Karelia-ammattikorkeakoulu
Tradenomi (AMK)
Tietojenkäsittelyn koulutus

3D-visualisointi suunnittelu – ja konsultointialan yrityksessä

Opinnäytetyö

Tuisku Saarelainen

Opinnäytetyö, lokakuu 2024

www.karelia.fi

Karelia AMMATTIKORKEAKOULU

OPINNÄYTETYÖ
Lokakuu 2024
Tietojenkäsittelyn koulutusohjelma

Tikkarinne 9
80200 JOENSUU
+358 13 260 600 (vaihde)

Tekijä
Tuisku Saarelainen

Nimeke
3D-visualisointi suunnittelu- ja konsultointialan yrityksessä

Toimeksiantaja
Ramboll Finland Oy

Tiivistelmä

Tässä portfoliopohjaisessa opinnäytetyössä tarkastellaan visualisoinnin projekteja, joissa olen ollut mukana ja sitä, kuinka olen kehittynyt niiden aikana. Opinnäytetyön toimeksiantaja on työnantajani, Ramboll Finland Oy. Opinnäytetyössä perehdytään ja avataan visualisoinnin projekteissa esiintyviä tekniikka-alueita. Nämä kuuluvat reaaliaikaiseen renderöintiin, jonka osa-alueita ovat niin valaistaminen, geometriat ja jälkiprosessiefektit.

Tietoperustana toimii Unreal Engine -luomismoottorin laaja dokumentaatio ja alan muiden yritysten teknologiakirjoitukset. Opinnäytetyössä käy ilmi reaaliaikaista renderöintiä hyödyntävän Unreal Engine -luomismoottorin päivittyminen. Uusien teknologioiden myötä projektitöiden tekeminen on saanut uudenlaisia menetelmiä, kuten elokuvateollisuudessa käytetty kompositointi.

Analysoivassa pohdinnassa tarkastellaan osaamisen kehittymistä ja kuinka projektien tekniikoita olisi voinut käyttää toisin. Projektien tekniikoita käsittelevissä kappaleissa ja analysoivassa pohdinnassa avataan myös tekniikoiden haasteita ja tarkastellaan vaihtoehtoisia tai päivittyneitä tapoja. Analysoivassa pohdinnassa huomataan, että työskentelytekniikat ja teknologiat ovat muuttuneet.

Kieli
suomi

Sivuja 65

Asiasanat
renderöinti, visualisointi, teknologia, efektit, geometria

THESIS
October 2024
Degree Programme in Business Information Technology

Tikkarinne 9
80200 JOENSUU
FINLAND
+ 358 13 260 600 (switchboard)

Author
Tuisku Saarelainen

Title
3D-visualization in a consulting company

Commissioned by
Ramboll Finland Oy

Abstract

In this portfolio-based thesis, I look at the visualization projects I have worked on and how I have improved during them. The thesis was commissioned by my employer, Ramboll Finland Oy. The thesis explains the technical areas found in visualization projects. These areas are related to real-time rendering, including lighting, geometry, and post-processing effects.

The information for this thesis comes from the large documentation of Unreal Engine and technology writings from other companies in the field. The thesis shows how Unreal Engine, which uses real-time rendering, has improved over time. With new technologies, projects can now use new methods, like compositing, which is used in the movie industry.

In the analysis section, I look at how my skills have developed and how the techniques used in the projects could have been applied differently. The chapters that discuss project techniques and the analysis section also explain the challenges of the techniques and consider alternative or updated methods. The analysis shows that working techniques and technologies have changed.

Language
Finnish

Pages 65

Keywords
rendering, visualization, technology, effects, geometry

Sisältö

1 Johdanto 1
2 Reaaliaikainen renderöinti 2
3 Projektien aloittaminen ja niissä työskentely 4
3.1 Visualisointiprojektin perustaminen 5
3.2 Versionhallintajärjestelmä 5
3.3 Unreal Engine -sovelluksen navigaatiojärjestelmä 6
3.4 Lopputuotteet 6
3.5 Sidosryhmät ja vuorovaikutus 6
4 Kehittyminen ja sen arviointi 10
4.1 Vaatimukset ja tavoitteet 10
4.2 Arvioinnista 12
5 Projektit 14
5.1 Valaistaminen erikoisvaloin 14
5.1.1 Valaistus virtuaalisessa maailmassa 14
5.1.2 Virtuaalitodellisuuden valaistaminen 16
5.1.3 Valaisimien optimointi virtuaalitodellisuudessa hyödyntäen lykättyä varjostinta 18
5.1.4 Uudet tekniikat 19
5.1.5 Projektin työstäminen 20
5.2 Kasvillisuuden päivittäminen uuteen 21
5.2.1 Kasvillisuusmallien optimointi 22
5.2.2 Ajoneuvoliikenne 24
5.2.3 Uudet tekniikat 26
5.3 Puistosuunnitteluun tukea fotogrammetrialla 27
5.3.1 Fotogrammetria 27
5.3.2 Aineiston optimointi 29
5.3.3 Projektin työstäminen 30
5.3.4 Nykyteknologia 31
5.4 Kauppatorin elävöittäminen 31
5.4.1 Hahmot 31
5.4.2 Monitasoinen valaistus 32
5.4.3 Äänet ympäristössä 33
5.5 Elokuvateollisuuden efektit ratahankkeessa 34
5.5.1 Digitaalinen kompositointi 34
5.5.2 Aineiston työstäminen 34
5.6 Valtatien visualisointia 38
5.6.1 Corine maanpeite 38
5.6.2 Projektin työstäminen 39
5.7 Maanpinnan toteuttaminen 46
5.7.1 Maanmittauslaitoksen maastotietoaineistot 46
5.7.2 Ortoilmakuvat paikkatietoaineistoista 48
5.8 Uusien tekniikoiden hyödyntäminen optimoinnissa 49
5.8.1 Nanite -virtualisoidut geometriat 49
5.8.2 Lumen ja Virtual Shadow Maps 52
6 Analysoiva pohdinta 56
6.1 Varjoista oppimiseen 57
6.2 Impostor ja ohjelmointi 57

6.3 Drone-operaattoriksi 58
6.4 Filmiteollisuuden keinot 59
6.5 Metsien generoiminen 60
6.6 Milloin pääsemme eroon optimoinnista? 61
Lähteet 62

1 Johdanto

Lähtiessäni opiskelemaan ohjelmointia Karelia-ammattikorkeakouluun, josta erikoistuisin harjoittelun kautta pelialaan, en osannut arvata työskenteleväni tulevaisuudessa suunnittelu- ja konsultointialan yrityksessä. Erään korkeakoulun työharjoittelun aikana perehdyin Unreal Engine -luomismoottoriin, jota tuolloin vielä kutsuttiin pelimoottoriksi sekä C++-ohjelmointiin. Työharjoittelupaikassani teemana olivat vahvasti pelit ja se toimi pelihautomossa. Tutustuin pelihautomossa erilaisiin pelialan yrityksiin. Työharjoittelun aikana toteutin First Person Shooter -pelin nykyisen kollegani kanssa.

Työni harjoittelupaikassa alkoi enemmän viitata rakennuskohteiden visualisointiin. Toteutin visualisointiprojektin pohjoiskarjalaiselle kaupungille osana kaavamuutosta, joka muuttaisi huomattavasti rautatien aseman aluetta. Tässä portfoliopohjaisessa opinnäytetyössäni kerron projekteistani ja siitä, kuinka reaaliaikaista renderöintiä hyödynnetään suunnittelu- ja konsultointialan yrityksessä. Tämän opinnäytetyön tavoite on tarkastella osaamiseni kehittymistä projektien välillä.

Visualisointia hyödynnetään suunnittelu- ja konsultointialan yrityksissä parantamaan ja nopeuttamaan hankkeiden valmistumista. Kuvien ja videoiden renderöintiin on saatavilla useita erilaisia sovelluksia, joista jokaisella on markkina-arvoonsa liittyviä ominaisuuksia, joita muilla sovelluksilla ei ole. Tämän portfoliopohjaisen opinnäytetyön projektit on toteutettu reaaliaikaisella renderöinnillä käyttäen Unreal Engine -sovellusta. Projektit on toteutettu vuosien 2019 ja 2024 välillä työskennellessäni Ramboll Finland Oy:ssä (myöh. Ramboll). Koska aikaväli on pitkä, sovellus on päivittynyt useaan otteeseen tänä aikana tuoden parannuksia edellisiin projekteihin verrattuna. Tässä työssä perehdytään projekteissa esiintyviin tekniikoihin, syihin niiden käyttöön sekä siihen, miten osaamiseni on kehittynyt näiden vuosien aikana. Luvussa kaksi kerrotaan reaaliaikaisesta renderöinnistä, jonka avulla luodaan virtuaalisia maailmoja visualisoinnin projekteihin. Valittu tekniikka auttaa esittelemään

suunnitelmia virtuaalitodellisuudessa tai näytöltä. Luvun kolme aikana perehdytään projektien perustamiseen, sekä kuinka visualisointiprojektit alkavat. Neljännessä luvussa kartoitetaan osaamistani eri osaamisalueilla ja viides luku avaa teille projekteja, joita olen työstänyt, sekä niissä käytettyjä tekniikoita, joita analysoidaan lopuksi kuudennessa luvussa.

2 Reaaliaikainen renderöinti

Reaaliaikainen renderöinti on kuvien tuottamista nopeasti tietokoneella ja niiden esittämistä (Akenine-Möller 2002). 2000-luvun alkupuolella kolmiulotteisuuteen keskittyneet näytönohjaimet alkoivat hyödyntää laitteistokiihdytystä renderöinnin eri osa-alueilla. Esimerkiksi NVIDIAn RTX -sarjan näytönohjaimien säteenseurantatekniikka on parantanut reaaliaikaisuuden tasoa ja tuonut mukanaan uusia, kuvanlaatua parantavia ominaisuuksia. Allen Eccles väitti vuoden 2000 GameSpy artikkelissaan, että sen ajan näytönohjain, 3Dfx Voodoo 1, paransi reaaliaikaisen renderöinnin tasoa sekä toi uusia kuvia parantavia ominaisuuksia. Reaaliaikaisen renderöinnin saavuttamiseksi on ymmärrettävä, miten pintamallien eri käsittelyvaiheet, kuten sovellus-, geometria- ja rasterointivaiheet, vaikuttavat vasteaikaan. (Akenine-Möller 2002, 11-12.) Vasteaika muodostuu useista tekijöistä, joita käsitellään tässä opinnäytetyössä. Tähän kuuluu prosessorin käyttämä aika liikkeiden, fysiikan ja muun toiminnallisuuden laskemiseen. Vasteaikaan kuuluu myös näytönohjaimen suorittama aika ruudun piirtämiseen. Näytönohjaimen renderöintiprosessiin kuuluu muun muassa pintojen, partikkeliefektien ja volumetristen aineistojen rasterointi (Akenine-Möller 2002).

Reaaliaikaisen renderöinnin vasteajaksi mielletään 33.3333 millisekuntia tai tätä vähemmän. Tämä vastaa 30 ruutua tai enemmän per sekunti ja 33.3333 millisekuntin vasteaika on vakiintunut pelialalla vähimmäisruutupäivitysvasteajaksi (Akenine-Möller 2018). Toisaalta reaaliaikaiseksi renderöinniksi voidaan myös laskea tätä suuremmat vasteajat, jos sovellus suorittaa käyttäjän reagoinnin jo seuraavalle ruudulle, mutta

reaaliaikaisen renderöinnin termi on vakiintunut kolmiulotteisuutta hyödyntäviin sovelluksiin, kuten videopeleihin. Videopelien kehityskaaressa tähdätään vähintään 30 ruudun päivitykseen per sekunti, jotta käyttäjä pystyy keskittymään ja reagoimaan pelin kulkuun havaitsematta vasteaikaa. Näyttöjen virkistystaajuus (Hz) vaikuttaa paljon siihen, miten monta kuvaa per sekunti voidaan esittää näytöllä.

Työskennellessä reaaliaikaista renderöintiä käyttävällä pelimoottorilla, Unreal Engine -sovelluksella, vaaditaan tietoa, kuinka reaaliaikaista renderöintiä käyttävä teknologia toimii. Ei-reaaliaikaisrenderöintitekniikkaa hyödyntävät teknologiat käyttävät enemmän aikaa valojen laskemiseen. Tällaisiin ei-reaaliaikaisrenderöintitekniikoihin kuuluvat esimerkiksi säteensuuntaus (ray casting), säteenseuranta (ray tracing) ja polunjäljitys (path tracing). Nykyisellä näytönohjainteknologialla nähdään, kuinka ei-reaaliaikaisrenderöintitekniikoista on tullut joissain tapauksissa entistä reaaliaikaisempia tietokonepeleissä (NVIDIA 2020). Tämä johtuu siitä, että NVIDIA on julkistanut uusia säteenseurantaa kiihdyttäviä näytönohjaimia. NVIDIAn RTX-sarjan näytönohjaimissa erilliset sirut käsittelevät säteenseurantaa, säteensuuntausta ja polunjäljitystä, nopeuttaen näiden laskentaa huomattavasti (NVIDIA 2018). Kun kolmiulotteiset videopelit alkoivat kehittymään, vaadittiin entistä tehokkaampia näytönohjaimia (Alba 2023).

Visualisoitavissa projekteissa suunnittelu- ja konsultointialan yrityksissä, kuten myös pelialalla, virtuaaliset maailmat voivat olla hyvinkin raskaita prosessorille ja näytönohjaimille. Tällöin pintamalleja sekä materiaaleja tulisi optimoida, esimerkiksi vähentämällä pintamallien verteksejä ja kolmioita. Myös materiaalit tulisi muotoilla kevyiksi. Jälkimmäiseen kuuluu esimerkiksi tekstuurien resoluution pitäminen ensimmäisen (1) ja kolmannentoista (13) kahden potenssissa. Myös materiaalin laskennan vähäisyys auttaa pitämään vasteajan reaaliaikaisena. Jokaisen ruudun pikseli lasketaan piirtämisessä, eli näin ollen suurempi resoluutio vie enemmän laskenta-aikaa (Akenine-Möller 2002, 409).

Kun kykenee käsittämään reaaliaikaisen renderöinnin osa-alueet, kyetään tuottamaan reaaliaikaisen renderöinnin kolmiulotteisuuteen perustuvia

visualisointeja suunnittelu- ja konsultointialan yrityksessä. Yhtenä lopputuotteen muotona toimii Unreal Engine -luomismoottorista uloskirjattu suoritettava sovellus, jota yrityksessä kutsutaan havainnemalliksi.

3 Projektien aloittaminen ja niissä työskentely

Projektit alkavat yrityksessäni tilaajapalaverilla ja sisäisellä palaverilla, joissa käydään läpi projektin eri osa-alueet, kuten mittaukset, vesihuolto, suunnittelu ja visualisointi. Näissä palavereissa keskustellaan siitä, mitä kukin tarvitsee toiselta oman työnsä tueksi. Visualisoinnin osalta esitellään tarpeet sisäisessä palaverissa projektiryhmälle ja jossa saadaan myös tietoa aikataulusta muilta ryhmiltä. Olen tottunut siihen, että visualisoinnin aikataulua voidaan muuttaa. Toisinaan ilmaistaan huolia siitä, että negatiiviset aikataulumuutokset vaikuttavat usein työn laatuun. Työ alkaa usein suunnittelualan projektin loppupuolella, jolloin suunnittelijat ovat jo saaneet valmiiksi suurimman osan projektin suunnitelmista, kuten valaistus-, maisema- ja katusuunnittelun. Ongelmana on ollut se, että nämä suunnitelmat voivat muuttua huomattavasti sen jälkeen, kun visualisoinnin työ on jo alkanut ja kesken. Tällöin joudutaan tekemään muutoksia manuaalisesti tai pahimmassa tapauksessa aloittamaan niiden aineistojen osalta alusta.

Koska DWG-formaatin piirtodata perustuu yrityksessä reaalimaailman koordinaatteihin, siirretään piirtodatan lähelle 0-koordinaattia, jotta saadaan eliminoitua renderöintiin liittyvät virheet, jotka johtuvat perustarkkuuden liukuluvun maksimiarvon ylityksessä sekä Unreal Engine -sovelluksen käyttäessä oletuksena sentimetriperusteista suorakulmaista koordinaattijärjestelmää (Epic Games 2024a). Suurin osa näytönohjaimista ei kykene käsittelemään kuin perustarkkuuden liukulukua, mutta uusimmissa näytönohjaimissa käytetään kahta tai useampaa perustarkkuusliukuluku -laskentaydintä kaksoistarkkuusliukulukuarvojen laskennan tehostamiseksi (NVIDIA 2023).

3.1 Visualisointiprojektin perustaminen

Projektin alkaessa Rambollilla perustetaan Unreal Engine -projekti, joka kopioidaan tietojärjestelmästämme. Unreal Enginen valmiit projektipohjat ovat hyviä, mutta ne sisältävät turhia objekteja. Työtä valuisi hukkaan, jos käsiteltäisiin jokaisen uuden projektin kohdalla kaikki asetukset, kentän oletusobjektit ja jälkiprosessointiasetukset uudestaan. Yrityksen oma Unreal Engine -projektipohja sisältää kaikki tarvittavat hyvän projektin aloittamiseksi; säätä ja ilmaa käsittelevä objekti, pelaajan oletussijainti ja jälkiprosessointiobjektit. Myös renderöintiin liittyvät asetukset on varmistettu jo valmiiksi. Unreal Enginessä projekteille annetaan nimet ja tämä projektipohja on saanut nimekseen RambollTemplate. Olen tehnyt sovelluksen, joka nimeää projektin ja sen sisältämät lähdekoodit uuden projektinimen mukaisiksi.

3.2 Versionhallintajärjestelmä

Jotta projekteja voidaan jakaa toisille kehittäjille sekä ylläpitää niistä varmuuskopioita, käytetään versionhallintaa. Tämän avulla kehittäjät voivat jakaa tekemiään muutoksia netin välityksellä. Versionhallintajärjestelmä kykenee tarjoamaan alimoduuleita (Submodules), jotka ovat toisissa lähteissä sijaitsevia projekteja. Nämä omaavat saman versiohallintajärjestelmän käytön. Visualisoinnin projektipohjaan initialisoituun Git-versionhallintajärjestelmään määritetyt alimoduulit haetaan projektille omalla komentojonotiedostolla. Windowsin tunnistaessa sen omaksi suoritettavaksi sovellukseksi, suorittaa se tiedostoon määritetyt komennot. Koska alimoduulit on valmiiksi määritetty projektin Git-versionhallintajärjestelmään, siirtyvät ne automaattisesti yrityksen versionhallintajärjestelmään komennolla *git push –set-upstream origin master*. Kun muut kehittäjät kloonaavat projektin versionhallinnasta, käyttävät he komentoa *git clone –recurse-submodules GITREPOSITORYPATH*, jonka viimeinen lauseke viittaa versionhallintajärjestelmän projektin osoitteeseen.

3.3 Unreal Engine -sovelluksen navigaatiojärjestelmä

Visualisointiprojektit eivät aina sisällä 3D-pintamallien työstämistä ja niiden esittämistä, vaan näitä voidaan täydentää erilaisilla toiminnoilla, joissa osaamiseni tulee näkyviin ohjelmoijana. Toimintoihin sisältyy usein ajoneuvojen liikkuminen ympäristössä ja jalankulkijoiden väistäminen suojateillä sekä jalankulkijoiden liikkuminen pisteestä A pisteeseen B hyödyntäen Unreal Engine -sovelluksen tarjoamaa navigaatiojärjestelmää. Navigaatiojärjestelmä sallii tekoälypohjaisten agenttien käyttävän järjestelmän polunhakua (Epic Games 2024b).

3.4 Lopputuotteet

Toteutuneista projektiaineistoista tuotetaan video, kuvia tai Unreal Engine -sovelluksesta uloskirjattu havainnemalli. Nämä riippuvat projektitarjoukseen kirjatuista lopputuotteista. Videot toteutetaan Unreal Engine -sovelluksen Level Sequence-ohjelmalla, jolla pystytään määrittämään virtuaalisen kameran liikkumisen kentässä, sen parametrit sekä ulkoisesti vaikuttavat tekijät, kuten liikkuvat ihmishahmot, ajoneuvot tai kentän kellonajan (Epic Games 2024c).

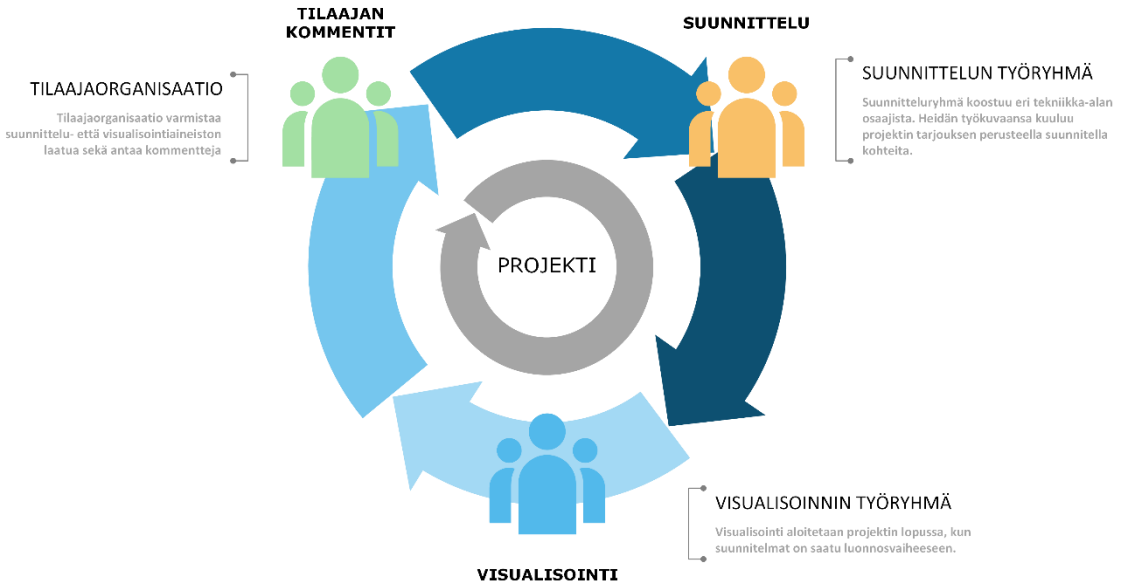
3.5 Sidosryhmät ja vuorovaikutus

Projektien sidosryhmät koostuvat useista eri tekniikka-alan asiantuntijoista. Näihin lukeutuu muun muassa projektijohtajat, suunnittelijat, maisema-arkkitehdit ja vuorovaikutusosaajat. Alussa vuorovaikutus yrityksen projektijohtajiin pidettiin korkealla, mainostaen osaamista reaaliaikaiseen renderöintiin ja erityisesti, pelialaan. Tätä alkuanalyysiä kirjoittaessani, en usko, että aivan kaikki projektijohtajat ovat edes tietoisia reaaliaikaisen renderöinnin osaamisesta. Jos olivat, heillä ei ollut aikaa perehtyä ottamaan visualisointia mukaan projekteihinsa. Meillä on vahva joukko projektijohtajia, jotka ovat ymmärtäneet visualisoinnin vahvuuden ja ottaneet mukaan suurimpaan osaan projektitarjouksia.

Projektia työstäessä on huomattu, että vuorovaikutus tekniikka-alan asiantuntijoihin on hyvä pitää, koska toisinaan ollaan jouduttu työstämään pintamallin useaan otteeseen näiden päivittyessä ulkoisista syistä. Myös tarkentavat kysymykset on lähetetty tekniikka-alan asiantuntijoille. Vuorovaikutuksen ylläpitäminen on tuottanut hedelmää; maisemasuunnittelijat ovat ryhtyneet valmiiksi tuottamaan sopivaa aineistoa. Tätä on esimerkiksi kasvillisuuden tarkentamista lajikohtaisesti. Uskoisin, että osalla maisemasuunnittelijoilla on kiinnostusta jakaa omia näkemyksiään enemmän visuaalisesti, kuin sanallisesti esitettynä suunnitelmadokumentissa.

Yrityksellä on useita asiakkaita, tärkeimpiä ovat kunnat ja kaupungit sekä virastot, kuten ELY-keskukset. Projektien määrä vaihtelee näiden asiakkaiden välillä. Projektia aloittaessa sovitaan yhdessä tilaajaorganisaation kanssa, kuinka useasti pidetään yhteisiä palavereja, joiden aiheena on suunnittelun ja visualisoinnin tilanteet sekä aineiston kommentointi.

Visualisointia ryhdytään esivalmistelemaan lukujen 3.1 ja 3.2 mukaisesti, ennen suunnitteluaineiston luovutusta visualisointiryhmälle. Projektin elinkaaressa (kuvio 1) visualisointia voidaan tarvittaessa suorittaa jo ennen varsinaisen visualisointiprojektin aloittamista, tämä antaa lisäarvoa tilaajalle sekä suunnitteluryhmälle.



Kuvio 1. Projektin yksinkertaistettu elinkaari sidosryhmineen visualisointiprojekteissa (Kuva: Tuisku Saarelainen).

Visualisoinnin työryhmässä suunnitelmia ja saatuja aineistoja ryhdytään käsittelemään visualisoinnin aloituspalaverin jälkeen, jossa katsotaan suunnittelualueen, visualisoinnin tarkkuuden sekä tehtävät lopputuotteet. Visualisoinnin projektin kehityskaaren aikana olemme tottuneet jakamaan tietoja keskenämme. Kun ryhmädynamiikka toimii psykologisella tasolla, myös ryhmän tehokkuus paranee (Connor & Nausha & Slocum 2018, 11).

Visualisointiprojekteissa on havaittu, että joissakin suunnitteluaineistoissa voi olla virheitä, kuten portaan askelma on liian matalalla katutasoon nähden. Näissä tilanteissa viestitään suunnittelijalle ja kerrotaan suunnitteluvirheestä. Joskus ollaan palautteenantajan roolissa suunnitteluryhmälle, koska he eivät välttämättä ole itse havainneet virheitä suunnitelmissa.

Projektin sidosryhmissä esiintyy toisinaan eriäviä mielipiteitä esimerkiksi materiaalin väriskaalasta tai kameran liikeradasta. Molemmilla osapuolilla, niin tilaajaorganisaatiolla kuin suunnitteluryhmillä, saattaa olla omat intressit väripaletin sävyistä, käytettävistä kalusteista ja pintamallien tarkkuudesta. Koska visualisoinnin projektit aloitetaan yleensä kesken suunnitteluprojektin, muuttuvia tekijöitä esiintyy sidosryhmien välisissä palavereissa. Visualisointia hyödynnetään myös palavereissa, jotka auttavat hahmottamaan suunnitelmia

sellaisessa tarkkuudessa, kuin ne ovat suunniteltu sen hetkisessä tilanteessa. Nämä antavat paremman käsityksen käytettävästä tilasta, suunnitelmista sekä sopivuudesta ympäristöön. Tilaaja hyötyy visualisoinnista monessa suhteessa, yhtenä esimerkkinä se edistää suunnitelmien hyväksyttävyyttä sekä visualisoinnin lopputuotteiden hyödyntämistä valmiina markkinointimateriaaleina (Heiniaho 2022, 55-56).

Joillakin sisäisillä, että ulkoisilla sidosryhmillä, saattaa olla hyvinkin korkeita vaatimuksia visualisoinnin laadusta. Ongelmaksi muodostuu usein budjetin riittävyys, johtuen liian tiukasti budjetoidusta osatarjouksista sekä muuttuvista suunnitelmista kesken visualisoinnin projektin sekä visualisoinnin laadusta. Laatua pyritään jo tarjousvaiheessa miettimään, millaisena työn tarjoamme. Jos tilaaja kykenisi tarjoamaan alustavaa budjettia käytettäväksi, helpottaisi se paljon osaprojektin budjetointia sekä tarjouksen sisällön laatimista (Mohammed 2023). Yrityksen asiantuntijatyössä nojataan usein työntekijän kykyyn budjetoida oma työpanos tarjottavaan projektiin.

Työssä vaaditaan uskallusta ja valmiutta kokeilla uusia työskentelymenetelmiä, koska jokaisen projektin kohdalla tilaaja ja sisäinen projektiryhmä on hyvin erilainen. Yksi saattaa vaatia ehdotonta tiukkaa aikataulua, kun toinen ei halua kiirehtiä. Paineensietokyky kuuluu myös työntekijän ominaisuuksiin, koska projektien aikataulut voivat muuttua hyvinkin radikaalisesti. Jokainen projekti sisältää aina omat haasteensa, jolloin täytyy olla valmis reagoimaan niihin ja soveltamaan osaamistaan.

Koska työ on tuottaa havainnollistavaa aineistoa, lopullinen käyttäjä aineistolle ovat yksityishenkilöt, kansalaiset, jotka saavat paremman ymmärryksen suunnitelmista havainnollistavalla aineistolla (Heiniaho 2022, 67-68).

4 Kehittyminen ja sen arviointi

Projekteihin reflektoituva osaaminen vaatii resurssia käyttää aikaa, joka tarkoittaa taloudellista tukea yritykseltä. Kehittymisen arviointi tässä portfoliopohjaisessa opinnäytetyössä rajautuu omiin projekteihin, reflektoiden jokaisen osa-alueen kehittymistä uusissa projekteissa. Uskaltaisin väittää, että jos meillä olisi enemmän resursseja jatkojalostaa tekniikoita ja automatisoida työtä, oltaisiin jo päivittämässä työntekoa enemmän tulevaisuuden ratkaisuihin, kuten tekoälypohjaisiin (AI) tekniikoihin.

4.1 Vaatimukset ja tavoitteet

Yrityksessä työlleni on asetettu seuraavia vaatimuksia, jotka ovat hyvin monialaisia ja laajentavat työkenttääni huomattavasti tavalliseksi mielletyn ohjelmoijan työtehtävistä. Minut luonnehditaan yrityksessä asiantuntijaksi tai konsultiksi.

Projektien laadunvarmistus

Laatua ei varsinaisesti mitata tietyillä dokumentaatioilla, vaan tätä tehdään vertaisvarmistuksella. Havainnollistamisaineistomme on yhtä laadukasta, kuin suunnitteluaineisto on. Jokainen ryhmän jäsen on velvollinen kommentoimaan epäkohtia suunnitteluryhmälle ja kehitysideoita omalle ryhmälleen. Ollaan kehitytty paljon siitä, mitä se oli aloittaessa. Koen olevani tässä aiheessa kokenut asiantuntija, koska olen antanut ja saanut kritiikkiä ja auttanut kollegoitani.

Visualisointiprojektien budjetin seuranta

Budjettia seurataan laskentataulukolla. Budjetti jaetaan visualisointiryhmän jäsenten kesken. Vastuulleni kuuluu raportoida esihenkilölleni, mikäli budjetti ei vaikuta riittävän. Tätä seurataan koko projektin elinkaaren aikana. Budjetin

seurantaa on kehitetty vuosien varrella. Budjetin seuranta oli alussa haasteellista johtuen kommunikaatiokatkoksista sekä käytänteiden puutteellisesta tiedottamisesta. Näiden kehityksen puutteiden ansiosta koen kuitenkin kehittyneeni taitavaksi suoriutujaksi, joka selviää haastavistakin ongelmista. Laskentataulukkomme on verrattain alkeellinen ja sisältää useita kehitysmahdollisuuksia.

Visualisoinnin kehittäminen

Olen osa Rambollin visualisointikehitysryhmää, jossa ideoidaan ja keskustellaan uusista teknologioista sekä käytänteistä, sekä näiden käytäntöön ottamisesta. Visualisoinnin kehittäminen kuuluu myös omaan visualisointiryhmääni, joka on erikoistunut reaaliaikaiseen rasterointiteknologiaan. Olen ottanut osaksi kehittymistäni alalle nelikopterin lennättämisen visualisoinnin työn tueksi. Nelikopterilla kuvattuihin tai videoituihin aineistoihin kyetään upottamaan renderöityä aineistoa, tai kyetään tuottamaan niiden avulla fotogrammetrisia aineistoja visualisoinnin käyttöön. Olen vielä aloitteleva toimija nelikopterin lennättämisessä, mutta vaivannäkö kannattaa sen tuodessa uusia tekniikoita pintamallien toteuttamiseen. Olen kokenut asiantuntija, koska kehitän ja opastan muiden sekä omaa toimintaa reaaliaikaiseen renderöintiin. Ammatillisessa mielessä, kun ymmärtää reaaliaikaisen renderöinnin ja optimoinnin perusteet sekä tuntee käytettävien ohjelmistojen dokumentaatiot ja rajapinnat, voi niitä hyödyntää esimerkiksi peli- tai elokuva-alalla, vaikka ei olisi pelitoiminallisuuksia tai elokuvia toteuttanutkaan.

Ohjelmointityö

Projektit ja työnteon kehittäminen sisältää usein ohjelmointia. Osaamiseni ohjelmointiin on kehittynyt paljon työkokemuksen karttuessa, mutta koska projektityöskentelyni on paljon muutakin kuin koodin kirjoittamista, on se jäänyt jonkin verran taka-alalle. Näkisin, että jos tekisin työtä täyspäiväisenä ohjelmoijana, ero olisi kuin pimeydellä ja valolla. Toisinaan voisin tehdä enemmänkin ohjelmointia, ja tästä on käyty keskustelua toisen yksikön kanssa,

joka on erikoistunut toteuttamaan työtä helpottavia, automatisoituja ratkaisuja eri sovelluksille. Koen olevani taitava suoriutuja, vaikka en ole päässyt tekemään ohjelmointitöitä kunnolla. Tämä johtuu siitä, että minulla ei ole ollut aikaa ohjelmoimiseen. Ohjelmointia on tarkoitus lisätä osaksi työtehtäviäni, jotta voisin kehittää työkaluja käyttöömme.

Tiimin johtaminen

Johtaminen vaatii päivittäistä oppimista. Johdan työpaikallani kolmen hengen tiimiä, johon itsekin lukeudun. Muut visualisointiin perehtyneet kollegat ovat aloittaneet käyttämään Unreal Engine -sovellusta. He ovat pyytäneet minulta apua teknisissä asioissa niin Unreal Engine -luomismoottoriin kuin työskentelytekniikoihin liittyen, kuten johtamani kollegat. Työni yleisesti ottaen vaatii tietynlaista osaamista johtaa ihmisiä, inspiroiden ja antaen yksilölle mahdollisuus oppia itse sekä yksilöllisesti kohdaten. Nämä ovat osa syväjohtamisen neljästä kulmakivestä (Nissinen 2000, 44). Harjoittelen vielä johtamista, mutta uskoisin, että olen saavuttanut jo paljon ja tulen saavuttamaan enemmän.

4.2 Arvioinnista

Visualisointiprojektit ovat varsin eläviä ja kaikissa on jotakin uutta. Sekä kehittymiseni että työtekniikoiden kannalta, tulisi panostaa enemmän aineiston automaattiseen hakemiseen ja sen käsittelyyn. Esimerkkinä maanpintamallin hakeminen verkosta palveluntarjoajan rajapintaa hyödyntäen sekä sen automaattinen käsitteleminen suoraan Unreal Engine -sovelluksessa. Myös puuaineiston hakeminen suoraan palveluntarjoajalta nopeuttaisi työtä. Asioiden automatisoinnin kohdalla, antaisin itselleni enemmän kehittymisen varaa, mutta yrityksen tarjoamilla resursseilla ja muiden kehitystöiden priorisoinneilla on vaikutusta asiaan. Tätä asiaa on myös pyritty delegoimaan yrityksen automatisointia tekevälle yksikölle.

Selviydyn tällä hetkellä minulle annetuista työtehtävistäni mainiosti, numerollisesti ilmaistuna asteikolla nollasta viiteen (0-5), arvioisin neljä (4) omiin osaamisvaatimuksiin. Jätän mielelläni yhden pisteen antamatta, koska aina voi kehittyä lisää. Koska niin tekniikat, teknologia kuin johtaminenkin kehittyvät, riittää työn sarkaa pysyä perillä kaikista uusista metodeista.

Näin voisi väittää, että olen jo kokenut asiantuntija, mutta voisin olla enemmänkin, kuten jo mainitsin, aina voi kehittyä lisää. Visualisointiin liittyviä osaamisvaatimuksia minulle ja ryhmälleni ei varsinaisesi ole esitetty. Minut palkattiin yritykseen täyttämään puuttuvaa asiantuntijuutta, jota kilpailijoilla ei vielä ollut, eli pelialan tuntemusta. Määrittelen vaatimuksia, joita kehitetään ja parannetaan sitä mukaan, kun projektien lomassa on aikaa. Jaamme viikoittain uusia ideoita ja onnistuneita kokeiluja, joita ryhdytään integroimaan osaksi tulevia visualisoinnin projekteja. Koska reaaliaikainen renderöinti ja visualisointi ovat murroksessa tekniikoiden, on hyvä pysyä ajan hermoilla oman osaamisen karttuessa. Tätä portfoliopohjaista opinnäytetyötä kirjoittaessa, tekoälyavusteiset sovellukset ovat kehittyneet paljon ja tarjoavat erilaisia työkaluja sekä ovat työnteon tukena.

Tulevaisuutta ajatellen, koska tekoälyyn pohjautuvat ratkaisut auttavat niin työn tekijää jo monessa asiassa, verbaalisesti että visuaalisesti, olen ottanut yhdeksi henkilökohtaiseksi kehittymisen aiheeksi tekoälyn integroimisen osaksi työntekoa sekä uusien tekoälyyn pohjautuvien ratkaisujen integroimisen osaksi visualisointia. Koen olevani aihepiirissä vielä aloitteleva toimija, koska en ole saanut vielä mahdollisuutta tutustua ohjelmoijana tekoälyn logiikkaan ja sen kirjoittamiseen. Tämä vaatii jatkuvaa perehtymistä jo olemassa oleviin ratkaisuihin, että uusien etsimiseen. Näkisin, että tulevaisuudessa tekoäly auttaa luomaan entistä fotorealistisempia havainnollistavia materiaaleja sekä vähentää työn määrää projekteihin.

5 Projektit

Tässä luvussa avataan projekteja, joita olen tehnyt sekä käsitellään niistä valitut tekniikka-alueet.

5.1 Valaistaminen erikoisvaloin

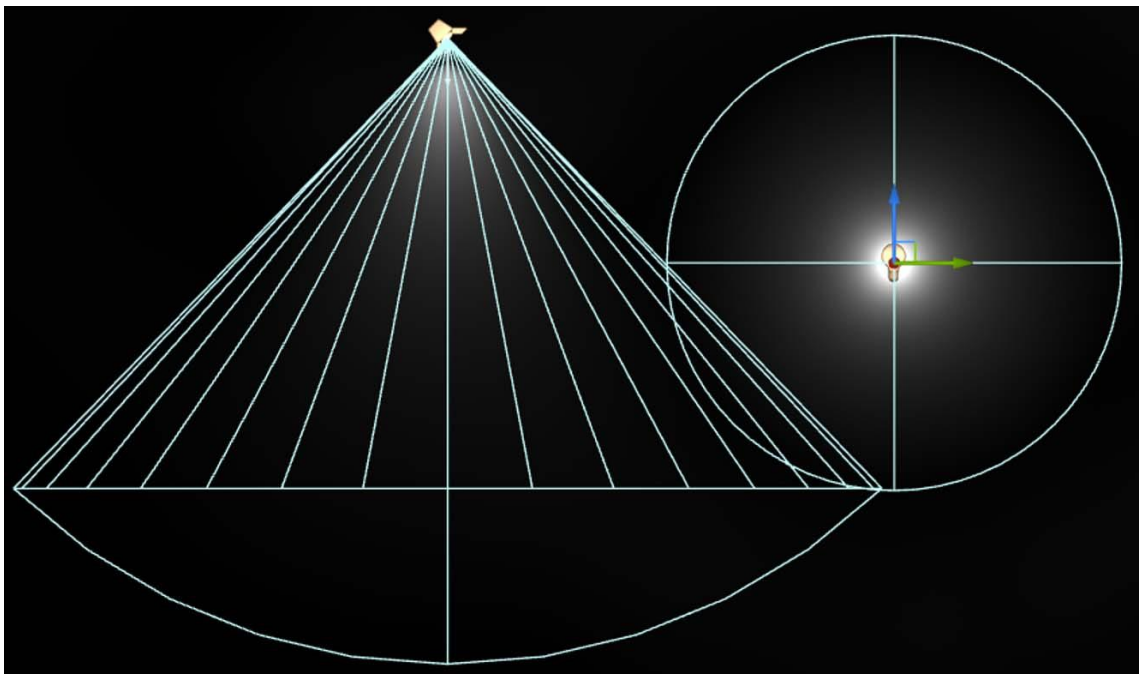
Pohjoiskarjalainen kaupunki tilasi yritykseltä konsultointia uuden sillan rakentamisesta vanhan, 1920-luvulla rakennetun sillan tilalle. Projektissa toteutettiin kolme eri siltavaihtoehtoa, joista jokainen visualisoitiin siltasuunnitelmien mukaisesti luotuun virtuaaliseen kaupunkikuvaan. Projekti oli erityinen siinä mielessä, että se toteutettiin toimimaan virtuaalitodellisuuslaseilla. Tämä toi omat haasteensa, erityisesti siltojen erikoisvalaistuksen osalta. Koska sillat valaistaan useilla valaisimilla, valaistuksen optimointi oli keskeinen osa projektia. Virtuaalitodellisuuden käyttö vaati huolellista valojen optimointia, jotta valaistus näytti realistiselta ja samalla mahdollisti sujuvan suorituskyvyn. Työ toteutettiin vuonna 2019 Unreal Engine -pelimoottorin versiolla 4.20.

5.1.1 Valaistus virtuaalisessa maailmassa

Jotta virtuaalisen maailman valaistaminen reaaliaikaisessa renderöinnissä ymmärrettäisiin paremmin, on tärkeää perehtyä siihen, kuinka tekniikka toimii ja millainen vaikutus sillä on suoritettavan ohjelman suorituskykyyn. Tekniikan negatiivinen vaikutus ohjelman suorituskykyyn heijastuu myös lopputuotteiden toteutukseen, sillä hitaampi renderöinti hidastaa lopputuotteiden valmistumista.

Virtuaalisissa ympäristöissä valonlähteet voidaan lajitella viiteen kategoriaan; suora valo, pistemäinen valo, spottivalo, suorakulmainen valo ja emissiivinen valo, sekä näistä periytyvä epäsuora valo. Jokaisella näistä on virtuaalisissa ympäristöissä omat hyödyt sekä haitat. Suoraa valoa voidaan verrata auringon valoon, joka ympäröi kaiken virtuaalisesta maailmasta. Kuten muillakin valon tyypeillä voi olla erilaisia parametrejä, joilla säädellään valon intensiivisyyttä,

väriä tai värilämpötilaa sekä varjoja. Edellä mainitut ovat kuitenkin yksinkertaisimmat parametrit, joilla valoja voi säädellä, usein eri parametrejä voi olla useita kymmeniä. Pistemäinen ja spottivalo omaavat saman laskenta periaatteen, mutta spottivaloa rajoitetaan rajoituskulmalla (kuva 1).



Kuva 1. Spotti ja pistevalot Unreal Engine -luomismoottorissa (Kuva: Tuisku Saarelainen).

Suorakulmaista valoa voi verrata studioiden suorakulmaisiin valaisimiin tai muihin vastaaviin valonlähteisiin, jotka tuottavat valon suorakulmaisena valokeilana. Suorakulmaista valoa voidaan säätää samalla periaatteella kuin spottivaloa, rajoituskulmalla, mutta tässä rajoituskulman aloitusetäisyyttä voidaan säätää, jotta epäsuoran valon vaikutusta saadaan lisää. Tätä rajoituskulman aloitusetäisyyden periaatetta voisi hyödyntää myös spottivalossa, mutta sen tekninen toteutus ei todennäköisesti ollut toteutettavissa 2000-luvun alun reaaliaikaisissa renderöinneissä, optimoidun epäsuoran valon puutteen vuoksi. Reaaliaikaisessa renderöinnissä on ollut tapana yksinkertaistaa prosesseja, jotta pysytään halutussa vasteajassa.

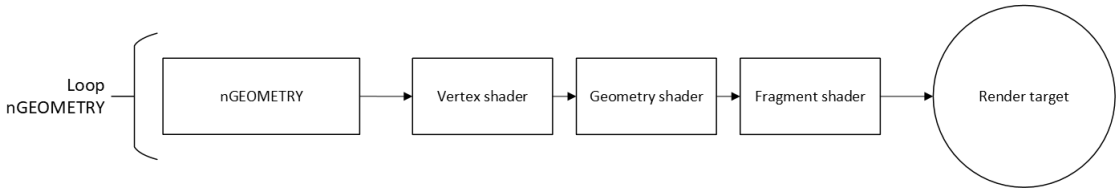
Emissiivisellä valonlähteellä ei ole yksittäistä aloituspistettä tai rajoituskulmaa, vaan sitä määrittelevät pintamallin muodot sekä mahdollisesti tekstuurit. Emissiivisellä valonlähteellä voi olla värejä sekä intensiivisyyttä pystytään

määrittämään. Reaaliaikaisessa renderöinnissä emissiiviselle valonlähteelle on ollut vaikeaa laskea valon vaikutusta muihin pintamalleihin johtuen siitä, ettei sillä ole optimaalista valonlähdettä, josta valo voitaisiin laskea.

Epäsuoralla valolla tarkoitetaan kaikkea sitä valoa, joka on taittunut jostakin pintamallista ja tai materiaalista toiseen. Maailmamme on hyvin synkkä ilman valoa, mutta vain hieman valoisampi ilman sen epäsuoraa valoa. Reaaliaikaisessa renderöinnissä virtuaalinen maailma valaistaan usein suoralla valonlähteellä, joka valaisee kaiken virtuaalisesta maailmasta, mutta sitä voidaan rajoittaa laskemalla varjot, jolloin jäljelle jää ammottava pimeys paikkoihin, mitä valo ei läpäise. Artistisesti kuitenkin varjojen syvyyttä säädetään parametrien avulla, sekä virtuaalista maailmaa voi valaista monella muullakin valonlähteellä, kuten näennäisesti ympäristövalolla. Epäsuora valo valaisee nämä pimeät alueet reaalimaailmassa, mutta tietoteknisesti se on kuviteltua haasteellisempaa (Akenine-Möller 2002, 276-282). Säteen- ja polunseurannassa nämä ongelmat on jo ratkottu (Akenine-Möller 2002, 282-286), mutta niiden tuominen reaaliaikaiseen virtuaaliseen ympäristöön on ollut lähes mahdotonta ennen NVIDIAn julkaisemia RTX-sarjan näytönohjaimia, jotka on tuotettu laskemaan säteenseurantaan kuuluvia valaistuksia, kuten epäsuoraa valoa.

5.1.2 Virtuaalitodellisuuden valaistaminen

Useissa käyttökohteissa, projekteissa, valaistus toimii hyvinkin samalla tavalla valitusta projektiasetuksista riippumatta, käyttörajoituksia huomioimatta. Käytettävä varjostinohjelman vaikuttaa suorasti siihen, kuinka valaistusta käsitellään sekä millaisia ominaisuuksia se käyttää. Virtuaalitodellisuutta käyttävään projektiin lisättäessä valaistusta on useita rajoituksia, kuinka monta valaisinta sekä minkä tyyppistä aiotaan käyttää. Usein virtuaalitodellisuutta käyttävät projektit hyödyntävät Forward shading nimistä varjostinta, jonka voisi suomentaa vapaasti objektikohtaiseksi varjostimeksi, johtuen varjostimen käsittelevän kaikkien objektien valaistuksen yksi kerrallaan (kuvio 2).



Kuvio 2. Objektikohtaisen varjostinohjelman prosessikaavio. (Kuvio: Tuisku Saarelainen)

Deferred rendering, joka suomeksi kääntyisi lykätyksi varjostimeksi, on puolestaan ensisijainen varjostinohjelma Unreal Engine -sovelluksessa. Lykätyssä varjostimessa valaistus hoidetaan viimeisimpinä asioina varjostimessa. Tätä edeltää geometriadatan tallentaminen omaksi parametrikseen, jota voidaan hyödyntää jatkossa varjostinohjelmassa. Koska geometriadata on tallennettu omaksi parametrikseen, voidaan täten vapauttaa enemmän resursseja laskemaan päällekkäisiä valaisimia. Koska lykätty varjostin on Unreal Engine -sovelluksen ensisijainen varjostinohjelma, on siihen ohjelmoitu kaikki valaisimet.

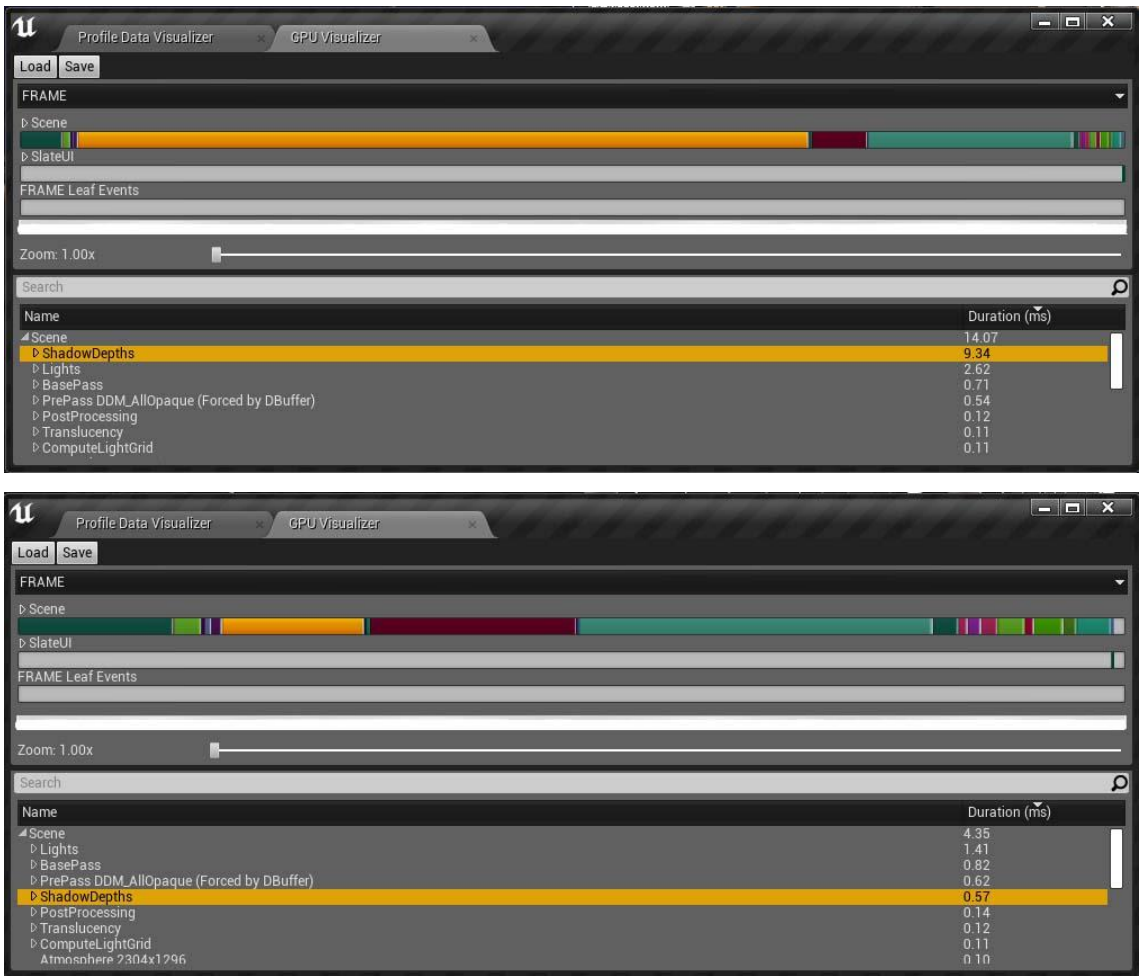
Unreal Engine, käytettävä luomismoottori, rajoittaa tärkeitä parametrejä valaistuksen osalta objektikohtaisessa varjostimessa. Rajoitettaviin parametreihin lukeutuu varjojen hyödyntäminen tietynlaisissa valoissa sekä olosuhteissa. Suoraa valoa on rajoitettu siten, ettei virtuaalisessa maailmassa voi olla kahta tai useampaa suoran valon lähdettä, jotka lisäävät varjoja maailmaan sekä suorakulmaista valoa ei ole voinut hyödyntää laisinkaan objektikohtaisessa varjostimessa (Epic Games 2024d).

Projektissa suorakulmaista valoa hyödynnetään useissa sillan elementeissä, kuten sillan kaiteen alapuolella. Koska objektikohtaisessa varjostimessa ei ole mahdollista hyödyntää suorakulmaista valoa, on meidän valittava lykätty varjostinohjelma projektin virtuaalitodellisuutta varten. Tämä asettaa monia haasteita optimoinnin osalta, koska lykättyä varjostinohjelmaa ei ole optimoitu virtuaalitodellisuutta varten.

5.1.3 Valaisimien optimointi virtuaalitodellisuudessa hyödyntäen lykättyä varjostinta

Virtuaalitodellisuudessa hyödynnetään mieluiten objektikohtaista varjostinta tämän renderöinnin nopeuden vuoksi. Koska geometriat renderöidään kerran näytölle nopeatempoisesti vasteajan puitteissa, on otettava huomioon, että virtuaalitodellisuuslaseissa käytetään kahta näyttöä. Tämä tarkoittaisi sitä, että yhden ruudun renderöinti veisi nyt tuplaten vasteaikaa. Unreal Engine, monien muiden luomismoottorien rinnalla, tarjoaa kuitenkin tätä varten mahdollisuuden instansoida geometriadatan kahdelle näytölle, joka toimii myös objektikohtaisessa varjostimessa. Tämä tarkoittaa, että varjostinohjelma renderöi geometriadatan yhtä-aikaisesti molemmille näytöille (Epic Games 2024e).

Valaisimet voivat nostaa yllättävästi vasteaikaa, ellei niitä pidä optimaalisina. Tähän lukeutuu esimerkiksi, kuinka kauas valaisin renderöi itseään sekä tuottaako se varjoja. Projektin kahteen siltaan on suunniteltu käytettäväksi useita suorakulmaisia valaisimia, jotka sijoitettaisiin kaiteen alapuolelle. Aloitan yhdistämällä valaisinaineistoa isompiin kokonaisuuksiin ja hyödynnän Unreal Engine -sovelluksen tarjoamaa suorakulmaista valaisinta. Kenttäeditorissa säädetään valaisimien sijainnit käsin ja optimoidaan valaisimien yhteistä parametriä, joka säätää valaisimen renderöinnin etäisyyttä. Poistan suorakulmaisista valaisimista varjojen laskennan, joka nopeuttaa renderöintiä huomattavasti (kuva 2).



Kuva 2. Renderöinnin vasteajat ennen optimointia (yllä) ja optimoinnin jälkeen (alla) (Kuvat: Tuisku Saarelainen).

5.1.4 Uudet tekniikat

Tässä osiossa käyn läpi, kuinka valaistus on muuttunut uusimmissa Unreal Engine -sovelluksen versioissa. Kun NVIDIA julkaisi RTX 2000-sarjan näytönohjaimet, jotka mullistivat renderöintiä niin kuvan laadun kuin tuoden säteenseurannan laitteistokiihdyttämistä reaaliaikaiseen renderöintiin. Tätä edelsi Microsoftin ilmoitus uudesta rajapinnasta osaksi DirectX 12 -versiota, joka toi mukanaan säteenseurannan, DirectX Raytracingin (Microsoft 2018). Vain muutamaa päivää myöhemmin, Epic Games julkaisi eräänlaisen tekniikkademon, joka oli tehty yhteistyössä Industrial Light & Magicin ja NVIDIAn kanssa (Epic Games 2018). Tämä kertoneen siitä, että Microsoft on tehnyt yhteistyötä yritysten kanssa jo ennen DirectX Raytracingin julkaisua.

Tästä alkoi eräänlainen tekniikka- ja teknologialoikka, mutta NVIDIAn suora kilpailija, AMD, jäi jälkeen säteenseurannan laitteistokiihdytyksen kehityksestä, julkaistessaan omat säteenseurantaa tukevat näytönohjaimet vasta marraskuussa 2020 (AMD 2020). Tuolloin Epic Games oli kehittämässä uutta Unreal Engine 5 versiotaan. Tämä Unreal Engine -luomismoottorin versio toisi mukanaan DirectX Raytracing-rajapintaan Lumen-epäsuoravalaistuksen, joka toimii sovelluspohjaisesti (Unreal Engine 2020). Tämä tarkoittaa, ettei Lumen tarvitse avukseen ollenkaan laitteistokiihdyttämistä, mutta sitä voidaan nopeuttaa entisestään antamalla laskennan hoidettavaksi sitä tukeville näytönohjaimille. Lumen julkaistiin yhdessä virtualisoidun geometriajärjestelmän, Naniten kanssa (Unreal Engine 2020). Lisäksi monia muita uusia ominaisuuksia julkaistiin, kuten virtualisoidut varjot (Unreal Engine 2020), joka Lumenin kanssa poistaa tarpeen laskemaan ennakkoon niin kutsuttuja valokarttoja. Virtualisoidut varjot ovat monin verroin tarkempia sekä realistisempia verrattuna vanhoihin valokarttoihin tai sarjaankytkettyihin varjokarttoihin (Cascade Shadow Maps). Valokartat mahdollistavat monitahoisen valaistuksen piirtämisen vähäisellä työkuormalla geometrian päälle, mutta eivät kuitenkaan olleet välttämättömyys (Epic Games 2024f). Tämä kuitenkin vaatii valaistuksen laskemisen ennalta.

5.1.5 Projektin työstäminen

Koska käytössä on lykätty varjostin ja se ei ole optimaalinen virtuaalitodellisuutta varten, joudutaan optimoimaan virtuaalista maailmaa ja sen objekteja parantaaksemme ohjelman suorituskykyä. Jotta saadaan nostettua tämän suorituskykyä, eli alennettua ohjelman renderöintiliukuhihnan vasteaikaa, tehdään poikkeuksellisia ratkaisuja. Olisin voinut käyttää ennalta laskettuja valokarttoja sillan valaistuksessa, mutta ongelmaksi muodostui se, että siltoja oli useampi. Unreal Engine -sovelluksessa ei ole mahdollista vaihtaa geometriaa ja ennalta laskettua valokarttaa, eikä yhdistellä valokarttoja. Tällä olisi ollut todella suuri vaikutus sovelluksen suorituskykyyn. Kun Unreal Engine on kehittynyt nykyiseen muotoon, olisin mahdollisesti voinut hyödyntää Lumen-epäsuoravalojärjestelmän emissiivisen valon piirtämistä suorakulmaisen valon sijaan. Lumen on edelleen kehitysasteella virtuaalitodellisuuden osalta, mutta

Theodore McKenzien julkaisu 80 Level -verkkosivulla vuonna 2022 viittaa mahdollisuuteen hyödyntää Lumen-epäsuoravalojärjestelmää virtuaalitodellisuudessa (McKenzie 2022).

Valitsin kehitysaskeleiksi hyödyntää viivästettyä varjostinohjelmaa yllä mainituista syistä. Koska siltamalleja oli useita ja jokaisessa oli uniikit valaistuksensa, yhdistettiin nämä yhdeksi kokonaisuudeksi Blueprint Visual Scripting-järjestelmäpohjaiseen Bridge-luokkaan. Tämän järjestelmän avulla kyetään vaihtamaan geometrioita sekä valaistuksia reaaliajassa. Järjestelmästä kerrotaan lisää kappaleessa 5.2.2.

Kun geometriat ja valaistukset oli yhdistetty Bridge-luokkaan sekä skriptattu sille toiminnallisuus vaihtamista varten, ryhdyttiin optimoimaan tätä valaistuksen osalta. Koska kyse oli projektista, jonka tuli suoriutua virtuaalitodellisuuslaseilla, yhdistelin suorakulmaisia valoja vähentääkseni renderöitävien valojen määrää ja muunsin pistemäisiä valoja spottivaloiksi. Säilytin vain siltarakenteen päällä olevien korkeiden katuvalaisimien pistevalot, koska halusin hyödyntää näissä IES-valonjakoformaatin tiedostoja, joiden avulla esitetään valoja realistisesti. IES-tiedostot toimivat Unreal Engine -luomismoottorissa parhaiten pistevaloissa, koska tämän tyypin valo ei rajoita IES-valonjakoa, toisin kuin spottivalo (Epic Games 2024g). Poistin valoista varjojen renderöinnin päältä pois, joka toi rutkasti suorituskykyä ohjelmaan. Tietynlaisia visuaalisia ilmentymiä jouduin karsimaan, jotta tekniikka toimii projekteissa moitteetta. Kirjoitushetkellä työstäessäni erästä toista projektia, tekniikka ja näytönohjainten tehot ovat kehittyneet valtavasti. Tällä hetkellä pystyn renderöimään useita satoja kappaleita pistevaloja ilman varjojen renderöimistä, kun tässä projektissa ja sen hetkisellä teknologialla vain muutamia kymmeniä.

5.2 Kasvillisuuden päivittäminen uuteen

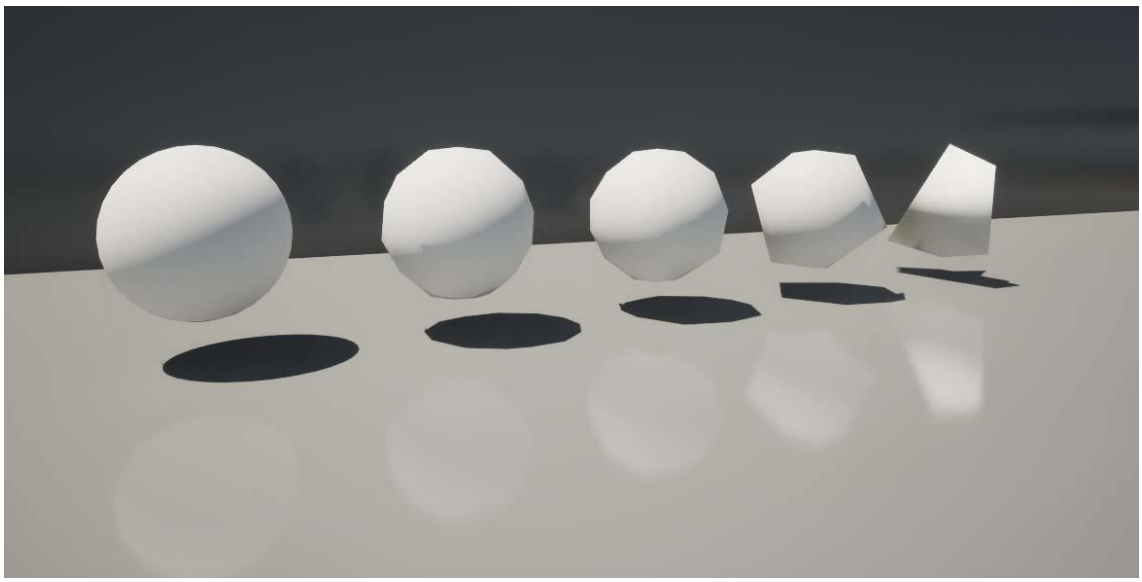
Visualisoinnin projektissa otettiin käyttöön entistä tarkempia kasvillisuusmalleja, jotka paransivat visuaalista ilmettä, mutta toivat esiin uuden huolenaiheen, reaaliaikaisen renderöinnin suorituskyvyn hidastumisen. Uusimaalaisessa

visualisoinnin projektissa otettiin myös käyttöön toteutettu liikenneominaisuus, jolla kyetään luomaan ajoneuvoliikennettä, jolla myös oli osaa suorituskyvyn hidastumiseen. Tässä kappaleessa kerron uusien kasvillisuusmallien optimoinnista sekä ajoneuvoliikenteen tuomisesta projektiin.

5.2.1 Kasvillisuusmallien optimointi

Reaaliaikaisessa renderöinnissä täytyy ottaa huomioon käytettävien pintamallien optimointi. Geometrioiden määrä ja niiden kompleksisuus vaikuttaa paljon renderöinnin vasteaikaan (Polydin 2024). Koska mallit voivat olla hyvinkin raskaita reaaliaikaiseen renderöintiin, tulisi niistä vähentää kolmioita. Jos pintamallia karsitaan ja asetetaan näkyville sellaisenaan, saatetaan vahingossa riisua siitä olennaisen pois näkyviltä. Tätä varten reaaliaikaiseen renderöintiin kehitettiin Level Of Detail-järjestelmä, joka pitää sisällään valmiiksi optimoidut versiot pintamallista (Akenine-Möller 2002, 390). Level Of Detail kääntyisi suomeksi tarkkuustasoksi.

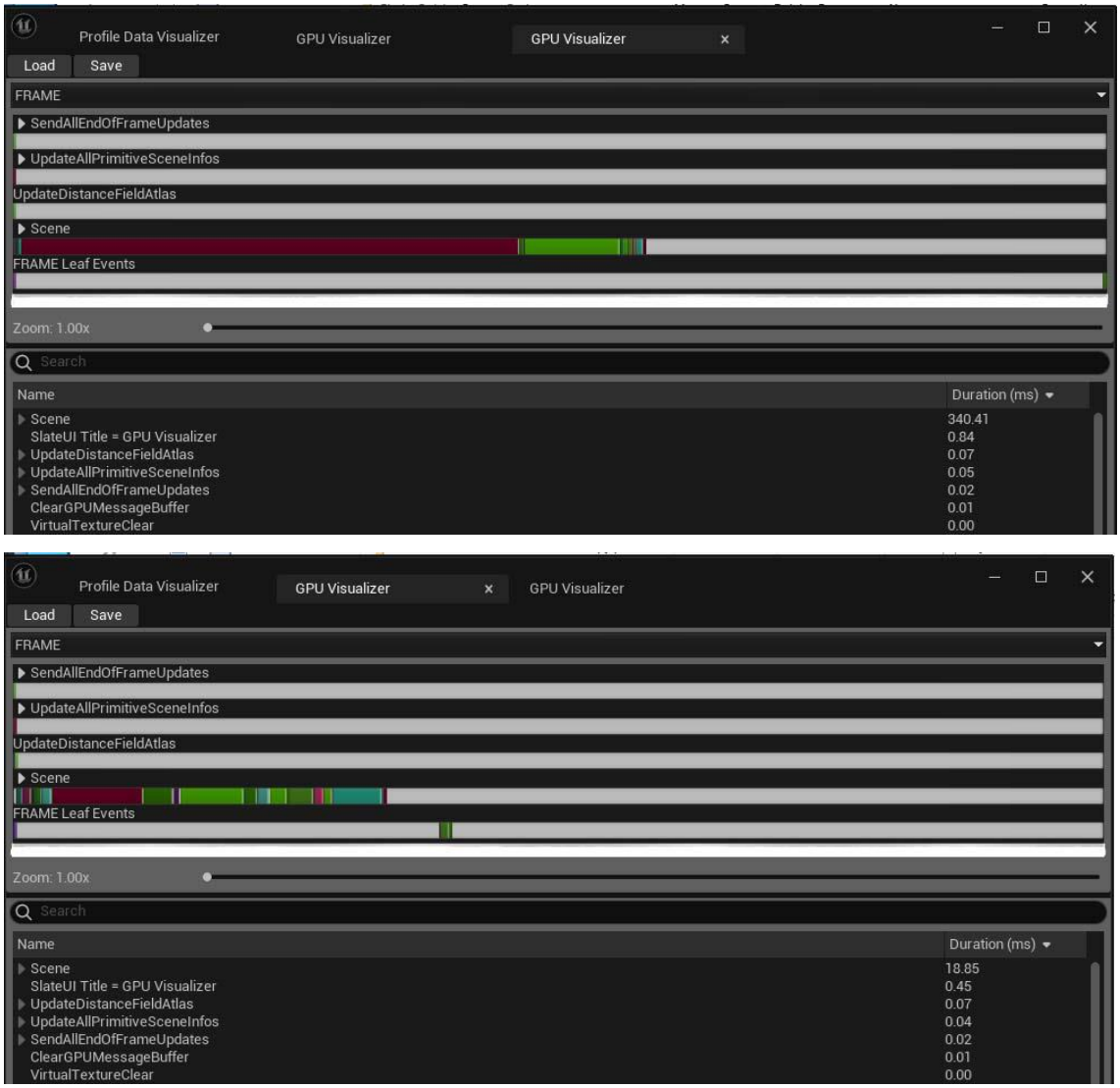
Yksittäisellä pintamallilla voi olla useita tarkkuustasoja, Unreal Engine -sovelluksessa niitä voi olla jopa kahdeksan (kuva 3). Tarkkuustason yksinkertainen määritelmä on olla esittämättä pintaa, jotka eivät pikseleissä näy. Unreal Engine tarjoaa valmiiksi loistavan työkalupaketin tarkkuustasojen määrittämiseksi pintamalleille. Nämä hoituvat usein kahden parametrin vaihtamisella, pintamallin Bounding boxin prosentuaalinen näytönpeitto sekä kolmioinnin prosentuaalisen määrän, joka lasketaan alkuperäisen pintamallin kolmiomäärästä. Bounding box on eräänlainen näkymätön suorakulmainen laatikko, jonka avulla voidaan määritellä, esiintyykö pintamalli renderöinnissä tai kuten tässä esimerkissä, kuinka paljon pintamalli peittää näyttöä. Näytönpeiton laskenta olisi huomattavasti raskaampaa, jos sen referenssiaineistona käytettäisiin raskaampaa pintamallia.



Kuva 3. Tarkkuustasot vierekkäin, vasemmalta tarkemmasta optimoituun oikealle (Kuva: Tuisku Saarelainen).

Impostor on eräänlainen kuvantamistekniikka, jossa pintamalli kuvataan mahdollisimman monesta eri kuvakulmasta ja tallennetaan omaksi kuvatiedostoksi. Tätä kuvatiedostoa kutsutaan nimellä kuvasarja. Kuvasarjasta voidaan nostaa esille pintamallin eri kuvakulmat laskemalla sen UV-lokaatio sekä piirtämällä se yksinkertaiseen pintamalliin, joka vähentää huomattavasti alkuperäisen pintamallin aiheuttamaa suorituskyvyn alentumista.

Aloitin käyttämään tätä tarkkuustasotekniikkaa, kun päivitin kasvillisuuskirjastoamme entistä tarkemmaksi. Unreal Engine tarjoaa impostor-työkalua valmiiksi, joka helpottaa kuvasarjan luomista. Huomataan, että sain nostettua suorituskykyä jopa 94% (kuva 4), kun olin generoinut maanpinnan täyteen kasvillisuuksia, jotka hyödynsivät Impostor-tekniikkaa viimeisenä tarkkuustasona.



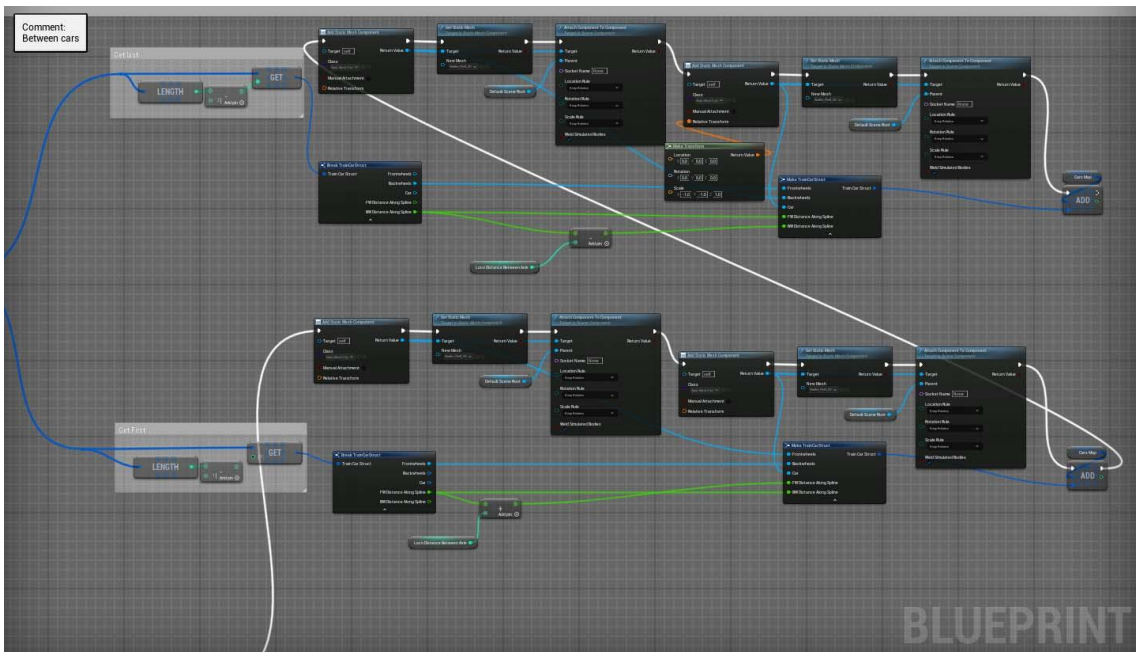
Kuva 4. Sovelluksen suorituskyky ennen impostor-tarkkuustasoa (yllä) ja sen jälkeen (alla) (Kuvat: Tuisku Saarelainen).

5.2.2 Ajoneuvoliikenne

Koska projekti sijoittuu vilkkaaseen Kehä III:seen, on myös todentuntuisempaa nähdä tätä samaa vilkasta liikennettä virtuaalisesta mallista. Otin tässä käyttöön Unreal Engine -sovelluksen tarjoaman Spline-komponentin, jonka avulla määrittellään, missä ajoneuvot liikkuvat.

Jotta pystyin hyödyntämään Spline-komponenttia, on ensin luotava initialisoitava ja suoritettava objekti, jota Unreal Engine -sovelluksessa kutsutaan Actoriksi. Annoin tälle objektille nimeksi ”VehicleSplinePath”. Tämän voi luoda käyttäen C++-ohjelmointikieltä tai Unreal Engine -sovelluksen omaa,

Blueprint Visual Scripting -järjestelmää, jolla pystytään tuottamaan graafisessa toimintaympäristössä eräänlaista visuaalista koodia (kuva 5). Olin valinnut tämän työtavaksi Blueprint-järjestelmän sen helpon ja nopean toiminnallisuuden vuoksi. Tätä tarvitaan myös toiselle Actorille, joka hoitaa ajoneuvon liikkumisen ja pitää sisällään komponentteja pintamallien esittämiseksi. Tämä objekti kantaa nimenään ”Vehicle”.

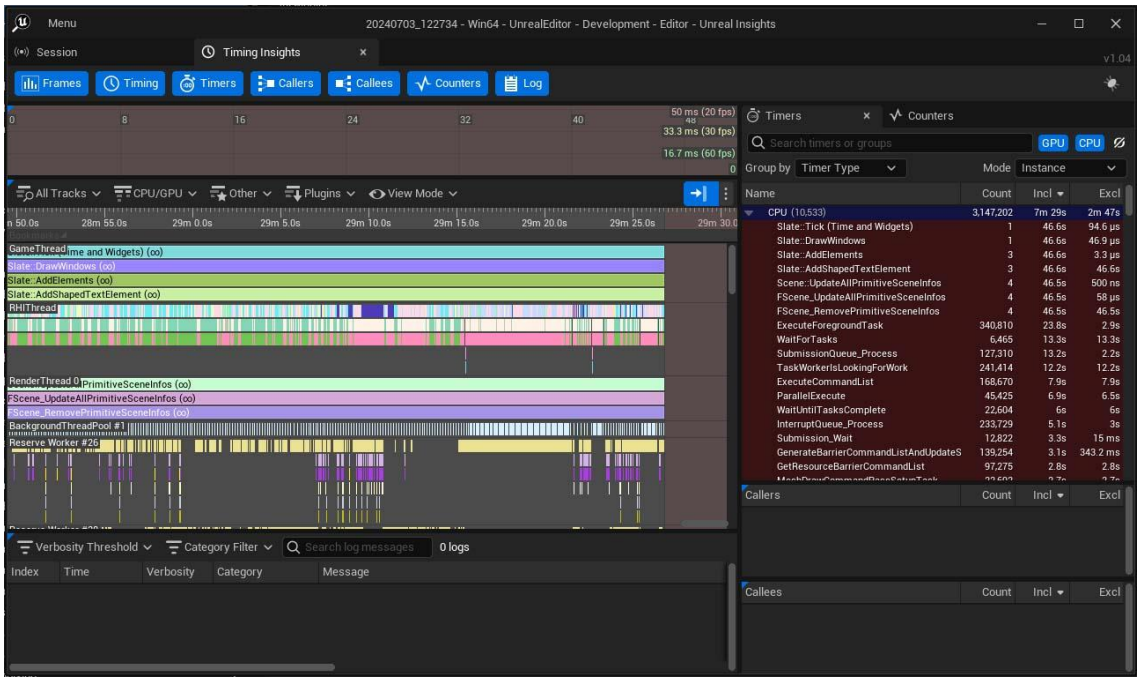


Kuva 5. Blueprint Visual Scripting (Kuva: Tuisku Saarelainen)

Kun suoritetaan virtuaalista maailmaa Unreal Engine -sovelluksessa, järjestelmä luo parametrien avulla halutun määrän Vehicle-objekteja kyseisillä VehiclePathSpline-objekteilla. Vehicle-objektit on määritetty siten, että ne liikkuvat VehicleSplinePath-objektin Spline-komponenttiin määritettyä polkua pitkin. Toteutus ei välttämättä ole optimoiduimmasta päästä, kun olen valinnut Blueprint-järjestelmän käytettäväksi. Tässä graafisesti määritetyt skriptit ajetaan virtuaaliympäristössä, jossa ne ajetaan toimivaksi ohjelmakoodiksi ja lopuksi käännetään koneelle luettavaan ja suoritettavaan muotoon (Epic Games 2014). C++-ohjelmointikielellä kirjoitetusta koodista käännetään suoraan koneelle luettavaksi ja suoritettavaan muotoon, joka on huomattavasti nopeampaa.

Nähdäkseen Vehicle-objektien vaikutuksen suorituskykyyn, käytetään Unreal Engine -sovelluksen tarjoamaa Insights-työkalua (kuva 6). Työkalulle pystytään

seuraamaan jokaisen renderöidyn ruudun aikana suoritetut ohjelmistokomennot sekä kuinka kauan ne ovat käyttäneet aikaa.



Kuva 6. Unreal Insights (Kuva: Tuisku Saarelainen).

5.2.3 Uudet tekniikat

Kun Epic Games julkisti luomismoottoristaan uusimman version, Unreal Engine 5:n, julkaisi se samalla myös reaaliaikaista renderöintiä helpottavan ja modernisoivan tekniikan, jota kutsutaan Naniteksi. Nanite on virtualisoitua geometriadataa, jossa pintamalli klusteroidaan useaksi alueiksi. Klusteroidut alueet toimivat omina tarkkuustasoina, mutta näitä yhdistää pintamallin muut klusterit, jotta vältytään halkeamilta. Klustereista vähennetään kolmiointia siten, että kolmiot ovat suurempia. Nanite pyrkii pitämään kolmiot vähintään yhtä suurina kuin se on esitetty pikselinä (Epic Games 2021). Tämä vähentää jatkossa erilaisten pintamallien optimointitekniikoiden käytön, mutta ei toistaiseksi korvaa vanhoja tekniikoita (Epic Games 2021). Vaikka Nanite mahdollistaa massiivisten pintamallien käytön, on siinä edelleen samoja ongelmia kuin tavallisessa kolmiulotteisessa renderöinnissä. Esimerkkinä omissa havainnoissani tutkiessani Nanitea, järjestelmää ei ole optimoitu suuriin määriin pintamalleihin, joissa on monia materiaaleja. Nanite automaattisesti instansoi kaikki samat pintamallit, jotka käyttävät samaa materiaalia, kun taas

tavallisessa kolmiulotteisessa renderöinnissä nämä pintamallit tulisi instansoida manuaalisesti omaksi tietueeksi, tai kuten Unreal Engine -sovelluksessa, komponentiksi.

5.3 Puistosuunnitteluun tukea fotogrammetrialla

Projektissa hyödynnettiin fotogrammetriaa luomaan todentuntuinen ympäristö puistosta sekä sen välittömästä läheisyydestä. Visualisoinnin työn aikana arvioitiin muistomerkin vaikutusta ympäristöön sekä ympäristön vaikutusta muistomerkkiin.

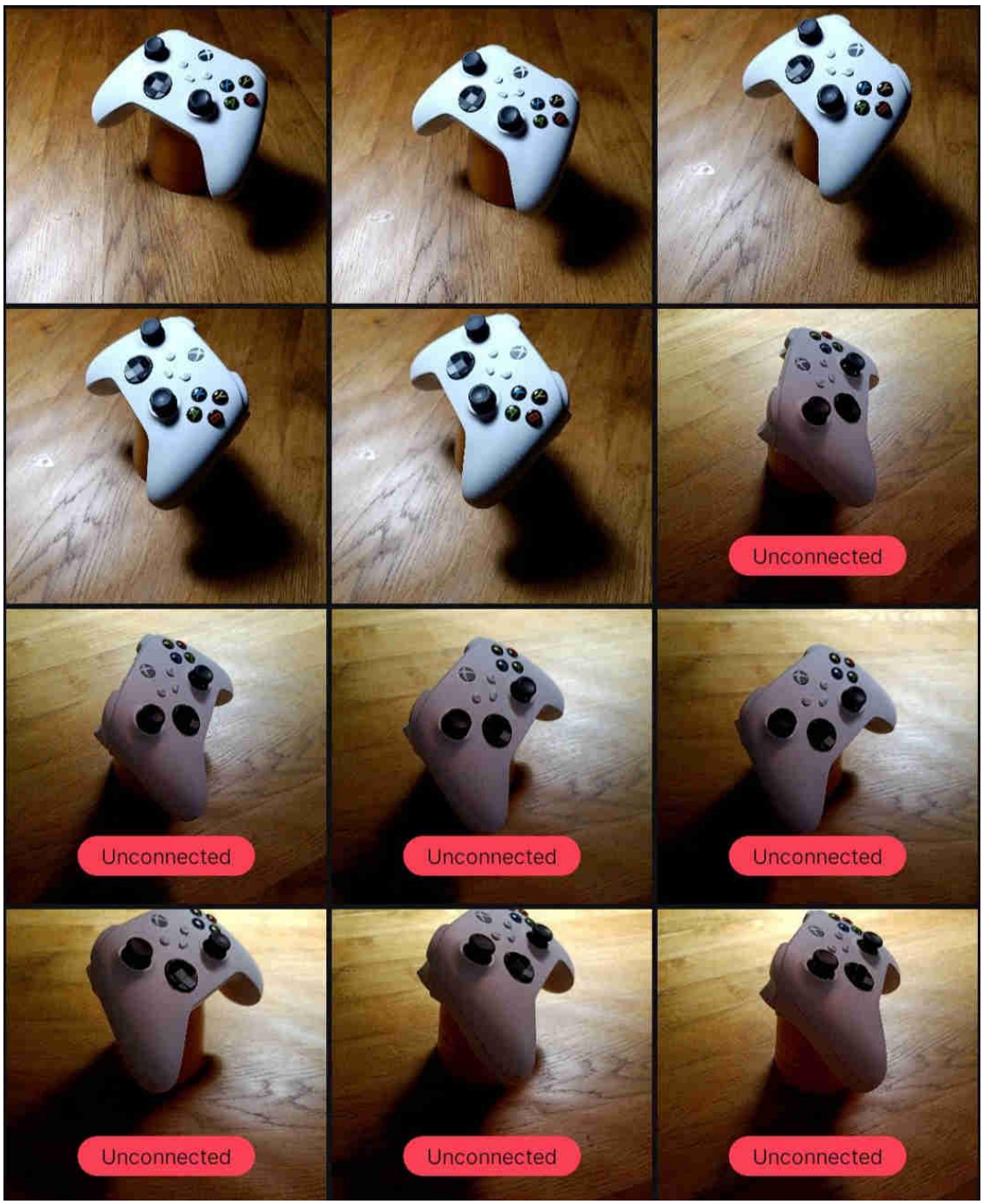
5.3.1 Fotogrammetria

Fotogrammetrialla tarkoitetaan kohteen muuttamista virtuaaliseksi pintamalliksi laskemalla matemaattisesti usean kuvan välinen eroavaisuus. Pintamallin tarkkuus riippuu käytettävien kuvien määrästä ja laadusta (Rönnholm 2023). Pintamallien laskeminen fotogrammetrialla vaaditaan vähintään 50-70% kuvapeitto vierekkäisiin kuviin nähden, jotta pintamallin laskenta olisi sujuvaa ja mallista tulee tarkempaa. Aineisto voidaan tuottaa niin kuvista kuin videoaineistosta. Käytämme projekteissamme Reality Capture-sovellusta fotogrammetria-aineiston laskemisessa. Koska teknologia on kehittynyt vuosien 2019 ja 2024 välillä, uusia ja erilaisia tekniikoita on kehittynyt, jotka parantavat visualisoinnin laatua. Fotogrammetria on yksi näistä kehittyneistä teknologioista, jossa kohde kuvataan kauttaaltaan maasta ja ilmasta kuvasarjoiksi ja lasketaan kolmiulotteinen pintamalli kuvien välisestä muutoksesta (IIETA 2019).

Kuvattaessa kohteita fotogrammetriaa varten tulee säätää kuvaavan kameran asetukset sopivaksi, jotta vältytään epätasaisesta valkotasapainosta ja suljinajoista sekä yli- että alivalaistuksesta (kuva 7). Mikäli kameran asetukset on määritetty väärin ennen kuvauksen aloittamista, joudutaan kuvaaminen aloittamaan alusta. Fotogrammetria-aineiston koheesio niin kuvapeitolta kuin väriavaruuden puolesta tulisi täsmätä kaikissa olosuhteissa. Tästä syystä paras sää kuvaukseen olisi pilvinen, jolloin ympäristö on täyttynyt auringonvalon hajaantuneesta epäsuorasta valosta. Aurinkoisella säällä kuvaus tulisi suorittaa

mahdollisimman nopeasti, koska kohteiden siirtyneet varjot tai muuten muuttunut ympäristön valaistus voivat lopulta haitata fotogrammetria-aineiston laskemista (kuva 8).

Kuva 7. Esimerkki suljinajan vaikutuksesta, vasemmalla suljinaika 1/320 s ja oikealla 1/4 s (Kuva: Tuisku Saarelainen).



Kuva 8. RealityScan-sovellus ilmoittaa kuvissa olevan eroavaisuuksia, eikä niitä voida hyödyntää (Kuva: Tuisku Saarelainen).

5.3.2 Aineiston optimointi

Lasketut pintamallit voivat olla hyvinkin raskaita niin geometriatiedoiltaan kuin tekstuureiltaan. Fotogrammetrialla tuotettujen pintamallien geometriaa siistitään, jotta renderöinnissä ne näyttäisivät paremmalta. Tuotetut pintamallit tulisi optimoida käytettävän renderöintitekniikan puitteissa, kuten tähän projektiin

valittiin objektikohtainen varjostin, jotta työryhmä ja tilaaja voisivat tutkia aluetta virtuaalitodellisuuslasien avulla. Projektin pintamalliaineistoa optimoitiin paljon, koska renderöinnin oli aluksi tarkoitus tapahtua itsenäisesti Meta Quest 2-virtuaalitodellisuuslaseissa, ilman lisäjohtoa (Quest Link Cable) tehokkaasta tietokoneesta virtuaalitodellisuuslaseihin.

Pintamallien optimoinnissa tuli ottaa kantaa moneen asiaan

- Kuinka paljon uskalletaan karsia geometriatiedosta
- Miten suuria tekstuureita Meta Quest 2-lasit kykenevät suorittamaan sekä
- Kuinka moneen pintamalliin joudutaan paloittelemaan fotogrammetrialla tuotetun ympäristögeometrian, jotta tekstuurien tarkkuus pysyy siedettävän ja tarkan välillä.

Koska tekstuurien resoluutiot ovat yleisesti kahden potensseja, suorituskyky laskee samassa suhteessa kuin tekstuurien resoluution koko kasvaa (Meta 2024). Suuremmilla virtuaalitekstuureilla saadaan aikaan näyttävämpiä aineistoja pienemmällä suorituskyvyn menetyksellä.

5.3.3 Projektin työstäminen

Aluksi aivan ongelmitta projektin vieminen Meta Quest 2 -laseihin ei onnistunut, enimmäkseen liittyen yritykseni IT-järjestelmiin ja Android Debug Bridge-työkalun (ADB) käyttöön. IT-järjestelmämme esti ADB:n pääsyn USB-laitteisiin. Tämä ongelma saatiin lopuksi ratkottua, jolloin pystyin jatkamaan tämän projektin osalta.

Unreal Engine -projektin virtuaalisessa maailmassa täytyi ottaa huomioon varjojen käyttäminen. Koska mallissa oli mukana muistomerkki ja sen erikoisvalaistuksen suunnitelmat, meidän täytyi käyttää valaistukselle esilaskettuja valoskenaarioita, jotta valaistuksen vaikutus renderöinnin vasteaikaan pysyisi rajallisena. Hyödynsin tässä erään Unreal Engine -käyttäjän luomaa, epävirallista valaistusmodifikaatiota Unreal Enginestä, joka kulkee Luoshuang’s GPULightmass -nimellä (Luoshuang 2018).

5.3.4 Nykyteknologia

Kuten kaikkien teknologioiden osalta, uusia tekniikoita kehitetään jatkuvasti. Uusimpia tulokkaita fotogrammetrian rinnalle tulleista luultavasti on Neural radiance fields, josta käytetään lyhennettä NeRF (Mildenhall, B. 2020). NeRF on pohjimmiltaan neuroverkkoa hyödyntävä tekoäly, joka interpoloi useasta kuvasta kolmiulotteisen ympäristön (NVIDIA 2024).

5.4 Kauppatorin elävöittäminen

Satakuntalainen kaupunki tilasi yritykseltä virtuaalitodellisuutta hyödyntävän visuaalisen mallin. Kaupungin kauppatori kokee ison uudistuksen, joten sitä esiteltiin kaupunkilaisille sekä kauppatoria hyödyntäville kauppiaille. Koska projektin täytyi toimia virtuaalitodellisuudessa, otin huomioon mm. valaistuksen, liikkuvat ihmishahmot, äänet ja kellonajan vaihtamisen päivästä iltaan sekä kuinka aika vaikuttaa ympäristöön.

5.4.1 Hahmot

Liikkuvat ihmishahmot tuovat paljon eloa virtuaaliseen maailmaan. Unreal Engine tarjoaa yksinkertaisesti toimivalle hahmo-objektille mahdollisuuden liikkua pisteestä A pisteeseen B. Jotta voidaan määritellä, missä hahmo-objekti voi liikkua, määritin sille navigaatioalueen. Navigaatioaluetta voidaan rajoittaa ja poistaa sen alueita käytöstä (Epic Games 2024h). Hahmo-objekteille ohjelmoidaan yksinkertainen toiminallisuus, joka kertoo, mihin se kulkee ja mitä se seuraavaksi tekee suorittaessaan edellisen tehtävän (kuva 9). Hahmo-objektit on toteutettu Blueprint-järjestelmällä. Navigaatiossa liikkuvia objekteja kutsutaan agenteiksi.

Kuva 9. Hahmo liikkumassa navigaatiojärjestelmää käyttäen (Kuva: Tuisku Saarelainen).

Navigaatiojärjestelmä luo monisäikeisen geometriadatan virtuaalisessa maailmassa esiintyvien pintamallien törmäysgeometriadatasta. Navigaation geometriadata jaetaan ruudukoihin projektin asetuksiin määritetyn parametrin arvon mukaisesti. Suurempi määrä näitä geometriaruudukoita aiheuttaa ongelmia suorituskyvyn kanssa, mutta parantaa geometriadatan tarkkuutta sekä agenttien navigointia virtuaalisessa maailmassa. Jatketaan näissä olosuhteissa Unreal Engine -luomismoottorin vakioasetuksilla, koska kauppatoria elävöitetään rakennelmilla ja kojuilla, jolloin hahmojen liikkumisalue muuttuu varsin yksinkertaiseksi. En nähnyt syytä muuttaa navigaatiojärjestelmän geometriaruudukon kokoa.

5.4.2 Monitasoinen valaistus

Valaistusta vaihdettiin Unreal Engine -sovelluksen Precomputed Lighting Scenarios -ominaisuudella. Suomeksi tämä kääntyisi esilasketuiksi valoskenaarioiksi tai yksinkertaisesti, valoskenaarioiksi. Nämä ovat eräänlaisia virtuaalisia maailmoja, jotka eivät sisällä pintamalleja, vaan valonlähteitä. Tässä pystytään luomaan useita valoskenaarioita, nämä voivat vaihdella esimerkiksi tapahtumien alku-, väli- ja loppupisteiden välillä. Esimerkkinä voisimme valaista

ympäristön aurinkoisen aamupäivän mukaiseksi, toiseksi iltapäivän pilvisen vesisateen ja illan pimeäksi katuvalojen korvatessa puuttuvan auringon- ja epäsuoran valon.

Kuten esilasketun valoskenaarion ensimmäisestä sanasta voi saada jo selvää, vaatii valoskenaariot valaistuksen esilaskentaa, jotta voimme hyödyntää valoskenaarioita. Unreal Engine rajoittaa valoskenaarioiden käytön yhteen kerrallaan, johtuen siitä, että näistä lasketaan valokartat. Kuten luvussa 5.1.4 totesin, että emme pysty yhdistelemään valokarttoja, joka toisaalta olisi varteenotettava ominaisuus. Koska valoskenaariot käyttävät valokarttoja objektien valaistuksen tallentamiseen, virtuaalisen maailman suorituskyky on korkealla. Suorituskyvyn alenemiseen voi liittyä liian suuriresoluutioiset valokarttakoot objekteissa. Valoskenaariot vaihdetaan Blueprint-järjestelmässä, mutta editorissa näitä vaihdetaan Levels-näkymästä.

5.4.3 Äänet ympäristössä

Äänet ovat keskeisin osa ihmisten aisteja, siinä missä kosketus ja näkökin. Unreal Engine -luomismoottorissa Audio Engine -järjestelmän avulla pystytään tuomaan ääniä virtuaaliseen maailmaan tehostaen immersiota. Unreal Engine tukee ainoastaan WAV-formaatin audio-tiedostoja (Epic Games 2024i). Audio Engine hyödyntää monikanavaista äänentoistoa, joita ovat esimerkiksi stereo- ja surround-äänet. Äänenlähteiden sijainteja voidaan kuulla hyvin tarkasti, koska Audio Engine tarjoaa mahdollisuudet äänensuuntaukselle ja spatiaaliselle äänelle.

Projektissa on käytössä kaksi skenaariota, kuten yllä mainittiin, päivä ja ilta. Tunnelma vaihtuu näissä kahdessa päivänajoissa huomattavasti, valoisasta katuvaloilla täydennettyyn pimeään kauppatoriin. Päiväskenaariossa kauppatorilla on paljon liikkuvia ihmisiä, kun iltaskenaariossa ei juurikaan.

5.5 Elokuvateollisuuden efektit ratahankkeessa

Ratahanketta tahdottiin esitellä uusilla tavoilla. Visualisoinnin projektissa tahdottiin lähteä liikkeelle filmiteollisuudenkin hyödyntämällä visuaalisten tehosteiden toteutuksilla, joita tässä projektissa tarkasteltiin ja laitettiin täytäntöön. Suunnitteluaineisto upotetaan nelikopterin tuottamaan videokuvaan ja hyödynnettiin Unreal Engine -sovellusta renderöimään kuvasarjat suunnitteluaineistosta upotusta varten. Unreal Engine valittiin tähän kiireisen aikataulun vuoksi, koska se hyödyntää reaaliaikaista renderöintiä, joka puolestaan nopeuttaa kuvien tulostamista jälkikäsittelyä varten.

5.5.1 Digitaalinen kompositointi

Jotta useat eri elementit saadaan yhteen, käytetään kompositointia. Kompositointia voidaan suorittaa digitaalisesti kuin fyysisesti, mutta tässä otan tarkastelun alle digitaalisen kompositoinnin. Kompositointia voidaan suorittaa, kun halutaan yhdistää vähintään kaksi eri elementtiä, esimerkiksi kuvan vuoristosta ja virtaavasta joesta. Oikeaoppisesti yhdistämällä nämä kaksi elementtiä saadaan kuva vuoristosta, josta virtaa joki. Pelkästään kahden elementin liittäminen yhteen ei välttämättä ole riittävää, vaan sitä täytyy täydentää, kuten reunanpehmennyksillä, värikorjauksilla ja perspektiivimuutoksilla.

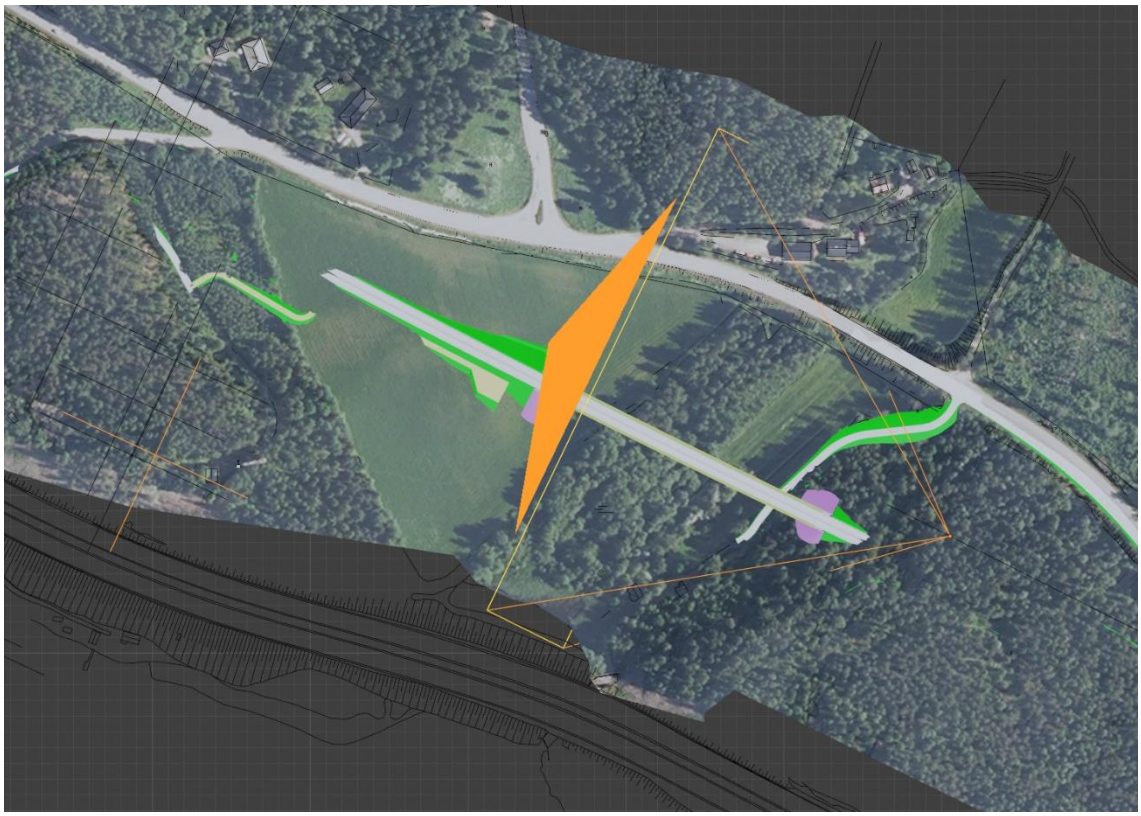
5.5.2 Aineiston työstäminen

Jotta aineistot saadaan liitettyä yhteen, täytyy aloittaa laskemalla fyysisen kameran perspektiivi sekä lentoranta. Valitsin yhdessä kollegani kanssa tähän työhön Blender-mallinnusohjelman, koska se on työssä todettu riittäväksi työkaluksi niin mallinnuksessa kuin animoinnissa. Ohjelmasta löytyy sisäänrakennettuna ominaisuus, jolla lasketaan videosta kameran liikerata sekä näennäinen perspektiivi. Jotta liikerata saadaan onnistuneesti laskettua, asetetaan vähintään kahdeksan kappaletta kiintopisteitä, joita ominaisuus hyödyntää liikeradan laskennassa. Liikeradan laskennassa tulisi videoaineiston olla pituudeltaan lyhyitä, koska ominaisuus kadottaa helposti kiintopisteitä, jos

ne muuttuvat liian kauan. Kiintopisteitä voidaan kuitenkin määritellä uudelleen, jos aiempi kiintopiste katoaa.

Projektissa videot olivat pitkiä, jopa yli viisikymmentä sekuntia. Ongelmaksi muodostui jatkuvasti liikeratojen laskennassa vakiokiintopisteiden puute, eli kiintopisteet, jotka eivät muutu tai eivät häviä laskentaruudukolta. Liikeratojen laskemiseen käytettiin huomattavasti enemmän aikaa, kuin käytin Unreal Engine -luomismoottorissa virtuaalimaailman toteutuksessa.

Kun videoaineistoista oli saatu laskettua liikeradat, tuli sen jälkeen asemoida ja skaalata virtuaalinen kamera oikealle kohdalle (kuva 10). Ominaisuus ei voi tietää fyysisen kameran ja oikean maailman tietoja, vaan ne tulee selvittää manuaalisesti.



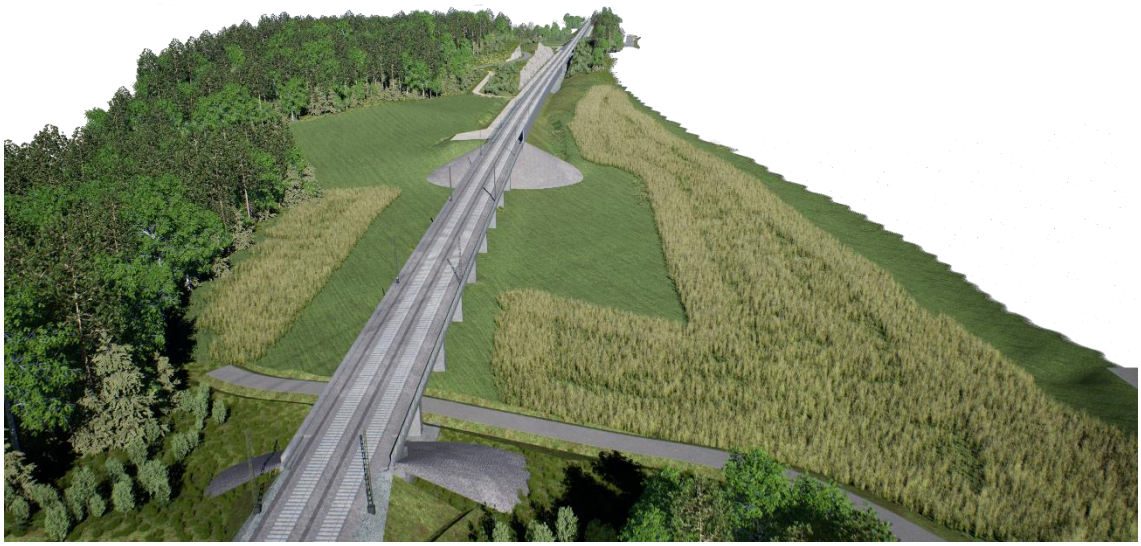
Kuva 10. Aineisto mallinnusohjelmassa ortografisessa näkymässä ylhäältä kuvattuna (Kuva: Tuisku Saarelainen).

Työstin skaalausta ja asemointia usean näytön avulla. Yhdellä näytöllä katsoin virtuaalisen kameran kautta (kuva 11), kun toisella näytöllä katsoin aineistoa ja

virtuaalista kameraa ylhäältä päin ortografisessa tilassa (kuva 10). Työ on osittain haastavaa siksi, koska en voinut määritellä yksittäisen kohteen oikeita kokoja perspektiiviväristymien vuoksi. Tämä antaa viitteellisen skaalan, jonka turvin jatkoin virtuaalisen kameran liikeradan sijoittelua ja skaalaamista virtuaaliseen ympäristöön.

Kuva 11. Aineisto mallinnusohjelmassa, nelikopterin kamerasta laskettavan liikeradan kameran näkymästä (Kuva: Tuisku Saarelainen).

Kun sain suoritettua liikeradan laskemisen ja asemoitua virtuaalisen kameran kulkemaan videoaineiston mukaisesti samaa liikerataa myös virtuaalisessa maailmassa, aloitin virtuaalisen kamera-aineiston viemisen Unreal Engine -luomismoottoriin. Hyödynsin tässä kollegani kanssa Blender For Unreal Engine -lisäosaa Blender-mallinnusohjelmassa, jonka avulla kyettiin viemään lasketun liikeradan ja virtuaalisen kameran Unreal Engine -projektiin. Kun olin vienyt aineiston lisäosaa hyödyntäen Unreal Engine -projektiin, asemoin kameran projektissa hyödyntäen samoja lokaatio, skaala ja rotaatioparametrejä kuin Blender-mallinnusohjelmassa. Nyt virtuaalinen kamera liikeratoineen on valmiina renderöintiä varten (kuva 12).



Kuva 12. Valmis aineisto Unreal Engine -sovelluksesta tulostettuna (Kuva: Tuisku Saarelainen).

Unreal Engine tukee läpinäkyvyyden jakamista Post Processiin (Epic Games 2024j), jota hyödynnetään videoiden renderöinnissä. Kun olin renderöinyt videosekvenssin projektista, pystyin viemään aineiston videonkäsittelysovellukseen. Videoiden käsittelyyn olen tottunut käyttämään Davinci Resolve -nimistä ohjelmaa. Ensin ohjelmassa asetetaan aineiston virkistystaajuus vastaamaan samaa virkistystaajuutta kuin alkuperäisessä, nelikopterilla tuotetussa videoaineistossa. Kun olin tehnyt tämän, toin Unreal Engine -projektissa renderöidyt videosekvenssit ja alkuperäisen videoaineiston videonkäsittelyohjelmaan. Renderöidyt ja kuvatut videoaineistot liitetään päällekkäin. Aineisto näyttää tässä vaiheessa hyvältä, liikeradat täsmäävät ja aloitus sekä lopetus on täysin samat (kuva 13). Ongelmaksi muodostuu kuitenkin renderöidyn videon värit, tarkemmin sanottuna värikylläisyys. Kohdistin värikylläisyyttä siten, että se muistuttaa samaa värikylläisyyttä kuin nelikopterilla kuvatussa aineistossa.

Kuva 13. Valmis aineisto upottamisen ja videoeditoinnin jälkeen (Kuva: Tuisku Saarelainen).

5.6 Valtatien visualisointia

Visualisoimme tiesuunnittelmaa Satakuntaan, jonka hankkeen tarkoituksena on parantaa ja nopeuttaa raskaan liikenteen kulkuyhteyksiä. Visualisoinnin projektissa kiinnitetään erityistä huomiota metsän generoimiseen Suomen ympäristökeskuksen tuottamasta Corine maanpeitepaikkatietoaineistosta sekä sen ongelmista nykyisessä muodossaan reaaliaikaisrenderöintisovelluksessa.

5.6.1 Corine maanpeite

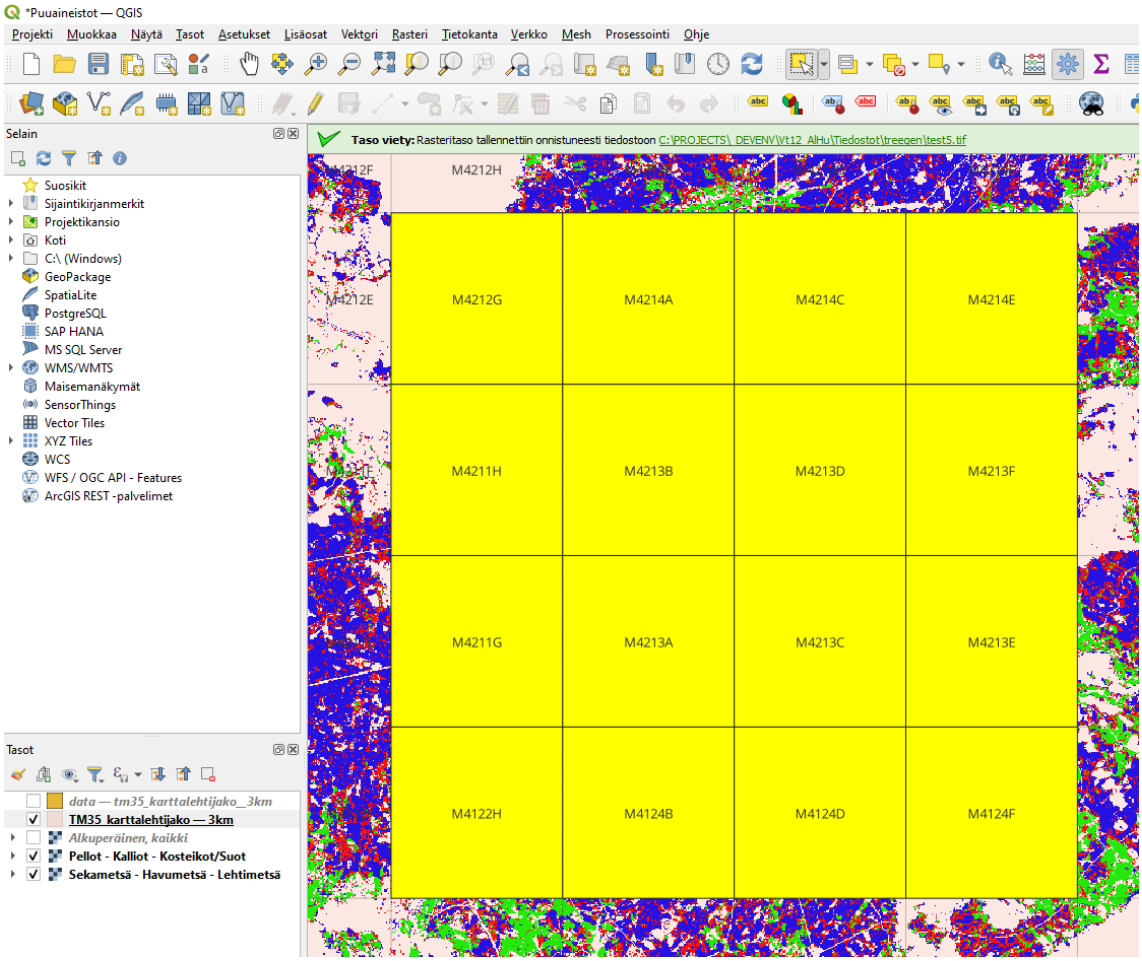
Corine maanpeite on rasterimuotoinen paikkatietoaineisto, joka kuvaa maankäyttöä sekä maanpeitettä koko Suomen alueelta. Aineiston tarkkuus on 20 m * 20 m per pikseli. Corine maanpeite on avointa paikkatietoaineistoa ja ladataan paikkatietosovellukseen WMS-rajapinnan kautta (Suomen ympäristökeskus 2024). Maanpeitteellä tarkoitetaan maanpinnan päällistä kasvillisuutta ja olosuhteita. Tällä voidaan eritellä, kasvaako paikalla esimerkiksi lehti- vai havupuita tai onko alueella peltoja, kallioita tai kosteikkoa. Tämä aineisto ei kuitenkaan kerro, minkä lajin kasvillisuutta alueella kasvaa, vaan ne ovat määritetty kasvupaikan yleistiedon mukaisesti. Aineisto on

luokiteltu 20 m tarkkuudella ja yhden lehtijaon mukainen alue kattaa 6 km * 6 km = 36km^2 aineistoa. Tällöin yksi karttalehti kattaa kokonaisuudessaan 90 000 pikseliä.

5.6.2 Projektin työstäminen

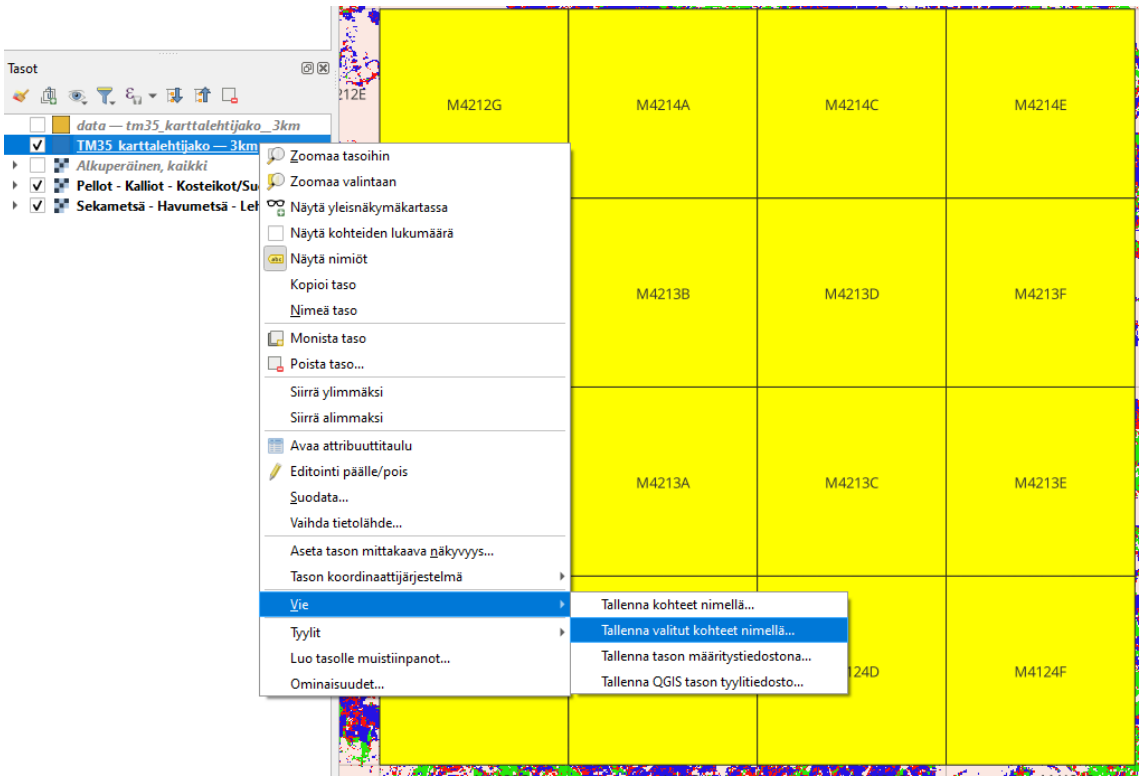
Aloitin visualisointiprojektin hakemalla tausta-aineistoa työn tueksi. Tallensin QGIS-sovelluksesta Maanmittauslaitoksen aineistoa, käyttäen palveluntarjoajan WMS-rajapintaa. Rajapinta tulostaa hakemiston käytettävistä aineistoista. Jotta pystyin erittelemään karttalehtijaon mukaisen tiedostolatauksen, hain karttalehtijaon Maanmittauslaitoksen avoimesta aineistosta. Aineisto on ladattavissa ZIP-pakettitiedostona. Purettuna ZIP-tiedosto sisältää GeoPackage-tiedoston, joka liitetään paikkatietosovellukseen.

Aloitin ensin valitsemalla käytettävät karttalehdet alueen määritystä varten. Tämä tapahtuu QGIS-ohjelman ”Valitse kohteita”-työkalun avulla (kuva 14). Tämän jälkeen tallennetaan valitut kohteet.

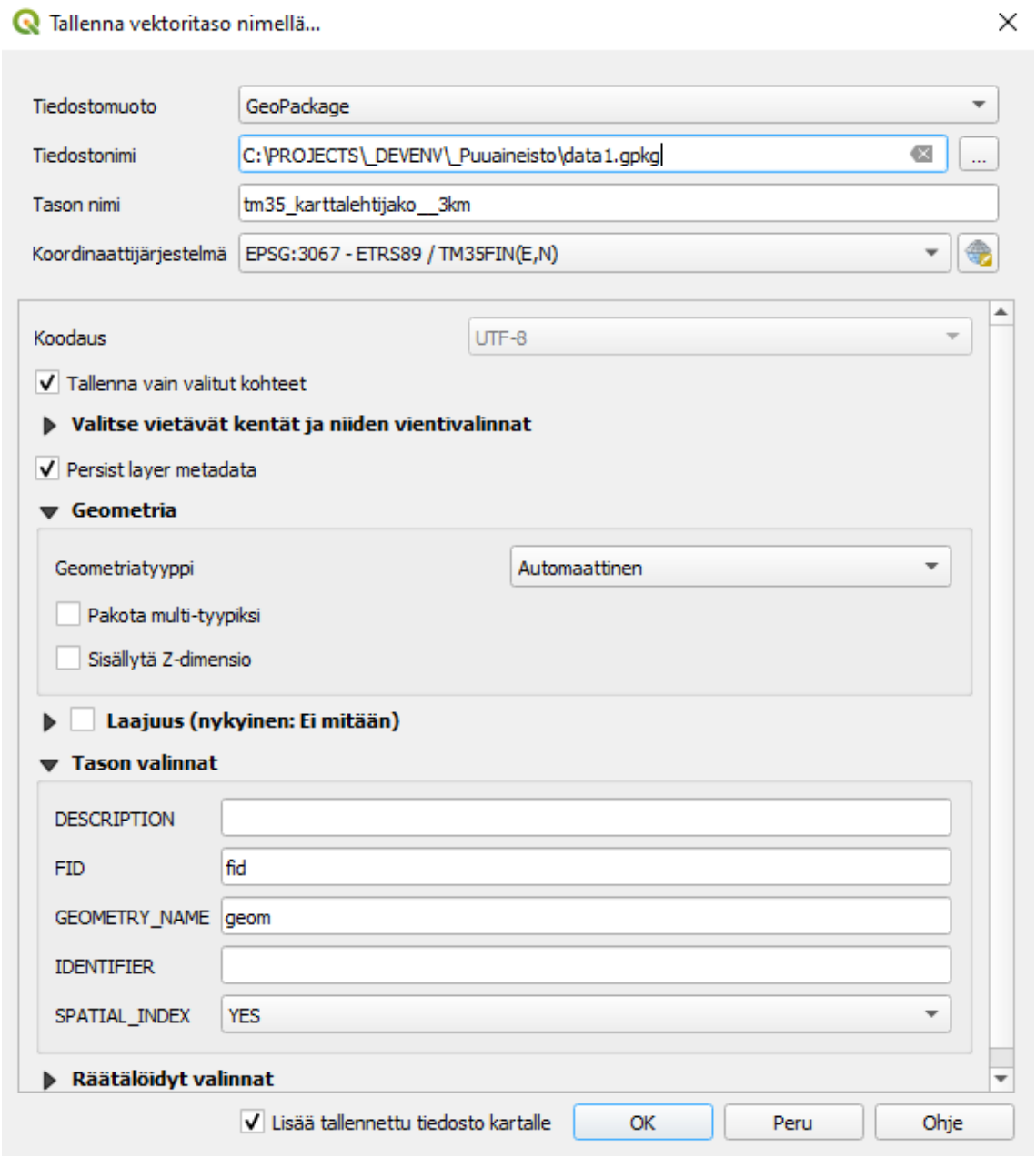


Kuva 14. Näkymä valituista kohteista QGIS-sovelluksessa (Kuva: Tuisku Saarelainen).

Valitsin halutulta alueelta ladatun karttalehtiaineiston mukaisen alueen, jonka jälkeen tallensin tämän erilliseen tiedostoon paikallisesti tietokoneen kovalevylle (kuvat 15 ja 16). Valittu paikkatietosovellus, QGIS, ei tarjoa ominaisuutena viedä valittuja elementtejä tasoista uusiin tasoihin ilman niiden erillistä tallentamista.



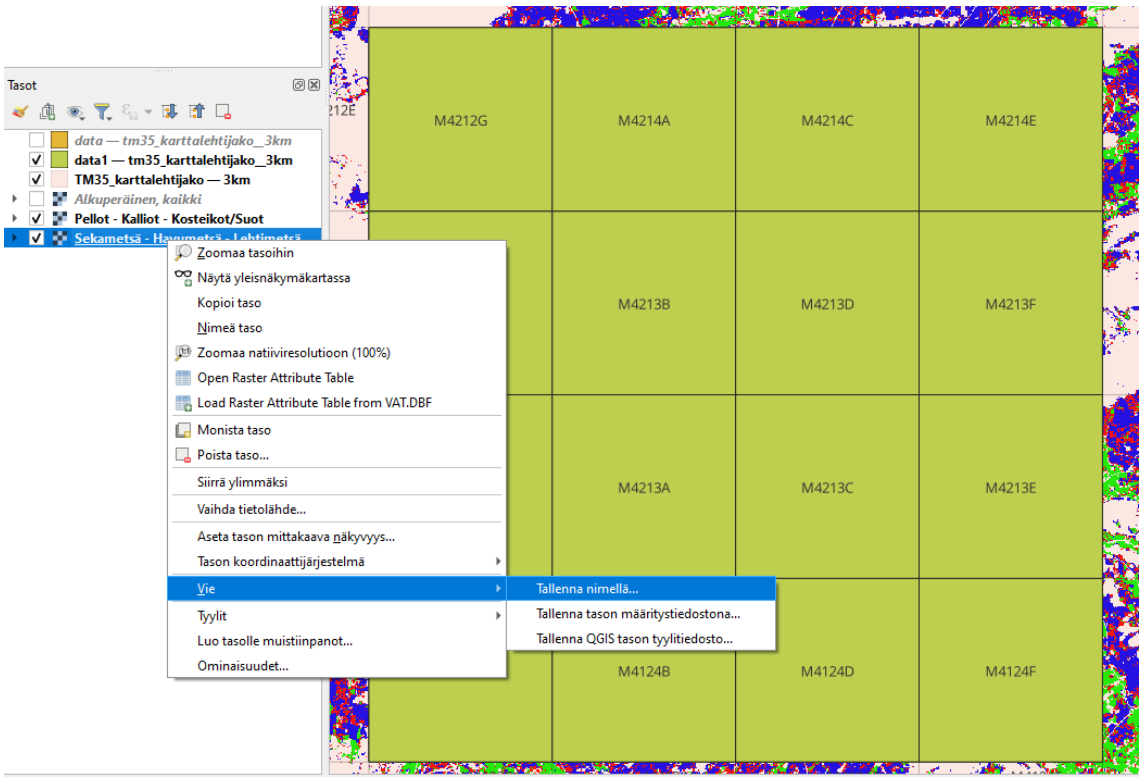
Kuva 15. Valitut karttalehtijaot esiintyvät QGIS-sovelluksessa keltaisina (Kuva: Tuisku Saarelainen).



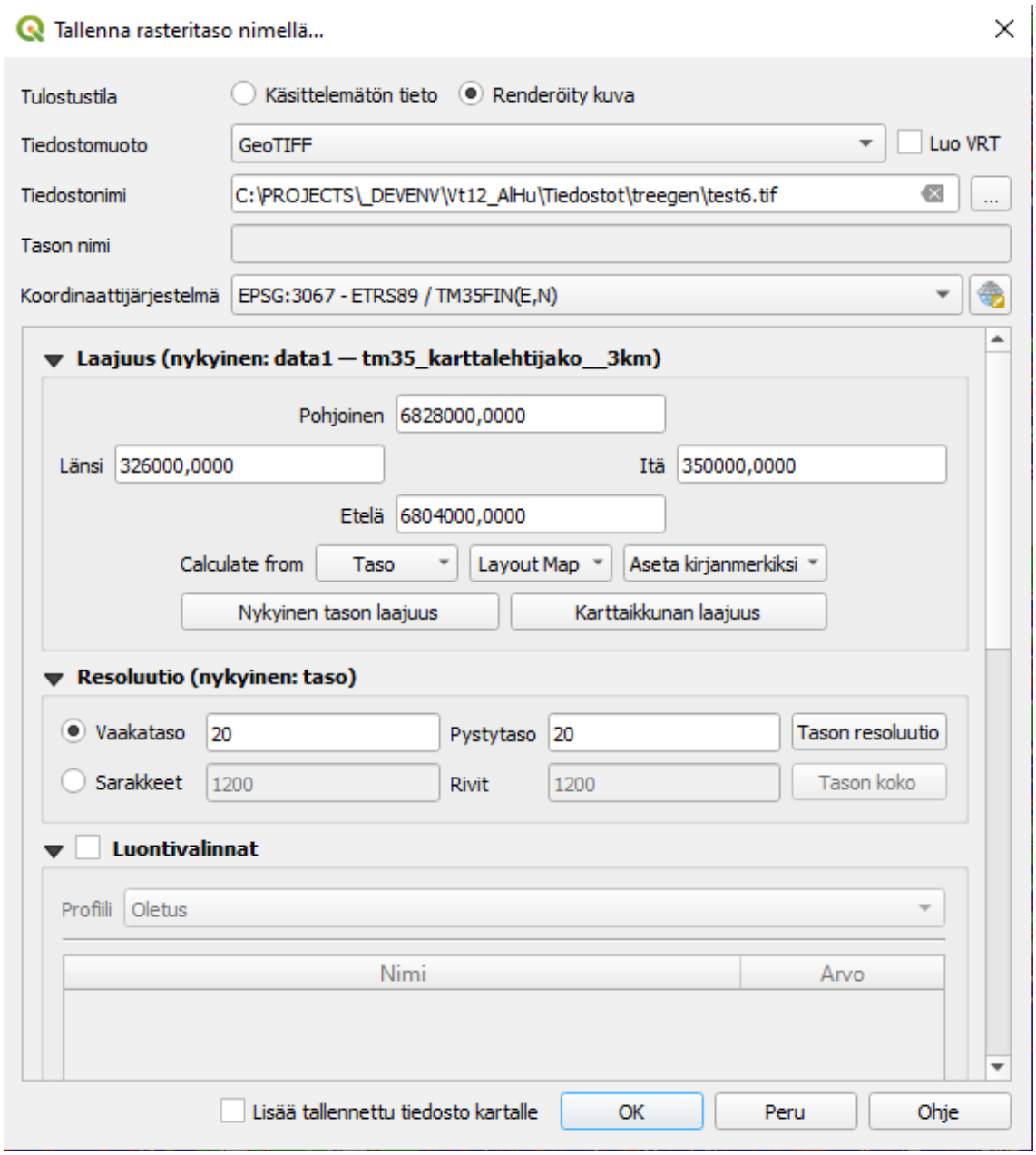
Kuva 16. Valitut karttalehtijaot tallennetaan kuvan tiedoilla omaksi GeoPackage-tiedostoksi. Tallennuksen jälkeen nämä viedään automaattisesti QGIS-sovellukseen omaksi tasoksi (Kuva: Tuisku Saarelainen).

Aineisto tallennettiin renderöitynä kuvana (kuva 17), koska QGIS-sovelluksessa tasoon määritetyt värikoodit eivät tulostu käsittelemättömään tietoon. Värikoodit ovat kriittisessä avainasemassa puuaineiston automaattisesta luomisesta Unreal Engine -sovelluksessa, koska hyödynnetään tulostetun aineiston värikanavia. Olin jo valmiiksi määrittänyt halutut Corine maanpeitetiedot omille värikanaville, tähän kuuluu mm. seka-, havu- ja lehtimetsät. Voin kerralla

tallentaa vain kolme Corine-maanpeitetietoa, johtuen kuvatiedostojen värikanavien määrästä. Laajuus-osiossa (kuva 18) määritetään tasosta renderöitävä alue, jonka olen aiemmin tallentanut. Resoluutioon tuli asettaa vaaka- ja pystytasossa arvoksi 20. Tämä kuvastaa 20 m tarkkuutta per pikseli. Arvon asettamisen jälkeen sovellus laskee automaattisesti sarakkeet ja rivit, joiden arvot puolestaan vastaavat resoluutiossa pikseleitä.

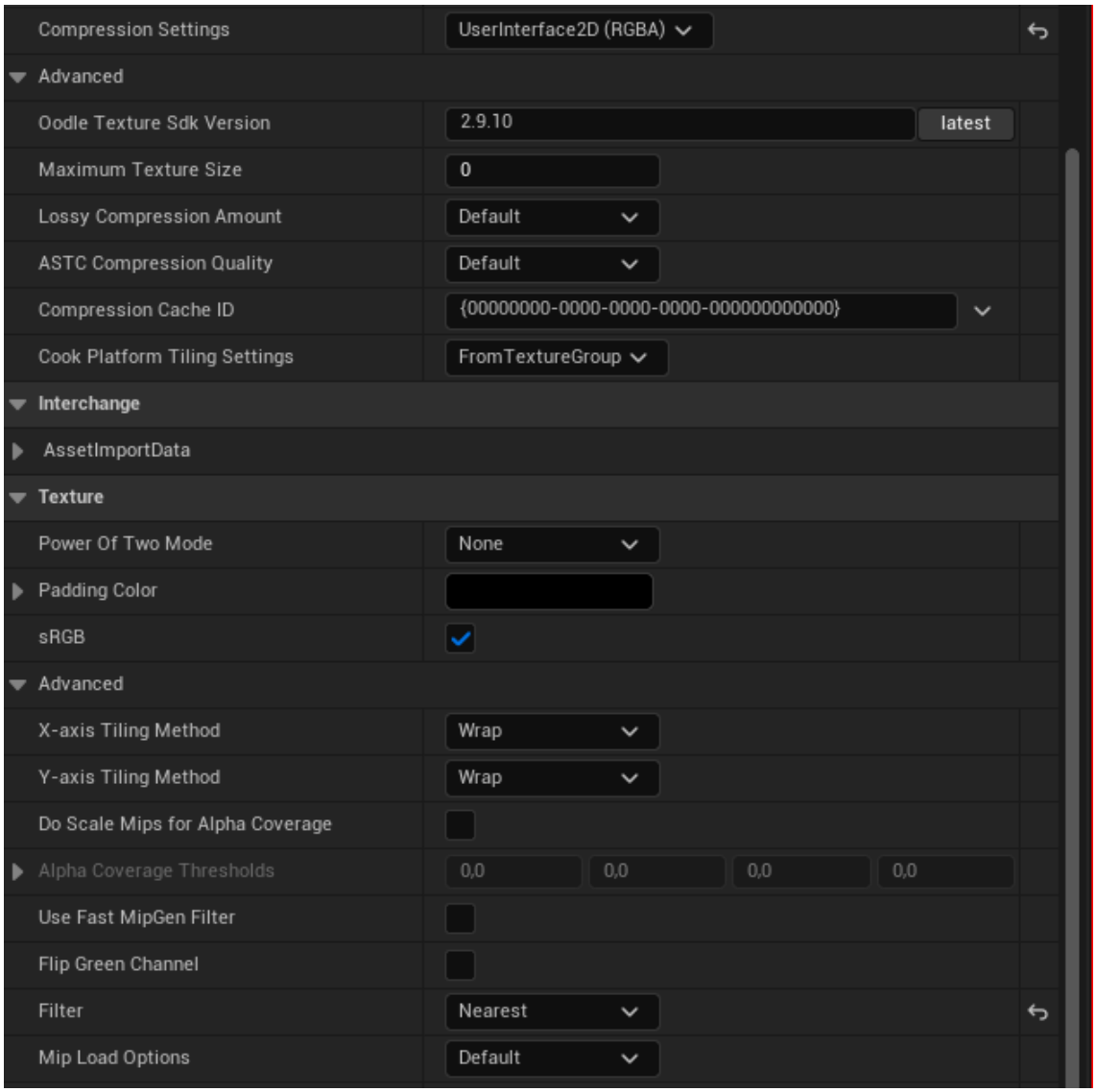


Kuva 17. Uusi karttalehtijakotaso auttaa määrittelemään koordinaatit renderöitävää tiedostoa varten (Kuva: Tuisku Saarelainen).



Kuva 18. Renderöitävän Corine maanpeitekuvan asetuksia (Kuva: Tuisku Saarelainen).

Näiden asetuksien ja arvojen määritysten mukaan pystyttiin tulostamaan valitusta tasosta tarvittavat tiedot. Kun aineistoa viedään Unreal Engineen, tarkistettiin seuraavat asetukset: Compression settings -parametrin arvo tulisi säätää UserInterface2D sekä Filter-parametrin arvo Nearest (kuva 19). Tämä johtuu siitä, millaisena Unreal Engine käsittelee tekstuuriaineistoa ja miten se renderöidään.



Kuva 19. Maanpeitekuva vaatii Unreal Engine -sovelluksessa vielä täsmennyksiä asetuksiin (Kuva: Tuisku Saarelainen).

Kun olin käsitellyt maanpeitekuvan Unreal Engine -projektiin, lisäsin sen valmiiseen maanpinnan materiaaliin. Lisäyksen yhteydessä säädin maanpeitteen UV-skaalan oikeaksi, jotta se täsmää maanpinnan paikkatietoaineistoa. Parametrien säätämisten jälkeen Unreal Engine alkaa automaattisesti asettelemaan kasvillisuutta maanpeitteen mukaisille alueille, jotka ovat valmiiksi määritelty omiin Grass type-tiedostoihin. Ongelmia tämän kanssa ilmenee, kuten Unreal Engine generoi suurille maanpinta-alueille todella hitaasti.

5.7 Maanpinnan toteuttaminen

Uusimaalainen kaupunki toteuttaa katusaneerausta sekä lisää elinvoimaisuutta merelliseen kaupunginosaansa. Visualisoinnin projektin tavoitteena on parantaa ja nopeuttaa päätöksentekoa, kun kuntalaisilla on selkeä näkymä toteutettavasta saneerauksesta. Projektissa paneudutaan päivitettyyn tapaan tuottaa virtuaalinen malli maanmittauslaitoksen aineistosta sekä kaupungin tarjoamasta ilmakuvasta.

Havainnollistavat materiaalit, kuten videot, kuvat ja virtuaalimalli ei yksinään suunnitteluaineiston kanssa ole mitään ilman pohjarakennetta, maanpintaa. Vuosien 2019 ja 2024 välillä päivitin tapaamme luoda virtuaalinen maanpinta hyödyntäen Maanmittauslaitoksen tarjoamaa korkeuskartta-aineistoa ja tässä käsittelen yhtä keskeisimmistä sovelluksista, jonka avulla saimme tuotua maanmittauslaitoksen korkeuskartta-aineiston Unreal Engine -sovellukseen.

5.7.1 Maanmittauslaitoksen maastotietoaineistot

Maanmittauslaitos tarjoaa avoimesti paikkatietoaineistoa, kuten korkeuskartta- ja ortoilmakuva-aineistoa (Maanmittauslaitos). Hyödynnämme näitä aineistoja kaikissa projekteissa, joita tässä portfoliopohjaisessa opinnäytetyössä esittelen. Päivitin tämän projektin yhteydessä tapaamme luoda maanpinnan Unreal Engine -projektissa. Pystymme lataamaan Maanmittauslaitoksen maastotietoa, joihin lukeutuu maanpinta sekä ortoilmakuva. Ortoilmakuvaa en valinnut tähän projektiin hyödynnettäväksi Maanmittauslaitoksen aineistoista, vaan käytin kaupungin tarjoamaa avointa paikkatietoaineistoa, joka sisältää tarkan ortoilmakuvan.

Maanpinta-aineisto koostuu Maanmittauslaitoksen aineistoissa kahdessa eri formaatissa, TIFF-kuvaformaatissa sekä ESRI ASCII-rasteriformaatissa. Käytin tässä työnkulussa jälkimmäistä formaattia. Käsittelen ESRI ASCII-rasteritiedostot harmaaskaalaiseen, 16-bittiseen Portable Network Graphics-tiedostoon, joka viedään Unreal Engine -sovelluksen maanpintamallin laskemiseksi. Koska tietokoneohjelmat lukevat eritavoin liukulukuja, joudutaan

muuttamaan liukulukujen desimaalierottajaa joko pilkusta pisteeksi, tai pisteestä pilkuksi. Tässä tapauksessa Maanmittauslaitos on määrittänyt ESRI ASCII-rasteritiedoston liukulukujen desimaalierottajaksi pisteen.

Sovellus, jolla käsitellään ESRI ASCII-rasteritiedostoja, lukee liukulukuja pilkullisella desimaalierottajalla. ESRI ASCII-rasteritiedostoja voi olla jopa kymmeniä, jolloin käsityönä pisteiden konvertointi pilkuiksi vie aikaa. Kehitetyllä sovelluksella konvertoidaan halutusta kansiosta kaikkien ESRI ASCII-rasteritiedostojen pisteet pilkuiksi. Kun tiedostot on käsitelty haluttuun sovellukseen ja sovelluksessa, uloskirjoitetaan tästä harmaaskaalainen, 16-bittinen Portable Network Graphics-tiedoston, jota kutsutaan tässä vaiheessa korkeuskartaksi. Tämän tiedoston avulla voidaan luoda 1:1 skaalaisen maanpintamallin Unreal Engine -projektiin. Korkeuskartan harmaasävyt on asetettu 0 ja 255 kokonaislukujen välille, joiden avulla Unreal Engine laskee maanpinnan yksittäisen vektoripisteiden sijainnin Z-akselilla (Epic Games 2024k).

Jotta harmaaskaalainen korkeuskartta saadaan vietyä Unreal Engine -projektiin 1:1 skaalassa, lasken sille XY- ja Z-skaalan. Koska Maanmittauslaitoksen ESRI ASCII-rasteritiedoston yksi tietue vastaa 2 metriä, luulee Unreal Engine skaalan olevan 100, joka on vakio ohjelmassa. Tämä vakioskaala on tarkoitettu korkeuskartoille, joiden yhden pikselikoon tietueen koon oletetaan olevan 1 metri. Käytän omana vakiokokona jokaisessa projektissa aina 200, johtuen Maanmittauslaitoksen korkeuskartta-aineiston kokoluokittelusta. Tämän jälkeen tehdään vielä skaalakonversio, koska valmis harmaaskaalainen korkeuskartta-aineisto täytyy skaalata Unreal Engine -luomismoottoriin sopivaan kokoon, joka usein riippuu projektin maanpinnan laajuudesta. Tässä projektissa käytetyn korkeuskartta-aineiston pikselirivin määrä on 12 000 x 12 000. Määrä ylittää huomattavasti Unreal Engine -luomismoottorin suositellun maksimikoon, joka on 8129 x 8129 (Epic Games 2024l). Koska jouduin skaalaamaan korkeuskartan pienempään resoluutioon, aineisto pieneni 47,619 %. Koska tiedetään korkeuskartan vakioskaala projektiin, kerrotaan XY-skaala kertoimella 1,47619, jonka tulos vastaa maanpinnan realistista skaalaa virtuaalisessa ympäristössä.

Jotta pystyin selvittämään Z-skaalan korkeuskartalle, olin kirjannut uloskirjoitettavasta sovelluksesta tärkeitä tietoja ylös. Näihin tietoihin sisältyy korkeusaineiston korkein ja matalin piste. Koska sovellus kääntää minkä tahansa arvon 0 ja 100 välille, korkeusaineiston matalin piste vastaa arvoa 0 ja korkein piste arvoa 100. Tämä tarkoittaa, että vaikka korkeimman pisteen arvo olisi enemmän kuin 100, häviää tästä tärkeää tietoa Z-akselin skaalan laskemista varten. Projektissa korkeuspisteiden arvot olivat 13,417 ja 91,241, jolloin meidän ei tarvinnut tehdä konversiolaskentaa korkeuskartalle. Lasketaan yksinkertaisesti Z = (korkeus * 100) * (1/512), jolloin saadaan selville Z-skaala. Edellä mainituilla laskennoilla tiedetään korkeuskartan skaala.

Näiden korkeuskarttojen laskentakaavojen ratkaisemisessa kesti useita vuosia, joista en ole ollut tyytyväinen. Useissa projekteissa tehtiin kipeitä ja vääriä ratkaisuja huijataksemme visuaalisesti loppukäyttäjiä. Vaikka maanpinta ei täysin tule keskeiseen rooliin, on sillä ollut suuri rooli suunnitelma-aineistojen yhteensovittamisessa Z-akselilla.

5.7.2 Ortoilmakuvat paikkatietoaineistoista

Maanpintamalliin ladataan samalla kerralla sopivat ortoilmakuvatiilet, jotka käsitellään UDIM-formaattiin. Tällä UDIM-formaatilla Unreal Engine luo virtuaalisen kokonaistekstuurin useasta eri tekstuurista. Unreal Engine -sovelluksessa virtuaalinen kokonaistekstuuri asetetaan maanpinnalle kehitettyyn materiaaliin, jossa lopuksi käsitellään sen UV-skaala kahden eri parametrin avulla. Tilaajalta löytyy kattava paikkatietoaineisto, jotka käsittävät myös ortoilmakuvia. Keräsin ortoilmakuvat WMS-rajapinnan avulla tämän avoimista paikkatiedoista käyttäen QGIS-sovellusta. Käytin samaa prosessia kuvien lataamisessa kuin 5.6.2 kappaleessa, mutta jälkikäsittelin kuvat UDIM-formaattiin. Joissakin projekteissa, joissa tilaajalla ei ole luovuttaa ortoilmakuvia, käytetään Maanmittauslaitoksen tarjoamia aineistoja.

UDIM-formaatin kuvat koostuvat useista yksittäisistä kuvista, jotka yhdistetään virtuaaliseksi kokonaiseksi tekstuuriksi. Kuville annetaan valmiiksi määritelty numeraatio, joka alkaa 1001 ja teoreettisesti päättyen 9999, jolloin kuvien

määrä olisi maksimissaan 8999 (Foundry 2024). Kyseisellä tekniikalla liitän ortoilmakuvat UDIM-formaattiin, jota Unreal Engine -luomismoottori tukee.

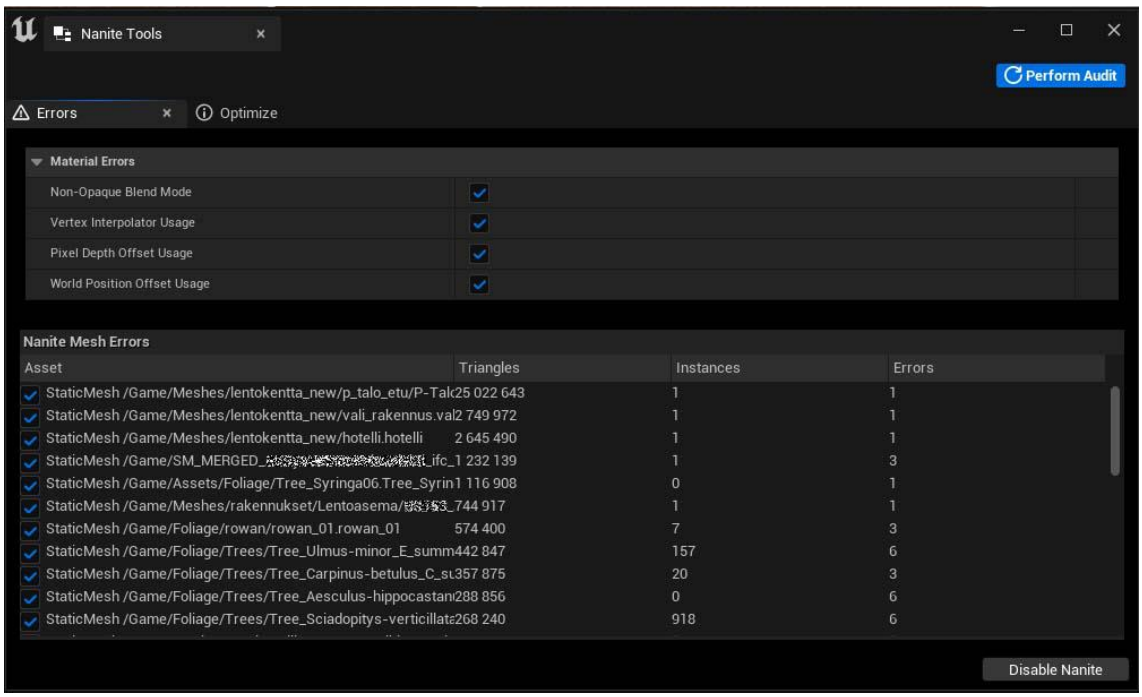
5.8 Uusien tekniikoiden hyödyntäminen optimoinnissa

Uusimaalainen kaupunki laajentaa ja lisää elinvoimaisuutta toteuttamalla lentoasemalta lähtevän ja päättyvän ratikkaliikenteen. Tässä projektissa haluttiin tuoda esille käytettävän sovelluksen päivittämisen toimivuuden versiosta uudempaan sekä vanhan aineiston kääntämistä käyttämään uusimpia teknologioita reaaliaikaisrenderöintiin. Unreal Engine -sovelluksen uusimpien teknologioiden käyttöönottoa tarkastellaan, koska niitä otettiin käyttöön kesken visualisoitavan projektin.

5.8.1 Nanite -virtualisoidut geometriat

Koska reaaliaikaisella renderöinnillä ja erityisesti näytönohjaimilla on omat haasteensa laskea miljoonia kolmioita ja raskaita pintamalleja, ei kyetä käyttämään tavallista fotogrammetriasovelluksen uloskirjoittamaa raskasta pintamallia ilman raskaita optimointitekniikoita. Vuosien kehityksen aikana reaaliaikaista renderöintiä hyödyntävä Unreal Engine -luomismoottoriin julkaistiin virtualisoidut geometriat, Nanite (Epic Games 2021).

Geometrioiden käännöstyö Naniteksi Unreal Engine -sovelluksessa on automatisoitu. Käännettävät geometriat valitaan pikavalikkoon, jossa pystyin muuttamaan nämä Naniteksi. Geometrioita muutetaan Naniteksi käyttämällä Nanite Tools-työkalua (kuva 20). Tällä työkalulla haetaan geometrioita kolmioiden määrän perusteella sekä avata. Hyödynsin tätä tekniikkaa projektissa sen nopean käytön vuoksi.



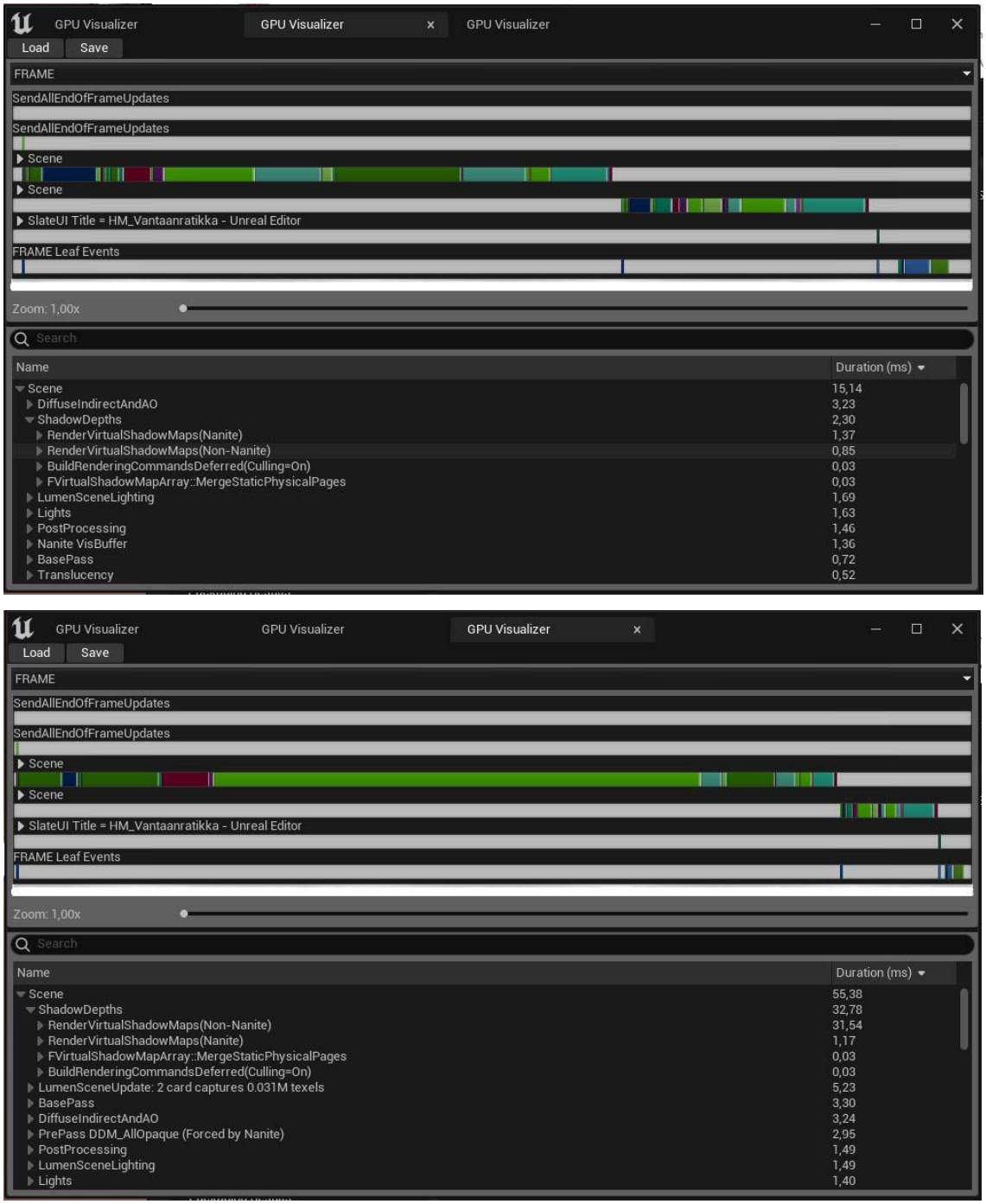
Kuva 20. Nanite Tools -työkalun valikko (Kuva: Tuisku Saarelainen).

Toisinaan geometriatiedostot voivat olla raskaita koneelle käsiteltäväksi. Edellinen tietokoneeni ei kyennyt käsittelemään projektin parkkitalon mallin julkisivua, jossa oli yli 25 miljoonaa kolmiota geometriassaan. Unreal Engine -projekti kaatui eikä suostunut käynnistymään, koska se käytti kaikki tietokoneen muistit, joita tuolloin koneessani oli 64 Gt. Projektin myötä sain uuden tietokoneen, jossa oli 128 Gt keskusmuistia, jolla ei ollut ongelmia käsitellä kyseistä julkisivun tiedostoa.

Nanitesta on ollut paljon hyötyä, kun ajatellaan pintamallien optimointia. Huomasin, että pystyimme helposti luopumaan osan pintamallien optimoinnista melkein kokonaan. Ongelmaksi on muodostunut kasvillisuuden pintamallien läpinäkyvyyksien renderöiminen, kuten aiemmin luvussa 5.2.3 mainittiin. Näissä on usein paljon tyhjää tilaa geometriassaan ottaen huomioon käytettävän tekstuurin Alpha-kanavalle määritetty läpinäkyvyys. Jopa Epic Gamesin kehittäjä suosittelee virallisella opetusvideolla, että Nanite kytkeytyisi pois päältä virtuaalisen kameran etäisyyden kasvaessa isommaksi objektiin nähden (Epic Games 2024m).

Kun pintamallit oli muutettu Naniteksi, reaaliaikaisen renderöinnin vasteaika laski 3,6 kertaa pienemmäksi verrattuna ennen vaihtoa (kuva 21). Tapauksessa pintamallit, jotka eivät käyttäneet Nanitea, rasittivat suorituskykyä varjojen laskennan osalta. Tämä kertoo, että Nanite kykenee suoriutumaan raskaista geometrioista jopa paremmin kuin käyttäisimme tavallista tarkkuustason käyttöä. Nanitea käyttäessä visuaalinen tarkkuus pysyy lähes ennallaan, koska järjestelmä pyrkii säilyttämään pintojen muodot mahdollisimman muuttumattomina.

Naniten avulla kyetään parantamaan visuaalista laatua jo pelkästään vähentämällä teksturoitujen tekniikoiden käyttöä. Suunnitteluprojektissa toteutettiin kiveysladonta, jossa käytettiin häivyttämistä 4 eri graniittikiven sävyn avulla. Koska kiveys oli laajalla alueella, tällaisen toteuttaminen tekstuureilla olisi ollut vaikeaa. Kiveys toteutettiin Rhino3D-suunnitteluohjelmiston polyline-luokassa (Rhino3D 2024), jonka ansiosta geometriamalli voitiin viedä sovelluksesta sopivaan kolmiulotteiseen tiedostomuotoon. Unreal Engine -projektissa Rhinolla luotu tiedosto tuodaan ja sille laitetaan Nanite päälle.



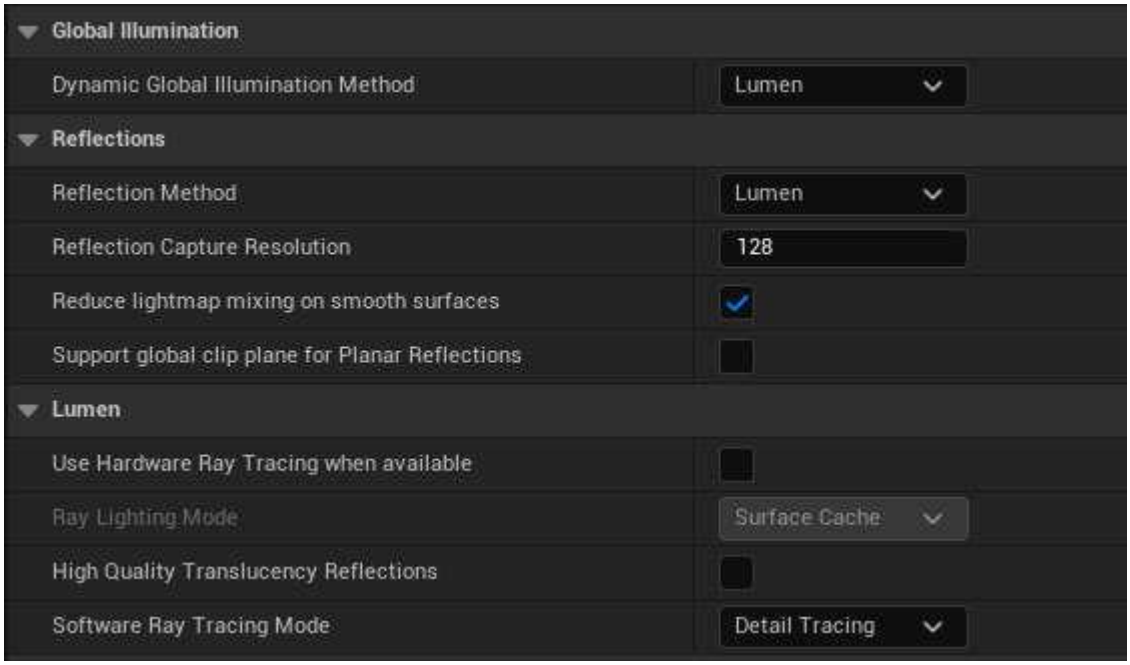
Kuva 21 . Vasteaikojen ero. Yllä osa pintamalleista muutettu Naniteksi, alla tavallinen, tarkkuustasoja käyttävät pintamallit (Kuva: Tuisku Saarelainen).

5.8.2 Lumen ja Virtual Shadow Maps

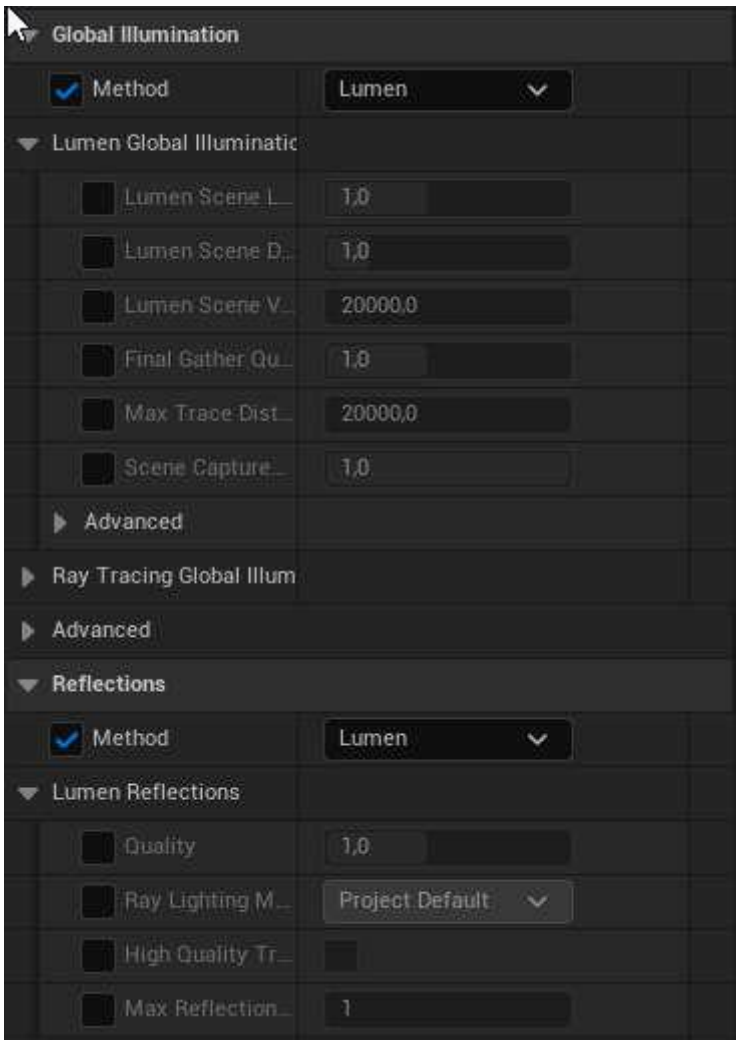
Lumen on valaistuksen ja heijastuksien järjestelmä, joka aloittaa valaistuksen laskemisen Screen Trace -tekniikkalla, jonka jälkeen tukeudutaan joko sovelluspohjaiseen tai laitteistolla kiihdytettyyn säteenseurantaan. Näiden jälkeen lasketaan heijastukset. Lumen järjestelmänään luo pintamalleille

yksinkertaistetut pinnat ja tallentaa nämä Surface Cache -muistiin (Epic Games 2024n).

Lumenin käyttöönotto Unreal Engine -luomismoottorissa on nopeaa. Projektin renderöinnin asetuksista valitaan Lumen laskemaan epäsuoraa valoa (kuva 22) sekä tarvittaessa Post Process-volyymin Lumenin asetuksia säädetään (kuva 23). Lumen kuitenkin vaatii toimiakseen Generate Mesh Distance Fields -asetuksen (Epic Games 2024n), joka vapaasti suomennettuna olisi geometrian etäisyyskentät. Etäisyyskentät ovat kuvapohjaisia presentaatioita objekteista, joiden avulla kyetään laskemaan esimerkiksi epäsuoraa valoa. Kun Lumen on kytketty päälle, tehdään pintamalleihin muutoksia, joissa tarkistetaan ja tarvittaessa muutetaan niiden Distance Field Resolution Scale -arvoa. Tämä arvo määrittää, kuinka tarkka geometrian etäisyyskenttä pintamallista halutaan luoda. Lumen käyttää etäisyyskenttiä valaistuksen laskemisessa, koska ne voivat olla huomattavasti kevyempiä kuin tavalliset pintamallit (Zucconi 2016).

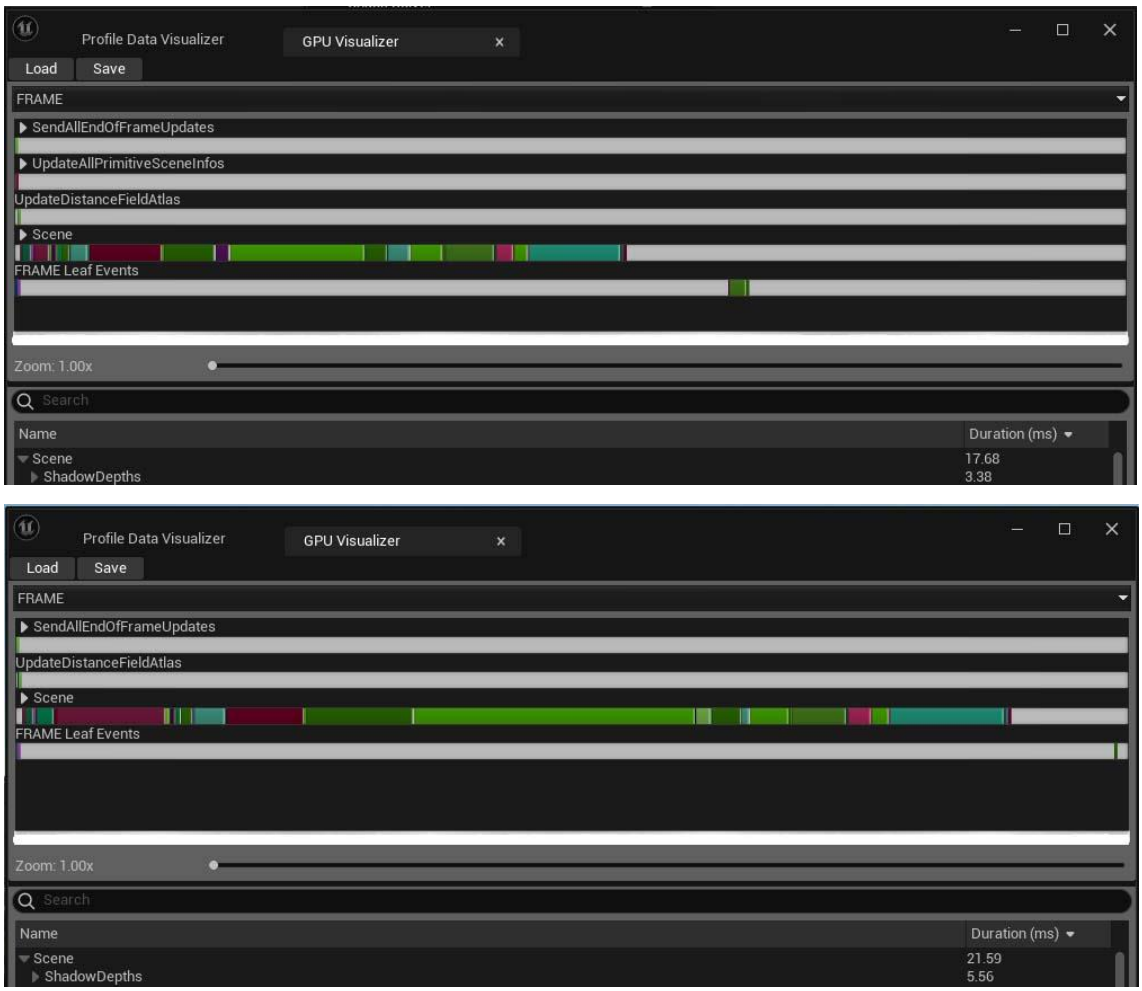


Kuva 22. Lumenin asetukset projektissa (Kuva: Tuisku Saarelainen).

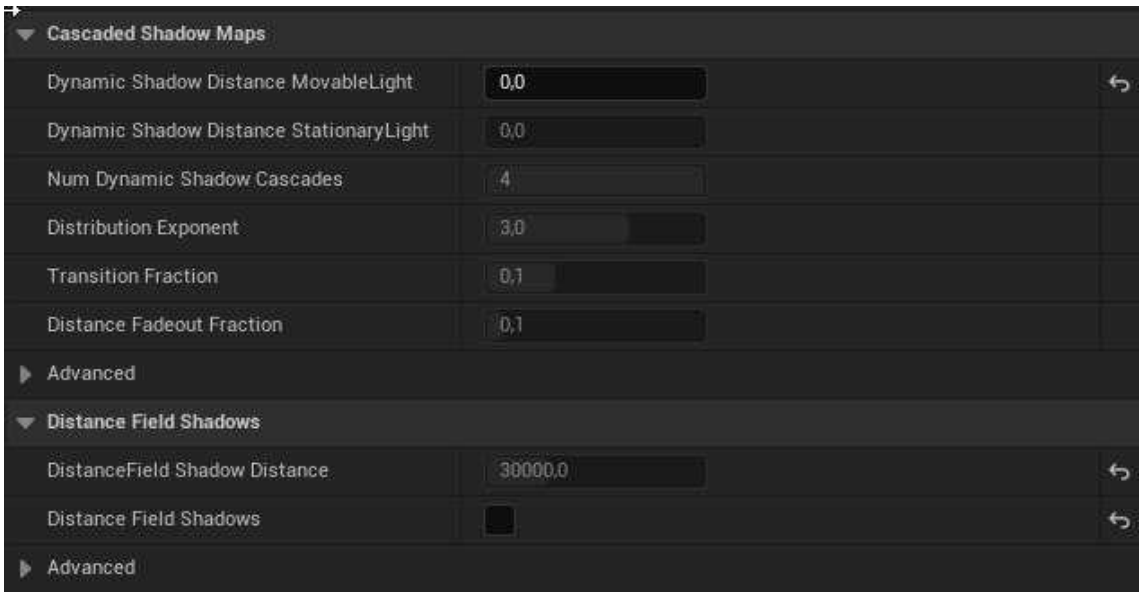


Kuva 23. Post Process-volyymissä Lumenin parametrit (Kuva: Tuisku Saarelainen).

Saadaksemme varjot näyttämään mahdollisimman realistisilta, otin käyttöön Virtual Shadow Maps -varjolaskentatekniikan. Virtual Shadow Maps on nimensä mukaisesti virtuaalinen tekstuuri, jota päivitetään reaaliaikaisesti. Virtual Shadow Maps on huomattavasti kevyempi kuin tavallinen, sarjaankytketyt varjokartat (kuva 24). Käyttäessä tätä ja Mesh Distance Fieldsiä tulee ristiriitaisuuksia, kun tarkastellaan virtuaalisen maailman varjoja. Suoran valon asetuksista tulee asettaa toistaiseksi Dynamic Shadow Distance MovableLight -parametri arvoksi 0 sekä DistanceFields Shadows poistetaan käytöstä (kuva 25). Tämä johtuu siitä, että Unreal Engine tulkitsee käyttävänsä joitain näistä tekniikoista varjojen laskemisessa, vaikka Virtual Shadow Maps olisikin kytketty päälle projektin asetuksista.



Kuva 24. Virtual Shadow Maps (yllä) ja Cascade Shadow Maps (alla) vasteaiat (Kuvat: Tuisku Saarelainen).



Kuva 25. Lumeniin vaikuttavat Dynamic Shadow Distance MovableLight ja Distance Field Shadows -parametrit suoran valon objektin asetuksissa (Kuva: Tuisku Saarelainen).

6 Analysoiva pohdinta

Tekniikan kehittyessä tulisi syventää omaa osaamistaan, vaikka kokisi tuntevansa tekniikka-alueen läpikotaisin. Tätä olen ajatellut jo siitä lähtien, kun aloitin harjoittelun eräässä pohjoiskarjalaisessa pelialan osuuskunnassa. Jo ensimmäisinä päivinä harjoitteluani havaitsin, etten vielä tiedäkään kaikkea tekniikka-alaan liittyen. Tässä portfoliopohjaisessa opinnäytetyössä tarkastelen itseäni kehittyvänä tietojenkäsittelijänä, joka on erikoistunut suunnittelu- ja konsultointialalla toimivaan visualisointiin. Koen työn tekemisen alalla olevan hyvin monipuolista ja innostavaa, vaikkei se ole sitä jota lähdin etsimään aloittaessani harjoittelun. Tässä osiossa tarkastellaan kriittisesti jokaisen projektin kohdalla kehittymistäni.

6.1 Varjoista oppimiseen

Luvussa 5.1 kerrotaan, kuinka valaisin sillan useilla valaisimilla ja kuinka optimoin sitä. Tuolloin vielä päädyttiin optimoimaan raskaasti virtuaalitodellisuuteen pohjautuvia projekteja, koska en vielä ollut tutustunut erilaisiin menetelmiin, joiden avulla varjojen laskemisen vaikutusta renderöinnin vasteaikoihin voitaisiin vähentää. Tuolloin olisi voitu kokeilla geometrioiden etäisyyskenttiin perustuvaa varjolaskentaa, Epic Gamesin mukaan sen käyttäminen olisi huomattavasti edullisempaa verrattuna tavalliseen, sarjaankytkettyihin varjokarttoihin (Epic Games 2024o).

En ollut kuitenkaan kerennyt tutustumaan etäisyyskenttä-tekniikkaan ja siihen tutustuminen olisi vaatinut aikaa; olisi pitänyt tietää, että jokaisen geometriaobjektin kuuluisi olla mieluiten omanaan, kuin isona yhtenäisenä geometriana (Epic Games 2024p). Olisi myös pitänyt tutkia, kuinka paljon sillan alkuperäinen, optimoimaton geometria-aineisto olisi vaikuttanut renderöinnin vasteaikaan. Keskityimme silloin, kuten edelleen, geometrioiden optimointiin. Toisaalta olen vähentänyt mallien optimointia, kun Nanite on tullut osaksi visualisointiprojekteja. Kerroin enemmän Naniten käyttöönotosta ja sen tuomista etuuksista projektia 5.8 käsittelevässä kappaleessa.

6.2 Impostor ja ohjelmointi

Luvun 5.2 projektissa otettiin käyttöön uusia kasvillisuusmalleja, joiden avulla nostettiin visualisointimme visuaalista laatua huomattavasti. Kerroin geometria-aineiston optimoinnista ja kuinka se olisi vaikuttanut vasteaikaan. Kerron tässä osiossa, mitä olisin voinut tehdä toisin. Jouduin todellisen haasteen eteen projektia työstäessämme. Ongelmaksi ilmeni, että uudet kasvillisuusmallimme näyttivät liiankin upeilta ja yksityiskohtaisilta, ja huomasin myös, että niiden geometrioiden kompleksisuus ylitti renderöintiin menevän budjettimme. Koska halusimme nostaa visuaalista laatua, emme voineet tehdä sitä aivan tavallisilla pelejä varten tehdyillä kasvillisuusmalleilla, jotka olisivat olleet todennäköisesti huomattavasti kevyempiä renderöintiliukuhihnaan.

Olin aiemmin tutustunut Impostor -tekniikkaan ennen tätä projektia, mutta sen hetkinen tietämykseni oli varsin vähäistä. Tekniikalla saatiin lisättyä varsin kevyitä tarkkuustasoja kasvien malleihin, mutta ongelmaksi muodostui Impostor-kuvan resoluution suuruus, jos halusimme esitellä kauempia objekteja yhtä tarkkana kuin sen hetken objekti olisi näyttänyt renderöitävältä resoluutioltaan, joiden määritelty näytönpeittokoko vaihteli 5-10% välillä. Havaitsin, että kuvien olisi tullut olla 2^{13}-potenssin resoluutiossa niin, että rivejä ja kolumneja olisi ollut 4, jolloin kuvakulmia kasvillisuusmallista olisi ollut 16, jotta malli näkyisi tarkasti. Yksittäinen kuva kasvista kuvasarjassa olisi siis 2^{11}-potenssin resoluutioinen. Tämän kokoinen kuva alkoi nostamaan renderöinnin vasteaikaa huomattavasti. Tuolloin en vielä ollut ottanut huomioon, että kasvillisuusmallien kuvasarjoja työstäessä jäi huomioimatta myös, että kuvasarjoissa kasvien lehdet vaikuttivat huomattavasti nuukahtaneilta. Tämä johtui siitä, että en ollut huomannut valaista kasveja. Kasvien valaistaminen ennen kuvasarjan toteutusta olisi lisännyt syvyyttä lehtien välisiin tiloihin.

Projektiin toteutettiin myös liikkuvia ajoneuvoja, jotka seuraavat tiettyä polkua. Olen vastuussa projektiemme toiminnallisuuksien ohjelmoimisesta ja niiden elinkaaren ylläpitämisestä sekä parantamisesta. Mielestäni on ollut suuri vääryys toteuttaa Vehicle-objektien toiminnallisuus Blueprint-järjestelmässä, eikä ohjelmoida sitä C++-ohjelmointikielellä. Vaikka Epic Games on parantanut Blueprint-järjestelmän suorituskykyä vuosittain eikä uutta korvaavaa järjestelmää ole esitelty, on sillä edelleen suuria vaikutuksia ohjelman suorituskykyyn (McCole 2023). Säästääkseen aikaa täytyy tehdä hankalia valintoja, jotka vaikuttavat tulevaisuuden projekteihin. Tämä saattaa olla kriittisin asia, jonka mainitsen konsultointialasta.

6.3 Drone-operaattoriksi

Olemme ottaneet osaksi visualisointiprojekteja fotogrammetria-aineiston hyödyntämistä tausta-aineistona. Aineiston työstössä ei juurikaan ole voitu vaikuttaa siihen, millaisessa valaistuksessa aineisto on kuvattu visualisoinnin

käyttöön. Kuvausryhmälle toteutettiin selkeä dokumentaatio, jonka avulla he kykenivät toimittamaan oikeanlaista aineistoa. Kuvaus olisi täytynyt suorittaa pilvisenä päivänä, jolloin valaistus olisi ollut optimaalinen. Kuitenkin kuvaustiimillä oli ollut omat aikataulunsa kuvauksien suhteen, joten saimme aineiston kuvattuna aurinkoisena päivänä. Tämä sotki osittain suunnitelmiani hyödyntää valmista fotogrammetria-aineistoa Unreal Engine -projektissa, jossa suorasta valosta olisi saatu varjojen piirtäminen. Toisaalta saimme myös pientä helpotusta renderöinnin vasteaikaan, koska varjot olivat jo osittain ”laskettu”. Tämän takia en lopulta voinut vaikuttaa myöskään kellonaikaan, jota virtuaalisessa maailmassa olisi simuloitu. Toisen kellonajan käyttö olisi voinut rikkoa immersiota virtuaalitodellisuuden mallissa. Olen ehdottanut, että ottaisin osakseni nelikopterilla toteutettavan kuvauksen, jota hyödynnetään fotogrammetria-aineiston tuottamiseen. Tämä voisi vähentää kuvausryhmään kohdistuvia paineita ja saisimme paremmin käyttöömme kohdennettua aineistoa. Yrityksessä on omat vaatimukset ja koulutukset nelikopterilla kuvaukseen, jotta toiminta olisi yhtenäistä kuvaajasta riippumatta. Suunnitelmissani on suorittaa drone-operaattorin koe mahdollisimman pian tämän opinnäytteen kirjoittamisen jälkeen.

Käyttämäni Luoshuangin GPULightmass -valaistus aiheutti sen, etten voinut säätää näiden väriä tai intensivisyyttä, jos valot käyttivät Stationary -mobiliteettia. Olisin saanut nämä valojen muutokset esilasketuissa valoskenaarioissa käyttämällä muokkaamatonta Unreal Engine -valaistuslaskentaa, mutta samalla olisin joutunut karsimaan epäsuorien valojen aiheuttamista varjoista, jotka olivat isossa roolissa tämän muistomerkin valaistuksessa. Ongelmaksi muodostui myös Stationary-mobiliteettia käyttävien valojen päällekkäisyys, joka on määritetty Unreal Enginessä 4:ään, kun käytettävien valojen määrä ylitti tämän.

6.4 Filmiteollisuuden keinot

Kun ensimmäisen kerran kuulin projektin johtoryhmän ehdotuksen esitellä ratahanketta uudella tavalla, myönnyin pyyntöön hetken harkinnan jälkeen.

Myönnän tässä vaiheessa, että projektin toteutuksen aikana koettiin monenlaisia tunteita. Alkuun oli jännitystä, alakuloa, vihaa ja lopussa intoa ja onnistumisen tunnetta. En ollut valmistautunut toteuttamaan yli kahdenkymmenen sekunnin pituisia nelikopterilla kuvattuja videoita, joita kertyi yhteensä kuusi. Pisin toteutettu videoaineisto oli alle kaksi minuuttia. Ongelmaksi muodostui jokaisen nelikopterilla kuvatun videon kohdalla kiintopisteiden puute, joiden avulla olisin voinut laskea nopeasti kameran liikerataa. Olin ajatellut, että esittelemme hankkeesta vain tietyt, ennakkoon sovitut alueet lyhyinä pätkinä, mutta esittelimme alueet pitkinä kamera-ajoina.

Videoihin pohjautuvaa virtuaalisen mallin työstämisessä ei juurikaan ollut mitään uutta. Tämän toteuttaminen pysyi hyvin pitkälle omissa rutiineissa, jos ei oteta huomioon pitkiä kamera-ajoja ja niiden työstämistä uudelleen useita kertoja päivässä. Jouduin aluksi pohtimaan, miten saisin renderöityä vain tietyt alueet videolle upottamista varten. Tämä oli itselleni uutta ja tuntematonta aluetta. Unreal Enginen tukiessa läpinäkyvyyden renderöinti auttoi valitsemaan tietyn työskentelytekniikan. Tässä ne alueet, joita ei haluttu renderöitäväksi, kyettiin kytkemään materiaalista pois päältä.

Olen miettinyt, kuinka olisin voinut parantaa visuaalista ilmettä kyseisestä projektista. Päädyttyäni pieniin ja johdonmukaisiin päätelmiin, olen ollut melko tyytyväinen tuloksiin. Maanpinnan häivyttäminen materiaalitasolla olisi voinut auttaa saamaan aikaan pehmeämmän siirtymän kuvattuun videoaineistoon. Myös renderöinnin asetuksiin jouduttiin tekemään joitakin muutoksia, jotta pysyisin reaaliaikaisessa renderöinnissä. Nämä muutokset vaikuttivat jonkin verran visuaaliseen laatuun, mutta ne saatiin poistettua jälkieditointivaiheessa.

6.5 Metsien generoiminen

Olen pitkään pohtinut, kuinka metsää saataisiin helposti ja nopeasti visualisoitua, jotta voisin vähentää käsin tehtävää työtä. Suomen Ympäristökeskus tarjoaa avointa aineistoa, josta löytyi juuri sellaista dataa, jota voimme hyödyntää. Tämän aineiston käsittely vaatii vielä paljon manuaalista

työtä, koska en ole vielä päässyt miettimään, kuinka tätä aineistoa voisi hakea automaattisesti projektia luodessa. Samu Nykänen oli kohdannut samankaltaisia ongelmia, jossa hän pyrki istuttamaan kasvillisuuksia automaattisesti maanpintaan (Nykänen 2018).

Voisin luoda projektipohjaan työkalun, jolla määritetään haettavien aineistojen, TM35-karttalehtijakoon pohjautuva nimistö. Työkalu lataisi määrityksen jälkeen nimistöön perustuen Suomen Ympäristökeskuksen ja Maanmittauslaitoksen rajapinnoista tarvittavat aineistot. Täytyisi myös tutkia, voisiko korkeuspintamalliaineistoa käsitellä suoraan valitsemallani työkalulla. Näiden tutkimusten jälkeen voisi pohtia, onnistuuko työn automatisointi vai tulisiko etsiä sellaisia sovelluksia, jotka tarjoaisivat tällaista rajapintaa.

Maanpinnan luomisessa meillä on Rambollilla ollut jatkuvasti ongelmia, johtuen todennäköisesti siitä, etten täysin ymmärrä, miten Maanmittauslaitoksen aineisto tulisi oikeaoppisesti kääntää Unreal Engineen sopivaksi. Olen sitä varten kokeillut erilaisia työtapoja eri projektien välissä ja ottanut koko virtuaalisen maanpinnan luomisen omaksi osakseni. Tiimissäni olen ollut enemmän vastuussa kehittämisestä tekniikan osalta.

6.6 Milloin pääsemme eroon optimoinnista?

Naniten suurin heikkous lienee päällekkäisyyksien ja läpinäkyvyyden vaikuttaminen renderöintiin. Samasta ongelmasta kärsii tavallinenkin kolmiulotteinen renderöintikin (Sergeev, A. Verstraete, S. 2021). Ehkä Nanite päivittyy vielä niin, että päällekkäisyyksistä pääsemme eroon luomalla yhtenäisen pintamallin kokonaisesta virtuaalisesta maailmasta, josta ylimääräiset kolmiot on poistettu? Olemmeko jo siinä pisteessä, että virtualisoitu geometria tekniikkana syrjäyttää perinteisen geometria-aineiston?

Lähteet

Akenine-Möller, T & Haines, E. 2002. Real-Time Rendering, 2. painos. Natick, Massachusetts, Yhdysvallat: A K Peters

Akenine-Möller, T & Haines, E. Hoffman, N. ym. 2018. Real-Time Rendering, 4. painos. Boca Raton, Florida, Yhdysvallat: Taylor & Francis Group

Alba, M. 2023. GPU, Reinvented: Graphics, AI and the Astonishing New Era of Simulation. https://www.engineering.com/gpu-reinvented-graphics-ai-and-the-astonishing-new-era-of-simulation/. 3.10.2024.

AMD. 2020. AMD Unveils Next-Generation PC Gaming with AMD Radeon™ RX 6000 Series – Bringing Leadership 4K Resolution Performance to AAA Gaming. https://ir.amd.com/news-events/press-releases/detail/978/amd-unveils-next-generation-pc-gaming-with-amd-radeon-rx. 23.9.2024.

Connor, D. Nausha, M. Slocum, J. 2018. Team Dynamics and Video Game Performance. Guild Hall Video Game School. http://dx.doi.org/10.13140/RG.2.2.26144.02566. 23.9.2024.

Eccles, A. 2001. The Diamond Monster 3Dfx Voodoo 1. https://web.archive.org/web/20010105133000/http://www.gamespy.com/halloffame/october00/voodoo1/. 15.3.2024.

Epic Games. 2015. Game Programming in UE4. Game Framework & Sample Projects. https://www.slideshare.net/slideshow/east-coast-devcon-2014-game-programming-in-ue4-game-framework/47168780. 23.9.2024.

Epic Games. 2018. Epic Games demonstrates real-time ray tracing in Unreal Engine 4 with ILMxLAB and NVIDIA. https://www.unrealengine.com/en-US/blog/epic-games-demonstrates-real-time-ray-tracing-in-unreal-engine-4-with-ilmxlab-and-nvidia. 23.9.2024.

Epic Games. 2020. A First Look at Unreal Engine 5. https://www.unrealengine.com/en-US/blog/a-first-look-at-unreal-engine-5. 23.9.2024.

Epic Games. 2021. Nanite, A Deep Dive. https://advances.realtimerendering.com/s2021/Karis_Nanite_SIGGRAPH_Advances_2021_final.pdf. 23.9.2024.

Epic Games. 2022. Lumen, Real-time Global Illumination in Unreal Engine 5. https://advances.realtimerendering.com/s2022/SIGGRAPH2022-Advances-Lumen-Wright%20et%20al.pdf. 23.9.2024.

Epic Games. 2024a. Unreal Engine for Unity Developers. https://dev.epicgames.com/documentation/en-us/unreal-engine/unreal-engine-for-unity-developers?application_version=5.1. 23.9.2024.

Epic Games. 2024b. Navigation system in Unreal Engine.https://dev.epicgames.com/documentation/en-us/unreal-engine/navigation-system-in-unreal-engine?application_version=5.1. 23.9.2024.

Epic Games. 2024c. Sequencer Overview. https://dev.epicgames.com/documentation/en-us/unreal-

engine/unreal-engine-sequencer-movie-tool-overview?application_version=5.1. 23.9.2024.
Epic Games. 2024d. Rect Lights. https://dev.epicgames.com/documentation/en-us/unreal-engine/rect-lights?application_version=4.27. 23.9.2024.
Epic Games. 2024e. VR Performance Features. https://dev.epicgames.com/documentation/en-us/unreal-engine/vr-performance-features?application_version=4.27. 23.9.2024.
Epic Games. 2024f. Understanding Lightmapping in Unreal Engine. https://dev.epicgames.com/documentation/en-us/unreal-engine/understanding-lightmapping-in-unreal-engine. 23.9.2024.
Epic Games. 2024g. IES Light Profiles. https://dev.epicgames.com/documentation/en-us/unreal-engine/ies-light-profiles?application_version=4.27. 23.9.2024.
Epic Games. 2024h. Modifying the Navigation Mesh. https://dev.epicgames.com/documentation/en-us/unreal-engine/overview-of-how-to-modify-the-navigation-mesh-in-unreal-engine?application_version=5.3. 23.9.2024.
Epic Games. 2024i. Audio Files. https://dev.epicgames.com/documentation/en-us/unreal-engine/audio-files?application_version=4.27. 23.9.2024.
Epic Games. 2024j. Render Passes. https://dev.epicgames.com/documentation/fr-fr/unreal-engine/cinematic-render-passes-in-unreal-engine. 23.9.2024.
Epic Games. 2024k. Importing and Exporting Landscape Heightmaps. https://dev.epicgames.com/documentation/en-us/unreal-engine/importing-and-exporting-landscape-heightmaps-in-unreal-engine?application_version=5.3. 23.9.2024.
Epic Games. 2024l. Landscape Technical Guide. https://dev.epicgames.com/documentation/en-us/unreal-engine/landscape-technical-guide-in-unreal-engine?application_version=5.3. 23.9.2024.
Epic Games. 2024m. Nanite for Artists | GDC 2024. YouTube-video. https://www.youtube.com/watch?v=eoxYceDfKEM&t=438s. 23.9.2024.
Epic Games. 2024n. Lumen Technical Details. https://dev.epicgames.com/documentation/en-us/unreal-engine/lumen-technical-details-in-unreal-engine. 23.9.2024.
Epic Games. 2024o. Distance Field Soft Shadows. https://dev.epicgames.com/documentation/en-us/unreal-engine/distance-field-soft-shadows?application_version=4.27. 23.9.2024.
Epic Games. 2024p. Mesh Distance Fields. https://dev.epicgames.com/documentation/en-us/unreal-engine/mesh-distance-fields?application_version=4.27. 23.9.2024.
Foundry. 2024. UDIM Workflow. https://learn.foundry.com/modo/content/help/pages/uving/udim_workflow.html. 20.8.2024.
Heiniaho, K. 2022. Visualisoinnin ja havainnollistamisen merkitys, sekä arvonmääritys suunnitteluprosessissa. LUT-yliopisto. School of Engineering Science, Tuotantotalous. Diplomityö. https://urn.fi/URN:NBN:fi-fe2022051836476. 5.8.2024.

Luoshuang. 2024. Luoshuang's GPULightmass. https://forums.unrealengine.com/t/luoshuangs-gpulightmass/109474. 20.3.2024

Microsoft. 2018. Announcing Microsoft DirectX Raytracing! https://devblogs.microsoft.com/directx/announcing-microsoft-directx-raytracing/. 28.4.2024.

Mohammed, R. 2023. How to Build a Better Pricing Strategy. https://hbr.org/podcast/2023/05/how-to-build-a-better-pricing-strategy. 15.7.2024.

McCole, C. 2023. Blueprint Optimization In Unreal (UE4/UE5) https://www.chrismccole.com/blog/blueprint-optimization-in-unreal

Internet Assigned Numbers Authority. 2024. Media types. https://www.iana.org/assignments/media-types/media-types.xhtml. 8.8.2024

Nissinen, V. 2000. Sotilasjohtamisen tutkimuksesta ja koulutuksesta, Tiede ja ase, 58(58), s. 30–58. https://journal.fi/ta/article/view/47829. 16.8.2024.

NVIDIA. 2018. NVIDIA Turing GPU Architecture. https://images.nvidia.com/aem-dam/en-zz/Solutions/design-visualization/technologies/turing-architecture/NVIDIA-Turing-Architecture-Whitepaper.pdf. 28.8.2024.

NVIDIA. 2020. GTC Digital: Crafting a Real-Time Path-Tracer for Minecraft RTX. https://developer.nvidia.com/blog/gtc-digital-crafting-a-real-time-path-tracer-for-minecraft-rtx/. 23.9.2024.

NVIDIA. 2024. Moving Pictures: Transform Images Into 3D Scenes With NVIDIA Instant NeRF. https://blogs.nvidia.com/blog/ai-decoded-instant-nerf/. 23.9.2024.

Nykänen, S. 2018. VR-sovelluksen optimointi ja metsän visualisointi Unreal Engine 4 -pelimoottorilla. Karelia-ammattikorkeakoulu. Opinnäytetyö. https://urn.fi/URN:NBN:fi:amk-2018121020670. 4.10.2024.

McKenzie, T. 2022. A Realistic Scene Made With UE5's Nanite & Lumen in VR. 80 Level. https://80.lv/articles/a-realistic-scene-made-with-ue5-s-nanite-lumen-in-vr/. 6.8.2024.

Meta. 2024. Draw Call Cost Analysis for Meta Quest. https://developers.meta.com/horizon/documentation/unity/po-draw-call-analysis. 23.9.2024.

Mildenhall, B. Srinivasan, P.P. Tancik, M. Barron, J.T. Ramamoorthi, R. Ng, R. 2020. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis.

Polydin. 2024. Maximizing Performance: The Essentials of 3D Model Optimization https://polydin.com/3d-model-optimization/. 20.9.2024.

Rönnholm, P. 2023. Aalto-yliopisto https://web.archive.org/web/20240812182148/https://mycourses.aalto.fi/pluginfile.php/1938585/mod_resource/content/1/KIG-C1010_Luento7b_fotogrammetria_2023.pdf. 2.8.2024

Suomen ympäristökeskus. Corine maanpeite. https://ckan.ymparisto.fi/dataset/corine-maanpeite-2018. 20.7.2024

Sergeev, A. Verstraete, S. 2021. Discussing the Possibilities and Drawbacks of Unreal Engine 5's Nanite. 80 Level. https://80.lv/articles/discussing-

the-possibilities-and-drawbacks-of-unreal-engine-5-s-nanite/. 14.5.2024.
Unreal Engine. 2020. Unreal Engine 5 Revealed! | Next-Gen Real-Time Demo Running on Playstation 5. YouTube-video. https://www.youtube.com/watch?v=qC5KtatMcUw. 23.9.2024.
Zhang, C.G., Zha, D.H., Zhou, S., Zhou, H.X., Jiang, H.D. 2019. 3D visualization of landslide based on close-range photogrammetry. Instrumentation Mesure Métrologie. https://doi.org/10.18280/i2m.180507. 16.9.2024.
Zucconi, A. 2016. Volumetric Rendering: Signed Distanced Functions. https://www.alanzucconi.com/2016/07/01/signed-distance-functions/. 16.9.2024.