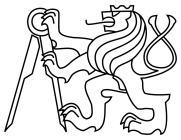
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ



ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název: Grafické uživatelské rozhraní pro aplikaci Admesh
Student: David Vyvlečka
Vedoucí: Ing. Marek Žehra
Studijní program: Informatika
Studijní obor: Web a multimédia
Katedra: Katedra softwarového inženýrství
Platnost zadání: do konce letního semestru 2015/16

Pokyny pro vypracování

STL je formát pro reprezentaci 3D modelů určených nejen pro 3D tisk. Navrhněte a implementujte nadstavbu v podobě grafického uživatelského rozhraní k aplikaci Admesh, která umožňuje automatickou úpravu topologie a manipulaci s uvedeným formátem. Využijte skutečnosti, že aplikace poskytuje rozhraní pro programovací jazyky C a Python. Cílem práce je vytvořit intuitivní uživatelské rozhraní s možností zobrazovat načtený soubor se zvýrazněním změn provedených aplikací Admesh. Uživatel bude mít možnost vybrat akce, které se mají provést, například selektivně opravit normálové vektory nebo škálovat 3D model.

Seznam odborné literatury

Dodá vedoucí práce.

L.S.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

prof.Ing. Pavel Tvrdík, CSc.
děkan

V Praze dne 1. prosince 2014

České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra softwarového inženýrství

Bakalářská práce

Grafické uživatelské rozhraní pro aplikaci ADMesh

David Vyvlečka

Vedoucí práce: Ing. Marek Žehra

6. května 2015

Poděkování

Rád bych poděkoval Bc. Miroslavu Hrončokovi za četné hodnotné konzultace a rady při vypracování bakalářské práce. Děkuji také spolužákům a členům laboratoře 3D tisku Fakulty informačních technologií ČVUT v Praze, kteří se podíleli na testování výsledné aplikace.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 6. května 2015 .....

Odkaz na tuto práci

Vyvlečka, David. *Grafické uživatelské rozhraní pro aplikaci ADMesh.* Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2015.

Abstrakt

Tato bakalářská práce se věnuje návrhu a implementaci rozšíření nástroje ADMesh v podobě grafického uživatelského rozhraní. Nástroj ADMesh slouží k manipulacím s 3D modely ve formátu STL a k jejich opravám. Grafické uživatelské rozhraní využívá poskytnuté rozhraní v jazyce C a je implementováno v jazyce C++ za podpory frameworku Qt a knihovny OpenGL. Rozhraní umožňuje uživateli zobrazit model v 3D prohlížeči, provádět vybrané akce a získat zpětnou vazbu.

Klíčová slova STL, C++, OpenQL, Qt, 3D grafika, 3D tisk, GUI

Abstract

This bachelor thesis is devoted to designing and implementation of an extension for ADMesh tool in the form of graphical user interface. ADMesh tool allows to manipulate and repair 3D models in the STL format. The graphical user interface uses ADMesh C API and is implemented in C++ with support of the Qt framework and the OpenGL library. The interface allows the user to view the model in 3D viewer, to perform selected actions and to get visual feedback of those.

Keywords STL, C++, OpenQL, Qt, 3D graphics, 3D printing, GUI

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Tato práce se věnuje implementaci jednoduchého open-source grafického nástroje pro prohlížení 3D modelů ve formátu STL a manipulaci s nimi. Dosud obdobný nástroj neexistoval. Nástroj je vybudován jako rozšíření stávající aplikace ADMesh, která zajišťuje potřebnou funkcionalitu, ale nabízí pouze rozhraní pro příkazovou řádku a knihovní rozhraní pro jazyk C. Nově vznikající aplikace byla nazvána ADMeshGUI.

Před započetím implementace bylo nezbytné seznámit se s metodami reprezentace 3D modelů v počítačové grafice a s chybami, které mohou modely obsahovat. Této problematice se věnuje kapitola *3D modely v počítačové grafice*. V kapitole *Analýza a návrh* následuje rozbor souborového formátu STL a představení vybraných aplikací, které umožňují s danými modely manipulovat. V sekci 3.3 je podrobně rozebrán nástroj ADMesh. Kapitola je završena představením návrhu rozhraní a celkového rozložení ovládacích prvků aplikace ADMeshGUI.

Kapitola 4 je celá věnována implementaci. Představeny jsou nejzajímavější problémy, které bylo nutné při implementaci vyřešit. Práce je zakončena kapitolou *Testování* věnovanou testům, které byly nad aplikací prováděny pro ověření správné funkčnosti.

Kapitola 1

Cíl práce

Cílem práce je implementace platformně nezávislé open-source aplikace nazvané ADMeshGUI začleňující funkcionality nástroje ADMesh do grafického uživatelského rozhraní a propojení s jednoduchým prohlížečem STL souborů.

1.1 Funkční a nefunkční požadavky

Pro upřesnění cílů, kterých je potřeba při implementaci aplikace ADMeshGUI dosáhnout, byl sestaven následující seznam funkčních a nefunkčních požadavků.

1.1.1 Funkční požadavky

- Aplikace umožní otevření ASCII i binárního STL souboru.
- Aplikace zobrazí otevřený 3D model v okně prohlížeče.
- Aplikace umožní prohlížet model (pohybovat se ve scéně, rotovat, přibližovat) nezávisle na operacích ADMeshe.
- Aplikace umožní několik módů zobrazení modelu (drátový, plný, plný se zvýrazněnými hranami).
- Aplikace zvýrazní zobrazitelné chyby modelu (špatná orientace facetu).
- Aplikace umožní provádět operace definované v knihovně ADMesh (viz sekce 3.3.1 na straně 15).
- Aplikace automaticky zobrazí změny prováděné na modelu.
- Aplikace umožní uložení a export modelu do formátů, které podporuje ADMesh.

1.1.2 Nefunkční požadavky

- Aplikace bude multiplatformní, zahrnuty jsou platformy Linux, Windows a Mac OS X.
- Aplikace poskytne základ pro lokalizaci do dalších jazyků.

Kapitola 2

3D modely v počítačové grafice

Tato kapitola se zabývá možnostmi reprezentace prostorových modelů v počítačové grafice a chybami, které se vyskytují u modelů v hraniční reprezentaci. Uvedené chyby mohou mít vliv na následnou produkci fyzických modelů, např. pomocí 3D tisku. Seznámení se s chybami je stěžejní pro pochopení primárního účelu, za kterým byl vytvořen nástroj ADMesh.

2.1 Reprezentace 3D modelů

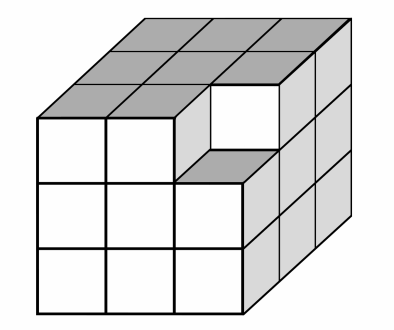
2.1.1 Objemové reprezentace

Výčtová reprezentace

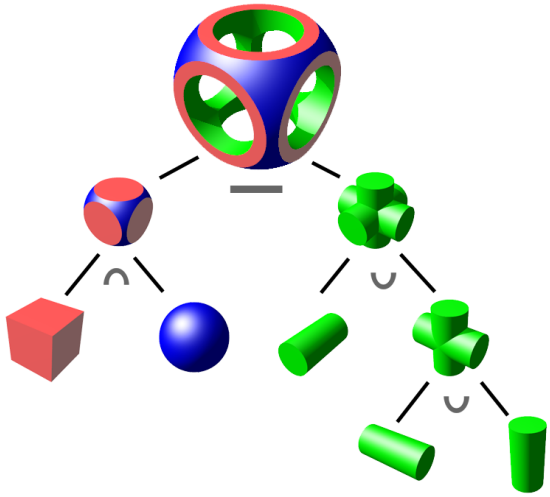
„Používá ‚pokrytí' tělesa shodnými elementárními tělesy, nejčastěji krychlemi, nebo výhodněji nahradí těleso systémem krychlí a celou strukturu popíše pomocí tzv. oktalového stromu. Tj. z každého uzlu grafu vychází osm hran. Podstata tvorby oktalového stromu spočívá v očíslování osmi krychlí, na které rozdělíme původní krychli.“[1] Princip výčtové reprezentace je názorně demonstrován na obrázku 2.1,

CSG (Constructive solid geometry)

CSG je přístupem k reprezentaci 3D modelů pomocí několika základních snadno parametrizovatelných geometrických objektů (krychle, koule, válec, ...) a nad nimi prováděných množinových operací (sjednocení, průnik, rozdíl, ...). Výsledný model může být reprezentován CSG stromem (obr. 2.2), který v listech obsahuje elementární geometrická tělesa, ve vnitřních uzlech množinové operace a na hranách geometrické transformace. Primární výhodou tohoto přístupu je parametrizovatelnost dílčích těles umožňující přesnou kontrolou nad rozměry objektu. CSG se tedy hodí pro tvorbu modelů technických součástek, např. pro tisk na 3D tiskárně. Nevýhody spočívají v nutnosti definovat chování množinových operací v hraničních situacích (vznik non-manifold



Obrázek 2.1: Princip výčtové reprezentace – výsledné těleso je tvořeno elementárními tělesy

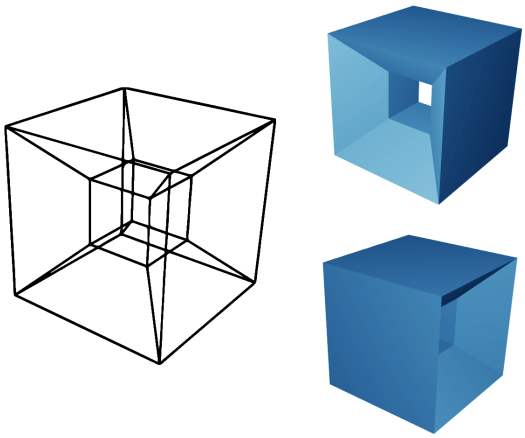


Obrázek 2.2: Znázornění CSG stromu, který obsahuje elementární tělesa a množinové operace. Znázorněné množinové operace obsahují ∩ pro průnik, ∪ pro sjednocení a − pro rozdíl [3]

meshe) a velice obtížném vytváření modelů složitějších organických tvarů. Pro zobrazení modelu vytvářeného pomocí CSG je navíc nutné model renderovat. Renderování modelu klade vyšší nároky na čas i prostředky.[2]

2.1.2 Drátový model

Drátový model používá k reprezentaci těles pouze vrcholy a hrany. Přestože je velmi úsporný, není vhodný pro modelovací operace. Modely reprezentované tímto způsobem mohou být nejednoznačné (není možné určit přesný tvar objektu, viz obr 2.3), případně je nemožné určit viditelnost jednotlivých částí.[4]



Obrázek 2.3: Nejednoznačnost drátového modelu: drátový model může představovat obě tělesa napravo, není možné jednoznačně rozhodnout které. Model byl vytvořen v aplikaci Blender

2.1.3 Hraniční reprezentace

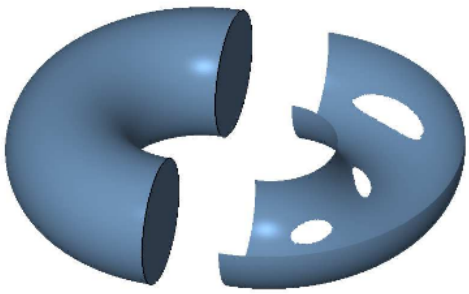
Metoda reprezentace 3D modelu rozšiřující reprezentaci pomocí drátového modelu o popis facetů (plochy tvořící stěny objektu ohraničené hranami). Model je popsán:

- vertexy = vrcholy,
- hranami,
- facety

a topologickými vztahy mezi nimi. Vertexy tvořící jeden facet by měly navíc splňovat podmínku shodné orientace, které se využívá při výpočtu normálového vektoru daného facetu. Shodná orientace normálových vektorů všech facetů (ven z tělesa) je důležitá např. pro zajištění správného osvětlení modelu. Výhody hraniční reprezentace spočívají ve snadné zobrazitelnosti a editovatelnosti.[5]

2.2 Chyby modelů v hraniční reprezentaci

Modely reprezentované pomocí hraniční reprezentace mohou obsahovat řadu chyb. Tyto chyby mají zásadní vliv zejména na produkci fyzických modelů (např. pomocí 3D tisku). Následuje popis typických chyb, pro zjednodušení je počítáno pouze s trojúhelníkovými facety.



Obrázek 2.4: Ukázka objektu s dírami. Vlevo uzavřený objekt, vpravo objekt s dírami [7]

2.2.1 Nemanifoldní modely

Zjednodušeně řečeno je nemanifoldním 3D modelem každý model, který není možné reálně vytvořit. Přesněji tento model nesplňuje podmínku 2-manifoldnosti. „n-manifoldnost je vlastnost bodu, která určuje počet regionů n, na které je prostor rozdělen pomocí tohoto bodu. Všechny body 3D modelu v hraniční reprezentaci by měly být právě 2-manifoldní – rozdělují prostor na dva regiony, ‚venku' a ‚uvnitř' .“ Stejná logika platí i u hran.[6]

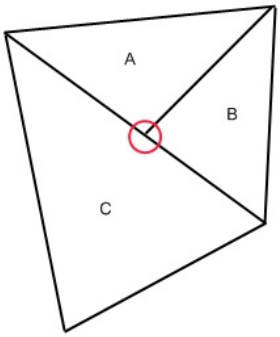
Obecně lze tedy nemanifoldnost 3D modelu spojit se vznikem jeho částí, které mají některý rozměr nulový. Například model obsahující pouze jeden trojúhelník (stěna reprezentovaná tímto trojúhelníkem má nulovou tloušťku) není 2-manifoldní, neboť žádný bod nerozděluje prostor na 2 regiony (není znám region „uvnitř“). Takový model je sice možné zobrazit v počítačovém prostoru, reálně však neexistuje.

2.2.2 Díry

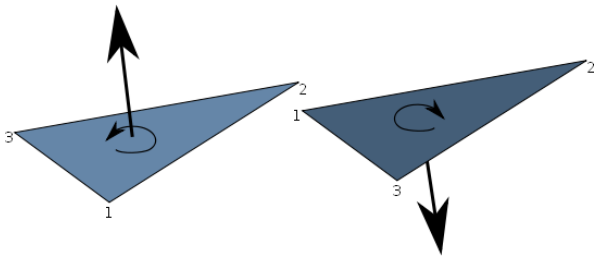
Aby mohl být model využit pro 3D tisk, musí být jeho povrch uzavřený. Každá hrana musí být připojena právě ke dvěma trojúhelníkům a všechny sousední trojúhelníky musí sdílet vnitřní hranu. 3D model, který obsahuje díry (místa, která neobsahují trojúhelníky a jsou ohraničena hranami připojenými k méně než dvěma trojúhelníkům), tyto podmínky nesplňuje a není dle definice manifoldnosti 2-manifoldní.[7] Obrázek 2.4 demonstruje rozdíl mezi uzavřeným modelem a modelem obsahujícím díry.

2.2.3 T-Vertex

„T-Vertex je ve zkratce bod, který leží na hraně nějakého trojúhelníku a zároveň podle vstupního souboru není součástí tohoto trojúhelníku.“[6] Situace je znázorněna na obrázku 2.5. T-Vertex může při manipulaci s modelem vyústit v další problémy, např. může dojít ke vzniku díry.[6]



Obrázek 2.5: Ukázka T-Vertexu [6]



Obrázek 2.6: Ukázka rozdílné orientace facetu. Vlevo jsou vrcholy facetu orientovány proti směru hodinových ručiček, vpravo po směru hodinových ručiček [7]

2.2.4 Degenerované trojúhelníky

Degenerovaným trojúhelníkem je každý trojúhelník, jehož všechny tři vrcholy leží na jedné přímce. Specifičtějším případem je trojúhelník, který má dva nebo všechny tři vrcholy totožné.[8] Vznik degenerovaných trojúhelníků je umožněn nepřesnými výpočty v aritmetice s plovoucí řádovou čárkou.[6] Obdobně může ke vzniku degenerovaných trojúhelníků dojít při konverzi souboru z formátu s dvojitou přesností do formátu s přesností jednoduchou.[9]

2.2.5 Nekoherentní orientace trojúhelníků

Orientace trojúhelníků určuje „vnější“ a „vnitřní“ prostor modelu. Toto rozdělení vychází z orientace jednotlivých vrcholů v trojúhelníku, která může být po nebo proti směru hodinových ručiček (obr. 2.6). Je zřemé, že orientace vrcholů se mění v závislosti na pozici, ze které je trojúhelník nahlížen. Např. formát STL správnou orientaci definuje tzv. pravidlem pravé ruky (viz následující kapitola, sekce 3.1). Jednotná orientace je stěžejní pro produkci reálných fyzických modelů.[7]

Na základě orientace trojúhelníku je možné spočítat jeho normálový vektor. Pro produkci reálného fyzického modelu je tedy nutné dodržet zejména správnou orientaci trojúhelníků, normálové vektory trojúhelníku mohou být před dalším zpracováním přepočítány.[10]

Kapitola 3

Analýza a návrh

Po seznámení se s reprezentací modelů v prostoru počítačové grafiky a s jejich chybami bylo možné přistoupit ke konkrétní analýze. Tato kapitola začíná rozborem souborového formátu STL a přehledem existujících aplikací, které slouží k manipulaci se soubory v tomto formátu. Následuje rozbor výchozí aplikace ADMesh, pro kterou je navrhováno rozšíření, a rozbor předpokládaných uživatelů výsledné aplikace ADMeshGUI. Kapitola je završena představením návrhu uživatelského rozhraní.

3.1 Souborový formát STL

Formát STL[11] byl představen společností 3D Systems na konci 80. let 20. století. Vznik nového formátu byl podnícen potřebou jednoduchého vstupního formátu pro zařízení vyrábějící fyzické modely na základě počítačových dat. Jednoduchost formátu přispěla k jeho rychlému rozšíření mezi většinu CAD systémů.

Model je v tomto formátu reprezentován jednotlivými trojúhelníkovými facety. Každý facet nese informaci o svém normálovém vektoru a o třech vrcholech, kterými je tvořen. V původní specifikaci musel STL formát dále splňovat následující pravidla[10]:

1. Normála a každý vrchol facetu jsou určeny třemi koordinátami, dohromady tedy 12 čísel na každý facet.

2. Každý facet tvoří hranici mezi vnitřním a vnějším prostorem tělesa. Orientace facetů je určena dvěma způsoby, které se musí shodovat. Zaprvé, normálový vektor směřuje ven z tělesa. Zadruhé, vrcholy jsou seřazeny proti směru hodinových ručiček při pohledu zvnějšku tělesa (pravidlo pravé ruky[1]).

[1] palec ve směru normály a pořadí vektorů podle směru ohnutých prstů

3. Každý facet musí sdílet dva vrcholy s každým ze sousedních facetů (pravidlo vertex-to-vertex).

4. Všechny koordináty vrcholů musí být kladné.

3.1.1 ASCII STL

ASCII je verze formátu STL čitelná pro člověka. Každý soubor v tomto formátu dodržuje následující syntax:

solid *name*

$$\left\{\begin{array}{ll} \textbf{facet normal} & n_i\ n_j\ n_k \\ \quad \textbf{outer loop} & \\ \quad\quad \textbf{vertex} & v1_x\ v1_y\ v1_z \\ \quad\quad \textbf{vertex} & v2_x\ v2_y\ v2_z \\ \quad\quad \textbf{vertex} & v3_x\ v3_y\ v3_z \\ \quad \textbf{endloop} & \\ \textbf{endfacet} & \end{array}\right\}$$

endsolid *name*

V daném schématu může být *name* vynecháno nebo nahrazeno libovolným popisem. Tučně jsou zvýrazněna klíčová slova, která musí být v souboru zapsána malými písmeny. K odsazení je nutné použít mezery, tabelátor není dovolen.[12] Mezi úvodním a závěrečným řádkem jsou definovány jednotlivé facety (obsah závorek {} se opakuje pro každý facet), kde n_i – n_k představují koordináty normály facetu a $v1_x$ – $v3_z$ koordináty vrcholů.

3.1.2 Binární STL

Binární verze formátu nabízí efektivnější uložení dat modelu. Všechny záznamy v binárním STL souboru jsou uloženy v pořadí little endian a všechna data v plovoucí řádové čárce mají jednoduchou přesnost v souladu s ANSI/IEEE standard 754.[11] Binární STL soubor začíná hlavičkou dlouhou 84 bytů, přičemž poslední 4 byty nesou informaci o počtu facetů uložených v souboru. Facety jsou uloženy ihned za hlavičkou, pro každý je potřeba 50 bytů. Normála i tři vrcholy mají po třech koordinátách, každá koordináta je desetinné číslo o velikosti čtyři byty. Dohromady tedy $4 \times 3 \times 4 = 48$ bytů, zbývající dva byty nejsou využity.[12] Strukturu binárního STL souboru názorně přibližuje následující schéma. Struktura uvnitř závorek {} je v souboru zopakována

pro každý facet.

80	ASCII	hlavička
4	unsigned long integer	počet facetů v souboru
4	float	i normály
4	float	j
4	float	k
4	float	x vertexu 1
4	float	y
4	float	z
4	float	x vertexu 2
4	float	y
4	float	z
4	float	x vertexu 3
4	float	y
4	float	z
2	unsigned integer	nevyužito

Striktní pravidla původní specifikace STL formátu znemožňující definovat non-manifold mesh (2. bod specifikace, viz 3.1) či umístit model v libovolném oktantu (4. bod specifikace, viz 3.1) vedla ke zobecnění STL formátu pro aplikace, které nevyžadují přesné dodržení původní specifikace.

Absence popisu dalších vlastností modelu ve formátu STL, jako např. barvy či materiálu, vedla s dalším rozvojem rapid prototypingu k pokusům rozšířit formát o podporu těchto informací. Na rozdíl od původní specifikace formátu či zobecněného STL se však žádná takto rozšířená verze nestala průmyslovým standardem.

Společnost 3D Additive Fabrication[13] poukazuje na tento nedostatek společně s kritikou neefektivního ukládání detailních modelů. Nutnost ukládat každý facet modelu zvyšuje celkovou velikost souboru, což může negativně ovlivnit čas potřebný k jeho zpracování. Tento nedostatek je možné částečně eliminovat použitím binárního STL formátu za cenu ztráty čitelnosti pro člověka.

Přes tyto nedostatky zůstává formát STL díky své rozšířenosti hlavním formátem v oblasti 3D tisku.[13]

3.2 Rozbor obdobných řešení

Blender

Blender [14] je zástupcem rozsáhlých 3D editorů primárně zaměřených na modelování a rendering (umožňuje ale i tvorbu animací a her). Rozsáhlost Blenderu se odráží i v jeho podpoře externích formátů, mezi nimiž nechybí ani formát STL. S načteným souborem je možné libovolně manipulovat, editovat topologii a ručně provádět opravy. Pro uživatele, který potřebuje rychle a jednoduše opravit model (např. kvůli přípravě pro 3D tisk), však není Blender vzhledem ke své rozsáhlosti a zaměření na modelování ideálním řešením. Především je opravy nezbytné provádět postupně, uživatel již navíc musí mít povědomí o rozložení ovládacích prvků, případně znalost klávesových zkratek.

MeshLab

MeshLab je aplikace pro vytváření a editaci 3D modelů. MeshLab vznikl na Fakultě informatiky Univerzity v Pise, vytvořili jej z převážné části sami studenti. Umožňuje importovat a exportovat různé formáty pro reprezentaci 3D dat. Kromě hlavní části aplikace umožňuje především jednoduše prohlížet 3D objekty ve formátu STL a zjednodušovat je.[15]

Jelikož je MeshLab patrně nejznámějším open-source editorem schopným opravit soubory ve formátu STL, jedná se o nejbližší konkurenci vytvářené aplikace ADMeshGUI. Vyzkoušením aplikace MeshLab byly zjištěny její následující nedostatky:

- aplikace se dlouho načítá,
- není možné provést veškeré automatické opravy najednou,
- nestandardní ovládání (např. opačný směr ovládání přiblížení, než je obvyklé, viz sekce 3.6.3), celkově obtížná orientace v uživatelském rozhraní.[2]

Netfabb Studio Basic

Netfabb Studio Basic je volně dostupnou verzí programu netfabb Studio. Jedná se o komplexní nástroj určený k prohlížení, manipulaci s modely nejen ve formátu STL a k jejich automatické opravě.[7] Automatické opravy zahrnují vyplnění děr, opravy orientace facetů (a s nimi spojených normálových vektorů) a opravy dalších chyb spojených s nemanifoldností objektu. V netfabbu je také možné pracovat s jednotlivými facety (ručně je přidat, odstranit nebo otočit jejich orientaci). Na rozdíl od vytvářené aplikace ADMeshGUI však není netfabb Studio Basic open-source programem.[16]

[2] Jedná se o názor autora práce, není podloženo dalším výzkumem.

Magics

Magics[17] společnosti Materialise je na rozdíl od výše uvedených volně dostupných aplikací proprietárním komerčním produktem . Magics je zaměřen na profesionální 3D tisk a náročného uživatele. Schopnosti aplikace sahají od zobrazení chybných částí modelu přes jeho editaci a automatické opravy až po finální přípravu modelu. Přestože existují jiné aplikace, které pokrývají určité části nástroje Magics, neexistuje další aplikace, která by obdobně zvládala kontrolu nad celým procesem přípravy modelu pro 3D tisk.[18]

Společnost Materialise nabízí také na vyžádání volně dostupný nástroj MiniMagics. MiniMagics umí zobrazit chyby modelu a upozornit na ně, neumožňuje však tyto chyby odstranit.[19]

Jiné

Mezi další, blíže nerozebrané nástroje pro manipulaci se soubory ve fromátu STL, patří:

- STLFix[6],
- Cloud netfabb[6],
- ReMESH[20],
- TriMM[20],
- MeshFix[20],
- Polymender[20].

3.3 ADMesh

Pro účely základní manipulace se soubory ve formátu STL a opravy chyb v jejich topologii, které omezují či zcela znemožňují produkci výsledných fyzických modelů, byl vytvořen open-source nástroj ADMesh[21]. Původní verze vytvořená Anthony D. Martinem na přelomu let 1995–1996 poskytovala rozhraní pouze pro příkazovou řádku. V současné době poskytuje ADMesh také knihovní rozhraní pro programovací jazyky C a Python.

3.3.1 Funkcionalita

Již ze své podstaty řádkově orientované aplikace není ADMesh editorem určeným k vytváření nových 3D modelů, umožňuje však s nimi provádět základní manipulace včetně oprav topologie v širokém nastavení. Následující seznam shrnuje funkcionality ADMeshe, které se zároveň staly předmětem zkoumání, neboť jejich začlenění do nově vytvářené aplikace se výrazně odráží i v rozvržení grafického uživatelského rozhraní.

1. **Manipulace**

   Translace – umožňuje s modelem pohybovat v soustavě souřadnic podle os x, y, z. Umožňuje jak relativní translaci (vzhledem k aktuální pozici modelu), tak posun do počátku soustavy souřadnic (model je zarovnán k počátku dle minimálních souřadnic ve směru jednotlivých os).

   Rotace – umožňuje s modelem rotovat podél os x, y, z proti směru hodinových ručiček o zadaný úhel.

   Zrcadlení – umožňuje model zrcadlově převrátit podél roviny určené osami x a y, x a z, y a z.

   Škálování – umožňuje model škálovat (měnit velikost) jednotlivě ve směru os x, y, z nebo globálně.

   Sloučení – umožňuje model sloučit s dalším modelem ve formátu STL určeným jménem souboru.

2. **Opravy**

   Zaplnění děr – chybějící facety modelu jsou vyplněny.

   Oprava směru normál – normálové vektory facetů jsou přepočítány, aby směřovaly ven z objektu (odpovídá orientaci vrcholů facetu proti směru hodinových ručiček).

   Oprava velikosti normál – normálové vektory jsou přepočítány, aby odpovídaly jednotkovým vektorům kolmým k příslušnému facetu.

   Odstranění degenerovaných facetů – facety s jedním a více shodnými vrcholy jsou odstraněny.

   Odstranění nepřipojených facetů – facety bez dalšího sousedního facetu jsou odstraněny.

   Oprava facetů – umožňuje opravit facety propojením s dalšími blízkými facety v určeném dosahu.

   Převrácení – umožňuje převrátit směr všech facetů a normál.

3. **Výstup**

   Uložení – model je uložen ve formátu STL, který je nástrojem možné znovu zpracovat (podporuje jak binární, tak ASCII verzi STL formátu), případně je exportován ve formátu OBJ, DXF, VRML nebo OFF.

3.3.2 Ukázka použití

Následující příkaz demonstruje použití nástroje ADMesh na vstupním souboru `example.stl`. STL model definovaný v tomto souboru je načten, otočen podle osy x proti směru hodinových ručiček o 90°, dvakrát zvětšen, jsou zaplněny díry v topologii, opraveny velikosti a směr normálových vektorů a výsledný model je uložen ve formátu ASCII STL do souboru `output.stl`.

```
admesh --x-rotate=90 --scale=2 --fill-holes \
       --normal-directions --normal-values \
       --write-ascii-stl=output.stl example.stl
```

V nástroji ADMesh je definována i zkrácená verze přepínačů:

```
admesh --x-rotate=90 --scale=2 -fdv \
       -a output.stl example.stl
```

Je však zřejmé, že i zkrácená verze klade u uživatele nemalé nároky na znalost přepínačů a neposkytuje vizuální přehled nad provedenými změnami.

3.3.3 Rozšíření nástroje ADMesh

Existují projekty[3], které využívají rozhraní poskytované nástrojem ADMesh. Obvykle se jedná o samostatné nástroje orientované na použití v příkazové řádce. Zatímco některé cílí na usnadnění konkrétních operací nástroje ADMesh (např. nástroj `snapz`[22] slouží k posunu modelu na nulovou souřadnici ve směru osy z, což má vliv na další fáze přípravy modelu pro 3D tisk), jiné přidávají zcela novou funkcionalitu. Příkladem takových nástrojů mohou být `stlsplit`[23] sloužící k rozdělení modelu na samostatné modely dle jednotlivých uzavřených částí nebo `stlcut`[24] určený k rozdělení tělesa dle zadané roviny.

Začleněním existujících rozšíření do ADMeshGUI nad rámec předem stanovených funkčních požadavků je možné zvýšit univerzálnost aplikace. Bylo rozhodnuto minimálně o přidání nástroje `stlsplit`. Nástroj `stlcut` bohužel není v současné době zcela dokončen, a proto není do ADMeshGUI přidán. Nástroj `snapz` není nutné přidávat, neboť jeho funkcionalitu je možné zajistit voláním funkce translace se specifickými parametry.

3.4 Možnosti řešení

Nástroj ADMesh poskytuje knihovní rozhraní pro jazyky C a Python. Knihovnu napsanou v jazyce C je dále možno využít v objektově orientovaném jazyce C++. Pro všechny tři programovací jazyky existují frameworky určené k tvorbě uživatelského rozhraní. Výběr frameworku byl částečně omezen

[3]Seznam projektů je dostupný na adrese https://github.com/admesh/admesh-projects.

Tabulka 3.1: Zvažovaná řešení

Jazyk	Framework pro GUI	Knihovna pro 3D obsah
`Python`	`PyQt`	`PyOpenGL`
`C`	`GTK+`	`OpenGL`
`C++`	`Qt`	`OpenGL`

podmínkou jeho svobodného použití pro open-source tvorbu, vyplývající z požadavků kladených na výslednou aplikaci. Framework musel navíc splňovat podmínku platformní nezávislosti. Na základě těchto omezení byla volba frameworku zúžena na GTK+ pro jazyk C, Qt pro jazyky C++ i Python a Tkinter pro Python.

Jelikož má výsledná aplikace umožnit prohlížení 3D modelů, byla dalším nezbytným krokem volba knihovny, která umožňuje vykreslování 3D obsahu. Na tuto knihovnu byly kladeny stejné nároky jako na framework pro tvorbu uživatelského rozhraní, tedy svobodné použití a multiplatformnost. Vhodným řešením je knihovna OpenGL v kombinaci s jazykem GLSL pro psaní shader programů.

Závěrečná zvažovaná řešení jsou shrnuta v tabulce 3.1.

3.5 Zvolené řešení

Prozkoumáním knihovních rozhraní nástroje ADMesh bylo zjištěno, že knihovna pro Python neposkytuje na rozdíl od knihovny určené pro jazyk C kompletní funkcionalitu ADMeshe. Vzhledem k tomuto faktu a předchozím zkušenostem s jazykem C++ bylo přistoupeno k řešení v jazyce C++ za podpory frameworku Qt, který zároveň umožňuje přímé použití knihovny OpenGL.

Společně s frameworkem Qt jsou k dispozici i vývojová prostředí Qt Creator (pro vytváření aplikací v jazyce C++) a Qt Designer[4] (pro vytváření uživatelského rozhraní). S cílem usnadnit tvorbu aplikace byla tato vývojová prostředí využívána po celou dobu implementace.

Pro umožnění budoucí lokalizace aplikace do dalších jazyků byl zvolen balíček `gettext`.

3.6 Uživatelské rozhraní

V oboru HCI (Human–computer interaction / Styk člověka s počítačem) se můžeme setkat s termínem „použitelnost artefaktu“. Použitelnost je kvalitativní vlastnost hodnotící, jak snadno se artefakt používá.[25] Artefaktem rozumíme cokoliv vyrobeného člověkem za nějakým účelem (může se jednat o software, dům, ...).[26] V kontextu této práce je artefaktem aplikace

[4] zahrnut i v Qt Creatoru

ADMeshGUI. Aby byla zvýšena použitelnost aplikace, byl její návrh proveden s ohledem na typického uživatele.

3.6.1 Typický uživatel

Vzhledem k typu sledovaného artefaktu a jeho úzkému zaměření na konkrétní souborový formát STL lze předpokládat, že typický uživatel bude středně pokročilý až pokročilý v ovládání počítače. STL formát je používán zejména v oblasti 3D tisku, uživatel bude pravděpodobně obeznámen alespoň na minimální úrovni i s jeho problematikou. Lze usuzovat, že výslednou aplikaci využije ve fázi přípravy modelu pro tisk a použije aplikaci pro konkrétní, předem známý úkon.

Vstupní STL soubor pro aplikaci může uživatel získat stažením z online repozitáře modelů nebo ho může sám vytvořit v libovolném 3D editoru, který umí modely do formátu STL exportovat. Je velmi pravděpodobné, že se uživatel orientuje ve sféře počítačové 3D grafiky, případně aktivně používá některý z 3D editorů a je zvyklý na ovládání 3D scény tohoto editoru.

3.6.2 Cíle uživatele

Při návrhu uživatelského rozhraní je stěžejním úkolem analýza klíčových operací prováděných s daným artefaktem. Výstupem této analýzy je seznam cílů, kterých chce uživatel typicky použitím daného artefaktu dosáhnout. Na základě cílů je následně možné optimalizovat jednotlivé části artefaktu tak, aby k jejich dosažení uživatel potřeboval co nejméně kroků a aby způsob dosažení odpovídal zvyklostem obdobných řešení. Následuje seznam předpokládaných uživatelských cílů při použití aplikace ADMeshGUI:

1. Zobrazit model ve formátu STL a vytvořit si představu o jeho geometrii.
2. Provést úpravu modelu nebo sekvenci úprav. Úpravou může být rotace, translace, změna velikosti, zrcadlení, sloučení nebo rozdělení.
3. Provést automatickou opravu modelu (např. pro potřeby 3D tisku).
4. Provést opravu modelu se specifickým nastavením.
5. Provést libovolnou činnost z bodů 1 – 4 s více modely zároveň.

3.6.3 Průzkum ovládání pohledu ve 3D editorech

Editory 3D modelů typicky obsahují hlavní okno s 2D náhledem na scénu. Tento 2D náhled vzniká projekcí aktuálního stavu scény do prostoru zobrazovacího zařízení. Aby bylo možné model prohlížet z různých úhlů, vzdáleností, pozic, případně aby bylo možné model vybrat pro další činnost, mají editory definováno ovládání pro interakci s 3D scénou. Jelikož vznikající aplikace

Tabulka 3.2: Výchozí ovládání 3D scény vybraných editorů. Tabulka obsahuje na posledním řádku pro porovnání i zvolené výchozí ovládání vytvářené aplikace. LMB – levé tlačítko myši, RMB – pravé tlačítko myši, MMB – prostřední tlačítko myši. Výběr objektů je obvykle umožněn i mimo scénu ve speciálním menu

Editor	Rotace	Translace	Výběr	Měřítko
Blender	`MMB`	`Shift + MMB`	`RMB (+ Shift)`	`MMB roll`
OpenSCAD	`LMB`	`RMB`	`/`	`MMB roll`
netfabb Basic	`RMB`	`MMB`	`LMB / RMB`	`MMB roll`
SketchUp	`MMB`	`Shift + MMB`	`LMB (+ Shift)`	`MMB roll`
MeshLab	`LMB`	`MMB`	`/`	`MMB roll (r)`
ADMeshGUI	`LMB`	`MMB`[5]	`RMB (+ Shift)`	`MMB roll`

okno prohlížeče obsahuje, bylo nezbytné provést průzkum ovládání stávajících editorů, aby se nová aplikace výrazně neodlišovala od zavedených zvyklostí.

V tabulce 3.2 je uveden přehled ovládání 3D scény získaný vyzkoušením vybraných 3D editorů. Pro přehlednost byla do tabulky doplněna i aplikace ADMeshGUI. Data v tabulce ukazují, že k ovládání 3D scény je používána primárně myš (obvykle tři základní tlačítka). Časté je také použití klávesy Shift pro odlišení různých úkonů při použití stejného tlačítka myši. Je však zřejmé, že shodné úkony v různých editorech nejsou prováděny přesně stejným tlačítkem (jedinou výjimkou je ovládání změny měřítka, které bylo u zkoumaných editorů vždy realizováno otáčením kolečka myši; u programu MeshLab pak byla výchozí reakce na otočení kolečkem myši opačná než u ostatních programů).

Jelikož formát STL umí použít či exportovat většina 3D editorů, není možné rozhodnout, který editor využívá uživatel zároveň používající aplikaci ADMeshGUI. Při dodržení základních principů (tlačítka myši + klávesa Shift) odpovídá ovládání interakce s 3D scénou zvyklostem. Bylo zvoleno řešení využívající levé tlačítko myši pro rotaci, pravé tlačítko myši pro výběr jednoho objektu, pravé tlačítko myši v kombinaci s klávesou Shift pro výběr více objektů, prostřední tlačítko myši nebo Shift + levé tlačítko myši (platforma Mac OS X nepodporuje prostřední tlačítko myši) pro translaci a otáčení kolečkem myši pro změnu měřítka.

3.6.4 Návrh rozhraní

S využitím poznatků získaných během analýzy bylo možné vytvořit prototyp aplikace znázorňující rozložení ovládacích prvků. Prvotní návrh byl realizován pomocí tužky a papíru a následně převeden pomocí aplikce Pencil do digitální

[5]Pro umožnění translace na platformě Mac OS X je zároveň možné použít kombinaci Shift + LMB.

podoby, viz obrázek 3.1 na straně 22. Hlavní okno aplikace bylo rozděleno do několika základních částí:

- horní menu,
- levé menu obsahující panel nástrojů a seznam otevřených souborů,
- prohlížeč 3D modelů,
- pravé menu obsahující ovládací prvky pro ADMesh.

Za účelem usnadnění prohlížení modelů (větší okno prohlížeče, méně rušivých prvků) bylo rozhodnuto o možnosti zavřít levé i pravé menu. Pravé menu bylo navrženo dle funkcionalit ADMeshe, obsahuje tak do jednotlivých celků seřazené nástoje pro manipulaci s modelem a nástroje pro opravy. Pravému menu ve spodní části dominuje výrazné tlačítko pro opravu modelu s výchozí volbou pro kompletní opravu, aby bylo možné automatickou opravu provádět co nejrychleji.

Bylo nezbytné rozhodnout o výchozích barevných kombinacích použitých při vykreslování modelu. Model se může nacházet ve dvou základních stavech, které je potřeba odlišit, může být aktivní (zvolený pro úpravy) nebo neaktivní. Může také obsahovat zvýraznitelné chyby. Aplikace se může nacházet ve třech módech zobrazení, přičemž jsou vykreslovány pouze hrany, facety nebo facety se zvýrazněnými hranami.

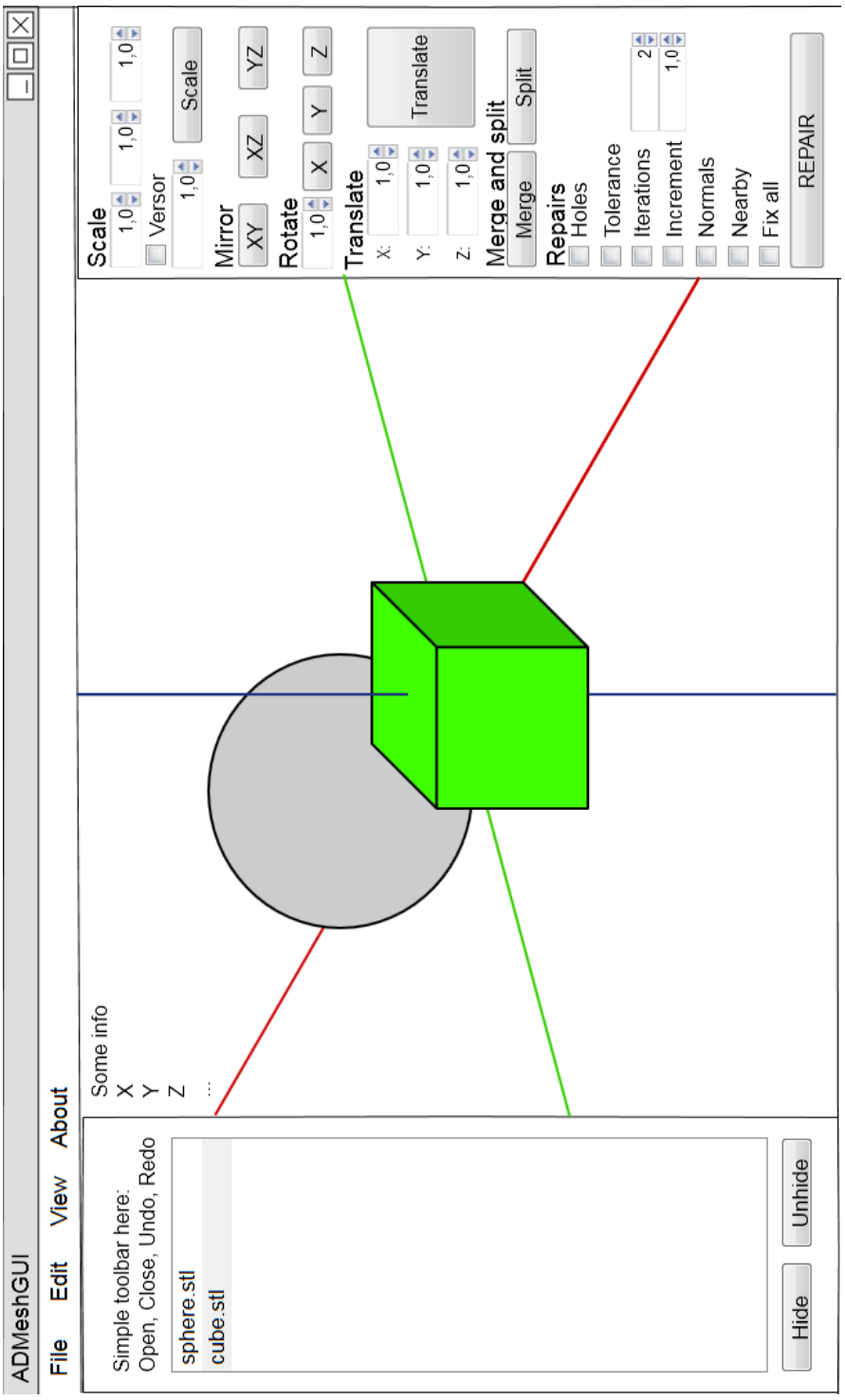
Výchozí barvy byly voleny s ohledem na snadné odlišení a vysoký kontrast.[6] Zvýraznění chyb se projevuje pouze u aktivních objektů.

- Drátový mód zobrazení: aktivní objekty černé, neaktivní šedé, chyby nezohledněny
- Plný mód zobrazení: aktivní objekty zelené, neaktivní šedé, chyby červené
- Plný mód zobrazení se zvýrazněnými hranami: aktivní objekty zelené s černými hranami, neaktivní objekty šedé, chyby červené

3.6.4.1 Ikony

Pro panel nástrojů obsažený v levém menu aplikace byla vytvořena sada ikon. Jednotlivé ikony (pro operace Přidat soubor, Uložit, Vrátit zpět změny, Znovu provést změny a Zavřít soubor) byly navrženy dle standardně používaných symbolů, avšak s důrazem na minimalistické grafické zpracování. Tyto ikony se zobrazují pouze v případě, že systém, na kterém je aplikace spuštěna, nepodporuje načítání systémových ikon.

[6] Barvu aktivních modelů a chyb v obou plných módech zobrazení je dále možné specifikovat v dialogovém okně Vlastnosti.



Obrázek 3.1: Návrh rozložení ovládacích prvků aplikace vytvořený v nástroji Pencil

3.7 Návrh tříd

Před započetím implementace byl vytvořen návrh tříd, které rozdělují aplikaci do jednotlivých logických celků. Dle tohoto návrhu byla aplikace ADMeshGUI implementována.

Window představuje hlavní okno aplikace. Stará se o vytvoření uživatelského rozhraní, načtení a uložení stavu aplikace při startu a ukončení a propojení signálů akcí uživatele (např. stisknutí tlačítka) s metodami, které vykonávají příslušné akce v aplikaci.

PropertiesDialog představuje dialogové okno umožňující provést základní nastavení aplikace.

RenderingWidget reprezentuje okno prohlížeče a přímo pracuje s prvky knihovny OpenGL. Stará se o hlavní vykreslování scény, stav scény a o obsluhu uživatelských akcí v okně prohlížeče (např. tah myši pro rotaci). Výsledná realizace prohlížeče je blíže popsána v sekci 4.1 kapitoly *Implementace*.

AdmeshController se stará o obsluhu veškerých akcí souvisejících s načtenými STL objekty. Tato třída udržuje aktuální seznam načtených objektů a má přístup k historii úprav. `AdmeshController` obsluhuje výběr objektů (blíže popsáno v sekci 4.2 kapitoly *Implementace*); akce (např. Uložit, Rotovat, Opravit, ...) se provádí pouze nad vybranými objekty.

HistoryList představuje historii úprav. Stará se o uložení stavu scény a o přístup k neaktuálním položkám. Řešení seznamu historie je popsáno v sekci 4.3 kapitoly *Implementace*.

MeshObject zapouzdřuje strukturu `stl_file` ADMeshe, která obsahuje data načteného modelu. Třída slouží zejména k usnadnění manipulace s načtenými modely.

Kromě jednotlivých tříd návrh počítá s hlavičkovým souborem udržujícím definice výchozích hodnot. Tento soubor slouží k usnadnění jejich případné editace bez nutnosti hledat každé použití dané hodnoty v jednotlivých souborech implementace.

Kapitola 4

Implementace

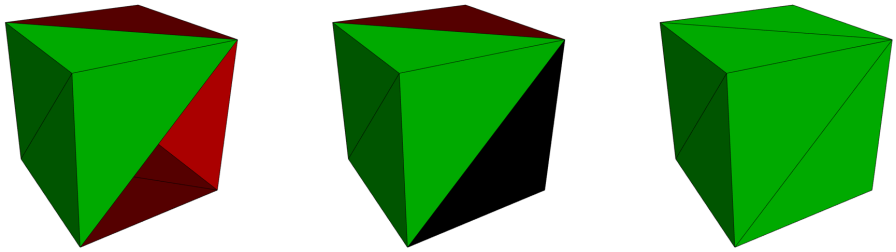
V této kapitole jsou zachyceny problematické nebo zajímavé aspekty implementace výsledné aplikace. Nejedná se o popis obvyklých implementačních zákonitostí ani celkových pravidel implementace v jazyce C++. Informace o třídách a metodách potřebných při implementaci byly čerpány z rozsáhlé dokumentace[27] frameworku Qt.

4.1 Prohlížeč, vykreslování objektů

Vykreslování 3D scény v aplikaci je dosaženo za pomoci knihovny OpenGL. Integrace OpenGL do frameworku Qt dále usnadňuje její použití, bylo však nutné vytvořit vazbu mezi daty modelu dodanými z rozhraní ADMeshe ve struktuře `stl_file` a samotným procesem vykreslování. Za tímto účelem (společně se snazší celkovou manipulací) byla vytvořena třída `MeshObject` zapouzdřující strukturu `stl_file`. Každá instance třídy `MeshObject` zároveň obsahuje další informace o modelu a číslo vlastního VBO (Vertex Buffer Object), které je instanci `MeshObject` přiděleno při vytvoření. VBO slouží k uložení dat do grafické paměti a oproti staršímu přímému způsobu urychluje vykreslování.

Problémem se ukázalo uložení dat vrcholů a normál ve struktuře `stl_file`. Struktura obsahuje pole struktur reprezentujících facet a každý facet obsahuje informaci o třech svých vrcholech a jednom normálovém vektoru. Toto rozdělení vychází z formátu STL, pro uložení do VBO je však nutné poskytnout normálový vektor každému vrcholu[7]. Při použití přímého způsobu vykreslování by bylo možné použít přímo data z dané struktury, jedná se ovšem o zastaralý a neefektivní způsob. Data struktury `stl_file` jsou tedy při zavolání metody `updateGeometry()` zkopírována do nového dočasného pole obsahujícího pouze číselné hodnoty v pořadí `vrchol.x`, `vrchol.y`, `vrchol.z`, `normála.x`, `normála.y`, `normála.z` pro všechny původní vrcholy za sebou. Toto dočasné pole je následně zkopírováno do VBO a připraveno na vykreslování.

[7] je možno normálový vektor i zcela vynechat



Obrázek 4.1: Znázornění postupných kroků oprav v prostředí aplikace ADMeshGUI. Zleva: před opravou, po zaplnění děr, po opravě normálových vektorů

Pro vyvrácení oprávněných obav o vzrůstající časovou náročnost spojenou s dvojím kopírováním byl prohlížeč implementován nejprve zastaralým přímým způsobem a až následně upraven pro použití VBO a shader programů.

Porovnáním měření obou přístupů bylo potvrzeno předpokládané chování. Při načtení stejného souboru vykazovala aplikace využívající starý přímý přístup vyšší zátěž CPU než aplikace využívající VBO. Naměřená zátěž byla přibližně trojnásobná. U aplikace s VBO nebyla zaznamenána viditelná prodleva způsobená dvojím kopírováním. Dvojité kopírování je navíc prováděno pouze jednorázově při změně dat struktury `stl_file` (načtení, manipulace, opravy) na rozdíl od vykreslování, které je často potřeba realizovat několikrát za vteřinu. Výhody vykreslování z grafické paměti tedy převažují nad nevýhodou dvojího kopírování.

Samotné vykreslování je realizováno dle dále uvedeného scénáře. Třída reprezentující prohlížeč (`RenderingWidget`) požádá třídu spravující kontejner všech objektů (`AdmeshController`) o vykreslení voláním metody `drawAll()`, která nastaví zvolený mód zobrazení a postupně vykreslí veškeré držené objekty `MeshObject` voláním metody `drawGeometry()`. Pro daný objekt je zvolen příslušný VBO a za pomoci dodaného shader programu je objekt vykreslen. Vykreslovací shader program aplikace se skládá z vertex (VS) a fragment (FS) shaderu. VS pouze vynásobí pozici vrcholu MVP maticí a přepošle společně s normálovým vektorem k dalšímu zpracování FS. FS kromě pozice a normály obdrží také dvě výchozí barvy (pro případné odlišení chybně orientovaných částí modelu). FS používá tzv. flat shading (stínování, při kterém má celý facet jednu barvu) a pro další urychlení nepočítá žádné osvětlení. Aby se modely jevily prostorové a aby byla orientace v topologii snazší, je pro vytvoření drobných rozdílů v odstínu barvy využito normálového vektoru a následujícího výpočtu:

```
float factor = (N.x + N.z + N.y + 3.0) / 6.0;
```

Získaným faktorem je přenásobena barva (zvolena podle rozhodnutí, má-li být facet vykreslen jako chybový) a nastavena na výstup FS. Faktor nabývá maximální hodnoty 1 a minimální hodnota je vždy větší než 0, čímž je zaručeno, že výsledná barva nebude příliš tmavá. Je-li nastavena barva černá (všechny složky nulové), faktor barvu nijak neovlivní.

4.2 Výběr objektů

Výběr objektů neboli picking byl implementován pomocí frame buffer objektu (FBO). FBO se skládá z několika částí, které dohromady tvoří výslednou obrazovou paměť. Výsledky veškerého vykreslování v OpenGL se ukládají do výchozího FBO, který od OpenGL verze 4.0 obsahuje minimálně paměť barvy, hloubky a šablony. Obvykle se používají dva FBO zároveň pro zvýšení plynulosti zobrazení – zatímco se do jednoho zapisuje, druhý je zobrazen.[28] Při implementaci výběru objektů bylo přistoupeno k vytvoření vlastního FBO, který se však ve výsledné aplikaci nezobrazuje (tzv. offscreen FBO) a využití jeho barevné složky.

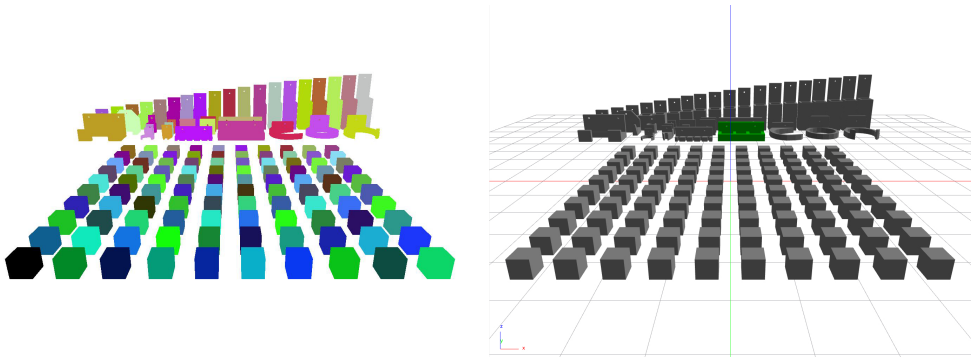
Při požadavku na výběr objektu (kliknutí myší do prostoru prohlížeče) je zavolána metoda `doPicking()`, v níž je vytvořen nový FBO pomocí třídy `QOpenGLFramebufferObject`. Pro požadavky aplikace je dostačující, pokud takto vytvořený FBO obsahuje paměť barvy a paměť hloubky (vybírán je pouze objekt, který není překryt). Následně jsou do tohoto FBO vykresleny veškeré objekty ve scéně pomocí vlastního shader programu. Příslušný VS aplikuje transformace shodné s transformacemi ve vykreslované scéně a FS pouze nastaví vstupní barvu na výstupní.

Barva je v tomto ohledu klíčová pro identifikaci. Při postupném vykreslování objektů je každému nastavena jedinečná barva vycházející z indexu v kontejneru všech objektů. Jakmile jsou všechny objekty vykresleny, z daného FBO se přečte hodnota odpovídající souřadnicím myši při kliknutí. Ze získané barevné hodnoty je spočítán odpovídající index objektu, se kterým bude dále manipulováno.

Při uvažování standardní bitové hloubky 8 bitů na barevný kanál a použití tří barevných kanálů (R, G, B) je možné tímto způsobem získat index s maximální hodnotou 16646654.[8] Tento počet je zcela postačující, při předpokládaném použití aplikace reálně nedosažitelný.

Obrázek 4.2 byl vytvořen uložením výběrového FBO na disk. Rozdíly mezi barvami jednotlivých objektů byly záměrně zvýšeny pro demonstraci principu.

[8] Hodnota byla získána pomocí vzorce $(255 + 255^2 + 255^3)$ a po odečtení 1 pro maximální hodnotu bílého pozadí.



(a) Výstup FBO (b) Odpovídající scéna v okně prohlížeče

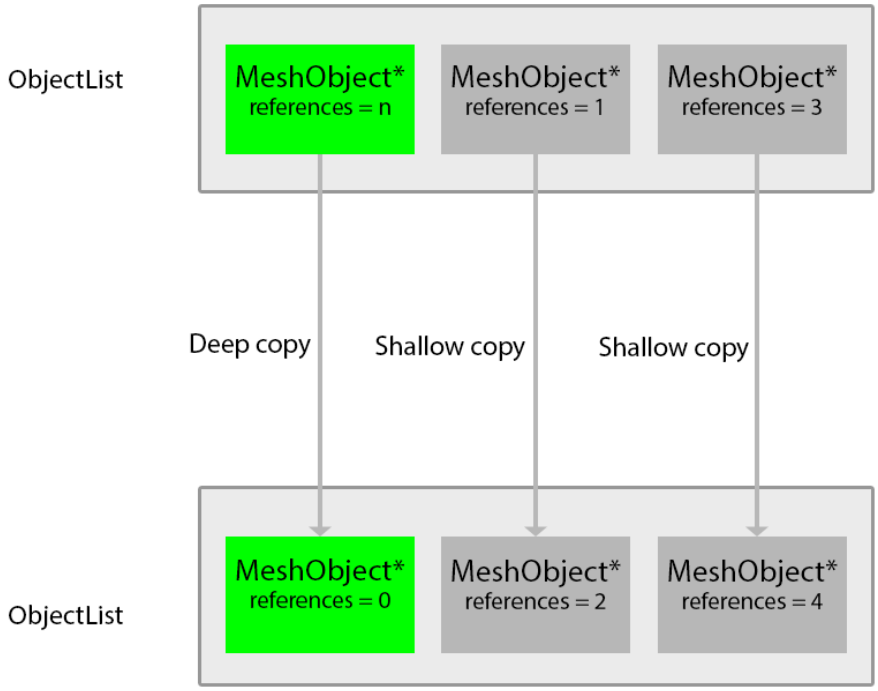
Obrázek 4.2: Rozdíly ve vykreslení objektů do výběrového FBO a do okna prohlížeče. Každému objektu ve scéně je přidělena jedinečná barva, která objekt identifikuje v FBO. Pro názornější demonstraci principu byly rozdíly mezi barvami objektů v FBO zvýšeny přenásobením indexu hodnotou 100000. Tímto způsobem bylo možné při daném počtu objektů získat i jiné barvy než odstíny modré. Krychle v levém dolním rohu se nachází na indexu nula; nulový index se mapuje na černou barvu

4.3 Historie úprav

Aby mohl uživatel aplikace vrátit zpět provedené změny (undo), případně tyto odvolané změny opětovně vyvolat (redo), bylo potřeba implementovat mechanismus undo-redo bufferu. Byla zvažována následující řešení:

- Historie obsahuje textový řetězec popisující provedenou operaci. Při undo / redo se provede operace inverzní.
- Historie obsahuje přímo záznamy objektů. Při undo / redo jsou aktuální objekty nahrazeny.

Z podstaty aplikace bylo nevhodné použít první způsob, neboť opakované inverzní operace by mohly mít nežádoucí vliv na přesnost desetinných čísel. Přestože jsou chyby v reprezentaci čísel v plovoucí řádové čárce malé, i jednoduché výpočty mohou mít na přesnost výsledku výrazný vliv. Sčítání a odčítání jsou nebezpečné operace, neboť při použití čísel odlišných řádů budou číslice menšího z nich ztraceny. Problém s přesností čísel se také stává závažnějším s narůstajícím počtem výpočtů.[29] Modifikace objektu použitím inverzní operace při průchodu seznamem historie by mohla zapříčinit vznik objektu odlišného od objektu výchozího. Tento problém by se dále násobil, opakoval-li by uživatel sekvenci úkonů: provést úpravu, vrátit změny zpět. Pro opravy navíc není možné definovat inverzní operace.

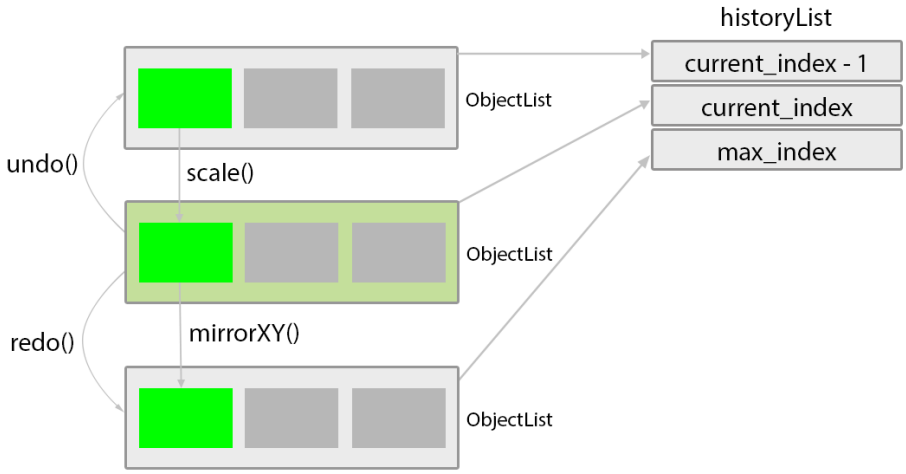


Obrázek 4.3: Grafické znázornění metody `renewList()`

Druhý způsob by naopak mohl mít příliš vysoké paměťové nároky, bylo tedy přistoupeno k jeho optimalizaci. Optimalizace spočívá zejména ve způsobu kopírování objektů do seznamu historie a ořezávání nejstarších záznamů při dosažení paměťového limitu.

Stav scény je reprezentován seznamem ukazatelů na všechny aktuálně otevřené objekty, přičemž každý objekt je buď aktivní (operace se nad ním provádí) nebo neaktivní (operace na daný objekt nemají vliv). Je-li uživatelem vyvolán požadavek na operaci s objekty, provede se nejprve kopie tohoto seznamu. O vytvoření kopie se stará metoda `renewList()`, která využívá počítání referencí. Ukazatele na neaktivní objekty jsou zkopírovány přímo, daným objektům se zvedne počet referencí o 1. U aktivních objektů je provedena hluboká kopie a vynulování referencí. Nově zkopírovaným seznamem je nahrazen seznam původní a je nad ním provedena požadovaná operace. Po dokončení operace je seznam vložen na konec historie. Konkrétní situaci znázorňuje obrázek 4.3, aktivní objekty jsou vyobrazeny zeleně.

Požádá-li uživatel o provedení operace undo, nahradí se aktuální seznam předchozím záznamem v historii. Všechny záznamy nadále zůstávají v historii, aby byla umožněna operace redo, která naopak aktuální seznam nahradí záznamem následujícím v historii.



Obrázek 4.4: Grafické znázornění seznamu historie a operací undo / redo

Bylo nezbytné vyřešit chování aplikace při situaci *operace 1 – undo – operace 2*. V tomto případě je záznam *operace 1* nahrazen záznamem *operace 2*. Aby byla správně uvolněna veškerá paměť, byla vytvořena metoda `deleteRow()` starající se o smazání jednoho záznamu v historii. Položky, jejichž počet referencí je roven nule, metoda odstraní, ostatním je počet referencí snížen o 1.

Obdobným způsobem bylo zajištěno velikostní omezení seznamu historie. Protože jednotlivé objekty mohou být datově objemné a velikost historie s každou provedenou operací narůstá, je nezbytné při překročení velikosti historie smazat nejstarší záznamy. Velikost historie je možné nastavit v dialogovém okně Vlastnosti.

4.4 Qt verze 5.4

Během tvorby prohlížeče bylo zjištěno, že aktuálně používané objekty typu `QGLWidget` budou od příští verze (5.4) frameworku Qt[30] zastaralé a budou nahrazeny novějšími `QOpenGLWidget`. Po dohodě s konzultantem bylo odstoupeno od návrhu použít beta-verzi Qt5.4. Čekání na novou verzi frameworku by zároveň znemožnilo další rozvoj projektu, prvotní vývoj byl proto uskutečněn ve starší verzi 5.3. S příchodem nové verze v termínu 10.12.2014 byl projekt upraven.

4.4.1 Provedené úpravy

Náhrada `QGLWidget` za `QOpenGLWidget`

Výměnou zastaralé třídy zajišťující vykreslování OpenGL za její moderní protějšek došlo k zajištění vyšší budoucí podpory aplikace. Tato změna dále umožnila jednodušší překreslování scény dodatečnou 2D grafikou (použito pro zobrazení informačního textového panelu) přímo v reimplementaci přetížené metody `paintGL()` namísto původní kombinace `paintGL()` (pro 3D obsah) s `paintEvent()` (pro 2D obsah).

Následující ukázka zjednodušeně demonstruje způsob vykreslování 3D i 2D grafického obsahu zároveň v reimplementaci metody `paintGL()` zděděné z třídy `QOpenGLWidget` s použitím objektu třídy `QPainter`. Po zavolání metody `beginNativePainting()` na objektu třídy `QPainter` je možné nastavit parametry OpenGL, připojit shader programy a vykreslit libovolný 3D obsah. Před zavoláním metody `endNativePainting()` je pro zajištění správného následného vykreslování 2D obsahu nezbytné uvolnit shader programy a vrátit nastavení OpenGL do původního stavu.

```
void RenderingWidget::paintGL()
{
    QPainter painter;
    painter.begin(this);
    painter.beginNativePainting();
        // draw 3D content here
    painter.endNativePainting();
        // draw 2D content here
    painter.end();
}
```

4.5 Merge

Aplikace ADMeshGUI měla dle funkčních požadavků podporovat také operaci sloučení (merge) dvou nebo více modelů. Průzkumem funkce `stl_open_merge` ADMeshe bylo zjištěno, že je tato funkce v ADMeshGUI nepoužitelná. Funkce `stl_open_merge` je založena na čtení dat (informace o facetech) jednoho modelu a jejich připojení na konec dat modelu druhého. Funkce však načítá data připojovaného modelu pomocí ukazatele na soubor. Problém v ADMeshGUI nastane, pokud není soubor na disku aktuální (nebyl uložen) nebo neexistuje. Ilustrujme případ na situaci, kdy uživatel s modelem bez uložení manipuluje (např. rotace) a následně jej sloučí. Dojde ke sloučení s původním (a nikoliv rotovaným) modelem. Obdobný problém nastává pří násobném slučování. Soubory není možné průběžně ukládat bez vědomí uživatele.

Jako nejjednodušší řešení problému bylo zvoleno vytvoření nové funkce `stl_merge`, která pracuje s již načtenými modely ve strukturách `stl_file` ADMeshe. Tato funkce přidává na konec dat prvního modelu data druhého modelu a spočítá celkové informace o sloučeném modelu (např. rozměry, počet již opravených částí, ...).

Kapitola 5

Testování

Již během implementace byla aplikace ADMeshGUI průběžně testována. Vždy po začlenění nové funkcionality byla přidaná část aplikace otestována na několika různých vstupních souborech. Následuje souhrn závěrečných testů, prováděných ve finální fázi implementace.

5.1 Test multiplatformnosti

Aplikace byla úspěšně spuštěna a otestována na následujících operačních systémech:

- Ubuntu 14.04 LTS
- Fedora 21
- Mac OS X 10.9 Mavericks
- Mac OS X 10.10 Yosemite
- Windows 7 64-bit

Během testů na různých platformách byly odhaleny a odstraněny následující problémy:

- **Při výběru objektu problikávající okno prohlížeče (Windows 7 64-bit)** – problém byl odstraněn přesunem volání metody pro obsluhu výběru objektu až za standardní vykreslení scény.
- **Prvky prohlížeče vykreslené pouze ve čtvrtině celkového prostoru okna (Mac OS X s Retina displayem)** – odstraněno nastavením velikosti okna prohlížeče v závislosti na zobrazovacím zařízení. Návratovou hodnotou volání `this->devicePixelRatio()` je na zařízení s Retina displayem celé číslo 2, na ostatních zařízeních 1.

```
w = width * this->devicePixelRatio();
h = height * this->devicePixelRatio();
glViewport(0, 0, w, h);
```

5.2 Test správy paměti

K otestování správného uvolňování přidělené paměti aplikací byl zvolen nástroj valgrind. Framework Qt (zejména jeho vykreslovací funkce) však používá správu paměti, která způsobuje, že valgrind při výchozím nastavení detekuje úniky paměti. V prostředí Qt Creatoru je možné k otestování použít valgrind, který chyby frameworku Qt potlačuje. Při tomto způsobu testování nebyly v aplikaci nalezeny žádné úniky paměti.

5.3 Test zátěže

Pro zátěžové testy byl použit notebook s následujícími parmetry:

- Intel i5-3210M
- 6GB RAM
- Nvidia GeForce 630M

V rámci tohoto testu byly provedeny dva dílčí testy: test souborem s velkým počtem polygonů a test větším počtem souborů. Tyto soubory byly cíleně pro potřeby testů vytvořeny v aplikaci Blender a exportovány do formátu STL. S cílem získat přesné výsledky byl do aplikace přidán čítač, který počítá čas potřebný pro vykonání hlavních funkcí programu. Čítač je dostupný i ve výsledné aplikaci, pokud je sestavena v režimu ladění.

Prvním souborem byl detailní model koule s celkovým počtem polygonů přesahujícím 1 milion. Operace prováděné nad tímto modelem vykazovaly na testovací sestavě časy od 0,25 s pro translaci do 1,52 s pro opravy (přesné časy uvedeny v tabulce 5.1). V těchto časech je zahrnuto vytvoření kopie objektu, provedení požadované operace i uložení do seznamu historie. Při nastaveném plném módu vykreslování modelu nevykazovala aplikace viditelné zpoždění při manipulaci se scénou prohlížeče. V módu drátovém nebo plném se zvýrazněnými hranami se již u takto detailního modelu projevuje postřehnutelné zpoždění[9] při interakci s prohlížečem. Toto zpoždění je způsobeno nižší podporou vykreslování čar oproti plným polygonům a nemá na funkčnost aplikace vliv.

[9]Zpoždění bylo patrné i v aplikaci Blender před exportem modelu.

Tabulka 5.1: Naměřené časy prováděných operací pro různý počet polygonů. Oprava byla prováděna s kompletním nastavením. Uvedené hodnoty jsou aritmetickým průměrem z pěti měření

	78000 facetů	1016000 facetů
Zrcadlení	0.0215 s	0.3198 s
Změna měřítka	0.0142 s	0.2965 s
Translace	0.0151 s	0.2527 s
Rotace	0.0430 s	0.6159 s
Oprava	0.0686 s	1.5165 s

Druhý soubor byl složen ze 100 vzájemně se neprotínajících krychlí. Model obsahující krychle byl rozdělen přímo v aplikaci funkcí split na 100 samostatných objektů. Cílem tohoto testu bylo zejména ověření správného fungování seznamu otevřených objektů v levém menu aplikace a dále fungování výběru objektů. Seznam automaticky doplňuje posuvník tak, že všechny objekty zůstávají přístupné, při výběru přímo ve scéně prohlížeče byl vždy správně vybrán požadovaný objekt.

5.4 Test nesprávných vstupů

Jelikož aplikace pracuje se soubory, bylo nezbytné ověřit správné chování v případech, kdy je uživatelem zadán nesprávný vstup. Taková situace může nastat, je-li aplikace spuštěna z příkazové řádky s parametry představujícími názvy vstupních souborů nebo v dialogovém okně akcí Otevřít, Uložit jako a Exportovat. V žádném z uvedených případů nebyl během testování zjištěn problém. Dialogové okno Otevřít nabízí pouze dostupné soubory STL. Je-li přesto zadán nesprávný název souboru, případně načten soubor s příponou .stl, jehož obsah neodpovídá specifikaci formátu, je uživatel informován prostřednictvím informačního panelu. Obdobně je řešeno načítání souborů dle řetězců dodaných na vstupu. Uložení nebo export souboru jsou úspěšně provedeny jak s názvem obsahující příponu, tak s názvem bez přípony, v tom případě je přípona automaticky doplněna.

5.5 Testy uživatelského rozhraní

Již v průběhu implementace byla aplikace při dosažení významných milníků předvedena vybraným uživatelům a dle připomínek dále upravována. V rámci předmětu Tvorba uživatelského rozhraní byla aplikace za asistence studentů podrobena kognitivnímu a heuristickému průchodu. Následuje seznam zjištěných problémů a jejich řešení.

- **Nejasné ovládání změny velikosti modelu (scale)** – bylo upraveno rozložení příslušných prvků.

- **Absence prvků podporujících orientaci v prostoru prohlížeče** – k hlavním osám souřadnic byly přidány pomocné osy do levého dolního rohu prohlížeče. Dále byla přidána možnost zobrazit pomocnou síť v rovině určené osami x a y. Rozestup mezi jednotlivými křížovými body sítě se mění v závislosti na míře přiblížení. Numerický údaj o velikosti rozestupu je přidán na konec informačního panelu.

- **Dialogová okna otevírající se pod aplikací** – problém byl odstraněn nastavením hlavního okna aplikace do konstruktoru dialogového okna.

- **Absence textové informace o provedené úpravě** – ke spodnímu okraji hlavního okna byla přidána lišta s textovou informací o provedené úpravě.

V následující fázi byla aplikace ADMeshGUI předložena k testování členům laboratoře 3D tisku Fakulty informačních technologií ČVUT v Praze. Členové laboratoře jsou předpokládanými typickými uživateli aplikace, dokáží se tedy zaměřit na problémy spojené s jejím reálným použitím. Následuje seznam získaných připomínek a jejich řešení.

- **Reakce na pohyb myši při rotaci a translaci je pevně nastavena** – do dialogového okna Vlastnosti byla přidána možnost invertovat směr otáčení scény při rotaci a translaci.

- **Nevhodné použití klávesy Delete pro mazání objektů** – na platformě Mac OS X je akce klávesy `Delete` (tak, jak je běžně používána např. na platformě Windows) simulována kombinací kláves `Fn + Backspace` (případně `Fn + Delete` dle pojmenování klávesy). Použití dvojice kláves pro smazání vybraných objektů je nepohodlné, aplikace ADMeshGUI byla proto upravena, aby i na platformě Mac OS X stačilo k vyvolání příslušné akce stisknutí jediné klávesy bez pomocné klávesy `Fn`.

- **Model v drátovém módu zobrazení splývá s pomocnou sítí** – tento problém byl způsobem použitím shodné černé barvy jak pro síť, tak pro model samotný. Byla proto upravena barva pomocné sítě.

- **Model není možné snadno duplikovat a centrovat okolo počátku soustavy souřadnic** – byla přidána patřičná funkcionalita a tlačítka umožňující duplikovat a centrovat aktivní modely.

5.5.1 Test použitelnosti

Nad upravenou aplikací byl proveden test použitelnosti. Testu se účastnili dobrovolníci z řad studentů předmětu 3D tisk vyučovaného na FIT ČVUT. Autor práce při testu působil jako moderátor zadávající jednotlivé úkoly. Úkoly spočívaly v plnění předem připravených scénářů, které pokrývají převážnou většinu schopností ADMeshGUI. Následuje seznam scénářů.

1. V domovském adresáři se nachází soubor *test1.stl*. Model reprezentovaný tímto souborem obsahuje chyby. Otevřete soubor a proveďte patřičné opravy.

2. Opravený soubor by měl být dále upraven. Zmenšete model ve směru osy z na polovinu velikosti a posuňte model do kladných hodnot ve směru osy z. Takto upravený model uložte do domovského adresáře jako *result1.stl* v binárním STL formátu.

3. V domovském adresáři se nachází soubor *test2.stl*. Otevřete soubor. Model reprezentovaný tímto souborem tvoří pouze polovinu požadovaného objektu. Model duplikujte, zrcadlete a slučte tak, aby výsledek tvořil navazující celek. Výsledek exportujte ve formátu OBJ jako *result2.obj*.

4. V domovském adresáři se nachází soubor *test3.stl* obsahující více samostatných částí. Otevřete soubor a rozdělte jej na samostatné části. Vyberte tři libovolné části, slučte je a ostatní části smažte.

5. Změňte barvu, kterou jsou vykreslovány modely.

Z testu vyplynulo několik základních poznatků. Především uživatelé používají různé způsoby otevření souboru (přes menu, tlačítko i přetažením do okna aplikace), je tedy v pořádku, že ADMeshGUI všechny způsoby umožňuje. Nalezení tlačítka `REPAIR` pro opravy je téměř okamžité. V kombinaci s výchozím nastavením umožňujícím provést všechny opravy najednou se jedná o uživatelsky efektivní způsob opravy modelu. K výběru objektů většina uživatelů používá primárně seznam v levém menu, někteří automaticky využili klávesových zkratek `Ctrl + A` pro výběr všech objetů a `Ctrl + I` pro inverzní výběr.

Největší potíže uživatelům činil scénář č. 2, konkrétně změna velikosti pouze ve směru jedné osy. Všichni se pokusili nejprve změnit přímo danou hodnotu příslušející ose z. Ve výchozím nastavení se však mění všechny hodnoty (pro osy x, y, z) zároveň, aby byla usnadněna jednotná změna velikosti ve všech směrech. Po nalezení zaškrtávacího pole `Fixed ratio`, které „zamyká“ nebo „odemyká“ možnost měnit hodnoty jednotlivě, uživatelům dokončení úpravy problémy nečinilo. Nutno podotknout, že hledání testujícím někdy trvalo i přes 10 s. Vzhledem k tomuto faktu bylo ovládání změny měřítka opět upraveno, ve výchozím nastavení je možné měnit hodnoty jednotlivě; ke změně

všech hodnot najednou je nutné tuto možnost povolit zaškrtnutím pole `Fixed ratio`.

Přestože většina uživatelů neměla se scénářem č. 3 žádné potíže (okamžitě nalezli a použili tlačítko `Duplicate`), jedenkrát se objevila připomínka, že by duplikaci mělo být možné provést přes menu nebo kombinací kláves `Ctrl + C, CTRL + V`. Hlavní aplikační menu neobsahuje manipulační položky, případná položka duplikace by mezi ostatními položkami nebyla dostatečně výrazná. Implementace duplikace pomocí kláves `Ctrl + C, CTRL + V` by přinesla další problémy, např. v podobě problematického rozhodnutí, jak pracovat s modelem ve schránce, pokud by byl před vložením upraven. Funkcionalita duplikace tedy nebyla nijak upravena.

Závěr

Cílem práce bylo vytvoření nové plně funkční open-source aplikace nazvané ADMeshGUI, která poskytuje grafické uživatelské rozhraní existující aplikaci ADMesh. Jelikož ADMesh slouží k manipulaci s 3D modely v souborovém formátu STL, stal se podstatnou součástí aplikace také 3D prohlížeč. ADMeshGUI, v porovnání s řádkově orientovanou aplikací ADMesh, nabízí jednodušší ovládání a poskytuje uživateli přímou vizuální kontrolu nad prováděnými operacemi. Oproti podobně orientované konkurenci pak vyniká především možností rychle provést automatickou opravu načtených modelů.

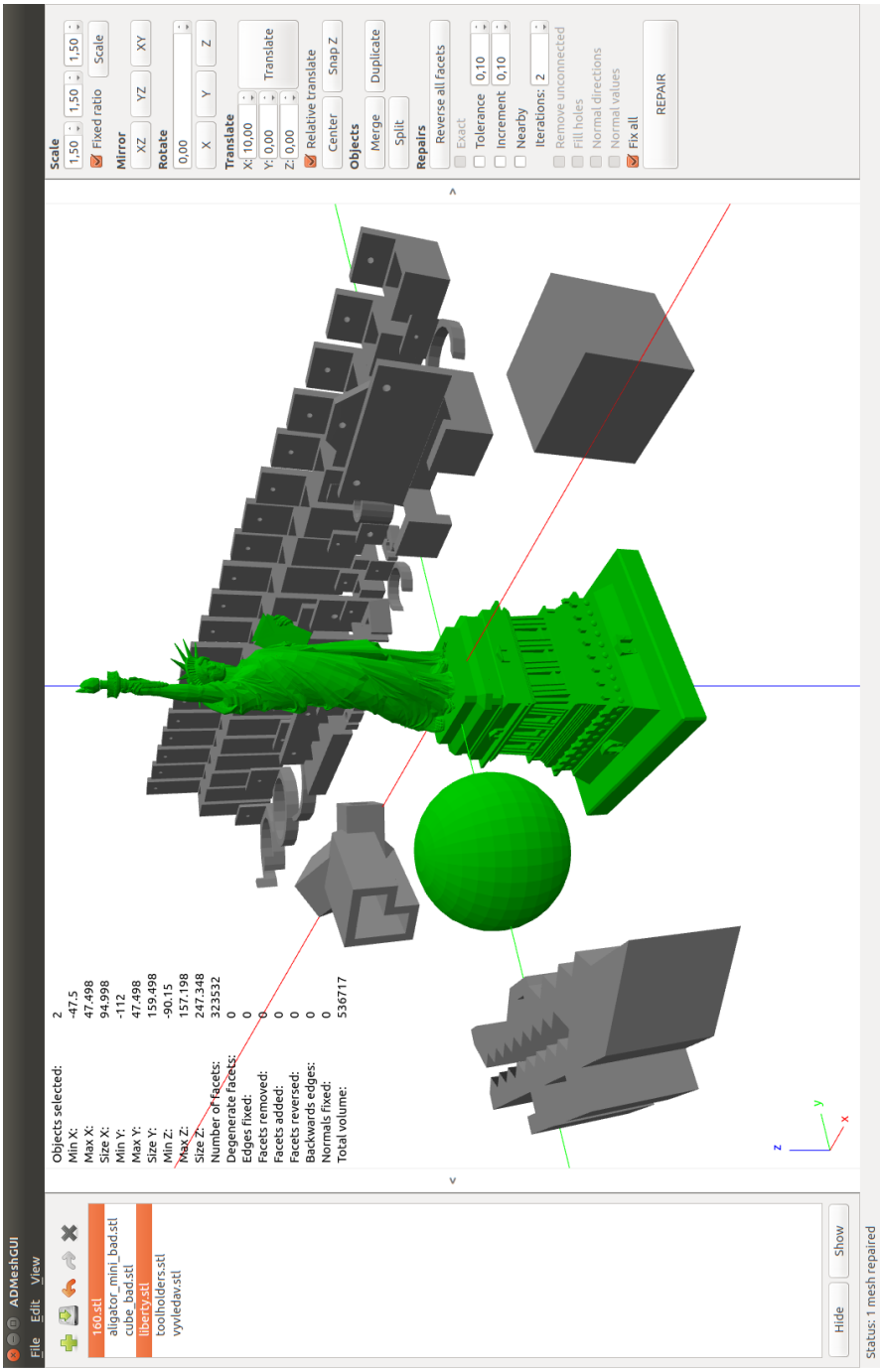
Návrh rozhraní byl proveden na základě rozboru souborového formátu STL, dodaného knihovního rozhraní aplikace ADMesh a podobných existujících řešení. Výsledná aplikace byla implementována v jazyce C++ za podpory frameworku Qt pro tvorbu rozhraní a knihovny OpenGL pro tvorbu 3D prohlížeče. Vykreslování 3D obsahu je realizováno pomocí shader programů napsaných v jazyce GLSL. Zdrojové kódy aplikace byly vydány pod licencí GNU Affero General Public License v. 3 a jsou volně dostupné na serveru www.github.com[10]. Představu o výsledném vzhledu aplikace ADMeshGUI je možné si vytvořit na základě obrázku 5.1, který se nachází na straně 41.

ADMeshGUI splňuje všechny funkční i nefunkční požadavky, které byly na tuto aplikaci kladeny. Umožňuje prohlížet načtené modely v okně prohlížeče, manipulovat s nimi nebo je opravovat. Aplikace zvýrazňuje chyby modelů (nesprávně orientované facety). Upravené modely mohou být uloženy v binární i ASCII verzi formátu STL nebo exportovány do dalších souborových formátů. Základ pro lokalizaci do dalších jazyků je zajištěn pomocí balíčku `gettext`. Nad rámec funkčních požadavků umožňuje ADMeshGUI rozdělit model na více jednotlivých modelů na základě začlenění nástroje `stlsplit`.

Aplikace ADMeshGUI nalezne využití zejména v oblasti 3D tisku, předpokládá se její začlenění do procesu přípravy tisknutých modelů v laboratoři 3D tisku[15] Fakulty informačních technologií ČVUT v Praze. Kromě svého

[10] https://github.com/vyvledav/ADMeshGUI

primárního účelu poslouží ADMeshGUI také vývojářům ADMeshe, kterým poskytne užitečnou zpětnou vazbu, neboť okamžitě uvidí výsledky jednotlivých funkcí. Díky pokračujícímu vývoji ADMeshe je zároveň možné aplikaci ADMeshGUI nadále rozšiřovat o nové funkce. V plánu je např. začlenění funkcionality `stlcut`[24] sloužící k rozdělení modelu na více částí dle zadané roviny.



Obrázek 5.1: Snímek obrazovky zachycující výslednou aplikaci ADMeshGUI v prostředí operačního systému Ubuntu 14.04 LTS. V dané ukázce je nastaven plný mód zobrazení, je možné shlédnout ovládací prvky levého i pravého menu a rozdíl v zobrazení aktivních a neaktivních objektů

Literatura

[1] Ježek, F. *Geometrické a počítačové modelování*. Západočeská univerzita v Plzni, Fakulta aplikovaných věd, Únor 2006, [cit. 12. 4. 2015]. Pomocný učební text. Dostupné z WWW: http://www.fd.cvut.cz/personal/voracsar/GM/PGR021/GM_Jezek.pdf

[2] Shene, C. K. *Constructive Solid Geometry*. [online][cit. 7. 4. 2015]. Michigan Technological University, Department of Computer Science. Dostupné z WWW: http://www.cs.mtu.edu/~shene/COURSES/cs3621/NOTES/model/csg.html

[3] Přispěvatelé Wikipedie. *Csg tree.png*. [online][cit. 19. 4. 2015]. Dostupné z WWW: http://commons.wikimedia.org/wiki/File:Csg_tree.png

[4] Shene, C. K. *Wireframe Models*. [online][cit. 20. 4. 2015]. Michigan Technological University, Department of Computer Science. Dostupné z WWW: http://www.cs.mtu.edu/~shene/COURSES/cs3621/NOTES/model/wireframe.html

[5] Shene, C. K. *Boundary Representations*. [online][cit. 4. 5. 2015]. Michigan Technological University, Department of Computer Science. Dostupné z WWW: http://www.cs.mtu.edu/~shene/COURSES/cs3621/NOTES/model/b-rep.html

[6] Otakar, J. *Možnosti detekce a opravy chyb 3D modelů v hraniční trojúhelníkové reprezentaci volně dostupným softwarem a webovými službami*. ČVUT v Praze, Fakulta elektrotechnická, Katedra počítačové grafiky a interakce, srpen 2013.

[7] netfabb GmbH. *netfabb Basic 5.2 User Manual*. červenec 2014, [online][cit. 28. 4. 2015]. Dostupné z WWW: http://www.netfabb.com/documentation.php

[8] Wah, W. H. *Introduction to STL format*. červen 1999, [online][cit. 14. 4. 2015]. Dostupné z WWW: http://download.novedge.com/Brands/FPS/Documents/Introduction_To_STL_File_Format.pdf

[9] Hashmi, S. *Comprehensive Materials Processing*. Newnes, 2014, ISBN 0080965334, 9780080965338, 5634 pp.

[10] Rypl, D.; Bittnar, Z. *Triangulation of 3D Surfaces Described by Stereolithography Files*. Stirling: Civil-Comp Press, 2004, ISBN 0-948749-97-0.

[11] Martin, A. D. *Analysis and repair of STL files*. Master's thesis, California State University, Department of Mechanical Engineering, California, Long Beach, 1998.

[12] Feng, W. *The STL Library*. [online][cit. 7. 4. 2015]. National University of Singapore, 10 Kent Ridge Crescent, Singapore 119260. Dostupné z WWW: http://www.eng.nus.edu.sg/LCEL/RP/u21/wwwroot/stl_library.htm

[13] 3DAddFab. *What is an STL file and is it obsolete?* Listopad 2011, [online][cit. 20. 3. 2015]. Dostupné z WWW: http://3daddfab.com/blog/index.php?/archives/4-What-is-an-STL-file-and-is-it-obsolete.html

[14] Blender Foundation. *Features - blender.org - Home of the Blender project - Free and Open 3D Creation Software*. [online][cit. 26. 4. 2015]. Dostupné z WWW: http://www.blender.org/features/

[15] 3DprintFIT. *3D tisk na FIT ČVUT*. [online][cit. 10. 2. 2015]. Dostupné z WWW: http://3dprint.fit.cvut.cz/

[16] Peels, J. *Netfabb is an awesome STL viewer & repair tool*. shapeways, Říjen 2009, [online][cit. 13. 4. 2015]. Dostupné z WWW: http://www.shapeways.com/blog/archives/312-Netfabb-is-an-awesome-STL-viewer-repair-tool.html

[17] Materialise. *Magics 19*. [software][přístup 13. 4. 2015]. Dostupné z WWW: http://software.materialise.com/magics

[18] DEVELOP3D. *Review: Materialise Magics 18*. DEVELOP 3D, Únor 2014, [online][cit. 13. 4. 2015]. Dostupné z WWW: http://www.develop3d.com/reviews/review-materialise-magics-18

[19] Materialise. *MiniMagics 3.0*. [software][přístup 13. 4. 2015]. Dostupné z WWW: http://software.materialise.com/minimagics

[20] Attene, M.; Campen, M.; Kobbelt, L. *Mesh Repairing Software on the Web*. [online][cit. 13. 4. 2015]. Dostupné z WWW: http://www.meshrepair.org/

[21] Martin, A. D.; several contributors. *ADMESH 0.98.1*. [software][přístup 29. 3. 2015]. Dostupné z WWW: `https://github.com/admesh/admesh/releases`

[22] Hrončok, M. *snapz*. [software][přístup 18. 4. 2015]. Dostupné z WWW: `https://github.com/hroncok/snapz`

[23] Ranellucci, A.; Hrončok, M. *stlsplit*. [software][přístup 15. 4. 2015]. Dostupné z WWW: `https://github.com/hroncok/stlsplit`

[24] Hrončok, M. *stlcut*. [software][přístup 31. 3. 2015]. Dostupné z WWW: `https://github.com/hroncok/stlcut`

[25] Nielsen, J. *Usability 101*: *Introduction to Usability*. Nielsen Norman Group, leden 2012, [online][cit. 1. 5. 2015]. Dostupné z WWW: `http://www.nngroup.com/articles/usability-101-introduction-to-usability/`

[26] Schmidt, J. *BI-TUR 1. Úvod*. 2011, [online prezentace][cit. 1. 5. 2015]. Dostupné z WWW: `https://edux.fit.cvut.cz/oppa/BI-TUR/prednasky/TUR1uvod.pdf`

[27] Qt project. *Qt Documentation*. [online][cit. 29. 3. 2015]. Dostupné z WWW: `http://doc.qt.io/`

[28] Přispěvatelé OpenGL Wiki. *Default Framebuffer*. OpenGL Wiki, Poslední modifikace únor 2006, [online][cit. 12. 4. 2015]. Dostupné z WWW: `https://www.opengl.org/wiki/Default_Framebuffer`

[29] Borgwardt, M. *Error Propagation*. [online][cit. 10. 4. 2015]. Dostupné z WWW: `http://floating-point-gui.de/errors/propagation/`

[30] Heikkinen, J. *Qt5.4 release plan*. [plán vydání][cit. 23. 11. 2014]. Dostupné z WWW: `http://qt-project.org/wiki/Qt-5.4-release`

Příloha **A**

Seznam použitých zkratek

GUI Graphical user interface

3D Three-dimensional (trojrozměrný)

STL STereoLithography

CAD Computer Aided Design

GLSL OpenGL Shading Language

FBO Frame Buffer Object

VBO Vertex Buffer Object

MVP matice Model - View - Projection matice

VS Vertex shader

FS Fragment shader

CPU Central Processing Unit

ČVUT České vysoké učení technické

GPL General Public License

Příloha **B**

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `exe` ..... adresář se spustitelnou formou implementace
- `src`
  - `admeshgui` ..... zdrojové kódy implementace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `BP_Vyvlecka_David_2015.pdf` ..... text práce ve formátu PDF