Opinnäytetyö (AMK)

Tietotekniikka

Peliteknologia

2016

Nina Mattila

3D-MALLIEN SUUNNITTELU JA TOTEUTUS PELIYMPÄRISTÖÖN

TURKU AMK
TURKU UNIVERSITY OF APPLIED SCIENCES

OPINNÄYTETYÖ (AMK) | TIIVISTELMÄ

TURUN AMMATTIKORKEAKOULU

Tietotekniikka | Peliteknologia

2016 | 46

Yliopettaja Mika Luimula, dosentti

Nina Mattila

3D-MALLIEN SUUNNITTELU JA TOTEUTUS PELIYMPÄRISTÖÖN

Tämän opinnäytetyön tarkoituksena oli tutkia peliympäristön suunnittelua ja toteutusta 3D-mallintajan näkökulmasta. Työssä keskityttiin tutkimaan mallintamiseen liittyviä teknisiä rajoitteita ja pelimalleihin soveltuvia mallinnustekniikoita. Tutkimusosuudessa tehtiin katsaus 3D-mallintajan rooliin pelinkehitysprojektissa ja tutkittiin 3D-grafiikan käyttötarkoituksia peleissä. Työssä esiteltiin myös käytetyimpiä 3D-mallinnusohjelmia sekä muita mallintamistyössä käytettäviä työkaluja.

Tutkimuksessa käytettiin aineistona pelinkehitykseen ja 3D-mallintamiseen liittyviä julkaisuja ja artikkeleita. Opinnäytetyön käytännönosuuden toimeksiantona oli Turun ammattikorkeakoulun ja Turun yliopiston Digital TimeTrek -projekti, joka on useammalle alustalle tarkoitettu sovellus, jossa opetetaan maailmankaikkeuden synnyn eri vaiheita. Tähän sovellukseen piti mallintaa käyttöliittymässä näkyvien ympäristöjen 3D-malleja. Projekti toimi käytännönesimerkkinä opinnäytetyöhön liittyvissä tutkimusaiheissa.

Tutkimusten tuloksena muodostui yleiskuva pelien 3D-graafiikan suunnittelun ja toteutuksen perusteista. Suunnittelua rajoittavat varsinkin kohdealusta ja pelin tyyli. Pelialalla 3D-graafikolla on useita työtehtäviä, joten perustietoa tarvitaan eri aihealueista. Käytännönesimerkki ja tutkimukset antavat yleiskatsauksen pelialalla toimivan 3D-graafikon työtavoista ja osaamistarpeista.

ASIASANAT:

3D-mallinnus, 3ds Max, peligrafiikka, peliympäristö, peliteknologia

BACHELOR´S THESIS | ABSTRACT

TURKU UNIVERSITY OF APPLIED SCIENCES

Information Technology | Game Technology

2016 | 46

Principal Lecturer Mika Luimula, Adj. Prof.

Nina Mattila

DESIGNING AND IMPLEMENTING 3D MODELS IN A GAME ENVIRONMENT

The goal of this thesis was to examine how game environments are designed and implemented from a 3D artist's point of view. This thesis focuses on examining technical restrictions affecting modeling and also examines the most common modeling techniques for game assets. The theoretical section of the thesis provides an overview of a 3D modeler's role in a game development project and also examines the usage of 3D graphics in games. The most commonly used 3D modeling software and other 3D tools are also introduced in this section.

The thesis uses publications and articles concerning 3D modeling and game design as a basis of the research. The practical part of this thesis was implemented as part of the Digital TimeTrek, a joint project of Turku University of Applied Sciences and University of Turku. It teaches about the evolution and development of the universe. The objective of this thesis was to create 3D models for this project. The project functioned as an empirical example for topics related to this thesis.

The results of this thesis give an overview of how basic 3D graphics are designed and implemented for games. The greatest restrictions are the target platform and the style of the game. 3D artists have several tasks to perform so basic knowledge of different subjects is needed. The empirical example of this thesis combined with research material gives an overview of the work and requirements for a 3D modeler in the game industry.

KEYWORDS:

3D modeling, 3ds Max, game graphics, game environment, game technology

SISÄLTÖ

KUVAT

TAULUKOT

SANASTO

Key frame	Animointityyli, jossa siirtyminen tapahtuu sulavasti animaation alkupisteestä loppupisteeseen
Lavaste	Ympäristön malli, joka on yleensä staattinen tai toiminnoiltaan yksinkertainen
LOD	Menetelmä, jossa mallin yksityiskohtien määrä vaihtelee välimatkan mukaan *(Level of Detail)*
Node	Objekteja, joilla voidaan ohjelmoida tapahtumia, funktioita tai muuttujia
Normaalikartta	RBG-tekstuurikartta, joka antaa illuusion yksityiskohdista laskettujen varjostuskohtien mukaan
Optimointi	Prosessi, jolla pyritään parantamaan suorituskykyä
Pelimoottori	Videopelin ohjelmistokehys
Plane	Kaksiulotteinen taso 3D-grafiikassa
Polygoni	Monikulmio, joiden joukko muodostaa 3D-mallin pinnan
Proseduraalinen	3D-grafiikassa tarkoitetaan mallinnustapaa, jossa ohjelma generoi mallin algoritmien pohjalta
Renderöinti	Prosessi, jossa 3D-grafiikka tulostetaan tarkkalaatuisena sisältäen tiedot valoista ja varjoista
Riggaus	Luiden asettaminen 3D-hahmomallille
Tekstuuri	Kaksiulotteinen kuva, jolla pinnoitetaan 3D-malli
UV-kartoitus	Prosessi, jossa 3D-mallin pinta luodaan kaksiulotteiseksi kartaksi
Verteksi	Pisteet, jotka yhdistävät polygonien reunat

1 JOHDANTO

Tämän opinnäytetyön tarkoituksena on tutkia peliympäristön suunnittelua ja toteuttamistapoja 3D-mallintajan näkökulmasta. Peliympäristöllä tarkoitetaan sellaista ympäristöä, johon peli sijoittuu. Se muodostuu lavasteista, joita ovat esimerkiksi puut, talot sekä muut taustalla esiintyvät objektit. Useammat nykyajan pelit toteutetaan käyttäen 3D-grafiikkaa, ja tämän takia 3D-mallintajan rooli peliprojektissa on tärkeä. 3D-mallintaja on yleensä vastuussa esimerkiksi peliympäristöjen lavasteiden toteutuksesta tai pelihahmojen mallintamisesta. Peliympäristön kokonaisuuden toteuttamisesta voi olla vastuussa useampia henkilöitä, mutta varsinkin pienimmissä projektitiimeissä mallintaja saattaa olla vastuussa koko ympäristön tyylin toteuttamisesta.

Mallintaminen on sekä teknistä ja taiteellista työtä. Varsinkin pelien 3D-grafiikan mallintamisessa tulee ottaa huomioon monia erilaisia asioita, joita ei tule vastaan esimerkiksi elokuvateollisuuden 3D-grafiikan tekemisessä. Videopelit toimivat laitteissa, joissa on paljon teknisiä rajoitteita. Tämän takia tekniset rajoitteet tulee huomioida jo mallintamistyötä aloittaessa. Pelien 3D-malleja pitää yksinkertaistaa mahdollisimman paljon, mutta niiden pitää silti näyttää hyvältä. Lisäksi peliympäristön suunnitteluun ja lavasteiden mallinnukseen vaikuttavat myös pelin toiminnallisuudet. Tämän takia opinnäytetyössä tutkitaan varsinkin peleihin sopivia mallinnustapoja, sekä mallien optimointia pelimoottoria varten.

Opinnäytetyön teoriaosuudessa esitellään ensiksi pelien 3D-grafiikan historiaa ja nykyaikaisia käyttötapoja. Vaikka 3D-grafiikka on kehittynyt paljon viime vuosikymmenen aikana, jotkin varhaisaikojen rajoitteet ovat vieläkin ajankohtaisia nykyajan mobiililaitteissa. Nykyaikaisista käyttötavoista käydään läpi pelien renderöintiä, koska renderöintitapa vaikuttaa paljon mallintamiseen ja pelin tyyliin. Työssä esitellään tarkemmin myös 3D-mallintajan työtehtäviä ja roolia peliprojektissa. Ympäristöjen toteutuksesta käydään läpi peliympäristön suunnittelun perusteita sekä käytännönvaiheita liittyen mallintamiseen. Esille otetaan myös erilaisia mallinnustekniikoita keskittyen varsinkin peleissä käytettyihin tekniikoihin, sekä esitellään 3D-mallien optimointitapoja. Teoriaosuuden lopussa vertaillaan käytetyimpiä 3D-ohjelmistoja, työkaluja ja pelimoottoreita 3D-grafiikan toteuttamisen kannalta. Pelien 3D-grafiikan toteutukseen on olemassa useampia ohjelmistoja ja aputyökaluja, jonka takia sopivien ohjelmien löytäminen voi olla haastavaa.

Opinnäytetyön käytännönosuuden toimeksiantona on Turun ammattikorkeakoulun ja Turun yliopiston Digital TimeTrek -projekti, joka on digitaalisella alustalla toimiva maailmankaikkeuden syntymisen teoriaa opettava opetuspeli. Projektin digitaalista sisältöä tehdään Turku Game Labissa. Sovelluksen käyttöliittymässä on tarkoitus näyttää kosmisen sekä geologisen ajan pääkohtia visuaalisesti aikajanana. Aikajana alkaa alkuräjähdyksestä ja kuvaa maapallon kehitystä syntymisestä nykyhetkeen asti. Toimeksiannon tarkoituksena oli luoda käyttöliittymään tarvittavia 3D-malleja kuvaamaan erilaisia maailmankaikkeuden syntymistä kuvaavia tapahtumia. Käytännönosuudessa kerrotaan mallinnustyön työvaiheista. Työvaiheisiin kuului suunnittelu, työkalujen valinta, mallien toteutus sekä niiden viimeistely. Projekti toimii käytännönesimerkkinä opinnäytetyöhön liittyvissä tutkimusaiheissa.

Tämän työn tavoitteena on koota tietoa pelinkehityksen kannalta tärkeistä asioista 3D-mallintajalle. Näitä asioita ovat esimerkiksi mallintamisen tekniikat, optimointi ja töiden viimeistely pelimoottoria varten. Opinnäytetyön tarkoituksena on antaa yleiskuva pelien 3D-grafiikan suunnittelun ja toteutuksen perusteista. Tutkimuksessa lähteinä käytetään julkaisuja ja artikkeleita liittyen pelinkehitykseen ja 3D-mallintamiseen. Käytännönesimerkki ja tutkimukset antavat yleiskatsauksen pelialalla toimivan 3D-graafikon työtavoista ja osaamistarpeista.

2 3D-GRAFIIKKA PELINKEHITYKSESSÄ

3D-grafiikka käytetään nykyaikana laajasti peleissä. Monien tietokoneiden sekä konsolien kyvyt mahdollistavat monimutkaisten ja vaikuttavien 3D-grafiikan teon. Myös älypuhelimet ja tablettitietokoneet ovat niin tehokkaita, että niille on mahdollista tehdä kolmiulotteisia pelejä. Pelit ovat interaktiivisia, joka tekee niistä uniikin taiteenmuodon. (Silverman 2013.) Tässä luvussa esitellään 3D-grafiikan käytön historiaa sekä sen nykyaikaisia käyttötarkoituksia. Sen lisäksi käsitellään pelien renderöinnin merkitystä ja käyttötapoja. Luvussa tehdään myös katsaus 3D-mallintajan rooliin pelialalla.

2.1 3D-grafiikan käyttö peleissä

Peleissä on ollut olemassa jo kauan 3D-grafiikkaa, mutta vasta viime vuosikymmenen aikana sitä on pystytty käyttämään laajemmin. 3D-grafiikka on mahdollistanut sen, että pelaaja voi kokea ja nähdä pelimaailman monesta eri kulmasta ja liikkua pelissä vapaammin ilman rajoitteita. Maailman elävyyden lisäksi myös pelihahmo pystyy vastaamaan luonnollisesti pelissä kohtaamiin asioihin, sillä hahmolle pystytään tekemään monia erilaisia animaatioita ja käyttämään pelifysiikan mahdollisuuksia. Myös nykyajan 2D-pelit saatetaan toteuttaa 3D-grafiikan avulla, sillä se antaa paremmat mahdollisuudet luoda animaatioita, eikä 2D-hahmoa tarvitse muokata yksitellen kuva-kuvalta. (TechRadar 2010.)

2D-grafiikkaa oli yleisin grafiikkatyyli 1990-luvun puoleenväliin saakka. 2D-grafiikassa pelinäkymä on yleensä kuvattu joko yläpuolelta tai sivusta, mutta myös isometrinen kuvakulma oli suosittu. Isometrinen kuvakulma antoi illuusiota kolmiulotteisuudesta, sillä kaksiulotteiset kuvat oli piirretty näyttämään kolmiulotteisilta. (Art of Video Games 2016a.) Esimerkiksi 1994-julkaistussa *Sim City 2000* -pelissä käytettiin kolmiulotteiselta näyttävää isometristä 2D-grafiikka (kuva 1). 1990-luvun puolenvälin jälkeen 3D-grafiikka alkoi pikkuhiljaa yleistymään peleissä. 3D-grafiikasta oli kuitenkin haasteellista tehdä yhtä hyvän näköistä kuin ajan 2D-grafiikasta. Sen ajan pelien rajoitteena olivat järjestelmät, jotka eivät kestäneet raskaita 3D-grafiikoita. Peleissä jouduttiin tyytymään hyvin yksinkertaiseen 3D-grafiikkaan. Yleensä mallien ja ympäristön pinnat olivat tasaisia ja tekstuurit epäselkeitä. 3D-grafiikan raskauden takia usein käytettiin 2.5D-grafiikkaa, jolloin todellisuudessa pelin objektit muodostuivat kaksiulotteisista kuvista,

jotka suurenivat ja pienenivät sen mukaan, kuinka lähellä pelaaja oli. Tämä tapa loi vaikutuksen kolmiulotteisuudesta, mutta oli samalla kevyempi ratkaisu rajoitetuille järjestelmille. (TechRadar 2010.)

Kuva 1. 1994-julkaistu Sim City 2000 käyttää kolmiulotteiselta näyttävää isometristä kuvakulmaa (Whitwam 2014).

Tietokoneiden teknologian kehittyessä myös 3D-grafiikan mahdollisuudet alkoivat kehittymään. Aiemmin tietokoneissa ei ollut tarvetta näytönohjaimille, joten pelien toiminta tukeutui useimmiten prosessoreihin. Kuitenkin näytönohjaimista alkoi pikkuhiljaa tulla pakollinen osa PC-pelien toimimista varten. Seuraava suuri edistys oli myös pelien renderöinnin kehittyminen, joka mahdollisti eläväisemmän 3D-grafiikan. Teknologian rajoitteet alkoivat vähenemään, jolloin pelien visuaaliseen puoleen voitiin panostaa paljon enemmän. Peleistä alkoi kuitenkin tulemaan myös lyhyempiä, koska näyttävien grafiikoiden toteutus vei enemmän aikaa ja rahaa pelien kehityksestä. Tätä ongelmaa varten pelialalla yleistyivät ohjelmat, joiden avulla pystyi proseduraalisesti kehittämään 3D-malleja. (TechRadar 2010.)

Nykyaikana peligrafiikan tyylin suunnittelu on vapaampaa, koska rajoituksia on vähemmän. 3D-grafiikasta on tullut yhä realistisempaa, mutta toisaalta pelit voidaan tehdä tyylitellysti sarjakuvamaiseksi. Esimerkiksi mobiilipeleissä rajoitukset ovat yhä hyvin ajankohtaisia, joten pienet pelifirmat yrittävät erottua joukosta luovilla ratkaisuilla. Tyylitellyn tyylin valinta on siitäkin hyvä ratkaisu, että liian realistiseen grafiikkaan pyrkiessä

ihminen alkaa huomaamaan räikeämmin realistisuudesta poikkeavia virheitä. Sen takia realistisen tyylin toteutus on osittain vielä hyvin vaikeaakin. (Art of Video Games 2016b.)

2.2 Pelien renderöinti

Valot, varjostukset sekä tekstuurit viimeistelevät peliympäristön ja mallinnustyön. 3D-grafiikan toteutuksen kannalta renderöinti on tärkeä osa lopputulosta. Renderöinnillä tarkoitetaan grafiikan tulostamista tarkempilaatuisena, kuin se on ollut käsittelyvaiheessa. Renderöinti punoo yhteen tiedot kolmiulotteisien objektien tekstuureista, kuvakulmasta ja valaistuksesta (kuva 2). Renderöintiä käytetään peleissä ja elokuvissa. Tarkoituksena on toteuttaa lopputulos, joka on visuaalisesti näyttävä. (3D Raamattu 2012.) Renderöintiä käytetään kuitenkin täysin eri tavalla näissä kahdessa viihteen alassa. Elokuvissa katsoja näkee kuvan vain yhdestä kuvakulmasta, jonka takia lopputulos voi olla tallennettua kuvamateriaalia. Pelit taas ovat interaktiivisia, jonka takia pelaajan käyttäytymistä ei pystytä ennustamaan pelissä. Pelaaja voi vapaasti liikkua ja tehdä asioita, jolloin grafiikan pitää vastata käyttäjän toimiin. (Silverman 2013.)

Kuva 2. CryEngine-pelimoottorilla renderöity ympäristö (CryEngine 2016).

Renderöinti voidaan jakaa kahteen erilaiseen toteutustapaan: esirenderöinti ja reaaliaikainen renderöinti. Reaaliaikaista renderöintiä käytetään silloin kun tarvitaan paljon

interaktiivisuutta. Sen takia se on suosituin peleissä. Esirenderöinti tarkoittaa kuvan renderöimistä valmiiksi materiaaliksi, ja se on suosittu varsinkin elokuvissa. (Rouse 2009.) Esirenderöintiä tarvitaan kuitenkin myös peleissäkin. Tässä luvussa esitellään näiden kahden renderöinnin eroavaisuuksia.

2.2.1 Reaaliaikainen renderöinti

Interaktiivisissa peleissä on mahdotonta ennustaa pelaajan käyttäytymistä, koska pelaaja voi liikkua tai kokeilla erilaisia asioita vapaasti. Valojen ja varjojen tulee näyttää luonnollisilta eri kuvakulmista, jolloin renderöinnin pitää tapahtua nopeasti pelin edetessä. 3D-peleissä käytetään tämän takia reaaliaikaista renderöintiä. Reaaliaikaisessa renderlinnissä pelissä näkyvä ympäristö generoituu silloin kun alue näkyy pelaajan ruudulla. (Silverman 2013.)

Koska renderöinti tapahtuu samalla hetkellä, 3D-objektien pitää olla riittävän yksinkertaisia renderöintiajan nopeuttamiseksi. Peleihin mallintamisessa erityistä on se, että optimointi ja polygonien määrä pitää olla harkittu laitteen ja pelimoottorin mukaisesti. Peleissä on tärkeintä saada lyhyt renderöintiaika, joten sekä pelimoottorin ja pelissä käytettyjen mallien tulee olla hyvin optimoituja. (Silverman 2013.) Reaaliaikaisen renderöinnin nopeus riippuu paljon näytönohjaimesta ja siitä kuinka paljon pelin informaatiosta on esikoottua. Yleensä esimerkiksi valaistukset ovat valmiiksi laskettuja renderöintiajan nopeuttamiseksi. Reaaliaikaisessa renderöinnissä nopeus on tärkeää, sillä jo pienikin ero piirtonopeudessa voi olla pelissä kohtalokasta. Jotta peli näyttää sulavalta, piirtonopeuden pitää olla vähintään 18-20 kuvaa sekunnissa. Piirtoajan nopeuden vähimmäistarpeet kuitenkin riippuvat paljon pelin tyypistä. (Slick 2014a.)

2.2.2 Esirenderöinti

Esirenderöintiä käytetään silloin kun tarvitaan monimutkaista grafiikkaa, jota on vaikea toteuttaa reaaliaikaisesti. 3D-animaationa toteutetut elokuvat voivat olla visuaalisesti erittäin näyttäviä, koska renderöinti tehdään aina esityönä. Katsoja näkee kuvan vain yhdestä kuvakulmasta, jonka takia lopputulos voi olla tallennettua kuvamateriaalia. Tämän takia elokuvien renderöinnissä ei ole rajoituksia nopeudelle. Yhden kuvan renderöinnissä voi kulua jopa useita tunteja aikaa. (Silverman 2013.)

1990-luvulla ja 2000-luvun alussa peleissä käytettiin usein esirenderöityjä taustoja, jotta taustoista saatiin realistisempia. Sen ajan pelimoottoreilla se olisi ollut vaikea toteuttaa reaaliaikaisesti. (Wikipedia 2016.) Esirenderöinnin ongelma on se, että se vähentää interaktiivisuutta, ja usein pakottaa käyttämään muuttumatonta kuvakulmaa. Kuitenkin nykyaikana esirenderöintiä käytetään peleissä videoiduissa välikohtauksissa ja varsinkin mobiilipeleissä, sillä mobiililaitteilla on yhä paljon teknisiä rajoitteita. (Wikipedia 2016.) Esimerkiksi Supercellin 2012 iOS:lle ja myöhemmin Androidille julkaistu mobiilipeli *Clash of Clans* käyttää esirenderöityä 3D-grafiikkaa (kuva 3). Pelin mallit on ensin tehty 3ds Max -mallinnusohjelmalla ja sen jälkeen renderöity 2D spriteiksi. (Jordan 2012.)

Kuva 3. Esirenderöityjä objekteja *Clash of Clans* -pelistä (Jones 2014).

2.3 3D-mallintajat työtehtävät

Varhaisissa videopeleissä ohjelmoijat tekivät kaiken grafiikan peleihin. Sen takia pelien grafiikka saattoi usein olla myös hyvin karkeaa. Nykyaikana peligraafikon työ on oma erikoisalansa, ja se voidaan jakaa moneen eri osa-alueeseen sekä 2D- ja 3D-grafiikan suhteen. (Rogers 2014, 19.)

3D-graafikko on sekä teknisen ja graafisen alan asiantuntija. Peliartistin työtehtävinä on tehdä visuaalisia elementtejä peliin, kuten hahmoja, ympäristöjä, lavasteita tai tekstuureja. Pelin ulkoasu on yksi tärkeistä osista pelien menestyksessä, sillä se on yleensä ensimmäinen asia, jonka käyttäjä huomioi. 3D-grafiikan tyyli saattaa peleissä vaihdella realistisesta tyyliteltyyn, jolloin tarpeet ovat erilaiset. Realistisessa grafiikassa mallinta-

jan tulee osata tehdä yksityiskohtaisia malleja, joissa käytetään realistisenomaisia tekstuureja. Tyylitelty grafiikka on paljon vapaampi toteuttaa ja tekstuurit saatetaan tehdä maalauksenomaiseksi. (Creative Skillset 2016.)

3D-mallintajien työt voidaan jakaa lavasteiden tai hahmojen mallintajiin. Yleensä 3D-mallintaja tekee mallit joko konseptiartistin hahmotelmien perusteella tai suunnittelee ne itse. 3D-mallintaja on usein myös vastuussa animaatioiden tekemisestä hahmoille. Toisaalta mallintaja voi toimia myös ympäristöartistina, johon liittyy tehtävät sekä mallinnusohjelman ja pelimoottorin puolella. Yleensä työssä vastataan staattisten tai vähän liikkuvien objektien mallinnuksesta. Ympäristöartisti saattaa myös työskennellä pelimoottorin sisällä lisäämällä ja järjestämällä tarvittavat mallit pelimaailmaan. (Gooch 2016.)

3D-mallintajien työtehtävät vaihtelevat työpaikan koon ja tarpeiden mukaan. Suurissa pelifirmoissa 3D-mallintajan työ saattaa keskittyä vain yhteen osuuteen, kuten hahmoihin, ympäristöön tai animointiin. Pienemmissä pelifirmoissa 3D-graafikko saattaa olla vastuussa kaikista peliin tarvittavasti 3D-grafiikasta. 3D-graafikolla tulee olla hyvä tuntemus käytetystä sovelluksista sekä myös pelimoottorista. (Creative Skillset 2016.) Yleensä pelialalla 3D-mallintajan tulee mallintamisen lisäksi myös tehdä tekstuurit. Tämän takia hyvät teksturointitaidot ovat tärkeä osa pelialalla. (Masters 2014a.)

3 3D-MALLINTAMINEN PELIYMPÄRISTÖÖN

Peliympäristö on pelin näyttämö, joka koostuu erilaisista lavasteista. Peliympäristön toteutukseen kuuluu yleensä kokonaiskuvan suunnitteleminen, jonka jälkeen toteutetaan yksittäiset 3D-mallit. Peliympäristöä luodessa tulee huomioida kohdealustan tekniset rajoitteet ja toteuttaa optimointi sen mukaisesti. Tässä luvussa esitellään peliympäristön suunnittelun perusteita, sekä esitellään käytännönvaiheita liittyen mallintamiseen. Esille otetaan myös erilaisia mallinnustekniikoita keskittyen varsinkin peleissä käytettyihin tekniikoihin. Lopuksi käsitellään pelien optimointia 3D-graafikon näkökulmasta.

3.1 Ympäristön suunnittelu

Pelin esteettinen tyyli riippuu kohdeyleisöstä, kohdealustan teknisistä rajoitteista ja projektin aikarajoitteista. Ideaalisesti pelin tyyli ja kohdennus on suunniteltu etukäteen ennen ympäristön suunnittelua, mutta todellisuudessa ympäristöä saatetaan alkaa rakentamaan ennen tarkkoja suunnitelmia. Hyvin tehdyssä 3D-ympäristössä on yhteneväinen tyyli, harkittu tilankäyttö sekä käytetty yksinkertaistettuja ratkaisuja. (Rabin 2009.)

Peliympäristön tyylin suunnittelussa tutkitaan, minkälaista tunnelmaa peliin halutaan ja minkälaiseen kohteeseen se sijoittuu. Ympäristön oikeanlaisen tunnelman löytämiseen käytetään apuna esimerkiksi kuvamateriaalia samankaltaisista paikoista. Toisaalta teemaan liittyvissä kohteissa saatetaan käydä myös tutustumassa. Näin tekevät varsinkin suuremmat peliyhtiöt. Esimerkiksi Ubisoftin *Assasins's Creed Syndicaten* kehittäjät tekivät vierailun Lontooseen, koska peliin haluttiin luoda uskottava viktoriaanisen ajan Lontoon tunnelma. (Dansereau 2015, 22-23.)

Pelimaailman elävöittämiseen käytetään erilaisia keinoja. Esimerkiksi pelimaailmaa voidaan elävöittää käyttämällä ylikorostettuja värejä tai suurta kontrastia. Pelaajan käsitys uskottavuudesta *(Suspension of Disbelief)* on paljon joustavampi pelimaailmassa. Värien liioittelua käytetään usein varsinkin valokuvaamisessa, jolloin ammatinvalokuvaajat parantavat värin intensiteettiä ja kontrastia suodattimien avulla. Kontrastin avulla peliin pystytään tuomaan syvyysvaikutelmaa, joka näkyy myös todellisessa maailmassa. Syvyysvaikutelma luodaan asettamalla lähempänä olevien objektien kontrasti korkeammaksi ja kauempana olevien objektien pienemmäksi. Yleensä peleissä syvyys-

vaikutelma saadaan sumun avulla, joka luo samalla myös kontrastieroja. Sumulla pystytään myös piilottamaan kaukana olevaa geometriaa (kuva 4). Kaukana näkyvän geometrian peittäminen auttaa nostattamaan piirtonopeutta, sillä silloin pelin ei tarvitse piirtää jokaista mallia samaan aikaan yhtä tarkasti. (Rabin 2009.)

Kuva 4. Sumun avulla voidaan peittää ylimääräistä geometriaa (World of Warcraft 2014).

Pelin perspektiivin huomioiminen on myös hyvin tärkeä osa suunnittelua. Ympäristöt tulee suunnitella pelaajan kamerakuvakulman mukaisesti, sillä tietynlaiset elementit ovat vaikeampi erottaa eri kuvakulmista. Pelinäkymistä käytetyimpiä on ensimmäisen ja kolmannen persoonan näkymä. Ensimmäisen persoonan kamera näyttää pelimaailman siten kuin peli nähtäisiin pelihahmon silmin, kun taas kolmannessa persoonassa pelihahmo nähdään pelihahmon selän takaa. (Rabin 2009.)

Peliympäristön suunnittelu ei ole aina sama asia kuin tason suunnittelu. Tasosuunnittelussa (*Level Design*) keskitytään pelikentän tekniikkaan ja toiminnallisuuksiin, kun taas ympäristön visuaalinen suunnittelu ja toteutus on taiteellista työtä. Toisaalta monissa tapauksissa tasosuunnittelusta ja ympäristön suunnittelusta on vastuussa sama henkilö. (Ahearn 2008.) Molemmat aihealueet vaikuttavat merkittävästi pelikokemukseen. Pelin ympäristö vaikuttaa pelin tunnelmaan ja uskottavuuteen kun taas tasosuunnittelu vaikuttaa pelin etenemiseen.

3.2 Lavasteiden mallinnus

Suurin osa pelien 3D-malleista muodostuu esineistä, jotka ovat tarkoitettu pelkästään taustan täytteeksi. Näitä ovat esimerkiksi taustalla olevat talot, puut tai esineet. Yleensä näistä puhutaan lavasteina. Lavasteet ovat tärkeä osa pelin elollistamisessa ja ne vaativat paljon suunnittelua, vaikka ne ovatkin pelihahmoja yksinkertaisempia. Ennen mallintamistyön aloittamista pitää miettiä, kuinka tärkeä osa mallinnettava lavaste on pelin kannalta. On tärkeää miettiä sitä, kuinka suuri kyseinen malli on tai kuinka läheltä pelaaja pystyy sitä tarkastelemaan. Mitä pienempi malli on, sitä vähemmän sen mallintamiseen kannattaa käyttää aikaa. Jos pelaaja ei tule läheltä tarkastelemaan kyseistä lavastetta, sen ei tarvitse olla yksityiskohtainen. Toisaalta voidaan myös miettiä, onko lavaste jotenkin vuorovaikutuksissa pelaajan kanssa. Esimerkiksi voiko pelaaja liikuttaa sitä tai onko siinä nappeja joita voi painaa. Tämä ratkaisee sen, miten malli kannattaa jakaa eri osiin. Ennen mallintamista on hyvä myös tehdä tutkimusta lavasteen ulkonäöstä ja ominaisuuksista. Lavasteiden lopullisesta versiosta voidaan tehdä myös konseptikuvia, jotka auttavat mallin rakentamisessa. (Pettit 2015a.)

Lavasteiden tarkoituksena on luoda illuusiota uskottavasta ympäristöstä. Pelikentässä on yleensä usein toistuvia malleja, kuten puita, kiviä tai taloja, joiden huono suunnittelu voi rikkoa pelikentän luonnollisuutta. Varsinkin suurissa pelikentissä samanlaiset mallit voivat olla niin toistuvia, että pelaaja voi huomata niiden olevan identtisiä. Tämän takia usein toistuvista lavasteista kannattaa tehdä useampia versioita. Samankaltaisten mallien ei kuitenkaan kannata olla liian erinäköisiä, sillä uniikit mallit erottuvan joukosta selkeimmin. Lavasteiden tulee sulautua maisemaan, joten parhain tapa on tehdä niistä vain hieman erilaisia. Tarkoituksena on luoda illuusiota siitä, että ympäristössä olisi realistisesti luonnollista vaihtelevuutta. (Vaccaro 2012.)

Lavasteita mallintaessa on hyvä valita työn kannalta sopiva mallinnustekniikka. Esimerkiksi laatikkomainen esine voidaan helpoiten aloittaa laatikko-primitiivistä. Mallinnuksessa voidaan myös miettiä, mikä osa on tarpeellista näyttää geometriassa ja mitä voidaan esittää pelkällä tekstuurilla. Jo pienikin uusi yksityiskohta geometriassa voi lisätä polygonien määrää huomattavasti. Mallin teon jälkeen seuraavaksi tehdään yleensä UV-kartta. UV-kartalla tarkoitetaan 2D-kuvapintaa, joka lisätään 3D-mallin pinnalle. UV-karttaa tehdessä tärkeintä on huomioida että tekstuuri näyttää mallin pinnalta tasaiselta. Itse tekstuurit toteutetaan UV-kartan jälkeen. Mallin tekstuurit voidaan tehdä joko kuvankäsittelyohjelmalla tai teksturointiin tarkoitetulla ohjelmalla. Nykyään tekstuu-

reja tehdään usein myös proseduraalisesti. Näiden työvaiheiden jälkeen malleille voidaan tehdä tarvittavat riggaukset tai animaatiot. Lopuksi malli voidaan siirtää pelimoottoriin FBX-tiedostomuodossa. (Pettit 2015b.)

3.3 Yleisimmät mallinnustekniikat

3D-mallintamiseen on olemassa monenlaisia eri toteutustekniikoita. Mallintamistekniikan valinta riippuu paljon toteutettavasta mallista ja käyttötarkoituksista. Kaikki mallinnustekniikat eivät ole parhaimpia mahdollisia valintoja pelien 3D-mallien rakentamiseen, sillä ne voivat olla liian raskaita.

3.3.1 Polygonaaliset mallinnustavat

Peleihin tehtävissä malleissa yleisin tapa on käyttää polygoni-mallinnusta, joka antaa paljon joustavuutta ja säätömahdollisuuksia. (Rabin 2009.) Tässä tekniikassa mallia aloitetaan tekemään joko yhdestä tai useammasta polygonista, jonka jälkeen sitä muovataan eteenpäin. Tekniikan vahvuutena on se, että sillä voi tehdä erittäin yksityiskohtaisia malleja. (Daniele 2008.)

Polygonaalisen mallinnuksen toisenlainen muoto on laatikkomallinnus, jossa mallin rakentaminen aloitetaan laatikkoprimitiivistä. Laatikkomallinnus on yksi suosituimpia mallinnustapoja, koska sitä on pitkään pystytty käyttämään lähes kaikissa 3D-mallinnusohjelmissa. Mallinnustapa on tehokas ja nopea, sekä siihen löytyy paljon materiaalia. (Daniele 2008.)

Korkealaatuisten kovapintaisten mallien tekemiseen saatetaan käyttää myös alajaottelua (*Subdivision Surface Modeling*), jossa yksinkertaiselle polygoni-mallille generoidaan pehmeämpi pinta. Prosessissa karkea 3D-malli hajotetaan useampaan osaan, jonka takia malli alkaa näyttämään pehmeämmältä (Russo 2005). Pelien 3D-mallintajat käyttävät tätä tekniikkaa mekaanisten kohteiden mallintamisessa, jolloin korkealaatuista versiota käytetään mallin normaalikarttana (Environment Artist 2B 2016).

3.3.2 Digitaalinen veisto

Yksi mallinnustekniikoista on digitaalista veisto, jota voi verrata saven muokkaukseen. Veistäminen antaa käyttäjälle vapaat kädet muovata hahmoa erilaisilla toiminnoilla, kuten vetämällä, tasoittamalla tai työntämisellä. Tällä tekniikalla pystyy tekemään paljon enemmän yksityiskohtia kuin polygoni-mallinnuksella on mahdollista, eikä käyttäjän tarvitse huolehtia mallin topologiasta. (Zizka 2014, 6.)

Peleissä muovausta käytetään useimmiten tekemään korkealaatuisia normaalikarttoja malleille. Normaalikartta antaa illuusion yksityiskohtaisemmasta mallista, koska pelin valaistus osaa laskea pinnan varjostukset normaalikartan mukaan. Yleisimmin työvaiheina on ensin luoda yksinkertaistettu 3D-malli, joka sen jälkeen muokataan yksityiskohtaisemmaksi muovausohjelmalla ja joka lopuksi tallennetaan normaalikartaksi, jota voidaan asettaa alkuperäisen yksinkertaisemman mallin päälle. (Rabin 2009.)

3.3.3 Muut tekniikat

Automaattisempaan mallien tekemiseen käytetään proseduraalista mallintamista. Proseduraalinen mallintaminen tarkoittaa algoritmillisesti generoituja malleja. Proseduraalista mallinnusta käytetään useimmiten monimutkaisten tai orgaanisten mallien tekemiseen, jotta niitä pystytään generoimaan nopeasti. Esimerkiksi peleissä apuna käytetty SpeedTree-ohjelma käyttää fraktaalipohjaista algoritmia, joka pystyy generoimaan uniikkeja puita ja pensaita. CityEngine on samankaltaista tekniikkaa käyttävä sovellus, jolla voi generoida proseduraalisesti kaupunkikuvaa. (Slick 2014b.)

Peleissä vähemmän käytetty mallinnustekniikka on NURBS-mallintaminen. NURBS-mallinnuksella pystytään kuvaamaan epäsäännöllisiä pintoja, joita on vaikea toteuttaa polygoneilla. NURBS-mallinnusta ei käytetä pelimallinnuksessa usein, koska useammat pelimoottorit eivät tue sen käyttöä. (Rabin 2009.)

Toisenlainen nopeaan lopputuloksen saamiseen käytetty tekniikka on 3D-skannaus. 3D -skannaustekniikalla voidaan skannata todellisia objekteja valokuvarealistisesti. Objekti analysoidaan jokaisesta suunnasta, josta se pystyy generoimaan tarkan polygoni- tai NURBS-pohjaisen verkon. (Slick 2014b.) 3D-skannattuja objekteja on mahdollista käyttää peleissä, mutta se vaatisi mallin polygonimäärän laskemista paljon pienemmäksi, sillä todellisessa koossa mallit olisivat liian raskaita.

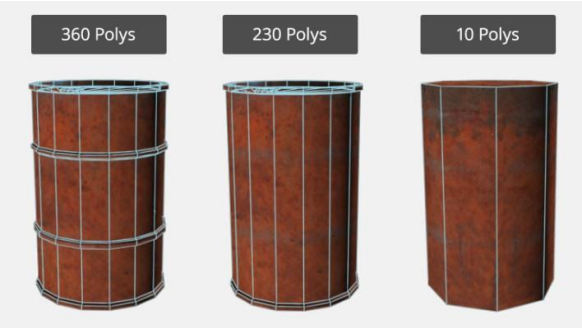
3.4 Optimointi

Pelin optimoinnilla tarkoitetaan sitä, että pelin suorituskykyä pyritään saamaan parhaimmaksi mahdolliseksi kohdealustaa varten. Optimoinnin perusperiaatteena on, että pelin tulee näyttää hyvältä sekä toimia hyvin. Optimointi tehdään useimmiten pelimoottorin puolella ja siitä on yleensä vastuussa ohjelmoitsija. Kuitenkin 3D-mallintajan tulee tietää optimoinnin perusteista varsinkin 3D-mallien tekemisen puolesta. (Ahearn 2008.)

Peleissä mallinnus on rajoitettu laitteiston ja pelimoottorien tehon mukaan. Nykyaikana monien tietokoneiden sekä konsolien kyvyt mahdollistavat monimutkaisemmankin grafiikan teon, mutta pelin sujuva toimiminen vaati silti polygonien määrän rajoituksien huomioimista. Polygonien määrä riippuu pelialustasta, pelimoottorista ja optimoinnista (Rabin 2009). Useimmiten pelit renderöidään reaaliaikaisesti ja peleissä tulee yrittää pitää ruudunpäivitysnopeus mahdollisimman nopeana. Monissa peleissä jopa pienikin hitaus ruudunpäivityksessä voi tehdä pelistä hankalan pelattavaksi. Nykyajan konsolit, kuten Playstation 4 ja Xbox One, mahdollistavat paljon yksityiskohtia, mutta vanhojen pelien tarkkaan rajattu polygonien määrä on erittäin ajankohtainen asia mobiilipelien luomisessa. (Masters 2014a.) Yleensä yksityiskohtia peleissä pyritään tuomaan muilla keinoilla esille. Esimerkiksi yksityiskohtia voidaan tuoda käyttämällä tekstuurikarttoja, joissa on informaatiota korkeuseroista. Esimerkiksi harmaasävyisellä bump-kartalla voidaan luoda illuusiota epätasaisista pinnoista. Samankaltaisesti toimii myös normaalikartta, mutta siinä käytetään värillistä tekstuurikarttaa. Normaalikartan avulla pystytään tekemään yksityiskohtia, jotka vaikuttavat mallin varjostuksiin. Kolmas karttatyyppi on displacement-kartta, joka pystyy muokkaamaan mallin geometriaa. (Assaf 2015.)

Peleissä usein käytetty menetelmä on LOD-tasojen (*Level of Detail*) käyttö. Tämä tarkoittaa sitä että yhdelle peliobjektille on määritelty useampia versioita mallista, joissa yksityiskohtien määrä vaihtelee (kuva 5). Mitä kauempana pelaaja on mallista, sitä yksinkertaisempi malli on näkyvissä. Kun pelaaja tulee lähemmäksi kyseistä mallia, se vaihtuu yksityiskohtaiseen versioon. Tämä tekniikka mahdollistaa sen, että pelikentän ei tarvitse heti renderöidä kaikkia ruudulla näkyviä yksityiskohtaisia malleja kerralla, vaan se voi renderöidä vain lähimpänä olevat tarkasti. Kauempana olevat mallit ovat sen verran kaukana pelaajasta, että pelaaja ei pysty huomaamaan eroa. Samankaltaista tekniikka käytetään myös ensimmäisen persoonan peleissä, jolloin pelaajan kädessä oleva ase käyttää yksityiskohtaisempaa mallia kuin muut pelissä näkyvät hahmot. (Masters 2014a.) Mallien LOD-tasoja voidaan tehdä joko manuaalisesti tai automaatti-

sesti. 3D-mallinnusohjelmissa on usein mukana asetuksia, joiden avulla mallien polygoni määrää voi muuttaa. Olemassa on myös pelkästään automaattiseen optimointiin tarkoitettuja ohjelmia. Esimerkiksi Simplygon sovelluksen avulla peliin tulevasta 3D-mallista voi tehdä useampia yksinkertaistettuja LOD-tiedostoja omien asetuksien mukaisesti. (Simplygon 2016.)



Kuva 5. Kolme LOD-tasoa (Masters 2014a).

4 3D-OHJELMISTOT JA TYÖKALUT

Pelien 3D-grafiikan toteutukseen on olemassa useampia ohjelmistoja ja aputyökaluja. Ohjelmistojen runsas määrä saattaa kuitenkin hankaloittaa sopivan työkalun löytämistä. Tässä luvussa esitellään suosituimpia 3D-mallinnusohjelmia ja muita 3D-grafiikan kannalta hyödyllisiä työkaluja. 3D-mallinnusohjelmapaketeista esitellään niiden tärkeimpiä ominaisuuksia sekä vertaillaan hintatasoa. 3D-mallinnuksen apuohjelmilla tarkoitetaan tässä työssä ohjelmia, jotka eivät vastaa kokonaisesta mallinnustyöstä vaan ovat erikoistuneet pelkästään tekemään tietynlaisia tehtäviä 3D-mallinnuksen osalta. Näitä ohjelmia ovat esimerkiksi veistämiseen ja teksturointiin tarkoitetut ohjelmat. Esille otetaan myös ohjelmia, jotka kehittävät malleja proseduraalisilla menetelmillä. Tässä luvussa ei esitellä 2D-kuvakäsittelyohjelmia, koska luvussa keskitytään vain 3D-grafiikkan erikoistuneita ohjelmia. Luvun loppupuolella esitellään käytetyimpiä pelimoottoreita ja niiden ominaisuuksia varsinkin grafiikan teon kannalta.

4.1 3D-mallinnusohjelmat

3D-mallinnusohjelma on 3D-graafikon päätyökalu mallinnustyön toteuttamisessa. Useammat suuremmat mallinnusohjelmat ovat kokonaisia paketteja, joista löytyy työkaluja moneen eri tarpeeseen. 3D-mallinnusohjelmapaketit soveltuvat yleensä esimerkiksi mallin rakentamiseen, UV-kartan tekemiseen, luurangon rakentamiseen, animaatioiden toteutukseen ja renderöintiin. Nykyaikana useammat mallinnusohjelmat ovat yhä enemmän toistensa kaltaisia, joten niiden eroavaisuuksia on vaikea huomata. Käytännössä ohjelman valinta ei vaikuta suuresti lopputulokseen, vaan tärkeintä on löytää työskentelytapoihin soveltuva ohjelma jonka hintasuhde on käyttäjälle sopiva.

4.1.1 3ds Max

Autodesk 3ds Max on Autodeskin kehittämä ohjelmisto, joka on yksi käytetyimpiä ammattitason 3D-mallinnusohjelmapaketteja viihde-alalla. Ohjelmiston ominaisuudet ovat keskitetty varsinkin mallinnukseen, animointiin ja renderöintiin. Ohjelma on saatavilla vain Windows käyttöjärjestelmälle. (Autodesk 2016a.) 3ds Max on perinteisesti käytetyin 3D-mallinnusohjelma pelialalla, vaikkakin nykyään myös muiden ohjelmistojen työ-

kalut ovat kilpailukykyisiä. 3ds Maxia pidetään yleisesti helppokäyttöisenä ohjelmana, sillä siinä on selkeät mallinnustyökalut. Tämän takia ohjelma on hyvä myös pelien 3D-mallien tekemiseen. Käyttöliittymän selkeyden takia uusien käyttäjien on helppo päästä käsiksi ohjelmaan. (Masters 2014b.)

3ds Maxin käyttö perustuu nykyään kuukausi- tai vuosimaksuihin. Kuukausittainen hinta 3ds Max 2017 versiolle on 242 euroa, mutta Autodeskillä on myös erilaisia hintavaihtoehtoja yhden, kahden tai kolmen vuoden lisensseille. Hintansa puolesta ohjelma on tarkoitettu ammattilaiskäyttöön, joten se on melko kallis vaihtoehto yksittäisille käyttäjille tai pienille pelifirmoille. Tutustumiskäyttöön ohjelmasta on ilmainen 30 päivän kokeiluversio, sekä opiskelijoille kolmen vuoden opiskelijalisenssi ei-kaupalliseen käyttöön. (Autodesk 2016b.)

4.1.2 Blender

Blender on avoimen lähdekoodin 3D-mallinnusohjelmisto, joka toimii Windows, OS X ja Linux käyttöjärjestelmissä. Toisin kuin muut suosituimmat 3D-mallinnusohjelmistot, Blender on täysin ilmainen. Blenderiä ei yleensä käytetä suurimmissa pelistudioissa, mutta se on erittäin suosittu varsinkin indie pelinkehittäjien ja harrastajien keskuudessa. Blenderin käyttäjiä on paljon, joten ohjelma ympärillä on suuri ja aktiivinen yhteisö. (Masters 2014b.)

Blender on pidetty varsinkin sen hyvien mallinnustyökalujen takia. Pelkästään pelien mallien tekemiseen ohjelma on yleensä hyvä vaihtoehto. Blenderistä löytyy muiden 3D-mallinnusohjelmien tavoin myös työkalut teksturointiin, animointiin, renderöintiin ja riggaamiseen. Blenderissä on myös sisäänrakennettu yksinkertainen pelimoottori, jota pystyy käyttämään prototyyppien testaamiseen. (Masters 2014b.)

4.1.3 Maya

Maya on 3ds Maxin rinnalla käytetyimpiä ammattitason mallinnusohjelmistoja, joka on myös Autodeskin omistuksessa. Myös Mayassa on perustyökalut animointiin, mallinnukseen sekä renderöintiin, mutta se tunnetaan varsinkin tehokkaana ohjelmana animaatioiden ja riggauksen tekemisessä. Maya toimii sekä Windows, OSX ja Linux käyttöjärjestelmissä, joten se on vapaammin käytettävissä eri laitteilla. Mayan nykyisin tar-

joamat hyvät mahdollisuudet tekevätkin siitä yhä tärkeämmän ja suositumman työkalun pelialalla. (Masters 2014c.)

Autodeskillä on myös yksinkertaistettu Maya LT -versio, joka on tarkoitettu varsinkin indie- ja mobiilipelinkehittäjille. Maya ja Maya LT jakavat samankaltaisen käyttöliittymän, joten ohjelmistot eivät käytännössä suuresti eroa toisistaan. Maya LT:n työkalut ovat tarkoitettu varsinkin peleihin mallintamiseen. Ohjelmassa on joitakin rajoituksia polygonien määrän suhteen eikä siihen sisälly renderöinti-työkaluja. (Masters 2014b.)

Myös Mayan käyttö perustuu kuukausi- tai vuosihintoihin. Kuukausittainen hinta Mayalle on 246 euroa, sekä edullisemmin voi ostaa useamman vuoden lisenssejä. Maya LT-versiota myydään 36,90 euron kuukausihinnalla, jonka takia se on hyvä edullisempi vaihtoehto. Steamistä ostettaessa Maya LT -ohjelmaan sisältyy myös Stingray-pelimoottori. (Autodesk 2016b.)

4.1.4 Modo

Modo on The Foundryn kehittämä 3D-mallinnusohjelmisto, josta löytyy myös tarvittavat työkalut mallinnukseen, animointiin ja renderöintiin. Pelinkehityksessä Modoa käytetään useimmiten tasojen ja peliobjektien luomiseen. (The Foundry 2016a.)

Modo on kertamaksulla ostettava ohjelma ja sen hinta on 1 459 euroa. Ohjelmasta on myös maksullinen opiskelijalisenssi, joka on vain ei-kaupalliseen käyttöön. Version voi kuitenkin päivittää halvemmalla hinnalla täysiversioksi. Ohjelmasta saa myös kokeiluversion käyttöön ilmaiseksi 30 päivän ajaksi. (The Foundry 2016a.) Täysiversion lisäksi Modosta on myös myös indie versio, jonka hintana on 14,99 euroa kuukaudessa. Modo indie sisältää perinteisten mallinnustyökalujen lisäksi myös mahdollisuudet digitaaliseen muovaukseen ja renderöintiin. (Steam 2016.)

4.1.5 Vertailu

Kokonaisuudessaan nykyaikaisissa 3D-mallinnusohjelmapaketeissa ei ole suuria eroja. Mallinnuksessa tärkeintä on se, että lopputulos on halutunlainen, eikä se millä ohjelmalla se on tehty. Ohjelman valintaan vaikuttavat suurimmaksi osaksi hintataso ja henkilökohtaiset mieltymykset. Lähes kaikista 3D-mallinnusohjelmista on ilmaiset kokeiluversiot, koska ohjelman käytön kokeilu on tärkein osa ohjelman valintaa. Myös opiskeli-

jalisenssit ovat hyviä ohjelmiin tutustumisessa, sekä ne ovat hyödyllisiä myös ohjelman ominaisuuksien opettelussa ja portfolion kasvattamisessa.

Vertailutaulukkoon on kerätty eri 3D-mallinnusohjelmien pääasiallisia ominaisuuksia ja hintavaihtoehtoja. Lisäksi vertailussa on ohjelmien tarjoamat opiskelijaversiot ja kokeiluversiot. Hintavertailuun on laitettu vain kokonaishinta, kuukausihinta tai vuosihinta. Hintavertailusta on jätetty pois useamman vuoden lisenssit. Hinnat saattavat vaihdella riippuen ostopaikasta ja hetkestä. (Taulukko 1.)

Taulukko 1. 3D-mallinnusohjelmien vertailu

	Alustat	Käyttötarkoitukset	Opiskelijaversio	Kokeiluversio	Hinta
3ds Max	Microsoft Windows	Mallinnus, animointi, valaistus, renderöinti	3 v ilmainen	30 pv	242 € kk tai 1 936 € vuosi
Blender	Microsoft Windows, Mac OS X, Linux, BSD, Solaris, AmigaOS 4, MorphOS	Mallinnus, animointi, valaistus, renderöinti, veistäminen	-	-	Ilmainen
Maya	Microsoft Windows, Mac OS X, Linux	Mallinnus, animointi, valaistus, renderöinti, tehosteet, veistäminen	3 v ilmainen	30 pv	242 € kk tai 1 936 € vuosi
Maya LT	Microsoft Windows, Mac OS X	Mallinnus, animointi, veistäminen	3 v ilmainen	30 pv	36,90 € kk tai 302,50 € vuosi
Modo	Microsoft Windows, Mac OS X, Linux	Mallinnus, animointi, renderöinti, veistäminen	180 € vuosi	30 pv	1 459 €
Modo indie	Microsoft Windows, Mac OS X	Mallinnus, animointi, renderöinti, veistäminen	-	-	14,99 € kk

4.2 3D-mallinnuksen apuohjelmat

3D-graafikko pystyy yleensä tekemään kaiken tarvittavan työn yhden 3D-mallinnusohjelman ja kuvankäsittelyohjelman avulla. 3D-mallinnukseen käytetyissä pääpaketeissa ei kuitenkaan ole aina tarpeeksi hyvin erikoistuneita työkaluja esimerkiksi teksturointiin tai veistämiseen. Monet apuohjelmat ovat tärkeitä korkealaatuisten mallien toteutuksessa, sekä niiden avulla työtahtia on helppo nopeuttaa. Toisaalta mallit saatetaan myös toteuttaa täysin proseduraalisesti, varsinkin jos tietynlaisia samankaltaisia malleja on paljon.

4.2.1 Veisto-ohjelmat

Digitaaliseen veistämiseen tarkoitetut ohjelmat sisältävät yleensä monipuolisia ominaisuuksia muovaamisenkin lisäksi. Yleensä veisto-ohjelmien avulla pystyy myös maalaamaan tekstuurit suoraan 3D-mallin päälle tai tehdä tekstuurikarttoja tarkkalaatuisen mallin avulla. (Slick 2016.) Digitaalisen muovausohjelman käytön hallinta on nykyään lähes välttämättömyys hyvälle 3D-mallintajalle. Joissakin tavallisissa mallinnusohjelmissa on myös sisäänrakennettuja veisto-ominaisuuksia, näitä ohjelmia ovat esimerkiksi Modo, Cinema 4D ja Blender. (Marshall 2014.)

Pixologicin kehittämä ZBrush on suosittu digitaaliseen muovaamiseen tarkoitettu ohjelma. ZBrushista löytyy työkalut esimerkiksi mallipohjan kokoamiseen sekä siveltimiä mallin muovaamiseen. ZBrushissa pystyy myös maalaamaan hahmon päälle. Pixologicilla on myös ilmainen hieman yksinkertaisempi muovausohjelma Sculptris. (Marshall 2014.)

Toinen suosittu muovausohjelma Mudbox, joka on Autodeskin kehittämä sovellus. Autodeskin ohjelmana se toimii saumattomasti Mayan ja 3ds Maxin kanssa yhdessä. Mudboxissa ei ole riittäviä työkaluja tekemään mallipohjia, joten normaalisti muovattava mallipohja luodaan ensin mallinnusohjelmassa, jonka jälkeen se siirretään Mudboxiin. Mudbox on tunnettu varsinkin hyvistä maalausominaisuuksistaan sekä yksinkertaisesta käyttöliittymästä. (Marshall 2014.)

4.2.2 Teksturointi- ja maalausohjelmat

Tekstuurien tekeminen on yksi tärkeimpiä osia mallinnustyössä. Nykyaikana on tarjolla monia ohjelmia auttamaan ja nopeuttamaan tekstuurien tekemistä. Tekstuurien tekemiseen voidaan käyttää joko maalausohjelmia tai tekstuurien generointiin tarvittavia ohjelmia. Tekstuurien maalaamisessa käytetään yleensä tavallista kuvankäsittelyohjelmaa, mutta monet 3D-maalausohjelmat antavat mahdollisuuden maalata suoraan hahmon päälle. Tekstuurien generoimiseen tarkoitetut ohjelmat taas ovat hyödyksi varsinkin realististen tekstuuripintojen ja niiden normaalikarttojen tekemisessä. Tekstuurien tekemiseen erikoistuneita ohjelmia ei välttämättä tarvita, mutta ne auttavat nopeuttamaan työprosessia merkittävästi.

3D-maalaamisen osalta pelin kehityksessä on tällä hetkellä suosiossa varsinkin Allegorithmicin kehittämä Substance Painter, jonka avulla voi tehdä tekstuureja sekä renderöintejä. Mallin päälle pystyy tekemään tekstuureja ja erilaisia tekstuurikarttatyyppejä suoraan ohjelmassa. Ohjelmassa on myös proseduraalisesti toimivia tekstuurityökaluja. (Allegorithmic 2016.) Toinen tunnettu maalausohjelma on The Foundryn kehittämä Mari. Mari pystyy käsittelemään jopa erittäin suurikokoisia malleja ja tekstuureita. Maalaustyökalujen lisäksi se sisältää monia asetuksia värien muokkaamiseen. Maria käytetään useimmiten varsinkin elokuvateollisuudessa, mutta siitä on nykyään myös tarjolla halvempi indie-versio. Ohjelma toimii myös saumattomasti Modo-mallinnusohjelman kanssa. (The Foundry 2016b.) Mari indien lisäksi toinen edullinen vaihtoehto on BodyPaint 3D, jossa voi käyttää myös esimerkiksi Photoshopin siveltimiä 3D-maalaamiseen. 3D maalaamiseen löytyy työkaluja yleensä myös muovaamiseen tarkoitetuista ohjelmista, näitä ovat esimerkiksi 3DCoat, Mudbox ja ZBrush. (Slick 2016.)

Proseduraalisten tekstuurien tekemiseen Allegorithmicillä on myös Substance Designer. Substance Designerissä on node-pohjainen materiaalityökalu. Ohjelman avulla pystyy generoimaan realistisia tekstuureja nopeasti. (Allegorithmic 2016.) Toisenlainen nopeaan teksturointiin tarkoitettu työkalu on Quixelin DDO Painter, jolla pystyy teksturoimaan lähes automaattisesti. Ohjelmalla pystyy tekemään nopeita tekstuuripohjia, mutta suurempi muokkaaminen vaati paljon aikaa. Quixelilla on myös NDO Painter, joka on Photoshopiin lisättävä laajennus normaalikarttojen tekemiseen maalaamalla. Pelkästään yksinkertaisiin normaalikarttojen tekemisiin löytyy erikoisohjelmia, jotka tekevät suoraan 2D kuvasta normaalikarttoja. Tällaisia ohjelmia on esimerkiksi XNormal ja Crazybump. (Slick 2016.)

4.2.3 Muita työkaluja

3D-mallien tekemiseen on olemassa laaja valikoima mallinnustyötä helpottavia ohjelmia. Toisaalta pelinkehitystyötä voidaan myös nopeuttaa ohjelmistoilla, jotka tekevät valmiita malleja. Valmiita malleja voidaan jälkikäteen muokata erikseen 3D-mallinnusohjelmassa tai käyttää sellaisenaan. Valmiiden mallien käyttö helpottaa työtä varsinkin sellaisten mallien osalta, jotka ovat peruspiirteiltään samankaltaisia, mutta aikaa vieviä toteuttaa.

Monissa peleissä ihmishahmot ovat yleensä tärkeässä osassa, mutta ihmisten mallintaminen on samalla myös hidasta. Esimerkiksi peleissä saatetaan usein tarvita monia

erinäköisiä pelin ohjaamia NPC-hahmoja. Ihmisten mallintamiseen on tehty useampia helpottavia työkaluja, joiden avulla pystyy pääpiirteissään luomaan ihmishahmoja. Esimerkiksi Mixamon Fusea käytetään ihmishahmojen tekemiseen. Sovelluksen avulla pystyy muokkaamaan ihmishahmosta toivotun näköisen vaatteita myöten, sekä lopuksi ohjelma pystyy tekemään hahmolla automaattisen riggauksen. Ohjelmaan voi tuoda myös omatekoisia osia mallinnusohjelmasta. (Mixamo 2016.) Samankaltaisia ohjelmia on myös esimerkiksi Autodeskin Character Generator sekä avoimen lähdekoodin Make Human. Useammissa tämänkaltaisissa ohjelmissa on kuitenkin vielä erittäin rajattuja mahdollisuuksia hahmon muokkaamiseen. Nämä ohjelmat toimivat monesti vain pelkästään aloituspisteenä mallille, joten animointi ja jälkimuokkaus jää mallinnusohjelmalle tehtäväksi. (Maher 2014.)

Peliympäristön tekemistä varten on tehty myös useampia erilaisia ohjelmia. Nämä ohjelmat yleensä generoivat malleja proseduraalisesti. Maaston ja korkeuskarttojen luomiseen on olemassa World Machine, jolla voidaan tehdä realistisen näköisiä maastokarttoja peleihin. Sovellus tekee korkearesoluutioisia verkkomalleja sekä tekstuureja. World Machine toimii hyvin varsinkin Unityn ja Unreal Enginen kanssa. (World Machine 2016.) Toinen proseduraalinen ympäristön luoja on CityEngine, jolla voi generoida kaupunkikarttoja. Se soveltuu sekä visualisointiin ja peleihin (Esri 2016). Peleissä ja elokuvissa suosittu ohjelma on myös SpeedTree, jonka avulla voi luoda proseduraalisesti puita ja kasvillisuutta. SpeedTree antaa sekä manuaalisia että automaattisia vaihtoehtoja kasviston luomiseen. (SpeedTree 2016.)

4.3 Pelimoottorit

Pelimoottori on videopelin ohjelmistokehys, joka tarjoaa lähdekoodit ja työkalut nopean pelin tekemiseen. Pelimoottori antaa mahdollisuuden valmiisiin objektien fysiikkaan, renderöintiin ja tekoälyyn. Pelimoottorien avulla pelinkehitys on nopeampaa ja antaa pelinkehittäjille helpomman tavan tehdä pelejä, ilman että koko ohjelmaa tarvitsee rakentaa itse. (Kalderon 2011.)

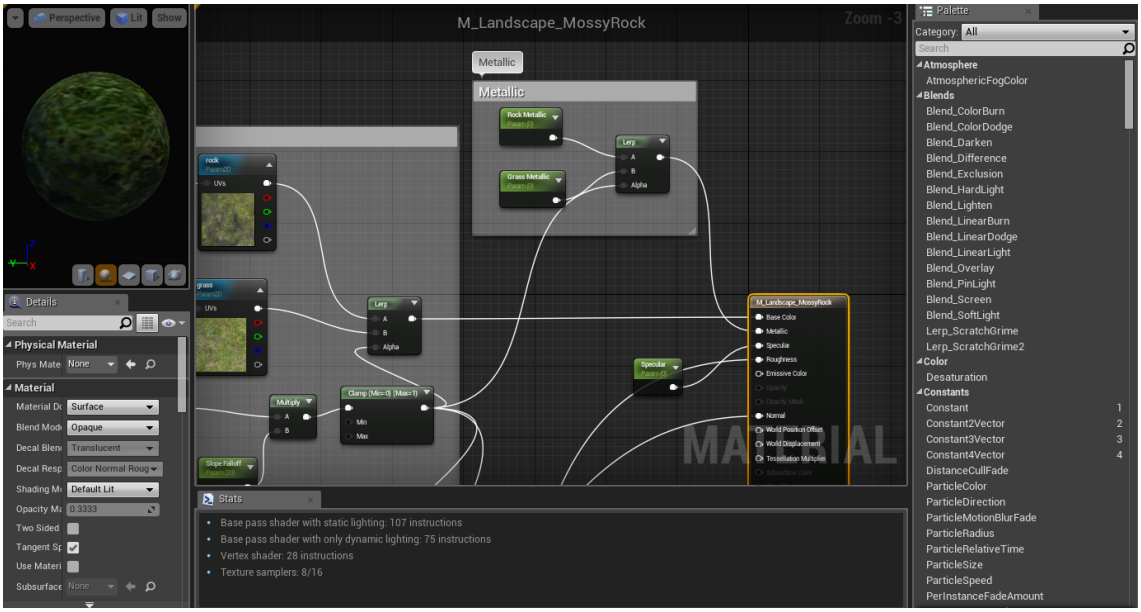
Nykyajan pelimoottorien kehityksessä panostetaan paljon käyttäjäystävällisyyteen. Peliala on kokoajan kasvattanut enemmän kiinnostusta ja pelinkehitykseen on lähtenyt enemmän uusia tekijöitä. Tämän takia nykyaikaisissa pelimoottoreissa houkuttelevuutena on helppokäyttöisyys ja opetuksen laaja tarjonta. Unity ja Unreal Engine ovat juuri siksi parhaimpia vaihtoehtoja, koska niiden käytön aloittaminen on helppoa laajan yh-

teisön ja tutoriaalien ansiota. Molemmissa on myös oma kauppapaikkansa, jossa pelimoottorin käyttäjät voivat myydä omia työkalujaan muille käyttäjille. Vaikka pelimoottorien hallinta ei ole välttämättä tärkeimpiä osuuksia 3D-mallintajalle, mallintaja voi silti olla vastuussa pelikentän kokoamisesta ja mallien siirtämisestä, sekä valaistusten ja materiaalien toteuttamisesta. Useammat pelimoottorit antavat nykyään hyvät työkalut näyttävän grafiikan toteuttamiseen.

4.3.1 Unreal Engine 4

Unreal Engine on Epic Gamesin kehittämä pelimoottori, jonka neljäs versio julkaistiin 2012. Pelimoottorin avulla on tehty useita pelidemoja ja kaupallisia pelejä. Unreal Engine 4 käyttää C++ -ohjelmointikieltä sekä Blueprint järjestelmää, joka mahdollistaa nopeita muokkauksia peliin visuaalisessa käyttöliittymässä. Unreal Engine 4:lla pysty tekemään pelejä mobiililaitteille, tietokoneille ja konsoleille. (Create 3D Games 2015.)

Unreal Enginellä pystyy toteuttamaan näyttävää grafiikkaa, koska siinä on pitkälle kehittyneet grafiikkatyökalut. Unreal Enginen renderöintisysteemi mahdollistaa dynaamisen valaistuksen, monimuotoisten varjojen toteutuksen ja partikkelien simulaation. Unreal Engineen sisältyy myös visuaalinen materiaalieditori, joka mahdollistaa mallien pinnoitteiden luomisen (kuva 6). Työkalun avulla materiaaleille voidaan määritellä esimerkiksi pinnan väri, tekstuuri tai kiiltävyys. (Unreal Engine 2016.)



Kuva 6. Material Editorin käyttöliittymä.

Pelimoottorin kaikki työkalut ovat ilmaiseksi käytettävissä, mutta käytöstä peritään 5% rojaltimaksuja jos pelimoottorilla tehty peli tuottaa yli 3 000 dollaria vuosineljänneksellä. Unreal Enginellä on myös oma Marketplace-kauppa, jossa käyttäjät voivat myydä omia tuotoksiaan toisille käyttäjille. Kaupan tarjonta on kuitenkin vielä tällä hetkellä hyvin rajallinen, mutta myytävät kohteet ovat laadukkaasti toteutettuja. (Greate 3D Games 2015.) Unreal Engine tarjoaa myös useita testattavia pelidemoja, joiden sisältöä saa vapaasti käyttää myös omissa Unreal Enginellä tehdyissä peleissä (Unreal Engine 2016).

4.3.2 Unity

Unity on Unity Technologiesin kehittämä pelimoottori, joka on suosituimpia pelimoottoreita indie pelinkehittäjillä. Ohjelmoinnissa Unityssä käytetään suuremmaksi osaksi C#-, JavaScript, ja Boo-ohjelmointikieliä. Unityssä on myös hyvät optimointimahdollisuudet ja se tukee useampia alustoja. Pelimoottorilla voi tehdä muun muassa PC-, konsoli- tai selainpelejä. (Create 3D Games 2015.)

Unity on tunnettu varsinkin hyvänä vaihtoehtona 2D-grafiikkaan sekä mobiilipeleihin. Viimeisimmän version myötä työkalut ovat kuitenkin kehittyneet yhä enemmän sopivaksi myös konsolipelien kehittämiseen. Uusin versio Unity 5 julkaistiin maa-

liskuussa 2015 ja se toi useita muutoksia grafiikan tekemiseen, sekä myös uudistetun fysiikkamoottorin. Uusimmalla versiolla pystyy renderöimään näyttäviä valaistuksia reaaliaikaisella globaalilla valaistuksella sekä fyysisiä varjostuksia. Pelin muokkaamisen osalta Unity tarjoaa paljon käyttäjien tekemiä lisäosia, työkaluja tai malleja, joiden avulla pelinkehitystä pystytään nopeuttamaan. Unity Assets Storessa on yli 1,5 miljoonaa käyttäjää ja erilaisia myytäviä lisäosia on 15 000. (Greate 3D Games 2015.)

Unityn ammattilaisversio Unity Pro on 75 dollaria kuukaudessa ja siihen sisältyy monia pelintekoa nopeuttavia lisätoimintoja. Unitystä on tarjolla yksityisille käyttäjille ilmainen versio, jossa työkalut ovat hieman rajatummat. Ilmaisversiossa on tulorajoitus, jonka ylitettyä käyttäjän tulee ostaa Unity Pro. Unity Pron oston jälkeen tulorajoituksia ei ole. (Greate 3D Games 2015.)

4.3.3 Muut pelimoottorit

Useammat nykyajan pelimoottorit pyrkivät olemaan samalla tavalla houkuttelevia kuin Unity ja Unreal Engine. Näitä houkuttelevia asioita ovat helppokäyttöisyys, yhteensopivuus ja laaja opetustarjonta. Ilmaisuus on myös yksi tärkeä osa haluttavuutta, johon on siirtyneet useat pelimoottorit.

Yksi tunnetuimpia pelimoottoreita on CryEngine, joka on tunnettu varsinkin sen upeista grafiikkamahdollisuuksista. Uusimmassa versiossa on esimerkiksi vokseli-pohjainen valaistus, dynaaminen vesi sekä uudistunut partikkelisysteemi. CryEngine on tänä vuonna muuttanut lisenssinsä sellaiseksi, että käyttäjä voi maksaa pelimoottorista haluamansa mukaan ilman erillisiä rojaltimaksuja. Myös CryEnginellä on oma kauppapaikkansa käyttäjien tuotoksille. (CryEngine 2016.)

Tällä hetkellä uusin pelimoottori on Autodeskin kehittämä Stingray, joka on tarkoitettu varsinkin pienimmille studioille. Pelimoottori on tarkoitettu varsinkin kehittäjille joilla ei ole paljon kokemusta ohjelmoinnista. Stingrayssa on mahdollista ohjelmoida visuaalisesti käyttäen node-objekteja sekä se toimii saumattomasti Mayan, Maya LT:n ja 3ds Maxin kanssa. Pelimoottoriin pystyy siirtämään materiaaleja tai valoasetuksia suoraan mallinnusohjelmista, ja ne toimivat täysin samalla tavalla. Tämä ominaisuus antaa mahdollisuuden käyttää Stingrayta myös pelkästään reaaliaikaiseen visualisointiin.

Tällä hetkellä Stingrayn saa Maya LT -mallinnusohjelman kanssa yhteisellä kuukausihinnalla. (Creative Bloq 2015.)

Lähiaikoina on tulossa myös Valvelta uusi pelimoottori Source 2, joka tulee olemaan ilmainen ja sisältää nykyaikaisten pelimoottorien ominaisuuksia. Valvelta on aikaisemmin tullut vanhempi pelimoottori Source Engine, joka on ollut suosittu varsinkin pelien modaamisessa. (World of Level Design 2015.) Lisäksi on olemassa myös monia pienempiä avoimen lähdekoodin pelimoottoreita, joita ovat esimerkiksi Panda 3D, Delta3D ja Cube (Kalderon 2011).

5 DIGITAL TIMETREK

5.1 Toimeksianto

Opinnäytetyön käytännönosuuden työnä oli Digital TimeTrek eli Aikavaellus -projekti, joka on digitaalisella alustalla toimiva maailmankaikkeuden syntymisen teoriaa opettava opetuspeli. Projektia tehdään yhteistyössä Turun ammattikorkeakoulun ja Turun yliopiston biokemian laitoksen kanssa. Turku Game Lab toteuttaa suurimman osan pelin digitalisoidusta sisällöstä.

Turku Game Lab on Turun ammattikorkeakoulun ja Turun yliopiston yhteinen kehitysympäristö, jonka tarkoituksena on saattaa yhteen pelialasta kiinnostuneita opiskelijoita yhteiseen työelämälähtöiseen oppimisympäristöön. Turku Game Lab kehittää pelejä ja peliteknologiaa hyödyntäviä sovelluksia, tarjoaa asiantuntija-apua yrityksille sekä tutkii uudenlaisia teknologian hyödyntämistapoja. (Turku Game Lab 2016.)

Digital TimeTrek -sovellusta on tarkoitus käyttää pelillistettynä opetusmateriaalina luonnontieteiden opettamisessa. Sovellus tehdään Unity-pelimoottorilla, ja se tulee lopulta toimimaan ainakin mobiililaitteilla ja tietokoneella. Siitä on myös suunnitteilla toisenlaisia versioita luonnontieteen museoiden tai planetaarioiden käyttöön. Sovelluksen käyttöliittymässä on tarkoitus näyttää kosmisen sekä geologisen ajan pääkohtia visuaalisesti aikajanana. Aikajana alkaa alkuräjähdyksestä ja kuvaa maapallon kehitystä syntymisestä nykyhetkeen asti. Aikajanassa tulee olemaan noin 10 jaksoa, joihin sisältyy minipelejä tai tehtäviä. Käyttöliittymässä eri ajat näkyvät pieninä kolmiulotteisina kuplina, joissa on kuvattu ajanjakson päätapahtumia visuaalisesti. Tämän toimeksiannon tarkoituksena oli luoda näitä käyttöliittymään tarvittavia 3D-malleja kuvaamaan erilaisia maailmankaikkeuden syntymistä kuvaavia tapahtumia.

5.2 Työvaiheet

Projektin työvaiheisiin kuuluivat mallien ja ympäristöjen suunnittelu sekä sen jälkeen niiden toteutus. Projektista oli jo alkuperäisiä suunnitelmia siihen tulevista ympäristöistä ja kentistä, kuitenkin nämä ideat tarvitsivat silloin vielä täydennystä. Aluksi määriteltiin mallinnustyöhön liittyvät rajoitteet, joiden pohjilta suunnitteluvaihe voitiin aloittaa. Tässä

luvussa kuvaillaan projektiin liittyvää suunnittelutyötä, työkalujen valintaa ja mallien toteutusta.

5.2.1 Suunnittelu

Ennen varsinaista mallintamista piti suunnitella, minkälaisia ympäristöjä käyttöliittymässä tulee näkymään ja miten niitä pystytään kuvaamaan visuaalisesti. Alkuperäisissä suunnitelmissa on useampia eri opetusjaksoja. Näitä ovat esimerkiksi alkuräjähdys, tähdet ja planeettakunnat, aurinkokunta, maan syntyminen sekä evoluution eri vaiheet. Ensimmäisenä työtehtävänä oli tehdä konseptikuva, jossa jokaista eri vaihetta kuvastetaan yhdellä kuvalla (kuva 7). Näiden kuvien pohjalta olisi siten tarkoitus tehdä lopulliset 3D-mallit, jotka näkyvät sovelluksen valikossa. Käytännössä tämä tapahtui siten, että Internetistä haettiin kuvia jokaisesta eri aikakaudesta ja valittiin konseptiin sellaisia kuvia, jotka parhaiten kuvastivat yleiskuvaa ajasta. Esimerkiksi jääkauden kuvaksi valikoitui kuva kahdesta mammutista lumisessa maisemassa, sillä mammutit ovat ikonisin kuvaus aikakauden eläinlajeista. Dinosaurusten aikaa kuvaa kuvat kahdesta tunnetusta dinosauruslajista: tyrannosaurus ja sauropodi.

Kuva 7. Konseptikuva eri aikakausista.

Ihmisten aikakauden visualisointi oli haasteellisempaa, sillä aikakausi tulisi kattamaan aiheen nykyihmisten syntymisestä nykykulttuurin asti. Yhdellä kuvalla olisi vaikea kuvata koko ihmisten aikakautta, sillä ihmisten kehityksen historia on paljon paremmin tunnettu. Ihmisten teknologia ja arkkitehtuuri on kehittynyt paljon, joten yksinkertaista ku-

vaustapaa oli vaikea löytää. Suunnitelmissa oli käyttää joko nykyajan suurkaupunkeja kuvaavaa maisemaa tai kuuluisia muinaisrakennelmia (kuva 8). Lopulta viimeiseksi ideaksi muodostui asetelma, jossa olisi koottu kuuluisimpia rakennuksia yhteiseen kaupunkimaisemaan. Siten yksi kuva pystyi visualisoimaan kehityksen asteita monipuolisemmin.

Kuva 8. Konseptikuva ihmisten aikakaudesta.

Konseptikuvien jälkeen opinnäytettä varten toteutettaviksi ympäristöiksi valikoitui ihmisten aikakausi ja jääkausi. Ympäristöjen tulisi olla halkaistun ympyrän päällä, ja koko asetelma olisi lopuksi läpinäkyvän kuplan sisällä. Siten maisema näyttäisi siltä, kuin se olisi pieni leikkaus kyseisestä aikakaudesta, joka leijailee käyttöliittymän avaruudessa. Halkaistu ympyrä, joka toimisi kyseisen alueen maastona, tuli olla halkaisijaltaan 10 m, jotta mallit pysyisivät samassa mittasuhteessa. Yhden ympäristön rajoituksena oli, että se saisi olla maksimissaan 20 000 verteksiä. Käyttöliittymässä tulisi näkymään niin monia eri ympäristöjä, että sovelluksen nopean toimivuuden vuoksi yhden kokoluokkaa piti rajata. Käytännössä myös käyttäjä tulisi näkemän yhden ympäristön aina hyvin pienenä, joten mallien tuli olla mahdollisimman yksinkertaisia.

5.2.2 Työkalujen valinta

Työkalut toimeksiannon tekemiseen valikoituivat niiden saatavuuden ja käyttötottumuksien mukaisesti. Hahmojen ja ympäristön mallintamiseen valittiin 3ds Max, sillä sen käyttö oli jo entuudestaan tuttu. 3ds Maxissa on myös erinomaiset työkalut nopeiden mallien rakentamiseen. Se soveltui työhön hyvin, sillä toimeksiannossa oli paljon erilaisia malleja toteutettavana. 3ds Maxilla oli tarkoitus toteuttaa mallit, UV-kartta, riggaus ja animaatiot. Tarkoituksena oli myös koota ympäristöjen mallit 3ds Maxin sisällä. Myös renderöintiä tarvittiin hahmottamaan suunnitelmia siitä, miltä asetelma tulisi näyttämään lopullisesti itse sovelluksen sisällä.

Teksturointityöhön valittiin Corel PaintShop Pro X6 -kuvankäsittelyohjelma, sillä sen työkalut soveltuivat hyvin kuvankäsittelyä varten ja se oli myös valmiina asennettuna mallinnukseen käytetyllä tietokoneella. Paint Shop Pron ominaisuudet vastaavat hyvin paljon Adobe Photoshopia, joten sillä pystyi tekemään kaiken tarvittavan työn tekstuurien toteutuksessa. Teksturoinnin apuun valikoitui Textures-sivusto, josta saa rekisteröidyttyä ladata ilmaiseksi pienempi resoluutioisia tekstuureita. Tekstuurit saivat olla pieniä, sillä lopullisen tekstuurien koko sovellusta varten tulisi olemaan melko pieniä. Näitä tarvittavia tekstuureja oli esimerkiksi kaupunkirakennusten tekstuurit, joita tarvittiin rakennusten toteutukseen. Malleille ei tarvittu normaalikarttoja tai muita yksityiskohtia lisääviä menetelmiä, joten mallinnusohjelman ja kuvankäsittelyohjelman lisäksi ei ollut tarvetta muille sovelluksille.

Alkuperäisissä suunnitelmissa oli myös tehdä SpeedTree-sovelluksella kasvillisuutta, mutta ohjelmalla tehtävät kasvit olisivat olleet liian raskaita pelin vaatimuksiin nähden. Lopulta siis kasvillisuus päädyttiin tekemään mallintamalla ja käyttäen kaksiulotteisia tasoja. Suunnitelmissa oli myös itse mallintaa ja teksturoida planeettoja, mutta Unityn Assets Store -kauppapaikasta löytyi työkalu, jonka avulla pystyi generoimaan planeettoja proseduraalisesti.

5.2.3 3D-mallien toteutus

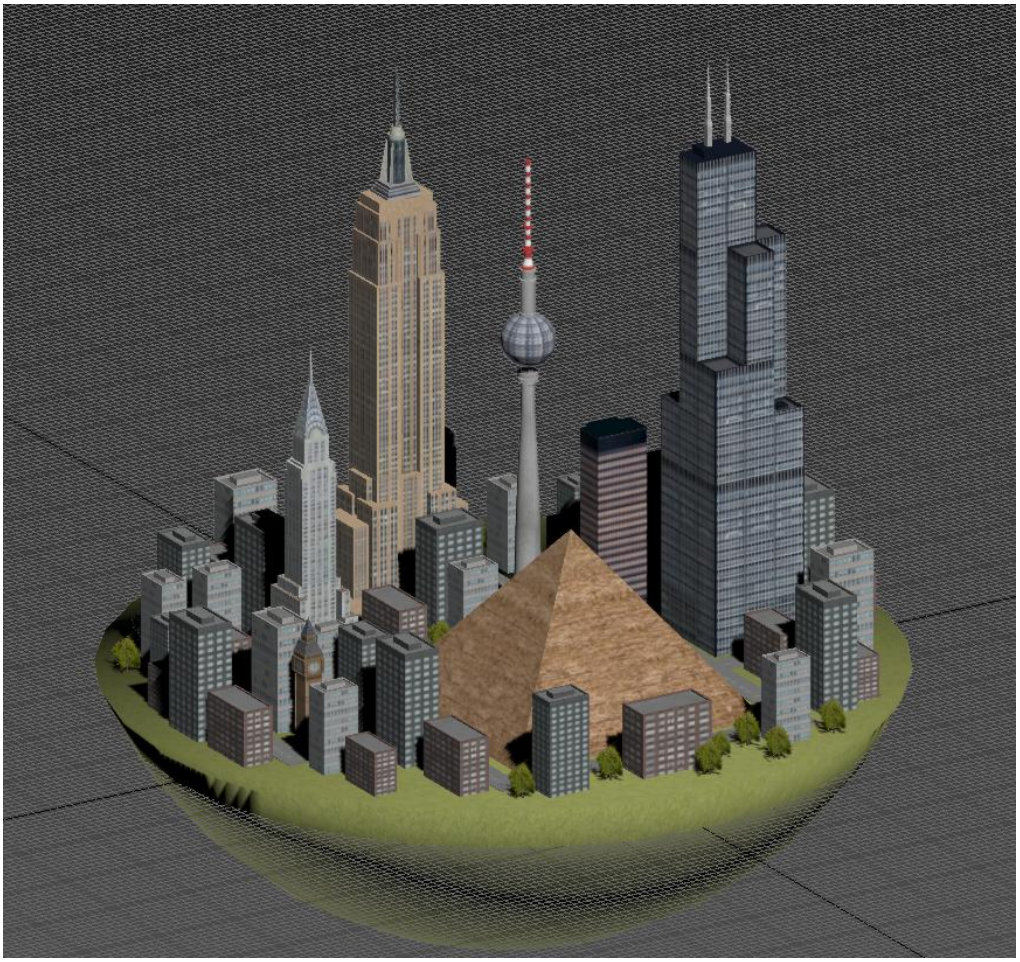
Mallintaminen aloitettiin ihmisten aikakausi teemasta, johon oli tarkoituksena tulla leikkaus kaupunkikuvasta. Inspiraationa asetelmalle oli esimerkiksi Electronic Artsin *SimCity BuildIt* -pelin logon tyyli, jossa suuret rakennelmat ovat kerätty tiukasti ryhmään.

SimCity BuildItin rakennukset toimivat myös muutenkin hyvänä inspiraationa asetelmalle, koska peli on tarkoitettu mobiililaitteille ja sen takia myös sen rakennuksen mallit olivat yksinkertaistettuja. Lopullisten suunnitelmien mukaan kaupunkiasetelmassa tulisi näkymään kuuluisia rakennuksia eripuolilta maailmaa. Rakennukset valittiin niiden tunnettavuuden ja yksinkertaisuuden mukaan. Mukana oli kuitenkin myös joitakin vähemmän kuuluisiakin rakennuksia. Kaupunkimaisemassa näkyviksi rakennuksiksi valittiin Kheopsin pyramidi, Empire State Building, Willis Tower, Seagram, Chrysler Building, Berliinin televisiotorni ja Big Ben.

Rakennuksia lähdettiin toteuttamaan laatikkomallinnustekniikalla, koska laatikko primitiivistä mallin aloittaminen oli selkeästi yksinkertaisin tapa toteuttaa suorakulmaisia rakennuksia. Mallien rakentamisessa käytettiin apuna rakennusten kaavakuvia, jotka auttoivat pitämään mittasuhteet todellisuutta vastaavina. Mittasuhteet auttavat siihen, että rakennus näyttäisi jopa yksinkertaistettuna silmämääräisesti tutun näköisenä. Monissa rakennuksissa oli paljon yksityiskohtia, joita piti jättää tekemättä yksinkertaisuuden pitämiseksi. Esimerkiksi ikkunoita tai koristepylväitä ei mallinnettu taloihin erikseen, vaan kaikki yksityiskohdat toteutettiin pelkillä tekstuureilla. Big Benissä myös kellotaulukin on toteutettu pelkillä tekstuureilla (kuva 9). Läheltä katsottuna mallit saattoivat olla melko karkeitakin, mutta kauempaa katsottuna pieniä yksityiskohtia ei kuitenkaan huomaisi.

Kuva 9. Big Benin 3D-malli teksturoituna.

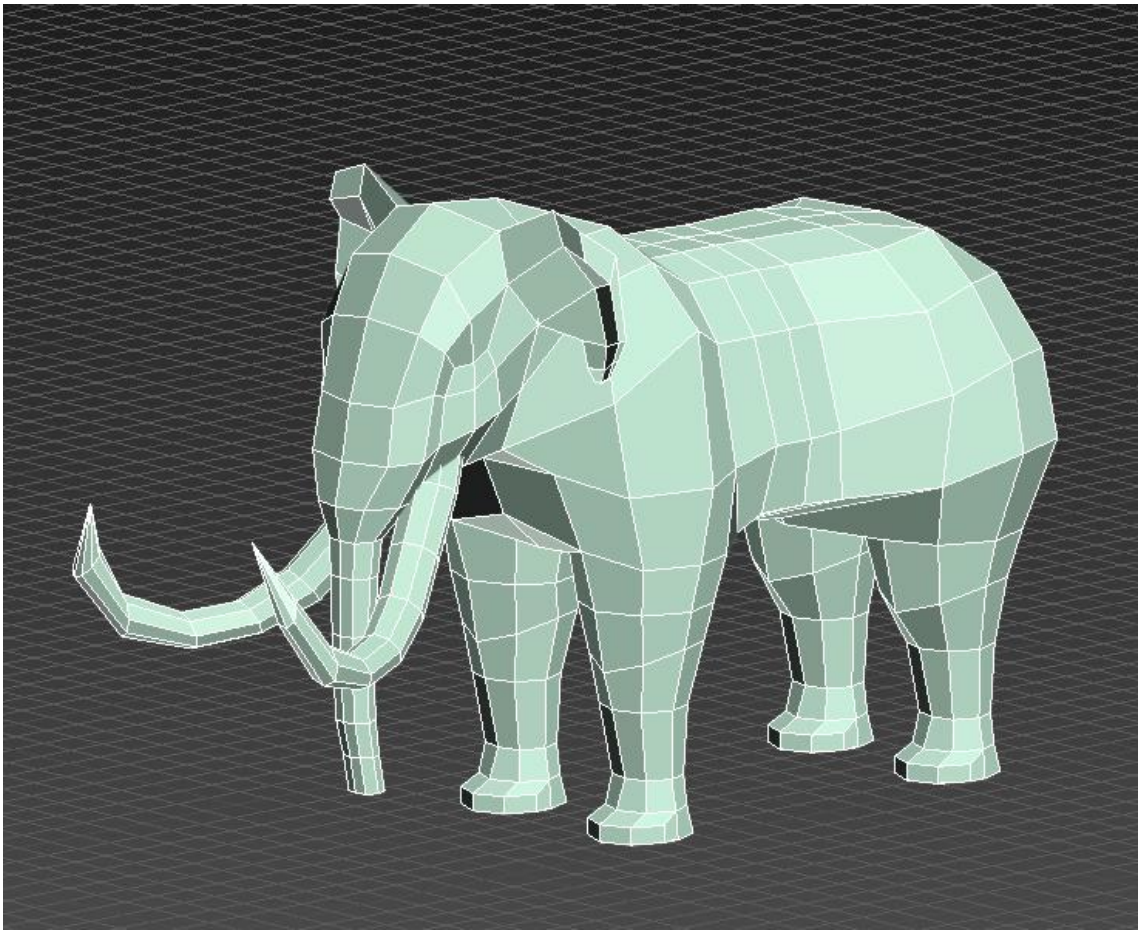
Kun rakennusten mallit olivat valmiit, niistä oli kuitenkin vaikea saada näyttävää asetelmaa. Maastossa jäi liikaa tyhjää tilaa, eikä rakennuksia voinut suurentaa, koska silloin ne eivät olisi mahtuneet kuplan sisälle. Tyhjän tilan täytteeksi ja elävöittämiseksi piti tehdä vielä tavallisia kerrostaloja. Käytännössä tehtiin kolme erilaista kerrostaloa, joita sijoitettiin moninkertaisina ympäristöön. Samanlaisten talojen kokoa vaihdeltiin, jotta nopealla silmäyksellä talot näyttäisivät erilaisilta. Talojen väliin aseteltiin myös symmetrinen autotie. Autotie tehtiin erillisenä mallina, koska se olisi ollut vaikea teksturoida maastoon. Erillisenä tietä pystyi myös helposti muokkaamaan ja liikuttamaan. Lopuksi maastolle vaihdettiin ruohotekstuuri, joka sai ympäristöstä selkeästi eloisamman näköisen. Myös esimerkiksi SimCityn kaltaisissa peleissä tyhjä maasto on yleensä oletukseltaan ruohomaastoa. Viimeiseksi ympäristöön tehtiin vielä yksinkertaisia puita, jotka käytännössä koostuivat ristikkäin asetelluista plane-tasoista ja kolmiulotteisesta rungosta. Sen jälkeen lopullinen kaupunkiasetelma oli käytännössä valmis (kuva 10).



Kuva 10. Lopullinen kaupunkiasetelma 3ds Max -ohjelman sisällä.

Seuraavaksi piti toteuttaa jääkausi asetelma, josta oli jo valmiiksi selkeä suunnitelma. Suunnitelman mukaisesti asetelma koostuisi lumimaisemasta, jossa havupuita ja kaksi mammuttia. Mammutit tulisi olla myös animoituja elävyyden lisäämiseksi. Tarkoituksena oli tehdä yksinkertainen animaatiosilmukka, joka toistaa samaa liikettä. Esimerkiksi toinen mammuteista söisi maaston sulaneesta kohdasta ruohoa kärsäänsä apuna käyttäen. Maiseman toteuttaminen aloitettiin maaston rakentamisesta. Maasto ei saanut olla samalla tavalla tasainen kuin kaupunkimaisemassa, sillä jääkauden maisemassa olisi luonnon muovaama ympäristö. Lumimaastossa käytettiin plane-tasoa, jolle tehtiin korkeuskartta. Korkeuskartan avulla maastoon oli mahdollista tehdä nopeammin luonnollisempia muotoja. Lumen päälle aseteltiin toinen maasto, jossa oli luminen ruohikko tekstuuri. Asetelmaan luotiin myös ruohoa sekä havupuu. Puun rungon päälle aseteltiin eri asentoihin planeja, jotta puu näyttäisi kolmiulotteiselta kaikista suunnista katsottuna. Lopuksi ympäristöjen tekstuurien värisävyjä piti tasoittaa samankaltaisiksi, koska alunperin puissa ja ruohossa oli räikeästi erilaiset sävyt. Kun kaikilla elementeillä oli samankaltainen värimaailma, ympäristö näytti heti paljon tasapainoisemmalta.

Viimeiseksi piti toteuttaa mammutit, jotka olisivat asetelman pääelementit. Hahmojen mallinnus oli paljon haasteellisempi työ, kuin jääkauden ympäristön toteutus. Mammuttien toteutukseen tarvitsi tehdä yksi mammutin 3D-malli, jota voitiin käyttää molemmissa mammuteissa. Animaatiot tulisivat kuitenkin olemaan molemmilla erilaiset. Mammutista piti tehdä yksinkertainen, mutta samalla myös elävän oloinen. Mammutti toteutettiin polygonimallinnuksella, sillä laatikosta olisi ollut vaikea muovata orgaanista hahmoa (kuva 11). Mammutin tekstuuri tehtiin käyttäen apuna karvatekstuuria joka maalattiin kloonaustyökalulla kuvankäsittelyohjelmassa. Joitakin kohtia piti piirtää myös käsin. Teksturoinnin ja mallinnuksen jälkeen mallille piti tehdä yksinkertainen riggaus, eli luuranko jonka avulla mammutti voidaan animoida. Riggauksen jälkeen piti tarkastaa, ettei tarvittavat liikkeet tehneet epämuodostumia. Epämuodostumia korjattiin Skin-työkalun avulla. Animaatiot toteutettiin key frame -animaationa. Toinen mammuteista käyttää kärsäänsä syömiseen ja toinen heiluttelee päätään katsoen ympärilleen.



Kuva 11. Mammutin 3D-malli tekovaiheessa.

5.2.4 Viimeistely

Molemmista asetelmista tehtiin myös renderöinnit sen jälkeen, kun kaikki mallit olivat valmiita. Renderöidyt kuvat tehtiin, jotta lopputuloksia pystyttiin visualisoimaan paremmin. Molemmat ympäristöt tultaisiin laittamaan itse sovellukseen vasta myöhemmin, joten niistä tarvittiin sitä ennen jonkinlaista konseptia. Koska lopullisessa sovelluksessa ympäristöt tulevat olemaan läpinäkyvän kuplan sisällä, niin renderöityyn kuvaan konseptoitiin sitä, miltä kupla tulisi mahdollisesti näyttämään. Sphere-työkalulla tehtiin pallo, jonka sisälle asetelma laitettiin. Pallolle tehtiin asetuksista lasimainen pinta, joka tulisi näkymään läpikuultavana mental ray -renderöinnin jälkeen. Asetelmaan laitettiin myös valot kolmesta eri suunnasta tuomaan valaistusta. Ympäristöön asetettiin myös renderöintiä varten taustakuva, joka heijastuisi hieman lasipinnasta. Taustakuvatekstuurina toimi aiempaa käyttöä varten CorelPaint Shop Prolla tehty avaruuskuva. Tausta toi hieman enemmän tuntua siitä, että pallo leijailisi jonkinlaisessa avaruudessa, joka

vastaisi edes hieman sitä, miltä se pelissä tulisi näyttämään. Renderöinnin ei kuitenkaan ollut tarkoitus näyttää täydelliseltä, koska kuvien tarkoitus oli vain nopeasti visualisoida lopputuloksia (kuva 12). Asetelmista otettiin myös tavallisia renderöintikuvia ilman kuplaa, jotta yksityiskohdat näkisi selkeämmin.

Kuva 12. Lopputuloksen esimerkkikuvana käytetty renderöinti jääkaudesta.

Lopuksi mallien toimivuutta testattiin tyhjässä Unity pelinäkymässä, sekä tekstuurit ja objektit nimettiin selkeästi. Opinnäytetyön käytännönosuuden lopputulokset laitettiin viimeiseksi projektin omaan Google Drive -kansioon, josta mallit voidaan myöhemmin siirtää peliin. Asetelmat tullaan laittamaan Unityyn sellaisenaan, mutta kuplat luultavasti tehdään erikseen myöhemmin. Tekstuurit laitettiin varmuuden vuoksi täysikokoisina, vaikkakin niitä luultavasti tarvitaan pelissä vain pienessä resoluutiossa. Käytännössä tämän vaiheen jälkeen opinnäytetyötä varten tehty toimeksianto oli suoritettu.

6 YHTEENVETO

Opinnäytetyössä tutustuttiin peliympäristön suunnittelun ja toteutuksen perusteisiin tutkimusten ja käytännönesimerkin avulla. Teoriaosuudessa käytiin läpi pelien 3D-grafiikan historiaa sekä nykyaikaisia käyttötapoja. Esille otettiin myös 3D-graafikon työtehtäviä ja osaamistarpeita pelialalla. Tutkimuksen pääkohtana oli tutkia peliympäristön suunnittelua ja siihen liittyviä käytännönvaiheita. Teoriaosuuden lopussa vertailtiin käytetyimpiä 3D-mallinnusohjelmia, aputyökaluja ja pelimoottoreita. Opinnäytetyön käytännönosuuden toimeksiantona oli Turun ammattikorkeakoulun Digital TimeTrek-projekti. Toimeksiantona oli luoda kaksi ympäristöä, jotka kuvasivat maapallon kehityksen vaiheita.

Tutkimusten perusteella saatiin selville, että ympäristöjen suunnitteluun vaikuttaa monia asioita sekä teknisesti ja pelillisesti. Ensimmäiseksi suunnitteluun vaikuttaa varsinkin kohdealusta. Kohdealustasta tulee ottaa huomioon sen tekniset rajoitteet ja optimointimahdollisuudet. Pelin lopullista ilmettä voidaan elävöittää renderöinnillä ja tekstuureilla. Peliä voidaan myös elävöittää kontrastin ja värien avulla. Pelaajan kuvakulma vaikuttaa paljon ympäristöjen suunnitteluun, jonka takia se tulee ottaa huomioon ensimmäiseksi ennen mallinnustyötä.

Lavasteita tehdessä on ensimmäisenä huomioitava mallinnettavan kohteen tärkeys pelimaailmassa. Yksityiskohtien määrä riippuu siitä, kuinka paljon pelaaja on sen objektin kanssa tekemisissä itse pelissä. Opinnäytetyössä otettiin esille myös eri mallinnustapoja, jotka sopivat varsinkin lavasteiden mallintamiseen. Pelimallien rakentamiseen polygonaaliset mallinnustavat ovat selkeästi suosituimpia. Yksinkertaisille malleille laatikkomallinnus on parhain tapa lähteä toteuttamaan mallia. Monimutkaisten mallien tekeminen kannattaa aloittaa mieluiten yhdestä tai useammasta polygonista.

Tutkimuksessa tuli esille, että nykyaikana on myös tarjolla monia erilaisia apuohjelmia, jotka nopeuttavat mallinnustyötä tai teksturointia. Niiden käyttö ei ole pakollista, mutta niistä on erittäin paljon hyötyä tehokkaaseen mallintamiseen. Tutkimuksessa tuli myös esille proseduraalisen mallintamisen mahdollisuudet. Peleihin tarvitaan useita lavasteita, joten niiden kaikkien tekeminen yksitellen olisi liian työlästä. Mallien generointiohjelmilla pystyy tekemään esimerkiksi maastoja, kasveja, kaupunkeja tai ihmishahmoja. Proseduraaliset työkalut ovat yhä tärkeämpi osa pelien tekemistä. Ne mahdollistavat

nopeamman ja näyttävämmän grafiikan, sillä mallintajalle jää enemmän aikaa muiden mallien hiomiseen.

Opinnäytetyön käytännönosuuden toimeksianto sujui onnistuneesti. Projektissa oli haasteena luoda malleista yksinkertaiset ja hyvän näköiset. Pelissä ne tulevat kuitenkin näkymään vain hyvin pieninä. Mallinnustyön tekemiseen oli haasteellista löytää sopivat mallinnustekniikat. Kuuluisten rakennusten mallintaminen oli vaikeaa, koska ne olivat monimutkaisen muotoisia, eikä esimerkkikuvia ollut joka suunnasta. Ihmisten aikakauden toteuttamisessa meni paljon aikaa suunnitteluun, koska lopputuloksessa näkyvät rakennukset piti harkita tarkkaan niiden muodon ja koon perusteella. Tarkoituksena oli pitää rakennusten kokoero silmämääräisesti todellisuutta vastaavana, jonka takia Big Ben oli lopulta hyvin pieni osa kokonaisuutta. Loput kuuluisista rakennuksista piti valita niiden korkeuden perusteella, siten niiden kokoero ei olisi liian suuri. Jääkausiympäristön toteuttaminen oli paljon helpompaa kuin kaupunkikuvan toteutus. Ainoana haasteellisena osana oli itse mammutin mallintaminen ja riggaus. Digital TimeTrek -projekti on vielä keskeneräinen, joten toimeksiannon lopputuloksia ei tultu näkemään pelin sisällä vielä opinnäytetyön tekovaiheessa. Projektissa on vielä useampia muitakin ympäristöjä tekemättä. Joitakin näistä ympäristöistä on tarkoitus mallintaa opinnäytetyön ulkopuolisena projektityönä. Opinnäytetyössä tutkitut aiheet tulevat auttamaan mallinnustyön jatkamista.

Peliympäristöjen toteutus on sekä taiteellista ja teknistä työtä. 3D-mallintaja, joka on vastuussa ympäristöstä tai lavasteista, tulee ymmärtää pelimoottorien toimintaa ja mallien optimointia. Pelien 3D-mallintajan tulee hallita erilaiset mallinnustekniikat, sekä osata toteuttaa tekstuureita. Erilaisista menetelmistä ja ohjelmista on hyvä hallita ainakin perusteet. Mallintamisessa on tärkeintä käyttää ohjelmaa, joka soveltuu tekijän työskentelytapoihin. Tutkimusten ja käytännönesimerkin pohjalta syntyi tutkimus, joka antaa yleiskatsauksen peliympäristöjen toteuttamisesta ja 3D-mallintajan roolista pelinkehitysprojektissa. Työ soveltuu peliympäristöjen suunnittelusta kiinnostuneille ja aloitteleville 3D-mallintajille perusteiden pohjaksi.

LÄHTEET

3D Raamattu.2012. Renderöinti. Viitattu 11.4.2016 https://3draamattu.wordpress.com/2012/07/19/renderointi/

Ahearn, L. 2008. 3D Game Environments: Create Professional 3D Game Worlds. 1st Edition. Focal Press.

Allegorithmic. 2016. Substance Painter. Substance Desinger. Viitattu 23.4.2016 https://www.allegorithmic.com/

Art of Video Games. 2016a. The History of Video Game Art. Viitattu 23.4.2016 http://www.artofvideogames.org/

Art of Video Games. 2016b. The Future of Video Game Art. Viitattu 23.4.2016 http://www.artofvideogames.org/future.php

Assaf, E. 2015. Rigging for Games: A Primer for Technical Artists Using Maya and Python. Focal Press, 41-44.

Autodesk. 2016a. 3ds Max Features. Viitattu 16.4.2016 http://www.autodesk.com/products/3ds-max/features/all

Autodesk. 2016b. Store. Viitattu 16.4.2016 http://www.autodesk.eu/store

Creative Bloq. 2015. 5 things you need to know about Autodesk's new games engine. Viitattu 23.4.2016 http://www.creativebloq.com/3d/how-build-your-own-indie-game-91516875

Creative Skillset 2016. Games Artist. Viitattu 13.4.2016 http://creativeskillset.org/job_roles/330_games_artist

CryEngine. 2016. Viitattu 23.4.2016 https://www.cryengine.com/

Daniele, T. 2008. Poly-Modeling with 3ds Max: Thinking Outside of the Box. Focal Press.

Dansereau, T. 2015. Secrets of the World Builders. 3D World. Issue 197. Future UK, 22-23.

Environment Artist 2B. 2016. Sub-D / Hard Surface Modeling. Viitattu 14.4.2016 http://environmentartist2b.com/skills/hard_surface_sub_d_modeling.html

Esri. 2016. Esri CityEngine. Viitattu 23.4.2016 http://www.esri.com/software/cityengine/industries/destroyed-city

Gooch, E. 2016. Getting a Job as a Game Artist. Viitattu 27.4.2016 http://www.cybergooch.com/tutorials/pages/gamejob/getting_a_games_art_job.htm

Jones, E. ‘Clash of Clans. Viitattu 28.4.2016 http://heavy.com/games/2014/09/clash-of-clans-tips-tricks-cheats-for-resource-buildings/

Greate 3D Games. 2015. Unity 5 vs Unreal Engine 4. Viitattu 18.4.2016 https://create3dgames.wordpress.com/2015/09/07/unity-5-vs-unreal-engine-4/

Jordan, J. 2012. Chart rush: The making of Clash of Clans. Viitattu 13.4.2016 http://www.pocketgamer.biz/feature/45814/chart-rush-making-of-clash-of-clans/

Kalderon, E. 2011. Game engines: What they are and how they work. Viitattu 13.4.2016 https://nullpwd.wordpress.com/2011/05/09/game-engines-what-they-are-and-how-they-work/

Maher, K. 2014. Evolving Characters. Viitattu 23.4.2016 http://www.cgw.com/Press-Center/In-Focus/2014/Evolving-Characters.aspx

Marshall, J. 2014. ZBrush or Mudbox: Sculpting Showdown Viitattu 23.4.2016 http://blog.digitaltutors.com/zbrush-mudbox-sculpting-showdown/

Masters, M. 2014a. What's the Difference? A Comparison of Modeling for Games and Modeling for Movies. Viitattu 13.4.2016 http://blog.digitaltutors.com/whats-the-difference-a-comparison-of-modeling-for-games-and-modeling-for-movies/

Masters, M. 2014b. 3ds Max, Maya LT or Blender - Which 3D Software Should I Choose for Asset Creation? Viitattu 16.4.2016 http://blog.digitaltutors.com/3ds-max-maya-lt-blender-3d-software-choose-asset-creation/

Masters, M. 2014c. 3ds Max vs. Maya: Is One Better than the Other? Viitattu 16.4.2016 http://blog.digitaltutors.com/3ds-max-vs-maya-is-one-better-than-the-other/

Mixamo. 2016. Fuse. Viitattu 23.4.2016 https://www.mixamo.com/fuse

Pettit, N. 2015a. Asset Workflow for Game Art: 3D Modeling. Viitattu 23.4.2016 http://blog.teamtreehouse.com/asset-workflow-game-art-3d-modeling

Pettit, N. 2015b. Asset Workflow for Game Art: Texture Mapping. Viitattu 23.4.2016 http://blog.teamtreehouse.com/asset-workflow-game-art-texture-mapping

Rabin, S. 2009. Introduction to Game Development, Second Edition. Charles River Media.

Rogers, S. 2014. Level Up! The Guide to Great Video Game Design Wiley, 19.

Rouse M. 2009. Rendering. Viitattu 13.4.2016 http://whatis.techtarget.com/definition/rendering

Russo, M. 2005. Polygonal Modeling: Basic and Advanced Techniques. Jones & Bartlett Learning.

Silverman, D. 2013. 3D Primer for Game Developers: An Overview of 3D Modeling in Games. Viitattu 11.4.2016 http://gamedevelopment.tutsplus.com/articles/3d-primer-for-gamedevelopers-an-overview-of-3d-modeling-in-games--gamedev-5704

Simplygon. 2016. Viitattu 23.4.2016 https://www.simplygon.com/games

Slick, J. 2014a. What is Rendering? Viitattu 13.4.2016 http://3d.about.com/od/3d-101-The-Basics/a/Rendering-Finalizing-The-3d-Image.htm

Slick, J. 2014b. 7 Common Modeling Techniques for Film and Games. Viitattu 14.4.2016. http://3d.about.com/od/3d-101-The-Basics/a/Introduction-To-3d-Modeling-Techniques.htm

Slick, J. 2016. List of Texturing, Surfacing, and UV Map Generation Software. Viitattu 23.4.2016 http://3d.about.com/od/A-Guide-To-3D-Software/tp/List-Of-Texturing-Surfacing-And-Uv-Map-Generation-Software.htm

Steam. 2016. MODO indie. Viitattu 16.4.2016 http://store.steampowered.com/app/321540/

TechRadar. 2010. The evolution of 3D games. Viitattu 11.4.2016 http://www.techradar.com/news/gaming/the-evolution-of-3d-games-700995/1

The Foundry. 2016a. MODO. Viitattu 16.4.2016 https://www.thefoundry.co.uk/products/modo/

The Foundry. 2016b. MARI. Viitattu 23.4.2016 https://www.thefoundry.co.uk/products/mari/

Turku Game Lab. 2016. About Game Lab. Viitattu 14.4.2016 http://www.turkugamelab.fi/?page_id=72

Unreal Engine. 2016. Viitattu 18.4.2016 https://www.unrealengine.com

Vaccaro, A. 2012. World Building. Teoksessa Hawkins, R. Vertex 1. Art by Papercut, 72-73.

Whitwam, R. 2014. Electronic Arts offers SimCity 2000 Special Edition for free. Viitattu 24.4.2016 http://www.geek.com/games/electronic-arts-offers-simcity-2000-special-edition-for-free-1611291/

Wikipedia. 2016. Pre-rendering. Viitattu 11.4.2016 https://en.wikipedia.org/wiki/Pre-rendering

World Machine. 2016. Viitattu 23.4.2016 http://www.world-machine.com/

World of Level Design. 2015. 16 Recommended 3D Game Engines Viitattu 23.4.2016 http://www.worldofleveldesign.com/categories/level_design_tutorials/recommended-game-engines.php

World of Warcraft: Warlords of Draenor. 2014. Blizzard Entertainment.

Zizka, T. 2014. 3d Modeling (21st Century Skills Innovation Library: Makers As Innovators). Cherry Lake Publishing, 6.