OULUN AMMATTIKORKEAKOULU

Sanni Rahkola

REALISTISEN 3D-YMPÄRISTÖN LUONTI UNITY-PELIMOOTTORIIN

REALISTISEN 3D-YMPÄRISTÖN LUONTI UNITY-PELIMOOTTORIIN

Sanni Rahkola
Opinnäytetyö
Kevät 2022
Tietojenkäsittely
Oulun ammattikorkeakoulu

TIIVISTELMÄ

Oulun ammattikorkeakoulu
Tietojenkäsittely, Tradenomi

Tekijä: Sanni Rahkola
Opinnäytetyön nimi: Realistisen 3D-ympäristön luonti Unity-pelimoottoriin
Työn ohjaaja: Ilpo Virtanen
Työn valmistumislukukausi ja -vuosi: Kevät 2022 Sivumäärä: 46

Tämän opinnäytetyön tavoitteena oli kuvata realistisen ympäristön suunnittelu- ja toteutusprosessia Unity-pelimoottoriin ja High Definition Renderpipeline -järjestelmään. Samalla tutustuttiin eri tekniikoihin ja ohjelmiin osana opinnäytetyöprosessia. Tämä toteutettiin suunnittelemalla, konseptoimalla, mallintamalla ja luomalla näiden pohjalta kyberpunk-henkinen ympäristö Unityn sisälle.

Opinnäytetyössä kuljetetaan teoriaa ja käytännön osuutta rinnakkain. Lukujen alussa tarkastellaan aiheen teoriaa ja lopussa kuvataan käytännön osuutta. Ensimmäisenä kuvataan ympäristön ja projektin suunnittelua, minkä jälkeen tarkastellaan konseptointia ja siihen liittyviä vaiheita. Tämän jälkeen kuvataan 3D-mallinnusprosessia eritoten Blender-mallinnusohjelmassa ja teksturointia Substance Painter -ohjelmassa. Lopuksi tarkastellaan Unity-pelimoottoria ja High Definition Renderpipeline -järjestelmässä työskentelyä.

Opinnäytetyön tietoperustana oli aiempi kokemus projektissa käytetyistä työkaluista ja menetelmistä, mutta tietoa etsittiin myös verkkosivuilta ja artikkeleista. Opinnäytetyö on kirjoitettu suurimmalta osalta pelinkehityksen näkökulmasta, mutta samoja periaatteita voi noudattaa myös visualisoinnissa ja muissa työtehtävissä, joissa tarkoituksena on realistisen ympäristön luonti pelimoottorin sisälle.

Asiasanat: 3D-mallinnus, Unity, HDRP

ABSTRACT

Oulu University of Applied Sciences
Degree Programme in Business Information Systems

Author: Sanni Rahkola
Title of thesis: Creating realistic 3D Environment in Unity Game Engine
Supervisor: Ilpo Virtanen
Term and year when the thesis was submitted: Spring 2022
Number of pages: 46

The aim for the project was to demonstrate and get familiar with the production of realistic environments in Unity game engine using High Definition Renderpipeline. This was achieved by designing, concepting, modeling and creating a realistic cyberpunk-themed environment in the Unity game engine.

Environment was created using Adobe Photoshop, Blender, Substance Painter and Designer and Unity. Concepts were created using Adobe Photoshop and 3D modeling by Blender. Substance Painter and Designer were used to create realistic textures and the environment was created in Unity game engine using High Definition Renderpipeline.

Most of the theory is written from previous experience with the tools and from various websites and articles. Even though this thesis is written mostly in the perspective of the game industry, the same principles can be followed in visualization and other work tasks where the focus is on creating realistic environments in game engines.

Keywords: 3D modeling, Unity, HDRP

SISÄLLYS

1 JOHDANTO

Erilaisten visualisointien ja ympäristöjen luonti pelimoottoreiden sisällä on iso kokonaisuus, johon tulee kiinnittää huomiota aina suunnittelusta lähtien toimivan kokonaisuuden luomiseksi. 3D-ympäristön luonnissa ja pelinkehityksessä pelimoottorin renderöintiputken valinta vaikuttaa visuaaliseen ilmeeseen ja kehitysvaiheisiin huomattavasti. Lisäksi eri renderöintiputket soveltuvat erilaisiin visuaalisiin tyyleihin ja käyttötarkoituksiin. Pari vuotta sitten uudet renderpipeline-järjestelmät julkaistiin Unity-pelimoottoriin helpottamaan visuaalisten ilmeiden luontia eri käyttökohteisiin. Näistä High Definition Render Pipeline (HDRP) pyrkii korkealaatuisen grafiikan ja ympäristön tuottamiseen tietokone- ja konsolikäyttöön.

Opinnäytetyön tavoitteena oli kuvata realistisen ympäristön suunnittelu- ja toteutusprosessia edellä mainittuun High Definition Renderpipeline -järjestelmään sekä tutustua kyseiseen, suhteellisen uuteen renderöintiputkeen tarkemmin. Työn tarkoituksena oli myös perehtyä paremmin toteutusprosessin tekniikoihin ja harjoitella eri ohjelmien käyttöä osana opinnäytetyöprosessia. Työllä ei ole toimeksiantajaa.

Käytännön osuuden aiheena on kyberpunk-henkisen kadun luominen Unity-pelimoottoriin, mihin toteutetaan osa tarvittavista 3D-objekteista alusta alkaen itse. Näin koko prosessi ympäristön luomisesta Unityn High Definition Renderpipelineen tulee tutuksi. Kyberpunk-henkinen katu ja ulkotilat valittiin, koska heijastavassa valaistussa ympäristössä HDRP:n vahvuudet pääsevät parhaiten esille.

2 3D-YMPÄRISTÖN SUUNNITTELUPROSESSI

Tässä luvussa tarkastellaan pelimaailman 3D-ympäristön toteuttamisessa käytettäviä periaatteita ja suunnittelutapoja. Suunnitteluprosessiin kuuluu ideointi, Moodboard-kuvakokoelman luonti, referenssikuvien haku, konseptointi ja näiden pohjalta 3D-mallintamisen suunnittelu. 3D-mallintamisen suunnitteluun kuuluu tuotettavien objektien listaus konseptoinnin ja referenssikuvien pohjalta.

2.1 Suunnittelu

Ensimmäisenä suunnitteluprosessissa voi lähteä liikkeelle yksinkertaisesta ideasta, jota kehitetään eteenpäin. Kun ympäristöä, paikkaa ja teemaa suunnittelee tarkemmin, on hyvä pitää mielessä myös projektin tarkoitus. Paikka ja teema voivat hyvinkin rajautua riippuen siitä, onko projektin tarkoituksena luoda isompi pelattava alue vai vain toteuttaa pieni huoneistotila esiteltäväksi asiakkaille. (World of Level Design 2021.)

Kun ympäristö, tarkempi paikka ja teema ovat selvillä, on suositeltavaa aloittaa kuvareferenssien etsiminen ja aiheen tutkiminen. Erilaiset ympäristö- ja paikkareferenssit helpottavat yleiskuvan ja kokonaisuuden hahmottamista sekä muodostamista. Moodboard-kuvakokoelma auttaa määrittämään oikean mielikuvan ympäristöstä ja muistuttamaan alkuperäisestä ideasta. Referenssikuvat erilaisista tuotettavista objekteista helpottavat mallintamista, ja valaistusreferenssit puolestaan helpottavat pelimoottorissa työskentelyä. Thumbnailit ja konseptikuvat määrittävät puolestaan konkreettisemmin luotavaa ympäristöä. Tyylisuunta tulee myös monesti määritettyä samaan aikaan, kun kuvareferenssejä etsitään ja tutkimusta aiheesta suoritetaan. (World of Level Design 2021.)

Visuaaliset maamerkit ja mahdolliset kartat on myös hyvä suunnitella ympäristöön. Visuaaliset maamerkit tekevät ympäristöstä uniikin ja helposti tunnistettavan. Erilaiset ylhäältä kuvatut kartat puolestaan helpottavat kokonaisuuden hahmottamista ja tukevat pelinkehitystä myöhemmin pelimoottorin sisällä.

Pelattavaa ympäristöä luodessa sen tarinaa tulisi suunnitella myös osana suunnitteluprosessia. Tarina voi tulla ilmi jo aiemmin pelisuunnittelun yhteydessä, mutta viimeistään kuvareferenssien

haut luovat ainakin jonkinlaista ideaa. (World of Level Design 2021.) Pelkästään reaalimaailman visualisointeihin tarinan luominen ei ole toki oleellista.

Lopuksi olisi suotavaa tehdä listoja ympäristöön tuotettavista kappaleista ja kuvista jo suunnitteluvaiheessa. Referenssikuvien ja konseptoinnin yhteydessä ilmenneet oleelliset objektit kannattaa luetella ja tehdä niiden pohjalta tuotantolista ja pohtia, kuinka tuotantoa hallitaan. (World of Level Design 2021.) Tuotannon hallinnassa käytetään yrityksissä erilaisia projektinhallintamenetelmiä. Pelialalla tähän ovat vakiintuneet erilaiset ketterät menetelmät, kuten Scrum tai Kanban (Clinton 2016).

Tämän opinnäytetyön käytännön osuuden ideana on toteuttaa pieni rajattu kyberpunk-henkinen ympäristö, missä High Definition -renderöintiputken vahvuudet voi tuoda hyvin ilmi. Ympäristö sijoitettiin tulevaisuuteen dystooppiseen maailmaan: projektiympäristön tapahtumapaikkana oli kuja, ja ympäristön teemana oli valaistu ja värikäs yö. Projektin tarkoituksena oli tutkia realistisen ympäristön toteuttamisen prosessia Unity-pelimoottorin High Definition -renderöintiputkeen. Ympäristön ei tarvinnut olla kovin erikoinen ja sisältää uniikkeja yksityiskohtia, koska kyseessä ei ollut myytäväksi tuleva peli vaan testausmielessä luotava tutkimusympäristö. Myös tarina jätettiin katsojalle, ja korkeintaan erilaiset ympäristöobjektit ja ilmapiiri viestivät tarinasta.

3D-objekteja kerättiin taulukkoon moodboard-kuvakokoelman, referenssikuvien ja konseptoinnin pohjalta. Objektit jaoteltiin numeerisesti sen mukaan, kuinka tärkeitä ne olivat opinnäytetyön kannalta, ja sen mukaan, mitkä objektit mallinnetaan itse ja mitkä objektit voisi löytää projektiin erilaisista materiaalipankeista. Mallinnusprosessi aloitettiin tämän pohjalta.

Priority	Structures	Doing yourself Y/N	Status	Notes
1	Walls Top 1-3	Y	Done 1/3	Modular pieces, easy to swap between different materials
1	Floor	Y	Done	Street/ground
1	Curb	Y	Done	
2	Wall Bottom 1-3	Y	-	Modular pieces, easy to swap between different materials
2	Windows 1-4	Y	Done 1/4	Simple housing unit windows and larger shopping windows
2	Gates 1-2	Y	Done 1/2	Metal railing, carage doors
2	Doors 1-2	Y	-	Smaller normal doors
2	Stair entrance	Y	-	
2	Backgroud Buildings	N	Added	Billboard(s) might be good enough/ Use ready pieces from Assetstore
2	Pillars 1-4	Y	Done 1/4	
	Props			
1	Neon Signs 1-6	Y	Done 3/6	Walls, above head, ground (needs mounting too)
1	Wall Lights 1-4	Y	-	
1	Air Vents 1-2	Y&N	Added 1	Small airvents on the wall
1	Storm sewers 1-2	Y	Done 1/2	
2	Trashcans	Y&N	Added 1	
2	Ladders	Y&N	-	
2	Wall Air Conditioner Unit 1-2	Y&N	Added 1	Two variants (or maybe prefab with different attachments)
2	Modular Cables	Y	Done	Hanging cables, going from one unit to another (probably not modular? Might need to be unique)
2	Modular Plumbing	Y	-	90°, connector, straight short, straight, long, end piece, 45° if needed
2	Utility Pole	Y	-	Can be added with different electrical units and cables
2	Roadblocks 1-2	N	Added 1	Pillars and boxes
2	Electrical Units 1-3	Y&N	Done 1 Added 1	
3	Roof Drainage System	Y	-	Same modular pieces as above
3	Modular Railing	Y	-	
3	Crates 1-2	Y&N	-	
3	Cardboard Boxes	Y&N	Added 1	
3	Non-Neon Signs 1-4	Y	-	Normal signs above places
3	Trashbags	Y	-	
3	Random Litter	Y	-	Decals?
3	Vending Machine	Y	-	
3	Ticket Machine	Y	-	
3	Barrels	Y&N	Added 1	
3	Security Camera	Y	-	
	Other (Decals etc.)			
2	Street Marks 1-3	Y	Done 1/3	Road markings etc.
2	Graffitis 1-6	Y&N	-	Do yourself and add graffitis from textures.com
2	Water	Y	Done	Check this what is the best way to do it (Unity had water decal samples already)
3	Smoke/Dust/Electricity VFX	Y	-	Unity might have already samples of these

Kuva 1. Kuvankaappaus taulukosta, johon mallinnettavat ja lisättävät objektit on luotu konseptoinnin ja moodboardin pohjalta.

2.2 Moodboard

Moodboard on kuvakokoelma inspiroivista ja esteettisistä kuvista, joista voi ottaa mallia esimerkiksi pelinkehityksessä tai ympäristön luonnissa. Kuvakokoelman teko on tärkeää, koska se helpottaa ympäristön luontia ja oikean mielikuvan luomista sekä tekijälle että katsojalle. Moodboard laittaa ideat visuaaliseen muotoon ja auttaa täten myös visualisointityötä ja mahdollisesti tarjoaa väripaletin tulevaan työhön. Kuvakokoelmaan palaaminen projektin aikana auttaa myös tekijää keskittymään alkuperäiseen ideaan. (Wolstenholme 2021.)

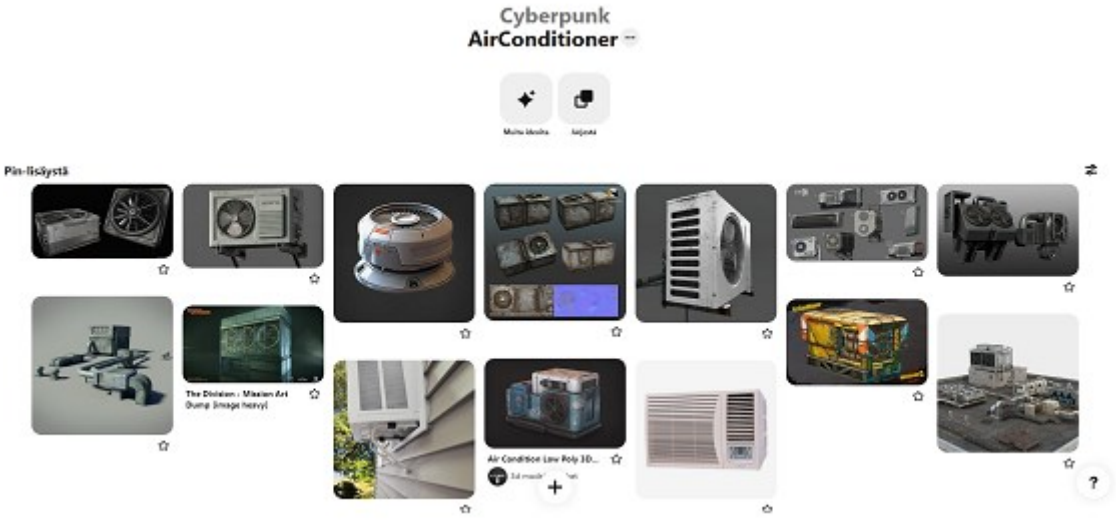
Moodboardiin voi saada kuvamateriaalia esimerkiksi Googlen kuvahaulla. Kuvahausta löydetyt kuvat kootaan yhteen sen jälkeen kuvankäsittelyohjelmassa. Toinen vaihtoehto kuvahaulle on esimerkiksi luoda kuvakokoelma Pinterest-sivustolla. Pinterest-sivustoon voi tehdä erilaisia kokoelmia ja tauluja, ja ne voivat toimia myös referenssikuvina myöhempiä vaiheita varten.

Opinnäytetyön käytännön osuudessa ensimmäiseen kuvakokoelmaan kuvat haettiin Googlen kuvahausta ja Pixabay-kuvapankista. Haetut kuvat koottiin yhteen Adobe Photoshop -ohjelmassa nopeaan taulumaiseen muotoon. Toinen kuvakokoelma koottiin Pinterest-sivustolla, missä erilaisia tunnelmaltaan tai teemaltaan sopivia kuvia lisättiin kokoelmaan. Pinterest-sivustolla koottiin myös samalla referenssikuvia, jotka kategorisoitiin.

Kuva 2. Moodboard-kuvakokoelma koottuna Adobe Photoshopissa käytännön osuutta varten.

2.3 Referenssikuvat

Referenssikuvat ovat kuvia, joista voi ottaa mallia konseptointiin ja visualisointiin. Referenssikuvat voivat riittää konseptoinnin sijasta, jos piirtäminen ei kuulu omiin vahvuuksiin. Näiden kuvien etsiminen helpottaa mallinnusta ja konkretisoi ideoita ja aiemmin koottuja moodboard-kuvakokoelmia. Referenssikuvia voi etsiä esimerkiksi erilaisista kuvapankeista, Pinterest-sivustolta tai omista valokuvista.



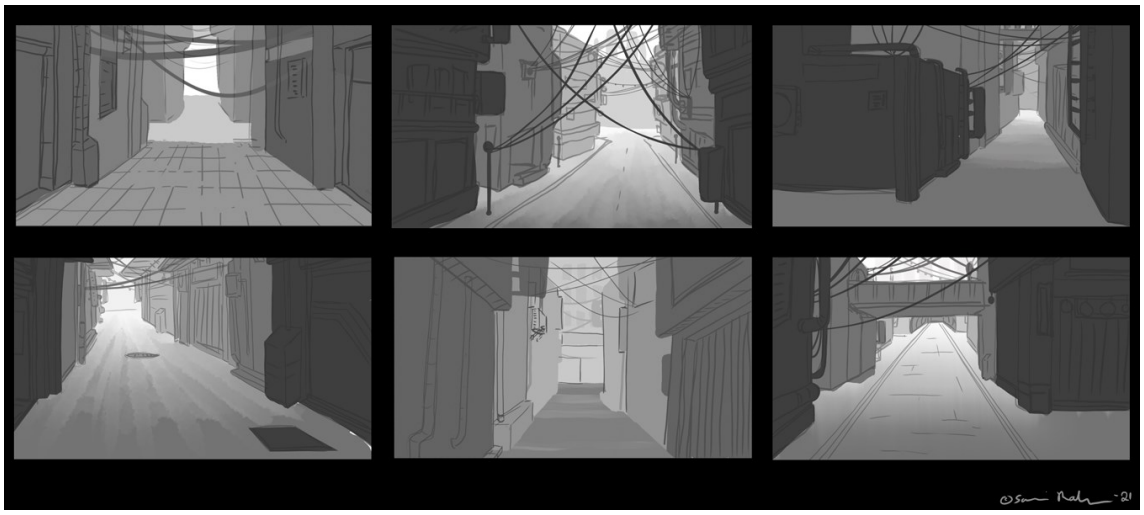
Kuva 3. Referenssikuvakokoelma erilaisista ilmastointilaitteista koottuna Pinterest-tauluun

Referenssikuvat auttavat erityisesti, jos kyseessä on todellinen objekti, joka pitää tuottaa 3D-objektiksi. On tärkeää nähdä mallinnettava objekti ja havaita sen yksityiskohdat ja mitat eri puolilta. Tämä ei usein ole mahdollista, joten referenssikuvat helpottavat luontiprosessia.

2.4 Konseptointi

Ympäristön konseptointi on tärkeä vaihe kolmiulotteisen ympäristön luonnissa. Konseptointi voi käsittää vain nopeita luonnoksia ja hahmotelmia mutta myös loppuun asti vietyä tarkempaa kuvitusta aiheesta. Tarvittaessa myös referenssikuvat voivat korvata konseptoinnin, jos piirtäminen ei ole omaa vahvinta osa-aluetta ja jos käytettävissä ei ole erikseen konseptitaiteilijaa. Konseptikuvat ovat kuitenkin tärkeä tukimateriaali sekä tekijälle itselleen että mahdollisesti muille tiimin jäsenille.

Thumbnail-luonnoksilla on hyvä aloittaa ideointia ja sommittelua ennen varsinaista tarkempaa luonnosta. Thumbnailit ovat nopeita pieniä luonnoksia, joiden avulla on helppoa testata nopeasti erilaisia asetelmia ja kokonaisuuksia, ennen kuin päättää lopullisen muodon. Ne ovat usein harmaasävyisiä, ja niistä on tarkoituksena nähdä nopeasti ympäristön yleisilme. Luonnoksien tekemisen jälkeen valitaan luonnoksista paras, jonka pohjalta voidaan tehdä tarvittaessa parempi konseptityö.



Kuva 4. Opinnäytetyön aiheesta tehtyjä thumbnail-luonnoksia, joissa jokainen pieni kuva on nopea harmaasävyluonnos kadusta ja sen rakennuksista.

Konseptikuvia voi myös käyttää konseptitestaukseen. Konseptitestaus on erityisesti pelialalla käytetty tutkimusmenetelmä, jossa tutkitaan, mihin suuntaan projektia kannattaa viedä testaajien palautteen pohjalta. Projektin alkuvaiheessa kohdeyleisölle näytetään luonnoksia, prototyyppejä, moodboard-kuvia tai muita visuaalista aineistoa ja katsotaan, mistä pidetään ja mistä ei pidetä hyvin rakennetun kyselyn pohjalta. Konseptitestaus auttaa aikaisen vaiheen kehitystä, helpottaa suuria päätöksiä ja näyttää kehittäjille, toimivatko heidän ideansa. (Player Research 2021.)

3 3D-MALLINNUSPROSESSI

Ensimmäiset 3D-mallit luotiin 1960-luvulla Sketchpad-ohjelman avulla, jonka oli luonut Ivan Sutherland. Yhdessä David Evansin kanssa he avasivat yhden ensimmäisistä tietotekniikan osastoista Utahin yliopistoon. Osasto kokosi eri puolilta maata yhteen monia lahjakkaita ihmisiä, jotka auttoivat alaa kehittymään. Eräs Sutherlandin opiskelijoista oli Edwin Catmull, joka toimii nykyään Pixarin ja Walt Disneyn animaatiostudioiden johdossa. (Prus 2016.)

Sutherland ja Evans perustivat myös ensimmäisen 3D-grafiikkaan erikoistuneen yrityksen vuonna 1969 nimellä Evans & Sutherland. Alussa 3D-mallinnusta hyödynnettiin lähinnä televisiossa ja mainoksissa. Ajan mittaan sitä on alettu kuitenkin käyttämään paljon hyvinkin eri osa-alueilla, kuten esimerkiksi peleissä ja visualisoinneissa. (Prus 2016.)

3.1 Peruskäsitteet

3D-mallinnuksessa käytetään tietokoneita luomaan grafiikkaa, joka näyttää kolmiulotteiselta katsojalle. Yksinkertaisimmillaan 3D-mallinnus sisältää geometrisen datan, kuten pisteiden, viivojen ja tasojen, yhdistämistä, mikä luo kolmiulotteisen mallin (Tyler 2021). 3D-mallinnukseen on myös saatavilla erilaisia ohjelmistoja, joista osa on keskittynyt hyvin laajasti 3D-tuotannon eri osa-alueisiin ja osa on erikoistunut tiettyyn osa-alueeseen. Tunnetuimpia ja alalla yleisesti käytettyjä mallinnusohjelmia ovat esimerkiksi Blender, 3ds Max ja Modo.

3D-ympäristöä ja -taidetta tehdessä on tärkeää pitää mielessä, mitkä ovat projektin mahdolliset rajoitteet. Rajoitteet määrittävät, kuinka työskennellä, jotta tekijä voi tuottaa hyvälaatuisia ja käytettäviä 3D-objekteja. Vaikka teknologia kehittyy koko ajan ja vaikka rajoitteet eivät ole enää kovin tiukkoja, niiden mielessä pitäminen auttaa mallintajaa tekemään monipuolisempia tuotoksia ja pelimoottoria toimimaan tehokkaasti. Esimerkiksi mobiilipelillä ja huonevisualisoinnilla voi olla hyvinkin erilaiset vaatimukset. (Silverman 2013.)

Pelialalla ympäristön luovat yhdessä kenttäsuunnittelija ja ympäristögraafikot. Ympäristögraafikot voivat myös keskittyä esimerkiksi pelkästään 3D-mallinnukseen tai teksturointiin. Erityisesti suu-

remmissa yrityksissä ympäristögraafikot voivat keskittyä pienempään osa-alueeseen, mutta monesti pienemmissä yrityksissä yksi työntekijä hoitaa ympäristön luomisen mallinnusohjelmista aina pelimoottoriin asti.

3.2 Työkalujen valinta

3D-mallinnukseen on tarjolla monia ohjelmia, joista osa on vakiintunut peli- ja mallinnusalalla yleiseen käyttöön. Näitä ovat esimerkiksi veistämiseen keskittyvä ZBrush, mallinnukseen käytettävät 3ds Max, Modo, Blender ja Maya, pelimoottorit Unity, Unreal ja CryEngine sekä teskturointiin käytettävät Substance Painter ja Substance Designer. Ohjelmia valittaessa on hyvä pitää mielessä, mitä haluaa tehdä sen sisällä, koska osa 3D-mallinnusohjelmista on keskittynyt laaja-alaisesti kaikkeen 3D-mallinnukseen liittyvään ja osaa voi käyttää vain tiettyyn 3D-mallinnuksen vaiheeseen. Saatavilla on myös sekä maksullisia että ilmaisia ohjelmia. Ilmaiset ohjelmat voivat myös muuttua maksullisiksi, jos yksityishenkilö tai yritys tuottaa 3D-malleja kaupalliseen käyttöön.

Valitsin opinnäytetyön käytännön osuuden toteutukseen ohjelmat, jotka olivat helposti saatavilla. Ohjelmien valintaa vaikutti myös, kuinka tuttuja ne olivat itselleni ja kuinka vakiintuneita ohjelmat ovat 3D-mallinnuksessa. 3D-mallinnukseen ja veistämiseen valitsin Blender-mallinnusohjelman. Se on alkanut nousta maksullisten ohjelmien, kuten 3ds Maxin, tasolle, ja sitä käytetään paljon eritoten indiepelien eli yksityishenkilöiden tai pienien ryhmien suunnittelemien pelien kehityksessä. Kyseinen mallinnusohjelma oli myös tuttu itselleni entuudestaan.

Substance Painter ja Substance Designer valittiin tekstuurien luontiin Adobe Photoshopin ohelle. Ne olivat myös entuudestaan tuttuja ohjelmistoja, sisälsivät laajan kirjaston valmiita materiaaleja ja vaikuttivat olevan erinomaisia ohjelmistoja tuottamaan realistisia tekstuureja. Pelimoottoriksi valikoitui Unity, koska opinnäytteen yksi tutustumiskysymys oli tutkia Unityn High Definition Renderpipelineä ja sitä, mihin kyseinen renderöintiputki kykenee ympäristön tuottamisessa.

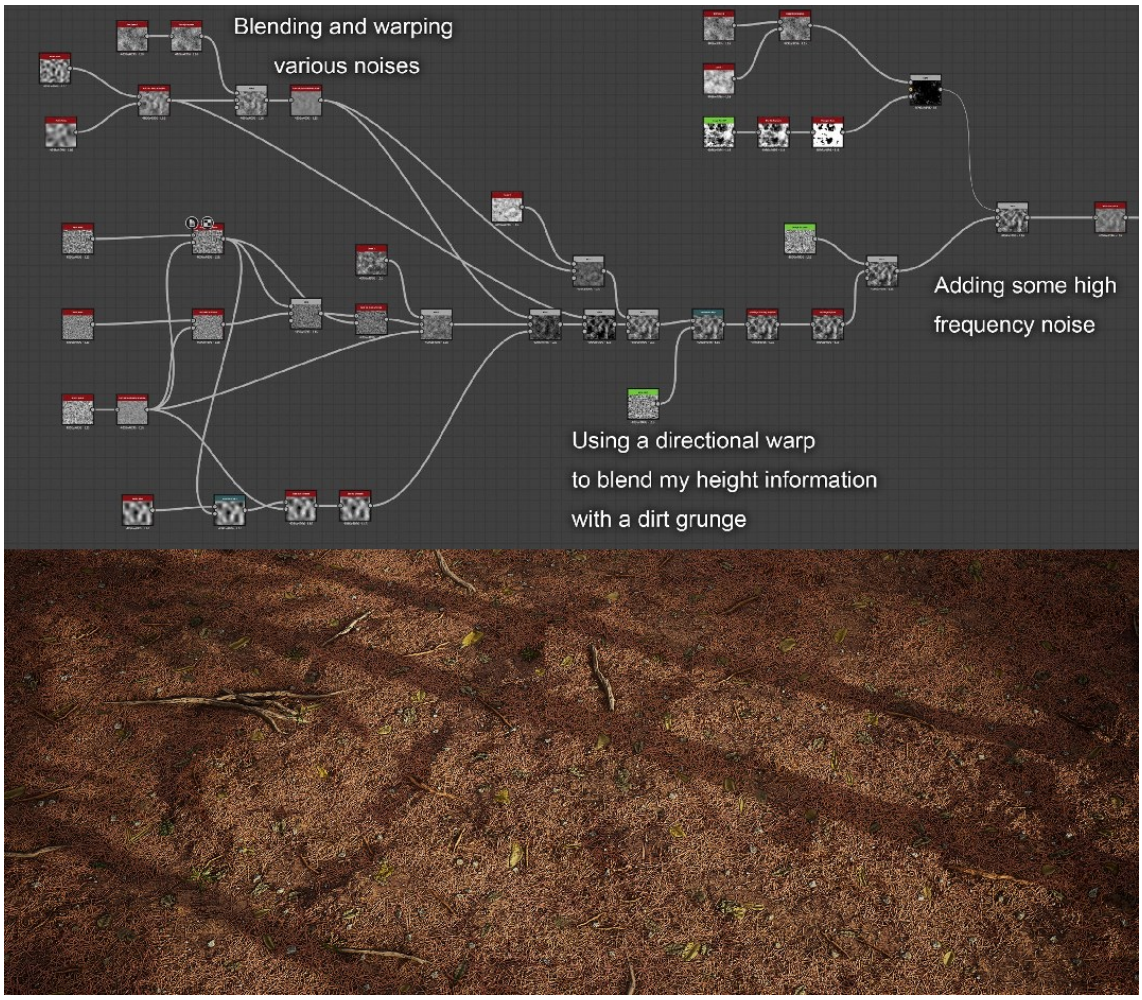
3.2.1 Blender

Blender on ilmainen avoimen lähdekoodin ohjelmisto 3D-tuotantoon. Sen tavoitteena on tarjota kaikille mahdollisuus tehdä 3D-tuotantoa ilmaiseksi hyvällä ohjelmistolla. Blenderin lisenssi tukee ohjelman käyttöä sekä kaupalliseen käyttötarkoitukseen että opetukseen, joten tuotteita voi tehdä

myytäväksi asti ilman lisenssimaksuja. Blender tukee kaikkia 3D-tuotannon eri vaiheita, kuten mallinnusta, riggausta eli animaatioluiden laittamista objektiin, animaatiota, simulaatiota, renderöintiä eli hahmonnusta ja videoeditointia. (Blender 2021.)

3.2.2 Substance Painter ja Substance Designer

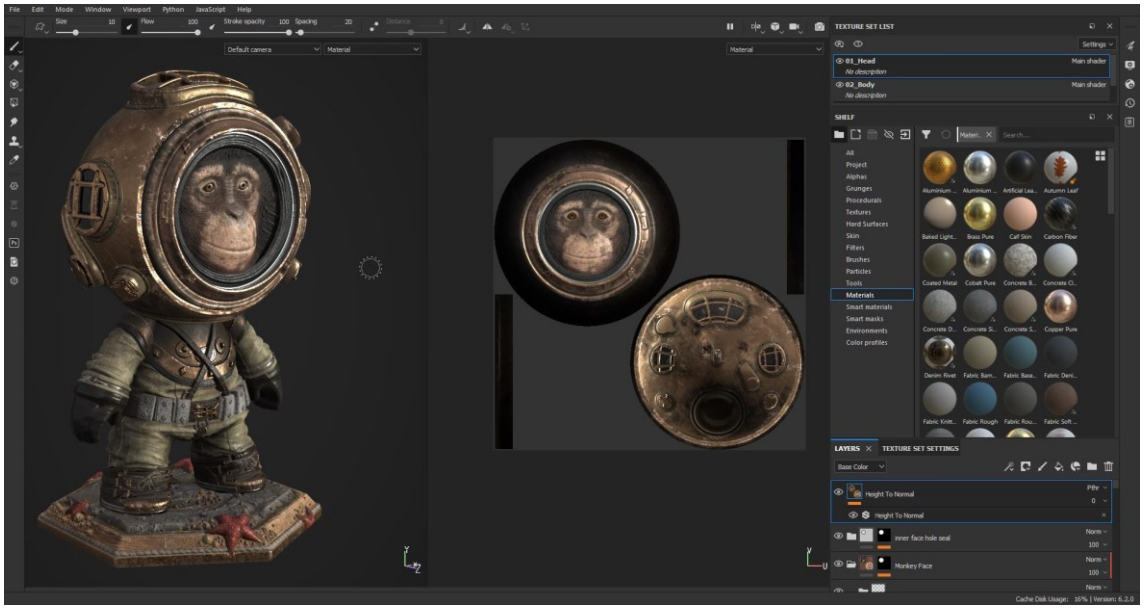
Substance Painter ja Designer ovat Adoben omistuksessa olevia 3D-mallien teksturointi- ja materiaaliluontisovelluksia. Ohjelmissa voi käyttää erilaisia älykkäitä materiaaleja, ja ne mahdollistavat realististen pintojen luomisen nopeasti joko suoraan mallin pinnalle tai noodipohjaisen luontityökalun avulla.



Kuva 5. Substance Designer ja noodipohjainen materiaalin luonti (Hemmens 2021)

Substance Painter käyttää dynaamisia siveltimiä, projisointityökaluja ja partikkeleita tekstuuripintojen luontiin. Substancen yhdistyttyä Adobeen Substancen ja Adobe Photoshopin ohjelmistot myös

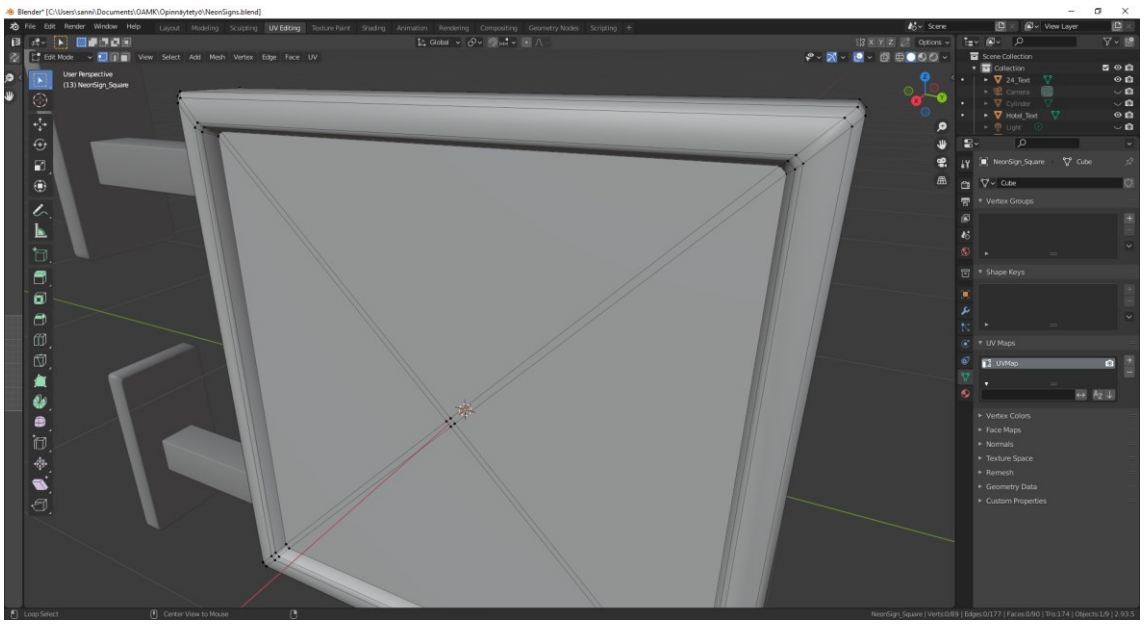
viestivät keskenään saumattomasti. Esimerkiksi Substance Painter tukee Photoshopin sivellinpohjien käyttöä. Älykkäät maskit ja materiaalit helpottavat realististen pintojen luontia, koska ne osaavat ottaa esimerkiksi huomioon kulmien mahdollisen kulumisen. Myös erilaisten todellisten materiaalien käyttäytymistä on helppoa jäljitellä näiden materiaalien avulla. (Adobe 2021.)



Kuva 6. Substance Painterin käyttöliittymä (Adobe 2021)

3.3 3D-mallinnus

Jokainen 3D-objekti käy ensimmäisenä läpi mallinnusvaiheen, kun suunnitteluvaiheen luonnokset ja referenssikuvat on tehty. Kuten aiemmin jo todettiin, yksinkertaisimmillaan 3D-mallinnus on geometrisen datan, kuten pisteiden, viivojen ja tasojen, yhdistämistä, mikä luo kolmiulotteisen mallin katsojalle (Tyler 2021). Tasoista voidaan käyttää myös nimitystä polygoni.



Kuva 7. 3D-malli, joka muodostuu pisteistä, viivoista ja tasoista.

Mallinnusprosessi on hyvin samanlainen eri käyttötarkoituksiin, mutta pelinkehityksen näkökulmasta on tärkeää ottaa huomioon polygonien eli tasojen määrä 3D-objektissa sekä pitää polygonit nelikulmaisina tai kolmioina. Mitä enemmän polygoneja on mallissa, sitä kauemmin kestää renderöidä malli ruudulle. Polygonien määrää voi vähentää esimerkiksi kiinnittämällä huomiota mallin siluettiin ja mallintamalla objektille oleellisimmat muodot. Jokaisella polygonilla pitää olla tarkoitus, eikä tarpeettomia polygoneja kannata olla mallissa. (Silverman 2013.)

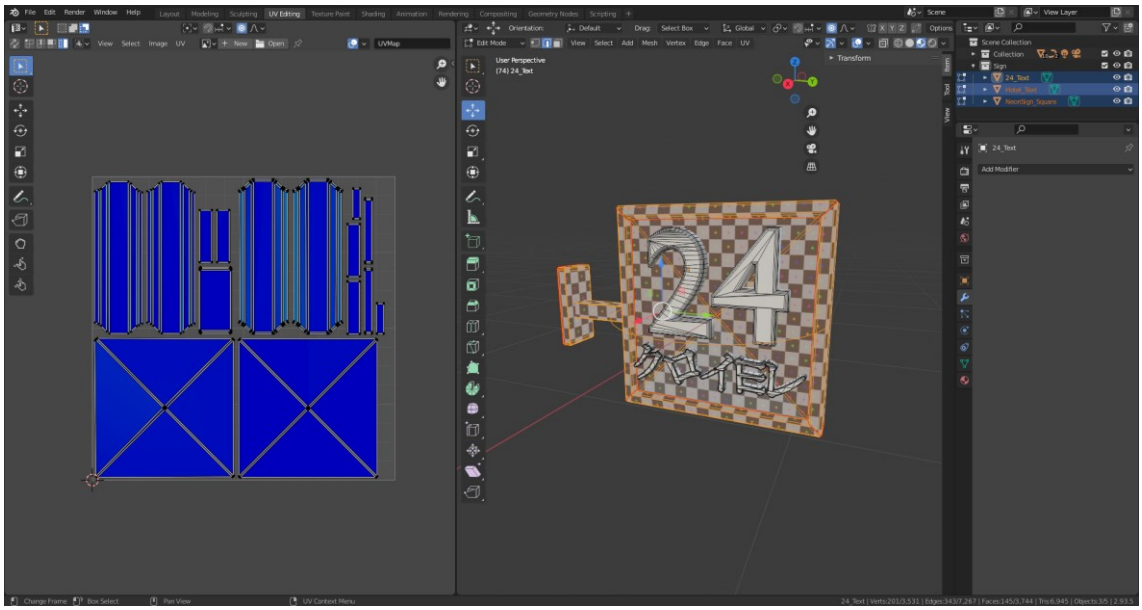
LOD-mallit (Level of Detail) ovat saman 3D-objektin eri versioita. Ne eroavat toisistaan polygonmäärältään, kompleksisuudeltaan ja mahdollisesti myös materiaaleiltaan. Kun kameraa viedään pelimoottorissa kauemmas, voi kamera näyttää yksinkertaisen 3D-objektin. Lähemmäs liikkuessaan kamera voi vaihtaa saman objektin paljon raskaammaksi ja yksityiskohtaisemmaksi versioksi. Prosessi helpottaa renderöintiä ruudulle, kun pelimoottorin ei tarvitse piirtää jokaista yksityiskohtaista mallia, jota hädin tuskin nähdään edes ruudulla. (Silverman 2013.)

Pelimoottoreissa on yleensä automaattiset LOD-luontisysteemit, mutta joskus voi olla hyvä tehdä mallit itse tai käyttää siihen prosessiin tarkoitettuja ohjelmistoja, kuten esimerkiksi Simplygon-optimointiohjelmaa. Näin taataan juuri halutunlaiset LOD-mallit ja mahdollisesti jopa tehokkaammat objektit.

Ympäristön mallinnuksessa kannattaa aloittaa ensimmäisenä isoimmista modulaarisista palasista, joilla kootaan esimerkiksi rakennukset. Tämän vaiheen jälkeen lisätään yksityiskohtaisemmat objektit ympäristöön ja muut mahdolliset yksityiskohdat. Käytännön osuudessa ympäristöä alettiin luoda kokoamalla valmiista palasista isompia kokonaisuuksia, joihin liitettiin myöhemmin pieniä itse luotuja objekteja.

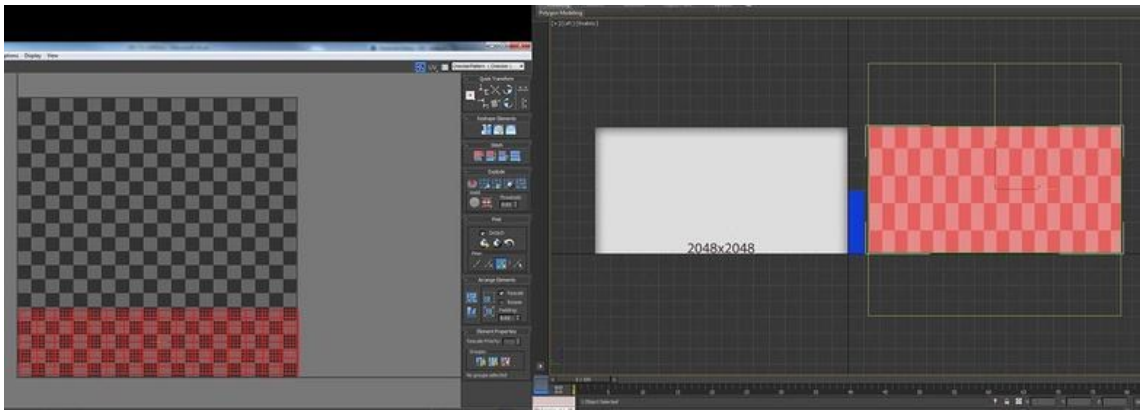
3.4 UV-kartoittaminen

Kun 3D-objekti on mallinnettu, se UV-kartoitetaan teksturointia varten. UV-kartoittamisessa mallinnettuun 3D-objektiin lisätään saumat, ja se leikellään 2D-tasolle palasiksi, eräänlaisiksi kartoiksi, joihin tekstuurit voidaan lisätä myöhemmin. Tätä tekstuurikarttaa voidaan verrata esimerkiksi ompelijoiden kaavoihin.



Kuva 8. UV-kartoittamista Blender-ohjelmiston sisällä. Oikealla näkyy kartoitettava objekti, ja vasemmalla näkyvät sen tasot projisoituina kartalle.

UV-kartoittamisen aikana olisi hyvä huomioida texel-tiheys ja katsoa, että kartta ei veny. Texel-tiheys tarkoittaa tekstuurin resoluutiota ja prosessia, jossa objektilla ja sen tekstuureilla on oikeat mittasuhteet ympäröivään maailmaan (Diaz 2013). Huono texel-tiheys aiheuttaa tekstuurin venymistä ja vääriä mittasuhteita objektin eri osiin tekstuurissa. UV-kartoittamisen aikana palaset kannattaa leikata kohdista, joista saumat eivät näy helposti katsojalle tai joissa on luonnollisesti sauma.



Kuva 9. Texel-tiheys ja venyminen tekstuurissa, jossa oikealla näkyy, kuinka tekstuuri venyy vasemmanpuoleiseen ruudukkokuvioon verrattuna (Diaz 2013).

3.5 Teksturointi ja materiaalit

Teksturoinnissa 3D-objektille luodaan erilaisia värikuvia ja pintamuotoja. Näitä käsinmaalattuja tai eri ohjelmistojen avulla tehtyjä kuvia kutsutaan tekstuurikartoiksi eli tekstuureiksi. Tekstuurityyppejä on monenlaisia riippuen käyttötarkoituksesta ja -ohjelmistosta. Tyypillisiä tekstuurityyppejä ovat esimerkiksi erilaiset värikartat, kuten diffuusi- ja albedokartta, normaalikartta, bump-kartta, displacement-kartta, emissio- eli hohtokartta ja spekulaarikartta. Nämä tekstuurikartat koostuvat kolmesta tai neljästä eri värikanavasta, koska kuvatiedostoja käsitellään yleensä RGB-väritilassa. R on punainen, B sininen ja G vihreä. Yleensä neljäntenä kanavana on mukana myös A eli alfakanava. Unityn High Definition -renderöintiputken tekstuurikartat käyttävät kaikkia näitä värikanavia hyödyksi. Materiaalit koostavat nämä tekstuurikartat yhteen ja näyttävät katsojalle, mikä on objektin värimaailma ja miten sen pinta reagoi ympäristöön.

Tekstuurikartoista diffuusiokartta on yksinkertainen, tasainen väritekstuuri objektissa ilman mitään erikoistehosteita. Se on käytännössä vain objektia kiertävä värillinen kuva. Sen tapaan albedokartta on myös tasainen väritekstuuri. Se eroaa diffuusiokartasta siten, että albedokartta on pelkkä puhdas väri. Diffuusiokartta puolestaan heijastaa lisäksi jonkin verran valaistusta. Käytännössä nämä termit menevät usein ristiin ja albedokartta on enemmän käytössä terminä nykyaikana.

Normaalikartta on yhdistelmä punaista, vihreää ja sinistä, jotka yhdessä ohjelmiston algoritmin avulla muodostavat eräänlaisen harhanäkymän suuremmasta geometriasta ja syvyydestä. (Silverman 2013.) Esimerkiksi pienet pintakohoumat ja saumat voi helposti toteuttaa normaalikartan

avulla ilman, että näitä yksityiskohtia tarvitsisi mallintaa erikseen. Näin säästää myös objektin polygoneja. Bump-kartta on puolestaan eräänlainen harmaasävyinen versio normaalikartasta. Se on normaalikarttaa heikompi ja huonolaatuisempi, minkä vuoksi normaalikartta on nykyään yleisemmin käytetty yksityiskohtien lisäämisessä ja vain harvoin bump-kartan käyttö on enää ajankohtaista.

Normaalikartta voidaan saada myös digitaalisen veistämisen aikana tuotetusta suuren polygonmäärän sisältämästä objektista beikkaamalla. Beikkaamisella tarkoitetaan prosessia, jossa siirretään veistetyn objektin pintatiedot suoraan yksinkertaisempaan 3D-malliin normaalikartan muodossa. Näin yksinkertainen malli voidaan saada näyttämään suuremman polygonimäärän versiolta, jota ohjelmien on huomattavasti kevyempi käsitellä. High poly -malli on objekti, joka sisältää suuren määrän polygoneja, joka on luotu esimerkiksi veistämällä digitaalisesti ja joka voidaan beikata esimerkiksi Substance Painterissa. Tämä prosessi antaa mahdollisuuden työstää Substance Painterin sisällä älykkäitä tekstuureja pelimoottorissa käytettävään malliin.

Displacement-kartta puolestaan muuttaa kokonaan objektin geometriaa ja siluettia ilman, että sitä tarvitsee mallintaa objektiin. Kaikki pelimoottorit eivät kuitenkaan tue displacement-karttoja, jotka ovat myös suhteellisen raskaita renderöidä, minkä vuoksi niitä tulisi käyttää harkiten (Glawion 2021).

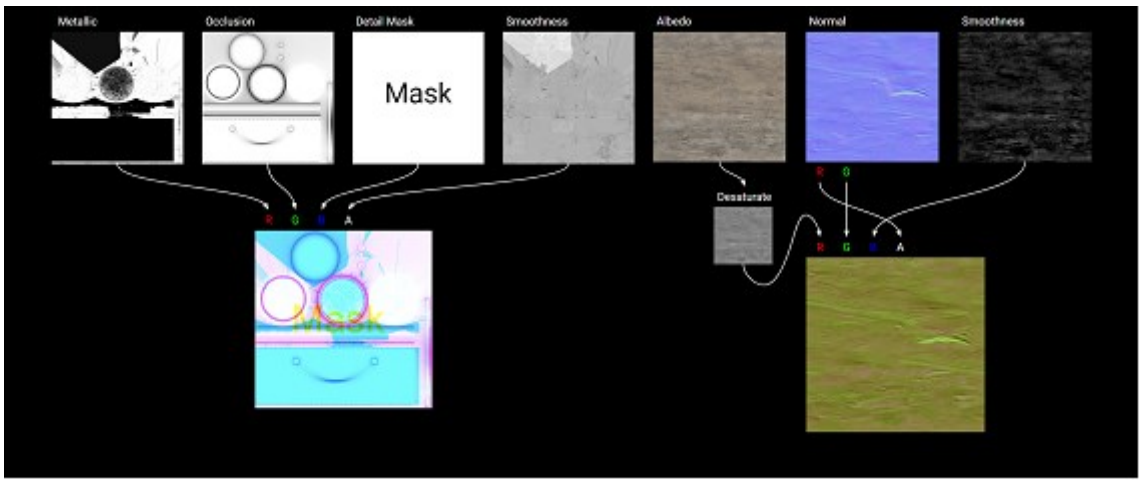
Kuva 10. Eri tekstuurityyppien vaikutukset 3D-objektiin (Glawion 2021)

Decalit ovat tekstuureita, joita voidaan laittaa 3D-pintojen päälle erikseen. Esimerkiksi pelimoottorin sisällä seinään voi lisätä erikseen graffiteja tai luodin jälkiä decal-tekstuurilla ilman, että sitä tarvitsee erikseen tehdä tekstuurikarttaan. Substance Painterissa voi myös laittaa decaleita suoraan mallin päälle erikseen.

Unity ja sen High Definition Renderpipeline käyttävät kanavapakattuja tekstuureja. Tämä tarkoittaa sitä, että yksi tekstuuri sisältää eri tekstuurikarttoja eri värikanavissa. HDRP:n maskikartta sisältää neljä harmaasävytekstuuria, jotka ovat tallennettuina eri päävärien ja läpinäkyvyyden värikanaville. Punaiselle värikanavalle on tallennettu metallitekstuuri, vihreälle okkluusiotekstuuri, siniselle detaljimaski ja läpinäkyvyyskanavaan objektin smoothness- eli kiiltävyyskartta. Lopputulos on yksi värillinen kuva, jota pelimoottori voi käyttää monen kuvan sijasta. (Unity Technologies 2021.)

Maskikartan lisäksi HDRP käyttää detaljikarttaa. Se puolestaan sisältää kaksi harmaasävytekstuuria ja yhden kaksikomponenttisen tekstuurin, joka on materiaalin normaalikartta. Detaljikartassa punaiseen kanavaan tallennetaan desaturoitu albedokartta, vihreään- ja alfakanavaan normaalikartta sekä siniseen objektin smoothness- eli kiiltävyyskartta. (Unity Technologies 2021.)

HDRP:n vaatimat pakatut tekstuurit voi tuottaa usealla eri tavalla. Helpoin tapa on esimerkiksi tallentaa ja tuoda tekstuurit Substance Painteristä ja Designerista HDRP:lle sopivaan muotoon jo valmiiksi. Monet teksturointiohjelmat ovatkin ottaneet nykyään huomioon tuontiasetuksissaan HDRP:n vaatimukset. Jos tekstuurikarttoja on luonut sopiviksi aiempiin renderöintiputkiin, Unity-pelimoottorin sisällä voi muuttaa vanhat tekstuurityypit omaavat materiaalit uuteen sopivaan muotoon ja säätää niitä haluamikseen. Tekstuureja tehdessä tärkeintä on kuitenkin ottaa huomioon, mihin värikanavaan mikäkin tekstuurikartta menee.

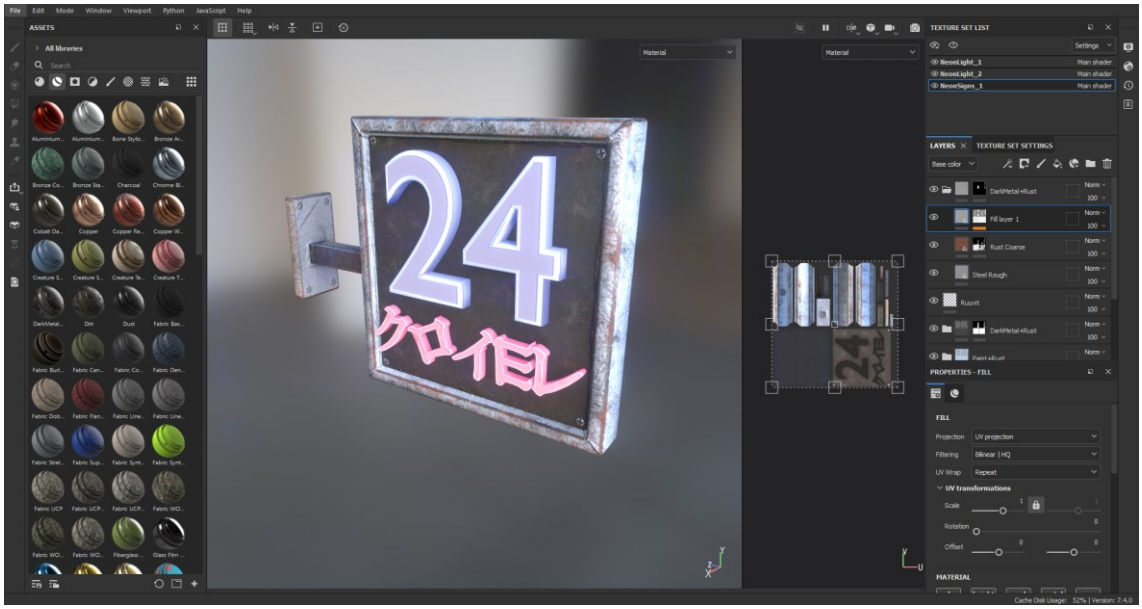


Kuva 11. Maskikartta ja sen neljä eri harmaasävytekstuuria eri värikanavissa sekä detaljikartta (Unity Technologies 2021)

Kun malli tuodaan Substance Painteriin tekstuurien luomista varten, ensimmäiseksi on suositeltavaa beikata malliin suuremman polygonimäärän sisältämä high-poly-objekti. Jos high-poly-versiota

ei ole, voi Painterissa valita käytettäväksi saman mallin, joihin tekstuurit ollaan luomassa. Tämä prosessi mahdollistaa älykkäiden materiaalien käytön ohjelman sisällä. Esimerkiksi materiaali tunnistaa reunat, joihin on voinut syntyä suurempaa kulutusta kuin muualle. Tämä mahdollistaa realistisen tekstuurien luomisen vähällä vaivalla.

Tekstuureja ja materiaaleja luodessa on hyvä pitää mielessä, mikä on objektin tarina ja minkälaisessa ympäristössä objekti tulee olemaan. Esimerkiksi likaisella ja syrjäisellä kujalla sijaitseva metallinen roskapönttö on todennäköisesti pinnaltaan ruosteinen, kulunut paljon ajan saatossa ja likainen.



Kuva 12. Substance Painterin käyttöliittymä, jossa käytännön osuuden yksi objekti on teksturoituna.

Substance Painterissä ja Substance Designerissä tekstuurit voi tallentaa ja viedä Unityn HDRP:lle sopivassa muodossa maskikarttana sekä detaljikarttana niiden luomisen jälkeen. Export-asetuksissa voi päättää, mitä dataa haluaa ottaa mukaan tekstuurien pakkaamiseen ja mitkä tekstuurikartat haluaa luoda.

Opinnäytetyön käytännön osuuden tekstuurit tehtiin Substance Painterissä ja Substance Designerissä, minkä johdosta tekstuurit pystyttiin tallentamaan suoraan HDRP:lle sopivaan muotoon. Substance Painteria käytettiin huomattavasti enemmän kuin Designeria. Sitä käytettiin lähinnä toistuvien tekstuurien, kuten rakennusten ja maassa olevien materiaalien, luontiin Substance Painterin ohella. Kun objektit oli luotu Blenderissä, ne tuotiin Substance Painterin sisälle maalattaviksi ja osaan beikattiin päälle Blenderin sisällä veistetyt ja luodut suuren polygonimäärän versiot. Kun

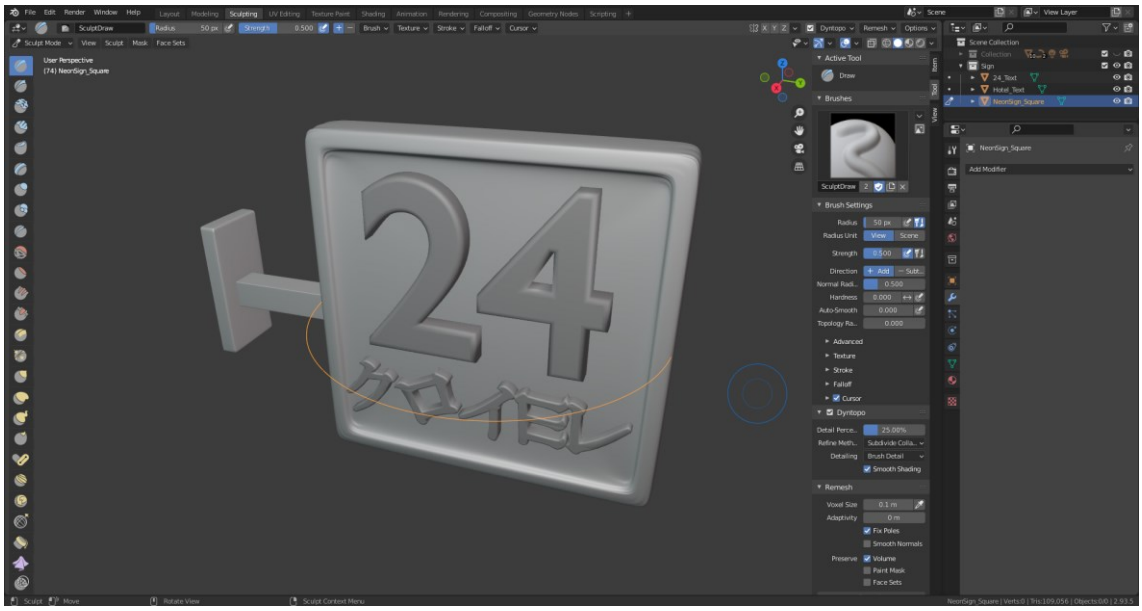
tekstuurikartat oli luotu, tekstuurit ja mallit tuotiin Unity-pelimoottoriin. Tämän prosessin kautta luotiin yksityiskohtia ja uniikkeja objekteja ympäristöön.

3.6 Digitaalinen veistäminen

Digitaalinen veistäminen tarkoittaa 3D-työkalujen käyttöä, jossa muovataan digitaalista savea ja manipuloidaan sitä mieleiseksi, aivan kuten perinteisessä saven muovaamisessa. Digitaaliseen veistämiseen tarkoitetuissa sovelluksissa on samanlaiset työkalut kuin tosielämän veistämisessä. Näitä työkaluja ovat esimerkiksi erilaiset siveltimet, silotus-, työntö- ja vetämistyökalut. Digitaalinen veistäminen eroaa 3D-mallinnuksesta siten, että veistäminen on hyvin orgaanista verrattuna 3D-mallinnuksen viivojen, muotojen ja vektorien muokkaukseen mallin luonnissa (Gaget 2019).

Digitaaliseen veistämiseen on olemassa kahdentyyppisiä ohjelmia. Ensimmäinen vaihtoehto käyttää hyödyksi mesh-pohjaista geometriaa muovailtavassa savessa ja toinen voxel-pohjaista geometriaa. Mesh-pohjaisessa geometriassa muovailtava savi koostuu polygoneista, aivan kuten 3D-mallinnuksessa. Näistä polygoneja vain lisätään ja siirretään saven kaltaisesti digitaalisten työkalujen avulla. (Gaget 2019.)

Voxel-pohjainen geometria koostuu puolestaan täydellisistä kuutioista, joita lisätään ja muovataan digitaalisilla työkaluilla. Voxel-pohjaisella ohjelmalla on nopea mallintaa, tuottaa tarkkaa jälkeä ja visualisoida volumetrista tietoa, kuten luonnon muodostumia 3D-skannauksen avulla. Huonoina puolina voxel-pohjaisessa geometriassa on sen vaativuus tuottaa monimutkaisia objekteja esimerkiksi ilman 3D-skannausta. Nykyiset tietokoneet ovat myös optimoituja renderöimään polygonipohjaista tietoa. (Spatial Team 2019.)



Kuva 13. Blenderin käyttöliittymä digitaalisessa veistämisessä

Digitaalinen veistäminen on ajankohtaista, jos työn tarkoituksena luoda hyvinkin yksityiskohtaista jälkeä välittämättä polygonien määrästä. Veistämisellä luodaan korkearesoluutioisia suuren polygonimäärän omaavia teoksia, joita voi hyödyntää visualisoinnissa, renderöinneissä, animaatioissa tai pelimoottoriin menevien objektien tekstuurien luonnissa. Tätä suuremman polygonimäärän omaavaa versiota objektista kutsutaan High-poly-malliksi. High-poly-mallista voidaan projisoida pintatietoa alemman polygonimäärän omaavaan objektiin tekstuurimuodossa. Joissain tapauksissa objekti ensin veistetään, minkä jälkeen siitä muokataan myöhemmin erillinen objekti pelimoottoriin.

Kovien ja mekaanisten pintojen digitaalisesta veistämisestä käytetään termiä Hard Surface Sculpting. Yleensä digitaalinen veistäminen jaotellaan orgaaniseen ja kovapintaiseen veistämiseen. Yksinkertaisimmillaan orgaaninen veistäminen on elollisten objektien veistämistä ja kovapintaisia rakennettuja objekteja. Veistäminen eroaa näissä lähinnä työkalujen käytössä.

Käytännön osuudessa digitaalisen veistämisen työkaluksi valittiin Blender, koska sovellus oli entuudestaan tuttu ja sen käyttö ei vaadi erillisen kalliin lisenssin ostamista. Koska mallinnus tapahtuu Blenderin sisällä, oli myös luonnollista jatkaa tarvittaessa veistämistä ohjelman sisällä. Veistämistä ei kuitenkaan tehty paljon, koska suurin osa yksityiskohdista luotiin suoraan mallintaessa sekä teksturoidessa Substance Painterissä. Digitaalista veistämistä ja Hard Surface-skulptausta tuli kuitenkin testattua muutamaan objektiin, jotta prosessi ja työkalut tulisivat tutuiksi.

4 UNITY-PELIMOOTTORISSA TYÖSKENTELY

Pelimoottori on ohjelmointikehys, jota käytetään pelinkehitykseen. Pelimoottori antaa kehittäjän esimerkiksi lisätä erilaisia elementtejä ja suorittaa kuvan renderöinnin näytöllä. Lisäksi kehittäjä voi suorittaa skriptejä eli muutaman rivin komentosarjoja sekä lisätä tekoälyä ja muita piirteitä ilman, että niitä tarvitsee ohjelmoida itse. Tämä puolestaan antaa aikaa pelinkehittäjälle luoda uniikkeja elementtejä, kuten 3D-malleja. Osa yrityksistä saattaa kuitenkin luoda omat pelimoottorit, jos yritykset kokevat ne tarpeelliseksi. (Tyler 2021.)

Unity on yksi tunnetuimmista pelimoottoreista Unreal Enginen ohella. Se on yksi helpoimmista pelimoottoreista käyttää sen yksinkertaisen käyttöliittymän vuoksi, ja se tukee pelinkehitystä eri alustoille, kuten Androidille, iOS:lle ja eri konsoleille. Unity-pelimoottori julkaistiin vuonna 2005, ja sen on kehittänyt Unity Technologies. Peruslisenssin käyttö on myös kokonaan ilmaista, joten aloittelijat ja harrastelijat pääsevät hyvin alkuun pelinkehityksen maailmaan sen avulla. (Interesting Engineering 2016.)

Käytännön osuuden toteuttamiseen valittiin Unityn pelimoottorin versio 2020.3.22f1, koska kyseinen versio pelimoottorista on uusin, niin että sillä on pitkän ajan tuki. Lisäksi kyseisestä versiosta löytyi hyvin dokumentaatiota.

4.1 Pelimoottoreiden historia

Pelimoottorit olivat pidemmän aikaa 1980- ja 90-luvuilla lähinnä pelialan yritysten sisäisiä omia tuotoksia, joiden teknologiaa ei annettu ulkopuolelle. Esimerkiksi nykyisin yleisessä käytössä oleva Unreal Engine aloitti talon sisäisenä teknologiana mutta on ajan saatossa muuttunut väliohjelmistoksi. Viimeisinä vuosina yritysten sisäisten pelimoottorien tekemisen kustannukset ovat kasvaneet, joten yritykset ovat alkaneet käyttää jo saatavilla olevia pelimoottoreita. (Ward 2008.) Unity-pelimoottorin kehitys alkoi puolestaan 2000-luvun alussa. (Brodkin 2013.)

4.2 Pelimoottorin käyttötarkoitukset

Pelimoottorien käyttötarkoituksesta tulee ensimmäisenä mieleen pelinkehitys, mutta pelien lisäksi Unity-pelimoottoria voi käyttää moneen muuhunkin visualisointitarkoitukseen. Esimerkiksi autoyritykset, kuljetuspalvelut ja tehdaslinjastot voivat käyttää hyväkseen Unityn reaaliaikaista 3D-teknologiaa. Myös arkkitehtuuriala, insinööritoimistot sekä eri rakennusalan yritykset hyödyntävät Unityn tarjoamia mahdollisuuksia. Pelimoottori tarjoaa valmiudet käyttää AR- ja VR-mahdollisuuksia reaaliaikaiseen operointiin ja yhteistyöhön eri ammattilaisten välillä. (Unity Technologies 2021.)

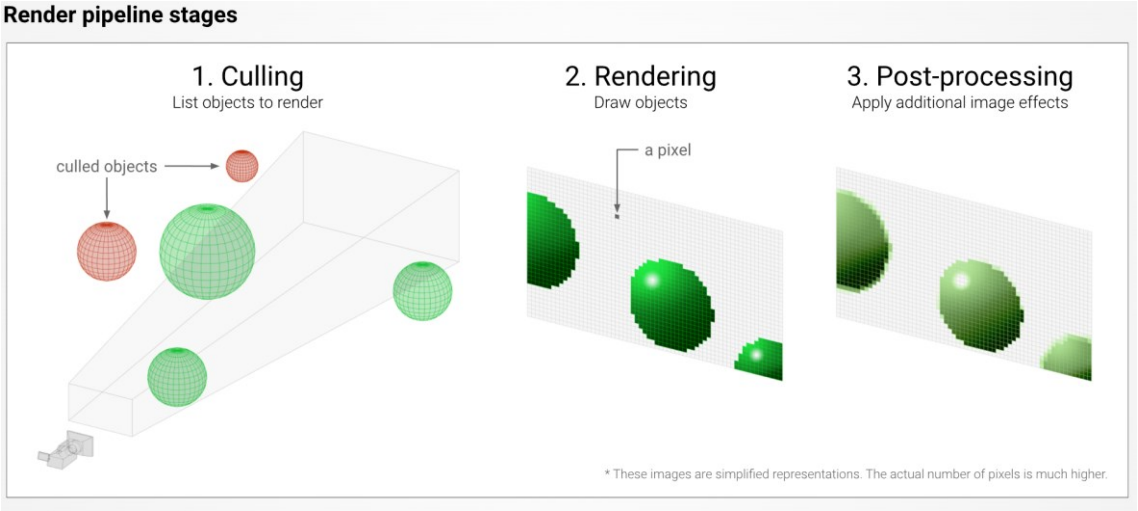
Kuva 14. Unityn High Definition Renderpipelinella tuotettu auton visualisointi (Donzallaz 2020)

Rakennusalan yritys Skanskan projekti on eräs hyvä esimerkki Unityn monipuolisesta käyttökohteesta. Skanska teki yhdessä OutHere-yrityksen kanssa Unity-pelimoottorilla VR-pohjaisen työpaikkaturvallisuusohjelman. Projektin tarkoituksena oli tuottaa realistisia virtuaalitodellisuuden tilanteita osana Skanskan työntekijöiden työturvallisuusharjoituksia ja vähentää sen avulla rakennustyömaalla tapahtuvia vahinkoja sekä tehdä työympäristöstä turvallisempi. VR-kokemus vähensi onnettomuuksia työmaalla ja lisäsi työntekijöiden tehokkuutta. (Unity Technologies 2021.)

4.3 Renderöintiputket

Renderöintiputki on sarja operaatioita, joissa otetaan huomioon pelimoottorissa oleva ympäristö sekä näytetään objektit katsojan ruudulla. Nämä operaatiot ovat culling-operaatio, missä kameran

ruudun ulkopuoliset objektit kytketään pois päältä hetkellisesti, renderöinti eli kuvan tuottamista ruudulle annetun tiedon pohjalta, ja post-processing eli kuvan jälkikäsittely. Renderöintiputkilla on erilaisia ominaisuuksia, ja eri putket soveltuvat eri käyttökohteisiin, kuten peleihin ja eri alustoille. Renderöintiputki onkin hyvä valita projektin alkuvaiheessa, koska sen vaihto kesken projektin voi olla vaikeaa ja tuottaa paljon ylimääräistä työtä. Eri renderöintiputkissa käytetään erilaisia varjostimen ominaisuuksia. Jos renderöintiputken vaihtaa myöhemmin uuteen, vaihto voi johtaa virheisiin, joita ei mahdollisesti voi korjata. (Unity Technologies 2021.)



Kuva 15. Renderöintiputken vaiheet, joista näkyvät esimerkit culling, rendering ja jälkikäsittely (Unity Technologies 2021).

Unity-pelimoottorissa on valittavana kolme erilaista renderöintiputkea. Ne ovat Built-in Render Pipeline, Universal Render Pipeline (URP) ja High Definition Render Pipeline (HDRP). URP ja HDRP pohjautuvat Scriptable Render Pipelineen eli SRP:hen, ja tämä mahdollistaa näiden kahden renderöintiputken muokkauksen omilla C#-skripteillä (Unity Technologies 2021). SRP:n avulla voi myös luoda tarvittaessa oman renderöintiputken.

4.3.1 Built-in Render Pipeline

Built-in Render Pipeline eli Unityn sisäänrakennettu renderöintiputki on sen jo aiemmin käyttämä vanha renderöintiputki. Ennen vuoden 2018 alkua muita renderöintiputkia ei ollut saatavilla Unityssä. Se on yleispätevä renderöintiputki, joka soveltuu vähän kaikkeen. Sillä on kuitenkin rajoitteita verrattuna toisiin Unityn omiin renderöintiputkiin. Sitä ei esimerkiksi voi muokata omilla C#-

skripteillä ja saada sitä kautta haluamaansa persoonallista lopputulosta. Sisäänrakennettu renderöintiputki kannattaa valita, jos haluaa laajan tuen eri alustoille ja vakaan renderöintiputken. (Unity Technologies 2021.)

4.3.2 Universal Render Pipeline (URP)

Universal-renderöintiputki on Unityn tekemä renderöintiputki, joka tunnettiin myös alun perin nimellä Lightweight Render Pipeline (LWRP). Se tarjoaa graafikoille helpomman käyttöliittymän ja antaa käyttäjän nopeasti sekä helposti optimoida grafiikkaa eri alustoilla. Universal Render Pipelineä voi käyttää mobiili-, konsoli- sekä tietokonealustoille suunnatuille ohjelmistoille. Se ei kuitenkaan voi tuottaa yhtä tarkkaa jälkeä kuin esimerkiksi HDRP, eikä se sisällä yhtä kehittynyttä reaaliaikaista varjostusta kuin HDRP:ssä. Se kuitenkin tuottaa optimoidumpaa ja parempilaatuista grafiikkaa kuin Unityn sisäänrakennettu renderöintiputki. (Unity Technologies 2021.)

4.3.3 High Definition Render Pipeline (HDRP)

HDRP eli High Definition Render Pipeline soveltuu AAA-tason peleille, erilaisiin automotiividemoihin, visualisointeihin, arkkitehtuurisiin sovelluksiin sekä muihin käyttötarkoituksiin, joissa tarkoituksena on tuottaa korkealaatuista grafiikkaa. Kyseinen renderöintiputki käyttää fysiikkapohjaista valaistusta ja materiaaleja sekä tukee myös Forward- ja Deferred-renderöintiä. HDRP kuitenkin vaatii toimiakseen sopivaa GPU-laitteistoa, koska se käyttää tietyntyyppistä varjostinteknologiaa. Se on hyvä valinta renderöintiputkeksi, jos projektin tarkoitus on tuottaa korkealaatuista grafiikkaa, käyttää hyödyksi reaaliaikaisia varjostuksia sekä tähdätä hyvän laskentatehon alustoille. HDRP ei kuitenkaan sovellu esimerkiksi mobiili- ja Switch-pelien kehitykseen sen vaatiman laskentatehon vuoksi. (Unity Technologies 2021.)

4.4 Unity ja HDRP

Projektin luonnin yhteydessä on suotavaa valita heti alussa oikea renderöintiputki, jota käytetään. Unity tarjoaa projektin luonnin yhteydessä mahdollisuuden valita HDRP-pohjan, jossa luodaan ja asennetaan projektin aloituksen yhteydessä esimerkiksi jälkikäsittelyefektit ja muut vaadittavat osat kyseiseen renderöintiputkeen. Renderöintiputki on mahdollista vaihtaa myös myöhemmin, mutta mitä myöhemmin vaihdon tekee, sitä todennäköisempää on, että projekti voi rikkoutua. Erityisesti

varjostimet voivat tuottaa päänvaivaa renderöintiputkea vaihtaessa. URP- ja HDRP-projektit eivät ole yhteensopivia, joten renderöintiputkia ei voi myöskään vaihtaa näiden välillä (Unity Technologies 2021).

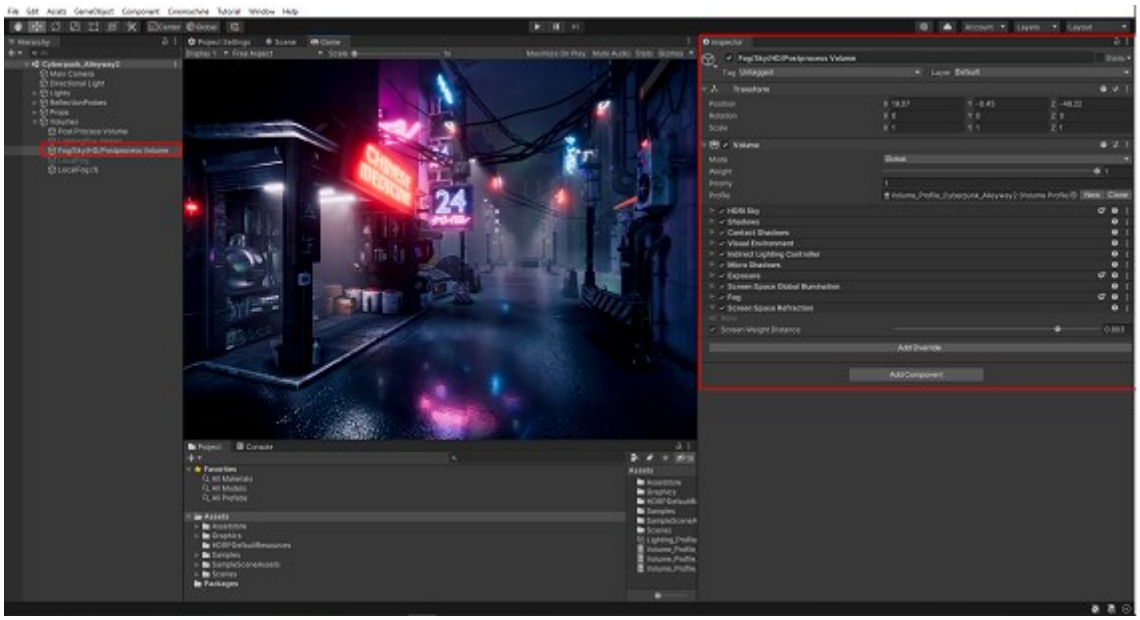
HDRP-projekti luo automaattisesti HDRP-assetin. Sen sisällä voi määrittää projektin globaalit renderöintiasetukset ja voi luoda samalla renderöintiputkelle instanssin. HDRP-assetin sisällä on esimerkiksi valittavissa renderöintityyppi. Näitä renderöintitapoja ovat suora renderöinti (forward rendering) ja viivästetty renderöinti (deferred rendering). Niiden ero on siinä, miten ne laskevat ympäristön valot. Suorassa renderöinnissä renderöintiputki laskee valot kerran jokaiselle materiaalille, ja viivästetyssä renderöinnissä valot lasketaan myöhemmin jokaiselle pikselille sen jälkeen, kun materiaalit on ensin piirretty ilman valaistusta. (Unity Technologies 2021.)

Renderöintitavan valinta riippuu valojen määrästä ja projektin käyttökohteesta. Esimerkiksi suora renderöinti on hyvä, jos valonlähteitä on vähän. Molempia renderöintivaihtoehtoja voi myös käyttää. Yleisesti ottaen Forward-renderöinti on laadultaan parempaa HDRP:ssä, mutta raskaampaa varsinkin silloin, jos valonlähteitä on monta. Deferred-renderöinti voi olla nopeampaa erityisesti, jos ympäristössä on paljon materiaaleja ja monta valonlähdettä. (Unity Technologies 2021.)

HDRP sisältää High Definition Render Pipeline Wizard -osion. Sen voi löytää Unity-pelimoottorissa Windows-valikon alta Render Pipeline -osiossa. Tämän avulla voi testata, että projekti toimii HDRP-ympäristössä ja asetukset ovat oikein. Se myös helpottaa tunnistamaan mahdollisia ongelmia.

4.4.1 Volyymit

HDRP käyttää volyymiviitekehystä (Volume). Näiden avulla voi hienosäätää luodun ympäristön valaistusta ja efektejä. Nämä volyymit voivat olla paikallisia tai globaaleja komponentteja. Paikalliset komponentit näkyvät vain tietyllä alueella ympäristössä, ja globaalit volyymit vaikuttavat koko ympäristön asetuksiin. Jokainen volyymi sisältää avatun scenen eli näkymän ympäristön arvoja, joiden avulla voi säätää asetuksia, kuten sumun väriä ja paksuutta, sekä vaikuttaa ympäristön tunnelmaan. Volyymiin tulee lisätä erikseen viittaus volyymiprofiiliin, jonka avulla lisätään ja muutetaan arvoja näkyvään ympäristöön. Esimerkiksi volyymiprofiiliin voi lisätä varjon asetukset, volumetrisen sumun ja erilaisia jälkikäsittelyefektejä, kuten hohdon. (Unity Technologies 2021.)

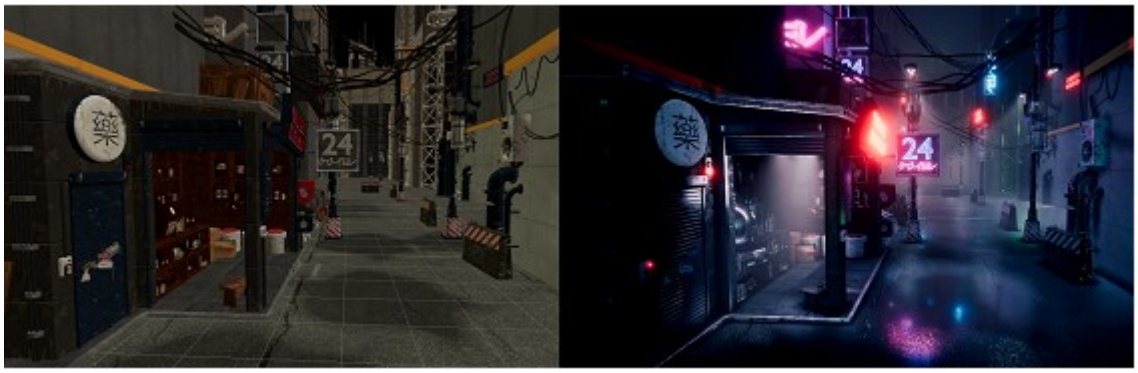


Kuva 16. Volyymi näkymässä, johon on lisätty volyymiprofiiliin erilaisia jälkikäsittelyefektejä, sumua ja valaistukseen liittyviä asetuksia.

4.4.2 Valaistus

Unityssä ja erityisesti HDRP:llä on mahdollisuus luoda hyvinkin realistista valaistusta. Valaistuksella on suuri merkitys ympäristön realistisuudelle ja tunnelmalle, joten valaistuksen säätäminen halutunlaiseksi Unityn sisällä on tärkeää hyvän lopputuloksen kannalta. HDRP käyttää Physical Light -yksikköä (PLU), joka perustuu todellisiin arvoihin. Toisin sanoen valaistus toimii Unity-pelimoottorissa siten, kuin se toimisi oikeassa maailmassa.

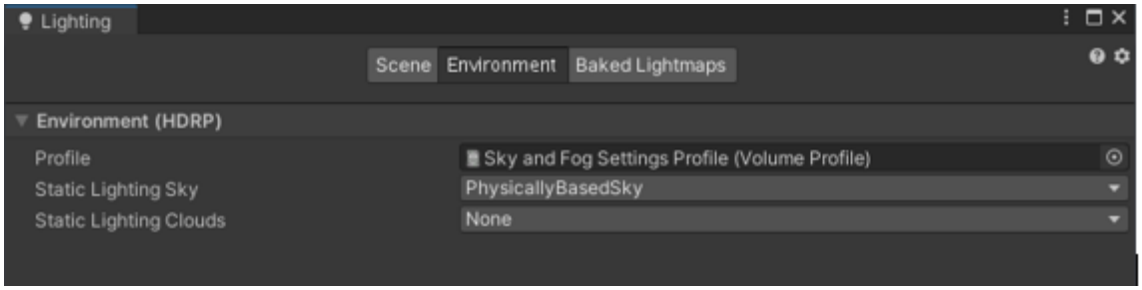
Saadakseen realistisen valaistuksen aikaiseksi Unity ottaa huomioon suoran ja epäsuoran valaistuksen ympäristöstä. Näiden käyttö riippuu siitä, kuinka projektin asetukset säädetään. Valaistusasetuksia voi säätää muun muassa volyymien ja valaistusikkunan kautta. (Unity Technologies 2021.)



Kuva 17. Valaistuksen ja jälkikäsittelyn merkitys ympäristössä ja pelimoottorissa, jossa oikealla näkyy opinnäytetyön valaistu ympäristö ja vasemmalla ympäristö, josta valot ja jälkikäsittelyefektit on poistettu.

Unityssä valaistus voidaan toteuttaa käyttämällä reaaliaikaista valaistusta, beikattua eli valmiiksi tallennettua valaistusta tai näiden kahden yhdistelmää. Reaaliaikainen valaistus toteutuu Unityssä samaan aikaan, kun sovellusta suoritetaan. Beikattu projisoitu valaistus puolestaan suoritetaan etukäteen ja tallennetaan valotietona, joka laitetaan ohjelmaan sen suorituksen aikana. (Unity Technologies 2021.) Reaaliaikaisella valaistuksella voi esimerkiksi liikuttaa valonlähdettä, kuten aurinkoa, ja saada vuorokaudenvaihtelut hyvin näkyviin. Reaaliaikaiset valaistukset voivat kuitenkin olla kalliita koneen suorituskyvylle niiden vaatimien laskutoimitusten vuoksi.

Ympäristön valaistukseen vaikuttavat monet eri lähteet ja asetukset Unityssä ja HDRP:ssä. HDRP käyttää ympäristön valaistuksessa kahta osaa, jotka ovat Visual Environment sekä valaistusympäristö (Environment lighting). Valaistusympäristöä kontrolloidaan Environment (HDRP) -välilehdeltä Unityn valaistusikkunasta. Visual Environment -osiota puolestaan kontrolloidaan volyymiviitekehyksen avulla. Sitä käytetään kontrolloimaan, minkä tyyppistä taivasta käytetään ympäristössä ja miltä taivas näyttää. Valaistusympäristö puolestaan kontrolloi, kuinka taivas vaikuttaa ambient- eli tunnelmavalaistukseen. HDRP:ssä voi valita kolme erilaista taivaan tyyppiä valaistukseen. Nämä ovat HDRI Sky, Gradient Sky ja Physically Based Sky. (Unity Technologies 2021.)



Kuva 18. Unity-pelimoottorin valaistusikkuna

Yhtenä tärkeänä osana ympäristön valon toimivuutta ovat heijastuksen säädöt. HDRP käyttää kolmea erityyppistä tekniikkaa laskeakseen, kuinka heijastukset toimivat. Nämä ovat Screen space -heijastukset, reaaliaikaiset ja beikatut reflection probe -heijastukset sekä taivaan heijastukset. Screen space -heijastukset ovat raskaita, reaaliaikaiset reflection probe -heijastukset ovat suhteellisen raskaita, ja loput ovat kevyitä tekniikoita. Nämä lasketaan edellä mainitussa järjestyksessä. Screen space käyttää hyväkseen volyymiviitekehystä, reflection probe -objektit pitää sijoittaa ympäristöön käsin, jotta ne toimivat, ja taivaan heijastuminen riippuu taivaan tyypistä sekä näkyy viimeisenä mahdollisena heijastuksena objektissa. Screen space ottaa huomioon vain ruudussa näkyvän ympäristön, joten saadakseen näkyviin ruudun ulkopuoliset heijastukset ympäristön objekteihin, reflection probe -objekteja olisi hyvä käyttää ympäristössä realistisemman lopputuloksen saamiseksi. (Unity Technologies 2021.)

HDRP mahdollistaa myös Ray Tracing -ominaisuuden käytön Unity 2019.3 -versiosta alkaen. Kyseessä on tekniikka, joka realistisesti simuloi valon käyttäytymistä ja heijastumista sekä mahdollistaa tiedon tallentamisen esimerkiksi valaistuksesta, joka ei näy ruudulla. Käytettävän laitteiston näytönohjaimen tulee myös olla suhteellisen uusi alkaen NVIDIA GeForce 20 -sarjasta, jotta ominaisuutta voi hyödyntää kunnolla. Sillä voi tuottaa hyvinkin tarkkaa fotorealistista jälkeä. (Unity Technologies 2021.) Käytännön osuudessa ei käytetty Ray Tracing -ominaisuuksia raskaiden laitteistovaatimuksien vuoksi.

4.4.3 Jälkikäsittelyefektit

Unity tarjoaa valmiiksi useita visuaalisia jälkikäsittelyefektejä, joilla sovelluksen ulkoasua voi kohentaa hetkessä. Näitä efektejä voi käyttää joko simuloimaan fyysistä kameraa ja videon ominaisuuksia tai vaihtoehtoisesti luomaan näyttäviä grafiikoita (Unity Technologies 2021).

Sisäänrakennettu renderöintiputki ei sisällä valmiiksi jälkikäsittelyefektejä, vaan siihen pitää ladata erikseen paketti. URP käyttää omaa jälkikäsittelyratkaisua, joka asentuu valmiiksi, kun projektia luodessa valitaan URP-pohja. HDRP käyttää samaan tapaan omaa jälkikäsittelyratkaisua ja asentuu myös projektiin, kun valitsee HDRP-pohjan projektia luodessa. Mahdollisia lisättäviä jälkikäsittelyefektejä ovat esimerkiksi ambient occlusion eli ympäristövarjostus, anti-aliasing eli sahalaitaisuuden poisto, auto exposure eli automaattivaloitus, bloom eli hohto, väriaberraatio, sumu, syvyysterävyysalue, erilaiset väriarvojen säädöt ja valkotasapaino. (Unity Technologies 2021.)

Kuva 19. Jälkikäsittelyefektien vaikutukset ympäristöön. Vasemmalla on ympäristö ilman tehosteita ja oikealla sama tila erilaisilla jälkikäsittelyefekteillä. (Unity Technologies 2021.)

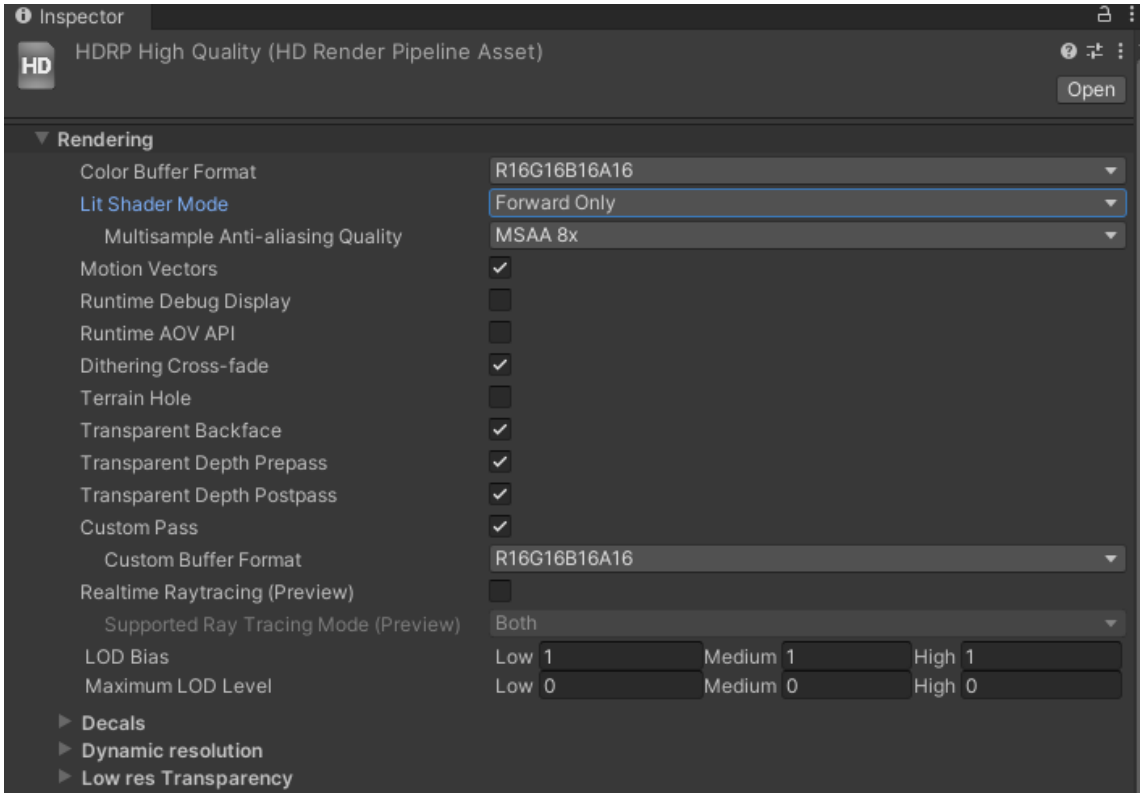
HDRP:ssä jälkikäsittelyefektit luodaan lisäämällä volyymikomponentti ja siihen volyymiprofiili, johon voi sen jälkeen lisätä erilaisia efektejä. Globaali-asetuksella kaikki efektit näkyvät kameran kautta. Paikallisia efektejä voi luoda, jos haluaa näyttää tiettyjä efektejä vasta, kun kamera saapuu tietylle alueelle.

4.5 Ympäristön rakentaminen

Kun HDRP-projekti on luotu Unity-pelimoottorissa, ympäristöä voi alkaa kokoamaan. Ensimmäisenä luodaan scene eli näkymä Unityssä. Kun Unityn ja projektin avaa ensimmäistä kertaa, projektissa on valmiina jo ensimmäinen nimetön näkymä. Uuden näkymän voi luoda valikosta, missä Unity tarjoaa HDRP-projektissa vaihtoehdoksi sisätila- tai ulkotilapohjan tyhjien pohjien ohella. Näihin valmiisiin pohjiin on jo valmiiksi säädetty jonkin verran volyymiasetuksia. Käytännön osuudessa valittiin Basic Outdoor (HDRP) -näkymäpohja, jota lähdettiin työstämään eteenpäin. Tämä pohja sisälsi valmiiksi jo esimerkiksi auringonvalon ja Fog and Sky HD -volyymin. Taivaan tyypiksi valittiin HDRI-taivas.

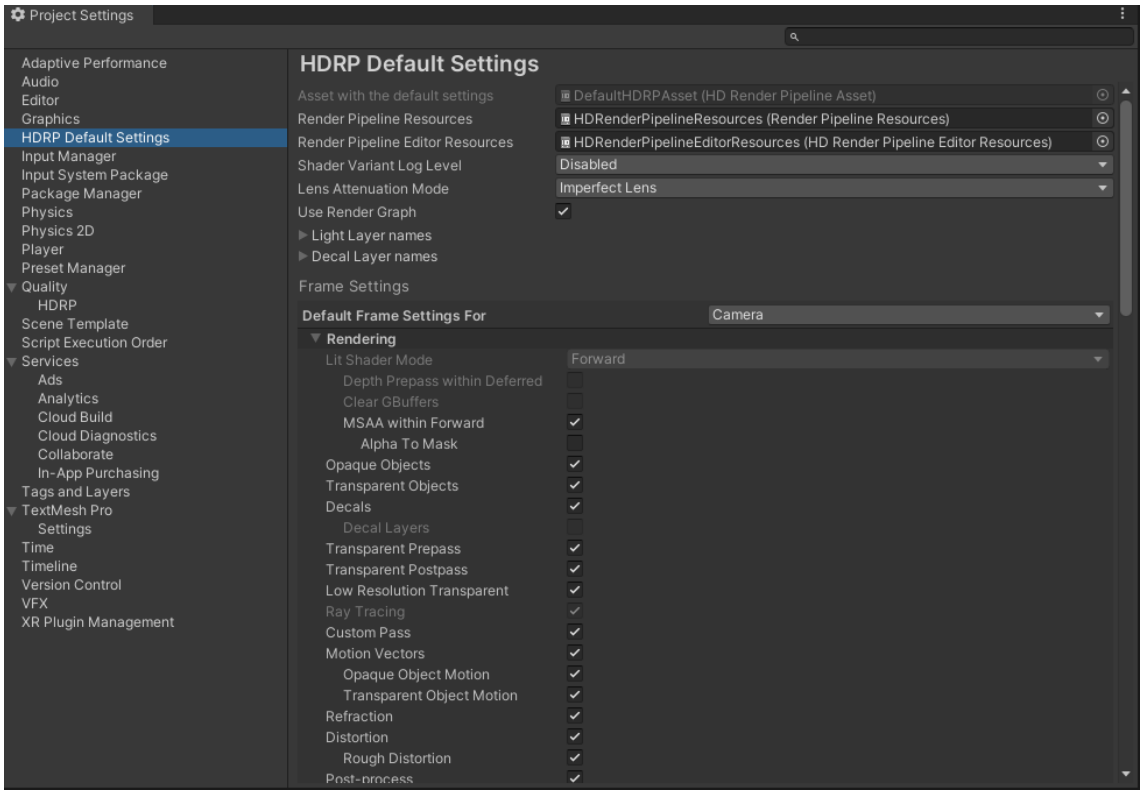
Tämän jälkeen valitaan renderöintitapa HDRP-assetin Rendering-osiosta. Samalla voi valita päälle Multisample Anti-aliasing Quality -vaihtoehdon, jos renderöintitapa antaa valita sen. Multisample Anti-aliasing (MSAA) tarkoittaa kuvan objektien reunojen vääristymisen korjaamista tehokkaalla mutta raskaalla menetelmällä, jossa jokainen pikseli objektin reunan ympärillä kootaan useista näytteistä (Unity Technologies 2021). Jos valitsee tämän prosessin, se pitää vielä erikseen kytkeä päälle Unityn projektiasetuksien HDRP Default -kohdasta Default Frame Settings -osion alta sekä kameralle että valoille. Valitsin projektiin Color Buffer Format -asetukseen vaihtoehdon

R16G16B16A16, renderöintitavaksi Forward-renderöinnin, ja kytkin päälle MSAA-vaihtoehdon. Color Buffer Format -asetus tarkoittaa, missä muodossa HDRP renderöi kuvaruudun, ja R16G16B16A16-asetus mahdollistaa alfakanavan käytön renderöinnissä (Unity Technologies 2021). Valitsin myös HDRP-assetin valaistusosiosta Screen Space -vaihtoehdon. Nämä asetukset antoivat hyvän ja laadukkaan lopputuloksen, mutta ne saattavat olla liian raskaita mahdolliseen pelikäyttöön. Visualisointiin ne vaikuttivat soveltuvan hyvin.



Kuva 20. HDRP-assetin asetukset, joissa voi valita esimerkiksi renderöintitavan.

Ruutuasetuksissa (Frame Settings) voi säätää, miten HDRP renderöi kameran, reaaliajan heijastukset, beikatut heijastukset ja kustomoidut heijastukset. Tarvittaessa näitä asetuksia voi myös säätää erikseen jokaiselle komponentille, kuten esimerkiksi näkymän kameralle. (Unity Technologies 2021.) Käytännön osuudessa ei säädetty erikseen komponenteille ruutuasetuksia, vaan kaikki tarvittava säädettiin alustavasti projektiasetusten sisällä.



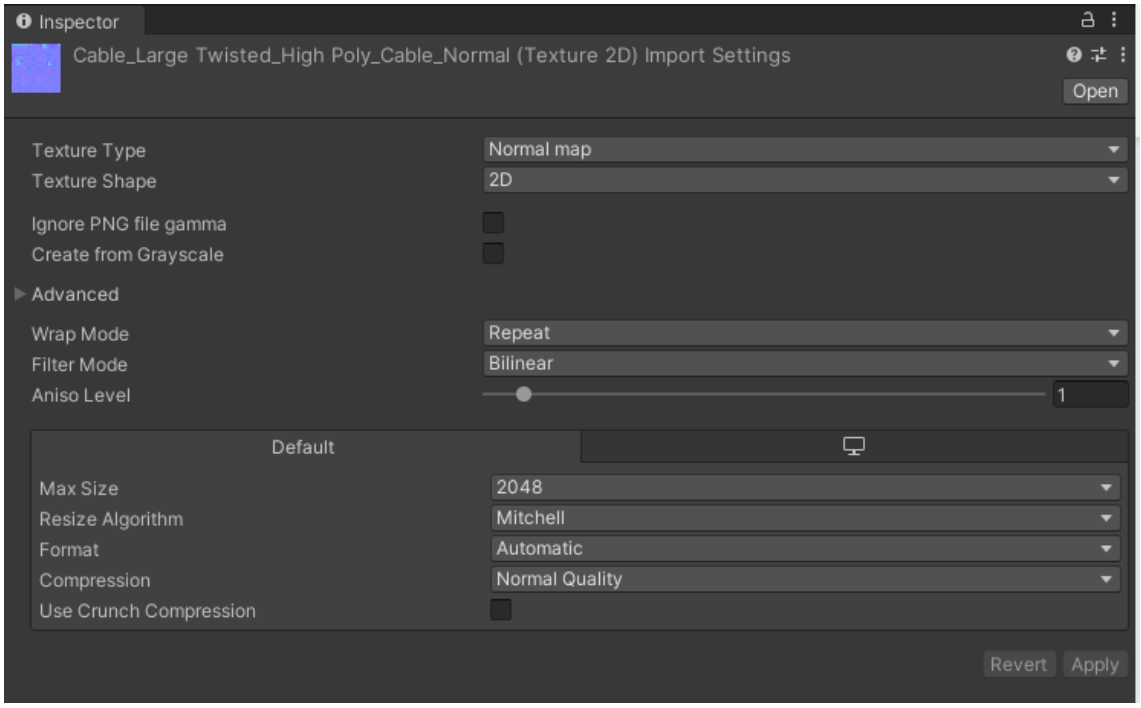
Kuva 21. Projektin ja HDRP:n asetukset, joissa voi valita ruutuasetukset kameralle.

Projektin alussa on suositeltavaa tarkistaa, ovatko kaikki asetukset kunnossa HDRP Wizardissa. Lisäksi sieltä kannattaa seurata, ilmeneekö mahdollisia ongelmia projektin aikana. HDRP Wizard osaa tunnistaa hyvin virhetilanteita kuten esimerkiksi sen, miksi jokin asetus ei välttämättä toimi ympäristössä. Muita mahdollisia virheitä voi etsiä Renderpipeline Debug -osiosta, joka löytyy Windows-valikon ja Renderpipeline-otsikon alta, mistä löytyy myös HDRP Wizard.

Kun taivasta ja efektejä alkaa lisäämään ympäristöön, on hyvä tuoda objekteja jo hyvissä ajoin näkymään ja muovaamaan ympäristöä valmiimmaksi esimerkiksi sijoittamalla ympäristöön yksinkertaisia valmiita muotoja. Tämä auttaa hahmottamaan paremmin, miten mikäkin asetus vaikuttaa. Todennäköisesti asetuksia joutuu säätämään myös myöhemmin, mutta alustavat heijastukset ja valonlähteet auttavat hahmottamaan tarvittavia asetuksia.

Kaikki objektit ja tekstuurit tuodaan projektiin liittämällä tai raahaamalla ne projektikansion juureen. Niille kannattaa luoda oma kansio niiden tyypin mukaan. Unity tukee 3D-malleista .fbx-, .dae-, .dxf- ja .obj-muotoja (Unity Technologies 2021). Tämän jälkeen objektit voi raahata avattuun näkymään ja lisätä ne ympäristöön haluamilleen paikoille. Tekstuurit kannattaa tuoda .png- tai .jpg-muodossa.

Tekstuureista normaalikartat pitää käydä muuttamassa sen omista asetuksista normaalikarttatyyppisiksi. Kun tekstuurikartat on tuotu projektiin, niille voi luoda materiaalin, johon ne lisätään.

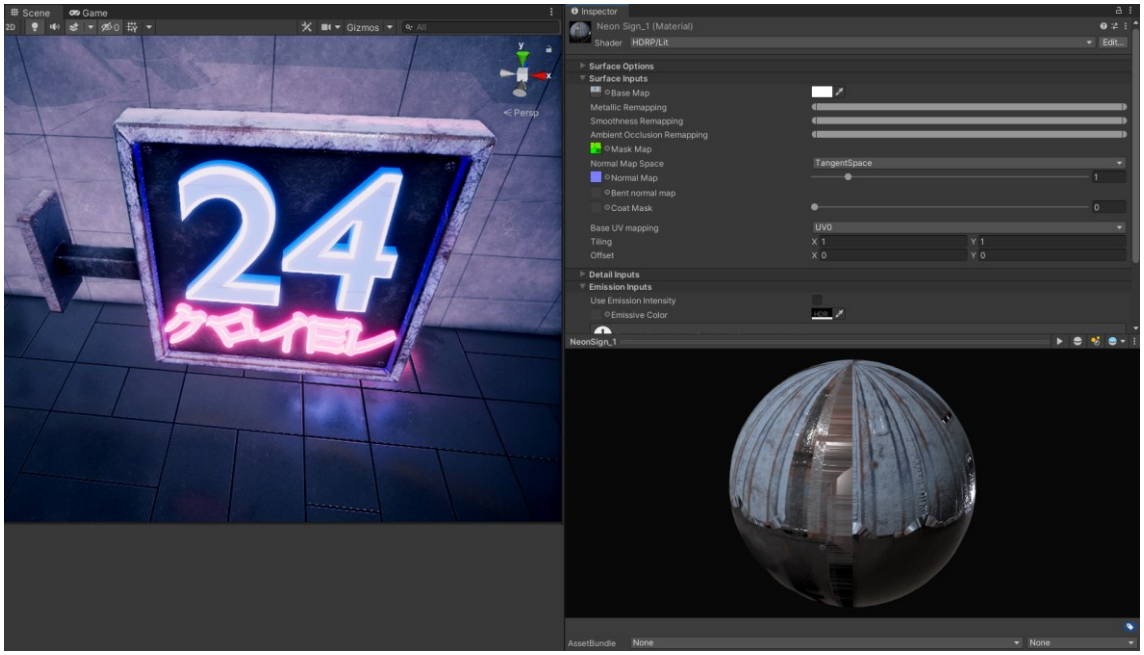


Kuva 22. Normaalikartta, jossa sen tyyppi on vaihdettu normaalikartaksi default-vaihtoehdon sijasta.

Unity-pelimoottorin HDRP käyttää omia varjostimia ja materiaaleja. Tämän vuoksi vanhan tyyppiset materiaalit ja tekstuurit pitää joko päivittää pelimoottorin sisällä tai tuoda suoraan oikeassa tiedostomuodossa. Vanhat materiaalit ja tekstuurit voi päivittää Edit-valikon alta Renderpipeline-osiosta. Nämä muutetut materiaalit voivat vaatia jonkin verran säätöä niiden omista asetuksista. Materiaaleja päivittäessä kannattaa myös muistaa, että vanhat tekstuurikartat muuttuvat suoraan päivitetyiksi ja ettei niitä voi muuttaa takaisin entiselleen. Helpointa onkin tuoda tekstuurikartat suoraan oikeassa muodossa esimerkiksi Substance Painterin kautta ja luoda niille suoraan oikeantyyppinen materiaali.

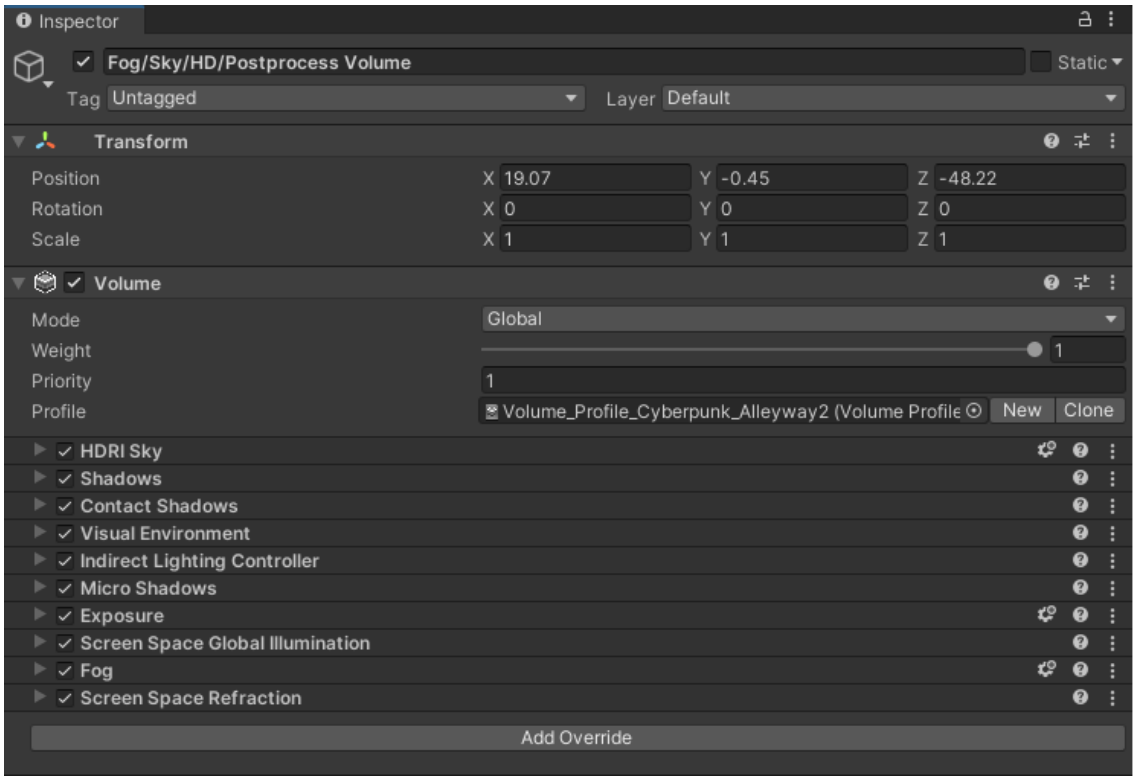
Kun materiaalit ja tekstuurit ovat oikeanlaisia, materiaali voidaan raahata objektin päälle ympäristöön, tai se voidaan sijoittaa objektin asetuksissa oletusmateriaaliksi, jolloin se saadaan näkymään ympäristössä objektissa. Käytännön osuudessa itse tehtyihin objekteihin tuotiin tekstuurikartat suoraan oikeassa muodossa, mihin luotiin materiaalit Unityn sisällä. Osaan ilmaisista objekteista piti kuitenkin päivittää tekstuurikartat oikeaan muotoon Unityssä. Suurin osa ilmaisista tekstuureista,

materiaaleista ja objekteista ladattiin Textures.com-sivustolta ja Unityn Asset Store -kauppapaikalta.



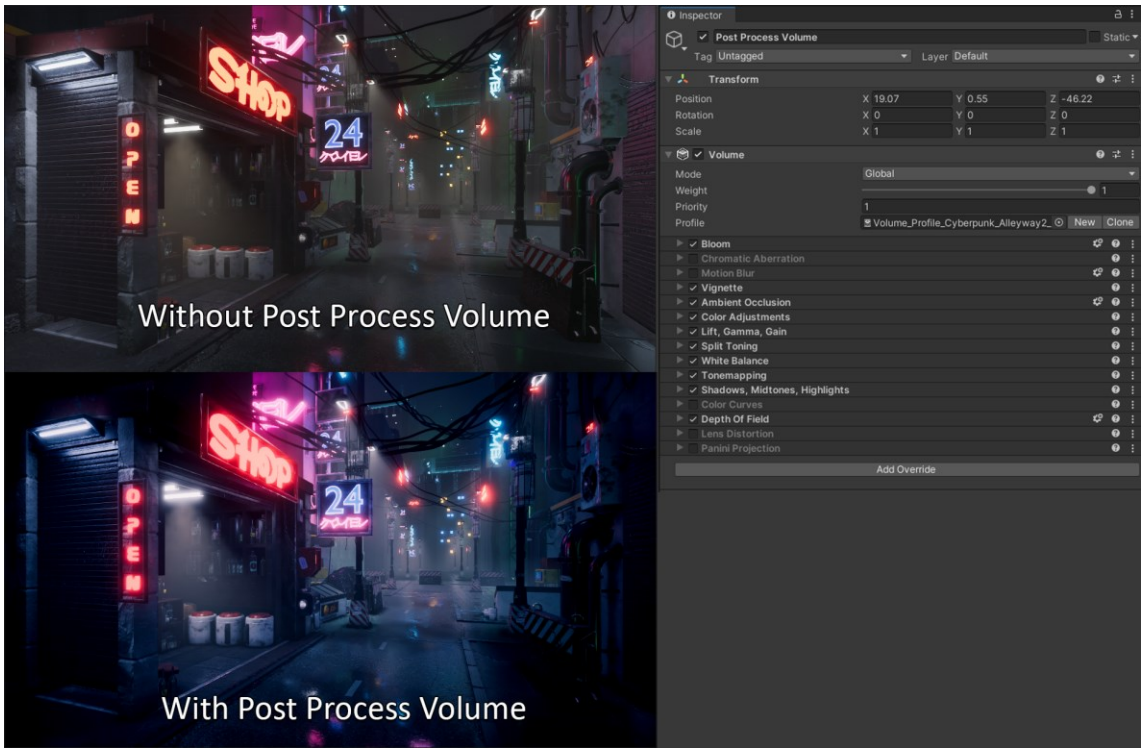
Kuva 23. 3D-Objekti ja HDRP-materiaali, johon on tuotu tekstuurikartat.

Taivasta, sumua, valaistusta ja jälkikäsittelyefektejä voidaan alkaa säätämään lisäämällä volyymikomponentti. Se kannattaa lisätä tekemällä tyhjä volyymi näkymään, jos näkymän luonnin yhteydessä ei volyymia tullut automaattisesti. Volyymissa vaikutusalueeksi valitaan globaali ja sille tehdään uusi volyymiprofiili. Tähän volyymiprofiiliin lisätään sen jälkeen Visual Environment-, Fog-, taivaan tyyppi- ja muita override-lisäyksiä. Samaan profiiliin voi lisätä myös jälkikäsittelyefektejä tai luoda niille oman volyymin, johon lisätään uusi volyymiprofiili. Pelimoottorin Gameobject-osiosta voi hakea erilaisia volyymejä, joihin on laitettu valmiiksi esiasetuksia, kuten taivaan tyyppi ja sumu.



Kuva 24. Volyymi, johon on asetettu volyymiprofiili ja erilaisia lisäyksiä, kuten sumu, valaistus ja taivaan tyyppi.

Visual Environment -lisäyksen omaava volyymiprofiili lisätään tämän jälkeen valaistusikkunassa ympäristöosioon ja valitaan taivaan tyyppi. Käytännön osuudessa volyymiprofiiliin lisättiin Visual Environment -osiossa taivaan tyypiksi HDRI-taivas, johon haettiin kuva Polyhaven.com-sivustolta. Sen lisäksi tehtiin erillinen volyymi ja volyymiprofiili jälkikäsittelyefektejä varten. Jälkikäsittelyefekteistä lisättiin lopulliseen ympäristöön ympäristövarjostus, vinjetti eli reunoilta häivytys, automaattivalaistus, hohde, erilaisia väriasetuksia sekä valkotasapaino. Jotta kaikki väriarvot saatiin näyttämään paremmilta jälkikäsittelyssä, valittiin myös Tonemapping-lisäys, johon laitettiin päälle ACES-vaihtoehto. ACES-vaihtoehto tekee väreistä elokuvamaiset ja vaikuttaa suoraan värien sävyyn ja saturaatioon (Unity Technologies 2020).



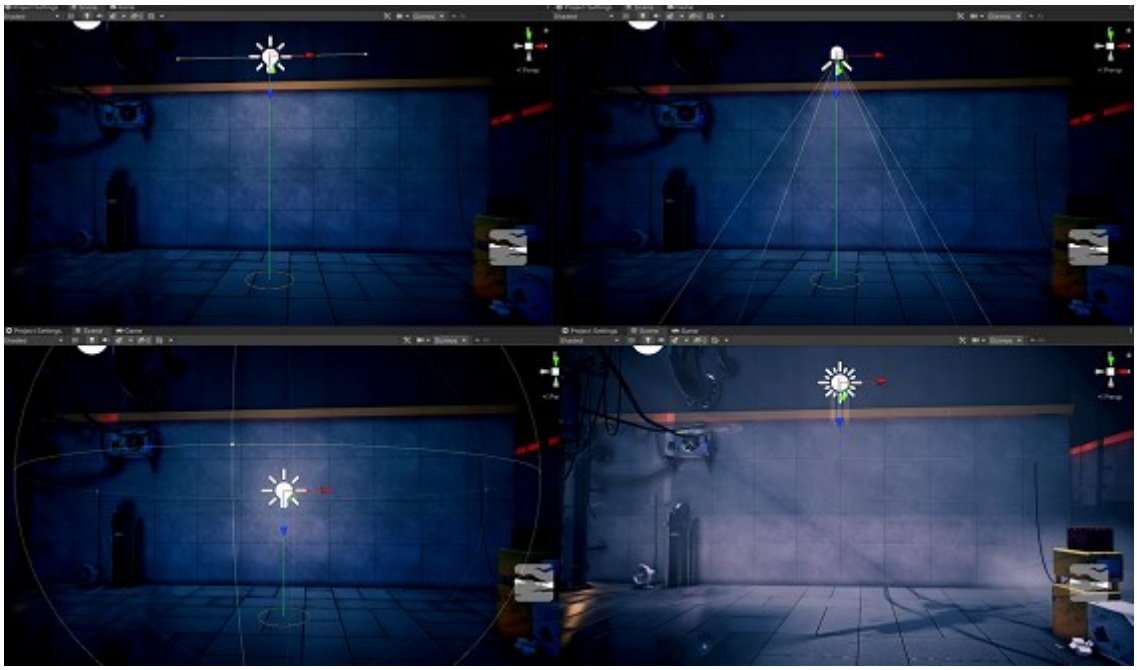
Kuva 25. Jälkikäsittelyefektien vaikutus sekä volyymikomponentti ja -profiili erilaisilla säädöillä

Jälkikäsittelyefektien lisäämisen jälkeen voi siirtyä kameran asetuksiin. Tarvittaessa kameralle voi luoda omat ruutuasetukset, jos haluaa käyttää poikkeavia asetuksia verrattuna muihin mahdollisiin kameroihin. Sen lisäksi kamerasta kannattaa käydä säätämässä kameran fysikaalisia asetuksia, kuten sulkimen nopeutta, ISO-lukemaa ja aukkoa. Fysikaalisista asetuksista säädetään kameran asetuksia aivan kuin tosielämässä.

Kameraan on suositeltavaa käydä lisäämässä Post Anti-aliasing eli sahalaitaisuuden poisto. Se sisältää vaihtoehdot Fast Approximate Anti-aliasing (FXAA), Temporal Anti-aliasing (TAA) ja Sub-pixel Morphological Anti-aliasing (SMAA). Näistä FXAA on kevyt suorittaa, mutta laatu ei ole niin hyvä. TAA ja SMAA ovat molemmat suhteellisen raskaita. Näistä TAA tuottaa tarkempaa laatua kameran liikkuessa mutta voi olla hyvinkin raskas koneelle ja voi mahdollisesti tuottaa ylimääräisiä artefakteja ruutuun. SMAA tuottaa tarkempaa jälkeä kuin FXAA, mutta se ei tuota niin tarkkaa lopputulosta kuin TAA kameran liikkuessa nopeasti. Näiden ohella voidaan HDRP-assetiin valita myös Multi-sample anti-aliasing (MSAA) -vaihtoehto, joka voi toimia yhdessä edellä mainittujen efektien kanssa. Se tuottaa tarkempaa jälkeä kuin aiemmin mainitut tekniikat mutta on hyvin raskas suorittaa. (Unity Technologies 2021.)

Valoja voi luoda Unityn näkymään ja ympäristöön kahdella tavalla. Uuden valon voi luoda Gameobject-osion alta valitsemalla valon, tai valokomponentin voi lisätä jo olemassa olevaan objektiin. Unity-pelimoottorissa valo voi olla neljää eri tyyppiä. Nämä ovat directional eli suora, point eli piste, spot eli spotti ja area eli alue. Suoran valon ominaisuudet ovat verrannollisia auringonvaloon, spottivalo valaisee määritettyyn suuntaan kartion muotoisen alueen, pistevalo valaisee jokaiseen suuntaan tilassa, ja aluevalo tuottaa valoa nelikulmaiselta pinnalta jokaiseen suuntaan. Näihin valotyyppeihin on erikseen säädettävissä esimerkiksi kulmat, muodot, valon voimakkuus, vaikutusala ja auringon ominaisuudet suorassa valossa. (Unity Technologies 2021.)

Lisättyihin valoihin voi myös määrittää, beikataanko valoa vai ei. Light Layerin eli valotason valitsemalla voi päättää, mihin valotasoihin kyseinen valo vaikuttaa. Jos valotasoa ei valitse, se vaikuttaa automaattisesti kaikkiin renderöitäviin objekteihin ja maahan. Valoista voi määrittää myös, kuinka ne käyttäytyvät volumetrisessä ympäristössä yhdessä sumun sekä varjojen laadun ja käyttäytymisen kanssa. (Unity Technologies 2021.)



Kuva 26. Erilaisia valotyyppejä samassa ympäristössä

Valojen ja ympäristön heijastuksen avuksi kannattaa luoda Reflection probe -objekteja. HDRP käyttää heijastumien laskuun kolmea eri tekniikkaa, joista beikatut ja reaaliaikaiset Reflection probe -objektit ovat yksi tapa. Nämä objektit toimivat hyvinkin samalla tavalla kuin kamera. Ne tallentavat näkymän ympäriltään. Ympäristöobjektit, joissa on heijastava pinta, voivat näyttää tarkemman heijastuksen ympäristöstään näiden avulla. Materiaalit ottavat myös huomioon ympäristön paremmin,

kun kameran kulma vaihtuu. Reflection probe -objekteihin voi valita niiden luonnin jälkeen, ovatko ne reaaliaikaisia, beikattuja vai käyttääkö itse tehtyä heijastuskarttaa. (Unity Technologies 2021.)

Reflection probe -objektin näkymä riippuu siitä, minkä tyyppistä Reflection probea käyttää. Tavallinen reflection probe tallentaa kuvan joka suunnasta ja tallentaa tiedon kuutiokarttana. Planar reflection probe tallentaa näkymän kameran paikasta ja rotaatiosta, minkä jälkeen se tallentaa tiedon 2D-tekstuuriksi. Reflection proben voi luoda Gameobject-valikosta valo-osion alta. (Unity Technologies 2021.)

Käytännön osuuteen luotiin paljon erityyppisiä paikallisia valoja sopiviin kohtiin. Näkymän luonnin yhteydessä ympäristöön tuli automaattisesti auringonvalo, jonka ominaisuuksia säädettiin himmeäksi auringonlaskuksi. Materiaaleissa oli myös vaihtoehto tuottaa valoa objektista, jota käytettiin erityisesti neonkylteissä. Valojen ohella luotiin Reflection probeja tuottamaan realistisempia heijastuksia.

Lopuksi ympäristöön voi lisätä erilaisia decaleita ja muita visuaalisia efektejä, kuten partikkeleita ja paikallista volumetristä sumua, luomaan lisää yksityiskohtia ympäristöön. Käytännön osuudessa lisättiin erilaisia decaleita luomaan railoja maahan ja vesilammikoita ympäristöön sekä paikallista sumua taustalle.

Kuva 27. Valmis kyberpunk-henkinen ympäristö, joka luotiin sekä ilmaisilla että itsetehdyillä 3D-malleilla Unity-pelimoottorissa.

5 POHDINTA

Opinnäytetyön tavoitteena oli tutustua ja tutkia realistisen ympäristön luontiprosessia Unity-pelimoottorissa käyttäen High Definition Renderpipeline -järjestelmää. Projektin lopputuloksena syntyi Unity-pelimoottorissa oleva, suhteellisen realistinen ympäristö, jossa oli käytetty sekä omia että ilmaisia 3D-objekteja suurin piirtein aikataulussa pysyen.

Unity ja HDRP vaikutti hyvältä järjestelmältä tuottamaan hyvännäköistä realistista ympäristöä suhteellisen vähällä vaivalla, varsinkin yhdessä päivittyneiden Substance Painterin ja Substance Designerin kanssa. HDRP on kuitenkin melko erilainen verrattuna esimerkiksi sisäänrakennettuun renderöintiputkeen, minkä takia käyttöliittymä ja logiikka tuntuivat aluksi hyvinkin haastavilta, kun oli tottunut lähinnä sisäänrakennettuun renderöintiputkeen. Kaikkien asetusten ja ympäristön arvojen säätäminen sekä niiden löytäminen vei enemmän aikaa, kuin alun perin oletin. Jonkin verran arvoja ja asetuksia joutui muuttamaan yrityksen ja erehdyksen kautta sekä silmämääräisesti sen mukaan, mikä näytti hyvältä. Unity-pelimoottorin omilla verkkosivuilla oli onneksi hyvin ohjeita, joita seurata.

Yksi suurimmista haasteista oli saada valaistus näyttämään hyvältä Unity-pelimoottorin sisällä. Päivänvalon luominen oli helppoa, ja siitä löytyi paljon ohjeistusta, mutta yöllisen valaistuksen saaminen hyvännäköiseksi eri valonlähteiden kanssa vaati enemmän aikaa. Ympäristöä luodessa siinä olevien objektien heijastus ja valonlähteiden asettelu oli yllättävän tarkkaa puuhaa. Luotu ympäristö alkoi näyttämään paremmalta heijastuksineen ja valoineen vasta, kun useampi objekti oli tuotu ympäristöön. Aluksi määritettyjä valaistus- ja jälkikäsittelyefektien arvoja tuli säädettyä moneen otteeseen myöhemmin tämän takia. Haasteita tuottivat myös päivittyneet käyttöliittymät monissa aiemmin tutuiksi tulleissa ohjelmissa, mikä hidasti objektien tuottamista ympäristöön. HDRP on myös suhteellisen uusi renderöintiputki, joten apua ongelmatilanteissa oli hankala löytää. Kyseiset ongelmat oli myös saatettu korjata uudemmissa Unityn ja HDRP:n versioissa.

Valaistuksen säätämisen vuoksi tuotantolistalla ollut järjestys ei pitänyt lopulta paikkansa. Keskityin lopulta tuottamaan alusta alkaen itse ensimmäisenä valoa tuottavia objekteja, heijastavia objekteja ja materiaaleja, minkä jälkeen siirryin muihin objekteihin. Itse prosessi ennen pelimoottorissa työskentelyä oli kuitenkin suhteellisen suoraviivaista ja helppoa. Suurin haaste siinä oli tekstuurien työstäminen laadukkaiksi.

Projektin toteuttaminen toi kuitenkin paljon lisää kokemusta Unity-pelimoottorissa HDRP:n kanssa työskentelystä ja antoi myös paljon uutta tietoa siitä, kuinka toteuttaa koko 3D-mallinnusprosessi aina konseptoinnista pelimoottorissa työskentelyyn asti. Erityisesti valaistuksen ja heijastuksien ymmärrys kasvoi huomattavasti projektin aikana. Valaistuksesta olisi riittänyt enemmänkin tutkittavaa ja testattavaa, mutta käytännön osuuden laajuuden takia sen teoria jäi osittain pintapuoliseksi.

Tarkoituksena oli luoda koko alue alusta alkaen pelkästään omilla resursseilla, mutta jotta pysyin aikataulussa, käytin myös ilmaisia 3D-malleja ja resursseja. Näitä ilmaisia objekteja korvasin niin monella itse tehdyllä objektilla kuin ehdin tuottamaan aikarajan sisällä. Tämän vuoksi lopputulos ei vastannut myöskään visuaalisesti ihan täydellisesti alkuperäistä mielikuvaa. Haasteita tuotti myös töiden ja opiskelun yhdistäminen korona-aikana.

Projektia voisi myöhemmin kehittää pidemmälle optimoimalla sitä huomattavasti paremmin, tutustumalla valaistuksen tekniseen puoleen syvällisemmin, parantamalla sekä tekstuurien että mallien laatua ja tuottamalla kuvan ympäristön kokonaan itse. Jatkokehitykseen voisi kuulua joko ympäristön muuttaminen pelattavaksi tai VR-mahdollisuuden lisääminen projektiin.

LÄHTEET

Adobe 2021a. Substance 3D Painter. Hakupäivä 9.11.2021. https://www.adobe.com/products/substance3d-painter.html.

Adobe 2021b. Substance 3D Designer. Hakupäivä 9.11.2021. https://www.adobe.com/products/substance3d-designer.html.

Blender 2021. About. Hakupäivä 10.10.2021. https://www.blender.org/about/.

Blender 2021. License. Hakupäivä 10.10.2021. https://www.blender.org/about/license/.

Brodkin, Jon 2013. How Unity3D Became a Game Development Beast. Hakupäivä 19.12.2021. https://insights.dice.com/2013/06/03/how-unity3d-become-a-game-development-beast/.

Clinton, Keith 2016. Agile, Scrum and Kanban for Video Game Development. Hakupäivä 31.01.2021. https://www.clintonkeith.com/resources/Agile-Scrum-and_Kanban-for-Video-Game-Development.pdf.

Diaz, Tim 2013. Texel Density for game art. Hakupäivä 19.12.2021. https://3dtotal.com/tutorials/t/texel-density-for-game-art-tim-diaz-udk-bricks-3ds-max-unreal-tutorial-environment.

Donzallaz, Pierre 2020. How to set up Unity's High Definition Render Pipeline for high-end visualizations. Hakupäivä 9.11.2021. https://blog.unity.com/aec/how-to-set-up-unitys-unitys-high-definition-render-pipeline-for-high-end-visualizations.

Gaget, Lucie 2019. All you need to know about digital sculpting. Hakupäivä 9.11.2021. https://www.sculpteo.com/blog/2019/06/12/what-is-digital-sculpting/.

Gahan, Andrew 2011. 3ds max modeling for games: insider's guide to game character, vehicle, and environment modeling. volume 1 (s. 9). Waltham, MA; Oxford: Focal Press.

Glawion, Alex 2021. Normal vs Displacement vs Bump Maps: Differences and when to use which. Hakupäivä 9.11.2021. https://www.cgdirector.com/normal-vs-displacement-vs-bump-maps/.

Hemmens, Jonjo 2021. Foliage Art for Forest Environments with Jonjo Hemmens. Hakupäivä 9.11.2021. https://substance3d.adobe.com/magazine/foliage-art-for-forest-environments-with-jonjo-hemmens/.

Interesting Engineering 2016. How Do Game Engines Work? Hakupäivä 9.11.2021. https://interestingengineering.com/how-game-engines-work.

Player Research 2021. Games User Research Methodology Series: Concept Tests. Hakupäivä 6.01.2022. https://playerresearch.medium.com/games-user-research-methodology-series-concept-tests-4875ec8ec5b8.

Prus, I. 2016. What is 3d Modeling? Things You've Got To Know Nowaday. Hakupäivä 6.01.2022. https://archicgi.com/product-cgi/3d-modeling-things-youve-got-know/.

Silverman, David 2013. 3D Primer for Game Developers: An Overview of 3D Modeling in Games. Hakupäivä 22.11.2021 https://gamedevelopment.tutsplus.com/articles/3d-primer-for-game-developers-an-overview-of-3d-modeling-in-games--gamedev-5704.

Spatial Team 2019. The Main Benefits and Disadvantages of Voxel Modeling. Hakupäivä 5.2.2022. https://blog.spatial.com/the-main-benefits-and-disadvantages-of-voxel-modeling.

Tyler, Dustin 2021. How to Choose the Best Video Game Engine. Hakupäivä 7.11.2021. https://www.gamedesigning.org/career/video-game-engines/.

Tyler, Dustin 2021. What You Need To Know About 3D Modeling for Games (You Asked. We Answered.). Hakupäivä 7.11.2021 https://www.gamedesigning.org/learn/3d-modeling/.

Unity Technologies 2020. High-Definition Render Pipeline. Hakupäivä 10.10.2021. https://docs.unity3d.com/Packages/com.unity.render-pipelines.high-definition@13.0/manual/index.html.

Unity Technologies 2021a. Unity. Hakupäivä 9.11.2021. https://unity.com/.

Unity Technologies 2021b. Unity Manual. Hakupäivä 10.10.2021. https://docs.unity3d.com/Manual/high-definition-render-pipeline.html.

Unity Technologies 2021c. OutHere and Skanska: A Unity for AEC case story. Hakupäivä 20.12.2021. https://unity.com/case-study/outhere-and-skanska.

Unity Technologies 2021d. Understanding Rendering Paths. Hakupäivä 31.10.2021. https://learn.unity.com/tutorial/understanding-rendering-paths-2019-3#5fab0333edbc2a001f145277.

Ward, Jeff 2008. What is a Game Engine? Hakupäivä 9.11.2021. https://www.gamecareerguide.com/features/529/what_is_a_game_.php.

Wolstenholme, Kevin 2021. 5 Reasons for creating mood boards in game development. Hakupäivä 11.10.2021. https://risinghighacademy.com/5-reasons-for-creating-mood-boards-in-game-development/.

World of Level Design 2021. How to Plan Level Designs and Game Environments in 11 Steps. Hakupäivä 11.10.2021. https://www.worldofleveldesign.com/categories/level_design_tutorials/how-to-plan-level-designs-game-environments-workflow.php.