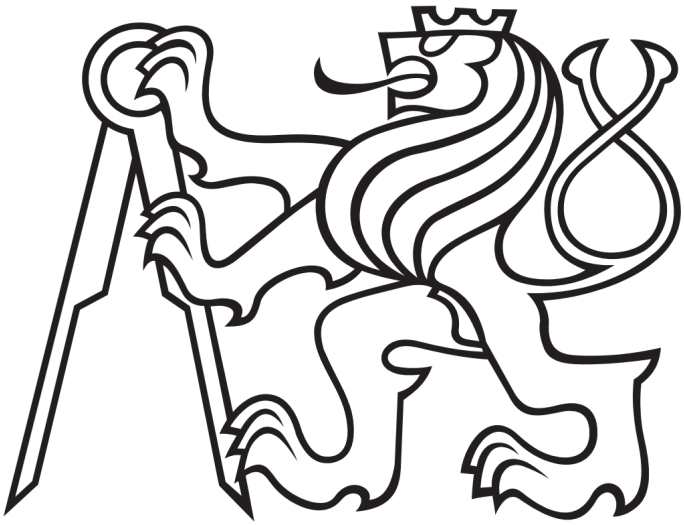
České vysoké učení technické v Praze
Fakulta elektrotechnická
Katedra počítačů



Bakalářská práce

Rozšířená realita (AR) v React Native

Vypracoval: Nikita Grigoryev

Vedoucí práce: Ing. David Sedláček, Ph.D.

Studijní program: Softwarové inženýrství a technologie

13.08.2021

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Grigoryev** Jméno: **Nikita** Osobní číslo: **465851**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačů**

Studijní program: **Softwarové inženýrství a technologie**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Rozšířená realita (AR) v React Native

Název bakalářské práce anglicky:

Augmented Reality with React Native

Pokyny pro vypracování:

Seznamte se s aktuálním stavem projektu EduARd (rozšířená realita pro školy) a s rozšířenou realitou (AR). Proveďte rešerši aplikací využívajících AR ve venkovním prostředí. Prostudujte knihovny pro rozšířenou realitu postavené nad frameworkem React Native.
Navrhněte a implementujte rozšíření stávající klientské aplikace v React Native aplikace portálu EduARd o podporu rozšířené reality v návaznosti na data, která poskytuje server projektu EduARd (formáty X3D, json, rastrové obrázky a video soubory). Implementujte jednosměrnou synchronizaci dat se serverem (stahování dat) a sestavení scény dle staženého scénáře (.json předpis). Implementujte načítání vybraných uzlů formátu X3D (dle dohody s vedoucím práce).
Postupujte dle metodiky UCD (User Center Design) pro návrh a testování uživatelského rozhraní s ohledem na dostupnost cílové skupiny.

Seznam doporučené literatury:

[1] Moderní počítačová grafika: Bedřich Beneš, Jiří Sochor, Petr Felkel, Jiří Žára, 2005, Computer press
[2] Augmented Reality: Principles and Practice: Dieter Schmalstieg, Tobias Höllerer, 2016, Addison Wesley
[3] Practical Augmented Reality: Steve Aukstakalnis, 2017, Addison Wesley
[4] T. Lowdermilk, User-Centered Design, O'Reilly Media, 2013
[5] X3D: Web 3D Consortium, [online], http://www.web3d.org/
[6] EduARd: David Sedláček, [online], https://eduard.fel.cvut.cz/

Jméno a pracoviště vedoucí(ho) bakalářské práce:

Ing. David Sedláček, Ph.D., katedra počítačové grafiky a interakce FEL

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

Datum zadání bakalářské práce: **11.02.2021** Termín odevzdání bakalářské práce: **13.08.2021**

Platnost zadání bakalářské práce: **30.09.2022**

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

III. PŘEVZETÍ ZADÁNÍ

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

. _____
Datum převzetí zadání

_____
Podpis studenta

Prohlášení

Prohlašuji, že jsem svou bakalářskou práci na téma „Rozšířená realita (AR) v React Native“ vypracoval samostatně s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu literatury na konci práce.

V Praze ……. dne …….

(podpis autora)

Poděkování

Tímto bych chtěl poděkovat Ing. Davidu Sedláčkovi, Ph.D. za vedení mé práce a za cenné informace a rady poskytnuté během práce na tomto projektu. Dále bych chtěl poděkovat své rodině a svým kamarádům za morální podporu. Zároveň bych také chtěl poděkovat své kamarádce Ksenii, která mi podporovala v mem studiu na vysoké škole. Nakonec bych chtěl podekovat své sestře Ekaterině a její manželu Patriku za pomoc s opravou češtiny.

Abstrakt

Tato bakalářská práce popisuje implementaci rozšíření do stávající mobilní aplikace o podporu rozšířené reality využívající GPS lokalizaci v React-Native.

Aplikace načítá 3D scény v rámci scénáře vytvořeného editorem scén. Všechna potřebná data pro správné zobrazení scény aplikace stahuje ze serveru EduARd. Aplikace byla vyvíjena v rámci projektu EduARd.

Cílovou skupinou této aplikace jsou žáci základních a středních škol a dále i pedagogičtí pracovnici. Z tohoto důvodu bylo rozšíření implementováno s důrazem na jednoduchost a srozumitelnost. Na základě provedené rešerše byl pro implementaci zvolen framework ViroReact (verze 2.17.0), který je postaven na React Native. Aplikace je kompatibilní se zařízeními s operačním systémem Android 6.0 a vyšší.

Klíčova slova: rozšířená realita, AR, GPS, GPS aplikace, Georeferencovaná, mobilní aplikace, ViroReact, React Native, Android.

Abstract

This bachelor thesis describes the implementation of an extension to the existing mobile application to support augmented reality using GPS localization in React-Native.

The application loads 3D scenes within a scenario created by the scene editor. The application downloads all the necessary data for the correct display of the scene from the EduARd server. The application was developed within the EduARd project.

The target group of this application are primary and secondary school pupils as well as pedagogical staff. For this reason, the extension has been implemented with an emphasis on simplicity and comprehensibility. Based on the search, the ViroReact framework (version 2.17.0) was chosen for implementation, which is based on React Native. The application is compatible with devices running Android 6.0 and higher.

Keywords: augmented reality, AR, GPS, GPS application, Georeferenced, mobile application, ViroReact, React Native, Android

Title translation: Augmented Reality with React Native

Obsah

Seznam obrázků

Seznam tabulek

Seznam vzorců

1 Úvod

1.1 Motivace

Již několik desetiletí zaznamenáváme neuvěřitelné tempo vývoje výpočetní techniky. Velký narůst ve výkonu za posledních několik desetiletí ovlivnil nejen způsoby práce u omezené skupiny lidí, jako jsou vědci, armáda nebo finanční instituce, ale také i nás. A to natolik, že v nynější době si nemůžeme představit naše životy bez počítačů, mobilních telefonů a internetu. Někteří lidé se obávají, že za dalších několik desetiletí člověk ztratí propojení s realitou a přesune do světa virtuálního. Tento fenomén můžeme pozorovat už dnes na některých lidech, u kterých se projevuje závislost na online hrách. Na druhou stranu však takové tempo vývoje techniky může vést naší civilizaci na následující stupeň evoluce, tedy propojení našeho těla s výpočetní technikou. Už v nynější době můžeme pozorovat ojedinělé první kroky v tom to směru.
Například již existující prototypy čipu, které nahrazují poškozené nervy a tím umožnují postiženým lidem vrátit se k normálnímu životu. Jednou z takových technologií je právě rozšířená realita.

Rozšířená realita se v posledním desetiletí velmi rozvíjela a stala se z neobvyklé věci za obvyklou. Málo kdo teď vnímá tuto technologii jako nějakou vědeckou high-tech technologii. Přes velký pokrok v této technologii nemůžeme říct, že se v tomto udála podobná revoluce jako u chytrých telefonů. Hlavní důvod, proč rozšířená realita v nynější době není tolik žádanou je, že způsob její používání není tak komfortní a přínosný. Většina nejpopulárnějších mobilních aplikací s touto technologií jsou pouze hry, a navíc k tomu založeny na stejném principu. Nejznámější hra tohoto žánru je *Pokémon Go*.

Takovou situaci může změnit příchod nových způsobů vnímání virtuálního obsahu, a jsou jím chytré brýle a čočky. Ty mnohem více usnadní použití rozšířené reality v běžném životě a přinesou nové způsoby interakce s rozšířenou realitou. Nejvíce se k takové podobě užívání rozšířené reality přibližuje helma *Microsoft Hololens*[1].

Tento projekt je součástí projektu *EduARd* [2]. Který má za cíl vytvořit systém pro tvorbu virtuálních a naučných stezek pro žáky základních a středních škol. *EduARd* umožní učitelům tvorbu učebních materiálů se zajímavým obsahem, například úlohami ve formě 3D scén. Tyto scény si žáci budou moci stáhnout do svých zařízení a spustit je v lokalitách definovaných GPS [3] souřadnicemi. Za hlavní motivací projektu je považováno zvýšení zájmu u dětí a mládeže o vědy a učební obory, zvýšení atraktivity školního materiálu, seznámení dětí a učitelů s novými technologiemi pro výuku a ukázat, že používání techniky, jako tablety a telefony může být užitečné i pro výuku.

[1] Microsoft Hololens 2 – převzato z [27].
[2] Projekt EduARd - převzato z [28].
[3] Viz sekce *2.4 Globální polohový systém.*

1.2 Cíle projektu a cílová skupina

Cílem této bakalářské práce je rozšíření stávajících mobilních aplikací o možnost prohlédnutí rozšířené reality. Aplikace bude schopna načítat scény v rámci scénáře vytvořeného editorem 3D scén, který implementován v rámci ekosystému *EduARd* [4] a zobrazovat tyto scény podle polohy uživatele. Veškeré potřebné soubory pro scény aplikace umožní stahovat přímo do mobilního zařízení, tím umožní uživateli používat mobilní aplikaci bez připojení k internetu, bude potřeba mít jen zapnutou GPS.

Hlavní cílovou skupinou jsou žáci základních a středních, tedy osoby přibližně od 6 do 18 let. Bude také potřeba brát v úvahu učitele, které také budou používat danou aplikací, hlavně z důvodu, aby byli schopni si poradit s případnými komplikacemi v mladších skupinách žaků. Z těchto důvodů je hlavním požadavkem na výslednou mobilní aplikaci intuitivní uživatelské rozhraní, které umožní rychlé zorientování a jednoduché používání aplikace.

1.3 Struktura práce

Práce je členěna do kapitol, sekcí a podsekcí. Každá kapitola se bude zabývat určitou fází vývoje projektu. Začátek práce je věnován terminologii, tj. vysvětlením důležitých pojmů vyskytujících se v textu. Dále se práce zabývá analýzou již existující aplikace *EduARd* [5], náhledem do již existujících projektů zabývajících se podobnou problematikou a rešerši dostupných nástrojů pro implementaci aplikace s podporou o rozšířenou realitu. Poté je uveden návrh, kde je detailněji popsán vybraný nástroj pro implementaci rozšíření a jsou sepsány požadavky na systém. V následující kapitole o implementaci jsou poté popsány metody realizace projektu. Poslední kapitoly se zabývají nejprve testováním aplikace, a nakonec závěrem shrnujícím vývoj a průběh práce, její kvality a nedostatky a budoucí vývoj.

[4] Webová aplikace pro tvorbu scén - převzato z [11].
[5] Mobilní aplikace pro zobrazení naučných stezek - https://gitlab.fel.cvut.cz/eduard/react_client.

2 Terminologie

Tato kapitola se bude zabývat definicemi termínů klíčových pro tuto práci. Budou zde vysvětleny pojmy jako, rozšířená realita, virtuální realita a globální polohový systém.

2.1 Rozšířená realita

Z článku Ronalda Azuma vydaném v roce 1997 pojem rozšířené reality popsán jako technologie umožňující uživateli vidět virtuální objekty v reálném světe, a to v ideálním případě tak, aby to vypadalo, že spolu tyto objekty koexistují ve stejném čase a prostoru [1].

Pro lepší znázornění Azuma vymezil tří základní charakteristiky rozšíření reality:

• Rozšířená realita je kombinací reálného a virtuálního světa.

• Rozšířená realita je interaktivní v reálném čase.

• Rozšířená realita je zaznamenána v 3D.

Pro zprostředkování rozšíření reality je nutné mít zařízení či kombinaci několika zařízení, které nám umožní sjednotit reálný a virtuální svět. Rozšířená realita je především vázaná na fotoaparát, kterým je snímaná realita, na výpočetním zařízení, které umožní provádět nutné výpočetní operace pro práce se speciálním softwarem a v neposlední řadě na zobrazovacím zařízení, pomocí kterého uživatel bude vnímat rozšířenou realitu.

Existují 3 různé druhy AR podle trekování. Jsou to Marker AR, Marker-less a Location-based AR.

2.1.1 Marker AR

Marker je vizuální prvek, například vytištěný na papíře. Aplikace pro rozšířenou realitu jej rozpozná a vyvolá patřičnou akci, například zobrazí virtuální model obrázku nebo hypertextové informace k expozici.

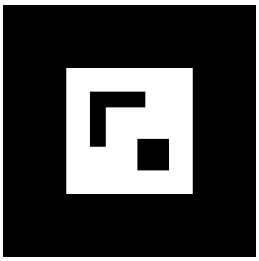
Druhy markerů:

- Čárové kódy a QR kódy. Předchůdce AR markeru. Viz obrázek *Obrázek 1*. QR kód.

Obrázek 1. QR kód [6]

[6] Převzato z https://www.napocitaci.cz/images/onlibpc/16_07qr_01.png

- Tištěné černobílé markery. V děsní době nejčastěji používané. Je to lehce rozpoznatelný i snadno generovaný obrázek. Marker má pevné dane okraje což zvětšuje přesnost rozpoznávaní. Viz obrázek *Obrázek 2. Tištěný* černobílý marker.



Obrázek 2. Tištěný černobílý marker[7]

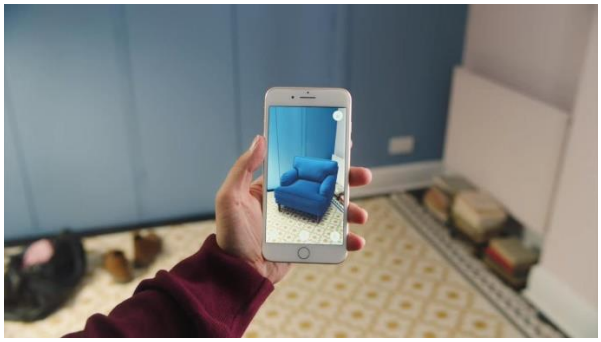
- Tištěné přirozené markery. Mohou vycházet z jakéhokoli obrázku.
- Reálné markery. Mohou vycházet z jakéhokoli objektu který navíc se může pohybovat.

2.1.2 Marker-less

Principem marker-less AR je skenování okolního prostředí. Tomu se říká *SLAM* – simultánní umístění a mapování. Pro načtení objektu není nutné mít žádný marker ani určovat polohu zařízení pomoci GPS.

Tyto aplikace většinou požádají uživatele najit rovný povrch pro umístění prvků AR. Takovými povrchy můžou být například stůl nebo podlaha.

Příkladem marker-less AR je aplikace IKEA Place viz obrázek *Ikea Place App*



Obrázek 3. Ikea Place App.

2.1.3 Location-based AR

Tento druh AR nepotřebuje žádné speciální značky (markery) pro rozpoznávaní, kam umístit virtuální objekt ve skutečném prostředí. Tyto aplikace používají k určení polohy a zjištění směru natočení objektivu kamery, GPS a digitální kompas, což je kombinace, která funguje velmi přesně. Aplikace poté odesílají dotazy senzorům zařízení a rozhodují, zda by měly na základě získaných dat přidat virtuální objekt [2]. Příklad *Location based AR* je na obrázku *Location Based AR*.

[7] Převzato z https://blog.arealidea.ru/images/articles/pcvision%20(2).png

Obrázek 4. Location Based AR[8]

2.2 Virtuální realita

Pojem virtuální reality je do dnes obtížné definovatelný. Existuje hodně pohledů na to, co ve skutečnosti VR je. Osobně jsem si oblíbil definici Jarona Laniere, který se považuje za autora pojmu VR. Podle něj virtuální realita je „*Počítačem vytvořené interaktivní trojrozměrné prostřední, do něhož se člověk totálně ponoří.*“. [3]

Většina současných prostředí virtuální reality jsou především vizuální zážitky, zobrazující se buď na monitoru počítače nebo přes stereoskopické zobrazení. Některé simulace dokážou vyvolat i další smyslové vjemy, jako zvuky, chutě, vůně nebo dotyky.

Tuto virtuální “skutečnost” generuje výpočetní technika, která pokrývá 360 stupňový obraz. Vždy se jedná o VR brýle, které si uživatel nasadí na hlavu a následně díky ovládacímu zařízení interaguje se svým prostředím. V některých případech se využívají senzory, které detekují váš pohyb a okolí ve VR je přizpůsobené právě vašemu reálnému prostředí.

Virtuální realita nachází svá využití v mnoha odvětvích. Například v armádě, ve vzdělávání, sportu, ve zdravotnictví, turistice a dalších.

Technologie používané při tvorbě virtuální reality:

- Stereoskopický displej - 3D displej, který poskytuje binokulární paralaxu.
- Hardware na snímání pohybu. Gyroskopy, akcelerometry a jiné nenákladné komponenty jsou používány k tvorbě VR headsetů, aby snímaly pohyby těla a natočení hlavy a data z těchto pohybů poslali do aplikace, která je poté zpracovává.
- Vstupní zařízení na snímání pohybů těla či rukou, nové gamepady, klávesnice či myši.
- Desktopové a mobilní platformy (počítač, operační systém, software, a tak dal).

[8] Převzato z https://www.cumanagement.com/sites/default/files/inline-images/5-23-18_2_0.jpg

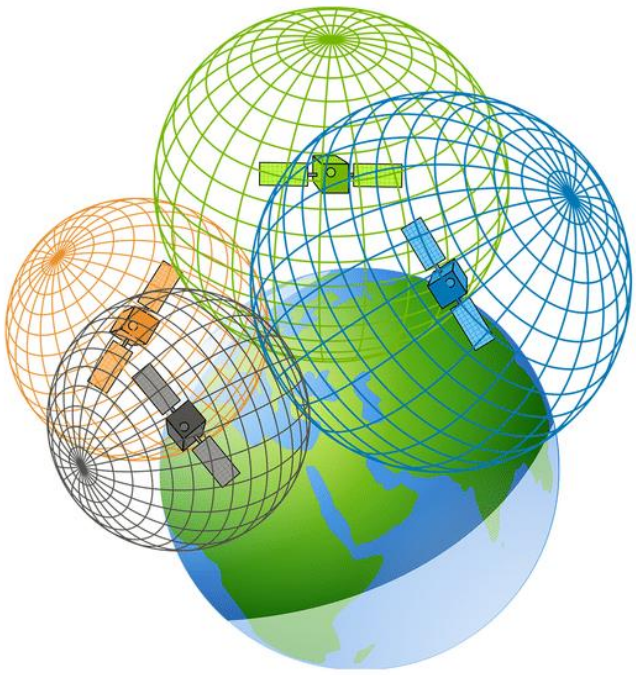
2.3 Rozdíly mezi VR a AR

Virtuální realita a rozšířená realita dosahují dvou velmi odlišných věcí dvěma velmi odlišnými způsoby. VR nahrazuje životní prostředí reálného světa simulovaným a přenáší vás někam jinam zatím co AR přispívá k realitě a promítá informace nad to, co už vidíte.

2.4 Globální polohový systém

Globální polohový systém (*Global Positioning System*) neboli GPS je satelitní systém složený z 24 satelitů umístěných americkým ministerstvem obrany na oběžné dráze Země ve výšce přibližně 20 350 km a pozemními sledovacími stanicemi spojenými do společné sítě. Globální souřadnicový systém pracuje za jakýchkoli meteorologických podmínek (ale je doporučené mít přímou viditelnost na oblohu), kdekoli na světě, 24 hodin denně. Každý ze satelitů posílá údaje o své poloze a času každých 12 hodin. Tyto informace jsou posílány pomocí rádiových signálů pohybujícími se téměř rychlostí světla. Přijímače (GPS lokátory, mobilní zařízení, navigace apod.) dokážou na základě těchto signálů vypočítat svou vzdálenost od daného satelitu.

Pro samotný výpočet aktuální polohy zařízení se využívá princip trilaterace viz obrázek *Obrázek* 5 Trilaterace . K určení aktuální polohy zařízení je potřeba nejméně tří satelitů, výsledná poloha je získána výpočtem průniku tří sfér . Čím více signálů zařízení zachytí, tím přesnější je výpočet polohy. Přesnost vypočtu může být také ovlivněná dalšími faktory, přesností hodin v zařízení, aktuální polohou, elektromagnetickými poli atd.



Obrázek 5 Trilaterace [9]

[9] Převzato z http://clipart-library.com/clipart/gps-cliparts_9.htm

3 Analýza

Tato kapitola se bude zabývat analýzou projektu. Prvním krokem bude provedena analýza již existujících projektů, které používají rozšířenou realitu s jejích následným vyhodnocením. Druhým krokem bude provedena analýza stávající mobilní aplikace pro zobrazení virtuálních naučných stezek *EduARd*, do které bude implementováno rozšíření. Třetím krokem bude provedena analýza dostupných nástrojů, které umožní implementovat rozšířenou realitu pro *Android* [10] s následným vyhodnocením a zdůvodněním vybrané technologii.

3.1 Analýza existujících projektů

V této podkapitole budou uvedené podobné již existující projekty, které se zabývají zobrazováním 3D scén a jsou zaměřené na žáky základních a středních škol. Potom bude provedeno jejich vyhodnocení. Pro přehlednost byla určena následující struktura analýzy těchto aplikací:

- Základní informace
- Princip použití
- Přínos
- Podporované platformy
- Určení lokality
- Cena

[10] Android je mobilní operační systém – převzato z [29].

3.1.1 Pokémon Go

Základní informace

Hra Pokémon Go byla vydána 6. července 2016 společností Niantic. Je to mobilní aplikace, která využívá principy rozšířené reality. Propojuje herní prostředí s reálným světem.

Obrázek 6. Hra Pokémon Go[11]

Princip použití

V pravidlech hry není určen cíl, kterého má hráč dosáhnout. Také neexistuje vítěz hry. Hra nikdy neskončí a každý hráč si vybere svůj vlastní cíl, kterého chce během hry dosáhnout. Avšak lze definovat globální cíl hry: hráči chtějí nasbírat všechny druhy pokémonů a co nejvíce je vylepšit, aby mohli v bojích proti ostatním hráčům zvítězit. Hra dovoluje hrát jak samostatně, tak i v týmech [4].

Přínos

Přínosem této hry je pohyb a navštěvování míst v přírodě zapříčiněné hledáním pokémonů a skutečná komunikace hráče s ostatními hráči například při soubojích či výměně pokémonů.

Podporované platformy

Hra je dostupná pro platformy *Android* a *iOS* [12].

Určení lokality

Pro lokalizaci používá nejen GPS, ale také *BTS* [13] stanice operátorů či *Wi-Fi* [14] sítě.

Cena.

Aplikace může být stažena a hrána bezplatně, ale uživatelé také mají možnost kupovat další přídavné předměty a vlastnosti hry navíc.

[11] Převzato z https://cdn.slashgear.com/wp-content/uploads/2017/12/pokemon-go-arplus-iphone-x-980x620.jpg

[12] iOS je mobilní operační systém pro telefony iPhone společnosti Apple – převzato z [30].

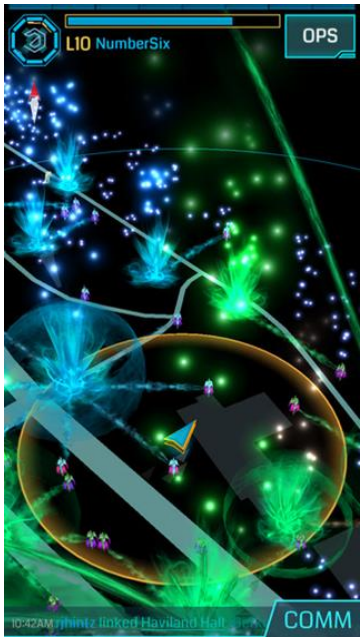
[13] Systém základnových stanic je system, který zodpovědný za přenos a příjem radiových signálů z mobilního telefonu.

[14] Wi-Fi je označení pro bezdrátové propojení – převzato z [40].

3.1.2 Ingress Prime

Základní informace

Hra Ingress Prime byla vydána 19. listopadu 2013 společností Niantic Labs, spadající pod společnost Google. Je to online hra, tj. zařízení být připojeno k mobilnímu internetu, pro mnoha hráčů využívající možností rozšířené reality [5]. Ukázka ze hry Ingress Prime je na obrázku *Hra Ingress Prime*.



Obrázek 7: Hra Ingress Prime[15]

Princip použití

Hra je založena na stejném principu jako předchozí hra – *Pokémon Go*, tj. hledání virtuálních předmětů, v daném případě místo pokémonů hráč obsazuje portály. Tyto portály se nachází na zajímavých místech po celém světě (tyto místa lze najit na mapě v aplikaci *Ingress*). Před začátkem hry hráč vybírá jeden ze dvou tymů, se kterým nadále bude spolupracovat. Cílem je dostat se na určité místo, obsadit portál a postavit na něm svou virtuální pevnost, kterou pak bude bránit před jiným týmem. Také je nutné během hry nasbírat "tajemnou hmotu", což je virtuální energie, kterou využije při obsazování portálů a následně jeho obraně.

Přínos

Přínosem této hry je, že hráč při hraní nesedí pouze doma, ale musí navštívit různé památky ve svém okolí a popřípadě se dozví nějaké informace o nich.

Podporované platformy

Hra je dostupná pro platformy Android a iOS.

Určení lokality

Pro určení polohy uživatele je používána GPS.

[15] Převzato z https://techcrunch.com/2018/11/05/what-is-ingress-prime/

Cena

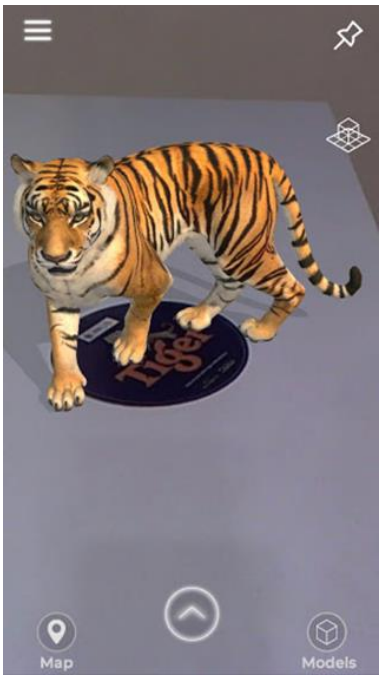
Stažení a hrání je zadarmo ale je možnost dokoupit některé položky hry.

3.1.3 ARLOOPA

Základní informace

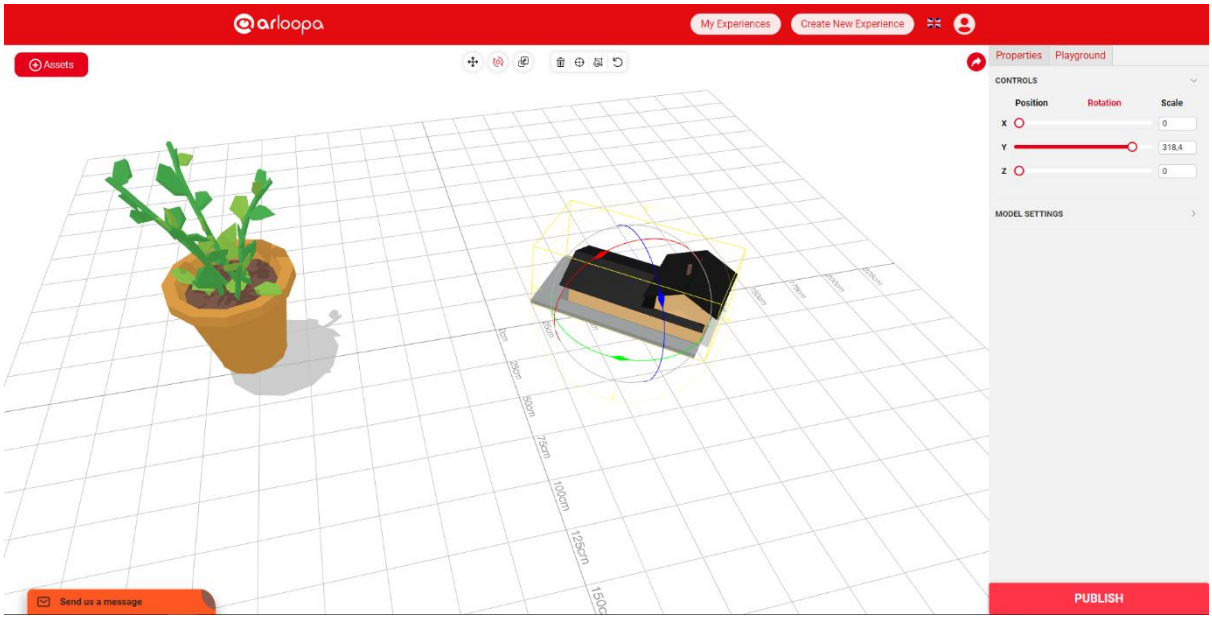
ARLOOPA je aplikace pro zážitky z AR. Byla vydaná společností ARLOOPA. Jedná se o mobilní aplikaci, která využívá technologii rozšířené reality, rozpoznávaní obrazu a počítačové vidění. Tato aplikace obsahuje možnost zobrazení vlastních vytvořených scén včetně modelů. Sestavit scénu se je možno pomocí webové aplikace [6].

Ekosystém ARLOOPA je v podstatě totožný s tím, jaký bude implementován v rámci projektu EduARd. Ukázka mobilní aplikace je na obrázku *ARLOOPA App*.



Obrázek 8. ARLOOPA App

Ukázka webové aplikace pro tvorbu scén je na obrázku *ARLOOPA Webová aplikace pro tvorbu scén*



Obrázek 9 ARLOOPA Webová aplikace pro tvorbu scén

Princip použití

Aplikace umožňuje marker-based skenování (skenování založené na značkách), marker-free sledování (sledování bez markerů), geolocation-based zkušenosti (zkušenosti založené na geografickém umístění) a skládá se z následujících částí:

- Skener AR. Aplikace umožní vybrat obrázek pro skenování z jejich oficiální stránky a následně zobrazit rozšířenou realitu (2D obrázek, video nebo 3D animace).
- Knihovna 3D modelů. 3D modely lze umístit kdekoli. Stačí naskenovat jakýkoli povrch a umístit na něj 3D objekt.
- Mapa. Značky založené na geografické poloze lze prozkoumat pouze na místech, kde se nacházejí v okruhu 50 metrů.

Přínos

Velmi užitečná pro vzdělávací cíle. Mnoho učitelů z celého světa již tuto aplikaci objevilo a tvrdí, že to zvyšuje zájem u dětí o probíranou látku, a že to skutečně stalo hitem ve školách.

Podporované platformy

Aplikace je dostupná pro platformy Android a iOS.

Určení lokality

Pro určení polohy uživatele je používána GPS.

Cena

Aplikace je zdarma ke stažení. Většina 3D modelů se musí kupovat, ale některé jsou i zdarma. Pro vytváření scén musí uživatel zakoupit předplatné, které startuje od 325 korun.

3.1.4 Vyhodnocení

V tabulce 1. jsou uvedené výhody a nevýhody existujících řešení. Také je potřeba říct, že existuje mnoho dalších aplikací, které se zabývají zobrazováním 3D scén a jsou zaměřené na žáky základních a středních škol jako například Vortex Planetarium, Visual Science, Rubius apod.

Aplikace	Výhody	Nevýhody
Pokémon Go	- Uživatelé tráví více času venku - Socializace dětí - Zjištění polohy třemi způsoby - Seznámení děti s novými technologiemi - Aplikace je zadarmo	- Zatíženy servery a kvůli tomu jsou občas výpadky aplikace - Aplikace rychle vybíjí chytré telefony
Ingress Prime	- Možnost navštívit nová místa a památky - Poznávání nových přátel - Příležitost pro socializaci - Aplikace je zadarmo	- Uživatele se stěžují na GPS - Vysoká spotřeba baterie - Zatíženy servery a kvůli tomu se někteří uživatelé nemohou přihlásit
ARLOOPA	- Zvyšuje zájem u dětí o vědy - Dovoluje učitelům zobrazovat žákům 3D modely - Velká knihovna 3D modelů - Široká funkcionalita - Možnost virtuálního cestování - Možnost vytváření vlastních scén a nahrávání vlastních modelů.	- Většina funkcionalit je potřeba dokoupit - Pro možnost sestavení scén, uživatel musí pořídit předplatné.

Tabulka 1. Výhody a nevýhody existujících řešení

3.2 Analýza stávající mobilní aplikací

3.2.1 Cíle aplikace

Cílem stávajících aplikací je přesunout výuku středoškoláku mimo školní lavice. Pro tento účel v dané aplikaci byla implementována možnost prohlédnutí takzvaných virtuálních naučných stezek, které slouží jako alternativa běžných učebnic. Pokud by student chtěl tyto učebnice prohlížet, musel by danou naučnou stezku sám projít a navštívit všechny body zájmu.

3.2.2 Implementované řešení

Tato mobilní aplikace byla vyvíjena pomocí frameworku *Expo*, který je postavený na *React Native* [16]. Framework *Expo* je dobrou volbou, pro daný typ aplikací, protože umožní sestavovat aplikace na serverech *Expo*, a nabízí je dále volně ke stažení mobilní aplikace *Expo*.

Ale tento způsob puštění aplikace klade určíte omezení pro implementací rozšíření. A to, že výše uvedený framework nepodporuje integrací s knihovny či frameworky, které nejsou v ekosystému *Expo*. Proto pří implementaci může být ztracena možnost sestavovat aplikace přímo na serverech *Expo*.

3.3 Analýza dostupných nástrojů pro rozšířenou realitu

V této podkapitole bude uvedeno několik dostupných nástrojů, které umožní implementovat rozšířenou realitu pro Android. Následně bude provedeno jejich vyhodnocení. Pro přehlednost byla určena následující struktura analýzy:

- Základní informace
- Základní funkce
- Omezení

3.3.1 Google ARCore

Základní informace

Google ARCore je SDK společnosti Google, která umožní vývojářům psát aplikace pro zobrazení obsahu rozšířené reality pro *Android* [7].

Základní funkce

Tyto tři funkce umožňují smartphonu, pomocí kamery a interních senzorů, porozumět svému reálnému okolí a zobrazovat v něm virtuální objekty.

- **Sledování pohybu** – umožňuje smartphonu vyhodnotit pozici zařízení vůči okolnímu světu, pomocí snímače *IMU* [17] a kamery přístroje k zobrazení bodů funkce v místnosti.

[16] Viz sekce *5.1.7 React-Native*.
[17] Inerční měřicí jednotka – převzato z [31].

- **Pochopení prostředí** – umožňuje smartphonu určit velikost a umístění všech typů povrchů (svislých, vodorovných a úhlových).
- **Vyhodnocení světelných podmínek** – umožňuje smartphonu posoudit aktuální světelné podmínky prostředí.

Omezení

Vývoj aplikací pouze pro Android.

3.3.2 ARToolKit

Základní informace.

ARToolKit je knihovna, která umožňuje programátorům snadno vyvíjet AR aplikace. Byla vyvinuta Hirokazem Kato v roce 1999 a je Open Source [18] s možností prodeje komerčních licencí společností ARToolWorks [8].

Základní funkce

Sledování pozice jedné či dvojice kamer, sledování rovinných obrazců a jakýchkoliv jiných čtvercových vzorů a je dostatečně rychlá pro rozšířenou realitu v reálném čase.

Omezení

ARToolKit používá pro trasování pozice kamery algoritmy počítačového vidění, které jsou schopny počítat rychle v reálném čase. Výsledky jsou však závislé na použitém hardware a videokameře. Nepodporuje rozšířenou realitu založenou na poloze.

3.3.3 ViroReact

Základní informace

ViroReact je Open Source framework pro vývoj jak rozšířené, tak i virtuální reality pomoci *React-Native*. Kvůli tomu, že ve své práci budu implementovat rozšířenou realitu, tak nadále budu uvažovat pouze o AR funkcionalitě knihovny *ViroReact* [9].

Základní funkce

Framework se skládá ze dvou částí. První je výkonný nativní 3D vykreslovací engine a druhou je značné množství *React* komponent, které jsou využívány jak pro vykreslování scén, tak i pro tvorbu jejich obsahu.

Platforma podporuje rozšířenou realitu v podobě ARCore pro *Android* i ARkitu pro *iOS*. Velkou výhodou platformy je dostupnost testovací aplikace umožňující jednoduché spuštění vyvíjené aplikace na skutečném zařízení bez nutného sestavování aplikace před každým spuštěním.

Omezení

Nenabízí plnou podporu všech funkcionalit implementovaných v ARCoru a ARKitu.

[18] Open source - je počítačový software s otevřeným zdrojovým kódem.

3.3.4 Vyhodnocení

	ARCore	**ARToolKit**	**Viro-React**
Platformy	Android 7.0 Nougat a vyšší	Windows, Mac OS X, Linux, iOS, Android,	Android, iOS
Licence	Open-Source	Open-Source, GNU Lesser GPL v3.0	Open-Source
Dokumentace	Dostatečná	Dostatečná	Nedostatečná
Podporované Frameworky	Unity, Xamarin, Flutter, Unreal	Unity	React Native
Formáty	.obj, .glTF, .fbx	.obj, .v2, .x3d	.obj, .glTF, .fbx
Location based AR	Podporuje	Nepodporuje	Podporuje

Tabulka 2 Porovnání knihoven

Z důvodu, že cílem teto práce je implementovat rozšíření mobilní aplikace, která je napsána v React Native s použitím *Expo*, ideální volbou pro vyvíjení AR rozšíření je platforma *ViroReact*.

Ale i pokud by tento požadavek nebyl určen, tak z mého pohledu *ViroReact* je stále dokonalá volba. Protože ARToolKit nepodporuje rozšířenou realitu založenou na poloze, což je pro moji práci zásadní požadavek a také nepodporuje framework *React-Native*. A ARCore je omezen vývojem pouze pro platformu Android.

3.4 Analýza scénáře a scén

V této podkapitole budou popsané scénáře a scény včetně modelů. Daná struktura je již implementována v rámci vývoje ekosystému *EduARd*. Kde pan *Bc. Dominik Truong,* v rámci své bakalářské přací, implementoval mobilní aplikaci pro zobrazeni scén v rozšířené realitě založenou na GPS lokalizaci [10]. A pan *Bc. Michal Mráz,* v rámci své bakalářské přací, implementoval webový portál pro přípravu podkladů pro rozšířenou realitu [11]. Následující popis scénáře a scén převzat s těchto prací.

3.4.1 Scénáře a scény

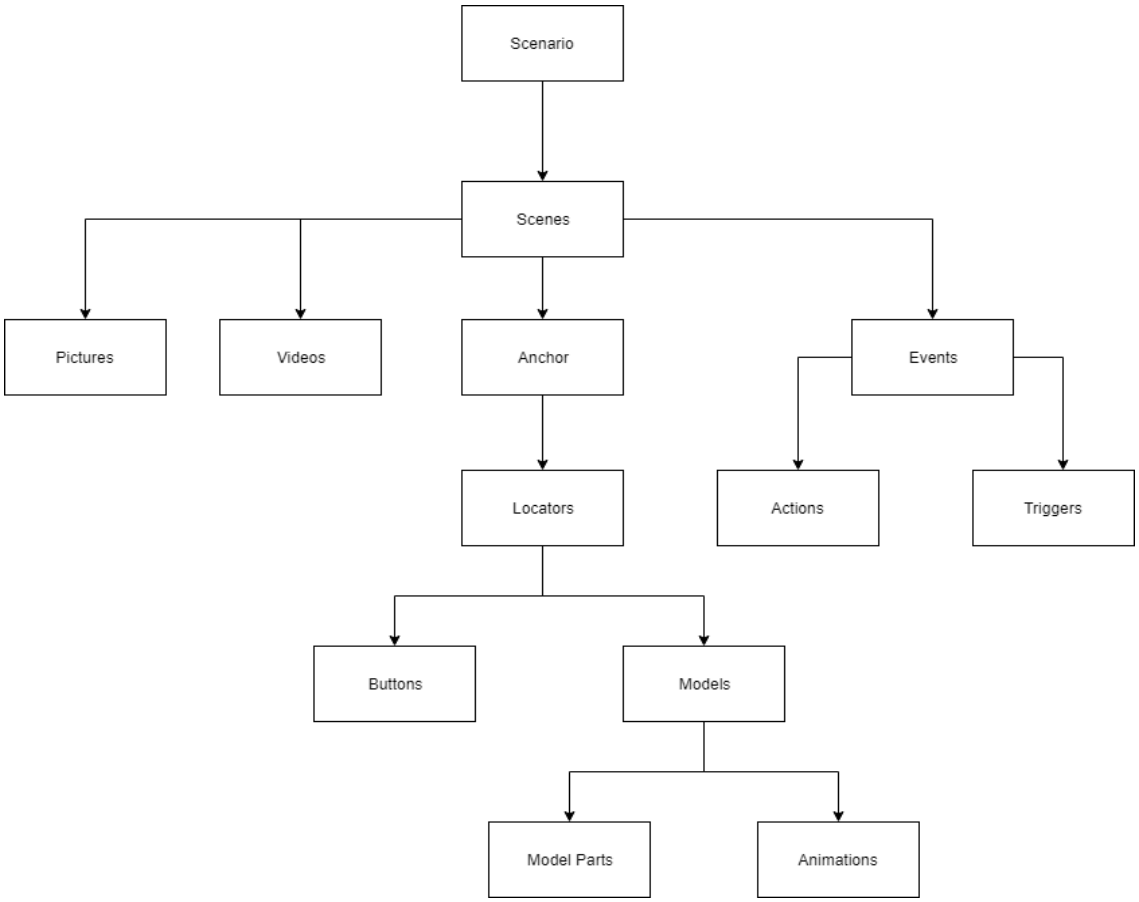
Scénář je množina scén, které definuji objekty, jejich vlastnosti a chovaní. Scénář je výstupem editoru 3D scén. Editor 3D scén je webová aplikace umožňující tvorbu a úpravu scénářů. Popis scénářů je ve formátu JSON [19]. Scénář je hierarchicky uspořádaný do uzlů. Uzlem je objekt.

Níže je uveden seznam objektů. Hierarchie těchto objektů je znázorněna na obrázku Obrázek *10 Hierarchie objektů* ve scénáři.

- **Scenario** – Objekt popisující scénář obsahující seznám scén.
- **Scene** – Objekt popisující scénu.
- **Anchor** – Objekt popisující počátek scény, která je definovaná GPS souřadnicemi.
- **Locator** – Objekt popisující lokátor je objekt sdružující určitou podmnožinu objektů ve scéně.
- **Model** – Objekt popisující model. Model je 3D objekt ve scéně.
- **Model Part** – Objekt popisující část modelu.
- **Animation** – Objekt popisující animaci ve scénáři.
- **Button** – Objekt popisující element uživatelského rozhraní – tlačítko.
- **Event** – Objekt popisující události, které definují interakce s objekty ve scéně pomocí seznamu spouštěčů a akcí.
- **Trigger** – Objekt popisující podmínky pro spuštění akcí.
- **Action** – Objekt popisující akci, která je spouštěná, pokud jsou splněné všechny podmínky pro spuštění akcí.
- **Picture** – Objekt popisující obrázek.

[19] Viz sekce *5.1.2 JSON*.

- **Video** – Objekt popisující video.



Obrázek 10 Hierarchie objektů ve scénáři

3.4.2 Modely

Vzhledem k tomu, že editor 3D scén je webovou aplikací, která využívá modely ve formátu X3D [20], který lze jednoduše zobrazit pomocí webového prohlížeče, je potřeba aby implementované rozšíření dokázalo načítat soubory tohoto formátu.

Vzhledem k tomu, že používaný framework *ViroReact* pro práci s rozšířenou realitou nepodporuje načítání X3D formátu, je potřeba napsat vlastní logiku pro načítání X3D formátu do struktur vybraného frameworku.

3.4.3 Interakce

Interakce je ve scénáři definována seznamem událostí. Událostí definují jednotlivé spouštěče a přiřazují jim jednotlivé akce. Aplikace pravidelně kontroluje, zdali je podmínka všech spouštěčů splněna. Pokud ano, vyvolá akce, které jsou přirazené.

Všechny spouštěče a podmínky pro jejich splnění:

[20] Viz sekce *5.1.3 X3D*

- **Vzdálenost od objektu** - spouštěč definuje vzdálenost objektu od uživatele. Pokud je uživatel se nachází blíž, tak podmínka je splněna.
- **Pohled na objekt** - pokud se uživatel dívá na objekt, který byl nastavený ve spouštěči, tak podmínka je splněná.
- **Stisknutí tlačítka** - Pokud uživatel stiskne tlačítko dané spouštěčem, tak podmínka je splněná.

Všechny akce a popis jejich chování:

- **Interpolace rotace** - změna rotace objektu v čase.
- **Interpolace barvy** - změna barvy materiálu v čase.
- **Interpolace pozice** - změna pozice objektu v čase.
- **Interpolace škály** - změna velikosti v čase.
- **Zobrazování obrázků** - zobrazení obrázku v uživatelském rozhraní.
- **Přehrávání videí** - přehrání videa v uživatelském rozhraní.

3.5 Shrnuti kapitoly

Prvním cílem této kapitoly bylo provést důkladnou a kvalitní analýzu existujících řešení, která je nezbytnou částí k pochopení problematiky. Druhým cílem byla analýza stávající mobilní aplikace *EduARd*. Třetím cílem byla analýza dostupných nástrojů, na základě které byl vybrán framework pro práci s rozšířenou realitou – *ViroReact*. Konečným cílem bylo provést analýzu scénářů včetně jejich interakce pro lepší pochopeni struktury dat, které se budou používat při vývoji rozšíření.
Cil této kapitoly považuji za splněný.

4 Návrh

Následující kapitola se bude zabývat návrhem řešení. Nejprve zde bude podrobněji popsán vybraný pro implementací rozšíření framework *ViroReact* a problémy s jeho použitím. Pote zde bude uvedeny požadavky na rozšíření.

4.1 Framework ViroReact

ViroReact je frameworkem, který umožňuje nejen snadnou práci s rozšířenou realitou, ale i poměrně velkou rychlost vývoje mobilní aplikace. Je to dáno tím, že je framework založen na *React Native* frameworku od společností Facebook, který umožní vývojáři se soustředit více na vývoj samotné aplikace, a ne na nastavení prostředí pro samotný vývoj.

Framework *ViroReact* využívá volně dostupné SDK ARkit od společností Apple a ARCore od společností Google. Z toho plyne, že tento framework umožní vývojáři napsat jen jednou aplikaci, která bude fungovat zároveň na dvou systémech (*iOS* a *Android*) bez velkých uprav v kódu.

ViroReact poskytuje vývojáři velkou sadu možností pro rychlý a snadný vývoj mobilních aplikací s rozšířenou realitou. První nepříjemnou vlastnosti tohoto frameworku, je to že, nepodporuje načítání modelů ve formátu X3D, který byl požadován v zadání práce. Naštěstí obsahuje komponentu, která umožní nahrávání geometrie 3D modelu, která je zapsaná jako *Indexed-Face-Set*. Přesně tento princip zápisu geometrie používá se ve formátu X3D. Tato komponenta také obsahuje možnost přiřazení materiálu a animaci.

Implementované rozšířeni musí také zachovat stávající stav aplikaci *EduARd*. Zachování stavu znamená, že žádná knihovna či kód nesmí byt odstraněny. Kvůli tomu je potřeba provést integraci frameworku *Expo* s frameworkem *ViroReact*, což považuji za největší problém této práce.

4.2 Požadavky na rozšíření

S ohledem na analýzu existujících řešení, analýzu nástrojů pro rozšířenou realitu a cíle projektu, byli definované následující požadavky na systém.

4.2.1 Funkční požadavky

Funkční požadavky zachycují zamýšlené chování systému. Toto chování lze vyjádřit jako služby, úkoly nebo funkce, které má systém provádět [12].

Jak už bylo uvedeno v podkapitole *1.2 Cíle projektu a cílová skupina* cílem této bakalářské práce je navrhnout a implementovat rozšíření mobilní aplikaci *EduARd* o možnosti zobrazení rozšířené reality za využitím GPS lokalizací. Na základě toho můžeme určit požadavky na toto rozšíření.

- **SRQ 101. Komunikace se serverem projektu EduARd**

Rozšíření umožní komunikaci se serverem projektu *EduARd* pro stahování dat, která slouží k načítání scény. Komunikace probíhá pomocí *REST API* [21]. Pro stahování scénářů je potřeba získat token, který je odlišný od tokenu používaného při stahování stezky.

- **SRQ 102. Stažení scén ze serveru**
  Rozšíření umožní stahovat scénáře, které obsahuji popis scén ve formátu *JSON* ze serveru EduARd a následně stahovat obsah těchto scén – 3D modely, představené ve formátu X3D, rastrové obrázky a videa.
- **SRQ 103. Zobrazení načtených scén**
  Rozšíření umožní uživateli zobrazit načtené scény v reálném světě pomocí technologie rozšířené reality skrze mobilní telefony anebo tablety. Scény se zobrazují, když skutečná poloha uživatele, která je určena pomocí GPS lokalizace, nachází poblíž vybrané scény.
- **SRQ 104. Načtení a vykreslení scény bez připojení k internetu**
  Rozšíření umožní uživateli načítat a vykreslovat scény bez připojení k internetu. Je nutné mít zapnuté GPS na zařízení uživatele.
- **SRQ 105. Určení vzdálenosti od scény.**
  Rozšíření umožní uživateli určit vzdálenosti od scény pomocí GPS lokalizace.
- **SRQ 106. Ukládaní scénáře.**
  Rozšíření umožní ukládat scénáře do mobilního zařízení a mazat je.

4.2.2 Kvalitativní požadavky a omezení

V této podkapitole budou uvedeny kvalitativní požadavky a omezení (nefunkční požadavky) kladené na rozšíření.

Kvalitativní požadavky a omezení (NFR) – v softwarovém systémovém inženýrství požadavek na software, který nepopisuje, co software bude dělat, ale jak to software bude dělat [13].

- **NFR 101. Kompatibilita.**
  Aplikace bude kompatibilní s operačním systémem (OS) *Android*.
- **NFR 102. Jednoduchost.**
  Aplikace bude mít intuitivní uživatelské rozhraní, které umožní se rychle zorientovat a nadále jednoduše používat tuto aplikaci.
- **NFR 103. Uživatelsky rozhraní.**
  Při návrhu uživatelského rozhraní je vhodné se inspirovat již existujícím řešením (mobilní aplikace *EduARd*).

[21] REST API je architektura rozhraní, navržená pro distribuované prostředí – převzato z [32].

- **NFR 104. Uložení scén.**
  Aplikace bude ukládat všechna data tykající se zobrazeni scén, aby scény se zobrazovali i když uživatel není připojený k internetu.
- **NFR 105. Rozšiřitelnost.**
  Aplikace bude umožňovat snadné rozšíření o další funkcionality.

.

5 Implementace

V teto kapitole je popsaná implementační část projektu. V první častí budou popsané použité nástroje a technologie pro realizací projektu. Následně zde bude vysvětlen postup integrace frameworku *ViroReact* s frameworkem *Expo*, na kterém je položena stávající aplikace *EduARd*. Potom zde bude popsaná implementace stahování dat ze serveru, GPS lokalizace, načítaní scénáře a scény včetně souborů.

5.1 Použité nástroje a technologii

5.1.1 IntelliJ IDEA

IntelliJ IDEA je vývojové prostředí od firmy JET BRAINS pro psaní a ladění kódu záměrně na programovací jazyk JAVA. V závislosti na verzi podporuje i jiné programovací jazyky a technologie.

IntelliJ IDEA je vydáváno ve dvou verzích: Community Edition a Ultimate Edition. Community Edition je zcela zdarma, poskytuje poměrně dostačující funkcionalitu pro psaní a ladění kódu, ale má své omezení. Jazyková podpora: Java, Groovy, XML, Regexp.

Ultimate Edition je placená verze. Oproti bezplatné verzi podporuje více jazyků, a to je SQL, HTML, XHTML, CSS, XSL, XPath, FreeMarker/Velocity, JavaScript/ActionScript, PHP. Také podporuje aplikační servery: Apache Tomcat, GlassFish, JBoss, WebLogic, WebSphere, Geronimo a Resin [14].

V mojí semestrální práci jsem použil Ultimate Edition, vzhledem k tomu, že pro studenty ČVUT je tato verze zcela zdarma v rámci podpory výuky.

5.1.2 JSON

JSON je známý jako standard pro výměnu dat. Je založen na podmnožině programovacího jazyka JavaScript, Standard ECMA-262 3rd Edition – December 1999. JSON je textový, na jazyce zcela nezávislý formát, který využívá konvence rodiny jazyku (C, C++, Java, JavaScript, Python a dalších). Díky tomu JSON stal ideálním kandidátem pro výměnu dat mezi distribuovanými aplikacemi. Během několika posledních let se JSON prosadil především jako silně odlehčená náhrada XML či jako alternativa k němu. Hlavní výhoda JSON oproti XML je menší velikost přenášených dat [15].

JSON formát založen na dvou jednoduchých pro pochopení strukturách.

- Kolekce párů název/hodnota. V programovacích jazycích tato struktura obvykle realizována jako objekt, záznam, struktura, slovník, hash tabulka, klíčový seznam nebo asociativní pole.
- Seřazený seznam hodnot. Ten je obvykle realizován jako pole, vektor, seznam nebo posloupnost.

5.1.3 X3D

X3D je grafický formátem vycházející z formátu *VRML* [22], kvůli tomu občas nazýván jako *VRML* 3.0. Oproti předchozí verzí, podporuje syntaxi zápisu založenou na *XML* [23]. To přináší řadu předností, především jednoduché zpracovaní celého dokumentu za pomoci volně dostupných knihoven a programových API pro práci s *XML*, možnost serializace a deseralizace prostorové scény do X3D a relativně snadné realizovatelné převody mezi X3D a dalšími formáty. Soubory X3D obsahují definice modelů ve scéně, například definice vrcholů, trojúhelníků, materiálů, texturovaných souřadnic (včetně odkazů na samotné textury) či transformací těchto objektů (translace, rotace, škálování apod.). Formát X3D také umožňuje definovat animace pomocí interpolátorů, senzorů a cest [16]. Příklad modelu ve formátu X3D je znázorněn na obrázku *Obrázek 11 Model ve* formátu X3D.

```
<X3D profile='Full'>
  <Scene>
    <Transform translation='0 2 0' scale="0.700735 0.700734 0.700734" >
      <Transform translation='-5 0 0' rotation="0 0.453540 0.891213 3.14">
        <Shape>
          <IndexedFaceSet DEF='IfsMeshBox' coordIndex='0 1 2 3 0 -1 7 6 5 4 7 -1 0 4 5 1 0 -1 1 5 6 2
            1 -1 2 6 7 3 2 -1 3 7 4 0 3 -1'>
            <Coordinate point='1 1 1 1 1 -1 -1 1 -1 -1 1 1 1 -1 1 1 -1 -1 -1 -1 -1 -1 -1 1'/>
          </IndexedFaceSet>
          <Appearance>
            <Material diffuseColor='1 0.2 0.2'/>
          </Appearance>
        </Shape>
      </Transform>
    </Transform>
  </Scene>
</X3D>
```

Obrázek 11 Model ve formátu X3D

5.1.4 Node.js

Node.js je nezbytnou častí vývoje v React-Native. Jedná se o prostředí, které umožní spouštět JavaScript mimo webový prohlížeč. Tím umožní používání JavaScript nejen pro vytvoření webových stránek, ale také i pro tvorbu jakýkoliv aplikaci. V Node.js kladen důraz na vysokou škálovatelnost a výkonnost, což přináší vývojářům moznost vytvářet aplikace bud mobilní nebo desktopové jen s dobrou znalosti programovacího jazyka JavaScript.

Node.js je postaven na *Chrome V8 Engine* napsaného v C++, který překládá kód napsaný vývojářem na rychlejší strojový kód [17].

[22] Angl. Virtual Reality Modeling Language.
[23] Angl. Extensible Markup Language.

5.1.5 Node Package Manager

Jedná se o správce balíčku (knihoven) pro Node.js. Používá se pro snadnou správu knihoven, které jsou většinou volně dostupné ke stažení přímo z konzole. Názvy knihoven, které jsou potřebné pro spuštění projektu, jsou vždy uloženy v souboru Package.json. Stažené knihovny lze najít ve složce *node_module*. V souboru *package-lock.json* jsou popsané všechny stažené knihovny, které jsou potřebné pro správné fungování knihoven uvedených v *package.json* [18].

5.1.6 Gradle

Gradle je nástroj pro automatizaci sestavování programu vzniklý původně pro prostředí Javy a stavící na zkušenostech s nástroji Apache Ant a Apache Maven a na programovacím jazyku Groovy. Jedná se o nástroj multiplatformní používající licenci Apache [19].

5.1.7 React-Native

React-Native je sada nástrojů, které jsou napsané v programovacím jazyce JavaScript pro tvorbu nativních mobilních aplikací a vycházející přímo z frameworku React. React-Native podporuje platformy jako Android, Android TV, iOS, macOS, Apple tvOS, Web, Windows a UWP [20].

5.1.8 Patch-package

Patch-package je knihovna která umožňuje vývojářům okamžitě vytvářet a udržovat opravy, které jsou závislé na NPM [21].

5.1.9 Jetifier

Nástroj Jetifier migruje knihovny závislé na podpoře, aby se místo toho spoléhaly na ekvivalentní balíčky AndroidX. Tento nástroj umožňuje migrovat jednotlivou knihovnu přímo, namísto použití pluginu Android gradle, který je součástí aplikace Android Studio [22].

5.1.10 ThreeJs

ThreeJs je JavaScript knihovna a aplikační rozhraní s širokou podporou webových prohlížečů, které slouží k vytváření a zobrazování 3D obsahu ve webovém prohlížeči. Využívá aplikačního rozhraní WebGL [23].

5.2 Integrace ViroReact a Expo

Stávající mobilní aplikace byla vytvořena pomocí frameworku *Expo*. Který má určité výhody v tom, že obsahuje skoro všechny komponenty, které jsou potřebné pro vývoj mobilní aplikace. Ale má také velkou nevýhodou v tom, že nepodporuje rozšíření své funkcionality za použití externích knihoven, které potřebuje takzvané linkování.

Linkování je proces, který je potřebný pro správnou instalaci některých knihoven, která spočívá v tom, že po instalaci knihovny pomocí správce balíčku *Nod Package Manager*, je potřeba dopsat cesty do vybraných knihoven v konfiguraci projektu a také upravit řídicí třídu *MainApplication.java* pro importovaní nalinkovaných knihoven. Tento proces není dostupný ve frameworku *Expo*, z toho důvodu, že *Expo* má tyto soubory uložené na svých serverech. Pravě kvůli této vlastnosti *Expo* projekt není potřeba sestavovat na počítací, toto sestavení se provádí na serverech *Expo* a pak pomocí mobilní aplikace *Expo* zobrazuje sestavenou aplikaci vývojáři. Právě tato nevýhoda *Expo* frameworku se projevila při integraci s frameworkem *ViroReact*, které toto linkování vyžaduje.

Po rešerši možného řešení bylo zjištěno, že pokusy o integraci vybraných frameworků byli, ale neúspěšné. Také na stránkách *Expo* byla nalezena prosba vývojářů o danou integraci, ale bohužel bez odpovědi ze strany vývojářů *Expo*. Ještě důležité upozornit, že stávající mobilní aplikace využívala *React-Native* ve verzi 61.1+ ale *ViroReact* podporuje jen verze 59.9.

Po několika pokusech integrace frameworku *ViroReact* do stávající aplikace bylo rozhodnuto vytvoření prázdného *ViroReact* projektu, který obsahoval všechny potřebné knihovny pro fungovaní rozšířené reality. Vzhledem k tomu, že stávající aplikace používala skoro všechny knihovny z prostředí *Expo*, nestačilo jen okopírovat *package.json* a samotný kód, protože skoro každá knihovna využívala vnitřní knihovny *Expo*. Tyto knihovny jsou součásti jádra *Expo*. Toto jádro nemělo v dané aplikaci svůj vlastní odkaz pro staženi, ale bylo součástí *React-Native* knihovny frameworku *Expo*.

Odkaz, který používala původní aplikace pro stáhnutí jádra *React-Native* a *Expo* je ukázán na obrázku *Obrázek 12 Odkaz do React-native knihovny v* stávající aplikaci.

```
"react-native": "https://github.com/expo/react-native/archive/sdk-38.0.0.tar.gz"
```

Obrázek 12 Odkaz do React-native knihovny v stávající aplikaci

Jak již bylo zmíněno výše, *ViroReact* podporuje *React-Native* ve verzi 59.9 a proto takový odkaz do knihovny *React-Native* v souboru *package.json* nemůžeme povolit.

5.2.1 Implementované řešeni

Řešením problému integrace v dané situaci bylo to, že kód ze stávající aplikace *Expo* byl postupně přenášen do nové vytvořeného *ViroReact* projektu. V průběhu přenosu kódu byly nalezeny knihovny, které jsou potřebné pro fungování *Expo* knihoven.

Seznam těchto knihoven:

- *unimodules-permissions-interface*
- *unimodules-task-manager-interface*
- *react-native-unimodules*
- *@react-native-community/masked-view*
- *expo-av*
- *react-native-fs*
- *react-native-safe-area-context*
- *react-native-webview*
- *react-navigation*
- *react-navigation-stack*

Některé z těchto knihoven bylo potřeba nalinkovat do konfigurace projektu. Zda knihovnu je potřeba linkovat nebo ne, je možno zjistit v repozitářích vybraných knihoven, kde je popsán instalační proces. Linkování knihoven bylo vyzkoušeno a otestováno jen pro platformu Android, z důvodu toho, že autor této práce nemá iOS zařízení.

5.2.2 Proces linkování knihoven

Proces linkování knihoven je možné provést dvěma způsoby. První je jednoduchý, po instalaci knihovny pomocí správce balíčku *npm* je potřeba do konzole napsat *"npx react-native link"*, což by mělo nalinkovat knihovnu pro Android a iOS současně. Ale dost často tento způsob negarantuje správné linkování, proto je potřeba nalinkovat knihovny ručně. Níže je popsán proces linkování knihovny *@react-native-community_masked-view* pro Android.

Prvním krokem je najít soubor *android/settings.gradle*. V tomto souboru přidat následující řádky, které jsou znázorněný na obrázku *Obrázek 13 Linkovaní knihovny v souboru android*/settings.gradle.

```
include ':@react-native-community_masked-view'
project(':@react-native-community_masked-view').projectDir = new
File(rootProject.projectDir, '../node_modules/@react-native-community/masked-
view/android')
```

Obrázek 13 Linkovaní knihovny v souboru android/settings.gradle

Druhým krokem je najít soubor *android/app/build.gradle*. V tomto souboru do sekce *dependencies* přidat následující řádky, které jsou znázorněny na obrázku *Linkovaní knihovny v souboru android/app/build.gradle* .

```
dependecies {
   implementation project(':@react-native-community_masked-view')
}
```

Obrázek 14 Linkovaní knihovny v souboru android/app/build.gradle

Třetím krokem je najít soubor *android/app/src/main/java/com/eduard/MainApplication.java*. V tomto souboru na začátku musí být naimportována knihovna jak je znázorněno na obrázku *Import knihovny*.

```
import org.reactnative.maskedview.RNCMaskedViewPackage;
```

Obrázek 15 Import knihovny

Dále ve funkci *getPackages* do proměnné *packages* je potřeba přidat dříve importovanou knihovnu, jak je znázorněno na obrázku *Přidáni knihovny do seznamu React knihoven*.

```
protected List<ReactPackage> getPackages() {
   List<ReactPackage> packages = Arrays.<ReactPackage>asList(
      new RNCMaskedViewPackage()
   );
   return packages;
}
```

Obrázek 16 Přidáni knihovny do seznamu React knihoven

Také je potřeba zmínit, že teď zodpovědnost za obsah souboru *android/app/src/main/AndroidManifest.xml* leží zcela na vývojáři, nejdůležitějšími řádky pro vývojáře budou řádky typu *<uses-permission android:name="... "/>*. Tyto řádky jsou zodpovědné za práva, která vaše aplikace bude schopna vyžadovat od uživatele.

Například řádek *<uses-permission android:name="android.permission.CAMERA" />* umožní aplikaci vyžadovat právo na používaní kamery a pak následně její používání.

5.2.3 Problém s nepodporovanými knihovnami

Jak již bylo zmíněné dříve *ViroReact* podporuje maximálně verze *React-Native* 59.9, což vedlo k problému, že dvě knihovny, které byly využívány ve stávající mobilní aplikaci, způsobily problémy při spuštění aplikace. Jde o knihovny *react-native-paper* a *react-native-elements*. Naštěstí tento problém nebyl tak zásadní, šlo jen o vlastnosti *accessibilityRole*. Tato vlastnost je součástí Accessibility[24] prostředí *React-Native*. Toto prostředí se využívá pro implementaci pomocných technologií pro ovládání aplikace. Například čtení textu z obrazovky. Vlastnost AccessibilityRole říká vývojáři, o jaký typ komponenty se jedná. Například *accessibilityRole* může mít role *button*.

[24] Accessibility (přístupnost) – převzato z [20]

V našem případě problém spočíval v tom, že vlastnost accessibilityRole byla ve verzi *React-Native* 60.0 doplněna o další role, jako například checkbox a progressbar. Tyto role nejsou implementované v *React-Native* verze 59.9. Vzhledem k tomu, že daná aplikace neobsahuje žádné funkce, které využívají vlastnosti accessibilityRole, bylo rozhodnuto změnit ve stažených knihovnách nepodporované role na „*none*“.

Také při ladění byl nalezen problém s balíčkem *metro-config*. Tento balíček odpovídá za takzvané balení Javascriptu. Celý JavaScript kód projektu převádí do jednoho souboru a také dělá převod JavaScript funkce, které nepodporuje zařízení do JavaScript funkce, které zařízení podporuje [24]. Jednalo se o proměnnou, která v sobě měla regulární výraz. Tento výraz *Node.js* ve verzi vyšší než 12.11.0 nepodporuje, proto bylo potřeba jej změnit také. Pro zachování změn v knihovnách byla použita knihovna *patch-package*.

5.3 Stahování dat ze serveru

Jelikož stahování ze serveru již bylo implementováno ve stávající mobilní aplikaci *EduArd* a je to v komponentě *DownloadScene*, bylo rozhodnuto doplnit tuto funkci o možnost stahování potřebných dat pro zobrazení rozšířené reality.

Vzhledem k tomu, že pro stahování scénářů ze serveru *EduArd* je potřeba mít odlišný *Access token*[25] od tokenu, který se používá pro stahování virtuálních stezek, a ještě k tomu dynamicky generovaný při přihlášení uživatele do systému *EduArd*, byla potřeba implementovat také logiku pro získávání tohoto tokenu před stahováním scénáře.

Pro stahování dat byli použité následující koncové body.

- **POST - /api/Auth/SignIn** – Vrací *Access token*, který se používá pro stahování scénáře.
- **GET - /api/Scenarios/{scenarioId}/download** – Vrací JSON soubor, který popisuje scénář s identifikátorem **scenarioId**. Tento identifikátor dostane aplikace ze drive stáhnuté virtuální stezky.
- **GET - /api/ScenarioAssets/** - Vrací seznam X3D modelů, obrázků a videa s příslušnými koncový body pro stáhnutí.
- **GET /api/ScenarioAssets/{scenarioAssetId}/download** - Stáhne soubor podle identifikátoru **scenarioAssetId.** Tento identifikátor dostane aplikace ze drive stáhnutého seznamu souboru.

[25] Přístupové tokeny umožňují klientům bezpečně volat chráněná webová rozhraní API a používají se k ověřování a autorizaci webovými rozhraními API

5.4 Vypočet vzdálenosti do scény

Prvním krokem pro vypočet vzdálenosti, je nalezeni skutečně polohy uživatele. Pro zjišťování polohy uživatele byla použita GPS lokalizace a to pomocí knihovny *expo-location*[26] která je modulem frameworku *Expo*. Vybraná knihovna, umožňuje definovat, jak často chceme aktualizovat polohu uživatele. Vzhledem k tomu, že průměrná rychlost chůze člověka je 1.5 metry za vteřinu a přesnost GPS zhruba 5 metrů, interval aktualizací polohy je nastaven na 3 vteřiny. Menší interval by neměl tak velký vliv na pohodlí používaní aplikací, ale naopak by rychleji vybíjel baterii mobilního zařízení.

Druhým krokem je vypočet vzdálenosti dvou bodů, kde prvním bodem je skutečna poloha uživatele a druhým bodem je pozice scény, která daná GPS souřadnicemi. Pro vypočet vzdáleností v dane aplikace implementován *Vzorec 1* Haversinuv vzorec [27]. Který umožnuje spočítat nejkratší délku mezí dvěma body na kulové ploše.

$$\Delta\varphi = \varphi 2 - \varphi 1$$

$$\Delta\lambda = \lambda 2 - \lambda 1$$

$$A = sin2(\Delta\varphi/2) + cos\varphi 1 \cdot cos\varphi 2 \cdot sin2(\Delta\lambda/2)$$

$$B = 2 \cdot arctg2(\sqrt{a}, \sqrt{1-a})$$

$$C = R \cdot B$$

Vzorec 1 Haversinuv vzorec

Kde Δφ je rozdíl dvou šířek bodů φ2 a φ1, Δλ je rozdílem zeměpisných délek dvou bodů λ2 a λ1. R je poloměr Země, který se rovna 6371 a C je výsledná vzdálenost dvou bodů v kilometrech.

5.5 Načítaní scénáře

Načítání scénáře probíhá v již existující komponentě *QuestionSlides* podle identifikátoru scénáře, který je uložen v popisu virtuální stezky. Jelikož scénář je uložen ve formátu *JSON*, ho načítaní probíhá za vyžitím standartní funkce *JSON.Parse* jazyku JavaScript. Tato funkce vrací objekt, který odpovídá hierarchii scénáře viz *, v rámci své bakalářské přací, implementoval* .

Po načtení scénáře aplikace spouští výpočet vzdálenosti od skutečně polohy uživatele do scén, které jsou definované ve scénáři. Jestliže je vzdálenost do nejbližší scény vice než 30 metrů, uživateli bude zobrazena vzdálenost do nejbližší scény a její název. V jiném případě se uživateli zobrazí tlačítko pro spuštění nejbližší scény v rozšířené realitě. Po kliknutí na aplikaci načte vybranou scénu a vykreslí jí.

[26] Knihovna pro práce s GPS – převzato z [33].
[27] Haversinuv vzorec – převzato z [34].

5.6 Popis použitých komponent ViroReact

V této sekci jsou krátce popsané komponenty frameworku ViroReact které byly použity v rámci implementace rozšíření [25].

- **ViroARSceneNavigator** – Reprezentuje kořenový uzel pro zobrazeni scén v rozšířené realitě. Obsahuje v sobě základní nastavení kvality vykreslených modelů a zapnutí grafických efektů.
- **ViroARScene** – Reprezentuje uzel samotné scény v rozšířené realitě. Obsahuje v sobě různé nástroje pro kontrolu scény.
- **ViroOmniLight –** Reprezentuje světelný objekt, který má svoji vlastní polohu a září ve všech směrech, jehož síla se vzdáleností klesá.
- **ViroAmbientLight –** Reprezentuje světelný objekt, který vyzařuje do okolí světlo, které působí na všechny modely stejně.
- **ViroNode –** Reprezentuje prázdný uzel, který má svou vlastní polohu, rotaci a velikost.
- **ViroImage –** Reprezentuje uzel pro zobrazení obrázku v rozšířené realitě.
- **ViroVideo –** Reprezentuje uzel pro zobrazení videa v rozšířené realitě.
- **ViroText –** Reprezentuje uzel pro zobrazení textu v rozšířené realitě.
- **ViroMaterials –** Reprezentuje komponentu, která se používá pro vytváření materiálů.
- **ViroAnimations –** Reprezentuje komponentu, která se používá pro vytváření animací.
- **ViroGeometry –** Reprezentuje uzel pro vykreslování vlastní definované geometrie. Zápis geometrie v tomto uzlu odpovídá zápisu *Indexed-Face-Set*.
- **ViroBox –** Reprezentuje krychle.
- **ViroSphere –** Reprezentuje koule.
- **ViroText –** Reprezentuje uzel pro zobrazení textu v rozšířené realitě.

5.7 Vykresleni a načtení scény

Načtení scény začíná v komponentě *ArViewer*. Po kliknutí na tlačítko, do této komponenty přejde popis scény, která má být vykreslena, její identifikátor a také identifikátor příslušné stezky. Komponenta *ArViewer* slouží jako počáteční bod pro zobrazení rozšířené reality. Do té komponenty byla vložena komponenta *ViroARSceneNavigator*. V dané komponentě bylo provedené základní nastavení. Tyto nastavení jsou zobrazený v tabulce *Tabulka 1 Nastaveni* Komponenty ViroARSceneNavigator.

HDR[28]	Zapnuto
PBR[29]	Zapnuto
Multisampling[30]	Zapnuto
Kvalita videa	Vysoké
Stíny	Zapnuto
Počáteční scéna	ArScene

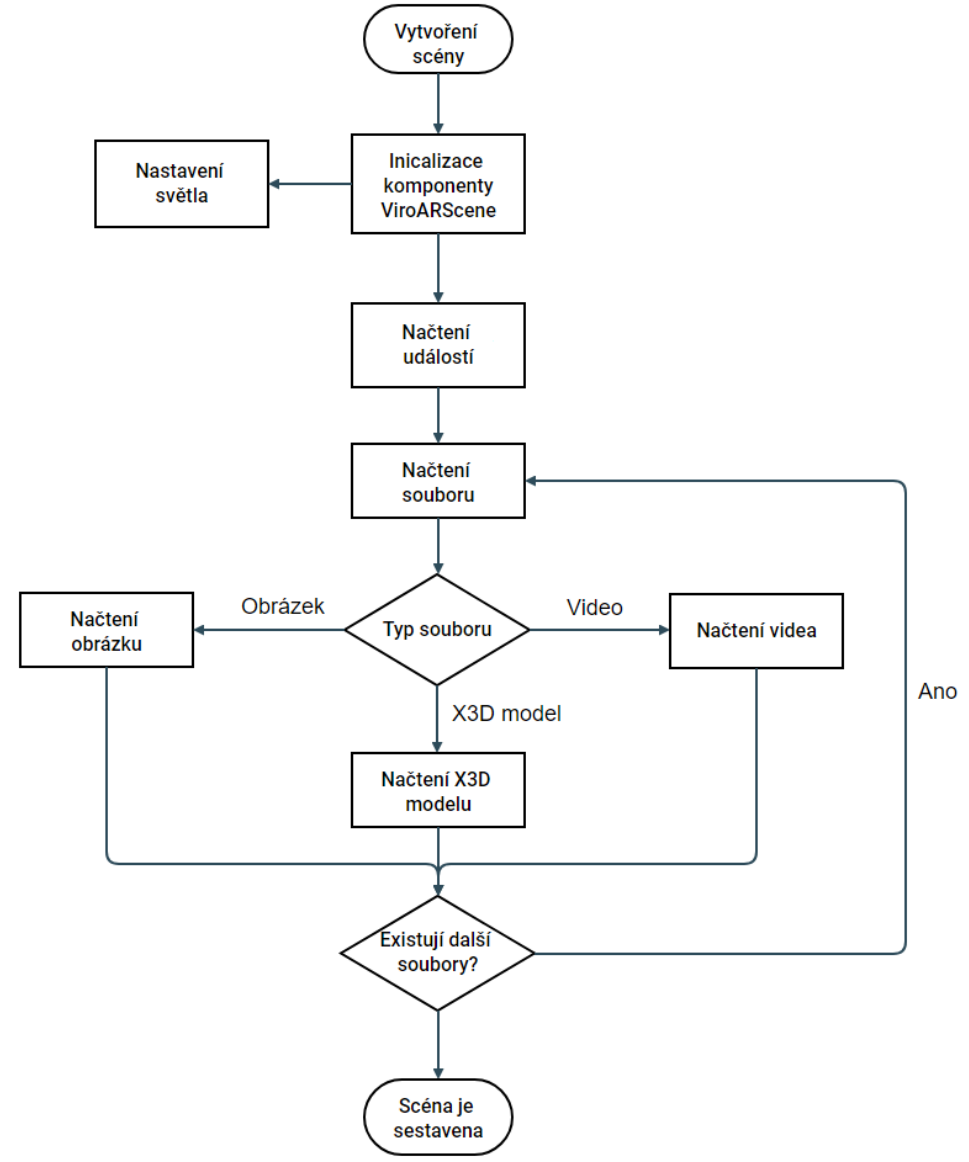
Tabulka 1 Nastaveni Komponenty ViroARSceneNavigator

Komponenta *ArScene* slouží pro kompletní sestavení scény a následně vykreslování. Do této komponenty z předchozí komponenty *ArViewer* přejde popis scény, identifikátory scény a stezky.

Prvním krokem je inicializace komponenty *ViroARScene*. Po inicializaci, aplikace dostane ukazatel do vnitřního stavu scény, který se používá pro zjišťování polohy uživatele v rámci scény. Druhým krokem je nastavení světla ve scéně. Třetím krokem je načtení událostí. Posledním krokem je načtení souborů pro zobrazení v rozšířené realitě. Jedná se o soubory obrázků, videa a 3D modelů ve formátu X3D.

Diagram sestaveni scény je znázorněn na obrázku *Obrázek 17* Sestavení scény.

[28] High Dynamic Range Imaging – převzato z [35].
[29] Physically-Based Rendering - převzato z [36].
[30] Multisample anti-aliasing – převzato z [37].



Obrázek 17 Sestavení scény

Sestavení scény může trvat nějakou dobu, proto uživateli bude zobrazen text, že probíhá načtení scény. Pro tento účel byla využita komponenta *ViroText*. Pokud sestavení scény proběhlo úspěšně, uživateli se zobrazí obsah této scény.

5.7.1 Inicializace komponenty ViroARScene

Inicializace *ViroReact* komponenty *ViroARScene* se provádí automaticky. Ale důvodem zmínky o této inicializaci je to, že musíme tento proces sledovat, abychom po inicializaci získali referenci na vnitřní stav scény. Tato reference slouží v rámci implementace rozšíření pro zjišťování polohy uživatelé v rámci samotné scény a je to pomoci *ViroReact* funkce *getCameraOrientationAsync()*.

5.7.2 Nastaveni světla

Pro nastavení světla byly využity komponenty *ViroOmniLight* a *ViroAmbientLight*. Kombinace těchto dvou komponent dává nejlepší výsledek podle subjektivního názoru autora této práce. Nastavení komponenty *ViroOmniLight* je ukázáno v tabulce *Tabulka 2 Nastavení* komponenty ViroOmniLight.

Barva	Bila
Počáteční útlumová vzdálenost	1
Intenzita	2000
Koncová útlumová vzdálenost	70
Poloha ve 3D prostoru	X: 20 - Y: 20 - Z: 20

Tabulka 2 Nastavení komponenty ViroOmniLight

Nastavení komponenty *ViroAmbientLight* je ukázáno v tabulce *Tabulka 3 Nastavení* komponenty ViroAmbientLight.

Barva	Bila
Intenzita	50

Tabulka 3 Nastavení komponenty ViroAmbientLight

5.7.3 Načtení události

Události slouží k interakci v rozšířené realitě, viz sekce *3.4.3* Interakce. Vzhledem k tomu, že způsob popisu stáhnutých událostí není vhodný pro další implementaci rozšířeni, bylo rozhodnuto tento zápis mírné modifikovat.

Hlavním cílem této modifikace je usnadnit přístup k potřebným položkám při implementaci načtení X3D modelu. Pro tyto účely byla implementována funkce, která vrací události a její položky ve formě slovníků, kde kořenový slovník má za klíč název modelu.

Tato funkce akceptuje jen události, které spouští akce ze vzdálenosti, kvůli tomu, že autor této práce počítal jen s tímto typem události při implementaci rozšíření. Při implementaci dané funkce byl také kladen důraz na možné další rozšíření, proto při budoucím vývoji bude snadné doplnit tuto funkci o podporu dalších typů událostí.

5.7.4 Načtení obrázku a videa

Vzhledem k tomu, že stávající web aplikace pro tvorbu scén *EduARd* nepodporuje přidání polohy a rozměru k obrázkům a videu, bylo potřeba vymyslet způsob jak tento problém obejít. Výsledným řešením bylo použití objektu *Locator* pro definování polohy obrázku či videa a objektu

Annotation, který obsahuje v sobě název souboru, rozměr, případnou rotaci, a je vložen do vybraného *Lokátoru*. Údaje, které jsou vložené do objektu *Annotation,* jsou zapsané v *JSON* formátu. Popis objektů *Locator* a *Annotation* jsou v sekci .

Kvůli tomu, že se *Locator* také používá pro sdružení modelů ve scéně, bylo rozhodnuto pojmenovat *Lokátory* podle jejich účelu. Přiřazení zkratek na konci nazvu *Lokátoru* podle jejich účelu je znázorněno v tabulce *Tabulka 4 Zkratky* podle účelu Lokátorů.

Účel Lokátoru	**Zkratka na konci názvu Lokátoru**
Zobrazeni scény obsahující X3D modely	_scene
Zobrazení obrázku	_picture
Zobrazení videa	_video

Tabulka 4 Zkratky podle účelu Lokátorů

Názvy jednotlivých položek a jejich formát, které jsou ukládané v objektu Annotation, jsou znázorněný v tabulce *Tabulka 5 Formát zápisu položek v* objektu Annotation.

Název položky	**Formát**	**Povine k vyplnění**
name	Libovolný řetězec	Ano
type	Řetězec (*image* nebo *video*)	Ano
scale	Pole formátu [x, y, z]	Ano
rotation	Pole formátu [x, y, z]	Ne

Tabulka 5 Formát zápisu položek v objektu Annotation

Pro zobrazení obrázků byla využita komponenta *ViroImage*. Do této komponenty se předá rozměr obrázku, cesta do souboru a rotace. Pokud rotace nebyla definovaná, pak položka rotací bude vyplněna jako *billboard*. Tato definice říká, že obrázek musí být vždy otočen k uživateli v nezávislosti na ho poloze.

Pro zobrazení videa byla využita komponenta *ViroVideo*. Tato komponenta má stejný způsob definovaní položek jako u komponenty *ViroImage*, ale navíc k tomu obsahuje položku *loop*. Tato položka definuje zda video má být spuštěno. Vzhledem k tomu, že nejsou implementovány interakce pro zapnutí či pozastavení videa, tato vlastnost je zapnuta.

Tyto dvě komponenty jsou obaleny do *ViroReact* komponenty *ViroNode*. V této komponentě je definovaná jejích poloha v rámci scény.

Přiklad zobrazeni obrázku v rozšířené realitě je ukázán na obrázku *Obrázek v rozšířené realitě*.

Obrázek 18 Obrázek v rozšířené realitě

5.7.5 Načtení souboru X3D

Vzhledem k tomu, ze ViroReact nepodporuje načítaní modelů ve formátu X3D, bylo potřeba implementovat vlastní načtení vybraného formátu do struktur frameworku. Pro tyto účely byl implementován rekurzivní algoritmus, který projde všemi uzly v souboru X3D a namapuje tyto uzly do *ViroReact* komponent *ViroNode* a *ViroGeometry*. Tento způsob umožní zachovat hierarchie vnitřních uzlů souboru X3D. Tento algoritmus byl implementován v komponentě *ModelToView*, která přijímá za parametry cestu k souboru X3D a ukazatel do scény a také příslušné tomuto modelu události, pokud existují.

Nejdůležitějším X3D uzlem je *Shape*. Tento uzel obsahuje další pod uzly které definují materiál a geometrie. Kde je geometrie zapsaná buď ve tvaru *Indexed-Face-Set* nebo pomocí geometrických primitiv. Geometrické primitivy, které podporuje X3D jsou koule, cylindry, kvádry, kužely a text.

Pro první způsob zápisu geometrie *Indexed-Face-Set* byla použita komponenta *ViroGeometry*, kde způsob ukládaní geometrie je stejný. Pro geometrické primitivy typu kvádr a koule byli využity *ViroReact* komponenty *ViroBox* a *ViroSphere*. Jelikož primitivy typu cylindr a kužel nejsou definované ve frameworku *ViroReact*, bylo potřeba tyto primitivy vytvořit ručně pomocí komponenty *ViroGeometry* a uložit je do vlastních komponent.

Schematické mapovaní jednotlivých uzlů je ukázáno v tabulce *Mapovaní uzlů* X3D *do ViroReact* komponent.

X3D	**ViroReact**
Scene	ViroNode
Transform	ViroNode
Group	ViroNode
Shape – IndexedFaceSet	ViroGeometry
Shape – Sphere	ViroSphere
Shape – Box	ViroBox
Shape – Cylinder	ViroGeometry
Shape – Cone	ViroGeometry
Shape - Text	Neimplementováno

Tabulka 6 Mapovaní uzlů X3D do ViroReact komponent

Přiklad komponenty *ViroGeometry* je na obrázku *Obrázek 19 Přiklad* komponenty ViroGeometry

```
<ViroGeometry
        position={[-2.5, 0, -0.1]}
        scale={[1.0, 1.0, 1.0]}
        materials="someMaterial"
        vertices={[ [1, 0, 0], [0, 1, 0], [-1, 0, 0] ]}
        normals={[ [0, 0, 1], [0, 0, 1], [0, 0, 1] ]}
        triangleIndices={[ [0, 1, 2 ] ]} />
```

Obrázek 19 Přiklad komponenty ViroGeometry

Příklad modelu který byl nahrán ze formátu X3D do *ViroReact* je na obrázku *Model v rozšířené realitě*.

Obrázek 20 Model v rozšířené realitě

5.8 Tvorba a přirazení materiálů

Pro tvorbu materiálů se využívá *ViroReact* funkce *ViroMaterials.createMaterials()*. Tato funkce přijímá jména materiálů a jejich vlastnosti. Tyto jména pak se používá pro přiřazení vytvořených materiálů do komponenty *ViroGeometry*.

Vlastností materiálů jsou znázorněný v tabulce *Vlastností materiálů.*

X3D	**ViroReact**
DiffuseColor	Rgb (DiffuseColor)
Shininess	Shininess
LightingModel – Phong	LightingModel – Phong
AmbientIntensity	Nepodporuje
EmissiveColor	Nepodporuje
SpecularColor	Nepodporuje
Transparency	Neimplementováno

Tabulka 7 Vlastností materiálů

5.9 Tvorba a přiřazení animací

Pro tvorbu animací se využívá *ViroReact* funkce *ViroAnimations.registerAnimations()*. Tato funkce přijímá jména animací a jejích vlastností. Tyto jména pak používá pro přiřazení vytvořených animací do komponent *ViroNode* a *ViroGeometry*. Seznam animace je znázorněn v tabulce *Tabulka 8* Typy animaci.

X3D	ViroReact
Animace rotace	Implementováno
Animace posice	Implementováno
Animace barvy	Implementováno
Animace rozměru	Není implementováno

Tabulka 8 Typy animaci

5.9.1 Animace pozic

Pro implementaci animace pozic byla implementována řetězcová animace, to znamená, že pro každý krok změny animace byla vytvořena vlastní animace. Seznam těchto animací byl přirazen do takzvané řídicí animace, která obsahuje seznam animací, které budou prováděny sekvenčně. Mapovaní pozic je přímočaré. Vlastnost *easing* je nastavena na *Linear*. Tato vlastnost říká, že animace má běžet celou dobu se stejnou rychlosti. Tento typ animace je možné přidat do komponent *ViroNode* a *ViroGeometry*.

Příklad vytvoření animací je uveden na obrázku *Vytvořeni animace*.

```
ViroAnimations.registerAnimations({
  moveRight: {
     properties: {
        positionX: 1,
        positionY: 0,
        positionZ: 0
        }, easing: “Linear”, duration: 1000},
  moveLeft: {
     properties: {
        positionX: 0,
        positionY: 0,
        positionZ: 0
        }, easing: “Linear”, duration: 1000},
  moveModel:[["moveRight"], ["moveLeft"]]
});
```

Obrázek 21 Vytvořeni animace

5.9.2 Animace materiálů

Pro implementaci animace materiálů byl použit stejný princip jako u předchozího typu animací. Tento typ animací se dá přidat do komponenty *ViroGeometry*. V současnou chvíli podporuje pouze změnu barvy.

5.9.3 Animace rotací

Pro implementaci animace rotací byl použít stejný princip jako u prvního typu animace. Při implementací bylo zjištěno, že při přímočarém převodu konvertované rotace z modelu ve formátu X3D do sktruktur *ViroReact* dochází ke špatnému chování otáčení. Špatné otáčení spočívá v tom, že se model začíná otáčet o 360 stupňů podle osy, na kterou byla aplikovaná rotace.

Problém se podařilo částečně vyřešit napsáním vlastního algoritmu, který mění formu rotace. Funguje to tak, že místo definice úhlu, kde musí animace rotace končit, je definován úhel, o který se má otočit část modelu, aby rotace skončila v definovaném úhlu. Vzhledem k tomu, že se může animace rotace otáčet jedním směrem a pak opačným, bylo potřeba uvažovat tím směrem, aby algoritmus mohl detekovat, zda se následující hodnota otáčí stejným směrem jako předchozí. Jestliže hodnota jde opačným směrem, algoritmus vytvoří dodatečnou takzvanou inverzní animaci, která kompenzuje předchozí změnu rotace.

Bohužel výše uvedený způsob nefunguje úplně správně. Na některých modelech daná animace ukazuje otočení nesprávně, což vede k závěru, že algoritmus není navržen správně.

5.9.4 Přirazení animace

Jak již bylo zmíněno výše, animace typu změna pozice a změna rotace se dá přidat ke komponentám *ViroNode* a *ViroGeometry*, animace typu materiál se dá přidat pouze do komponenty *ViroGeometry*. I když jsou vytvořené animace definovány jako řetězcové animace, je potřeba přidat pouze řídicí animace do vybraných komponent.

Příklad přirazení animace je ukázán na obrázku *Přirazeni animaci do komponenty ViroNode*.

```
<ViroNode
    position={[0, -1, -2]}
    scale={[.1, .1, .1]}
    animation={
       {
          name: 'moveModel',
          run: true,
          loop: true

       }

    }
 />
```

Obrázek 22 Přirazeni animaci do komponenty ViroNode

5.9.5 Spouštění animace

Animace na modelech se spouští podle kritérií, které jsou definované v sekci *3.4.3* Interakce. V rámci implementaci dané práce, byla vytvořena logika pro spuštění animace podle vzdálenosti od modelu v rámci samotné scény.

Při spuštění scény je pozice uživatele v rámci scény definovaná stejně jako počátek scény. Po průchodu scénou uživatel mění svoji pozici v rámci samotné scény a tato pozice se dá zjistit pomocí funkce *getCameraOrientationAsync(),* která je dostupná z reference scény, kterou aplikace získala při její inicializaci. Interval získávání současné pozice uživatele je nastaven na 1 vteřinu, protože uživatel v rámci scény prochází malou vzdálenost a přesnost měření vzdálenosti od modelu je vysoká.

Níže je uveden vzorec pro výpočet vzdáleností mezi dvěma body ve trojdimenzionálním prostoru.

$$\text{distance} = \left((x2 - x1) * 2 + (y2 - y1) * 2 + (z2 - z1)\ 2\right) * 1/2$$

Vzorec 2 Výpočet vzdáleností dvou bodů v trojrozměrném prostoru

5.10 Převody vlastností z X3D formátu

Vzhledem k tomu, že ne všechny položky je možné přímo namapovat do struktur frameworku *ViroReact*, bylo potřeba implementovat konverzi těchto položek. Níže jsou krátce sepsány nejdůležitější převody.

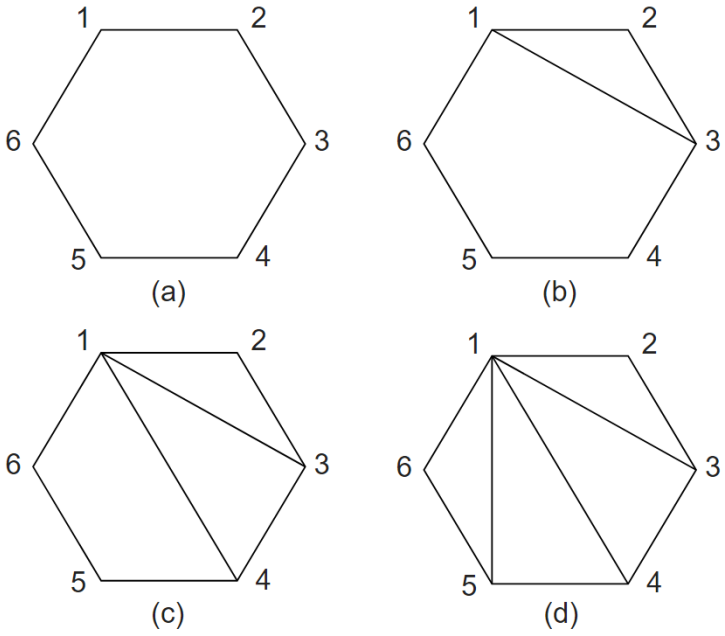
5.10.1 Rotace

Ačkoliv se ve formátu X3D používá zápis rotace ve formátu *axis-angle*[31], ve frameworku se pro zápis rotací používá *Eulerův zápis*[32]. Bylo potřeba implementovat konverzi těchto zápisů. Pro implementaci byla použita JavaScript knihovna *ThreeJs*.

Tato knihovna nepodporuje přímou konverzi mezi zápisem axis-angle a Eulerovým zápisem. Proto byla využita konverze z *asix-angle* do matice transformace a následně do Eulerova zápisu.

5.10.2 Polygonová triangulace

Ve formátu X3D zápis geometrie pomocí *Indexed-Face-Set* může obsahovat libovolný počet souřadnic pro vykreslení polygonů. Nicméně komponenta *ViroGeometry* podporuje vykreslování pouze trojúhelných polygonů. Proto byla potřeba implementovat takzvanou polygonovou triangulaci. Tento proces rozděluje libovolný konvexní polygon[33] na trojúhelníky, které se pak dají použít pro zápis geometrie v komponentě *ViroGeometry*. Proces polygonální triangulace je znázorněn na obrázku *Obrázek 23 Polygonální* triangulace šestiúhelníku.



Obrázek 23 Polygonální triangulace šestiúhelníku

[31] Zápis rotací axis-angle – převzato z [38].
[32] Eulerovy zápis rotace – převzato z [38].
[33] Konvexní polygon – převzato z [39].

Pseduo kód polygonální triangulace šestiúhelníku je znázorněn na obrázku *Pseduo kód polygonální triangulace*.

```
const polygon = [1, 2, 3, 4, 5, 6];
const triangles = [];
for (let i = 0; i < polygon.length; i++) {
   if (i + 2 < polygon.length) {
      const triangle = [polygon[0], polygon[i + 1], polygon[i + 2]];
      triangles.push(triangle);
   }
}

// triangles => [[1, 2, 3], [1, 3, 4], [1, 4, 5], [1, 5, 6]]
```

Obrázek 24 Pseduo kód polygonální triangulace

5.11 Generovaní normál

Normály ve *3D* grafice slouží především pro kalkulací odrazu světla. Modely ve formátu X3D, kde geometrie modelů zapsaná pomocí *Indexed-Face-Set*, nemusí obsahovat definiční normál. Vzhledem k tomu, že komponenta *ViroGeometry* musí mít definované normály, bylo potřeba implementovat generovaní normál pro tuto komponentu.

V komponentě *ViroGeometry* jsou normály seskupeny do trojúhelníků stejně jako vrcholy a souřadnice. Při vykreslování trojúhelníku je normála v každém bodě interpolována z normál definovaných každým vrcholem trojúhelníku.

Pseudokód algoritmu zajišťujícího vytvoření normál je ukázán na obrázku *Pseduo kód generace normál*.

```
const normals = [];
const vertices = [[1, 0, 0], [0, 1, 0], [-1, 0, 0]];
const triangleIndices = [[0, 1, 2]];
for (let j = 0; j < vertices.length; j++) {
  let newNormal = Vector3(0, 0, 0);
  for (let i = 0; i < triangleIndices.length; i++) {
    if (triangleIndices[i].includes(j)) {
      // Get an index of a vertex in the triangleIndices.
      const vertexIndex = getVertexIndex(triangleIndices[i], j);
      let vectorA = Vector3(vertices[indexedFaceSet[i][vertexIndex]]).multiplyScalar(-1);
      let vectorB = Vector3(vertices[indexedFaceSet[i][(vertexIndex + 1) % 3]]);
      let vectorC = Vector3(vertices[indexedFaceSet[i][(vertexIndex + 2) % 3]]);
      let vectorV = vectorB.add(vectorA);
      let vectorW = vectorC.add(vectorA);
      const tempNormal = Vector3().crossVectors(vectorV, vectorW);
      let vectorVAndWLength = vectorV.length() * vectorW.length();
      let sinAlpha = vectorN.length() / vectorVAndWLength !== 0 ? vectorVAndWLength : 1;
      vectorN.multiplyScalar(sinAlpha);
      vectorN.normalize();
      newNormal = newNormal.add(tempNormal);
    }
  }
}
newNormal.normalize();
normals.push(newNormal);
// normals => [[0, 0, 1], [0, 0, 1], [0, 0, 1]];
```

Obrázek 25 Pseduo kód generace normál

6 Testování

Po integraci stávající mobilní aplikaci s frameworkem *ViroReact*, byla ztracena možnost spuštění aplikace pomoci mobilní aplikace *Expo*, protože sestavení aplikace probíhá na počítači vývojáře a ne na serverech Expo. Proto je potřeba mít nainstalované Android SDK. Pro instalaci Android SDK bylo použito volné dostupné vývojové prostředí Android Studio. Pro testovaní bylo použito Android *SDK Tools* verze 26.6.1 a *5.1.4 Node*.js verze 14.16.0.

Vzhledem k tomu, že autor této práce nemá iOS zařízení ani MacOS, tato práce byla otestovaná pouze pro Android zařízeni. Aplikace byla otestovaná na fyzickém zařízení Xiaomi Mi 8.

Jelikož proces spuštění a ladění aplikaci po integraci změnil, je v následující sekci sepsán novy způsob spuštění a ladění aplikace. Poté jsou v následující sekci sepsané možné problémy a jejích řešení.

6.1 Proces spuštění aplikace

Pro spuštění aplikace musí být instalované Android SDK a také nainstalovány balíčky pomocí příkazu „*5.1.5* Node Package Manager *install“*, který provede nejen samotné stahování knihoven, ale také i takzvané *post install* příkazy. V našem případě jde o příkazy „*npx path-package* a *npx jetifier“*. První příkaz změní stažené knihovny podle vytvořených během integrace šablon pro změny, které je možné najít ve složce *patches*. Jde o knihovny *metro-config, react-native-elements, react-native-paper*. Druhý příkaz spustí jetifier, což nám umožní sestavovat projekt bez pomoci prostředí Android Studio.

Před sestavením aplikace, musí být mobilní zařízení propojeno s počítačem přes *USB* kabel. Pro sestavení a následnou instalaci aplikace do mobilního zařízeni v režimu *debug* musíme v kořenové složce projektu vyvolat příkaz „*react-native run-android --variant=gvrDebug“*. Pro sestavení a instalaci aplikace v režimu *release* musíme v kořenové složce projektu zavolat příkaz „*react-native run-android --variant=ArRelease“*.

Při instalaci uživatel musí potvrdit instalaci aplikace na mobilním zařízení a mít zapnutý *developer mod* na zařízení.

6.2 Možné problémy a jejich řešeni

Zde jsou sepsané možné problémy při sestavení a instalaci aplikace s možným řešením. Vzhledem k tomu, že autor této práce vyvíjel aplikaci na systému Windows, řešení sepsaných problémů bude platit pouze pro dané systémy.

- **Sestavení aplikace padá s neznámou chybou**

  Tento problém občas způsobí špatné předchozí sestavení aplikace. Ve většině případů postačí spustit příkaz „*./gradlew clean*“ ve složce *android*. Pokud nic se nezměnilo, je potřeba zkusit vymazat složku *node_module*s a *package-lock.json* a nainstalovat jí znovu.

- **Sestavení aplikace padá s chybou, že *gradle* již má spuštěné úkoly**

  Tato chyba se objeví, když předchozí sestavení a instalace aplikace nebyly řádné ukončené. Ve většině případů postačí vyvolat následující příkaz

```
WMIC PROCESS where "Name like 'java%' AND CommandLine like '%GradleDaemon%'" Call Terminate
```

- **Sestavení aplikace padá s chybou, že mobilní zařízení pro instalaci není nalezeno**

  To znamená, že uživatel mobilního zařízení nemá spuštěný takzvaný *developer mode*. Spuštění tohoto modu pro různé mobilní zařízeni jsou odlišné. Zkuste vyhledat na internetu informaci o tom jak zapnout developer mode na svém smartphonu.

7 Závěr

Cílem této bakalářské práce byla implementace rozšíření stávající mobilní aplikace *EduARd* o podporu rozšířené reality s GPS lokalizací. V první kapitole *1* Úvod byla popsaná motivace a cíle dané bakalářské práce. Ve druhé kapitole *2* Terminologie byly popsány důležité pojmy pro pochopení problematiky rozšířené reality. Dále v kapitole *3 Analýza* byla provedena rešerše již existujících aplikací s podporou rozšířené reality, pote v této kapitole byla provedena základní analýza již existující mobilní aplikace *EduARd*. V následující sekci dané kapitoly byla provedena rešerše dostupných nástrojů pro tvorbu aplikace s podporou rozšířené reality, kde byl vybrán framework *ViroReact*. V poslední sekci *3.4 Analýza scénáře a* scén byla popsaná struktura scénáře a scén. Na základě analýzy v kapitole *4* Návrh byl popsán detailněji framework *ViroReact* a možné problémy s jeho použitím v rámci implementaci a zároveň byly definovány požadavky na systém. V následující kapitole *5* Implementace byla popsána implementační část projektu. V poslední kapitole *6* Testování, byly popsány techniky testovaní a způsoby řešení možných problémů.

Výsledkem této práce je pochopení problematiky vývoje mobilních aplikací s rozšířenou realitou za pomocí frameworku *ViroReact* a implementace rozšíření stávající mobilní aplikace o podporu rozšířené reality. Největšími překážkami k úspěšné implementaci rozšíření považuji integraci frameworků *ViroReact* a *Expo*, a vytvoření algoritmů pro načítaní modelu ve formátu X3D do struktur frameworku *ViroReact*.

V budoucím vývoji by bylo vhodné se zaměřit na úpravu převodu animace rotací a přidání dalších uzlů X3D, které aplikace bude schopna načítat. Jinak je možné konstatovat, že framework *ViroReact* byl dobrou volbou a mocným nástrojem pro tvorbu aplikace s podporou rozšířené reality.

8 Literatura

[1] R. T. A. AZUMA. *Survey of Augmented Reality*, Malibu: Hughes Research Laboratories, 1997.

[2] K. a. K. W.-J. YOUNG-GEUN. *Implementation of Augmented Reality System for Smartphone Advertisements* [Online]. Dostupné z: **http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.446.5970&rep=rep1&type=pdf**. [Přístup získán 22.04.2021].

[3] J. Lanier. *Jaron Lanier* [Online]. Dostupné z: **http://www.jaronlanier.com/**. [Přístup získán 15.06.2021].

[4] Niantic. *Pokémon Go* [Online]. Dostupné z: **https://www.pokemon.com/us/pokemon-video-games/pokemon-go/**. [Přístup získán 5.05.2021].

[5] NianticLabs. *Ingress* [Online]. Dostupné z: **https://www.ingress.com**/. [Přístup získán 5.05.2021].

[6] ARLOOPA Inc. *ARLOOPA* [Online]. Dostupné z: **https://arloopa.com/**. [Přístup získán 3.04.2021].

[7] Google. *ARCore* [Online]. Dostupné z: **https://developers.google.com/ar**. [Přístup získán 16.06.2021].

[8] H. Kato. *ARToolKit* [Online]. Dostupné z: **https://artoolkit.org/**. [Přístup získán 7.06.2021].

[9] ViroMedia. *ViroReact* [Online]. Dostupné z: **https://viromedia.com/viroreact**. [Přístup získán 5.1.2021].

[10] Bc. Domik. Truong. *Georeferencovaná rozšířená realita* [Online]. Dostupné z: **https://dspace.cvut.cz/handle/10467/86066**. [Přístup získán 7.05.2021].

[11] Bc. Michal Mráz. *Webový portál pro přípravu podkladů prorozšířenou realitu* [Online]. Dostupné z: **https://dspace.cvut.cz/handle/10467/92885**. [Přístup získán 1.04.2021].

[12] D. B. Ruth Malan. *Functional Requirements and Use Cases* [Online]. Dostupné z: **http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.436.4773&rep=rep1&type=pdf**. [Přístup získán 18.05.2021].

[13] B. A. N. E. Y. J. M. Lawrence Chung. *Non-Functional Requirements in Software Engineering*. Media New York, 1999.

[14] JetBrains. *IntelliJ IDEA* [Online]. Dostupné z: **https://www.jetbrains.com/idea/**. [Přístup získán 2.04.2021].

[15] Json Org. *Úvod do JSON* [Online]. Dostupné z: **https://json.org/json-cz.html**. [Přístup získán 2.05.2021].

[16] P. Tišnovský. *XML + 3D = X3D* [Online]. Dostupné z: **https: //www.root.cz/clanky/xml-3d-x3d**. [Přístup získán 3.05.2021].

[17] Ryan Dah. *Node JS* [Online]. Dostupné z: **https://nodejs.org/**. [Přístup získán 6.04.2021].

[18] Open Source. *Node Package Manager* [Online]. Dostupné z: **https://www.npmjs.com/**. [Přístup získán 7.05.2021].

[19] Open Source. *GRADLE* [Online]. Dostupné z: **https://gradle.org/**. [Přístup získán 2.04.2021].

[20] Facebook. *React Native* [Online]. Dostupné z: **https://reactnative.dev/**. [Přístup získán 2.06.2021].

[21] Open Source. *Patch Package* [Online]. Dostupné z: **https://github.com/ds300/patch-package**. [Přístup získán 5.03.2021].

[22] Open Source. *Jetifier* [Online]. Dostupné z: **https://github.com/mikehardy/jetifier**. [Přístup získán 7.06.2021].

[23] Open Source. *ThreeJS* [Online]. Dostupné z: **https://threejs.org/**. [Přístup získán 4.04.2021].

[24] Facebook. *Metro Config* [Online]. Dostupné z: **https://github.com/facebook/metro**. [Přístup získán 6.05.2021].

[25] ViroMedia. *ViroReact Documentation* [Online]. Dostupné z: **https://docs.viromedia.com/docs**. [Přístup získán 9.05.2021].

[26] D. L. Brutzman D. *X3D: Extensible 3D Graphics for Web Authors 1st edition*, 2007.

[27] Microsoft. *Microsoft Hololens* [Online]. Dostupné z: **https://www.microsoft.com/cs-cz/hololens/hardware**. [Přístup získán 2.05.2021].

[28] EduARd. *Ekosystem EduARd* [Online]. Dostupné z: **https://eduard.fel.cvut.cz**. [Přístup získán 1.04.2021].

[29] Google. *Android* [Online]. Dostupné z: **https://www.android.com/**. [Přístup získán 1.04.2021].

[30] Apple. *iOS* [Online]. Dostupné z: **https://www.apple.com/cz/ios**. [Přístup získán 2.06.2021].

[31] HW server s.r.o. *Vyvoj HW* [Online]. Dostupné z: **https://vyvoj.hw.cz/kdyz-gps-nestaci-inercialni-mereni-imu-mu-pomuze.html**. [Přístup získán 15.04.2021].

[32] D. Hanák. *Průvodce REST API* [Online]. Dostupné z: **https://www.itnetwork.cz/programovani/nezarazene/stoparuv-pruvodce-rest-api**. [Přístup získán 16.03.2021].

[33] Expo. *Expo Location* [Online]. Dostupné z: **https://docs.expo.dev/versions/latest/sdk/location/**. [Přístup získán 2.05.2021].

[34] A. Upadhyay. *Haversine Formula* [Online]. Dostupné z: **https://www.igismap.com/haversine-formula-calculate-geographic-distance-earth**. [Přístup získán 1.06.2021].

[35] M. Khanna. *HDR Imaging: What is an HDR image anyway?* [Online]. Dostupné z: **https://towardsdatascience.com/hdr-imaging-what-is-an-hdr-image-anyway-bdf05985492c**. [Přístup získán 7.05.2021].

[36] Marmoset. *Physically-Based Rendering, And You Can Too!* [Online]. Dostupné z: **https://marmoset.co/posts/physically-based-rendering-and-you-can-too/**. [Přístup získán 7.05.2021].

[37] Arvilab. *THE MEANING OF ANTI-ALIASING: FXAA, SMAA, MSAA, SSAA, TXAA ALGORITHMS* [Online]. Dostupné z: **https://vr.arvilab.com/blog/anti-aliasing**. [Přístup získán 2.05.2021].

[38] Martin John Baker. *EuclideanSpace Geometry* [Online]. Dostupné z: **https://www.euclideanspace.com/maths/geometry/rotations/axisAngle/index.html**. [Přístup získán 9.04.2021].

[39] M. O. Reference. *Convex Polygon* [Online]. Dostupné z: **https://www.mathopenref.com/polygonconvex.html**. [Přístup získán 3 06 2021].

[40] Wi-Fi Alliance Inc. *Wi-Fi* [Online].
Dostupné z: **https://www.wi-fi.org/**. [Přístup získán 1.04.2021].

Příloha A

Slovník pojmů a zkratek

Pojem / Zkratka	**Vysvětleni**
3D	
QR	
Framework	Softwarový balík usnadňující programování. Má větší rozsah než knihovna. Typicky zapouzdřuje a řeší komplexní problém
JSON	JavaScript Object Notation Jedná se o jednoduchý datový formát. Využívá se pro přenos strukturovaných dat. Oproti XML neobsahuje silné typové definice, díky tomu je přenos dat menší.
SDK	Software development kit – soubor nástrojů pro vývoj software.
PBR	Physically based rendering
REST	Representational state transfer
UCD	User-centered design
VRML	Virtual Reality Modeling Language
X3D	Extensible 3D
XML	Extensible Markup Language
AR	Augmented reality
API	Application programming interface
GPS	Global Positioning System
ČVUT	České vysoké učení technické
BTC	Systém základnových stanic
VR	Virtual reality
SLAM	Simultaneous localization and mapping
HDR	High Dynamic Range Imaging
2D	Dvoudimenzionální

Tabulka 9 Slovník pojmů a zkratek

Příloha B

Seznam X3D uzlů

Následující tabulka uvádí všechny uzly souboru X3D a jejich atributy (popř. hodnoty atributů), které lze v klientské aplikaci načítat.

Uzel	Atributy (hodnoty atributů)
Scene	
Transfrom	DEF, USE, translation, rotation, scale, center
Group	DEF, USE, translation, rotation, scale, center
Shape	
TimeSensor	cycleInterval, loop
Route	fromNode, toNode, toField
OrientationInterpolator	DEF, keyValue, key
PositionInterpolator	DEF,keyValue, key, toField(set_translation, translation)
Normal	vector
Material	DEF, USE, diffuseColor, shininess
Cylinder	radius, height
Coordinate	DEF, USE, point
IndexedFaceSet	coordIndex
ColorInterpolator	keyValue, key

Tabulka 10 Seznam X3D uzlů