VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

INTERAKTIVNÍ VIZUALIZACE POVĚTRNOSTNÍCH VLIVŮ POMOCÍ ZAŘÍZENÍ SANDBOX

INTERACTIVE WEATHER VISUALISATION WITH SANDBOX DEVICE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **MICHAL ONDREJKA**
AUTHOR

VEDOUCÍ PRÁCE **doc. Ing. VÍTĚZSLAV BERAN, Ph.D.**
SUPERVISOR

BRNO 2024

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Zadání bakalářské práce

150230

Ústav: Ústav počítačové grafiky a multimédií (UPGM)
Student: **Ondrejka Michal**
Program: Informační technologie
Název: **Interaktivní vizualizace povětrnostních vlivů pomocí zařízení SandBox**
Kategorie: Uživatelská rozhraní
Akademický rok: 2023/24

Zadání:

1. Seznamte se se zařízením Interactive-SandBox, metodami pro vizualizaci povětrnostních vlivů v terénu a návrhem interaktivních aplikací.
2. Navrhněte aplikaci pro zařízení Interactive-SandBox, kdy bude uživatel pomocí gest ovlivňovat srážky nad 3D terénem, který v písku předem vytvořil. Povětrnostní vlivy v terénu budou v reálném čase vizualizovány.
3. Navrženou aplikaci realizujte s využitím existujícího zařízení a vhodných knihoven.
4. Řešení testujte na uživatelích a vyhodnoťte.
5. Prezentujte klíčové vlastnosti řešení formou plakátu a krátkého videa.

Literatura:

- Dieter Schmalstieg, Tobias Hollerer. *Augmented Reality: Principles and Practice.* Addison-Wesley, 2016. ISBN: 978-0321883575.
- Gary R. Bradski, Adrian Kaehler. *Learning OpenCV: Computer Vision with the OpenCV Library*, ISBN 10: 0-596-51613-4, September 2008.
- Dále dle pokynu vedoucího.

Při obhajobě semestrální části projektu je požadováno:
Body 1., 2. a částečně bod 3.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Beran Vítězslav, doc. Ing., Ph.D.**
Vedoucí ústavu: Černocký Jan, prof. Dr. Ing.
Datum zadání: 1.11.2023
Termín pro odevzdání: 9.5.2024
Datum schválení: 9.11.2023

Abstrakt

Cieľom tejto práce je návrh a implementácia interaktívneho výukového systému, ktorý dovolí užívateľovi preskúmať a pochopiť poveternostné podmienky v reálnom čase. Pri realizácii sa vychádzalo z existujúceho riešenia AR SanBox. Za účelom simulácie toku tekutín sú v tejto práci preberané základné algoritmy pre simuláciu vody. Ďalej sú popísané spôsoby interakcie za pomoci rozpoznávania gest ruky. Okrem iného aplikácia umožňuje interaktívne modelovanie terénu a vizualizáciu nadmorských výšok. Toto riešenie umožňuje užívateľovi prívetivo získavať poznatky z oblasti topografie a hydrológie. Teoretická časť bakalárskej práce uvádza prehľad odbornej literatúry potrebnej na správny návrh riešenia. Praktická časť práce je zameraná na realizáciu návrhu.

Abstract

The goal of this work is the design and implementation of an interactive learning system that allows the user to explore and understand weather conditions in real time. The implementation is based on the existing AR SanBox solution. In order to simulate the flow of liquids, basic algorithms for water simulation are discussed in this work. Next, methods of interaction using hand gesture recognition are described. Among other things, the application enables interactive terrain modeling and elevation visualization. This solution enables the user to gain knowledge in the field of topography and hydrology in a friendly way. The theoretical part of the bachelor's thesis presents an overview of the professional literature necessary for the correct design of the solution. The practical part of the work is focused on the implementation of the proposal.

Klíčová slova

hĺbkový senzor, interaktívne metódy, simulácia tekutín, rozpoznávanie gest ruky

Keywords

depth sensor, interactive methods, fluid simulation, hand gestures recognition

Citace

ONDREJKA, Michal. *Interaktivní vizualizace povětrnostních vlivů pomocí zařízení SandBox*. Brno, 2024. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce doc. Ing. Vítězslav Beran, Ph.D.

Interaktivní vizualizace povětrnostních vlivů pomocí zařízení SandBox

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana doc. Ing. Vítězslava Berana, Ph.D. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Michal Ondrejka
6. května 2024

Poděkování

Týmto by som veľmi rád poďakoval doc. Ing. Vítězslavovi Beranovi, Ph.D. za poskytnutý čas, cenné rady a odbornú pomoc pri riešení tejto práce.

Obsah

Seznam obrázků

Kapitola 1

Úvod

V dnešnej dobe, kedy sa spoločnosť neustále vyvíja a moderné technológie sa stávajú súčasťou každodenného života, sa otvárajú dvere k novým a inovatívnym formám vzdelávania a vizualizácie. Tu nastáva obrovská príležitosť preskúmať interaktívne možnosti simulácie vody prostredníctvom platformy AR SandBox. V rôznych vzdelávacích inštitúciách sa stretávame s modernými nástrojmi, ktoré spájajú ľudskú hravosť a zvedavosť. Takéto metódy vzdelávania sú pútavejšie a efektívnejšie ako klasické zastaralé metódy. Rozšírená realita v podaní SandBox spája poznanie hmatom a zrakom, kedy užívateľ úpravou prostredia zažíva unikátny zážitok. Jemný piesok, ktorý sa zvyčajne používa v takýchto zariadeniach a dotyk s ním, má okrem vzdelávacieho aspektu naviac pozitívny a upokojujúci vplyv na človeka. Ďalšou výhodou takéhoto prístroja je, že je určený pre všetky vekové kategórie, takže nech už je umiestnený kdekoľvek, bude naplno využívaný.

Výber témy bakalárskej práce *Interaktívna vizualizácia poveternostných vplyvov pomocou zariadenie SandBox* má dva dôvody. Ako bývalého športovca orientačného behu ma veľmi zaujala možnosť pracovať s výškovými datami a terénom. Vzhľadom k vynikajúcim skúsenostiam s výškovými mapami, som rád, že teraz môžem aplikovať svoje skúsenosti. Ako študent informatiky som rád, že môžem zdokonaliť svoje schopnosti vo viacerých významných oblastiach naraz. Táto práca má pre mňa potenciál pre zlepšenie v oblastiach užívateľského rozhrania, spracovania obrazu, simulácií a neurónových sietí.

Cieľom tejto práce je navrhnúť a vytvoriť vzdelávaciu aplikáciu, ktorá bude zobrazovať simuláciu tekutín v reálnom čase. Aplikácia má slúžiť ako vzdelávací prostriedok v oblasti topografie a hydrológie. Bude obsahovať projekciu na piesok a spracovanie vstupov. Užívateľ bude mať k dispozícii interaktívne prvky na úspešné vytvorenie tekutín. Následne bude môcť sledovať tok tekutín v reálnom čase. Okrem tejto hlavnej funkcionality bude môcť užívateľ upravovať terén simulácie ako v klasickom AR SandBox zariadení, kedy sa textúra terénu zmení na základe momentálne zvoleného dizajnu.

Táto práca sa skladá z niekoľkých častí. Teoretická časť je zameraná na priblíženie teoretického základu ako je analýza už existujúcich riešení a lepšiemu pochopeniu problematiky rozpoznávania gest rúk, simulácii tekutín a iných potrebných teoretických celkov. Návrh riešenia obsahuje dekompozíciu problému, analýzu zadania, návrh platformy a architektúru systému. V realizačnej časti je popísaná implementácia jednotlivých kľúčových častí aplikácie, použité technológie a následné testovanie s koncovými užívateľmi.

Kapitola 2

Teoretická časť a prieskum existujúcich riešení

Teoretická časť tejto práce zohráva kľúčovú úlohu pri poskytovaní dôkladného a podrobného rámca pre pochopenie problematiky, ktorú sa chystám skúmať. Zameriam sa na kľúčové teoretické koncepty a metódy, ktoré poskytnú čitateľovi základ pre lepšie pochopenie tejto práce. Na začiatku pojednáva o existujúcich riešeniach. Následne je popísaná prirodzená interakcia gestami. Venuje sa problematikou rozpoznávania gest rúk a simuláciou vody. Na záver tejto kapitoly je priblíženie spôsobu testovania užívateľmi.

2.1 Existujúce riešenia

Rozvoj moderných technológií viedol k vzniku nástrojov v oblasti, ktorú táto bakalárska práca pokrýva. Preto v tejto sekcii analyzujem a zhodnocujem už existujúce riešenia a prístupy k simulácii vody. Je možné, že mnohí mohli mať rovnaky nápad ako ja, avšak kľúčovým faktorom je premenenie myšlienky do reálneho diela. Zisťujem, aké nástroje a technológie sú v súčasnosti dostupné a zároveň sa snažím identifikovať miesto na inováciu a zlepšenie v rámci existujúcich riešení. Táto analýza mi lepšie umožní pochopiť súčasný stav problematiky a poskytne mi východisko pre ďalší vývoj v tejto oblasi.

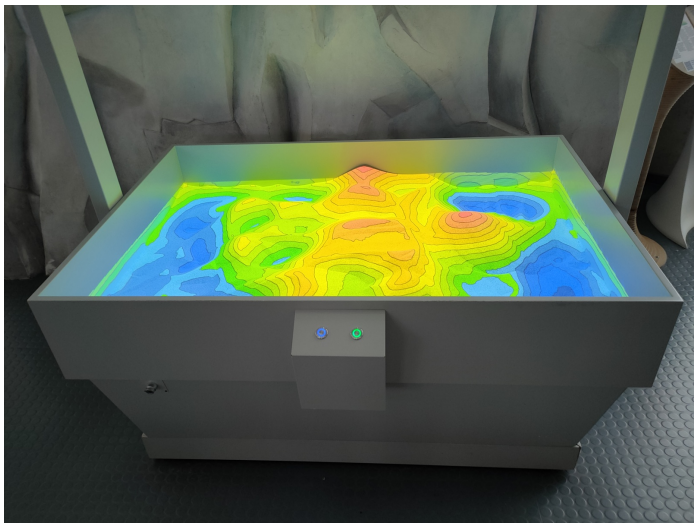
AR SandBox

AR SandBox, je 3D, interaktívny, dynamický vzdelávací nástroj, ktorý pomáha vizualizovať a experimentovať s topografiou v reálnom čase. Tento nástroj používa hĺbkový senzor na meranie vzdialenosti, špecializovaný počítačový softvér na spracovanie hĺbkových dát a projektor, ktorý zobrazí výslednu textúru. Cieľom je zobraziť geografické a geologické procesy. Je to zábavný spôsob učenia sa hrou pre všetky vekové kategórie.

V práci *3D Výuková aplikácia s využitím hĺbkových senzorov* [1] sa autor venuje návrhom a stavbou samotného SandBoxu a jednoduchého softwaru na výber mapy, ktorej terén užívateľ interaktívne vymodeluje z piesku. Jednou z tém, ktorým sa autor v tejto práci venuje je kalibrácia. Toto je veľmi závažný problém kedy pri použití viacerých zariadení, ktoré majú vlastný súradnicový systém, je nutné nájsť vzťah medzi ich súradnicovými systémami. Moje riešenie je postavné nad týmto fyzickým riešením, obrázok 2.1.

Obrázek 2.1: Fyzické riešenie AR SandBoxu.

Platforma od spoločnosti *ar-sandbox.eu*[1] predstavuje efektívne a najprepracovanejšie riešenie. Toto unikátne prostredie ponúka užívateľom základnú funkcionalitu obohatenú o interaktívne prvky, ako sú tlačidlá (obrázok 2.2) a dotyková obrazovka, ktoré umožňujú ľahké a efektívne úpravy dizajnov. Užívateľ dokáže detailne meniť farby dizajnu podľa svojich preferencii a následne si obrázok aj exportovať. Pri interakcii rukou vznikajú v textúre menšie defekty na okrajoch ruky, ale inak veľmi dobre rozpoznávajú v hĺbkových dátach medzi rukou a terénom.



Obrázek 2.2: Riešenie od spoločnosti *ar-sandbox.eu* s tlačidlami.

[1] https://ar-sandbox.eu/

Na *kalifornskej univerzite vo výskumnom centre DataLab*[2] sa podarilo vytvoriť rozšírenie klasického zariadenia AR SandBox o simuláciu vody, obrázok 2.3. Na textúru terénu bol použitý klasický farebný gradient. Pre získanie informácií o hĺbke terénu bola využitá staršia verzia hĺbkového senzora Kinect v1. Toto riešenie malo obrovský úspech a je umiestnené na tisíckach miest po celom svete. Taktiež toto riešnie prezentovali v Bielom dome vo Washingtone D.C. Tu sa potvrdzuje obrovský potenciál tohto zariadenia.

Obrázek 2.3: Riešenie z kalifornskej univerzity.

2.2 Interaktívne systémy a prirodzená interakcia gestami

Užívatelia očakávajú intuitívne a prirodzené spôsoby komunikácie s digitálnymi prostriedkami a tak sa do popredia vývoja užívateľských rozhraní dostáva prirodzená interakcia gestami. Táto sekcia sa venuje preskúmaniu a zhodnoteniu prístupov k prirodzenej interakcii gestami. Táto interakcia je kľúčová v mojej práci a je pre mňa veľmi dôležité, aby bola interakcia s mojím riešením efektívna a použiteľná. Okrem toho je v tejto časti popísaný postup pri vývoji interaktívneho systému.

Human computer interaction (HCI), inak nazvané Man-Machine Interaction (MMI), alebo computer–human interaction (CHI) je obor, ktorý sa zaoberá návrhom, implementáciou, a používaním interaktívnych systémov a ako ovplyvňujú jednotlivcov, organizácie a spoločnosť [2]. Zahŕňa nielen jednoduchosť použitia, ale aj vývoj nových interaktívnych techník na podporu potrieb používateľa, ktoré poskytujú lepší prístup k informáciám. Zaoberá sa, ako vstupné a výstupné zariadenia zobrazujú a príjmajú informácie. Cieľom je vytvoriť použiteľný a bezpečný systém. Pre splnenie tohto cieľu musí vývojár [2]:

- **Pochopiť faktory**, ktoré určujú, ako ľudia používajú danú technológiu.
- **Vyvinúť nástroje** a techniku, ktorá umožní vybudovať vhodný systém.
- **Dosiahnuť efektívnu** a bezpečnú interakciu.

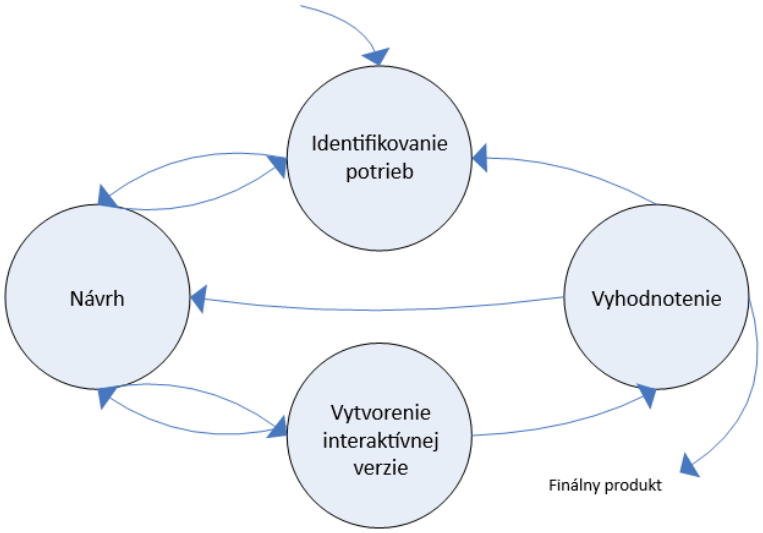
[2] https://datalab.ucdavis.edu/ar-sandbox/

- **Dať potreby užívateľa na prvé miesto.**

Interakciu možno vnímať ako dialóg medzi počítačom a používateľom. Pri splnení týchto požiadavok a pochopení, ako užívateľ komunikuje s počítačom, môžeme navrhovať lepšie systémy.

Je rozdiel medzi gestom ruky (angl. hand gesture) a pozíciou ruky (angl. hand posture). Gesto ruky je definované ako dynamický pohyb skladajúci sa z postupnosti rôznych pozícií rúk v krátkom čase, ako je na príklad mávanie. Návrh gest ruky vyžaduje základné pochopenie anatómie ľudských rúk s cieľom určiť, aké spôsoby držania ruky sú pohodlné. Ľudská ruka má zložitú anatomickú štruktúru pozostávajúcu z mnohých častí a kĺbov, zahŕňajúcich komplexné vzťahy medzi nimi poskytujúce celkovo zhruba 27 stupňov voľnosti [3]. Stupne voľnosti označujú počet základných spôsobov, akými sa pevný objekt môže pohybovať v 3D priestore. Dobré gesto by malo byť dostatočne odlišné od ostatných gest, aby ho vedel program a užívateľ správne rozpoznať. Najznámejšie a najpoužívanejšie gestá sú gestá posunkovej reči.

Pred návrhom systému je dôležité analyzovať potrebné požiadavky na systém. Schéma návrhu interaktívneho systému, využívajúci gesta alebo inú modalitu pre interakciu je na obrázku 2.4.



Obrázek 2.4: Vývoj interaktívneho systému[3].

V prvej fáze: **Identifikovanie potrieb** je dôležité položiť si otázky ako na príklad: Kto je užívateľ? Aké má užívateľ potreby? Aké sú obmedzenia systému a užívateľov?

Vo fáze: **Návrh** sú všetky predošlé analýzy použité na návrh systému. Návrh grafického užívateľského rozhrania určuje, ako bude systém vyzerať a návrh interakcie definuje interakciu so systémom.

[3] https://www.pling.org.uk/cs/doi.html

Tretia fáza: **Vytvorenie interaktívnej verzie**. Rýchle vytváranie interaktívnych prototypov je spôsob ako skoro a lacno identifikovať problémy s použiteľnosťou pred vynaložením veľkého množstva zdrojov. Ide o otestovanie použiteľnosti. S použitím prototypov dokážeme lepšie porozumieť potrebám používateľov.

Posledným krokom je fáza **Vyhodnotenie**, kedy sa produkt testuje a vyhodnocuje sa použiteľnosť systému. Vykonávajú ho užívatelia prevedením určitých úloch s prototypom, zatiaľ čo pozorovatelia zaznamenávajú poznámky o tom, čo každý užívateľ robí a hovorí. Cieľom väčšiny testov je odhaliť všetky problémy, ktorými sa testéri stretnú, aby sme mohli tieto problémy opraviť.

2.3 Užívateľské rozhranie a testovanie

Z minulej kapitoly vyplýva, že je nevyhnutné venovať pozornosť užívateľom a ich skúsenostiam s interaktívnymi systémami. Táto kapitola sa zameriava na významný aspekt vývoja užívateľsky zameraných produktov. Cieľom tejto kapitoly je poskytnúť pohľad na metódy a postupy testovania, ktoré umožnia systematicky zlepšiť a zdokonaliť užívateľskú skúsenosť a užívateľské rozhranie. Uvedomujem si, že intuitívnosť a prívetivosť výsledného produktu mojej práce nespočívajú v technickej dokonalosti mojej práce, ale predovšetkým v schopnosti efektívne komunikovať s užívateľmi. V nasledujúcich odsekoch tejto kapitoly budem podrobne skúmať spôsoby testovania s užívateľmi.

Návrh užívateľského rozhrania (User Interface - UI) [4] je proces vytvárania rozhraní v softvéri alebo počítačových zariadeniach so zameraním na vzhľad alebo štýl. Návrhári sa snažia vytvárať návrhy, ktoré používatelia považujú za jednoduché a príjemné. Návrh používateľského rozhrania sa zvyčajne vzťahuje na grafické používateľské rozhrania, ale zahŕňa aj iné rozhrania, napríklad hlasom ovládané.

Užívateľská skúsenosť (User Experience - UX) [4] je o úplnom pochopení používateľa. Návrhári UX vykonávajú používateľsky prieskum, vytvárajú používateľské persony a testujú výkon a použiteľnosť, aby zistili, ktoré návrhy sú najefektívnejšie pri vedení užívateľa k jeho konečnému cieľu tým najpríjemnejším možným spôsobom. Používateľská persona je fiktívna postava alebo reprezentácia skupiny užívateľov, ktorá pomáha návrhárom a vývojárom lepšie porozumieť potrebám a preferenciám cieľových užívateľov.

Testovanie nie je len o hľadaní a opravovaní chýb. Ide tiež o zabezpečenie toho, aby produkty spĺňali potreby a očakávania koncových používateľov, ktorí ich budú používať v reálnych scenároch. Testovanie sa dá rozdeliť do šiestich krokov:

1. **Identifikovanie koncových používateľov**: Prvým krokom je identifikovať koncových užívateľov (kto bude s aplikáciou narábať). Toto sa dá zistiť pomocou prieskumu, rozhovorov alebo pozorovaním. Táto informácia umožňuje pochopiť ciele, preferencie a kontext používania.

2. **Definovanie požiadavok používateľa**: Druhým krokom je definovanie užívateľských požiadaviek, čo sú vlastnosti a funkcie, ktoré softvérový produkt musí mať, aby uspokojil potreby koncových užívateľov. Požiadavky sa dajú získať analýzou údajov z prieskumu a uprednostnením najdôležitejších požiadaviek.

3. **Návrh používateľského rozhrania**: Tretím krokom je návrh používateľského rozhrania, ktoré je súčasťou softvérového produktu, s ktorým koncoví používatelia interagujú. Aplikuju sa tu princípy návrhu používateľského rozhrania, ako je jednoduchosť, konzistencia, dostupnosť a intuitívnosť. Na vytvorenie a otestovanie návrhu používateľského rozhrania pred jeho implementáciou sa používajú nástroje, ako sú wireframy, mockupy alebo prototypy.

4. **Implementácia používateľského rozhrania**: Štvrtým krokom je implementácia užívateľského rozhrania, cez ktoré budú užívatelia komunikovať so systémom. Odporúča sa dodržiavať osvedčené postupov vývoja softvéru, ako sú kontrola verzií, testovanie a ladenie.

5. **Vyhodnotenie používateľskej skúsenosti**: Piatym krokom je vyhodnotenie používateľskej skúsenosti, čo je celkový dojem a spokojnosť koncových používateľov so softvérovým produktom. Dá sa zistiť používateľským testovaním, kedy pozorujeme a zhromažďujeme spätnu väzbu od koncových používateľov, ktorí používajú softvérový produkt v realistických alebo simulovaných situáciách.

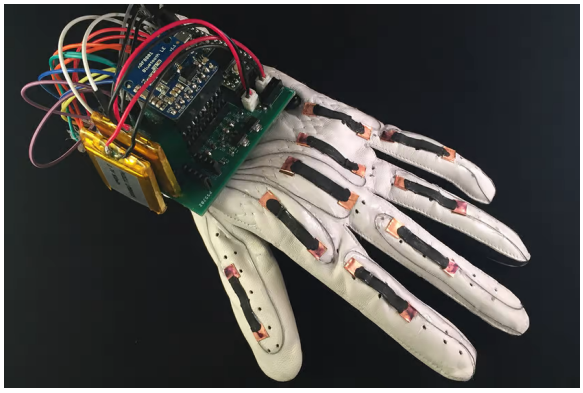
6. **Opakované získavanie spätnej väzby od užívateľov**: Šiestym a posledným krokom je opakované získavanie spätnej väzby od používateľov. Získavať takúto informáciu dokážeme pomocou prečítania spätnej väzby a komentárov produktu. Takto sa identifikujú silné a slabé stránky softvérového riešenia. Na plánovanie a vykonávanie zmien a vylepšení, ktoré zlepšia softvérový produkt a uspokoja potreby koncových používateľov, sa používajú nástroje ako backlog alebo sprint.

2.4 Rozpoznávanie gest rúk

Súčasťou tejto práce je rozpoznávanie gest rúk. K tomu je potreba túto komplexnú tému naštudovať a pochopiť. Na začiatku tejto kapitoly zaoberám rozborom dostupných vstupných zariadení. Následne sa venujem preskúmaním troch hlavných spôsobov rozpoznávania gest rúk.

Rukavica so senzormi

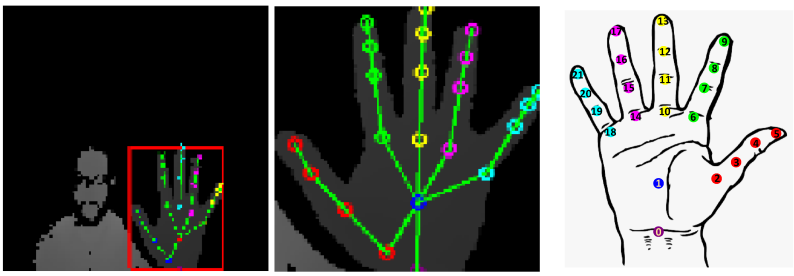
Rukavice sú vybavené senzormi a sledovacími prvkami, pomocou ktorých zaznamenávajú pohyby rúk a prstov, vďaka čomu vie software sledovať a interpretovať gestá. Táto metóda poskytuje presný záznam pohybov. V súčasnej dobe existuje viacero prístupov k rozpoznávaniu gest rukami pomocou rukavíc. Existuje celá rada riešení a zariadení, napríklad v papieri [5] bola použitá rukavica s piatimi flex senzormi a Inertial Measurement Unit (IMU) na rozpoznávanie gesta. Bol použitý algoritmus, ktorý zistil orientáciu prstov a orientáciu celej ruky. Príklad takejto rukavice je vyobrazený na obrázku 2.5.



Obrázek 2.5: Príklad rukavice na rozpoznávanie gest rúk[4].

Rozpoznávanie za použitia hĺbkových dát

S rozvojom hardvérovej technológie sú nízke náklady na hĺbkové kamery, a tak sa takýto typ senzoru stal veľmi stabilným nástrojom pre spracovanie obrazu. 3D kamera získa dáta vo forme hĺbkových dát, ktoré sa následne musia spracovať. V práci [6] je spracovanie týchto dát pomocou algoritmu, ktorý sa skladá zo štyroch časti: detekcia ruky, analýza tvaru ruky, analýza trajekcie ruky, klasifikácia. Toto riešenie nie je citlivé na zmenu osvetlenia a funguje aj v absolútnej tme. Na obrázku 2.6 je možno vidno postup detekcie. Najprv sa získajú dáta, ktoré obsahujú ruku a následne sa z nich získajú orientačné body ruky.



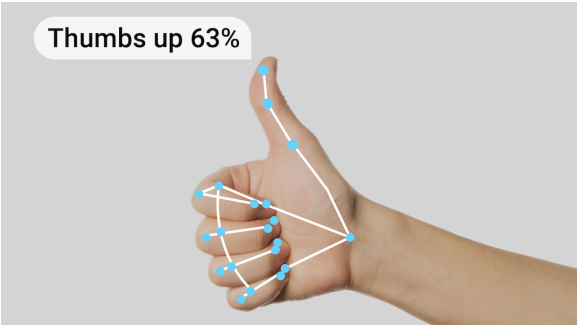
Obrázek 2.6: Príklad spracovania hĺbkových dát [16].

Rozpoznávanie za použitia kamery

Táto metóda je V praxi najpoužívanejšia. Takéto systémy s rozpoznávaním gest sa v posledných rokoch výrazne zlepšili. Ľudia v dnešnej dobe majú prístup k výkonným zariadeniam. Táto metóda pozostáva zo štyroch základných krokov ako je získavanie obrazu, predbežné spracovanie, extrakcia prvkov a klasifikácia. Získavanie obrazu prebieha pomocou rôznych typov senzorov, ktoré dokážu zachytiť obraz objektu. Predbežné spracovanie surových dát je

[4] https://newatlas.com/sign-language-translate-glove/50474/

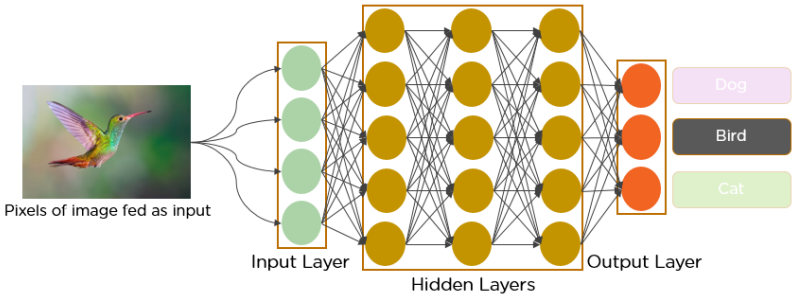
veľmi dôležité. Cieľom je extrahovať gesto oblasti od pozadia obrázka. Dôležitá je efektívna segmentácia rúk, kedy sa obrázok oreže iba na časť s rukou, aby rozpoznávanie prebiehalo čo najrýchlejšie a nemuselo spracovať zbytočné dáta. Proces segmentácie závisí od typu gesta, ak ide o dynamické gesto, potom je potrebné gesto ruky lokalizovať a sledovať, ak ide o statické gesto, vstupný obrázok musí byť iba segmentovaný. Výsledné orientačné body, typ pózy a pravdepodobnosť istoty môžeme vidieť na obrázku 2.7.



Obrázek 2.7: Rozpoznávanie za použitia farebnej kamery pomocou knižnice MediaPipe[5].

Aj keď je tento typ veľmi populárny, prináša zopár prekážok, ktoré je treba prekonať. Tieto výzvy zahŕňajú rôzne kontexty, viaceré interpretácie a časopriestorové variácie gest. V práci [7] sa spomína problém s detekciou farby pleti, keďže takýto systém je veľmi citlivý na svetelné podmienky. Tento problém nastáva nielen pri detekcii farby pleti, ale aj pri detekcii gest. Tu sa ukazuje náročnosť natrénovania flexibilného modelu, ktorý bude spoľahlivo zvládať tento problém. V článku [8] je opísaný problém s komplexnými pozadiami a rôznymi orientáciami rúk.

Na rozpoznanie gest rúk pomocou *deep learning* sa používa viacero typov neurónových sietí. Najprv sa použije konvolučná neurónová sieť (obrázok 2.8). CNN (Konvolučná neurónová sieť alebo ConvNet) [9] je typ doprednej umelej siete, ktorej neuróny sú inšpirované organizáciou zrakovej kôry zvierat. CNN využíva priestorové korelácie, ktoré existujú so vstupnými údajmi. Konvolučná sieť sa používa na trénovanie modelov na súbore obrazových údajov. Výstupom CNN je súbor informácií o polohe kĺbov ruky. Následne sú tieto dáta vložené do ďalšej neurónovej siete, ktorá vráti, o aký typ gesta sa jedná.



Obrázek 2.8: Schéma konvolučnej neurónovej siete.

[5] https://developers.google.com/mediapipe/solutions/vision/gesture_recognizer

Na systém, ktorý je schopný rozpoznávať gestá ruky, je nutné nazbierať dostatočné množstvo dát, vytvoriť model a následne natrénovať a otestovať model. Zber dát môžeme chápať ako samotné zbieranie inormácie (získavanie obrázkov z kamery) a priradenie dôležitých poznatkov k daným informáciám (označenie, kde na obrázku sa nachádza ruka). Takéto označené dáta sa rozdelia v pomere približne 9 : 1 na dáta na trénovanie a dáta na testovanie. Algoritmy potrebujú na efektívne učenie veľké množstvo vysokokvalitných údajov. Je dôležité nazbierať kvalitné a rôznorodé dáta, aby bol model čo najefektívnejší. Vo fáze trénovania model dostáva trénovacie dáta a učí sa ich význam. Vo fáze testovania model prechádza cez súbor testovacích údajov, aby sa vyhodnotila presnosť a úspešnosť modelu. Pre nekritické systémy stačí úspešnosť nad 80 %. Keď sa dosiahne požadovaná účinnosť úpravou parametrov modelu, ako sú počty neurónových uzlov a váhy, potom je možné takýto model používať.

2.5 Simulácia tekutín

V práci je hlavný prvok simulácia tekutín, a tak je potrebné nájsť efektívne metódy, ktoré tento problém riešia. Vzhľadom na to, že mám k dispozícii iba pohľad zvrchu, nie je nutné hľadať komplexné algoritmy na dokonalé zobrazenie vody ako sa používajú v najnovších hrách. Namiesto toho sa zameriam na výpočetne nenáročnejšie spôsoby.

Makroskopická a mikroskopická škála

V makroskopickej škále je tekutina súvislý materiál s určitými vlastnosťami (hustota, rýchlosť) a jej správanie možno opísať Navierovými-Stokesovými rovnicami (viac viď ďalej). V mikroskopickej škále je tekutina vnímaná ako súbor častíc, a preto veličiny ako hustota alebo rýchlosť nadobúdajú úplne iný význam[6]. Pre komplexitu a výpočetnú náročnosť simulácie v mikroskopickej škále a potreby zobraziť simuláciu iba zvrchu sa budem venovať makroskopickej škále.

Výpočtová dynamika tekutín

Výpočtová dynamika tekutín (anglicky: Computational fluid dynamics - CFD) [10] je veda, ktorá s pomocou digitálnych počítačov vytvára predpovede javov prúdenia tekutín na základe zákonov zachovania (zachovanie hmoty, hybnosti a energie), ktorými sa riadi pohyb tekutín. CFD metódy fungujú na makroskopickej škále a používajú Navier-Strokové rovnice.

Navier-Strokové rovnice

Navier-Strokové rovnice sú často používané pre nestlačiteľné tekutiny [11], ktoré opisujú pohyb viskóznych tekutých látok. Tieto rovnice vznikajú z aplikácie druhého Newtonovho zákona na pohyb tekutiny. Ich formulácia vychádza z princípov zachovania hybnosti a hmotnosti. Navier-Stokesove rovnice vo vektorovej forme pre nestlačiteľnú tekutinu sú vyjadrené nasledovne [12]:

$$a = -\frac{\nabla p}{\rho} + \nu \nabla^2 u + g \tag{2.1}$$

[6] https://feaforall.com/creating-cfd-solver-lattice-boltzmann-method/

$$\nabla \cdot u = 0 \tag{2.2}$$

,kde:

- a je zrýchlenie.
- p je tlak.
- ρ je hustota.
- ν je kinematická viskozita.
- u je velocita.
- g je gravitačné zrýchlenie, zvyčajne 9.8 $\mathrm{m/s^2}$.

Rovnica (2.1) vyjadruje, že každá častica tekutiny sa zrýchľuje po tlakovom gradiente, že velocity blízkych oblastí tekutiny sú podobné a že reaguje na vonkajšie sily, ako je gravitácia. Rovnica (2.2) iba hovorí, že hustota tekutiny, musí byť všade rovnaká.

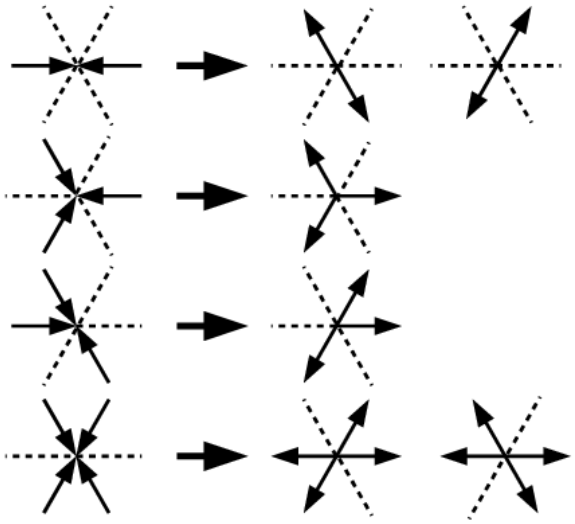
Stavové automaty

Pri simulácii tekutín je jedným z prístupov pravidelné rozdelenie priestoru na menšie časti. Je potrebná štruktúra, ktorá umožní šíriť zmeny v okolitých častiach. Tomuto sa venujú stavové automaty, ktoré dokážu jednoducho simulovať tekutiny.

Stavový automat[7] je kolekcia buniek na mriežke špecifického tvaru, ktorá sa vyvíja v niekoľkých diskrétnych časových krokoch podľa súboru pravidiel založených na stave susedných buniek. Pravidlá sa aplikujú iteračne na toľko časových krokov, koľko systém potrebuje. Von Neumann bol jedným z prvých ľudí, ktorí uvažovali o takomto modeli. Jeden z najznámejších je *Game of Life* (Hra Života)[8]. Je možné rozdeliť priestor simulácie na práve takúto mriežku a pristupovať k simulácii tekutiny práve pomocou celulárneho automatu.

Lattice Gas Automaton (LGA) je typ celulárneho automatu používaného na simuláciu tokov tekutín. LGA nevyžaduje riešenie zložitých rovníc a spája výhodu nízkych výpočtových nákladov a schopnosti napodobňovania realistickej dynamiky tekutín. Na rozdiel od celulárneho automatu je mriežka zložená z trojuholníkov, teda každá častica sa môže hýbať v šiestich smeroch [12]. Pravidlá kolízií sú deterministické, obrázok 2.9.

[7] https://mathworld.wolfram.com/CellularAutomaton.html
[8] https://en.wikipedia.org/wiki/Conway%27s_Game_of_Life



Obrázek 2.9: Schéma mriežky a kolízie pri Lattice Gas Automaton.

Smoothed particle hydrodynamics

Smoothed-particle hydrodynamics[9] (SPH) je výpočtová metóda používaná na simuláciu súvislých mechaník, ako je mechanika pevných látok a prúdenie tekutín. Vyvinuli ho Gingold a Monaghan a Lucy v roku 1977, pôvodne pre astrofyzikálne problémy. Používa sa v mnohých oblastiach výskumu vrátane astrofyziky, balistiky, vulkanológie a oceánografie. Je to meshfree Lagrangeova metóda (kde sa súradnice pohybujú s tekutinou bez mriežky) a rezolúcia metódy môže byť ľahko upravená s ohľadom na premenné, ako je hustota.

V práci [13] sa spomínajú výhody tejto metódy. Dynamická interakcia je riešená automaticky, pretože formulácia je Lagrangianovská. Vlastnosti tekutín spojené s rôznymi fázami môžu byť ľahko začlenené. Metóda je numericky stabilná.

Avšak, papier taktiež popisuje aj nasledovné nevýhody: Aproximácia okrajových podmienok. Nedostatok modelovania turbulencií. Veľké kolísanie tlaku v dôsledku rušivých tlakových vĺn v neošetrenej slabo stlačiteľnej formulácii. Tvorba dutín. Pri veľkom počte malých častíc vysoké výpočtové náklady.

Táto metóda je založená na tom, že tekutina je reprezentovaná časticami, ktoré majú nejakú polohu a velocitu. Je dosť všeobecná na použitie v simulácii akejkoľvek tekutiny. Túto metódu je možné rozšíriť o špecifickejšie vlastnosti [15], pre dosiahnutie ešte presnejšej simulácie tekutín. Pre jednotlivé častice môžeme odvodiť viskozitné a tlakové silové polia za použitia Navier-Strokovej rovnice.

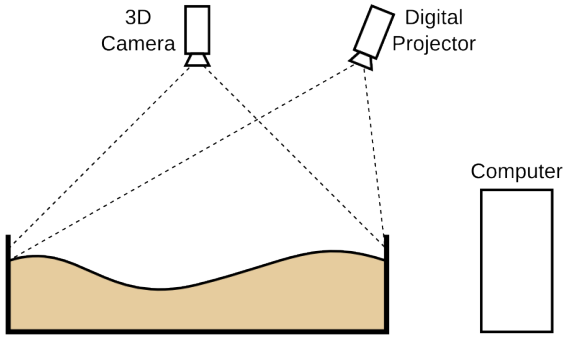
Po výpočte polohy, velocity a iných požadovaných vlastností je potrebné častice zobraziť ako tekutinu. Keďže je reprezentovaná časticami bez správneho zobrazenia, nevyzerá ako tekutina. Používajú sa rôzne metódy na zobrazovanie, no v mojom riešení je pohľad na simuláciu fixne zhora, takže nepotrebujem komplexné troj-dimenzionálne zobrazovanie.

[9] https://en.wikipedia.org/wiki/Smoothed-particle_hydrodynamics

2.6 Kalibrácia a senzory

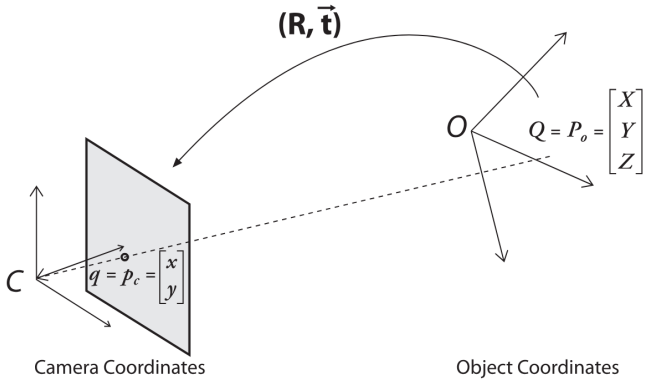
Kalibrácia kamery je základnou úlohou počítačového videnia, ktorá je kľúčová v rôznych aplikáciách, ako je 3D rekonštrukcia, sledovanie objektov, rozšírená realita a analýza obrazu. Presná kalibrácia zaisťuje presné merania a spoľahlivú analýzu pomocou korekcie skreslení a odhadu vnútorných a vonkajších parametrov kamery. V tejto kapitole sa budem venovať práve tomuto problému. Okrem toho popíšem aké senzory obsahuje Kinect v2, ich schopnosti a limity.

Ná obrázku 2.10 je zobrazená poloha 3D kamery a projektoru v klasickom zariadení AR SandBox. Vzhľadom na to, že sa kamera a projektor nachádzajú v dvoch rozdielnych bodoch, vidia dve rôzne súradnicové sústavy. Je nutné nájsť vzťahy medzi nimi.



Obrázek 2.10: Obrázok polohy 3D kamery a projektoru v zariadení AR SandBox.

Túto prekážku je možné prekonať pomocou rotačnej matice. Pre každý záber konkrétneho objektu, ktorý kamera sníma, môžeme opísať jeho polohu vzhľadom na kamerový súradnicový systém v zmysle rotácie a posunu. Táto rotrácia a translácia je vyobrazená na obrázku 2.11.



Obrázek 2.11: Rotácia a translácia súradníc.

Vo všeobecnosti možno rotáciu opísať ako násobenie súradnicového vektore štvorcovou maticou vhodnej veľkosti. V konečnom dôsledku je rotácia ekvivalentná zavedeniu nového

popisu polohy bodu v inom súradnicovom systéme. Rotácia v trojrozmernom priestore sa dá rozložiť na dve dvojrozmerné rotácie okolo každej osi.

Rotačná matica R má tú vlastnosť, že jej inverzná matica je zároveň jej transponovaná matica, teda platí $R^T R = RR^T = I$, kde I je identita. Ak rotujeme okolo osí x, y a z postupne s príslušným uhlami ψ, ϕ a θ, výsledkom je matica celkovej rotácie R, ktorá je daná produktom troch matíc, $R = R_x(\psi), R_y(\phi), R_z(\theta)$, kde:

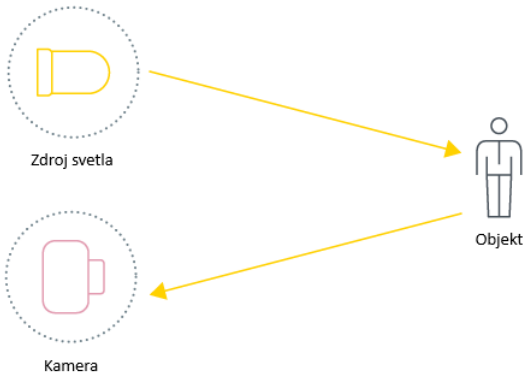
$$R_x(\psi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\psi & \sin\psi \\ 0 & -\sin\psi & \cos\psi \end{bmatrix} \tag{2.3}$$

$$R_y(\phi) = \begin{bmatrix} \cos\phi & 0 & -\sin\phi \\ 0 & 1 & 0 \\ \sin\phi & 0 & \cos\psi \end{bmatrix} \tag{2.4}$$

$$R_z(\theta) = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.5}$$

Kinect

Jedno z najpoužívanejších zariadení s hĺbkovým senzorom je práve Kinect[10]. V mojom riešení mám k dispozícii Kinect v2. Hĺbkový senzor funguje na princípe Time of Flight (ToF). Infračervený projektor na Kinecte vysiela modulované infračervené svetlo, ktoré potom zachytí senzor (obrázok 2.12). Infračervené svetlo odrážajúce sa od bližších objektov bude mať kratší čas letu ako tie vzdialenejšie, takže infračervený senzor zachytí, ako veľmi bol modulačný vzor deformovaný od času letu, pixel po pixeli. Takéto meranie je presné a vypočítané za krátky čas, čo umožňuje detekovať viac snímok za sekundu. Rozlíšenie takéhoto senzora je 512 x 424 pixelov so snímovaciou frekvenciou 30 snímkov za sekundu. Hĺbkový rozsah je 0.5m – 4.5m [17]. Jednotlivé pixely obsahujú vzdialenosť v mm od senzora. Zorné pole hĺbkového senzora je 70.6° × 60°. Farebná kamera má zorné pole 84.1° × 53.8°. Farebná kamera má rozlíšenie 1920 × 1080 px (2 Mpx).



Obrázek 2.12: Princíp technológie ToF.

[10] https://en.wikipedia.org/wiki/Kinect

Kapitola 3

Návrh riešenia

Táto časť rieši kreatívny proces navrhovania a konceptualizácie projektu, ktorý vznikol s cieľom vylepšiť a rozšíriť už existujúcu platformu AR SandBox. Konkrétne sa zameriavam na inováciu v oblasti interaktívnych zážitkov a vzdelávania prostredníctvom virtuálneho prostredia a fyzickej manipulácie s pieskom. Podrobný návrh riešenia je kritickým krokom v procese vytvárania, kde prechádzam od teoretických konceptov k praktickým aplikáciám vedomostí. V nasledujúcich kapitolách sa venujem návrhu užívateľského rozhrania a integráciu riešenia do existujúceho prostredia, tak aby bol vytvorený súvislý a pôsobivý zážitok pre užívateľa.

3.1 Analýza zadania

V rámci analýzy zadania sa budem dôkladne venovať kľúčovým aspektom z pohľadu koncových užívateľov a klientov. Zameriam sa na identifikáciu a rozbor potrieb, očakávaní a požiadavok. Počas analýzy sa sústredím na jednotlivé užívateľské scenáre a prípady použitia, aby som získal hlbšie porozumenie tomu, ako budú užívatelia používať výsledný produkt.

Užívatelia majú možnosť zadávať všetky potrebné údaje a informácie presne, aby simulácia prebiehala podľa ich predstáv. Vzhľadom k tomu, že takéto vzdelávacie zariadenia sa nachádzajú predovšetkým v múzeách, na výstavách a v iných vzdelávacích inštitúciách, hľadám riešenie, ktoré bude prívetivé pre užívateľov všetkých vekových kategórii. Cieľová skupina užívateľov zahŕňa deti až po dospelých, a preto musí moje riešenie zohľadňovať všeobecný prístup, aby všetci užívatelia mohli ľahko a bez problémov interagovať so simuláciou.

Zmena plochy terénu

Jeden zo základných prvkov AR SandBoxu je možnosť upravovať terén presýpaním piesku. Užívateľ má schopnosť presunúť ľubovoľné množstvo piesku, pričom túto interakciu môže vykonávať viacero užívateľov naraz. Všetky tieto zmeny sú zaznamenané hĺbkovým senzorom, ktorý je popísaný v kapitole 2.6. Po získaní dát výpočtové zariadenie vygeneruje textúru, ktorá sa následne pomocou projektoru zobrazuje na ploche AR SandBoxu. Systém potom cez projektor ukazuje novú projekciu. Vďaka tejto mechanike je možné vizuálne rozlíšiť a zvýrazniť body s rovnakou výškou, čím sa zlepšuje celkový vizuálny výstup.

Zmena dizajnu terénu

Ďalším možným prípadom užitia je schopnosť meniť textúru terénu. Pre moje riešenie navrhujem týchto päť vzhľadov:

- **Farebný dizajn**. Textúra je zafarbená od najnižšieho bodu po najvyšší týmito farbami, v tomto poradí: modrá, zelená, žltá a červená.
- **Farebný dizajn s vrstevnicami**. Na rozdiel od predošlého dizajnu má viac farieb, princíp farebného prechodu ostane rovnaký. Taktiež zobrazuje vrstevnice, ktoré budú zobrazené medzi jednotlivými prahami farieb. Vrstevnice majú čiernu farbu.
- **Prirodzený dizajn**. Tento dizajn vyzerá ako terén v miernom podnebnom pásme. Najnižšie body majú textúru trávy, potom je textúra lesu, skalnatá textúra a v najvyšších bodoch sneh.
- **Vulkanický dizajn**. V tomto type je najmenej textúr. Textúry majú rôzne skalnaté a vulkanické textúry.
- **Exotický dizajn**. Na rozdiel od predošlých dizajnov má v nízkych až stredných výškach textúru vody, tak aby výsledná plocha vyzerala ako oceán so súostrovím.

Ovládanie simulácie

Simulácia je plne ovládaná užívateľom. Užívateľ zadáva vstup v ploche SandBoxu nad terénom. Tento vstup môže byť zadávaný kdekoľvek v celom prostredí zariadenia. Systém následne zaznamená súradnice tohto vstupu a v danej oblasti vytvorí tekutinu. Užívateľ môže zadávať vstup pre vytvorenie tekutiny aj keď už prebieha simulácia. Simulácia má dva typy tekutín. Simulácia vody, kedy simulovaná tekutina má podobné vlastnosti ako voda. Simulácia lávy, kedy simulovaná tekutina má hustotu podobnú hustote lávy. Typ simulovanej tekutiny závisí od momentálne zvoleného dizajnu. Okrem hustoy tekutín je ešte rozdiel vo farbe aby užívateľ vedel intuitívne rozpoznať, že ide o iný typ tekutiny. Farba lávy je červená až svetlo-oranžová a farba vody bude tyrkysová.

Správa systému

Okrem užívateľov je dôležité analyzovať aj prípady použitia správcom zariadenia, keďže takýto typ zariadenia býva na verejných miestach. Správca má pri spustení aplikácie možnosť nakalibrovať zariadenie a spustiť rozšírenú realitu so simuláciou. Je schopný efektívne a precízne nakalibrovať projekciu, aby sa správne zobrazovala na teréne AR SandBoxu.

3.2 Užívateľská interakcia

V tejto kapitole sa zamýšľam nad efektívnymi spôsobmi interakcie v rámci výukovej platformy AR SandBox, využívajúcej senzor Kinect. Okrem základnej interakcie, ktorou je manipulácia s pieskom, sa venujem hľadaniu optimálnych spôsobov zadávania ďalších vstupov. Zameriavam sa na vytvorenie interakcie, ktorá nielenže využíva technologický potenciál zariadenia, ale tiež poskytuje užívateľom pútavý a pohodlný spôsob ovládania simulácie.

Jeden z najväčších problémov pri návrhu tohto projektu je, akým spôsobom bude užívateľ so systémom komunikovať vstupy spomínané v predošlej kapitole (3.1). Vzhľadom na to, že zariadenie AR SandBox má k dispozícii málo vstupných zariadení, pomocou ktorých užívateľ môže komunikovať so systémom, som dosť obmedzený. Musím zvážiť výhody a obmedzenia jednotlivých zariadení. Platforma AR SandBox obsahuje dataprojektor, hĺbkový senzor, kameru a výpočtový systém.

Interakcia s pieskom

Základným spôsobom ovládania je manipulácia s pieskom pomocou rúk, ktorá sa zaznamená hĺbkovým senzorom, čo predstavuje kľúčovú súčasť každého zariadenia SandBox. Užívateľ je schopný fyzicky presúvať piesok a tým formovať diverzifikovaný terén. Táto interakcia je vyobrazená na obrázku 3.1. Tento vstup nie je obmedzený počtom rúk v ploche SandBoxu, teda hocikoľko užívateľov môže presýpať piesok naraz. Avšak je dôležité, aby sa textúra nemenila na základe rúk, ale iba na základe terénu. Hĺbkový senzor nevie rozoznať, o aký predmet sa jedná. Aplikácia dokáže do značnej miery ignorovať hĺbkové dáta rúk.

Obrázek 3.1: Interakcia viacerých užívateľov s pieskom.

Avšak, pre dosiahnutie plného rozsahu funkčnosti a zabezpečenie efektívneho, presného a jednoduchého zadávania ďalších vstupov, je potreba preskúmať ďalšie inovatívne riešenie. Interaktívna výuková platforma AR SandBox využíva vstupné zariadenie Kinect, ktoré disponuje niekoľkými senzormi s obrovským potenciálom pre interakciu.

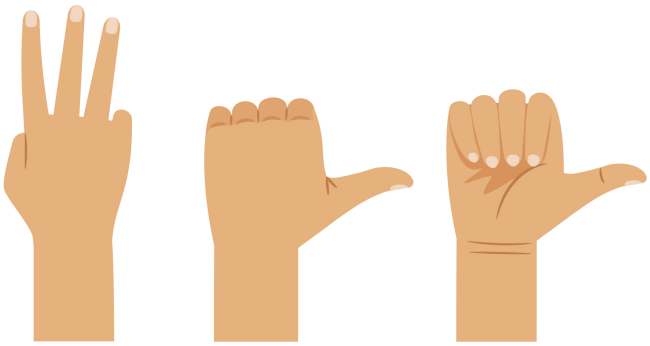
V priebehu prieskumu existujúcich riešení som narazil na platformu od spoločnosti AR-SandBox, viď 2.1, ktorá sa snažila riešiť túto výzvu pridaním tlačidiel, ktoré spúšťajú určité akcie po ich stlačení. Rovnako som objavil riešenie, ktoré zahŕňalo zabudovanú dotykovú obrazovku. Hoci tieto možnosti predstavujú jednoduché riešenia, nespĺňajú plný potenciál, ktorý AR SandBox ponúka.

Interakcia so simuláciou

Preto som sa rozhodol pre inovatívny prístup - integrovať rozpoznávanie gest rúk. Toto riešenie ponúka užívateľom interaktívnu a jednoduchú manipuláciu so simuláciou vody, umožňujúc im vyjadrovať svoje požiadavky. Týmto spôsobom sa snažím posunúť hranice interaktivity v AR prostredí a ponúknuť užívateľom nový zážitok v rámci vzdelávacieho

prostredia AR SandBox. Je nutné integrovať dve gestá. Gesto pre vytvorenie tekutiny a gesto pre zmenu dizajnu simulácie.

V mojom riešení navrhujem gesto pre vytvorenie tekutiny ako sekvenciu póz vyobrazených na obrázku 3.2 vľavo. Pre zmenu vhľadu terénu navrhujem gesto pozostávajúce z póz z obrázku 3.2 v strede alebo vpravo. Ak palec ukazuje doprava tri sekundy, dizaj terénu sa zmení na nasledujúci. Ak palec smeruje doľava dizajn sa prepne na predošlý.



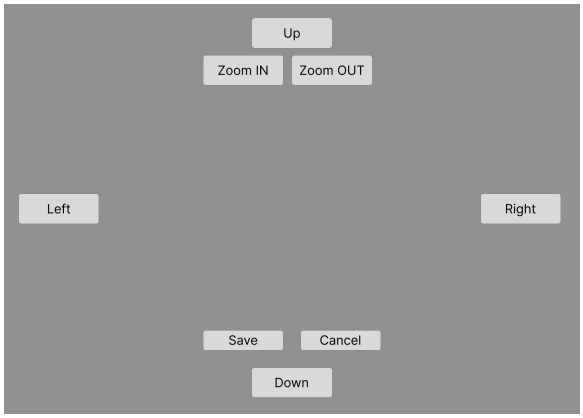
Obrázek 3.2: Navrhované pózy rúk.

Z teoretického prieskumu problematiky interakcie pomocou gest (viď 2.2) som sa rozhodol využiť prirodzené a jednoduché gestá. Gestá su taktiež navrhnuté dostatočne rozdielne, aby ich systém vedel správne rozoznať. Treba si uvedomiť, že pri interakcii s pieskom užívateľ umiestňuje sovju ruku pred kinect senzor. Nemôžem teda použiť gestá, ktoré by mohol používateľ omylom zadať pri interakcii s terénom.

3.3 Kalibrácia systému

Ako som už spomínal, je nutné vytvoriť kalibračný systém na kalibráciu AR SandBoxu. Dataprojektor nepokrýva celý terén. Správca musí mať možnosť pohybovať virtuálnu kameru v trojrozmernom priestore, aby vedel správne napasovať textúru na terén a správne sa vyfarbili body s rovnakou výškou. Pri pohybe sa mení transformácia virtuálnej kamery v priestore. Ďalej musí mať možnosť rotovať hĺbkové dáta, keďže hĺbkový senzor nie je upevnený presne nad stredom terénu a dáta z neho nie sú ortografické. Získané hĺbkové dáta sú rotované pomocou rotačnej matice spomínanej v skecií 2.6. Táto rotácia okrem správneho vygenerovania textúry, zaručí presnejší tok tekutiny po teréne.

Operátor systému dokáže kalibrovať pomocou klávesnice a myši. Pri spustení aplikácie sa zobrazí hlavné menu, z ktorého je možné spustiť simuláciu alebo nakalibrovať systém. Na obrázku 3.3 je navrhované užívateľské rozhranie kalibrácie, ktorý umožňuje pohybovať kameru v simulácii smerom hore, dole, doľava a doprava. Taktiež správca systému môže priblížiť alebo vzdialiť textúru. S týmito prvkami je možné presne nakalaibrovať simuláciu. Nie je nutné, aby bolo toto rozhranie dizajnovo prepracované, keďže slúži iba na rýchlu kalibráciu pred spustením hlavnej časti. Kinect senzor sa nachádza na citlivom kĺbe, preto som sa rozhodol pre takéto riešenie kalibrácie.



Obrázek 3.3: Návrh UI na kalibráciu.

Pri kalibrácii sa zobrazí na pozadí užívateľského rozhrania (obrázok 3.3) farebný dizajn spomínaný v kapitole 3.1, keďže tento typ je najvhodnejší na kalibráciu z dôvodu kontrastru jednotlivých textúr.

3.4 Architektúra systému

V tejto kapitole sa detailne venujem architektúre môjho simulačného systému, ktorý je navrhnutý s cieľom interaktívne a realisticky vizualizovať pohyb vody. Architektúra systému zohľadňuje nielen spracovanie obrazu a hĺbkových dát, ale aj interakciu s užívateľom prostredníctvom gest a dynamického upravovania simulácie.

Princíp spracovnia rámcov Kinectu

Pravidelne sa získavajú z hĺbkového senzoru Kinectu hĺbkové dáta. Tie je následne potreba pomocou rotačnej matice zrotovať. Senzor nevie rozpoznať, či sa jedná o terén, alebo práve užívateľ presýpa piesok rukou. Preto systém obsahuje prahovaciu funkciu, ktorá aktualizuje hĺbky na základe nastavených prahov, aby sa do modulu, kde sa vytvára textúra terénu, nedostali hĺbkové dáta ruky. Z kapitoly 2.6 viem, že rozlíšenie hĺbkového senzora, 512 x 424 je pomerne nekvalitné. Okrem zlého rozlíšenia meranie hĺbky pixelov nie je dokonalé a pri statickom objekte nevracia konštantné hodnoty. Nasnímané hodnoty je nutné vyhladiť pomocou lineárnej interpolácie. Bez vyhladenia nebude vyzerať vygenerovaná textúra korektne. Zorné pole hĺbkového senzora je rozdielné od klasických kamier, preto treba spracovať hĺbkové dáta aj vzhľadom na tento problém.

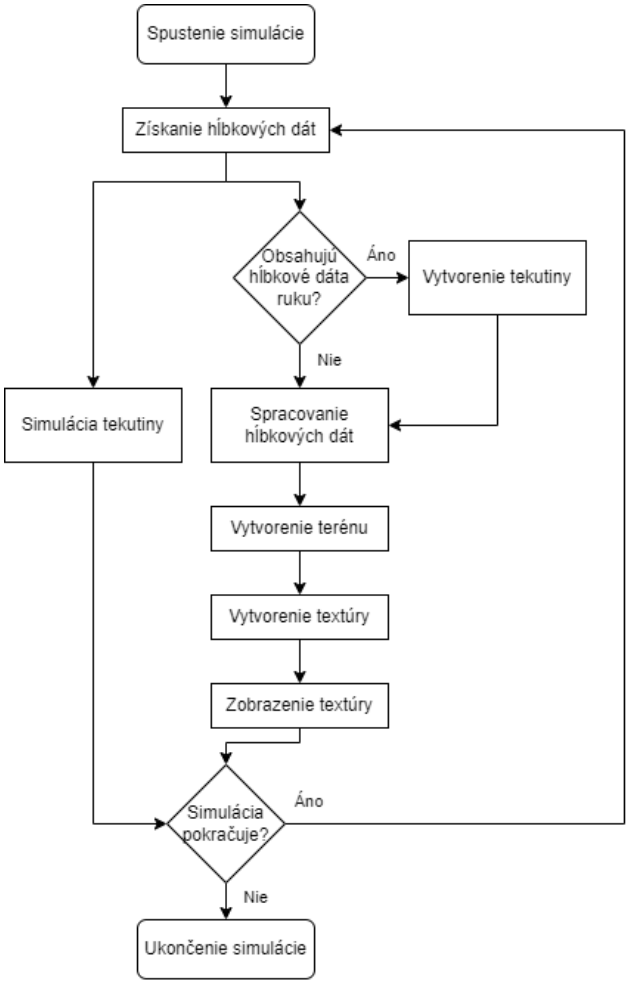
Texúrovanie terénu

Spracované hĺbkové dáta sa použijú na vytvorenie troj-dimenzionálneho terénu vo virtuálnom priestore. Rozmery terénu sú určené rozlíšením hĺbkového senzoru. Jednotlivé výšky terénu sú nastavené podľa získaných hĺbkových dát. Po aktualizácii výšok terénu sa musí aktualizovať textúra terénu, keďže princíp AR SandBoxu je, že je generovaná na základe bodov s rovnakou výškou. Textúra terénu má rovnaké rozlíšenie ako hĺbkový senzor. Prechádzajú sa jednotlivé body a nastavuje sa textúra v danom bode. Textúra sa nastavuje podľa momentálne zvoleného dizajnu a výšky terénu.

Rozpoznávanie gest

Rozpoznávanie rúk je riešené pomocou neurónových sietí, viď 2.4. Systém avšak nemôže používať klasickú kameru, keďže vďaka projektoru je pri zadávaní gesta projekcia zobrazená aj na ruke. Z bodu kamery potom nie je dostačne rozoznať obrysy ruky. Preto systém využíva prvod hĺbkových dát na šedotónový obrázok. Súčasťou riešenia je modul pre vytváranie takýchto obrázkov. Tieto obrázky sú najprv použité na zaznamenanie gest, aby som mohol pretrénovať už existujúci model, keďže natrénovanie modelu na novo by bolo náročne a v mojej práci na to nemám priestor. Následne sa tento pretrénovaný model použije vo výslednej aplikácii. Je schopný rozoznať požadované gestá s dostatočnou presnosťou a spoľahlivosťou. Rozpoznávanie je dosť náročné zdroje, preto treba do modelu posúvať v pravidelom intervale obrázky.

Architektúra simulácie je pomocou vývojového diagramu vyobrazená na obrázku 3.4.



Obrázek 3.4: Vývojový digram architektúry simulácie tekutiny.

3.5 Priebeh simulácie

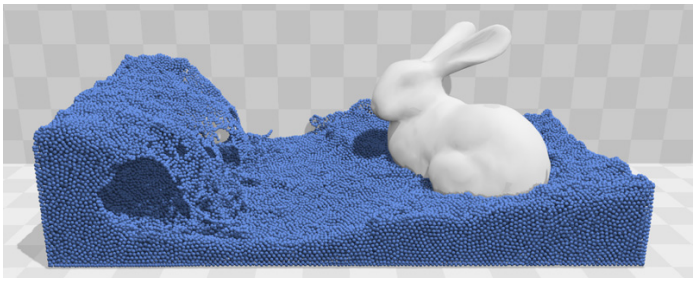
Simulácia vody je neoddeliteľnou súčasťou mojho projektu v AR Sandbox, ktorý umožňuje interaktívne modelovať a vizualizovať simuláciu. V tejto kapitole preskúmam priebeh simulácie vody. Simulácia vody neslúži iba na estetické účely, ale aj na prieskum vplyvu topografie na tok vody. V rámci mojej práce skúmam a experimentujem s dvoma prístupmi ku simulácii vody. Jeden je založený na Lattice Gas Automaton a druhý na Smoothed particle hydrodynamics.

Lattice Gas Automaton

Jeden z možných spôsobov je práve Lattice Gas Automaton. Začína sa definovaním počiatočného stavu systému, kde každá bunka v priestore reprezentuje určitú oblasť a má pridelený stav. Následne sa vykonávajú iteratívne kroky simulácie, pri ktorých sa aktualizuje stav každej bunky na základe pravidiel pohybu vody a jej interakcii s okolím. Tok vody je modelovaný prostredníctvom jednoduchých pravidiel pre pohyb vody v jednotlivých smeroch. Tieto pravidlá zohľadňujú fyzikálne vlastnosti vody, ako je na príklad gravitácia, interakcia medzi časticami, atď. V každej iterácii simulácie sa aplikujú tieto pravidlá na každú bunku, čo vedie k postupnej zmene stavu a pohybu vody v priestore. Dôležitým prvkom simulácie je vizualizácia výsledkov. Táto časť zahŕňa vytvorenie grafických reprezentácií pohybu vody, kde môžeme sledovať vývoj systému v čase. Výsledky simulácie sú prezentované vo forme animácií, ktoré umožňujú ľahkú interpretáciu zmien v priestore. Vizualizácia vody je vyobrazená nad základnou projekciou hĺbkových dát, kedy sa prefarbia body, v ktorých sa voda nachádza.

Smoothed Particle Hydrodynamics

Následuje návrh simulácie s využitím metódy Smoothed Particle Hydrodynamics (viď 2.5). Tekutina sa generuje na základe polohy gesta pre vytváranie simulovanej tekutiny. Tekutina je reprezentovaná ako množina častic. Majú tvar sféry, pôsobí na nich gravitácia, kolízie a trenie. Jedna častica sama o sebe nevyzerá ako tekutina, ale pri väčšiom počte dokážeme aproximovať tok tekutiny. Simulácia využíva častice a ich vzájomné vzťahy k realizácii vývoja masy vody v čase. Toto je vyobrazené na obrázku 3.5. Výsledkom tohto procesu je tekutina, ktorá sa v reálnom čase simuluje. Ďalším dôležitým krokom je renderovanie simulácie. Čisto zobrazovanie takýchto častíc nevyzerá dostatočne. Preto je potreba aplikovať metódu, ktorá zobrazí častice ako objekt pripomínajúci vodu. Pri veľkom počte častíc a komplexnom teréne dochádza k veľkým počtom kolíziam, čo je náročne na zdroje. Aplikácia bude teda obmedzená počtom simulovaných častíc, aby bola dosiahnutá plynulosť.



Obrázek 3.5: Príklad Smoothed Particle Hydrodynamics bez úpravy renderovania.

Simulácia zohľadňuje dostupné zdroje. Keďže to nie je jediný prvok systému, je dôležité, aby simulácia mala konštantnú rýchlosť. To pri veľkom počte simulovanej tekutiny nemusí platiť. Preto je systém obmedzený množstvom simulovanej vody. Po zadaní gestá pre vytvorenie tekutiny, je potreba aby sa najprv overilo, či je v danom čase možné pridať do simulácie viac tekutiny. Okrem tohoto obmedzenia, je dôležité si uvedomiť, že zo zariadením bude pracovať viacero užívateľov v krátkom časovom intervale. Systém neobsahuje gesto pre reštartovanie simulácie a tak je tento problém riešený časovačom, ktorý sa pri každej interakcii pomocou gesta vynulujú. Ak časovač dosiahne určitú hodnotu, tak sa vymaže všetka tekutina. Táto hodnota je získaná pomocou užívateľského testovania tak, aby užívateľ mal dosť času na simulovanie vody, aby sa mu náhodou simulácia neskončila skôr a aby ďalší užívateľ nemusel zbytočne čakať. Pri simulácii užívateľ zadá gesto a potom čaká a sleduje simuláciu. Toto treba zohladniť v časovači, aby sa náhodou simulácia neresetovala kvôli neaktivite, keď užívateľ chcel ešte sledovať tento proces.

Moje riešenie obsahuje dva typy tekutín, ktoré sú simulované v závislosti na momentálne zvolenom dizajne terénu. Hlavná simulácia je simulácia vody, ktorá čo najpresnejšie pripomína reálny tok vody terénom. Láva má na rozdiel od vody väčšiu hustotu a tečie pomalšie. Tento rozdiel je možné jednoducho vyriešiť tak, že sa zmení farba tekutiny a spomalí sa čas simulácie. Pri pomalšiom priebehu to pripomína tok hustej látky, v tomto prípade lávy.

Kapitola 4

Realizácia, experimenty a vyhodnotenie

V tejto časti sa zameriam na praktickú realizáciu navrhovaného riešenia z predošlej kapitoly. Mojím cieľom je integrovať interakciu s tekutinami do existujúceho prostredia AR SandBoxu, kde užívatelia môžu používať gestá ruky na riadenie pohybu a manipuláciu s virtuálnymi tekutinami. Zaoberám sa technickými detailmi implementácie, vrátane výberu vhodných algoritmov. Následne je popísané prevedené testovanie užívateľmi a jeho výsledky. Na záver sa venujem vyhodnoteniu výsledkov mojej práce.

4.1 Použité technológie

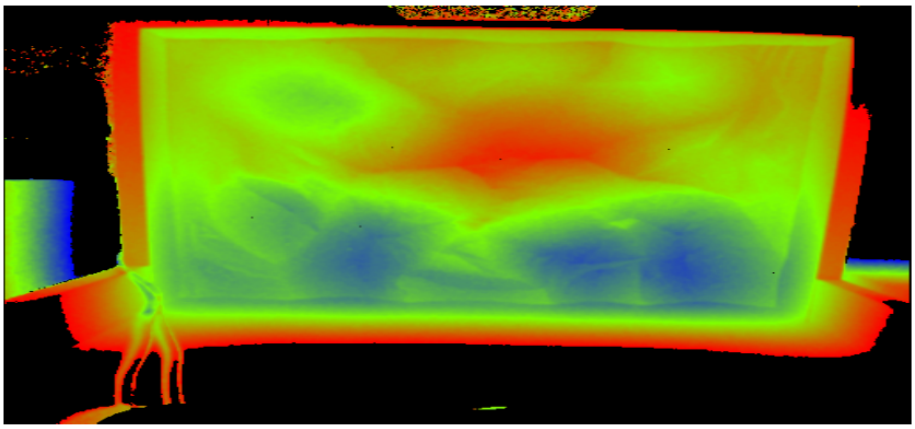
Vzhľadom na to, že moje riešenie pokrýva veľké množstvo rôznych tematických okruhov, je výber veľmi dôležitý výber správnych technológií. V tejto časti overujem hardverový základ. Detailne sa zameriavam na softvérové nástroje, ako *Kinect for Windows 2.0* na spracovanie hĺbkových dát, vývojové prostredie *Unity* na programovanie samotnej aplikácie a technológie *TensorFlow*, *MediaPipe*, ktoré mi umožňia rozpoznávať objekty v obrázkoch.

Moje riešenie je postavené už nad existujúcim fyzickým riešením AR SandBoxu. Avšak tu je priestor na pár rýchlych úprav, aby som mal čo najlepší základ pre vývoj mojej aplikácie. Pôvodny senzor Kinect v1 z roku 2009 som nahradil novším senzorom Kinect v2. Kinect v1 fungoval na základe vysielania bodkovaného vzora, ktorý sa na objekte deformoval a na základe toho získaval hĺbkové dáta. Kvôli tomu, že aplikáciu vyvíjam na operačnom systéme Windows 11, som sa rozhodol vymeniť tento senzor spolu s jeho upevnením. Túto zmenu je možné vidieť na obrázku 4.1. Okrem toho balíček Kinect for *Windows SDK 2.0* a senzor Kinect v1 nie sú kompatibilné. Touto úpravou som získal presnejší a kvalitnejší zdroj hĺbkových dát.

Obrázek 4.1: Kinect v2 a dataprojektor.

Hĺbkovéé dáta získavam pomocou spomínaného *Kinect for Windows SDK 2.0*, čo je vlastne vývojový balík od spoločnosti Microsoft, ktorý umožňuje tvorbu aplikácií pre Windows využívajúcich senzor Kinect. Knižnica poskytuje nástroje na rozpoznávanie pohybov, gest tela, hlasu a sledovanie telesnej polohy. Pri použití *Kinect for Windows SDK 2.0* sa hĺbkové dáta generujú ako udalosti. Takúto udalosť môžeme zachytiť v našej aplikácii. V prostredí *Unity* využívam *Multi Source Manager*, ktorý je súčasťou balíčka, pomocou ktorého synchronizujem viacero zdrojov dát z Kinect senzora. Inak povedané zachycuje eventy a dovoľuje mi jednoducho pristupovať k dátam.

Najprv som skúšal generovať textúru terénu v čistom *c#*. Podarilo sa mi vygenerovať farebný gradient (viď obrázok 4.2), no kvôli komplexite ostatných dizajnov som sa rozhodol použiť *Unity*. *Unity* je výkonný herný engine a vývojové prostredie, ktoré poskytuje robustné nástroje na tvorbu interaktívnych virtuálnych prostredí vrátane realistických terénov a detailných textúr. Obsahuje vstavaný systém na vytváranie terénov, ktorý umožňuje modelovanie a texturovanie terénov. Poskytuje nástroje na optimalizáciu výkonu, čo je dôležité pre simulačné aplikácie. V editore sú k dispozícii nástroje pre tvorbu UI, ktoré využívam pri tvorbe kalibračného modulu.



Obrázek 4.2: Farebný gradient získaný na základe hĺbky

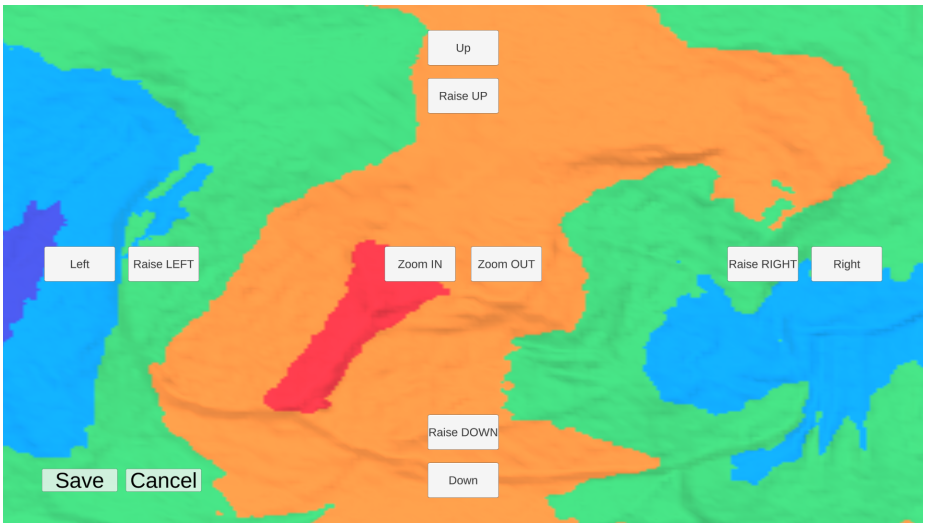
Posledným technologickým prvkom je rozpoznávanie gest rúk. *Kinect for Windows SDK 2.0* obsahuje natrénovaný model na rozpoznávanie tela, ale neobsahuje natrénovaný model

na rozpoznávanie gest rúk. Pre svoju prácu budem potrebovať ďalšie nástroje alebo knižnice na rozpoznávanie gest rúk, ktoré umožnia identifikáciu a interpretáciu pozícii a gest rúk pomocou kamery Kinect. Tieto nástroje mi umožnia integrovať rozpoznávanie gest rúk do mojej aplikácie a vytvoriť riešenie, ktoré môže reagovať na rôzne gestá vykonávané pred kamerou. Použil som *TensorFlow*, čo je open-source knižnica na strojové učenie vyvinutá spoločnosťou Google. Je známa pre svoju flexibilitu a robustnosť pri implementácii rôznych typov modelov strojového učenia vrátane neurónových sietí. *TensorFlow* poskytuje nástroje na vytváranie, trénovanie a nasadenie modelov, vrátane modelov na rozpoznávanie a klasifikáciu obrazových dát. *MediaPipe* je ďalšia populárna knižnica od spoločnosti Google, ktorá poskytuje nástroje na spracovanie multimediálnych dát, vrátane obrazu a videa. Je postavená nad knižnicou *TensorFlow*. *MediaPipe* obsahuje preddefinované modely a nástroje na detekciu a sledovanie objektov v reálnom čase, ako aj na rozpoznávanie gest a pohybov. Integrácia týchto knižníc do mojej aplikácie mi umožní rozpoznávať gestá rúk.

4.2 Kalibrácia

V tejto kapitole sa venujem dôležitému procesu kalibrácie, ktorý je kľúčovým prvkom optimalizácie a presnosti môjho systému. Kalibrácia je neoddeliteľnou súčasťou každej technologicko-vedeckej aplikácie, ktorá pracuje so senzormi a dátami vo virtuálnom prostredí. Cieľom tejto kapitoly je poskytnúť kompletný prehľad o procese kalibrácie a jeho implementácii.

Po spustení aplikácie je možnosť buď spustiť simuláciu, alebo kalibrovať systém. Užívateľské rozhranie (obrázok 4.3) som riešil pomocou *Unity UI system*. Vytvoril som objekty *Button - TextMeshPro*, ktoré na rozdiel od klasických tlačidiel poskytujú kvalitnejšie možnosti úpravy. Následne som každému tlačidlu nastavil, aká funkcia sa má spustiť pri stlačení. Pomocou týchto funkcii a *SceneManager* dokážem prechádzať medzi scénami pri stlačení tlačidiel. Aplikácia má tri scény: hlavné menu, kalibrácia a simulácia.



Obrázek 4.3: Rozhranie kalibračnej scény

Na nespracované hĺbkové dáta je najpr nutné aplikovať rotačnú maticu, keďže senzor sa nenachádza nad stredom plochy terénu a naskenovaný terén teda neodpovedá reálnym výškam. To, ako moc majú byť otočené, dokáže správca systému zadať v kalibračnej časti aplikácie za použitia tlačidieľ *Raise UP*, *Raise LEFT*, *Raise DOWN* a . Podľa stlačeného tlačidla sa rotuje plocha terénu. Pri stlačení sa avšak v *Unity* spustí iba raz zvolená funkcia a ja potrebujem, aby pri držaní tlačidla sa konzistente menily potrebné hodnoty. To som dosiahol pomocou *IPointerDownHandler* a *IPointerUpHandler*, ktoré definujú metódy *OnPointerDown* a *OnPointerUp*, čo su vlastne obsluhy udalostí. Tieto události sa vyvoláva pokiaľ je tlačidlo stlačené. Keďže počet snímkov závisí od dostupných zdrojov a chcem dosiahnuť konzistenciu, využil som *Time.deltaTime*, čo je *float* hodnota v sekundách medzi jednotlivými snímkami.

Okrem rotovania sa dá hýbať virtuálne kamera pomocou tlačidiel *Up*, *Left*, *Down* a *Right*, kedy sa virtuálne kamera podľa stlačeného tlačidla hýbe po x a z osi. Mením tzv. *transform property* kamery, čo je trojrozmerný vektor *Vetor3* so súradniciami kamery. Konzistentné menenie hodnôt pri dlhšom stlačení tlačidla som dosiahol rovnako ako v prípade kalibrácie rotácie. Virtuálne kamera je ortogonálna (bez perspektívy) a je možné meniť veľkosť kamery, čím som dosiahol efekt približovania. Okrem tlačidiel je možne použiť klávesy WASD pre pohyb virtuálnej kamery a EQ pre približovanie.

Kalibračná scéna naviac obsahuje tlačidlá *Save* a *Cancel*, pomocou ktorých je možné zmeny kalibrácie uložiť alebo zrušiť. Perzistenciu kalibračných nastavení som dosiahol pomocou *PlayerPrefs*, ktorý obsahuje metódu *SetFloat(key, value)*, kde kľúč je reťazec s názvom prvku, ktorý chcem perzistentne uložiť a value je hodnotá, ktorá má byť uložená. Následne som schopný pomocou metódy *GetFloat(key, defaultValue)* získať vopred uloženú hodnotu. V prípade, že ešte žiadna nebola uložená, metóda vracia základnú hodnotu. Okrem týchto funkcii tlačidlá *Save* a *Cancel* presmerujú správcu na scénu hlavného menu.

Kalibračné UI má na pozadí za tlačidlami zobrazenú textúru, ktorú dokážeme napasovať na terén a tak správne nakalibrovať aplikáciu. Dizajn textúry je farebný dizajn s vrstevnicami, keďže s kontrastnými farbami je možné prívetivo kalibrovať.

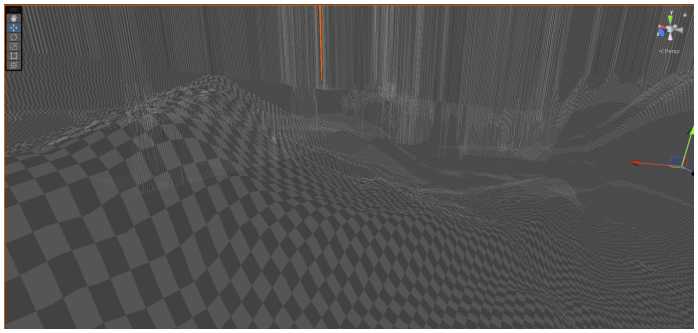
4.3 Terén a textúra

Hĺbkové dáta získavam pomocou *MultiSourceManager* každú snímku. Sú uložené do premennej *rawDepthData*, keďže ich ešte treba spracovať pred vytvorením terénu. Dáta majú formát pole *ushort*, toto pole je jedno-dimenzionálne, čo je dosť nepraktické. Počet hodnôt je 512 * 424, čo je určené rozlíšením hĺbkového senzora.

Hĺbkový senzor má obmedzenia, a preto musím pred vytvorením terénu spracovať dáta. Vytvoril som priemerovaciu funkciu, ktorá na základe okolitých bodov spriemeruje hĺbky a upraví momentálne spracovanú hĺbku tak, aby výsledné hodnoty vyzerali hladšie. Toto mi umožní lepšie vygenerovať hladšiu textúru terénu a lepšie nasimulovať tekutiny.

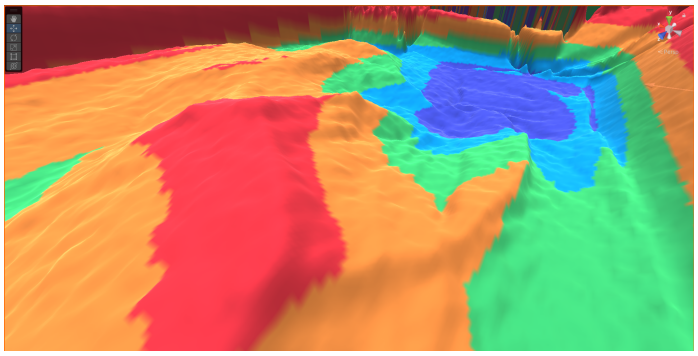
V *Unity* je špecifický objekt *Terrain*, ktorý sa používa na vytváranie terénu v hrách. Tento koncept v mojej práci využívam a na základe získaných hĺbkových dát generujem reprezentatívny terén. Údaje o teréne sú uložené v súbore *TerrainData*. Pre zmenenie výšok terénu je nutné zmeniť výšky v tomto súbore. Najprv získam referenciu na *TerrainData*.

Terén má vždy tvar štvorca a tak nastavujem rozlíšenie terénu na 512*512 pomocou *terrainData.heightmapResolution*. Výšky sa do *terrainData* ukladajú ako *float* s hodnotami od 0 do 1. Hĺbkové dáta teda normalizujem, aby mali hodnotu na tomto intervale. Zmenu výšky celkového terénu som urobil pomocou *terrainData.size*. Výšky sa ukladajú pomocou dvoj-dimenzionálneho pola *float* a tak iterujem cez hĺbkové dáta, normalizujem ich a postupne ukladám adekvátne hodnoty do pola *heights[,]*. Nakoniec pomocou metódy *terrainData.SetHeights(heights)* zmením výšky terénu. Po aktualizácii výšok terénu sa zmení *Terrain* objekt v *Unity*. Príklad výsledného objektu je možné vidieť na obrázku 4.4.



Obrázek 4.4: Terén získaný na základe hĺbkových dát v *Unity*.

Po zmenení výšok je nutné aktualizovať textúru terénu, aby zodpovedala aktuálny výškam (obrázok 4.5). V *Unity* editory sa nachádzajú nástroje pre textúrovanie terénu. Avšak ja potrebujem dynamicky textúrovať počas behu aplikácie, preto som to musel urobil v skripte. *Texture splatting* je metóda kombinovania rôznych textúr na základe *AlphaMap*, čo je pole *float* o dĺžke odpovedajúcej počtu rôznych požadovaných textúr. Hodnoty sa nastavujú od 0 po 1, čo určuje ako moc je daná textúra odhalená. Pomocou cyklu prechádzam body terénu a nastavujem hodnoty *AplhaMapy* na základe výšky bodu. Po nastavení všetkých hodnôt *AlphaMapy* pomocou metódy *terrainData.SetAlphamaps(splatmapData)* aplikujem zmeny. Textúry sú obrázky, ktoré sa k terénu priradzujú v *Unity* editory pomocou *Terrain* komponentu. Textúry po priradení majú index od 0 vzostupne.



Obrázek 4.5: Výsledný terén po aplikovaní textúry v *Unity*.

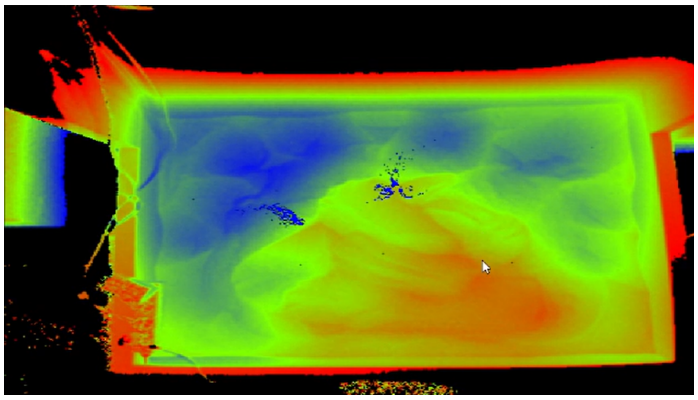
Kvôli osvetleniu vo virtuálnom prostredí som jednotlivým textúram nastavil určité vlastnosti. *Metallic* parameter určuje, do akej miery je textúra terénu "metalická"alebo "lesklá".

Tento paramter som nastavil, aby textúra vyzerala čo nareálnejšie. Okrem toho som nastavil parameter *SizeX* a *SizeY* na základe rozlíšenia textúry, tak aby textúra na teréne mala správnu veľkosť. Kameru vo virtuálnom prostredí som polohoval na terén, tak aby sa ortogonálne pozerala zhora dole na terén.

4.4 Simulácia vody

V tejto kapitole implementujem a porovnávam dve metódy simulácie vody. Prvá metóda využíva Lattice Boltzmann Method a druhá metóda Smoothed Particle Dynamics. Spomeniem, ako som pristupoval k realizácii týchto metód a prípadne prekážky, ktoré som musel prekonať.

Simuláciu pomocou Lattice Boltzman method som skúšal v čistom *c#* kóde, bez *Unity*. Definoval som dvojrozmerné pole o veľkosti rozlíšenia, generovanie ruky som dosiahol iba pomocou klikania myšou. Podarilo sa mi aplikovať základné vlastnosti ako je gravitácia a tok podľa tvaru terénu. Simulovaná voda zobrazená nad farebným gradientom je na obrázku 4.6. Senzor nedokáže presne nasnímať hĺbky, obsahujú teda akýsi šum, ktorý v tomto type metódy dosť prekážal, keď sa voda zasekla na kopci, kvôli lokálnemu minimu. S výlsedkom som nebol spokojný, a tak som sa rozhodol pre metódu Smoothed Particle Dynamics.

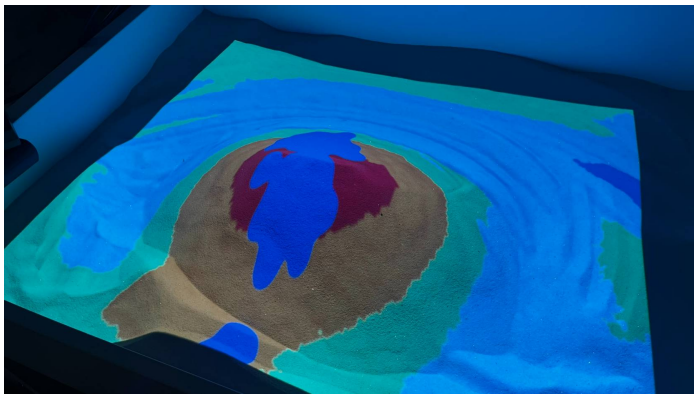


Obrázek 4.6: Simulácia pomocou LBM.

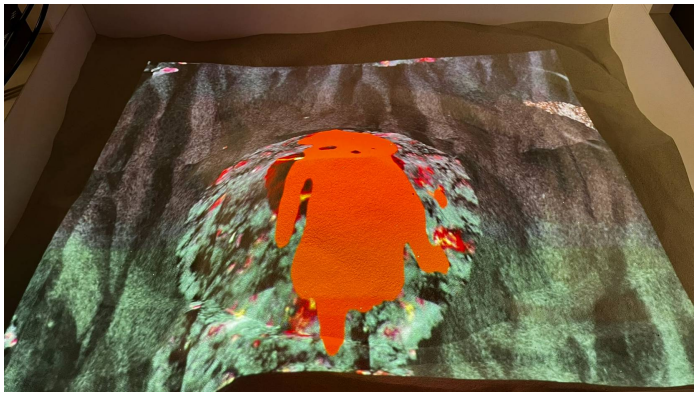
Do finálnej verzii mojej práce som teda použil simuláciu vody pomocou častíc. V *Unity* som vytvoril prefabrikovaný objekt pre jednotlivé častice tekutiny. *Prefab* v *Unity* je znovupoužiteľný objekt, ktorý môžeme vytvoriť, upravovať a použiť viackrát v scéne. Zmeny vykonané na *prefab* sa automaticky prejavia vo všetkých jeho inštanciách v hre. Tomuto objektu som pridal *Sphere Collider*, ktorý slúži na detekciu kolízií a interakcií s ostatnými objektmi. Ďalej som mu pridal *Mesh Renderer* a *Sphere (Mesh Filter)*. Tieto komponenty sú v *Unity* používané na zobrazenie a vykresľovanie 3D geometrie. *Mesh Filter* určuje tvar objektu pomocou siete (*mesh*), zatiaľ čo *Mesh Renderer* zabezpečuje, že táto geometria je viditeľná v hre. Ďalší komponent, ktorý som pridal, je *RigidBody*, ktorý pridáva fyzikálne vlastnosti k objektu, umožňujúc mu reagovať na gravitáciu, kolízie a silu. Posledný komponent tohto prefabrikovaného objektu je skript, ktorý každú snímku kontroluje pozíciu objektu. Ak je pozícia na osi y menšia ako nula, tak sa objekt vymaže. Toto je z dôvodu,

keď užívateľ zvýši terén v oblasti vody, častica sa týmpádom dostane pod terén a keby ju skript nevymazal, padala by donekonečna, čo nie je optimálne na zdroje.

Výsledkom súboru častíc, na ktoré podliehajú fyzikálne zákony ako je gravitácia, trenie a kolízie, je prvok pripomínajúci tok vody. Ďalším krokom je vytvorenie spôsobu, ktorý sférické častice zobrazí ako objekt podobný tekutine. V *prefab* som nenastavoval farbu tekutiny, teda je univerzálny, pre všetky typy tekutín. V scéne simulácie som vytvoril virtuálnu kameru, ktorej som nastavil, že vidí iba častice a následne som použil *shader*, ktorý spracuje vstup z kamery a vytvorí textúru vody. Táto textúra sa zobrazí tesne pred hlavno kamerou. Pixely, ktoré neobsahujú tekutinu, sú priehladné a tak je možné vidieť aj textúru terénu. Výsledná simulácia vody a lávy je zobrazená na obrázkoch 4.7 a 4.8.



Obrázek 4.7: Výsledná simulácia vody pomocou SPH.



Obrázek 4.8: Výsledná simulácia lávy pomocou SPH.

4.5 Rozpoznávanie gest rúk

V tejto časti sa venujem implementácie rozpoznávania gest rúk na základe vopred navrhnutého návrhu. Tento problém predstavuje najväčšiu prekážku v mojom riešení. Zisťujem efektívnosť a použiteľnosť prístupov a zhodnocujem ich vlastnosti.

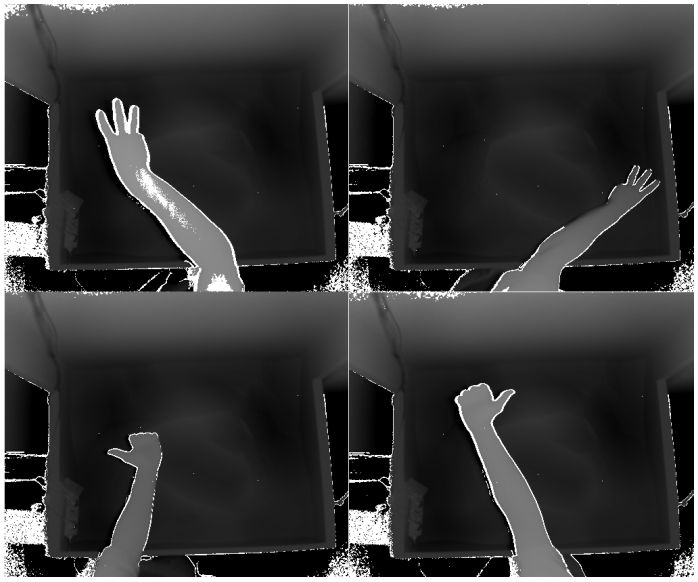
Prvý prístup, ktorý som mal k tomuto problému, bolo použiť klasickú farebnú kameru (obrázok 4.9), keďže v súčasnosti je tento typ rozpoznávania najpoužívanejší a existujú

knižnice, ktoré umožňuju tento problém jednoducho riešiť. Po spojaznení *Tensorflow* a *Mediapipe* som však narazil na obrovský problém, ktorý som v návrhu nezvážil. Senzor Kinect a dataprojektor sú umiestnené tak, že projekcia pri zadávaní gesta osviecuje ruku. Z farebnej kamery sa nie vždý dá rozpoznať ruku, keďže je osvietená projekciou (obrázok 4.10). Z prieskumu som zistil, že sa dá rozpoznávanie robiť aj pomocou hĺbkových dát, no na toto riešenie nemám dostatok priestoru.

Obrázek 4.9: Úspešné rozpoznávanie pri projekcii na ruku.

Obrázek 4.10: Neúspešné rozpoznávanie pri projekcii na ruku.

Preto som sa rozhodol urobiť z hĺbkových dát šedotónový obrázok (viď obrázok 4.11). To sa mi podarilo pomocou skriptu v *Unity*. Hĺbkové dáta sú jednodimenzionálne, čo je na prevod na obrázok nepraktické. Preto sa pred spracovaním hĺbkových dát volá metóda *MakeDepthPicture()*, ktorá deklaruje *Texture2D*, čo je v *Unity* objekt ekvivalentný obrázku. Prechádzam jednotlivé hĺbky, normalizujem ich, prevádzam na tri rovnaké RGB hodnoty a následne aplikujem na textúre. Textúru následne ukladám ako obrázkový súbor. Takto som schopný vytvoriť dáta na natrénovanie modelu, ale aj neskôr posielať tento obrázok do natrénovaného modelu, ktorý rozpozná obrázok.



Obrázek 4.11: Šedotónové obrázky získané z hĺbkových dát.

Následne som spomínané tri pózy ruky z návrhu nasnímal. Vytvoril som obrázky na každú pózu v rôznych častiach plochy a v rôznych výškach. Následne som musel obrázky anotovať, aby model pri trénovaní vedel, kde sa nachádza ruka na daných obrázkoch. Na toto som použil aplikáciu *LabelImg*[1], ktorá slúži na označenie objektov v obrázku. Po označení sa ku každému obrázku vygeneroval *xml* súbor, ktorý obsahoval kľučové informácie ako sú typ gesta, polohu a iné.

Posledným krokom bolo použiť získané dáta na natrénovanie modelu. K detekcií póz ruky som použil model od *TensorFlow* s názvom *MobileNet V2 FPNLite 320*. Nepodarilo sa mi na mojích dátach dosiahnuť úspešnosť viac ako 60 %. Táto nízka by znemožnila ovládať ovládať užívatelovi aplikáciu. Preto som sa nakoniec rozhodol pre detekciu ruky na základe vymaskovaných hĺbkových dát. Takto sa dá pomocou pohybu ruky nad terénom generovať voda. Zmenu štýlu dizajnu som realizoval pomocou klávesy medzerník. Bohužiaľ, je to veľký zásah do môjho návrhu interakcie s aplikáciou, ale s ohľadom na množstvo funkčných blokov, ktoré celá moja práca integruje v jeden komplexný celok, som musel pristúpiť k tomuto obmedzeniu a nevenovať sa detailne iba tomuto funkčnému detekčnému modulu.

Generovanie vody je tým pádom riešené na základe vymaskovaných hĺbkových dát, kedy pomocou spomínanej prahovacej funkcie, pomocou ktorej rozpoznávam medzi dátami ruky a dátami terénu, získam masku. Z tejto masky získam polohu ruky. Vo finálnom riešení nie som schopný rozpoznávať gestá a tak je zmena dizajnu riešená stlačením medzeníka, kedy sa prepne na následujúci dizajn.

4.6 Testovanie s užívateľmi

Posledným prvkom mojej bakalárskej práce je otestovanie môjho riešenia. V tejto kapitole popisujem, akým spôsobom som prevádzal testovanie. Taktiež uvádzam, aké problémy

[1] https://github.com/HumanSignal/labelImg

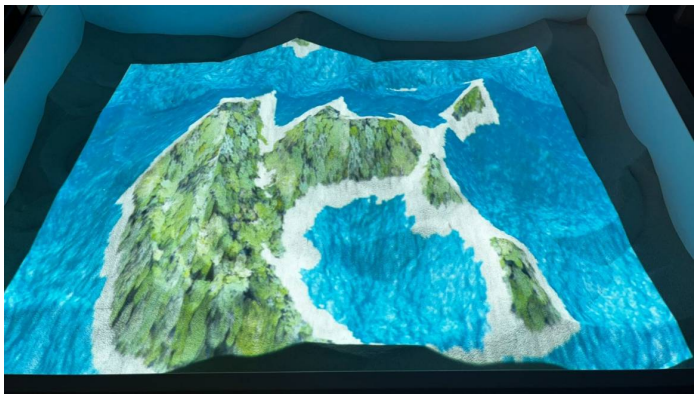
testovanie odhalilo a ich úspešnú, prípadne neúspešnú elimináciu. Okrem problémov sa venujem pozitívnym aspektom môjho riešenia, ktoré koncových užívateľov milo prekvapilo.

Testovanie som rozdelil do dvoch kôl. Tieto kolá používali rozdieľne prístupy ku testovaniu. V prvom kole som koncovým užívateľom nevysvetlil, čo ako funguje. Iba som im spomenul, že je to rozšírená realita, v ktorej sa dajú simulovať tekutiny. V tomto kole užívatelia testovali hlavne intuitívnosť a prívetivosť systému.

Výsledkom tohto testu bolo, že všetci účastnici zistili, ako sa vytvára tekutina, keď omylom umiestnili pred hĺbkový senzor svoju hlavu alebo pri presýpaní piesku zdvihli vyššie ruku. Z tohoto som zistil, že musím prahovacej funkcii nastaviť vyšší prah, aby pri presýpaní piesku nedochádzalo k neúmyselnému vytváraniu tekutiny. Ďalej som zistil, že výškové intervaly jednotlivých textúr neboli precízne nastavené, kedy na príklad pri farebom dizajne nebolo skoro nikdy vidno úplne spodnú modrú textúru. Tento problém nastával aj pri horných textúrach. Zmenu dizajnu terénu som netestoval, keďže sa mi nepodarilo vytvoriť dobrý model na rozpoznávanie gest rúk. Zmena sa prevádza pomocou medzerníka na klávesnici.

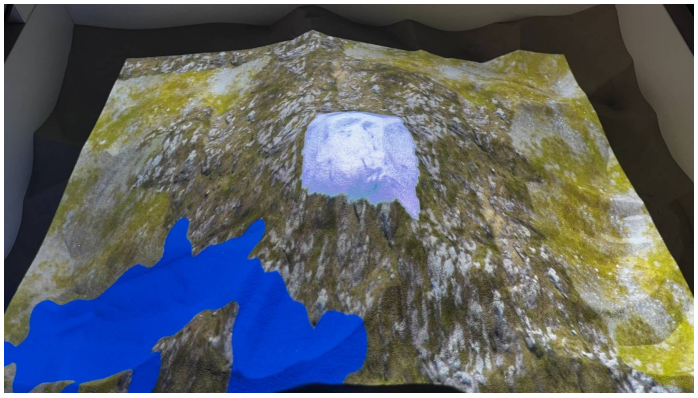
V druhom kole testovania užívatelia už dostali špecifické informácie, o tom, čo sa ako ovláda a funguje. Pri vývoji som stanovil, že sa pod rukou vytvára väčší počet častíc vody, aby som bol schopný rýchlo priebežne testovať aplikáciu. V praxi sa však ukázalo, že užívatelia preferuju postupne simulovať menšie množstvo vody. Kvôli tomuto som znížil počet generovaných častíc. Taktiež užívatelia spomenuli, že pri tvarovaní terénu sa v textúre na chvíľu objavila ich ruka. Tento problém som opravil zmenou nastavení výšok citlivé zóny pre detekciu polohy pre nastavenie zdroja vody.

Celkovo užívatelia najviac ocenili exotický dizajn (obrázok 4.12). Tento koncept ich veľmi zaujal a venovali mu veľa času, pri tvorení súostrovia.



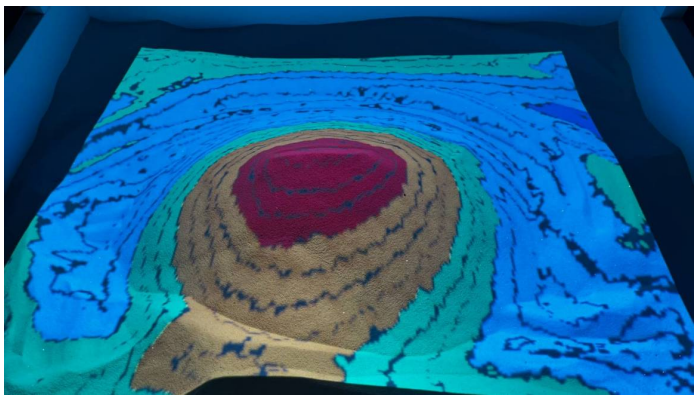
Obrázek 4.12: Exotický dizajn.

Najkritickejši hodnotili reálny dizajn (obrázok 4.13), kvôli jeho zastaranému vzhľadu vyplývajúcemu z použitých textúr.



Obrázek 4.13: Reálny dizajn s vodou.

Posledným dizajnom je farebný s vrstevnicami (obrázok 4.14). Tento užívatelia hodnotili pozitívne. Ocenili možnosť vizualizácie bodov s rovnakou výškou formou vrstevnice.



Obrázek 4.14: Farebný dizajn s vrstevnicami.

Kapitola 5

Záver

V rámci tejto práce bola narvhnutá a implementovaná aplikácia, ktorá rožširuje klasickú funkcionalitu AR SandBoxu o simuláciu toku tekutín na základe dynamicky vyformovaného terénu. Cieľom bolo navrhnúť komplexný systém pre zariadenie AR SandBox, ktorý umožní na základe tvaru terénu modelovaného pomocou piesku, nastavovať zdroje tekutín a v simulácií zobraziť tok týchto tekutín terénom. Tento cieľ som z väčšej časti úspešne naimplementoval ako desktopovú aplikáciu, ktorá na základe hĺbkových dát zobrazí adekvátnu textúru a dovolí simuláciu tekutín. Vzhľadom na to, že táto práca pokrýva množstvo veľkých tematických celkov, ktorým som sa musel venovať, som nebol schopný vytvoriť model na rozpoznávanie, ktorý by som mohol použiť vo finálnom riešení. Myslím si, že týmto moja práca poukazuje na náročnosť tohto problému a vytvára priestor pre zlepšenie a inováciu v tejto oblasti.

Pred splnením zadania bolo najprv dôležité pochopiť teoretické základy z oblastí, ktoré táto práca pokrýva. Na začiatku boli spomenuté už existujúce podobné riešenia. Ďalej som sa venoval predstaveniu jednoduchých metód pre simuláciu tekutín, ktoré sa dajú vypočítať a zobraziť v reálnom čase. Okrem toho som popísal postup pri vytváraní softvéru slúžiaci na rozpoznávanie gest rúk a problémy, s ktorými je nutné rátať pre prekonanie tejto požiadavky.

Aplikácia si prešla veľkým počtom zmien návrhu, logiky a aj použitých technológii. Keďže som s takýmito nástrojmi nemal dostatočne veĺa skúseností, bol som nútený výskúšať rôzne metódy a princípy, aby som dosiahol, čo najelpší výsledok.

Z mojích skúseností získaných pri vývoji tejto aplikácie by sa dalo pokračovať viacerými smermi. Prvá možnosť je vytvorenie väčšieho a diverzifikovanejšieho dátoveho setu a natrénovanie modelu z čisto hĺbkových dát. Týmto by sa znížila náročnosť na zdroje. Ďalej by sa dala simulácia zefektívniť, keby sa pohyb častíc vypočítaval pomocou *Compute shader*, ktorý by efektívne využil grafickú kartu pre výpočet velocít častíc a iných potrebných vlastností. Posledný priestor pre inováciu je pri textúrovaní terénu. Toto je v *Unity* prekvapivo náročný proces a aplikáciu na veľmi krátky čas zasekne, čo je trochu vidno na simulovanej tekutine.

Výsledné riešenie sa nachádza na Fakulte informačných technológií VUT v Brne vo *FIT Creative Showroom & Open Space*. Je dostupná pre všetky vekové kategórie a vyzdvihuje kreatívnu atmosféru priestoru. Dúfam, že moje riešenie inšpiruje aj iných vývojárov v tejto oblasti.

Literatura

[1] ZUBRIK, T. *3D výuková aplikace s využitím hloubkových senzorů*. Brno, 2019. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Martin Veľas. Dostupné z: https://www.vut.cz/www_base/zav_prace_soubor_verejne.php?file_id=197214.

[2] PARMAR, D. *Human computer interaction*. Kyambogo University, 2023. Dostupné z: https://www.academia.edu/12108034/Human_Computer_Interaction_HUMAN_COMPUTER_INTERACTION.

[3] ELKOURA, G. a SINGH, K. *Handrix: Animating the Human Hand, Department of Computer Science*. University of Toronto, 2003. Dostupné z: https://www.dgp.toronto.edu/~gelkoura/noback/scapaper03.pdf.

[4] NILAKSHI, J. *UI Design : Key to Captivate User Understanding*. September 2023. Dostupné z: https://www.researchgate.net/publication/374155710_UI_Design_Key_to_Captivate_User_Understanding.

[5] BHASKARAN, K. A., NAIR G. A. a RAM, K. D. *Smart gloves for hand gesture recognition: Sign language to speech conversion system*. December 2016. Dostupné z: https://ieeexplore.ieee.org/document/7931887.

[6] LIU, X. a FUJIMURA, K. *Hand gesture recognition using depth data*, Ohio State University. 2009. Dostupné z: https://ieeexplore.ieee.org/document/1301587.

[7] GARG, P., AGGARWAL, N. a SOFAT, S. *Vision Based Hand Gesture Recognition*, World Academy of Science, Engineering and Technology. 2009. Dostupné z: https://www.academia.edu/65711329/Vision_Based_Hand_Gesture_Recognition?from_sitemaps=true&version=2.

[8] SAID, S., KALMS, L., GOHRINGER, D. a GHANY, M. A. E. Nemecká univerzita v Káhire a Technická univerzita v Drážďanoch. 13 Október 2019. Dostupné z: https://www.researchgate.net/publication/338796081_HardwareSoftware-Codesign_for_Hand_Gestures_Recognition_using_a_Convolutional_Neural_Network.

[9] NAWAZ, B., VERMA, I., SHARMA, S. a GUPTA, M. Hand *Gesture Recognition*. Máj 2022. Dostupné z: https://www.academia.edu/82092586/Hand_Gesture_Recognition.

[10] Howard H. Hu, *Fluid Mechanics (Fifth Edition)*. 2012. s. 421-472. ISBN 978-0-12-382100-3. Dostupné z: https://www.sciencedirect.com/science/article/abs/pii/B9780123821003100101.

[11] Medvecký-Heretik, J. *Real-time Water Simulation in Game Environment*. Brno, 2018. Diplomová práce. Masarykova univerzita, Fakulta informatiky. Vedúci práce : Mgr. Jiří Chmelík, Ph.D. Dostupné z: https://is.muni.cz/th/mfkg2/master-thesis.pdf.

[12] LIM, H. A. *Mathematical and Computer Modelling*. 1990. s. 720-727. Dostupné z: https://www.sciencedirect.com/science/article/pii/089571779090276S. -

[13] LIND, S. J., ROGERS, B. D. a STANSBY, P. K. *Review of smoothed particle hydrodynamics*. September 2020. Dostupné z: https://royalsocietypublishing.org/doi/10.1098/rspa.2019.0801.

[14] BRADSKI, G. a KAEHLER, A. *Learning OpenCV*. s. 378 - 381. ISBN 9780596554040. Dostupné z: https://www.bogotobogo.com/cplusplus/files/OReilly%20Learning%20OpenCV.pdf.

[15] MÜLLER, M., CHARRYPAR, D. a GROSS, M. *Particle-Based Fluid Simulation for Interactive Applications*. Federálny technologický inštitút v Zürichu, Katedra informatiky. 2003. Dostupné z: https://matthias-research.github.io/pages/publications/sca03.pdf.

[16] NHU-TAI, D., SOO-HYUNG, K., HUYNG-JEONG, Y. a GUEE-SANG, L. *Robust Hand Shape Features for Dynamic Hand Gesture Recognition Using Multi-Level Feature LSTM*. Chonnamská národná univerzita, Katedra konvergencie umelej inteligencie. 2020. Dostupné z: https://www.mdpi.com/2076-3417/10/18/6293.

[17] KURILLO, G., HEMINGWAY, E.,CHENG M. a CHENG, L. *Evaluating the Accuracy of the Azure Kinect and Kinect v2*. Marec 2022. Dostupné z: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9002889/.

Příloha A

Obsah priloženého pamäťového média

Priložené pamäťové médium obsahuje nasledujúcu adresárovú štruktúru:

```
/
├── src/ ..... Zdrojové súbory aplikácie
├── app/ ..... Spustiteľná aplikácia
├── latex/ ..... Zdrojové súbory tohoto dokumentu vo formáte LaTeX
└── bp.pdf ..... Tento dokument
```