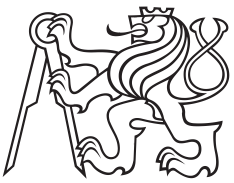
Diplomová práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická
Katedra počítačové grafiky a interakce

Interaktivní prezentace Pražského hradu

Bc. Jan Husák
Studijní program: Otevřená informatika
Studijní obor: Počítačová grafika a interakce

Květen 2017
Vedoucí práce: prof. Ing. Jiří Žára, CSc.

Poděkování / Prohlášení

Chtěl bych poděkovat za vedení práce a rady při vývoji aplikace svému vedoucímu prof. Ing. Jiřímu Žárovi CSc. Dále všem přátelům, kteří se zúčastnili uživatelského testování nebo mi jen poskytli tipy na možné úpravy. A nakonec a především děkuji své rodině za podporu během celého mého studia.

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne 22. 5. 2017

.....

Abstrakt / Abstract

Tato diplomová práce se zabývá vývojem webové aplikace, která zobrazuje Pražský hrad a přilehlé okolí. Uživateli aplikace umožňuje volně se procházet scénou nebo vytvářet virtuální procházku, která uživatele scénou provede. Důležitým aspektem aplikace je grafické zpracování a důvěryhodnost zobrazované scény. Zároveň je nutné aby aplikace fungovala plynule. Tato diplomová práce navazuje na předchozí závěrečnou práci a podrobuje ji hodnocení. Na základě kvalit a nedostatků předchozí práce jsou navrženy úpravy, rozšíření a další nové funkce. Zobrazovaná scéna aplikace je rozšířena o nové objekty. Popsána je implementace nových funkcí. Aplikace je podrobena uživatelskému testování a jsou navrženy možnosti a návrhy dalšího rozšiřování.

Klíčová slova: Pražský hrad, webová vizualizační aplikace, virtuální procházka, WebGL, Three.js

This diploma thesis deals with the development of a web application that visualizes the Prague Castle and the surrounding area. The application lets the user freely browse through the scene or create a virtual walk which leads user through the scene instead. An important part of the application is the graphical quality and credibility of the displayed scene. At the same time, the application must run smoothly. This diploma thesis builds on the previous one which is evaluated. Based on the qualities and shortcomings of the previous application, modifications and other new features are proposed. The scene of the application is expanded with new objects. Described is the implementation of new features. The application is subjected to user testing, and possibilities for further expansion are suggested.

Keywords: Prague Castle, web-based visualization, virtual walk, WebGL, Three.js

Title translation: Interactive presentation of Prague Castle

Obsah /

1 Úvod 1
2 Předchozí práce - shrnutí 2
2.1 Použité technologie a knihovny .. 2
2.1.1 JQuery 2
2.1.2 Three.js 5
2.1.3 RStats 5
2.1.4 Stats 5
2.1.5 Dat.gui 6
2.2 3D editor 6
2.2.1 SketchUp 6
2.2.2 Blender 3D 6
2.2.3 Placené aplikace 7
2.2.4 Závěr 7
2.3 Uživatelské rozhraní a funkce ... 7
2.3.1 Vizuální efekty 7
3 Srovnání s jinými aplikacemi 10
3.1 Mapy.cz 10
3.2 Melown 10
3.3 Google Earth 12
3.4 VBS Blue 12
3.5 Cesium 14
4 Návrh rozšíření předchozí aplikace 15
4.1 Způsob a smysl použití 15
4.1.1 Proč modelovat památky? 15
4.2 Technologické úpravy 16
4.2.1 Úprava načítací obrazovky 16
4.2.2 Rozdělení na klientskou a serverovou část ... 16
4.2.3 Aktualizace Three.js 17
4.2.4 Detekce kolizí 17
4.2.5 WebGL 2.0 17
4.3 Datové úpravy 17
4.3.1 Doplnění modelu 17
4.3.2 Informační podklady 18
4.3.3 Doplnění jmen oblastí 18
4.4 Nové uživatelské funkce 19
4.4.1 Virtuální procházka 19
4.4.2 Zobrazení ostatních virtuálních návštěvníků .. 20
4.4.3 Zobrazení na celou obrazovku 20
4.5 Návrh uživateslkého rozhraní .. 20
5 Model Pražského hradu 23
5.1 Areál Pražského hradu 23
5.2 Podklady 23
5.2.1 Institut plánování a rozvoje hl. m. Prahy ... 23
5.2.2 Mapy.cz 24
5.2.3 Google mapy 25
5.2.4 Open Street Map 25
5.2.5 Papírový model pražského hradu 25
5.3 Model 26
5.3.1 Geometrie 26
5.3.2 Texturování 27
5.3.3 Struktura scény v Blender 3D 29
5.4 Export modelu 30
6 Implementace 32
6.1 Formát tras virtuální procházky 32
6.1.1 Trasa virtuální procházky 33
6.1.2 Picking 33
6.1.3 Webové úpravy 33
6.1.4 Načítací obrazovka 34
6.1.5 Favicon 34
6.1.6 Minimalizace 34
6.1.7 HTTPS 35
6.1.8 Open Graph protokol 36
6.2 Uživatelské rozhraní 37
6.2.1 Celoobrazový režim 37
6.2.2 Animace otevírání oken .. 37
6.2.3 Zobrazení ostatních virtuálních návštěvníků .. 38
6.3 Osvětlení 38
6.4 Grafické efekty 39
6.4.1 Animace vody 39
6.4.2 Normal a bump mapping 40
6.4.3 Úrovně detailu 40
7 Testování 42
7.1 HW a SW nároky 42
7.2 Uživatelské testy 42
7.2.1 Testovací scénáře 43
7.2.2 Subjektivní dotazy 44
7.2.3 Uživatelé 45
7.2.4 Shrnutí testování 49

8 Možnosti budoucího rozšiřování 50
8.0.1 Unity 50
8.0.2 Electron 50
8.0.3 Spolupráce s fakultou stavební při ČVUT 50
9 Závěr 52
Literatura 53
A Uživatelská příručka 59
A.1 Odkaz na aplikaci 59
A.2 Seznam uživatelských funkcí .. 59
A.3 Klávesové zkratky 59
B Příručka pro vývojáře 61
B.1 Vygenerované pohledy 61
B.2 Ovládání 61
B.3 Mapa a minimapa 61
B.4 Textové stránky 61
B.5 Obsah oken 61
B.6 Doplnění názvů 62
B.7 Doplnění textur a materiálů ... 62
B.8 Generování výškové mapy 62
B.9 Praktické zkušenosti z vývoje . 62
C Obrázky z aplikace 64
D Obsah přiloženého CD 69

Tabulky / Obrázky

5.1. Rozdělení vrstev modelu 30
6.1. Porovnání počtu trojúhelníků . 41
7.1. Rychlost vykreslování 42
7.2. Rychlost vykreslování 42

2.1. Porovnání vyhledávání termínů jQuery, Angular a React ... 3
2.2. SSAO 8
2.3. Stíny 8
2.4. Záře světla 9
2.5. Antialiasing 9
3.1. Mapy.cz 11
3.2. Melown 11
3.3. Gogle Earth 13
3.4. VBS 13
3.5. Cesium 14
4.1. Oblasti scény 18
4.2. Virtuální procházka 19
4.3. Hlavní obrazovka uživatelského rozhraní 22
4.4. Okno uživatelského rozhraní ... 22
5.1. Pražský hrad 23
5.2. Mapa z IPR 24
5.3. Model z Mapy.cz 25
5.4. Model z Open Street Map 26
5.5. Papírový model Pražského hradu 26
5.6. Porovnání geometrie 27
5.7. UV mapa 28
5.8. Šum normál 28
5.9. Konverze normál 29
5.10. Nástroj úrovní 29
5.11. Nastavení exportu 31
6.1. Geometrie cest virtuální procházky 32
6.2. Načítací orbazovka 34
6.3. Přenost po síti 36
6.4. Sdílení aplikace 37
6.5. Nastavení času 40
8.1. Statistika využití enginů 51

Kapitola 1
Úvod

Tato závěrečná práce si dává za cíl vytvořit co možná nejvěrohodnější vizualizaci Pražského hradu. Navazuje na předchozí závěrečnou práci[1], z které vychází, snaží se jí rozšířit a upravit. Veškerý obsah je nutno brát s ohledem na předchozí již jednou proběhlou a popsanou rešerši, výzkum i implementaci.

Základním cílem je doplnit nové funkce a zobrazovanou scénu. Bude nutné zhodnotit stav a kvalitu předchozího řešení a upravit ho tak, aby odpovídal novým požadavkům. Aplikace bude upravena technologickými a datovými rozšířeními a novými funkcemi, které jsou popsány dále v samostatných kapitolách. Hlavním funkčním rozšířením bude virtuální procházka, která uživatele provede scénou po vybrané trase.

V úvodní části textu bude zdokumentován areál Pražského Hradu a jeho přilehlé okolí. Tato část bude určující pro rozsah následně vytvářeného 3D modelu scény. V této části bude určeno, kterým objektům se v průběhu 3D modelování scény budu věnovat a také jak detailně a jakým způsobem budou části scény zpracovány.

Důležitou částí bude závěrečné testování aplikace uživateli. Zde dojde k zevrubnému zhodnocení kvality aplikace. Posuzovány budou objektivní i subjektivní aspekty aplikace. Toto hodnocení bude mimo jiné určovat problematické části aplikace, které bude třeba zohlednit při dalších úpravách a rozšířeních této závěrečné práce.

Kapitola 2
Předchozí práce - shrnutí

V této sekci bude popsán stav předchozí aplikace a možnosti rozšíření. Bude znovu porovnána s ostatními podobnými aplikacemi, které za uplynulou dobu měly možnost značně pokročit. Diskutována bude použitá technologie, vybrané knihovny, kvalita a stav modelů a také použité programy a nástroje pro tvorbu modelu.

2.1 Použité technologie a knihovny

V této kapitole je provedeno hodnocení použitých knihoven a zhodnocena je i jejich nutnost pro fungování aplikace. Následně je rozhodnuto o tom, které knihovny bude nutné a příhodné nadále používat a které by bylo lepší z aplikace odstranit nebo nahradit.

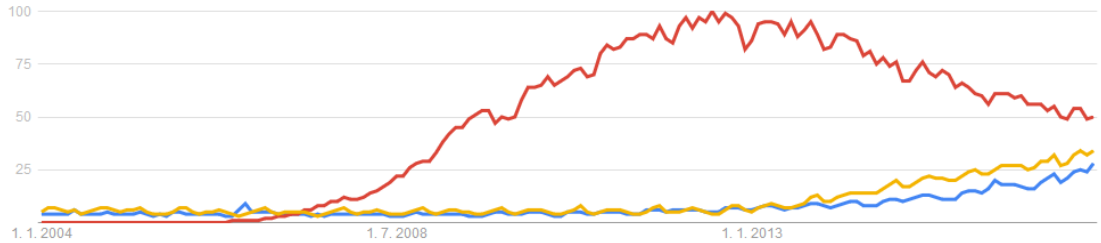
Pro vytvoření aplikace byl zvolen jazyk Javascript s využitím WebGL a knihovny Three.js. Dále bylo použito několik knihoven pro práci s uživatelským rozhraním. v předchozí závěrečné práci proběhla diskuze nad zvoleným jazykem, knihovnou Three.js i technologií WebGL, ale nijak nebyly probrány klady a zápory ostatních použitých knihoven. Dnes je možné vytvořit aplikaci jakéhokoli rozsahu s využitím základních technologí na straně klienta jako je javascript (ES 5 [2]), HTML 5 [3] a CSS 3 [4]. Samozřejmě knihovny třetích stran urychlují a zjednodušují vývoj, ale často také zavádějí vlastní závislosti a problémy, které pak vývoj naopak zpomalují.

Mou snahou zde bylo co nejvíce knihoven odstranit, protože každá z nich má jiné API a je těžké s nimi všemi pracovat dohromady. Navíc jsou některé značně zastaralé a neodpovídají moderním postupům vývoje webových stránek. Vedle nedostatků používání velkého množství knihoven třetích stran je nutné zmínit pozitivní přínos, který spočívá v dramatickém snížení času nutného k vývoji aplikace. Pokud by se zdálo nemožné vytvořit v potřebném čase náhradu některých knihoven, je nutné je zachovat. Následuje seznam použitých knihoven a diskuze ke každé z nich.

2.1.1 JQuery

Knihovna jQuery [5] se v průběhu času stala synonymem pro javascript v prostředí webu[6]. Knihovna je dodnes aktivně vyvíjena a je to jedna z nejpoužívanějších knihoven na webu vůbec. To má pozitivní i negativnní dopady. Za pozitivní lze určite považovat, že se takovýmto obrovským rozšířením vytvořilo jednotné nebo minimálně jednotnější prostředí, na které bylo možno dále navazovat tvorbou další pluginů. JQuery totiž ve svých začátcích především sjednocovalo rozdílnosti javascript API jednotlivých prohlížečů a nahrazovalo funkcionalitu tam, kde nativní podpora prohlížeče chyběla. Dnes jsou ale rozdíly v podporovaných funkcích javascriptu mezi prohlížeči minimální a liší se především v moderních nebo experimentálních funkcích.

Z posledně zmíněné skutečnosti ale vyplývá zásadní argument proti dalšímu používání a rozšiřování jQuery. Totiž pokud bylo jQuery inspirací pro vývoj moderního javascriptu a tento moderní javascript je dnes dostupný[7] a široce rozšířený[8], jaký je důvod nadále jQuery používat? Pokud se jedná o sjednocení různorodosti prohlížečů a zprovoznění aplikace pro prohlížeče starší existují dnes tak zvané polyfilly [9], které



Obrázek 2.1. Porovnání vyhledávání termínů jQuery (červeně), Angular (žlutě) a React (modře) ve vyhledávači Google od roku 2004 po současnost.

se věnují přesně tomuto problému. Pozvolný ústup od používání jQuery ukazuje graf vyhledávání z Google Trends [10] na obrázku 2.1.

JQuery nahrazuje vlastním způsobem práci se strukturou dokumentu. Je to zároveň jeden z největších kladů knihovny i zásadní problém. Javascript ve webovém prostředí nedělá téměř nic jiného, než že hledá a upravuje elementy na stránce a reaguje na vstup uživatele. i uživatelský vstup je navázán na elementy dokumentu. Pokud tedy nahradíme základní funkce pro práci s elementy dokumentu, což jQuery dělá, stane se, že se nějaká funkce knihovny jQuery bude používat téměř ve všech skriptech aplikace a to na mnoha různých místech. s trochou nadsázky se dá říct, že jQuery se tímto způsobem dostane na každý řádek naší aplikace. Tímto způsobem ovlivňuje jQuery podobu celého kódu a při pozdějším odstranění by bylo nutné celý kód přepsat od začátku. i když tento fakt hovoří spíše pro odstranění této knihovny, aplikace už je teď v situaci, kdy tuto knihovnu naopak není možné odstranit, protože by to znamenalo, přepsání celé aplikace a na to není dostatek času.

Problém je o to větší, že jQuery podporuje imperativní programování, které v praxi vypadá tak, že se jednotlivým elementům přiřazuje akce reagující na uživateslký vstup, ale nijak už není ošetřena propagace této akce na ostatní elementy a celkový stav aplikace. Každá reakce tak sama přiřazuje ostatním elementům změnu stavu a brzy tak začne docházek k jejich překrývání a kolizím. Tím se jQuery liší od moderních knihoven jako jsou AngularJS[11], React[12] nebo Vue.js[13], které jako základní stavební kámen používají návrhový vzor observer, díky kterému dokáží lépe propagovat změny stavu aplikace.

Kvůli množství nutných změn, které by nastaly při odstranění jQuery, jsem se rozhodl jQuery v aplikaci zachovat, ale alespoň omezit množství navázaných a často jednoduchých pluginů. Každý zvlášť je popsán dále.

jQuery UI - JQuery UI[14] je doplněk knihovny jQuery, který umožňuje vytvářet složitá uživatelská rozhraní. Knihovna je stejně velké jako samotné jQuery a její použití v aplikaci je těžko obhajitelné. Většina funkcí aplikace provádí akorát reakce na kliknutí uživatele na konkrétní element, což lze udělat s čistým javascriptem nebo se samotným jQuery.

JQuery UI je v aplikaci použito zejména na zobrazení oken. Touto knihovnou je nadefinováno chování oken a také jejich struktura. Obsah některých oken navíc není dostupný v žádném HTML souboru, ale je dynamicky generovaný javascriptem a touto knihovnou. Dále jsou knihovnou zobrazena tlačítka na hlavní obrazovce, která ale mají jednoduché chování, které by bylo snadné po odstranění knihovny nahradit.

Knihovna je použitá na mnoha místech kódu a vedle jQuery je základním kamenem uživatelského rozhraní. Přitom většinu funkcí funkcí knihovny by bylo možné nahradit statickými HTML soubory s využitím moderního CSS. Knihovna navíc styly jednotli-

vých elementů zapisuje do atributu style ke každému elementu, který upravuje. Jelikož je to javascriptová knihovna, tak ani nemá jinou možnost. Pokud ale bude potřeba změnit styl uživatelského rozhraní přes CSS, nezbyde jiná možnost než přebít tyto styly direktivou !important. Tím se ale nedostatky skriptu projevují v CSS a dále tím komplikují vývoj aplikace.

Navrhoval bych jQuery UI odstranit, ale její zásah v aplikaci je na to příliš velký. Předpokládané úpravy uživatelského rozhraní nejsou tak velké, aby se vyplatilo strávit čas nahrazováním knihovny, když to pak uživatel ani nepozná. Velký problém bude vytvořit nový jednotný vzhled celé aplikace kvůli atributům style na jednotlivých elementech, ale to má své řešení. Nakonec jsem se tedy rozhodl jQuery UI v aplikaci ponechat a nadále s ním pracovat. Knihovna je aktivně vyvíjena a dá se očekávat, že v případě problémů, bude stačit aktualizovat knihovnu na novější verzi.

jQuery SVG - JQuery SVG[15] je doplněk knihovny jQuery pro práci s SVG soubory. Jedná se o malou knihovnu (17kb po minimalizaci) a dělá jednu konkrétní věc. Je škoda, že knihovna přestala podporovat Internet Explorer verze 9, protože právě podpora starších prohlížečů je to, co se od jQuery a jemu příbuzných knihoven očekává. Nicméně Internet Explorer není v předchozí závěrečné práci zmíněn mezi podporovanými prohlížeči a tedy tento nedostatek zde nevadí. Knihovna nemá příliš velký zásah do kódu, i když je opět použita v několika souborech, a šla by snadno nahradit, kdyby začala způsobovat problémy. Poslední úprava proběhla před třemi lety a dá se očekávat, že se objeví problémy nebo nedostatky, které v té době nemusely být patrné. Knihovnu jsem se rozhodl zachovat.

jQuery event drag - JQuery event drag[16] je knihovna určená pro přetahování elementů po stránce. Standardně se touto knihovnou řeší dynamické řazení seznamů pomocí myši. v aplikaci je knihovna použita na rozhlížení uživetele, které probíhá posunem myši se stisklým levým tlačítkem, což je přesně akce která je knihovnou podporována. Jedná se o malou knihovnu (5kb po minimalizaci) a plní jednu konkrétní fukci. Nebylo by složité ji nahradit, ale zatím to není zapotřebí. Poslední aktualizace knihovny proběhla před pěti lety.

jQuery tooltipster - Knihovna jQuery Tooltipster [17] je v aplikaci použita na zobrazení názvů budov pod kurzorem. Obecně se dá použít na zobrazení popisků elementu nebo části obrazovky ve chvíli, kdy na ni uživatelel najede myší. Knihovna je malá (16kb minimlizovaná) a je aktivně vyvíjena. v aplikaci jsou funkce knihovny použity jen v jednom souboru a šla by lehce odstranit. Většinu funkcí knihovny nevyužíváme a pouze zobrazujeme text vedle kurzoru. Knihovnu jsem se rozhodl v aplikaci zatím ponechat, ale bylo by vhodné, aby se použila na popisky ostatních částí aplikace, jinak je vcelku zbytečná.

jQuery perfect scrollbars - Knihovna Perfect scrollbars[18] je knihovna, která nahrazuje nativní scroll bar prohlížeče za vlastní. Umožňuje tak vytvořit jednotný vzhled scroll baru mezi všemi podporovanými prohlížeči. v aplikaci je knihovna použita pouze v jQuery UI oknech a je poměrně velká (30kb minimalizovaná). Knihovnu jsem se rozhodl odstranit a nadále využívat nativní scroll bar prohlížeče.

icheck.js - JQuery icheck[19] je knihovna, která zjednodušuje zobrazení vlastních checkboxů a radio buttonů namísto nativních vykreslených prohlížečem. Je malá (2kb minimalizovaná), ale už tři roky není vyvíjena. Navíc nahrazení nativních checkboxů a radio buttonů lze udělat přes CSS pomocí selektoru :checked[20]. Knihovnu jsem se rozhodl odstranit, protože nepřináší nic, co by nebylo možné vytvořit pár řádky CSS.

jQuery spin - Knihovna JQuery spin[21] je malá knihovna (2kb minimalizovaná) na vytváření načítací animace. v aplikaci je použita pouze na načítací obrazovce. Efekt na-

čítání byl nahrazen jiným jak bude popsáno v kapitole 4.2.2, který nevyžaduje stahování žádných skriptů. Knihovnu jsem se jako nepoužívanou rozhodl odstranit.

2.1.2 Three.js

Knihovna Three.js[22] je nadstavba nad WebGL. Klade si za cíl, aby práce s ní byla jednoduchá a přímočará. v ideálním případě se procesy zpracování a vykreslování scény změní na export scény z 3D modeláře a konfiguraci vykreslování na straně Three.js. Snaží se schovat před programátorem příkazy WebGL, ale pokročilejším umožňuje pracovat s shadery, které chápe jako materiál objektů.

Ve srovnání s obdobnými knihovnami je rozhodně daleko napřed. Za jejím vývojem stojí přes 18.000 commitů na git repozitáři s 800 přispěvateli[22] a přibližně 10.000 tagů na stackoverflow [23].

Nicméně otázka jejího použití se odvíjí od potřeb zadání aplikace a zde je několik problémů. Zadání aplikace totiž prozatím nezmiňuje žádné pokročilé techniky renderování ani dynamických změn scény nebo animace. Scéna je statická struktura modelů. Není ani potřeba si udržovat graf scény. Každý objekt může být samostatný s geometrií určenou vůči počátku. Takový způsob vykrelosvání je relativně jednoduchý a není na něj třeba používat velikou víceúčelovou knihovnu, kterou Three.js je.

Předchozí práce popisuje problémy s načítáním Colladda formátu do Three.js. Modul zajišťující nahrávání nepodporuje všechna potřebná data. Nedostatek je řešen dalším zpracováním dat na straně aplikace. Tento problém by se dal lépe řešit úpravou modulu načítání formátu Collada nebo použitím vlastního formátu s tím, že by bylo potřeba pro takový formát vytvořit modul načítání od nuly. Další možností je použít jiný formát podporovaný Three.js, ale už v předchozí práci byl zvolen formát Collada jako nejvhodnější.

Velkým nedostatkem knihovny Three.js je její velikost 814kB. Toto je především důležité ve chvíli, kdy se knihovna přenáší po síti ke klientovi. Knihovna má tak zcela zásadní vliv na množství javascriptu v aplikaci. Nicméně je nutné dodat, že model Pražského hradu má přibližně 10MB (2.31MB po gzip kompresi) a tím poněkud zastírá nedostatek ve velikosti této knihovny. Navíc velikost skriptů lze částečně řešit jejich minimalizací.

I když bych osobně preferoval knihovnu Three.js odstranit, není toto možné, protože stejně jako v případě jQuery stojí na Three.js celá aplikace a odstranění této knihovny by znamenalo její kompletní přepsání.

2.1.3 RStats

RStats[24] je knihovna, která pomáhá debugovat chování WebGL a průběh vykreslování. v aplikaci nebyla knihovna dosud použita, i když byla k aplikaci přiložena a zřejmě bylo její použití zvažováno. Na debug WebGL mají navíc prohlížeče vlastní nástroje, které neovlivňují kód aplikace. Jako nepoužívanou jsem se rozhodl knihovnu odstranit.

2.1.4 Stats

Stats[25] je knihovna na zobrazení FPS běžící aplikace. Umožňuje přepínání mezi zobrazením počtu snímků za sekundu a počtem milisekund na jeden snímek. Knihovna má v minimalizované verzi 2kb a má minimální zásah do kódu aplikace. Mimochodem je od stejného autora jako knihovna Three.js. Rozhodl jsem se knihovnu v aplikaci ponechat, protože plní dobře svou funkci.

2.1.5 Dat.gui

Knihovna dat.gui [26] Umožňuje rychle vytvořit jednoduché GUI, ve kterém se dá interaktivně nastavovat hodnoty navázaných parametrů. Knihovna je ideální pro interaktivní hledání konstant během vývoje. Zjednodušuje práci během vývoje a pro produkční verzi se dá odebrat. Rozhodl jsem se tuto knihovnu odebrat, jelikož konstanty pro funkce osvětlení už byly pomocí knihovny nalezeny v předchozí práci a knihovna tak splnila svůj účel.

2.2 3D editor

V předchozí závěrečné práci byl použit 3D editor SketchUp 2014[27]. 3D editor by v ideálním případě měl být zcela nezávislý na vytvářené aplikaci, ale pouze s kvalitním 3D editorem je tohoto ideálu možné dosáhnout. v opačném případě je nutné nedostatky 3D editoru dohánět na straně aplikace, což ji ale zbytečně komplikuje. Tento posun nároků na řešení problémů v aplikaci namísto 3D editoru byl v předchozí práci popsán (renderování výškové mapy v aplikaci) a bylo zmíněno i několik problémů (obtížná práce se skupinami geometrie a nedostatky exportu pro formát Collada). Jelikož nebyl SketchUp 2014 jako použitý 3D editor podrobněji srovnán s ostatními možnostmi, provedeme tuto rešerši nyní.

Důležité parametry pro nás jsou především kvalita modelářských nástrojů, kvalita práce s materiály, scriptování a možnosti exportu, možnosti nastavení vlastních atributů k objektům scény a také cena. Naopak nepodstatné jsou v tomto případě kvalita a rychlost renderu, protože budeme používat vlastní. Nepodstatná je také kvalita nástrojů na animaci, protože se momentálně s žádnou nepočítá. Preferovány budou volně dostupné programy, které ale alespoň částečně splní kladené nároky.

2.2.1 SketchUp

SketchUp[27] byl použit v předchozí práci. Je to jednoduchý 3D modelář, který byl původně vytvořen společností @Last Software. v roce 2006 koupil celou společnost Google[28] a ten uvolnil verzi programu, která je zadarmo. Poté byl SketchUp využíván jako program, ve kterém nadšenci z komunity vytvářeli 3D modely pro Google Earth[29]. Nesourodost těchto modelů byla diskutována v předchozí práci.

Bohužel SketchUp nedodržuje principy vytváření kvalitní geometrie. Ta se různě překrývá nebo naopak nenavazuje. Neumí ani zobrazit geometrickou síť a nijak uživateli neumožňuje geometrii optimalizovat. Toto je zásadní nedostatek pro použití výsledné geometrie v aplikaci. Navíc program neumožňuje scriptování ani přidávání vlastních atributů k objektům scény. Tento nedostatek byl zmíněn už v předchozí závěrečné práci.

2.2.2 Blender 3D

Blender 3D [30] je volně dostupný 3D modelář, který si klade za cíl, umožnit v jednom programu vytvořit digitální video od modelování přes texturování a animace až po render a post-process. Ne všechny tyto funkce jsou stejně kvalitní a například na texturování dnes existují specializované programy, které Blender 3D dalece přesahují. Například Mari[31] nebo Substance Painter[32].

Kvalita geometrie vytvářené Blenderem 3D je ale nesporná a jednoznačně překonává výstup ze zmíněného modeláře SketchUp. Celkově je jeho výstup srovnatelný s výstupem profesionálních a placených programů. Jeho kvalita je ověřena vlastní filmovou produkcí [33][34][35].

Program podporuje scriptování v jazyce Python. Obsahuje textový editor vhodný pro skriptování a také úpravu textových informací, které lze později exportovat dohromady se scénou. Blender 3D také podporuje množství formátů pro export a umožňuje tak, přenést model do jiného programu nebo aplikace, pokud by to bylo později nutné.

2.2.3 Placené aplikace

Z velkých jmen profesionálních a často velmi drahých programů je třeba zmínit především tři. Je to Maya [36] a 3ds Max [37] od firmy Autodesk a Cinema 4D [38] od firmy MAXON Computer. Všechny tři by šlo bez problémů použít a zcela jistě dostačují nárokům kladeným na kvalitu výstupní geometrie a modelářské nástroje.

Nicméně fakt, že jsou tyto nástroje placené a Blender 3D se jim vyrovná, mi umožňuje rozhodnout se pro použití právě Bledneru 3D.

2.2.4 Závěr

Rozhodl jsem se použít pro vytváření modelů scény Blender 3D. Zásadní pro mě byla kvalita modelářských nástrojů, cena a také částečně osobní preference. Možnosti exportu jsou dostačující, aby mohl být model přenesen do jakéhokoli jiného kvalitního 3D modeláře.

2.3 Uživatelské rozhraní a funkce

Většina uživatelského rozhraní je postavena nad knihovnami jQuery a jQuery UI. Jejich klady a zápory byly popsány výše v kapitole 2.1.1.

Kvalita zpracování uživatelského rozhraní je subjektivní otázka a pro každého uživatele to může znamenta něco jiného. Jakékoli změny uživatelského rozhraní bude nutné reflektovat v uživatelském testování a bude nutné potvrdit nebo vyvráti, že změny znamenaly zlepšení.

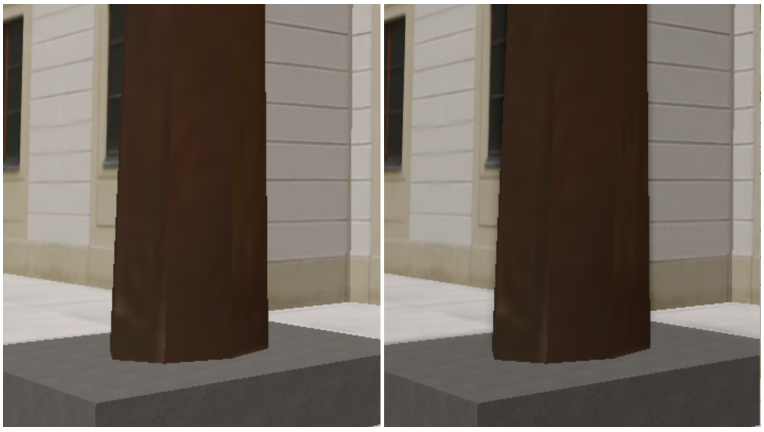
Uživatelské rozhraní se vejde na obrazovku všech podporovaných zařízení. Pokud by se měla aplikace rozšířit o podporu mobilů a menších tabletů, bylo by potřeba přidat responzivní varianty uživatelského rozhraní, ale tato zařízení nejsou momentálně podporován z důvodu nízkého výkonu. Nemá tedy cenu pro takováto zařízení přizpůsobovat uživatelské rozhraní, když by aplikace nezobrazovala ani jeden snímek za sekundu.

Osobně bych preferoval, aby uživatelské rozhraní zabíralo menší část obrazovky a ukazovaly se jen ty prvky, které jsou v danou chvíli důležité nebo zrovna aktivní. Když je jádrem aplikace zobrazování 3D scény Pražského hradu, je myslím dobré, aby tato vizualizace zabírala co největší část obrazovky a nebyla příliš překryta uživatelským rozhraním. Konkrétní změny a nový jednotný styl uživatelkého rozhraní popisuje kapitola 4.4.1.

2.3.1 Vizuální efekty

Tato sekce obsahuje seznam vizuálních efektů, které jsou použity při zobrazování scény. Popsány jsou jen stručně. Detailněji již byly popsány v předchozí závěrečné práci. Na přiložených obrázcích je zobrazen vliv jednotlivých efektů na vykreslovanou scénu.

Bump a normal mapping jsou techniky, která simulují hrbolatost nebo nerovnost povrchu pomocí změn normál na povrchu objektu[39]. v předchozí závěrečné práci sice byly obě tyto techniky připraveny, ale pro žádný materiál nakonec nebyla vytvořena příslušná textura a tak nebylo ani možné získat obrázek z aplikace s těmito efekty. Zároveň byla v poslední verzi aplikace v nastavení schována nabídka na zobrazení normálového nebo bump mapování.



Obrázek 2.2. SSAO efekt z předchozí závěrečné práce na přiblíženém výřezu scény. Vlevo je efekt vypnutý a vpravo zapnutý.

Screen Space Ambient Occlusion simuluje stín v rozích modelů a tím dodává scéně na plastičnosti[40]. Původně se efekt také nazýval **dirty map**, protože připomíná v rozích usazenou špínu. Efekt je vidět na obrázku 2.2.

Stíny jsou promítány pouze hlavním světlem v průběhu dne a jsou vykreslovány metodou shadow mapping s PCF (precentage close filtering)[41]. Stíny jsou vidět na obrázku 2.3.

Obrázek 2.3. Stíny hlavního světla z předchozí závěrečné práce.

Záře světla se vykresluje při pohledu kamery do hlavního světla[42]. Efekt je vidět na obrázku 2.4.

Antialiasing využívá implementaci FXAA pro Three.js[43]. Efekt je vidět na obrázku 2.5

Obrázek 2.4. Záře při pohledu do hlavního světla.

Obrázek 2.5. Antialiasing z předchozí závěrečné práce na přiblíženém výřezu scény. Vlevo je efekt vypnutý a vpravo zapnutý.

Kapitola 3
Srovnání s jinými aplikacemi

Důležitým aspektem aplikace je její srovnání s jinými podobnými aplikacemi. Proto jsem se rozhodl zopakovat část rešerše z předchozí práce a znovu srovnat grafickou a funkční kvalitu konkurenčních aplikací.

Hned z kraje je nutné přiznat, že aplikace ve webovém prostředí postavená nad WebGL nemá šanci dosáhnout kvalit aplikací pro stolní počítače a notebooky a těžko dosahuje i kvalit nativních aplikací pro mobilní zařízení. i když se blíží oficiální vydání druhé verze WebGL[44], je stále třeba chápat tuto technologii jako experimentální a málo podporovanou ze strany webových prohlížečů. Důležité je proto zejména srovnání s jinými webovými aplikace a až druhotné je srovnání s desktopovými aplikacemi.

3.1 Mapy.cz

Mapy.cz[45] je česká firma, která se soustředí na tvorbu digitální map České republiky. Tak to bylo napsáno i v předchozí závěrečné práci, ale nyní už Mapy.cz pokrývají celý svět. Samozřejmě různé části různě detailně. Rozsah pokrytí pro nás není podstatný. Důležitá je hlavně oblast Pražského hradu a způsob, jakým mají zpracováno uživatelské rozhraní.

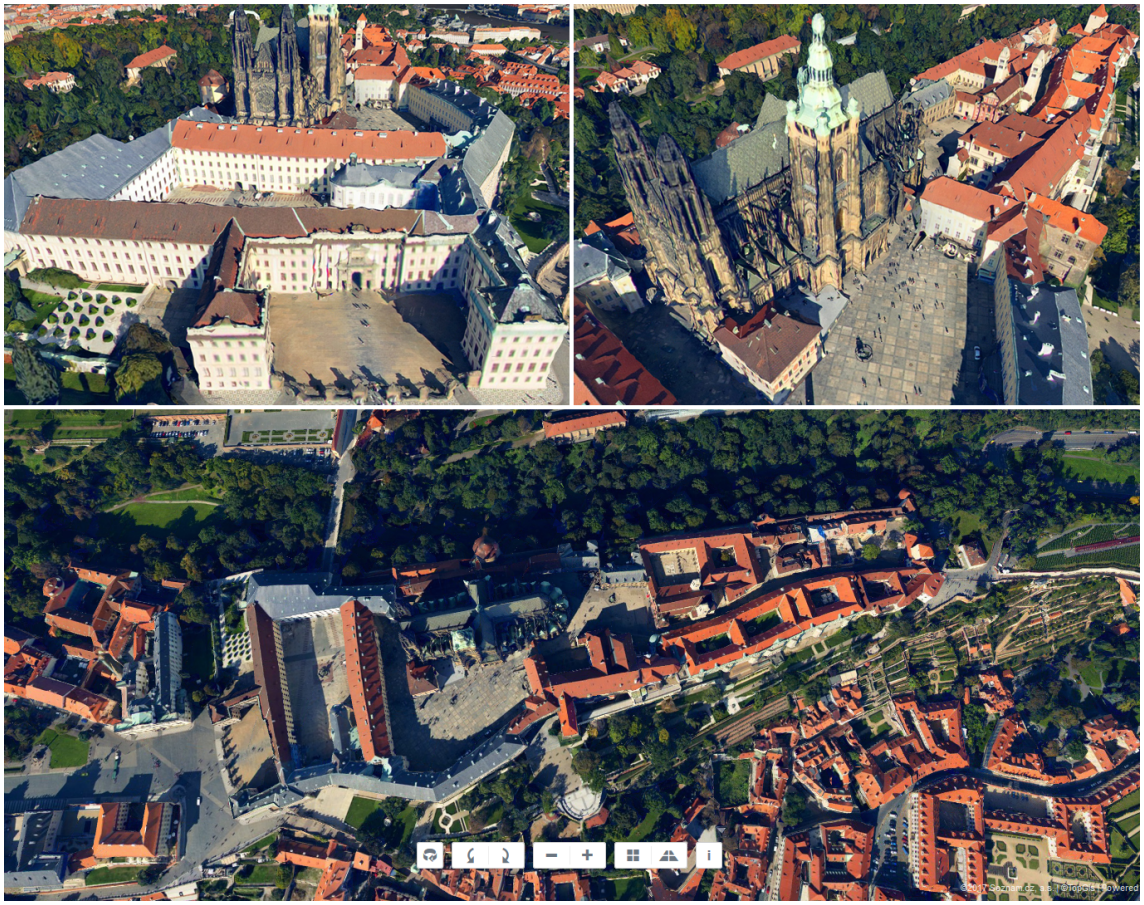
Struktura uživatelského rozhraní zůstala od předchozí práce zachována. Byl ale změněn vzhled tlačítek. Kulatá tlačítka s ikonami se změnila na hranatá se zaoblenými rohy a tlačítka se spojila do skupin, které navazují bez oblých rohů 3.1. Před tím any nyní neměla tlačítka žádné popisky a to ani po najetí myši. Autoři se zřejmě spoléhají na dostatečně jednoznačné ikony. Uživatelské rozhraní obsahuje tlačítko nápovědy, které je v nové verzi stejně velké jako ostatní tlačítka. Nápověda obsahuje pouze informace o ovládání myší a popis funkcí jednotlivých tlačítek nikde není.

Pohyb scénou je volný, ale není možné se scénou procházet z pohledu od země. Na procházení mapy po zemi mají Mapy.cz samostatný mód panorama, který ale nezobrazuje 3D model, ale pouze panoramatické fotografie. v tomto módu se uživatel může pohybovat pouze skokově. Navíc mód panorama nemá zmapován areál Pražského hradu. i když je pohyb v 3D modelu volný, tak přiblížení kamery je skokové a omezené, aby se nedostalo příliš blízko k modelu.

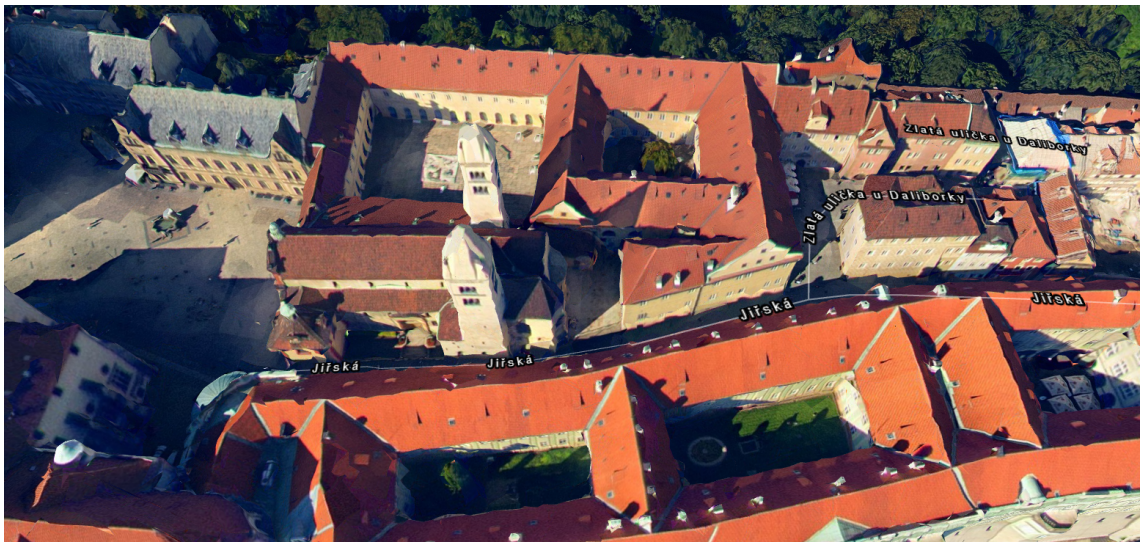
Kvalita modelu vypadá obdobně jako v předchozí rešerši. Změnily se sice použité fotografie, ale celková kvalita geometrie a textur zůstává stejná. Nové mapové podklady pro 3D mapy připravuje firma Melown.

3.2 Melown

Melown[46] je sesterská firma Seznamu, která připravuje open source řešení pro zpracování trojrozměrných map v globálním rozsahu. Jejich engine dokonce již dnes dokáže zpracovat i povrchy jiných planet, pokud jsou pro ně podklady. Výstup tohoto zpracování se mimo jiné použije jako 3D mapa pro Mapy.cz, které jsou také vlastněné Seznamem.



Obrázek 3.1. Pohled na Pražský hrad, jak ho zachycuje funkce 3D pohledu na Mapy.cz. Spodní obrázek obsahuje minimalistické uživatelské rozhraní.



Obrázek 3.2. Pohled na Baziliku sv. Jiří a přilehlý klášter na Pražském hradě z aplikace firmy Melown.

Samotný výstup je dnes stejný jako na Mapách.cz, ale jak je vidět na orbázku 3.2, jsou na mapě kromě 3D modelu zobrazeny také názvy ulic. Mapy.cz tuto funkci zatím nemají. Zobrazení názvů ulic ve scéně by mohlo být zajímavé i pro aplikaci této závěrečné práce.

3.3 Google Earth

Google Earth v poslední verzi funguje pouze v prohlížeči Google Chrome. Kvůli optimalizaci vykreslované scény je část aplikace Google Earth zahrnuta přímo v jádru prohlížeče. Tím se ztrácí rozdíl mezi webovými a nativními aplikacemi. Google Earth tak ostatní webové 3D mapy jednoznačně překonává, ale spustit ho lze pouze v jednom prohlížeči.

Při srovnání s ostatními 3D mapami je na první pohled patrná čistota geometrie. s automatickým generováním geometrie z fotografií jsou většinou spojeny deformace geometrie, které zde ale téměř nejsou. Zdi budov jsou čistě rovné a na zem navazují ostrým přechodem, tak jako ve skutečnosti. Skutečně zarážející kvalita je ukázána na nosnících chrámu sv. Víta. Textury sice nejsou příliš kvalitní, ale samotná geometrie vypadá jako by se umělá inteligence googlu učila z příruček gotických staveb. Detail nosníků je na obrázku 3.3. Na limit narazil Google Chrome zřejmě při zpracování hradní brány se sousoším Zápasících titánů, kde jsou viditelné deformace.

Osvětlení scény je zdá se trochu přesvětlené, ale na celkovém dojmu to neubírá. Ve scéně jsou zapečené stíny a pohyblivé objekty, jako jsou auta, se změnily pouze na barevný flek v textuře. Celkově působí scéna až překvapivě kvalitně, ale stále se jedná o statickou scénu ve které není možné měnit směr ani styl osvětlení. Společně s modelem lze zobrazit ve scéně názvy objektů a míst. Tyto popisky jsou navíc interaktivní a po kliknutí zobrazují popis a fotografii místa nebo objektu.

Uživatelské rozhraní je rozděleno na dvě části. Na levé straně obrazovky je lišta nástrojů, která se zleva rozbaluje do širší nabídky a vpravo dole jsou umístěny nástroje pro pohyb ve scéně. Celé uživatelské rozhraní je zpracováno s jednotnou a nevýraznou barvou pozadí. Tlačítka jsou doplněna bílými ikonami. Popisky tlačítek se zobrazují až po najetí myši. Otevřená levá nabídka obsahuje kromě ikon rovnou i popisky jednotlivých tlačítek. Je škoda, že se ve scéně nedá procházet u země. Pro toto procházení je zachován původní mód vytvořený z panoramatických fotografií mezi kterými uživatel přeskakuje. Nicméně dá se očekávat, že i tato funkce bude brzy podporována.

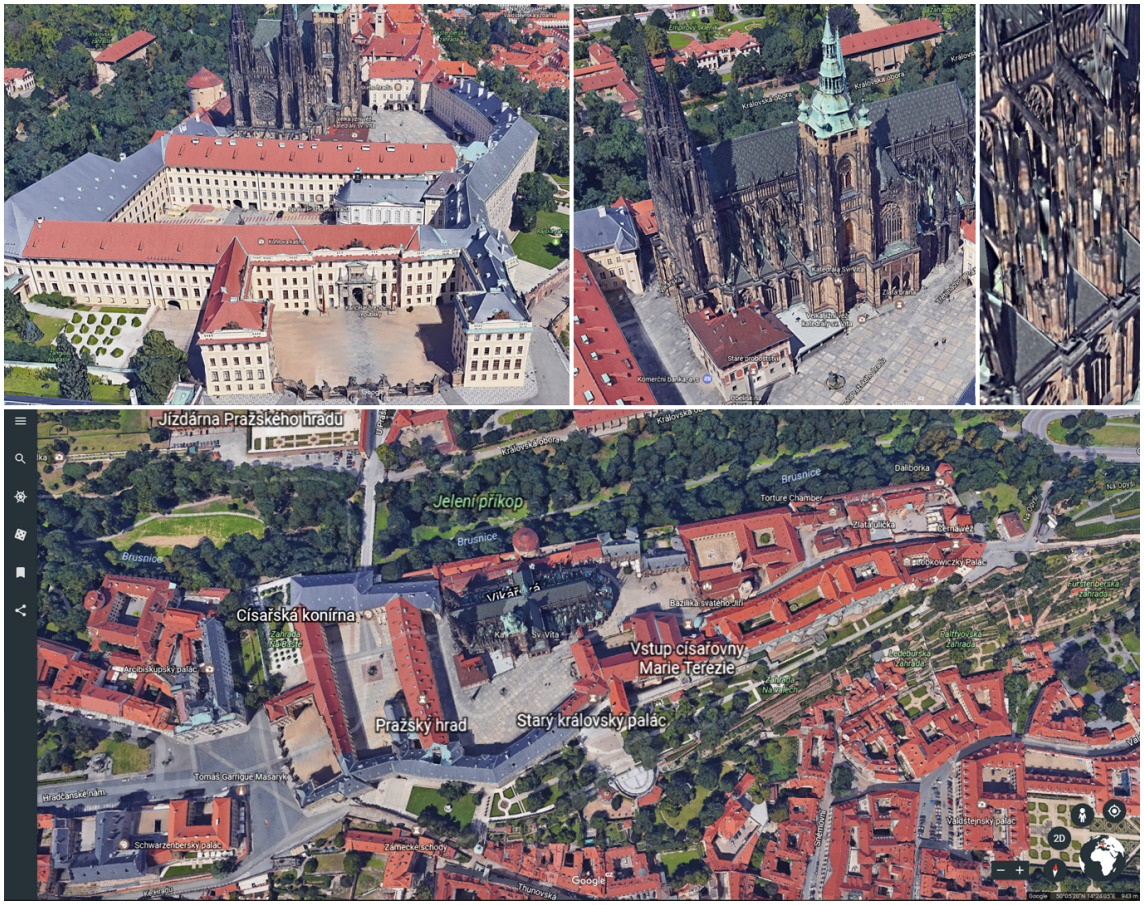
Zajímavou částí Google Earth je také úvodní průvodce, který obsahuje pouze 5 kroků, ale ty uplně stačí, aby uživatel poznal nové funkce a způsob ovládání aplikace. Tímto způsobem je uživateli mimo jiné představena funkce nazvaná `I'm feeling lucky`, která uživatele přenese na náhodné místě na planetě.

3.4 VBS Blue

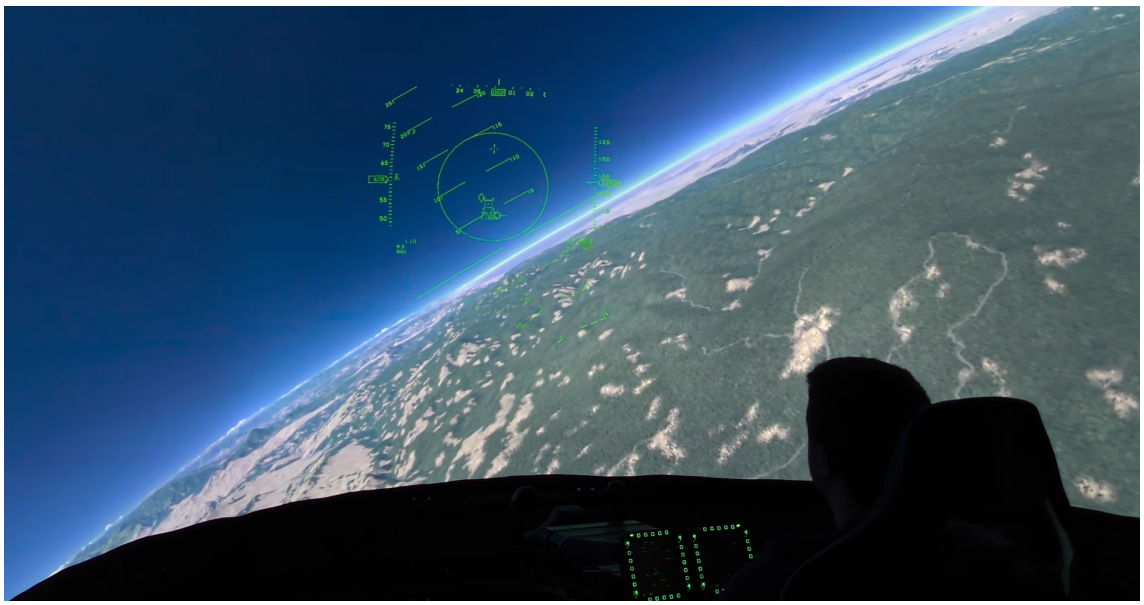
VBS Blue[47] je projekt české firmy Bohemia Interactive Simulations. Ta se specializuje na vojenské simulátory pro armádu. v tomto novém projektu se snaží vytvořit simulaci, která by obsahovala celou planetu.

Při vytváření scény kombinují mapové podklady s procedurálně generovanými částmi. Například výška a tvar terénu vychází z mapových podkladů, ale textury jsou v nízké kvalitě a jsou doplňovány procedurální texturou, která pouze zlepšuje vizuální kvalitu, ale nesouvisí se skutečným vzhledem původní scény.

Pro aplikaci této závěrečné práce by bylo srovnání s VBS zajímavé, protože jako jedna zmála umožňuje volný pohyb hráče scénou a to jak ve vzduchu tak přímo na terénu. Zároveň VBS Blue kombinuje 3D mapy s dalšími objekty, což je této závěrečné práci také bližší než samotné 3D mapy. Bohužel z projektu VBS blue zatím není mnoho známo a zveřejněno je jen několik videí a obrázků.



Obrázek 3.3. Pohled na Pražský Hrad z aplikace Google Earth. Pravý horní obrázek ukazuje detail nosníků katedrály sv. Víta. Spodní obrázek obsahuje uživatelské rozhraní.

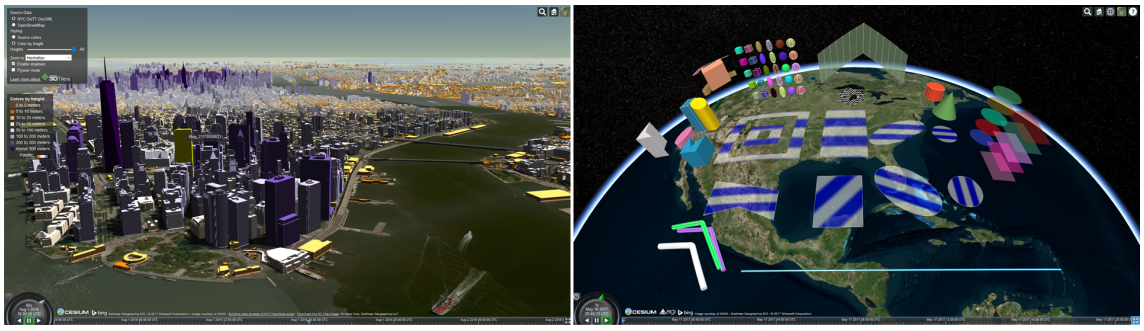


Obrázek 3.4. Scéna zobrazená z VBS Blue v leteckém simulátoru.

3.5 Cesium

Cesium[48] je knihovna pro zobrazovnání map. Zobrazuje celý svět, ale mód s 3D zobrazením umí jen pro pár ukázkových oblastí. Zřejmě zde chybí fotografické podklady. Planeta jde ve scéně zobrazit celá a geometrie obsahuje i různou výšku terénu, ale jednotlivé objekty jako jsou budovy nebo stromy už nejsou pro většinu planety zobrazeny ve 3D. Zmíněna je zde kvůli zajímavému efektu vodní hladiny. Ve scéně jsou totiž automaticky vyhledávány vodní plochy, kterým je přidána animace, což oživuje celou scénu. Ostatní mapy Casium převyšují, ale pořád zobrazují statický obsah bez animace a s minimem interakce.

Cesium podporuje interakci s objekty. Zároveň umí vkládat nové objekty. které respektují povrch terénu. Pro oblasti kde Cesium zná 3D modely budov, jsou dynamicky vykreslovány stíny a osvětlení a i samotné budovy lze obarvovat a dynamicky upravovat3.5.



Obrázek 3.5. Scéna zobrazená v knihovně Cesium. Vlevo je New York se zapnutým vykreslováním stínů a obarvením budov podle jejich výšky. Vpravo je ukázková scéna s vloženými objekty.

Kapitola 4
Návrh rozšíření předchozí aplikace

Tato kapitola popisuje nové uživatelské funkce a odůvodnění jejich přidání na základě předchozí závěrečné práce 2.3.1. Jejich implementace je popsána dále v kapitole 6.1. Kapitola je rozdělená na technologické úpravy, datové úpravy a nové uživatelské funkce.

4.1 Způsob a smysl použití

V této sekci zběžně popíšeme důvody tvorby takovéto aplikace a srovnáme tyto důvody s jinými obory, které se také zobrazováním 3D modelů historických staveb zabývají. Rozdílnost těchto pohledů se promítá i do požadavků na aplikaci. Mění se tím nároky na uživatelské funkce i způsob zobrazení scény.

4.1.1 Proč modelovat památky?

Název sekce je odvozen od názvu konference v Anežském klášteře v Praze[49], která představila posluchačům postup a výsledek rekonstrukce Anežského kláštera. Během rekonstrukce byl mimo jiné vytvořen 3D model celého objektu. Konference se zabývala především tvorbou tohoto virtuálního 3D modelu. Je zajímavé konfrontovat cíle a postupy mé práce s cíli a postupy použitými při vytváření 3D modelu Anežského kláštera, jak je popsali řečníci na této konferenci.

3D model vytvářela v rámci závěrečné práce Fakulta stavební při ČVUT[50]. Jejich cílem bylo zaznamenat objekt v přesně takovém stavu v jakém po rekonstrukci byl. Jako vstupní data pro model využívali měření pomocí geodézie a také stavební plány, které měli k dispozici. Ze zásady se vyhýbali vlastním zásahům do modelu, který by neodpovídal skutečnosti, ať už z důvodu estetických nebo na základě snahy dodělat dnes už neexistující prostory. Během konference byl popsán případ, kdy chtěli zaznamenat starou a dnes už neexistující klenbu. Místo toho aby do modelu zanesli vlastní představu, jak tehdy klenba pravděpodobně vypadala, tak do modelu přidali pouze červený obrys základního tvaru, který naznačoval, že na místě původně klenba byla, ale nenarušoval zobrazení dnešního stavu. Tím věrně zaznamenali skutečný stav objektu, ale zároveň ho doplnili o historickou informaci.

Celý jejich postup hodnotím jako ryze vědecký a naopak svůj v této práci jako spíše umělecký. Zásadním hlediskem mé práce je hlavně vizuální kvalita výsledné práce. Naopak jestli je nějaký bod na centimetr přesně umístěn není v mé práci zcela podstatné. Navíc získat plány Pražského hradu pro mou aplikaci není možné, protože plány celého objektu podléhá státnímu tajemství.

Výsledný model Anežského kláštera bude zobrazen v interaktivní aplikaci v rámci prohlídky Anežského kláštera a je možné ho i zobrazit v jednoduché podobě na webu[51].

Dalším z řečníků byl Knut Paasche z Norského institu NIKU (Norwegian Institute for Cultural Heritage Research)[52]. NIKU digitálně snímá a zpracovává především norské památky, ale v rámci mezinárodní spolupráce i zahraniční. Používají laserové skenery a podobná měřící zařízení.

Zásadním rozdílem oproti mé práci je fakt, že oni pracují s bodovými mračny, které je pro ně dostatečným grafickým výstupem měření. z bodových mračen později poznávají strukturu budovy a její stav. Při více snímání v čase lze i získat porovnání pohybu budovy a předejít tak jejímu případnému pádu. Nicméně až na výjimky nevytváří 3D modely ve smyslu trojúhelníkové sítě, protože to není k jejich vědecké práci potřeba a zabralo by to hodně neopodstatněného času.

Tedy 3D model může mít smysl pro interaktivní procházky a podobné aplikace, ale z hlediska architektů nemá moc velký význam. Jde hlavně o vizuální aspekt problému. s tímto přístupem a pochopením rozdílu mezi obory architektury a grafiky jsem postupoval i při tvorbě modelu Pražského hradu. Ten je detailněji popsán v kapitole 5.1.

4.2 Technologické úpravy

V této sekci jsou popsány technologické úpravy a rozšíření oproti předchozí závěrečné práci. Opraveny byly dílčí části vykreslování, grafické efekty aplikace a byly doplněny moderní nástroje pro jednodušší tvorbu webových aplikací.

4.2.1 Úprava načítací obrazovky

Načítací obrazovka potřebuje nejdříve načíst scripty a navíc používá externí SVG logo. Bylo by lepší, aby celá načítací obrazovka, včetně skriptů, byla vložena do hlavního souboru a mohla se tak zobrazit téměř okamžitě.

Zlepšit se dá i moment skrytí načítací obrazovky. Momentálně zmizí hned jak jsou načteny modely scény. Poté ale aplikace ještě čeká na načtení textur modelů. Načítání by mělo být skryto v momentě, kdy se může celá aplikace kompletně zobrazit.

4.2.2 Rozdělení na klientskou a serverovou část

Momentální řešení je celé postavené pouze na klientské části. Aplikace je díky tomu přenositelná jako složka souborů. Toto je zřejmá výhoda nicméně to neodpovídá standardním webovým aplikacím. Hlavním problémem je, že musíme aplikaci nejdříve stáhnout na svoje zařízení a až teprve poté ji můžeme spustit. Pokud využijeme serverovou část, pak uživateli stačí zadat správné URL naší aplikace a o zbytek už se postará právě server. Dále nám server usnadňuje vývoj aplikace v několika oblastech, které probereme dále.

Přidáním serverové části a nasazení frameworku se velmi zjednoduší vývoj uživatelského rozhraní aplikace. Momentální stav, kdy jednotlivé pohledy jsou v samostatných souborech, je nadále těžko udržitelný. Tyto pohledy jsou neprovázané HTML soubory a pokud bychom chtěli zasáhnout do celkové struktury uživatelského rozhraní, museli bychom tuto změnu zadat do každého souboru zvlášť. Naproti tomu framework a webový server nám umožňuje pohledy dynamicky skládat s minimem úsilí.

Dalším přínosem je možnost nasazení HTTPS a s ním HTTP/2, které umožňuje rychlejší načítání jednotlivých souborů[53]. Oproti starším verzím HTTP totiž umožňuje souběžné načítání více souborů najednou. Navíc způsobem, který šetří i výpočetní nároky serveru. Momentální verze pouze s klientskou částí nemůže tuto funkcionalitu využít, protože prohlížeč otevírá aplikaci na přímo přes cestu k souboru a nevyužívá protokol HTTP ani HTTPS.

Dalším přínosem je zpřístupnění optimalizačních nástrojů jako je webpack[54], gulp[55] a podobné. Tyto nástroje by bylo možné použít i bez serverové části, ale v rámci moderních frameworků je práce s nimi mnohem snažší a není třeba zdlouhavé konfigurace. v případě této aplikace je zlepšení výkonu optimalizačními nástroji o něco

složitější, protože největší zátěží při spouštění aplikace je soubor obsahující geometrii scény, který lze jen velmi složitě optimalizovat.

4.2.3 Aktualizace Three.js

Aktualizace stěžejní knihovny celé aplikace by mohla a měla vyřešit množství bugů, které jsme při vývoji doposud řešili. Nicméně už nasazení o jedna lepší verze způsobuje nefunkčnost aplikace.

Za velmi nepříjemný bug považuji například fakt, že matematická funkce na zjištění vzdálenosti mezi dvěma body ve 3D upravuje vnitřní hodnoty prvního bodu a tím ho posouvá v prostoru.

Nefunkčnost novějších verzí knihovny je dána změnami, které v předchozí diplomové práci byly v jádru knihovny udělány. Knihovnu tedy není možné aktualizovat aniž by byl změněn kód na straně aplikace nebo znovu změněn kód v novějších verzích na straně knihovny. Rozsah těchto změn je pak těžce odhadnutelný a hodnotím ho jako mimo rozsah a zaměření této práce.

4.2.4 Detekce kolizí

Momentálně jsou kolize řešeny prostřednictvím tří heightmap a nad nimi postavenou funkcionalitou. Toto je detailně popsáno v předchozí práci.

Nedostatek vzniká v místech průchodů, které mají být uživateli přístupné. Průchody jsou totiž řešeny pomocí dvou výškových map určujících strop a podlahu průchodu. Pro jeden průchod aplikace funguje, ale pokud by měly být dva nad sebou tak už ne. Takovýto případ může nastat v prostředí Pražského hradu například u Býčího schodiště, které má několik úrovní.

Tento nedostatek je řešitelný implementací ray tracingu a jeho řešením kolizí. Navíc by se zlepšila i přesnost kolizí, protože momentální verze nedokáže rozpoznat malé objekty jako jsou ploty, které se ve výškové mapě kvůli relativně nízkému rozlišení vůbec nezobrazí. Počet paprsků v hodnotě několika desítek je i pro javascript zvládnutelné množství[56], ale celková náročnost takové implementace by vydala za samostatnou závěrečnou práci. Toto zlepšení tedy navrhuji jako možnost dalšímu rozšiřiteli apliakce.

I když je to velmi náročné rozšíření, bylo by dobré ho nakonec dosáhnout, protože žádná jiná podobná aplikace z rešerše toto neumožňuje.

4.2.5 WebGL 2.0

Používaná verze WebGL plně závisí na knihovně Three.js. Jelikož tu není z důvodů změn v kódu možné aktualizovat, lze předpokládat, že i funkce z WebGL verze 2 nebudou v aplikaci dostupné.

Oproti WebGL 1.0 nabízí druhá verze množství rozšíření, které přibližně odpovídají nativnímu OpenGL ES 3.0.

4.3 Datové úpravy

4.3.1 Doplnění modelu

V předchozí závěrečné práci se scéna omezovala na hrubý model areálu Pražského Hradu a byla doplněná podložkou s okolní krajinou, která sloužila jako část skyboxu. Detailně bylo vytvořeno pouze okolí prvního nádvoří a na něm byly prezentovány klady a funkce předchozí aplikace.

V této práci je model doplněn a rozšířen o všechny budovy v rozsahu mapy uvedené dále v kapitole 5.1. Detailněji je přistupováno k budovám areálu Hradu a budovy okolí jsou vytvořeny hlavně na dokreslení prostředí památkové rezervace, ve které se Pražský hrad nachází. Okolní budovy sice nejsou hlavním obsahem vytvářené aplikace, ale celkově přidávájí scéně na kvalitě.

Rozšíření modelu znamená nutnost doplnění textových a mapových podkladů. Tato rozšíření jsou samostatně uvedena v následujících kapitolách.

S přepracováním scény dojde zároveň ke sloučení souborů terrain.dae a scene.dae do jednoho. Původní terrén v souboru (terrain.dae) byl pouze ilustrativní a sloužil spíše jako součást skydomu. Nový terén bude součástí scény a bude tak i modelován.

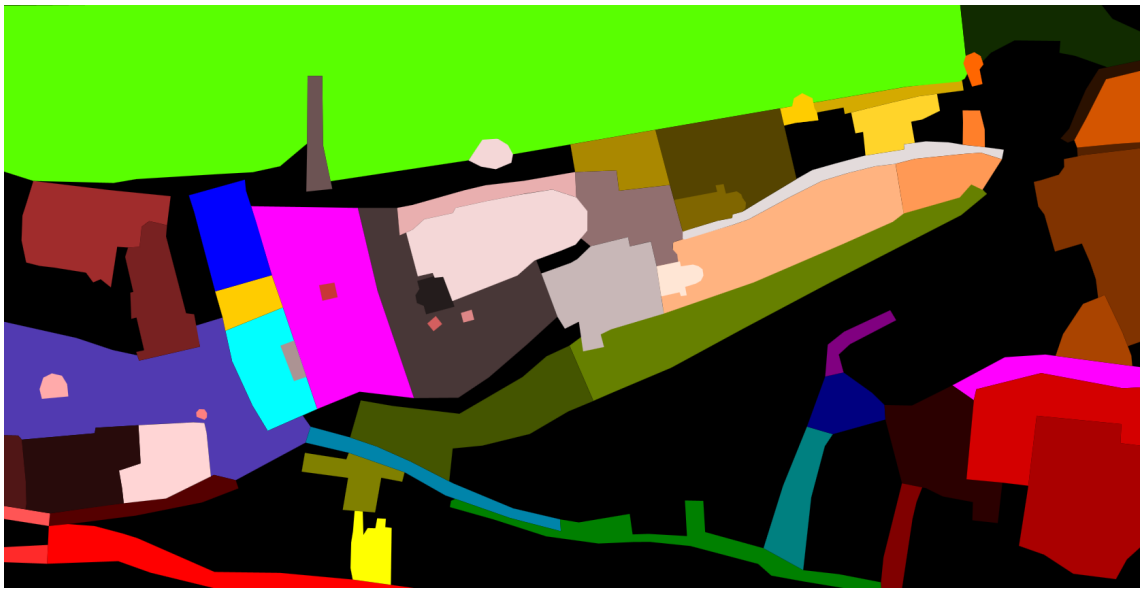
4.3.2 Informační podklady

Jelikož se model scény oproti předchozí závěrečné práci rozšířil, je potřeba přidat i textové informace k novým objektům. Především pak k objektům významným. Ideálním zdrojem pro tyto texty je elektronická encyklopedie Wikipedie. Ze stejného zdroje je možné použít obrázky pro ilustraci jednotlivých objektů. Nedostatkem Wikipedie je málo přeložených článků o budovách Pražského hradu. Objevuje se často francouzština a maďarština ale jen zlomek je přeložen do angličtiny.

Popis objektu je v aplikaci dostupný po kliku na budovu ve scéně. Popis se otevře jako samostatné okno a ukáže se případný ilustrační obrázek, popis objektu a samozřejmě odkaz na zdroje. Aby se popisky vůbec mohly ukázat, je potřeba rozšířit seznam objektů na které lze v aplikaci kliknout.

4.3.3 Doplnění jmen oblastí.

Nad minimapou se ukazuje jméno oblasti, ve které se uživatel ve scéně nachází. Tato jména jsou načítána z vektorového souboru s barevnými n-úhelníky, které přes nastavené id určují názvy oblastí. Výsledné upravené oblasti jsou vidět na obrázku 4.1. Protože se oblast určuje podle barvy, je potřeba, aby každá měla svou barvu unikátní. Tím vzniká omezení na počet oblastí rovné počtu barev osmibitového RGB obrázku. To by ale neměl být problém a při rozsahu scény by toto omezení nemělo být nikdy dosaženo.



Obrázek 4.1. Oblasti scény jsou definovány jako barevné n-úhelníky v SVG souboru.

4.4 Nové uživatelské funkce

4.4.1 Virtuální procházka

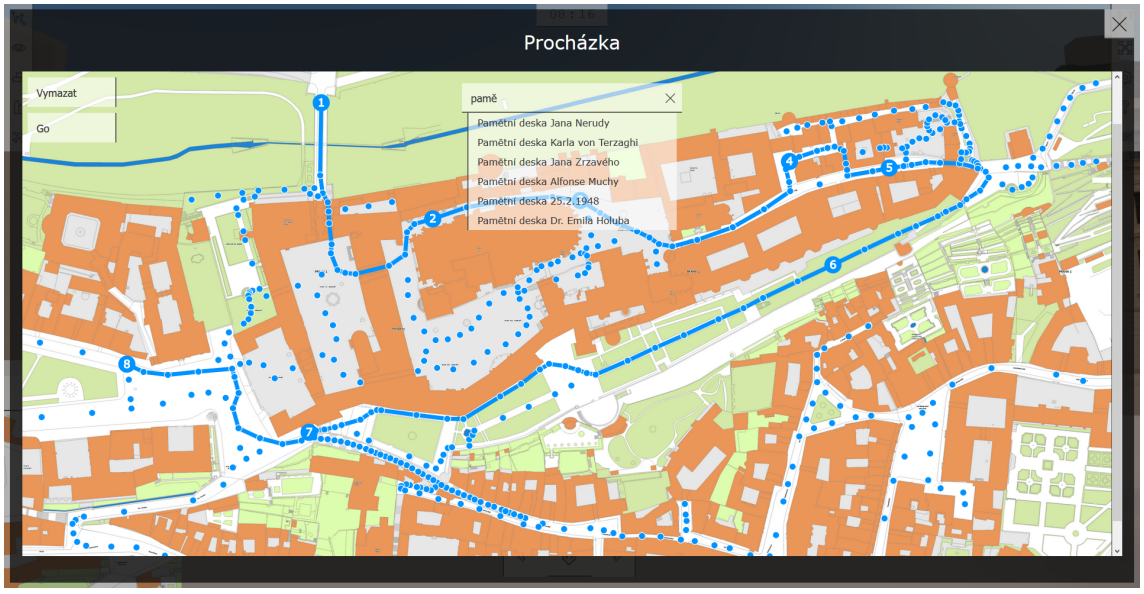
V předchozí aplikaci byla vytvořena možnost volného pohybu uživatele scénou. To je vhodný způsob ovládání pokud, si chce uživatel sám scénu procházet a prohlížet, ale už není zcela vhodný ve chvíli, kdy uživatel scénu nezná, ale stále by rád došel na nějaké konkrétní místo. Tento nedostatek byl i diskutován v předchozí práci a posudcích a tato funkcionalita se ho snaží vyřešit.

Cílem je přidat do uživatelského rozhraní mapu Pražského hradu, ve které si uživatel vybere body, které chce projít a poté ho aplikace sama těmito body provede. Body bude možné také přidávat přes textové vyhledávání na základě turistických informací přidaných k modelu. v průběhu procházky bude možné se volně rozhlížet stejně jako při volném pohybu modelem. Procházku bude možné zastavovat a zase spustit a bude možné ji i přerušit a volně se pohybovat a poté se k ní vrátit.

Na hlavní obrazovku aplikace přibude tlačítko pro otevření nastavení virtuální procházky s mapou Pražského hradu a další dvě tlačítka pro pozastavení procházky a pro pokračování v procházce. Přibude také zobrazení vzdálenosti a času procházky a to ve formě celkové a zbývající hodnoty. Doplňující tlačítka k procházce se zobrazí na hlavní obrazovce aplikace a to pouze tehdy, pokud bude procházka aktivní. v opačném případě budou tlačítka i informace o vzdálenosti schována, aby zbytečně nepřekážela zobrazované scéně.

Samotné okno nastavení virtuální procházky bude vedle zadávání cesty a textového vyhledávání umožňovat zobrazení vybrané cesty. To bude zpracováno formou čar mezi body, kterými vybraná cesta povede. Zároveň bude nutné zvýraznit vybrané body a ideálně je i očíslovat, aby bylo vidět pořadí, ve kterém budou po spuštění procházky navštíveny. Bude zde i možnost procházku vymazat a zadat ji znovu. Zároveň bude možné i odebírat jednotlivé dříve vybrané body z již zadané procházky.

Zajímavou možností by bylo doplnit procházku o průvodce ve smyslu zvukového podkladu, kde by byly popsány zajímavé blízké objekty a podobně. To je ovšem mimo rozsah této práce a nabízí se to jako budoucí rozšíření.



Obrázek 4.2. Virtuální procházka.

4.4.2 Zobrazení ostatních virtuálních návštěvníků

Aplikace bude s podporou knihovny Firebase nabízet jednoduchou interakci mezi uživateli. Každý uživatel který si tuto aplikaci otevře se ostatním uživatelům zobrazí ve scéně jako nějaký jednoduchý objekt.

Výhodou Firebase[57] je, že zajišťuje komunikaci a synchronizaci stavu mezi klienty a serverem aniž by se o to programátor musel starat. Na serveru je umístěna NoSQL databáze do které se v tomto případě ukládají pozice uživatelů. Firebase se typicky používá s nějakou formou přihlášení, aby bylo uživatele možné jednoznačně identifikovat, ale přidávání přihlášení by zde nebylo vhodné a tak bude použit jednoduchý účet, který je jednoznačně identifikován pouze na základně unikátního ID, které je vygenerováno při prvním spojení se serverem.

Do nastavení aplikace by měla přibýt možnost ostatní uživatele nezobrazovat.

4.4.3 Zobrazení na celou obrazovku

Moderní API javascriptu umožňuje webovému prohlížeči nastavit, aby se některý element stránky zobrazil na celou plochu obrazovky. Takovéto zobrazení více připomíná nativní vizualizační aplikace a i pro tuto aplikaci je zajímavé.

Do aplikace tak budou přidány dvě funkce podporující režim zobrazení na celou obrazovku. První bude samotné přepnutí na celou obrazovku a druhou funkcí bude přepínání zobrazení uživatelského rozhraní. Při schování uživatelského rozhraní zůstane viditelné pouze toto přepínací tlačítko, aby bylo možné uživatelské rozhraní opět zobrazit. Obě tyto funkce by měly podpořit pozitivní vnímání zobrazované scény.

4.5 Návrh uživateslkého rozhraní

Tato sekce popisuje změny starých i nových částí uživatelského rozhraní, které vedly k jeho vizuálnímu sjednocení. Je definován vzhled a chování jednotlivých prvků, tak aby se vzájemně nerušily a byly v rámci aplikace vždy rozeznatelné. Důraz byl kladen na zmenšení oblasti, kterou uživatelské rozhraní zabírá na hlavní obrazovce. Některé položky mají navíc doplněnou animaci, která reaguje na uživatele. Animace mimo líbivého efektu umožňuje zobrazení jinak skrytých informací, které byly v předchozí závěrečné práci vypsány přímo na obrazovku. Nyní jsou tyto informace zobrazeny až ve chvíli, kdy je uživatel hledá, což se projeví tím, že najede kurzorem na příslušné tlačítko. Výsledný vzhled uživatelského rozhraní je zobrazen na obrázcích 4.3 a 4.4.

Části uživatelského rozhraní se dělí především na okna a tlačítka. Okna se typicky otevírají po kliku na některé z tlačítek. Tlačítka jsou umístěna na hlavní obrazovce i v jednotlivých oknech. Kromě otevírání oken, mají tlačítka další různé funkce. Zbylými položkami jsou minimapa, čas, posuvníky a aktivní body v mapách.

Na hlavní obrazovce jsou tlačítka rozdělena na levou a pravou stranu. Levá strana obsahuje položky, kterými uživatel interaguje se scénou a pravé strana obsahuje nastavení zobrazení. Dále jsou na dolní střed umístěny šipky pro ovládání pohybu bez klávesnice, v levém dolním rohu je umístěna minimapa se zobrazením aktuální pozice a v horním středu jsou umístěny hodiny a v některých stavech délka nastavené procházky.

Základními grafickými prvky, které jsou použité napříč všemi položkami uživatelského rozhraní, jsou, tmavý obrys o šířce dvou pixelů z jedné strany položky, polo průhledné pozadí, zneprůhlednění pozadí při najetí myší a doplňková animace.

Tlačítka mají bílé polo průhledné pozadí a tmavý text. z jedné strany je zobrazen tmavý obrys. Tento obrys určuje oddělení tlačítka nebo určuje směr nějaké animace nebo akce na kterou je tlačítko navázáno. v případě pohybových tlačítek je rámeček

zobrazen ve směru šipky tlačítka. v případě položek levého a pravého menu je rámeček směřován do středu obrazovky a tím jemně odděluje prostor nabídky od zbytku aplikace.

Tlačítka menu jsou zobrazeny pouze jako ikony, aby nezabírala příliš místa. Při najetí myši se zobrazí i text položky, který jednoznačně popisuje funkci tlačítka, i když by to pouze z ikony nemuselo být zřejmé. Text tlačítka se otevírá do středu aplikace ve směru obrysu a ikona zůstává na místě.

Okna mají tmavé polo průhledné pozadí a bílý text. Polo průhledné pozadí umožňuje vidět změny, které nabídky některých oken ve scéně způsobují. Například zobrazení SSAO.

Všechna okna mají stejnou velikost a jsou zobrazena na celou obrazovku a až uvnitř nich je vytvořen sloupec, který ohraničuje obsah a obsahuje vertikální posuvník, kde je to nutné. Okna obsahují dvě šířky sloupců. Je to plná šířka okna, která je použita pro mapy a výběr pohledů a pak šířka přibližně poloviční, která je použita pro textové stránky a nastavení. Menší šířka pro textové stránky je důležitá zejména pro velké monitory, kde by se text obtížně četl, pokud by byl na celou šířku obrazovky. Na menších zařízeních mohou oba typy sloupců splývat v jednu velikost přes celou šířku okna.

Otevírání oken je doplněno animací, která rychle zvětšuje okno z nuly na celou velikost obrazovky. Tím dostává uživatelské rozhraní dojem hloubky i když je celé vytvořené jen ve dvou rozměrech. Zavírání oken je doprovozeno animací opačnou. Samotné zavírání oken je možné přes klávesu escape nebo kliknutím na tlačítko křížku v pravé horní části obrazovky.

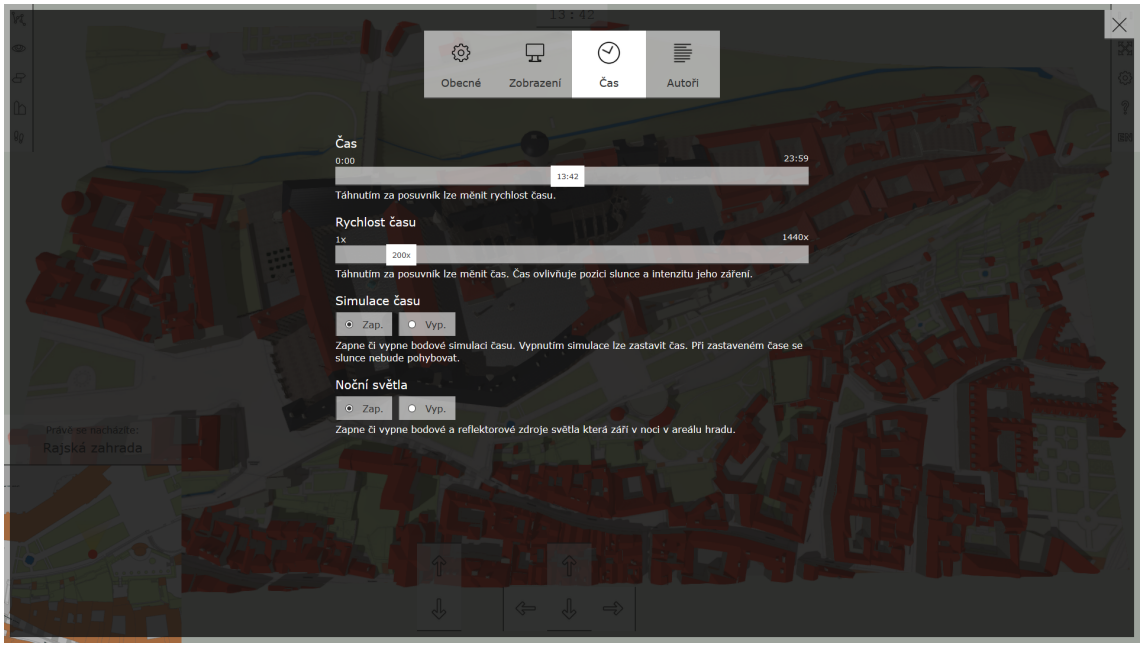
Hodiny jsou zobrazeny jako tlačítko i tak fungují, protože otevírají nastavení času. Tmavé ohraničení je použito směrem do středu obrazovky. Stejně tak je pod hodinami zobrazena délka a čas virtuální procházky.

Speciální případ obsahu oken jsou mapy. Ty jsou dvě. Mapa celého areálu hradu z vyznačenými objekty zájmu. a mapa virtuální procházky. Obě používají stejné grafické prvky. Jsou to modrý kruh se slabým bílým ohraničením, který je jasně viditelný na pozadí barevné mapy. Takto jsou znázorněny aktivní body na které lze v mapě kliknout. Ohraničení je důležité hlavně pro mapu virtuální procházky, kde vytváří předěly mezi jednotlivými úseky vybrané trasy procházky. Doplňující položky okna virtuální procházky vizuálně odpovídají návrhu tlačítek.

Základem posuvníku je vizuální styl tlačítka, kde samotná hlava posuvníku je zobrazena s plně bílým pozadím. Plně bílé pozadí jí vizuálně odlišuje od tlačítek a naznačuje tak jinou funkcionalitu.

Minimapa a čas využívají polo průhledného bílého pozadí a tmavého textu a tím zapadají do ostatních položek hlavní obrazovky. v minimapě je navíc použit tmavý obrys jako předěl mezi samotnou minimapou a popisem lokace na které se uživatel právě nachází.

Obrázek 4.3. Scéna se zobrazeným uživatelským rozhraním hlavní obrazovky.



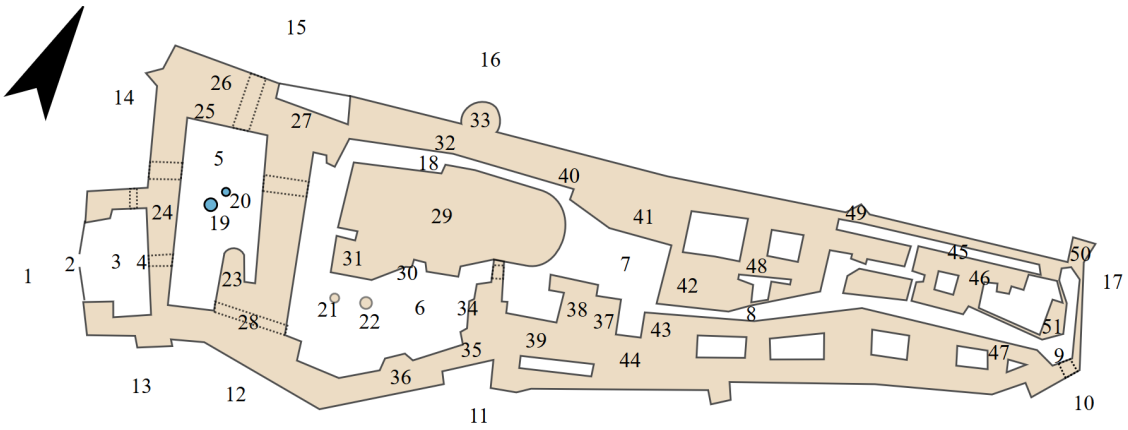
Obrázek 4.4. Scéna s otevřeným oknem nastavení času v záložce nastavení.

Kapitola 5
Model Pražského hradu

V této kapitole je popsán areál Pražského hradu a následně postup vytváření jeho modelu. Je zde definován rozsah modelu a cílené detaily jeho jednotlivých částí.

5.1 Areál Pražského hradu

Na obrázku 5.1 je zobrazen plán areálu Pražského hradu. Aplikace se snaží zejména o zobrazení těchto objektů. Tyto objekty jsou hlavní částí památkové rezervace a jsou i nejzajímavější částí pro turisty a návštěvníky. Zároveň jsou právě tyto objekty umístěny na hřebeni kopce a tvoří ikonický horizont, který je použitý i v načítací obrazovce a ikoně v této aplikaci.



Obrázek 5.1. Areál Pražského Hradu a jeho významné objekty[58].

5.2 Podklady

V této části jsou popsány podklady použité pro vytvoření modelu Pražského hradu. Při tvorbě scény jsem používal 3D modely Pražského hradu třetích stran a množství fotografií a zejména map. 3D modely třetích stran byly použity jako vodítko a každý z nich byl kompletně upraven. Stejně tak fotografie a mapové podklady.

5.2.1 Institut plánování a rozvoje hl. m. Prahy

Institut plánování a rozvoje hl. m. Prahy (IPR)[59] je institut, který z pověření pražského magistrátu mapuje momentální stav hl. m. Prahy. Na základě tohoto mapování později vytvoří plán rozvoje města, který se nazývá Metropolitní plán. Pro nás jsou zajímavé vytvořené mapové podklady, které jsou volně dostupné a momentálně je to nejpodrobnější zdroj map Prahy, který lze volně nalézt. IPR takto poskytuje mapové podklady jak parcel, tak ortofotomapu, tak 3D model a mnohé další[60].

Základní barevnou mapu jsem vytvořil složením z částí dostupných na webu IPR 5.2. Tím vznikl podklad pro půdorysy celého prostoru Pražského hradu. Uvažoval jsem dále o vytvoření obdobného obrázku, který by určil výšky střech jednotlivých budov. Tuto informaci IPR má, ale místo obrazových podkladů map jsem nakonec použil 3D model, který tuto informaci samozřejmě také obsahuje. Navíc kromě výšky hrany střech, je v 3D modelu informace i výšce celkové.

Obrázek 5.2. Složená mapa scény z Geoportálu Prahy od IPR[60].

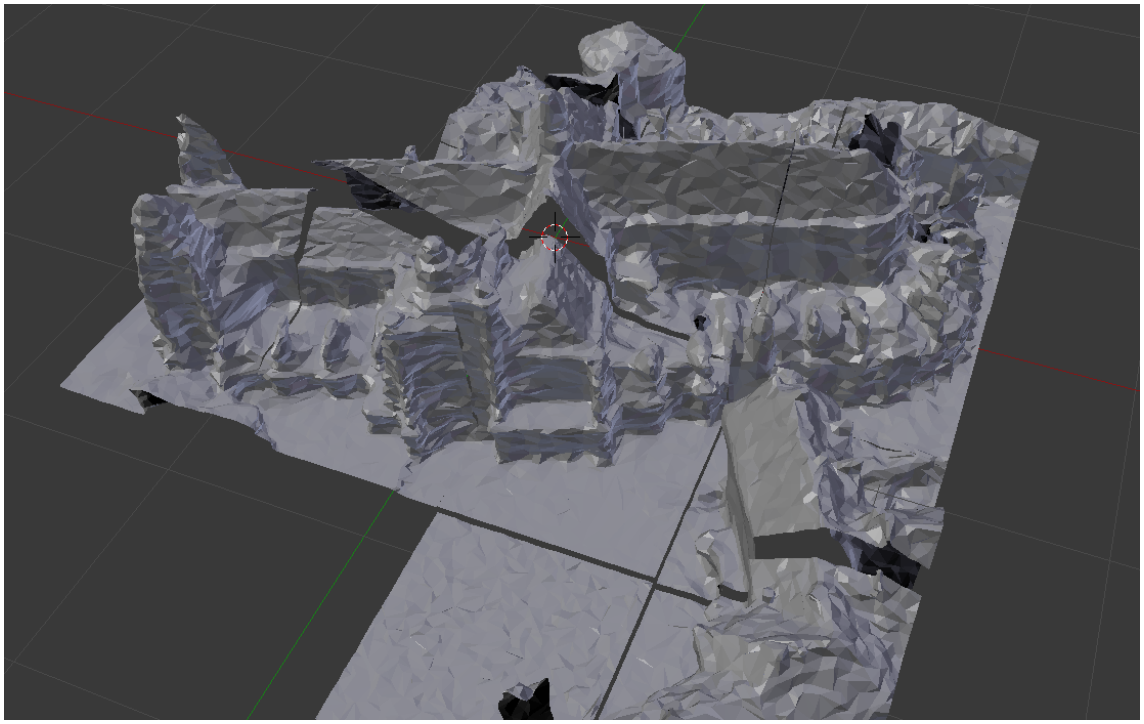
3D model jsem použil jako základ, abych věděl přibližnou výšku a tvar budov. Geometrie byla ale nekvalitní a obsahovala spoustu protínajících se částí. Bylo nutné ji nejdříve vyčistit, ale na pochopení základního tvaru byl model ideální. Odchylka v měření je dle IPR do 0,5m. 3D model terénu jsem získal obdobným způsobem. Zde navíc použitý model od IPR už neobsahoval vegetaci a budovy, čímž se proces vytvoření modelu terénu ještě zjednodušil. Stále bylo ale potřeba celý model vyčistit. Postup vytvoření modelů z těchto podkladů je v samostatné kapitole dále 5.3.1.

5.2.2 Mapy.cz

Z webové stránky mapy.cz je možné získat 3D model, který se zobrazuje při aktivovaném 3D pohledu. Tento model je sice lepší než na dále zmíněných Google mapách, ale stále velmi primitivní 5.3. Model na webu mapy.cz vypadá dobře díky texturám, ale pro naše účely není použitelný. Navíc je scéna rozdělena na části, které se postupně načítají a zobrazují, ale na jejich složení do celistvé scény by bylo třeba stáhnout kromě samotných částí modelu i meta soubory, které těmto částem upravují pozici a zvětšení ve vertikální ose. Bez těchto metadat části scény nenavazují.

Pro modelování podle vizuální předlohy (modelování od oka) je 3D pohled velmi nápomocný. Stejně tak ortofotomapa. v průběhu modelování jsem používal oba zdroje, protože 3D pohled sice zobrazuje více než ortofotomapa, ale zase obsahuje deformace a chyby způsobené automatickým zpracováním.

Panorama (street view) není v oblasti Hradu vůbec zachyceno a zde se musíme spolehnout na jiný zdroj. Například Google mapy popsané dále.



Obrázek 5.3. Geometrie částí modelu z Mapy.cz zobrazená v Blender 3D.

5.2.3 Google mapy

Při modelování od oka jsem použil street view, který zobrazuje většinu míst na Hradě z pohledu od země. Tím bylo možné zjistit jak jednotlivá místa vypadají, bez toho abych musel Pražský hrad zevrubně fotografovat nebo jinak zaznamenávat. Navíc přímo při procházení modelem nabízí Google mapy fotky pořízené návštěvníky na místě. Tyto fotografie mi byly také velmi nápomocné, protože často obsahovaly části, které ani samotné Google mapy neměly vyfocené.

3D model je už od pohledu velmi primitivní a ani jsem se nesnažil ho získat. Mapy.cz mají geometrii modelu zjevně kvalitnější, ale ani ta se bohužel nedá použít. Tato kapitola byla napsána před vydáním nové verze Google Earth (18. 4. 2017), který kvalitu geometrie značně posunul.

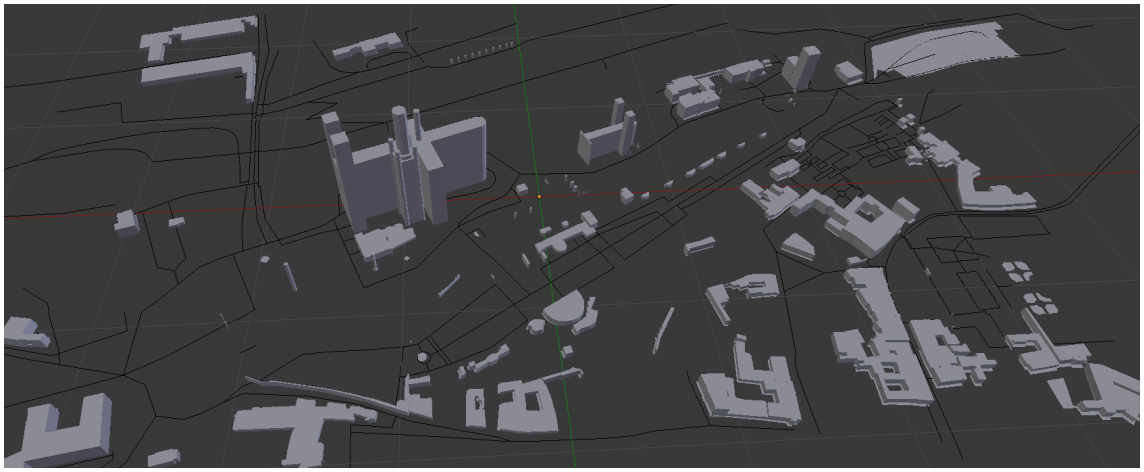
5.2.4 Open Street Map

Open Street Map na svém webu[61] umožňují export částí některých měst. Typickým příkladem je Manhattan v New Yorku, pro který obsahují Open Street Map téměř všechny budovy. Naproti tomu je sice Praha obsažena v nabídce měst ke stažení, ale model obsahuje minimum budov v oblasti Hradu a je velmi primitivní 5.4. Navíc je z nějakého důvodu otočen jinak než model z IPR - přibližně o 20■ kolem vertikální osy.

Export z Open Street Map, ale kromě objektů obsahuje i ulice, názvy a tvar náměstí a zahrad a další mapové podklady zaznamenané jako 3D objekty. To může být užitečné při dalším rozšiřování aplikace. Například při automatickém pojmenování všech ulic scény, které se momentálně dělá ručně.

5.2.5 Papírový model pražského hradu

Architekt Richard Vyškovský vytvořil papírový model Pražského hradu 5.5. Tento model vycházel v časopise ABC s přestávkami od roku 1978. Poslední verze vyšla v roce



Obrázek 5.4. Model z Open Street Map[61]. Vysoká budova uprostřed je Chám sv. Víta.

2002. Výhodou tohoto modelu je, že už je do něj zapracováno zjednodušení originálních objektů. Toto zjednodušení jsem používal jako vodítko, kterým jsem určoval, co ještě modelovat a co už ponechat do textur případně nezobrazovat vůbec, pokud jde o nepodstatný objekt nebo jeho část.

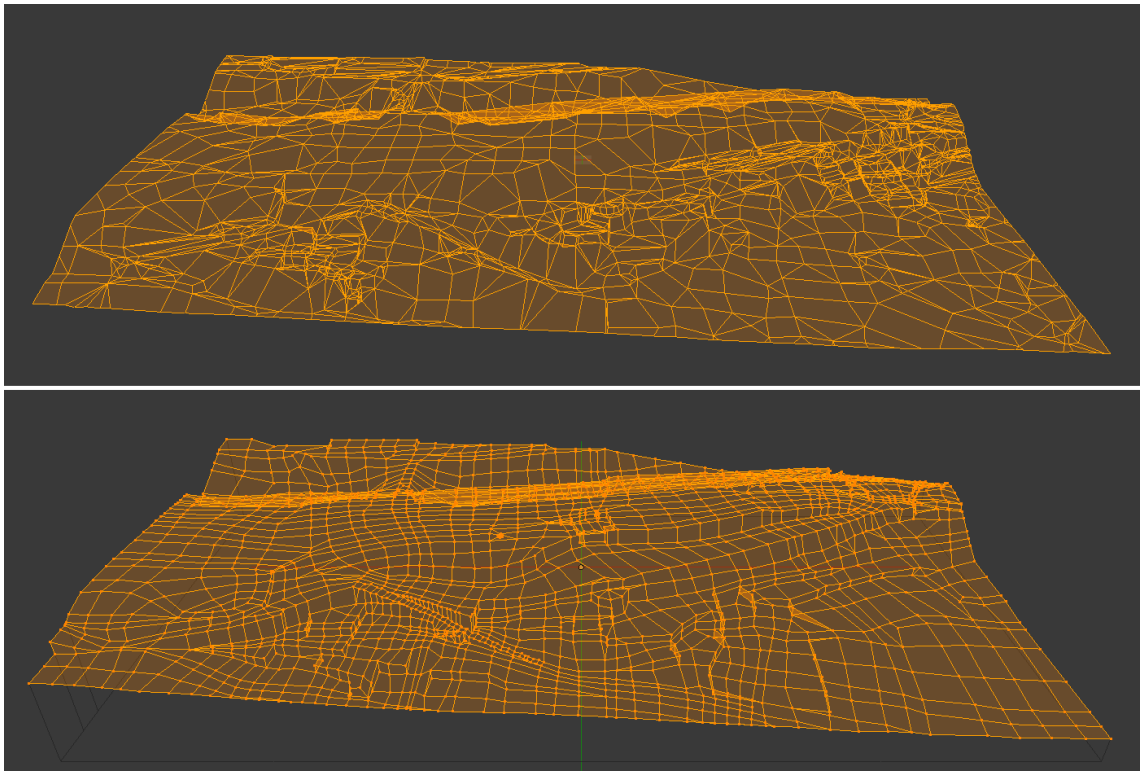
Obrázek 5.5. Papírový model Pražského Hradu od Richarda Vyškovského[62].

5.3 Model

5.3.1 Geometrie

V předchozí závěrečné práci byl popsán postup, podle kterého se nejdříve vytvořil 2D model půdorysu budov a potom nástojem vytažení se z 2D půdorysu vytvořil 3D model. Tímto způsobem lze vytvořit základní tvar budov, ale nejdříve musíme znát výšku jednotlivých budov. Tento základní tvar, lze získat snadněji a s větší přesnosti z jiných zdrojů. Institut plánování a rozvoje Prahy v rámci svých aktivit při vytváření územního plánu, vytvořil 3D model celé Prahy [63]. Ten je volně dostupný ve formátu DWG (CAD formát). Výhoda tohoto postupu je, že všechny rozměry budou přibližně správné a to včetně výšek budov. Uváděná odchylka je do 0,5 metru. Stejně tak budou při tomto postupu budovy umístěny ve správné výšce a budou navazovat na terén.

Základní tvar terénu byl získán také od IPR. Jedná se o výškový scan terénu bez budov a vegetace. Byl potřeba celý předělat pro potřeby interaktivní aplikace. Geometrie byla kompletně předělána a opravena. Oblast použitého terénu pak odpovídá rozsahu



Obrázek 5.6. Porovnání vstupní geometrie terénu z IPR[60] (nahoře) a výsledné mnou upravené (dole).

mapových podkladů použitých v aplikaci. Porovnání zdrojové a výsledné geometrie terénu je na obrázku 5.6.

Přes tento základní tvar šlo snadno vytvořit kvalitnější model budov už jen na základě fotografií. Půdorys budov přibližně sedí, ale jako vodítko je lepší použít složený obrázek z ortofotomapy (v tomto případě také od IPR).

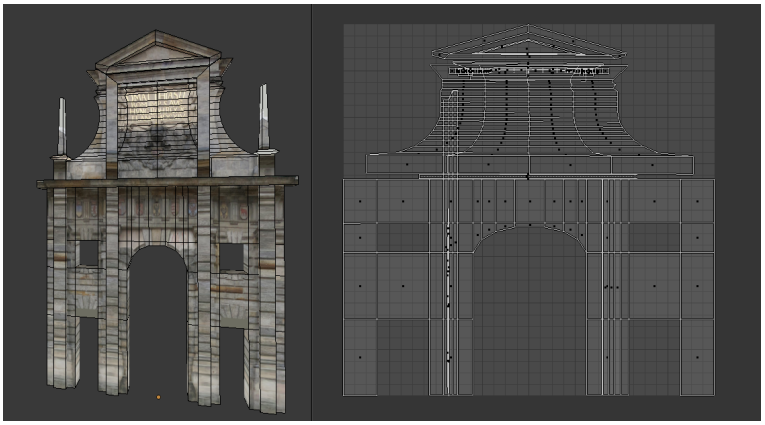
Deltailnější a přesnější model je možné vytvořit na zákaldě geodetickýh měření. Tento postup byl použit například při rekonstrukci Anežského kláštera, jak bylo popsáno v kapitole 4.2. Ze získaných bodů z měření byl v 3D modeláři vytvořen goemetricky přesný model kláštera. i tak musel být model zjednodušený. Čím více je vyžadováno detailu, tím více musí být naměřeno bodů a to více času a prostředků je potřeba.

Nejpřesnější geometrie lze dosáhnout 3D scanováním. Tento postup používá například již zmíněný NIKU. Ti ale ze získaného mračna bodů nevytváří 3D model, ale pouze provádí výzkum stavu památky.

5.3.2 Texturování

Před samotným texturováním je potřeba vytvořit pro každý model jeho UV souřadnice. Tento postup se nazývá UV-unwrap a definuje propojení mezi daty v textuře a geometrií modelu[64]. Příklad výsledných UV souřadnic je na obrázku 5.7. Je vidět, že se plochy tvořené UV souřadnicemi mohou překrývat a mohou i přesahovat plochu textury. Tím je způsobeno, že se textura po objektu opakuje, což je vhodné pro střechy a jiné generické plochy s opakujícím se vzorem.

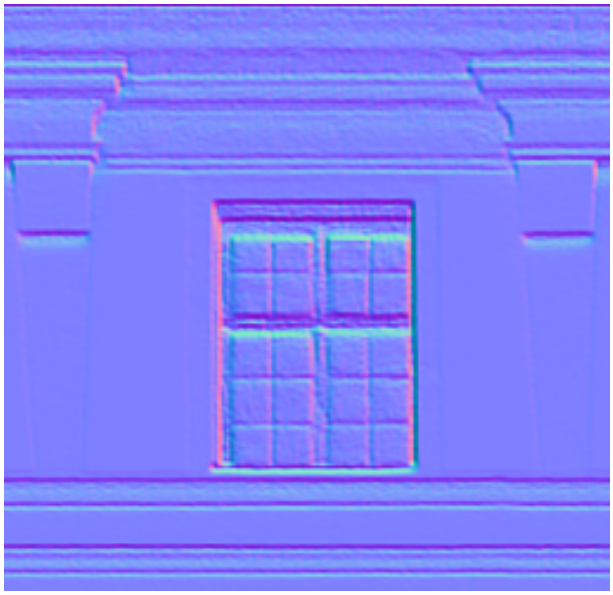
Načítání modelů z formátu Collada (.DAE) podporuje více materiálů na jednom modelu. Stejně tak je více materiálů na jednom modelu podporováno naším 3D modelářem - Blenderem 3D. Díky tomu je přiřazování textur k částem modelu přímočaré.



Obrázek 5.7. Příklad modelu s texturovacími souřadnicemi.

Textury modelů jsou převzaté z předchozí závěrečné práce. Ta ale obsahovala pouze textury s difusní barvou modelů. Dále je popsán postup a problémy při tvorbě doplňujících normálových textur a bump textur.

Normálové a bump textury jsou vytvořeny z původních textur difusních. Protože difusní textury pocházejí z fotografií a navíc jsou uloženy ve formátu JPG, tak obsahují viditelný šum. v případě difusních textur je tento šum vhodný a dodává texturám na realističnosti, ale v případě normálového mapování, se šum projevuje jako chyba. Šum z difusních textur obsažený v normálové textuře je na obrázku 5.8.

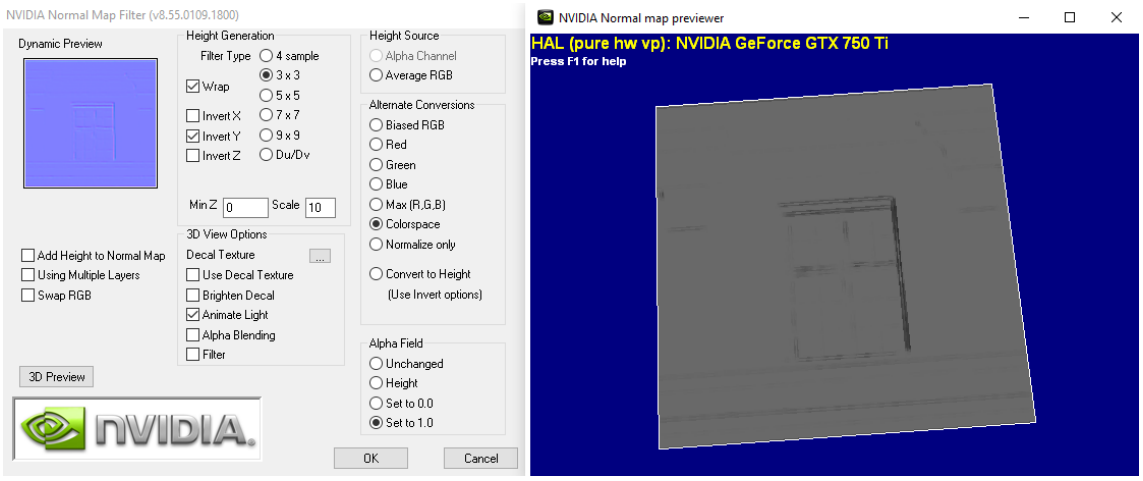


Obrázek 5.8. Šum z difusní textury po konverzi na normálovou mapu.

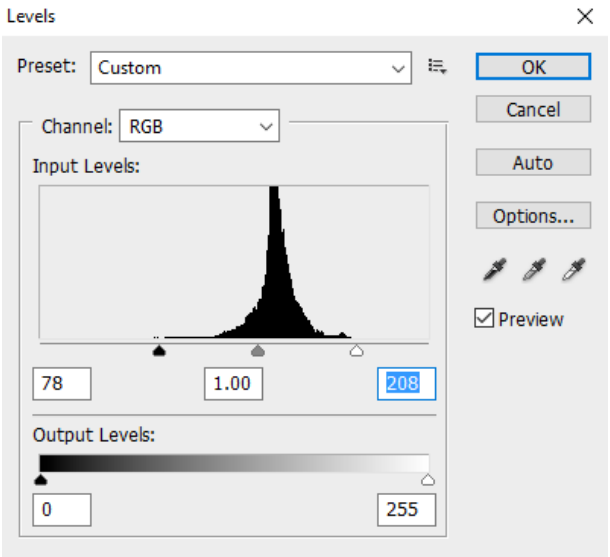
Kvůli šumu difusních textur je nelze automatizovaně a jednoduše převést na normálové textury, respektive bump textury. Před samotným převodem je nutné difusní textury lehce rozmazat. U většiny textur stačilo rozmazání v rozsahu do dvou pixelů funkcí Gaussova rozmazání. Toto je ale u každé textury trochu jiné a nelze to určit obecně. Po rozmazání se už šum neprojevil při konverzi na normálovou texturu respektive bump texturu. Samozřejmě s rozmazáním je potřeba postupovat opatrně, aby byly zachovány základní tvary obsažené v difusní textuře, jinak se může stát, že se data

v textuře po konverzi zobrazí jako rovné nebo téměř rovné plochy. Samostatně byla navíc rozmazána plocha oken, které jinak působily nepříjemným dojmem.

Konverzi upravených difusních textur jsem pak prováděl přes doplněk photoshopu NVIDIA Normal Map Filter[65]. Používané nastavení a náhled přímo v programu Photoshop je vidět na obrázku 5.9. Před samotnou konverzí se mi osvědčilo převést difusní texturu do odstínů šedi a razantně zvýšit kontrast a naopak snížit jas. Konkrétní hodnoty závisí na použité textuře. Obdobně lze upravit černobílou difusní texturu nástrojem úrovní, tak aby používala celý rozsah 8 bytů 5.10.



Obrázek 5.9. Nastavení a náhled konverze normálové mapy.



Obrázek 5.10. Nastavení rozsahu barev přes nástroj úrovní.

5.3.3 Struktura scény v Blender 3D

Scéna je v modeláři Blender 3D rozdělena do vrstev. To umožňuje snadnejší práci a orientaci v modelu. Obsah jednotlivých vrstev je popsán v tabulce 5.1.

Index vrstvy	Obsah vrstvy
1	Areál Pražského Hradu
2	Okolní budovy
3	Terén
4	Zdroj cest pro vyhledávání procházky
5	Stropy a podlahy pro průchody
6	Noční světla
10	Vrstvy 1, 2, 3 a 20
11	Vrstvy 1, 2, 3 a 5
19	Původní model Pražského hradu z předchozí závěrečné práce - bez terénu
20	Kamera a světla pro vykreslování v Blender 3D

Tabulka 5.1. Obsah vrstev zdrojového modelu.

5.4 Export modelu

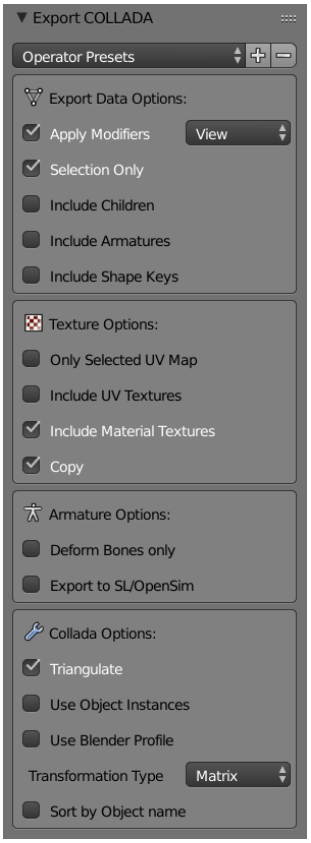
Model scény otevřený v Blenderu 3D obsahuje ve vrstvě 11 (standardně klávesová zkratka alt + 1) všechny objekty, které se zobrazují v aplikaci. Obsaženy jsou modely terénu, areálu Pražského hradu, okolních budov, a objekty stropů a podlah. Stropy a podlahy lze pro finální export odstranit a používat je pouze pro generování výškové mapy. Nicméně je to pouze zlomek geometrie (1100 trojúhelníků). Stropy a podlahy jsou v každém případě potřeba pro generování výškové mapy.

V předchozí závěrečné prácí byl pro export scény určen formát Collada (.dae), který jsem použil i já. Do exportu se dostaneme přes File -¿ Export -¿ Collada (default) (.dae). Při samotném exportu je vhodné použít nastavení dle obrázku 5.11. Tím se zajistí správné zobrazení scény. Důležitá je zejména položka Apply modifiers, která zajistí aplikování modifikátorů na geometrii.

V exportu by bylo možné vybrat položku `Use Object instances`, která objekty se sdílenou geometrií vyexportuje jako instance. Tím dojde ke zmenšení výsledného souboru scény o 10%, ale aplikace není na instance připravena a ty se zobrazují pouze jednou. Model po exportu dosahuje velikosti 9.7MB, což je na webovou aplikaci hodně. Naštěstí při aktivované gzip kompresi se velikost modelu sníží na 2.31MB. Kompresi bude na většině serverů potřeba aktivovat ručně. Appache jde takto nakonfigurovat přidáním řádku následujícího kódu do souboru .htaccess.

```
AddOutputFilterByType DEFLATE model/vnd.collada+xml
```

Zároveň jsou s modelem scény uložené dva skripty. Skript `ExportWalk` slouží k exportu struktury procházky do formátu JSON. Skript `ExportLights` vyexportuje světla. Oba skripty jsou napsány v jazyce Python 3 [66]. Oba exportují pouze vybrané objekty. Pro export procházky je tak nutné vybrat objekt procházky a spustit skript (standardně klávesová zkratka alt + p). Odbobně u světel lze vybrat jen některá světla a pouze ty vyexportovat. Před použitím skriptů je nutné nastavit cestu k výslednému souboru. Je vhodné tuto cestu nastavit přímo do složky s aplikací.



Obrázek 5.11. Nastavení exportu scény do formátu Collada v Blender 3D.

Kapitola 6
Implementace

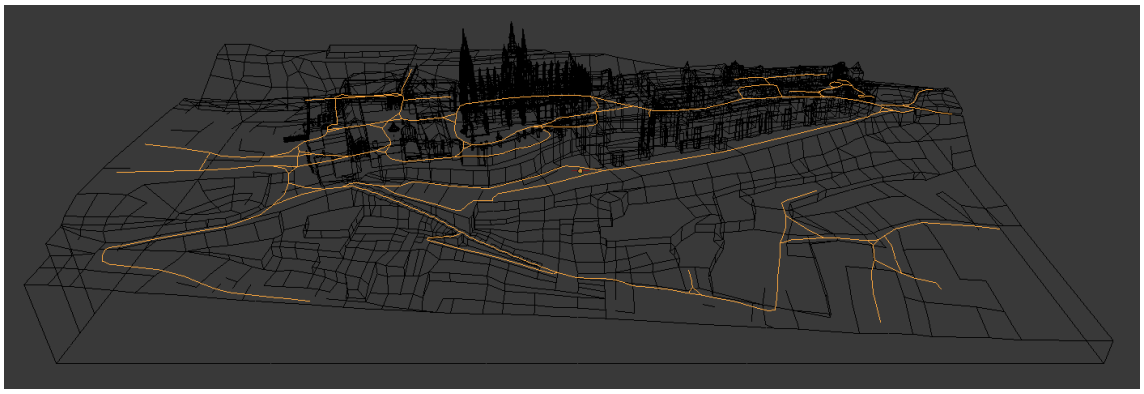
Do aplikace je přidán objekt Tour, který spravuje stav a průběh virtuální procházky.

6.1 Formát tras virtuální procházky

Graf virtuální procházky a názvy míst jsou uloženy v samostatném souboru ve formátu JSON s následující strukturou.

```
{
  vertices: [1,2,3,4, …],
  indicies: [1,2,3,4, …],
  texts: {"Kaple svatého Kříže": 1, …}
},
...
```

Vertices je pole souřadnic vrcholů ve třech dimenzích. **Indicies** je pole indexů do pole vrcholů, které po dvojicích tvoří jednotlivé hrany. **Texts** obsahuje názvy míst přiřazené jednotlivým vrcholům. Podle tohoto názvu lze body vyhledat.



Obrázek 6.1. Geometrie cest virtuální procházky.

V Blenderu 3D je procházka reprezentována jako mesh složený pouze z vrcholů a hran, viz obrázek 6.1. Jeho podoba přesně odpovídá datům ve vyexportovaném souboru. k exportu slouží skript napsaný v pythonu, který z právě vybraných objektů provede export. Zásadní část skriptu, kdy se převedou data z interních struktur Blenderu 3D do struktury odpovídající výslednému JSON je zobrazen v následujícím kódu.

```
def makeAttributes(object, mesh):
  expRound = 3;
```

```
    vertices = [round(j, expRound) for i~in mesh.vertices for j in i.co.xyz]
    indicies = [j for i~in mesh.edges for j in i.vertices]
    texts = {
      object.vertex_groups[j.group].name: id
        for id, i~in enumerate(mesh.vertices)
        for j in i.groups
    }
    return {
      'vertices': vertices,
      'indicies': indicies,
      'texts': texts
    }
```

Získání vrcholů a hran z modelu je triviální. Problém je pouze s názvy míst, které jsou reprezentovány jako pojmenované skupiny vrcholů. Tyto skupiny by měli mít pouze jeden vrchol.

6.1.1 Trasa virtuální procházky

Dříve vyexportovaný JSON s daty virtuální procházky je během spouštění aplikace načten a je z něj vytvořen graf. Za sousední jsou považovány ty vrcholy, mezi kterými leží hrana. Zároveň je spočtena a zaznamenána vzdálenost mezi vrcholy, aby se při hledání nemusela počítat stále dokola.

Uživatel zadává body zájmu prostřednictvím klikání na ně v mapě nebo zadáním názvu místa do textového pole u mapy. Ve chvíli, kdy jsou zadány dva nebo více bodů vyhledá aplikace nejkratší cestu mezi zvolenými body a zobrazí ji uživateli.

Úseky cesty jsou vždy hledány v pořadí, v jakém byly body zájmu zadány. Pokud uživatel bod zájmu odebere, je pořadí ostatních upraveno tak, aby si zachovaly své pořadí v jakém byly zadány. Nový bod zájmu je vždy zařazen za konec stávající cesty.

Vyhledání cesty mezi zadanými body je zajištěno Dijkstrovým algoritmem[67]. Nalezená cesta je vždy nejkratší. Cesta je uživateli v mapě zobrazena zvýrazněním hran mezi vrcholy, kterými prochází. Jakmile je cesta nalezena, může se uživatel rozhodnout ji projít. k tomu slouží tlačítko v mapě. Pokud se mu cesta nelíbí může ji celou smazat druhým tlačítkem z mapy.

Průchod cestou je řešen jako posun kamery uživatele scénou. v každém snímku aplikace je kamera o kus posunuta po vyhledané trase. Aplikace určuje pouze polohu kamery a naopak nechává uživateli volnou rotaci kamerou, která je ovládána stejně jako v případě volného procházení, popsaného v předchozí diplomové práci.

6.1.2 Picking

V předchozí závěrečné práci byla pro určení modelů s aktivním pickingem použita předpona `pickable_`. Rozhodl jsem se toto omezení odstranit a místo toho umožnit picking pro všechny objekty, které mají text. Tyto objekty jsou určeny podle nastaveného názvu v ContentLibrary.js. Jakmile má objekt nastavený název, předpokládám, že je mu vytvořen i popis. Název a popis je vše, co se zobrazí po kliknutí na objekty s aktivním pickingem a tedy všechny tyto objekty mohou mít picking aktivní.

Toto řešení je vhodné, protože se název objektu používá pouze v souvislosti s pickingem. Zobrazuje se v titulku okna, které se ukáže po kliknutí a také jako text u kurzoru pokud uživatel najede kurzorem na objekt s aktivním pickingem.

6.1.3 Webové úpravy

6.1.4 Načítací obrazovka

Zobrazení načítací obrazovky v předchozí závěrečné práci vyžadovalo stažení skriptu, který ovládal její chování a stažení obrázku, který se na středu obrazovky zobrazoval.

Nahradil jsem tuto načítací obrazovku novou, která je celá obsažena v kořenovém souboru index.html. Tento soubor se načte jako první a všechny ostatní se načítají až dle jeho obsahu. Aplikace tak okamžitě po otevření zobrazí načítací obrazovku, která signalizuje, že se něco děje. Grafické zpracování je na obrázku 6.2.

V obou případech je načítací obrazovka pouze indikátorem, že je třeba počkat a neukazuje jednoznačný ani odhadovaný čas do plného spuštění. To je dáno z části těžko předvídatelnou dobou přenosu po síti, ale především také tím, že i po stažení všech skriptů je potřeba ještě nějaký čas na jejich zpracování prohlížečem. Čas na zpracování v prohlížeči je v našem případě tak velký, že dokonce způsobuje blokování celé aplikace a tím i záseky v animaci načítací obrazovky.

Jakmile jsou všechny podstatné části aplikace načteny, zejména model scény, je načítací obrazovka schována a uživateli se zpřístupní celá aplikace.

Obrázek 6.2. Načítací obrazovka

6.1.5 Favicon

Aplikace předchozí závěrečné práce neměla žádnou ikonu v záložce prohlížeče. Rozhodl jsem se tuto ikonu doplnit. Jako obsah ikony jsem zvolil tvar střechy a věží katedrály sv. Víta, která je pro Pražský hrad ikonickou a nejznámější stavbou.

K tvorbě jsem použil veřejně dostupný nástroj[68], který z obrázku vytvoří nejen ikonu pro prohlížeč, ale i její verze v několika velikostech pro zobrazení na mobilních zařízeních. Pokud si uživatel, uloží aplikaci na plochu svého mobilního zařízení, objeví se na ploše místo standardní ikony webového prohlížeče právě tato.

6.1.6 Minimalizace

Jelikož se příliš knihoven nepodařilo odebrat, protože to nebylo možné kvůli rozsahu nutných změn, rozhodl jsem se řešit alespoň problém velikosti skriptů a stylů v aplikaci. Tento problém jsem částečně vyřešil prostřednictvím automatické minimalizace.

Jako nástroj pro automatické zpracování skriptů a stylů jsem zvolil Gulp[55]. v něm jsem definoval dva způsoby zpracování. Jeden pro vývoj aplikace a druhý pro produkční verzi.

Běh pro vývojovou verzi kontroluje změny při uložení skriptů a stylů a kopíruje tyto soubory do cílové složky. Před zkopírováním by mohly proběhnout nějaké úpravy jako v produkčním běhu, ale v tomto případě nejsou zapotřebí.

Produkční běh provádí minimalizace a optimalizace skriptů a stylů. Ze skriptů jsou tak odstraněny komentáře a nepotřebné bílé znaky a zkráceny názvy proměnných. Tím

jsou skripty zmenšeny a jejich přenos přes síť je méně náročný. Tím se i zkrátí doba, po kterou je aplikace při otevření neaktivní a čeká se právě na stažení skriptů. Ze stylů se také odstraňují bílé znaky a navíc je automaticky zajištěna zpětná kompatibilita mezi různými prohlížeči a speciálně pro Internet Explorer do verze 9. To je zajištěno doplněním prefixů nebo nahrazením stylů specifickými styly konkrétního prohlížeče a jeho verze.

V případě knihoven třetích stran je běžné používat v produkčním prostředí minimalizované verze knihoven dodané přímo jejich autory. v případě této aplikace byly ale některé knihovny třetích stran upraveny v rámci předchozí závěrečné práce. Použití minimalizovaných verzí od původních autorů tedy není možné, protože by došlo ke ztrátě našich změn. z toho důvodu byl pro minimalizaci knihoven použit stejný postup jako pro minimalizaci našich skriptů.

V rámci produkční verze je navíc nad skripty použita optimalizace, která není zaměřena na zrychlení stahování, ale na zrychlení zpracování skriptu prohlížečem [69]. Tato optimalizace vychází z předpokladu, že moderní javascriptové enginy provádí před celkovým parsováním skritptu parsování částečné, které je rychlé a akorát se při něm odhaduje, které části kódu jsou spuštěny okamžitě a které později. Později spuštěný kód je například reakce na kliknutí myši. Tato heuristika přeskakuje takzvané immediately-invoked funkce, které jsou ale jedním ze základních kamenů na členění jmeného prostoru v javascriptových aplikacích. Použitá optimalizace tyto funkce zaměňuje tak, aby nebyl narušen jejich význam a aby se jimi javascriptový engine zabýval hned při prvním rychlém parsování.

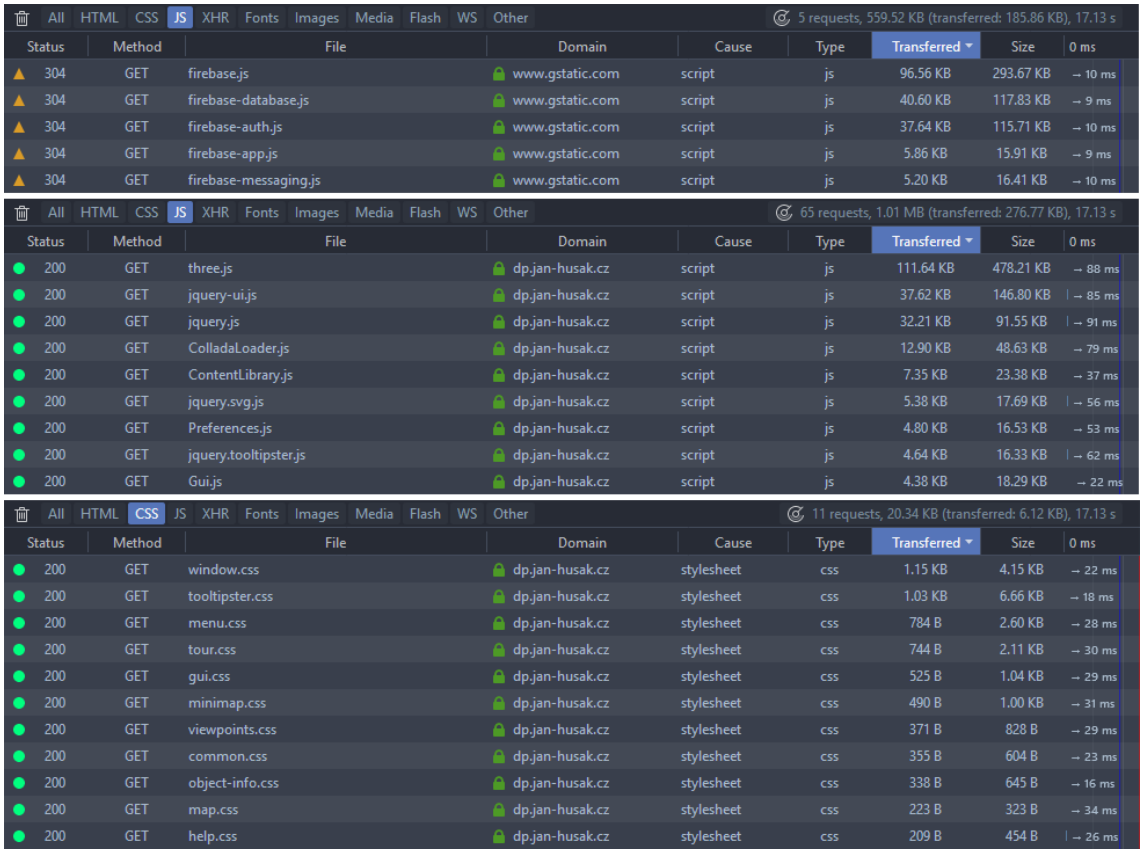
Další velmi používaný postup je složení skriptů do jednoho souboru a stejně tak stylů. Tento postup je užitečný především pro webové aplikace, které běží přes protokol HTTP a tedy nemají k dispozici takzvané proudy protokolu HTTP/2[70]. Proudy totiž umožňují přenos více souborů najednou v rámci jednoho spojení a tím zrychlují stažení a tedy i zobrazení aplikace. HTTP/2 je ale podporováno pouze na zabezpečeném spojení protokolu HTTPS. v našem případě předpokládám, že pokud bude aplikace zveřejněna na webu a bude zobrazována přes síť, tak poběží na protokolu HTTPS a tedy i HTTP/2. Pokud je aplikace spuštěna lokálně z disku, což je možné, pak tento odstavec nemá žádný význam. Ani v jednom případě není nutné bránit se většímu množství souborů. Skládání souborů do jednoho má navíc negativní vliv na vývoj, kdy je nutné hledat chybu během běhu nejdříve ve složeném souboru a až podle něho v původním zdrojovém skriptu nebo stylu. Toto lze řešit načítáním složených souborů pro produkční verzi a původní struktury souborů pro verzi vývojovou, ale v tomto kontextu je to práce navíc, která nemá dostatečné opodstatnění.

Minimlizací jsem zmenšil celkovou velikost skriptů původní aplikace z 3.12MB MB na 1.75MB. Navíc pokud aplikace běží na dobře nakonfigurovaném serveru s nastavenou gzip kompresí přenosu dat, tak se skripty změnší na pouhých 524KB. Celkově pak aplikace přenáší po síti 20.14MB, kde většina z toho jsou obrázky, které dohromady mají 17.23MB.

6.1.7 HTTPS

Použitý síťový protokol není přímo součástí aplikace, nicméně má dostatečný význam pro běh i prestiž aplikace, aby zde byl zmíněn.

Jak už bylo řečeno v optimalizaci skriptů, HTTPS umožňuje rychlejší načtení aplikce. Původní záměr HTTPS je ale především zabezpečení spojení pro přenos citlivých informací jako jsou hesla, čísla platebních karet apod. v této aplikaci k takovému přenosu informací sice nedochází, ale i tak zvýšíme prestiž aplikace u technicky znalých uži-



Obrázek 6.3. Statistiky přenosu po síti při otevření aplikace. Firebase (nahoře), větší skripty aplikace (uprostřed) a styly (dole).

vatelů a její hodnocení vyhledávacími roboty, pokud aplikaci nasadíme na protokol HTTPS. Moderní prohlížeče se totiž snaží, aby bylo HTTPS použito plošně a dosahují toho zobrazením kladného nebo varovného symbolu v záhlaví prohlížeče respektive aktivní záložky. Stejným způsobem jako prohlížeče přistupují k problematice webové vyhledávače, akorát to není tak zřejmé.

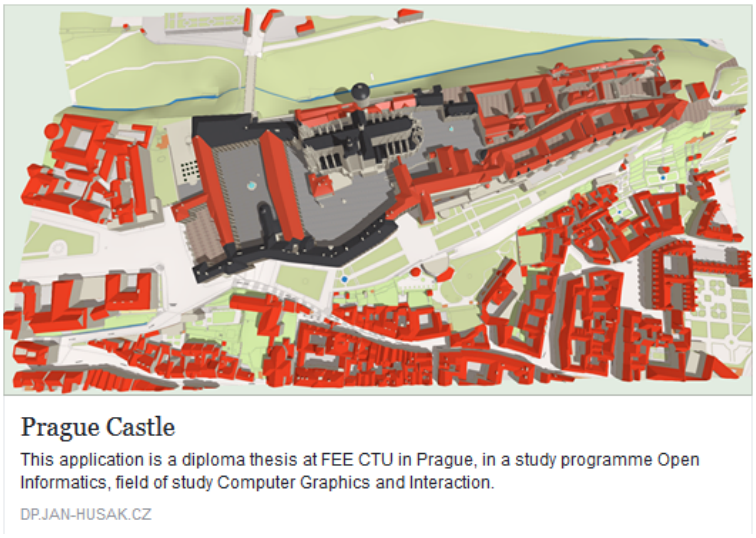
6.1.8 Open Graph protokol

Open Graph Protocol[71] určuje tagy, kterými je možné popsat dokument, aby byl jednoduše zařaditelný. Facebook vyvinul tento protokol, aby mohl snadno třídit a zobrazovat webové stránky, které uživatelé skrz jeho sociální sít sdílí.

Rozhodl jsem se tyto tagy doplnit do hlavičky dokumentu, abych mohl nasdílet aplikaci na Facebooku. Sliboval jsem si od toho rychlou zpětnou vazbu uživatelů, kterou jsem i dostal. Ta je předmětem uživatelského testování, které je popsáno dále 7.1.

Nastavení Open Graph Protokolu se provádí pomocí tagů v hlavičce HTML dokumentu. Tyto tagy obsahují popisky aplikace a její obrázek obrázek. Výsledný příspěvek je na obrázku 6.4 a následující kód obsahuje definici meta tagů.

```
<meta property="og:type" content="website" />
<meta property="og:title" content="Prague Castle" />
<meta property="og:description" content="This application is..." />
<meta property="og:url" content="https://dp.jan-husak.cz" />
<meta property="og:image" content".../prague-castle.png" />
```



Obrázek 6.4. Formát příspěvku na sociální síti Facebook.

6.2 Uživatelské rozhraní

6.2.1 Celoobrazový režim

Celoobrazový režim byl implementován podle článku Davida Walshe[72]. Samotné spuštění celoobrazového režimu je snadné, ale je třeba mít na paměti různorodost prohlížečů. Jelikož se jedná o poměrně novou funkci, tak ji některé prohlížeče podporují pouze ve verzi s prefixem ve jméně. Kód přepnutí do režimu celé obrazovky pak vypadá jako v následujícím kódu. Kód není ničím převratným, ale jedná se o typické řešení různorodosti prefixů a podporovaných funkcí v jednotlivých prohlížečích. Kód testuje všechny známé verze funkce na přepnutí do režimu celé obrazovky, a když nějakou najde, tak ji použije.

```
if(element.requestFullscreen) {
  element.requestFullscreen();
} else if(element.mozRequestFullScreen) {
  element.mozRequestFullScreen();
} else if(element.webkitRequestFullscreen) {
  element.webkitRequestFullscreen();
} else if(element.msRequestFullscreen) {
  element.msRequestFullscreen();
}
```

Standardně se z tohoto režimu uživatel vrací klávesou escape. To je zabudované chování prohlížečů a může se lehce lišit. Escape nešťastně koliduje s klávesovou zkratkou pro zavírání oken v aplikaci, ale jedná se o typickou klávesu pro takovou funkci, takže použití jiné by nedávalo příliš smysl.

6.2.2 Animace otevírání oken

Animace otevírání oken je zobrazena prostřednictvím transitions, což je část CSS 3. Ve chvíli, kdy je okno otevřeno, je mu nastavena třída, která obsahuje transformaci zvetšení a posunu. Zvětšení i posun je animováno a okno tak při otevření vypadá, jako by k uživateli přijíždělo z dálky v oblouku.

Transitions fungují tak, že se nastaví základní stav nějaké css vlastnosti, její druhý stav v rámci doplňující třídy nebo pseudo třídy (najetí myši na element) a pak už stačí jen nastavit časování a způsob přechodu mezi těmito dvěma stavy.

Základním stavem je zde element zmenšený na nulu a o kus posunutý pod střed obrazovky. Druhým stavem je zvětšení na hodnotu jedna a tedy jeho faktické anulování a posunutí přesně na střed.

6.2.3 Zobrazení ostatních virtuálních návštěvníků

Aplikace během svého běhu zobrazuje ostatní aktivní návštěvníky. Každý návštěvník je tak viděn ostatními jakmile otevře aplikaci, je vidět jeho pohyb v průběhu používaní aplikace a po zavření aplikace je z virtuální scény odstraněn.

Standardně vyžadovala podobná funkcionalita vytovření serverového řešení, které by zpracovávalo požadavky uživatelů a provádělo synchronizaci. To je velmi náročné a stačilo by to na samostatnou závěrečnou práci. Naštěstí existuje služba Firebase, která celou tuto problematiku za vývojáře řeší a umožňuje tak soustředit se na vývoj uživatelských funkcí namísto serveru pro aplikaci. Díky tomu může naše aplikace zůstat přenosná a není závislá na ničem, co by bylo nutno instalovat nebo jinak nastavovat.

Firebase je databáze, která provádí automatickou synchronizaci s připojenými klienty. Mimo jiné umožňuje dotazy na data, autentizaci uživatelů a mnoho dalšího, ale v našem případě je důležitá hlavně zmíněná automatická synchronizace.

Jakmile uživatel otevře aplikaci spustí se stahování skriptů Firebase. Jakmile ty se stáhnou je odeslán na servery Firebase požadavek o novém připojení. v reakci na požadavek je v databázi vytvořen záznam o uživateli a jeho pozici ve scéně. Následuje reakce ze serveru, která předá klientovi pozice všech ostatních aktivních uživatelů.

Během používání aplikace jsou pozice ostatních uživatelů synchronizovány automaticky a na jejich správné zobrazení ve scéně stačí správné nastavení funkcí, které reagují na požadavky ze serveru. Zároveň každý pohyb uživatele ve scéně je předán staženému skriptu Firebase a ten už sám zařídí komunikaci se serverem a předání informace ostatním uživatelům.

Jakmile uživatel opustí aplikaci je spojení přerušeno a záznam o uživateli na serveru je zrušen a tím je odstraněn i z virtuální scén ostatních uživatelů.

V několika jednoduchých krocích je tak vytvořen virtuální prostor ve kterém se mohou uživatelé vzájemně potkávat. Momentálně není vytvořena žádná možnost komunikace mezi uživateli, protože už toto zobrazení ostatních uživatelů je na hraně zadání, ale bez větších obtíží by bylo možné přidat textovou komunikaci, nastavení stylu zobrazení uživatele nebo jakoukoli jinou interakci.

Při používání služby Firebase se v aplikaci někdy stane, že uživatel zůstane zaznamenán i po ukončení aplikace a nezmizí. Jelikož navíc spojení už je zrušeno, záznam o uživateli, tak zůstane uložen v databázi. Navíc není připojení uživatele do naší aplikace ani Firebase databáze vázáno žádnou registrací ani přihlášení a nemá tedy žádný unikátní identifikátor na který by se dalo odkázat při opětovném připojení uživatel. Tím vznikají ve scéně hromadící se pohrobky odpojených uživatelů. Tento problém jsem nakonec vyřešil vymazáním obsahu celé databáze při ukončení spojení s jakýmkoli uživatelem. Tím sice přijdeme o veškerá data o uživatelích, ale ty jsou vzápětí obnovena příchozími požadavky ze zařízení ostatních uživatelů.

6.3 Osvětlení

Aplikace již z předchozí závěrečné práce podporovala noční osvětlení, které dodávalo scéně na realističnosti a dobře vystihovalo noční atmosféru Pražského hradu. Nicméně

vzhledem k rozsahu detailní části modelu, který se omezoval zejména na okolí prvního nádvoří, bylo použito jen malé množství světel.

Doplnění celého modelu o noční osvětlení pomocí bodových světel rozmístěných po scéně, znamenalo razantní zvýšení náročnosti vykreslování. Po prozkoumání původní funkcionality se ukázalo, že noční osvětlení je řešeno dopředným vykreslováním a celá scéna je tak pro každé světlo vykreslena samostatně.

Navíc použití většího množství světel (už okolo 15), způsoboval spotřebování dostupných varying proměnných v shader programech. To dále způsobylo chybu při linkování shader programu a objekty využívající takový shader program se ve scéně vůbec nezobrazily. Navíc toto selhání linkování programů vyústilo v dlouhý zásek aplikace v jednotkách vteřin, který byl uživatelsky nepříjemný.

Problém je o to větší, že Three.js skládá kód shader programů dynamicky při běhu aplikace na základě nastavení svých interních materiálů, které jsou odvozeny z materiálů objektů. Kód je skládán z předem připravených částí (chunků), které jsou základem knihovny Three.js a není možné, a rozhodně není vhodné, do chunků kódů zasahovat.

Zároveň aby aplikace běžela plynule byl už v rámci předchozí práce změněn styl stínování z Phongova na Gouraudovo. To dále upozorňuje na problém s velkým množstvím světel a celkovou náročností vykreslování.

Aby se scéna plynule vykreslovala, ale zároveň bylo možné osvětlit celý rozsah scény, pomocí malých světel jako je tomu ve skutečnosti, bylo by ideální přepracovat způsob vykreslování na deferred shading[73], který je přizpůsoben na vykreslování většího počtu světel.

Seznam světel a jejich pozice byly původně součástí kódu aplikace v souboru Scene.js. Rozhodl jsem se přesunout seznam světel do samostatného souboru. Tento soubor je generovaný skriptem ExporterLights, který je součástí zdrojového souboru scény pro Blender. Skript vyexportuje z Blenderu vybranné objekty světel a uloží je ve formátu JSON.

Seznam světel by také mohl být uložen rovnou v rámci souboru scény. To by ale znamenalo zesložitění ColladaLoader.js, který je ve skutečnosti dodaný třetí stranou a je pouze upravený pro potřeby této aplikace. Rozhodl jsem se proto uložit seznam světel do samostatného souboru, který je načítán s osatními skripty aplikace a není potřeba řešit jeho dynamické načítání.

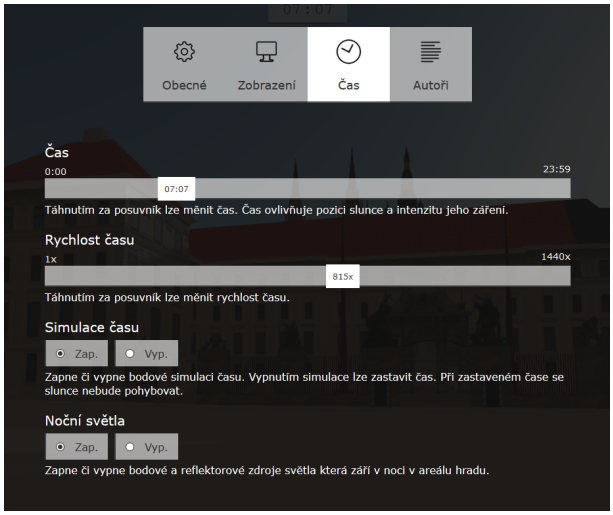
Dále jsem noční osvětlení rozšíril o nastavitelnou hodnotu rozsahu osvětlení pro jednotlivá světla. Tím je možno nastavit osvětlení zapadlých koutů scény i rozsáhlých ploch, jako jsou nádvoří. Stísněné i velké otevřené prostory jsou ve scéně obsaženy a různě silná světla jsou proto nutností.

S nočním a denním osvětlením souvisí cyklus střídání dne a noci. Aplikace již z předchozí závěrečné práce obsahoval tento cyklus obsahovala, ale nebylo možné nastavit rychlost plynutí času v rámci tohoto cyklu. Doplnil jsem proto toto nastavení, které je zobrazeno jako posuvník určující násobek rychlosti plynutí času oproti skutečnosti 6.5.

6.4 Grafické efekty

6.4.1 Animace vody

Modely s prefixem `water_` mají animovanou texturu. Tento prefix lze změnit v Preferences.js. Animovaná textura znamená pohyb textury po vertikální ose při každém překreslení scény. Tato animace je použita pro simulaci pohybu vody v kašnách a z toho i vyplývá zvolený prefix.



Obrázek 6.5. Záložka nastavení času s nově přidaným nastavením rychlosti času.

Použitá textura musí být beze švů ve vertikální ose, jinak by se švy během animace ukázaly a kazily by celkový dojem animace. Pokud je použita v textuře průhlednost, jako v případě vody, je nutné použít formát PNG, podle kterého je automaticky určeno, že materiál obsahuje průhlednost.

Three.js má zabudovanou vlastnost offset, která určuje posunutí textury. Přes úpravu této vlastnosti v každém snímku je vytvořena animace. Navíc je třeba objektům textur nastavit opakování (wrap), které se ze zdrojového souboru scény samo nenačítá.

6.4.2 Normal a bump mapping

Aplikace nově obsahuje modely s normálovým nebo bump mapováním[74]. Postup tvorby potřebných textur je popsán v kapitole Model Pražského hradu 5.1. Zde je popsán způsob fungování normálového a bump mapování v rámci aplikace.

Oba typy mapování jsou jednotlivým modelům přiřazeny podle přípon v jejich jménech. Lze zobrazit vedle sebe dva modely, které používají různý typ mapování a nebo žádný a vykreslují se tedy akorát se standardní difusní texturou. Navíc modely nemusí mít ani difisní texturu a zobrazí se pouze v difusní barvě. Bez textur jsou vykreslovány okolní stavby Pražského hradu.

Přípony jmen modelů jsou využívány pro nastavení efektů textur, protože při exportu scény z Blender 3D není jak jinak předat tuto informaci až do aplikace. Ze stejného důvodu je potřeba během načítání scény tyto efekty samostatně nastavit. Toto nastavení probíhá v ColladaLoader.js.

6.4.3 Úrovně detailu

Nový model zabírá velkou oblast a obsahuje mnoho samostatných budov. Důležitým bodem pro vykreslování takové scény je určení nejnáročnějších částí vykreslování a pokud jsou náročné příliš, je třeba pro ně vytvořit odpovídající úrovně detailu, aby vykreslování vzdálených objektů zbytečně nenarušovalo plynulost aplikace.

Oproti předchozí závěrečné práci obsahuje model scény více trojúhelníků. Přibližné počty trojúhelníků jsou v tabulce 6.1. Mnoho trojúhelníků bylo ušetřeno na terénu, který zabírá menší oblast a je oproti předchozímu modelu ručně zpracován. Naproti tomu samotný areál Pražského hradu obsahuje trojúhelníků více.

Celkově vzrostl počet trojúhelníků o přibližně 10 000, což není z hlediska vykreslování geometrie mnoho a v aplikaci to nezpůsobuje nárust náročnosti vykreslování.

část modelů	trojúhelníků v původním model	trojúhelníků v novém modelu
Areál Pražského hradu	31 500	53 000
Terén	28 300	4 300
Okolní budovy	-	11 300
Celkem	59 800	68 900

Tabulka 6.1. Obsah vrstev zdrojového modelu.

Další náročnou částí vykreslování je normálové a bump mapování některých materiálů. Tyto efekty v předchozí závěrečné práci nebyly a zvyšují tedy náročnost vykreslování. Normálové respektive bump mapování je standardním efektem, který nevyžaduje příliš výkonu grafické karty. Zejména oproti náročnějším efektům textur, jako je například paralax mapping[75]. Nicméně je zbytečné, aby se tyto efekty vykreslovaly na vzdálených objektech, na kterých stejně nebudou vidět, protože se například celý objekt zobrazí na několik málo pixelů.

Kapitola 7
Testování

7.1 HW a SW nároky

Stejně jako v předchozí závěrečné práci platí, že na stolních počítačích běží aplikace plynule, ale na tabletech, mobilech a obecně menších přenosných zařízeních běží aplikace velmi pomalu. Každý snímek zabere k vykreslení 35 volání draw* funkcí a celkově 802 až 812 volání WebGL funkcí. Druhý zásadní nedostatek na přenosných zařízeních je velikost aplikace. Samotný model dosahuje po kompresi 2.31MB a přenos po síti tak může trvat velmi dlouho. Konkrétní hodnoty ukazují tabulky 7.1 a 7.2. Testování probíhalo v prohlížeči Google Chrome.

Zařízení	CPU	GPU	RAM
Stolní PC s Windows i	Intel Core i7, 3.40GHz	NVIDIA GeForce GTX 750 Ti	16GB
Stolní PC s Windows II	Intel Core i7, 3.40GHz	NVIDIA GeForce GTX 580	8GB
MacBook Air Sierra	Intel Core i5, 1.6GHz	Intel HD 6000	8GB
Notebook s Windows	Intel Atom, 1.33GHz	Intel HD Z3745D	2GB
Tablet s Android	MediaTek MT8127 1.3GHz	Mali-450MP4	1GB

Tabulka 7.1. Parametry testovacíh zařízeních. U CPU je nepodstatný počet jader, protože javascript se v aplikaci vykonává vždy jen na jednom jádře.

Zařízení	Bez efektů	s efekty	samotné noční osvětlení
Stolní PC s Windows i	60	60	60
Stolní PC s Windows II	60	60	60
MacBook Air Sierra	60	39	40
Notebook s Windows	20	10	12
Tablet s Android	3	0	1

Tabulka 7.2. Rychlost vykreslování na testovacíh zařízeních ve snímcích za sekundu. Vykreslování je vertikální synchronizací omezené na maximálně 60 snímků za sekundu.

7.2 Uživatelské testy

Tato část popisuje uživatelské testování aplikace. Je definován postup testování, který vychází z předchozí závěrečné práce (1. až 4. testovací scénář) a jsou testovány subjektivní i objektivní aspekty aplikace. Subjektivní uživatelské testy spočívají v testovacích scénářích, které každý testovací uživatel prochází. Tyto scénáře poskytují jednoznačné odpovědi. Vedle toho je definován seznam subjektivních dotazů na vzhled a celkové fungování aplikace. Scénáře a dotazy jsou doplněny o mé zhodnocení orientace uživatele v aplikaci a jejím rozhraní.

7.2.1 Testovací scénáře

- Scénář 1 - volná procházka
  - Poznáváte některé budovy? Které?
  - Cílem je nechat uživatele volně se procházet scénou a bez nátlaku a nápověd ho nechat poznávat některé objekty. Mimo to nám scénář dává první zpětnou vazbu o líbivosti grafického zpracování. z tohoto scénáře je dále zřejmé, které části scény jsou pro uživatele dobře zpracovány a které ne.
- Scénář 2 - fontána
  - Ve scéně je umístěna fontána. Dokážete jí najít a zjistit její jméno a kdy byla postavena?
  - Cílem je zjistit, jak je uživatele schopen se navigovat virtuální scénou, pokud hledá něco konkrétního. Scénář je také zaměřen na interakci mezi uživatelem a objekty ve scéně. Scénář je splněn, pokud uživatel použije kliknutí na objekt fontány ve 3D scéně. Tím se mu otevře okno s detailem fontány a jejím popisem.
- Scénář 3 - jméno vedoucího závěrečné práce
  - Nalezněte jméno vedoucího této závěrečné práce.
  - Scénář testuje, jestli je uživatelské prostředí dostatečně dobře rozloženo, aby byl uživatel schopen takovou informaci najít. Tato informace je umístěna v okně poděkování a pro uživatele nemusí být toto umístění intuitivní. Navíc se jedná o pro uživatele méně zajímavou informaci.
- Scénář 4 - létání
  - Dokážete se dostat na střechu?
  - Tento scénář testuje, jestli uživatel dokáže splnit náročnější úkol na pohyb ve scéně. To že aplikace obsahuje možnost mód létání, zde nemusí být všem uživatelům zřejmé.
- Scénář 5 - virtuální procházka
  - Dokážete nastavit, aby Vás aplikace sama provedla mezi dvěma objekty které poznáváte?
  - Cílem je otestovat nově přidanou funkci virtuální procházky. Je důležité, aby uživatel uměl tuto funkci najít a intuitivně věděl, jak ji používat. Pokud ne, je důležité, jestli dokáže návod na použití nalézt bez nápovědy.
- Scénář 6 - rychlost plynutí času
  - Nastavte rychlost plynutí času.
  - Uživatel musí nalézt nastavení rychlosti plynutí času. Umístění nastavení nemusí být zcela zřejmé. Nachází se v záložce čas v nastavení. Navíc je nastavení řešeno jako posuvník, který nemusí všichni uživatelé umět ovládat.
- Scénář 7 - zámecké schody
  - Dokážete najít nové zámecké schody?
  - Zámecké schody jsou součástí modelu terénu a jsou zřejmé pouze tvarem. Otexturovány jsou pouze mapou oblasti. i když uživatel nebude vědět, kde tyto schody ve skutečnosti jsou, je možné, že je najde jen podle tvaru geometrie, což by bylo vhodné.

- Scénář 8 - klávesové zkratky
  - Použijte pro pohyb ve scéně klávesové zkratky.
  - Uživatel musí použít pro pohyb klávesové zkratky. Pokud intuitivně neví které, musí být alespoň schopen dohledat si je v nápovědě aplikace. Obojí je pro nás varováním.
- Scénář 9 - zobrazení více uživatelů
  - Dokážete zapnout zobrazení více uživatelů ve scéně a sledovat jiného uživatele?
  - Cílem je zjistit, jestli uživatel dokáže nastavit zobrazování ostatních uživatelů, jestli dokáže nějakého jiného uživatele ve scéně najít a jestli se dokáže ve scéně plynule pohybovat, aby jiného uživatele následoval.

7.2.2 Subjektivní dotazy

Subjektivní dotazy určují základní otázky, které uživatelé v průběhu testování odpovídají. z nich se samozřejmě odvíjí další otázky, které se můžou mezi jednotlivými uživateli různit. Tento seznam je jen vodítko pro určení subjektivního vztahu uživatelů k grafickému zpracování scény a uživatelského rozhraní a také k možným rozšířením.

- Byli jste na Pražském hradě? Znáte objekty pražské hradu?
- Otázka určuje vztah uživatele k Pražskému hradu jako takovému. Pro uživatele, kteří reálný Pražský hrad nenavštívili nebo ho navštívili už hodně dávno, nebude pravděpodobně důležité, nakolik jsou objekty zobrazené v aplikaci shodné s reálnými, ale spíše ocení celkovou kvalitu nebo líbivost zpracování.

- Jak se Vám líbí virtuální procházka v porovnání s volným pohybem?
- Otázka se ptá na hlavní funkci přidanou v této závěrečné práci. Cílem je zjistit, jestli je tato funkce pro uživatele zajímavá nebo by jí nikdy nepoužili. Jak bylo již popsáno dříve, procházka sice může uživateli ukázat zajímavá místa a trasu mezi nimi, ale bez volného pohybu by aplikace mohla připomínat spíše průchod 3D mapou jak ji má vytvořenou Seznam nebo Google.

- Jsou pro Vás i okolní budovy Pražského hradu povědomé?
- Tato otázka směřuje k otestování vizuálního zpracování okolních objektů Pražského Hrad. Ty jsou neotexturované a mají pouze nastaveny šedý materiál pro stěny a červený materiál pro střechy. Pokud se ukáže, že tyto objekty pro uživatele doplňují scénu a jsou pro ně důležité, může dávat smysl doplnit scénu o množství menších objektů v podobném stylu.

- Líbí se Vám grafické zpracování aplikace?
- Uživatel zde hodnotí celkové zpracování 3D modelu scény a jeho zobrazení v aplikaci. Důležitým faktorem zde bude rychlost vykreslování, která mimo jiné subjektivní vjemy bude velmi ovlivňovat uživatelův vztah k zobrazované scéně.

- Vyhovuje Vám zpracování a rozložení uživatelského rozhraní?
- Zde uživatel hodnotí přehlednost rozhraní a také jak se mu líbí celkové zpracování. Toto subjektivní hodnocení bude velmi ovlivněno objektivními odpověďmi ze scénářů uživatelského testování. Pokud uživatel nebo moci něco najít nebo některé funkce budou fungovat jinak než by očekával, bude zde samozřejmě hodnotit uživatelské rozhraní negativně bez ohledu na to, jak se mu grafické zpracování líbí.

- Co Vám v aplikaci schází a co byste ocenili?
- Obecná ale důležitá otázka na možná rozšíření z pohledu uživatele. Je možné že uživatelé mají vlastní zkušenost s podobnými aplikace a tímto způsobem lze získat nápady pro další úpravy aplikace. Samozřejmě je nutné k těmto doporučením přistupovat s odstupem a hodnotit je v souvislosti s ostatními aspekty stávající aplikace.

- Stáhli byste si tuto aplikaci do počítače nebo chytrého telefonu?
- Momentální aplikace je vyvinuta ve webových technologiích, ale oproti standardní webové stránce nemá serverovou část a je tedy volně přenosná a spustitelná i bez webového serveru. Tím se nachází na pomezí standardních desktopových a mobilních aplikací a webových stránek. Tato otázka určuje, zda je pro uživatele příjemnější takovouto aplikaci zobrazovat jako webovou stránku nebo jestli je pro ně téma dost zajímavé na to, aby si stáhli a nainstalovali aplikaci na své zařízení.

7.2.3 Uživatelé

Uživatel 1 - Uživatel standardně pracuje na počítači s kancelářskými programy, má okrajovou znalost databází a tvorby webových stránek a hraje počítačové hry. Pražský hrad je pro uživatele známým místem a poznává objekty i bez textury. Zejména ty objekty, které jsou významnější jako jsou Arcibiskupský nebo Salmovský palác. Otexturované objekty jsou pro uživatele poznatelné jednoznačně. Testovací scénáře byly bez problémů splněny.

Funkci virtuální procházky považuje za dobrou a oceňuje textové vyhledávání. Schází mu ale možnost návratu na začátek procházky a za pozitivní považuje možnost přerušení procházky. Oceňuje dále grafické efekty aplikace. Zejména efekt světla při pohledu do slunce.

Výhrady směřovaly zejména k rozložení klávesových zkratek. Uživatel by považoval za přirozené ovládat pohyb scénou přes klávesy W, S, A, D a případně i klávesami Q a E na úkroky stranou. v uživatelském rozhraní by ocenil oblast ve které se projevuje scroll v okně na celou šířku okna.

Navrhoval by doplnění modelu o bezpečnostní rámy a hlídky u jednotlivých bran, které se dnes na Hradě nachází. Dále navrhoval použít na neotexturované objekty textury s šumem v příslušné barvě právě místo jednolité barvy. Dále by považoval za vhodné, aby aplikace při prvním otevření, popsala uživateli několik základních funkcí a způsob pohybu ve scéně. Jako poslední narvhoval zobrazení názvů a šipek ve směrech blízkých objektů. Ideálně v rovině země a uživatel by tak mohl šipku následovat při volném pohybu.

Celkově se uživatel orientoval v aplikaci bez nápověd a intuitivně věděl, co kde hledat. Drobně se ztratil jen při úkolů hledání vedoucího závěrečné práce, kdy hledal v nápovědě, ale i přes ní se k informaci dostal.

Uživatel 2 - Uživatel je programátor a student 3D grafiky. Sám vytváří obdobné aplikace v OpenGL a okrajově zná i v aplikaci použité webové technologie. Pražský hrad je pro uživatele známý. Jednotlivé objekty poznává, ale opět jen ty významnější.

Virtuální procházku považuje za dobrou, ale navrhuje několik zlepšení. Za matoucí považuje zobrazení zbývající vzdálenosti a místo ní navrhuje zobrazovat posuvník, který by mohl uživatel ovládat a tím by se i mohl posouvat v trase procházky. Volné otáčení v průběhu procházky nepovažuje za dobré a ocenil by, aby se sice kamera mohla volně rozhlížet, ale pokud k tomu nedochází, tak aby se otáčela zpět do směru trasy procházky. Při výběru trasy procházky se snažil vybrat modrý bod, který byl ve skutečnosti pouze

na mapě zobrazená kašna. Možná by tedy bylo vhodné vybrat pro zobrazení bodů a trasy procházky jinou barvu. Byl by pro posunutí kamery o něco výše. a nakonec by ocenil možnost, hledání procházky z místa na kterém zrovna stojí.

Stejně jako předchozí uživatel by ocenil změnu rozložení pohybových kláves. Nápovědu by zvažoval umístit jako záložku do nastavení. Při pohybu ke vzdálenějším objektům intuitivně otevřel mapu a přemístil se přes ní. Uživatel má výhrady ke zpracování stínů, které mají na rovných plochách viditelné grafické artefakty. Naopak oceňuje efekt při pohledu do slunce a SSAO. Ocenil možnost zobrazení ostatních uživatelů a navrhoval její rozšíření. Prvním bodem by byla synchronizace času aplikace.

Uživatel se v aplikaci bez problémů orientoval a splnil i všechny testovací scénáře. Rozporuplný pro něj byl ale celkový smysl aplikace, kdy se ptal na cílovou skupinu a naznačoval, že aplikace jednoznačně necílí na turisty, pro které by mohla být zajímavá nejvíce.

Uživatel 3 - Uživatel je programátor a student 3D grafiky a vytváří obdobné aplikace v OpenGL. Pražský hrad zná, ale opět jen významné objekty, které ale poznává i v aplikaci.

Ve virtuální procházce by uživatel ocenil automatické otáčení kamery ve směru pohybu. Volné rozhlížení je dobré, ale když uživatel nechal volnou procházku běžet a kamera pak ve scéně couvala, tak to nepovažoval za dobré.

Uživatel by navrhoval ovládání pohybu stejně jako první testovací uživatel. Pohyb přes šipky byl v pořádku, ale matoucí byl právě pohyb přes písmena. Oceňuje grafické efekty aplikace a i neotexturované modely považuje za dostačující. Navrhoval by drobné úpravy uživatelského rozhraní. Zejména zavedení oblých rohů pro tlačítka a zmenšení pohybových tlačítek, které by dle návrhu nemusely mít ani pozadí.

Uživatel se v aplikaci orientoval bez problémů a ocenil minimalistické zpracování uživatelského rozhraní. Stejně tak byly bez problémů splněny uživatelské scénáře. Opět akorát ve třetím scénáři hledal uživatel jméno vedoucího nejdříve v nápovědě.

Uživatel 4 - Uživatel se živí správou serverů a počítačových sítí, hraje počítačové hry a v oblastech počítačové grafiky a technologií webových stránek má jen okrajové znalosti. Na Pražském hradě zná pouze významné objekty.

Při hledání virtuální procházky otevřel uživatel mapu a snažil se nastavit si trasu v ní. Obsah oken virtuální procházky a mapy je podobný a uživatel navrhoval jejich sjednocení.

Uživatel navrhoval nastavit cyklus dne a noci tak, aby odpovídal přesně nějakému dni v roce. Navrhoval také, aby se čas synchronizoval mezi uživateli, když už aplikace obsahuje zobrazení ostatních uživatelů. Mezi uživateli také navrhoval umožnit komunikaci a jiné interakce. Ocenil možnost celoobrazového režimu a schování tlačítek na hlavní obrazovce.

Grafické zpracování považoval uživatel za dostatečné a oceňoval zejména rozsah modelu. Také ocenil grafické efekty. Neotexturované objekty jako dokreslení prostředí připadaly uživateli ideální. Obdobnou aplikaci by si na své zařízení nestáhl, ale uporoznil, že uživatelské rozhraní by se muselo radikálně změnit, pokud by se aplikace měla zobrazovat na menších zařízeních.

V aplikaci se bez problému orientoval, ale obdobně jako ostatní otevřel při třetím scénáři nejdříve nápovědu. Uživatelské scénáře proběhly jinak v pořádku.

Uživatel 5 - Uživatel používá na počítači zejména kancelářské programy a webový prohlížeč. Nemá žádnou praxi v programování ani přehled ve webových technologiích. Naopak zná dobře Pražský hrad a poznával většinu objektů. Problém měl akorát poznat baziliku sv. Jiří.

Virtuální procházku ovládal intuitivně, ale trvalo nějaký čas než nalezl, kterým tlačítkem se nabídka virtuální procházky otevírá. Navrhoval, aby procházka měla nastavení rychlosti a to ideálně i v jejím průběhu.

V aplikaci uživatel ocenil grafické efekty, a ptal se, jestli noční osvětlení odpovídá rozmístění reálných světel na Pražském hradě. Nerozumněl popisku tlačítka na přepínání celoobrazového režimu.

Uživatel navrhoval přemístit minimapu do pravého dolního rohu, kde byla původně v předchozí aplikaci. Byl překvapen skokem, který je tak vysoký, že se uživateli pletl s módem létání.

Uživatel 6 - Uživatel je student fakulty stavební, ovládá vytváření 3D modelů, ale s webem ani tvorbou grafických aplikací nemá žádnou zkušenost.

Virtuální procházku ovládal uživatel intuitivně, ale chvíli trvalo než nabídku nalezl. Tlačítka hlavní obrazovky nebyla pro uživatele příliš jasná, a podle ikon se v nich naprosto neorientoval. Po projetí celé nabídky a přečtení textů jednotlivých tlačítek už to bylo lepší. Samotné ovládání virtuální procházky už šlo uživateli bez problémů.

Aplikaci uživatel ovládal přes talčítka směrů na hlavní obrazovce a klávesové zkratky šipek použil až v osmém scénáři, kde k tomu byl vyzván. Mód létání uživatel nevyužíval a při snaze dostat se na střechu si uživatel akorát přepnul pohled na pozici nad scénou.

V aplikaci uživatel velmi preferoval otexturované modely před neotexturovanými a i doporučoval doplnit textury ke všem modelům.

Uživatel navrhoval scénu oživit zobrazením návštěvníků Pražského hradu, ale zas by jich nemělo být moc.

Uživatel 7 - Uživatel je absolvent studia počítačové grafiky a nyní se živí tvorbou webu. Rozumí tak oběma zásadmín částem této závěrečné práce.

Virtuální procházku ocenil uživatel jako zajímavou funkci, ale pro pohyb ve scéně raději preferoval ovládání šipkami. To bylo v některých chvílích pro uživatele matoucí a stěžoval si stejně jako ostatní uživatelé zejména na klávesy WSAD, které dělaly něco jiného než by přirozeně očekával.

Z grafických efektů ocenil střídání dne a noci a také animaci vody v kašnách. Oba tyto efekty jsou dohromady jediným oživením jinak statické scény. Navrhoval proto, aby se do scény přidalo více takovýchto efektů. Například vítr, ptáci nebo jiní návštěvníci prostřednictvím simulace davu.

Uživatel testoval aplikaci na zařízení s dotykovou obrazovkou a směr kterým se kamera otáčela při používání dotykové obrazovky mu přišel nepřirozený a opačný. Doporučoval tedy, aby se kamera chovala odlišně pro uživatele s myší a uživatele s dotykovou obrazovkou. Navrhoval také aby se kamera o kus zvedla od země.

Celkově se uživatel v aplikaci orientoval a rozmístění uživatelského rozhraní pro něj bylo intuitivní. i zpracování scény hodnotil kladně, ale doporučoval její oživení drobnými objekty s kterými by se dalo interagovat nebo už zmíněmými efekty. Neotexturované objekty mu přišly v pořádku a jejich vzhled považoval za cílený způsob zpracování.

Uživatel 8 - Antonín Smrček - Antoní smrček je autor předchozí závěrečné práce. Zároveň svolil k uživatelskému testování této a poskytl mi rozsáhlou zpětnou vazbu, za kterou mu děkuji.

Ocenil především rozsah modelu a detail geometrie u Katedrály sv. Víta. Také ocenil různé drobnější detaily jako schody a průchody, ploty a také tekoucí vodu. Zároveň by uvítal, kdyby i katedrála dostala korektní texturu, protože ji považuje za důležitý objekt, který by sám o sobě aplikaci hodně pozvedl, pokud by byl dokonale zpracovaný.

Ocenil také možnost virtuální procházky a velké množství bodů mezi kterými se dá procházka nastavit. Obecně měl z aplikace dobrý dojem, který byl ale narušován množstvím menších nedostatků.

V menu se mu nelíbilo použití základních HTML prvků pro radio buttony, scroll bar a dropdown, které považoval za zastaralé. Ocenil nastavení rychlosti času, ale popisek formou násobného nastavení rychlosti nepovažoval za dobrý a přišel mu matoucí. Byl by raději, aby byl popisek formulován jako čas, za který v aplikaci uběhne 24 hodin.

Při otáčení ovatara šipkami se avatar zároveň posouvá do strany ve směru otáčení. To mu přišlo zvláštní a spíše by byl pro, aby avatar zůstával při samotném otáčení na místě.

U velké mapy ocenil zobrazení pohledových bodů v podobě modrých teček a zobrazení jmen bodů při najetí myši. Považoval by za dobré, aby bylo možné zobrazit jména všech bodů skrz nějaké nastavení. Popisky by se u většího množství blízkých bodů zarovnávaly tak, aby se nepřekrývaly. Použil by nějakou jinou barvu teček než modrou, protože se mu v jednom případě zaměnil bod za kašnu v Rajské zahradě. Názvy bodů, které jsou blízko pravého okraje se ztrávejí za okrajem okna a nelze je přečíst a v oblastech s vyšší hustotou bodů, se některé body zobrazují přes popisek jiného bodu.

Dále měl několik poznámek k uživatelskému rozhraní. Intuitivně chápal pozicování rámečku u tlačítek šipek, ale na tmavších pozadích, kde rámeček není příliš vidět, se mu tento způsob zobrazení nelíbil. Očekával by doplnění nabídky pohledů podle bodů zvýrazněných na velké mapě nebo alespoň větší seznam těch zajímavějších. Ocenil zobrazení aktuálního času a možnost přepínání na cleou obrazovku. Velmi se mu pletly ikony mapy a zajímavostí, které by preferoval zpracovat jiným způsobem, aby je nešlo tak snadno zaměnnit. Kromě těchto dvou považoval ikony za dostatečně názorné a po chvíli používání pochopitelné. Ocenil by, aby se při kliku na okraj obrazovky otevřená okna zavírala. U nápovědy by ocenil podrobnou aktualizaci k novým funkcím.

U informací o objektech by doporučoval používat různé barvy. Nyní jsou všechny kliknutelné objekty zvýrazněny modrou barvou a není poznat, kde jeden končí a druhý začíná. Odhalil také chybně nastavené popisky objektů, které jsem přehlédl. Doporučoval by, aby byly pro všechny popisky objektů použity ilustrační obrázky.

U virtuální procházky by ocenil zvýraznění bodů, které se ukazují jako zajímavosti ve velké mapě. Ocenil by tlačítko pro odstranění posledního přidaného bodu, protože mu chvíli trvalo pochopit, že se body dají odstranit kliknutím. Zároveň by ocenil stručné informace jak procházku zadávat přímo v obrazovce jejího nastavení. Také by ocenil, aby se body v mapě zvýrazňovaly už při najetí myší na položku v nabídce textového vyhledávání a vadilo mu, že se při textovém vyhledávání pole nevymazalo, poté co si nějaký bod vybral. Ocenil by, aby šlo udělat okružní procházku, která skončí na stejném místě jako začala. Zároveň se mu líbilo vyhledávání bodů podle názvů. Také možnost jejího přerušení a pokračování a také zobrazení informace o vzdálenosti a času. Ocenil by, kdyby se u informace o vzdálenosti ještě objevoval popisek, co konkrétně ta čísla znamenají.

Nakonec přidal pár dle něj ne příliš podstatných připomínek, které by ale aplikaci mohly dále vylepšit. Okolí hradu by doplnil hrubým terénem, který už i v předchozí závěrečné práci byl, aby hrad nestál uprostřed ničeho. Mramorovou texturu z prvního nádvoří by změnil na jinou. Stáhnul by sílu normálového mapování, které u některých materiálů nepůsobilo přirozeně. Ve scéně chybí průchod mezi 1. a 2. nádvoří vpravo od Matyášovy brány a také průchod mezi 3. nádvořím a Rajskou zahradou. Také by ocenil lepší texturování vnitřních částí Matyášovy brány.

Uporoznil také na pár textových bugů, které byly záhy opraveny.

7.2.4 Shrnutí testování

Uživatelé znali Pražský hrad a poznávali objekty i v aplikaci. Navrhovali doplnění modelu o bezpečností rámy a fronty. Samotný objekt Pražské hradu vyvolával v uživatelích politické konotace, které nejsou předmětem této práce, ale je důležité je vnímat při tvorbě takovéto aplikace.

Funkci virtuální procházky hodnotili uživatelé kladně, ale měli výhrady k zobrazení průběhu procházky a volnému pohybu kamery. Oceňovali grafické efekty, zejména záři při pohledu do slunce a cyklus střídání dne a noci. Efekty textur oceňovali a vnímali pouze uživatelé zběhlí v 3D grafice a ostatní se o tom sami nezmiňovali. i neotexturované budovy tak byly přijímány pozitivně a hlavní bylo jejich červené zabarvení střech, které uživatelům připomínalo skutečný pohled od Pražského Hradu. Kladně vnímána byla také možnost zobrazení ostatních uživatelů.

Otázka na instalaci na mobilní zařízení budila rozpaky a žádný z uživatelů na ní neodpověděl kladně. Směřování aplikace jako webové stránky, tak do budoucna dává větší smysl. U nativních aplikací, které by se museli instalovat, se uživatelé obecně domnívali, že takováto aplikace by neměla co nabídnout, když už teď aplikace funguje ve webovém prohlížeči.

Některým uživatelům trvalo než se zorientovali v uživatelském rozhraní. Sice oceňovali drobná tlačítka, které nepřekážejí zobrazované scéně, ale podle ikon nebylo uživatelům zcela jasné, které tlačítko, co dělá. Na místě by tedy bylo přidat do aplikace drobný návod, který by uživateli ukázal základní funkce aplikace.

Vedoucí závěrečné práce byl vždy hledán nejdříve v nápovědě a okno nápovědy a nastavení bylo zaměňováno. Bylo by vhodné okno nápovědy zahrnout do záložek okna nastavení, ale to by tím získalo jiný význam a spíše by se pak jednalo o obecné menu aplikace.

Kapitola 8
Možnosti budoucího rozšiřování

Při plánování většího rozšíření bych doporučil zvážit použití enginu Unity a jeho export do WebGL namísto rozšiřování stávajícího procesu vykreslování a zpracování scény. Za taková rozšíření považuji kosterní animace, zobrazení davu (návštěvníků), částicové efekty, kolize a podobně.

8.0.1 Unity

Unity[76] je herní engine s největším podílem na trhu. Podíl používaných herních enginů ukazuje graf 8.1. Unity podporuje tvorbu aplikací pro velké množství platforem včetně webu. Pro web Unity používá technologií WebGL a Asm a už dále není doporučeno používat původní Unity plugin, který sloužil právě pro distribuci aplikací vytvořených v Unity na web.

Aplikace vytvořená v Unity by mohla být zcela totožná jako ta, která je popisována v této a předchozí práci, ale cesta k ní je zcela odlišná. Při přenosu momentální aplikace do prostředí Unity by bylo bez problémů použít 3D model scény a šlo by použít i některé skripty, protože Unity podporuje javascript jako skriptovací jazyk. Nicméně i tyto scripty by bylo třeba upravit. Problém je zejména uživatelské rozhraní, které by bylo potřeba vytvořit kompletně znovu.

Samozřejmě Unity má mnohem větší seznam funkcí, než které jsou v této práci diskutovány natož skutečně použity. Tím se velmi rozšiřují možnosti dalších vylepšení a úprav aplikace. Kromě jiného umožňuje Unity snadné použití sledování paprsku pro kolize, virtuální realitu, fyzikální funkce, globální osvětlení apod.

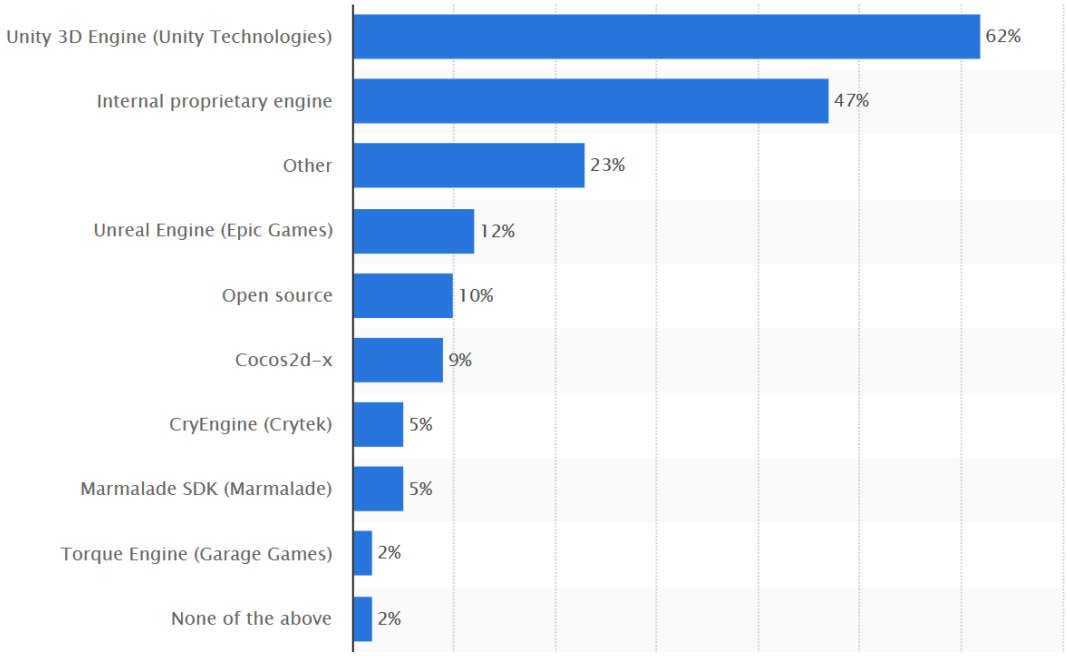
8.0.2 Electron

Electron [78] je technologie, která zjednodušuje tvorbu desktopových aplikací. Takové aplikace jsou přenosné na různé platformy (Windows, Mac, Linux) bez jakýchkoli změn v kód. Samozřejmě pokud nejsou použity platformě specifické funkce jako třeba registry operačního systému Windows.

Electron je balíček pro node.js [79] a je postaven kolem technologie V8, což je jádro prohlížeče Google Chrome. Obsah aplikace je tak vytvářen ve webových technologiích jako je javascript, HTML, CSS, Typescript a podobně. Důležité je si uvědomit, že díky začlenění do balíčků pro node.js máme k dispozici všechny ostatní balíčky a ty můžeme bez problémů použít při vytváření aplikace. Tím máme možnost vytvořit desktopovou aplikaci s podporou rozsáhlého ekosystému knihoven a nástrojů pro tvorbu webových aplikaci.

Protože byla zatím aplikace této závěrečné práce vytvářena zejména tak, aby byla přenosná, a její umístění na webový server bylo jen okrajovou částí, bylo by tedy zajímavé vytvořit plnohodnotnou desktopovou aplikaci. Právě k tomu je Electron ideální, protože umožňuje vytvořit takovou aplikaci s minimálním zásahem do stávajícího kódu.

8.0.3 Spolupráce s fakultou stavební při ČVUT



Obrázek 8.1. Graf herních enginů podle popularity ve Velké Británii za rok 2014[77]. Uživatelé mohou používat více enginů naráz a proto je součet větší než 100%.

Do budoucna by mohlo být zajímavé oslovit fakultu stavební při ČVUT, která se věnuje mimo jiné geodézii a o podobné aplikace má zájem a může nabídnout i množství zkušeností.

Zajímavá je i práce studentského klubu fakulty stavební, která zobrazuje 3D model vytištěný na 3D tiskárně[80].

Kapitola 9
Závěr

Cílem této závěrečné práce bylo rozšířit předcházející závěrečnou práci o nové funkce, 3D modely a vizuální efekty. Nejdříve byly určeny klady a nedostatky předchozí závěrečné práce a aplikace byla znovu srovnána s podobnými aplikacemi konkurenčními. Od tohoto srovnání se později odvozovala konkrétní rozšíření aplikace.

Do aplikace tak byl přidán mód virtuální procházky, který se během uživatelského testování setkal s kladnými ohlasy. Kladně hodnocená byla také možnost zobrazení ostatních uživatelů.

Po technologické stránce bylo zmenšeno množství knihoven používaných aplikací a byl doplněn moderní vývojářský nástroj Gulp, kterým se zmenšila velikost skriptů a stylů.

Bylo doplněno množství textových a mapových informací scény a samotný model scény byl značně rozšířen. Na počátku byl definován rozsah modelované scény a ten byl naplněn. Množství objektů zůstalo bez textury, ale jejich barva byla přizpůsobena tak, aby se scéna alespoň celkovým dojmem co nejvíce podobala skutečnosti. Doplněno bylo i noční osvětlení, které původně osvěltovalo pouze oblast prvního nádvoří.

Uživatelskému rozhraní byl definován jednotný styl, který byl aplikován v rámci celé aplikace. Cílem bylo zmenšit a sjednotit prvky uživatelského rozhraní a dle ohlasů z uživatelského testování se tento cíl podařilo naplnit. v rámci uživatelského rozhraní byla přidána možnost zobrazení scény na celou obrazovku a se schovaným uživateslkým rozhraním, což nechalo samotnou scénu dále vyniknout. Uživatelské rozhraní bylo doplněno drobnými animacemi a více tak nyní odpovídá standardu moderních webových aplikací.

Rožšírení byla aplikována aniž by razantné zvýšila hardwarové a softwarové nároky na plynulý běh aplikace. Aplikace tak běží s podobným FPS jako předchozí a to i s novou rozšířenou scénou Pražského Hradu a okolí.

Nakonec bylo navrženo několik možných rozšíření, které mohou aplikaci dále vylepšit.

Literatura

[1] Antonín Smrcek. *Visualization of Prague Castle*. http://leyfi.felk.cvut.cz/prague-castle-3d/. [cit. 10. 4. 2017].

[2] Ecma International. *Ecma script 5.1 - specifikace*. https://www.ecma-international.org/ecma-262/5.1/. [cit. 6. 5. 2017].

[3] Web Hypertext Application Technology Working Group. *HTML 5 - specifikace*. https://www.w3.org/TR/html5/. [cit. 6. 5. 2017].

[4] Cascading Style Sheets working group. *CSS 3 - specifikace*. https://www.w3.org/Style/CSS/specs.en.html. [cit. 6. 5. 2017].

[5] The jQuery Foundation. *Knihovna jQuery*. https://github.com/jquery/jquery. [cit. 2. 5. 2017].

[6] BuildWith. *Statistika využití jQuery na webu*. https://trends.builtwith.com/javascript/jQuery. [cit. 2. 5. 2017].

[7] Adam Schwartz a Zack Bloom. *Ukázky použití nativního javascriptu bez použití jQuery*. http://youmightnotneedjquery.com. [cit. 2. 5. 2017].

[8] Juriy Zaytsev. *Podpora ES5 v prohlížečích*. http://kangax.github.io/compat-table/es5/. [cit. 4. 5. 2017].

[9] Paul Irish a kol. *Knihovna Modernizer*. https://github.com/Modernizr/Modernizr/wiki/HTML5-Cross-browser-Polyfills. [cit. 5. 5. 2017].

[10] Google. *Databáze trendů ve vyhledávání internetového vyhledávače Google*. https://trends.google.com/trends. [cit. 5. 5. 2017].

[11] Google. *Knihovna AngularJS*. https://angularjs.org. [cit. 5. 5. 2017].

[12] Facebook. *Knihovna React*. https://facebook.github.io/react. [cit. 5. 5. 2017].

[13] Evan You. *Knihovna Vue.js*. https://vuejs.org. [cit. 5. 5. 2017].

[14] The jQuery Foundation. *Knihovna jQuery UI*. https://github.com/jquery/jquery-ui. [cit. 2. 5. 2017].

[15] Keith Wood. *Knihovna jQuery SVG*. https://github.com/kbwood/svg. [cit. 2. 5. 2017].

[16] threedubmedia. *Knihovna event.drag*. https://github.com/threedubmedia/jquery.threedubmedia/tree/master/event.drag. [cit. 2. 5. 2017].

[17] Caleb Jacob. *Knihovna Tooltipster*. https://github.com/iamceege/tooltipster. [cit. 2. 5. 2017].

[18] Hyunje Jun. *Knihovna jQuery perfect scrollbas.*
https://github.com/noraesae/perfect-scrollbar. [cit. 4. 5. 2017].

[19] Damir Sultanov. *Knihovna jQuery icheck.*
https://github.com/fronteed/icheck. [cit. 4. 5. 2017].

[20] Sara Cope. *CSS Tricks - :checked.*
https://css-tricks.com/almanac/selectors/c/checked/. [cit. 4. 5. 2017].

[21] Felix Gnass. *Knihovna jQuery spin.js.*
https://github.com/fgnass/spin.js/. [cit. 4. 5. 2017].

[22] Mr.doob. *Knihovna Three.js.*
https://github.com/mrdoob/three.js/. [cit. 4. 5. 2017].

[23] Stack Overflow. *Stack Overflow - Three.js tag.*
https://stackoverflow.com/questions/tagged/three.js. [cit. 4. 5. 2017].

[24] Jaume Sanchez. *Knihovna RStats.*
https://github.com/spite/rstats. [cit. 4. 5. 2017].

[25] Mr.doob. *Knihovna Stats.*
https://github.com/mrdoob/stats.js/. [cit. 4. 5. 2017].

[26] Jono Brandel a kol. *Knihovna dat.gui.*
https://github.com/dataarts/dat.gui. [cit. 6. 5. 2017].

[27] Trimble Inc. *ShetchUp.*
http://www.sketchup.com/. [cit. 9. 5. 2017].

[28] Matt Donley. *History of SketchUp.*
https://mastersketchup.com/history-of-sketchup/. [cit. 9. 5. 2017].

[29] Google. *Google Earth.*
https://www.google.cz/intl/cs/earth/. [cit. 9. 5. 2017].

[30] Blender Foundation. *Blender 3D.*
https://www.blender.org/. [cit. 9. 5. 2017].

[31] Foundry Visionmongers. *MARI.*
https://www.foundry.com/products/mari. [cit. 9. 5. 2017].

[32] Allegorithmic. *Substance Painter.*
https://www.allegorithmic.com/products/substance-painter. [cit. 9. 5. 2017].

[33] Blender Animation Studio. *Agent 327: Operation Barbershop.*
https://www.youtube.com/watch?v=mN0zPOpADL4. [cit. 10. 5. 2017].

[34] Blender Foundation. *Tears of Steel.*
https://www.youtube.com/watch?v=R6MlUcmOul8. [cit. 10. 5. 2017].

[35] Blender Foundation. *Sintel.*
http://archive.blender.org/blenderorg/blender-institute/durian-open-movie/index.html. [cit. 10. 5. 2017].

[36] Autodesk. *MAYA.*
https://www.autodesk.com/products/maya/overview. [cit. 10. 5. 2017].

[37] Autodesk. *3ds Max.*
https://www.autodesk.com/products/3ds-max/overview. [cit. 10. 5. 2017].

[38] MAXON. *Cinema 4D.*
http://www.cinema4d.cz/. [cit. 10. 5. 2017].

[39] James F. Blinn. *Simulation of wrinkled surfaces.* 1978. [cit. 8. 5. 2017].

[40] Lillandt Benjamin Kvarfordt Daniel. *Screen Space Ambient Occlusion.* http://www.cse.chalmers.se/edu/year/2016/course/TDA361/Advanced Computer Graphics/SSAO.pdf. [cit. 8. 5. 2017].

[41] Felkel Petr Ambrož David. *Shadows for real-time graphics. Computer Graphics 2 – Lecture 9. Czech Technical University in Prague.* https://cent.felk.cvut.cz/courses/PGR2/lectures/09-gpu_shadows.pdf. [cit. 8. 5. 2017].

[42] Joshua Koo. *webgl - sky + sun shader. Three.js examples.* http://threejs.org/examples#webgl_shaders_sky. [cit. 9. 5. 2017].

[43] DesLauriers Matt. *three-shader-fxaa.* https://github.com/mattdesl/three-shader-fxaa. [cit. 8. 5. 2017].

[44] Khronos Group Inc. *Webgl 2.0 - specifikace.* https://www.khronos.org/registry/webgl/specs/latest/2.0/. [cit. 10. 5. 2017].

[45] Mapy.cz. *Mapy.cz.* https://mapy.cz/. [cit. 12. 5. 2017].

[46] *Melown.* https://www.melown.com/maps/. [cit. 12. 5. 2017].

[47] Bohemia Interactive Simulations. *Building VBS Blue.* https://bisimulations.com/content/fri-11202015-1008/building-vbs-blue-conversation-bisim%E2%80%99s-chief-technical-officer. [cit. 12. 5. 2017].

[48] Cesium. *Cesium.* http://cesiumjs.org. [cit. 12. 5. 2017].

[49] Národní galeriev Praze. *3D modelování památkových objektů: využití v muzejní praxi a památkové péči.* http://www.ngprague.cz/program-detail/3d-modelovani-pamatkovych-objektu-vyuziti-v-muzealni-praxi-a-pamatkove-peci. [cit. 15. 5. 2017].

[50] Kateřina Čechurová. *Tvorba a vizualizace prostorového modelu Anežského kláštera v Praze.* http://geo.fsv.cvut.cz/proj/bp/2012/katerina-cechurova-bp-2012.pdf. [cit. 15. 5. 2017].

[51] Kateřina Čechurová. *Tvorba a vizualizace prostorového modelu Anežského kláštera v Praze - web.* http://geo3.fsv.cvut.cz/bp/cechurova/model.html. [cit. 15. 5. 2017].

[52] *Norwegian Institute for Cultural Heritage Research.* http://niku.no/en/. [cit. 15. 5. 2017].

[53] Root.cz. *Protokol HTTP/2 byl dokončen. Prohlížeče už ho podporují.* https://www.root.cz/clanky/protokol-http-2-byl-dokoncen-prohlizece-uz-ho-podporuji/. [cit. 17. 5. 2017].

[54] *webpack.* https://webpack.github.io/. [cit. 17. 5. 2017].

[55] *Gulp.* http://gulpjs.com/. [cit. 17. 5. 2017].

[56] Jonas Wagner. *Realtime Raytracing in Javascript!* https://29a.ch/2010/6/2/realtime-raytracing-in-javascript. [cit. 17. 5. 2017].

[57] Google. *Firebase.* https://firebase.google.com/. [cit. 17. 5. 2017].

[58] Petr Kadlec. *Schématická mapa Pražského hradu - Wikipedie*.
https://cs.wikipedia.org/wiki/Pra%C5%BEsk%C3%BD_hrad#/media/File:Prague_Castle_map.svg. [cit. 15. 5. 2017].

[59] *Institut plánování a rozvoje hl. m. Prahy*.
http://www.iprpraha.cz/. [cit. 15. 5. 2017].

[60] Institut plánováníarozvojehl.m. Prahy. *Geoportál*.
http://www.geoportalpraha.cz/cs/mapy-online. [cit. 15. 5. 2017].

[61] *OpenStreetMap*.
http://www.openstreetmap.org. [cit. 15. 5. 2017].

[62] Papírová archeologie. *Historie vystřihovánek - Pražský hrad*.
www.papirovaarcheologie.cz/fotogalerie/architektura/hrady-zamky-tvrze#prazsky-hrad-RV. [cit. 15. 5. 2017].

[63] Institut plánováníarozvojehl.m. Prahy. *3D model Prahy*.
http://www.iprpraha.cz/clanek/1308/3d-model-prahy. [cit. 15. 5. 2017].

[64] Jiří Žára, Bedřich Beneš, Jiří Sochor a Petr Felkel. *Moderní počítačová grafika*. 2004. [cit. 15. 5. 2017].

[65] NVIDIA. *NVIDIA Texture Tools for Adobe Photoshop*.
https://developer.nvidia.com/nvidia-texture-tools-adobe-photoshop. [cit. 17. 5. 2017].

[66] Python Software Foundation. *Python*.
https://www.python.org. [cit. 17. 5. 2017].

[67] Edsger W. Dijkstra. *A Note on Two Problems in Connexion with Graphs*. 1959.
http://www-m3.ma.tum.de/foswiki/pub/MN0506/WebHome/dijkstra.pdf. [cit. 17. 5. 2017].

[68] Dan's Tools. *Favicon generator*.
http://www.favicon-generator.org/. [cit. 17. 5. 2017].

[69] Nolan Lawson. *optimize-js*.
https://github.com/nolanlawson/optimize-js#javascript-api. [cit. 17. 5. 2017].

[70] @AnthumChris. *HTTP vs HTTPS Test*.
http://www.httpvshttps.com. [cit. 17. 5. 2017].

[71] Facebook. *The Open Graph protocol*.
http://ogp.me/. [cit. 17. 5. 2017].

[72] David Walsh. *Fullscreen API*.
https://davidwalsh.name/fullscreen. [cit. 5. 5. 2017].

[73] Ian Mallett, Cem Yuksel a Amit Prakash. *Adaptive deferred shading*.
http://dl.acm.org/citation.cfm?id=2876007. [cit. 17. 5. 2017].

[74] Venkat Krishnamurthy a Marc Levoy. *Fitting Smooth Surfaces to Dense Polygon Meshes*. 1996.
http://www.graphics.stanford.edu/papers/surfacefitting/. [cit. 17. 5. 2017].

[75] Tomomichi Kanek et al. *Detailed Shape Representation with Parallax Mapping*. 2001.
https://www.researchgate.net/publication/228583097_Detailed_shape_representation_with_parall
[cit. 17. 5. 2017].

[76] Unity Technologies. *Unity*.
https://unity3d.com/. [cit. 17. 5. 2017].

[77] Statista. *What game engines do you currently use?*
https://www.statista.com/statistics/321059/game-engines-used-by-video-game-developers-uk. [cit. 17. 5. 2017].

[78] Cheng Zhao a kol. *Electron*.
https://electron.atom.io/. [cit. 8. 5. 2017].

[79] Node.js Foundation. *Node.js*.
https://nodejs.org. [cit. 8. 5. 2017].

[80] ŠTUK. *3D model Prahy*.
http://stuk.fsv.cvut.cz/3d-model-prahy/. [cit. 17. 5. 2017].

Příloha A
Uživatelská příručka

A.1 Odkaz na aplikaci

Aplikace je dostupná na adrese **https://dp.jan-husak.cz**.

A.2 Seznam uživatelských funkcí

Tato sekce obsahuje seznam všech uživatelských funkcí obsažených v aplikaci a jejich krátký popis. Mnohé z nich byly již diskutovány. Tato část slouží zejména jako celkový seznam, protože některé funkce byly přidány nově a některé jsou v aplikaci obsaženy už z předchozí závěrečné práce.

- **Virtuální procházka** - Uživatel si může nastavit trasu virtuální procházky a aplikace ho po ní provede. Aplikace také zobrazuje délku a čas zvolené procházky.
- **Přednastavené pohledy** - Aplikace obsahuje seznam přednastavených pohledů. Přepínají se klávesami 1- 8 nebo jsou dostupné ve vlastním okně.
- **Mapa** - Velká mapa scény zobrazuje seznam zajímavých míst, na které se může uživatel kliknutím myši přesunou.
- **Minimapa** - Minimapa zobrazuje pozici uživatele ve scéně.
- **Zajímavé objekty** - Aplikace umožňuje zvýraznit zajímavé objekty. Na ty je možno kliknout a zobrazit tak jejich popis.
- **Létání** - Nad scénou je možné se pohybovat vzduchem.
- **Chůze** - Režim chůze umožňuje uživateli prozkoumat scénu ze stejného pohledu, jako kdyby Pražský hrad navštívil.
- **Schování uživatelského rozhraní** - Uživatelské rozhraní lze schovat a zobrazit.
- **Celoobrazovkový režim** - Aplikace podporuje celoobrazovkový režim, který překryje uživatelské rozhraní prohlížeče.
- **Uživatelské natavení** - V aplikaci je množství nastavitelných funkcí a efektů. Ty je možné upravovat z uživatelského nastavení.
- **Nápověda** - Aplikace obsahuje nápovědu k jejímu ovládání.
- **Přepínání jazyků** - Lze přepínat jazyk aplikace mezi češtinou a angličtinou.
- **Plynutí času** - V aplikaci plyne čas a střídá se zobrazení dne a noci. V noci je zobrazeno noční osvětlení.
- **Zobrazení ostatní uživatelů** - Aplikace zobrazuje ostatní uživatele, kteří používají aplikaci ve stejnou chvíli.

A.3 Klávesové zkratky

- **Pohyb šipkami**

- Šipka nahoru - posun vpřed
- Šipka dolů - posun dozadu
- Šipka vlevo - otáčení vlevo nebo úkrok vlevo
- Šipka vpravo - otáčení vpravo nebo úkrok vpravo

- **Otočení pohledu písmeny**
  - W - otočení pohledu vzhůru
  - X - otočení pohledu dolů
  - A - otáčení vlevo
  - D - otáčení vpravo
- **Úkroky**
  - , - úkrok vlevo
  - . - úkrok vpravo
  - G - přepínání úkroků a otáčení na šipkách do stran
- **Ryhlost pohybu**
  - P - zrychlení pohybu
  - O - zpomalení pohybu
- **Pohledy - klávesy přemístí uživatele na místo pohledu**
  - 1 - pohled 1 - Hradčanské náměstí
  - 2 - pohled 2 - Matyášova brána
  - 3 - pohled 3 - Kaple sv. Kříže
  - 4 - pohled 4 - Druhé nádvoří
  - 5 - pohled 5 - 4. Nádvoří
  - 6 - pohled 6 - Na baště
  - 7 - pohled 7 - Prašný most
  - 8 - pohled 8 - Pohled z nebe na celou scénu
  - V - okno se seznamem pohledů
  - R - restart pohledu - přenesení na 1. pohled
- **Létání**
  - Shift - velký skok
  - Ctrl - přepínání mezi módem létání a chůzí
  - Page Up - let vzhůru
  - Page Down - let dolů
- **Efekty**
  - S - přepínání zobrazení SSAO
  - B - přepínání zobrazení normálových a bump map
  - F - přepínání antialiasingu
  - L - přepínání efektu záře při pohledu do světla
- **Ostatní**
  - ESC - otevření menu, zavírání aktivních oken a ukončení celoobrazového režimu
  - F1 - nápověda
  - Mezerník - pozastavení virtuální procházky
  - C - přepínání detekce kolizí
  - TAB - otevírá velkou mapu

Příloha B
Příručka pro vývojáře

Tato příloha popisuje vazby dílčích částí aplikace pro usnadnění pochopení závislostí a struktur následujícími vývojáři. Jsou popsány základní postupy rozšíření aplikace o nová data.

B.1 Vygenerované pohledy

Seznam vygenerovaných pohledů je uložen v souboru src/app/Preferences.js od řádku 555. Pohled je definován pozicí, natočením a textovým klíčem. Klíč je použit pro překlad názvu pohledu a zároveň je klíš shodný s názvem obrázku pohledu, který je umístěn ve složce data/gui/snapshots.

B.2 Ovládání

Ovládání se upravuje v souboru src/control/Control.js Je zde především definován seznam kláves a jejich reakcí. Samotný pohyb uživatele je pak definován v souboru entity/Entity.js.

B.3 Mapa a minimapa

Minimapa, velká mapa a nastavení procházky používají stejnou mapu umístěnou v data/gui/map/map.png. Při změnách mapy je dobré zachovat poměr stran, jinak by se pozicování minimapy muselo změnit. To by se nastavovalo v souboru src/gui/Bigmap.js od řádku 243.

B.4 Textové stránky

Textové stránky pro objekty jsou uloženy v samostatné složce info jako jednotlivé HTML soubory. Je zde uložena i vzorová prázdná stránka, kterou doporučuji kopírovat, při doplňování stránke nových. Stránky jsou navázány na objekty přes upravené jméno objektu. Jménu jsou odstraněny tečky a pomlčky a mezery jsou nahrazeny podtržítkem. Nakonec jsou sekvence podtržítek nahrazena jedním. Npaříklad `Katedrala sv. Vita` se převede na `katedrala_sv_vita`. Soubory jsou vkládány do stránky dynamicky a samostatně nemají žádný smysl. Proto také nejde o validní HTML dokumenty s hlavičkou tagem HTML a BODY.

B.5 Obsah oken

Obsah oken je definován v samostatných HTML souborech ve složce gui. Některé části oken jsou generovány dynamicky. Tyto dynamické části jsou definovány především v skriptu src/gui/Gui.js a pro procházku ve skriptu src/gui/Tour.js.

B.6 Doplnění názvů

V rámci aplikace jsou na různých místech zobrazeny názvy a popisky míst, ulic a objektů. Ty jsou čteny z různých zdrojů, ale názvy z těchto zdrojů jsou použity jen jako klíče, kterým se dohledávají celé názvy v souboru ContentLibrary.js.

Oblasti jsou definovány v souboru `map_area.svg`. Každá oblast musí mít unikátní barvu a ID s předponou `area_`. Předpona lze změnit na něco jiného v soubotu Preferences.js přes konstantu `SVG_AREA_MAP_LOCATION`.

Při běhu aplikace je promítnuta pozice hráče do bitmapy vygenerované z `map_area.svg` a podle barvy přečtené oblasti je získáno ID oblasti, které je přeloženo skrz jazykové konstatnty v souboru ContentLibrary.js. Až to se ukazuje nad minimapou.

B.7 Doplnění textur a materiálů

Geometrie modelu oproti předchozí závěrečné práci značně pokročila na kvalitě už jen svým rozsahem. Nicméně rozsah textur zůstal původní. Pro doplnění chybějících textur je třeba mít na paměti způsob zpracování materiálů aplikací. Vše se nastavuje v souboru src/app/Preferences.js. Zejména pak v části `prefixes`.

Materiály s normálovým nebo bump mapováním musí mít texturu s příponou rovnou konstantě `_diffuse`. Díky té budou v průběhu načítání rozpoznány. To probíhá v souboru src/loaders/ColladaLoader.js. Textury pro normálové mapování mají příponu `n` nebo `_normal` a textury pro bump mapování mají příponu `b` nebo `_bump`.

Přípony lze změnit v Preference.js a v rámci jednoho materiálu není možné použít bump a normal mapování dohromady.

B.8 Generování výškové mapy

Při změně geometrie scény je vhodné nově vygenerovat výškovou mapu, protože na ní přímo závisí řešení kolizí. Výškovou mapu je možné vygenerovat při spuštění aplikace s nastavenou konstantou `GENERATE_HEIGHTMAP` v souboru Preferences.js. Pokud není generování výškové mapy povoleno, použije se naposledy vygenerovaná výšková mapa, která ovšem nemusí odpovídat změnám geometrie.

B.9 Praktické zkušenosti z vývoje

Three.js Bug: Three.js ve verzi 77 obsahuje nepříjmený bug metody distanceTo objektu Vector3. Tato metoda by měla spočítat vzdálenost bodu o jiného. Mimo to ale mění pozici bodu, nad kterým je volána a ten je fakticky posouván. Pokud se chceme tomuto problému vyhnout, je třeba bod nejdříve zkopírovat metodou clone.

```
A.clone();
A.distanceTo(B);
```

Kolizní objekty: Objekty stropů a podlah potřebují UV souřadnice, aby se zobrazily, i když nejsou na nic použity během vykreslování.

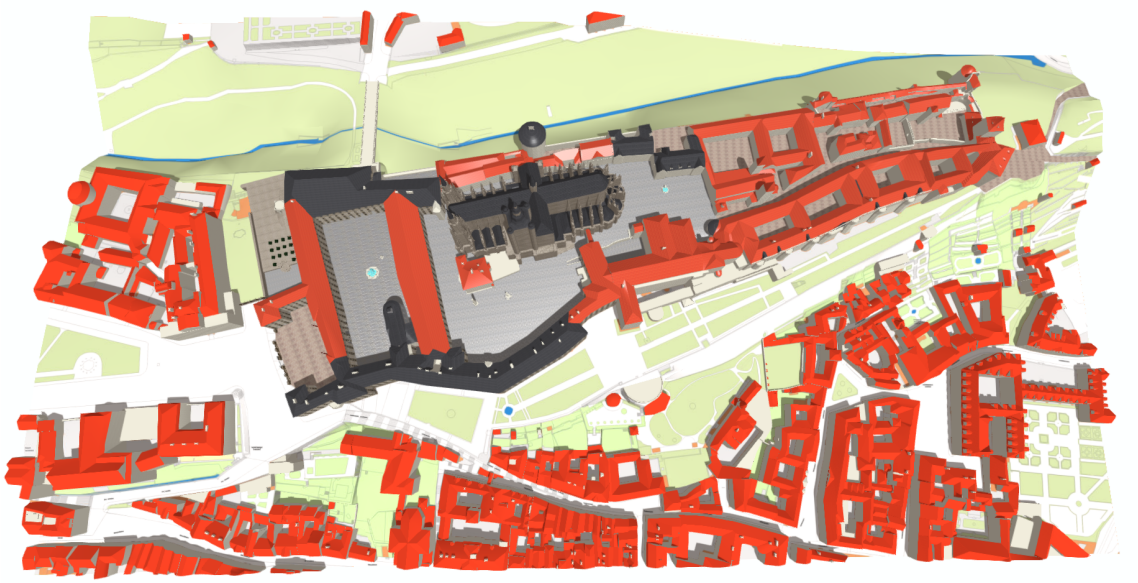
Upravené knihovny třetích stran: Některé knihovny byly v rámci předchozí závěrečné práce upraveny. To zesložiťuje jejich minimalizaci a možné nasazení nových verzí. Každá úprava je označena komentářem.

Omezení počtu světel: V aplikaci není použitý pro vykreslování osvětlení defered renderer. Ten je sice využíván pro vykreslování efektu SSAO a aplikace ho tedy obsahuje, ale osvětlení je vykreslováno bez použití vytvořeného G-bufferu.

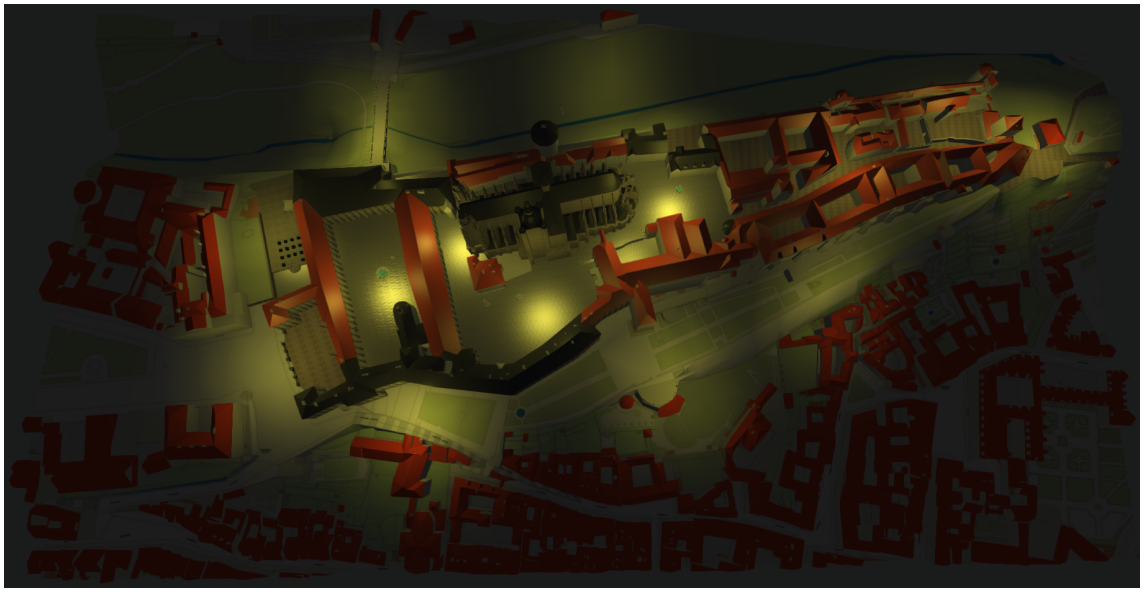
Kvůli dopřednému vykreslování je potřeba počítat s omezeným počtem světel. To se týká především světel nočních, protože ve dne je ve scéně světlo jen jedno. Větší počet světel (už kolem 15) způsobí vyčerpání varying proměnných a selhání kompilace shaderu. Některé materiály se díky tomu nezobrazí. Také to způsobuje bug při kterém se aplikace při přechodu na noc, v momentě zobrazování nočních světel, zasekne. Je to způsobeno právě selháním kompilace shaderu a logováním chyb do konzole.

Textury: U textur ve formátu PNG je automaticky předpokládána průhlednost. Pokud jsou požity na něco jiného než sprity, je skrze ně vidět efekt SSAO. Namísto PNG formátu použijte JPG a PNG formát využijte pouze pro sprity. Sprity jsou během vykreslování problémových efektů schovány.

Příloha C
Obrázky z aplikace

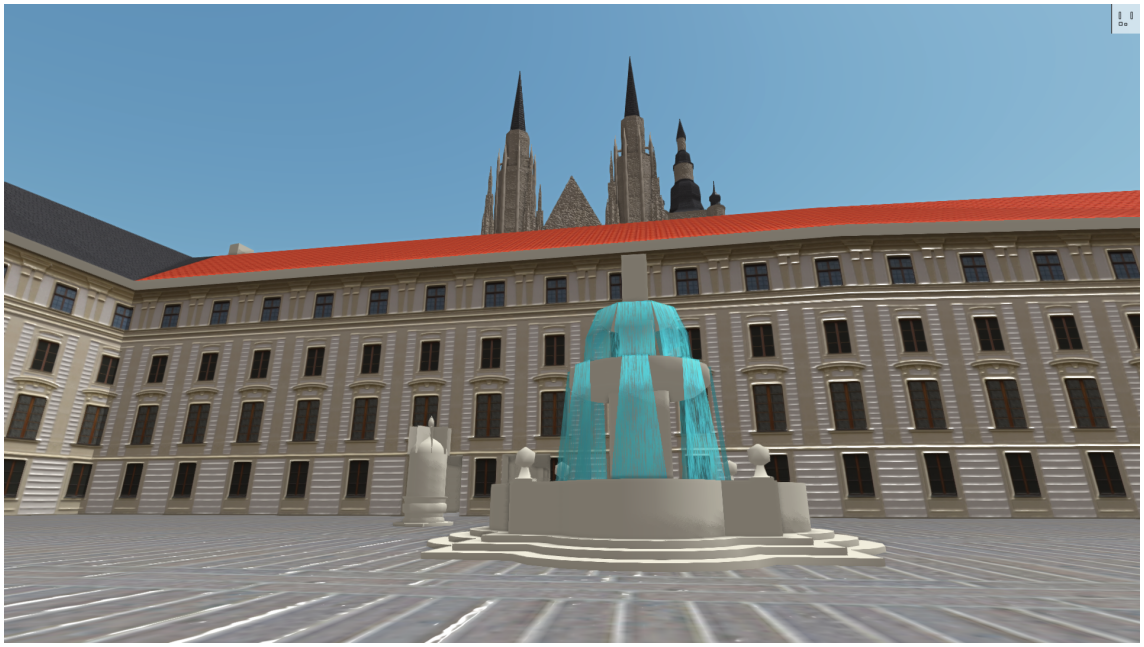


Obrázek C.1. Scéna v celoobrazovkovém režimu bez uživatelského rozhraní ve dne.

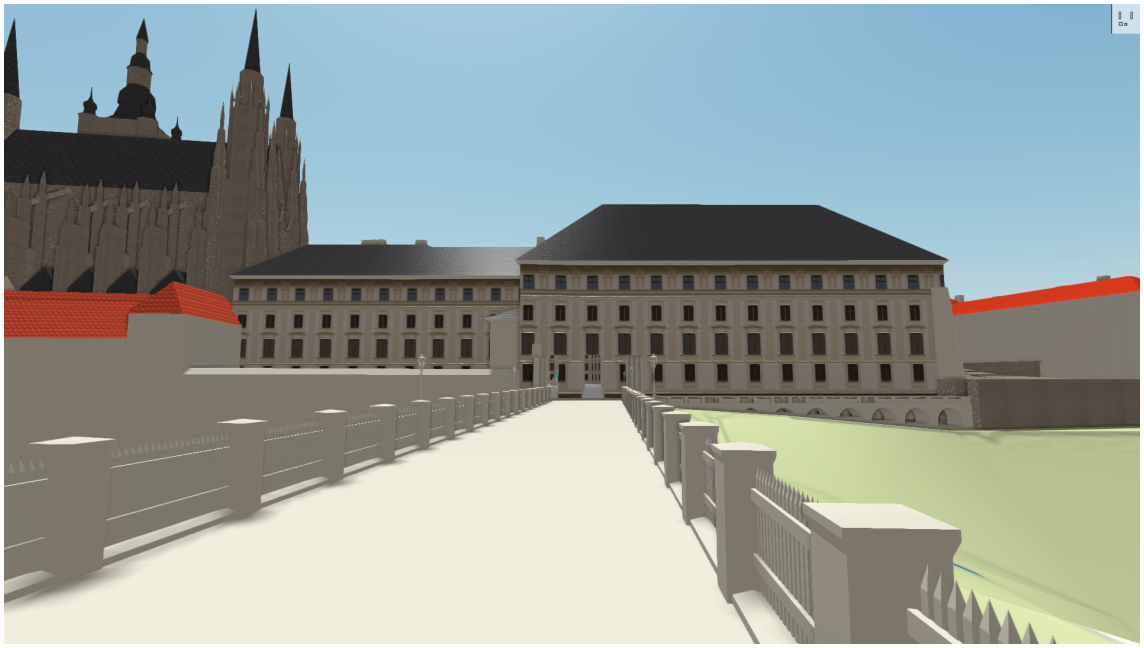


Obrázek C.2. Scéna v celoobrazovkovém režimu bez uživatelského rozhraní v noci.

Obrázek C.3. Pohled z Hradčasnkého náměstí



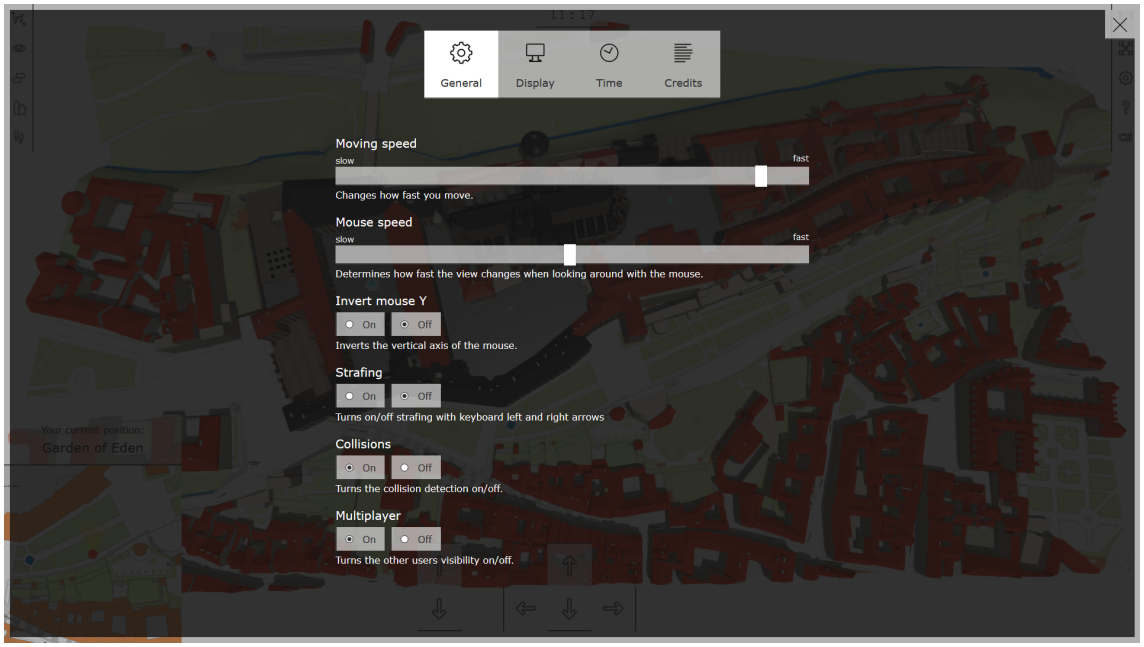
Obrázek C.4. Druhé nádvoří s Kohlovou kašnou



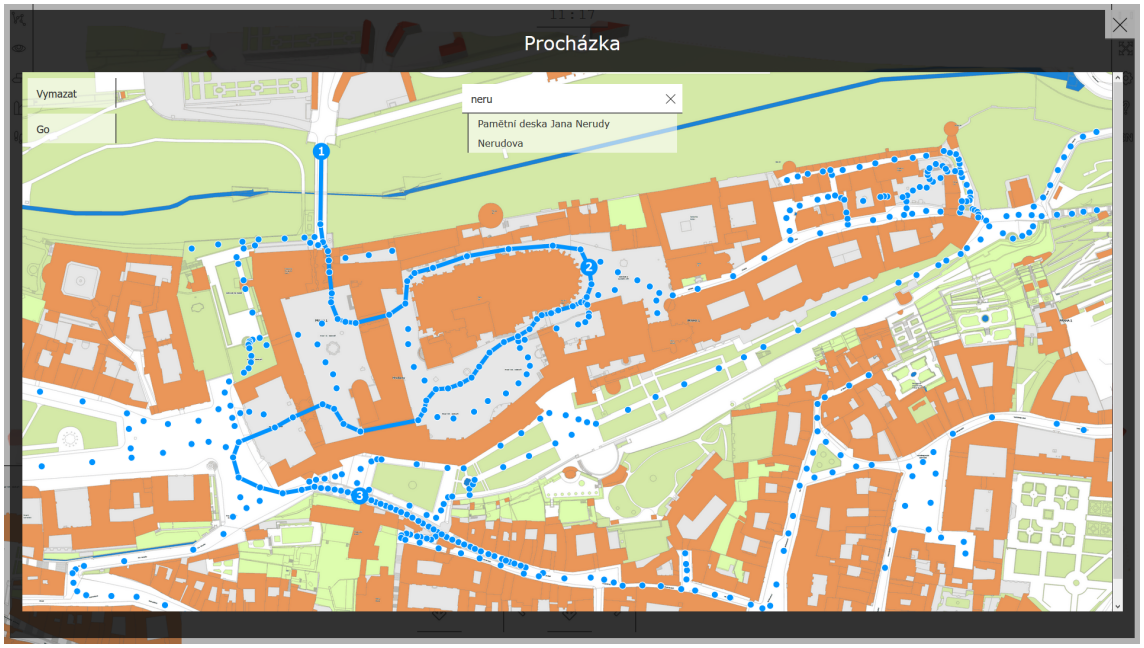
Obrázek C.5. Prašný most

Obrázek C.6. Pohled od východu

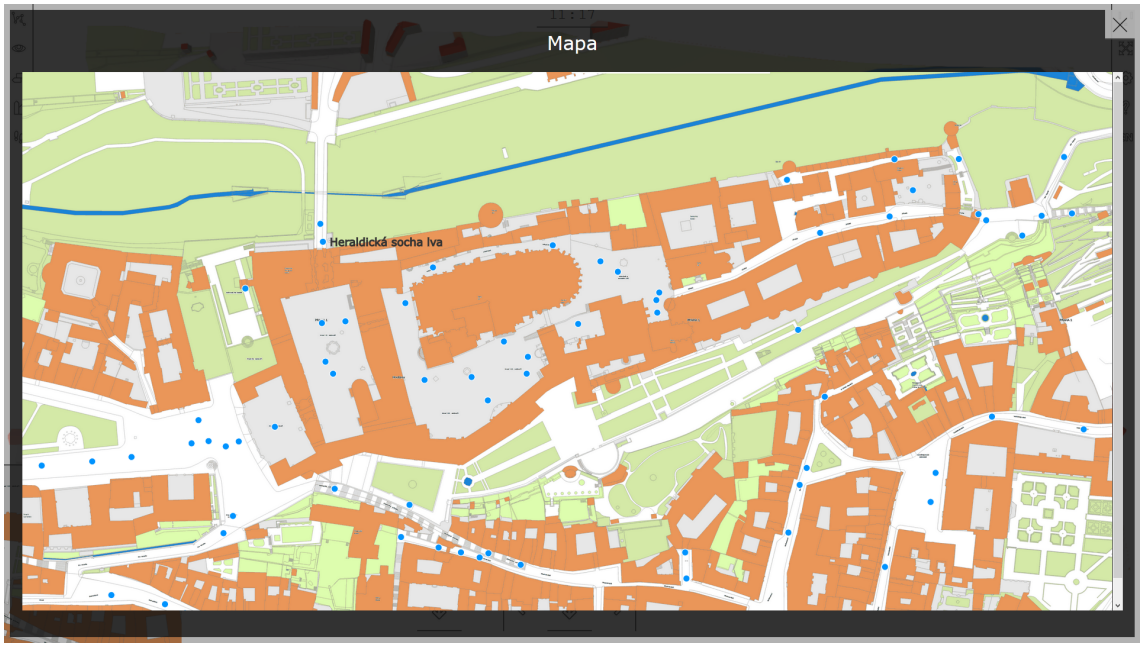
Obrázek C.7. Uživateslké rozhraní - hlavní obrazovka



Obrázek C.8. Uživatelské rozhraní - okno s nastavením



Obrázek C.9. Uživatelské rozhraní - virtuální procházka



Obrázek C.10. Uživatelské rozhraní - mapa

Příloha D
Obsah přiloženého CD

Přiložené CD obsahuje nádledující strukturu.

- application/
  - samotná aplikace, zdrojové kódy a vstupní soubor index.html
- link/
  - obsahuje odkaz na demo aplikace umístěné na internetu
- thesis/
  - `DP_Husak_Jan_2017.pdf`
    - tento text závěrečné práce.
- content/
  - Model/
    - výsledný model scény
  - Maps/
    - mapové podklady, které jsou použil při tvorbě modelu
  - 3D assets/
    - 3D podklady, které jsem použil při tvorbě modelu