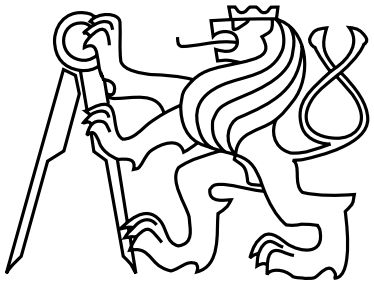
Zadání

Analyzujte možná vstupní data pro poloautomatické generování testového prostředí pro simulátor auta z mapových podkladů. Prostudujte a navrhněte převodní postupy pro vybrané druhy dat (po dohodě s vedoucím) a navrhněte formát ukládání dat pro interní použití generátoru. Implementujte modulární převodní řetězec, který pro zadanou oblast na mapě (např. pomocí GPS souřadnic) vygeneruje realisticky vypadající 3D model prostředí. Celý převodní řetězec může být poskládaný z vhodných existujících nástrojů, ale měl by být propojen do jednoduše použitelného celku. Jednotlivé moduly řetězce budou zodpovědné za konverzi/generování určitého typu dat a jejich vylepšení bude tedy probíhat lokálně bez ovlivnění zbytku řetězce. Cílové použití dat bude v simulátoru autonomního vozidla postaveném na platformě Unreal Engine 4 (např. CARLA, AirSim).

Řešení ověřte přovodem vybrané části mapy o rozloze 5x5km v okolí malého města (lokalitu vyberte po dohodě s vedoucím) a vygenerováním minimálně dvou jednominutových videosekvencí z průjezdu (jedna skrz město, druhá po okolí). Zhodnoťte kvalitu převodu z pohledu realističnosti, například srovnáním vygenerovaných videosekvencí s panoramatickými fotografiemi z google-street view.

- 1] Roman Janovský. Operátorské stanoviště pro vizualizaci a řízení autonomních bezpilotních prostředků. Diplomová práce ČVUT, FEL 2017.
- 2] Alexey Dosovitskiy, et al. CARLA: An open urban driving simulator. In Proceedings of the 1st Annual Conference on Robot Learning, volume 78 of Proceedings of Machine Learning Research - PMLR, Nov 2017.
- 3] Shital Shah, et al. AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles. Field and Service Robotics 2017.
- 4] E. Galin, et. al. Procedural Generation of Roads. Eurographics 2010.
- 5] CityEngine: http://www.esri.com/software/cityengine

České vysoké učení technické v Praze
Fakulta elektrotechnická
Katedra počítačové grafiky



Diplomová práce

Procedurální generování 3D modelu dle mapových podkladů

Bc. Jana Kejvalová

Vedoucí práce: Ing. David Sedláček Ph.D.

Studijní program: Otevřená informatika, Magisterský

Obor: Počítačová grafika

7. ledna 2019

Poděkování

Na tomto místě bych chtěla poděkovat vedoucímu práce, panu Ing. Davidu Sedláčkovi, Ph.D., za ochotu, cenné rady a vedení. Dále chci poděkovat svému Alešovi, Aničce, rodině, přátelům a Žárovkám za podporu, nekonečnou trpělivost a tolik potřebnou motivaci.

Prohlášení

Prohlašuji, že jsem práci vypracovala samostatně a použila jsem pouze podklady uvedené v přiloženém seznamu.
Nemám závažný důvod proti užití tohoto školního díla ve smyslu §60 Zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon).

V Praze dne 4. 1. 2019 .....

Abstract

The aim of this diploma thesis is to design a conversion tool that generates a realistic 3D model of the environment for the chosen area on the map.

The following two approaches were used for the solution: The first one with the use of open source tools only and the other using the CityEngine program and grammatical rules to create realistic scenes further modified by Python scripts.

This thesis includes implementation of both the approaches, which are also described in detail. Recommendations for possible improvements are included.

The second approach which uses the CityEngine program is described more in detail, as the resulting scenes are more realistic. The program uses grammatical rules to generate objects, making creation of the resulting scene automated and thus easier.

Both approaches were verified on a 5x5 km map view and visually compared to google street view and panorama mode of the map server mapy.cz.

Further development and extension of this thesis is expected.

Abstrakt

Cílem této diplomové práce je navrhnout převodní řetězec, který pro vybranou oblast na mapě vygeneruje realisticky vypadající 3D model prostředí.

Pro řešení byly použity následující dva přístupy: První pouze s pomocí open source nástrojů, druhý pomocí programu CityEngine s využitím gramatik na vytvoření realistických scén dále upravených skripty v jazyce Python.

Práce obsahuje implementaci obou výše uvedených přístupů. Oba přístupy jsou detailně popsány, včetně doporučení na případné vylepšení.

Detailněji je rozpracovaný postup využívající program CityEngine, jelikož výsledné scény jsou realističtější. Program využívá ke generování objektů gramatická pravidla, díky čemuž je vytvoření výsledné scény automatizované, a tím i snažší.

Řešení obou přístupů bylo ověřeno na výřezu z mapy s rozsahem 5x5 km a bylo vizuálně srovnáno s google street view a s režimem panorama v rámci mapy.cz.

Předpokládá se následný vývoj a další rozšiřování této práce.

Obsah

Seznam obrázků

Kapitola 1

Úvod

Díky technologickému pokroku jsou dnešní automobily a vozidla vybavena čím dál sofistikovanějšími systémy, které mají usnadňovat práci řidiči. Existuje pět úrovní asistenčních systémů definovaných v dokumentu organizace SAE (Společnost automobilových inženýrů). Tento dokument blíže specifikuje rozdíly mezi jednotlivými stupni autonomity. Nejnižší je první úroveň, takzvaná podpora řidiče, kam patří běžně se vyskytující systémy, jakými je například adaptivní tempomat či systém automatického brzdění. Tyto systémy řidiči ulehčují práci, neumí ale spolupracovat mezi sebou. Nejvyšší stupeň pak má číslo 5 a značí plně autonomní provoz, tedy řízení bez nutnosti zásahu či přítomnosti řidiče[11].

Autonomní vozidla se orientují pomocí počítačových systémů, které snímají okolí vozidla pomocí senzorů a na základě zjištěných údajů určují jeho další chování. Mezi používané senzory patří např. odometr pro snímání pohybu kol, laserový senzor pro snímání okolních objektů pomocí světelného paprsku, radarový snímač, ultrazvukový senzor, kamera či GPS[12].

Potenciální výhodou plně autonomních vozidel je jejich relativní bezpečnost - nejčastější příčinou dopravní nehody bývá lidská chyba. Stroj, narozdíl od člověka, dokáže vyhodnotit případnou nebezpečnou situaci a rychleji na ni reagovat. Další předpokládanou výhodou je zvýšení pohodlí a časová úspora pro řidiče. Jelikož se jedná o mediálně atraktivní téma, jednotlivé automobilky se předhánějí v tom, která dokáže jako první vyvinout plně automatizované vozidlo.

Součástí tohoto vývojového procesu je i snaha vyvinout software, který bude přijímat data ze senzorů, vyhodnocovat je a na základě výsledků se bude rozhodovat. Základní algoritmy jako sledování jízdního pruhu nebo zastavování na semaforech jsou jednoduše testovatelné například na testovacích tratích nebo i v reálném provozu. Vozidlo se ale potřebuje naučit i náročnější úkony, například detekci překážek na silnici a vyhodnocení situace (brzděním nebo úhybným manévrem). Pro testování rozhodovacích algoritmů se proto používá simulace, kde lze vytvořit potřebný scénář a testovat chování vozidla například v situaci, kdy do vozovky náhle vběhne dítě.

Cílem této diplomové práce je navrhnout postup, díky kterému si uživatel bude moci zvolit oblast na mapě (například nepřehlednou křižovatku) a nechat si vygenerovat její model. Tyto modely následně mohou být využívány při testování rozhodovacích algoritmů pro autonomní vozidla. To ale již není součástí této práce.

Kapitola 2

Popis problému, specifikace cíle

Tato kapitola se zabývá popisem přístupu a definicí kartografických termínů. Pro potřeby diplomové práce je nutné zavést kartografické pojmy a přístupy zpracování mapových podkladů.

2.1 Přístup

Součástí zadání je vytvoření modulárního nástroje. Pro generování reálného světa je vhodné použít procedurální algoritmy, aby se minimalizovala manuální práce a co možná nejvíce funkcí pracovalo automaticky.

2.1.1 Modulární programování

Nazývá se též "návrh shora-dolů". Jedná se o návrh softwaru rozdělující program na nezávislé zaměnitelné moduly. Výhodou je možnost úpravy jednotlivých částí bez nutnosti opravovat ostatní. Každý modul se stará jen o část programu a neovlivňuje další. Jednotlivé moduly jsou do kompletního programu instalovány pomocí rozhraní, které určuje, co má který modul na starosti, jaké jsou jeho vstupy a výstupy.

Pojmem moduly se v této práci rozumí jednotlivé části získávání dat, konverze do potřebných formátů, upravování dat podle typu, vykreslování a práce s jednotlivými vrstvami mapových podkladů (např. budovy, silnice, řeky...).

2.1.2 Procedurální generování

Jedná se o způsob generování dat, aby nemusela být vytvářena ručně. Automatické generování je umožněno pomocí algoritmů, pravidel či gramatik. Výhodou je, že lze vygenerovat velké množství obsahu s minimální ruční prací, případně se tímto přístupem dá uspořit paměť a velikost souborů. Místo ukládání celých map a scén lze na základě zadaných pravidel vytvořit svět s generovanými objekty[66][2]. Procedurální generování se používá například u počítačových her[63], kde se mapa může vytvářet náhodně a hráč má při každém průchodu hrou jiný zážitek. Samozřejmě se procedurální generování nevyužívá pouze při vytváření počítačových her, lze jím vytvořit podklady pro filmy, ať už ve formě rozmanitých měst či

davů a armád, podklady pro vizualizaci architektury či k simulaci chování v reálném světě. Například při tvorbě filmu Pán prstenů byl použit software `MASSIVE`[49] umožňující vytvořit statisíce animovaných postav[50] bez nutnosti obsazení lidí, či jejich kopírování v postprodukci.

2.2 Práce s mapami

Pro upřesnění termínů, které se v souřadnicových systémech používají, následuje krátký výčet používaných obratů a definic. Všechny níže uvedené definice se vztahují k normě `EN ISO 19111`[30]. Po zavedení pojmů jsou vysvětleny principy zobrazení mapových podkladů a popsány vybrané systémy použité v této práci. Pro vyhledávání konkrétních EPSG kódů byl použit registr `http://www.epsg-registry.org/`.

2.2.1 Definice termínů

- **EN ISO 19111** je ISO norma platná od 1. 10. 2011, týkající se reprezentace geografické informace. Stanovuje schéma vyjádření prostorových referencí souřadnicemi. Popisuje k tomu potřebné datové prvky, vztahy a přidružená metadata[53].
- **datem** se rozumí parametr nebo soubor parametrů, který vymezuje polohu počátku, měřítko a orientaci souřadnicového systému
- **geodetickým datem** se rozumí datum popisující vztah souřadnicového systému k Zemi
- **souřadnicovým systémem** se rozumí matematická pravidla pro stanovení, jak se mají souřadnice přidružit k bodům
- **souřadnicovým referenčním systémem** se rozumí souřadnicový systém, který je vztažen k reálnému světu datem, v souladu s `EN ISO 19111`. Tato definice zahrnuje souřadnicové systémy vycházející z geodetických nebo kartézských souřadnic a souřadnicové systémy vycházející z kartografických zobrazení,
- **kartografickým zobrazením** se rozumí změna souřadnic založená na jednoznačném vztahu, od geodetického souřadnicového systému do roviny na základě stejného data
- **složeným souřadnicovým referenčním systémem** se rozumí souřadnicový referenční systém, který pro popis polohy používá dva další nezávislé souřadnicové referenční systémy, jeden pro horizontální složku a jeden pro vertikální složku
- **geodetickým souřadnicovým systémem** se rozumí souřadnicový systém, ve kterém je poloha vymezena geodetickou šířkou, geodetickou délkou a (v trojrozměrném případě) elipsoidickou výškou
- **EPSG kódem** je rozumí unikátní celé nezáporné číslo vyjma nuly, které odkazuje na databázi zemských elipsoidů, geodetických dat, zeměpisných a kartografických souřadnicových systémů a měrných jednotek. Každé kartografické zobrazení má jedinečný kód[28].

2.2.2 Souřadné systémy

Informace o souřadných systémech jsou vybrané ve vztahu k České Republice. Na světě je mnoho dalších zobrazení, která ale ve výčtu chybí. Následuje seznam systémů, se kterými se pracuje v této práci:

S-JTSK

Systém jednotné trigonometrické sítě katastrální. S tímto systémem je možné se setkat v datech poskytnutých Zeměměřickým úřadem[23]. S-JTSK je závazným geodetickým referenčním systémem na území ČR. V prohlížecích a stahovacích službách jsou podporovány souřadnicové referenční systémy reprezentované EPSG kódy 5514 (S-JTSK / Krovak East North) a 5221 (S-JTSK (Ferro) / Krovak East North) s matematickou orientací souřadnicových os (osa `x` směřuje na východ, osa `y` směřuje na sever, na území ČR jsou obě souřadnice záporné)[24].

WGS84

Světový geodetický systém 1984. WGS84 je závazným geodetickým referenčním systémem na území ČR. V prohlížecích a stahovacích službách jsou podporovány souřadnicové referenční systémy EPSG kódy 4326 (WGS 84), 32633 (WGS 84 / UTM zone 33N - základní poledník 15°), 32634 (WGS 84 / UTM zone 34N - základní poledník 21°) a 3857 (WGS 84 / Pseudo-Mercator)[24].

2.2.3 Reprezentace map

Mapové zobrazení neboli kartografická projekce je způsob, jakým se převádí zobrazení povrchu Země ze zakřiveného trojrozměrného povrchu do roviny. Vzhledem k tomu, že povrch Země je aproximován koulí či elipsoidem, a ani tyto zjednodušené geometrické tvary nejsou rozvinutelné do roviny, je zapotřebí využít objekt, který rozvinutelný je. Převod kulové plochy na válec či kužel ale vždy představuje částečné zkreslení informace. Není možné zachovat zároveň délky, úhly a plochy. Vždy dochází ke zkreslení alespoň jednoho z těchto údajů.

Z toho důvodu byla vyvinuta řada mapových zobrazení kladoucí důraz na různé údaje. Při vytváření map menších území (do několika set m^2) je zkreslení zanedbatelné.

Existují čtyři typy mapového zobrazení podle kartografického zkreslení[48][37]:

- **ekvidistantní** - nezkresluje vzdálenosti v určitém směru
- **ekvivalentní** - nezkresluje poměr ploch, zkresluje úhly
- **konformní** - nezkresluje úhly, zkresluje plochy
- **vyrovnávací** - zobrazení je záměrně spočítáno tak, aby zkreslení úhlů a ploch bylo pokud možno v rovnováze

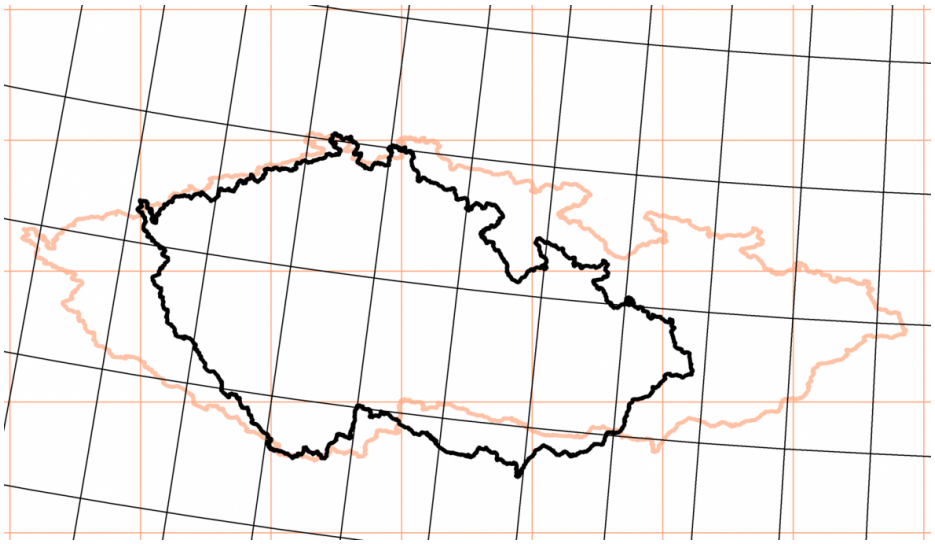
Křovákovo zobrazení

Křovákovo zobrazení[37][38] je druhem konformního kuželového zobrazení. Zemský povrch je zobrazen na kuželu, který je v obecné poloze. Toto zobrazení se používá jako závazné pro všechna státní mapová díla v České Republice a na Slovensku. Ukázka zobrazení je na obrázku 2.1.

Byl vytvořen českým geodetem Ing. Josefem Křovákem v roce 1922.

Používá souřadnicový systém S-JTSK. Zobrazení zachovává úhly a délky ve dvou kartografických rovnoběžkách.

Křovákovo zobrazení nezobrazuje sever na horní okraj mapy, ale mírně odchýlený doprava.



Obrázek 2.1: Česká republika ve Křovákově zobrazení (černě) a WGS84 (oranžově)[38]

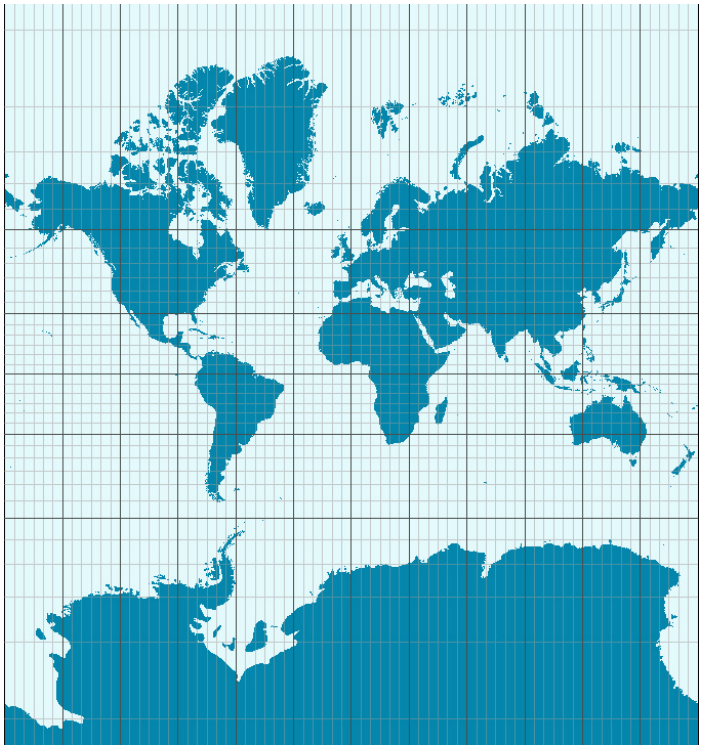
Mercatorovo zobrazení

Mercatorovo zobrazení[37][51] je druhem konformního válcového mapového zobrazení. Základem zobrazení je válec dotýkající se glóbu na rovníku. Po zobrazení povrchu koule na válec a po rozvinutí pláště válce do roviny vznikne pravoúhlá síť poledníků a rovnoběžek. Blízko u pólů je zkreslení v délce větší. Ukázka zobrazení je na obrázku 2.2.

V roce 1569 toto zobrazení navrhl vlámský kartograf Gerhard Mercator.

Používá se pro rovníkové oblasti a naopak není vhodné ho použít pro póly. Proto se nehodí pro přehlednou mapu světa. Zobrazení zachovává úhly, ale silně zkresluje plochy.

V souvislosti s Mercatorovým zobrazením se používá i UTM (Universal Transverse Mercator). Jedná se o síť 60 zón zobrazených pomocí Mercatorova zobrazení. V případě, že body leží ve stejné zóně lze měřit jejich vzdálenosti s pomocí Pythagorovy věty. Střed souřadnic je pro každou zónu jiný a tvoří jej průsečík středového poledníku zóny s rovníkem. Od tohoto středu se měří vzdálenosti v metrech po ose `x` rostoucí od středového poledníku směrem na východ (tzv. eastings) a po ose `y` rostoucí od rovníku směrem na sever (tzv. nordings). Systém vyvinula armáda USA a je založen na WGS 84[37].



Obrázek 2.2: Mercatorovo zobrazení světa pro 86. stupeň severní a jižní zeměpisné šířky[51]

2.3 Cíl

Tato diplomová práce řeší konverzi dat z existujících mapových podkladů do 3D modelů ve scéně. Pro dosažení cíle práce je potřeba vybrat vhodnou reprezentaci mapových podkladů spolu se správnými geografickými metadaty. Jelikož má práce odrážet skutečný svět, je třeba se zaměřit na převody mezi soustavami a zobrazeními a zároveň dodržet modularitu programu.

Kapitola 3

Analýza a návrh řešení

Cílem diplomové práce je sestavit nástroj, který vytvoří model z výřezu z mapy. Testovaný úsek je okolí Týnce nad Sázavou, jehož rozloha je přibližně 20x20km. Okolí Týnce bylo vybráno pro svou rozmanitost. Město se nachází v kopcovitém terénu, obsahuje jak místní malé silnice, tak větší pozemní komunikace. V okolí se nachází různé zdroje vody - větší řeka Sázava, menší potoky a rybníky.

Finální podoba nástroje by měla získat pouze informace o souřadnicích. Veškeré mapy a úpravy by se měly vytvořit automaticky. Převod může být částečně interaktivní a vyžadovat akci uživatele. Obrázek 3.1 zobrazuje zadaný výřez mapy.

(a) mapy.cz

(b) openstreetmap.org

Obrázek 3.1: Výřez mapy ze serveru mapy.cz (vlevo) a ze serveru openstreetmap.org (vpravo)

3.1 Mapové podklady a data

Je zapotřebí, aby použitá data byla obsáhlá, kompletní a poskytující služba se neustále vyvíjela. Staré a již neaktualizované projekty se pro potřeby diplomové práce nehodí. V následující části jsou uvedeny existující mapové podklady a data, ze kterých by modulární nástroj mohl čerpat.

3.1.1 Geografická data

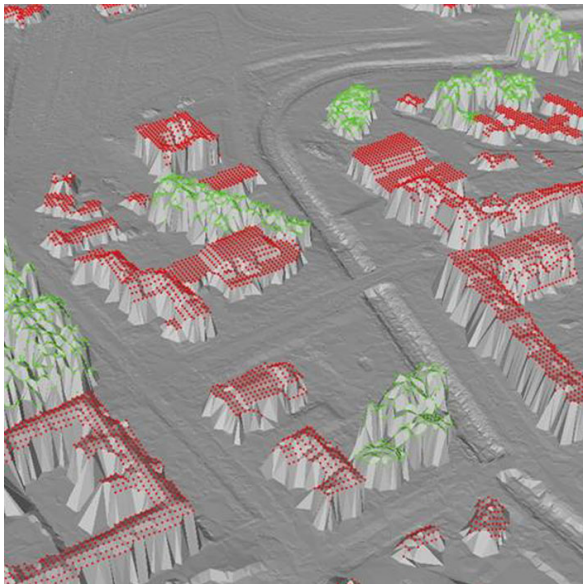
Grafická data se dělí na vektorová a rastrová data. Vektorová data jsou popsána pomocí geometrických parametrů. U rastrových dat je objekt popsán jako shluk bodů. Úsečka popsaná rastrem je množina bodů o různých souřadnicích, které dohromady tvoří obraz úsečky. V případě úsečky ve vektorovém formátu bude uložena pouze informace o počátečním a koncovém bodě, o barvě a tloušťce.

Geografická data jsou taková data, která nesou geografickou informaci. Mohou být vektorová (např. `DXF` - Drawing Exchange Format) nebo rastrová (např. `GeoTIFF`). Následuje seznam vybraných geografický dat.

3.1.1.1 Geoportál ČÚZK

Některá data pro zpracování práce zapůjčil Zeměměřický úřad[24]. Jedná se o Digitální model reliéfu České republiky 5. generace (dále DMR 5G) a Digitální model povrchu České republiky 1. generace (dále DMP 1G).

Digitální model povrchu České republiky 1. generace obsahuje data v nepravidelné síti výškových bodů s úplnou střední odchylkou 0,4 m pro budovy a 0,7 m pro objekty přesně neohraničené, jakými jsou např. lesy a rostlinný pokryv. Používá se např. v kombinaci s DMR 5G pro zjištění výškových poměrů terénu a geografických objektů. Data jsou v souřadnicovém systému S-JTSK a Krovak EN. Ukázka dat je na obrázku 3.2.



Obrázek 3.2: Ukázka DMP 1G[24]

Digitální model reliéfu České republiky 5. generace zobrazuje zemský povrch opět v nepravidelné trojúhelníkové síti bodů se souřadnicemi `x`, `y` a `h`, kde `h` reprezentuje nadmořskou výšku s úplnou střední odchylkou 0,18 m v odkrytém terénu a 0,3 m v zalesněném terénu. DMR 5G se používá např. při projektování pozemkových úprav, též je základním zdrojem pro tvorbu vrstevnic. Data opět využívají souřadnicový systém S-JTSK a anglickou verzi Křovákova zobrazení.

3.1.1.2 OpenStreet map

Jedním z nejrozšířenějších mapových podkladů jsou `OpenStreet mapy`[61]. Jsou uloženy ve formátu `.osm` obsahující `.xml` obsah. Tvoří je komunita uživatelů, kteří přidávají a udržují data o silnicích, cestách, kavárnách, železničních stanicích a mnohých dalších po celém světě. Klade se důraz na místní znalost. Přispěvatelé využívají letecké snímky, GPS přístroje i klasické mapy, aby ověřili, že `OSM` je přesné a akuální[55].

Jednotlivé části krajiny se vykreslují pomocí reprezentativních prvků do mapy. Základní prvky `OSM` dat jsou:

- **Body** (Node) - bod označuje polohu objektu. Pomocí dvou bodů se zakreslují úsečky.
- **Cesty** (Way) - cesty tvoří seznam bodů zobrazených jako propojené úsečky. Pomocí nich se zakreslují silnice, ulice, atd.
- **Uzavřené cesty** nebo **Plochy** (Closed Way/Area) - pomocí uzavřených cest se zakreslují plochy, jako například parky, lesy, rybníky či budovy. Jedná se o do kruhu spojené cesty.
- **Relace** (Relation) - spojují dohromady cesty, které nereprezentují stejné fyzické objekty. V relaci je napsáno, jakou roli každá z obsažených cest má. Pomocí ní se v mapě vytváří popisy omezení odbočení na křižovatkách nebo turistické trasy. Velmi dlouhé cesty (např. mezinárodní silnice, řeky) se skládají z více kratších částí, které se pomocí relace spojují do jednoho celku.

Ke všem výše uvedeným prvkům se dají přidávat značky (Tags), kterými lze popsat jejich název, druh budovy, silnice, atd.

Vytváření geodat pro databází `OSM` je proces, ve kterém se modeluje generalizovaný obraz reálného světa. Modelovaný svět se skládá z objektů, které jsou v modelu reprezentovány výše uvedenými základními prvky.

Výhodou `OpenStreet map` je jednoduchý způsob exportu - stačí ve webovém rozhraní ručně vybrat oblast na mapě a daný úsek uložit ve formátu `.osm`, který obsahuje `.xml` obsah dobře čitelný pro strojové zpracování. Nevýhodou je na druhé straně nemožnost stáhnout větší a komplexnější části. Pro stahování větších celků je doporučováno stažení map z předpřipravených databází jakými jsou `Planet OSM`[64] nebo `Geofabrik`[32]. `OpenStreet mapy` fungují pod ODbL licencí dovolující uživatelům svobodně sdílet, upravovat a používat databázi za podmínky poskytnutí stejné svobody ostatním uživatelům[54].

Příklad formátu `.osm` vypadá následovně:

```
<?xml version="1.0" encoding="UTF-8"?>
<osm version="0.6" generator="JOSM">
    <node id="42" lat="50.1097431" lon="14.4741222"  visible="true" version="1" />
        <tag k="name" v="Death Valley" />
        <tag k="natural" v="desert" />
        <tag k="update" v="5. 7. 2017" />
    </node>
</osm>
```

`OpenStreet mapy` mají své API[57], díky kterému je možné proces exportu a přístup k databázi spravovat pomocí příkazové řádky, tedy automatizovat. Aktuálně používaná verze API v0.6 je z dubna roku 2009.

3.1.1.3 SRTM

Shuttle Radar Topography Mission je mezinárodní výzkumný projekt, jehož cílem je vytvoření nejkompletnější digitální topografické databáze. Mise SRTM sestávala ze speciálně upravených radarových systémů nacházejících se na palubě raketoplánu Endeavour během 11 denní mise STS-99 v únoru 2000. Měřící zařízení se skládalo ze dvou antén, jedné umístěné v nákladovém prostoru raketoplánu a druhé umístěné na 60 m dlouhém stožáru vztyčeném do volného prostoru z nákladního prostoru, jak je vidět na obrázku 3.3.

SRTM data obsahují údaje nadmořské výšky ve vysokém rozlišení[73]. Výškový model je převeden na jednotlivé dlaždice pokrývající vždy jeden stupeň zeměpisné šířky a jeden stupeň zeměpisné délky. Data jsou dostupná ve dvou úrovních:

- SRTM1 pokrývající území Spojených států a jejich teritorií a držav. Má rozlišení jednu úhlovou minutu (anglicky minute of angle), neboli šedesátinu úhlového stupně. Kolem rovníku je přesnost na 30 metrů. Jedna dlaždice obsahuje 3601 řádků s 3601 16 bitovými buňkami ve formátu bigendian.
- SRTM3 je úroveň pro zbytek světa (Afrika, Austrálie, Eurasie, Ostrovy, Severní a Jižní Amerika) má rozlišení 3 úhlové minuty. Tato data jsou generována průměrem ze tří jednominutových vzorků. Taková dlaždice obsahuje 1201 řádků a 1201 vzorků.

Původní `SRTM` data byla vypočítána ve WGS84 (anglicky new World Geodetic System), následně používala formát EGM96 (Earth Gravitation Model). Data jsou na internetu volně ke stažení[26] a jsou pojmenována podle zeměpisné šířky a délky levého dolního rohu dlaždice. Pro Týnec nad Sázavou je zapotřebí dlaždice 49N14E. Data jsou ve formátu `.hgt`.

3.1.2 Mapové dlaždice

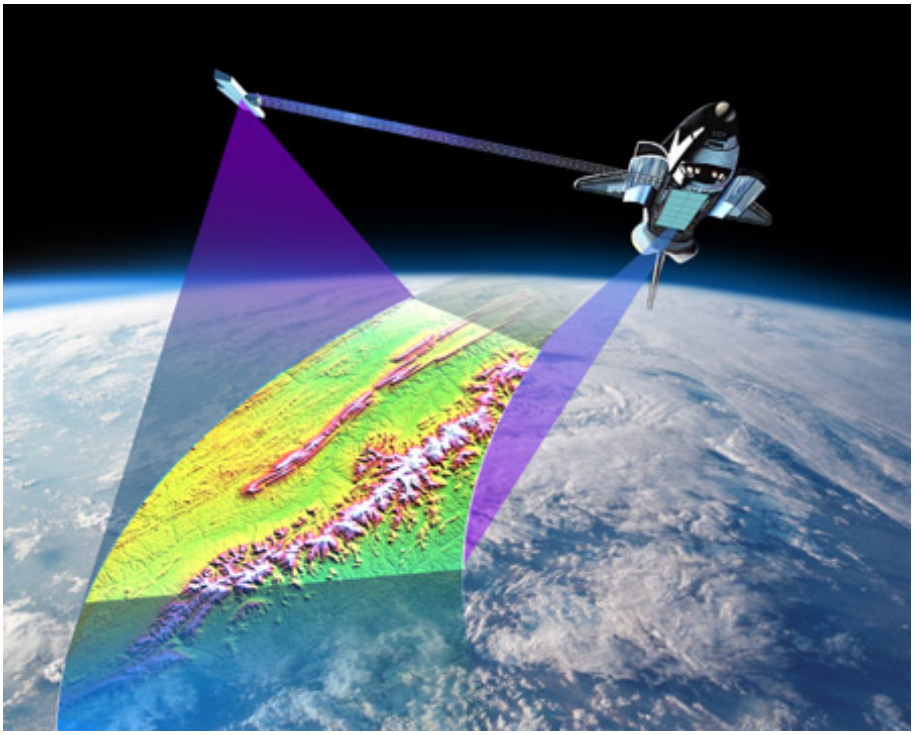
Webové služby pro zobrazování map využívají mapových dlaždic. Každá dlaždice má 256x256 pixelů. Při každém přiblížení je jedna dlaždice 256x256 pixelů nahrazena čtyřmi dlaždicemi o stejné velikosti. Princip mapových dlaždic pro úroveň 1-3 je vidět na obrázku 3.4. Dlaždice obsahující celý svět v Mercatorově zobrazení, představuje úroveň 0.

Dlaždice lze kódovat různým způsobem. Na obrázku 3.4 byl použit Quadtree, který používá čtyřkovou soustavu. Rozděluje dlaždici na čtyři kvadranty a podle pozice přiblížené dlaždice zvolí cifru. Délka quadtree je rovna přiblížení.

Následující služby poskytují možnost stahování dat pomocí dlaždic.

3.1.2.1 Mapzen

Webová služba, která umožňovala stažení vektorových dat z map je `Mapzen`[47], jehož způsob a použití jsou popsány v doporučené literatuře[3]. Bohužel služba na začátku roku 2018 (2.



Obrázek 3.3: Proces získávání dat z raketoplánu Endeavour[74]

1. 2018) vydala prohlášení, že ukončuje k 1. 2. 2018 svou činnost. Použití dat z `Mapzenu` proto již není aktuální. Služba na svých stránkách uvedla seznam jiných služeb pro získávání podkladových map či vektorových dat, jsou jimi např. Mapbox, OpenMapTiles a Thunderforest. `Mapzen` využíval vektorová data ve formátu `GeoJSON` obsahující opět informace z `OpenStreet map` spolu s georeferencovanými daty.

3.1.2.2 Mapbox

Mapbox[43] je další poskytovatel vektorových map. Data pochází od `OpenStreet map`, `NASA` a DigitalGlobe. Na Githubu lze najít několik pluginů, například knihovnu pro import mapy do prostředí `Unity 3D`[44]. Na konferenci Unite 2016 v Los Angeles[39] přednášel Eric Gundersen z `Mapboxu` o možnosti použití map v Unity. Zdůrazňoval však, že firma není herní společnost, pouze poskytuje stahování map. Používá formát `GeoJSON`, což je `JSON` obsahující určitou strukturu pro popis geografických dat (popsáno níže). `Mapbox` používá vektorová dlaždicová data, která obsahují několik vrstev od nadmořské výšky povrchu, přes vrstvu domů a silnic, až po vrstvy elektrického vedení.

Výše uvedená vektorová data se po registraci a získání API klíče na stránkách dají získat. Pokud uživatel službu nevyužívá komerčně, lze v rámci plánu `Pay as you go` stáhnout až 50000 map zdarma. Pokud uživatel tento limit překročí, za každých 1000 map platí 0.5$. Pro komerční účely se měsíčně připlácí 499$[65]. Pro účely této diplomové práce hranice 50000 map stačí.

Výškové mapy (Mapbox Terrain-RGB) obsahují globální výšková data zakódovaná v

Obrázek 3.4: Princip mapových dlaždic[45]

rastrovém formátu `.png`. Pomocí dekódování barev lze získat výšku v metrech. Každý kanál na každé pozici může dosahovat 256 hodnot, barevné kanály jsou tři (červený, zelený a modrý). Znamená to, že je možné získat 256^3 neboli 16777216 různých hodnot. Z těchto hodnot lze vypočítat výšku pomocí rovnice 3.1[13].

$$height = -10000 + ((R * 256 * 256 + G * 256 + B) * 0.1), \tag{3.1}$$

kde proměnné R, G, B představují v tomto pořadí celočíselnou hodnotu červeného, zeleného a modrého kanálu v rozsahu $< 0; 255 >$.

Pro získání dat terénu se používá následující odkaz:

```
https://api.mapbox.com/v4/mapbox.terrain-rgb/{z}/{x}/{y}.pngraw?access_token={API_KEY},
```

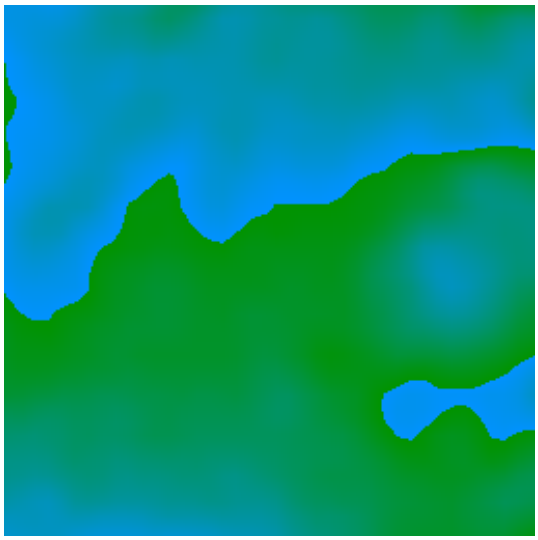
kde `x` a `y` představují pozici dlaždice na ose `x` a na ose `y`. Hodnota `z` představuje zoom a `API_KEY` představuje klíč ke službě, kterou uživatel najde po registraci ve svém osobním profilu. Ve výchozím nastavení se tímto příkazem stáhnou data v rozlišení $256x256$. Výšková data poskytují přiblížení až do hloubky 15, přičemž větší úroveň zoomu již rozlišení nezvýší.

Z Mapboxu lze získat i další druhy dlaždic, například satelitní snímky. Jak je vidět na obrázku 3.5, při stahování dat okolo Týnce se na satelitních snímcích vyskytují mraky a tudíž satelitní data v práci využita nebyla.

3.1.2.3 OpenMapTiles

Jedná se o open source software poskytující možnost upravovat si vzhled map. Přejímá data z `OpenStreet map` a převádí je na vektorové dlaždice. Projekt je pokračovatelem `OSM2VectorTiles`,

(a) Vrstva Mapbox Satellite



(b) Vrstva Mapbox Terrain-RGB

Obrázek 3.5: Ukázka dlaždice okolí Týnce ze služby Mapbox s přiblížením $zoom = 15$. Satelitní vrstva (vlevo) a výšková vrstva (vpravo).

což je služba převádějící data z formátu `.osm` na vektorové dlaždice. `OpenMapTiles` poskytuje možnost stáhnout si vektorová data s různou strukturou - klasická vektorová data z `OpenStreet Map` či linie a kontury, dále rastrová data ze satelitu či rastr povrchu země. Po registraci lze zdarma stáhnout vektorová data pro Českou Republiku ve formátu `MBTiles`, mající velikost 788.89 MB. `MBTiles` je formát pro ukládání mapových dlaždic v jednom souboru. Technicky vzato se jedná o SQLite databázi. Ukázka ze služby `OpenMapTiles` je na obrázku 3.6.

3.1.2.4 Thunderforest

Thunderforest nabízí vybrat si z více barevného rozvržení map. Poskytuje též mapy zaměřené na cyklistiku, veřejnou dopravu, dále na terén a outdoorové sporty. Dokonce, aby ukázali, že možností upravit si mapu je neomezeně, vytvořili též death metalovou mapu (viz obrázek 3.7), která je temná a veškeré vodstvo má barvu rudého ohně a některé samohlásky v názvech jsou přehláskované. Toto rozšíření není vůbec vhodné pro použití v popisovaném projektu, Thunderforest jím pouze

Obrázek 3.7: Ukázka metalového stylu map Týnce nad Sázavou od Thunderforest

Obrázek 3.6: Ukázka zobrazení mapy Týnce nad Sázavou od OpenMapTiles

velmi poutavě a názorně ukazuje, že fantazii se při úpravě mapových podkladů meze nekladou.

U všech mapových podkladů, které poskytují vektorová data jsou data složena z jednotlivých dlaždic.

3.2 Práce s osm daty

Existuje řada aplikací, které umožňují konverzi dat typu `.osm` na modelové soubory. Například programy `OSM2World`[56] či `Osmosis`[60]. Také existují aplikace, které bez nutnosti stahování dokážou zobrazit výsledný model online. Příkladem je `3D viewer`[7].

3.2.1 OSM2World

Tento program vytváří 3D modely za použití dat z `OpenStreet map`. Je možné ho použít jako samostatný nástroj či jako knihovnu v Java programech. Je publikován pod licencí LGPL[40], může být použit zdarma i v komerčním softwaru. Nabízí též možnost podílet se na jeho zlepšení. Na jejich stránkách[56] lze nalézt odkaz na Github. Aplikace má v sobě zabudovaný program `Osmosis`, který též pracuje s daty z `OpenStreet map`.

Umí zpracovávat následující otagovaná data:

- amenity=bench - lavička
- barrier=bollard - sloupek

  barrier=fence - plot

barrier=wall - zeď
barrier=hedge - živý plot

- building=garage - garáž
- building:material=brick - fasáda z cihel
  building:material=concret - fasáda z betonu
  building:material=glass - čelo pokryté sklem
  building:material=plaster - budova omítnuta
- highway=bus_stop - autobusová zastávka
  highway=street_lamp - lampa
- amenity=waste_basket - koš na odpadky
- natural=tree - strom
- roof:shape=gabled - sedlová střecha
  roof:shape=hipped - valbová střecha
- tunnel=building_passage - cesta která prochází budovou
- railway=rail - koleje

Program obsahuje i základní uživatelské rozhraní. Stačí jako parametr do příkazové řádky připsat `--gui`. Model je možné procházet či exportovat do formátů `.png`, `.pov` nebo `.obj`. Zároveň se jedná pouze o vizualizaci v nízké kvalitě. Slouží především pro účely ladění modelu.

Pokud je program obsluhován příkazovou řádkou, používají se parametry `-i/--input <file>` pro vstupní data (jimiž mohou být data ve formátu `.osm`, `.osm.gz`, `.osm.bz2` nebo `.osm.pbf` a parametr `-o/--output <file>` pro jeden až více výstupních souborů. Lze exportovat data ve formátech `.png`, `.pov` nebo `.obj`.

3.2.2 Osmosis

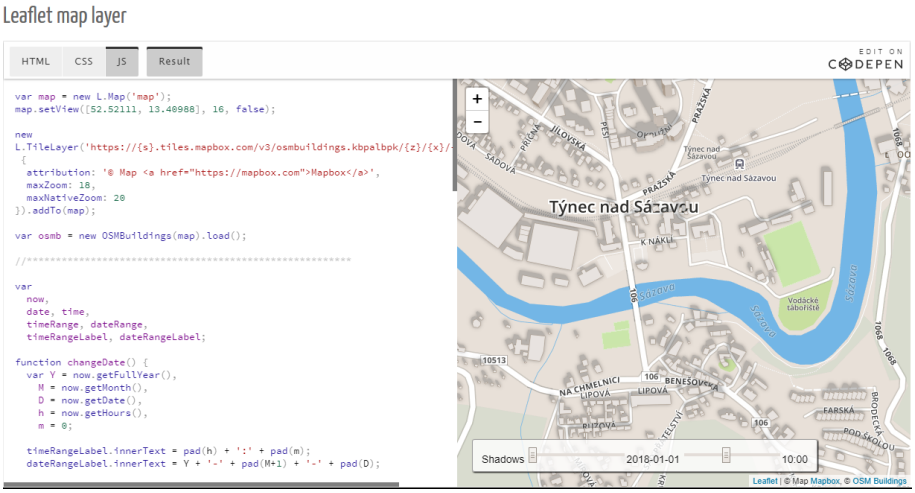
`Osmosis` je Java aplikace pro zpracování `.osm` dat fungující jednoduše z příkazové řádky. Byla navržena tak, aby se do ní snadno přidávala rozšíření a nebylo potřeba přepisovat základní úlohy, jako práce se soubory a s databází. I `Osmosis` má vlastní Github[60], kam mohou uživatelé přidávat poznámky či rozšíření. Na webu `OpenStreet map` je přímo návod, jak data zpracovávat pomocí tohoto programu.

Výhodou je, že `Osmosis` obsahuje mnoho parametrů, se kterými lze program volat. Je tak možné formát `.osm` před extrahováním upravit. Lze vybrat pouze určitá data (například pouze silnice a budovy, vynechat adresy a čísla popisná) pomocí parametrů `--tag-filter`. Takto je možné původně velký a obsáhlý soubor zmenšit pouze na požadovaná data.

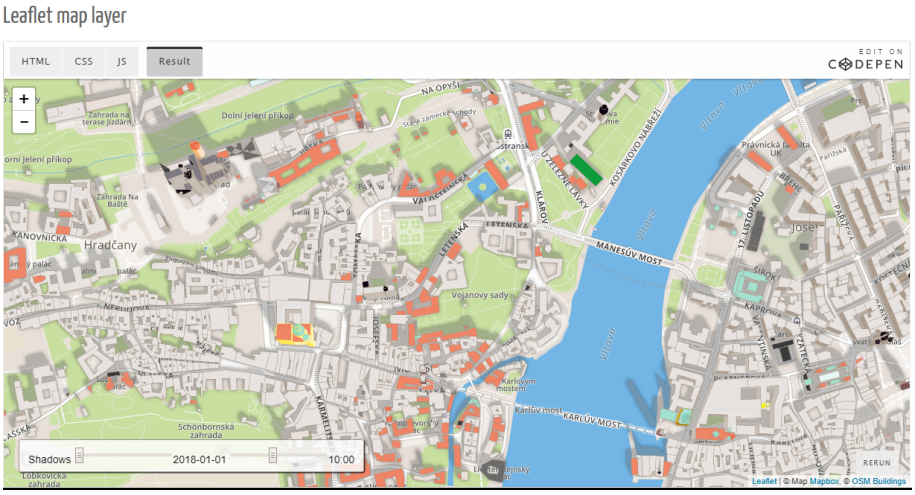
Jako jeden z argumentů lze programu poslat soubor `.poly` obsahující polygon reprezentovaný seznamem bodů. `Osmosis` pak provede pouze extrakci dat ohraničené oblasti. Velká databáze měst se nachází na GitHubu Jamese Chevaliera[35]. Databáze obsahuje i Českou republiku, přehledně rozdělenou na jednotlivé kraje a jejich města.

3.2.3 OSM Building

`OSM Building` poskytuje celosvětově data budov. Na jejich stránkách[58] lze vyzkoušet online 3D viewer. Jejich data vychází z `GeoJSONu` a vektorových dlaždic, z individuálních objektů a částí ploch. Nabízí využití zdarma při maximálně 50000 požadavcích za měsíc. Vykreslení detailů 3D budov je znatelné hlavně u větších měst. Berlín i Praha obsahují oproti Týnci mnohem více známých a zajímavějších budov, proto pro ukázku toho, co `OSM Building` umí, byla použita i mapa Prahy, viz obrázek 3.8, kde jsou vidět kostely a Pražský hrad. U Týnce tato aplikace ukazuje pouze obyčejné hranoly reprezentující budovy a není z ní proto patrné, co dokáže.



(a) openstreetmap.org



(b) 3dviewer.net

Obrázek 3.8: Ukázky OSM Building: Týnec nad Sázavou s JavaScriptovým kódem (nahoře), Praha (dole)

3.2.4 CityEngine

CityEngine[17] je pokročilý nástroj pro generování procedurálních měst. Byl vyvinut v roce 2007 firmou `Procedural Inc.` ve Švýcarsku. Firmu v roce 2011 převzala společnost `Esri`[29] a přejmenovala na `Esri R&D Center Zurich`.

CityEngine obsahuje nástroj pro import mapových dat. Od verze 2016.1 lze využít funkci `Get Map Data`, která uživateli umožňuje vybrat si oblast na mapě a program stáhne vybraná `.osm` data. Pokud je uživatel registrován i na stránkách `esri`, je možné k datům stáhnout i informace o terénu. V dřívějších verzích programu se import musel vykonávat ručně a bylo zapotřebí k obrazovým datům v různém softwaru přidávat geografická data. Jedním z používaných programů byl i další produkt firmy Esri, a to ArcGIS. V aktuální verzi už geografická data program dokáže zpracovat sám.

Nová verze vychází od roku 2014 zpravidla 2x ročně[17], nejnovější verze je `2018.1` a je velmi pravděpodobné, že v novém roce vyjde další verze, a to `2019.0`.

Následuje výčet hlavní prvků programu:

- **Procedurální modelování** - software využívá gramatiku `CGA` (computer generated architecture). `CGA` pravidla umožňují kontrolovat a upravovat geometrii objektů, velikost, textury budov nebo silnic.
- **Get Map Data** - výše zmíněný pomocník při stahování mapových podkladů. Uložená `CGA` pravidla umí pracovat s importovanými `osm` daty a jejich atributy.
- **GIS/CAD podpora** - program podporuje standardní formáty používané pro geografické informační systémy či počítačem podporované projektování (computer aided design). Jsou jimi například `Esri Shapefile`, geodatabáze či `osm` data.
- **Parametrické modelování** - rozhraní kontrolující konkrétní parametry ulic či budov, jako jsou např. výška, typ a další parametry definované v `CGA` pravidlech.
- **Dynamické rozvržení města** - interaktivní editace a modifikace měst sestávající z ulic, bloků a parcel.
- **Silniční síť** - nástroj pro rozvoj ulic pro vytváření měst.
- **Průmyslové standardní 3D formáty** - podpora Autodesk FBX, 3DS, Wavefront OBJ, Collada a dalších.
- **Export do 3D Webové scény** - model vytvořený v CityEnginu lze přímo exportovat do WebGL scény v prohlížeči.
- **Podpora virtuální reality** - z prostředí lze vygenerovat sérii panoramatických fotografií, díky kterým je možné scénu prohlížet ve virtuální realitě.
- **Skriptovací rozhraní v Pythonu** - díky tomuto rozhraní lze spravovat celé pasáže opakujících se příkazů ze skriptu.
- **Podpora** - CityEngine je dostupný pro platformu Windows, Mac OS X a Linux.

3.3 Analýza dat z okolí Týnce

Existuje řada nástrojů pro prohlížení a editací `.osm` dat. Jedním z doporučených komunitou `osm` je i open source program `JOSM`[36]. Jedná se o desktopovou aplikaci napsanou v jazyce Java umožňující prohlížení a úpravu `osm` dat. Kromě načítání `osm` dat (ať už ze souboru či stažením online), v ní lze načítat například GPX záznamy, což jsou GPS data zaznamenaná v XML formátu.

3.3.1 JOSM

Náhled programu je vidět na obrázku 3.9.

Obrázek 3.9: Týnec nad Sázavou v programu JOSM[36]

Jedna z informací, která se z programu `JOSM` dá vyčíst je i zdroj dat. Následuje seznam opakujících se zdrojů a registrů dat z okolí Týnce:

Územně identifikační registr základních sídelních jednotek[22]

- OpenStreet Map tag: `source=csu:uir-zsj`
- registr UIR-ZSJ je součástí Registru sčítacích obvodů a budov, spravovaný Českým statistickým úřadem. Tento registr již s projektem `osm` od roku 2014 nespolupracuje[68]. Data jsou i přesto v mapách stále obsažena.
- V mapě Týnce se tento zdroj vyskytuje u názvů obcí a jejich částí.

Ústav pro hospodářskou úpravu lesů[78]

- OpenStreet Map tag: `source=uhul:wms`
- WMS je zkratka pro Web Map Service. Jedná se standard vyvinutý skupinou Open Geospatial Consortium, což je mezinárodní organizace vytvářející standardy mimo jiné pro geoprostorová data a služby nebo GIS. Služba umožňuje k rastrovým obrázkům (mapám) přidávat geografické informace. Používá se např. v softwaru GIS.
- Na mapě Týnce se jedná o lesy a ojediněle i o silnice.
- OpenStreet Map tag: `source=uhul:ortofoto`
- Tato vrstva se od roku 2015 nepoužívá. Jednalo se o panchromatickou leteckou analogově snímanou, později digitalizovanou ortofotomapu.
- Vzácně se dá tento tag najít u některých ulic (např. ulice Sadová).

Český úřad zeměměřický a katastrální[23]

- OpenStreet Map tag: `source=cuzk:km`
- Zkratka km zde znamená katastrální mapa. Ta je součástí údajů z katastru nemovitostí obsahující informace o parcelách a budovách.
- Zdroj odkazuje na ulice (např. ulice Příčná), parky a budovy.
- OpenStreet Map tag: `source=cuzk:ruian`
- RÚIAN je zkratka pro registr územní identifikace, adres a nemovitostí. Převážná většina budov v Týnci je převzata z tohoto registru. RÚIAN je jedním ze čtyř základních registrů ČR (ostatními třemi jsou registr osob, registr obyvatel a registr práv a povinností[70]). Jedná se o veřejný seznam obsahující mj. i adresy.
- Většina budov a územní hranice.

Veřejný registr půdy[41]

- OpenStreet Map tag: `source=lpis`
- Jedná se o registr zemědělské půdy, jehož původním účelem bylo ověřování údajů v žádostech o dotace. Nyní se používá jako podklad pro vedení evidencí o použití hnojiv a pastvy, nebo pomáhá monitorovat oblasti výskytu škodlivých organismů.
- V okolí Týnce je použit pro louky a lesy.

Digitální báze vodohospodářských dat[27]

- OpenStreet Map tag: `source=dibavod`
- Tato databáze je primárně určená pro tvorbu tématických kartografických výstupů s vodohospodářskou tématikou a tématikou ochrany vod.
- Například Janovický potok, Chářovický nebo Podhájský potok. Dále i menší nepojmenované vodní toky.

Další a podrobnější seznam použitých zdrojů a dat pro Českou republiku, včetně informací o licencích, rozsahu a kvality dat lze nalézt na stránkách Projektu Česko[68]. Ten se zabývá informacemi vztahujícími se k mapování Čech, Moravy a Slezska. Dále poskytuje návody jak zapisovat `osm` data pro Českou republiku.

Kapitola 4

Existující řešení

V této kapitole jsou vypsány existující simulátory zabývající se realistickým zobrazením. Na internetu lze dohledat více projektů zabývajících se realistickým zobrazením silnic pro testování autonomních vozidel jakožto i simulací. Jsou jimi například projekty CARLA[14], AirSim[8], Apollo[10] či UE4Sim[77].

4.1 Rozhraní pro simulátor

Simulátory poskytují levnější způsob testování a ověřování rozhodovacích algoritmů. Také jsou bezpečnější a vhodnější pro ověřování chování vozidla při nebezpečných situacích, jakými jsou například simulace nárazů nebo typicky děti běžící přes silnici. V reálném světě je testování finančně i materiálně náročnější a nebezpečné situace se špatně napodobují.

4.1.1 CARLA

Jedním z příkladů simulátoru je projekt CARLA. Jedná se o open source simulátor pro výzkum autonomního řízení. Byl vytvořen s cílem tyto systémy podporovat a dále rozvíjet. CARLA poskytuje velké množství objektů (budovy, dopravní prostředky, chodce, města...), které mohou být dále využity[1].

CARLA (Car Learning to Act) oproti závodnímu simulátoru TORCS[76] (The Open Racing Car Simulator, umožňující hráči účastnit se automobilového závodu), poskytuje komplexní a interaktivní prvky, jakými jsou například ulice i s chodci, křižovatky se světly, značky a další. Veškerý obsah je možné dále upravovat a používat. Lze nastavovat a měnit počasí nebo čas ve scéně, případně přidávat další prvky či kolize.

Projekt se nachází na Githubu[16] a po spuštění simulátoru v UnrealEnginu má uživatel na výběr ze dvou scén. Každá z nich obsahuje detailně propracované město, nechybí rozbitá silnice, projíždějící auta, funkční semafory, veřejné osvětlení nebo chodci. Lze potkat chodce s taškou, kytarou nebo i s kufrem. Projíždějící auta jsou různých značek a nechybí ani policejní auto. Ve scéně se dá klávesou `C` měnit počasí a vybrat si ze 14 druhů. Mění se jak denní doba (odpoledne, večer), tak oblačnost (s mraky nebo bez) a síla deště. Ukázky ze scény `Town01` jsou na obrázku 4.1.

(a) Ukázka s počasím

(b) Pohled na scénu

(c) Ukázka chodců

Obrázek 4.1: Ukázky ze simulace CARLA: Odpoledne po dešti (nahoře vlevo), pohled na scénu z UnrealEnginu (nahoře vpravo), příklad vygenerovaných chodců (dole)

Přes komplexnost a propracovanost scény program ve verzi `0.8.4` (aktuální v první části vypracování diplomové práce) umožňovala pouze omezené úpravy mapy[15]. Autoři doporučovali duplikovat existující scénu a dále upravovat skript generující silnice. Tou dobou CARLA podporovala pouze rovné silnice a pravoúhlé zatáčky. V dlouhodobém horizontu plánovali přidat realističtější silnice[33][34].

V druhé polovině listopadu uvedli novou verzi, a to verzi `0.9.1`, která umožnila uživatelům vytvářet si vlastní silnice a poloautomatické generování cest. Aktuálně využívají import silnic za pomoci nástroje RoadRunner[71] od společnosti VectorZero, se kterou blízce spolupracují. Na stránkách tohoto generátoru map lze najít informaci o tom, že se chystají přidat podporu importu GIS dat, neboli například OpenStreet map. Společnost jsem zkontaktovala ohledně dotazu, kdy plánují přidat podporu OpenStreet map. Odepsali, že podporu OSM pro vizualizaci plánují na únor 2019 a plnou podporu (automatické vytváření silnich, křižovatek a budov) později v roce 2019.

Vzhledem k tomu, že plně uživatelské generování silnic ještě není funkční, ale pracuje se na něm, rozhodla jsem se CARLU nevyužít, jelikož řešení, ke kterému bych došla, by s největší pravděpodobností bylo do roka zastaralé a nedalo by se dále používat.

4.1.2 AirSim

AirSim je simulátor pro drony a auta, určený pro `Unreal Engine`. Opět se jedná o open source plugin a lze ho nainstalovat na platformu Windows a Linux. Cílem vývojářů je vytvořit platformu pro výzkum umělé inteligence, počítačového vidění a pro učící se algoritmy pro autonomní vozidla. Pro tyto účely slouží jejich API[6].

Obrázek 4.2: Ukázka z AirSim: dron s módem počítačového vidění.

Ve výchozím nastavení uživatel po spuštění obsluhuje dron. Pro ovládání lze použít počítačovou klávesnici, nebo jiný ovladač. Auto se dá řídit volantem, nebo opět klávesami. Ze scény lze získat statistické údaje z jízdy nebo letu, případně pro sběr trénovacích dat spustit funkci nahrávání. Od té chvíle se uloží sekvence snímků, dokud není nahrávání ukončeno.

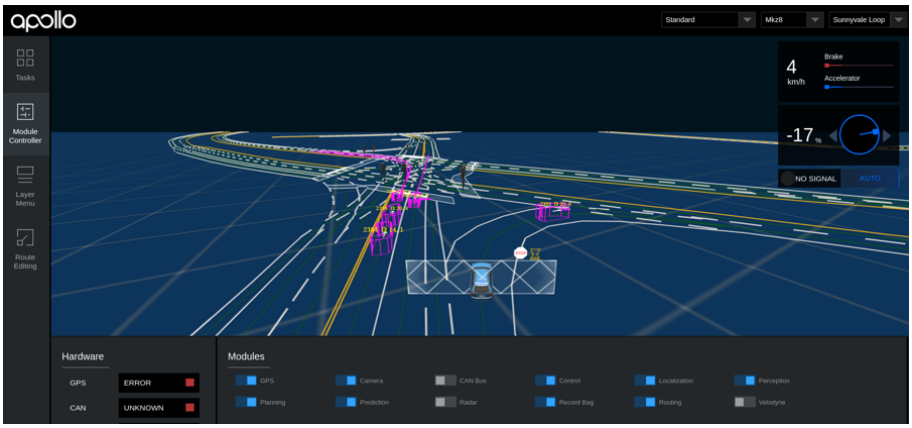
AirSim obsahuje i mód počítačového vidění. Jak je vidět na obrázku 4.2, v tomto módu lze ze scény dostat data jako hloubka nebo segmentace obrazu.

Oproti programu CARLA nemá AirSim propracované počasí nebo pohybující se předměty, na druhou stranu do něj lze vkládat vlastní scény a tudíž byl původně použit pro testování výsledných vygenerovaných dat.

4.1.3 Apollo

Další příklad projektu zabývajícího se testováním autonomních vozidel je projekt Apollo. Oproti projektu CARLA je k dispozici pouze na žádost. Poskytuje ale možnost otestovat si prostřednictvím softwaru navigační algoritmy. Obsahuje sadu scénářů od přímé jízdy, přes předjíždění přes pruhy až po rozhodování se na křižovatce.

Simulace obsahuje možnost projet se virtuálně po milionu kilometrů spolu s okolní dopravou a dalšími autonomními vozidly. Díky scénářům lze algoritmy rychle testovat, ověřovat a optimalizovat. Uživatel si může upravit silnice, přidávat překážky či využívat značky a semafory. Software zatím dokáže testovat detekci kolizí a objektů mimo jízdní pruhy, reakci na dopravní signály, rychlostní limity a natrasování cesty. Vše je podpořeno vizualizací a informačním panelem o vozidle (viz obrázek 4.3).



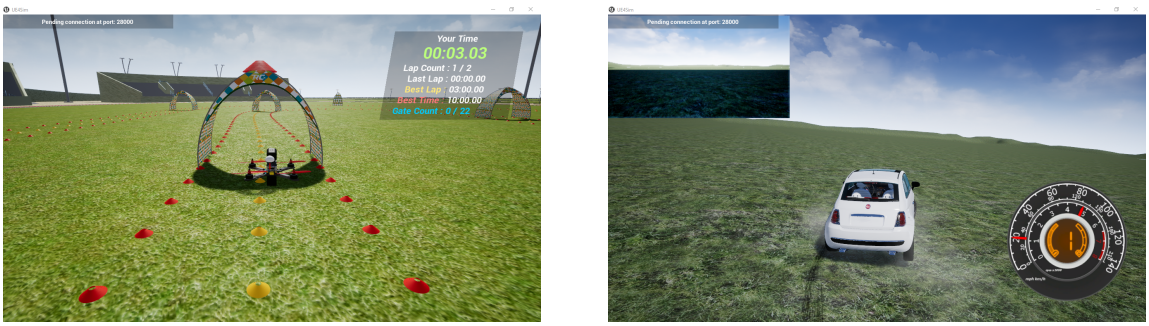
Obrázek 4.3: Ukázka scény autem a výpočtem kolizí

Projekt Apollo se ale zabývá hlavně testováním připravených algoritmů, nikoli zobrazováním realistické scény. Pro tuto práci je proto neužitečný.

4.1.4 UE4Sim

UE4Sim je fotorealistický simulátor pro použití v počítačovém vidění. Je vyvíjen v `Unreal Enginu` a dokáže simulovat jízdu v autě, provoz bezpilotních letadel, obsahuje realistické 3D prostředí s pohybujícími se lidmi[5].

Stažení simulátoru je podmíněno zadáním jména, emailové adresy a krátkým popisem, k čemu uživatel program použije. Bohužel se nejedná o kompletní realistický simulátor, ale pouze o jakousi závodní hru, kde si uživatel může vybrat ze dvou scén, viz obrázek 4.4. V první scéně řídí dron, ve druhé auto. Je možné si vybrat ze dvou map a čtyř typů vozidla. Po spuštění závodu se ale zobrazila pouze scéna s trávou, bez silnic a realistických stromů. Scéna s dronem obsahuje stadion a checkpointy, bohužel je ale celý simulátor, ač podle článku vypadal velmi zajímavě, pro tento projekt zcela nepoužitelný.



(a) UE4Sim s dronem

(b) UE4Sim s autem

Obrázek 4.4: Ukázka scény ze simulátoru UE4Sim s dronem (vlevo) a s autem (vpravo)

Kapitola 5

Realizace

Realizovány byly dva přístupy. První využívá Open source knihovny, druhý využívá široké možnosti komerčního nástroje CityEngine. Obě varianty jsou v následující části popsány a jsou zohledněny jejich výhody a nevýhody. V obou případech pro mapové podklady byla použita Mapbox data a pro vektorovou vrstvu `osm` data.

5.1 Open source varianta

První přístup využívá pouze open source programy a knihovny. K jeho funkčnosti není potřeba žádný komerční software. Následuje rozbor použitých komponent, postup realizace a zhodnocení přístupu.

5.1.1 Komponenty

V open source variantě se využívají výše popsané programy, kterými jsou `osm2world`, `osmosis` a `osmfilter` umožňující filtrovat data podle tagu. Pro přípravu dat terénu byl použit program `Meshlab`, což je systém pro zpracování a úpravu 3D triangulární sítě. Poskytuje řadu nástrojů pro editaci, čištění, opravování, texturování a vykreslování meshe.

5.1.1.1 Meshlab

Pro převod dat z mapových podkladů ČÚZM byl použit program Meshlab[52], přesněji jeho konzolová verze. Uživatel si v programu vytvoří vlastní filtr, který dále z příkazové řádky spouští a aplikuje na další data. Pro vytvoření filtru pro převod dat z formátu `x y h`, kde `x` představuje zeměpisnou šířku, `y` zeměpisnou délku a `h` výšku v metrech, na mesh se nejprve dopočítají normály pro vrcholy ze zadaného mračna bodů a následně se pomocí Poissonovy funkce pro rekonstrukci povrchu[4] vytvoří mesh. Výstup je uložen ve formátu `.mlx`, což je zkratka pro MeshLab Scripting File Format. Takto připravený filtr se použije k vytvoření objektu ve formátu `.obj` následujícím příkazem:

```
meshlabserver -i input.xyz -o output.obj -s filter.mlx
```

5.1.1.2 Osmfilter

Aby se data dala zpracovat samostatně (samostatná vrstva budov, silnic, lesů, elektrického vedení, atd.), pro jejich oddělení byl použit program `osmfilter`[59], který dokáže `osm` data podle zadaných parametrů exportovat na menší `osm` soubory obsahující jen zvolené vrstvy. Takto lze ošetřit, aby se při načítání dat pracovalo například pouze s lesy, silnicemi či domy. Program jako první parametr dostane data, ze kterých bude filtrovat, následují vyhledávací pravidla (lze použít `keep` pro vybrání uzlů, cest a vztahů, `drop` pro vynechání dat, `keep-tag` pro vybrání daného tagu). Pro filtrování silnic první a druhé třídy lze použít následující příkaz:

```
./osmfilter tynec.osm --keep="highway=primary =secondary" >streets.osm
```

5.1.1.3 OSM2World

Jak již bylo popsáno v kapitole 3, program `OSM2World` umí převést data z formátu `.osm` do formátu `.obj`. Lze mu jako parametr dodat vlastní konfigurační soubor, ve kterém lze specifikovat hustotu zalesnění povrchu se stromy, minimální a maximální výšku stromů, textury či barvy povrchu a budov. Program funguje i bez konfiguračního souboru s předdefinovanými hodnotami.

Program ve výchozím nastavení umožňuje exportovat soubor jako jeden zapečený model, nebo lze v konfiguračním souboru přidat hodnotu maximálního počtu vytvořených objektů. Bohužel se ale již nedá ošetřit, které vrstvy se vygenerují spolu. Například pokud je ve scéně 7 domů a tři zalesněné oblasti, nevygeneruje se 10 samostatných objektů, ale vygeneruje se jich méně. Například jeden objekt se dvěma zalesněnými oblastmi, nebo trojice domů. A to i přes to, že je parametr maximálního počtu objektů nastaven na vyšší hodnotu, než jaký je počet objektů ve scéně. Pro potřeby této práce by bylo potřeba mít samostatně všech 10 objektů, v ideálním případě by se v objektu zalesněné oblasti dalo hýbat i s jednotlivými stromy. Bohužel tuto schopnost `OSM2World` nemá.

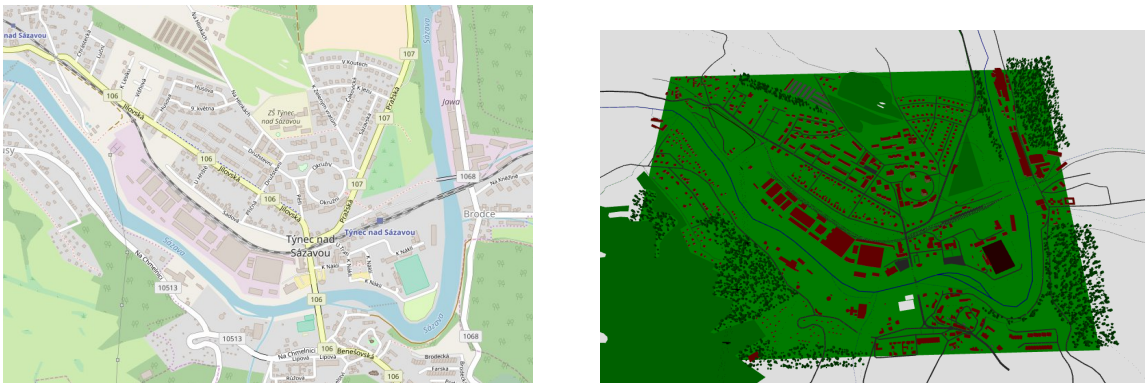
Dále se lze dočíst[56], že program podporuje výšková data ve formátu `SRTM`. Autor programu považuje funkci terénu za experimentální, což by mohl být důvod, proč se výšková data do scény nepodařila přidat. Stabilní verze `OSM2World` je 0.1.9 vydaná 11.1.2012. Od té doby projekt funguje na githubu, kde poslední úprava proběhla 2. 10. 2018. Není proto vyloučeno, že `SRTM` data budou v budoucnu funkční.

Následuje příklad převodu s použitím konfiguračního souboru:

```
./OSM2World.jar -i input.osm -o output.obj --config conf.properties
```

5.1.2 Postup

První přístup byl postaven na Open source programech a OpenStreet mapách, jelikož jsou velmi rozšířené, převážná část jiných mapových programů je využívá ke své potřebě a s velkou pravděpodobně nehrozí náhlé ukončení jejich podpory. Z toho plyne i zvolený formát vstupních dat, kterým je `.osm`. S tímto formátem pracují jak program `OSM2World`, tak z `OpenStreet map` komunity doporučovaný a dokumentovaný `Osmosis`.

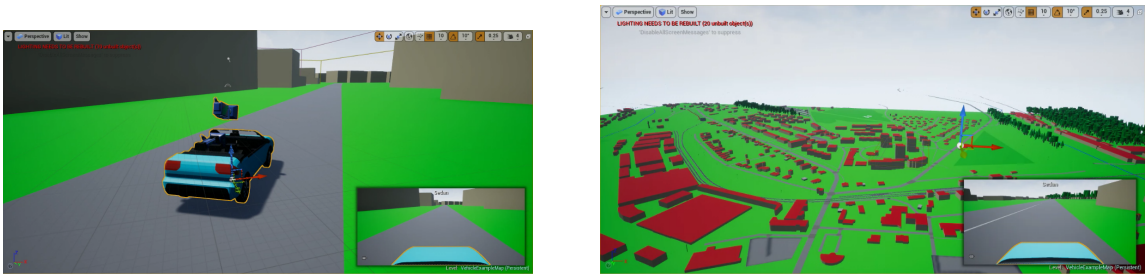


(a) openstreetmap.org (b) 3dviewer.net

Obrázek 5.1: Menší výřez mapy ze serveru openstreetmap.org (vlevo), vizualizace dat v online nástroji (vpravo)

Za pomoci aplikace `OSM2World`[56] byl výřez mapy převeden na soubor typu `.obj`. Formát `.obj` byl zvolen z toho důvodu, že `Unreal Engine`, ve kterém se tato práce bude vyvíjet, od verze 4.2 (aktuální verze je 4.21) umí modely tohoto typu importovat. Bylo třeba importovat model po částech, jelikož velká data (okolo $20km^2$) se do enginu naráz vložit nedají.

Menší výřez (přibližně $5km^2$) se dá vizualizovat online nástrojem `3D viewer`[7] a již se dá i vložit do herního enginu. Menší exportovaná data jsou na obrázku 5.1.



(a) Ukázka z blízka (b) Ukázka z dálky

Obrázek 5.2: Ukázka z realizace

Pro testování převedeného objektu byla vytvořena scéna v Unreal Enginu, ve které se nacházelo enginem předdefinované řiditelné auto. Po importu objektů bylo potřeba upravit velikost importovaného objektu na hodnotu 100. `Unreal Engine` používá ve výchozím nastavení metriku s centimetry, importovaný objekt ale používá metry. Výchozí nastavení lze změnit, v tomto případě stačí změnit hodnotu při importu.

Model je tvořen několika samostatnými vrstvami - domy, lavičky, parkovací místa, stanice vysokého napětí, silnice a cesty, stromy a další. Tyto vrstvy ale dále dělitelné nejsou. Při konfiguraci s více `.obj` daty je objektů z vrstev více, stále jsou ale sdružené (místo zapečené vrstvy 20 laviček, se vygenerují čtyři vrstvy po 5 lavičkách). Základní verze bez úprav barvy terénu a budov je vidět na obrázku 5.2.

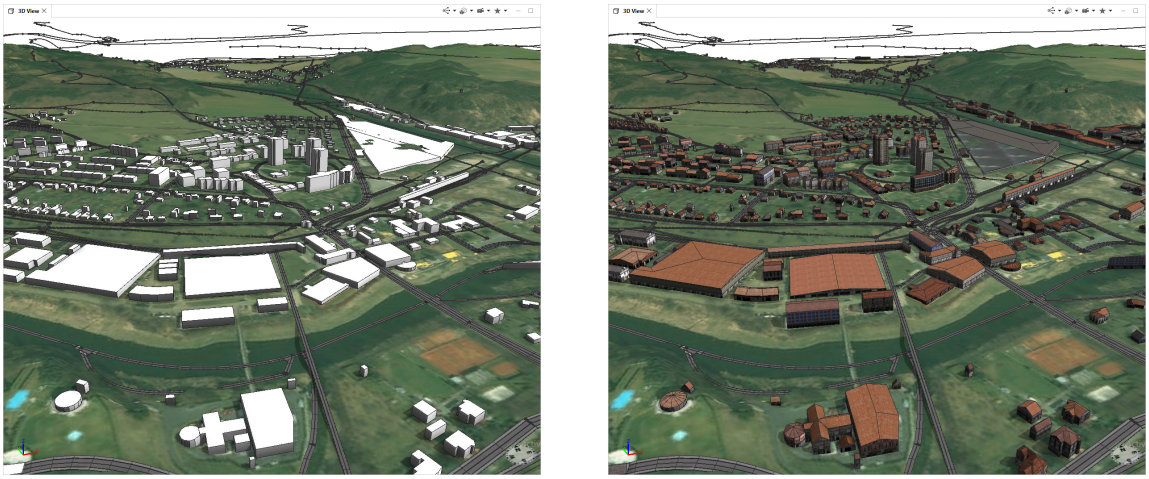
5.1.3 Zhodnocení

Výhoda tohoto přístupu je dostupnost programů, které jsou volně stažitelné a možnost ovládání kroků z příkazové řádky. `OSM2World` je tvořen moduly a hodnoty lze upravovat pomocí konfiguračního souboru. Také v případě rozšíření programu `OSM2World` by práce nad jednotlivými částmi byla lehce oddělitelná. Struktura programu totiž modulový přístup podporuje.

Nevýhodou je nerealističnost a velká jednoduchost výsledného modelu, a z toho plynoucí časová náročnost následných úprav. Pro úpravu budov a silnic by bylo zapotřebí potřebné moduly ve zdrojovém kódu `OSM2World` upravit. Moduly se v případě stažení zdrojového kódu vyskytují ve složce `osm2world-src/src/org/osm2world/core/world/modules`. Každý modul spravuje jeden z 19 importovatelných objektů (most, budova, bazén, ulice, stromy, voda a další). Další nevýhodou je nemožnost práce s terénem. Terén se dá do scény vložit jako mesh, ale poté by bylo potřeba každý objekt posunout nahoru podle výšky terénu v daném bodě. Vzhledem k tomu, že data nenesou geografickou informaci, bylo by posouvání časově náročnější.

5.2 CityEngine varianta

Druhý přístup využívá širokých možností programu CityEngine. Jeho 30 denní verze se dá zdarma stáhnout na stránkách CityEnginu[17]. Tento nástroj i v základním rozhraní umožňuje vytvářet oproti `OSM2World` realistické scény. Na obrázku 5.3 je vidět základní import dat ve formátu `.osm` a vygenerování silnic a budov pomocí realistické reprezentace s texturovanou fasádou bez žádných dalších úprav.



(a) Import osm dat (b) Vygenerované domy

Obrázek 5.3: Použití pravidla pro realistické fasády budov

Od verze 2016 přibyla možnost zadat výřez z mapy přímo v CityEnginu pomocí nástroje `Get Map Data`. Program sám stáhne z internetu po registraci výšková data, vektorová data z `osm` a lze si zvolit i texturu terénu. Nevýhoda tohoto přístupu je pro tuto práci v nutnosti

uživatele stáhnout si všechna data najednou a následně se všemi najednou i pracovat. Problém může nastat u velkých map, kde export dat již trvá velmi dlouho a ne vždy se podaří. Z toho důvodu byl zvolen přístup importu předzpracovaných výškových a vektorových dat do CityEnginu.

Terén v CityEnginu se vytváří pomocí obrazových dat, neboli výškovou mapou. Tyto referenční hodnoty slouží i k operacím umožňujícím zarovnat s terénem silniční síť a objekty. Pro vytvoření terénu se importuje výšková mapa a textura. Pokud obrazová data obsahují geografická metadata, předvyplní se minimální a maximální hodnota výškové mapy spolu s rozměrem a případným offsetem, který určuje lokaci terénu v metrech. Pokud data geografické údaje neobsahují, nastaví se hodnoty na výchozí, které ale neodpovídají skutečnosti. Pro účely této práce bylo potřeba konvertovat data do formátu, který metadata obsahuje, aby se daly načítat jednotlivé dlaždice a při exportu na sebe navazovaly. Takovým formátem je např. `GeoTIFF`.

5.2.1 Komponenty

V této sekci se nachází popis programů a knihoven, které se pro variantu se CityEnginem používají.

5.2.1.1 Georeferenční údaje

`GeoTIFF` je formát používaný pro aplikace využívající geografický informační systém (GIS). Ten umožňuje ukládat, spravovat a analyzovat prostorová data. Jedná se o soubor typu `.tiff` obsahující navíc referenci geografických dat. Příkladem je informace o projekci, souřadnicovém systému či datumu. Datem se rozumí parametr nebo soubor parametrů, který vymezuje polohu počátku, měřítko a orientaci souřadnicového systému[30]. Pro úpravu souborů bez georeferenčních údajů lze použít Open source knihovnu `GDAL`. Tato knihovna umí načítat a zapisovat geoprostorový rastrový a vektorový formát. Obsahuje množství služeb pro zpracování či úpravu dat pomocí příkazové řádky. Projekce a transformace jsou podporovány i knihovnou `PROJ.4`[67], která umožňuje konverzi mezi kartografickými projekcemi. Knihovnu `GDAL` využívá např. Google Earth, Grass GIS či ArcGIS.

V této práci byla pro konverzi dat a přidání georeferenčních údajů použita knihovna `GDAL`. Výchozím souborem bez geografických dat byly stažené dlaždice ve formátu `.png` ze služby Mapbox.

5.2.1.2 GDAL

GDAL[31] - Geospatial Data Abstraction Library je open source knihovna pro čtení a zápis rastrových a vektorových dat. Knihovna využívá jednoduchý abstraktní datový model pro všechny podporované datové formáty. Nabízí také nástroje pro konverzi a zpracování dat dostupné i z příkazové řádky.

Knihovna GDAL se v této diplomové práci stará o většinu převodních operacích a vytváření výsledné výškové mapy v `tif` souboru.

5.2.1.3 Rasterio

Rasterio je knihovna umožňující číst a zapisovat rastrová prostorová data. Rasterio podporuje Python od verze 2.7, je přístupná pro Linux, OS X i Windows. Projekt se nachází na githubu[69], kde je možné stáhnout binární soubory.

V práci se knihovna používá pro načtení hodnot barevných kanálu `png` dlaždice, ze kterých je následně dopočítaná výška terénu.

5.2.1.4 Gdal2tiles

Tato knihovna umí převádět souřadnicové systémy mezi sebou, z čísla dlaždice dopočítá zeměpisnou šířku a výšku, a naopak. Tato knihovna výrazně snížila složitost napsaných skriptů, jelikož obsahuje všechny potřebné metody pro konverzi dat[9].

5.2.2 Postup

Druhý přístup nejprve stáhne data ze zadané oblasti, zpracuje je a přidá georeferenční údaje. Úprava dat ve scéně probíhá v prostředí CityEngine, odkud se vygenerují modely a následně se načtou do herního enginu `UnrealEngine`.

Pro co nejvíce automatizovanou práci s následujícími skripty je potřeba dodržovat tuto souborovou strukturu:

```
+---DIP_data
    +---CityEngine
        +---Workspace
            +---Tynec
    +---osm-raw
    +---osm-tiled
    +---scripts
    \---terrain
        +---csv
        +---czmu
        +---mapbox-tiles
        +---satellite
        +---tiff
```

Všechny spouštěcí soubory jsou spouštěny z kořenové složky `DIP_data`.

Pro spuštění celé konverze stačí v kořenové složce `DIP_data` spustit následující skript:

```
./scripts/run.bat
```

Veškeré konstanty a cesty k souborům, která jsou pro práci vyžadována jsou přehledně v konfiguračním souboru `config.ini`.

5.2.2.1 Funkce jednotlivých skriptů

V následující části jsou popsány skripty, které se používají pro automatické stažení dat a jejich zpracování pro následný import do CityEnginu.

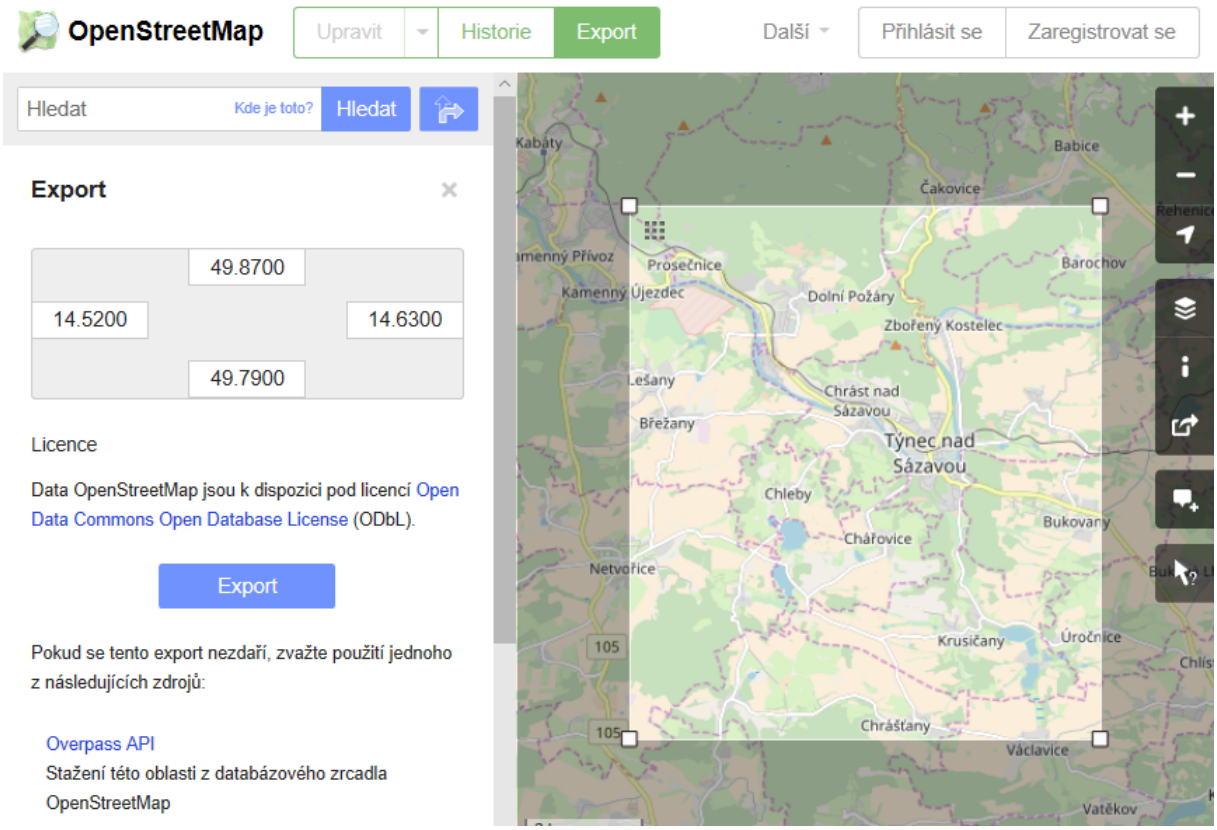
Automatizovaný skript

Jmenuje se `run.bat` a pokud se ve složce `osm-raw` vyskytuje soubor `map.osm`, automaticky stáhne jednotlivé dlaždice z mapboxu a k nim potřebné `osm` dlaždice. Jelikož se z mapboxu data stáhnou ve formátu `.png`, následuje konverze na výškovou mapu pomocí rovnice 3.1 do formátu `.tif` obsahující díky knihovně `GDAL` i geografická metadata. Díky nim se při importu dat do CityEnginu předvyplní souřadnice. Ke správné funkčnosti tohoto skriptu je zapotřebí mít nainstalovaný `cURL` a mít nastavené proměnné systému, aby se program dal spustit odkudkoli.

Stažení mapbox dat

Uživatel může výřez z mapy udělat dvěma způsoby:

- První způsob je export dat ze stránky `https://www.openstreetmap.org/export`, kde si lze vybrat část mapy, kterou chce uživatel stáhnout. Pokud `osm` soubor pojmenuje `map.osm` a uloží do složky `osm-raw`, po spuštění batch skriptu `run.bat` se data dále zpracují a na výstupu budou geograficky referencovaná výšková data. Tento postup je pro plnou automatizaci vyžadovaný. Ukázka exportu je na obrázku 5.4.



Obrázek 5.4: Příklad výřezu z mapy - stažení dat se provede kliknutím na tlačítko Export

Pokud se soubor bude jmenovat jinak nebo bude v jiné složce, je třeba skriptům přidat cestu k souboru jako parametr. Již nelze spustit skript `run.bat`, ale je potřeba pro stažení mapbox dat zavolat následující skripty:

```
python ./script/get_lat_lon.py [CESTA_K_OSM_SOUBORU]
./script/download-mapbox-tiles.bat
```

Skript `get_lat_lon` bez argumentu hledá ve složce `osm-raw` soubor s názvem `map.osm`, ze kterého získá informace o hodnotách zeměpisných šířek a délek dat. Z těch dopočítá další potřebné hodnoty, jakými jsou hodnoty `x` a `y`, které používá mapbox pro definici pozice dlaždice. Potřebná data uloží do textového souboru. S argumentem se jen změní cesta k souboru, ze kterého data získává.

Skript `download-mapbox-tiles` vezme převedené hodnoty z výše vygenerovaného souboru a stáhne dlaždice z potřebného rozsahu.

- Druhý způsob vytvoření výřezu mapy je zadáním souřadnic bounding boxu v pořadí minimální zeměpisná výška, minimální zeměpisná délka, maximální zeměpisná výška, maximální zeměpisná délka v desetinném formátu s tečkou. Jinak formulováno se zadá dolní okraj bounding boxu, následně levý, horní a nakonec pravý okraj. V tomto případě opět nefunguje automatický skript `run.bat`. Oproti předchozímu příkladu se změní název prvního volaného skriptu:

  ```
  python ./script/get_lat_lon_boundaries.py [minLat] [minLon] [maxLat] [maxLon]
  ./script/download-mapbox-tiles.bat [./terrain/osm_latlon_data.txt]
  ```

  V případě stejného výřezu jako na obrázku 5.4 by příkaz vypadal takto:

  ```
  python ./script/get_lat_lon_boundaries.py 49.79 14.52 49.87 14.63
  ```

  Skript `get_lat_lon_boundaries` je téměř totožný se skriptem `get_lat_lon`, pouze nenačítá data ze souboru, ale používá hodnoty, které jsou v argumentech. Na pořadí hodnot záleží.

Stažení osm dlaždic

Pro velké plochy mohou být i `.osm` data extrémně veliká, a proto se i z těchto dat vytvoří dlaždice. Tentokrát pomocí skriptu

```
python ./script/download-osm-tiles
```

Počet dlaždic `osm` je totožný s počtem dlaždic terénu. Takto vygenerované dlaždice je potřeba upravit filtrem, aby se neopakovaly stejné hodnoty. `Osm` struktura totiž sestává ze vztahů mezi uzly a tudíž se stahují i části přesahující zadaný výřez. Pokud třeba do dvou dlaždic bude zasahovat řeka, objeví se v obou slaždicích. `Osm` data se neořezávají, jelikož by se ořezem ztratily relační informace. Bez použití filtru by se celé úseky opakovaly a zbytečně by zvětšovaly výsledný model.

5.2.2.2 GDAL konverze

Pro automatické vyplnění údajů v CityEnginu do dat při importu musí být dodány i georeferenční informace. Aby se dala využít i data jiného formátu (např. `.xyz` jako data ze Zeměměřického ústavu) je krok konverze na `.tif` rozdělen na dva kroky:

- konverze z `png` do `csv`
- konverze z `csv` do `tif`

Pouze modifikací přípony lze místo z formátu `csv` exportovat data z formátu `.xyz`. V prvním skriptu se z rastrového souboru dopočítá skutečná výška terénu, která je následně uložena v tabulkovém formátu v pořadí `x`, `y`, `výška`. Hodnoty `x` a `y` jsou posunuty v každé další dlaždici o offset velikosti celé dlaždice umožňující správné mapování výsledných dat.

Druhý skript využívá knihovnu `GDAL` a k datům přidává i georeferenční informace pomocí virtuálního formátu `.vrt`. Tento formát se vygeneruje pro každou dlaždici zvlášť, jelikož musí obsahovat cestu ke zpracovávanému souboru a jeho název. Příklad použitého virtuálního formátu:

```
<OGRVRTDataSource>
   <OGRVRTLayer name=[NAME_OF_FILE]>
      <SrcDataSource>[PATH_TO_FILE].csv</SrcDataSource>
      <GeometryType>wkbPoint</GeometryType>
      <LayerSRS>EPSG:32633</LayerSRS>
      <GeometryField encoding="PointFromColumns" x="field_1" y="field_2" z="field_3"/>
   </OGRVRTLayer>
</OGRVRTDataSource>
```

`SrcDataSource` je jediný povinný argument ve vrstvě `OGRVRTLayer`, jehož hodnota je cesta a název datového zdroje.

`GeometryType` je nepovinný argument udávající typ geometrie, který byl ve vrstvě použit. Hodnota může nabývat např. "wkbNone", "wkbUnknown", "wkbPoint", "wkbLineString", "wkbPolygon", "wkbMultiPoint", "wkbMultiLineString", "wkbMultiPolygon", "wkbGeometryCollection"a další. Zkratka `wkb` představuje binární verzi `WKT` (Well-known text), což je značkovací jazyk pro reprezentaci vektorové geometrie objektů na mapě a transformace mezi referenčními systémy.

`LayerSRS` je také nepovinný argument, jehož hodnota je např. ve formátu `WKT`, `PROJ.4` či `XML`. Hodnota `EPSG:32633` představuje souřadnicový systém WGS84 a obsahuje téměř celou Českou Republiku.

`GeometryField` definuje, v jakém pořadí se data vyskytují v daném `.csv` souboru. Výchozí řazení je podle hodnot `y` a dále podle `x` vzestupně.

5.2.2.3 Import připravených dat

Exportovaná a připravená data jsou výše uvedenými skripty uložena do složky `./terrain/tiff/`. Počet souborů záleží na velikosti původního výřezu. Dalším krokem je import dat do prostředí CityEngine. Díky možnosti skriptování v jazyce Python je ve výchozím nastavení potřeba spustit skript s názvem `createMap.py`, který se nachází ve složce `./CityEngine/Workspace/Tynec/scripts`. Skript stáhne připravená data terénu.

Teprve po importu jednotlivých dlaždic se ukázalo, že na sebe dlaždice úplně nenavazují. Samozřejmostí jsou okrajové hodnoty dlaždic, které se musí interpolovat z těch okolních. Bohužel na sebe ale dlaždice nepasovaly ani v osách `x` a `y`. Navíc byly dlaždice georeferencované na protější polokouli. I po důkladném zkoumání knihovny GDAL se nepodařilo zjistit, proč je hodnota zeměpisné šířky záporná, ač na severní polokouli by hodnota měla být kladná.

Vzhledem k dlouhodobé nemožnosti tento problém vyřešit jsem se nakonec rozhodla výše uvedený převodní řetězec nechat v původním stavu pro případnou úpravu a do výsledné aplikace využila importovaný nedlaždicový terén ze CityEnginu. Po případném vyřešení problému je skript v práci přiložený.

5.2.2.4 Řešení po vrstvách

CityEngine zatím neumožňuje plně automatizovaný import vrstev z `osm` dat. Import umožňuje vybrat všechny prvky pomocí zaškrtávacího tlačítka `import all`. Díky tomuto přístupu by uživatelská interakce zůstala na minimu, na druhou stranu pokud by se s danou vrstvou nepočítalo při přiřazování CGA pravidel, daná vrstva by se automaticky reprezentovala jako silnice (pokud by se jednalo o graf - `GraphSegment`), nebo jako budova (pokud by se jednalo o objekt - `Shape`). Z tohoto důvodu byl využit přístup postupného importování po vrstvách. CityEngine při importu vždy nechá vybranou vrstvu silnic (highways). Uživatel proto musí vybrat vrstvu, kterou chce importovat, a nesmí zapomenout na odškrtnutí vrstvy silnic.

Jednotlivé vrstvy jsou ovládány pomocí připraveného Python skriptu. Přiřadí se CGA pravidlo definované pro danou vrstvu, které obstará vygenerování vzhledu. Skripty jednotlivých vrstev dědí od hlavního skriptu, který obsahuje společné metody jakými jsou například nastavení startovního pravidla, CGA pravidla, generování modelu, změna názvu vrstvy, definice proměnných.

Při ukázce stavu aplikace byl změněn přístup ke generování domů. CityEngine obsahuje textury realistických fasád, které se na budovu pouze aplikují a budova je jinak plochá. Vzhledem k potřebě, aby z pohledu 1. osoby vypadaly více realisticky, bylo využito pravidlo na vytvoření komplexnější fasády domů s okny a balkóny. Stále se používají různé textury fasády, tentokrát ale s ohledem na informace z `osm` dat[72]. Využívá se k tomu tag `building:ruian:type`, který nabývá číselných hodnot 1 až 29, jak je uvedeno v číselníku nemovitostí v sekci způsobu využití stavby[21]. Příklad možných hodnot:

- building:ruian:type = 1 průmyslový objekt
- building:ruian:type = 2 zemědělská usedlost
- building:ruian:type = 3 objekt k bydlení

- building:ruian:type = 4 objekt lesního hospodářství
- building:ruian:type = 5 objekt občanské vybavenosti
- building:ruian:type = 6 bytový dům
- building:ruian:type = 7 rodinný dům
- building:ruian:type = 8 stavba pro rodinnou rekreaci
- building:ruian:type = 9 stavba pro shromažďování většího počtu osob

Vzhledem k tomu, že se aplikuje na vrstvu CGA pravidlo obsahující switch s 29 možnými hodnotami, je ve složce `scripts` připraven skript `_createCGA.py`, kterému uživatel zadá rozsah dat a konkrétní hodnoty načitelné z `csv` souboru, a CGA skript se vytvoří automaticky.

Následuje ukázka začátku kódu, kterou musí uživatel upravit, aby se mu CGA pravidla vytvořila podle jeho specifikací:

```
'''mandatory methods to define'''
# name of OSM attribute which will be used by CGA rule
def getAttrName():
    return "highway"
# value of OSM attribute which will be used by CGA rule
def getAttrValue():
    return ""
# range of possible values
# i.e.[i+1 for i in range(getMaxValue())] for values 1,2,3...
#       fromFile("highway", "value") for data from csv file
#       ["string1","string2","string3"] for user defined values
def getPossibleValues(): #range of possible values
    return fromFile("highway", "value")

'''recommended methods to define'''
# define head of a library (information)
def getInfoLib():
    return "CGA Library"
# maximum of possible values - if using csv file, use 'len(getPossibleValues())'
def getMaxValue():
    return len(getPossibleValues())
# name of Start Rule
def getStartRuleName():
    return "Lot"
# name of denerated CGA rule file (without .cga)
def getCGAfileName():
    return "TestCGA"
```

Takto upravený a spuštěný skript vygeneruje následující cga pravidlo (ukázka je zkrácena):

```
/*CGA Library*/
version "2017.1"

import Color_Names: "rules/General/Color_Names.cga"

//variables from object attributes
@Group("From Object Attributes", 1)
attr highway = ""

//default constant
@Group("default constants", 2)
attr HEIGHT = 1

@StartRule
Lot -->
case highway == "motorway":
color("#B83ED1")
extrude(HEIGHT)
case highway == "primary":
color("#FC72FB")
extrude(HEIGHT)
case highway == "secondary":
color("#871B39")
extrude(HEIGHT)
case highway == "tertiary":
color("#EC64C9")
extrude(HEIGHT)
else :
color("#FF0000")
extrude(50*HEIGHT)
```

Výše uvedené pravidlo vezme z importovaných silnic atribut `highway` a podle typu, zda se jedná o `motorway`, `primary`... nastaví náhodnou barvu. Pokud atribut neodpovídá možným hodnotám (ať už z výčtu nebo z csv souboru, tato část se obarví červeně a její výška se nastaví na 50. Tato hodnota byla zvolena jen s ohledem na to, aby uživatel hned viděl, jakému objektu schází definovaná hodnota a jestli se s ní dá v algoritmech počítat. Pomocí tohoto skriptu byla testována i hodnota `height` u budov. Díky tomu se ukázalo, že `osm` data v okolí Týnce často neobsahují ani základní údaj o výšce, a proto se při generování výšky budov musí výška nastavovat u některých budov ručně.

Aktuálně podporované vrstvy se jmenují vždy stejně, jako `OSM` tag. V projektu se pracuje s vrstvami buildings (budovy), highways (silnice), waterways (řeky a jezera), bridges (mosty), landuse (lesy a louky), railways (železnice) a power (elektrické vedení). Import konkrétních `obj` objektů se v `CGA` pravidle používá pomocí funkce `i("CESTA_K_SOUBORU")`.

Porovnání scény ze CityEnginu a Google map je na obrázku 5.5.

(a) Pohled na vygenerované město v CityEnginu (b) Satelitní snímek z Google map

Obrázek 5.5: Porovnání scény ze CityEnginu (vlevo) a Google maps (vpravo)

5.2.3 Implementace

Adresářový a souborový systém používaný v CityEnginu je následující:

```
Tynec
+---assets          # potrebne assety a data k vygenerovani dane vrstvy
|   +---Buildings
|   +---Landuses
|   +---Powers
|   +---Railways
|   \---Streets
+---data            # datove soubory jako csv seznamy ci dlazdice osm map
|   +---csv
|   \---osm
+---maps            # vyskova data s texturou a plnou osm mapou
+---models          # slozka pro ukladani vygenerovanych modelu
+---rules           # pravidla pro jednotlive vrsty
|   +---Bridge
|   +---Building
|   +---General
|   +---Highway
|   +---Landuse
|   +---Power
|   +---Railway
|   \---Waterway
+---scenes          # CityEngine sceny
\---scripts         # Python skripty pro generovani sceny a CGA pravidel
```

5.2.3.1 Hlavní skript

Hlavní skript `main.py` se stará o práci při generování jednotlivých vrstev. Oproti výše uvedenému skriptu `createMap.py` nevyužívá dlaždicová data, ale výškovou mapu a `osm` data ze složky `./CityEngine/Workspace/Tynec/maps`, která byla stažena pomocí funkce `Get Map Data` v CityEnginu.

Při spuštění skriptu se nejprve celá scéna smaže pomocí funkce `deleteAll`. Smazání scény je dobré ve chvíli, kdy chce uživatel vidět, jak se celý řetězec úprav ve výchozím nastavení provede. Následuje funkce `importAll`, která přidá dočasně do systémové cesty adresář s přidanými skripty, importuje terén a následné vrstvy.

Nevýhoda aktuálního načítání vrstev je ta, že uživatel musí vybrat konkrétní vrstvy, které se importují. Vzhledem k tomu, že `osm` data mohou obsahovat různé tagy a atributy, nelze zatím automaticky vybrat chtěné vrstvy. Proto si musí uživatel dát pozor, v jakém pořadí se vrstvy importují, aby nevybral například místo budov silnici. Pak by zákonitě nefungovala CGA pravidla.

Aktuální pořadí importovaných vrstev je následující:

1. building
2. railway
3. highway
4. bridge
5. power
6. waterway
7. landuse

Při importu lze sledovat konzoli, která vždy vypíše název aktuálně zpracovávané vrstvy. Nejprve se importuje Python module přidružený k importované vrstvě, přejmenuje se vrstva ve scéně pro případnou práci s ní, zobrazí se dialog pro vybrání vrstev, vrstva se zarovná s terénem, nastaví se jí CGA pravidla a vygeneruje se model.

5.2.3.2 Přidání vrstvy

Pro přidávání další `osm` vrstvy je potřeba vytvořit ve složce `scripts` skript, který dědí od hlavního skriptu `SuperObject.py`. Ten obsahuje společné metody pro všechny vrstvy, jako import dat, nastavení startovního pravidla, přidání CGA pravidla, vygenerování modelu, vyhledávání či mazání objektů a zarovnání vrstvy k terénu. Lze ho v případě potřeby rozšířit o další metody.

Pokud vrstva neobsahuje speciální požadavky, jako generování konkrétních uzlů z `obj` modelu, vytváří se nová vrstva následovně:

```
from scripting import *

# get a CityEngine instance
ce = CE()

from SuperObject import Layer

class NEW_LAYER_CLASSNAME_HERE(Layer):
    ruleName = "rules/CGA_NAME_HERE.cga"
    startRule = "START_RULE_NAME_HERE"
```

Do hlavního skriptu `main.py` se následně přidá do metody `importAll()` takto vytvořená nová vrstva jako

```
importAndGenerate("SCRIPT_NAME_HERE","NEW_LAYER_CLASSNAME_HERE")
```

.

Pro pozdější možnost s vrstvou pracovat vrací metoda `importAndGenerate()` i celou aktuálně zpracovávanou vrstvu. Lze ji proto uložit do samostatné proměnné. Toho využívají vrstvy waterway a power. Ty se po importu ještě zpracovávají. Proto lze metodu volat odděleně - nejprve import a následně generování a přiřazování `CGA` pravidel.

5.2.3.3 Vrstvy

V následující části se nachází detailní popis jednotlivých použitých vrstev, obrazová ukázka spolu s poměřením výsledku se službou mapy.cz a tipy na doporučené rozšíření.

Buildings

Parametry pro Python skript `buildings.py`:

```
ruleName = "rules/Building/BuildingShell.cga"
startRule = "BuildingMultipatch"
```

Soubor obsahující CGA pravidla pro budovy se nazývá `BuildingShell.cga`. Script pochází z databáze existujících CGA pravidel[18]. Toto pravidlo bylo vybráno po konzultaci s doc. Ing. Jiřím Bittnerem, Ph.D. Původně používané realistické fasády neobsahovaly žádné architektonické prvky. Jednalo se pouze o ploché stěny s texturou, nikoli o trojrozměrné objekty jakými jsou okna, dveře či balkóny.

Do skriptu byl přidán atribut výšky z `osm` dat. Bohužel bylo zjištěno, že mnoho budov neobsahuje informace o výšce budovy. Výška se proto nastavuje z atributů `building:levels` a `building:flats`.

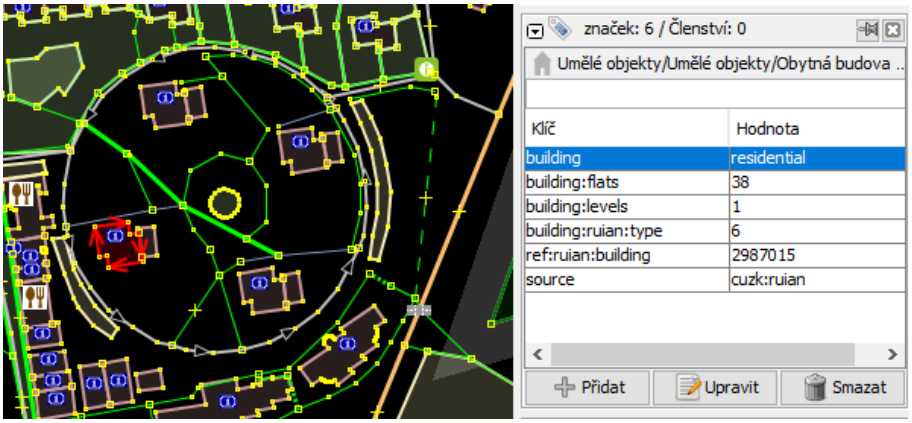
Atribut `building:levels` se používá pro označení počtu nadzemních podlaží. Podzemní podlaží a střecha se nezapočítávají. Doporučuje se kombinovat tento atribut s výškou `height=*` v metrech, ale tato kombinace se v mapách Týnce nevyskytuje.

Atribut `building:flats` obsahuje počet bytových jednotek v budově. Dá se pomocí něho odhadnout velikost budovy.

Vzhledem k tomu, že jsou často tyto dva atributy jedinými nacházejícími se v objektech budov v okolí Týnce a objevují se i špatné hodnoty, pro výpočet výšky bylo využito maximum z těchto dvou atributů vynásobené konstantou 3, reprezentující přibližnou výšku jednoho patra podle systému typů panelových domů v Praze[62] v metrech.

Příklad nedostatečného otagování budov je vidět na panelových domech ve středu města. Na obrázku 5.6 je vidět, že budovy jsou všechny minimálně 10 pater vysoké, ale na obrázku 5.7 je vidět, že atribut výšky `levels` je chybně nastaven na hodnotu 1.

Obrázek 5.6: Pohled na panelové domy[46]



Obrázek 5.7: Chybějící atribut výšky[36]

Pravidla v `BuildingShell.cga` odkazují na tři další pravidla vyskytující se ve složce `Referenced`. Jsou jimy pravidla `Building_Facades` pro vzhled fasád, `Building_Interiors` upravující vzhled podlah a interiéru budovy a `Building_Roofs` nastavující různé střechy.

Pravidlo `Building_Facades` bylo mírně upraveno tak, aby byla fasáda budovy ovlivněna typem budovy podle RÚIAN[72].

Následuje ukázka, jak se nastavuje fasáda podle hodnoty atributu `building:ruian:type`

```
attr Exterior_Wall_Texture =
    case building__ruian__type == 1: "Stucco Redlands Red"
    case building__ruian__type == 2: "Stucco Redlands Brown"
    case building__ruian__type == 3: "Stucco Redlands White"
    ...
```

Konkrétní fasády jsou vybrány náhodně z přiložených textur. Zpřesnění fasád by pomohl výzkum toho, jaký typ budovy má jakou nejčastější fasádu. Vzhledem k tomu, že tato práce má za cíl vytvořit modulární program, je tato část ponechána takto a lze na ni v budoucnu navázat a hodnoty zpřesnit. Je také možné se inspirovat přístupem a ke každému typu budovy zavolat specifický CGA styl.

Ukázka aplikace CGA pravidla pro budovy je na obrázku 5.8.

(a) CE

(b) Mapy.cz

Obrázek 5.8: Ukázka pravidla pro budovy - CityEngine (nahoře) a mapy.cz (dole).

Buildings - vylepšení

Pro vylepšování budov navrhuji zaměřit se na vytvoření více pravidel pro vytváření fasád. Aktuální pravidlo má pěkně vyřešené balkony a okna, bohužel nic dalšího. Vhodná je analýza možných budov a vytvoření pravidel podle jejich vzhledu. CityEngine umožňuje vytváření komplexních a realistických budov. Je k tomu ale zapotřebí projít si důkladně manuál[42] a seznámit se se všemi funkcemi na úpravu objektů.

Místo výše zmíněného výběru textury podle typu budovy lze vytvořit různá pravidla pro rodinné domy, industriální budovy, nádražní haly, školy atd.

Obecné zlepšení realističnosti lze rozšířením `osm` dat o atributy jakými je typ střechy, výška budovy, typ fasády a další. Možné atributy pro všechny vrstvy lze nalézt na wiki[68] stránkách `osm` projektu.

Bridges

Parametry pro Python skript `bridges.py`:

```
ruleName = "rules/Bridge/Pier.cga"
startRule = "PierShape"
```

Soubor obsahující CGA pravidla pro mosty se nazývá `Pier.cga`. Pravidlo bylo vystaveno na stránkách Esri komunity[19]. Původně bylo použito na generování mostů, ale automatická aplikace nefunguje podle očekávání a proto se přiexportu do UnrealEnginu mosty musí smazat. Vstupní a koncová část mostu se totiž také oplotí a z toho důvodu nelze přes most přejet.

Pro použití tohoto pravidla by bylo potřeba jej ještě upravit. Pravidlo bylo vytvořeno pro typ objektu `Shape` ale výše generované mosty jsou typu `Graph Segment`. Proto se při aplikování pravidla na větších silnicích objevuje zábradlí jak vedle silnice, tak vedle silnicí vygenerovaného chodníku pro pěší. Na

Indexy, kterými se nastavují strany, které není třeba oplocovat se v CGA pravidle nalézají pod názvy:

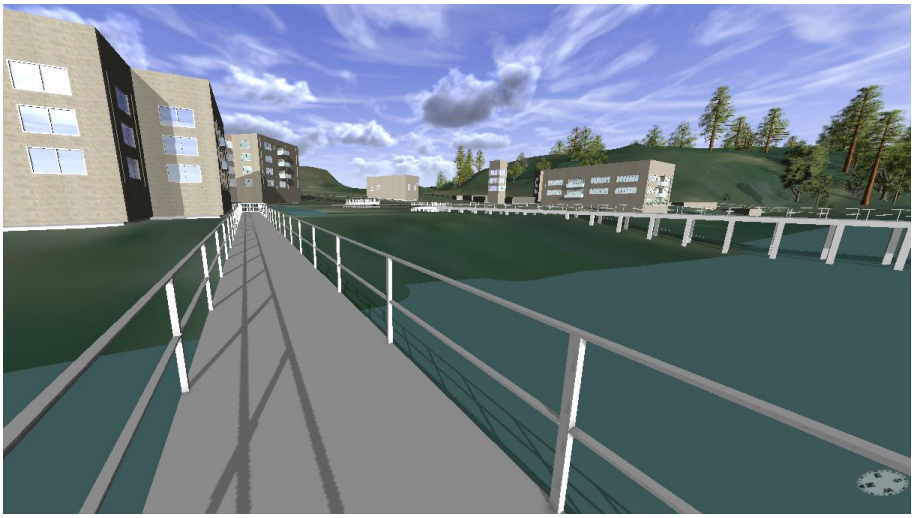
```
attr OmitEdgeNumber1
attr OmitEdgeNumber2
```

Ukázka aplikace CGA pravidla pro mosty je na obrázku 5.9. Obrázek nemá zdvojené zábradlí, jelikož se jedná o menší silnici (nikoli o dálnici či dvoupruhovou silnici).

Bridges - vylepšení

Pravidlo je potřeba upravit tak, aby se v případě uzavřených segmentů cesty nevytvářely na okrajích kruhové oplocené části. Pokud je most součástí silnice (atribut bridge=yes), vygeneruje se most správně. Když jsou ale data stahována z vrstvy `bridges`, vždy jsou z obou stran ohraničeny kruhovými částmi a tedy přes ně při vytvoření zábradlí nelze projet.

Pravidlo již obsahuje možnost zadat indexy stran, které se oplotit nemají. Bohužel pro každý most je tento index jiný. Jedna možnost je, v případě malého množství mostů, upravit tyto indexy ručně. Druhá možnost je se pravidlem pouze inspirovat a napsat si nové.



(a) CE

(b) Mapy.cz

Obrázek 5.9: Ukázka pravidla pro mosty - CityEngine (nahoře) a mapy.cz (dole).

Highways

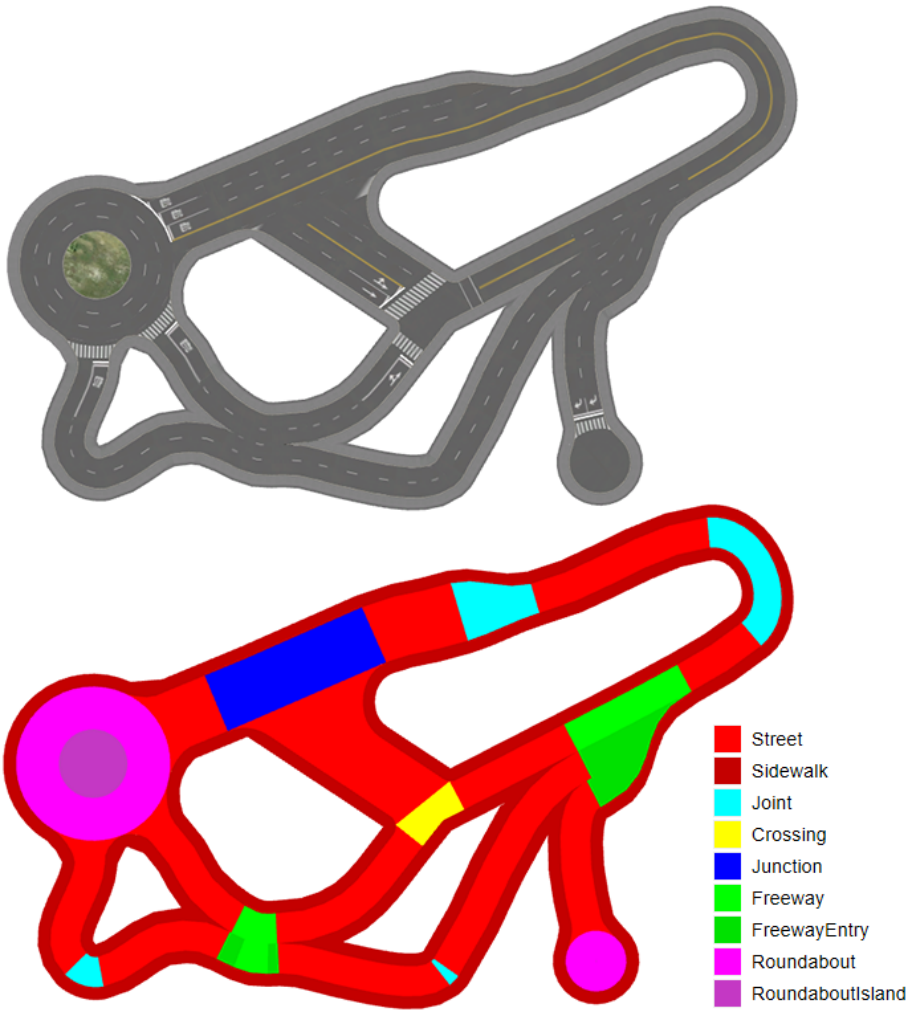
Parametry pro Python skript `highways.py`:

```
ruleName = "rules/Highway/Highway_Simple.cga"
startRule = ""
```

Soubor obsahující CGA pravidla pro mosty se nazývá `Highway_Simple.cga`. Jako jediný skript je použit výchozí CityEngine skript. Ten přebírá z `osm` atributů typ silnice a podle toho nastavuje šířku. Existují dvě varianty - jednoduchá, která obarví silnice na barvu asfaltu a složitější obsahující i možnost výběru barvy středového pruhu, přidružení chodníku pro pěší a další.

Vzhledem k tomu, že CityEngine aplikuje toto pravidlo na jakoukoli importovanou vrstvu typu `GraphSegment`, je startovní pravidlo vynecháno - CityEngine sám rozlišuje 9 typů startovních pravidel u silnic, jak je popsáno v manuálu[42] a vidět na obrázku 5.10 převzaného z manuálu.

Ukázka aplikace CGA pravidla pro silnice je vidět na obrázku 5.8 s budovami nebo na obrázku 5.11 s železnicí.



Obrázek 5.10: Ukázka silniční sítě a reprezentace typů objektů v CE[42]

Highways - vylepšení

V `osm` datech existují atributy na označení maximální rychlosti (maxspeed) či , jednosměrky (oneway=yes), podle kterých by se daly přidávat značky s omezením. Výchozí skript nastavuje šířku a vzhled silnice podle typu (jednosměrná ulice má malý obrubník, vícepruhá silnice má navíc středovou čáru). Lze rozšířit pravidla u křižovatek o semafory. Vzhledu silnic se dá věnovat spousta času, na druhou stranu mnohem důležitější je vyřešit následující dva problémy: nezapouštění silnice do terénu a vyřešení kolizí.

Aktuálně tyto problémy nejsou nijak ošetřeny. Problém s prolínáním vrstvy terénu a silnice se dá řešit buď úpravou terénu (vyrovnání v okolí silnic), nebo přidáním více referenčních bodů k silnicím, podle kterých se vypočítává zarovnání s terénem.

Kolize umí CityEngine řešit automaticky pomocí čistících nástrojů - bohužel se tím ale často silniční síť výrazně upraví a dále již neodpovídá reálného světu. Takto chybné úseky CityEngine zobrazí červeně a je potřeba manuálně silnice upravit tak, aby byly sjízdné (dalo se po nich přejet) a zároveň co nejvíce odpovídaly realitě.

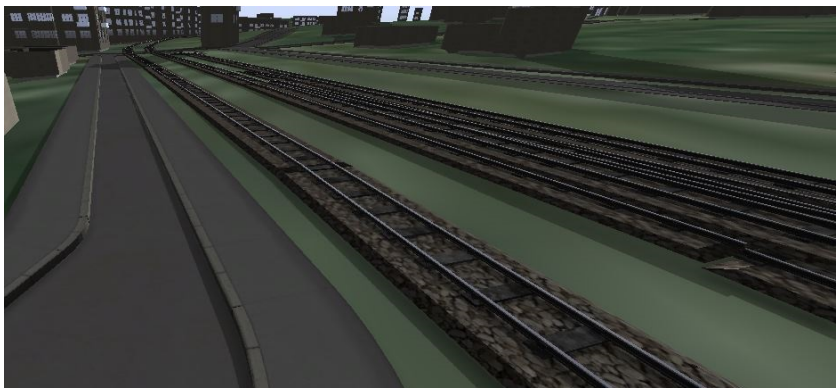
Railways

Parametry pro Python skript `railways.py`:

```
ruleName = "rules/Railway/tracksRule.cga"
startRule = "Street"
```

Soubor obsahující CGA pravidla pro mosty se nazývá `tracksRule.cga`. Při hledání vhodných pravidel byl na stránkách komunity přiložen tento skript[20] jako zip soubor. Autorem je Matthias Buehler. Pravidlo sestává z nastavitelných parametrů velikosti šířky a délky kolejnic, obsahuje texturu kolejnice i okolí kolejí.

Ukázka aplikace CGA pravidla pro železnici je na obrázku 5.11.



(a) CE

(b) Mapy.cz

Obrázek 5.11: Ukázka pravidla pro železnici - CityEngine (nahoře) a mapy.cz (dole).

Railways - vylepšení

Nalezené pravidlo vypadá velmi dobře. Obstojně zvládá větvení trati. Opět jsou slabou částí pravidla koncová kruhová zakončení, kdy koleje vedou do kruhu. Vzhledem k tomu, že z řidičského hlediska není podstatné, jak vypadá začátek a konec kolejí, není potřeba tuto vrstvu výrazně upravovat.

Naopak je potřeba ošetřit místa přejezdů - při zarovnávání vrstev železnice a silnice s terénem se tyto vrstvy překrývají a může pod silnicí prosvítat textura kamene. Opět lze tento fakt řešit individuálně, neboť když ze silnice pouze vystupují kolejnice, může být tento stav žádaný.

Waterways

Parametry pro Python skript `waterways.py`:

```
ruleName = "rules/Waterway/water.cga"
startRule = "Street"
```

Soubor obsahující CGA pravidla pro vodní plochy se nazývá `water.cga`. Tento skript je celý nově vytvořený. Dá se nastavovat barva vody pomocí hodnot RGB. Jak bylo výše vysvětleno, CityEngine nedokáže importovat do scény koryto řeky Sázavy, ale pouze body vyznačující její směr. Aby se rozlišila řeka (river) od potoku (stream) a hráze (weir), byla před aplikací `CGA` pravidel přes skript upravena šířka toků podle typu.

Skript `waterways.py` je první ukázkou rozšíření z důvodu úpravy importovaných objektů. Všechny výše uvedené skripty obsahovaly pouze informaci o `ruleName` a `startRule`. Ve skriptu `waterways.py` se nachází metoda `changeWidth()` upravující šířku jednotlivých částí toků podle typu. Aktuálně se používají hodnoty atributu `waterway` `river`, `canal`, `stream`, `weir` a `riverbank`. V okolí Týnce se nachází část `riverbank`, bohužel ale ne u nejvýraznějšího toku - Sázavy.

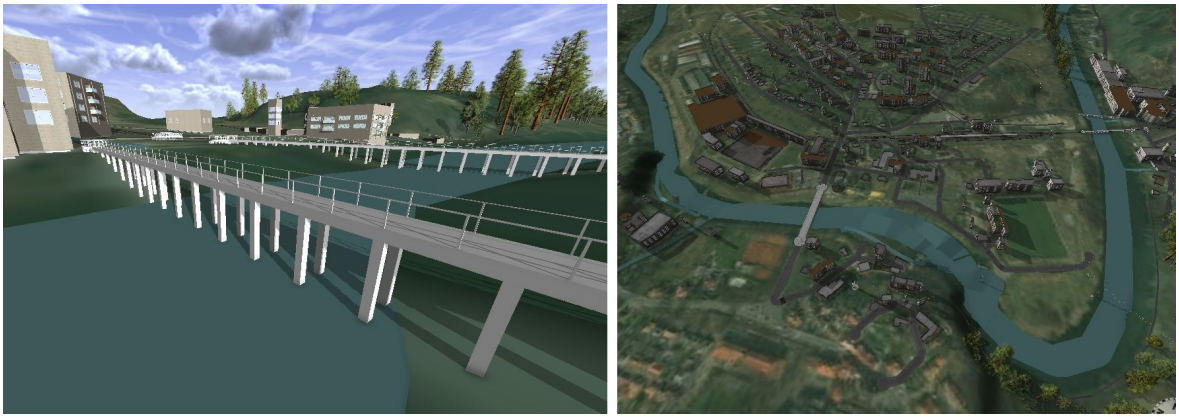
Metoda vybere všechny části vrstvy s vodou a po jednom segmentu nastavuje šířku. Při importu tento proces zabere řádově desítky sekund. Kdyby se nejdříve aplikovala `CGA` pravidla, při rozšíření by mohla vzniknout další zakončení (viz startovací pravidla u silnic výše), která by se ale automaticky vytvořila se silničním pravidlem. Proto je potřeba dodržet pořadí nejdřív import dat, následně změna šířky silnic a teprve potom aplikace `CGA` pravidel.

Ukázka aplikace CGA pravidla pro vodní plochy je na obrázku 5.12.

Waterways - vylepšení

Vrstvu s vodními toky postihuje stejný problém jako vrstvu se silnicemi - místy kolidují s terénem. Další problém je výše zmíněná nemožnost importu koryta Sázavy. Nepodařilo se najít příčinu, proč se ve vzdálenější části řeky atribut `waterway=riverbank` nalézá a v městské části ne.

Další vylepšení je použití textury s UV či animací, aby řeka nebyla pouze modrá plocha, ale vypadala realisticky.



Obrázek 5.12: Ukázka pravidla pro vodní plochy

Landuses

Parametry pro Python skript `landuses.py`:

```
ruleName = "rules/Landuse/landuse.cga"
startRule = "Lot"
```

Pomocí atributu `landuse` se popisuje způsob využití pozemku člověkem. Vzhledem k tomu, že tento klíč může nabývat cca 30 hodnot, byl na vytvoření `CGA` pravidla použit skript pro vytváření pravidel automaticky. Ze stránek openstreetmap[61] byl vytvořen `csv` soubor obsahující hodnoty, jichž může atribut nabývat. Tento byl následně přidán do složky `data/csv/` jako `landuse.csv`. Pomocný skript pro generování `CGA` pravidel `_createCGA.py` byl upraven, aby nastavoval jednotlivým objektům texturu.

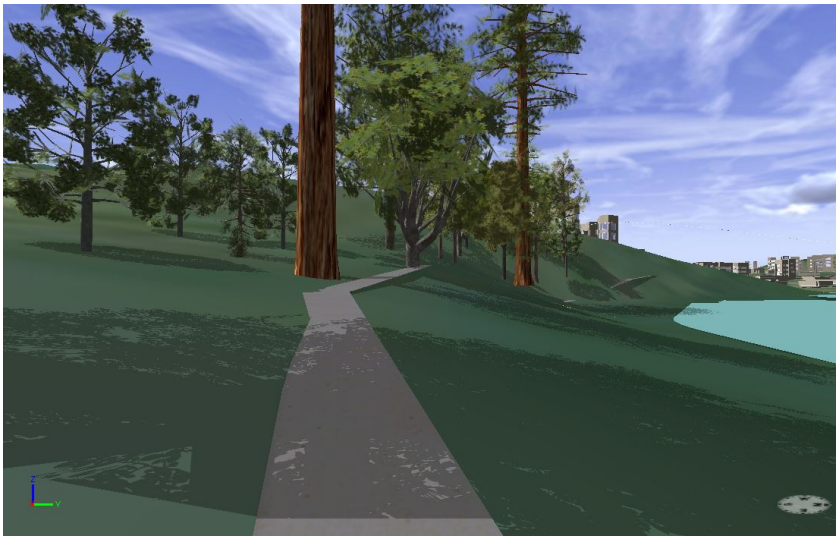
`CGA` pravidlo z objektu vezme atribut `landuse` a podle jeho hodnoty aplikuje texturu. V CityEnginu je potřeba k textuře přidat i informaci o UV projekci, jinak se nevykreslí. Aktuálně pravidlo nastaví všem plochám kromě `farmland` a `meadow` (nejčastěji se vyskytujícím hodnotám v Týnci) travnatou texturu. Těmto dvěma nastaví jinou texturu, aby nebyla tráva všude stejná. Další výjimkou je hodnota `forest`, která aplikuje pravidlo `Forest` generující na ploše stromy. Toto pravidlo je také součástí CityEnginu. Lze na něm upravit typ stromů, které se generují, obsahuje funkce, aby se stromy nepřekrývaly a lze je po ploše distribuovat náhodně, nebo například v řadách.

Vzhledem k tomu, že i objekty mají problémy se srovnat s terénem, po importu vrstvy se ještě v `main.py` skriptu smažou objekty s hodnotami `residential`, `garages`, `industrial` a `commercial`. Tyto objekty často zasahují do již pěkně vygenerovaných částí jakými jsou budovy. Ve skriptu lze snadno přidat vrstvu ke smazání - stačí její hodnotu napsat do připraveného listu `valsToDelete`.

Ukázka aplikace CGA pravidla pro využití půdy je na obrázku 5.13

Landuses - vylepšení

V první řadě je nutné upravit skript na generování lesa - aktuálně se plocha osází stromy, ale uživatel by jistě měl přehled o tom, kde se který strom nachází. Dalším problémem je fakt,



(a) CE

(b) Mapy.cz

Obrázek 5.13: Ukázka pravidla pro využití půdy - CityEngine (nahoře) a mapy.cz (dole).

že stromy takto vygenerované se nedají upravovat a chovají se jako jedna vrstva - některé v kopcovitém terénu mohou levitovat a jiné být zabořené. Na takto vygenerované stromy nefunguje zarovnání s terénem.

Případně je možné celou tuto vrstvu vynechat a generovat stromy jiným způsobem. Například si zapamatovat souřadnice, kde se stromy mají nacházet a vygenerovat je buď v UnrealEnginu nebo v následném programu, kam se scéna ze CityEnginu importuje.

Dále povrch řešení pouze aplikací textury není realistický. Je zapotřebí pokrýt plochy travou či texturou s UV.

Celá tato vrstva je primárně vytvořena proto, aby se v budoucnu nemusel používat terén s texturou. V ideálním případě se výšková mapa zachová, ale pokryje se realističtějšími prvky.

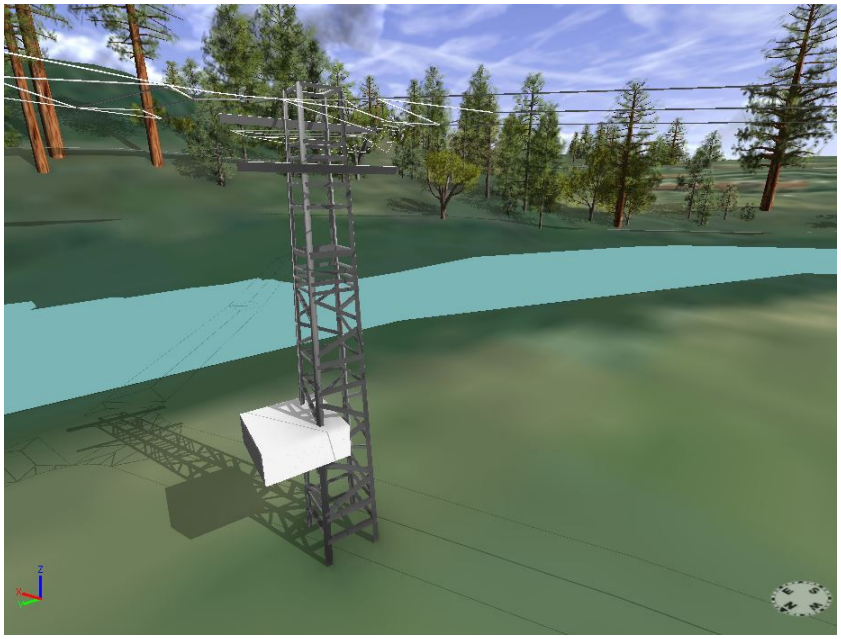
Powers

Parametry pro Python skript `powers.py`:

```
ruleName = "rules/Power/Tower.cga"
startRule = "Pole"
```

Vrstva s elektrickým vedením byla přidána jako ukázka importu objektů. V `osm` datech se elektrické vedení nachází pod atributem `power`. Opět může nabývat několika hodnot, v `CGA` pravidle se ale pracuje pouze se třemi: `pole`, `minor_line` a `tower`. Podle hodnoty atributu `power` se v gramatice volá pravidlo pro vložení objektu nebo vytvoření elektrického vedení. Pro správnou funkčnost je potřeba, aby byl pivot objektů vycentrovaný a nacházel se na spodní straně modelu. Díky tomu bude odpovídat umístění objektu.

Ukázka aplikace CGA pravidla pro využití elektrického vedení je na obrázku 5.14.



Obrázek 5.14: Ukázka pravidla pro využití elektrického vedení

Powers - vylepšení

Lze změnit načítané modely za realističtější. Dále je potřeba upravit skript na vytváření drátů. Aktuální verze vytvoří tři rovnoběžné dráty. Pro lepší realistišnost je potřeba vedení nemít pouze jako rovné pruty, ale prověsit je.

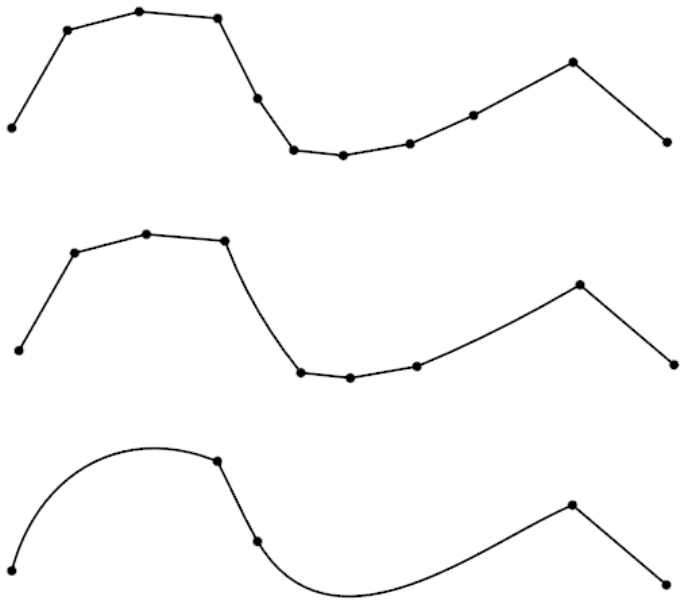
Další možnost zlepšení je přidání více modelů a rozšíření `CGA` pravidla o další hodnoty, kterých může atribut `power` nabývat.

5.2.3.4 Problémy

V následující části jsou vypsány problémy, se kterými jsem se při implementaci setkala, a které jsou potřeba v budoucnu vyřešit.

Konflikty u silnic

CityEngine je nástroj primárně určen k vytváření fantastických a imaginárních měst. Používá se zpravidla pro vizualizaci shora. Proto obsahuje pravidla, která vypadají pěkně z dálky, ale detaily propracované nejsou. Pro silnice a grafovou vrstvu lze spustit pomocné nástroje na práci s kolizemi a konflikty. Jsou jimi nástroj na zjednodušení sítě (Simplify Graph), který podle zadaného thresholdu zjednoduší křivky, jak je vidět na obrázku 5.15. Další je nástroj na čištění grafu (Cleanup Graph Tool). Ten opraví nepřesnosti v segmentech, jak je vidět na obrázku 5.16.

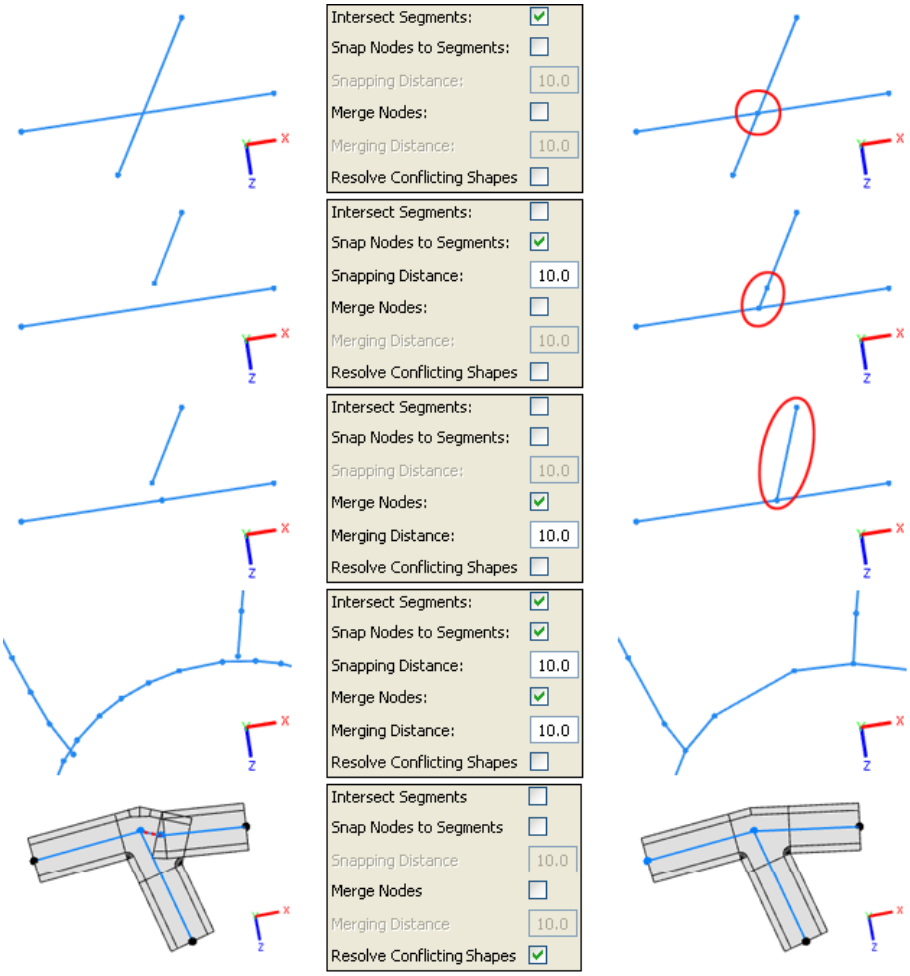


Obrázek 5.15: Příklad využití nástroje Simplify Graph. Nahoře importovaná silnice, uprostřed threshold = 10, dole threshold = 50[42]

Tyto nástroje sice vytvoří pěkně vypadající silnice, ale ty už by neodpovídaly realitě. Všechny nalezené problémy jsou po importu zvýrazněny červenou barvou. Při nastavování parametrů se nepodařilo vytvořit takovou kombinaci, aby křižovatky odpovídaly reálnému světu. Proto je potřeba konflikty zatím vyřešit ručně.

Zarovnání s terénem

Další problém se silnicemi je jejich zarovnávání do terénu. I na tento požadavek je v CityEnginu připraven nástroj s názvem Align Graph to Terrain, který vyrovná silnice s terénem.



Obrázek 5.16: Příklady využití nástroje Cleanup Graph[42]

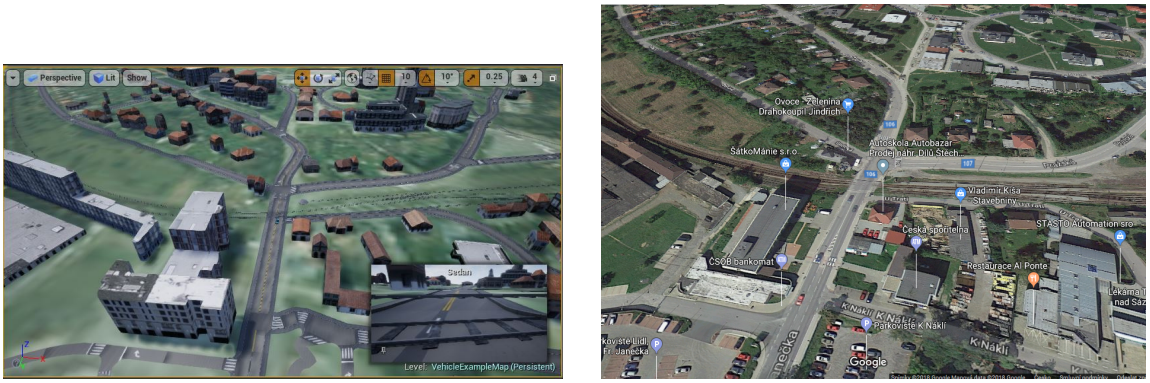
Bohužel počítá pouze s vrcholy silnice, a proto se některé úseky "zanořují"do terénu. Řešením by bylo graf představující silnici převzorkovat, aby obsahovala více vrcholů. Na druhou stranu by se tím zvýšila paměťová náročnost dat. Dále je možné testovat viditelnost silnice shora a tak, kde je silnice pod terénem terén upravit. Tato oprava nebyla provedena, ale je nezbytná. Pro snížení zanořených silnic a dalších objektů byl po importu terén posunut níže.

Tvar řek

V nástroji na zobrazení `osm` dat `JOSM` bylo zkontrolováno, že řeky a vodní toky jsou tvořeny atributy `riverbank` a `multipolygon`, které udávají jejich konkrétní tvar. Bohužel tento atribut CityEngine nedokáže zpracovat a tudíž v aktuální verzi jsou řeky importované jako klasické silnice a jejich tvar neodpovídá skutečnosti. Sázava má nyní nastavenou větší šířku, aby lépe odpovídala skutečnosti, a menší potoky a řeky jsou užší.

5.2.3.5 Export modelu

CityEngine umožňuje export do více formátů, např. `FBX`, `OBJ` nebo podporuje v Beta verzi export přímo do Unreal Enginu. Tuto možnost poskytuje Unreal Engine plugin s názvem `uDataSmith`, díky kterému se data včetně textur dají importovat přímo do scény v Unreal Enginu. Ukázka modelu v Unreal Enginu je na obrázku 5.17. Na obrázku je vidět, že ačkoli `Google maps` umožňuje úpravu úhlu pohledu, pouze překresluje 2D data. Vysoké budovy jsou zkreslené a naopak ve scéně s Unreal Enginem vypadají lépe, než na pohledu z `Google maps`.



(a) Scéna v Unreal Enginu

(b) Google map street view

Obrázek 5.17: Srovnání scény z Unreal Enginu (vlevo) a Google maps (vpravo)

Unreal Datasmith

Jedná se o soubor nástrojů pro import dat to `Unreal Enginu`. Obsahuje sadu pluginů umožňující uživateli jednoduše importovat data z více než dvaceti CAD a DCC 3D formátů. Unreal Datasmith významně snižuje čas potřebný k úpravě přesouvaných dat[25].

CityEngine Template Project

Od verze CityEngine 2017.1 byla přidána podpora exportu do Unreal Enginu. Dříve bylo nutné využívat převod do `FBX`. Nové řešení je efektivnější a pojme i větší scény s desítky miliony polygony a objekty. Také už není limitující počet materiálů.

Ve verzi CityEngine 2018 byly opraveny drobné chyby, jako například odstranění úniků paměti (memory leaks) a ošetření limitu maximálního počtu objektů v UnrealEnginu.

V UnrealEnginu je možné z MarketPlace stáhnout CityEngine Template Project obsahující shadery a optimalizovaná světla pro vizualizaci na bázi CityEnginu. Materiál importované scény se automaticky propojí s Unreal Engine shadery. Díky tomu není potřeba žádná manuální úprava scény.

Pro využívání pluginu v Beta verzi je potřeba se přihlásit k odběru `Unreal Studio Beta`. Po přihlášení se v `Epic Games launcheru`, objeví sekce `UnrealStudio Beta`, ve kterém lze plugin najít.

Další přístup

Aby bylo možné v budoucnu přejít i na jiný engine, je praktičtější využít export do podporovaného 3D formátu (např. `OBJ`). Při následném importu do scény v Unreal Enginu už je zapotřebí objekty otočit o 90 stupňů v ose `x` a zvětšit škálování na hodnotu `100`, jelikož měřítko dat je v metrech, zatímco v UnrealEnginu ve výchozím nastavení je měřítko v centimetrech. Dále je zapotřebí u všech objektů nastavit u vytvoření kolizní obálky "Use Complex Collision As Simple", aby se vytvořila těsná obálka. V opačném případě by měly všechny objekty kvádrovou kolizní obálku a případné auto by nemohlo scénou projíždět, jelikož by se vyskytovalo v obálce. Pokud by k tomu došlo, auto by se začalo nekontrolovatelně točit.

5.3 Zhodnocení

Tento přístup má velké ambice co se týče realističnosti a snadné úpravě města. V aktuální verzi se do scény načte terén vygerenovaný samotným CityEnginem a pomocí python skriptu se vygeneruje podle daných pravidel základ realistického města. Oproti prvnímu přístupu jsou data realističtější. Přístup je modulární z pohledu úprav gramatik a skriptů po vrstvách. Lze importovat a používat hotové modely, nebo aplikovat a rozšiřovat existující gramatiky.

Nevýhoda tohoto přístupu je v nutnosti pořízení softwaru. Také tento nástroj není všemocný a je potřeba ručně upravit chybné úseky. Například se ne vždy správně vygenerují dobře mosty a silnice. Aby scéna odpovídala realitě, jsou nutné následné ruční opravy nebo se zaměřit na opravu samotných `osm` dat, ze kterých se scéna generuje.

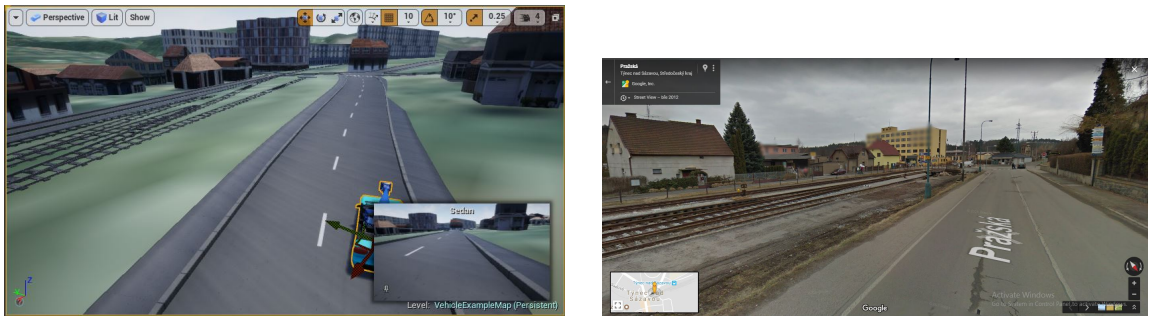
Kapitola 6

Testování

Testování proběhlo na datech o velikosti 5x5km. Po stažení vektorových dat a dat terénu byl spuštěn upravující skript, který vytvořil v CityEnginu scénu Týnce nad Sázavou. Pro porovnání realističnosti byly použity `Google street view`[75] mapy a rozhraní panorama od `mapy.cz`[46]. Srovnání s `Google street view` je vidět na obrázku 6.1.

Hlavní rysy scény odpovídaly skutečnosti. Silnice odpovídaly svým tvarem realitě, místy se ale stávalo, že byly částečně zapuštěny do terénu. Je to z toho důvodu, že při mapování silnic na terén se mapuje pomocí klíčových bodů, což jsou body určující tvar silnice. Těchto bodů je na rovných úsecích málo, jelikož není potřeba pro přímou silnici detailnějších informací. To s sebou ale nese problémy s mapováním v případě pokřiveného terénu. Jako řešení tohoto problému by bylo potřeba přidat více klíčových bodů určujících trasu a směr silnice, čímž se na jednu stranu realističnost zpřesní, na druhou stranu jsou pak objekty složitější.

Oproti reálnému prostředí neodpovídají vzhledem fasády budov. Pro větší realističnost bylo využito pravidlo generující i balkony a okna. V aktuální verzi se barva fasády mění podle typu budovy. Fasády jsou ale namapovány náhodně - například obytná budova je z kamene, industriální ze dřeva.



(a) Scéna v Unreal Enginu (b) Google map street view

Obrázek 6.1: Porovnání scény z Unreal Enginu (vlevo) a Google maps (vpravo)

Některé silnice mají trochu jiný tvar než ve skotečnosti. Je to dáno tím, že CityEngine byl původně projekt pro modelaci smyšlených měst a tudíž obsahuje nástroje pro zjednodušení silniční sítě či vyřešení nepřesností. Jeho nástroje pro vyčištění grafu silniční sítě například sdružuje silnice, které jsou blízko u sebe, nebo v případě, že silnice skončí těsně před další,

tak ji na ni napojí. Pro nerealistické scény je tento přístup vhodný, jelikož je tím docíleno ucelené město s hezky vypadající a funkční silniční sítí. Naopak pro realistické scény je nutné používat čistící algoritmus s rozmyslem, aby se zachovalo co nejvíce původních importovaných informací. Menší úpravy jsou ale vyžadovány v obou případech, jelikož při generování silnic a úpravy šířky se může stát, že na sebe nepasují křižovatky. V tom případě je lepší graf mírně upravit, aby se ve scéně neobjevila křižovatka, kterou nelze projet. Také mosty je potřeba zkontrolovat ručně. Někdy se v datech objevují atributy, které s rozhodnutím, zda je daný úsek most ulehčují. Nástroj pro vytváření mostů v CityEnginu tvoří mosty z užších silnic, které křižují silnice širší (například silnice 1. třídy nebo dálnice).

Další problém je vidět při průjezdu po mostě přes řeku Sázavu. V `osm` datech je u větších řek vždy pomocný polygon s atributem `riverbank`, který určuje tvar koryta řeky. Přes program `JOSM` bylo ověřeno, že atribut s břehy řek se na mapě Týnce nachází. Při kontrole atributů se kromě `riverbank` u daného objektu vyskytovalo i `Uncomplete polygon`. `Osm` data jsou tvořena mj. i relacemi, proto se s výřezem z mapy vždy stahují i části, které oblast přesahují. Děje se to právě z toho důvodu, aby relace byly kompletní. V tomto případě se ale do scény s Týncem nepodařilo atribut `riverbank` dostat. Proto byla Sázava vygenerována jako klasická úzká cesta a následně byla rozšířena. Z toho důvodu je po celé délce stejně široká, což neodpovídá skutečnosti.

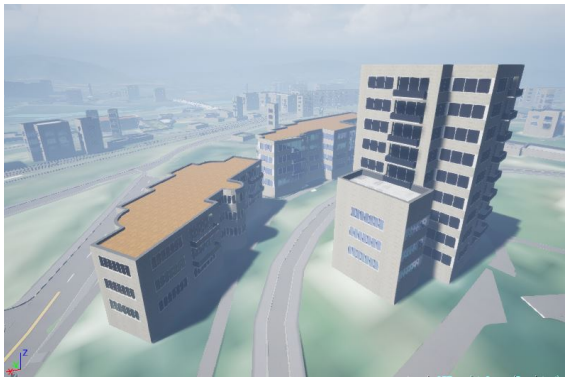
Celkově se podařilo vytvořit scénu, která s určitými nepřestnostmi představuje model z mapy okolo Týnce nad Sázavou, jak je vidět na obrázku 6.2. Další vývoj a práce je pro realističnost nutností. Bohužel se nepodařilo nasnímat videosekvence z důvodu použití nedlaždicového přístupu a terénu. Jak bylo popsáno výše, nástroj na přidání georeferenčních dat vracel špatnou hodnotu pro zeměpisnou délku. Ve scéně v UnrealEnginu lze ovládat vozidlo a scénou sekaně projíždět, pro plynulé video by ale bylo zapotřebí přidat možnost postupného načítání dat, nikoli načtení celé velké scény se všemi objekty a terénem.

6.1 Statistiky

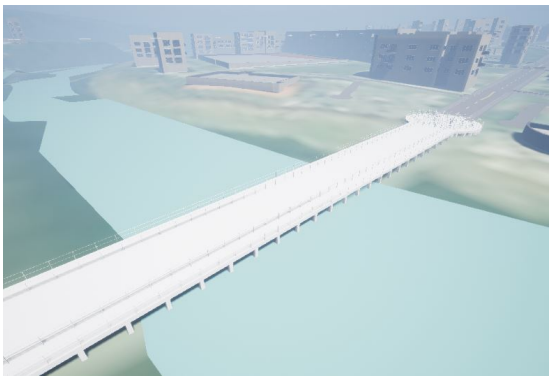
Po importu všech výše uvedených vrstev se ve scéně nachází

- 322 objektů budov
- 1 objekt a 472 segmentů železnic
- 3520 segmentů silnic
- 40 segmentů mostů
- 3 objekty a 281 segmentů elektrického vedení
- 2053 segmentů vody
- 4 objekty půdy

Celkem se jedná o 4142 objektů. Export ze CityEnginu do formátu UDatasmith, který je kompatibilní s Unreal Enginem se vytvoří do jedné minuty. UDataSmith soubor má velikost 8552 KB, ale k němu přidružená složka s assety má 236 MB. Následný import do Unreal

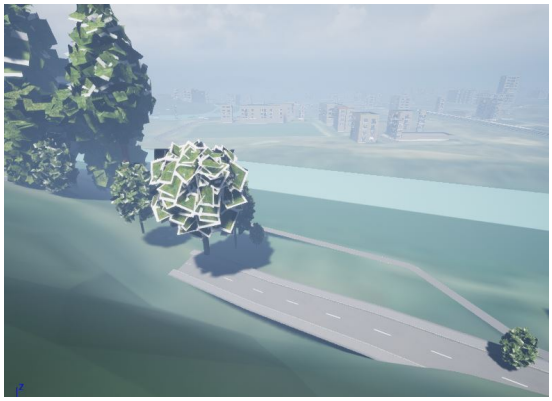


(a) Budovy a silnice



(b) Most a řeka

(c) Elektrické vedení a železnice



(d) Stromy a cesty

Obrázek 6.2: Další ukázky finální scény v Unreal Enginu

Enginu se pohybuje v řádu jednotek minut, přibližně 4 minuty. V Unreal Enginu je objektů ve scéně mnohem více (25634), jelikož oproti CityEnginu se zde například stromy reprezentují po jednom a díky tomu se dají posunout a scéna ještě finálně upravit.

Kapitola 7

Závěr

Cílem diplomové práce bylo navrhnout postup, který z výřezu z mapy o velikosti 5x5km nacházející se v okolí Týnce nad Sázavou vytvoří realistický model světa. Byly navrženy dva přístupy - první klade větší důraz na modularitu a využívá open source řešení, druhý klade důraz na realističnost a využívá software CityEngine. Oba přístupy byly testovány na výřezu z mapy v okolí Týnce nad Sázavou.

Ze dvou výše uvedených přístupů byl jako hlavní zvolen postup kladoucí důraz na realističnost. Jelikož má práce v budoucnu sloužit pro testování algoritmů pro autonomní vozidla, je důležité, aby scéna co nejvíce odpovídala realitě. Zásadní překážkou je v případě open source varianty především nemožnost do programu importovat data terénu. Sami autoři programu `OSM2World` navíc tuto funkcionalitu (import `SRTM` dat) uvádí jako experimentální a s množstvím nedostatků a omezení. Dalším kritériem pro výběr byla rovněž skutečnost, že software CityEngine je aktivně vyvíjen - každý rok vychází nová verze představující vylepšení a opravující známé chyby. Open source varianta je postavena na programu `OSM2World`, který je aktualizován nepravidelně a v menším rozsahu.

Vybraný postup ze souřadnic získaných výběrem oblasti na mapě vytvoří výškovou mapu. Následuje import jednotlivých vrstev, přičemž každá vrstva je řízena svým vlastním skriptem v jazyce Python a `CGA` pravidly určujícími její vzhled. Každá vrstva je detailně popsaná výše a je k ní přidáno i doporučení na následný vývoj.

Po vytvoření celé scény v CityEnginu se následně exportuje do `uDataSmith` formátu, který optimalizuje import scény ze CityEnginu do Unreal Enginu. Pro nezávislost na výchozím enginu poskytuje CityEngine i možnost exportovat data do formátu `obj`, `fbx` a dalších. Scéna byla následně importována do herního enginu Unreal Engine, jak bylo požadováno v zadání práce.

Testování proběhlo v prostředí vygenerovaném CityEnginem a v Unreal Enginu. Pro ovládání vozidla byl do `Unreal Enginu` přidán plugin integrující simulátor `AirSim`.

Videosekvence z průjezdu nebyly v této práci realizovány, jelikož import do Unreal Enginu není tvořen po dlaždicích, jak bylo výše vysvětleno. Generování po dlaždicích by mělo mít v případném rozšíření této práce hlavní prioritu.

Literatura

[1] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.

[2] E. Galin, A. Peytavie, N. Maréchal, and E. Guérin. Procedural generation of roads. *Computer Graphics Forum*, 29(2):429–438, 2010.

[3] R. Janovský. *Operátorské stanoviště pro vizualizaci a řízení autonomních bezpilotních prostředků*. Praha, 2017. Diplomová práce. České vysoké učení technické v Praze. Vypočetní a informační centrum.

[4] M. M. Kazhdan and H. Hoppe. Screened poisson surface reconstruction. *ACM Trans. Graph.*, 32(3):29:1–29:13, 2013.

[5] M. Mueller, V. Casser, J. Lahoud, N. Smith, and B. Ghanem. UE4Sim: A Photo-Realistic Simulator for Computer Vision Applications. *ArXiv e-prints*, Aug. 2017.

[6] S. Shah, D. Dey, C. Lovett, and A. Kapoor. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and Service Robotics*, 2017.

[7] 3D online viewer. [online]. [cit. 2018-1-13].
Dostupné z: `https://3dviewer.net/`.

[8] Airsim. [online]. [cit. 2018-2-28].
Dostupné z: `https://github.com/Microsoft/AirSim`.

[9] Knihovna pro převod mezisouřadnými soustavami. [online]. [cit. 2018-5-22].
Dostupné z: `http://www.maptiler.org/google-maps-coordinates-tile-bounds-projection/`.

[10] Apollo simulation platform. [online]. [cit. 2018-4-18].
Dostupné z: `http://apollo.auto/platform/simulation.html`.

[11] Úrovně autonomních vozidel. [online]. [cit. 2018-5-9].
Dostupné z: `http://www.auto.cz/urovne-autonomnich-aut-jaky-nimi-rozdil-a-ktera-opravdu`

[12] Samořídítelná vozidla. [online]. [cit. 2018-5-24].
Dostupné z: `https://en.wikipedia.org/wiki/Autonomous_car`.

[13] Výšková data služby mapbox. [online]. [cit. 2018-4-12].
Dostupné z: `https://www.mapbox.com/help/access-elevation-data/`.

[14] CARLA simulátor. [online]. [cit. 2018-1-13].
Dostupné z: https://carla.org/.

[15] Carla úprava map. [online]. [cit. 2018-4-17].
Dostupné z: http://carla.readthedocs.io/en/latest/map_customization/.

[16] Carla dokumentace. [online]. [cit. 2018-2-28].
Dostupné z: http://carla.readthedocs.io/.

[17] City engine. [online]. [cit. 2018-4-2].
Dostupné z: http://www.esri.com/software/cityengine.

[18] Cga pravidlo - building shell with detail. [online]. [cit. 2018-12-26].
Dostupné z: https://www.arcgis.com/home/item.html?id=475d984754c44aa0ad19281defe33c0d.

[19] Cga pravidlo - pierl. [online]. [cit. 2018-12-12].
Dostupné z: https://community.esri.com/thread/124195.

[20] Cga pravidlo - rais. [online]. [cit. 2018-12-12].
Dostupné z: https://community.esri.com/thread/171221?commentID=582552.

[21] Číselník k nemovitostem. [online]. [cit. 2018-11-9].
Dostupné z: https://www.cuzk.cz/Katastr-nemovitosti/Poskytovani-udaju-z-KN/Ciselniky-ISKN/Cis

[22] Český statistický úřad. [online]. [cit. 2018-5-4].
Dostupné z: https://www.czso.cz/csu/rso/uir_zsj.

[23] Státní správa zeměměřictví a katastru. [online]. [cit. 2018-1-18].
Dostupné z: http://www.cuzk.cz/.

[24] Mapové podklady © Český úřad zeměměřický a katastrální. [online]. [cit. 2018-5-17].
Dostupné z: geoportal.cuzk.cz/.

[25] Datasmith workflow. [online]. [cit. 2018-4-17].
Dostupné z: https://www.unrealengine.com/en-US/blog/introducing-datasmith-a-workflow-toolkit-

[26] United states geological survey. [online]. [cit. 2018-5-4].
Dostupné z: https://dds.cr.usgs.gov/srtm/version2_1/.

[27] Výzkumný ústav vodohospodářský t. g. masaryka. [online]. [cit. 2018-1-18].
Dostupné z: http://www.dibavod.cz/.

[28] Databáze epsg. [online]. [cit. 2018-4-20].
Dostupné z: www.epsg.org.

[29] Esri - environmental systems research institute. [online]. [cit. 2018-4-2].
Dostupné z: https://www.esri.com.

[30] Nařízení o interoperabilitě sad prostorových dat a služeb prostorových dat. [online].
[cit. 2018-4-21].
Dostupné z: https://eur-lex.europa.eu/legal-content/CS/TXT/?uri=CELEXy\%3A32010R1089.

[31] Geospatial Data Abstraction Library. [online]. [cit. 2018-4-12].
Dostupné z: `https://gdal.com`.

[32] Databáze map planet osm. [online]. [cit. 2018-1-13].
Dostupné z: `https://download.geofabrik.de/`.

[33] Carla github issue 110. [online]. [cit. 2018-2-28].
Dostupné z: `https://github.com/carla-simulator/carla/issues/110`.

[34] Carla github issue 224. [online]. [cit. 2018-2-28].
Dostupné z: `https://github.com/carla-simulator/carla/issues/224`.

[35] Github jameschevalier s databází ohraničení měst. [online]. [cit. 2018-1-13].
Dostupné z: `https://github.com/JamesChevalier/cities/`.

[36] Josm - java openstreetmap. [online]. [cit. 2018-12-26].
Dostupné z: `https://josm.openstreetmap.de/`.

[37] Kartografická zobrazení. [online]. [cit. 2018-5-18].
Dostupné z: `https://is.mendelu.cz/eknihovna/opory/zobraz_cast.pl?cast=59996`.

[38] Křovákovo zobrazení. [online]. [cit. 2018-5-17].
Dostupné z: `https://cs.wikipedia.org/wiki/Křovákovo_zobrazení`.

[39] Záznam z konference z unite 2016 v LA. [online]. [cit. 2018-1-13].
Dostupné z: `http://amara.org/v/08pm/`.

[40] GNU Lesser General Public License. [online]. [cit. 2018-4-12].
Dostupné z: `https://opensource.org/licenses/LGPL-3.0/`.

[41] Ministerstvo zemědělství. [online]. [cit. 2018-1-18].
Dostupné z: `http://eagri.cz/public/web/mze/farmar/LPIS/`.

[42] Cityengine manuál. [online]. [cit. 2018-12-12].
Dostupné z: `https://cehelp.esri.com/help/index.jsp`.

[43] Mapbox. [online]. [cit. 2018-4-12].
Dostupné z: `https://www.mapbox.com`.

[44] Knihovna pro použití mapboxu v unity 3d. [online]. [cit. 2018-1-13].
Dostupné z: `https://github.com/mapbox/mapbox-ar-unity`.

[45] Princip mapových dlaždic. [online]. [cit. 2018-4-12].
Dostupné z: `https://msdn.microsoft.com/en-us/library/bb259689.aspx`.

[46] Mapy.cz. [online]. [cit. 2018-5-19].
Dostupné z: `https://mapy.cz`.

[47] Mapzen - mapovací nástroj. [online]. [cit. 2018-4-18].
Dostupné z: `https://mapze.com`.

[48] Mapové zobrazení. [online]. [cit. 2018-5-18].
Dostupné z: https://cs.wikipedia.org/wiki/Mapové_zobrazení.

[49] Massive software. [online]. [cit. 2018-4-17].
Dostupné z: http://www.massivesoftware.com/.

[50] Procedural generation. [online]. [cit. 2018-4-17].
Dostupné z: https://medium.com/@homicidalnacho/a-look-into-procedural-generation-dfa62fe7536

[51] Mercatorovo zobrazení. [online]. [cit. 2018-5-18].
Dostupné z: https://cs.wikipedia.org/wiki/Mercatorovo_zobrazení.

[52] Meshlab. [online]. [cit. 2018-5-17].
Dostupné z: http://www.meshlab.net/.

[53] Norma en iso 19111. [online]. [cit. 2018-4-20].
Dostupné z: https://shop.normy.biz/detail/88890.

[54] Open Data Commons Open Database license. [online]. [cit. 2018-4-12].
Dostupné z: https://opendatacommons.org/licenses/odbl/1.0/.

[55] Open street map. [online]. [cit. 2018-4-12].
Dostupné z: https://www.openstreetmap.org/.

[56] Konvertor, který vytváří 3D model z dat z openstreetmap. [online]. [cit. 2018-1-13].
Dostupné z: http://osm2world.org/.

[57] Api pro openstreet mapy. [online]. [cit. 2018-1-13].
Dostupné z: https://wiki.openstreetmap.org/wiki/API.

[58] Osm buildings. [online]. [cit. 2018-1-13].
Dostupné z: https://osmbuildings.org/.

[59] Osmfilter. [online]. [cit. 2018-4-12].
Dostupné z: https://wiki.openstreetmap.org/wiki/Osmfilter.

[60] Osmosis - nástroj na zpracování dat z open street map. [online]. [cit. 2018-4-18].
Dostupné z: https://github.com/openstreetmap/osmosis.

[61] Wikipedia openstreetmap. [online]. [cit. 2018-2-28].
Dostupné z: https://wiki.openstreetmap.org/wiki.

[62] Panelový dům. [online]. [cit. 2018-12-12].
Dostupné z: https://cs.wikipedia.org/wiki/Panelový_dům#Typ_T_0xB.

[63] Procedural content generation wiki. [online]. [cit. 2018-4-17].
Dostupné z: http://pcg.wikidot.com/.

[64] Databáze map planet osm. [online]. [cit. 2018-1-13].
Dostupné z: https://planet.openstreetmap.org/.

[65] Mapbox - cenová politika. [online]. [cit. 2018-1-13].
Dostupné z: `https://www.mapbox.com/pricing`.

[66] Procedurální programování. [online]. [cit. 2018-4-17].
Dostupné z: `https://en.wikipedia.org/wiki/Procedural_programming`.

[67] Generický software pro transformaci souřadnicových dat. [online]. [cit. 2018-4-2].
Dostupné z: `https://proj4.org/`.

[68] Projekt Česko. [online]. [cit. 2018-5-12].
Dostupné z: `https://wiki.openstreetmap.org/wiki/Cs:WikiProjekt_Česko/freemap`.

[69] Knihovna rasterio. [online]. [cit. 2018-4-12].
Dostupné z: `https://github.com/mapbox/rasterio`.

[70] Správa základních registrů. [online]. [cit. 2018-5-4].
Dostupné z: `www.szrcr.cz/co-jsou-to-zakladni-registry`.

[71] Vector zero. [online]. [cit. 2018-12-26].
Dostupné z: `https://www.vectorzero.io/products`.

[72] Osm tag registr územní identifikace, adres a nemovitostí. [online]. [cit. 2018-11-9].
Dostupné z: `https://wiki.openstreetmap.org/wiki/Cs:RÚIAN`.

[73] Shuttle radar topography mission. [online]. [cit. 2018-4-2].
Dostupné z: `https://en.wikipedia.org/wiki/Shuttle_Radar_Topography_Mission`.

[74] Shuttle radar topography mission. [online]. [cit. 2018-5-9].
Dostupné z: `http://komunitas-atlas.blogspot.cz/2011/11/shuttle-radar-topography-missio`

[75] Google street view. [online]. [cit. 2018-5-24].
Dostupné z: `https://www.instantstreetview.com`.

[76] TORCS závodní simulátor. [online]. [cit. 2018-4-12].
Dostupné z: `http://torcs.sourceforge.net/`.

[77] Foto-realistický simulátor pro použití v počítačovém vidění. [online]. [cit. 2018-1-13].
Dostupné z: `https://ue4sim.org/`.

[78] Ústav pro hospodářskou úpravu lesů. [online]. [cit. 2018-5-4].
Dostupné z: `http://www.uhul.cz/`.

Příloha A

Seznam použitých zkratek

ODbL Open Data Commons Open Database License

LGPL Lesser General Public License

OSM Open Street Map

JSON JavaScript Object Notation

XML Extensible Markup Language

GPS Global Positioning System

CARLA Car Learning to Act

CGA Computer Generated Architecture

ČÚZK Český úřad zeměměřický a katastrální

DMR 5G Digitální model reliéfu České republiky 5. generace

DMP 1G Digitální model povrchu České republiky 1. generace

SRTM Shuttle Radar Topography Mission

EPSG European Petroleum Survey Group

CAD Computer Aided Design

DCC Digital Content Creation

GDAL Geospatial Data Abstraction Library

SAE Společnost automobilových inženýrů

Příloha B

Instalační a uživatelská příručka

Knihovny potřebné pro open source variantu jsou přiloženy ve složce `.\opensource-version`. Vzhledem k tomu, že tento přístup nebyl po vyzkoušení automatizován, uživatel si pro jeho využití musí přečíst manuály přiložených programů.

Pro CityEngine variantu jsou všechny potřebné knihovny a programy přiloženy ve složce `.\cityengine-version\dependencies-binary`

Pro správnou funkčnost automatického skriptu je zapotřebí mít v operačním systému nainstalované:

- python (testováno na verzi 3.6.5)
- curl (testováno na verzi 7.55.1)

tyto programy se spouští z příkazové řádky, je zapotřebí, aby v proměnném prostředí systému byly k těmto programům správně uložené cesty.

B.1 Instalace knihoven

Pro instalaci knihoven GDAL a rasterio:

```
$ dir .\dependencies-binary
$ pip install GDAL-1.11.2-cp27-none-win32.whl
$ pip install rasterio-0.24.0-cp27-none-win32.whl
```

Pro instalaci modulu utm:

```
$ pip install utm
```

B.2 Formát textového souboru

Textový soubor `.\terrain\osm_latlon_data.txt` vypadá následovně:

```
17710 11135 2 3
49.82 14.57 49.84 14.6,
```

kde hodnoty představují proměnné:

```
x, y, offsetX, offsetY
minlat, minlon, maxlat, maxlon
```

a význam hodnot ve stejném pořadí je:

```
x, y hodnoty určující mapbox dlaždici v zoomu 15, offsetX, offsetY určující počet dlaždic
dolní, levá, horní a pravá hranice původního stahovaného .\osm-raw\map.osm
```

B.3 Postup

1. stáhnout si výřez mapy ve formátu .osm ze stránek `https://www.openstreetmap.org/export`
2. pojmenovat soubor `map.osm` a uložit do složky `.\osm-raw\`
3. zavolat automatický skript `.\scripts\run.bat`
4. zkopírovat osm soubory do složky data `robocopy .\osm-tiled .\CityEngine\City\data`
5. zkopírovat tif soubory do složky maps `robocopy .\terrain\tiff .\CityEngine\City\maps`
6. spustit program CityEngine 2017.1 (tato verze byla testována)
7. otevřít skript `createMap.py`
8. nastavit hodnoty `offsetX` a `offsetY` podle počtu dlaždic (tuto hodnotu lze vyčíst ze souboru `.\terrain\osm_latlon_data.txt`)
9. spustit skript `createMap.py` (pomocí klávesy F9, nebo kliknutím na `Python/run script`)
10. podle počtu generovaných dlaždic se postupně objeví $offsetX * offsetY$ dialogových oken pro import `.osm` map. Je třeba zaškrtnout možnost vybrat všechny vrstvy (Select/deselect all) a dále stisknout tlačítko 'Finish'

B.4 Možné problémy

- V případě, že `.\script\run.bat` neproběhne v pořádku, lze volat skripty ručně a odhalit, kde je problém

  ```
  $ python .\scripts\get_lat_lon.py
  ```

  # vypočítá z exportované mapy hodnoty, které se budou v dalších skriptech používat

  ```
  $ python .\scripts\download-mapbox-tiles.py
  ```

  # stáhne potřebné dlaždice terénu ze služby mapbox

  ```
  $ python .\scripts\download-osm-tiles.py
  ```

  # stáhne stejný počet menších osm map, jako je terénu

  ```
  $ python .\scripts\convert-png-to-csv.py
  ```

  # převede data z rgb do textového formátu

  ```
  $ python .\scripts\convert-csv-to-tif.py
  ```

  # převede data z textového formátu do tifu a přidá geografické údaje

- V případě, že se při spuštění Python skriptu v CityEnginu objeví dialogové okno pro výběr souřadného systému, je třeba do vyhledávacího pole napsat "32633"a zvolit "WGS 1984 UTM Zone 33N"

B.5 CityEngine

Po spuštění programu lze ve složce `.\Tynec\scripts` zavolat skript `main.py`. V konzoli se objeví jméno vrstvy, která má být importovaná a objeví se okénko s importem. Uživatel musí odškrtnout vrstvu `highway`, která je zaškrtnuta automaticky, a zaškrtnout vrstvu, jejíž jméno je v konzoli.

Takto uživatel postupuje, dokud se nestáhnou všechny implementované vrstvy.

Pro rozšíření skriptů a vrstev jsou soubory přehledně roztříděny a návod na rozšíření lze nalézt v kapitole Realizace.

Příloha C

Obsah přiloženého CD

```
D:
|   README.txt
|
+---latex
|   |
|   \---figures
|
+---src
|   +---cityengine-version
|   |   |
|   |   |   README_ce.txt
|   |   |
|   |   +---CityEngine
|   |   +---dependencies-binary
|   |   +---osm-raw
|   |   +---osm-tiled
|   |   +---scripts
|   |   \---terrain
|   |
|   \---opensource-version
|       |
|       |   README_os.txt
|       |
|       +---osm-converted
|       |
|       \---programs
|           +---osm2world
|           +---osmfilter
|           \---osmosis
|
\---text
        DIP_kejvaja1_2018.pdf
```

- README.txt - soubor popisující strukturu a funkce složky
- latex - obsahuje potřebná data pro úpravu LaTeXové šáblony
- src - obsahuje dvě složky se zdrojovými kódy dvou přístupů
- cityengine-version - obsahuje data pro realistický přístup
- opensource-version - obsahuje data pro open source přístup
- text - obsahuje pdf kopii diplomové práce