VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

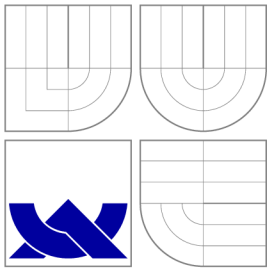
POKROČILÁ ANIMACE POSTAV

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

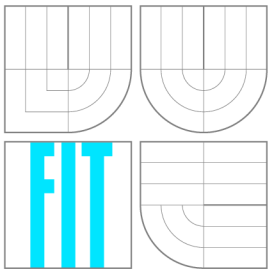
AUTOR PRÁCE
AUTHOR

MICHAL MATÝŠEK

BRNO 2013



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

POKROČILÁ ANIMACE POSTAV

ADVANCED CHARACTER ANIMATION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — MICHAL MATÝŠEK
AUTHOR

VEDOUCÍ PRÁCE — Ing. RUDOLF KAJAN
SUPERVISOR

BRNO 2013

Abstrakt

Tato práce popisuje moderní techniky animací postav v reálném čase a jejich praktickou implementaci v jazyce C++ s použitím knihoven DirectX a PhysX. První část textu se zabývá výkladem vybraných způsobů animace a je zaměřena především na techniky animace pomocí klíčových snímků, skeletální animaci, metodu morfing a ragdoll simulaci. Následuje popis použitých knihoven DirectX a PhysX a popis metody snímání pohybu postav s využitím softwaru iPi Desktop Motion Capture. Druhá část práce je zaměřena na konkrétní způsoby implementace popsaných animačních technik.

Abstract

This thesis describes modern real-time character animation techniques and their implementation in C++ language using DirectX and PhysX libraries. The first part of the text deals with a presentation of selected animation principles and is focused mailny on key-frame animation, skeletal animation, morphing and ragdoll simulation. Then follows a description of used DirectX and PhysX libraries and description of motion capturing, using iPi Desktop Motion Capture software. The second part of this thesis is focused on particular ways of implementation of the described animation techniques.

Klíčová slova

Animace postav, Skeletální animace, Animace pomocí klíčových snímků, Morfing, Ragdoll simulace, Snímání pohybu, Skinning, DirectX, PhysX, iPi Desktop Motion Capture

Keywords

Character animation, Skeletal animatoin, Key frame animation, Morphing, Ragdoll simulation, Motion Capture, Skinning, DirectX, PhysX, iPi Desktop Motion Capture

Citace

Michal Matýšek: Pokročilá animace postav, bakalářská práce, Brno, FIT VUT v Brně, 2013

Pokročilá animace postav

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Rudolfa Kajana a v seznamu literatury jsem uvedl všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Michal Matýšek
15. května 2013

Poděkování

Děkuji vedoucímu práce Ing. Rudolfu Kajanovi za odbornou pomoc a podnětné připomínky při psaní bakalářské práce.

Děkuji právnímu zástupci společnosti Valve Corporation za poskytnutí souhlasu s využitím některých autorských děl Valve Corporation v této práci a v demonstrační aplikaci.

Obsah

Kapitola 1

Úvod

Lidský pohyb je komplexní souhrn projevů, který zahrnuje vedle pohybu segmentů těla a těla jako celku v prostoru i neverbální výrazové prostředky jako jsou gesta, mimika, posturika a další. Tyto projevy jsou důležitou součástí lidské komunikace a společně s dalšími faktory vytvářejí osobnost postav a definují jejich chování. Animace pohybových projevů jsou tedy zásadní při tvorbě charakteru postav nejen v počítačových hrách.

Spolu s rozvojem herního průmyslu a se stále rostoucí snahou o realistickou vizuální prezentaci scén ve hrách je nutné i pohyb postav napodobit co nejvěrněji, což vzhledem k jeho složitosti není snadné. Člověk je navíc na vnímání pohybových projevů zvyklý, proto jakékoliv odchylky od očekávaného přirozeného pohybu upoutávají pozornost pozorovatele a negativně ovlivňují výsledný dojem ze scény.

Aby bylo možné v reálném čase dosáhnout autentické animace postav i na běžných počítačích, je nutné využívat funkcí moderních grafických karet. Tato práce se zabývá technikami animací, které pro urychlení výpočtů používají grafické akcelerátory a jsou tedy vhodné pro implementaci do animačních systémů v počítačových hrách.

Text práce je rozdělen do pěti oddílů. Po úvodu následuje kapitola věnována teoretickému popisu animačních technik. Před samotný výklad je vložen blok krátce popisující historii a vývoj metod, které moderním technikám předcházely. Další podkapitola již vysvětluje způsob animace pomocí klíčových snímků a interpolaci mezi snímky. Jádrem práce je pak popis konkrétních technik animací postav. Tato část se věnuje výkladu metod animace pomocí kosterního systému, technice morfing a ragdoll simulaci. Kapitola Použité technologie je zaměřena na knihovny DirectX a PhysX, které jsou využity pro implementaci zvolených metod animací. Dále pak obsahuje popis procesu snímání pohybu známý jako motion capture a popis softwaru iPi Desktop Motion Capture, pomocí kterého byly nahrávány některé animované sekvence pro tuto práci. Kapitola Návrh a implementace se orientuje na realizaci demonstračního programu. Popisuje strukturu vytvořené aplikace a použitou reprezentaci dat modelů, animací a scén. Především se však zabývá konkrétními způsoby implementace uvedených metod animací postav.

Závěr práce zhodnocuje dosažené výsledky a uvádí možná rozšíření popisovaných technik.

Kapitola 2

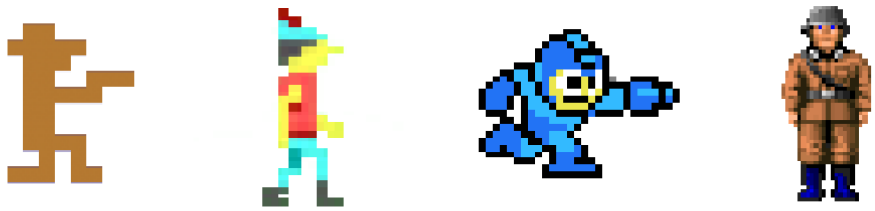
Techniky animací postav

Následující kapitola popisuje významné techniky animací postav v oblasti trojrozměrné počítačové grafiky. Mimo úvodní část kapitoly, která v krátkosti shrnuje historický vývoj animace a reprezentace postav v počítačových hrách, je popis technik zaměřen výhradně na animace modelů reprezentovaných polygonální (trojúhelníkovou) sítí.

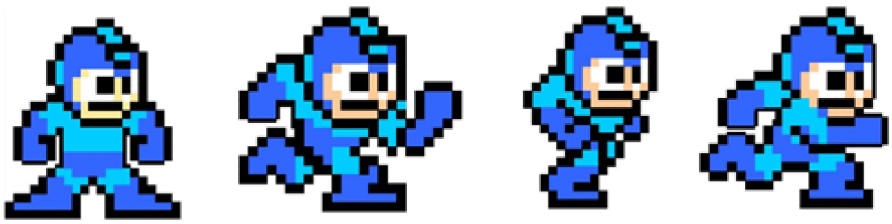
Primárními zdroji pro psaní této kapitoly byly knihy [5], [11] a [15], dále pak články [7], [14] a [13].

2.1 Historie

Principy prvních animačních technik používaných pro vizualizaci pohybu postav ve videohrách úzce souvisí s tradiční kreslenou filmovou animací. Přibližně od druhé poloviny 70. let minulého století se začaly objevovat tendence a možnosti vykreslovat a animovat herní charaktery. Jednotlivé postavy a všechny jejich pózy byly původně reprezentovány sadou několika dvourozměrných rastrových obrázků (později označovaných jako sprite [15]). Postupným promítáním rychle se střídajících snímků jednotlivých póz a posunem v rámci scény byla docílena iluze spojitého pohybu.



Obrázek 2.1: Reprezentace postav na počátku rozvoje videoher



Obrázek 2.2: Příklad sekvence snímků využitých pro animaci pohybu postavy

Způsob vizualizace postav se v období 80. let téměř nezměnil. Zvyšovala se pouze úroveň detailů – počet barev, rozlišení i množství snímků v animovaných sekvencích. Technika kreslených postav animovaných pomocí skupiny statických obrázků přetrvala až do prvních 3D her. Na počátku 90. let byly uvedené principy využity při animaci postav v trojrozměrných hrách Wolfenstein 3D nebo DOOM. Na rozdíl od původního konceptu využití rastrových obrázků ve dvourozměrné grafice byly postavy pro účely zobrazení v prostoru kresleny z osmi úhlů. Podle polohy kamery, pozice postav a jejich aktuálního směru otočení se pak ve scéně zobrazovaly konkrétní snímky odpovídající těmto parametrům. Technika zobrazení 2D obrázku do definované roviny ve 3D scéně se nazývá billboarding a v moderní počítačové grafice se tento koncept stále používá například pro částicové systémy nebo vykreslování vzdálených objektů reprezentovaných obrázkem [15].

Obrázek 2.3: Snímky jedné pózy postavy ze hry DOOM kreslené z osmi stran pro účely animace v prostoru

V roce 1992 byla vydána hra Alone in the Dark, kde se poprvé objevila plně trojrozměrná polygonální reprezentace postav promítaná na předkreslené pozadí. Pohyb takto vytvořených postav byl animován interpolací mezi definovanými pózami. Jedná se o první známé využití interpolované 3D animace [19].

Plně trojrozměrné hry se začaly objevovat především od poloviny 90. let. Postavy byly reprezentovány stále nízko-polygonálními modely a animace pohybu jejich těla byly prováděny technikou zvanou morfing.

Morfing pracuje na principu míchání dvou nebo více variant modelů dohromady na bázi jednotlivých vrcholů. Tato technika tedy funguje pouze v případě, mají-li všechny varianty modelu stejný počet bodů a jsou-li jejich polygony, respektive trojúhelníky, strukturovány stejným způsobem. Přístup na bázi interpolace vrcholů byl v minulosti využíván pro animaci celého těla. V dnešní době se však tento postup z důvodu vysoké paměťové náročnosti při animaci pohybu celého modelu používá téměř výhradně pro animaci tváře (více v kapitole 2.4).

V roce 1998 přišla hra Half-life s implementací skeletálního animačního systému, který řešil většinu problémů výše uvedené techniky morfing [5]. Skeletální animace pracuje na principu vytvoření abstrakce kostry, která manipuluje s modelem dle polohy jednotlivých kostí. Rozdíly moderních technik skeletální animace oproti prvotním implementacím jsou především ve způsobu, jakým se k jednotlivým kosem kosterní struktury připojují vrcholy reprezentující povrch modelu a dále pak v principu výpočtu deformací modelu (více v kapitole 2.3.2).

Techniky morfing (kapitola 2.4) a skeletální animace (kapitola 2.3) jsou dnes společně využívány pro animaci pohybu postav ve většině moderních počítačových hrách a budou popisovány dále.

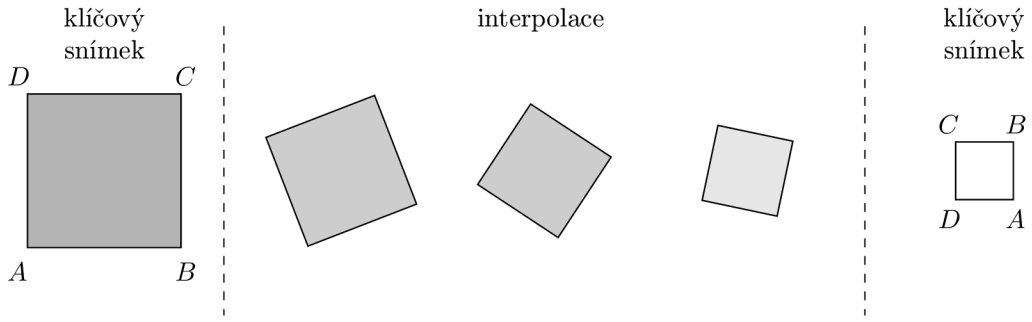
2.2 Animace pomocí klíčových snímků

Animace pomocí klíčových snímků je jednoduchá metoda inspirovaná principem tradiční filmové animace. Postup při tvorbě animovaného filmu klasickým způsobem spočíval ve snímání sekvence nakreslených obrázků reprezentujících pohyb postav v závislosti na čase. Protože byla tvorba celého filmu tímto způsobem velmi zdlouhavá, začal se uplatňovat proces zvaný *keyframing* (klíčování). Práce na animaci filmu byla rozdělena do dvou fází. Hlavní animátoři nakreslili nejdůležitější (klíčové) snímky zachycující pohybově významné polohy postav. Snímky mezi těmito definovanými polohami pak dokreslovali asistenti specializující se na tuto práci [5].

Při animaci postav v oblasti prostorové počítačové grafiky se uplatňuje podobný postup. Animátor ale určí pouze klíčové snímky. Pohyb mezi klíčovými snímky je pak různými způsoby dopočítáván. Animace pomocí klíčivých snímků udává základ metodám popisovaným v dalších kapitolách.

2.2.1 Klíčové snímky

V trojrozměrné počítačové animaci klíčové snímky reprezentují definice významných hodnot parametrů pohybu v určitých diskrétních časových momentech. Těmito parametry mohou být polohy objektů nebo bodů v prostoru, barva, průhlednost, úhel a další. Metoda animace pomocí kosterního systému (dále v kapitole skeletální animace) například využívá klíčové snímky na určení pozic jednotlivých kostí a jejich orientaci v prostoru.



Obrázek 2.4: Příklad sady klíčových snímků určujících polohu vrcholů a barvu čtyřúhelníku s naznačenou interpolací

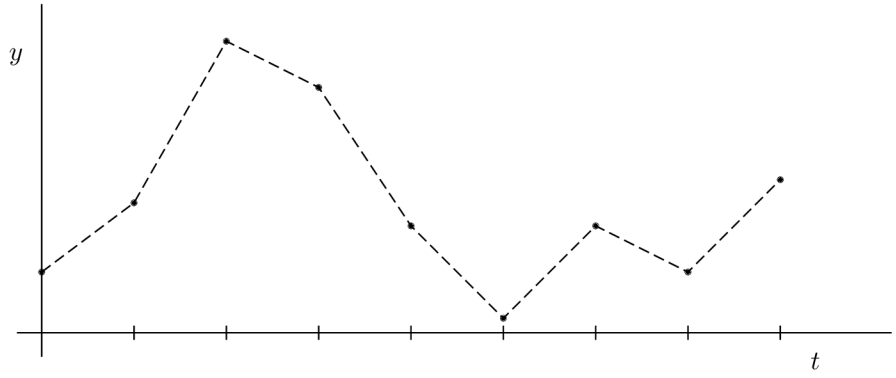
2.2.2 Interpolace

Pro zajištění plynulého přechodu animace mezi jednotlivými klíčovými snímky je nutné vypočítat takzvané *mezisnímky*. Výpočty lze provádět mnoha způsoby interpolace, případně aproximace.

Pro účely animací bude v této práci, z důvodu dostatečné frekvence klíčových snímků v použitých animovaných sekvencích, stačit pouze jednoduchá lineární interpolace. V případě potřeby zpřesnit nebo zjemnit trajektorii pohybu (ať už kvůli nízké frekvenci klíčových snímků nebo například při nutnosti zpomalit čas) je možné použít další varianty interpolace, viz kapitola v knize [11].

Lineární interpolace mezi klíčovými snímky

Klíčové snímky udávají pro každou složku pohybu zvlášť množinu bodů, mezi kterými se provádí interpolace. Tyto body jsou určeny časem snímku a hodnotou konkrétní pohybové složky (například pozicí animovaného objektu na ose x). Lineární interpolace pak spočívá v proložení vždy dvou sousedních bodů z výše uvedené množiny pomocí přímky.



Obrázek 2.5: Příklad lineární interpolace uzlových bodů definovaných klíčovými snímky (osa t představuje čas, osa y hodnoty dané pohybové složky)

Vzorec 2.1 popisuje lineární interpolaci hodnot určených sekvencí klíčových snímků, kde i představuje index snímku, $p_0, \ldots, p_n$ jsou hodnoty parametrů konkrétní složky pohybu v diskrétních časových okamžicích $t_0, \ldots, t_n$; $t_0 < t_1 < \ldots < t_n$.

Při výpočtu výsledné interpolované hodnoty P v určitém čase T je nutné nejprve zjistit mezi kterými snímky se hledaný bod nachází. První část výpočtu lineární interpolace tedy zahrnuje nalezení dvou sousedních snímků i a $i+1$ v časech t_i a t_{i+1}, pro které platí $T \in \langle t_i, t_{i+1})$. V případě animovaných sekvencí používaných v této práci jsou snímky rozmístěny rovnoměrně v čase, platí tedy $h = t_{i+1} - t_i$, kde h udává dobu mezi libovolnými dvěma sousedními snímky. Pro výpočet i lze proto využít vzorec $i = \lfloor (T - t_0)/h \rfloor$.[1] Interpolová hodnota P_T se pak vypočítá podle následujícího vzorce:

$$P_T = p_i + (p_{i+1} - p_i)\left(\frac{T - t_i}{h}\right) \tag{2.1}$$

Uvedený vzorec lze stejně použít i pro data reprezentovaná vektorem (například souřadnice ve trojrozměrném prostoru). V tomto případě by proměnné p_i a p_{i+1} představovaly místo skalárních hodnot právě vektory.

[1] Funkce $\lfloor x \rfloor$ označuje dolní celou část reálného čísla x a udává největší celé číslo menší nebo rovno číslu x.

2.3 Skeletální animace

Skeletální animace patří mezi nejpoužívanější techniky vizualizace pohybu postav v trojrozměrné počítačové grafice. Je typicky používána pro dosažení realistické animace těla, zejména pro vizualizaci změn vzájemného postavení segmentů tělesné pohybové soustavy. Postupy používané při skeletální animaci jsou inspirovány právě principem fungování lidského těla, respektive kostry, odkud také pochází používaná terminologie.

Modely animované touto metodou jsou reprezentovány dvěma základními složkami. Povrch modelu typicky představuje trojúhelníková síť označována jako *mesh*. Jednotlivé vrcholy trojúhelníků v této síti jsou poté jistým způsobem navázány na abstraktní hierarchickou strukturu kostí a kloubů. Tato struktura se nazývá *kostra* (kapitola 2.3.1) a představuje pomocný a oproti realitě velmi zjednodušený pohybový model, který poskytuje pro deformaci povrchové sítě modelu podobnou funkci jako kosterní soustava pro oporu a deformaci těla. Kostra i povrch modelu jsou vždy zadány v základní referenční poloze, čehož se využívá při animaci modelu (viz dále). Koncept animace postav, tedy manipulace s povrchovou sítí modelu pomocí kosterní struktury, zachycuje následující obrázek.

Obrázek 2.6: Princip animace pomocí kosterního systému

Princip skeletální animace je založen pouze na pohybu kostry. Deformaci povrchu modelu pak zajišťují pravidla, která určují způsob, jakým jsou jednotlivé vrcholy ovlivňovány na základě polohy korespondujících kostí. Proces deformace modelu v závislosti na kosterní hierarchii je označován jako *skinning* a bude podrobněji popsán v kapitole 2.3.2.

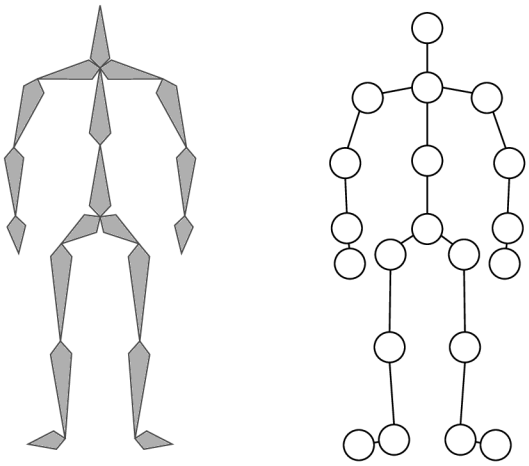
Protože jsou modely animovány pouze pomocí kostry a deformace povrchových sítí jsou dopočítávány, je tato technika na rozdíl od animace metodou morfing (viz kapitola 2.4), paměťově velmi úsporná. Místo pozic všech vrcholů modelu v jednotlivých snímcích stačí v těchto snímcích definovat pouze polohu a orientaci kostí. Výhody skeletální animace také doplňuje možnost akcelerovaného výpočtu deformace modelu na grafickém procesoru (GPU). Nejčastěji používaný a zároveň jeden z nejrychlejších způsobů výpočtu však může vést v některých případech k nepřirozeně vypadajícím výsledkům, což představuje zásadní problém skeletální animace, který je taktéž popsán dále v kapitole 2.3.2.

Přístup k animaci na principu kostry, která manipuluje s povrchem modelu, je velmi komplexní a vedle animace postav může být použit pro animaci pohybu téměř libovolných objektů zachovávajících si určitý tvar a objem (například zvířata, stromy, stroje a další).

2.3.1 Kostra

Kostra představuje abstraktní strukturu, která řídí pohyb a deformaci modelu. Lze ji definovat jako acyklický neorientovaný graf, tedy strom, jehož uzly reprezentují klouby a hrany kosti. Kostra se obecně využívá pouze jako pomocný prvek, který nebude při animaci vykreslen.

Protože se jedná o stromovou strukturu, obsahuje kostra právě jeden kořenový uzel. Pro každý uzel, mimo kořenového, také platí, že existuje právě jeden nadřazený uzel. Tímto způsobem lze definovat komplexní struktury vyhovující různým modelům. Následující obrázek zachycuje příklad kostry s odpovídajícím grafem.



Obrázek 2.7: Schéma kostry pro animaci postavy a příklad odpovídajícího grafu

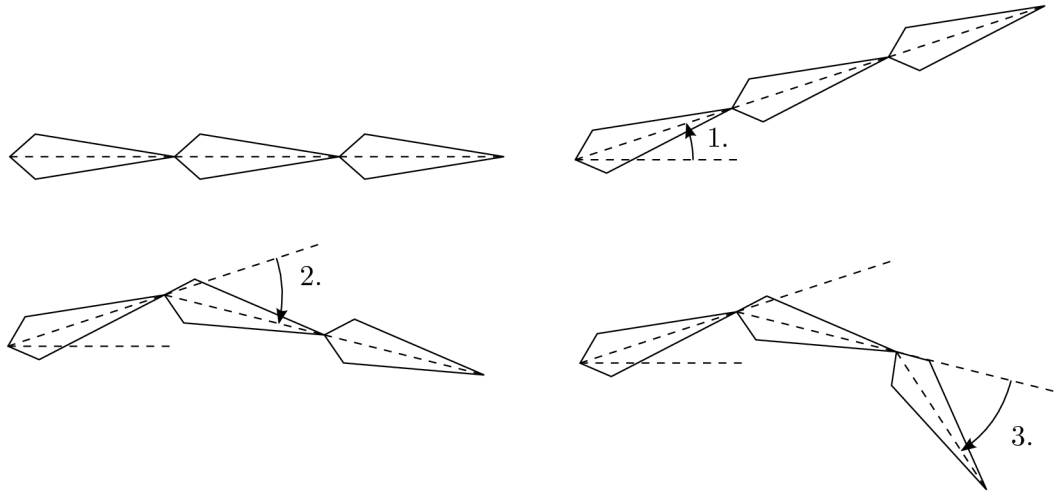
Aby bylo možné kostru, jejíž struktura je popsána stromovým grafem, umístit v prostoru, je nutné určit parametry kostí, respektive kloubů, tedy jejich pozici a orientaci. Tyto informace ponesou uzly v grafu kostry a budou ukládány relativně vzhledem k rodičovským uzlům. Každý kloub je tedy určen polohou v lokálním souřadnicovém systému svého předka a svou orientací, která udává rotaci souřadnicového systému tohoto kloubu (opět relativně vzhledem k předkovi), což představuje takzvanou *lokální transformaci*. Transformace aplikované na konkrétní kost pak ovlivňují nejen tuto kost, ale také všechny potomky – uzly, které v grafu následují. Dochází tak k postupnému skládání transformací označovanému jako *kombinovaná transformace*. Tento postup může být jednoduše ilustrován ve dvourozměrném prostoru na obrázku 2.8, který zachycuje způsob postupné transformace kostí v závislosti na předcích v jednoduché nerozvětvené struktuře.

Budou-li transformace uzlů reprezentovány pomocí matic, lze matematicky popsat ilustrovaný postup vzorcem

$$C_i = L_i P_i, \tag{2.2}$$

kde C_i je matice kombinované transformace i-tého uzlu získaná vynásobením matice lokální transformace tohoto uzlu L_i s maticí P_i, která představuje kombinovanou transformaci předka i-tého uzlu. V případě kořenového uzlu ($i = 0$) platí, že matice P_0 je rovna jednotkové matici.

Takto získané kombinované transformace jsou spolu s lokálními transformacemi jednotlivých kostí potřebné pro výpočty deformace modelu.



Obrázek 2.8: Příklad postupné rotace kostí (translace je pro ilustraci aplikována přímo)

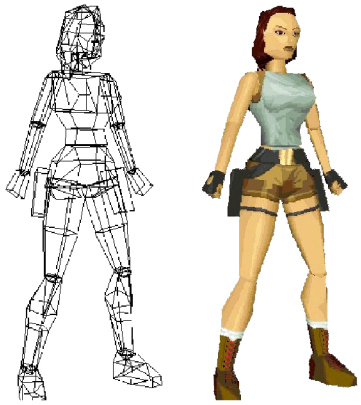
2.3.2 Deformace modelu – skinning

Skinning označuje proces deformace povrchu modelu v závislosti na poloze kostry. Pro každý vrchol jsou definovány vazby na korespondující kosti, které tento vrchol ovlivňují. Způsobů, jak mohou být vazby vrcholů na kosti určeny, existuje několik. V následujícím textu budou popisovány techniky:

- Základní skeletální animace (*rigid body skinning*)
- Míchání vrcholů (*linear blend skinning*)

Základní skeletální animace

Nejzákladnější metoda přiřazuje každý vrchol povrchové sítě modelu právě jedné kosti. Animace těla tímto způsobem produkuje v oblasti kloubů nepřirozené výsledky. V prvních počítačových hrách, které využívaly skeletální animaci, byl však tento přístup z důvodu jednoduchosti výpočtu často využíván. Každá kost ovlivňovala určitý segment, který reprezentoval konkrétní části lidského těla. Následující obrázek zachycuje příklad takto zpracované postavy, ze hry Tomb Raider (1996).



Obrázek 2.9: Postava tvořená pevnými segmenty

Metoda deformace modelu, kdy každý vrchol ovlivňuje pouze jedna kost, udává základ pro další postupy a v moderní počítačové animaci se používá například pro vizualizaci pohybu robotů či strojů, které se skládají z pevných neohebných segmentů. Tento způsob animace bývá v literatuře označován také jako *rigid body* animace.

Aby bylo možné povrch modelu animovat, je nutné určit nejen vazby mezi kostmi a vrcholy, ale také pozice vrcholů vzhledem k těmto kostem. Z tohoto důvodu bývá povrch modelu a kostra zadána v určité *referenční poloze*.

Je-li vrchol v ve své referenční poloze zadán v souřadnicovém systému modelu, pozice vrcholu v_l vzhledem k odpovídající kosti, tedy v lokálním souřadnicovém systému kosti, se poté vypočítá pomocí inverzní kombinované transformační matice C_i^{-1} této kosti v referenční poloze (viz předchozí kapitola).

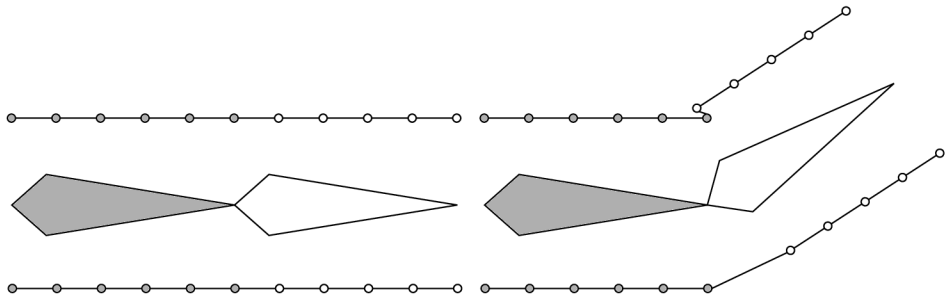
$$v_l = C_i^{-1} v \tag{2.3}$$

Vrchol v_l v lokálním souřadnicovém systému odpovídající kosti je pak možné spolu s touto kostí transformovat, a to vynásobením kombinovanou transformační maticí M_i kosti v cílové animované póze.

$$\begin{aligned} v' &= M_i v_l \\ v' &= M_i C_i^{-1} v \end{aligned} \tag{2.4}$$

Vrchol v' nyní reprezentuje výslednou polohu bodu v souřadnicovém systému modelu dle kombinované transformační matice M_i konkrétní kosti. Tato matice může být určena například klíčovým snímkem animované sekvence (viz následující kapitola).

Za povšimnutí stojí situace, kdy je cílová póza shodná s referenční polohou, tedy kombinované transformační matice M_i a C_i jsou stejné, pak při vynásobení $M_i C_i^{-1}$ vznikne jednotková matice a pozice vrcholu v' tak bude rovna přímo v, což je očekávané.



Obrázek 2.10: Příklad deformace povrchu modelu metodou základní skeletální animace

Obrázek 2.10 ilustruje výše popsanou metodu a problém, který nastává při deformaci souvislého povrchu modelu v oblastech kloubů. Tento problém částečně řeší velmi často používaná technika *vertex blending* (*linear blend skinning*) neboli míchání vrcholů.

Míchání vrcholů

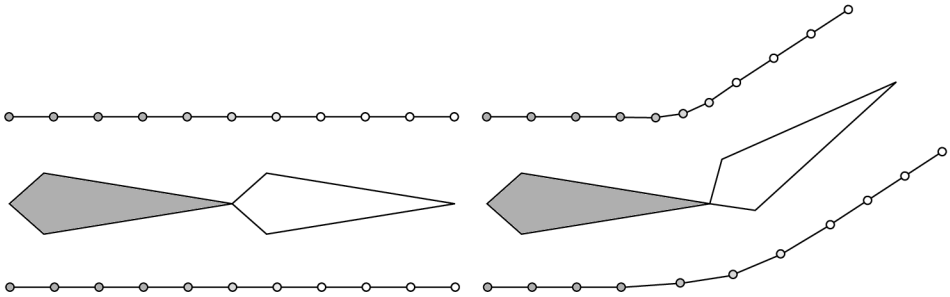
Způsob deformace modelu metodou míchání vrcholů produkuje oproti předchozímu přístupu přirozenější výsledky. Vrcholy jsou, stejně jako v případě základní skeletální animace, svázány s kostmi. Vazba je však popsána obecněji. Vrchol může ovlivňovat jedna či více kostí a míra vlivu konkrétních kostí na transformaci vrcholu je určena váhami [15].

Je-li vrchol v ve své referenční poloze zadán v souřadnicovém systému modelu, cílová pozice vrcholu v' se vypočítá jako konvexní kombinace transformací tohoto vrcholu dle kostí s příslušnými váhami podle rovnice

$$v' = \sum_{i=1}^{n} \left(w_i M_i C_i^{-1} v \right), \tag{2.5}$$

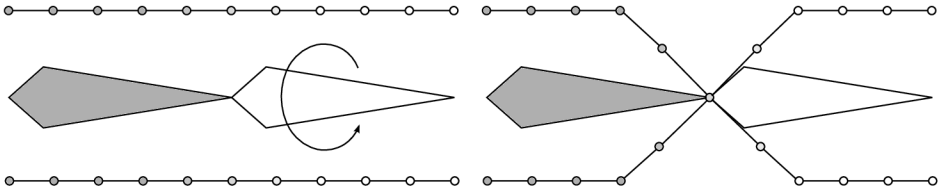
kde n udává počet kostí, které ovlivňují vrchol v a i označuje konkrétní kost. Matice M_i jsou kombinované transformační matice navázaných kostí v animované poloze, C_i^{-1} jsou inverzní kombinované transformační matice navázaných kostí v referenční poloze a w_i představuje váhy jednotlivých kostí, pro které musí platit

$$\sum_{i=1}^{n} w_i = 1, w_i \geq 0 \ \forall i \in \{1, \ldots, n\}\,. \tag{2.6}$$



Obrázek 2.11: Příklad deformace povrchu modelu metodou míchání vrcholů

Obrázek 2.11 ilustruje příklad deformace modelu technikou míchání vrcholů dle výše popsaných rovnic. Oproti základní skeletální animaci probírané v předchozí části kapitoly (obrázek 2.10) produkuje míchání vrcholů výrazně přirozenější výsledky. Tato technika patří pro svoji jednoduchost a rychlost výpočtu (možnost akcelerace výpočtu na grafické kartě) mezi nejčastější způsoby deformace modelů postav v závislosti na kostře při animaci v reálném čase. Přesto však vykazuje míchání vrcholů v určitých stavech značné nedostatky. Obrázek 2.12 zachycuje jev zvaný candy-wrapper artefakt, k němuž při jistých rotacích kostí dochází [15].



Obrázek 2.12: Candy-wrapper artefakt

Tento zásadní nedostatek metody míchání vrcholů se snaží řešit řada dalších pokročilých technik skinningu, například *Dual Quaternion Skinning* popsaný v dokumentu [6], *Spherical Blend Skinning* [1] a další.

2.3.3 Animace, dopředná a zpětná kinematika

V předešlých částech kapitoly Skeletální animace byly uvedeny základní techniky popisující reprezentaci a deformaci modelů postav, které májí být animovány. Tato část kapitoly se bude zabývat přímo jejich animací.

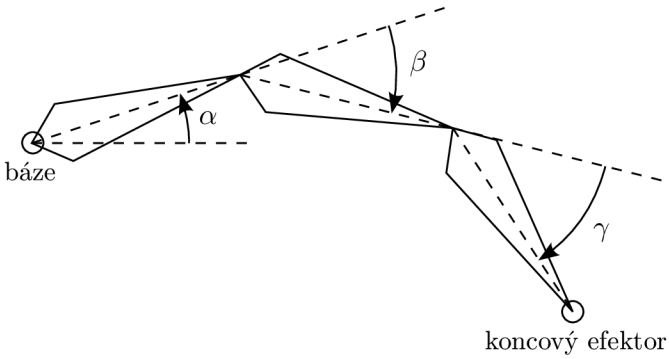
Cílem animace je vhodným způsobem vizualizovat pohyb. Pomocí výše zmíněných technik lze pohyb postav řídit jednoduše pohybem kostry, který může být určen sadou klíčových snímků, viz kapitola 2.2. Tyto snímky definují pózy kostry v určitých časových okamžicích, a to tak, že každý snímek obsahuje informace o transformacích všech kostí.

Je-li animovaná sekvence řízena klíčovými snímky, tedy jsou-li dopředu pevně určeny polohy a rotace jednotlivých kostí (například animátorem nebo nasnímány pomocí motion capture), pak lze kostru jednoduše transformovat od kořenového uzlu až po koncové uzly tak, jak tomu bylo při postupné transformaci popsané v kapitole 2.3.1. Postup animace, kdy je konkrétní poloha kostry transformována přímo podle předem známého předpisu, bývá označován jako *přímá* nebo také *dopředná kinematika*.

Existuje však mnoho situací, kdy je dopředná kinematika nevhodná nebo nepoužitelná. Jako příklad lze uvést stav, kdy je nutné animovat postavu, jež má uchopit do ruky libovolný předmět (kliku od dveří, knihu, sklenici s vodou, ...). Způsob, jakým se postava k předmětu natáhne a jak jej uchopí, závisí na poloze předmětu, poloze postavy, tvaru předmětu a mnoha dalších faktorech. Není proto snadné (někdy možné) takto obecný pohyb předem definovat či nasnímat. Metody, které tento problém řeší, využívají takzvanou *inverzní kinematiku*. Původ inverzní kinematiky spadá do robotiky a jde o postup, kdy se póza kostry (většinou jen její části, například paže), vypočítá dle definovaného cíle. V případě výše uvedeného příkladu jde tedy o postup výpočtu natažení paže pro předmět umístěný na konkrétním místě.

2.3.4 Stavový prostor, stavový vektor

Pro formalizaci výše uvedených přístupů k animaci kostry metodami dopředné a zpětné kinematiky je nutné uvést základní pojmy. Obrázek 2.13 ilustruje segmentovou strukturu se třemi segmenty, které reprezentují část kostry. Tato struktura je ve svém počátku ukotvena, druhý konec je volný. Bod na volném konci bývá označován jako *koncový efektor*, ukotvený bod se nazývá *báze* [15]. Celá struktura představuje *kinematický řetězec*.



Obrázek 2.13: Segmentová struktura se třemi segmenty

Veškeré stavy (pózy), ve kterých se může segmentová struktura nacházet, vytvářejí *stavový prostor*. Konkrétní stav lze poté popsat pomocí *stavového vektoru*. Stavový vektor je v literatuře [15] označován jako

$$\Theta = (\theta_1, \theta_2, \ldots, \theta_n), \tag{2.7}$$

kde θ_i jsou možné parametry segmentů dané struktury. Tímto vektorem je jednoznačně určen koncový efektor. Stav struktury na obrázku 2.13 tedy popisuje stavový vektor, jehož složky představují trojici úhlů α, β, γ.

$$\Theta = (\alpha, \beta, \gamma) \tag{2.8}$$

Délka stavového vektoru stanovuje počet stupňů volnosti, což jsou veličiny jednoznačně určující stav systému.

Dopředná kinematika

Metoda dopředné kinematiky určuje polohu koncového efektoru X transformací f sestavenou na základě známého stavového vektoru Θ [15].

$$X = f(\Theta) \tag{2.9}$$

Výpočet pozice koncového efektoru na obrázku 2.13 tedy spočívá v postupném natočení segmentů dle úhlů α, β, γ.

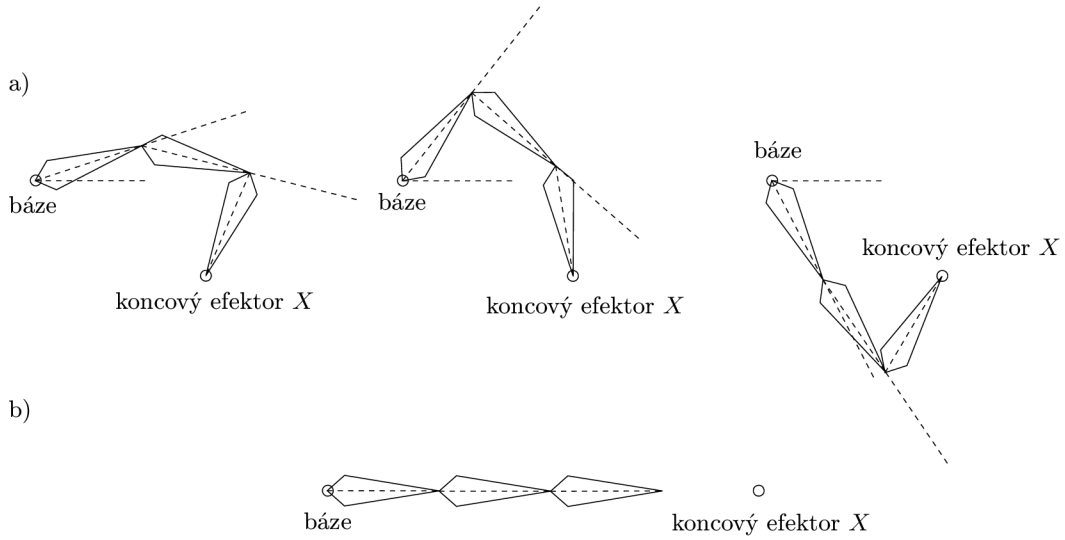
Zpětná kinematika

Zpětná kinematika je metoda, jejíž postup je vůči dopředné kinematice opačný. Cílem je vypočítat jednotlivé složky stavového vektoru Θ na základě znalosti pozice koncového efektoru X. Stavový vektor Θ je určen inverzní funkcí f^{-1} [15].

$$\Theta = f^{-1}(X) \tag{2.10}$$

Inverzní kinematika představuje oproti relativně jednoduchému postupu výpočtu dopředné kinematiky problém spočívající v nelineárnosti funkce f (kvůli rotacím). Analytické řešení inverzní funkce f^{-1} je pro složitější struktury v podstatě nemožné. Inverzní funkce f^{-1} nemusí pro určité polohy koncového efektoru existovat, na druhou stranu je také možné získat více řešení, kdy různé stavové vektory dosáhnou stejnou polohu koncového efektoru, viz obrázek 2.14. Tento problém se obvykle řeší ohraničením možného stavového prostoru, například omezením rotací kloubů v některých směrech nebo omezením rotace dle intervalu v určitém směru.

Metody inverzní kinematiky se v této práci a v demonstrační aplikaci téměř nevyužívají. Podrobně tuto techniku popisuje například práce [4].

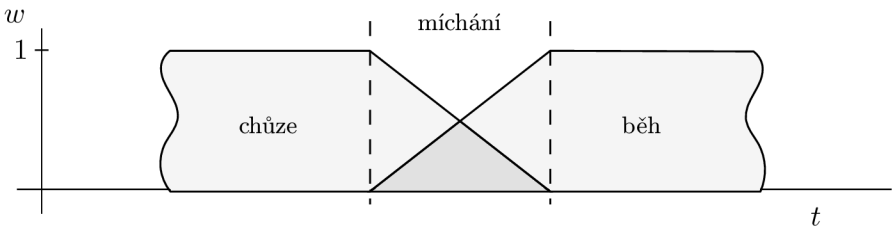


Obrázek 2.14: Inverzní kinematika – a) více řešení pro stejnou polohu koncového efektoru, b) neexistující řešení

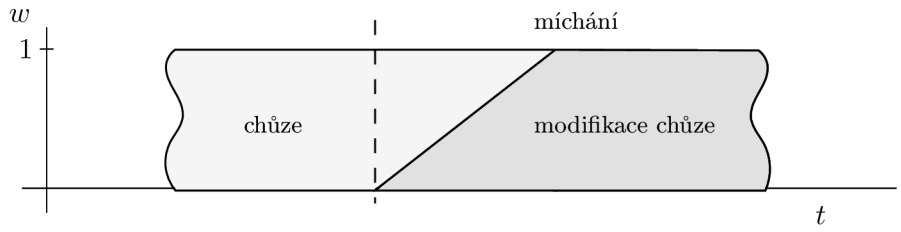
2.3.5 Skládání animací

Poslední technika popisovaná v této čísti kapitoly je skládání animací. Jedná se o jednoduchý postup, který však značně přispívá k flexibilitě skeletální animace. Základní princip spočívá ve skládání dvou či více animovaných sekvencí reprezentujících určité pohyby za účelem vytvořit pohyby nové, a protože lze animace skládat různými způsoby a vytvořit tak různé modifikace existujících pohybů, je tato technika velmi univerzální.

První způsob skládání animací je takzvané míchání. Jde o postup, při kterém se dvě nebo více animací skládají na základě vah, které jim jsou přiřazeny. Míchání animací se využívá především pro vytváření plynulých přechodů mezi určitými pohyby nebo pózami. Jako příklad lze uvést animaci přechodu mezi chůzí a během. Takový přechod je možné zhotovit pouze pomocí animovaných sekvencí chůze a běhu. Změnu mezi těmito pohyby lze určit vahami, kdy v případě přechodu chůze do běhu je postava (kostra) animována napřed plnou váhou sekvence představující chůzi a nulovou vahou běhu. Tyto váhy se postupně převracejí, až je nakonec postava animována pouze pomocí běhu a chůze již nemá na animaci žádný vliv, viz graf na obrázku 2.15. V tomto grafu je součet vah w v diskrétních časech vždy roven 1, není to však nutné. Například mícháním animací, kdy jsou váhy určeny jako na obrázku 2.16, lze docílit animaci představující chůzi kulhajícího člověka. Sekvence chůze může být stále aplikována s váhou 1 a váha sekvence, jež modifikuje chůzi tak, aby postava kulhala, se postupně zvyšuje.



Obrázek 2.15: Příklad váhy při míchání animací přechodu chůze v běh



Obrázek 2.16: Příklad váhy při míchání animací s modifikací chůze

Jsou-li transformace jednotlivých kostí v míchaných sekvencích reprezentovány pomocí matic, lze výsledné transformace v určitém čase t popsat rovnicí

$$F_i(t) = \sum_{j=1}^{n} (w_i(t) T_{ij}(t_j)), \tag{2.11}$$

kde $F_i(t)$ označuje výslednou transformaci pro kost i v čase t, j udává animovanou sekvenci, $w_j(t)$ váhu sekvence j v čase t, $T_{ij}(t_j)$ představuje transformaci kosti i v sekvenci j v čase t_j a n je počet míchaných sekvencí. Čas t_j je možné pro každou animovanou sekvenci zvolit nezávisle.

Druhý způsob, jakým se dají animace skládat, je kombinování dvou nebo více sekvencí. Kombinování animací lze provádět opět různě. Na určité kosti či na celý podstrom v grafu kostry se aplikují transformace jedné sekvence pohybu, na další části kostry se aplikují jiné transformace z jiných sekvencí. Příkladem takto vytvořených pohybů může být složení animované sekvence chůze s aplikací sekvencí různých gest (mávání, ukazování) na paže animované postavy.

Dva výše popsané postupy lze také zobecnit a používat společně, což umožňuje tvorbu složitějších animací na základě existujících a programově generovaných animací. Pravděpodobně největší přínos skládání animací je využití tohoto principu v kombinaci s fyzikální simulací nebo inverzní kinematikou, kdy se transformace z nasnímaných či umělci vytvořených animovaných sekvencí skládají s transformacemi získanými na základě simulací nebo výpočtů inverzní kinematiky. Tímto postupem lze pak dosáhnout komplexní a přirozeně vypadající pohyb postav. Jako příklad může posloužit jízda v autě, kdy se postavy jednak pohybují dle definované sekvence, zároveň však pohyby reagují na situace počítané fyzikální simulací jako je brždění, zatáčení, rozjezd auta a podobně. Animace v kombinaci s inverzní kinematikou zase umožňují například přizpůsobení chůze terénu či schodům. Animace chůze (nebo chůze po schodech) je skládána s transformacemi, které se počítají na základě aktuálního sklonu terénu pod chodidlem, případně dle polohy schodů.

2.4 Morfing

Metoda morfing se v moderní počítačové grafice při animaci postav používá téměř výhradně pro vizualizaci výrazů tváře a projevů řeči. Je založena na principu interpolace mezi dvěma či více variantami konkrétního modelu na bázi jednotlivých vrcholů. Všechny varianty modelu tak musí mít stejný počet vrcholů a jejich polygony musí být strukturovány stejným způsobem.

Z hlediska vývoje jde o postup, kterým byly dříve animovány pohyby celého těla (například hra Quake, 1996). Pohyb byl definován jako klíčovaná animace s několika snímky představující významné pózy, mezi kterými se pak prováděla lineární interpolace. Na rozdíl od skeletální animace musela být, kvůli paměťové náročnosti, frekvence klíčových snímků pro animaci metodou morfing výrazně nižší. Každý snímek obsahoval polohy všech vrcholů modelu v dané póze, což v podstatě znamenalo uložit jeden model v desítkách až stovkách póz (a to pouze pro jednoduché a krátké animace). Pohyby postav pomocí takto vytvořených animací navíc vypadaly, kvůli lineární interpolaci mezi relativně vzdálenými klíčovými snímky, nepřirozeně. Při snaze vytvářet detailnější modely a přirozenější animace, bylo potřeba ukládat stále větší objemy dat s vyšší frekvencí klíčových snímků. Technika klíčované animace metodou morfing se tak stala pro účely animace těla v reálném čase velmi brzy nepoužitelnou a postupně ji nahradila skeletální animace.

Na druhou stranu, animace výrazů tváře a projevů řeči představuje postup mírně odlišný a pro tuto techniku vhodný. Místo sekvencí klíčových snímků se využívají určité cílové varianty modelu, které reprezentují výrazy obličeje. Stejně jako klíčové snímky používané pro metodu morfing, definují tyto tvary pozice všech vrcholů (typicky však pouze pro obličejovou část modelu), sekvenčně na sebe ale nijak nenavazují. Jednotlivé varianty představují konkrétní mimické projevy, například úsměv, pláč, emoční výrazy rozčílení, nadšení, překvapení a další. Mezi tyto projevy se řadí také pohyby úst při vyslovování určitých hlásek. Výsledná animace výrazu tváře nebo řeči je následně tvořena na základě interpolace mezi uvedenými stavy, kdy lze pomocí váhy určit, jak moc mají určité výrazy vliv na obličej a tím definovat, do jaké míry se postava usmívá, mračí a podobně.

2.4.1 Míchání tvarů modelu

Metoda morfing pracuje s referenčním bázovým tvarem modelu, na který se aplikují cílové tvary. Bázový tvar obvykle představuje neutrální výraz. Obrázek 2.17 ilustruje sadu několika výrazů animované postavy.

Míchání tvarů je prováděno lineární interpolací na bázi vrcholů. Výsledný vrchol v' v případě míchání dvou variant konkrétního modelu lze popsat rovnicí

$$v' = v + ((p - v)\, w), \tag{2.12}$$

kde v představuje vrchol v bázovém tvaru modelu, p je odpovídající vrchol v míchaném tvaru modelu a w je váha. Pro interpolaci mezi více variantami modelu lze použít následující rovnici

$$v' = v + \sum_{i=1}^{n} ((p_i - v)\, w_i), \tag{2.13}$$

kde p_i jsou odpovídající vrcholy v jednotlivých míchaných tvarech, w_i představuje váhu těchto tvarů a i zastupuje konkrétní tvar.

Stejně jako vrcholy je většinou nutné míchat i normály, případně další atributy vrcholů (texturové souřadnice, tangenty a podobně).

Obrázek 2.17: Míchání různých výrazů obličeje postavy

2.4.2 Kombinace skeletální animace s metodou morfing

Skeletální animace a morfing představují dva značně odlišné přístupy k animaci postav. Každá metoda se používá pro vizualizaci jiného typu pohybu, a proto je ve výsledku většinou nutné oba přístupy zkombinovat.

Společné využití těchto metod nepředstavuje zásadní problém. Jejich aplikace se provádí postupně ve dvou krocích. V prvním kroku je na model postavy technikou morfing aplikována deformace dle výše popsaných postupů. Typicky jde pouze o změnu v části modelu reprezentující obličej postavy. Modifikují se tak pozice vrcholů, jejich vazby na kosti však zůstanou zachovány. Ve druhém kroku se na takto upravený model následně aplikuje klasická skeletální animace. (Předpokládá se animace modelu, který je definován tak, jak vyžaduje skeletální animace, viz kapitola 2.3.)

Tímto způsobem lze dostáhnout komplexní, přirozeně vypadající vizualizace pohybu postav. V praxi se zmíněný postup pouze mírně liší ve zpracování, kdy se pro každý vrchol modelu provádí oba kroky animace zároveň, není tedy nutné napřed celý model modifikovat technikou morfing a až poté aplikovat skeletální animaci.

2.5 Ragdoll simulace

Ragdoll simulace je metoda procedurální animace těla založená na principech skeletální animace. Tyto principy zůstávají plně zachovány tak, jak byly popsány v kapitole 2.2. Jediný, přesto však zásadní rozdíl představuje zdroj animačních dat. Transformace kostí, tedy jejich polohu a orientaci v prostoru, v tomto případě neurčují klíčové snímky z animovaných sekvencí, ale výsledky fyzikální simulace zjednodušeného modelu těla postavy.

Metoda ragdoll simulace typicky nahrazuje dříve používané statické animované sekvence určitých pohybů v situacích, kdy je vzhledem k prostředí a dalším faktorům obtížné nebo nemožné tyto pohyby předpřipravit tak, aby výsledná animace vypadala přirozeně. Situací, kdy je nutné pohyb postav simulovat, je mnoho. Tato metoda se používá především při animaci pádů a kolizí bezvládného těla s terénem a jinými prvky prostředí nebo při reakcích těla na různé podněty (výbuchy, srážky s předměty a podobně). Ragdoll simulace také udává určitý základ pro pokročilé syntézy pohybu, které se v poslední době začínají při animaci těla postav často využívat.[1]

Samotná problematika fyzikální simulace představuje velmi rozsáhlé téma. Vzhledem k zaměření této práce budou z oblasti fyziky a fyzikální simulace konceptuálně uvedeny pouze základní principy nutné pro popis metody ragdoll simulace a její následnou implementaci pomocí existujících knihoven (v tomto případě PhysX).

Komplexní teoretický i praktický pohled na fyzikální simulaci v reálném čase pro potřeby počítačových her nabízí například knihy [9] a [2]. Z oblasti fyziky budou v této práci využity především principy klasické mechaniky, tedy Newtonovy pohybové zákony, mechanika hmotného bodu, mechanika tuhého tělesa a další související témata, která jsou popsána v uvedených publikacích. Zjednodušeně (právě pouze pro implementaci ragdoll simulace) se touto oblastí fyzikální simulace zabývá také kniha [5].

2.5.1 Fyzikální model postavy

Cílem metody ragdoll simulace je dosažení přirozeně vypadajících animací těla na základě fyzikální simulace. Aby bylo možné simulaci provádět, je nutné vytvořit vhodný simulační (fyzikální) model reprezentující tělo animované postavy. Kvalita tohoto modelu zásadním způsobem ovlivňuje výslednou animaci. Je proto důležité, napodobit podstatné vlastnosti chování lidského těla co nejlépe a zajistit určitou korespondenci simulačního modelu s kostrou a polygonálním modelem animované postavy. Na druhou stranu je, z důvodu náročnosti simulace na výpočetní výkon a nutnosti simulaci provádět v reálném čase nezbytné, model, při zachování uvedených vlastností, co nejvíce zjednodušit.

Způsobů, jakými lze vhodně reprezentovat tělo postav pro účely fyzikální simulace, existuje více. V rámci této práce bude fyzikální model těla představovat *soustava tuhých těles*, viz obrázek 2.18.

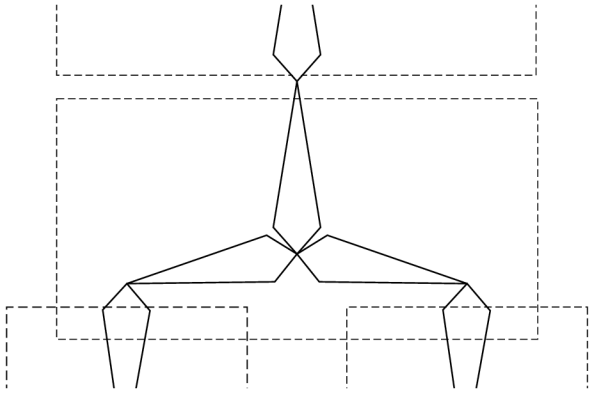
Jednotlivé pohybově významné segmenty těla a kosti, kterými jsou tyto segmenty ovládány, jsou reprezentovány tuhými tělesy. Objem a tvar soustavy těchto těles přibližně odpovídá objemu a tvaru animované postavy. Pro dosažení požadované struktury se tuhá tělesa spojují pomocí *kloubů* (viz kapitola 2.5.3), jejichž pozice odpovídá pozicím kloubů kostry.

Obecně se mezi každými dvěma klouby kostry vytvoří tuhé těleso s těžištěm ve středu spojnice těchto dvou kloubů. Tvar tělesa je zvolen vhodně dle délky kosti, kterou toto těleso zastupuje a dle tvaru konkrétního segmentu animovaného modelu. Orientace tělesa se určí

[1] Například technologie *NaturalMotion* viz www.naturalmotion.com.

Obrázek 2.18: Model postavy a jeho fyzikální reprezentace v podobě soustavy tuhých těles

podle kombinované orientace souřadnicového systému odpovídající kosti, tedy podle orientace, kterou udává kombinovaná transformace nadřazeného kloubu. V oblasti, kde se graf kostry rozvětvuje, bývá skupina kostí nacházejících se v grafu na stejné úrovni nahrazena jedním tuhým tělesem viz obrázek 2.19 (nadřazený kloub, jež udává rotaci souřadnicového systému, ve kterém se dané kosti nacházejí, je pro všechny tyto kosti společný).



Obrázek 2.19: Nahrazení skupiny kostí jedním tuhým tělesem

V uvedených příkladech jsou jako tuhá tělesa zvoleny kvádry. Často se však volí i jiné tvary (válce, kapsle), případně různé kombinace dle možností použité fyzikální simulační knihovny.

Některé pohybově nevýznamné segmenty, například prsty, se obvykle v rámci fyzikálního modelu zanedbávají. Zanedbané segmenty pak nejsou animovány (pohybují se jen na základě nadřazených transformací, viz skládání animací v kapitole 2.3.5 a aplikace fyzikální reprezentace postavy na kostru v kapitole 2.5.4).

Podle výše popsaného postupu lze definovat fyzikální model, pomocí kterého se provádí simulace pohybu postavy. Takový model je možné vytvořit různými způsoby. V rámci této práce jsou použité modely vytvořeny ručně. Vhodným principem pro použití při tvorbě automatizovaného generátoru fyzikálního modelu dle kostry a polygonálního modelu postavy se zabývá například [17].

2.5.2 Tuhá tělesa

Tuhá tělesa představují idealizaci skutečných těles. V případě ragdoll simulace jsou používána jako abstrakce konkrétních segmentů těla především z důvodu jejich relativně nenáročné simulace.

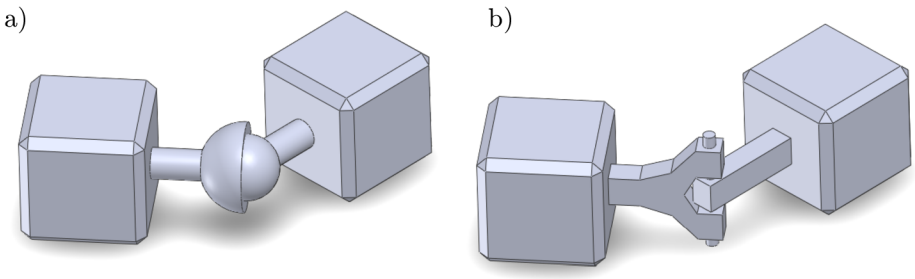
Tuhá tělesa mají tvar, hmotnost a jsou nedeformovatelná. Jsou určena polohou těžiště v prostoru vzhledem k určité vztažné soustavě a dále svou rotací. Rotace bývají v tomto případě reprezentovány pomocí kvaternionů [3]. Pohyb tělesa se tak skládá z otáčivého a posuvného pohybu (rotace a translace). Tyto pohyby jsou přenášeny na odpovídající kosti animovaného modelu, viz dále.

V reálném světě jsou tělesa ovlivňována mnoha silami, jejichž působení se skládá. Při simulaci bývají uvažovány pouze nejvýznamnější působící síly - gravitační, odporové, třecí (v případě kolize tělesa s jiným tělesem) a dále síly vzniklé vazbami tuhých těles na sebe a omezením těchto vazeb.

2.5.3 Klouby

Klouby v kontextu fyzikální simulace definují relativní vazby mezi tuhými tělesy a vlastnosti těchto vazeb. V rámci simulačního modelu jsou klouby umísťovány na pozice, které co nejpřesněji odpovídají pozicím kloubů kostry. Pro dosažení uvěřitelné animace je nutné vlastnosti vazeb nastavit tak, aby omezení pohybu co nejblíže korespondovalo s reálnými možnostmi ohybů konkrétních kloubů lidského těla.

Parametry kloubů, tedy jejich pozice a orientace, se určují relativně vzhledem k připojeným tělesům. Možnosti omezení pohybu nebo rotace a další vlastnosti vazeb závisí na typu použitého kloubu. Obrázek 2.20 konceptuálně ilustruje dvě nejčastěji používané varianty kloubů.



Obrázek 2.20: Varianty kloubů

Použití určitých typů kloubů také závisí na jejich podpoře v simulační knihovně. PhysX například podporuje šest různých kloubů se specifickými vlastnostmi a možností definovat kloubům určité limity [8]:

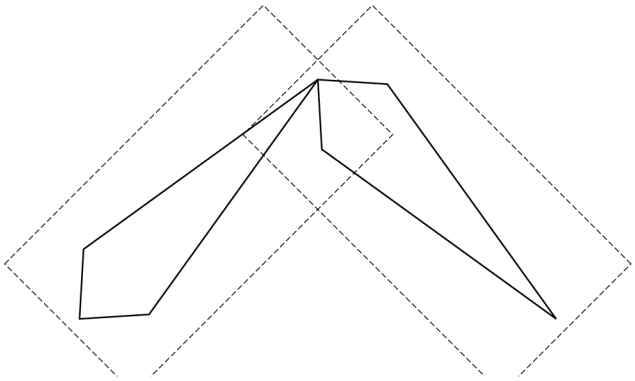
- *fixed joint* napevno sváže dva objekty bez možnosti rotace nebo posunu v rámci vazby
- *distance joint* udržuje dva objekty v konkrétní vzdálenosti bez ohledu na jejich rotaci
- *spherical (ball) joint* umožňuje spojeným objektům rotovat v místě kloubu okolo libovolné osy

- *revolute (hinge) joint* umožňuje volnou rotaci dvou objektů okolo jedné osy v závislosti na definované orientaci kloubu vzhledem k objektům
- *prismatic joint* zabraňuje otáčivému pohybu v rámci vazby, dovoluje pouze vzájemný posun svázaných objektů po jedné ose
- *D6 joint* je plně konfigurovatelný kloub s možností manuálně specifikovat jednotlivé stupně volnosti a omezení

Pro určení vazeb mezi segmenty fyzikálního modelu těla se využívají především kloubní spojení typu *spherical joint*, *revolute joint* a *D6 joint*. Spherical joint (obrázek 2.20a) je kloub vhodný pro simulaci ramenního případně kyčelního kloubu. Umožňuje nastavit limity rotace v jednotlivých směrech a tím napodobit limity ohybů modelovaných kloubů lidského těla. Revolute joint (obrázek 2.20b) představuje vhodnou variantu vazby pro simulaci kolenního a loketního kloubu. D6 joint je univerzální konfigurovatelný kloub, pomocí něhož lze komplexně definovat jednotlivé stupně volnosti a jejich limity. Určitými konfiguracemi kloubu typu D6 je možné zastoupit všechny výše popsané klouby.

Podstatnou vlastností kloubů je možnost definovat *tvrdé* a *měkké* limity. V případě dosažení tvrdého limitu se pohyb nebo rotace v určitém směru jednoduše zastaví, což při simulaci kloubů lidského těla může v některých případech vypadat nepřirozeně. Měkký limit naopak poskytuje přirozenější alternativu, kdy se při dosažení limitu pohyb nezastaví, ale ve směru opačném k dosaženému omezení začne působit specifikovaná síla a tlumení.

Pro správnou funkci simulace je také důležitý fakt, že mezi dvěma tělesy, která jsou přímo spojena pomocí kloubu, nevznikají kolize.



Obrázek 2.21: Ignorování kolizí přímo svázaných těles

2.5.4 Aplikace fyzikální reprezentace postavy na kostru

Posledním krokem k animaci postavy metodou ragdoll simulace je přenos výsledků simulace ze simulovaného fyzikálního modelu na kostru animovaného modelu. Postup aktualizace polohy kostry spočívá v určení pozice kořenového uzlu (kloubu) v grafu kostry a následném natočení všech simulovaných kostí dle orientace odpovídajících tuhých těles. Konečné transformace kostí se spočítají postupnou transformací (popsanou v kapitole 2.3.1) na základě získané polohy kořenového uzlu, původních lokálních pozic všech kostí a orientací kostí získaných při simulaci.

Kapitola 3

Použité technologie

Následující kapitola udává stručný přehled technologií využitých při vývoji demonstrační aplikace, jejíž implementace byla jedním z cílů této práce. V první části kapitoly budou souhrnně popsány knihovny DirectX a PhysX, pomocí nichž byla aplikace implementována. Závěr kapitoly je věnován popisu techniky motion capture, která v dnešní době představuje klíčovou součást tvorby počítačových her i filmů. V rámci této práce byl pro snímání pohybu a tvorbu některých animací využit software iPi Desktop Motion Capture, který bude taktéž popsán na konci této kapitoly.

Primárními zdroji pro následující část textu byly dokumentace k popisovaným knihovnám [8] a [12] a knihy [16] a [5].

3.1 DirectX

DirectX je technologie společnosti Microsoft zahrnující kolekci knihoven, které poskytují aplikační programová rozhraní pro jednotný efektivní přístup k hardwaru. Jednotlivá aplikační rozhraní vytvářejí vrstvu nad ovladači příslušných typů zařízení, například grafických a zvukových karet nebo vstupních zařízení a umožňují s nimi tak pracovat bez znalosti přesných specifikací konkrétních modelů.

DirectX se primárně používá při vývoji videoher pro platformy *Windows* a *Xbox*. Komponenty DirectX jsou rozděleny dle účelu a funkcí. Následující přehled uvádí pouze komponenty použité při implementaci demonstrační aplikace:

- *Direct2D* zajišťuje hardwarově akcelerované vykreslování 2D geometrie, bitmap a textů
- *Direct3D* umožňuje hardwarově akcelerované vykreslování 3D primitiv – trojúhelníků, čar a bodů
- *DirectX Graphics Infrastructure* (*DXGI*) poskytuje správu zařízení – grafických adaptérů (nezávisle na Direct3D)
- *DirectInput* poskytuje rozhrání pro přístup ke stavům vstupních zařízení, jako jsou klávesnice, myši a herní ovladače
- *XNA Math* je matematická knihovna implementující optimalizované operace s maticemi, vektory a kvaterniony
- další komponenty jsou popsány například v knize [16]

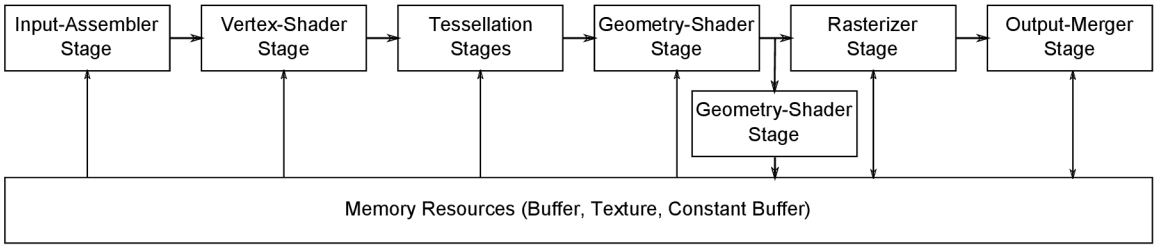
3.1.1 Možnosti využití rozhraní DirectX při animaci

Implementace demonstrační aplikace je postavena na rozhraní DirectX ve verzi 11. Použití této verze přináší (zejména oproti DirectX 9) zásadní změny. V roce 2006 byla uvedena desátá verze DirectX s množstvím významných úprav především v architektuře komponenty Direct3D. Došlo k jejímu výraznému odlehčení a zjednodušení. Podstatným krokem bylo odebrání tzv. *fixed-funcion pipeline*, která představovala pevně danou množinu funkcí a algoritmů zabudovaných do aplikačního rozhraní. Tyto funkce umožňovaly vykreslování 3D geometrie s využitím základních (napevno zabudovaných) efektů a transformací [16].

V rámci změn došlo také k úpravám v knihovně *D3DX* (součást Direct3D). Tato knihovna zahrnovala v DirectX 9, mimo jiné, množství integrovaných podpůrných funkcí a tříd, které umožňovaly načítání modelů a animací, obsahovaly kontroléry skeletální animace a datové struktury pro definici kostry modelu nebo klíčových snímků. Veškerá uvedená funkcionalita týkající se modelů a animací byla od verze DirectX 10 z knihovny D3DX odstraněna. Implementace demonstrační aplikace proto vyžadovala vlastní řešení všech funkcí pro načítání modelů i animací a dále funkcionality samotného řízení animací, tedy implementaci vhodných kontrolérů.

3.1.2 Direct3D, vykreslovací řetězec

Direct3D je jednou z nejdůležitějších komponent kolekce rozhraní DirectX. Umožňuje hardwarově akcelerované vykreslování 3D primitiv, která jsou určena vrcholy. Každý vrchol je popsán souřadnicemi v prostoru, dále normálovým vektorem, souřadnicemi textur a případně dalšími atributy. Pro definici složitějších primitiv jsou vrcholy spojeny vazbami. Vrcholy a jejich vazby pak tvoří obrazová data, která jsou zpracovávána pomocí *vykreslovacího řetězce* [18], viz následující obrázek.



Obrázek 3.1: Zjednodušené schéma vykreslovacího řetězce Direct3D (DirectX 11)

Vykreslovací řetězec zpracovává vstupní data (primitiva určená vrcholy a vazbami) v několika úrovních. Pro účely animací je významná především úroveň *Vertex Shader Stage* poskytující paralelní, plně programovatelné zpracování jednotlivých vrcholů. Tato fáze umožňuje definovat program, tzv. *vertex shader*, provádějící transformace vrcholů a jejich atributů dle postupů popsaných výše v kapitolách věnovaných technikám animací.

Jazyk k tvorbě shaderů (obecně pro libovolnou programovatelnou úroveň zobrazovacího řetězce) je na platformě DirectX označován jako *HLSL*. Implementace popisovaných animačních technik s využitím vertex shaderu je uvedena v kapitolách 4.4.3 a 4.5.2.

Další úrovní vykreslovacího řetězce, kterou se tato práce zabývá, je *Pixel Shader Stage*. Jde o fázi zpracovávající jednotlivé body získané rasterizací z *Rasterizer Stage*. Program, v tomto případě *pixel shader*, umožňuje definovat operace prováděné nad každým bodem výsledného obrazu. Určuje barvu bodů dle textur a výpočtu osvětlení a případně průhlednost.

Pixel shader byl při implementaci aplikace použit například pro výpočty stínování technikou *Illustrative rendering*, viz kapitola 4.

Detailní popis funkcí celého zobrazovacího řetězce a dalších vlastností souvisejících s vykreslováním pomocí Direct3D zasahuje mimo téma této práce. Direct3D a další komponenty rozhraní DirectX jsou popsány například v knize [16].

3.2 PhysX

Technologie PhysX je multiplatformní fyzikální engine společnosti NVIDIA primárně zaměřený na fyzikální simulaci pro potřeby počítačových her. PhysX umožňuje hardwarově akcelerovat určité typy výpočtů na grafických kartách NVIDIA. Binární sestavení této knihovny je volně dostupné a může být použito zdarma pro komerční i nekomerční účely.

Základní komponenty PhysX poskytují podporu pro simulaci tuhých a deformovatelných těles, simulaci kapalin, látek, silových polí a podobně, viz popis [20].

Pro simulaci fyzikálního modelu postavy v rámci animační techniky ragdoll simulace byla v této práci použita verze PhysX 3.2 a podpora simulace tuhých těles zahrnující detekci kolizí pro množství geometrických primitiv (koule, kvádr, plocha, ...) s možností definovat parametry simulovaných objektů a různé typy vazeb – klouby, viz kapitola 2.5.3.

Popis implementace techniky ragdoll simulace spolu s popisem základních principů simulace v knihovně PhysX je uveden v kapitole 4.6.

3.3 Motion Capture

Motion capture označuje proces zahrnující snímání pohybu objektů a následnou rekonstrukci pohybu pro účely tvorby animací. V kontextu zaměření této práce je motion capture technika snímání pohybu těla herců a aplikace nasnímaných dat na kostry modelů virtuálních postav (viz skeletální animace 2.3.1). Tento proces produkuje, oproti ruční animaci pohybu těla, velmi rychle realistické výsledky a představuje proto nedílnou součást tvorby počítačových her i filmů.

Pro snímání pohybu herců existuje řada různých technik (mechanické, magnetické, optické a další). Pravděpodobně nejčastěji se využívá postup optický, tedy snímání scény mnoha kamerami. Optický způsob zachytávání pohybu bývá obvykle doplněn o značky, které zjednodušují rozpoznávání zachyceného obrazu a umožňují dosažení přesnějších výsledků.

Nevýhodou téměř všech technik snímání pohybu jsou především vysoké náklady jak na hardware (speciální zařízení, kamery a další příslušenství), tak na software. Tento problém částečně řeší například software iPi Desktop Motion Capture, který umožňuje snímání pohybu postav pomocí běžně prodávaných zařízení, jejichž cena je oproti profesionálnímu vybavení minimální.

Software iPi Desktop Motion Capture byl pro snímání pohybu postav a tvorbu některých animovaných sekvencí použit i v této práci a jeho stručný popis je uveden dále.

3.3.1 iPi Desktop Motion Capture

iPi Desktop Motion Capture[1] je nástroj poskytující relativně levný a přístupný způsob snímání pohybu postav pro účely tvorby animací. Záznam pohybu je snímán pomocí jednoho nebo dvou senzorů *Kinect*,[2] případně tří až šesti kamer *PlayStation Eye*[3] a dalších zařízení.

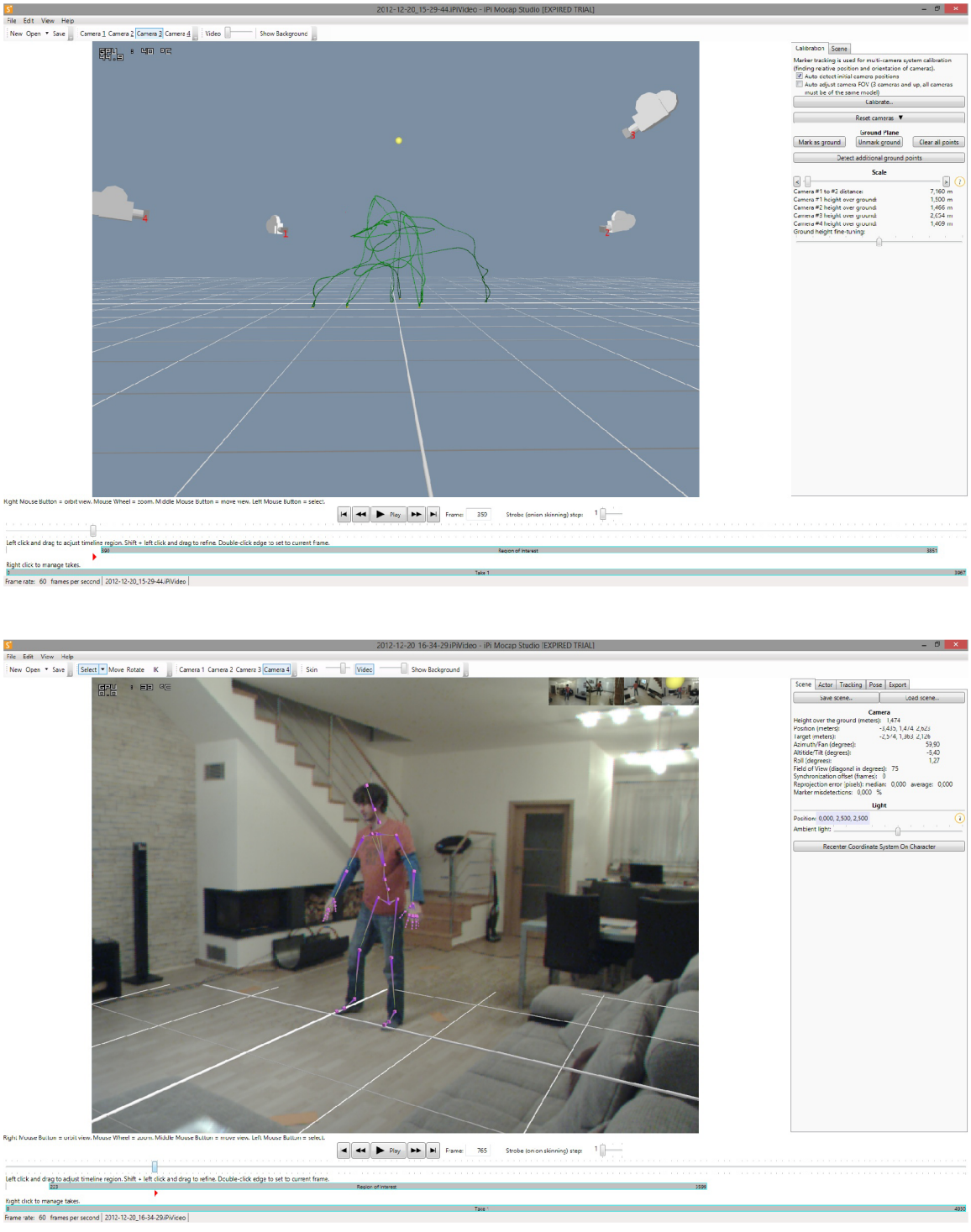
Rekonstrukce nasnímaných dat je pak založena primárně na rozpoznávání obrazu z kamer nebo zpracovávání informací o hloubce zachycených senzory Kinect. Jedná se tedy o optickou variantu motion capture bez použití značek (*markerless*). Pro dosažení kvalitnějšího výstupu, například přesnějšího snímání natočení hlavy nebo rukou herce, je možné využít pohybové senzory konzolí typu *WiiMote*[4] a *PlayStation Move*.[5]

Při experimentování s tímto nástrojem byly využity konfigurace čtyř kamer (v kruhu i v půlkruhu) a dále dva senzory Kinect. Subjektivně nejlepší, až překvapivé, výsledky produkovaly kamery rozmístěné v kruhu.

Klíčovou fází pro dosažení kvalitního výstupu je kalibrace kamer, která je oproti kalibraci senzorů Kinect podstatně složitější. Při snímání pohybu pomocí kamer je pro rekonstrukci pohybu také nutné kalibrovat postavu herce zahrnující definice tělesných proporcí a barvu oblečení, kůže a vlasů. Záleží tedy na vhodně zvoleném oblečení, barevně odlišeném od okolí, i na osvětlení.

Zpracovaná pohybová data lze dále upravovat aplikací několika filtrů pro odstranění nepřesností a chvění v pohybu. Výsledné sekvence je možné vyexportovat do mnoha běžně využívaných formátů. Pohybová data získaná tímto nástrojem byla zpracována a využita v jedné ze scén, které prezentuje demonstrační aplikace. Následující obrázky zachycují fáze zpracování nasnímaných sekvencí.

[1] http://www.ipisoft.com
[2] http://www.microsoft.com/en-us/kinectforwindows/
[3] http://us.playstation.com/ps3/accessories/playstation-eye-camera-ps3.html
[4] http://www.nintendo.com/wii/what-is-wii/#/controls
[5] http://us.playstation.com/ps3/accessories/playstation-move-motion-controller-ps3.html



Obrázek 3.2: Proces kalibrace kamer (nahoře) a *tracking* – rekonstrukce pohybu (dole) při snímání pohybu pomocí iPi Desktop Motion Capture

Kapitola 4

Návrh a implementace

Jedním z cílů této práce bylo vytvoření demonstrační aplikace, která prezentuje výše popsané techniky animací postav. Vývoji a implementaci aplikace, včetně popisu implementace použitých animačních technik, se věnuje následující část textu.

4.1 Koncepce aplikace

Aplikace byla navržena s cílem efektivně demonstrovat použití jednotlivých způsobů animací postav v praxi na konkrétních příkladech. Pro implementaci programu byl zvolen jazyk C++, knihovny DirectX a PhysX a rozhraní Windows API.

Výsledný program vychází ze dvou klíčových konceptů. První koncept je založen na prezentaci animačních technik pomocí statických animovaných scén, které demonstrují vlastnosti a způsoby využití skeletální animace, morfingu, ragdoll simulace a jejich kombinací. Druhý koncept představuje využití těchto technik jako odezvu na interakci uživatele (například uživatelem řízený pohyb postavy). Volbu konkrétních demo scén a ukázek pak v rámci aplikace umožňuje jednoduché uživatelské rozhraní v podobě grafického menu.

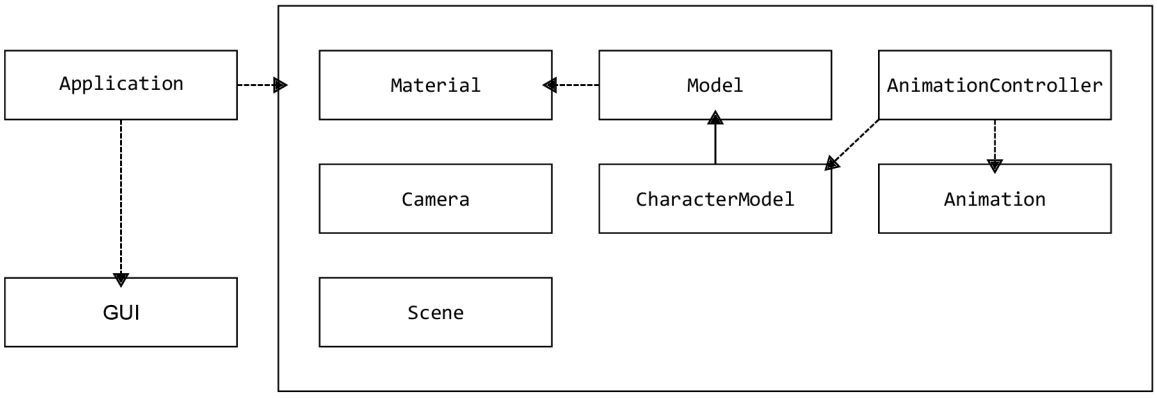
Pro dosažení kvalitního vizuálního dojmu ze scén, který je při animaci podstatný, byly pro účely této práce využity modely a další materiály (textury, některé animované sekvence) pocházející z počítačové hry Team Fortress 2. Tyto materiály jsou majetkem společnosti Valve Corporation a jsou použity se souhlasem právního zástupce této společnosti.

Protože jsou uvedené modely stylizovány, bylo pro dosažení odpovídajícího komiksového vzhledu nutné implementovat Illustrative Rendering popsaný přímo od autorů Team Fortress 2 v dokumentu [10].

Formáty reprezentace použitých dat (modely, animační sekvence, popisy materiálů, trajektorie kamer, popisy vlastností animovaných postav a další) byly navrženy přímo pro účely této práce. Specifikace formátů jsou uvedeny dále v kapitole Reprezentace dat a v přílohách.

4.2 Struktura aplikace

Demonstrační aplikace je členěna do tří hlavních částí. Nejdůležitější prvky její struktury zachycuje diagram na obrázku 4.3. Činnost celého programu řídí třída `Application`. Tato třída implementuje, pomocí skupiny statických metod, veškerou funkcionalitu spojenou s inicializací okna a knihoven DirectX a PhysX, zajišťuje obsluhu událostí generovaných systémem, v separátním vlákně spouští načítání zdrojů (animací, modelů, textur, shaderů, ...) a na základě událostí vyvolaných uživatelským rozhraním řídí vykreslování scén.



Obrázek 4.1: Diagram struktury demonstrační aplikace

Uživatelské rozhraní (GUI) obsahuje prvky pro ovládání aplikace. Vedle klasické funkcionality, založené na šíření událostí mezi komponentami uživatelského rozhraní a vyvolávání reakcí na ně, umožňuje různými průběhovými funkcemi[1] animovat libovolné atributy (pozice, průhlednost, velikost, ...) jednotlivých komponent.

Jádro aplikace je pak soubor tříd, které implementují funkcionalitu animačního systému a vykreslování. Nejdůležitějšími prvky této části jsou:

- Třída `Model`, která poskytuje metody pro načítání a vykreslování modelů. Instance této třídy jsou pak objekty nesoucí informace o geometrii a kostře konkrétních modelů s možností jejich vykreslení na základě přiřazeného materiálu.

- Třída `Material` umožňuje načtení definic různých materiálů, podle nichž jsou vykreslovány modely. V rámci této třídy je implementován jednoduchý statický správce zdrojů, který efektivně řídí načítání, alokaci, inicializaci a dealokaci textur, pixel shaderů a vertex shaderů.

- Třída `CharacterModel`, jež je potomkem bázové třídy `Model`, implementuje funkcionalitu pro načítání a vykreslování modelů postav dle souborů, které definují popis postav (soubor s geometrií modelu, pozice očí, počet částí modelu, materiály pro jednotlivé části modelu, počet výrazů tváře a podobně). Tato třída také umožňuje animaci těla (skeletální animaci) i obličeje (morfing).

- Třída `Animation` poskytuje funkcionalitu pro načítání animovaných sekvencí. Objekty této třídy nesou data konkrétních sekvencí.

- Třída `AnimationController`, jejíž instance představují kontroléry skeletální animace, zahrnuje implementaci řízení animací. Kontroléry řídí animaci konkrétních modelů na základě připojených animovaných sekvencí (instancí třídy `Animation`). Umožňují míchání animací s určením různých průběhů váhy míchaných sekvencí v čase[1] a poskytují možnosti plánování animací a *callback* reakce na ukončení animace naplánovaných sekvencí.

- Třída `Camera` implementuje načítání souborů popisujících trajektorii kamery a umožňuje pohyb kamery animovat na základě načtených záběrů.

[1] *Easing Equations*, viz například http://www.gizma.com/easing/

- Třída `Scene` je abstraktní třídou připravenou pro definici konkrétních scén. Její metody zajišťují inicializaci, aktualizaci a vykreslování scén.

4.3 Reprezentace dat

Demonstrační aplikace vyžaduje pro svůj běh načtení různých dat, například definice materiálů, geometrii modelů, animované sekvence nebo animované záběry kamery a dále pak textury a výrazy pro animaci metodou morfing uložené také v texturách (kapitola 4.5.2). Mimo klasických textur byly všechny použité formáty navrženy přímo pro využití v aplikaci. Došlo tak ke značnému urychlení načítání (využití binárních formátů pro popis animace a geometrie modelů) a k optimalizaci velikosti souborů se zdroji. Následující část kapitoly proto shrnuje použitou reprezentaci dat.

4.3.1 Model

Modely jsou popsány binárními soubory s příponou `.model`. Tyto soubory definují geometrii – počet vrcholů, počet trojúhelníků, jednotlivé vrcholy a seznam trojic vrcholů tvořících trojúhelníky. Pro každý vrchol jsou uloženy pozice a normálové vektory v referenční poloze, dále pak texturové souřadnice a indexy a váhy kostí, které daný vrchol ovlivňují.

Soubory tohoto typu definují také kostru. Kostra je stejně jako geometrie modelu udána v referenční poloze. Je popsána pomocí globálních pozic kloubů a globálních orientací jejich souřadnicových systémů. Orientace jsou uloženy jako jednotkové kvaterniony pouze pomocí komponent x, y a z. Poslední komponenta kvaternionu w je při načítání vypočítána postupem dle příkladu 4.1. Při načítání jsou také pro všechny kosti spočítány matice kombinovaných transformací v referenční poloze, viz kapitoly 2.3.1 a 4.4.1. Struktura kostry je pro jednotlivé uzly (klouby) definována pomocí indexů na nadřazené uzly (index nadřazeného uzlu pro kořen kostry je roven -1).

```
float w = 1.0f - ( x * x ) - ( y * y ) - ( z * z );
w = ( w < 0.0f ) ? 0.0f : - sqrtf( w );
```

Příklad 4.1: Úsek výpočtu komponenty w jednotkového kvaternionu

4.3.2 Materiál

Textové soubory `.material` popisují definice materiálu a způsoby jejich vykreslení. V rámci jednoho souboru je možné definovat více typů materiálů. Každý materiál pak obsahuje informace o pixel a vertex shaderech, které mají být použity při vykreslování a cesty k souborům využitých textur.

4.3.3 Model postavy

Modely postavy jsou popsány pomocí textových souborů `.character`, které určují soubor s geometrií modelu `.model`, soubor s materiálem `.material`, dále definují polohy očí, počet možných animovaných výrazů a ostatní parametry potřebné pro animaci.

4.3.4 Animace

Animace jsou uloženy jako binární soubory typu `.anim`. Tyto soubory udávají počet snímků v animovaných sekvencích, snímkovou frekvenci a strukturu kostry, pro kterou je animace

vytvořena. Jednotlivé snímky zahrnují definice polohy kostry, které jsou uloženy stejným způsobem, jako referenční poloha kostry v souborech `.model`.

4.3.5 Kamera

Animaci kamery, tedy trajektorii kamery, její orientaci, počet snímků v záběru a snímkovou frekvenci udávají soubory typu `.cam`. Orientace kamery je uložena pomocí tří složek jednotkového kvaternionu, je tedy nutné, stejně jako v předchozích případech, dopočítávat poslední komponentu kvaternionu.

4.4 Implementace techniky skeletální animace

Způsob implementace skeletální animace vychází z principů popsaných v kapitole 2.3. Následující text je zaměřen na techniku deformace povrchu modelu metodou míchání vrcholů uvedenou v části 2.3.2. Při implementaci jsou však využity veškeré vlastnosti popsané v kapitole věnované Skeletální animaci.

Vizuální kvalita výsledné animace je výrazně závislá na kvalitě animovaného modelu, návrhu jeho kostry, definici vazeb vrcholů na kosti a použití vhodné referenční pózy. Jak již bylo uvedeno, referenční póza představuje neutrální pózu, dle které se počítá deformace modelu. Model v této póze by proto měl být co nejméně zohýbaný. Důležitost použití vhodné referenční pózy ilustruje obrázek 2.21.

Obrázek 4.2: Příklad vhodné (vlevo nahoře) a nevhodné referenční pózy (vpravo nahoře) a stejné pózy (dole) vypočítané na základě těchto referenčních póz (postava v pravo dole má z důvodu využití nevhodné referenční pózy v oblasti kloubů značné deformace)

Výpočet deformace modelu je prováděn pomocí následujícího vzorce.

$$v' = \sum_{i=1}^{n} \left(w_i M_i C_i^{-1} v \right)$$

Výsledné pozice každého vrcholu v v animovaném stavu v' jsou určitou vahou závislé na transformačních maticích M_i a C_i^{-1} příslušných kostí, viz popis rovnice 2.5, kde M_i reprezentuje kombinované transformační matice kostí v animované poloze a C_i^{-1} jsou inverzní kombinované transformační matice kostí v referenční poloze. Uvedené matice představují pro jednotlivé snímky konstantní parametry. Postup samotné animace lze tedy rozdělit do dvou fází. V první fázi je nutné vypočítat transformační matice zahrnující pro každou kost i transformace $M_i C_i^{-1}$. Tento krok definuje transformaci celé kostry v aktuální animované póze. Druhý krok následně zahrnuje deformaci, tedy výpočet nových pozic vrcholů modelu v závislosti na kostech pomocí transformací získaných při výpočtu v prvním kroku. První krok se provádí pro všechny kosti kosterní struktury modelu, druhý krok poté pro všechny vrcholy povrchové sítě modelu a dále také například pro normálové vektory, případně tangenty.

Protože je druhý krok výrazně složitější (je nutné počítat pozice, normály a další parametry pro tisíce až desetitisíce vrcholů povrchové sítě modelu v každém snímku animace), bývá pro implementaci této fáze animace téměř vždy využit paralelní přístup k výpočtu pomocí vertex shader programu na grafickém akcelerátoru. Tento způsob animace se nazývá *GPU skinning* (*hardware skinning*) a popisem jeho implementace se zabývá kapitola 4.4.3.

Výpočet transformace jednotlivých kostí pro následné zpracování deformací je naopak typicky v režii CPU, viz kapitola 4.4.1.

V některých případech, například při generování stínů metodou stínových těles, je nutné provádět i druhou fázi výpočtu na CPU. Tento způsob animace je označován jako *CPU skinning* (*software skinning*) a je popsán v kapitole 4.4.2. Při implementaci demonstrační aplikace nebyla tato technika použita, je však uvedena pro ilustraci rozdílu mezi jednotlivými způsoby implementace.

4.4.1 Výpočet transformací kostí

Obecný přístup k transformaci kostí metodou postupné (kombinované) transformace byl popisován v rámci kapitoly 2.3.1. Pro výpočet deformací modelu je nutné získat pro každou kost i transformaci zahrnující $M_i C_i^{-1}$ (viz předchozí část textu). Uvedené transformační matice M_i a C_i bývají typicky spočítány při načtení modelu (C_i jako kombinované transformace kostí v referenční póze) nebo animace (M_i jako kombinované transformace kostí konkrétního snímku animované sekvence). Postup výpočtu je při znalosti těchto transformací přímočarý. Pro každou kost v rámci kosterní hierarchie se matice M_i vynásobí s maticí C_i^{-1}. Výstupem první fáze je tedy pole matic, jehož prvky odpovídají finálním transformacím jednotlivých kostí, které budou následně aplikovány při deformaci povrchu modelu. Výstupní pole se v případě implementace CPU skinningu využije přímo, v případě skinningu na GPU se například nastaví jako konstanty vertex shaderu.

V rámci optimalizací je vhodné tuto fázi částečně eliminovat. Jde-li o animaci pomocí statických animovaných sekvencí, jsou výsledné hodnoty pro jednotlivé snímky sekvence vždy konstantní a lze je tedy před počítat a uložit, což přináší jisté urychlení. Při míchání animací, případně při interpolaci mezi dvěma sousedními snímky je však nutné jednotlivé výsledné transformace dále skládat. Skládání i interpolaci lze vypočítat dle vzorce 2.11 a až tento výstup je pak uložen do výsledného pole matic finálních transformací kostí.

4.4.2 CPU skinning

Deformaci povrchu modelu prováděnou pomocí CPU lze popsat více způsoby. Pseudokód v rámci příkladu 4.2 představuje sériově zpracovávanou obdobu paralelního výpočtu deformace, který bude uveden v kapitole GPU skinning 4.4.3.

Vstupními hodnotami pro výpočet jsou již zmíněné matice finálních transformací kostí `Bones` a dále vrcholy `Vertices` jako pole struktur obsahující pozice `.p`, váhy s hodnotami a indexy `.w` a normálové vektory `.n` (orientace normálových vektorů je pro urychlení výpočtu předem přepočítána do lokálního souřadnicového systému odpovídajících kostí). Výstupem jsou transformované vrcholy `TVertices`, které budou použity při vykreslování.

Výpočet probíhá postupně pro všechny vrcholy. V první fázi cyklu se spočítá finální transformační matice `Transform` podle transformací kostí připojených k vrcholu, ve druhé fázi se dle této matice provede transformace vrcholu a normálového vektoru.

```
in: matrix Bones[], struct Vertices[]
out: struct TVertices[]

FOR i = 0 to COUNT( Vertices )
    CLEAR matrix Transform
    FOR each Weight in Vertices[ i ].w
        Transform += Weight.Value * Bones[ Weight.Bone ]
    END FOR
    TVertices[ i ].p = TRANSFORM( Vertices[ i ].p, Transform )
    TVertices[ i ].n = TRANSFORM( Vertices[ i ].n, Transform )
END FOR
```

Příklad 4.2: Pseudokód výpočtu deformace modelu na CPU (CPU skinning)

4.4.3 GPU skinning

Protože je výpočet deformací modelu prováděn pro každý vrchol, je na grafické kartě zpracováván ve vertex shaderu. Při využití tohoto přístupu je proto nutné jako parametry vrcholů posílat ke zpracování do grafické karty vedle normál, texturových souřadnic a dalších typických hodnot také indexy kostí, které ovlivňují daný vrchol a váhy popisující míru ovlivnění vrcholu konkrétní kostí. V tomto případě se stejně jako u ostatních parametrů vrcholů využívají vektory. Kvůli zarovnání a omezení velikosti vektoru na 4 složky bývá často omezen i maximální počet kostí ovlivňujících vrchol na 4. Váhy a indexy kostí jsou tedy uloženy ve dvou čtyřrozměrných vektorech. Ve vertex shaderu může být taková struktura vrcholu definována následovně (ukázka kódu v HLSL, obecně je však přístup identický v libovolném vysokoúrovňovém jazyku pro popis shaderů):

```
struct VS_INPUT
{
    float4 Position : POSITION;         // Pozice vrcholu
    float4 Normal   : NORMAL;           // Normálový vektor
    float4 Weights  : WEIGHTS;          // Váhy kostí
    float4 Indices  : INDICES;          // Indexy kostí
    float2 TexCoord : TEXCOORD;         // Texturové souřadnice
}
```

Příklad 4.3: HLSL definice struktury vrcholu jako vstup vertex shaderu

Každá složka vektoru `Weights` odpovídá váze kosti ovlivňující vrchol, jejíž index je uložen v odpovídající složce vektoru `Indices`. Vstupní vrchol je možné popsat i složitěji a využít například dva čtyřrozměrné vektory pro definici vah a dva pro definici indexů kostí. Výše uvedený způsob implementace ale téměř vždy stačí.

Druhým důležitým vstupem pro výpočet deformace jsou transformační matice (viz 4.4.1). Protože jsou tyto hodnoty pro každý snímek konstantní, bývá pro jejich uložení obvykle využit buffer konstant (případně textura). Matice jsou uloženy v poli. Složky vektoru `Indices` v definici vrcholu udávají indexy do tohoto pole. V jazyce HLSL vypadá popsaný buffer například následovně:

```
cbuffer ConstantBuffer
{
    matrix World;                       // Světová matice
    matrix View;                        // Pohledová matice
    matrix Projection;                  // Projekční matice
    matrix Bones[ LENGTH ];             // Matice kostí
}
```

Příklad 4.4: HLSL příklad definice bufferu konstant

Samotná deformace je poté vypočítána způsobem uvedeným na příkladu 4.5. Opět jde o interpretaci vzorce popsaného v úvodu této podkapitoly. V první části funkce je vypočtena matice zahrnující kombinace transformací všech kostí svázaných s daným vrcholem dle určených vah. Podle této matice se nakonec provede transformace pozic vrcholu a normálového vektoru a výsledky se zapíší na výstup vertex shaderu.

```
VS_OUTPUT VertexShader( VS_INPUT Input )
{
    VS_OUTPUT Output = ( VS_OUTPUT ) 0;
    // Výpočet finální transformační matice
    matrix FinalTransform = BoneWeights.x * Bones[ BoneIndices.x ];
    FinalTransform += BoneWeights.y * Bones[ BoneIndices.y ];
    FinalTransform += BoneWeights.z * Bones[ BoneIndices.z ];
    FinalTransform += BoneWeights.w * Bones[ BoneIndices.w ];
    // Transformace pozice
    Output.Position = mul( Input.Position, FinalTransform );
    Output.Position = mul( Output.Position, World );
    Output.Position = mul( Output.Position, View );
    Output.Position = mul( Output.Position, Projection );
    // Transformace normálového vektoru
    Output.Normal   = mul( Input.Normal, FinalTransform );
    Output.Normal   = mul( Output.Normal, World );
    Output.Normal   = normalize( Output.Normal );
    // Nastavení zbývajících parametrů vrcholu na výstup
    Output.TexCoord = Input.TexCoord;
    return Output;
}
```

Příklad 4.5: HLSL výpočet deformace vrcholu v závislosti na kostech

4.5 Implementace techniky morfing

Následující text konkretizuje teoretický popis animační techniky morfing uvedený v kapitole 2.4 a shrnuje způsoby její implementace.

Metoda morfing produkuje animaci na základě interpolace mezi různými variantami konkrétního modelu. Pro interpolaci se využívá následující vzorec (vzorec 2.13 popsaný v části 2.4).

$$v' = v + \sum_{i=1}^{n} \left((p_i - v)\, w_i \right)$$

Výsledné pozice každého vrcholu v v požadovaném animovaném stavu v' jsou jistou váhou w_i závislé na pozicích odpovídajících bodů p_i v míchaných tvarech modelu. Výpočet výsledného tvaru modelu zahrnuje jak výpočet finálních pozic vrcholů, tak i normálových vektorů, případně texturových souřadnic a dalších atributů (při použití stejného interpolačního vzorce).

Podobně jako v případě skeletální animace lze implementaci této metody provést softwarově (*CPU morfing*, kapitola 4.5.1) nebo akcelerovat hardwarově (*GPU morfing*, kapitola 4.5.2). Metoda CPU morfing je uvedena opět pro ilustraci rozdílů mezi zmíněnými postupy, při implementaci demonstrační aplikace použita nebyla.

4.5.1 CPU morfing

Klasická implementace animační techniky morfing počítána pomocí procesoru je přímou interpretací vzorce popsaného výše. Postup výpočtu uvádí pseudokód v příkladu 4.6.

Vstupem pro výpočet výsledného tvaru modelu `FinalMesh` je bázová neutrální varianta modelu `BaseMesh` nesoucí informace o vrcholech a dalších potřebných atributech (pozice `.p`, normálový vektor `.n`). Bázový tvar modelu je pak ovlivňován vstupními tvary `Targets` dle definovaných vah `Weights` (v uvedeném příkladu jednotlivé prvky pole `Targets`, tedy cílové tvary míchaných variant modelu, odpovídají prvkům v poli `Weights` obsahujícím váhy těchto variant).

```
in: struct BaseMesh, struct Targets[], struct Weights[]
out: struct FinalMesh

FOR i = 0 to COUNT( BaseMesh.Vertices )
    FinalMesh.Vertices[ i ].p = BaseMesh.Vertices[ i ].p
    FinalMesh.Vertices[ i ].n = BaseMesh.Vertices[ i ].n
    FOR j = 0 to COUNT( Targets )
        FinalMesh.Vertices[ i ].p += Weights[ j ] *
        ( BaseMesh.Vertices[ i ].p - Targets[ j ].Vertices[ i ].p )
        FinalMesh.Vertices[ i ].n += Weights[ j ] *
        ( BaseMesh.Vertices[ i ].n - Targets[ j ].Vertices[ i ].n )
    END FOR
END FOR
```

Příklad 4.6: Pseudokód výpočtu techniky morfing na CPU (CPU morfing)

4.5.2 GPU morfing

Základní přístup k implementaci animací metodou GPU morfing je podobný jako v příkladu výše, samotný výpočet je však prováděn paralelně ve vertex shaderu. Stejně jako u metody

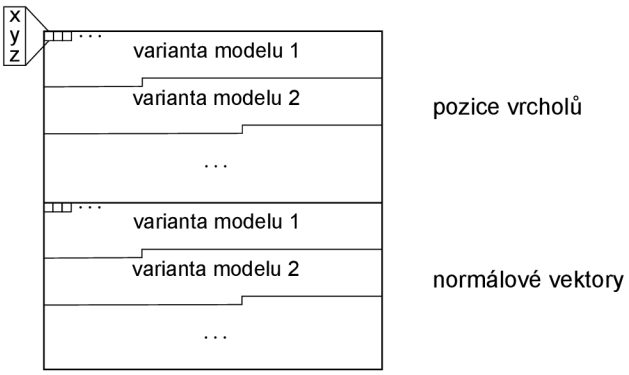
GPU skinning (kapitola 4.4.3) je proto nutné posílat ke zpracování do grafické karty další atributy vrcholů, podle kterých se výpočet provádí.

Nejjednodušší postup je založen na rozšíření struktury popisující vrchol (vstup vertex shaderu) o několik položek, které nesou vedle informace o pozici vrcholu v bázovém modelu také pozice odpovídajících vrcholů v dalších míchaných modelech. Váhy míchaných modelů jsou ukládány do bufferu konstant. Při znalosti těchto hodnot je poté možné provést výpočet konečné pozice vrcholu dle vzorce 2.13.

Uvedený přístup má ale mnoho nevýhod. Základní problém je v nutnosti rozšiřovat jak buffer konstant, tak strukturu vrcholu. Ve struktuře vrcholu je navíc nutné zahrnout pro správný výpočet osvětlení i normálové vektory všech míchaných variant modelů. Z těchto důvodů byla pro implementaci metody morfing zvolena technika jiná, značně flexibilnější.

Použitá technika je založena na odlišném přístupu k získávání vstupních hodnot potřebných pro výpočet. Na rozdíl od předchozí varianty nedošlo (až na drobnou úpravu) k významným zásahům do struktury vrcholu, která popisuje vstup vertex shaderu. Všechna potřebná data – pozice vrcholů a normálové vektory všech tvarů animovaného modelu, vyjma bázového, byla uložena do textury.

Struktura použité textury byla navržena následovně. Pro formát textury byl zvolen typ `DXGI_FORMAT_R32G32B32_FLOAT`, který definuje jeden texel jako trojici komponent typu `float` o velikosti 32bitů. Do takto definovaného texelu je možné uložit třísložkový vektor popisující například pozici vrcholu v prostoru nebo normálu. Textura byla logicky rozdělena na poloviny. První polovina byla určena pro pozice vrcholů, druhá pro normálové vektory. Všechny varianty modelu potřebné pro animaci byly uloženy postupně za sebe (s ohledem na rozdělení polovin), viz následující schéma.



Obrázek 4.3: Schéma struktury textury pro animaci metodou morfing

Výpočet deformace modelu uvedenou technikou je založený na vyhledání konkrétních pozic vrcholů a normálových vektorů v takto připravené textuře. Popsaná technika je aplikována ve vertex shaderu při vykreslování bázového modelu. Pro vyhledání je nutné znát, mimo celkového počtu vrcholů modelu a rozlišení textury i index vrcholu, který se právě zpracovává. Indexaci vrcholu zajišťuje systémově generovaná hodnota. V jazyce HLSL je tato hodnota popsaná sémantikou `SV_VertexID`. Ostatní parametry (váhy, indexy míchaných modelů, velikost textury a počet vrcholů modelu) jsou ukládány v bufferu konstant. Následující příklady ukazují definici struktury vrcholu a způsob načtení hodnoty z textury. Výsledné deformace jsou poté spočítány, stejně jako v předchozích případech, pomocí vzorce 2.13.

```
struct VS_INPUT
{
    float4 Position : POSITION;         // Pozice vrcholu
    float4 Normal   : NORMAL;           // Normálový vektor
    float2 TexCoord : TEXCOORD;         // Texturové souřadnice
    uint   VertexID : SV_VertexID;      // Generovaný index vrcholu
}
```

Příklad 4.7: HLSL příklad definice struktury vrcholu při použití indexace `SV_VertexID`

```
// Pozice v textuře získaná z indexu vrcholu a případného odsazení
uint Sum = VertexID + Align;
// Výpočet souřadnice v textuře
uint XCoord = Sum % TEXTURE_WIDTH;
uint YCoord = Sum / TEXTURE_HEIGHT;
// Načtení hodnoty z textury
float4 Value = Texture.Load( float3( XCoord, YCoord, 0 ) );
```

Příklad 4.8: HLSL příklad získání hodnot z textury

4.6 Implementace ragdoll simulace

Technika ragdoll simulace byla popsána v kapitole 2.5. Jde o procedurální způsob animace těla založený na fyzikální simulaci a skeletální animaci (viz kapitola 2.3). Simulace postavy vyžaduje vytvoření fyzikálního modelu, který reprezentuje animovanou postavu a okolní scénu. Tato část byla implementována pomocí fyzikální simulační knihovny PhysX.

4.6.1 Definice fyzikálního modelu

Prvním krokem pro implementaci ragdoll simulace je definice scény. Scéna je v knihovně PhysX reprezentována objektem `PxScene` a její popis určuje struktura `PxSceneDesc`. Tato struktura definuje klíčové parametry simulovaného prostředí, mimo jiné například směr a velikost gravitační síly. Vytvoření objektu `PxScene` je závislé na objektech `PxPhysics` a `PxFoundation`, které inicializují samotnou knihovnu PhysX a umožňují vytvářet instance objektů PhysX (scény, materiály, tělesa a podobně).

Do vytvořené scény je možné zahrnout simulovatelné objekty (v tomto případě tuhá tělesa). V knihovně PhysX se takové objekty nazývají *Actors* a jejich vlastnosti jsou určeny dvěma základními třídami – `PxRigidStatic` a `PxRigidDynamic`. Dle těchto tříd jsou objekty buď statické, které jsou ve scéně umístěny napevno a nejsou ovlivňovány simulací, nebo dynamické. Dynamické objekty jsou simulovány, případně mohou být jejich umístění a orientace v prostoru nastavovány aplikací za běhu. Pomocí statických a dynamických tuhých těles bylo v aplikaci modelováno prostředí scény a animovaná postava (dle postupu uvedeného v kapitole 2.5.1). Fyzikální model postavy byl vtvořen převážně ručně.

Důležitou částí implementace je vhodné nastavení vazeb mezi tuhými tělesy v rámci modelu postavy. Varianty vazeb (kloubů), které knihovna PhysX podporuje, byly uvedeny již v kapitole 2.5.3. Při vytváření fyzikálního modelu postavy byly použity klouby typu `PxRevoluteJoint`, `PxD6Joint` a `PxSphericalJoint` umožňující určit potřebné definice vazeb (viz kapitola 2.5.3).

Klouby a tuhá tělesa byla sestavena dle referenční pózy animovaného modelu. Pozice kloubů (fyzikálních vazeb) odpovídají pozicím kloubů kostry, orientace a pozice tuhých těles odpovídají simulovaným kostem (pozice po přepočtu – střed tělesa se nachází ve středu spojnice dvou kloubů reprezentujících kost).

Takto definovanou scénu lze pak simulovat a výsledky simulace aplikovat přímo na kostru modelu. V tomto případě jde konkrétně o aplikaci pozice tělesa představujícího kořen kostry na kořenový uzel kosterní struktury a o aplikaci orientace těles na jim odpovídající kosti.

Kapitola 5

Závěr

Cílem této práce bylo prostudovat, popsat a implementovat vybrané moderní techniky animací postav, které se používají v počítačových hrách. Principy vybraných technik spolu s úvodem do historie vývoje animace a reprezentace postav ve videohrách byly uvedeny v rámci kapitoly Techniky animací postav. Tato kapitola byla zaměřena na animaci pomocí klíčových snímků, skeletální animaci, metodu morfing a ragdoll simulaci.

Implementace popsaných metod byla zrealizována v podobě demonstrační aplikace, která na konkrétních scénách ukazuje využití a vlastnosti jednotlivých způsobů animace. Protože je každá technika specificky využívána pro vizualizaci určitého typu pohybu, bylo pro dosažení komplexní animace těla i tváře nutné uvedené metody kombinovat.

Část práce byla věnována technologii motion capture a softwaru iPi Desktop Motion Capture. Tato část představovala velmi zajímavou zkušenost se snímáním pohybu, zpracováváním nasnímaných dat, s přenosem dat do modelovacího softwaru a jejich následnou úpravou do podoby navržené pro reprezentaci dat v demonstrační aplikaci.

Mimo použití knihoven DirectX a PhysX byla veškerá funkcionalita implementována vlastními způsoby řešení, včetně návrhu vlastní reprezentace dat či uživatelského rozhraní.

Jednoznačným přínosem bylo bezesporu využití vizuálně kvalitních a složitých modelů. Pro dosažení požadované vizuální podoby bylo nutné řešit detaily jako jsou pohyby očí, Illustrative Rendering, průhlednost brýlí, efektivní využití velkého počtu výrazů obličeje a podobně. Tato fáze zahrnovala také studium struktury jiných formátů animací a modelů.

Potencionální rozvoj této práce pak vyplývá z textu technické zprávy. V kapitole Skeletální animace v části 2.3.2 byly popsány dvě základní metody deformace povrchu modelu v závislosti na poloze kostry a jejich nedostatky. První směr možného rozvoje je tedy studium a implementace techniky Dual Quaternion Skinning, která byla uvedena jako možné řešení nedostatků popisovaných technik. Další rozvoj práce lze také směřovat k pokročilým technikám inverzní kinematiky, kterým zde nebylo věnováno tolik prostoru jako klasické přímé kinematice.

Literatura

[1] Andersen, K. A.: Spherical Blend Skinning on GPU. `http://image.diku.dk/projects/media/kasper.amstrup.andersen.07.pdf`, [cit. 2013-03-08].

[2] Bourg, D. M.: *Physis for game developers*. Boston: O'Reilly, 2002, iSBN 05-960-0006-5.

[3] Dam, E. B.; adn Martin Lillholm, M. K.: Quaternions, Interpolation and Animation. `http://bit.ly/YmjH5k`, 1998 [cit. 2013-02-15].

[4] Franc, B.: *Animace s pomocí inverzní kinematiky*. Diplomová práce, ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE, 2005 [cit. 2013-04-02].

[5] Granberg, C.: *Character animation with Direct3D*. Boston, Mass: Course Technology/Cengage Learning, 2009, iSBN 15-845-0570-2.

[6] Kavan, L.; Collins, S.; Žára, J.; aj.: Skinning with Dual Quaternions. `http://isg.cs.tcd.ie/kavanl/papers/sdq-tog08.pdf`, [cit. 2013-03-08].

[7] Luna, F.: Skinned Mesh Character Animation with Direct3D 9.0c. `http://mathinfo.univ-reims.fr/image/dxMesh/extra/d3dx_skinnedmesh.pdf`, 2004-02-20 [cit. 2013-01-13].

[8] Microsoft: DirectX SDK Documentation. 2010 [cit. 2013-04-02].

[9] Millington, I.: *Game physics engine development*. Boston: Morgan Kaufmann Publishers, 2007, iSBN 978-012-3694-713.

[10] Mitchell, J.; Francke, M.; Eng, D.: Illustrative Rendering in Team Fortress 2. `http://bit.ly/TMzL0`, 2004 [cit. 2013-04-02].

[11] Msrselas, H.: *Game programming gems*, kapitola Interpolated 3D Keyframe Animation. Hingham: Charles River Media, 2000, s. 465–470, iSBN 15-845-0049-2.

[12] NVIDIA: PhysX SDK Documentation. 2010 [cit. 2013-04-02].

[13] van Oosten, J.: GPU Skinning of MD5 Models in OpenGL and Cg. `http://3dgep.com/?p=1356`, 2011-03-14 [cit. 2013-01-28].

[14] van Oosten, J.: Loading and Animating MD5 Models with OpenGL. `http://3dgep.com/?p=1053`, 2011-03-14 [cit. 2013-01-28].

[15] Žára, J.: *Moderní počítačová grafika*. Brno: Computer Press, 2010, iSBN 80-251-0454-0.

[16] Sherrod, A.; Jones, W.: *Beginning DirectX11 Game Programming*. Boston, Mass: Course Technology/Cengage Learning, 2012, iSBN 1-4354-5895-8.

[17] Verth, J. V.: *Game programming gems 4*, kapitola Using Covariance Matrix for Better-Fitting Bounding Objects. Hingham: Charles River Media, 2004, s. 465–470, iSBN 978-158450295.

[18] WWW stránky: Programming Guide for Direct3D 11: Graphics Pipeline. `http://bit.ly/17HumKF`, 2012-11-28 [cit. 2013-05-01].

[19] WWW stránky: Alone in the Dark. `http://en.wikipedia.org/wiki/Alone_in_the_Dark`, 2013-01-02 [cit. 2013-01-18].

[20] WWW stránky: PhysX. `https://developer.nvidia.com/physx`, 2013 [cit. 2013-05-02].

Příloha A

Obsah DVD

Přiložené DVD obsahuje:

- text bakalářské práce ve formátu PDF
- zdrojové kódy textu práce pro systém LaTeXvčetně obrázků
- dokument detailněji popisující navržené a použité formáty reprezentace dat
- zdrojové kódy demonstrační aplikace, modely, textury, animované sekvence a další materiály
- ukázková videa z demonstrační aplikace a video snímání pohybu pomocí iPi Desktop Motion Capture
- manuály popisující postup překladu demonstrační aplikace, strukturu DVD a obsah jednotlivých adresářů