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

VYUKOVÝ PROGRAM PRO DEMONSTRACI METOD TEXTUROVÁNÍ 3D OBJEKTŮ

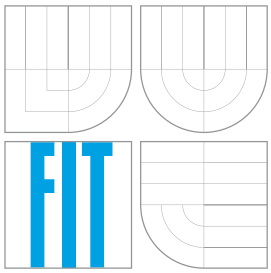
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

MICHAL KUKLA

BRNO 2008

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

VYUKOVÝ PROGRAM PRO DEMONSTRACI METOD TEXTUROVÁNÍ 3D OBJEKTŮ

EDUCATION COMPUTER PROGRAM FOR DEMONSTRATION METHODS OF 3D OBJECTS TEXTURING

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — MICHAL KUKLA
AUTHOR

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

BRNO 2008

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačové grafiky a multimédií Akademický rok 2007/2008

Zadání bakalářské práce

Řešitel: **Kukla Michal**
Obor: Informační technologie
Téma: **Vyukový program pro demonstraci metod texturování 3D objektů**
Kategorie: Počítačová grafika

Pokyny:

1. Prostudujte problematiku texturování 3D objektů v počítačové grafice
2. Prostudujte problematiku tvorby výukových a demonstračních programů
3. Navrhněte výukový program pro demonstraci principu texturování 3D objektů v počítačové grafice
4. Implementujte navržený program ve vybraném jazyce (C/C++, Java, Python, C#)
5. Ve vytvořeném programu implementujte popsané principy

Literatura:

- Žara J., Beneš B., Felkel P.: Moderní počítačová grafika. 1. vyd. Praha, Computer press 1998, 448 s., ISBN 80-7226-049-9

Při obhajobě semestrální části projektu je požadováno:

- Bez požadavků.

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 nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Kršek Přemysl, Ing., Ph.D.**, UPGM FIT VUT
Datum zadání: 1. listopadu 2007
Datum odevzdání: 23. ledna 2008

L.S.

doc. Dr. Ing. Pavel Zemčík
vedoucí ústavu

LICENČNÍ SMLOUVA
POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami

1. Pan

Jméno a příjmení: **Michal Kukla**
Id studenta: 84284
Bytem: Hodžova 8, 949 01 Nitra - Staré Mesto
Narozen: 15. 04. 1985, Nitra
(dále jen "autor")

a

2. Vysoké učení technické v Brně

Fakulta informačních technologií
se sídlem Božetěchova 2/1, 612 66 Brno, IČO 00216305
jejímž jménem jedná na základě písemného pověření děkanem fakulty:

.....
(dále jen "nabyvatel")

Článek 1
Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):
   bakalářská práce

Název VŠKP: Vyukový program pro demonstraci metod texturování 3D objektů
Vedoucí/školitel VŠKP: Kršek Přemysl, Ing., Ph.D.
Ústav: Ústav počítačové grafiky a multimédií
Datum obhajoby VŠKP: .....

VŠKP odevzdal autor nabyvateli v:
tištěné formě počet exemplářů: 1
elektronické formě počet exemplářů: 2 (1 ve skladu dokumentů, 1 na CD)

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.
3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.
4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

Článek 2
Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizování výpisů, opisů a rozmnoženin.
2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.
3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti:
   - ☒ ihned po uzavření této smlouvy
   - ☐ 1 rok po uzavření této smlouvy
   - ☐ 3 roky po uzavření této smlouvy
   - ☐ 5 let po uzavření této smlouvy
   - ☐ 10 let po uzavření této smlouvy

   (z důvodu utajení v něm obsažených informací)
4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/ 1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

Článek 3
Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.
2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.
3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísni a za nápadně nevýhodných podmínek.
4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: .....

..... Michal Rukola .....

Nabyvatel Autor

Abstrakt

Práca sa zaoberá vývojom aplikácie pre demonštrovanie metód texturovania 3D objektov. Práca obsahuje teoreticky rozobrané jednotlivé metódy textúrovania, ktoré tvoria základ pre návrh obsahu demonštrácie. Požiadavky na aplikáciu sú analyzované a aplikované pre vytvorenie návrhu aplikácie. Postupy pri implementácii jednotlivých metód textúrovania a popis ovládacích prvkov aplikácie sú popísané v závere práce. Program môže slúžiť ako demonštračný nástroj pri výučbe alebo tiež ako študijný materiál pre žiakov.

Klíčová slova

metódy textúrovania, 3D objekt, výukový počítačový program, OpenSceneGraph

Abstract

This work is focused on developing an application for demonstration methods of 3D objects texturing. Work contains theoretical description of particular texturing methods which create base for designing subject of demonstration. Application requirements are analysed and applied to create aplication design. Aproaches used in implementing particular texturing methods and description of application controls are discussed in final part. Application can be used as demonstration tool in lectures or as study material for learners.

Keywords

texturing methods, 3D object, education computer program, OpenSceneGraph

Citace

Michal Kukla: Vyukový program pro demonstraci metod texturování 3D objektů, bakalářská práce, Brno, FIT VUT v Brně, 2008

Vyukový program pro demonstraci metod texturování 3D objektů

Prohlášení

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

.....
Michal Kukla
27. ledna 2008

Poděkování

Rád by som sa poďakoval vedúcemu práce Ing. Přemyslu Krškovi, Ph.D. za ochotu a odborné rady, ktoré mi poskytol pri tvorbe tejto práce.

Obsah

Kapitola 1

Úvod

Problematika textúrovania 3D objektov a s ňou spojené metódy sú rozrastajúcou sa oblasťou súčasnej počítačovej grafiky. Od doby aplikovania prvej textúry na objekt vznikol súbor metód a postupov, ktorý v dnešnej dobe umožňuje renderovať scény s vysokým stupňom fotorealizmu. V súčasnosti nieje výnimkou aplikovanie viacerých druhov textúr súčasne, implementovaných v grafickej karte a použitie shaderov pri textúrovaní. Pre rozvoj problematiky je preto vhodné budúcich vývojárov oboznámiť so základnými princípmi spojenými s textúrovaním. Jednou z alternatív je demonštrácia, ktorá bezprostredne spojí teoretické znalosti a využitie konkrétnej metódy v praxi.

Cieľom mojej bakalárskej práce bolo vytvoriť aplikáciu, ktorá užívateľa zoznámi s metódami textúrovania demonštračnou formou. Samotnému realizovaniu predchádza teoretický rozbor tejto problematiky spracovaný v kapitole 2. Táto poslúžila ako základ pre rozpracovanie obsahu jednotlivých demonštrácii rozobranej v kapitole 3. Východiskom bol návrh implementácie aplikácie rozpracovaný v kapitole 4. Na základe návrhu som implementoval aplikáciu, pričom postupy pri aplikovaní metód textúrovania som diskutoval v kapitole 5. V závere 6 zhŕňam výsledok práce a poukazujem na možné rozšírenia aplikácie.

Kapitola 2

Teoretický rozbor problematiky textúrovania

Kapitola uvádza do problematiky textúrovania 3D objektov, načrtáva význam jednotlivých metód v súvislosti so zobrazovaním, ich prínos pri riešení nedostatkov obrazu, špecifických nežiadúcich efektov, či zrýchlení procesu renderovania scény. Objasňuje dôvody vzniku, bezprostredne súvisiace s problémami, ktoré odstraňujú. Textúrovanie sa však dotýka aj iných oblastí počítačovej grafiky, hlavne osvetlenia scény a reprezentácie 3D objektov. Z tejto oblasti budem čerpať v miere potrebnej na zadefinovanie pojmov týkajúcich sa hlavnej problematiky.

Výsledkom rozboru podstaty a významu jednotlivých metód by mal byť podklad pre výber vhodných postupov demonštrácie. V kapitole si tiež dovolím rozdeliť metódy na základe spoločných znakov a nadväznosti použitých algoritmov a pojmov. Z hľadiska cieľa práce to pokladám za kľúčové postupovať od elementárnych k zložitejším textúrovacím metódam. Témou typy textúr poukazujem na zmenu vlastností povrchu objektu pri ich aplikácii a interakciu so svetelným zdrojom. Textúrovacie súradnice popisujú ich generovacie algoritmy a využitie. V téme Filtrovanie textúr porovnávam druhy filtrovania, výhody pri potláčaní aliasu resp. dopady na rýchlosť renderovania.

2.1 Typy textúr

Kritérium pre rozdelenie textúr na typy je parameter resp. súbor parametrov povrchu, ktoré sa ich aplikáciou zmenia. V súčasnej dobe je trendom používať súbor typov textúr a ich súčasnou aplikáciou dosiahnuť zmenu viacerých parametrov povrchu. Takýmto postupom možno napr. pomocou kombinácie difúznej a bump textúry vytvoriť pomerne komplexný povrch pôsobiaci dojmom množstva polygónov, priaznivým efektom je zníženie výpočtovej náročnosti pri zachovaní detailov povrchu 3D objektu.

2.1.1 Difúzne textúry

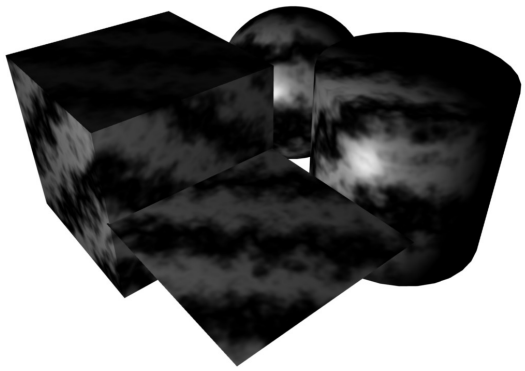
Sú najčastejšie používaným typom textúr. Hodnota bodu telesa závisí od hodnoty difúznej zložky osvetlenia v danom bode, difúznej zložky materiálu telesa a hodnoty texelu, resp. texelov, ktoré sú mapované do daného bodu.

Pri tomto teoretickom rozbore budem vychádzať z Phongovho[8] modelu osvetlenia. Je to empirický model simulujúci osvetlenie materiálu v scéne. Rozlišuje difúznu, ambientnú a spekulárnu zložku svetla. Pri tomto type textúr je z hľadiska výpočtu dôležitá difúzna

zložka osvetlenia. Intenzita zložky v danom bode telesa je závislá na vzájomnej polohe zdroja svetla a bodu telesa. Pri výpočte uvažujeme normálový vektor smerujúci kolmo od povrchu osvetleného telesa v danom bode. Podľa vzťahu [8] 2.1 vypočítame difúznu zložku odrazu.

$$I_d = I_L r_d(\vec{l}.\vec{n}) \tag{2.1}$$

, kde I_L je farebné zloženie dopadajúceho svetla r_d je hodnota difúznej zložky materiálu, vektor $\vec{l}$ je smer dopadajúceho svetla vzhľadom na bod telesa a $\vec{n}$ je normálový vektor v danom bode. Obrázok 2.1 demonštruje dôsledky tohto modelu, pričom na povrchu gule jasne vidieť zmenu intenzity difúznej textúry vzhľadom na smer normálového vektora.



Obr. 2.1: Difúzna textúra

2.1.2 Ambientné textúry

Pri tomto type je hodnota bodu telesa závislá na ambientnej zložke osvetlenia a materiálu telesa.

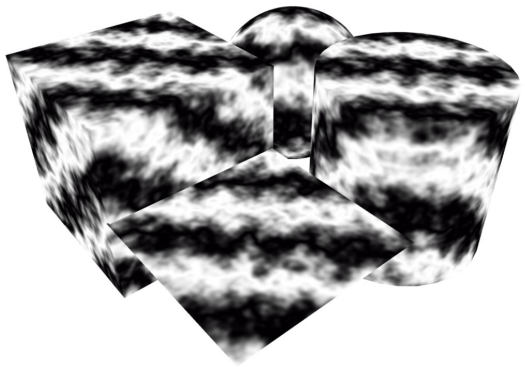
Ambientná zložka svetla simuluje rozptýlené svetlo, pričom jeho intenzita je v priestore konštantná a všesmerová. Pre výpočet hodnoty odrazeného ambientného svetla využijeme vzťah [8] 2.2

$$I_a = I_A r_a \tag{2.2}$$

,kde I_A je ambientá zložka osvetlenia a r_a je ambientná zložka materiálu telesa. Hodnota bodov telesa bude teda závisieť od hodnôt texelov textúry. Obrázok 2.2 demonštruje aplikovanie ambientnej textúry.

2.1.3 Spekulárne textúry

Výpočet hodnoty bodu telesa pri aplikácii spekulárnej textúry závisí od hodnoty spekulárnej zložky svetla, spekulárnych vlastností materiálu telesa (lesklosť, intenzita odlesku), polohy pozorovateľa vzhľadom na bod telesa a hodnoty texelu mapovaného do daného bodu telesa.



Obr. 2.2: Ambientná textúra

Pri spekulárnom osvetlení vzniká na telese charakteristický odlesk, ktorého poloha je závislá od polohy svetla a pozorovateľa vzhľadom na teleso. Pre výpočet hodnoty spekulárneho odlesku v danom bode použijeme vzťah [8] 2.3

$$I_s = I_L r_s (\vec{v}.\vec{r})^h, \tag{2.3}$$

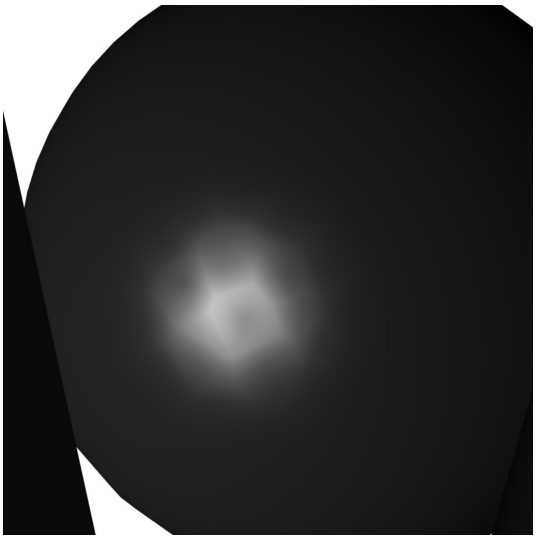
,kde I_L je je farebná charakteristika svetelného zdroja, $\vec{v}$ je jednotkový pohľadový vektor (smer k pozorovateľovi), $\vec{r}$ je jednotkový vektor ideálneho odrazu svetla, ktorý vypočítame podľa vzťahu [8] 2.4, h je hodnota lesklosti - vlastnosti materiálu určujúcej závislosť intenzity spekulárneho odlesku od odklonu vektora pozorovateľa od vektora ideálneho odrazu.

$$\vec{r} = 2(\vec{l}.\vec{n})\vec{n} - \vec{l} \tag{2.4}$$

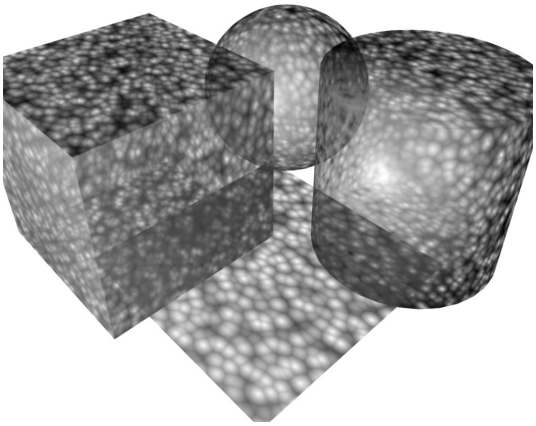
Hodnota bodu telesa s aplikovanou spekulárnou textúrou teda úzko súvisí so spekulárnym odleskom a hodnotou texelu mapovaného do bodu telesa. Demonštruje to obrázok 2.3, kde možno pozorovať zmenu intenzity spekulárneho odlesku na základe mapovanej textúry.

2.1.4 Transparentné textúry

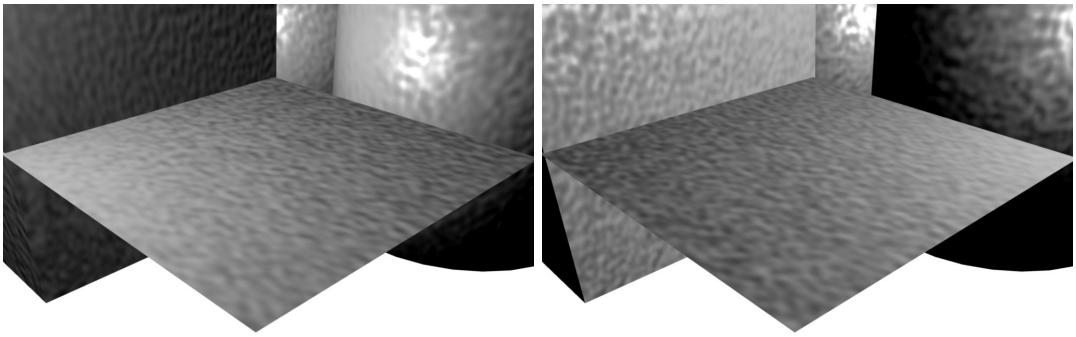
Tento druh textúry ovplyvňuje priehľadnosť materiálu. Vytvára sa tým dojem zmeny geometrie objektu. Priehľadnosť bodu telesa s aplikovanou textúrou je daná hodnotou texelu mapovaného do tohto bodu. Miera priehľadnosti materiálu je obyčajne daná alfa zložkou dát texelov textúry. Pri renderovaní 3D objekov s aplikovanou textúrou je však nutné zabezpečiť aby sa vplyvom optimalizácie zobrazili objekty nachádzajúce sa za transparentnou textúrou. Zvyčajne je to zabezpečené dvojpriechodovým renderovaním scény. Vzhľad objektu s aplikovanou transparentnou textúrou demonštruje obr. 2.4



Obr. 2.3: Spekulárna textúra



Obr. 2.4: Transparentná textúra



Obr. 2.5: Bump textúra

2.1.5 Bump textúry

Tento druh textúr ovplyvňuje normálový vektor v bode na základe hodnoty texelu resp. texelov mapovaných v tomto bode. Týmto postupom sa vytvára dojem hrbolatosti povrchu, pričom v tomto prípade nedochádza k pozmeneniu polohy vertexov 3D objektu, na ktorý je aplikovaná textúra. Výsledkom je povrch pôsobiaci dojmom veľkého počtu polygónov, efekt je však v porovnaní s použitím množstva geometrie výpočtovo menej náročný. Ako zdroj textúry sa v súčasných implementáciách používajú dva druhy textúr. Ide o height map (využíva aj displacement textúrovanie) a normal map.

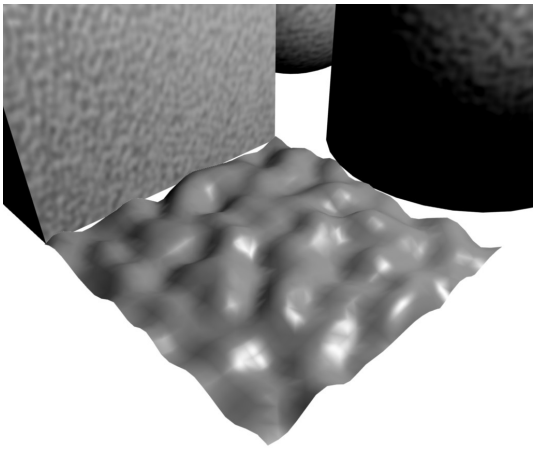
Pri použití height máp ako zdroja textúry je postup výpočtu normálového vektora daného bodu telesa zložitejší. Hodnota height mapy mapovanej v danom bode určuje zdanlivú transláciu povrchu v smere pôvodného normálového vektora. Na základe interpolácie so susednými texelmi sa v danom bode pomocou parciálnej derivácie (v obidvoch osiach textúry) určí nový normálový vektor[8].

Texely normálovej mapy obsahujú priamo hodnotu normálového vektora v danom bode, táto je odvodená od farebných zložiek texelu. Najznámejšou implementáciou tejto metódy je DOT3 bump mapping, ktorý je súčasťou vybavenia väčšiny súčasných grafických kariet. Hodnota difúzneho odrazu je výsledkom tzv. *dot product*, čo je skalárny súčin normálového vektora a vektora smerujúceho k svetlu.

Výsledný bod povrchu je následne osvetlený podľa osvetľovacieho modelu a vlastností materiálu povrchu objektu. Zmenu vzhľadu povrchu objektu pri zmene polohy svetelného zdroja demonštruje obr. 2.5

2.1.6 Displacement textúry

Tento typ textúry na rozdiel od predošlých pozmeňuje polohu vertexov resp. bodov povrchu telesa na základe hodnoty texelu. Využíva sa v systémoch zobrazovania geografických dát pri vizualizácii krajiny na základe údajov o nadmorskej výške povrchu v daných súradniciach. Slúži tiež na tvorbu parametrických plôch, pričom displacement textúra sa generuje procedurálne a aplikuje sa na nedeformovanú plochu. Zmenu geometrie telesa demonštruje



Obr. 2.6: Displacement textúra

obrázok 2.6, na rozdiel od 2.5.

2.1.7 Procedurálne textúry

V súčasnej dobe sa stále viac okrem procedurálnej geometrie používajú aj procedurálne textúry. Jednotlivé hodnoty texelov procedurálnej textúry sú výsledkom určitej procedúry resp. algoritmu. Mapovanie takýchto textúr prebieha buď vygenerovaním do textúry v pamäti, alebo inverzným mapovaním a následným výpočtom na základe zdrojového algoritmu. Ďalšou metódou je renderovanie do textúry. V tomto prípade sa scéna vyrenderuje v prvom prechode a skopíruje do pamäte textúry, tá sa v druhom prechode aplikuje na teleso. Využitie je napr. pri cube mapovaní prostredia aplikovaného na navzájom sa pohybujúce objekty.

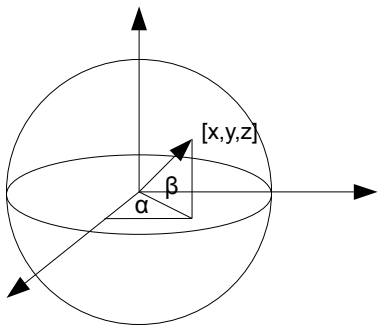
2.1.8 Rozmery textúr

Textúry možno rozdeliť taktiež z hľadiska ich rozmeru. Jednorozmerné textúry, využívané pri imitácii opakujúcich sa vzorov ako aj pri implementácii niektorých prvkov nefotorealistického zobrazovania sú reprezentované vektorom hodnôt texelov. Najviac využívané sú 2D textúry, používané na textúrovanie povrchov 3D objektov. Sú dané tabuľkou, kde je hodnota zodpovedajúceho texelu závislá na 2 textúrovacích súradniciach. 3D textúry resp. objemové textúry sa využívajú pre dosiahnutie efektu tzv. vyrezania telesa z jedného bloku materiálu. Taktiež nájdu uplatnenie pri zobrazovaní objemových dát v medicíne či geografii.

2.2 Mapovanie textúr

V tejto podkapitole rozoberiem metódy mapovania textúry a ich využitie pri priblížení skutočných vlastností materiálov.

Mapovanie textúr je nevyhnutnou súčasťou postupu nanášania textúry na 3D objekt. Je to proces, pri ktorom sa priradia bodu na telese požadované dáta z textúry. V počítačovej



Obr. 2.7: Inverzné sférické mapovanie

grafike sa metódy mapovania textúr často používajú pri generovaní textúrovacích súradníc v objektoch zložených z polygónov. Pri tomto procese sa snažíme nájsť vhodný algoritmus pre mapovanie textúry na teleso požadovaným spôsobom, pričom sa snažíme predchádzať deformáciám.

2.2.1 Generovanie textúrovacích súradníc

Je to časť procesu aplikácie textúry na 3D objekt. V súčasnej počítačovej grafike sa používa metóda inverzného mapovania textúr, pri ktorej používame inverznú mapovaciu funkciu. Funkcia priradí každému bodu telesa zodpovedajúci texel resp. texely v priestore textúry. V praxi sa používajú mapovania na povrch gule, valca, kocky, kvádra a iných štandardných geometrických telies. Ako príklad uvediem mapovanie na plášť gule. V prvom kroku sa pre každý bod telesa vztýči polpriamka s počiatkom v strede telesa. Bod v ktorom polpriamka pretne plášť gule sa odčítajú súradnice textúry v sférických súradniciach gule.

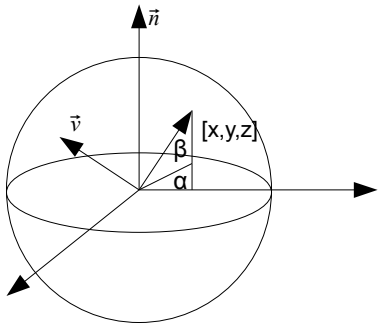
Transformáciu súradníc telesa na súradnice textúry prevádza inverzná mapovacia funkcia. Odčítanie súradníc demonštruje obrázok 2.7. Pre inverzné sférické mapovanie môžeme na základe sférických súradníc a ich prevodu na homogénne priestorové použiť vzťahy [8] 2.6.

$$u = \begin{cases} \frac{1}{2\pi} \arccos \frac{x}{\sqrt{x^2+y^2}} & \text{pre } y \leq 0 \\ 1 - \frac{1}{2\pi} \arccos \frac{x}{\sqrt{x^2+y^2}} & \text{pre } y > 0 \end{cases} \tag{2.5}$$

$$v = 0.5 + \frac{1}{\pi} \arcsin \frac{z}{r} \tag{2.6}$$

2.2.2 Sférické mapovanie prostredia

Táto metóda je špeciálnym druhom mapovania textúry, pričom využíva princíp inverzného mapovania na plášť gule. Pre určenie súradnice textúry sa tu však využíva vektor odrazu od



Obr. 2.8: Sférické mapovanie okolia

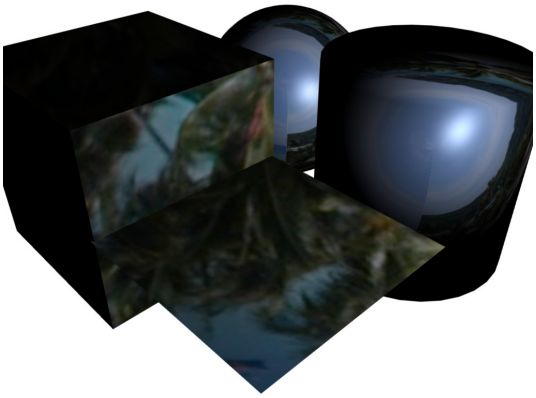
povrchu. V každom bode telesa je pomocou pohľadového vektora a normály vztýčený odrazový vektor. Priesečník odrazového vektora a plášťa gule so stredom v počiatku normály dáva polárne súradnice, ukazujúce do priestoru textúry. Táto metóda sa využíva pri imitácii zrkadlového povrchu a je dobrou aproximáciou skutočného odlesku na objektoch pri neporovnateľne nižšej výpočtovej náročnosti napr. v porovnaní s raytracingom. V polárnych oblastiach však spôsobuje alias, kde celý riadok textúry mapujeme do jedného bodu, čo sa prejaví pri vhodnej polohe pozorovateľa a objektu. Princíp vygenerovania súradnice textúry pri sférickom mapovaní demonštruje obrázok (podĺa [8]) 2.8, pričom $\vec{n}$ je normálový vektor $\vec{v}$ je vektor k pozorovateľovi, výsledná súradnica textúry sa získa zo vzťahu 2.6, pričom vektorom pre odčítanie je odrazový vektor vzhľadom na $\vec{n}$ a $\vec{v}$. Demonštruje to obrázok 2.9 v ktorom možno pozorovať aj vznik aliasu v polárnych oblastiach textúry.

2.2.3 Cube mapovanie prostredia

Využíva princíp inverzného mapovania na plášť kocky a postup získania odrazového vektora je zhodný s predošlou metódou. V procese mapovania je však použitý súbor 6-ich textúr mapovaných na plášť kocky. Pre každú os zhodne po 2 textúry v zápornej a kladnej oblasti. V tomto prípade sa alias v polárnych oblastiach potláča, celkový efekt je dokonalejší a vytvára lepšie priblíženie skutočného odrazu okolia. Vzniká však nedokonalosť v oblastiach spojov textúr, vytvorením ostrých hrán, ktorá sa dá potlačiť vhodnou náväznosťou textúr. Generovanie súradníc pre daný bod telesa demonštruje obrázok 2.10, konkrétnu scénu s využitím cube mapy obrázok 2.10, pre zdôraznenie napojenia textúr je pre každú stenu kocky použitá rovnaká textúra.

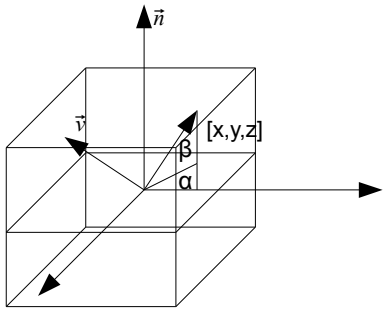
2.3 Filtrovanie textúr

Metódy filtrovania textúr sa využívajú pri inverznom mapovaní textúry na 3D teleso. Vhodnou metódou možno dosiahnuť kompromis medzi výpočtovou náročnosťou a vizuálnou stránkou renderovanej scény



Obr. 2.9: Použitie sférického mapovania okolia

Obr. 2.10: Cube mapovanie okolia



Obr. 2.11: Princíp cube mapovania okolia

Pri inverznom mapovaní textúry takmer vždy vzniká situácia, že výsledkom mapovacej funkcie pre istý bod telesa sú neceločíselné súradnice textúry. Filtrovanie textúr rieši akým spôsobom sa vypočíta hodnota pixelu v bode telesa. Spolu s filtrovaním textúr sa využíva aj technika mipmappingu, riešiaca magnifikáciu a minifikáciu textúry aj v extrémnych prípadoch.

2.3.1 Metóda najbližšieho suseda

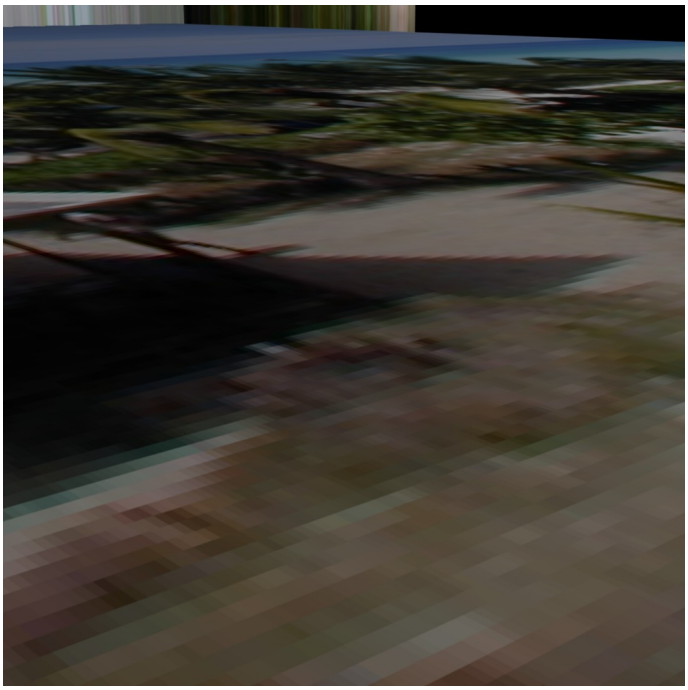
Jej podstatou je aritmetické zaokrúhľovanie súradnice textúry vygenerovanej inverznou mapovacou funkciou. Výhodou je nízka výpočtová náročnosť, ktorá je však pri dnešnom grafickom hardwari zanedbateľná, preto sa používa iba v minimálnom počte prípadov. Pri magnifikácii vytvára alias v podobe ostrých prechodov medzi texelmi textúry. Pri minifikácii vzniká alias v prípade ak je rozmer texelu menší ako je rozmer pixelu obrazovky, prejaví sa to preblikávaním resp. moiré efektom hlavne v textúrach s vysokým frekvenčným rozsahom. Aliasy metódy najbližšieho suseda (*nearest neighbour*) demonštruje obrázok 2.12.

2.3.2 Bilineárne filtrovanie

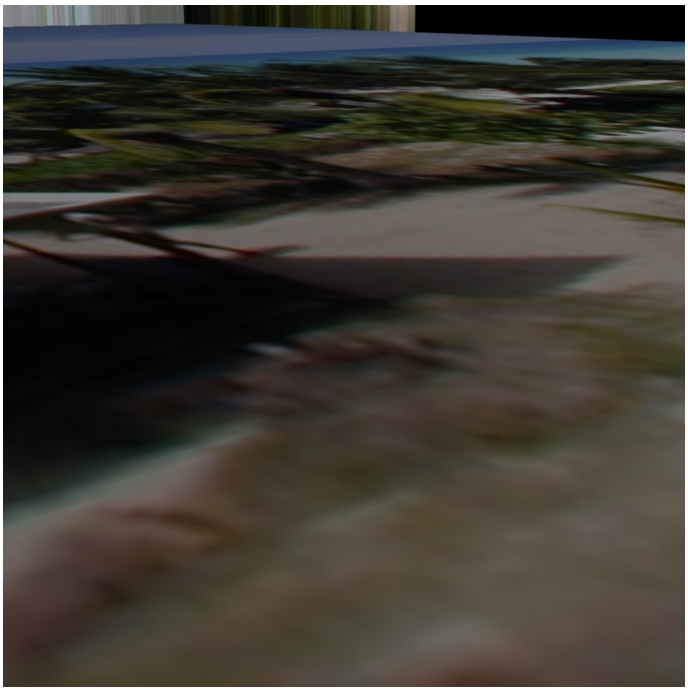
Vizuálne prijateľnejší výsledok poskytuje metóda bilineárneho filtrovania. Pre určenie hodnoty texelu pri neceločíselných súradniciach využíva lineárnu interpoláciu 4 susedných texelov. V obrázku 2.14 môžeme vidieť princíp získania hodnoty požadovaného bodu, kde sa pre každú dvojicu vypočíta hodnota texelov na spojnici jednej osi a následne priemer hodnôt v smere druhej osi textúry. Bilineárne filtrovanie potláča alias, avšak pri extrémnom zväčšení spôsobuje rozmazanie textúry, pri zmenšení textúry vzniká alias zhodný s predošlou metódou. Aplikáciu demonštruje obrázok 2.13.

2.3.3 Mipmapping

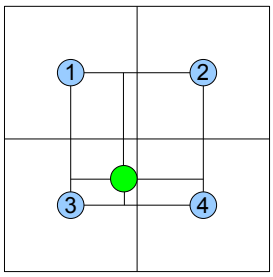
Z latinského *multum in parvo* pochádza skratka MIP (v preklade mnoho v malom). Je metóda, ktorá rieši aliasing textúr pri extrémnom zmenšení a znižuje výpočtovú náročnosť



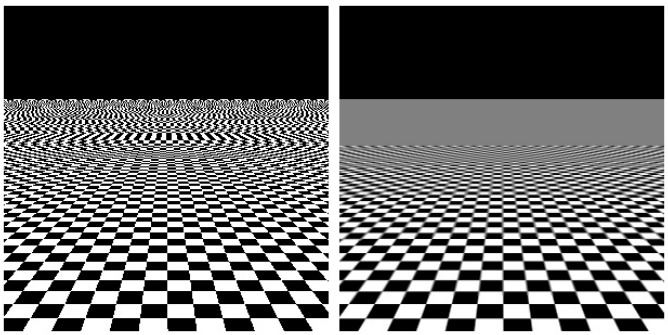
Obr. 2.12: Metóda najbližšieho suseda



Obr. 2.13: Bilineárne filtrovanie



Obr. 2.14: Princíp bilineárneho filtrovania



Obr. 2.15: Využitie mipmappingu

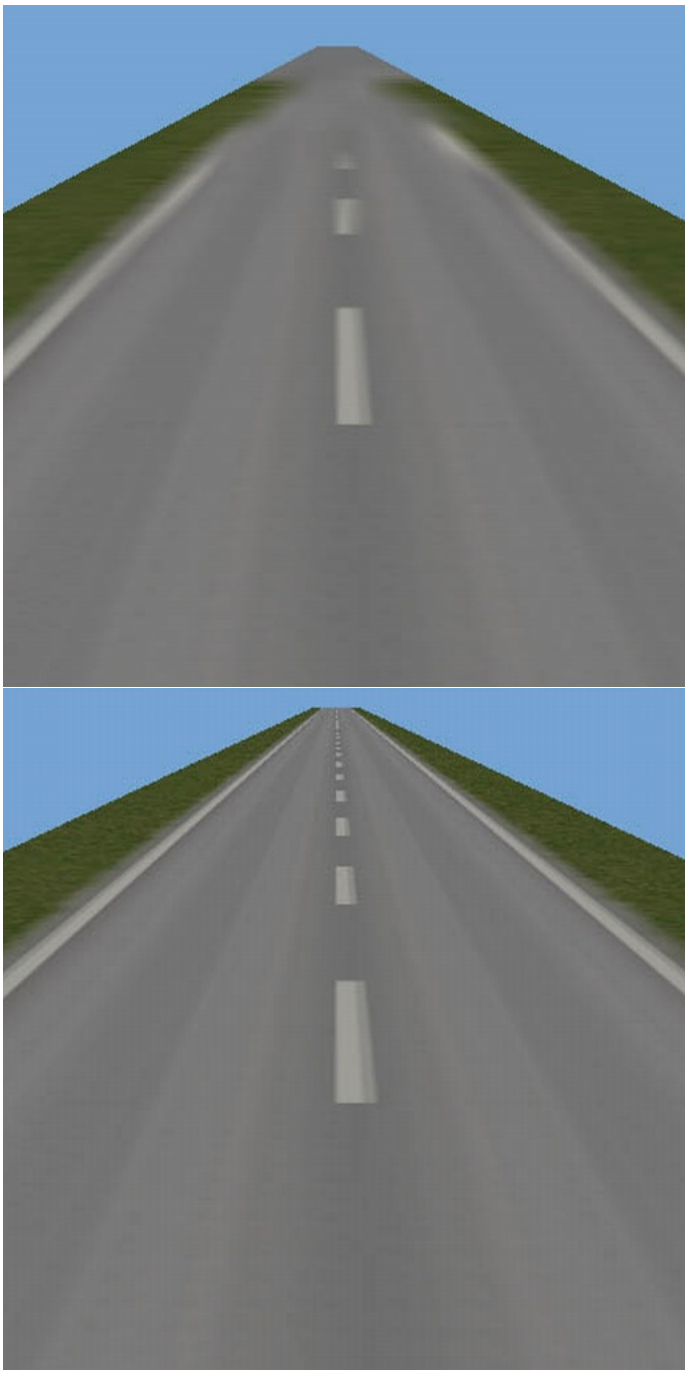
renderovania scény. Metóda spočíva v súčasnom uložení textúry v zmenšených verziách v pamäti. Použitie jednotlivých úrovní textúry záleží na metóde filtrovania textúry, tzv. biasu, ktorý určuje úroveň mipmapy a vzdialenosti pixelu od pozorovateľa. V spojení z anizotropickým filtrovaním sa používa metóda pri ktorej sa tvoria mipmapy s iným pomerom strán ako zdrojová textúra. Potlačenie aliasu pri využití tejto metódy demonštruje obr 2.15

2.3.4 Trilineárne filtrovanie

Trilineárne filtrovanie úzko súvisí s metódou mipmappingu, podobne ako bilineárne filtrovanie používa lineárnu interpoláciu okolia 4 texelov textúry. Do úvahy sa však berie aj aplikácia mipmáp, pričom výsledná hodnota texelu sa vypočíta z 2 mipmáp lineárnou interpoláciou. Táto metóda rieši vznik ostrých prechodov medzi úrovňami mipmapy, je však oproti metóde bilineárneho filtrovania výpočtovo náročnejšia.

2.3.5 Anizotropické filtrovanie

Anizotropické filtrovanie je špeciálnym druhom trilineárneho filtrovania, pri ktorom sa rieši problém rozmazania textúr, ktoré sú sklonené v určitom uhle vzhľadom na pozorovateľa, ktorý vytvára anizotropiu v súvislosti s nerovnomernou veľkosťou osí súradníc textúry. Zatiaľ čo pri trilineárnom filtrovaní sú pre takto sklonenú textúru v určitom bode generované 2 mipmapy (izotropické), pri anizotropickom filtrovaní sa generujú mipmapy neizotropické vo veľkostiach, ktoré záležia od stupňa anizotropie. Pri filtrovaní sa potom používajú v bode telesa
mipmapy podľa sklonu polygónu voči pozorovateľovi [4]. Anizotropické filtrovanie poskytuje najuspokojivejšie výsledky v scénach s množstvom sklonených povrchov, na druhej strane je výpočet hodnoty pixelu náročnou operáciou, výpočtovo aj pamäťovo. Porovnanie trilineárneho a anizotropického filtrovania vyššieho stupňa je v obr. 2.16



Obr. 2.16: Porovnanie trilineárneho a anizotropického filtrovania

Kapitola 3

Rozbor koncepcie výukového demonštračného programu pre textúrovanie

Kapitola pojednáva o cieľoch aplikácie v súvislosti s demonštrovaním metód využívaných pri textúrovaní 3D objektov. Cieľom aplikácie má byť interaktívnou formou zoznámiť užívateľa s danou problematikou. Najdôležitejším prvkom demonštrácie je vizualizácia využitia metód, ktorá je v tomto konkrétnom prípade nevyhnutnou súčasťou aplikácie. Metódy textúrovania sú často krát kombinované, resp. sa vždy využívajú súčasne (napr. mipmapping s filtrovaním textúr). Z tohto dôvodu som obsah demonštrácie rozdelil do súvisiacich celkov, pričom som kládol dôraz na to aby sa pokročilejšie metódy demonštrovali po pochopení základov textúrovania a reprezentácie 3D objektov, ktoré sú nevyhnutné pre postup demonštrácie.

Pre podporu výuky zvolenej metódy využijem vhodné prvky nefotorealistického a názorného zobrazovania 3D grafiky (kóty, súradnicové osy) v spojení s objektami s aplikovanou textúrovacou metódou. Nemenej dôležitou súčasťou je interakcia užívateľa s aplikáciou, navrhnutá tak aby umožnila zmenou presne daných a ohraničených parametrov (vertexy objektu, k nim prislúchajúce textúrovacie súradnice a.i.) scény demonštrovať podstatu metódy, jej interakciu s 3D objektom, pozorovateľom a svetelným zdrojom. Každá metóda je rozpracovaná na viacerých scénach, pre uvedenie užívateľa do problematiky, pre názorné porovnanie objektov bez aplikácie metódy a pre umožnenie zmeny parametrov scény, ktoré bezprostredne súvisia s demonštrovanou metódou.

Súčasťou aplikácie bude aj teoretický základ metód, ktorý poslúži pre lepšie pochopenie významu danej problematiky.

3.1 Využitie nefotorealistického zobrazovania pri demonštrácii

Pre podporu demonštrácie a zlepšenie pochopenia tematiky textúrovania je použitie nefotorealistického zobrazovania žiaduce. Dôležitým faktorom je výrazné odlíšenie objektov, resp. metód tohto zobrazovania od geometrie na ktorú sa aplikuje demonštrovaná metóda. Spolupráca tohto zobrazovania s demonštračnou scénou by mala byť zásadným spôsobom odlíšená.

Pre podporu výuky je vhodné použitie už zaužívaných prvkov, ktoré sa využívajú v diagramoch resp. v názorných ukážkach v literatúre a v multimediálnych prezentáciách. V konkrétnom prípade, pri metódach textúrovania a s nimi spojených oblastiach počítačovej

grafiky je pre pochopenie problematiky výhodné popisovať 3D objekty, zobrazovať súradnicové osi priestoru a textúry. Zobrazovať drôtené (wireframe) modely telies.

Pre uvedenie do problematiky osvetlenia je výhodné zobraziť vektory významné pri aplikácii osvetľovacieho modelu, v konkrétnej časti demonštrácie typu textúr, ktoré priamo súvisia s osvetlením scény. Vhodné je aj doplnenie demonštrácie o animáciu prvkov nefotorealistického zobrazovania. Pri vhodnom aranžovaní do scény spolu s pohybujúcim sa svetelným zdrojom napomôžu pri pochopení demonštrovanej metódy.

Dôležitou súčasťou je aj použitie názorných textúr, pre zdôraznenie aspektov demonštrovanej metódy, či umelé vyvolanie nežiadúceho efektu. Tieto textúry možno využiť aj v diagramoch pre demonštráciu generovania textúrovacích súradníc vo vhodnej kombinácii s vektormi. Užívateľ v konkrétnom prípade pochopí spojitosť medzi priesečníkom vektora vo sférických súradniciach s názornou textúrou aplikovanou na objekt a texelom textúry, ktorú mapuje.

3.2 Výber objektov a textúr pre demonštráciu

Ako základ pre výber objektov do konkrétnej demonštrácie poslúži dôvod resp. účel vzniku konkrétnej metódy textúrovania.

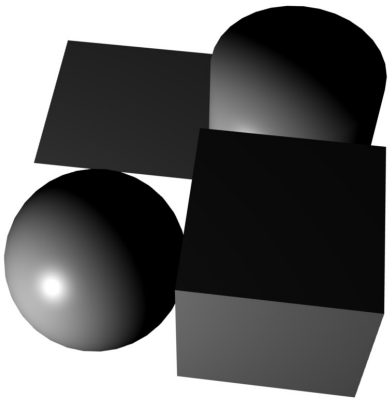
Pre potrebu demonštrácie je využitie zložitých modelov z množstvom polygónov z demonštračného hľadiska nevýhodné. V konkrétnych prípadoch je vhodné použitie procedurálnej geometrie, základných geometrických útvarov pre potreby demonštrácie konkrétnej metódy.

Na základe súboru metód, ktoré budem demonštrovať som vybral geometrické útvary vhodné pre aplikáciu textúr, generovanie textúrovacích súradníc resp. filtrovania. Dôležitým kritériom je spojitosť povahy povrchu 3D objektu (zakrivenie, spojitosť) s využitím metódy ktorá sa na teleso bude aplikovať. Pre potreby demonštrácie využijem nasledovné geometrické útvary a ich drôtenú alternatívu pre aplikáciu metód v obr 3.1 v poradí zľava zhora:

- Grid - Ohraničená rovina poslúži pri demonštrácii metód v ktorých je potrebné objasniť význam osvetlenia, resp. normálového vektora a pri nefotorealistickom zobrazovaní.
- Cylinder - Poslúži v znázornení generovania textúrovacích súradniciach na plášť cylindra.
- Sféra - Využijem pri demonštrácii generovania textúrovacích súradníc na plášť gule resp. ako demonštrácia vzniku aliasu pri mapovaní prostredia.
- Kocka - Aplikovaním názornej textúry objasním cube mapovanie, resp. poslúži ako demonštrácia pre typy textúr.

3.3 Demonštrácia typov textúr

Cieľom tejto časti demonštrácie textúrovania je oboznámiť užívateľa s typmi textúr používaných v súčasnej počítačovej grafike, demonštrovať mu interakcie textúr so zložkami materiálu a svetelného zdroja. Užívateľ by mal byť po absolvovaní demonštrácie schopný popísať typy textúr, ich interakciu so svetelným zdrojom, či geometriou objektu, popísať rozdiely medzi nimi a identifikovať ich.



Obr. 3.1: 3D objekty v demonštrácii

3.3.1 Difúzne textúry

Demonštračné témy rozdelím na viacero častí, v ktorých postupne uvediem užívateľa do problematiky, schematickou ukážkou princípu metódy v spojitosti s osvetľovacím modelom. V prvej časti zobrazím grid, znázorním normálový vektor s adekvátnym označením. Scéna bude obsahovať animáciu zmeny polohy svetelného zdroja s označením polohy a popisom. Ďalej zobrazím vektor smerujúci k svetlu so zodpovedajúcou animáciou. Scéna bude interaktívna v zmysle zmeny polohy pozorovateľa. Týmto demonštrujem zmenu intenzity osvetlenia gridu zmenou polohy svetla vzhľadom na normálu povrchu. V druhej časti zobrazím scénu podobným spôsobom, užívateľ bude mať možnosť vybrať textúru z vopred pripravenej palety. Animáciou polohy svetelného zdroja ozrejmím interakciu medzi difúznou zložkou svetla a zmenou vzájomnej polohy prvkov demonštrácie.

3.3.2 Ambientné textúry

V tejto demonštrácii nadviažem na predošlú a podobným spôsobom (užívateľovi už známym) vysvetlím význam aplikovania ambientnej textúry na teleso v scéne osvetlenej svetelným zdrojom. V prvej časti zobrazím podobne ako v predošlej demonštrácii animáciu svetelného zdroja a grid s normálovým vektorom. Zmenou polohy svetla sa osvetlenie gridu nezmení, čím poukážem na vlastnosti ambientného osvetlenia. V druhej časti bude mať užívateľ možnosť porovnať predošlý typ textúry so súčasným, pri jednom animovanom svetelnom zdroji. Objasním tak rozdiel medzi povahou ambientnej a difúznej textúry v spojení so svetelným zdrojom. Užívateľ bude mať možnosť meniť textúru nezávisle na oboch gridoch.

3.3.3 Spekulárne textúry

Cieľom demonštrácie je oboznámiť užívateľa so spekulárnou zložkou osvetlenia a interakciou so spekulárnou textúrou, ktorá ovplyvňuje vlastnosti tohto efektu. V prvej časti zobrazím grid s normálovým vektorom, vektorom smerujúcim k svetlu a vektorom ideálneho odrazu. Demonštrovať budem vznik spekulárneho odrazu na telese v momente smerovania odra-

zového vektora k pozorovateľovi. V druhej časti dám užívateľovi možnosť porovnať predošlé typy textúr so spekulárnou textúrou, pričom použijem animovaný svetelný zdroj meniaci polohu vzhľadom na gridy. Týmto uľahčím pochopenie rozdielov medzi typmi textúr a ich náväznosť pri použití svetelného zdroja.

3.3.4 Transparentné textúry

Transparentnosť textúry budem demonštrovať v prvej časti pomocou gridu s aplikovanou difúznou textúrou. Medzi pozorovateľom (umiestneným nad gridom) umiestnim grid s aplikovanou transparentnou textúrou, ktorý zmenou polohy bude demonštrovať transparenciu. V druhej časti budem animovať objekty cylinder, sféru a kocku nad gridom s difúznou textúrou. Užívateľ bude mať možnosť výberu z palety textúr, obsahujúcej textúry s transparenciou v alfa kanále a textúry ktoré alfa kanál neobsahujú, samostatne pre každý objekt.

3.3.5 Bump textúry

Demonštrácia bude obsahovať grid s aplikovanou difúznou a normálovou mapou, pričom bude doplnená animáciou translácie polohy svetelného zdroja. Užívateľ bude mať možnosť zmeniť bump textúru z pripravenej palety. Demonštrovať budem zmenu vzhľadu povrchu na základe polohy svetla bez zmeny polohy vertexov objektu.

3.3.6 Displacement textúry

Dôležite v tejto demonštrácii je nadviazať na predošlú metódu a ozrejmiť užívateľovi, že displacement textúra mení geometriu povrchu. V prvej časti demonštrácie zobrazím objekt s dvojicou identických textúr mapovaných ako difúznu a displacement textúru. Animáciou rotácie vo vhodnej osi bude mať užívateľ možnosť pozorovať v profile gridu zmenu geometrie povrchu resp. súvis difúznej a displacement textúry na základe farieb texelov difúznej textúry. V druhej časti môže užívateľ meniť difúznu ako aj displacement textúru na gride, otáčať demonštračnou scénou.

3.3.7 Procedurálne textúry

V tejto demonštrácii umožním užívateľovi aktiváciou ovládacieho prvku vygenerovanie a následnú aplikáciu textúry na teleso. Generovať sa bude textúra šumu a gradientu. V demonštrácii zobrazím 3 identické telesá, na ktorých sa bude dať aktiváciou ovládacieho prvku vygenerovať textúra na základe parametra resp. vybrať textúru z palety. Demonštrovať sa tu bude variabilita procedurálnych textúr bez nutnosti externého zdroja.

3.3.8 Rozmery textúr

Hlavným cieľom demonštrácie je ukázať vizuálne rozdiely medzi telesami s aplikovanou textúrou, ktorá ma daný rozmer. V demonštrácii zobrazím 3 objekty typu kocka s aplikovanou textúrou konkrétneho rozmeru. Textúrovacie súradnice 3D textúry budem na telese animovať vo všetkých osiach, čím zdôrazním efekt vyrezania objektu z jedného kusu materiálu.

3.4 Demonštrácia mapovania textúr

Cieľom tejto kapitoly je oboznámiť užívateľa s problematikou textúrovacích súradníc, ich generovania, využitia pri špeciálnych metódach mapovania textúry, hlavne pri sférickom mapovaní prostredia a pri cube mapovaní prostredia. Užívateľovi budú demonštrované telesá s textúrovacími súradnicami vygenerovanými rôznymi spôsobmi, pričom budú aplikované difúzne textúry resp. budú zobrazené difúzne textúry kombinované s reflexnými textúrami. Pri mapovaní prostredia je dôležitým prvkom zmena smeru pohľadu a polohy pozorovateľa v scéne, pri ktorom dôjde k vizuálnej zmene povrchu telesa, ktorá simuluje odraz prostredia. Ako pomocné objekty využijeme názorné zobrazenie normál telesa a k nim prislúchajúcim odrazovým vektorom vzhľadom na polohu telesa. Užívateľ bude mať možnosť zmeniť aplikovanú textúru výberom z palety.

3.4.1 Generovanie textúrovacích súradníc

V prvej časti demonštrácie ozrejmím užívateľovi mapovanie textúr na plášť gule. Použijem schému podobnú obr. 2.8. Pričom ju prevediem do 3D priestoru za pomoci wireframe modelu sféry, gridu rovnobežného s plochou danou súradnicami XY priestoru. Do stredu sféry umiestnim vektor, ktorého rotáciu okolo počiatku budem animovať. Vektor bude pretínať wireframe sféru v určitých bodoch. Ako druhý objekt zobrazím grid na ktorom sa bude pohybovať ukazovateľ. Tretím objektom bude sféra so súradnicami mapovanými inverzným sférickým mapovaním. Užívateľ zistí spojitosť medzi priesečníkom vektora s wireframe sférou, ukazovateľom do textúry a textúrou namapovanou na sféru.

3.4.2 Sférické mapovanie prostredia

Pri tejto demonštrácii nadviažem na inverzné sférické mapovanie textúry na teleso pričom do demonštrácie pridám vektor pozorovateľa a vektor odrazu, vhodnou kombináciou animovanej rotácie ozrejmím vznik odrazového vektora a jeho význam pri generovaní textúrovacích súradníc. V druhej časti zobrazím viacero objektov, ktorých súradnice budú generované práve touto metódou. Zmenou polohy pozorovateľa užívateľ zistí zmenu mapovania textúry, pričom bude možné vybrať zodpovedajúcu textúru z palety.

3.4.3 Cube mapovanie prostredia

Demonštráciou nadviažem na predošlú tému a spôsob prezentácie metódy, pričom zdôrazním rozdiel a výhody cube mapovania prostredia (deformácia na póloch). Pre zvýšenie názornosti využijem textúru tvorenú symbolmi cube máp, odlíšených farbami. V prvej časti nadviažem na sférické mapovanie a zobrazím vertexy telesa spolu s normálami, odrazové vektory, pričom v tomto prípade bude wireframe sféra nahradená kockou, v ktorej strede bude grid. Miesto priesečníka odrazového vektora a steny kocky je súradnica textúry v danom vertexe. V ďalšej časti bude mať užívateľ možnosť zmeniť textúru z palety názornej textúry a textúry vytvorenej z fotografií reálneho prostredia. Užívateľ bude mať možnosť meniť polohu pozorovateľa scény, sledovať aliasy resp. výhody cube mapovania.

3.5 Demonštrácia filtrovania textúr

V tejto skupine má aplikácia demonštrovať použitie metód filtrovania textúr. Pre lepšie pochopenie tejto problematiky poukážem na dôvod vzniku filtrovania textúr, ktorým je

alias pri priblížení resp. vzdialení objektu, keď sa viacero texelov mapuje do pixelu resp. rozmazanie obrazu pri priblížení. Pre rozlíšenie použitej metódy musí byť užívateľovi prezentované vybrané objekty s pomocnými textúrami, pričom vzájomná poloha pozorovateľa a objektu má vytvárať alias resp. ho potláčať pri zvolenej metóde filtrovania. Pri tejto skupine demonštrácií využijem animácie objektov pre navodenie prípadu vzniku aliasu, pričom pri porovnaní metód zhodnou animáciou ozrejmím prípadné rozdiely v potláčaní aliasu

3.5.1 Porovnanie metód filtrovania textúr

V tejto demonštrácii porovnám metódu najbližšieho suseda s metódou bilineárneho filtrovania. Použijem gridy, na ktoré aplikujem zhodnú textúru. Vhodnou animáciou približovania resp. vzďaľovania umožním používateľovi porovnať zvolené metódy z hľadiska vzniku aliasu na textúre vplyvom zmenšenia a zväčšenia. V druhej časti zobrazím telesá, pri ktorých bude mať užívateľ možnosť zmeny textúry. Pohybom kamery môže vyvolať aliasy a tak lepšie porozumieť problematike.

3.5.2 Mipmapping

V tejto demonštrácie je dôležité poukázať na aliasy, ktoré vznikajú bez použitia mipmappingu, hlavne pri minifikácii textúry. V prvej časti porovnám aplikovanie a neaplikovanie metódy na zhodné gridy pričom použijem bilineárne filtrovanie, kde mipmapy budú aplikované na základe metódy najbližšieho suseda. V druhej časti porovnám bilineárny a trilineárny mipmapping animáciou meniacou vzdialenosť gridov a polohy pozorovateľa. V tretej časti dám užívateľovi možnosť meniť uhol pohľadu na scénu, pričom bude môcť meniť textúry a porovnávať aliasy.

3.5.3 Anizotropické filtrovanie textúr

Dôležitým cieľom tejto demonštrácie je ukázať užívateľovi rozdiel v zobrazení textúry s rôznym stupňom anizotropie a porovnanie s predošlými metódami. V prvej časti zobrazím 3 gridy, ktorých textúry bude tvoriť text s adekvátnym názvom filtrovacej metódy. Vhodnou animáciou dosiahnem zmenu uhla pozorovania a roviny povrchu objektov, čím docielim aplikáciu mipmáp s menším rozlíšením. Kritériom pre porovnávanie bude čitateľnosť textu v textúrach. V druhej časti bude mať užívateľ na zhodnej scéne možnosť zmeny textúry z palety individuálne pre každý objekt a zmenu úrovne anizotropie.

3.6 Požiadavky na demonštračnú aplikáciu

Kapitola obsahuje súhrn požiadaviek na program vyplývajúcich z teoretického rozboru a koncepcie výukového programu, ako aj špecifickosti konkrétnych metód.

3.6.1 Ovládanie aplikácie

Pre výukový program je rozmiestnenie a celková intuitívnosť ovládania nevyhnutnou súčasťou aplikácie. Je dôležité nezaťažovať užívateľa zložitým manuálom a nezrozumiteľným rozhraním. Aplikácia by tým stratila demonštračnú hodnotu. Výsledkom by bolo odradenie užívateľa od používania. Pre potreby tohto konkrétneho zadania je dobré s ohľadom na obsah tejto kapitoly určiť množinu parametrov scény, ktoré bude mať možnosť užívateľ určitým spôsobom zmeniť. Preto po analýze obsahu demonštrácií potrebujeme zabezpečiť:

- Voľbu metódy ktorá sa bude demonštrovať
- Zobrazenie súvisiacej teórie
- Zobrazenie nápovede
- Navigáciu v demonštrácii
- Voľbu objektu, na ktorom budeme prevádzať zmeny textúry
- Zmenu jednotlivých druhov textúr na zvolenom telese
- Zmenu úrovne anizotropie
- Zmenu sady cube textúr mapovaných na teleso
- Zmena polohy a smeru pohľadu v scéne

3.6.2 Zobrazovanie scény

Kľúčovou požiadavkou je však zobrazenie demonštrácií. V konkrétnom prípade sú na zobrazovanie kladené nároky na jednotnosť zobrazenia efektu pokiaľ možno na každej inštalácii. Dôležitým prvkom je aj zobrazovanie v reálnom čase, pre potrebu animácií a interakcie objektov a textúr. Z hľadiska intuitívnosti ovládania je dobre dodržať zaužívané postupy pri zmene polohy kamery a smeru kamery pomocou myši resp. iného polohovacieho zariadenia. Konzistentnosť ovládania pohľadu je vo všetkých scénach nevyhnutná, opak by viedol užívateľa k opätovnému študovaniu manuálu. Na zobrazovanie preto kladiem tieto požiadavky:

- Zmena scény pri zvolení metódy a jej jednotlivých častí
- Zmena parametrov objektov v reálnom čase
- Zmena polohy a smeru pozorovateľa
- Uvedenie do východiskovej polohy
- Podpora zobrazovania v reálnom čase s podporou zvolených metód

Kapitola 4

Analýza požiadaviek a návrh aplikácie

V tejto kapitole odôvodním zvolené vývojové prostriedky, načrtnem spojitosť medzi požiadavkami a objektovým návrhom aplikácie, pričom charakterizujem triedy z hľadiska účelu v aplikácii.

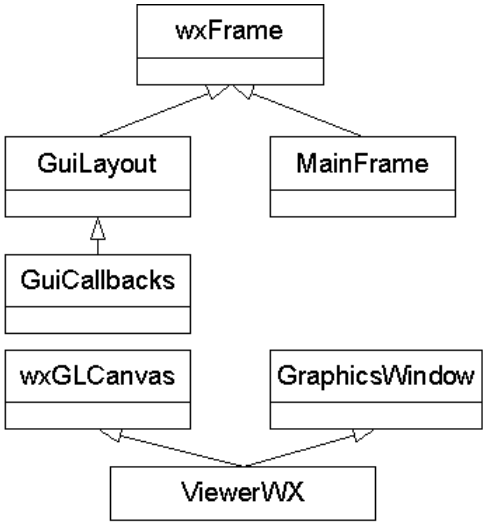
4.1 Definovanie vývojových prostriedkov a platformy

Aplikáciu som sa rozhodol implementovať v objektovo orientovanom jazyku C++, ktorý sa hodí pre rozsiahlejšie aplikácie doplnené užívateľským rozhraním. S výhodou som využil princíp dedičnosti pri návrhu tried.

Ako už z požiadaviek vyplýva, aplikácia nevyhnutne vyžaduje GUI. Z dostupných nástrojov pre programovanie užívateľského rozhrania som zvolil API wxWidgets [7], ktoré umožňuje tvorbu GUI, pričom je platformovo nezávislé. Umožňuje transparentný prístup k tvorbe ovládacích prvkov, ktoré narozdiel od iných nástrojov neemuluje, ale používa natívne, prístupné na danej platforme. V aplikácii som rozdelil rozmiestnenie a funkcionalitu ovládacích prvkov do samostatných tried. Pre návrh vzhľadu som využil aplikáciu wxFormBuilder [6], ktorá poskytuje wysiwyg editor rozhrania a deklarovanie funkcií pre obsluhu udalostí generovaných ovládacími prvkami.

Na základe požiadaviek pre zobrazovanie scény a prácu s 3D objektami som sa rozhodol pre vizualizáciu použiť knižnicu OpenSceneGraph[3](OSG). Je to rozhranie ktoré zapuzdruje grafické API OpenGL. Obsahuje množstvo užitočných metód a tried, ktoré spĺňa požiadavky aplikácie a uľahčia samotnú implementáciu jednotlivých metód. Dôležitým faktorom pre rozhodnutie bola aj platformová nezávislosť a možnosť spolupráce s API pre tvorbu GUI. OSG obsahuje nástroje pre integráciu grafického kontextu OpenGL do okna aplikácie. Zmenu parametrov scény v reálnom čase, možnosti animovania a dokonca základné možnosti tvorby GUI priamo v grafickom okne. Veľkou výhodou je aj transparentná práca s rozšíreniami OpenGL[2].

Platformou aplikácie je operačný systém Windows XP. Ako prostredie pre vývoj slúžilo MS Visual Studio 2005, pre preklad programu do binárnych súborov bol použitý prekladač, ktorý je súčasťou tohto IDE.



Obr. 4.1: Diagram dedičnosti tried GUI

4.2 Návrh aplikácie

V tejto podkapitole nadviažem na požiadavky a spojím ich so samotnou implementáciou, ktorej predchádza návrh programu. V tomto prípade vytvorím objektovo orientovaný návrh, v ktorom naznačím dedičnosť tried. Komunikáciu medzi nimi na základe požiadaviek a ich charakteristiku objasním v podkapitolách 4.3. Objektový návrh som z hľadiska odlišnosti jednotlivých častí rozdelil do kategórii tried implementujúcich GUI, tried reprezentujúcich objekty scény a triedy implementujúce samotné demonštrácie.

4.2.1 GUI

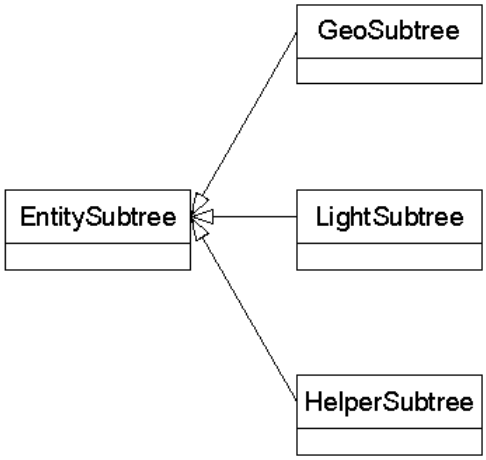
V tejto časti rozoberiem oblasť aplikácie zodpovednej za obsluhu užívateľského rozhrania. Na základe 3.6.1 som navrhol hierarchiu tried na obr. 4.1. Význam oddelenia vzhľadu od funkcie demonštruje rodičovský vzťah tried `GuiLayouts` a `GuiCallbacks`, pričom pri dedení sa predáva vzhľad a špecializujú sa funkcie pre ošetrenie udalostí ovládacích prvkov. Trieda `ViewerWX` implementuje grafické okno a jedná sa o viacnásobnú dedičnosť. GUI je odvodené ako z tried wxWidgets tak aj z OSG.

4.2.2 Objekty scény

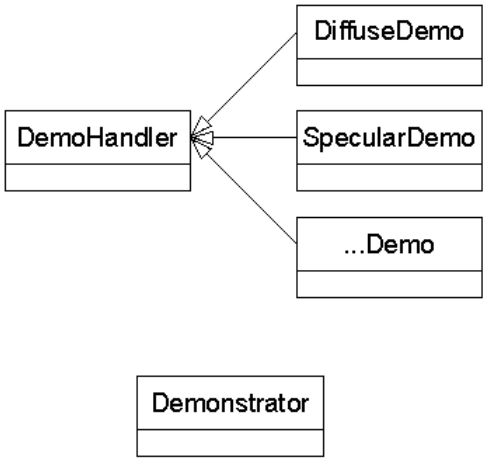
Návrh reprezentácie objektov odráža ich špecifickosť vzhľadom na vnútornú štruktúru. Ich predok `EntitySubtree` však disponuje spoločnými vlastnosťami napr. zmeny parametrov translácie, definícia animácie či pridanie textového popisu objektu. Diagram dedičnosti je na obr. 4.2.

4.2.3 Demonštrácie

Špecifickosť jednotlivých metód textúrovania evokovala hierarchiu tried implementujúcich jednotlivé demonštrácie (obr. 4.3). Obsah demonštrácie je špecifikovaný v každom potomkovi (napr. `DiffuseDemo`), pridávanie objektov je však generalizované v základnej triede



Obr. 4.2: Diagram dedičnosti tried objektov v scéne



Obr. 4.3: Diagram dedičnosti tried demonštrácií

DemoHandler.

4.3 Charakteristika tried

4.3.1 Trieda Demonstrator

Jej inštancia sa vytvára pri inicializácii aplikácie. Poskytuje zoznam ukazovateľov na jednotlivé inštancie tried zdedených z DemoHandler, pričom obsahuje metódy na získanie ukazovateľa podľa poradia demonštrácie. Ukazovateľ na inštanciu tejto triedy je priradený do verejného atribútu triedy MainFrame. Trieda sa používa pri zmene parametrov objektov pomocou predania ukazovateľa na inštanciu potomkov DemoHandler a následného prehľadania zoznamu interaktívnych objektov na základe voľby v GUI.

4.3.2 Trieda MainFrame

Implementuje hlavné okno demonštrácie. Obsahuje ukazovateľ na inštanciu triedy Demonstrator, volaním jej metód a prístupom k inštanciám tried zdedených z DemoHandler pripája podstrom časti demonštrácie do ViewerWX, na základe voľby položky v menu. Pri zmene demonštračnej scény volá metódy deinicializácie a inicializácie inštancií zdedených z DemoHandler. Obsahuje ukazovateľ na inštanciu GuiCallbacks, v ktorom mení aktiváciu ovládacích prvkov na základe konkrétnej demonštrácie.

4.3.3 Trieda DemoHandler

Je základnou triedou všetkých implementácií demonštrácií. Definuje metódy pre deinicializáciu konkrétnej inštancie triedy od nej zdedenej. Obsahuje zoznam zoznamov inštancií tried vytvorených z GeoSubtree,LightSubtree,HelperSubtree. Ďalej obsahuje zoznam koreňových uzlov stromu scény pre každú časť demonštrácie a zoznam ciest k súborom textúr. Implementuje metódy pre vkladanie objektov do scény, posun na ďalšiu resp. na predošlú časť demonštrácie. Pre úplnosť dodávam zoznam tried a demonštrácií ktoré som implementoval:

- DiffuseDemo - Difúzne textúry
- AmbientDemo - Ambientné textúry
- SpecularDemo - Spekulárne textúry
- TransparentDemo - Transparentné textúry
- BumpDemo Bump - textúry
- DisplaceDemo - Displacement textúry
- ProceduralDemo - Procedurálne textúry
- DiamDemo - Porovnanie rozmerov textúr
- FMethodsDemo - Metódy filtrovania textúr
- MipMapDemo - Mipmapping
- AnisoDemo - Anisotrofické filtrovanie

- `TexgenDemo` - Generovanie textúrovacích súradníc
- `SpheremapDemo` - Sférické mapovanie prostredia
- `CubemapDemo` - Cube mapovanie prostredia

4.3.4 Trieda `EntitySubtree`

Táto trieda implementuje metódy spoločné pre všetky objekty, obsahuje tú časť podstromu objektu, ktorá je spoločná pre všetky objekty. Implementuje taktiež metódy pre vytvorenie animácie, vkladanie textového popisu objektu do podstromu na základe rozmerov objektu. Obsahuje tiež metódy pre povolenie translácie a rotácie objektu v scéne.

4.3.5 Trieda `GeoSubtree`

Implementuje geometrické objekty a ich textúrovanie. Obsahuje metódy pre načítanie textúr a ich následnú aplikáciu na objekt. Jej inštancie sú pridávané pri inicializácii každej triedy implementujúcej demonštrácie odvodenej od triedy `DemoHandler`. Metódy triedy umožňujú zmenu zložiek materiálu. Ďalej obsahuje metódy pre vytvorenie a pripojenie 3D objektov do podstromu objektu.

4.3.6 Trieda `LightSubtree`

Trieda implementuje svetelný zdroj. Obsahuje preťažené metódy pre transláciu objektu, vytvorenie schematického názvu, ktorý sa zobrazuje v priestore spolu zo svetlom a naznačuje jeho polohu. Ďalej obsahuje metódy pre zmenu intenzity zložiek svetla.

4.3.7 Trieda `HelperSubtree`

Táto trieda implementuje pomocné objekty nefotorealistického zobrazovania, ktoré sú nápomocné pre zdôraznenie demonštrovanej tematiky. Obsahuje metódu pre vpisovanie textu do okna časti demonštrácie, pre vytváranie drôtených modelov geometrických objektov, či vkladanie osí súradníc s popisom do scény.

4.3.8 Trieda `ViewerWX`

Trieda zdedená z `wxGLcanvas` a `GraphicsWindow` z knižnice OSG implementuje zobrazovacie okno ktoré je umiestnené v rámci priestoru okna, ktoré implementuje trieda `MainFrame`. Obsahuje metódy pre zmenu polohy a smeru pozorovateľa v scéne, ako reakciu na udalosti kliknutia myši. Umožňuje vizualizáciu demonštračných scén a interakciu pohybu kamery s akciami užívateľa.

Kapitola 5

Implementácia

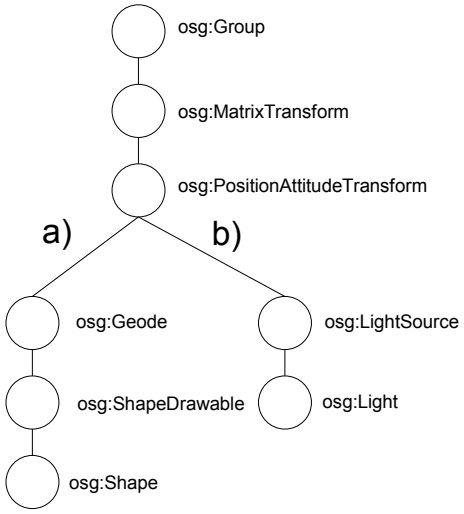
V tejto kapitole popíšem niektoré metódy, ktoré majú význam pre celú koncepciu programu, hlavne inicializáciu a deinicializáciu tried implemetujúcich jednotlivé demonštrácie resp. postupy pri vytváraní demonštrácií. V tejto časti popíšem aj implementáciu niektorých efektov v súvislosti s OSG alebo priamo s OpenGL[2][1].

5.1 Inicializácia demonštrácie

Pri inicializácii aplikácie sa vytvoria inštancie tried zdedených z `DemoHandler`, ktorá obsahuje virtuálnu metódu `Init()`. Tá je v každej triede implementujúcej demonštrácie definovaná iným spôsobom v závislosti na obsahu demonštrácie. Pri zvolení položky menu sa súčasná, do stromu vizualizačného okna pripojená trieda deinicializuje. V kontexte zvolenej položky menu sa zavolá metóda `Init()` inštancie triedy. Metóda obsahuje sekvenciu príkazov vytvárajúcich objekty scény, rozdelených do častí na základe obsahu demonštrácie. Pre vytvorenie časti demonštrácie sa volá metóda `createStage()`. Jednotlivé časti demonštrácie sú uložené ako atribút implementovaný ako zoznam zoznamov ukazovateľov na objekty (pre každý druh). Po pridaní jednotlivých inštancií do scény metódami `AddGeometryToCurrentStage`,`AddLightToCurrentStage` a `AddHelperToCurrentStage` (predaním ukazovateľa) sa zavolá metóda `conectObjects()` ktorá pridá do atribútu `groupList()` koreň podstromu časti demonštrácie.

5.2 Deinicializácia demonštrácie

Pri každej zmene demonštrácie voľbou položky z menu `MainFrame` sa súčasná demonštrácia deinicializuje. Tento postup som sa rozhodol implementovať na základe pamäťových požiadaviek každej scény. Súčasná inicializácia všetkých metód a použitie väčšieho množstva rozmerných textúr by spôsobilo značné spomalenie celej aplikácie. Preto sa pri zmene demonštrácie volá metóda `DeInit()`. Táto vnorenými cyklami pre každý druh objektu v scéne zavolá ich metódy `CleanGeoSubtree()`, `CleanHelperSubtree()` a `CleanLightSubtree`, ktoré dealokujú pamäť. Zároveň sa dealokujú položky zoznamu koreňových uzlov častí demonštrácie a paleta textúr.



Obr. 5.1: Diagram štruktúry reprezentácie 3D objektov

5.3 Reprezentácia 3D objektu

Koncepciu reprezentácie 3D objektov som založil na spôsobe reprezentácie scény v OSG. Pre potreby aplikácie som zvolil štruktúru 5.1. Koreň stromu tvorí inštancia triedy `osg::Group`, slúži pre pripájanie do hlavného uzla časti demonštrácie. Spolu s uzlom inštancie triedy `osg::MatrixTransformation` (pre účel animácií) a `osg::PositionAttitudeTransform`, tvoria atribúty triedy `EntitySubtree`. Zdedením a rozšírením do triedy `GeoSubtree` dopĺňa štruktúru stromu, ktorá je špecifická pre 3D objekt (vetva a)). Pre reprezentáciu svetelného zdroja som dedením a rozšírením do triedy `LightSubtree` zvolil atribúty špecifické pre tento druh (vetva b)). Trieda `HelperSubtree` neobsahuje z hľadiska štruktúry stromu rozšírenie atribútov. Vytvorenie tohoto typu objektu realizujem pripojením podstromu do uzla `osg::PositionAttitudeTransform`.

5.4 Difúzne, ambientné a spekulárne textúry

Tieto druhy textúr ovplyvňujú intenzitu zložiek odrazeného svetla. V programe som sa ich rozhodol implementovať pomocou vhodného nastavenia zložiek materiálu a aplikovaním textúry. Zmena materiálu objektu v OSG prebieha vytvorením atribútu - inštancie tried `osg::Material`, ktorá zapúzdruje `glMaterial` v OpenGL. Nastavenie zložiek materiálu objektu som zvolil na základe typu textúry, ostatné som ignoroval nastavením na nulové hodnoty.

5.5 Transparentné textúry

Pre transparentné textúry je dôležité, aby zdroj textúry obsahoval informáciu o priehľadnosti. Implementoval som aktiváciou stavu GL_BLEND. OSG v tomto prípade poskytuje pre renderovanie transparentných textúr abstrakciu v podobe samostatného renderovania 3D objektov s aplikovanou transparentnou textúrou a následnú syntézu s ostatnými na základe

hodnoty z-bufferu. Predíde sa tak neželaným optimalizáciám scény, ktoré by prekryté objekty odstránili.

5.6 Bump textúry

Pri implementácii som využil prostriedky OSG, konkrétne som odvodil vlastnú triedu `BumpCallback` z triedy `osg::NodeCallback` (preťažením konštruktora a definíciou metódy `operator`).Metóda `operator` sa zavolá pri priechode uzlom scény, do ktorého je priradená inštancia triedy `osg::NodeCallback`. Inštanciu triedy `BumpCallback` som zaregistroval do uzla, v ktorom je pripojený podstrom objektu. Zmenou zložky materiálu som simuloval zmenu polohy svetla. Následnou aplikáciou normálovej mapy a nastavením stavu `glEnvCombine`(kombináciou primárnej farby materiálu a normálovej mapy na základe DOT3) som dosiahol efekt bumpmapping. Do nasledujúcej textúrovacej jednotky som priradil difúznu textúru. Vhodnou animáciou svetla som dosiahol súlad so vzhľadom povrchu objektu.

5.7 Displacement textúry

Pre implementáciu som použil inštanciu triedy `osg::HeightField`. Trieda umožňuje vytvoriť pravidelné pole vertexov, ktorých translácia v smere normály sa dá pozmeniť. V tomto prípade som použil užívateľom vybranú textúru, na základe ktorej som previedol transláciu pre dosiahnutie súvisu medzi difúznou textúrou a vzhľadom objektu.

5.8 Procedurálne textúry

Procedurálne textúry som implementoval jednoduchým algoritmom ktorý pristupoval k dátam inštancie triedy `osg::Image`. Pre vytvorenie textúry šumu som použil generátor pseudonáhodných čísel `rand()` ktorý je súčasťou C++, pre generovanie gradientu, vstupný parameter a jednoduchý algoritmus založený na súradniciach texelu.

5.9 Generovanie textúrovacích súradníc

Táto časť implementácie pokrýva porovnanie rozmerov textúr, generovanie textúrovacích súradníc inverzným mapovaním, mapovanie prostredia na povrch gule a kocky (cube environment mapping). Využil som triedu osg::Texgen, ktorá zapúzdruje stav glTexGen OpenGL API.

Pri aplikácii 1D textúry na 3D objekt som nastavil stav generovania textúry na základe súradnice S (nastavením stavu GL_TEXTURE_GEN_S) a deaktivoval stav pre ostatné rozmery. Pri aplikácii 2D textúry som aktivoval generovanie súradnice aj pre os T (nastavením stavu GL_TEXTURE_GEN_T).

Pre zobrazenie 3D textúry som zvolil jej vygenerovanie na základe algoritmu. Vytvoril som triedu `TexgenCallback` odvodením od triedy `NodeCallback`, ktorá pri priechode stromom animuje 3D textúru po objekte, čím sa zvýrazní efekt vyrezania objektu z kusu materiálu. Pre korektné zobrazenie 3D textúry som aktivoval OpenGL stav GL_TEXTURE_GEN_R, pre generovanie tretej súradnice textúry na vertexoch objektu.

Pri implementácii mapovania prostredia som aktivoval stav GL_REFLECTION_MAP_ARB, ktorý generuje súradnice na vertexoch objektu na základe

polohy, smeru pohľadu a normálového vektora vertexu. Pri mapovaní prostredia na plášť kocky som využil triedu `osg::TextureCubeMap`, ktorá zapúzdruje funkcionalitu cube mapovania v OpenGL.

5.10 Filtrovanie textúr

Filtrovanie textúr zapúzdruje OSG do metód tried odvodených od `osg::Texture`. V mojej implementácii som filter textúr nastavoval volaním metódy `setFilter` zodpovedajúcej triedy. Táto metóda zapúzdruje stav `glTexParameteri(target,pname,param)`, ktorý volí textúrovací filter na základe nastavenia parametrov. Pre implementáciu metódy najbližšieho suseda som použil nastavenie GL_NEAREST, pre bilineárne filtrovanie GL_LINEAR.

Aktivácia mipmappingu je analogická s nastavením filtra textúr na GL_LINEAR_MIPMAP_NEAREST pre bilineárne filtrovanie a GL_LINEAR_MIPMAP_LINEAR pre trilineárne filtrovanie pri mipmappingu. Dôležité z hľadiska demonštrácie bolo použitie mnou vytvorených súborov mipmáp. Ako zdroj textúry som využil súbory typu DirectDraw Surface (DDS) [5] s ohľadom na možnosti OSG. Tento formát mi umožnil uloženie jednotlivých mipmáp v poradí a veľkosti, ktorá bola vhodná pre danú demonštráciu.

Kapitola 6

Záver

V závere by som chcel zhodnotiť výsledky a význam tejto práce. Cieľom bolo implementovanie aplikácie pre demonštrovanie metód textúrovania 3D objektov, ktorá by slúžila pre podporu výuky resp. ako študijná pomôcka pre študentov. V dnešnej dobre je trendom súčasné používanie viacerých metód a typov textúr pri aplikácii na 3D objekt. Prezentácia týmto spôsobom by svoj účel, vzhľadom na neprehľadnosť a nemožnosť odlíšenia jednotlivých metód, určite nesplnila. Jedným s cieľov bolo aj nájdenie takého spôsobu prezentácie jednotlivých metód, ktorý by užívateľa motivoval k ďalšiemu štúdiu tejto problematiky.

V mojom návrhu som sa snažil o prehľadné rozdelenie a následnosť demonštrácií ako aj postupné zavádzanie pojmov. Týmto postupom som sa snažil užívateľa motivovať rozložením problematiky na menšie a ľahšie pochopiteľné celky. Dôležitým prvkom bolo aj porovnanie metód s dôrazom na nedostatky obrazu, ktoré potláčajú. V aplikácii je možné kedykoľvek zobraziť teoretický základ, čo uľahčuje pochopenie a napomáha k upevneniu vedomostí. Ovládanie aplikácie je jednoduché, pričom ale obsahuje množinu všetkých prvkov významných pre implementované demonštrácie. Z implementačného hľadiska som sa snažil vzhľadom na knižnicu OpenSceneGraph a jej súvislosť s grafickým API OpenGL, využívať čo možno najmenšiu množinu rozšírení. Tento postup sa mi zdal z hľadiska prenositeľnosti a kompatibility aplikácie s grafickým hardwarom dôležitý. Pri implementácii demonštrácií, ktoré využívajú pamäťovo náročné a rozmerné textúry a geometriu bolo dôležité správne uvoľňovať pamäť pri zmene demonštrácie. Aj napriek mojej snahe, používaniu smart pointerov, obsiahnutých v knižnici OpenSceneGraph, sa počas používania programu agreguje neuvoľnená pamäť.

Prínosom práce pre mňa bola možnosť aplikovať získané teoretické poznatky pri implementácii aplikácie, zoznámenie sa s problematikou demonštrácií a v neposlednom rade s knižnicou OpenSceneGraph a grafickým API OpenGL.

Možností vylepšenia aplikácie je mnoho. Ako kľúčové rozšírenie by bolo odstránenie problémov s uvoľňovaním pamäte. Z hľadiska funkčnosti je možnosť rozšírenia súboru demonštrovaných metód nielen z pohľadu textúrovania ale aj z iných oblastí počítačovej grafiky resp. obohatenie aplikácie o editáciu demonštračných scén, čím by sa program stal univerzálnym demonštračným nástrojom.

Dodatek A

Ovládanie programu

V tejto kapitole stručne popíšem funkčnosť ovládacích prvkov aplikácie a spôsob ovládania demonštračnej scény.

Aplikácia pozostáva z dvoch samostatných okien. Pre výber konkrétnej demonštrácie a zobrazenie nápovede a teórie slúži hlavné okno programu, ktoré zároveň slúži aj na zobrazenie demonštračnej scény a jej ovládanie. Pre zmenu parametrov jednotlivých objektov užívateľ využije druhé okno. Funkčnosť ovládacích prvkov je závislá na vybranom objekte a konkrétnej demonštrácii.

A.1 Výber demonštračnej scény

Pre spustenie demonštrácie konkrétnej metódy textúrovania a zobrazenie pomoci a teórie v programe slúži štruktúrované menu hlavného okna aplikácie A.1 a),b). Po zvolení sa v okne demonštrácie zobrazí prvá časť vybranej demonštrácie, pričom sa nastavia ovládacie prvky v sekundárnom okne. Aktiváciou položky menu Pomoc A.1 c) sa spustí prednastavený internetový prehliadač v ktorom sa zobrazí teória alebo popis ovládania programu.

A.2 Zmena parametrov objektu

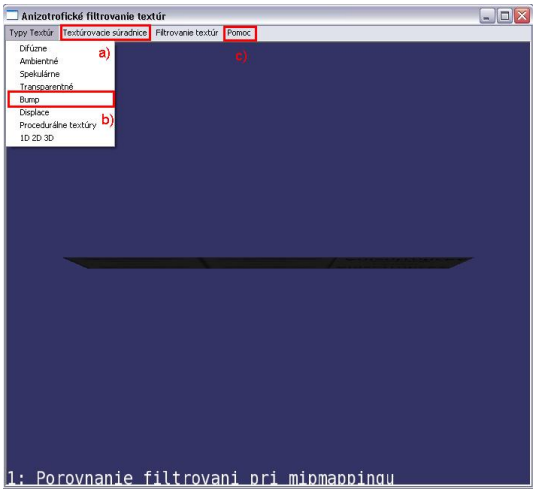
Pre zmenu parametrov demonštračnej scény resp. jej časti slúži sekundárne okno aplikácie. Pre navigáciu v rámci častí demonštrácie slúžia ovládacie prvky na obr. A.2 a). Umožňujú presun medzi časťami demonštrácie. V prípade nevhodnej manipulácie s pohľadom v grafickom okne je možné aktiváciou ovládacieho prvku *Východzia poloha* umiestniť pohľad do scény na predvolenú pozíciu.

V časti demonštrácie, kde je predvolená interakcia s 3D objektami, je možné zvoliť objekt ktorého parametre chceme meniť A.2 b).

Zmenu textúry z predvolenej palety pre konkrétnu demonštráciu je možné výberom z ovládacieho prvku A.2 c). Textúra sa pri dokončení výberu aplikuje na objekt.

Pre potreby demonštrácie anizotropického filtrovania slúži ovládací prvok A.2 d). Zvolený stupeň anizotropie textúry vybraného objektu sa aplikuje okamžite pri zmene ovládacieho prvku.

Generovanie procedurálnej textúry je realizované skupinou ovládacích prvkov A.2 d). Zvolením typu procedurálnej textúry, nastavením parametru a aktivovaním ovládacieho prvku 'Vygeneruj' sa vygeneruje a aplikuje textúra na vybraný objekt.



Obr. A.1: Výber demonštrácie a pomoci

Obr. A.2: Navigácia v rámci demonštrácie

A.3 Ovládanie demonštračného okna

Ovládanie demonštračného okna je pre všetky demonštrácie realizované na základe spracovania udalostí generovaných pri práci s myšou v rámci okna, v ktorom sa renderuje demonštrácia.

Zmena polohy kamery scény je realizovaná aktiváciou ľavého tlačítka myši a ťahom v rámci plochy okna. Zmenu vzdialenosti kamery od ohniskového bodu je realizované aktiváciou pravého tlačítka myši a ťahom v požadovanom smere. Súčasný posun ohniskového bodu a kamery je realizované aktiváciou stredného tlačítka myši a ťahom v požadovanom smere, poloha sa mení v rámci plochy, ktorá je kolmá na spojnicu kamery a jej ohniska.

Dodatek B

Obsah CD

Adresárová štruktúra priloženého CD je nasledovná:

- Adresár `technicka_sprava`:
  - `bp.pdf` - technická správa bakalárskej práce
  - `bp_zdroj` - adresár obsahuje zdrojové súbory technickej správy bakalárskej práce
- Adresár `zdrojove_subory`:
  - `wrender.sln` - súbor solution pre VS 2005
  - `bin` - adresár obsahuje potrebné knižnice pre skompilovaný spustiteľný súbor
  - `source` - adresár obsahuje zdrojové súbory programu a súbor `wrender.vcproj` programu VS 2005
  - `data` - adresár obsahuje súbory textúr, fontov a pomoci nevyhnutných pre chod demonštrácie
- Adresár `exe`:
  - `data` - Adresár obsahuje súbory textúr, fontov a pomoci nevyhnutných pre chod demonštrácie
  - `bin` - Adresár obsahuje spustiteľný súbor `wrender.exe`
- Adresár `doc`:
  - `html` - Adresár obsahuje programovú dokumentáciu vygenerovanú programom Doxygen, vstupom je súbor `index.html`
- Adresár `library`:
  - `osg2.2.0_vs80_setup_2007-10-08.exe` - inštalačný súbor knižnice OpenSceneGraph
  - `wxMSW-2.8.7-Setup.exe` - inštalačný súbor knižnice wxWidgets

Literatura

[1] Richard S. Wright Jr. aj. *OpenGL superbible : comprehensive tutorial and reference*. 4. edition, Addison-Wesley Publishing Company, 2007. ISBN 0-321-49882-8.

[2] WWW stránky. Opengl. `http://www.opengl.org/`. [cit. 19.1. 2008].

[3] WWW stránky. Openscenegraph. `http://www.openscenegraph.org/projects/osg`. [cit. 19.1. 2008].

[4] WWW stránky. Wikipedia - anisotropic filtering. `http://en.wikipedia.org/wiki/Anisotropic_filtering`. [cit. 19.1. 2008].

[5] WWW stránky. Wikipedia - directdraw surface. `http://en.wikipedia.org/wiki/DirectDraw_Surface`. [cit. 20.1. 2008].

[6] WWW stránky. wxformbuider. `http://wxformbuilder.org/`. [cit. 19.1. 2008].

[7] WWW stránky. wxwidgets. `http://www.wxwidgets.org/`. [cit. 19.1. 2008].

[8] Jiří Žára aj. *Moderní počítačová grafika*. 2.vydání, Brno, Computer Press, 2004. ISBN 80-251-0454-0.