Technická univerzita v Košiciach
Fakulta elektrotechniky a informatiky

Terapia pomocou technológií virtuálnej reality

Diplomová práca

2021 **Sára Javorková**

Technická univerzita v Košiciach
Fakulta elektrotechniky a informatiky

Terapia pomocou technológií virtuálnej reality

Diplomová práca

Študijný program:	Informatika
Študijný odbor:	9.2.1. Informatika
Školiace pracovisko:	Katedra počítačov a informatiky (KPI)
Školiteľ:	doc. Ing. Branislav Sobota PhD.
Konzultant:	

Košice 2021 **Sára Javorková**

Názov práce:	Terapia pomocou technológií virtuálnej reality
Pracovisko:	Katedra počítačov a informatiky, Technická univerzita v Košiciach
Autor:	Sára Javorková
Školiteľ:	doc. Ing. Branislav Sobota PhD.
Konzultant:	
Dátum:	23. 4. 2021
Kľúčové slová:	A-frame, kolaboratívna virtuálna realita, rehabilitácia hornej končatiny

Abstract: Cieľom práce je vytvoriť systém pre rehabilitáciu hornej končatiny s previazaním na kognitívne poruchy v prostredí kolaboratívnej virtuálnej reality. Riadime sa požiadavkami terapeutov, ktorý so systémom experimentovali v spolupráci s človekom po mozgovej príhode. Hlavná podstata systému spočíva vo vytvorení animácií, ktoré nahrádzajú reálny pohyb ruky pri bežných činnostiach, ako je napríklad úchop pohára. Navrhujeme pokračovať v testovaní a implementovať viac animácií podľa požiadaviek terapeutov.

Thesis title:	Therapy with virtual reality technology
Department:	Department of Computers and Informatics, Technical University of Košice
Author:	Sára Javorková
Supervisor:	doc. Ing. Branislav Sobota PhD.
Tutor:	
Date:	23. 4. 2021
Keywords:	A-frame, collaborative virtual reality, upper limb rehabilitation

Abstract: The aim of the diploma thesis is to create a system for rehabilitation of upper limb with the connection to cognitive disorders in the environment of collaborative virtual reality. We follow the requirements of the therapists, who were experimenting with system in cooperation with person after a stroke. The main gist of the system is to create animations, which can replace the real move of the arm during common activities, such as to hold a cup. We suggest proceeding with testing and implementing more animations according to the requirements of the therapists.

60722

TECHNICKÁ UNIVERZITA V KOŠICIACH

FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Katedra počítačov a informatiky

Z A D A N I E
D I P L O M O V E J P R Á C E

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

Názov práce:

Terapia pomocou technológií virtuálnej reality

Virtual reality technologies based therapy

Študent: **Bc. Sára Javorková**
Školiteľ: **doc. Ing. Branislav Sobota, PhD.**
Školiace pracovisko: **Katedra počítačov a informatiky**
Konzultant práce:
Pracovisko konzultanta:

Pokyny na vypracovanie diplomovej práce:

1.Naštudovať problematiku virtuálnej reality, kolaboratívnej virtuálnej reality a jej technológií a základnú problematiku rehabilitačných procesov.
2.Analyzovať problematiku systémov kolaboratívnej virtuálnej reality v prostredí rehabilitácií rôzneho typu s dôrazom na horné končatiny s previazaním na možné kognitívne poruchy.
3.Navrhnúť terapeutický systém na báze kolaboratívnej virtuálnej reality a jej technológií vrátane časti systému terapeuta a časti systému pacienta, u ktorého sa terapia/rehabilitácia vykonáva.
4.Pri návrhu sa sústrediť na základe pokynov vedúceho práce najmä na podporu koordinácie pohybu horných končatín, podporu precizácie, výberu a umiestnenia vrátane scenárov terapeutických pohybov.
5.Na základe bodov 2-4 implementovať programové vybavenie na báze technológií kolaboratívnej virtuálnej reality.
6.Vykonať experimentálne práce s implementovaným systémom vrátane spracovania alebo dotvorenia potrebných modelov resp. funkcií a zhodnotiť dosiahnuté výsledky.
7.Vypracovať dokumentáciu podľa pokynov vedúceho práce.

Jazyk, v ktorom sa práca vypracuje: slovenský
Termín pre odovzdanie práce: 23.04.2021
Dátum zadania diplomovej práce: 30.10.2020

.....
prof. Ing. Liberios Vokorokos, PhD.
dekan fakulty

Čestné vyhlásenie

Vyhlasujem, že som záverečnú prácu vypracovala samostatne s použitím uvedenej odbornej literatúry.

Košice, 23.4.2021 .....

Vlastnoručný podpis

Poďakovanie

Chcela by som sa poďakovať vedúcemu mojej práce doc. Ing. Branislavovi Sobotovi, PhD. za všetky jeho rady a pripomienky. Tiež by som sa rada poďakovala aj svojej rodine a priateľom za podporu.

Obsah

Zoznam obrázkov

Úvod

Táto práca je zameraná na rehabilitáciu kognitívnych porúch horných končatín za pomoci virtuálnej reality. V prvom rade ma na práci zaujala spolupráca s tímom terapeutov zo Slovenskej akadémie vied a testovanie na reálnych subjektoch, ktorým verím, že táto práca pomôže ku lepšiemu životnému štýlu. Zaujala ma predstava pomôcť ľudom, ktorých obyčajná rehabilitácia nudila a nemotivovala. Chcela som trochu oživiť tento proces modernou technológiou, ktorá sa neustále posúva vpred.

V prvom rade bolo potrebné si naštudovať ako fungujú kolaboratívne virtuálne prostredia a ako fungujú v spojení s rehabilitáciami. Ďalej určiť na čom sa bude stavať výsledný systém a s konzultáciami tímu terapeutov systém implementovať a odskúšať na subjektoch s kognitívnymi poruchami.

Hlavným problémom, ktorý riešime v práci, je implementovať program, ktorý bude súčasťou väčšieho systému rehabilitačného procesu. Tento systém zahŕňa aj zariadenie na vytváranie elektroencefalogramu a zariadenie na elektrickú stimuláciu svalov. Nami vytvorená časť projektu sa bude spúšťať pomocou prilby Oculus. Program má byť kompatibilný s elektrickou stimuláciou svalov, aby animácia prebiehala spolu s ňou.

Keďže je aktuálny rehabilitačný proces spojený aj s inou technológiou, ktorá posiela impulzy do ruky pacienta, nie je možné urobiť plnohodnotnú rehabilitáciu na diaľku. No pre rehabilitácie v budúcnosti, ktoré budú môcť byť vykonávané aj na diaľku, bude systém implementovaný na báze kolaboratívneho prostredia. Terapeut sa teda bude vedieť pripojiť do prostredia, kde sa nachádza pacient a sledovať priebeh rehabilitácie a komunikovať s pacientom. Terapeut bude mať navyše od pacienta v prostredí niekoľko tlačidiel, ktorými vie ovládať základné funkcie sekvencie, ako spustiť sekvenciu, zastaviť ju a prepnúť aktuálnu animáciu.

Formulácia úlohy

Hlavným cieľ práce bolo navrhnúť a implementovať rehabilitačný proces ako sekvenciu animácií hornej končatiny. Sekvencia má obsahovať niekoľko animácií, ktoré sa budú obmieňať. Napríklad pri pohybe úchopu pohára sa najprv pohár objaví v priestore a pacient má za úlohu si daný pohyb predstaviť v hlave. Animácia sa nespustí hneď, no čaká na podnet od terapeuta, ktorý skúma jeho mozgovú činnosť, či došlo ku zmene. Keď systém dostane správu od terapeuta o zmene jeho oscilačných rytmov, spustí animáciu. Kým animácia beží, terapeut replikuje pacientove signály do nervových zakončení ruky pomocou prístroja na elektrickú stimuláciu svalov zapojenom na pacientovom predlaktí.

Ďalším cieľom bolo vytvoriť rozhranie, kde sa môžu nastaviť parametre rehabilitácie. Hlavným parametrom je zadanie počtu animácií v sekvencii. Ďalej je možné nastaviť niekoľko časových intervalov v sekundách. Jeden z nich je hodnota doby, kedy program čaká na správu od terapeuta na spustenie animácie. Program prejde do stavu relaxu ak prejde daná doba a program správu nedostane. Alebo ak obdrží správu, spustí animáciu a po jej ukončení program prejde do relaxačného stavu. Doba trvania tohto stavu je ďalší parameter, ktorý bude možné nastaviť. Počas tohto stavu je v priestore zobrazená len ruka, bez ďalšieho objektu a čaká sa na koniec času relaxu. Po tejto dobe sa objaví cieľový objekt ďalšej animácie. Sekvencia skončí po uskutočnení počtu animácií, ktorý bol daný, bez ohľadu, či sa animácia vykonala alebo len čakala na správu. Samozrejme počas celej sekvencie má terapeut možnosť sekvenciu úplne vypnúť alebo len prejsť na ďalšiu animáciu.

1 Virtuálna realita

Vývoj nových technológií, konkrétne výpočtovej techniky, graduje každým rokom a v dnešnej dobe aj efektívna interakcia medzi počítačom a človekom zažíva obrovský pokrok, čo sa týka technológií. Jednou z týchto technológií je aj virtuálna realita (skr. VR), predstavujúca „prostredie modelované prostriedkami počítača simulujúce kvázi skutočnosť. Primárne sa ním chápe vytváranie vizuálneho zážitku zobrazovaného na obrazovke počítača, prípadne cez špeciálne zobrazovacie zariadenia.“ [23] Myseľ človeka dokáže vnímať tri rozmery, tzv. dimenzie a rozoznáva tvary a objekty pohybujúce sa v prostredí, ktoré majú vzájomné vzťahy alebo sa ovplyvňujú.

Virtuálna realita modeluje digitálnu podobu simulovaného reálneho sveta, resp. prostredia priamo v počítači. Cieľom je vytvorenie primárne vizuálneho zážitku, ktorý môže byť zdieľaný cez obrazovku počítača, prípadne cez špeciálne okuliare. Avšak základom virtuálnej reality je poskytnúť používateľovi čo največernejšie zobrazenie priestorových modelov a scén, pričom je možné s nimi manipulovať a pohybovať sa v trojrozmernom priestore, a to v reálnom čase. Taktiež základom je aj vytvorenie reálneho sveta so všetkými jeho zákonitosťami a pravidlami. V zložitejších prípadoch sa používateľ vnára do virtuálneho sveta zapojením aj ostatných zmyslov človeka ako sluch, čuch alebo aj hmat. [24] Existuje viacero rozdelení virtuálnej reality, resp. systémov VR. Systémy VR sa kategorizujú na základe aplikovaných technických prostriedkov, na základe dynamiky pozorovateľa a prostredia alebo na 4 základné typy, a to systémy VR pre osobné počítače, imerzívne systémy, rozširujúce realitu a projekčné systémy.

1.1 Stupne virtuálnej reality

Virtuálna realita vytvára prostredie, ktoré môže byť sprostredkované rôznymi zariadeniami a s ktorými je možné interagovať rôznymi spôsobmi. Na základe týchto faktov sa virtuálna realita delí na tri základné stupne.

Prvým stupňom virtuálnej reality je pasívna aplikácia, ktorú si môžeme predstaviť ako film. Prostredie v tomto stupni je možné vidieť a počuť, pričom nie je možno svet modifikovať a vplývať naň svojou činnosťou. Druhý stupeň predstavuje aktívna aplikácia, kde je možné virtuálne prostredie ľubovoľne skúmať, to znamená voľný pohyb v celom prostredí. V tomto stupni taktiež nie je možné modifikovať prostredie virtuálnej reality a z voľného pohybu vyplýva, že používateľ nemá žiadnu interakciu s objektami virtuálneho sveta. To znamená, že sa v celom prostredí pohybuje ako duch a môže prechádzať všetkými objektami v prostredí VR.

Najvyšším stupňom je interaktívna aplikácia predstavujúca najdokonalejšiu a zároveň najnáročnejšiu aplikáciu. Virtuálne prostredie je možné modifikovať a medzi používateľom a objektmi sveta existuje interakcia. Používateľ dokáže premiestňovať objekty, spúšťať funkcie virtuálnych objektov, stláčať rôzne tlačidlá a pod.[24]

1.2 Interakcia s virtuálnou realitou

Vnorenie používateľa do prostredia virtuálnej reality je umocnené interakciou s prostredím použitím špecifických zariadení prenášajúcich obraz, zvuk a iné zmyslové vnemy k používateľovi. Zložitejšie zariadenia dokážu zaznamenať pohyb a reč tela používateľa prostredníctvom rôznych senzorov a odoslať programu dáta, ktoré sa premietnu do virtuálneho sveta. Medzi takéto zariadenia patria napríklad rôzne 2D a 3D ovládače, dátové rukavice, dátové helmy, prípadne celé obleky.

Medzi 2D ovládače patria základné vstupné technické vybavenia počítača ako napríklad počítačová myš, klávesnica alebo pákový ovládač (joystick). Ovládače s viacerými stupňami voľnosti (3D ovládače) sú napríklad spacepilot (Obr. 1.1), spaceball alebo 3D joystick. [21] Tieto ovládače spolu s herným volantom poskytujú najreálnejšiu interakciu s prostredím virtuálneho sveta v rámci ovládačov.

Dátové rukavice sú vstupným zariadením a slúžia zaznamenávanie pohybu

Obr. 1.1: 3D ovládač Spacepilot [7]

rúk a prstov. Na základe pohybu ruky používateľa sa tento pohyb premietne do virtuálneho sveta. Najnovšie dátové rukavice už nie sú len vstupnými zariadeniami, ale dokážu používateľovi poskytnúť hmatový zážitok. Projekt študentov z univerzity Rice v Houstone vyvíja dátovú rukavicu, ktorá by mala hráčovi umožniť úchop virtuálneho objekty a pomocou vzduchom plnených vankúšikov v rukavici navodiť reálny pocit veľkosti a tvaru uchopeného objektu. [22] Na báze dátových rukavíc sú založené aj kompletné dátové obleky, ktoré sledujú kompletný pohyb používateľa, a taktiež dokážu poskytnúť spätnú väzbu prostredníctvom nafukovacích vankúšikov alebo vibrácií obleku.

Displeje pre virtuálnu realitu môžu byť jednoduchšie, ako napríklad monitor počítača, televízia alebo najnovšie mobilný telefón. Miera ponorenia používateľa do VR je v takomto prípade značne obmedzená, preto boli vyvinuté špeciálne displeje ako súčasť špeciálnej dátovej prilby (z ang. Head mounted displays - HMD), prípadne displeje pre veľké projekčné VR systémy označované ako CAVE.

Dátové helmy, resp. prilby, ponúkajú hlbšie a presvedčivejšie vnorenie sa do virtuálneho sveta. Dátová helma predstavuje mobilnú zobrazovaciu jednotku, ktorá sa pohybuje spoločne s používateľom. [21] Používateľ má nasadenú helmu, ktorú zakrýva celé zorné pole používateľa. Zobrazenie virtuálnej reality prebieha cez stereoskopické displeje, ktoré môžu byť dva samostatné pre každé oko alebo jeden rozdelený na dve polovice, pričom obrazy sú premietané na každé oko

zvlášť a pod odlišným uhlom.

Existujú samostatné dátové helmy, ktoré nevyžadujú ďalší hardvér a helmy na báze prepojenia s počítačom. Sledovanie pohybu, spracovanie údajov o sledovaní pohybu a vykreslenie sa vykonáva na jednotke náhlavnej súpravy (z ang. headset) VR. Samostatné dátové helmy sú v porovnaní s helmami spojenými s hardvérom počítača značne limitované v rámci kvality vykreslenia. Na druhej strane, helmy prepojené s počítačom sú závislé od kvality hardvéru. [17] Väčšinou sú kombinované s rôznymi ovládačmi, dátovými rukavicami alebo s celým dátovým oblekom.

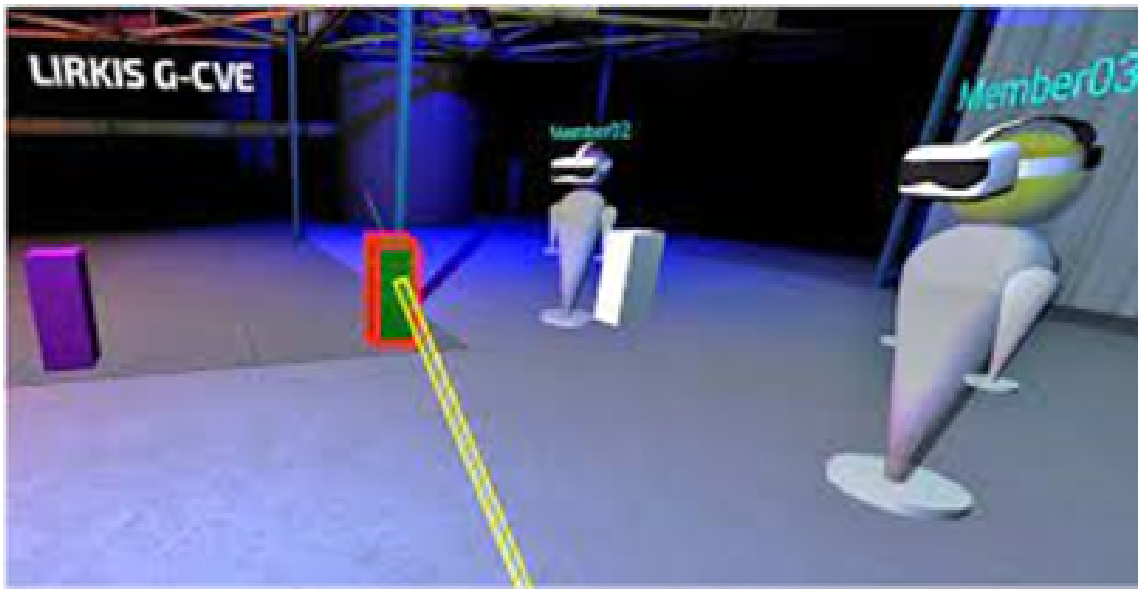
Obr. 1.2: Dátová prilba Oculus [17]

2 Kolaboratívna virtuálna realita

V dnešnej dobe existuje viacero projektov spojených s virtuálnou realitou, zameraných na rôzne oblasti, ako napríklad rôzne hry, výskumy alebo edukácie. V princípe je virtuálny svet zameraný na jedného používateľa, ktorý dokáže interagovať s prostredím, a to aj na veľkú vzdialenosť.

Zdieľanie informácií je ústredným bodom spolupráce. Princíp kolaboratívneho virtuálneho prostredia je spolupráca viacerých používateľov v tom istom čase v rámci jedného virtuálneho prostredia. Kolaboratívne virtuálne prostredie (z ang. Collaborative virtual environments – CVE) predstavuje distribuovaný systém virtuálnej reality, ktorý ponúka grafické potenciálne nekonečné digitálne krajiny. V rámci prostredia môžu jednotlivci zdieľať informácie prostredníctvom vzájomnej interakcie, a taktiež prostredníctvom individuálnej alebo spoločnej interakcie s reprezentáciami dát. [1]

Väčšina virtuálnych prostredí podporuje reprezentáciu používateľov generáciou modelov označovaných ako avatar (obr. 2.1). Vzájomná interakcia medzi používateľmi môže byť formou vizualizácie, prípadne audio komunikácie. Interakcia používateľov so samotným systémom môže byť realizovaná rôznymi prostriedkami, ako napr. ovládanie myšou, rečovými príkazmi alebo zadávaním príkazov – textom. [1]



Obr. 2.1: Prostredie Lirkis G-CVE [8]

Pri zdieľaní virtuálneho prostredia je potrebné, aby všetky zariadenia boli kompatibilné a všetci používatelia mohli byť súčasťou zdieľaného virtuálneho prostredia, bez ohľadu na geografickú vzdialenosť. Rýchly technologický vývoj spôsobuje nedostatky kolaboratívnych virtuálnych systémov, konkrétne optimalizácia 3D grafiky a vstupov pri multiplatformových VR systémoch. Riešením je koncept, ktorý umožňuje dynamicky rozpoznávať zariadenia a je založený na webových technológiách – webové kolaboratívne virtuálne prostredie. [8]

Nasadenie webových kolaboratívnych VR prostredí môže poskytnúť globálnu spoluprácu bez obmedzenia geografického umiestnenia používateľov. Webová platforma môže zjednodušiť prístup používateľov a skrátiť tak čas na vývoj. [8]

2.1 Existujúce systémy kolaboratívnej VR

Vývoj kolaboratívnych systémov pre virtuálnu realitu sa v poslednej dobe rozrástol. Dôkazom sú rôzne existujúce systémy s rôznorodým využitím. Na Slovensku vznikli dva kolaboratívne systémy, a to LIRKIS G-CVE a systém Coven. Vo svete sú známejšie kolaboratívne systémy napr. Hubds.

2.1.1 LIRKIS G-CVE

Webový kolaboratívny VR systém bol vytvorený aj na Technickej univerzite v Košiciach v rámci laboratória počítačovej grafiky LIRKIS (Obr. 2.2). Akronymom slova LIRKIS je Laboratórium Inteligentných Rozhraní Komunikačných a Informačných Systémov a predstavuje pracovisko určené na výskum, vývoj a výučbu v oblasti aplikácií paralelných, distribuovaných a sieťových počítačových systémov. [5] Virtuálno-realitnú webovú kolaboráciu viacerých používateľov v reálnom čase, LIRKIS G-CVE (akronym LIRKIS Global - Collaborative Virtual Environment), založil Ing. M. Hudák.

Webové kolaboratívne prostredie môže byť využité na rôzne účely. Podľa prezentačného videa projektu je vidieť, že LIRKIS G-CVE môže byť využité pri výučbe v oblasti strojárstva, stavebníctva, automobilovej výroby, či medicíny .

Obr. 2.2: Laboratórium LIRKIS na Technickej univerzite v Košiciach [4]

2.1.2 Coven

Kolaboratívne virtuálne prostredie pod názvom Coven vzniklo na Fakulte informatiky a informačných technológií (FIIT STU) v spolupráci so slovenskou organizáciou Moving Medical Media. Prototyp projektu sa využíva v rámci spomínanej fakulty na vzdialenú výučbu základných princípov procedurálneho programovania. Pomocou kolaboratívneho prostredia sa môžu aj pozorovatelia prezentácie zapojiť do priebehu a interagovať s prostredím, skúšať rôzne reakcie, prezerať 3D súčiastky, ktoré by pri obyčajných prezentáciách videli len ako obrázok. V budúcnosti je možné využitie v rôznych odvetviach, napríklad v školstve, priemysle či medicíne. Príkladom môže byť tréning zamestnancov rizikových odvetví, ako napríklad vojakov, pracovníkov elektrárne alebo môže slúžiť na simuláciu medicínskych operácií. [18]

Názov projektu COVEN pochádza z anglického slovného spojenia „Collaborative Virtual Environments“, čo v preklade znamená kolabiratívne virtuálne prostredia. Samotný názov COVEN je taktiež akronymom slova v angličtine, ktorého „význam sa v rôznych prameňoch líši, avšak jeden zo zaužívaných významov je zhromaždenie čarodejníc za účelom spoločného vykonávania rôznych rituálov. Preto majú napríklad niektorí avatari v ich virtuálnom prostredí čarodejnícke klobúky.“ [18] Zaujímavosťou je aj základ slova, keďže latinské slovo „coventum“ znamená stretnutie.

Obr. 2.3: Avatar v projekte COVEN [18]

2.1.3 Hubs

Hubs je platforma pre virtuálnu spoluprácu v prostredí webového prehliadača od spoločnosti Mozilla. Umožňuje vytvárať vlastné 3D priestory a pozvať ostatných používateľov k pripojeniu do prostredia. Projekt Hubs je určený pre bežných ľudí, ktorí sa chcú stretnúť na diaľku. Predstavuje spôsob, akým sa môžu ľudia stretnúť spoločne vo virtuálnom prostredí. Môže mať aj rôzne iné využitia, napríklad používateľ môže založiť konferenciu alebo vyučovať, nakoľko umožňuje aj zdieľanie obrázkov, PDF súborov, odkazov alebo rôznych 3D modelov. Hubs funguje na rôznych platformách, takže pripojenie do prostredia je umožnené prostredníctvom počítača, ale rovnako aj prostredníctvom dátovej prilby. [10]

Obr. 2.4: Virtuálne prostredie platformy Hubs [10]

3 Virtuálne prostredie v prehliadači

Virtuálna realita (VR) znamená, že ak ste v takomto prostredí, nachádzate sa v úplne odlišnej realite, v akej reálne ste. Narozdiel od toho, rozšírená realita (AR – Augmented reality) je vylepšená verzia reality, ktorá dopĺňa reálny svet o digitálne informácie. Príkladom môže byť známa aplikácia Pokémon Go. Termín XR (Extended reality) označuje ktorýkoľvek z týchto typov rozšírenej reality. [20] Prepojenie medzi aplikáciami sprostredkúva API (z ang. Application Programming Interface) a zabezpečuje komunikáciu medzi aplikáciami, resp. technológiami.

Na interakciu rôznych zariadení virtuálnej reality (napr. HMD helmy) s webovým prehliadačom bolo vytvorené WebVR API, ktoré podporuje prepojenie pohybu z displeja zariadenia do 3D scény virtuálneho prostredia. Pomocou metódy dokáže rozpoznať všetky zariadenia VR pripojené k počítaču a vytvorí z nich objekt obsahujúci informácie o samotnom zariadení, spojené ovládače s displejom a pod. WebVR API nikdy nebolo webovým štandardom, podporovalo ho málo webových prehliadačov a preto bolo nahradené WebXR API rozhraním.

WebXR predstavuje skupinu štandardov, ktoré sa používajú pre podporu vykresľovania 3D scén na hardvér určený na znázornenie virtuálnej scény alebo na pridávanie grafických prvkov do reálneho sveta, čo znamená vytvorenie rozšírenej reality. Zariadenia, ktoré plne podporujú štandard WebXR úplne pohlcujú používateľa do virtuálneho sveta. Sú to napríklad náhlavné súpravy (spomínané dátové helmy). Zariadenia na vykreslenie rozšírenej reality môžu byť aj mobilné telefóny, ktoré snímajú reálny svet fotoaparátom a scéna je rozšírená pomocou počítača. [15]

V roku 2007 spoločnosť Mozilla prvýkrát predstavila Canvas 3D, ktorý umožňoval vykreslenie interaktívnej 3D grafiky na webe. Ďalším krokom bolo vytvorenie API a do roku 2009 skupina Khronos založila pracovnú skupinu WebGL. Prvá

verzia špecifikácie bola vydaná v roku 2011. [20]

3.1 Rozhranie WebGL

WebGL (z ang. Web Graphics Library) je JavaScript API rozhranie založené na OpenGL ES 2.0, ktoré poskytuje vykreslenie vysoko výkonnej interaktívnej 3D a 2D grafiky v ľubovoľnom kompatibilnom webovom prehliadači do HTML5 prvku canvas. Zjednodušene povedané, umožňuje vykreslenie 3D scény vo webovom prehliadači bez potreby použitia doplnkov prehliadača. Výhodou je vysoká podpora najpoužívanejších webových prehliadačov. Pripravuje sa aj nadstavba s označením WebGL 2 API, ktoré predstavuje podporu pre veľkú časť funkcií OpenGL ES 3.0.[14]

Tvorba VR sveta prostredníctvom WebGL môže byť zdĺhavá a náročná, preto vznikli knižnice a rámce, ktoré uľahčujú vývoj VR projektu.

3.2 Knižnice pre virtuálnu realitu

Používanie zariadení VR na interakciu s webovým prehliadačom poskytuje skvelý zážitok používateľom webovej aplikácie. Pri tvorbe zaujímavého VR dizajnu webovej aplikácie vývojári používajú rôzne knižnice a webové rámce pre kvalitnejšiu, efektívnejšiu a rýchlejšiu tvorbu VR aplikácií. Populárne WebGL knižnice sú napríklad Three.js, Babylong.js, Argon.js, React VR alebo webové rámce PlayCanvas a A-Frame. V rámci diplomovej práce sme použili webový rámec A-Frame, ktorý je vytvorený ako nadstavba knižnice Three.js a vývojové prostredie Glitch.

3.2.1 Webový rámec A-Frame

Webový rámec vytvorený tímom Mozilla VR v roku 2015 slúži hlavne na vytváranie projektov s virtuálnou, ale aj rozšírenou realitou použitím HTML jazyka a bez väčšej znalosti WebGL. Práve vďaka založeniu na jazyku HTML je prístupný a ľahko použiteľný. Podporuje komunikáciu s mnohými VR prilbami a ovládačmi. Výhodou rámca je open-source princíp a veľká komunita vývojárov.

A-Frame je založený na architektúre komponentov entít a poskytuje deklaratívnu, zostaviteľnú a opakovane použiteľnú štruktúru entít a komponentov. A-Frame sa

používa na základe HTML jazyka, ale vývojári majú možnosť využiť aj Javacript jazyk, DOM API, knižnicu Three.js, WebVR a WebGL. [18]

Podporuje komunikáciu s mnohými VR prilbami a ovládačmi, pričom využíva WebVR API na získanie prístupu k dátam senzorov náhlavnej súpravy VR (poloha, orientácia) na transformáciu kamery a na vykreslenie obsahu priamo do zariadenia.[**browserSup**

3.2.2 Knižnica Three.js

Three.js predstavuje 3D knižnicu, ktorá uľahčuje prístup k 3D obsahu na webovej stránke. Často sa zamieňa s WebGL, ktorý je na nízkej úrovni a dokáže vykresliť iba body, čiary a trojuholníky. Knižnica naopak využíva WebGL pre vykreslenie 3D objektov do webového rozhrania. Obsahuje mnoho objektov, scén, funkcií, prácu so svetlami, tieňmi, textúrami alebo 3D matematikou. Všetky tieto implementácie by boli zložité a zdĺhavé napísať v čistom WebGL. Inštalácia knižnice nevyžaduje špeciálne aplikácie alebo príkazový riadok. Knižnicu stačí importovať prostredníctvom jedného odkazu na súbor do hlavičky HTML kódu.

Práca s Three.js sa vyznačuje virtuálnym priestorom nazývaným scéna, do ktorého je možné vkladať jednotlivé objekty. Tieto objekty sa dajú behom chodu programu vytvárať a je nimi podľa potreby možné hýbať, respektíve ich rotovať, či meniť ich veľkosť.

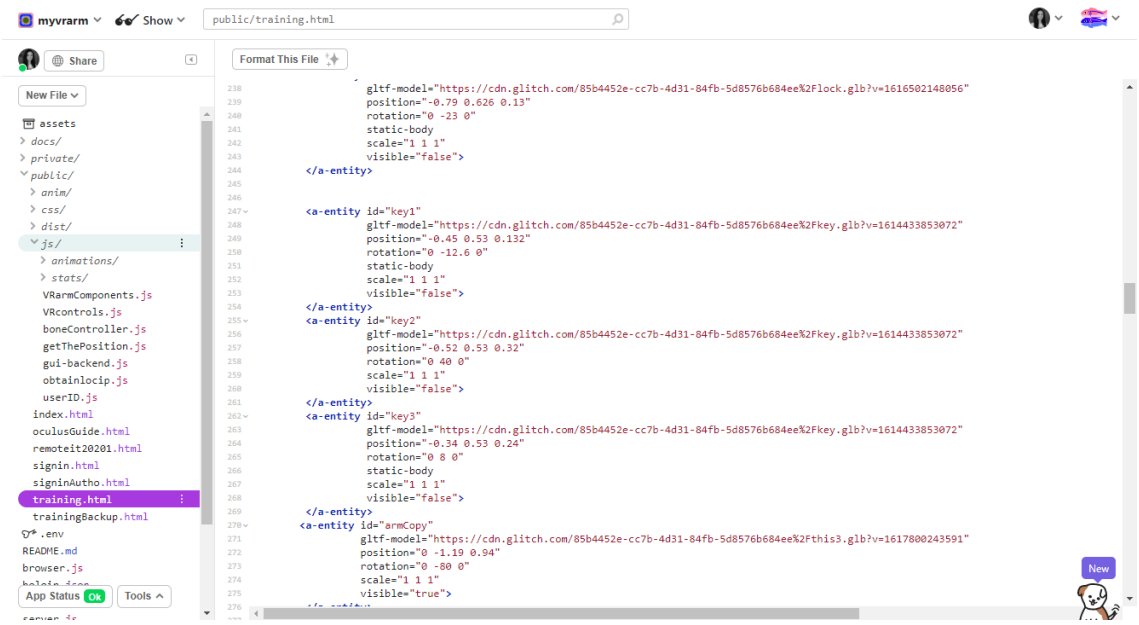
3.3 Glitch – webové vývojové prostredie

Glitch je skvelá, bezplatná platforma k vývoju webových projektov. Vývojári platformy nasadzujú vývojárske nástroje, ktoré sú vhodné pre každého a je možné ľahko vytvoriť demo verzie aplikácií alebo prototypy aplikácie založených na jazyku JavaScript. Podporuje základné JavaScriptové knižnice, ale aj zložitejšie webové rámce ako React, Angular alebo Vue. Na rozdiel od iných webových nástrojov založených na JavaScripte, Glitch podporuje aj serverové webové technológie, ako Node.js a Express.js.

Vo všeobecnosti Glitch predstavuje komunitu, kde je možné nájsť skvelé aplikácie, weby a funkčné časti projektov, ktoré je možné použiť a stavať na nich svoj vlastný projekt. Vývojári svoje projekty môžu ukázať verejnosti alebo ponechať si ich v súkromí. Predstaviť si ho môžeme ako platformu Youtube, kde sú uložené

videá a pesničky, toto funguje rovnako, ale pre aplikácie webov alebo VR projektov. Platforma je vhodná pre každého, kto chce pracovať s VR, tvoriť webové aplikácie alebo sa inšpirovať z už vytvorených projektov, ktoré sú voľne dostupné.

Najväčšou výhodou platformy je, že ponúka vstavaný textový editor (obr. 3.1), ktorý zabezpečuje automatické nasadenie projektu, kolaboráciu s ostatnými vývojármi v reálnom čase, a to všetko vo webovom prehliadači. Týmto vzniká kolaboratívne vývojové prostredie, ktoré je podobné ako služba Google Docs, kde viacerí používatelia môžu v reálnom čase upravovať ten istý dokument.



Obr. 3.1: Textový editor platformy Glitch

4 Rehabilitácia v prostredí VR

Vo všeobecnosti podľa MUDr. Tomáša Jakubíka rehabilitácia predstavuje „všetky metódy, ktoré dopomáhajú k obnoveniu a regenerácii pôvodného stavu pacienta pomocou rôznych metód a techník. Komplexná rehabilitácia je vzájomne previazaný, koordinovaný a cielený proces, ktorý má svojich špecialistov, diagnostiku a stanovené postupy.“[12] Čiže jej cieľom je zlepšiť priebeh ochorenia, zmenšiť deficit a poskytnúť čo najväčší návrat pacienta do pôvodného stavu komplexným procesom.

Častokrát po rehabilitáciách pod odborným dozorom pacient začína s domácou rehabilitáciou. Domáca rehabilitácia by mala byť pravidelná a poctivá s cieľom zvýšiť rýchlosť a kvalitu návratu pacienta do pôvodného stavu. Častokrát pacienti sú pokúšaný lenivosťou, nezáujmom a samotné rehabilitačné cviky ich nudia. Čo ak by pacienti mohli v domácej rehabilitácií, ale aj priamo pri odbornom dozore využívať virtuálnu realitu, ktorá by bola zábavnejšia a v niektorých prípadoch aj efektívnejšia ?

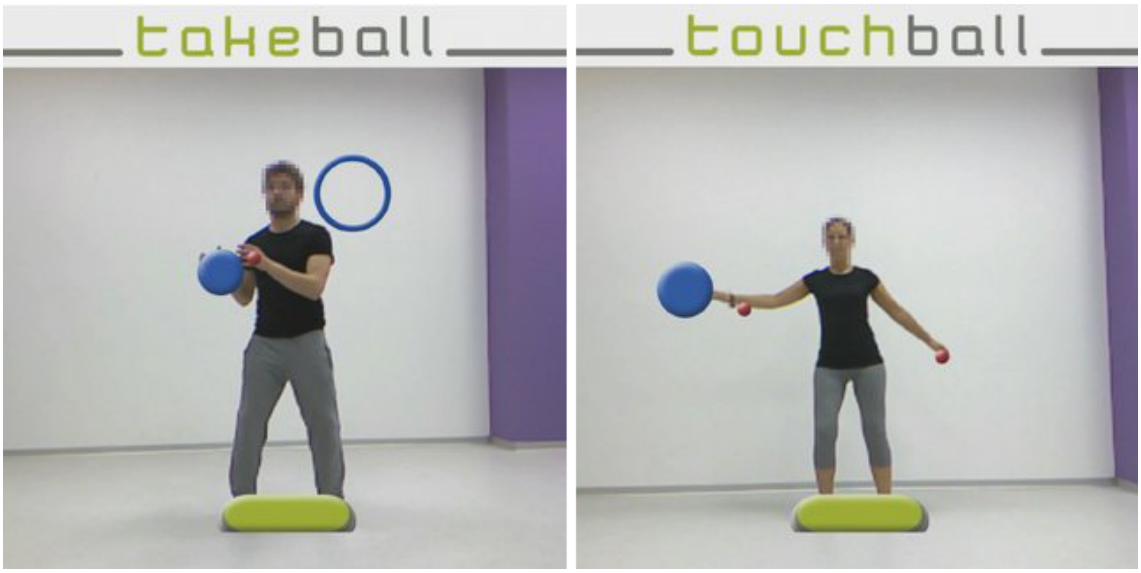
„Virtuálna realita môže pomôcť pacientom, ktorí sa spamätávajú zo závažných zranení či operácií. Podľa výskumu vedcov z Warwickskej univerzity totiž dokáže skvalitniť rehabilitáciu v domácom prostredí.“ [19] Je založená na plnení jednoduchých úloh a tréningov, vďaka ktorým sa pacient dokáže znovu naučiť rôzne pohyby, znovu získať nové schopnosti a zručnosti.

4.1 Rehabilitácia kognitívnych porúch prostredníctvom VR

Vo svete, ale mnoho aj na Slovensku, sa objavuje mnoho štúdií, ktoré sa venujú liečeniu kognitívnych porúch prostredníctvom VR. Hlavne ide o liečenie fóbií, ktoré sú ľahko napodobiteľné vo virtuálnom svete, ako napríklad strach z hmyzu alebo aj strach z rozprávania na verejnosti a podobne. V oblasti rehabilitácie končatín je ich už menej.

4.1.1 Projekt REMOVIEM

Jedna z rozšírenejších je rehabilitácia pomocou Kinect od Microsoftu s názvom REMOVIEM. Je to vytvorené pre ľudí so sklerózou multiplex, ktorý si potrebujú precvičovať pohyb celého tela. Program ponúka niekoľko cvikov, ktoré môže pacient cvičiť aj sám.



Obr. 4.1: REMOVIEM cvičenie [6]

Pacient potrebuje ku cvičeniu zariadenie Kinect a obrazovku, na ktorej sa zdieľa obraz z Kinect. Okrem toho, že pacient môže cvičiť bez dozoru fyzioterapeuta, doktor je schopný kontrolovať jeho výsledky a prispôsobovať plán programu pacientovi.[6]

4.1.2 Projekt IREX

Ďalším príkladom je projekt IREX (Interactive Rehabilitation and Exercise Systems) od spoločnosti GestureTek. Na rozdiel od predchádzajúceho príkladu, tento projekt sa snaží ľudí odpútať od reality zdĺhavej a nudnej rehabilitácie. Každá časť rehabilitácie je podaná formou jednoduchých hier, pri ktorom sa zabavia ako deti, tak aj dospelí.

Obr. 4.2: IREX hry [3]

Program obsahuje niekoľko hier, ktoré sa zaoberajú bilancovaním, otáčaním, ohýbaním alebo zdvíhaním vecí. Existuje viacero hier na dané cvičenia, takže si pacient môže vybrať, ktorá mu vyhovuje. Na výber je napríklad futbal, hra na bubnoch, volejbal a mnoho ďalších.[3]

5 Slovenská akadémia vied

Projekt bol vyvíjaný so spoluprácou doktorov zo Slovenskej akadémie vied, ktorý sa venujú rehabilitáciám po mozgových príhodách. Ich štúdie sa zaoberali aj napríklad odzrkadľovaním zdravej končatiny pomocou Mirror-box alebo rozhýbavaním znecitliveného zápästia zariadením BCI-RAS (Brain-computer interface with robot-assisted).

Experiment s použitím Mirror-box prebiehal ako séria cvikov na zdravých dobrovoľníkoch a na pacientoch po cievnej mozgovej príhode.

Obr. 5.1: Mirror-box[2]

Experiment pozostával z niekoľkých krokov:

- fáza relaxovania
- pozorovanie pohybu
- vykonávania a pozorovanie pohybu v zrkadlovom boxe

Počas cvikov sa neustále zaznamenávala mozgová aktivita pomocou desiatich elektród rozmiestnených nad senzo-motorickou oblasťou a jednej elektródy nad okcipitálnou oblasťou mozgu. Výsledok niekoľko dňového testovania boli zmeny oscilačných rytmov v relaxačnom stave oproti stave v pohybe.

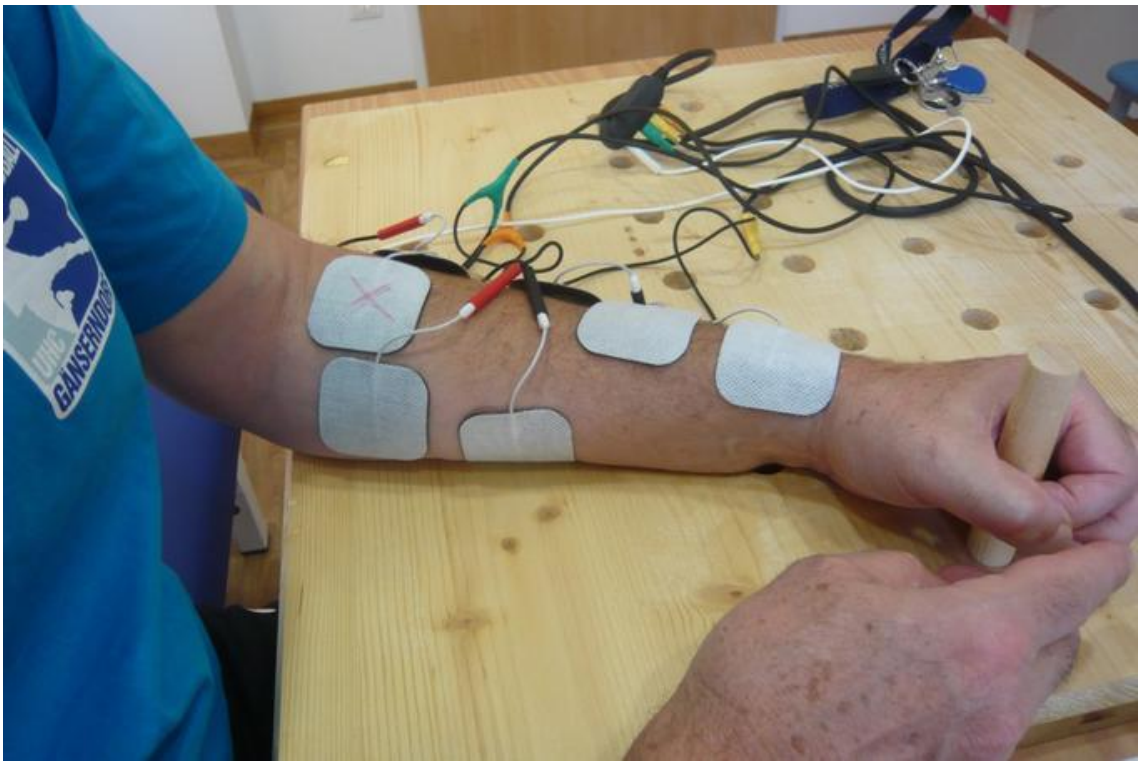
Po niekoľkých cvičeniach s Mirror-boxom pacient mohol prejsť na cvičenia s robotickým ramenom BCI-RAS. Z predošlého experimentu sa dalo predikovať, kedy si pacient reálne predstavuje pohyb ruky pomocou výsledných hodnôt z EEG.

Obr. 5.2: Robotické rameno BCI-RAS [2]

Ako v predchádzajúcom teste, pacient bol v stave relaxovania, keď dostal pokyn na predstavu pohybu ruky. Pri zmene oscilačných rytmov sa spustilo robotické rameno, ktoré vykonalo pohyb zápästia hore a dole. [13]

5.1 Funkčná elektrická stimulácia

Functional Electrical Stimulation (FES) je prístroj, ktorý vytvára malé elektrické stimulačné impulzy cez povrchové elektródy, ktoré slúžia na vyvolanie svalových kontrakcií. Ak sú tieto svalové kontrakcie vykonávané správne, vedú k funkčným pohybom, ktoré sa dajú použiť na terapeutické zákroky a na doplnenie stratených funkcií. [9]



Obr. 5.3: Functional Electrical Stimulation [11]

6 Návrh systému

Pri návrhu systému sme sa riadili požiadavkami skupiny doktorov zo Slovenskej akadémie vied. Bola stanovená predstava o konečnom produkte, spolu s tým, že popri testovaní softvéru sa budú požiadavky ešte dopĺňať.

Časť systému už bola implementovaná v diplomovej práci Petri Hardoňovej[16]. Projekt bol vyvíjaný na platforme A-frame kvôli širokému poľu pôsobnosti. Program sa bude dať spustiť na ktoromkoľvek zariadení s web prehliadačom, ktorý podporuje webový rámec WebXR, ako sú napríklad:

- Chrome
- Firefox
- Opera
- Microsoft Edge
- Samsung Internet
- Oculus Browser

Systém teda bude spustiteľný aj na VR headsetoch, ktoré disponujú takýmto prehliadačom, ako napríklad Hololens alebo Oculus Rift. Program bude dostupný vo webovom nástroji Glitch, ktorý je prístupný zadarmo. Táto aplikácia slúži na vytváranie webových aplikácií a priamo v jej prostredí je možné kód editovať, pričom sa aplikácia neustále aktualizuje. Na aplikáciách je možné robiť kolaboratívne v danom čase. Glitch ponúka aj web hosting zadarmo a každá aplikácia má svoju vlastnú url (nazov-projektu.glitch.me), kde je možné ju spúšťať.

Rozhranie systému sa skladá z 2 častí:

- Rozhranie pre doktora – nastavenie parametrov rehabilitácie

- Rozhranie pre pacienta – len sledovanie zmien v programe

Základné časti oboch rozhraní už boli implementované. Požiadavka bola mierne prerobiť existujúci systém a doplniť ho o ďalšie požiadavky, hlavne o zlepšenie animácií a doplnenie parametrov testu. Pôvodný systém ponúkal len spustenie animácie úchopu ruky. Jedna z požiadaviek na systém bola aj na prerobenie animácie za účelom dosiahnutia vyššej kvality a realistickejšieho pohybu.

Ďalej by mal systém vykonať sekvenciu týchto animácií, ktoré sú zamerané na podporu koordinácie pohybu horných končatín. Po zadaní parametrov a spustení programu by mal systém začať sekvenciu určitého počtu animácií ruky.

Ďalšie scenáre sa budú pravdepodobne dotvárať v priebehu implementácie systému. Ako napríklad presun kocky, zdvíhanie pera alebo držanie guličky.

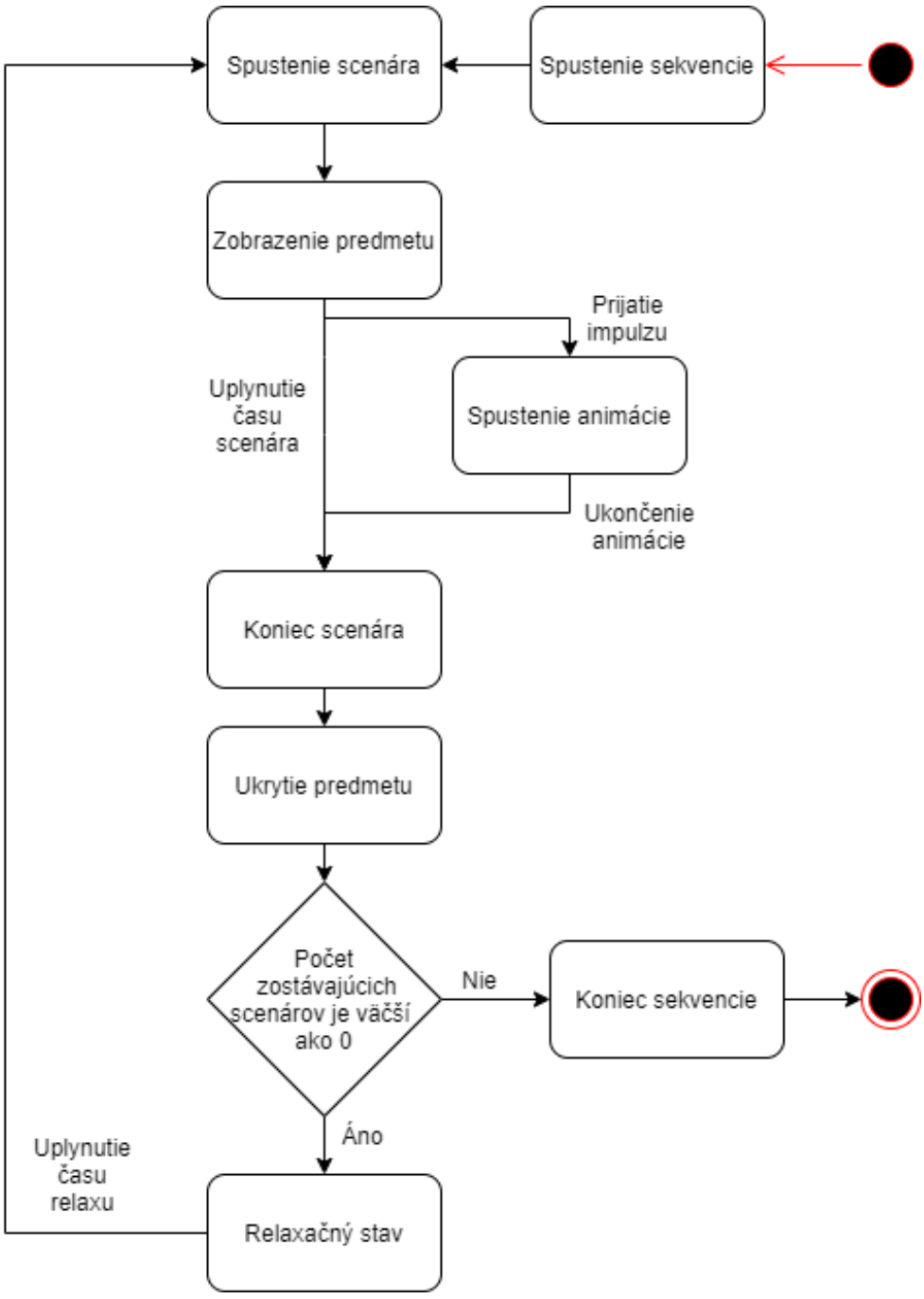
6.1 Návrh rehabilitačného procesu

Rehabilitácia sa bude vykonávať ako sekvencia niekoľkých scenárov s animáciami. Doktor bude vedieť zadať počet daných scenárov. Medzi scenármi bude čas, ktorý bude slúžiť ako doba na relax, len pohľad pred seba bez akejkoľvek činnosti. Jeden scenár bude trvať určitý čas, ktorý sa bude dať nastaviť. Na začiatku scenára sa objaví predmet, ktorý slúži ako pomocný objekt ku animácii, napríklad pri úchopu pohára, bude daný objekt samotný pohár. Ukončenie scenára bude možné dvoma spôsobmi:

- Čas scenára uplynie
- Príde príkaz na vykonanie animácie

Pri obdržaní správy a spustení animácie, sa čas scenára restuje a keď sa animácia skončí, tak program prejde do stavu relaxu a predmet sa schová. Celá sekvencia sa ukončí, ak:

- Terapeut resetuje sekvenciu zo svojho rozhrania
- Všetky scenáre už boli absolvované



Obr. 6.1: Proces rehabilitácie bez zásahu terapeuta

6.2 Návrh rozhrania pre pacienta

Pacient je v príjemnej miestnosti, bez nejakého väčšieho rozptyľovania. Pred sebou má stôl na ktorom je položený 3D objekt ruky spolu s 3D objektom pohára.

Obr. 6.2: Pôvodné rozhranie pre pacienta

Cieľom je prerobenie animácie a doplnenie systému o sekvenciu animácií. Pohár, ako cieľový objekt, bude na začiatku skrytý. Po spustení sekvencie z rozhrania určeného pre doktora, sa pohár objaví, zatiaľ bez pohybu ruky. Pacient má za úlohu si daný pohyb predstaviť. Ak doktor uvidí zmeny v jeho elektroencefalogramu, pošle signál do prostredia pacienta, čo spustí animáciu ruky, ktorá chytí pohár a vráti sa na pôvodné miesto. Po ukončení animácie sa pohár skryje a nasleduje ďalší scenár alebo koniec sekvencie.

6.3 Návrh rozhrania pre doktora

Doktor vidí pacientov pohľad v danom čase, aby vedel ako rehabilitácia prebieha a aby vedel poslať impulz do systému na spustenie pohybu.

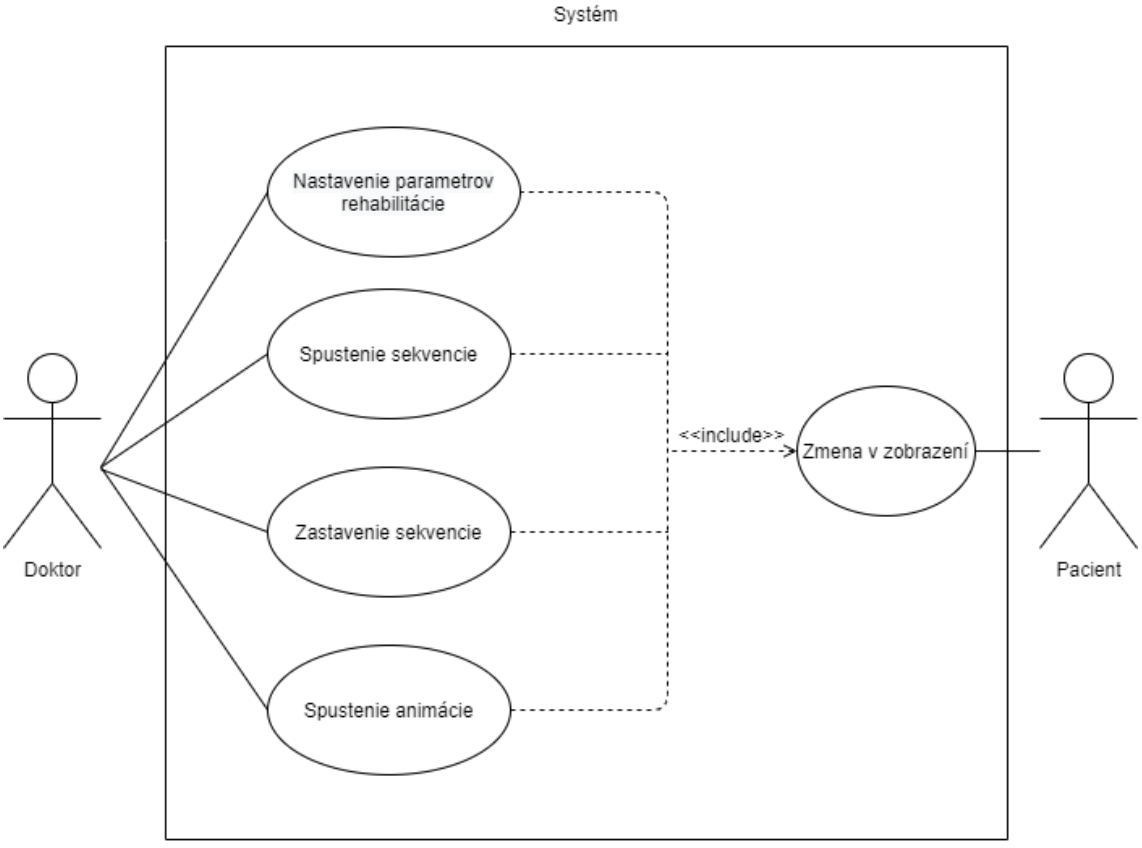
Jednou z požiadaviek na vylepšenie tohto rozhrania bola, aby systém vedel prijať daný impulz, ktorý spustí animáciu. Impulz by mal doktor byť schopný poslať aj z webového rozhrania, nie len z Robo-arm. Systém by mal ponúknuť možnosť zvoliť počet animácií v sekvencii a dĺžku doby, kým sa čaká na prijatie impulzu. Po prijatí signálu sa spustí animácia. Po jej ukončení sa spustí ďalší scenár alebo ukončí sekvencia. Doktor chce byť informovaný o aktuálnom počte nasledujúcich scenárov. Posledná požiadavka bola aj na ukončenie sekvencie v jej priebehu, zmeniť nastavenia a spustiť novú sekvenciu.

6.4 Prípady použitia

V systéme sa nachádzajú len dvaja aktéri, doktor a pacient. Každý má svoje rozhranie, ktoré mu umožňuje vykonávať rozličné funkcie.

Na obrázku 6.3 môžeme vidieť možnosti, ktoré by systém mal byť schopný poskytnúť doktorovi:

- Nastavenie parametrov rehabilitačného procesu
- Spustenie sekvencie
- Zastavenie sekvencie
- Spustenie animácie



Obr. 6.3: Diagram prípadov použitia

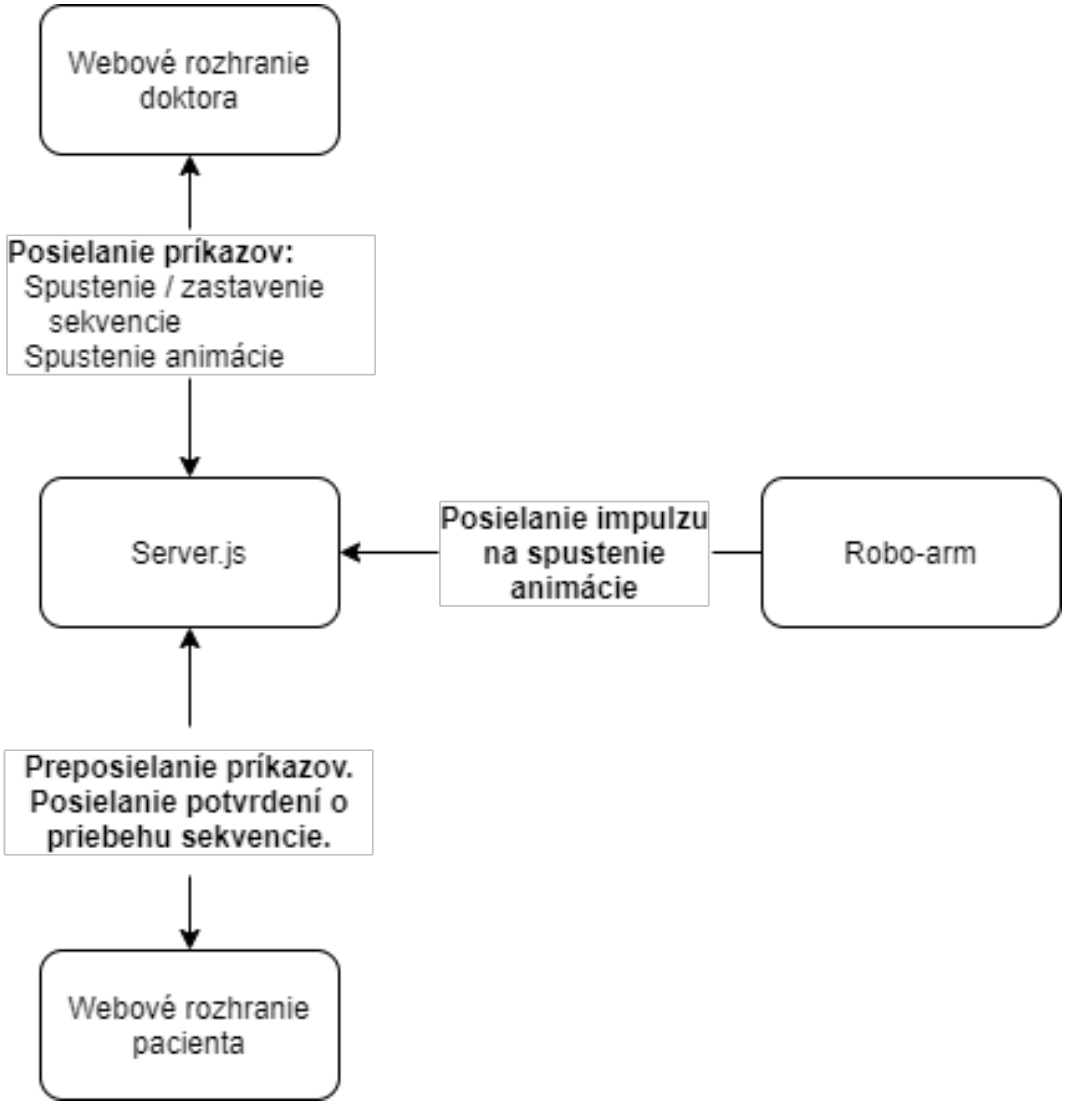
Pacient len pozoruje zmeny v prostredí a snaží sa na nich reagovať.

Základný úspešný priebeh programu zahŕňa:

1. Systém zobrazí v rozhraní pre doktora vizualizáciu z pohľadu pacienta a formulár na zadávanie parametrov
2. Doktor zadá parametre rehabilitačného procesu:
   - Počet scenárov v sekvencii
   - Dĺžka zobrazenia scenára (čakanie na impulz)
3. Doktor spustí sekvenciu
4. Systém zobrazí v rozhraní pacienta objekt, s ktorým má byť vykonaná animácia
5. Pacient si predstaví pohyb a ukáže sa to na elektroenefalograme
6. Doktor pošle impulz do systému
7. Systém spustí animáciu s aktuálnym predmetom
8. Systém opakuje proces z bodov 4 až 7 dokým ich nevykoná toľko krát, koľko bolo zadaných počet scenárov v sekvencii v bode 2
9. Systém ukončí sekvenciu

6.5 Komunikácia rozhraní

Pre daný systém je veľmi dôležitá komunikácia medzi rozhraniami doktora a pacienta. Webový rámec Express pre knižnicu Node.js je skvelé riešenie pre posielanie správ v menších projektoch. V našom systéme nie je potrebná zložitá alebo šifrovaná komunikácia. Stačí nám posielať jednoduché správy, ktoré budú obsahovať príkazy potrebné na ďalšie fungovanie inej časti rozhrania. Je potrebné si vytvoriť server, ktorý bude slúžiť na preposielanie správ medzi dvoma rozhraniami a zároveň bude aj prijímať správu z Robo-arm.



Obr. 6.4: Diagram komunikácie medzi rozhraniami

Posielanie správ bude fungovať na princípe soketov. Príklad posielania správ cez funkciu *socket.emit*(*message, messageBody*), kde

- message – vlastné pomenovanie eventu
- messageBody – samotná správa, ktorá môže obsahovať aj objekty

Funkcia *socket.send*(*messageBody*) je rovnaká ako *socket.emit*(*"message", messageBody*). Posiela správy pod názvom eventu "message".

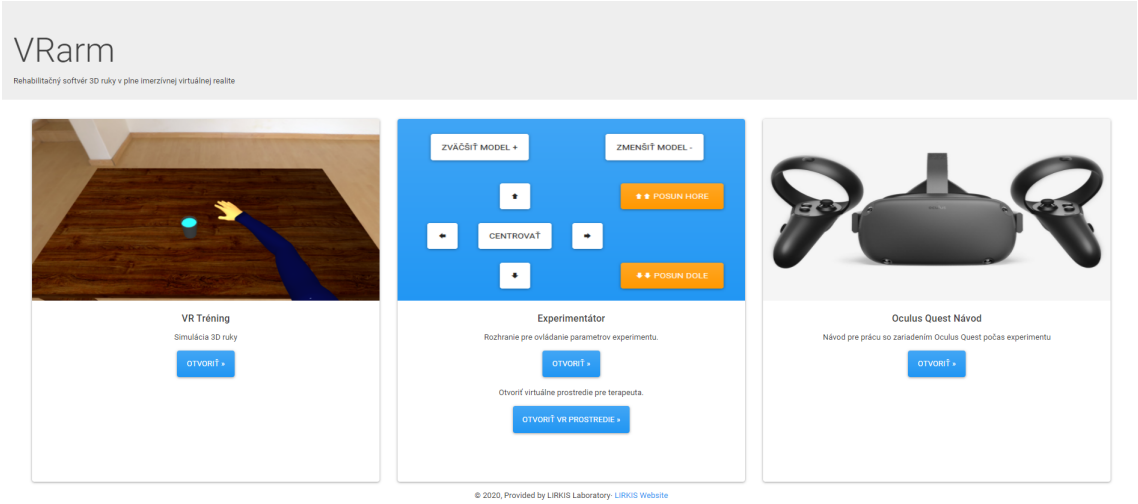
Príklad prijímania správ cez funkciu *socket.on*(*message, function*(*messageBody*)), kde

- message – pomenovanie eventu
- function – funkcia, ktorá sa spustí pri odchytení soketu
- messageBody – samotná správa ako parameter funkcie

7 Implementácia systému

Aplikácia sa skladá z 2 hlavných častí. Jedna časť je navrhnutá pre doktora, ktorý ovláda prostredie aplikácie. V jeho časti systém kontroluje priebeh sekvencií, ako je aj napríklad výber ďalšej animácie. V druhej časti, určenej pre pacienta, ktorý prechádza rehabilitačným procesom, sa vykonávajú kalkulácie animácií a ich celkový manažment.

Na obrázku 7.1 môžeme vidieť úvodnú stránku programu. Na prvej karte je odkaz pre pacienta na virtuálne prostredie s 3D objektom ruky. Na druhom má terapeut na výber dve možnosti. Rozhranie, v ktorom sa zadávajú parametre cez vstupy z klávesnice alebo rozhranie, kde sa nachádza v rovnakom prostredí ako pacient.



Obr. 7.1: Úvodná stránka programu

Vďaka kolaboratívnemu prostrediu sa môžu navzájom vidieť a aj spolu komunikovať. Zdieľanie entít je zabezpečené rámcom *Networked-Aframe*. Do entity scény bolo potrebné pridať komponent *networked-scene* a vytvoriť šablónu na pripojeného avatara v prostredí, ktorá som pridá cez jej id do entity avatara. Zvuk sme pridali taktiež pomocou tohto rámca, do entity scény cez parameter audio. Audio sa používa len cez adaptér, ktorý ho podporuje. Výsledná entita scény:

```
<a-scene
  networked-scene="audio:true; adapter: easyrtc">
```

Následne je potrebné pridať komponent *networked-audio-source* do šablóny avatara, ktoré zabezpečí, že sa zvuk bude prenášať od užívateľa, ktorému daná entita patrí. Takže výsledná šablóna a entita avatara vyzerá takto:

```
<template id="avatar-template">
      <a-entity class="avatar" networked-audio-source>
      </a-entity>
</template>
<a-entity
    networked="template:#avatar-body;attachTemplateToLocal:false;">
</a-entity>
```

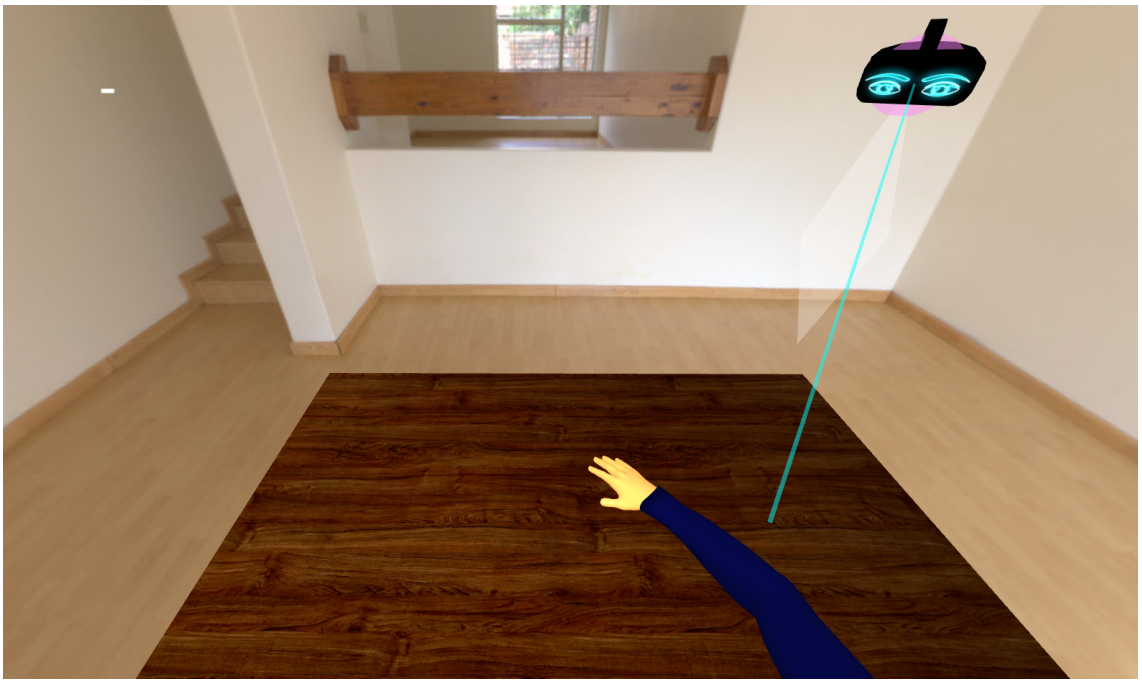
Keď je parameter *attachTemplateToLocal* nastavený na pravdivostnú hodnotu, ktorá je nastavená automaticky, entita sa ukáže aj u užívateľa, ktorému patrí.

Pri tejto implementácií sa bude prenášať len poloha, rotácia a zvuk druhého používateľa. Ak chcem zdieľať aj ďalšie informácie o používateľovi, napríklad zmeny šírky, je potrebné to nastaviť navyše cez *Naf.schema*:

```
NAF.schemas.add({
  template: '#avatar-template',
  components: [
    'position',
    'rotation',
    'scale',
    {
      selector: '.head',
```

```
        component: 'material',
        property: 'color'
      },
    ]
  });
```

Výsledok implementácie môžeme vidieť na obrázku 7.2, ktorý je odfotený z pohľadu pacienta a oproti vidíme avatara druhého používateľa, terapeuta.

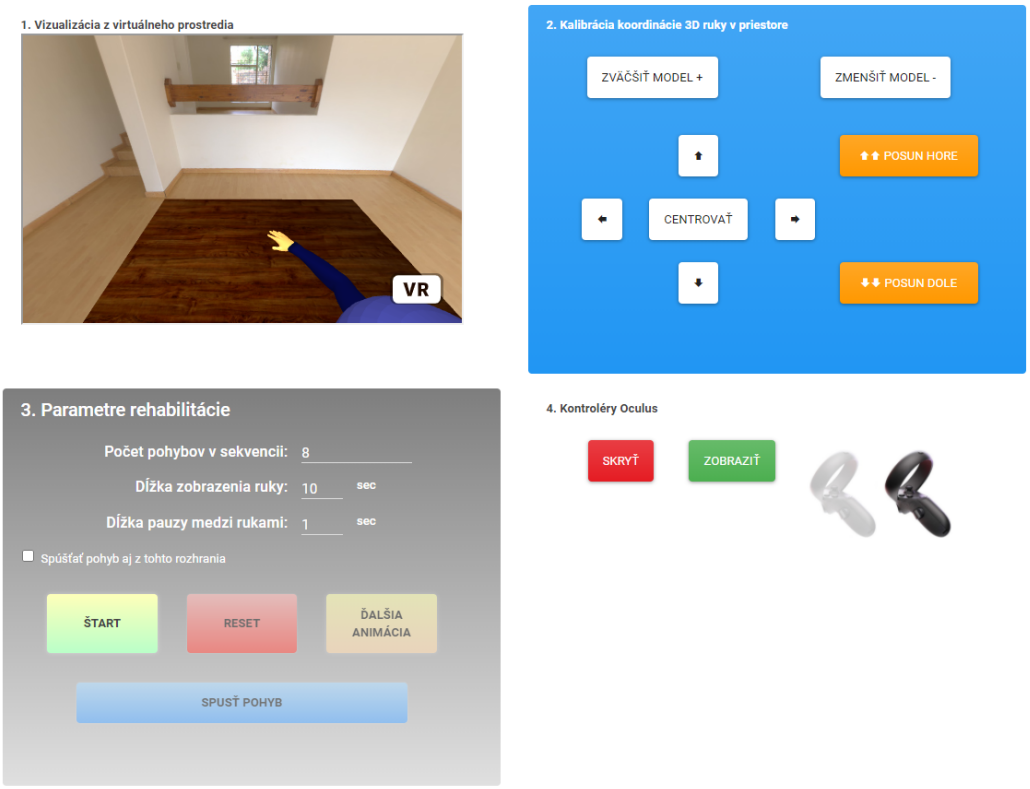


Obr. 7.2: Avatar terapeuta z pohľadu pacienta

7.1 Rozhranie pre doktora s parametrami

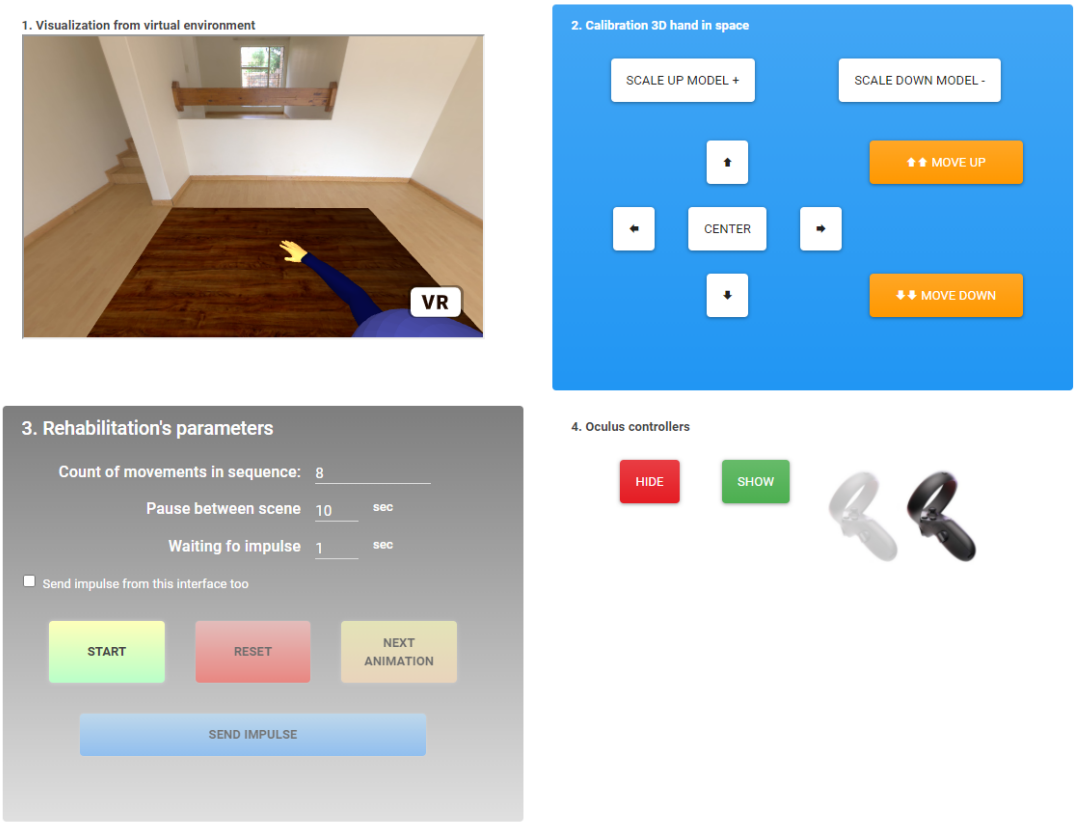
Prvou možnosťou doktora bolo prejsť do rozhrania, kde sa nastavujú parametre rehabilitačného procesu, ktoré môžeme vidieť na obrázku 7.3. Rozhranie je rozdelené na niekoľko častí:

- vizualizácia virtuálneho prostredia, v ktorom sa nachádza pacient
- implementované v predchádzajúcom systéme - konfigurácia polohy 3D ruky.
- nachádzajú parametre rehabilitácie a potrebné ovládanie sekvencie.
- implementované v predchádzajúcom systéme – možné skrytie ovládačov prilby Oculus



Obr. 7.3: Stránka pre doktora

Text stránky je možné prepnúť aj do angličtiny.



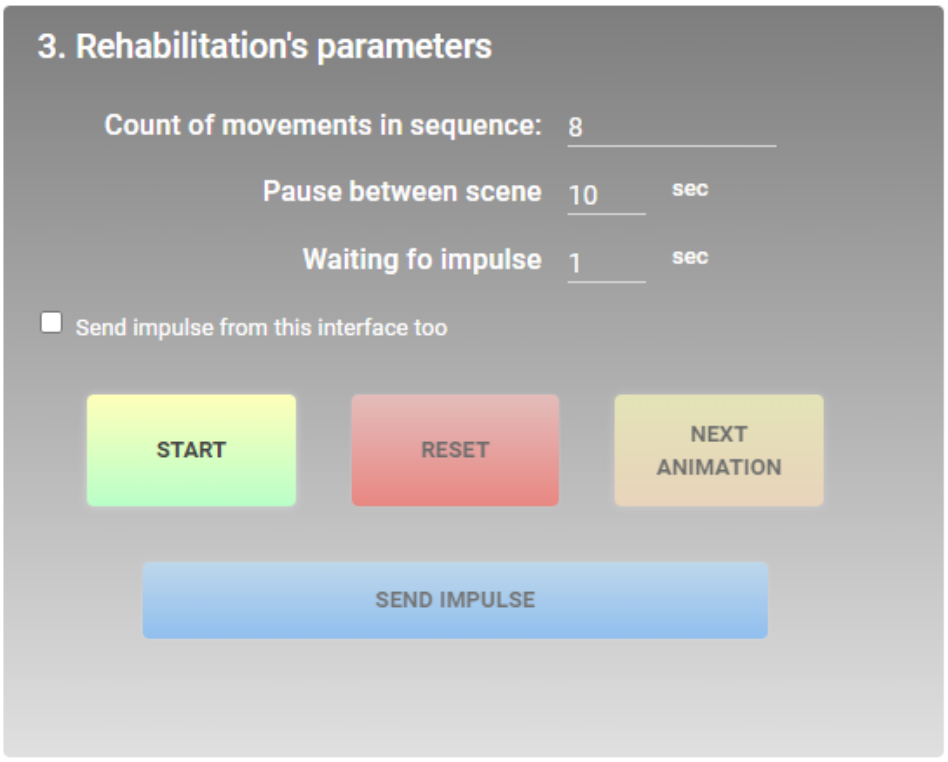
Obr. 7.4: Stránka pre doktora v anglickom jazyku

Pred spustením sekvencie vie terapeut nastaviť niekoľko parametrov:

- Počet pohybov v sekvencii – určuje celkový počet animácií v jednej sekvencii
- Dĺžka zobrazenia ruky – určuje čas v sekundách, ktorý program čaká na impulz, ktorý spustí animáciu
- Dĺžka pauzy medzi zobrazeniami – medzi zobrazeniami ruky sa všetky objekty, okrem ruky, skryjú na daný počet sekúnd
- Potvrdenie, že sa impulz môže poslať aj z prostredia programu, nie len zo zariadenia Robo-arm

Doktor môže kedykoľvek sekvenciu prerušiť pomocou tlačidla "reset" a zmeniť nastavenia, no počas sekvencie sa už parametre meniť nemôžu. Impulz na spustenie animácie zabezpečuje Robo-Arm, ale doktor si môže zvoliť posielať príkaz aj

pomocou tlačidla "spusť pohyb". Po stlačení tlačidla "štart" sa spustí sekvencia. Po prijatí soketu s pokynom na štart sekvencie sa náhodne zvolí prvá animácia. Tlačidlo "ďalšia animácia" slúži na prepnutie aktuálnej animácie na ďalšiu.



Obr. 7.5: Parametre rehabilitácie

Na výber sú 3 rôzne druhy animácií:

- Pohár – ruka sa pohne ku poháru, chytí ho a vráti sa na pôvodnú pozíciu
- Kocka – ruka sa pohne ku kocke, zdvihne ju a presunie na iné miesto
- Kľúče – program náhodne vyberie, ktorý z 3 kľúčov ruka zoberie a do ktorej zámky ho presunie, otočí s ním a vráti sa na začiatočnú pozíciu

Pred spustením sekvencie vie doktor nastaviť parametre a spustiť sekvenciu s danými parametrami, pričom tlačidlá „reset“ a „spusť pohyb“ sú vypnuté. Po spustení sekvencie sa tlačidlo „štart“ vypne, „reset“ zapne a tlačidlo „spusť pohyb“ sa zapne, ak bolo potvrdené, že sa impulz môže poslať aj z prostredia programu.

Na začiatku každého zobrazenia vykoná animáciu kópia ruky s kópiami daných predmetov, aby pacient vedel, čo si má predstaviť. Keď sa animácia skončí, kópie objektov sa skryjú. Ak si to pacient predstaví, doktor pošle správu systému, aby spustil animáciu originálnymi objektmi.

Na začiatku každej animácie sa spúšťa odpočet pomocou funkcie

```
setTimeout(() => {
    //kód, ktorý sa spustí po danom čase
}, duration * 1000);
```

kde duration je dĺžka zobrazenia ruky z menu. V menu sa dĺžka udáva v sekundách, zatiaľ čo v tejto funkcii v milisekundách, preto je potrebné túto hodnotu násobiť.

Ak čas uplynie a nepríde impulz na spustenie animácie, duration sa nastaví na dĺžku pauzy medzi zobrazeniami. Ak impulz príde, timeout sa vynuluje a keď po dokončení animácie v rozhraní pacienta príde správa o jej ukončení, timeout sa znova nastaví na pauzu. Po uplynutí času pauzy sa znova vyberie náhodne, ktorá animácia nasleduje a proces sa opakuje toľko krát, koľko bolo zvolených pohybov v sekvencii. Po zvolení animácie sa posiela správa do rozhrania pacienta, ktorá obsahuje informácie o daných animáciách:

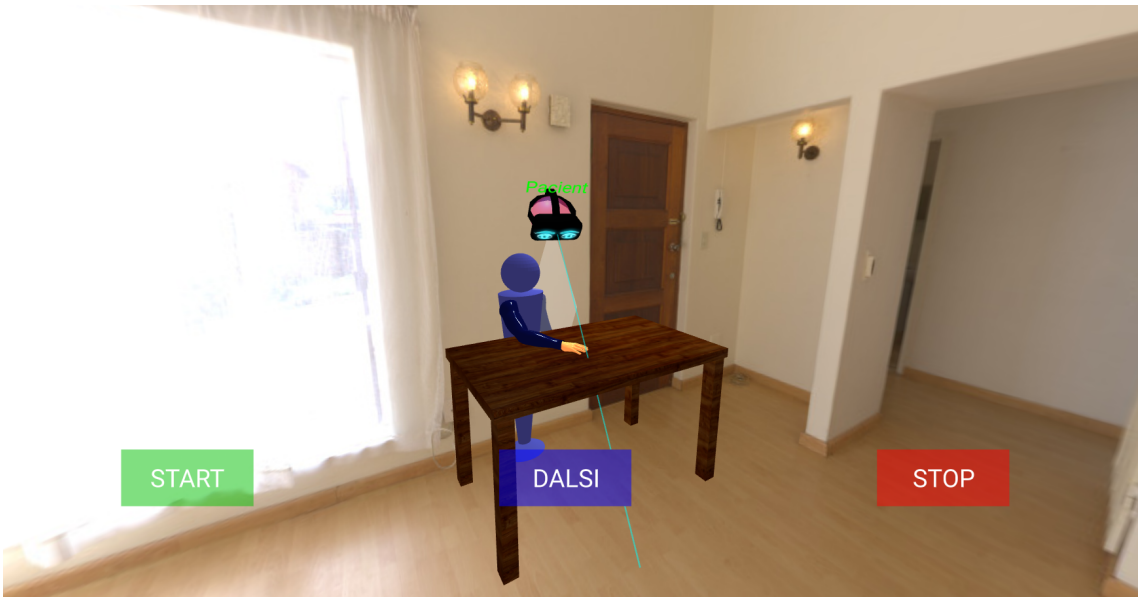
```
socket.send({key: false, cube: false, cup: true });
```

Táto konkrétna správa informuje o tom, že nasledovná animácia bude animácia úchopu pohára.

7.2 Rozhranie pre doktora vo virtuálnom prostredí

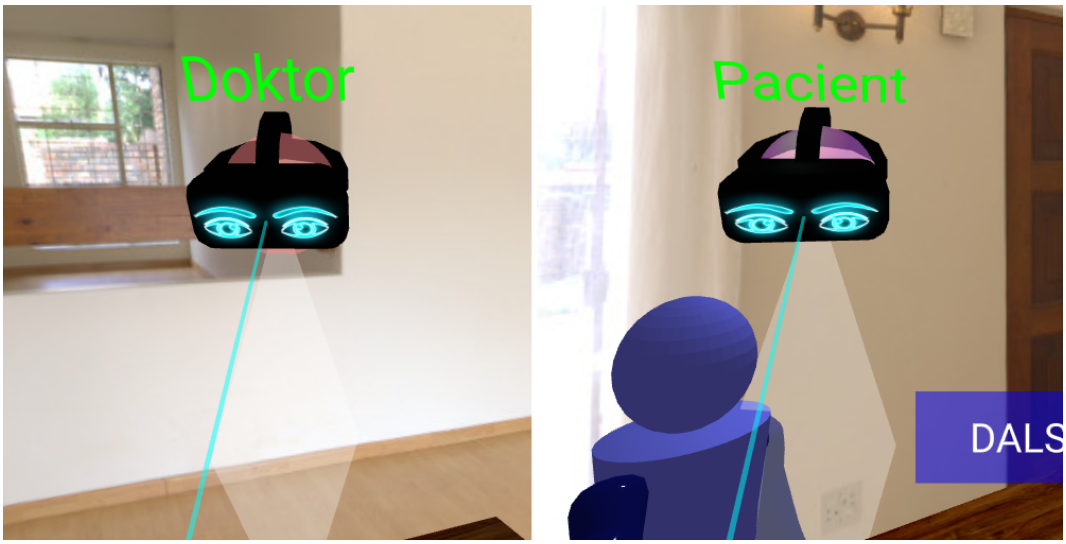
Druhá možnosť terapeuta je pripojiť sa do rovnakého prostredia, v akom sa nachádza pacient cez prilbu Oculus. V rozhraní vidí 3 tlačidlá, ktoré môže stlačiť pomocou Oculus ovládačov. Dané tlačidlá nie sú zdieľané pomocou *networked*, takže ich vidí len doktor.

- ŠTART – začne rehabilitáciu s defaultne nastavenými parametrami
- STOP – zastaví sekvenciu, ak je v priebehu
- ĎALŠÍ – prepne aktuálnu animáciu na ďalšiu



Obr. 7.6: Pohľad z virutálneho rozhrania doktora

Každý, kto sa prihlási do systému prostredníctvom stránky pre doktorov, bude mať nad svojím avatarom napísané, že ide o entitu doktora. Tak isto pacient, po pripojení cez svoju stránku, bude mať nad svojím avatarom napísané *„Pacient“*.



Obr. 7.7: Avatar druhého aktéra scény z pohľadu pacienta a doktora

7.3 Rozhranie pre pacienta

Pacient sa nachádza vo virtuálnej miestnosti, kde pred sebou vidí stôl a ruku. Bez zmeny v prostredí doktora nedôjde k zmene ani v prostredí pacienta.

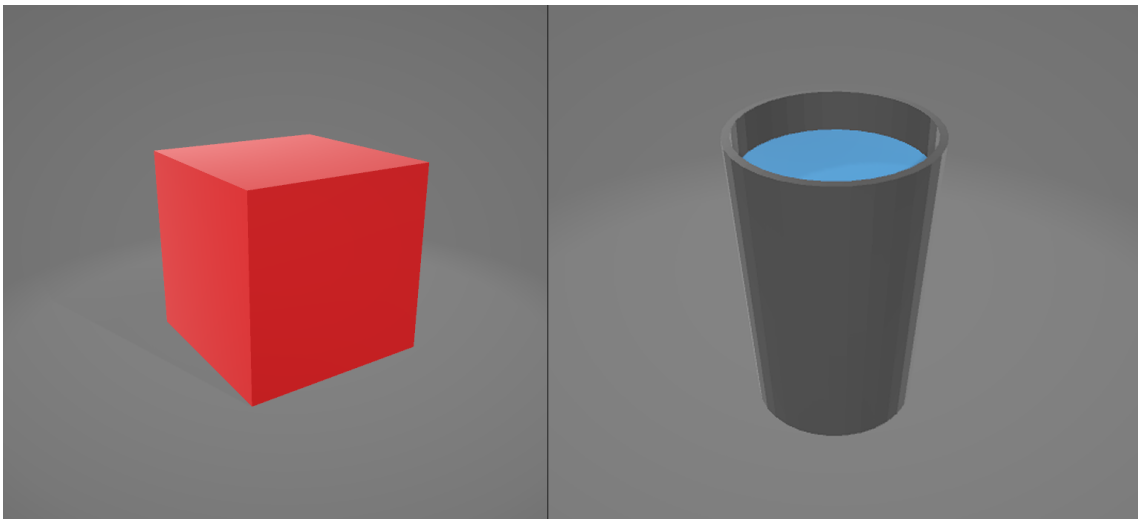
Využili sme funkciu *AFRAME.registerComponent*() na registráciu komponentu v javascriptovom súbore. Pri použití funkcie tejto funkcie je potrebné zadať názov komponentu a vložiť ho do entity.

Cez komponent sme si už vedeli poslať referencie na iné entity, preto nám stačilo registrovať jeden komponent a ostatné si poslať pomocou schémy.

```
<a-entity id="arm"
    gltf-model="https://cdn.glitch.com/85b4452e-cc7b-4d31-84fb-
    5d8576b684ee%2Fthis3.glb?v=1617800243591"
    manager="key1: #key1; key2: #key2; key3: #key3; lock1: #lock1;
    lock2: #lock2; lock3: #lock3; armCopy: #armCopy; cup: #cup;
    cube: #cube">
</a-entity>
```

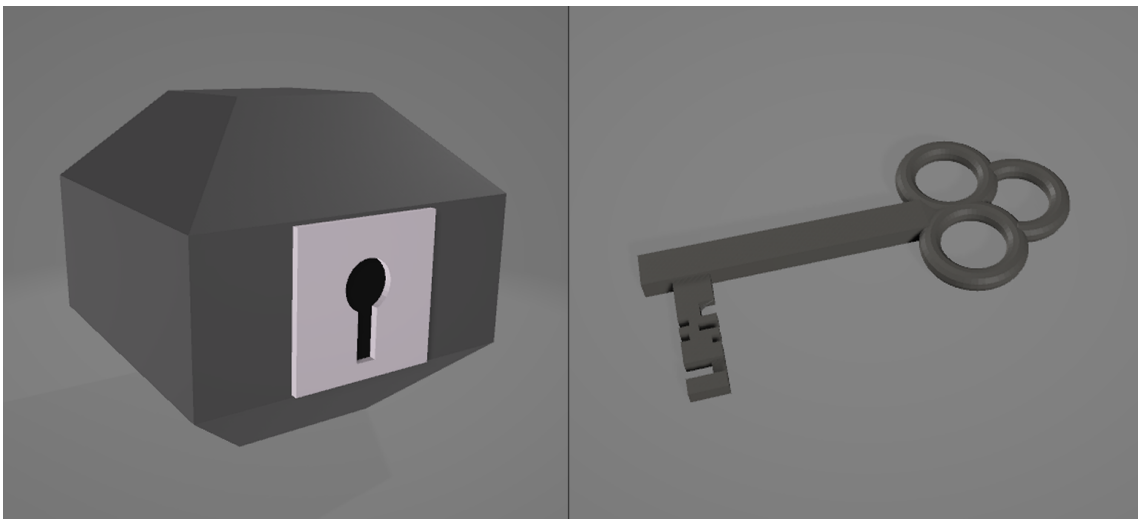
```
AFRAME.registerComponent("manager", {
  schema: {
    key1: {type: 'selector', default: ""},
    key2: {type: 'selector', default: ""},
    key3: {type: 'selector', default: ""},
    lock1: {type: 'selector', default: ""},
    lock2: {type: 'selector', default: ""},
    lock3: {type: 'selector', default: ""},
    armCopy: {type: 'selector', default: ""},
    cup: {type: 'selector', default: ""},
    cube: {type: 'selector', default: ""}
  }})
```

Všetky objekty sú vyexportované modely z prostredia Blenderu ako glb súbor a načítavané pomocou *gltf-model* parametra v aframe entite. Model sa do projektu importuje do balíka assets, kde im Glitch pridá url adresu, ktorá sa dá skopírovať a vložiť do entity.



Obr. 7.8: 3D objekty kocky a pohára

Zatiaľ čo objekty kocka a pohár sa v priestore nachádzajú len raz, kľúče a schránky sú po troch. Po tom ako sa náhodne vyberú cieľové objekty, program zmení ich farbu na zelenú, aby pacient vedel s ktorými objektami má pracovať.



Obr. 7.9: 3D objekty schránky s kľúčom

Ďalšia funkcia z komponentu, ktorú sme využili je funkcia init(), ktorá sa vykoná po inicializácii entity.

```
AFRAME.registerComponent("manager", {
  schema: {},
  init: function(){}
```

```
})
```

V tejto funkcii sa načítavajú všetky entity zo schémy, z ktorých je potrebné dostať model. Entity zo schémy sa nachádzajú v premennej data, ktorá je zahrnutá už v komponente, pod svojím názvom. Na získanie modelu z entity je potrebné pridať EventListener, ktorý zabezpečí, že sa spustí daná funkcia po načítaní objektu v entite.

```
cup = this.data.cup;
cup.addEventListener('model-loaded', function () {
  cup = cup.components["gltf-model"].model.parent;
});
```

Po načítaní všetkých objektov sa posielajú modely do inicializačných funkcií daných animácií. Každá animácia má svoje funkcie, ktoré sa starajú o priebeh a výpočty danej animácie, opísaných v kapitole 8.

Animácie sú zapínané po obdržaní správy z rozhrania doktora. V rozhraní pacienta sa prijíma soket, ktorého správa obsahuje, ktorá animácia sa má pripraviť. Tu sa správa preposiela do entity ruky, v ktorej bol komponent registrovaný.

```
var arm = document.querySelector('#arm');
arm.emit("cupAnimationShowUp", {}, false);
```

Komponent ho následne prijme cez EventListener.

```
arm.addEventListener('cupAnimationShowUp', function () { })
```

Entita počúva na niekoľko správ:

- cupAnimationShowUp – pripraví sa na scénu animácia úchop pohára
- cubeAnimationShowUp - pripraví sa na scénu animácia s kockou
- keyAnimationShowUp - pripraví sa na scénu animácia s kľúčmi a schránkami
- hideAll – skryje všetky objekty, využíva sa pri pauzách medzi animáciami alebo keď sa skončí sekvencia
- move – spustí sa animácia aktuálne zobrazeného objektu

Po obdržaní prvých 3 správ, komponent nastaví animáciu a vo funkcii *tick*() sa spúšťajú funkcie adekvátnej animácie. Daná funkcia sa vykonáva pri každej aktualizácii scény.

```
AFRAME.registerComponent("manager", {
  schema: {},
  init: function(){},
  tick: function(){}
})
```

Informácie o tom, ktorá animácia sa má aktuálne zobraziť sa uchovávajú v enumeračných typoch, ktoré sledujú aj či sa aktuálne hýbe kópia alebo originál ruky, alebo sa čaká na správu.

V *tick*() funkcii už len stačí switch, ktorý volá funkcie podľa aktuálnej hodnoty z enumeračného typu.

```
const animStates = { WAIT: 1, KEY: 2, CUBE: 3, CUP: 4 };
var animState;

switch(animState){
 case animStates.KEY:
     break;
 case animStates.CUBE:
     break;
 case animStates.CUP:
     break;
 }
```

Vo funkciách, ktoré očakávajú správu na spustenie animácie, stačí len zmeniť stav a vo switch funkcii sa zavolajú funkcie danej animácie.

8 Animácie

Každá animácia má svoje vlastné výpočty, aj keď sú veľmi podobné. Aj úchop skutočnej ruky sa líši vzhľadom na predmet, ktorý sa chystá chytiť.

Pred vykonaním animácie sa vypočítajú koncové rotácie všetkých častí ruky. Následne program upravuje len rotáciu kostí v modeli, aby sa dostali na koncové pozície. Pozície kostí sa nesmú meniť. Pri posune jednej z kostí sa posunie celá kostra spolu s ňou. Preto ak chceme zmeniť pozíciu napríklad zápästia, je potrebné zmeniť rotáciu predošlých kĺbov, ako aj v reálnom svete.

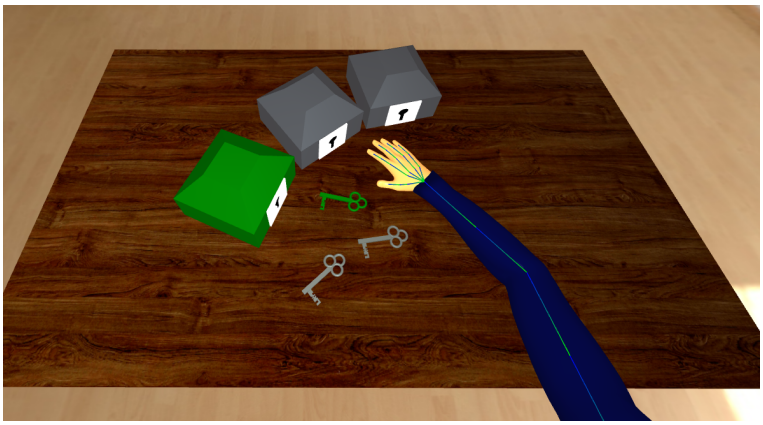
Obr. 8.1: Vykreslené kosti ruky pomocou triedy *SkeletonHelper*

Kalkulácie rotácií častí ruky fungujú na princípe inverznej kinematiky. Pri každom pohybe máme určený cieľ a algoritmy vypočítavajú uhly kostí v hierarchii, aby sa dostali na koncovú pozíciu. Keďže sa koncové pozície počítajú, je možné

presunúť predmety podľa potreby. Ale ak cieľové objekty nie sú v dosahu ruky, animácia sa nevykoná. Koncové pozície prstov sú dané staticky, kedže aj pre rôzne pozície predmetov sa uchopujú rovnako.

8.1 Algoritmus animácie kľúča

Animácia je rozdelená na niekoľko častí. Posun ruky ku zvolenému kľúču, zdvihnutie kľúča, presun ruky ku zámke, posun kľúča do zámky, otočenie kľúča v zámke a vrátenie sa na pôvodnú pozíciu. Každá časť má koncové pozície kostí ruky, ktoré bolo potrebné vypočítať.



Obr. 8.2: Animácia s kĺúčmi

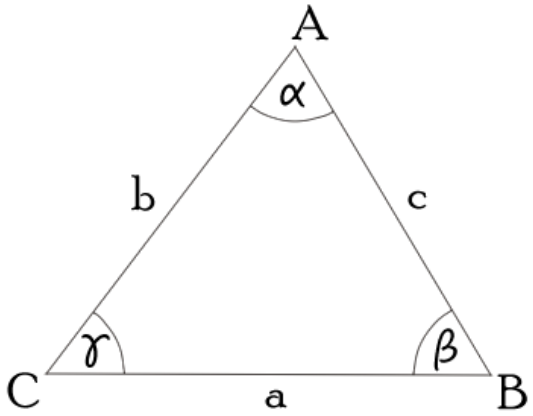
8.1.1 Výpočet súradníc pre presun ku cieľovému objektu

Ako prvá sa najprv ruka presunie ku cieľovému kľúču, ktorý je volený náhodne programom spolu s cieľovou zámkou. Pre zvýraznenie cieľových objektov bola použitá zmena materiálu na zelenú farbu, aby pacient vedel, aký konkrétny pohyb si má predstaviť.

Rotácie je možné počítať aj v 3D priestore. 2 vrcholy máme určené, kľúč a rameno ruky. Tretí vrchol, lakeť, nie je problém získať, no výsledkom je celá množina bodov, kružnica. Výpočet vhodného bodu by bol zložitejší, ak by sa dala vôbec zabezpečiť vysoká úspešnosť výpočtov. Preto sme zvolili jednoduchší algoritmus. Rotácia kostí sa počíta na každej osi zvlášť. Na výpočet uhlov bol zvolený algoritmus na základe kalkulácie uhlov v trojuholníku v 2D priestore. Strany trojuholníka nebolo zložité zistiť. Všetko sú len vzdialenosti bodov v priestore, čo má

pokryté už samotná knižnica THREE.js pomocou funkcie *Vector3.distanceTo*(*Vector3*). Keďže sme počítali v každej osi osobitne, stačilo vynulovať danú os vo vektoroch a následne vypočítať jednotlivé dĺžky. V konečnom trojuholníku, bude zápästie na rovnakej pozícii ako kľúč.

- **a** - vzdialenosť pozície ramena (**B**) od pozície kľúča (**C**)
- **b** - vzdialenosť pozície lakťa (**A**) od pozície zápästia (**C**)
- **c** - vzdialenosť pozície ramena (**B**) od pozície lakťa (**A**)



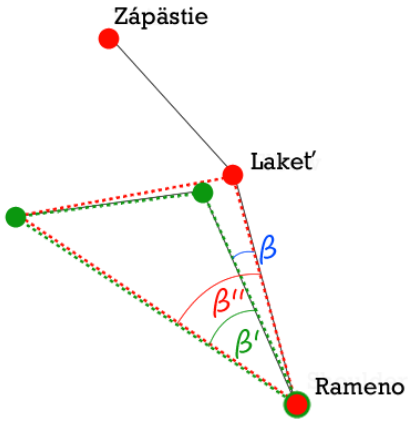
Obr. 8.3: Výpočet uhlov v trojuholníku

Po dosadení do vzorca

$$\beta' = \arccos(\frac{a^2 + c^2 - b^2}{2ac})$$

získame uhol β', ktorý sa rovná uhlu ramena vzhľadom na kľúč. Síce sme získali uhol vo výslednom trojuholníku, no nestačí ho len pripísať danej kosti, v tomto prípade ramenu. Aby sme zistili o koľko sa má rameno otočiť, je potrebné vypočítať ešte jeden trojuholník, začiatočný. V tomto prípade len nahradíme dĺžku **b** za vzdialenosť medzi pozíciami lakťa a kľúča. Týmto sme získali druhý uhol β'', vďaka ktorému po odčítaní uhlu β', získame uhol otočenia ramena.

$$\beta = \beta'' - \beta'$$



Obr. 8.4: Výpočet výsledného uhla ramena

Programu nezáleží na tom či je výsledný uhol menší alebo väčší ako nula. Po pričítaní mínusových hodnôt sa rameno otočí do opačného smeru. Tak získame uhol aj pre odklonenie sa do druhej strany ku cieľovému predmetu.

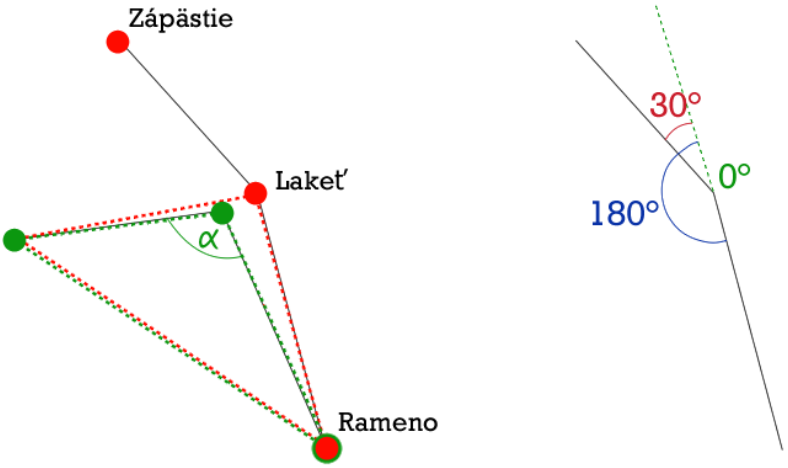
Po vypočítaní všetkých osí a aplikovaní na kosť ramena, by sme dostali výslednú pozíciu lakťa. V tomto konkrétnom výpočte sme neaplikovali os y a os x. Kosti majú, ako každý objekt v THREE.js, svoje vlastné osi, ktoré sa prispôsobujú predošlým kostiam v hierarchii. Y-ová os je potrebná pre presnú výslednú pozíciu, ale v tomto prípade sú to len malé zmeny, ktoré sa dajú pokryť otáčaním lakťa.

Na vypočítanie uhla kosti lakťa sme mali dve možnosti. Jedna bola z výsledného trojuholníka z predošlého výpočtu. Nebolo potrebné meniť dĺžky strán, stačilo len vypočítať odlišný uhol ako pri rotácii ramena. Museli sme si ale uvedomiť, že výsledný uhol, nie je rovný uhlu lakťa. Ako je zobrazené na obrázku 8.5, je potrebné tento uhol odčítať od uhlu 180° aby sme získali žiadaný uhol.

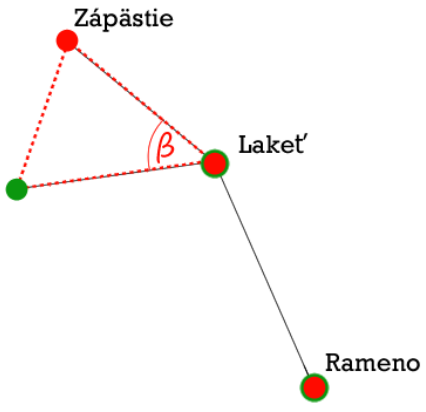
$$\alpha = \arccos(\frac{a^2 + b^2 - c^2}{2ab})$$

Alebo druhá možnosť bola vypočítať uhol, ktorý zviera lakeť vzhľadom na kľúč. Keďže sme pri počítaní uhla ramena zvolili len počítanie hlavnej osi otáčania, v oboch prípadoch je potrebné urobiť výpočet na dvoch osiach.

Stačilo nám vypočítať jeden uhol, ktorý pripíšeme k uhlu, ktorý už lakeť zviera. Po vypočítaní oboch osí získame zápästie priamo na pozícii cieľového objektu, čož ale nebol náš úplný zámer. Je potrebné si predstaviť ako sa pohybuje ruka v reálnom svete. Nejde rovno na pozíciu daného objektu. Ostane vedľa objektu



Obr. 8.5: Výpočet výsledného uhla lakťa



Obr. 8.6: Výpočet výsledného uhla lakťa

a chytí ho. V prípade animácie kľúča sme sa rozhodli upraviť pozíciu tak, aby dlaň skončila mierne nad objektom a aby palec a ukazovák ostali v Y osi sveta priamo nad objektom, keďže kľúč sa zdvíha medzi týmito dvoma prstami. Na začiatok sme prerobili výpočty, aby sa nepočítali vzhľadom na lakeť, ale na ukazovák. Ostatné parametre výpočtov mohli ostať nezmenené. Ďalej bolo potrebné trochu zmenšiť uhol na osi Y, ktorý bol vypočítaný. Takto sa zápästie neposunulo až úplne dole k objektu, ale ostalo nad ním.

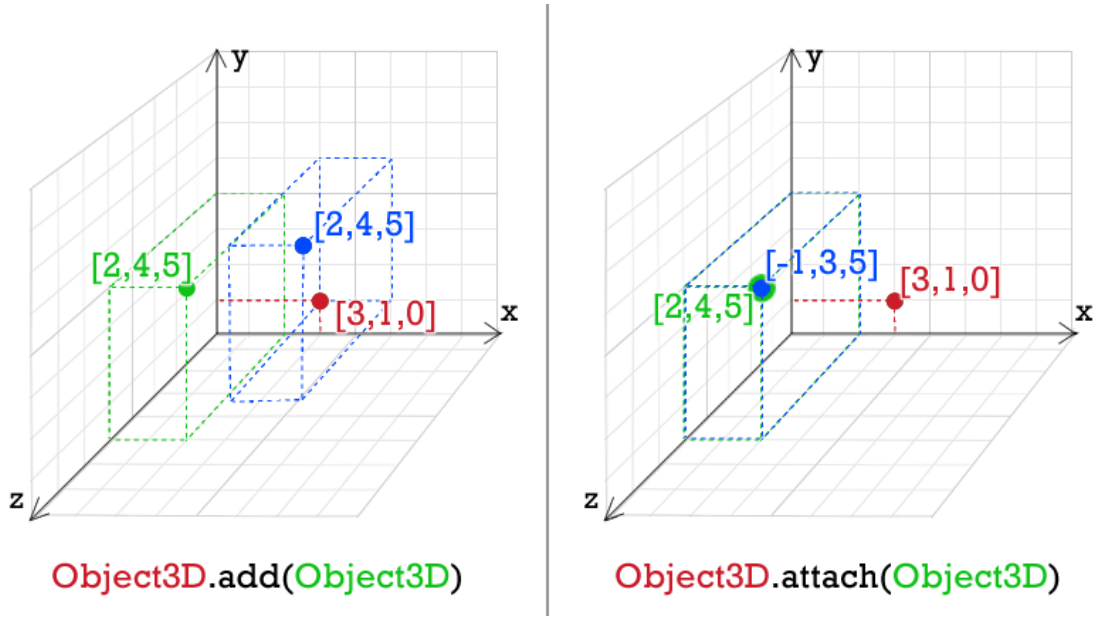
8.1.2 Zodvihnutie cieľového objektu

Rotácia prstov je daná staticky do pozície zdvihnutia kľúča. V tejto časti sme sa hlavne venovali rotáciám a pozíciám kľúča. Daný kľúč sme pridali ako potomka ku palcu. To znamená, že svet daného potomka sa odvíja od rodiča. Ak sa pohne rodič, zároveň s ním sa pohne aj potomok. Aby sa nemuseli neustále meniť pozície všetkých potomkov, ich pozícia a rotácia je počítaná od pozície rodiča ako od základu sveta daného objektu.

Priradiť potomka ku rodičovi sa dá 2 spôsobmi, ako môžeme vidieť aj na obrázku 8.7:

- *Object3D.add*(*Object3D*) – jeho lokálna pozícia sa nemení, preto sa posunie na pozíciu od nového rodiča
- *Object3D.attach*(*Object3D*) – jeho globálna pozícia sa nemení, lokálna sa vypočíta vzhľadom na nového rodiča, takže pri pohľade na túto zmenu sa nič neudeje, ale parametre objektu sa zmenia

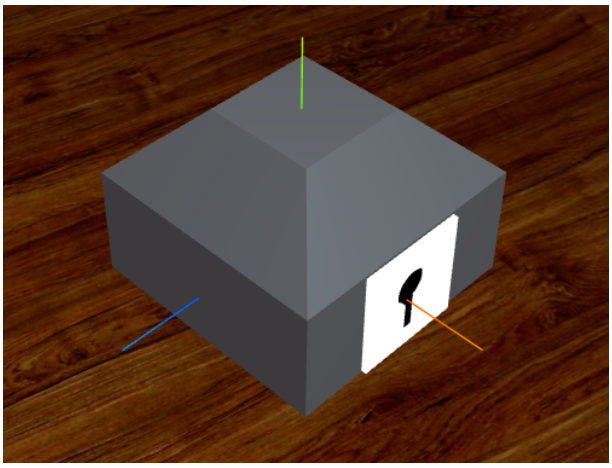
V tomto prípade nám stačilo použiť prvý prípad, pretože sme rotáciu a pozíciu počítali osobitne. Neskôr sme využívali aj druhý spôsob. Pozícia a rotácia kľúča boli vypočítané na miesto medzi prstami, ktoré ho zdvíhajú.



Obr. 8.7: Rozdiel medzi *add*() a *attach*() metódou

8.1.3 Presun ku zámke

Na báze predchádzajúcich metód sme vedeli presunúť ruku, už s kľúčom, ku zámke. Cieľový bod sme ale museli pozmeniť. Cieľom bolo, aby sa ruka zastavila pred schránkou tak, aby kľúč približne smeroval do zámky.



Obr. 8.8: Zobrazenie osí pomocou AxesHelper z THREE.js knižnice

Stačilo nám vytvoriť vektor, ktorý sa lokálne vzhľadom na schránku posunie dopredu na X-ovej osi. Následne pomocou funkcie *Object3D.localToWorld*() zmeniť lokálny vektor na globálny. Týmto sme dostali vektor na pozícii pred schránkou,

bez ohľadu na to ako je otočená.

Ruku sme síce dostali pred schránku, ale bolo potrebné ešte natočiť zápästie tak, aby kľúč smeroval do zámky. Robili sme to znova podľa výpočtov uhlov v trojuholníku. Pri týchto výpočtoch je potrebné si uvedomiť, kde sa nachádzajú dané vektory, to znamená, od ktorého objektu sa odvíjajú. Ak chceme zistiť bod zo schránky a následne ho počítať s bodmi z ruky, musíme ich konvertovať. Toto sme dosiahli použitím funkcií *localToworld*() a *worldToLocal*().

- *Object3D.localToWorld*(*Vector3*) zmení daný vektor z dimenzie objektu do dimenzie scény
- *Object3D.worldToLocal*(*Vector3*) zmení daný vektor z dimenzie scény do dimenzie objektu

Tieto funkcie posielajú ako návratovú hodnotu novú hodnotu vektora, no zároveň aj prepíšu daný vektor.

Pomocou týchto funkcií sme dokázali preniesť vektor zo schránky do scény a zo scény do ruky, kde už môžeme počítať. Nakoniec sme pridali aj kalkulácie uhlov pre samotný kľúč, aby to vyzeralo čo najdôveryhodnejšie.

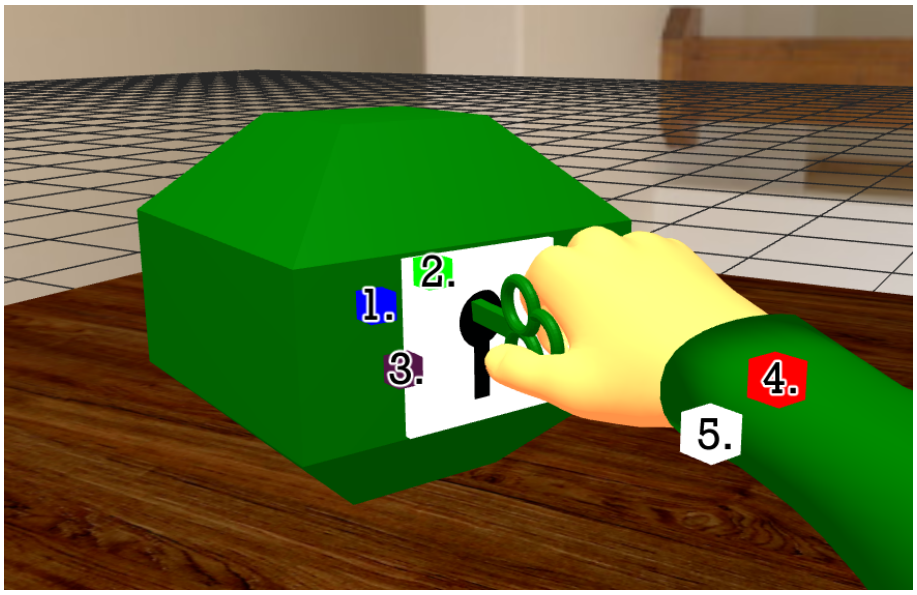
8.1.4 Presun do zámky

Vďaka prípravám z predošlej kapitoly bol tento krok jednoduchší. Jediné, čo bolo potrebné vykonať, je zmeniť cieľový vektor. Pred tým bol mierený pred schránku, tentokrát do vnútra schránky. A stačilo len zopakovať postup.

8.1.5 Otočenie v zámke

Ako sme na začiatku kapitoly spomenuli, tieto algoritmy slúžia len na výpočty koncových pozícií v jednotlivých etapách animácie. Otočenie v zámke ale nie je ten prípad. Je to jediná časť programu, kde sa nedá len vykalkulovať cieľovú rotáciu a riadiť sa podľa hodnôt. Príčinou je ťažisko zápästia, okolo ktorého sa objekt otáča. Našim cieľom bolo otočiť zápästie, ale nie vzhľadom na svoje ťažisko, ale okolo zámky.

Na obrázku 8.9 môžeme vidieť princíp otáčania okolo iného ťažiska ako je ťažisko daného objektu, pomocou zobrazených kociek:



Obr. 8.9: Otočenie podľa zadaného ťažiska

1. ťažisko schránky
2. ťažisko zápästia prenesené do lokálneho priestoru schránky s vynulovanou X-ovou osou
3. cieľová pozícia 2. kocky
4. pozícia 2. kocky prenesená do lokálneho priestoru zápästia s vynulovanou X-ovou osou
5. ťažisko zápästia pred posunom zápästia na pozíciu 4. kocky

Pričom 2. a 3. kocka sú potomkami 1. kocky a 4. kocka je potomkom 5. kocky. Pre tento algoritmus je dôležité ktorá kocka je ktorej potomkom. Z prvých troch kociek vieme vypočítať uhol o ktorý sa má ruka otočiť okolo zámky.

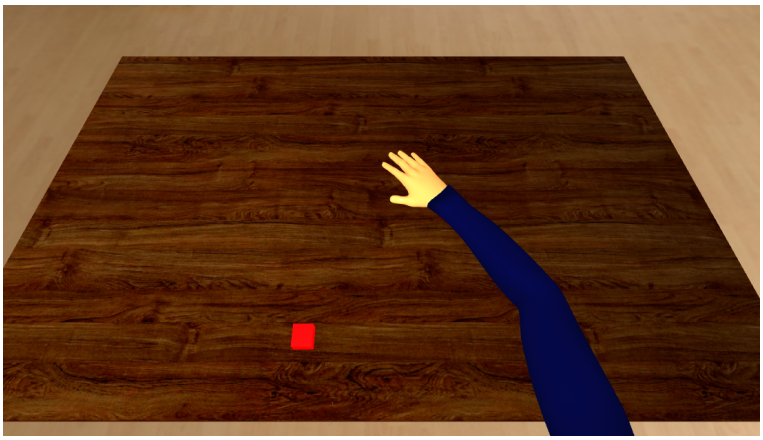
Podstata tejto rotácie okolo iného ťažiska je pomerne jednoduchá. Zároveň s otáčaním zápästia na Y-ovej osi, budemem otáčať aj kocky 1 a 5 pod rovnakým uhlom. Spolu s nimi sa budú otáčať aj ich potomkovia. Poloha zápästia sa pri každej zmene presunie na novú polohu 4. kocky, pomocou už spomínanej funkcie, ktorá rotáciou lakťa dostane zápästie na žiadané miesto. Keďže kľúč je ešte stále potomkom zápästia, bude sa točiť spolu s ním. No pre reálnejšiu animáciu sme pre istotu pridali aj rotáciu kľúča do stredu schránky.

8.1.6 Pustenie kľúča a vrátenie ruky na počiatočnú pozíciu

Odobratie kľúča ako potomka ruky sme dosiahli už spomínanou funkciou *Object3D.attach*(*Object3D*). Tu sme žiadne výpočty pre pohyb ruky nepotrebovali, pretože sa len vráti do stavu, ktorý už vypočítaný máme.

8.2 Algoritmus animácie kocky

Ruka má za úlohu zobrať kocku všetkými prstami a presunúť ju na iné miesto. Program využíva funkcie z predošlej kapitoly s drobnými zmenami.



Obr. 8.10: Animácia s kockou

Ako prvé je potrebné, aby sa ruka dostala ku kocke. Ruka by mala ostať nad predmetom, ako pri animácii s kľúčom, no na rozdiel od nej tento krát ruka zastane dlaňou priamo nad kockou.

Pri chytení kocky sa kocka nepremiestňuje, ani neotáča. Po chytení sme kocku presunuli pod nového rodiča, ruku, pomocou funkcie *Object3D.attach*(*Object3D*).

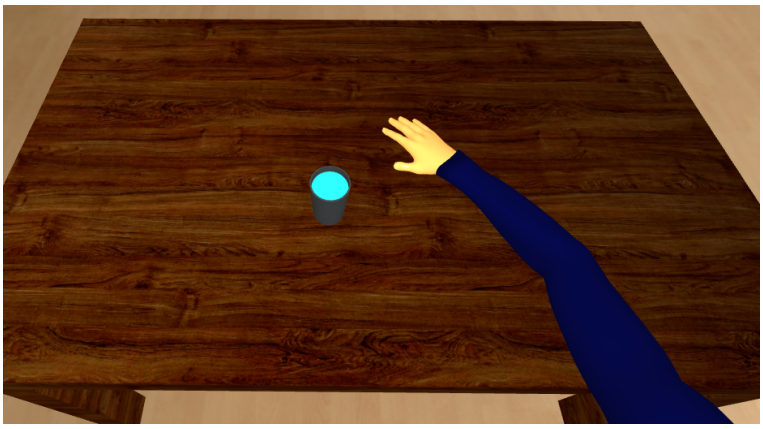
Ruka sa spolu s kockou presunie na určité miesto pomocou výpočtov uhlov v 2D trojuholníku a priblíži sa ku stolu.

Kocka, ako potomok, sa presunie z ruky na scénu pomocou už spomínanej funkcie *Object3D.attach*(*Object3D*), takže sme nemuseli riešiť novú pozíciu alebo rotáciu vzhľadom na scénu.

Následne sa už len ruka vrátila do pôvodnej pozície.

8.3 Algoritmus animácie pohára

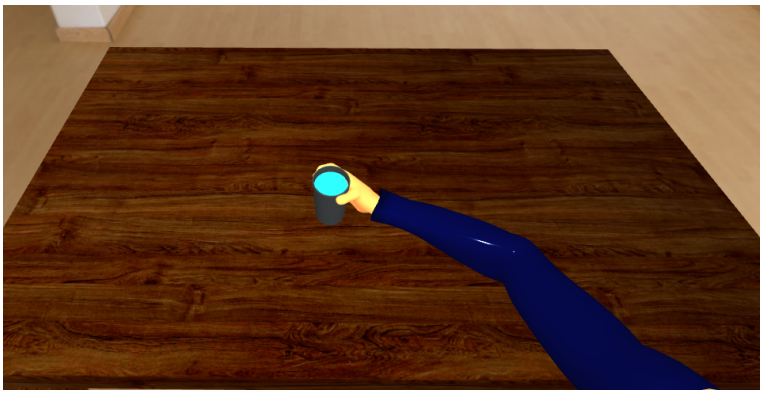
Ruka má za úlohu pohár len chytiť a vrátiť sa na pôvodnú pozíciu. Zo všetkých troch variácií je animácia s pohárom najjednoduchšia vzhľadom na problematiku pohybu.



Obr. 8.11: Animácia s pohárom

Aj táto animácia využíva funkcie na vyrátanie uhlov, aby sa zápästie dostalo ku poháru, no tento krát sme to museli zmeniť na pozíciu pred objektom spolu s rotáciou zápästia, aby bolo pripravené uchopiť pohár.

V tomto prípade sme nepotrebovali riešiť zmenu rodiča alebo rotáciu a polohu cieľa.



Obr. 8.12: Ruka drží pohár

8.4 Pohyb animácií

Ako sme spomínali v kapitole 7.3, základom pohybu je *tick*() funkcia v komponente. Po získaní koncových rotácií a pozícií nám už len stačilo docieliť, aby sa kosti a predmety dostali na cieľové súradnice.

V systéme sú len dva typy objektov, ktoré menia rotáciu alebo pozíciu. Kľúč mení obe tieto vlastnosti, zatiaľ čo kosti ruky len rotáciu. Keďže máme vypočítanú začiatočnú a koncovú pozíciu objektu, je jednoduché si zistiť, o koľko sa má objekt otočiť, resp. posunúť, za daný čas.

Tick() funkcia sa spúšťa pri vykresľovaní každej snímky, takže rýchlosť animácie závisí od rýchlosti vykresľovania zariadenia, v ktorom je program spustený. Aby sme predišli príliš rýchlej alebo príliš pomalej animácii, zvolili sme spôsob, kde si zistíme počet snímkov za sekundu a z výsledku sa prepočíta, koľko snímkov bude animácia trvať. Takýmto spôsobom sme zabezpečili, aby animácia trvala rovnako dlhý čas, bez ohľadu na zariadenie.

Na zistenie počtu snímkov za sekundu sme použili triedu stats.js. Na začiatok funkcie *tick*() sme pridali nasledujúci kód

```
tick: function() {
    stats.begin();
    stats.end();
}
```

pričom funkcia *end*() nám vráti výslednú hodnotu snímkov za sekundu.

Funkcia, ktorá má na starosti otáčanie a posúvanie objektov, vypočítava zo súradníc uhol alebo posun na každej osi zvlášť.

$$\frac{startedAngle - endedAngle}{fps}$$

kde

- startedAngle je začiatočná rotácia na jednej osi
- endedAngle je koncová rotácia na rovnakej osi
- fps je počet snímkov za sekundu.

Následne kontrolujeme či aktuálna rotácia už dosiahla cielené stanovisko. Ak nie, hodnota sa pripočíta alebo odčíta, podľa potreby. Ak sa objekt už nemá posúvať v žiadnom smere, funkcia vráti pravdivostnú hodnotu, aby program vedel, že daný pohyb už skončil.

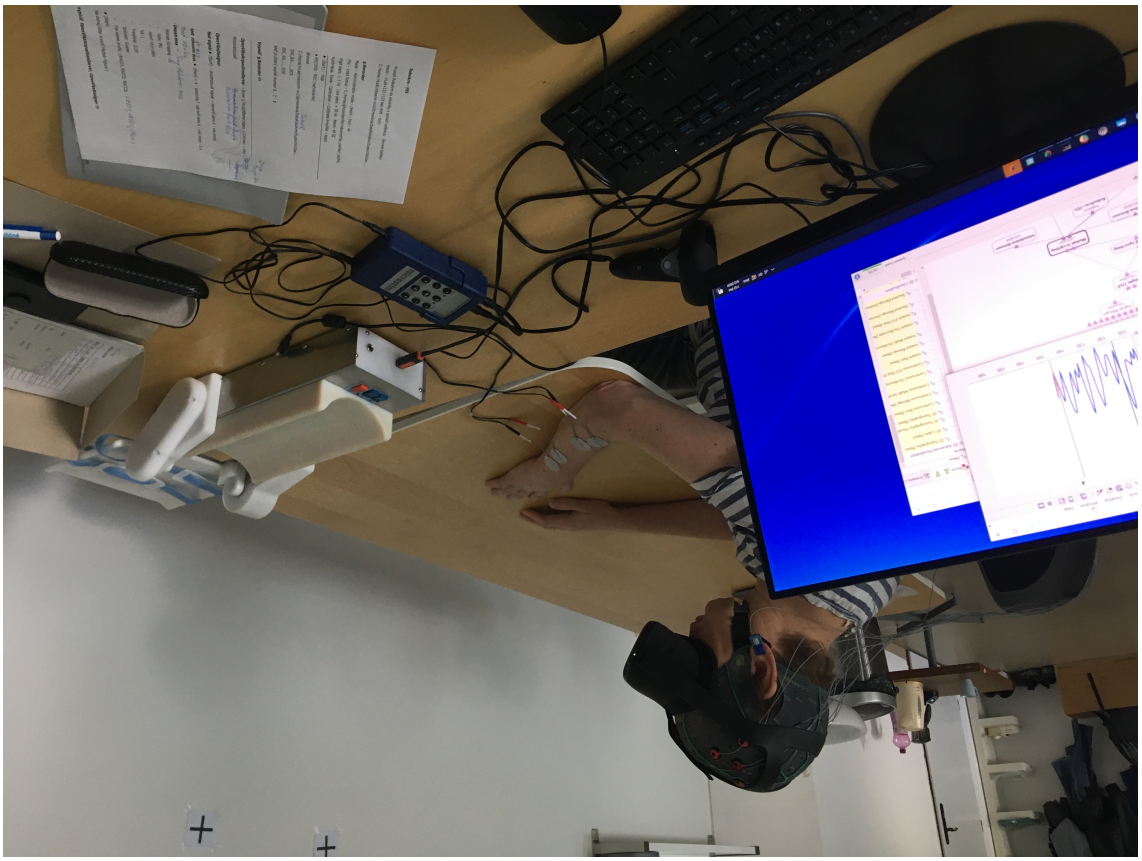
Každá animácia má svoje vlastné štádia, ktoré spolu dávajú celú animáciu. Ako príklad si ukážeme animáciu s kľúčmi.

- WAIT – čaká na signál na spustenie animácie
- GETTOKEY – posun ruky ku kľúču
- PICKUPKEY – zodvihnutie kľúča a zmena rodičov daného kľúča
- GETTOLOCK – posun ruky s kľúčom ku zámke
- GETINTOLOCK – posun ruky do zámky
- ROTATE – otočenie ruky v zámke a následne zmena rodiča kľúča
- RETURN – vrátenie ruky na pôvodnú pozíciu

Keď jedna časť pohybu skončila, pokračuje sa na ďalšej. Keď sa skočí celá animácia, funkcia vráti pravdivostnú hodnotu do *tick*() funkcie, ktorá vyhodnotí pokračovanie programu. Ak sa práve skončila animácia kópie ruky, program sa dostane do stavu čakania na správu o zapnutí animácia s originálnymi objektami. V opačnom prípade sa všetky predmety schovajú a pošle sa správa do rozhrania doktora, že animácia skončila a sekvencia sa môže prepnúť do nasledujúceho stavu.

9 Testovanie systému

Systém bol testovaný už počas implementácie s už dokončenými časťami programu. Zo začiatku boli animácie exportované z prostredia Blender. Tieto animácie zahŕňali všetky tri druhy pohybu, ako aj na konci implementácie, úchop pohára, presun kocky na iné miesto a úchop kľúča a otočenie ho v zámke. No posledná animácia s kľúčom obsahovala len jeden kľúč a jednu schránku. Po pár experimentoch sa požiadavka zvýšila na tri kľúče a tri schránky. Ak by sme to chceli aj naďalej animovať v Blenderi, museli by sme urobiť každú kombináciu osobitne, čo by znamenalo urobiť z jednej animácie, deväť skoro rovnakých. Preto sme sa rozhodli robiť pohyb priamo v systéme inverznou kinematikou. Aj napriek zníženiu kvality pohybu objektov, tento krok uľahčil dotváranie animácií v budúcnosti. Animácie nie je potrebné vytvárať samostatne a následne ich importovať do programu. Systém už má vytvorené funkcie na priblíženie sa ku objektom, potrebná je len minimálna zmena pri jednotlivých úchopoch. Navyše, animácie sú univerzálnejšie. Aj po posnutí objektu sa animácia uskutoční, ak je predmet v dosahu ruky. Spolupracujúci terapeuti si do budúcnosti predstavujú napríklad presun pešiakov po šachovnici. Pomocou už implementovaných metód by nemalo byť zložité túto animáciu pridať.



Obr. 9.1: Experiment z prostredia SAV

Kvôli aktuálnej spoločenskej situácií bol systém testovaný len zo začiatku a po prerobení animácií bol program kontrolovaný len terapeutmi. Pacient, ktorý bol zapojený do programu nemal povolené vychádzať zo svojho prostredia a tento experiment vyžaduje návštevy aspoň dva krát do týždňa. Systém je síce postavený na kolaboratívnom prostredí, no pacient potrebuje mať zariadenia u seba a potrebuje, aby mu s nimi niekto pomáhal, preto bolo výhodnejšie dochádzať do prostredia, kde sa experiment vykonával.

Spolu s pridaním objektov kľúčov a schránok, bola pridaná aj možnosť prepnúť animáciu na ďalšiu, ak by mal pacient problém s náročnosťou aktuálneho pohybu.

Ďalšia pridaná požiadavka bola na animácie s kópiami objektov, aby pacient videl, čo si má predstaviť. No daná implementácia môže pacienta aj zmiasť a preto by sme ešte do budúcnosti chceli dať možnosť vypnúť dané animácie.

Zo začiatku bol program nastavený na kolaboratívne prostredie, aby terapeut videl to, čo vidí aj pacient a mohol sa s ním dohovoriť, ak by v budúcnosti neboli v rovnakej miestnosti. Ku koncu implementačného procesu bola pridaná poži-

adavka na ovládanie experimentu aj z virtuálneho prostredia terapeuta. Aj keď požiadavka bola splnená, vzhľadom na krátkosť času na implementovanie je nevyhnutné aby bolo zapnuté aj prostredie s parametrami sekvencie, ktoré ovláda priebeh sekvencie.

Po zmiernení situácie je naplánované vykonanie experimentov s daným systémom spolu s dotváraním kolaboratívneho prostredia a väčších možností ovládania terapeutického procesu z virtuálneho rozhrania terapeuta.

10 Záver

V spolupráci so Slovenskou akadémiou vied sme implementovali ich požiadavky do virtuálneho prostredia. Všetky hlavné požiadavky sú úspešne nasadené v systéme. Terapeut dokáže nastaviť parametre sekvencie vo svojom rozhraní. Vie nastaviť koľko bude pohybov zahrnutých v sekvencii, spustiť ju a aj zastaviť. Vie nastaviť dva časové parametre, ktoré slúžia na dobu relaxu medzi scenármi a dĺžku samotného scenára. Program očakáva správu na spustenie animácie v scenári zo zariadenia Robo-arm alebo aj z rozhrania terapeuta.

Animácie sú implementované inverznou kinematikou, ktoré pracuje s pozíciami objektov a rotáciou kostí ruky. Takže je možné objekty presúvať podľa potreby, kým sú v dosahu ruky. Ak nie sú, animácia sa nevykoná. V systéme sú tri druhy animácií. Úchop pohára, presun kocky na inú pozíciu a zdvihnutie kľúča, vloženie do zámky a otočenie kľúčom v zámke. Pohár aj kocka sa v priestore nachádzajú práve raz, no kľúče so schránkami sú v trojici. V budúcnosti by mohlo pribudnúť viac objektov aj v ostatných animáciách, ako aj samotné animácie.

Okrem prostredia pre zadávanie parametrov, má terapeut aj druhú možnosť sledovať scénu. Môže sa pripojiť priamo do miestnosti, ktorá funguje na báze kolaboratívneho virtuálneho prostredia. Týmto spôsobom má možnosť priamo vidieť virtuálne prostredie ako aj pacient. Najväčšou výhodou kolaboratívneho prostredia je v tomto prípade komunikácia medzi terapeutom a pacientom. Aj keď sa táto funkcia aktuálne nepoužíva, v budúcnosti je v pláne uberať sa týmto smerom.

Pri ďalšej implementácii by sa systém dal rozvinúť pridaním ďalších objektov do scény, ako je aktuálne scéna s kľúčmi. Samozrejme by sa systém dal rozšíriť o rôzne ďalšie animácie, zamerané nie len na mobilitu horných končatín. Aktuálne sa v sekvencii môžu vyskytnúť všetky druhy scenárov, aj keď pohyb spojený s kľúčom by mohol byť na začiatok komplikovaný. Preto by sa do systému mohla

pridať možnosť výberu animácií, ktoré sa budú v sekvencii nachádzať. Alebo rovno aj navoliť poradie animácií.

Najväčším problém terapeutov je synchronizácia posielania signálov do ruky spolu s prehrávaním animácií. Riešenie by bolo, aby boli animácie rovnako dlhé. No komplikovanosť pohybu v scenári s kľúčom, je dôvod jej dlhšieho trvania oproti ostatným dvom animáciám. Vyriešiť by sa to dalo zrýchlením, respektíve spomalením ostatných animácií, no tým utrpia na dôveryhodnosti pohybu. Najlepšie by bolo aby rovnako komplikované animácie boli spolu v sekvencii. Takto by sa zvyšovala alebo znižovala náročnosť celej sekvencie. Ale toto riešenie by si vyžadovalo viac animácií ako má systém aktuálne na výber.

Aj keď vzhľadom na aktuálnu situáciu nedošlo k testovaniam výsledného produktu na pacientoch, tím doktorov zo SAV bol spokojný s implementovaným systémom. Po zlepšení situácie sa systém bude využívať na daných experimentoch a aj ďalej rozrastať.

Literatúra

[1] D. Snowdon E. F. Churchill. *Collaborative Virtual Environments: An Introductory Review of Issues and Systems*. URL: https://link.springer.com/content/pdf/10.1007/BF01409793.pdf. (accessed: 21.02.2021).

[2] Igor Farkaš. *Rozhranie mozog–počítač s adaptívnym robotickým ramenom na rehabilitáciu*. URL: https://fmph.uniba.sk/detail-novinky/browse/1/back_to_page/z-riesenych-vedeckych-projektov/article/rozhranie-mozog-pocitac-s-adaptivnym-robotickym-ramenom-na-rehabilitaciu-projekt-apvv/.

[3] GestureTek. *IREX IS GESTURETEK HEALTH'S FLAGSHIP PRODUCT FOR THE HEALTHCARE ARENA*. URL: http://www.gesturetekhealth.com/products/irex.

[4] Marián Hudák. *Laboratórium počítačovej grafiky - LIRKIS*. URL: https://kpi.fei.tuke.sk/sk/content/laboratorium-pocitacovej-grafiky-lirkis. (accessed: 08.03.2021).

[5] Marián Hudák. *O laboratóriu*. URL: http://lirkis.kpi.fei.tuke.sk/sk/o-laboratoriu/. (accessed: 08.03.2021).

[6] J A Gil-Gómez J A Lozano-Quilis H Gil-Gómez. *Virtual Reality System for Multiple Sclerosis Rehabilitation using KINECT*. URL: http://rehab-workshop.org/2013/papers/136696042743759.pdf.

[7] Kelvw. *3D Connexion SpacePilot revisited*. URL: https://kelvinreview.wordpress.com/2012/07/19/3d-connexion-spacepilot-revisited/. (accessed: 28.03.2021).

[8] Branislav Sobota Marián Hudák Štefan Korečko. *Enhancing Team Interaction and Cross-platform Access in Web-based Collaborative Virtual Environments*. URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9119312. (accessed: 06.03.2021).

[9] Medel. *FUNCTIONAL ELECTRICAL STIMULATION*. URL: https://www.medel-hamburg.de/electrical-stimulation/functional-electrical-stimulation/.

[10] Mozilla. *Welcome to Hubs*. URL: https://hubs.mozilla.com/docs/welcome.html.

[11] MSc OÄ Dr. Andrea Kulisev. *Polyneuropathy*. URL: https://www.klinik-pirawarth.at/en/diseases/polyneuropathy/.

[12] Clinica orthopedica. *Čo je rehabilitácia a prečo je pre pacienta dôležitá?* URL: https://clinicaorthopedica.sk/co-je-rehabilitacia-a-preco-je-pre-pacienta-dolezita.

[13] Roman Rosipal. *Brain-computer interface with robot-assisted training for rehabilitation (BCI-RAS)*. URL: http://aiolos.um.savba.sk/~roman/rrLab/projects.html.

[14] MDN contributors. "WebGL: 2D and 3D graphics for the web". In: *Mozilla developer* (2021). DOI: https://developer.mozilla.org/en-US/docs/Web/API/WebGL_API.

[15] MDN contributors. "WebXR Device API". In: *Mozilla developer* (2021). DOI: https://developer.mozilla.org/en-US/docs/Web/API/WebXR_Device_API.

[16] Hardoňová Petra. "Využitie datovej prilby v procese terapie". MA thesis. Technická univerzita v Košiciach, 2020.

[17] Felix Rudert Sado Rabaudo Daniel Tinkham. "VR for remote collaboration". In: *WorldWiz* (2020). DOI: https://www.worldviz.com/post/vr-for-remote-collaboration.

[18] Redakcia TouchIT. "Slovenský projekt vytvoril kolaboratívne virtuálne prostredia: nová dimenzia vzdelávania, podnikania a spolupráce vo virtuálnej realite". In: *TouchIT* (2020). DOI: https://touchit.sk/slovensky-projekt-vytvoril-kolaborativne-virtualne-prostredia-nova-dimenzia-vzdelavania-podnikania-a-spoluprace-vo-virtualnej-realite/306978.

[19] Dušan Valent. "Rýchlejšie zotavenie pri domácej rehabilitácii? Pomôžu VR fyzioterapeuti". In: *Zive* (2020). DOI: https://zive.aktuality.sk/clanok/145689/rychlejsie-zotavenie-pri-domacej-rehabilitacii-pomozu-vr-fyzioterapeuti/.

[20] Or Goldfus. "Building AR/VR with Javascript and HTML". In: *Hackernoon* (2018). DOI: https://hackernoon.com/building-ar-vr-with-javascript-and-html-28acd1da0371?gi=646fa127cedb.

[21] Ing. František Hrozek PhD. doc. Ing. Branislav Sobota PhD. "Systémy virtuálnej reality". MA thesis. Technická univerzita v Košiciach, 2015.

[22] Petr Vašíček. "Rukavice Hands Omni umožní hráčům cítit virtuální objekty". In: *CDR* (2015). DOI: https://cdr.cz/clanek/rukavice-hands-omni-umozni-hracum-citit-virtualni-objekty.

[23] Ing. Miloš Liba. "VIRTUÁLNA REALITA A JEJ VYUŽITIE PRI RIEŠENÍ VYBRANÝCH ÚLOH". In: *Transfer inovácií* (2008). DOI: https://www.sjf.tuke.sk/transferinovacii/pages/archiv/transfer/12-2008/pdf/155-158.pdf.

[24] CSc. Ing. Jana Mihalíková doc. Ing. Ondrej Líška. "VYUŽITIE VIRTUÁLNEJ REALITY VO VZDELÁVACOM PROCESE". In: *Transfer inovácií* (2006). DOI: http://www.sjf.tuke.sk/transferinovacii/pages/archiv/transfer/9-2006/pdf/83-85.pdf.