Univerzita Komenského, Bratislava

Fakulta Matematiky, Fyziky a Informatiky

Použitie LabanXML v kombinácii s Blenderom

Bakalárska práca

Ján Ruhalovský, 2012

Univerzita Komenského, Bratislava

Fakulta Matematiky, Fyziky a Informatiky

Použitie LabanXML v kombinácii s Blenderom

Bakalárska práca

Študijný program: Informatika
Študijný odbor: 2508 Informatika
Školiace pracovisko: Katedra informatiky
Vedúci práce: RNDr. Jana Dadová

Bratislava, 2012

Ján Ruhalovský

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Ján Ruhalovský
Študijný program: informatika (Jednoodborové štúdium, bakalársky I. st., denná forma)
Študijný odbor: 9.2.1. informatika
Typ záverečnej práce: bakalárska
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Použitie LabanXML v kombinácii s Blenderom

Cieľ:
Naštudovanie problematiky zapisu a analyzy pohybu
Naštudovanie zápisu pohybu do LabanXML
Navrhnutie možnosti použitia LabanXML v spojení s Blenderom
Implementácia navrhnutého riešenia
Použitie implementácie na skúšobných dátach

Vedúci: RNDr. Jana Dadová
Katedra: FMFI.KAGDM - Katedra algebry, geometrie a didaktiky matematiky

Dátum zadania: 04.10.2011

Dátum schválenia: 25.10.2011

doc. RNDr. Daniel Olejár, PhD.
garant študijného programu

.....
študent

.....
vedúci práce

Čestne prehlasujem, že som túto prácu vypracoval sám za pomoci uvedenej literatúry a konzultácií s vedúcim práce.

. . . . . .

Poďakovanie

Ďakujem vedúcej mojej bakalárskej práce, RNDr. Jane Dadovej, za vedenie tejto práce, cenné rady pri konzultáciách, ústretovosť a za zadanie témy. Moja vďaka patrí aj rodine, predovšetkým za trpezlivosť a podporu pri písaní.

Abstract

The aim of this Bachelor's thesis is designing and implementation of application movements from the xml document on humanoid model and also creation animations for this document. Moreover the thesis tries to analyse the reverse process. The labanotation is used for recording the movement and LabanXML is record of labanotation into xml structure. In summary, the thesis firstly exposes the problem of loading movements from LabanXML file and subsequent evaluation. Secondary, it solves the problem of the applying movement from xml database to model. The opposite process, process of creating LabanXML record from animation, is also noticed in this thesis. In addition, we offer a graphical user interfaces to work with these facts. The solution is implemented in programming language Python, especially in application programming interface of modeling tool Blender.

KEYWORDS: LabanXML, animation, movement, labanotation, Blender, Python

Abstrakt

Cieľom tejto bakalárskej práce je návrh a implementácia riešenia pre aplikovanie pohybov zapísaných v xml dokumente na humanoidný model postavy s následným vytvorením animácie, a zároveň pokus o opačný proces. Pre zápis pohybu nám slúži Labanova notácia, konkrétnejšie jej zápis do xml súboru nazývaného LabanXML. V práci riešime problém načítavania pohybov z LabanXML súboru s následným vyhodnotením a aplikovaním pohybov z xml databáz na model postavy. Opačný proces, vytvorenie LabanXML zápisu z animácie, je tiež v tejto práci načrtnutý. Zároveň ponúkame používateľovi grafické rozhranie na prácu s týmito skutočnosťami. Naše riešenie je implementované v programovacom jazyku Python, konkrétnejšie v aplikačnom programovacom rozhraní modelovacieho nástroja Blender.

KĽÚČOVÉ SLOVÁ: LabanXML, animácia, pohyb, Labanova notácia, Blender, Python

Zoznam obrázkov

Obsah

Úvod

Pohyb je s človekom a prírodou spätý od jeho samotného vzniku. Je súčasťou života, ba dokonca nutnosťou preň, a samozrejme aj pre existenciu celého vesmíru. Pohyb má mnoho podôb a je jasne badateľný predovšetkým u živých tvorov. Najzaujímavejším a najviac ceneným je pohyb človeka. Pomocou pohybu može vyjadrovať svoje pocity, náladu a psychické rozpoloženie, v ktorom sa práve nachádza.

Mnohých ľudí napadne pri pohybe zaujímavá otázka, ako by daný pohyb vyznel, ak by bol vykonávaný viacerými osobami súčasne. Týmto problémom sa v dnešnej dobe zaoberajú hlavne choreografi a členovia tanečných skupín. Aby to mali pri svojej profesii jednoduchšie, bolo by pre nich výhodné mať akýsi nástroj na vizualizáciu a zápis pohybov pri tanečných kreáciách, ktorý by im uľahčil predstavenie si súladu a synchronizovanosti medzi jednotlivými tanečníkmi. Náčrt takéhoto nástroja a spôsob riešenia jeho fungovania a používania sa pokúsime načrtnúť a priblížiť aj v tejto práci.

Cieľom tejto bakalárskej práce je použiť štruktúru LabanXML dokumentu pre zápis Labanovej notácie, jednej z možností zápisu pohybu, a následne sa pokúsiť vytvoriť animovaný pohyb postavy, tzn. z formátu xml aplikovať jednotlivé pohyby na vopred vymodelovanú postavu v programe Blender, aplikácii pre 3D grafiku, pomocou pythonovského rozhrania, ktoré tento grafický program poskytuje. Zároveň úlohou tejto práce je pokúsiť sa aj o zápis pohybu do xml formátu z už vopred naanimovaných pohybov modelu postavy.

V 1. kapitole s názvom Prehľad problematiky sa budeme venovať vysvetleniu jednotlivých základných pojmov a použitých nástrojov súvisiacich s touto témou. V 2. kapitole, ktorej názov je Motivácia a súvisiace riešenia, uvedieme aká je motivácia práce a spravíme prehľad známych riešení zaoberajúcich sa danou problematikou. 3. kapitola s pomenovaním Návrh riešenia nám opisuje problém a takisto návrh pre jeho riešenie. Témou 4. kapitoly s názvom Riešenie je opis podrobností o jeho implementácii a postupoch pri tvorbe riešenia. V poslednej 5. kapitole s pomenovaním Výsledky je zhrnutie dosiahnutých výsledkov tejto práce.

1

Prehľad problematiky

V úvodnej kapitole tejto práce sa budeme venovať vysvetleniu základných pojmov používaných v tejto práci, povieme si o základných princípoch Labanovej notácie, notácie pre zápis pohybu. Ďalej sa oboznámime so zápisom Labanovej notácie do xml súboru, čiže LabanXML.

1.1 Základné pojmy

Kvôli jednoznačnosti najprv uvedieme niekoľko pojmov spolu s ich krátkym opisom a vysvetlením ich významu, ako sú chápané v tejto práci. V tejto kapitole sa budeme venovať aj použitým aplikáciám a použitým formátom.

Kosť - základný stavebný prvok kostry. Je určená pozíciou svojho začiatku, dĺžkou a rotáciou jej lokálnych osí voči osi globálnej [Par07].

Kostra - armatúra, je to zoskupenie kostí do stromovej štruktúry, pričom jednotlivé kosti sú vrcholmi daného stromu. Spojenie kostí je vytvárané medzi koncovými bodmi dvoch kostí, ktoré sa nazývajú kĺby. Tak ako v klasickej stromovej štruktúre má každá kosť svojho predchodcu, čo neplatí pre koreňovú kosť. Kostra môže byť tvorená z viacerých komponentov [Par07].

XML - eXtensible Markup Language (angl.), prekladom rozšíriteľný značkovací jazyk. Bol vyvinutý skupinou XML Working Group formovanou pod záštitou W3C konzorcia v roku 1996. Hlavným cieľom XML je oddelenie obsahu dát od ich vzhľadu, a preto je XML dokument súborom, ktorý obsahuje dáta a nezaoberá sa ich samotnou reprezentáciou. Tento jazyk je určený hlavne na zápis údajov a na ich samotný prenos medzi jednotlivými aplikáciami. Syntax XML dokumentu je podobná, ale prísnejšia než pri HTML dokumentoch. Špecifickou vlastnosťou pre XML je, že nemá preddefinované tagy [BPSM+08].

Blender - je to free open source 3D softvér, dostupný pre viacero operačných systémov, šíriteľný pod GNU GPL licenciou (GNU General Public License). V čase písania tejto kapitoly bola poslednou verziu tohto softvéru verzia 2.63a.
Tento free open source softvér je pre naše účely mimoriadne vhodný, keďže je silným nástrojom pre 3D vizualizáciu, má vyhovujúce pracovné prostredie, a je zároveň porovnateľný aj s inými komerčne dostupnými aplikáciami podobného charakteru. Významnou vlastnosťou tejto aplikácie je aj jej podpora používateľských skriptov programovacieho jazyka Python, ktoré budeme v tejto práci využívať [bleb].

Python - je výkonný dynamický programovací jazyk, podporovaný na viacerých platformách. Pôvodne bol vytvorený ako skriptovací jazyk. Je častokrát porovnávaný s jazykmi ako sú Perl, Ruby alebo Java.
Jeho výhodou je jasne čitateľná syntax a aj jeho jednoduchá rozšíriteľnosť. Moduly (knižnice), ktoré môžu byť napísané aj v inom jazyku (napr. C alebo C++), rozširujú Python o ďalšiu funkcionalitu, sprístupňujúcu aplikačné programovacie rozhrania (API) rôznych iných programov, v ktorých funguje ako skriptovací jazyk. Práve Blender patrí medzi takéto programy [pyt].

Blender Python API - aplikačné programovacie rozhranie programu Blender. Toto rozhranie obsahuje moduly, ktoré nám umožňujú zasahovať a manipulovať s časťami interných dát a funkcií programu. Pomocou tohto rozhania a jeho modulov je možné vytvárať skripty, pomocou ktorých môžeme napríklad vytvárať objekty, spúšťať iné

skripty, vykresľovať grafické užívateľske rozhranie (GUI) a pridávať mnohú ďalšiu funkcionalitu [blea].

Labanova notácia - labanotation (angl.), je spôsob zápisu rôznych (predovšetkým tanečných) pohybov osôb. Je známa aj pod názvom Labanova kynetografia. Pomenovaná je podľa bratislavského rodáka a tanečníka Rudolfa Labana (1879 – 1958), ktorý je zároveň jej autorom. Spôsob a systém zápisu bude opísaný v ďalšej sekcii [HG05].

LabanXML - xml súbor, v ktorom je Labanova notácia prepísaná do xml štruktúry. Podrobnejšie sa mu budeme venovať v samostatnej časti.

Gestá - pod týmto pojmom budeme rozumieť všetky pohyby, ktoré sú vykonávané bez prenesenia váhy na pohybujúcu sa časť tela.

Segmentová (artikulovaná) štruktúra - štruktúra, ktorá sa skladá z konečnej postupnosti pevných častí, prepojených kĺbami. V kĺboch možno s jednotlivými časťami otáčať.
Artikulovaná štruktúra je ukotvená na jednom konci, no druhý má voľný. Voľný koniec tejto štruktúry voláme koncový efektor. Príkladom takejto segmentovej štruktúry je napríklad dolá končatina, kde efektormi sú prsty [WW91].

Dopredná kinematika - reprezentuje algoritmicky jednoduché manipulovanie so štruktúrou skladajúcou sa z pevných častí spojených kĺbami (stromová štruktúra), kde sa pozícia častí vyhodnocuje od koreňovej častice k listovej častici [Par07].

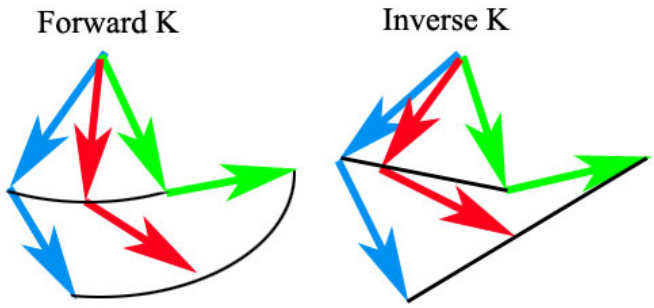
Inverzná kinematika - je zložitejšie manipulovanie so štruktúrou skladajúcou sa z pevných častí spojených kĺbami, kde sa pozícia jednotlivých častí vyhodnocuje vzhľadom na časť, ktorá mení svoju pozíciu zväčša určenú používateľom [Par07].

Interpolácia - proces dopočítania hodnôt medzi dvoma vstupnými hodnotami. V našom prípade budú vstupnými hodnotami kľúčové snímky (keyframes).

Kľúčové snímky (keyframes) - snímky animácie, v ktorých sú zadané kľúčové parametre animovaného objektu. Pre ostatné snímky medzi kľúčovými snímkami sa parametre modelu dopočítajú interpoláciou [SK96].

1.2 Dopredná a inverzná kinematika

V tejto časti bližšie ozrejmime pojmy dopredná kinematika (forward kinematics) a inverzná kinematika (inverse kinematics).



Obr. 1.1: Dopredná a inverzná kinematika [pic]

Obe tieto kinematiky sa zaoberajú priestorovou polohou efektora v segmentovej (artikulovanej) štruktúre a natočením ostatných segmentov.
Dopredná kinematika sa zaoberá výpočtom polohy efektora segmentovej štruktúry, kde jedinou neznámou je samotná poloha tohto bodu (efektora), lebo natočenia ostatných segmentov sú známe. Poznáme ich vďaka tomu, že pri doprednej kinematike sa otočenie jednotlivých častí vyhodnocuje od koreňového segmentu až k efektoru.
Narozdiel od doprednej kinematiky je úlohou **inverznej kinematike** vypočítať parametre natočení jednotlivých segmentov segmentovej štruktúry a to na základe polohy efektora. Riešenie pre tento prípad nie je jednoznačné, ba dokonca sa môže vyskytovať nekonečne množstvo riešení [MBBI05].

1.3 Labanova notácia

V tejto časti sa budeme venovať zápisu pohybu do Labanovej notácie, ozrejmime si jednotlivé symboly tohto zápisu a povieme ako a na aké účely sa Labanova notácia používa.

Labanova notácia (Labanotation, Kinetography Laban) je systém zaznamenávania ľudského pohybu. Autorom tejto notácie je Rudolf von Laban známy tiež ako Rudolf Laban (1879 – 1958), ktorý je dôležitou postavou európskeho moderného tanca. Túto notáciu publikoval prvýkrát v roku 1928. Nevyužíva sa len na zápis pohybov pre tanečníkov, ale aj v oblasti antropológie, atletiky a fyzioterapeutike.

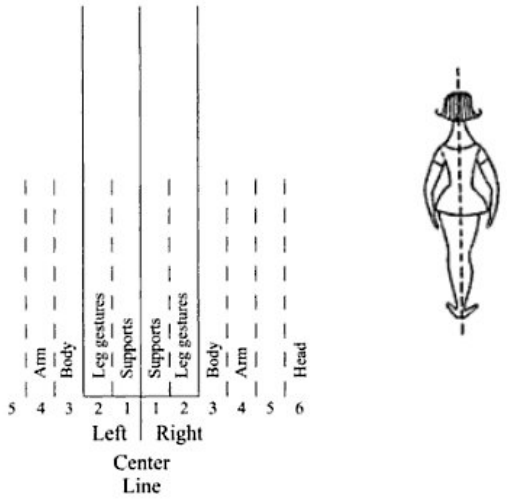
Labanova notácia plní pre tanec rovnakú funkciu ako notový zápis pre hudbu. Narozdiel od notového zápisu, ktorého dĺžka závisí od počtu nôt, dĺžka Labanovej notácie nezávisí od počtu značiek potrebných na zápis pohybu [HG05].

Labanov zápis jasne určuje začiatok a koniec pohybu, a finálnu polohu častí tiel. Avšak expliciitne nešpecifikuje pózy medzi začiatkom a koncom daného pohybu, takže spôsob vykonania pohybu je často ponechávaný na dedukcii samotného aktéra pohybu [CWRF05].

1.3.1 Princíp Labanovej notácie

Labanova notácia používa osnovu (staff) s tromi vertikálnymi čiarami. Tieto čiary reprezentujú telo človeka (stredná čiara korešponduje s osou ľudského tela, pravá časť osnovy s pravou a ľavá časť osnovy s ľavou stranou človeka).

Pohyby trupu a jeho častí, hlavy a ramien sú zapisované za vonkajšou čiarou osnovy. Vo vnútri osnovy sú 4 hlavné vertikálne stĺpce (2 vľavo od stredovej čiary a 2 vpravo). Obrázok 1.2 nám ilustruje, ktorá časť tela je reprezentovaná daným stĺpcom. V skutočnosti však môžeme mať týchto stĺpcov viac na oboch stranách. Ak by sme chceli zaznamenať pohyb nejakej konkrétnej časti tela, môžeme si pridať stĺpec, ale mali by



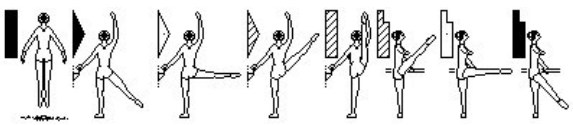
Obr. 1.2: Labanova osnova a ľudské telo [HG05]

sme ho dostatočne popísať, aby bolo zrejmé, ktorej časti sa týka [HG05].

V tejto práci sa budeme zaoberať klasickou osnovou pre Labanovu notáciu, to znamená, že nebudeme uvažovať o pridaných stĺpcoch pre špeciálne časti tela postavy človeka.

1. stĺpec: Podpora (Support)
Tento stĺpec má špeciálnu funkciu, zaznamenáva sa doň, kde je prenesená váha človeka. Teda tento sĺpec nám hovorí, ktorá časť tela nesie váhu človeka, alebo nie je nesená žiadnou časťou tela (vtedy ide o skok človeka). Symboly vpísané do tohto stĺpca nám určujú postup celého tela, to znamená, ktorým smerom je prenášané ťažisko človeka.
Všetky gestá ostatných častí tela sú zaznamenané v ďalších stĺpcoch.

2. stĺpec: Gestá nôh (Leg Gestures)

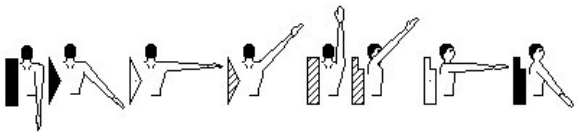


Obr. 1.3: Gestá nôh [laba]

V tomto stĺpci sa zaznamenávajú gestá nôh. Napríklad ak je noha vystretá dopredu vo vzduchu a presunieme ju dozadu, tak tento pohyb bude zaznačený v tomto stĺpci. Tento stĺpec sa používa pre individuálne časti nohy - stehno, predkolenie a chodidlo - tie sú zaznačené špeciálnymi značkami pre jednotlivé časti. Pre zápis sa používa značenie ako je na obrázku 1.3 (podrobnejšie vysvetlenie značenia uvedieme v ďalšej sekcii).

3. stĺpec: Telo (Body)
Stĺpec bezprostredne za oblasťou 'Gestá nôh' je používaný pre telo, trup a jeho časti. Takže hruď, panva, pás, teda horná polovica tela je zaznamenávaná v tejto časti danými značkami.

4. stĺpec: Ruka (Arm)



Obr. 1.4: Gestá rúk [laba]

Obrázok 1.4 nám znázorňuje hlavné gestá rúk tohto stĺpca. Nachádza sa za stĺpcom 'Body' na ľavej aj pravej strane Labanovej osnovy. Tento stĺpec popisuje gestá celých rúk. Môžeme si všimnúť, že pre ne poznáme podobné značenie ako pre značenie pohybu nôh.

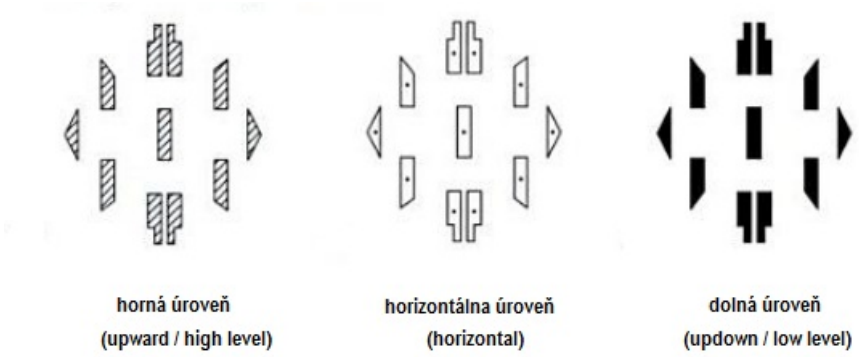
5. stĺpec: Hlava (Head)
Posledný stĺpec v klasickej Labanovej osnove je stĺpec pre hlavu. Tento stĺpec sa nachádza iba na pravej strane Labanovej osnovy. Značenie pre pozíciu a smer pohybu hlavy je podobná ako pri pozíciach a smere pohybu nôh a rúk.

1.3.2 Základné symboly

Smery indikovaných pohybov, sú zaznačené symbolmi smeru (direction symbols).

Obr. 1.5: Základné symboly smeru [laba]

Na obrázku 1.5 sú znázornené základné symboly umožňujúce zaznamenať zmenu smeru pohybu o 45°, ale použitím zložitejších kombinácií symbolov, je možné prerozdeliť túto mieru viac presnejšie.

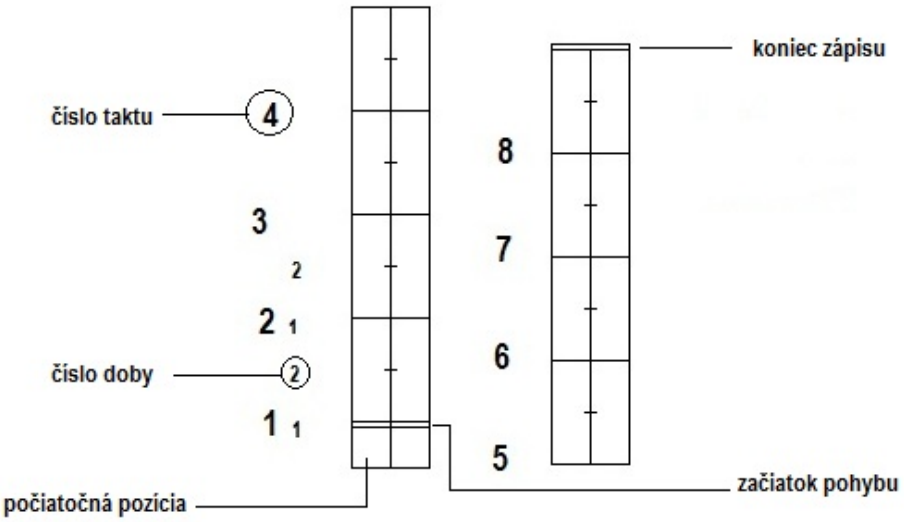


Obr. 1.6: Tri úrovne pohybov [HG05]

Symboly smeru môžu mať rôzne tieňovanie, ktoré nám vraví, aká je úroveň daného pohybu - horná úroveň (upward/high level), dolná úroveň (updown/low level), horizontálna úroveň (horizontal/middle). Tieto úrovňe vidíme na obrázku 1.6.

1.3.3 Časovanie

Narozdiel od hudobného zápisu, Labanova notácia nemá špeciálne značky pre dĺžku trvania určitého pohybu. V Labanovom zápise sa vyjadruje časová dĺžka pohybu dĺžkou samotného symbolu, ktorý sa na zápis použil. Labanovu osnovu môžeme podobne ako aj notovú osnovu rozdeliť na menšie časti - takty (bar). Takty môžeme opäť rozdeliť na ešte menšie časti - doby (beat/pulse), ktoré nám budú reprezentovať časové úseky. Podobne ako v notovom zápise môže mať jeden takt viacero dôb. Podľa počtu časových úsekov, ktoré symbol pohybu prekrýva, vieme určiť, ako dlho sa má daný pohyb vykonávať.



Obr. 1.7: Časová štruktúra Labnovej notácie

Podobne ako aj pri notovom zápise, je potrebné určiť, kde sa začína pohyb vykonávať a aká predpríprava je nutná pred samotným zahájením. Ak nie je určené inak, tak sa pohyb začína vykonávať za prvou dvojitou čiarou, ktorá pretína tri vertikálne čiary Labanovej notácie. Pred touto dvojitou čiarou je zaznamenaná počiatočná póza, z ktorej sa začína. Záver celého pohybu je zaznačený tiež dvojitou čiarou na konci Labanovej osnovy. Časovú štruktúru vidiíme na obrázku 1.7.

Ak je sekvencia pohybov príliš dlhá, môže sa zápis rozdeliť do viacerých osnov ukončených jednoduchou čiarou [laba].

1.4 LabanXML

V tejto časti sa budemem venovať xml reprezentácii Labanovej notácie, ktorú zostavila dvojica japoncov Minako Nakamura a Kozaburo Hachimura v roku 2006.

Vznik LabanXML bol podnietený skôr zostaveným xml zápisom pre hudbu (MusicXML). LabanXML zápis je vlastne prepísanie Labanovej notácie do štruktúry xml dokumentu. Stručne popíšeme niektoré elementy xml štruktúry a ich význam.

Koreňovým (root) elementom LabanXML dokumentu je element *laban*, ktorý v sebe zahŕňa elementy *attribute* a *notation*.
Element *attribute* obsahuje element *time*, ktorého potomkami sú elementy *beat* (počet dôb v jednom takte) a *beat-type* (typ jednej doby).
Najpodstatnejšia časť je zapísaná v elemente *notation*, ktorá obsahuje *repeat* (opakované časti zápisu) a *measure* (samotný takt). Element *measure* obsahuje elementy, ktoré rozdeľujú strany tela, na ktorých sa pohyb vykonáva a to *left*, *support* a *right*. V týchto častiach sú už elementy, ktoré popisujú samotné časti tela ako ruky, nohy, hlava, telo a podobne.
Každý z jednotlivých elementov má vlastné atribúty, ktoré bližšie určujú čas vykonávania, trvanie a úroveň pohybu, a zároveň aj smer [HN06].

Krátky príklad LabanXML zápisu:

```
<laban>
  <attribute>
    <time>
      <beat>4</beat>
      <beat-type>4</beat-type>
    </time>
  </attribute>

  <notation>
    <measure num="0">
      <left>
        <arm duration="1" ticknum="1">
          <direction>Right</direction>
          <level>M</level>
```

```
          </arm>
          <leg duration="1" ticknum="1">
            <direction>Left</direction>
            <level>M</level>
          </leg>
        </left>
        <support side="left" duration="1">
          <direction>Place</direction>
          <level>M</level>
        </support>
        <support side="right" duration="1">
          <direction>Place</direction>
          <level>M</level>
        </support>
        <right>
          <arm duration="1" ticknum="1">
            <direction>Forward</direction>
            <level>L</level>
          </arm>
        </right>
      </measure>
  </laban>
```

2

Motivácia a súviciace riešenia

V tejto časti sa budeme venovať vysvetleniu toho, prečo je potrebné, respektíve vhodné riešiť a zaoberať sa problémom aplikovania pohybu z dát uložených v istej forme na model postavy a takisto aj možnosťou zápisu pohybu s jeho následným uložením do vhodnej štruktúry. Uvedieme aj niekoľo oblastí, ktorých sa daná problematika týka, ako aj spôsoby využitia rovnakých alebo podobných prístupov.

2.1 Motivácia

Dôvodov, prečo zapisovať pohyb človeka vo forme dát a zaoberať sa ich aplikovaním na model postavy, je hneď niekoľko. My sa ich pokúsime zhruba načrtnúť.

Je všeobecne známe, že medzi dvoma aplikáciami môže dochádzať k výmene dát. Avšak informácia, ktorú tieto dáta obsahujú, môže byť rôzna v závislosti od samotných aplikácii, predovšetkým od toho, na akú prácu sú používané. Jednou z informácií, ktorú sú schopné si vymeniť, by mohla byť správa o pohybe ľudského tela. Ale na to, aby sme dokázali túto správu poslať, potrebujeme formát správy, ktorému by rozumeli oba tieto aplikácie a dokázali ju spracovať. Riešením môže byť zápis pohybu do xml štruktúry. Pre náš prípad, kedy chceme vedieť zapísať informáciu o pohybe, by bolo vhodné použiť LabanXML, ktorý je vlastne prepisom Labanovej notácie, notácie

pre pohyb, do štruktúry xml. S týmto formátom by potom mohli pracovať aplikácie, ktoré zobrazujú pohyby postáv, zapisujú a prenášajú ich. Užitočné môžu byť najmä pre tanečníkov a choreografov, ktorí by si mohli daný pohyb vizualizovať.

Ak máme nástroj, ktorý nám dokáže aplikovať daný pohyb na model, núti nás to dať si ďalšiu otázku, aké možnosti nám daný nástroj ponúka. Veľmi vhodnou vlastnosťou, by bola možnosť aplikovať daný pohyb na viacero postáv (modelov) bez toho, aby sme museli každý model animovať samostatne. Praktické využitie je badateľné najmä pre tanečné skupiny a ich choreografie, alebo pri animovaní pohybov ľudí v dave a pohybe samotného davu ako celku.
Vhodnou vlastnosťou, ktorú tento prístup ponúka, je aj možnosť priestorového rozvrhnutia pohybov skupiny ľudí pri vopreed známom modeli priestoru, v ktorom sa má daný pohyb vykonávať.
Ďalšou, o čosi náročnejšou požiadavkou je, aby sme z vopred naanimovaného modelu postavy, vedeli pohyb zapísať do dátovej štruktúry a dokázali ho poskytnúť ďalej na ďalšie spracovanie iným aplikáciám.

V tejto práci sa budeme zaoberať len možnostiam aplikovania pohybu na model postavy z dátovej štrutúry, a zápisu pohybu do rovnakej štruktúry dát s následným porovnaním.

2.2 Súvisiace riešenia

V tejto podkapitole uvedieme niekoľko projektov, ktoré sa zaoberajú problematikou Labanovej notácie, zápisu tejto notácie a grafickou vizualizáciou pohybu.

LabanWriter je voľne dostupný softvér pre platformu Macintosh vyvinutý na Ohio State University v roku 1987. Je to vlastne editor na zápis Labanovej notácie, ktorý obsahuje vyše 700 symbolov danej notácie. Tento editor ponúka možnosť kopírovať, upravovať a ukladať pohyby, pričom sú uložené v špeciálnom type súboru, ktorý je vlastne postupnosťou ASCII znakov usporiadaných podľa istých pravidiel. Pri zobra-

zovaní pohybov v tejto aplikácii vidíme na obrazovke zápis Labanovej notácie tak, ako by sme ho písali na papier priamo do Labanovej osnovy [labc].

LabanReader je voľne dostupný softvér pre platformu Macintosh, vyvinutý na Ohio State University v roku 1999, určený na zobrazovanie zápisu pohybu zaznamenaného v Labanovej notácii. Tento softvér pracuje s typom súboru vytvoreným nástrojom LabanWriter [labb].

LabanEditor je interaktívna grafická aplikácia programovaná v jazyku Java, ktorú vytvorila japonská trojica Kazuya Kojima, Kozaburo Hachimura, Minako Nakamura v roku 2002. Editor má 5 hlavných funkcií pre prácu s Labanovou notáciou: vkladať, upravovať, tlačiť, zobrazovať a exportovať dáta. Narozdiel od predošlých dvoch nástrojov, ktoré manipulovali len so zápisom Labanovej notácie, LabanEditor má obsiahnutú funkciu exportovania, ktorá konvertuje výsledok notácie do VRML2.0 formátu, ktorý môže byť čítaný a zobrazovaný vo webových prehliadačoch. LabanEditor vniesol lepšiu interakciu medzi používateľa a samotnú aplikáciu, keďže samotný používateľ je schopný pozrieť si výslednú animáciu pohybu po zadaní symbolov vložených pomocou používateľského rozhrania. Je to v podstate akoby LabanWriter doplnený o 3D znázornenie pohybov. Interná reprezentácia notácie v tomto nástroji je uložená v špeciálnom LND súbore. Po zadaní symbolov v grafickom rozhraní používateľom a následnom vyhodnotení vstupu sa z tohto súboru načítajú potrebné údaje, kotré sú potom konvertované na údaje potrebné k zobrazeniu 3D animovaného pohybu modelu postavy. Pri animovaní pohybu sú využívané kľúčové snímky (keyframes), medzi ktorými sú hodnoty zvyšných častí pohybu generované lineárnou interpoláciou [KHN02].

V tejto práci budeme podobne ako v LabanEditore používať lineárnu interpoláciu pre generovanie pohybov medzi kľúčovými snímkami a taktiež funkciu LND súboru bude v našej práci spĺňať xml súbor (xml databáza). Všetky tieto nástroje pre prácu s Labanovou notáciou sa zameriavajú buď na grafické znázornenie zápisu notácie alebo na aplikovanie pohybu na model postavy. V našej práci sa však pokúsime čiastočne nahliadnuť aj do oblasti detekovania pohybu na animovanej postave a následného

zápisu do danej štruktúry, tzn. zápisu pohybu z animácie a nie z grafického znázornenia Labanovej notácie.

3

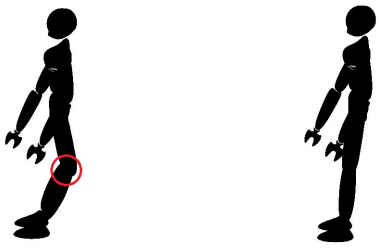
Návrh riešenia

Predtým ako sa budeme venovať návrhu riešenia a podrobnostiam jeho implementovania, zhrnieme, aký problém riešime a aké sú ciele a požiadavky pre toto riešenie. Cieľom tejto práce, je použiť štruktúru LabanXML dokumentu pre zápis vyššie popísanej Labanovej notácie tak, aby po načítaní dát z dokumentu bol následne vytvorený animovaný pohyb vopred vymodelovanej postavy v programe Blender, aplikácii pre 3D grafiku, a aby aspoň sčasti zodpovedal reálnemu pohybu reálnej postavy. Zároveň úlohou tejto práce je aj pokus o zápis pohybu do xml formátu z už vopred naanimovaných pohybov postavy, a to tak, aby po znovuaplikovaní výstupného xml súboru na nový vstupný nenaanimovaný model, bol pohyb novonaanimovanej postavy približne rovnaký ako pohyb vstupnej animácie. Pri animácii sa budeme sústrediť len na niektoré významné ľudské pohyby, čo znamená, že menej dôležité črty, ktoré nie sú natoľko podstatné, môžeme pri animovaní zanedbať. Konkrétnejšie, budeme sa venovať základným pohybom nôh (chôdzi) a rúk.

3.1 Popis problematiky

V tejto časti si popíšeme, aké problémy sa vyskytujú pri aplikovaní pohybov na model postavy, a aké aspekty je potrebné riešiť. Zhrnieme, ktorými aspektami sa v tejto práci zaoberáme, a ktoré naopak neriešime.

Našim cieľom je zaoberať sa a pokúsiť sa riešiť tieto problémy:
Odstrániť prípadné nereálne ohýbanie sa kostí v kĺboch modelu postavy - je potrebné vyhnúť sa neprirodzeným ohybom v kĺboch, napríklad prehybu kolena do opačnej strany.



Obr. 3.1: Naľavo ukážka neprirodzeného ohýbania sa kĺbov. Napravo korektný stav.

Minimalizovať chyby pri polohe modelu pod úrovňou roviny piat - v scéne by nemalo viditeľne dochádzať k situácii, keď chodidlá modelu sú pod rovinou ich piat, model by si mal tak udržať svoju líniu.

Obr. 3.2: Naľavo ukážka päty pod danou úrovňou. Napravo korektný stav.

Minimalizovanie chýb pri interpolácii - pri interpolácii medzi dvama kľúčovými snímkami môže dochádzať k vzniku medzipolohy, ktorá nie je chcená, respektíve sa príliš vymyká od reálnej sitácie. Túto chybu treba ošetriť tak, aby nechcená situácia bola čo najmenej badateľná.

Vzájomná závislosť pozície chodidiel pri prenášaní váhy - pri tvorbe riešenia bude potrebné brať ohľad aj na to, že pozície nôh pri prenášaní váhy človeka, napríklad pri chôdzi, vzájomne na seba naväzujú a ovplyvňujú sa.

Naopak, pri riešení nie je našim cieľom zaoberať sa týmito javmi:
Zápis nereálneho pohybu v LabanXML dokumente - našou úlohou nebude zisťovať, či zápis pohybu v LabanXML dokumente má alebo nemá zmysel, tzn. nezaoberáme sa otázkou, či pohyb zapísaný v súbore je reálne uskutočniteľný, keďže samotný zápis Labanovej notácie nám nebráni vytvoriť neuskutočniteľný pohyb zápisom.

Prenášanie pohybov na iné štruktúry - nebudeme sa zaoberať prenosom tohto pohybu na iné kostry, ktoré by mohli byť zložené z podobných segmentov (kostí).

3.2 Návrh riešenia

V tejto časti sa budeme venovať nášmu návrhu riešenia pre animovanie pohybu z dát uložených v súbore.

Ako sme už spomínali, riešenia v časti 2 sú väčšinou zamerané len na grafické zobrazenie Labanaovej notácie, jej zápisu a zobrazenia tejto notácie. Síce sme spomenuli nástroj LabanEditor, ktorý zobrazuje daný pohyb v 3D grafike, no nezaoberá sa aj zápisom pohybu z animovanej postavy. My sa však v našej práci pokúsime aspoň naivne podporovať aj túto možnosť a zistiť, ako takýto prístup ovplyvní výslednú animáciu.
Keďže sa chceme venovať animovaniu 3D modelu postavy z dát uložených v súbore, tak sme sa rozhodli siahnuť po riešení v podobe pythonovského skriptu pre 3D grafický softvér Blender, ktorý nám umožňuje vytvoriť si model postavy a spúšťať skripty pomocou Blender Python API.

Základné princípy - základom celého aplikovania pohybu na model postavy by malo byť nastavenie polôh pivotných kostí, ktorých poloha je rozhodujúca pre výsledný vizuálny efekt daného pohybu, respektíve danej pózy. Vopred definované polohy pivotných kostí by mali byť uložené v databáze, z ktorej ich dokážeme načítať na zákalde zápisu Labanovej notácie v LabanXML súbore a to pre každú pózu osobitne. Po aplikácii polôh načítaných zo súboru vložíme kľúčovú snímku kvôli zapamätaniu si tejto polohy pre animáciu. Následne po ukončení fázy vkladania kľúčových snímkov by bolo

možné spustiť animáciu, v ktorej interpoláciu medzi kľúčovými snímkami zabezpečí Blender.

Neprirodzené ohýbanie sa kostí v kĺboch - tetno problém budeme riešiť pomocou inverznej kinematiky. Vloženie inverznej kinematiky na určité časti modelu postavy zabezpečí to, aby pri interpolácii medzi dvoma kľúčovými snímkami nedochádzalo k ohýbaniu v kĺboch nesprávnym smerom. Inverzná kinematika vlasnte obmedzí uhly rotácii v daných kĺboch.

Minimalizovanie chýb polohy modelu pod úroveň piat - pridanie pivotných kostí na pätu a na prsty na nohe spoločne s definovaním správnych parametrov pri inverznej kinematike minimalizuje chybovosť pri animácii a obmedzí zachádzanie modelu pod hranicu podlahy.

Minimalizovanie chýb pri interpolácii - medzi dva kľúčové snímkami obsahujúce začiatočnú a koncovú pózu vykonávaného pohybu vložíme ďalší kľúčový snímok, ktorý bude obsahovať medzipolohu medzi týmito dvoma pózami. Táto medzipoloha sa taktiež načíta zo samostatnej databázy medzipolôh rovnakým spôsobom ako základné polohy. Pred vložením však musíme zdetekovať aká je poloha v začiatočnom kľúčovom snímku a koncovom, a na základe nich vybrať z databázy vhodnú medzipolohu s následným aplikovaním na model postavy.

Pri zápise pohybu do formátu LabanXML z už vopred naanimovanej postavy zvolíme naivný prístup, ktorý bude detekovať pózu daného modelu v presných intervaloch. Správnosť detekcie bude záležať od toho, či je daná póza presne uložená v databáze, ak nie, tak vyberieme alogritmom vyhodnotenú najlepšiu vyhovujúcu pózu, ktorá sa ale môže od presnej líšiť. Úspešnosť detekcie bude záležať aj od zvolenia kontrolného intervalu, ktorého nastavenie ponecháme na používateľa. Ten si tak môže zvážiť, či je výsledok preňho postačujúci.

4

Riešenie

V tejto kapitole sa budeme zaoberať samotným riešením daného problému. Keďže naše riešenie je implementované ako súčasť nástroja pre 3D grafiku, aplikácie Blender, budeme mu venovať pozornosť v samostatnej podkapitole. Uvedieme, aké možnosti nám ponúka pre našu prácu. Taktiež sa pokúsime vysvetliť ako v ňom niektoré prvky pracujú. V tejto časti sa potom budeme venovať aj samotným podrobnostiam našej implementácie.

4.1 Blender a Blender Python API

Táto časť popíše niektoré prvky nástroja Blender, ktorý budeme pri našej implementácii využívavať.

Kostra (Armature) - je to vlastne skeleton nášho modelu, ktorú môžeme použiť v Blenderi pri rôznych objektoch. Je zložená z kostí (Bones) a má stromovú štruktúru. Tieto kosti sú v kostre uložené v takzvanej pokojovej pozícii (Rest Position). Táto pozícia sa dá meniť v editačnom móde (Edit Mode) [blec].

Póza (Pose) - definuje nám pre kostru zmeny a vzťahy v nej na objektovej úrovni. Každá kosť v kostre má vytvorený pozičný kanál (Pose Channel), ktorý obsahuje in-

formáie o rotácii, škálovaní, translácii a umožňuje vytvoriť obmedzenia (Constraints) pri prepojení s iným pozičným kanálom alebo objektom. Pristupovať k póze kostry môžeme v pozičnom móde (Pose Mode) [blec].

Kosti kostry v editačnom móde budeme nazývať dátové kosti (DataBones) a kosti v pozičnom móde pozičné kosti (PoseBones).

Akcia (Action) - ukladá animačné krivky (Animation Curves) pre pozičné kanály a ich obmedzenia. Túto akciu môžeme definovať ako kolekciu všetkých animačných zmien, ktoré chceme aplikovať na pózu kostry [blec].

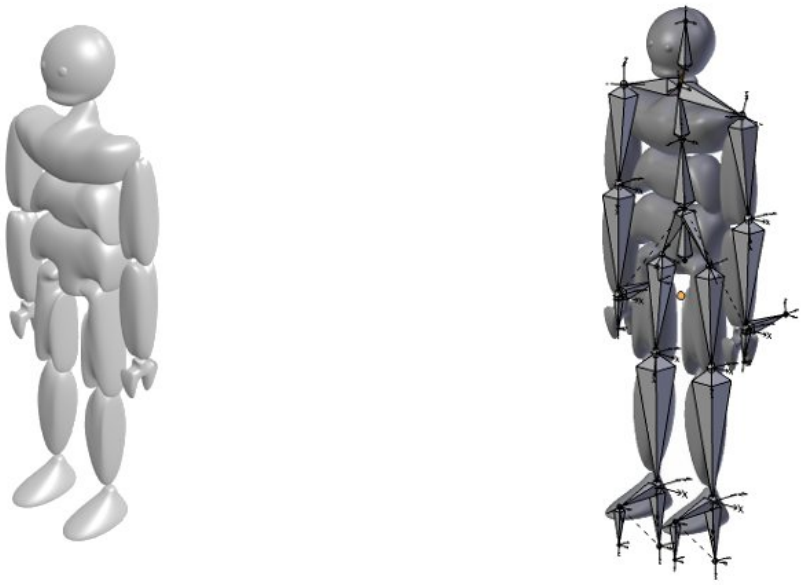
4.2 Implementácia

V tejto časti sa budemem venovať implementačným podrobnostiam. Uvedieme spôsob manipulovania s modelom kostry, ukladania dát, spomenieme metódy a časti ich kódu, ktoré tvoria samotné riešenie tohto problému. Popri tom spomenieme aj niektoré problémy, ktoré bolo potrebné vyriešiť.

4.2.1 Model postavy a kostra

Predpokladom pre riešenie problému aplikovania pohybov na postavu je mať vymodelovanú postavu, na ktorej budeme testovať náš prístup. Keďže Blender je nástroj pre 3D grafiku, túto postavu si dokážeme pomocou neho vytvoriť.

Model postavy použitý pre naše účely je zobrazený na obrázku 4.1. Na tomto istom obrázku je zobrazený model spoločne s kostrou, ktorá slúži na to, aby sme na tento medel mohli aplikovať pohyb. Kostra sa skladá z kostí, ktoré sú pomenované podľa tej časti modelu, v ktorej sa nachádzajú a to tak, že v názve vyjadríme časť prislúchajúceho ľudského tela a stranu. Kosti v kostre rozdeľujeme na základné a pivotné. Pomenovania základných kostí kostry sú nasledovné: *Chrbtica1*, *Chrbtica2*, *Chrbtica3*, *Krk*, *Hlava*, *Klucna_kost_L*, *Rameno_L*, *Predlaktie_L*, *Ruka_L*, *Bedro_L*, *Stehno_L*,

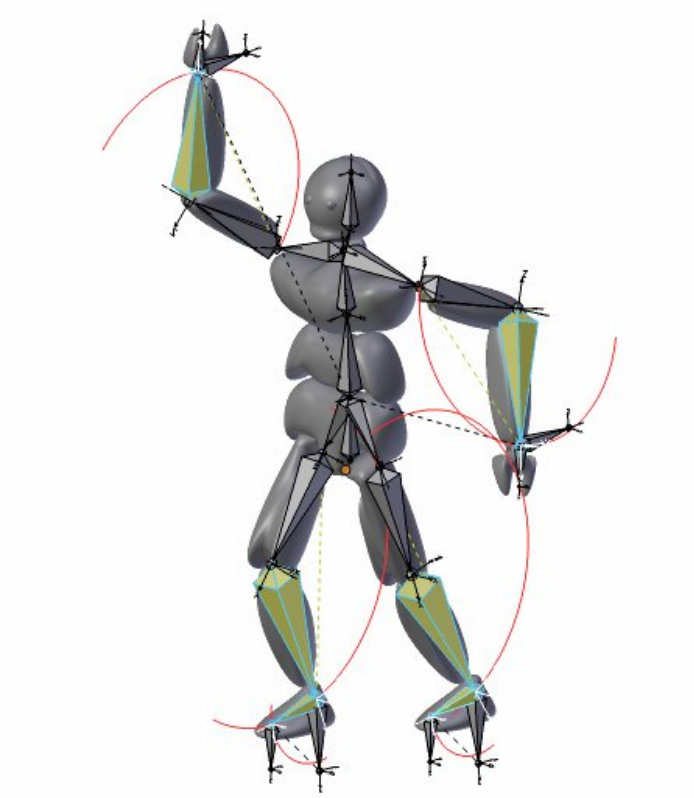


Obr. 4.1: Vľavo samotný model, vpravo model postavy s kostrou.

Predkolenie_L, *Chodidlo_L*, *Klucna_kost_R*, *Rameno_R*, *Predlaktie_R*, *Ruka_R*, *Bedro_R*, *Stehno_R*, *Predkolenie_R*, *Chodidlo_R*.
Pomenovania pivotných kostí je takéto:
Pivot_noha_L, *Pivot_noha_prsty_L*, *Pivot_ruka_L*, *Pivot_noha_R*, *Pivot_noha_prsty_R*, *Pivot_ruka_R*.

Pre animovanie kostry sú pre nás najdôležitejšie pivotné kosti a koreňová kosť hlavnej kostry (*Chrbtica1*). Pomocou zmeny polôh pivotných kostí a koreňovej kosti budeme meniť polohu ostatných kostí vďaka inverznej kinematike aplikovanej na niektoré kosti kostry.

Na obrázku 4.2 môžeme pozorovať ako je aplikovaná inverzná kinematika na kostru a ako sú na jednotlivých kostiach použité rotačné obmedzenia, aby nedochádzalo k nesprávnemu ohybu v oblasti kĺbov.



Obr. 4.2: Kostra s inverznou kinematikou a rotačnými obmedzeniami.

4.2.2 Databázy pohybov

Základom možnosti aplikovania pohybov na model postavy je mať databázu pohybov, z ktorej by sme mohli tieto pohyby vyberať a postupne aplikovať. V našom prípade to bude databáza zapísaná v xml súbore. Z tejto databázy vyberáme polohy pivotných kostí, na ktorých túto zmenu polôh prevedieme. Polohy kostí v databáze sú reprezentované vektorom, ktorý nám udáva súradnice miesta, na ktoré sa má pivotná kosť premiestniť vzhľadom na jej pokojovú pozíciu.

Pri implementácii použijeme spolu dva typy databáz:

- databáza pre základné polohy/pohyby Labanovej notácie
- databáza pre medzipohyby/medzipohyby

Databáza pre základné pohyby obsahuje informáciu o začiatočných a koncových pózach kostry, tak ak je to určené v Labanovej notácii, v ktorej sú priamočiaro zadané len začiatočné a koncové pózy. To ako sa má kostra pri animácii správať medzi týmito dvoma pózami definujeme v databáze pre medzipohyby.
Polohy pivotných kostí pre určitý pohyb určitej časti tela sú v databáze pre základné pohyby uložené nasledovne:

```
<database>
  <BodyPart_Side>
    <Direction1>
      <L>
        <Pivot_bone_name1>(2.0, 0.0, 0.0)</Pivot_bone_name1>
        <Pivot_bone_name2>(2.0, 1.3, 0.0)</Pivot_bone_name2>
        ...
        <Pivot_bone_nameN>(1.0, 0.9, 0.4)</Pivot_bone_nameN>
      </L>
      <M>
        <Pivot_bone_name1>(1.0, 0.0, 0.4)</Pivot_bone_name1>
        <Pivot_bone_name2>(2.4, 1.3, 0.6)</Pivot_bone_name2>
        ...
      </M>
      <U>
        <Pivot_bone_name1>(1.0, 0.9, 0.4)</Pivot_bone_name1>
        <Pivot_bone_name2>(0.4, 1.5, 0.6)</Pivot_bone_name2>
        ...
      </U>
    </Direction1>

    <Direction2>
      ...
    </Direction2>

    ...
  </BodyPart_Side>
  ...
<\database>
```

Element *Pivot_bone_nameN* určuje názov pivotnej kosti, na ktorú sa má aplikovať polohový vektor. Elementy *L*, *M* a *U* určujú úroveň (level) pohybu tak ako je to známe z Labanovej notácii. Elementom *Direction* určujeme, pre ktorý smer pohybu je poloha určená a elementy *BodyPart_Side* nám vravia, pre ktorú časť tela sa majú dané zmeny vykonať.

Polohy pivotných kostí pre medzipohyb určitej časti tela sú v databáze pre medzipohyby uložené takýmto spôsobom:

```
<database>
  <BodyPart\_Side1_BodyPart\_Side2>
    <Direction1_Direction2>
      <L_L>
        <Pivot_bone_name1>(2.0, 0.0, 0.0)</Pivot_bone_name1>
        <Pivot_bone_name2>(2.0, 1.3, 0.0)</Pivot_bone_name2>
        ...
        <Pivot_bone_nameN>(1.0, 0.9, 0.4)</Pivot_bone_nameN>
      </L_L>
      <L_M>
        ...
      </L_M>
      <L_U>
        ...
      </L_U>
      <M_M>
        ...
      </M_M>
      <M_U>
        ...
      </M_U>
    </Direction1_Direction2>
  <\BodyPart\_Side1_BodyPart\_Side2>

  ...
<\database>
```

Názvy elementov pri databáze medzipolôh sú zostavené ako dvojice názvov elementov z databázy základných polôh, kde pravá časť mena elementu zodpovedá začiatočnej a druhá časť koncovej póze kostry.
Informácie v oboch týchto databázach je potrebné vopred si definovať. My sme v tejto práci používali neúplú databázu, v ktorej neboli doplnené všetky možné kombinácie pohybov, len tie, ktoré sme pri práci potrebovali na testovanie.

4.2.3 Načítanie LabanXML dokumentu a aplikovanie póz

Ak máme pripravený model, tak môžeme načítať samotný LabanXML súbor a aplikovať v ňom zapísaný pohyb na model postavy. Postupne načítavame pohyby pre jednotlivé časti tela v každom takte zápisu.
Najprv je potrebné načítať časť pre prenášanie váhy (Support), pri ktorej sa vyskytuje problém, že ak načívame túto časť pre pravú a ľavú časť tela samostatne, práve načítaný pohyb na jednej strane môže úplne zmeniť pohyb na strane druhej. Preto

je potrebné časť *Support* v LabanXML súbore vyhodnotiť súčasne pre obe strany. Pri celkovom vyhodnocovaní časti *Support* je preto ešte potrebné do databázy pre základné polohy doplniť časť, keď je váha nesená na oboch častiach tela:

```
<Support>
  <DirectionR1_DirectionL1>
    <LevelR_LevelL>
      <Pivot_bone_name1>(2.0, 0.0, 0.0)</Pivot_bone_name1>
        <Pivot_bone_name2>(4.0, 1.0, 0.1)</Pivot_bone_name2>
        ...
        <Pivot_bone_nameN>(1.0, 3.9, 0.4)</Pivot_bone_nameN>
    </LevelR\_LevelL>
  </DirectionR1\_DirectionL1>

  ...

  <DirectionRN_DirectionLN>
    ...
  </DirectionRN\_DirectionLN>
</Support>
```

Po načítaní a vyhodnotení pohybov pre časť *Support* aplikujeme tieto pózy na postavu a vložíme kľúčové snímky pre každú dobu (beat) jedného taktu (bar).
Potom taktiež vyhodnotíme samostatné časti tela, pri ktorých nám stačí vyhodnocovať každú stranu tela osobitne, keďže tieto pohyby sa neovplyvňujú. V práci sme sa nezaoberali všetkými časťami tela, ale ich vyhodnocovanie by fungovalo na rovnakom princípe ako spracovanie časti rúk, ktoré v práci zahŕňame.
Časť kódu pre samotné priradzovanie pozícii danej kostry vyzerá nasledovne:

```
try:
  #prechadzame kosti v databaze
  for i in positions[posit][direct][lev].keys():
    #priradime maticu z positions databazy
    _armature.pose.bones[i].location = positions[posit][direct][lev][i]
    bpy.ops.anim.keyframe_insert_menu(type='Location')
    support_keyframes.append(scn.frame_current)
except Exception as e1:
  print ('Couldn\'t set location:', e1)
```

4.2.4 Vkladanie medzipolôh

Ak už máme v Blenderi vložené kľúčové snímky pre základné pohyby, môžeme spustiť prvotnú animáciu, lenže interpolácia medzi jednotlivými polohami môže byť čudná

a pre pohyb človeka neprirodzená alebo nereálna. Navyše, medzi dvom kľúčovými snímkami, medzi ktorými je veľká vzdialenosť, môže dôjsť k nechcenej intrpolácii, napríklad ak poloha chodidla v susedných kľúčových snímkoch je rovnaká, tak Blender sa pokúša interpolovať aj medzi nimi, čo môže byť príliš viditeľné v samotnej animácii. Vložením medzipolôh minimalizujeme nevhodnú interpoláciu a zároveň docielime nami chcený efekt pre pohyb medzi dvoma základnými pózami.

Na to, aby sme vedeli ktorú medzipolohu z databázy pre medzipolohy vybrať a aplikovať, musíme zistiť polohy v jednotlivých kľúčových snímkoch. Problémom však je, že jednotlivé pohyby majú rôznu dĺžku vykonávania a pri porovnávaní dvoch susedných kľučových snímkoch nemusí byť niektorá časť tela ešte úplne v základnej polohe a mi nesprávne vyhodnotíme medzipolohu. Preto si pri vkladaní polôh musíme pamätať kľúčové snímky pre jednotlivé časti tela, a potom pre tieto jednotlivé snímky vkladať medzipolohy samostatne pre každú časť tela modelu.

Medzipolohy potom vkladáme rovnako ako základné polohy s tým rozdielom, že používame databázu medzipolôh a nie základných polôh.

4.2.5 Naivná detekcia a zápis do LabanXML súboru

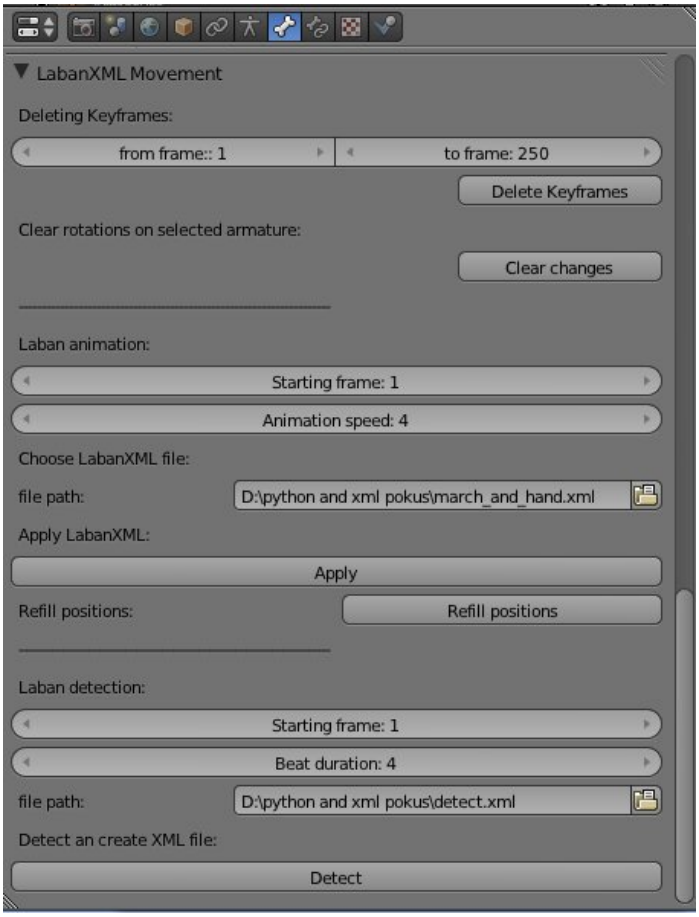
Pri detekcii pohybu z naanimovaného modelu postavy zvolíme naivný prístup. Tento prístup bude spočívať v tom, že sa pri detekcii budeme spoliehať na to, že zmeny základných pohybov postavy sa vykonávajú v pravidelných intervaloch. Zmenu hodnoty tohto intervalu ponecháme na používateľa, ktorý sa sám rozhodne, či zvolený interval a výsledný zápis čo najlepšie zodpovedá jeho požiadavkám.
Pri prechádzaní snímkov animácie po týchto intervaloch budeme hľadať pre dané časti tela, čo najvhodnejšiu polohu v databáze základných polôh. Algoritmus považuje za najvhodnejšiu polohu takú, ktorej čo najväčší počet pivotných kostí kostry v danej pozícii má rovnaké polohy ako polohy pivotných kostí v databáze. Ak je takýchto polôh viac, algoritmus vyberie jedinú takúto pozíciu.

Nevýhodou tohto tohto prístupu je, že nie vždy dostaneme takú istú pozíciu aká

bola v pôvodnej animácii, no pre niektoré animácie je postačujúca. Vylepšenie tohto algoritmu ponechávame do ďalšej práce.

4.2.6 Grafické používateľské rozhranie - GUI

Kvôli tomu, aby si užívateľ mohol vyskúšať prácu s LabanXML, animáciou a zápisom animácie do LabanXML, sme vytvorili jednoduché grafické prostredie znázornené na obrázku 4.3 zabudované v grafickom rozhraní aplikácie Blender, ktoré umožňuje aplikovať pohyby na model postavy, dopĺnať medzipozície a následne vytvoriť animáciu. V tomto paneli sú pridané možnosti aj na detekciu póz pre animovanú postavu s následným zápisom do LabanXML súboru s následno možnosťou opätovného aplikovania.



Obr. 4.3: Ovládací panel pre použitie LabanXML.

5

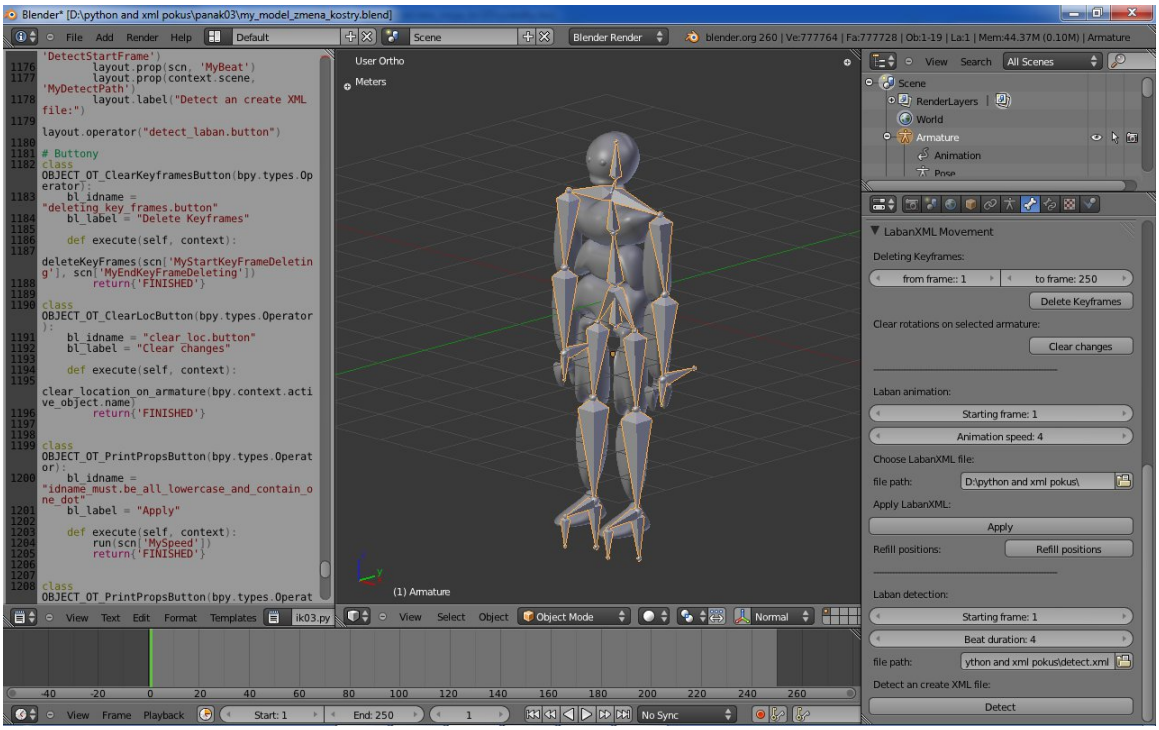
Výsledky

Táto kapitola je venovaná samotným výsledkom nášho riešenia pre použitie LabanXML v aplikácii Blender. Predstavíme ovládanie a používanie skriptu, ako aj ukážky aplikovania pohybu na model postavy.

5.1 Ovládanie

Po spustení skriptu v textovom editore (Text Editor) programu Blender získame možnosť aplikovať pohyby z LabanXML súboru na nami označenú kostru. Táto možnosť nám pribudene v paneli pre vlastnosti (Properties) a to v časti pre vlastnosti kosti (Bone) pod názvom *LabanXML Movement*. Po označní samotnej kostry si môžeme zvoliť xml súbor so zápisom Labanovej notácie (LabanXML súbor), ktorý chceme na danú kostru aplikovať stlačním voľby *Apply*. Ak chceme výslednú animáciu doplniť o medzipolohy, použijeme možnosť *Refill positions*. Ešte pred samotným aplikovaním pohybov na model si môžeme zvoliť rýchlosť animácie (*Animation speed*) a začiatočný snímok animácie (*Starting frame*). Výslednú animáciu si potom môžeme spúšťať v paneli Časový prehľad (Timeline).

V danom paneli je nám poskytnutá aj možnosť detekovania pohybu a vytvorenia LabanXML súboru a to možnosťou *Detect*. Pred samotnou detekciou si môžeme navoľť počiatočnú snímku detekcie (*Starting frame*) a intervat detekcie (*Beat duration*), ako

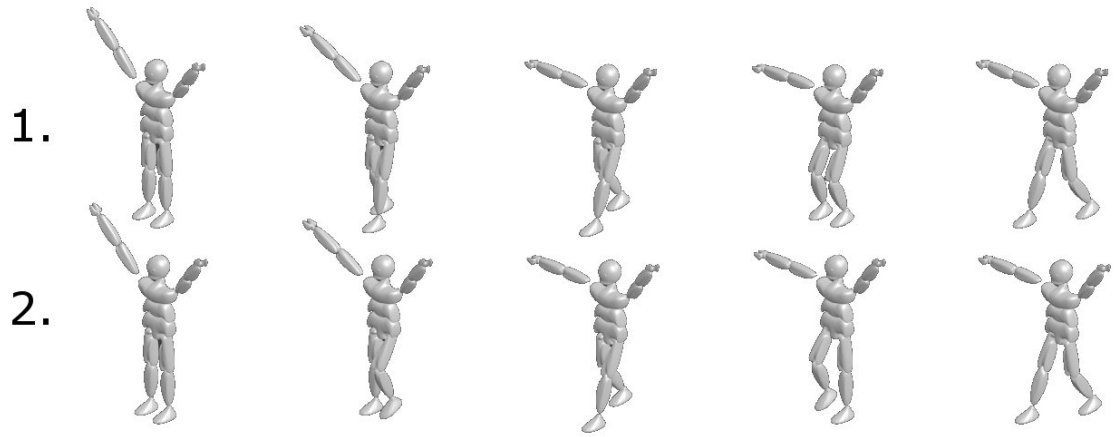


Obr. 5.1: Prostredie Blenderu s ovládaním.

aj názov a miesto uloženia LabanXML súboru, do ktorého sa zapíše zdetekovaný pohyb.

5.2 Ukážky výsledkov

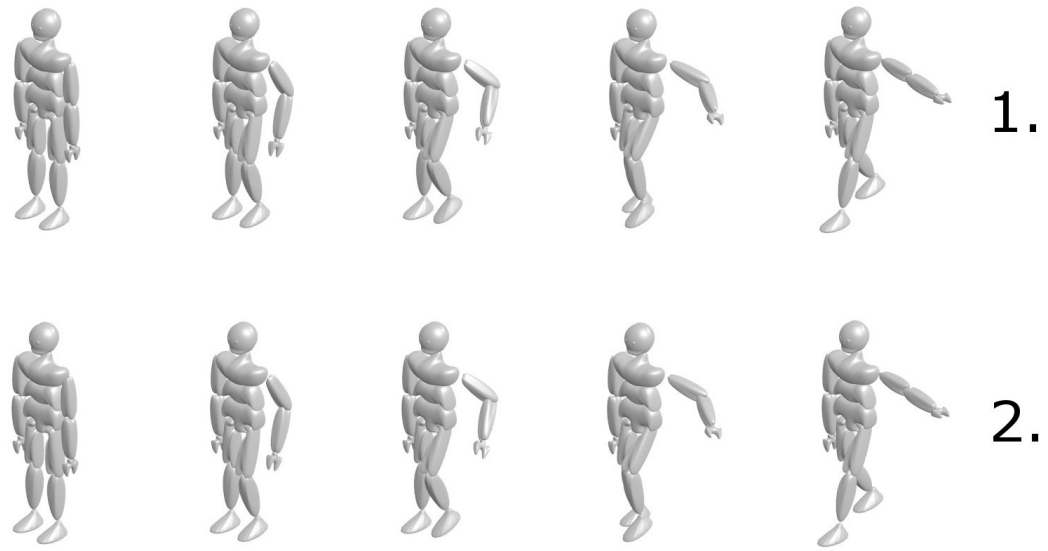
Keďže výsledkom našej práce je animácia, tak v tejto časti uvedieme výsledky v podobe obrázkov, na ktorých sa budeme snažiť priblížiť samotnú animáciu.



Obr. 5.2: Animácia bez medzipolôh (1.), s doplnenými medzipolohami (2.) I.

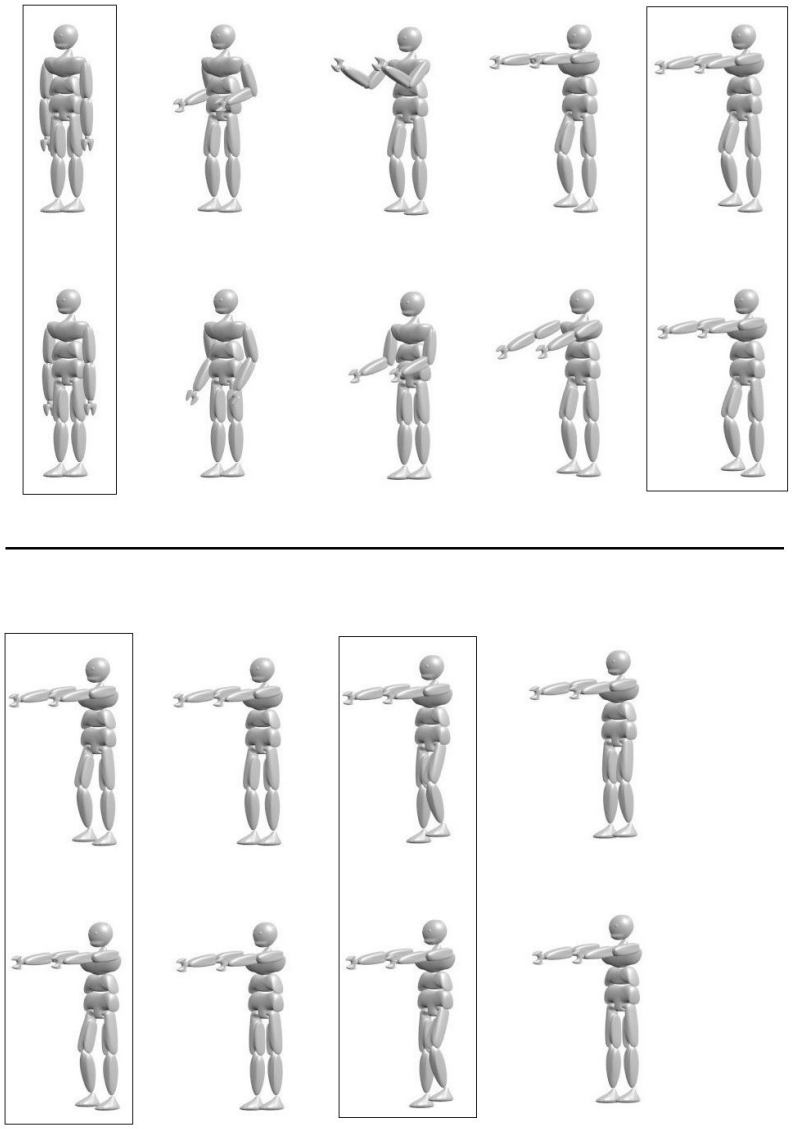
Obr. 5.3: Animácia bez medzipolôh (1.), s doplnenými medzipolohami (2.) II.

Na obrázkoch 5.2 a 5.3 porovnávame animáciu, ktorá vznikla po aplikovaní základných pohybov z LabanXML súboru a tú istú animáciu s doplnenými medzipolohami z databázy pre medzipolohy.



Obr. 5.4: Pôvodná animácia (1.), detekovaná animácia so zhodným intervalom doplnená o medzipolohy (2.).

Na obrázku 5.4 vidíme pôvodnú animáciu vytvorenú z LabanXML súboru a animáciu, ktorá vznikla detekovaním pôvodnej animácie. Pri detekcii bol použitý rovnaký interval ako mala pre svoj pohyb pôvodná animácia.



Obr. 5.5: Zmeny pri detekcii oproti pôvodnej animácii. V rámčekoch sú zhodné pozície.

Na obrázku 5.5 je znázornená chyba pri určení nepresného intervalu detekcie a znázornenie rozdielov. V rámčekoch vidíme pozície, ktoré boli správne detekované.

Obr. 5.6: Animácia I.

Obr. 5.7: Animácia II.

Na obrázkoch 5.6 a 5.7 vidíme ďalšie animácie, ktoré vznikli načítaním pohybov z LabanXML súboru.

Záver

V tejto bakalárskej práci sme sa zaoberali návrhom a implementáciou riešenia pre aplikovanie pohybov zapísaných v xml dokumente na model postavy s následným vytvorením animácie. Pokúsili sme sa aj o obrátený proces, proces zápisu pohybov z animácie do súboru LabanXML. Našim cieľom bolo poukázať na to, že daným spôsobom by sa dalo pracovať s týmto zápisom a modelom humanoidnej postavy, aj keď Labanova notácia nie je v niektorých skutočnostiach priamočiara a jednoznačná.

V prvej kapitole sme si postupne priblížili a ozrejmili niekoľko základných pojmov týkajúcich sa tejto problematiky. Oboznámili sme sa so zápisom Labanovej notácie pre pohyb, ako aj s jej zápisom do xml štruktúry, LabanXML. V ďalšej kapitole sme hovorili o motivácii tejto práce a spomenuli aj niektoré známe riešenia danej alebo príbuznej problematiky. Po stručnom popise riešeného problému sme sa zaoberali návrhom na jeho riešenie spolu s analýzou vyskytujúcich sa problémov a ich prípadným odstránením alebo minimalizovaním. Riešenie sme implementovali ako pythonovský skript pre Blender, modelovací a animačný nástroj pre 3D grafiku, ktorého súčasťou je rohranie jazyka Python, ktoré nám s daným skriptom umožnilo pracovať. Pri riešení sme použili nami vytvorené xml databázy polôh, ktoré obsahovali len základné pohyby pre pohyb rúk a prenos váhy postavy, ktoré boli pre našu prácu potrebné. Načrtli sme aj spôsob naivnej detekcie z vopred nanimovaného pohybu modelu postavy. V poslednej kapitole sme priblížili dosiahnuté výsledky tejto práce.

V budúcnosti by sme naše riešenie chceli obohatiť o viacero druhov pohybov, hlavne v xml databázach a pokúsiť sa spraviť inteligentnejšiu detekciu pohybu so vstupnej animácie. Skript by sa dal rozšíriť aj o aplikovanie rôznych druhov medzipohybov, ktoré by mohli zodpovedať rôznym tanečným štýlom s možnosťou voľbu štýlu prenechať na používateľa, ktorý by tak mohol viac interagovať s aplikáciou. To by však podnietilo aj rozšírenie počtu databáz medzipohybov.

Zoznam literatúry

[blea] Blender documentation contents - blender 2.63.7 - api documentation [online][cit. 25.5.2012]. http://www.blender.org/documentation/blender_python_api_2_63_7.

[bleb] blender.org - home [online][cit. 25.5.2012]. http://www.blender.org.

[blec] blender.org - how armatures work [online][cit. 27.5.2012]. http://www.blender.org/development/release-logs/blender-240/how-armatures-work/.

[BPSM+08] Tim Bray, Jean Paoli, C. Michael Sperberg-McQueen, Eve Maler, and François Yergeau. Extensible markup language (xml) 1.0 (fifth edition). World Wide Web Consortium, Recommendation REC-xml-20081126, November 2008.

[CWRF05] Tom Calvert, Lars Wilke, Rhonda Ryman, and Ilene Fox. Applications of computers to dance. *IEEE Computer Graphics and Applications*, 25:6–12, 2005.

[HG05] Ann Hutchinson Guest. *The System of Analyzing and Recording Movement*. Routledge, Great Britain, 4th edition, 2005.

[HN06] Kozaburo Hachimura and Minako Nakamura. An xml representation of labanotation, labanxml, and its implementation on the notation editor labaneditor2, 2006.

[KHN02] K Kojima, K Hachimura, and M Nakamura. Labaneditor: Graphical editor for dance notation. *Proceedings 11th IEEE International Workshop on Robot and Human Interactive Communication*, pages 59–64, 2002.

[laba] Introduction to labanotation [online][cit. 5.1.2012]. http://user.uni-frankfurt.de/~griesbec/LABANE.HTML.

[labb] Labanreader [online][cit. 26.5.2012]. http://dance.osu.edu/3_research_gallery/laban_reader.html.

[labc] Labanwriter [online][cit. 26.5.2012]. http://dance.osu.edu/3_research_gallery/laban_writer.html.

[MBBI05] Oliver MORAVCÍK, Pavol BOŽEK, Pavol BEZÁK, and Miriam IRINGOVÁ. *CONTRIBUTION TO CREATING OF ROBOTIZED WORKPLACE IN VIRTUAL SCENE*. 2005.

[mina] 19.7. xml.dom.minidom — lightweight dom implementation - python v2.7.3 documentation [online][cit. 24.1.2012]. http://docs.python.org/library/xml.dom.minidom.html.

[minb] Minidom - pythoninfo wiki [online][cit. 20.1.2012]. http://wiki.python.org/moin/MiniDom.

[Par07] R. Parent. *Computer animation: algorithms and techniques*. Morgan Kaufmann Series in Computer Graphics and Geometric Modeling. Morgan Kaufmann, 2007.

[pic] Udn - three - animationnodes [online][cit. 30.5.2012]. http://udn.epicgames.com/Three/AnimationNodes.html.

[pyt] Python programming language - official website [online][cit. 25.5.2012]. http://www.python.org.

[SK96] Laszlo Szirmay-Kalos. *Theory of Three Dimensional Computer Graphics*. Akademiai Kiado, Budapest, 1996.

[WW91] Alan Watt and Mark Watt. *Advanced animation and rendering techniques*. ACM, New York, NY, USA, 1991.

Prílohy

Príloha A - Obsah elektronického média

- pythonovský skript **apply.py**
- súbor s modelom postavy **model.blend**
- videá s ukážkami v priečinku **video**
- obrázky s ukážkami v priečinku **images**
- xml databázy v priečinku **databases**
- príklady LabanXML v priečinku **examples**
- elektronická verzia bakalárskej práce **RuhalovskyJan.pdf**