Diplomový projekt

Generátor virtuální městské zástavby

2006 Rolf Čábelka

Zadání:

Název: Generátor virtuální městské zástavby
Vedoucí: Pečiva Jan, Ing., UPGM FIT VUT

1.Nastudujte si problematiku rendrování virtuálních scén, speciálně se zaměřte na reprezentaci objektů ve scéně.
2.Navrhněte algoritmy pro umisťování budov na nerovný virtuální zemský povrch.
3.Navržený algoritmus implementujte. Volitelně můžete využít k implementaci knihovny Open CASCADE.
4.Pokuste se aplikovat stejný algoritmus na objekty typu cesty, silnice apod. Pokud se ukáže jako nevhodný navrhněte a ale alespoň částečně implementujte vhodnější.
5.Pokuste se vhodným způsobem rozmísťovat budovy způsobem který by budil zdání "reality".
6.Vyhodnoťte a diskutujte dosažené výsledky.

Kategorie: Počítačová grafika

Implementační jazyk: C++

Volně šířený software: Coin

Literatura :

[1.] Tutoriál k Open Inventoru na www.root.cz
[2.]Josie Wernecke, The Inventor Mentor, Addison-Wesley Professional 1994 ISBN: 0201624958

Prohlášení:

Prohlašuji, že jsem tento semestrální projekt vypracoval samostatně pod vedením Ing. Jana Pečivy

Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Brně dne 18.5.2006

……
Rolf Čábelka

Poděkování

Rád bych tímto poděkoval vedoucímu diplomové práce ing. Janu Pečivovi za poskytnutou pomoc.

Abstrakt :

Projekt se zabývá generováním a zobrazováním schématu městské zástavby. Cílem bylo vytvořit program, který generuje virtuální městskou zástavbu bez cest na členitý terén a s cestami na plochu. Základními stavebními elementy jsou budovy které tvoří městské struktury, centra, čtvrtě, náměstí a předměstí.

Algoritmus generování virtuální městské zástavby bez cest na členitý terén se skládá ze čtyř hlavních kroků. V prvním kroku algoritmu vyhledáváme na mapě vhodnou lokalitu pro vytvoření centra města, jestliže se vhodná lokalita najde vytvoří se centrum města. V druhém kroku algoritmu hledáme vhodnou lokalitu pro předměstí, které stejně jako centrum po úspěšném vyhledání vytvoříme. Třetí krok algoritmu je závislí na úspěšnosti předchozích kroků. Jedná se o tvoření ulic nacházející se mezi centrem města a jeho předměstím.V posledním kroku algoritmus zastaví okolí města.

Algoritmus generování virtuální městské zástavby s cestami na plochu si lze představit jako šachovnici, kde na bílá pole jsou umístěny předem připravené městské struktury a na černá pole se generují náhodné městské struktury.

Zobrazování scény je provedeno s použitím Coin3D, což je sada knihoven používaných pro tvorbu 3D grafických aplikací vystavěných nad OpenGl. V Coin3D je scéna tvořena jako graf, jenž má stromovou strukturu. Každý z listových uzlů nese informaci o grafické operaci, kterou budeme provádět a nelistové uzly slouží k uspořádání scény do hierarchických struktur.

Klíčová slova :

Open Inventor, Coin, OpenGl, Krajina, Budova, Městská struktura, Centrum, Čtvrť, Náměstí, Předměstí, Ulice, Generování scén, Zobrazování scén, Textura, Model.

Abstract :

Diploma thesis „Virtual Town Development" deals with generating and rendering diagram of town development. The purpose was to create a programme which generates Virtual Town Development without roads on a broken terrain and with roads on a surface. Fundamental building elements are buildings which create town structure, centres, districts, squares and suburbs.

Algorithm of generation Virtual Town Development without roads on a broken terrain consists of four main steps.In the first step of an algorithm we search on a map for the right locality to create the town centre. If it exists the town centre will be created. In the second step of an algorithm we search the right locality for the suburbs which we will create as well as a centre after successful searching. The third step of an algorithm depends on a success of previous steps. It is about creating streets between town centre and its suburbs. In the last step an algorithm will build up surrounding of the town.

Algorithm of generation Virtual Town Development with roads on a surface is possible to present like a chessboard where town structures are set in advance on a white field and chance town strucrures generate a black field.

Rendering scene is provided by Coin3D, which is a set of libraries used for production 3D graphic application raising above OpenGI. In Coin3D scene is created like a graph that has tree structure. Each of leaf nodes presents information about graphic operation which we will perform and non leaf nodes serving to set up the scene into hierarchical structures.

Key words :

Open Inventor, Coin, OpenGl, Landscape, Building, Town structure, Centre, District, Square, Suburb, Street, Generateing scenes, Rendering scenes, Texture, Model.

Obsah:

0. Úvod 08

1. Historie urbanizace krajiny
aneb počátky měst a urbanismus v kostce 09

2. Zobrazování měst 13

2.1 Od papírové mapy ke GIS
(geografickému informačnímu systému) 13
2.2 Města v počítačových hrách 15
2.3 Modely měst, simulovaní a generování města 17

3. Programové prostředí 20

3.1 Úvod 20
3.2 Historie 20
3.3 Licence 20
3.4 Stručný popis Coin 21

4. Zobrazení scény 23

4.1 Jak Coin zobrazuje scénu 23
4.2 Graf scény virtuální městské zástavby na členitý terén 23
4.3 Modely budov 23

5. Generování zástavby na členití terén bez komunikací 30

5.1 Centrum 31
5.2 Předměstí 32
5.3 Ulice 33

6. Generování zástavby na plochu s komunikacemi 34

6.1 Knihovna Open CASCADE, přechod z členitého terénu na plochu 34
6.2 Tvorba městských struktur 35

6.2.1 Centra, osvětlení kódu programu a jeho napojení na graf scény 38
6.2.2 Čtvrtě 44
6.2.3 Náměstí 45
6.2.4 Předměstí 47

6.3 Generování náhodné městské struktury 48
6.4 Spojení městských struktur, vznik města 50

7. Vyhodnocení dosažených výsledků 52

7.1 Hodnocení programu 52

7.1.1 Hodnocení budov 52
7.1.2 Hodnocení generování virtuální městské zástavby na členitý terén 53
7.1.3 Hodnocení městských struktur 53
7.1.4 Hodnocení generování virtuální městské zástavby s komunikacemi 54

7.2 Shrnutí hodnocení programu 54

9. Použitá literatura 55

Příloha 1 – modely budov 56
Příloha 2 – městská zástavba na členitém terénu 59
Příloha 3 – městské struktury 60
Příloha 4 – městská zástavba s komunikacemi 62
Příloha 5 – rozšířená městská zástavba bez textur 64
Příloha 6 – graf scény programu 65

Úvod

Jak již název práce Generátor virtuální městské zástavby napovídá, jedná se v mém projektu o vygenerování struktur vzhledově se blížících městské zástavbě a vytvoření prostředků na zobrazování těchto algoritmem vygenerovaných struktur.

V první kapitole se věnuji uvedení do problematiky urbanismu a v druhé pak osvětlení řešení zobrazování měst samotný vývoj až k současnosti. V třetí kapitole se věnuji softwaru použitému k vypracování projektu, jeho stručným popisem a naznačením historie jeho vývoje. Ve čtvrté kapitole se věnuji popisu použitých knihoven Coin3d, jejich využití a posléze i samotné implementaci mého programu. Popisuji zde tvorbu jednotlivých budov. Páté kapitola pak je věnována popisu generátoru zástavby na členitý terén a jeho dílčím částím. Šestá kapitola začíná od popisu městských struktury až k popisu generátoru virtuální městské zástavby s komunikacemi. Sedmá kapitola zachycuje vyhodnocení dosažených výsledků.

Tento projekt plynule pokračuje již od ročníkového projektu, přes semestrální.Etapy vývoje byli následující. V ročníkovém projektu jsem si vytvořil skoro všechny modely budov s jednou texturou a provedl náhodné zaplnění krajiny budovami tvořící elipsový obrazec. Generování trvalo velmi dlouho a zabíralo hodně paměti důvodem byla neúměrná velikost textur.V semestrálním projektu jsem přepracoval všechny textury budov (zmenšil) a ke každé budově jsem dodělal navíc další textury, aby bylo možné jednu budovu potáhnout čtyřmi různými texturami. Dodělal jsem poslední budovy a rozmanitost budov se pěkně zvedla ještě jsem zde stihl začít generovat město na krajinu ve dvou strukturách vyplněním prostoru mezi nimi budovami jsem se zase snažil o elipsový obrazec města. V diplomovém projektu jsem se nechal inspirovat těmito strukturami, a začal jsem si je vytvářet společně s cestami. Přešel jsem z krajiny na plochu a spojením městských struktur pomocí náhodně generované městské struktury vytvořil město s komunikacemi.

1. Kapitola – Historie urbanizace krajiny aneb počátky měst a urbanismus v kostce

Počátky urbanizace spatřujeme již v primitivních přístřešcích jimiž můžeme označovat stany, zemnice, polozemnice, chýše, přístřešky a další. Hlavními materiály netrvalého charakteru jako jsou : hlína, kosti, rákos, větve, dřevo, kamení či kůže.

obr.1.1 chýše

obr.1.2 zemnice

Trvalejší sídla vytvořily až pozdější civilizace spadající do období řádově několik tis.př n. letopočtem,jež se usidlovali v povodí řek Nilu , Indu, Gangy, Amazonky, Eufratu a Tigridu jako Etruskové, Aztékové, Majové, Egypťané a další. Tyto civilizace stojí za vznikem měst : Ur, Uruk, Tenochitlan, Memphis a mnoho dalších.

obr.1.3 mapa města Ur

obr.1.4 město Ur

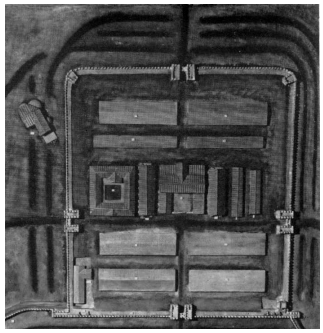
obr.1.5 model města Ur

obr.1.6 město Uruk obr.1.7 mapa města Uruk obr.1.8 a 1.9 modely města Tenochitlan



obr.2.0 města Memphis, obr.2.1 Májové, město Tikal,obr.2.2 Lost city of Pyramid

Města jako taková vznikla jako rostlá (samovolným dlouhodobým procesem) nebo jako založená panovníkem či vůdcem preferující pravidelnou zástavbu(inspirována např. římskými kastry (tábory) s pravidelnou půdorysnou osnovou šachovnicové sítě.)

obr.2.3 a 2.4 římské kastry (opevnění,tábory)

V Starém Řecku a Římě se objevovali již obytné domy, chrámy, arény, odeony, akvadukty, lázně, obelisky, ale co je důležité i náměstí s řečištěm a tržnicemi ,které sloužily jako veřejné prostranství. Římané vynalezli dokonce prapůvodce dnešního betonu „emplekton“, jež dal vzniknout stavbám jako je římské koloseum.

br.2.5 a 2.6 Odeony

obr.2.7 a 2.8 Akvadukty

Dále se měnilo město ve středověku v opevněné hradiště , kdy bylo vše schováno za mohutnými hradbami a centrum hradu tvořila kaple ze sídlem pána přičemž dokola byly uspořádány hospodářské budovy a obydlí chudiny. Ve středověku vznikla hustá neregulovaná zástavba.Středověké velkoměsto Benátky čítalo v té době sto tisíc obyvatel.Městská struktura je podnes zachována a slouží dnes spíše k turistickým účelům.Mezi dalšími velkoměsty na evropském kontinentu, které mohu jmenovat jsou např.: Praha,Paříž,Amsterdam,Londýn.

obr.2.9 model Hlásenické tvrze

obr.3.0 Hrad Švihov

obr.3.1 města Benátky

obr.3.2 mapa města Paříž r.1572

Novověk dal vzniknout mimo jiné také zámkům letním sídlům a byl přínosný pro církev jež zbudovala značné množství církevních staveb. Města se počala geometrizovat a zjednodušovat. Objevili se návrhy měst půdorysně formovaných do n-úhelníků.

obr.3.3 zámek Telč

obr.3.4 návrh pravidelného města

Až v devatenáctém století zaznamenal urbanismus velký zvrat,kdy se prohlásilo město nehygienické a počalo se s asanací a novými smělými plány , preferující symetrické uspořádání, průhledy a široké bulváry a další. S nárůstem průmyslu se objevily návrhy průmyslového města. Velmi podmětné byly návrhy zahradních měst , které se v Anglii dokonce realizovaly. V padesátých létech dvacátého století se uplatnily montované stavby(panelové domy) , jež značně změnily urbanismus krajiny . Vnesly do krajiny plochy veřejné zeleně a rozbily strukturu blokové zástavby.

obr.3.5 a 3.6 průmyslová města

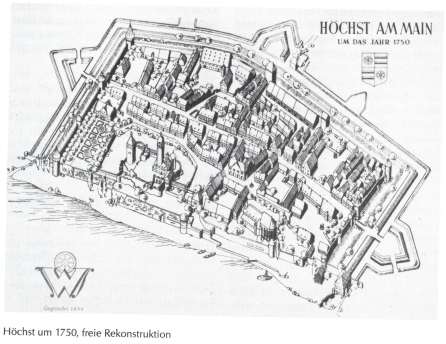
obr.3.7 propagace zahradního města obr.3.8 návrhu zahradního města

obr.3.9 a 4.0 panelových domů

2. Kapitola – Zobrazování měst

2.1 – Od papírové mapy ke GIS (geografickému informačnímu systému)

Zobrazování měst prošlo dlouhým obdobím a stále se snaží co nejlépe přiblížit k realnému zobrazení měst. Za počátek zobrazování měst lze označit plány budov, mapy. Plány budov měli za úkol poskytnout představu o podobě zobrazovaných budov. Mapy, především mapy měst potom poskytovali informace o propojení celého města (ulice,náměstí,čtvrtě,atd.). Modely měst přicházely nejčastěji s architektonickými návrhy jak nových budov tak přestavby nebo dostavby části měst ,kde nesli lepší informaci o vzhledu zastavěné oblasti než mapy.

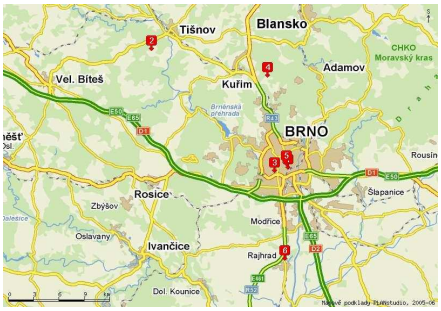


Höchst um 1750, freie Rekonstruktion

obr.4.1 a 4.2 plány budov, mapy
(na levé straně Hoechst 1750, na pravé straně Odessa)

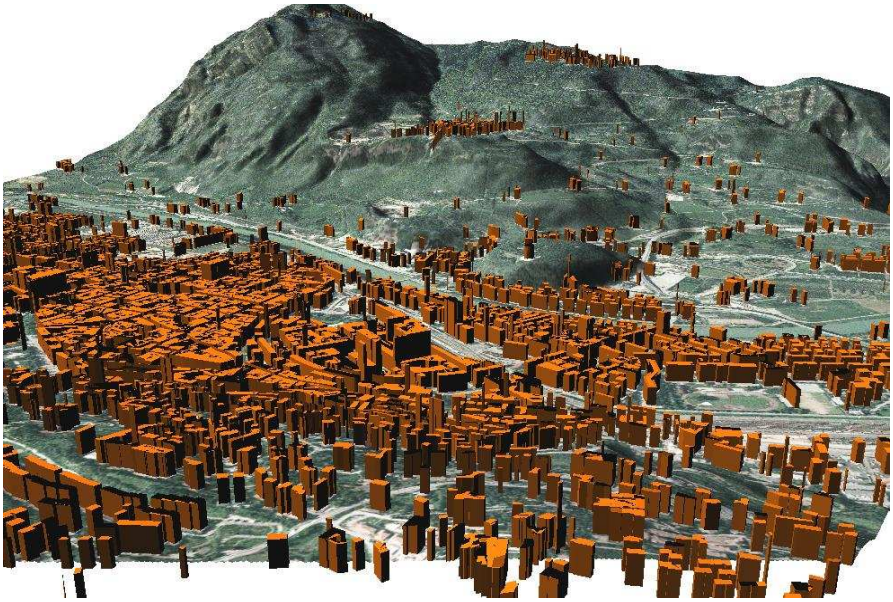
obr.4.3 model středověkého města Jihlava

V současnosti společnost vynakládá nemalé úsilí k digitalizaci těchto informací. Vznikají neustále nové mapy označovány jako elektronické nebo digitální. Výhody těchto map oproti papírovým jsou zřejmé (rychlejší vyhledávání, životnost, aktualizace,…).Snaha o globalizaci informací všech map dala za vznik GIS (geografické informační systémy), které v databázovém uspořádání překypují informacemi o dané oblasti (informaci o složení půdy, informaci o počtu obyvatel, informaci o výměrech pozemcích, informaci o počasí,…). Nejznámějším GIS je systém GRASS, který zobrazuje digitalizované krajiny i města.Více na o systému GRASS naleznete na stránkách http://grass.itc.it/index.php .



obr. 4.4 a 4.5 město Brno zobrazeno na digitální mapě napravo,
s leteckým snímkem nalevo

obr.4.6,4.7 a 4.8 nahoře ortografické mapy,
dole GIS - GRASS zobrazení města 3D



2.2 – Města v počítačových hrách

Významný podíl na dnešním zobrazování měst sehrály počítačové hry jejich rychlý vývoj a neustála snaha k přiblížení se skutečnosti s rozvojem hardweru, především pak s rozvojem grafických karet stoupli možnosti zobrazení města v širším kontextu. Města se už nemuseli znázorňovat jako 2D geometrickými prvky, nebo 2D textury.Začal se klást důraz na detail a začali vznikat města složena z jednoduchých modelů budov obalných texturou.

obr.4.9 2D město – City simulátor 1993

obr.5.0 3D město – Panter General 3D

Zobrazování měst v počítačových hrách lze pomyslně rozdělit na dvě kategorie na zobrazování měst v simulátorech a zobrazovaní měst ve strategických hrách.V simulátorech letadel se město zobrazuje většinou z vysoké výšky kde se toto město snaží co nejvíce přiblížit leteckým snímkům měst. Oproti simulátorům aut, kde se zase naopak město zobrazuje z velmi blízké vzdálenosti a většinou je vidět maximálně jedna ulice.

obr.5.1 letecký simulátor – Microsoft flight simulator 2004

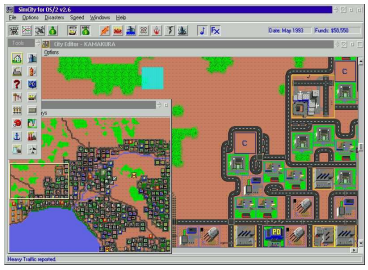
obr.5.2 simulátory aut – Need for speed

O detailní kreslení textur budov měst se asi nejvíce zasloužili strategické hry. Tyto textury mají většinou za úkol vytvořit z budovy jednoduchého tvaru, budovu budící dojem velké členitosti. Problémem měst ve strategických hrách je jejich omezenost počtem budov z toho plynoucí opakování se které při větším počtu sráží pěkný vzhled města.

obr.5.3 Civilizace

obr.5.4 Age of Empire

Jako samostatnou sekci k zobrazování měst v počítačových hrách ale i samostatné zobrazování města na počítači lze považovat hru SimCity. V této hře se snažíme vystavět město na krajině. Ukázky z této zatím čtyřdílné hry nejlépe popisuje historii zobrazení měst na počítači až k současnosti.Více na http://simcity.ea.com/about/simcity4/screenshots.php



obr.5.5-5.9 postupně od levého horního rohu SimCity,SimCity2000,SimCity3000 a SimCity4

2.3 – Modely měst, simulovaní a generování města

V této poslední podkapitole bych chtěl obrázky a odkazy poukázat na stránky zajímavých projektů zobrazování modelu měst, simulování či generování města.

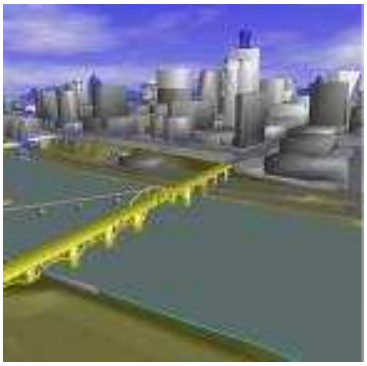
- **modely měst**

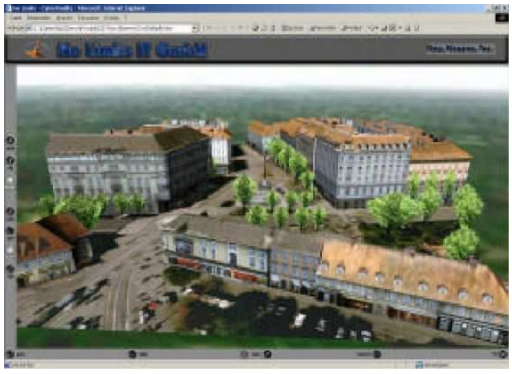
3D modely měst starověkého Egyptu
http://www.digitalegypt.ucl.ac.uk/3d/

obr.6.0 a 6.1 starý Egypt (nalevo Koptos, napravo Memphis)

3D model města St.Paul
http://www.ahpcrc.org/publications/archives/v12n2/Story3/

modelování měst pomocí geodet
http://www.riegl.com/terrestrial_scanners/3d_projects_/city_modeling_/geodata_projects_/citymodeling_geodata_.htm.





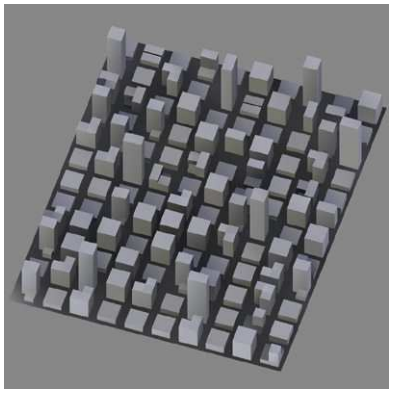
obr.6.2 model města St.Paul

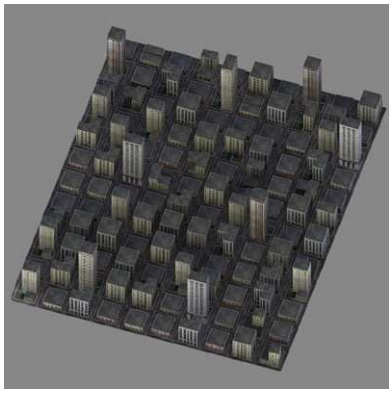
obr. 6.3 modelování měst pomocí geodet

3D models Cityscape
http://www.turbosquid.com/FullPreview/Index.cfm/ID/249203.



obr. 6.4, 6.5, 6.6 náhledy detailu v Cityscape





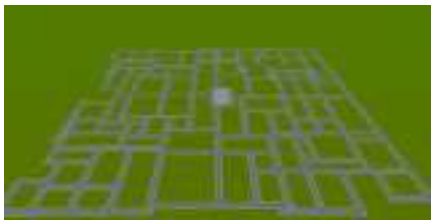
obr. 6.7, 6.8, 6.9 náhledy na Cityscape

- **simulátor města** - www.happypenguin.org/show?Open%20City.

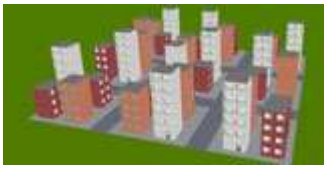
obr. 7.0 simulátor města

- **generátory měst**

Diplomový projekt ČVUT - parametrické generování modelu města
http://www.cgg.cvut.cz/vyuka/36MUS/bastam1



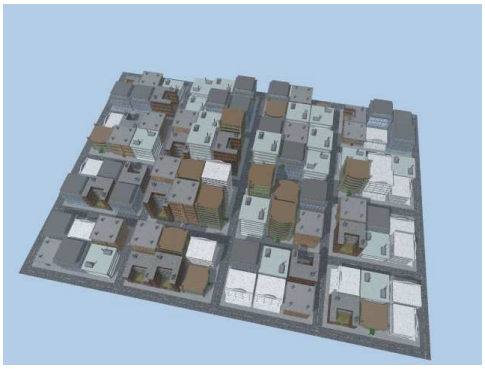




obr. 7.1-7.5 parametrické generování modelu města

City generátor

http://www.geocities.com/SiliconValley/Lakes/1434/citygen.html







obr. 7.6-8.2 City generátor

3. Kapitola – Programové prostředí

3.1 - Úvod

Tento projekt jsem vypracoval v Microsoft Visual C++ 6.0 s použitím knihoven Coin3D, což je sada knihoven používaných pro tvorbu 3D grafických aplikací vystavěných nad OpenGl . Jádro těchto knihoven je nezávislé na platformě, na níž běží. Může tedy běžet jak v prostředí Microsoft Windows, Mac OS X, GNU/Linux, SGI IRIX nebo na jiných platformách UNIXu. Potřebuje pouze překladač C++ jazyka a knihovnu OpenGl (nebo jinou knihovnu, která implementuje OpenGl API, jako je například Mesa). Grafické uživatelské prostředí může být od běžného prostředí Microsoft Windows přes Trollech's QT / Motif X Windows a nebo Mac OS X.

Aplikační a uživatelské prostředí knihoven z Coin je plně kompatibilní s SGI Open Inventorem (http://oss.sgi.com/projects/inventor/), což je v podstatě standardní grafické API pro komplexní vizualizaci 3D od firmy SGI (http://www.sgi.com/).

3.2 - Historie

Začátky Coin sahají až do roku 1995, kdy vývojáři software System in Motion vytvořili 3D grafickou knihovnu, která byla určena pro soubory formátu VRML 1.0. Tato knihovna byla určena převážně pro zobrazování těchto souborů. Během let rozšiřování, optimalizací vyvstala potřeba vše seriozně přepracovat a vytvořit i lepší design. Postupně tak vznikla knihovna, kterou autoři pojmenovali Coin.

Firma System in Motion (http://www.sim.no/), je v dnešní době světový producent real-timové grafiky (grafiky v reálném čase). Mezi zákazníky této firmy patří například Kawasaki, Mitsubishi, Boeing, nebo Bmw. Tato firma byla založena v roce 1994 v Norsku v městě Trondheim v úzké spolupráci s Norskou univerzitou Science and Technology. V roce 1997 si SiM (System in Motion) otevřela první kancelář v Oslu a během roku 2000 již vydala Coin 1.0. a v roce 2003 Coin 2.0 více o historii se lze dozvědět na webových stránkách http://www.sim.no/company/history/ .

3.3 - Licence

Knihovna Coin3D je implementována v jazyce C++ a šířena pod třemi různými licencemi. Kromě verze Coin Free Edition, která je použita v mém projektu, jsou to licence Coin Evaluation Edition a Coin Professional Edition.

Coin Professional Edition: pro použití v komerčních, nebo jinak ziskových aplikacích.
Coin Evaluation Edition: pro zhodnocení Coin3D (potřebuje registraci na stránkách SIM).
Coin Free Edition: je pod GPL pro volný vývoj software.

Více o licencích se lze dozvědět na webových stránkách firmy System in Motion na http://www.sim.no/products/Coin3D/licensing/

3.4 - Stručný popis Coin

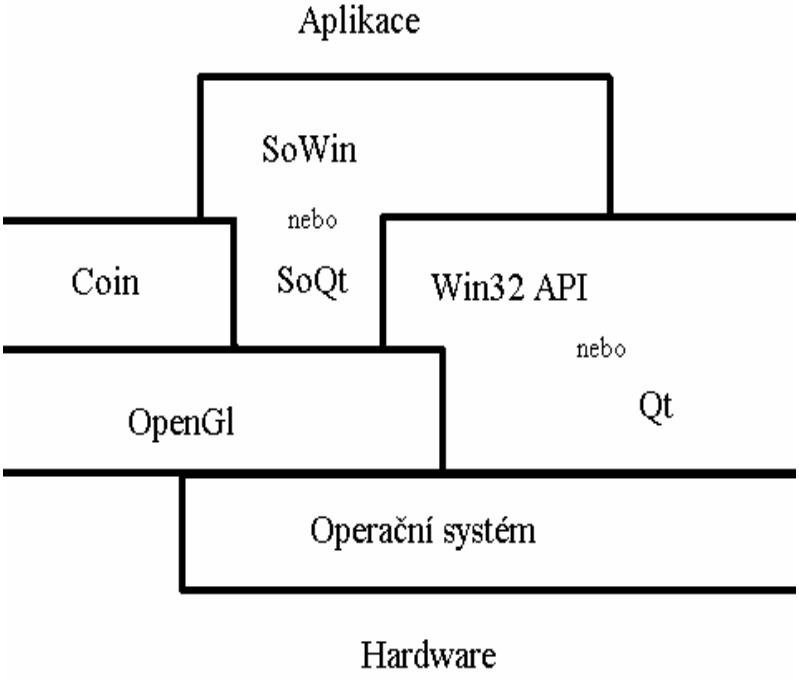
Chod systému Coin je zajišťován několika podpůrnými knihovnami, které pomáhají zobrazit výsledky vaší práce v oknech (viz obrázek 2). Jsou to:

SoQt je pro zobrazení v Trolltech's přes platformu Qt toolkit (UNIX, Windows, Mac OS X).
SoWin je pro zobrazení ve Win32 API na platformách Microsoft Windows.
SoXt je pro zobrazení v Xt/Motif na X Windows.
SoGtk je pro zobrazování GTK+.

Další důležitou knihovnou je SoImage, která slouží k nahrávání, ukládání a manipulacím s obrázky a videem. Mezi podporované formáty patří :

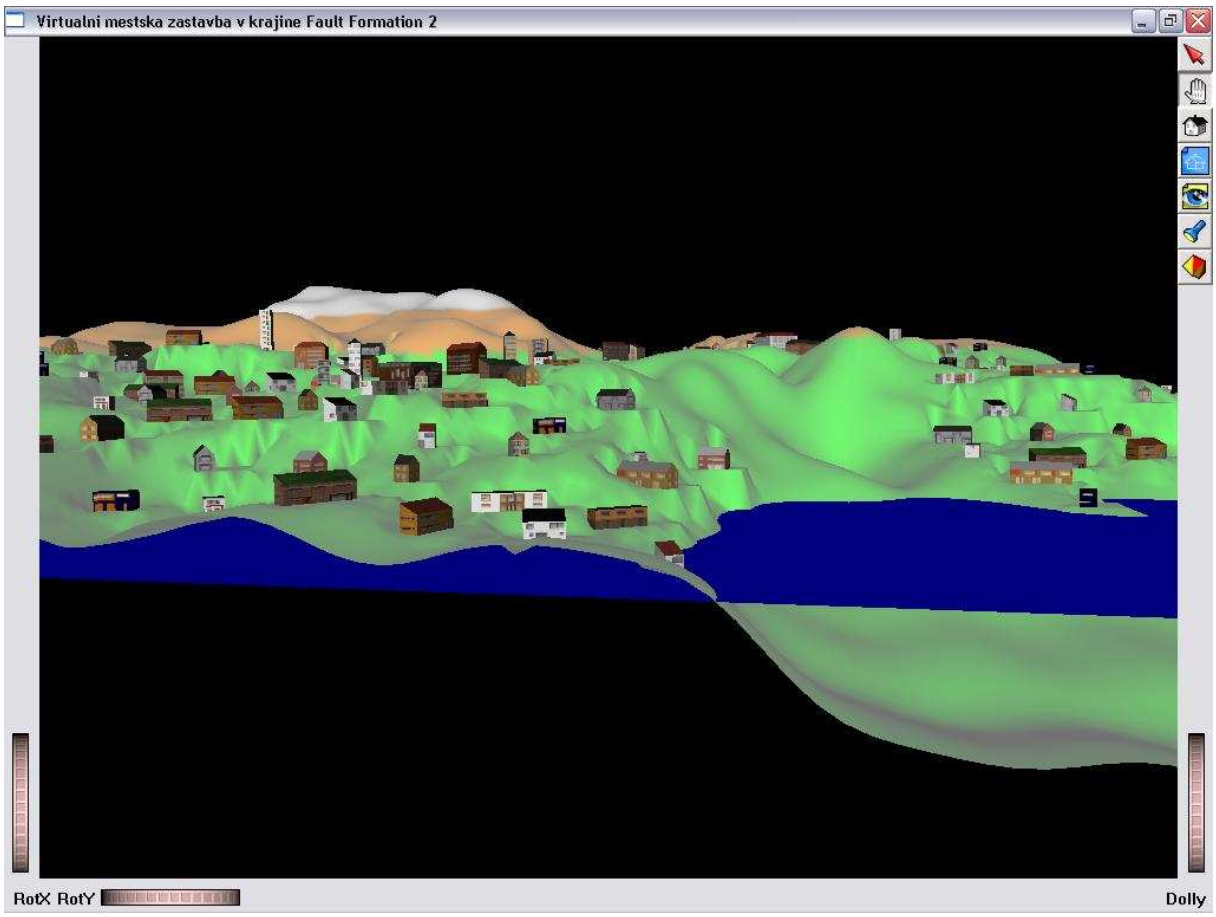
- AVI, MPEG
- JPEG (čtení a zápis přes libjpeg)
- PNG (čtení a zápis přes libpng a zlib)
- GIF, TIFF
- RGB (pouze načítání)
- PIC (pouze načítání)
- TGA (pouze načítání)
- EPS (pouze zápis)

Na tomto obrázku je znázorněno, jak jsou knihovny SoWin a SoQt začleněny do systému.



obr. 8.3 propojení knihoven v systému

Na tomto obrázku je znázorněn prohlížeč 3D scény s použitím knihovny SoWin.



obr. 8.4 Virtuální městská zástavba v krajině Fault Formation 2

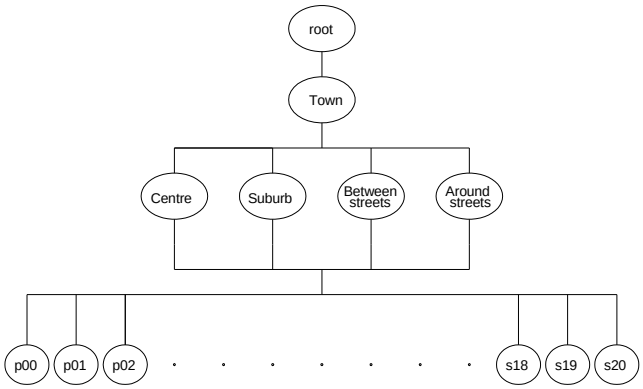
4. Kapitola – Zobrazení scény

4.1 – Jak Coin zobrazuje scénu

V Coin se scéna tvoří jako graf, do kterého postupně přidáváme uzly. Tento graf má stromovou strukturu. Nelistové uzly slouží k uspořádání scény do hierarchických struktur. Každý z listových uzlů nese informaci o grafické operaci, kterou budeme provádět. Což může být nastavení světla, posunutí souřadnic, nebo vytvoření některého z grafických primitiv (např. SoCube – hranol, SoCylinder – válec, SoSphere – koule, SoCone – kužel a podobně).Výhodu stromové struktury v Coin nejvíce oceníme, kromě přehlednosti programu, na samém konci programu.Po skončení programu je dobrým zvykem vrátit vše co jsme používali (hlavně paměť) do původního stavu tzv.po sobě uklidit. Tento úklid se v Coin provedeme velmi jednoduše zrušením hlavního uzlu strumu (většinou rootu).

4.2 – Graf scény virtuální městské zástavby na členitý terén

Jak již bylo řečeno základními stavebními elementy pro generování virtuální městské zástavby jsou budovy. Představme si je tedy jako uzly stromu (viz. předchozí kapitola). Z budov se tvoří další uzly.Ulice v okolí města, ulice uvnitř města, předměstí a v neposlední řadě i samotné centrum města.Všechny tyto uzly jsou následně spojeny v jeden uzel nazvaný město, který je navázán na hlavní uzel root ve kterém se generuje krajina scény algoritmem FaultFormation.Lepší představu vám poskytne obrázek 8.5.



obr. 8.5 graf scény, virtuální městské zástavby na členitý terén

Uzly se vytváří pomocí třídy (SoSeparator) , navazování uzlů se provádí metodou (addChild) této třídy, a rušení navázání se provádí metodou (removeChildren) taktéž této třídy více možností lze nalézt v dokumentaci k třídě SoSeparator na (http://doc.coin3d.org/Coin/classSoSeparator.html).V následujících příkladech popisuji jen nejzákladnější operace nad touto třídou.

Příklad vytvoření uzlu:

```
SoSeparator* název uzlu (root,Town,Center, Between_streets,
Around_streets) = new SoSeparator;
Název uzlu ->ref();
```

Příklad navazování uzlů:

```
root->addChild(Town);
Town->addChild(Center);
Center->addChild(Building);
```

Příklad rušení navázání uzlů:

```
root -> removeChildren(název uzlu);
```

Příklad zrušení uzlů:

```
název uzlu ->unrefNoDelete();
```

Uzly určují jednotlivé objekty a společně s jejich listy dotváří celkovou informaci o uzlu.Mezi jednotlivé listy např. patři: nastavení posunutí (SoTransform), nastavení velikosti (SoScale),nastaveni rotace (SoRotationXYZ), vytvořeni a nastavení normál (SoNormal,SoNormalBinding), nastavení textur(SoTexture2), tvorba souřadnice (SoCoordinate3), pro tvorbu trojúhelníkové sítě (SoIndexedTriangleStripSet) a mnoho jiných lze nalézt v přehledu tříd v dokumentaci Coin na (http://doc.coin3d.org/Coin/classes.html).

4.3 – Modely budov

V této kapitole si ukážeme jak vzniká model budovy krok za krokem. Všechny modely budov v našem generátoru virtuální městské zástavby jsou vytvořeny obdobným způsobem mění se samozřejmě tvar budovy, tím pádem se musíme přepočítat normály a samozřejmě se mění i textury jednotlivých stěn budovy. Jestliže se chcete dozvědět více o vytváření jednoduchých modelů v OpenInventoru vřele doporučuji článek na webových stránkách www.root.cz od Ing.Jana Pečivy přesněji na (http://www.root.cz/clanky/open-inventor-jednoduchy-model/).

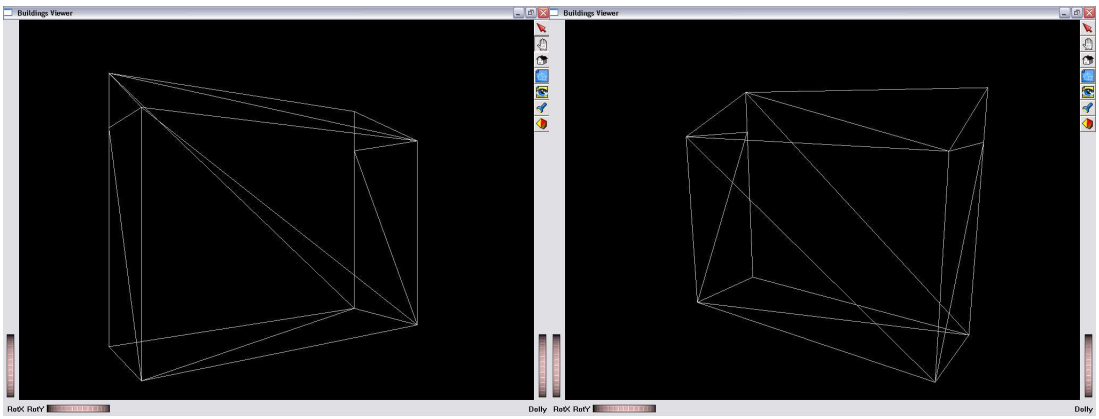
Nyní již k samotné tvorbě budov. Vizuální nástin tvorby budov je ukázán na budově s19(v příloze 1 je výsledný render hotové budovy).

Budovu si představme zjednodušeně jako těleso v prostoru. Poloha a tvar tělesa je definován pomocí souřadnicového systému jež je umístěn vždy ve středu podstavy tělesa (souřadnicově je vztažná rovina k tělesu popsána : x, y = 0, z). Pomocí souřadnicového systému pak popisujeme jednotlivé hraniční body (vertexy) ,jež definují tvar tělesa a jsou nezbytné pro další operace. Každý z těchto bodů (vertexů) se skládá ze třech prostorových souřadnic x,y,z.Jednotlivé vertexy (s19) jsou definovány

```
// VERTICES
// body v prostoru jejich propojením vytvářím model budovy
static float s19_vertices[][3] = {
-1.5f, 0, -0.5f,  // 0
 1.5f, 0, -0.5f,  // 1
 1.5f, 2, -0.5f,  // 2
-1.5f, 2, -0.5f,  // 3

-1.5f, 2.5, 0.5f,  // 4
 1.5f, 2.5, 0.5f,  // 5

 1.5f, 2, 0.5f,  // 6
-1.5f, 2, 0.5f,  // 7
-1.5f, 0, 0.5f,  // 8
 1.5f, 0, 0.5f,  // 9
};
```

Pro jednoznačnou definici tělesa tvoříme trianglstrip . Pomocí trojúhelníkové sítě ukotvíme jednotlivé body stěn. Tedy vzájemnou spojnicí tří bodů definujeme rovinu, ve které se nachází stěny objektu. Většina trojúhelníků sdílí vrcholy se svými sousedy. Abychom získaly přesný tvar jednotlivých stěn , použijeme proměnné 1→ n trianglsrips. Viz obr. 8.6.

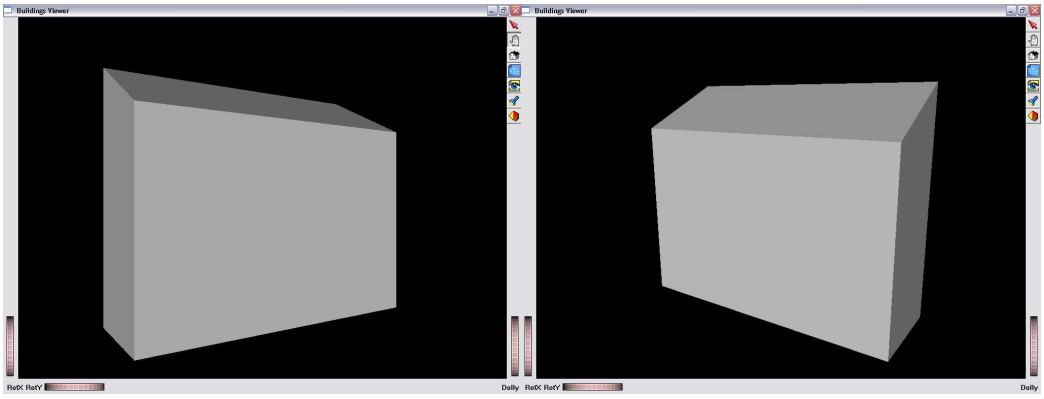
```
// jednotlive steny budovy
static int32_t s19_indices_front[] = { 1,0,2,3, -1 };
static int32_t s19_indices_up[] = { 2,3,5,4, -1 };
static int32_t s19_indices_left[] = { 5,2,6,1,9, -1 };
static int32_t s19_indices_right[] = { 4,3,7,0,8, -1 };
static int32_t s19_indices_back[] = { 4,5,8,9, -1 };
static int32_t s19_indices_bottom[] = { 0,1,8,9, -1 };

 // propojeni bodu modelu budovy
 SoCoordinate3 *coords = new SoCoordinate3;
 coords->point.setValues(0,sizeof(s19_vertices)/sizeof(float), s19_vertices);
 s19->addChild(coords);
```



obr. 8.6 propojení bodů v prostoru,
pomocí trojúhelníkové sítě znázorňuje kostru modelu budovy
tzv. drátěný model (wire model)

Pro reálné osvětlení tělesa spočítáme jednotlivé normály, tedy body kolmé k rovinám stěn budovy. Viz obr. 8.7.

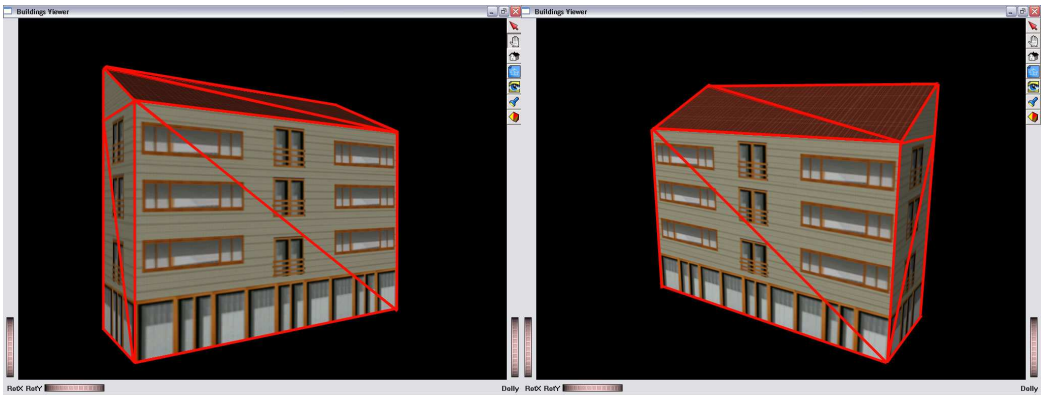
```
// NORMALS
// normaly pro dobre osvetleni sten modelu budovy
static float s19_normal[][3] = {

1, 0, 0, // 0 x
0, 1, 0, // 1 y
0, 0, 1, // 2 z

-1,  0,  0, // 3 -x
 0, -1,  0, // 4 -y
 0,  0, -1, // 5 -z

 0,0.89442f,-0.44721f  // 6 front_up sklon strechy
 };

 // normaly k jednotlivym stenam
 static int normalIndex_s19_front[] = { 5, 5 };
 static int normalIndex_s19_up[] = { 6, 6 };
 static int normalIndex_s19_left[] = { 0, 0, 0 };
 static int normalIndex_s19_right[] = { 3, 3, 3 };
 static int normalIndex_s19_back[] = { 2, 2 };
 static int normalIndex_s19_bottom[] = { 4, 4 };

 // normaly modelu budovy
   SoNormal *myNormals = new SoNormal;
   myNormals -> vector.setValues(0, 11, s19_normal);
   s19 -> addChild(myNormals);

 SoNormalBinding *myNormalBinding = new SoNormalBinding;
 myNormalBinding -> value = SoNormalBinding::PER_FACE_INDEXED;
 s19 -> addChild(myNormalBinding);
```



obr. 8.7 osvětlení pevného modelu budovy (solid model)

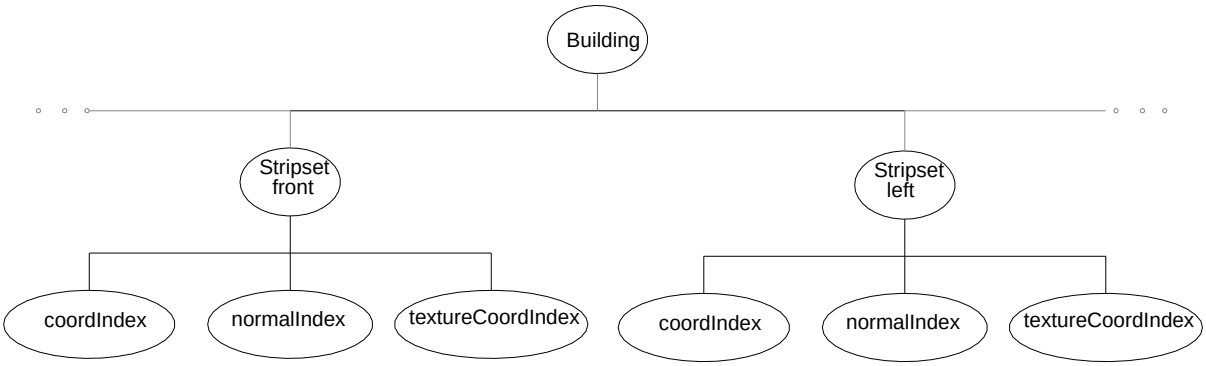
Pro usnadnění jsem si rozložil těleso na jednotlivé stěny, na které jsem nanesl textury. Jednotlivé textury (front, back, right, left, up)se přiřadí k příslušné stěně pomocí texturovacích souřadnic. Ty texturu podle texturovacích souřadnic upraví (natáhnou , či smrští) do potřebné velikosti.

```
// texturovaci souřadnice k  jednotilivým stenam budovy
static int32_t TexCoordIndex_s19_front[] = { 0,1,2,3, -1 };
static int32_t TexCoordIndex_s19_up[] = { 0,1,2,3, -1 };
static int32_t TexCoordIndex_s19_left[] = { 3,4,5,0,1, -1 };
static int32_t TexCoordIndex_s19_right[] = { 3,4,5,0,1, -1 };
static int32_t TexCoordIndex_s19_back[] = { 2,3,0,1, -1 };
static int32_t TexCoordIndex_s19_bottom[] = { 0,1,2,3, -1 };


SoTextureCoordinate2 *texCoord = new SoTextureCoordinate2;
texCoord->point.set1Value(0, SbVec2f(0,0));
texCoord->point.set1Value(1, SbVec2f(1,0));
texCoord->point.set1Value(2, SbVec2f(0,1));
texCoord->point.set1Value(3, SbVec2f(1,1));
texCoord->point.set1Value(4, SbVec2f(0,0.6666f));
texCoord->point.set1Value(5, SbVec2f(1,0.6666f));
s19->addChild(texCoord);
```



obr. 8.8 otexturovaný model budovy

Zjednodušený graf uzlu budovy



obr. 8.9 zjednodušený graf budovy

Na tomto schématu je zobrazen ve zkratce obecný postup tvorby modelu budovy. Postup je popsaný výše. Kód připojený níže se v grafu ztotožňuje s položkou stripset front. Dle schématu se napojuje k uzlu budovy.

```
// front
SoIndexedTriangleStripSet *stripSet = new SoIndexedTriangleStripSet;

stripSet->coordIndex.setValues
(0, sizeof(s19_indices_front)/sizeof(int32_t), s19_indices_front);
stripSet->normalIndex.setValues
(0,sizeof(normalIndex_s19_front)/sizeof(int),normalIndex_s19_front);
stripSet->textureCoordIndex.setValues
(0,sizeof(TexCoordIndex_s19_front)/sizeof(int32_t),TexCoordIndex_s19_fro
nt);

s19->addChild(stripSet);
```

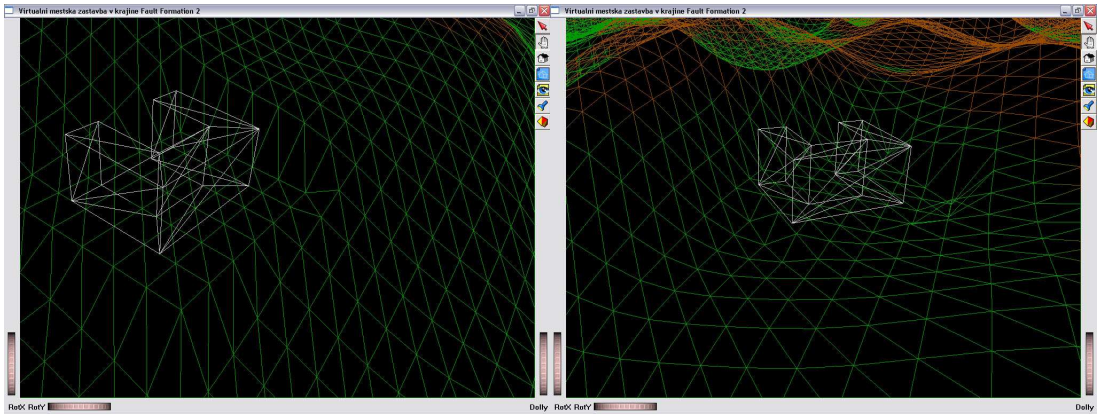
5. Kapitola – Generování zástavby na členitý terén bez komunikací

Cílem této kapitoly je popis umístění budovy na vygenerovaný terén. Morfologie našeho terénu je složitá a dochází v něm k podstatným výškovým skokům (kotliny, hornatý terén, nížina). O to složitější úlohou bylo umístění budov a vytvoření atributů virtuálního města. Lokace budovy je možná pouze mimo vodní prostory a hornatý terén. Jinými slovy budovy můžeme umístit v nížinách, kotlinách , mírně svažitém terénu a v horském terénu (nikoli však ve skalách).

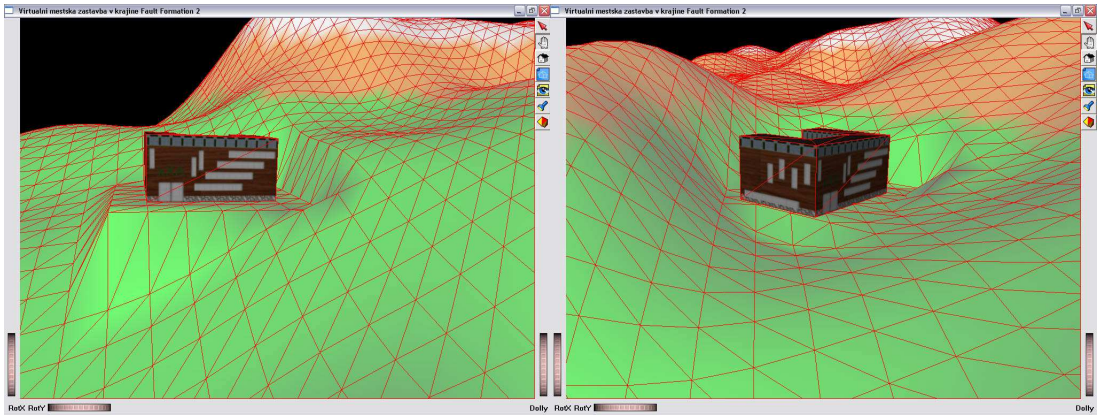
Umístění budovy lze v kostce popsat následujícími body:

1. Budovu umisťujeme do krajiny bodem x,y (odpovídající středu podstavy), který leží na krajině
2. Dle označení budovy dopočítáme souřadnice hraničních bodů v závislosti na bodu x,y
3. Provedeme přepočet hraničních bodů podstavy podle úhlu rotace
4. Přímkovým algoritmem počítáme průměrnou nadmořskou výšku oblasti , pro umístění budovy
5. Leží-li průměrná nadmořská výška oblasti umístění budovy nad úrovní vody a zároveň pod úrovní skal, zmodifikujeme výšku dané oblasti přímkovým algoritmem na průměrnou výšku této oblasti.

Daný postup je znázorněn na uvedených obrázcích.



obr. 9.0 a 9.1 umístění budovy na členitý terén,
se zapnutím zobrazovacího módu v režimu drátěného modelu

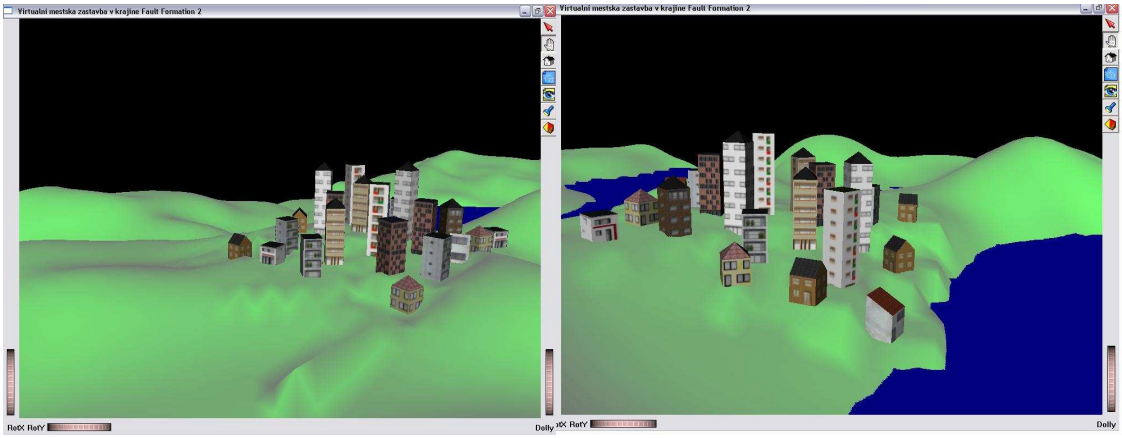


obr. 9.2 a 9.3 umístění budovy na členitý terén, se zapnutím zobrazovacího módu v režimu překrývání pevného modelu drátěným modelem (wireframe overlay)

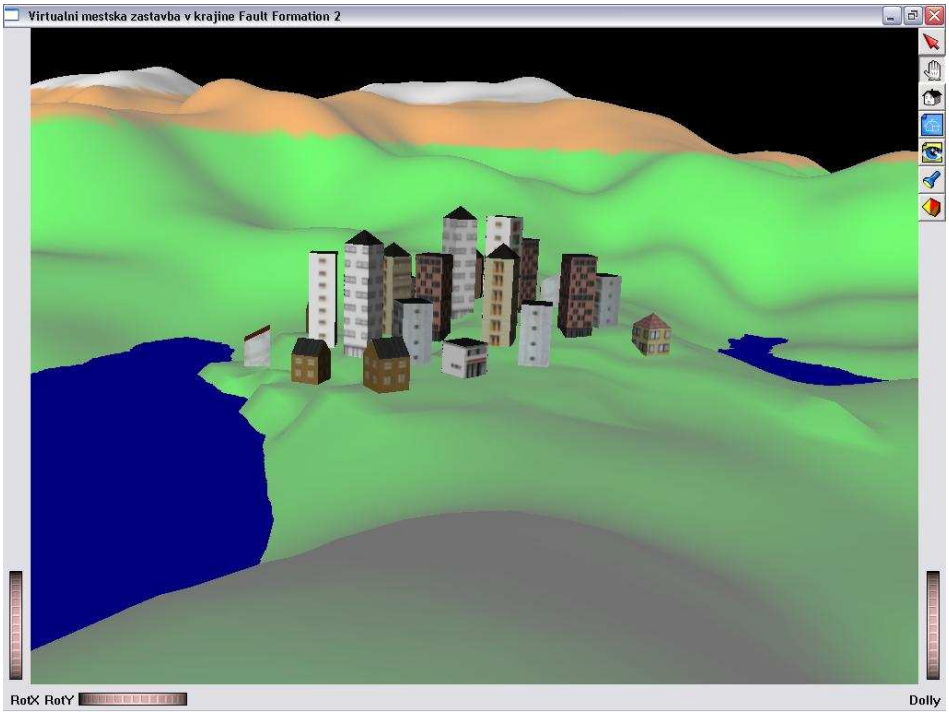
5.1 Centrum

Centrum je nejdůležitější částí města. I v mém programu hledám vhodnou lokalitu nejprve pro umístění centra a potom následují až další prvky. Algoritmus na vygenerovaném terénu nesmí umístit centrum na vodní plochu ani do skal, což je hlavní podmínkou.

Struktura mého centra byla nadefinována půdorysně s malými odlišnostmi jako šachovnicové struktura výškově vygradovaná směrem do středu struktury . Bezprostřední střed obsahuje výškové budovy . Ty jsou obklopeny nižšími budovami s určitým pootočením. Hustota zástavby klesá s vzrůstající vzdáleností od výškových budov. Na okraji centra jsou situovány nejnižší budovy řádově čítající 2 nadzemní podlaží.Obecně všechny budovy mají minimální plochu podstavy . To vše se odvíjí od drahých cen parcel.



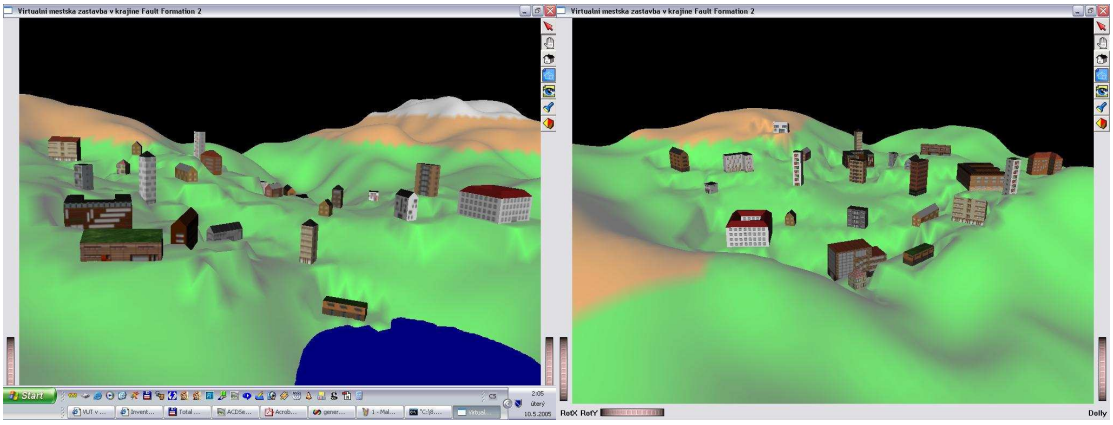
obr. 9.4 a 9.5 ukázka generování centra na členitý terén



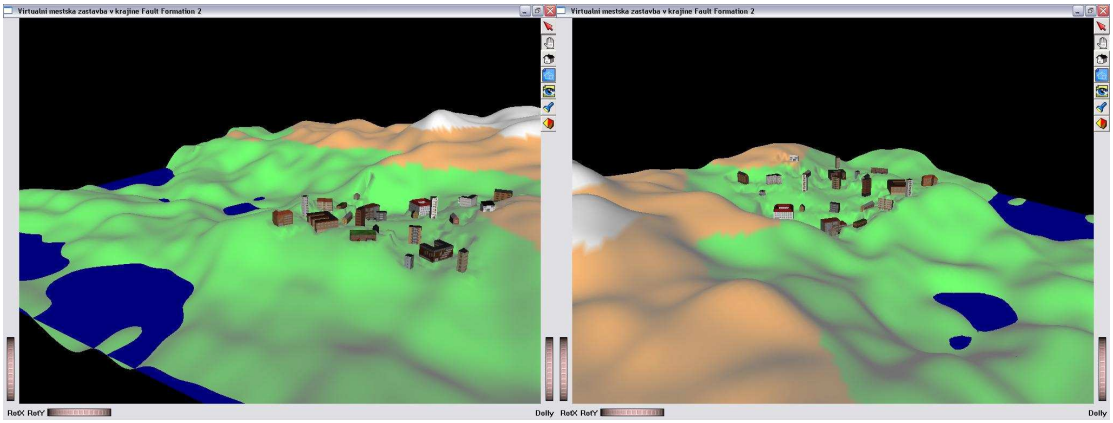
obr. 9.6 ukázka generování centra na členitý terén

5.2 Předměstí

Předměstská struktura je vzdálena od města. Díky tomu se o předměstí může hovořit jako o malém centru , které však postrádá vysokou hustotu zastavění centra ,ale také výškovou regulaci budov. S centrem je úzce spojeno avšak funguje nezávisle na něm. Jelikož cena parcel je oproti centru nepoznatelně nižší , jsou zde vystavěny budovy s velkým nárokem na plochu.



obr. 9.7 a 9.8 ukázka generování předměstí na členitý terén



obr. 9.9 a 10.0 ukázka generování předměstí na členitý terén

5.3 Ulice

Ulice v mém programu fungují jako propojení centra s předměstím . Jelikož jsem moje město pojal v novodobém smyslu , to znamená vybudované na „zelené louce“ není klasická uliční struktura viditelná na první pohled , což je umocněno umístěním vil v zeleni. Vhodným rozšířením mého projektu by bylo včlenění silničních komunikací do krajiny , které by podpořily vizuálně městský výraz.

Dalším útvarem v mé městské struktuře jsou ulice around. Z město tvorného hlediska převažují v dnešní době neohraničené městské struktury, které se stále vyvíjí. Hranice mapy mojí krajiny byla pravoúhlá. Abych navodil atmosféru rostoucího města pokusil jsem ukončit městský prostor radiálně.

6. Kapitola – Generování zástavby na plochu s komunikacemi

6.1 – Knihovna Open CASCADE, přechod z členitého terénu na plochu

Jedním z požadavků aplikace bylo nejen generovaní budov ale i generátor silnic a cest. Největší překážkou tohoto kroku byl samotný model krajiny Fault Formation 2 především pak jeho rozlišení. Velikost mapy lze zvětšit z původní velikosti 64x64 bodů na rozlišení 256x256. Větší rozlišení mapy začíná působit nevěrohodně. Tento fakt jak jsem po sléze zjistil se nachází u všech dostupných generátorů krajin.Při generování krajin se nepředpokládá že by někdo potřeboval model mapy krajiny s co největším rozlišením. V těchto případech se postupuje přesně naopak z logických důvodů menší rozlišení map vede k menším nárokům na paměť a urychlování celého procesu generování modelu krajiny.

Vyvstala otázka jak řešit problém, kde v přiblížení se skutečnému městu by bylo potřeba vygenerovat budovy měst do mapy v rozměru maximálně 2x2 bodů ale jak je z předešlé kapitoly o umísťování budov na krajinu patrné samotná podstava nejmenší budovy zabírá na modelu oblast 1x1 bodů kterou si zarovnává na průměrnou výšku před samotným umístěním budovy na krajinu zanedbáme-li zarovnaní nejbližšího okolí. Jinak řečeno, na model krajiny mapy o velikosti 64x64 bodů lze těsně vedle sebe znázornit 64x64 budov které by neměli žádné rozestupy a zarovnáváním svých míst v krajině kompletně zakryly krajinu, kterou by jsem v českých podmínkách mohli velikostně přiblížit ke kraji.

Tento velký nepoměr mezi budovami a krajinou je nejvíce patrný při vygenerování centra a porovnání jedné z výškových budov zhruba o osmi patrech s horou na krajině se zasněženým vrcholkem, které sahá zhruba k jedné třetině. Dosazením přibližných hodnot třetina z minimální výšky hory cca.1000m/3 = 300m a jedno patro budovy maximálně 3mx(8pater + 2patra – přibližná výška od hladiny) = 30m lze konstatovat že budova minimální zvětšení o proti velikosti krajiny je deset.

Řešení se zdá být velmi jednoduché buď můžeme zvětšit mapu deset krát nebo zmenšit budovy deset krát. Nevýhodou řešení spočívajícího ve zmenšování modelů budov je že tyto budovy budeme po zmenšení umísťovat na zarovnanou plošku krajiny tedy celé město se bude nacházet v jedné rovině. Naopak zvětšením rozlišením modelu krajiny by vyhovovalo našemu účelu. Problémem ale je, že v podstatě nezvětšujeme rozlišení

modelu krajiny ale jeho velikost tedy dochází k protažení krajiny které má za následek její deformaci.

Konzultací tohoto problému s vedoucím diplomové práce přineslo v roli řešitele problému knihovnu Open CASCADE http://www.opencascade.org/. Knihovna Open CACADE disponuje možností zvětšování či zmenšování rozlišení grafických modelů. Bohužel tato knihovna vyžadovala ke své funkčnosti podporu další knihovny Qt http://www.trolltech.com/.
Po zaregistrování a stažení zmenšené verze této knihovny bohužel se Open CACADE pořád při překladu s Coin3D dožadoval chybějících funkcí z knihovny Qt.

Tento neúspěch s knihovnou Open CASCADE mě přesvědčil dokončit celý projekt v Open Inventoru s knihovnami Coin3D. Možná že se tento problém jeví jako nepodstatný (postavit město na plochu či krajinu). Vrátíme-li se ale k současnému zobrazování měst popsanému v předchozích kapitolách je zřejmé že tento problém stále ve světě přetrvává. Doposud nejlepší řešení z mého pohledu přináší GIS GRASS a hra SimCity4.GRASS představuje ovšem velký databázový systém a nejedná se o žádnou malou aplikaci a ve hře SimCity4 mi členitost terénu připomíná schody patrná z cest. Tyto dva programy si po právu zaslouží obdiv, protože jiné programy nebo aplikace zobrazující město na členitý terén jsem nenašel.

6.2 – Tvorba městských struktur

Základní myšlenkou generování virtuální městské zástavby na plochu jsem si z generování virtuální městské zástavby na členitý terén odnesl v podobě generovaní města z jednotlivý městských struktur. Následné spojení těchto městských struktur tvoří výslednou virtuální městskou zástavbu.Oproti generování na členitý terén jsem musel velikosti všech městských struktur přizpůsobit jednotnému rozměru.

Všechny městské struktury si lze prohlédnou v programu pod volbou 3, kde si je lze natočit z libovolných stran a zde na konci v přílohách. Jedinou městskou strukturu, kterou si lze prohlednout jen se samostatným generováním virtuální městské zástavby na plochu je náhodná městská struktura RandomTownStucture, která zajišťuje spojování ostatních struktur města a vše o ní lze najít v následující podkapitole 6.3 – Náhodná městská struktura. V této podkapitole se nejprve zaměříme na problémy cest a po nich poukážeme jak vznikali jednotlivé městské struktury (centra, čtvrtě, náměstí a předměstí).

Dalším z požadavků bylo vhodným způsobem rozmísťovat budovy přesněji způsobem, který by budil zdání “reality“. Je zřejmé, že musí spolupracovat s požadavkem na generování silnic. Lépe řečeno generování silnic by asi mělo předcházet rozmísťování

budov i když v opačném případě by vznikl algoritmus který by znal pozice domů a snažil by se protahovat silnici mezi domy a přivést silnici ke každému z nich.

Před samotným generováním silnic je vhodné udělat malé ohlednutí po ostatních projektech s podobnou tématikou shrnout v kostce jaké jsou současné možnosti. Z kapitoly zobrazování měst je na první pohled patrné že se zatím generování silnic udržuje jen s pravoúhlém zatáčením silnic, jistě by mohl někdo namítat že na ukázce ze hry Panzer General je tomu jinak a měl by částečně pravdu.

Vysvětlení této malé záhady nalezneme na domovské stránce hry (http://hartmann.valka.cz/panzergeneral/). Zobrazení této hry je na rozdíl od ostatních projektů provedeno s hexadecimálním rastrem oproti ostatním projektům, které jsou všechny asi zobrazovány s kubickým rastrem. Tedy v této hře nalezneme i zatáčení silnic pod pětačtyřiceti stupňovým úhlem ale jen silnic mimo město. Samotné město je taktéž jako v jiných projektech pravoúhlé jen je nakonec celé otočeno pod pětačtyřiceti stupňovým úhlem.Obdobný případ nalezneme po lepším prohlédnutí ukázky ze hry Civilizace kde nejsou silnice ale cesty a jejich zatáčení je jiné než pravoúhle taktéž pouze mimo město (http://www.volny.cz/tdpar/Civ4_images.htm) domnívám se že zde již byla vytvořena sada textur zastupujících jednotlivé pootočení cest.

V mém projektu jsem postupoval obdobně, také jsem hned zjistil proč je vhodné si u města vystačit jen s pravoúhlým zatáčením cest. Jinak řečeno přivedením jiného než pravoúhlého otáčení složitost celého generování silnic posune o řád výš, který není neřešitelný ale velmi pracný.

Zamyšlením se je dobré si uvědomit přenos složitosti do problému rozestavování budov, kde nastává větší problém, protože místo v rastru na jednom čtverci , na kterou naneseme texturu se silnicí s jiným pootočením než pravoúhlé, jasně neuvádí zda se dá na tento povrch postavit budova či ne. Označíme-li tyto čtverce jako vhodné pro umístění budov mohou budovy zakrývat část komunikaci. Naopak označíme-li tyto čtverce za nevhodné pro umístění budov nenastane žádný konflikt mezi budovou a komunikací ale výsledné město bude budit dojem kostrbatosti, domy nebudou přesně kopírovat tažení komunikace.

Vlastní řešení generování komunikací v mém projektu lze zjednodušeně popsat následovně. Podstavu celé městské struktury (v programu ground) rozděl v určitém rastru a na povrch každé takto vzniklé nové části z podstavy nanes texturu s případnou komunikací. V programu tento postup nalezneme v souborech ground.cpp a road.cpp v prvním souboru nalezneme rozdělení podstavy budovy (pole prvku) i texturování jednotlivých prvků . Jakou texturou má být mít který prvek zajišťuje druhý soubor ve kterém je procedurou MarkRoad nastaven příznak každému prvku poli podle kterého se daný prvek otexturuje typ otexturovaní se nachází v hlavičkovém souboru structure.h a z daných názvů je hned zřejmé o jakou pak asi texturu se jedná (GRASS, PAWEMENT, ROAD01_HORIZ, ROAD02_VERT….).

Základní přehled použitých textur k texturování povrchu se skládá z pěti možných textur, které jsou zobrazeny níže na následujících obrázcích.Pro představu křížení silnic jsem dodal základní textury pro jednoduché typy křižovatek.Všechny textury jsou u umístěny v projekty pod adresářem Texture/road.

Základní použité textury pro generování cest:

GRASS – textura znázorňuje městskou zeleň a lze na ni umisťovat budovy

SIDEWALK – textura tvořící malé chodníky mezi silnicemi a městskou zelení

PAWEMENT – textura znázorňující chodník

ROAD01 – textura znázorňující menší silnice s chodníky

ROAD02 – textura znázorňující větší silnice

obr. 10.1-10.5 pět základních textur

CROSS01 – textura znázorňující křižovatku silnic ROAD01

CROSS02 – textura znázorňující křižovatku silnic ROAD02

obr. 10.6 a 10.7 základní textury křižovatek

Dále již přistoupíme k samotnému zobrazování jednotlivých městských struktur. Obrázky připojené k těmto strukturám (komunikace bez budov) myslím dostatečně osvětlují řešení jak a kam jsou jednotlivé textury nanášeny mimo to předpovídají rozmísťování budov.

6.2.1– Centra, osvětlení kódu programu a jeho napojení na graf scény

Základní městkou strukturou je centrum. Tato struktura má za úkol zobrazovat dva druhy center. V prvním případe si nejlépe pod slovem centrum představíme Americká městská centra, kde se na malém prostoru těsní nespočet budov a čím jsou centru města blíže tím je jejich výška vetší. Myslím si že zde je nejvyšší čas opět zabloudit do kódu části programu a vysvětlit si na jednom příkladu jak jsem tvořil struktury mest a jejich napojení na graf scény. Podobně jako tomu bylo při vysvětlování tvorby budov. Zde mi ovšem nejrychlejší a nejefektivnější způsob připadá doplnit kód,o pár vet které rychle poskytnou přehled, případným zájemcům o kusy kódu.

```
//=============================================================
// soubor center00.cpp
//=============================================================

#include "..\structures.h"
```

Hlavičkový soubor obsahuje definice funkcí potřebné především k výběrům budov podle jejich rozměrům.Napsal jsem hodně funkcí jak vybrat budovu ale jako nejvíce potřebné jsem zjistil že by postačovaly dvě možná jedna. Jsou to fce. na výběr budovy podle dané podstavy x,z do maximální výšky y, nejelegantnější fce. pro výběr budov jsem napsal po-té co jsem zjistil že vždy potřebuji vybrat nějakou budovu podle vždy jiného rozměru x,y,z od do určitých mezí. Dále zde nalezneme fce. pseudonáhodného výběru či jen posunu jako textury a budovy. Nejdůležitější fce. zde je samozřejmě fce. pro umístění budovy.

```
POSITION center00;

// ---------------------------------------------------------------------------
// fce.vytvori mestskou strukturu center00
// ---------------------------------------------------------------------------

SoSeparator* MakeCentre00() {

        center00.x_min = 0;
        center00.z_min = 0;
        center00.x_max = MAX_GROUND;
        center00.z_max = MAX_GROUND;
```

Funkce MakeCentre00 nám vrací centr00 který si zde vytvoříme. Tedy k poslednímu uzlu scény napojíme centr00 na který jsou napojeny jednotlivé budovy viz.níže a vše co chceme aby tato struktura obsahovala.

```
        SoSeparator* center00 = new SoSeparator;
```

Význam jednotlivých proměnných lze odvodit od názvu pro přesnost v kostce doplním. Alfa – úhle otoční jednotlivých budov struktury, Building – aktuální číslo budovy od které se vybírá další budova (malá náhodnost aby se budova opakoval co nejméně), x a z jsou čísla určující polohu budovy na povrchu struktury, x1,x2…z1,z2 – jsou proměnné určující povolený interval daného rozměru budovy, pom_x a pom_z pro přechování hodnoty x,z (budovy se umísťují v cyklu jednotlivé polohy jsou přibližné a dochází k jejich malým úpravám podle toho kam chceme budovu umístit, x,z si před těmito změnami uložíme po umístění budovy jim z pom_x a pom_z navrátíme původní hodnoty) , texture – je typu TEXTURE jsou čtyři typy cihlová, barevná, kamenná a dřevená podlé této hodnoty se bude na budovu nanesena příslušná textura.

```
float alfa=0;
int building=0;
float x=1.f;
float z=1.f;
float y1=2.0f;
float y2=2.5f;
float x1=0.5f;
float x2=0.5f;
float z1=0.5f;
float z2=0.5f;
float x_pom,z_pom;
TEXTURE texture = COLOR;
```

Zde probíhá nejtěžší část tvoření městské struktury centrum. V podstatě umisťujeme na povrch struktury dané budovy aby budily zdání reality a zároveň nedocházelo ke kolizím (hledáme jim vhodnou lokalitu k umístění x,z a vhodným otočení budovy alfa) . V těchto strukturách je kladen důraz na vybuzení dojmu reality a kolize je řešena přesným nastavením. V náhodné městské struktuře je tomu ne naopak ale přibližně tam je kladen důraz na vyhnutí se kolizím a vybuzení dojmu reality je druhořadé. Také o tyto kolize u náhodné struktury je o podstatný kus kódu zvětšen kód náhodné struktury.

```
for(int i=1; i<=6; i++, x+=2.8f) {
  for(int j=1; j<=6; j++, z+=2.8f) {
```

Jak již bylo řečeno uložíme si pozici před její úpravou.

```
x_pom=x; z_pom=z;
```

Dále v cyklu následuje kus kódu, který se v podstatě zameruje na určitou nebo určité pozice budov a na těchto upravených pozicích volí vhodné rozměry budov. Proto si

ho uvedeme jen ve zkráceném tvaru. Pro osvětlení tvoření mostských struktur nás už bude jen zajímat konec cyklu.

```
if(i>2 && i<5 && j>2 && j<5) {y1=4.0f; y2=4.5f;}
else
…
…
…
if(j==6 && i==2) {x+=0.5f;}
if(i==1 && j==1 || i==1 && j==6) x+=0.5f;
if(i==6 && j==6 || i==6 && j==1) x-=0.5f;
```

Konečné po všech doladění pozice vybereme vhodnou budovu k umístění na tuto pozici. Její texturu a uzel budovy připojíme na uzel centra tím se daná budova ocitne na připravené pozici.

```
building = SB_from_to_xyz(building,x1,x2,y1,y2,z1,z2);
texture = NextTexture(texture);
center00 -> addChild(PutBuilding(building,x,z,alfa,texture));
```

Nastavíme opet počáteční hodnoty příslušným pramenným a postup opakujeme

```
        x=x_pom;z=z_pom;
        alfa=0;
        y1=2.0f; y2=2.5f;
        x1=0.5f; x2=0.5f;
        z1=0.5f; z2=0.5f;
        }
z=1.f;
//texture = NextTexture(texture);
}
```

```
center00 -> addChild(PutBuilding(27,8.5f,8.5f,degree_to_radian(10),texture));
```

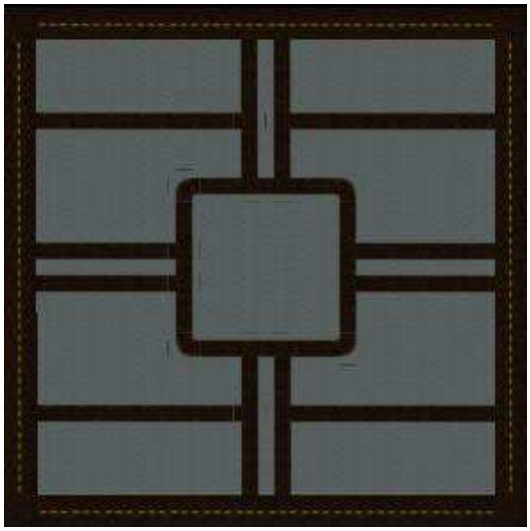
Nakonec vrátíme uzel centra který se naváže bud ke generovanému městu nebo v případe prohlížeče městských struktur přímo na uzel root.

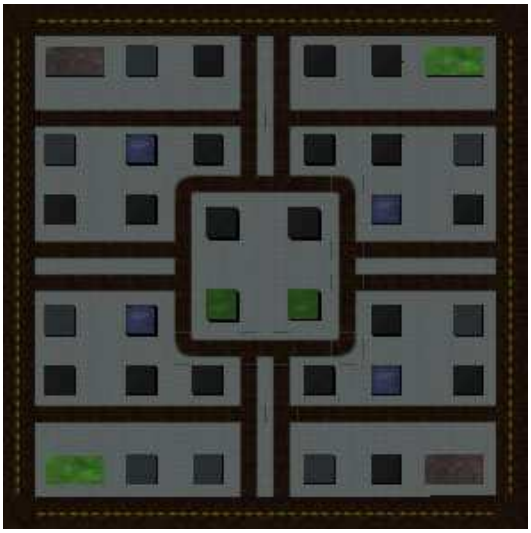
```
return(center00);
}
// ---------------------------------------------------------------------------


// ===========================================================================
// konec souboru center00.cpp
// ===========================================================================
```

Městský urbanismus je složitá struktura zahrnující hmotné i psychické hodnoty.Mezi hmotné radíme budovy,komunikace,zpevněné plochy, zelen,vodní prvky,technickou infrastrukturu a další. Psychické hodnoty města jsou utvářeny prostorovým uspořádáním,které je tvořeno rozmístěním hmotných prvku jejich tvarem, výškou ,polohou a kompozicí utvářející.Podle popsaných kritérií je mostská zástavba dělena na :centra, předměstí , nákupní zóny, čtvrtě monofunkční či polyfunkční…atd.

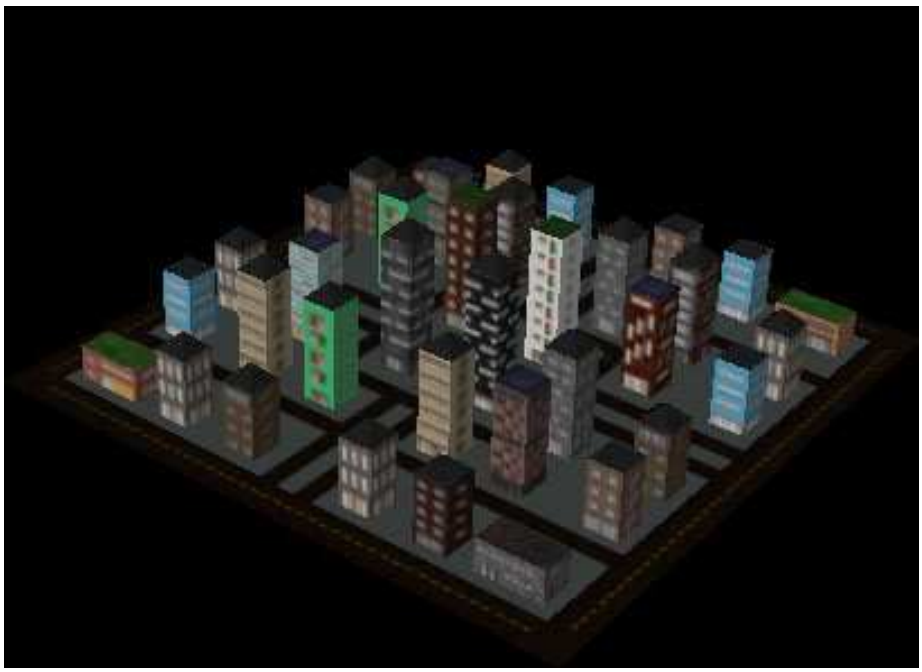
Hlavní tepnou městské struktury je centrum(náměstí,ulice), které si můžeme jednoduše představit jako historickou zástavbu vymezenou bloky nebo novodobější centrum tvořeno samostatně stojícími budovami ,budovami vytvářející uliční frontu , a z části opět blokovou strukturu.

Na vložených obrázcích jsem se snažil vytvořit několik typu center. Novodobé centrum jsem zobrazil jako výškovou zástavbu s pravidelnou sítí komunikací.V centrální pozici jsem umístnil největší budovy , které svou dominancí poutají pozornost , zlepšují orientační smysl a jasně zvýrazňují pozici centra.Centrální parcely jsou velmi cenné a tak je centrum má velmi nezvyklý poměr plochy parcely a zastaveného objemu. K centru vedou všechny významné komunikace .Ostatní budovy jsem rozmístnil šachovnicovým způsobem s vzrůstající vzdáleností jsem snižoval výšku zástavby.

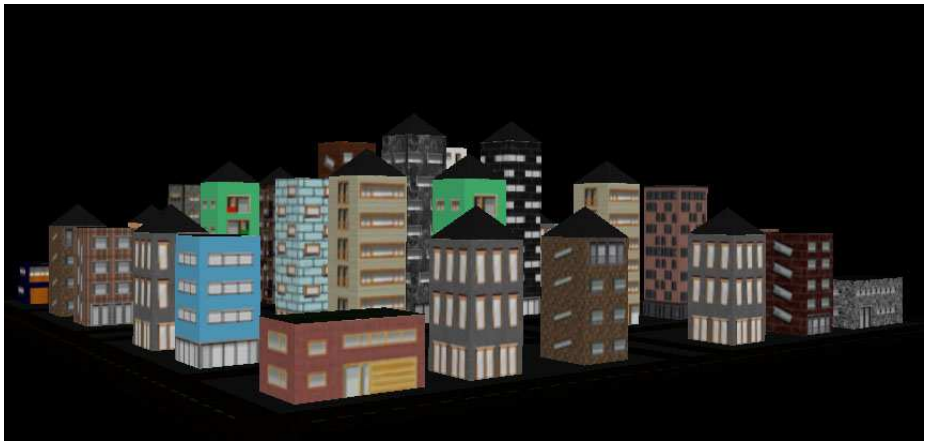




obr.10.8 a 10.9 centrum1 – nalevo silniční sít, napravo doplněna zástavbou



obr.11.0 centrum1 – náhled

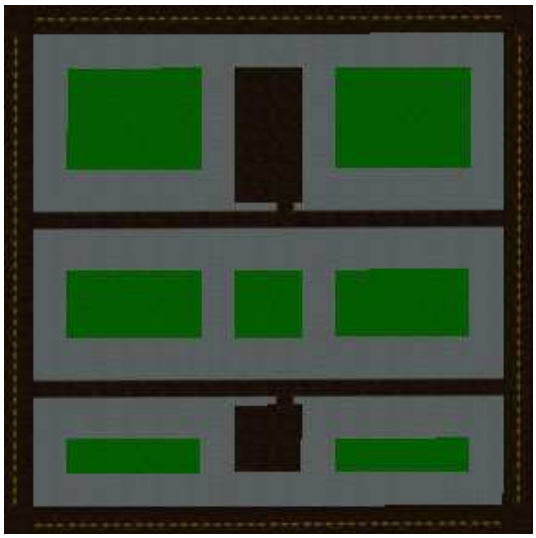


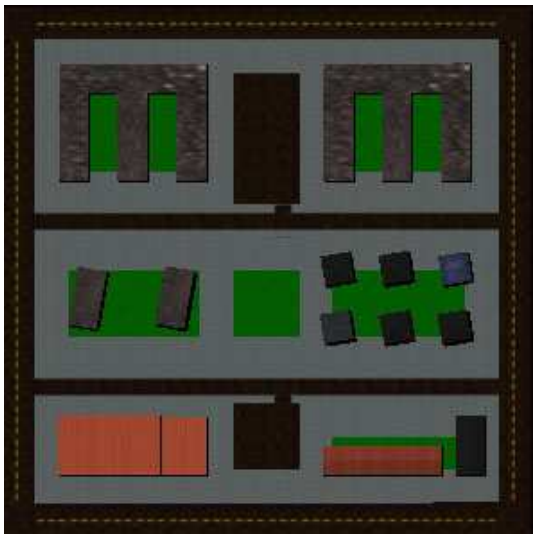
obr.11.1 centrum1 – přiblížení

Jako další typ centra jsem zobrazil také novodobý model který tvoří centra menšího typu.Toto centrum se většinou vyskytuje v jednotlivých čtvrtích města , kde je tvořeno povětšinou základní veřejnou vybaveností jako jsou knihovny, školy,obchodní centra.Je zde oproti prvnímu typu zahrnut vetší podíl bydlení a drobného podnikaní.

Zvláštním typem menšího centra je pak monofunkční centrum které se uplatňuje ve vojenských areálech, vysokoškolských komplexech,nemocničních areálech které je svým způsobem také nezávislé a můžeme je na základe toho nazvat také centrem.

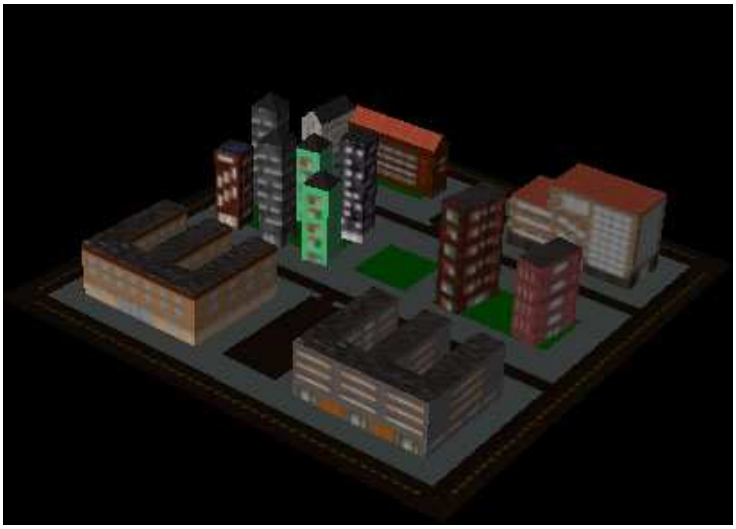
Na obrázcích je ukázka menšího typu centra.Oproti prvnímu jsem ho vymezil silniční strukturou tvořící uprostřed jaký si typ bulváru kolem kterého jsou budovy rozmístněny. Nejedná se už pouze o výškové stavby,ale i o stavby s rozlišnou výškou. Použil jsem i reprezentativní budovy půdorysně znázorněné do písmena e, které tvoří veřejnou vybavenost. V mém případe jsem si představoval banky, školy, pošty, a jiné úřady.





obr.11.2 a 11.3 centrum2 – nalevo silniční sít, napravo doplněna zástavbou

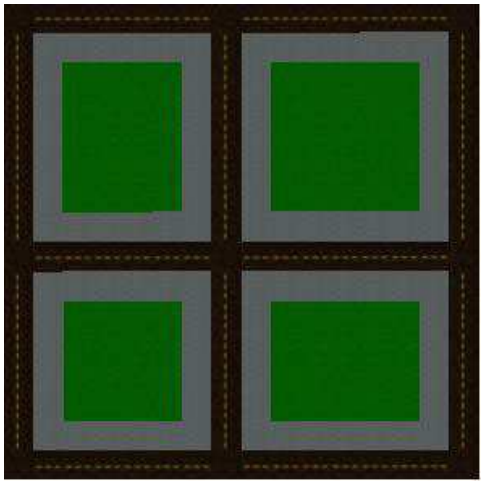
obr.11.4 centrum2 – přiblížení

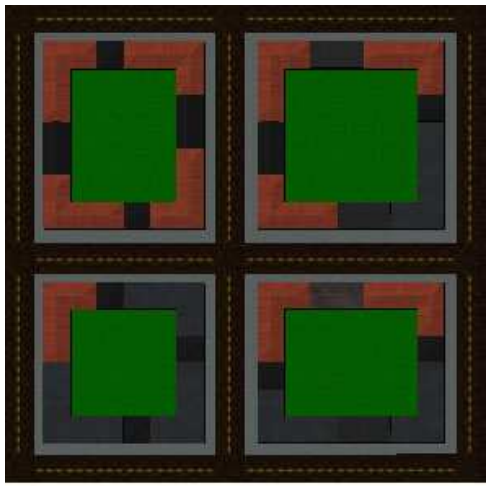


obr.11.5 centrum2 – náhled

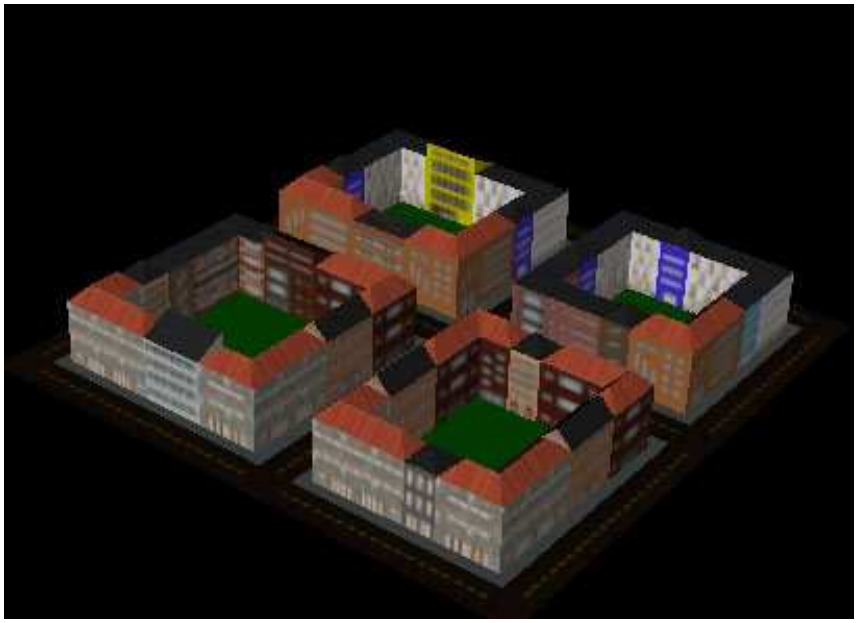
6.2.2– Čtvrtě

Čtvrť je jakýsi částečné závislý element na městském jádru avšak každá čtvrť si vytváří svoje centrum popsané na předešlé struktuře. Vytvořil jsem čtvrť definovanou městskými bloky. Pravidelný rastr silnic vymezil pravidelné bloky které mají vesměs živý parter a ve vyšších patrech je zpravidla bydlení. Výška zástavby se u mojí čtvrtě zvlášť neodlišuje , což je hodně blízko skutečnému utváření těchto obytných čtvrtí s polyfunkčním parterem. Mají vesměs předepsanou výšku která je určena okolními domy v bloku a liší se řádově jedním až dvěmi podlažími a závisí na rychlosti a stáří výstavby.Uvnitř bloku je situován vždy hospodářský dvůr, nebo zelen pro obyvatele bloku. Jedná se tedy o soukromý prostor.Prostorové uspořádání působí celistvě, snadnějším rozdělením parcel je jasně definován typ prostoru.

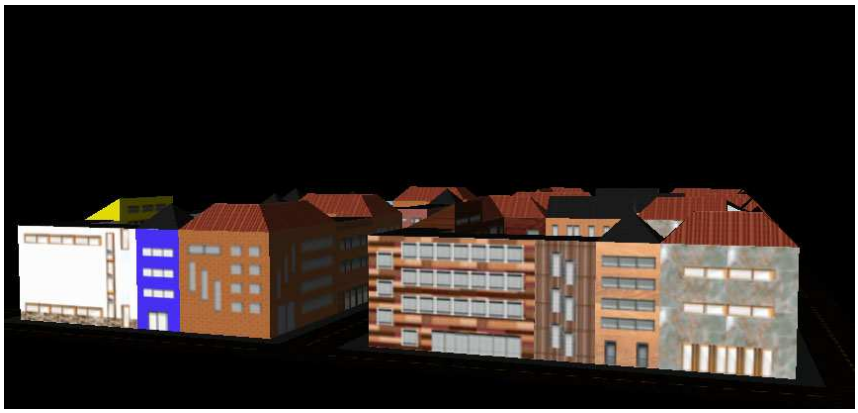




obr.11.6 a 11.7 čtvrť – nalevo silniční sít, napravo doplněna zástavbou



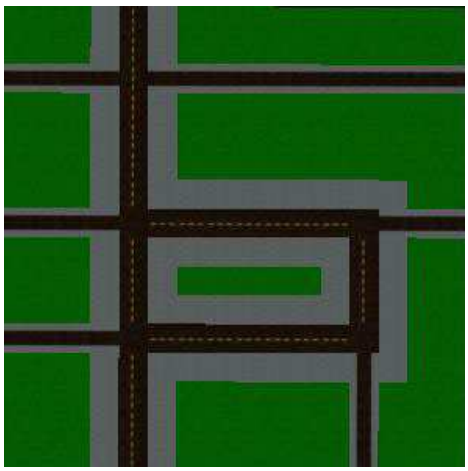
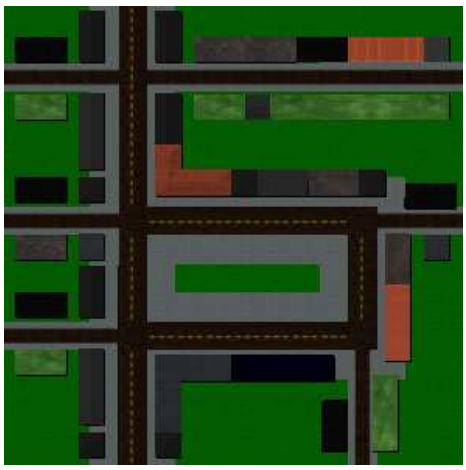
obr.11.8 čtvrť – náhled



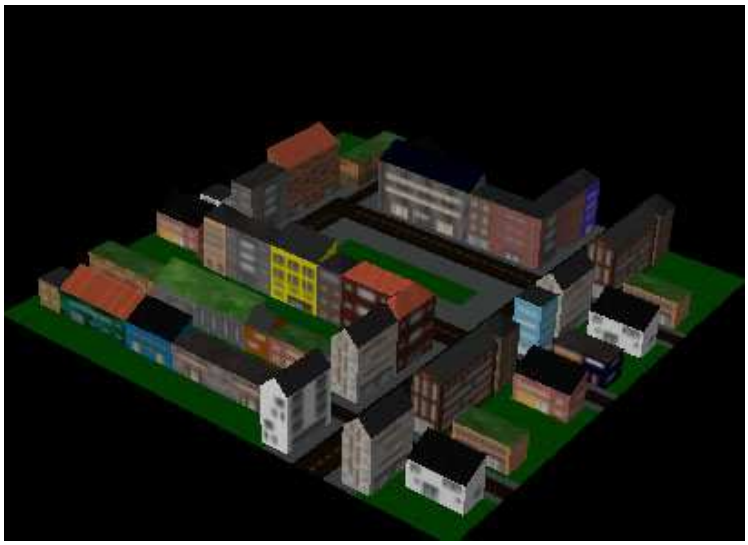
obr.11.9 čtvrť – přiblížení

6.2.3– Náměstí

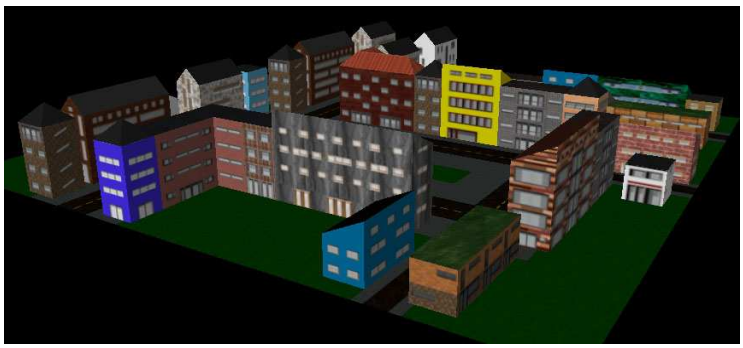
Náměstí je dalším typem centra muže být jednak půdorysné,dopravně či funkčně odlišeno.Na obrázku je zobrazeno náměstí obdélníkového uspořádání (bežne se vyskytují náměstí kruhová, čtvercová, trojúhelníková, mnohoúhelníková, nepravidelná či liniová) , které je dopravně protknuto a jehož okolní zástavba je tvořena ulicemi solitérními zástavbami. Uprostřed jsem situoval zelenou plochu, která by sloužila jako park na náměstí. Zajišťovala by relaxaci. Další variantou je vydláždění plochy náměstí a vložení vodních prvku, soch, atd.

obr.12.0 a 12.1 náměstí – nalevo silniční sít, napravo doplněna zástavbou



obr.12.2 náměstí – náhled

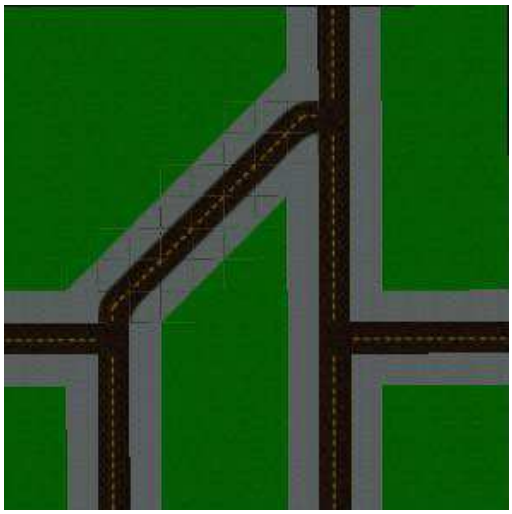


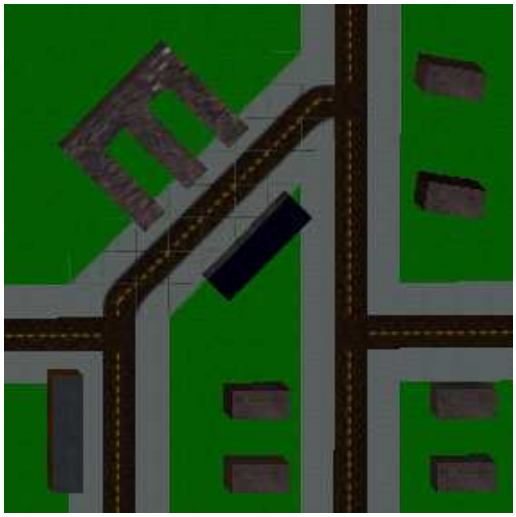
obr.12.3 náměstí – přiblížení

6.2.4– Předměstí

Předměstí je situováno daleko od centra města. Vzniklo jednak díky hladu po nových a rozsáhlých parcelách, které jsou v centru nedostatečné a finančně nadsazené. Předměstí jsou svoji polohou vhodné pro velkoobchody, je zde situován také průmysl a bydlení (vily, domy s dostatečným pozemkem, sídliště).

Předměstí přitahují investory i lidi, což má za důsledek postupné vyklidňování center mest které jsou substituovány např.:shopingparky. Z hlediska využití parcely je většina zelená louka, potřebné zpevněné komunikace a nadbytek silničních komunikací. Předměstí se většinou vytváří na dálničních výpadech nebo na křižovatkách silnic.Přitahují k sobe lidi cestující dnes stále často automobilem. Dalším typem předměstí jsou klidné ulice s obytnými domy s řídkou zástavbou.





obr.12.4 a 12.5 předměstí – nalevo silniční sít, napravo doplněna zástavbou

obr.12.6 a 12.7 předměstí – nalevo náhled, napravo přiblížení

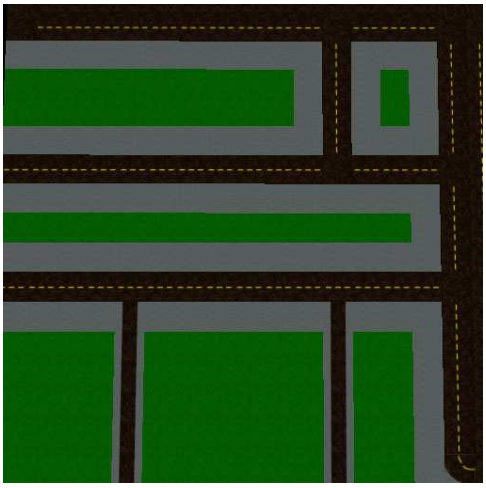
6.3 – Generování náhodné městské struktury

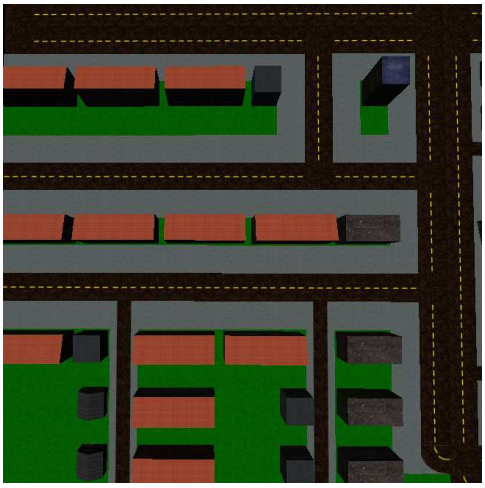
Další městskou strukturou je náhodná městská struktura. Tuto strukturu nelze z předchozího dělení zařadit. Úkolem náhodné městské struktury je snaha plynule navázat na předchozí typy struktur.

Generování náhodné městské struktury lze rozdělit do dvou kroku:

1. Generování cest – vytvořit vhodnou síť cest která navazuje na okolní městské struktury.

2. Generování budov – zastavět strukturu budovami, aby co nejvíce splynula s okolními městskými strukturami





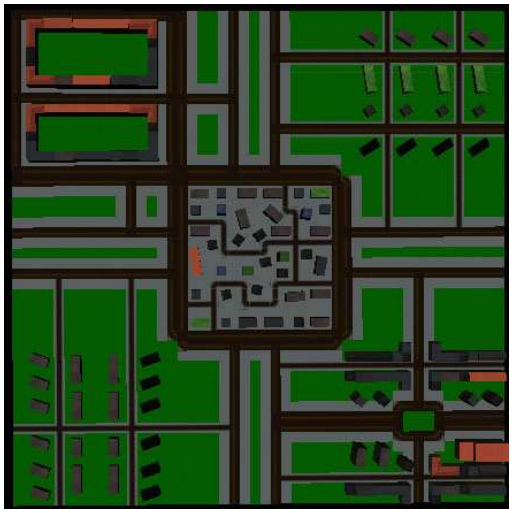
obr.12.8 a 12.9 Výřez náhodné struktury – nalevo silniční síť, napravo doplněna zástavbou

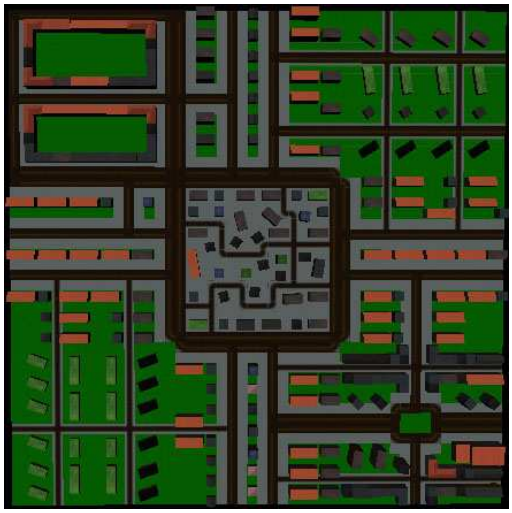
Generování sítě cest se skládá ze tří hlavních kroku. V prvním kroku si náhodná městská struktura převezme od svých okolních městských struktur sousedící okraje tím je zajištěna návaznost, přitom ještě počítá kolik silnic vstupuje a z jaké strany do náhodné městské struktury a provede následné ošetření rohu.

V druhém kroku se pokusíme přetnout náhodnou městskou strukturu ve směru od centra města ven. Nenachází-li se zde žádné vstupní silnice přidáme si je. Tímto krokem jsme si zajistily návaznost pro ostatní silnice vstupující z jiných stran. V posledním kroku napojíme tyto zbylé silnice na silnici-ce které přetínají náhodnou městskou strukturu.

Umísťování budov na náhodnou městskou strukturu odpovídá průchodu náhodnou městskou strukturu a zjišťování vhodné lokality (u cesty na městské zeleni nesmí nad ní v struktuře již být postavena budova) pro umístění budovy od největší po nejmenší rozměrově.

obr.13.0 a 13.1 Náhodné struktury – napojení na připravené městské struktury,
nalevo pohled protínající silnicí ve směru k centru města,
napravo udržení rázu ulic,
(dvě rovnoběžné silnice směrem doprava nahoru,
odpovídají naším návazným silnicím od centra města ven,
s principem přetnutí náhodné městské struktury)





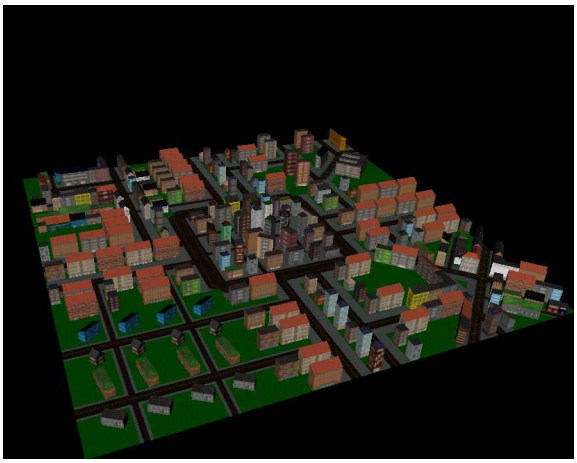
obr. 13.1 a 13.2 Náhodné struktury v městě – nalevo jen silniční síť
odhalující pozice náhodných městských struktur, napravo doplněny zástavbou

6.4 – Spojení městských struktur, vznik města

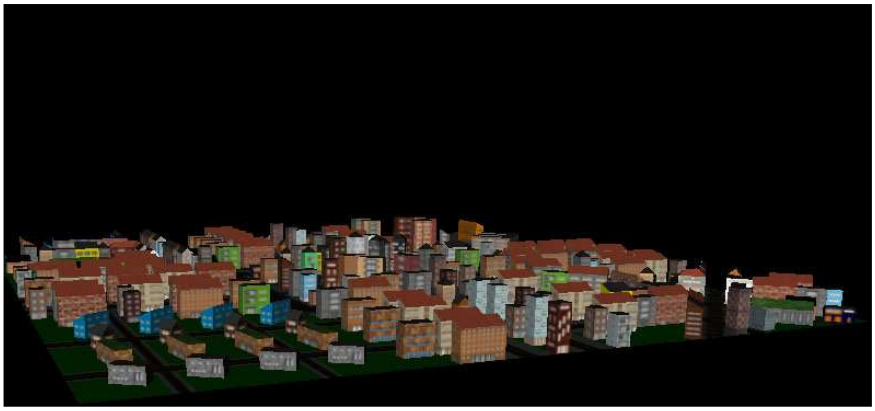
Propojením devíti struktur města na ploše z toho, pět připravených a čtyři generované vznikne město. Myslím si že následující obrázky není třeba komentovat. Za zmínku stojí že v původním zamýšleném plánu jsem počítal s městem 5x5, bohužel můj osobní počítač (AMD Athlon 2000+ s vestavenou grafikou a 512RAM) tuto zátěž textur unese jen ve stavu 3x3. Ale bez textur není problém město roztáhnout viz.příloha.

Poslední poznámka pro ty kteří by chtěli přece jen zvětšit počet struktur města musí obohatit generování silnic v náhodné struktuře v druhém kroku (protínající silnici od centra ven), protože by tato silnice nevedla ven ale na další strukturu. Moje představa byla ‚najít na této venkovní straně a straně k centru silnice, pokusit se napojit prostrení silnice z nich ostatní v tomto směru zatím nechat, pak postupovat jako v třetím kroku (napojit na tuto silnici kolmé silnice), no a naposled napojit zbylé silnice na silnice tyto silnice.

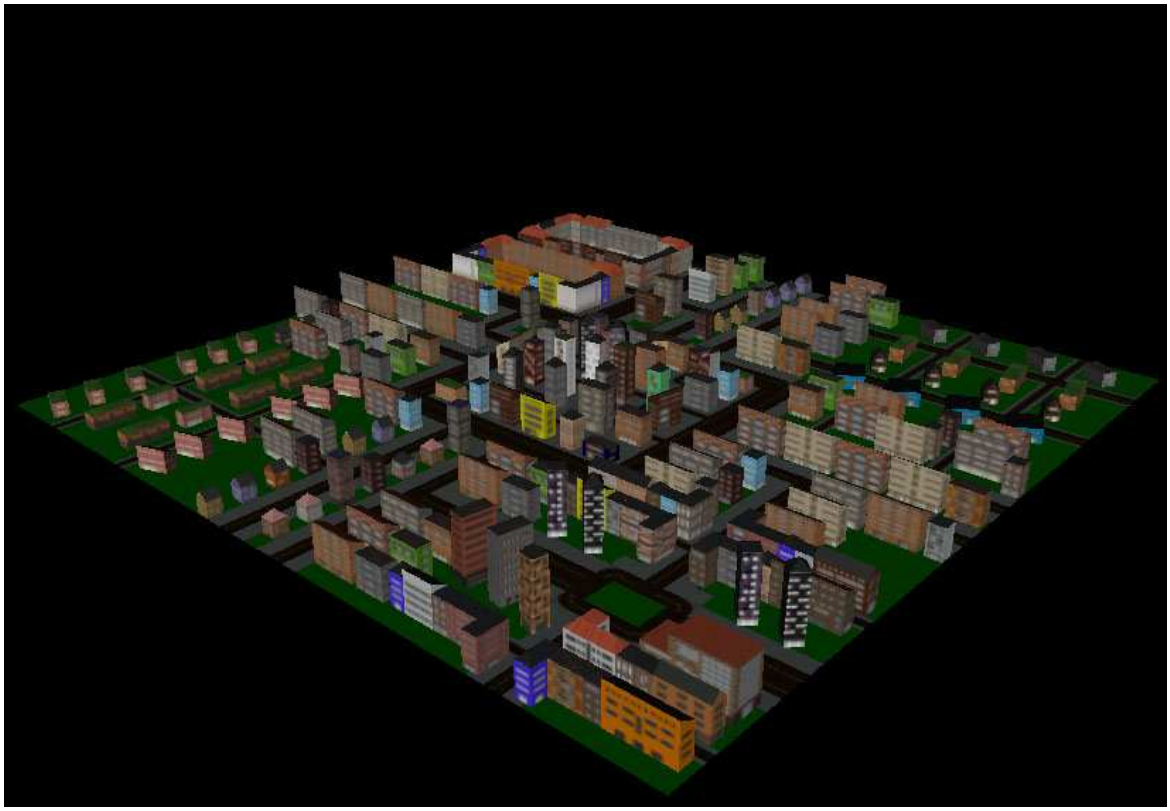
V programuje je toto napojovaní silnic řešeno tak že silnice postupuje směle vpřed ve svém směru a když narazí na kolmou silnici v jejím směru napojí se na ni křižovatkou.



obr.13.3 městská zástavba – náhled



obr.13.4 Městská zástavba – Přiblížení



obr.13.5 Virtuální městská zástavba

7. Kapitola – Vyhodnocení dosažených výsledků

7.1 – Hodnocení programu

V prvním kroku je nejlépe se pokusit udělat malé ohlédnutí po programu a nesnažit se ho hned uchopit jako celek ale postupně odkrývat jednotlivá zákoutí.

Výsledný program nabízí čtyři možnosti:

1. Prohlížeč modelů budov
2. Generování virtuální městské zástavby na členitý terén
3. Prohlížeč městských struktur
4. Generování virtuální městské zástavby s komunikacemi

Proto nejdříve postupně v následujících podkapitolách provedu ohodnocení takto rozdělených částí programu, kterým odpovídá i první čtyři přílohy. V každé části také ihned uvedu možné problémy se kterými jsem se setkal, cesty řešení, nápady ,připomínky, varování, a hned i rozšíření.

7.1.1 - Hodnocení budov

V programu se nachází pětatřicet tvarem odlišných modelů budov s možností nanesení textur. Textur pro nanesení na jeden model budovy jsou čtyři (podle fasády rozděleny na BRICK,COLOR,STONE,WOOD tedy cihla,barva,kámen a dřevo). Dále lze budovy dělit podle střech na (FLAT,SADDLE,PYRAMID,AISLE) tedy na plochou, sedlovou, pyramidovou a šikmou nebo-li pultovou. Rozměry budov jsou odvozeny od tvaru dané budovy počínaje od tvaru krychle až po budovy tvaru L a E písmen. Ukázku jednotlivých budov naleznete v první příloze.

Jak každý ví pro generování města je nejlepší mít připravených co nejvíce modelů budov popřípadě generátor budov. Chtěl bych upozornit ž e navázání generátoru budov by nebil u mě velký problém jakožto přidávání modelu budov v souboru. Jediné o co by jsme se museli postarat je přidání rozměrů budovy.

Jinak si myslím že se budovy v zástavbě moc často neopakují a jejich počet je vyhovující. Zde bych chtěl upozornit na dva malé problémy textur a tvaru budovy, které lze lehce přehlédnout. Velmi doporučuji dodržovat velikosti textur budov ve velikosti řady binární posloupnosti (2,4,8,..) v pixlech knihovna Coin jinak hned převádí rozměry k nejbližší nižší z této řady (rychlost) a také nepřehnat velikost mé textury začínají od

32x32, dobrou radou je dodržovat při kreslení textur jednotné výšky pater o to více je pak dojem reality větší především v těsném navazovaní budov při tvorbě ulice.

Shrnutí hodnocení budov, počet různých budov které lze vytvořit je 35x4 tedy 140 různých budov. Čas potřebný k vytvoření jedné budovy cca.hodina na tvar a dvě hodiny na texturu u tvarově jednoduchých budovách. Vytváření modelů budov je časově náročná práce.

7.1.2 - Hodnocení generování virtuální městské zástavby na členitý terén

Výsledný algoritmus umisťuje budovy do terénu se složitou morfologií . I přes složitost terénu se podařilo vytvořit iluzi města v mém případě města s bodovou zástavbou. Snažil jsem se vytvořit město soudobé, jehož budovy formují městskou strukturu. Centra svojí výškovou gradací připomínají americké downtown, které také vzniklo bez širších historických souvislostí.

Omezením pro můj záměr i přes to byl složitý terén, který není až tak vhodný pro zakládání měst. V historii bych zmínil nejvýše položené město Brazilia. Většina měst byla založena na nížině čí mírné pahorkatině. Dále můj terén měl velikost 128x128, což mě při rozšiřování lokality pro městskou strukturu značně omezilo.Tento nejzávažnější problém jsem osvětlil v kapitole 6.1.

Možná že někomu při testování programu nepřijde výsledek moc pěkný ale je dobré si uvědomit že velikost a tvar města závisí na modelu vygenerované krajině.a také při ohlednutí se do kapitoly není moc programů řešící tento problém (GRASS,SimCity4). V druhé příloze naleznete mé nejpěkněji vygenerované virtuální městské zástavby na členitý terén.

7.1.3 - Hodnocení městských struktur

V programu se nachází šestnáct předem připravený městských struktur. Každá z nich obsahuje jinou silniční síť spolu s chodníky. Je dobré si uvědomit že i tyto textury osahují kousek náhodnosti a to rozmístění infrastruktury je stejné ale budov vyhovujících k umísťování na dané pozice může být víc (nejčastěji jiné výšky) a i kdyby se na pozici umísťoval budova jednoho tvaru její možnost nanesení textury ze čtyř připravených dostatečně mění ráz zástavby.

Dále bych zde chtěl poukázat na dvě připravené městské struktury, ze kterých vycházejí silnice pod pětačtyřiceti stupňovým úhlem tyto struktury nepřidávám do výsledného města ale není problém je tam dodat. Problém je vyřešení návaznosti s náhodnou městskou strukturou, tento problém je popsán na str.38 (narůstající složitost)

v porovnání s ostatními projekty v úvodu si nemyslím že by si moje silniční sítě stály špatně.

Samozřejmě že přidáváním dalších textur znázorňují různé komunikace od jiné trávy, chodníku a silnic dopřejí městu opět větší rozmanitost jak je tomu stejně i u budov ale v mém městě o rozměru 3x3 městských struktur je na velmi dobré úrovni.

Dobré je se zde také zmínit o náhodné struktuře, a její zajímavé možnosti rozšíření městských struktur a to ne předem připravených ale náhodné. Myslím si že i v mém případě měst 3x3 by výsledný dojem byl ještě více reálnější u větších měst by se tímto mohla částečně rozbíjet opakování principů algoritmu náhodné městské struktury které se při větším počtu začnou odhalovat.

Tvorba městských struktur ač se to nezdá je hodně časově náročná činnost. Jestliže si přejeme, aby naše struktury budily dojem reality. No možná nejméně časově náročné je tvoření předměstí a nejvíce pak náhodná struktura, přesto si myslím že to není vyhozený čas. Konečný výsledek mě vždy mile překvapil. O vzhledu městských struktur si nejlépe uděláte představu po prohlédnutí třetí přílohy. Nejlépe pak prohlédnutím v samotném programu kde si strukturu natočíte a přiblížíte libovolně dle potřeby.

7.1.4 - Hodnocení generování virtuální městské zástavby s komunikacemi

Nejlepší zhodnocení této části poskytuje čtvrtá příloha kde jsou znázorněny výsledky po generování virtuální městské zástavby s komunikacemi. V páté příloze naleznete město 5x5 bez textur, které bylo původně zamýšleno viz.kapitola 6.4.

Osobně mě zde mrzí dvě věci ošetření návaznosti rohů povrchu městské struktury a že není město větší. Jestliže by někdo pokračoval v této zástavbě a neměl technické problémy s větším městem (moc textur). Pak si myslím že posledním velkým přínosem by bylo doplnit komunikační síť o tramvajovou linku nebo nadzemní dráhu.

7.2 – Shrnutí hodnocení programu

Na celý program se lze podívat ze dvou směrů. V prvním směru bych chtěl poukázat na robustnost objemu dat mám na mysli jak souboru programu tak textur, řekl bych že je to dobrý odraz dvouleté práce (její rozdělení viz. Kapitola Úvod). A naproti tomu jsme se snažil program vyvíjet jako jednoduchou stavebnici. Nejlépe je tento fakt patrný na obrázku grafu programu v poslední příloze.

Použitá literatura :

[1] Kolektiv autorů: Domovské webové stránky firmy System in Motion
http://www.sim.no/

[2] Kolektiv autorů: Domovské webové stránky Coin3D
http://www.coin3d.org/

[3] Kolektiv autorů: Stránky dokumentace Kolektiv autorů: k Coin3D
http://doc.coin3d.org/

[4] Kolektiv autorů: Domovské webové stránky firmy SGI
http://www.sgi.com/

[5] Kolektiv autorů: Stránky dokumentace Open Inventoru
http://oss.sgi.com/projects/inventor/

[6] Tutoriál k Open Inventoru na
www.root.cz

[7] Ročníkový projekt Generátor Virtuální městské zástavby, autor: Bc.Jan Kytýr

[8] The Inventor Mentor, autor :Josie Wernecke

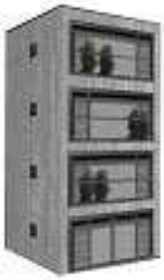
příloha č.1

p0

p1

p2

p3



p4

p5

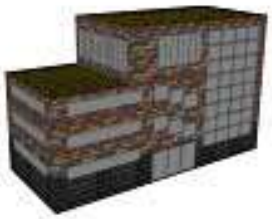
p6

p7

p8

p9

p10



p11

p12

p13

s0

s1

s2

s3

s4

s5

s6

s7

s8

s9

s10

s11

s12

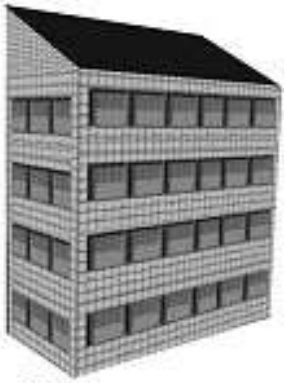
s13

s14

s15

s16

s17

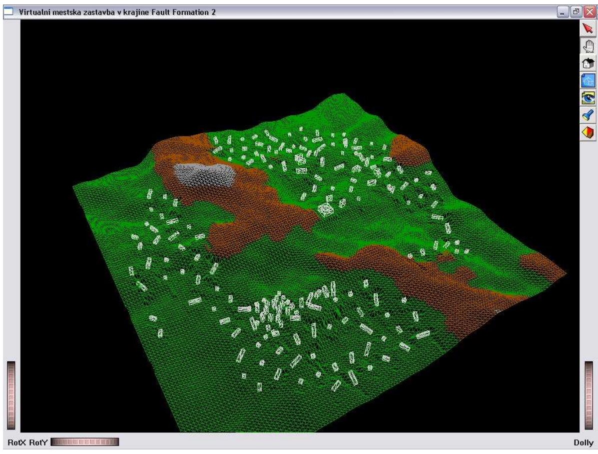


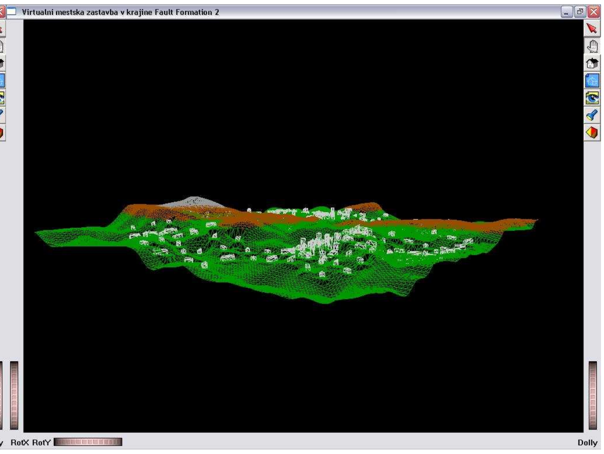
s18

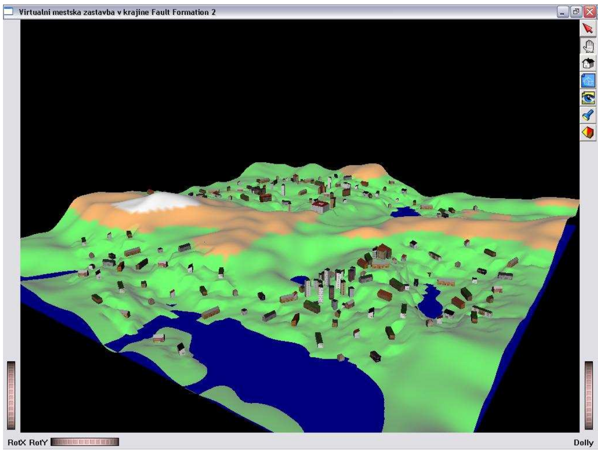
s19

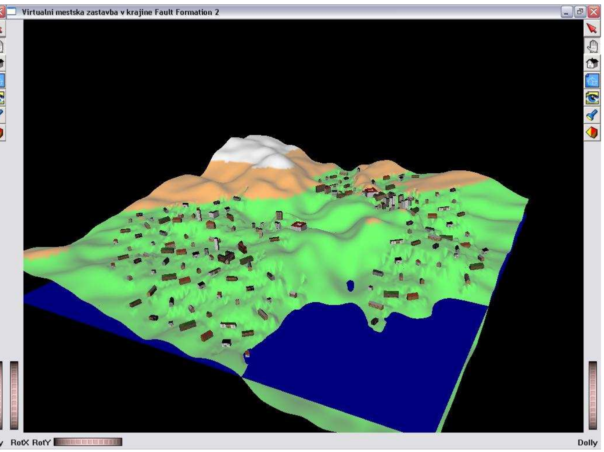
s20

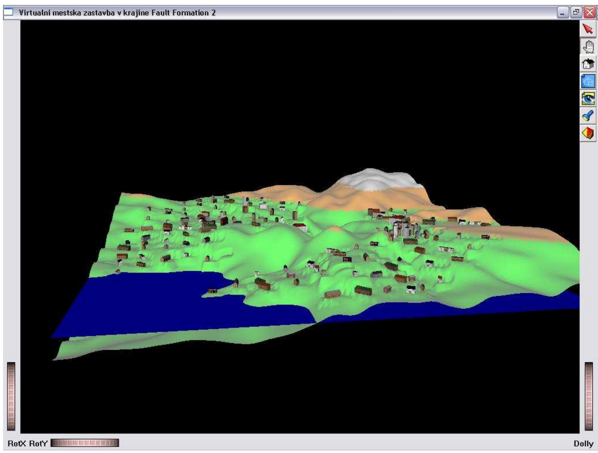
příloha č.2

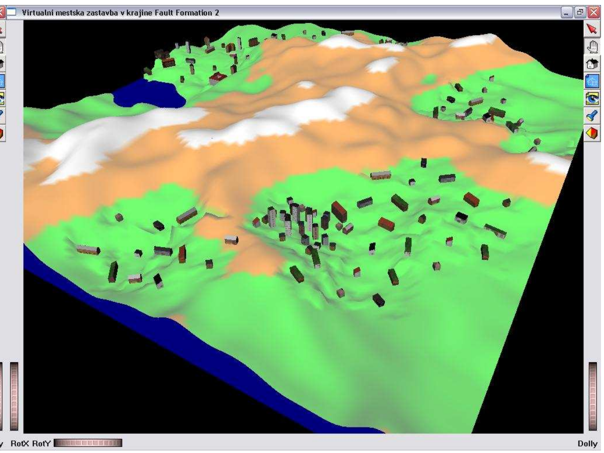

Virtualni mestska zastavba v krajine Fault Formation 2
RotX RotY
Dolly


Virtualni mestska zastavba v krajine Fault Formation 2
RotX RotY
Dolly

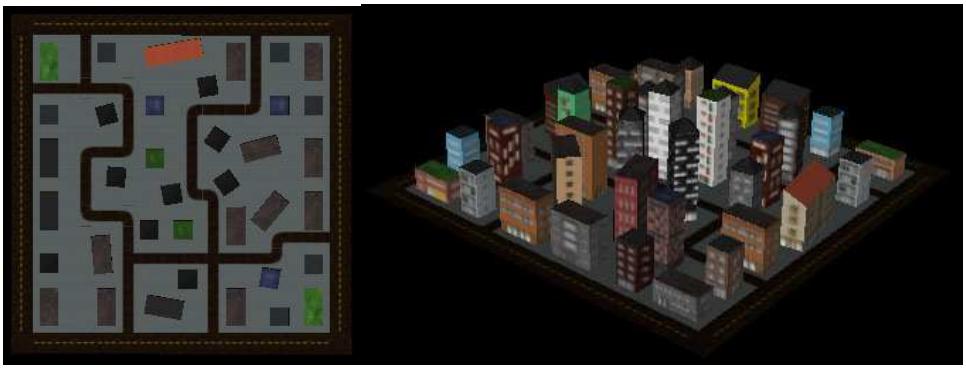

Virtualni mestska zastavba v krajine Fault Formation 2
RotX RotY
Dolly

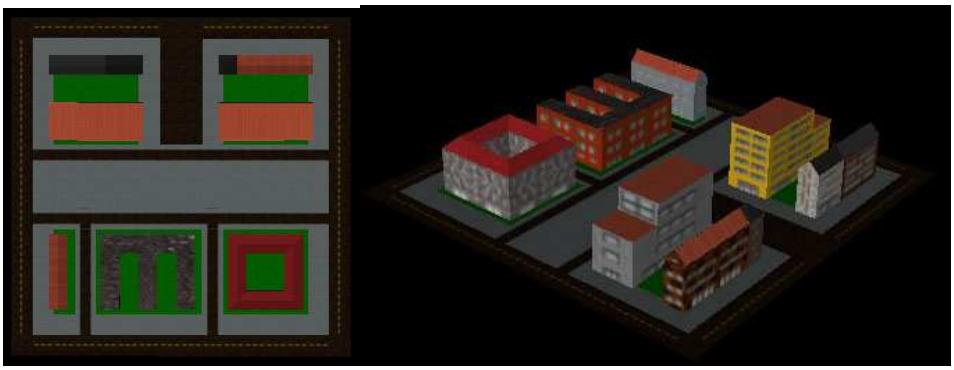

Virtualni mestska zastavba v krajine Fault Formation 2
RotX RotY
Dolly

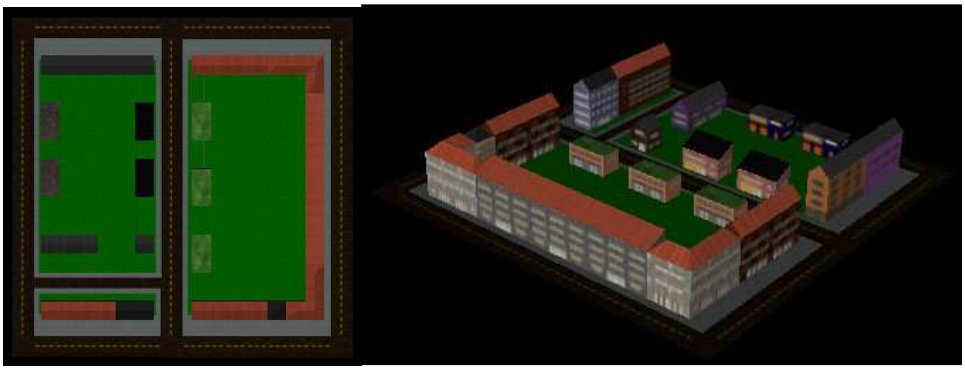

Virtualni mestska zastavba v krajine Fault Formation 2
RotX RotY
Dolly

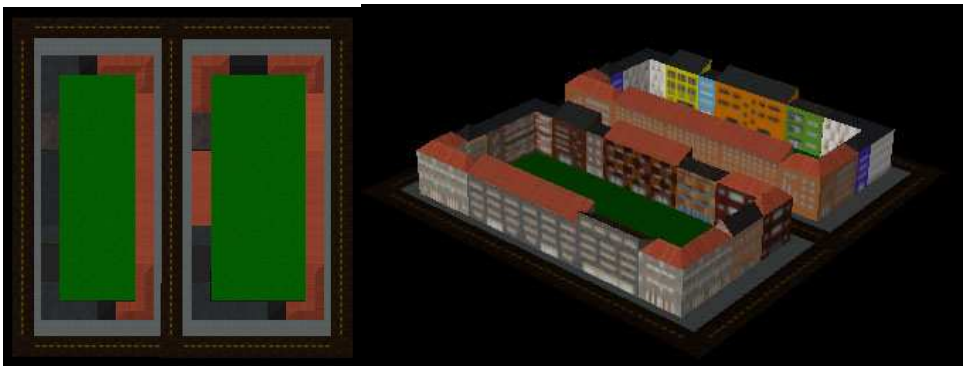

Virtualni mestska zastavba v krajine Fault Formation 2
RotX RotY
Dolly

příloha č.3 centre00, centre03, disrict02, distric03

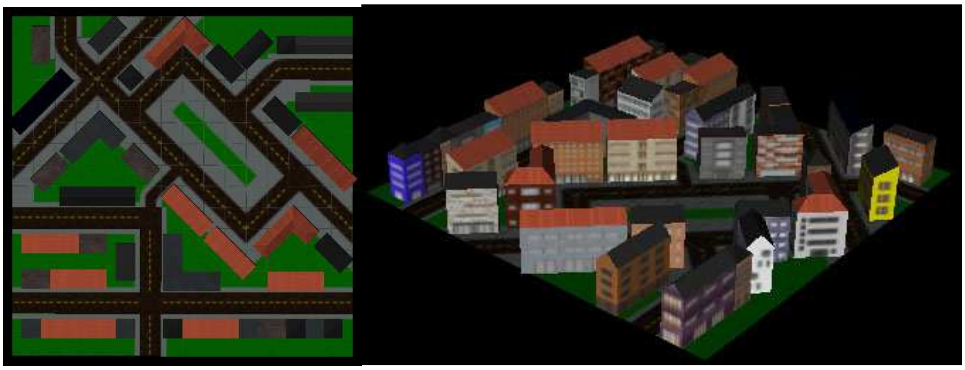


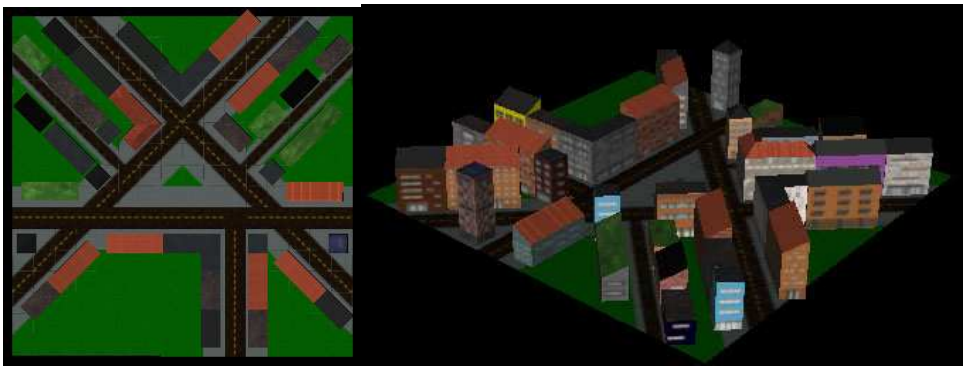


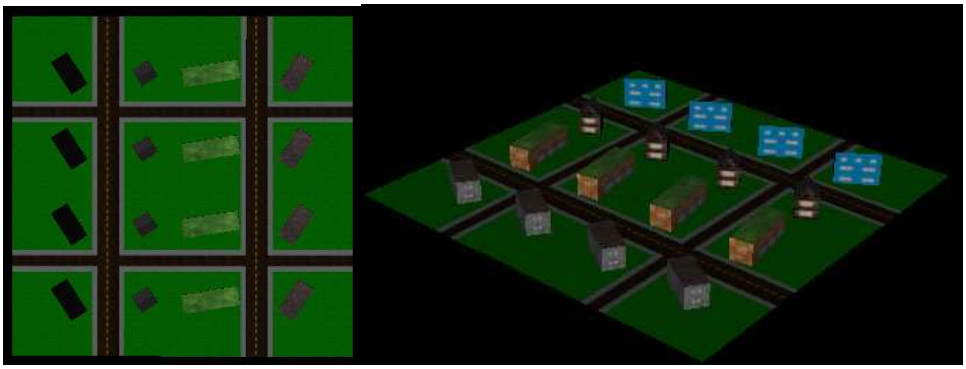


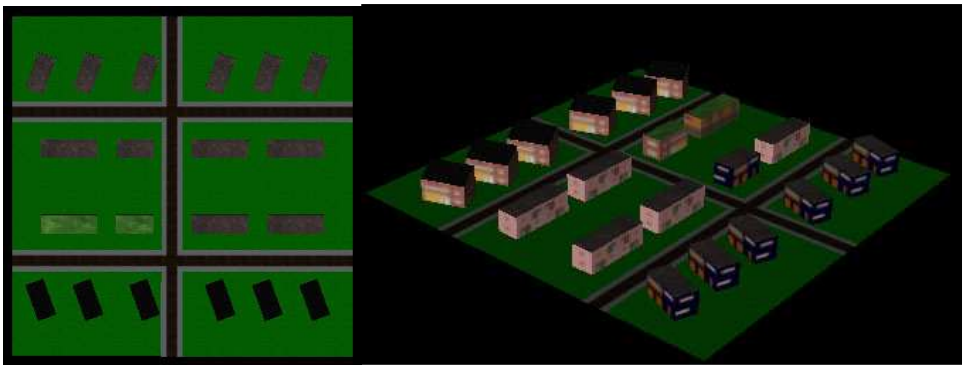


square00, square01, suburb01, suburb02

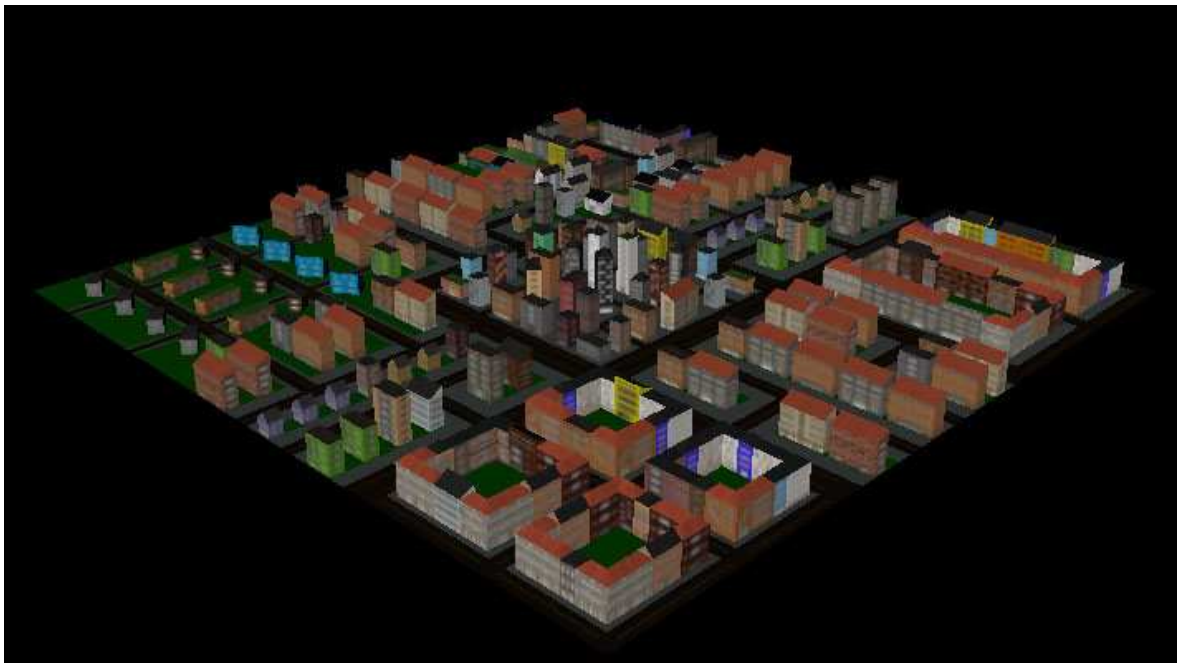


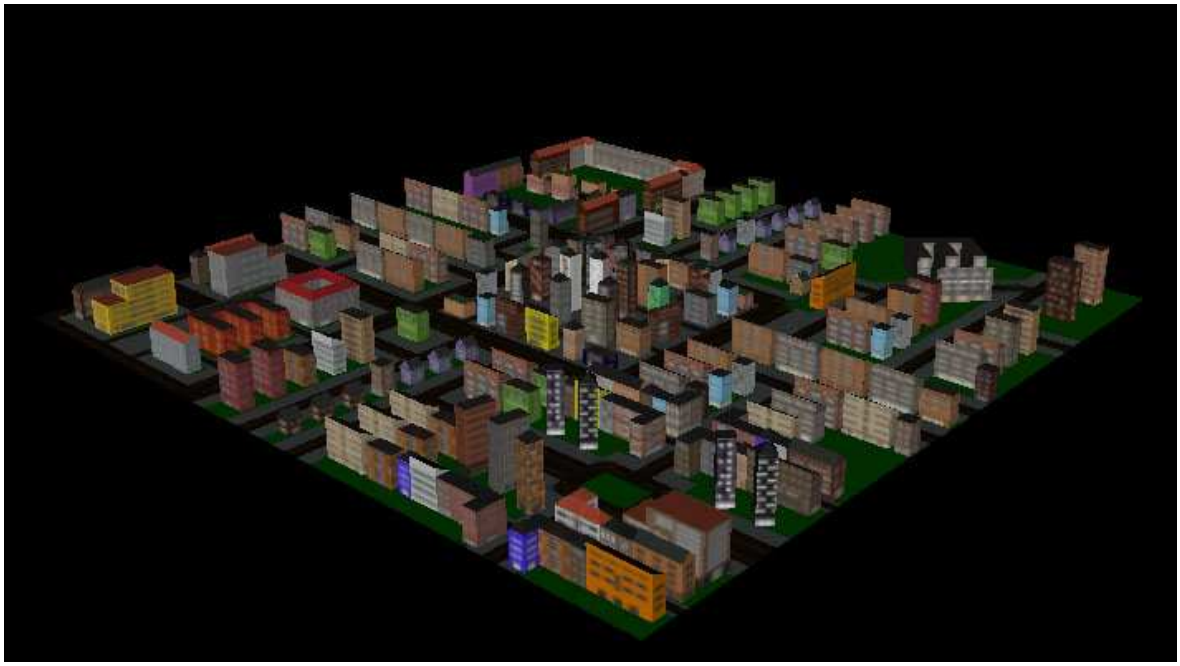


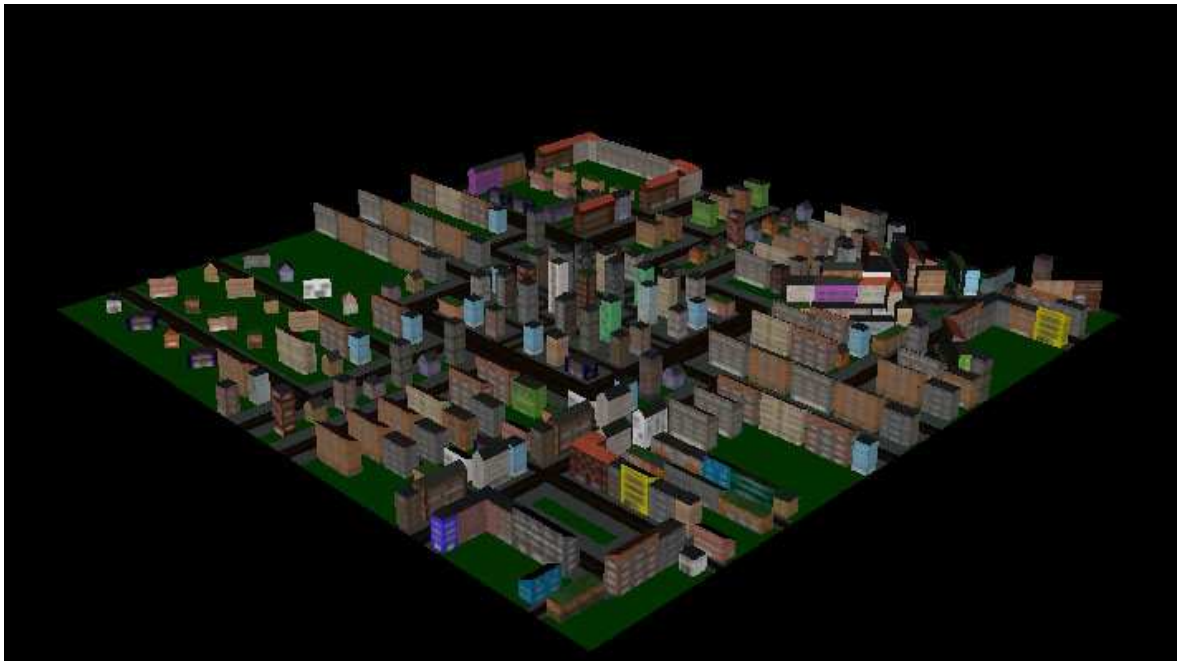




příloha č.4







příloha č.5