Niko Lauerma

Pelimallien teksturointitekniikat

Pelimallien tekstuurien toteuttaminen ja keinoja suorituskyvyn parantamiseksi

Metropolia Ammattikorkeakoulu

Muotoilija

3D-animointi ja -visualisointi

Opinnäytetyö

21.04.2025

Tiivistelmä

Tekijä(t):	Niko Lauerma
Otsikko:	Pelimallien teksturointitekniikat
Sivumäärä:	43 sivua + 1 liite
Aika:	21.04.2025
Tutkinto:	Muotoilija (AMK)
Tutkinto-ohjelma:	Muotoilun tutkinto-ohjelma
Pääaine:	3D-animointi ja -visualisointi
Ohjaaja:	Lehtori Kristian Simolin

Tämä opinnäytetyö käsittelee pelimoottoreissa käytettävien 3D-assettien teksturointiprosessia, keskittyen erityisesti työtehokkuutta sekä suorituskykyä parantaviin tuotantotapoihin.

Opinnäytetyön tueksi toteutettiin Unreal Engineen rakennettu pienoispeliympäristö, jonka tuotannossa hyödynnettiin opinnäytetyössä esiteltyjä työtapoja ja tekniikoita.

Asiasanat: 3D, teksturointi, optimointi, pelituotanto, peligrafiikka

---

Opinnäytetyön alkuperä on tarkastettu Turnitin Originality Check -ohjelmalla.

Abstract

Author(s):	Niko Lauerma
Title:	Techniques in game model texturing
Number of Pages:	43 pages + 1 appendix
Date:	21.04.2025
Degree:	Bachelor of Culture and Arts
Degree Programme:	Design
Major:	3D Animation and Visualisation
Instructor:	Senior Lecturer Kristian Simolin

This thesis discusses the texturing process of 3D assets used in game engines, focusing particularly on production methods that improve work efficiency and performance.

A miniature game environment built in Unreal Engine was created to support the thesis, utilizing the work methods and techniques presented in the thesis during its production.

Keywords: 3D, texturing, optimization, game development, game graphics

---

This thesis has been checked using Turnitin Originality Check service.

Sisällys

1 Johdanto

Teksturointi on 3D-tuotannon työvaihe, jossa määritellään 3D-mallin pinnan ominaisuudet. Näitä ominaisuuksia voi olla muun muassa väritys, karkeus tai läpinäkyvyys. Tämä opinnäytetyö painottuu videopelituotannon 3D-elementtien teksturointiin ja sellaisiin työtapoihin ja tekniikoihin, joilla voidaan parantaa sekä suorituskykyä, että työskentelytehokkuutta.

Teksturointi jakaantuu karkeasti kahdesta kolmeen työvaiheeseen. Ensimmäinen, vielä mallinnusvaiheessa tapahtuva preparoiva työvaihe on UV-kartoitus, jossa mallin pinta levitetään kaksiulotteiseen tasoon. Tälle tasolle levitettävien tekstuurikarttojen avulla on mahdollista toteuttaa mallille seuraavassa työvaiheessa valokuvista editoidut tai esimerkiksi käsin maalatut tekstuurit. Mikäli työskennellään käyttäen esimerkiksi trimsheet-tekniikoita, voidaan teksturointivaihe aloittaa jo mallinnuksen kanssa yhtä aikaa, ja mallintaa tekstuurikuvien ehdoilla. Viimeisessä työvaiheessa tekstuurit ja mallit yhdistetään pelimoottorissa muun muassa materiaaleja käyttäen.

Koska tekstuureilla on videopeleissä merkittävä vaikutus pelin suorituskykyyn valitusta päätelaitteesta riippumatta, pyritään niiden vaikutusta suorituskykyyn minimoimaan eri tekniikoin läpi tuotantoprosessin, alkaen hyvin suunnitellusta UV-kartoittamisesta jatkuen koko matkan muun muassa tekstuurikuvien pakkausmetodeihin ja pelimoottorin materiaalien instansoimiseen. Tässä opinnäytetyössä käydään tämä tuotantoketju läpi keskittyen samalla erilaisiin suorituskyvyn maksimoimisen kannalta tärkeisiin työtapoihin ja tekniikoihin.

Opinnäytetyön tueksi toteutettiin Unreal Engineen rakennettu peliympäristö, jonka työvaiheissa pyrittiin hyödyntämään mahdollisimman laaja-alaisesti eri metodeja mahdollisimman näyttävän ja realistisen lopputuloksen luomiseen samalla käyttäen yleisimpiä ja tehokkaimpia tekniikoita suorituskyvyn parantamiseksi.

2 Peliassettien teksturoinnin pohjatietoa

2.1 Sanasto

Assetti	Digitaalinen elementti videopelissä, joka voi olla esimerkiksi 3D-malli, tekstuurikartta tai videotiedosto. Tässä opinnäytetyössä asseteilla viitataan useimmiten teksturoituun 3D-malliin.
Leipominen	(Baking) Prosessi, jossa esilasketaan tekstuuri- tai valaisutietoa yleensä suorituskyvyn parantamiseksi yksityiskohtaisuutta uhraamatta.
Materiaali	3D-ohjelmassa tai pelimoottorissa objektin pintaominaisuudet määrittelevä sävyttimeen ja muun muassa tekstuurikarttoihin liittyvä elementti.
Noodi	(Node) Pelimoottoreissa noodit toimivat ohjelmointirakenteina, jotka edustavat tiettyjä toimintoja tai objekteja. Tällaisia voi olla esimerkiksi materiaalin hallinta tai pelimoottorin tapahtumaketjut.
Optimointi	Prosessi, jossa parannetaan suorituskykyä laaja-alaisesti eri tekniikoilla esimerkiksi pelimoottorissa, teksturoinnissa usein optimoimalla muistin riittävyyttä.
PBR	(Physically based rendering) Sävytys- ja renderöintitekniikka, jossa pyritään mahdollisimman realistiseen valon käyttäytymiseen materiaaleissa.
Sävytin	(Shader) Peligrafiikassa käytettävä erikoistunut ohjelma, joka laskee, miten grafiikkaa piirretään näytölle, kuten valojen, värien ja tekstuurien laskemista malleille ja ympäristöille.

Tekselitiheys	(Texel density) 3D-mallin pinta-alan ja sille allokoitavan tekstuurikartan pikselitarkkuuden suhde, esimerkiksi 256 pikseliä per neliömetri.
Tekstuurikartta	Teksturoidessa luotu kuvatiedosto, joka heijastetaan 3D-mallin pintaverkolle.
UV-kartoitus	3D-mallinnuksen prosessi, jossa mallin pinta saumoitetaan ja levitetään kaksiuloitteiselle tasolle teksturointia varten.

2.2 Mitä tarkoitetaan teksturoinnilla?

Teksturoinnilla tarkoitetaan 3D-tuotannossa työvaihetta, jossa kolmiulotteisen mallin pintaverkolle määritetään pintaominaisuudet, kuten värit ja materiaalit. Tavoitteena tässä prosessissa on lisätä mallin visuaalista näyttävyyttä, yksityiskohtaisuutta ja realistisuutta (ks. kuva 1). Valtaosa pelien tuotannossa käytettävistä 3D-malleista ovat tavalla tai toisella teksturoituja.

Kuva 1. Kuva mallinnetusta ympäristöstä sekä tekstuureitta, että niiden kanssa (Vaskevich i.a.).

3D-mallien teksturointi jakaantuu yleensä kahteen päävaiheeseen, joista toinen, preparoiva vaihe, on UV-kartoitus, ja toinen itse teksturointiprosessi. UV-kartoituksessa kolmiulotteisen mallin pintaverkko saumoitetaan ja avataan kaksiulotteiseen tasoon, joka mahdollistaa tekstuurikuvien heijastamisen niiden pinnalle (Villanueva, 2021).

Teksturointivaiheessa mallille tuotetaan itse pintaominaisuudet, joihin kuuluvat muun muassa värit, karkeus ja metallisuus (Goodarzi & Shahbazi 2024; Li, Watkins, Arevalo & Tovar, 2020, 89). Myös pinnan läpinäkyvyys sekä itsevalaisevuus eli emissiivisyys voidaan määritellä teksturoimalla.

Yleisimmät teksturointiprosessin tekniikat ovat digitaalinen maalaaminen sekä valokuvien käyttö, mutta teksturointia voidaan toteuttaa myös lukuisilla muilla tavoilla, ja tämä vaikuttaa huomattavasti työjärjestykseen ja -tekniikoihin (Goodarzi & Shahbazi 2024). Teksturointia voidaan tehdä lähes missä tahansa 3D-mallinnusohjelmassa. Siihen on myös erikoistuneita ohjelmistoja (Li ym. 2020, 92).

2.3 Hyvien työtapojen ja -tekniikoiden keskeinen tavoite

Pelituotannossa optimoinnilla tarkoitetaan sekä tuotannon aikana iteratiivisesti toteutettavaa, että tuotannon loppuvaiheessa tehtävää työtä, jolla pyritään maksimoimaan pelin tasalaatuinen suorituskyky halutuille kohdealustoille, uhraamatta kuitenkaan liikaa pelin toiminnallisuutta tai esimerkiksi visuaalista näyttävyyttä ja yksityiskohtaisuutta.

Tässä opinnäytetyössä keskitytään erityisesti sellaisiin UV-kartoituksen ja teksturoinnin työtapoihin ja tekniikoihin, joilla pyritään helpottamaan optimointiprosessia, vähentämään tekstuureiden negatiivista vaikutusta suorituskykyyn, sekä parantamaan työtehokkuutta tekstuureita luodessa.

Koska pelien ominaisuuksien sekä näyttävyyden suurimpana rajoitteena voidaan pitää päätelaitteen suorituskykyä, on tuotannossa tärkeää käyttää rajalli-

set resurssit mahdollisimman tehokkaasti. Tavoitteena onkin siis saada mahdollisimman sujuva ja näyttävä lopputulos teknisten rajoitteiden puitteissa. (Dickinson, 2017.) Hyvät työtavat mahdollistavat myös myöhemmässä tuotantovaiheessa resurssien uudelleenallokointia esimerkiksi uuden ominaisuuden implementoinnissa tuotantoon.

Suorituskyvyn parantamista edistävien tuotantotekniikoiden kanssa on tärkeää tunnistaa tilanteet, joissa tällaisesta ennakoivasta optimoinnista on todellista hyötyä. Tämä tarkoittaa tietämystä päätelaitteen rajoitteista sekä siitä, miten jonkin ominaisuuden implementointi vaikuttaa suorituskykyyn. (Dickinson, 2017.) Kautta tuotannon hyvin suunnitelluista ja toteutetuista työtavoista huolimatta optimointityö pelituotannossa painottuu usein tuotannon loppupuolelle, jolloin resurssien kulutusta voidaan analysoida konkreettisin mittaustuloksin.

Pelin grafiikoiden osalta tärkeimmät suorituskykyyn vaikuttavat laitteiston rajoittavat tekijät ovat prosessorin laskentateho, käytettävissä olevan muistin (RAM) määrä ja nopeus, mahdollisen näytönohjaimen laskentateho sekä sen oman dedikoidun muistin (VRAM) määrä ja nopeus. (Dickinson, 2017.)

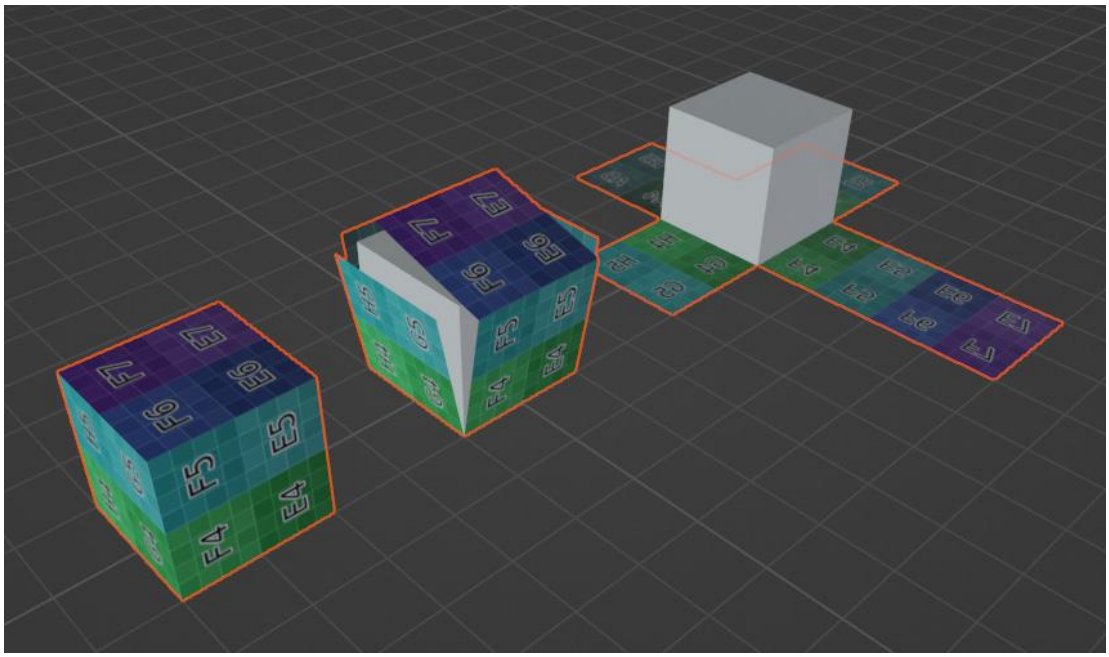
Tärkeitä, tunnettuja painopisteitä pelin 3D-assettien resurssienhallinnassa suorituskyvyn kannalta ovat mallin polygonaalinen monimutkaisuus, tekstuurikarttojen määrä ja koko sekä piirtokutsujen (eng. Drawcall) määrän minimointi. Näistä erityisesti kahta jälkimmäistä käsitelläänkin tässä opinnäytetyössä myöhemmin.

3 UV-kartoitus

3.1 Mitä on UV-kartoittaminen?

UV-kartoitus on 3D-mallintamisen työvaihe, jossa mallin pinta saumoitetaan ja levitään kaksiulotteiseen muotoon (ks. kuva 2). UV-kartta koostuu saarekkeista, jotka ovat saumoilla jaettuja mallin osia. Saarekkeet levitetään kaksiulotteiselle, neliön muotoiselle alueelle, joita kutsutaan vaihtelevasti UV-neliöiksi tai UV-laa-

toiksi. Tälle neliölle tai laatalle heijastetaan tekstuurikuva, joka puolestaan heijastetaan mallin polygoniverkon päälle. UV-kartoitus on tärkeä työvaihe mallinnuksessa, ja erityisesti merkittävä osa mallin valmistelua teksturointia varten. (Belec 2023, Villanueva 2021.)



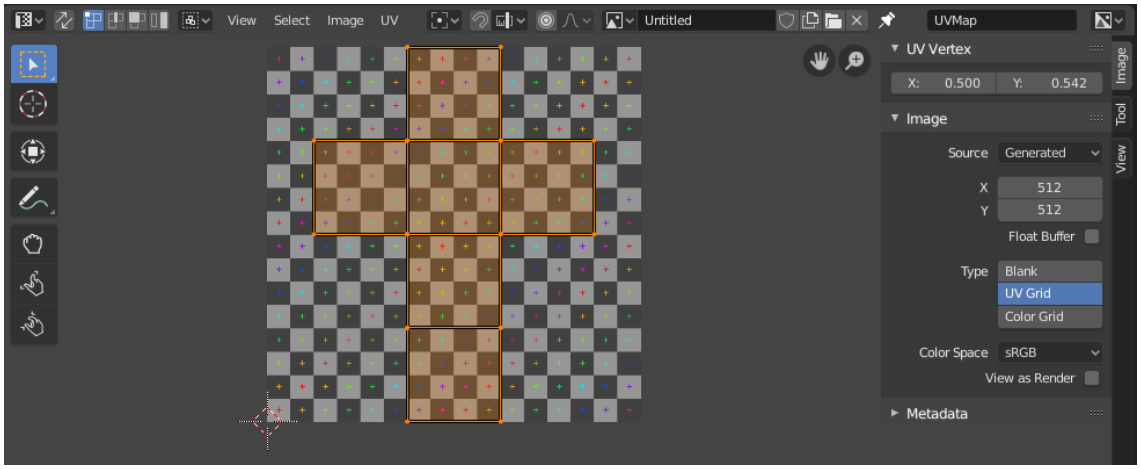
Kuva 2. Visualisoitu esimerkki 3D-mallin avaamisesta kaksiulotteisiin muotoon UV-kartoitusprosessissa (Belec 2023).

Joskus yksittäinen malli materiaaleineen voi käyttää myös montaa UV-laattaa. Tätä työtapaa kutsutaan UDIM-laatoitukseksi (Mendoza Guevarra 2020).

3.2 UV-saarekkeiden asettelu

Mallin voi saumoittaa joko automaattisesti tai määrittelemällä saumojen sijainti käsin. Yleisesti ottaen käsin saumoittaminen on huomattavasti tarkempaa ja varsinkin videopeleissä käytettävien mallien suorituskyvyn kannalta suotavaa (Villanueva 2021). Teksturoidessa saumakohdat ovat usein ongelmallisia, ja ne kannattaakin asetella mallissa mieluiten huomaamattomiin paikkoihin (Belec 2023).

Saumoittamisen jälkeen saumojen erottelemat saarekkeet levitetään kaksiulotteiseen muotoon. Nimitys UV-kartoittaminen tuleekin saarekkeiden sijaintia määrittävästä koordinaatistosta, joka jakaantuu U- ja V-akseleille. Saarekkeiden sijainti kartalla määrittelee, mikä osa tekstuureista heijastetaan mihinkin kohtaan mallia (ks. kuva 3). (Villanueva 2021.)



Kuva 3. UV-saari aseteltuna laatalleen. Kuva Blender-ohjelmiston UV-editorinäkymästä (Blender i.a.).

Saarekkeiden skaala puolestaan määrittelee, miten paljon UV-laatan tekstuurikuvan pinta-alaa allokoidaan suhteellisesti kyseiselle saarekkeelle. 3D-mallin pinta-alan ja siihen allokoitavan tekstuurikuvan pikselimäärän suhdetta kutsutaan tekselitiheydeksi (eng. Texel density). (Villanueva 2021.)

Joskus pinta-alaltaan erittäin suuret mallit eivät mahdu yhdelle laatalle ennaltamääritellyn tekselitiheyden puitteissa, mutta piirtokutsujen minimoimiseksi ei kuitenkaan haluta käyttää montaa materiaalia. Tällöin voidaan hyödyntää monesta laatasta koostuvaa UDIM-kartoitusta. (Mendoza Guevarra 2020.) On huomion arvoista, etteivät kaikki ohjelmat ja pelimoottorit suoraan tue UDIM-laatoista koostuvaa UV-karttaa.

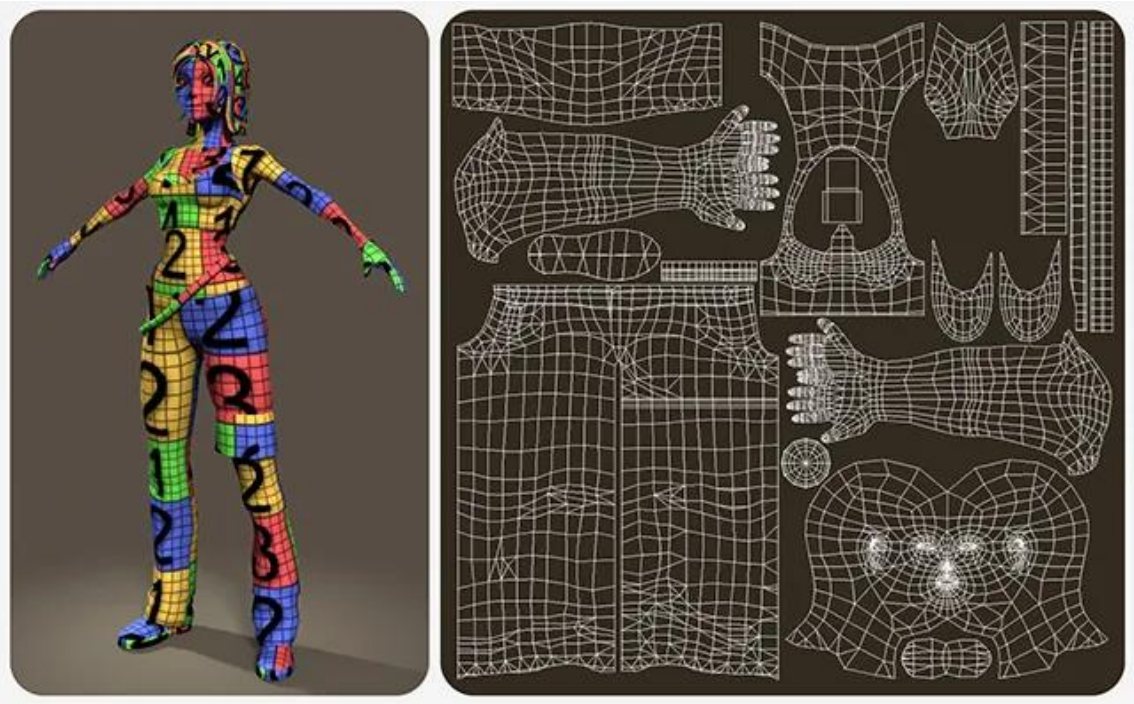
Visuaalisen eheyden takaamiseksi tulisi 3D-malli saumoittaa ja saarekkeet asetella UV-laatalle siten, etteivät saarekkeet venyisi liikaa. Tätä voi välttää sau-

moittamalla malli siten, ettei yhdellä saarekkeella ole liian monimutkaista polygoniverkkoa, esimerkiksi jakamalla alue useampaan saarekkeeseen. (Belec 2023.)

3.3 UV-karttojen yhteys suorituskykyyn

3D-mallissa käytettävien kuvakarttapohjaisten tekstuureiden koko ja määrä vaikuttavat suoraan suorituskykyyn. Tämän takia varsinkin videopeleihin tähdätyssä tuotannossa pyritään rajoittamaan tekstuurikarttojen kokoa ja materiaalien määrää, ja se heijastuu suoraan UV-kartoittamiseen.

Saarekkeiden lukumäärällä on suorituskykyvaikutus. Mikäli se on muun tuotannon kannalta mielekästä, kannattaa suosia mielummin suurempia UV-saarekkeita, kuin pilkkoa malli suureksi määräksi pieniä saarekkeita (ks. kuva 4). (Villanueva 2021.)



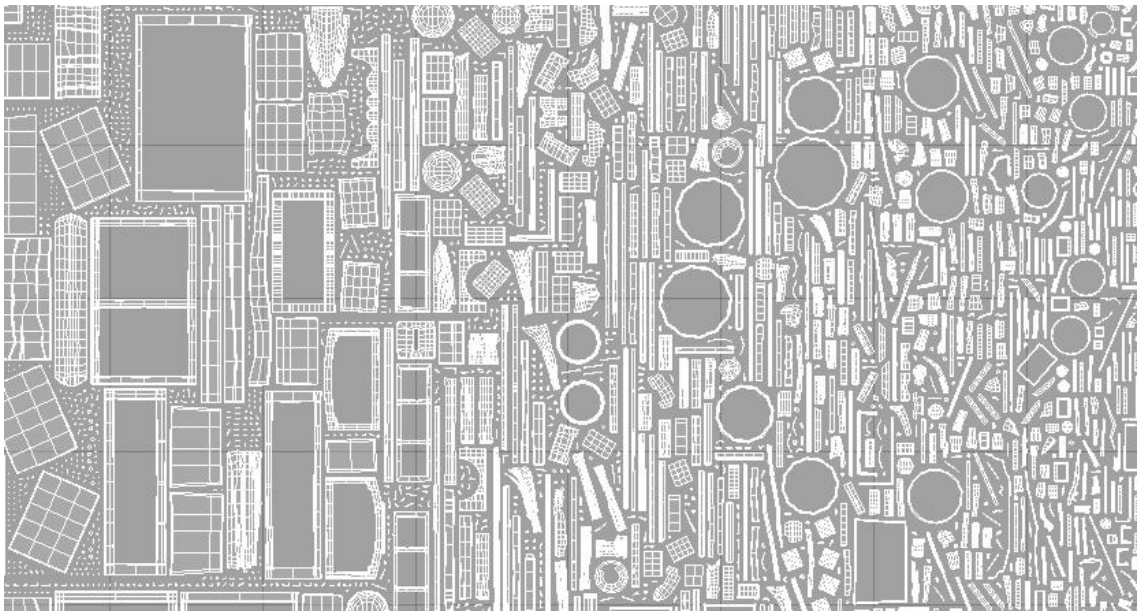
Kuva 4. Hahmomallin pintaverkko saumoitettuna ja levitettynä mahdollisimman suuriksi eheiksi saarekkeiksi (Dixon 2016).

Koska jokainen erillinen materiaali pelimoottorissa aiheuttaa piirtokutsuja, pyritään mallissa käyttämään niitä mahdollisimman vähän. Tällöin mallin tai ympäristöelementtien eri osat kannattaa UV-kartoitusvaiheessa pyrkiä sijoittelemaan samalle UV-laatalle, jolloin yksittäinen tekstuurikuva saadaan heijastettua koko mallikokonaisuudelle piirtokutsujen minimoimiseksi. Tällaista UV-karttaa käyttävää tekstuurikuvaa kutsutaan atlakseksi. (Watkins 2012; Li ym. 2020, 74.) Atlasointi on erittäin tehokas ja käytetty optimointikeino (Watkins 2021). Atlasoinnin hyöty korostuu mobiilipelituotannossa, jossa piirtokutsut ovat usein merkittävä pullonkaula suorituskyvyn suhteen.

Niinkutsuttu trimsheet -työskentely liittyy erottamattomasti atlasointiin ja sitä käytetään suorituskyvyn ja työskentelytehokkuuden parantamiseksi erityisesti ympäristöelementtien teksturoinnissa. Tätä tekniikkaa käyttäessä iteratiivinen, epälineaarinen työskentely UV-kartoitus-, mallinnus- ja teksturointiprosessissa on erityisen korostunutta.

Trim-tekstuurikartta luodaan ennakkoon, ja siinä käytetään yhdeltä UV-akseliltaan saumattomasti toistuvia tekstuureita. Tätä tekstuurikarttaa käytetään luomisen jälkeen itse mallin UV-kartoituksen pohjana, jolloin tarkalla saarekkeiden määrittelyllä ja asettelulla voidaan luoda sekä monimutkaisia, että huomattavan paljon saumattomuutta tekstuureissaan hyödyntäviä modulaarisia mallikokonaisuuksia käyttäen vain yhtä tekstuurikarttakuvaa ja materiaalia. (Burns 2023.) Uudelleenkäytettävyytensä takia trimsheet -karttojen hyödyntäminen vähentää merkittävästi siihen pohjautuvien elementtien suorituskykyvaikutusta, vähentäen moduulikokonaisuuden muistin käyttöä sekä piirtokutsujen määrää.

Aikaisemmin mainittujen rajoituksien vuoksi myös saarekkeiden pakkaaminen UV-laatalle tulisi toteuttaa siten, ettei laatalla olevan tekstuurikuvan tilaa menisi hukkaan (ks. kuva 5). Tarkalla pakkaamisella saadaan tekstuurikuvasta mahdollisimman paljon irti. (Villanueva 2021.) Tämän voi toteuttaa joko manuaalisesti asettamalla saarekkeet huolellisesti laatalle, tai hyödyntämällä automatiikkaan pohjautuvia pakkaustyökaluja ja niihin erikoistuvia lisäosia.



Kuva 5. UV-laatalle voidaan tarkalla pakkaamisella asetella huomattava määrä saarekkeita (Foundry i.a.).

Yleisesti ottaen yhtenäisen visuaalisen lopputuloksen takaamiseksi mallien pinta-alan ja sille allokoitavan tekstuurikartan pikselimäärän tulisi olla mallien välillä sama. Tätä suhdetta kutsutaan tekselitiheydeksi, ja se ilmoitetaan usein pikseli per neliömetri -suhteella. (Villanueva 2021.) Tekselitiheyden laskeminen manuaalisesti UV-kartoitusvaiheessa on usein haastavaa, ja onkin tehokkaampaa käyttää siihen tietokoneen laskentaan perustuvaa automatiikkaa. Tällaiset työkalut löytyvät suoraan mm. Autodeskin Maya ja 3ds Max -ohjelmistoista, ja Blenderiin nämä ominaisuudet saadaan jälkiasennettavilla lisäosilla.

Tekselitiheydestä kerrotaan tarkemmin opinnäytetyössä sitä erikseen käsittelevässä osiossa. Aihe mainittiin kuitenkin myös tässä yhteydessä, sillä tekselitiheyden määrittely tehdään UV-kartoitusvaiheessa, vaikka sen näkyvä vaikutus ilmenee vasta teksturoidessa.

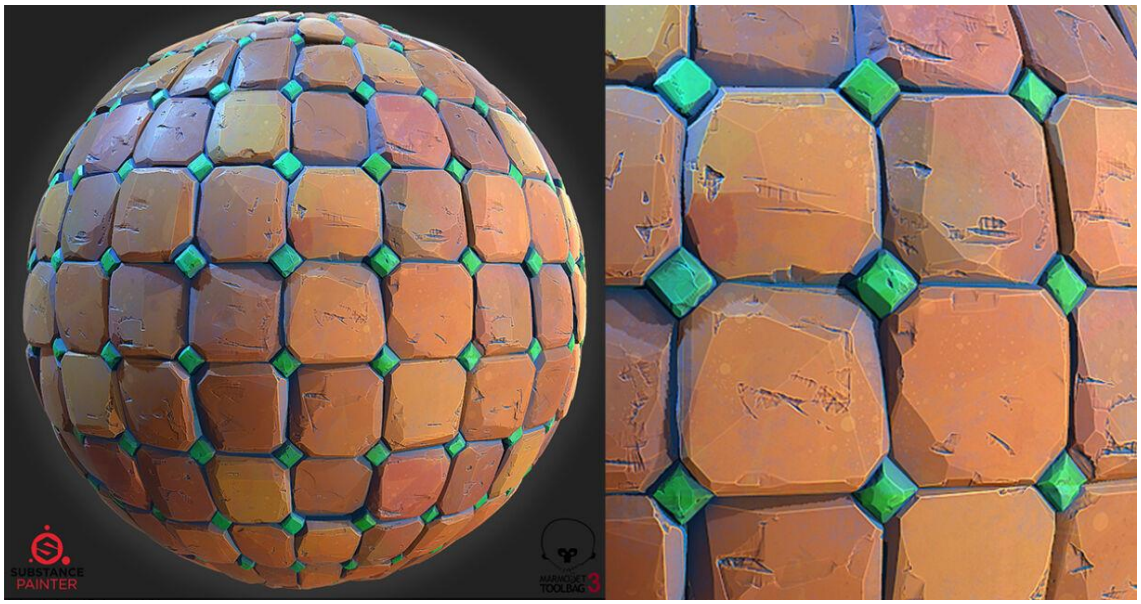
4 Teksturoinnin perusteet

4.1 Erilaiset menetelmät tekstuurien tuotannossa

Teksturointia voidaan toteuttaa lukuisilla eri tavoilla maalaamisesta valokuvaamiseen. Peliteollisuudessa tärkeimmät teksturointitekniikat lienevät digitaalinen maalaaminen, valokuvien käyttö ja fotogrammetria sekä proseduraalinen teksturointi (Goodarzi & Shahbazi 2024; Adobe i.a.). Artisti voi useissa teksturointiohjelmistoissa piirtää suoraan kolmiulotteisen mallin päälle hieman samaan tapaan kuin pienoismallia maalatessa. Useimmiten, käytetystä ohjelmistosta riippumatta, teksturoinnissa tuotetaan tekstuurikarttoja.

Osalla tekstuurikuvakartoista heijastetaan materiaalissa mallin pintaan itse värejä, ja osassa käytetään väridataa informaation välitykseen esimerkiksi maskaamiseen, tai pintageometrian muokkamiseen ohjaamalla valon heijastumista. Erilaisia tekstuurikarttatyyppejä käsitellään tässä opinnäytetyössä omassa osiossaan.

Digitaalisten, käsin maalattujen tekstuureiden suurin etu on vapaus tyyliteltyyn lopputulokseen, sekä hallittu, laaja muokattavuus ja korkea yksityiskohtaisuus (ks. kuva 6) (Adobe i.a.). UV-kartoituksen myötä saatuun 3D-mallin kaksiulotteiseksi avattuun pintaverkkoon tuotetaan kaksiulotteisia kuvia, joihin maalataan eri ohjelmistoja hyödyntäen värejä, erilaisia metallisuuksia, karkeuksia ja muita vastaavia ominaisuuksia. Digimaalaamista käytetään usein hahmoja teksturoidessa (Goodarzi & Shahbazi 2024).



Kuva 6. Esimerkki tyylitellystä teksturoinnista (Harrington 2019).

Maalaamisen lisäksi toinen suosittu tekniikka on toteuttaa valokuvien pohjalta, tai fotogrammetrian keinoin skannaamalla, esineelle kirjaimellisesti fotorealistiset tekstuurit, sillä valokuvia voidaan käyttää muokkaamisen jälkeen heijastaen valokuvattua aineistoa 3D-mallin pinnalle (ks. kuva 7) (Goodarzi & Shahbazi 2024; Adobe i.a.). Tämä tekniikka on suosittua fotorealistisiin peleihin tuotetussa sisällössä sekä sen autenttisen lopputuloksen, että teksturointiprosessin tehokkuuden takia. Koska valokuvaustilanteen valaistus ja varjot vaikuttavat fotogrammetrian keinoin tuotettuihin tekstuureihin, tulisi niihin kiinnittää sekä kuvaustilanteessa että tekstuurikuvia editoitaessa erityistä huomiota, ja pyrkiä minimoimaan ne. Fotogrammetriaa hyödynnetään usein tuote- ja arkkitehtuurivisualisoinnissa sekä luonnon elementtien, kuten kivien, teksturoimisessa (Goodarzi & Shahbazi 2024).

Kuva 7. Fotogrammetrian keinoin toteutettuja tekstuureita (Lau, 2018).

Kolmas peliteollisuudessa hyödynnetty teksturointitekniikka on proseduraalinen teksturointi (Goodarzi & Shahbazi 2024). Tällaisessa teksturoinnissa käytetään parametreihin ja algoritmeihin perustuvaa laskentaa tekstuurien luomisessa, ja niiden vahvuus on tekstuurien tarkkuuden käytännössä rajaton skaalautuvuus (Kumar, 2020). Proseduraalisilla teksturointitekniikoilla realististen tekstuurien luominen on käsin maalaamista tehokkaampaa (Adobe i.a.). Laskennallisen tehokkuuden takia noodipohjaisia proseduraalisia tekstuureita hyödynnetään paljon mobiilipeleissä, joissa päätelaitteen laskentateho tai esimerkiksi muistin määrä on rajallinen. Tyyliteltyä, niinkutsuttua käsin maalattua teksturointia on vaikeaa toteuttaa proseduraalisesti.

4.2 Yleisimmät teksturointiohjelmat

Useimmissa 3D-mallinnusohjelmissa, kuten Autodeskin ohjelmistoissa, Maxonin ZBrushissa sekä Blenderissä, on mukana teksturointityökaluja omasta takaa, mutta peliteollisuudessa käytetään tähän työvaiheeseen yleensä erikoistuneita teksturointiohjelmistoja.

Teksturointityökaluna Adoben nykyään omistama Substance Painter on peliteollisuuden standardisoitunut teksturointiohjelmisto (Kumar, 2020). Alun perin

ranskalaisen Allegorithmicin kehittämä ohjelmisto oli aikaisemmin mahdollista ostaa kertamaksulla, mutta Adoben omistukseen siirtymisen jälkeen on ohjelmisto siirtynyt progressiivisesti kohti tilausmallia, jolloin sen käyttäminen pohjautuu yhä useammin kuukausimaksuihin.

Substance Painterissa pystytään yhdistelemään käsin maalaamista ja proseduraalista työtapaa. Substance Painter toimii suorassa yhteydessä sisarohjelmansa, Substance Designerin kanssa. Jälkimmäisessä voidaan noodipohjaisella käyttöliittymällä rakentaa omia materiaaleja, joita sitten käytetään Painterissa itse teksturointiin. Substancen ohjelmistoista on muodostunut yksi suosituimmista teksturointityökaluista. (Kumar, 2020.) Substance Painterin vahvuuksiin kuuluu myös se, että maalausprosessissa voidaan maalata yhtäaikaisesti lukuisia eri kanavia, väri-, korkeus- ja karkeuskartoista emissiivisiin tai läpinäkyviin tekstuureihin.

Maininnan ansaitsee myös toinen Adoben ohjelma, Photoshop. Tämä digitaalisessa maalaamisessa ja valokuvaeditoinnissa valtavan suosittu ohjelmisto taipuu myös 3D-teksturointiin. Sen rooli peliteollisuudessa teksturointityökaluna on vähentynyt vuosi vuodelta. Digimaalaamistyökalujensa ansiosta Photoshop on kuitenkin edelleen suosittu työkalu vahvasti tyylitellyn, käsimaalatun teksturoinnin luomisessa. Adobe ajaa Photoshopin vanhempia 3D-teksturointityökaluja alas, ja painottaa nykyään sen integraatiota Substance Painterin kanssa.

Substancen ohjelmistojen kanssa pidemmän aikaa osittain samoista markkinoista kilpaillut 3DCoat on Substancen ohjelmistojen Adoben omistukseen siirtymisen mukanaan tuoman tilauspohjaisen maksumallin sijaan edelleen mahdollista ostaa lisenssin kertasijoituksella omaan omistukseen. 3DCoat eroaa em. kilpailijastaan myös sillä, että siinä on teksturointiominaisuuksien lisäksi jatkuvasti kasvava, kattava 3D-mallinnuspuoli mm. veisto- ja retopologiaominaisuuksineen.

Muita mainitsemisen arvoisia teksturointiohjelmia ovat muun muassa The Foundryn kehittämä, elokuvateollisuudessa suosittu Mari sekä uusi, kertamaksupohjaisella matalalla hinnoittelulla kilpaileva PBR-teksturoimiseen erikoistunut ArmorPaint (ks. Taulukko 1).

Taulukko 1. Peligrafiikan tuotannossa yleisimpiä teksturointiohjelmistoja vuoden 2024 hintoineen.

Kehittäjä	**Ohjelmisto**	**Hinta**
Adobe	Photoshop	27,41 €/kk
Adobe	Substance	50,19 €/kk
Pilgway	3DCoat	379 € (kertamaksu)
The Armory	ArmorPaint	19 € (kertamaksu)
The Foundry	Mari	78 €/kk

Yllä olevassa taulukossa (taulukko 1) on listattuna yleisimpiä peliassettien teksturointiin käytettyjä ohjelmistoja kehittäjineen sekä marraskuun 2024 hintoineen. On huomion arvoista, että yleisimmät teksturointiohjelmistot ovat siirtyneet enenevissä määrin kuukausimaksupohjaisuuteen kertamaksulla ostettavan pysyvän lisenssin sijaan.

4.3 Sävyttimet ja PBR

Sävyttimet ovat mm. pelimoottoreissa toimivia pieniä itsenäisiä näytönohjaimen laskennan varassa toimivia ohjelmia, joiden tehtävänä on määritellä miten pelissä esiintyvät objektit, valaisu ja muut visuaaliset efektit renderöidään näytölle (Ilett, 2022). Sävyttimillä on lukuisia tehtäviä ja ne ovatkin erottamaton osa pelimoottorin toimintaa, mutta tässä opinnäytetyössä niitä sivuutetaan vain niiden tärkeän roolin vuoksi teksturoinnissa, eikä ole lukijan kannalta tarkoituksenmukaista syventyä niihin pintaraapaisua enempää.

Sävyttimiä on lukuisia eri tyyppejä. Näistä muutamia esimerkkejä ovat verteksisävyttimet, jotka laskevat 3D-mallin verteksien lokaatioita kolmiulotteisesta

maailmasta kaksiulotteiseen muotoon ja voivat muokata muun muassa verteksin sijaintia tai väriä; Geometriasävyttimet, jotka verteksien sijaan muokkaavat kokonaisia polygoneja luoden esimerkiksi uusia pintoja mallissa; tai teksturoinnissa suuressa roolissa oleva fragmentti- eli pikselisävytin, joka laskee ruudulle piirrettävien pikselien värejä, ja jonka alaisuuteen kuuluu muun muassa tekstuurien ja valojen laskenta lukuisien muiden toimintojen lisäksi. (Ilett, 2022.)

Pikselisävyttimet määrittelevät pelimoottorissa sen, miten tekstuurikarttojen tuomaa dataa tulkitaan visuaalisen kokonaisuuden laskennassa. Esimerkiksi normaalikartat antavat sävyttimelle informaatiota siitä, miten valon tulee heijastua yksityiskohtaisistakin pinnanmuodoista. Näiden tekstuurikarttojen sijainti mallin pinnalla lasketaan sävyttimessä UV-karttatiedon perusteella.

Moderneissa, realismia tavoittelevissa peleissä käytetään nykyään usein pikselisävyttimessä laskettavaa PBR-sävytystä. Ilmaus tulee englannin kielen termistä Physically based rendering, fysiikkaan perustuva renderöinti, ja sen erikoisuus on realistinen valon heijastuminen, joka pohjautuu teksturoidun mallin pintamateriaaleihin (Ilett, 2022). PBR-sävytys mainitaan erikseen tässä opinnäytetyössä, sillä projektivaiheessa käytettiin PBR-tekstuureita sekä mahdollisimman realistisen lopputuloksen takaamiseksi, että siksi, että projektissa käytetty pelimoottori, Unreal Engine 5, käyttää PBR-sävytystä.

4.4 Tekstuurikartat

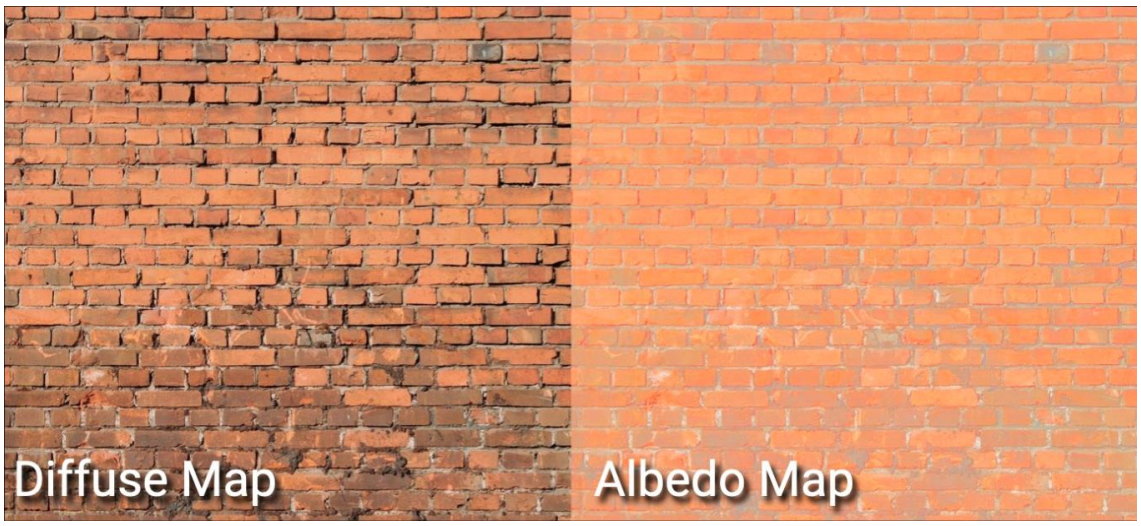
4.4.1 Mitä ovat tekstuurikartat?

Teksturointiprosessissa tekstuurien luomisen lisäksi oleennaista on, millaisia kuvakarttoja kyseisistä tekstuureista julkaistaan pelimoottorin käyttöön. Tekstuurikartat ovat kuvatiedostoja, jotka heijastetaan UV-kartoituksessa määritellysti 3D-mallin pinnalle. Tekstuurikuvakarttoja on lukuisia eri tyyppejä, joista peligrafiikan osalta tärkeimpiin paneudumme seuraavissa alaotsikoissa. Nämä kuvakartat voivat määritellä 3D-mallin pinnan värin, karkeuden, metallisuuden, valon heijastumiseen liittyviä ynnä muita ominaisuuksia.

4.4.2 Diffuusi- ja albedokartat sekä ympäristöokkluusio

Diffuusikartat edustavat tekstuurikarttatyyppiä, joka edustaa 3D-mallin värejä (ks. kuva 8). Tämä on yleisin käytettävä tekstuurikarttatyyppi. Diffuusikartta on kuvatiedosto, joka voidaan tuottaa lukuisilla eri tavoilla, kuten digitaalisesti maalaamalla, valokuvaamalla tai skannaamalla fotogrammetrian keinoin. Pohjavärien lisäksi diffuusikartta sisältää valo- ja varjoinformaatiota. (Kumar, 2020.)

Albedokartta on käytännössä riisuttu versio diffuusikartasta, sillä se ei sisällä informaatiota valoista tai varjoista, vaan pelkästään väriarvot (ks. kuva 8). (Kumar, 2020.) Albedokarttoja käytetään metallisuus-karkeusjärjestelmän PBR-materiaaleissa, joissa valon heijastuminen lasketaan sävyttimen toimesta pohjautuen karkeus- ja metallisuuskarttoihin. Albedokarttoja kutsutaan välillä myös pohjavärikartoiksi (eng. Base color map).



Kuva 8. Diffuusi- ja albedoversiot pohjavärikartasta (Lyons i.a.).

Albedokartta hyötyy varjo- ja valoinformaation puutteensa vuoksi ympäristöokkluusiokartan (eng. Ambient occlusion map) lisäämisestä pelimoottorin materiaaliin, joko itsenäisenä karttana, tai värikanavana osassa toista tekstuurikarttaa. Esimerkki tällaisesta kartasta on Unreal Enginen käyttämä ORM-kartta, jossa yhdistyy okkluusio-, karkeus- ja metallikartat yhdistettyinä, käyttäen kolmea eri värikanavaa eri kartoille, punaista, vihreää ja sinistä, samassa kuvassa.

4.4.3 Metallisuus-, karkeus- ja kiiltävyyskartat

Nämä kartat määrittelevät nimensä mukaisesti teksturoitavan mallin pinnan karkeuden, kiiltävyyden ja metallisuuden. Spekulaari- ja kiiltävyyskartat ilmentävät pinnan kiiltävyyttä ja heijastavuutta eri kuvakulmista (Kumar 2020).

PBR-sävyttimien kanssa työskennellessä käytetään yleisesti joko spekulaari-kiiltävyysjärjestelmää, tai metallisuus-karkeusjärjestelmää. Myös valittu pelimoottori määrittää, kumpaa järjestelmää käytetään – kahdesta suosituimmasta pelimoottorista Unity käyttää spekulaari- ja kiiltävyyskarttoja, Unreal Engine metallisuus- ja karkeuskarttoja PBR-laskennassaan (Kumar 2020).

Esimerkiksi metallisuuskartta määrittelee, onko pintamateriaali metallia vai ei, ja sävytin voi tämän tiedon perusteella laskea, miten valo heijastuu tai absorboituu kyseisestä pinnasta. Karkeuskartta määrittelee puolestaan pinnan karkeuden. Etuna PBR-materiaaleissa on fysiikan lakeihin pohjautuva laskenta sävyttimessä, joka johtaa erittäin realistiseen lopputulokseen.

4.4.4 Läpinäkyvyyskartat

Läpinäkyvyyskartta (eng. Opacity tai transparency map) määrittelee 3D-mallin läpinäkyvät alueet. Nämä kartat ovat mustavalkokuvia, joissa useimmiten valkoinen määrittelee läpinäkymätöntä, ja musta läpinäkyvää aluetta. (Kumar 2020.)

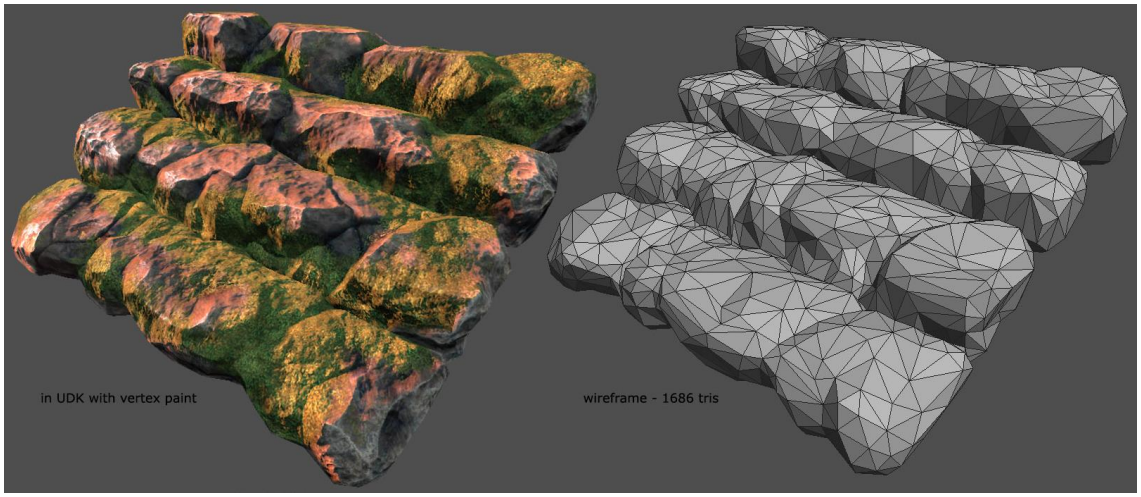
Läpinäkyvyyttä hyödynnetään pelimoottoreissa monella tavalla, joista mainittakoon tässä yhteydessä kaksi esimerkkiä, ensimmäisenä alpha clipping, josta käytetään myös termiä alpha testing. Tässä tekstuurityypissä mustavalkokuvalla määritellään tekstuurissa läpinäkyvyys kokonaisuudessaan, ja 3D-pinta voi olla halutuissa kohdissa kauttaaltaan läpinäkymätön tai läpinäkyvä (PCMag i.a.). Esimerkki tällaisesta tekstuurista voi olla repaleinen kangas, jossa reiät ilmennetään tekstuurikartan avulla 3D-tasossa, jossa on kuitenkin yhtenäinen pintaverkko.

Toisessa yleisessä läpinäkyvyystyypissa, alpha blendingissä, materiaali voi olla myös osittain läpinäkyvä. Tällainen materiaali voi olla esimerkiksi samea lasi. Läpinäkyvyyskartta sisältää tällaisessa kartassa myös harmaan sävyjä, joka määrittelee, miten läpinäkyvä tietty kohta tekstuuria on (PCMag i.a.). Alpha blending on tekniikkana pelimoottorissa laskentatehon kannalta usein huomattavan raskas, ja sen liiallista käyttöä tuleekin varoa.

4.4.5 Normaalikartat

Normaalikartat ovat hieman aikaisemmista poikkeava tekstuurikarttatyyppi, sillä ne eivät suoranaisesti määrittele pintamateraalin ominaisuuksia, vaan luovat siihen illuusion korkeuseroista, joita käytetään yksityiskohtaisuuden lisäämiseen.

Normaalikarttoja luodaan usein esilaskentaprosessissa, jota kutsutaan leipomiseksi (eng. Baking). Tässä prosessissa monimutkaisemmasta, korkeatiheyksisestä mallista tuotetaan kaksiulotteinen kuva, joka projektoidaan matalatiheyksisen mallin pinnalle. (Kumar 2020.) Tämä leivottu tekstuurikartta käyttää väridataa ohjatakseen pelimoottorin sävyttimiä heijastamaan valoa imitoiden monimutkaista pintarakennetta (ks. kuva 9). Normaalikartat eivät voi vaikuttaa esineen siluettiin.



Kuva 9. Normaalikartan avulla saavutettu huomattavasti 3D-mallin pohjaverkkoa yksityiskohtaisempi pinnanmuotojen toteutus (Chadwick i.a.).

Joskus leipomisprosessissa saattaa muodostua epätoivottuja häiriöitä pinnanmuotojen laskennassa. Tämä voi johtua esimerkiksi puuttuvista osista korkea- ja matalapolygonisen mallien välillä. (Villanueva, 2021.) Näitä häiriöitä kutsutaan usein artifakteiksi. Liian suuret eroavaisuudet kahden eri tasoisen mallin geometriassa välillä ovat toinen yleinen syy artifaktien muodostumiseen. Leipomisprosessin ja normaalikartan muodostumisen jälkeen kannattaakin käydä mallin tekstuurit tarkoin läpi.

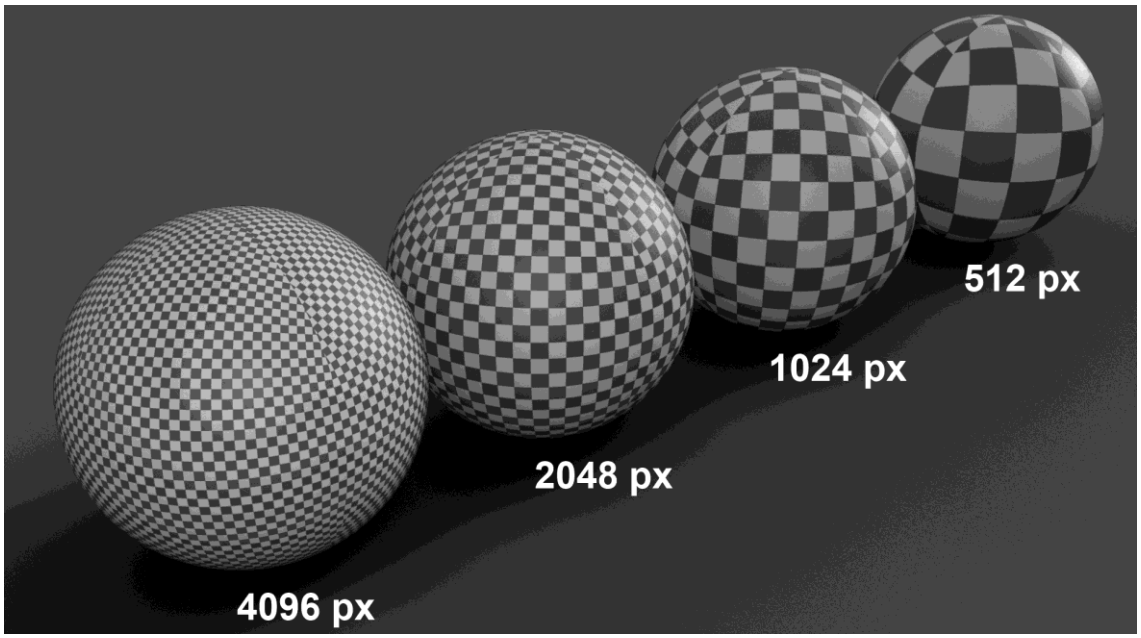
Normaalikarttojen käyttäminen pienten pinnanmuotojen luomiseen on suorituskyvyn kannalta erittäin kannattavaa, ja nykyisin käytännössä erottamaton osa korkealaatuisen peligrafiikan tuotantoa.

5 Tekstuurien suorituskykyyn vaikuttavia seikkoja

5.1 Tekselitiheys ja sen rooli tuotannossa

Sana tekselitiheys on lyhennys sanasta tekstuurielementtitiheys (eng. Texture element density). Tekselitiheys kertoo mallin pinnan tekstuureiden tarkkuuden pikselimäärästä suhteutettuna pinta-alaan, ja sitä ilmaistaan usein pikseleinä per neliö- tai senttimetri (ks. kuva 10). Teksturoidessa on yleisesti ottaen tarkoituksenmukaista pitää lähes sama tekselitiheys läpi mallin, sillä katsojan on helppo huomata suuret eroavaisuudet tekstuurien tarkkuudessa. (Villanueva, 2021.)

Tekselitiheydellä ja tekstuurikuviin vaadittavalla resoluutiolla on suora yhteys toisiinsa. Koska näiden kuvien resoluutio puolestaan on merkittävä tekijä muisti- ja kiintolevytilakäytön suhteen, sekä vaikutukseltaan latausaikoihin, tulisi peliassettien tekstuureita luodessa kiinnittää erityistä huomiota tekselitiheyteen visuaalisen tasalaatuisuuden lisäksi myös suorituskykyä parantaakseen.



Kuva 10. Sama malli neljällä eri tekselitiheydellä, ruudukkotekstuurilla havainnollistettuna (Radivojevic 2023).

Opinnäytetyössä aikaisemmin mainitun atlasoinnin funktio usein onkin maksimoida tekselitiheys samalla pitäen tekstuurikuvien resoluutio mahdollisimman matalana suorituskyvyn parantamista ja päätelaitteen rajoitteissa pysymistä varten piirtokutsujen minimoimisen lisäksi.

Peligrafiikan suunnittelussa valittu tekselitiheys riippuu usein pelissä käytettävästä perspektiivistä. Niinkutsutuissa ensimmäisen persoonan peleissä, joissa kamerakuvakulma on pelattavan hahmon silmistä, tarvitaan korkeita tekselitiheyksiä, sillä ympäristöä voidaan katsella todella läheltä. Jos peli on puolestaan kuvattu lintuperspektiivistä, voidaan käyttää matalampia tekselitiheyksiä. (Li ym. 2020, 72.)

Peliassettien tekstuureita optimoitaessa kannattaa käyttää korkeita tekselitiheyksiä vain, jos elementtejä pystyy katsomaan läheltä. Maisemassa kaukana esiintyviin objekteihin ei ole syytä uhrata korkean tekselitiheyden vaatimia suuria tekstuurikarttoja. Yhdessä peliympäristössä hyödynnetäänkin tarvittaessa useampaa tekselitiheyden tasoa, jolloin esimerkiksi maisemassa etualalla ole-

vissa objekteissa tekselitiheys on korkea, mutta kauempana horisontissa olevissa suuremmissa mallikokonaisuuksissa, kuten taustamaiseman vuoristossa matalampi (ks. kuva 11). (Li ym. 2020, 72.)

Kuva 11. Tekselitiheyden vaihtelua ympäristössä. Pelattavalla alueella ympäristöä käytetään korkeampaa tekselitiheyttä, kuin taka-alalle jäävässä taustamaisemassa (Dries 2023).

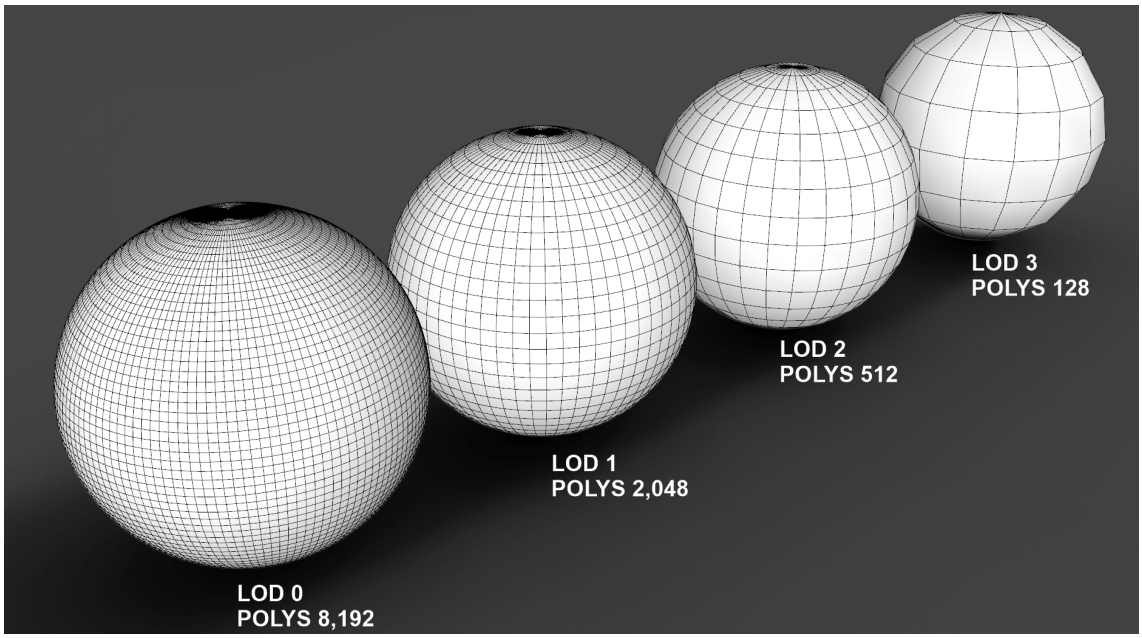
Joissakin tilanteissa on kannattavaa vaihdella myös mallien sisäistä tekselitiheyttä. Mikäli mallissa on alueita tai osia, jotka jäävät todennäköisesti matalalle huomiolle tai useimmiten kokonaan näkymättömiin, voi niiltä alueilta allokoida tekstuuritilaa toisiin, visuaalisen kokonaisuuden kannalta tärkeämpiin alueisiin. Esimerkkinä tästä sisätilaympäristöön tulevan kirjahyllymallin takaseinän pintaan, joka jää useimmiten piiloon hyllyn ja seinän väliin, ei kannata käyttää tekstuuripinta-alaa samassa suhteessa, kuin hyllyn etupuoleen. Mikäli puolestaan yksityiskohtaisuutta on tarpeen lisätä, kuten esimerkiksi usein pelihahmojen kasvoja teksturoidessa, voidaan kyseiselle alueelle allokoida muuta mallia korkeampi tekselitiheys.

Tekselitiheyksien valinta ja niissä pysymisen tärkeys korostuu suuremmissa tuotannoissa ja studioissa, sillä sen rooli on suuri tekstuurien tarkkuuden yhtenäistämisen tuomassa tasalaatuisuudessa peliassettien suhteen. Yleensä suurissa tuotannoissa tekselitiheydet lukitaan jo tuotannon alkuvaiheissa.

5.2 Yksityiskohtaisuustasot ja mipmappaus

Yksityiskohtaisuustaso (eng. Level of detail) tai ”loditus”, tarkoittaa pelimoottoreissa terminä tekniikkaa, jossa käytössä olevista asseteista, kuten malleista tai tekstuureista, luodaan monta laadultaan eri tasoista iteraatiota. Näitä tarkkuudeltaan eri tasoisia versioita vaihdetaan yleensä pelimoottorin toimesta automatisoidusti ja progressiivisesti riippuen siitä, miltä etäisyydeltä kyseistä assettia renderöidään. (Villanueva, 2021.)

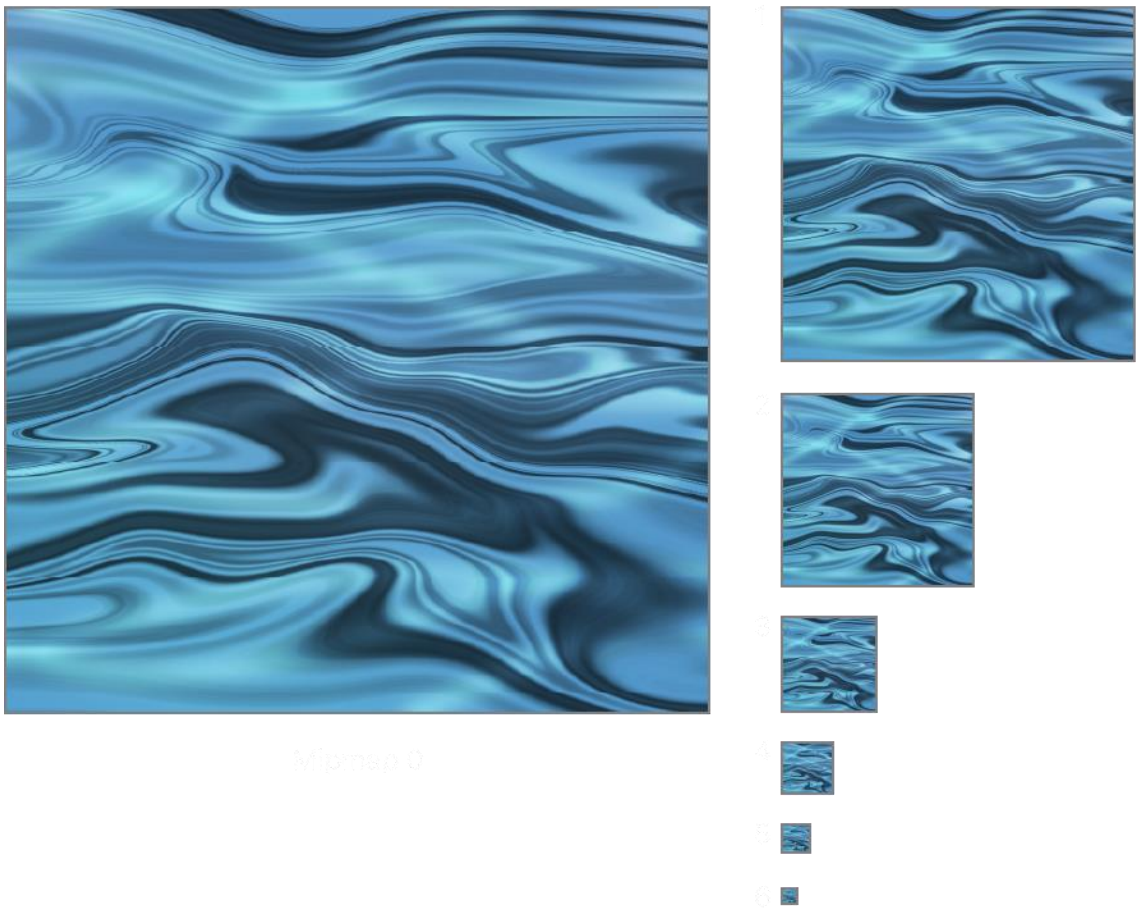
Mallien suhteen tällä tarkoitetaan yleensä sitä, että kaukana näkyvästä mallista näytetään desimoitua versiota, joissa polygonimäärä puolitetaan tai jaetaan neljällä. Yhdestä mallista saattaakin olla lukuisia loditettuja versioita riippuen mallin lähtökohtaisesta monimutkaisuudesta (ks. kuva 12).



Kuva 12. Neljä eri tasoista LOD-versiota samasta mallista polygonimäärineen (Radivojevic 2023).

Tekstuurikarttojen yksityiskohtaisuustasoituksessa puolestaan madalletaan kuvien resoluutiota renderöintietäisyyden kasvaessa. Tekstuurikarttojen yhteydessä tätä prosessia kutsutaan yleensä mipmappaamiseksi (eng. Mipmapping). Tällä tarkoitetaan tekstuurikartan resoluution puolittamista, ja näitä puolitettuja

tekstuuriversioita voidaan myös puolittaa entisestään. Tällaista kuvasarjaa kutsutaan mipmap-ketjuksi (ks. kuva 13). (Anuworakarn, 2019.) Käytössä oleva ohjelmisto, kuten pelimoottori, valitsee ketjusta sopivimman mipmapin riippuen siitä, miltä etäisyydeltä kyseessä olevaa teksturoitua objektia milloinkin renderöidään (Mendoza Guevarra 2020). Muun muassa Unreal Engine mipmappaa projektiin lisättävät tekstuurikuvakartat automaattisesti, mikäli artisti on toteuttanut ne power-of-two-resoluutiopolitiikan puitteissa (Epic Games i.a.d).



Kuva 13. Mipmap-ketju, jossa sama tekstuurikuva yhdellä kuvakartalla lukuisina eri tasoisina versioina (Apple i.a.).

Artistin kannattaa hyödyntää mipmappausta optimoinnin hyötyjen lisäksi siksi, että sillä on positiivinen vaikutus siihen, miltä kaukana olevat tekstuurit näyttävät. Mikäli korkearesoluutioista tekstuurikarttaa piirretään liian kaukaa, saattaa siihen ilmestyä aliasoitumista tai artifakteja, ja esilaskemalla matalampia mipmap-versioita tekstuureista tätä voidaan ennaltaehkäistä. (Anuworakarn, 2019.)

5.3 Kuvaformaatit ja bittisyvyys

Tekstuurikartat ovat kuvatiedostoja, joihin pätee yleisesti ottaen samat lainalaisuudet kuin kaikkiin kuvatiedostotyyppeihin. Koska resoluution lisäksi formaatin myötä bittisyvyys vaikuttaa kuvatiedoston kokoon, ja siten peliassetteja renderöidessä näytönohjaimen ja muistin resurssien käyttöön, on tärkeää tiedostaa eri formaattien ja pakkausmetodien haitat ja hyödyt. Tekstuurikartat edustavat myös merkittävää osaa pelin kiintolevyltä vaatimasta tilasta (Mower i.a).

Pelimoottorien pakkausalgoritmit painottavat suorituskyvyn maksimointia reaaliaikasesti renderöivässä ohjelmassa. Tämän takia pelimoottorien käyttämät pakkausmetodit eivät yleisesti ottaen ole häviöttömiä. Pakkauksessa tavoitellaankin kompromissia laadun ja suorituskyvyn välillä. (Mower i.a.)

Päätavoite sopivan bittisyvyyden ja oikeanlaisen kompressiometodin valinnassa on saavuttaa mahdollisimman korkea suorituskyky näytönohjaimen resurssien ja muistinkäytön suhteen uhraamatta kuitenkaan liikaa kuvanlaatua, välttäen kompressioartifakteja ja värien juovittumista (eng. Color banding).

Esimerkiksi normaalikarttoja pakatessa kannattaa käyttää korkeampaa bittisyvyyttä artifaktien minimoimiseksi, kuin pakatessa diffuusi- ja albedokarttoja. Bittisyvyydellä ja kompressiolla on merkittävä rooli normaalikarttojen näkymisessä oikein pelimoottorissa (Lampel 2017).

Yleisimmät pelimoottorit, kuten Unreal Engine ja Unity, pyrkivät automaattisesti kuvakarttoja lisätessä tunnistamaan kuvatyypit ja asettamaan niille optimaalisimmat pakkausasetukset. Pelimoottoreista löytyy vaihtoehtoisia esiasetuksia kompressiometodiksi erityyppisille tekstuurikartoille (Mower i.a).

Teksturointiohjelmasta kuvakarttoja julkaistaessa artistin kannattaa painottaa kuvanlaatua, sillä pelimoottorin omat kompressiometodit pääsevät silloin työskentelemään mahdollisimman korkealaatuisen lähdeaineiston pohjalta. Tästä huolimatta kannattaa tekstuurikuvia julkaistaessa käyttää teksturointiohjelmiston omia, tiettyyn pelimoottoriin tähdättyjä julkaisuesiasetuksia, mikäli tällaisia on

saatavilla. Tämä johtuu muun muassa siitä, että pelimoottorien tukemat kuvaformaatit, kuvakoot ja tekstuurikarttakuvatyypit ovat rajallisia. Esimerkiksi Unreal Engine tukee vain niinkutsutun power of two -koon kuvia, mikäli halutaan hyödyntää mipmappaamista (Epic Games i.a.d). Julkaisuesiasetuksien käyttö parantaa ja helpottaa merkittävästi artistin julkaisemien kuvakarttojen yhteensopivuutta kohteena olevassa pelimoottorissa.

5.4 Materiaalit ja niiden instansointi

Materiaaleilla tarkoitetaan pelimoottoreissa grafiikkaan liittyvää elementtiä, joissa luodaan tavalla tai toisella esimerkiksi 3D-mallin pintaominaisuudet (Epic Games i.a.a; Unity i.a.a). Sävyttimet ja materiaalit toimivat läheisessä yhteistyössä keskenään (Unity i.a.a). Useampi malli voi käyttää samaa materiaalia, ja tällä on positiivinen vaikutus suorituskykyyn, erityisesti käytettäessä materiaali-instansointia.

Materiaaleja muokataan siihen tarkoitetussa materiaalieditorinäkymässä, jossa työskennellään pääasiassa noodeilla. Tällä tavoin artistit voivat työskennellä sävyttimien parissa pelimoottorissa käyttäen koodaamisen sijaan visuaalista käyttöliittymää (Epic Games i.a.a).

Materiaalieditorissa määritellään miten valo reagoi 3D-mallin pintoihin määrittelemällä materiaalille ominaisuudet, joita voivat muun muassa väritys, pinnan heijastavuus ja karkeus, tai vaikkapa sen läpinäkyvyyden ominaisuudet (Epic Games i.a.a).

Materiaaleissa käytetään usein tekstuurikuvia, jotka yhdistetään noodeilla haluttuihin kanaviin itse materiaalissa (Epic Games i.a.d). Tekstuurikuvat tuotetaan yleensä teksturointiohjelmissa, ja niitä julkaistaessa tulee olla tarkkana, millaisessa muodossa ja formaatissa ne kannattaa tuoda pelimoottoriin materiaalin käytettäväksi, jotta teksturointiohjelmassa saavutettu tyyli siirtyisi pelimoottoriin muuttumattomana ja hyvälaatuisena.

Tekstuurikuvat eivät kuitenkaan ole pakollisia materiaaleja luodessa, ja niitä voidaan toteuttaa pelimoottorissa niiden sijaan myös pelkästään noodeilla, proseduraalisuutta ja parametrejä hyödyntämällä. Vastavuoroisesti tekstuurikuvia voidaan erikoistilanteissa käyttää myös ilman materiaalia, kuten luotaessa ruudulla näkyvää HUD-käyttöliittymägrafiikkaa (Epic Games i.a.d).

Joissakin pelimoottoreissa voidaan myös instansoida materiaaleja. Unreal Enginessä näitä kutsutaan materiaali-instansseiksi. Unityssä käytetään termiä materiaalivariantti (Unity i.a.b). Instansoidessa alkuperäisestä materiaalista luodaan kopio, jolle periytyy dynaamisesti niinkutsutun parent-materiaalin ominaisuudet (Epic Games i.a.b). Materiaali-instanssissa jotkut materiaalin ominaisuudet tai noodit parametrisoidaan, jotta sen ominaisuuksia voidaan muokata suhteessa parent-materiaaliin (Epic Games i.a.b).

Instansointi parantaa työtehokkuutta, sillä useampi artisti voi työskennellä omien materiaali-instanssiensa parissa niitä kustoimoiden, ja alkuperäinen parent-materiaali säilyy entisellään. Parenttiin tehtävät muutokset puolestaan periytyvät automaattisesti instansseille. Tämä mahdollistaa tehokkaan päällekäisen sekä iteratiivisen työskentelyn projektissa. (Epic Games i.a.b.)

Hieman käytössä olevan pelimoottorin ominaisuuksien mukaan, kannattaa materiaali-instasseilla työskentelyä priorisoida myös suorituskyvyn parantamiseksi. Unreal Enginessä materiaali-instanssin ominaisuudet voivat vaihdella pelin aikana, kun itse parent-materiaalin puolestaan ei, sillä se lasketaan eli kompiloidaan etukäteen ennen itse pelaamista (Epic Games i.a.c). Tämän lisäksi osa pelimoottoreista, kuten Unreal Engine, pystyvät syöttämään materiaali-instanssit yhdistettyinä kokonaisuuksina, niinkutsuttuina batcheina, päätelaitteiston laskettavaksi hyödyntäen batchingiksi kutsuttua metodia, joka on merkittävä piirtokutsuja vähentävä optimointikeino.

6 Projekti: Saaridiorama

6.1 Tavoitteet

Projektin tavoitteena oli läpikäydä peliympäristön grafiikan suunnittelun työjärjestys erityisesti teksturointitekniikoihin ja erilaisiin tekstuureiden suorituskykyä parantavien työtapoihin keskittyen. Aiheeksi valikoitui pieni saaridiorama, jossa esiintyy luonnonmuotoja, rakennuksia, proppeja sekä kasveja (ks. kuva 14). Tällä tavoin tuotantoon saatiin läpileikkuu ympäristön luomisen ja erityisesti teksturoinnin tekniikoista peliympäristötuotannossa sekä optimoinnista pitäen kuitenkin työ mahdollisena toteuttaa aikarajoitteissaan.

Kuva 14. Saaridiorama Unreal Enginessä renderöitynä lopullisessa muodossaan (Lauerma 2024).

Ympäristön luomisprosessissa otettiin alusta asti huomioon ympäristösuunnittelun ennakoivan optimoinnin kulmakivet, eli polygonaalinen monimutkaisuus ja polygonien määrän tavoitteellinen minimointi; tekstuurikarttojen mahdollisimman pieni koko ja monipuolisen uudelleenkäytettävyyden painottaminen; piirtokutsu-

jen minimointi elementtejä ja tekstuurikarttoja yhdistelemällä ja materiaaleja instansoimalla, sekä valaisun että partikkeliefektien vaikutus ympäristön reaaliaikaiseen suorituskykyyn. Kaikkien näiden tekniikoiden tavoitteena on prosessorin sekä näytönohjaimen laskennallisen kuormituksen vähentäminen, sekä erityisesti näytönohjaimen muistinkäytön minimoiminen.

6.2 Suunnittelu ja konseptointi

Projekti aloitettiin ideointi- ja suunnitteluvaiheesta. Tarkka suunnittelutyö helpotti työskentelyä sekä optimointia huomattavasti myöhemmissä työvaiheissa. Rakennusten modulaarisuuteen kiinnitettiin erityistä huomiota, ja moduulien mitat laskettiin tarkkaan ennakkoon yhteensopivuuden takaamiseksi. Tässä työvaiheessa päätettiin, mitkä mallikokonaisuudet jakavat keskenään samat materiaalit ja tekstuuriatlakset. Assettien nimeäminen ja organisoiminen ennakkoon todettiin myös tärkeäksi, jotta monimutkaisen ja -osaisen kokonaisuuden projektinhallinta pysyisi mahdollisimman kevyenä.

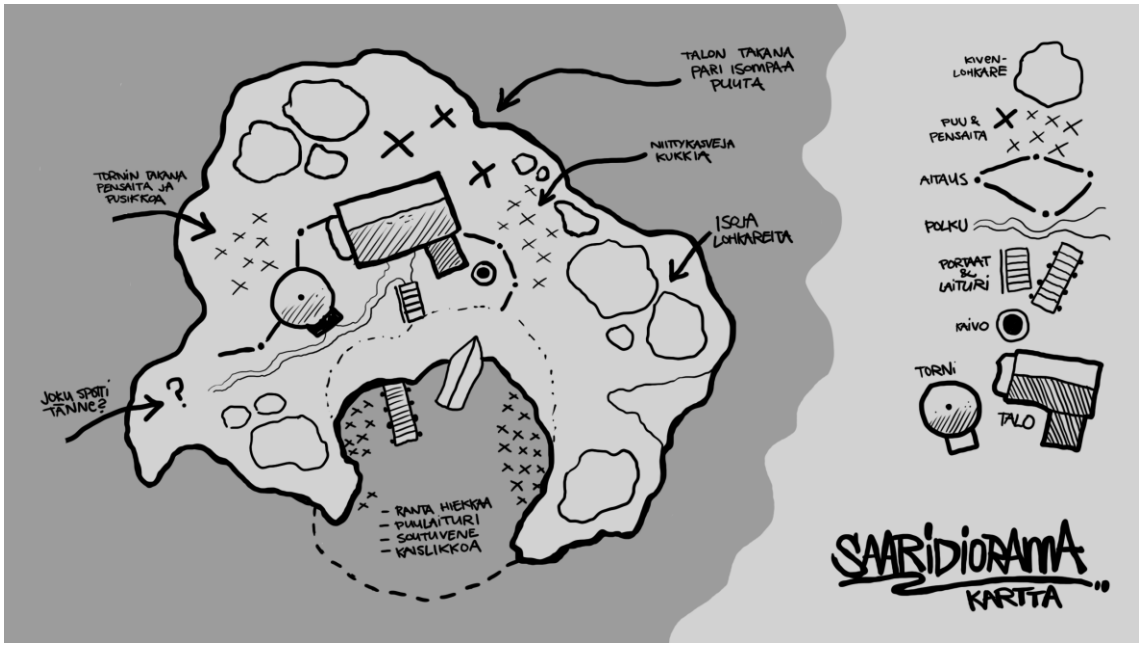
Visuaaliseksi tyylilajiksi valikoitui realismi. Realististen ympäristöjen toteutuksessa haastavinta on tarvittavien assettien yleisesti korkea määrä ja yksityiskohtaisuus, jotka molemmat sekä vaativat, että mahdollistivat monipuolisen ennakoivan optimoinnin tarpeen, joka puolestaan tuki opinnäytetyön projektiosuuden tavoitteita.

Ympäristön teemaksi valittiin aikakaudeksi keskiaika ja miljööksi pieni itämerellinen saari. Koska peliympäristöissä erilaisilla sää- ja valaisuolosuhteilla on huomattavasti merkitystä sekä tunnelman, että karttojen uudelleenkäytettävyyden kannalta, haluttiin ympäristöstä tehdä sekä päivä- että yöversiot. Näille versioille luotiin huomattavan erilaiset valaisu- ja sääolosuhteet sekä efektit.

Saari miljöönä mahdollisti luonnollisesti suljetun ympäristön, eikä tuotantovaiheessa tarvittu meren luomista lukuun ottamatta toteuttaa muuta taustamaisemaa. Ulkosaaristossa sijatsevan pienen luodon kallioinen, karu rakenne yksinkertaisuudessaan ja kapeassa biodiversiteetissään mahdollisti myös graafisten

elementtien vähäisyyden pysytellen edelleen tavoitteessa realistisesta visuaalisesta tyylistä.

Rinnakkain ympäristön suunnittelun kanssa tehtiin konseptointia. Ympäristön visualisointi ja kartoittaminen tehtiin yksinkertaisin viivapiirroksin, sillä projektin aikarajoitteet eivät sallineet suurien illustraatiokokonaisuuksien toteuttamista. Konseptipiirroksia käytettiin apuna tarvittavien rakennusten elementtien sekä saaren luonnonelementtien listaamisessa. Yksinkertaiset kartat puolestaan tukivat ennakkoon kokonaiskuvan ja mittakaavan hahmottamista (ks. kuva 15).



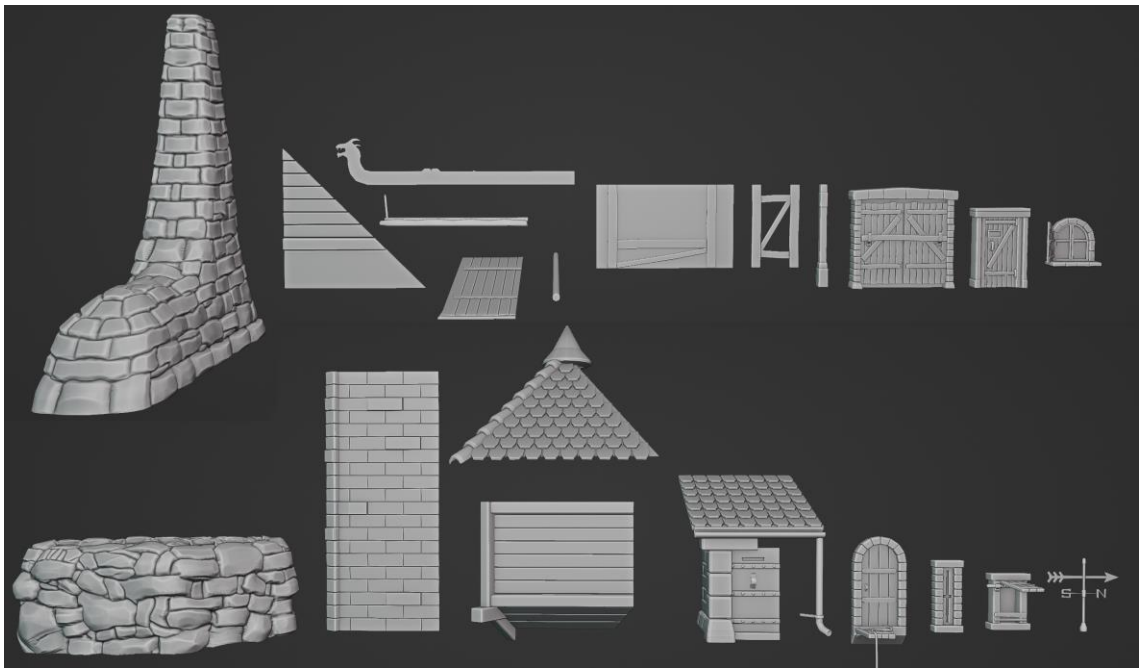
Kuva 15. Viivapiirroksena toteutettu kartta ympäristöstä (Lauerma 2024).

Projektissa käytettävät ohjelmistot päätettiin ennakkoon suunnitteluvaiheessa. Tämä helpotti työvaiheiden jäsentelyä ja toteuttamista ohjelmakohtaisesti, sekä loi pohjaa mahdollisimman iteratiivisille työtavoille.

6.3 Mallinnusvaihe ja modulaarisuus

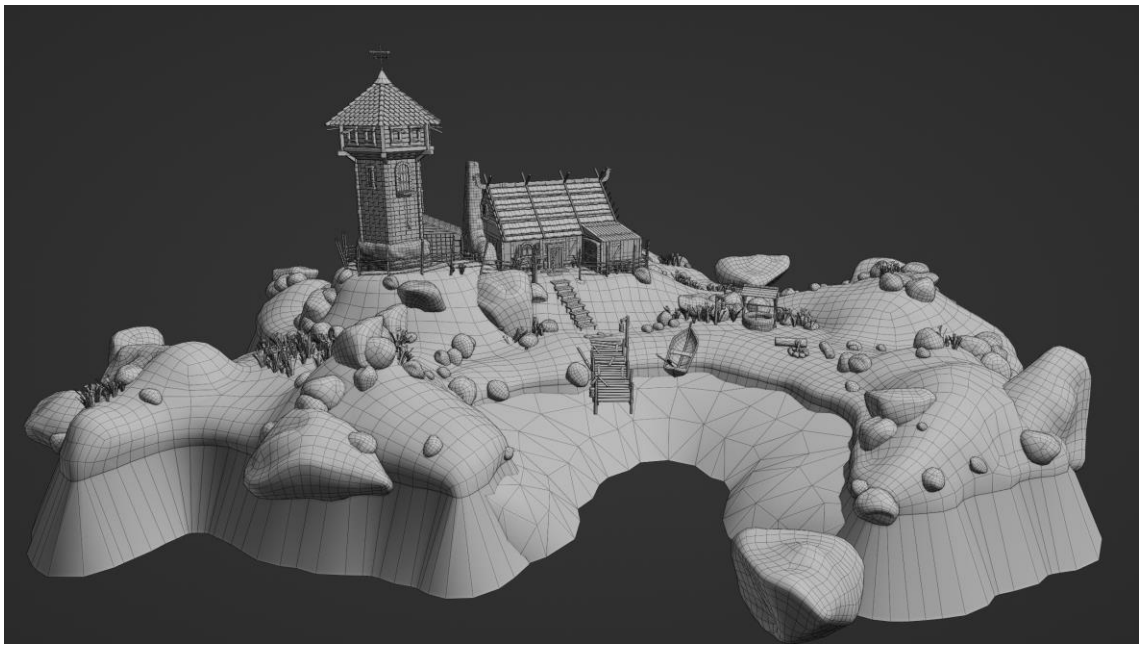
Ympäristön mallinnus toteutettiin lähes kokonaan Blenderissä. Rakennuksien keskiössä oleva torni sekä talo toteuttiin mahdollisimman modulaarisesti. Tällä

tarkoitetaan sitä, että rakennukset koostuvat tarkoituksellisesti vähäisestä määrästä erilaisia osia, jotka ovat kuitenkin keskenään mahdollisimman yhteensopivia ja monikäyttöisiä. Täten näennäisesti pienellä määrällä komponentteja pystyy luomaan huomattavan määrän erilaisia yhdistelmäkokonaisuuksia. Modulaarisuudella on tärkeä rooli optimoidun peliympäristön toteuttamisessa läpi tuotannon, mutta se vaatii erityistä tarkkuutta suunnitteluvaiheessa.



Kuva 16. Rakennuksien 3D-malleissa hyödynnettiin modulaarisuutta (Lauerma 2024).

Koska ympäristössä oli rakennuksien lisäksi huomattava määrä luonnonmuotoja, etenkin saaristomaisemaan vahvasti kuuluvia kiviä ja kallioita, hyödynnettiin perinteisen laatikkomallinnuksen lisäksi digitaalista veistoa. Tämän tyylisiä assetteja toteutetaan usein fotogrammetrian keinoin, mutta tässä projektissa siihen ei aika-, sää- ja ohjelmistorajoitteiden puitteissa tarjoutunut helposti lähestyttävää mahdollisuutta, eikä projektissa haluttu käyttää ulkopuolisia assetteja. Vaikka Blenderissä on omasta takaa varsin monipuolinen kattaus työkaluja digitaaliseen veistoon, käytettiin tässä projektissa siihen erillistä ohjelmaa, Zbrushia. Veistettyjä, korkeapolygonaalisia malleja käytettiin normaalikarttojen leipomiseen matalatiheyksisten mallien pinnalle yksityiskohtien luomista varten.



Kuva 17. Saaren 3D-malli kokonaisuudessaan. Rakennukset mallinnettiin perinteisesti, mutta saaren maanmuodot ja kivet veistettiin ZBrushissa (Lauerma 2024).

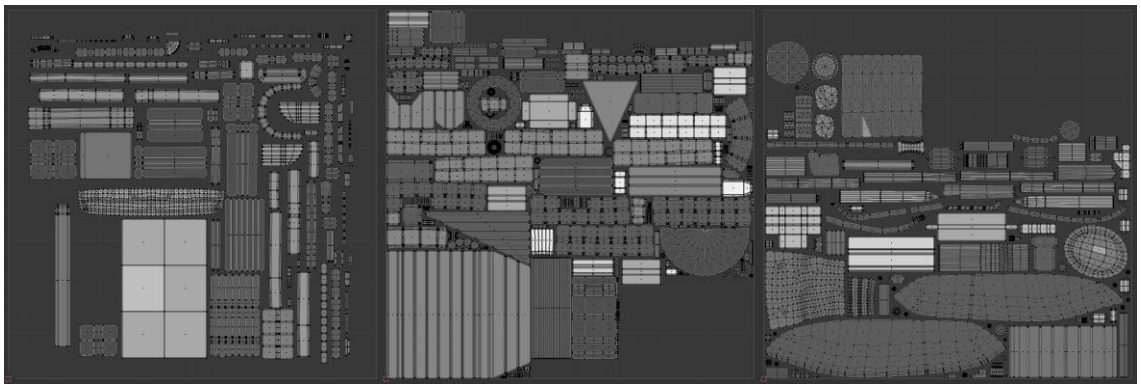
Itse saaren mallinnusvaiheessa pohjamuodot ja maaperä ensiksi veistettiin. Sen jälkeen rakennukset asetettiin halutuille paikoille, maata muovattiin niiden kohdalta tasaisemmiksi ja perustuksia varten sopivammiksi, ja sille luotiin projektiin sopiva polygonaalinen tarkkuus retopologialla (ks. kuva 17). Viimeisenä saari maalattiin kolmella verteksivärillä, joiden avulla se pystyttiin maskaamaan ja teksturoimaan erityisesti tähän tarkoitukseen luoduilla materiaaleilla Unrealissa, joissa käytettiin saumattomia, niinkutsuttuja tile-tekstuureita.

6.4 UV-kartoitus ja atlasointi

Ympäristöelementtien UV-kartoittaminen tehtiin Blenderissä. Koska UV-saarekkeiden rooli tekstuureiden suorituskyvyn parantamisessa on tärkeää, tehtiin tämä työvaihe erityisellä tarkkuudella.

Yksi modulaarisesti mallinnettujen mallikokonaisuuksien mukanaan tuoma etu, UV-saarekkeiden vähäisyys, hyödynnettiin asettelemalla saarekkeet rakennuskohtaisesti kaikista elementeistä samalle UV-kartalle. Koska takeita siitä, että

talo todella sijaitsisi kaikissa karttaiteraatioissa tornin kanssa samassa paikassa ei ollut, ei tornin ja talon elementtien UV-saarekkeita yhdistetty keskenään. Kahden rakennuksen lisäksi kolmas tällainen UV-atlas luotiin ympäristön propeille, kuten veneelle, kaivolle, laiturille ja aidoille (ks. kuva 18).



Kuva 18. Rakennuksien ja proppien UV-atlakset (Lauerma 2024).

Kaikkien ympäristön rakennelmien tekstuuriatlaksien tekselitiheys on kuitenkin sama. Tämä takaa yhtenäisyyden visuaalisessa tarkkuudessa tekstuurien suhteen. Peliympäristön rakennuksien sekä proppien tekselitiheydeksi valittiin 512 pikseliä per neliömetri, joka on yleinen tarkkuus peleissä, joissa kamerakuvakulma on pelihahmon olalta (nk. kolmannen persoonan kuvakulma). Mikäli ympäristöä olisi ollut tarpeen päästä katsomaan ensimmäisen persoonan kuvakulmasta, olisi tekselitiheys todennäköisesti tuplattu, mikä olisi puolestaan nelinkertaistanut tekstuurikarttojen koon ja siten nostanut huomattavasti resurssien kulutusta tekstuurien suhteen esimerkiksi muistinkäytössä.

Saarekkeiden skaalat määriteltiin haluttuun tekselitiheyteen käyttäen Blenderin ZenUV-lisäosaa, jolla usein konstikas tekselitiheyden yhtenäistäminen saatiin tehtyä automatisoidusti. Samalla työkalulla kaikki identtiset saarekkeet aseteltiin päällekäin, jotta ne vievät mahdollisimman vähän tilaa UV-kartalla.

Lopuksi saarekkeet pakattiin Packmaster -lisäosalla, joka käyttää saarekkeiden pakkaamiseen näytönohjaimen laskentatehoa ja luo algoritmein mahdollisim-

man tehokkaan pinta-alan käytön ja saarekkeiden tasaisen asettelun UV-laatalle. UV-karttoihin jätettiin kuitenkin tietoisesti tyhjiä alueita, jotta mahdollisesti tarpeen mukaan jälkikäteen karttoihin lisättäviä osia voitaisiin kartoittaa samalle kartalle ja käyttämään samaa materiaalia. Näin myös kävi projektin aikana.

Itse saaren maanmuotojen mallille tai maisemoinnissa oleennaisessa roolissa oleville kiville ei tehty manuaalisesti käsin aseteltuja UV-karttoja, vaan niille generoitiin yhteensopivuuden takaamiseksi automaattiset UV-kartat, ja niissä käytettiin Unreal Enginessä luotuja erityismateriaaleja. Näistä materiaaleista lisää myöhemmin omassa kappaleessaan.

6.5 Tekstuurien maalaaminen

Mallien teksturointia varten tehdyn suunnittelu-, mallinnus- ja UV-kartoitusvaiheen jälkeen oli aika aloittaa itse tekstuurien maalaaminen. Ohjelmistona käytettiin tässä projektin osuudessa nykyään Adoben omistamaa, aikaisemmin Allegorithmicin kehittämää Substance Painter -ohjelmistoa. Kyseinen ohjelma on muodostunut pelialalla suosituksi ja standardisoituneeksi osaksi tuotantoa, erityisesti koskien tässä projektissa käytettävää PBR-teksturointia ja tavoitellessa realistista lopputulosta. Substance Painterista löytyy myös kohdennetusti Unreal Engineä varten luotuja tekstuurien julkaisuun tarkoitettuja esiasetuksia, jotka helpoittivat tekstuurikuvien julkaisemista suoraan Unrealissa toimiviksi versioiksi.

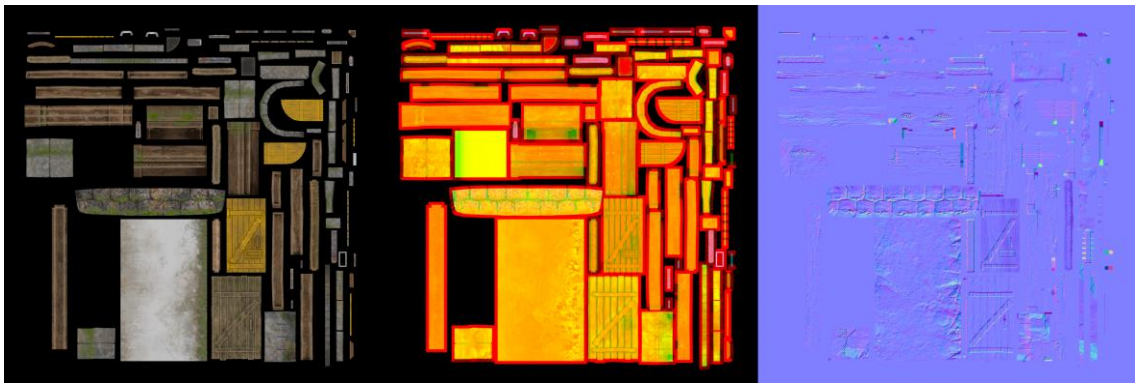
Yksilölliset ja yksityiskohtaiset tekstuurit maalattiin suoraan mallien päälle Substance Painterissa. Keskiaikanen teema mahdollisti mielenkiintoiset ja monimutkaiset luonnonmateriaalit, kuten sammaloituneet kivet tai sääolosuhteiden kuluttamat puurakenteet (ks. kuva 19).

Kuva 19. Substancessa teksturoituja ja renderöityjä ympäristön assetteja (Lauerma 2024).

Aikasemmassa työvaiheessa digitaalisessa veisto-ohjelmassa toteutetut korkeapolygonaaliset, kivitiilistä rakennetut talon ja tornin perustukset, toimivat normaalikarttojen leipomisprosessin lähteenä, ja siten saatiin heijastettua huomattavan yksityiskohtaisia kokonaisuuksia suhteellisen matalatiheyksisen pohjaverkon päälle. Yksityiskohtien leipomista normaalikarttoihin olisi voitu projektissa hyödyntää enemmänkin, sillä esimerkiksi tornin seinien kivitiilipinta ei olisi välttämättä tarvinnut oikeaa geometriaa, vaan tasaisessa pinnassa kohoavat tiilet olisi oikean geometrian hyödyntämisen sijaan voitu leipoa tasaisiin, polygonaalisesti kevyempiin tasopintoihin normaalikarttoja käyttämällä.

Tekstuurikarttojen koko määräytyi tarvittavan tekselitiheyden mukaan. Jo mallinnusvaiheessa tehty päätös tiheydestä oli sidottu kiinni siihen, että yksittäisen tekstuurikuvan koon tulisi olla maksimissaan nk. 4k-resoluutiossa, eli kuvan koon tulisi olla enintään 4096x4096 pikseliä. Mikäli tekstuurien tarkkuutta olisi haluttu myöhemmin laskea, olisi resoluutiota voitu pudottaa pelimoottorissa pakkaamalla.

Yksi Substance Painterin vahvuuksista perinteisiin maalausohjelmistoihin verrattuna on mahdollisuus maalata useita tekstuurikuvakarttoja yhtä aikaa, jolloin maalatessa on mahdollista vaikuttaa yhtäaikaisesti suoraan sekä diffusikuvakartan väreihin, karkeuskarttojen karkeuksiin, sekä metallisuuteen maalausjäljessä (ks. kuva 20). Tästä oli merkittävästi hyötyä ympäristössä, jossa esiintyy samalla alueella esimerkiksi lahonnutta puuta, ruosteisia nauloja, olkikattoa tai graniittia, joista jokaisella on huomattavasti toisistaan eroavia hienovaraisia eroja esimerkiksi heijastavuudessa.



Kuva 20. Talon tekstuureita: Pohjaväri-, ORM- ja normaalikartta. ORM on lyhennys sanoista occlusion, metal ja roughness, ja siinä yhdistyy eri värikanavissa kolme eri tekstuurikarttaa (Lauerma 2024).

Yksityiskohtien nopeaa tuottamista varten käytettiin paljon Substance Painterin generaattoreita, joilla pystyy automatisoimaan esimerkiksi esineiden kohoumiin muodostuvia kulumia. Samoilla generaattorityökaluilla voitiin myös nopeuttaa lautojen väleihin, kivien halkeamiin ja vastaaviin syvennyksiin kertyvää likaa, sammalta ja kosteutta visuaalisen yksityiskohtaisuuden maksimoimiseksi.

Ympäristössä esiintyvät kasvit teksturoitiin hyödyntämällä läpinäkyvyyttä polygonimäärän vähentämiseksi (ks. kuva 21). Muutamasta polygonista koostuvan tason päälle maalattiin tarkasti kasvin eri osat yksityiskohtineen, ja läpinäkyvä alue määriteltiin mustavalkokartalla, jota käytettiin maskina Unreal Enginessä. Tätä tekniikkaa kutsutaan alpha clippingiksi, ja se on peliympäristöjen kasvien

luomisessa standardisoitunut metodi. Kasveille leivottiin korkeapolygonaalisemmasta versiosta myös normaalikartat, jolloin illuusio pehmeistä muodoista korostui.

Kuva 21. Kasvit toteutettiin yksinkertaisen pohjaverkon päälle mustavalkokuvaa läpinäkyvyysmaskina käyttäen. Vivahteikkuutta lehtien muotoon luotiin leipomalla normaalikartat korkeatiheyksisestä kasvimallista (Lauerma 2024).

Overdraw-ilmiön aiheuttamaa negatiivista vaikutusta suorituskykyyn pyrittiin ehkäisemään kasvimalleissa minimoimalla läpinäkyvän pinnan määrä. Tämä tehtiin rajaamalla mallin pintaverkon pinta-ala olemaan mahdollisimman lähellä kasvin tekstuurien näkyviä osia.

Koska ympäristön yöversiossa haluttiin hyödyntää talon sekä tornin ikkunoista paistavaa valoa tunnelmanluojana, tehtiin niille erilliset emissiokartat, jotka pystyttiin kytkemään Unrealissa päälle tai pois riippuen valo-olosuhteista (ks. kuva 22).

Kuva 22. Tornin ja talon ikkunoissa sekä pihalampuissa hyödynnettiin emissiokarttaa valoisan pinnan luomiseen ympäristön yöversiossa (Lauerma 2024).

Näiden emissiivisten tekstuurien yhteydessä käytettiin projektin toteutuksen loppuvaiheessa tukena dynaamisia valonlähteitä realistisen vaikutelman tukemiseksi.

6.6 Saumattomien tekstuurien käyttö ja erityismateriaalit

Rakennusten käsin maalattujen uniikkien tekstuurien lisäksi projektissa hyödynnettiin muutamaa Substance Painterissa tehtyä saumatonta tekstuuria, eli niinkutsuttua tile-tekstuuria. Tämä on tarpeellista tilanteessa, jossa mallilla on liian suuri pinta-ala valitun tekselitiheyden puitteissa, kuten tässä projektissa itse saaren maanmuodoilla.

Mikäli saaren pinta olisi UV-kartoitettu aikasempien elementtien tapaan, olisi tekselitiheys todella matala, ja tekstuuripinta näyttäisi erittäin epätarkalta muiden teksturoitujen elementtien rinnalla. Halutun tekselitiheyden saavuttaminen olisi vaatinut lukuisia, valtavan kokoisia tekstuurikarttoja UDIM-kartoitetussa pinnassa, joka olisi resurssienkäytön suhteen erittäin raskasta ja epäoptimoitua.

Maaperää varten tuotettiin kolme saumatonta tekstuurivarianttia: Rantahiekka, sammaleinen ruohikko sekä kalliopinta (ks. kuva 23). Saumattomuutta tehdessä

kiinnitettiin erityistä huomoita tekstuurin tasalaatuisuuteen, sillä suurella pinnalla tekstuuria replikoidessa on katsojan erityisen helppo huomata pinnalla toistuvat muodot tai kuviot.

Kuva 23. Saumattomia tekstuureita luotiin maaperän teksturointia varten. Kuvassa niiden pohjavärikartat (Lauerma 2024).

Mallinnusvaiheessa maanpinnan vertekseille käytettyjä kolmea verteksiväriä varten luotiin Unreal Enginessä erityismateriaali. Sen pystyy toteuttamaan graafisessa noodipohjaisessa materiaalieditorissa. Materiaaleille luotiin parametrit, joissa käytössä olevia tekstuurikarttoja voidaan skaalata ja rotatoida halutulla tavalla. Tämä mahdollistaa tekstuurikarttojen visuaalisen vaikutelman editoimisen suoraan pelimoottorissa, mikäli joku tekstuuri näyttääkin lopputuloksessa epärealistiselta.

Verteksivärien pohjalta luotiin materiaalissa maskit, jotka määrittivät mille alueelle mitäkin pintamateriaalia sijoitetaan. Siten jyrkkiin seinämiin saatiin kalliopintaa, tasaisiin pintoihin nurmikkoa, sekä rannoille hiekkaa. Unreal Engine osaa sekoittaa materiaalien kohtauspaikoissa tekstuureita saumattomasti, ja lopputulos on visuaalisesti tyydyttävä (ks. kuva 24).

Kuva 24. Unreal Engine sekoittaa saumattomasti eri maamateriaaleja kolmella saumattomalla tekstuurikartalla verteksivärien pohjalta (Lauerma 2024).

Toinen erityismateriaali luotiin kiville. Koska niiden UV-kartoittaminen olisi ollut maanpinnan tavoin suuren pinta-alan takia suorituskyvyn kannalta tehotonta, käytettiin materiaalia, jossa hyödynnettiin automaattista triplanaarista UV-kartoitusta. Sen jälkeen kivien materiaaliin kiinnitettiin aikasemmin toteutettu saumaton kivitekstuurikartta. Saman kartan käyttö pinnan muodoissa sekä irtokivissä loi illuusion yhtenäisestä kivimateriaalista kautta ympäristön.

6.7 Dekaalit

Unreal Enginessä pystyy toteuttamaan dekaaleiksi (eng. Decal) kutsuttuja heijastamiseen pohjautuvia lisätekstuureita, joita voidaan sijoittaa jo rakennetun ympäristön päälle. Näitä käytetään yleisesti videopeliteollisuudessa väliaikaisten efektitekstuurien luomiseen, kuten esimerkiksi räjähdyksien jättämiin jälkiin, luodinreikiin tai veriroiskeisiin. Tässä projektissa niitä hyödynnettiin tilanteissa, joissa kivien ja maanpinnan polygoniverkkojen kohtauspisteet erottuivat liian selkeästi esimerkiksi äkkinäisen materiaalimuutoksen aiheuttamana. Tällainen dekaalitekstuuri asetettiin mm. kaivon alla olevaan maahan (ks. kuva 24). Dekaalitekstuurin reuna erottuu verteksivärein toteutettua tekstuurin sekoittumista terävämpänä.

Aikaisemmin maanpintaa ja kiviä varten tuotettuja saumattomia tekstuureita hyödynnettiin uudelleen, tällä kertaa rajaten ne läpinäkyvyyskarttoja maskeina

käyttäen, ja ne kiinnitettiin Unreal Enginen dekaalijärjestelmään. Sen jälkeen niitä sijoiteltiin käsin kohtiin, missä tarvittiin kahden pinnan välillä sulavampaa transitiota materiaalista toiseen. Niillä lisättiin myös tarpeen tullen joidenkin suurempien kivenlohkareiden päälle ruohoa ja hiekkaa, jotta luonnonmuovaaman orgaanisen ympäristön illuusio olisi mahdollisimman hyvä.

6.8 Valaisu

Ympäristön mallintamisen, teksturoimisen ja miljöön Unreal Enginessä rakentamisen jälkeen oli aika toteuttaa ympäristön valaisu. Pelimoottorissa valaisutekniikaksi valikoitiin uusimman Unreal Enginen viidennen version mukana tullut lumen -järjestelmä, jota tuettiin laitteistopohjaisella säteenseurannalla (eng. Hardware ray tracing). Tämä yhdistelmätekniikka mahdollistaa erittäin realistisen, mutta silti laskennaltaan perinteistä esirenderöintiä nopeamman, reaaliaikaiseen käyttöön sopivan valaisun. Tämänkaltainen valaisutekniikka on viime aikoina yleistynyt videopeleissä, mutta on huomion arvoista, että se on huomattavasti raskaampaa, kuin paljolti peliteollisuudessa käytetty perinteinen esilaskettu, leivottuun valaisuun perustuva järjestelmä.

Ympäristön tuotannossa kauttaaltaan painotettu suorituskyvyn optimointi kuitenkin mahdollisti raskaan valaisun realismia tavoitellessa, joten projektissa päädyttiin käyttämään tarpeelliset resurssit tähän, sillä valaisun rooli lopputuloksessa on huomattava, erityisesti PBR-tekstuurien osalta. Mikäli ympäristöä olisi kuitenkin ollut tarpeen käyttää vanhemmalla laitteistolla, joka ei tukisi säteenseurantaa, tai esimerkiksi optimoida sitä laskentateholtaan heikommille laitteille, olisi voitu valaisumetodista palata takaisin perinteiseen esilaskettuun, valaisu-UV-karttoihin luotuun leivottuun valaisuun.

6.9 Partikkeliefektit ja videoiden renderöinti

Ympäristöön toteutettiin pelimoottorissa valaisun lisäksi partikkeliefektejä käyttäen Unreal Enginen omaa Niagara-partikkelimoottoria. Tämän avulla talon savupiipusta sekä rannalla sijaitsevasta nuotiopaikasta saatiin nousemaan savua,

jonka partikkeleihin tehtiin Photoshopissa kuvamanipuloimalla ja maalaamalla omat alpha-kanavaa hyödyntävät tekstuurit (ks. kuva 25). Nuotiopaikalle tehtiin myös savun mukana nousevia kipinöitä, jotka tukivat realistista vaikutelmaa luonnostaan tuulisessa ympäristössä. On huomion arvoista, että tällaiset niinkutsuttua alpha blendingiä hyödyntävät partikkeliefektit ovat laskennallisesti erittäin kuormittavia suorituskyvylle, ja niitä tulee käyttää sen takia harkiten.

Kuva 25. Alphakanavallinen savuhattara sekä sitä käyttävä pelimoottoriin luotu partikkeliefekti (Lauerma 2024).

Savujen lisäksi toteutettiin kaksi pelkästään yöversiossa esiintyvää partikkeliefektiä, joista toinen oli kukkapusikoissa ja rantakaislikoissa lentävät tulikärpäset, ja toinen yöllä päällä olevien öljylamppujen ympärillä parveilevat pienhyönteiset.

Lopuksi ympäristöön sijoitettiin muutamia kameroita, joille tehtiin animoimalla omat kamera-ajot. Unreal Engine mahdollistaa kameroita luodessa esiasetusten käytön, jotka imitoivat oikeita elokuva-alla käytettyjä kameroita esimerkiksi polttoväleineen. Näitä käyttäen ympäristöstä renderöitiin esittelyvideo (liite 1) sekä yö-, että päiväolosuhteissa.

7 Yhteenveto

Teksturointi on yksi merkittävimmistä yksittäisistä työvaiheista peleihin tuotettavien 3D-mallien visuaalisuuden kannalta. On tärkeää tietää esituotantovaiheesta alkaen, minkälaiseen päätelaitteeseen grafiikkaa luodaan ja selvittää sekä sen tekniset rajoitteet, että määritellä pelissä haluttu visuaalinen tyyli.

Koska tekstuurien tuotanto ja käyttö jakaantuu useimmiten monelle eri ohjelmistolle, kuten mallinnusohjelmalle (UV-kartoitus), teksturointiohjelmalle (tekstuurien tuotanto ja kuvakarttojen julkaisu) ja pelimoottorille (tekstuurien käyttö sävyttimissä, materiaaleissa ja optimointi), on myös tärkeää päättää, mitkä ohjelmistot ja tekniikat tukevat parhaiten päätelaiterajoitusten ja valitun tyylin huomioon ottaen itse tuotantoketjua.

Eri assettien visuaalinen yhtenäisyys on tärkeä tekijä peligrafiikan näyttävyyden kannalta ja siten teksturointiprosessissa usein monien eri artistien yhteistyössä tuottaman sisällön kannalta on tärkeää tunnistaa työtavat, joilla tätä yhtenäisyyttä voidaan korostaa sekä ylläpitää. Suunnitteluvaiheessa sovitut yhteiset säännöt koskien tuotantotapoja edesauttavat tätä visuaalista yhtenäisyyttä. Myös UV-kartoituksen rooli on erottamaton osa itse teksturointiprosessia, ja on kriittisen tärkeää, että sitä toteuttavat mallintajat sekä itse tekstuureita luovat artistit tekevät saumatonta yhteistyötä jo esituotantovaiheessa, mikäli nämä työvaiheet jakaantuvat kahdelle eri ryhmälle artisteja.

Tekstuureilla on merkittävä vaikutus pelimoottoreissa reaaliaikaisesti laskettavan grafiikan suorituskykyyn. Koska tekstuureiden suorituskykyä optimoidaan lähtökohtaisesti läpi koko tuotantoketjun vastakohtana pelkästään jälkituotannossa tehtäviin optimointitoimiin, kannattaa tekstuureiden tuotantovaiheessa kiinnittää erityistä huomiota toteutustapoihin, ja pyrkiä minimoimaan niiden negatiivinen vaikutus suorituskykyyn hyvillä ja yhteinäisillä työskentelytavoilla. Tämä saavutetaan tarkalla suunnittelulla sekä laaja-alaisella tietämyksellä erilaisista teksturointitekniikoista.

Lähteet

Adobe i.a. 3D texturing solution with Adobe Substance 3D. Verkkosivu. Adobe. https://www.adobe.com/products/substance3d/discover/3d-texturing.html#:~:text=3D%20texturing%20techniques.&text=You%20can%20paint%20and%20create,combination%20of%20all%20three%20methods (viitattu 17.10.2024).

Anuworakarn, Benjamin 2019. Why you really should be using mipmapping in your graphics applications. Imagination. Blogi 15.8.2019. https://blog.imaginationtech.com/why-you-really-should-be-using-mipmapping-in-your-graphics-applications/ (viitattu 9.10.2024).

Belec, Arijan 2023. Photorealistic materials and textures in Blender cycles - fourth edition. E-kirja. Birmingham: Packt Publishing. https://learning.oreilly.com/library/view/photorealistic-materials-and/9781805129639/ (viitattu 4.5.2024).

Burns, Scot Daniel 2023. Trimsheets. Verkkosivu. Beyond Extent. https://www.beyondextent.com/deep-dives/trimsheets (viitattu 18.11.2024).

Dickinson, Chris 2017. Unity 2017 Game Optimization – Second Edition. E-kirja. Birmingham: Packt Publishing. https://learning.oreilly.com/library/view/unity-2017-game/9781788392365/ (viitattu 16.10.2024).

Epic Games i.a.a. Essential material concepts. Verkkosivu. Epic Games. https://dev.epicgames.com/documentation/en-us/unreal-engine/essential-unreal-engine-material-concepts (viitattu 18.10.2024).

Epic Games i.a.b. Creating and using material instances. Verkkosivu. Epic Games. https://dev.epicgames.com/documentation/en-us/unreal-engine/creating-and-using-material-instances-in-unreal-engine (viitattu 18.10.2024).

Epic Games i.a.c. Instanced materials. Verkkosivu. Epic Games. https://dev.epicgames.com/documentation/en-us/unreal-engine/instanced-materials-in-unreal-engine (viitattu 18.10.2024).

Epic Games i.a.d. Textures. Verkkosivu. Epic Games. https://dev.epicgames.com/documentation/en-us/unreal-engine/textures-in-unreal-engine (viitattu 18.10.2024).

Goodarzi, Mehdi & Shahbazi, Nazanin 2024. Exploring the world of 3D textures: A comprehensive guide. Blogi 11.7.2024. Pixune. https://pixune.com/blog/3d-texturing/ (viitattu 17.10.2024).

Ilett, Daniel 2022. Building quality shaders for Unity: Using shader graphs and HLSL shaders. E-kirja. New York: Apress. https://learning.oreilly.com/library/view/building-quality-shaders/9781484286524/ (viitattu 10.10.2024).

Kumar, Abhishek 2020. Beginning PBR texturing: Learn physically based rendering with Allegorithmic's Substance Painter. E-kirja. New York: Apress. https://learning.oreilly.com/library/view/beginning-pbr-texturing/9781484258996/ (viitattu 11.10.2024).

Lampel, Jonathan 2017. Bit depth & how compression affects normal maps. Verkkosivu. CG Cookie. https://cgcookie.com/posts/bit-depth-how-compression-affects-normal-maps (viitattu 23.10.2024).

Li, Jingtian & Watkins, Adam & Arevalo, Kassandra, & Tovar, Matthew 2020, Creating Games with Unity, Substance Painter, and Maya. E-kirja. Milton: Taylor & Francis Group. https://ebookcentral.proquest.com/lib/metropolia-ebooks/detail.action?docID=6384449 (viitattu 8.10.2024).

Mendoza Guevarra, Ezra Thess 2020. Creating game environments in Blender 3D: Learn to create low poly game environments. E-kirja. New York: Apress. https://learning.oreilly.com/library/view/creating-game-environments/9781484261743/ (viitattu 4.5.2024).

Mower, Nick i.a. Your guide to texture compression in Unreal Engine. Verkkosivu. Techarthub. https://techarthub.com/your-guide-to-texture-compression-in-unreal-engine/ (viitattu 23.10.2024).

PCMag i.a. Encyclopedia. Verkkosivu. PCMag. https://www.pcmag.com/encyclopedia/term/alpha-blending (viitattu 15.10.2024).

Unity i.a.a. Introduction to materials. Verkkosivu. Unity Documentation. https://docs.unity3d.com/Manual/materials-introduction.html (viitattu 18.10.2024).

Unity i.a.b. Introduction to material variants. Verkkosivu. Unity Documentation. https://docs.unity3d.com/Manual/materialvariant-concept.html (viitattu 18.10.2024).

Villanueva, Nova 2021. Beginning 3D game assets development pipeline: Learn to integrate from Maya to Unity. E-kirja. New York: Apress. https://learning.oreilly.com/library/view/beginning-3d-game/9781484271964/ (viitattu 4.5.2024).

Watkins, Adam 2012. Creating games with Unity and Maya. E-kirja. Lontoo: Routledge. https://learning.oreilly.com/library/view/creating-games-with/9780240818818/ (viitattu 4.5.2024).

Kuvalähteet

Kuva 1. Vaskevich, Stefan i.a. 3D model texturing guide. Verkkosivu. Cyberfox. https://cyber-fox.net/blog/how-to-texture-3d-model/ (viitattu 18.11.2024).

Kuva 2. Belec, Arijan 2023. Photorealistic materials and textures in Blender cycles - fourth edition. E-kirja. Birmingham: Packt Publishing. https://learning.oreilly.com/library/view/photorealistic-materials-and/9781805129639/ (viitattu 4.5.2024).

Kuva 3. Blender i.a. Introduction. Verkkosivu. Blender 4.2 Manual. https://docs.blender.org/manual/en/latest/editors/uv/introduction.html (viitattu 18.11.2024).

Kuva 4. Dixon, Sean 2016. UV mapping, texturing and shaders, rigging and animation. Verkkosivu. Medium. https://medium.com/@sdixon3/uv-mapping-texturing-and-shaders-rigging-and-animation-be9b4ddf0d48 (viitattu 18.11.2024).

Kuva 5. Foundry i.a. Pack UVs. Verkkosivu. Foundry. https://learn.foundry.com/modo/content/help/pages/uving/pack_uv.html (viitattu 18.11.2024).

Kuva 6. Harrington, Andrew 2019. Stylized tiling texture workflow. Verkkosivu. 80.lv. https://80.lv/articles/stylized-tiling-texture-workflow/ (viitattu 18.11.2024).

Kuva 7. Lau, Oliver 2018. Creating photogrammetry-based materials. Verkkosivu. 80.lv. https://80.lv/articles/creating-photogrammetry-based-materials/ (viitattu 18.11.2024).

Kuva 8. Lyons, Tony i.a. CG compositing series – 2.4 material AOVs – albedo & RAW lighting. Verkkosivu. Compositing Mentor. https://compositingmentor.com/2024/01/ (viitattu 18.11.2024).

Kuva 9. Eric Chadwick i.a. 3D models. Verkkosivu. Ericchadwick.com. https://ericchadwick.com/3d.html (viitattu 18.11.2024).

Kuva 10. Radivojevic, Filip 2023. Understanding texel density: a comprehensive guide. Renderhub. Blogi 30.10.2023. https://www.renderhub.com/blog/understanding-texel-density-a-comprehensive-guide (viitattu 18.11.2024).

Kuva 11. Dries, Timothy 2023. What is texel density. Verkkosivu. Beyond Extent. https://www.beyondextent.com/deep-dives/deepdive-texeldensity (viitattu 18.11.2024).

Kuva 12. Radivojevic, Filip 2023. Understanding texel density: a comprehensive guide. Renderhub. Blogi 30.10.2023. https://www.renderhub.com/blog/understanding-texel-density-a-comprehensive-guide (viitattu 18.11.2024).

Kuva 13. Apple i.a. Improving texture sampling quality and performance with mipmaps. Verkkosivu. Apple. https://developer.apple.com/documentation/metal/textures/improving_texture_sampling_quality_and_performance_with_mipmaps (viitattu 18.11.2024).

Kuva 14. Lauerma, Niko 2024. Saaridiorama Unreal Enginessä.

Kuva 15. Lauerma, Niko 2024. Ympäristön karttakonsepti.

Kuva 16. Lauerma, Niko 2024. Rakennuksien moduuleita.

Kuva 17. Lauerma, Niko 2024. Saaren 3D-malli Blenderissä.

Kuva 18. Lauerma, Niko 2024. Rakennuksien ja proppien UV-atlakset.

Kuva 19. Lauerma, Niko 2024. Teksturoituja ympäristöassetteja.

Kuva 20. Lauerma, Niko 2024. Talon tekstuurikarttoja.

Kuva 21. Lauerma, Niko 2024. Kasvin tekstuurit ja wireframe-kuva.

Kuva 22. Lauerma, Niko 2024. Rakennuksien itsevalaisevia tekstuureita.

Kuva 23. Lauerma, Niko 2024. Kolme saumatonta tekstuurikarttaa.

Kuva 24. Lauerma, Niko 2024. Saumattomat tekstuurit ympäristössä.

Kuva 25. Lauerma, Niko 2024. Savupartikkeliefekti.

Liitteet

Liite 1. Unreal Enginessä renderöity esittelyvideo ympäristöstä