VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

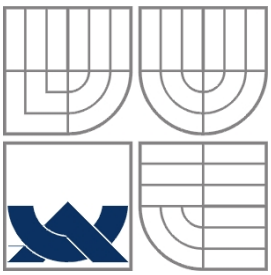
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

VYHLAZOVÁNÍ POLYGONÁLNÍCH MODELŮ

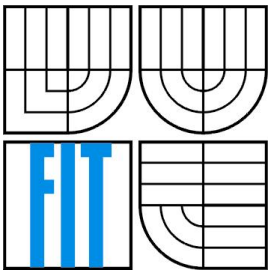
BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE Ondřej Nečas
AUTHOR

BRNO 2007



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

VYHLAZOVÁNÍ POLYGONÁLNÍCH MODELŮ

POLYGONAL MODELS SMOOTHING

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE — Ondřej Nečas
AUTHOR

VEDOUCÍ PRÁCE — Ing. Přemysl Kršek, Ph.D.
SUPERVISOR

BRNO 2007

Zadání bakalářské práce

Vyhlazování polygonálních modelů

Polygonal Models Smoothing

Vedoucí:
Kršek Přemysl, Ing., Ph.D., UPGM FIT VUT
Oponent:
Herout Adam, Ing., Ph.D., UPGM FIT VUT
Přihlášen:
Nečas Ondřej
Zadání:

1. Seznamte se s problematikou vyhlazování polygonálních modelů
2. Analyzujte problematiku vyhlazování trojúhelníkových nestrukturovaných polygonálních modelů
3. Navrhněte systém pro vyhlazování trojúhelníkových nestrukturovaných polygonálních modelů
4. Implementujte vybranou část systému ve vybraném jazyce (C/C++, Java, Python, C#).
5. Zhodnoťte dosažené výsledky a stanovte další vývoj projektu

Kategorie:
Počítačová grafika
Implementační jazyk:
C/C++, Java, Python, C#
Operační systém:
MS Windows, Linux, Unix
Literatura:
Žara J., Beneš B., Felkel P.: Moderní počítačová grafika. 1. vyd. Praha, Computer press 1998, 448 s., ISBN 80-7226-049-9

Licenční smlouva

Licenční smlouva je uložena v archivu Fakulty informačních technologií Vysokého učeni technického v Brně.

Abstrakt

Výsledek digitalizace povrchu trojrozměrného tělesa ovlivňuje mnoho faktorů. Na povrchu digitalizovaného tělesa se může objevit řada různých artefaktů. Odlišnosti od originálu jsou způsobeny rozlišením zařízení nebo jeho přesností. K eliminaci těchto artefaktů existuje mnoho vyhlazovacích algoritmů. Podle povahy modelu a cíle jeho určení budeme chtít vybrat ten nejvhodnější algoritmus, proto je dobré znát jejich vlastnosti a omezení.

Klíčová slova

Vyhlazování povrchu, srovnání, metriky, Laplace, polygon, hrana, vertex

Abstract

The result of digitalization 3d surface is affected by many factors. After digitalization can on the surface emerge some unwanted artefacts. The differences from original are caused by resolution of the device or its limited precision. For elimination of this artefact exists many smoothing algorithms, but we want to choose that which best suits this purpose. So it is useful to know their limits and characteristics.

Keywords

Mesh smoothing, comparison, metrics, Laplace, polygon, edge, vertex

Citace

Ondřej Nečas: Vyhlazování polygonálních modelů, bakalářská práce, Brno, FIT VUT v Brně, 2007

Vyhlazování polygonálních modelů

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Přemysla Krška, Ph.D.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

Ondřej Nečas
15.5.2007

Poděkování

Chtěl bych především poděkovat svému vedoucímu Ing. Přemyslu Krškovi, Ph.D. za trpělivost a poskytnutí odborných informací, rad a materiálů. Také bych chtěl poděkovat za poskytnutí nástroje MDSTk všem jeho autorům.

Obsah

1 Úvod

S prudkým zvýšením hardwarového výkonu počítačů v nedávné době nastal rozvoj počítačové grafiky. Počítače jsou schopné zobrazit komplexní scény s mnoha složitými objekty v reálném čase nebo mohou provádět složité výpočty nad těmito objekty. Využití trojrozměrných objektů je hlavně v oblasti grafické vizualizace, ale i pro různé vědecké výpočty.

Objekty mohou být reprezentovány různým způsobem. Nejrozšířenější je pomocí polygonálního povrchu – povrch modelu je popsán ploškami v prostoru, které ohraničují jeho objem. Nejčastěji jsou to trojúhelníky pro jejich snadné a rychlé zobrazení. Vždy ovšem nemáme k dispozici přímo polygonální model, ale jinak popsaný model, který potřebujeme na polygonální model převést. Mnohdy máme k dispozici volumetrický model získaný naskenováním reálného objektu pomocí zařízení CT či MRI nebo jiných a potřebujeme je převést na hraniční polygonální model.

Při tomto procesu dochází ke vzniku různých artefaktů na povrchu modelu, které se budeme snažit odstranit nebo eliminovat. Eliminace těchto nechtěných artefaktů spadá do problematiky vyhlazování polygonálních modelů, které se bude věnovat následují práce.

Celá problematika vyhlazování polygonálních modelů je velmi rozsáhlá. Od naskenování trojrozměrného objektu a s ním spojený vznik šumů a dalších nepřesností přes extrakci polygonálního povrchu s těchto naskenovaných dat po samotné vyhlazení a další úpravy na modelu. Každá etapa tohoto procesu nějak ovlivňuje výsledný model a podle toho je třeba zvolit nejvhodnější vyhlazovací algoritmus se správnými parametry. Vyhlazením ovšem proces získání finálního modelu nekončí – této problematice se věnují studie o decimaci a restrukturalizaci polygonální sítě. To už ale není předmětem této práce.

Některé potřebné znalosti o reprezentaci 3d modelů, procesu extrakce polygonální sítě a vyhlazovacích algoritmech jsou nastíněny v teoretickém rozboru. Následuje návrh systému, kde je popsáno jedno možné řešení a finální návrh. V implementaci je podrobněji popsána implementace algoritmů a pomocných metod. Ve výsledcích jsou vybrány některé porovnání algoritmů demonstrované na polygonálním modelu. V závěru jsou shrnuty poznatky a navržena možná rozšíření projektu a navazující projekty.

2 Teoretický rozbor

V této kapitole bude stručně popsána problematika reprezentace trojrozměrných objektů a nastíněno několik nejdůležitějších příkladů. Také zde bude přiblížen algoritmus marching cubes, který je nejpoužívanější pro převod volumetrických dat na polygonální povrch. U polygonálních modelů bude popsána struktura okřídlené hrany pro efektivní procházení sítí. Důležité při vyhlazování polygonální sítě je znát také původ nechtěných artefaktů, který bude popsán v kapitole „Extrakce polygonálního modelu z objemových dat“. Nakonec budou popsány některé vyhlazovací algoritmy.

2.1 Reprezentace 3D objektů

Každá grafická scéna se skládá s trojrozměrných objektů, které jsou nastíněné různými metodami, jsou na ně aplikované různé efekty a transformace. Tyto objekty je nutné nějak přesně matematicky popsat. Záleží na konkrétním použití a možnostech v dané oblasti, který model zvolit. V oblasti CAD se používá CSG metod nebo hraniční reprezentace. Pro zobrazení v reálném čase jsou výhodné polygonální modely. V lékařství na počítačovém tomografu se zase vytvoří volumetrický model snímané oblasti.

2.1.1 CSG (Constructive solid geometry)

Objekty jsou tvořeny pomocí základních prostorových těles nebo těles vytvořených protažením plochého profilu po křivce či jeho rotací kolem osy v prostoru. Tyto tělesa dále pomocí operací sjednocení, průniku a rozdílu spolu s příslušným stromem těchto operací tvoří komplexnější objekty.

Uložení objektu tímto způsobem je paměťově úsporné a geometricky přesné. Pro výslednou vizualizaci se často objekty převádí na hraniční reprezentaci a nebo se zobrazují přímo metodami raycastingu. Nejčastější použití techniky CSG je v CAD systémech.

2.1.2 Implicitní plochy

Jinak též blobs, soft objects nebo metaballs. Model je definován pomocí funkce vyjadřující potenciální pole. V místě, kde je funkce nulová se nachází povrch tělesa.

Používají se zejména pro organické objekty, ale lze je kombinovat s CSG a využít v CAD systémech. Pro zobrazení se nejčastěji navzorkuje objem tělesa a převede pomocí marching cubes na polygonální reprezentaci. Tuto techniku je možné použít i pro zobrazení v reálném čase. Přesnější, ale náročnější zobrazení se provádí metodou raycastingu.

2.1.3 Dekompoziční modely

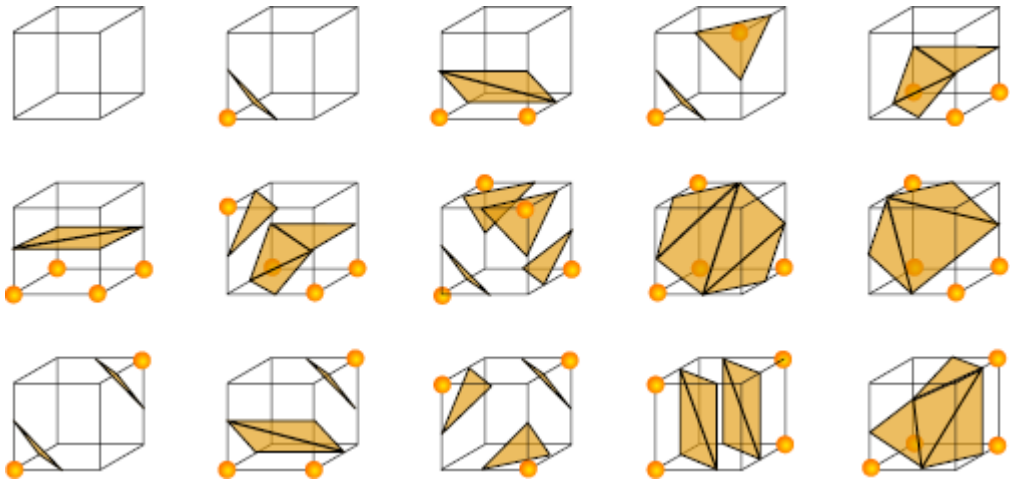
Model je tvořen tzv. voxely (spojení slov volume a pixel). Jde vlastně o trojrozměrnou bitmapu, proto voxely nesou nejčastěji informace o barvě, průhlednosti nebo jiné, ale nikoli informaci o pozici v prostoru, která je zřejmá z mřížky. Pro podobnost s bitmapou je možné použít sekvenci obrázků a interpretovat je jako trojrozměrný objekt. Např. 400 obrázků o rozměrech 512x512.

Používají se hlavně v lékařství jako data získaná pomocí CT/MRI technik. Mimo jiné byly použity v několika počítačových hrách k zobrazení předmětů, ale v tomto směru se příliš neujaly. Dnes se voxely používají k modelování terénu. Zobrazují se opět algoritmem marching cubes který je jednoduchý a efektivní.

2.1.3.1 Algoritmus marching cubes

Algoritmus byl poprvé představen na konferenci SIGGRAPH v roce 1987 autory Lorensenem Clinem. Jak je zmíněno výše algoritmus se používá hlavně na převod implicitních ploch a dekompozičních modelů na polygonální síť. Princip algoritmu je velice jednoduchý a lze ho snadno implementovat do automatizovaných systémů. Takto vzniklé modely trpí typickými schodovými artefakty nebo velkými plošinami, které je nutné eliminovat.

Na začátku se načtou dvě vrstvy voxelů a postupně se prochází pomyslnou krychlí, jejíž vrcholy tvoří osm voxelů – po čtyřech sousedních vrcholech ze spodní a po čtyřech z horní vrstvy. Stanoví se hodnoty v těchto vrcholech. Pokud je buňka plná, tak je hodnota 1 jinak 0. Když mají všechny vrcholy krychle stejnou hodnotu, můžeme ji přeskočit, protože krychle povrch neprotíná. Z uspořádaných hodnot se vytvoří osmibitový index do tabulky možných kombinací povrchů a poskládá se výsledný objekt. Potom je ještě nutné ošetřit případy nekompatibilit některých krychlí vedle sebe.



Obrázek 1: 15 jedinečných možností krychlí – 256 všech možných konfigurací lze odvodit otočením nebo inverzí. ([7] Wikipedia, the free encyclopedia, http://en.wikipedia.org/wiki/Marching_cubes)

2.1.4 Hraniční reprezentace

Definován je pouze povrch objektů – informace o výplni tělesa se dají odvodit z povrchu. Pro aplikaci některých operací je nutné, aby byl povrch tělesa uzavřený. Mezi tyto operace patří třeba Booleovské operace: sjednocení, průnik a rozdíl, nebo výpočet objemu tělesa a mnoho dalších. Povrch může být definován křivkovými plochami. Pro modelování organických nebo pravidelných těles se zaoblenými plochami se často používají NURBS plochy. Existuje ještě mnoho dalších druhů křivkových povrchů, ale záleží na konkrétním účelu a požadovaných vlastnostech. NURBS mají velkou výhodu, protože jimi lze snadno definovat kruh. Pro zobrazení se tyto plochy často převádí na polygonální síť. Zvolí se vhodná úroveň dělení křivkové plochy a aproximuje se polygonálním povrchem. Takovýto model, ale může mít velký počet polygonů. Zobrazení scén s velkým počtem polygonů dnes není problém, protože dnešní grafické akcelerátory zvládají renderovat stovky milionů trojúhelníků za sekundu.

2.1.4.1 Polygonální modely

Nejrozšířenější hraniční reprezentací je definice trojrozměrného objektu pomocí polygonální sítě. Polygony bývají nejčastěji pro jednodušší manipulaci trojúhelníkové nebo čtyřúhelníkové. U polygonů z větším počtem vrcholů je problém v případě, že body neleží v jedné rovině. Tyto příliš deformované polygony je třeba eliminovat, kvůli jejich nekorektnímu nebo nemožnému zobrazení. Řešením tohoto problému bývá třeba rozdělit příliš deformované polygony na trojúhelníky. Dalším problémem můžou být polygony s dírou.

Polygonální model bývá uložen jako tři lineární seznamy vrcholů (vertexů), hran a polygonů. Z tabulky polygonů se odkazují do tabulky hran a odtud na jednotlivé vrcholy. Polygony jsou definovány jako posloupnost hran, musí začínat a končit stejným vrcholem. Hrana je orientovaná úsečka tvořená počátečním a koncovým vrcholem. Pro větší efektivitu každý vrchol může obsahovat zpětný ukazatel na hranu, které náleží a hrana zase zpětný ukazatel na polygon, kterému náleží. Tato definice pomocí tří tabulek může být omezena jen na tabulku vrcholů a polygonů, ale při renderování drátového modelu můžou být některé hrany renderovány zbytečně dvakrát. Dalším zjednodušením je úplné vynechání tabulky vrcholů a hran. Zde nastává problém několikanásobného uložení souřadnic některých vrcholů.

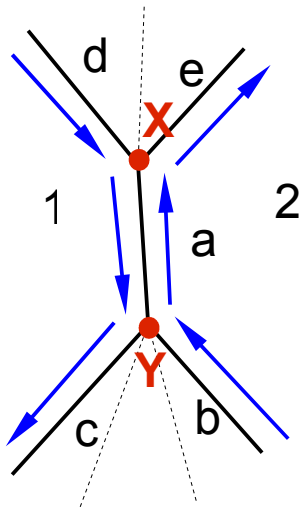
Každý s těchto základních prvků může nést nějaké další informace. V polygonu bývá uložena informace o jeho normále pro rychlejší zobrazení. Vrcholy často obsahují UV souřadnice textur, kterých může být více vrstev a několik druhů jako třeba barva, průhlednost, odrazivost a nebo pro další speciální efekty: optické nerovnosti povrchu (bump mapping), detailní posunutí vrcholů (displacement mapping). Někdy může vrchol tyto informace obsahovat přímo a není nutné je uchovávat v externí textuře, ale rozlišení této informace bude záležet na detailnosti sítě. Jiná velice efektivní struktura na uložení polygonálního modelu je tzv. okřídlená hrana.

2.1.4.2 Okřídlená hrana (Winged edge)

Pro implementaci některých složitějších algoritmů na transformaci nebo modelování polygonální sítě je vhodné uložit topologii sítě do nějaké struktury pro efektivní procházení od vrcholů přes hrany k polygonům. Mezi tyto algoritmy můžeme zařadit vyhlazování polygonové sítě nebo z modelování třeba zkosení hran. Už delší dobu se pro tyto účely modely ukládají do struktury známé jako okřídlená hrana, která poskytuje potřebné možnosti.

Nejdůležitější částí je tabulka hran, kde každá hrana obsahuje ukazatele na:

- počáteční bod hrany
- koncový bod hrany
- polygon napravo
- polygon nalevo
- předchozí hranu nalevo
- následující hranu nalevo
- předchozí hranu napravo
- následující hranu napravo



Obrázek 2: Příklad hrany

Tabulka 1: Příklad tabulky hran k obr. 2

Hrana	Počáteční vrchol	Koncový vrchol	Pravý polygon	Levý polygon	Předchozí hrana vlevo	Následující hrana vlevo	Předchozí hrana vpravo	Následující hrana vpravo
a	X	Y	1	2	d	c	b	e

Následují tabulka vrcholů a polygonů, které obsahují ukazatele na příslušnou hranu.

2.2 Extrakce polygonálního modelu z objemových dat

Jednou z nejdůležitějších oblastí využití vyhlazovacích algoritmů v praxi je právě vyhlazení polygonálního modelu získaného extrakcí z objemových dat. S tímto postupem se setkáváme hlavně v lékařství nebo na vědeckých pracovištích, kde se pracuje s CT/MRI systémy. Takto získaná data se dají vizualizovat volumetricky, ale někdy je nezbytné získat hraniční polygonální model pro další výpočty nebo úpravy.

Nejprve se naskenují jednotlivé řezy snímaného objektu. Zvolí se vhodné rozlišení pro vzorkování. Při nízkém rozlišení by se mohly ztratit některé důležité prvky, které chceme zkoumat a zase při zvolení příliš velkého rozlišení může být výsledný model zbytečně náročný na zpracování.

Potom následuje proces segmentace, kde se rozhoduje, které části budou tvořit výsledný objem a které ne. Nejjednodušší způsob jak stanovit masku by mohlo být stanovením intervalu hodnot, které patří sledovanému objemu, ale tento postup nemusí produkovat uspokojivé výsledky. Potom je nutné zvolit složitější algoritmus.

Samotná rekonstrukce polygonálního povrchu z vymaskovaných volumetrických dat se provádí nejčastěji algoritmem marching cubes, ale existují i další alternativní postupy např. marching tetrahedrons. Vznikají modely s velkým počtem polygonů v řádu desítek tisíců a více.

Nakonec se aplikují vyhlazovací filtry, kterým se věnuje tato práce. Záleží na povaze a cílovém účelu objektu, který algoritmus s jakými parametry zvolit. Každý algoritmus ovlivňuje jinak objem tělesa, jeho drobné artefakty na povrchu nebo vzdálenosti mezi jeho částmi.

V každé etapě převodu objemových dat na polygonální model dochází ke vzniku nechtěných artefaktů, které lze do jisté míry redukovat vhodnými postupy.

2.2.1 Tvorba obrazu jednotlivých vrstev objemu

V této fázi vzniká v obraze šum způsobený fyzikálními vlivy při měření, který lze jen těžko eliminovat. Hlavní problém vyplývá z principu digitalizace spojité informace do bitmapy s omezeným rozlišením a přesností. Při vzorkování hranice oblasti dochází jejímu rozmlžení. Pokud se jedná o lékařské měření s živými objekty, které dýchají, pohybují se a jsou jinak nestálé, může dojít ke znehodnocení celého měření v důsledku tohoto pohybu.

2.2.2 Segmentace

V této fázi se pro každý voxel stanoví bitová maska, pokud patří do zkoumaného objemu či nikoliv. Základní metodou je thresholding, kdy se stanoví pevná hodnota se kterou se porovnává hodnota jednotlivých voxelů. Podobně lze stanovit interval do kterého musí hodnota padnout. Pokud však tyto metody nestačí musí se vrstvy analyzovat celým objemem a nikoli jen v jednotlivých vrstvách. Při segmentaci se může změnit objem tělesa v důsledku neostrých okrajů nebo můžou vznikat různé díry, oddělené části nebo výstupky, které do zkoumaného objemu ve skutečnosti nepatří.

2.2.3 Extrakce polygonální sítě

K převodu na polygonální síť se nejčastěji používá algoritmus marching cubes nebo podobný. Tento algoritmus způsobuje na modelu typické schodové artefakty a velké plošiny. Výsledný model mívá velké množství polygonů, což může způsobit značné zpomalení dalších výpočtů a úprav. Zde se aplikují vyhlazovací filtry. Dále pokud není výsledek optimální je možno použít ještě nějakou

techniku změny topologie sítě při zachování jistých proporcí (remeshing). Pro zobrazení v reálném čase na systému, který není schopen dostatečně rychle vykreslit velké množství polygonů, existují různé metody redukce počtu polygonů (surface decimation).

2.3 Vyhlazovací metody

Existuje celá řada vyhlazovacích metod pro polygonální povrchy. Vyhlazování anorganických modelů součástek s ostrými hranami vyžaduje algoritmy, které tyto hrany zachovají a vyhladí jen větší souvislé plochy. Na ostatní modely, kde není třeba zachovat hrany, lze aplikovat původní jednoduché algoritmy jako třeba Laplaceovo vyhlazování nebo jeho modifikace, nebo filtr dolní propusti publikovaný Gabrielem Taubinem. Tyto metody se zvláště hodí pro vyhlazování organických objektů v lékařské oblasti.

2.3.1 Laplaceovo vyhlazování

Nejrozšířenější vyhlazovací algoritmus, který je implementován v mnoha programech na úpravu polygonálních modelů. Jeho implementace je velice snadná. Algoritmus produkuje už při několika málo iteracích dobré výsledky. Největší nevýhodou tohoto algoritmu je jeho schopnost smršťovat vyhlazovaný objekt, proto existují jeho různé modifikace např. Laplace+HC.

Algoritmus prochází postupně polygonální síť a pro každý vrchol spočítá novou pozici. V každém vrcholu spočítá jeho novou pozici jako průměr jeho přímých sousedních vrcholů. Vrcholy můžou ještě obsahovat váhy jejich vlivu na průměrnou pozici a též je možné nastavit míru ovlivnění původní pozice novou pozicí vrcholu. Tento postup je možné iterativně opakovat, dokud není dosaženo optimálního výsledku.

Existují dvě verze Laplaceova vyhlazování. První sekvenční verze vypočítá novou pozici vrcholu a hned upraví jeho pozici, takže pozice dalších vrcholů už budou budou ovlivněny novou pozicí tohoto vrcholu. Druhá simultánní varianta si nové pozice vrcholů během výpočtu uchovává v paměti a aktualizuje je až je celá síť přepočítána. Tato varianta je paměťově náročnější, ale ve většině případů produkuje lepší výsledky.

$$p' = p + \frac{\lambda}{n} * \sum_{i=0}^{n-1} \omega_i * (q_i - p)$$

Vzorec 2.1: Laplaceův algoritmus

p' – nová pozice vrcholu

p – původní pozice vrcholu

λ – míra posunutí do geometrického středu okolních bodů, musí platit: $0 < \lambda \leq 1$

ω_i– váhy jednotlivých hran (stejné, délka hrany, velikost úhlů, plocha trojúhelníků, atd.)

Součet vah všech hran napojených na vrchol je roven jedné.

q_i– sousední vrcholy

2.3.2 Algoritmus Laplace+HC

Jedná se o modifikované Laplaceovo vyhlazování, které se snaží redukovat míru smrštění povrchu během vyhlazování. Dosahuje toho s využitím zpětného navracení vrcholů do jejich předešlých pozic a započtením originální pozice vrcholu (pozice vrcholu před aplikací algoritmu).

Algoritmus má stejné parametry jako Laplaceův a dva další.

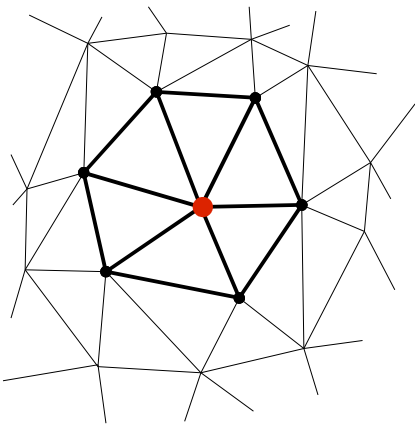
α – míra ovlivnění vrcholu jeho originální pozicí

β – míra ovlivnění vrcholu průměrem rozdílů nových pozic (spočítaných pomocí Laplaceova operátoru) od původních případně originálních pozic okolních vrcholů

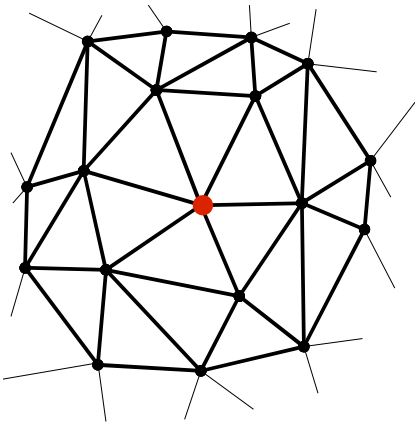
2.3.3 Algoritmus dolní propusti podle Taubina

Jedná vlastně o Laplaceův algoritmus aplikovaný ve dvou krocích. První krok se zcela shoduje s Laplaceovým algoritmem. V druhém kroku je akorát místo parametru λ použit parametr μ, který zajišťuje zpětné posunutí vrcholu do jeho předchozí pozice a zabrání tak nadměrnému smrštění. Musí platit, že $\mu <- \lambda$.

Oba poslední jmenované algoritmy mohou použít pro výpočet Laplaceova operátoru okolí vertexu druhého řádu. Samotný Laplaceův algoritmus se s okolím druhého řádu většinou nepoužívá.



Obrázek 3: Okolí prvního řádu



Obrázek 4: Okolí druhého řádu

3 Návrh

Cílem této práce je navrhnout systém na vyhlazování polygonálních modelů. Implementovat vybrané vyhlazovací algoritmy. Vyzkoušet jejich účinek na různé modely a porovnat výsledky. Pro porovnání výsledků je nutné použít nějaké objektivní postupy.

Základem objektového návrhu je implementovat programy efektivně, aby se zbytečně neopakoval kód a programy byly snadno rozšířitelné v budoucnu. Vhodné je navrhnout třídy a jejich metody tak, aby je bylo možné snadno použít v jiném projektu. Důležité s hlediska přenositelnosti je také volba programovacího jazyka, který je možné zkompilovat na různých platformách.

3.1 Návrh integrovaného řešení

Pro vyhlazování polygonálních modelů ručně uživatelem by bylo nejlepší integrované prostředí. Uživatel by si načetl model, který by se mu zobrazil v okně. Model by si mohl otáčet, přiblížit a podobně. Program by nabízel několik možných vyhlazovacích algoritmů. Po zvolení algoritmu by se zadaly vstupní parametry a algoritmus by se aplikoval na načtený model. V náhledu by byl hned viditelný výsledek operace a mohly by být implementovány ještě operace zpět nebo vpřed pro případ nevyhovujícího výsledku. Nakonec by si uživatel mohl vybraný model uložit do souboru ve vybraném formátu.

Jak je uvedeno tento návrh by se hodil spíše pro vizuální kontrolu výsledku a testování. Avšak připomíná více návrh integrovaného prostředí na úpravu polygonálních modelů. Zde by musel uživatel vše zadávat ručně nebo by systém musel podporovat skriptování, což by nebyla jednoduchá záležitost a netýká se této práce.

3.2 Finální návrh

Nejjednodušším efektivním způsobem je implementace vyhlazovacích metod do jednoduchých programů, které budou mít jen jeden účel. Veškeré parametry by se nastavovaly v příkazové řádce stejně jako vstupní a výstupní soubory. Z těchto jednoduchých jednoúčelových programů, které by fungovaly jako filtry lze poskládat i docela složitý systém. Každá významnější platforma také podporuje nějaký skriptovací jazyk. S jeho použitím by se dal celý proces zautomatizovat. Rozšíření takového systému by mělo být snadno proveditelné. Stačí do procesního řetězce přidat nový program, který bude nějakým způsobem navazovat na vstupy či výstupy stávajících programů.

Stejným způsobem by mělo jít implementovat metody pro porovnání výsledků vyhlazovacích metod. Některé metody budou jednoduše produkovat jedno číslo jako například výpočet objemu. Jiné by počítaly soubor hodnot pro další zpracování a analýzu. Různé odchylky na polygonové síti by

se předávaly na zpracování dalším programům, které by spočítaly další statistické výsledky. Nebo by se soubor těchto hodnot předal jinému programu, který b jej vizualizoval na vybraném modelu.

4 Implementace

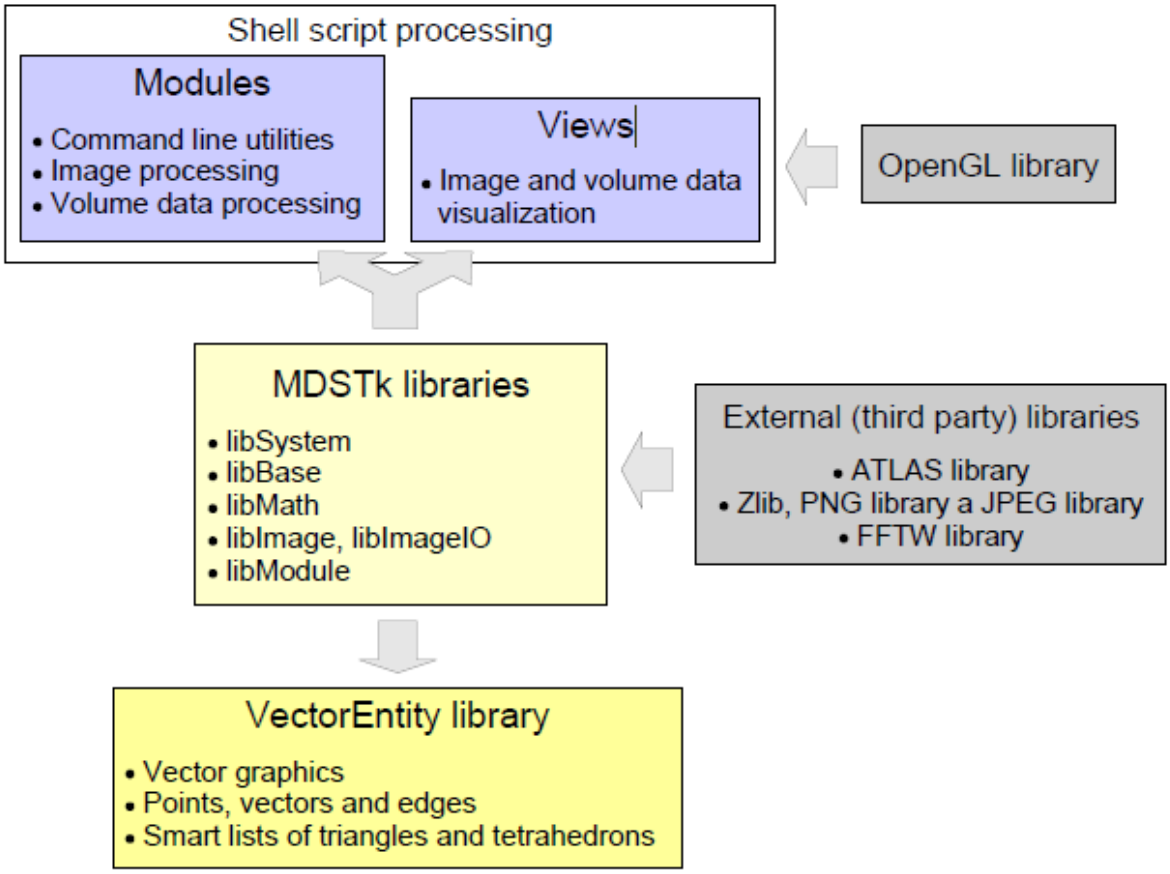
Systém je implementován v jazyce C++ na platformě Microsoft Windows. Díky jazyku C++ je kód snadno přenositelný na jiné platformy. Vizualizace modelů a případných odchylek zprostředkovává knihovna OpenSceneGraph 1.2, která je postavená na OpenGL a rovněž je napsána v jazyce C++. Pro implementaci pokročilých algoritmů na úpravu polygonálních sítí je nutné efektivně procházet od vrcholů přes hrany k polygonům. Tyto možnosti poskytuje Medical Data Segmentation Toolkit (použitá verze MDSTk v0.5.3beta), konkrétně jeho součást knihovna VectorEntity. Zdrojové soubory jsou dokumentovány pomocí systému Doxygen, který umožňuje automatickou generaci dokumentace.

4.1 OpenSceneGraph

Jedná se o open source 3d grafický toolkit psaný v jazyce C++. Vyžaduje knihovnu OpenGL a rozšiřuje její možnosti. Knihovna je určená na simulaci virtuální reality, tvorbu her nebo vědeckou vizualizaci. Zapouzdřuje volání OpenGL funkcí a umožňuje rychle a snadno produkovat složité scény. Knihovna je snadno rozšířitelná a portovatelná na většinu běžných systémů. Lze ji jednoduše propojit s produkty jiných výrobců, třeba přidat grafické uživatelské rozhraní.

OpenSceneGraph používá pro zobrazení graf scény. Graf scény je datová struktura představující uspořádání objektů uvnitř scény. Jedná se o soubor uzlů uspořádaných do stromové struktury. Jednotlivé uzly jsou ve scéně postaveny do vztahu rodič-dítě. Při ovlivnění rodiče budou stejně ovlivněni jeho následovníci. Např. Transformace rodiče transformuje obdobně všechny jeho následovníky. Princip grafu scény se využívá v celé řadě jiných grafických aplikací, her nebo např. v podobné knihovně Open Invertor.

4.2 Medical Data Segmentation Toolkit (MDSTk)



Obrázek 5: Architektura MDSTk (MDSTk - A Brief Guide)

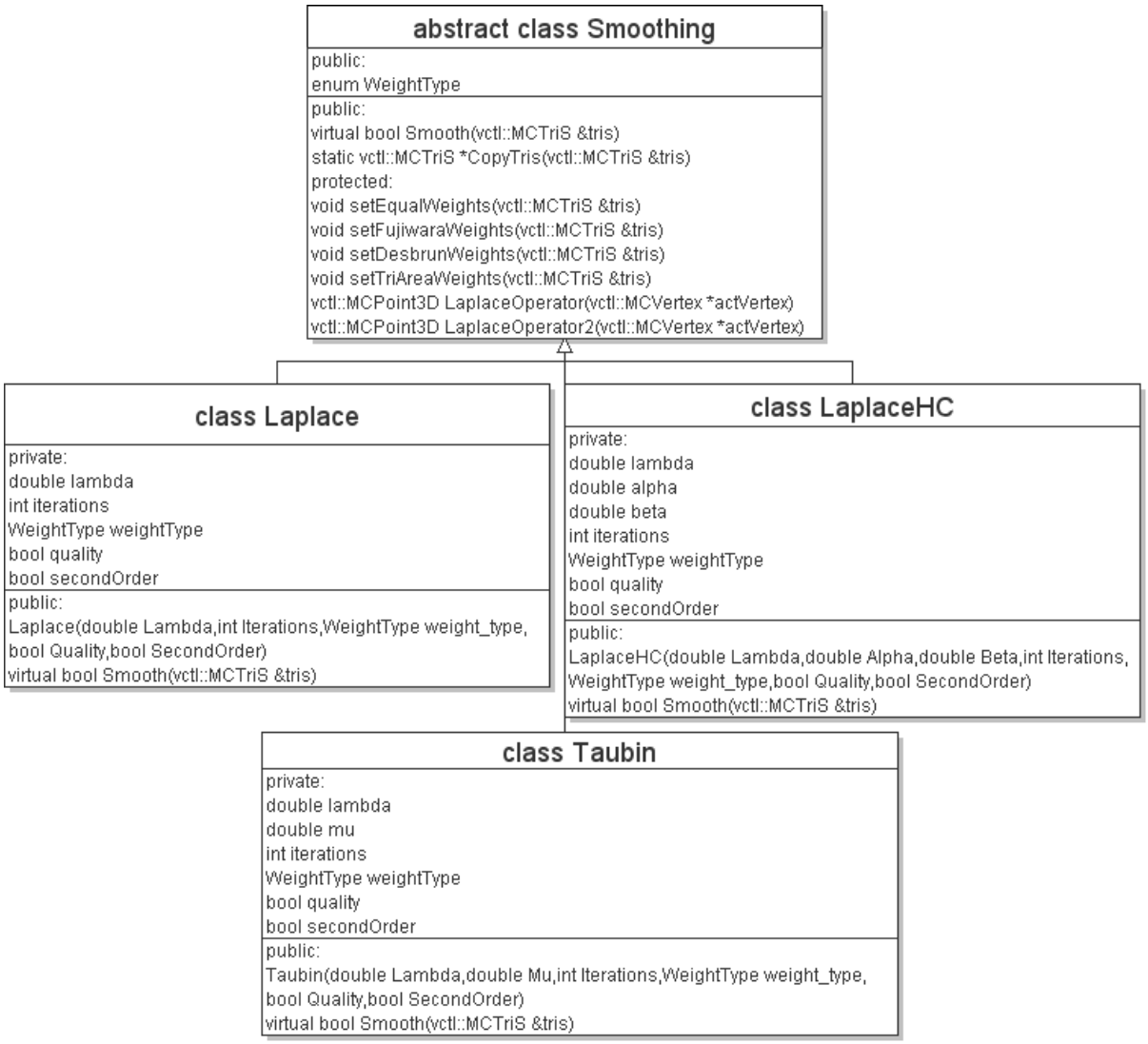
Soubor knihoven vyvíjený na Fakultě informačních technologií VUT v Brně v Ústavu počítačové grafiky a multimédií. Stejně jako OSG se jedná o open source projekt psaný v jazyce C++ pro platformy Microsoft Windows a Linux. Obsahuje nástroje na segmentaci lékařských objemových či obrazových dat.

Je vybavena systémem pro správu modulů. Moduly jsou konzolové aplikace s jednotným rozhraním. Můžou mezi sebou komunikovat pomocí rour a dalšími způsoby. Podporují více vstupů a výstupů a další rozšiřující parametry příkazové řádky. Modul vždy spouští tři základní metody, které musí být definovány. Metoda startup() se spouští na začátku a slouží hlavně k načtení a kontrole parametrů. Metoda main() je jako hlavní funkce programu. A na závěr se spustí metoda shutdown(), kde může být případně uvolněna alokovaná paměť nebo uzavřeny otevřené soubory. Pokud některá z prvních dvou uvedených metod vrátí hodnotu false, modul je předčasně ukončen.

Součástí MDSTk je knihovna VectorEntity, která poskytuje funkce pro práci s polygonálními modely a matematické funkce pro práci s vektorovou grafikou. Podporuje jak trojúhelníkové polygony, tak i polygony čtyřúhelníkové. Ve verzi 0.5.3beta knihovna podporovala načítání a

ukládání souborů v binárním formátu STL. Model se načte do kontejneru trojúhelníků a odtud je možné procházet kontejner vertexů nebo vygenerovat kontejner hran a dále s ním pracovat.

4.3 Implementace vyhlazování a pomocných metod



Obrázek 6: Diagram tříd vyhlazování

Jak znázorňuje diagram tříd pro implementované vyhlazovací metody, základem je abstraktní třída Smoothing, která zapouzdřuje metody použitelné ve všech uvedených vyhlazovacích algoritmech. Metody jsou implementovány pouze pro vyhlazování trojúhelníkových polygonálních sítí.

Ve třídě je výčtový typ WeightType sloužící k určení typu vážení hran při výpočtu Laplaceho operátoru. Metoda Smooth() je čistě virtuální metodou. Všechny třídy, které dědí od Smoothing musí mít tuto metodu implementovanou. Má jeden parametr – síť, která má být vyhlazena. Metoda vrací

typ bool pro případ, že bude síť prázdná nebo budou špatně nastaveny parametry. Pokud nic nebrání v použití parametru typu reference, je síť vždy předávána jako reference. Odpadá totiž kontrola na hodnotu NULL jako u ukazatelů.

Metoda CopyTris() vytvoří kopii sítě, ale nekopíruje zvláštní data uložená ve vrcholech, hranách nebo polygonech. V některých algoritmech je nutné vytvoření kopie polygonální sítě – potřebná je i topologie sítě např. při použití kvalitnějšího vyhlazování, kdy se nejprve přepočítají nové pozice všech vrcholů, a potom se teprve celá síť aktualizuje.

Dále třída obsahuje několik metod pro ohodnocení hran. Pro použití těchto metod je nutné mít vygenerován kontejner hran. Hrany jsou postupně procházeny a je jim přiřazena hodnota funkcí SetValue(). Jsou to metody:

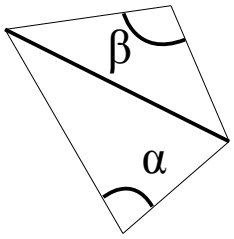
setEqualWeights() - všechny hrany jsou ohodnoceny stejně

setFujiwaraWeights() - ohodnocení hrany záleží na její délce

setDesbrunWeights() - ohodnocení hrany podle součtu kotangent protilehlých úhlů

setTriAreaWeights() - součet obsahů přilehlých trojúhelníků

Ohodnocení hran podle Desbruna:



Obrázek 7: Ohodnocení hran Desbrun

$$\omega = cotg(\alpha) + cotg(\beta)$$

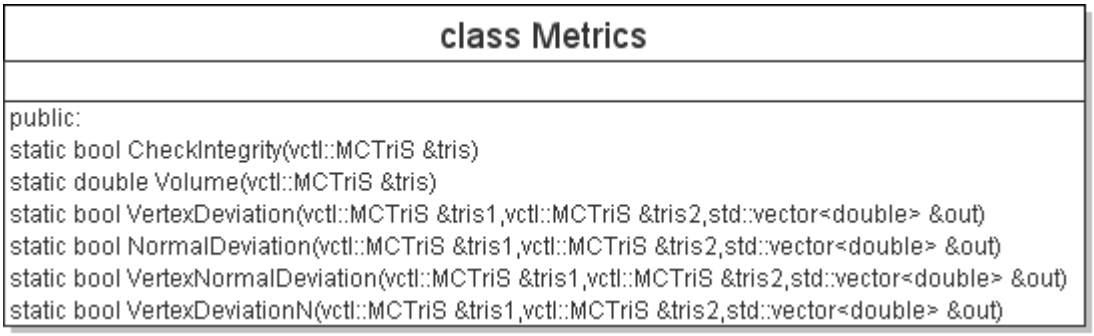
Vzorec 4.1.: Výpočet váhy podle Desbruna

Základ všech tří implementovaných metod tvoří výpočet Laplaceova operátoru. Výpočet se provádí podle vzorce 2.1. Ve třídě Smoothing jsou implementovány dvě verze - pro okolí prvního a druhého řádu. Před použitím těchto metod musí být nastavena hodnota jednotlivých hran některou z výše uvedených metod. Při výpočtu se teprve z hodnoty hrany vypočítá její váha – míra jakou vrchol přispívá k výsledné pozici. Součet všech vah napojených vrcholů musí být roven jedné, jinak by mohlo dojít k nadměrnému posunu cílového vrcholu jiným směrem. Metody mají jeden parametr – ukazatel na vrchol pro který se bude Laplaceho operátor počítat a návratovou hodnotou je výsledný bod v prostoru. Pro použití musí být vygenerován kontejner hran.

Vlastní vyhlazovací metody dědí z této abstraktní třídy a mají implementovánu virtuální metodu Smooth(), kde se skrývá vlastní vyhlazovací algoritmus. Parametry vyhlazování se předávají konstruktorem a ukládají se do privátních proměnných třídy. Kromě parametrů unikátních pro každou vyhlazovací metodu je to počet iterací algoritmu, způsob ohodnocení hran, využití okolí prvního nebo druhého řádu a způsob aktualizace nových pozic vrcholů – najednou nebo postupně. Metoda Smooth() má vždy podobné schéma: nejprve se zkontrolují parametry, pokud je vše v pořádku, pokračuje se dál, jinak metoda vrací false a ukončí se. V dalším kroku se nastaví hodnoty hran. Toto

je třeba v každé iteraci nebo aspoň jednou za určitý počet iterací přepočítat. Nakonec se vypočítají nové pozice vrcholů a aktualizují se. Jak je uvedeno výše může se to dít průběžně nebo až po přepočítání všech vrcholů.

4.4 Implementace srovnávacích metod



Obrázek 8: Diagram třídy Metrics

Třída Metrics zapouzdřuje několik metod pro výpočet rozdílů mezi polygonálními sítěmi nebo zjištění nějaké metriky polygonální sítě. Metody jsou všechny statické, protože se se jedná o jednoduché nezávislé matematické algoritmy.

Metoda CheckIntegrity() zkontroluje uzavřenost polygonálního modelu. Každá hrana musí sdílet alespoň dva polygony. Metoda Volume() jednoduše spočítá objem modelu a vrací tuto hodnotu. Pokud není povrch modelu uzavřený, nelze objem tělesa určit.

Algoritmus výpočtu objemu tělesa, jehož povrch je popsán trojúhelníkovými polygony, je docela jednoduchý. Stačí si zvolit osu podle, které se budou polygony promítat. Dále zjistit ohraničení tělesa v této ose, protože je vhodné zvolit jako promítací rovinu takovou rovinu, která objekt neprotíná. Nyní, když je známa poloha promítací roviny, můžeme postupně promítat jednotlivé trojúhelníky na tuto rovinu. Spočítáme objem tělesa, které se vytvoří mezi skutečným trojúhelníkem a jeho obrazem v rovině. Lze to spočítat například jako obsah tří čtyřstěnů tvořících toto těleso. Nakonec se objem tohoto trojhranného kvádru s jednou šikmou podstavou přičte k celkovému objemu nebo se od něj odečte. To záleží na otočení normály polygonu. Zde stačí porovnat velikost normály v promítací ose s nulou a podle výsledku přičíst objem nebo odečíst. Pokud jsme promítací rovinu zvolili na spodním ohraničení tělesa v této ose, budeme objem přičítat, jestliže je polygon viditelný z kladného směru osy.

Ve třídě Metrics jsou ještě metody pro výpočet různých druhů odchylek mezi dvěma polygonálními sítěmi. Metody mají jako parametry reference na tyto sítě a vektor ve kterém vrací hodnoty odchylek. Implementovány jsou absolutní vzdálenosti vrcholů, odchylky normál polygonů, odchylky normál vertexů a posunutí vrcholů ve směru normál v těchto vrcholech, což asi nejlépe popisuje deformace na objektu.

4.5 Výsledný systém

Jednotlivé součásti systému jsou implementovány do modulů, které poskytuje MDSTk. Moduly jsou jednoduché jednoúčelové programy.

Vyhlazení začíná programem pro načtení souboru STL. Jeho výstup se může předat vyhlazovacímu programu, který vrátí upravený výsledek podle parametrů příkazové řádky. Výsledek se může uložit opět do souboru STL nebo analyzovat pomocí programů na výpočet odchylek či objemu. Objem je spočítán a vráceno jedno číslo na výstup. Při výpočtu odchylek je vrácen soubor hodnot začínající číslem o počtu těchto hodnot. Tyto hodnoty můžou být dále zpracovány a analyzovány programy pro výpočet průměru, minima, maxima nebo směrodatné odchylky. Nebo můžou být hodnoty zobrazeny jako barva ve vrcholech na vybraném modelu, ale musí odpovídat počet vrcholů a hodnot.

5 Výsledky

K porovnání vyhlazovacích algoritmů jsem využil MSYS – Minimal SYStem, který je součástí projektu MinGW. Umožňuje použití POSIX/Bourne skriptů na platformě Microsoft Windows. Jeho součástí je sada volně šiřitelných programů známých hlavně uživatelům Linuxu. Mezi tyto programy patří např. make, awk, grep a mnoho dalších.

5.1 Srovnávací technika a nastavení

Základ porovnání tvoří několik Bourne shell skriptů. Skript zpracuje zadaný vstupní soubor STL a aplikuje vyhlazovací filtr vždy s parametry lamba 0,1-1,0 s krokem 0,1 a počtem iterací 5-25 s krokem 5. Všechny hrany jsou ohodnoceny rovnocenně. Pro každou z těchto konfigurací se provede měření na výsledném modelu. Pro Taubinův filtr je parametr mí nastaven na -1,02. Modifikovaný Laplace má nastaveno alpha na 0,0 a beta na 0,8. Skript vypočítá objem modelu a dále spočítá odchylky jednotlivých vrcholů v absolutní vzdálenosti a ve směru normál ve vrcholech. Výsledkem je soubor hodnot, který je pomocí programu awk uspořádán do tabulky a potom už není problém tato data načíst do jakéhokoli tabulkového procesoru pro přehledné zobrazení nebo další operace porovnání výsledků různých skriptů mezi sebou.

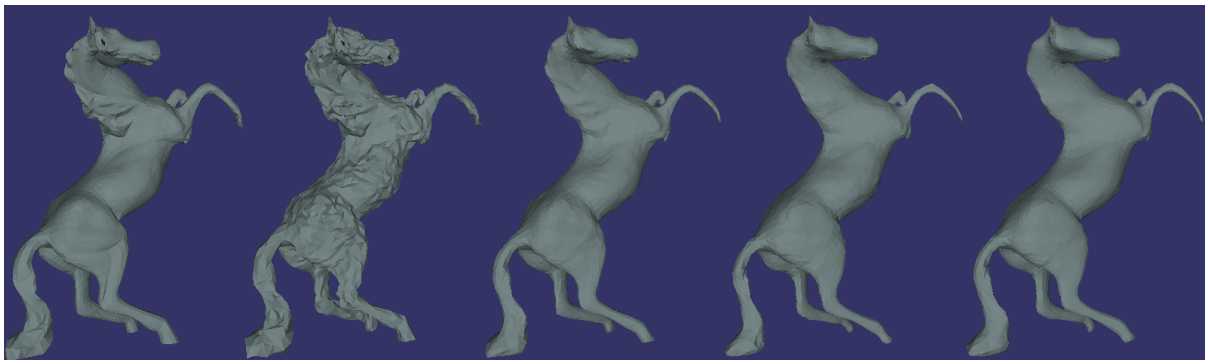
Jako testovací objekt jsem použil 3d model koně, který reprezentuje kategorii objektů spíše velkých zakulacených modelů, ale jeho končetiny jsou tvořeny menším počtem polygonů a zde lze pozorovat vliv jednotlivých algoritmů na tyto ostré výstupky. Na model byl aplikován náhodný šum, aby se vyhlazování mohlo lépe projevit. Model má 3582 vrcholů a 7172 polygonů (trojúhelníků).

5.2 Vybrané výsledky

Následují ilustrace srovnává algoritmy se základním nastavením: sekvenční aktualizace vrcholů, okolí prvního řádu, rovnocenné ohodnocení hran. Na vybraných modelech bylo aplikováno vždy 15 iterací s různým parametrem lambda, který podává nejlepší výsledky.

Tabulka 2: Doplňující údaje pro obr. 9

	Laplace	***Taubin***	***Laplace+HC***
Lambda	0,1	0,6	1,0
Objem	0,94	0,90	0,95
Čas (s)	0,640	0,578	0,563

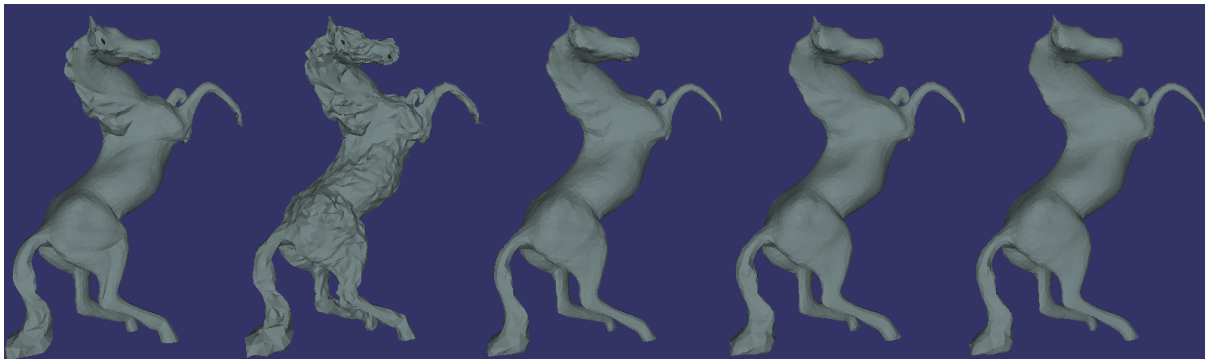


Obrázek 9: Základní nastavení filtrů, sekvenční aktualizace vrcholů. Zprava do leva: originální model, model po aplikaci šumu, Laplace, Taubin, Laplace+HC

V tomhle případě nejlepší výsledky podává modifikovaný Laplaceův algoritmus. Nejlépe si udržuje objem a zachovává objem i v tenkých končetinách. Laplaceův algoritmus při malých hodnotách lambda a vyšším počtu iterací neztrácí objem, tak rychle jako při lambda blížícím se jedné, avšak vyhlazení není tak dokonalé jako ostatní dva algoritmy. Tabinův algoritmus s tímto nastavením nepodává moc použitelné výsledky. Výsledky měření času pomocí příkazu time ukázaly, že pro model ze sedmi tisíci polygonů nemají smysl, protože časy jsou příliš malé, aby je šlo reálně porovnávat. Proto by bylo lepší měřit čas pomocí ticků procesoru uvnitř programu.

Tabulka 3: Doplňující údaje pro obr. 10

	Laplace	***Taubin***	***Laplace+HC***
Lambda	0,1	0,8	1,0
Objem	0,94	1,00	0,97

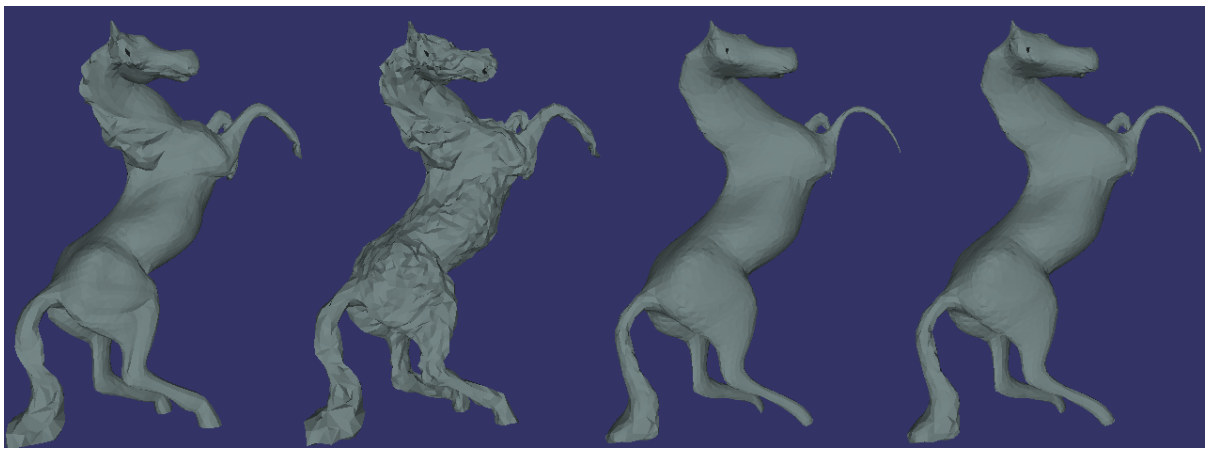


Obrázek 10: Základní nastavení filtrů, simultánní aktualizace vrcholů. Zprava do leva: originální model, model po aplikaci šumu, Laplace, Taubin, Laplace+HC

Při nastavení simultánní aktualizace vrcholů, které je náročnější na paměť, se výsledky u Laplaceova algoritmu nijak výrazně nezměnily. Objem je téměř shodný a ani po vizuální stránce není patrný rozdíl. Naopak ostatní dva algoritmy zaznamenaly výrazné zlepšení. Taubinův algoritmus si nejlépe zachoval objem i v končetinách koně, kde při sekvenční aktualizaci vrcholů výrazně ztrácel. Laplace+HC na rozdíl od Taubina vyhlazuje i některé větší artefakty, které by v některých případech bylo třeba zachovat.

Tabulka 4: Doplňující údaje pro obr. 11

	Taubin	***Laplace+HC***
Lambda	0,8	1,0
Objem	0,98	0,95

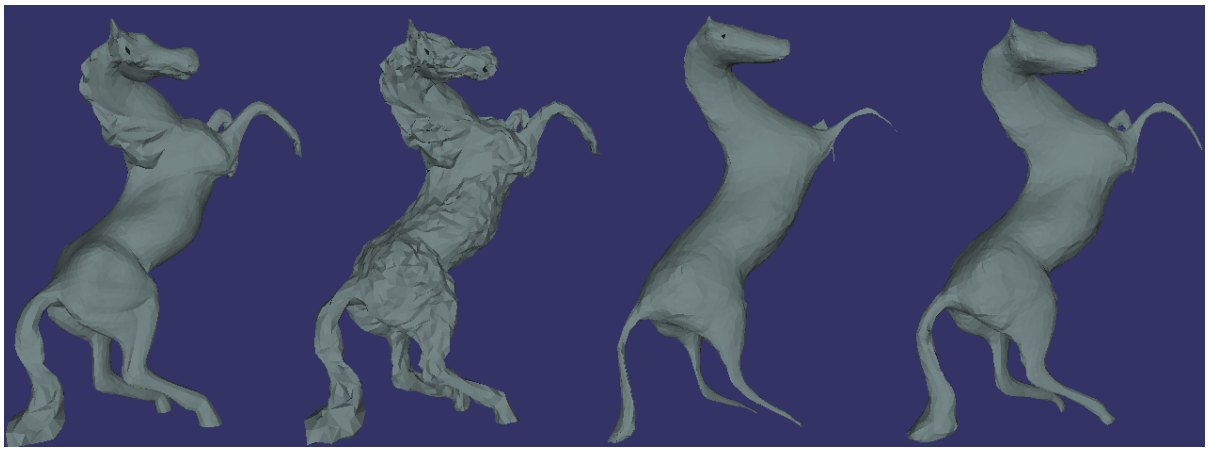


Obrázek 11: Základní nastavení filtrů, simultánní aktualizace vrcholů, okolí druhého řádu. Zprava do leva: originální model, model po aplikaci šumu, Taubin, Laplace+HC

Při nastavení simultánní aktualizace vrcholů a okolí druhého řádu Taubinův algoritmus opět lépe zachovává celkový objem, ale Laplace+HC udržuje více objemu i v ostrých vystouplých částech modelu. Je zde znatelné drastické vyhlazení vhodné jen pro hodně nerovné modely.

Tabulka 5: Doplňující údaje pro obr. 12

	Taubin	***Laplace+HC***
Lambda	0,7	1,0
Objem	0,73	0,94

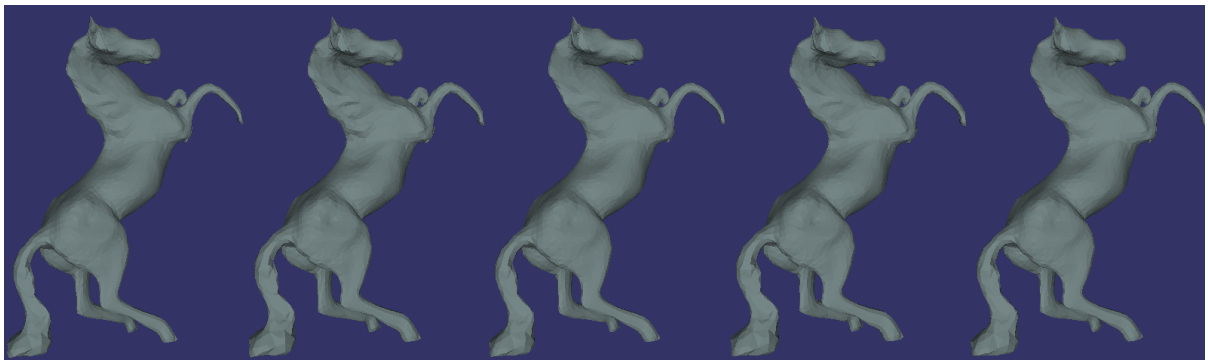


Obrázek 12: Základní nastavení filtrů, sekvenční aktualizace vrcholů, okolí druhého řádu. Zprava do leva: originální model, model po aplikaci šumu, Taubin, Laplace+HC

Jen pro srovnání s použitím sekvenční aktualizace vrcholů a okolí druhého řádu jsou výsledky zvláště u Taubinova algoritmu nevyhovující.

Tabulka 6: Doplňující údaje pro obr. 13

Iterace	***5***	***10***	***15***	***20***	***25***
Objem	0,98	1,03	1,00	1,06	1,03

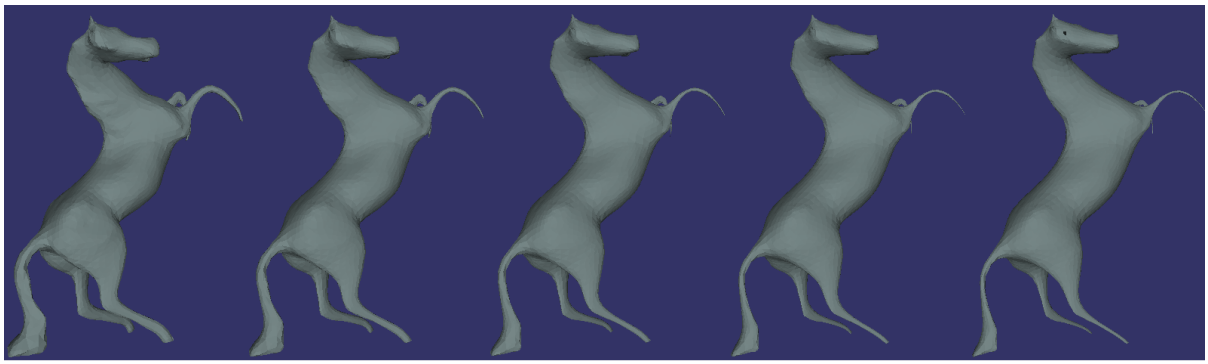


Obrázek 13: Taubinův filtr. Iterace zleva do prava : 5, 10, 15, 20, 25

Porovnání výsledků při různém počtu iterací. Použit Taubinův algoritmus s parametrem lambda 0,8 a simultánní aktualizace vrcholů. Zde je vidět, že se zvyšujícím se počtem iterací se model mírně zvětšuje. Také je pozorovatelné, jak algoritmus pracuje ve dvou krocích. Při sudém počtu iterací je nakonec aplikován zvětšující krok a model je o něco větší než při liché následující iteraci.

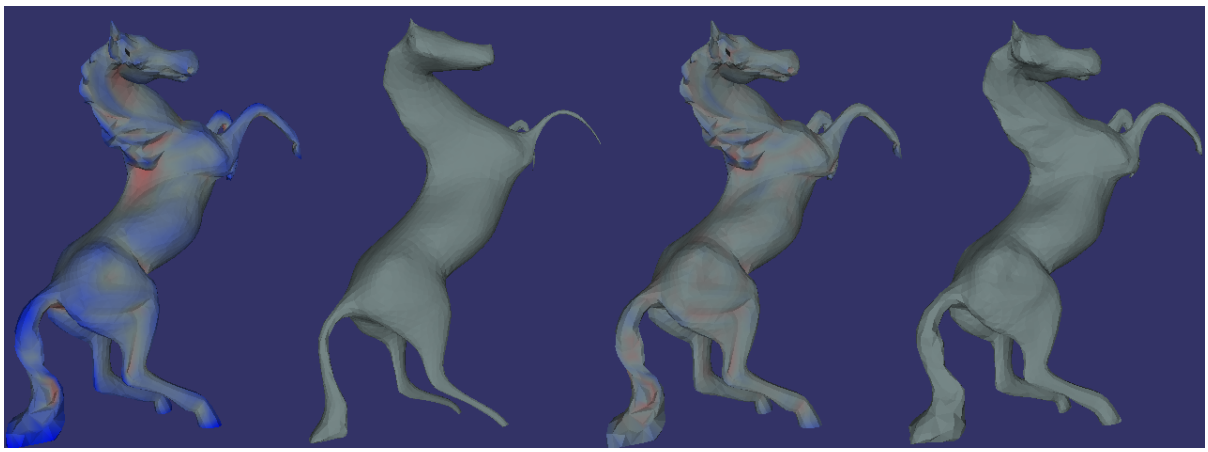
Tabulka 7: Doplňující údaje pro obr. 14

Iterace	***5***	***10***	***15***	***20***	***25***
Objem	0,91	0,85	0,81	0,77	0,73



Obrázek 14: Laplaceův filtr. Iterace zleva do prava: 5, 10, 15, 20, 25

Následky aplikace Laplaceova algoritmu s různým počtem iterací. Lambda nastavena na hodnotu 0,5 a vše ostatní jako v předešlém případě. Samotný Laplaceův algoritmus není vhodný pro vyhlazování modelů s drobnými výstupky. Smrštění při vyšším počtu iterací je příliš znatelné. Model má tendenci se mírně protahovat ve směru velkých ploch.



Obrázek 15: Odchylky od originálního modelu: originál Laplace, vyhlazený Laplace, originál Taubin, vyhlazený Taubin

Na posledním obrázku je znázornění odchylek vrcholů ve směru jejich normály. Nastavení je shodné se dvěma předchozíma srovnáními a je použito právě 15 iterací. Druhý model zleva je vyhlazen Laplaceovým algoritmem a odchylky jsou znázorněny na modelu vlevo. Obdobně je na pravé straně zobrazen model vyhlazený Taubinovým algoritmem. Pohyb vrcholů ve směru normály je znázorněn červenou barvou a obráceně modrou. Pokud je odchylka nulová je ponechána původní barva. Na první pohled je vidět, že model vyhlazený Laplaceovou metodou je značně deformovaný, ale model vyhlazený Taubinovou metodou se odchyluje jen minimálně na některých výstupcích. Algoritmus Laplace+HC jsem tímto způsobem nemohl porovnat, protože má tendenci celý model posunout v prostoru jistým směrem. Zřejmě je to implementační chyba a nebo nevhodná volba parametrů.

6 Závěr

Shrnutí výsledků práce a doporučení při použití implementovaných algoritmů. Následuje případné rozšíření práce a možné navazující projekty.

6.1 Závěrečná doporučení

Porovnáním bylo zjištěno, že Laplaceův algoritmus vyhlazuje velmi účinně, je snadný na implementaci a jeho výpočet je nejméně náročný. Je nevhodný pro vyhlazování podlouhlých modelů z důvodu jeho tendence model smršťovat, což se na těchto podlouhlých modelech zvláště drasticky projeví. Taubinův algoritmus je také velmi snadno implementovatelný a podobně náročný jako Laplaceův. Asi nejlépe zachovává objem ve všech částech modelu. Je použitelný i pro vyhlazování těch modelů, kde je třeba zachovat drobnější detaily. Algorimus Laplace+HC je podstatně náročnější na implementaci a o to je složitější jeho výpočet. Má více parametrů, proto může být náročnější najít optimální konfiguraci. Výsledky jsou podobné Taubinovu filtru. Lze ho doporučit s využitím okolí druhého řádu, kde lépe zachovává objem v podlouhlých částech než ostatní algoritmy.

6.2 Zhodnocení práce a možná rozšíření

Samotná práce porovnává výsledky jen několika málo základních algoritmů. Porovnány jsou pouze iterační algoritmy, které nezachovávají případné hrany a linie na objektech, proto by bylo zajímavé doplnit nějaký takový algoritmus a srovnat ho s těmito iteračními algoritmy. Na druhou stranu by se jednalo o jinou kategorii modelů. Pro stanovení optimálního počtu iterací pro zadaný algoritmus by bylo vhodné implementovat měření zakřivení povrchu modelu. Pro zjištění odchylek dvou různých modelů se používá Hausdorffova vzdálenost, ale její implementace není triviální.

Srovnání bylo provedeno pro různý počet iterací, různou sílu vyhlazení a další parametry, ale chybí srovnání dopadu volby vážení hran (odtud míra účasti vrcholu na výsledné pozici), protože toto nastavení zkresluje výsledky samotných vyhlazovacích algoritmů.

Pro dosažení optimálních výsledků vyhlazování by bylo vhodné zaměřit další výzkum na procesy předcházející a následující etapu vyhlazování. Před vyhlazením by to mohl být proces segmentace dat a extrakce polygonálního povrchu z těchto dat. Po vyhlazení zase procesy spojené s redukcí počtu polygonů na modelu nebo přestavba sítě modelu (remeshing).

Literatura

[1] Žára, J., Beneš, B., Felkel, P., Moderní počítačová grafika, ComputerPress, 1999

[2] Materiály k přednáškám "Základy počítačové grafiky", Brno 2007
http://www.fit.vutbr.cz/study/course-l.php?id=92

[3] J. Vollmer, R. Mencel, H. Müller, Improved laplacian smoothing of noisy surface meshes, Research report No. 711 /1999, June 1999

[4] G. Taubin. Geometric Signal Processing on Polygonal Meshes, EUROGRAPHICS ’2000

[5] R. Bade, J. Haase, B. Preim, Comparison of Fundamental Mesh Smoothing Algorithms for Medical Surface Models, University of Magdeburg 2006

[6] Wikipedia, the free encyclopedia, http://en.wikipedia.org/, 2001 – 2007

Seznam příloh