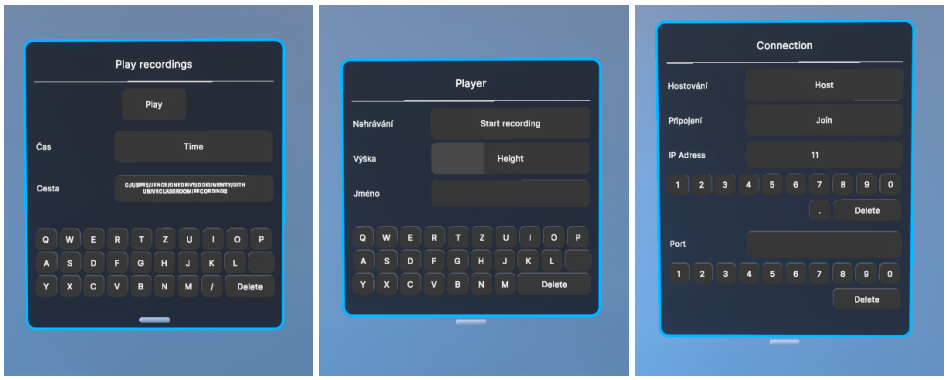
Nahrávání záznamu probíhá přes nástrojové panely, kde lze zapnout a vypnout záznam akcí a mikrofonu. Akce i mikrofon se nahrávají najednou po zmáčknutí tlačítka `Recording`. Po začátku nahrávání se text tlačítka změní na Stop Recording.

5.5.6 Struktura scén

V aplikaci VRC2 se využívá celkem čtyř scén. První scéna je pro načtení základů pro síťovou vrstvu, ihned po načtení se načte scéna hlavní. V hlavní scéně se vyskytuje většina aplikace. Další dvě scény slouží pro nastavení prostředí. Projekt má také připravené scény pro změnu prostředí, jednu noční a jednu denní. Přepínání mezi různými prostředími však bude implementováno až v dalších částech projektu.

5.5.7 Panely

Panely v hlavní scéně najdeme tři. První slouží pro nastavení parametrů pro síťovou vrstvu a hostování nebo připojení. Zde lze nastavit port a cílovou IP adresu hosta. Druhým panelem uživatel mění svou virtuální výšku, jméno v aplikaci a má možnost zapnout nahrávání akcí. Třetí panel pak slouží pro přehrání editovaného záznamu. U tohoto panelu je nutné nastavit cestu ke složce se záznamem.



Obrázek 5.3: Ukázka panelů

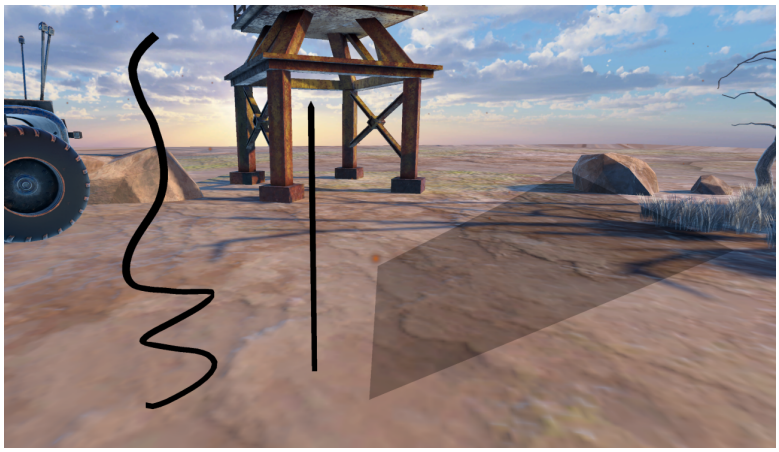
5.6 Knihovny využité v VRC2 a editoru

Knihoven se v projektu využívá mnoho, proto zde zmíním jen ty nejdůležitější a externí.

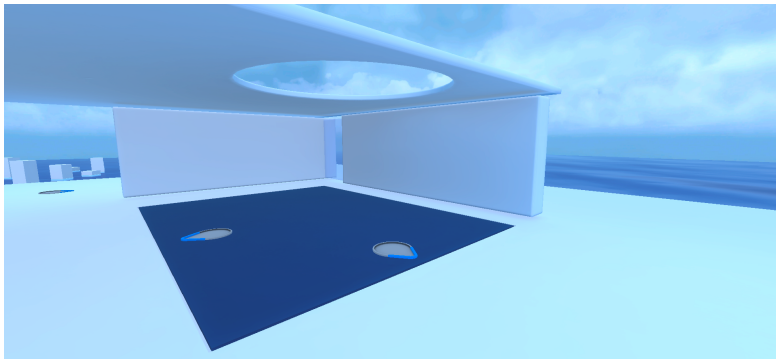
- Netcode - Knihovna pro síťovou vrstvu aplikace.
- SimpleFileBrowser - Uživatelské rozhraní na vložení cesty k souborům.
- Shader graph - Sestavování shaderů a materiálů za pomocí bodového grafu.
- Android logcat - Debug a analyzování běhu aplikace.
- XR core, OpenXR, OculusXR - podpora VR funkcí.

5.7 Grafické porovnání VRC1 a VRC2

V této sekci bych chtěl porovnat scény starší a novější verze virtuální třídy. Byl kladen důraz na zpětnou vazbu ohledně pozornosti, kterou prostředí může brát od uživatele během výuky. Nové prostředí je navrženo z geometrických primitiv a s myšlenkou minimálního ruchu.



Obrázek 5.4: Porovnání hlavních scén - starší verze



Obrázek 5.5: Porovnání hlavních scén - nová verze

5.8 Struktura aplikace VR Classroom 2.0

- VRClassroom
  - Assets
    - Materials
    - NetworkPrefabs ..... Objekty pro síťové prvky aplikace
    - Player ..... Objekty a animace pro uživatele
    - Plugins ..... Externí knihovny
    - Prefabs ..... Objekty nástrojů
    - Scenes ..... Herní scény
    - Scripts ..... Scripta pro fungování aplikace v C#
    - Settings ..... Nastavení pro rendering, světlo, post processing
    - XR, XRI ..... Nástroje a objekty pro zjednodušení vývoje
  - Builds
  - Library
  - Logs
  - Packages
  - ProjectSettings

Data mezi aplikacemi 6

Aplikace mezi sebou komunikují obousměrně. VRC2 bude vytvářet soubor s daty, který pak editor využije jako vstup a ten pozmění. Po ukončení editace lze data nahrát zpět do VRC2 a přehrát ve 3D prostoru.

6.1 Možné formáty

Formát by měl vyplývat z povahy dat a jednoduchosti práce. Popřípadě také z velikosti dat. Jako jedna z možností přicházely následující formáty:

- CSV (Comma-Separated Values): Jednoduchý formát, ve kterém jsou data oddělena čárkami. Často se používá pro tabulková data.
- JSON (JavaScript Object Notation): Formát, který se používá pro přenos strukturovaných dat. Je čitelný pro člověka a snadno zpracovatelný strojem.
- XML (eXtensible Markup Language): Další formát pro přenos strukturovaných dat, který je hierarchický a rozšiřitelný.
- YAML (Yet Another Markup Language, později YAML Ain't Markup Language): Formát, který je snadno čitelný pro lidi a používá odsazení pro určení hierarchie dat.
- Binární data: Velice efektivní co se týče velikosti dat, avšak data pak nejsou čitelná a nelze je poupravovat ručně.

6.2 Formát

Jako vybraný formát byl po konzultacích zvolen JSON [1]. Vychází z programovacího jazyka JavaScript, je nezávislý na počítačové platformě a umí uchovávat datové typy: Číslo, Řetězec, Boolean, Array, Objekt - spojení hodnot a klíče. Aplikace si pomocí jednoho .json souboru předávají seznam akcí.

[1] Vlastnosti vychází z dokumentace (www.json.org)

6.3 Typy vytvářených objektů

Akce vytváří a poupravují objekty v Unity scéně. Tyto objekty vznikají z předdefinovaných předpisů, které mají aplikace čtyři. Prvním je objekt kreslení, který využívá LineRenderer komponentu a zobrazuje list úseček s danou tloušťkou a barvou. Pro tento objekt je nutné po dokončení vytvořit MeshCollider komponentu, abychom mohli detekovat kolize a objekt případně smazat. Dalšími objekty jsou pak úsečka, plocha a krychle. Všechny tyto definice mají svojí škálu a při tvorbě se jejich materiál změní na vybranou barvu.

6.4 Ukládané akce

Akce jsou čtyři[2] a všechny využívají jednu třídu Action v obou programech. Právě jednotná definice dovoluje snadný přenos a čtení dat na obou stranách v JSON formátu. Typy akcí:

- Vytvoření objektu - vytvoří se objekt podle vybrané definice s daným int ID na pozici Vector3 pos
- Vytvoření bodu kreslení - najde se objekt pro kreslení pod daným ID a pro něj se vytvoří a zařadí další bod na konec seznamu
- Změna transformace objektu - tedy pozice, rotace a škály
- Začátek přehrávání audioclipu (string audioClipName, int offset, double length)

6.5 Audio

Složka, kterou si programy přehrávají, musí také obsahovat audio ve formátu .mp3. Audio si zatím vyučující musí zařídit sám, společně s komunikací se žáky. Do budoucna je však ve virtuální třídě připraveno rozhraní pro nahrávání mikrofonu.

[2] sedm, pokud počítáme rozlišené typy vytvořených objektů.

Implementace editoru

7

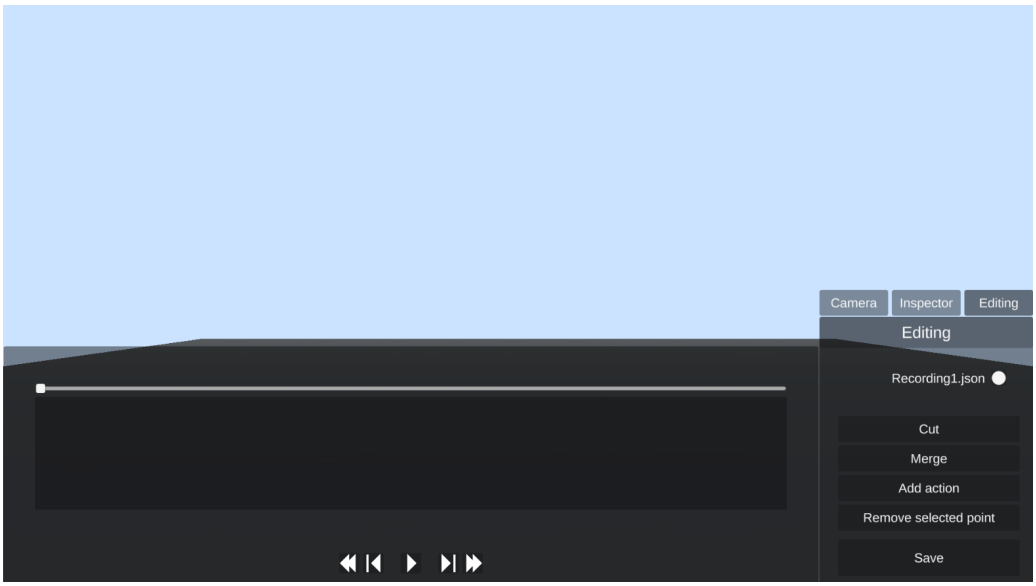
7.1 Architektura

Základními využitými datovými strukturami jsou hlavně slovník a listy akcí. Slovník se využívá pro rychlé vyhledání .mp3 souboru k dané akci. Listy pak pro jednoduchou práci s mnoha akcemi v jednom záznamu. Scéna editoru se skládá ze tří částí. První částí je prostředí, tedy plocha, na které se zobrazují objekty ze záznamu. Druhou částí je kamera a k ní přidružené uživatelské rozhraní s nástroji. Třetí část tvoří objekty obsahující skripta s chováním aplikace. Editor má skripta pro chování oddělené jako jedináčky, kteří se inicializují při zapnutí aplikace. Takto rozdělené jsou právě skripta, děděné od třídy `MonoBehaviour`, aby mohli být ve scéně jako komponenta objektu. Mezi tato skripta patří vše od jednotlivých manažerů pro nástroje po vstupní a výstupní chování aplikace. Ovládání kamery zařizuje `CameraInspector.cs`, který má odkazy na uživatelské rozhraní a detekuje, zda je kamera zamčena či odemčena a podle nastavení zařizuje chování herního objektu kamery. Hlavní je zde pohyb, který zařizuje změna pozice a rotace objektu. Zobrazování vlastností vybrané akce zařizuje `Inspector.cs`, má odkazy na textové pole, které při změně výběru bodu změní. Tento skript také zapíná a vypíná okna nástrojů podle interakce s uživatelem. `EditorManager.cs` se stará o automatické ukládání z editoru. Aktuálně je nastaven pro uložení každých šedesát vteřin od zapnutí aplikace, pokud nastane změna. Další velkou částí pro fungování aplikace je pak `Timeline.cs`, který v sobě ukládá list bodů, list vybraných bodů, nastavení a funkce pro generaci uživatelského rozhraní a zároveň také obstarává přehrání audia během editace. Tento skript pak využívá `RecordObjectCreator.cs`, aby vytvořil herní objekty podle záznamu přímo ve scéně. O vstup a výstup se pak stará `RecordManager.cs`, který si uchovává načtené soubory, cesty a audioklipy. Audioklipy jsou zde uloženy ve slovníku, kde klíčem je název souboru a obsahem je přeformátované audio do Unity `AudioClip.cs`.

7.2 Rozložení nástrojů

7.2.1 Popis aplikačního okna

Aplikace obsahuje dva panely pro nástroje. Prvním je panel pro časovou osu, dalším pak pro tři různé nástroje pro kameru, inspekci a editaci. Na obrázku 7.1 můžete vidět, jak vypadá aplikace před načtením záznamu k editaci. Po načtení pak vypadá aplikace jako na obrázku 7.2.



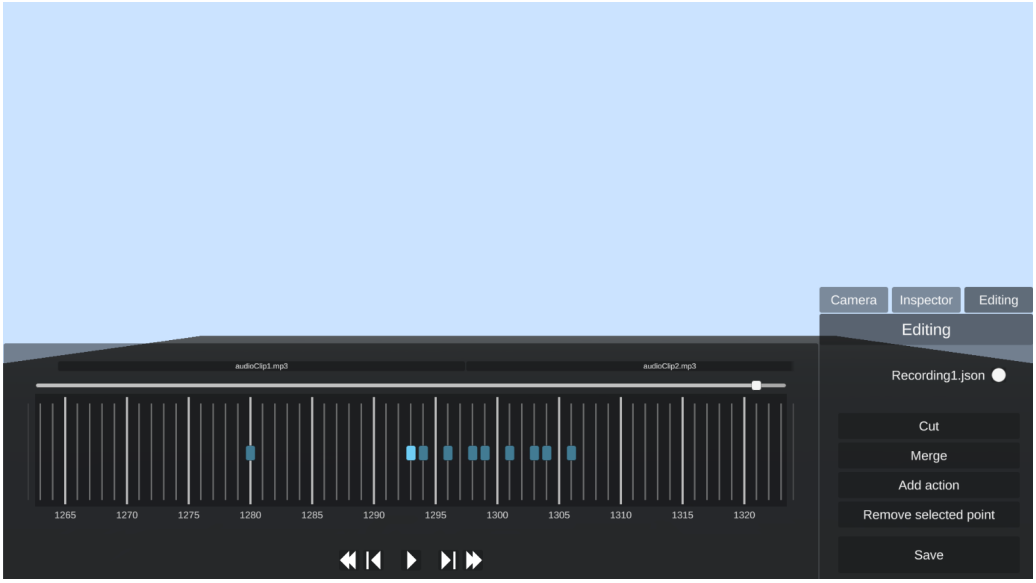
Obrázek 7.1: Virtual Reality Capture Editor (před načtením zdrojové složky)

7.2.2 Vstup pro aplikaci

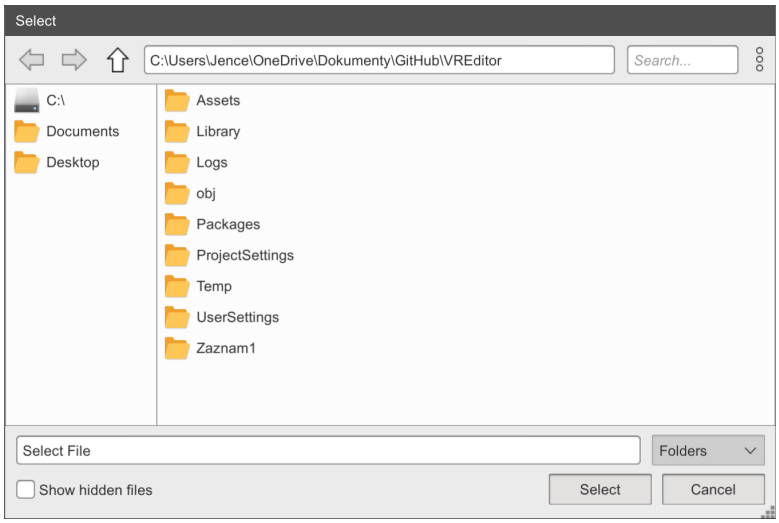
Vstup probíhá hned po zapnutí aplikace formou složkového formuláře, který můžete vidět na obrázku 7.3. Tento formulář je vytvořen za pomoci knihovny SimpleFileBrowser. Jeho cílem je získat cestu k cílovému adresáři, kde jsou uchována data k záznamu. Konkrétně se jedná o jeden .json soubor akcí a vícero zvukových záznamů ve formátu .mp3.

7.2.3 Uchování informací

Informace jsou uchovány přímo v datových strukturách aplikace, při přesunu dat mezi aplikacemi pak v souborech, avšak pro svižný chod aplikace všechny změny probíhají nejdřív interně a do souborů se zaznamenávají pouze při automatickém či cíleném exportu.



Obrázek 7.2: Virtual Reality Capture Editor (po načtení vstupních dat)

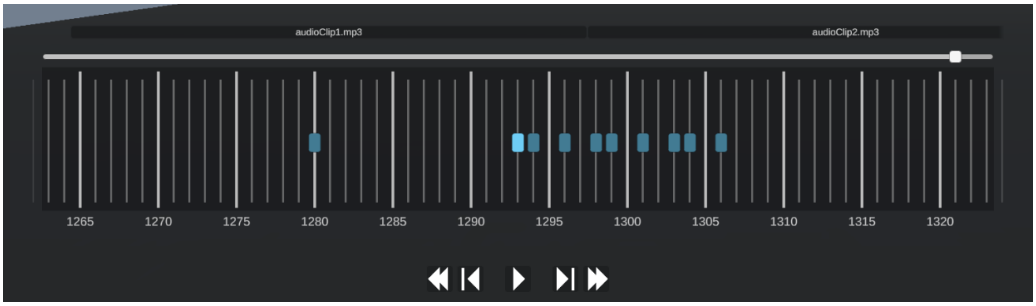


Obrázek 7.3: Formulář pro vstupní složku editoru

7.2.4 Panel časové osy

Časová osa je inspirována animačním editorem z Unity. Horní část časové osy ukazuje celý záznam a aktuální polohu v záznamu. Dolní část pak ukazuje jednotlivé snímky a akce k nim přiřazené. Každá čára představuje právě jeden snímek záznamu, tedy 1/60 vteřiny. Pro snadnější ovládání časové osy je zde umístěno pět tlačítek. Prostřední představuje zastavení a spuštění záznamu v editoru. Panel můžete vidět na obrázku 7.4 Tlačítko [«] přesune záznam na první bod akce ze začátku. Tlačítko [»] pak na první bod z konce. Tlačítka [|<] a [>|] zaměří další bod akce zleva, respektive

zprava.



Obrázek 7.4: Panel časové osy

7.2.5 Vstup a výstup

Vstup pro aplikaci je implementován pomocí knihovny SimpleFileBrowser. Ihned po zapnutí aplikace vyskočí formulář pro výběr vstupní složky. Z této složky se načte první nalezený .json soubor a všechny .mp3 soubory jako instance třídy `Audioclip`.

7.2.6 Ovládání kamery

Vzhledem k tomu, že záznam je trojrozměrný, je nutné mít možnost posunout či rotovat kameru. Transformaci kamery lze docílit dvěma způsoby. Prvním je zadání nových vlastností do textových polí. Druhým pak zamknutí a odblokování kamery a pohyb po scéně pomocí [`W, S, A, D`] a pravého tlačítka myši. Panel obsahuje také tlačítko pro vyresetování pozice a rotace na původní hodnoty. Ukázku můžete vidět vlevo na obrázku 7.5.

7.2.7 Inspektor

Inspektor je naprogramován tak, aby zobrazil informativní hodnoty vybraného bodu akce. Tyto vlastnosti jsou: typ, přiřazený snímek, identifikační číslo objektu, pozice, rotace, škála, jméno objektu. Ukázku můžete vidět uprostřed obrázku 7.5.

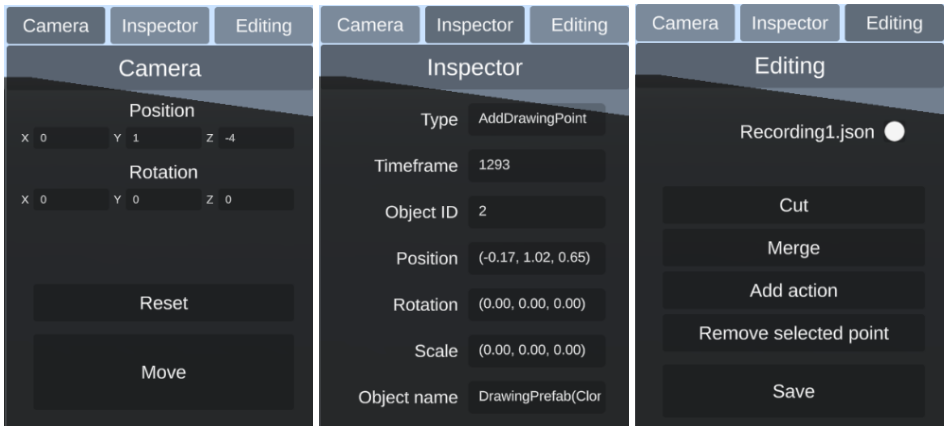
7.2.8 Změna aktuálního záznamu

V případě potřeby lze aktuální záznam změnit za jiný bez vypnutí a zapnutí aplikace a to zmáčknutím tlačítka napravo od názvu záznamu.

7.2.9 Funkce pro editaci

Přidání bodu proběhne přidáním uživatelského rozhraní pro bod akce na časovou osu a v pozadí programu pak přidáním do struktur s akcemi. Změna bodu pouze

mění data daného bodu v datech. Střih ukazuje červenou oblast přes časovou osu pro daný interval střihu, potvrzení probíhá přes tlačítko `Confirm`, střih odstraní všechny body jak vizuálně v aplikaci, tak v datových strukturách a zkrátí počáteční dobu pro všechny akce, které jsou za koncem střihového intervalu. Odstranění vybraného bodu pak znovu graficky upraví časovou osu a odstraní bod z datových struktur. Ukázky můžete vidět vpravo na obrázku 7.5.



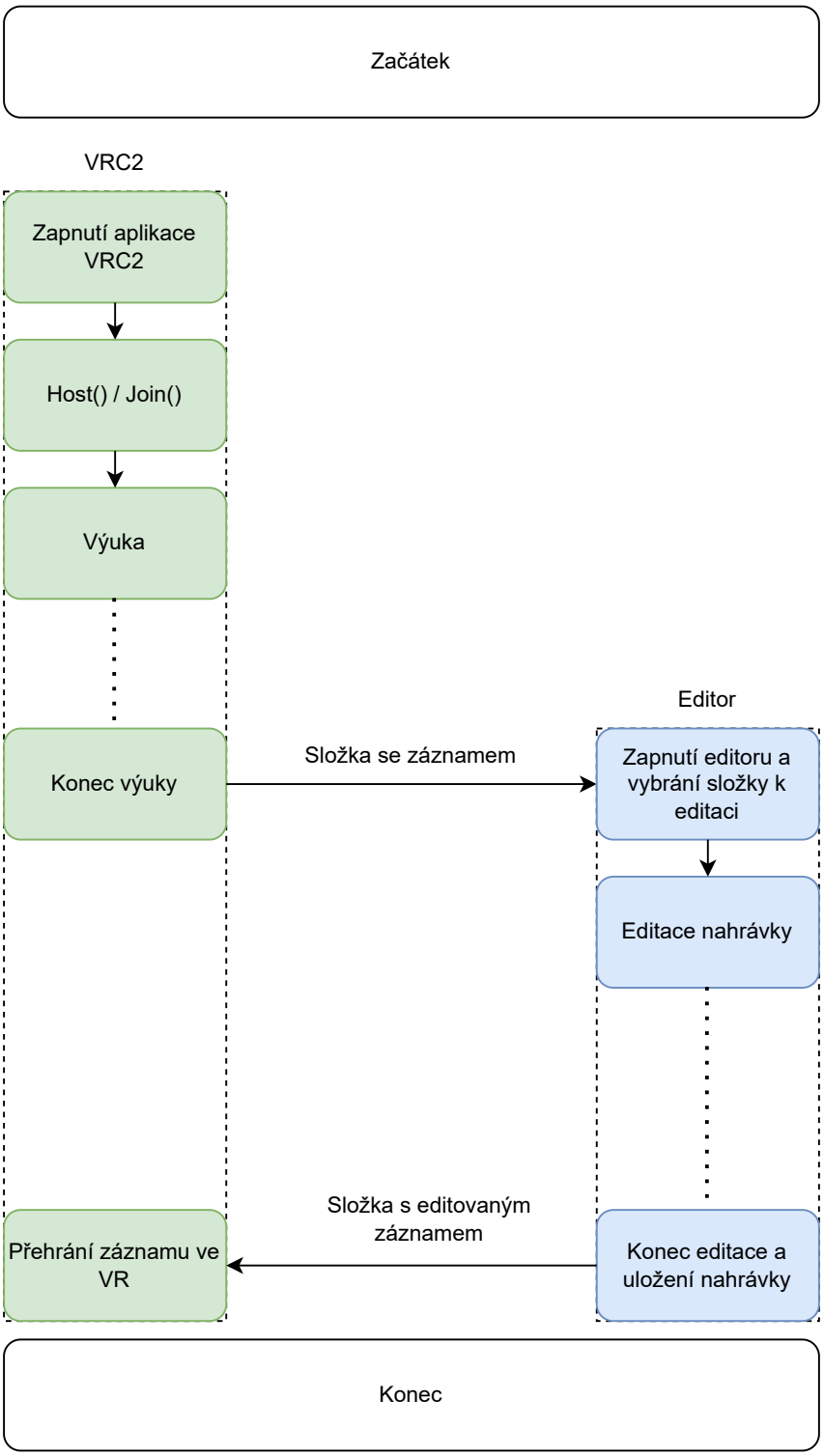
Obrázek 7.5: Uživatelské rozhraní pro editaci

7.3 Struktura aplikace VR Capture Editor

```
VRClassroom
├── Assets
│   ├── Materials
│   ├── Plugins
│   ├── Prefabs ..... Objekty pro UI, objekty pro akce
│   ├── Rendering ..... Nastavení pro světlo a render
│   ├── Scenes ..... Scény aplikace
│   ├── Scripts ..... Skripta pro chování aplikace
│   ├── TextMeshPro ..... Knihovna pro Textové UI
│   └── UI ..... Grafika pro aplikaci
├── Builds
├── Library
├── Logs
├── Packages
└── ProjectSettings
```

7.4 Diagram aktivit mezi aplikacemi

Diagram aktivit popisuje, jak by měl vypadat pracovní postup vyučujícího, který použije obě aplikace k vytvoření čistého záznamu z hodiny ve virtuální realitě. Nejdříve pracuje v nové aplikaci virtuální třídy, kde nahraje hrubý záznam společně s výukou. Záznam pak doplní o .mp3 audio soubory. A pomocí editoru pak záznam vyčistí a připraví k přehrání zpátky ve virtuální třídě.


Začátek
VRC2
Zapnutí aplikace VRC2
Host() / Join()
Výuka
Konec výuky
Složka se záznamem
Editor
Zapnutí editoru a vybrání složky k editaci
Editace nahrávky
Složka s editovaným záznamem
Přehrání záznamu ve VR
Konec editace a uložení nahrávky
Konec

Testování a dokumentace

8

8.1 Testování VRC2

Testování VRC2 proběhlo na dni otevřených dveří FAV v roce 2024, kde aplikaci testovalo celkem 12 subjektů s pozitivní odezvou. Hlavní kritikou byla složitost přepínání mezi nástroji pro tvorbu. Naopak si chválili možnosti, které aplikace nabízí pro kreativní tvorbu. Pozitivní zpětná vazba byla také v ohledu na intuitivnost jednotlivých nástrojů. Dokázali vytvořit jak 3D schémata z úseček a ploch, tak kreativní obrázky pomocí malování v prostoru.

Obrázek 8.1: Testování na dni otevřených dveří 2024

8.2 Testování editoru

Testování editoru proběhlo formou krátké instruktáže a pak vyzkoušením aplikace editoru. Tabulka 8.1 pak ukazuje základní funkčnost a 8.2 výsledky z formulářů pro editor. Aplikaci otestovaly čtyři subjekty.

- Zapnutí aplikace.
- Vložení testovacího scénáře - tvorba 3D spirály.
- Ozkoušení nástrojů pro kameru a inspekci bodů akcí.
- Export.
- Vyplnění formuláře.

Testovací scénář povrchně testoval aplikaci akcemi. Jednoduchost aplikace z testování vyplynula õ0 . Kompletní výpis z testování editoru je přiložen v přílohách.

Aplikace šla spustit	Aplikace byla stabilní	Aplikace běžela plynule
Ano 4	Ano 4	Ano 4
Ne 0	Ne 0	Ne 0

Tabulka 8.1: Testování základní funkčnosti aplikace (počty)

Nástroj pro kameru	Nástroj inspektoru	Nástroj pro editaci
4.5 / 5	4.75 / 5	5 / 5

Tabulka 8.2: Testování jednoduchosti použití nástrojů (průměry)

8.3 Dokumentace a logování

Jako dokumentace k návrhu obou aplikací slouží jak tato bakalářská práce, tak návodné video s příkladem využití. Skripta aplikací jsou z většiny okomentována standardem pro jazyk C# a obsahují zprávy pro logování. Logování bylo otestováno přes aplikaci Android LogCat, kterou lze otevřít přímo uvnitř Unity.

Zdrojový kód 8.1: Příklad komentářů a logování

```
1     /// <summary>
2     /// Changes color for objects creation, main porpuse of
3     /// this script is to change the following:
4     /// PlayerPrefs.SetFloat("ColorR", currentColor.r);
5     /// PlayerPrefs.SetFloat("ColorG", currentColor.g);
6     /// PlayerPrefs.SetFloat("ColorB", currentColor.b);
7     /// Uses ControllerTool interface.
8     /// </summary>
9     public class ColorTool : MonoBehaviour, ControllerTool
10    {
11      ...
12      Debug.Log("Color changed in playerprefs.");
13      ...
14    }
15
```

Omezení a možné rozšíření VRC2

9

9.1 VRC2

Aplikace nepodporuje vložení trojúhelníkových sítí z důvodů nutného rozšíření síťové vrstvy, avšak většina systémů je na to připravená. Aplikace při běhu s více uživateli ukazuje signálové symboly a ukazovátko, ale tyto akce nejsou uloženy pro záznam. Rozšiřitelnost je zde založená na modulárnosti řešení nástrojů a funkcí, ať už se jedná o nová primitiva nebo nový panel, který bude vkládat právě zmíněné trojúhelníkové sítě. Doporučeným postupem je nejdříve vybrat, jestli funkce bude přímo v ovládání pomocí ovladače či implementována přes panely s uživatelským rozhraním. Během konzultací vznikly i myšlenky ohledně možnosti řezů objektů či kreslení na textury vytvořených objektů. Panely vycházejí jako jednodušší možnost a jejich implementace je o něco jednodušší než vložení nového nástroje. Po přidání nástroje se musí také nástroj vložit do listů nástrojů u skriptů uživatele. Nutnou podmínkou je také implementace připraveného rozhraní. Co se týče síťové vrstvy, je zde několik nedořešených situací. Například, pokud uživatel odejde, nezmění se vlastnictví objektů a objekty nezmizí. Řešení pro tyto situace lze v budoucnu implementovat listem pro každého uživatele a zavoláním metody
při odpojení. Standardním postupem by bylo předání kompetencí a vlastnictví objektů. Dalším možným rozšířením je změna prostředí (viz 5.5.6) nebo rozlišení mezi statickým a mixovaným světlem, které by při úpravě scén dovolilo generovat dynamické stíny pro uživatelem vytvořené objekty.

9.2 Editoru

Editor zatím nemá podporu pro vybrání a editaci vícero bodů najednou. Funkcionalita je v kódu připravena, ale pro plné nasazení je nutné vyřešit vícero problémů. Editor také neudržuje žurnál editací a uživatel tedy nemůže udělat krok zpět či napřed. Možná symbolická označení typů akcí přímo v časové ose. Body v editoru neukazují na první pohled svůj typ. Jeho vyčtení je možné až po vícero kliknutí přes inspektor akcí. Další možností pro rozšíření je nástroj, který by umožnil stříhat audio nebo ho ručně doplnit nahráváním mikrofonu. Vyučující by si například všiml, že se v určité části přeřekl a mohl by daný audioklip smazat a namluvit přímo pomocí nástroje v editoru. Další částí, na které by se dalo zapracovat je zlepšení UX[1] například více interaktivním a responzivním prostředím.

Zdrojový kód 9.1: Příklad zakomentované a připravené struktury pro multiselect

```
1     public TimelinePointData data;
2     /// <summary>
3     /// On TimelinePoint being clicked.
4     /// </summary>
5     public void OnClick()
6     {
7       Debug.Log("Clicked on Timeline point");
8       Timeline.Instance.SetCurrentPoint(this);
9       // if (Input.GetKey(KeyCode.LeftControl) || Input.
      GetKey(KeyCode.RightControl))
10      // {
11      //   Timeline.Instance.AddCurrentPoint(this);
12      // }
13      // else
14      // {
15      //   Timeline.Instance.SetCurrentPoint(this);
16      // }
17
```

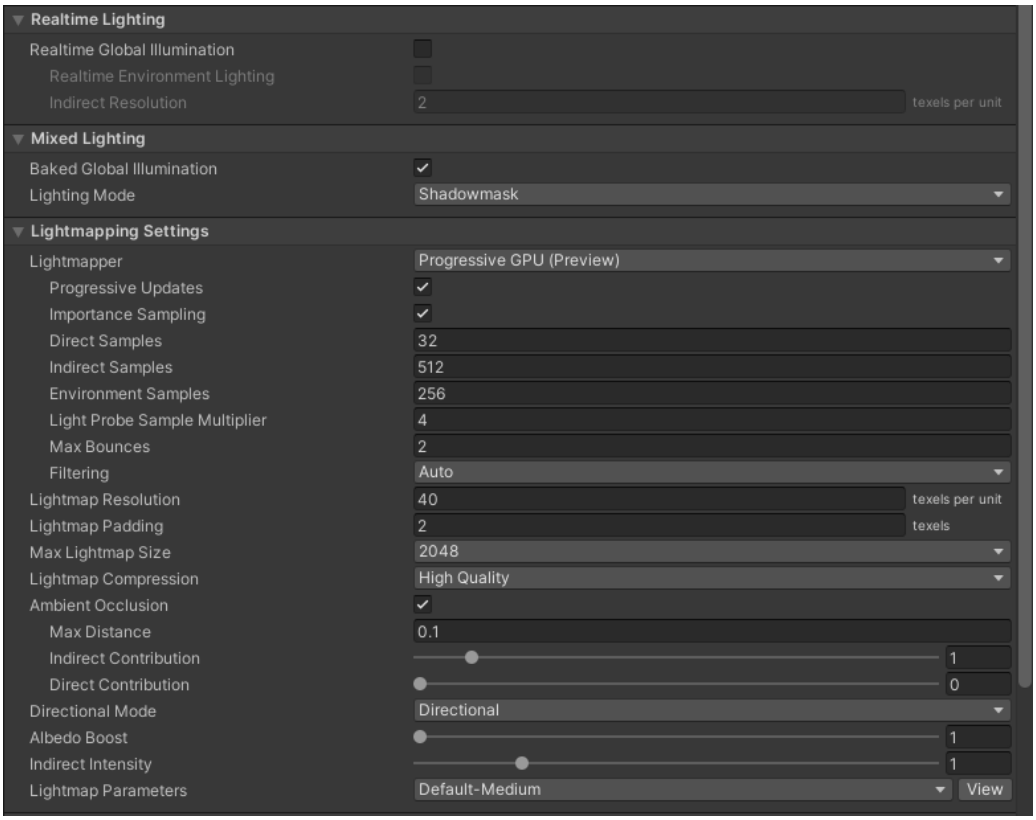
[1] UX = user experience = zkušenost uživatele

Optimalizace 10

Během vývoje se vyskytlo vícero problémů týkajících se výkonu aplikací. Prvním z nich byl problém vykreslování mnoha objektů ve VRC2, tento problém byl vyřešen metodou `oclusion culling` a změněním osvětlení scény. `Occlusion culling` šetří výkon aplikace vynecháním objektů mimo zorné pole kamer či objektů, které jsou schované za neprůhledným objektem.

Statické světlo funguje na principu definování scény, která se nemění. Dále jsou nutná světla, která produkují paprsky a následně výpočet odrazů světelných paprsků pro mapy se světlem. Tyto mapy pak změní jas objektů v závislosti na odrazech vůči okolí. Toto světlo nelze použít na vkládané objekty. Tedy je aplikováno jen na prostředí, které aplikace poskytuje při zapnutí. Všechny vytvořené objekty mají buďto původní světlost, nebo obsahují materiál, který na světlo nereaguje. Vypočtení tohoto světla probíhá pomocí vysílání paprsků ze zdroje a postupným propočítáváním odrazů a dopadů. Nastavení výpočtů pro světlo se dá změnit přímo v projektu aplikace, je ale vždy nutné mapy po změně přepočítat. Aktuální nastavení je zobrazeno na obrázku 10.1. Dalším problémem bylo zobrazování uživatelského rozhraní na časové ose editoru. Problémem bylo zobrazování mnoha průhledných objektů přes sebe. Jako řešení pak stačilo přidat masku, a tím zredukovat počet bodů a hran na obrazovce, změnit typy materiálů, některé nechat průhledné, u jiných mít hodnotu alfa kanálu vždy rovnou 1.

Nakonec aplikace pro virtuální realitu dosahuje více než standardních 90 snímků za sekundu, editor běží plynule a jen některé akce pro velké záznamy způsobí čekání na odezvu. Při přehrávání se akce přehrávají za sebou bez nutnosti opakování předešlých akcí. Pokud se však hrubou silou změní čas přehrávání, musí se projít a aplikovat všechny akce od počátku záznamu do tohoto bodu.



Obrázek 10.1: Nastavení světla pro VRC2 v Unity

Závěr 11

Během vypracování bakalářské práce jsem provedl analýzu současného stavu technologií a konkurenčních aplikací týkajících se výuky ve virtuálním prostoru a možností editačního softwaru pro trojrozměrný záznam. Tato analýza poskytla cenné poznatky o aktuálním stavu technologií v oblasti vzdělávání a editace virtuální reality, které byly klíčové pro návrh a vývoj nových aplikací.

Po analýze následoval návrh změn stávajícího softwaru, který zahrnoval přepracování aplikace virtuální třídy do modernější podoby s využitím nejlepších praktik a nových knihoven prostředí Unity. Tato aktualizace byla provedena s ohledem na potřeby uživatelů a připravila půdu pro další rozvoj a inovace. Aplikace pro virtuální třídu byla úspěšně otestována na dni otevřených dveří FAV v roce 2024, což poskytlo užitečnou zpětnou vazbu od uživatelů a přispělo k dalšímu zdokonalení softwaru. Na základě této zpětné vazby byl navržen a vyvinut také editační software, který umožňuje úpravu a vylepšení záznamů vytvořených ve virtuální třídě.

Součástí práce bylo také stanovení pracovního postupu a architektury pro práci s daty, což zajistilo efektivní a strukturovaný vývoj aplikací. Výsledkem této práce jsou dvě otestované aplikace, které jsou plánovány k využití na Západočeské univerzitě a na kterých se bude nadále pracovat a budou se rozvíjet v následujících letech.

Obě aplikace byly pečlivě optimalizovány, aby zajistily plynulý provoz a hladký běh. Navíc bylo vytvořeno instruktážní video, které slouží jako užitečný nástroj pro uživatele k seznámení se s funkcemi a možnostmi aplikací.

I přesto, že obě aplikace mají svá omezení, představují důležitý krok směrem k lepšímu využití virtuální reality ve vzdělávání a poskytují solidní základ pro budoucí projekty na Západočeské univerzitě.

11.1 Obsah přílohy

V přílohách práce najdete celkem 4 přílohy.

11.1.1 Složka se soubory pro testování editoru

Testovací soubor, použitý pro testování editoru. Obsahuje akce, které v prostoru vytváří spirálu. Složka také obsahuje jeden .mp3 soubor pro otestování zvukových funkcí editoru.

11.1.2 Výsledky testování

Vygenerované .pdf stránky z testování editoru.

11.1.3 Exporty obou aplikací

Exportované složky s poslední verzí obou aplikací, obsahující .exe spoušťecí soubory. Následující exporty mají tyto struktury:

```
VRClassroomBuild.zip
├── MonoBleedingEdge
├── VRClassroom_Data
├── UnityCrashHandler64.exe
├── UnityPlayer.dll
└── VRClassroom.exe ....... Spouštěcí soubor pro aplikaci virtuální třídy
EditorBuild.zip
├── MonoBleedingEdge
├── Editor_Data
├── UnityCrashHandler64.exe
├── UnityPlayer.dll
└── Editor.exe ..................Spouštěcí soubor pro aplikaci editoru
```

Bibliografie

[God18] GODOT. *Godot Game Engine* [online]. 2014-4-18. [cit. 2024-04-28]. Dostupné z: https://godotengine.org/.

[HVV21] HÁCHA, Filip; VANECEK, Petr; VÁŠA, Libor. A Virtual Reality Platform for Immersive Education in Computer Graphics. In: SOUSA SANTOS, Beatriz; DOMIK, Gitta (ed.). *Eurographics 2021 - Education Papers*. The Eurographics Association, 2021. ISBN 978-3-03868-132-8. ISSN 1017-4656. Dostupné z DOI: 10.2312/eged.20211001.

[Hel92] HELSEL, Sandra. Virtual Reality and Education. *Educational Technology* [online]. 1992, roč. 32, č. 5, s. 38–42 [cit. 2024-04-13]. ISSN 00131962. Dostupné z: http://www.jstor.org/stable/44425644.

[Mic22] MICROSOFT. *Průvodce jazykem C#* [online]. 2020-1-22. [cit. 2024-04-26]. Dostupné z: https://learn.microsoft.com/cs-cz/dotnet/csharp/.

[Poó22] POÓR, Vítek. Zaznamenávání akcí ve virtuální realitě pro účely výuky. 2022. Dostupné také z: portal.zcu.cz.

[Uni30] UNITY. *Dokumentace Unity* [online]. 2023-5-30. [cit. 2024-04-24]. Dostupné z: https://docs.unity.com/.

[Uni05] UNITY. *Unity Asset Store* [online]. 2007-03-05. [cit. 2024-04-26]. Dostupné z: assetstore.unity.com.

[Unr23] UNREAL. *Unreal Game Engine* [online]. 2024-04-23. [cit. 2024-04-29]. Dostupné z: https://docs.unity.com/.

[Val12] VALVE, Corporation. *Obchod Steam*. 2003-9-12. Dostupné také z: https://store.steampowered.com/.

Seznam obrázků

Seznam tabulek

Seznam výpisů