České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra počítačové grafiky a interakce

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student: Bc. Kateřina Švecová

Studijní program: Otevřená informatika
Obor: Počítačová grafika a interakce

Název tématu: Zobrazování komplexní vegetace v Unreal Engine

Pokyny pro vypracování:

Zmapujte existující metody pro vytváření a zobrazování rozsáhlých vegetačních celků. Prostudujte možnosti zobrazování vegetace na platformě Unreal Engine. Vytvořte několik testovacích prostředí s různě složitou reprezentací vegetace včetně její statické a dynamické varianty. Nalezněte hranice pro interaktivní zobrazování vegetačních celků, jak co se týká kvality, tak co se týká množství zobrazovaných dat. Navrhněte metodu umožňující poloautomatické generování scén reprezentujících stejnou krajinu v různých vegetačních obdobích a za různého počasí (např. jasno a bezvětří, zataženo, silný déšť a vítr). Pomocí implementované metody vytvořte odpovídající scény pro nejméně dvě různě rozsáhlé krajiny. Výsledky zhodnoťte z hlediska dosažené vizuální kvality a z hlediska rychlosti zobrazování na různých hardwarových platformách.

Seznam odborné literatury:

[1] A. Emilien, U. Vimont, M.-P. Cani, P. Poulin, B. Benes. WorldBrush: interactive example-based synthesis of procedural virtual worlds. ACM Transactions on Graphics (TOG), Proceedings of ACM SIGGRAPH 2015.
[2] S. Pirk, T. Niese, T. Hädrich, B. Benes, O. Deussen. Windy trees. computing stress response for developmental tree models. ACM Transactions on Graphics (TOG), Proceedings of ACM SIGGRAPH Asia 2014, 2014.
[3] B. Benes, O. Deussen, S. Pirk, B. Chen, R. Mech, T. Ijiri. Modeling plant life in computer graphics. ACM SIGGRAPH 2016 Courses, 2016.
[4] A. Kučera. Realistické zobrazování modelů vegetace v reálném čase. Diplomová práce ČVUT FEL, 2011.
[5] B. Bedřich, J. M. Soto Guerrero. Clustering in virtual plant ecosystems. In proceedings of WSCG, 2004.

Vedoucí: Doc. Ing. Jiří Bittner, Ph.D.

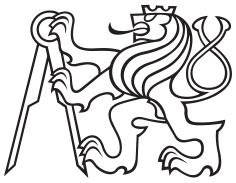
Platnost zadání: do konce zimního semestru 2018/2019

prof. Ing. Jiří Žára, CSc.
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 29.3.2017

Diplomová práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická
Katedra počítačové grafiky a interakce

Zobrazování komplexní vegetace v Unreal engine 4

Kateřina Švecová

Vedoucí: doc. Ing. Jiří Bittner, Ph.D.
Obor: Počítačová grafika a interakce
Studijní program: Otevřená informatika
Květen 2017

Poděkování

Děkuji svému vedoucímu práce doc. Ing. Jiřímu Bittnerovi, Ph.D. za podnětné rady při vedení práce. Děkuji rodičům za finanční a psychikou podporu. Děkuji celé rodině, příteli a kamarádům, že se mnou měli trpělivost a podporovali mě během studií.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracovala samostatně, a že jsem uvedla veškerou použitou literaturu.

V Praze, 26. května 2017

Abstrakt

Tato práce se zabývá zobrazováním komplexní vegetace v Unreal Engine. V práci jsou popsány obecné techniky zobrazení a generování vegetace. Dále se práce zabývá zobrazováním vegetace v Unreal Engine 4. Výsledkem práce je popsané testování hranic kvality a kvantity zobrazení vegetačních celků v Unreal Engine 4 a nástroj na poloautomatické generování krajiny v různých ročních obdobích.

Klíčová slova: Unreal Engine 4, UE4, komplexní vegetace, roční období, polo-automatický nástroj, zobrazování

Vedoucí: doc. Ing. Jiří Bittner, Ph.D.
Praha 2, Karlovo náměstí 13, E-421

Abstract

This thesis is focused on display of complex vegetation in Unreal Engine. There are common techniques of generation and displaying described in the thesis. Next the thesis deals with displaying of vegetation in Unreal Engine 4. Described testing of limits between quality and quantity of displaying of vegetation complexes in Unreal Engine4 and the tool for semiautomatic generation of landscape in different seasons are the result of the work.

Keywords: Unreal Engine 4, UE4, complex vegetation, season, semi-automatic tool, render

Obsah

Úvod

Zobrazování vegetace a rozsáhlých vegetačních celků je v počítačové grafice vyžadováno hned v několika odvětvích. Každým rokem se také zvyšují nároky a požadavky uživatelů, kteří se nespokojí s osekaným modelem, ale chtějí vidět vegetaci co nejvíce podobnou té reálné. Aby vegetace vypadala reálně, je třeba velmi složitého modelu. Například u stromu jsou zapotřebí, kromě vymodelovaného stromu, také textury kmene, větví, listů, jejich normálové mapy a správně nastavené texturovací souřadnice. Pro rozsáhlé vegetační celky je třeba takových modelů několik stovek až tisíců. Zde však narážíme na omezení hardwaru, a proto se potýkáme s vysokou paměťovou a/nebo časovou náročností. Díky různým technikám a přístupům k zobrazování a generování vegetace lze zjednodušit scénu tak, aby se dojem uživatele příliš nezhoršil. Tyto techniky vycházejí především z omezení lidského vnímání. Využívá se zobrazení vegetace na různých úrovních detailu. Pro objekty blízké pozorovateli je zapotřebí detailní model, protože je člověk schopen vnímat jednotlivé listy. Čím jsou objekty vzdálenější, tím méně je schopno lidské oko vidět detaily. A právě této vlastnosti se využívá při zobrazování modelů v počítačových hrách, kde se připraví několik modelů s různými detaily a tyto modely se ve scéně vyměňují v závislosti na vzdálenosti od pozorovatele. Pro velmi vzdálené objekty lze dokonce nahradit model pouze jedním statickým obrázkem. Nejenže je omezen vjem lidského zraku, ale zároveň jsou limitována i výstupní zařízení počítače (konečné rozlišení obrazovky).

V této práci jsou v první kapitole rozebrány techniky generování a zobrazení vegetace, podrobněji jsou uvedeny metody LOD, Billboards a generování vegetace pomocí L-systémů. V druhé kapitole jsou popsány obecné charakteristiky herních engine. Následuje popis Unreal Engine. Jsou uvedeny přístupy, jak engine využívat, popis komponent, šablon her, vztahy mezi třídami a herní režimy. Třetí kapitola se zabývá modely vegetace a nástroji na tvorbu a manipulaci s vegetací v Unreal Engine 4. Nejdříve je popsáno obecné použití a funkčnost jednotlivých nástrojů, následně jsou uvedeny konkrétní ukázky použití. Ve čtvrté kapitole jsou shrnuty dosažené výsledky. Je popsán polo-automatický nástroj na ovládání ročního období, jeho použití a ukázky z krajin, kde byl nástroj použit. Dále jsou popsány přístupy k testování a tvorba testovacíh scén. Výsledky testů jsou shrnuty v tabulkách a zobrazeny v grafech a dále popsány v diskuzi naměřených dat. Následně je popsána možnost interagovat s prostředím a vegetací v Unrela Engine 4. Jedná se o chození ve sněhu se zanecháváním stop a manipulaci s vegetací za běhu hry (vkládání, mazání, kácení a sekání).

Kapitola 1

Generování a zobrazení vegetace

Tato sekce se zaměřuje na metody zjednodušení scény, pomocí kterých se výazně sníží nároky na výkon počítače, ale kvalita zobrazení zůstane dostatečná. Při tom se využívá vlastností lidského vnímání, kdy mají některé části scény malý vliv na celkový vjem pozorovatele (například vzdálené objekty). Podrobně jsou uvedeny techniky stupňů *LOD* a použití *Billboards*. Další část se zabývá možnostmi tvorby a generováním vegetace.

1.1 Zjednodušení scény

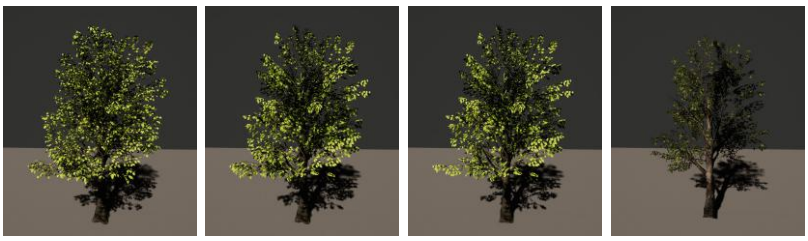
Zobrazování vegetace není snadný úkol. Na jednu stranu je požadováno, aby byla vegetace velmi realistická, zároveň však chceme poměrně často zobrazovat rozsáhlý vegetační celek. Nejsložitější je přitom zobrazování stromů, jelikož se jedná o objekt složený z kmene, větví, ale především několika tisíců až desetitisíců listů (vzrostlý dub může mít okolo 200 000 listů), které se v reálném světě pohybují vlivem větru. Naivní přístup, kdy máme jeden list jako samostatný objekt, není v současné době možný (omezuje nás hardware). Je třeba si také uvědomit, že vegetace slouží často k dotváření prostředí a vjemu scény, ale není tím hlavním, a tudíž by neměla příliš vytěžovat prostředky počítače. Asi nejznámějším příkladem jsou počítačové hry. Jednou z možností je se uskromnit a mít ve hře pouze pár stromů. V dnešních počítačových hrách se ale těžko spokojíme s tímto omezením a je vyžadováno zobrazení mnoha instancí. Proto existují techniky, jak přesvědčit uživatele, že se dívá na realistickou krajinu, přestože má reprezentace vegetace v této krajině od té reálné velmi daleko. Tyto techniky se snaží zredukovat složitost

modelu, ale přitom co nejvíce zachovat vjem uživatele a nějakým způsobem ho „ošálit“. Vybrala jsem si pro bližší popis dvě metody, které se často používají právě u zobrazení vegetace. Tou první je *Level Of Detail* (dále pouze *LOD*), či-li *„Úroveň detailu“* a druhou je použití obrázků tzv. *„Billboards“*.

1.1.1 Stupně LOD

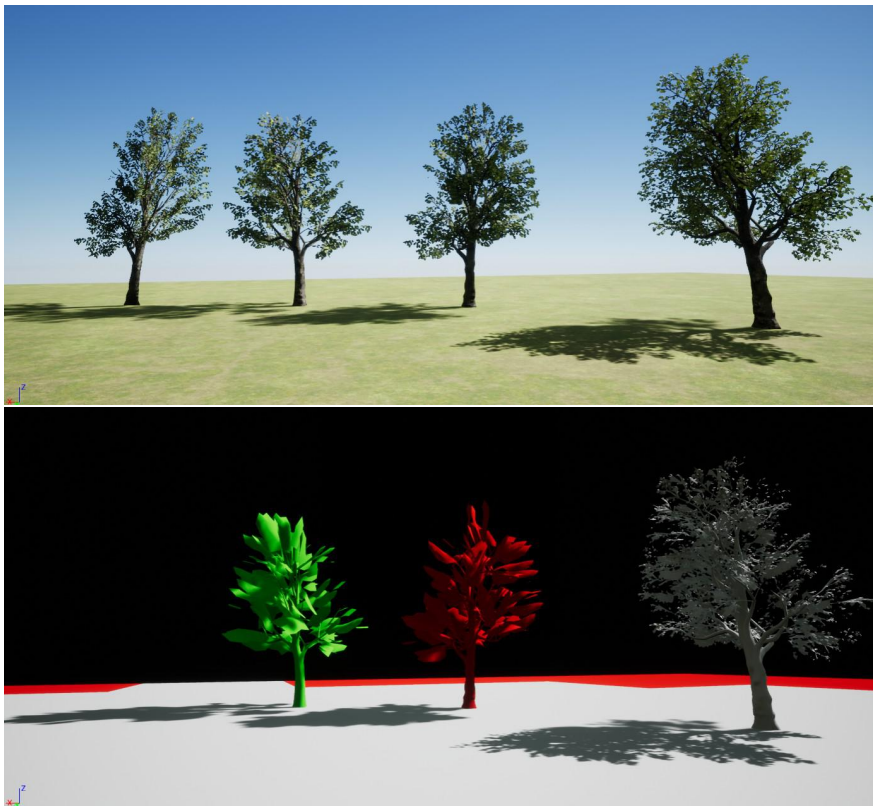
Stupně LOD využívají vlastnosti lidského zraku. Pokud člověk stojí blízko stromu, vidí všechny jeho detaily, jako strukturu kmene a jednotlivé listy. Čím dále od stromu (obecně jakéhokoliv objektu) stojí, tím méně vnímá jeho detaily. LOD pracuje s více modely, které mají různý počet detailů a různý počet trojúhelníků.
Stupně LOD lze z hlediska reprezentace scény rozdělit do dvou částí, spojité a diskrétní stupně LOD. Spojité stupně LOD provádí malé změny v datové struktuře a po malých krocích (přidání či odebrání trojúhelníku) se postupně mění stupeň detailu. Tento způsob ve své práci nepoužívám, a proto zde uvádím odkazy na metodu postupné trojúhelníkové sítě (viz. [4]) a metodu uložení všech trojúhelníků úrovní najednou (viz. [5]) a dále se spojitým LOD nezabývám.
Pro vegetaci a zejména stromy je využíváno diskrétní LOD. To znamená, že se pro model stromu předpřipraví několik (zpravidla 3 až 5) jeho variant, kdy postupně klesá množství detailů (zpravidla počet trojúhelníků). Následně se tyto modely vyměňují v závislosti na daném kritériu. Tyto modely lze vytvořit samostatně, jako několik rozdílných a navržených modelů (viz obr. 1.1), nebo lze použít nástroj na decimaci trojúhelníkové sítě. Při použití decimace stačí sice jeden model, ten nejdetailnější, ze kterého se ubírají detaily, ale u složitějších modelů nemusí výsledek vypadat podle očekávání. Algoritmy na decimaci samozřejmě nepracují náhodně, ale zkoumají důležitost trojúhelníků pro model. i přesto zde může dojít ke špatným výsledkům. Jako ukázku jsem zvolila decimační nástroj v UE4 a použila ho na model stromu (viz obr. 1.3). Jelikož se jedná o velmi jednoduchý strom, došlo k deformaci kmene (viz obr. 1.7), což vytváří rušivý efekt a výsledek nepůsobí vůbec dobře. Na druhou stranu u některých objektů metody decimace dávají kvalitní výsledky (viz obr. 1.5).

Jednotlivým modelům vegetace se přiřadí příslušný stupeň, podle snižujících se detailů, přičemž nejvyšší stupeň LOD obsahuje všechny detaily. Stupeň detailu se mění v závislosti na měnících se podmínkách ve scéně. Nejčastěji se jako kritérium používá vzdálenost pozorovatele od objektu (ukázka na modelech stromu viz obr. 1.6). Mezi další patří například způsob lidského vnímání, kdy se detailněji zobrazují objekty ve středu obrazovky a méně detailněji ty po stranách.



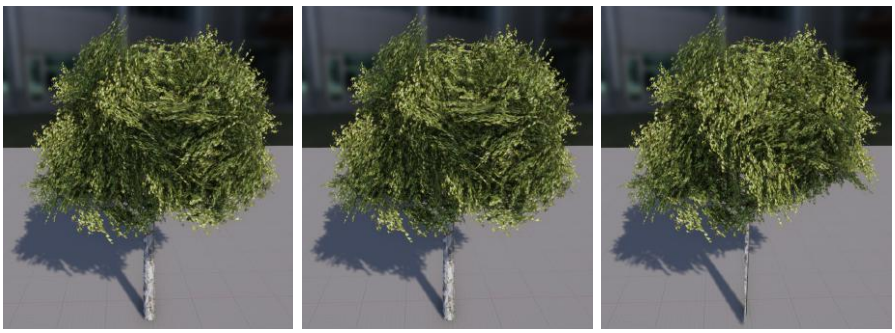
(a) (b) (c) (d)

Obrázek 1.1: Různé úrovně detailu stromu v Unreal Engine. (a) LOD0, počet trojúhelníků: 58 219. (b) LOD1, počet trojúhelníků: 4317. (c) LOD2, počet trojúhelníků: 2849. (d) LOD3, počet trojúhelníků: 32



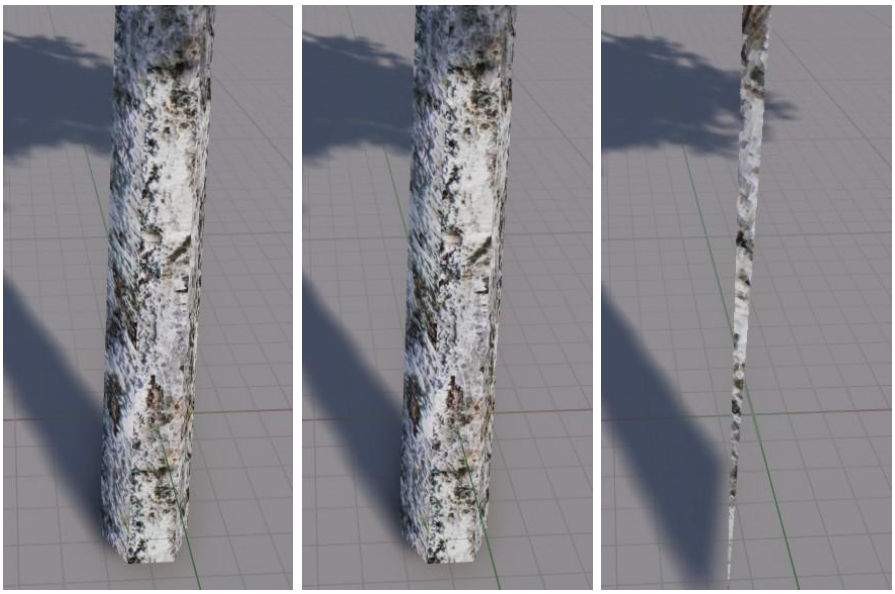
Obrázek 1.2: Stejný pohled do scény s různými viewmode. Nahoře klasické zobrazení. Dole obarvení LOD, LOD0: šedá, LOD1: červená barva, LOD2: zelená barva

Každý stupeň má tedy přiřazen údaj, kdy má nastat změna. Při této změně se vymění jeden model za druhý podle příslušného LOD. Při této výměně nastává zpravidla diskrétní skok, kdy některé části objektu „vyskočí“ (*popping effect*), nebo naopak zmizí. Pokud se uživatel na takový model přímo dívá, působí tento skok velmi rušivě. Tento nežádoucí jev lze zjemnit pomocí technik jako



(a) (b) (c)

Obrázek 1.3: Decimace jednoduchého stromu z jednoho modelu pomocí nástroje v Unreal Engine 4. (a) LOD0, počet trojúhelníků: 1192, původní model. (b) LOD1, počet trojúhelníků: 833, redukce na 70% trojúhelníků. (c) LOD2, počet trojúhelníků: 357, redukce na 30% trojúhelníků



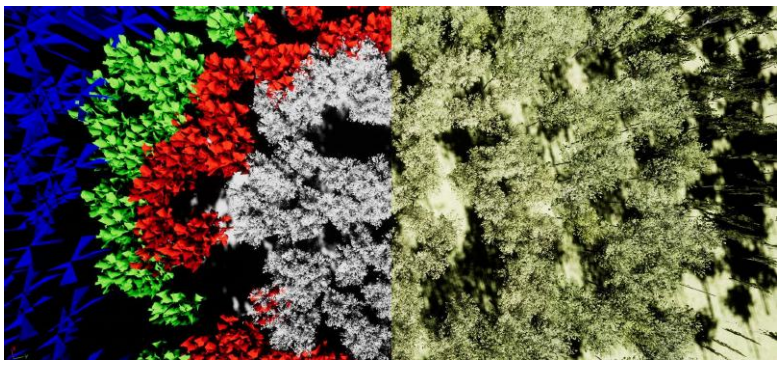
(a) (b) (c)

Obrázek 1.4: Problém s kmenem při automatické decimací stromu. (a) LOD0, původní model. (b) LOD1, redukce na 70% trojúhelníků, vše v pořádku. (c) LOD2, redukce na 30% trojúhelníků, nepřirozený tvar kmenu

je interpolace [13], *blending* [14] a *morphing*. Při interpolaci se dopočítávají snímky mezi snímkem před změnou a po přepnutí LOD. Technika blending používá míchání sousedních LOD. Při morphingu se plynule mění tvar jednoho modelu a transformuje se do druhého.

(a) (b) (c)

Obrázek 1.5: Decimace kamene z jednoho modelu pomocí nástroje v Unreal Engine. (a) LOD0, počet trojúhelníků: 1228, původní model. (b) LOD1, počet trojúhelníků: 858, redukce na 70% trojúhelníků. (c) LOD2, počet trojúhelníků: 368, redukce na 30% trojúhelníků



Obrázek 1.6: Pohled kamery ze shora na les. V levé části jsou obarveny LOD. V pravé části je klasický pohled. LOD0: šedá, LOD1: červená barva, LOD2: zelená barva, LOD3: modrá barva - jsou vidět jednotlivé billboardy

1.1.2 Billboards

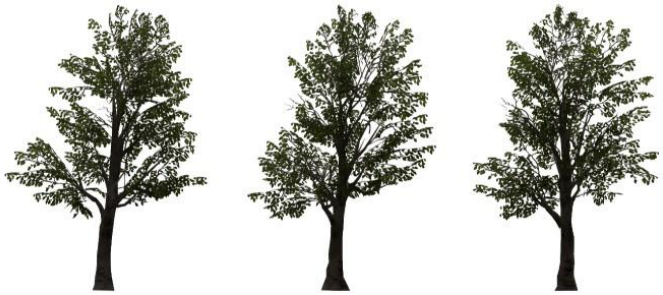
Metoda použití *billboards* zachází ve zjednodušení mnohem dále a celý 3D model nahradí 2D obrázkem. Vytvoří se textura (trávy, větve, stromu,...), která se aplikuje na jednoduchý polygon, zpravidla obdélník (viz obr. 1.9). Ta část textury, kde není zobrazen náš objekt (například list) je zcela průhledná. Obrázek je pevně umístěn na dané místo ve scéně a při zobrazování se může natáčet, aby byla rovina kolmá k pohledu uživatele. Díky tomu vidí uživatel obrázek správně a nestane se tak, že by spatřil polygon z boku nebo velmi zkresleně. Obrázek, který se vždy natáčí se nazývá *Billboard* [2]. Obrázek, který se nenatáčí se označuje jako *Sprite*. V obou případech jsou polygony připraveny předem, takže je výpočetní náročnost zobrazení velmi nízká. Díky

tomuto zjednodušení dochází k velkému zrychlení zobrazení oproti použití složitého modelu.
U vegetace lze tato metoda využít jak pro celý model (strom), tak pro jeho části. Často se využívá ke seskupení listů. Tímto problémem se zaobírali autoři v článku *Fast rendering of leaves* [6]. Další možností je použít více obrázků pro jeden model. V nejjednodušším případě se jedná o dva obrázky, které jsou na sebe kolmé. Je však možné vytvářet trsy billboardů (viz. [7] a [8]). Trsy billboardů nahrazují původní modely množinou různě orientovaných obrázků. Jednotlivé obrázky v trsech se narozdíl od samostatného billboardu nenatáčí směrem k pozorovateli, ale zachovávají si svoji orientaci. Speciálním případem použité trsů je přidání rovnoběžných obrázků. Pro každý obrázek z trsu, se ještě přidá několik rovnoběžných obrázků. U samotných billboardů nastává problém s paralaxou, která zde chybí kvůli natáčení billboardu. U trsů rovnoběžných billboardů lze tento efekt simulovat použitím průhledných obrázků. Obrázky jsou navrženy tak, aby co nejvěrohodněji napodobily původní geometrii. Tímto nahrazním původního modelu za trs billboardů se zabývají autoři v článku *Adaptive Billboard Clouds for Botanical Tree Model* [20]. Podobně jako pro složitý původní model šlo vytvořit několik LOD, lze tyto úrovně využít i v případě trsů billboardů. Trsy, které se zobrazují blíže, budou mít více obrázků a tedy i detailů, vzdálenější objekty mohou být reprezentovány pouze dvěma kolmými obrázky.

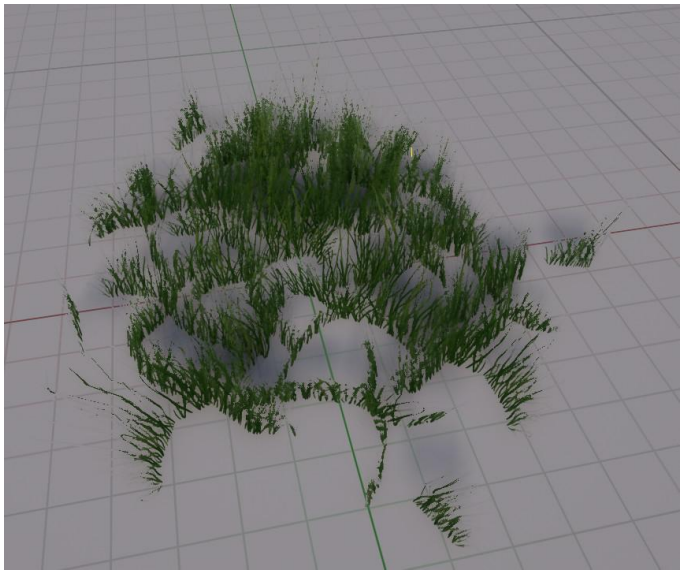
Obrázek 1.7: Viditelné použití metody billboards na koruně stromu

1.2 Generování vegetace

Existuje několik přístupů jak si 3D modely vegetace vytvořit. Tím nejzákladnějším je použít program na obecné modelování jako je 3ds Max, Blender nebo Maya. Pro generování mnoha objektů lze použít naivní přístup, kde se



Obrázek 1.8: Obrázky (billboards) stromů

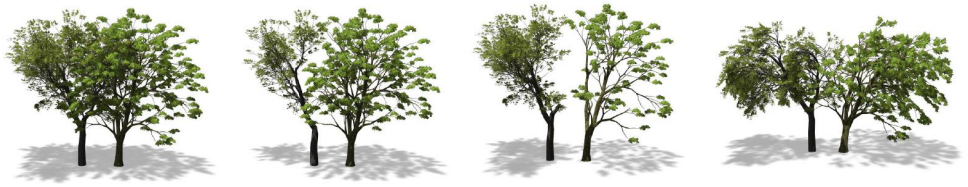


Obrázek 1.9: Obrázky (billboards) trávy

plocha určená k růstu vegetace rozdělí na čtverce, do prostředku se zasadí pomyslné semínko, kde má vyrůst strom (keř, rostlina) a následně se náhodně posune ve svém čtverci. Nakonec se může část semínek vypustit (neuchytila se). Takto vytvořený celek však nepůsobí příliš přirozeně, protože nedochází ke shlukům, jako v reálném životě. Další způsoby vycházejí z pozorování rostlin. Simulování reálných vlivů na růst stromu je velmi složitá úloha. Mezi hlavní vlivy patří světlo, gravitace, látky v půdě, ovzduší a okolní prostředí. V reálném světe bojují rostliny jak o území ,tak i o zdroje. Díky stále se měnícím podmínkám se mění i ekosystém v průběhu let. Díky některým podmínkám začnou mít některé rostliny navrch a v období několika let se změní rozmístění a skladba vegetace v daném prostředí. Při interakci stromů může dojít k několika různým výsledkům. Pokud používáme statické modely a neuvažujeme žádnou interakci, budou se stromy vedle sebe překrývat a zasahovat jeden do druhého. Pokud budeme řešit interakci stromů, může nastat ohýbání a prořezávání, silné prořezávání nebo zvýšené ohýbání (ukázka viz obr. 1.10). Zajímavé články na toto téma jsou od Bedřicha Beneše a kolektivu

[9], [10] a [11]. Pokud dochází k interakcím u menších rostlin, může jedna rostlina vyhrát a druhá je potom zcela potlačena (v dalších iteracích růstu zmizí).

Nejsložitější modely vegetace jsou stromy, proto se nyní zaměřím právě na



Obrázek 1.10: Ukázka interakce stromů. Zleva statické modely bez interakce, ohýbání a prořezávání, silné prořezávání, zvýšené ohýbání. Převzato z [9]

ně. S generování stromů úzce souvisí jejich reprezentace. Jednou z reprezentací je biologická. Strom se skládá z kmene, větví a koruny. Koruna je dále dělena na kosterní a vedlejší větve, na kterých rostou listy, poupata, květy a plody. Z biologické reprezentace lze odvodit geometrickou reprezentaci. Z geometrické reprezentace vycházejí Lindenmayerovy systémy. Další možnou reprezentací jsou datové struktury, kterým se říká také stromy, obecně grafy. Takto reprezentovaný strom má kořen a na něj navázané potomky. Jednotlivé uzly systému reprezentují listy, poupata, květy a plody.

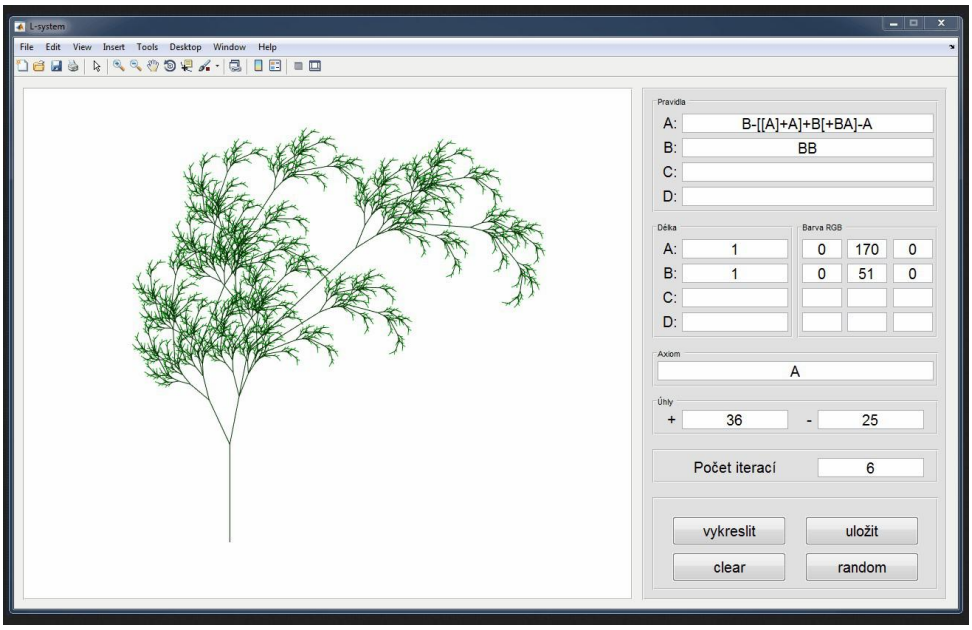
1.2.1 Lindenmayerovy systémy

Lindenmayerovy systémy, zkráceně L-systémy jsou variantou formální gramatiky, kde se paralelně přepisují řetězce podle určitých pravidel. Používají se symboly, ke kterým jsou dána přepisovací pravidla. Dále volíme počet iterací, po které se budou pravidla znovu přepisovat. V každé iteraci se každý znak z původního řetězce vymění za nový podle daného pravidla. L-systémy mohou také používat zásobník. Výsledný řetězec je interpretován a vznikne buď obrázek (2D), nebo přímo geometrická reprezentace rostliny 3D. V tomto případě znaky ve výsledném řetězci představují části rostliny. Na tomto principu je v dnešní době založeno mnoho programů na vytváření vegetace, například software SpeedTree a Xfrog, které nabízejí velmi realistické modely rostlin.

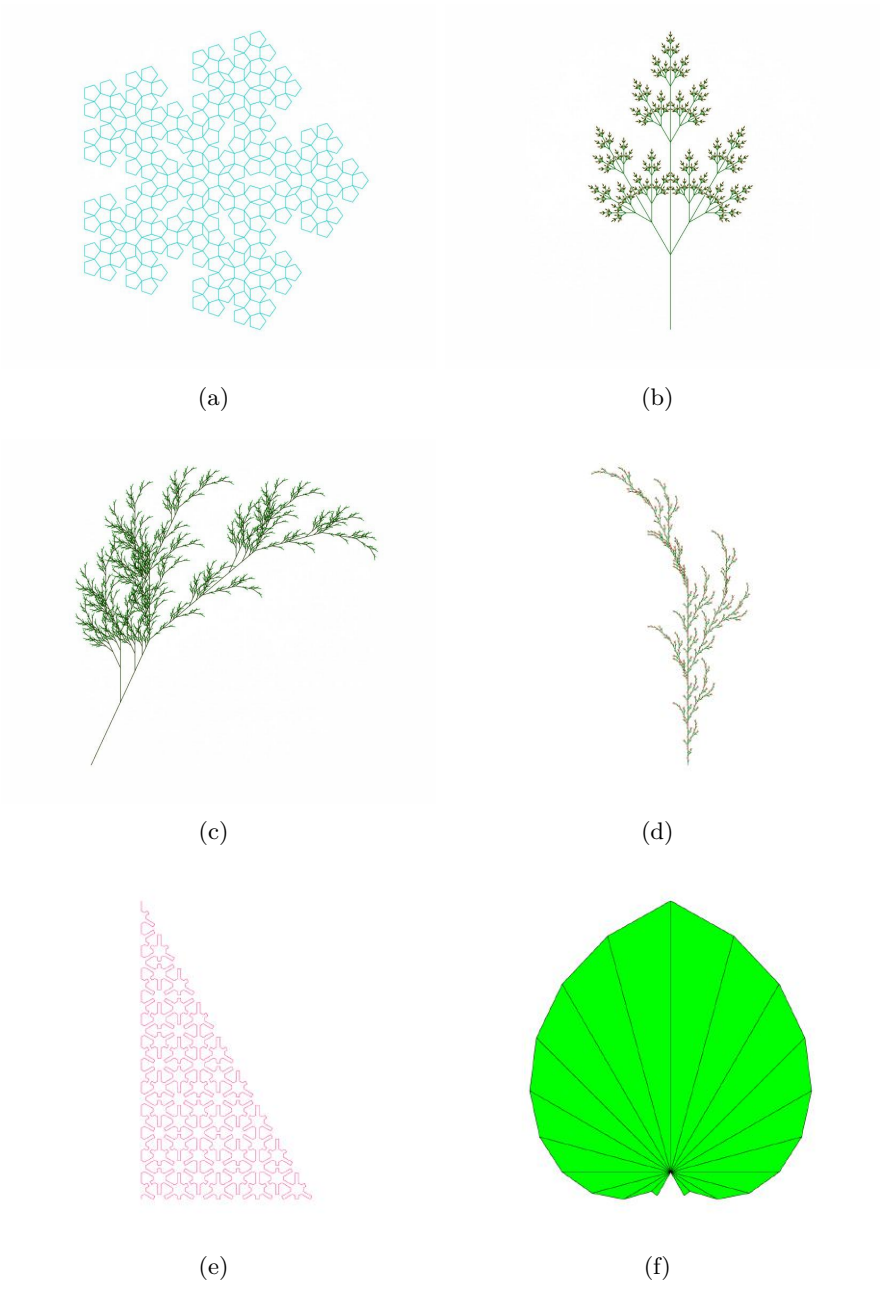
Díky L-systémům lze generovat rozvětvené struktury, jako jsou rostliny a stromy. Složitější L-systémy (*„Otevřené“*) mohou interagovat s prostředím, což je velmi užitečné právě pro růst rostlin, kdy lze simulaci více kontrolovat a ovlivňovat ([12]). Počet iterací u rostlin většinou simuluje, jak dlouho rostlina roste a tedy, jestli je rozbočená, má poupě, kvete apod.

Na vytváření 2D obrázků pomocí pravidel L-systémů jsem vytvořila aplikaci v programovém prostředí *Matlab* (viz obr. 1.11). Pomocí počátečního slova, přepisovacích pravidel, velikosti úhlů a volby barev lze docílit zajímavých

obrázků fraktálů a rostlinek (viz. obr. 1.12).



Obrázek 1.11: Aplikace na generování 2D L-systémů



(a) (b) (c) (d) (e) (f)

Obrázek 1.12: Použití L-systému ke generování 2D obrázků. (a, e) obecné použití. (b, c, d) 2D jednoduché rostliny pomocí L-systémů. (f) vyplnění polygonu barvou

Kapitola 2

Unreal engine

V této kapitole jsou nejdříve uvedeny obecné charakteristiky herních engine, jejich použití a zmíněny některé hry vytvořeny v nejznámějších engine. Druhá část kapitoly se zaměřuje na Unreal Engine, jeho využití, použití, šablony her, vztahy mezi hlavními třídami a režimy spuštění hry.

2.1 Herní engine

Herní engine je software, sloužící k vývoji her. Občas dochází k mylné představě, že herní engine umožňuje vývoj hry bez použití dalších programů. Což je možné pro velmi jednoduchou hru bez zajímavých efektů, modelů a mateirálů. Pro komplexní a zajímavou hru, je třeba všechny modely a textury připravit mimo engine a následně je importovat (souhrný název angl. *assets*). Herní engine zajišťuje některé z následujících funkcí

- render pro 2D a/nebo 3D grafiku
- zpracování fyziky a/nebo detekce kolizí
- zvuk
- skriptování
- animace

- umělá inteligence
- sítě, správa paměti, vlákna
- graf scény
- podpora videa

Engine se skládají z komponent: hlavní herní program, render, audio, fyzika, umělá itneligence. Dříve byly herní engine přizpůsobeny především jednomu typu hry. Dnes už mají rozšířenou funkcionalitu a je možno vytvářet v jednom engine více typů her. V tuto chvíli jsou na trhu tři hlavní engine a to: Unreal Engine 4, Unity a CryEngine. Tyto připravené herní engine mohou posloužit jak úplným začátečníkům, tak velkým herním studiím. Pokud si chce uživatel vyzkoušet tvorbu hry, je celkem příhodné použít některé z volně dostupných herních engine. Jen díky nim si může vytvořit uživatel hru zcela sám. Tím je myšleno složitější hru, s mechanikou, fyzikou, umělou inteligencí apod. Obyčejné piškvorky, hada, nebo třeba miny je snadné vytvořit i bez použití engine. Navíc v tuto chvíli jsou některá prostředí natolik uživatelsky přívětivá, že si může funkční hru vytvořit i člověk, který neumí vůbec programovat. Samozřejmě umět programovat je nespornou výhodou, ale není nutnou podmínkou. Najde se pár nadšenců, kteří si napíší svůj vlastní engine, ale těžko se vyrovná těm, na kterých pracují komerčně velká studia a stále je zdokonalují. Často mají herní studia smlouvy se studiem pracujícím na herním engine, který používají. Díky tomu mohou herní studia dávat požadavky na zdokonalení části engine, kterou potřebují vylepšit, nebo přidat některou funkcionalitu kvůli hře, kterou v engine vyvíjejí. Pod záštitou EpicGames vznikly v různých verzích Unreal Engine hry jako Unreal Tournament, série her Harry Potter nebo Borderlands. V CryEngine byla vytvořena hra Crysis a dokončuje se česká hra Kingdom Come:Deliverance. Zajímavá strategická 2D hra s názvem Plague Inc:Evolved byla vytvořena v Unity engine.
Většina herních studií si vytváří vlastní engine. Nevýhodou je čas, který se musí strávit navíc vývojem engine. Velikou výhodou je, že se engine vyvíjí směrem, který je pro hru potřeba a je jí zcela přizpůsoben. Další výhodou je, že mají autoři přístup ke všem kódům a ví přesně, jak co funguje, narozdíl od ostatních engine, které jsou často poskytnuty jako *black box* (víme co je vstupem a výstupem, ale nevidíme dovnitř kódu a můžeme se jen domýšlet, jak fungují procesy uvnitř). Ukázkou engine, které si vytvořilo herní studio čistě pro své účely je například engine s názvem ZenGin a Genome Engine, kterÉ si vytvořila německá společnost Piranha Bytes. V engine ZenGin vyšla hra Gothic a Gothic II, v Genome Engine vyšlo pokračování a to Gothic 3, Risen, Risen 2 a Risen 3. Při přechodu mezi engine byl vidět velký skok mezi grafickým zpracováním. Na obrázcích 2.1 jsou ukázky prostředí z her Gothic, Gothic II a Gothic 3, kde je názorně vidět, jak se vyvíjelo zobrazení vegetace v čase. Nejprve byl materiál na krajině velice jednoduchý, nebyly používány modely trávy a rostlin, Stromy byly složeny z několika obrázků. V další verzi

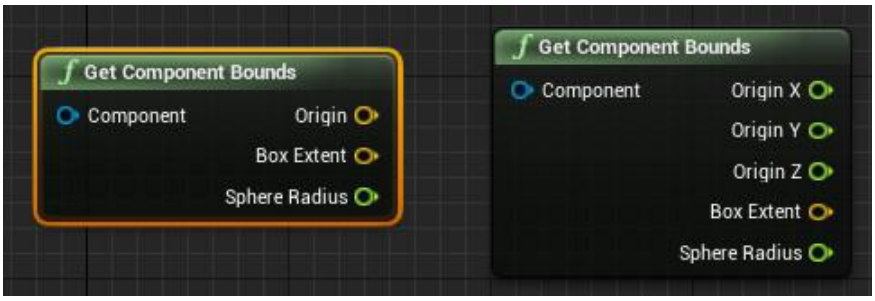
se objevily modely trávy a dalších rostlin, kvalita stromů se zlepšila. Ve třetá verzi se zobrazení vegetace zlepšilo mnohem více, než při přechodu na dřívější verzi.

Obrázek 2.1: Ukázky ze hry Gothic (nahoře) převzato z [15], Gothic II (uprostřed) převzato z [16] a Gothic 3 (dole) převzato z [17]

2.2 Práce s Unreal Engine 4

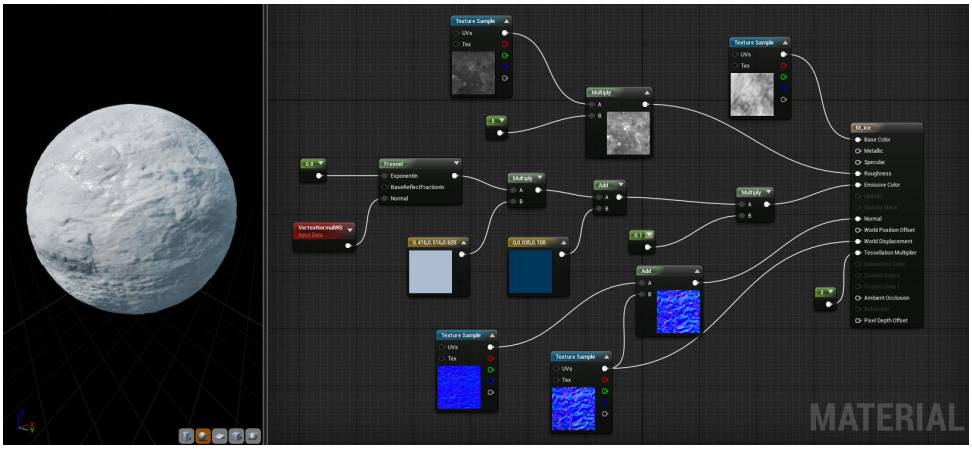
Unreal Engine [1] je herní Engine, vytvořený firmou Epic Games. Jeho první verze vyšla v roce 1998 a od té doby vyšlo mnoho nových verzí a stále se tento Engine zdokonaluje. V tuto chvíli je nejnovější verze 4.15 a brzy má vyjít verze 4.16. Od verze 4 mohou uživatelé implementovat kód v C++ a celý Engine umožňuje vývojářům větší kontrolu nad svým projektem, především se zde objevil nový debugger a při testování lze provádět přímo změny v kódu, namísto složitého procesu v předešlých verzích. Velikou výhodou je, že lze používat UE4 v tuto chvíli zdarma. Dříve se platila měsíční licence, ale došlo ke změně, kdy lze stáhnout a tvořit v UE4 zadarmo. Zpoplatněné nastává až u her, které komerčně vydělávají větší částky a zde se z celkového výdělku odvádí 5% Epic Games. Ve své práci jsem používala nejnovější verzi a také v další části textu budu popisovat pouze Unreal Engine 4.
Unreal Engine je primárně určen k tvorbě her. Další možné využití je například k vizualizaci architektonických staveb a následné možnosti touto stavbou procházet s možností měnit materiály a získat tak lepší představu, jak bude výsledná stavba vypadat.

Jsou dva základní přístupy, jak používat tento engine, pokud chceme naši

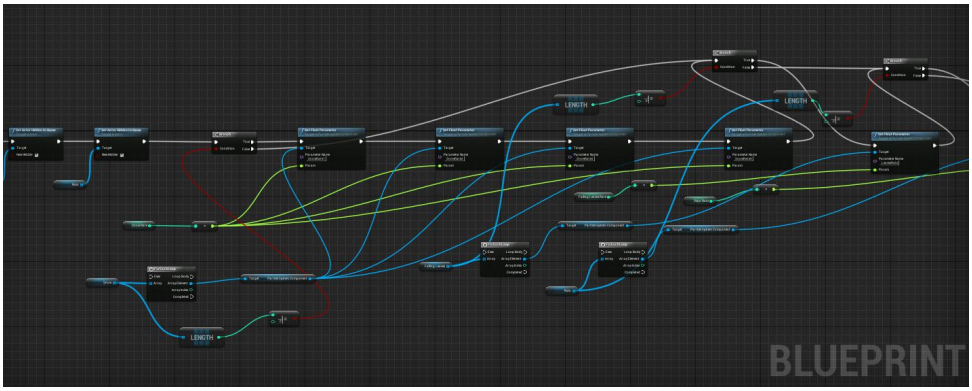


Obrázek 2.2: Ukázka uzlů v blueprints. Vpravo stejný uzel po rozdělení pinu počátku

scénu rozhýbat a přidat do ní dynamické prvky. Tím prvním je klasické programování, zde konkrétně v jazyce C++. Druhý způsob je využití tzv. *blueprints*, kde je možnost vizuálně skriptovat za použití uzlů angl. *nodes* (viz obr. 2.3 a 2.4), které slouží k definování a práci s objektově orientovanými třídami. Díky tomu mohou využít engine a udělat si vlastní funkční hru i neprogramátoři, jelikož se jedná o velmi flexibní a mocný nástroj, který je srovnatelný s programováním. Každý uzel má své jméno, funkci, vstupy a výstupy. Práce s uzly je velmi snadná a intuitivní. Hledat je lze podle jména, kategorie, nebo je vložit přímo přes zkratku. Prostředí nedovolí napojit vstupy a výstupy, které nejsou stejného typu. Pokud je typ blízký, například z proměnné typu int na float, tak se samo provede přetypování (přidá se uzel). Pokud má uzel jako vstup nebo výstup například barvu, či pozici, je možno



Obrázek 2.3: Ukázka uzlů v editoru materiálu sněhu

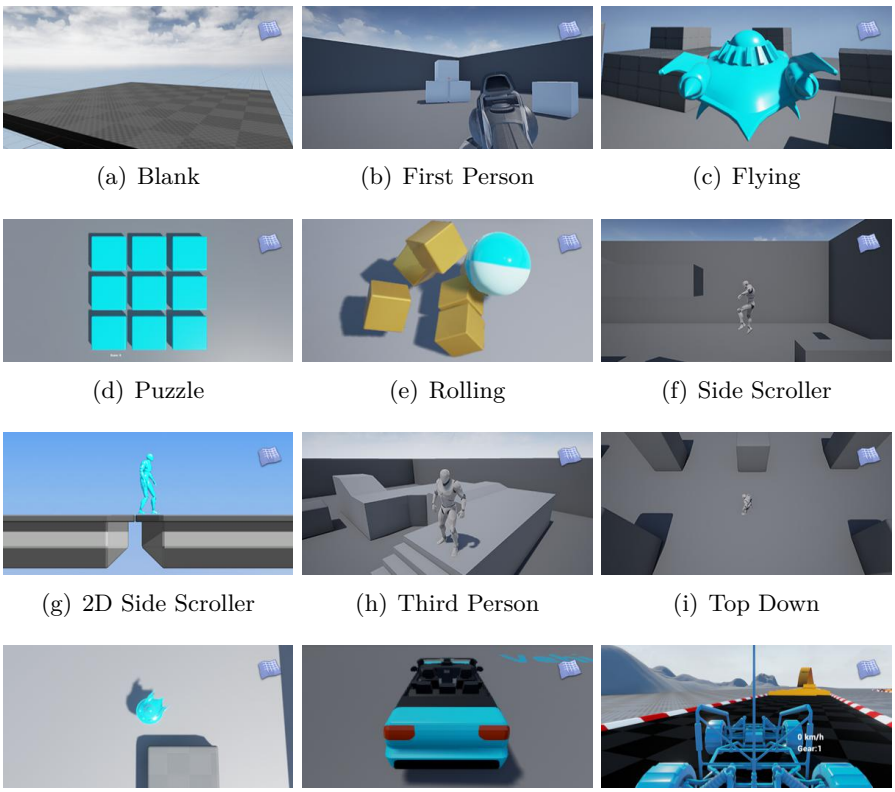


Obrázek 2.4: Ukázka uzlů v editoru blueprintu

toto napojení rozložit na jednotlivé složky (viz obr. 1.2), tedy barvu na R, G, B a A apod.

Unreal Engine byl původně určen pouze pro střílečku z první osoby (*First Person*), ale postupně byl rozšířen i na další herní módy. V UE4 existuje v tuto chvíli několik šablon, které velmi usnadňují vývoj hry (viz obr. 2.5). Tyto šablony poskytují veliký komfort, protože si může uživatel zvolit například šablonu první osoby a hned po vytvoření projektu může zapnout hru a běhat, skákat a střílet. Ke každé šabloně je také vytvořený level, ve kterém se nachází pár modelů (obyčejné tvary, jako kvádry) a lze si vyzkoušet interakci s nimi. V šabloně pro třetí osobu je vytvořená postava, která má připraveny všechny pohyby a animace.

Další důležitou částí jsou komponenty. Komponenty v UE4 jsou součástí funkcí, které lze přidat k herci - angl. *Actor*. Tyto komponenty nemohou existovat samy o sobě, ale po přidání k herci, bude mít herec přístup k funkcím poskytovaných komponentou a může je používat. Jako herce si lze představit například auto, které je složeno z jednotlivých komponent, jako kola, světla,



(a) Blank (b) First Person (c) Flying

(d) Puzzle (e) Rolling (f) Side Scroller

(g) 2D Side Scroller (h) Third Person (i) Top Down

(j) Twin Stick Shooter (k) Vehicle (l) Vehicle Advanced

Obrázek 2.5: Šablony her v UE4

klakson. Každá komponenta plní svoji funkci. Takto vytvořeného herce lze umístit do scény, aniž by měl přiřazen skript. Teprve přidání kódu (C++ nebo blueprint), je možno rozhýbat herce. V UE4 se nachází mnnoho rozdílných komponent

- AI (umělá inteligence)
- Zvuk
- Kamera
- Světla
- Pohyb
- Navigace
- 2D paper (modifikace a přidání 2D obrázků)
- Fyzika

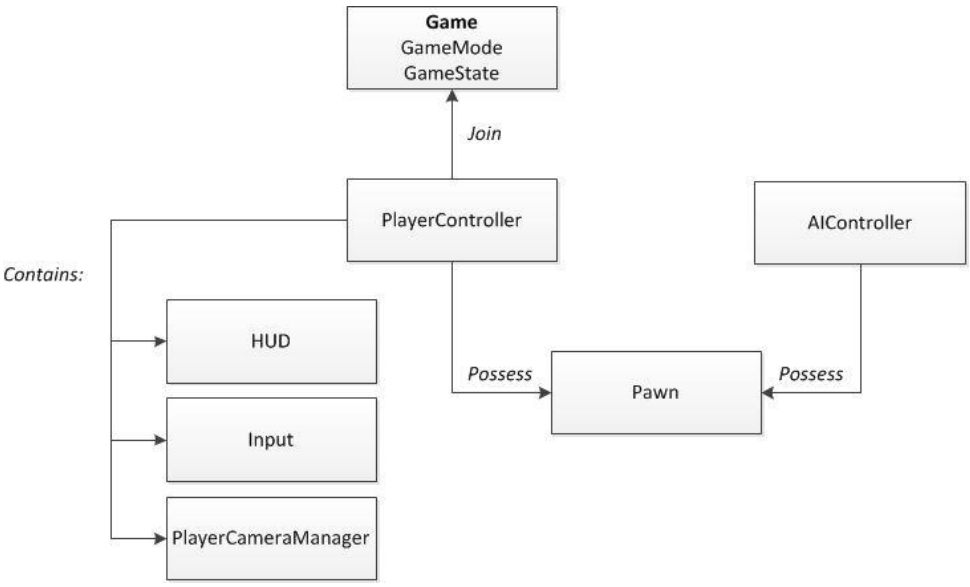
- Render
- Tvary
- Skeletální mesh
- Statický mesh
- Widget
- Nástroj

Framework Class Relationships

Na obrázku 2.6 je vidět vztah mezi rámcovými třídami v UE4. Hra je tvořena *GameMode* a *GameState*. Hráči jsou propojeni s *PlayerControllers*, který umožňuje hráči mít ve hře pěšáky (*Pawn*), díky kterému mají fyzikální reprezentaci v levelu. *PlayerControllers* také poskytuje hráčovi vstupní ovládací prvky, *HUD* (*heads-up* zobrazení) a *PlayerCameraManager* pro ovládání zobrazení kamery. *GameMode* v sobě skrývá definici hry, herní pravidla, podmínky výhry apod. Typicky by neměl mít mnoho dat, které se mění během hry. *GameState* obsahuje stav hry, což může zahrnovat propojení hráčů, skóre, seznam splněných úkolů apod. *PlayerState* je stav účastníka hry, jak lidského hráče, tak postav ve hře (bot), které simulují hráče. Umělá inteligence (*NPC, Non Player Character*), která ve hře existuje jako část hry by neměla mít *PlayerState*. *HUD* je používání jako 2D zobrazení, kde jsou informace, které hráč potřebuje vědět, jako počet životů, munice, energie apod..

Game Flow

Herní tok ukazuje proces od spuštění engine až po spuštění hry (viz obr. 2.7). V UE4 jsou dvě možnosti jak spustit hru a to z editoru nebo jako samostatnou hru. Obecné pořadí událostí je inicializace engine, vytvoření a inicializování *GameInstance*, načtení levelu a spuštění hry. **Standalone mode** je režim, kdy se spouští hra mimo editor. Objekty potřebné k hraní hry jsou vytvořeny a inicializovány bezprostředně po inicializaci engine na začátku. Objekty jako GameInstance jsou vytvořeny a inicializovány před spuštěním engine a výchozí mapa je načtena okamžitě po zavolání startovní funkce engine. Hra začíná oficiálně v bodě, kdy se v levelu vytvoří příslušný režim a stav hry (*Game a State mode*), následně ostatní herci (*Actors*). V **Editor mode** režimu, kde se hra spouští funkcí *Play In Editor* (PIE) a *Simulate In Editor* (SIE), je použit odlišný postup. Engine se inicialuzuje a spustí ihned, ale



Obrázek 2.6: Framework Class Relationships převzato z [18]

vytvoření a inicializování objektů jako *GameInstance* jsou odloženy, dokud uživatel nezmáčkne talčítko ke spuštění PIE nebo SIE relace. Navíc, Actor v levelu je duplikován, takže změny neovlivní level v editoru a každý objekt, včetně *GameInstance* objektů má oddělenou kopii pro každou PIE instanci. Tento Editor režim se napojuje na Standalone režim se začátkem hry ve třádě *UWorld*.

Obrázek 2.7: Herní tok pro samostatmý režim a režim v editoru, převzato z [19]

Kapitola 3

Vegetace v Unreal engine 4

V této kapitole jsou popsány možnosti zobrazování vegetace v Unreal Engine 4. Dále jsou uvedeny nástroje na manipulaci s vegetací a jejich použití v nástroji na ovládání ročního období. Díky těmto nástrojům lze vkládat více instancí najednou, volit různá nastavení a tím docílit variabilně vypadajícího celku, přestože máme pouze omezený počet modelů. Také lze snadno přistupovat k parametrům materiálů vegetace a měnit je.

3.1 Modely vegetace

Modely vegetace je možno získat v podstatě třemi způsoby. První možností je vytvořit model v nějakém programu pro 3D grafiku. Mezi 3D programy patří například 3ds Max, Maya a Blender. Tento proces je složitý, zdlouhavý a není zaručen kvalitní výsledek. Navíc je potřeba mít připravené textury a dále je upravit ve 2D editoru. Druhý způsob je zakoupení licence na modelář od SpeedTree [3]. Tento nástroj slouží primárně k vytváření modelů stromů, které jsou kompatibilní a přizpůsobené exportu právě do UE4. Uživateli je usnadněna práce s vytvářením modelu, kdy si pár kliknutími vytvoří kmen a větve stromu a dále je upravuje podle svých preferencí. Takto vytvořené modely mohou být však exportovány pouze do UE4.

Poslední možností je získat již hotový model. Na internetu lze najít některé modely zdarma od jiných uživatelů. Tyto modely bývají nekvalitní a nepůsobí reálně (viz obr. 3.1). Na druhou stranu si můžete zakoupit již hotový model buď přímo v UE4 obchodě (*Marketplace*), nebo někde jinde. SpeedTree nabízí kromě měsíční licence také hotové modely, které lze zakoupit většinou

Obrázek 3.1: Ukázka jednoduchých modelů dostupných zdarma

v balíčku po více kusech. Tyto modely bývají často stylizované do určitého podnebí nebo oblasti. Velikou nevýhodou je cena, která je poměrně vysoká. Dalším placeným nástrojem je Xfrog. Xfrog umožňuje vytvářet 3D animované modely stromů, květin a efekty založené na přírodních pravidlech. Podobně jako SpeedTree, nabízí i Xfrog placené balíčky vegetace, kde se nacházejí stromy z určitého podnebí a oblastí světa. 3D profesionální modely nabízí společnost TurboSquid, která se sice nespecializuje pouze na modely vegetace, ale nabízí i sekci krajiny, kde jsou dostupné modely stromů, květin, trav apod. Tyto modely jsou nabízeny jak jednotlivě, tak v rozsáhlých kolekcích.
Naštěstí je zde ještě jedna možnost, kterou uvítají především nováčci, kteří si chtějí vyzkoušet vytvořit svoji první hru v UE4 a nechtějí za to utrácet peníze. UE4 nabízí přímo několik volně dostupných balíčků zadarmo. Lze je najít v obchodě UE4, ale také v sekci výuky (angl. *Learn*). Tato sekce nabízí jak balíčky (modely, textury, materiály apod.), tak hotové projekty, které demonstrují využití UE4 (odrazy, particle systémy, strategická hra apod.).

Obrázek 3.2: Modely z ukázkového balíčku SpeedTree

Já sama jsem se proto rozhodla ve své práci využít pouze balíčky přímo od UE4, abych demonstrovala, co lze dosáhnout za výsledky, bez použití dalších nástrojů, modelářů a hlavně zadarmo. Je třeba zmínit, že SpeedTree nabízí jeden balíček zdarma, takovou ukázku se čtyřmi modely (viz obr. 3.2). Nejdříve jsem s tímto balíčkem pracovala, ale nakonec jsem se rozhodla pro balíček od UE4, protože zde balíček od SpeedTree nemusí být vždy dostupný zadarma. Balíček, který využívám od UE4 se nazývá *Open World Demo Collection* (viz obr. 3.3) a je také využíván přímo v dokumentaci UE4 pro demonstraci použití některých nástrojů. Jako bonus naleznete v tomto balíčku kromě modelů stromů také modely trávy, rostlin, kamenů, zajímavé textury

a další. Pro doplnění bych zmínila, že je možnost při spuštění nového projektu vybrat, jestli chce uživatel přidat startovní balíček. Tento balíček obsahuje některé modely, textury, ale co se týče vegetace, tak pouze jeden model keře (viz obr. 3.4).

Obrázek 3.3: Open World Demo Collection - přehlídka modelů a materiálů z balíčku

Obrázek 3.4: Model keře dostupný ve startovním balíčku

3.2 Nástroje UE4

V UE4 existuje mnoho nástrojů a možností (viz obr. 3.5), jak si práci s vegetací usnadnit a zároveň dosáhnout lepších (lépe vypadajících) výsledků. Jedná se zejména o vkládání a nastavení vkládaných modelů, parametry a materiály.

- Material
- Material Function

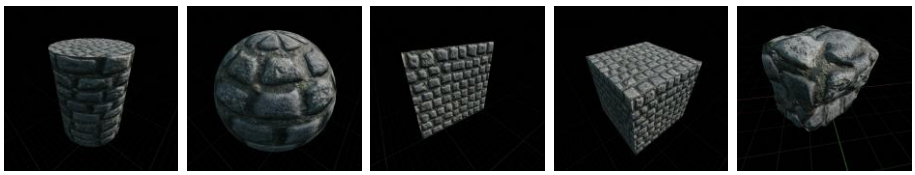
- Material Instance
- Material Parameter Collection
- Landscape Grass Type
- Foliage Type
- Foliage

Tyto nástroje (zobrazeny na obr. 3.8, 3.9, 3.10 a 3.11) se nacházejí v sekci *Advanced Asset* (pokročilé nástroje). Pouze možnost Foliage je specifická a nachází se mezi módy (*Modes)* v hlavní nabídce.
V následující části se budu zabývat popisem jednotlivých nástrojů a jejich využití (konkrétní ukázky v sekci 3.3).

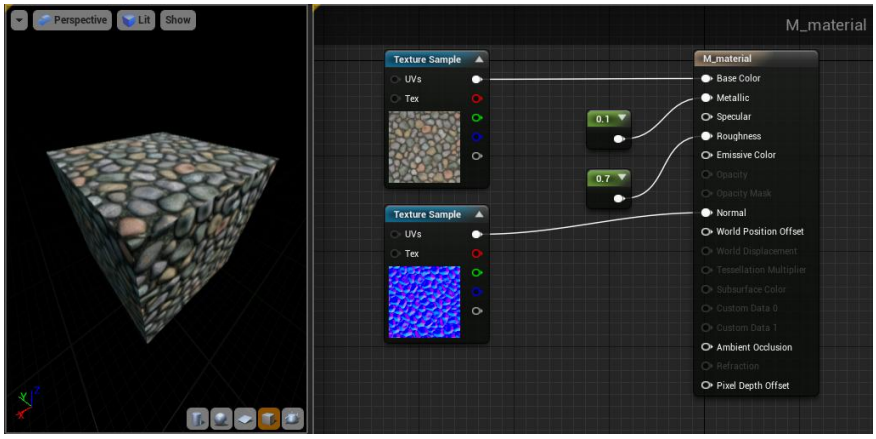
Obrázek 3.5: Ukázka vytvořených nástrojů pro manipulaci s vegetací. Z leva FoliageType, LandscapeGrassTool, Material, MaterialInstance, MaterialFunction, MaterialParameterCollection, Texture.

Material

Materiál (*Material*) je nástroj, který může být aplikován na Mesh (objekt složený ze sítě trojúhelníků) a ovlivňuje jeho vizuální vzhled jako například barvu, odrazivost, průhlednost, lesk a další. Díky tomu lze simulovat reálný povrch (materiál) objektů. Tyto materiály se vytvářejí v editoru materiálu, kde se používá stejný systém jako u plánků (blueprints), kde se ve vizuálním prostředí napojují jednotlivé uzly na výsledné složky materiálu (viz obr. 3.7). Při práci v editoru materiálu lze vidět ihned úpravy provededené na materiálu, které lze mapovat na primitiva válce, koule, plochy a kostky. Poslední možností je mapování na jakkýkoliv objekt, která je označen v prohlížeči složek (ukázky jednotlivých mapování viz obr. 3.6). Pro aplikaci změn na původní materiál a promítnutí změn do všech objekt, které tento materiál využívají je třeba aplikovat materiál (*Apply*).
Instance materiálu (*Material Instance*) se využívají ke změně vzhledu materiálu, aniž by bylo třeba u materiálu překompilovat shadery (což je drahé). Díky tomu lze měnit některé vlastnosti přímo za běhu hry. k tomu slouží



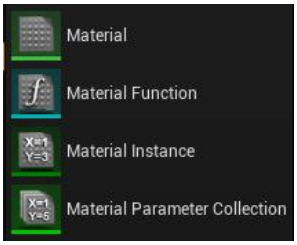
Obrázek 3.6: Ukázka mapování materiálu na různé objekty, zleva: válec, koule, plocha, kvádr a kámen



Obrázek 3.7: Ukázka z editoru materiálu

parametry, které definují právě ty vlastnosti a proměnné, které se budou měnit bez nutnosti kompilace. Jako příklad použití může být rozzáření a změny barvy objektu, pokud se dostaneme za hry do jeho blízkosti. Parametry je možné vytvořit pouze jako skalární (jedna složka) a vektorové (čtyři složky). Parametr lze v materiálu vytvořit buď přímo, nebo je možno konvertovat konstanty na parametry. Pokud je však konstanta dvousložková, bude konvertována na čtyřsložkový vektor. Potom je třeba použít maskování, nebo si vybrat pouze výstupy z uzlu, které jsou aktuální. Instance materiálu se dělí na konstantní a dynamické. Konstantní lze měnit v editoru a nastavovat podle potřeby, ale nelze je již ovládat za běhu hry. Dynamické lze ovládat jak v editoru, tak za hry. Díky těmto parametrům je možno v editoru okamžitě vidět změny, například při změně barvy (vektorový parametr).
Funkce materiálu (*Material Function*) jsou části grafů materiálu, které mohou být uloženy jako balíčky a použity pro více materiálů. To umožňuje vytvářet složitou a rozsáhlou strukturu propojených uzlů, která je následně uložena jako jeden uzel s danými vstupy a výstupy. Díky tomu můžeme použít tuto funkci jedním snadným napojením na různé materiály a jakákoliv změna ve funkci se automaticky promítne do všech materiálů, kde je funkce napojena.
Kolekce parametrů (*Material Param Collection*) je nástroj, který umožňuje ukládání skalárních a vektorových parametrů, které lze následně použít (odkázat) v libovolném materiálu. Díky tomu se jedná o silný nástroj, který lze využít k ovládání globálních dat v mnoha materiálech, ale také k řízení

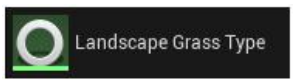
hodnot přenášených mezi jednotlivými levely.



Obrázek 3.8: Nástroje pro práci s materiály

Grass Tool

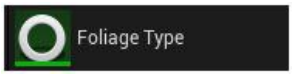
Jedním ze způsobů vložení více modelů najednou je nástroj *Grass Tool*. Tento nástroj, jak už název napovídá, je určen na vkládání travnatého porostu přímo na povrch krajiny. Nejprve je zapotřebí si vytvořit krajinu nástrojem *Landscape* a dále materiál pro krajinu. Právě krajina má tu speciální vlastnost, že lze vytvářet několik vrstev a následně po krajině kreslit štětci. Každá vrstva představuje typ pokrytí krajiny (například travnatá, hornatá, cesty,...). Do konkrétních vrstev lze napojit *Landscape Grass Type* a vykreslovat tím modely trávy na konkrétní vrstvu (ukázka vrstev viz obr. 3.12 a materiálu krajiny viz. 3.13).
V *Landscape Grass Type* se volí typy trávy. Lze přidávat a odebírat jednotlivé typy. Ve skutečnosti lze tento nástroj použít na jakýkoliv *Mesh*, ne pouze na trávu. Vhodný je ale především pro trávu, květiny, kapradí nebo malé keře. Hlavním důvodem je to, že po nakreslení štětcem není možné jednotlivými modely dále manipulovat. Lze sice nastavit hustotu, velikost, náhodnost rozmístění, ale není zde možnost modely označit. Proto se nehodí pro použití větších objektů, jako jsou stromy, kde občas náhodné rozmístění není tak náhodné a je zapotřebí modely posouvat, otáčet apod. V samotném nástroji *Landscape Grass Type* se zvolí počet trav (rostlin), které bude jednotlivý typ nést. U každého modelu lze nastavit několik parametrů. Mezi ty nejzajímavější patří hustota, škálování, zarovnání a rotace. Takto vytvořený typ se použije v materiálu krajiny, kde se přiřadí do jednotlivé vrstvy.



Obrázek 3.9: Nástroj Landscape Grass Type

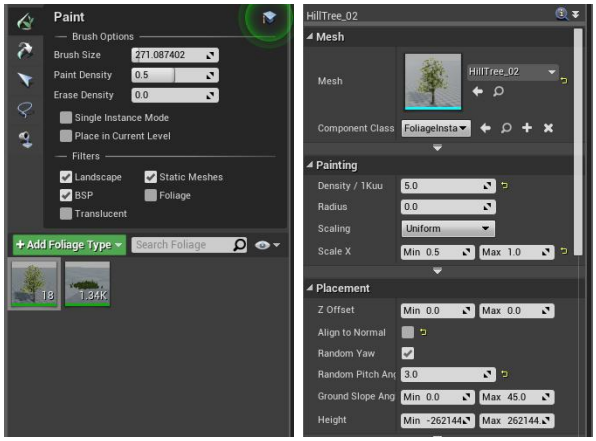
Foliage Type

Foliage Type je podobný *Landscape Grass Type*, má však více nastavení a širší využití, ale přiřazuje se mu pouze jeden *Mesh*. Tento typ se využívá v nástroji *Foliage* 3.2.



Obrázek 3.10: Nástroj FoliageType

Foliage



Obrázek 3.11: Nástroj Foliage

Druhým způsobem, jak vložit více objektů najednou je nástroj *Foliage*, který patří mezi hlavních pět módů (*Place, Paint, Landscape, Foliage a Geometry Editing*). Slouží ke vkládání více instancí najednou. Pomocí štětce se maluje na terén. Štětci se nastavuje poloměr a hustota vkládaných objektů. Nejprve je třeba si zvolit modely, které se budou štětcem vkládat na terén. Tyto objekty mohou být buď *Mesh* nebo *Foliage Type*. Pokud je vložen model s konkrétním názvem, nelze znovu vložit ten samý model. Toto lze řešit právě pomocí *Foliage Type*, kde lze vytvořit více typů, například *FT_tree01, FT_tree02,* atd. a každému typu přiřadit stejný model stromu, ale nastavit různé parametry. Před použitím štětce se zvolí, které modely budou v aktuálním malování použity a které ne (jednoduchým za/odškrtnutím). Pro každý *Mesh* a *FoliageType*, které jsou umístěny ve *Foliage* lze nastavit několik parametrů, což se hodí především pro samotné modely. *FoliageType* si nese už nějaké vlastní nastavení.
Mezi možnosti, které nejvíce ovlivňují vzhled a polohu modelů patří

- hustota (hustota výskytu tohoto modelu)
- výška výskytu (minimální a maximální výška souřadnic, kde se může tento model vyskytovat)
- sklon terénu (minimální a maximální úhel terénu, kde se může tento model vyskytovat)
- zarovnání (zarovnání kolmo k terénu, nebo zanechání modelu rovnoběžně s osou)
- velikost (minimální a maximální parametr, kterým se škáluje velikost objektu - náhodnou hodnotou v rozmezí)
- otočení (otočení o kolik stupňů a kolem které osy)

Díky těmto nastavením lze vytvářet reálněji vypadající krajiny s malým počtem modelů (ukázka použití viz 3.3). Narozdíl od *Grass Tool* se lze přepnout do módu, kdy je možné vybírat jednotlivé modely vložení nástrojem *Foliage* a volně s nimi pohybovat a měnit je ve scéně. Také lze nastavit štětci, že nebude vkládat, ale naopak mazat jednotlivé modely (samozřejmě ovlivňuje a maže pouze ty vložené nástrojem *Foliage*). Opět se volí které modely a s jakou hustotou je bude mazat.

3.3 Použití nástrojů

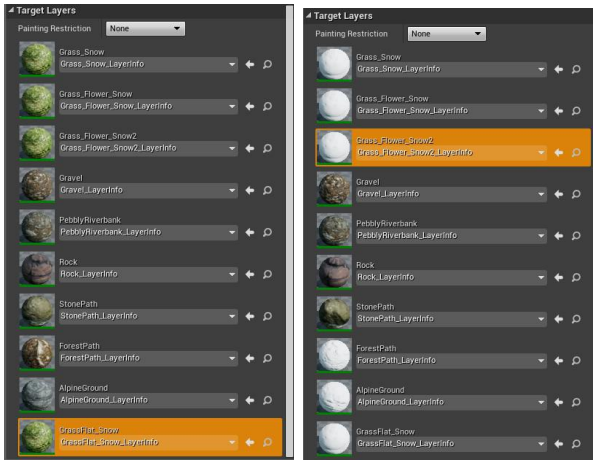
V této části naleznete ukázky, jak jsem jednotlivé nástroje z předešlé sekce využila ve svém projektu.

Materiál

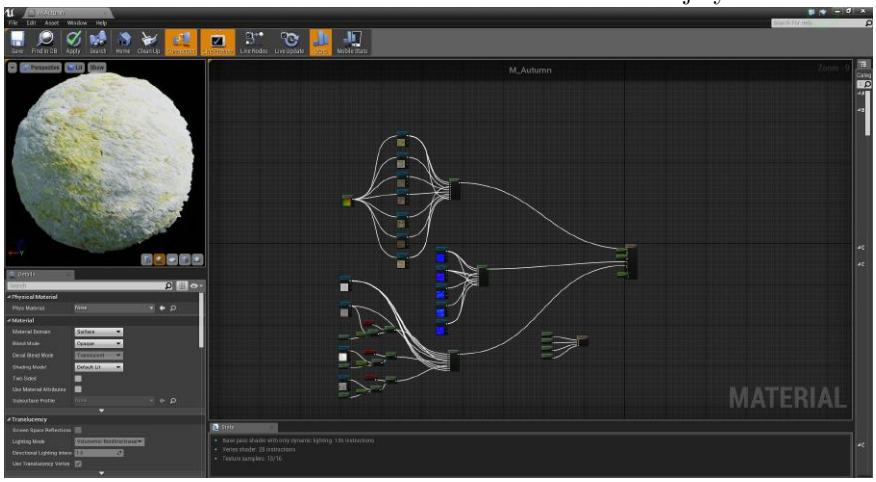
Ve své práci mám možnost přepínat mezi ročními obdobími. Za tímto účelem jsem vytvářela následující materiály.

- Krajiny - čtyři různé materiály pro jaro, léto, podzim a zimu. Všechny tyto materiály jsou určeny pouze pro krajinu (*Landscape*), jelikož obsahují vrstvy a také napojení na *Landscape Grass Type* (viz obr. 3.12 a 3.13).

- Částice - materiály používané pro částice v particle systémech. Jedná se o materiály sněhu, deště a listu.
- Pokrytí - materiál sněhu a mechu na pokrývání stromů a hornin.
- Sníh - speciální materiál na interakci chůze ve sněhu, kdy třetí osoba zanechává stopy (sešlápnutí sněhu) (viz obr. 3.14).



Obrázek 3.12: Ukázka vrstev v materiálu krajiny

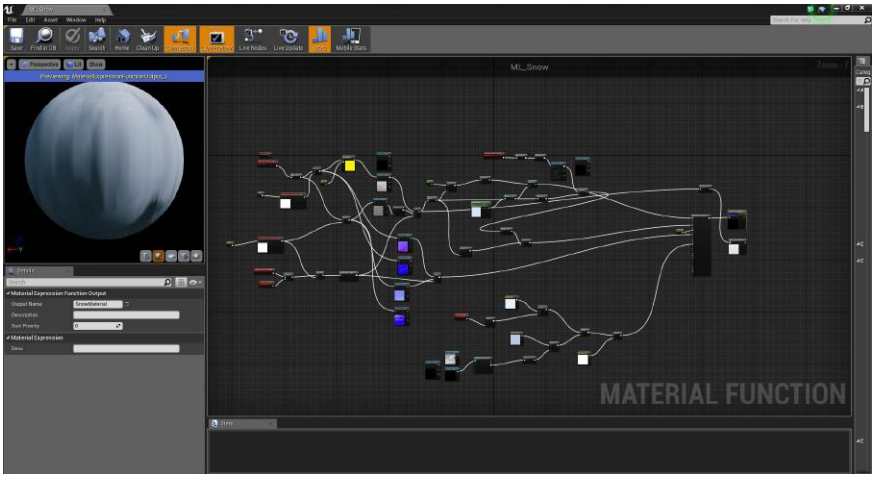


Obrázek 3.13: Ukázka napojení vrstev v materiálu pro podzimní krajinu

Při tom jsem narazila na nepříjemný problém, kdy může být v jednom materiálu použito maximálně 16 vzorků textur. Jelikož jsem vytvořila 8 vrstev na krajinu a každá měla k sobě 3 textury, tak jsem počet 16 jednoduše přesáhla a musela jsem materiály upravovat. S podobným problémem jsem se setkala při používání parametrů z kolekce. Jeden materiál totiž připouští použít maximálně dvě různé kolekce. V kolekci může být samozřejmě mnoho parametrů, takže jsem ve výsledku jednotlivé kolekce spojila do jedné a pracovala pouze s tou. Původně jsem si totiž chtěla rozdělit parametry podle

Obrázek 3.14: Materiál sněhu pro interakci chůze třetí osoby ve sněhu

toho co ovládají (kolekce pro barvu, pro stromy, pro roční období apod.). Kolekci parametrů jsem využila hlavně pro ovládání ročního období a pro manipulaci s materiálem trávy a stromů. Díky tomu mohu v materiálech podmíněnými příkazy (IF) rozpoznat roční období a upravit materiál. Pro jaro měním barvu trávy a listů stromů, stejně tak pro podzim. Pro zimu listy vůbec nejsou. Dále kontroluji míru pokrytí sněhem a mechem a jejich barvu (viz obr. 3.17, 3.18, 3.19 a 3.20).



Obrázek 3.15: Material function pro pokrytí sněhem

Funkci materiálu využívám pouze pro pokrytí sněhem a mechem (viz obr. 3.15, 3.16). Právě v tomto případě je vidět síla tohoto nástroje. Vytvořila jsem si totiž materiál pro mech a sníh. Následně jsem si připravila funkce, které existujícímu materiálu přidávají pokrytí sněhem a mechem a to tak, že se pokrytí přidává z vrchu, tedy řídí se světovými souřadnicemi a díky

Obrázek 3.16: Pokrytí sněhem na stromě

tomu vypadá pokrytí velmi realisticky a pokud otočíme modelem kamene, zůstane pokrytí na správném místě (ze shora). Tyto funkce jsem přiřadila všem materiálům na kterých chci, aby se toto pokrytí objevovalo. V tomto případě stromy a všechny druhy kamenů. Také jsem funkce napojila tak, aby při pokrytí jak sněhem tak mechem najednou, byl sníh nad mechem (občas mech vyleze na místech, kde je sníh řídký, protože nemají stejně vytvářený styl pokrytí).

Obrázek 3.17: Jemné pokrytí mechem na kameni a stromech

Grass Tool

Nástroj *Grass Tool* jsem využila ve všech ročních obdobích. Díky nastavení jsem mohla měnit výšku a hustotu trávy (v létě vyšší a hustší než na podzim).

Obrázek 3.18: Silné pokrytí mechem se změnou barvy na kameni

Obrázek 3.19: Jemné pokrytí sněhem na kamenech a stromech

Obrázek 3.20: Silné pokrytí sněhem na kamenech a stromech

Na jaře (viz obr. 3.21) a na podzim (viz obr. 3.23) jsem k trávě přidávala i některé rostliny. V létě (viz obr. 3.22) a na podzim kapradí. V zimně je tráva velmi nízká a samozřejmě bez rostlin (viz obr. 3.24).

Obrázek 3.21: Ukázka použití GrassTool na materiálu pro jarní krajinu

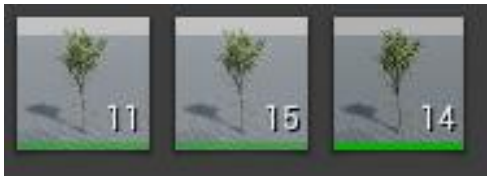
Foliage Type

Vytvořila jsem si několik typů, kde jsem mohla plně kontrolovat nastavení modelů a měla jsem tak možnost vkládat různé a různě uspořádané stromy pouze z jednoho modelu (viz obr. 3.25).

Obrázek 3.22: Ukázka použití GrassTool na materiálu pro letní krajinu

Obrázek 3.23: Ukázka použití GrassTool na materiálu pro podzimní krajinu

Obrázek 3.24: Ukázka použití GrassTool na materiálu pro zimní krajinu



Obrázek 3.25: Tři různé Foliage Type v nástroji Foliage pro jeden model

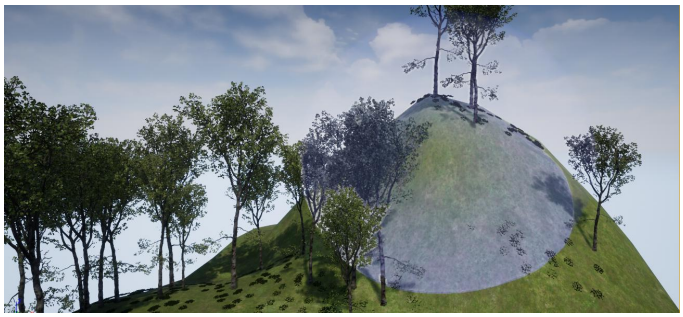
Foliage

Nejenže lze pomocí tohoto nástroje vkládat efektivně veliké množství vegetace najednou, ale hlavně lze kontrolovat pomocí nastavení, kde se bude jaký model vyskytovat a jaké bude mít nastavení. Pro ukázku, jak snadno lze výsledný vzhled ovlivnit jsem vytvořila dva lesy. První je za použití jednoho modelu, který je pouze vložen pomocí *Foliage* (viz obr. 3.26). Druhý les je tvořen stejným modelem, ale nastavila jsem mu náhodnou velikost mezi 50% a 100%, náhodné otočení kolem své osy a naklonění náhodně o maximálně 3 stupeň (viz obr. 3.27).

Obrázek 3.26: Stromy vložené nástrojem Foliage bez úpravy nastavení

Obrázek 3.27: Stromy vložené nástrojem Foliage s úpravou nastavení

Velikou výhodou je také nastavení výšky a sklonu výskytu daného modelu. Díky tomu lze nastavit, aby se stromy nevyskytovaly od nějaké výšky a na moc prudkých místech. Díky tomu lze táhnou štětcem a vkládat vegetaci, aniž bych si musela dávat pozor, aby jsem nevložila stromy na strmé srázy, takto se tam prostě nevloží (viz obr. 3.28).



Obrázek 3.28: Ukázka vkládání štětcem Foliage stromů a trávy na krajinu

Kapitola 4

Výsledky

V této kapitole jsou popsány dosažené výsledky (viz obr. 4.1). Jedná se o nástroj (blueprint) na ovládání ročního období, testování hranic zobrazení (kvantity), vytvořené scény a interakce s vegetací a terénem .
Při vytváření nástroje a testování jsem využívala pouze materiály dostupné přímo v UE4. Postupně jsem prošla a zpracovala dostupné modely, materiály a textury. Pomocí nich jsem vytvořila nové materiály, blueprinty, particle a levely (souhrn viz tabulky 4.1 a 4.2). V ukázkách od UE4 jsou již existující materiály (např. dřevo, kov, kamenná zeď), které využívají přiložené textury. Já jsem používala výhradně textury a vytvářela materiály nové. Materiály, které jsem upravovala, byly spjaté s vegetací a ostatními modely. Díky těmto úpravám mohu kontrolovat vlastnosti jako barvu a pokrytí.

	Mesh	Foliage	Kameny	Stromy	Textury	Materiály
OpenWorldDemo	42	17	21	4	138	14
StarterContent	51	1	1	-	103	67

Tabulka 4.1: Souhrn počtu modelů a materiálů, které jsem od UE4 zpracovávala

	Mesh/Foliage	Materiály	Textury	Bluperint	Particle	Levely
Upraveno	5	9	-	5	-	-
Vytvořeno	-	16	5	7	3	18

Tabulka 4.2: Souhrn počtu modelů a materiálů, se kterými jsem pracovala

Nejprve jsem vytvářela nástroj na změnu ročního období a počasí (sekce

Obrázek 4.1: Výsledky práce. Ze shora: nástroj na ovládání ročního období, testovací krajina, interakce se sněhem, interakce s vegetací

	rozměry krajiny v cm	délka testovací cesty (spline) v cm
malá krajina	6000 x 6000	22 820
střední krajina	12 400 x 12 400	47 700
velká krajina	25 200 x 25 200	80 320
krajina	12 400 x 12 400	49 920

Tabulka 4.3: Rozměry testovacích krajin a délky testovacích tras

	rozloha [m^2]	stromy	květiny	kapradí	kameny	tris
malá krajina	14 641	135	359	8	24	368 680
střední krajina	62 001	525	2154	286	21	500 070
velká krajina	255 025	1173	1282	1310	30	929 013
krajina	62 001	715	510	6	24	403 013

Tabulka 4.4: Rozloha, počty objektů a trojúhelníků jednotlivých scén

4.1). Postupně jsem si vytvářela jednolitvé materiály a upravovala existující. Dále jsem vytvořila kolekci na ovládání parametrů. Pomocí té jsem začala ovládat barvy stromů. Díky podmíněným příkazům vybírám v textuře místa, kde se nacházejí listy a ty dále měním (zprůhledním, přebarvím, apod.). Takto nedojde k nechtěnému přebarvení větví. Následně jsem vytvořila skript (blueprint), kterým lze ovládat parametry z kolekce, ale také objekty a vztahy objektů ve scéně (herní prvky). Vytvořila jsem materiály krajiny a particle sněhu, deště a padajících listů. Do skriptu jsem přidala ovládání větru a světla. Jako poslední jsem vytvořila funkce materiálů na pokrytí sněhem a mechem a přidala ovládání do skriptu. Pomocí skriptu je možno ovládat všechna nastavení z editoru, kde je vytvořeno grafické rozhraní podle viditelných proměnných ve skriptu. Proměnným jsem nastavila omezení, aby nebylo možné nastavit nesmyslné hodnoty. Nakonec jsem vytvořila grafické menu, které lze používat pouze ve hře a umožnila tak ovládání skriptu přímo za běhu hry.

Poté jsem se zabývala testováním (sekce 4.3). Pro tento účel jsem vytvořila čtyři krajiny (tři pro první testování a jednu pro druhé viz sekce 4.2). Ty jsem dále měnila a ukládala jako jednotlivé levely. Snažila jsem se vytvořit rozmanitou krajinu a použít širokou škálu modelů a materiálů. Při testování jsem pozorovala změnu FPS při průchodu krajinou. V krajině se měnila hustota modelů, počet LOD, zapnutý/vypnutý vítr, stacionární/dynamická světla, detaily (Epic, High a Medium). Výsledky jsem vizualizovala grafem (porovnání) a tabulkou průměrných FPS.

Rozměry jednotlivých krajin a délka trasy, po které se pohyboval charakter při testování jsou uvedeny v tabulce 4.3. Přehled počtu modelů a počtu trojúhelníků pro jednotlivé krajiny jsou shrnuty v tabulce 4.4. Je třeba zmínit, proč je počet trojúhelníků ve scéně tak malý, přestože se v ní nachází několik set modelů. Počet trojúhelníků pro jednotlivé modely stromů a rostlin uvádím

	tris LOD0	tris LOD1	tris LOD2	tris LOD3
HillTree Tall	58 321	4292	3564	32
HillTree	49 495	4317	2849	32
BogMyrtleBush	6124	1670	32	-
FieldScabious	3362	95	-	-
Yarrow	2846	535	164	-
Fern	1923	769	192	-
FieldGrass	765	278	18	-
Heather	376	188	37	-

Tabulka 4.5: Počty trojúhelníků modelů vegetace pro jednotlivé LOD

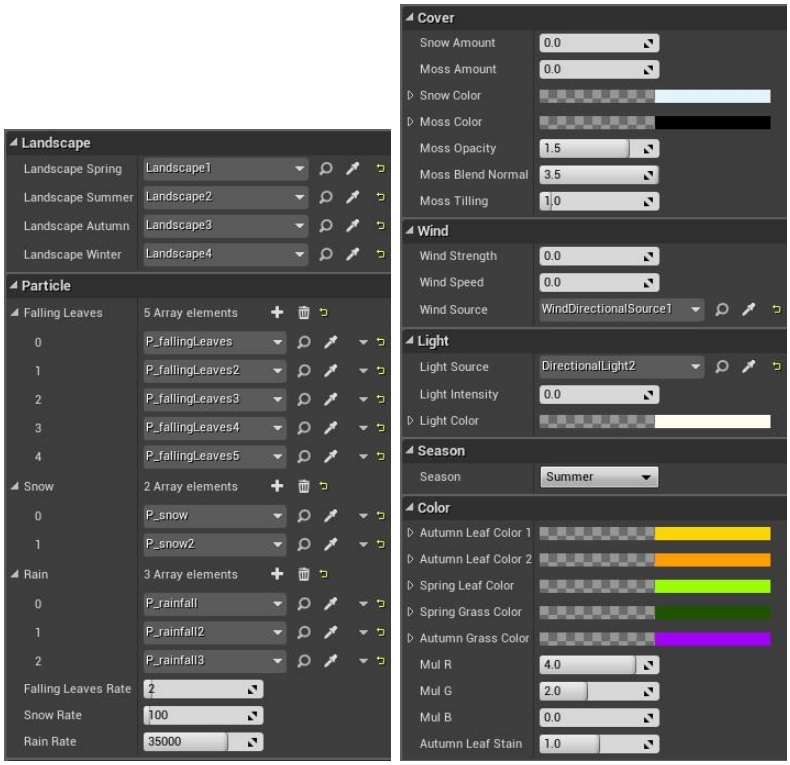
v tabulce 4.5. Hned na první pohled je zřejmé, že je ve scéně výrazně menší množství trojúhelníků, než je počet modelů krát počet trojúhelníků daného modelu. Ve všech testovacích scénách se vkládaly stromy a květiny pomocí nástroje Foliage. Tento nástroj vytváří instance, které jsou automaticky seskupovány dohromady (stejné modely zastoupeny pouze jednou) a při vykreslování se použije hardware instancování. Díky tomu může být mnoho instancí vyrenderováno se zavoláním pouze jednoho kreslení.
Také jsem umožnila interakci hráče s prostředím (sekce 4.4). Jako první jsem vytvořila možnost zanechávat stopy ve sněhu. Pro tento účel jsem vytvořila speciální mapu, kde materiál krajiny simuloval sníh. Sněhu lze nastavit výšku a při pohybu postavy (pohled ze shora) se sníh deformuje tak, že se sníží vrstva (sešlápnutý sníh) sněhu na minimum. Ve druhé interakci je možné odstraňovat a vkládat modely vegetace. Odstranění je různé. Stromy se skácí (pouze pád, nikoliv narušení kmene), kytku lze rozseknout na mnoho malých částí, které se rozletí a kapradí úplně zmizí. Při vkládání lze přepínat mezi modely.

4.1 Nástroj na ovládání ročního období

Tento nástroj je ve formě *blueprint* (viz obr. 4.2) a umožňuje měnit krajinu z původní (letní) podoby do jarní, zimní a podzimní, pomocí následujících atributů

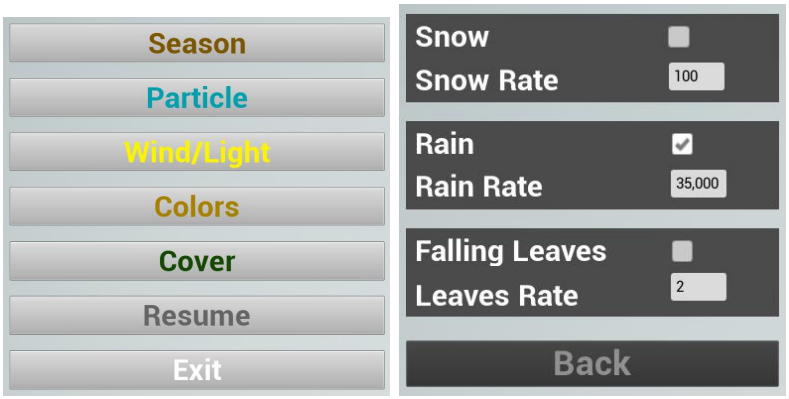
- Roční období - jaro, léto, podzim, zima (viz obr. 4.6), 4.7, 4.8 a 4.9)
- Vítr - síla a rychlost
- Světlo - intenzita a barva

- Pokrytí sněhem - míra pokrytí, barva sněhu (viz obr. 4.14)
- Pokrytí mechem - míra pokrytí, barva mechu, průhlednost a velikost vzorku (viz obr. 4.14)
- Barvy - barva jarní trávy a listů, barva podzimních listů a podzimní (zároveň zimní) trávy, flekatost podzimních listů (viz obr. 4.15)
- Particle - sněhu, deště a padajících listů a jednotlivé intenzity (viz obr. 4.13, 4.11 a 4.12)

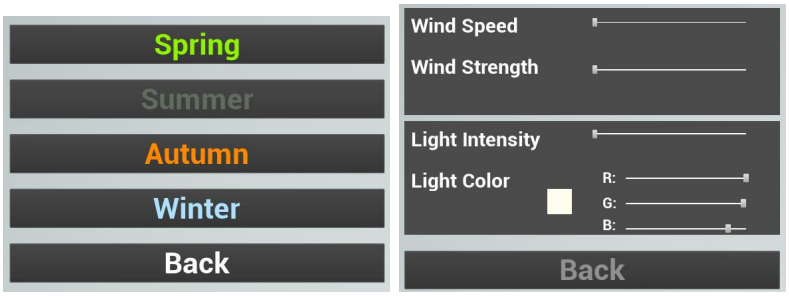


Obrázek 4.2: Blueprint na ovládání ročního období

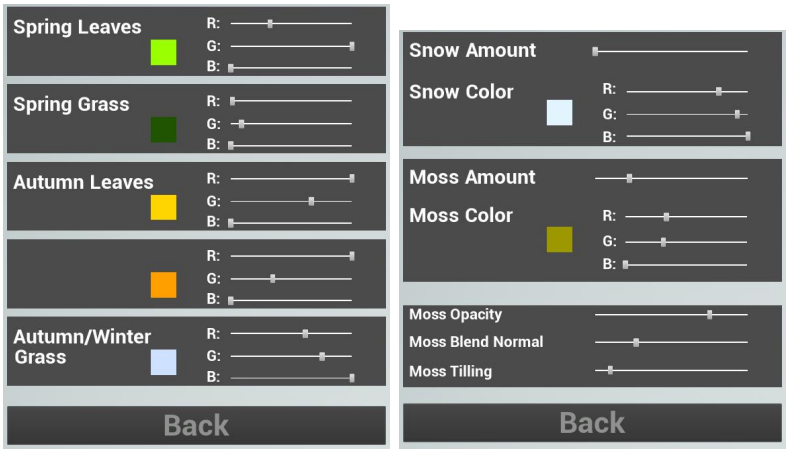
Tyto změny lze ovládat dvěma způsoby. Prvním je změna proměnných přímo v editoru (viz obr. 4.2). Nejprve je třeba vložit *Blueprint* do scény a přiřadit mu jednolivé objekty, které ovládá. Poté lze přistoupit k detailům z grafu scény, nebo kliknutím na *Blueprint* ve scéně. Druhý způsob je přímo za běhu hry, kdy lze vyvolat *menu* (viz obr. 4.3, 4.4 a 4.5). Díky němu lze ovládat téměř všechny atributy, jako v editoru. Výjimkou jsou některé konstanty, které si lze vhodně přednastavit a není třeba je dále měnit.



Obrázek 4.3: Menu ve hře na ovládání ročního období a nastavení částicových systémů



Obrázek 4.4: Menu ve hře na ovládání ročního období, nastavení rořního období, větru a světla



Obrázek 4.5: Menu ve hře na ovládání ročního období, nastavení barev a pokrytí

4.1.1 Rozhodnutí při vytváření nástroje

Při vytváření nástroje jsem postupně narazila na některé hranice UE4 a musela jsem se tak rozhodnout, jak tyto problémy vyřešit. Hlavní problém byl změna materiálu krajiny. Nejdříve jsem si zkoušela změnu materiálu na kostce, kde jsem používala materiál dřeva a cihel. Změna proběhla za běhu hry v pořádku a ihned. Poté jsem chtěl to samé aplikovat na krajinu, ale narazila jsem na chybovou hlášku, že nelze napojit na změnu materiálu jako cílový objekt krajinu. V této chvíli nelze vyměnit materiál krajiny za běhu hry. Jedná se o velmi složitý materiál, na který mohou být napojeny i modely rostlin, a proto nelze tyto materiály změnit jinak, než přímo v editoru v detailech dané krajiny. Proto jsem zvolila řešení, které není úplně dokonalé, ale funguje. Nejprve si vytvořím krajinu s materiálem. Použiji vrstvy, podle toho jak potřebuji a následně tuto krajinu třikrát nakopíruji. Poté každé krajině nastavím jiný materiál (jaro, léto, podzim, zima) a tak se mi přiřadí správné vsrtvy. V nástroji tyto čtyři krajiny přiřadim pod konkrétní roční období. Tento krkolomný způsob by nebyl třeba, pokud by mi stačilo ovládání v editoru. Bylo vytvořeno především pro možnost měnit roční období i s modely vegetace za hry.

Obrázek 4.6: Jarní krajina

Dále bylo třeba vybrat typ světel. Pro lepší vzhled jsem přidala do scény atmosferickou mlhu (*Atmosferic Fog*) a světlo z oblohy (*Sky Light*). Jako hlavní světelný zdroj jsem zvolila přímé světlo (*Directional Light*), které je ideální volbou pro simulaci světla přicházejícího ze slunce. Díky změně intenzity osvětelní a změně barvy lze dotvářet denní scénu (viz obr. 4.10). V UE4 se používají tři typy světel a to *Static*, *Stationary* a *Movable*. Statická světla jsou nepohyblivá a nelze ani měnit žádné parametry. Se světlem stacionárním

Obrázek 4.7: Letní krajina

Obrázek 4.8: Podzimní krajina

Obrázek 4.9: Zimní krajina

nelze také pohybovat, ale lze mu měnit intenzitu a barvu. Dynamická světla se mohou pohybovat i měnit. Protože jsem potřebovala měnit intenzitu a barvu, vybrala jsem si stacionární světla. Ty je však potřeba předpočítat v editoru před spuštěním hry. Při tomto procesu se vytváří *Lightmap*, *Shadowmap* za použití *Distance Field Shadow*. Při každém pohybu objektu ve scéně, jako přidání stromu, posunutí stromu, se musí světla znovu přepočítat. Zde jsem pozdější narazila na další problém, protože jsem vytvářela interakci s modely ve scéně. Mohla jsem mazat a přidávat jednotlivé modely do scény za běhu hry, ale v UE4 není žádná možnost, jak přistoupit k informacím o světelech a například vypnout některé stíny. Proto zde není ani možnost, jak znovu přepočítat světla za běhu hry. Proto jsem nakonec zvolila světla dynamická.
Dále bylo potřeba vyřešit, jak se bude přistupovat k částicovým systémům.

Obrázek 4.10: Různá poloha, intenzita a barva dynamických světel

Samotné částice vytváří emitor (angl. *emitter*). Emitor se chová jako zdroj částic a umisťuje se do 3D prostoru. U sněžení a deště je možnost „připnout“ emitor na charakter hráče. V nastavení se zvolí pozice umístění, která je vztažena k souřadnému systému charakteru a lze tak umístit emitor, aby se pohyboval s hráčem a částice se objevovaly kolem hráče, který je vidí při průchodu scénou, přestože se ve skutečnosti nachází pouze na malém území. Druhá možnost byla umisťovat emitory do scény. Nakonec jsem zvolila obě možnosti. Díky tomu je možno simulovat déšť a sníh pouze na částech scény. Zároveň lze zapínat a vypínat emitor připojený na charakter, nezávisle na zapnutí/vypnutí emitorů umístěných ve scéně. U částic padajících listů je možné pouze umisťovat emitory do scény, protože by šlo umístit zdroj podle pozice stromů, ale bylo by komplikované volit, které stromy budou vybrány a které ne. Jelikož jsem chtěla mít možnost umístit více emitorů stejného typu, zvolila jsem pole emitorů, kde si sám uživatel volí velikost pole a konkrétní emitory, které chce ovládat.

Obrázek 4.11: Particle deště

Obrázek 4.12: Particle padajících listů

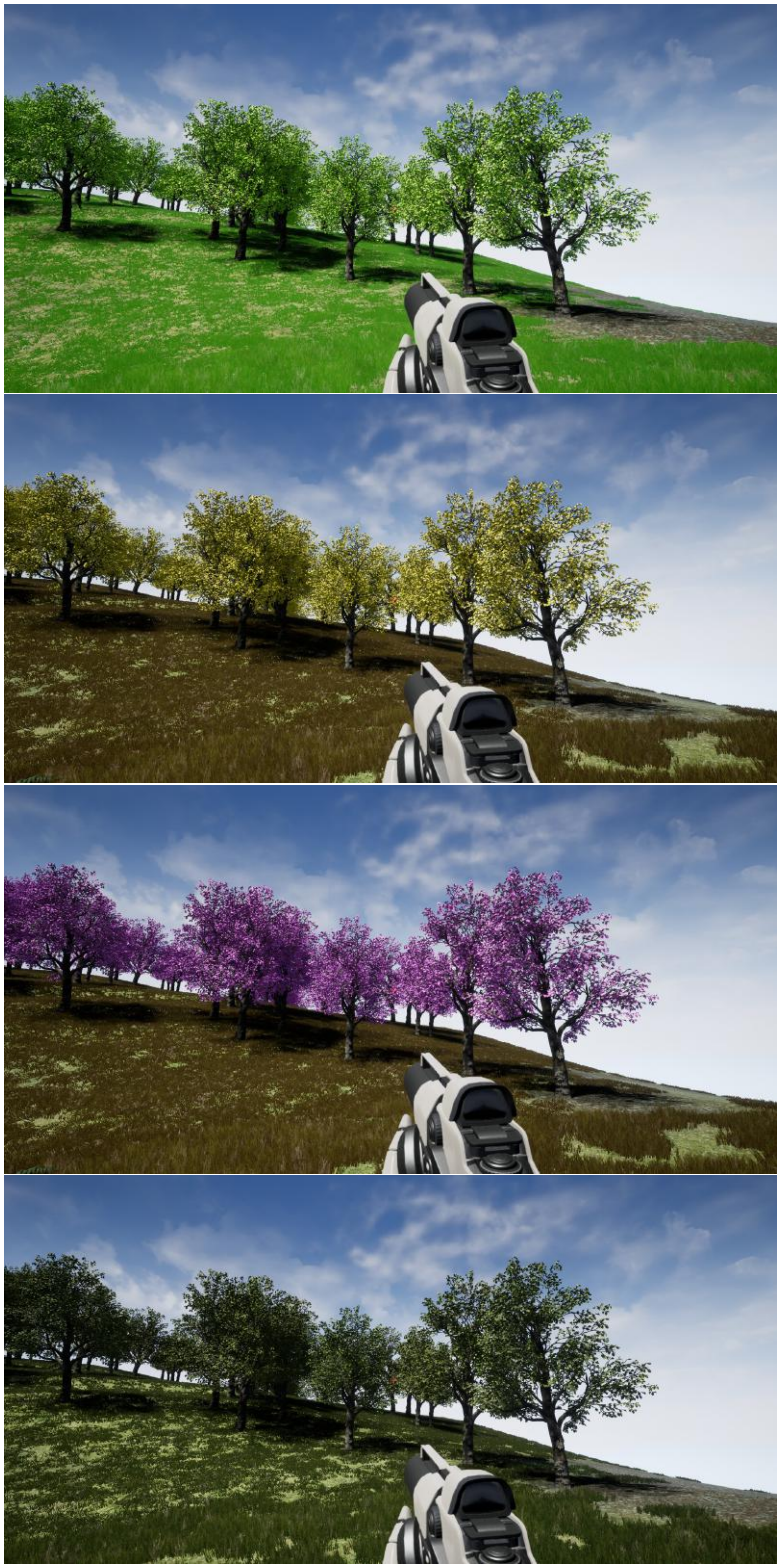
Obrázek 4.13: Particle sněhu

Obrázek 4.14: Pokrytí sněhem a mechem

4.2 Vytvoření scén pro testování

Testování hranic zobrazení, co se týká kvantity, jsem si rozdělila na dvě části (ukázka testovacích krajin viz obr. 4.16). Pro první testování jsem vytvořila tři různě velké scény. Podle rozlohy krajiny jsem si je rozdělila na malou, střední a větší. Ve všech třech scénách se opakoval stejný postup na vytvoření krajiny a zaplnění modely. Hlavním rozdílem byla velikost krajiny. Samozřejmě jsou si tyto krajiny podobné, ale rozhodně nejsou stejné, pouze jsem se snažila vytvořit stejné situace opakovaně, proto jsem vždy zahrnula následující:

- Složitý materiál krajiny (*Landscape*), obsahující 8 vrstev, z nichž tři obsahují *Grass Tool*.



Obrázek 4.15: Změna barev stromů a trávy

- Postupné použití všech materiálů krajiny.
- Stromy s různým zahuštěním podél scény.
- Další modely, přidané nástrojem *Foliage* v různé hustotě.
- Kameny.
- Jezírko.
- Různé zvlněný terén.

Poté jsem volila průchod scénou, aby kamera postupně narazila na všechny zmíněné objekty. Nastavila jsem průchod scény po pomyslné spirále. Nejdříve se charakter pohybuje po vnější části scény a následně se pomalu stočí do vnitřní části, kde ještě projde lesem, aby byl pohled jak z vnějšku na les, tak ze vnitřku.
Ve druhém testování využívám pouze jednu scénu, kterou jsem však v průběhů testů měnila a to tím způsobem, že se mění hustota stromů. Tato scéna, stejně jako tři předešlé obsahuje stejné ukázky použití různých modelů. V této scéně jsem zvolila průchod scénou po cestě do kopce a následně po stejné trase zpět. Ke konci se trasa uhýbá do lesa, aby byl zahrnut i pohled z vnitřku lesa.

4.3 Testování

Testování proběhlo na sestavě GenuineIntel Intel(R) Core(TM) i5-7400 CPU, frekvence 3.00GHz, NVIDIA GeForce GTX 1060 3GB, OS Windows 10. Jak jsem již psala, rozdělila jsem si testování na dvě části. V první části se snažím zjistit, jak se změní vytížení zobrazení při změně nastavení a velikosti scény. Toto měření však není moc vypovídající, protože jsem rozmisťovala objekty náhodně, pouze jsem se snažila všechny umístit tak, aby se na ně v průchodu scénou zaměřila kamera. Druhé testování se zaměřuje na složité modely vegetace, v tomto případě stromy, a do jaké míry ovlivňují zobrazování scény. Během testů se mění počet stromů ve scéně a další nastavení. V obou případech jsem měřila *Frame rate*, tedy počet snímků za vteřinu (FPS). Během testů jsem se zaměřila na následující proměnné

- Světla - stacionární/dynamická
- Vítr - stacionární/dynamická scéna
- LOD stromů - 1 LOD/3 LOD

Obrázek 4.16: Ukázka testovacích krajin. Nahoře malá krajina z prvního testu, dole krajina z druhého testu.

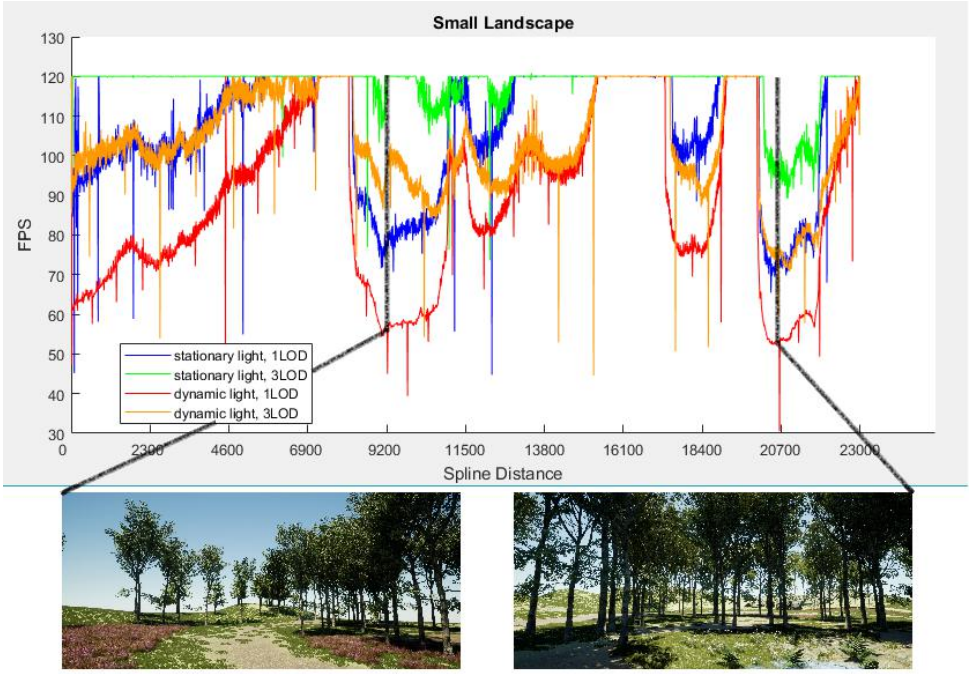
- Počet stromů - změna zahuštění ve scéně
- Velikost krajiny - rozloha
- Detaily - vysoké/střední

Abych zajistila pokaždé stejný průchod danou scénou, vytvořila jsem si *Blueprint*, který má v sobě *Spline* (viz obr. 4.17). Díky tomu jsem definovala cestu, po které se bude postava pohybovat. Na počkátku je vzdálenost uražené cesty nulová. Při každém zavolání události *EventTick* (volá se každý frame) se přičte ke stávající vzdálenosti konstanta posunu (10 nebo 20 cm) a nastaví se pozice a otočení charakteru. Aby začínalo měření a končilo na stejném místě, přidala jsem také dva spouštěcí boxy (*Trigger Box*), kde první zapne měření a druhý ho ukončí. Měření provádím z pohledu první osoby (First Person) a proto jsem přidala, aby se kamera otáčela podle toho, jak se zahýbá a mění samotná trajektorie. Díky tomu se kamera otáčí a zabírá různé pohledy ve scéně, jak na vnitřní část scény, tak na vnější části.

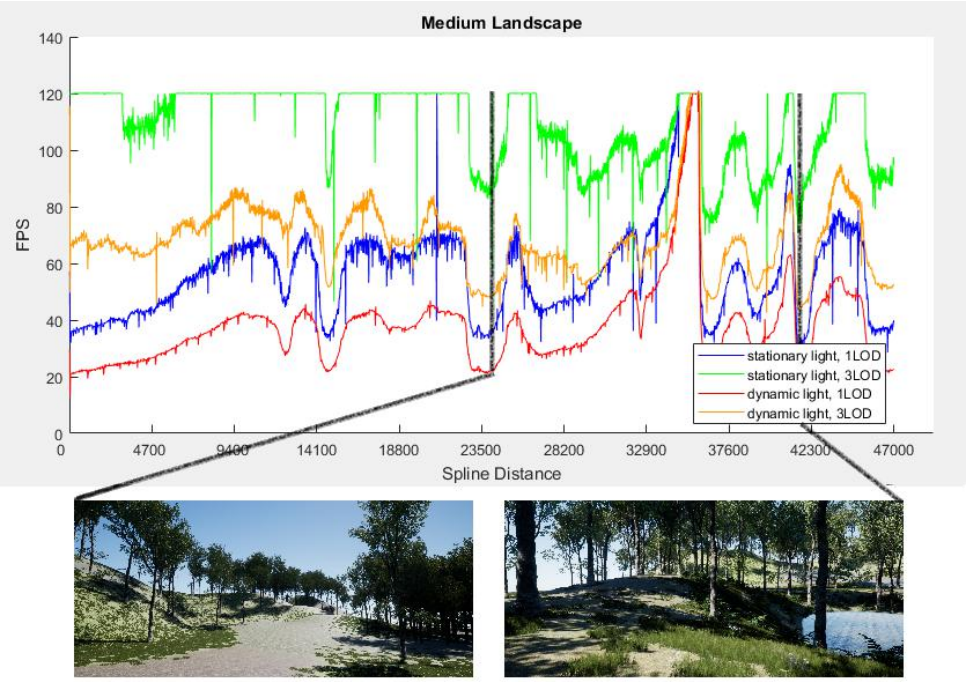
Obrázek 4.17: Ukázka Spline a Trigger Boxu

4.3.1 Testování různě rozlehlých scén

Na všech třech krajinách jsem testovala následující parametry: vítr, LOD, světla. Z naměřených údajů jsem vytvořila grafy, abych porovnala, jak se mění FPS v závislosti na průchodu různými částmi scény s různými nastaveními. Jelikož jsem po několika měřeních zjistila, že má vliv větru minimální vliv na FPS, tak jsem toto měření nezahrnula do porovnání v grafu. Díky tomu mi pro každou scénu zůstaly čtyři naměřené množiny dat, namísto původních osmi, a proto jsem se rozhodla vizualizovat naměřené výsledky pro každou scénu zvlášť do jednoho grafu, kde jsem barevně odlišila tyto čtyři měření.
Následně jsem procházela naměřená data a pomocí grafu určila, kde došlo k výraznému poklesu FPS. Pro testování jsou důležité zejména tyto lokální extrémy, kde jsou FPS minimální. Do těchto extrémů nepočítám krátké změny FPS v jednom, nebo druhém směru, kdy došlo ke změně pouze v jednom tiku a to o více jak 10FPS oproti sousedním snímkům. Jednalo se o chyby při měření a nemá tedy smysl se jimi zabývat. Pro relevantní výsledky je třeba se zabývat spojitým a delším poklesem FPS. Poté jsem scénu znovu procházela a pořízovala obrázky míst, kde k těmto poklesům došlo. Některé poklesy bylo možno předpokládat již při tvorbě scény (např. veliká koncentrace vegetace). Výsledné grafy jsem vizualizovala (viz obr. 4.18, 4.19 a 4.20) pomocí programu *Matlab*. Jelikož by bylo nepřehledné zahrnout porovnání všech scén do jednoho grafu, přidala jsem také tabulku průměrných FPS pro všechny testovací scény (tabulka viz. 4.6).



Obrázek 4.18: Graf měření FPS na malé scéně, doplněný dvěma pohledy ze hry, při kterých došlo k největšímu poklesu FPS



Obrázek 4.19: Graf měření FPS na střední scéně, doplněný dvěma pohledy ze hry, při kterých došlo k největšímu poklesu FPS

	stacionární		dynamická	
	3LOD	1LOD	3LOD	1LOD
malá krajina bez větru	116.35	103.15	100.87	82.41
malá krajina s větrem	116.53	103.30	100.10	82.57
střední krajina bez větru	105.90	53.33	66.05	33.52
střední krajina s větrem	105.69	53.18	66.17	33.95
velká krajina bez větru	82.63	28.65	47.88	19.22
velká krajina s větrem	82.49	28.98	47.95	19.06

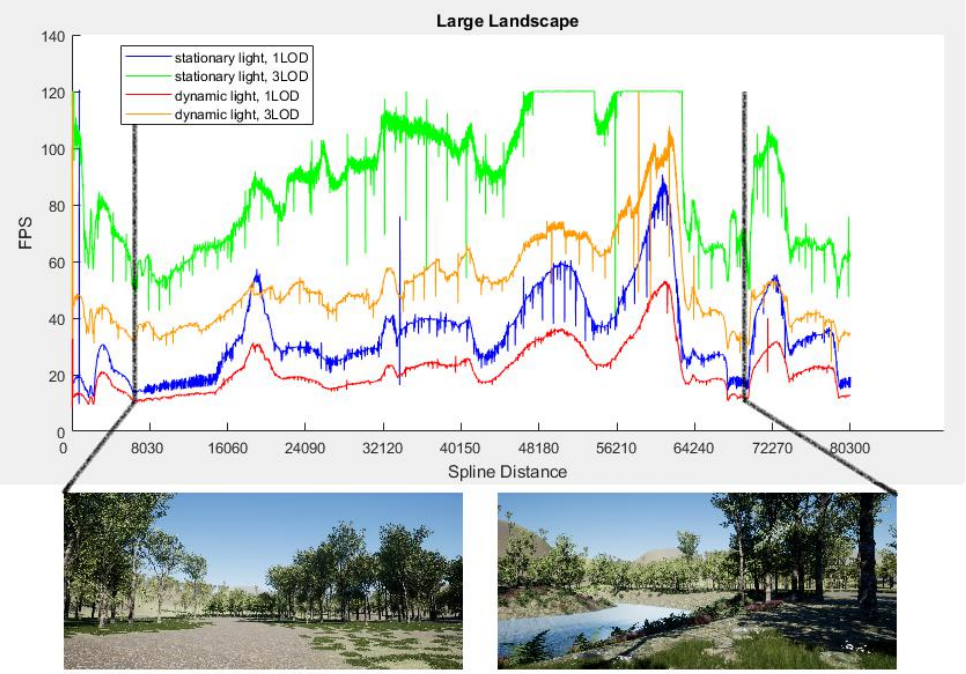
Tabulka 4.6: Průměrné FPS jednotlivých měření v prvním testování

4.3.2 Diskuze

Z naměřených hodnot je na první pohled zřejmé, že jsou zde veliké rozdíly mezi FPS jednotlivých měření. Ze všech tří grafů je patrné, že dochází k poklesům a nárůstu FPS na stejných místech v krajině, nezávisle na parametrech měření. Nejlepších FPS dosahuje nastavení stacionární světla a 3LOD. Nejhorších FPS dynamická světla a 1LOD. Což odpovídá i předpokladům, protože dynamická světla zatěžují vykreslování scény více, než stacionární a více LOD zatěžuje scénu méně, než jedno LOD. Zajímavější dvojice je stacionární světla a 1LOD, ve srovnání s dynamická světla a 3LOD. Jak je z grafu vidět, na malé scéně jsou FPS podobná. Z počátku jsou téměř totožná, poté mají dynamická světla a 3LOD lepší FPS a následně horší. Ve střední scéně už je rozdělení jasnější. Až na malé výjimky má více FPS dynamická světla a 3 LOD. V největší scéně je rozdělené ještě více zřejmé. Pouze na dvou místech, tam kde je kombinace stacionární světla a 1LOD v lokálních minimech, dosahuje horších výsledků dynamická světla a 3LOD. Z těchto výsledků vyplývá, že se zvyšující se plochou krajiny (rozsáhlost) má větší váhu na zobrazení složitost vegetace, než volba světel.
Jak jsem dříve zmínila, z testů také vyplynulo, že nastavení větru nemá na měření téměř žádný vliv. Z tabulky průměrných FPS (viz 4.6) je na první pohled vidět, že jsou si hodnoty měřené za stejných, podmínek pouze s rozdílným větrem, velmi podobné. Dokonce je v některých případech lepší průměrné FPS z měření s větrem. Tyto drobné nesrovnalosti jsou způsobeny především odchylkami v měření a občasnými výstřely FPS do extrémů.

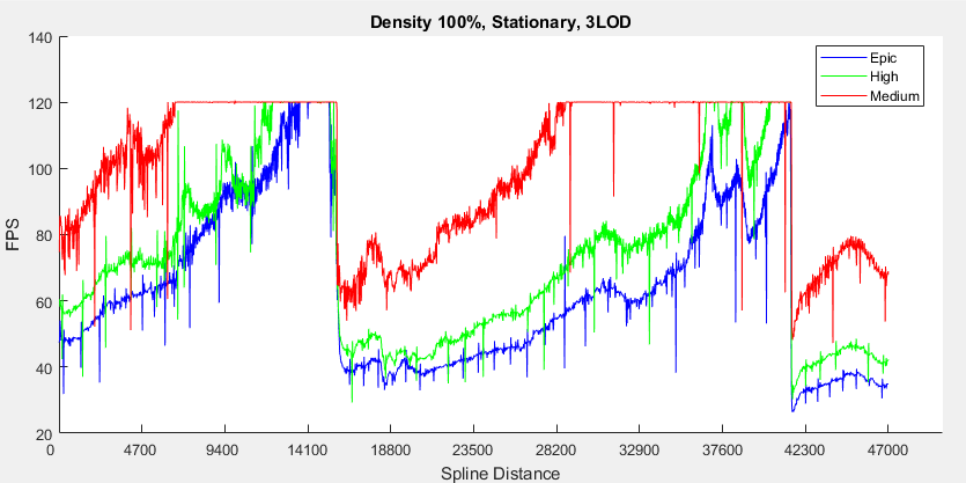
4.3.3 Testování různé hustoty stromů ve scéně

Při druhém testování jsem používala pouze jednu krajinu. Postupně jsem snižovala počet stromů v krajině. Začala jsem na zahuštěné krajině, kterou jsem považovala za 100% počet stromů. V této krajině se nachází 715 stromů.

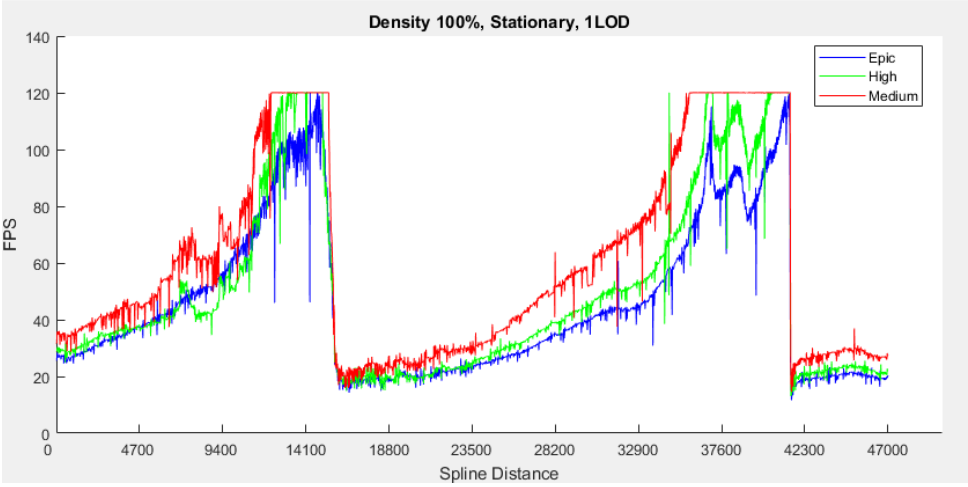


Obrázek 4.20: Graf měření FPS na velké scéně, doplněný dvěma pohledy ze hry, při kterých došlo k největšímu poklesu FPS

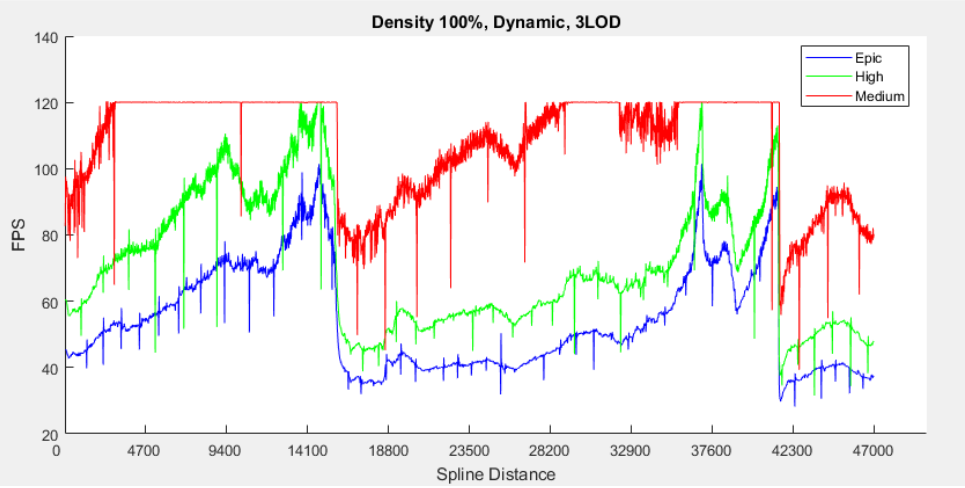
Ubírala jsem počet po 20% až na 0 stromů v krajině. Díky tomu mi vzniklo šest různých zahuštění (počet stromů 715, 572, 429, 286, 143, 0). Pro každou různě zahuštěnou scénu jsem testovala různá světla, LOD a detaily hry. Výsledky jsem shrnula do tabulky (viz 4.7). Některé změny FPS jsem vizualizovala pomocí grafu (grafy pro 100% zahuštění 4.21, 4.22, 4.23, 4.24 a 4.23).
Co se týká kvality (detailů), nabízí UE4 možnosti *Low, Medium, High, Epic a Cinematic*. Toto nastavení se dále dělí na jednotlivé možnosti: *View Distance, AntiAliasing, Post Processing, Shadows, Textures, Effects a Foliage*. Nejprve jsem vyzkoušela testování jednoltivých nastavení, abych zjistila kvalitu výsledku. Nastavení na *Epic* odpovídá 100% zobrazení, vynechala jsem proto verzi *Cinematic*, protože mi přišla zbytečná (Cinematic je lepší než Epic). Také jsem vyzkoušela detaily na Low, ale vegetace vypadala natolik špatně, že by takové zobrazení nebylo možno využít ani jako doplnění ve hře, proto jsem ho také vynechala. Testovala jsem tedy kvalitu *Medium, High a Epic*, ale změnila jsem u možnosti *AntiAliasing* nastavení na *Epic* u všech tří možností. Při nižší kvalitě stromy a tráva velice nepříjemně zrnily, ale s nastavením na *Epic* bylo vše v pořádku a nevytvářelo rušivý efekt.



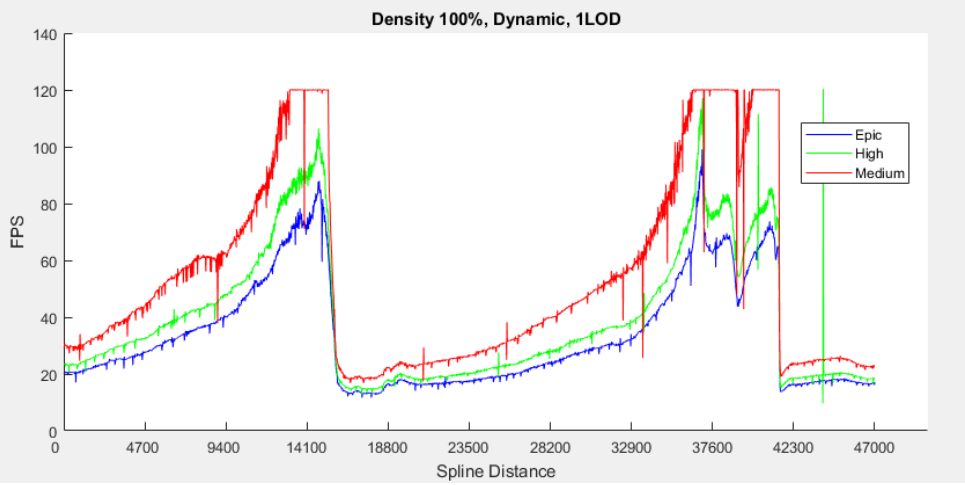
Obrázek 4.21: Testování FPS na druhé scéně. Hustota 100%, stacionární světla, 3LOD



Obrázek 4.22: Testování FPS na druhé scéně. Hustota 100%, stacionární světla, 1LOD



Obrázek 4.23: Testování FPS na druhé scéně. Hustota 100%, dynamická světla, 3LOD



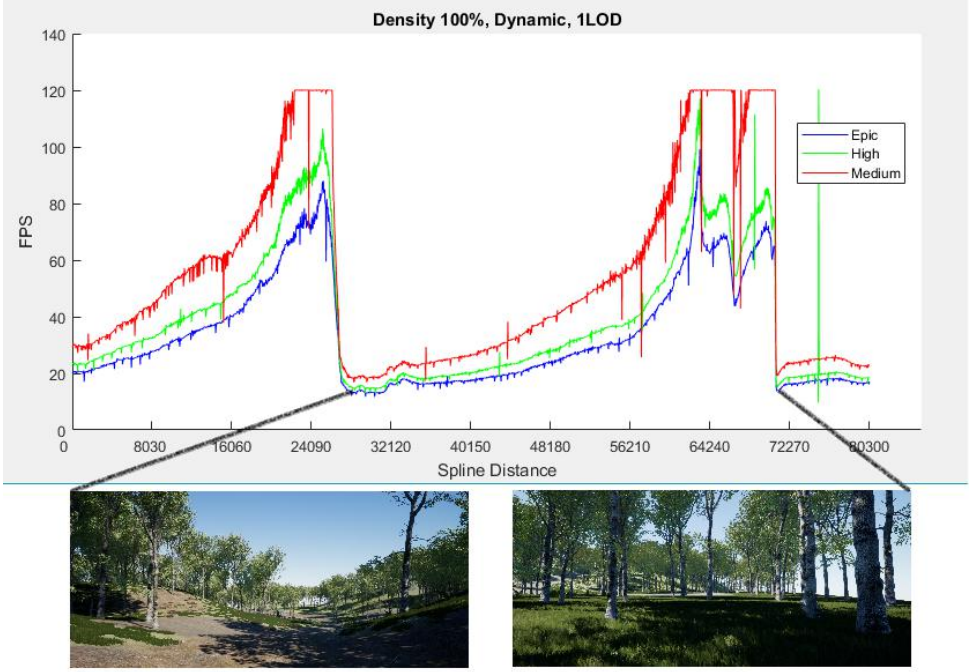
Obrázek 4.24: Testování FPS na druhé scéně. Hustota 100%, dynamická světla, 1LOD

Krajina	Světla	1LOD			3LOD		
		Epic	High	Medium	Epic	High	Medium
1.	stacionární	33.16	35.53	43.91	55.24	65.26	95.26
	dynamická	25.47	29.33	38.65	50.14	65.78	105.08
2.	stacionární	35.56	39.61	50.48	56.59	66.87	98.04
	dynamická	28.84	33.94	45.12	51.94	68.46	108.63
3.	stacionární	40.25	44.56	57.82	56.32	67.14	98.20
	dynamická	33.58	40.43	55.70	53.43	71.10	109.03
4.	stacionární	45.00	50.83	68.24	58.93	69.74	101.58
	dynamická	38.53	47.32	66.90	55.51	73.97	110.05
5.	stacionární	51.33	59.57	83.58	58.50	69.91	101.47
	dynamická	46.63	58.64	84.68	56.36	74.87	110.61
6.	stacionární	61.12	73.18	104.64	61.12	73.18	104.64
	dynamická	60.73	79.78	115.26	60.73	79.78	115.26

Tabulka 4.7: Průměrné FPS jednotlivých měření v druhém testování

4.3.4 Diskuze

Z tabulky průměrných FPS (viz 4.7) je vidět, že se hodnoty průměrných FPS zvyšují s klesajícími detaily (Epic -> High -> Medium). Také jsou hodnoty FPS vyšší pro stacionární světla, než pro dynamická a pro 3LOD, než pro 1LOD. Průměrné FPS se zvyšuje i pro klesající hustotu stromů ve scéně. Všechny předešlé poměry byly očekávané, protože při náročnější volbě bylo očekáváno nižší FPS.
Nečekaný poměr průměrných FPS nastal při porovnání stacionární a dynamické varianty světel u detailů *Medium* a *High*. Očekávaný výsledek je, že jsou FPS menší u použití dynamických světel oproti stacionárním při zachování ostatních nastavení. U některých nastavení tomu bylo však naopak. U volby 1LOD nastal tento případ pro detaily *High* pro scénu 6, pro detaily Medium pro scénu 5 a 6. Pro volbu 3LOD pro detaily *High* a *Medium* pro všechny scény. Postupně jsem procházela různá nastavení a zjistila jsem, že je tento rozdíl v FPS způsoben pravděpodobně způsobem vytváření a uchovávání stínů. Za použití GPU Visualizer nástroje jsem porovnala stacionární a dynamická světla u krajiny 5 pro 3LOD a Medium detaily. Ve světlech je jednou z částí ShadowedLight. Dynamická světla zde měly hodnotu 0.15 ms na frame a stacionární světla 5.24 ms na frame. Což znamená rozdíl 5ms na každý frame.



Obrázek 4.25: Ukázka míst v testovací scéně, kde dochází k největším poklesům FPS

4.4 Interakce s prostředím

Jako ukázku interakce s prostředím jsem vytvořila dvě odlišné použití. Tím prvním je chůze ve sněhu se zanecháváním stopy. V té druhé se jedná o manipulaci s vegetací, konkrétně vkládání, mazání, kácení a rozseknutí.
Pro chůzi ve sněhu jsem si vytvořila speciální materiál, kterému lze nastavit výšku a použila ho na krajinu. Při této interakci používám pohled třetí osoby. Nejprve mi osoba procházela skrz materiál a nezanechávala žádnou stopu. Dále jsem nastavila, aby chodila po sněhu. Nakonec jsem nastavila, aby při šlápnutí chodidlem na zem (levým nebo pravým) se sníh deformoval. Pro tento účel jsem si vytvořila jednoduchou texturu stopy za použití 2D nástroje GIMP. Podle tvaru stopy se sníží vrstva sněhu na minimum. Pokud hráč používá skoky, nedeformuje se pod ním žádný sníh, až při dopadu (ukázka viz obr. 4.1).
Při interakci s vegetací využívám modely stromu, kapradí a kytky (viz obr.4.1). Tyto tři modely mohu vkládat pomocí tlačítka klávesnice a dalšími přepínat, který model se aktuálně bude používat. Jiným tlačítkem mohu tyto modely odstranit. Pro strom jsem použila kácení, kdy je modelu stromu nastavena fyzikální váha a strom spadne na zem. Kapradí se uplně vymaže a květina je rozseknuta na kousky, které se volně rozlétnou a dopadnou na zem. Vkládání a mazání jsem také přidala k nástroji na ovládání vegetace a tak je možnost si

vegetaci za hry do určité míry upravovat. Kácení a rozseknutí jsem vynechala, aby nevznikal ve scéně zbytečný nepořádek.

4.5 Problémy

V této sekci se zmíním o problémech, se kterými jsem se setkala během používání UE4.
Během posledního roku byly k dispozici celkem tři verze. Nejprve jsem svoji práci vytvářela ve verzi 4.13, následně 4.14 a nakonec 4.15. V tuto chvíli se již připravuje verze 4.16. Všechna testování, realizace nástroje na ovládání ročních období a interakce s prostředím byla provedena v aktuálně nejnovější verzi 4.15. V každé verzi vycházejí nová vylepšení a funkce a zároveň dochází k opravám problematických částí. Díky těmto změnám se může změnit i celkový přístup a používání některých nástrojů a je třeba upravit projekt, aby byl komaptibilní. Poté co vyjde nová verze se objevují chyby, které jsou sice nahlášeny, ale nějaký čas trvá, než se najde řešení, jak problém vyřešit, nebo se chyba opraví. Některé chyby způsobují špatnou funkčnost nástrojů, ale může docházet i k pádům editoru.
Pro práci s UE4 jsou popsány doporučené hardwarové a softwarové specifikace. Pro horší komponenty, než jsou minimální specifikace není zaručeno, že bude UE4 správně fungovat. Pro hardware s využitím operačního systému Windows jsou doporučeny: procesor Quad-core Intel or AMD, 2.5 GHz nebo rychlejší, 8 GB RAM, grafická karta kompatibilní s DirectX 11, Windows 7/8 64-bit. Já sama jsem nejdříve požívala svůj notebook, který byl bohužel nedostačující, což se projevovalo extrémně častými pády editoru, zvláště při úpravách materiálu a manipulací s vegetací. Zároveň mi nefungovaly některé nástroje jako GrassTool. Po přechodu na stolní počítač mi nástroje fungovaly správně a k pádu editoru došlo v ojedinělých případech. Bohužel jsem se ke konci své práce setkala s velmi nepříjemnou chybou. V UE4 je možnost vytvořit balíček z projektu a získat tak spustitelný .exe soubor. Tyto balíčky se mi vytvářely bez chybových hlášení a vše fungovalo správně. Poté co jsem k nástroji na ovládání ročního období přidala možnost manipulovat s parametry za běhu hry, tedy menu, začaly se mi objevovat chyby. Paradoxně vše při spuštění hry v editoru fungovalo správně, změna hodnot měla ihned odezvu, vše správně komunikovalo. Při vytváření balíčku jsem byla upozorněna, že nebyly nalezeny odkazy na všechny položky v menu (tlačítka, obrázky, textová pole apod.). Po krátkém hledání jsem nalezla v diskuzích uživatele se stejným problémem, ale nikde nebylo řešení. Já sama jsem psala na diskuzní fórum ohledně této chyby, ale doteď se mi nedostalo odpovědi. Chybu jsem nevyřešila. Naštěstí jsem zjistila, že lze přesunout projekt na můj notebook, kde je možné vytvořit balíček nez problémů a chybových hlášení.

Závěr

V rámci této práce byly popsány metody generování a zobrazení vegetace a popsány možností herního engine, konkrétně Unreal Engine. Následně byly popsány dostupné nástroje, které umožňují manipulaci s vegetací v Unreal Engine 4. Na základně použití těchto nástrojů a skriptování byl vytvořen blueprint pro ovládání ročního období, který byl rozšířen o možnost ovládání ze hry pomocí menu. Použití toho nástroje bylo popsáno a byly přiloženy ukázky použití na testovací scéně. Dále byla provedena série testování, kde se porovnaly různě kvalitní a komplexní zobrazení scény. Výsledky těchto měření byly zahrnuty do tabulek a grafů. V poslední části byly ukázány dvě interakce hráče ve hře. První je chození sněhem se zanecháváním stop, druhá je přidávání a odebírání (kácení, rozseknutí, vymazání) vegetace ve scéně. Vkládání a mazání bylo přidáno do nástroje a umožňuje interakci hráče s vegetací za běhu hry.

Pro účely testování byly vytvořeny různě rozsáhlé scény s různým počtem

rozloha [m^2]	počet stromů	počet květin	počet trav
255 025	5300	53 300	712 000

Tabulka 4.8: Rozloha a počty objektů ve scéně

stromů. Pro nejvyšší detaily, dynamická světla a zobrazení stromů pouze s jedním LOD (maximální detaily, nedochází k přechodu mezi LOD, nenastává žádný rušivý efekt) klesly FPS v některých místech na velmi nízkou hodnotu, pro kterou již není obraz plynulý a seká se. Taková místa nastala především, když pohled hráče zabíral příliš mnoho stromů. Vytvořila jsem plně zahuštěnou krajinu (rozloha a počty objektů viz. tabulka 4.8). Použila jsem jednoduchý materiál na krajinu, model jednoho stromu, trávy a květiny. V krajině jsem

vytvořila lehce zvlněný povrch. Krajinou je možno se pohybovat v reálném čase, avšak hodnota FPS se výrazně mění. Navíc, pokud hodnota FPS příliš klesne (cca 5 FPS), přestane se zobrazovat tráva (viz obr. 4.10), ale vše ostatní se zobrazuje. Při pohybu krajinou je možno sledovat pokles FPS ze 120 až na 5. Čím méně stromů je v záběru hráče, tím větší FPS jsou. Pokud se v některém místě vypne zobrazování trávy, zvýší se tím počet FPS. Při zobrazování scény tedy nehraje až takovou roli, jak rozsáhlá je, ale především jak moc stromů se nachází v záběru kamery.

Obrázek 4.26: Ukázka dvou míst ve scéně, které jsou blízko sebe. Na obrázku nahoře se nezobrazuje tráve. Na obrázku dole se zobrazuje tráva.

Možné rozšíření do budoucna

Moji práci lze rozšířit v základě dvěma směry. Tím prvním je přidání dalších nastavení a ovládání vegetace a druhým je zlepšení a zrychlení zobrazení.
Největší problém je, že se Unreal Engine stále vyvíjí a přináší nové možnosti a zároveň je zobrazování komplexní vegetace velice rozsáhlé téma, a proto je zapotřebí se zaměřit pouze na některé části, aby bylo možné kvalitní a podrobné prozkoumání a zpracování. Já sama jsem se s herním Enginem setkala poprvé před rokem, kdy jsem měla předmět Počítačové hry a vytvářeli jsme 2D hru v Unity. Poté jsem se dostala k zadání této práce a poprvé v životě jsem použila Unreal Engine a začala pracovat ve 3D prostředí. Proto mi zabralo mnoho času se naučit UE4 používat alespoň na základní úrovni a dále jsem mohla nacházet nové funkce a nástroje, které jsem dále využívala. Ve své práci jsem se musela prokousat mnoha rozdílnými částmi, a proto jsem je prozkoumala převážně povrchově. Nejprve jsem se učila pohybovat v editoru, používat zkratky, dále jsem se učila skriptovat, napojovat skripty. Poté jsem přešla ke zkoumání vegetace, vztahů, jak fungují jednotlivé části a jak se zobrazují. Naučila jsem se používat parametry, ovládat materiály ze skriptu a postupně jsem se dostala až k interakci s prostředím, vytvářením funkcí, vytváření funkčního menu a v neposlední řadě ke způsobu, jak vůbec testovat. Při testování byla nejjednoduší část vytvořit krajinu, pak jsem musela zajistit aby byla testování stejná a vytvořit systém, jak bude postava procházet scénou po stále stejné trajektorii, přitom se bude otáčet a budou se ukládat údaje o FPS. Zpočátku jsem si zkoušela funkčnost možných nástrojů a teprve po delší době jsem byla schopná navrhnout, jak spolu budou jednotlivé komponenty komunikovat, jak se bude předávat informace o změně parametrů. Mnohokrát jsem udělala chyby a musela začít od začátku, protože tudy cesta nevedla. Proto bych doporučila, aby se další zájemci o téma Unreal Engine a zobrazování komplexní vegetace zamysleli a vybrali si užší zaměření, které zpracují více do detailů, nebo se smířili s tím, že bude jejich práce více povrchová.

Rozšíření ovládání vegetace

Při vytváření nástroje mě napadla některá rozšíření, která jsem nestihla realizovat. Uvádím zde ta nejzajímavější.
Přidání dalších particle systémů a to především mlhy, padajících květů a padajících hrudek sněhu. V UE4 se sice nachází možnost přidat mlhu do scény, ale jedná se o uniformně rozloženou mlhu, která má vliv na světlo a zobrazování scény v dálce. Možným rozšířením je vytvořit neuniformní mlhu, která by se zobrazovala jako jednotlivé kusy mračen. Další dva particle systémy

jsou vhodné pro roční období a tím prvním je efekt padajících květů na jaře. V tomto případě je zaptořebí použít jiné modely stromů, nebo rozšířit stávající model o poupata a květy, aby měl tento particle systém ve scéně smysl. Samotná realizace je velmi podobná padajícím listům. Posledním particle systémem je padání hrudek sněhu ze stromů v zimě, kdy by docházelo k deformaci sněhu podobně jako při zanechávání stop ve sněhu. Samotné provedení by mohlo vycházet právě ze zmíněné interakce se sněhem.
Další možné rozšíření je přidání dalších proměnných k materiálům listů, a ještě více přiblížit jejich zobrazování tomu z reálného světa. Jedná se především o průhlednost listů za přímého slunce (a i barva, která je listem propouštěna) a odrazivost listů při dešti. Právě s jednotlivými prvky ve scéně, jako je intenzita světla a déšť by mohly být parametry propojeny. Stejně tak by šla vytvořit závislost mezi padáním listů a sílou větru, kde by vítr ovlivňoval množství, směr a rychlost padajících listů.
Při vytváření ročního období se nabízí možnost simulovat průběh jednotlivých změn během celého roku. Od růstu trávy a rostlin na jaře, změnu barev, opadání listů až po postupné vrstvení sněhové pokrývky a následném roztátí. Pro přirozenou simulaci je zapotřebí použít možnost *Morph*, kdy je možno objekty deformovat v určených místech. Díky této deformaci lze simulovat postupné zvětšování rostliny a růst jejich částí až po rozkvetení.
Možností jak rozšířit ovládání vegetace je nespočetně mnoho, záleží pouze na fantazii a dostupných možnostech.

Vylepšení zobrazování

UE4 nabízí širokou škálu nastavení. V realistickém zobrazení hraje roli mnoho parametrů, jako světla, post processing, materiály, odrazy, modely a stínování. Při zobrazování v UE4 lze ovlivnit v hlavním nastavení *View Distance, AntiAliasing, Post Processing, Shadowsn Textures, Effects, Foliage* na detaily *Low, Medium, High, Epic, Cinematic* a *Auto*. To je pouze hrubé nastavení, ve skutečnosti se za každou složkou skrývají další parametry, kterým lze nastavovat více hodnot. Všechna tato nastavení hrají velikou roli při zobrazování. V ideálním případě je třeba tyto parametry přizpůsobit konkrétní scéně a požadavkům, aby se dosáhlo kvalitního zobrazení a zároveň se co nejvíce snížily nároky na počítač. Další kritická nastavení jsou u světel, stínů, volbě LOD a kritérií, kdy se budou stupně LOD přepínat. Tímto způsobem by šlo efektivněji zobrazovat scény a zvýšily by se tím hodnoty FPS.

Literatura

[1] *Unreal Engine* [online]. Epic Games, ©2004-2017. Dostupné z: `https://www.unrealengine.com/what-is-unreal-engine-4`.

[2] Žára, Beneš, Sochor a Felkel. *Moderní počítačová grafika*. Vyd 2. Brno: Computer Press, 2010. ISBN 80-251-0454-0.

[3] *SpeedTree* [online]. Interactive Data Visualization, Inc. (IDV), ©2016. Dostupné z: `http://www.speedtree.com/`.

[4] Hugues Hoppe. 1996. Progressive meshes. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques* (SIGGRAPH '96). ACM, New York, NY, USA, 99-108. DOI=`http://dx.doi.org/10.1145/237170.237216`.

[5] Andrea Ciampalini, Paolo Cignoni, Claudio Montani, and Roberto Scopigno. 1996. *Multiresolution Decimation Based on Global Error*. Technical Report. Centre National de la Rech. Scientifique, Paris, France, France.

[6] C. Rebollo, J. Gumbau, O. Ripolles, M. Chover, and I. Remolar. 2007. Fast rendering of leaves. *In Proceedings of the Ninth IASTED International Conference on Computer Graphics and Imaging* (CGIM '07), Enrico Gobbetti (Ed.). ACTA Press, Anaheim, CA, USA, 46-53.

[7] Jakulin A., Interactive vegetation rendering with slicing and blending. In Proc. *Eurographics 2000: Short Presentations*, 2000.

[8] Garcia, Sbert a Szirmay-Kalos. *Tree Rendering with Billboard Clouds* [online]. Third Hungarian Conference on

Computer Graphics and Geometry, Budapest, 2005. Dostupné z: https://pdfs.semanticscholar.org/6459/32f54d17a418b907f42da5e3f5286d112144.pdf.

[9] Sören Pirk, Bedrich Benes, Takashi Ijiri, Yangyan Li, Oliver Deussen, Baoquan Chen, and Radomir Měch. 2016. Modeling plant life in computer graphics. *In ACM SIGGRAPH 2016 Courses* (SIGGRAPH '16). ACM, New York, NY, USA, , Article 18 , 180 pages. DOI: http://dx.doi.org/10.1145/2897826.2927332.

[10] Bedřich Beneš, Michel Abdul Massih, Philip Jarvis, Daniel G. Aliaga, and Carlos A. Vanegas. 2011. Urban ecosystem design. In *Symposium on Interactive 3D Graphics and Games* (I3D '11). ACM, New York, NY, USA, 167-174. DOI=http://dx.doi.org/10.1145/1944745.1944773.

[11] H. Kang, M. Fiser, B. Shi, F. Sheibani, P. Hirst and B. Benes, IMapple — Functional structural model of apple trees, 2016 IEEE International Conference on Functional-Structural Plant Growth Modeling, Simulation, Visualization and Applications (FSPMA), Qingdao, 2016, pp. 90-97. doi: DOI=http://dx.doi.org/10.1109/FSPMA.2016.7818293.

[12] Radomír Měch and Przemyslaw Prusinkiewicz. 1996. Visual models of plants interacting with their environment. In Proceedings of the 23rd annual conference on Computer graphics and interactive techniques (SIGGRAPH '96). ACM, New York, NY, USA, 397-410. DOI=http://dx.doi.org/10.1145/237170.237279.

[13] Daniel Scherzer, Michael Wimmer, *Frame Sequential Interpolation for Discrete Level-of-Detail Rendering* Computer Graphics Forum (Proceedings EGSR 2008), 27(4):1175-1181, June 2008. Dostupné z : https://www.cg.tuwien.ac.at/research/publications/2008/SCHERZER-2008-FSR/SCHERZER-2008-FSR-draft.pdf.

[14] Markus Giegl, Michael Wimmer, *Unpopping: Solving the Image-Space Blend Problem for Smooth Discrete LOD Transitions*, Computer Graphics Forum, 26(1):46-49, March 2007. Dostupné z: https://www.cg.tuwien.ac.at/research/publications/2007/GIEGL-2007-UNP/GIEGL-2007-UNP-Preprint.pdf.

[15] Freitas, Rui. *Ruizana Games, Video Game Reviews* [online]. In: . 2014 . Dostupné z: http://ruizanagames.blogspot.cz/.

[16] Haden, David. G*uide to installing Gothic II on Windows 7 32-bit.* In: Perfect Worlds [online]. 2010. Dostupné z: https://perfectworlds.wordpress.com/2010/12/26/guide-to-installing-gothic-ii-on-windows-7-32-bit/.

[17] Dianne. *Gothic 3 Optimization Steps*. In: Steam [online]. © Valve Corporation, 2013. Dostupné z: `https://steamcommunity.com/sharedfiles/filedetails/?id=155304006`.

[18] *Gameplay Framework Quick Reference* [online]. In: . Epic Games, ©2004-2017. Dostupné z: `https://docs.unrealengine.com/latest/INT/Gameplay/Framework/QuickReference/`

[19] *Game Flow Overview* [online]. In: . Epic Games, ©2004-2017. Dostupné z: `https://docs.unrealengine.com/latest/INT/Gameplay/Framework/GameFlow/index.html`

[20] Kratt, Coconu, Dapper, Schliep, Paar, Deussen. *Adaptive Billboard Clouds for Botanical Tree Models* [online]. University of Konstanz, 2014. Dostupné z: `https://kops.uni-konstanz.de/bitstream/handle/123456789/27882/Kratt_278823.pdf?sequence=2`.

Obrázky

Tabulky

Příloha A

Seznam použitých zkratek

2D	Two-Dimensional
3D	Three-Dimensional
FPS	Frame Per Second
LOD	Level Of Detail
UE4	Unreal Engine 4
PIE	Play In Editor
SIE	Simulate In Editor
GPU	Graphics Processing Unit

Příloha B

Uživatelská příručka

Možnosti jak použít nástroj na ovládání ročního období jsou dvě. Tou první je spustitelný .exe soubor a druhou je práce přímo v editoru Unreal Engine 4. Soubor .exe se spouští bez zadávání vstupních parametrů. Po spuštění aplikaci se objeví charakter na kraji krajiny z pohledu první osoby. Charakter se může volně pohybovat po krajině. Pro pohyb po krajině a ovládání menu se používá klávesnice a myš. Používané klávesy jsou shrnuty v tabulce B.1. Hráč se pohybuje pomocí *W,S,A,D* a rotuje kamerou pomocí myši. Pomocí klávesy *P* se spouští menu. V menu se používá myš a klávesnice pouze pro psaní hodnot do textových boxů. Při spuštěném menu se hráč nemůže pohybovat, ale hra stále běží bez pauznutí, aby byly okamžitě vidět změny nstavení. Ukázka všech položek v menu je na obrázcích 4.3, 4.4 a 4.5.
Klávesami *F* a *G* se manipuluje s vegetací. Pomocí zbraně a zaměřovače lze vybrat místo, kde bude vegetace vložena nebo odstraněna. Klávesou *F* lze vkládat malý strom, vysoký strom, kapradí a rostlinu. Mezi těmito volbami se přepíná Num klávesami. Odstraňovat lze stejné typy vegetace namířením na konkrétní objekt a stisknutím klávesy *G*. Ve scéně jsou rozmístěny emitory částic pro padající listy, déšť a sníh, které se ovládají z menu. Umístění emitoru, který se vždy pohybuje s charakterem se zapíná klávesami *R* a *T* a vypíná klávesou *E* (vypne oba).
Pro práci v editoru je třeba vložit bleprint *B_season*, který se nachází ve složce *Conent/SeasonTool/B_season* do scény. Poté se objeví ve scéně grafu a bude přístupný z panelu detailů (obr. 4.2). Pro správnou funkčnost je třeba přiřadit jednotlivé krajiny, světlo a zdroj větru blueprintu do k tomu vyhrazených políček. Materiály krajiny jsou ve složce *Conent/SeasonTool/Landscape* a při použití je třeba přiřadit jednotlivým vrstvám informace (přetažení na stejně pojmenovanou vrstvu) ze složky *Conent/SeasonTool/Landscape/LayerInfo*. Vložení particle do scény lze přetažením particle ze složky *Conent/SeasonTool/Particle*. Jednotlivé parametry lze ovládat z blueprintu *B_season*. Po spuštění hry

W	pohyb dopředu
S	pohyb dozadu
A	pohyb doleva
D	pohyb doprava
Space	skok
P	otevření menu
F	vkládání vegetace
G	mazání vegetace
E	zastaví particle nad hráčem
R	spouští particle deště nad hráčem
T	spouští particle sněhu nad hráčem
Num1	nastaví vkládání nízkého stromu
Num2	nastaví vkládání vysokého stromu
Num3	nastaví vkládání kapradí
Num4	nastaví vkládání rostliny

Tabulka B.1: Popis použití kláves

v editoru zůstane zachované nastavení z editoru.

Příloha C

Obsah přiloženého flash disku

- `bin` ..... spustitelný .exe soubor pro platformu Windows
- `images` ..... screenshoty z aplikace
- `src` ..... projekt Unreal Engine
- `text` ..... text diplomové práce
  - `pdf` ..... text práce ve formátu pdf
  - `latex` ..... zdrojové soubory LaTeX
- `README`