Tuomas Ahonen

Varjostusartefaktien välttäminen normaalikartan beikkaamisessa

Tradenomi

Tietojenkäsittely

Kevät 2021

KAMK • University of Applied Sciences

Tiivistelmä

Tekijä: Ahonen Tuomas

Työn nimi: Varjostusartefaktien välttäminen normaalikartan beikkaamisessa

Tutkintonimike: Tradenomi (AMK), tietojenkäsittely

Asiasanat: 3D-grafiikka, normaalikartan beikkaaminen

Opinnäytetyössä tutkittiin normaalikarttojen beikkaamista, jossa yksityiskohtaisesta korkearesoluutioisesta 3D-objektista siirretään tietoa matalaresoluutioiseen peliobjektiin normaalikartan muodossa. Kyseinen tekniikka on tärkeä työkalu reaaliaikaisessa peligrafiikassa, jossa prosessoitavan tiedon määrää joudutaan rajoittamaan suorituskyvyn takaamiseksi, mutta halutaan myös visuaalisesti näyttäviä 3D-assetteja.

Normaalikartta on tekstuuri, joka luo illuusion yksityiskohdista ja muodoista kappaleen pinnalla. Yleisimmin käytetyt normaalikartat ovat riippuvaisia 3D-objektin verteksien normaaleista. Niiden käyttäminen ei aina takaa siistiä lopputulosta, ja siksi on tärkeää tietää, miten työnkulun eri vaiheet vaikuttavat lopputulokseen.

Teoriaosuudessa käydään läpi tarvittava taustatieto normaalikartoista ja beikkaamisprosessista, jota sovellettiin myöhemmin työn toteutuksessa. Sitä varten mallinnettiin reaaliaikaiseen renderöintiin tarkoitettu 3D-objekti, johon kuului sekä korkea- että matalaresoluutioiset versiot. Mallintaminen tehtiin Blenderissä ja Zbrushissa. Normaalikartta beikattiin Substance Painterissä, jossa objekti myös teksturoitiin. Vastaan tulleet ongelmat ratkaistiin teoriaa hyödyntämällä ja suurimmat virheet varjostuksessa saatiin korjattua.

Lopputulosta tarkasteltiin reaaliaikaiseen renderöintiin tarkoitetussa ympäristössä. Vaikka pieniä renderöintiartefakteja jäi, niiden havaitseminen oli vaikeaa. Ainoastaan hyvin läheltä ongelmakohtia varjostuksessa pystyttiin erottamaan. Täydellisen siistiin lopputulokseen ei päästy, mutta peliobjekteissa onkin tärkeää tietää, milloin tulos on riittävän hyvä. Tehokas ajankäyttö on myös tärkeä osa artistin työtä, eikä kaikkia pieniä ongelmia ole järkevää lähteä korjaamaan.

Abstract

Author: Ahonen Tuomas

Title of the Publication: Avoiding Shading Artefacts in Normal Map Baking

Degree Title: Bachelor of Business Administration, Business Information Technology

Keywords: 3D graphics, normal map baking

This thesis explores normal map baking, where surface normal information from a high-resolution 3D model is projected onto a low-resolution game ready 3D asset and saved into a normal map texture. This technique is important in real time video game graphics where the amount of data being processed every frame must be limited to guarantee smooth performance. Baking normal maps allows optimized results while also maintaining high levels of detail.

Normal map is a 2D image which creates an illusion of details and shapes on a surface of a mesh. The most common type of normal map, a tangent space normal map, is based on the vertex normals of the object. To avoid shading artefacts, it is important to know how the different stages of the 3D asset modeling work-flow affect the outcome and the normal map.

First, the theory and required background information of normal maps and the baking process is gone through in detail. Then it is put into practice in the latter part of the thesis. Both high and low resolution meshes of a game ready 3D asset were created to bake normal maps. The modeling was done in Blender and Zbrush. The baking and texturing were done in Substance Painter.

The result was evaluated in a real time rendering environment. Even if some smaller shading artefacts re-mained, they were difficult to notice. Only when zooming really close to the surface some minor problems were visible. In game assets, perfection is not the goal and it is important to be able to tell when the result is good enough. Being efficient and not wasting time is also a part of being a 3D artist.

Sisällys

Symboliluettelo

Beikkaaminen	Prosessi, jossa kappaleen renderöintiin käytettävää tietoa tallennetaan tekstuuriin.
Peliobjekti	Matalaresoluutioinen 3D-objekti, jonka verteksien määrä on optimoitu reaaliaikaista renderöintiä varten. Pelissä käytettävä lopullinen assetti. Englanniksi low poly object.
Raskasobjekti	Korkearesoluutioinen 3D-objekti, jota käytetään tekstuureiden beikkaamiseen. Verteksien määrältä liian suuri käytettäväksi reaaliaikaisessa renderöinnissä. Englanniksi high poly object.
Renderöinti	Tietokonegrafiikassa renderöinti tarkoittaa kuvan tuottamista annetun informaation, kuten 2D-kuvien, 3D -mallien, ympäristöjen ja valaistuksen pohjalta.
Shaderi	Ohjelma, joka määrittää, mitä näytölle piirretään.
UV-kartoitus	3D-objektin pintojen levittämistä 2D-koordinaatistoon teksturointia varten. Englanniksi unwrapping.
UV-sauma	Kahden verteksin välissä oleva jana, jonka mukaan 3D-objektin pinta jaetaan pienempiin saarekkeisiin UV-kartoitusta varten.
Varjostusartefakti	Virhe kappaleen pinnan renderöinnissä, joka saa valon heijastumaan ei-toivotulla tavalla.
Varjostussauma	Kahden verteksin välissä oleva jana, joka on määritetty varjostuksen kannalta teräväksi. Englanniksi hard edge.
Verteksi	Piste 3D-avaruudessa. Verteksit määrittävät kappaleen muodon, mutta niihin voidaan varastoida muutakin tietoa, kuten UV-koordinaatit ja verteksinormaalit.
Verteksinormaali	Jokaisen verteksin sisältämä vektori, jonka suunta määrittää, miten valo käyttäytyy osuessaan kappaleen pintaan.

1 Johdanto

Peleissä ja muussa reaaliaikaisessa grafiikassa joudutaan tasapainoilemaan optimaalisen suoritusnopeuden ja näyttävän grafiikan välillä. Mitä tarkempia 3D-mallit ovat, sitä enemmän laskutehoa niiden piirtäminen näytölle vaatii. Samalla, kun laitteiston suorituskyky on kasvanut, on myös kehitetty menetelmiä, jotka mahdollistavat näyttävämmän grafiikan käyttämisen reaaliaikaisissa ympäristöissä säästäen tietokoneen laskutehoa. Tässä opinnäytetyössä käsitellään yhtä tärkeimmistä tekniikoista, joka jokaisen 3D-artistin on hallittava: normaalikarttojen beikkaamista. Sen avulla yksinkertaiset 3D-mallit saadaan näyttämään paljon yksityiskohtaisemmilta kuin ne oikeasti ovat.

Laitteiston asettamat rajoitukset 3D-grafiikan renderöimiselle on yksi pitkäaikaisimmista haasteista tietokonegrafiikassa. Reaaliaikaisessa renderöinnissä, kuten peleissä, missä käyttäjä pystyy vaikuttamaan ruudulle piirrettävään grafiikkaan, kaikki kuvan tuottamiseen tarvittavat laskutoimitukset tehdään lennossa ja tulos piirretään näytölle välittömästi. Yleinen miniminopeus kuvien tuottamiselle on 30 kuvaa sekunnissa. Siihen päästääkseen prosessoitavan tiedon määrää joudutaan rajoittamaan, mitä kutsutaan optimoinniksi. [1.]

Normaalikartat ovat mahdollistaneet matalaresoluutioisten objektien renderöimisen yksityiskohtaisina, sillä ne voivat muuttaa valon heijastumista kappaleen pinnasta luomalla illuusion monimutkaisesta geometriasta. Se miten valo käyttäytyy pintaan osuessaan, riippuu normaalikartan sisältämästä informaatiosta. Normaalikartan beikkaamisessa pinnanmuotojen ero kahden objektin välillä tallennetaan tekstuuriin. Tällä tavalla yksityiskohtaisesta objektista voidaan projisoida normaalikartta yksinkertaisempaan objektiin, mikä saa sen näyttämään samankaltaiselta. [2.]

Tekniikkana informaation siirtäminen korkearesoluutioisista objekteista matalaresoluutioisiin on ollut olemassa jo yli 20 vuotta. Vuonna 1998 IEEE Visualization -konferenssissa esiteltiin yleispätevä tapa, jolla kappaleiden normaalit voitaisiin siirtää objektista toiseen [3]. Normaalien tallentaminen tekstuuriin on vieläkin käytössä oleva tapa, jota hyödynnetään lähes jokaisessa 3D-pelissä.

Opinnäytetyön tarkoituksena on kartoittaa normaalikarttojen beikkaamiseen liittyvää työnkulkua. Normaalikarttojen käyttäminen ei takaa siistiä lopputulosta, jos mallinnusprosessin eri vaiheisiin liittyvä perustieto ei ole hallussa.

2 3D-objektien ja tekstuurien perustiedot

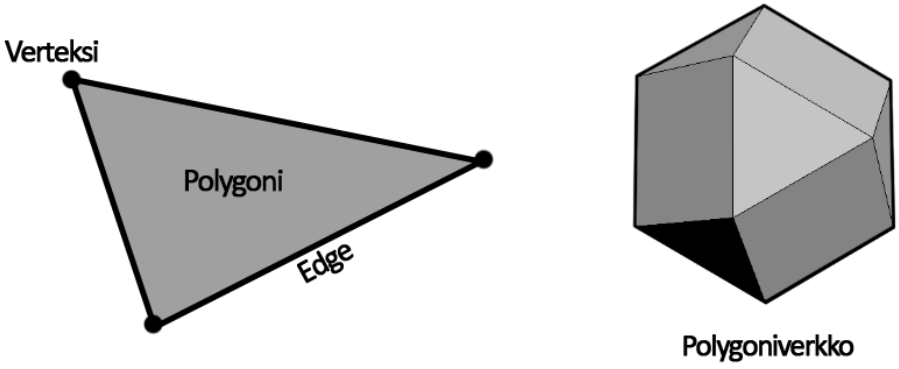
Normaalikarttojen tehokasta käyttämistä varten on tiedettävä perusasioista 3D-objekteista, tekstuureista, beikkaamisprosessista sekä siitä, miten ne vaikuttavat toisiinsa. Ilman riittävää taustatietoa mahdollisten renderöintiartefaktien korjaaminen on huomattavasti vaikeampaa.

2.1 3D-objektit

3D-mallintamisessa esineestä tai pinnasta luodaan digitaalinen representaatio, jonka muotoa voidaan muuttaa manipuloimalla sen osia 3D-avaruudessa. Muotoilun lisäksi 3D-objekteihin liittyy myös muita elementtejä, jotka vaikuttavat suoraan lopputulokseen optimaalisuuden ja visuaalisen siisteyden kannalta. Näitä ovat esimerkiksi UV-kartoitus ja varjostus.

2.1.1 Rakenne

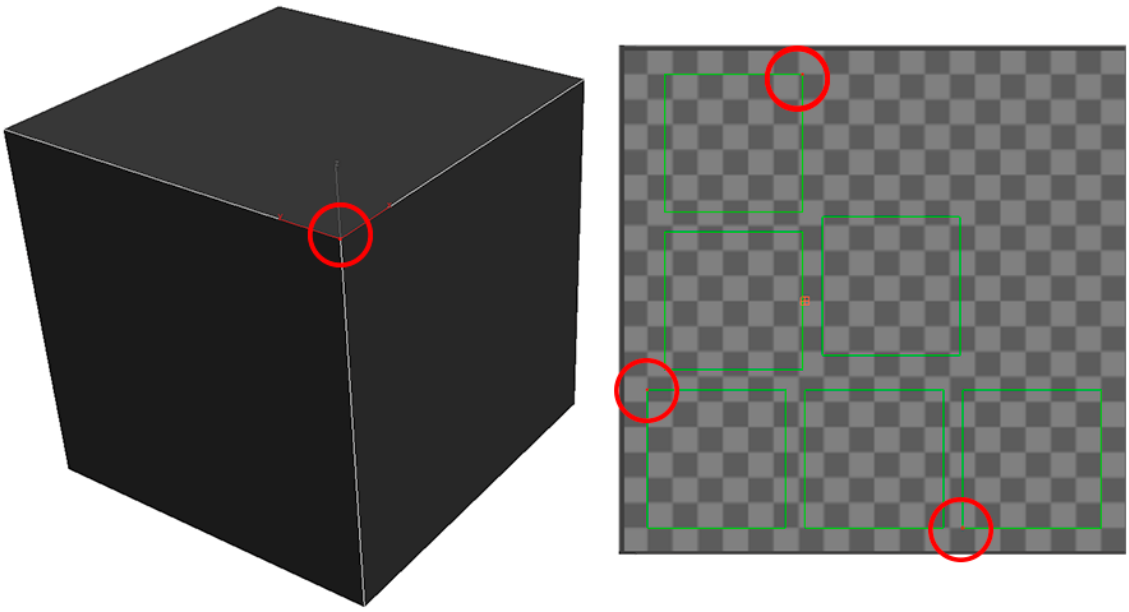
3D-objektit koostuvat vertekseistä, edgeistä ja polygoneista. Verteksit ovat pisteitä avaruudessa, jotka määrittävät kappaleen muodon. Kun kaksi verteksiä yhdistetään, niiden väliin muodostuvaa janaa kutsutaan edgeksi. Polygoni puolestaan on kolmen tai useamman toisiinsa yhdistetyn verteksin väliin muodostuva pinta. Enemmän kuin neljä verteksiä sisältävää polygonia kutsutaan N-goniksi. Näiden komponenttien muodostamaa kokonaisuutta kutsutaan polygoniverkoksi (englanniksi mesh). [4, s. 10.] Kuvassa 1 on havainnollistettu 3D-objektin eri osia.



Kuva 1. 3D -objektien rakenne.

2.1.2 UV-koordinaatit

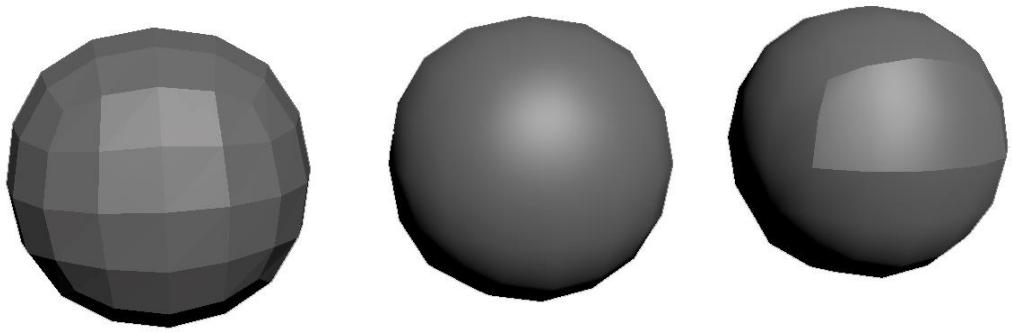
Verteksit määräävät muutakin kuin pelkän kappaleen muodon. Jokainen verteksi sisältää informaatiota, jota hyödynnetään 3D-grafiikan renderöinnissä, kuten sijainti, uv-koordinaatit ja verteksinormaalit. UV-koordinaattien määrittämistä 3D-objektille kutsutaan UV-kartoitukseksi (unwrapping). Se on prosessi, jossa kappaleen jokainen pinta levitetään 2D-koordinaatistoon teksturoimista varten. UV-kartoituksessa objektiin määritetään saumakohtia, joiden mukaan pinnat jaetaan pienempiin osiin UV-saarekkeisiin. Saumojen määrä vaikuttaa kahteen asiaan: tekstuurien vääristyneisyyteen ja lopulliseen verteksien määrään. Suurempi määrä saumoja vähentää vääristymiä tiettyyn pisteeseen asti [5], mutta jokainen saumakohdassa oleva verteksi joudutaan renderöintivaiheessa laskemaan useaan kertaan, koska se on useassa eri UV-saarekkeessa, mikä nostaa verteksien todellista lukumäärää [6]. Kuvassa 2 havainnollistetaan, miten UV-saumat vaikuttavat verteksien määrään.



Kuva 2. UV-saumat ja verteksien lukumäärä. Valittu verteksi on ympäröity punaisella värillä objektista ja UV-koordinaatistosta.

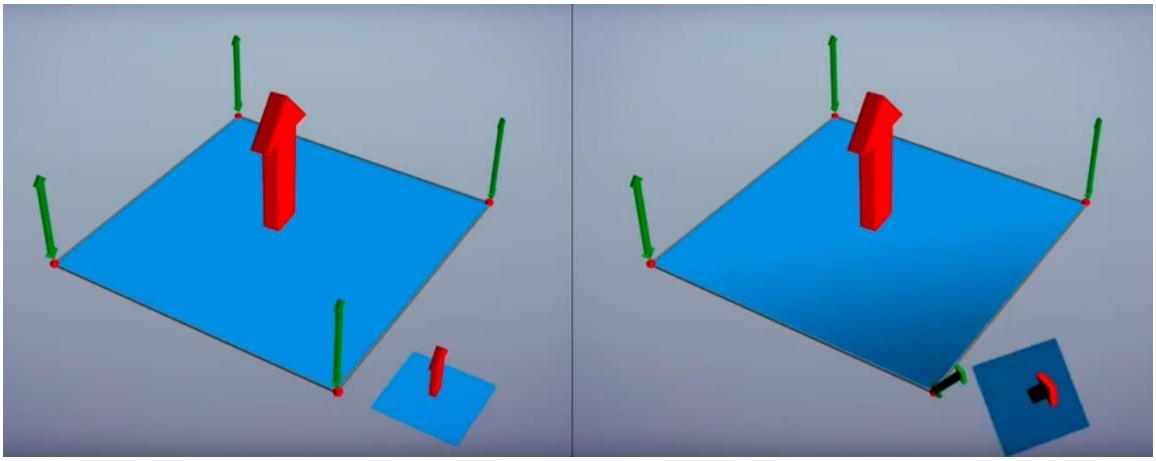
2.1.3 Varjostus ja normaalit

3D-grafiikassa varjostus tarkoittaa kappaleen pinnan väri- ja kirkkausarvojen laskemista valonlähteen valaisemana. Pelkkä kappaleen muoto ei määrää, miten valo käyttäytyy osuessaan sen pintaan vaan varjostustekniikalla on myös merkittävä osa. [7, s. 39.] Sileällä varjostustekniikalla (smooth shading) kulmikkaita ja yksinkertaisia pintoja voidaan renderöidä pehmeinä, ilman teräviä varjostussaumoja polygonien välissä, mikä on tärkeää renderöintinopeuden kannalta. Pintoihin voidaan kuitenkin manuaalisesti määrittää teräviä kohtia, mikä ei vaikuta kappaleen muotoon, vaan varjostukseen. [8.] Kuvassa 3 havainnollistetaan tätä tekniikkaa matalaresoluutioisella 3D-objektilla. 3D-mallinnusohjelmissa teräviä kohtia varjostukseen voidaan määrittää joko varjostussaumoilla (hard edge) tai polygoniryhmillä (smoothing group).



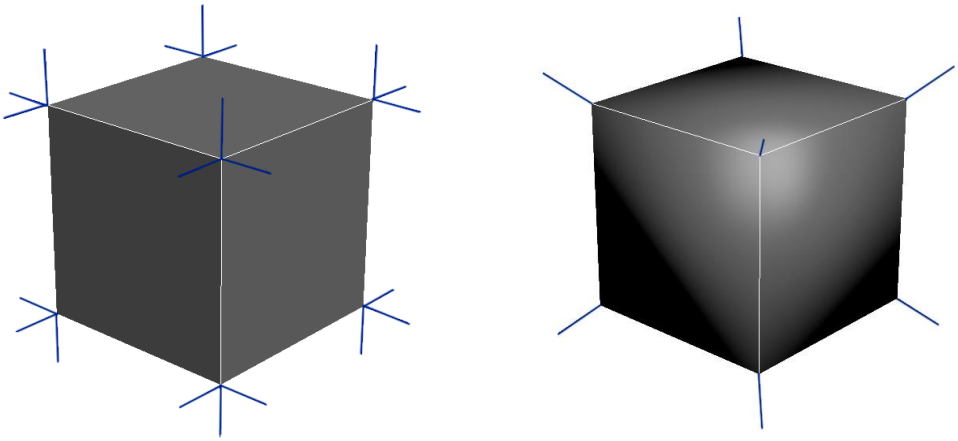
Kuva 3. Sileä varjostus. Kaikki kuvan pallot ovat muodoltaan ja geometrialtaan täysin identtisiä. Ainoa ero on varjostustekniikan avulla määritetty reunojen terävyys. Vasemmanpuoleisimmassa pallossa kaikkien polygonien reunat ovat teräviä, keskimmäisessä sileitä, ja oikeanpuoleisimmassa polygoniryhmillä on rajattu alue.

3D-grafiikassa objekteja voidaan renderöidä useilla eri tekniikoilla, jotka käyttävät erilaisia menetelmiä valaistuksen laskemiseen ja simulointiin. Yleisimmät tekniikat hyödyntävät vektoreita, joita kutsutaan verteksien normaaleiksi. Matemaattisen määritelmän mukaan normaali on pinnasta kohtisuorassa oleva vektori [9], mutta verteksien normaalien suuntaa voidaan muuttaa, jolloin ne eivät enää ole kohtisuorassa pintaan nähden. Normaalivektorin suunta vaikuttaa tulevan valon heijastumisen simulointiin kappaleen pinnasta. Kun verteksin normaali osoittaa suoraan valonlähdettä kohti, on kappaleen pinta sen kohdalta kirkkaimmillaan. [10.] Kuvasta 4 on nähtävissä, miten verteksinormaalit vaikuttavat varjostukseen.



Kuva 4. Verteksinormaalien suunnan vaikutus varjostukseen [8].

Tyypillisin tapa muuttaa objektin normaaleita on määritellä sen pintaan teräviä varjostussaumoja polygonien välisiin edgeihin. Varjostussauman kohdalla verteksi saa useita normaalivektoreita. Muussa tapauksessa verteksin normaalivektorin suunta lasketaan keskiarvona kappaleen pinnan kaarevuuden mukaan ja verteksien väliset arvot interpoloidaan, mikä tekee pinnasta sileän näköisen. [8.] Kuvassa 5 verteksinormaaleja ja varjostusta on havainnollistettu yksinkertaisella kuutionmuotoisella objektilla. Yksi viiva tarkoittaa yhtä normaalia. Kuvasta on huomattavissa, että vertekseissä, joissa useat varjostussaumat kohtaavat, on useita viivoja eli normaaleja riippuen siitä, kuinka monessa saumoilla rajatussa alueessa verteksi on mukana. Ilman niitä matalaresoluutioisten ja kulmikkaiden objektien varjostus näyttää sotkuiselta, kuten oikeanpuoleisesta kuutiosta on nähtävissä.



Kuva 5. Verteksien normaalivektorit. Vasemmanpuoleisessa kuutiossa jokainen reuna on määritetty teräväksi varjostussaumoilla, kun taas oikeanpuoleisessa kaikki reunat ovat pehmeitä.

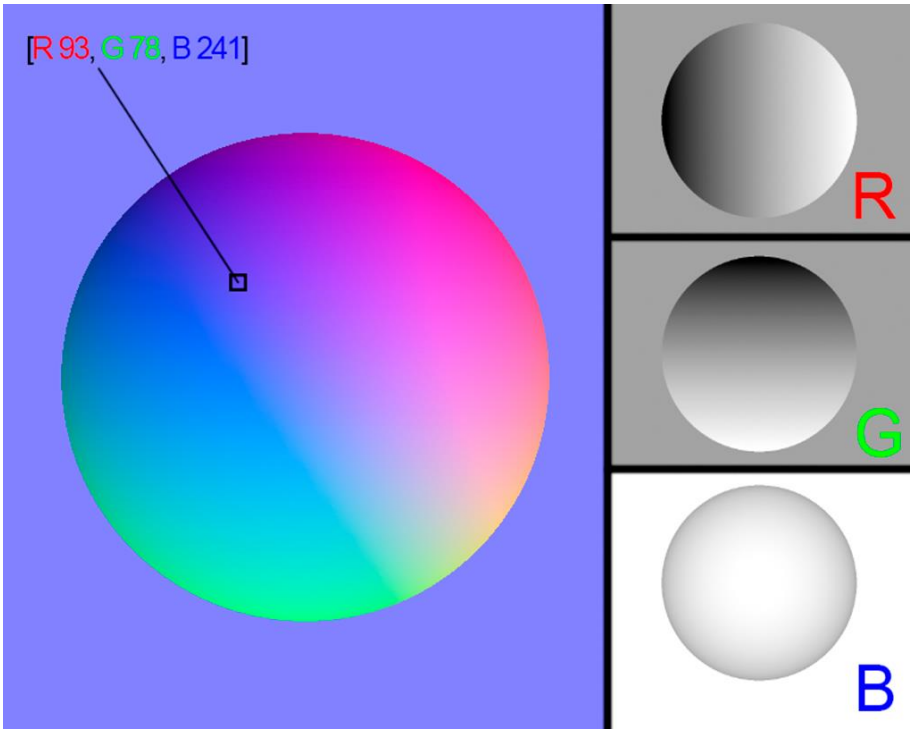
2.2 Tekstuurit

Tekstuurit ovat 2D-kuvia, jotka yhdessä shadereiden ja materiaalien kanssa määrittävät, miltä 3D-objektin pinta näyttää. UV-koordinaatit määrittävät, mihin kohtaan pintaa mikäkin osa tekstuurista tulee. Tekstuureita on useita eri tyyppejä, jotka vaikuttavat pinnan renderöintiin eri tavoilla. Tekstuurit on jaettu kolmeen eri värikanavaan: punainen, vihreä ja sininen (RGB). Joissakin tekstuureissa voi olla myös neljäs kanava, jota käytetään yleisesti määrittämään tekstuurin läpinäkyvyyttä (alpha). Shaderistä riippuen värikanaviin voidaan kuitenkin pakata mitä tahansa tietoa. Esimerkiksi neljä eri harmaasävyistä tekstuuria, jotka kaikki määrittävät materiaalin eri ominaisuuksia, kuten kiiltävyys ja metallisuus. [11.]

2.2.1 Normaalikartat

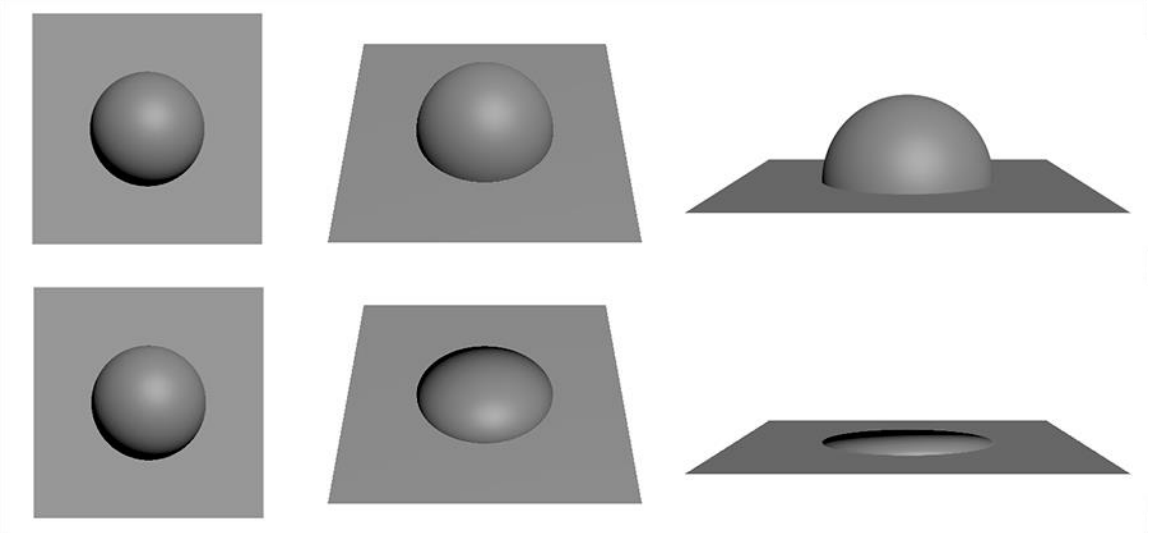
Normaalikartat ovat 3D-objektin pinnan normaalien esittämistä tekstuurimuodossa [12]. Niiden avulla voidaan luoda illuusio muodoista ja yksityiskohdista kappaleen pinnalla muuttamatta sen geometriaa. Äkkiseltään niiden aikaansaamaa efektiä voisi verrata korkeuskarttoihin (height-map), jotka sisältävät tietoa pinnan korkeuseroista. Normaalikartat eivät kuitenkaan määritä, mikä kohta pinnasta on korkeammalla kuin toinen, vaan ainoastaan, missä kulmassa pinta on vertauskohtaan nähden. Normaalikartan tyypin mukaan yksi pikseli kuvaa normaalin suuntaa verrattuna pinnan oman tangentin suuntaan (tangent space), objektin lokaaliin orientaatioon (object space) tai ympäristön orientaatioon (world space). [13.]

Vaikka normaalikartat ovat muiden tekstuurien tavoin kuvatiedostoja, niiden väri ei ole olennaista niiden sisältämään tiedon kannalta. Normaalikartoissa pikselit kuvaavat normaalivektoreita käyttämällä tekstuurin r-, g- ja b -värikanavia. Jokainen värikanava puolestaan kuvaa eri suuntaa koordinaatistossa XYZ. Punainen kanava merkitsee suuntaa X eli vasemmalta oikealle, vihreä kanava suuntaa Y eli alhaalta ylös ja sininen värikanava suuntaa Z eli syvyyttä. Esimerkiksi väristä RGB (93, 78, 241) voidaan muodostaa vektori, jossa X = 93, Y = 78 ja Z = 241. Renderöintiprosessissa nämä arvot kuitenkin normalisoidaan välille -1, 1 [14.] Kuvassa 6 on esimerkki normaalikartasta ja sen värikanavista.



Kuva 6. Tangenttiavaruusnormaalikartta ja sen värikanavat.

Kuvassa 7 havainnollistetaan, miltä kuvan 6 normaalikartta näyttäisi tasaisessa pinnassa eri kulmista tarkasteltuna. Ylärivissä on nähtävissä geometria, jonka muotoa normaalikartta esittää, ja alapuolella normaalikartta tasaisessa pinnassa. Kuvasta on nähtävissä, että suoraan ylhäältäpäin eroa geometrisen representaation ja normaalikartan välillä ei pysty huomaamaan, mutta vaakatasoa lähestyttäessä normaalikartan luoman illuusion teho vähenee.



Kuva 7. Normaalikartta ja geometrinen representaatio tarkasteltuna eri kuvakulmista. Ylärivi esittää geometriaa, jonka pohjalta normaalikartta on tehty, ja alarivi normaalikarttaa tasaisessa pinnassa.

2.2.2 Normaalikarttatyypit

Tangenttiavaruusnormaalikartat ovat yleisimpiä normaalikarttoja. Ne ovat riippuvaisia objektin verteksinormaaleista ja tarvitsevat oikeanlaisen tangenttiperustan (tangent basis) saumattoman valaistuksen laskemiseen. Normaalikartan normaalit käyttävät tangenttiavaruutta, kun taas peliympäristössä valaistukseen käytetyt valonsäteet maailma-avaruutta (world space). Valaistusta laskettaessa normaalit ja valonsäteet on muunnettava samaan avaruuteen, että jokainen pikseli tulisi valaistuksi oikein. Tähän hyödynnetään tangenttiperustaa. Sen voi laskea usealla eri tavalla, mikä voi aiheuttaa ongelmia varjostuksen kanssa. Jos normaalikartta on beikattu eri tangenttiperustaa hyödyntävällä sovelluksella kuin mitä pelimoottori käyttää, tuloksena on virheitä kappaleen varjostuksessa, mikä on nähtävissä erityisesti UV-saumojen kohdalla. [15.]

Tangenttiavaruusnormaalikartat ovat yleisesti käytännöllisempiä kuin objektiavaruusnormaalikartat, sillä ne eivät ole riippuvaisia objektin verteksien sijainnista sen keskipisteeseen nähden, vaan mukautuvat kappaleen pinnan tangenttien mukaan. Sen vuoksi samaa normaalikarttaa voidaan käyttää usealla eri objektilla [16]. Siksi ne myös toimivat paremmin kappaleissa, joiden pinnan muoto muuttuu, kuten animoiduilla hahmoilla. Optimoinnin kannalta on myös edullista, että

tangenttiavaruuskartoissa sininen värikanava pystytään generoimaan shaderien avulla, mikä vähentää tarvittavan muistin määrää. Huono puoli sen sijaan on, että niiden kanssa on vaikeampi välttää varjostukseen liittyviä virheitä. [15.]

Objektiavaruusnormaalikartat (object space normal map) eivät ole riippuvaisia kappaleen verteksinormaaleista, joten niitä käytettäessä voidaan varjostussaumojen määrittäminen jättää välistä. Sen vuoksi niiden avulla on helpompi välttää vähäisestä geometriasta ja verteksinormaaleista johtuvia varjostusvirheitä. Ne eivät kuitenkaan ole yhtä käytännöllisiä kuin tangenttiavaruusnormaalikartat, koska jokainen erimuotoinen kappale vaatii oman normaalikarttansa. [15.]

Maailma-avaruusnormaalikartat (world space normal map) ovat vielä rajoittuneempia, sillä ne ovat riippuvaisia kappaleen orientaatiosta ympäristöön nähden. Niitä käytettäessä objektien tulisi olla täysin staattisia eikä niitä voisi kopioida ja kääntää esimerkiksi kenttiä rakentaessa. Niitä kuitenkin voidaan hyödyntää teksturointivaiheessa. [13.]

2.3 Tekstuureiden beikkaaminen

Beikkaaminen on prosessi, jossa 3D-objektista siirretään tietoa toiseen objektiin tallentamalla se tekstuuriin. Erilaisia käyttökohteita sille on useita ja ne säästävät myös paljon manuaalista työtä. Normaaleiden lisäksi beikkaamista hyödynnetään usean erityyypisen tiedon tallentamiseen, kuten värien ja valaistuksen [17]. Esimerkiksi peliympäristöä optimoidessa tarvitaan matalaresoluutioisempia objekteja ja tekstuureita, kun siirrytään kauemmaksi tarkastelun kohteesta, mitä kutsutaan lodaamiseksi (level of detail). Lodien välillä kappaleen rakenne ja UV-koordinaatit muuttuvat, minkä vuoksi samat tekstuurit eivät aina sovi, mutta beikkaamalla tekstuurit voidaan siirtää vastaaviin kohtiin eri loditasojen välillä eikä niitä tarvitse teksturoida uudestaan.

Beikkaamiseen tarvitaan vähintään yksi objekti, joka on yleensä peliin vietävä matalaresoluutioinen 3D-assetti eli peliobjekti (low poly object). Joitakin tekstuureita voidaan tapauskohtaisesti beikata peliobjektista itsestään, kuten valaistus ja varjot (ambient occlusion). Yleensä kuitenkin halutaan tallentaa tietoa yhdestä tai useammasta korkearesoluutioisesta ja yksityiskohtaisesta raskasobjektista (high poly object). Peliobjekti voi myös koostua useasta eri objektista, kunhan niiden UV-koordinaateissa ei ole päällekkäisyyksiä beikkaamisvaiheessa. [18.] Beikkaamisessa peli- ja raskasobjektit asetetaan päällekkäin. Tekstuurin renderöintiä varten määritetään etäisyys, jolta lähetetään säteitä kohti raskasobjektia. Osumiskohdasta tallennetaan tietoa peliobjektin

UV-koordinaattien määrittämään kohtaan tekstuurissa. Tekstuuriin tallentuva tieto riippuu renderöintiasetuksista. [17.]

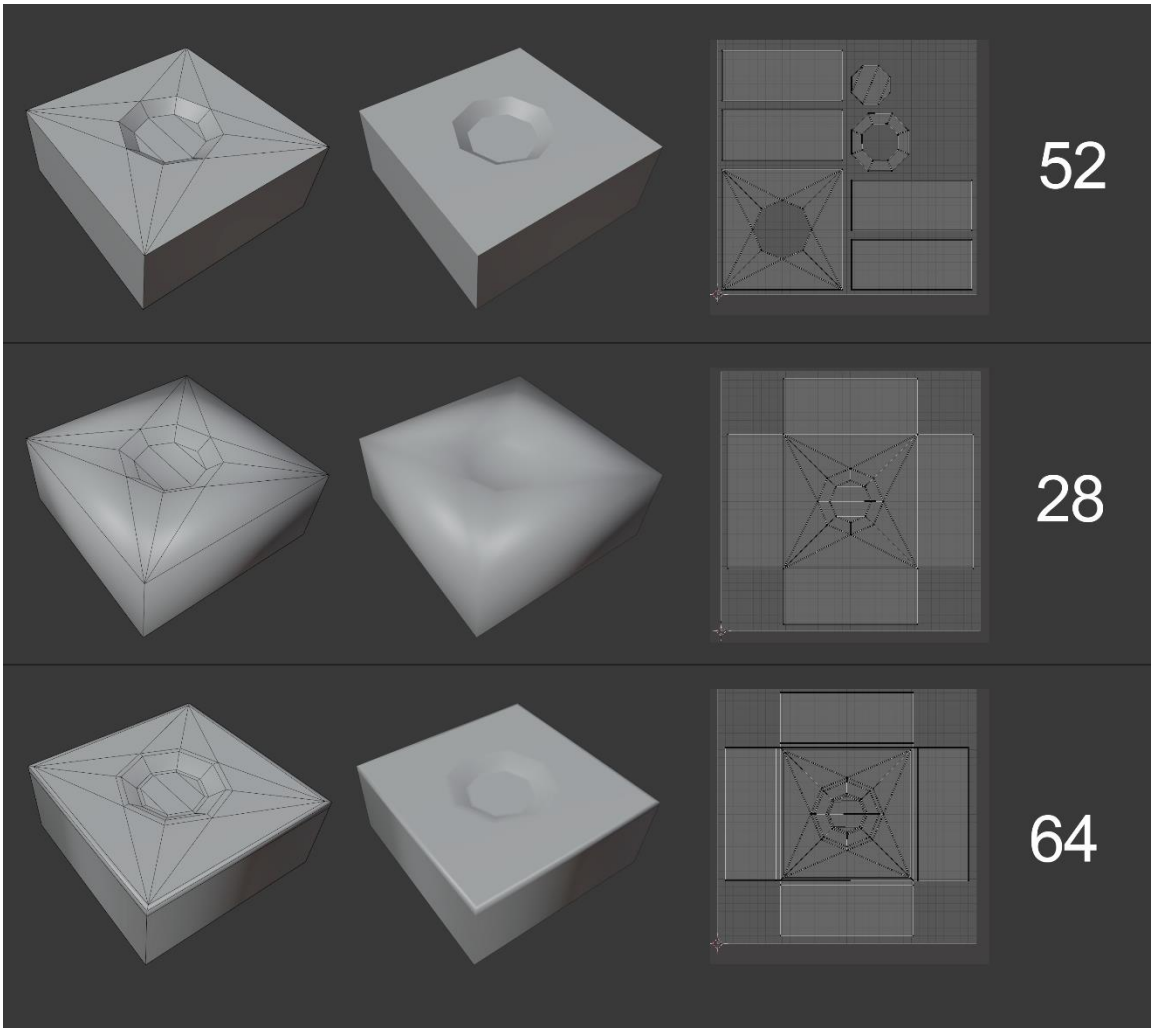
3 Varjostuksen kontrollointi

Kappaleen varjostusta voidaan hallita verteksinormaaleilla ja normaalikartoilla. Normaalikarttoja beikatessa toisesta objekteista lähestymistapoja on useita, sillä ne voivat muuttaa varjostuksen kokonaan ja tehdä lopputuloksesta siistin, vaikka ilman normaalikarttaa varjostus näyttäisi sotkuiselta. [6.]

Ensimmäinen lähestymistapa on määrittää varjostussaumat pinnanmuotojen välisen kulman perusteella tai valita silmämääräisesti kaikki kulmat, jotka näyttävät tarpeeksi teräviltä. Tällä tavalla saadaan siisti varjostus myös ilman normaalikarttaa, mutta se ei ole aina tarpeellista. Riittävä määrä varjostussaumoja on käytännöllistä esimerkiksi, jos muistin säästämiseksi tekstuurien kokoa halutaan rajoittaa eikä pikselien määrä muuten riittäisi korjaamaan varjostusartefakteja. Beikkaaminen kuitenkin vaatii, että jokaista varjostussaumaa on myös käytettävä UV-saumana, mikä lisää verteksien määrää. [6.]

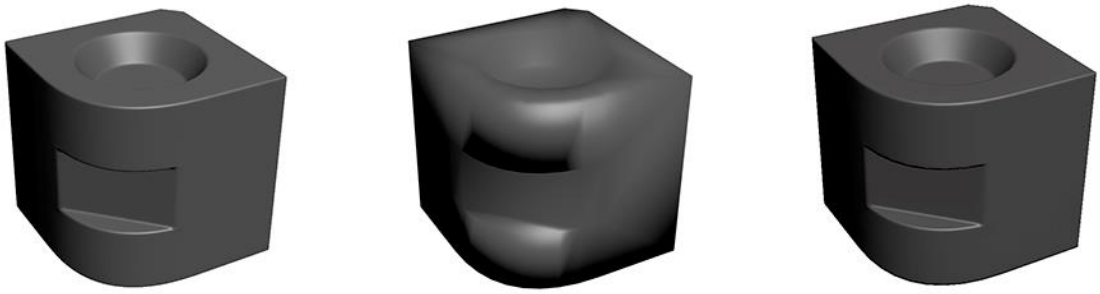
Varjostussaumat voi myös jättää kokonaan määrittämättä ja jättää varjostus sotkuiseksi. Tällöin normaalikartan on tehtävä kaikki työ sen korjaamiseksi. Laskentatehon kannalta tämä on kaikkein optimaalisinta, sillä ilman varjostussaumoja ei tarvita yhtä paljon UV-saumoja, jotka lisäisivät verteksien määrää. Ongelmia voi kuitenkin aiheutua, jos normaalikartan resoluutio ei ole riittävä korjaamaan verteksinormaaleista aiheutuvia varjostusartefakteja. Myös väärän tangenttiperustan käyttämisen aiheuttamat ongelmat korostuvat, jos varjostussaumoja käytetään vähän tai ei ollenkaan. [6.]

Kolmas keino on lisätä geometriaa reunustamaan jyrkkiä kulmia. Tämä johtaa yleensä suurimpaan määrään verteksejä, mutta vähentää tarvittavien UV-saumojen ja normaalikartalle jäävän työn määrää. Esimerkiksi hahmojen anatomiaa mallintaessa ei yleensä käytetä varjostussaumoja, vaan lisätään verteksejä jyrkempien pinnamuotojen muutosten väliin. Kuvassa 8 havainnollistetaan edellä mainittuja tekniikoita ja niiden vaikutusta verteksien määrään sekä varjostukseen. Eri tekniikat eivät ole toisiansa poissulkevia, vaan on suositeltavaa käyttää niitä eri kohdissa 3D-objektia tarpeen mukaan. [6.]



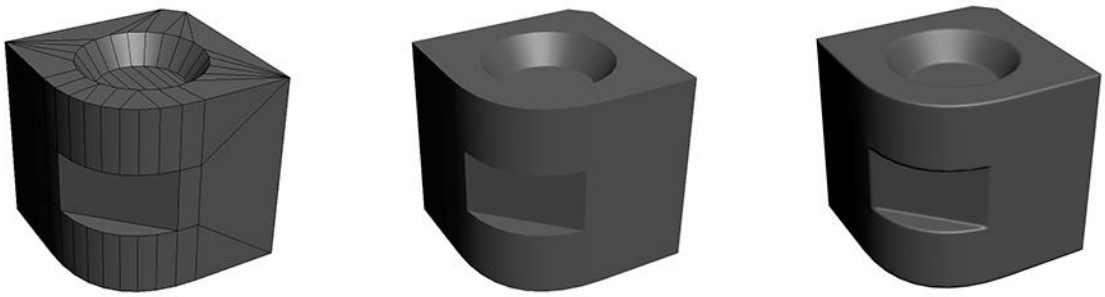
Kuva 8. Varjostuksen kontrollointi. Kaikki kuvan objektit ovat muodoltaan identtisiä. Ylärivillä varjostussaumat on määritetty kulmien terävyyden mukaan, keskimmäisellä ja alimmaisella rivillä niitä ei ole, mutta lisäksi alimmalla rivillä on lisätty ylimääräistä reunageometriaa jyrkkien pinnanmuutosten väliin. Kolmannella pystyrivillä on myös nähtävissä, miten UV-koordinaatit on määritelty kussakin tapauksessa. Oikeassa reunassa oleva lukema kuvaa verteksien määrää.

Beikkaamisen tarkoitus on saada peliobjektin pinta näyttämään samalta kuin raskasobjektin. Tangenttiavaruusnormaalikartat muodostavat yhdessä verteksinormaalien kanssa pinnan lopullisen varjostuksen, kun taas objektiavaruusnormaalikartat ohittavat verteksinormaalit täysin [15]. Kaikissa myöhemmissä esimerkeissä käsitellään tangenttiavaruusnormaalikarttoja, koska niitä käytetään eniten. Kuvissa 9, 10 ja 11 on esitelty edellä mainittuja tekniikoita normaalikarttojen kanssa. Kohteeksi on mallinnettu objekti, jossa on sekä teräviä että pyöreitä muotoja eri tilanteiden havainnollistamista varten.



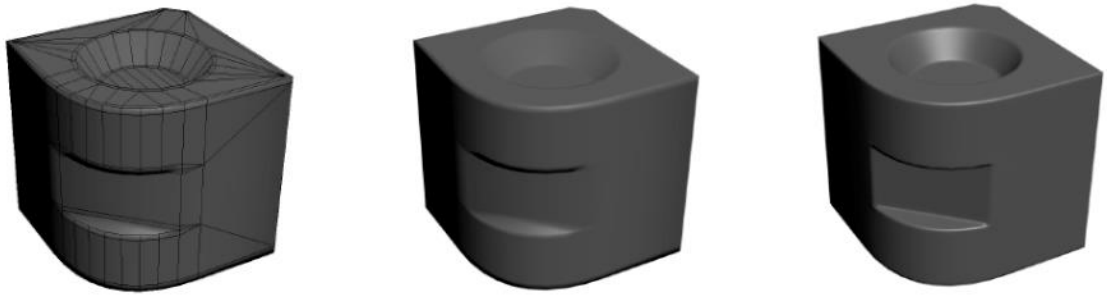
Kuva 9. Varjostus ilman varjostussaumoja. Vasemmanpuoleisin kappale on raskasobjekti, jonka pohjalta normaalikartta on renderöity. Keskellä peliobjekti ilman normaalikarttaa ja oikealla normaalikartan kanssa. Normaalikartan beikkaaminen ja tarkastelu tehtiin 3DS Maxissa.

Kuten kuvasta 9 on huomattavissa, normaalikartan avulla sotkuinen varjostus saadaan korjattua täysin. Tämä kuitenkin edellyttää, että normaalikartan beikkaamiseen käytetty sovellus käyttää samaa tangenttiperustaa kuin sovellus, jossa lopullista objektia tarkastellaan normaalikartan kanssa. Tätä kutsutaan synkronoiduksi työnkuluksi. [19.] Kuvassa 10 on käytetty samaa objektia, mutta tällä jyrkimpiin kulmiin on määritetty teräviä varjostussaumoja. Kuvasta on nähtävissä, että lopputulos on yhtä siisti kuin edellisessä tapauksessa, mutta tällä kertaa varjostus näyttää siistimmältä myös ilman normaalikarttaa.



Kuva 10. Varjostussaumojen hyödyntäminen. Vasemmassa reunassa on nähtävissä käytetyn objektin polygoniverkko, joka on myös identtinen kuvan 8 peliobjektin kanssa. Keskellä on sama objekti ilman normaalikarttaa ja oikealla normaalikartan kanssa. Normaalikartan beikkaamiseen käytettiin samaa raskasobjektia kuin kuvassa 8. Normaalikartan beikkaaminen ja tarkastelu tehtiin 3DS Maxissa.

Kuvassa 11 kaikki varjostussaumat on korvattu ylimääräisellä reunageometrialla. Kaikissa tapauksissa lopputulos on siisti, mikä johtuu oikean tangenttiperustan käyttämisestä. Koska normaalikartat on beikattu ja kuvat renderöity 3DS Maxissa, varjostusartefakteja ei ole nähtävissä.



Kuva 11. Varjostus ylimääräisen reunageometrian kanssa.

4 Työnkulkuun liittyviä ohjeita

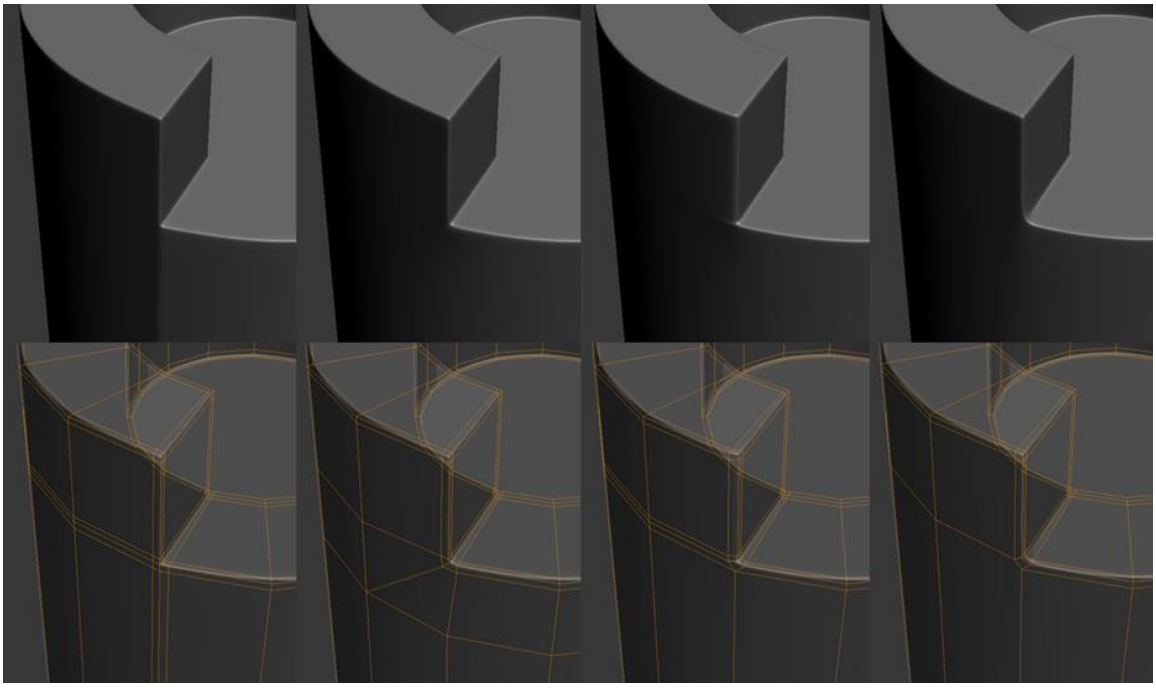
Normaalikarttojen beikkaamiseen liittyen ei ole olemassa varsinaisia kiveen hakattuja sääntöjä, joita pitää ehdottomasti noudattaa. Kuitenkin tiettyjä ohjeita noudattamalla suurimmilta varjostusvirheiltä voidaan välttyä.

4.1 Mallintaminen, topologia ja optimointi

Topologialla tarkoitetaan objektin rakennetta eli sitä, miten verteksit, edget ja polygonit yhdistyvät toisiinsa muodostaen 3D-objektin pinnan. Samanmuotoisella objektilla on lukematon määrä erilaisia tapoja, joilla sen muoto voidaan rakentaa. Hyvä topologia helpottaa objektin työstämistä mallinnusvaiheen jälkeen, takaa siistimmän lopputuloksen ja vähentää prosessoitavan tiedon määrää renderöintivaiheessa. [20.]

Raskasobjektin topologiassa verteksien määrällä ei ole väliä. Tärkeintä on, että pinta näyttää siistiltä. Korkearesoluutioisia objekteja mallintaessa käytetään yleensä erilaisia algoritmeja, joilla matalaresoluutioisen objektin pinta jaetaan pienempiin osiin, mikä pyöristää kappaleen muotoja ja lisää verteksien määrää (subdivision surface). Kulmien terävyyttä säädetään joko lisäämällä geometriaa kulmien lähelle tai säätämällä edgejen terävyyttä pinnan jakamisessa käytettävien parametrien avulla (creasing). Lopputuloksen kannalta siisti topologia on tärkeä, koska se vaikuttaa pinnan tasaisuuteen subdivision-algoritmien käytön yhteydessä. Kolmioita ja n-goneja kannattaa välttää, koska ne aiheuttavat epätasaisuuksia kaarevissa pinnoissa. [21.]

Raskasobjektia mallinnettaessa on syytä tehdä terävistä kulmista hieman pyöreämpiä kuin ne oikeasti olisivat, sillä se saa kulmat näyttämään paremmalta kauempaa katsottuna. Raskasobjekti voi koostua useasta pienemmästä palasesta tai objektista. Useamman osan käyttäminen on suositeltavaa, koska se helpottaa siistin topologian muodostamista, kun kaikkia osia ei tarvitse yhdistää toisiinsa. Pieniä yksityiskohtia, kuten syvennyksiä, paneeleita, muttereita ja ruuveja voidaan mallintaa täysin irrallisena geometriana (floating geometry), jotka projisoidaan normaalikarttaan muiden osien yhteydessä. Jos raskasobjektissa on kohtisuoria pinnanmuotoja yksityiskohtina, joita peliobjektissa ei ole, ne eivät tule näkymään normaalikartassa. [19.] Kuvassa 10 näytetään, miten topologia vaikuttaa pinnan tasaisuuteen raskasobjektissa.



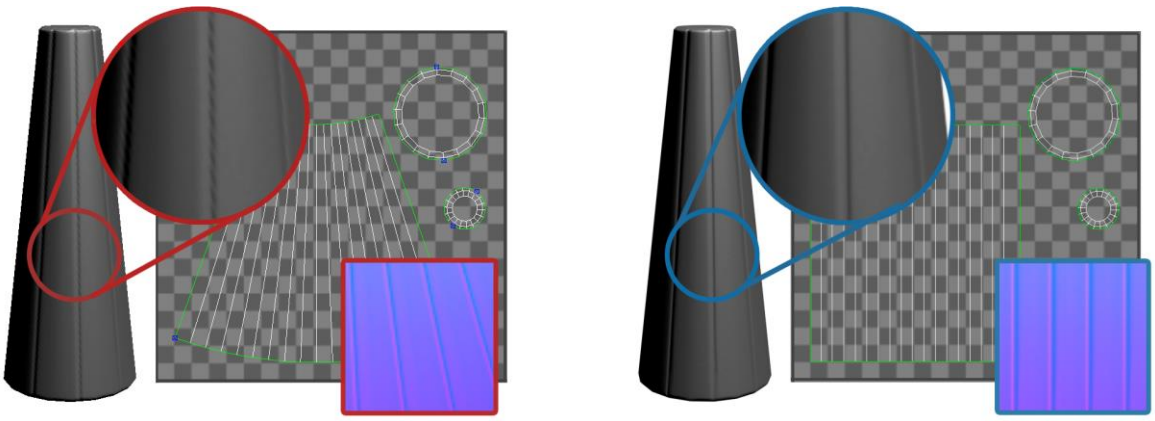
Kuva 12. Esimerkki raskasobjetkin topologian vaikutuksesta pinnan tasaisuuteen [21].

Peliobjektissa hyvällä topologialla säästetään verteksejä ja saadaan siisti deformaatio animoitavilla objekteilla. Pyöreät muodot tarvitsevat enemmän geometriaa kuin terävät kulmat ja tasaiset pinnat. [20.]

Kaksi yleistä tapaa hallita objektin varjostamista on verteksien lisääminen kohtiin, joissa on teräviä kulmia tai muita suuria muutoksia pintageometriassa, ja varjostussaumojen määrittäminen. Kummassakin keinossa on hyvät ja huonot puolensa, ja siksi niitä kannattaa käyttää eri kohdissa, riippuen siitä, mikä takaa parhaimman lopputuloksen. Varjostussaumojen kohdalla verteksit saavat useita normaalivektoreita. Käytännössä tämä lisää verteksien määrää, vaikka kappaleen pinnalla näyttäisi edelleen olevan vain yksi verteksi. 3D-mallinnusohjelmat eivät myöskään välttämättä kerro verteksien todellista määrää, sillä ne eivät ota huomioon varjostus- tai UV-saumojen vaikutusta. Objektin todellinen verteksimäärä voi olla huomattavasti suurempi riippuen siitä, miten sen varjostus- ja UV-saumat on määritelty. [6.]

4.2 Ohjeita UV-kartoitukseen

Saumoja pyritään piilottamaan kohtiin, jotka eivät ole näkyvissä, teksturointiprosessin helpottamiseksi [5]. Yleensä UV-saarekkeista pyritään poistamaan kaikki vääristyneisyys, mutta niiden suoristamisella vertikaali- ja horisontaaliakseleiden mukaisesti on myös puolensa, vaikka tämä aiheuttaisi pientä vääristymistä. Näin voidaan välttää sahalaitaisia kohtia tekstuurissa etenkin pienillä resoluutioilla. Tätä ei kuitenkaan kannata tehdä kaikille UV-saarekkeille. Jos pinta on hyvin monimutkainen ja epätasainen, ei sitä välttämättä kannata suoristaa UV-koordinaatistossa. [22.] Kuvassa 12 vasemmanpuoleisen objektin UV-koordinaateissa ei tapahdu vääristymistä, mutta UV-koordinaattien vinoittaisuus tuo esiin matalasta resoluutiosta johtuvan sahalaitaisuuden. Kummassakin tapauksessa on käytetty normaalikarttaa, jonka resoluutio on 256x256. Oikeanpuoleisessa objektissa UV-koordinaatit on suoristettu, mikä aiheuttaa pientä vääristymistä, mutta poistaa tekstuurin sahalaitaisuuden.



Kuva 13. UV-saarekkeiden suoristaminen. Kappaleet ovat muuten täysin identtisiä paitsi UV-koordinaattien suhteen.

Optimoinnin kannalta ja varjostusartefaktien välttämiseksi jokainen varjostussaumojen erottama pinta on suositeltavaa jakaa omaan UV-saarekkeeseensa. Toisin sanoen varjostussaumoja kannattaa käyttää myös UV-saumoina parhaan lopputuloksen saavuttamiseksi, kun tarkoituksena on renderöidä normaalikartta raskasobjektista. Vaikka varjostus- ja UV-saumat lisäävät verteksien todellista määrää, ne eivät kuitenkaan kasaudu siten, että samassa kohtaa olevat saumat lisäisivät kumpikin määrää yhdellä, vaan niiden määrittäminen samaan kohtaan on jopa kannattavaa, koska se ei maksa enempää renderöintivaiheessa. Tämä ei kuitenkaan tarkoita, että jokainen UV-sauma pitäisi myös määrittää varjostussaumaksi. Esimerkiksi pallon tai minkä tahansa muun pyöreän kappaleen UV-kartoituksessa tarvitaan saumoja kohdissa, joissa pinnassa ei ole tarkoitus olla saumoja varjostuksessa. [6.]

Normaalikartan renderöintiä varten on myös varmistettava, että UV-saarekkeissa tai yksittäisissä polygoneissa ei ole päällekkäisyyttä UV-koordinaatistossa. Sen lisäksi jokaisen saarekkeen ja UV-koordinaatiston reunojen välillä tulee olla riittävästi tilaa, ettei tekstuuri vuoda väärään osaan 3D-objektin pintaa. [23.]

4.3 Symmetria

Tekstuurit vievät tietokoneen muistia, minkä vuoksi on yleistä hyödyntää symmetriaa peliobjekteissa. Koska symmetrisen kappaleen molemmat puolet ovat identtisiä ja hyödyntävät samaa kohtaa tekstuurista, vie se huomattavasti vähemmän tilaa. Normaalikarttojen beikkaamisen yhteydessä symmetrian käyttöön liittyy ongelmia, jotka voivat aiheuttaa virheen varjostuksessa symmetria-akselin kohdalla. Lopputulos voi näyttää samalta kuin akselin kohdalla olisi varjostussauma, vaikka pinnan kuuluisi olla varjostuksen kannalta sileä. Koska symmetria-akseli on keskellä objektia, on virhe yleensä myös kaikkein näkyvimmällä kohdalla. [23.]

Yleinen tapa beikata symmetrisiä objekteja on siirtää symmetrian toinen puoli yhden yksikön verran UV-koordinaatiston ulkopuolelle. Koska tekstuurit ja UV-koordinaatit toistuvat loputtomiin, symmetrinen puoli saa tismalleen saman alueen tekstuurista, mutta ei aiheuta ongelmia beikkaamisessa. Jos symmetrisen puolen poistaa beikkaamista varten, symmetria-akselin reunalla olevat verteksinormaalit eivät täsmää, koska toisen puolen verteksit eivät enää vaikuta niiden suuntaan, mikä tekee varjostuksesta epäyhtenäisen. [23.]

Vaikka edellä mainittuja ohjeita noudattaisi, ei symmetria-akselin kohdalla oleva sauma välttämättä häviä kokonaan. Sauman voi siirtää kohtaan, jossa sitä ei erota tai sen voi piilottaa paremmin esimerkiksi teksturoimalla. Tässä tapauksessa kappaleen keskeltä, jossa symmetria-akseli normaalisti menisi, voidaan erottaa alue, joka UV-kartoitetaan normaalisti ilman symmetriaa. [23.]

4.4 Beikkaaminen

Beikkaamiseen liittyy monta vaihetta. Ensin suoritetaan tarvittavat valmistelut, jotka vaihtelevat hieman riippuen käytettävästä ohjelmasta. Valmisteluihin kuuluu esimerkiksi objektien hajauttaminen, jos niissä on monta lähellä toistaa olevaa osaa, ja joissain tapauksissa projisointiobjektin

eli häkin määrittäminen. Kun normaalikarttaa siirrytään beikkaamaan, suoritetaan usein ensin testejä, joiden perusteella mahdolliset ongelmakohdat voidaan paikallistaa. Ongelmat voivat liittyä moneen eri asiaan, kuten projektioetäisyyteen, UV-koordinaatteihin tai verteksinormaaleihin. Kun ongelmat on korjattu, voidaan normaalikartta renderöidä tarkemmilla asetuksilla teksturointia varten.

4.4.1 Valmistelut

Normaalikartan renderöintiä varten peliobjekti on asetettava samaan kohtaan kuin raskasobjekti. Jos objektit eivät ole täysin yhtenäisiä ja koostuvat useammasta toisensa lävistävästä osasta, on olemassa muutamia eri lähestymistapoja. Yksi keino on tehdä peliobjektista yhtenäinen, vaikka raskasobjekti koostuisikin useasta palasesta. Toinen keino on hajauttaa kummatkin objektit siirtämällä irtonaiset osat kauemmaksi toisistaan, ettei niissä ole päällekkäisyyttä. Kolmas keino on määrittää, mitkä osat vastaavat toisiaan peli- ja raskasobjektien välillä. [17.] Esimerkiksi Substance Painterissä objektit voidaan erotella pienempiin osiin ja nimetä siten, että vastaavilla osilla on muuten sama nimi, mutta eri takaliite. Tällöin kappaleiden eri osat voivat olla aivan kiinni toisissaan tai läpäistä toisensa, koska beikatessa ainoastaan samannimiset pinnat huomioidaan. [18.] Kuvassa 14 on esimerkki objektin hajauttamisesta beikkaamista varten.

Kuva 14. Objektin hajauttaminen beikkaamista varten. Kuvassa on nähtävissä pelkkä raskasobjekti selkeyden vuoksi. Peliobjekti tulisi hajauttaa samalla tavalla, siten että vastaavat osat niiden välillä olisivat päällekkäin.

Ennen renderöintiä on myös syytä kolmioida objekti parhaan lopputuloksen saavuttamiseksi, sillä kolmiointi saattaa muuttua riippuen siitä, suoritetaanko se vasta pelimoottoriin vietäessä vai beikkaamissovelluksessa ja aiheuttaa ei-toivottuja artefakteja [19]. Ennen beikkaamista on myös varmistettava, ettei objektilla ole negatiivista skaalaa, joka voi aiheutua esimerkiksi peilaamisesta

symmetristä objektia mallinnettaessa. Myös rotaatio voi aiheuttaa ongelmia normaalikartan renderöinnissä.

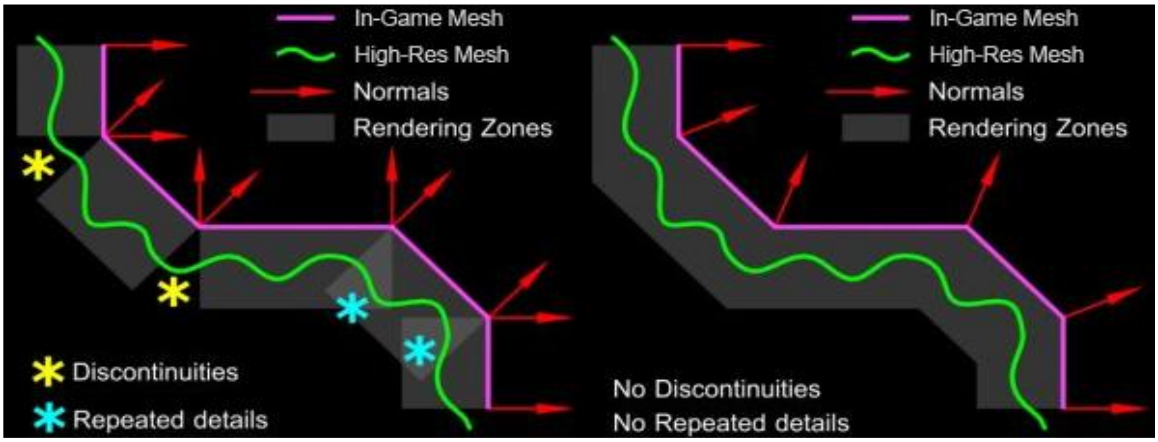
Eri mallinnusohjelmat ja pelimoottorit voivat käyttää eri tangenttiperustaa tangenttien laskemiseen. Ennen normaalikartan renderöintiä on tiedettävä, mitä metodia beikkaamisen suorittava ohjelmisto ja kohdepelimoottori käyttävät. Jos normaalikartan renderöinti suoritetaan eri tangenttiperustalla kuin mitä pelimoottori käyttää, on lopputuloksena virheitä kappaleen varjostuksessa. Yleisintä menetelmää tangenttien laskemiseen kutsutaan nimellä MikkTspace, ja sitä käyttävät esimerkiksi Blender, Unreal Engine, Unity ja Substance-ohjelmat. [19.]

Yleensä mallinnusohjelmat sisältävät tarvittavat työkalut normaalikarttojen renderöimiseen. Jos ohjelmassa ei kuitenkaan ole sama tangenttiperustan laskutapa kuin kohdepelimoottorissa, on syytä käyttää erillistä sovellusta. Beikkaamisen voi suorittaa esimerkiksi kolmannen osapuolen sovelluksissa, kuten Xnormal, Handplane ja Substance Painter. Valmistelut voidaan suorittaa mallinnussovelluksessa, josta tarvittavat objektit viedään beikkaamissovellukseen normaalikartan renderöintiä varten.

4.4.2 Projisointi

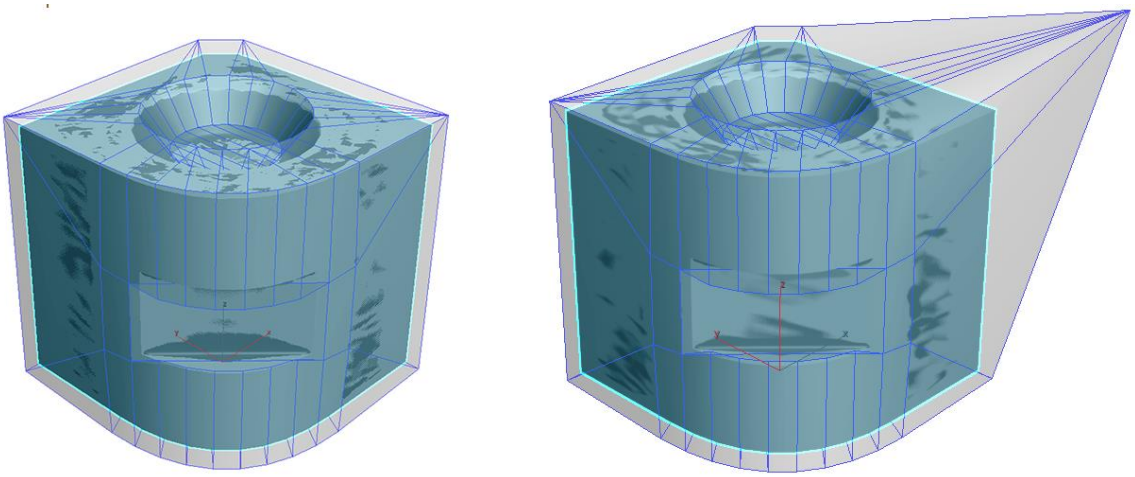
Pinnanmuotojen eroavaisuus peli- ja raskasobjektin välillä tallennetaan normaalikarttaan lähettämällä säteitä määritetyltä etäisyydeltä. Säteen osuessa raskasobjektin pintaan osumiskohdasta lasketaan pinnan normaalivektori ja tieto tallennetaan normaalikarttaan [17], jossa väriarvot r, g ja b kuvaavat osumiskohdan normaalivektoria. Säteiden lähtemiskohdan on oltava tarpeeksi kaukana. Joissakin kohtaa raskasobjektin pinta voi olla huomattavasti kauempana peliobjektista. Jos säde lähetetään tässä tapauksessa liian läheltä, se voi mennä kokonaan ohi raskasobjektin pinnasta, jolloin normaalikarttaan tulee virhe.

Projisoinnissa on tärkeää käyttää normaalien keskiarvoa (averaged normals) tai häkkiä (cage) säteiden lähettämiskohdan määrittämiseen. Jos verteksinormaaleja käytetään täsmällisesti (explicit normals), jokaisen varjostussauman kohdalle jää aukko, koska niiden kohdalla verteksinormaalit osoittavat eri suuntiin, ja lopputulos on virheellinen. [24.] Kuvan 15 vasemmalla puolella on nähtävissä varjostussaumojen aiheuttamat ongelmat, jos projektioetäisyyden määrittämiseen käytetty tekniikka on väärä. Oikeassa reunassa käytetään normaalien keskiarvoa, mikä antaa paremman lopputuloksen.



Kuva 15. Projektioetäisyyden määrittäminen normaalien mukaan. Vasemmalla täsmällinen ja oikealla keskiarvo. [17.]

Säteiden lähettämiskohdan määrittämiseen on kaksi keinoa: etäisyyden määrittäminen numeraalisella arvolla, jolloin säteiden lähetyskohta siirtyy normaalien suuntaisesti annetun arvon verran ulospäin, tai häkin asettaminen, jolla projisointietäisyyden voi määrittää manuaalisesti eri kohtiin kappaleen pintaa [17]. Kuvassa 16 on aiemmassa esimerkissä käytetyn kappaleen normaalikartan renderöintiin käytetty skenaario. Sininen polygoniverkko kuvastaa beikkaamiseen käytettyä häkkiä.

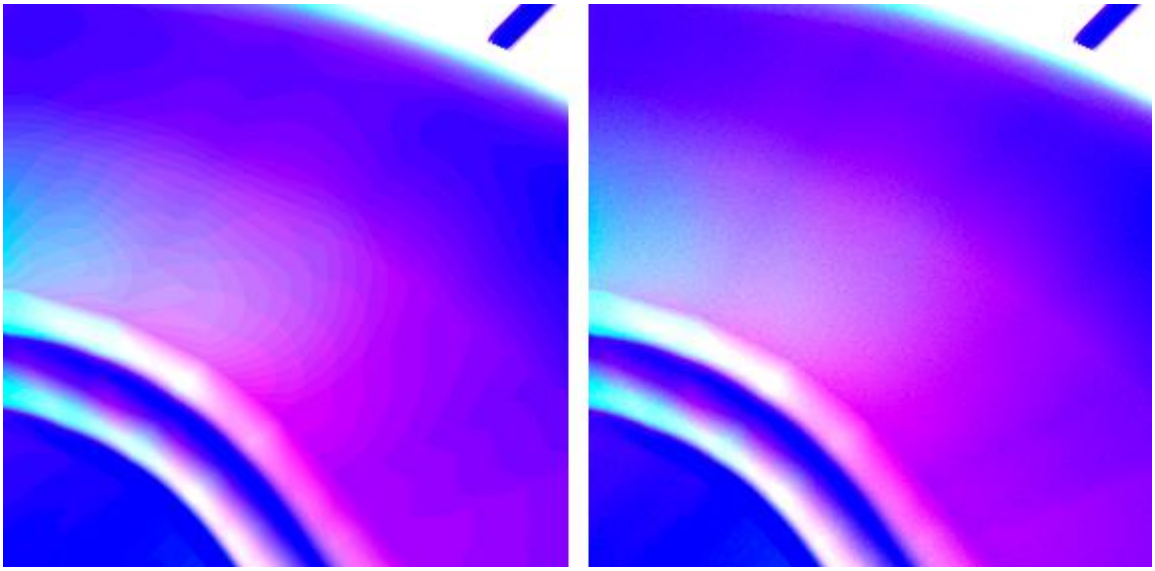


Kuva 16. Häkki. Harmaan ja valkoisen värin vaihtelut häkin sisällä johtuvat peli- ja raskasobjektien päällekkäisyydestä. Oikeanpuoleisessa reunassa häkin yhtä kulmaverteksiä on vedetty ulospäin havainnollistamaan, että sen geometriaa voidaan muuttaa manuaalisesti verteksi kerrallaan.

Jos peliobjekti ei peitä koko raskasobjektia, saattaa projektiokohdan joutua määrittämään manuaalisesti häkin avulla, sillä säteen lähtökohta voi paikoittain liikkua liian kauaksi ja mennä raskasobjektin sisään, jos pinnan muodossa on radikaaleja muutoksia, kuten reikiä tai syvennyksiä. Häkin avulla projektioetäisyys voidaan määrittää verteksikohtaisesti. Häkki on topologisesti täysin identtinen peliobjektin kanssa, mutta sen pintaa on siirretty ulospäin beikkaamista varten, kunnes se peittää alleen koko raskasobjektin [17]. Sen avulla on myös helpompi visualisoida ja paikallistaa ongelmat säteiden lähtökohdan määrittämisessä. Paras lähestymistapa on kuitenkin tehdä suurin osa työstä jo peliobjektissa, koska silloin häkkiä ei tarvitse asetella joka kerta uudestaan, jos kappaleen geometriaan on tehtävä muutoksia ja beikkaaminen suoritettava useaan kertaan. Jos peliobjekti on toteutettu hyvin, selvitään usein projisoinnista pelkästään numeraalisen arvon avulla [24].

4.4.3 Renderöintiasetukset

Normaalikarttoja renderöitäessä on suositeltavaa käyttää reunanpehmennystä sahalaitojen välttämiseksi. Saman voi käytännössä saavuttaa renderöimällä tekstuurin isommalla resoluutiolla ja sitten pienentämällä sitä kuvankäsittelyohjelmassa [17]. Toinen normaalikartan laatuun vaikuttava seikka on tekstuurin bittien määrä. Yleensä tekstuurit ovat 24-bittisiä, mikä tarkoittaa 8 bittiä tai 256 arvoa per värikanava. Tämä voi aiheuttaa ongelmia, sillä tangenttiavaruusnormaalikartoissa ei yleensä hyödynnetä koko käytössä olevaa spektriä, vaan väriarvot liikkuvat pienemmällä alueella. Seurauksena on portaittaiset sävyjen vaihtelut normaalikartassa, mikä johtaa samanlaiseen lopputulokseen objektin renderöinnissä pelimoottorissa. Suuremmilla bittimäärillä näitä ongelmia ei ole, mutta ne vievät enemmän muistia. 16 bittiä per värikanava tarkoittaa 65536 eri arvoa, mikä lisää käytettävissä olevien sävyjen määrää merkittävästi. Pelimoottoreissa näitä ei kuitenkaan yleensä käytetä muistin säästämiseksi. Portaittaisesta arvojen vaihtelusta tekstuurissa voi kuitenkin päästä eroon hyödyntämällä dithering-tekniikkaa, mikä antaa paremman lopputuloksen pienemmälläkin bittimäärällä. Kuvassa 17 näytetään, miten dithering-tekniikka vaikuttaa normaalikarttaan. [25.] Yleensä sovellukset käyttävät tekniikkaa automaattisesti eikä sitä tarvitse erikseen valita.



Kuva 17. Dithering 8-bittisellä normaalikartalla. Vasemmassa reunassa ilman ja oikeassa dithering-tekniikan kanssa. [25.]

Normaalikarttojen värikanavat kuvaavat eri suuntia XYZ-koordinaatistossa. Eri shaderit saattavat käyttää käännettyjä arvoja joillakin kanavilla. Yleisin eroavaisuus on Y-akselin eli vihreän värikanavan suunta. Jos normaalikartan on renderöinyt väärällä asetuksella y-koordinaatin suhteen, voi ongelman korjata yksinkertaisesti invertoimalla tekstuurin vihreän värikanavan. [15.]

4.4.4 Vienti- ja tuontiasetukset

Kun objekti viedään joko pelimoottoriin tai toiseen sovellukseen normaalikartan renderöintiä varten, täytyy tietää kohdesovelluksen tangenttiperusta. Ideaalissa tilanteessa sekä normaalikartan renderöintiin että lopullisen teksturoidun objektin tarkasteluun käytetyn sovelluksen tangenttiperusta on sama. Silloin saadaan paras mahdollinen lopputulos. Mallintamiseen käytetyn sovelluksen tangenttiperusta voi poiketa beikkaamis- ja kohdepelimoottorista ilman, että se aiheuttaa ongelmia. Jos objektia ei ole jo kolmioitu, voi sen suorittaa myös vientivaiheessa. Olennaista on käyttää beikkaamisessa samaa kolmiointia kuin pelimoottorissa. Jos kyseessä on synkronoitu työnkulku, tangentteja ei tarvitse sisällyttää tiedostoon, koska kohdesovellus osaa laskea ne käyttäen oikeaa menetelmää. Jos tangenttiperusta on erilainen, eikä beikkaamissovellukseen ole saatavissa lisäosaa, jolla normaalikartan renderöinnin voisi suorittaa samoilla asetuksilla, on tangentit sisällytettävä tiedostoon. [19.] Tämä ei kuitenkaan välttämättä takaa yhtä hyvää lopputulosta,

jolloin mallintamisvaiheessa verteksinormaaleja joudutaan siistimään joko lisäämällä varjostus-saumoja tai ylimääräisellä geometrialla.

5 Projektityö

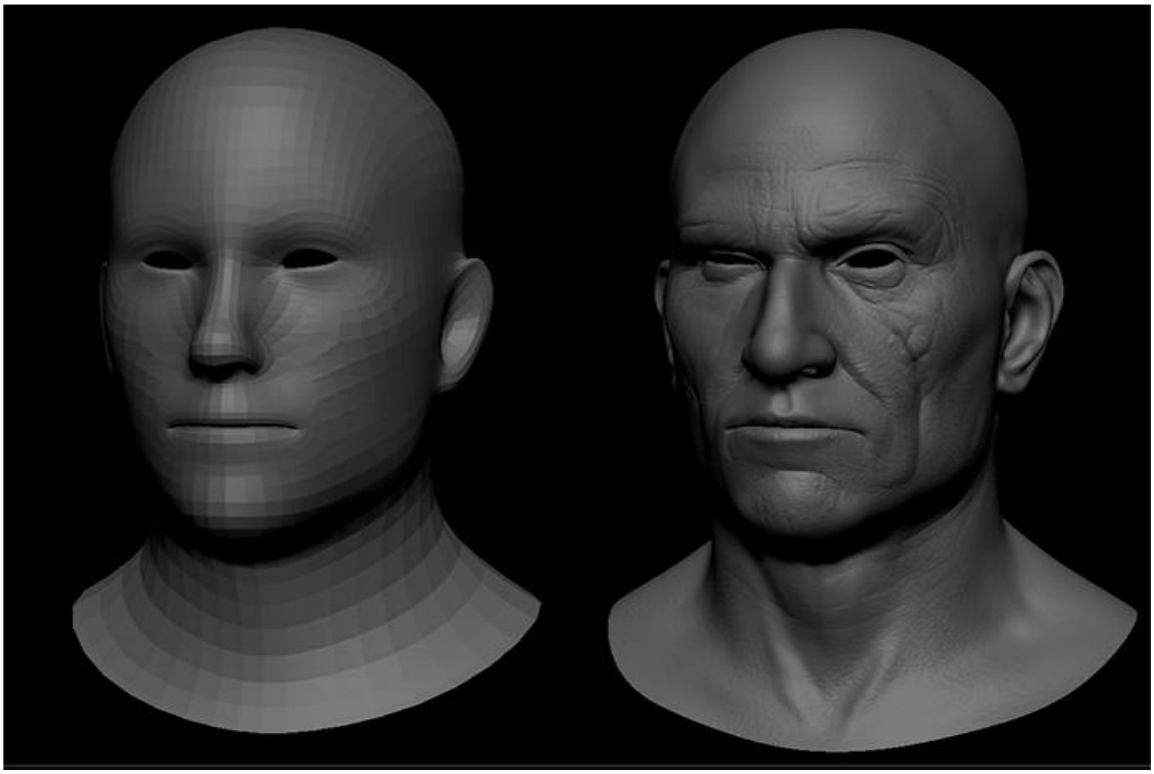
Projektityön tarkoituksena oli mallintaa reaaliaikaiseen renderöintiin tarkoitettu 3D-objekti ja beikata siihen normaalikartta korkearesoluutioisesta objektista. Tällöin joudutaan miettimään pelinkehityksessä vastaan tulevia optimointiin liittyviä haasteita, kuten verteksien määrän kontrollointia ja tekstuurien tehokasta käyttämistä. Samalla vastaan tulisi useita normaalikartan beikkaamisen liittyviä ongelmia, joissa teoriaosiossa käytyjä tekniikoita päästäisiin soveltamaan ja testaamaan, kuinka siistiä jälkeä niitä noudattamalla saataisiin aikaan. Kohteeksi valittiin ihmishahmo, joka rajattiin vain hahmon yläosaan, eli mukaan tulisi pää, vähän torsoa ja siihen liittyviä varusteita. Koko hahmoa ei mallinnettu ajansäästämisen vuoksi, eikä se myöskään ole tarpeellista opinnäytetyön aiheen kannalta. Lopuksi objekti teksturoitiin. Projektivaiheen dokumentoinnissa on myös pysytty aihealueen sisällä eli työvaiheissa, jotka vaikuttavat lopputuloksen siisteyteen varjostuksen kannalta.

5.1 Toteutus

Työn toteutuksessa käytettiin mallintamiseen Blenderia ja Zbrushia. Beikkaamiseen ja teksturointiin Substance Painteriä. Lopputulosta tarkasteltiin reaaliaikaisen renderöintiin tarkoitetussa ympäristössä.

5.1.1 Raskasobjekti

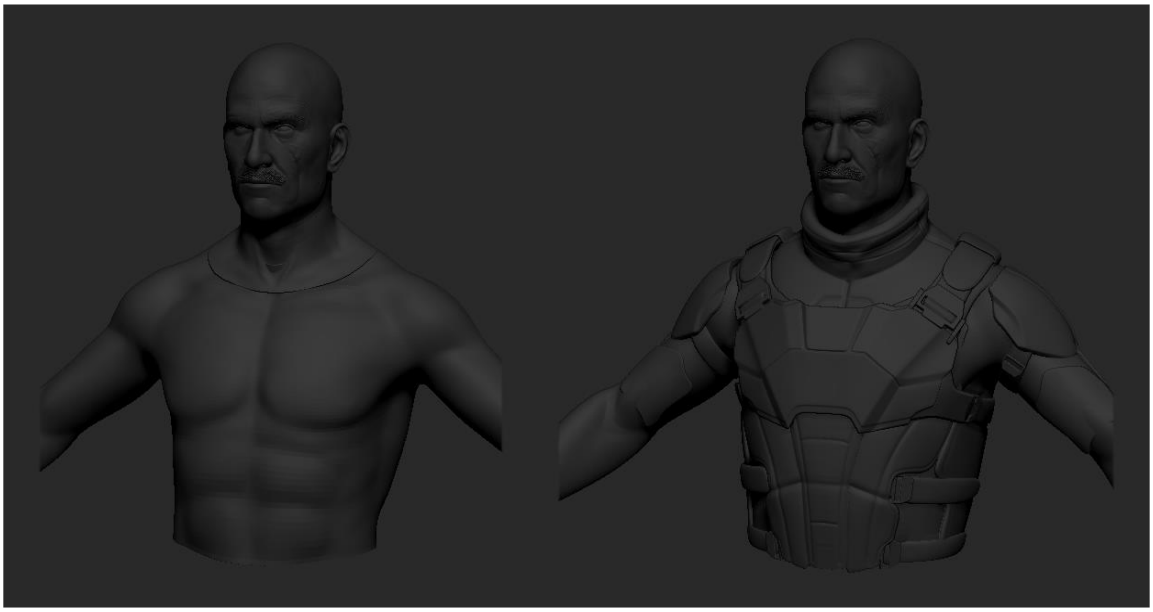
Prosessi alkoi keskittymällä raskasobjektiin. Koska työn tarkoitus oli testata normaalikarttojen beikkaamiseen liittyviä tekniikoita, hahmon visuaaliseen suunnitteluun ei käytetty turhaa aikaa muutaman referenssikuvan keräämistä lukuun ottamatta. Alussa hyödynnettiin yksinkertaista valmiiksi tehtyä 3D-objektia ihmishahmosta pohjana, että myöhemmässä vaiheessa säästyttäisiin topologian siistimiseltä peliobjektia tehdessä. Pää oli erillinen kokonaisuutensa, joka tehtiin ensimmäisenä. Lähtökohtana käytetyn mallin topologia oli valmiiksi suunniteltu vastaamaan pään muotoja ja toimimaan myös animoiduissa objekteissa. Myös verteksien määrä oli sopiva reaaliaikaiseen renderöintiin. Tavoitteena oli tehdä päästä realistinen ja sitä varten apuna käytettiin referenssikuvia. Pään muotoilu tehtiin skulptaamalla Zbrushissa. Lähtökohta ja lopputulos pään mallintamisessa on nähtävissä kuvassa 18.



Kuva 18. Pään toteutus. Vasemmalla lähtökohta ja oikealla viimeistelty korkearesoluutioinen malli.

Päästä tehtiin tarkoituksella epäsymmetrinen ja lisättiin ilmeikkyyttä, koska hahmoa ei ollut tarkoitus animoida myöhemmin.

Loput hahmosta toteutettiin käyttäen apuna toista pohjaobjektia kokonaisesta ihmisestä, mutta se rajattiin pelkkään torsoon. Suunnittelua tehtiin mallintamisen yhteydessä. Pohjamallista rajattiin osia Zbrushissa, joista muotoiltiin erilaisia pienempiä objekteja hakien samalla mielenkiintoista kokonaisuutta (kuva 19). Hahmotteluvaiheessa hahmosta tehtiin muutama eri versio, joista parasta alettiin työstämään pitemmälle.

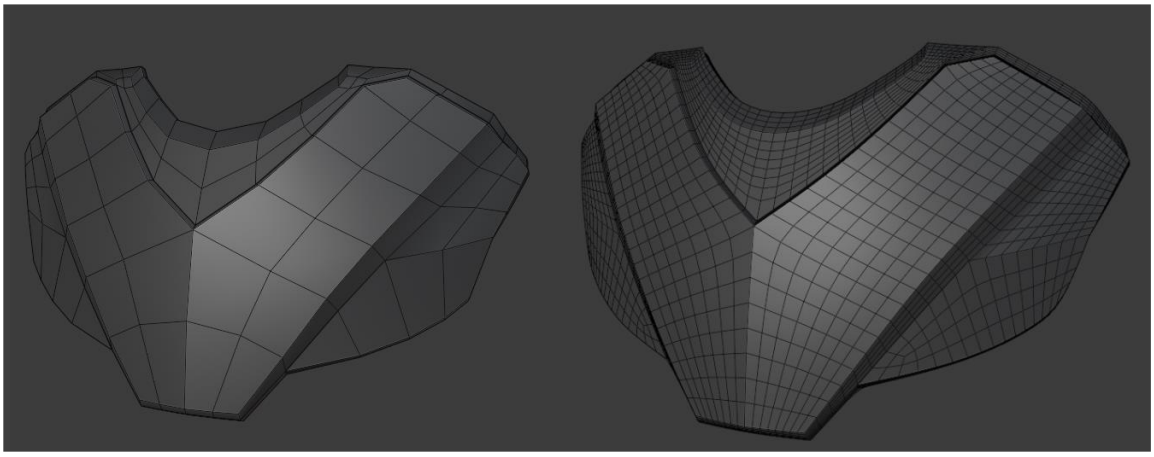


Kuva 19. Esimerkki lähtötilanteesta ja yhdestä hahmotteluvaiheen versiosta.

Kun mielenkiintoinen kokonaisuus löytyi, vietiin objektit Blenderiin, jossa niille tehtiin siistimpi topologia. Kovapintaiset objektit, kuten rintapanssari ja kiinnikkeet, tehtiin alusta loppuun Blenderissä. Muut objektit vietiin takaisin Zbrushiin, jossa niihin tehtiin vielä muita yksityiskohtia, kuten laskoksia ja saumoja. Siistitty topologia on nähtävissä kuvassa 20.

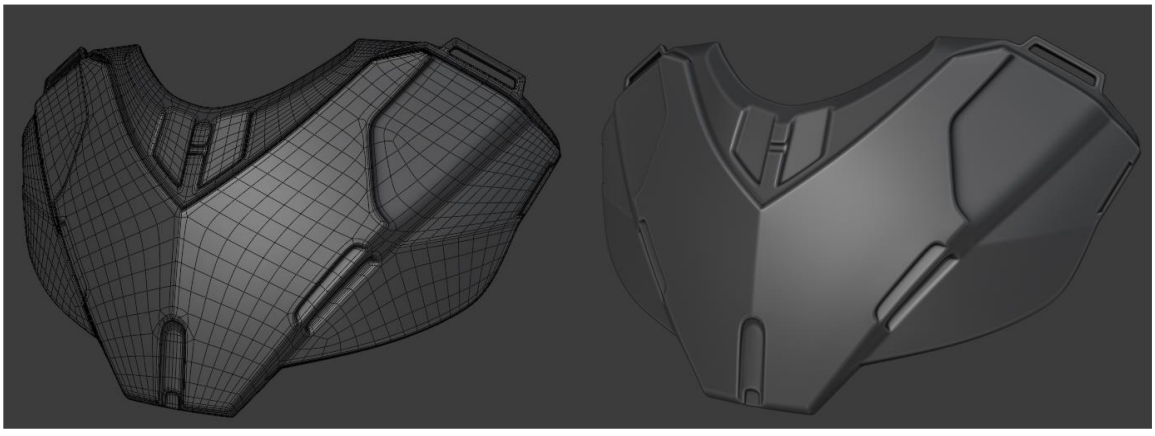
Kuva 20. Varusteiden siistitty topologia.

Ennen pienempien yksityiskohtien mallintamista rintapanssarin verteksimäärää nostettiin kahdella subdivision-tasolla (kuva 21). Tällä tavalla pintaan ei tule vääristymiä, jotka aiheutuvat tiheämmästä geometriasta yksityiskohtien kohdalla verrattuna muuhun objektiin. Kulmiin ei tässä vaiheessa vielä haluttu pyöristystä, joten niiden Creasing-arvoksi määritettiin 1, mikä sai ne säilymään terävinä.



Kuva 21. Verteksien määrän lisääminen yksityiskohtien tekemistä varten.

Tämän jälkeen alettiin lisäämään yksityiskohtia. Kuvasta 22 nähdään, että verteksimäärän lisäämisen ansiosta pinta säilyy siistinä, vaikka topologia ei olekaan täydellinen. Lopuksi kulmia pyöristettiin hyödyntämällä bevel-modifieria ja vielä yhtä subdivision-tasoa. Kiinnikkeet olivat yksinkertaisempia, eikä niihin tehty yksityiskohtia. Ainoastaan kulmia pyöristettiin raskasobjektia varten.



Kuva 22. Rintapanssarin korkearesoluutioisen version lopullinen topologia ja yksityiskohdat.

Muut osat vietiin Zbrushiin, jossa niihin veistettiin yksityiskohtia. Koska kyse oli vaatteista ja muista varusteista, joiden pinnan ei tarvitse olla tasainen, ne toteutettiin eri tavalla. Jokaiseen

veistettävään objektiin lisättiin Zbrushissa riittävä määrä jakoja käyttäen saman tyyppistä subdivide-toimintoa kuin Blenderissä, kunnes verteksejä oli riittävästi (kuva 23). Myös rintapanssari ja kiinnikkeet vietiin lopulta Zbrushiin, mutta niihin ei tehty muutoksia.

Kuva 23. Yksityiskohtien veistäminen Zbrushissa.

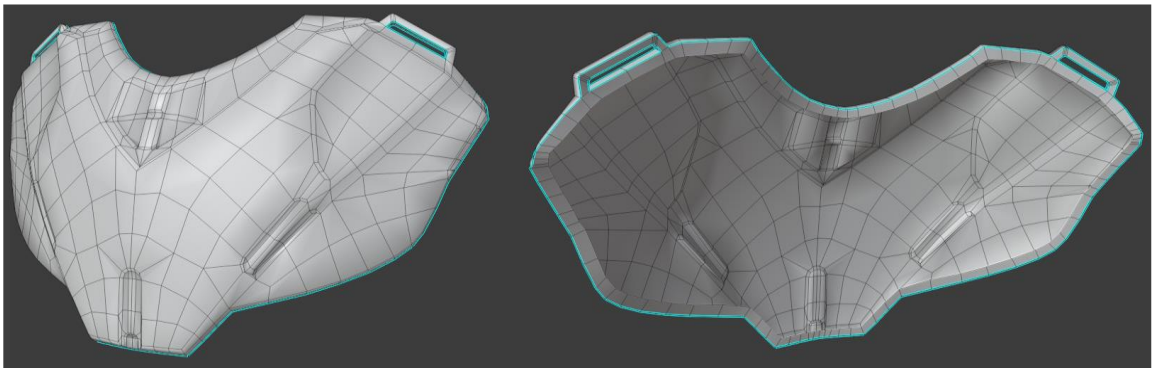
Kun kaikki yksityiskohdat oli veistetty, hahmoteltiin tekstuureja ja materiaaleja maalaamalla eri värejä objektiin pintaan. Lopullinen raskasobjekti koostui 20 pienemmästä objektista (kuva 24).

Kuva 24. Lopullinen raskasobjekti.

5.1.2 Peliobjekti

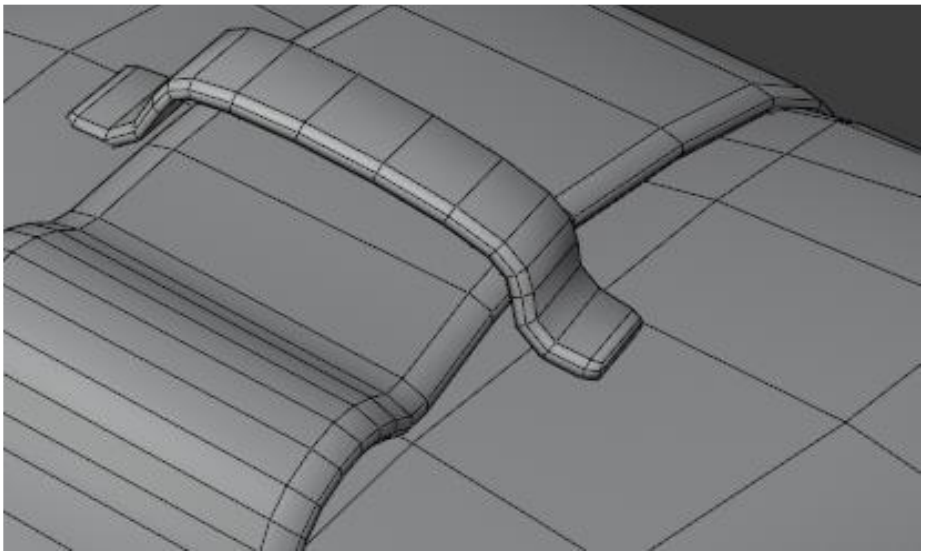
Optimoinnin kannalta tavoitteena oli tehdä objekti, joka sopisi reaaliajassa pyörivään pelimaailmaan. Koska hahmosta tehtiin pelkkä yläosa, lopullisen verteksimäärän tuli olla suhteutettu siihen. Koska raskasobjektien topologiasta tehtiin jo aiemmin siisti, Zbrushissa veistettyjen objektien toiseksi alimmasta subdivision-tasosta tehtiin suoraan lopulliset peliobjektit. Rintapanssarin matalaresoluutioinen versio tehtiin manuaalisesti Blenderissä. Geometriasta poistettiin sisäänpäin jäävä puoli ja toisten objektien peittämät polygonit verteksien säästämiseksi.

Polygonien määrän ansiosta suurempia varjostusartefakteja, jotka voisivat aiheuttaa ongelmia myöhemmässä vaiheessa, ei ollut havaittavissa. Ongelmallisin kohta oli rintapanssari, mikä johtui jyrkemmistä pinnanmuotojen vaihteluista. Varjostusta olisi voinut helposti korjata lisäämällä verteksejä tai varjostussaumoja jyrkkien kulmien ympärille, mutta suurin osa työstä jätettiin tietoisesti normaalikartan tehtäväksi. Näin tarvittiin vähemmän UV-saumoja, ja UV-koordinaateissa pystyttiin käyttämään isompia saarekkeita. Varjostussaumoja käytettiin ainoastaan kiinnikkeissä ja rintapanssarin sisäänpäin jäävän puolen reunalla (kuva 25).



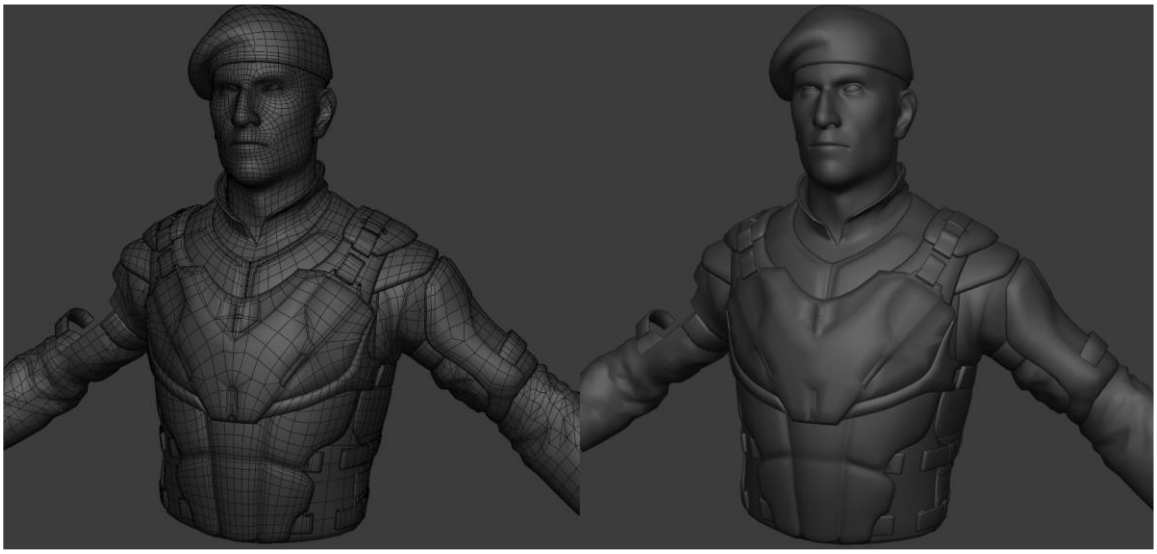
Kuva 25. Varjostussaumojen käyttö. Sininen viiva kuvaa varjostussaumoiksi määritettyjä edgejä.

Muissa objekteissa, kuten kiinnikkeiden hihnoissa, käytettiin ylimääräistä geometriaa varjostussaumojen sijaan, koska hyvin pienien UV-saarekkeiden määrää haluttiin pitää alhaisena (kuva 26). Yksi vaihtoehto olisi ollut jättää myös lisäämättä ylimääräisiä verteksejä, mutta silloin varjostus olisi ollut sotkuisempi ja ongelmia olisi voinut ilmetä tekstuurin resoluution kanssa.



Kuva 26. Ylimääräinen geometria kiinnikkeiden hihnojen reunoissa.

Lopullinen verteksien määrä oli 19 000:n paikkeilla, mikä oli järkevissä rajoissa optimoinnin kannalta. Koko hahmon tekeminen samalla tarkkuudella olisi todennäköisesti nostanut verteksimäärän 50 000:n ja 60 000:n välille, mikä arvioitiin täysin sopivaksi nykyteknologialla. Peliobjektin topologia on nähtävissä kuvassa 27.



Kuva 27. Peliobjektin lopullinen topologia polygoniverkon kanssa ja ilman.

5.1.3 Symmetria ja tekstuureiden käytön optimointi

Tekstuurien käytön optimoimiseksi monessa kohtaa hyödynnettiin symmetriaa. Objektit, jotka olivat muodoltaan samanlaisia ja riittävän kaukana toisistaan tai joissa tekstuureiden symmetria ei näkyisi liian selvästi edestäpäin katsottuna, peilattiin toiselle puolelle hahmoa käyttäen samaa kohtaa UV-koordinaatistossa. Tämän ansiosta tekstuureista voitiin tehdä tarkempia, koska UV-saarekkeita muodostui vähemmän, ja olemassa olevia voitiin skaalata suuremmiksi.

Jotkin objektit, kuten kiinnikkeet, eivät hyödyntäneet symmetriaa, mutta olivat muuten keskenään identtisiä, joten niistä tehtiin aluksi vain yksi kappale, jota tekstuureiden beikkaamisen jälkeen kopioitiin eri puolille hahmoa. Kuvassa 28 vihreällä värillä havainnollistetaan symmetrisiä ja punaisella ei-symmetrisiä pintoja. Kaikki lähellä symmetria-akselia olevat pinnat tehtiin UV-koordinaattien suhteen ei-symmetrisiksi, etteivät tekstuurit näyttäisi peilikuvalta toisella puolella symmetria-akselia.

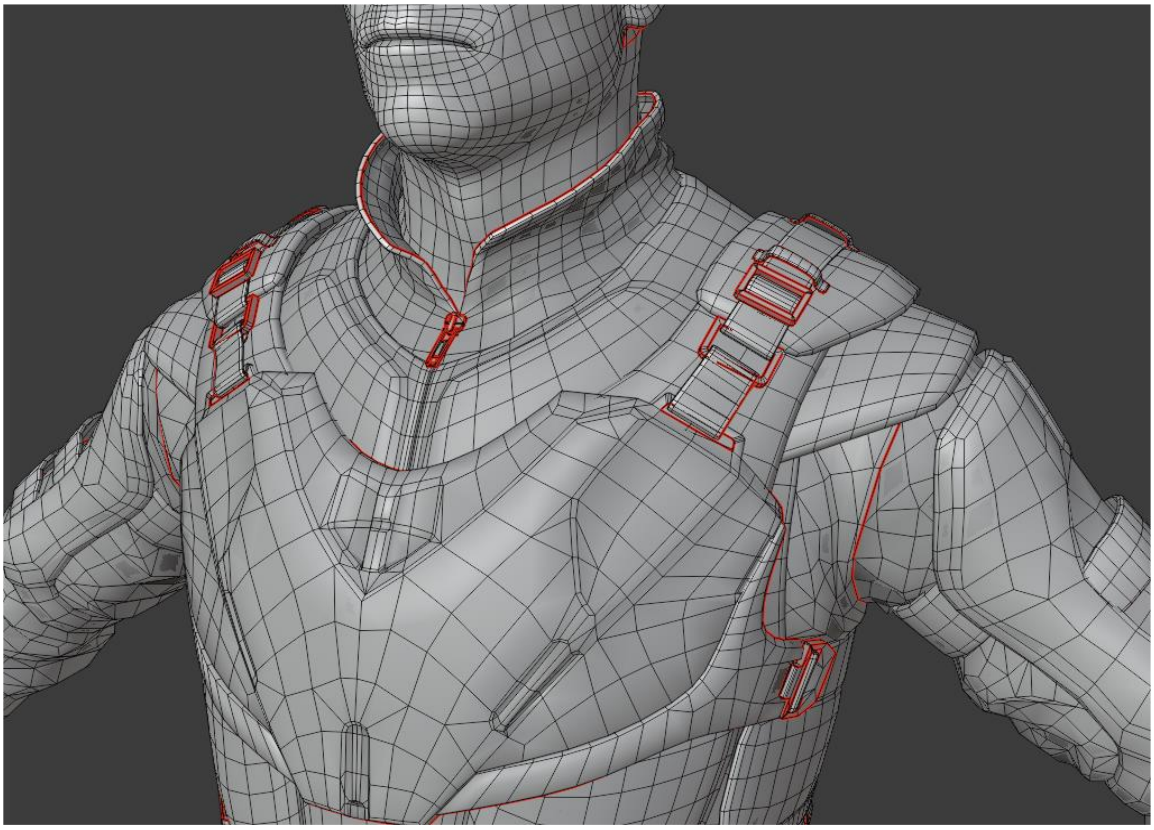
Kuva 28. Symmetria. Vihreät pinnat ovat UV-koordinaattien suhteen symmetrisiä, eli ne käyttävät samaa kohtaa tekstuurista kuin toinen puoli. Punaiset ovat ei symmetrisiä ja vievät siten enemmän tilaa UV-koordinaatistossa.

5.2 UV-kartoitus

Peliobjekti jaettiin kahteen eri materiaaliin, joista kummatkin käyttivät omia tekstuurejaan. Päälle ja varusteille annettiin eri materiaalit johtuen niiden eri tekstuurivaatimuksista. Esimerkiksi pää ei tulisi tarvitsemaan metallinkiiltoa eivätkä varusteet ihon renderöintiin tarvittavia tekstuureita. UV-kartoitusta varten peliobjektin eri osia ei tarvinnut yhdistää samaan objektiin, koska Blenderissä on mahdollista valita useita objekteja ja tehdä niihin muutoksia samanaikaisesti. Tästä oli hyötyä myöhemmin, kun objektit hajautettiin beikkaamista varten.

Ainoat varjostussaumojen määräämät UV-saumat olivat rintapanssarin takaosassa sekä kiinnikkeissä. Muut UV-saumat piilotettiin vaikeasti havaittaviin kohtiin objektin takareunoihin tai kohtiin, joissa varusteissa on luonnollinen sauma, kuten paidassa olkapään ja hihan välisessä koh-

dassa. Päässä saumat sijoitettiin korvien taakse ja takaraivoon. Monessa osassa saumoja ei tarvinnut käyttää ollenkaan, koska niiden toiselta puolesta poistetut polygonit toimivat ikään kuin saumoina, joiden avulla polygonit pystyttiin levittämään UV-koordinaatistolle ilman päällekkäisyyttä tai liikaa vääristymistä. Kokonaisuutta edestäpäin katsoen UV-saumoja on suurimmalta osin havaittavissa ainoastaan kiinnikkeissä (kuva 29).



Kuva 29. UV-saumojen sijoittelu. Punaiset viivat kuvaavat UV-saumoja.

Kiinnityshihnojen UV-koordinaateille tehtiin erikoistoimenpiteitä. Samoja työkaluja käyttämällä niiden UV-koordinaatteihin tulisi mutkia, mikä on täysin odotettua, koska niiden geometria ei ole täysin suorassa. Teksturointia varten ne päätettiin suoristaa Blenderin follow active quads -toiminnolla.

Beikkaamista varten symmetristen objektien toisen puolen UV-koordinaatteja ei siirretty normaalin UV-koordinaatiston rajojen ulkopuolelle yhden koordinaattiyksikön päähän. Sen sijaan symmetria otettiin pois päältä. Tämä oli mahdollista, koska minkään symmetrisen objektin geometria ei ollut kosketuksissa toiseen puoleensa, vaan ne olivat käytännössä eri objekteja, joilla oli saamat UV-koordinaatit. Symmetria laitettiin takaisin päälle beikkaamisen jälkeen.

5.3 Beikkaaminen

Tässä vaiheessa sekä peli- että raskasobjektit olivat päällekkäin niissä kohdissa, joihin ne alun perin mallinnettiin. Beikkaamista varten ne hajautettiin siten, että mitkään samaan objektiin kuuluvat osat eivät olisi liian lähellä toisiaan, mutta vastaavat osat peli- ja raskasobjektissa olisivat päällekkäin. Peliobjektista tehtiin kopio hajauttamista varten, että sitä ei tarvitsisi myöhemmin kasata uudestaan.

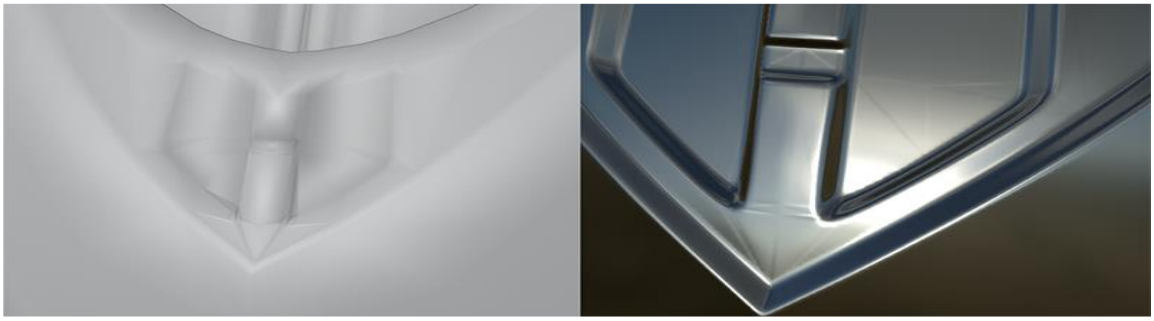
Kuva 30. Hajauttaminen beikkaamista varten.

Sekä peli- että raskasobjektit vietiin FBX-tiedostoina Substance Painteriin, jossa tekstuurien beikkaaminen suoritettiin. Koska Substance Painter ja Blender käyttävät samaa MikkTspace-tangenttiperustaa, vientiasetuksiin ei tarvinnut sisällyttää tangentteja.

Aluksi tehtiin muutamia testejä, joiden tarkoitus oli paikantaa mahdolliset ongelmat. Renderöinti tehtiin resoluutiolla 2048x2048, mikä oli myös lopullinen resoluutio tekstuureille. Pienempää resoluutiota ei käytetty testejä varten, koska beikkaaminen on Substance Painterissä nopeaa. Ongelmakohtien paikantamista varten luotiin materiaali, jonka metallisuus ja kiiltävyysarvot olivat täysillä. Tämä korosti objektin varjostukseen liittyviä ongelmia.

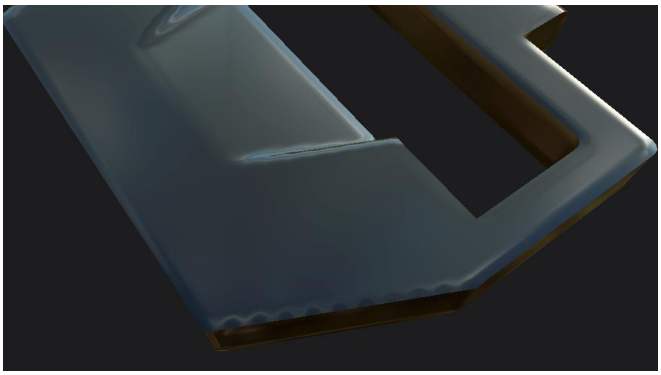
Yksi ongelma oli tekstuurien resoluutio, joka teki varjostusartefakteista näkyvämpiä rintapanssarissa (kuva 31) ja aiheutti sahalaitaisia reunoja erityisesti pienemmissä objekteissa (kuva 32). Resoluutiota ei haluttu nostaa optimointisyistä. Sen sijaan pienempien objektien kokoa nostettiin

UV-koordinaatistossa, koska tyhjää tilaa oli jäänyt isompien UV-saarekkeiden väliin. Muille varjostusartefakteille ei päätetty tehdä mitään, koska niitä oli huomattavasti vaikeampi erottaa kauempaa ja silloin, kun ei käytettäisi kiiltävää materiaalia.



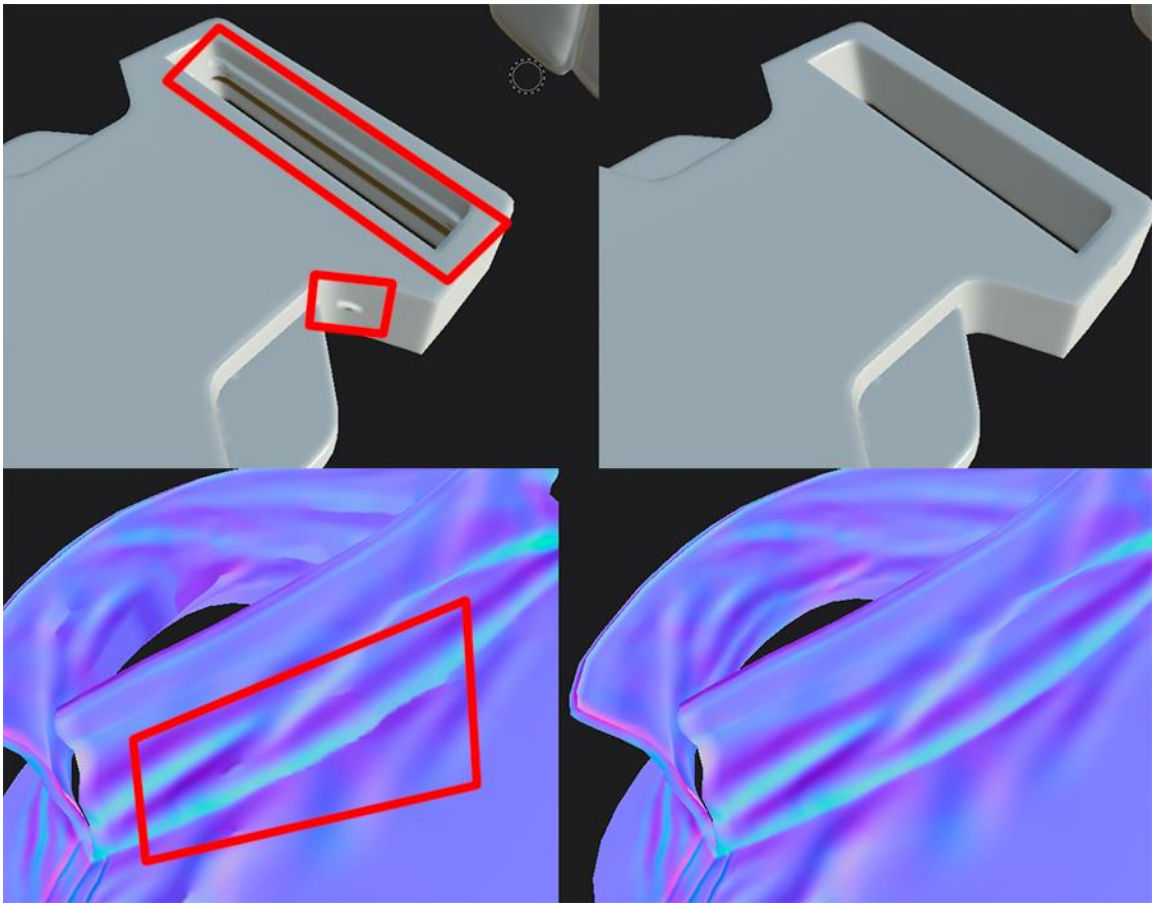
Kuva 31. Ongelmakohta rintapanssarissa. Vasemmalla ilman normaalikarttaa ja oikealla normaalikartan kanssa.

Kuvasta 31 on nähtävissä, että jyrkät muutokset geometriassa voivat aiheuttaa ongelmia myöhemmässä vaiheessa myös normaalikartan kanssa. Ongelman olisi voinut korjata käyttämällä hard edgejä tai lisäämällä edgeloopeja ongelmakohdan ympärille.



Kuva 32. Ongelmakohta kiinnikkeessä.

Toinen ongelma tuli vastaan säteiden lähettämisetäisyydessä, mikä johtui kokoerosta pienempien ja isompien objektien kohdalla. Sekä liian suuri että liian pieni arvo aiheutti ongelmia eri kohdissa objektia (kuva 33). Alkuperäinen arvo oli 0.01, mikä oli liian suuri tässä tapauksessa. Muutaman kokeilun jälkeen sopivaksi etäisyydeksi valittiin 0.0028 Substance Painterin ”Bake mesh maps” -asetusten max frontal ja max rear distanceen.



Kuva 33. Säteiden lähettämisetäisyydestä johtuvat ongelmat. Vasemmalla ylhäällä arvo on liian suuri ja vasemmalla alhaalla liian pieni. Oikeassa reunassa lopputulos sopivan etäisyyden löydyttyä.

Ongelmien korjaamisen jälkeen renderöinti suoritettiin reunanpehmennystä käyttäen resoluutiolla 4096x4096. Suuremman resoluution käyttäminen teksturointivaiheessa on yleinen käytäntö. Lopputuloksesta tuli riittävän hyvä, vaikka varjostusartefakteja jäikin muutamaan kohtaan (kuva 34).

Kuva 34. Beikkaamisen lopputulos. Vasemmalla ilman normaalikarttaa ja oikealla normaalikartan kanssa.

5.4 Teksturointi ja lopputuloksen tarkastelu

Beikkaamista varten kaikki objektit oli hajautettu eikä duplikaatteja ollut mistään osista. Teksturoidessa käytettiin eri tiedostoa, johon peliobjektista tehtiin kaksi versiota. Yhdessä versiossa kaikki osat olivat oikeilla paikoillaan ja symmetria päällä, että nähtäisiin miltä tekstuurit ja objektit näyttäisivät kokonaisuudessaan. Toinen versio oli hajautettu, kuten beikkaamisessavaiheessa, että jokainen pinta päästäisiin teksturoimaan ilman, että eri osat olisivat toistensa edessä. Hajautettu versio siirrettiin Blenderissä kasatun version yläpuolelle, ja kummatkin vietiin samassa tiedostossa Substance Painteriin. Beikattu normaalikartta ja muut teksturoinnissa käytetyt tekstuurit tuotiin beikkaamistiedostosta teksturointitiedostoon.

Teksturointivaiheen jälkeen tekstuurit vietiin Blenderiin, jossa lopputulosta tarkasteltiin reaaliaikaista renderöintimoottoria Eeveetä käyttäen. Varjostusartefaktien erottaminen oli huomattavasti vaikeampaa, mikä johtui osittain siitä, että yksityiskohtien lisäämisen jälkeen huomio ei kiinnity yhtä helposti pieniin virheisiin objektin pinnalla. Toisaalta niitä oli myös vaikeampi erottaa, koska enää ei käytetty kiiltävää materiaalia (kuva 35).

Kuva 35. Lopputulos tekstuureineen Blenderin Eeveellä renderöitynä.

6 Pohdintaa

Laadultaan mallinnetusta objektista tuli tyydyttävä: Se täyttää reaaliaikaiseen renderöintiin tarkoitetun peliobjektin vaatimukset ja vastaa opinnäyteyön käsittelemää aihealuetta. Kuitenkin jälkeenpäin mietittynä osuvampiakin valintoja olisi voinut tehdä projektiosuuden suhteen.

6.1 Työn arviointia

Koska aiheena oli siistien normaalikarttojen renderöinti, osuvampi valinta mallintamisen kohteeksi olisi voinut olla scifi-teemainen proppi. Objekti, jossa on sekoitettuna paljon siistejä kaarevia pintoja ja kulmikkaita muotoja sekä pienempiä yksityiskohtia, olisi voinut tuoda selkeämmin esille normaalikarttojen beikkaamisen yhteydessä esiintyviä ongelmia. Toisaalta myös kokonainen hahmo olisi voinut olla huomattavasti mielenkiintoisempi puolikkaan sijaan.

Toteutuksessa vastaan tuli yleisimpiä ongelmia, joiden ratkaisemiseen löytyi keino. Täydellisyyttä ei lähdetty tavoittelemaan, koska optimointia tehdessä on myös tärkeää pystyä arvioimaan, milloin ongelmat, kuten varjostusartefaktit, ovat niin isoja, että ne pitää korjata, ja milloin niiden voidaan antaa olla, koska ajan tehokas käyttäminen on myös osa peliassettien tekemistä. Pienet varjostusvirheet olisi voinut korjata esimerkiksi lisäämällä geometriaa ongelmakohtien jyrkkien muotojen ympärille, mikä olisi lisännyt verteksinormaalien määrää ja tehnyt varjostuksesta siistimmän myös ilman normaalikarttaa.

6.2 Onko beikkaaminen tarpeellista tulevaisuudessa?

Normaalikartoille on todennäköisesti käyttöä vielä pitkään, koska pienten pinnanmuotojen simulointi niiden avulla on tehokasta ja helppoa. Jos johonkin pintaan halutaan pienillä yksityiskohdilla luoda illuusio mistä tahansa materiaalista, kuten puusta tai kivetyksestä, on järkevämpää käyttää normaalikarttaa kuin geometriaa.

Sen sijaan voidaan miettiä, onko normaalikarttojen beikkaaminen yhtä tarpeellista tulevaisuudessa. Esimerkiksi Unreal 5 -pelimoottorin trailerissa käytettiin useita assetteja, jotka olivat verteksimäärältään raskasobjekteja. Matalaresoluutioista objektia tai beikkaamista ei tehty, mikä säästää aikaa. Normaalikarttojen käyttäminen ja siten matalaresoluutioisemman geometrian

käyttö säästäisi kuitenkin laskutehoa, mitä voitaisiin hyödyntää pelin muilla osa-alueilla. Tietenkin, jos peliä pyörittävät koneet olisivat niin tehokkaita, että normaalikartoilla säästetty verteksimäärä ei toisi merkittävää hyötyä, niin beikkaamistarve vähenisi.

Jos esimerkiksi pilvipelaamisesta tulisi suosittua, pelaajilla olisi pääsy tehokkaisiin laitteisiin, joista peli striimattaisiin pelaajille internetin välityksellä. Tämä voisi vähentää optimoinnin tarvetta, mutta aiheuttaisi muita ongelmia. Siinä tapauksessa pelin kehityksessä olisi tehtävä jo alusta lähtien päätös, että peli tulee vain tietylle alustalle, koska jokaisen 3D-assetin optimointi eri alustoille kääntämistä varten toisi vain lisää työtä, jonka olisi voinut välttää suoraan käyttämällä beikkaamista. Voidaan myös arvioida, kuinka todennäköistä pilvipelaamisen yleistyminen on. Olisivatko pelaajat valmiita luopumaan esim. pelien modaamisesta, jos eivät pääse käsiksi pelin tiedostoihin? Jos pilvipelaamisesta tulisi äkillisesti äärimmäisen suosittua, kestäisikö infrastruktuuri datan siirron nousun? Mutta ainahan normaalikarttojen beikkaamisen voisi myydä ympäristötekona, koska vähemmän laskettavaa tarkoittaa pienempää energiankulutusta, oli sitten kyseessä oma tietokone tai jonkin yrityksen massiivinen tietokonefarmi.

Beikkaamistarve vähenee joka tapauksessa laitteiston kehittyessä, ja todennäköisesti sen aiheuttaa tehokkaampien ja edullisten tietokonekomponenttien yleistyminen. Esimerkkinä laskutehon kasvamisesta vuonna 1996 julkaistussa Super Mario 64 -pelissä Marion kolmiomäärä oli noin 800 [26], kun taas vuonna 2019 julkaistussa pelissä Resident Evil 2 Remake päähahmossa oli noin 140 000 kolmiota [27]. Nykyään pelihahmon sormen kynnessä on siis enemmän geometriaa kuin 20 vuotta sitten koko hahmossa.

Voi myös olla, että pullonkaula korkean ruudunpäivitysnopeuden saavuttamiseksi siirtyy johonkin muuhun osa-alueseen kuin geometrian optimointiin. On myös mahdollista, että tekstuurien viemä muisti tulee olemaan isompi ongelma kuin verteksien määrä, ja se onkin yksi muista peligrafiikan pitkäaikaisimmista haasteista.

7 Yhteenveto

Normaalikarttojen oikeaoppinen beikkaaminen on prosessi, joka vaatii laajaa tietämystä myös sitä edeltävistä työvaiheista. On hallittava topologian tehokas käyttäminen optimoinnin ja varjostuksen siisteyden vuoksi. UV-kartoituksessa on tiedettävä, minne ja miten paljon saumoja kannattaa määrittää sekä mitä muita työkaluja voidaan hyödyntää, kuten UV-koordinaattien suoristamista. Käsitteistä on tärkeää hallita varjostukseen liittyvät perusteet, kuten verteksinormaalit sekä varjostussaumat, ja miten eri tekniikat varjostuksen kontrollointiin vaikuttavat myöhempiin työvaiheisiin. Beikkaamisessa olennaista on tietää erot synkronoidun ja ei-synkronoidun työnkulun välillä ja selvittää, mitä tangenttiperustaa assettien tekemiseen käytetyt sovellukset käyttävät.

Projektityöstä ilmeni, että ohjeita noudattamallakaan kaikilta pieniltä varjostusvirheiltä on vaikea välttyä. Lopputulosta tarkastellessa voidaan kuitenkin vetää johtopäätös, että kaikkia niitä ei tarvitsekaan välttää, koska viimeisten työvaiheiden jälkeen niitä ei pysty erottamaan, ellei tiedä tarkalleen mihin katsoa ja tarkastele hyvin läheltä. Voidaan myös kysyä, ovatko pienet virheet sellaisia joihin pelaajat kiinnittävät huomiota.

Yksi tärkeä elementti on myös ajankäyttö. Koska peliassettien tekeminen alusta loppuun on pitkä ja monivaiheinen prosessi, on eri työvaiheet tehtävä tehokkaasti. Jos esimerkiksi teksturointivaiheessa huomataan jokin virhe, joka vaatii muutoksia objektin topologiaan ja UV-koordinaatteihin, ja siten myös normaalikartan uudelleen beikkaamista, säästetään paljon aikaa, jos peliobjekti on tehty niin hyvin, että beikkaamisesta selvitään ilman häkkiä. Muuten häkki pitäisi asettaa manuaalisesti joka kerta uudestaan, kun virheitä korjataan. Ajan säästämiseksi on myös osattava arvioida, milloin virheet vaativat korjaamista vai voidaanko ne piilottaa jollain tavalla. Paras vaihtoehto voi myös olla antaa niiden olla.

Lähteet

1 Leonard T. (15.10.2010). CG Science for Artists – Part 1: Real-Time and Offline Rendering. Haettu 2.10.2020, sivustolta CG Channel. Internetosoite: http://www.cgchannel.com/2010/10/cg-science-for-artists-part-1-real-time-and-offline-rendering/

2 Lemos C. (24.3.2020). Tutorial: How Normal Maps Work & Baking Process. Haettu 2.10.2020, sivustolta 80 level. Internetosoite: https://80.lv/articles/tutorial-how-normal-maps-work-baking-process

3 Cignoni P, Montani C, Rocchini C, Scopigno R. (1998). A general method for preserving attribute values on simplified meshes (PDF). IEEE Visualization 1998. Haettu 4.9.2020. Internetosoite: http://vcg.isti.cnr.it/publications/papers/rocchini.pdf

4 Totten C. Game Character Creation with Blender and Unity. Indianapolis: John Wiley & Sons, Incorporated; 2012.

5 Bech-Yagher C. (23.10.2018). UV mapping for beginners. Haettu 2.10.2020 sivustolta creativebloq. Internetosoite: https://www.creativebloq.com/features/uv-mapping-for-beginners

6 handplane3d. (14.2.2013). Controlling Shading Behavior. Haettu osoitteesta https://www.youtube.com/watch?v=ciXTyOOnBZQ

7 Rodriguez E. Computer Graphic Artist. Delhi: Global Media; 2006.

8 GuerrillaCG. (23.11.2008). Smooth Shading. Haettu osoitteesta https://www.youtube.com/watch?v=PMgjVJoglbc

9 Math Open Reference. Normal. Haettu 18.7.2020. Internetosoite: https://www.mathopenref.com/normal.html

10 Polycount wiki. Vertex Normal. Haettu 18.7.2020. Internetosoite: http://wiki.polycount.com/wiki/VertexNormal

11 Polycount wiki. Texturing. Haettu 18.7.2020. Internetosoite: http://wiki.polycount.com/wiki/Texturing

12 Makin' Stuff Look Good. (3.4.2017). Shader Fundamentals – Normal Mapping. Haettu osoitteesta https://www.youtube.com/watch?v=6_-NNKc4lrk&t=266s

13 CG Cookie. (28.4.2017). Intro to Normal Maps in Blender (And Why Games Use Them). Haettu osoitteesta: https://www.youtube.com/watch?v=GyfVQmoFX4c

14 CG Cookie. (2.8.2018). Deconstructing a Normal Map (CGC Weekly #18). Haettu osoitteesta https://www.youtube.com/watch?v=oOOeV3IU2Yo

15 Polycount wiki. Normal Map Technical Details. Haettu 18.7.2020. Internetosoite: http://wiki.polycount.com/wiki/Normal_Map_Technical_Details#Tangent_Basis

16 Cryengine. Tangent Space Normal Mapping. Haettu 18.7.2020 osoitteesta https://docs.cryengine.com/display/SDKDOC4/Tangent+Space+Normal+Mapping#TangentSpaceNormalMapping-FurtherReading

17 Polycount wiki. Texture Baking. Haettu 18.7.2020. Internetosoite: http://wiki.polycount.com/wiki/Texture_Baking

18 Substance by Adobe. (13.8.2015). Substance Painter Tutorial – Fundamentals 05: Baking textures. Haettu osoitteesta https://www.youtube.com/watch?v=ePnLTuzRAbg

19 Substance by Adobe. (13.8.2015). Substance Painter Tutorial – Model Preparation 03: Tangents & shading for Normal maps. Haettu osoitteesta https://www.youtube.com/watch?v=gR3r7Xmhmlk&list=PLB0wXHrWAmCx994Cb7iRFSmupY-HFw5DTx&index=21

20 Danan. (21.9.2016). Why Do We Need Topology in 3D Modeling. Haettu 18.7.2020, sivustolta thilakanathan studios. Internetosoite: http://thilakanathanstudios.com/2016/09/why-do-we-need-topology-in-3d-modeling/.

21 Polycount wiki. Subdivision Surface Modeling. Haettu 18.7.2020. Internetosoite: http://wiki.polycount.com/wiki/Subdivision_Surface_Modeling

22 Péloquin, S. (13.11.2018). Best Practices for Hard Surface Normal Map Baking. Haettu 2.10.2020, sivustolta 80 level. Internetosoite: https://80.lv/articles/best-practices-for-hard-surface-normal-map-baking/

23 Polycount wiki. Normal map Modeling. Haettu 2.10.2020. Internetosoite http://wiki.polycount.com/wiki/Normal_Map_Modeling

24 Nimimerkki EarthQuake. (10/2012). You’re making me hard. Making sense of hard edges, uvs, normal maps and vertex counts. Haettu 2.10.2020 sivustolta Polycount. Internetosoite: https://polycount.com/discussion/107196/youre-making-me-hard-making-sense-of-hard-edges-uvs-normal-maps-and-vertex-counts/p1.

25 Nimimerkki EarthQuake. (2/2015). Of Bit Depths, Banding and Normal Maps. Haettu 2.10.2020 sivustolta Polycount. Internetosoite: https://polycount.com/discussion/148303/of-bit-depths-banding-and-normal-maps/p1

26 Nimimerkki Penguin Mario. A Close Look at Mario Models Throughout the Years. Haettu 6.2.2021. Internetosoite: https://www.marioboards.com/threads/38871/

27 Nimimerkki goekbenjamin. Polycounts in next gen games thread! Haettu 6.2.2021 sivustolta Polycount. Internetosoite: https://polycount.com/discussion/141061/polycounts-in-next-gen-games-thread