VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

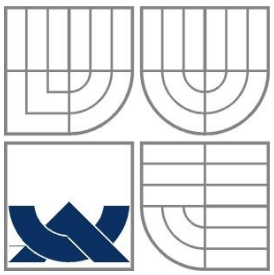
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ROZŠÍŘENÁ REALITA NA MOBILNÍM ZAŘÍZENÍ

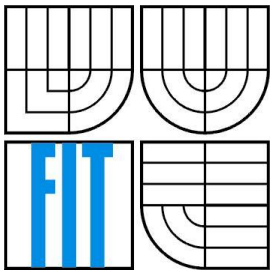
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE MAREK VALENTA
AUTHOR

BRNO 2015



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ROZŠÍŘENÁ REALITA NA MOBILNÍM ZAŘÍZENÍ

AUGMENTED REALITY FOR MOBILE DEVICES

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — MAREK VALENTA
AUTHOR

VEDOUCÍ PRÁCE — Ing. JOZEF KOBRTEK
SUPERVISOR

BRNO 2015

Abstrakt

Bakalářská práce se zabývá návrhem a implementací aplikace, která umožňuje rozšířit realitu pomocí mříže jako hrací plochu pro hru. Hra by měla spočívat v pokládání bloků na tuto plochu. Jsou zde rozebrány možnosti rozšířené reality, vysvětleno proč je použita rozšířená realita pomocí markeru. Obsahuje také postup vypracování práce a částečně popisuje důležité skripty.

Abstract

This thesis deals with design and implementation of an application that let's you expand your reality throught he grid which serve as theplaying surface for the game. The game works on the basis of laying the blocks on this surface. There are analyzed the possibilities of augmented reality, explains why augmented reality is used with a marker. It also contains a description of the work and describes some important scripts.

Klíčová slova

Rozšířená realita, Mobilní zařízení, Android, Unity 3D, Vuforia SDK

Keywords

Augmented reality, Mobile devices, Android, Unity 3D, Vuforia SDK

Citace

Valenta Marek: Rozšířená realita na mobilní zařízení, bakalářská práce, Brno, FIT VUT v Brně, 2015

Rozšířená realita na mobilní zařízení

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Jozefa Kobrtka Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Marek Valenta
10.3.2015

Poděkování

Děkuji mému vedoucímu Ing. Jozefu Kobrtkovi a Ing. Romanu Jaškovi za odborné vedení a podněty, které mi při řešení tohoto projektu poskytli.

Obsah

1 Úvod

Téma Rozšířená realita je v dnešní době velice populární, využívá se ve velkém množství aplikací, např. pro orientaci v reálném světě a pro vykreslení různých předmětů (např. u reklamy vykreslí výrobek, takže si jej může uživatel prohlédnout detailněji). Je to skvělý prostředek, který využije grafické možnosti mobilního zařízení pro zlepšení reality. Zatím jsou tyto technologie ve vývoji a to jak z pohledu softwaru tak i hardwaru. Není zcela možné využít potenciál, který si uživatel představí pod pojmem rozšířená realita.

Hry, které využívají tuto funkci, většinou postrádají kreativitu. Tím je myšleno, že zde není možnost něco budovat ve virtuálním světě. Toto byla pro mne hlavní myšlenka pro tvorbu takovéto aplikace, která bude obsahovat virtuální realitu spojenou s kreativitou neboli možností něco budovat. Nakonec jsem se rozhodl vytvořit aplikaci, která převezme ideu ze slavné hry minecraft a vloží ji do rozšířené reality.

V této práci se podíváme na návrh a implementaci aplikace, která bude schopna vykreslit nám, do obrazu snímaného kamerou, mříž jako hrací plochu. Zaměříme se na technologie a postupy, které jsou potřebné pro dosažení cíleného výsledku. Dále se budeme zabývat ovládáním takovéto aplikace, aby bylo uživatelsky příjemné. Toto bude obsahovat uživatelské rozhraní a správu gest, které jsou možné na systému android. V neposlední řadě budeme implementovat hru, která bude pracovat s touto mříží. Hra bude čerpat inspiraci hry Minecraft. Budeme na naší mříž (hrací plochu) vkládat bloky. Pomocí těchto bloků bude uživateli umožněno poskládat si libovolnou stavbu. Hra by měla být primárně určena dětem, měla by být graficky příjemná, ovládání musí být jednoduché a intuitivní. Aplikace jako celek bude tvořena ze dvou částí. První část tvorba mříže bude oddělena od samotné hry. Tímto způsobem bude zaručena rozšiřitelnost této aplikace. Bude možnost vytvářet na mříž jakoukoli hru, která bude schopna využít tento typ hrací plochy.

Druhá kapitola pojednává o použitých technologiích. Ze začátku kapitoly je to stručné později se však budou věci brát do detailů. Bude zde vysvětleno, proč jsou některé principy a technologie použité a proč jsem nevybral jiné. Pokusím se zde nastínit pro a proti, jednotlivých technologií.

Třetí kapitola ukazuje, jak bude výsledná aplikace vypadat. Popisuje návrh a funkčnost aplikace. Kapitola je rozdělena podle scén tak jak by měli být naprogramovány ve výsledné aplikaci. Budou zde i ukázky aplikace jak bude vypadat.

Čtvrtá kapitola popisuje implementaci návrhu. Vezme návrh jako návod a postupně vše naprogramuje. Jsou zde nastíněny všechny důležité algoritmy, které jsou použity v aplikaci. K této části je vhodné mít znalosti na úrovni technika. Kapitola je rozdělena podle scén, jak jsou naprogramovány v aplikaci. U scén bude ukázka, jak aplikace vypadá pro lepší představu čtenáře.

Pátá kapitola se zabývá testováním aplikace v průběhu programování a na výsledném produktu. Bude vysvětleno, na jaké věkové kategorii byl program testován, jak testování dopadlo a jaké změny jsem uskutečnil na základě testování.

Šestá kapitola bude ukazovat aplikace a programy, které využívají podobné technologie. Aplikace zde budou nejprve popsány a uvedeny jejich positiva a negativa. Poté uvedu rozdíl mezi touto aplikací a již zmíněními aplikacemi.

2 Použité technologie a programy

Na začátek se podíváme na různé technologie, které se v souvislosti s rozšířenou realitou využívají. Použití správných technologii a programů nám umožní a zjednoduší tvorbu aplikace s rozšířenou realitou. Popíšeme si, jak tyto technologie fungují a proč jsou některé vybrány pro implementaci této aplikace.

2.1 Rozšířená realita

Rozšířená realita v tomto kontextu znamená, zjednodušeně, snímání kamerou reálný prostor, do kterého je vložen a vykreslen 2D nebo 3D objekt, který nám nějakým způsobem rozšiřuje realitu o nové informace. V následujících odstavcích si ukážeme různé způsoby, jak se rozšířená realita využívá, budou zde uvedeny jen zjednodušeně a v další kapitole si některé z nich probereme detailně.

Je zde hodně způsobů jak tuto technologii použít. Jedna z nich je orientace ve světě. Existují aplikace, které ukazují co je kolem vás a zobrazují to na mobilním zařízení. Tyto aplikace se využívají s pomocí GPS modulu a elektronického kompasu. Fungují tak, že se na určité souřadnice uloží obrázek, který se má vykreslit. Z GPS a Wi-Fi modulu získá vaší pozici a pomocí elektronického kompasu a akcelerometru zjistí, jakým směrem se díváte. Tímto vytvoří váš pozorovací úhel. Na základě tohoto úhlu určí, jestli vidíte daný bod s obrázkem nebo ne. Pokud ano, spočítá se vzdálenost od tohoto bodu k vám a vykreslí se obrázek nepřímo úměrný této vzdálenosti. Mobilní zařízení musí obsahovat GPS modul (pro určení pozice ve světě), kompas (pro určení směru kam se díváme), G-Senzor (pro zjištění v jaké poloze se mobil nachází) a Wi-Fi modul nebo mobilní internet, který nahrává data ze serveru a pomáhá určit vaší pozici. Bez těchto čtyř modulů není možné takovou aplikaci zprovoznit. Proto jsem se vydal cestou menších HW nároků.[3]

Další z možností je vykreslování na základě rozpoznání předmětu. Tato technologie je složitější než předchozí, z důvodu počtu dat, které nutné zpracovat. Snímaný objekt je zpracováván mnoha způsoby jako např. rozpoznání hran v obraze a kalkulace stínů a viditelnosti. Ze získaných dat je potom vypočtena pravděpodobnost a podobnost s objektem v databázi. Pokud se shoduje v dostatečné míře tak na základě těchto dat vykreslí obrázek do scény. Tento princip je funkční v SDK Metaio ale je zde nutnost vlastnit placenou licenci. Technologie je to relativně nová, ale není funkční tak aby byla vhodná pro použití v naší aplikaci.

Aplikace bude využívat technologii rozšířené reality s pomocí markeru. Tato technologie funguje na principu rozpoznávání v obrazu. V první řadě musíme mít mobil vybavený kamerou pro snímání obrazu. Dále potřebujeme marker. Marker je obraz nebo obrazec vytisknutý nebo zobrazený tak aby byl viditelný na kameře. Aplikace ho musí dokázat rozpoznat v obrazu snímaném kamerou. Jsou zde kritéria, která musí marker splňovat, ty budou uvedeny dále. Pokud aplikace zachytí marker, tak podle jeho velikosti a natočení vykreslí 3D nebo 2D prostor, který jsme připravili. Tento princip zatím není možný bez použití markeru. Byl by možný, pokud bychom měli dvě kamery a snímali prostor 3D. Tím bychom získali vzdálenosti reálného prostoru a na základě toho bychom mohli vykreslit scénu. Díky technologii snímání 3D prostoru funguje Xbox kinect.

Existují tedy různé typy rozšířené reality, o některých se budeme detailněji bavit dále. My budeme využívat technologii, která je zde zmíněna jako třetí, tedy s pomocí markeru. Pro naši aplikaci bude nejvhodnější, protože splňuje požadavky, které jsou nutné pro vytvoření cílené aplikace.

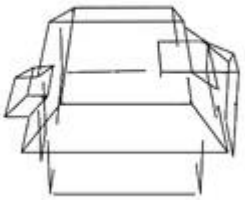
2.2 Využívání senzoru a technologií pro rozšířenou realitu

Pro rozšířenou realitu potřebujeme senzory, které určí jak se má scéna vykreslit. Nejdůležitější ze senzoru je kamera, ta je potřeba ve všech typech. V této kapitole si ukážeme některé technologie a popíšeme je detailněji.

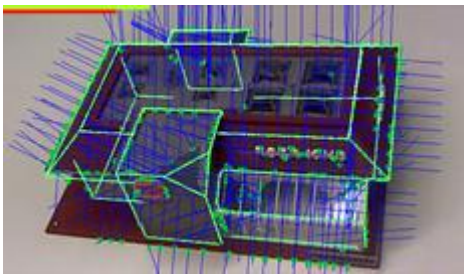
2.2.1 Rozšířená realita bez markeru

Pro to aby scéna byla vykreslena správně, musíme vykreslovat objekt ze stejného místa z jakého je obraz snímán kamerou. Rozpoznání objektů může využívat různé metody. Tyto metody většinou obsahují dvě stádia: sledování a rozpoznání. Jako první je obraz nebo bod zájmu detekován v obraze snímaném kamerou. Sledování může využívat detekci rysů, detekci hran nebo jiné metody zpracovávající obraz pro interpretování obrazu. Využívají se dvě metody pro sledování. První je založena na rysech a druhá na modelu. První se snaží objevit shodu mezi 2D obrázkem rysů a jejich 3D souřadnicemi. Druhá metoda využívá rysy modelů, které jsou vytvořeny např. pomocí CAD programu (obrázek 2.1). Pokud nalezneme shodu mezi 2D obrázkem a 3D obrazem reálného světa (obrázek 2.2), můžeme zjistit pozici kamery pomocí

projekce 3D souřadnic rysů do 2D obrazu. Rozpoznání využívá data z předešlé fáze snímání k tomu, aby byla schopná zrekonstruovat reálné souřadnice v prostoru.[5]



Obrázek 2.1 CAD model [5]



Obrázek 2.2 CAD model namapovaný[5]

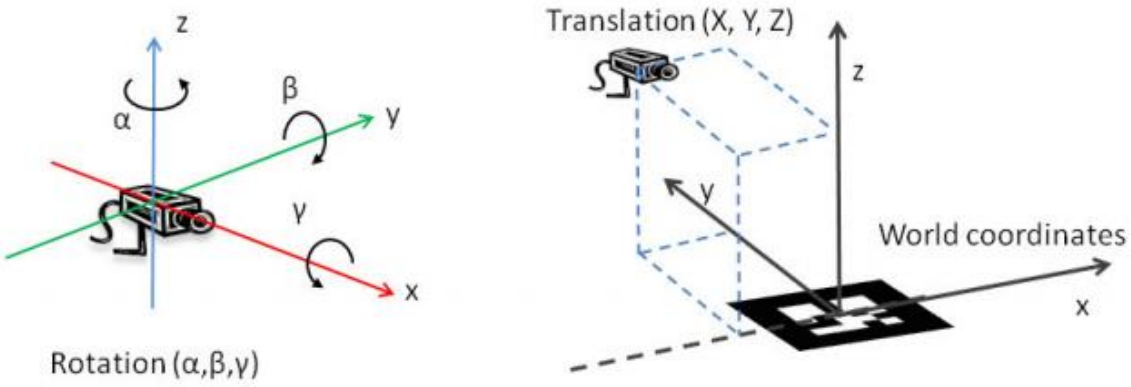
2.2.2 Rozšířená realita s markerem

Hledání markeru probíhá tak, že se snažíme najít ohraničení potencionálního markeru. Po té se snažíme zjistit lokaci rohů v obraze. Dále je u některých důležité zjistit, jestli se jedná o skutečný marker, u kterého dále zjišťujeme identitu. [1]

Základní procedura hledání markeru obsahuje tyto kroky:

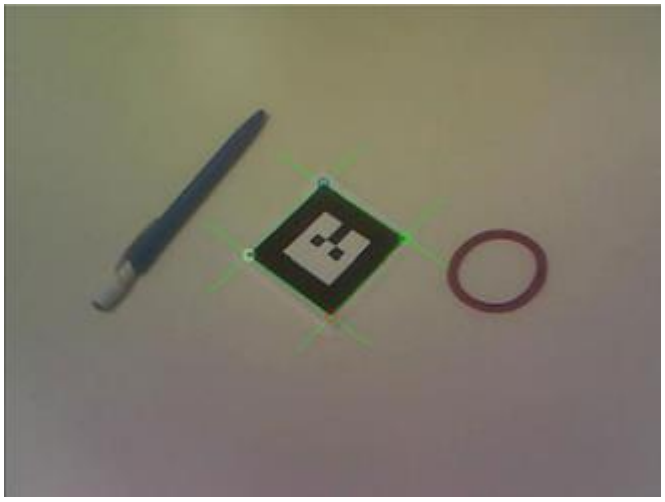
1. Získávání obrazu
    a. Získávání intensity obrazu
2. Preprocesor
    a. Jednoduché zpracování obrazu
    b. Hledání hran (čar)
    c. Detekce ohraničení obrázků
3. Detekce potencionálních markeru a vyřazení obrázků, které jednoznačně nejsou marker
    a. Rychlé vyřazení jednoznačně nevyhovujících markerů
    b. Rychlý test pro potencionální markery
4. Identifikace a dekódování markerů
    a. Zkouška jestli zachycený marker vyhovuje šabloně
5. Kalkulace pozice markeru
    a. Odhad pozice
    b. Iterativní kalkulace přesné pozice

Získání pozice markeru funguje tak, že pozice objektu ukazuje jeho lokaci a orientaci. Lokace může byt vyjádřena pomocí souřadnic (x, y, z) a orientace jako (α, β, γ) okolo tří os.(obrázek2.3)



Obrázek 2.3 ukázka souřadnicových os a orientace a vztah jaký má kamera k markeru[1]

Pozice kamery jde jednoznačně určit z nejméně čtyř koplanárních ale nekolineárních bodů. Tedy systém může spočítat pozici markeru (relativní ke kameře) v 3D soustavě pomocí čtyř rohů markeru (obrázek 2.4).



Obrázek 2.4 nalezení čtyř bodu (rohů) markeru [1]

Pokud marker, objekty i kamera jsou statické rozpoznávání pozice markeru je jednoduché. V případě, že kamera i marker je v pohybu, jsme schopní zjistit pozici kamery a markeru, ale nejsme schopni přesně určit pozici k zemi. Zde je možnost přidání dalšího senzoru např. akcelerometru. [1]

2.3 Unity 3D

Unity 3D je multiplatformní herní engine, který je velice populární pro své ovládání. Je zde možnost zaměřit se při programování na samotnou funkčnost hry. Unity obsahuje pracovní plochu, do které vkládáme objekty. S těmito objekty lze manipulovat (posun, rotace atd..). Dále je zde možnost přidávat jim vlastnosti (komponenty). Není zde nutnost psát kód pro vykreslování v OpenGL, protože to nám zařídí samo Unity.

2.3.1 Komponenty

Jsou to atributy objektů. Tyto komponenty mohou nabývat různých typu např. různé funkce herního enginu a skripty. Funkce herního enginu jsou skripty, které nabízí přímo Unity 3D a slouží např. ke kolizím objektů jejich fyzikálním vlastnostem atd.. Skripty psané uživatelem je možno psát v jazyce C# nebo Java Script. V návrhu a implementaci si ukážeme jak s těmito komponenty pracovat. Co vše je nutné pro to, aby aplikace fungovala.[9]

2.3.2 Hierarchie v objektech

Každý objekt ve scéně může mít hierarchický vztah k jinému objektu. Jsou zde dva vztahy child (potomek) a parent (rodič). Každý potomek může mít právě jednoho rodiče a každý rodič může mít neomezené množství potomků. Tyto vztahy se využívají z důvodu např. transformační závislosti, kdy změna velikosti nebo rotace rodiče ovlivní i potomka, dále posílání zpráv mezi všechny jeho potomky nebo pro přístup ke komponentům potomků. Tyto vlastnosti hierarchie urychlují proces vyhledávání a komunikace.

2.3.3 Překlad na různá zařízení

Unity pracuje s jazykem C# a např. na systém Android se programuje v jazyce Java. Zde uvedu jak je možné že stejný kód napsaný v Unity je spustitelný na různých operačních systémech. Důvodem je open source Software Mono. Tento software je založený na platformě .NET Framework, který dovoluje aby byl program spustitelný na jiných platformách. Funguje to na principu přeložení C# do .NET bytecode. Když pak Unity sestavuje Android aplikaci, tak vkládá .NET bytecode interpretovaný v nativním kódu, založeném na Mono. Při spuštění aplikace se pak spouští tento bytecode.[9]

2.3.4 Důvod výběru Unity 3D

Vybíral jsem mezi třemi herními enginy. Dále uvedu, proč jsem si vybral právě unity. První je Unreal Engine ten umožnuje psaní skriptů v jazyce C# stejně jako Unity. Je zaměřen

hlavně na PC a konzole. Je to engine zaměřen na fotorealistické zobrazení grafiky, které my v naší aplikaci nevyžadujeme. Další unikátní vlastností je přístup ke kódu subsystému tohoto enginu, který se využívá k optimalizaci. Pro naše účely je tento engine, který je zaměřen spíše na rozsáhlé projekty na zařízení, které využijí potenciál, příliš komplikovaný a zbytečný.[8]

Další z řady dostupných enginu je CryENGINE. Tento engine také podporuje programování v jazyku C#. Je to další engine, který se zaměřuje na fotorealistickou grafiku jako Unreal Engine tedy toto je nevýhoda. Je velice složitý na ovládání a není jednoduché v něm vytvořit to, co uživatel potřebuje bez předchozích znalostí. Jako předchozí zmíněný engine je to podle mne velký nástroj a zbytečný pro naše potřeby.[7]

Unity, na rozdíl od předchozích dvou, není tak zaměřený na fotorealistickou grafiku kde zaostává v možnostech renderovací techniky. Ale vyniká v podpoře na mobilní zařízení a webové stránky. Je zde dostatek materiálů k ovládnutí jej na úrovni jakou potřebujeme. Další pro nás důležitý důvod je ten, že Unity na rozdíl od ostatních podporuje kvalitně zpracované Vuforia SDK.

Výběr správného enginu je složitý. Jsou zde zmíněny tři nástroje, z toho dva jsou graficky zaměřené výhradně na PC a konzole. Unity zaostává v grafickém zpracovávání. Všechny jsou zdarma pod podmínkou splnění některých licenčních podmínek.

2.4 Použitý mobilní systém

Program díky Unity 3D, by měl být spustitelný na vícero mobilních systémů nebo počítačových systémů. Výběr máme z IOS, Android, Windows Phone a Blackberry. Každý systém má své přednosti. Naše aplikace bude vyvíjena pod systémem Android.

2.4.1 Android

Je to nejpoužívanější mobilní systém[2]. Je postaven na linuxovém jádru. Díky tomu, že jej vytvořila společnost Google je zde jistota, že aktualizace na tento systém budou pravidelné a také je zde velká podpora vývojářů, kteří nemusí složitě žádat o licenci jako u jiných systému. Dále je zde přístup k velkému množství dokumentací, návodů nebo tutoriálů. V neposlední řadě je možné ho spouštět na jiných zařízeních než mobilní telefon např.: tablet nebo televize.

2.5 Vuforia Augmented Reality SDK

Rozšíření do programu Unity 3D. Umožňuje práci s rozšířenou realitou. Vytvořila ho společnost Qualcomm. Je to platforma využívající kvalitní, stabilní a efektivní počítačové rozpoznávání v obraze. Nabízí několik funkcí, které zpřístupňují mobilním aplikacím a osvobozují programátory od technických omezení.[10]

Vuforia se skládá z různých komponent, jako je Target Managment System, dostupný na stránkách výrobce, Cloud Target Database a Device Target Database a Vuforia Engine. Programátor jednoduše nahraje obraz, jenž chce využit jako obraz, který se bude hledat. Takto nahraný marker může být zpřístupněn pomocí mobilní aplikace skrz odkaz na cloud nebo přímo nahrán z lokálního uložiště. [10]

Vuforia SDK aplikace se skládá z kamery, která sejme obraz a pošle ho do vyhledavače, obrazový konvertor jednoduše konvertuje obraz sejmutý kamerou do formátu, který je vhodný pro OpenGL ES (mobilní verze OpenGL) vykreslování a pro interní sledování. Vuforia obsahuje vyhledavač, který může nahrát a aktivovat více datasetů (data, které jsou vygenerovány pomocí Target Manageru a obsahují obrázky, které se budou sledovat tedy markerů) najednou. To znamená, že je možnost sledovat více markerů zároveň. Vyhledavač obsahuje algoritmy pro detekci a sledování objektů ve snímku zachyceným kamerou.[10]

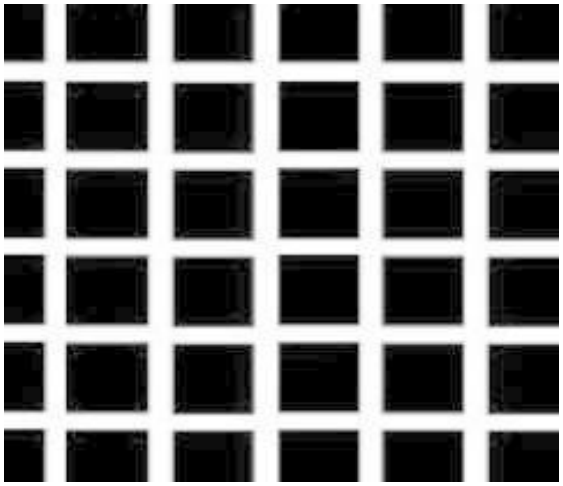
Předností tohoto SDK je rychlá lokální detekce markerů, které dokáže sledovat až 5 najednou, dále efektivní sledování při nedostatku světla nebo částečného zakrytí markeru.[10]

Extended Tracking je technologie, která umožnuje vykreslovat objekty i přestože v záběru již není marker. Funguje na principu snímání okolí markeru. Po prvním rozpoznání a zachycení se načte i mapa okolí. Na základě této mapy se vykreslují objekty, pokud marker není v obraze. Není to zcela dokonalé a je to HW náročné, protože po každé se musí tato mapa znova vytvářet. Je zde možnost zapnutí persistentního módu, který dovoluje načíst okolní mapu jen jednou. Dále má programátor možnost v persistentním módu resetovat tuto mapu. Což může být využito při uspání aplikace a znovu načtení nebo také pro šetření zdrojů HW.[4]

2.5.1 Marker

Toto SDK umí pracovat s markery, které jsou reprezentovány jako 2d obrázky s texturou, která musí vyhovovat pravidlům. Nesmí se opakovat nějaký vzor (obrázek 2.5),

nesmí být jednobarevný. Nejlepší obrázek je takový (obrázek 2.6), který je různorodý a dá se jednoduše poznat jak je otočený. Je to důležité pro to abychom poznali jakým směrem vykreslit scénu. [4]



Obrázek 2.5- špatný vzor

Obrázek 2.6 – dobrý vzor

2.5.2 Důvod výběru

Oproti ostatním SDK pro rozšířenou realitu je tento zcela zdarma. Neobsahuje žádné watermarky a má kvalitní podporu Unity 3D. Na druhou stranu dokumentace nebyla úplně čitelná a naučení tohoto SDK je náročnější než u některých placených verzí. Vyzkoušel jsem více SDK např. Metaio SDK, které bylo zpoplatněno, a verze zdarma nebyla dostačující pro mé potřeby.

Pro naše účely využijeme Vuforia SDK a 2D Tracking (hledání 2d markeru v prostoru) a následného vykreslení scény. Pro testovací účely spustíme i Extended Tracking.

2.6 TouchScript

Další rozšíření do Unity 3D. Slouží k rozpoznání a práci s gesty na mobilním zařízení. Obsahuje základní gesta, jako je roztahování nebo stahování dvěma prsty, posunutí pomocí dvou prstů a rotaci pomoci dvou prstů. Tyto gesta je možné různě modifikovat, spojovat a docílit výsledku jaký chcete. TouchScript Obsahuje funkce pro práci s multidotykovými gesty. Tyto funkce budeme potřebovat, protože k ovládání aplikace budeme využívat výhradně gesta, která budou přepínatelná pomocí tlačítek na obrazovce. [6]

2.6.1 Důvod výběru

Je zde více doplňků do unity, které dokáží zpracovávat gesta jako např. Simple Touch Camera Script, který ale nepodporuje zachytávání gest přímo na objekt. Místo toho všechny gesta zachytává na celé obrazovce, což nevyhovuje našim potřebám. Kdežto TouchScript funguje na principu vrstev (layers). Funguje to tak, že si vytvoříte vrstvy a na každé vrstvě mohou fungovat různá gesta. Pro náš projekt je to důležité protože jednak máme gesta, které ovládají pohyb mříže, ale také vytváříme bariéry, které se vytváří v bodu dotyku ve scéně a ne přímo na obrazovce. Po naprogramování jsem ale došel k závěru, že tento nástroj má opravdu nečitelnou dokumentaci, takže naprogramovat něco pomocí této knihovny je opravdu těžké. A do příště bych asi zvolil jiný nástroj.

3 Návrh aplikace

Zde si rozebereme návrh aplikace. Ukážeme si zde, jak fungují jednotlivé funkce a co všechno by naše aplikace měla splňovat. Důraz zde bude kladen na Graphical User Interface (dále jen GUI). Kapitola bude rozdělena do scén, jak by to mělo být naprogramováno v Unity 3D. U každé scény by měl být uveden obrázek, aby si čtenář mohl představit, o čem se zde píše. V další kapitole si zde uvedené věci uvedeme také, ale tam se budeme zabývat implementací.

3.1 Scéna 1. Menu

První scéna aplikace a jako taková musí zahrnovat rozcestí pro všechny části aplikace. Žádnou jinou funkčnost tato scéna nepotřebuje. Budou zde tři tlačítka. První pro přechod do scény s tvorbou mříže, která bude popsána v následující kapitole. Druhé pro přechod do scény kde je možnost načítat data ze souboru. Třetí pro ukončení aplikace. Po stisku prvního tlačítka zde musí být zvolena zpětná vazba, obrázku s textem „Loading“, z důvodu delšího čekání na načtení a spuštění kamery. Na obrázku 3.1 jde vidět, jak bude vypadat výsledné menu. Obrázek na pozadí je obrázek přímo ze hry.

Obrázek 3.1 Ukázka jak vypadá menu

3.2 Scéna 2. Tvorba mříže

Hlavní scéna, kterou naše aplikace bude obsahovat. Bude zde funkčnost pro tvorbu mříže z jednotlivých čtverců (obrázek 3.2). Mříž bude hrací plocha pro hru. Lze si jí představit jako šachovnici, kde každé pole může být něčím obsazeno. Budeme se zde snažit, aby ovládání

v této části aplikace bylo co nejintuitivnější a pokud možno nejjednodušší i při zachováním velké funkčnosti. Ovládat se bude pomocí gest a tlačítek.

Obrázek 3.2 Ukázka jak vypadá tvorba hrací plochy (mříže)

Gesta pro tvorbu mříže budou posuny prstem po obrazovce zařízení ve směru, ve kterém budeme chtít zvětšit mříž. Aplikace bude začínat s jedním čtvercem uprostřed markeru (obrázek 3.3). Pokud použijeme gesto tak se vytvoří ve směru gesta jeden čtverec navíc. Na obrázku 3.4 můžeme vidět aplikované gesto směrem do pravé strany. Pokud bychom aplikovali další gesto směrem, např. dolů, vytvořili by se dva čtverce směrem dolů. Takto po řadách můžeme pokračovat, dokud nenarazíme na omezení aplikace z důvodů hardwarových prostředků. Pro mazání připravíme obdobný proces, kdy po zmáčknutí příslušného tlačítka se nám tatáž gesta změní v mazání v opačném směru. Toto je vše co uživatel potřebuje pro tvorbu libovolné mříže.

Obrázek 3.3 Ukázka tvorby mříže

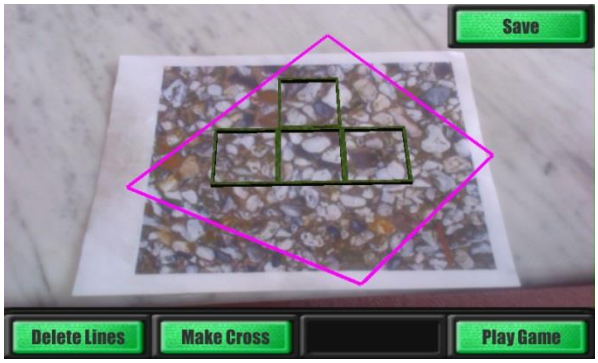
Obrázek 3.4 Ukázka přidání čtverce

Naše aplikace bude obsahovat i manipulaci s mříží pomocí gest. Pro rotaci celé mříže podle prvního prvku využijeme gesto pomocí dvou prstů, které po přiložení na monitor otáčíme po kružnici ve směru, ve kterém chceme otočit mříž. Dále zde bude možnost měnit měřítko mříže. Tedy měnit velikost každého čtverce. Díky těmto gestům nebude problém nastavit

libovolně velikou mříž jakkoli natočenou tam kde ji uživatel potřebuje. Na obrázku 3.5 vidíme, jak lze využít rotaci a zmenšení abychom umístili více čtverců do plochy.

Při používání se ale může stát, že budeme potřebovat více modifikací mříže a tím je ohraničení pomocí přímek. Toto je vhodné využit, např. když chceme mříž omezit jen na nějaký předmět jako stůl. Na toto zde bude také tlačítko, které vypne předešlé gesta a při stisknutí na monitor mimo GUI se budou postupně přidávat body mnohoúhelníku. Tento útvar může být uzavřen nebo otevřen a vždy se vztahuje na první prvek, který zde byl při spuštění aplikace. Musí zde být tlačítko, které se zobrazí, po přechodu do tvorby přímek, které dokáže odstranit přímku. Na obrázku 3.6 vidíme stisknuté tlačítko pro vytváření přímek. Dále je zde udělán čtverec jako ohraničení pro mříž.

Obrázek 3.5 Ukázka změny měřítka a rotace



Obrázek 3.6 Ukázka ohraničení pomocí přímek

Dále zde bude tlačítko pro uložení aplikace. Toto tlačítko po stisknutí zobrazí pole, do kterého lze zapsat jméno souboru. Toto jméno lze pomocí tlačítka odsouhlasit nebo zrušit pomocí tlačítka zpět. Po případném stisknutí tlačítka uložit se buďto soubor uloží nebo v případě, že již takový soubor existuje, zeptá se na jeho možné přepsání. Zde je zase možná volba souhlasu či nesouhlasu.

3.3 Scéna 3. Hra

Hra je dokonalá ukázka jak zobrazit využití mříže v aplikaci. Jak už bylo řečeno, budeme zde vyvíjet aplikaci podobnou hře minecraft. Tato hra již teď je legenda a dokonale se hodí na mříží stavěnou hru. Po přechodu z druhé nebo čtvrté scény se dostaneme na mříž, se kterou již není možné manipulovat. Je zde jen možnost přidávat prvky. Tato část musí co nejvíce vyhovovat uživateli, protože zde stráví nejvíce času. Nesmí zde být moc ovládacích prvků, které by znemožňovali pohled na hrací plochu. Je tedy nutné co nejvíce věci skrýt.

Ovládání pomocí doteku na čtverec hrací plochy způsobí, že na daném místě se zobrazí krychlový blok. Bloky je možné skládat na sebe nebo vedle sebe. Proto je důležité, aby zde byla možnost detekovat dotek na stěnu krychle a na tuto stěnu krychli připojit. Bloky je možné umisťovat jen na hrací plochu omezenou velikosti mříže. Nebude zde tedy možnost stavět neomezeně. Omezení je i na výšku z důvodu hardwarových nároku. Dále je nutné poznat, jestli na místě již nějaký blok existuje. Je nutné tedy zavést systém, kdy budeme uchovávat souřadnice v poli a na základě toho rozhodovat.

Možnost stavění různých typů bloků. Tato možnost je zde důležitá, aby hra byla různorodá a byli zde možnosti stavět z různých materiálů. Proto zde bude možnost gesta posunutí prstu po obrazovce směrem do levé nebo pravé strany. Podle směru se vysune/zasune boční panel, který bude obsahovat tlačítka. Tyto tlačítka budou pojmenována podle toho, o jaký typ bloku se jedná. Je zde výběr z několika bloků tento obsah se dá jednoduše doplnit podle libosti. Po výběru je možné panel zasunout zpět a pokračovat ve stavbě nových bloků.

Dále je zde možnost ukládání jako ve druhé scéně. Princip je stejný, jen se zde bude ukládat podstatně větší množství informací. Ale to bude vysvětleno v další kapitole.

3.4 Scéna 4. Načítání ze souboru

Tato scéna bude obsahovat výpis souborů, které půjde spustit. Naše aplikace bude potřebovat dva možné druhy uloženého souboru. Tedy i dva druhy výpisu. Toto bude docílené pomocí dvou tlačítek pro přepínání mezi jednotlivými druhy souborů. Pokud se jeden z nich stiskne, bude uprostřed vypsán jen ten druh souborů, který vyhovuje. Samozřejmostí zde bude také možnost pro posun v zobrazených souborech. Posun nahoru a dolů bude pomocí gesta, kdy prsten posunujeme po monitoru nahoru a dolů. Dále zde bude tlačítko pro smazání souborů, které smaže všechny soubory, co jsou vytvořeny naší aplikací. Při stisku nějakého tlačítka, které nese název souboru a je uprostřed se spustí příslušná scéna tedy buďto scéna pro vytvoření mříže nebo již samotná hra. Zpětná vazba by zde zase měla být v důsledku delšího načítání kamery a souboru.

4 Implementace

V této kapitole se podíváme na implementaci věcí, které byly zmíněny v předešlé kapitole. Budou zde řešeny i problémy, které nejsou na první pohled vidět, protože jsou na pozadí aplikace. Jsou to algoritmy, bez kterých by se program neobešel. Dále zde budou vždy uvedeny všechny objekty ve scéně spolu s jejich komponentami. Obsah kapitoly bude rozdělen na scény, což jsou jednotlivé obrazovky, které vidí uživatel.

4.1 Scéna 1. Menu

Menu je první obrazovka, kterou jsme vytvořili. Je načtena a spuštěna hned po úvodní obrazovce Unity. Její hlavní náplní bude směrovat chod aplikace do dalších scén. Také slouží pro vypnutí aplikace.

4.1.1 Objekty ve scéně

Tato scéna obsahuje jen GUI objekty. Pro vytvoření ovladatelného GUI v Unity je potřeba vytvořit Canvas. Canvas si můžeme představit jako plochu, která se vykresluje na monitor. Je zde nutné nastavit některé komponenty, aby se GUI vykreslovalo správně na co největším počtu zařízení. První komponentou je *Canvas*u kterého nastavíme, aby se vykreslovalo přes plochu snímanou kamerou (je zde možnost nastavit, že Canvas např. putuje spolu s kamerou, ale pro naše účely budeme překreslovat vše). Dále je zde *Canvas Scaler*, který se snaží usměrňovat velikost prvků podle potřeby a velikosti monitoru. Zde je nastaveno výchozí rozlišení 800px * 480px (rozlišení, které podporuje můj mobil). Pokud tato aplikace bude spouštěna na zařízeních s jiným rozlišením tak se bude počítat nová délka a šířka všech objektů v canvasu. Výpočet pomocí vzorce:

$$\frac{rozliseni\_obrazovky\_sirka}{vychozi\_rozliseni\_sirka} \times sirka\_objektu = nova\_sirka\_objektu$$

Pro výpočet délky je vzorec obdobný. Tímto získáme všechny objekty roztaženy na většinu monitorů. Dále je nutnost aby vše co se bude vykreslovat, bylo potomkem tohoto objektu. Zde vysvětlený postup bude aplikován na všechny další GUI elementy v aplikaci a nebude již znovu vysvětlován.

Canvas obsahuje 3 objekty. Tyto objekty jsou tlačítka, které budou řídit další činnost aplikace. Tlačítko je objekt, který je vykreslen jako čtverec s texturou. Jako potomek tohoto objektu je objekt se jménem Text, který složí pro vykreslení textu přes texturu tlačítka. Zde je možnost nastavit velikost textu, font, barvu atd. Největší předností tohoto objektu je komponenta, která sleduje stisk tlačítka. Je zde tedy možné nastavit, co se spustí za akci po stisknutí. U všech tlačítek posíláme zprávu objektu menu, který obsahuje komponentu, co zprávě rozumí.

Objekt Menu je objekt, který ve scéně nelze vidět ale je zde z důvodu, že obsahuje komponentu pro reakci na zprávy, kterou mu pošle tlačítko. Podle zprávy se rozhodne, jestli má spustit hru (první tlačítko), spustit obrazovku pro načítání dat ze souboru (druhé tlačítko) nebo ukončení aplikace (třetí tlačítko). Přechod do jiné scény tuto scénu zcela ukončí. Pokud bychom chtěli znovu spustit tuto scénu tak se musí znovu cela načíst.

4.2 Scéna 2. Tvorba mříže

Tato scéna je hlavní částí aplikace. Je nejsložitější a pokusíme se jí projít co možná nejpodrobněji. Scéna se spustí hned po stlačení tlačítka z předešlé scény. Je zde prodleva v načítání z důvodu spouštění kamery a načtení všech potřebných objektů ve scéně.

4.2.1 Rozšířená realita – Vuforia SDK

Pomocí Vuforia SDK skriptů a objektů je vytvořen základ pro rozšířenou realitu. Je zde objekt ARcamera, která vychází z normální kamery nutné pro každou scénu. Tato kamera je nastavená tak, aby snímala obraz zachycený defaultní kamerou zařízení. Pro mobilní zařízení to znamená kameru ze zadní strany telefonu pro např. notebook je to webkamera. Dále je zde nastavena kvalita snímání a rozpoznávání na položku default z důvodů že ani jedna z dalších položek nevyhovovala, protože byla buďto pomalá nebo nekvalitní. V neposlední řadě je zde nastavena komponenta *DataSetLoadBehaviour,* ve které je nutné nastavit, jaké markery budeme snímat. Tyto markery musí byt vygenerovány pomocí web aplikace na stránkách Vuforia. Tato aplikace vygeneruje dataShiet, který obsahuje markery pro snímání ve formátu, jaký potřebuje Vuforia SDK.

Dále objekt ImageTarget obsahuje komponentu pro nastavení příslušného dataShiet. Je zde také skript pro nastavení co se má stát pokud dojde k nalezení nebo ztrátu markeru. Tento skript je upraven tak aby některé objekty skrýval a některé ne pro potřeby aplikace. Pokud je

vše správně nastavené tak jakýkoli objekt jako potomek tohoto objektu se začne vykreslovat, pokud nalezneme marker.

4.2.2 Ovládání gest – TouchScript

TouchScript podle nastavení sleduje gesta. Hned na začátku je nutné mít ve scéně objekt TouchScript, ve kterém je nastavené jaký vstup je irelevantní. V tomto případě máme nastaven vstup na počítačovou myš a Vstup mobilního zařízení. Počítačová myš je tu z důvodu testování a programování aplikace na počítači, s výslednou aplikací nebude mít nic společného. Dále je zde *Touch Manager*, který ovládá vrstvy. V našem programu máme dvě vrstvy FullScreen (neboli celomonitorová) a dále ARcamera což je pohled kamery. První nám slouží pro zachytávání gest ovládání mříže. Druhá je pro ohraničení pomocí přímek.

Zachytávání gest funguje vždy na principu vytvoření listeneru (neboli posluchače) a handleru (neboli skriptu, který obsluhuje funkčnost gesta).

4.2.3 Vytváření mříže

Vytváření mříže se spouští pomocí gest, které jsou popsány v předešlé kapitole. Mříž si lze představit jako 2D pole čtverců. Toho je zde využito a pro ukládání a manipulaci je zde uděláno 2D pole. Je Omezeno na 100 x 100 protože neomezené pole by mohlo dělat problém s HW nároky mobilního zařízení. Pří načtení scény je do středu pole vložen první čtverec, jak bylo ukázáno v návrhu. Tento prvek je zobrazen již ve scéně a je to základní stavební prvek, na který se bude dále přidávat. Vytváření funguje na principu přidání hrany obdélníku. Nové části mříže jsou posunuty ve směru, kam chceme vytvořit hranu. Samotný posun funguje ale jen v případě, že mříž není nijak natočená. Pokud je zde natočení musíme posun centra nových částí počítat pomocí vzorce:

$$posun\_osa\_x = velikost\_části \times \cos(natoceni\_y) + souradnice\_predchozi\_casti$$
$$posun\_osa\_y = velikost\_části \times \sin(natoceni\_y) + souradnice\_predchozi\_casti$$

Pro různé strany je potřeba změna znamének, ale princip zůstává stejný. Pří odebírání stran pracujeme na principu, kdy odebereme všechny indexy pole, které nevyhovují.

4.2.4 Vytváření ohraničení

Tvorba ohraničení pomocí přímek funguje tak, že pomocí LineRendereru a stisku na monitor vytváříme body přímky. Při prvním stisknutí se vytvoří první startovní bod přímky. Při

dalším stisknutí se již vytváří body dva koncový a začátek druhé přímky. Po každé úspěšně udělané přímce (druhé a potom každé kliknutí na monitor) se na přímce vytvoří boxCollider. BoxCollider je prvek v Unity, kterým je možno detekovat, jestli se ho nedotýká jiný boxCollider. Díky tomuto prvku pak části mříže, které narazí do boxCollideru dočasně mizí. Tvorba tohoto boxCollideru je složitá protože musíme vytvořit takový kvádr, který přesně kopíruje přímku. Na obrázku 4.1 lze vidět jak je mříž 3x3 omezená růžovou přímkou. Také zde lze vidět zelená přímka, které ukazuje část kvádru boxCollideru. Z důvodů že kvádr je obrovský, nelze ho ukázat celý. Princip tvorby boxCollideru:

1. Výpočet středu přímky vzorec 4.1.

$$\overrightarrow{střed} = [\frac{(začátek_x + konec_x)}{2} , \frac{(začátek_y + konec_y)}{2}]$$

(vzorec 4.1)

2. Výpočet vektoru přímky vzorec 4.2

$$\overrightarrow{vektorPřímky} = \overrightarrow{konec} - \overrightarrow{začátek}$$

(vzorec 4.2)

3. Výpočet náklonu přímky vzorec 4.3

$$náklon = arctg2(vektorPřímky_x , vektorPřímky_y) \cdot 180 \cdot \pi$$

(vzorec 4.3)

4. Výpočet středu kvádru boxCollideru vzorec 4.4 a vzorec 4.5

$$\vec{v} = \overrightarrow{konec} - \overrightarrow{střed}$$

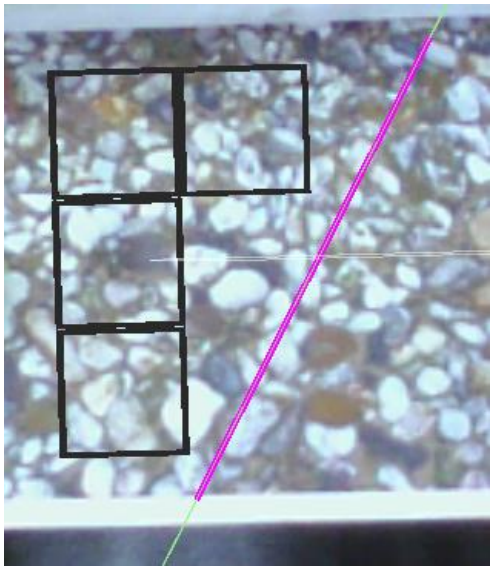
(vzorec 4.4)

$$\overrightarrow{vektorKvádru} = [\frac{-v_y}{\sqrt{{v_x}^2 + {v_y}^2}} \cdot velikost , \quad \frac{v_x}{\sqrt{{v_x}^2 + {v_y}^2}} \cdot velikost]$$

(vzorec 4.5)

5. Vektor_kvádu posuneme tak aby vycházel ze středu přímky

Podle náklonu zjistíme, jakým směrem ukazuje a na jaké straně přímky máme vytvořit boxCollider. Tímto jsme vytvořili funkční ohraničení přímkou.



Obrázek 4.1Ukázka BoxCollideru

4.2.5 Ukládání mříže

Ukládání funguje na principu serializace. Tedy data, které potřebujeme uchovat, jsou přepsány do tvaru, kdy mohou být uloženy do souboru. Unity je schopno převést některé objekty do struktury YAML. Typ objektů je velmi omezen, a proto není možné serializovat např. GameObject nebo Vector3 (toto jsou velice používané objekty v programování v Unity). Je zde tedy nutnost vytvořit si vlastní objekty, anebo vytáhnout data z Objektů a obejít tak ukládání celku. V našem projektu je nutné ukládat: polohu všech čtverců mříže, rotaci, měřítko, a další méně podstatné věci. Pro ukládání pozic čtverců je nutné vytvořit strukturu, která z objektu Vector3 (Unity objekt, který představuje 3-rozměrný vektor v prostoru) udělá tři čísla. Konvertovat je možné i zpět do Vector3. Tímto je možné ukládat všechny potřebné pozice všech objektů, jejich velikost a rotaci (vše toto je uloženo nativně ve Vector3). Data si tedy uložíme do struktury, kde máme obsaženy všechny věci, které chceme ukládat. Tuto strukturu serializujeme. Vytvoříme soubor a do něj všechny data uložíme.

4.3 Scéna 3. Hra

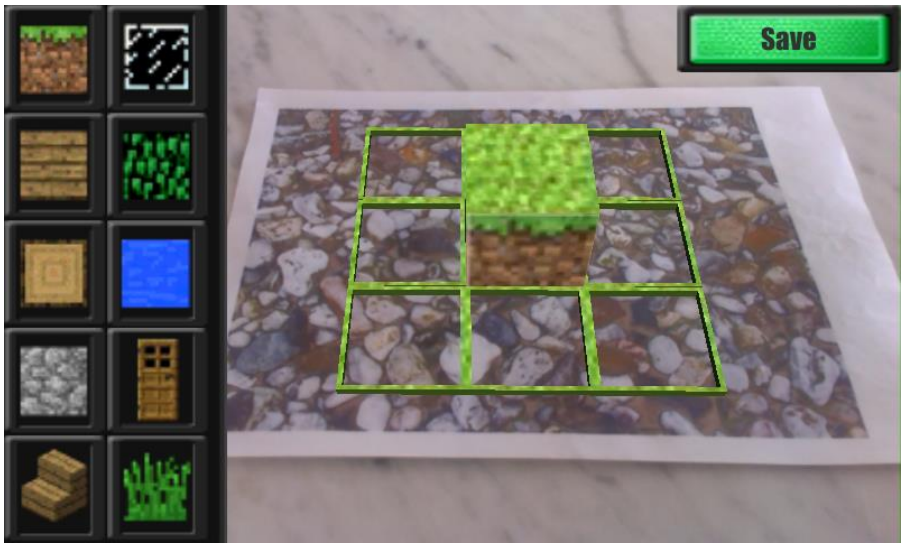
Tato scéna obsahuje ukázku, jak je možné využít naimplementovanou mříž. Jak již bylo zmíněno, vytvoříme hru podobnou hře Minecraft. Ukážeme si jak dosáhnout orientaci v prostoru pomocí pole. Jak získávat pozici stisku a různé jiné postupy, jak docílit výsledku.

4.3.1 Objekty ve scéně

Musí zde být prvky pro rozšířenou realitu jako v předchozí scéně. Scéna obsahuje stejné tlačítko ukládání se stejnou funkčností, jen zde ukládá jiné věci. Dále je zde nutné mít prvky pro vytváření bloků. Na všechny bloky ve scéně máme připravený tzv. Prefab. Prefab znamená prototyp objektu, který má uložené vlastnosti. Je vygenerován pomocí 3D grafického programu, ve kterém použijeme export do tvaru .fbx. Je zde nutnost mít objekt, který bude hlídat ovládání pomocí stisknutí. Byl vytvořen skript, který bude naslouchat tomuto stisknutí, a vložíme jej do scény.

4.3.2 Ovládání pomocí doteku na obrazovku

Scéna se bude ovládat výhradně stisknutím na obrazovku. Každé stisknutí na mříž vyvolá skript pro vytvoření bloku. Tvorba bloku funguje na principu indexu v poli. Je tedy nutné si vytvořit 3D pole a v něm ukládat jestli je toto místo obsazené nebo nikoli. Díky tomuto principu je jednoduché implementovat další rozšíření jako optimalizaci nebo fyziku těles. Protože přesně víme, jestli jsou vedle sebe nějaké tělesa nebo ne. Dále každý blok si musí uchovávat své souřadnice v tomto poli. Každému bloku tedy přidáme skript, který bude obsahovat 3 čísla neboli souřadnice. Díky tomuto principu jsme schopní při mazání bloku zjistit, jaké místo máme uvolnit. Mazání bloku provedeme přidržením na jednom bloku. Pokud se prst při zvednutí posune na jiný blok, nedojde ke smazání. Tímto jsem chtěl zaručit, že nechtěně nevymažeme objekt, který nechceme. Na obrázku 4.2 vidíme, jak je na mříž aplikován jeden blok po stisknutí na čtverec hrací plochy.

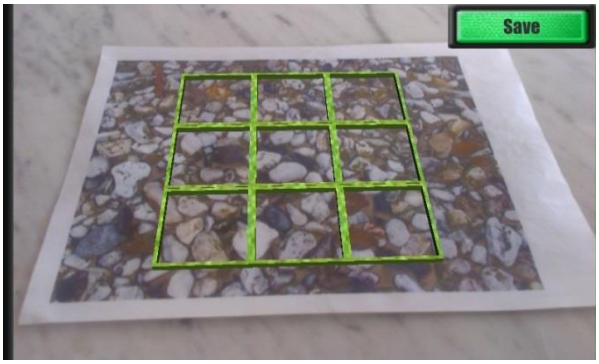


Obrázek 4.2 Ukázka hry – přidání prvního bloku

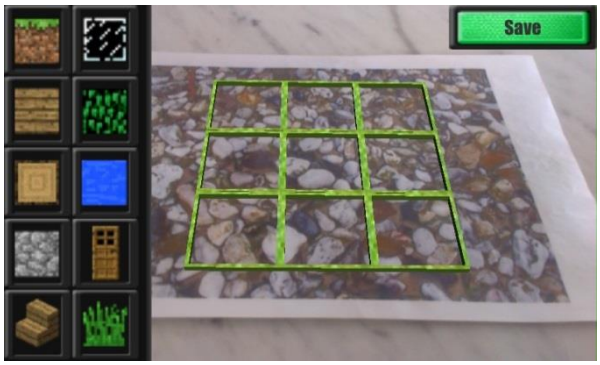
V pravé části monitoru máme možnost vysunout panel. Vysunutý panel obsahuje tlačítka pro změnu bloku. Pokud se tlačítko stiskne tak dojde k nastavení proměnné ve skriptu

pro tvorbu bloků. Je tedy nutné si tento údaj pamatovat stejně jako souřadnice bloku. Vysouvání bloku panelu funguje tak, že sledujeme bod stisknutí, pokud bod zvednutí prstu je vzdálený dostatečně daleko tak zapneme simulaci pohybu. Tento pohyb vysune panel. Na obrázku 4.3 vidíme základní obrazovku bez vysunutého panelu. Na obrázku 4.4 vidíme vysunutý panel.

Unity např. pro Android nativně neumí filtrovat stisknutí přes GUI. Je důležité vytvořit skript tak aby si s tímto poradil. Na tento princip lze aplikovat hodně postupů, jen spousta jich také nejde na systém Android. Proto zde zvolíme algoritmus sledování souřadnic tlačítek a vysouvacího panelu a podle doteku se budeme řídit, jestli jsme v panelu nebo venku.



Obrázek 4.3 Ukázka zasunutého panelu



Obrázek 4.4Ukázka vysunutého panelu

4.3.3 Tvorba a typ bloků

Tvorba bloku funguje na principu kopírování Prefabu do scény. Je zde nutnost nastavit objektu velikost, jméno a pozici v poli. Pozice se určuje podle bodu stisknutí. Každý blok má na sobě BoxCollider, který snímá bod doteku. Pokud se dotkneme na straně bloku (je zde tedy 6 variant jak je možné blok vytvořit) je zavolána příslušná funkce, která tímto směrem vytvoří. Kontrola hrany, které se dotkneme, je řešena pomocí trigonometrie, a na základě úhlu určíme stěnu. Pří stisknutí první zjistíme hranu a tedy i směr, kterým chceme vytvořit blok. Musíme tedy v tomto směru otestovat, jestli zde už není blok. Pokud není, můžeme zde blok vytvořit a zapsat mu příslušné hodnoty.

Blok musí být vždy krychle o hraně velké, jako je čtverec mříže. Je to z důvodu aby se dalo vše navazovat na sebe. Naše hra obsahuje jednoduché bloky typu: hlína, kámen, dřevo, desky. Tyto bloky jsou jen přesný kvádr s texturou lze je dávat kdekoli, kde je místo. Dále tu jsou bloky typu: sklo a listy. Tyto bloky jsou zvláštní tím, že jsou částečně průhledné, aby hra dostala více možností. Dále je možnost usadit schody a dveře ty jsou zvláštní tím, že je lze pokládat natočené směrem od/ke kameře. Je to z důvodu, že tyto věci nejsou pravidelné a uživatel musí mít možnost je otočit. Dveře jsou speciální ještě tím, že jsou složeny ze dvou bloků. Toto je

nutné ošetřit a přidávání a mazaní je tedy složité. A je např. nutné, aby si jednotlivé bloky dveří pamatovali své ostatní části. Jako poslední je zde voda. Voda je modrý blok, který je o něco menší než ostatní bloky. Toto je velký problém protože BoxCollider vytvořený na tomto objektu je menší, tedy zachytávání dotyku je špatné. Z tohoto důvodu je nutné aby boxCollider byl zvětšen. Další, co zde kontrolujeme je to, že nad vodou není možnost stavět blok. Je to z důvodu větší logiky hry.

Na obrázku 4.5 lze vidět, jak je možné na sebe vrstvit jednotlivé bloky a co všechno lze vytvořit.

Obrázek 4.5 Ukázka čeho je aplikace schopná

4.3.4 Ukládání hry

Ukládání funguje na stejném principu jako ukládání mříže. Jen je zde rozdíl, že dat je více. Musíme si pamatovat stejně jako v předešlém ukládání pozici všech čtverců mříže, jejich natočení a velikost. Pro bloky si musíme ukládat pozici bloků, index bloku v 3D poli, natočení jednotlivých bloků, protože jsou bloky, které nesdílejí stejné natočení jako mříž a typ bloku. Všechny tyto informace jsou důležité pro správné vykreslení scény po načtení ze souboru.

4.4 Scéna 4. Načítání ze souboru

Scéna, bez které by ukládání souboru bylo zbytečné. Tato scéna nám umožní vybrat si z uložených souborů a načte ho. Jedná se převážně o GUI objekty výhradně tedy tlačítka. Dále si zde řekneme o tom, jak se data načítají do každé scény, i když to není přímo úkol této scény.

4.4.1 Objekty ve scéně

GUI vypadá jednoduše, máme zde Canvas, ve kterém jsou tři butony hned při načtení scény. Pokud stiskneme levé nebo pravé tlačítko, tak se nám spustí skript. Ten vygeneruje další tlačítka. Těmto tlačítkům přidá akci, které se vykoná po jeho kliknutí a nastaví jim jméno na jméno souboru, který reprezentují. Tímto docílíme výpis všech souborů jako list tlačítek. Mezi listy je možno libovolně přepínat a vždy jsou zobrazeny jen soubory pro tvorbu mříže nebo pro samotnou hru. Po kliknutí se hned spustí příslušná scéna. Do scény, která se spustí, je vložen objekt, který reprezentuje, že scéna je načítána ze souboru a je potřeba změnit její chování tak aby se načetli data ze souboru. Jak scéna vypadá, se můžeme podívat na obrázku 4.6.

Obrázek 4.6 Ukázka jak vypadá scéna načítání ze souboru

Možnost posouvání při velkém počtu uložených souborů. Může se stát, že počet souborů je větší než můžeme pod sebe na jedné obrazovce ukázat. Z tohoto důvodu jsem vytvořil skript, kterým je možné pohybovat tlačítky nahoru nebo dolů pomocí táhnutím prstu po obrazovce v daném směru. Unity nativně nemá žádný UI prvek pro takový druh pohybu, proto bylo nutné vymyslet a implementovat skript. Skript využívá polohu tlačítek pro posuzování, jestli jich je moc. Pokud jich je moc tak je možnost posunout dokud poslední tlačítko není kousek nad spodem obrazovky. To samé platí o posunu nahoru.

4.4.2 Načítání dat ze souboru

V aplikaci máme, dva typy souborů s uložením. Jeden pro tvorbu mříže a druhý pro hru. Proto zde máme i dva možné způsoby práce s daty. Toto načítání není přímo v této scéně, je obsažené ve všech scénách, ale pro lepší přehled to rozebereme zde. Načítání začíná v této scéně, kdy dochází k přechodu do jiné scény s objektem, který nese informaci o tom, že se

budou načítat data. Scéna spustí skript, který deserializuje data ze souboru, který jsme uložili. Deserializací se nám zaplní příslušné hodnoty proměnných, ke kterým budeme přistupovat.

Pro druhou scénu je nutné vytvořit mříž, kterou jsme si uložili. Toho docílíme tak, že vytváříme znova čtverce mříže a dáváme jim pozici a velikost jakou máme uloženou. Protože první vytváříme objekty, již pozicované, je nutné jim dávat rotaci a měřítko zvlášť. Je to opačný postup, než když vytváříme mříž od začátku, kde se rotace aplikuje na celek a ne na jednotlivé prvky. Dále je zde nutné pamatovat si přímky, které ohraničují prostor. U těchto si pamatujeme vždy dva body tedy začátek a konec. Postupně tedy tyto body aplikujeme a přímky se vytvoří.

Pro třetí scénu tedy hru potřebujeme více věcí. Je zde nutné načíst i bloky, které v předchozí scéně nejsou. Tedy načítání mříže bude stejné jako v předchozí scéně. Bloky vytváříme na základě uložené pozice a uložené informaci o typu bloku. Dále je nutné používat podružné informace jako je rotace nebo měřítko. V naší aplikaci jsme si zavedli princip orientace v prostoru pomocí indexů v poli. Tyto hodnoty je nutné aplikovat na objekty, aby další přidané objekty mohli spolupracovat.

5 Testování

V této kapitole se podíváme na výsledky testů, které by měli říct, jestli jsem aplikaci vytvořil dobře a splňuje předpokládané prvky jako je intuitivnost prostředí, jednoduchost, pochopitelnost atd. Dále aplikace byla testována v průběhu programování, abych mohl jednotlivé chyby odhalit a řešit dříve.

5.1 První testování návrhu aplikace

První testování probíhalo v duchu návrhovém. Snažil jsem se na základě prvních střípku aplikace zjistit, co by vyhovovalo uživatelům takovéto aplikace. Zaměřil jsem se na mladší věkovou skupinu dětí ve věku 10-15 let, protože aplikace by měla být vyvíjena pro tuto věkovou kategorii. Samotné testování tedy probíhalo tak, že jsem vybraným dětem vysvětlil, jak by aplikace měla vypadat, co by měla obnášet a ukázal jsem jim prototyp aplikace. Aplikace byla ve stádiu, kdy uměla jen ukázat pár vybraných objektů jako rozšířenou realitu. Děti rády spolupracovaly a dávali zpětnou vazbu mnohdy lepší než dospělí. Rozdal jsem jim dotazník s otázkami a na základě odpovědí jsem vytvořil souhrn věcí, které by byli dobré pro mou aplikaci. Výsledkem testování bylo, že aplikace musí být napsána česky (92% dotázaných). Musí obsahovat grafiku plnou barev (83% dotázaných). Ovládání musí byt pomocí jednoduchých gest (78% dotázaných). Na základě těchto výsledků jsem se zaměřil na tyto aspekty a využil jsem hlavně ovládání pomocí gest. Aplikaci jsem ale naprogramoval v jazyku anglickém, protože test byl dělán s dětmi, které jiný jazyk pořádně neovládají, takže tato část testu je irelevantní.

5.2 Testování intuitivnosti uživatelského rozhraní

Druhé testování bylo již na hotové aplikaci, kde bylo nutné zjistit, jak moc se mi povedlo splnit cíle dané v počátku této práce. Testovací skupina zde byla částečně stejná jako minule. Děti ve věku 10-15 let a dále jsem dal otestovat aplikaci i ostatním věkovým skupinám, které ale byli v menším počtu. Vytvořil jsem dotazník, který vyplňovali v průběhu testování aplikace. Byli seznámeni s použitím aplikace před testováním. Dotazník měl odhalit chyby, které vznikly v průběhu programování aplikace. Výsledky testování byli překvapivé a na aplikaci byli nutné změny. Po velice špatné odezvě na tvorbu přímek, kdy 73% dotázaných uvedlo, že pokládání přímek není přesné a často dochází ke tvorbě přímky, i když to uživatel nevyžaduje tedy překlepem. Bylo tedy nutné vymyslet jiný způsob tvorby přímek a to tak, že je vytvořeno tlačítko, které umožnuje tvorbu přímek. Dále testování odhalilo problém, klikání skrz grafické rozhraní, což bylo nežádoucí a bylo nutné tento problém řešit. V neposlední řadě ve hře

uživatelé uvedli, že vysouvací panel s bloky, který v té době byl vysouván pomocí tlačítka, je špatně přístupný. Proto jsem zavedl gesto pro vysunování panelu. Jako poslední zde uvedu, že většina (94% dotázaných) poukazovala na to, že aplikace má na mnoha místech špatnou zpětnou vazbu. Z tohoto důvodu jsem vytvořil zpětnou vazbu např. po spuštění hry, kdy docházelo k čekání na načtení aplikace, každé tlačítko dostalo grafiku tak aby šlo dobře poznat, že bylo tlačítko stlačeno.

6 Porovnání s aplikacemi podporující rozšířenou realitu

Tato kapitola bude pojednávat o aplikacích, které se zabývají buďto rozšířenou realitou nebo tvorbou map. Uvedeme si zde rozdíl a pokusíme se zde popsat výhody této aplikace oproti jiným. Dále se podíváme na rozdíly mezi touto aplikací a hrou Minecraft.

6.1 Minecraft

Minecraft je již legendární sandboxová, RPG hra. Její hlavní přínos je v jednoduchém ovládání a možnostem v otevřeném náhodně generovaném světě. Ovládání musí být jednoduché, protože se aplikace zaměřuje na mladší populaci. Ovládá se výhradně jen pomocí kláves W, S, A, D a myši. Dále tu jsou klávesové zkratky, které jsou intuitivní, a není nutnost je používat. Ovládání navíc velice zjednodušuje velká podpora z komunity hráčů, kteří píšou návody a natáčejí videa z prostředí hry. Dále hra obsahuje obrovskou hratelnost a možnosti. Možnosti jsou jak ve stavění z bloků tak z vytváření věcí ve hře. Kombinování různých materiálů, které následně lze použít dále. Co se týká stavění hra je skoro neomezená a na internetu lze najít nepřeberné množství staveb. Další obrovské plus této aplikace je grafika. Je pixelová a to, že její textury se snaží být co nejvíce jednoduché. Aplikace tedy podle mě splňuje vše, co si mladší hráč může přát.

Naše aplikace se snaží napodobit některé aspekty této hry. Máme podobnou jednoduchou a líbivou grafiku, jednoduché a intuitivní ovládání. Aplikace minecraft ale může být omezující v pohybu. V normálním módu je možnost pohybovat se jen po zemi a surovin není neomezeně. V módu creative je možnost létání a všech surovin máte neomezeně a můžete použít cokoli. Létání je ovšem omezující na ovládání a není zcela intuitivní. Má aplikace umožňuje dívat se na hru z různých úhlů jednoduše a rychle. V tomto je hlavní výhoda naší aplikace. Člověk se díky virtuální realitě připadá jako součást světa, který si postupně vytváří. Toto byl hlavní účel této aplikace přiblížit hru uživateli co možná nejvíce.

6.2 Typy aplikaci využívající rozšířenou realitu

Aplikací využívající rozšířenou realitu je v dnešní době spousta. Většina těchto aplikaci plní reklamní, poznávací nebo naučný účel. Většina aplikaci, které jsou postavené na GPS technologii, jsou reklamní nebo se snaží uživateli pomoc ve vyhledávání. Např. aplikace a SDK Wikitude. Je to velice známá aplikace i využívané SDK. Obsahuje jak GPS pozicování, tak rozšířenou realitu pomoci markeru. Využívaná je především pro reklamní účely, ať je to vyhledávání nejbližších restaurací nebo ukázka předmětu z časopisu na vašem mobilu ve 3D. Další velký zástupce této kategorie je Junaio od dříve uvedené společnosti Metaio. Zase většinové využití aplikace je orientace v prostoru. Nebo reklamní účely, které je možné vidět např. na některých zastávkách v Brně. Jedním z vůbec nejslavnější aplikace využívající rozšířenou realitu je Star Chart, která při pohled na hvězdy na nebi dokáže vyhledat a určit o jaké souhvězdí jde a přímo vám to ukáže na monitoru. V počtu stažení dominuje takovýmto aplikacím. Takovýmto aplikacím ale podle mě chybí nějaká možnost interakce. Možnosti těchto velikánů jsou obrovské, ale jsou využity jen na komerční účely. Zde si myslím, že má aplikace má opravdu co nabídnout. Oproti těmto aplikacím v mé aplikaci opravdu můžete něco dělat. Můžete si hrát, můžete se předvádět ostatním.

Dále tu jsou aplikace, které již dokáží zaujmout a jsou interaktivní. Aplikace typu X-Rift. Augmented Reality Game, ve které máte kolem sebe pomocí GPS zobrazeny nepřátele, můžete na zem pokládat různé předměty. Je to tedy interaktivní hra, která neplní reklamní účel. Je dokonce online a můžete si ji zahrát s přáteli. Velká nevýhoda je čekání na možnost připojit se do hry. Musíte se poslat žádost o přijetí, která v mém případě je stále nevyhověna (po půl roce). Takže pokud chcete zkusit takovouto aplikaci, buďto musíte čekat velice dlouho nebo najít někoho kdo už má do hry přístup. Na rozdíl od naší aplikace nabízí online propojení s více hráči, je to akční hra, ve které jde o to být nejlepší. Je nutnost mít internetové připojení a také mít přistup a čas chodit ven. Naší aplikaci si můžete zahrát z pohodlí domova, není zde potřeba žádného připojení k internetu.

7 Závěr

Výsledkem této práce je mobilní aplikace, která využívá technologie rozšířené reality pro vykreslení hrací plochy do obrazu snímaného kamerou. Ovládání aplikace je jednoduché a intuitivní jak bylo plánováno. Grafické prostředí je barevné tak aby se líbilo dětem, což je hlavní skupina, pro kterou je aplikace vyvíjena. Samotná hra, kladení různých typů bloků na hrací plochu, je uzpůsobena tak aby šlo vše ovládat co nejjednodušeji a přesně. Dále je program naprogramován tak aby vygenerovaná hrací plocha šla využít i jinak. Je zde tedy neomezené množství možností jak tuto čtvercovou síť využit.

Přínosem pro mě bylo seznámení se s technologii rozšířené reality, která má velkou budoucnost a v následujících letech si myslím, že dojde k velkému pokroku. Seznámeni s enginem Unity 3D, což je skvělý nástroj pro tvorbu her a aplikací. Jsem velice rád, že jsem se s ním naučil pracovat. V Unity 3D jsem se naučil využívat knihovny/assety jako je Vuforia SDK a TouchScript. Vuforia SDK je podle mě skvělý nástroj a dobře se sním pracovalo. TouchScript byl na druhou stranu pro mě špatný nástroj a v další prací bych jistě zvolil jiný. V neposlední řadě pochopení různých principů jako jsou: programování pod systém android, programování v jazyce C# a pochopeni principu rozšířené reality.

Možnosti rozšíření by mohli být v principu tvorby map. Toto rozšíření by fungovalo ve vytvoření mříže a generování mapy, neboli hrací plochy z bloků, náhodně. Hra by potom více připomínala hru Minecraft. Další možnost rozšíření je změna hry. Jak jsem již říkal, program byl vytvořen tak, že hra jen používá mříž. Je zde tedy možnost vytvořit jakoukoli hru, která má možnost využít síť ze čtverců. V neposlední řadě je zde samozřejmě možnost vytvořit více bloků ve hře. Lze naprogramovat jednoduchou fyziku těles, které by např. padaly, pokud by pod nimi nebyl blok. Nebo doplnit umělou inteligenci a spolu s tím i zvířata nebo jiné živočichy.

8 Literatura

[1] Sanni Siltanen. *Theory and applicationsof marker-basedaugmented reality.* Espoo: VTT PL 1000, 02044 VTT, Puh. 020 722 111 , 2012. ISBN 978-951-38-7449-0.

[2]Gartner.. *Gartner, Inc. and/oritsAffiliates..* [online]. 10.5.2015 [cit. 2015-05-10]. Dostupné z: *http://www.gartner.com/newsroom/id/2954317*

[3]metaio Developer Portal. *metaioGmbH.* [online]. 10.5.2015 [cit. 2015-05-10]. Dostupné z: *http://dev.metaio.com/junaio/documentation/channels/location-based-channels/*

[4]Qualcom Vuforia Developer Library. *Developer Vuforia.* [online]. 10.5.2015 [cit. 2015-05-10]. Dostupné z:*https://developer.vuforia.com/library/articles/Training/Extended-Tracking*

[5] CARMIGNIANI, Julie, Borko FURHT, Marco ANISETTI, Paolo CERAVOLO, Ernesto DAMIANI a Misa IVKOVIC. Augmented reality technologies, systems and applications. *MultimediaTools and Applications* [online]. 2010, vol. 51, issue 1, s. 341-377 [cit. 2015-05-11]. DOI: 10.1007/s11042-010-0660-6.

[6]TouchScript. *TouchScript, Valentin Simonov, InteractiveLab..* [online]. 13.5.2015 [cit. 2015-05-13]. Dostupné z: *http://touchscript.github.io/*

[7] dcryengine. *cryengine – crytek.* [online]. 10.5.2015 [cit. 2015-05-13]. Dostupné z: *http://cryengine.com/support*

[8] Unreal Engine Community Wiki. Unreal Engine – EpicGames. [online]. 13.5.2015 [cit. 2015-05-13]. Dostupné z: *https://wiki.unrealengine.com/Main_Page*

[9] Unity. *Unity Technologies.* [online]. 10.5.2015 [cit. 2015-05-13]. Dostupné z: *http://unity3d.com/learn*

[10] *International journal on computational science: COMPARATIVE STUDY OF AUGMENTED REALITY SDK'S* [online]. 2015. India [cit. 2015-05-13]. ISBN 2200-0011. ISSN 2200 - 0011. Dostupné z: http://airccse.org/journal/ijcsa/papers/5115ijcsa02.pdf

Seznam příloh:

A Obrázek markeru

B Ovládání aplikace

B. 1 Menu

Menu obsahuje tři tlačítka. 1. Make Cross pro spuštění části s vytvářením mříže. 2. Load, kterým se dostaneme na seznam uložených souborů. 3. Exit, kterým aplikaci vypneme.

B. 2 Tvorba mříže

Zde po načtení a zachycení markeru máme možnost využit gesta pro zvětšení mříže.

1. Posun vpravo – pro vytvoření řady čtverců na pravou stranu
2. Posun vlevo – pro vytvoření řady čtverců na levou stranu
3. Posun nahoru – pro vytvoření řady čtverců na horní stranu
4. Posun dolů – pro vytvoření řady čtverců na spodní stranu

Takto vytvoříme mříž velikosti, jakou potřebujeme. Pro odebírání řad je zde možnost stisknout tlačítko DeleteCross, které změní tyto 4 gesta na opačný směr, ve kterém maže jednotlivé řady.

Pro manipulaci s celou mříží máme možnost použít dvě gesta. Rotace pomocí dvou prstů přiložených na monitor a pohyb po kruhu ve směru, kterým chceme otáčet. Nebo změnu měřítka mříže pomocí dvou prstů přiložených na monitor a posun prstů k sobě nebo od sebe pro zmenšení nebo zvětšení.

Pro tvorbu ohraničení pomocí přímek stiskneme tlačítko Make Lines, které nám vypne možnost přidávat a manipulovat s mříží ale zapne nám možnost tvorby přímek. Po stisku na obrazovku se nám vytvoří bod. Po dalším stisknutí se vytvoří další bod a spojí se s tím předchozím. Takto vytvoříme jakýkoli úhelník. Dále je zde tlačítko pro smazání přímek Delete Lines.

Pro uložení je zde tlačítko Save, kdy se zobrazí pole, do kterého zapíšete název a dáte uložit. Pokud již takový název je, budete dotázáni, zda chcete soubor přepsat nebo ne.

B. 3 Hra

Po kliknutí na čtverec ve hře se na poli objeví blok. Stisknutím na jakoukoli hranu bloku se vytvoří další blok, který se přilepí na tu hranu, na kterou jste stiskli. Pro změnu bloku, který pokládáte, stačí použít gesto táhnutí prstem z levé strany monitoru na druhý. Vysune se z pravé strany panel, kde jsou tlačítka s obrázkem bloků. Po stisknutí můžete klást bloky, které jste si vybrali. Pro zasunutí panelu gesto táhnutí prstem z pravé strany monitoru na levý.

Ukládání stejně jako ve tvorbě mříže.

B. 4 načítání ze souboru

Pro načtení dat ze souboru je nutné první vybrat, jaký soubor chcete. Jestli se jedná o tvorbu mříže tak stisknete tlačítko CrossSave, kdy se vám uprostřed zobrazí všechny soubory, které jste si uložil. To samé platí pro Game Save, kdy se zobrazí soubory se hrou. Po stisknutí některého z tlačítek se jménem souboru se hra spustí.

C Obsah CD

- Složka BakalarskaPrace – obsahuje Unity projekt
    - Složka Assets
        - Složka MyScripts – skriptky, které jsem naprogramoval
        - Složka MyFonts – fonty, které jsem použil
        - Složka MyTextures – textury, které jsem vytvořil
        - Složka MyModels – modely, které jsem vytvořil
        - Složka MyScenes – Unity scény, které jsem vytvořil
        - Složka Qualcomm Augmented Reality – obsahuje Vuforia SDK
        - Složka TouchScript – obsahuje TouchScript
- Složka Application – obsahuje již zkompilovanou aplikaci pro android
- Složka Documentation – obsahuje tuto dokumentaci ve formatu PDF
- Složka Demonstration – obsahuje plakát a video prezentující aplikaci