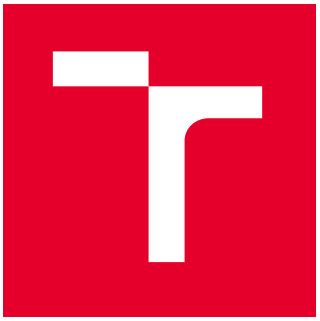


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ

FACULTY OF MECHANICAL ENGINEERING

ÚSTAV MECHANIKY TĚLES, MECHATRONIKY A BIOMECHANIKY

INSTITUTE OF SOLID MECHANICS, MECHATRONICS AND BIOMECHANICS

VYTVOŘENÍ 3D MODELU ČLOVĚKA PRO VÝUKU ZNAKOVÉHO JAZYKA

CREATING A 3D HUMAN MODEL FOR TEACHING SIGN LANGUAGE

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Tomáš Lička**
AUTHOR

VEDOUCÍ PRÁCE — **doc. Ing. Jiří Krejsa, Ph.D.**
SUPERVISOR

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA STROJNÍHO INŽENÝRSTVÍ

Zadání diplomové práce

Ústav:	Ústav mechaniky těles, mechatroniky a biomechaniky
Student:	**Bc. Tomáš Lička**
Studijní program:	Mechatronika
Studijní obor:	bez specializace
Vedoucí práce:	**doc. Ing. Jiří Krejsa, Ph.D.**
Akademický rok:	2022/23

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma diplomové práce:

Vytvoření 3D modelu člověka pro výuku znakového jazyka

Stručná charakteristika problematiky úkolu:

Znakový jazyk je tvořen pohyby horní poloviny těla a mimikou obličeje. Obvyklá praxe při jeho výuce zahrnuje buď přítomnost lektora nebo přehrávání videí. Cílem této práce je vytvoření 3D modelu člověka, který by dokázal předvést libovolný pohyb. Tento pohyb by byl definován parametry, a tedy by umožnoval vygenerovat libovolnou sekvenci znaků a přechodů mezi těmito znaky. Hodnoty parametrů by byly získány např. pomocí frameworku Mediapipe.

Cíle diplomové práce:

1. Popište dostupné aplikace pro výuku znakového jazyka.
2. Vytvořte 3D model člověka pro potřeby prezentace znakového jazyka.
3. Vytvořený model člověka rozpohybujte dle parametrického popisu.
4. Vytvořte aplikaci pro výuku znakového jazyka.

Seznam doporučené literatury:

Blan J.M.: The Complete Guide to Blender Graphics: Computer Modeling & Animation, A K Peters/CRC Press; 7th edition, 2022

Termín odevzdání diplomové práce je stanoven časovým plánem akademického roku 2022/23

V Brně, dne

L. S.

prof. Ing. Jindřich Petruška, CSc. ředitel ústavu	doc. Ing. Jiří Hlinka, Ph.D. děkan fakulty

Abstrakt

Tato práce se zabývá vývojem mobilní aplikace na výuku českého znakového jazyka. Jednotlivé znaky budou předváděny 3D modelem člověka. Předlohou pro jeho pohyb budou videa, kde tlumočnice předvádí znaky. Analýzou těchto videí bude docíleno toho, že model bude předvádět stejné pohyby jako tlumočnice na videu. Výsledná mobilní aplikace bude kompatibilní s mobilními zařízeními používající operační systém Android.

Klíčová slova

Český znakový jazyk, mobilní aplikace, inverzní kinematika, 3D model

Abstract

This thesis deals with developing of a mobile application for learning czech sign language. Individual signs will be shown by 3D model of human. Its movement will be based on videos where lector is demonstrating signs. By analyzing these videos, it will be achieved that the model will perform the same movement as the lector. Final mobile application will be compatible with mobile phones using Android operating system.

Key words

Czech sign language, mobile app, inverse kinematics, 3D model

Bibliografická citace

LIČKA, Tomáš. *Vytvoření 3D modelu člověka pro výuku znakového jazyka* [online]. Brno, 2023 [cit. 2023-05-05]. Dostupné z: https://www.vut.cz/studenti/zav-prace/detail/149636. Diplomová práce. Vysoké učení technické v Brně, Fakulta strojního inženýrství, Ústav mechaniky těles, mechatroniky a biomechaniky. Vedoucí práce Jiří Krejsa.

Čestné prohlášení

Prohlašuji, že tato práce je mým původním dílem, zpracoval jsem ji samostatně pod vedením doc. Ing. Jiřího Krejsy, Ph.D s použitím informačních zdrojů uvedených v seznamu.

V Brně 12. května 2023 …….

Podpis

Poděkování

Rád bych poděkoval vedoucímu diplomové práce panu doc. Ing. Jiřímu Krejsovi, Ph.D. za ochotu a cenné rady při zpracovávání této závěrečné práce. Poděkování také patří Mgr. Markétě Mikulové za ochotu být předlohou pro pohyb 3D modelu. Zároveň bych rád poděkoval rodině a přítelkyni za motivaci a podporu při tvorbě práce.

Obsah

Úvod

Komunikace mezi slyšícími a neslyšícími osobami je zprostředkována pomocí znakového jazyka. V dnešní době se jedná již o plnohodnotný jazyk, kterým je možné se vyjadřovat stejně jako v mluveném slově. Pro učení znakového jazyka je nutný vizuální kontakt s lektorkou či lektorem, v případě mobilních aplikací jsou nutná předtočená videa.

Cílem této práce je vytvoření mobilní aplikace, kde místo předtočených videí bude jednotlivé znaky předvádět 3D model člověka. Tento přístup zatím nebyl v žádné nalezené aplikaci použit a je tím unikátní. Použití modelu člověka s sebou přináší i značné výhody. Jednou z hlavních je možnost poskládání libovolného počtu znaků za sebe, kde přechody mezi jednotlivými znaky budou plynulé. V případě použití videí je nutný střih a přechod na další. Plynulé přechody jsou možné pomocí parametrického popisu pohybu modelu.

Využití modelu místo předtočených videí má i výhodu v tom smyslu, že výsledná mobilní aplikace bude zabírat méně úložného prostoru v paměti mobilního zařízení. Již dostupné mobilní aplikace šetří místo v paměti ukládáním videí na vzdálená úložiště, ze kterých poté videa přehrávají. Pokud by však videa byla uložena v aplikacích, celková aplikace by zabírala značnou část uživatelské paměti.

V rámci této práce bude pomocí modelu člověka a skriptu napsaného v jazyce Python připravena mobilní aplikace, která funguje ve dvou režimech: Procvičování a Slovník. Pro vývoj této aplikace byla použita videa, na kterých tlumočnice předváděla jednotlivé znaky. Tato videa byla dále zpracována a převedena na pohyb modelu.

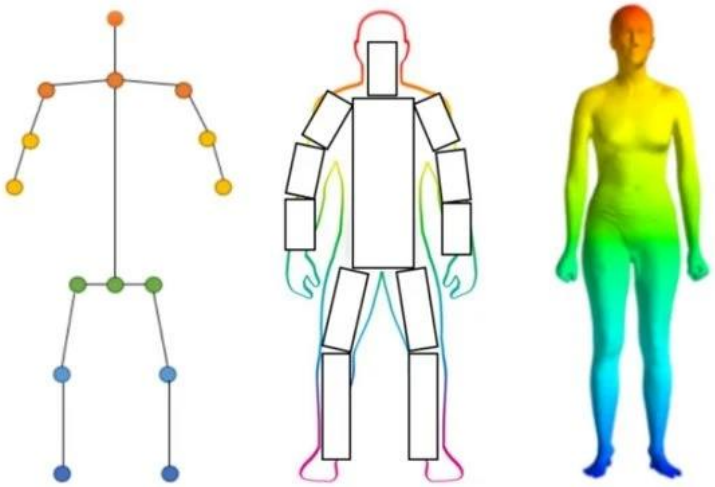
Znakový jazyk se vyznačuje i svojí vlastní gramatikou, která se v některých případech liší od gramatiky mluveného jazyka. Z toho důvodu je tato diplomová práce koncipována tak, že model bude předvádět pouze jednotlivé znaky. Z důvodu odlišné gramatiky nebude možné skládat více znaků za sebe.

1 Rešerše

V této části budou popsány metody, jakými dosáhnout rozpohybovaného modelu. Předlohou pro jeho pohyb jsou videa, tudíž je nutné najít způsob, jakým z videa vyseparovat souřadnice kloubů ruky. Budou zároveň popsány programy, ve kterých je možné vytvořit daný 3D model, společně s jejich výhodami a nevýhodami. K rozpohybování je třeba správně zvolený přístup inverzní kinematiky, jejíž možnosti zde budou také popsány. V poslední části této kapitoly budou představeny nástroje pro export souborů do podoby, která bude kompatibilní s mobilním zařízením.

1.1 Zpracování obrazu

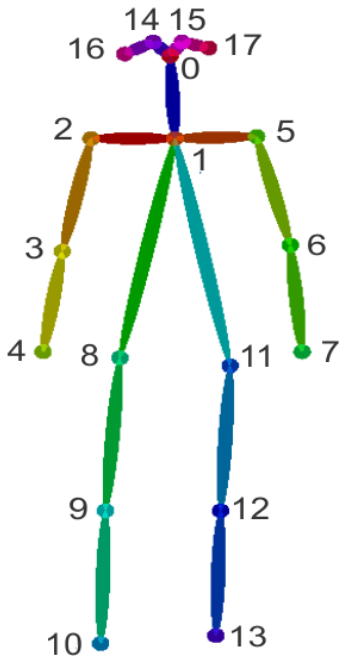
Detekce pózy, polohy rukou i výrazu v obličeji je klíčová komponenta při práci s videi, či fotkami. Její důležitost se projevuje například v kooperaci robotů s lidmi, při vizuálním porozumění a interpretování pohybů lidského těla, což může být využito například ve fyzioterapii. Existují různé frameworky využívající neuronové sítě pro přesnou estimaci souřadnic kloubů lidského těla. V rámci této práce budou představeny pouze kinematické modely, které mají jednotlivé klíčové body spojené úsečkami. Rovinné a objemové modely nebudou zmíněny. Rozdíly mezi těmito modely jsou zobrazeny na Obr. 1 pocházejícího z [1].



Obr. 1 Kinematické modely

- **OpenPose**

  Prvním z frameworků, které budou představeny v této práci, je OpenPose [2]. Tento framework byl použit při vývoji slovníku, který je popsán v podkapitole Slovník českého znakového jazyka a vznikl v Kalifornii v roce 2019. Framework dokáže najít 18 klíčových bodů pózy a určit jejich souřadnice x a y. Tyto klíčové body jsou zobrazeny na Obr. 2, jejich zdrojem je [3].



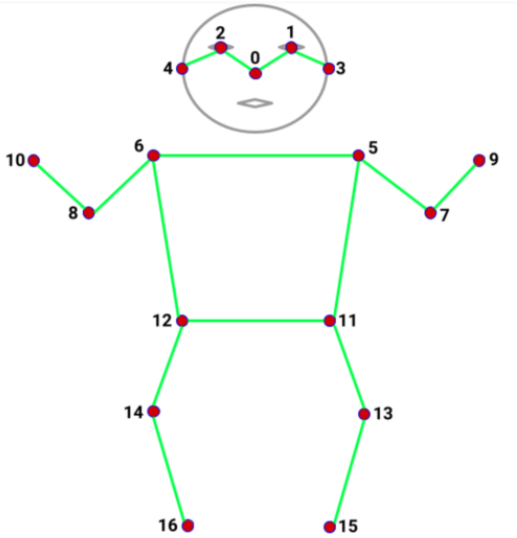
Obr. 2 Skeleton OpenPose

Estimace pozic klíčových bodů není vždy dokonalá, zejména pokud se člověk nachází v nějaké neobvyklé poloze nebo je vyfocen z nezvyklého úhlu. V takovém případě je nutná ruční korekce dat. Další problém může nastat, pokud je na snímku osoba společně se sochou, kterou algoritmus vyhodnotí také jako osobu. Tento framework zároveň dokáže rozeznat i více osob na jednom snímku. To je hlavní výhodou oproti frameworku Mediapipe, který je toho také schopen, ale uživatel musí zadat počet lidí na snímku. Kromě osob je OpenPose schopen rozeznávat i dopravní prostředky.

Framework OpenPose je schopen určovat až 25 klíčových bodů spojených s pózou, 21 klíčových bodů na ruce a 70 bodů na obličeji. Rychlost nalezení klíčových bodů je závislá na počtu osob na snímku [4].

- **MoveNet**

  Dalším frameworkem, který bude zmíněn je MoveNet [5]. Tento framework používá Tensorflow, jež je hojně využíván při vytváření neuronových sítí. Framework MoveNet nalezne 17 klíčových bodů pózy, které jsou zobrazeny na Obr. 3. Tyto klíčové body jsou také k vidění v [6].



Obr. 3 Skeleton MoveNet

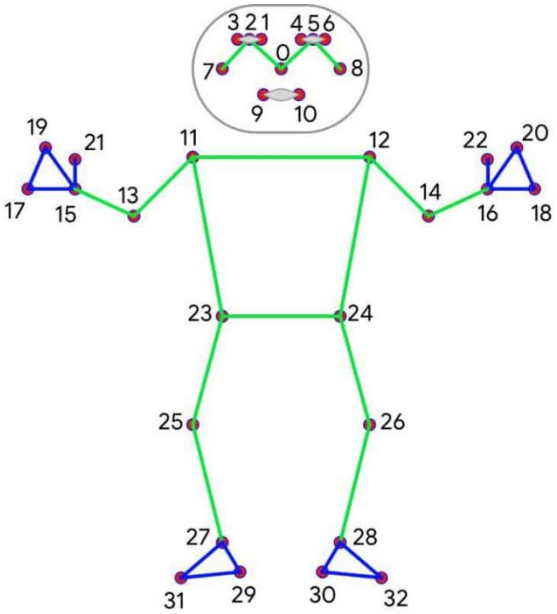
Vývojáři rozdělili MoveNet na dvě části, Lightning a Thunder. Tyto dvě části se liší jejich použitím. Lightning je schopen velmi rychle určit klíčové body i při větším počtu osob na snímku. Přesnost těchto bodů však může být horší. Na druhou stranu Thunder je cílen na co nejpřesnější určení souřadnic klíčových bodů. Obě tyto varianty však fungují v reálném čase.

MoveNet je schopen správně určovat souřadnice klíčových bodů i při rychlých pohybech, s čímž má Mediapipe, který byl použit v této práci, někdy problémy. Zároveň je tento framework schopen detekovat klíčové body správně i při nezvyklých pózách, jak je ukázáno v [6].

- **Mediapipe**

  Mediapipe [7] je framework vytvořený společností Google, jehož používání je zdarma. V této diplomové práci byl použit právě Mediapipe, konkrétně model *holistic*, který slučuje rozeznávání pózy, obou rukou a také obličeje. Snímání obličeje ale nebylo dále zpracováváno.

  Jedná se o soubor několika závislých neuronových sítí pracujících společně pro větší efektivitu řešení. Tento framework byl použit z několika důvodů. Jako jediný při určování pózy určuje i významné klíčové body na dlaních, jak je ukázáno na Obr. 4 pocházejícího z [8]. Celkový počet klíčových bodů pózy je 33.



Obr. 4 Skeleton Mediapipe

Každý klíčový bod pózy má 4 souřadnice:

x, y – normované souřadnice klíčového bodu ve snímku, interval 0-1

z – odhadovaná vzdálenost klíčového bodu od kamery

viditelnost – pravděpodobnost, jestli je daný klíčový bod viditelný, interval 0-1

Klíčové body pózy je možné také získat v metrech místo normovaných souřadnic.

Klíčových bodů na obličeji je dohromady 468, což umožňuje velmi detailní mapování mimiky, která je u znakového jazyka důležitým prvkem. Klíčové body obličeje mají 3 souřadnice:

x, y – normované souřadnice klíčového bodu, interval 0-1

z – hloubka klíčového bodu vzhledem ke středu hlavy, s menší vzdáleností roste tato hodnota

Souřadnice klíčových bodů rukou mají stejný význam jako souřadnice obličeje. Jediným rozdílem je souřadnice z, která určuje hloubku vzhledem k zápěstí. Těchto klíčových bodů je 21 pro každou ruku.

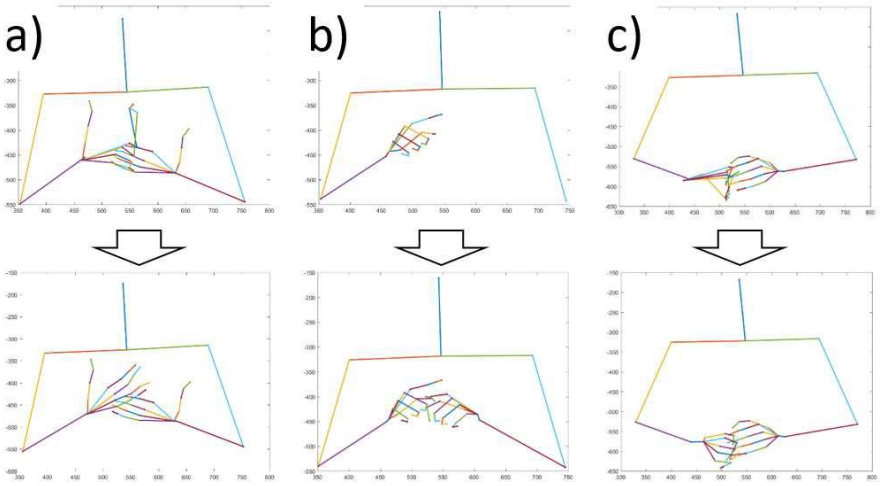
1.2 Slovník českého znakového jazyka

Českým znakovým jazykem (ČSL) se zabývají také na Západočeské univerzitě v Plzni, kde se vytváří slovník českého znakového jazyka [9]. Hlavním cílem je ze zadaného textu vytvořit model, který by naznačoval pozici kloubů v prostoru. Jejich předlohou jsou videa z předpovědi počasí České televize. Pro tvorbu tohoto algoritmu jsou použita data od září 2015 až do července 2018, celkem 947 videí předváděných jinými překladateli.

Pozice kloubů je získávána pomocí frameworku OpenPose, který z každého snímku vytvoří 135 klíčových bodů na těle, obličeji a rukou. Při využití této metody se projevily problémy, které jsou při práci s jednotlivými snímky z videa nevyhnutelné. Prvním tímto

problémem jsou rozmazané ruce při rychlých pohybech překladatelky na videu. V takovém případě je pro OpenPose velmi náročné správně určit souřadnice kloubů. Druhý problém činí neviditelnost některých kloubů, zejména na rukou. U znakového jazyka je častým jevem, že některé prsty jsou sevřené, zobrazené pod různými úhly a identifikace jednotlivých kloubů je tím značně omezena. OpenPose tyto dva problémy řeší tak, že dané klíčové body vynechá.

Pro tvorbu modelu, který se skládá pouze ze souřadnic kloubů, je však nutné, aby byly určeny všechny souřadnice. Proto byla vyvinuta metoda založená na iterativním algoritmu využívajícím backpropagation. V tomto procesu byly délky kostí, jednotlivé úhly a souřadnice počátku kostí trénovatelnými parametry, jejichž estimaci prováděla neuronová síť. Tímto způsobem je možné doplnit chybějící souřadnice klíčových bodů tak, jak je zobrazeno na Obr. 5**.**



Obr. 5 Oprava uspořádání kostí

Na Obr. 5 v částech a) a c) je zobrazeno špatné uspořádání kostí, kterých není možné dosáhnout. V části b) je zobrazena korekce chybějících kostí.

Korekce jednotlivých klíčových bodů závisela také na tom, kdo byl v daném videu v roli překladatele (jiné délky kostí, jiná postava, …). Do neuronové sítě bylo proto přidáno i ID jednotlivých tlumočníků pro rychlejší natrénování. K trénování bylo použito 875 videí obsahujících půl milionu snímků, pro testování neuronové sítě bylo použito zbylých 36 videí obsahujících 20 000 snímků. Z videí předpovědi počasí bylo natrénováno téměř 600 českých slov, pro které byly vytvořeny skeletony ve 3D.

V rámci této práce nemohl být tento soubor použit z důvodu natáčení tlumočníků pouze ze předu. Pro kvalitní pohyb modelu musí být známá i informace o pohybech vpřed a vzad. Z videí natočených z čelního pohledu jsou tyto informace velmi těžce získávány.

1.3 Tvorba modelu

Základem této diplomové práce je 3D model člověka vytvořený v grafickém editoru. Na výběr je několik velmi používaných softwarů, jejichž použití je zdarma a dokumentace k nim je dostatečně obsáhlá.

Maya

Prvním z takových softwarů je Maya, která je hojně využívaná při vytváření animací. Jejich tvorba v tomto programu je velmi efektivní a rychlá. Z toho důvodu se velmi často používá při tvorbě animovaných filmů a v herním průmyslu. Soubory vytvořené v programu Maya jsou ukládány pomocí textu, tudíž modely lze upravovat i bez přímého použití tohoto softwaru. Stejně jako u většiny grafických softwarů je křivka učení velmi pomalá.

Blender

Druhým hojně využívaným softwarem je Blender. Hlavní výhody tohoto softwaru jsou uvedeny níže. Zdrojem těchto informací je [10] a vlastní zkušenosti s tímto programem.

- **Je zdarma a open-source**

  Tento software je pod licencí GNU, tím pádem je jeho používání zcela zdarma a účel jeho využití je nepodstatný. Pro jednorázové použití v této diplomové práci je jeho využití dostačující.
- **Vyšší diverzifikace**

  Možnosti, které Blender nabízí, jsou velmi rozsáhlé a ke stejnému cíli je možné se dostat více způsoby. V této diplomové práci byly použity pouze některé možnosti. Klíčovým faktorem byla schopnost exportovat data a přistoupit k nim ve skriptu v jazyce Python.
- **Komunita uživatelů**

  Blender samotný je vylepšován a vyvíjen svojí komunitou. Ta založila fórum, jehož cílem je řešit problémy ostatních uživatelů. Každý uživatel ho používá k něčemu jinému, tudíž hledání informací je velmi jednoduché. Zároveň nové úpravy a nové verze softwaru vychází s velkou frekvencí, právě kvůli velké komunitě uživatelů.
- **Psaní programů v softwaru Blenderu**

  Přestože se jedná o základní funkci grafických editorů, měla v této práci významnou roli v usnadňování úkonů. Skripty v jazyce Python je možné spouštět přímo v Blenderu a tím automatizovat činnosti, jejichž ruční provedení by zabralo velké množství času. V této diplomové práci bylo této vlastnosti několikrát využito při exportování vah uzlů a v testovacím skriptu na identifikaci jednotlivých uzlů.

Stejně jako každý software má i Blender své nevýhody. Mezi ty nejvýznamnější patří:

- **Plochá křivka učení**

  Při prvních použitích tohoto programu může být orientace v něm velmi složitá a bez znalosti klávesových zkratek i velmi zdlouhavá a náročná. Ve všech tutoriálech jsou ale tyto klávesové zkratky použity, tudíž jejich naučení nezabere velké množství času.

- **Funkcionalita**

  Jelikož se jedná o open-source program, je možné, že některé nové prvky budou způsobovat problémy a špatnou funkcionalitu. Tento problém však může být vyřešen velmi rychle vlivem častého vydávání nových verzí programu.

- **Obtížné programování**

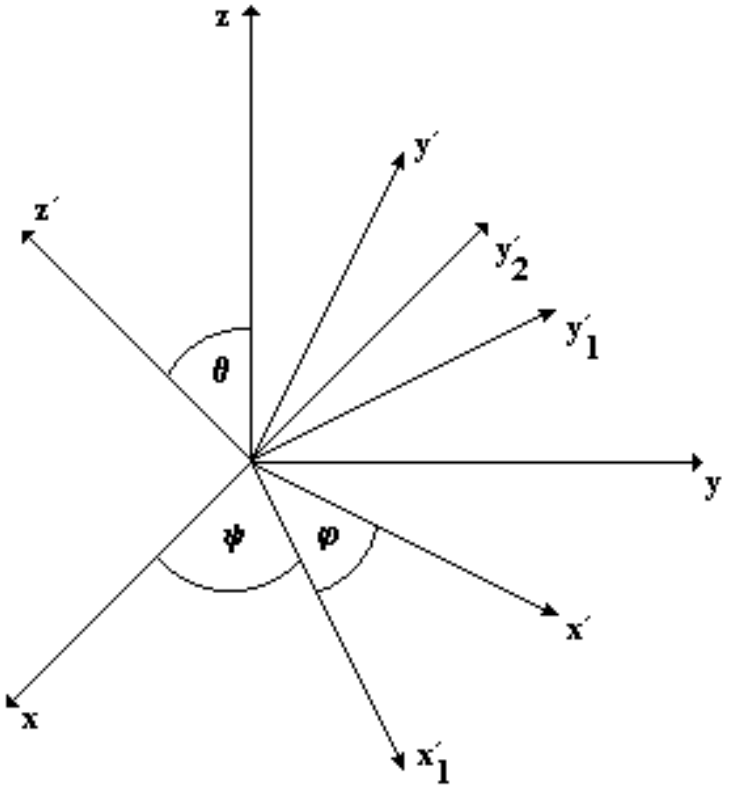
  Psaní programů v Pythonu samo o sobě není těžké, jedná se o jeden z nejméně náročných programovacích jazyků. Jeho aplikace v Blenderu je však obtížná. Editor v tomto softwaru neumožňuje nápovědy a je velmi těžké najít vlastnost, ke které chce uživatel přistoupit ve složitých defaultně vytvořených strukturách. Tento problém částečně řeší komunita uživatelů a fórum, kde je možné kopírovat kostry programů, do kterých je nutné implementovat pouze drobné změny.

1.4 Kinematika pohybu

Model vytvořený v programu Blender je ve výchozí pozici s upaženýma rukama. Pro předvádění znaků je třeba použít vhodnou formu kinematického popisu. V této práci budou uvažovány pouze rotace, translace nebudou popisovány. Způsoby popsání obecné rotace v 3D prostoru jsou následující:

- **Eulerovy úhly**

  Prvním ze způsobu popsání rotace jsou Eulerovy úhly. Eulerovy úhly popisují sférický pohyb kolem okamžité osy otáčení, která je zároveň i nositelkou úhlové rychlosti otáčení ω. Eulerovy úhly jsou tři: vlastní rotace φ, precese ψ a nutace υ. Tyto tři úhly jsou zobrazeny na Obr. 6 pocházejícího z [11].



Obr. 6 Eulerovy úhly

Nevýhodou použití Eulerových úhlů je jev zvaný gimbal lock, při kterém je při nevhodném natočení možná ztráta jednoho stupně volnosti. To zapříčiní, že rotace kolem dvou souřadných os bude mít na dané těleso stejný účinek.

- **Rotační matice**

  Dalším způsobem je použití tří rotačních ortogonálních matic, které popisují rotaci kolem souřadných os x, y a z o daný úhel. Tyto tři rotační matice mají tvar (1). V každé z těchto rovnic jsou pouze 4 nezávislé prvky, zbylých 5 je buď závislých nebo mají pevně danou hodnotu.

$$\boldsymbol{R}_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \sin\varphi & \cos\varphi \end{bmatrix},$$

$$\boldsymbol{R}_y = \begin{bmatrix} \cos\varphi & 0 & \sin\varphi \\ 0 & 1 & 0 \\ -\sin\varphi & 0 & \cos\varphi \end{bmatrix}, \quad (1)$$

$$\boldsymbol{R}_z = \begin{bmatrix} \cos\varphi & -\sin\varphi & 0 \\ \sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

kde $\boldsymbol{R}_x$ je matice rotace kolem osy x, $\boldsymbol{R}_y$ je matice rotace kolem osy y, $\boldsymbol{R}_z$ je matice rotace kolem osy z a φ je úhel rotace. Ten může být pro každou z rotačních matic odlišný.

Výsledné natočení je výsledkem vzájemného násobení rotačních matic $\boldsymbol{R_x}$, $\boldsymbol{R_y}$ a $\boldsymbol{R_z}$. Nevýhodou použití rotačních matic je určení správného pořadí násobení, protože operace násobení není pro matice asociativní. Z toho důvodu není vhodné použít tuto metodu pro popis pohybu modelu člověka, který koná velký rozsah pohybů.

- **Kvaterniony**

  Kvaterniony vznikly v roce 1843 jako rozšíření vyjádření rotací pomocí ortogonálních matic o rozměrech 3x3 popsaných výše. Jejich hlavní výhodou je jednodušší použití při rotacích kolem osy, která se neshoduje ani s jednou ze souřadných os. Jejich výpočet se navíc skládá z menšího počtu operací než u násobení matic. Své využití si našli zejména v herním průmyslu, kde je nutné rychle a efektivně počítat rotace kolem libovolné osy.

  Kvaterniony jsou čtyřdimensionálním rozšířením komplexních čísel, které se skládají z reálné části a z imaginární části. Základní formu kvaternionu q je možné psát ve tvaru (2).

$$q = s + v_1 i + v_2 j + v_3 k, \quad (2)$$

  kde s je reálná část, (v_1, v_2, v_3) je vektor a (i, j, k) jsou souřadné osy ve tvaru (1,0,0), (0,1,0) a (0,0,1). Zkráceně je možné kvaternion zapsat ve tvaru (3).

$$q = (s, v), \quad (3)$$

  kde v je třídemensionální vektor.

  Pro získání konkrétního kvaternionu je nutné znát pouze úhel φ, o který chceme daný bod rotovat, a osu n ve tvaru normovaného vektoru, kolem níž bude bod rotovat. Kvaternion q se získá pomocí vztahu (4).

$$q = cos(\varphi), n \cdot sin(\varphi). \quad (4)$$

  Stejně jako u rotačních matic, tak i při práci s kvaterniony záleží na pořadí násobení. Pravidla pro násobení dvou kvaternionů jsou zobrazena na Obr. 7 pocházejícího z [12].

	1	i	j	k
1	1	i	j	k
i	i	−1	k	−j
j	j	−k	−1	i
k	k	j	−i	−1

Obr. 7 Pravidla pro násobení kvaternionů

Z Obr. 7 je patrné, že pořadí násobení určuje znaménko výsledné osy. Například součin $ij = k$, ale při přehození pořadí je výsledek $ji = -k$. Stejně jako u komplexních čísel platí, že $i^2 = j^2 = k^2 = ijk = -1$.

Pro každý kvaternion je možné vytvořit i jeho konjugovanou formu $\bar{q}$ takovým způsobem, že se změní znaménka u imaginární části kvaternionu. Výsledkem je tedy vztah (5).

$$\bar{q} = q_0 - v_1 i - v_2 j - v_3 k = (s, -v) \tag{5}$$

Pro získání nových souřadnic po rotaci kolem osy je potřeba provést operaci (6).

$$\bar{u} = quq\bar{} \tag{6}$$

kde u jsou původní souřadnice bodu, který chceme rotovat kolem počátku souřadného systému a $\bar{u}$ jsou nové souřadnice po provedení rotace. Osa, kolem které bude bod rotovat, je definována jako třídimensionální vektor (v_1, v_2, v_3) uvedený ve vztahu (5). Tento vektor musí být normovaný. Úhel φ, o nějž bude bod rotovat kolem dané osy n, je nutné zadat poloviční. To je dáno vlastností kvaterninoů, protože po provedení vztahu (6) orotují bod o dvojnásobek zadaného úhlu. Výsledný kvaternion použitý pro výpočet rotace byl ve tvaru (7).

$$q = cos(\varphi/2) \ + n \cdot sin(\varphi/2). \tag{7}$$

1.5 Interpolace pohybu

Z důvodu optimalizace a uchovávání co nejmenšího počtu dat, byla provedena interpolace pohybu mezi snímky videí, které obsahovaly jednotlivé znaky. Interpolace pohybu zároveň umožní vyšší plynulost celého pohybu, což je pro uživatele mobilní aplikace příjemnější, než kdyby byl pohyb sekaný.

Způsobů interpolace pohybu je celá řada. Pro účely této práce bylo nutné použít interpolace rotace, protože všechny pohyby vykonávané modelem jsou právě rotace. Způsoby, jak interpolovat rotační pohyb, jsou v podstatě dva:

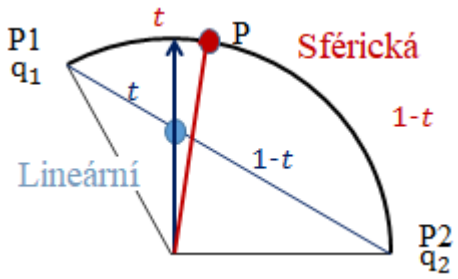
- **Lineární interpolace**

První možností interpolace pohybu je použití lineární interpolace, která počáteční a koncový bod proloží přímkou. Tuto interpolaci lze vyjádřit vztahem (8).

$$P = P_1 + (P_2 - P_1)t, \tag{8}$$

kde P_1 je počáteční bod interpolace, P_2 je koncový bod interpolace, P je aktuální bod a t je parametr interpolace. Tento parametr se pohybuje v rozmezí od 0 do 1 (0 % až 100 %). Jestliže je parametr $t = 0$, poté bude aktuální bod P shodný s počátečním bodem P_1. Pokud bude naopak platit, že $t = 1$, budou souřadnice aktuálního bodu P shodné s koncovým bodem P_2. Doplněk k parametru t se dá vyjádřit jako $1 - t$.

Nevýhodou tohoto způsobu interpolace je, že nezachovává délku ramene, jak je ukázáno na Obr. 8. Aktuální bod P by se pohyboval po modré přímce zobrazené na Obr. 8, což by způsobovalo nepřehlednost v pohybu modelu.



Obr. 8 Interpolace pohybu

- **Sférická lineární interpolace**

Druhým způsobem interpolace je použití sférické lineární interpolace (v anglickém jazyce se používá zkratka SLERP). Jedná se o interpolaci, při níž se zachovávají vzdálenosti a interpolace probíhá po oblouku jednotkové koule [13].

Interpolace sférickou lineární transformací je na Obr. 8 zobrazena černě.

Sférickou interpolaci je vhodné použít za pomoci kvaternionů, které jsou na Obr. 8 označeny q_1 a q_2. Pro získání souřadnic bodu P z Obr. 8 je třeba použít vztah (9).

$$P = (q_2\overline{q_1})^t q_1, \tag{9}$$

kde $\overline{q_1}$ je konjugovaná forma kvaternionu popsaná vztahem (5). Umocnění parametrem t lze zapsat pomocí vztahu (10).

$$q^t = \cos(t\varphi) + n \cdot sin(t\varphi), \tag{10}$$

kde n je osa rotace kvaternionu, φ je úhel, o který bude bod rotovat a parametr t má stejný význam jako u lineární transformace.

Výhodou použití sférické lineární interpolace je fakt, že interpolace probíhá vždy po nejkratším oblouku a konstantní úhlovou rychlostí ω. Krok interpolace je možné nastavovat pomocí inkrementů parametru t pro získání hladšího a plynulejšího pohybu.

1.6 Tvorba mobilní aplikace

Pro vytvoření mobilní aplikace bylo nutné transformovat soubor s koncovkou .py na soubor s koncovkou .apk. Mobilní zařízení používající operační systém Android potřebují mít instalační balíčky právě s touto příponou pro správné spuštění. Z nabídky dostupných nástrojů pro transformaci souboru na příslušnou příponu byl použit framework Kivy [14] společně s nástrojem Buildozer.

Kromě tohoto frameworku je možné použít i jiné nástroje pro tvorbu mobilních aplikací. Jednou z variant je například použití nástroje Android studio. V rámci něj je možné budovat mobilní aplikace především v programovacím jazyce JavaScript v kombinaci s dalšími jazyky. Jelikož bylo nutné do aplikace naimportovat 3D model, který je ve formátu .obj, nebyl tento způsob zvolen, protože správné nakonfigurování importovaného souboru by bylo složitější.

Použití framework Kivy umožňuje spustit program s příponou .py na zařízeních používajících rozdílné operační systémy. Kromě již zmíněného operačního systému Android se může jednat o systémy MAC OS, iOS a Linux. Samotný framework Kivy je vytvořen pomocí jazyka Python s kombinací dvou hlavních knihoven, PyGame a OpenGL.

Výhody tohoto frameworku jsou následující. Některé z nich nebyly přímo v aplikaci využity, ale je dobré je zmínit.

- **Umožňuje multitouch**

  Frameworků pro tvorbu mobilních aplikací a her je více (například PyQt), ale Kivy má oproti nim výhodu v tom, že umožňuje multitouch, který je v některých mobilních hrách nutný.

- **Jeho použití je zdarma**

  Stejně jako u softwaru Blender je možné framework Kivy podporovat malými dary, nebo se stát oficiálními sponzory. Komunita Kivy je ale méně početná, než je tomu u Blenderu.

- **Vlastní designový jazyk**

  V rámci používání tohoto frameworku je možné využít i designový jazyk, který umožňuje oddělení logiky aplikace od jejího vzhledu. Podobnou roli hraje CSS u programování webových stránek. Soubor vzniklý tímto designovým jazykem má příponu .kv a jeho naimportování do hlavního programu v jazyce Python je jednoduché.

Použití frameworku se sebou nese i nevýhody, které byly při vývoji aplikace citelné.

- **Vlastní designový jazyk**

  Tato položka byla uvedena i mezi výhodami, protože je přehlednější mít oddělený vzhled aplikace a její logiku. Na druhou stranu bylo těžké správně napsat program v tomto speciálním jazyce. Všude, kde by se v jazyce Python psal znak "=", je nutné použít dvojtečku. Zároveň není možné zobrazit si nápovědu ke kódu, načež navazuje další nevýhoda.

- **Poměrně malá komunita uživatelů**

  Oficiální web frameworku Kivy nabízí několik ukázkových příkladů pro řešení různých situací. Tyto příklady jsou však velmi okrajové a zdaleka nepokryjí všechny problémy, které mohou nastat.

  Jeden z ukázkových příkladů se týká i zobrazování 3D modelů na obrazovku [15]. Základ tohoto programu byl využit i v mobilní aplikaci, ale 3D model v tomto příkladu je umístěn v popředí a nedá se s ním manipulovat. Správné řešení bylo tedy nutné hledat na diskuzních fórech, ve kterých počet příspěvků k tomuto frameworku není mnoho.

Následná transformace na soubor s příponou .apk byla provedena pomocí nástroje Buildozer. K jeho spuštění je nutné použít operační systém Linux, nebo alespoň podsystém Linux. Ten je možné spustit z operačního systému Windows. Buildozer celý proces transformace souboru na příponu .apk zautomatizuje i se stažením potřebných položek, jako je python-for-android, Android SDK (Software Developer Kit) a Android NDK (Native Developer Kit).

V průběhu transformace je zároveň vytvořen soubor s příponou .spec, který umožňuje upravit vlastnosti aplikace. Je možné upravit ikonu aplikace, její název, knihovny použité v programu Python a další. Rychlost procesu transformace je silně ovlivněná výkonností počítače, ale napoprvé je čas transformace kolem 40 minut.

2 Rozbor cílů

V této kapitole budou podrobněji rozebrány cíle diplomové práce, které se nachází v zadání. Prvním cílem této práce je srovnání již dostupných aplikací z mnoha hledisek. Pro srovnání budou použity nejen recenze ostatních uživatelů, ale i mé vlastní poznatky vycházející z týdenního používání aplikací. V rešeršní části práce byly popsány nástroje, s jejichž pomocí bude dosaženo dalších cílů práce. Bude vysvětleno, proč byl zvolen přístup 3D modelu člověka místo přehrávání připravených videí. Cílem práce bylo také vytvořený 3D model rozpohybovat. Detailnější popis této problematiky bude popsán v kapitole Inverzní kinematika. Posledním cílem bylo vytvoření mobilní aplikace, která by mohla sloužit pro výuku znakového jazyka.

Popište dostupné aplikace pro výuku znakového jazyka

Všechny dostupné mobilní aplikace využívají pro výuku znakového jazyka natočená videa, kde lektor dané znaky předvádí. Natočená videa však zabírají velké množství úložného prostoru zařízení. Z toho důvodu jsou tato videa uložena mimo samotnou aplikaci, která k úložišti pouze přistupuje. Aplikace vytvořená v této práci využívá pro zobrazování znaků textové soubory, jež jsou mnohonásobně menší než uložená videa. Další nevýhodou některých aplikací jsou vyskakovací oznámení, která uživateli připomínají, že si daný den znakový jazyk ještě neprocvičil. Tento a další problémy budou popsány v kapitole Dostupné aplikace.

Vytvořte 3D model člověka pro potřeby prezentace znakového jazyka

Druhým cílem diplomové práce bylo vytvořit 3D model člověka, který bude v rámci aplikace předvádět znakový jazyk. Tento model byl vytvořen v programu Blender popsaném v podkapitole Tvorba modelu. Byla vytvořena pouze horní polovina těla, spodní polovina těla není pro znakový jazyk podstatná. Důvodem využití programu Blender bylo, že je následně možné pro každý uzel modelu předepsat podle jaké entity (kosti) se má pohybovat a jak moc bude danou kostí ovlivněn. V případě, že uzel bude ovlivňován více kostmi zároveň, bude výsledná souřadnice uzlu vypočítána jako vážený průměr souřadnic uzlu po rotacích jednotlivými kostmi. Tohoto by nebylo možné dosáhnout v programech jako je Inventor, SolidWorks a podobně, proto musel být použit grafický software. Dalším důvodem zvolení tohoto softwaru bylo jednoduché propojení s programováním v jazyce Python.

Vytvořený model člověka rozpohybujte dle parametrického popisu

Pro potřeby znakového jazyka je nutné mít kvalitně popsanou inverzní kinematiku. Některé znaky vyžadují sevření ruce v pěst, jiné pouze pokrčení prstů a další jsou kombinací pohybu obou rukou. Pro matematický popis takových pohybů je nutné zvolit jiný nástroj než Eulerovy úhly nebo rotační matice. Za určitých okolností u nich hrozí i ztráta jednoho stupně volnosti. Z toho důvodu byly pro popis inverzní kinematiky zvoleny kvaterniony. Jejich výhodou je možná rotace kolem jakékoli osy bez ohledu na to, zda se jedná o souřadnicovou osu. Kombinací kvaternionů aplikovaných na jednotlivé klouby v lidském těle je možné vytvořit jednotlivé znaky. Parametrický popis, který udává úhel a osu daného kloubu, je v aplikaci uložen pomocí textových souborů, jejichž velikost je zanedbatelná.

Vytvořte aplikaci pro výuku znakového jazyka

Posledním cílem diplomové práce je vytvoření aplikace sloužící pro výuku znakového jazyka. Tato aplikace bude určena především pro mobilní zařízení používající operační systém Android. Musí tedy mít příponu .apk. Zároveň funguje i na všech počítačích, které mají nainstalovaný Python a nezbytné knihovny. Mobilní aplikace bude naprogramovaná v jazyce Python s využitím knihovny Kivy a následně převedena do takové podoby, aby byla spustitelná na mobilních zařízeních pomocí frameworku Buildozer. Při vytváření aplikace byl brán ohled na již dostupné aplikace, z nichž byla čerpána inspirace. Při jejím návrhu bude snaha o eliminaci špatných vlastností již existujících aplikací. Veškerá data potřebná pro její chod budou součástí souboru s příponou .apk, tudíž nebude nutné připojení k internetu, ověřování uživatele, ani žádná další opatření.

3 Dostupné aplikace

Součástí této kapitoly je představení dosavadních mobilních aplikací, jejichž stáhnutí je zdarma. Placené aplikace v této kapitole chybí. Budou ukázány aplikace jak na český znakový jazyk, tak na americký znakový jazyk (ASL), na který bylo již vyvinuto větší množství aplikací. Práce je však zaměřena na český znakový jazyk, proto bude z výukových aplikací ASL uvedena pouze jedna, Pocket sign. Z českých mobilní aplikací budou představeny dvě, Tichý jazyk a Znakuj s Tamtamem. Aplikace budou hodnoceny z hlediska dostupných recenzí a poznatků získaných jejich používáním v období jednoho týdne.

Pocket Sign

Pomocí aplikace Pocket Sign je možné učit se americký znakový jazyk, stejně jako britský znakový jazyk (BSL). Tyto dva jazyky jsou si velmi podobné stejně jako v mluveném slově. V aplikaci je nahráno přes 1000 videí s přehledným popisem. Kromě samostatných slov obsahuje aplikace i některé fráze, znakovou abecedu a základní číslovky. Ke každému znaku je v pár slovech napsáno, jak si znak lépe zapamatovat (vysvětlení pohybu rukou).

Pocket Sign umožňuje uživatelům filtrovat slova podle kategorie, úrovně obtížnosti a abecedního pořadí. Aplikace rovněž obsahuje funkci pro vyhledávání a ukládání oblíbených slov a frází pro pozdější opakování.

Pro odemčení všech funkcí této aplikace je však nutné platit poplatek. Verze, která je zdarma, je však dosti obsáhlá, tudíž pro naučení základů postačuje. Aplikaci je možné stáhnout na mobilní zařízení používajících operační systém Android i iOS.

Tichý jazyk

Aplikace Tichý jazyk vznikla jako diplomová práce studenta ČVUT [16], z toho důvodu je velmi dobrý přístup k pozadí této aplikace. Aplikace využívá předtočená videa uložená na serveru, což významně snižuje její celkovou velikost. Zároveň je možné aplikaci spustit i z jejího webového prostředí.

Aplikace je rozdělena do modulů s rozdělením do jednotlivých kapitol. V nich si uživatel může učit danou kategorii znaků. Příklady kategorií jsou Seznámení, Číslovky, Měsíce a datum, Vlastnosti a další. Znakový jazyk má odlišnou gramatiku než mluvené slovo a aplikace Tichý jazyk je schopna rozložit zadanou větu pomocí gramatiky znakového jazyka. Součástí aplikace je i slovník všech dostupných znaků.

Drobnou nevýhodou této aplikace je nutnost přihlášení, což může některé uživatele odradit od jejího používání.

Znakuj s Tamtamem

Tato aplikace byla vytvořena v rámci projektu "TamTam", jehož cílem je zkvalitnit komunikaci mezi neslyšícími a slyšícími. Její vývoj byl spolufinancován grantem z programu "Inovační vzdělávací projekty" Ministerstva školství, mládeže a tělovýchovy. Aplikace je kompletně zdarma a je možné používat veškeré její režimy.

Stejně jako předchozí zmíněné aplikace jsou znaky ukazovány pomocí předtočených videí, jejich počet je přes 600. Tato aplikace se od ostatních odlišuje tím, že umožňuje uživatelům přímo komunikovat s neslyšícími a tím si získává na autentičnosti. Kromě čelního pohledu je možné sledovat předváděné znaky i z boku, což je u znakového jazyka důležitým aspektem.

Aplikace je také určena pro rodiny, jejichž členem je neslyšící dítě. Je proto vhodná pro společné učení rodičů i dětí. U každého znaku jsou dvě tlačítka, aby se mohli učit rodiče s dětmi dohromady. Každý z nich může zvolit, jestli už daný znak ovládá nebo ne. Tím je možné učení více osob přes jeden mobilní telefon. Její drobnou nevýhodou může být relativně malý počet slov obsažený ve slovníku aplikace.

4 Realizace

V této kapitole bude detailně popsáno, jaký byl zvolen postup při vývoji mobilní aplikace. V první podkapitole bude představen proces tvorby modelu člověka společně s jeho exportováním, aby bylo možné ho ovládat i pomocí skriptu napsaného v jazyce Python. Dále bude představen celý software pohybující modelem. Na závěr bude představena mobilní aplikace se svými dvěma režimy.

4.1 3D model člověka

Základem celé aplikace je model předvádějící znakový jazyk. Byl vytvořen v programu Blender, který byl popsán v podkapitole Tvorba modelu.

4.1.1 Realizace modelu

Základ modelu byl přebrán z [17]. Jedná se o volně stažitelný model celého člověka. Tento model byl následně upraven tak, aby měl menší velikost a byl jednodušší pro matematický popis.

Prvním zjednodušením bylo odstranění spodní poloviny těla. Tato část není pro znakový jazyk důležitá a ve výsledné aplikaci není ani zobrazená, pro využití aplikace tedy není relevantní. Další úpravou bylo odstranění pomocných uzlů nacházejících se uvnitř modelu. Tyto uzly sloužily původnímu tvůrci při tvorbě pro dodržení například tloušťky rukou, reálné délce prstů a podobně. Tyto uzly se ve velké míře nacházely také v obličejové části, kde označovaly pozici očí, zubů a uší. Po odstranění všech těchto uzlů model obsahoval stále více než 23 000 uzlů.

Další optimalizace modelu byla již z důvodu rychlosti programu, který jim hýbe. Velký počet uzlů tvořil velké množství detailů, jež pro výslednou aplikaci nebyly třeba. Z toho důvodu byla naprostá většina těchto uzlů smazána a jejich hustota tím zmenšena. Po odstranění nadbytečných uzlů se jejich finální počet dostal na 4412. Každý z nich má svůj index a dané souřadnice v prostoru. Tyto souřadnice byly měněny tak, aby dobře simulovaly pohyb celého modelu podle předlohy.

Hustota uzlů byla na určitých místech rozdílná. V oblasti zad a týlu hlavy je hustota velmi malá, protože tyto části modelu ani nemohou být zobrazeny ve finální aplikaci. Menší hustota uzlů je také v oblastech, kde nemůže dojít k ohybu vlivem rotace kloubů. Příkladem takové oblasti je například předloktí a články prstů, kde je dostačující menší počet uzlů. Větší hustota uzlů je potřeba v místech nacházejících se na rozmezí mezi kostmi. Tyto uzly jsou ovlivňovány více kostmi a pro dokonalejší simulaci pohybu je vyšší počet uzlů nutností. Rozdílné hustoty uzlů jsou zobrazeny na Obr. 9, kde jednotlivé uzly jsou reprezentovány černými tečkami.

Obr. 9 Hustota uzlů v modelu

Na výsledný model byla následně použita textura přiřazující každému uzlu z 3D modelu souřadnice z 2D snímku. Jako textura byla použita koláž, jejíž obrázky pochází z [18] a [19]. Tato koláž je vidět na Obr. 10.

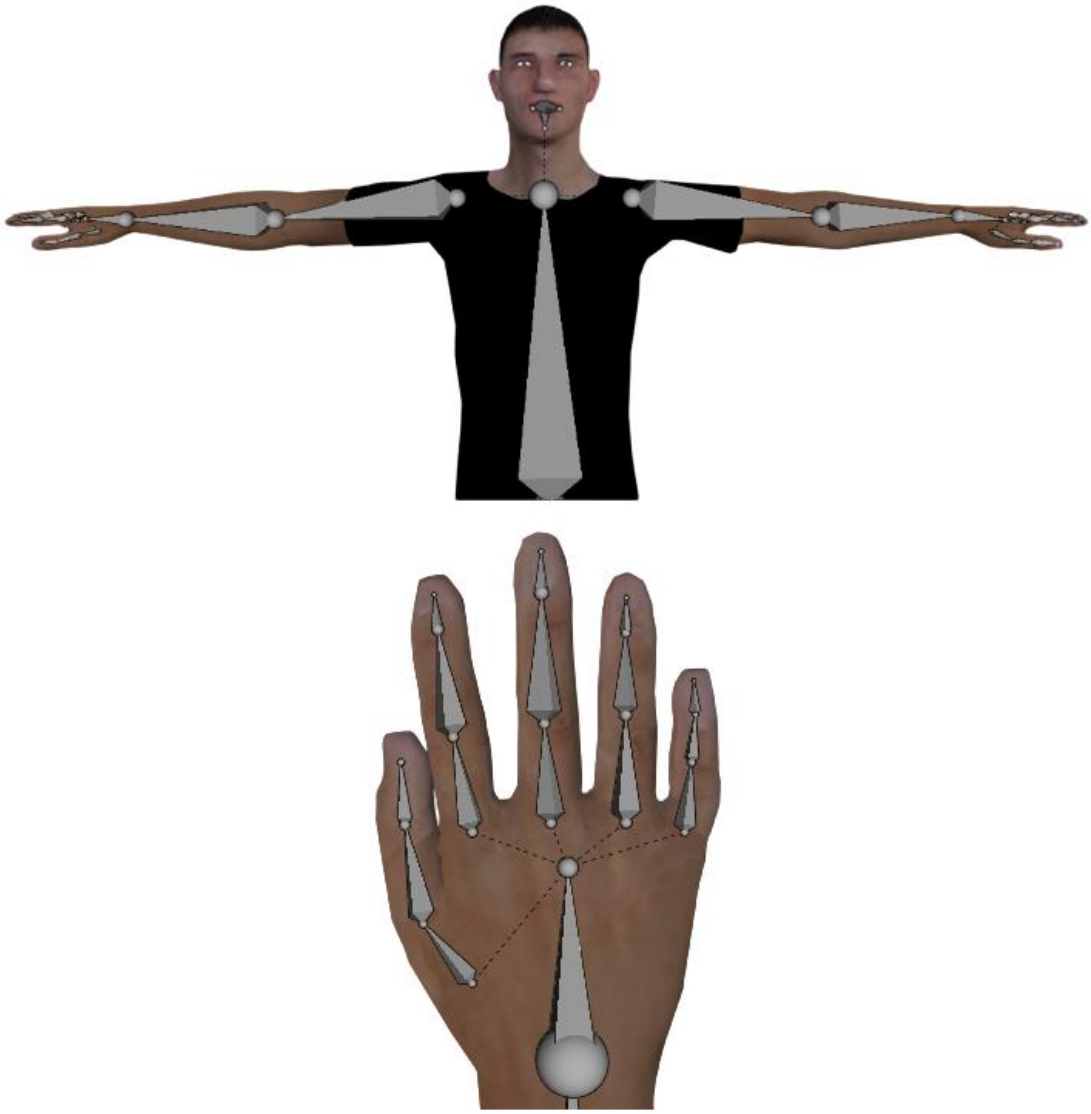
Obr. 10 Textura modelu

4.1.2 Kosti modelu

Znakový jazyk, stejně jako jakýkoli jiný jazyk, je třeba interpretovat s dostatečnými detaily, aby oslovený věděl, co se mu dotyčný snaží sdělit. U znakového jazyka jsou rozdíly mezi slovy často pouze nepatrné. Příkladem mohou být slova dům a skříň. Znaky pro obě tyto slova vypadají téměř identicky, liší se pouze v pokrčení dvou prstů na obou rukou. Vytvořený model člověka bylo potřeba rozpohybovat takovým způsobem, aby neslyšící dokázal rozeznat, o jaké znaky se jedná. Popřípadě aby mohl být rozpohybovaný model člověka využit pro výuku znakového jazyka.

Pro rozpohybování modelu člověka do něj byly v Blenderu přidány objekty kostí. Cílem těchto kostí bylo napodobovat anatomii člověka v takové míře, aby výsledné pohyby vypadaly reálně. Jednotlivé kosti jsou charakterizovány souřadnicemi počátku a konce a také svým natočením. Počátky a konce všech kostí byly vybrány tak, aby co nejvíce odpovídaly rozmístění v lidském těle. V anatomii modelu a člověka jsou určité rozdíly, o nichž bude zmínka v podkapitole Rozdíly v anatomii.

V celém modelu se nachází 41 kostí definující pohyby jednotlivých uzlů modelu. Všechny vytvořené kosti jsou vidět na Obr. 11 společně s modelem člověka. Jak je vidět na Obr. 11, tak prsty modelu jsou tvořeny vždy třemi články, což umožňuje modelovat plnohodnotné pohyby prstů, které jsou ve znakovém jazyce velmi důležité.



Obr. 11 Kosti modelu

Každá z kostí má 3 stupně volnosti tvořené rotacemi kolem všech souřadných os x, y, z. Jakékoli translační pohyby nejsou umožněny. Některé kosti ve skutečnosti nemohou rotovat kolem všech os, ale například pouze kolem jedné. To, aby kosti nerotovaly kolem pro ně nereálných os, zajišťují vstupní data z Mediapipe pocházející z reálných pohybů člověka. Tím pádem nemůže nastat situace, kdy by kost rotovala kolem nesprávné osy.

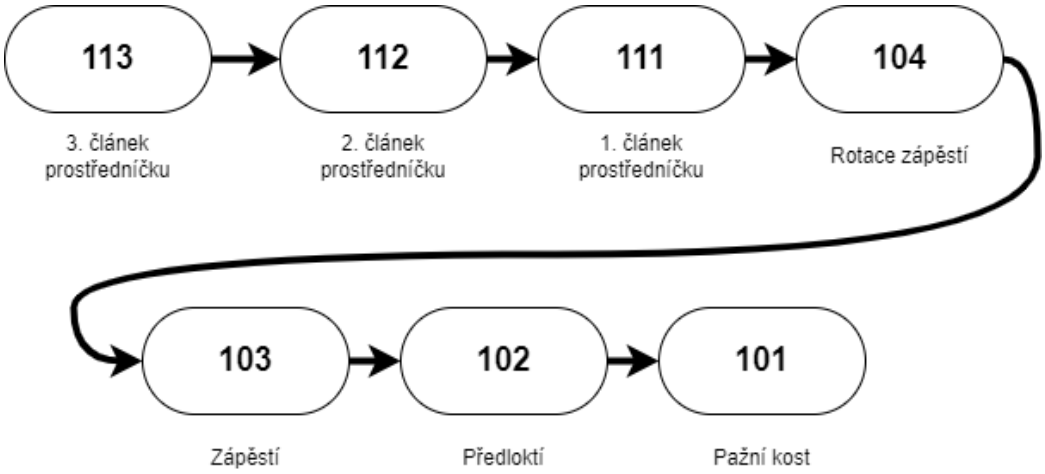
V modelu byl zaveden princip nadřazených kostí stejně tak, jak je tomu v anatomii lidského těla. Každá kost, pro příklad uveďme kosti předloktí, je závislá na svých nadřazených kostech. V našem případě by se jednalo o kost pažní. V případě, že podřazená kost změní své souřadnice v důsledku pohybu kosti nadřazené, je zřejmé, že stejným způsobem musí změnit své souřadnice i všechny uzly vázané k dané podřadné kosti.

V programu při požadavku rotace kosti kolem nějaké osy je nutné mít přístup k jednotlivým kostem. Za tím účelem byla vytvořena číselná reprezentace každé kosti. Tento číselný systém je zobrazen v Tab. 1. Z ní je patrné, že každá kost je reprezentována

třímístným číslem. Kosti pravé ruky začínají číslem 1 (od 101 do 119). Kosti levé ruky začínají číslem 2 (od 201 do 219) a kosti umístěné v obličeji začínají číslem 3 (od 301 do 303).

Kosti končící číslovkou 4 (104 a 204) reprezentují rotaci předloktí a zápěstí kolem jejich podélných os. Jelikož se tyto osy liší od os kostí 102 a 103, byla vytvořena tato nová, imaginární kost.

Vyvinutý číselný systém reprezentace kostí je také výhodný při hledání nadřazených kostí. Pro nalezení všech nadřazených kostí je pouze nutné dekrementovat číslo kosti, než je docíleno čísla končícího číslicí 1, tedy kosti pažní. Tento styl určování nadřazených kostí funguje až na výjimku, kdy je dosaženo konce prstů, které jsou reprezentovány čísly končícími na 05, 08, 11, 14 a 17. U těchto kostí je nutné přeskočení až na číslo končící 04, aby byly přeskočeny kosti, které s daným pohybem nemají být spojeny. Příklad tohoto přeskoku je zobrazen na Obr. 12.



Obr. 12 Postup určování nadřazených kostí

Na Obr. 12 je zobrazen postup hledání nadřazených kostí pro kost 113. Toto číslo se postupně snižuje až na hodnotu 111, na kterou je potřeba navázat číslem 104. Od tohoto čísla je pokračováno v dekrementování až do hodnoty 101.

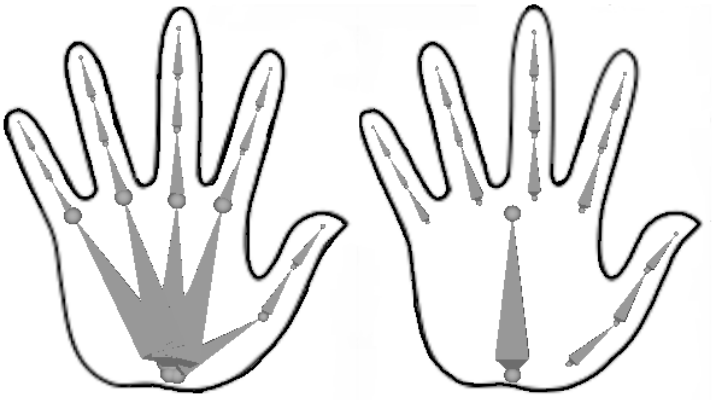
Pažní kost pravé ruky	101	Pažní kost levé ruky	201	Pohyb čelisti dolů	301
Předloktí pravé ruky	102	Předloktí levé ruky	202	Pohyb pravého koutku úst	302
Zápěstí pravé ruky	103	Zápěstí levé ruky	203	Pohyb levého koutku úst	303
Rotace pravé ruky	104	Rotace levé ruky	204		
1. článek palce pravé ruky	105	1. článek palce levé ruky	205		
2. článek palce pravé ruky	106	2. článek palce levé ruky	206		
3. článek palce pravé ruky	107	3. článek palce levé ruky	207		
1. článek ukazováčku pravé ruky	108	1. článek ukazováčku levé ruky	208		
2. článek ukazováčku pravé ruky	109	2. článek ukazováčku levé ruky	209		
3. článek ukazováčku pravé ruky	110	3. článek ukazováčku levé ruky	210		
1. článek prostředníčku pravé ruky	111	1. článek prostředníčku levé ruky	211		
1. článek prostředníčku pravé ruky	112	1. článek prostředníčku levé ruky	212		
1. článek prostředníčku pravé ruky	113	1. článek prostředníčku levé ruky	213		
1. článek prsteníčku pravé ruky	114	1. článek prsteníčku levé ruky	214		
2. článek prsteníčku pravé ruky	115	2. článek prsteníčku levé ruky	215		
3. článek prsteníčku pravé ruky	116	3. článek prsteníčku levé ruky	216		
1. článek malíčku pravé ruky	117	1. článek malíčku levé ruky	217		
2. článek malíčku pravé ruky	118	2. článek malíčku levé ruky	218		
3. článek malíčku pravé ruky	119	3. článek malíčku levé ruky	219		

Tab. 1 Číselný systém kostí

4.1.3 Rozdíly v anatomii

Cílem vytvořeného modelu člověka bylo co nejlépe simulovat pohyby lidského těla. Jedná se však o model, tudíž je možné provést jistá zjednodušení. Ta by v reálném světě znamenala ztrátu integrity těla. V modelu je však možné dané změny provést. Řeč je o rozdílech v anatomii kostí člověka a softwarového modelu. Tyto rozdíly byly tvořeny za účelem menšího výsledného počtu kostí a jednoduššího parametrického popisu celého modelu.

První rozdíl v anatomii člověka a vytvořeného 3D modelu je zobrazen na Obr. 13.



Obr. 13 : Rozdíly v anatomii ruky

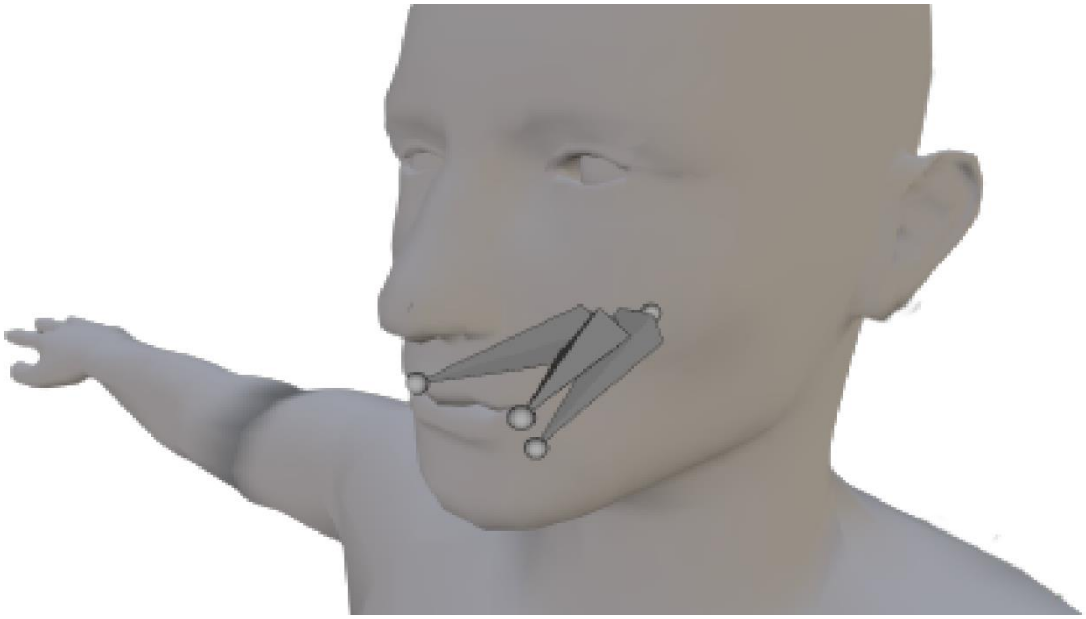
Na levé části Obr. 13 je zjednodušeně zobrazeno, jak jsou kosti uspořádané v ruce lidského těla. Všechny záprstní kosti vycházející ze zápěstí vedou ke kořenům prstů. Odtud následují 3 články ke konci prstu, u palce jsou to pouze 2 články.

Na pravé části Obr. 13 je zobrazeno uspořádání kostí v modelu člověka. Na první pohled je zřejmé, že některé kosti nejsou pevně spojeny. I přes to jsou však nakonfigurovány tak, aby pohyby odpovídaly reálnému pohybu a jsou virtuálně spojené se svojí nadřazenou kostí, jednou kostí záprstní. Druhým rozdílem v této konfiguraci je, že palec má 3 články místo 2. Důvodem přidání dalšího článku byla potřeba lépe nasimulovat pohyby palce. I přes to je však výsledný počet kostí menší než v lidské ruce a pohyb zápěstí je definován pomocí jedné kosti.

Dalším rozdílem v anatomii jsou kostí předloktí. Lidské tělo se v této části skládá ze dvou kostí, vřetenní a loketní. Tyto dvě kosti umožňují rotaci zápěstí podél jeho podélné osy. V modelu člověka byla použita pouze jedna kost, jejíž cílem je pouze spojovat loket se zápěstím, tato kost je také zobrazena na Obr. 11.

Poslední změna v anatomii modelu a lidského těla byla provedena za účelem nasimulování pohybu koutků úst. V lidském těle jsou tyto pohyby tvořeny pomocí mimických svalů rozmístěných v obličejové části hlavy. Z důvodu zjednodušení popisu pohybu modelu byly místo svalů přidány dvě kosti, které jsou zobrazeny na Obr. 14. Tyto kosti v lidském těle vůbec nefigurují. Kořeny těchto dvou kostí jsou zhruba v místech

počátku horní čelisti. Uzly navázané na tyto kosti tvoří spolu s dolní čelistí mimiku modelu. Na Obr. 14 je zobrazena i kost pohybující dolní čelistí ve svislém směru.



Obr. 14 Přidané kosti v obličejové části hlavy

4.1.4 Ukládání vah

Jednotlivé uzly modelu člověka jsou k příslušným kostem vázány pomoci normovaných vah. Pro každý uzel musí platit, že součet všech vah od všech kostí ovlivňujících daný uzel musí být roven 1. V modelu se nachází přes 4 000 uzlů, z nichž ne všechny jsou spojeny s některou z kostí. Takové uzly se nachází například na hrudi či na hlavě. Souřadnice těchto uzlů se tedy během jakéhokoli pohybu nezmění.

Na Obr. 15 je pro ukázku zobrazena příslušnost jednotlivých uzlů ke kosti, která nahrazuje kosti vřetenní a loketní. Na tomto obrázku jsou kosti reprezentovány pouze spojnicemi.

Obr. 15 Váhy předloktí

Jak je vidět na Obr. 15, tak váhy jsou v Blenderu reprezentovány barevným spektrem od modré po červenou. Červená barva značí uzly, jejichž pohyb je ovlivněn pouze danou kostí, tudíž mají váhu 1. Modré uzly nejsou danou kostí ovlivňovány vůbec a k dané kosti mají váhu rovnu 0. Na Obr. 15 jsou tyto uzly v oblasti zápěstí a pažní kosti. Uzly

s odlišnou barvou jsou danou kostí ovlivňovány pouze částečně. Sytost barvy značí, do jaké míry jsou danou kostí ovlivněny. Některé uzly mohou být ovlivněny více kostmi zároveň. K těmto situacím dochází na přechodu mezi kostmi, jinými slovy v místě ohybu kloubů. Na těchto místech byl vytvořen postupný gradient od červené k modré tak, aby pohyb ve výsledku odpovídal realitě.

Aby bylo možné měnit souřadnice jednotlivých uzlů pomocí programu v Pythonu, bylo nutné váhy exportovat z Blenderu v takovém formátu, aby mohly být do programu importovány. Za tím účelem byl vytvořen skript *get_weights.py* na ukládání vah do textového souboru s názvem *weights.txt*. Část tohoto souboru je zobrazena na Obr. 16.

```
['4407' '1.0' 'R_Arm' '-2.098' '0.864' '4.486']
['4408' '1.0' 'R_Arm' '-2.098' '0.864' '4.486']
['4409' '1.0' 'R_Arm' '-2.098' '0.864' '4.486']
['1332' '0.217' 'R_Elbow' '-4.941' '0.826' '4.153']
['1333' '0.194' 'R_Elbow' '-4.941' '0.957' '4.278']
['1340' '0.222' 'R_Elbow' '-4.941' '0.957' '4.278']
```

Obr. 16 Ukázka textového souboru *weights.txt*

Struktura exportovaného souboru je následující:

- první sloupec označuje index uzlu
- druhý sloupec označuje váhu k příslušné kosti
- třetí sloupec označuje název kosti v Blenderu
- čtvrtý až šestý sloupec jsou souřadnice počátku kosti v Blenderu

Nevýhodou logování tímto způsobem je, že jsou všechny hodnoty ukládány pomocí datového typu string a zabírají tedy více místa než ukládání v číselných hodnotách.

Obdobným způsobem jsou logovány i váhy sloužící k rotaci podél podélné osy předloktí. Tyto váhy musely být odděleny od hlavního textového souboru *weights.txt*. Důvodem je, že rotace podél podélné osy ovlivňuje určité uzly jinou mírou. Pro přehlednost tedy byly vytvořeny dva nové textové soubory, *weights_forearm.txt* a *weights_wrist.txt*. Jak už název napovídá, tyto soubory obsahují váhy pro rotaci předloktí a zápěstí kolem podélné osy. V těchto textových souborech už není nutné uchovávat souřadnice počátku kostí. Na Obr. 17 je vidět část textového souboru *weights_forearm.txt*.

```
[1.332e+03 2.170e-01]
[1.333e+03 1.950e-01]
[1.34e+03 2.22e-01]
[1.341e+03 3.200e-01]
```

Obr. 17 Ukázka textového souboru *weights_forearm.txt*

Struktura těchto dvou nových souborů je podobná jako struktura souboru *weights.txt*. Obsahují však pouze dva sloupce, index uzlu a váhu k dané kosti. Žádná další informace není třeba. Při exportování z Blenderu došlo k tomu, že všechny údaje byly zapsány pomocí mocniny čísla 10. To vedlo k vytvoření nové funkce, která vyčítá data z těchto dvou textových souborů.

4.1.5 Export modelu

3D model člověka vytvořený v programu Blender bylo potřeba exportovat takovým způsobem, aby jeho pohyby byly řízeny pomocí příkazů v jazyce Python. K vizualizaci pohybů byla využita knihovna OpenGL. Toto rozhraní se využívá při vývoji CAD systémů, počítačové grafiky do počítačových her, ve virtuální realitě nebo pro vizualizaci.

Samotný model člověka byl vyexportovaný ve formátu .obj. Ten v sobě nese informace o všech uzlech (bodech v prostoru), texturách, normálách ploch, spojení uzlů do ploch a také o použitém materiálu. Tento formát není schopen přenášet informace o animacích modelu, proto musely být tyto animace naprogramované až v Pythonu.

Informace o materiálu ani o normálách ploch nebyly ukládány, protože tyto informace jsou použity pro nasvícení scény, což nebylo v této práci provedeno. Místo materiálu byly použity textury z 2D obrázků, které byly nasuperponovány na 3D model člověka. Souřadnice všech uzlů jsou třídimenzionální a řádek každého nového uzlu začíná označením „v“. Textury jsou dvourozměrné, protože popisují polohu uzlu na 2D obrázku, řádek je označen „vt“. Všechny plochy nesou označení „f“ a dávají informaci o tom, které uzly tvoří danou plochu a jaké texturové souřadnice mají být použity pro vykreslení. Všechny tyto parametry ploch jsou odděleny lomítkem. Ukázka souboru .obj vyexportovaného z Blenderu je zobrazena na Obr. 18.

```
v -1.068753 -0.590777 3.083771
v -1.083476 -0.598345 2.933036
vt 0.168615 0.110605
vt 0.168338 0.111642
f 1/1 3308/2 3/3
f 4/4 2/5 1/6
```

Obr. 18 Ukázka .obj souboru

Na straně Pythonu musely být tyto informace předány OpenGL v podobě bufferu, který obsahoval informace z .obj souboru. Do bufferu byly informace řazeny podle informací o plochách (řádky začínající písmenem “f”). Na Obr. 18 v řádku začínajícím písmenem “f” je první údaj 1/1. To znamená, že do bufferu byly napřed uloženy prostorové souřadnice uzlu umístěného v textovém souboru na prvním místě. Jinými slovy se jedná o uzel s indexovým číslem 1. Hned za těmito souřadnicemi byly do bufferu řazeny texturové souřadnice, uložené v textovém souboru na prvním místě. Další sérií souřadnic bylo podle řádku s počínajícím písmenem “f” uzel s číslem 3308 a textura s číslem 2. Takto vznikl jeden velký buffer, který byl poté předložen OpenGL.

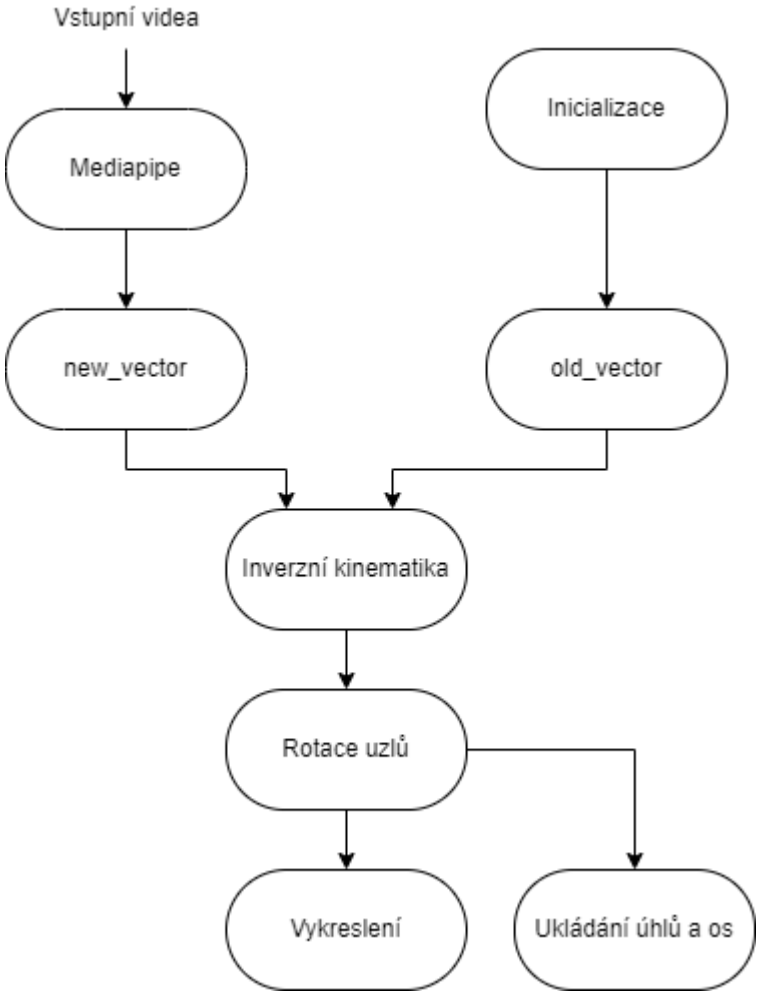
Na Obr. 18 je možné si všimnout, že uzel s indexem 1 se objevuje vícekrát. To způsobilo, že některé informace jsou v bufferu opakovaně a dělají tím velikost bufferu větší. Je to však nezbytné pro to, aby byly všechny uzly pospojovány správně a tvořily tak plochy stejně jako v programu Blender. Všechny plochy tvořící model člověka mají trojúhelníkový tvar.

4.2 Software

V rámci této diplomové práce byly vytvořeny dva programy v jazyce Python sloužící k transformaci pohybů tlumočnice na 3D model člověka. Cílem této podkapitoly je představit oba tyto programy a popsat jejich hlavní části. První z těchto programů pracuje se samotnými videi a jeho výstupem jsou textové soubory. Druhý program slouží pro manuální korekci dat uložených v textovém souboru pro lepší přednes jednotlivých znaků. V dalších částech této podkapitoly budou detailně popsány jejich hlavní části pro lepší pochopení.

1. Videos2txt.py

   První z programů vytvořených v této práci nese název *videos2txt.py*. Jak z názvu vyplývá, tak účelem je vytvořit textové soubory dané struktury (viz Obr. 24), které budou použity v mobilní aplikaci. Na Obr. 19 je zobrazeno zjednodušené schéma tohoto programu. Na vstupu tohoto programu jsou videa, jejichž zpracování způsobuje malou rychlost tohoto programu

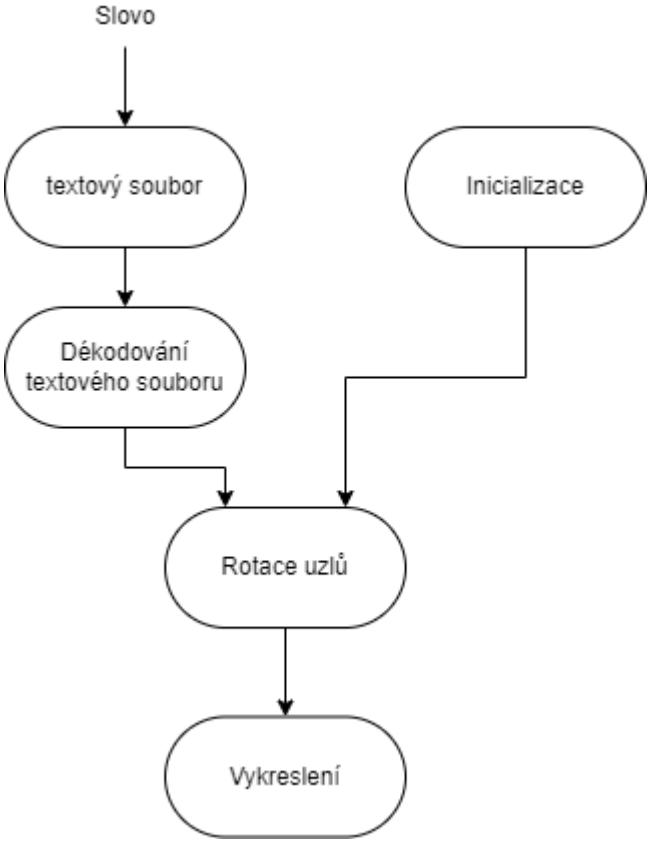


Obr. 19 Schéma programu Videos2txt.py

2. Signs_visualisation.py

Druhým programem je *Signs_visualisation.py*. Na vstupu tohoto programu je textový soubor, který může být před jeho spuštěním měněn. Tím je dosaženo optimalizace předváděných znaků a korekce chyb z programu *videos2txt.py*. Jádro tohoto programu je totožné s předchozím programem. Jediným rozdílem je jeho vstup tentokrát tvořený pouze textovým souborem. Schéma programu je zobrazeno na Obr. 20. Jak je vidět na Obr. 20, tak schéma programu *Signs_visualisation.py* je méně obsáhlé než u programu *videos2txt.py*. To má za následek i vyšší rychlost tohoto programu. Ta je ovšem také schválně omezena z důvodu lepší identifikace možných chybných informací ve vstupních textových souborech.

V dalších podkapitolách budou více rozvedeny hlavní části programů.



Obr. 20 Schéma programu Signs_visualisation.py

4.2.1 Sběr dat

Jedním ze zásadních požadavků kladených na mobilní aplikaci bylo, aby znaky předváděné modelem byly autentické. Hlavním cílem bylo, aby byl zachován účel výukové mobilní aplikace. Pohyb modelu proto musel být popsán tak, aby neslyšící osoby byly schopny rozeznat jednotlivé znaky. Při sběru dat s pomocí tlumočnice to vyžadovalo správné nastavení kamer. Pro dokonalejší záznam pohybů tlumočnice při znakování bylo nutné použít tři kamery. První z nich byla umístěná před tlumočnicí, zbylé dvě byly umístěny ze stran. Tyto dvě kamery umožňovaly zaznamenávat informace o pohybech, které první kamera nedokázala zjišťovat. Tímto způsobem byl zaznamenán pohyb tlumočnice ve všech třech souřadných osách.

Nároky na umístění kamer byly také velmi striktní. Všechny tři zmíněné kamery musely být umístěné ve stejné výšce a ve stejné vzdálenosti od tlumočnice. V případě, že by tyto požadavky nebyly splněny, došlo by ke zkreslení dat. Dalšími požadavky byly stejná vzorkovací frekvence použitých kamer a také jejich rozlišení. Z toho důvodu byly zvoleny tři mobilní telefony od stejného výrobce.

Při nahrávání videí byl kladen důraz na zvýšenou míru mimiky. Zároveň bylo důležité, aby tlumočnice byla neustále čelem k přední kameře, aby nedocházelo ke zkreslování úhlů pro tvorbu mimiky. Všechny pohyby tlumočnice navíc musely být pomalé, aby při následné analýze pomocí frameworku Mediapipe bylo snazší nalézt klíčové body a tím urychlit celý proces.

V celém průběhu videa bylo potřeba, aby byly kamery umístěny staticky ideálně bez jakéhokoli pohybu. Nakloněním mobilů kolem kterékoli osy by způsobilo zkreslení dat, což by zapříčinilo chybný výpočet úhlů a os pro jednotlivé kosti. Pohyb modelu by tím pádem byl nepřesný.

Celkem bylo natočeno 20 znaků, z nichž každý byl pro jistotu natočen dvakrát, kdyby data nebyla z nějakého důvodu v pořádku. Aby bylo možné natočená videa použít pro popis modelu, musela si navzájem odpovídat i jednotlivými snímky, aby jedno video nepředbíhalo druhé. Tato synchronizace byla časově náročnější.

Framework Mediapipe z videí vyseparoval souřadnice klíčových bodů, jež byly základem pro pohyb modelu.

4.2.2 Inverzní kinematika

Cílem této kapitoly je detailně popsat způsob, jakým je počítána kinematika modelu člověka, a proč byl zvolen tento způsob. Kinematika lidského těla je kvůli velkému počtu stupňů volnosti velmi složitá. Matematický popis kinematiky proto musel co nejvíce odpovídat realitě. Jednoduše je možné představit si lidské tělo jako víceramenný prostorový manipulátor, v němž se nachází pouze rotační vazby. Jednotlivé rotační vazby mají různý počet stupňů volnosti, jež musely být dodrženy pro dobrý popis pohybu. Články prstů stejně jako loket mají pouze jeden stupeň volnosti, zatímco rameno a zápěstí jich mají více.

Vstupní data

Data pro výpočet inverzní kinematiky byla získaná z frameworku Mediapipe popsaného výše. Z každého snímku bylo možné pomocí tohoto frameworku získat normované souřadnice klíčových bodů těla, ruky a obličeje.

Ne všechna data byla dále použita, protože pro další výpočet nebyla nutná. K jednotlivým klíčovým bodům bylo dále třeba postupně přistoupit a vytvořit normované vektory reprezentující spojnice mezi klíčovými body. Pro korektní zápis těchto vektorů bylo nutné brát v potaz i rozlišení snímků v pixelech.

Ze snímku pořízeného přední kamerou byly použity jak souřadnice x, tak souřadnice y. Ze snímku pořízeného pravou kamerou byla použita pouze souřadnice x. Tato souřadnice určovala posun dopředu respektive dozadu. Z těchto tří souřadnic byl poté vytvořen vektor popisující, jakým směrem daná kost míří. Na závěr byl vzniklý vektor znormován. Stejná situace platila i pro levou kameru.

Takto vytvořené vektory byly uloženy do proměnné typu slovník s názvem *new_vector*. Aby bylo jasné, k jaké kosti se daný vektor pojí, byl pro ně použit stejný číselný systém jako pro soustavu kostí popsaný v Tab. 1. Proměnná *new_vector* uchovává pozici, do níž se má model dostat. Výchozí pozice, ze které model začíná svůj pohyb, je uchována v proměnné *old_vector*. Tato proměnná má zavedený opět stejný číselný systém.

Úhly a osy

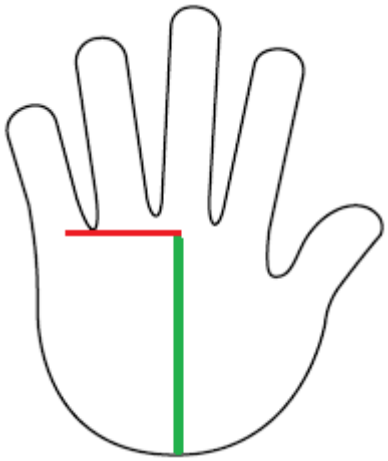
Pro každou dvojici z proměnných *old_vector* a *new_vector* bylo třeba vypočítat vzájemně svírající úhel a osu, podél níž je třeba rotovat. Kvůli složitosti nebylo možné rozložit rotaci na rotace kolem jednotlivých souřadných os poskládaných za sebe. Bylo třeba určit jednu osu definující celou rotaci.

V prvním kroku byla určena osa rotace pomocí vektorového součinu každé z odpovídajících dvojic *old_vector* a *new_vector*. Úhel mezi těmito dvěma vektory byl určen pomocí vztahu (11).

$$\varphi = \arccos\left(\frac{a \cdot b}{\|a\|\|b\|}\right) \tag{11}$$

kde a a b reprezentují použité vektory. Jelikož byly oba vstupní vektory normované, tak jmenovatel vztahu (11) nebylo třeba uvažovat. Pro osy i úhly byl pro pořádek použit opět stejný číselný systém jako pro vektory popsaný v Tab. 1.

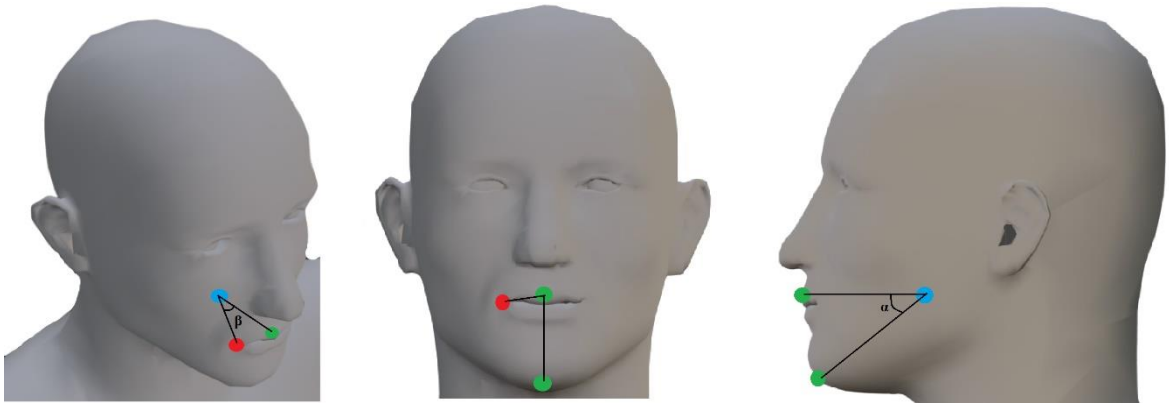
Jiný způsob byl použit pro výpočet úhlu a osy rotace reprezentující orientaci zápěstí (kosti s čísly 104 a 204). U nich je osa dána spojnicí zápěstí a začátku prostředníčku, na Obr. 21 je tato osa znázorněna zeleně. Úhel rotace je poté určen pomocí spojnice začátku prostředníčku a začátku malíčku, na Obr. 21 červeně. Při změně orientace ruky podél podélné osy se změní i směr spojnice prostředníčku a malíčku, čímž vznikne úhel rotace.



Obr. 21 Osa pro rotaci kolem podélné osy

Při modelování mimiky modelu byl použit rozdílný způsob, než jaké byly doposud popsány. Osy u třech kostí, které společným pohybem tvoří mimiku obličeje a jsou zobrazeny na Obr. 14, byly dopředu známé, z toho důvodu, že hlava modelu je statická. U výpočtu úhlů byla situace složitější.

Pro pohyb dolní čelisti byly vybrány dva vztažné body. Prvním byl vrchní okraj horního rtu a druhým spodní část brady. Tyto dva body jsou zvýrazněny na Obr. 22 zeleně. Oba zelené body jsou spojeny s modrým bodem, který představuje počátek rotace. Změna úhlu α mezi zelenými body je úhel, o jaký se změní poloha spodní čelisti ve svislém směru. Pohyb koutků rtů zajišťovaly opět dva body. První z nich byl opět vrchní okraj horního rtu a druhým byl pravý koutek úst. Na Obr. 22 je koutek úst zobrazen červeně. Počátkem této rotace byl opět modrý bod zobrazen na Obr. 22.



Obr. 22 Úhly mimiky

Pro výpočet úhlu musela být přepočítána vzdálenost v souřadném systému získaného z Mediapipe do souřadného systému samotného modelu. Souřadný systém pocházející z Mediapipe je normovaný, tudíž je možné získat vzdálenost dvou požadovaných bodů, které ale neodpovídají vzdálenosti stejných dvou bodů v modelu. Bylo nutné získat

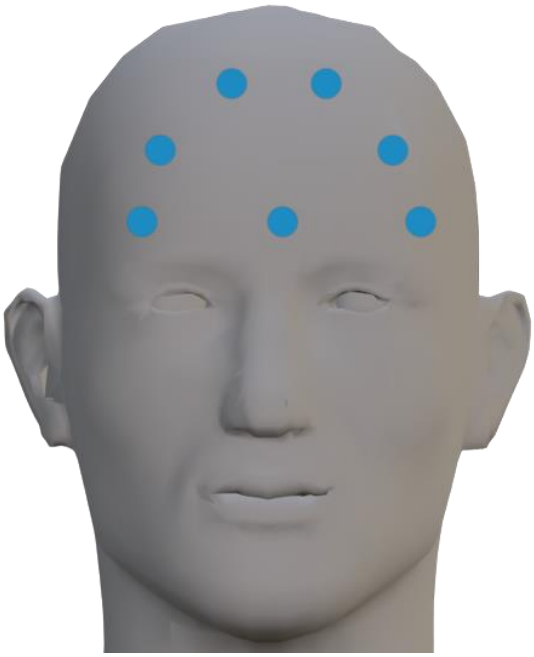
koeficient c vyjadřující přepočet mezi těmito dvěma souřadnými systémy. Matematicky by se tento problém dal zapsat podle vztahu (12).

$$d_b = cd_m, \tag{12}$$

kde d_m je vzdálenost dvou bodů v souřadném systému Mediapipe a d_b je souřadný systém modelu v Blenderu.

Proces získání koeficientu c byl následující. Vzhledem k tomu, že velikost tohoto koeficientu je závislá na vzdálenosti předvádějící osoby od kamery, tak není možné, aby byl tento koeficient konstantní. Bylo tedy vybráno 7 dalších klíčových bodů získaných z analýzy frameworkem Mediapipe nacházejících se na čele. Tyto body jsou na Obr. 23 zobrazeny modře. Následně byla vypočítána vzdálenost mezi všemi těmito body a porovnána se vzdálenostmi na modelu člověka. Každá dvojice vzdáleností byla následně vydělena mezi sebou a medián všech těchto podílů byl prohlášen za hledaný koeficient c. Výpočet tohoto koeficientu nebyl zcela přesný, protože framework Mediapipe poskytuje spolehlivě pouze souřadnice v rovině. Prostorová souřadnice obsahuje velký šum a nebyla proto použita. Naopak souřadnice v modelu jsou již třídimenzionální. Drobná chyba ve výpočtu koeficientu c však nebyla zásadní.

Díky němu už bylo možné převést všechny potřebné vzdálenosti do souřadného systému modelu a vypočítat úhly, o které se mají dané kosti orotovat.



Obr. 23 Pomocné body na čele

Násobení kvaternionů

Jak bylo řečeno v podkapitole Kinematika pohybu, při násobení dvou kvaternionů záleží na pořadí. Prohozené pořadí kvaternionů způsobí změnu znaménka výsledného kvaternionu. Při provádění interpolace pohybu SLERP podle vztahu (9) je nutné násobit dva kvaterniony. Aby nebylo nutné programovat pravidla pro násobení dvou kvaternionů zobrazená na Obr. 7, byl vytvořen jednodušší způsob popsaný vztahy (13).

$$\begin{aligned} p_1 &= r_1q_1 - r_2q_2 - r_3q_3 - r_4q_4, \\ p_2 &= r_1q_2 + r_2q_1 - r_3q_4 + r_4q_3, \\ p_3 &= r_1q_3 + r_2q_4 + r_3q_1 - r_4q_2, \\ p_4 &= r_1q_4 - r_2q_3 + r_3q_2 + r_4q_1, \end{aligned} \tag{13}$$

kde vstupní kvaterniony pro násobení jsou ve tvaru (r_1, r_2, r_3, r_4) a (q_1, q_2, q_3, q_4). Výsledný kvaternion po násobení má tvar (p_1, p_2, p_3, p_4).

4.2.3 Ukládání slov

Cílem výsledné mobilní aplikace je minimalizace její velikosti z pohledu zabraného místa v úložném prostoru. Z toho důvodu není vhodné mít videa, kde jsou jednotlivé znaky předváděny, uložena v samotné aplikaci, a proto v jejích souborech ani nejsou. Místo ukládání celých videí byl vyvinut systém ukládání klíčových údajů nesoucích v sobě nezbytné minimum informací pro správný pohyb modelu člověka.

Každý znak získaný z videí byl přetvořen do textového souboru s maximální délkou 14 řádků. Ten zabírá oproti videu zanedbatelně malou velikost. Všech dvacet textových souborů nesoucích informace o dvaceti zaznamenaných znacích je uloženo ve složce *Slova*, která je součástí mobilní aplikace.

Struktura textových souborů je následující:

- **Počet snímků, které byly přeskočeny**

  Tato hodnota je ve většině případů 10. Z důvodu urychlení programu není nutné uvažovat každý snímek videa. Při analyzování každého desátého snímku dojde k výpočtu os rotací všech kostí společně s úhly, o které mají rotovat. Zpracováním pouze těchto snímků se v datech neztratí žádné důležité informace.

- **Počet mezisnímků, které slouží k interpolaci**

  Tato hodnota je nastavena buď na 10 nebo 5 v závislosti na tom, jak rychlý byl daný pohyb na nahraném videu. V případě, že pohyb byl rychlejší, tak je nastavena na 5. Tím pádem se model pohne dvacetkrát (100 / 5), než dojde k analýze nového snímku. V případě, že byl pohyb pomalejší, je hodnota nastavena na 10 a model se pohne desetkrát (100 / 10). Tento údaj slouží k plynulejšímu pohybu modelu.

- **Úhly a osy rotací pro pravou paži**

  Popořadě pro kosti 101 až 119

- **Úhly a osy rotací pro levou paži**

  Popořadě pro kosti 201 až 219

- **Úhly a osy rotací mimiky**

  Popořadě pro imaginární kosti 301 a 302. Úhel pro kost s indexem 303 je stejný jako úhel pro kost 302. Jediné, v čem se liší tyto dvě kosti, je znaménko os. Proto není nutné mít v textovém souboru uloženy obě kosti.

Na Obr. 24 je zobrazena ukázka prvního řádku souboru pro slovo *Ahoj*. Na něm je vidět počet přeskočených snímků, počet mezisnímků pro interpolaci a osy a úhly pro kosti 101 a 102.

```
[10][10][2.097][-0.982  0.076 -0.075][-1.075][0.957 0.0 -0.287]
```

Obr. 24 Ukázka slova Ahoj

Osy v textovém souboru jsou reprezentovány pomocí normálových vektorů a úhly jsou v radiánech. Aby bylo možné použít všechny tyto textové soubory, je nutné, aby model člověka již neměl upažené paže (viz Obr. 11). Z toho důvodu byl vytvořen ještě jeden textový soubor stejné struktury, jehož cílem je změnit polohu rukou do výchozí pozice, která je vidět v aplikaci (viz Obr. 26). Tento textový soubor je pojmenován *0.txt*.

Z důvodu kompatibility s frameworkem Kivy z pohledu diakritiky byly textové soubory uloženy pomocí čísel, ne slov. Jména textových souborů s odpovídajícími znaky jsou zobrazeny v Tab. 2.

Ahoj	*1.txt*	Modrá	*11.txt*
Babička	*2.txt*	Moje	*12.txt*
Český	*3.txt*	Otec	*13.txt*
Dědeček	*4.txt*	Skříň	*14.txt*
Děkuji	*5.txt*	Slyšící	*15.txt*
Dům	*6.txt*	Stát	*16.txt*
Jablko	*7.txt*	Tričko	*17.txt*
Jazyk	*8.txt*	Věta	*18.txt*
Jméno	*9.txt*	Zelená	*19.txt*
Matka	*10.txt*	Znak	*20.txt*

Tab. 2 Jména textových souborů

Ukládání dat tímto způsobem je nejen méně náročnější na velikost než pomocí videí, ale je také efektivnější, co se týče výpočtového času. Videa by se musela načíst, zanalyzovat pomocí Mediapipe a následně by se úhly a osy pro jednotlivé kosti musely spočítat. Jedná se tedy o ušetření nemalého procenta času.

4.3 Tvorba mobilní aplikace

V této kapitole bude detailně popsáno, jak bylo dosaženo souboru s příponou .apk nutného pro správné fungování na mobilních zařízeních používajících operační systém Android. Framework Kivy umožňuje vytvořit mobilní aplikaci i pro zařízení značky Apple, na něž ale tato práce není zaměřena. Dále budou popsány dva režimy aplikace, z nichž si může uživatel zvolit, Procvičování a Slovník. U obou bude popsáno, jaké procesy běží na pozadí a tento popis bude současně sloužit jako návod k používání aplikace.

4.3.1 Kompilace aplikace

Pro vytvoření souboru s příponou .apk, který je možné spustit na mobilním zařízení, bylo nutné správně použít framework Buildozer. Tento proces je z velké části automatizován, ale je třeba správně nastavit určité parametry v souboru s příponou .spec. Tento soubor je vygenerovaný v základní podobě automaticky. Uživatel musí v tomto souboru zapsat a vyplnit následující vstupy:

- Název aplikace
- Přípony souborů, se kterými bude aplikace pracovat

  V případě této aplikace bylo potřeba pracovat s několika různými typy souborů, jejichž všechny přípony musely být uvedeny v kolonce Include extentions. Výčet přípon zahrnoval:
  - .py - skripty napsané v jazyce Python
  - .png - textura modelu
  - .jpg - ikona aplikace
  - .kv - rozvržení jednotlivých prvků aplikace
  - .txt - všechny textové soubory použité pro ukládání (Model člověka uložený s příponou .obj byl předělán na koncovku .txt.)
  - .glsl - informace pro správné vykreslení a fungování zobrazení
- Další náležitosti aplikace

  Tyto náležitosti obsahují informace o použitých knihovnách a aplikované verzi jazyka Python. Dané informace se doplňují do kolonky Requirements. V této práci bylo potřeba do této kolonky doplnit:
  - python3
  - kivy
  - regex - knihovna pro snadné hledání informací v textu, separace čísel a tak dále.
- Ikona aplikace – Cesta k obrázku, který má být použit jako ikona
- Barva pozadí při načítání aplikace - kolonka preplash_color

Soubor s příponou .spec byl následně uložen a byla spuštěna kompilace. Tento soubor se nachází v přílohách této diplomové práce. Po kompilaci je celková velikost aplikace kolem 50 MB. Aplikace by zabírala méně úložného prostoru, kdyby byla data uložena na serveru a ne přímo v mobilní aplikaci. I přes to je ale velikost aplikace přijatelná.

4.3.2 Popis aplikace

Na mobilním zařízení se aplikace ukazuje pod jménem ZnaKouč (spojení slov Znak a Kouč). Její ikona je černá silueta ruky na modrém pozadí. Byla zvolena silueta místo skutečného obrázku ruky, protože v aplikaci se nachází pouze model člověka. Ikona aplikace je zobrazena na Obr. 25.

Obr. 25 Ikona aplikace

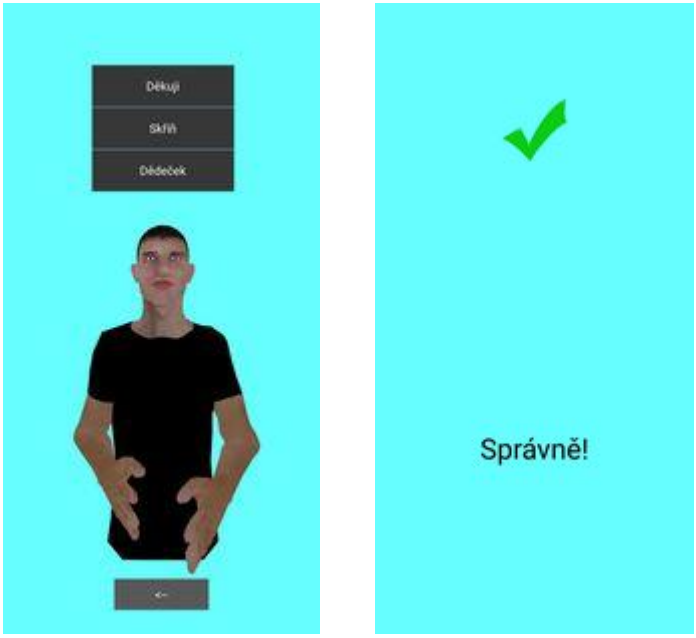
Po otevření aplikace se ikona vyobrazená na Obr. 25 zobrazí ještě ve větším měřítku, zatímco na pozadí se inicializují veškeré knihovny potřebné pro správnou funkcionalitu aplikace. Zároveň se připraví i nezbytné součásti. Po této fázi inicializace se na obrazovce s modrým pozadím objeví dvě tlačítka, Procvičování a Slovník. Jedná se o dva režimy, ve kterých tato aplikace funguje. Oba tyto režimy budou detailněji popsány odděleně.

Procvičování

Jak již název napovídá, jedná se o zdokonalování se a poznávání znaků. Ukázka tohoto režimu je zobrazena na Obr. 26.

Z celkových 20 slov jsou vždy vybrána tři slova, která jsou následně vypsána na třech tlačítkách umístěných nad sebou v horní části obrazovky. Ve vybírání těchto tří slov není aplikovaná žádná logika na základě úspěšnosti uživatele. Jedná se pouze o náhodný výběr z dostupných slov.

Po půlvteřinové pauze určené pro orientaci a přečtení navrhovaných slov začne model předvádět jeden z těchto tří znaků. Model se pohybuje s frekvencí 17 Hz. Tato frekvence je o trochu nižší, než kterou člověk vnímá jako plynulý pohyb. Pokud by byla tato frekvence však vyšší, model by provedl znak příliš rychle a bylo by velmi obtížné poznat, jaký znak byl předveden. Model předvede znak pouze jednou.



Obr. 26 Režim Procvičování

Tlačítka v horní polovině obrazovky je možné zmáčknout i když se model ještě pohybuje. Jestliže uživatel zvolí špatnou možnost, pozadí tlačítka se změní na červené. V opačném případě se jeho pozadí změní na zelené a po dokončení pohybu modelu se ukáže obrazovka zobrazená na Obr. 26 vpravo. Tato obrazovka je zobrazena po dobu dvou vteřin, poté opět zmizí a celý proces se opakuje. Jediným způsobem ukončení tohoto režimu je návrat na domovskou obrazovku aplikace stisknutím tlačítka v dolní části obrazovky. Po opětovném přístupu do režimu Procvičování model začíná z výchozí polohy s novou sadou tří slov.

Slovník

Druhý režim, který umožňuje aplikace, je pojmenovaný Slovník. Po vstupu do tohoto režimu se na obrazovce zobrazí seznam všech 20 slov uložených v aplikaci v abecedním pořadí. Tato obrazovka je zobrazena na Obr. 27.

Po kliknutí na některé ze slov se objeví model člověka a vybrané slovo předvede. Společně s modelem se objeví i předváděné slovo pro snadnější zapamatování. Tlačítko zpět zobrazené v horní části obrazovky vrátí uživatele na seznam všech slov. Model se v tomto režimu pohybuje opět s frekvencí 17 Hz. Opuštění tohoto režimu je opět možné pomocí tlačítka umístěného v dolní části obrazovky.

Obr. 27 Režim Slovník

5 Výsledky

V rámci této diplomové práce byl vytvořen 3D model člověka předvádějící jednotlivé znaky. Model se skládá z celkem 4412 uzlů rozmístěných v prostoru. Jedná se o velmi malé množství vzhledem k původním modelu převzatého z [17]. Zmenšením počtu uzlů byly odstraněny některé detaily, jež vzhledem ke vzdálenosti zobrazeného modelu nebyly ani viditelné. Všechny tyto uzly mají kromě souřadnic přiřazený i bod textury, který zobrazují. Model byl vyexportovaný pomocí textového souboru, který je v aplikaci načten. Velikost tohoto souboru je 534 kB. Model není možné srovnávat s modely vytvořenými profesionály, ale pro účely této práce je dostatečný. Cílem této práce bylo ukázat způsob, jakým nahradit předtočená videa pohybem modelu, což se povedlo. Vytvořený model je zobrazen na Obr. 28.

Obr. 28 Výsledný model člověka

Pomocí přiřazených vah jednotlivým uzlům modelu bylo následně možné celý model rozpohybovat. Váhy jsou přiřazeny k jednotlivým kostem. Hlavní důraz byl kladen na správné pohyby rukou, mimika byla modelována velmi jednoduše, jako složený pohyb tří kostí. Z toho důvodu je mimika modelu celkem omezená. Váhy jsou uloženy pomocí textových souborů, což značně zmenšuje velikost celé aplikace. Textové soubory s normovanými váhami byly tři - váhy celého těla, váhy rotace zápěstí a váhy rotace předloktí způsobené rotací zápěstí. Textové soubory měly velikost 170 kB, 3 kB a 8 kB popořadě. V posledních dvou textových souborech byly uloženy pouze čísla uzlů a jejich váhy. První textový soubor je největší, protože obsahuje váhy pro celý model, z nichž největší část tvoří váhy zajišťující mimiku modelu.

Vyvinutá inverzní kinematika s číselným systémem kostí dobře popisuje veškeré pohyby, které má model předvést. Pro algoritmus počítání inverzní kinematiky pomocí kvaternionů nebyla použita žádná existující knihovna, aby byl dobrý vhled do samotného algoritmu. To samé platí pro interpolaci pomocí SLERP. Je možné, že použití již existujících knihoven by způsobilo zrychlení celého programu a výsledná aplikace by byla ještě menší. Pro jednoduchost byl ale zvolen přístup s vlastním algoritmem.

Pro účely vytvoření mobilní aplikace byl zvolen framework Kivy, pro jeho snadnou implementaci na skriptech napsaných v jazyce Python. Jedná se o jednoduchou aplikaci nabízející pouze dva režimy. Nejsou naprogramovány žádné prvky podporující uživatele v pokračování učení, pochvalování za více správných odpovědí po sobě atd. Aplikace

prošla i beta testováním na třech uživatelích, kteří s funkcionalitou aplikace nebyli předem nijak seznámeni. Velmi rychle se s aplikací seznámili a její používání jim nezpůsobovalo žádné problémy. Z důvodu malého počtu nahraných slov se tato slova opakovala s menší periodou, což způsobilo rychlejší naučení se jednotlivých znaků.

Dalším důvodem přístupu ke znakovému jazyku pomocí modelu člověka bylo navázání libovolného počtu znaků za sebe. Tyto posloupnosti mohou obsahovat jakákoli slova v jakémkoli pořadí. Tímto krokem by bylo možné aplikaci vyvinout i dále, zapojit do ní gramatiku znakového jazyka a začít tak tvořit jednotlivé věty. V samotné mobilní aplikaci není možné navazovat jednotlivé znaky za sebe, nicméně při spuštění programu *Signs_visualisation.py* je možné tvořit řetězce libovolných znaků. Textové soubory, pomocí nichž jsou znaky uloženy, jsou připraveny na to být poskládány za sebe a vytvořit tak sousloví nebo větu. Do aplikace nebyl tento prvek implementován proto, že gramatika znakového jazyka nebyla zkoumána ani nijak nahrávána do aplikace, tudíž by se mohlo stát, že by pohyb modelu neodpovídal správnému znakování.

5.1 Hodnocení aplikace

Výsledná mobilní aplikace byla ukázána dvěma slečnám používajícím znakový jazyk ve svém zaměstnání. Účelem bylo zjištění kvality předváděných znaků a možnosti aplikování aplikace ve výuce. Jejich komentáře k aplikaci budou v této sekci rozebrány.

První hodnocení aplikace je následující:

„Kdybych se z toho učila, tak mám občas problém třeba u toho postavení ruky nebo prstů. Pokud už člověk ví, co tam má být, tak ty znaky jdou celkem poznat.“

Toto hodnocení je komplexní bez konkrétních příkladů, kde je postavení ruky nebo prstů nepřesné. Model se hýbe podle předlohy z videa, tudíž je možné, že framework Mediapipe některé z klíčových bodů neurčil zcela přesně, což by zapříčinilo nepřesnosti pohybu. Na druhou stranu je možné textové soubory uchovávající osy a úhly jednotlivých znaků ručně upravit a docílit tím přesnějšího pohybu. V případě dalšího vývoje mobilní aplikace by bylo vhodné ke každému znaku zvlášť dostat konkrétní informace na zlepšení. I přes to ale byly všechny znaky rozeznány.

Hodnocení od druhé slečny je již rozsáhlejší a bude rozděleno na několik částí. I v tomto případě byly správně určeny všechny znaky.

„Důležité je si uvědomit, že mimika a orální komponenty (současné vyslovování slov během znakování) mají velký význam. Jakože to, co uděláš rukou a změníš u toho mimiku, může znamenat úplně jiné slovo. Výrazné vyslovování bych zakomponovala víc.“

V této práci byl kladen velký důraz na správný pohyb rukou. Důležitost mimiky však nelze rozporovat a u některých slov je spíše nevýrazná. Její namodelování by bylo možné udělat pomocí více kostí, čímž by se zvýšila kvality vyslovování. Jedná se ovšem o první verzi této aplikace, tudíž bylo důležité ukázat, jak by mohla fungovat.

„U některých znaků (babička, slyšící) není úplně poznat, že se tou rukou dotýká brady, ucha apod. Při čemž to umístění ruky v prostoru je taky hodně důležité. Napadlo mě, jestli by to třeba nešlo udělat i z boku pro lepší orientaci.“

Pohled ze strany by více napověděl o postavení ruky v prostoru a jejich dotycích určitých částí obličeje. Na druhou stranu pro některé znaky by nebylo vhodné mít pohled ze strany, protože by nebyl průkazný. Ideální by byl pohled mírně vyosený, ze kterého by bylo možné se lépe orientovat v jednotlivých dotycích.

„Pak jsem si ještě říkala, jestli by třeba nešlo, aby ten model měl v základní poloze ruce podél těla. V jednom případě mě zmátlo, že je dal před sebe a nevěděla jsem tak, jestli to je součástí znaku nebo to je jeho základní postavení.“

Při natáčení videí byl kladen velký důraz na to, aby se ruce tlumočnice vrátily do podobné polohy s pokrčenýma rukama mírně před tělem. Tato poloha byla vybrána jako výchozí a do ní se model vrací po každém předvedeném znaku. Výchozí poloha modelu opět možná změnit ručně v textových souborech, ale už by neodpovídala natočeným videím.

6 Závěr

Tato diplomová práce měla cíle vytyčené v kapitole Rozbor cílů a všechny byly splněny. Přístup k tvorbě mobilní aplikace byl zvolen prostřednictvím 3D modelu člověka předvádějícího pohyby místo předtočených videí. V rámci rešerše nebyla nalezena žádná aplikace používající podobný princip. Výsledná mobilní aplikace kombinuje výhody třech zmíněných aplikací na výuku nejen českého znakového jazyka, ale i amerického. Zmíněné aplikace jsou popsány v kapitole Dostupné aplikace.

Pro tvorbu modelu člověka byl použit software Blender. Byl použit již vytvořený model, jež byl zjednodušen téměř o 82 % všech uzlů a po přidání textur exportován ve formátu .obj, tedy ve formátu textového souboru. V něm jsou uloženy souřadnice jednotlivých uzlů společně s informacemi o textuře. Program v jazyce Python daný textový soubor načetl, čímž bylo docíleno ovládání 3D modelu mimo Blender. V kapitole Export modelu je rozepsáno, jakým způsobem byl textový souboru interpretován ve skriptu.

Do modelu byla vložena i skrytá kostra řídící pohyb celého modelu. Pomocí textových souborů bylo možné exportovat i váhy jednotlivých uzlů k daným kostem. V případě, že jeden uzel příslušel dvěma kostem, jsou jeho finální souřadnice váženým průměrem dílčích výsledků. Váhy pro rotaci zápěstí a předloktí musely být modelovány zvlášť, proto jsou uloženy v oddělených textových souborech. Tento princip společně se strukturou textových souborů jsou demonstrovány v kapitole Ukládání slov.

Inverzní kinematika modelu byla implementována pomocí kvaternionů společně s interpolací pomocí SLERP. Program vyvinutý v této diplomové práci mění souřadnice jednotlivých uzlů, ke kterým jsou přiřazeny váhy. Kinematika byla navržena tak, aby odpovídala pohybům lidského těla. Byl zaveden systém nadřazených kostí ovlivňující veškeré podřízené kosti. Celý uvedený program je napsán v jazyce Python.

Mobilní aplikace pojmenovaná ZnaKouč byla vytvořena pomocí frameworku Kivy a Buildozer. S jeho pomocí byl skript napsaný v jazyce Python převeden na soubor kompatibilní s operačním systémem Android. V aplikaci je prostřednictvím textových souborů uloženo 20 slov. V těchto textových souborech jsou uloženy pouze úhly a osy, kolem kterých mají jednotlivé kosti rotovat. Popis výsledné aplikace je zachycen v kapitole Popis aplikace. Aplikace byla dána na testování korektnosti předváděných znaků dvěma slečnám používajícím znakový jazyk v zaměstnání. Ty měly výhrady, ale pozitivní zprávou je, že poznaly všechny znaky. Výhrady se týkaly výchozí pozice modelu, postavení ruky a prstů nebo možného pohledu z boku, jež by lépe nastínil, zda se model dotýká brady nebo ucha.

Jedná se o první aplikaci používající model člověka místo předtočených videí. Vzhledem k prvotní variantě takové aplikace byl zvolen nejprve základní model člověka. Pro budoucí použití lze upravit jeho grafickou podobu pro reálnější vzhled. Mnou vytvořená varianta obsahuje možnosti procvičování a slovníku. V dalších verzích by mohl být zaveden systém odměn, bonusových bodů a dalších motivačních nástrojů. Jedná se o výchozí verzi aplikace, tudíž bylo cílem ukázat cestu dalšího vývoje. Také z toho důvodu bylo nahráno pouze 20 slov, jejichž rozšíření nezabere mnoho úsilí. Přidání dalších slov vyžaduje pouze správné natočení tlumočnice a spuštění programu *videos2txt.py* s těmito nahranými videi. Program sám vygeneruje textový soubor, který je možné dále použít v aplikaci. Slova by následně mohla být i rozřazena do skupin a tématických celků.

Použitá literatura

[1] Models for human body modeling. In: *Arxiv* [online]. [cit. 2023-04-19]. Dostupné z: https://arxiv.org/pdf/2012.13392.pdf

[2] CAO, Zhe. *OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields* [online]. 2019 [cit. 2023-04-19]. Dostupné z: https://arxiv.org/pdf/1812.08008.pdf

[3] OpenPose skeleton with keypoints labeled. In: *Reddit* [online]. [cit. 2023-04-19]. Dostupné z: https://www.reddit.com/r/StableDiffusion/comments/11cuztu/openpose_controlnet_preprocessor_options/

[4] Openpose. *Github* [online]. 24. 10. 2022 [cit. 2023-04-19]. Dostupné z: https://github.com/CMU-Perceptual-Computing-Lab/openpose

[5] Next-Generation Pose Detection with MoveNet and TensorFlow.js. In: *TensorFlow Blog* [online]. 17. 5. 2021 [cit. 2023-04-19]. Dostupné z: https://blog.tensorflow.org/2021/05/next-generation-pose-detection-with-movenet-and-tensorflowjs.html?fbclid=IwAR21c_NMKUpznPTP07FMtVvN8X3a0sNhB_mfCiL8MFaDZ8TtfLl8Kxp4fQI

[6] Keypoints: Used in MoveNet. In: *Github* [online]. 2021 [cit. 2023-04-19]. Dostupné z: https://github.com/tensorflow/tfjs-models/tree/master/pose-detection

[7] MediaPipe [online]. [cit. 2021-12-15]. Dostupné z: https://google.github.io/mediapipe/

[8] Body estimation using 33 landmarks. In: *Arxiv* [online]. 13. 10. 2021 [cit. 2023-04-19]. Dostupné z: https://arxiv.org/pdf/2110.06877.pdf

[9] *Neural Sign Language Synthesis: Words Are Our Glosses*. Plzeň, 2020. University of West Bohemia, Faculty of Applied Sciences.

[10] CHILLINGWORTH, Alec. The Pros & Cons of Creating 3D Content With Blender Software. In: *Epidemic Sound Blog* [online]. 30. 3. 2023 [cit. 2023-05-18]. Dostupné z: https://www.epidemicsound.com/blog/blender-software/#is-blender-completely-free

[11] REICHL, Jaroslav. Eulerovy úhly. In: *Encyklopedie fyziky* [online]. 2006 [cit. 2023-05-19]. Dostupné z: http://fyzika.jreichl.com/main.article/view/108-eulerovy-uhly

[12] Quaternion multiplication table. In: *Wikipedia* [online]. [cit. 2023-05-18]. Dostupné z: https://en.wikipedia.org/wiki/Quaternion

[13] Spherical Linear Interpolation (Slerp). In: *Read The Docs* [online]. 6. 12. 2022 [cit. 2023-05-18]. Dostupné z: https://splines.readthedocs.io/en/latest/rotation/slerp.html

[14] Lekce 1 - Představení Kivy frameworku a tvorba prvních aplikací. In: *Itnetwork.cz* [online]. [cit. 2023-05-18]. Dostupné z: Lekce 1 - Představení Kivy frameworku a tvorba prvních aplikací Zdroj: https://www.itnetwork.cz/python/kivy-framework-pro-python/predstaveni-kivy-frameworku-a-tvorba-prvnich-aplikaci

[15] 3D Rotating Monkey Head. In: *Kivy: The Open Source Python App Development Framework.* [online]. [cit. 2023-05-24]. Dostupné z: https://kivy.org/doc/stable-1.11.1/examples/gen__3Drendering__main__py.html

[16] ŽÁK, Jindřich. *INTEGRACE A NASAZENÍ MOBILNÍ APLIKACE PRO STUDENTY ZNAKOVÉHO JAZYKA V ORGANIZACI TICHÝ SVĚT*. Praha, 2021. Diplomová práce. ČVUT. Vedoucí práce Ing. Michal Valenta, Ph.D.

[17] Detailed Man. In: *TURBOSQUID* [online]. 9. 7. 2010 [cit. 2023-05-18]. Dostupné z: https://www.turbosquid.com/3d-models/free-obj-mode-human/544305?fbclid=IwAR2FW4XqNiIvtf0zczKRojF-ZDfPGRkt04h2qa_h751thRV5jk1V_BCV_TI

[18] P., Thomas. Creating face/head textures for human(oid) game characters. In: *Free3dtutorials* [online]. [cit. 2023-05-18]. Dostupné z: http://www.free3dtutorials.com/creating-facehead-textures-for-humanoid-game-characters.php/2

[19] Jack Sparrow Arms texture. In: *DeviantArt* [online]. 18. 5. 2021 [cit. 2023-05-18]. Dostupné z: https://www.deviantart.com/dazinbane/art/Jack-Sparrow-Arms-texture-879883932

Seznam použitých zkratek a symbolů

Symbol	Název	Jednotka
ASL	americký znakový jazyk	-
$\vec{a}, \vec{b}$	vektory	-
$\alpha, \beta, \nu, \psi, \varphi$	označení úhlů	rad
BSL	britský znakový jazyk	-
c	koeficient vzdáleností	-
ČSL	český znakový jazyk	-
d_b, d_m	vzdálenosti	pixel
i, j, k, x, y, z	souřadné osy	-
NDK	Native Developer Kit	-
n	osa rotace	-
$p_1, p_2, p_3, p_4, r, r_1, r_2, r_3, r_4, q, q_1, q_2, q_3, q_4, \bar{q}, \overline{q_1}$	kvaterniony	-
P, P_1, P_2	body interpolace	-
R_x, R_y, R_z	rotační matice	-
s	reálná část kvaternionu	-
SDK	Software Developer Kit	-
t	parametr interpolace	-
$u, \bar{u}$	rotovaný bod	-
v, v_1, v_2, v_3	imaginární část kvaternionu	-
ω	úhlová rychlost	rad/s

Seznam obrázků

Seznam tabulek

Seznam příloh

1) Human.txt
2) texture.jpg
3) get_weights.py
4) weights.txt
5) weights_forearm.txt
6) weights_wrist.txt
7) videos2txt.py
8) Signs_visualisation.py
9) Slova
10) buildozer.spec
11) main.py