ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA STAVEBNÍ

OBOR GEODÉZIE A KARTOGRAFIE
KATEDRA GEOMATIKY

Procedurální modelování historických objektů a krajiny

semestrální práce

Pavel Tobiáš

V Praze dne 18. ledna 2016

Geografické informační systémy

Obsah

Úvod

Trojrozměrné virtuální modely významných budov jsou v dnešní době již poměrně obvyklé. Přitom se většinou jedná o manuálně vytvořené modely, které tvůrce bod po bodu rekonstruoval v CAD softwaru a které zachycují pouze zájmový objekt, tedy většinou samotnou historickou památku. Pro veřejnost by ale mohly být zajímavější a přínosnější 3D vizualizace obsahující kromě hradu, zámku, či jiné významné budovy také její okolí, tedy krajinu a zástavbu, která památku obklopuje.

Tvorba vizualizací rozsáhlých zastavěných oblastí klasickým přístupem je velice časově náročná. Tato semestrální práce proto popisuje procedurální modelování, které se od ručního modelování v CAD programu odlišuje vysokou mírou automatizace. Místo rekonstrukce předlohy bod po bodu je výsledek založen na jejím popisu v textovém, člověkem čitelném souboru. Ten obsahuje strukturu budovy nebo skupiny budov a slouží jako základ pro automatické generování modelu.

V následujících kapitolách bude nejprve stručně popsána tvarová gramatika CGA, která byla od začátku navržena jako prostředek pro procedurální popis architektury. Ve druhé kapitole bude potom představena implementace této tvarové gramatiky, tedy program City Engine. Kapitola třetí nakonec obsahuje shrnutí rešerše existujících prací, které se zabývají aplikací procedurálního modelování pro tvorbu počítačových rekonstrukcí historických památek a jejich okolí. Hledány přitom byly také alternativní přístupy k procedurálnímu modelování, které nevyužívají tvarovou gramatiku CGA a program City Engine.

1 Tvarová gramatika CGA

Procedurální modelování architektury je postaveno na produkčních systémech, do kterých patří například Semi-Thue procesy[1] nebo Chomského gramatiky[2]. Zatímco uvedené systémy pracují nad množinou symbolů (abeceda), z nichž vytvářejí řetězce podle zadaných pravidel, existují i další gramatiky, které místo symbolů operují s tvary. Z těch jsou aplikací pravidel vytvářeny geometrické obrazce. Příkladem mohou být L-systémy[3], které Przemyslaw Prusinkiewicz a Aristid Lindenmayer použili pro popis růstu jednoduchých organismů (obr. 1). Gramatika CGA, která je od začátku primárně určena pro procedurální modelování architektury, potom patří

[1] https://en.wikipedia.org/wiki/Semi-Thue_system
[2] Americký lingvista Avram Noam Chomsky. Viz: https://chomsky.info/
[3] Lindenmayerovy systémy: https://en.wikipedia.org/wiki/L-system

do tvarových gramatik poprvé popsaných v práci George Stinyho a Jamese Gipse v roce 1971 [1, 2, 3].

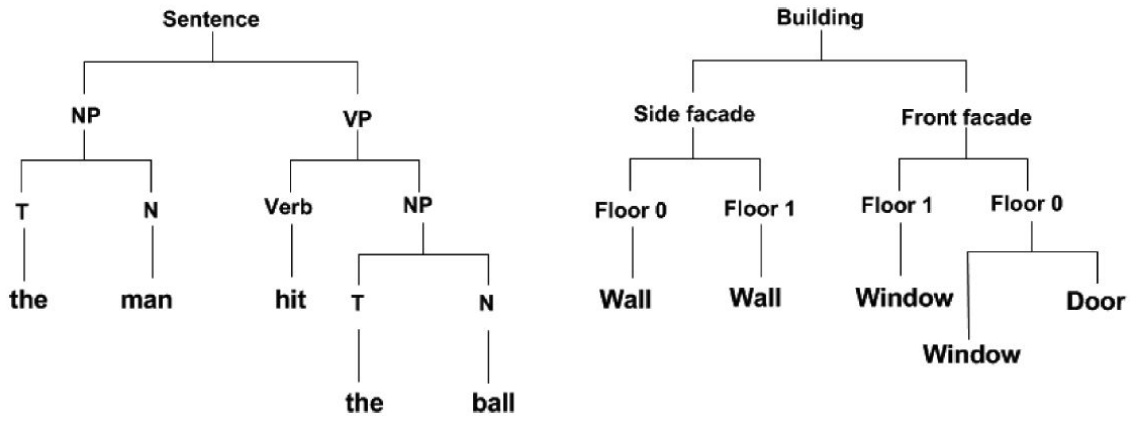
Obr. 1: Traviny vymodelované pomocí L-systémů (zdroj: Wikipedie)

1.1 Historie

Historii tvarové gramatiky CGA a tedy vlastně i programu City Engine lze vysledovat do roku 2001. V tomto roce Yoav I. H. Parish a Pascal Müller prezentovali v článku *Procedural modeling of cities* [4, 5], jak lze s využitím tvarové gramatiky rychle vytvářet *jednoduché* trojrozměrné modely města na základě 2D polygonů, kterým je přiřazena výška, jsou vysunuty do prostoru a jsou na ně aplikovány textury. V roce 2003 bylo Wonkou a kolektivem v článku *Instant architecture* [6] naopak předváděno modelování geometrických *detailů* na fasádě. Popsána zde byla tzv. dělící gramatika (split grammar), která umožňuje rozdělovat 3D objekty na části. Kombinace obou metod byla potom představena v článku Müllera, Wonky a kolektivu *Procedural modeling of buildings* [2]. Tento článek byl vydán v roce 2008 a je zde popsána již také implementace nové gramatiky, nazvané CGA (Computer Generated Architecture), v programu City Engine. Ten se objevil na trhu ve stejném roce [1, 2, 7].

1.2 CGA tvar

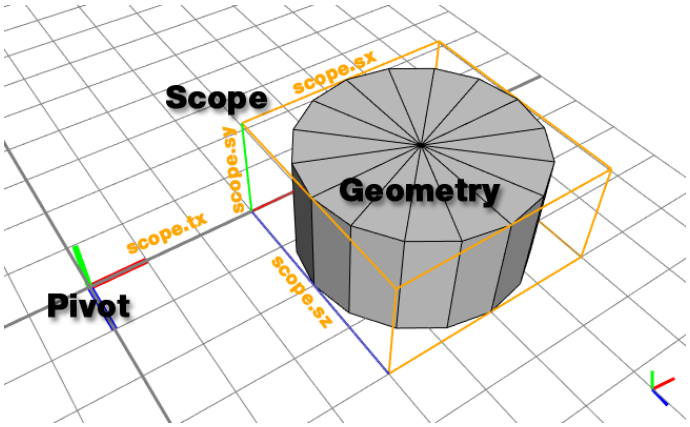
CGA je tvarovou gramatikou. Proto je jejím základním elementem *tvar* (shape). Každý tvar se skládá z označení (shape symbol, rule name), parametrů a atributů. Označení tvaru je jméno elementu, tímto jménem je na něj odkazováno v souboru pravidel gramatiky a to tak, že pravidlo s odpovídajícím jménem je použito pro generování nástupce tvaru (viz dále).



Obr. 2: Chomského koncept bezkontextové gramatiky a její adaptace pro budovu dle Stinyho a Müllera [8]

Každý tvar může mít přiřazeny parametry. Tyto parametry jsou implicitně definovány v pravidle, které daný tvar vytváří. Podporovány jsou parametry logické (bool), číselné (numeric, interně reprezentovány jako double-precision float) a textové řetězce (string).

Nakonec atributy popisují geometrii a prostorové umístění tvaru. Nejdůležitějšími atributy jsou geometrie (geometry), rámec (scope) a pivot (počátek souřadnicového systému tvaru). Uvedené pojmy nejlépe vysvětlí obrázek 3. Tvary v CGA mohou být terminální nebo neterminální, přičemž, jak je z názvu zřejmé, pro terminální tvar nejsou definována žádná pravidla, která by ho nahrazovala tvarem jiným [2, 3, 9].



Obr. 3: Tvar, jak je definován v CGA [9]

1.3 CGA pravidlo

Výsledný vzhled procedurálně generovaného modelu popisuje soubor pravidel. Ten obsahuje pravidla, která iterativně vylepšují vzhled modelu postupným přidáváním detailů. S pomocí tvarové gramatiky CGA je nejprve vymodelován hrubý objemový model (mass model). Poté je vytvořena struktura fasády a nakonec jsou přidány detaily (okna, dveře, ozdobné prvky). Vlastní přidávání detailu je realizováno nahrazováním jednodušších tvarů několika tvary detailnějšími. V rámci modelování je tak vlastně definována hierarchická struktura modelu a sémantika, čehož může být využito pro procedurální obměny. Může tedy být vygenerována celá řada různých modelů, které společně vytvoří např. model města [2].

Obecný tvar CGA pravidla vypadá následovně:

$$předchůdce --> následník$$

Na levé straně pravidla je označení tvaru, který má být nahrazen. Na straně pravé je potom odkazováno na tvar nebo množinu tvarů, které původní tvar nahrazují. Dále zde může být popsána tvarová operace pro tvorbu nových tvarů nebo podmínka či stochastické pravidlo. Je tedy možno zajistit větvení pravidla, kdy se rozhoduje o tom, který tvar bude vybrán pro nahrazení nebo která tvarová operace bude použita.

První pravidlo v souboru pravidel je označeno jako startovací (start rule). Na levé straně má toto pravidlo kořenový tvar (výchozí tvar, root shape). Dále jsou pravidla vyhodnocována postupně a tvarové operace (a tedy nahrazování) jsou prováděny, dokud existují tvary, které mají definována pravidla pro jejich nahrazení, tj. jsou neterminální. Proces generování je tedy ukončen, když zbývají pouze tvary terminální.

Tvary, které jsou nahrazeny, nejsou mazány, ale jsou označeny jako neaktivní. Tím je stále v průběhu celého procesu generování zachovávána hierarchická struktura modelu. Tuto strukturu lze vyjádřit pomocí stromu tvarů. Výsledný model je množinou všech listů stromu (tedy terminálních tvarů) [2, 3, 9].

1.4 Tvarové operace

Pomocí tvarových operací je možné modifikovat geometrické atributy (změna velikosti, umístění rámce) i vzhled tvaru (barva, textura), případně i vytvářet tvary nové [3, 9]. Nejdůležitější tvarové operace gramatiky CGA budou uvedeny dále v této kapitole.

Posunutí

Operace posunutí mění polohu rámce a tedy i samotného tvaru. V CGA syntaxi se zapisuje:

$$\mathrm{t}(tx,ty,tz),$$

kde *tx*, *ty* a *tz* jsou velikosti posunutí v jednotlivých osách (viz obr. 3 pro představu o orientaci os, y jde nahoru!). Velikosti mohou být buď absolutní nebo relativní (vztažené k rozměrům rámce).

Otočení

Operace otočení, zapisovaná:

$$\mathrm{r}(centerSelector,\ xAngle,\ yAngle,\ zAngle),$$

provede otočení tvaru. Středem rotace je dle parametru *centerSelector* buď počátek souř. systému tvaru (scopeOrigin, implicitní) nebo střed rámce tvaru (scopeCenter).

Změna velikosti

Poslední transformační operace změna velikosti:

$$\mathrm{s}(float\ xSize,\ float\ ySize,\ float\ zSize)$$

změní velikost rámce tvaru. Stejně jako u translace mohou být hodnoty parametrů zadány relativně (v syntaxi CGA s apostrofem).

Vložení 3D objektů

Tato operace umožňuje vložit instanci geometrického primitiva (krychle, válec) nebo 3D model vytvořený v jiném modelovacím softwaru v podporovaných formátech, tedy OBJ nebo COLLADA DAE. Druhá uvedená možnost může být velmi vhodná pro vkládání detailů, které by již nebylo efektivní modelovat procedurálně (ozdobné prvky, detailní modely oken, dveří...). Operace má zápis:

$$\mathrm{e}(geometryPath),$$

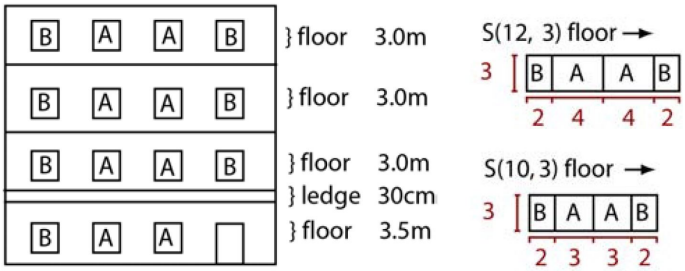
kde *geometryPath* je identifikátor objektu, tedy relativní cesta, proměnná nebo název geometrického primitiva.

Pravidlo pro rozdělení

Pravidlo pro rozdělení (split rule) ve tvaru:

$$\text{split}(splitAxis)\ \{size(1){:}operations(1)\ |\ size(n){:}operations(n)\}$$

rozdělí tvar na části podél jedné zvolené osy. Parametry *size(n)* vyjadřují vzdálenosti, ve kterých se rozdělení provede (tj. velikosti výsledných elementů). Tyto hodnoty mohou být přibližné nebo může být specifikováno, že se má určitý element s přibližnou velikostí opakovat n-krát v rámci původního tvaru. Typicky je této operace využíváno pro rozdělení hrubého tvaru budovy na patra a následně na dlaždice (tiles) obsahující okna nebo dveře.



Obr. 4: Příklad rozdělení fasády na části pomocí pravidla pro rozdělení [2]

Rozložení na díly

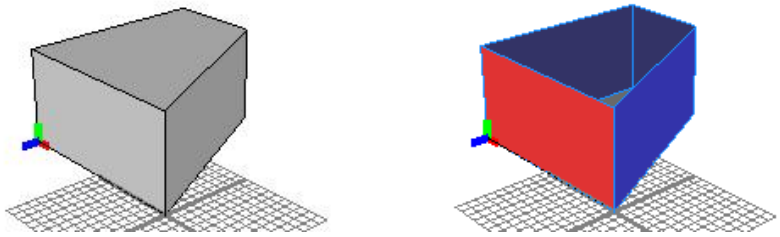
Operace rozložení na díly (component split) slouží pro rozdělení původního trojrozměrného tvaru na tvary nižších dimenzí (tj. na jeho dílčí části – plochy, hrany nebo vrcholy). Syntaxe pro zápis operace je následující:

$$\text{comp}(compSelector)\ \{selector\ :\ operations|selector\ :\ operations\ldots\}.$$

Parametr *compSelector* určuje, zda rozdělení proběhne na plochy (f), vrcholy (v) nebo hrany (e). Parametry *selector* potom specifikují umístění části (tedy nového nahrazujícího tvaru) v rámci tvaru původního. Umístění může být vyjádřeno např. vzhledem k světovým stranám v rámci souřadnicového systému scény, s ohledem na úhly, které tvary svírají se základními rovinami scény (vhodné pro střechy) nebo může být použit i index v rámci původního tvaru. Příklad rozdělení hrubého objemového modelu budovy na hlavní a boční fasády je na obr. 5. Pro rozdělení bylo použito pravidlo:

```
Building-->
   comp(f) {
      front : color("\#ff0000") Main |
      side  : color("\#0000ff") Side
   }
```

Aplikací pravidla bude tvar `Building` nahrazen tvary hlavní a vedlejších fasád a na fasády budou pro rozlišení aplikovány příslušné barvy.



Obr. 5: Příklad rozdělení hrubého objemového modelu na díly [9]

Vysunutí

Na rozdíl od tvarové operace rozložení na díly, která rozděluje 3D tělesa na jednodušší části a tedy vlastně ubírá jeden či dva rozměry, vysunutí tvarům rozměr přidává. Operace pracuje v základní podobě s parametrem *height*, tedy s výškou vysunutí:

$$\text{extrude}(height).$$

Operace je často využívána pro vytvoření hrubého objemového modelu budovy z 2D půdorysu, což je nezbytné pro práci s dvourozměrnými vrstvami importovanými z GIS [2, 3, 9].

1.5 Větvení pravidel

Dle [2] může být procedurální pravidlo obecněji zapsáno takto:

$$\text{predecessor: cond.} -\!-\!> \text{successor: prob.}$$

Každé pravidlo tedy může obsahovat větvení. O tom, která větev pravidla bude vybrána (a tedy který tvar bude použit pro nahrazení), rozhoduje buď podmínka

(cond.) nebo stochastické pravidlo (prob.). Podmínka, která je založena většinou na porovnávání atributů nebo parametrů tvaru, je zapsaná s využitím klíčových slov *case* a *else*, přičemž druhé uvozuje poslední větev pravidla, například:

```
Footprint(type) -->
      case type == "residential" : extrude(10) House
      case type == "industrial" : extrude(15) Factory
      else : NIL
```

Stochastické pravidlo je potom založeno na specifikování pravděpodobnosti, s jakou bude provedeno nahrazení jednotlivými možnými tvary ve větvích pravidla. Výběr konkrétní větve tedy neprobíhá na základě splnění podmínky, ale v podstatě na základě náhodné vygenerované hodnoty [2, 9]:

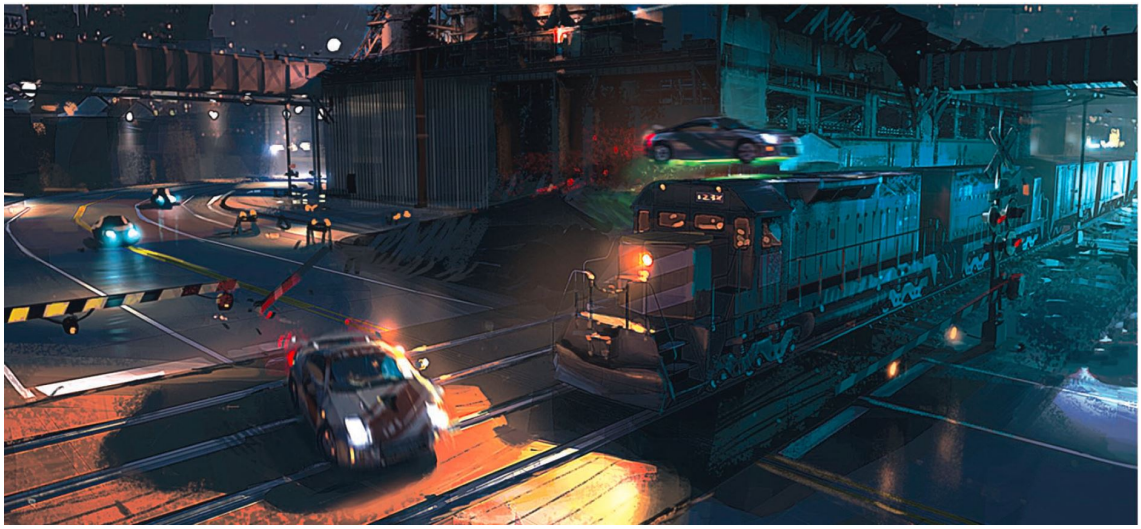
```
Lot -->
    30%  : Residential
    20%  : Retail
    else : Industrial
```

2 Implementace CGA – program City Engine

Popisovaná tvarová gramatika CGA je s využitím jazyka C++ implementována v programu City Engine, který byl, jak již bylo zmíněno výše, komerčně představen v roce 2008. Hlavním záměrem tvůrců bylo vytvořit program, s pomocí kterého by bylo možno rychle generovat rozsáhlé modely měst a osídlené krajiny, využitelné v počítačových hrách a ve filmovém průmyslu [10]. Taková města měla být spíše fiktivního charakteru, čemuž odpovídají některé funkce programu, které umožňují velmi rychle automaticky navrhovat například i vzhled uliční sítě [2].

2.1 Modelování v programu City Engine

Nejefektivnější metodou, jak navrhnout model fiktivního města, je vytvořit prázdnou scénu a generovat maximální množství prvků automaticky. Pomocí funkce *Growing Street Networks* je možné nechat program navrhnout uliční síť na základě zvolených parametrů v předdefinovaném stylu. Ten může odpovídat původně středověkému městu, městu, které bývalo původně pevností nebo na zelené louce vzniklému sídlu, jakým je například New York. Po vygenerování ulic jsou v prostoru mezi nimi



Obr. 6: Prostředí pro hru Need for Speed bylo vytvořeno s využitím CGA [10]

vytvořeny stavební parcely (lot) a ty jsou rozděleny na polygonové tvary. Těmto polygonům je nakonec možno přiřadit CGA soubor pravidel, který pomocí tvarové gramatiky vygeneruje vlastní budovy. Podkladem pro tvorbu města mohou být také digitální modely terénu nebo rastry popisující překážky (např. vodní plochy) [7, 9].

I když je z výše uvedeného patrné, že je City Engine optimalizován pro rychlou tvorbu fiktivních měst, již od začátku se počítalo s tím, že by mohl být využit i pro potřeby 3D GIS [2]. Není proto překvapivé, že si tohoto softwaru všimla společnost ESRI a v roce 2011 zakoupila firmu Procedural, která do té doby City Engine vyvíjela[4]. Od té doby probíhá snaha o začlenění programu do portfolia ESRI (například v rámci ArcGIS Pro).

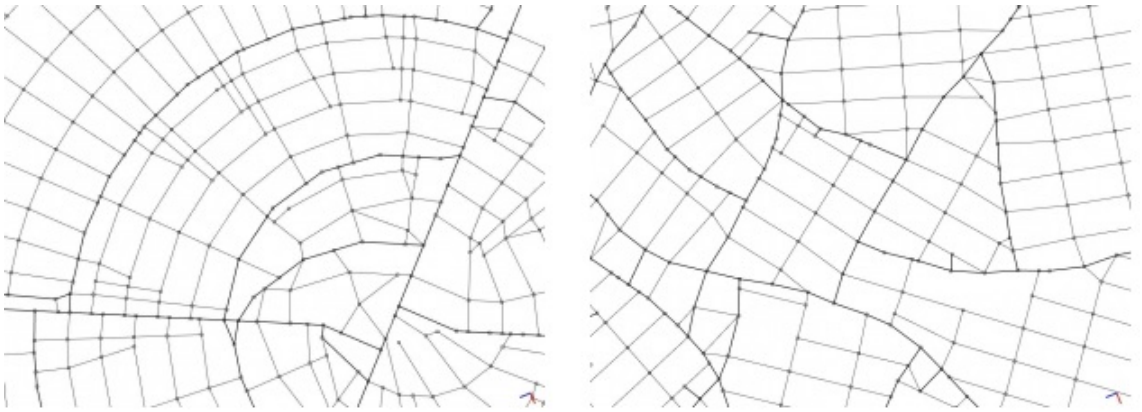
Už ve vlastnictví firmy Procedural bylo možné do City Enginu importovat půdorysy (footprinty) budov nebo hrubé objemové modely původem z GIS a ty dále detailně zpracovávat pomocí tvarových operací. V současné době lze importovat jak vrstvy z ESRI geodatabáze, tak ESRI shapefile a výsledek zpět do těchto datových struktur exportovat. Teoreticky je tak program dobře připraven i pro vytváření modelů existujících měst, které mohou být dále využity například pro územní plánování a urbanistické studie [10].

[4] http://www.esri.com/news/releases/11-3qtr/esri-acquires-3d-software-company-procedural.html

2.2 Slabé stránky a specifika

Každý software má své slabé stránky a také u City Enginu je možné nalézt nedostatky plynoucí z původního hlavního určení programu. I když program nabízí klasické modelování a manuální editaci, není práce s dotyčnými nástroji příliš pohodlná. Zaměření na procedurální modelování fiktivních krajin se projevuje také například chybějícími nástroji pro měření a tedy i pro kontrolu vytvořených modelů. Jako další nedostatek je uváděno nedokonalé rozvrstvení scény oproti GIS, kdy se nad sebou ležící vrstvy prolínají, dochází k mísení textur a výsledek je nepoužitelný [7]. To ale plyne především z jiného způsobu zobrazování ve 2D GIS nástrojích a 3D scénách City Enginu.

Uvedené nedostatky sice mohou práci znepříjemnit, ale těžko je lze označit za zásadní. Podstatný problém mohou představovat potíže s integrací City Enginu do portfolia ESRI a s tím související problémy s kompatibilitou programu například se softwarem ArcScene, které jsou uvedeny v [7]. Uvedená práce sice popisuje nedostatky ve starší verzi programu, každopádně i ve verzi 2014.1 byly pozorovány těžkosti při zápisu výsledku do ESRI geodatabáze (typ multipatch), jako například zablokování scény po exportu.



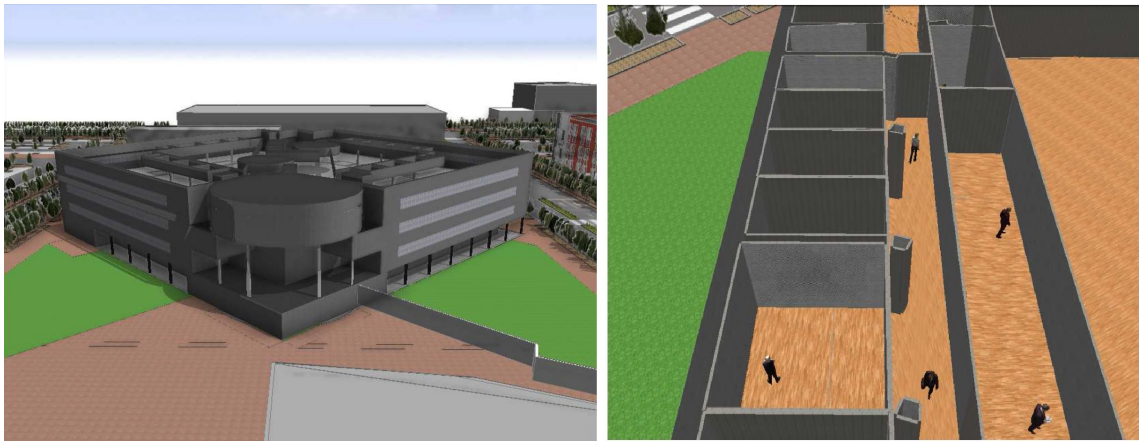
Obr. 7: Příklady uliční sítě vygenerované v programu City Engine [9]

Specifikem procedurálního modelování v programu City Engine, nad kterým je nutno se zamyslet, je shoda výsledného modelu s předlohou. Program je navržen pro vysoce efektivní modelování v téměř libovolném detailu. V případě kulis pro počítačové hry nebo filmy tak lze vytvořit vysoce detailní a věrohodné prostředí. Pokud ale modelujeme rozsáhlé existující město, budeme se muset téměř vždy smířit s tím, že výsledek bude sice vypadat hezky, ale nebude se přesně shodovat s realitou. Pokud bychom chtěli modelovat každou budovu přesně dle předlohy, procedurální modelování by nepředstavovalo výhodu oproti klasickému přístupu.

Na druhou stranu ale lze s dostupnými nástroji programu vytvořit model, který bude přibližně respektovat vzhled jednotlivých částí města. Například u moderních sídlišť nebo oblastí s kancelářskými budovami, kde se jednotlivé objekty liší jen málo, tak lze vystihnout celkový dojem poměrně snadno. Výsledná scéna bude potom obsahovat modely budov typických pro danou část a to může být s ohledem na předpokládané využití (např. územní plánování, skyline analýzy) dostačující. Každopádně stále je nutno počítat s tím, že jednotlivé budovy nebudou při detailním pohledu odpovídat svým protějškům ve skutečném městě. Tento problém se týká i modelování historického vzhledu krajiny a měst, kdy se připojuje také nedostatek podkladů pro modelování a bude popisován i dále v textu.

2.3 Procedurální modelování vnitřních prostor

Poslední poznámka, která je již poněkud na okraji popisovaného tématu, tedy modelování historických objektů a krajiny, se týká možnosti procedurálního generování vnitřních prostor budov. Většina prací o procedurálním modelování se zabývá tvorbou modelů rozsáhlých zastavěných oblastí, ať už historických nebo současných, s důrazem na exteriér. Snahu o vytvoření modelu budovy včetně interiéru je možno nalézt v [7] a v menší míře také v [11]. V [7] je přitom popisována tvorba modelu rektorátu univerzity v Castelló de la Plana ve Španělsku. Základem je 2D CAD dokumentace budovy, která je importovaná do GIS. Na výsledné půdorysy jednotlivých pater jsou aplikována procedurální pravidla obdobně, jako je tomu v klasickém případě vysunování 2D půdorysů budov při tvorbě exteriérů.



Obr. 8: Procedurálně vytvořený model univerzitního kampusu včetně interiérů [7]

Na rozdíl od klasického modelovacího přístupu jsou v procedurálním modelování důležité informace o struktuře a sémantika. Prostorové a sémantické vztahy mezi objekty (= tvary) jsou významnější než absolutní geometrické souřadnice a atributy

vzhledu [1, 2]. Tento přístup je tedy velmi blízký problematice informačního modelování budov (BIM). Vzhledem k tomu, že je v současné době dokumentace většiny stavebních objektů vedena v podobě 2D CAD výkresů, je možno předpokládat, že by se procedurální modelování mohlo v budoucnosti využívat jako rychlý způsob převodu 2D dokumentace do 3D a to i v případě některých památkových objektů. Otázkou v případě City Enginu je další postup integrace s ostatními produkty ESRI a s tím související možnosti exportu výsledného modelu a práce s ním v dalších softwarových komponentách, jako je ArcScene. Kromě výsledku výše uvedené práce (obr. 8) je příkladem použití City Enginu pro vytvoření 3D scény včetně interiéru dobře známá ukázka modelu kampusu ESRI v Redlands[5].

3 Snahy o využití procedurálního modelování pro historické objekty

Zřejmě největší přínos má procedurální modelování při tvorbě modelů fiktivních měst a krajiny, protože je možno využít v maximální míře výhody automatického generování objektů a tvůrci nejsou omezeni požadavky na shodu modelu s existující předlohou. Přesto je tento přístup vhodný i pro modely zachycující stav sídel v minulosti nebo přímo pro 3D rekonstrukce hypotetického vzhledu archeologických nalezišť. Procedurální modelování je možno využít v případě, že je třeba vytvořit přibližnou rekonstrukci rozsáhlého území na základě dostupných pramenů, přičemž významné budovy, o kterých je známo více, mohou být modelovány klasicky. Na druhou stranu mohou být generovány i detailní modely jednotlivých staveb, pokud jsou z podkladů zřejmá architektonická pravidla pro dané období a oblast.

Protože je program City Engine přímo určen pro procedurální modelování architektury, není s podivem, že je využíván i v řadě prací zabývajících se 3D rekonstrukcemi historických objektů. O využití v archeologii pojednává článek Haeglera a kolektivu *Procedural Modeling for Digital Cultural Heritage* [1], který předkládá několik příkladu využití procedurálního modelování na základě výsledků zkoumání archeologických nalezišť. Podobné příklady je možno nalézt také v článku Watsona a kolektivu *Procedural Urban Modeling in Practice* [10]. Stejně tak Calogero, Kaminsky a Arnold [8] používají City Engine pro rekonstrukci vzhledu východního křídla Louvru na základě odmítnutých návrhů a pro porovnání s návrhem realizovaným.

Na druhou stranu lze nalézt i přístupy, které pro generování procedurálních modelů využívají jiné prostředky. Jedná se například o práci Rodriguese a kolektivu [11],

[5] http://arcg.is/1mrm06k

Obr. 9: Rekonstrukce různých návrhů vých. křídla Louvru podle [8]

kteří používají L-systémů pro modelování římských domů na nalezišti Conímbriga v Portugalsku. V článku Laycocka, Drinkwatera a Daye [12] je potom rozebráno modelování změn památkových objektů a měst v čase s využitím vlastního blíže nespecifikovaného postupu procedurálního modelování. V následujícím textu budou s využitím uvedených pramenů a dalších podkladů rozebrány možnosti a přínosy procedurálního modelování v archeologii a obecně při modelování historických objektů.

3.1 Procedurální modelování v archeologii

Ve výše zmiňovaném článku Haegler a kolektiv [1] uvádí, že v současné době roste poptávka po realistických 3D modelech historického stavu archeologických nalezišť. Takové modely mohou samozřejmě velmi dobře sloužit pro prezentaci výsledků archeologického badání široké veřejnosti. Na druhou stranu ale nemusí zůstat

jen u toho a dobře vytvořené 3D scény mohou být využity i jako „dynamické výzkumné modely“, které mohou sloužit jako podklad pro další diskuzi o původním vzhledu místa.

Základním metodickým dokumentem pro tvorbu 3D modelů v archeologii je *London Charter*[6] [13]. Text je dostupný online v několika podobách a shrnuje nejdůležitější zásady pro 3D modelování historických památek. Dále je uveden přibližný překlad šesti základních principů:

1. Princip: Implementace

Principy London Charteru platí všude, kde je aplikována počítačová vizualizace za účelem výzkumu nebo šíření kulturního dědictví.

2. Princip: Cíle a metody

Metoda počítačové vizualizace by měla být použita pouze tehdy, jedná-li se o nejvhodnější dostupnou metodu pro daný účel.

3. Princip: Výzkumné prameny

Aby byla zajištěna intelektuální integrita metod počítačové vizualizace a jejich výsledků, měly by být strukturovaným a dokumentovaným způsobem hledány a vyhodnocovány relevantní výzkumné prameny.

4. Princip: Dokumentace

Mělo by být dokumentováno a šířeno dostatečné množství informací tak, aby bylo umožněno pochopení a posouzení metod počítačové vizualizace a výsledků ve vztahu ke kontextu a k účelům, pro které byly použity.

5. Princip: Udržitelnost

Měla by být naplánována a implementována strategie s cílem zajistit dlouhodobou udržitelnost výsledků a dokumentace počítačové vizualizace zabývající se kulturním dědictvím, aby se zabránilo ztrátě této rostoucí části lidského intelektuálního, sociálního, ekonomického a kulturního dědictví.

6. Princip: Přístup

Tvorba a šíření počítačových vizualizací by mělo být plánováno takovým způsobem, aby bylo zajištěno, že budou co nejpřínosnější pro studium, pochopení, interpretaci, zachování a správu kulturního dědictví.

[6] V překladu Londýnská charta. S ohledem na stejnojmenný historický dokument raději v textu dále nepřekládám.

Pokud budou dodrženy principy London Charteru, měly by být odstraněny kritizované vlastnosti 3D modelů archeologických nalezišť, tedy zejména jejich netransparentnost (z jakých dat se vychází?) a nedostupné vizualizace alternativního vzhledu památek (předložena je často pouze jedna hypotéza) [14]. Jak může být pro zajištění souladu vizualizace s výše uvedenými principy využito procedurální modelování, bude uvedeno v následující kapitole.

3.2 Rozpor mezi realismem a skutečnými znalostmi

Velkým problémem, který je třeba řešit při tvorbě 3D rekonstrukcí historických památek, je rozpor mezi vizuální kvalitou prezentovaného modelu a skutečnými znalostmi o modelované předloze. Vysoce fotorealistické modely s velkou úrovni detailu (LOD) určitě přilákají více zájemců z řad veřejnosti než modely jednoduché a schématické. Na druhou stranu může velký detail a kvalita vizualizace vést k dojmu, že vzhled všech částí modelu je podložen vědeckými fakty. Existuje proto několik přístupů k modelování nejistoty, tedy k vyjádření toho, jak podrobné jsou znalosti o jednotlivých objektech modelu.

Obvyklým přístupem je modelování nejistoty pomocí průhlednosti částí modelu, pomocí textur nebo pomocí různých LOD [1]. Části modelu, u nichž je skutečná podoba méně podložena známými fakty, budou samozřejmě zobrazeny v nižším LOD, se světlejšími nebo méně podrobnými texturami nebo průhledně. Příklad uvedeného je možno najít v [8], kde je využito následujícího rozdělení jednotlivých objektů do skupin dle míry vědeckých poznatků o nich:

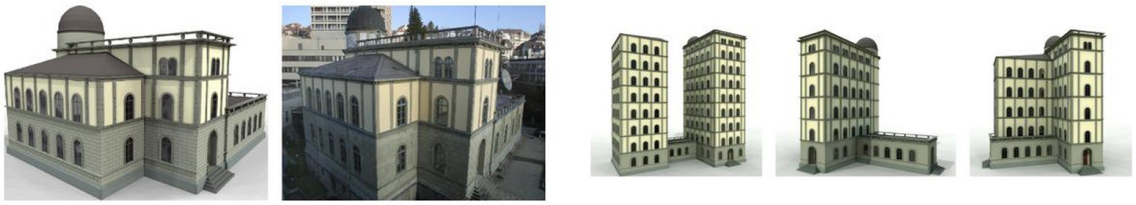
Empirické znalosti: tvar, uspořádání, rozměry a materiál známy na 80% – 100%. Fotorealistická vizualizace, vysoký LOD.

Interpretace: tvar, uspořádání, rozměry a materiál známy na 50% – 80%. Semifotorealistická vizualizace, střední LOD.

Domněnka: tvar, uspořádání, rozměry a materiál známy na méně než 50%. Nefotorealistická vizualizace, nízký LOD.

S využitím výše uvedeného je možné modelovat nejistotu poměrně názorně. Takovýto přístup ale může výrazně snížit zážitek z prohlížení, obzvláště pak v případech, kdy je rekonstrukce historického vzhledu archeologické lokality či památkové budovy spíše výsledkem odborného odhadu. V [1] je proto propagován odlišný přístup. Výsledkem úsilí modelářů nemá být jedna vizualizace, ale více modelů, které zobrazují různé hypotézy. Protože časová náročnost klasického přístupu tvorby 3D modelů by se v takovém případě samozřejmě znásobila, navrhuje článek využít procedurální

modelování založené na tvarové gramatice. Takové modelování je velmi efektivní a umožňuje poměrně rychle vygenerovat více modelů, které mohou být vysoce fotorealistické a tedy příjemné na pohled. Vjem nejistoty je potom zprostředkován pomocí většího množství předložených variant. Jednotlivé modely mohou být stejné tam, kde víme poměrně přesně, jak objekt vypadal. Naopak v místech s větší nejistotou, tedy tam, kde existují různé hypotézy o vzhledu, se budou modely lišit.

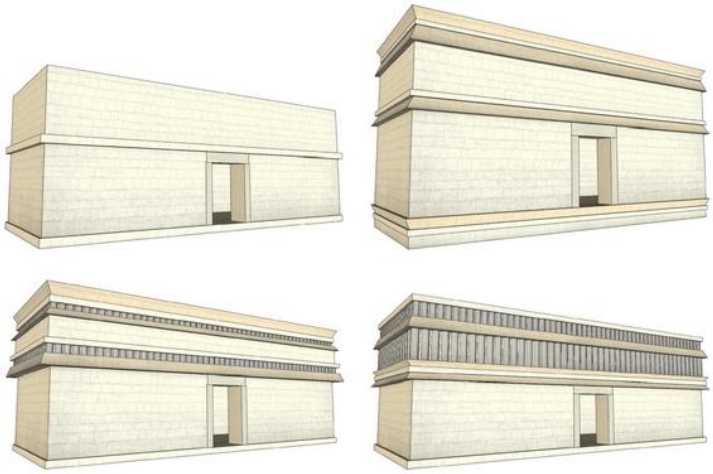


Obr. 10: 3D model Semperovy observatoře v Curychu [1]

Další výhodu procedurálního modelování si lze uvědomit, pokud ho porovnáme s modelováním tradičním. Při využití klasického přístupu modelář obdrží náčrtky s předpokládaným vzhledem památek, které vytvořili příslušní odborníci, tedy historikové a architekti. Na základě těchto skic potom vytváří virtuální podobu objektů interaktivně pomocí postupného zadávání vrcholů, linií, ploch, materiálů a barev tak, aby výsledek *vizuálně* co možná nejlépe odpovídal předloze. Krok po kroku je tak rekonstruován vzhled budovy a přitom nemusí struktura modelu odpovídat architektonické struktuře modelovaného objektu.

Naproti tomu je při procedurálním modelování struktura předlohy velmi důležitá, protože na základě její znalosti je vytvářen procedurální popis modelu v souboru pravidel. Tímto popisem je tak možno v podstatě slovně vyjádřit myšlení architekta předlohy. Přitom může být s výhodou využita literatura popisující architektonická pravidla v daném historickém období. V komentářích v souboru pravidel lze potom odkazovat na prameny, ze kterých vychází volba určité tvarové operace nebo parametru (například výška okna). Poznámky, které odkazují na hypotézy využité pro tvorbu modelu, tak mohou být uvedeny vlastně přímo ve zdrojovém kódu. Stejně tak lze takových vysvětlivek využít i ve výsledné vizualizaci a uživatel tak má dobrou představu o podkladech, ze kterých bylo čerpáno, a tedy i o věrohodnosti modelu. Protože je struktura modelu přehledně uložena v souboru pravidel, je možné model postupně zpřesňovat, jak roste počet věrohodných informací o rekonstruované památce. V případě použití klasického modelovacího přístupu jsou naproti tomu jakékoliv změny a zpřesňování hotového modelu velmi složité.

Pomocí většího množství modelů lze poměrně přesně vizualizovat i pravděpodobnosti jednotlivých hypotéz. Pokud je 90% pravděpodobnost hypotézy o vzhledu



Obr. 11: Čtyři hypotézy o vzhledu budovy v Kiuicu v Mexiku [1]

nějakého prvku, může ho v této podobě obsahovat devět z deseti modelů. Zbylý desátý model potom představuje méně pravděpodobnou hypotézu. Z [1] není bohužel dost dobře patrné, jak je přesně tento návrh více modelů myšlen. Pokud máme více hypotéz o vzhledu domů určitého typu ve městě, bude nutné, abychom vytvořili 10 modelů celého města? Nestačilo by, abychom různé hypotézy zohlednili při tvorbě jednotlivých domů a tedy modelovali město jednou, přičemž by nejvíce domů mělo vzhled odpovídající nejpravděpodobnější hypotéze? To bohužel není možné z předložených příkladů jednoznačně zjistit. Pravděpodobně bude záležet vždy na konkrétním případu. Každopádně bude nutné uvážit také to, jaký počet variant má smysl modelovat. Nebude deset odlišných modelů, byť pěkných a fotorealistických, pro uživatele již příliš?

Obr. 12: Procedurální rekonstrukce starověkých Pompejí [1]

Ačkoliv nelze v [1] nalézt uspokojivé odpovědi na výše uvedené otázky, je z tohoto článku alespoň možné převzít obrázky příkladů procedurálního modelování s využitím tvarové gramatiky v programu City Engine. Uvedené příklady dobře ilustrují jednotlivé výhody procedurálního přístupu pro modelování historické architektury.

Prvním příkladem je 3D model neoklasicistní observatoře v Curychu, navržené švýcarským architektem Gottfriedem Semperem (obr. 10). Model vznikl jako *přesná* virtuální rekonstrukce skutečného objektu (na obrázku druhý zleva) a ukazuje tak, že procedurální modelování je u vhodných objektů možno využít i pro tento účel. Model vytvořený dle skutečnosti je na obrázku první zleva, zbývající tři obrázky vpravo potom dokumentují, jak by se procedurální pravidla dokázala adaptovat na změnu počtu pater, případně na změnu půdorysu budovy. Uvedený princip by mohl být velmi výhodný v případě, že bychom potřebovali modelovat několik mírně odlišných domů podobného stylu.

Na příkladě budovy postavené v klasickém mayském stavebním stylu puuc (obr. 11) je ukázáno použití kontrolních parametrů, jejichž změnou je možno zásadně obměnit vzhled výsledného modelu. Hodnoty parametrů všech čtyř vyobrazených variant jsou přitom založeny na archeologických datech a nejsou tedy generovány náhodně.

Zatímco v předchozích dvou příkladech nebyly modely příliš rozsáhlé a zachycovaly vlastně jen jeden stavební objekt, další ukázka – rekonstrukce starověkých Pompejí (obr. 12) již představuje model celého města. Přitom díky procedurálnímu přístupu nemusí nijak výrazně klesat úroveň detailu jednotlivých jeho budov. Model byl vytvořen na základě půdorysů a archeologických skic budov různých typů. Celé město včetně uliční sítě a umístění rostlin popisuje asi 190 pravidel. Výsledný vzhled budov přitom závisí nejenom na jejich typu, ale také na poloze. Na zřetel jsou brány význam či hustota osídlení jednotlivých městských částí. Protože v některých částech Pompejí ještě v době modelování nebyly dokončeny vykopávky a o řadě budov proto nebyly k dispozici detailnější informace, jsou chybějící parametry (například výšky) generovány náhodně.

O detailní rekonstrukci starověkého města se snažil i projekt Rome Reborn[7]. Pro zachycení vzhledu starověkého Říma (obr. 13) byl použit kombinovaný přístup, kdy významné objekty (např. Koloseum) byly modelovány klasicky manuálně v běžném CAD softwaru a pro okolní zástavbu bylo použito procedurální modelování. Procedurálně bylo vytvořeno asi 7000 obytných domů (insulae) a také 60% vizualizovaných chrámů. Zajímavé je, že základem pro generování nebyly půdorysy budov, ale hrubé objemové modely vzniklé laserovým skenováním plastiky Říma od Italo Gismondiho[8]. Procedurální pravidla byla tedy použita až pro detailní popis fasád, přičemž musela být dostatečně adaptabilní, aby se byla schopna přizpůsobit různým

[7] http://romereborn.frischerconsulting.com/

[8] https://it.wikipedia.org/wiki/Italo_Gismondi

tvarů a rozměrům objemových modelů. O tom, jak detailní modely je možné vytvářet procedurálně, svědčí fakt, že musel být u modelů obytných domů v blízkosti významných objektů omezen detail, aby se přiblížil LOD manuálně vytvářených modelů těchto pamětihodností.

Obr. 13: Vizualizace projektu Rome Reborn (zdroj: web projektu)

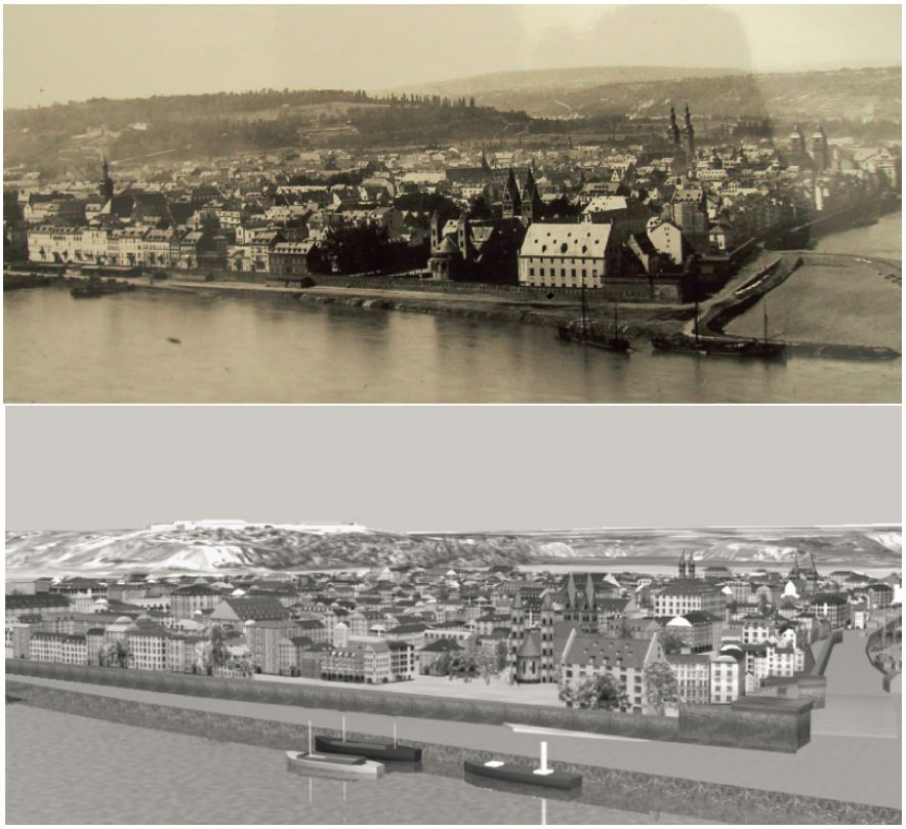
3.3 Modelování změn v čase

Metoda více modelů popisovaná v [1] představuje poměrně zajímavé řešení prezentace hypotetického vzhledu historických objektů včetně vyjádření nejistoty ve vědeckých poznatcích. Stejně tak ale může být zajímavé využít většího množství modelů pro zachycení vývoje zájmových objektů v čase. Článek dokonce navrhuje skloubit oba principy a vytvářet více modelu na základě více hypotéz pro každou zkoumanou epochu. Jestli by byla taková „sada sad“ modelů ještě zajímavá a čitelná pro běžného uživatele je otázkou, každopádně myšlenka zachycení změn památky v čase je samozřejmě velmi zajímavá. Vzhledem k nutnosti vytvářet více modelů může být i v takovém případě procedurální modelování výhodné.

Myšlenku zachycení vývoje památkových objektů v čase je možné nalézt také v článku Laycocka, Drinkwatera a Daye [12]. Zachycení historických změn stavby vnímají jako zásadní pro její lepší pochopení, přičemž doporučují modelovat nejen památku samu, ale také její okolí. Výsledkem je potom vlastně 4D vizualizace, kdy je čtvrtým rozměrem čas. Jaké softwarové nástroje byly použity není v článku uvedeno, každopádně je navržený postup poloautomatický a využívá opět procedurální modelování pro generování okolních budov v kombinaci s klasickým přístupem pro tvorbu vlastní památky.

Článek byl vydán již v roce 2008 a od té doby samozřejmě nástroje pro 3D modelování a vizualizace značně pokročily. Prezentovaným výsledkem jsou dvě interaktivní videa, kdy je uživateli k dispozici pouze několik vrcholů/pozic kamery a hran, které umožňují mezi vrcholy přecházet. Pohledy ze všech možných pozic

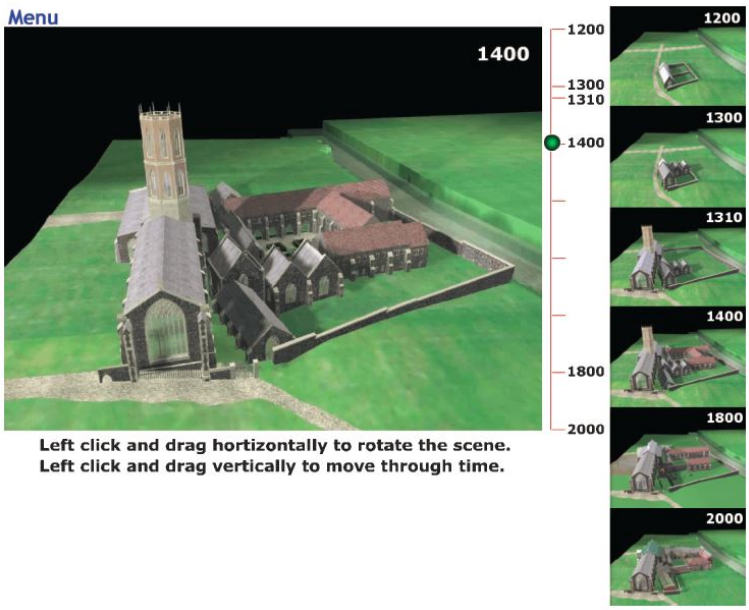
kamery jsou přitom renderovány předem. Vzhledem k pevným, dopředu daným pozicím kamery je možné rozvrhnout detailnost jednotlivých objektů scény tak, aby byly detailní budovy v popředí a schématické modely naopak v pozadí. První interaktivní video zachycuje město Koblenz v 19. století (obr. 14). Podkladem byla pouze jedna stará mapa (viz dále) a nejedná se tedy o 4D scénu. Druhá rekonstrukce zachycuje i změny v čase a zobrazuje tak vývoj St. Andrew's Hall v Norwichi (obr. 15), přičemž ale zase modeluje pouze nejbližší okolí památky.



Obr. 14: Historická fotografie a 3D rekonstrukce města Koblenz v 19. stol. [12]

I když uvedené výstupy, vytvářené s ohledem na tehdejší schopnosti běžných počítačů, již plně neodpovídají současným metodám a zvyklostem pro 3D vizualizace, je postup tvorby modelu stále poměrně aktuální. Základem byl digitální model terénu, generovaný z dat leteckého laserového skenování. Na tento model terénu byly potom umísťovány 3D modely budov, přičemž o jejich umístění bylo rozhodováno na základě půdorysů poloautomaticky získaných ze staré mapy. Mapa byla nejprve filtrována pomocí několika Gaussových filtrů a tak byl odstraněn šum. Následovalo přemapování odstínů šedi a výsledkem byly černé pixely hranic parcel a bílý podklad. Vektorizace potom probíhala automaticky, přičemž uživatel musel nejprve tečkou označit stavební parcely.

Na základě půdorysů následně proběhlo generování 3D modelů. Výšky budov zadával modelář ručně, pokud byly známy. Pokud ne, byly přiděleny náhodné hodnoty. Na příkladu rekonstrukce města Koblenz na obr. 14 je tak jasně patrné již výše v textu uvedené specifikum procedurálního modelování. Scéna je skutečnému městu v daném období velice podobná, řada generovaných modelů se ale od předloh zásadně liší. To nemusí být pro účely vizualizace problém, je s tím ale nutno počítat. Autoři článku si všímají toho, že nejdůležitějším faktorem ovlivňujícím podobnost vizualizace s archivními pohlednicemi je silueta města. Pro dosažení věrohodnosti je tedy nutno co nejlépe vystihnout výšku budov a vzhled střech. Toho dosahují například přidáním charakteristických detailů, jakým jsou v případě Koblenze střešní vikýře v několika řadách nad sebou. Věrohodnost scény také samozřejmě zvyšuje přítomnost klasicky v CAD softwaru modelovaných významných budov (zde kostely s jasně viditelnými věžemi).



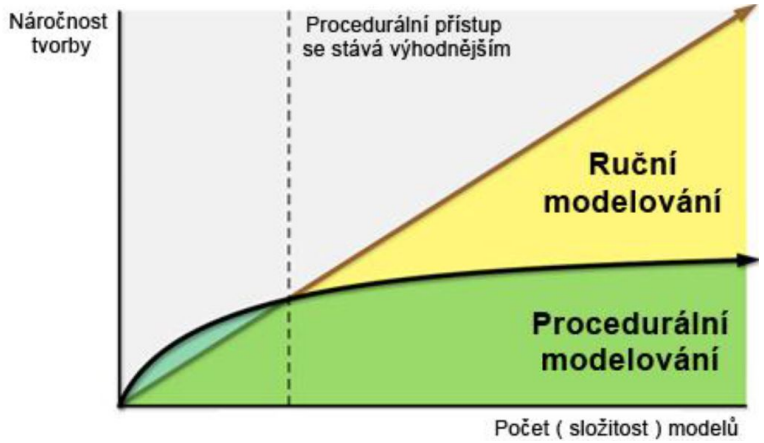
Obr. 15: 4D scéna zachycující vývoj zájmové historické lokality [12]

Závěr

Cílem této semestrální práce bylo shrnout dosavadní poznatky o využití procedurálního přístupu pro modelování památkových objektů a jejich okolí. Teoretický úvod byl zaměřen na popis tvarové gramatiky CGA, která se zdá být nejpoužívanějším nástrojem pro generování modelů. Je to dáno zřejmě zejména tím, že byla již od počátku navrhována právě pro tvorbu modelů architektury a v rámci programu City

Engine je ihned připravena k použití bez toho, aby uživatel musel vyvíjet vlastní postupy. Přitom ale lze i v nedávné době nalézt práce, které využívají pro procedurální popis památkových objektů jiné, na CGA nezávislé metody.

Hlavní výhodou procedurálního přístupu je určitě rychlost a efektivita generování modelů rozsáhlých komplexů budov s velkou úrovní detailu. Při tvorbě přesných modelů samostatných staveb se již časová výhodnost oproti klasickému přístupu ztrácí. Procedurální modelování však lze využít i pro takové účely a výhodou může být zachování textového popisu památky v souboru pravidel a snadnější pozdější přidávání detailu. Vzhledem k důrazu na strukturu modelu a sémantiku se potom nabízí otázka, zda by nešlo tohoto přístupu využít pro tvorbu informačních modelů budov na základě 2D CAD dokumentace.



Obr. 16: Porovnáni náročnosti modelování [3] podle [9]

Na druhou stranu budou procedurální modely rozsáhlých měst vždy zachycovat skutečnou podobu jednotlivých budov pouze přibližně. Při snaze vytvořit zcela přesný model by časová náročnost modelování již nebyla výhodná. Klasickým postupem je tak rozdělení města na několik typů budov, kdy každý typ má určitý charakteristický vzhled. V historickém městě se může jednat například o typ obytný dům, typ chrám, lázně, atd. V rámci jednotlivých typů dochází k drobným procedurálním obměnám, základní rysy kategorie ale zůstávají zachovány. V případě, že se snažíme zachytit ráz určité lokality, bude takový zjednodušený přístup plně dostačovat a to zejména tehdy, je-li historická podoba místa známá pouze z archeologických hypotéz.

Pokud ale jsou k dispozici například historické fotografie města, je nutno se smířit s tím, že uživatel odhalí při detailním pohledu rozdíly ve vzhledu budov na fotografiích a ve 3D vizualizaci. Řešením může být modelování charakteristických detailů a vhodná úprava nejvíce viditelných rysů, tedy výšky budov a vzhledu střech. Při

dostatečném množství podkladových informací by mohly být každé budově přidány atributy, jako je právě výška a typ střechy a dále materiál, vzhled a barva fasády, velikost oken, atd. Tyto atributy by mohly být využity jako parametry při generování a výsledný model by měl předloze odpovídat výrazně lépe. Budou-li ale informace chybět, bude je stejně nutno generovat náhodně.

Na závěr je nutno uvést, že řada 3D modelovacích softwarů obsahuje rozhraní využívající skriptovací jazyk, které lze použít pro automatizaci většiny jinak manuálně prováděných operací. Příkladem může být SketchUp Ruby API nebo MAXScript v programu 3ds Max. Pokud bychom se inspirovali principy tvarové gramatiky, bylo by možné generovat jednoduché modely budov například i s využitím Ruby API v programu SketchUp. Na potíže by ale takový přístup zřejmě narazil u složitějších modelů a tehdy, pokud bychom chtěli budovy umísťovat na podrobný digitální model terénu.

Literatura

[1] HAEGLER, Simon, Pascal MÜLLER a Luc Van GOOL. Procedural Modeling for Digital Cultural Heritage. *EURASIP Journal on Image and Video Processing* [online]. 2009, č. 1 [cit. 11. leden 2016]. ISSN 1687-5281. Dostupné z: doi:10.1155/2009/852392

[2] MÜLLER, Pascal, Peter WONKA, Simon HAEGLER, Andreas ULMER a Luc VAN GOOL. Procedural modeling of buildings. *ACM Transactions on Graphics* [online]. 2006, roč. 25, č. 3, s. 614 – 623 [cit. 11. leden 2016]. ISSN 0730-0301. Dostupné z: doi:10.1145/1141911.1141931

[3] MERZ, Otakar. Vytváření formálních popisů budov. Praha, 2012. Bakalářská práce. České vysoké učení technické v Praze, Fakulta elektrotechnická, katedra počítačů. Vedoucí práce: RNDr. Elena Dušková.

[4] PARISH, Yoav I. H. a Pascal MÜLLER. Procedural Modeling of Cities. In: *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques* [online]. New York, NY, USA: ACM, 2001, s. 301–308 [cit. 11. leden 2016]. SIGGRAPH '01. ISBN 1-58113-374-X. Dostupné z: doi:10.1145/383259.383292

[5] MULLER, Pascal. Procedural Modeling of Cities. In: *ACM SIGGRAPH 2006 Courses* [online]. New York, NY, USA: ACM, 2006, s. 139–184

[cit. 11. leden 2016]. SIGGRAPH '06. ISBN 1-59593-364-6. Dostupné z: doi:10.1145/1185657.1185716

[6] WONKA, Peter, Michael WIMMER, François SILLION a William RIBARSKY. Instant Architecture. In: *ACM SIGGRAPH 2003 Papers* [online]. New York, NY, USA: ACM, 2003, s. 669–677 [cit. 11. leden 2016]. SIGGRAPH '03. ISBN 1-58113-709-5. Dostupné z: doi:10.1145/1201775.882324

[7] EDVARDSSON, Kristinn Nikulás. 3d GIS modeling using ESRI´s CityEngine, A case study from the University Jaume I in Castellon de la Plana Spain. B.m., 2013. University Jaume I in Castellon de la Plana Spain.

[8] CALOGERO, Erica, Jaime KAMINSKI a David ARNOLD. Using procedural modeling to explore alternative designs for the louvre. *Journal on Computing and Cultural Heritage (JOCCH)* [online]. 2013, roč. 6, č. 4, s. 1 – 22 [cit. 11. leden 2016]. ISSN 1556-4673. Dostupné z: doi:10.1145/2532630.2512883

[9] ESRI R&D CENTER ZURICH. *City Engine Help* [online]. 2015 [cit. 11. leden 2016]. Dostupné z: http://cehelp.esri.com/help/index.jsp

[10] WATSON, B., P. MULLER, P. WONKA, C. SEXTON, O. VERYOVKA a A. FULLER. Procedural Urban Modeling in Practice. *IEEE Computer Graphics and Applications* [online]. 2008, roč. 28, č. 3, s. 18–26. ISSN 0272-1716. Dostupné z: doi:10.1109/MCG.2008.58

[11] RODRIGUES, N., L. MAGALHAES, J. MOURA a A. CHALMERS. Reconstruction and generation of virtual heritage sites. *Digital Applications in Archaeology and Cultural Heritage* [online]. 2014, roč. 1, č. 3–4, s. 92–102 [cit. 12. leden 2016]. ISSN 2212-0548. Dostupné z: doi:10.1016/j.daach.2014.06.003

[12] LAYCOCK, R. G., D. DRINKWATER a A. M. DAY. Exploring Cultural Heritage Sites Through Space and Time. *Journal on Computing and Cultural Heritage (JOCCH)* [online]. 2008, roč. 1, č. 2, s. 11:1–11:15 [cit. 12. leden 2016]. ISSN 1556-4673. Dostupné z: doi:10.1145/1434763.1434768

[13] LONDONCHARTER.ORG. *The London Charter for the Computer-Based Visualisation of Cultural Heritage* [online]. 2009 [cit. 12. leden 2016]. Dostupné z: http://www.londoncharter.org/

[14] DENARD, Hugh. *A New Introduction to The London Charter* [online]. 2012 [cit. 12. leden 2016]. Dostupné z: http://www.londoncharter.org/fileadmin/templates/main/docs/ch6_denard.pdf