

**Technická univerzita v Košiciach
Fakulta elektrotechniky a informatiky**

**Integrácia ovládania snímaním rúk do
systému virtuálnych výučbových prostredí**

Bakalárska práca

**Technická univerzita v Košiciach
Fakulta elektrotechniky a informatiky**

**Integrácia ovládania snímaním rúk do
systému virtuálnych výučbových prostredí**

Bakalárska práca

Študijný program:	Informatika
Študijný odbor:	Informatika
Školiteľ:	doc. Ing. Štefan Korečko PhD.
Školiace pracovisko:	Katedra počítačov a informatiky (KPI)

Košice 2026

Mykola Martyshchuk

Abstrakt v slovenčine

Táto bakalárska práca sa zaoberá rozšírením systému virtuálnych výučbových prostredí LirkisEduVePn o ovládanie pomocou snímania rúk, ktoré nahrádza tradičné ovládače virtuálnej reality. Predchádzala tomu analýza existujúcich riešení gestového ovládania v prostredí A-Frame, vrátane natívneho sledovania rúk cez WebXR Hand Tracking API, knižnice Super Hands a vlastných prístupov. Na základe analýzy bola navrhnutá a implementovaná modulárna architektúra pozostávajúca z centrálného komponentu hand-tracking-bridge a piatich špecializovaných interakčných modulov pokrývajúcich uchopenie objektov, fyzikálnu prítomnosť ruky v scéne, dotykovú interakciu, teleportáciu a interakciu na diaľku. Riešenie je integrované do existujúcej architektúry projektu a zachováva kompatibilitu s riadením scenárov pomocou Petriho sietí aj s fyzickými ovládačmi. Funkčnosť systému bola overená používateľským testovaním.

Kľúčové slová v slovenčine

virtuálna realita, sledovanie rúk, gestové ovládanie, výučbové prostredia, WebXR, A-Frame, Petriho siete

Abstrakt v angličtine

This bachelor's thesis focuses on extending the LirkisEduVePn virtual educational environment system with hand tracking control, replacing traditional virtual reality controllers. The work was preceded by an analysis of existing gesture control solutions in the A-Frame rámec, including native hand tracking via the WebXR Hand Tracking API, the Super Hands library and custom approaches. Based on the analysis, a modular architecture was designed and implemented, consisting of a central hand-tracking-bridge component and five specialized interaction modules covering object grabbing, physical hand presence in the scene, fingertip touch interaction, teleportation and remote interaction. The solution is integrated into the existing project architecture and maintains compatibility with Petri net-based scenario control as well as physical controllers. The functionality of the system was verified through user testing.

Kľúčové slová v angličtine

virtual reality, hand tracking, gesture control, educational environments, WebXR, A-Frame, Petri nets

Bibliografická citácia

MARTYSHCHUK, Mykola. *Integrácia ovládania snímaním rúk do systému virtuálnych výučbových prostredí*. Košice: Technická univerzita v Košiciach, Fakulta elektrotechniky a informatiky, 2026. 44 s. Vedúci práce: doc. Ing. Štefan Korečko PhD.

TECHNICKÁ UNIVERZITA V KOŠICIACH
FAKULTA ELEKTROTECHNIKY A INFORMATIKY
Katedra počítačov a informatiky

ZADANIE
BAKALÁRSKEJ PRÁCE

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

Názov práce:

Integrácia ovládania snímaním rúk do systému virtuálnych výučbových prostredí

Hand Tracking Control Integration to Virtual Educational Environment System

Študent: **Mykola Martyshchuk**

Školiteľ: **doc. Ing. Štefan Korečko, PhD.**

Školiace pracovisko: **Katedra počítačov a informatiky**

Konzultant práce:

Pracovisko konzultanta:

Pokyny na vypracovanie bakalárskej práce:

1. Oboznámiť sa so systémom virtuálnych výučbových prostredí so scenármi vo forme Petriho sietí, vytvorenom na domovskom pracovisku.
2. Analyzovať existujúce riešenia ovládania snímaním rúk vhodné pre systém virtuálnych výučbových prostredí.
3. Na základe analýzy navrhnúť a implementovať komponent ovládania snímaním rúk a integrovať ho do existujúcich virtuálnych výučbových prostredí v systéme.
4. Návrhové a implementačné práce koordinovať s ďalšími riešiteľmi prác na systéme virtuálnych výučbových prostredí.
5. Vypracovať dokumentáciu podľa pokynov vedúceho práce.

Jazyk, v ktorom sa práca vypracuje: slovenský
Termín pre odovzdanie práce: 29.05.2026
Dátum zadania bakalárskej práce: 31.10.2025




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

Čestné vyhlásenie

Vyhlasujem, že som záverečnú prácu vypracoval samostatne s použitím uvedenej odbornej literatúry, ako aj s využitím nástrojov generatívnej umelej inteligencie v súlade s Metodickým pokynom o záverečných a kvalifikačných prácach a Etickým kódexom študenta Technickej univerzity v Košiciach. Umelá inteligencia bola použitá ako podporný nástroj bez porušenia zásad akademickej etiky a autorskej integrity.

Podakovanie

Na tomto mieste by som sa rád poďakoval svojmu vedúcemu práce doc. Ing. Štefanovi Korečkovi, PhD. za jeho čas, cenné rady a vopred navrhnutý harmonogram konzultácií, ktoré mi výrazne pomohli počas riešenia mojej záverečnej práce. Ďakujem tiež Bc. Michalovi Sakáčovi za jeho čas a ochotu zdieľať skúsenosti s projektom, ako aj Milanovi Halčinovi za spoluprácu.

Rovnako by som sa rád poďakoval svojim rodičom a priateľom za ich podporu a povzbudzovanie počas celého môjho štúdia. Osobitnú vďaku vyslovujem aj účastníkom testovania systému za ich čas a ochotu zúčastniť sa používateľského testovania.

Obsah

Úvod	1
1 Analýza	3
1.1 Analýza predchádzajúcich riešení	3
1.1.1 Počiatočný vývoj systému	3
1.1.2 Rozšírenie o klient-server architektúru	4
1.1.3 Stabilizácia a nasadenie systému	4
1.1.4 Rozšírenie o nové výučbové scenáre	4
1.1.5 Optimalizácia a refaktORIZÁCIA systému	5
1.1.6 Rozšírenie interaktívnych komponentov	5
1.1.7 Zhrnutie a identifikácia obmedzení	5
1.2 Analýza možností gestového ovládania	6
1.2.1 Natívne sledovanie rúk cez A-Frame	6
1.2.2 Použitie Superhands pre sledovanie rúk	8
1.2.3 Hybridný režim: kombinácia VR ovládača a sledovania rúk	8
1.2.4 Vlastný hand-grab systém	9
1.2.5 Použitie externých knižníc	10
1.3 Integrácia gest s procesným riadením prostredia	10
1.3.1 Problémy spôsobené integráciou	10
1.3.2 Prepojenie so simuláciou Petriho siete	12
1.3.3 Komponenty v konflikte	13
1.4 Záver analýzy	13
2 Návrh riešenia	15
2.1 Východiskový problém a cieľ návrhu	15
2.2 Kľúčové požiadavky	16
2.3 Architektúra riešenia	16
2.3.1 Vstupná vrstva	17
2.3.2 Uzlový komponent	17

2.3.3	Interakčné moduly	17
2.3.4	Externé systémy	18
2.4	Štruktúra komponentov	18
3	Implementácia	20
3.1	Komponent hand-tracking-bridge	20
3.1.1	Štruktúra komponentu	20
3.1.2	Inicializácia komponentu	21
3.1.3	Verejné rozhranie	21
3.1.4	Aktualizácia stavu a distribúcia udalostí	21
3.2	Komponent hand-grab	22
3.2.1	Schéma komponentu	22
3.2.2	Začiatok uchopenia	23
3.2.3	Manipulácia počas držania	24
3.2.4	Pustenie objektu a hádzanie	25
3.3	Komponent hand-collision-bodies	25
3.3.1	Inicializácia a tvorba kolíznych telies	26
3.3.2	Synchronizácia a ghosting	26
3.4	Komponent hand-point-fingertip-touch	27
3.5	Komponent hand-teleport	28
3.5.1	Aktivácia a deaktivácia teleportačného režimu	29
3.5.2	Aktivácia oblúka a potvrdenie teleportácie	29
3.6	Komponent hand-point-shoot	30
3.6.1	Sledovanie cieľa lúčom	31
3.6.2	Pritiahnutie objektu a odovzdanie riadenia	32
3.7	Zhrnutie implementácie	33
4	Vyhodnotenie	34
4.1	Metóda vyhodnotenia	34
4.2	Priebeh testovania	35
4.3	Výsledky a interpretácia	36
4.3.1	System Usability Scale (SUS)	36
4.3.2	NASA Task Load Index (NASA TLX)	37
4.3.3	Virtual Reality Sickness Questionnaire (VRSQ)	38
4.3.4	Igroup Presence Questionnaire (IPQ)	38
4.3.5	Hodnotenie gestovej interakcie	39
4.4	Zhrnutie vyhodnotenia	40

5 Záver	41
Zoznam použitej literatúry	43
Zoznam skratiek	45
Zoznam príloh	46
A Používateľská príručka	47
A.1 Funkcia programu	47
A.2 Požiadavky a spustenie	47
A.2.1 Požiadavky na technické prostriedky	48
A.2.2 Spustenie projektu lokálne	48
A.2.3 Prvé prihlásenie a konfigurácia	51
A.3 Použitie programu	52
A.3.1 Prihlasovacia obrazovka	52
A.3.2 Hlavná obrazovka	52
A.3.3 Administračný panel pre učiteľov a administrátorov	53
A.4 Použitie ovládania rukami	54
A.4.1 Prehľad gest	54
A.4.2 Uchopenie a manipulácia s objektmi	55
A.4.3 Dotyková interakcia s klávesnicou a tlačidlami	56
A.4.4 Otváranie dverí	56
A.4.5 Teleportácia	57
A.4.6 Interakcia na diaľku	57
B Systémová príručka	59
B.1 Funkcia programu	59
B.2 Popis riešenia	59
B.3 Štruktúra súborov	60
B.4 Popis implementovaných A-Frame komponentov	60
B.4.1 Komponent hand-tracking-bridge	60
B.4.2 Komponent hand-grab	62
B.4.3 Komponent hand-collision-bodies	63
B.4.4 Komponent hand-point-fingertip-touch	64
B.4.5 Komponent hand-teleport	65
B.4.6 Komponent hand-point-shoot	66
B.5 Integrácia komponentov v scéne	67
B.6 Integrácia s existujúcimi komponentmi systému	68

B.7	Využitie technológié	68
B.8	Spustenie a nasadenie	69

Zoznam obrázkov

1.1	Schéma 25 kĺbov ruky podľa špecifikácie WebXR Hand Input Module	7
2.1	Komponentový diagram navrhovaného systému gestového ovládania	19
3.1	Sekvenčný diagram inicializácie komponentu hBridge	22
3.2	Sekvenčný diagram aktualizácie stavu hBridge a distribúcie udalostí	23
3.3	Sekvenčný diagram začiatku uchopenia objektu	24
3.4	Sekvenčný diagram manipulácie s objektom počas držania	25
3.5	Sekvenčný diagram pustenía objektu a aplikácie hodu	26
3.6	Sekvenčný diagram inicializácie komponentu hCollision a tvorby kolíznych telies	27
3.7	Sekvenčný diagram synchronizácie kolíznych telies a prepínania režimu ghosting	28
3.8	Sekvenčný diagram dotykovej interakcie špičkou ukazováka	29
3.9	Sekvenčný diagram aktivácie a deaktivácie teleportačného režimu . .	30
3.10	Sekvenčný diagram aktivácie teleportačného oblúka a potvrdenia teleportácie	31
3.11	Sekvenčný diagram sledovania cieľa lúčom	32
3.12	Sekvenčný diagram pritiahnutia objektu a odovzdania riadenia komponentu hGrab	33
A.1	Prihlasovacia obrazovka	52
A.2	Hlavná obrazovka žiaka	53
A.3	Výber úlohy	53
A.4	Administračný panel pre učiteľov a administrátorov	54
A.5	Prehľad histórie úloh pre skupinu	54
A.6	Štatistiky úlohy jednotlivca	55
A.7	Uchopenie a manipulácia s objektom snímaním rúk	56
A.8	Dvere s tlačidlom – otvorenie stlačením tlačidla gestom štípnutia . . .	56
A.9	Dvere s páčkou – otvorenie potiahnutím páčky nadol	57
A.10	Tajné dvere – zadanie hesla pomocou virtuálnej klávesnice	57
A.11	Teleportácia ovládaná gestami ruky – zobrazenie oblúka smeru	58

A.12 Gesto ukazovania – natiahnutý ukazovák určuje smer lúča	58
A.13 Interakcia na diaľku – modrý lúč znamená, že v jeho smere nebol naj- dený žiadny uchopiteľný objekt, žltý lúč signalizuje zameraný objekt, ktorý možno pritiahnuť	58

Zoznam tabuliek

4.1	Predchádzajúce skúsenosti účastníkov	36
4.2	Výsledky dotazníka SUS	37
4.3	Výsledky dotazníka NASA TLX	37
4.4	Výsledky dotazníka VRSQ	38
4.5	Súhrnné výsledky dotazníka IPQ	39
4.6	Hodnotenie gestovej interakcie	39
A.1	Základné gestá ovládania rukami	55

Zoznam výpisov

A.1	Príklad súboru docker-compose.yaml	48
A.2	Voliteľný súbor servers.json pre predkonfiguráciu pgAdmin	50
A.3	SQL dotaz na pridanie administrátorského účtu	51
B.1	Konfigurácia hand-tracking entít v scéne	67

Úvod

Technológie virtuálnej reality (Virtual Reality, **VR**) sú každým rokom dostupnejšie a integrujú sa do všetkých sfér života. Podľa výskumov z posledných rokov [1, 2] môže **VR** zvýšiť záujem a angažovanosť študentov vo vzdelávacom procese, ktorý poskytuje interaktívne prostredie, kde sa abstraktné koncepty stávajú hmatateľnými.

Aplikácie založené na **VR** umožňujú umiestniť študentov do prostredia, v ktorom by sa v reálnom svete nemohli nachádzať, alebo kde by ich skutočná prítomnosť bola pre nich riziková, ako napríklad historické udalosti, vedecké experimenty alebo technické zručnosti. Takýto systém vyučovania motivuje študentov k vykonávaniu praktických úloh a podporuje kolaboratívne učenie v spoločných virtuálnych priestoroch [3], kde sa študent môže súčasne radiť s učiteľom.

Pre lepšie osvojenie si učiva a väčšie zapojenie študenta do vzdelávacieho procesu môžu byť tradičné metódy ovládania, ako sú fyzické ovládače, obmedzujúce, najmä v školskom prostredí, kde je kľúčová dostupnosť zariadení. Ovládanie gestami rúk založené na detekcii pohybov ponúka prirodzený a pre študenta optimálny prístup k interakcii a pocitu prítomnosti vo virtuálnom svete [4]. Tento typ ovládania sa už používa v hernom priemysle a simuláciách na vyučovania odborníkov, pre ktorých je kľúčová presnejšia manipulácia s objektmi [5].

Táto bakalárska práca nadväzuje na sériu prác projektu LirkisEduVePn vyvinutého na Katedre počítačov a informatiky Technickej univerzity v Košiciach. Nadväzuje na diplomovú prácu Ing. Adama Kašelu [6], ktorá položila základy systému vzdelávacích aplikácií vo virtuálnej realite na báze procesných grafov typu Petriho siete [7], a tiež na bakalárske práce Bc. Dmytra Demianenka [8] a Bc. Lukáša Pisarčíka [9], ktorí rozšírili systém o autentifikáciu, správu úloh a zber údajov o aktivitách používateľov. Predchádzajúce práce sa tiež zameriavali na vytváranie scenárov riadených Petriho sieťami v platforme A-Frame¹ pre 3D prostredia, Angular² pre klientskú časť (angl. frontend) a Java (Spring Boot)³

¹A-Frame. Dostupný na: <https://aframe.io>

²Angular. Dostupný na: <https://angular.dev/>

³Spring Boot. Dostupný na: <https://spring.io/projects/spring-boot>

pre serverovú časť (angl. backend) s cieľom vytvoriť netriviálne procesy učenia podobné hrám.

V tejto práci sa zameriame na rozšírenie systému gestovým ovládaním rúk, ktoré nahradí tradičné ovládače. Tým sa zvýši interaktivita v spoločných virtuálnych vzdelávacích prostrediach, kde budú študenti môcť prirodzene interagovať s objektmi a medzi sebou.

Formulácia úlohy

Cieľom tejto práce je rozšíriť existujúci systém scenárov riadených virtuálnych vzdelávacích prostredí o ovládanie pomocou gest rúk, ktoré by umožnilo prirodzenejšiu interakciu používateľa bez potreby špecializovaného hardvéru. Zámerom je zároveň zachovať kompatibilitu s existujúcim riešením založeným na Petriho sieťach, ktoré slúžia na riadenie komplexnejších procesov v rámci vzdelávacích scenárov.

V rámci práce sa predpokladá analýza existujúcich technológií detekcie a sledovania gest rúk, najmä knižníc ako MediaPipe⁴ a WebXR Hand Tracking [10], a posúdenie možností ich integrácie do platformy A-Frame. Súčasťou bude aj návrh spôsobu prepojenia detekcie gest s Petriho sieťami tak, aby bolo možné ovplyvňovať priebeh a stav scenárov na základe interakcie používateľa.

Ďalším cieľom je navrhnuť architektúru rozšírenia, v ktorej budú jednotlivé gestá interpretované ako akcie vo virtuálnom prostredí, napríklad manipulácia s objektmi, navigácia alebo spúšťanie udalostí. Dôraz sa kladie na flexibilitu mapovania gest, aby bolo možné systém jednoducho rozširovať alebo upravovať.

Súčasťou práce bude návrh a implementácia prototypu gestového ovládania v testovacom scenári, ktorý umožní overiť funkčnosť navrhnutého prístupu v praxi a zabezpečiť jeho kompatibilitu so spoločnými VR priestormi.

Na záver sa predpokladá vyhodnotenie navrhnutého riešenia prostredníctvom používateľského testovania so zameraním na použiteľnosť, prirodzenosť interakcie a efektívnosť ovládania, pričom výsledky budú porovnané s tradičným ovládaním pomocou VR ovládačov.

⁴MediaPipe. Dostupný na: https://developers.google.com/mediapipe/solutions/vision/hand_landmarker

1 Analýza

Cieľom tejto kapitoly je analyzovať súčasný stav systému LirkisEduVePn, preskúmať podobné riešenia v oblasti vzdelávania vo virtuálnej realite a zhodnotiť technológie, ktoré je možné použiť na implementáciu gestového ovládania.

Analýza vychádza z odborných článkov a vedeckých prác z databáz ako *Scopus*, *Google Scholar* a *MDPI*, ako aj z predchádzajúcich prác projektu [6, 8, 9, 11, 12, 13, 14].

Cieľom je identifikovať obmedzenia súčasného systému a pripraviť návrh rozšírenia o prirodzené ovládanie prostredníctvom gest rúk, ktoré zlepší interaktivitu a dostupnosť platformy.

1.1 Analýza predchádzajúcich riešení

Projekt LirkisEduVePn vznikol ako dlhodobá iniciatíva na Katedre počítačov a informatiky Technickej univerzity v Košiciach s cieľom vytvoriť systém virtuálnych vzdelávacích prostredí riadených procesnými grafmi typu Petriho siete. Cieľom tohto prístupu je umožniť tvorbu výučbových scenárov, ktoré majú logickú štruktúru a reagujú na činnosti používateľa. Študent v prostredí vykonáva sériu akcií, pričom ďalší priebeh scenára závisí od splnenia predchádzajúcich krokov. Takýto systém umožňuje adaptívne učenie, ktoré je interaktívne a viac pripomína herný prístup k vzdelávaniu [6].

1.1.1 Počiatočný vývoj systému

Základy systému položil v roku 2022 Adam Kašela vo svojej diplomovej práci [6]. Vytvoril prvý funkčný prototyp výučbového prostredia zameraného na dejepis, konkrétne na tému Habsburskej monarchie. Prostredie bolo navrhnuté ako virtuálne múzeum, v ktorom sa používateľ mohol pohybovať a interagovať s exponátmi. Každý objekt obsahoval informačné texty a žiak si mohol overiť svoje vedomosti prostredníctvom úloh či kvízových otázok. Implementácia bola založená

na rámci A-Frame, ktorý umožňuje zobrazovanie trojrozmerných prostredí priamo v prehliadači, a na Petriho sieťach, ktoré riadili priebeh výučby. Práca ukázala, že je možné využiť Petriho siete ako formálny model riadenia výučbového procesu v prostredí virtuálnej reality.

1.1.2 Rozšírenie o klient-server architektúru

Na tento základ nadviazali v roku 2023 Dmytro Demianenko a Lukáš Pisarčík [8, 9], ktorí rozšírili projekt o kompletnú architektúru typu klient–server. Ich cieľom bolo premeniť pôvodné prostredie na plnohodnotný systém so správou používateľov, scenárov a štatistík. Klientská časť bola implementovaná v Angulari, čo umožnilo dynamickú prácu s rozhraním a komunikáciu so serverom prostredníctvom HTTP požiadaviek vo formáte JSON. Serverová časť bola vytvorená v jazyku Java s použitím rámca Spring Boot, ktorý zabezpečuje spracovanie údajov, autentifikáciu používateľov a prístup k relačnej databáze PostgreSQL⁵. Zavedenie Docker kontajnerizácie prinieslo možnosť nasadzovať systém na rôzne prostredia bez zmeny konfigurácie. Týmto riešením projekt získal modulárnu architektúru, ktorá podporuje rozširovanie a údržbu.

1.1.3 Stabilizácia a nasadenie systému

Juraj Rudy vo svojej diplomovej práci z roku 2024 [11] sa zamerail na technické vylepšenia, bezpečnosť a nasadenie systému na univerzitný server. Implementoval SSL certifikát, prechod na protokol HTTPS a prispôbenie klientskej časti pre rôzne typy zariadení vrátane mobilných telefónov a VR náhlavných súprav. V rámci jeho práce prebehlo aj testovanie s používateľmi, ktoré ukázalo, že aj netechnicky zameraní učitelia dokážu pochopiť princípy Petriho sietí a využívať ich pri tvorbe výučbových scenárov. Výsledky experimentov potvrdili vhodnosť systému ako pedagogického nástroja, ktorý môže byť využitý vo vzdelávacích inštitúciách.

1.1.4 Rozšírenie o nové výučbové scenáre

V roku 2025 pokračoval v rozvoji projektu Michal Sakáč [12], ktorý vytvoril nové výučbové prostredie zamerané na výučbu skladania počítačov. Scéna mala podobu kancelárskeho priestoru s rozloženými komponentmi, ktoré študent mohol vkladať do počítačovej skrine a prakticky simulovať proces montáže. Pre realis-

⁵PostgreSQL. Dostupný na: <https://www.postgresql.org/docs/>

tickú interakciu s objektmi použil fyzikálnu knižnicu Ammo.js⁶. Všetky úlohy boli riadené Petriho sieťami, čo umožnilo dynamické vyhodnocovanie postupu študenta a opakovanie krokov v prípade chýb. Táto práca ukázala, že systém je možné aplikovať aj na technicky orientované predmety a že architektúra je dostatočne flexibilná pre pridávanie nových scén bez zásahu do základného kódu.

1.1.5 Optimalizácia a refaktorizácia systému

Súbežne s tým realizoval Dmytro Kolosovskiy [13] rozsiahlu refaktorizáciu systému a overenie jeho funkcionality. Jeho cieľom bolo zjednotiť kód serverovej časti, zlepšiť výkonnosť databázy a opraviť nepresnosti vo výmene dát medzi klientom a serverom. Zameral sa aj na evidenciu používateľských aktivít a ich prepojenie so štatistickými modulmi systému. Vďaka týmto úpravám sa zvýšila spoľahlivosť aplikácie a pripravenosť systému na implementáciu ďalších rozšírení, vrátane nových interakčných metód.

1.1.6 Rozšírenie interaktívnych komponentov

Kristián Zsigó [14] vo svojej diplomovej práci pridal do systému sadu nových komponentov určených na jednoduchšie vytváranie scén v A-Frame. Vyvinul množstvo interaktívnych objektov, ktoré demonštrovali rôzne typy aktivít, ako hádzanie predmetov, otváranie dverí alebo triedenie objektov. Vytvoril aj ukážkové prostredie slúžiace ako príklad pre tvorcov výučbových scén, ktoré pomáha učiteľom pochopiť možnosti interakcie v rámci systému. Okrem toho upravil komponenty zodpovedné za komunikáciu medzi scénami a Petriho sieťami, čím zjednodušil ich integráciu do nových projektov.

1.1.7 Zhrnutie a identifikácia obmedzení

Analýza doterajšieho vývoja projektu ukazuje, že LirkisEduVePn dosiahol vysoký stupeň technickej vyspelosti a ponúka stabilnú architektúru pre tvorbu a správu výučbových prostredí. Systém integruje klientskú časť v Angulari, serverovú časť v Spring Boot a databázu PostgreSQL, pričom využíva rámec A-Frame na zobrazenie trojrozmerných scén.

Napriek tomu stále chýba prirodzenejšia forma ovládania – gestové riadenie rúk. Táto funkcionality by mohla odstrániť potrebu používať ovládače alebo klávesnicu a zároveň zvýšiť úroveň interaktivity a prítomnosti študenta vo virtuál-

⁶Ammo.js. Dostupný na: <https://github.com/kripken/ammo.js>

nom prostredí. Implementácia takéhoto riešenia predstavuje logické pokračovanie vývoja systému a je hlavným cieľom tejto práce.

1.2 Analýza možností gestového ovládania

Cieľom tejto časti je preskúmať dostupné prístupy k implementácii gestového ovládania vo virtuálnych prostrediach a zhodnotiť ich vhodnosť pre integráciu do systému LirkisEduVePn. Pozornosť je venovaná najmä možnostiam sledovania rúk v prostredí WebXR, ich spracovaniu v rámci A-Frame a obmedzeniam, ktoré tieto technológie prinášajú pri práci s fyzikálnymi interakciami a existujúcimi komponentmi systému. Na základe tejto analýzy budú identifikované vhodné prístupy pre návrh vlastného riešenia gestového ovládania.

1.2.1 Natívne sledovanie rúk cez A-Frame

Rámec A-Frame poskytuje základnú podporu pre sledovanie rúk prostredníctvom WebXR Hand Tracking API [10], ktoré je implementované v komponentoch `hand-controls`⁷ a `hand-tracking-controls`⁸. Tento prístup využíva štandardizované rozhranie WebXR Device API [15] pre prístup k polohovým dátam jednotlivých kĺbov ruky, pričom prehliadač komunikuje priamo s hardvérom VR náhlavnej súpravy.

Rozdiel medzi `hand-controls` a `hand-tracking-controls`

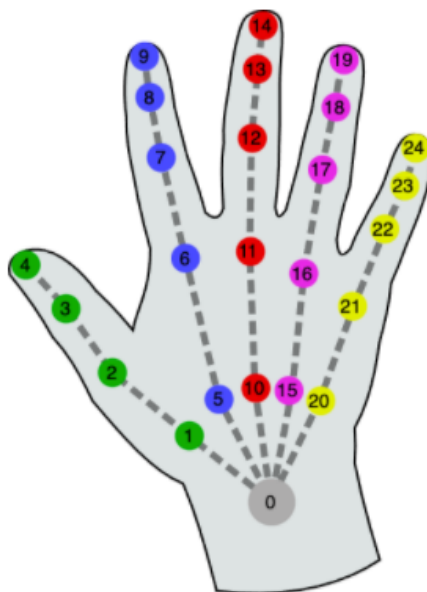
Komponent `hand-controls` je určený na zobrazenie a ovládanie fyzických VR ovládačov. Poskytuje vizuálne modely ovládačov pre rôzne VR náhlavné súpravy a reaguje na tlačidlá a spúšťače (angl. `triggers`). Generuje udalosti ako `trigger-down`, `gripdown` a `thumbstickdown`, ktoré sú štandardnou súčasťou ovládania vo VR aplikáciách.

Na druhej strane, komponent `hand-tracking-controls` slúži na zobrazenie a sledovanie skutočných rúk používateľa bez potreby fyzických ovládačov. Využíva kamery VR náhlavnej súpravy na detekciu pozícií až 25 kĺbov na každej ruke (obrázok 1.1) a zobrazuje 3D model ruky v reálnom čase. Na rozdiel od `hand-controls`, tento komponent negeneruje štandardné udalosti tlačidiel (an-

⁷A-Frame komponent `hand-controls`. Dostupný na: <https://aframe.io/docs/1.7.0/components/hand-controls.html>

⁸A-Frame komponent `hand-tracking-controls`. Dostupný na: <https://aframe.io/docs/1.6.0/components/hand-tracking-controls.html>

gl. button events), ale poskytuje iba pozičné dáta jednotlivých kĺbov, čo vyžaduje vlastnú implementáciu detekcie gest.



Obrázok 1.1: Schéma 25 kĺbov ruky podľa špecifikácie WebXR Hand Input Module

Hlavnou výhodou natívneho riešenia je vizuálna presnosť a stabilita modelu ruky v trojrozmernom priestore. Systém dokáže v reálnom čase sledovať až 25 kĺbov na každej ruke, čo umožňuje prirodzenú reprezentáciu pohybov prstov a zápästia. Táto presnosť je vhodná najmä pre interakcie s používateľským rozhraním, ako je stláčanie virtuálnych tlačidiel, posúvanie panelov alebo výber položiek z menu.

Obmedzenia natívneho riešenia

Problém však nastáva pri pokuse o fyzikálne interakcie s objektmi v scéne. A-Frame natívne nepodporuje kolíznú detekciu pre jednotlivé kĺby ruky, čo znamená, že model ruky prechádza cez objekty bez akejkoľvek fyzikálnej reakcie. Pri pokuse o uchopenie objektu chýba mechanizmus na vytvorenie fyzikálneho spojenia medzi rukou a objektom. Táto situácia sa komplikuje pri použití fyzikálnej knižnice Ammo.js, ktorá vyžaduje explicitné definovanie kolíznych oblastí a fyzikálnych väzieb.

Ďalším obmedzením je nekompatibilita s komponentom Super Hands⁹, ktorý očakáva vstupy z tradičných ovládačov a nie je optimalizovaný pre prácu s dátami

⁹Super Hands. Dostupný na: <https://github.com/c-frame/afame-super-hands-component>

zo sledovania rúk. Pri súčasnom použití oboch systémov dochádza ku konfliktom v refazci spracovania udalostí (angl. event pipeline), kde udalosti zo sledovania rúk nie sú správne rozpoznávané ako interakčné vstupy.

1.2.2 Použitie Superhands pre sledovanie rúk

Komponent Super Hands bol pôvodne navrhnutý pre interakcie s VR ovládačmi a poskytuje robustný systém pre manipuláciu s objektmi – uchopenie (angl. grabbing), pustenie (angl. dropping), ťahovanie (angl. stretching) a presúvanie (angl. drag-and-drop) operácie. Jeho hlavnou prednosťou je komplexný systém udalostí, ktorý pokrýva široké spektrum interakčných scenárov.

Pri použití s fyzikálnymi objektmi Super Hands funguje spoľahlivo, pretože obsahuje vlastné mechanizmy na detekciu kolízií a správu stavov objektov: grabbed, stretched a dragover. Umožňuje tiež definovať rôzne reakcie objektov na interakcie, čo je užitočné pri tvorbe výučbových scenárov.

Integrácia Super Hands s WebXR sledovaním rúk však prináša technické výzvy. Komponent nebol projektovaný s ohľadom na gestové ovládanie, preto vyžaduje vytvorenie neviditeľných fyzikálnych telies (angl. collider proxy) priradených ku kĺbom ruky, ktoré slúžia ako sprostredkovatelia kolízií. Tieto telesá musia byť synchronizované v každej snímke s pozíciou hand-tracking-controls kĺbov, čo môže viesť k neželaným mikropohybom (angl. jitter) a nestabilite pri manipulácii s objektmi.

Ďalším problémom je, že Super Hands očakáva vstupy typu triggerdown a triggerup z ovládačov, zatiaľ čo component hand-tracking-controls poskytuje kontinuálne pozičné dáta bez explicitne definovaných spúšťacích udalostí. Je preto potrebné implementovať detekciu gest (napr. štipnutie prstami) a ich mapovanie na spúšťacie udalosti, čo pridáva ďalšiu vrstvu zložitosti.

1.2.3 Hybridný režim: kombinácia VR ovládača a sledovania rúk

Hybridný prístup predstavuje kompromis medzi plnohodnotným sledovaním rúk a tradičným ovládaním. V tomto režime zvyčajne dominantná ruka využíva ovládač s komponentom hand-controls, zatiaľ čo druhá ruka používa pre gestové interakcie hand-tracking-controls.

Táto konfigurácia umožňuje využiť výhody oboch systémov, keďže ovládač poskytuje presné a spoľahlivé uchopenie objektov cez Super Hands, zatiaľ čo sledovaná ruka môže byť použitá na gestové ovládanie používateľského rozhrania: ukazovanie, aktiváciu tlačidiel alebo navigáciu v menu.

Z pohľadu používateľského zážitku však tento prístup vytvára asymetriu, ktorá môže byť mäťúca. Používateľ musí vedome rozlišovať, ktorú ruku má použiť na ktorý typ interakcie, čo zvyšuje kognitívnu záťaž a narúša pocit prirodzenosti virtuálneho prostredia. V edukačnom kontexte, kde je cieľom minimalizovať technické prekážky a umožniť študentom sústrediť sa na obsah, tento nedostatok môže byť významný.

Hybridný režim je však vhodný ako prechodné riešenie počas vývoja alebo ako možnosť pre používateľov, ktorí preferujú kombinované ovládanie.

1.2.4 Vlastný hand-grab systém

Implementácia plne vlastného systému pre gestové ovládanie predstavuje najkomplexnejšie, ale zároveň najflexibilnejšie riešenie. Tento prístup zahŕňa priamu prácu s WebXR Device API na nižšej úrovni, kde máme plnú kontrolu nad spracovaním dát zo sledovania rúk.

Základom vlastného systému je detekcia dotyku medzi kĺbmi ruky a objektmi v scéne. To vyžaduje vytvorenie vlastných fyzikálnych telies pre jednotlivé segmenty ruky – špičky prstov, kĺby, dlaň. Tieto kolízne objekty musia byť optimalizované tak, aby minimalizovali výpočtovú záťaž, ale zároveň poskytovali dostatočnú presnosť pre detekciu kontaktu.

Pri zachytení objektu je potrebné implementovať fyzikálne väzby cez Ammo.js alebo podobnú knižnicu. Väzba definuje spojenie medzi objektom a rukou, pričom musí zohľadňovať orientáciu, vzdialenosť a silu uchopenia. Pre realistický pocit je nutné implementovať mechanizmy, ako minimálna vzdialenosť a konfigurácia prstov pre uchopenie (angl. grab threshold) a podmienky pre pustenie objektu (angl. release detection).

Hlavnou výhodou tohto prístupu je úplná kompatibilita s WebXR bez závislosti na externých komponentoch A-Frame, ktoré môžu spôsobovať konflikty. Systém môže byť presne prispôsobený potrebám projektu LirkisEduVePn, vrátane integrácie s Petriho sieťami a špecifickými vzdelávacími scenármi.

Nevýhodou je extrémna komplexnosť implementácie. Musíme riešiť nie len detekciu kolízií, ale aj optimalizáciu výkonu, správu stavov, okrajové prípady (napr. súčasné uchopenie viacerých objektov). Vývoj takéhoto systému vyžaduje hlboké pochopenie WebXR špecifikácie, fyzikálnych simulácií a architektúry systému LirkisEduVePn.

1.2.5 Použitie externých knižníc

V open-source komunite existuje niekoľko pokusov o vytvorenie hand physics knižníc pre WebXR a A-Frame. Projekty ako `handy-work`¹⁰, odvetvenia `Hand Physics Three.js`¹¹ alebo `A-Frame Hand Physics`¹² ponúkajú vopred implementované riešenia pre detekciu kolízií a interakcie uchopenia.

Výhodou týchto knižníc je, že poskytujú hotové komponenty, ktoré teoreticky eliminujú potrebu vytvárať vlastný systém od nuly. Niektoré z nich sú postavené priamo na `Three.js`¹³ alebo `A-Frame` a sľubujú jednoduchú integráciu.

Problémom je kolízia s `A-Frame` entity systémom. Mnohé externé knižnice predpokladajú priamy prístup k `Three.js` scéne a objektom, zatiaľ čo `A-Frame` pracuje s vyššou úrovňou abstrakcie `entity-component-system`. Pri integrácii môže dochádzať k problémom, keď externá knižnica nedokáže správne pracovať s objektmi vytvorenými v `A-Frame` alebo keď sa tieto systémy navzájom ovplyvňujú.

Napriek týmto problémom môžu byť niektoré knižnice užitočné ako inšpirácia alebo ako základ pre vlastnú implementáciu.

1.3 Integrácia gest s procesným riadením prostredia

V tejto časti sú analyzované technické problémy vyplývajúce z integrácie gestového ovládania do systému.

1.3.1 Problémy spôsobené integráciou

Implementácia gestového ovládania v systéme `LirkisEduVePn` vedie k viacerým technickým konfliktom vyplývajúcim z kombinácie rôznych rámcov a knižníc, z ktorých každá má svoj vlastný model správy stavov.

Konkurencia medzi komponentmi

V typickej konfigurácii systému `LirkisEduVePn` beží súčasne niekoľko `A-Frame` komponentov, ktoré si navzájom konkurujú o kontrolu nad entitami:

- `hand-controls` a `hand-tracking-controls` – oba sa pokúšajú ovládať tú istú entitu reprezentujúcu ruku

¹⁰Handy-work. Dostupný na: <https://github.com/AdaRoseCannon/handy-work>

¹¹Three.js Hand Physics. Dostupný na: <https://threejs.org/examples/?q=hand>

¹²A-Frame Hand Physics. Dostupný na: <https://github.com/c-frame>

¹³Three.js. Dostupný na: <https://threejs.org/>

- `super-hands` – pridáva vlastné poslucháče udalostí (angl. event listeners) a logiku pre kolízie
- `blink-controls`¹⁴ – riadi teleportáciu používateľa
- `raycaster`¹⁵ – deteguje interakcie s objektami na diaľku
- `aframe-physics-system`¹⁶ – spravuje fyzikálne simulácie a kolízie

Každý z týchto komponentov registruje svoje obsluhy udalostí, modifikuje entity properties a očakáva určitú konfiguráciu scény. Pri súčasnom použití môže dochádzať k situáciám, kde jeden komponent prepíše zmeny vykonané iným komponentom, alebo kde udalosť je zachytená viacerými komponentmi a spracovaná viacnásobne.

Konflikty názvov komponentov

Problém sa objavuje aj pri pomenovaniach. Napríklad atribút `grabbable` môže byť definovaný natívnym A-Frame komponentom, ale aj Super Hands komponentom, pričom oba očakávajú rôzne parametre a správanie. Pri registrácii oboch komponentov v jednom projekte dochádza ku kolízii, kde jeden prepíše definíciu druhého, čo vedie k nefunkčnosti interakcií.

Kolízie pri fyzikálnych obmedzeniach

`Ammo.js`, používaný pre fyzikálne simulácie, vyžaduje, aby kolízne objekty boli presne definované a aby fyzikálne spojenia medzi objektmi boli správne nakonfigurované. Pri sledovaní rúk je potrebné dynamicky vytvárať a rušiť obmedzenia pri každom uchopení a pustení objektu. Ak tieto operácie nie sú synchronizované s fyzikálnou simuláciou, môže dôjsť k situáciám, keď objekt zostane „prilepený“ k ruke aj po pustení, alebo keď sa spojenie medzi rukou a objektom neočakávane odstráni, čo spôsobí nerealistické správanie.

Inline scriptovanie a reload WebXR session

WebXR session je inicializovaná pri vstupe do VR režimu a je citlivá na zmeny v štruktúre scény. Ak je potrebné dynamicky pridávať alebo odstraňovať entity

¹⁴A-Frame komponent `blink-controls`. Dostupný na: <https://github.com/jure/aframe-blink-controls>

¹⁵A-Frame komponent `raycaster`. Dostupný na: <https://aframe.io/docs/1.6.0/components/raycaster.html>

¹⁶Aframe-physics-system. Dostupný na: <https://github.com/n5ro/aframe-physics-system>

počas behu aplikácie (napr. pri prechode medzi vzdelávacími scenármi), môže dôjsť k zrušeniu WebXR session, čo núti používateľa znovu vstúpiť do VR.

1.3.2 Prepojenie so simuláciou Petriho siete

Hlavnou úlohou integrácie gestového ovládania do systému LirkisEduVePn je zabezpečiť, aby gestá používateľa mohli slúžiť ako vstupy pre Petriho siete, ktoré riadia priebeh výučbových scenárov.

Gestá ako spúšťače prechodov

V Petriho sieti každý prechod (angl. transition) vyžaduje splnenie určitých podmienok – prítomnosť tokenov v vstupných miestach a často aj externú udalosť. Gestá môžu byť mapované na tieto externé udalosti. Napríklad:

- Gesto "uchop objekt" – spustí prechod "aktivuj úlohu"
- Gesto "ukáž na tlačidlo" – spustí prechod "zobraz nápoje"
- Gesto "otvorená dlaň" – spustí prechod "resetuj scénu"

Tento prístup vyžaduje vytvorenie mapovania medzi rozpoznanými gestami a identifikátormi prechodov v Petriho sieti. Klientská časť musí detegovať gesto, transformovať ho na štruktúrovanú správu (napr. JSON objekt obsahujúci typ gesta, timestamp a prípadné parametre) a odoslať túto správu cez Angular službu do Spring Boot serverovú časť.

Potreba deterministického systému udalostí

Kritickou požiadavkou je deterministickosť – rovnaké gesto vykonané v rovnakej situácii musí vždy viesť k rovnakej reakcii systému. To je problematické pri sledovaní rúk, kde pozície kĺbov sú poskytované ako plynulé hodnoty s vysokou frekvenciou, typicky 60–90 Hz. Systém musí implementovať mechanizmy potlačenia opakovaných udalostí (angl. debouncing) a prahové hodnoty (angl. threshold), aby zabránil viacnásobnému spusteniu rovnakého prechodu pri jednom geste.

Ďalším problémom je, že WebXR udalosti gest nie sú štandardnou súčasťou A-Frame reťazca spracovania udalostí. Zatiaľ čo tradičné udalosti ako `click` alebo `triggerdown` sú automaticky propagované cez A-Frame komponent systém, vlastné udalosti gest musia byť explicitne emitované a zachytené. Ak tento mechanizmus nie je správne implementovaný, obsluhy udalostí v Angulari nemusia zaznamenať výskyt gesta.

1.3.3 Komponenty v konflikte

grabbable (A-Frame) a grabbable (Superhands)

A-Frame obsahuje základný `grabbable` atribút, ktorý označuje, že entita môže byť uchopená ovládačom. Super Hands zavádza vlastný komponent `grabbable` s rozšírenou funkcionalitou, ktorá zahŕňa fyzikálne uchopovanie objektov (angl. *physics-based grabbing*), obsluhu udalostí a konfiguráciu správania pri ťažení objektov. Keď sú oba registrované v projekte, dochádza ku konfliktu názvov, kde jeden prepíše druhý. Riešením je explicitné premenovanie jedného z komponentov, alebo použitie len jednej z implementácií.

Sledovanie rúk (A-Frame) oproti uchopovaniu pomocou kolíznych objektov (Super Hands)

Natívny A-Frame hand tracking poskytuje vizuálnu reprezentáciu rúk, ale bez fyzikálnych colliderov. Super Hands očakáva kolízne oblasti priradené k entitám, ktoré majú komponent `hand-controls`. Pri pokuse o kombináciu oboch vzniká situácia, kde model ruky je viditeľný, ale nemá fyzikálne telo, takže Super Hands nemôže detegovať kolízie. Riešenie vyžaduje vytvorenie synchronizovaných fyzikálnych telies alebo implementáciu vlastného systému detekcie kolízií.

Synchronizácia fyzikálnych kolíznych objektov a interakcia založená na kĺboch

`Hand-tracking-controls` poskytuje pozície kĺbov v priestore, zatiaľ čo fyzikálny systém (*Ammo.js*) pracuje s tuhými telesami (angl. *rigid body*) a kolíznymi objektmi. Synchronizácia medzi týmito dvoma systémami je náročná – kolízne objekty musia byť aktualizované v každom snímku podľa pozícií kĺbov, pričom nesmie dôjsť k situáciám, kde fyzikálne telo "zaspí" (*physics sleep*) a prestane reagovať na zmeny pozície. Interakcie založené na kĺboch (napr. štipnutie prstami) vyžadujú vlastnú detekčnú logiku, ktorá analyzuje vzdialenosti a uhly medzi konkrétnymi kĺbmi, čo pridáva ďalšiu vrstvu výpočtov do aktualizáčnej slučky (angl. *update loop*).

1.4 Záver analýzy

Analýza preukázala, že implementácia gestového ovládania je prirodzeným a efektívnym spôsobom, ako rozšíriť interaktivitu a funkcionalitu systému LirkisEdu-VePn. Pre projekt bolo ako najvhodnejší základ identifikované natívne sledova-

nie rúk cez WebXR Hand Tracking API, ktoré je priamo podporované rámcom A-Frame a moderné VR náhlavné súpravy (Meta Quest 2/3) ho zvládajú bez potreby ďalšieho hardvéru.

Analýza zároveň odhalila technické obmedzenia existujúcich riešení. Medzi komponentmi hand-controls, hand-tracking-controls a Super Hands dochádzalo ku konfliktom. Natívne sledovanie rúk neposkytuje fyzickú prítomnosť modelu ruky v scéne. Knižnica Super Hands bola navrhnutá primárne pre fyzické ovládače, čo komplikovalo jej integráciu. Náročná bola aj synchronizácia kolíznych telies s fyzikálnou simuláciou Ammo.js. Tieto obmedzenia ukazujú, že priame použitie existujúcich knižníc nie je pre potreby projektu postačujúce.

Z analýzy preto vyplýva, že najvhodnejším riešením je navrhnuť vlastný modulárny systém postavený nad natívnym sledovaním rúk, ktorý bude poskytovať jednotnú vrstvu pre jednotlivé typy interakcií, zachová kompatibilitu s existujúcimi komponentmi systému a umožní bezproblémovú integráciu s logikou Petriho sietí. Tento prístup predstavuje logické pokračovanie vývoja systému a vytvára priestor pre ďalšie experimenty v oblasti adaptívnej a interaktívnej výučby vo virtuálnej realite.

2 Návrh riešenia

V tejto kapitole je opísaný návrh systému gestového ovládania v rámci projektu LirkisEduVePn. Návrh vychádza zo záverov analýzy uvedených v kapitole 1 a z požiadavky umožniť prirodzenú interakciu vo virtuálnom prostredí bez potreby fyzických VR ovládačov.

Navrhované riešenie je založené na natívnom sledovaní rúk cez WebXR Hand Tracking API a na rámec A-Frame. Namiesto využitia univerzálnej knižnice pre manipuláciu s objektmi bol zvolený vlastný modulárny systém, ktorý poskytuje väčšiu kontrolu nad spracovaním gest, fyzikálnym správaním objektov a integráciou s existujúcou architektúrou projektu.

2.1 Východiskový problém a cieľ návrhu

Pôvodný systém LirkisEduVePn využíval na interakciu s virtuálnym prostredím najmä fyzické VR ovládače. Tento prístup je funkčný, no z pohľadu používateľského komfortu a dostupnosti má viacero nevýhod. Používateľ musí mať k dispozícii ovládače, ich ovládanie je menej prirodzené než priame používanie rúk a v školskom prostredí môže byť problém zabezpečiť dostatočný počet zariadení pre väčší počet študentov.

Cieľom návrhu bolo vytvoriť systém, ktorý umožní používateľovi ovládať virtuálne prostredie priamo rukami. Navrhované riešenie má pokrývať základné typy interakcií potrebné vo vzdelávacích scenároch: uchopenie a manipuláciu s objektmi, dotykové ovládanie používateľského rozhrania, pohyb po scéne a interakciu na diaľku, obdobnú ako pri fyzických VR ovládačoch.

Dôležitou požiadavkou bolo zároveň zachovať kompatibilitu s existujúcimi komponentmi projektu a podporiť hybridný režim, v ktorom môžu mať fyzické ovládače prioritu pred sledovaním rúk.

2.2 Kľúčové požiadavky

Na základe analýzy existujúcich riešení a odbornej literatúry boli definované tieto hlavné požiadavky:

- **Prirodzená interakcia** – používateľ musí vedieť vykonávať základné akcie pomocou jednoduchých a zrozumiteľných gest, napríklad uchopenie objektu štipnutím prstov alebo potvrdenie teleportácie rovnakým gestom.
- **Fyzická prítomnosť ruky v scéne** – virtuálna ruka musí byť schopná fyzikálne reagovať na objekty v prostredí, aj keď natívne sledovanie rúk A-Frame priamo neposkytuje kolízne telá pre kĺby.
- **Presná manipulácia s objektmi** – objekt musí byť možné plynulo presúvať, otáčať a po pustení hodiť so zachovaním vypočítanej rýchlosti.
- **Podpora hybridného režimu** – systém musí korektne spolupracovať aj s fyzickými VR ovládačmi, pričom pri ich aktivite musí vedieť obmedziť logiku komponentu hand-tracking-controls.
- **Interakcia na diaľku** – používateľ musí vedieť interagovať so vzdialenými objektmi bez potreby priameho fyzického kontaktu.
- **Výkon v reálnom čase** – všetky detekcie a aktualizácie musia prebiehať v každom snímku bez citelného poklesu plynulosti scény.

2.3 Architektúra riešenia

Na základe výsledkov analýzy bol zvolený prístup vlastnej implementácie. Dôvodom bolo najmä to, že univerzálne riešenia typu Super Hands neboli vhodné pre čistý hand-tracking-controls a spôsobovali konflikty v spracovaní udalostí aj pri fyzikálnej manipulácii s objektmi.

Navrhnutý systém má modulárnu architektúru. Jadrom riešenia je komponent hand-tracking-bridge, ktorý sprostredkúva údaje z komponentu hand-tracking-controls ostatným častiam systému. Nad týmto komponentom je vytvorených päť špecializovaných interakčných modulov, pričom každý rieši jeden typ interakcie.

2.3.1 Vstupná vrstva

Vstupnú vrstvu tvorí WebXR Hand Tracking API, ktoré poskytuje polohy kĺbov ruky v reálnom čase. V prostredí A-Frame sú tieto dáta sprístupnené prostredníctvom komponentu `hand-tracking-controls`. Tento komponent vytvára model ruky a umožňuje prístup ku konkrétnym kĺbom, ako sú zápästie, články prstov alebo ich špičky.

Na rozdiel od komponentu `hand-controls`, ktorý je orientovaný na fyzické VR ovládače, `hand-tracking-controls` neposkytuje klasické udalosti tlačidiel. Poskytuje najmä priestorové dáta. Z tohto dôvodu bolo potrebné vytvoriť vlastnú vrstvu, ktorá tieto dáta zjednotí a sprístupní ostatným častiam systému vo forme jednoduchého API rozhrania.

2.3.2 Uzlový komponent

Komponent `hand-tracking-bridge` predstavuje centrálny bod navrhovaného riešenia. Je pripojený na tú istú entitu ako `hand-tracking-controls` a slúži ako adaptačná vrstva medzi natívnymi WebXR dátami o sledovaní rúk a interakčnými modulmi.

Jeho úlohou je uchovávať referenciu na `hand-tracking-controls`, sprístupniť ostatným modulom prístup ku geometrii ruky a jednotlivým kĺbom, priebežne počítat bod štipnutia, preposielať natívne udalosti a detegovať aktivitu fyzického ovládača.

Týmto spôsobom vzniká jednotná komunikačná vrstva, cez ktorú môžu všetky moduly reagovať konzistentne a bez priamej závislosti od natívnej implementácie hand trackingu.

2.3.3 Interakčné moduly

Nad uzlovým komponentom je postavená sada piatich samostatných interakčných modulov, pričom každý z nich pokrýva jednu špecifickú oblasť funkcionality.

Modul `hand-grab` zabezpečuje uchopenie objektov a ich následnú manipuláciu vrátane presúvania, otáčania a hádzania. Modul `hand-collision-bodies` vytvára kolízne telesá priradené ku kĺbom ruky, čím zabezpečuje jej fyzikálnu prítomnosť v scéne. Modul `hand-point-fingertip-touch` realizuje dotykovú interakciu s používateľským rozhraním pomocou špičky ukazováka. Modul `hand-teleport` implementuje teleportačný pohyb založený na gestách a integ-

ruje sa s komponentom `blink-controls`. Modul `hand-point-shoot` umožňuje interakciu na diaľku pomocou raycastu (angl. raycasting) a následného pritiahnutia objektov.

Všetky moduly využívajú spoločné dáta a udalosti poskytované komponentom `hand-tracking-bridge`, čo zabezpečuje jednotné rozhranie a zjednodušuje integráciu systému.

2.3.4 Externé systémy

Navrhnuté riešenie spolupracuje s viacerými existujúcimi technológiami projektu. Na úrovni fyzikálnej simulácie využíva engine `Ammo.js`, ktorý zabezpečuje správanie objektov pri kolíziách, manipulácii a hádzaní. Na úrovni interakcií je integrovaný komponent `blink-controls`, ktorý realizuje teleportáciu používateľa.

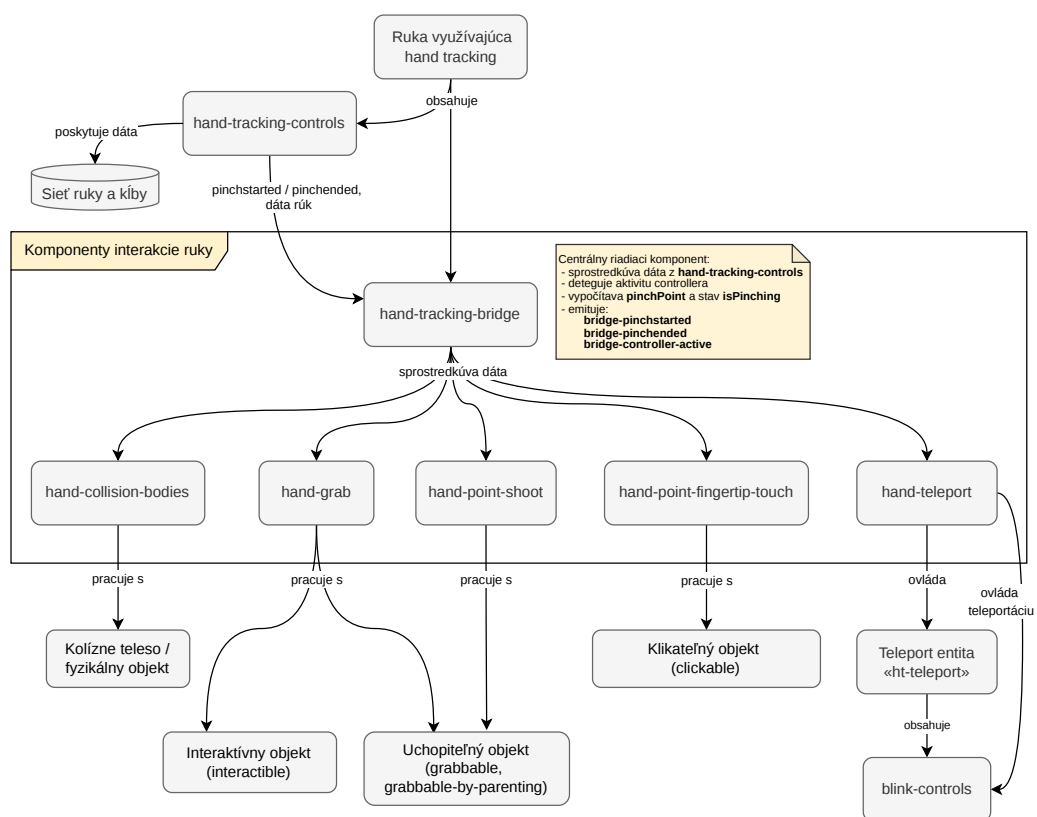
Systém zároveň pracuje s existujúcimi A-Frame atribútmi a triedami v scéne, ako sú `[grabbable]`, `[grabbable-by-parenting]`, `.clickable`, `.collidable` alebo `[keyboard-button]`, ktoré definujú interaktívne správanie objektov.

Z pohľadu architektúry ide o riešenie, ktoré rozširuje A-Frame o vlastnú vrstvu gestového ovládania.

2.4 Štruktúra komponentov

Na obrázku 2.1 je znázornená štruktúra navrhovaného systému. Komponent `hand-tracking-controls` poskytuje model ruky a údaje o kĺboch. Tieto údaje sú prístupné komponentu `hand-tracking-bridge`, ktorý slúži ako centrálny sprostredkovateľ udalostí a stavu. Nad ním sú postavené jednotlivé interakčné moduly.

Komponent `hand-grab` a `hand-collision-bodies` reagujú najmä na udalosti začiatku a konca štipnutia. Komponent `hand-teleport` pracuje so stavom štipnutia a s polohami kĺbov pri rozpoznávaní teleportačných gest. Komponent `hand-point-fingertip-touch` využíva najmä pozíciu špičky ukazováka a komponent `hand-point-shoot` používa kĺby ruky na výpočet smeru ukazovania a na pritiahnutie objektu. Po úspešnom pritiahnutí objektu komponent `hand-point-shoot` odovzdáva riadenie komponentu `hand-grab`.



Obrázok 2.1: Komponentový diagram navrhovaného systému gestového ovládania

3 Implementácia

V tejto kapitole je opísaná implementácia jednotlivých komponentov navrhovaného systému. Pre každý komponent je najprv zhrnutá jeho zodpovednosť a vnútorná štruktúra, následne sú jednotlivé fázy činnosti popísané spolu so sekvenčnými diagramami zachytávajúcimi komunikáciu s ostatnými časťami systému. Poradie sekcií zodpovedá vrstveniu architektúry – od centrálného uzlového komponentu po jednotlivé interakčné moduly.

Z dôvodu čitateľnosti diagramov sú v nich použité skrátené názvy komponentov:

hBridge	hand-tracking-bridge
hGrab	hand-grab
hCollision	hand-collision-bodies
hFingertip	hand-point-fingertip-touch
hTeleport	hand-teleport
hPointShoot	hand-point-shoot

3.1 Komponent **hand-tracking-bridge**

Komponent **hand-tracking-bridge** predstavuje základ celého systému. Jeho úlohou nie je priamo realizovať interakcie, ale poskytovať jednotnú vrstvu nad natívnym sledovaním rúk a sprístupniť tieto dáta ostatným modulom v konzistentnej forme. Vďaka tomu jednotlivé interakčné komponenty nemusia pracovať priamo s detailmi implementácie **hand-tracking-controls**.

3.1.1 Štruktúra komponentu

Komponent obsahuje konfiguračné parametre a interný stav potrebný pre riadenie interakcie. Konfiguračné parametre umožňujú určiť, s ktorou rukou komponent pracuje (**hand**), riadiť vizualizáciu bodu štípnutia (**sphereRadius**, **pinchSphereOpacity**) a zapínať ladiace mechanizmy (**debugSphere**, **debugLog**).

Zároveň obsahuje prepínače ako `collisionEnabled`, `grabEnabled`, `finger-tipTouchEnabled`, `teleportEnabled` a `pointShootEnabled`, ktoré umožňujú selektívne aktivovať jednotlivé interakčné moduly. Parameter `autoAttachModules` určuje, či sa tieto moduly majú automaticky pripojiť k entite ruky.

Interný stav komponentu zahŕňa referencie na `hand-tracking-controls` a `hand-controls`, príznak aktivity ovládača (`controllerActive`) a stav gesta štípnutia (`isPinching`). Kľúčovou dátovou štruktúrou je `pinchData`, ktorá obsahuje pozície špičky palca a ukazováka a z nich odvodený bod štípnutia (`pinchPoint`), ktorý slúži ako hlavný referenčný bod pre interakciu.

3.1.2 Inicializácia komponentu

Pri inicializácii (obrázok 3.1) komponent uloží referenciu sám na seba do entity ruky, čím umožní ostatným modulom jednoduchý prístup. Následne čaká, kým bude pripravený komponent `hand-tracking-controls`, a po jeho dostupnosti vykoná vlastné nastavenie – pripraví detekciu fyzického ovládača, vytvorí pomocnú vizualizáciu bodu štípnutia a zaregistruje natívne udalosti `pinchstarted` a `pinchended` prichádzajúce z `hand trackingu`.

Ak je nastavený parameter `autoAttachModules`, komponent v rovnakom kroku automaticky pripojí k entite ruky aj ostatné interakčné moduly. Každý z týchto modulov si pri svojej vlastnej inicializácii vyžiada referenciu na `bridge` a začne čakať, kým bude k dispozícii. Po ukončení inicializácie majú teda všetky moduly k dispozícii jednotné rozhranie a môžu okamžite reagovať na udalosti emitované `bridge` komponentom.

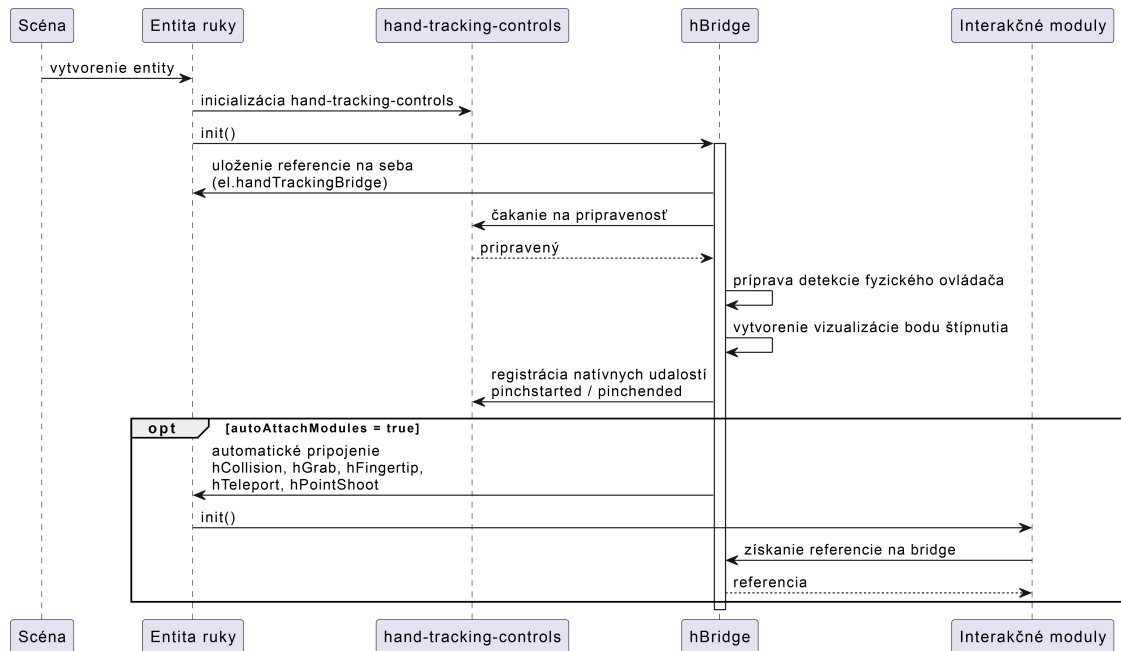
3.1.3 Verejné rozhranie

Komponent poskytuje ostatným modulom jednoduché rozhranie, ktoré umožňuje získať `model` ruky, jednotlivé kĺby a aktuálny bod štípnutia. Okrem toho sprístupňuje pomocné metódy pre zisťovanie aktuálneho režimu interakcie, napríklad či je aktívna `teleportácia` alebo či ruka práve drží objekt.

Táto vrstva abstrahuje detaily práce s natívnym `hand trackingom` a zjednodušuje komunikáciu medzi jednotlivými modulmi systému.

3.1.4 Aktualizácia stavu a distribúcia udalostí

V každom snímku komponent najprv zisťuje, či je aktívny fyzický ovládač (obrázok 3.2). Ak áno, emituje udalosť `bridge-controller-active`, čím umožňuje ostatným modulom pozastaviť `hand tracking` logiku. Ak fyzický ovládač aktívny



Obrázok 3.1: Sekvenčný diagram inicializácie komponentu hBridge

nie je, komponent aktualizuje pinchData na základe pozícií špičky ukazováka a palca a vypočíta aktuálny bod štipnutia ako stred medzi nimi.

Natívne udalosti pinchstarted a pinchended z komponentu sledovania rúk vedú k aktualizácii stavu isPinching a následnému spusteniu udalostí bridge-pinchstarted a bridge-pinchended. Tým sa zabezpečí, že všetky moduly reagujú na rovnaké vstupy jednotne a bez potreby duplikovať tú istú logiku.

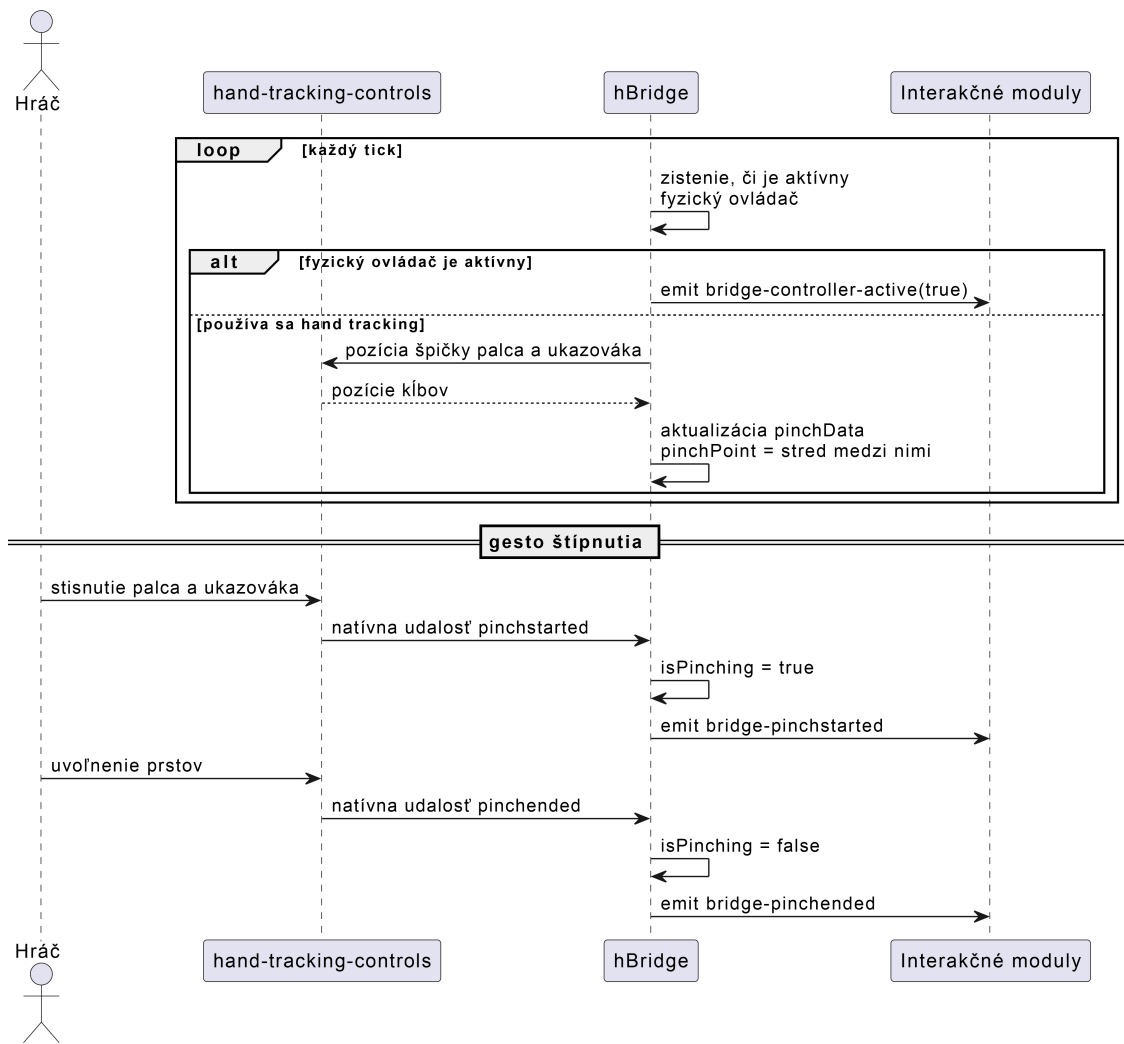
3.2 Komponent hand-grab

Komponent hand-grab zabezpečuje uchopenie objektov a ich následnú manipuláciu. Je navrhnutý tak, aby interakcia pôsobila prirodzene, bola plynulá a zároveň korektne spolupracovala s fyzikálnym systémom.

3.2.1 Schéma komponentu

Komponent je konfigurovateľný pomocou viacerých parametrov, ktoré ovplyvňujú jeho správanie, ako sú maximálna vzdialenosť vyhľadávania uchopiteľných objektov, parametre pre výpočet rýchlosti hodu, nastavenia plynulosti pozície a rotácie, ako aj vzdialenosť pre klikovú interakciu. Voliteľne je možné aktivovať režim snapToHand, ktorý zabezpečuje presné zarovnanie objektu s rukou.

Z hľadiska interného stavu komponent uchováva referenciu na komponent hand-tracking-bridge, referenciu na aktuálne držaný objekt a transformačné



Obrázok 3.2: Sekvenčný diagram aktualizácie stavu hBridge a distribúcie udalostí

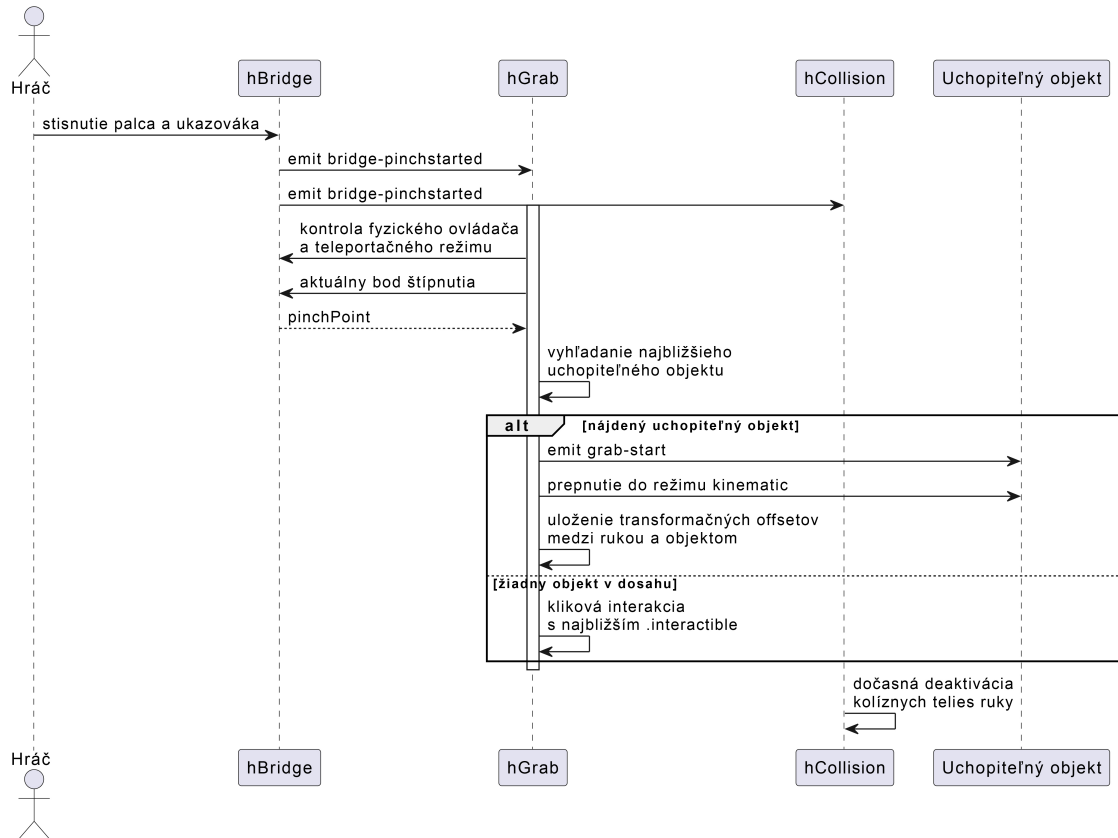
posuny medzi rukou a objektom. Okrem toho si vedie históriu rýchlostí pohybu ruky, ktorá sa využíva pri výpočte výslednej rýchlosti hodu po uvoľnení objektu.

3.2.2 Začiatok uchopenia

Uchopenie objektu sa iniciuje prijatím udalosti `bridge-pinchstarted` (obrázok 3.3). Komponent najprv overí, či nie je aktívny fyzický ovládač a či ruka nie je v teleportačnom režime. Následne získa aktuálnu pozíciu bodu štípnutia od bridge komponentu a vyhladá najbližší objekt označený ako uchopiteľný.

Ak je vhodný objekt nájdený, komponent vyšle udalosť `grab-start`, prepne jeho fyzikálne telo do režimu `kinematic` a uloží transformačné posuny potrebné pre ďalšiu manipuláciu. V prípade, že sa v dosahu nenachádza žiadny uchopiteľný objekt, komponent namiesto toho vykoná klikovú interakciu s najbližším ele-

mentom označeným ako `.interactable`. Paralelne s uchopením komponent `hand-collision-bodies` dočasne deaktivuje kolízne telesá ruky, čím sa predchádza nežiaducim fyzikálnym interakciám s uchopeným objektom.

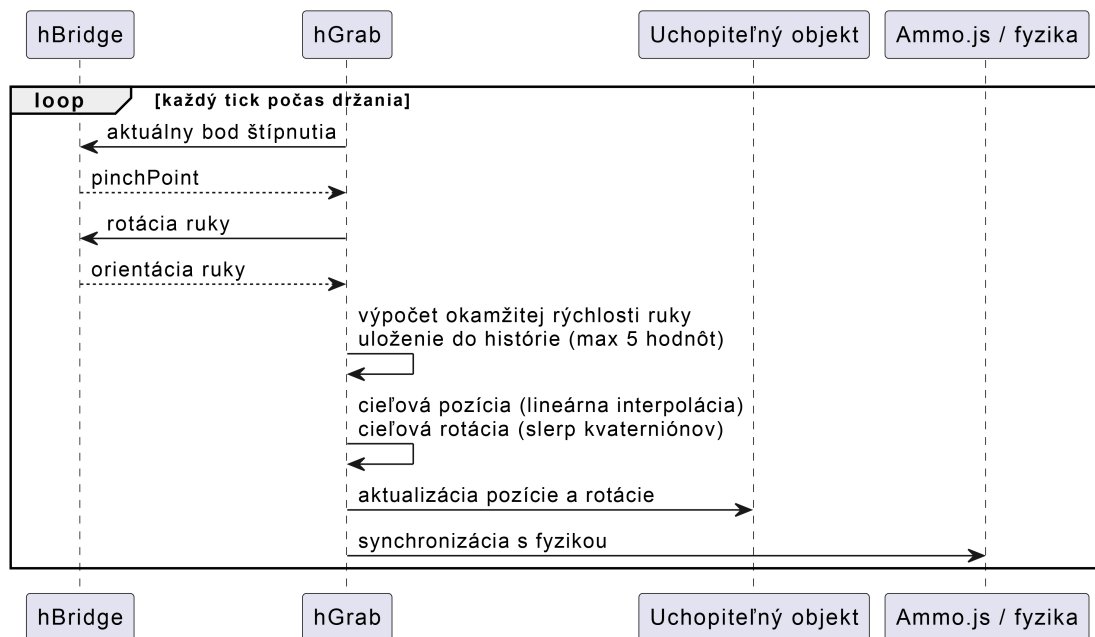


Obrázok 3.3: Sekvenčný diagram začiatku uchopenia objektu

3.2.3 Manipulácia počas držania

Počas držania objektu komponent v každom snímku priebežne aktualizuje jeho transformáciu na základe pohybu ruky (obrázok 3.4). Získava aktuálny bod štipnutia a rotáciu ruky od bridge komponentu, z ktorých odvodzuje cieľovú pozíciu a rotáciu objektu, a zároveň počíta okamžitú rýchlosť pohybu ruky. Táto rýchlosť sa ukladá do histórie, kde sa uchováva najviac päť posledných hodnôt pre neskoršie použitie pri výpočte hodu.

Pozícia objektu je aktualizovaná pomocou lineárnej interpolácie, zatiaľ čo rotácia je riadená sférickou interpoláciou kvaternionov. Tento prístup umožňuje potlačiť drobné nepresnosti sledovania ruky a zabezpečiť stabilný a plynulý pohyb. Výsledná transformácia je následne synchronizovaná s fyzikálnym systémom Ammo.js.



Obrázok 3.4: Sekvenčný diagram manipulácie s objektom počas držania

3.2.4 Pustenie objektu a hádzanie

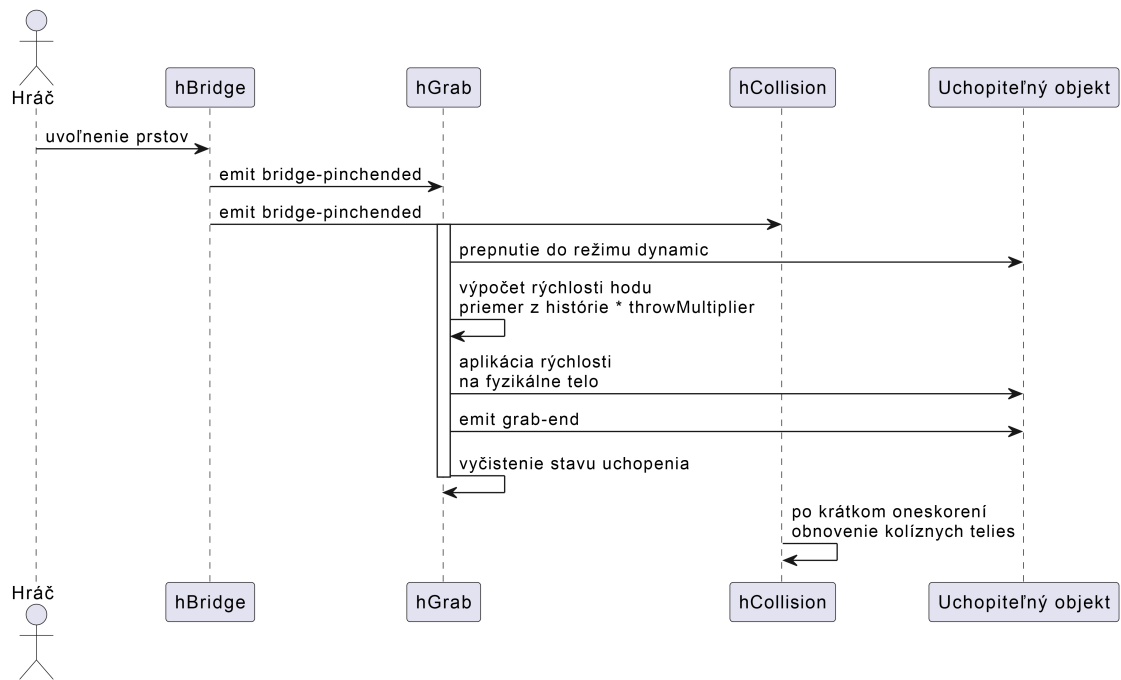
Pri prijatí udalosti `bridge-pinchended` komponent ukončí držanie objektu a prepne jeho fyzikálne telo späť do režimu `dynamic`, čím sa objekt opäť začne riadiť fyzikálnou simuláciou (obrázok 3.5). Na základe uloženej histórie rýchlostí ruky sa vypočíta výsledná rýchlosť hodu ako priemer posledných hodnôt vynásobený koeficientom `throwMultiplier`, ktorá sa následne aplikuje na fyzikálne telo objektu.

Týmto spôsobom je možné dosiahnuť prirodzené správanie pri pustení aj odhodení objektu bez potreby dodatočných špeciálnych mechanizmov. Komponent `hand-collision-bodies` po krátkom oneskorení obnoví aktivitu kolíznych telies ruky, čím sa znižuje riziko nestability fyzikálnej simulácie tesne po uvoľnení objektu.

3.3 Komponent `hand-collision-bodies`

Komponent `hand-collision-bodies` zabezpečuje fyzikálnu prítomnosť ruky v scéne. Natívny model ruky síce poskytuje informácie o polohách jednotlivých kĺbov, avšak bez samostatných kolíznych telies by ruka nedokázala fyzikálne interagovať s objektmi v prostredí.

Konfigurácia komponentu zahŕňa parametre určujúce polomer kolíznych gúľ, oneskorenie po pustení objektu, nastavenia kolíznych filtrov a voliteľné ladiace



Obrázok 3.5: Sekvenčný diagram pustenia objektu a aplikácie hodu

vizualizácie kolíznych bodov. Interne si komponent uchováva prepojenie medzi fyzickými telesami a jednotlivými kĺbmi ruky, reprezentované ako dvojice {sphere, joint}.

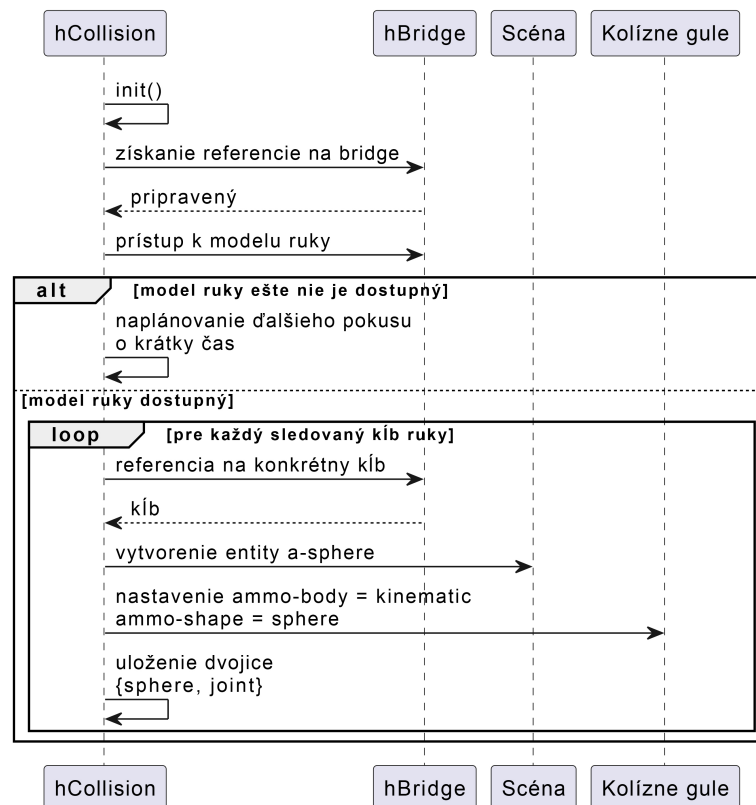
3.3.1 Inicializácia a tvorba kolíznych telies

Po inicializácii komponent najprv získa referenciu na bridge komponent a následne sa pokúsi pristúpiť k modelu ruky (obrázok 3.6). Ak model ešte nie je pripravený, vytvorenie telies sa odloží a pokus sa po krátkom čase zopakuje. V prípade úspechu komponent prejde sledovanými kĺbmi ruky a pre každý z nich vytvorí samostatnú entitu a-sphere s priradeným komponentom ammo-body v režime kinematic. Tieto entity sú vkladané priamo do scény, nie ako potomkovia entity ruky, čím sa predchádza problémom s dedením transformácií.

3.3.2 Synchronizácia a ghosting

V každom snímku komponent synchronizuje polohy kolíznych telies s aktuálnymi pozíciami kĺbov ruky a tieto zmeny prenáša do fyzikálneho systému Ammo.js (obrázok 3.7). Ak je aktívny fyzický ovládač, aktualizácia sa nevykonáva, čím sa šetria výpočtové prostriedky.

Počas držania objektu môže dochádzať k nežiaducim fyzikálnym interakciám medzi rukou a objektom. Z tohto dôvodu komponent reaguje na udalosti bridge



Obrázok 3.6: Sekvenčný diagram inicializácie komponentu `hCollision` a tvorby kolíznych telies

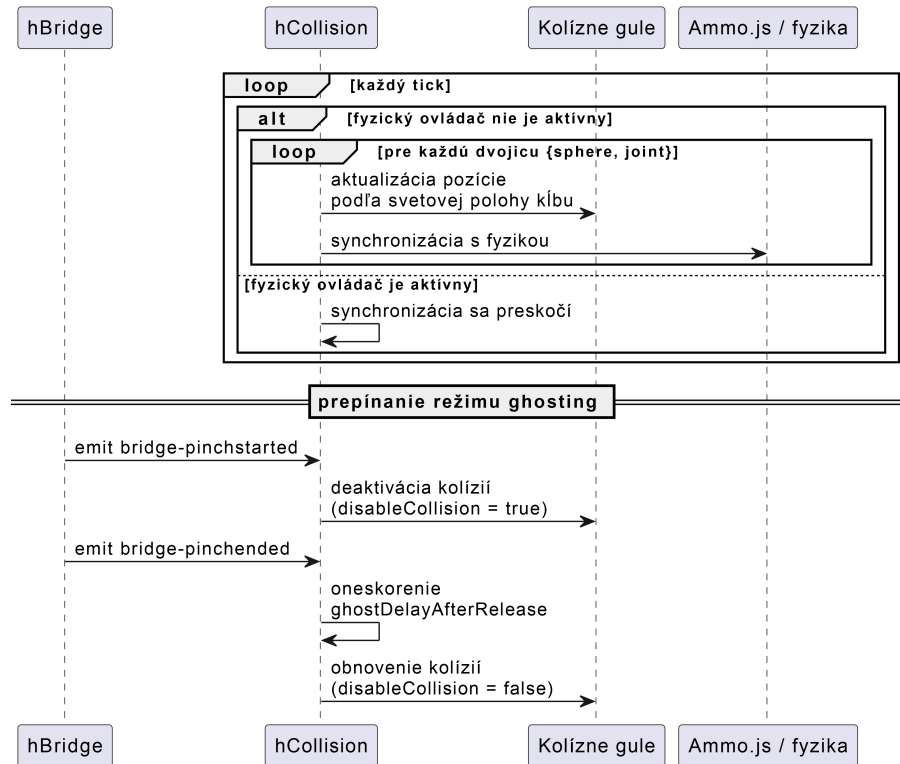
komponentu – pri začiatku štipnutia dočasne deaktivuje kolízie kolíznych telies a po uvoľnení objektu ich obnoví, no nie okamžite, ale až po krátkom oneskorení danom parametrom `ghostDelayAfterRelease`. Tento prístup znižuje riziko nestability simulácie tesne po pustení uchopeného objektu.

3.4 Komponent `hand-point-fingertip-touch`

Komponent `hand-point-fingertip-touch` slúži na dotykové ovládanie používateľského rozhrania. Umožňuje interakciu s prvkami UI prostredníctvom špičky ukazováka, čím simuluje prirodzený dotyk.

Konfigurácia komponentu zahŕňa parametre určujúce polomer detekcie dotyku, interval skenovania a jeho aktiváciu. Okrem toho si komponent uchováva referenciu na bridge komponent, naposledy dotknutý prvok a kešovaný zoznam interaktívnych elementov.

Detekcia dotyku prebieha v pravidelných intervaloch a jej priebeh je zachytený na obrázku 3.8. Komponent najprv overí, či nie je aktívny fyzický ovládač, či neprebíha teleportácia a či ruka aktuálne nedrží objekt – v ktoromkoľvek z tých-



Obrázok 3.7: Sekvenčný diagram synchronizácie kolíznych telies a prepínania režimu ghosting

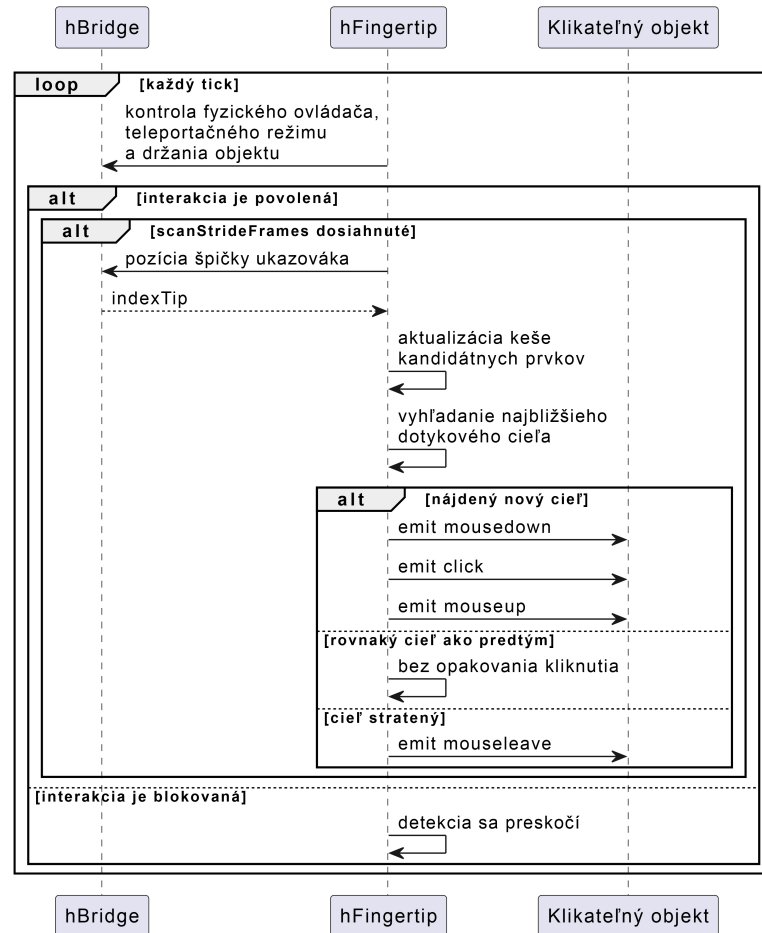
to prípadov sa detekcia preskočí. Ak je interakcia povolená, komponent získa pozíciu špičky ukazováka od bridge komponentu a porovná ju s pozíciami interaktívnych prvkov v scéne, ako sú elementy označené `[keyboard-button]`, `.collidable` alebo `.clickable`.

Pri zistení dotyku na novom prvku komponent vyšle sekvenciu udalostí, konkrétne `mousedown`, `click` a `mouseup`, čím simuluje kliknutie myšou. Ak špička zostáva nad rovnakým prvkom, opakovaný klik sa nevyvolá. Pri vzdialení sa od prvku komponent emituje udalosť `mouseleave`. Na zníženie výpočtovej náročnosti sa detekcia nevykonáva v každom snímku, ale iba v definovanom intervale, a zoznam kandidátnych prvkov je kešovaný a aktualizovaný len periodicky.

3.5 Komponent **hand-teleport**

Komponent `hand-teleport` zabezpečuje pohyb používateľa po scéne pomocou teleportácie riadenej gestami ruky. Na samotnú realizáciu teleportácie využíva komponent `blink-controls`, ktorého správanie programovo riadi na základe detegovaných gest.

Konfigurácia komponentu zahŕňa parametre určujúce aktiváciu, cieľový ovlá-



Obrázok 3.8: Sekvenčný diagram dotykovej interakcie špičkou ukazováka

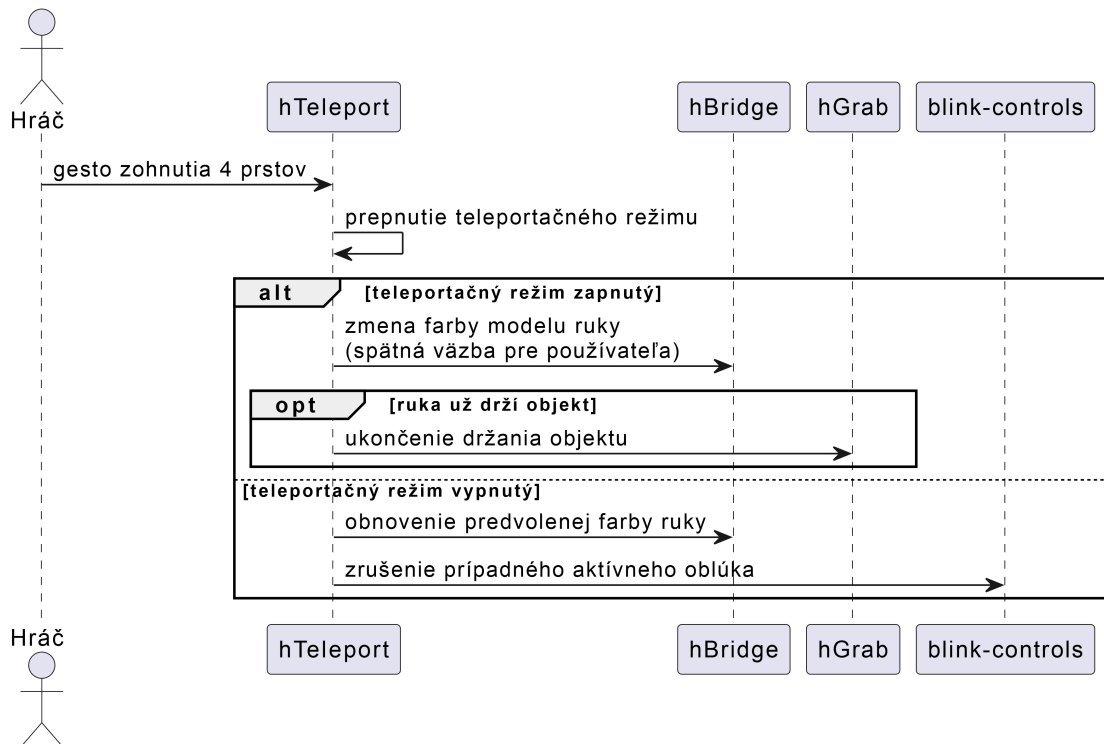
dač teleportácie, počet snímkov potrebných na potvrdenie gesta a vizuálne nastavenia. Interný stav komponentu sleduje najmä aktivitu teleportačného režimu, stav oblúka, počítadlá detekcie gest a pomocné príznaky pre riadenie interakcie.

3.5.1 Aktivácia a deaktivácia teleportačného režimu

Teleportačný režim sa zapína a vypína gestom zohnutia štyroch prstov k dlani (obrázok 3.9). Pri prvom rozpoznaní tohto gesta komponent prepne teleportačný režim do aktívneho stavu, zmení vizuálnu reprezentáciu ruky (čím poskytne používateľovi spätnú väzbu), a ak ruka v tom momente drží objekt, ukončí jeho držanie. Pri opätovnom vykonaní rovnakého gesta sa režim vypne, farba ruky sa vráti do predvoleného stavu a prípadný aktívny teleportačný oblúk sa zruší.

3.5.2 Aktivácia oblúka a potvrdenie teleportácie

Počas aktívneho teleportačného režimu komponent sleduje gesto otvorenej dlane (obrázok 3.10). Po jeho stabilnej detekcii počas dostatočného počtu snímkov



Obrázok 3.9: Sekvenčný diagram aktivácie a deaktivácie teleportačného režimu

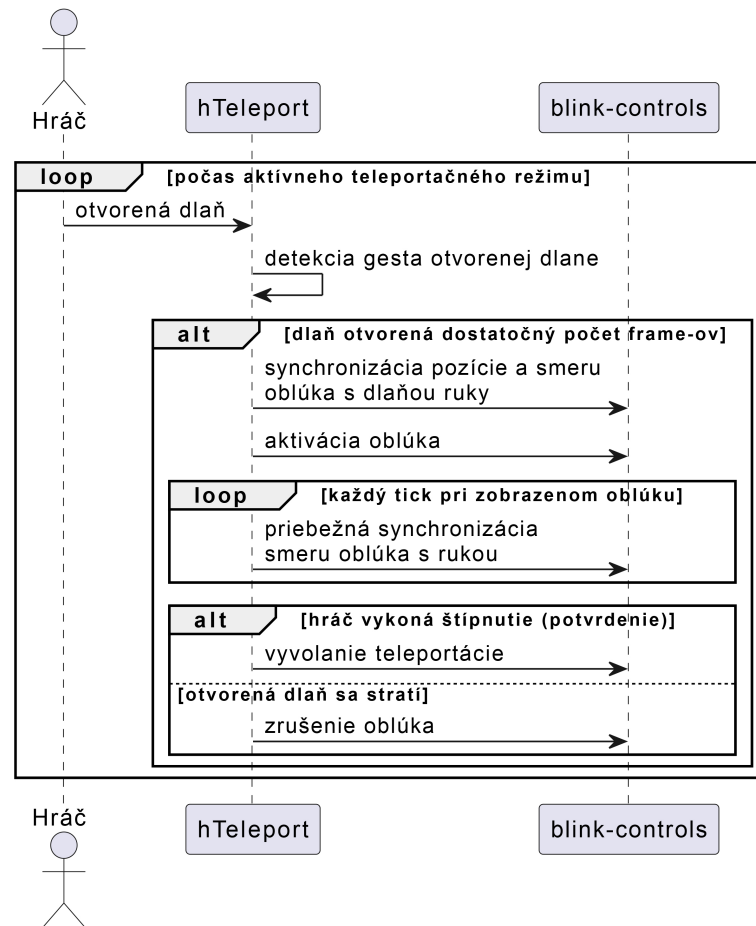
sa aktivuje teleportačný oblúk, ktorého pozícia a smer sú priebežne synchronizované s dlaňou ruky. Smer oblúka je tak intuitívne riadený orientáciou ruky používateľa.

Používateľ môže teleportáciu potvrdiť gestom štípnutia, čím sa vyvolá príslušná funkcia komponentu `blink-controls` a používateľ sa premiestni na vybrané miesto. Ak je gesto otvorenej dlane prerušené pred potvrdením, oblúk sa deaktivuje a teleportácia sa nevykoná, čím sa zabráni nechcenému presunu.

3.6 Komponent `hand-point-shoot`

Komponent `hand-point-shoot` implementuje interakciu na diaľku. Používateľ môže ukazovaním označiť vzdialený objekt a následne ho pritiahnuť k ruke.

Konfigurácia komponentu zahŕňa parametre určujúce maximálnu vzdialenosť interakcie, rýchlosť pritiahnutia objektu, aktivačnú vzdialenosť a vizualizáciu lúča. Interný stav obsahuje informácie o aktivite ukazovacieho gesta, aktuálnom ciele, priťahovanom objekte a vyhladenom smere lúča.

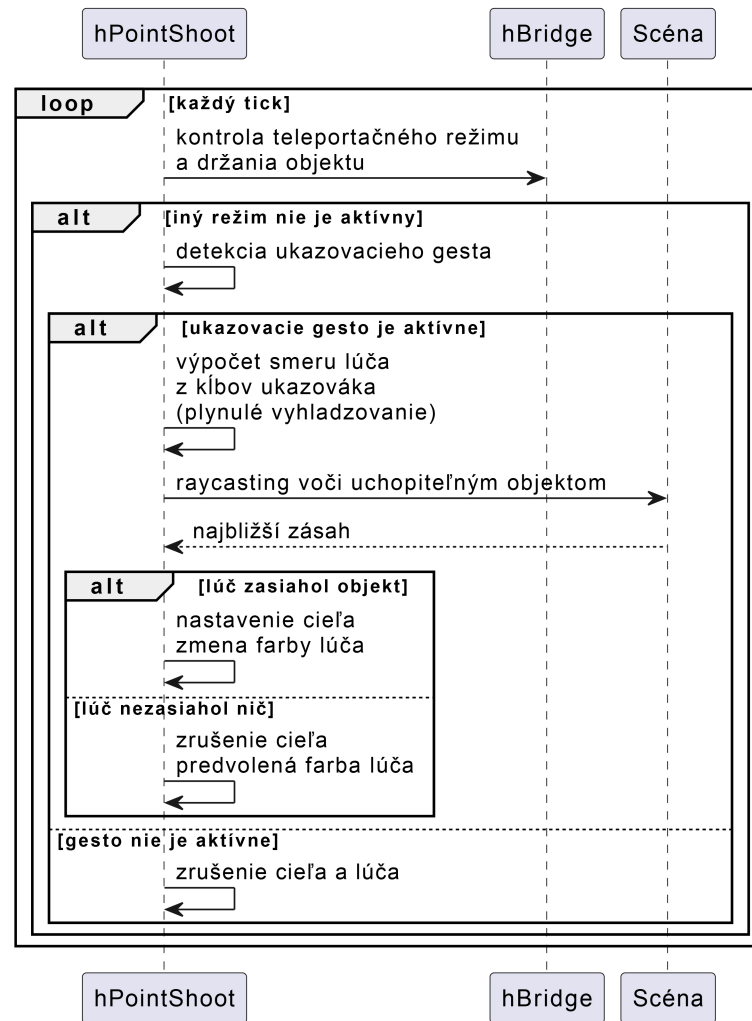


Obrázok 3.10: Sekvenčný diagram aktivácie teleportačného oblúka a potvrdenia teleportácie

3.6.1 Sledovanie cieľa lúčom

V každom snímku komponent kontroluje, či nie je aktívny iný režim interakcie – ak prebieha teleportácia alebo ruka drží objekt, sledovanie sa preskočí (obrázok 3.11). V opačnom prípade sa pokúsi rozpoznať ukazovacie gesto. Ak je gesto aktívne, komponent vypočíta smer lúča na základe orientácie kĺbov ukazováka. Tento smer je následne plynulo vyhladzovaný, aby sa obmedzili drobné nepresnosti sledovania spôsobené jemným chvením ruky.

Pomocou raycastingu sa zisťuje prienik lúča s uchopiteľnými objektmi v scéne. Ak je objekt zasiahnutý, označí sa ako aktuálny cieľ a farba lúča sa zmení, čím získava používateľ vizuálnu spätnú väzbu. Ak gesto zanikne alebo lúč nezasiahne žiadny objekt, cieľ sa zruší a lúč sa vráti do predvoleného stavu.

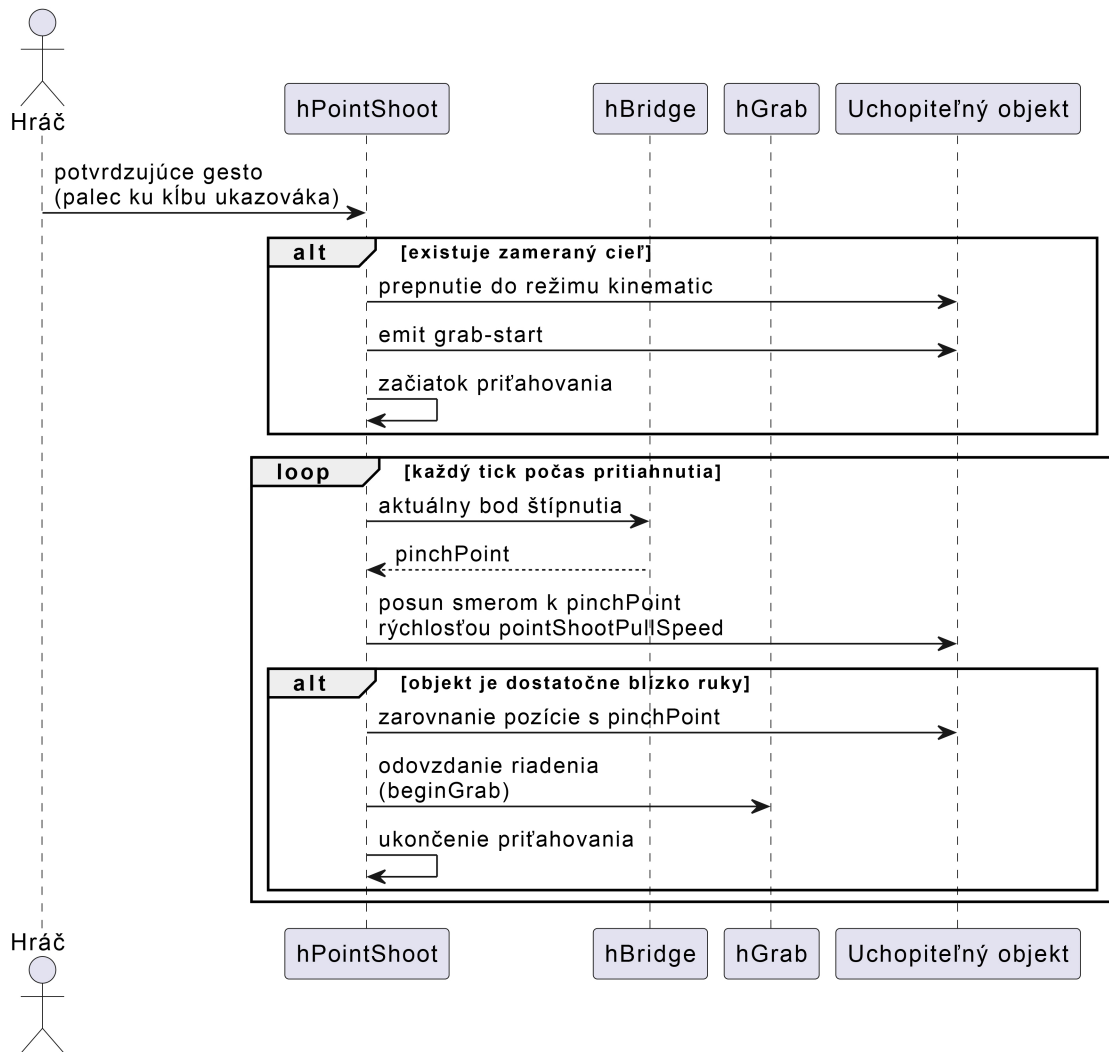


Obrázok 3.11: Sekvenčný diagram sledovania cieľa lúčom

3.6.2 Pritiahnutie objektu a odovzdanie riadenia

Po vykonaní potvrdzujúceho gesta (priblíženie palca ku kĺbu ukazováka) sa spustí pritiahnutie aktuálne zameraného objektu (obrázok 3.12). Cieľový objekt sa prepne do režimu *kinematic* a v každom snímku sa plynulo posúva smerom k bodu štípnutia ruky rýchlosťou definovanou parametrom `pointShootPullSpeed`.

Keď sa objekt dostatočne priblíži k ruke, jeho pozícia sa zarovná s bodom štípnutia a riadenie sa odovzdá komponentu `hand-grab`, ktorý objekt automaticky uchopí. Týmto spôsobom sa zabezpečí plynulý prechod z interakcie na diaľku do štandardného režimu manipulácie s objektom.



Obrázok 3.12: Sekvenčný diagram pritiažnutia objektu a odovzdania riadenia komponentu hGrab

3.7 Zhrnutie implementácie

Implementované riešenie je založené na jednom centrálnom komponente a piatich špecializovaných moduloch. Takéto rozdelenie umožňuje oddeľovať jednotlivé typy interakcií, znižuje vzájomné väzby medzi časťami systému a uľahčuje ďalší vývoj.

Komponent hand-tracking-bridge zabezpečuje jednotnú vrstvu nad natívnym hand trackingom, zatiaľ čo ostatné moduly realizujú konkrétne typy interakcie – uchopenie, fyzikálnu prítomnosť, dotyk, teleportáciu a interakciu na diaľku. Výsledkom je riešenie, ktoré je prispôbené potrebám projektu Lirki-sEduVePn a zároveň rešpektuje technické obmedzenia A-Frame, WebXR a Ammo.js.

4 Vyhodnotenie

Hlavným cieľom tejto práce je rozšírenie systému LirkisEduVePn o ovládanie pomocou snímania rúk a zároveň umožniť jeho efektívne využitie vo virtuálnych prostrediach. Súčasťou riešenia bolo navrhnuť a implementovať rôzne typy interakcií v scénach tak, aby boli pre používateľa čo najviac intuitívne, jednoduché použiteľné a zároveň reprodukovateľné aj v budúcich scénach. Zároveň bol kladený dôraz na možnosť rozšírenia systému o vlastné interakčné prvky podľa potreby.

Cieľom tejto kapitoly je vyhodnotiť navrhnuté a implementované interakčné komponenty, ktorých zámerom bolo nahradiť používanie tradičných VR ovládačov. Najskôr uvádzame priebeh testovania v reálnych podmienkach vrátane použitého technického vybavenia a testovacích scenárov. Následne sa venujeme analýze získaných výsledkov a pozorovaní z testovania.

4.1 Metóda vyhodnotenia

Vyhodnotenie systému LirkisEduVePn bolo navrhnuté ako používateľské testovanie zamerané na posúdenie použiteľnosti, kvality interakcie a subjektívneho používateľského zážitku vo virtuálnej realite.

Metodika kombinuje plnenie vopred definovaných úloh v simulovanom virtuálnom prostredí s následným zberom subjektívnych hodnotení prostredníctvom štandardizovaných dotazníkov. Tento prístup umožňuje zachytiť ako objektívne správanie používateľov pri interakcii so systémom, tak aj ich subjektívne vnímanie kvality interakcie.

Použité hodnotiace nástroje pokrývajú viaceré dimenzie používateľského zážitku:

- Škála použiteľnosti systému (System Usability Scale, SUS) – celková použiteľnosť systému,
- Index záťaže úloh NASA (NASA Task Load Index, NASA TLX) – subjek-

tívna kognitívna a fyzická záťaž,

- Dotazník nevoľnosti vo virtuálnej realite (Virtual Reality Sickness Questionnaire, **VRSQ**) – miera diskomfortu spojeného s používaním **VR**,
- Dotazník prítomnosti (Igroup Presence Questionnaire, **IPQ**) – miera pocitu prítomnosti vo virtuálnom prostredí.

Okrem kvantitatívnych údajov boli zbierané aj doplňujúce demografické informácie a odpovede na otvorené otázky, ktoré slúžia na kvalitatívnu interpretáciu výsledkov. Testovanie sa uskutočnilo na scéne zameranej na skladanie počítača, ktorá je podrobnejšie opísaná v sekcii 4.2. Z tohto dôvodu boli v rámci demografických údajov sledované aj predchádzajúce skúsenosti účastníkov so skladaním počítača.

4.2 Pribeh testovania

Testovanie prebiehalo medzi študentmi z internátov Technickej univerzity v Košiciach. Systém bol spustený na notebooku, pričom **VR** náhlavná súprava bola priamo pripojená k lokálnej sieti. Na testovanie boli použité **VR** okuliare Oculus Quest 3S s podporou sledovania rúk bez použitia fyzických ovládačov počas vykonávania úloh.

Každý účastník testovania a dotazníka potvrdil svoju účasť podpisom informovaného súhlasu. Následne účastníci postupne vykonávali úlohy v systéme pomocou vopred pripravených testovacích účtov. Každému účastníkovi bolo pridelené identifikačné číslo, ktoré neskôr uvádzal v dotazníku. Testovanie prebiehalo individuálne, takže účastníci nemali možnosť vidieť interakciu ostatných so systémom ani spôsob riešenia úloh pred vlastným testovaním. Každému účastníkovi boli vysvetlené ciele úloh bez poskytnutia podrobných krokových inštrukcií, pričom im bolo zároveň ukázané, ako používať ruky na interakciu s virtuálnym prostredím.

Na testovanie bola použitá scéna Michala Sakáča [12] zameraná na skladanie počítača, ktorá obsahovala dve hlavné úlohy a viacero interaktívnych objektov. Tieto objekty sa líšili veľkosťou a charakterom interakcie, čo umožnilo overiť, či je manipulácia s objektmi rôznych rozmerov pre používateľa intuitívna a pohodlná, ako aj či je komfortné ich pripájanie a odpájanie. Pred testovaním na vybranej scéne bolo rozhodnuté nastaviť hodnotu `grabRadius` komponentu `hand-grab` približne na 0,15, čo zodpovedá približne 15 herným centimetrom. Toto rozhodnutie bolo urobené s cieľom zlepšiť interakciu s objektmi v obmedzenom priestore

počas skladania počítača. Pri väčšej hodnote by uchopovanie predmetov umiestnených blízko seba mohlo viesť k nereálnemu správaniu, zatiaľ čo pri menšej hodnote by hráč musel vynaložiť väčšie úsilie na interakciu s prvkami scény.

Voliteľne mohli účastníci navštíviť aj druhú scénu – ShowCase Room od Kristiána Zsigóa [14]. Táto scéna však mohla spôsobovať dlhšie načítavanie alebo nestabilitu systému, preto ju neabsolvovali všetci účastníci.

Po úspešnom dokončení úloh mali účastníci možnosť voľne preskúmať virtuálne prostredie, vyskúšať interakcie mimo definovaných úloh a taktiež si vyskúšať používanie ovládačov bez opustenia systému. Pre mnohých z nich išlo o prvú skúsenosť s VR zariadením, preto bolo dôležité umožniť aj porovnanie interakcie pomocou rúk a ovládačov.

4.3 Výsledky a interpretácia

Testovania sa zúčastnilo celkovo 15 účastníkov mužského pohlavia vo veku 17 až 20 rokov. Väčšina pochádzala z technických odborov (IT, kyberbezpečnosť, elektrotechnika). Predchádzajúce skúsenosti účastníkov sú uvedené v tabuľke 4.1.

Tabuľka 4.1: Predchádzajúce skúsenosti účastníkov

Skúsenosť	Áno	Nie
Virtuálna realita	6 (40 %)	9 (60 %)
Sledovanie rúk	3 (20 %)	12 (80 %)
Skladanie počítača	7 (47 %)	8 (53 %)

4.3.1 System Usability Scale (SUS)

SUS hodnotí celkovú použiteľnosť systému. Výsledné skóre sa vypočíta zo všetkých desiatich odpovedí, pričom pre kladne formulované otázky (1, 3, 5, 7, 9) sa od odpovede odčíta 1 a pre záporne formulované otázky (2, 4, 6, 8, 10) sa od odpovede odčíta od 5. Súčet týchto hodnôt sa následne vynásobí číslom 2,5, čím sa získa výsledné skóre na škále 0 až 100. Hodnoty nad 70 sa všeobecne považujú za dobré, hodnoty nad 80 za vynikajúce.

Systém dosiahol priemerné skóre **82,3 bodov**, čo zodpovedá hodnoteniu *vynikajúci*. Priemerné odpovede na jednotlivé otázky sú uvedené v tabuľke 4.2.

Účastníci hodnotili systém ako ľahko použiteľný a cítili sa pri ňom sebaisto. Miernejšie hodnotenie pri otázke o technickej podpore naznačuje, že niektorí

Tabuľka 4.2: Výsledky dotazníka SUS

Otázka	Priemer
Myslím si, že by som rád(a) používal(a) tento systém často.	4,00
Systém mi prišiel zbytočne komplikovaný.	1,67
Systém sa mi zdal ľahko použiteľný.	4,40
Potreboval(a) by som technickú podporu na používanie systému.	2,47
Funkcionality systému sú dobre zapracované.	4,27
V systéme je priveľa nekonzistentností.	1,47
Väčšina ľudí by sa systém naučila veľmi rýchlo používať.	4,33
Tento systém mi príde ťažkopádny na použitie.	1,20
Pri používaní systému som sa cítil(a) sebaisto.	4,60
Musel(a) som sa naučiť veľa vecí pred začiatkom práce.	1,87
Celkové skóre SUS	82,3

účastníci by pri prvom použití uvítali krátke zaškolenie, čo je pochopiteľné vzhľadom na to, že 80 % z nich nemalo predchádzajúcu skúsenosť so sledovaním rúk.

4.3.2 NASA Task Load Index (NASA TLX)

NASA TLX meria, ako náročné bolo pre účastníkov plnenie úloh. Hodnotí šesť dimenzií na škále 1 až 10, pričom päť z nich (mentálne zaťaženie, fyzické zaťaženie, časový nátlak, úsilie, frustrácia) sú formulované tak, že nižšia hodnota znamená lepší výsledok, zatiaľ čo dimenzia výkon je formulovaná opačne – vyššia hodnota znamená lepší výsledok. Priemerné hodnoty jednotlivých dimenzií sú uvedené v tabuľke 4.3.

Tabuľka 4.3: Výsledky dotazníka NASA TLX

Dimenzia	Priemer
Mentálne zaťaženie	2,53
Fyzické zaťaženie	2,67
Časový nátlak	2,27
Výkon	8,73
Úsilie	2,67
Frustrácia	1,47

Výsledky ukazujú, že záťaž pri plnení úloh bola celkovo nízka. Výkon bol hodnotený veľmi vysoko (priemer = 8,73), čo znamená, že účastníci považovali svoje plnenie úloh za úspešné. Frustrácia bola minimálna (priemer = 1,47), čo potvrdzuje, že gestové ovládanie nevyvolávalo zbytočný stres. Mierne vyššia hodnota fyzického zaťaženia (priemer = 2,67) môže súvisieť s tým, že pri ovládaní rukami musia účastníci držať ruky pred sebou v zornom poli kamier náhlavnej súpravy, čo je dlhodobo náročnejšie ako držanie ovládačov.

4.3.3 Virtual Reality Sickness Questionnaire (VRSQ)

VRSQ meria, či účastníci pociťovali počas používania VR nejaké fyzické ťažkosti (závraty, nevoľnosť, únava očí a podobne). Škála je od 1 do 4, pričom 1 znamená žiadny príznak. Výsledky sú uvedené v tabuľke 4.4.

Tabuľka 4.4: Výsledky dotazníka VRSQ

Príznak	Priemer
Všeobecný diskomfort	1,40
Únava	1,53
Námaha očí	1,93
Ťažkosti so zaostrením	1,40
Bolesť hlavy	1,27
Tlak v hlave	1,53
Rozmazané videnie	2,13
Závraty pri zatvorených očiach	1,13
Vertigo (točenie hlavy)	1,07

Väčšina príznakov bola hodnotená blízko minima škály. Mierne vyššia hodnota pri rozmazanom videní (priemer = 2,13) môže súvisieť s dlhším pobytom v náhlavnej súprave alebo obmedzeným priestorom pri testovaní. Niektorí účastníci v otvorených komentároch uviedli, že obraz im pripadal mierne rozmazaný, čo môže prispievať k námahe očí. Celkovo výsledky potvrdzujú, že systém nespôsoboval výraznú nevoľnosť.

4.3.4 Igroup Presence Questionnaire (IPQ)

IPQ meria, ako veľmi mal účastník pocit, že sa skutočne nachádza vo virtuálnom prostredí. Jednotlivé položky sa hodnotia na sedembodovej škále od 0 do 6, pričom vyššia hodnota znamená silnejší pocit prítomnosti. Otázky sú rozdelené do

troch sub-škál: *Priestorová prítomnosť* (SP) – pocit fyzického „bytia tam“, *Zapojenie* (INV) – miera odpojenia od reálneho sveta, a *Realizmus* (REAL) – podobnosť VR s reálnym svetom. Súhrnné výsledky pre jednotlivé sub-škály sú uvedené v tabuľke 4.5.

Tabuľka 4.5: Súhrnné výsledky dotazníka IPQ

Sub-škála	Priemer
Priestorová prítomnosť (SP)	4,14
Zapojenie (INV)	3,13
Realizmus (REAL)	3,15

Pocit priestorovej prítomnosti bol vysoký (priemer = 4,14) – účastníci mali silný dojem, že sa fyzicky nachádzajú vo virtuálnom svete. Zapojenie dosiahlo strednú hodnotu, čo naznačuje, že účastníci si počas používania systému stále čiastočne uvedomovali okolitý reálny svet. Realizmus dosiahol nižšiu hodnotu, čo je očakávané pre vzdelávacie VR aplikácie tohto typu, kde cieľom nie je dokonalá ilúzia reality, ale funkčná a príjemná interakcia. Najvyššie individuálne hodnotenie získali otázky týkajúce sa pochopenia fungovania gest (priemer = 4,53) a pocitu prítomnosti vo virtuálnom svete (priemer = 4,27).

4.3.5 Hodnotenie gestovej interakcie

Na záver účastníci hodnotili samotné gestové ovládanie v piatich oblastiach na škále 1 až 5. Výsledky sú uvedené v tabuľke 4.6.

Tabuľka 4.6: Hodnotenie gestovej interakcie

Otázka	Priemer
Celkové hodnotenie použitých gest.	4,27
Zrozumiteľnosť fungovania gest.	4,53
Jednoduchosť používania gest.	3,73
Prirodzenosť a pohodlnosť gest.	4,07
Presnosť a spoľahlivosť reakcie gest na pohyby.	3,87
Celkový priemer	4,09

Gestová interakcia bola hodnotená pozitívne vo všetkých oblastiach. Najvyššie bola zrozumiteľnosť fungovania gest (priemer = 4,53), čo naznačuje, že účastníci rýchlo pochopili, ako gestá fungujú. Najnižšie skóre dosiahla jednoduchosť

používania (priemer = 3,73), čo poukazuje na priestor pre ďalšie zlepšenie detekcie pohybov rúk. Tento záver potvrdzujú aj komentáre účastníkov v otvorených otázkach, kde niekoľkí spomenuli, že niektoré gestá občas nereagovali presne podľa očakávania, najmä pri ukazovaní na vzdialené predmety.

4.4 Zhrnutie vyhodnotenia

Testovanie potvrdilo, že implementované gestové ovládanie je funkčné a použiteľné pre vzdelávacie VR prostredie. Systém dosiahol vynikajúce hodnotenie použiteľnosti (SUS = 82,3), účastníci plnili úlohy úspešne a bez výraznej frustrácie. Nevoľnosť spojená s používaním VR bola minimálna a pocit prítomnosti vo virtuálnom prostredí bol silný, najmä z pohľadu priestorovej prítomnosti (IPQ SP = 4,14). Gestová interakcia bola hodnotená pozitívne (priemer 4,09 z 5), pričom najvyššia bola zrozumiteľnosť gest a najnižšia jednoduchosť ich používania.

Hlavným priestorom na zlepšenie zostáva spoľahlivosť rozpoznávania pohybov rúk, ktorú účastníci označili za najslabšiu stránku systému. Túto vlastnosť však nie je možné výrazne zlepšiť na úrovni implementácie, keďže závisí najmä od kvality kamier a algoritmov sledovania rúk v samotnej VR náhlavnej súprave.

5 Záver

Cieľom tejto bakalárskej práce bolo rozšíriť existujúci systém virtuálnych výučbových prostredí LirkisEduVePn o ovládanie pomocou snímania rúk, ktoré by nahradilo tradičné VR ovládače a umožnilo prirodzenejšiu interakciu vo virtuálnom prostredí.

V analytickej časti boli preskúmané existujúce možnosti gestového ovládania v prostredí A-Frame, vrátane natívneho sledovania rúk cez WebXR Hand Tracking API, knižnice Super Hands a vlastných riešení. Analýza odhalila viaceré technické konflikty medzi existujúcimi komponentmi systému a potvrdila, že najvhodnejším prístupom je vlastná modulárna implementácia.

Na základe analýzy bol navrhnutý a implementovaný systém pozostávajúci z centrálnemu komponentu hand-tracking-bridge a piatich špecializovaných interakčných modulov pokrývajúcich uchopenie objektov, fyzikálnu prítomnosť ruky v scéne, dotykovú interakciu, teleportáciu a interakciu na diaľku. Systém bol integrovaný do existujúcej architektúry projektu a zachováva kompatibilitu s Petriho sieťami aj s fyzickými VR ovládačmi.

Funkčnosť riešenia bola overená používateľským testovaním s 15 účastníkmi na scéne skladania počítača. Výsledky dotazníkov potvrdili vynikajúcu použiteľnosť systému ($SUS = 82,3$), nízku záťaž pri plnení úloh a silný pocit prítomnosti vo virtuálnom prostredí ($IPQ\ SP = 4,14$). Gestová interakcia bola účastníkmi hodnotená pozitívne, s priemerným skóre 4,09 z 5.

Hlavným otvoreným problémom zostáva presnosť detekcie pohybov rúk, ktorá závisí od kvality kamier headsetu a podmienok osvetlenia. Túto vlastnosť nemožno výrazne zlepšiť na úrovni samotnej implementácie, môže sa však postupne zlepšovať s nástupom novších generácií VR zariadení.

Modulárna architektúra navrhnutého riešenia zároveň otvára priestor pre ďalšie rozšírenia. Vďaka centrálnemu komponentu hand-tracking-bridge je možné jednotlivé interakčné moduly nezávisle pridávať, upravovať alebo nahrádzať bez zásahu do jadra systému. Do budúca by bolo vhodné doplniť nové typy interakcií pre špecifické výučbové scenáre, implementovať mechanizmus selek-

tívneho blokovania nežiaducich gest podľa aktuálneho stavu scenára, otestovať riešenie na väčšej a rozmanitejšej vzorke účastníkov a porovnať efektivitu výučby pri použití gestového ovládania oproti tradičným VR ovládačom v rovnakých scenároch.

Zoznam použitej literatúry

1. SATHYA, Thangavel. Revolutionizing Education Through Augmented Reality (AR) and Virtual Reality (VR): Innovations, Challenges and Future Prospects. *Challenges and Future Prospects (March 03, 2025)* [online]. 2025 [cit. 2025-10-30]. Dostupné z: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=5195697.
2. LAMPROPOULOS, Georgios; BOSQUE, Antonio del; FERNÁNDEZ-ARIAS, Pablo; VERGARA, Diego. Virtual Reality in Medical Education, Healthcare Education, and Nursing Education: An Overview. *Multimodal Technologies and Interaction* [online]. 2025, roč. 9, č. 7, s. 75 [cit. 2025-10-30]. Dostupné z: <https://www.mdpi.com/2414-4088/9/7/75>.
3. PAULSEN, Lucas; DAU, Susanne; DAVIDSEN, Jacob. Designing for collaborative learning in immersive virtual reality: A systematic literature review. *Virtual Reality* [online]. 2024, roč. 28, č. 1, s. 63 [cit. 2025-11-01]. Dostupné z: <https://link.springer.com/article/10.1007/s10055-024-00975-4>.
4. PENG, Ruinan; AN, Hongbo; JIANG, Yida. A review of gesture interaction technology based on virtual reality. *Available at SSRN 4359350* [online]. 2023 [cit. 2025-11-01]. Dostupné z: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4359350.
5. HERBERT, Oswaldo Mendoza; PÉREZ-GRANADOS, David; RUIZ, Mauricio Alberto Ortega; CADENA MARTÍNEZ, Rodrigo; GUTIÉRREZ, Carlos Alberto González; ANTUÑANO, Marco Antonio Zamora. Static and dynamic hand gestures: a review of techniques of virtual reality manipulation. *Sensors* [online]. 2024, roč. 24, č. 12, s. 3760 [cit. 2025-11-01]. Dostupné z: <https://www.mdpi.com/1424-8220/24/12/3760>.
6. KAŠELA, Adam. *Výučbové scenáre v rozšírenej realite využívajúce procesné grafy*. Košice, 2022. Diplomová práca. Technická univerzita v Košiciach, Fakulta elektrotechniky a informatiky. Vedúci práce: Ing. Štefan Korečko, PhD.

7. REISIG, Wolfgang. *Understanding petri nets* [online]. Springer, 2016 [cit. 2026-05-04]. Dostupné z: <https://link.springer.com/content/pdf/10.1007/978-3-642-33278-4.pdf>.
8. DEMIANENKO, Dmytro. *Spracovanie údajov z virtuálnych výučbových prostredí*. Košice, 2023. Bakalárska práca. Technická univerzita v Košiciach, Fakulta elektrotechniky a informatiky. Vedúci práce: Ing. Štefan Korečko, PhD.
9. PISARČÍK, Lukáš. *Scenárom riadené výučbové prostredia vo virtuálnej realite*. Košice, 2023. Bakalárska práca. Technická univerzita v Košiciach, Fakulta elektrotechniky a informatiky. Vedúci práce: Ing. Štefan Korečko, PhD.
10. GOREGAOKAR, Manish; CABANIER, Rik. *WebXR Hand Input Module - Level 1* [online]. 2024-06. [cit. 2026-01-28]. W3C Working Draft. World Wide Web Consortium (W3C). Dostupné z: <https://www.w3.org/TR/webxr-hand-input-1/>.
11. RUDY, Juraj. *Systém virtuálnych výučbových prostredí riadených procesnými grafmi*. Košice, 2024. Diplomová práca. Technická univerzita v Košiciach, Fakulta elektrotechniky a informatiky. Vedúci práce: Ing. Štefan Korečko, PhD.
12. SAKÁČ, Michal. *Virtuálne výučbové prostredie pre skladanie počítačov*. Košice, 2025. Bakalárska práca. Technická univerzita v Košiciach, Fakulta elektrotechniky a informatiky. Vedúci práce: Ing. Štefan Korečko, PhD.
13. KOLOSOVSKYI, Dmytro. *Overenie funkcionality a refaktORIZÁCIA systému virtuálnych výučbových prostredí*. Košice, 2025. Bakalárska práca. Technická univerzita v Košiciach, Fakulta elektrotechniky a informatiky. Vedúci práce: Ing. Štefan Korečko, PhD.
14. ZSIGÓ, Kristián. *Rozšírenia a ukázková scéna pre systém virtuálnych výučbových prostredí*. Košice, 2025. Diplomová práca. Technická univerzita v Košiciach, Fakulta elektrotechniky a informatiky. Vedúci práce: Ing. Štefan Korečko, PhD.
15. JONES, Brandon; GOREGAOKAR, Manish; CABANIER, Rik. *WebXR Device API* [online]. 2024-10. [cit. 2026-01-28]. W3C Recommendation. World Wide Web Consortium (W3C). Dostupné z: <https://www.w3.org/TR/webxr/>.

Zoznam skratiek

IPQ Igroup Presence Questionnaire – dotazník prítomnosti.

NASA TLX NASA Task Load Index – index záťaže úloh.

SUS System Usability Scale – škála použiteľnosti systému.

VR Virtual Reality – virtuálna realita.

VRSQ Virtual Reality Sickness Questionnaire – dotazník nevoľnosti vo virtuálnej realite.

Zoznam príloh

Príloha A Používateľská príručka

Príloha B Systémová príručka

Príloha C Zdrojové kódy výsledného projektu sú dostupné na adrese: <https://git.kpi.fei.tuke.sk/skor-lirkis-projects/LirkisEduVePn>

Príloha D Videozáznamy ovládania rukami sú dostupné na adrese: [Google Drive](#) alebo <https://bit.ly/lirkis-videos>

Príloha E CD médium – záverečná práca v elektronickej podobe:

- doc – záverečná práca a prílohy v čitateľnom formáte
- tex – zdrojové súbory záverečnej práce a výsledky testovania v priečinku evaluation
- lib – dokumentácia projektu
- bin – docker obrazy klientskej a serverovej časti exportované do TAR archívov
- src – zdrojové kódy vytvoreného riešenia projektu

A Používateľská príručka

Táto používateľská príručka nadväzuje na príručky priložené k predchádzajúcim prácam rozširujúcim systém LirkisEduVePn [8, 9, 11, 12, 14]. Nová časť sa týka výlučne ovládania pomocou snímania rúk. Inštalácia, konfigurácia systému, prihlasovacia obrazovka, hlavná obrazovka žiaka, administračný panel (angl. dashboard) pre učiteľov a administrátorov a ostatné scény zostali v rovnakej podobe ako v predchádzajúcej verzii systému.

A.1 Funkcia programu

Práca rozširuje systém LirkisEduVePn o ovládanie snímaním rúk, ktoré nahrádza tradičné fyzické ovládače virtuálnej reality. Používateľ tak môže s virtuálnym prostredím komunikovať priamo rukami, bez špecializovaného príslušenstva, pričom kompatibilita s existujúcimi Petriho sieťami, správou scenárov aj s fyzickými VR ovládačmi zostáva zachovaná.

Nová funkcionálna pokrýva päť typov interakcie – uchopenie objektov a ich manipuláciu vrátane hádzania, fyzikálnu prítomnosť ruky v scéne pomocou kolíznych telies, dotykovú interakciu s prvkami rozhrania špičkou ukazováka, teleportáciu ovládanú gestami a interakciu na diaľku s možnosťou pritiahnuť vzdialený objekt.

A.2 Požiadavky a spustenie

Pred použitím systému je potrebné overiť, či cieľové zariadenie spĺňa technické požiadavky uvedené v tejto sekcii, a následne vykonať postup spustenia popísaný v ďalších podsekciiach.

A.2.1 Požiadavky na technické prostriedky

Sledovanie rúk funguje cez vstavané kamery VR headsetu. Minimálne hardvérové požiadavky pre WebGL zostávajú oproti predchádzajúcej verzii systému nezmenené – 64-bitový operačný systém, aspoň 4 GB pamäte RAM, integrovaná grafika s podporou WebGL 2.0 / OpenGL ES 3.0, dvojjadrový procesor Intel/AMD a aktuálna verzia prehliadača Chrome, Firefox alebo Opera.

Pre samotné sledovanie rúk je navyše potrebný VR headset s podporou WebXR Hand Tracking API (testované na Meta Quest 3S) a prístup k sieti VPN TUKE alebo Wi-Fi TUKE. Fyzické VR ovládače povinné nie sú – systém funguje plne bez nich. Ak sú pri použití pripojené a aktívne, majú automaticky prednosť pred sledovaním rúk.

Pre lokálne spustenie projektu je okrem toho potrebný nainštalovaný nástroj Docker¹⁷, aspoň 10 GB voľného miesta na disku a pri prvom spustení pripojenie k internetu na stiahnutie obrazov kontajnerov. VR headset musí byť pripojený do rovnakej lokálnej siete ako počítač, na ktorom aplikácia beží; IP adresu počítača možno zistiť príkazom `ipconfig` (Windows) alebo `ip addr` (Linux).

A.2.2 Spustenie projektu lokálne

Postup inštalácie a spustenia systému zostal nezmenený oproti predchádzajúcim verziám [12, 14]. Pre úplnosť uvádzame základný postup nasadenia pomocou nástroja Docker.

V prázdnom priečinku vytvorte konfiguračný súbor `docker-compose.yml`, ktorý definuje štyri služby systému – databázu PostgreSQL, nástroj pgAdmin, server Spring Boot a aplikáciu Angular. Príklad obsahu je vo výpise A.1.

Výpis A.1: Príklad súboru `docker-compose.yml`

```
name: "lirkiseduvepn"
services:

  db-postgres:
    image: postgres:15.1
    container_name: lirkis-database
    ports:
      - "15432:5432"
    volumes:
      - lirkis-data:/var/lib/postgresql/data
```

¹⁷Docker je dostupný na <https://www.docker.com/>

environment:

- POSTGRES_DB=postgres
- POSTGRES_USER=postgres
- POSTGRES_PASSWORD=postgres

pgadmin:

image: dpage/pgadmin4:7.1

container_name: lirkis-pgadmin

ports:

- "3000:80"

volumes:

- lirkis-pgadmin-data:/var/lib/pgadmin
- ./servers.json:/pgadmin4/servers.json

environment:

- PGADMIN_DEFAULT_EMAIL=admin@admin.com
- PGADMIN_DEFAULT_PASSWORD=admin
- PGADMIN_REPLACE_SERVERS_ON_STARTUP=True

depends_on:

- db-postgres

spring:

build: ./LirkisEduVePn-service

container_name: lirkis-service

ports:

- "8443:8443"

environment:

- LIRKIS_BACKEND_URL=https://localhost:8443
- LIRKIS_FRONTEND_URL=https://localhost:4200

depends_on:

- db-postgres

angular:

build: ./LirkisEduVePn-UI

container_name: lirkis-ui

ports:

- "4200:443"

environment:

- LIRKIS_BACKEND_URL=https://localhost:8443

depends_on:

- spring

volumes:

lirkis-data:

lirkis-pgadmin-data:

Konfigurácia URL adries

Premenné prostredia LIRKIS_BACKEND_URL ako aj LIRKIS_FRONTEND_URL určujú, na akých adresách budú serverová časť a klientská časť dostupné, a používajú sa najmä na konfiguráciu CORS. Pri lokálnom spustení (na tom istom počítači, kde beží Docker) možno ponechať predvolené hodnoty s localhost. Pri nasadení na vzdialený server alebo pri sprístupnení aplikácie z **VR** headsetu z lokálnej siete je potrebné localhost nahradiť skutočnou IP adresou alebo doménovým menom servera – v opačnom prípade prehliadač zablokuje komunikáciu medzi klientskou a serverovou časťou. Premennú adresy serverovej časti LIRKIS_BACKEND_URL treba zmeniť na oboch miestach (v službe spring aj angular).

Voliteľná predkonfigurácia pgAdmin

Vyššie uvedený docker-compose.yaml pripája do kontajnera pgadmin aj súbor servers.json z aktuálneho priečinka. Tento súbor je voliteľný – jeho zmyslom je automaticky pridať pripojenie k databáze do pgAdminu pri prvom spustení, aby ho používateľ nemusel pridávať ručne. Ak ho nepoužijete, riadok s pripojením ./servers.json:/pgadmin4/servers.json v sekcii volumes služby pgadmin odstráňte (aj premennú PGADMIN_REPLACE_SERVERS_ON_STARTUP), pgAdmin sa potom spustí s prázdny zoznamom serverov a databázu doň pridáte cez používateľské rozhranie. V prípade záujmu o automatickú predkonfiguráciu vytvorte v tom istom priečinku ako docker-compose.yaml aj súbor servers.json s obsahom podľa výpisu **A.2**.

Výpis A.2: Voliteľný súbor servers.json pre predkonfiguráciu pgAdmin

```
{
  "Servers": {
    "1": {
      "Name": "Local",
      "Group": "Servers",
      "Host": "db-postgres",
      "Port": 5432,
```

```

    "MaintenanceDB": "postgres",
    "Username": "postgres",
    "SSLMode": "prefer",
    "SSLCert": "<STORAGE_DIR>/.postgresql/postgresql.crt",
    "SSLKey": "<STORAGE_DIR>/.postgresql/postgresql.key",
    "SSLCompression": 0,
    "Timeout": 10,
    "UseSSHTunnel": 0,
    "TunnelPort": "22",
    "TunnelAuthentication": 0
  }
}
}

```

Po vytvorení potrebných súborov spustíte systém príkazom `docker compose up` na systémoch Windows, resp. `docker-compose up` na systémoch Linux. Po úspešnom spustení je aplikácia dostupná v prehliadači na adrese <http://localhost:4200>. Pre prístup z VR headsetu zadajte v prehliadači headsetu adresu `http://<IP adresa počítača>:4200`, pričom headset musí byť pripojený do rovnakej lokálnej siete.

A.2.3 Prvé prihlásenie a konfigurácia

Pri prvom spustení je potrebné manuálne vytvoriť administrátorský účet. Otvorte nástroj pgAdmin na adrese <http://localhost:3000> a prihláste sa údajmi z `docker-compose.yml`. Po prihlásení otvorte databázu a v nástroji Query Tool, spustíte SQL dotaz z výpisu A.3.

Výpis A.3: SQL dotaz na pridanie administrátorského účtu

```

INSERT INTO users(
    email, firstname, is_enabled,
    lastname, username, password, role
)
VALUES (
    'admin@admin', 'admin', true, 'admin', 'admin',
    '$2a$12$3HMoKSwt7wbRayzLmoApieJTfvP0oqtc5ZTYmsPyo4PSJoy9TeDmW'
    ,
    'ADMIN'
);

```

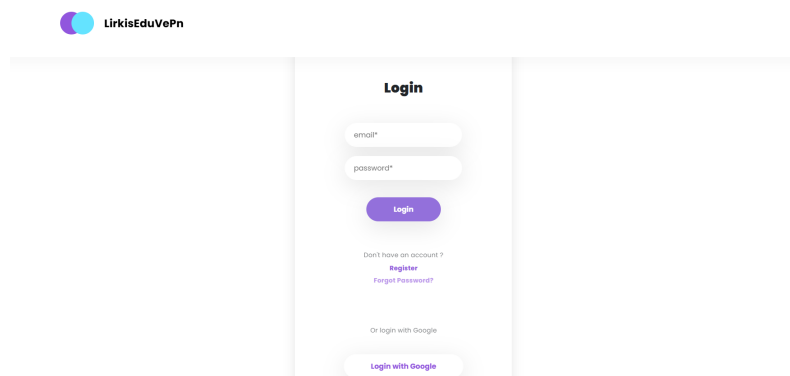
Po vykonaní dotazu sa do aplikácie prihláste menom `admin@admin` a heslom `admin`. Heslo odporúčame po prvom prihlásení zmeniť. Bežní používatelia sa môžu registrovať samostatne cez registračný formulár v aplikácii.

A.3 Použitie programu

Program ponúka používateľom plnenie úloh vo virtuálnom prostredí podľa scenárov riadených Petriho sieťami a učiteľom či administrátorom poskytuje nástroje na správu týchto scenárov, úloh a používateľov. V tejto sekcii popisujeme prihlasovaciu obrazovku, hlavnú obrazovku žiaka a administračný panel, ktoré vo vzhľade ani funkcionalite neboli touto prácou menené. Podrobnejší popis nájdete v príručkách priložených k predchádzajúcim prácam.

A.3.1 Prihlasovacia obrazovka

Po spustení aplikácie sa zobrazí prihlasovacia obrazovka, kde používateľ zadá svoj e-mail a heslo. Ak ešte nemá vytvorený účet, môže sa zaregistrovať cez odkaz pod prihlasovacím formulárom. Po odoslaní správnych údajov sa dostane na hlavnú obrazovku aplikácie. Prihlasovacia obrazovka je zobrazená na obrázku [A.1](#).

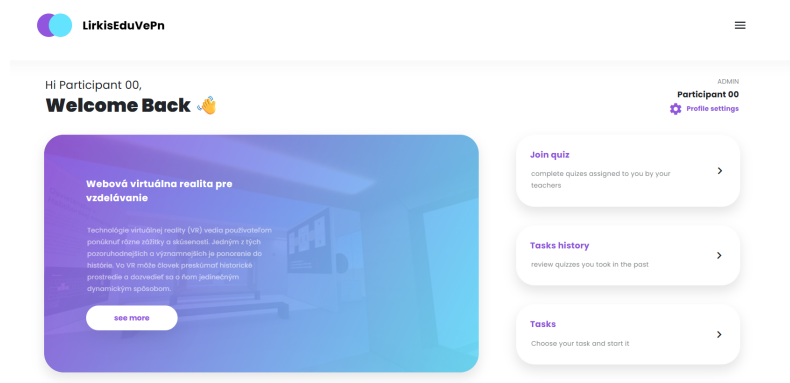


Obrázok A.1: Prihlasovacia obrazovka

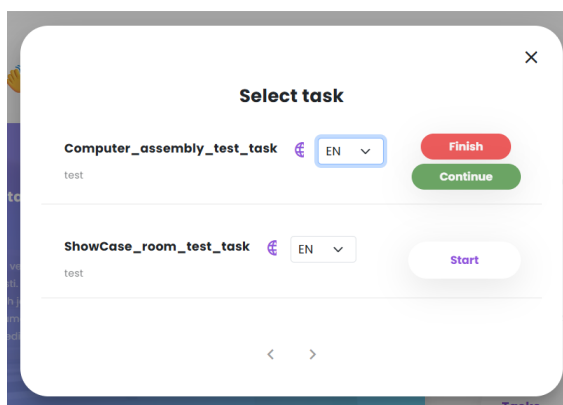
A.3.2 Hlavná obrazovka

Hlavná obrazovka pre žiakov zobrazuje úlohy pridelené učiteľom a zároveň umožňuje prezeráť si výsledky predchádzajúcich pokusov. Jej podoba je na obrázku [A.2](#). Po kliknutí na úlohu sa otvorí dialóg, obrázok [A.3](#), s výberom jazyka, v ktorom môže žiak začať novú úlohu, pokračovať v rozbehnutej alebo ju ukončiť.

Z hlavnej obrazovky sa dá tiež dostať k nastaveniam profilu a zmene informácií o účte.



Obrázok A.2: Hlavná obrazovka žiaka



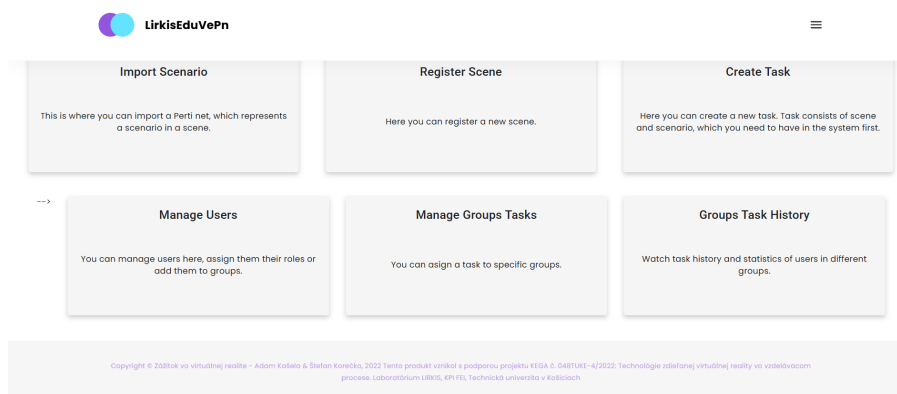
Obrázok A.3: Výber úlohy

A.3.3 Administračný panel pre učiteľov a administrátorov

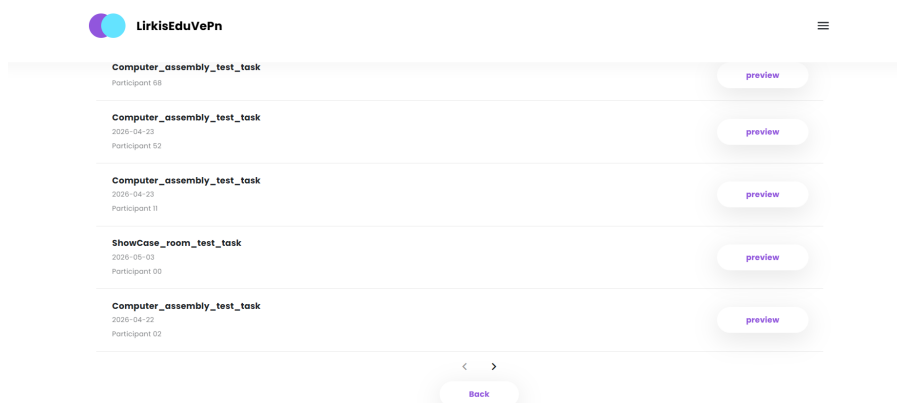
Učitelia a administrátori majú k dispozícii samostatný panel (obrázok A.4), z ktorého spravujú scenáre, úlohy aj používateľov. Cez funkciu *Vytvoriť scenár* pridávajú nový scenár v podobe Petriho siete spolu s jazykovým súborom obsahujúcim popisy, cez *Vytvoriť scénu* prepoja scenár s niektorou z vopred pripravených scén systému a cez *Vytvoriť úlohu* vytvoria pre študentov konkrétnu úlohu, pri ktorej zadajú jej názov, popis, scenár a scénu. Administrátori navyše môžu cez *Spravovať používateľov* vytvárať, upravovať a odstraňovať používateľské účty, priradovať ich do skupín a spravovať ich oprávnenia.

Skupiny študentov sa spravujú cez funkciu *Spravovať skupiny* a slúžia na organizáciu podľa tried, kurzov či iných kritérií; učitelia môžu skupinám priradovať úlohy a sledovať ich priebeh. Históriu plnenia úloh sprístupňuje funkcia *História*

úloh skupín – po vybratí konkrétnej skupiny sa zobrazia všetky úlohy vykonané jej členmi (obrázok A.5). Tlačidlom *preview* si učiteľ zobrazí štatistiky pre danú úlohu (obrázok A.6) a tlačidlom *View extended* aj odpálenia jednotlivých prechodov Petriho siete spolu s informáciou, kedy a či úspešne sa každý prechod odpálil.



Obrázok A.4: Administračný panel pre učiteľov a administrátorov



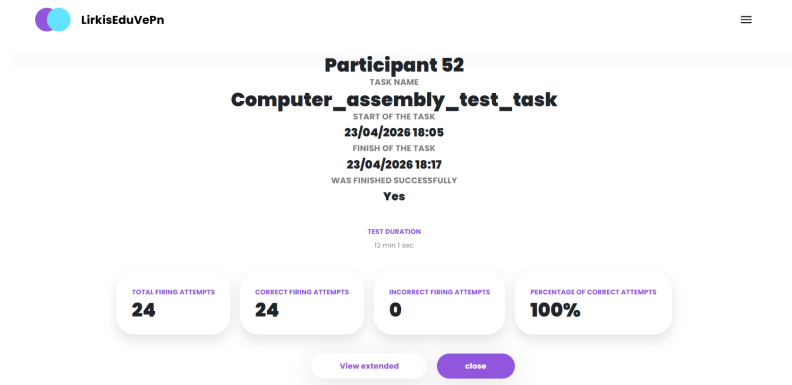
Obrázok A.5: Prehľad histórie úloh pre skupinu

A.4 Použitie ovládania rukami

Po vstupe do VR prostredia systém sám rozpozná, či používateľ drží fyzické ovládače, alebo používa ruky, a aktivuje príslušný režim. Bez ovládačov sa zapne sledovanie rúk; akonáhle používateľ počas relácie ovládače uchopí, systém plynulo prepne na ne.

A.4.1 Prehľad gest

Celé ovládanie je postavené na štyroch gestách a ich kombináciách, ktoré sú zhrnuté v tabuľke A.1.



Obrázok A.6: Štatistiky úlohy jednotlivca

Tabuľka A.1: Základné gestá ovládania rukami

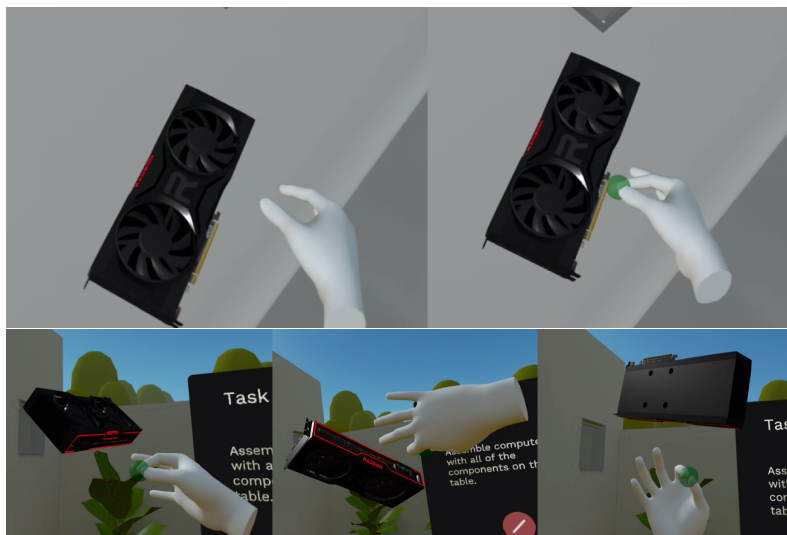
Gesto	Popis a použitie
Štípnutie	Priblíženie palca a ukazováka k sebe. Slúži na uchopenie objektu, potvrdenie teleportácie a kliknutie na prvky rozhrania.
Zohnutie 4 prstov	Zohnutie štyroch prstov (okrem palca) smerom k dlani. Prepína teleportačný režim zapnutý / vypnutý.
Otvorená dlaň	Natiahnutá ruka s rozťahnutými prstami. V teleportačnom režime zobrazí oblúk smeru.
Ukazovanie	Natiahnutý ukazovák smerujúci na vzdialený objekt. Aktivuje lúč pre interakciu na diaľku.

A.4.2 Uchopenie a manipulácia s objektmi

Objekty označené ako uchopiteľné možno zdvihnúť, presúvať, otáčať aj hádzať priamo rukami. Stačí priložiť ruku k vybranému objektu, vykonať gesto štípnutia palcom a ukazovákom a objekt sa pripojí k ruke; nasleduje jej pohyb až do uvoľnenia štípnutia, po ktorom sa objekt odpojí a padá podľa fyzikálnej simulácie. Ak ruka v momente uvoľnenia rýchlo smeruje, objekt sa odhodí s rýchlosťou, ktorú systém vypočíta z pohybu ruky tesne pred pustením.

Ukážka uchopenia a manipulácie je na obrázku [A.7](#).

Ak sa v dosahu ruky nenachádza žiadny uchopiteľný objekt, gesto štípnutia sa namiesto uchopenia interpretuje ako kliknutie na najbližší interaktívny prvok scény.



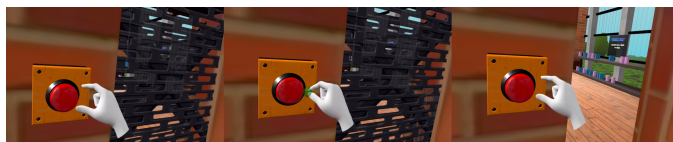
Obrázok A.7: Uchopenie a manipulácia s objektom snímaním rúk

A.4.3 Dotyková interakcia s klávesnicou a tlačidlami

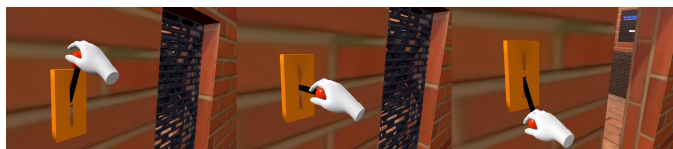
Virtuálne tlačidlá a klávesnicu možno ovládať jednoduchým dotykom – gesto štípnutia tu nie je potrebné. Špička natiahnutého ukazováka sa priloží k prvku a ten sa automaticky aktivuje; po odsunutí prsta sa odaktivuje (obrázok A.10). Klávesnicu možno tiež posunúť na pohodlnejšie miesto – stačí uchopiť jej telo gestom štípnutia, presunúť a pustiť, rovnako ako pri ovládaní myšou v desktopovom režime.

A.4.4 Otváranie dverí

V projekte sa nachádzajú tri typy dverí, pri ktorých sa ovládanie pri snímaní rúk nemení. Dvere s tlačidlom otvoríte stlačením tlačidla vedľa nich gestom štípnutia (obrázok A.8), dvere s páčkou potiahnutím páčky nadol – páčka sa pri tom neuchopuje, stačí na ňu zatlačiť (obrázok A.9). Tajné dvere sa otvárajú zadaním hesla cez virtuálnu klávesnicu, kde sa jednotlivých klávesov dotýka špička ukazováka (obrázok A.10).



Obrázok A.8: Dvere s tlačidlom – otvorenie stlačením tlačidla gestom štípnutia



Obrázok A.9: Dvere s páčkou – otvorenie potiahnutím páčky nadol



Obrázok A.10: Tajné dvere – zadanie hesla pomocou virtuálnej klávesnice

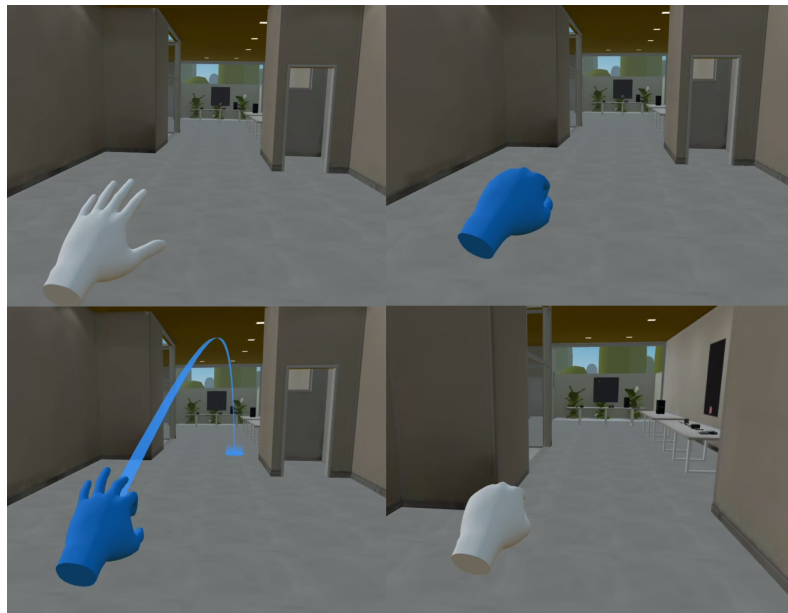
A.4.5 Teleportácia

Pohyb po scéne prebieha pomocou teleportácie ovládanej gestami. Najprv treba zohnúť štyri prsty okrem palca k dlani – ruka zmení farbu a aktivuje sa teleportačný režim. Natiahnutou otvorenou dlaňou potom používateľ namieri na cieľové miesto a zobrazí sa oblúk naznačujúci, kam sa premiestni. Po potvrdení gestom štipnutia sa teleportácia vykoná; ak ju používateľ chce zrušiť, stačí dlaň zatvoriť pred vykonaním štipnutia. Opakovaným zohnutím štyroch prstov sa teleportačný režim vypne.

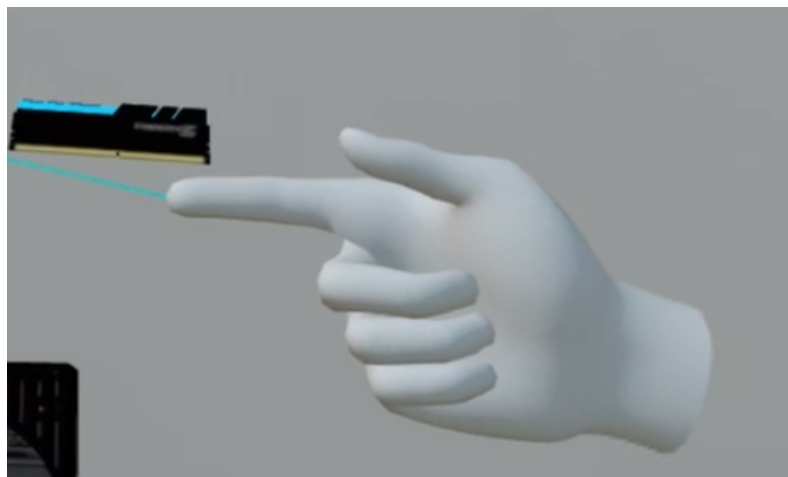
Ak je v momente aktivácie teleportačného režimu uchopený nejaký objekt, jeho držanie sa automaticky ukončí. Ukážka teleportácie je na obrázku [A.11](#).

A.4.6 Interakcia na diaľku

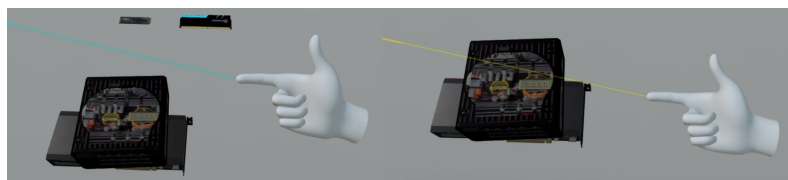
Vzdialené objekty, ku ktorým sa používateľ nemôže fyzicky priblížiť, dokáže priťahovať pomocou lúča ovládaného ukazovák. Natiahnutý ukazovák (obrázok [A.12](#)) namierený na uchopiteľný objekt zobrazí lúč a cieľ sa zvýrazní. Lúč pritom mení farbu podľa toho, či sa v jeho smere nachádza uchopiteľný objekt, ako je znázornené na obrázku [A.13](#). Následným gestom štipnutia sa objekt začne plynulo priťahovať k ruke a po dosiahnutí dlane sa automaticky prepne do bežného režimu uchopenia. Uvoľnením štipnutia ho potom používateľ pustí.



Obrázok A.11: Teleportácia ovládaná gestami ruky – zobrazenie oblúka smeru



Obrázok A.12: Gesto ukazovania – natiahnutý ukazovák určuje smer lúča



Obrázok A.13: Interakcia na diaľku – modrý lúč znamená, že v jeho smere nebol nájdený žiadny uchopiteľný objekt, žltý lúč signalizuje zameraný objekt, ktorý možno pritiahnuť

B Systémová príručka

Táto systémová príručka opisuje technickú stránku rozšírenia systému LirkisEduVePn o ovládanie pomocou snímania rúk. Nadväzuje na systémové príručky priložené k predchádzajúcim prácam [8, 9, 11, 12, 14], kde sú popísané architektúra systému, klientská časť v Angulari, serverová časť v Spring Boote, databáza PostgreSQL aj postup spustenia pomocou nástroja Docker. Tieto časti systému neboli v rámci tejto práce menené, preto sa ich popis tu neopakuje.

B.1 Funkcia programu

Rozšírenie dopĺňa systém LirkisEduVePn o vlastnú vrstvu gestového ovládania nad natívnym sledovaním rúk z WebXR Hand Tracking API a frameworku A-Frame. Vrstva pokrýva päť typov interakcie – uchopenie a manipuláciu s objektmi, fyzikálnu prítomnosť ruky v scéne, dotykovú interakciu s používateľským rozhraním, teleportáciu ovládanú gestami a interakciu na diaľku s možnosťou pritiahnuť vzdialený objekt. Riešenie je súčasťou existujúcej architektúry projektu, zachováva kompatibilitu s riadením scenárov Petriho sieťami aj s tradičnými VR ovládačmi a medzi režimami prepína samo podľa toho, či používateľ drží fyzický ovládač.

B.2 Popis riešenia

Riešenie tvorí šesť A-Frame komponentov v jazyku JavaScript. Centrálnym prvkom je `hand-tracking-bridge`, ktorý poskytuje dáta zo sledovania rúk ostatným piatim modulom – `hand-grab`, `hand-collision-bodies`, `hand-point-fingertip-touch`, `hand-teleport` a `hand-point-shoot`. Každý modul rieši jeden typ interakcie a možno ho zapnúť alebo vypnúť nezávisle od ostatných. Moduly medzi sebou nekomunikujú priamo, ale cez udalosti, ktoré emituje most.

Komponenty sú navrhnuté tak, aby ich bolo možné použiť v ľubovoľnej scéne projektu – existujúcej aj budúcej. V rámci tejto práce sme nové scény nevytvára-

li, len sme do HTML šablón existujúcich scén z predchádzajúcich prác [12, 14] pridali hand-tracking entity. Sceňové skripty ani Petriho siete sme nemeni.

B.3 Štruktúra súborov

Všetky nové komponenty sa nachádzajú v priečinku `LirkisEduVePn-UI/src-/app/components/tasks/js/components`, spolu s ostatnými A-Frame komponentmi systému. Ide o súbory `handTrackingBridge.component.js`, `handGrab.component.js`, `handCollisionBodies.component.js`, `handPointFingertipTouch.component.js`, `handTeleport.component.js` a `handPointShoot.component.js`. Účel a obsah jednotlivých komponentov je popísaný v ďalšej sekcii.

B.4 Popis implementovaných A-Frame komponentov

V tejto sekcii popisujeme jednotlivé komponenty – ich konfiguračné parametre, vstupné a výstupné udalosti a kľúčové aspekty implementácie. Sekvenčné diagramy s priebehom činnosti jednotlivých komponentov sú v kapitole 3.

B.4.1 Komponent hand-tracking-bridge

Most medzi natívnym A-Frame komponentom `hand-tracking-controls` a ostatnými interakčnými modulmi. Poskytuje im dáta o kĺboch ruky, priebežne počíta bod štipnutia (stred medzi špičkami palca a ukazováka), deteguje, či nie je aktívny fyzický ovládač, a vlastnými udalosťami informuje moduly o zmenách stavu.

Parametre

- `hand` – ktorá ruka, hodnoty `'left'` alebo `'right'` (predvolene `'right'`).
- `sphereRadius` – polomer vizualizačnej guľôčky bodu štipnutia v metroch, kladná hodnota (predvolene `0.015`).
- `pinchSphereOpacity` – priehľadnosť vizualizačnej guľôčky, rozsah `0.0–1.0` (predvolene `0.5`).
- `successPinchSphere` – či zobrazí zelenú guľôčku počas aktívneho štipnutia (predvolene `true`).

- `debugSphere` – zobrazenie červenej guľôčky pre účely ladenia (predvolene `false`).
- `debugLog` – výpis ladiacich správ do konzoly (predvolene `false`).
- `defaultHandColor` – predvolená farba modelu ruky v hex formáte (predvolene `#FFFFFF`).
- `collisionEnabled`, `grabEnabled`, `fingertipTouchEnabled`, `teleportEnabled`, `pointShootEnabled` – prepínače pre automatické pripojenie zodpovedajúcich interakčných modulov (predvolene `true`).
- `autoAttachModules` – ak je `true`, komponent pri inicializácii sám pripojí povolené moduly k entite ruky (predvolene `false`).

Emitované udalosti

Most emituje tri vlastné udalosti: `bridge-pinchstarted` signalizuje začiatok gesta štípnutia, `bridge-pinchended` jeho ukončenie a `bridge-controller-active` sa volá pri zmene stavu fyzického ovládača a v parametri `active` odovzdáva, či je ovládač aktívny.

Verejné rozhranie

Pre ostatné moduly most ponúka niekoľko metód. `getMesh()` vracia 3D mesh ruky, `getJoint(jointName)` konkrétny kĺb podľa názvu a `getPinchPoint()` kópiu aktuálneho bodu štípnutia ako `THREE.Vector3`. Farbu modelu ruky možno zmeniť cez `setHandModelColor(color)`. Stav interakcie kontrolujú `isTeleportModeActive()` a `isHoldingObject()`.

Implementačné detaily

Pri inicializácii sa most zapíše do vlastnosti `el.handTrackingBridge` entity ruky, čím ho ostatné moduly nájdu bez ďalšej konfigurácie. V každom snímku najprv kontroluje, či nie je aktívny fyzický ovládač – ak je, výpočet `pinchPointu` sa preskočí a moduly o zmene režimu dostanú udalosť. V opačnom prípade získa svetové pozície špičky palca a ukazováka, vypočíta ich stred a uloží do objektu `pinchData`, z ktorého si moduly čítajú priamo.

B.4.2 Komponent hand-grab

Stará sa o uchopenie objektov, ich manipuláciu počas držania (presúvanie, otáčanie) a následné pustenie alebo hodenie. Reaguje na udalosti `bridge-pinchstarted` a `bridge-pinchended` z mostu.

Parametre

- `grabRadius` – maximálna vzdialenosť, v ktorej sa hľadá uchopiteľný objekt, kladná hodnota v metroch (predvolene 0.15).
- `throwMultiplier` – násobiteľ rýchlosti hodu, kladná hodnota (predvolene 2).
- `maxSpeed` – maximálna povolená rýchlosť hodu v m/s, kladná hodnota (predvolene 10).
- `smoothPosition`, `smoothRotation` – zapnutie plynulého vyhladzovania pozície a rotácie (predvolene true).
- `positionLerpFactor`, `rotationLerpFactor` – koeficienty interpolácie pre vyhladzovanie, rozsah 0.0–1.0 (predvolene 0.3).
- `clickInteractionDistance` – maximálna vzdialenosť pre kliknutie na prvok `.interactable`, ak v dosahu nie je žiadny uchopiteľný objekt, kladná hodnota v metroch (predvolene 1.7).
- `snapToHand` – ak je true, objekt sa pri uchopení presne zarovná s bodom štípnutia bez zachovania pôvodného posunu (predvolene false).

Vyhľadávané objekty a emitované udalosti

Komponent vyhľadáva objekty s atribútmi `[grabbable]` alebo `[grabbable-by-parenting]`. Ak v dosahu `grabRadius` taký objekt nie je, vykoná sa kliknutie na najbližší prvok s triedou `.interactable`. Pri uchopení objektu emituje na ňom `grab-start`, pri pustení `grab-end`.

Implementačné detaily

Pri uchopení sa fyzikálne telo objektu prepne do kinematického režimu, čím sa vyradí z fyzikálnej simulácie a možno ním manuálne hýbať. Počas držania komponent v každom snímku počíta novú cieľovú pozíciu a rotáciu objektu na základe pohybu ruky a vedie si históriu rýchlostí (najviac päť posledných hod-

nôt). Tá sa potom využije pri uvoľnení: telo sa vráti do dynamického režimu a na objekt sa aplikuje priemerná rýchlosť z histórie vynásobená koeficientom `throwMultiplier`, s horným ohraničením `maxSpeed`.

B.4.3 Komponent `hand-collision-bodies`

Bez tohto komponentu by virtuálna ruka prechádzala objektmi v scéne bez akejkoľvek fyzikálnej reakcie, pretože natívne sledovanie rúk v A-Frame poskytuje len vizuálny model. `hand-collision-bodies` preto pre vybrané kĺby vytvára kolízne gule riadené fyzikálnym enginom Ammo.js, čím dáva ruke fyzikálnu prítomnosť v scéne.

Parametre

- `handCollisionRadius` – polomer kolíznych gúl v metroch, kladná hodnota (predvolene `0.008`).
- `ghostDelayAfterRelease` – oneskorenie obnovenia kolízií po pustení objektu v milisekundách, nezáporné celé číslo (predvolene `600`).
- `collisionFilterGroup`, `collisionFilterMask` – konfigurácia kolíznych filtrov Ammo.js ako bitová maska, celočíselná hodnota (predvolene `2` a `-1`).
- `debugHandCollision` – zobrazenie kolíznych gúl pre účely ladenia (predvolene `false`).
- `debugColor`, `debugMetacarpal`, `debugWrist` – farby ladiacich gúl pre rôzne typy kĺbov v hex formáte.

Sledované kĺby a spracovávané udalosti

Komponent vytvára kolízne gule pre 25 kĺbov ruky – zápästie, špičky všetkých prstov, distálne, intermediátne a proximálne články prstov a metakarpálne kosti. Zoznam je v poli `handJointNames` v zdrojovom kóde a zodpovedá špecifikácii WebXR Hand Input.

Z udalostí mostu komponent počúva `bridge-pinchstarted` (na začiatku štipnutia dočasne vypne kolízie, aby pri uchopení nedošlo ku kolíznemu konfliktu s objektom), `bridge-pinchended` (po uplynutí `ghostDelayAfterRelease` kolízie opäť zapne) a `bridge-controller-active` (pri aktívnom fyzickom ovládači sú kolízie vypnuté úplne).

Implementačné detaily

Kolízne gule sú entity `a-sphere` s komponentmi `ammo-body` (typ `kinematic`) a `ammo-shape` (`sphere`). Vkladajú sa priamo do scény, nie ako potomkovia entity ruky – inak by sa dvakrát aplikovala transformácia ruky a gule by sa pohybovali nepredvídateľne. Model ruky sa môže načítavať asynchrónne, preto komponent v prípade neúspechu opakuje pokus o vytvorenie gúl v intervale 150 ms, najviac dvadsaťkrát. V každom snímku potom synchronizuje pozície gúl so svetovými pozíciami zodpovedajúcich kĺbov a zmeny propaguje do fyzikálnej simulácie cez `syncToPhysics()`.

B.4.4 Komponent `hand-point-fingertip-touch`

Dotyková interakcia špičkou ukazováka – pri jej priblížení k interaktívnemu prvku scény vyvolá na ňom sekvenciu udalostí myši a tým simuluje kliknutie. Hodí sa najmä na ovládanie virtuálnej klávesnice a tlačidiel, kde gesto štipnutia pôsobí neprirodzene.

Parametre

- `enabled` – aktivácia komponentu (predvolene `true`).
- `fingertipTouchRadius` – polomer detekcie dotyku v metroch, kladná hodnota (predvolene `0.03`).
- `scanStrideFrames` – skenovanie prebehne každý N-tý snímok, celé číslo ≥ 1 (predvolene `3`).

Vyhľadávané objekty a emitované udalosti

Interaktívne prvky sa vyhľadávajú selektormi `[keyboard-button]`, `.collidable` a `.clickable`. Zoznam kandidátov sa kešuje a obnovuje raz za 180 snímok, čím sa znižuje výpočtová záťaž. Pri novej detekcii dotyku komponent emituje na cieľovom prvku trojicu udalostí `mousedown`, `click`, `mouseup`; keď prst prvok opustí, vyšle `mouseleave`. Kým špička zostáva nad rovnakým prvkom, opakovaný klik sa nevyvolá.

Komponent nepracuje, ak je aktívny fyzický ovládač, prebieha teleportácia alebo ruka drží objekt. Tieto stavy zisťuje cez verejné metódy mostu.

B.4.5 Komponent hand-teleport

Pohyb používateľa po scéne pomocou teleportácie ovládanej gestami ruky. Samotné premiestnenie robí komponent `blink-controls`, ktorý je v scéne pripojený na osobitnú entitu (`#ht-teleport`) – `hand-teleport` ho len riadi programovo.

Parametre

- `enabled` – aktivácia komponentu (predvolene `true`).
- `teleportControllerEl` – A-Frame výberový identifikátor entity s komponentom pre ovládanie teleportácie `blink-controls`, ktorá bude použitá ako cieľový ovládač teleportácie.
- `teleportGestureHoldFrames` – minimálny počet snímkov pre stabilnú detekciu gesta otvorenej dlane pred aktiváciou oblúka, celé číslo ≥ 1 (predvolene 8).
- `teleportColor` – farba modelu ruky v teleportačnom režime v hex formáte (predvolene `#4D93FD`).

Detekované gestá

Komponent rozpoznáva tri gestá. Zohnutie štyroch prstov (ukazovák, prostredník, prsteňák, malíček zohnuté pod uhlom menším ako 90°) prepína teleportačný režim a vyhodnocuje sa v metóde `isGrabGesture()`. Otvorená dlaň, ktorú overuje `isOpenPalmGesture()`, vyžaduje natiahnutie prostredníka aspoň na 1,4-násobok dĺžky jeho proximálneho článku a dostatočné natiahnutie prsteňáka a malíčka – aktivuje teleportačný oblúk. Štípnutie vykonané počas aktívneho oblúka teleportáciu potvrdí.

Implementačné detaily

V každom snímku komponent synchronizuje pozíciu a orientáciu entity `blink-controls` s dlaňou ruky. Pozícia je stred medzi zápästím a metakarpálom prostredníka, smer je vektor zo zápästia ku špičke prostredníka. Pri štípnutí počas aktívneho oblúka komponent zavolá `onButtonUp()` na `blink-controls`, čím sa spustí samotná teleportácia. Ak ruka pred potvrdením stratí gesto otvorenej dlane, oblúk sa zruší cez `cancel()`. Ak je v momente aktivácie režimu ucho-

pený objekt, jeho držanie sa programovo ukončí volaním `onPinchEnded()` na `hand-grab`.

B.4.6 Komponent `hand-point-shoot`

Interakcia na diaľku – používateľ označí vzdialený objekt ukazovákom a následným gestom ho pritiahne k ruke. Po priblížení komponent odovzdá riadenie modulu `hand-grab`, ktorý objekt uchopí.

Parametre

- `enabled` – aktivácia komponentu (predvolene `true`).
- `pointShootMaxDistance` – maximálna vzdialenosť, na ktorú dosiahne lúč, kladná hodnota v metroch (predvolene 10).
- `pointShootPullSpeed` – rýchlosť pritiahnutia objektu, kladná hodnota v m/s (predvolene 5).
- `triggerDistance` – maximálna vzdialenosť medzi špičkou palca a kĺbom ukazováka pre aktiváciu interakcie, kladná hodnota v metroch (predvolene 0.04).
- `showPointRay` – zobrazenie vizualizácie lúča (predvolene `true`).
- `rayDefaultColor`, `rayDetectColor` – farby lúča pri prázdnom zameraní a pri zameraní na objekt v hex formáte (predvolene `#00FFFF` a `#FFFF00`).
- `debugPointGestureSphere`, `debugPointShootSphere` – zobrazenie ladiacich gúľ (predvolene `false`).

Detekované gestá

Komponent rozpoznáva dve gestá. Ukazovaním rozumieme polohu, v ktorej je ukazovák natiahnutý (uhol medzi metakarpálom a špičkou väčší ako 150°) a prostredník zohnutý (uhol pod 150°). Spúšťacie gesto je priblíženie špičky palca ku kĺbu ukazováka na vzdialenosť menšiu ako `triggerDistance`.

Implementačné detaily

Smer lúča komponent počíta zo smerov medzi kĺbmi ukazováka – metakarpál–proximál a proximál–intermediálny článok – kombinovaných v pomere 0,4 : 0,6. Výsledný smer plynulo vyhladzuje lineárnou interpoláciou s koeficientom 0,25,

čím sa potlačí drobné chvenie ruky. Prieniky lúča s objektmi zisťuje THREE.Ray a Box3 oproti všetkým objektom s atribútmi [grabbable] alebo [grabbable-by-parenting]. Pri spustení interakcie sa cieľový objekt prepne do kinematického režimu, plynulo sa posúva k bodu štípnutia rýchlosťou pointShootPullSpeed a po dosiahnutí ruky sa cez vlastnosť pointShootLockedTarget komponentu hand-grab odovzdá riadenie ďalej.

B.5 Integrácia komponentov v scéne

Komponenty sa pripájajú k entitám, ktoré reprezentujú ruky používateľa pri sledovaní cez WebXR Hand Tracking API. V scéne sú samostatné od entít fyzických VR ovládačov, takže systém dokáže podporovať oba režimy súbežne. Príklad konfigurácie hand-tracking entít v HTML šablóne scény je vo výpise B.1.

Výpis B.1: Konfigurácia hand-tracking entít v scéne

```
<!-- Hand-tracking hands (WebXR hand tracking) -->
<a-entity id="lhand-ht"
  hand-tracking-controls="hand: left"
  hand-tracking-bridge="hand: left; teleportEnabled: true"
  hand-grab
  hand-collision-bodies
  hand-point-fingertip-touch
  hand-teleport="teleportControllerEl: #ht-teleport"
  hand-point-shoot="enabled: false">
</a-entity>

<a-entity id="rhand-ht"
  hand-tracking-controls="hand: right"
  hand-tracking-bridge="hand: right; pointShootEnabled:
    true"
  hand-grab
  hand-collision-bodies
  hand-point-fingertip-touch
  hand-teleport="enabled: false"
  hand-point-shoot>
</a-entity>

<!-- Teleport entity -->
<a-entity id="ht-teleport"
```

```

      blink-controls="cameraRig: #player;
                    teleportOrigin: #playerhead;
                    collisionEntities: #navmesh-entity;
                    button: thumbstick;
                    rotateOnTeleport: false"
      visible="false">
</a-entity>

```

V uvedenej konfigurácii ľavá ruka rieši teleportáciu a pravá interakciu na diaľku, pričom uchopenie objektov, fyzikálnu prítomnosť aj dotykovú interakciu podporujú obe. Most `hand-tracking-bridge` musí byť pripojený ako prvý, pretože ostatné moduly očakávajú jeho prítomnosť cez vlastnosť `el.handTrackingBridge`. Entita `#ht-teleport` je samostatný ovládač teleportácie nezávislý od fyzických ovládačov a jej referenciu komponent `hand-teleport` prijíma cez parameter `teleportControllerEl`.

B.6 Integrácia s existujúcimi komponentmi systému

Vrstva gestového ovládania spolupracuje s viacerými komponentmi z predchádzajúcich prác, ktoré sme v tejto práci nemenili. Z natívnych A-Frame komponentov ide o `hand-tracking-controls` (poskytuje dáta o polohách kľbov cez WebXR Hand Tracking API) a `hand-controls` (spravuje fyzické VR ovládače – ich aktivita má vždy prednosť pred sledovaním rúk). Z externých komponentov projektu používame `blink-controls` pre samotnú teleportáciu a komponenty `ammo-body` a `ammo-shape` z fyzikálneho enginu Ammo.js, ktoré spravujú fyzikálne telá a kolízne tvary.

Okrem komponentov sa využívajú aj existujúce A-Frame atribúty a CSS triedy zo scén systému: `[grabbable]`, `[grabbable-by-parenting]`, `[keyboard-button]`, `.clickable`, `.collidable` a `.interactable`. Tieto označenia zostali zachované, takže nové komponenty fungujú so scénami predchádzajúcich autorov bez ďalších úprav.

B.7 Využité technológie

Rozšírenie je postavené na frameworku A-Frame vo verzii 1.6.0 a vyššej, ktorý zaisťuje tvorbu VR scén v prehliadači a poskytuje natívnu podporu pre sledovanie rúk cez WebXR Device API a WebXR Hand Input Module Level 1. A-Frame stavia na knižnici Three.js, z ktorej používame triedy `THREE.Vector3`,

THREE.Quaternion, THREE.Ray a THREE.Box3 na prácu s geometriou a detekciu prienikov. Fyziku rieši Ammo.js – JavaScriptový port enginu Bullet. Všetky komponenty sú napísané v JavaScripte (ES6+).

B.8 Spustenie a nasadenie

Postup spustenia projektu lokálne aj nasadenia na univerzitný server sa rozšírením o gestové ovládanie nezmenil – podrobnosti sú v používateľskej príručke (príloha A, sekcie 2.2 a 2.3) a v systémových príručkách predchádzajúcich prác [11, 12, 14].

Pre využitie gestového ovládania je navyše potrebné, aby pripojený VR headset podporoval sledovanie rúk cez WebXR Hand Tracking API. Žiadnu ďalšiu konfiguráciu systém nevyžaduje – prítomnosť alebo absenciu fyzických ovládačov deteguje sám a podľa toho prepína medzi režimami.