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

DECIMACE POLYGONÁLNÍCH MODELŮ

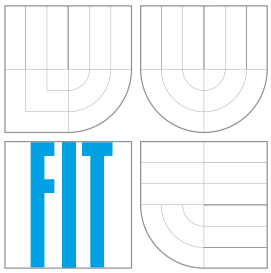
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

DANIELA JOHANNESOVÁ

BRNO 2008

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

DECIMACE POLYGONÁLNÍCH MODELŮ

POLYGONAL MODELS DECIMATION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — DANIELA JOHANNESOVÁ
AUTHOR

VEDOUCÍ PRÁCE — Ing. PŘEMYSL KRŠEK, Ph.D.
SUPERVISOR

BRNO 2008

Licenční smlouva je uvedena v archvním výtisku uloženém v knihovně FIT VUT v Brně.

Abstrakt

Při reprezentaci objektů pomocí polygonálních modelů často vznikají modely obsahující velmi vysoké množství polygonů. Pokud chceme pracovat s takovým modelem v reálném čase, tak je nutné ho zjednodušit. Proces zjednodušování modelu se nazývá decimace. Je známých několik decimačních metod pro zjednodušení modelů. Podle charakteristiky modelu a požadovaných vlastností zjednodušenného modelu chceme vybrat vhodnou metodu, proto je nutné znát vlastnosti jednotlivých metod a také jejich omezení.

Klíčová slova

Decimace, polygonální modely, zjednodušování, zmenšení velikosti, srovnání

Abstract

When representing objects with polygonal models, we must often use models consisting of large number of polygons. If we want to work in real time with such model the model must be simplificated. The process of simplification is called decimation. We know several decimation techniques for simplification of models. According to characteristics of model and target features of simplified model, we want to choose suitable method. So we need to know characteristics of particular methods and also their limits.

Keywords

Decimation, polygonal models, simplification, size reduction, comparison

Citace

Daniela Johannesová: Decimace polygonálních modelů, bakalářská práce, Brno, FIT VUT v Brně, 2008

Decimace polygonálních modelů

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracovala samostatně pod vedením pana Ing. Přemysla Krška, Ph.D. Uvedla jsem všechny literární prameny a publikace, ze kterých jsem čerpala.

.....
Daniela Johannesová
14. května 2008

Poděkování

Chtěla bych poděkovat svému vedoucímu Ing.Přemyslu Krškovi, Ph.D. za poskytnuti odborných rad a informací a také za trpělivost. Také děkuji autorům MDSTk za poskytnutí tohoto nástroje.

Obsah

Kapitola 1

Úvod

V počítačové grafice se setkáváme stále častěji s potřebou reprezentovat objekty reálného světa pomocí polygonálních modelů. V určitých aplikacích se můžeme setkat s modely obsahujícími až několik miliard polygonů. Zdrojem takovýchto rozsáhlých modelů jsou nejčastěji 3D skenery, systémy počítačového vidění, lékařská zobrazovací zařízení a CAD systémy. Ovšem tyto modely je nutné často zobrazovat na běžných osobních počítačích a proto je potřeba modely zjednodušit, před tím než jsou zobrazeny. Díky tomu je možné zjednodušené modely v interaktivních programech rychleji zpracovávat za cenu nižší kvality. Zmenšování počtu polygonů v modelu se nazývá decimace.

Další důvod proč využívat decimaci modelů je v případě, že potřebujeme jeden model v různých rozlišeních. Toho se využívá například v počítačových hrách, kde jsou použity různé velikosti modelu v závislosti na vzdálenosti modelu od pozorovatele. V tomto případě je vždy aktuální velikost modelu volena tak, aby uživatel pokud možno nepoznal zda se jedná o zjednodušený nebo původní model.

Je známo několik obecných typů simplifikačních technik, každá pro určitý typ modelů. Existují metody zaměřující se na zjednodušování křivek, výškových polí, povrchů reprezentovaných polygonální sítí atd. Právě na zjednodušování polygonálních sítí se zaměřím v této práci, jelikož těmto technikám se v posledních letech dostává největší pozornosti.

Existuje relativně velké množství variant algoritmů pro decimaci a u většiny z nich je nutné udělat určitý kompromis mezi rychlostí vytvoření aproximace a její kvalitou. Existují na jedné straně algoritmy, které pracují velmi rychle, ale produkují málo kvalitní aproximace a na druhé straně jsou metody produkující velmi kvalitní aproximace, ovšem jsou nepoužitelné pro velké množství aplikací kvůli časové náročnosti.

V dalším výkladu budeme pro decimaci modelů používat také výrazy zjednodušování redukce, případně simplifikace.

1.1 Rozvržení dokumentu

Tato práce obsahuje 5 kapitol, po této úvodní kapitole následuje kapitola, kde jsou rozebrány některé znalosti o reprezentaci 3D modelů. V této kapitole jsou také teoreticky rozebrány implementované decimační metody a způsoby ohodnocení úspěšnosti decimace. V následující kapitole je popsána implementace decimačních metod a použité nástroje. Čtvrtá kapitola ukazuje vybrané výsledky a porovnává metody. Poslední pátá kapitola uzavírá práci a naznačuje možné další pokračování a rozšíření práce.

Kapitola 2

Rozbor problematiky

2.1 Reprezentace 3D objektů

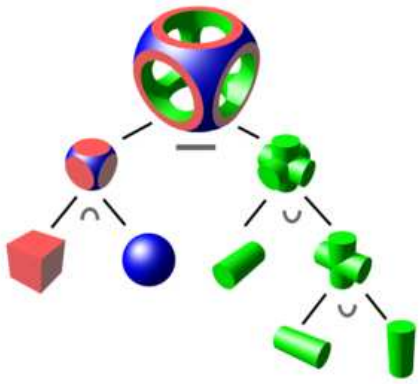
Každá grafická scéna se skládá s trojrozměrných objektů, které mohou být reprezentovány různými metodami. Zároveň je možné na objekty aplikovat různé efekty jako je transformace nebo jiné. Způsob reprezentace modelů je závsilý na oblasti ve které je model vytvořen a používán.

Následující informace byly čerpány z knihy [9] a z elektronických materiálů k předmětu IZG na FIT VUT v Brně [3], z těchto materiálů byly převzaty i některé následující obrázky.

Existuje velké možností popisu modelu ve 3D, já zde popíši tři z těchto možností, přičemž nejvíce se zaměřím na hraniční reprezentaci, která je pro mou práci nejdůležitější.

2.1.1 Konstruktivní geometrie

Nazývané také CSG z anglického Constructive Solid Geometry. Tento způsob reprezentace je používán např. v oblasti CAD. Model je popsán pomocí základních prostorových těles a komplexnější scény jsou tvořeny transformacemi těchto základních těles a operacemi průniku, sjednocení nebo rozdílu. Transformace a operace mezi základními tělesy jsou uloženy ve stromu. Vytváření takového objektu je naznačeno na 2.1



Obrázek 2.1: Příklad modelu reprezentovaného pomocí CSG

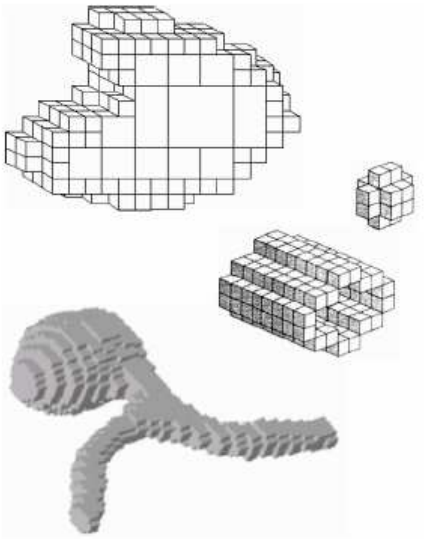
Při každé změně v modelu je třeba celý strom znova regenerovat, proto se pro urychlení

průchodu stromem používá rozdělení modelu pomocí tzv. oktalového stromu. Při použití oktalového stromu je původní strom rozdělen na podstromy a není tedy nutné procházet celý původní strom

Nevýhodou této reprezentace modelu je, že nejsou uloženy informace o povrchu modelu a proto je někdy nutné převést model do hraniční reprezentace 2.1.3.

2.1.2 Dekompoziční modely

Model je reprezentován pomocí základních objemových jednotek, tyto jednotky mají nejčastěji tvar krychle. Jedna elementární jednotka se nazývá voxel. Slovo voxel vzniklo kombinací slov pixel a volume, reprezentuje tedy pixel ve 3D prostoru. Voxely jsou nejčastěji umístěny v pravidelné mřížce a nesou informace o barvě nebo materiálu. Není potřeba uchovávat informace o poloze voxelu jelikož ta je jasná z pozice v mřížce. Jedná se v podstatě o bitmapu ve 3D jak je naznačeno na obrázku 2.2.



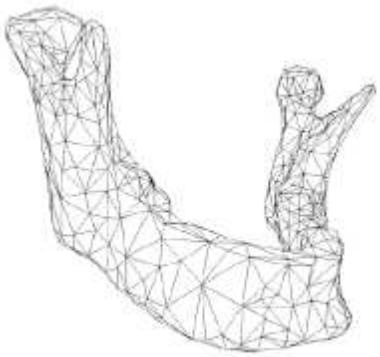
Obrázek 2.2: Model reprezentovaný pomocí voxelů

Dekompoziční modely se používají hlavně v lékařství jako data získaná pomocí CT/MRI technik, případně ve strojírenství . Existují také pokusy o použití této reprezentace v některých počítačových hrách, ovšem toto se příliš neujalo především kvůli vysoké paměťové náročnosti.

Pro snížení paměťové náročnosti je možné využít uložení do oktalového stromu, kdy se opsaný hranol modelu rekurzivně dělí na osm částí. Uložení pomocí oktalového stromu je vhodné především pro malou hustotu dat, jelikož zhoršení přístupového času k datům je vyváženo vysokou úsporou paměti.

2.1.3 Hraniční reprezentace

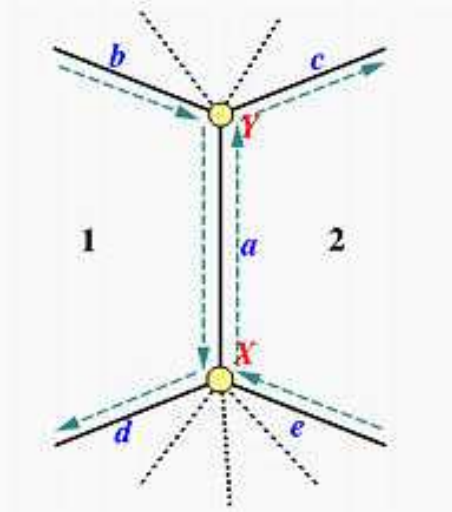
Zkráceně nazýváno také B-rep, což je zkratka z anglického boundary representation. Je to jedna z metod reprezentace modelů pomocí jejich okrajů, tedy modely jsou uloženy jako propojené plošné útvary. Objekty jsou definovány pomocí vrcholů, hran(nejčastěji úsečky,



Obrázek 2.3: Příklad použití hraniční reprezentace

ale obecně to může být i libovolná křivka) a stěn(nejčastěji trojúhelníky, ale je možné použít libovolné tvary polygonů nebo třeba spline plochy).

Reprezentace stěn objektu pomocí trojúhelníku (viz. obrázek 2.3) je většinou nejlepší volba, jelikož vykreslování trojúhelníků je široce podporováno grafickými kartami a je proto nejvýhodnější z hlediska výkonnosti.



Obrázek 2.4: Okřídlená hrana

Při zpracování takto reprezentovaných modelů je důležité efektivní uložení modelu v paměti, které umožňuje procházení modelem (hledání sousedních vrcholů, hran nebo trojúhelníků které používají daný vrchol apod.) a zároveň v paměti nezabírá zbytečně moc místa. Nejznámější uložení je pomocí tzv. okřídlené hrany 2.4.

Při použití okřídlené hrany je použit seznam okřídlených hran, kde každá hrana je reprezentována datovou strukturou obsahující s položky pro:

- koncové vrcholy hrany
- polygony sousedící s hranou

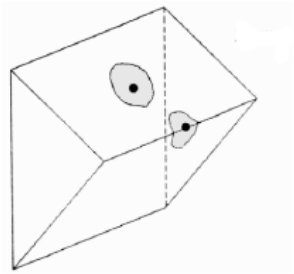
- předcházející a následující hranu na levé straně
- hrany na pravé straně

2.2 Manifold a non-manifold povrchy

Manifold nebo také 2-manifold se nazývá takový povrch, jehož všechny body mají okolí topologicky shodné s diskem. Manifold s hranicí je povrch, jehož všechny body mají buď diskové nebo půl-diskové okolí.

Jednodušeji řečeno, za manifold těleso je považováno takové těleso, kde každá hrana sdílí právě se dvě plochami a jehož hrany neprotínají jiné plochy, zároveň žádný osamocený bod nesmí spojovat dvě části tělesa.

Příklad tělesa, které je 2-manifold je na obrázku 2.5.



Obrázek 2.5: Manifold těleso

Všechny implementované decimační metody umí zpracovávat i non-manifold povrchy, ovšem metoda decimace vrcholů (viz kapitola 2.3.1) nezaručuje vždy dosažení požadované velikosti zjednodušeného modelu, pokud jako vstup dostane non-manifold těleso. Některé metody také mohou samy vyprodukovat non-manifold objekty, jedná se např. o metodu kontrakce hran (viz kapitola 2.3.2).

2.3 Přehled decimačních metod

2.3.1 Decimace vrcholů

Tato metoda byla jako první představena v práci Williama J. Schroedera. Metoda je podrobněji popsána v [11] a [12]. V dalším textu bude pro tuto metodu používán také název Schroederova metoda. Při použití této decimační metody jsou vrcholy polygonálního modelu ohodnoceny na základě důležitosti v modelu.

Vrcholy jsou seřazeny s použitím tohoto ohodnocení a v každém kroku je odstraněn nejméně významný vrchol. Ohodnocení odstraněného vrcholu je distribuováno do okolních bodů a tím se sníží pravděpodobnost že v dalším kroku bude některý z těchto sousedních bodů odstraněn.

Tato metoda je použitelná i pro non-manifold povrchy a povrchy obsahující díry. Ovšem při implementaci metody dle [12] není zaručeno, že metoda dosáhne požadované redukce polygonální sítě, důvod tohoto faktu bude podrobněji popsán dále.

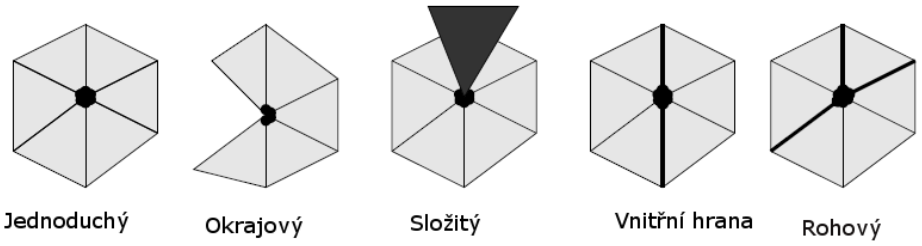
Výsledný model obsahuje podmnožinu bodů původního modelu (poloha bodů se nemění, pouze se vybrané body odstraňují).

Metoda není příliš efektivní a rychlá ve srovnání se zbývajícími dvěma implementovanými metodami.

Ohodnocení vrcholů

Před ohodnocením jsou vrcholy rozděleny do následujících kategorií, tak jak je naznačeno na obrázku 2.6:

- Jednoduché
- Složité
- Okrajové
- Ležící na vnitřní hraně
- Ležící na rohu



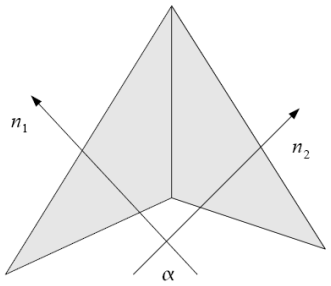
Obrázek 2.6: Různé kategorie vrcholů

Dle [11] je možné rozlišovat ještě další dvě kategorie vrcholů : vrcholy degenerované (u takového vrcholu je některý ze sousedních trojúhelníků degenerovaný) a vrcholy ležící na okraji trhliny.

Pokud trojúhelníky v okolí bodu tvoří souvislý kruh a každá z hran, sbíhajících se v daném je sdílena právě dvěma trojúhelníky, tak je daný vrchol označen jako jednoduchý. Vrchol, jehož okolní trojúhelníky tvoří vějíř, tzn. dvě z hran sbíhajících se do daného bodu mají pouze jeden sousední trojúhelník, je označen jako okrajový. Všechny ostatní vrcholy jsou označeny jako složité.

Jednoduché vrcholy se dále dělí do dalších kategorií na základě počtu tzv. významných hran.

Jako významnou označíme takovou hranu, která vychází ze zkoumaného vrcholu a k níž normály vycházející z jí příslušejících trojúhelníků svírají prostorový úhel α menší než zadaný mezní úhel α_{max}, jak je naznačeno obrázku 2.7.

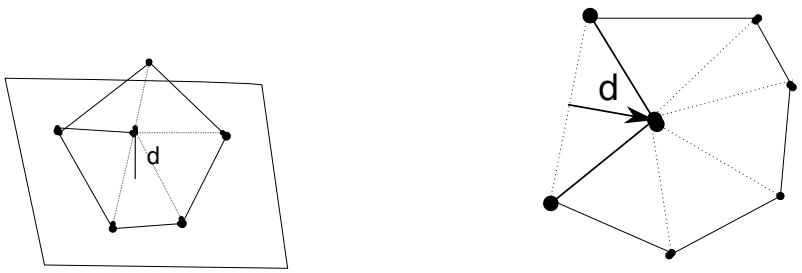


Obrázek 2.7: Určení významné hrany v modelu

Pokud z vrcholu vychází dvě významné hrany, tak je označen jako vrchol ležící na vnitřní hraně. Při dvou nebo více hranách se jedná o rohový vrchol. V ostatních případech zůstane vrchol označen jako jednoduchý.

Způsob ohodnocování závisí na tom zda budou z modelu odstraňovány rohové vrcholy a vrcholy ležící na vnitřní hraně. V případě, že budou tyto vrcholy zachovávány, tak jsou odstraňovány pouze vrcholy označené jako jednoduché a může to vést k tomu, že nebude dosaženo požadované redukce velikosti modelu. Ovšem tato strategie odstraňování vrcholů vede většinou k o něco lepším výsledkům jelikož zachovává více detailů.

Druhá možnost dle [11] je, že budou odstraňovány všechny vrcholy včetně složitých. To vede k o něco horší výsledné aproximaci, ale zároveň umožňuje větší redukci modelu. Při použití tohoto způsobu odstraňování vrcholů je možné dosáhnout téměř libovolné velikosti aproximace původního modelu.



(a) Vzdálenost od roviny

(b) Vzdálenost od hrany

Obrázek 2.8: Způsob ohodnocení vrcholu

Při ohodnocování složitého vrcholu je mu nastavena maximální možná hodnota, ta je nastavena případně i rohovým vrcholům a vrcholům na vnitřní hraně pokud nejsou odstraňovány.

Pro ohodnocení vrcholů jednoduchých a případně také rohových byl použit algoritmus, který každému bodu přiřadí hodnotu dle jeho vzdálenosti od tzv. průměrné roviny, což je rovina proložená body, které jsou v okolí ohodnocovaného vrcholu (viz. obrázek 2.8a).

V případě okrajových vrcholů případně vrcholů na vnitřní hraně je dle [12] pro ohodnocení bodů použita vzdálenost od přímky, tak jak je naznačeno na obrázku 2.8b.

Zbývá dodat způsob výpočtu vzdálenosti od průměrné roviny. K výpočtu jsou použity středy x_i, normály $\vec{n_i}$ a plochy A_i trojúhelníků obklopujících vrchol následujícím způsobem:

$$\vec{N} = \frac{\sum \vec{n_i} A_i}{\sum A_i} \quad \vec{n} = \frac{\vec{N}}{|\vec{N}|} \quad \vec{x} = \frac{\sum \vec{x_i} A_i}{\sum A_i} \tag{2.1}$$

Vzdálenost bodu $\vec{v}$ od roviny je potom určena následujícím vztahem:

$$d = |\vec{n}(\vec{v} - \vec{x})| \tag{2.2}$$

Triangulace

Po odstranění vrcholu a přilehlých trojúhelníků vznikne v modelu díra, kterou je potřeba nahradit nově vytvořenými trojúhelníky. Tento proces se nazývá triangulace. Nové trojúhelníky musí být vytvořeny tak, aby se neprotínaly a aby nebyly degenerované. Po odstranění vrcholu vznikne cyklus okolních hran, který je rekurzivně dělen dokud je počet hran v cyklu větší než tři.

Pro rozdělení cyklu hran je vybrána dvojice nesousedních vrcholů, které budou společně s normálou na průměrnou rovinu původních trojúhelníků tvořit plochu dělící cyklus hran, tzv. dělící rovinu. Tato plocha je použita pro rozhodnutí, zda je daná dvojice vrcholů přípustná pro rozdělení cyklu hran. Výpočet je naznačen na obrázku 2.9.

Obrázek 2.9: Triangulace [12]

Vybraná dvojice bodů rozdělí cyklus hran do dvou menších cyklů a proto aby daná dvojice bodů přípustná, tak musí všechny body prvního cyklu ležet na jedné straně dělící roviny a body druhého cyklu musí ležet na druhé straně dělící roviny. Pokud se nepodaří najít vhodnou dvojici, která by splňovala toto kritérium, tak je odstraňovaný bod přeskočen a pokračuje se dalším bodem.

Samozřejmě v některých případech je možné najít více než jednu dvojici vhodných bodů, které odpovídají uvedenému kritériu. V takovém případě je nutné vybrat nejvhodnější dvojici bodů. Nejvhodnější dvojice je vybrána na základě poměru vzdálenosti bodů cyklů od aktuálně testované dělící roviny.

2.3.2 Kontrakce hran

Při použití této metody zjednodušování modelu spočívá v odstraňování celých hran. Budu se věnovat variantě této metody, kterou představil M. Garland v [2] nebo podrobněji v [6] a která zobecňuje metodu decimace hran a rozšiřuje ji o možnost odstraňovat libovolné dvojice vrcholů, nejen ty dvojice, které jsou spojeny hranou. V dalším výkladu bude tato metoda nazývána také přímo jako Garlandova metoda.

Tato metoda může někdy měnit topologii modelu, může tedy dojít k odstranění děr v modelu, nebo spojení oddělených částí modelu. Z těchto důvodu není tato metoda příliš vhodná například pro aplikace v medicíně, kde je kladen důraz na přesnost a zachování topologie a je lépe využitelná například v počítačových hrách. Pokud tuto metodu modifikujeme tak, aby odstraňovala pouze dvojice vrcholů spojené hranou, tak nedochází ke změnám topologie, ovšem v případě že se jedná o objekt s dírami nebo o model složený z několika oddělených částí tak metoda nemusí dosáhnout požadovaného zjednodušení modelu.

Tato metoda produkuje velmi kvalitní aproximace původního modelu a pracuje i relativně rychle, její rychlost je srovnatelná s metodou decimace vrcholů.

Při výpočtech nejsou brány v úvahu všechny dvojice bodů v modelu, ale tyto dvojice jsou určeny při inicializaci. Výpočty tedy provádíme s dvojicemi bodů, které tvoří hrany a dvojicemi, kde vzdálenost bodů je menší než určitá hraniční d_{max}.

Ohodnoceni hran

Před tím než jsou ohodnoceny a seřazeny hrany je nutné ohodnotit vrcholy v modelu. Každý vrchol je ohodnocen pomocí tzv. kvadratické matice, což je matice o rozměrech 4×4 . Ovšem pro uložení matice stačí pouze deset čísel v plovoucí řádové čárce, jelikož jak bude ukázáno v podkapitole 2.3.2 matice jsou symetrické dle své hlavní diagonály.

Po odstranění hrany je nově ohodnocený vrchol ohodnocený kvadratickou maticí Q, která vznikne jako součet matic Q_1 a Q_2 původních dvou bodů.

Chyba která vznikne při nahrazení bodů v_1 (matice Q_1) a v_2 (matice Q_2) bodem v(ohodnocený maticí Q) je:

$$\Delta(v) = v^T Q v \ = v^T (Q_1 + Q_2) v \tag{2.3}$$

Pokud jsou nahrazovány body v_1 a v_2 bodem v, tak existuje několik možností umístění nového bodu v:

- Do bodu v_1 nebo bodu v_2
- Do geometrického středu těchto bodů, tedy do bodu $\frac{v_1+v_2}{2}$
- Umístit bod tak, aby se minimalizovala výsledná chyba $\Delta(v)$

Umístění nového bodu do místa s minimální chybou samozřejmě produkuje nejkvalitnější aproximace, ovšem zároveň je také výpočetně nejnáročnější. Ideální výsledná poloha bodu se vypočítá na základě následujícího vztahu:

$$\vec{v} = \begin{bmatrix} q11 & q12 & q13 & q14 \\ q21 & q22 & q23 & q24 \\ q31 & q32 & q33 & q34 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = Q^{-1} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{2.4}$$

kde Q reprezentuje matici ohodnocující výsledný vrchol $\vec{v}$.

Výsledné umístění nového bodu se vypočítá pro každou dvojici bodů předem a chyba která vznikne nahrazením dané dvojice bodů novým bodem představuje ohodnocení dané hrany. Odstraňují se hrany s nejmenším ohodnocením, jejichž odstraněním vznikne nejmenší chyba.

Inicializace

Zbývá doplnit způsob výpočtu chyby při počáteční inicializaci. Počáteční chyba je vyjádřena pomocí rovin trojúhelníků, které procházejí ohodnocovaným vrcholem. Každá rovina je reprezentována obecnou rovnicí $a * x + b * y + c * z + d = 0$, kterou je možné vyjádřit také vektorem: $r = \begin{bmatrix} a & b & c & d \end{bmatrix}^T$,

Kvadratická matice bodu v se vypočítá následujícím způsobem:

$$Q = \sum_{r \in roviny(v)} K_p \tag{2.5}$$

kde K_p je následující matice:

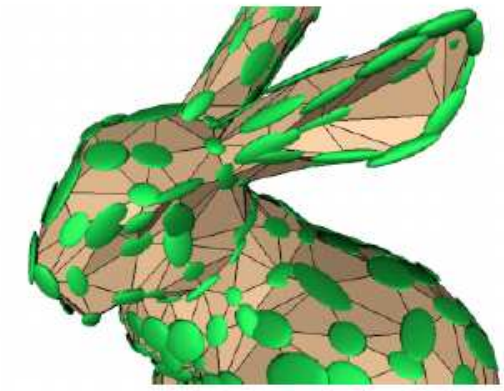
$$K_p = pp^T \begin{bmatrix} a2 & ab & ac & ad \\ ab & b2 & bc & bd \\ ac & bc & c2 & cd \\ ad & bd & cd & d2 \end{bmatrix} \tag{2.6}$$

Zde je jasně vidět proč pro uložení matice stačí pouze deset hodnot, jelikož sčítané matice jsou symetrické dle své diagonály.

Geometrická interpretace

Kvadratická matice v každém bodě určuje rovnici plochy. Body této plochy mají vůči ohodnocenému bodu chybu ϵ a jsou reprezentovány rovnicí : $\Delta(v) = \epsilon$. Plocha vyjádřená touto rovnicí představuje kvadratickou plochu, nejčastěji je touto plochou elipsoid, v některých případech může být tento elipsoid degenerovaný. Příklady elipsoidů příslušející k jednotlivým vrcholům jsou vykresleny na obrázku 2.10, kde je příklad zjednodušeného modelu králíka. Na tomto obrázku je přesněji vidět jak kvadratická matice vyjadřuje geometrickou chybu (obrázek převzat z [2]).

Pokud se podíváme na tvar elipsoidů např. na uších králíka, tak je vidět že jsou hodně zploštělé, což znamená, že nejmenší chyba je v tom směru, ve kterém je kvadrika zploštělá a největší chyba je ve směru kolmém. Naproti tomu elipsoidy např. na zádech mají tvar více kulový, což znamená, že geometrická chyba ve všech směrech je téměř stejná.



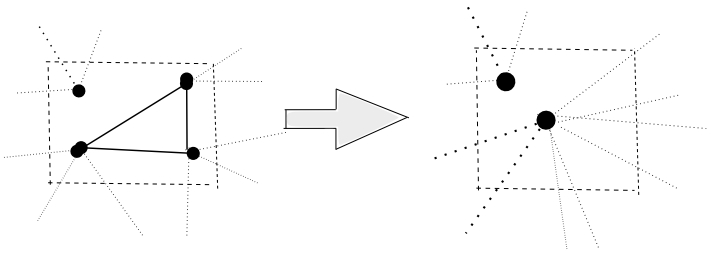
Obrázek 2.10: Okolo každého vrcholu je vykreslen elipsoid vyjadřující chybu

2.3.3 Vertex clustering

Tato metoda byla poprvé představena v [10]. Jedná se o velice rychlou, ale ne příliš přesnou metodu a při použití této metody není možné přesně určit velikost výsledného modelu.

Tato metoda používá shlukování (angl. clustering) více vrcholů do jednoho. Prostor modelu je rozdělen pomocí pravidelné mřížky na menší oblasti. Tyto oblasti budeme nazývat také clustery. Všechny body, které padnou do stejné oblasti jsou nahrazeny jedním bodem. Velikost buněk mřížky určuje přesnost výsledného modelu, čím menší budou buňky v mřížce tím přesnější bude výsledný model.

Jelikož se jedná o velmi efektivní algoritmus ovšem zároveň o algoritmus neposkytující příliš přesné výsledky, tak bylo navrženo několik dalších variant, které se snaží vylepšit výsledky získané touto metodou. V [4] nejsou clustery umístěny v pravidelné mřížce, ale jsou rozmístěny v závislosti na topologii modelu a mohou se tudíž i překrývat. Další možnosti vylepšení algoritmu jsou navrženy v [5], zde je pro dělení modelu použit oktalový strom a v každém clusteru jsou odstraňovány pouze vrcholy spojené hranou ležící v dané oblasti, jak je naznačeno na obrázku 2.11.



Obrázek 2.11: Výběr vhodných hran pro odstranění

Ohodnoceni vrcholů

U této metody není nezbytné ohodnocovat vrcholy, při výběru vrcholu, který nahradí ostatní body v dané buňce je možné použít např. geometrický střed buňky, případně jeden z původních vrcholů, který je nejblíže středu buňky.

Pokud je použito ohodnocení vrcholů, tak se využije pro ohodnocení kombinace dvou faktorů. První je pravděpodobnost že daný bod tvoří obrys modelu, druhá je maximální velikost polygonu sousedícího s ohodnocovaným vrchol.

Dle [4] je první faktor určen pomocí maximálního úhlu mezi dvěma hranami, protínající se v danému vrcholu. Pokud je tento úhel α tak pravděpodobnost, že úhel leží na obrysu modelu je určena jako $\cos(\frac{\alpha}{2})$, druhý faktor může být určen jako maximální délka hrany náležející k danému vrcholu.

Eliminace vrcholů

Pro každý cluster je vybrán jeden vrchol, který nahradí všechny odstraňované vrcholy náležící do této oblasti. Je vybrán buď vrchol s největším ohodnocením v rámci clusteru nebo vrchol, který je nejblíže středu clusteru.

Dále jsou vrcholy původních trojúhelníků nahrazeny nově určenými vrcholy, podle toho do které oblasti patřily původní vrcholy trojúhelníků. Některé trojúhelníky se tímto stanou

duplikátní jelikož jejich vrcholy původně s různými souřadnicemi jsou nahrazeny jedním vrcholem dle příslušnosti k dané oblasti. Duplikátní trojúhelníky je třeba odstranit. Je třeba odstranit také trojúhelníky které se staly degenerované, tedy takové trojúhelníky kde jsou dva případně všechny tři vrcholy shodné.

Jinou možností jakým způsobe provádění decimace naznačený na obrázku 2.11. V tomto případě jsou postupně procházeny hrany v modelu a pokud koncové body hrany leží ve stejném clusteru, tak je daná hrana odstraněna a s ní i náležející trojúhelníky a hrana je nahrazena vybraným bodem reprezentujícím cluster.

Při odstraňování vrcholů je třeba kontrolovat zda nedochází k degeneraci polygonální sítě. Jednou z možností je kontrolovat zda nahrazování původního bodu novými body nedochází k převrácení původního trojúhelníku. Toto je možné testovat pomocí skalárního součinu normály původního trojúhelníku ($n_{puvodni}$) a normály trojúhelníku, kde byly nahrazeny původní body (n_{nove}). Pokud je skalární součin těchto normál menší než nula , tak je daný trojúhelník ponechán s původními vrcholy. Toto kritérium samozřejmě nemůže předcházet naprosto všem degeneracím trojúhelníků, ovšem dává poměrně dobré výsledky a zároveň je efektivní.

2.4 Ohodnocení zjednodušené polygonální sítě

Je velmi důležité ohodnotit zjednodušenou polygonální síť, jelikož bez tohoto ohodnocení by bylo nemožné určit, který ze zjednodušujících algoritmů byl úspěšnější. Pokud je původní model označen jako M a aproximace jako $M^{'}$, tak je potřeba najít funkci $E(M, M^{'})$ jejíž hodnota pro dané dva modely určuje chybu aproximace.

Možnosti vyhodnocení podobnosti aproximace vzhledem k původní sítí jsou poměrně podrobně popsány v [6] odkud jsem také čerpala následující informace.

2.4.1 Podobnost vzhledu

Jako první je možné ohodnotit aproximaci dle podobnosti vzhledu s původním modelem. Toto ohodnocení se může jevit jako nejlepší, jelikož polygonální modely jsou nejčastěji zjednodušovány, aby mohly být vykresleny.

Vzhled modelu M vzhledem podmínkám pozorování ξ je určen rastrovým obrázkem I^{ξ}, který se vykreslí při zobrazení. Dva modely M_1 a M_2 jsou identické z pohledu ξ, pokud jsou jejich rastrové obrázky I_1^{ξ} a I_2^{ξ} identické.

Rozdíl mezi dvěma rastrovými obrázky I_1 a I_2, oba o rozměrech $m \times m$, je možné definovat následovně:

$$\| I_1 - I_2 \|_{img} = \frac{1}{m^2} \sum_u \sum_v \| I_1(u,v) - I_2(u,v) \|^2 \tag{2.7}$$

kde je $\| I_1(u,v) - I_2(u,v) \|$ Euklidovská vzdálenost mezi dvěma RGB pixely. Pokud M_2 bude dobrou aproximací modelu M_1 tak bude hodnota $\| I_1 - I_2 \|_{img}$ malá. Celkový rozdíl mezi dvěma modely je možné získat pokud integrujeme $\| I_1^{\xi} - I_2^{\xi} \|_{img}$ pro všechna ξ. V praxi se samozřejmě použijí vzorky pro určitý konečný počet hodnot ξ.

Toto ohodnocení aproximace je výhodné především z toho důvodu, že ohodnocuje přímo vzhled objektu, což nás při vykreslování zajímá nejvíce. Má ovšem také jednu podstatnou nevýhodu, kvůli které se příliš nepoužívá , a tou je nutnost vykreslení modelu z určitého počtu pohledů, aby bylo možné vyhodnotit podobnost modelů. Jednak je velmi obtížné

určit, které pohledy je nejvhodnější vykreslit a také to může být časově velmi náročné, jelikož je nutné několikrát vykreslit nejen zjednodušený model, ale i ten původní.

2.4.2 Geometrické vyjádření chyby

Používání geometrického vyjádření chyby bývá většinou jednodušší než porovnání na základě podobnosti vzhledu, a proto se také používá častěji. Nejčastěji se používá měření vzdálenosti bodů původního modelu od polygonů nového modelu. V [6] je použito následující vyjádření průměrné odchylky:

$$E_{avg}(M_i, M_f) = \frac{1}{\mid X_n \mid + \mid X_i \mid}(\sum_{v \in x_n} d^2(v, M_i) + \sum_{v \in x_i} d^2(v, M_n)) \tag{2.8}$$

a maximální odchylka:

$$E_{max}(M_i, M_f) = max(max_{v \in x_n} d^2(v, M_i), max_{v \in x_i} d^2(v, M_n)) \tag{2.9}$$

kde X_i jsou body původního modelu a X_n jsou body zjednodušeného modelu a d je vzdálenost mezi daným bodem a nejbližším polygonem modelu.

Použití maximální odchylky jako metriky dává většinou lepší výsledky, ale použití průměrné odchylky je více odolné proti šumu.

Další možnost ohodnocení sítě je změna objemu oproti původnímu modelu, případně změna plochy vůči původnímu modelu.

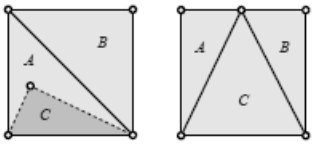
2.4.3 Kvalita polygonální sítě

Dalším parametrem, pomocí kterého je možné ohodnotit kvalitu aproximace je kvalita polygonální sítě. Nejdůležitějším a nejčastějším požadavkem je vhodný tvar trojúhelníků. Optimalizovat tvar trojúhelníků je možné již pří samotné decimaci, kdy jsou upřednostňovány trojúhelníky s maximálním minimálním úhlem, nebo podobné kritérium je požadavek na trojúhelníky s minimálním maximálním úhlem. Z toho plyne, že nežádoucí jsou trojúhelníky, které mají jeden z úhlů velmi malý, téměř nulový. Proto se někdy používá, pokud je to možné, tzv. Delaunayova triangulace, která se snaží splnit výše uvedená kritéria na tvar trojúhelníků.

Použití výše uvedené optimalizace je ovšem nutné zvážit, protože povolení trojúhelníků s malými úhly někdy vede k lepším aproximacím původního modelu.

2.4.4 Topologie

Při zjednodušování je třeba předcházet degeneraci polygonální sítě. Nejčastější je tzv. překlopení (mesh fold-over) sítě. Příklad je na 2.12. Obě triangulace na obrázku jsou ohodnoceny stejnou chybou, ovšem vlevo trojúhelník C splývá s trojúhelníkem A, což neodpovídá předpokladům pro manifold povrchy tak jak byly definovány v kapitole 2.2.



Obrázek 2.12: Příklad překlopení polygonů

Kapitola 3

Popis implementace

Cílem této práce bylo navrhnout systém pro decimaci polygonálních modelů. Byly implementovány tři výše uvedené zjednodušovací algoritmy. Také byly implementovány metody umožňující ohodnotit úspěšnost a výkonnost těchto tří metod.

Programy byly implementovány s použitím jazyka C++ na platformě Microsoft Windows s využitím vývojového prostředí Microsoft Visual Studio 2005, ovšem zdrojové kódy je možné zkompilovat také na platformě Linux s využitím překladače g++ jazyka C++. Jazyk C++ byl zvolen jelikož umožňuje přenositelnost mezi různými platformami a také protože poskytuje dobrou výkonnost.

Při implementaci decimačních algoritmů je nutné efektivně procházet polygonální sítí, což umožňuje Medical Data Segmentation Toolkit (použitá verze MDSTk v0.7.2beta), konkrétně nejvíce byla využita jeho součást knihovna VectorEntity

Zdrojové soubory jsou dokumentovány pomocí systému Doxygen, který umožňuje automatické generování dokumentace. Pro vizualizaci modelů byla použita knihovna OpenSceneGraph (konkrétně verze OpenSceneGraph-2.4) a pro zobrazení grafů v této práci byl použit nástroj gnuplot.

3.1 MDSTk

Medical Data Segmentation Toolkit je kolekce nástrojů pro zpracování 2D a 3D obrazových dat, původně určených pro segmentaci medicínských dat. Osahuje moduly pro zpracování volumetrických dat, ale také právě knihovnu VectorEntity, což je nízkoúrovňová knihovna pro zpracování 3D povrchů reprezentovaných sítí trojúhelníků, případně čtyřúhelníků.

Jedná se o soubor knihoven vyvíjený na Fakultě informačních technologií vysokého učení technického v Brně, konkrétně pod Ústavem počítačové grafiky a multimédií. Jedná se o open source projekt psaný v jazyce C++ a je určený pro platformy Microsoft Windows a Linux. Bližší informace a zdrojový kód je možné nalézt na internetových stránkách [8].

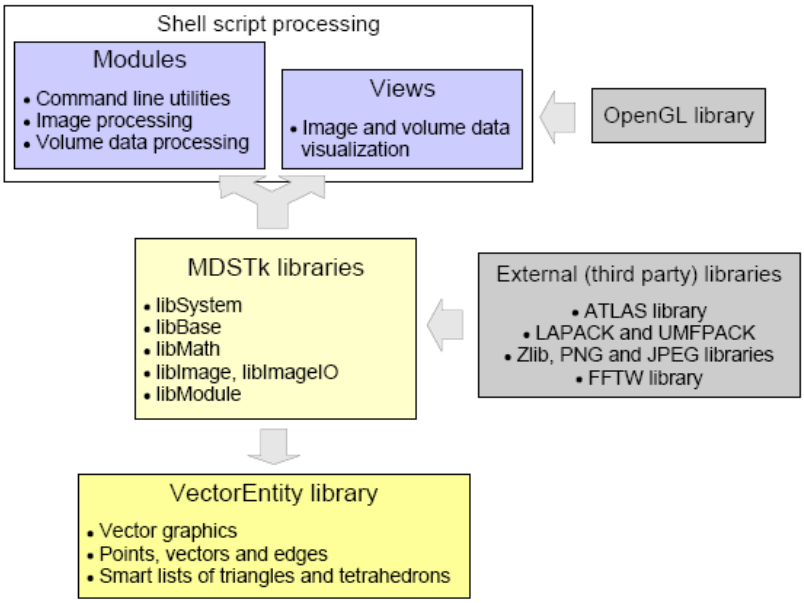
Základem pro tvorbu programů pomocí této knihovny je třída CModule, která tvoří základ téměř všech programů vytvořených pomocí této knihovny. Tato třída umožňuje vytvářet modulární konzolové aplikace s jednotným rozhraním. Aplikace mezi sebou mohou komunikovat pomocí sdílené paměti nebo rour, případně je možné v budoucnu způsoby komunikace rozšířit ještě na další možnosti. Všechny moduly mají jednotné rozhraní pro předání vstupních a výstupních dat pomocí parametrů příkazové řádky a ostatní parametry je možné přizpůsobit dle daného modulu.

Modul vždy spouští tři základní metody, které musí být definovány. Metoda startup()

je spuštěna na začátku a slouží především pro kontrolu parametrů případně načtení dat. Metoda main() představuje hlavní funkci programu. Pokud některá z předchozích dvou metod vrátí hodnotu false, tak je program předčasně ukončen. Na závěr je volána metoda shutdown(), která provádí uvolnění paměti a jiných prostředků. Další metodou, která musí být vždy definována je metoda writeExtendedUsage(), která popisuje parametry programu unikátní pro daný modul.

Díky výše popsané modularitě je možné psát jednodušší programy a jejich funkcionalitu spojit, případně rozšířit až dle potřeby pomocí například nějakého skriptovacího jazyka.

Pro mou práci byla nejdůležitější knihovna VectorEntity, jelikož bylo nutné zpracovávat modely reprezentované sítí trojúhelníků. Tato knihovna obsahuje také třídy reprezentující matematické funkce, především funkce pro práci s vektory a maticemi.



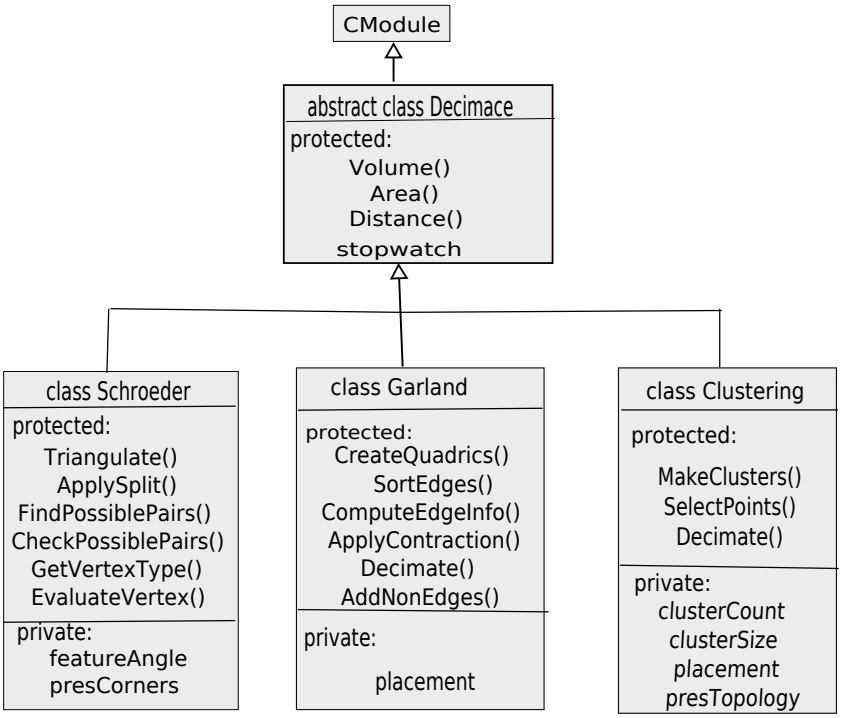
Obrázek 3.1: Architektura MDSTk převzato z [7]

Obrázek 3.1 naznačuje architekturu MDSTk. Základem je soubor knihoven, které využívá knihovna VectorEntity a také všechny moduly vytvořené s pomocí tohoto toolkitu. Dále jsou použity některé externí knihovny, např. pro některé matematické operace nebo pro zpracování určitého druhu obrázků.

Knihovna podporuje načítání a ukládání souborů v binárním formátu STL a také ukládání ve formátu VRML. Model se načte do kontejneru trojúhelníků a odtud je možné procházet kontejner vrcholů nebo vygenerovat kontejner hran a dále s ním pracovat.

3.2 Popis implementovaných tříd

Diagram tříd které byly implementovány je zobrazen na obrázku 3.2. Základem všech tříd je třída mds::mod::CModule, která je definována v MDSTk, konkrétně v knihovně Module.



Obrázek 3.2: Diagram tříd

3.2.1 Třída Decimace

První implementovaná třída je Decimace, která deklaruje proměnné, potřebné všemi třemi zbývajícími třídami, mimo jiné proměnná stopwatch sloužící k měření doby běhu metody. Dále jsou v této třídě definovány metody Distance(),Area() a Volume().

Metoda Distance() vrací vzdálenosti bodů původního modelu od trojúhelníků zjednodušeného modelu. Oproti vzorcům 2.8 a 2.9 je tady výpočet poněkud zjednodušen, jelikož dle těchto vzorců jsou brány v úvahu i vzdálenosti nového modelu od původního modelu.

Metoda Volume() vrátí objem polygonálního modelu. Algoritmus výpočtu objemu je poměrně jednoduchý. Pro výpočet objemu je nutné zvolit promítací rovnu, do které se promítne každý trojúhelník v modelu. Následně je vypočítán objem tělesa tvořeného původním trojúhelníkem a trojúhelníkem promítnutým do zvolené roviny. Poté je nutné rozhodnout zda se bude objem tohoto tělesa přičítat nebo odečítat k celkovému objemu. Toto rozhodnutí je provedeno na základě normály původního trojúhelníku, přesněji je použita složka normály kolmá na promítací rovinu. Pokud je tato hodnota kladná tak se objem přičítá, jinak se odečítá.

3.2.2 Třída Schroeder

Třída Schroeder implementuje decimaci použitím Schroederovy metody decimace vrcholů. Tato třída obsahuje dvě privátní proměnné featureAngle a presCorners. Proměnná featureAngle určuje úhel od kterého je již hrana považována za významnou hranu. Nalezené

významné hrany pro jsou uloženy pro každý vrchol, který je označen jako vrchol ležící na vnitřní hraně nebo rohový. Další proměnnou, kterou tato třída využívá je presCorners, která určuje zda bude metoda zachovávat nebo odstraňovat rohové vrcholy.

Dále je zde definováno několik metod. Metoda GetVertexType() dostává jako parametr vrchol a nastavuje tomuto vrchol příznak dle toho o jaký vrchol se jedná. Následně je vypočítáno ohodnocení tohoto bodu s využitím metody EvaluateVertex(). Metody FindPossiblePairs() a CheckPossiblePairs() najdou všechny dvojice bodů pro triangulaci a zkontrolují zda se jedná o přípustné dvojice. Metoda Triangulate() následně provede triangulaci na základě dvojic bodů nalezených v předchozích dvou metodách.

3.2.3 Třída Garland

Tato třída obsahuje privátní proměnnou placement, která určuje kam se bude umisťovat bod nahrazující hranu po jejím odstranění. Třída rozlišuje následující tři možnosti:

1. BEST = pokusí se najít optimální polohu nového bodu
2. CENTER = umístí nový bod do geometrického středu původní hrany
3. CENTER-BOUNDS = rozhodne se mezi středem hrany a jejími krajními body

dále jsou ve třídě Garland implementovány tyto metody:

- AddNonEdges() najde dvojice bodů, které nejsou spojeny hranou, ale jejichž délka d je kratší než zadaná délka d_{max} a tyto dvojice bodů přidá mezi hrany se kterými algoritmus pracuje.
- CreateQuadrics() inicializuje počáteční hodnotu chyby pro všechny vrcholy v polygonálním modelu.
- ComputeEdgeInfo() na základě polohy bodu, který by nahradil tuto hranu spočítá chybu která vznikne odstraněním této hrany.
- SortEdges() seřadí hrany na základě jejich ohodnocení.
- ApplyContraction() provede kontrakci hrany a upraví její okolí.
- GetBestPosition() získá nejvhodnější pozici pro umístění nového bodu, který nahrazuje hranu

3.2.4 Clustering

Poslední implementovanou třídou je VertexClusering. Obsahuje privátní proměnné clusterCount, clusterSize. Proměnná clusterCount označuje počet clusterů, které budou vytvořeny v jednom rozměru. Celkově tedy bude vytvořeno $pocetClusteru^3$ clusterů. Počet clusterů, které budou vytvořeny zadává uživatel jako parametr při spuštění programu. Proměnná clusterSize je inicializována na základě počtu clusterů, které budou vytvořeny a rozměrech modelu.

Další proměnné, které ovlivňují chování této třídy jsou placement a presTopology. Proměnná placement určuje, který vrchol bude vybrán jako výsledný vrchol v clusteru. První možností je, že v každém clusteru bude vybrán vrchol nejblíže středu clusteru. Další

možnost je ohodnotit vrcholy dle polohy v modelu a vybrat vždy vrchol s největším ohodnocením v rámci clusteru.

Proměnná presTopology určuje způsob jakým je prováděna decimace modelu, zda bude program postupně odstraňovat pouze hrany ležící celé v jednom clusteru, tak jak je naznačeno na obrázku 2.11. Druhou možností je, že program bude postupně procházet trojúhelníky modelu a nahrazovat jejich vrcholy a přitom bude ještě odstraňovat degenerované a duplikátní trojúhelníky.

Kapitola 4

Výsledky

4.1 Způsob testování

Každý z vytvořených programů po ukončení decimace volá funkci, která zajistí tisk informací o průběhu decimace do souboru. Jsou tisknuty mimo jiné informace o délce běhu programu, objemu a ploše zjednodušenného modelu a také o průměrné a maximální odchylce původních bodů od nově vytvořeného modelu. Pro generování dat tedy stačí jednoduchý skript, který postupně spustí jednotlivé programy se vstupními modely a s požadovanou velikostí výsledného modelu. Jako testovací velikosti zjednodušenného modelu jsme vybrali velikosti 1,2,4,6,8 a 10 procent původní velikosti. Poněkud složitější je generování dat s použitím clusteringu, zde je možné výslednou velikost pouze odhadnout na základě zadaného počtu clusterů. Soubory s daty o proběhlých decimacích jsou zpracovány s využitím skriptu v jazyce PHP.

4.2 Modely pro testování

Modely jsou zobrazeny na obrázku 4.1 a rozměry těchto modelů jsou v tabulce 4.1. Rozsah velikostí modelů je poměrně velký, nejmenší model zubu má pouhých 20000 trojúhelníků a největší model blade má přibližně 1,7 milionu polygonů.

Název modelu	počet trojúhelníků	počet vrcholů
zub	20460	10232
bunny	69664	34834
horse	96966	48485
angel	474010	237000
hand	654666	327323
dragon	870891	435403
buddha	1084236	542060
blade	1765388	882954

Tabulka 4.1: Velikosti testovaných modelů

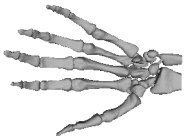
(a) angel

(b) blade

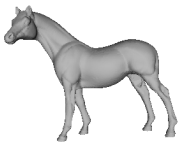
(c) buddha

(d) bunny

(e) dragon



(f) hand



(g) horse

(h) zub

Obrázek 4.1: Testovane modely

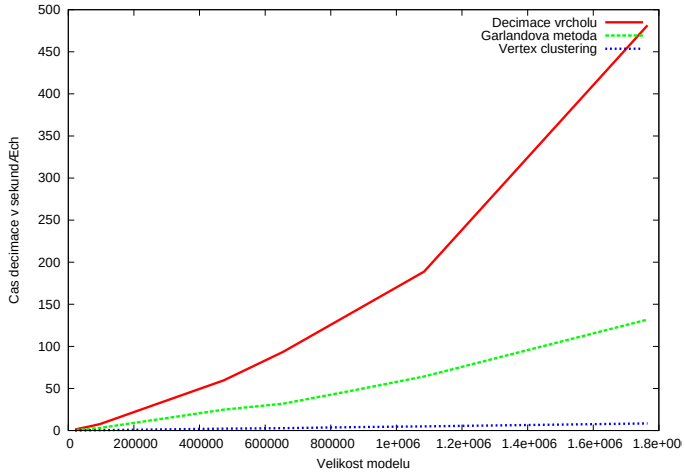
4.3 Rychlost metod

V tabulce 4.2 jsou uvedeny časy v sekundách dosažené při provádění redukce modelů na deset procent původní velikosti, každou ze tří metod. U metody vertex clusering je možné velikost výsledného modelu určit pouze přibližně a proto naměřené časy pro tuto metodu je nutné brát pouze orientačně.

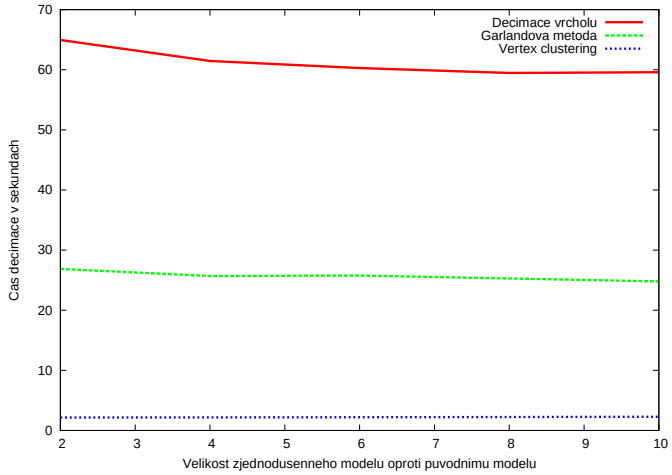
Z grafu 4.2 je jasně vidět, že nejhorší časovou charakteristiku má metoda decimace vrcholů. Čas potřebný touto metodou pro zpracování modelu roste exponenciálně s velikostí modelu. Obdobně u Garlandovy metody roste délka zpracování exponenciálně, ovšem

Název modelu	Decimace vrcholů	Garlandova metoda	Vertex clusering
zub	1.231	0.517	0.089
bunny	5.300	2.146	0.298
horse	7.738	3.109	0.403
angel	59.587	24.791	2.276
hand	93.438	31.914	2.863
dragon	141.410	47.834	4.050
buddha	188.845	64.222	4.964
blade	481.440	131.850	8.368

Tabulka 4.2: Tabulka porovnávající rychlosti jednotlivých metod



Obrázek 4.2: Časy decimace jednotlivých metod v závislosti na velikosti modelu



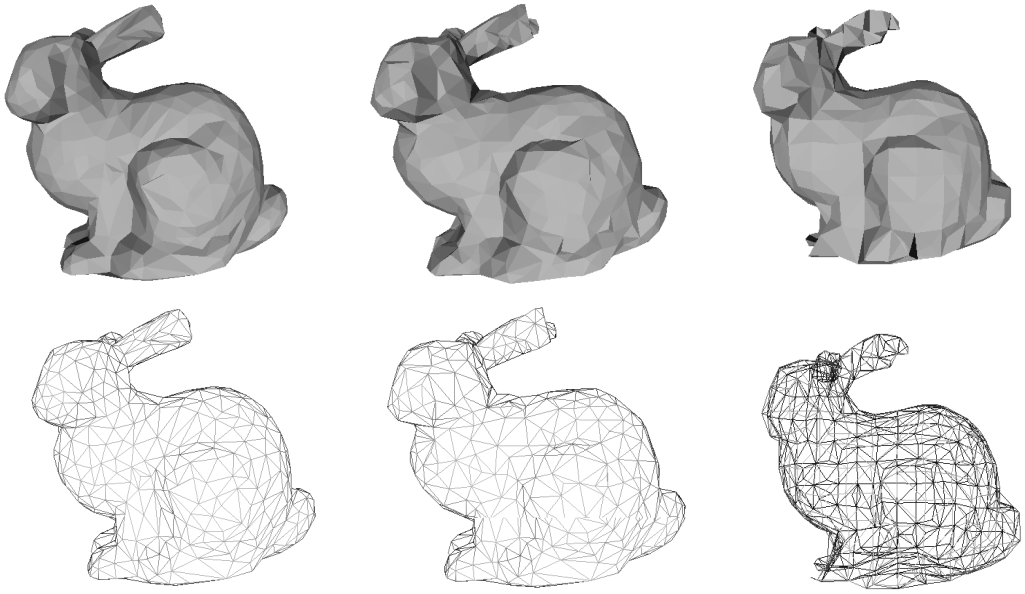
Obrázek 4.3: Časy decimace jednotlivých metod v závislosti na velikosti modelu

v případě této metody není narůst tak zřetelný. Nejlepších časů dosahuje metoda vertex clustering, u které roste potřebný čas pouze velice pomalu s velikostí modelu.

Na dalším obrázku 4.3 je srovnání růstu času potřebného metodami v v závislosti na velikosti požadované aproximace původního modelu. Je zde srovnána doba potřebná jednotlivými metodami pro zredukování modelu na 10,8,6,4 a 2 procenta původní velikosti modelu. Narůst času v případě požadavku na menší aproximaci modelu je zřetelný pouze u decimace vrcholů. U zbývajících dvou metod není patrný vliv velikosti.

4.4 Vyhodnocení kvality aproximace

Jak bylo uvedeno v kapitole 2.4, chybu aproximace je možné vyjádřit několika způsoby, první z možností je porovnat odchylku bodů původního modelu od nového modelu. Další možnost jak ohodnotit kvalitu aproximace je porovnat poměr objemu původního modelu



(a) Garlandova metoda (b) Decimace vrcholů (c) Clustering

Obrázek 4.4: Zjednodušení modelu bunny pomocí jednotlivých metod

a nového modelu, případně je možné také porovnat poměr plochy původního modelu a nového modelu. Také je velmi užitečné porovnat vzhled původního a nového modelu

4.4.1 Podobnost vzhledu

Při vyhodnocování podobnosti vzhledu není nezbytně nutné používat matematické vztahy uvedené v kapitole 2.4.1. Je dostačující porovnání pouze podle vzhledu výsledného modelu.

Na obrázku 4.4 je srovnání modelu bunny s požitím všech tří metod při redukci na deset % původní velikosti modelu. Je zde zobrazen povrch modelu i jeho trojúhelníková síť. Pro získání modelu na obrázku 4.4a byla použita garlandova metoda, model na obrázku 4.4b decimace vrcholů a na 4.4c vertex clustering.

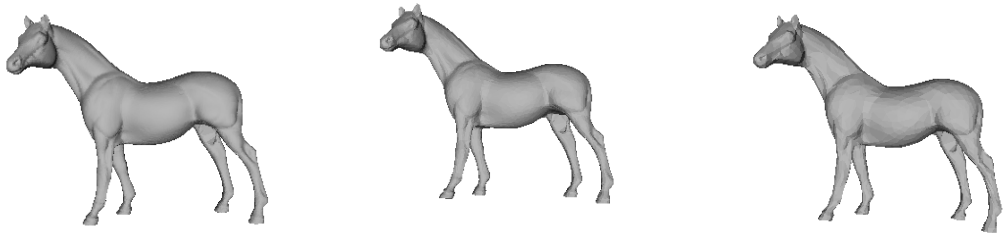
Z obrázků je vidět, že decimace i Garlandova metoda produkují obě kvalitní aproximace. Aproximace pomocí vertex clusteringu, je více odlišná od původního modelu, ovšem stále se jedná o relativně kvalitní aproximaci.

Na obrázku 4.5 obrázku je vidět postupné zjednodušování modelu horse s využitím garlandovy metody až na pouhých sto trojúhelníků. Tato metoda produkuje velice kvalitní aproximace, jelikož i při tak výrazné redukci jako je na obrázcích 4.5e a 4.5f jsou stále zachovány výrazné prvky původního modelu. Další příklady zjednodušenných modelů jsou v příloze B.

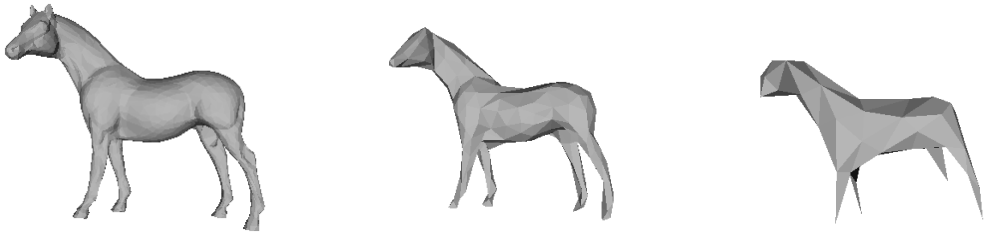
4.4.2 Vzdálenost bodů původního modelu

Aby bylo možné lépe porovnávat vzdálenosti u rozdílných modelů, tak byla před zobrazením naměřená vzdálenost vydělena velikostí modelu podél osy x.

Opět byly použity aproximace jednotlivých modelů na deset % původní velikosti. Na obrázku 4.6 je zobrazeno srovnání průměrné naměřené vzdálenosti a na obrázku 4.7 jsou



(a) Původní model = 96966 trojúhelníků (b) Model zredukovaný na 10 % (c) Model zredukovaný na 4 %



(d) Model zredukovaný na 1 % (e) 500 trojúhelníků (f) 100 trojúhelníků

Obrázek 4.5: Postupné zjednodušování modelu koně pomocí Garlandovy metody

zobrazeny maximální vzdálenosti.

Při porovnání metod tímto způsobem se u většiny modelů jeví nejúspěšnější Garlandova metoda a nejméně úspěšný je vertex clustering. Překvapivě u průměrné vzdálenosti u modelu zubu vychází jako nejúspěšnější právě vertex clustering a u zbývajících metod byly při zjednodušování tohoto modelu naměřeny horší hodnoty, přestože se jedná o velice jednoduchý model.

4.4.3 Změna objemu modelu

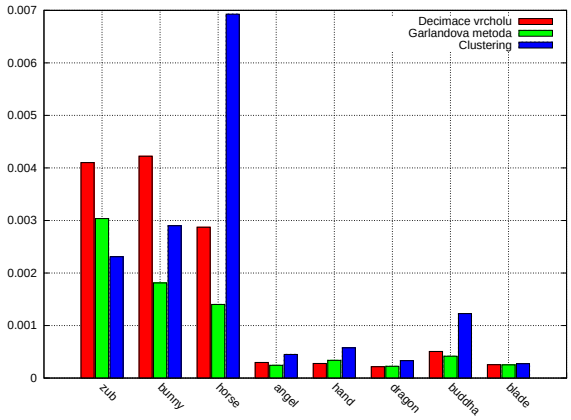
Zachování původního objemu modelu je v případě požadavku na kvalitní aproximaci velmi důležité. Není ovšem možné porovnávat přímo objemy jednotlivých modelů, jelikož jejich objemy se můžou dost lišit. Proto jsem se rozhodla porovnávat objem nově vytvořené aproximace modelu vůči původnímu modelu. Opět jsem pro vyhodnocení použila decimaci pomocí každé z metod na velikost modelu rovnou 10 % původního modelu.

Výsledky porovnání jsou zobrazeny na obrázku 4.8. Pro nejúspěšnější metodu by se poměry mezi objemy měly blížit hodnotě 1. U většiny modelů nejlépe zachovává objem Garlandova metoda a nejhůře metoda vertex clustering. Opět je zajímavé, že u modelu horse a hand dává vertex clustering lepší výsledky než zbývající dvě metody.

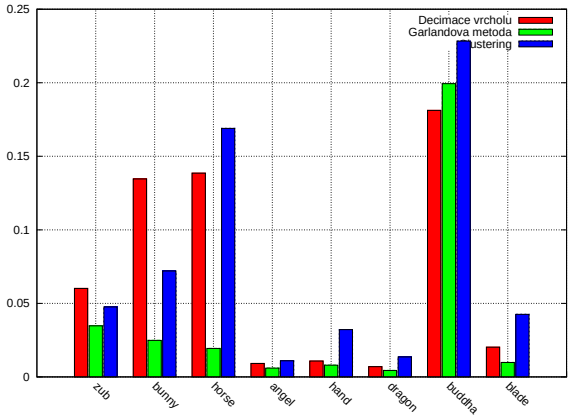
4.4.4 Změna plochy modelu

Podobně jako u srovnání objemu i zde vyjádřím poměr plochy zjednodušeného modelu vůči původnímu modelu, výsledky tohoto porovnání jsou zobrazeny v grafu 4.9.

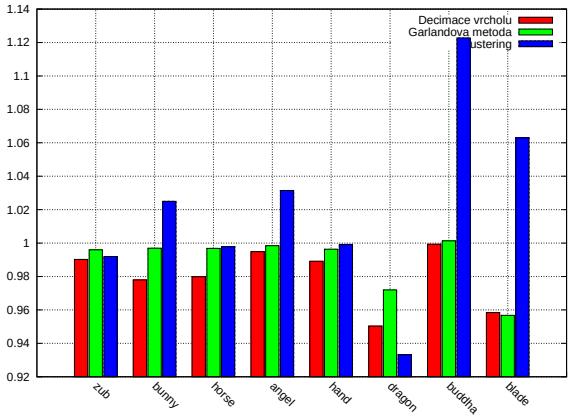
Dle tohoto kritéria je ve většině případů opět nejpřesnější Garlandova metoda. Opět i metoda vertex clustering je u některých modelů velmi přesná, např. u modelu bunny.



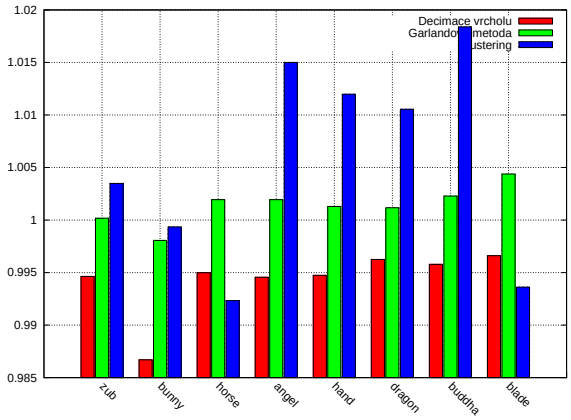
Obrázek 4.6: Průměrná vzdálenost bodů původního modelu od zjednodušeného modelu



Obrázek 4.7: Maximální vzdálenost bodů původního modelu od zjednodušenného modelu



Obrázek 4.8: Poměry objemu zjednodušenného modelu vůči původnímu modelu



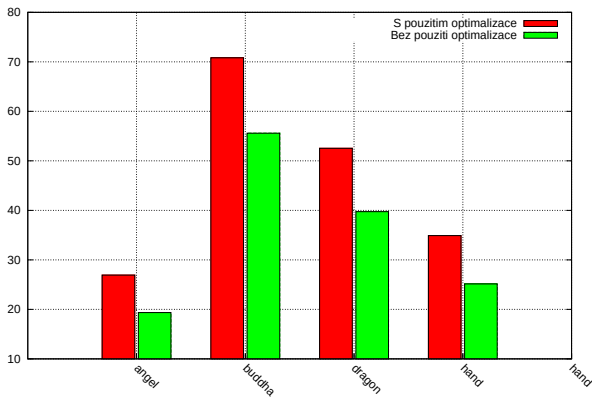
Obrázek 4.9: Poměry ploch zjednodušenného modelu vůči původnímu modelu

4.4.5 Optimalizace metody kontrakce hran

Jak již bylo zmíněno v kapitolách 3.2.3 a 2.3.2, při použití Garlandovy metody existuje více možností kam umístit nový bod nahrazující původní hranu. Tato podkapitola porovnává rozdíl mezi dvěma variantami umisťování nového bodu. V prvním případě se program pokusí najít optimální umístění každého nového bodu v druhém případě umístí bod vždy do geometrického středu původní hrany.

Pro testování jsem použila čtyři modely: angel,buddha,dragon a hand. Každý z těchto čtyř modelů byl zjednodušen na jedno procento původní velikosti modelu nejdříve s použitím optimalizace umístění nového bodu a při další decimaci byl model zjednodušen vždy s použitím geometrického středu hrany.

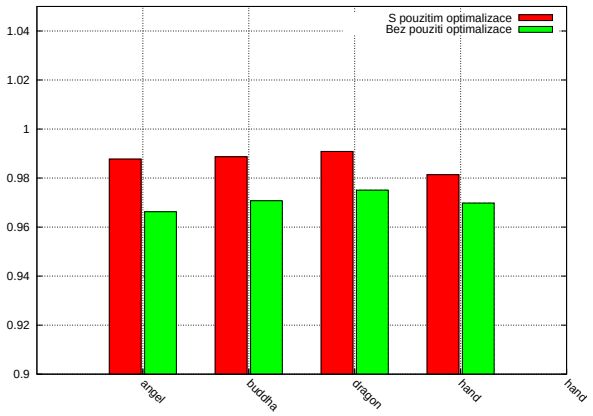
Výsledky těchto porovnání jsou zobrazeny na obrázcích 4.10, 4.11, a 4.12. Nejvýraznější vliv mělo použití optimalizace na časovou náročnost zpracování modelu, čas potřebný pro zjednodušení modelu se samozřejmě poměrně výrazně prodloužil.



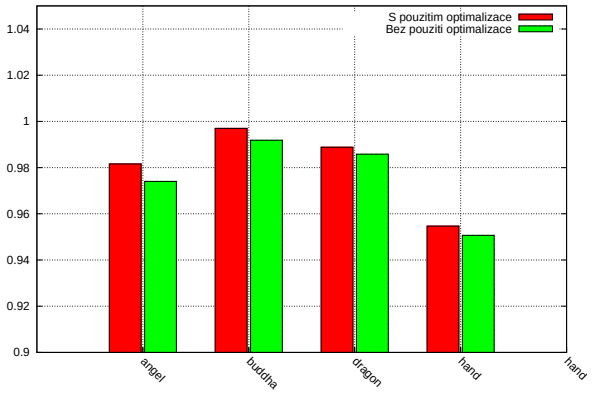
Obrázek 4.10: Porovnání doby metod v závislosti na umisťování nového bodu

V grafu a 4.11 jsou zobrazeny výsledky porovnání plochy modelů zjednodušenných nejdříve s použitím optimalizace a bez použití optimalizace. Podobně v grafu je porovnán objem 4.12 modelů.

Z naměřených výsledků vyplývá, že pokud metoda optimalizuje nově vložené body, tak dochází ke zlepšení vlastností výsledného modelu, ovšem toto zlepšení není tak výrazné jako prodloužení času potřebného pro zjednodušení modelu. Proto je nutné se při použití této metody rozhodnout zda je důležitější rychlost metody, nebo kvalitnější zjednodušené modely. Samozřejmě je možné způsob umístění také volit při spuštění programu pomocí parametru příkazové řádky.



Obrázek 4.11: Porovnání plochy modelů



Obrázek 4.12: Porovnání objemu modelů

Kapitola 5

Závěr

V této bakalářské práci byly implementovány tři vybrané metody decimace polygonálních modelů. Metody byly srovnány na základě několika různých charakteristik.

Porovnáním výsledků všech tří decimačních metod bylo zjištěno, že ve většině testovaných případů vede k nejlepším výsledkům použití Garlandovy metody. Ovšem pokud je nutné zjednodušovat velmi rozsáhlé modely, kde je kromě kvality aproximace kladen důraz na rychlost decimace, tak se jako nejvhodnější jeví použití metody vertex clustering. Zatímco čas provedení decimace u zbývající dvou metod roste exponenciálně s velikostí modelu, u vertex clusteringu roste čas pouze lineárně.

Metoda decimace vrcholů nemá příliš dobré vlastnosti. Kvalita aproximací vytvořených touto metodou je většinou o něco menší než při použití Garlandovy metody, ovšem čas potřebný k provedení decimace je obzvláště u velmi rozsáhlých modelů podstatně větší.

5.1 Možná rozšíření práce

Metoda decimace pomocí vertex clusteringu byla implementována pouze v základní variantě uvedené v [10]. Jelikož je metoda extrémně rychlá ve srovnání se zbývajícími dvěma metodami, tak by bylo vhodné implementovat některé z vylepšení této metody, aby metoda produkovala kvalitnější aproximace. Jedna z možností vylepšení je dělení prostoru pomocí oktalového stromu, tak jak je popsáno v [5]. Tento způsob provádění decimace by měl zachovávat topologii modelu lépe než námi implementovaná základní metoda simplifikace.

Decimace vrcholů neodstraňuje z modelu složité vrcholy. To vede k tomu, že u některých modelů nemusí být dosaženo požadované aproximace. Dalším krokem by tedy bylo implementovat odstraňování složitých vrcholů.

Také by bylo možné zobecnit Garlandovu metodu o decimaci modelů i s informacemi o textuře dle [1]. Ovšem pro vytvoření tohoto rozšíření by bylo nutné doimplementovat načítání modelů i z jiného formátu než je formát STL, jelikož ten neukládá informace o barvě modelu. Vhodným formátem by byl např. formát VRML.

Literatura

[1] with Color, S. S.; using Quadric Error Metrics, T.: Surface simplification using quadric error metrics. Technická zpráva, Carnegie Mellon University, 1997.

[2] Garland, M.; Heckbert, P.: Surface simplification using quadric error metrics. Technická zpráva, SIGGRAPH 97 Conference Proceedings, 1997.

[3] Kršek, P.; Španěl, M.: Materiály k přednáškám "Základy počítačové grafiky". <https://www.fit.vutbr.cz/study/courses/IZG/private>, 2008.

[4] Low, K.-L.; Tan, T.-S.: Model Simplification Using Vertex-Clustering. Technická zpráva, National university of Singapore, 1997.

[5] Low, K.-L.; Tan, T.-S.: A Topology-Preserving Polygonal Simplification Using Vertex Clustering. Technická zpráva, Osaka institute of technology, 2005.

[6] Michael, G.: *Quadric-based polygonal surface simplification*. Dizertační práce, School of Computer Science Carnegie Mellon University, 1999.

[7] Španěl, M.: Medical Data Segmentation Toolkit: A Brief Guide. 2007, [ONLINE].

[8] Španěl, M.: Medical Data Segmentation Toolkit = MDSTk. <http://www.fit.vutbr.cz/~spanel/mdstk/>, 2008.

[9] Žára, J.; Beneš, B.; Sochor, J.; aj.: *Moderní počítačová grafika*. Computer Press, 2004.

[10] Rossignac, J.; Borrel, P.: *Multi-resolution 3D approximations for rendering complex scenes*. Springer-Verlag, 1993, v Geometric Modeling in Computer Graphics, str. 455-465, B. Falcidieno and T. L. Kunii.

[11] Schroeder, W. J.: A Topology Modifying Progressive Decimation Algorithm. Technická zpráva, GE Corporate Research a Development Center, 1997.

[12] Schroeder, W. J.; Zarge, J. A.; Lorensen, W. E.: Decimation of triangle Meshes. Technická zpráva, General Electric Company Corporate Research a Development ConSolve, Inc, 1992.

Dodatek A

Uživatelský manuál

Po přeložení zdrojových kódů jsou vytvořeny tři spustitelné soubory clustering, garland a schroeder. Každý ze souborů provádí decimaci pomocí příslušné metody. Programy mají několik společných parametrů. První dva parametry jsou zděděny ze základní třídy pro modul v MDSTk. Jsou to parametry '-i' pro zadání vstupního souboru a '-o' pro zadání výstupního souboru. Jako vstupní nebo výstupní soubor lze také zadat hodnoty `stdin` pro standardní vstup a `sednout` pro standardní výstup. Hodnoty `stdin` a `stdout` jsou považovány za implicitní pokud je některý z těchto parametrů vynechán.

Další společný parametr již není zděděný ze základní třídy, jedná se o parametr '-stat 1' jehož zadání způsobí, že budou tisknuty statistiky o provedené decimaci do logovacího souboru. V případě, že je tento soubor vytvářen, tak je nazván stejně jako výstupní soubor s modelem , pouze je mu přidána přípona .log.

Další přípustné parametry určují velikost výsledného modelu. Pro programy garland a schroeder jsou to parametry '-c' určující počet trojúhelníků ve výsledném modelu a '-p' určující velikost výsledného modelu v procentech oproti původnímu modelu. U clusteringu není možné takto přesně určit velikost výsledného modelu a proto je zde použit pouze parametr '-c' určující počet clusterů do kterých bude model dělen.

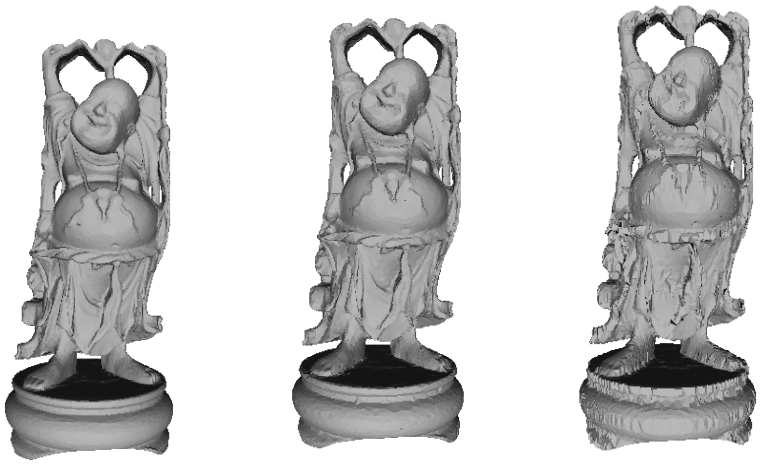
U programu garland je možné použít parametru '-plac', určující míru optimalizace umístění nového bodu, přípustné hodnoty pro tento parametr jsou `CENTER`, `CENTER_ENDS` a `BEST`. Implicitně je použita hodnota `BEST`. Další volbou u tohoto programu je zda se budou odstraňovat i dvojice bodů, které nejsou spojeny hranou. V případě, že mají být odstraňovány tak je nutné zadat parametr '-add-pairs1 1'.

Pro program schroeder existuje pouze jeden unikátní parametr a to je parametr určující zda budou odstraňovány v modelu i rohové vrcholy. Implicitně jsou rohové vrcholy odstraňovány. Pokud nemají být odstraňovány, tak je nutné zadat parametr '-pres-corners 1'.

Chování programu clustering je možné ovlivnit také parametrem '-plac'. Tento parametr přijímá dvě možné hodnoty, první je `VALUE` a druhá `CENTER`. Parametr ovlivňuje výběr výsledného bodu v rámci clusteru, pokud je zadána hodnota `CENTER`, tak je vybrán bod nejblíže středu clusteru a při zadání hodnoty `VALUE` je bod vybrán na základě ohodnocení vrcholů. Jako implicitní je zvolena hodnota `CENTER`.

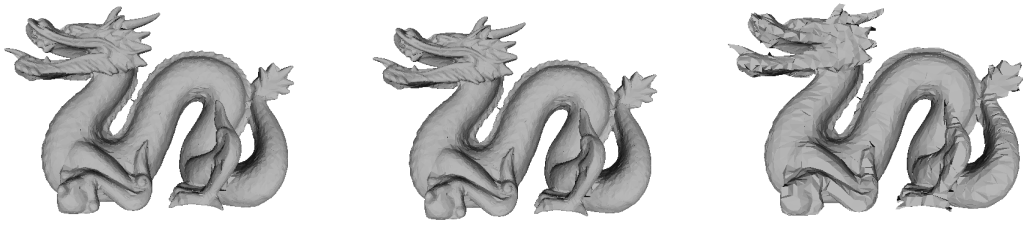
Dodatek B

Příklady zjednodušených modelů



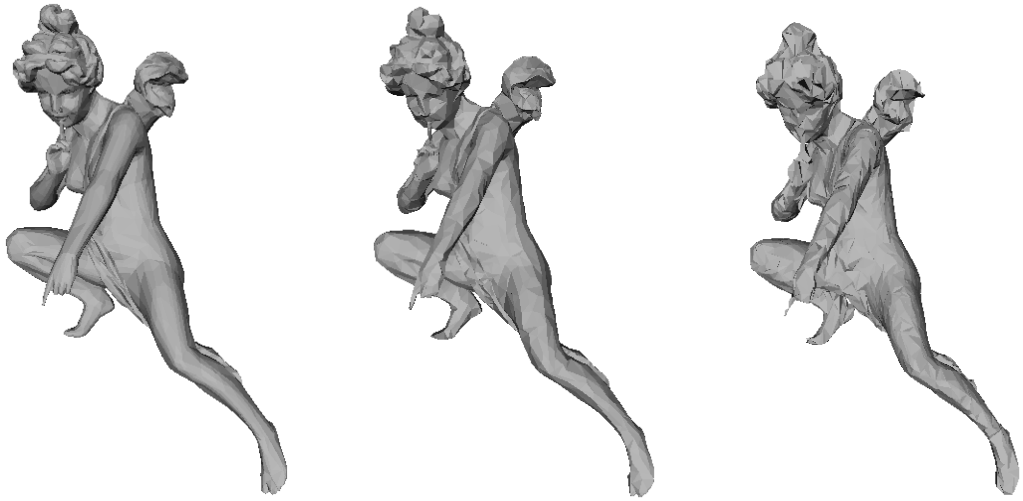
(a) Garlandova metoda (b) Decimace vrcholů (c) Clustering

Obrázek B.1: Srovnání metod při zjednodušování modelu buddha (zjednodušeno na 4% původní velikosti)



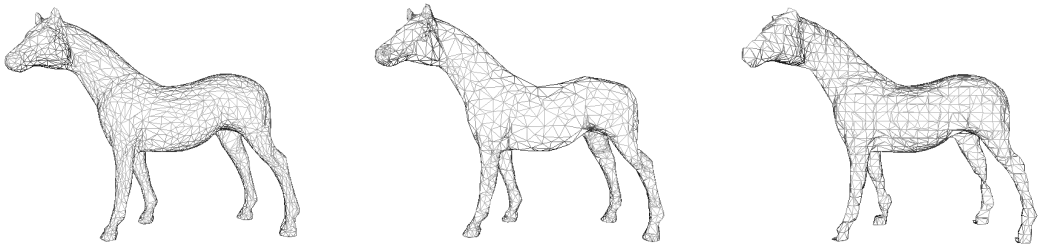
(a) Garlandova metoda (b) Decimace vrcholů (c) Clustering

Obrázek B.2: Zjednodušený model dragon (zjednodušeno také na 4% původní velikosti)



(a) Garlandova metoda (b) Decimace vrcholů (c) Clustering

Obrázek B.3: Zjednodušený modelu angel (zjednodušeno na 2% původní velikosti)



(a) Garlandova metoda (b) Decimace vrcholů (c) Clustering

Obrázek B.4: Zjednodušený model horse (zjednodušeno na 4% původní velikosti)

Dodatek C

Obsah přiloženého CD

1. Testované modely
2. Ukázky zjednodušených modelů vytvořených jednotlivými metodami
3. Soubory se zdrojovými kódy
4. Dokumentace kódu vytvořená programem Doxygen
5. Adresář se skripty pro testování metod a soubory pro vytvoření grafů použitých v této práci pomocí programu gnuplot