VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

3D SPRÁVCE SOUBORŮ

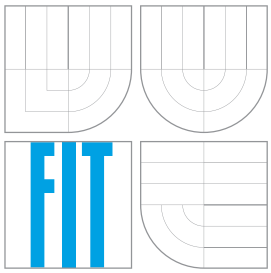
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

ONDŘEJ MARTINÁK

BRNO 2007

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

3D SPRÁVCE SOUBORŮ

3D FILE MANAGER

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

ONDŘEJ MARTINÁK

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. VÍTĚZSLAV BERAN

BRNO 2007

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačové grafiky a multimédií Akademický rok 2006/2007

Zadání bakalářské práce

Řešitel: **Martinák Ondřej**
Obor: Informační technologie
Téma: **Správce souborů ve 3D**
Kategorie: Uživatelská rozhraní

Pokyny:

1. Prostudujte, analyzujte a zdokumentujte nástroje na trovbu uživatelkých rozhraní (GUI) ve 3D. Vyberte jeden nejvhodnější pro řešení zadání.
2. Seznamte se vybavením pro virtuální realitu (virtuální brýle), prostudujte jeho rozhraní a zdokumentujte.
3. Navrhněte objektovou nadstavbu vybraného nástroje umožňující práci s virtuálními brýlemi.
4. Navržené rozšíření implementujte.
5. Implementujte demonstrační aplikaci typu správce souborů ve 3D.
6. Diskutujte dosažené výsledky a možné pokračování práce.

Literatura:

- Žára, J., Beneš, B., Felker, P.: Moderní počítačová grafika, Computer Press, 1998.
- Hlaváč, V.: Zpracování signálů a obrazů, Praha, ČVUT, 2005.
- Rafael C. Gonzalez, Richard E. Woods: Digital image processing, Prentice-Hall, 2002.
- Galbiati, L. J.: Machine vision and digital image processing fundamental, Prentice-Hall, 1990.
- dále dle pokynu vedoucího

Při obhajobě semestrální části projektu je požadováno:

- Body 1., 2. a 3.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním paměťovém médiu (disketa, CD-ROM), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Beran Vítězslav, Ing.**, UPGM FIT VUT
Datum zadání: 1. listopadu 2006
Datum odevzdání: 15. května 2007

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačové grafiky a multimédií
612 66 Brno, Božetěchova 2

L.S.

doc. Dr. Ing. Pavel Zemčík
vedoucí ústavu

Licenční smlouva

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

Abstrakt

Tato práce je zaměřena na návrh 3D uživatelského rozhraní pro práci se souborovým systémem. Obsahuje stručný úvod k 3D uživatelským rozhraním a techniky používané pro jejich návrh a implementaci. Dále uvádí stručný přehled knihoven využitelných pro vývoj aplikací virtuální reality a popis zařízení, které bylo použito pro grafický výstup a získávání informací o pohybech uživatelovy hlavy.

Klíčová slova

3D uživatelské rozhraní, virtuální realita, virtuální prostředí, head mounted display, vizualizace souborového sytému

Abstract

This thesis concentrates on a design of 3D user interface for interaction with a file system. It presents a short description of 3D user interfaces and techniques used for their design and implementation. It also presents a brief description of libraries which might be used for development of virtual reality applications and a description od device which was used for graphical output and an acquisition of informations about user’s head movements.

Keywords

3D user interfaces, virtual reality, virtual environment, head moutned display, file system visualization

Citace

Ondřej Martinák: 3D Správce souborů, bakalářská práce, Brno, FIT VUT v Brně, 2007

3D Správce souborů

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Vítězslava Berana.

. . . . . .
Ondřej Martinák
31. července 2007

Poděkování

Rád bych poděkoval svému vedoucímu, Ing. Vítězslavu Beranovi, za to, že se mi snažil vyjít maximálně vstříc, a za jeho rady a vedení, díky kterým jsem se mnohému naučil.

Obsah

Kapitola 1

Úvod

1.1 Seznámení s tématem práce

Tato bakalářská práce se zabývá oblastí 3D uživatelských rozhraní, konkrétně vizualizací hierarchické struktury jakou je souborový systém v prostoru. Samotná problematika vizualizace hierarchických struktur je komplexní a náročná a cílem této práce není její analýza ani snaha o implementaci optimální vizualizační techniky. Cílem této práce je návrh a implementace rozhraní mezi zobrazenými daty a uživatelem tak, aby bylo intuitivní pro práci, aby nevyžadovalo zdlouhavý osvojovací proces, aby uživateli poskytovalo zobrazená data přehledné a účelně a aby s těmito daty mohl uživatel snadno a efektivně pracovat.

Tato práce má sloužit jako úvod do problematiky návrhu rozhraní mezi uživatelem a počítačem, zejména pak 3D grafických rozhraní. Proto nezabíhá do některých specifických detailů, ale raději se snaží ukázat základní techniky a metody návrhu a implementace 3D UI na demonstrační aplikaci 3D správce souborů.

1.2 Struktura dokumentace

Tato dokumentace je rozčleněna do kapitol, které pojednávají o jednotlivých aspektech bakalářské práce.

- **Kapitola 1** je kapitola úvodní a má za úkol seznámit čtenáře s tématem a cílem bakalářské práce. Součástí této kapitoly je také úvod do historie uživatelských rozhraní a možnosti jejich použití.
- **Kapitola 2** obsahuje výčet knihoven a toolkitů, které mohou být použity pro vývoj aplikací virtuální reality. Snahou nebylo poskytnout vyčerpávající informace o technických detailech jednotlivých knihoven, ale spíše shrnout jejich charakteristiky a poukázat na jejich přednosti, případně nedostatky. Závěrem kapitoly uvádím toolkity zvolené pro vývoj demonstrační aplikace 3D správce souborů a důvody, které mě k tomuto rozhodnutí vedly.
- **Kapitola 3** popisuje zařízení použité pro vizuální výstup. V první části kapitoly je toto zařízení popsáno obecně, zatímco v druhé částí jsem se zaměřil konkrétně na použitý model eMagin Z800 3DVisor.
- **Kapitola 4** je zaměřena na problematiku vizualizace souborového systému. V první části popisuje souborový systém jako hierarchickou strukturu a metody vizualizace této struktury. Ve druhé pak popisuje přístup zvolený v rámci této práce.

- **Kapitola 5** se zabývá návrhem a implementací demonstrační aplikace 3D správce souborů. Tato kapitola využívá poznatků získaných v předchozích kapitolách a slučuje dohromady jednotlivé části, které tvoří výslednou aplikaci.

- **Kapitola 6** tvoří závěr práce a uvádí tak její výsledky, výhody a nevýhody použitých technik a zařízení a také možnosti rozšíření práce. Dále shrnuje které části práce prezentuje demonstrační aplikace a které ještě nejsou implementovány.

1.3 Úvod do 3D uživatelských rozhraní

Na tomto místě by možná bylo vhodné uvést přehled historie uživatelských rozhraní, ale myslím, že to už zdárně zvládli jiní. Příkladem budiž obsáhlý článek na Ars Technice, jehož autorem je Jeremy Reimer (http://arstechnica.com/articles/paedia/gui.ars/1). Já bych chtěl raději vyzvednout jeden zajímavý jev, který, jak se zdá, se v historii opakuje a který je zřetelný i v jiných odvětvích.

Vývojáři se často nechávají unést novými technologiemi. Dobře patrné je to např. v průmyslu počítačových her, kde je pokrok velmi rychlý. S příchodem nové platformy nebo technologie se vývojáři snaží ukázat její potenciál a rozdíl oproti starým technikám a často tak vznikají produkty, kde jsou nové techniky použity jenom proto, aby byly použity, nebo jsou použity zcela neúměrně. Příkladem budiž např. využití fyziky v počítačových hrách. S příchodem procesorů dostatečně výkonných pro výpočet fyzikálních simulací v reálném čase se tyto simulace začaly používat i ve hrách, aby hráčům poskytly ještě reálnější a intenzivnější zážitek. Paradoxně však byly tyto simulace často velmi zveličené a fyzikálním zákonům se spíš vzdalovaly než přibližovaly.

Ve světě 3D uživatelských rozhraní tomu bylo nejinak. S příchodem grafického hardwaru, který umožňoval vykreslovat složité prostorové scény, se objevila možnost přenést do prostoru i uživatelská rozhraní. Tak začaly vznikat čistě prostorové UI, které byly možná efektní, ale už méně efektivní. Postupem času, když si vývojáři na novou platformu zvykli a poznali její výhody a omezení, začaly vznikat rozhraní kombinující 2D a 3D techniky. A právě tady je podstata návrhu 3D UI, využít prostor jako podporu a vhodně jej zkombinovat s 2D prvky a metodami.

3D uživatelská rozhraní jsou tedy taková rozhraní, která využívají pro interakci 3D prostředí. Tato prostředí mohou být čistě virtuální, tedy počítačové simulace nejrůznějšího charakteru, nebo se může jednat o rozšířenou realitu, kde počítačem generovaná grafika obohacuje reálné prostředí.

Aplikace využívající 3D UI mohou využívat například inženýři a architekti pro převod schémat do 3D prostředí. Prostorové zobrazení může lépe prezentovat danou práci a umožňovat snadnější interakci se zákazníkem, který nemusí technickým schématům vždy rozumět. Aplikace virtuální reality mohou být také využity jako podpora psychiatrické léčby pacientů trpících strachem z určitých objektů nebo situací. Ve virtuálním prostředí je možné tyto objekty nebo situace simulovat a pacient si tak na ně může postupně zvykat a interagovat s nimi. Vědci a inženýři mohou 3D rozhraní používat pro vizualizaci dat získaných z různých experimentů. Dalším potenciálem virtuálních prostředí jsou kolaborativní aplikace, které mají za úkol usnadnit spolupráci lidí, kteří se nemohou fyzicky setkat. Mohou se ale spojit elektronicky a potřebují aplikace, které jim umožní sdílet nejrůznější data a navzájem komunikovat.

Kapitola 2

Toolkity pro 3D uživatelská rozhraní

2.1 Komplexní toolkity

Tyto toolkity se vyznačují tím, že jsou komplexní, tedy často zahrnují všechny aspekty aplikace virtuální reality jako správu zařízení, rendering, 3D zvuk, správu 3D scény, apod. Z důvodu snadnější manipulace bývají tyto toolkity hodně modulární, takže vývojáři mohou využít ty části, které právě potřebují.

2.1.1 Colosseum3D

Colosseum3D je framework pro vývoj interaktivních 3D simulací. Je postaven nad několika open-source knihovnami pro rendering, 3D zvuk, skriptování, animaci postav a jednou komerční knihovnou pro dynamiku rigidních těles.

Má také podporu pro interaktivní zařízení jako Ascention Motionstar, Polhemus Fastrak, Polhemus Liberty a Fakespace PinchGlove. Díky Sensable OpenHaptics API také podporuje Phantom force feedback zařízení.

Hlavní myšlenkou Colosseum3D je poskytnout funkcionalitu v oddělených modulech. Proto mohou být některé balíčky staženy osamoceně. Příkladem jsou osgAL, OpenAL++, osgHaptics a ReplicantBody.

2.1.2 DIVERSE

DIVERSE je multiplatformní, open-source API pro vývoj aplikací virtuální reality. V současné době DIVERSE běží na Linuxu a IRIXu. Podpora pro Windows XP a Mac OS X je ve vývoji.

Účelem DIVERSE je umožnit vývojářům rychle vyvíjet aplikace, které poběží jak na desktopových tak na různých “immersive” systémech. Aby se zabránilo překážkám při vývoji, tak byl DIVERSE navržen decentralizovaným způsobem, takže vývojáři nohou použít jen ty části, které zrovna potřebují. To umožňuje DIVERSE interagovat s mnoha dalšími toolktiy a API jako OpenGL, Open Scene Graph, SGI Open GL Performer a Coin.

DIVERSE je distribuován jako free software (LGPL a GPL licence).

2.1.3 VR Juggler

VR Juggler poskytuje platformu pro vývoj aplikací virtuální reality a umožňuje uživatelům spouštět aplikace na téměř jakémkoli systému virtuální reality. VR Juggler pracuje jako rozhraní mezi ostatními Juggler komponentami. VR Juggler je rozšiřitelný od jednoduchých desktopových systému

jako PC po komplexní multi-obrazovkové systémy běžící na high-end pracovních stanicích a superpočítačích.

Dalšími částmi Juggler systému jsou:

- Gadgeteer systém pro správu zařízení
- JCCL konfigurační systém založený na XML
- VPR VR Juggler portable runtime, poskytuje na platformě nezávislé abstrakce pro vlákna, sokety a sériový I/O
- Sonix vysokoúrovňová abstrakce pro zvukový hardware nebo zvuková API
- PyJuggler podpora jazyka Python
- VRJ.NET kolekce C/C++ knihoven a .NET asembly, které poskytují provázání VR Juglleru a Common Language Infastructure (CLI)

VR Juggler je distribuován pod LGPL licencí.

2.1.4 Coin3D

Coin3D je 3D grafický toolkit pro vývoj multiplatformních real-time 3D aplikací. Je portovatelný na velké množství systémů: jakákoli UNIX / LINUX / *BSD platforma, všechny verze MS Windows a Mac OS X.

Coin3D je high-level API postavené nad OpenGL a zapouzdřuje tak veškeré volání rutin této grafické knihovny. I tak je ale možné mixovat kód těchto dvou knihoven, například při postupném přechodu z OpenGL na Coin3D. Toolkit také obsahuje knihovny pro import/export souborů pro obrázky, zvuky a videa a pro manipulaci s DXF a MultiGen Open Flight 3D soubory. Dále nabízí knihovny pro provázání s GUI MS Windows, Trolltech's QT, Xt/Motif na X Windows a Mac OS X.

Coin3D je plně kompatibilní s SGI Open Inventor 2,1 a zahrnuje podporu VRML97, 3D zvuků, 3D textur a paralelního renderingu na více procesorech.

Coin3D je distribuován ve dvou licencích:

- Coin Professional Edition - licencován pod Coin Professional Edition License (Coin PEL), umožňuje vytvářet proprietární software a nabízí technickou podporu.
- Coin Free Edition - licencován pod GNU General Public License (GPL), určen pro vývoj Free Softwaru.

2.1.5 Virtual Environments Library

Virtual Environments Library (veLib) je malý multiplatformní toolkit pro distribuované simulace virtuální reality v reálném čase vyvíjený v Institutu Maxe Plancka pro Biologická Kybernetika. Byl navržen tak, aby poskytoval pohodlné a jednotné rozhraní pro různé druhy vstupních a výstupních zařízení a aby zakryl odlišnosti pro různé systémové architektury pod jednoduchou abstrakční vrstvou.

Centrálním bodem veLib je rozhraní pro abstraktní zařízení, od kterého jsou odvozeny rozhraní pro zařízení specifická. Skrze tyto rozhraní je možné přistupovat k zařízení pro renderování grafiky (s použitím knihovny OpenGL), zařízení pro správu okna, zpracování sytému událostí a zpracování vstupu od uživatele a také zařízením pro práci se zvukem nebo sítovou komunikací. Tento návrh také umožňuje velmi snadno vytvářet vlastní zařízení a tyto začlenit do systému veLib. Příkladem mohou

být dostupné rozšíření v podobě dalších zařízení pro prostorový zvuk nebo zařízení využívající pro práci se zvukem knihovnu SDL. Další možností jak funkcionalitu veLib rozšířit je použití pluginů.

veLib je komplexním toolkitem a snaží se tedy nabízet vše, co mohou vývojáři potřebovat pro vývoj aplikací virtuální reality, ale dělá to tak, aby si zachoval svou jednoduchost. Nabízí tak (mimo už zmíněných) rozhraní pro práci s prostorovou scénou a objekty v ní, práci s 2D grafikou a základními prvky, které mohou být využity pro výstavbu 2D GUI, rozhraní pro přístup ke vstupním zařízením (mimo standardní např. joystick), základní pohybový model a detekci kolize s terénem a různé matematické rutiny.

veLib také umožňuje obejít klasický systém vykreslování 3D grafiky a použít tak funkce knihovny OpenGL přímo, což může být v některých situacích výhodné.

veLib je distribuován pod LGPL licencí.

2.2 Úzce zaměřené toolkity

Tyto toolkity jsou specifické svým zaměřením na jednu konkrétní oblast na rozdíl od výše zmíněných komplexních toolkitů. I tak to jsou často velké knihovny, zvláště v případě toolkitů pro 3D grafiku, a tak bývají také členěny na moduly. Vhodnou kombinací těchto toolkitů můžeme dosáhnout stejné funkcionality jakou mají toolkity komplexní a při tom máme větší flexibilitu při pokrytí jednotlivých částí aplikace těmito knihovnami.

2.2.1 OpenSceneGraph

OpenSceneGraph je open source multiplatformní grafický toolkit pro vývoj vysoce výkonných grafických aplikací jako například letecké simulátory, hry, aplikace virtuální reality a vědecké vizualizace. Postaven na konceptu *SceneGraph*u poskytuje objektově orientovaný framework nad OpenGL a umožňuje tak vývojářům oprostit se od implementace a optimalizace nízkoúrovňových grafických volání. Poskytuje také mnoho dalších utilit pro zrychlení vývoje aplikací.

OpenSceneGraph klade velký důraz na výkon. Technologie *SceneGraph*u a na něj aplikovaných metod ořezávání scény, správy úrovně detailů, řazení, vertexová pole a display seznamy z něj činí jedno z nejvýkonnějších dostupných řešení.

OpenSceneGraph je rozdělen na jádro a přídavné moduly například pro práci s částicovými systémy, s vysoce kvalitním textem, speciálními efekty nebo OpenGL shader jazykem. Tyto moduly mohou být také vyvíjeny třetí osobou (komunitou), příkladem budiž modul pro podporu NVidia vertex, fragment extenzí a Cg shader jazyka nebo modul pro integraci OSG a Cal3D.

OpenSceneGraph využívá C++ a OpenGL a je proto snadno portovatelný na velké množství systémů: IRIX, Linux, Windows, FreeBSD, Mac OS X, Solaris, HP-UX a dokonce PlayStation2.

OpenSceneGraph je distribuován pod OpenSceneGraph Public Licencí (OSGPL), která je založena na Lesser Gnu Public Licenci (LGPL) a zahrnuje wxWidgets dodatky k LGPL.

2.2.2 Horde3D

Horde3D je snahou o vytvoření malého a lehkého 3D grafického toolkitu, který si klade za cíl poskytnout omračující vizuální efekty, což mimo jiné znamená potřebu OpenGL 2.0 kompatibilní grafické karty. Horde3D má jednoduché a intuitivní rozhraní přístupné z prakticky kteréhokoliv programovacího jazyka a je obzvláště vhodný pro renderování velkého počtu animovaných postav.

Tento toolkit je ještě v rané fázi vývoje (aktuálně verze 0.6.1), nicméně vypadá velmi slibně. Podporovaná platforma je MS Windows.

Horde3D je distribuován pod LGPL licencí.

2.3 2D GUI toolkity

I přes to, že je aplikace koncipována jako 3D virtuální prostředí, uživatel často potřebuje pracovat s 2D ovládacími prvky jako jsou rozbalovací nabídky, dialogová okna, tabulková data apod. Většinou nemá smysl tyto prvky transformovat do třetího rozměru, protože by to přineslo více zmatku než užitku.

Protože 2D GUI jsou ve vývoji již několik desetiletí a v dnešní době jsou v nějaké formě přítomny prakticky na každém počítači, je také k dispozici dostatek kvalitních knihoven pro práci s nimi. Pro projekt 3D Správce souborů jsem zvažoval následující tři. Všechny jsou multiplatformní a jsou šířeny jako open source software.

2.3.1 wxWidgets

Knihovna wxWidgets byla započata Julianem Smartem na University of Edinburgh. Jejím původním cílem bylo poskytnout API pro vývoj aplikací na systémech Unix a Windows. V průběhu času postupně přibývali další vývojáři a knihovna se začala rozrůstat, takže dnes podporuje mnoho dalších platforem jako Mac, WinCE a další.

wxWidgets by se daly popsat jako knihovna nativní, protože na každé platformě používá její konkrétní nástroje pro tvorbu GUI, místo aby se je snažila napodobit pomocí grafických primitiv jako to dělají některé jiné toolkity. Proto výsledné aplikace vypadají pro každou platformu přirozeně. wxWidgets se také snaží být komplexní a poskytovat tak všechny potřebné nástroje pro vývoj aplikací, aby nebylo nutné skládat více toolkitů dohromady, a proto vedle klasických GUI komponent nabízí rozhraní pro práci s databází, TCP/IP sítěmi, vícevláknové programování, integraci OpenGL a další.

wxWidgets jsou šířeny pod wxWindows licencí, která je založena na licenci LGPL, a navíc říká, že výsledné projekty mohou být distribuovány v binární formě za podmínek, které si stanoví uživatel.

2.3.2 Fast Light Toolkit

Fast Light Toolkit (FLTK) je malá knihovna pro tvorbu GUI aplikací na systémech Windows, Unix/Linux a MacOS X. FLTK abstrahuje nízkoúrovňové funkce jednotlivých systémů a vytváří tak vlastní rozhraní pro vykreslování a práci s GUI a práci se systémem událostí, Mimo to FLTK také podporuje 3D grafiku skrze OpenGL a vestavěnou emulaci GLUT.

FLTK také nabízí program pro návrh GUI, zvaný FLUID. Tento program dokáže výsledný návrh zkonvertovat do kódu jazyka C, který lze připojit k aplikaci, která má navržené GUI používat.

FLTK je distribuován pod LGPL2 licencí s výjimkami, které umožňují statické linkování k aplikacím, které nemají kompatibilní licenci.

2.3.3 Crazy Eddie's GUI

Crazy Eddie's GUI (CEGUI) je, narozdíl od předchozích dvou toolkitů, knihovna pro návrh a práci s GUI zaměřená na vývojáře her. Velkým rozdílem tedy je to, že se nesnaží o nativní podobu GUI na různých systémech a ani nevolá klasické funkce pro výstavbu GUI. Místo toho jednotlivé prvky vykresluje pomocí knihoven Direct3D nebo OpenGL nebo herních enginů Ogre3D nebo Irrlicht. Prvky nejsou sestavovány z grafických primitiv, ale jsou uloženy jako obrázky v souborech. Každý prvek tak může mít jinou grafickou reprezentaci v závislosti na použitém obrázku. Takto je možné vytvářet *skiny*, neboli sady obrázků, které sdílejí podobné charakteristiky pro všechny prvky.

Výsledkem je velice flexibilní systém, který umožňuje vývojářům upravit si grafickou reprezentaci GUI tak, jak potřebují pro konkrétní aplikaci.

K flexibilitě celé knihovny také přispívá fakt, že CEGUI nezpracovává vstup z klávesnice ani myši a raději tak nechává na vývojáři, aby tento vstup poskytl. Tím je knihovna oproštěna od provázání s API, které by se o to staralo, a které bývá většinou platformně závislé. Programátor má také možnost vstup před tím, než jej předá CEGUI, zpracovat.

Dalším důkazem velké nastavitelnosti knihovny je možnost zvolit si jednu nebo více knihoven pro specifické úlohy jako jsou rendering, kodeky pro nahrávání obrazových dat, XML parsery a nebo skriptovací modul Lua. Tímto CEGUI vychází vstříc širokému spektru uživatelů, kteří mohou používat různé knihovny. Podle zvoleného nastavení se potom CEGUI zkompiluje a obsahuje tak pouze ty části, které vývojář potřebuje.

Zmínil jsem možnost různých XML parserů, ty jsou pro CEGUI velmi důležité, protože všechny konfigurační soubory, soubory obsahující popis vzhledu GUI a soubory s popisem navrženého GUI pro výslednou aplikaci jsou ve formátu XML. CEGUI také obsahuje program pro návrh GUI, jehož výstupem je rovněž XML popis.

CEGUI je od verze 0.5.0 distribuován pod MIT licencí. Předchozí verze jsou šířeny pod licencí LGPL.

2.4 Toolkity zvolené pro aplikaci *3D správce souborů*

Pro demonstrační aplikaci jsem zvolil kombinaci knihovny veLib s toolkitem CEGUI. veLib se stará o základnu aplikace, kdy vytváří okno, zpracovává události operačního systému a poskytuje vstup od uživatele. Dále vytváří a spravuje scénu s 3D objekty, reprezentující aplikační data, a zajišťuje její vykreslování. V neposlední řadě také poskytuje snadné rozhraní pro práci s XML soubory, což je výhodné např. pro nahrávání konfigurace aplikace. I když má veLib základní podporu pro elementární prvky pro výstavbu 2D GUI, je výhodnější použít specializovaný toolkit.

Tímto toolkitem je CEGUI. CEGUI byl navržen zejména pro vývojáře počítačových her a taková hra není nic jiného než specializovaná aplikace virtuální reality. Z tohoto důvodu je připojení CEGUI ke správci souborů velmi snadné z programového hlediska a vhodné z hlediska grafického designu. CEGUI ve správci souborů používá pro vykreslování knihovnu OpenGL a nezavádí tak další externí knihovny, které by mohly zavést ostatní 2D GUI toolkity. Jediné co je potom potřeba udělat, je předání informací o vstupu (myš, klávesnice) od uživatele a zavolání funkce pro vykreslení GUI. Grafická reprezentace prvků GUI je závislá na použitém *skinu* a dá se tak vhodně sladit s další grafikou v aplikaci. Toto by nemuselo být až tak snadné s použitím toolkitů, které používají grafickou reprezentaci nativní pro daný operační systém nebo které se ji snaží přiblížit. I tady je na novějších systémech možnost použití *skinů*, ale ten uživatel často volí tak, aby byl příjemný pro celý operační systém (resp. jeho okenní manažer) a také se jej většinou snaží sladit s obrázkem na pozadí. To ovšem nemusí být zrovna nejlepší volba pro 3D správce souborů.

Více informací o implementačních detailech a návrhu integrace obou knihoven do 3D správce souborů je možno nalézt v kapitole 5.

Kapitola 3

Head Mounted Display

3.1 HMD obecně

Obrázek 3.1: HMD Visette45 SXGA (obrázek je vlastnictvím Cybermind Interactive Nederland: www.cybermindnl.com)

Head Mounted Display (HMD) je zobrazovací zařízení, které má uživatel nějakým způsobem spojené s hlavou tak, že jakkoli s ní pohne, vždy má obraz před očima. Obrázek 3.1 takové zařízení ukazuje.

HMD je značně náročný na konstrukci jak z hlediska ergonomického tak technologického. Proto v uplynulých letech vzniklo mnoho různých modelů, které se tento problém snažily řešit různými způsoby. V principu je každý HMD složený z nějakého vizuálního zobrazovacího zařízení (může být pro obě oči dohromady, nebo pro každé oko zvlášť), úchytného systému a často také sledovacího (tracking) zařízení a zvukového výstupu.

Pro obrazový výstup se používá třech zařízení. Prvním je CRT (cathode ray tube), tedy metoda, kde elektrony vystřelené z elektronového děla procházející magnetickým polem, které mění jejich dráhu, dopadají na obrazovku a rozsvěcují na ní jednotlivé body. Takovéto zařízení jsou však dosti rozměrné a těžké, protože elektrony potřebují dostatečnou vzdálenost mezi elektronovým dělem a obrazovkou na to, aby mohly být vychylovány v potřebném rozsahu. Výhodou však je vyšší kvalita obrazu a rozlišení, které může dosahovat až 1280x1024 pixelů.

Druhým typem je LCD (liquid crystal display). LCD je tenký panel tvořený molekulami tekutých krystalů umístěnými mezi průhlednými elektrodami a polarizačními filtry. Tyto filtry jsou vůči

sobě pravoúhle orientované, takže za normálních okolností nepropustí žádné světlo. Díky změně napětí mezi elektrodami se mění orientace molekul tekutých krystalů a tím orientace světelných paprsků, které jimi procházejí. Čim více se takto světelné paprsky přizpůsobí výstupnímu filtru, tím více světla panel vyzařuje. Díky tomu jsou LCD panely tenké a lehké a umožňují tak pro uživatele příjemnější konstrukci HMD. Nevýhodou je však menší kvalita a rozlišení těchto displayů, které zatím dosahuje maximálně 800x600 pixelů.

Novinkou mezi tenkými panely je technologie OLED (organic light-emitting diode). OLED panely jsou z ergonomického hlediska shodné s LCD panely, ale oproti těmto poskytují kvalitnější obraz a menší spotřebu.

Třetí alternativou je Head Mounted Projective Display (HMPD). Tady jsou k HMD připojeny malé LCD projektory, které promítají obraz do prostoru. V tomto prostoru jsou strategicky rozmístěny předměty z reflexivního materiálu, které odrážejí obraz zpět k uživateli nezávisle na úhlu dopadu. HMPD jsou tak ideální pro kolaborativní aplikace, protože každý účastník vidí obraz ze svého pohledu.

Hlavním cílem HMD jako vizuálního zobrazovacího zařízení byla snaha poskytnout uživateli grafický výstup z počítače tak, aby skutečné prostředí uživatele bylo co nejméně viditelné a umožnilo mu tak se co nejvíce ponořit do prostředí virtuálního. Tento pocit může být ještě zesílen použitím sledovacího (head tracking) zařízení, které umožní uživateli rozhlížet se kolem sebe. Výstupem tohoto zařízení je nejčastěji informace o rotaci kolem všech třech os. Proto pokud se uživatel chce ve virtuálním prostředím pohybovat, je potřeba použít některou další techniku (např. použití klávesnice).

Snaha omezit skutečné prostředí však měla za následek stíženou manipulaci s reálnými předměty nebo například interakci s kolegy. Proto začaly být některé modely HMD navrhovány tak, aby uživatel skrze ně viděl. Toho může být dosaženo konstrukčně, použitím průhledných zrcadel tak, aby zobrazovací jednotka neblokovala výhled, nebo připojením kamery. Použití kamery je zvlášť výhodné v rozšířené realitě, kde je vstupní obraz počítačově zpracován a poté odeslán na výstup HMD. Takto je možné do skutečného prostředí zakomponovat počítačem generovanou grafiku.

HMD spojené s head tracking zařízením sice poskytuje FOR[1] 360° (kamkoli se uživatel podívá, vždy má obraz před očima), ale FOV[2] je ve srovnání s ostatními zařízeními malý, obvykle kolem 30°–60°. To způsobuje pocit tunelového vidění a může některým jedincům působit problémy. Také to může ztěžovat orientaci v prostoru. Dalším konstrukčním problémem může různá vzdálenost očí. Některé modely HMD umožňují upravit vzdálenost zobrazovacích jednotek, ale ty, které to neumožňuji, mohou uživateli způsobovat zhoršené vnímání obrazu nebo až bolest očí.

Stereoskopie může být u HMD docíleno dvěma způsoby. U modelů s jednou zobrazovací jednotkou je možné použít (stejně jako u zařízení s podobnou konstrukcí, jako jsou klasické počítačové monitory) časový multiplexing, kdy jsou střídavě zobrazovány obrázky pro levé a pravé oko. Tato technika klade velké nároky na obnovovací frekvenci zařízení, protože kvůli multiplexingu bude nakonec poloviční. Dostatečnou frekvenci mají zatím jen CRT obrazovky. U modelů se dvěma zobrazovacími jednotkami je možné každému oku zobrazovat jiné obrázky najednou a vyhnout se tak problému s obnovovací frekvencí. Problémem ale je, že grafické zařízení, které posílá data HMD, musí podporovat dva simultánní kanály.

V praxi mohou HMD používat návrháři a inženýři pro zobrazování schémat v 3D s využitím stereoskopie, policie nebo armáda je může využívat pro obohacování skutečného obrazu taktickými informacemi a v neposlední řadě se s vývojem stále levnějších technologií začínají HMD prosazovat také v zábavním průmyslu, zejména pak počítačových hrách.

[1] field of regard – rozsah fyzického prostoru ve kterém jsou zobrazována vizuální data

[2] field of view – vizuální rozsah, který může být najednou zobrazen na zařízení

Více informací o HMD, head tracking zařízeních a technikách jejich použití je možné nalézt v [1] nebo [3]. http://www.stereo3d.com/hmd.htm obsahuje přehledný seznam velkého množství HMD.

3.2 eMagin Z800 3DVisor

Obrázek 3.2: eMagin Z800 3DVisor (obrázek je vlastnictvím eMagin: www.emagin.com)

3.2.1 Konstrukce

Konstrukce Z800 3DVisoru je klasická ve smyslu, který jsem popsal v předchozí podkapitole. Sestává z audiovizuálního a úchytného systému a head tracking zařízení. Pro obrazový výstup je použito dvou OLED displayů s rozlišením 800x600 pixelů, kontrastem 200:1 a svítivostí 50 cd. Displaye nabízejí FOV zhruba 40°. Zařízení obsahuje také stereo sluchátka a mikrofon s potlačením šumu. Úchytný systém nabízí velkou míru nastavitelnosti, je možné upravovat pásky, které drží HMD na hlavě, stejně jako je možné upravovat vzdálenost a sklon displayů od očí. Velkou výhodou je také možnost nastavení vzdálenosti mezi oběma displayi. Úchytný systém se tak snaží vyjít vstříc co největšímu počtu a rozmanitosti uživatelů. 3DVisor má navíc zabudovaný head tracker složený ze tří kompasů, gyroskopů a akcelerometrů a umožňuje tak sledovat rotační pohyb ve všech třech osách.

Celý HMD je díky použití OLED displayů velmi lehký a má malou spotřebu, což může být výhodné zejména pro uživatele notebooků. Uchycení na hlavě je z ergonomického hlediska precizně vyřešeno a 3DVisor je tak pohodlný a snadno se s ním manipuluje. Jedinou nevýhodou je objemnější datový kabel, který může v některých situacích omezovat pohyb zařízení.

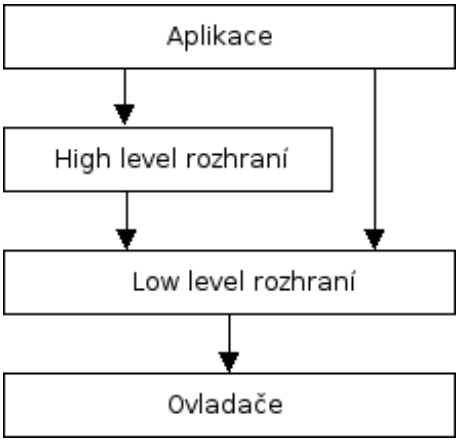
Samotné zařízení je k počítači připojeno USB portem, kterým probíhá obousměrná komunikace, VGA kabelem, kterým proudí obrazová data, a několika kabely pro zvukový vstup a výstup. Zařízení je napájeno buď to přes USB nebo externě.

3.2.2 Programování 3DVisoru

Rozhraní pro programování aplikací využívající Z800 3DVisor se dají rozdělit do vrstev, které ukazuje obrázek 3.3.

Ovladače jsou na nejnižší úrovni a poskytují nejzákladnější prostředky pro komunikaci HMD s operačním systémem. Aktuálně jsou ovladače dostupné pouze v binární formě pro systémy Windows 9x/ME a Windows 2000/XP.

Low level rozhraní stojí těsně nad ovladači a je to první rozhraní, se kterým může vývojář pracovat. Programátoři by nikdy neměli přistupovat přímo k ovladačům. Na této úrovni mohou



Obrázek 3.3: struktura programových rozhraní

vývojáři inicializovat zařízení, nastavovat jeho parametry, dotazovat se na data z head trackeru a řídit funkcionalitu emulace myši.

Pro zpřístupnění informací o rotaci HMD jsou použity dva různé mechanismy. Prvním je emulace myši, kdy jsou pohyby uživatele zasílány aplikaci formou zprávy `WM_MOUSEMOVE` jako pohyby myši. Výhodou tohoto přístupu je snadná implementace, pokud už například aplikace obsluhuje pohyb myši (např. ve hrách pro rozhlížení se po okolí) tak bude tento pohyb automaticky generován head trackerem. Nevýhodou ovšem je, že v některých situacích může být takové chování nevhodné, např. při navigaci v GUI a podobně. Dalším problémem je, že pohyb myši je dvourozměrný, takže se zpracovávají rotace pouze kolem dvou os.

Druhým je potom přístup k samotným datům z head trackeru. API 3DVisoru nabízí možnost dotázat se přímo na informace o rotaci kolem jednotlivých os. Tyto informace jsou ve formě úhlů ve stupních a dají se použít například pro sestavení rotační matice. Výhodou tohoto přístupu je fakt, že myš zůstane použitelná pro jiné úlohy. Tak je možné kombinovat data ze dvou (nebo více) vstupních zařízení. Optimální frekvence dotazů je 10-12 Hz. Pokud chce programátor analyzovat akcelerační data (např. pro použití gest) tak 30 Hz. Limitem zařízení je 33 Hz.

K tomuto rozhraní je možné přistupovat pomocí jazyka C nebo C++ a také přes platformu Microsoft .NET.

High level rozhraní zapouzdřuje předchozí rozhraní a nabízí tak programátorům nástroje, které staví na funkcionalitě popsané výše. Těmito nástroji jsou EMATracker a EMAGestureAnalyzer.

Aby se zabránilo opotřebení displayů 3DVisoru, tak v případě delší nečinnosti se HMD vypíná. Proto pokud chce programátor umožnit uživateli používat zařízení po delší dobu, musí jednou za určitou periodu (asi 5 minut) zaslat zařízení zprávu *Keep Alive*. EMATracker tento proces automatizuje, takže vývojář jej pouze zapne a dál se nemusí o nic starat, Navíc je tato kontrola vykonávána v separátním vlákně, aby co nejméně rušila běh aplikace.

Další funkcí EMATrackeru je překlad pohybů HMD na aplikační data. EMATracker analyzuje vstupní data z head trackeru a převádí je na zprávy, které zasílá aplikacím. Tyto zprávy mohou být použity například pro rolování obsahu oken.

EMAGestureAnalyzer jak název napovídá analyzuje gesta. Gesty je zde míněna určitá sada pohybů (může to být i pohyb jeden), přičemž roli nemusí hrát jen směr ale i zrychlení. Analyzátor tak sleduje vstupní data z head trackeru a pokud detekuje pohyb nebo pohyby, které má definované jako určité gesto, zavolá obslužnou funkci, kterou má s tímto gestem spojenou. Tuto obslužnou funkci definuje a registruje programátor.

Toto rozhraní (stejně jako rozhraní předchozí) je přístupné z jazyka C, C++ a přes platformu Microsoft .NET.

Podrobnější informace o programovém modelu Z800 3DVisoru lze nalézt v dokumentaci přiložené u SDK, které není volně dostupné a je potřeba si jej vyžádat u podpory společnosti eMagin.

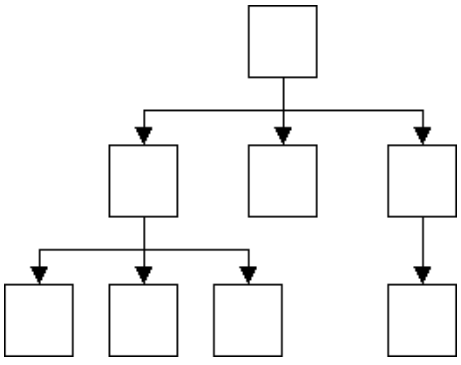
Kapitola 4

Vizualizace souborového systému

4.1 Souborový systém obecně

Souborový systém je obecně struktura, jejímž cílem je poskytnout uživateli možnost ukládat a spravovat data na záznamovém médiu. Tato struktura může být plochá, kdy všechna data jsou na stejné úrovni, nebo hierarchická, kdy jsou data řazeny podle potřeby do různých úrovní, které se mohou dále větvit. Příkladem souborového systému s plochou strukturou je *Macintosh File System*. Tady je ovšem potřeba dodat, že správce tohoto systému zpřístupňoval data v něm uložená tak, že systém se jevil jako částečně hierarchický. Nicméně i tak byl omezen tím, že každý soubor musel mít jedinečné jméno v rámci celého systému.

Plochá struktura tak byla rychle nahrazena strukturou hierarchickou. Příklad takové struktury ukazuje obrázek 4.1. Tato struktura by se dala popsat jako orientovaný acyklický graf[1] s tím omezením, že se větví od kořenového prvku směrem dál. Takže nemůže nastat situace, kdy některý prvek bude mít více rodičovských prvků. Pro systém zakládání dokumentů do složek a složek do šuplíků, ze kterého se souborový systém vyvinul, je toto chování přirozené. Není možné aby jeden dokument byl na dvou místech zároven (tedy možné to je, ale to by vedlo k duplicitě dat a k popření smyslu tohoto systému).

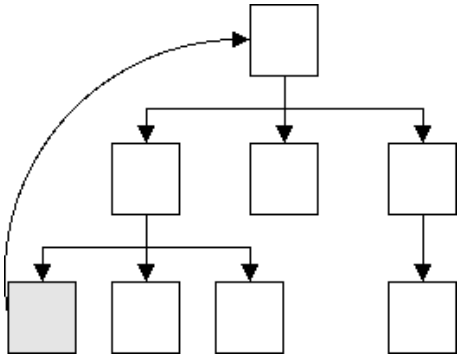


Obrázek 4.1: příklad hierarchické struktury

Souborový systém se sice vyvinul ze systému dokumentů a záložek, ale dál už kráčel svou vlastní cestou a to po mnoho let. Takže s tím jak se vyvíjel, přibývaly další změny. Jednou takovou významnou změnou, která narušila strukturu DAG, bylo začlenění odkazů. S odkazy mohou různé

[1] directed acyclic graph - DAG

souborové systémy pracovat různě, ale v principu je to prvek struktury, který nenese žádná vlastní data, ale odkazuje na jiný prvek. Přehledně to ukazuje obrázek 4.2. Odkazy jsou důležité také z pohledu vizualizace, protože mohou zobrazení struktury, která je obsahuje, zkomplikovat.



Obrázek 4.2: příklad hierarchické struktury s odkazem

Souborové systémy využívající hierarchické struktury se vyvíjí již mnoho let a proto vznikly nejrůznější varianty které postupně podporovaly větší a větší soubory, uměly soubory více zhustit nebo podporovaly žurnálování. Více informací o detailech jednotlivých systémů lze najít v [7].

V poslední době vznikají také souborové systémy, které se snaží obejít omezení systémů s hierarchickou strukturou, které zamezuje prvku mít více rodičů. Tyto tzv. sémantické souborové systémy jsou vyvíjeny s myšlenkou řadit a ukládat soubory ne podle hierarchické příslušnosti, ale podle jejich sémantického významu. Tímto významem může např. být zda se jedná o CAD dokument, video soubory z dovolené, zdrojové texty daného programu a další, uživatelem specifikovatelné významy. Každý soubor může být také označen jedním nebo více klíčovými slovy, podle kterých je možné je vyhledávat a řadit. Principem sémantických souborových systémů tedy je dát uživateli možnost vyhledávat a pracovat se soubory podle jejich významu a vztahu, který k nim uživatel má. Více informaci o těchto systémech lze najít v [2].

4.2 Vývoj vizualizace a její cíl

S prvním výskytem souborových systémů vyvstal i problém jak data v něm uložené předložit uživateli. V počátcích vývoje těchto systémů byl k dispozici výstup na obrazovku (nebo jiné médium) ve formě textu. Proto první podoby vizualizace byly textové seznamy jednotlivých souborů a adresářů. S tím, jak se do popředí dostávaly systémy s hierarchickou strukturou, začalo být potřeba zobrazovat data na jednotlivých úrovních, v jednotlivých adresářích. Uživatel se tak mohl pomocí příkazů operačního sytému pohybovat ve stromové struktuře souborového systému a zobrazovat si obsah adresářů ve kterých se nacházel. Tím docházelo k přirozené strukturalizaci zobrazovaných dat, kterých mohlo být v celém systému obrovské množství. Tento přístup, kdy uživatel viděl jen obsah adresáře, ve kterém se zrovna nacházel, nenaskýtal příliš možností zobrazit sytém jako celek a nechával na uživateli aby si tuto představu vytvořil sám.

S příchodem grafického výstupu se objevila možnost zobrazovat souborový systém i jinou formou než jen textem. Proto se začalo zkoumat, jaké další metody mohou být vhodné pro vizualizaci a co tyto metody mohou nabídnout. Jak je přirozené (a z historie patrné), s rozvojem nové platformy se začaly zkoušet různé techniky využívající jejího potenciálu. Nakonec nejlépe použitelné techniky kombinovaly grafickou formu s textovou. Jako nejpřehlednější a nejsnáze použitelná forma se tak

vyvinul textový seznam podpořený grafickým náhledem (ikonami). Ikony dávají možnost udělat si rychlou představu o obsahu daného adresáře, popř. s vhodným řazením rychle lokalizovat skupinu souborů nebo adresářů, které uživatele zajímají, a k nim přiřazený text pak upřesňuje jednotlivé detaily. Další pomůckou pro navigaci je zobrazení stromové struktury souborového systému. Ten ale může obsahovat obrovské množství dat a proto tato struktura často zohledňuje jen adresáře. Obsah vybraného adresáře se pak zobrazuje např. v dalším okně.

Důkazem a příkladem tohoto kompromisu budiž všechny široce používané souborové manažery jako Průzkumník nebo Total Commander v systému Windows nebo např. Nautilus nebo Konqueror v systému Linux.

S příchodem grafického hardwaru, který umožňoval zobrazovat náročné prostorové scény v reálném čase, se v podstatě opakovala situace, která nastala při přechodu z textového režimu do grafického. Opět se objevila snaha co nejvíce využít potenciálu nové platformy a provádět celou vizualizaci v prostoru. A opět se lepším řešením ukázala být kombinace nové platformy se starou, která poskytovala už zažitou přehlednost a efektivitu práce. I tak jsou zatím souborové manažery využívající prostorového zobrazení spíše experimentální. Některé zajímavé projekty je možné najít na http://printf.nl/item/3. Trochu výjimkou je Tactile3D (http://www.tactile3d.com/), který je velmi propracovaný. Jeho autor shrnul některé své myšlenky, které formovaly vývoj aplikace, ve svém článku [8].

Ať už je souborový systém zobrazován textově, 2D nebo 3D grafikou, vždy je cílem nabídnout maximální možnou přehlednost, účelnost a efektivitu a pohodlnost práce. Některé z těchto cílů se mohou dostávat do konfliktu a tak bývá návrh souborových manažerů otázkou zvolení vhodného kompromisu, který je ovlivněn částí, na kterou chtěli dát vývojáři důraz.

4.3 Používané metody vizualizace

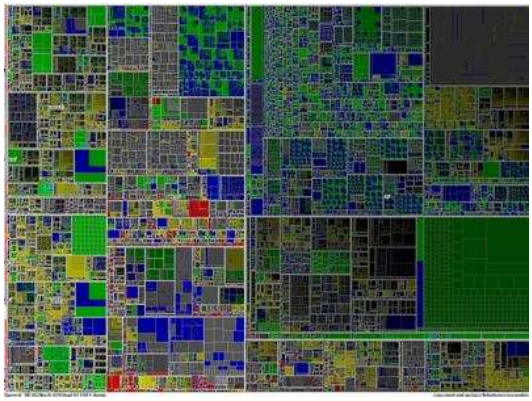
Na tomto místě bych rád popsal několik metod vhodných pro vizualizaci souborového sytému. Jedná se převážně o techniky původně používané ve 2D vizualizaci, které ale mohou být vhodně rozšířeny do prostoru.

První metodou je zobrazení hierarchické struktury pomocí *tree maps*. Zde je kořenový prvek reprezentován nejčastěji formou obdélníku (ale může to být i obecný polygon) a podprvky jsou do něj vnořeny, takže původní obrazec se rekurzivně dělí na menší a menší části, viz obrázek 4.3. Relativní velikost těchto částí pak může zobrazovat různou sémantiku dat, které strom reprezentuje (např. velikost souborů).

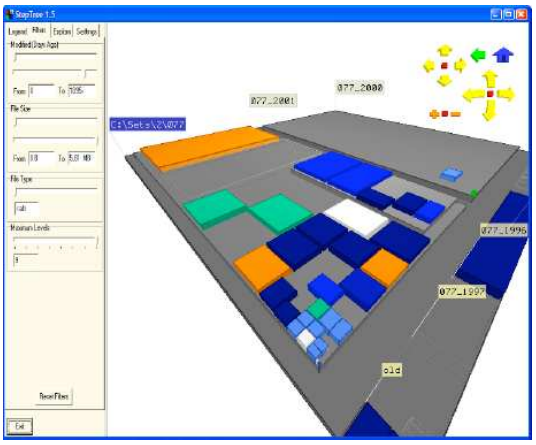
Rozšíření této techniky do prostoru pak prezentuje článek [4]. Jednotlivé polygony jsou vyzvednuty do prostoru v závislosti na velikosti souborů. Tato 3D prezentace pak může být vykreslována s různou průhledností v závislosti na čase poslední změny souborů (viz obrázek 4.4).

Další metoda je založena na *bubble trees* [5]. Zde je hierarchická struktura zobrazena jako do sebe zanořené bubliny, kde každá bublina představuje jeden prvek stromu a může obsahovat další podprvky (viz obrázek 4.5). Pro snadnější navigaci stromem je bublina představující kořenový prvek uzavřena a nejsou tedy vidět žádné její prvky. Po jejím otevření se tyto prvky zpřístupní. Takto je možné navigovat napříč strukturou. I tak se ale může nahromadit spousta dat k zobrazení a proto bubliny, které přímo nesouvisí s tou, na kterou se uživatel právě zaměřuje, jsou posunuty pryč z vizuálního kontextu (například zmenšením).

Podobnou metodou je použití *cone trees* [6]. Tato technika zobrazuje stromovou strukturu formou navzájem provázaných kuželů, viz obrázek 4.6. Nevýhodou je překryv jednotlivých kuželů, který může způsobit zhoršenou orientaci v grafu. Proto vždy, když uživatel vybere některý prvek, strom se natočí tak, aby tento prvek a cesta k němu byla zřetelně viditelná. Jednotlivé prvky je



Obrázek 4.3: Tree mapa zobrazující milion položek (vlastníkem obrázku je Jean-Daniel Fekete: http://www.cs.umd.edu/hcil/VisuMillion)

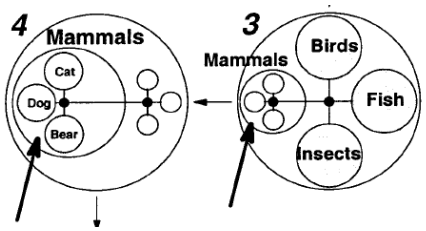


Obrázek 4.4: Ukázka aplikace StepTree využívající tree mapy v prostoru (obrázek převzat z [4])

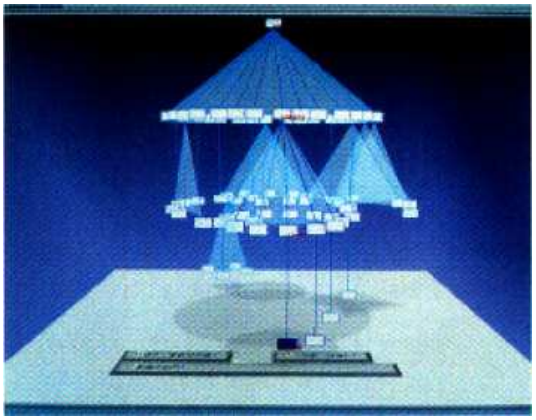
možné uzavírat a tak zobrazovat jen ty části struktury, které uživatele zajímají. Uživatel také může (např. pro zlepšení viditelnosti nebo vytvoření lepší paměťové mapy) strukturu přeorganizovat přesouváním jednotlivých uzlů stromu.

Existují i další metody, ale účelem této práce není poskytnout jejich vyčerpávající výčet. Spíš je důležité uvědomit si co tyto metody řeší, jak to řeší, co mají společné a v čem se naopak liší.

Všechny tyto metody se snaží vyřešit problém vizualizace hierarchické struktury reprezentující velké množství dat (tisíce až statisíce položek). Protože není možné zobrazit všechna data najednou, poskytují nějaký mechanismus pro zvýraznění těch prvků, které uživatele zajímají, a potlačení těch zbylých. Zároveň se však snaží nabídnout celkový pohled na strukturu, aby si uživatel mohl právě zobrazovaná data zařadit do celkového kontextu. Jinak by pro vizualizaci stačil obyčejný seznam prvků. Často se tyto metody snaží využít třetího rozměru pro zvýraznění určité sémantiky spjaté se zobrazovanými daty (např. velikost souborů, čas poslední změny, apod.).



Obrázek 4.5: Bubble tree (obrázek převzat z [5])



Obrázek 4.6: Cone tree (obrázek převzat z [6])

4.4 Vizualizace ve 3D správci souborů

Návrh rozhraní ve 3D správci souborů určovalo několik klíčových hledisek:

- uživatel používá HMD, které snímá rotaci hlavy, ale ne její polohu
- vstupními zařízeními jsou myš a klávesnice
- snaha o maximální přehlednost a minimální překryv grafických prvků
- snaha o maximální přímočarost a účelnost interakce s rozhraním
- snaha o co nejkratší osvojovací proces

HMD poskytuje informace o rotaci uživatelovy hlavy ale ne o její poloze. Proto pokud by se chtěl uživatel v prostoru pohybovat, musel by k tomu použít další vstupní zařízení a vhodnou techniku (například klasický pohyb pomocí kurzorových šipek, nebo metoda point-n-click, kde se uživatel přesune na místo, kam klikne). Pohyb uživatele v prostoru ale může být matoucí a může způsobovat ztrátu kontextu v rámci celé hierarchie (představme si např., že by uživatel vplouval do koulí představující adresáře; po několika zanořeních by bylo jistě složité udržovat si přehled o ostatních adresářích a souborech). Jako lepší řešení se jeví udržovat uživatele stále na jednom místě a celou strukturu kolem něj vhodně rozprostřít.

Míra rozprostření je závislá na rozsahu pohybu hlavy uživatele, který je pro něj ještě pohodlný (otáčení hlavy do krajních poloh po delší dobu by mohlo způsobovat bolesti krčních svalů). Při rozmisťování grafických elementů je nutné dbát na to, aby se co nejméně překrývaly jednotlivé

úrovně zanoření. Jako vhodná metoda zobrazení se jeví varianta technik *tree maps* a *bubble trees*, upravená tak, aby respektovala popsanou kompozici.

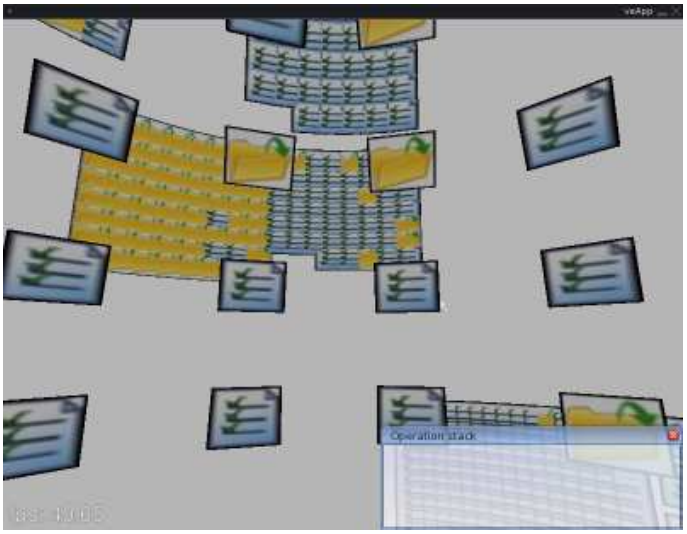
Myšlenkou tedy je, ne aby se uživatel přizpůsoboval prostředí (např. pohybem a hledáním dat, která ho zajímají), ale aby se prostředí přizpůsobilo uživateli. Uživatel chce mít maximální pohodlí při interakci s rozhraním a proto je vhodné, aby měl všechny prvky tzv. na dlani.

Dalším hlediskem je složitost rozhraní vs. osvojovací proces[2]. Jak píše Tristan Grimmer ve svém článku [8], nová rozhraní musí čelit problému, že uživatelé na ně nejsou zvyklí a proto si musí nejprve vybudovat paměťové mapy a vzory, které jim umožní tato rozhraní efektivně používat. Pokud bude ale tento proces příliš zdlouhavý nebo náročný, uživatelé se raději vrátí k osvědčeným rozhraním, se kterými už mají zkušenosti. Proto musí vývojáři rozhraní navrhovat a upravovat tak, aby byl proces osvojení akceptovatelný nebo dostatečně zábavný. Je to opět problém zvolení vhodného kompromisu.

3D správce souborů se vydal cestou, která se snaží udělat rozhraní dostatečně intuitivní a známé, aby jeho osvojení bylo co nejsnažší.

4.4.1 První verze rozhraní

Tato první verze rozhraní, kterou ukazuje obrázek 4.7, využívá rozprostření elementů stromové struktury kolem uživatele. Dobře je to patrné na obrázku 4.8, kde je kamera posunuta ze středové pozice, aby byla vidět kulová kompozice struktury. Obrázek ukazuje dvě úrovně zanoření, kde prvky formující vnější kouli jsou podadresáře a soubory, které obsahují adresáře tvořící kouli vnitřní.

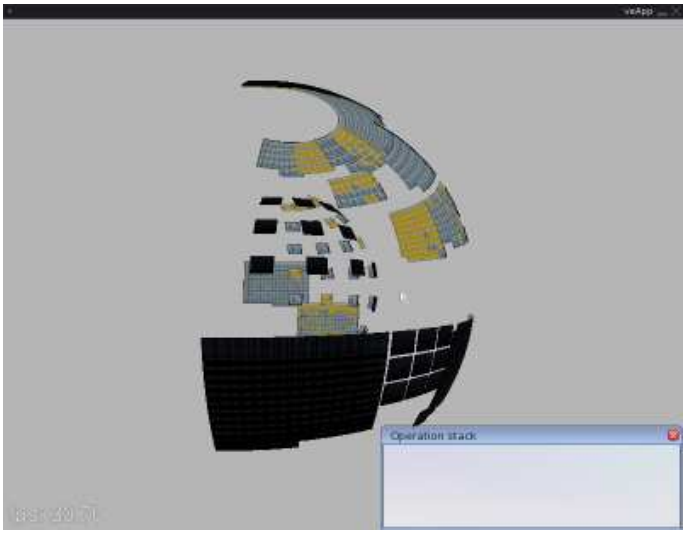


Obrázek 4.7: První verze 3D správce souborů

Rozprostření netvoří celou kouli kolem uživatele, ale pouze polokouli, která je přímo před ním, takže její okraje končí 90° na všechny strany. Algoritmus, který generuje rozložení prvků v prostoru, bere jako parametry rozsah úhlů v rovině *yz* a *xy*, takže je snadné toto rozložení kontrolovat. Dalším parametrem je potom vzdálenost mezi vrstvami zanoření.

Cílem této vizualizace bylo vyzkoušet rozprostření prvků v praxi a zjistit, jaký dopad to bude mít na přehlednost a orientaci v prostoru. Výsledek je takový, že dochází k větší míře překryvu prvků (která velmi strmě narůstá s počtem úrovní zanoření). Dalším problémem je omezený prostor

[2] learning curve

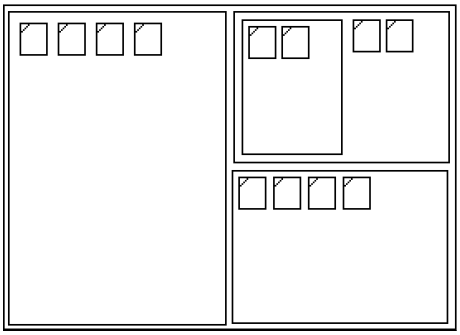


Obrázek 4.8: Ukázka rozprostření prvků

pro každý podadresář. Každý prvek je reprezentován grafickým objektem o stejné velikosti a proto při velkém počtu souborů nebo adresářů dojde k přeplnění vyhrazeného prostoru a k prudkému zhoršení přehlednosti. Částečným řešením by mohla být dynamická změna vyhrazeného prostoru v závislosti na počtu objektů v adresářích na stejné úrovni.

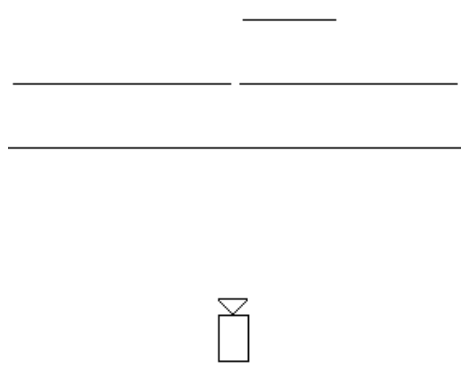
4.4.2 Vylepšená verze rozhraní

Lepší metodou je použití prostorové varianty *tree maps* nebo *bubble trees*, která je vhodně upravena tak, aby zohledňovala použití HMD. Jak naznačuje obrázek 4.9, struktura souborů je reprezentována obdélníky, které představují adresáře, a objekty, které představují soubory. Prvky, které náleží některému adresáři, jsou zobrazeny tak, že leží v prostoru jeho obdélníku. Navíc jsou posunuty v prostoru směrem od uživatele (obrázek 4.10), což posiluje vnímání zanoření jednotlivých úrovní.



Obrázek 4.9: pohled na strukturu ze předu

Jak bylo popsáno v podkapitole o metodách vizualizace, je také potřeba mít nějaký mechanismus, který zdůrazní data pro uživatele zajímavá a potlačí ta nezajímavá. Toho se dá docílit prostým zmenšením obdélníků, které reprezentují nezajímavá data. Při zmenšení je také vhodné skrýt jejich



Obrázek 4.10: pohled na strukturu z hora

obsah, aby se zachovala lepší přehlednost. Navíc je možné tyto obdélníky zprůsvitnit, což je ještě více potlačí z vizuálního kontextu. Z tohoto pohledu by se vzniklé rozhraní dalo popsat jako ZUI[3].

Proměnlivá velikost obdélníků by potom měla zajistit, že se všechny vejdou do prostoru pro ně vyhrazeného. V praxi, s velkým množstvím souborů, to však i tady může být problém.

Důležitým aspektem každého správce souborů je možnost mít více uživatelsky definovaných "pohledů" do souborového systému. Díky ním může uživatel pracovat ve více adresářích na jednou, nebo přenášet data mezi adresáři aniž by je musel neustále procházet tam a zase zpátky. Je to podobné dnes klasickým okenním manažerům, které umožnují otevřít více oken pro prohlížení adresářů. Proto i vylepšená verze rozhraní 3D právce souborů nabízí více pohledů, se kterými může uživatel manipulovat dle libosti. Oproti 2D oknům však není žádoucí, aby se docházelo k vzájemnému překryvu, protože v prostoru by to bylo matoucí a nepřehledné.

Nová verze rozhraní řeší některé problémy verze předchozí, jako horší přehlednost nebo částečně problém s prostorem. Navíc nabízí možnost mít více pohledů do souborového systému (v předchozí verzi by se to provádělo obtížně, kvůli nárokům na prostor).

[3] Zoomable User Interface - volně přeloženo asi jako Zvětšovatelné Uživatelské Rozhraní

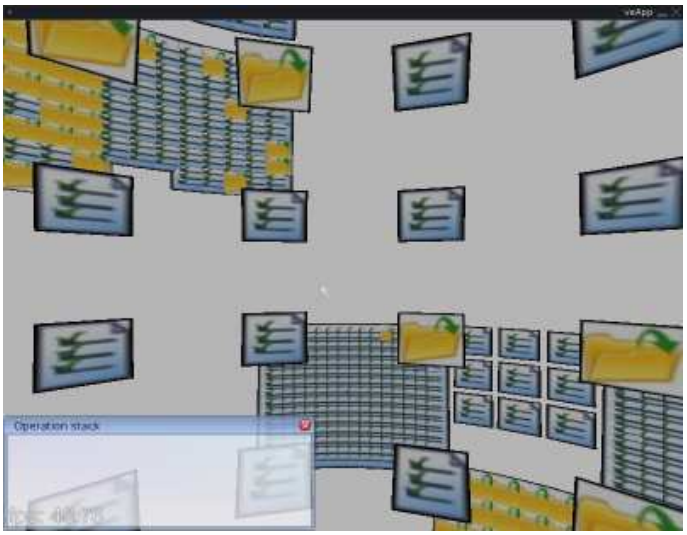
Kapitola 5

Demonstrační aplikace 3D správce souborů

5.1 Úvod

3D správce souborů je aplikace pro prohlížení a správu souborů a adresářů souborového systému. Typickými aplikacemi toho typu budiž například Průzkumník Windows v systému Windows nebo Nautilus v sytému Linux (Gnome). Avšak co tyto aplikace odlišuje je přidaný třetí rozměr v reprezentaci dat.

Cílem aplikace 3D správce souborů je tedy efektivně využit tento nový prvek a to jak z hlediska přístupu k datům tak jejich reprezentace. Tato aplikace by měla ukázat, zda může třetí rozměr napomoci přehlednosti zobrazených dat a zda může přispět k pohodlnosti, rychlosti a efektivitě práce s těmito daty.



Obrázek 5.1: 3D správce souborů

5.2 Implementace

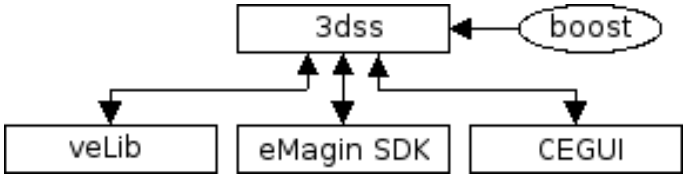
Aplikace je implementována pomocí toolkitů veLib a CEGUI. Knihovna veLib poskytuje základnu pro vývoj aplikací virtuální reality. Nabízí abstraktní rozhraní pro tvorbu a správu okna a správu událostí operačního systému. Poskytuje správu vstupních zařízení a rovněž spravuje scénu a vykresluje 3D grafiku pomocí knihovny OpenGL.

CEGUI je knihovna pro tvorbu 2D uživatelských rozhraní a práci s nimi. Nabízí širokou nabídku standardních prvků rozhraní a jejich grafickou reprezentaci načítá ze souborů, takže je možné ji přizpůsobit dané aplikaci. Grafiku vykresluje pomocí knihovny OpenGL, což zajišťuje snadnou integraci s knihovnou veLib.

Demonstrační aplikace také využívá knihovnu *boost*, která nabízí kolekci nástrojů pro práci s pamětí, ukazateli, textem, obecnými algoritmy, iterátory a spoustu dalších rutin a tříd, které často vývojáři potřebují a které nejsou standardní součástí jazyka C/C++. Aplikace 3D správce souborů z této knihovny využívá *smart pointery*, které zjednodušují práci s dynamicky alokovanými objekty. Zvláště efektivní je jejich použití ve spojení se standardními kontejnery STL (jako jsou pole, seznamy, aj.).

Další použitou knihovnou je eMagin SDK, která zajišťuje komunikaci s Z800 3DVisorem. Následující podkapitola pojednává o její integraci v aplikaci.

Vztah mezi těmito knihovnami a jejich vzájemnou integraci spolu se zbytkem demonstrační aplikace naznačuje obrázek 5.2.



Obrázek 5.2: Vztah knihoven a aplikace

Jednou z výhod knihovny veLib je její podpora XML souborů. Konfigurace aplikace a informace o objektech, které tvoří 3D rozhraní, jsou tak načítány z externího souboru, což minimalizuje potřebu zasahovat do kódu, kvůli některým změnám. CEGUI, jak je popsáno výše, rovněž načítá konfiguraci a grafickou reprezentaci prvků rozhraní z externích souborů. Výsledná kombinace pak umožňuje měnit vzhled aplikace a prvků 2D i 3D rozhraní jen změnou konfiguračních souborů, tedy bez potřeby zásahu do kódu aplikace a její následné rekompilace.

5.2.1 Integrace HMD

Jak bylo popsáno v podkapitole 3.2.2, Z800 3DVisor má aplikační rozhraní propracované jak na nižší tak i na vyšší úrovni. Demonstrační aplikace využívá pouze low level rozhraní. V praxi to pak vypadá tak, že po inicializaci zařízení se v hlavní smyčce periodicky volá (potřebná frekvence viz 3.2.2) metoda `EMADevice::PollHeadTrackerRawData`, která vrací informace o rotaci zařízení.

Z těchto informací se potom sestaví vektor, jehož jednotlivé hodnoty udávají otočení v dané ose. Tento vektor se potom použije pro rotaci kamery.

Dále je potřeba jednou za určitou periodu (asi 5 minut) poslat zařízení zprávu *KeepAlive*, aby se nepřepnulo do úsporného režimu. Toho se docílí zavoláním metody `EMADevice::KeepAlive`, která nuluje odpočet pro přechod do úsporného režimu.

5.3 Rozsah implementace

Aktuálně je implementována první verze rozhraní popsaného v podkapitole 4.4, modifikována tak, aby poskytovala více pohledů do souborového systému. Novější verze bude předmětem dalšího rozšíření této aplikace.

Kapitola 6

Závěr

6.1 Zhodnocení práce

Tato práce se snažila poskytnou náhled na návrh a tvorbu aplikací virtuální reality využívající 3D uživatelská rozhraní. Jejím cílem bylo navrhnout rozhraní pro práci se souborovým systémem tak, aby bylo přehledné, snadné a účelné na použití a nevyžadovalo zdlouhavý osvojovací proces.

Využití HMD se ukázalo být jako dvojsečná zbraň. Výhodou je větší ponoření do virtuálního prostředí a možnost snímat pohyby uživatelovy hlavy, čehož se dá využít pro efektivnější rozvržení dat v prostoru. Na druhou stranu HMD blokuje reálné prostředí a proto je interakce s ním podstatně složitější, zejména pak využití klávesnice. Řešení tohoto problému navrhuje následující podkapitola.

Z této práce je vidět, že tvorba prostorového souborového manažeru není jednoduchá, protože se zde mísí několik protichůdných aspektů a kompromisů, které je potřeba vyvažit a hlavně je potřeba si zdůvodnit použití třetího rozměru. Je nutné položit si otázku co nového nebo jaká vylepšení může přechod z roviny do prostoru přinést, zda může nějak obohatit aplikace, které už existují.

Myslím si, že 3D správce souborů ve spojení s HMD (a navíc třeba snímanými rukavicemi) může být velice efektivní a intuitivní nástroj, ale těžko si představit běžného uživatele, který bude takovou aplikaci využívat při své práci na PC. Využití takové aplikace bude spíše pro specifické systémy a problémy, kde má třetí rozměr co nabídnout a kde budou patřičně přizpůsobena nebo specializována pracoviště pro práci s uvednými V/V zařízeními.

6.2 Možné rozšíření

Jelikož HMD blokuje pohled do reálného prostředí a uživatel tak nevidí, s čím a jak manipuluje, je vhodné tuto manipulaci graficky reprezentovat ve virtuálním prostředí. Toho je možné docílit použitím datových rukavic, které snímají pohyb uživatelových rukou a prstů (podrobnější informace o různých typech rukavic lze najít v [3]). Některé typy rukavic dokonce nabízejí hmatovou zpětnou vazbu, aby interakci přiblížily skutečnosti. Takto by bylo možné přenést interakci plně do virtuálního prostředí, kde by mohla být virtuální klávesnice, na které by uživatel mohl psát. Tímto způsobem by se dalo virtualizovat jakékoli vstupní zařízení.

Další možností ovládání pomocí snímaných rukavic je použití gest. Na nejrůznější pohyby rukou a prstů by se daly namapovat volitelné příkazy, jako např. rozbalení nabídky, kopírování nebo mazání souborů, apod.

Literatura

[1] Oliver Bimber a Ramesh Raskar. *Spatial Augmented Reality: Merging Real and Virtual Worlds*. A K Peters, Ltd., 2005. ISBN 1-56881-230-2.

[2] David Gifford aj. Semantic file systems. In *ACM Operating Systems Review, October 1991*, pages pp. 16–25, 1991.

[3] Doug A. Bowman aj. *3D user interfaces: theory and practice*. Addison-Wesley/Pearson Education, 2005. ISBN 0-201-75867-9.

[4] Thomas Bladh, David A. Carr, and Jeremiah Scholl. Extending tree-maps to three dimensions: A comparative study. In *APCHI 2004: Proceedings of the 6th Asia-Pacific Conference on Computer-Human Interaction*, pages 50–59. Springer Berlin / Heidelberg, 2004.

[5] Richard Boardman. Bubble trees the visualization of hierarchical information structures. In *CHI '00: CHI '00 extended abstracts on Human factors in computing systems*, pages 315–316, New York, NY, USA, 2000. ACM Press.

[6] George G. Robertson, Jock D. Mackinlay, and Stuart K. Card. Cone trees: animated 3d visualizations of hierarchical information. In *CHI '91: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 189–194, New York, NY, USA, 1991. ACM Press.

[7] WWW stránky. File system - wikipedia, the free encyclopedia. http://en.wikipedia.org/wiki/File_system.

[8] WWW stránky. Where computer interfaces are going : 3d beyond games. http://www.tactile3d.com/overview/documentation/interface_article/article.html.