VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

SEGMENTACE POLYGONÁLNÍHO MODELU

POLYGONAL MESH SEGMENTATION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **MATÚŠ ŠVANCÁR**
AUTHOR

VEDOUCÍ PRÁCE **Ing. MICHAL ŠPANĚL, Ph.D.**
SUPERVISOR

BRNO 2019

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav počítačové grafiky a multimédií (UPGM) Akademický rok 2018/2019

Zadání bakalářské práce

Student: **Švancár Matúš**
Program: Informační technologie
Název: **Segmentace polygonálního modelu**
Polygonal Mesh Segmentation
Kategorie: Počítačová grafika

Zadání:

1. Prostudujte dostupné materiály na téma reprezentace a zpracování polygonálních sítí, ohodnocení křivosti povrchu, apod.
2. Seznamte se s metodami segmentace polygonální sítě na části.
3. Vyberte vhodné metody a navrhněte nástroj pro segmentaci polygonálních modelů zvoleného typu.
4. Experimentujte s vaší implementací a případně navrhněte vlastní modifikace metod.
5. Porovnejte dosažené výsledky a diskutujte možnosti budoucího vývoje.
6. Vytvořte stručný plakát prezentující vaši práci, její cíle a výsledky.

Literatura:

- Dle pokynů vedoucího.

Pro udělení zápočtu za první semestr je požadováno:

- Splnění prvních tří bodů zadání.

Podrobné závazné pokyny pro vypracování práce viz http://www.fit.vutbr.cz/info/szz/

Vedoucí práce: **Španěl Michal, Ing., Ph.D.**
Vedoucí ústavu: Černocký Jan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2018
Datum odevzdání: 15. května 2019
Datum schválení: 2. dubna 2019

Abstrakt

Táto bakalárska práca rozoberá a približuje problematiku segmentácie polygonálnych modelov. Prezentuje návrh interaktívnej metódy inšpirovanej metódou popísanou v článku Interactive Mesh Segmentation Based on Feature Preserving Harmonic Field. Metóda využíva graph-cut a je implementovaná vo forme webovej aplikácie. Aplikácia podporuje formáty .obj a .stl, umožňuje používateľovi načítať model, nekresliť po povrchu modelu náčrtky reprezentujúce popredie a pozadie, a spustiť segmentáciu. Po dokončení si môže používateľ výsledné modely stiahnuť, alebo pokračovať v segmentácii s jedným z nich.

Abstract

This bachelor thesis analyzes and approaches the issue of segmentation of polygonal models. It presents a design of an interactive method inspired by the method described in the Interactive Mesh Segmentation Based on Feature Preserving Harmonic Field. The method uses graph-cut and is implemented as a web application. The application supports .obj and .stl file formats, allows the user to load a model, draw sketches representing foreground and background on the surface of the model, and to start segmentation. Once completed, the user can download the resulting models or continue segmenting with one of them.

Kľúčové slová

segmentácia, polygonálny model, graph cut, geodetická vzdialenosť, web, klient-server, Node.js, Three.js, CGAL, OpenMesh,

Keywords

segmentation, mesh, graph cut, geodesic distance, web, client-server, Node.js, Three.js, CGAL, OpenMesh,

Citácia

ŠVANCÁR, Matúš. *Segmentace polygonálního modelu.* Brno, 2019. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Michal Španěl, Ph.D.

Segmentace polygonálního modelu

Prehlásenie

Prehlasujem, že som túto bakalársku prácu vypracoval samostatne, pod vedením pána Ing. Michala Španěla, Ph.D. a uviedol som všetky literárne pramene a publikácie, z ktorých som čerpal.

.....
Matúš Švancár
12. mája 2019

Poďakovanie

Chcel by som sa poďakovať vedúcemu práce, Ing. Michalovi Španělovi, Ph.D. za odborné konzultácie, trpezlivosť, cenné rady a návrhy, a tiež vecné pripomienky pri písaní tejto práce.

Obsah

Kapitola 1

Úvod

Polygonálne modely sú najčastejšou reprezentáciou objektov v 3D počítačovej grafike. Keďže tieto modely chceme analyzovať, upravovať, či inak spracovávať, je často vhodné ich zjednodušiť. Zjednodušenie sa dá chápať rôznymi spôsobmi a jedným z prístupov zjednodušovania je segmentácia. Segmentácia je proces rozdeľovania celku na jednoduchšie časti, alebo na časti, ktoré dávajú v kontexte väčší zmysel. Rôzne metódy segmentácie delia modely podľa rôznych kritérií a majú špecifické využitie.

Cieľom tejto práce je preštudovať problematiku a existujúce riešenia segmentácie polygonálnych modelov a implementovať nástroj používajúci niektorú zo segmentačných metód. Potom s implementáciou experimentovať a analyzovať dosiahnuté výsledky.

V rámci práce vznikla webová aplikácia pre interaktívnu segmentáciu, ktorá dokáže v krátkom čase na základe vstupu užívateľa rozdeliť model na dve časti. Metóda segmentácie, ktorá bola vo vytvorenej aplikácii implementovaná, vychádza z metódy popísanej v článku Interactive Mesh Segmentation Based on Feature Preserving Harmonic Field [11]. Metóda preberá niektoré myšlienky algoritmu, ale je implementovaná nanovo a prispôsobená požiadavkám aplikácie. Sementačný nástroj bol vytvorený formou interaktívnej webovej aplikácie, do ktorej používateľ nahrá svoj polygonálny model, načrtne oblasti, ktoré chce oddeliť a spustí segmentáciu. Segmentačná metóda ohodnotí hrany a trojuholníky vstupného modelu na základe vzdialeností a dihedrálnych uhlov, prevedie model na graf a vykoná minimálny rez grafom. Po úspešnej segmentácii si môže používateľ výsledok prezrieť, modely stiahnuť, či prípadne s jedným z nich pokračovať ďalšou segmentáciou.

V kapitole 2 je priblížená problematika segmentácie a aj popis graph-cutu. Kapitola 3 uvádza niektoré existujúce metódy segmentácie polygonálnych modelov. Nasledujúca kapitola 4 popisuje návrh segmentačnej aplikácie vytvorenej v tejto práci po stránke používateľského rozhrania, aj po algoritmickej stránke. Implementácia aplikácie, a technológie na to použité sú popísané v kapitole 5. Kapitola 6 prezentuje testovanie aplikácie, zhodnotenie jeho výsledkov a možnosti nadväzujúcej práce.

Kapitola 2

Segmentácia v počítačovej grafike

Segmentácia v počítačovej grafike má dlhú históriu, pôvodne išlo a aj dnes najčastejšie ide o segmentáciu obrazu. Spracovanie obrazu využíva segmentáciu na mnoho účelov, ako je rekonštrukcia obrazu, rozpoznávanie objektov v obraze a mnoho ďalšieho. Na segmentáciu obrazu sa používa veľmi veľa prístupov a existujú desiatky algoritmov, ktoré riešia tento problém, najjednoduchšie sú napríklad prahovanie, či histogramové metódy. Niektoré z algoritmov, využívané na segmentáciu obrazu sa neskôr preniesli aj do segmentácie polygonálnych modelov, ako napríklad clustering [14] [7], graph-cut [7], či metódy s rastom oblastí [14] [18].

Ako bolo povedané v úvode, segmentáciou sa chápe proces rozdeľovania dát (v tomto prípade 3D modelov) na jednoduchšie časti, alebo ich premena tak, aby dávali v kontexte väčší zmysel. V podstate ide o izolovanie pre nás relevantných dát.

2.1 Segmentácia polygonálnych modelov

Účel segmentácie polygonálnych modelov je veľmi podobný všeobecnému účelu segmentácie, a to získať z modelu jednoduchšie a v kontexte zaujímavejšie časti. Segmentácia modelov má široké využitie pri 3D modelovaní, analýze povrchových modelov získaných skenovaním, animácii, spracovaní modelov, kompresii a pri mnohých nástrojoch využívajúcich segmentáciu interne.

Na segmentáciu modelov sa využívajú rôzne prístupy [14] líšiace sa nie len postupom a zložitosťou, ale aj využitím. Preto je použitie vhodnej metódy kľúčovým prvkom procesu vytvárania segmentačných nástrojov. Poznáme dva základné prístupy k segmentácii polygonálnych modelov, a to: segmentácia na časti a segmentácia na povrchové oblasti (viď Obrázok 2.1). Segmentácia na povrchové oblasti hľadá oblasti, ktoré spolu súvisia z povrchového hľadiska. Výsledné oblasti nemusia nutne dávať ľudskému vnímaniu zmysel. Uplatnenie takejto segmentácie je najmä pri CAD aplikáciách, mapovaní textúr a podobne. Táto práca sa však zaoberá segmentáciou na časti, čím je myslené rozdelenie na časti podľa sémantického významu. Ďalším rozdielom v prístupe k segmentovaniu býva autonómnosť metód. Automatické metódy sa sústredia na analýzu modelu podľa vybranej heuristiky. Takéto segmentácie často generujú feature pointy na reprezentáciu výbežkov.

Interaktívne metódy potom využívajú nielen dáta obsiahnuté v obraze, či modeli, ale aj užívateľský vstup. Vstup býva väčšinou vo forme výberu bodov, alebo nakreslenia jednoduchých náčrtkov. Tieto dáta naviac umožňujú efektívnejšie a rýchlejšie dosiahnuť lepšie výsledky. Takúto segmentáciu však nemožno použiť v prípadoch, kedy nie je prítomný

používateľ, ktorý by s nástrojom interagoval. Interaktívne metódy sa často zakladajú na graph-cut algoritme, ktorý je popísaný v nasledujúcej kapitole.

Obr. 2.1: Modely zobrazujúce rôzne typy segmentácie, (naľavo) part-type, (napravo) patch-type, prebraté z [14].

2.2 Graph-cut

Myšlienka graph-cutu, teda rezu grafom, pochádza z teórie grafov. Teória grafov popisuje graf ako štruktúru tvorenú uzlami a hranami spájajúcimi tieto uzly. Matematicky je graf definovaný ako $G = (V, E)$, pričom V je množina uzlov a E je množina hrán, pre ktoré platí $E \subseteq \{\{x, y\} \mid (x, y) \in V^2 \bigwedge x \neq y\}$.

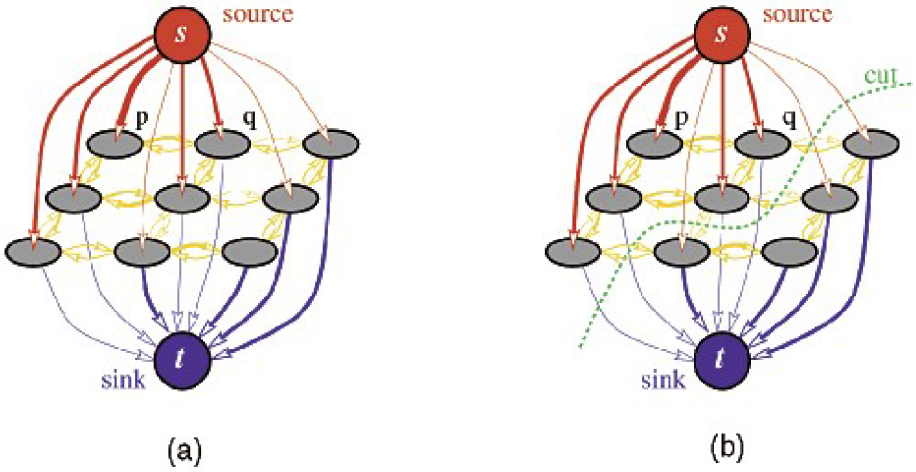
Takto definovaný graf je možné rozšíriť na váhovaný graf. Váhovaný graf má všetky hrany a uzly ohodnotené váhou, čo je reálna hodnota, reprezentujúca dôležitosť. Všetky grafy môžu byť chápané ako váhované s váhou hrán 1.

Grafová teória ďalej definuje orientované a smerované grafy. Orientovaný graf je taký, ktorý medzi každou dvojicou spojených uzlov obsahuje hrany v oboch smeroch. Smerovaný graf je potom orientovaný graf, ktorý pri niektorých dvojiciach uzlov obsahuje hranu iba v jednom smere.

Tokové siete (flow network) sú orientované grafy, obsahujúce dva špeciálne uzly - *source* a *sink*. Váhy sa pri tokových sieťach nazývajú kapacitami, sú nezáporné a značia hodnotu, ktorá môže maximálne cez hranu pretiecť [5]. Uzly nemajú kapacitu a okrem source a sink uzlov pre ne platí, že súčet tokov pritekajúcich do uzlu sa rovná súčtu tokov z uzlu odtekajúcich. Tok je funckia, ktorá priradzuje hranám nezápornú hodnotu.

Pri grafoch je možné urobiť rez. Rezom sa myslí operácia, ktorá odstráni množinu hrán tak, aby rozdelila graf na dve oddelené časti. Rez $C = (S, T)$ je rozdelenie množiny uzlov V z grafu $G = (V, E)$), do dvoch podmnožín S a T. Množina hrán rezu $C = (S, T)$ je množina $\{(u, v) \in E \mid u \in S, v \in T\}$, čiže množina takých hrán, ktorých jeden koncový bod patrí do S a druhý do T. Rezy grafom sa líšia podľa účelu. V počítačovej grafike, a špeciálne segmentácii je dôležitý minimálny rez (minimum cut), čiže taký rez, ktorý má najmenšiu hodnotu. Hodnota rezu pri hodnotenom grafe je súčet váh hrán, ktoré rez pretína.

Pri segmentácii sa potom graph-cut používa tak, že sa obraz, či model prekonvertujeme na graf. Pre počítanie minimálneho rezu sa používa funckia energie, ktorá má dve zložky



Obr. 2.2: (a) znázornenie tokového grafu G, (b) rez na grafe G, prebraté z [2]

energie a má spravidla tvar:

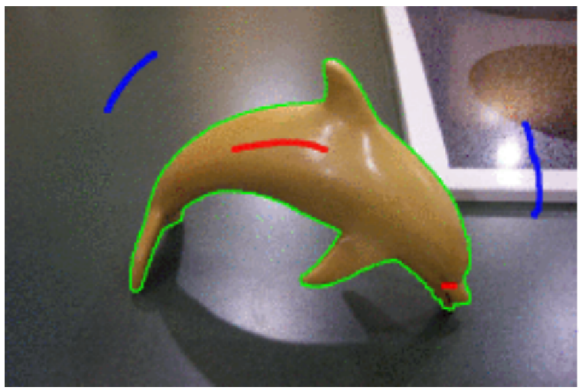
$$E(A) = \lambda \cdot R(A) + B(A) \tag{2.1}$$

Zložka energie $R(A)$ sa nazýva *region term* a vyjadruje penalizáciu za pridelenie uzlu do oblasti (pozadie/popredie) a zložka $B(A)$, označovaná ako *boundary term* vyjadruje penalizáciu za nesúvislosť susedných uzlov. Koeficient λ sa používa na vyváženie jednotlivých zložiek podľa potreby. Rozdelené grafy sa potom namapujú späť na pôvodné dáta a výsledkom je rozdelenie na dve, alebo viac častí.

Graph-cut v segmentácii obrazu

Segmentácia založená na graph-cute je v spracovaní obrazu veľmi populárna na riešenie binárnej segmentácie, čiže rozdelenie obrázku na pozadnie a popredie. Často sa používa uživateľský vstup zložený z náčrtkov dvoch farieb, ktoré označujú pozadie a popredie.

Príkladom poloautomatickej segmentácie, využívajúcej graph-cut je metóda spomínaná v [13]. Region term časť energie používa geodetickú vzdialenosť, zohľadňujúcu rozdiel farieb. Boundary term zase porovnáva intenzitu farieb susedných pixelov.



Obr. 2.3: Segmentácia obrazu, využívajúca geodetickú vzdialenosť a graph-cut, prebraté z [13]

Kapitola 3

Existujúce riešenia segmentácie polygonálnych modelov

Pri segmentácii polygonálnych modelov sa používajú dva základné pohľady – z geometrického alebo zo sémentického hľadiska, niekedy aj ich kombinácia. Metódy, ktoré k segmentácii pristupujú geometricky, segmentujú modely podľa nízkoúrovňových geometrických rysov, ako je napríklad krivosť. Sémantický prístup sa zase snaží analyzovať model a rozdeliť ho podľa významu, ktoré častiam prideľujú ľudia. Posledné roky sa rozvíjali algoritmy snažiace sa o sémantický prístup a snažili sa riešiť problémy, ktoré tu vznikajú. Hlavným problémom je získavanie vysokoúrovňovej predstavy tvaru modelu na základe geometrických rysov.

3.1 Feature Sensitive Mesh Segmentation

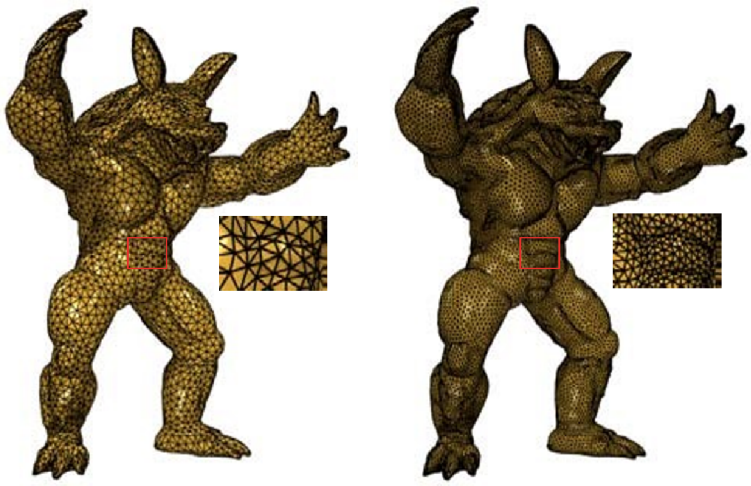
Táto metóda popísaná v článku [9] nadväzuje na klasické metódy prístupy k segmentácii, ako je hierarchická segmentácia, využívajúca k-means clustering, avšak prináša významné zlepšenia, ktoré sa prejavujú najmä v rýchlosti.

Obr. 3.1: Ukážka jednotlivých krokov algoritmu, prebraté z [9].

Algoritmus, ktorého niektoré úrovne možno vidieť v Obrázku 3.1 postupuje v nasledujúcich bodoch:

1. Predspracovanie (voliteľné)
2. Hierarchické premodelovanie citlivé na rysy
3. Hierarchická segmentácia segmentácia k-means clusteringom
4. Namapovanie výsledkov na pôvodný model (voliteľné)

5. Vyhladenie hraníc citlivé na rysy



Obr. 3.2: Armadillo model (pôvodne s 345,944 trojuholníkmi) premodelované s 11,756 (naľavo) a 47,024 trojuholníkmi (napravo), prebraté z [9].

Segmentačné algoritmy, založené na clusteringu používajú vzdialenosti medzi pármi trojuholníkov. Keďže je k týmto vzdialenostiam nutné počas k-means clusteringu pristupovať, je potrebné ich mať uložené v pamäti. Pre veľké modely je takýto prístup pamäťovo aj časovo náročný. Preto táto metóda vytvára 1-3 jednoduchšie modely, vždy približne o 1/4 počtu trojuholníkov, oproti predošlému. Namiesto obyčajného zjednodušenia modelu (mesh simplification), používa izotropné, rysy zachovávajúce premodelovanie (Obrázok 3.2), popísané v [8].

Po premodelovaní sa spustí samotná hierarchická segmentácia používajúca k-means clustering, ktorá postupuje krokmi:

1. Najprv sa vždy prepočítajú vzdialenosti medzi trojuholníkmi (metrika vzdialenosti zohľadňuje geodetickú vzdialenosť, krivosť a textúru).

2. Vyberú sa počiatočné body, buď na základe používateľského vstupu, alebo sa vyberie jeden náhodný a ostatné sa určia podľa najväčšej priemernej vzdialenosti od predošlých počiatočných bodov.

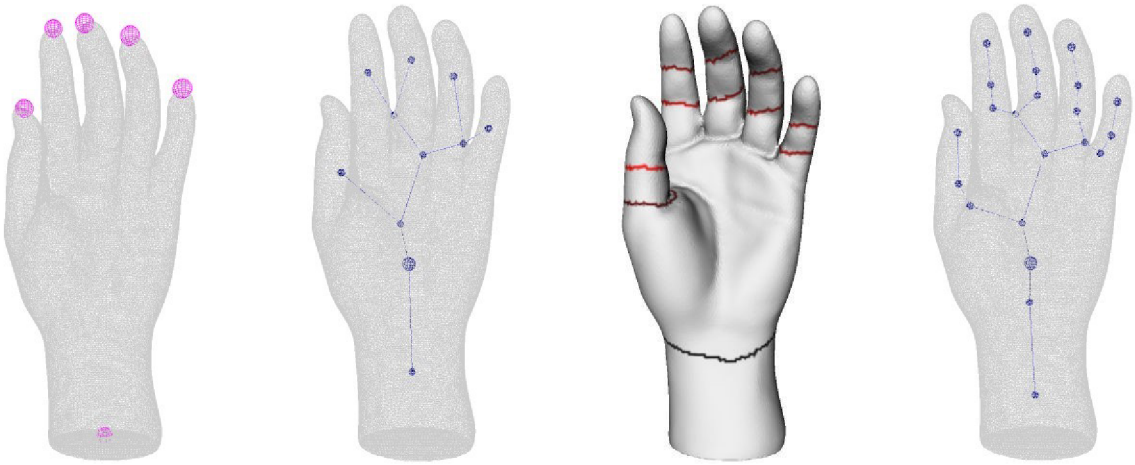
3. Trojuholníky sa priradia k oblasti s najbližším zástupcom.

4. Prepočítajú sa zástupcovia oblastí.

3.2 Topology Driven 3D Mesh Hierarchical Segmentation

Táto metóda, narozdiel od predošlej metódy, využíva viac sémantický prístup. Myšlienkou metódy je vytvorenie kostry modelu na základe feature bodov a následnej segmentácie, podľa vytvorenej kostry.

Najprv sa vytvorí kostra modelu, podľa prístupu z článku [17]. Najprv sa extrahujú feature body, potom je pre každý vrchol definovaná mapovacia funkcia f_m, používajúca geodetickú vzdialenosť. Potom je pre každý vrchol vypočítaná disktétna kontúra, ktorá



(a) (b) (c) (d)

Obr. 3.3: Postup skeletonizácie, prebraté z [16]

aproximuje kontúru f_m. Po pokrytí celého modelu diskrétnymi kontúrami je možné analyzovať charakteristiky kontúr, buď na detekciu topologických zmien a vytvorenie kostry modelu(Obrázok 3.3b), alebo na detekciu zmien krivosti, čo určí hranice v modeli (Obrázok 3.3c).

(a) zobrazenie grafu kostry (b) prvotná segmentácia (c) zjednodušený graf (d) finálna segmentácia

Obr. 3.4: Postup segmentačného algoritmu, prebraté z [16]

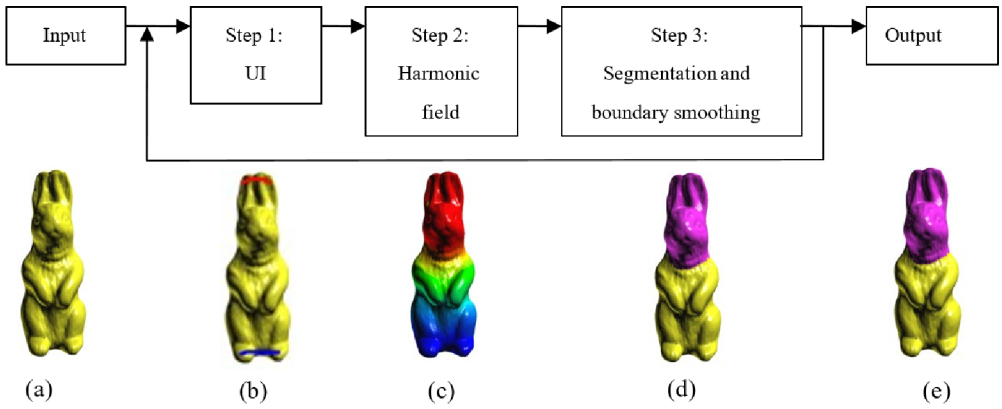
Graf, ktorý tvorí kostra modelu sa zjednoduší, tj. blízke uzly sa pospájajú a vytvoria sa tri stupne uzlov, ktoré označujú oblasti podľa vzdialenosti od jadra. Ak má uzol stupeň 1 je to koncový uzol, uzly so stupňom 2 sú tubulárne uzly a uzly so stupňom väčším ako 2 sú spojovacie uzly, spojovacie uzly sa zoskupia do spojovacích oblastí. Pri segmentovaní potom záleží na tom, ku ktorej oblasti trojuholník patrí. Je taktiež možné vykonať hierarchickú segmentáciu, kedy sa prechádza od jadra ku koncovým uzlom a koncové uzly sa rozdeľujú podľa hraníc (Obrázok 3.4c).

Výhodou tejto metódy je, že veľmi nezáleží na deformácii modelu, počte trojuholníkov, či šume, naopak, záleží hlavne na tvare modelu. Najlepšie výsledky metóda dosahuje pri modeloch s vyčnievajúcimi časťami.

3.3 Interactive Mesh Segmentation Based on Feature Preserving Harmonic Field

Predošlé metódy boli automatické, čím vznikla potreba odhadnúť zámer segmentácie. Prístupy riešenia tohto problému sú rôzne, napríklad generovanie feature bodov, či kostry modelu ako v 3.2 . Táto metóda však využíva vstup používateľa, v podobe náčrtkov oblastí, na špecifikáciu jeho zámeru. Toto dovoľuje metóde sústrediť sa aj na geometrické rysy aj na príslušnosť oblasti k náčrtku. Algoritmus sa riadi nasledujúcimi krokmi:

1. Používateľ nakreslí na model náčrtky pozadia a popredia.
2. Na základe náčrtkov sa vygeneruje harmonické pole zachovávajúce rysy.
3. Aplikuje sa graph-cut a vyhladenie hraníc.



Obr. 3.5: Postup segmentačnej metódy, prebraté z [11].

Na základe používateľského vstupu sa vypočíta harmonické pole, ktoré odráža geometrické rysy (rozdiely normál, zmeny krivosti, uhly medzi trojuholníkmi).

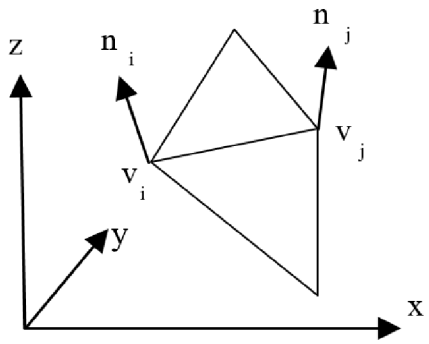
Na vytvorenie skalárnej funkcie $u\colon V \to R$, ktorá priraďuje hodnotu každému vrcholu v článku uvažujú, že $\Delta u = \nabla^2 u = f$ je ustálená eliptická rovnica, kde Δ je Laplace-Beltramiho operátor. Pri trojuholníkovom modeli je diskrétna forma Laplacovho operátoru:

$$\Delta u_i = \sum_{j \in N_i} w_{ij}(u_i - u_j) \tag{3.1}$$

Kde N_i je jednookolie vrcholu vi a wij je skalárna váha hrany (i,j), váhovanie je nasledovné:

$$w_{ij} = k w_{ij}^n + (1-k) w_{ij}^c \tag{3.2}$$

kde $w_{ij}^n = (1+\frac{\alpha_{ij}}{avg(\alpha_{(ij)}}))^{-1}$, α_{ij} je uhol medzi normálami ni a nj vrcholov vi a vj (znázornené v obrázku 7), $w_{ij}^c = \frac{1}{\sqrt{(|f_n|+1)}}$, $f_n = 2\frac{((v_i - v_j)n_i)}{\|v_i - v_j|}^2$ je aproximácia normálovej krivosti v smere



Obr. 3.6: Vizualizovanie pozícií trojuholníkov, vrcholov a normál, prebraté z [11].

hrany (i, j), k je konštanta medzi 0 a 1, ktorú vyberá používateľ. Pre modely s jasnými hranami sa používa malá hodnota a pre vyhladenejšie modely väčšia.

Riešenie je potom dosiahnuté minimalizáciou energie funkcie:

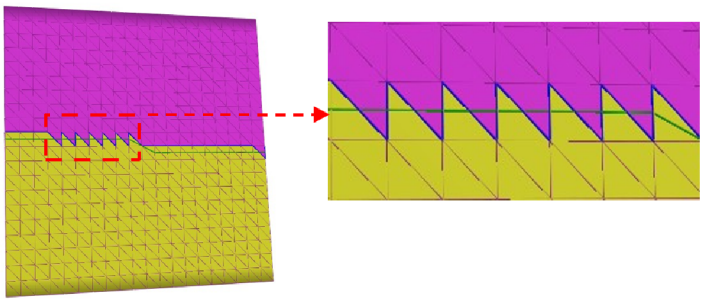
$$E(X) = \sum_{v_i \in V} E_1(x_i) + \lambda \sum_{(v_i, v_j) \in E} \tag{3.3}$$

Jednotlivé zložky funkcie sú potom definované ako:

$$E_1 = \begin{cases} u_i, x_i = 1 \\ 1 - u_i, x_i = 0 \end{cases} \tag{3.4}$$

$$E_2(x_i, x_j) = \frac{1}{\sqrt{\mid v_i - v_j \mid^2 + \beta \mid n_i - n_j \mid^2}} \tag{3.5}$$

Kde v_i,v_j,n_i,n_j sú pozície vrcholov a normálové vektory vrcholov v_i a v_j ,β je upraviteľný parameter, v článku bola použité hodnota 6. Parameter λ vo funkcii 3.3 značí relatívnu dôležitosť zložiek a môže byť upravený používateľom na dosiahnutie čo najlepšej segmentácie. Zložka E_1 značí penalizáciu za nesprávne priradenie vrcholu a E_2 značí súčet cien hrán pozdĺž hranice. Po rozdelení modelu graph-cutom na dve časti sa aplikuje vyhladenie. Na počítanie vyhladenia a rozdelenia trojuholníkov sa používajú dve funkcie energie, v ktorých sa používa harmonické pole z prvej časti algoritmu, dihedrálne uhly a dĺžky hrán.



Obr. 3.7: Pílkovité hranice a následné rozdelenie trojuholníkov, prebraté z [11].

3.4 Nástroje na spracovanie polygonálnych modelov

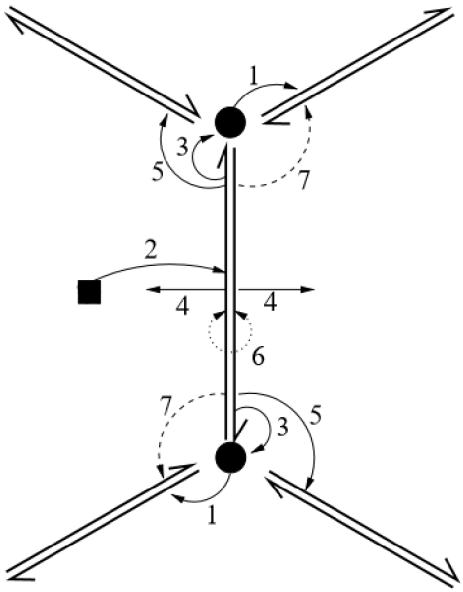
Známe nástroje na výrobu modelov, spracovanie, či úpravu modelov, sú napríklad Blender[1], alebo Meshlab[2]. Tieto nástroje sa zameriavajú na profesionálnu prácu s modelmi. Pre vývojárov sú potom dostupné knižnice, ktoré obsahujú štruktúry a geometrické algoritmy pre spracovanie 3D modelov.

OpenMesh

Openmesh[3] je C++ knižnica popísaná v článku [1], slúžiaca na manipuláciu s polygonálnymi modelmi. Využíva halfegde štruktúru, ktorá umožňuje efektívnu reprezentáciu polygonálnych modelov. Halfedge štruktúra je vytvorená rozdelením hrany na dve orientované časti. Halfedge obsahuje nasledujúce informácie o okolí:

- Jeden vrchol
- Jednu stranu
- Ďalšia halfedge
- Opačná halfedge
- Voliteľne aj predošlá halfedge

Táto štruktúra teda značne zjednodušuje prácu s modelom, hlavne orientáciu na modeli, dovoľuje jednoducho cirkulovať po prvkoch (hrany, trojuholníky, vrcholy), a vždy mať dostupné informácie o pozícii a okolí.



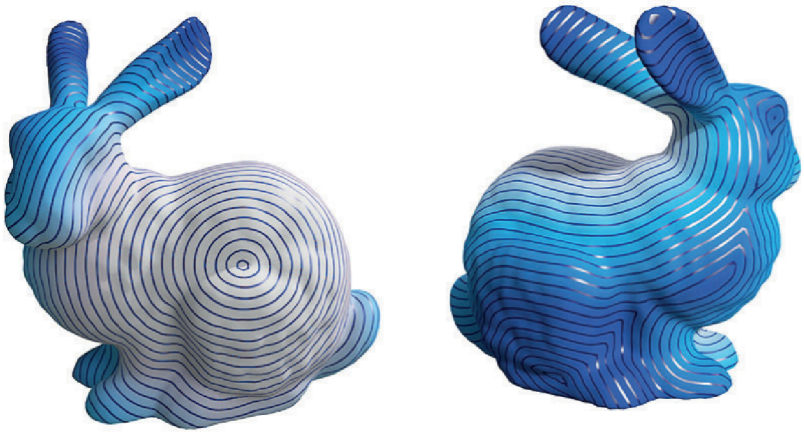
Obr. 3.8: Vizualizácia halfedge štruktúry, čísla označujú časti halfedge: 3 vrchol, 4 strana (face), 5 nasledujúca halfedge, 6 opačná halfedge, 7 predošlá hafedge, prebraté z [1].

[1] https://www.blender.org
[2] http://www.meshlab.net
[3] https://www.openmesh.org

CGAL

CGAL[4] (The Computational Geometry Algorithms Library) je Open source C++ knižnica poskytujúca ekeftívne geometrické algoritmy. Jedným z nástrojov, ktoré táto knižnica poskytuje je aj nástroj na segmentáciu polygonálnych modelov. Nástroj v tejto knižnici implementuje algoritmus, využívajúci *shape diameter function*, ktorá hodnotí plochy modelu na základe zmeny priemeru [15].

Od CGAL verzie 4.14 je v knižnici dostupný nástroj Heat method, ktorý slúži na aproximáciu geodetickej vzdialenosti bodu, alebo rozmedzia bodov ku všetkým vrcholom modelu. Heat method je implementovaná podľa článku [4].



Obr. 3.9: Vizualizácia výsledku Heat method, prebraté z [4]

[4] https://www.cgal.org/index.html

Kapitola 4

Návrh webovej aplikácie na interaktívnu segmentáciu 3D modelu

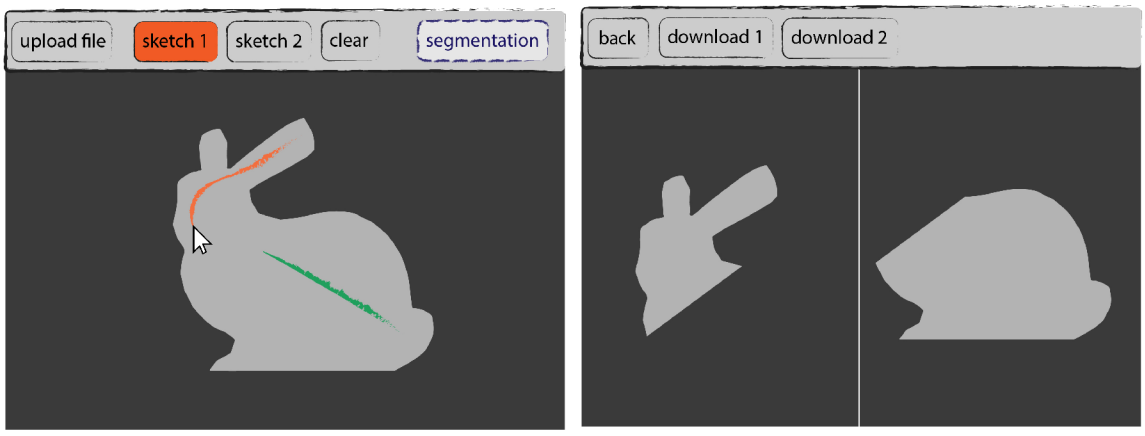
Počas študovania problematiky segmentácie 3D modelov som sa oboznámil s problémami spájajúcimi sa s automatickými metódami segmentácie, ako je napríklad náročný odhad ľudského vnímania, či časovo náročné výpočty. S prihliadnutím na tieto problémy a inšpirovaný metódou z kapitoly 3.3 som sa rozhodol vytvoriť nástroj na interaktívnu segmentáciu polygonálnych modelov.

Nástroj vytváram s cieľom rýchlej a dostupnej segmentácia, čo ma priviedlo k nápadu vytvoriť webovú aplikáciu. V takejto webovej aplikácii by mohol používateľ interaktívne rozdeliť model v čo najkratšom čase a výsledok si stiahnuť. Predlohou segmentačnej metódy bola metóda Interactive Mesh Segmentation Based on Feature Preserving Harmonic Field, ktorá je bližšie popísaná v predchádzajúcej kapitole.

4.1 Návrh používateľského rozhrania

Hlavným účelom tejto aplikácie je schopnosť segmentovať modely, a preto bude používateľské rozhranie čo najviac zamerané na tento cieľ. Pri vytváraní grafického používateľského rozhrania som sa riadil iteratívnym prístupom. Najprv som vymyslel veľmi jednoduché rozhranie, so základnými prvkami nutnými k funkčnosti aplikácie, pričom som sledoval pri používaní aplikácie nedostatky, zbytočné prvky a možnosti na zlepšenie kvality používateľského rozhrania. Tieto poznatky som potom použil pri úpravách používateľského rozhrania, ktoré sú popísané v sekcii 6.1.

V prvotnom náčrte používateľského rozhrania sú dve hlavné časti, a to menu a plocha pre zobrazenie modelu (Obrázok 4.1a). V návrhu sa v menu nachádzajú ovládacie prvky na načítanie modelu, prepínanie náčrtov, mazanie náčrtov a tlačidlo na spustenie segmentácie. Na zobrazovacej ploche je možné manipulovať s modelom a robiť náčrtky pre pozadie a popredie. Na manipuláciu s modelom sa používa najmä myš, držaním tlačidiel myši možno model otáčať, či posúvať a kolieskom myši približovať pohľad. V obrázku 4.1b je druhá časť rozhrania, ktorá sa objaví po úspešnej segmentácii. Rozhranie obsahuje dve zobrazovacie plochy pre modely, v menu sú prvky na stiahnutie výsledných modelov a možnosti vrátenia sa späť.



(a) Návrh rozrhania pred segmentáciou (b) Návrh rozrhania po segmentácii

Obr. 4.1: Prvotný návrh grafického používateľského rozhrania.

4.2 Popis algoritmického riešenia

Kostra algoritmu vychádza z pôvodnej metódy, a jeho jednotlivé kroky sú:

1. Načítanie modelu
2. Ručná anotácia formou náčrtkov na modeli
3. Hodnotiaca funkcia
4. Transformácia modelu na graf a vykonanie graph-cutu
5. Namapovanie výsledku na pôvodný model a výstup

Už z kostry je zrejmé, že metóda je zjednodušená a je odstránený postprocesing vo forme vyhladenia hraníc. Vyhladenie hraníc bolo z metódy odobrané hlavne kvôli zjednodušeniu algoritmu a je možné ho v budúcnosti pridať.

Vstup

Vstup algoritmu je trojuholníkový polygonálny model a užívateľský vstup. Používateľský vstup obsahuje náčrtky vytvorené ručnou anotáciou, ktoré sú reprezentované dvomi množinami bodov, a tiež parameter, ktorý ovplyvňuje relatívnu dôležitosť region – boundary termov.

Hodnotiaca funckia

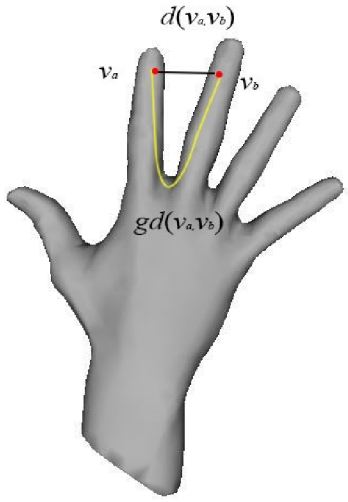
Po načítaní modelu sú na model aplikované hodnotiace funkcie, jedna hodnotiaca funkcia pre trojuholníky a jedna pre hrany modelu. Trojuholníky (faces) sú hodnotené funkciou, ktorú som navrhol, aby čo najlepšie odrážala penalizáciu za vzdialenosť, pričom je normalizovaná priemernou vzdialenosťou. Funckia má nasledovný tvar:

$$R_f = -ln(1 - e^{\frac{d_f}{avg(d_f)}}) \tag{4.1}$$

kde d je priemer vzdialeností vrcholov hodnoteného trojuholníku f k obom vstupným náčrtom. Funkcia hodnotí príslušnosť trojuholníku f k oblasti (source/sink), preto sa používa dvakrát pre každý trujuholník. Hodnotiaca funkcia využíva buď euklidovskú, alebo geodetickú vzdialenosť, podľa používateľského výberu. Euklidovská vzdialenosť je vzdialenosť vzdušnou čiarou ((d) v obrázku 4.2) a pre body vi,vj počíta sa ako:

$$d(v_i, v_j) = \sqrt{(v_{i1} - v_{j1})^2 + (v_{i2} - v_{j2})^2 + (v_{i3} - v_{j3})^2} \tag{4.2}$$

Geodetická vzdialenosť je povrchová vzdialenosť ((gd) v obrázku 4.2) medzi bodmi, počítaná ako najmenší súčet euklidovských vzdialeností medzi hranami, spájajúcimi dané body.



Obr. 4.2: Porovnanie geodetickej a euklidovskej vzdialenosti pri modeli ruky, prebraté z [10]

Pre hrany je použitá hodnotiaca funkcia:

$$B_e = -ln(\alpha_e) \tag{4.3}$$

Kde α je normalizovaný dihedrálny uhol medzi trojuholníkmi, ktoré spája hodnotená hrana e. Hrana e môže byť reprezentovaná aj susediacimi trojuholníkmi f_i,f_j , ktoré hranu zvierajú. Uhol je normalizovaný funkciou:

$$\alpha = \frac{\mid ((\alpha + 180) \cdot \mod 360) - 180 \mid}{180} \tag{4.4}$$

Takto normalizované uhly je ešte možné váhovať, podľa ich konvexnosti, či konkávnosti. V mojom riešení konvexné uhly násobím konštantou 0.1, aby som znížil ich relevantnosť, ale podľa potreby je možné túto hodnotu zmeniť, napríklad pri CAD modeloch by bola lepšia väčšia hodnota.

Celková energia má potom podobne ako v [10] tvar :

$$E(L) = \lambda \sum_{p \in F} R_p(L_p) + \sum_{e \in H} B_e(L_e) \tag{4.5}$$

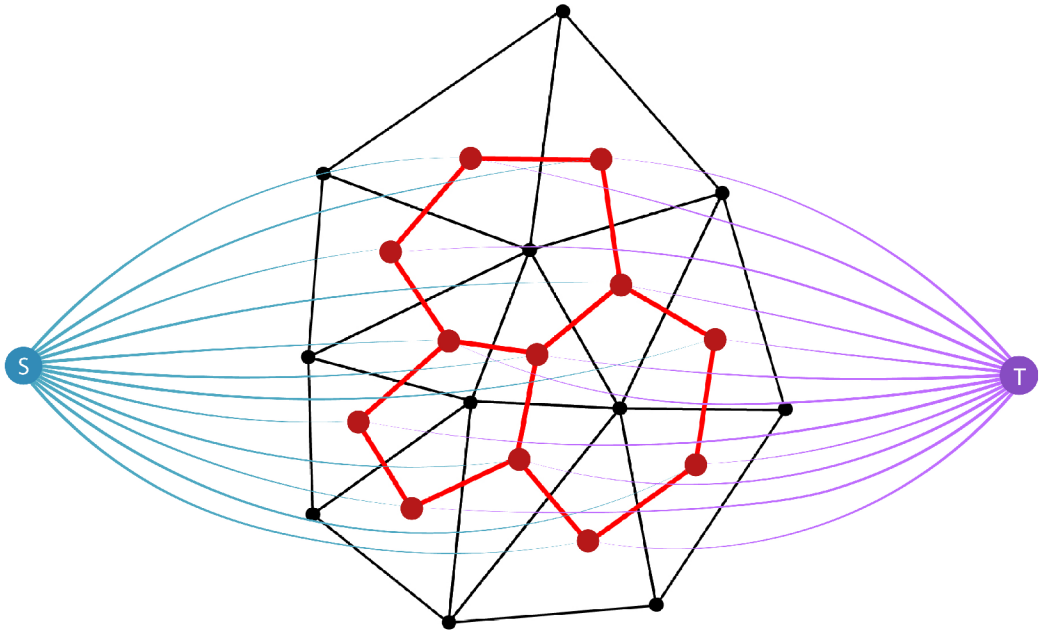
Kde λ je parameter, ktorý vyjadruje relevantnú dôležitosť jednotlivých zložiek, R_p je region term zložka energie, B_e je boundary term zložka energie, H je množina hrán medzi pármi susediacich trojuholníkov, L_p je označenie priradenia oblasti trojuholníku p, a

F je množina všetkých trojuholníkov modelu. Podobne, ako bolo uvedené v 2.2, region term zložka energie vyjadruje penalizáciu za priradenie uzlu do oblasti a boundary term reprezentuje podobnosť susedných uzlov grafu.

Od parametru λ potom záleží, či sa bude segmentácia prikláňať k rozdeľovaniu modelu podľa vzdialenosti od vstupov, alebo podľa zmien v uhloch medzi trojuholníkmi. Východisková hodnota tohto parametru je 0.12, no je možné, aby ju používateľ zmenil podľa potreby a dosiahol tak požadovaný výsledok. Pre modely s jasne odlíšiteľnými časťami sa hodia menšie hodnoty, pretože sa tak dosiahnú presnejšie hranice, naopak pre modely s vyhladenejším povrchom je vhodná veľká hodnota, aby sa model vôbec rozdelil.

Prevod modelu na graf

Polygonálne modely sú bežne reprezentované formou dátových štruktúr, ako napríklad *half-fedge*, ktorá je bližšie popísaná v 3.4. Takáto reprezentácia je vhodná pre geometrické algoritmy, no pre využitie graph-cut algoritmu je nutné model previesť na graf, konkrétne na tokovú sieť, popísanú v 2.2. Trojuholníky sú transformované na uzly a každý uzol je napojený na source/sink uzly takzvanými t-hranami, ktorých váhy reprezentujú region term zložku energie. Hrany medzi priľahlými trojuholníkmi sa zmenia na hrany v grafe (pre predstavu vizualizované na obrázku 4.3). Prevod prebieha iteráciou po trojuholníkoch a hranách modelu, ktoré sa pridávajú do vznikajúceho grafu. Bolo by možné prevod robiť aj počas jednej iterácie po hranách, čo by však ovplyvnilo konzistenciu poradia hrán grafu a modelu. Následne sú hranám priradené kapacity, vypočítané hodnotiacimi funkciami.



Obr. 4.3: Znázornenie prevodu časti modelu na graf, povrch modelu je znázornený čiernou farbou a vytvorený graf červenou farbou. Červené hrany predstavujú *boundary term*, modrou a fialovou farbou sú vyobrazené uzly *source* a *sink* a ich napojenia, ktoré predstavujú *region term*.

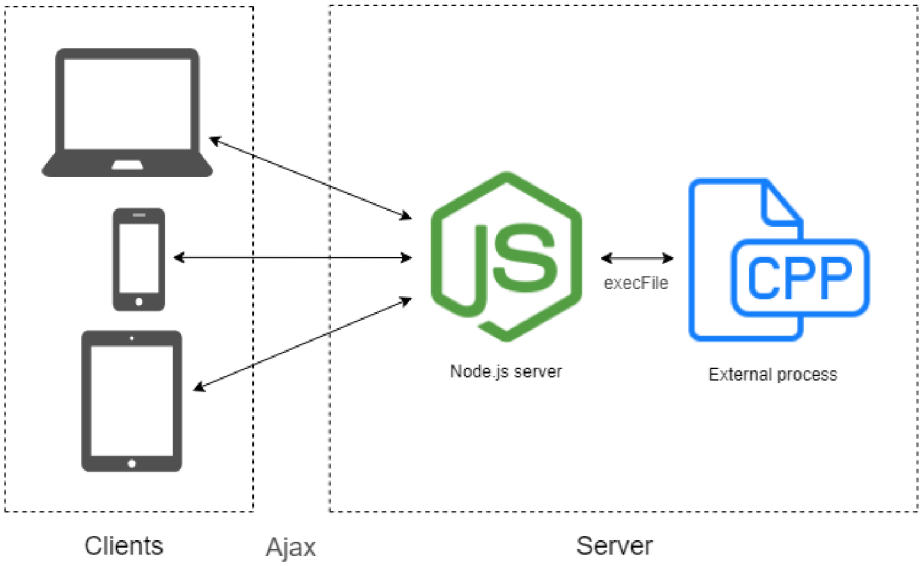
Segmentácia

Samotná segmentácia už spočíva len v reze grafom. Na tento účel slúžia algoritmy, hľadajúce minimálny rez spomínaný v 2.2. Toto riešenie používa Boykov-Kolmogorovov algoritmus [2], hlavne kvôli jeho rýchlosti, vďaka ktorej dnes ide o najpoužívanejší algortimus pre hľadanie minimálneho rezu. Po segmentácii sa časti grafu namapujú na pôvodný model a vzniknú tak dva rozdelené modely.

Kapitola 5

Implementácia

Keďže bol segmentačný nástroj implementovaný ako webová aplikácia, bola kvôli jednoduchosti použitá architektúra klient server. Používateľské rozhranie je zobrazené v internetovom prehliadači na strane klienta. Po načítaní modelu a nakreslení vstupu používateľ spustí segmentáciu. Samotná segmentácia je vykonávaná programom implementovanom v jazyku C++. Ten je preložený na spustiteľný program, ktorý je potom spúšťaný funkciou v Node.js. Programu sú predané parametre – dve množiny bodov, reprezentujúce náčrtky na modeli, samotný model, parameter λ (viď 4.2) a metóda, ktorou sa má vyhodnocovať vzdialenosť. Po segmentácii program vráti klientovi výsledné modely v odpovedi požiadavku a klient ich zobrazí.



Obr. 5.1: Diagram architektúry aplikácie.

5.1 Klientská časť

Klientská časť aplikácie je vytvorená v HTML, CSS a JavaScripte. Táto časť aplikácie obsahuje používateľské rozhranie zobrazované webovým prehliadačom a slúži na interakciu medzi aplikáciou a používateľom. Na zjednodušenie niektorých úkonov, ako napríklad výber elementov, prácu s používateľským rozhraním, či vytváranie ajax požiadavkov používam

javascriptovú knižnicu JQuery[1]. Komunikácia je iniciovaná zo strany klienta a je riešená v podobe Ajax (Asynchronous JavaScript and XML) požiadaviek.

Three.js

Three.js[2] je knižnica, slúžiacia na zobrazovanie 3D obsahu vo webových prehliadačoch. Je napísaná v jazyku Javscript a využíva aplikačné rozhranie WebGL. Hlavné prvky, potrebné na zobrazenie modelu sú scéna, kamera a renderer. V scéne sú uložené objekty, ktoré chceme zobraziť – napríklad náš model, zdroje svetla a podobne. Kamera potom reprezentuje oblasť, na ktorú sa pozeráme. Renderer vykresľuje scénu z pohľadu kamery, vykresľovanie je volané v slučke v animačnej funkcii.

V Three.js bol implementovaný nástroj na kreslenie náčtrkov, repretentujúcich pozadie a popredie. Pri implementácii kreslenia po modeli bol využitý *raycaster*, ktorý je obsiahnutý v Three.js. Vytvárajú sa body určené priesečníkom mezdi lúčmi *raycasteru* a povrchom modelu. Vypočítané body sú potom pridávané do scény ako úsečky. Tieto úsečky nie sú zobrazené klasickou THREE.Line, ale externou náhradou THREE.MeshLine[3]. MeshLine vyvažuje nedostatky obyčajných úsečok v Three.js, ktoré napríklad nemôžu na všetkých platformách meniť šírku čiary.

Práca s modelmi v Three.js sa vyznačuje aj pamäťovou náročnosťou, preto je nutné správne pracovať s pamäťou. Javascript používa garbage collector, čo je nástroj na dealokáciu nepotrebnej pamäte. V princípe garbage collector uvoľní pamäť, na ktorú už neexistuje referencia, no v Three.js sa často stáva, že nám ostanú referencie v renderer objekte, scéne a podobne. Preto používam metódu dispose, obsiahnutú v Three.js, ktorá slúži na odstraňovanie referencií na špecifické objekty.

5.2 Serverová časť

Serverová časť aplikácie by sa dala rozdeliť na dve časti, prvou je webový server, starajúci sa o komunikáciu s klientskou časťou a druhou je externý program, v ktorom je vykonávaná segmentácia.

Prvou časťou je teda webový server a ako je zvykom pri klient-server architektúre, server je pasívnym prvkom, ktorý čaká na požiadavky klienta. Po prijatí požiadavky server reaguje príslušnou akciou, napríklad vrátenie webovej stránky. Server je riešený ako Node.js aplikácia s využitím Express frameworku.

Druhá časť, čiže segmentačný program je implementovaný v jazyku C++. Na načítanie vstupného modelu, vytváranie výstupných modelov a internú reprezentáciu modelu bola použitá knižnica Openmesh. Knižnica CGAL je využitá na niektoré výpočty, ako napríklad výpočet vzdialeností, či dihedrálnych uhlov, keďže sú v nej implementované príslušné metódy. Na spracovanie používateľského vstupu vo formáte JSON je použitá voľne dostupná pomocná knižnica *nlohmann_json*.

OpenMesh

OpenMesh knižnicu, spomínanú v 3.4 používam na spracovanie vstupu a výstupu C++ aplikácie, pretože CGAL nepodporuje načítavanie modelov vo formátoch ako OBJ. Open-

[1] https://jquery.com
[2] https://threejs.org
[3] https://github.com/spite/THREE.MeshLine

Mesh obsahuje triedu *IOManager*, umožňujúcu načítavanie a zapisovanie modelov v rôznych formátoch. Konkrétne podporuje formáty OFF, OBJ, PLY, STL a OM.

CGAL

Po načítaní modelu pomocou OpenMesh knižnice je model reprezentovaný ako *Polyhedron*[4]. CGAL potom využívam na počítanie dihedrálnych uhlov funkciu *approximate_dihedral_angle()*, ktorá vypočíta normalizovaný dihedrálny uhol. Na počítanie geodetickej vzdialenosti používam Heat method, spomínanú v 3.4. Výhodou tejto metódy je hlavne jej rýchlosť, predtým ako som používal túto metódu mi počítanie geodetickej vzdialenosti trvalo mnohonásobne dlhšie. Treba však brať do úvahy, že pri modeloch s veľkým počtom trojuholníkov časová náročnosť aj tak závratne rastie. Počítanie geodetickej vzdialenosti pomocou *heat method* je možné dvomi spôsobmi: buď si vytvoriť inštanciu triedy, ktorej pridelíme množinu bodov a následne metódou spočítame vzdialenosť, alebo použiť verejnú metódu - tento prístup však vyžaduje knižnicu Eigen[5], aspoň verziu 3.3. Ja som sa rozhodol pre druhú možnosť, hlavne kvôli kompatibilite reprezentácie modelu.

Maxflow

Maxflow[6] je knižnica implementujúca Boykov-Kolmogorovov graph-cut algoritmus popísaný v [2]. Obsahuje jednduché rozhranie na vytváranie grafu pridávaním uzlov – *add_node()*, hrán – *add_edge()* a váh hranám – *add_tweights()*. Potom je možné metódou *maxflow()* vypočítať maximálny tok/minimálny rez. Takto rozdelený graf sa potom dá jednoducho namapovať na pôvodný model.

Node.js

Node.js[7] je javascriptové prostredie postavené na Chrome V8 JavaScript engine. Hlavným účelom Node je vytváranie škálovateľných webových aplikácií, pričom v Node sa tvorí serverová časť. Node server je natívne konkurentný, tj. dokáže obsluhovať viac používateľov naraz. Node je riadený asynchrónnymi udalosťami a aktívne beží iba pri obsluhe požiadaviek, narozdiel od iných technológií, ktoré vyťažujú zdroje neustále.

Pre rozšírenie funkcionality Node sa používajú moduly, dostupné na inštaláciu pomocou *npm*[8] (node package manager). *Npm* je predvoleným správcom balíčkov pre Node a momentálne aj najrozsiahlejším adresárom balíčkov na svete.

Na veľkú časť implementácie som využil *Express.js*[9], čo je Node framework pre webové aplikácie. Jednou z vecí, ktoré Express umožňuje je osobitné spracovanie HTTP požiadaviek. Pomocou funkcií post a get je možné jednoducho riadiť smerovanie, teda vykonávať rôzne akcie podľa HTTP metódy a URI. V obsluhe požiadaviek je niekedy nutné spracovávať odpovede, na čo je taktiež možné použiť spomínaný framework. Express tiež zjednodušuje sťahovanie súborov, obsahuje totiž pomocnú metódu *res.download()*, ktorá klientovi spustí sťahovanie súboru s nastaveným menom. Pomocou Express teda riadim celú komunikáciu medzi klientom a serverom z pohľadu serveru.

[4] https://doc.cgal.org/latest/Polyhedron/classCGAL_1_1Polyhedron__3.html
[5] http://eigen.tuxfamily.org/index.php?title=Main_Page
[6] https://github.com/gerddie/maxflow
[7] https://nodejs.org/en/
[8] https://www.npmjs.com
[9] https://expressjs.com

Pracujem tiež so súborovým systémom, na túto prácu je vhodný modul *fs* (file system). Všetky funkcie tohto modulu sú dostupné v synchrónnej aj asynchrónnej forme.

Keďže je segmentačná metóda napísaná ako C++ aplikácia, je potrebné ju explicitne spúšťať, na túto úlohu používam modul *child_process*. Modul poskytuje širokú škálu funkcií na spúšťanie procesov, v synchrónnej aj asynchrónnej forme. Obsahuje aj funckiu *fork()*, slúžiacu na vytvorenie nového Node procesu. Segmentačnú aplikáciu spúšťam funkciou *execFile()*, takto sa vytvorí nový proces, ktorý po skončení vráti výstup späť do funkcie.

Kapitola 6

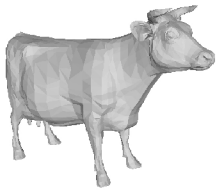
Experimentovanie

Na experimenty a testovanie bola použitá testovacia sada pozostávajúca z nasledujúcich 12 modelov. Do sady boli vybrané modely rôznych zložitostí a tvarov a jedná sa prevažne o modely zvierat, či postáv, teda modely ideálne sa hodiace pre vytvorenú aplikáciu.

(a) gourd 648 faces

(b) teddy 3192 faces

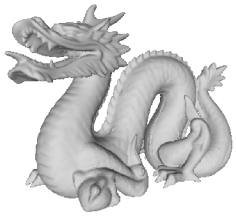
(c) bunny 4968 faces



(d) cow 5804 faces

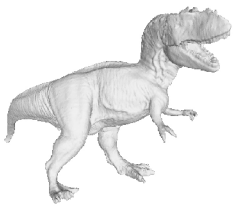
(e) big-bunny 69666 faces

(f) hand 72958 faces



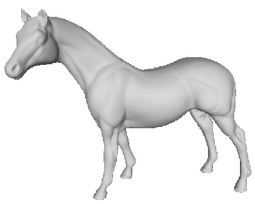
(g) dragon 100000 faces

(h) rabbit 141312 faces



(i) t-rex 200000 faces

(j) armadillo 212574 faces



(k) horse 225280 faces

(l) angel 448880 faces

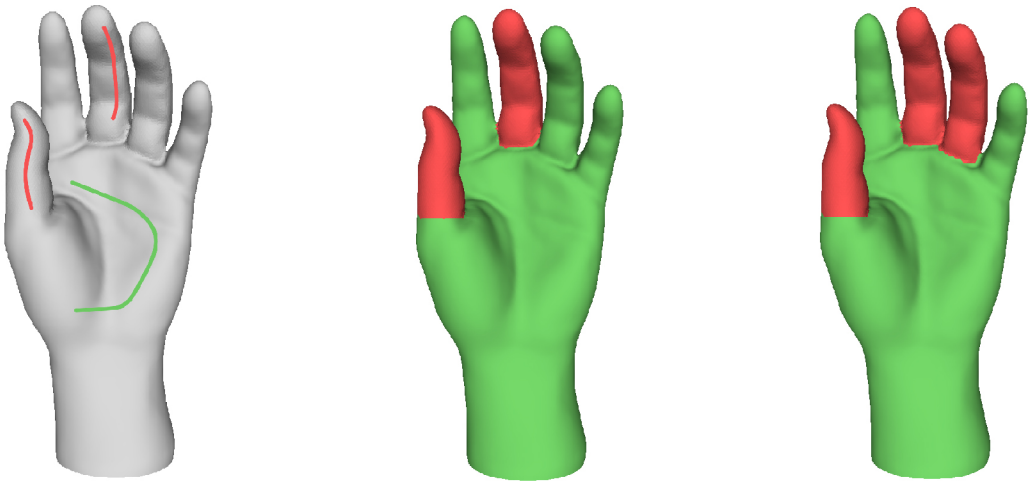
Obr. 6.1: Testovacia sada modelov

Počas implementácie som robil experimenty s rôznymi druhmi modelov, z ktorých niektoré nie sú zahrnuté v testovacej sade. Toto testovanie odhalilo niektoré nedostatky vytvorenej aplikácie. Ideálne aplikácia funguje s modelmi, ktoré vznikli skenovaním reálnych objektov, pretože tak ide o spojitý povrch jedného objektu. Modely vytvárané ručne pomocou modleovacieho softvéru sú často, aj keď to na prvý pohľad nie je zrejmé, zložené z viacerých oddelených častí, ako na obrázku 6.2. Spôsobuje to problémy pri výpočte geodetickej vzdialenosti, a to kvôli nesprávnym prepojeniam častí modelu, keďže geodetická vzdialenosť je počítaná po spojitej ceste. Čiastočným riešením je využitie euklidovskej vzdialenosti, na ktorú nemá vplyv oddelenie častí.

(a) Používateľský vstup (b) Využitá geodetická vzdialenosť (c) Využitá euklidovská vzdialenosť

Obr. 6.2: Ukážka výsledku aplikácie pre model opičej hlavy (suzanne), oči sú v tomto modeli oddelená časť

V obrázku 6.3 je porovnaná segmentácia s využitím geodetickej vzdialenosti (b) a euklidovskej vzdialenosti (c). Je vidieť, že v takýchto prípadoch sa prejaví rozdiel vo vyhodnocovaní vzdialenosti. Na dostiahnutie rovnakého výsledku by bolo potrebné väčšie množstvo náčrtkov, alebo lepšie nakreslené náčrtky. Počas implementácie aj testovania sa mi viac osvedčila euklidovská vzdialenosť, je síce menej presná z povrchového hľadiska, no rozdiel je väčšinou zanedbateľný, keď ho kompenzujeme viacerými náčrtkami. Na druhú stranu, rozdiel v časovej zložitosti býva často veľký.



(a) Používateľský vstup

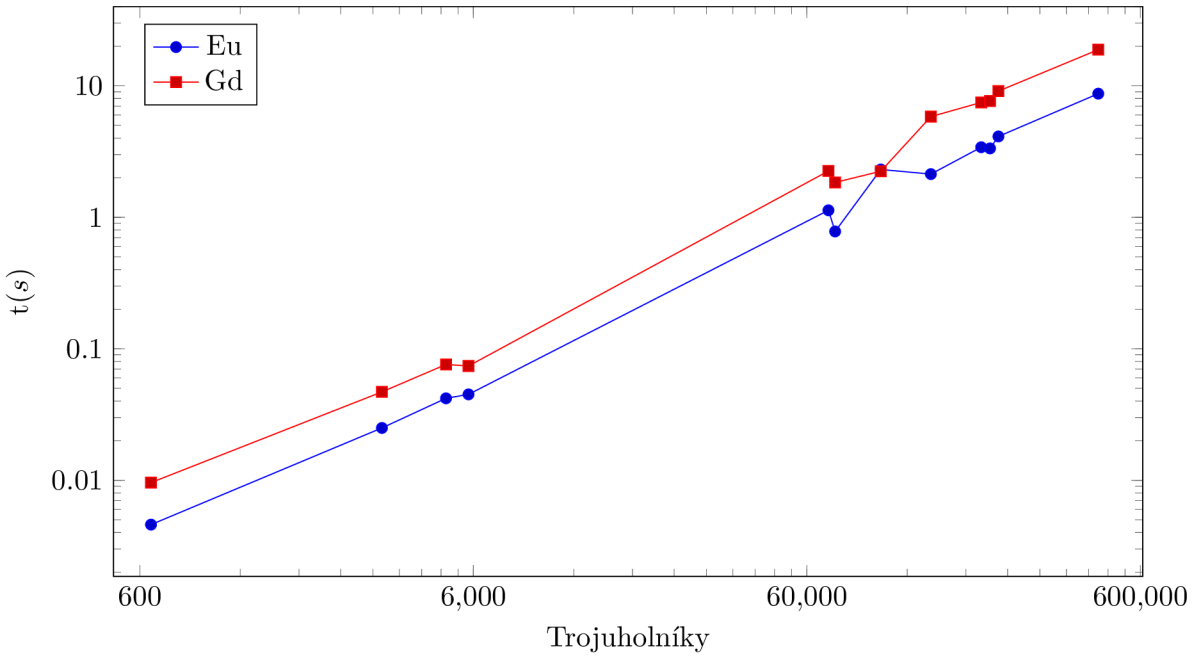
(b) Využitá geodetická vzdialenosť

(c) Využitá euklidovská vzdialenosť

Obr. 6.3: Ukážka rozdielneho výsledku aplikácie pre model ruky pri využití rôznych metód počítania vzdialenosti, keďže euklidovská vzdialenosť, počítaná vzdušnou čiarou je do červenej oblasti priradený prst naviac.

Finálne testovanie aplikácie prebiehalo na Amazon EC2 t2.micro serveri s operačným systémom Ubuntu Server 18.04. Testovanie na externom serveri, narozdiel od lokálneho serveru, viac odráža reálnu situáciu používania, kedy majú požiadavky väčšiu odozvu. Nasledujúca tabuľka zhŕňa časové výsledky testovania segmentácie a v prílohe B sú dostupné výsledné modely zoradené podľa zložitosti.

Aj keď som v Obrázku 6.4 zahrnul len celkový čas trvania chodu segmentácie robil som aj podrobnejšie testovanie časovej náročnosti jednotlivých úkonov aplikácie, čo mi dalo jasnejší obraz o tom, ktoré úkony sú najpomalšie, a ktoré časti by bolo potrebné optimalizovať. Prekvapivo nezanedbateľnú časť celkového trvania zaberá prevod modelu do internej podoby. Tento čas sa líši v závislosti od modelu, ale v porovnaní s celkovým trvaním segmentácie, používajúcej euklidovskú vzdialenosť ide približne o 30 % času. Najdlhší čas trvá hodnotiaca metóda, čo je hlavná časť aplikácie, a to hlavne pri zložitých modeloch. Vykonávanie graph-cutu trvá väčšinou len veľmi krátko. Doba prenosu modelov medzi klientom a serverom potom záleží hlavne na rýchlosti siete.



Obr. 6.4: Graf zobrazujúci časovú zložitosť v sekundách, v závislosti od zložitosti modelu v trojuholníkoch, Gd značí geodetickú a Eu euklidovskú vzdialenosť.

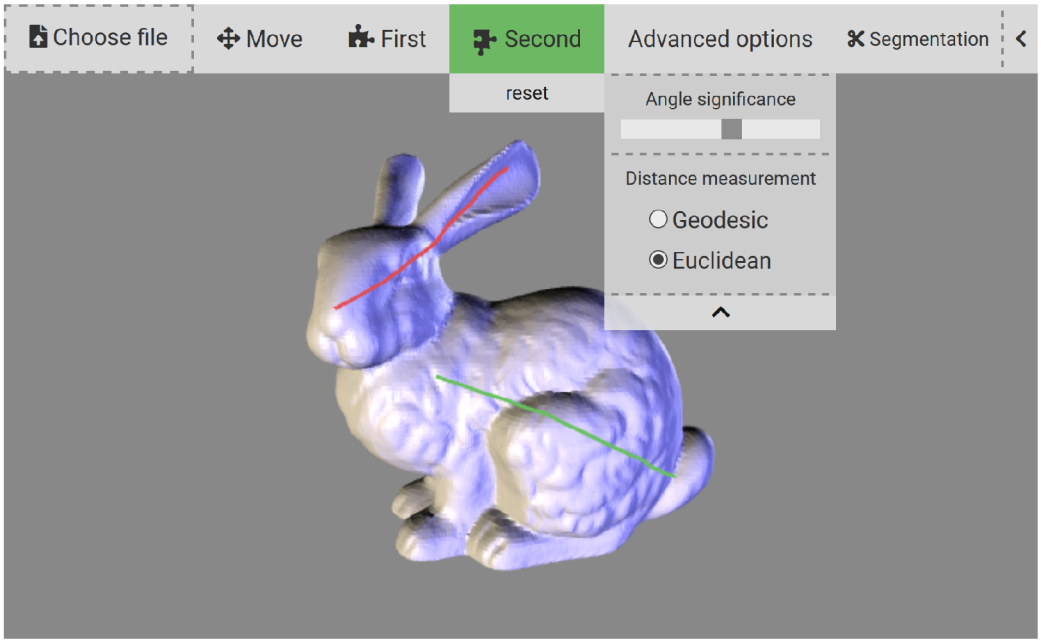
6.1 Zmeny používateľského rozhrania

Používaním aplikácie som odhalil nedostatky prvotného používateľského rozhrania, preto som ho postupne upravoval. Prvou zmenou bolo pridanie prepínača do stavu manipulovania modelom, pretože nebolo jasné kedy sa má model otáčať a kedy po ňom kresliť, čo spôsobovalo neočakávané a nechcené správanie aplikácie. V stave manipulovania je teraz možné pohybovať, otáčať a približovať kameru a po prepnutí do kreslenia je model nehybný.

Ďalšou zmenou bolo presunutie mazania náčrtov do samostatných položiek, nachádzajúcich sa pod prepínačmi náčrtov. Takto je možné mazať náčrty samostatne, bez vymazania oboch skupín.

Pridal som tiež pokročilé nastavenia, v ktorých je možné prepínať metriky vyhodnocovania vzdialenosti a slideru, ktorým sa určuje parameter λ spomínaný v 4.2, tieto vstupné parametre boli pôvodne staticky nastavené. Pridané rozšírenia vstupu majú za následok väčšiu voľnosť pri segmantovaní modelov, čo umožní skúsenejším užívateľom dosiahnuť výsledky bližšie ich zámeru.

Poslednou zmenou bolo pridanie do menu po segmentácii možnosť pokračovať s jedným z výsledných modelov a možnosť vrátiť sa späť k predošlým náčrtkom, ktoré by chcel používateľ poupravovať. Možnosť pokračovať v segmentovaní s jedným z modelov je praktická pri viacnásobnej segmentácií, nie je totiž nutné stále model sťahovať a znovu nahrávať do aplikácie.



Obr. 6.5: Finálna podoba grafického uživateľského rozhrania (pred segmentáciou).

Obr. 6.6: Finálna podoba grafického uživateľského rozhrania (po segmentácii).

6.2 Zhodnotenie výsledkov

Hodnotenie výsledkov segmentácie je všeobecne veľmi náročná úloha, pretože nie je veľmi jasné, aký je ideálny výsledok. Metódy segmentácie sa tiež líšia prístupom k segmentácii ako takej. Problematiku hodnotenia trochu objasňuje [3], ale článok ponúka metodiku hodnotenia len pre automatické metódy. Článok však prezentuje myšlienku hodnotenia segmentačných metód na základe ľudského vnímania a porovnávania s dátami, nameranými pri delení modelov ľuďmi. Vzhľadom na túto skutočnosť možno povedať, že užívateľ svojím vstupom z veľkej časti ovplyvňuje kvalitu výslednej segmentácie pri interaktívnych metódach. Preto bude zhodnotenie výsledkov zväčša subjektívne.

Prihliadajúc na spomínané fakty hodnotím výsledky segmentačnej aplikácie ako veľmi dobré. Čas segmentovania je pri väčšine modelov prijateľný, pri extrémne zložitých modeloch je samozrejme použitie geodetickej vzdialenosti nepraktické. Výsledné modely sú vo väčšine prípadov blízko zamýšľanej segmentácii, prípadne je možné používateľský vstup poupraviť a dosiahnuť presnejšiu segmentáciu.

Negatívne hodnotím obmedzenú použiteľnosť, a to hlavne kvôli rôznym spôsobom vytvárania modelov – pri ručne vytváraných modeloch často dochádza k nesprávne prepojeným častiam, čo sa veľmi negatívne prejavuje na výsledkoch. Ďalším negatívnym faktorom aplikácie je nutnosť istého porozumenia problematike segmentácia pre dosiahnutie ideálnych výsledkov – toto platí najmä pre λ parameter popísaný v časti 4.2.

Reálna využiteľnosť aplikácie je možná, no bolo by potrebné ďalšie testovanie na veľkom počte používateľov, a na rôznych serveroch, čo by priblížilo možnosti využiteľnosti v praxi.

6.3 Nadväzujúca práca

V budúcnosti by sa rozšírenia tejto práce mohli zamerať po prvé - na pomocné funkcie, alebo po druhé – na zlepšenia segmentačného algoritmu.

Prvou pomocnou funkciou by mohla byť decimácia modelu, čiže jeho zjednodušenie spojením trojuholníkov. Decimáciu by bolo možné implementovať s využitím Opemnesh knižnice, ktorú už aplikácia využíva a bola popísaná v 3.4. Bolo by tiež možné implementovať algoritmus z [6], ktorý je rýchly a vhodný pre extrémne veľké modely. Toto rozšírenie by malo význam pre úpravu modelu pred samotnou segmentáciou. Bolo by ho tiež možné implementovať podobne ako v 3.1, kde bolo použité niekoľkonásobné premodelovanie. Na tieto zjednodušené modely bola aplikovaná segmentácia a výsledky boli následne namapované na pôvodný model. Táto zmena by priniesla značné zrýchlenie segmentácie pri modeloch s veľkým počtom trojuholníkov, pri ktorých je náročné najmä počítanie geodetických vzdialeností.

Zlepšením segmentačnej metódy mohlo byť započítanie mean curvature do hodnotiacej funkcie hrán modelu. Problematiku krivostí dobre rozoberá článok [12]. Zohľadnenie krivosti by malo zlepšiť výsledky segmentácie, hlavne pri jemnejších modeloch s veľkým počtom trojuholníkov. Jedna z vecí, ktorá je však na zváženie je tiež čas segmentácie. Keďže ide o webovú aplikáciu, slúžiacu na používanie v reálnom čase, ďalšie výpočty by mohli mať za následok zhoršenie použiteľnosti, za cenu zvýšenia kvality segmentácie.

Ďalším rozšírením metódy by bolo zjemnenie hraníc postprocesingom, ako to bolo robené napríklad v pôvodnej metóde 3.3. Na jednej strane by takéto rozšírenie mohlo priniesť v mnohých prípadoch zlepšenie výslednej segmentácie, keďže by sa krajné trojuholníky rozdelily, no na druhej strane je to zásah do modelu samotného, preto by bolo vhodné toto rozšírenie implementovať pod prepínačom, ktorý by umožnil zapnutie/vypnutie vyhladenia hraníc.

Kapitola 7

Záver

V tejto práci bola priblížená problematika segmentácie polygonálnych modelov a boli v nej prezentované niektoré metódy segmentácie. Na základe jednej z prezentovaných metód[11] bol vytvorený nástroj na interaktívnu segmentáciu. Implementovaná metóda je upravená a zjednodušená, no pre účely aplikácie plne dostačujúca. Segmentačná metóda, založená na graph-cute berie do úvahy vzdialenosti trjuholníkov modelu od vstupných náčrtkov a dihedrálne uhly medzi trojuholníkmi. Nástroj bol implementovaný vo forme webovej aplikácie, čo umožňuje vykonávať segmentáciu každému s prístupom na internet.

Aplikácia je plne funkčná a dosahuje dobré výsledky v pomerne krátkom čase. Najlepšie výsledky sú dosiahnuté pri malých až stredne veľkých modeloch zvierat, ľudí, či iných modeloch s vyčnievajúcimi časťami. Pre niektoré modely, hlavne pre modely pozostávajúce z viacerých oddelených častí aplikácia nedosahuje vždy dobré výsledky. Časová náročnosť je väčšinou prijateľná, bližšie popísaná je v grafe 6.4.

S prihliadnutím na výsledky testovania, by sa budúci vývoj mohol sústrediť na vylepšenie metódy formou zohľadňovania krivostí hrán, či interné zjednodušovanie modelov s veľkým počtom trojuholníkov, čo by malo prispieť k dosiahnutiu lepších výsledkov.

Literatúra

[1] Botsch, M.; Steinberg, S.; Bischoff, S.; aj.: OpenMesh - a generic and efficient polygon mesh data structure. 02 2002.

[2] Boykov, Y.; Kolmogorov, V.: An experimental comparison of min-cut/max- flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, ročník 26, č. 9, Sep. 2004: s. 1124–1137, ISSN 0162-8828, doi:10.1109/TPAMI.2004.60.

[3] Chen, X.; Golovinskiy, A.; Funkhouser, T.: A Benchmark for 3D Mesh Segmentation. *ACM Trans. Graph.*, ročník 28, č. 3, Júl 2009: s. 73:1–73:12, ISSN 0730-0301, doi:10.1145/1531326.1531379.
URL http://doi.acm.org/10.1145/1531326.1531379

[4] Crane, K.; Weischedel, C.; Wardetzky, M.: The Heat Method for Distance Computation. *Commun. ACM*, ročník 60, č. 11, Október 2017: s. 90–99, ISSN 0001-0782, doi:10.1145/3131280.
URL http://doi.acm.org/10.1145/3131280

[5] DEMEL, J.; JAN, J.: Segmentace 3D obrazovỳch dat s využitím grafové reprezentace. 2014.

[6] Isenburg, M.; Lindstrom, P.; Gumhold, S.; aj.: Large mesh simplification using processing sequences. In *IEEE Visualization, 2003. VIS 2003.*, Oct 2003, s. 465–472, doi:10.1109/VISUAL.2003.1250408.

[7] Katz, S.; Tal, A.: Hierarchical Mesh Decomposition Using Fuzzy Clustering and Cuts. *ACM Trans. Graph.*, ročník 22, č. 3, Júl 2003: s. 954–961, ISSN 0730-0301, doi:10.1145/882262.882369.
URL http://doi.acm.org/10.1145/882262.882369

[8] Lai, Y.; Zhou, Q.; Hu, S.; aj.: Robust Feature Classification and Editing. *IEEE Transactions on Visualization and Computer Graphics*, ročník 13, č. 1, Jan 2007: s. 34–45, ISSN 1077-2626, doi:10.1109/TVCG.2007.19.

[9] Lai, Y.-K.; Zhou, Q.-Y.; Hu, S.-M.; aj.: Feature Sensitive Mesh Segmentation. In *Proceedings of the 2006 ACM Symposium on Solid and Physical Modeling*, SPM '06, New York, NY, USA: ACM, 2006, ISBN 1-59593-358-1, s. 17–25, doi:10.1145/1128888.1128891.
URL http://doi.acm.org/10.1145/1128888.1128891

[10] Liu, L.; Sheng, Y.; Zhang, G.; aj.: Graph Cut Based Mesh Segmentation Using Feature Points and Geodesic Distance. In *2015 International Conference on Cyberworlds (CW)*, Oct 2015, s. 115–120, doi:10.1109/CW.2015.31.

[11] Meng, M.; Ji, Z.; Liu, L.: Sketching Mesh Segmentation Based on Feature Preserving Harmonic Field. *Journal of Computer-Aided Design & Computer Graphics (in Chinese)*, ročník 20, č. 9, 2008: s. 1146–1152.

[12] Meyer, M.; Desbrun, M.; Schröder, P.; aj.: Discrete Differential-Geometry Operators for Triangulated 2-Manifolds. In *Visualization and Mathematics III*, editácia H.-C. Hege; K. Polthier, Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, ISBN 978-3-662-05105-4, s. 35–57.

[13] Price, B. L.; Morse, B.; Cohen, S.: Geodesic graph cut for interactive image segmentation. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 2010, ISSN 1063-6919, s. 3161–3168, doi:10.1109/CVPR.2010.5540079.

[14] Shamir, A.: A survey on Mesh Segmentation Techniques. *Computer Graphics Forum*, ročník 27, č. 6, 2008: s. 1539–1556, doi:10.1111/j.1467-8659.2007.01103.x.
URL
https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-8659.2007.01103.x

[15] Shapira, L.; Shamir, A.; Cohen-Or, D.: Consistent mesh partitioning and skeletonisation using the shape diameter function. *The Visual Computer*, ročník 24, č. 4, Jan 2008: str. 249, ISSN 1432-2315, doi:10.1007/s00371-007-0197-5.
URL https://doi.org/10.1007/s00371-007-0197-5

[16] Tierny, J.; Vandeborre, J.; Daoudi, M.: Topology driven 3D mesh hierarchical segmentation. In *IEEE International Conference on Shape Modeling and Applications 2007 (SMI '07)*, June 2007, s. 215–220, doi:10.1109/SMI.2007.38.

[17] Tierny, J.; Vandeborre, J.-P.; Daoudi, M.: 3D Mesh Skeleton Extraction Using Topological and Geometrical Analyses. 10 2006: s. 85–94.

[18] Vieira, M.; Shimada, K.: Surface mesh segmentation and smooth surface extraction through region growing. *Computer Aided Geometric Design*, ročník 22, č. 8, 2005: s. 771 – 792, ISSN 0167-8396, doi:https://doi.org/10.1016/j.cagd.2005.03.006.
URL http://www.sciencedirect.com/science/article/pii/S0167839605000282

Príloha A

Obsah CD

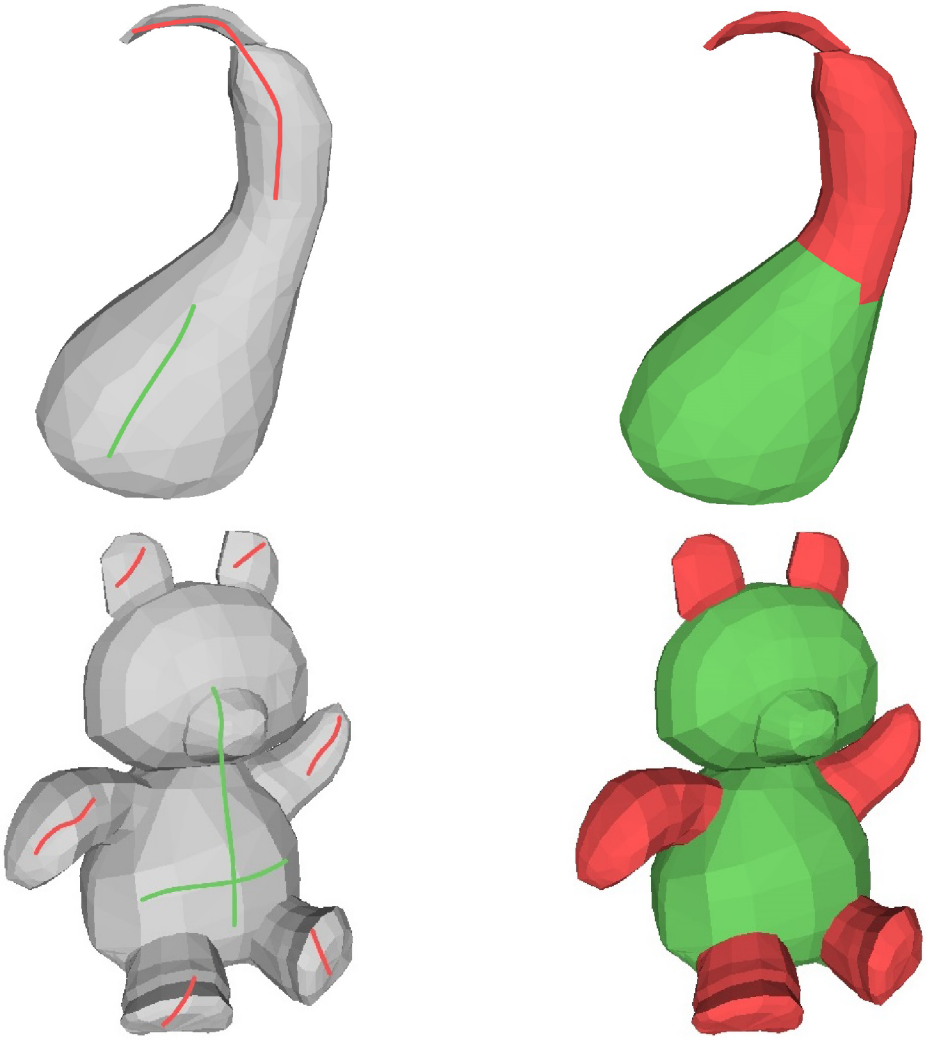
Štruktúra priloženého CD je nasledovná:

- **bin** - zložka obsahujúca knižnice
- **doc** - zložka obsahujúca programovú dokumentáciu a túto správu
- **models** - zložka obsahujúca testovaciu sadu
- **server** - zložka obsahujúca webovú aplikáciu
- **src** - zložka obsahujúca zdrojové súbory segmentačného programu
- README - manuál na obsluhu aplikácie
- Plagát

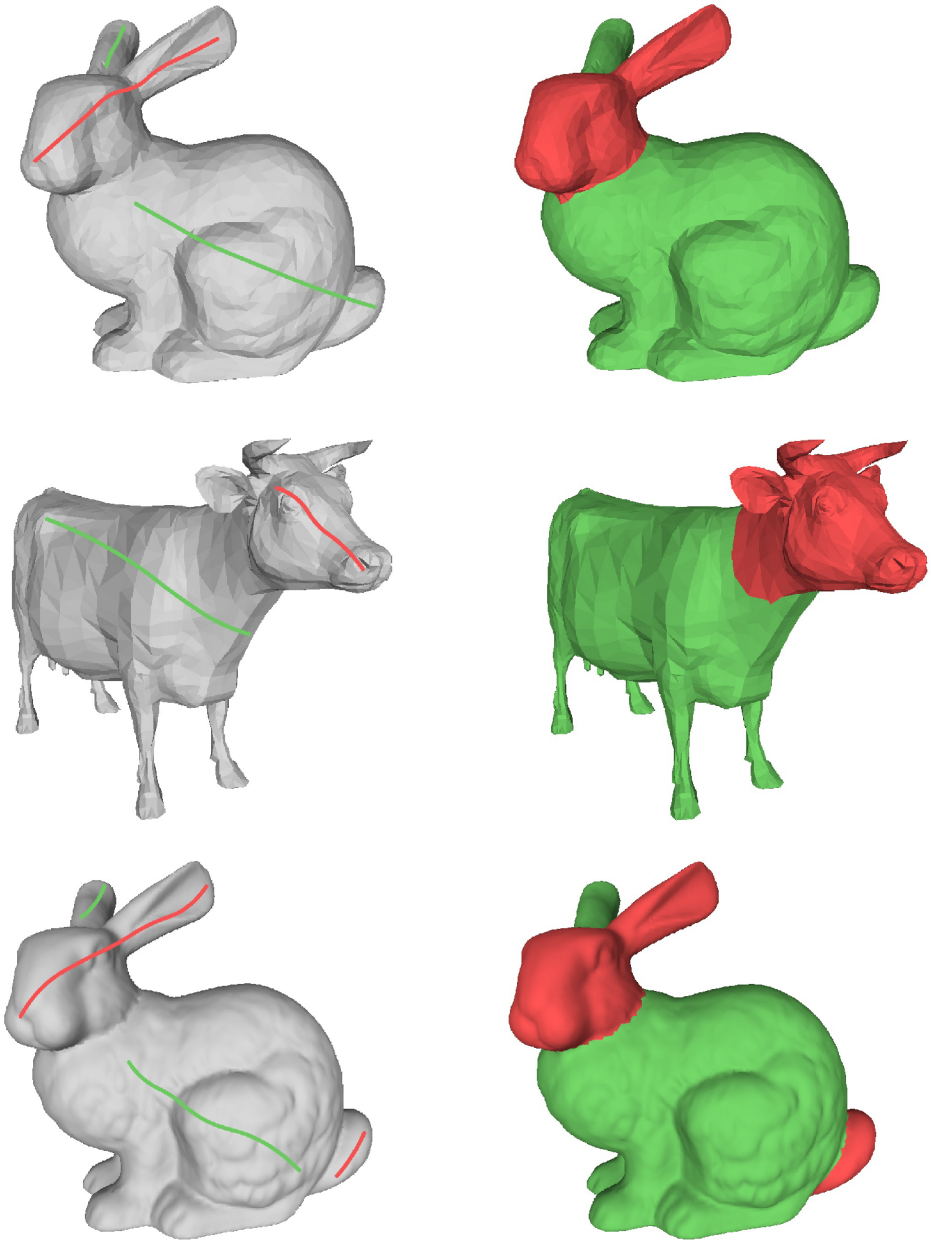
Príloha B

Výsledky testovania

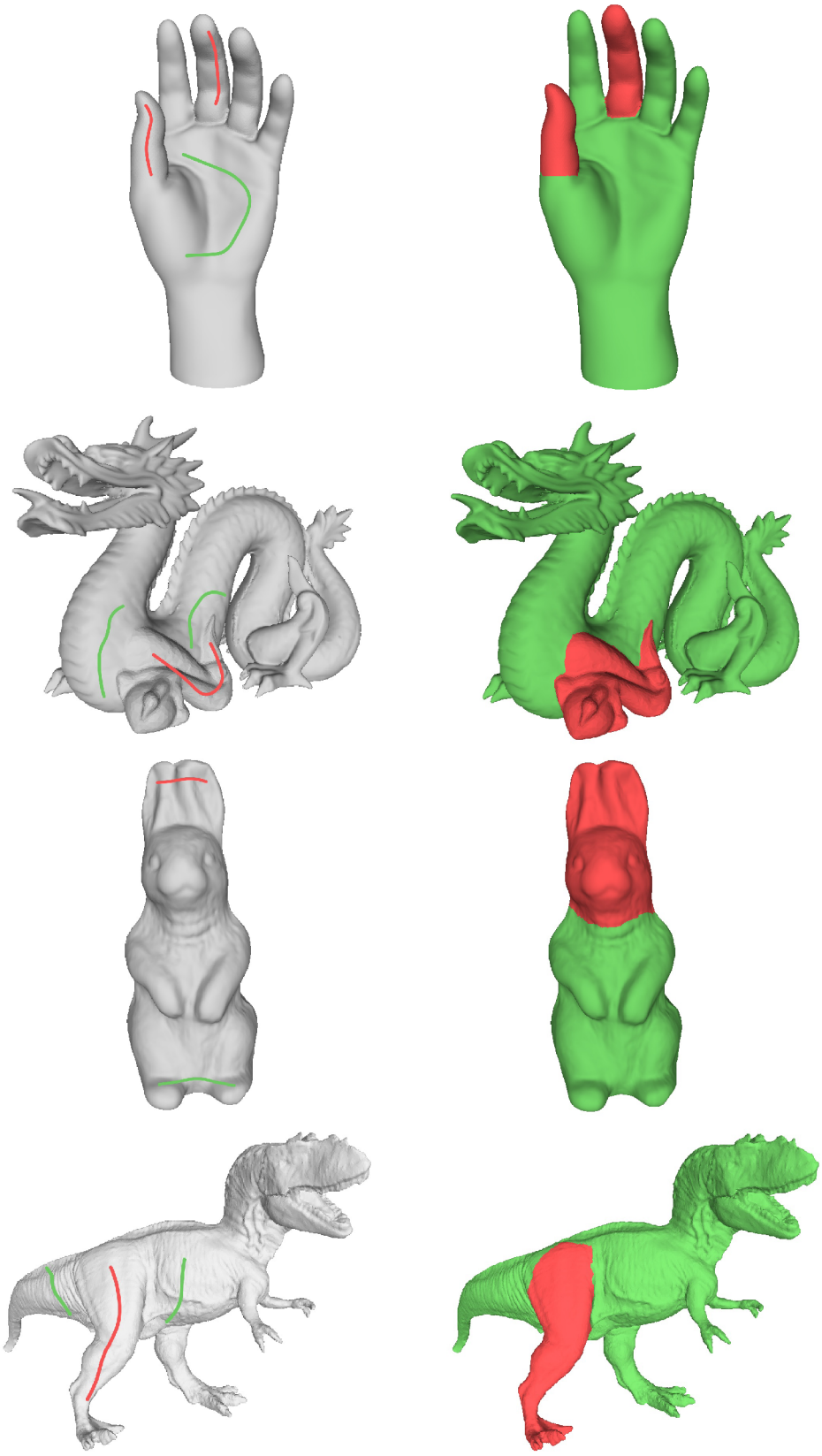
Nasledujúce obrázky zobrazujú uživateľský vstup naľavo a výsledok segmentácie napravo. Výsledky sú zoradené podľa veľkosti modelu vzostupne.



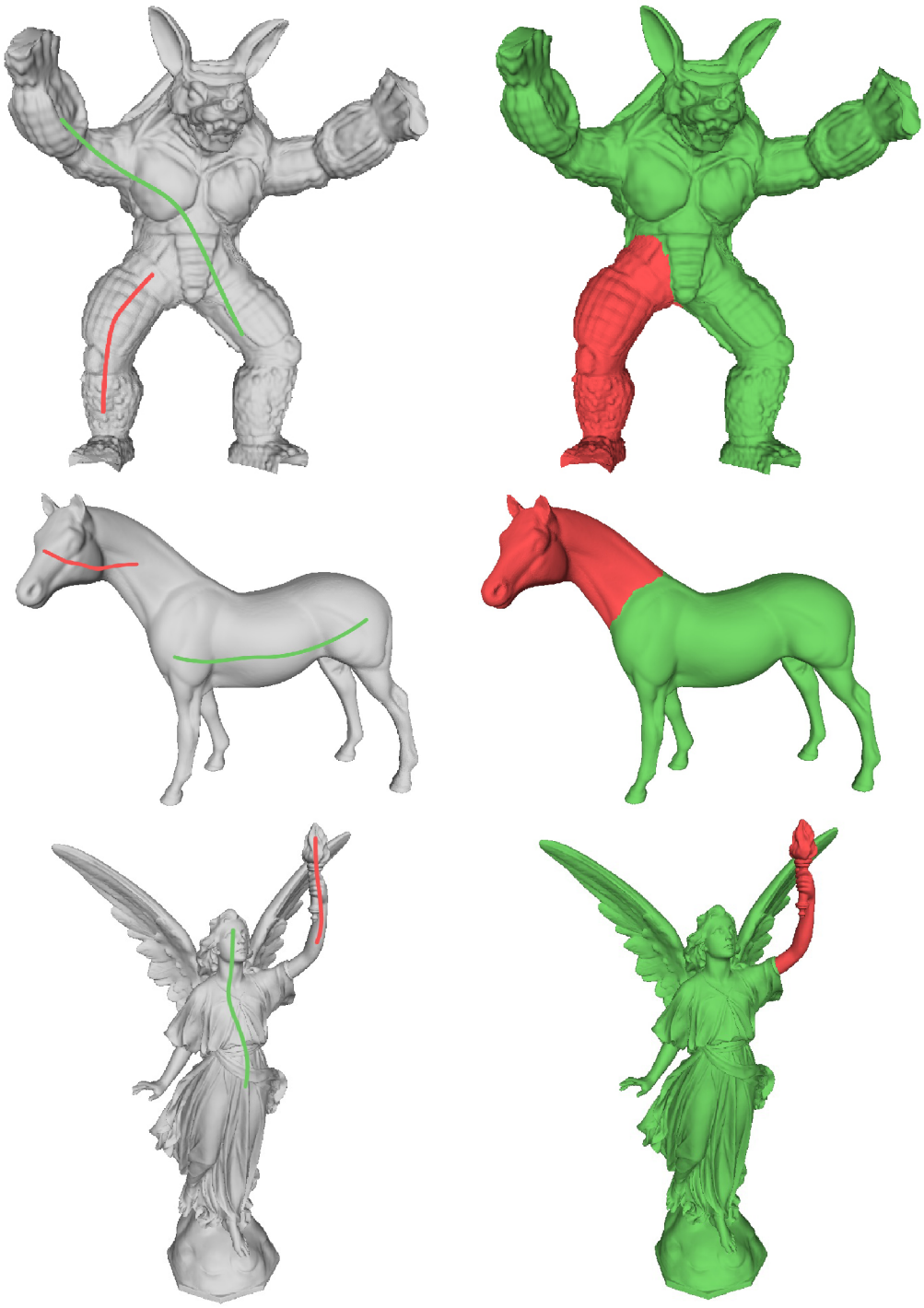
Obr. B.1: Výsledky testovania pre modely gourd a teddy.



Obr. B.2: Výsledky testovania pre modely bunny, cow a big-bunny.



Obr. B.3: Výsledky testovania pre modely hand, dragon, rabbit a t-rex.



Obr. B.4: Výsledky testovania pre modely armadillo, horse a angel.

Príloha C

Plagát