Outi Ohra-aho

3D-mobiilipelihahmon digitaalinen veistäminen

Tradenomi

Tietojenkäsittely

Kevät 2018

KAJAANIN
AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

KAJAANIN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

Tiivistelmä

Tekijä(t): Ohra-aho Outi

Työn nimi: 3D-mobiilipelihahmon digitaalinen veistäminen

Tutkintonimike: Tradenomi (AMK), tietojenkäsittely

Asiasanat: 3D, peli, videopeli, mobiilipeli, pelihahmo, digitaalinen veistäminen, 3D-veistäminen, 3D-mallinnus, ZBrush

Opinnäytetyössä tutkittiin digitaalisen veistämisen ottamista osaksi 3D-mobiilipelihahmon työnkulkua. Perinteisesti mobiilipelihahmot eivät ole voineet hyödyntää joitain 3D-veistämisen etuja kuten normaalikarttoja, joten ne on usein mallinnettu alusta loppuun traditionaalisissa mallinnusohjelmissa kuten 3ds Max tai Blender. Teknologian kehittyessä mobiililaitteet tukevat entistä monimutkaisempia grafiikoita ja tutkimuksessa selvitettiin, mitä hyötyä digitaalisen veistämisen ottamisesta osaksi mallinnusprosessia on nyt tai tulevaisuudessa.

Teoriaosuudessa syvennyttiin mobiilipeleihin ja pelihahmonkehityksen työnkulkuun. Työ toteutettiin digitaalisesti veistämällä ZBrushissa 3D-hahmo, joka retopologisoitiin Blenderissä. Raskasmallista beikattiin erilaisia tekstuurikarttoja, joita hyödynnettiin teksturointiprosessissa. Lopuksi hahmo autorigattiin Mixamossa ja renderöitiin Marmoset Toolbag 3:ssa.

Työn tuloksissa todettiin, että digitaalinen veistäminen on intuitiivisempaa ja luovempaa varsinkin henkilölle, jolla on jo perinteistä taidetaustaa. Lopputuloksesta tuli näyttävämpi kuin mitä perinteisesti mallintamalla olisi osattu tehdä, mutta itse prosessi oli hieman hitaampi. Käytännön työssä opittiin, millaisia ongelmia eri työvaiheissa voi tulla vastaan ja miten niitä voidaan välttää tulevaisuudessa. Esimerkiksi tekstuurikarttojen beikkaus ei ollut yksinkertaista, vaan jouduttiin opettelemaan muun muassa mallin osien toisistaan räjäyttäminen ja eri ohjelmien väliset erot tangenttiperustassa. Työssä opittiin henkilökohtaisella tasolla paljon ohjelmistosta ja veistämisestä.

Tulevia käyttö- tai jatkomahdollisuuksia työlle voisivat olla muun muassa erilaisten mallinnus- ja teksturointiohjelmien vertailu, autoretopologiatyökalut, PBR-materiaalit mobiilialustoilla ja viehättävän mobiilipelihahmon suunnittelu.

KAJAANIN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

Abstract

Author(s): Ohra-aho Outi

Title of the Publication: Sculpting a 3D Mobile Game Character

Degree Title: Bachelor of Business Administration, data processing

Keywords: 3D, game, video game, mobile game, game character, sculpting, 3D modeling, ZBrush

This thesis explores adopting 3D sculpting as part of the development workflow for creating 3D mobile game characters. Traditionally mobile games have not been able to utilize some perks of sculpting a high-poly mesh (such as normal maps) which is why mobile game characters have often been modeled entirely in traditional 3D modeling software like 3ds Max or Blender. Mobile platforms will be able to support more complex graphics in the future as technology keeps moving forward.

The implementation was done by sculpting a 3D character in ZBrush and optimizing the model to be available for mobile game use by retopologizing it in Blender and utilizing the high-poly model in the baking and texturing process. The character was auto-rigged in Mixamo and rendered in Marmoset Toolbag 3. The theory focuses mostly on mobile games and the general workflow of creating 3D video game characters.

It was discovered that digital sculpting is more intuitive than traditional modeling, especially for someone who already possesses a background in traditional arts. The model was superior to what would have been the result of traditional modeling and hand painting, although the process itself was slightly slower. Multiple challenges were resolved during the project which made the learning process a success.

Some future research possibilities presented by this thesis include exploring the pros and cons between different modeling and/or texturing software, retopologizing a character using auto-retopology tools, studying PBR-materials in mobile games and designing an appealing mobile game character.

Alkusanat

Kiitos perheelleni, joka on uskonut minuun aina ja varsinkin, kun lähdin opiskelemaan videopelejä Kainuuseen. Kiitos kaikille ystävilleni, jotka tukivat minua opinnäytetyöprosessissa - etenkin Saanalle, Eerolle, Jannelle, Jarkolle, Jesselle, Tommille, Mikalle, Samulille, Santerille, Teijalle, Antille, Juholle, Kasperille ja Davidille. Kiitos Samille, Hegelle, Youngminille ja muille kollegoilleni, jotka antoivat palautetta ja ohjeita läpi prosessin. Kiitos myös opettajilleni Kajaanin ammattikorkeakoulussa ja erityisesti ohjaajalleni Suvannolle.

Sisällys

Symboliluettelo

2D-grafiikka	kaksiulotteiset digitaaliset kuvat ja niiden tuottamiseen käytetyt tekniikat (pituus ja leveys)
3D-grafiikka	kolmen tilaulottuvuuden suhteen sisäisesti mallinnettua tietokonegrafiikkaa (pituus, leveys ja syvyys)
3D-malli	digitaalinen kappale, joka on käytännössä kokoelma pisteitä, joihin on tallennettu tieto objektin geometriasta, pinnan tekstuurista ynnä muusta
3D-skene	3D-tila, jossa pelin ympäristö, menutila tai muu vastaava sijaitsee (engl. 3D scene)
AR	lisätty todellisuus (engl. augmented reality)
assetti	digitaalinen assetti on binääritiedosto, jolla on käyttöoikeus: esimerkiksi 3D-malli (engl. digital asset)
beikkaus	datan siirtämistä yksinkertaisempaan ja pysyvämpään muotoon (engl. bake)
Blender	3D-mallinnusohjelma
diffuusikartta	määrittää 3D-mallin pinnan värin (engl. diffuse map)
digitaalinen veistäminen	3D-objektin manipulointia esimerkiksi vetäen, työntäen tai silottaen (engl. digital sculpting, 3D sculpting)
ekstruusio	polygonin tai sen osan jatkaminen vetämällä niin, että siitä syntyy lisää geometriaa (engl. extrude/extrusion)
FBX	tiedostoformaatti, jota käytetään digitaalisen sisällön siirtämiseen eri ohjelmistojen välillä (lyhenne englannin kielen sanasta Filmbox)
FPS	kuvataajuus (engl. frames per second)
isometrinen projektio	ortografista projektiota, jossa kolmiulotteinen kappale esitetään kaksiulotteisena niin, että kappaleesta näh-

	dään kolme sivua ja akseleiden suuntaiset viivat ovat samassa skaalassa
kevytmalli	polygonimesh, jonka polygonimäärä on suhteellisen matala (engl. low-poly model)
laatikkomallinnus	3D-mallinnus aloitetaan yksinkertaisesta geometrisesta muodosta, kuten laatikosta, jota muovaillaan, kunnes se on halutunlainen (engl. box modeling)
Marmoset Toolbag 3	ohjelmisto 3D-renderöintiin, animointiin ja beikkaukseen
materiaali	tietokonegrafiikassa oikeaa materiaalia simuloiva materiaali, jolla voi olla ominaisuuksia kuten hohtavuus
mesh	3D-muoto, joka koostuu polygoneista: polygoniverkko
Mixamo	Autoriggausohjelma
modifikaattori	automatisoitu toiminto, joka vaikuttaa 3D-objektiin muuttamatta objektin perusgeometriaa (engl. modifier)
normaali	kohtisuorasti pintaa leikkaavan suoran suunta (kts. verteksinormaali)
normaalikartta	kuvatiedosto, johon on tallennettu jokaisen pikselin suunta eli normaali. Vaikuttaa siihen, miten valo osuu 3D-mallin pintaan (engl. normal map).
OBJ	OBJ-tiedosto eli objektitiedosto on tiedostoformaatti 3D-mallista tai malleista
ortografinen projektio	kolmiulotteisen kappaleen esittäminen kaksiulotteisena
PBR	fysiikkaperusteinen renderöinti (engl. physically based rendering)
pelimoottori	videopelin ohjelmistokehys, jolla voi rakentaa pelejä
piirtopyyntö	kutsu materiaalilta näytölle (engl. draw call)

pikseli	voi viitata peligrafiikassa kahteen asiaan: pikseleihin, joista kuvatiedostot rakentuvat tai pikseleihin, jotka pelimoottori renderöi tietokonenäytölle (engl. pixel, lyhenne englannin kielen sanoista picture element)
piksoli	eräänlainen älypikseli, johon on tallennettu tieto myös piksolin syvyydestä, suunnasta ja materiaalista (engl. pixol)
pinnan jakaminen	mallin pinnan polygonit jaetaan niin, että pinnan geometria lisääntyy ja se on sileämpi (engl. subdivision surface)
polygoni	polygonimallin pinta koosuu polygoneista, jotka ovat yleensä kolmi- tai nelikulmaisia monikulmioita
päällepiirto	arvo, joka kertoo, kuinka monta kertaa joka pikseli jouduaan päällekirjoittamaan renderöintiprosessissa yhden kuvan tai kuvasarjan aikana (engl. overdraw)
raskasmalli	polygonimesh, jonka polygonimäärä on suuri (engl. high-poly model)
RGB-värimalli	väritila, jossa eri värejä muodostetaan sekoittamalla additiivisesti punaisen, vihreän ja sinisen väristä valoa (lyhenne englannin kielen sanoista red, green, blue)
reuna	polygonin reuna yhdistää kaksi verteksiä (engl. edge)
reunaluuppi	toisiinsa yhdistynyt reunaketju, joka kulkee meshin pinnan poikki (engl. edge loop)
renderöidä	bittikarttagrafiikkakuvan esittäminen esimerkiksi 3D-malleista tietokoneohjelman avulla
retopologia	topologian uudelleenrakentaminen
riggaus	riggaus tai skinnaus tarkoittaa polygonimeshin painottamista luurankoon niin, että se voidaan animoida (engl. rigging, skinning)

shader	tietokonekoodia, joka määrittelee, miten pinta renderöidään
sivu	polygonin verteksien ja reunojen väliin jäävä alue eli polygonin pinta (engl. face)
spekulaarikartta	määrittelee 3D-mallin pinnan heijastavuuden (engl. specular map)
tangenttiavaruus	yksi 3D-maailman koordinaattisysteemeistä, jonka tarkoitus on määrittää tekstuurikoordinaatit 3D-tilassa. Käyttää u, v, n -akseleita (engl. tangent space)
tasoittava ryhmä	ryhmä polygoneja, jotka muodostavat tasaisen pinnan (engl. smoothing group)
tekseli	tekstuurikartan pikseli (engl. texel, lyhenne englannin kielen sanoista texture element)
teksturointi	3D-mallin pinnoittaminen kuvatiedostolla eli tekstuurilla sekä tekstuurikuvien eli -karttojen tuottaminen
topologia	3D-mallin pinnan sommittelu eli miten reunat ja sivut on sijoiteltu muodostamaan meshin pinta
toteutusaste	se määrä pikseleitä, jonka näytönohjain voi renderöidä ruudulle per sekunti
UI	käyttöliittymä (lyhennys englannin kielen sanoista user interface)
UV-kartoitus	prosessi, jossa ohjelmisto venyttää kiinnitettyjen pisteiden (UV-koordinaattien) välisen tilan mallin pinnalle
UV-kartta	3D-mallin pinta purettuna 2D-tilaan teksturointiprosessia varten
UV-koordinaatit	u,v - tekstuurikoordinaatit määrittävät mallin verteksien yksittäiset sijainnit suhteessa tekstuurikuvan korkeuteen ja leveyteen

unwrappaus	mallin kolmiulotteisen XYZ-informaation "purkaminen" kaksiulotteiseen UV-tilaan (engl. unwrapping)
valokartta	kuvatekstuuri, johon on beikattu valmiiksi renderöity tieto 3D-tilan valaistuksesta (engl. lightmap)
valoluotain	tallennettu tieto valon matkustamisesta tyhjän 3D-tilan poikki (engl. light probe)
verteksinormaalit	näkymättömiä suoria, jotka osoittavat ulospäin 3D-mallin pinnasta jokaisen verteksin kohdalla
VR	virtuaalitodellisuus (engl. virtual reality)
ZBrush	ohjelmisto, jota käytetään yleisesti digitaaliseen veistämiseen

1 JOHDANTO

Digitaalinen veistäminen eli 3D-veistäminen on ikään kuin savella veistämistä, mutta saven sijaan muokataan 3D-massaa. Veistämällä on mahdollista mallintaa yksityiskohtaisempia malleja kuin perinteisillä mallinnustekniikoilla ja prosessi on luovempi. Digitaalista veistämistä ei ole käytetty yleisesti osana työnkulkua mobiilipelinkehityksessä, sillä jotkin siitä saatavat hyödyt, kuten normaalikartat, ovat olleet käyttökelvottomia mobiililaitteiden performanssiongelmien vuoksi. Teknologian kehittyessä nykypäivän mobiililaitteet tukevat yhä monimutkaisempia grafiikoita ja mobiilipeliala on kirjoitushetkellä erinomaisessa asemassa haastaakseen perinteisesti suuremmat pelialustat ja murtaakseen ihmisten ennakkoluulot mobiilipeleistä.

Opinnäytetyön aihealue on digitaalisen veistämisen ottaminen osaksi 3D-videopelihahmon työnkulun vaiheita mobiilipelinkehityksessä. Aihe valittiin, sillä se on tärkeä mobiilipeligrafiikan tulevaisuuden kannalta ja oppimisalueena paitsi kiinnostava myös hyödyttävä. Tutkimusongelma on selvittää, toimiiko digitaalinen veistäminen mobiilipelihahmon mallinnuksessa ja kannattaako sitä hyödyntää eli onko mallinnus helpompaa kuin perinteisesti polygonimallintamalla ja onko lopputulos parempi. Tavoite oli luoda dokumentti, joka kiinnostaa 3D-grafiikasta kiinnostuneita artisteja sekä kehittää ammattiosaamista työn aihepiiristä.

Työn toteutus aloitettiin tutkimalla ja analysoimalla mobiilipelejä sekä hyödyntämällä muita aiheeseen liittyviä lähteitä. Teoriaosuus jakautuu kahteen osaan, joista ensimmäisessä käsitellään mobiilipelejä yleiseti ja toisessa 3D-malleja ja niiden kehittämiseen käytettyjä tekniikkoja. Projektityöosuudessa teoriaa sovellettiin opinnäytetyötä varten kehitettyyn hahmomalliin, jonka työvaiheet käydään yksityiskohtaisesti läpi suunnittelusta veistämiseen, optimointiin ja lopuksi autoriggaukseen ja renderöintiin.

Aikaisempia opinnäytetöitä samaa aihepiiriä koskien ovat esimerkiksi "Workflows for Creating 3D Game Characters" (Terävä, T. 2017.), "Videopelihahmon mallintaminen ja teksturointi nykymenetelmin" (Hekkala, E. 2017.) ja "3D-hahmojen toteutus mobiilipeliin" (Kemppainen, M. 2012.).

2 MOBIILIPELIGRAFIIKKA

2.1 Mobiilipelien historia ja tulevaisuus

2.1.1 Lyhyesti mobiilipeleistä

Mobiilipeli on videopeli, jota pelataan nimensä mukaisesti kannettavalla laitteella eli esimerkiksi matkapuhelimella, älypuhelimella, tablet-laitteella tai muulla vastaavalla. Mobiilipelit käsitteenä kattavat alkeelliset pelit kuten matopeli Snake sekä uudemmat pelit, joissa voidaan käyttää edistynyttä teknologiaa kuten 3D-grafiikkaa. (Technopedia 2018.)

Koska mobiililaitteiden resurssit ovat rajoittuneet, mobiilipelien ominaisuudet jäävät yleisesti ottaen pelkistetyimmiksi kuin PC- tai konsolipelien. Mobiilipelinkehitykseen liittyviä haasteita ovat esimerkiksi pelin kontrollien toteuttaminen kosketusnäytölle sekä laitteiden huono prosessointivoima. (Technopedia 2018.)

Kirjoitushetkellä suurin osa mobiilipeleistä ladataan tärkeimpien älypuhelinvalmistajien eli Android ja iOS-käyttöjärjestelmien sovelluskaupoista. Ilmaispelit ovat tällä hetkellä tuottavimpia, mutta myös niiden kehittäjät kohtaavat taloudellisia haasteita - vuonna 2014 julkaistun tutkimuksen mukaan 66 % uusista pelaajista poisti lataamansa ilmaispelin ensimmäisen vuorokauden aikana ja vain 2,2 % uusista pelaajista suoritti pelinsisäisiä ostoksia ensimmäisen 90 päivän aikana. (Bautista, C. B. 2014.)

2.1.2 Mobiilipelien historia

Ensimmäinen mobiilipeli oli Tetris, joka julkaistiin vuonna 1994 Hagenuk MT-2000 -mobiilipuhelimelle, mutta mobiilipelien todellinen läpimurto koitti vasta 1997, kun matopeli Snake julkaistiin Nokia 6610 -alustalle. Snake jäi historiaan yhtenä kaikkien aikojen pelatuimmista videopeleistä ja se löytyy yli 350 miljoonasta laitteesta maailmassa. Kuva Snaken jatko-osasta Snake II kuvassa 1. (Wang, Y. 2018.)

Kuva 1. Historiallinen matopeli "Snake II". (Wang, Y. 2018.)

Mobiilipelien historiassa mainitsemisen arvoinen askel oli Nokian N-Gage hybridimatkapuhelin (kuva 2), joka julkaistiin vuonna 2003. N-Gage oli tarkoitettu sekä puhelimeksi että pelikonsoliksi, tarkoituksenaan houkutella pelaajia muun muassa Game Boy -alustalta. Vaikka jälkikäteen ajatellen Nokia oli aikaansa edellä, N-Gage ei ollut suuri menestys aikanaan muun muassa kontrollien jähmeydestä johtuen. (Martin, M. 2016; Wikipedia 2018.)

Kuva 2. Nokia N-Gage -puhelin, joka soveltui myös pelien pelaamiseen. (Amos, E. 2014.)

Kun Apple julkaisi ensimmäisen iPhonen vuonna 2007, alkoi mobiilipelien uusi aikakausi. Älypuhelimille oli mahdollista kehittää aikaisempaa hienompia ja monimutkaisempia grafiikoita. iPhonen julkaisua seurasi liuta menestyneitä mobiilipelejä, kuten Angry Birds, joka oli ensimmäinen kaupallisesti menestynyt mobiilipeli, sekä Plants vs. Zombies, Cut the Rope, Temple Run, Candy Crush, Angry Birds ja monta muuta. (Wang, Y. 2018.)

2.1.3 Mobiilipelien tulevaisuus

Kun puhutaan videopeleistä, mobiilipelit eivät välttämättä tule ensimmäisenä mieleen. Mobiililaitteita ei tulisi kuitenkaan väheksyä pelialustana, sillä ne tavoittavat vuosi vuodelta enemmän ihmisiä. Esimerkiksi yritys Newzoo arvioi älypuhelinkäyttäjen määrän kasvavan vuonna 2020 jo 3,6 miljardiin. Newzoon markkinatutkimuksessa arvioidaan myös, että mobiilisovellusten liikevaihto, josta mobiilipelien osuuden arveltiin olevan 76 %, kasvaa 38,6 miljardista dollarista 46,2 miljardiin dollariin vuosina 2016-2020. Tutkimus pohjautunee siihen, että sekä vuosina 2016 että 2017 mobiilipelien tuotto kasvoi noin 20 %. Suurin osa kasvusta tapahtui Kiinassa. Mobiilipelaajat ovat isompi kohderyhmä kuin konsoli- ja PC-pelaajat, mikä motivoi kehittäjiä ja sijoittajia ajamaan alaa eteenpäin. Esimerkiksi tunnettu japanilainen videopelikehittäjä Konami on päättänyt keskittyä tekemään pelejä pääasiallisesti mobiilialustoille tulevaisuudessa. (Diver, M. 2015; Cowley, R. 2017; Cross, K. 2017; IMGA, 2017; Takahashi, D. 2017.)

Mitkä trendit määrittelevät mobiilipelien tulevaisuuden? Isot ilmaispelimarkkinat kannustavat tulevaisuuden mobiilipelinkehittäjiä ottamaan mainonnan ja markkinoinnin tosissaan. Mobiilipeleillä ei ole vielä omaa superbrändiä, kuten Mario, joten ehkä pelinkehittäjien voidaan odottaa panostavan brändäykseen tulevaisuudessa. (Cowley, R. 2017; Cross, K. 2017; IMGA, 2017.)

Tulevaisuudessa saatetaan nähdä myös enemmän valmiiksi hinnoiteltuja pelejä riippuen suunnasta, jonka yhteiskunnallinen keskustelu ottaa. Tällä hetkellä mobiilipeleistä suuri osa on "ilmaispelejä" eli ne voidaan ladata ilmaiseksi. Ala on jatkuvassa muutoksessa pelinkehittäjien yrittäessä tasapainottaa ilmaispelien pelinsisäisten ostosten hinnoittelua. Euroopassa ovat kuohuttaneet lähihistoriassa niin kutsutut "gacha"-palkinnot, jotka muistuttavat luonteeltaan uhkapelaamista. (Cowley, R. 2017; Cross, K. 2017; IMGA, 2017.)

Teknologia ja pelit keskittyvät nykyään sosiaaliseen kanssakäymiseen, joka on yksi mobiilipelien vahvuuksista niiden moninpeli- ja verkostoitumismahdollisuuksien vuoksi. Nykynuoret kasvavat internetin ja pikaviestien maailmassa ja tulevaisuudessa mobiilipelinkehittäjien kannattanee ottaa huomioon peliskenen kilpailullinen ja sosiaalinen luonne. Elektroninen urheilu on tällä hetkellä keskittynyt miltei täysin PC- ja konsolipelimarkkinoihin, mutta pelit, kuten Vainglory (kuva 3) ja Clash Royale uurtavat jo uraa mobiilipeli-urheilulle (Cowley, R. 2017; Cross, K. 2017; IMGA, 2017.)

Kuva 3. Moderni mobiilipeli Vainglory 5v5 on visuaalisesti vakuuttava. (Super Evil Megacorp, 2018.)

Nykypäivänä mobiilipelejä pidättelee mobiililaitteiden rajoittunut suorituskyky, mutta asiaan voidaan ehkä odottaa muutosta, kun esimerkiksi pelaajille suunnatuista tietokoneista ja lisävarusteista tunnettu yhtiö Razer osoitti kiinnostusta mobiilipelaukseen julkaistessaan marraskuussa 2017 ensimmäisen Razer-älypuhelimen. Puhelin on teknisesti optimoitu pelaamiseen näyttönsä, äänentoistonsa ja tehokkuutensa säädeltävyyden puolesta. Siinä on myös tehokas uusi jäähdytysratkaisu, joka mahdollistaa paremman suorituskyvyn: vaikka tavallisessa nykyhetken älypuhelimessa riittäisi prosessointitehoja, ilo on usein lyhytkestoinen puhelimen ylikuumentuessa. Tavallisillekin mobiilialustoille on jo nyt kehitteillä tai kehitetty VR- ja AR-pelejä, joista varsinkin AR-pelit kääntävät mobiilin heikkouden pelialustana sen vahvuudeksi hyödyntämällä laitteen ominaisuuksia (kameraa ja GPS-lähetintä) pelimekaniikkoina. (Cowley, R. 2017; Cross, K. 2017; IMGA, 2017; Lehtiniitty, M. 2017; Razer, 2018; Takahashi, D. 2017.)

Lyhyesti sanottuna mobiilipeliala on nyt erinomaisessa asemassa luodakseen uusia historiallisia brändejä sekä haastaakseen perinteisesti suuremmat pelialustat.

2.2 2D- ja 3D-grafiikka mobiilipeleissä

Termeillä 2D ja 3D viitataan ulottuvuuksiin. Ulottuvuus tarkoittaa "mitä tahansa mitattavaa tilallista alaa"; esimerkiksi pituus, leveys, syvyys tai paksuus. Käytännössä ulottuvuus viittaa objektin sivuihin, kuten pituuteen ja leveyteen. (Difference Between, 2017.)

Termi 2D on lyhenne kaksiulotteisesta ja 3D vastaavasti kolmiulotteisesta. 2D-grafiikassa objekti esitetään kahden ulottuvuuden suhteen ja 3D-grafiikassa objekti esitetään kolmen ulottuvuuden suhteen. Havainnollistavaksi esimerkiksi voi kuvitella vaikkapa kissan. Kissalla on kolme ulottuvuutta, kuten kaikella meidän todellisuudessamme: sillä on pituus, leveys ja syvyys. Piirustuksella kissasta on kuitenkin vain kaksi ulottuvuutta, pituus ja leveys, sillä piirustuksen syvyyttä ei voi mitata. (Difference Between, 2017.)

Matematiikassa ja fysiikassa 2D-tilassa on kaksi akselia, X-akseli ja Y-akseli. 3D-tilassa akseleita tulee yksi lisää, Z-akseli. Sama pitää paikkansa videopeligrafiikan suhteen. (Difference Between, 2017.)

2.2.1 2D-grafiikka

2D-grafiikka on tietokonegrafiikkaa, jolla on kaksi tilaulottuvuutta (X- ja Y-akselit). Tarkemmin termillä 2D-grafiikka tarkoitetaan kaksiulotteisia digitaalisia kuvia ja niiden tuottamiseen tarkoitettuja tekniikoita. 2D-tietokonegrafiikka sai alkunsa 1950-luvulla vektorigrafiikkalaitteista, jotka bittigrafiikkalaitteet laajalti syrjäyttivät sitä seuraavina vuosikymmeninä. Bittikarttakuvat- eli pikselikuvat ovat edelleen yleisin digitaalinen kuvamuoto. (Wikipedia 2018.)

2D-peleissä kuvat voidaan yleisesti ottaen jaotella spriteihin eli kuviin, joita liikutellaan muun grafiikan päällä, sekä tile-laattoihin eli suorakulmaisiin taustakuviin. Tällaisilla asseteilla voidaan rakentaa suhteellisen tehokkaasti monimutkaisiakin pelimaailmoja. (Google Play, 2018; Sanakirja.org, 2018; Wikipedia, 2018.)

Itse 2D-pelit voidaan jakaa kolmeen kategoriaan lähinnä kamerakulmansa perusteella. Niin kutsutuissa top-down-peleissä eli yläpuolelta kuvatuissa peleissä kamera näyttää pelaajan ja alueen heidän ympärillään lintuperspektiivistä. Se oli aikoinaan yleinen kuvakulma esimerkiksi 2D-roolipelivideopeleissä, rakennus- ja simulaatiopeleissä kuten

SimCity sekä toiminta- ja seikkailupeleissä kuten The Legend of Zelda. (Giant Bomb, 2018; Wikipedia, 2018.)

Side-scrolling-videopeleissä kamerakulma on ikään kuin sivussa ja ruudulla olevat hahmot kulkevat yleensä vasemmalta oikealle. Side-scrolling-pelit hyödyntävät joskus parallaksivieritystä luodakseen illuusion maailman 3D-syvyydestä. (Webopedia, 2018; Wikipedia, 2018.)

2.5D-peli, 3/4-perspektiivistä kuvattu peli ja pseudo-3D-peli viittaavat kaikki viimeiseen kategoriaan kuuluviin peleihin, eli sellaisiin 2D-peleihin, jotka yrittävät väärentää 3D-kuvakulmaa 2D-grafiikan keinoin. Aiheesta puhutaan enemmän luvussa "2.2.3. 2.5D-grafiikka". (Wikipedia, 2018.)

2.2.2 3D-grafiikka

3D-grafiikka on tietokonegrafiikkaa, joka on mallinnettu kolmen tilaulottuvuuden suhteen (X-, Y- ja Z-akselit). Tyypillisesti 3D-grafiikka esitetään kaksiulotteiselle kuvapinnalle projisoituna. 3D-grafiikka on yleensä vektorigrafiikkaa, jonka peruselementti on kolmio tai muu monikulmio. (Wikipedia 2018.)

3D-grafiikkaa sekoitetaan usein käsitteenä 3D-mallin kanssa. 3D-malli tarkoittaa kolmiulotteisen objektin matemaattista esitysmuotoa eli se sisältää tiedot esimerkiksi 3D-objektin geometriasta ja tekstuurista, eikä ole varsinaisesti grafiikkaa ennen kuin se tuodaan visuaalisesti esille. 2D-kuvan generoimista 3D-mallista tietokoneohjelmalla kutsutaan renderöinniksi. (Wikipedia 2018.)

2.2.3 2.5D-grafiikka

Kuten luvussa "2.2.1 2D-grafiikka" mainittiin, on myös pelejä, joiden grafiikkaa voidaan kuvata parhaiten termillä 2.5D-grafiikka eli kaksi- ja puoliulotteinen grafiikka. 2.5D-grafiikka on 2D-grafiikkaa, joka simuloi 3D-grafiikan ulkoasua esimerkiksi isometrisellä kuvakulmalla tai muilla keinoilla. 2.5D-grafiikka oli eräänlainen edeltävä askel 3D-grafiikalle. (Just Total Tech, 2014; Wikipedia, 2018.)

Nykypäivänä termiä 2.5D-grafiikka käytetään yleensä, kun puhutaan peleistä, jotka yhdistävät kolmiulotteiset polygonigrafiikat lukittuun 2D-perspektiiviin (kuva 4). Pelimeka-

niikat tapahtuvat kaksiulotteisella tasolla ja pelaaja voi harvoin hallita kameraa tai hahmon liikkumista 3D-tilassa. Vanhempi, joskin validi, käyttötarkoitus termille on kuvata liutaa pseudo-3D-grafiikkatekniikoita, joilla yritetään luoda kolmiulotteinen maailma käyttämättä 3D-polygoneja suorituskykyongelmien vuoksi: esimerkiksi vuonna 1996 julkaistussa pelissä Duke Nukem 3D (kuva 5) emuloidaan 3D-maailman syvyyttä käyttämällä vain 2D-assetteja. Myös pelit, jotka käyttävät aksonometristä projektiota tai isometristä kuvakulmaa, voidaan luokitella 2.5D-peleiksi. (Nimimerkit Galamoth IcyEyes, Jagged85, Vigorousjammer, 2017; Wikipedia, 2018.)

Kuva 4. Supercellin moderni 2.5D-peli Clash of Clans (Supercell, 2018.)

Kuva 5. Ruutukaappaus 2.5D-pelistä Duke Nukem 3D, joka julkaistiin vuonna 1996. (DOS Games Archive, 2018.)

2.2.4 2D- ja 3D-grafiikan hyödyt ja haitat pelinkehittäjälle

2D- ja 3D-grafiikkaa on hankala verrata toisiinsa, sillä ne ovat kaksi erilaista kokemusta ja molempia tyylilajeja voi käyttää pelinkehityksessä onnistuneesti. 2D- ja 3D-grafiikan hyviä ja huonoja puolia voi kuitenkin rationalisoida punnitsemalla niiden eri ominaisuuksia esimerkiksi peliprojektin lajityylin, kohdealustan ja resurssien kannalta. (Just Total Tech, 2014; Red Apple, 2017.)

Yleisesti sanottuna 3D-pelinkehitys vaatii enemmän tietotaitoa artisteilta, sillä 3D-työnkulun oppimiskurvi on jyrkkä ja 3D-assetit ovat luonteeltaan monimutkaisempia kuin 2D-assetit. 3D-assettien työnkulussa työvaiheita on useampi, jolloin riski törmätä ongelmiin kasvaa. 3D-grafiikan tekniset rajoitukset ovat myös haastavammat testata kuin 2D-grafiikan: 2D-grafiikassa 2D-kuvatiedostojen maksimikoon ja -määrän tunteminen antaa hyvät valmiudet performanssiongelmien välttämiseen, mutta 3D-skenaariossa on enemmän muuttujia. On otettava huomioon tekstuurikarttojen maksimikoko, eri tekstuurikarttatyypit, maksimiverteksimäärä, valaistus, piirtopyynnöt, päällepiirtäminen, avain-

kehysten ja luiden maksimimäärä ja niin edelleen. On myös harvinaista, että yksi graafikko olisi hyvä esimerkiksi sekä konseptoinnissa, mallintamisessa että valaistuksessa, joten 3D-grafiikan tuottamiseen tarvitsee usein useamman artistin. (Nimimerkit Brian Ortiz, Katana314, munificent, M2tM, yusef ghatavi, 2010-2016; Naser, A. 2015; Walters, A. 2014.)

3D-grafiikalla voidaan kuitenkin luoda entistä realistisempia ja immersiivisempiä pelimaailmoja ja 3D-assettien uudelleenkäyttö on helppoa. Esimerkiksi 3D-animointiprosessi voi joskus olla nopeampi ja helpommin iteroitava kuin perinteinen 2D-animointiprosessi. Moderneilla shadereilla on jopa mahdollista luoda 2D-grafiikan näköistä 3D-grafiikkaa ja kerran jo mallinnettua 3D-assettia voi hyödyntää monesta kuvakulmasta. On myös harhaluulo, että 2D-pelit veisivät aina vähemmän tilaa kuin 3D-pelit, sillä 2D-peleissä on paljon kuvatiedostoja, jotka ovat todellisia tilasyöppöjä. (Nimimerkit Brian Ortiz, Katana314, munificent, M2tM, yusef ghatavi, 2010-2016; Naser, A. 2015; Walters, A. 2014.)

Pelilajityylille saattaa olla vakiintunut käyttää joko 2D- tai 3D-grafiikkaa. Muun muassa korttipelit toteutetaan usein 2D-grafiikalla. Käytännöt voidaan kuitenkin rikkoa onnistuneesti, sillä esimerkiksi Hearthstone on menestynyt 3D-korttipeli, jonka alkuvaiheen 2D-designia sen kehittäjät ovat myöhemmin kuvailleet "kauheaksi, litteäksi pergamenttidesigniksi” ("really awful, flat parchment design"). Kuva 6 esittää, miten erilaiselta Hearthstone näytti kehitysvaiheessaan verrattuna sen lopulliseen ulkoasuun. Hearthstonen kehitystiimi päätti suhteellisen aikaisin, että pelin tulisi käyttää 3D-grafiikkaa, jotta se tuntuisi fyysisemmältä. 3D-korttien haluttiin näyttävän oikeilta materiaaleilta, kuten metallilta ja nahalta, sillä se sai ne tuntumaan arvokkaammilta ja todellisemmilta pelaajille. 3D-korttipakkojen avaaminen jäljittelee oikean korttipakan avaamista, jolloin pelaajista tuntuu, että heidän peliin käyttämänsä aika palkitaan tuntuvammin. (Sakamoto, D. 2015; Hearthstone Wiki, 2015-2017.)

Kuva 6. Hearthstone-korttipelin alkuperäinen visuaalinen ilme, jota kehittäjät myöhemmin kuvailivat "litteäksi pergamenttiroskaksi", sekä pelin lopullinen, 3D-grafiikkaa hyödyntävä ulkoasu. (Sakamoto, D. 2015; Hearthstone Wiki, 2015-2017.)

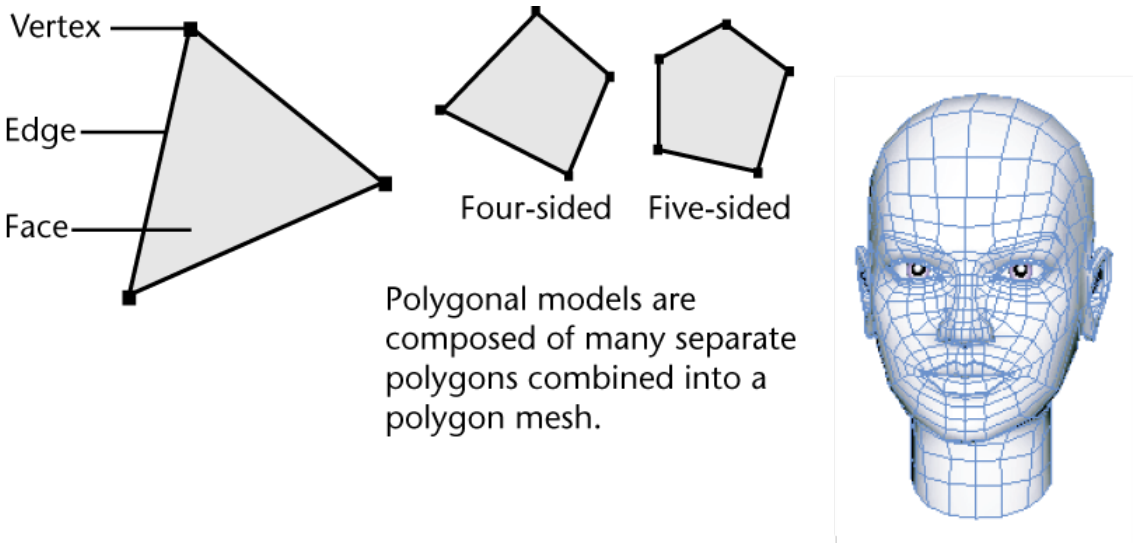
3 MALLINNUKSEN TYÖVAIHEET

Videopelimallin luomisprosessiin kuuluu monta työvaihetta projektista riippuen. Hahmo tai muu objekti konseptoidaan yleensä ensin 2D-muodossa tai sitä varten etsitään mallikuvia. Sitten se mallinnetaan erilaisilla mallinnustekniikoilla, kuten digitaalisesti veistäen ja perinteisesti polygonimallintaen. Malli unwrapataan, jotta se voidaan teksturoida ja mikäli mallista on olemassa korkearesoluutioinen versio, osa tekstuureista voidaan beikata. Jotkin assetit, kuten hahmot, rigataan ja animoidaan. Lopuksi videopelimalli tuodaan 3D-ohjelmasta pelimoottoriin. (Pettit, N. 2015.)

3.1 Perinteinen polygonimallinnus

3.1.1 3D-mallin ominaisuuksista

Polygonit eli monikulmiot ovat geometriaa, joka koostuu vertekseistä, reunoista ja sivuista (kuva 7). Polygoneilla voidaan rakentaa erilaisia 3D-malleja ja niitä käytetään 3D-sisällön tuottamiseen esimerkiksi videopeleihin, elokuviin ja internetiin. Selvästi suurin osa 3D-malleista nykyään on polygonimalleja, sillä ne ovat joustavia ja tietokoneet pystyvät renderöimään ne nopeasti. (Autodesk, 2016; Wikipedia, 2018.)

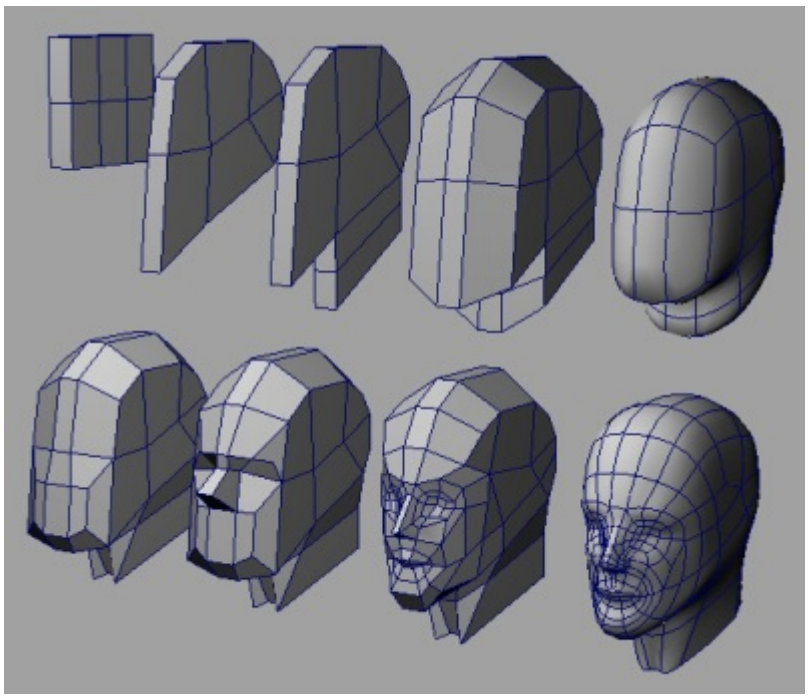


Kuva 7. Polygonimalli ja polygoneja. Polygonit koostuvat vertekseistä, reunoista ja sivuista. (Autodesk, 2016.)

Polygonimallinnuksessa käytetään yleensä kolmisivuisia polygoneja eli kolmioita tai nelisivuisia polygoneja eli nelikulmioita, mutta polygoneilla voi olla useampikin sivu. Nelisivuiset polygonit aiheuttavat vähiten ongelmia animoinnissa ja varjostuksessa ja niiden reunaluuppien virtaus tekee mallintamisesta sulavaa. (Autodesk, 2016.)

3.1.2 Perinteinen mallinnusprosessi

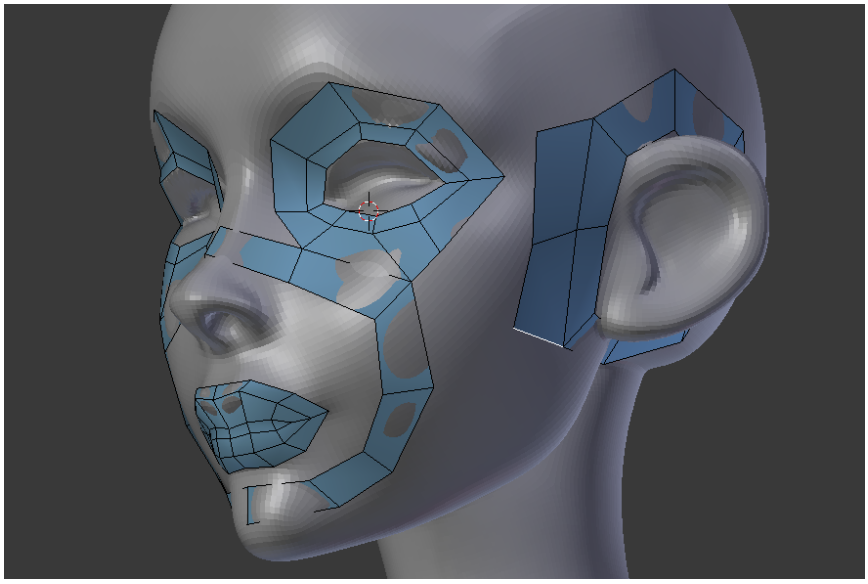
Polygonisen mallin rakentamista voi lähestyä monella eri tavalla. Yleinen perinteinen tekniikka on laatikkomallinnus, jolloin artisti aloittaa yksinkertaisesta geometrisesta muodosta kuten laatikosta tai pallosta ja muovailee sitä, kunnes se on halutunlainen. Laatikkomallinnusprosessi aloitetaan yleensä matalaresoluutioisesta mallista, jonka pinnan geometria voidaan myöhemmin jakaa, mikäli mallin muotoa halutaan pyöristää tai tarvitaan suurempi resoluutio yksityiskohtien lisäämistä varten (kuva 8). (Nimimerkki GiantCowFilms, 2015; Slick, J. 2017; Wikipedia, 2018.)



Kuva 8. Ihmispään mallintaminen laatikosta. (Don College Design Wiki, 2016.)

Laatikkomallintaminen vaatii mallintajalta etukäteen suunnittelua ja kokemusta mallinnuksesta, mutta siltikin joitain muotoja saattaa olla hankala toteuttaa. Eri mallinnustekniikoita käytetäänkin usein yhdessä ja laatikkomallinnuksen työnkulkua voi täydentää esimerkiksi ekstuusiomallintamalla (kuva 9). Ekstruusiomallinnuksessa malli rakenne-

taan pala palalta sijoittamalla polygonisivuja mallin ulkonevien pintojen mukaisesti (Slick, J. 2017.)



Kuva 9. Mallin kasvojen topologiaa on aloitettu uudelleenrakentamaan reunaluupeilla ekstruusiomallintaen (siniset polygonit kuvassa).

3.1.3 Ohjelmisto

3D-ohjelmiston alan jättiläiset ovat tällä hetkellä 3Ds Max, Maya ja Blender. Ohjelman nimeä tärkeämpää on kuitenkin mallintajan ammattitaito.

Autodeskin 3ds Max on perinteisesti ollut alan standardi, mutta muut sovellukset ovat alkaneet kuroa väliä kiinni. 3ds Max tarjoaa käyttäjälleen voimakkaat mallinnustyökalut ja erinomaiset UV-muokkaustyökalut. 3Ds Max on kuitenkin kallis, noin 2 000 euroa vuodessa, ja se on saatavissa ainoastaan Windows-käyttöjärjestelmälle. (Autodesk, 2018; Masters, M. 2015.)

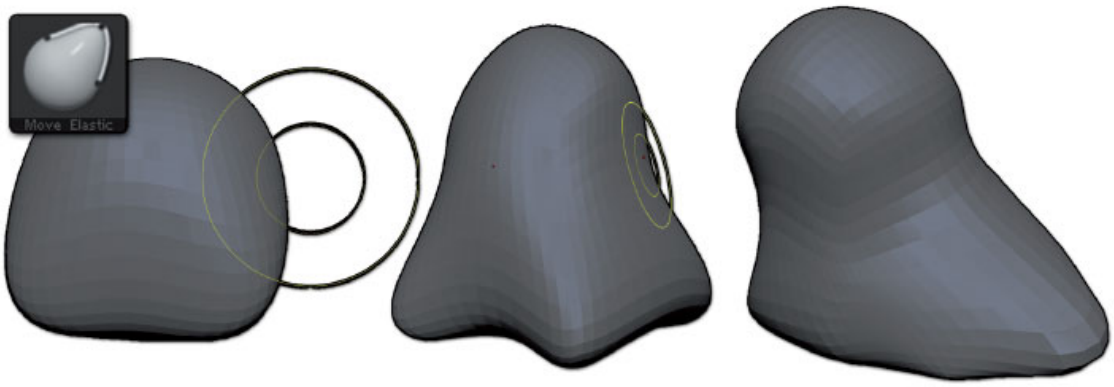
Maya, joka on myös Autodeskin sovellus, on toinen pelialan yleisimmin käytetyistä mallinnussovelluksista. Mayasta on olemassa kevyempi versio Maya LT, joka on suunniteltu erityisesti pelinkehitykseen. Siinä on vähemmän ominaisuuksia kuin alkuperäisessä Mayassa, mutta se on huomattavasti koko versiota halvempi. Mayan mallinnusominaisuudet eivät yllä täysin 3ds Maxin tasolle, mutta se loistaa animoinnissa ja riggauksessa ja siinä on voimakkaat skriptaustyökalut. (Autodesk, 2018; Masters, M. 2015.)

Blender on alalla ja varsinkin isoissa pelistudioissa vähemmän käytetty kuin 3ds Max tai Maya. Blenderin mallinnusmetodit ovat intuitiiviset ja ohjelma nojaa pitkälti pikanäppäimiin. Blender on kerännyt ympärilleen yhteisön Blenderintoilijoita sekä pelinkehittäjiä, sillä Blenderissä on avoin lähdekoodi ja se on täysin ilmainen. (Masters, M. 2015.)

Muita varteenotettavia ohjelmistovaihtoehtoja ovat esimerkiksi Cinema 4D, LightWave 3D ja Modo. (Polycount Wiki, 2017.)

3.2 Digitaalinen veistäminen

Veistäminen on prosessi, jossa materiaalia kuten kiveä tai puuta työstetään, kunnes sen muoto on halutunlainen. Digitaalisellä veistämisellä eli 3D-veistämisellä viitataan samankaltaiseen digitaaliseen prosessiin - 3D-massaa voidaan muovata erilaisiin muotoihin, siihen voidaan lisätä tai siitä voidaan poistaa volyymia ja sen pintaa voidaan muokata hyvinkin yksityiskohtaisesti (kuva 10). (Pixologic, 2018.)



Kuva 10. 3D-mallia voi muokata vetäen ja työntäen 3D-veistämisohjelmassa. (Pixologic, 2018.)

Kuten perinteistäkin veistämisprosessia, digitaalista veistämistä lähestytään usein vaiheittain tai tasoittain. Mallin muokkaaminen aloitetaan isoista muodoista mahdollisimman kevyellä geometrialla ja pinta jaetaan tarvittaessa, kun edetään yhä pienempien yksityiskohtien veistämiseen. Digitaaliseen veistokseen saattaa kulua aikaa puolesta tunnista satoihin tunteihin riippuen projektin monimutkaisuudesta ja artistin taitotasosta. (Heginbotham, C. 2018.)

Ennen kuin digitaalinen veistäminen oli mahdollista, 3D-artistien oli pidettävä aina mielessä geometria ja polygonit. Mallit rakennettiin polygoni kerrallaan ja teksturoitiin lopuksi käsin. Hyvän artistin oli oltava suunnitelmallinen ja matemaattinen, mikä ei aina sovi yhteen taiteen luovan luonteen kanssa. Veistämissovellukset, kuten ZBrush, mahdollistavat intuitiivisemman luomisprosessin, joka muistuttaa enemmän perinteisillä taidevälineillä, kuten savella, veistämistä. (Nimimerkki MarkG, 2009; Heginbotham, C. 2018; Yot, R. 2012; Wikipedia, 2018.)

Veistämällä malleihin on mahdollista lisätä yksityiskohtia, joiden tekeminen olisi vaikeaa tai mahdotonta perinteisiä 3D-mallinnustekniikkoja käyttäen (kuva 11). Perinteinen polygonimallinnus toimii muotoja, viivoja ja vektoripisteitä manipuloimalla ja on luonteeltaan geometristä. Se sopii hyvin kulmikkaiden objektien, kuten tuolien, luomiseen, mutta vapaamuotoisempi veistäminen on erityisen kätevää orgaanisia muotoja mallinnettaessa. Veistäminen ja muut mallinnustekniikat täydentävät toisiaan ja niitä voi käyttää vahvuuksiensa mukaan eri vaiheissa mallinnusprosessia. (Nimimerkki MarkG, 2009; Heginbotham, C. 2018; Wikipedia, 2018.)

Kuva 11. Sheridan Doosen 3D-veistos Blizzard Entertainmentin hahmosta Illidan Stormrage. Esimerkiksi tällainen yksityiskohtien taso olisi vaikeaa tai mahdotonta saavuttaa perinteisillä mallinnusmetodeilla. (Doose, S. 2016.)

Laajalti käsitetään, että ZBrush oli ensimmäinen sovellus, joka nosti digitaalisen veistämisen muiden mallinnusmetodien rinnalle. ZBrush julkaistiin vuonna 1999, mutta sen rinnalle on ilmestynyt lukuisia muita 3D-veistämisohjelmia, kuten Sculptris ja Mudbox. ZBrush hyödyntää sen 2.5D-maalaushistoriasta juontuvaa alkuperäisteknologiaa, pik-

solia, joka mahdollistaa veistämiseen vaadittujen resoluutioiden käsittelemisen tietokoneella. Piksoli on eräänlainen älypikseli: tavalliset 2D-kuvat koostuvat pikseleistä, joihin on tallennetu tieto pikselin sijainnista 2D-tilassa sekä sen väri, mutta piksoliin on tallennettu myös tieto piksolin syvyydestä, suunnasta ja materiaalista. (Creative Bloq Staff, 2014; Pixologic, 2018; Wikipedia, 2018.)

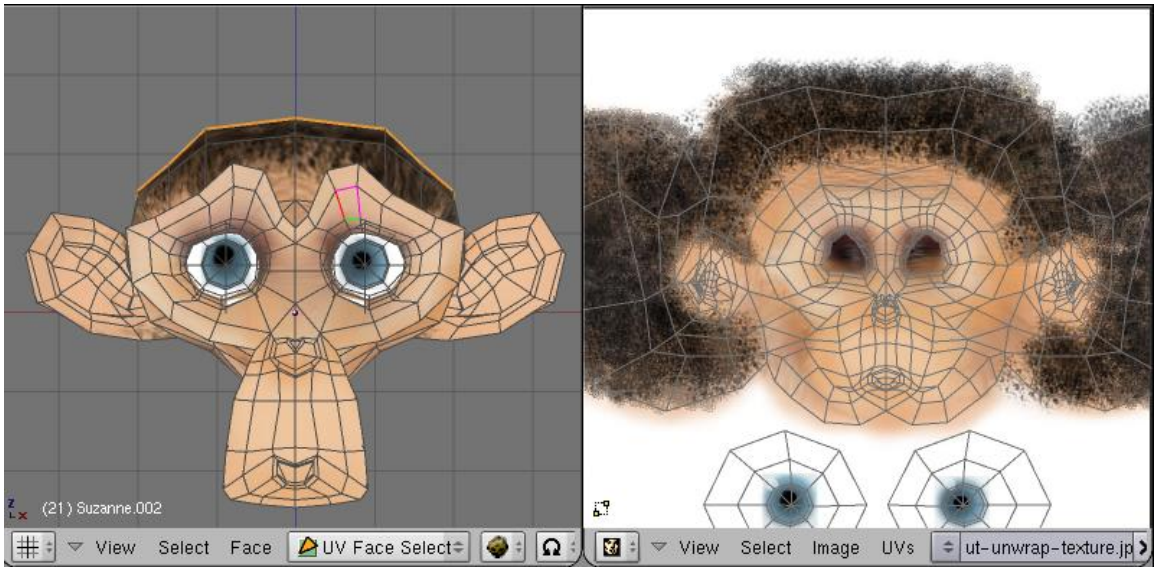
3.3 Teksturointi

Teksturointi tarkoittaa 2D-kuvien tuottamista 3D-malleja varten. Tekstuurikuvia kutsutaan yleisesti tekstuurikartoiksi. Tekstuurikartat, materiaalit ja shaderit vaikuttavat yhdessä siihen, miten malli renderöidään. 3D-malleilla on oltava tieto tekstuurikoordinaateista eli UV-koordinaateista, jotta tekstuuri voidaan "kietoa" mallin pinnalle. Tekstuurikarttoja voidaan tuottaa monella eri metodilla, joihin lukeutuvat perinteinen käsinmaalaaminen, kuvamanipulointi, proseduraalinen generointi noodipohjaisessa materiaalieditorissa ja kuvadatan beikkaus. (Polycount Wiki, 2016.)

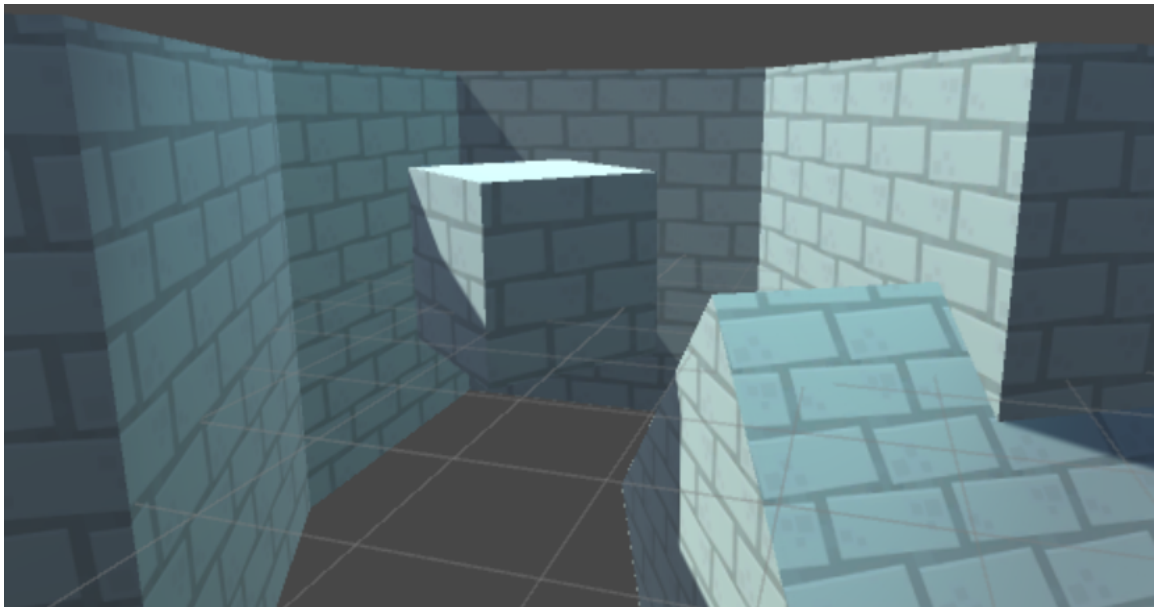
Pelistä riippuen tekstuurikarttoja tarvitaan useita erilaisia opastamaan shaderin eri osa-alueita: esimerkiksi diffuusikartta määrittelee pinnan värin ja metallisuuskartta pinnan metallisuuden. Nykyaikainen peliassetti saattaisi käyttää esimerkiksi diffuusi-, spekulaari- ja normaalikarttoja. (Polycount Wiki, 2016.)

3.3.1 UV-koordinaatit ja -kartat

UV-koordinaatit tarkoittavat u, v -tekstuurikoordinaatteja eli pisteitä, jotka määrittävät mallin verteksien yksittäiset sijainnit suhteessa tekstuurikuvan korkeuteen ja leveyteen (kuva 12). Käytännössä UV-koordinaatit ovat numeropareja tallennettuna 3D-mallin vertekseihin. Koordinaatit sijoittuvat U- ja V-akseleille sekaannuksien välttämiseksi, sillä X-, Y- ja Z-akselit ovat jo käytössä 3D-tilassa. Tekstuurin koordinaatit mitataan skaalalla 0,0 - 1,0 tekstuurikuvan vastakkaisista kulmista. Mikäli UV-arvo on suurempi kuin 1,0, tekstuuri toistaa itseään mallin ympäri (kuva 13). (Van Der Byl, L. 2002; Polycount Wiki, 2016; Wikipedia, 2018.)

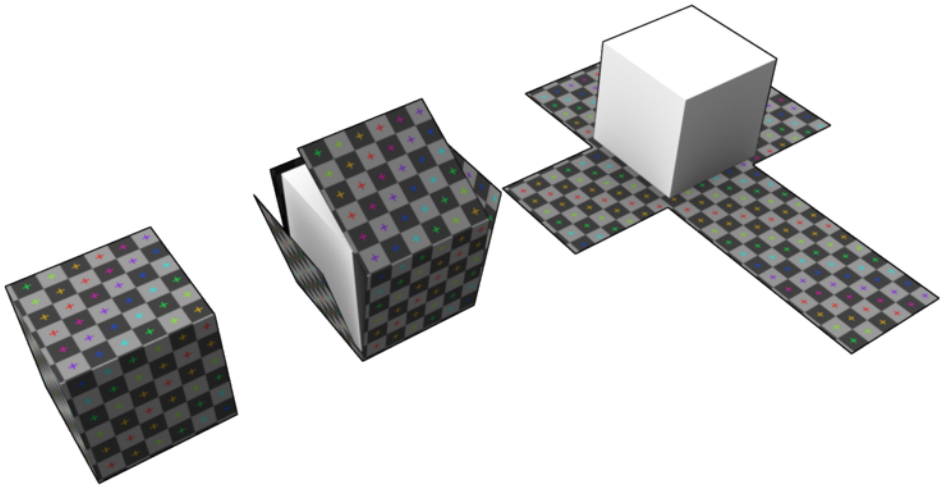


Kuva 12. Pisteet UV-kartalla määrittävät, miten kuva sijoitellaan mallin pinnalle. (Andaur, C. 2004.)



Kuva 13. Esimerkki 3D-mallista, jonka UV-koordinaatit on sijoiteltu niin, että tekstuuri toistaa itseään. (Nimimerkki DMGregory, 2017.)

UV-koordinaatit voi kuvitella virtuaalisiksi nastoiksi tai vaateneuloiksi, jotka pitelevät tekstuuria paikoillaan. Ohjelmisto venyttää kiinnitettyjen pisteiden välisen tilan parhaan kykynsä mukaan mallin pinnalle; prosessi, jota kutsutaan UV-kartoitukseksi. Kolmiulotteisen X-, Y- ja Z-informaation siirtämistä litteälle UV-alustalle kutsutaan unwrappaukseksi. Unwrappaus on jonkin verran abstrakti käsite, mutta se esitetään visuaalisesti havainnollistettuna kuvassa 14. (Van Der Byl, L. 2002; Wikipedia, 2018.)



Kuva 14. 3D-kuution pinta on unwrapattava eli purettava, jotta sen pinnalle voidaan kartoittaa shakkilautatekstuuri. (Wheeler, R. 2008.)

Unwrappaus voidaan suorittaa automaattisesti, mutta yleensä tietokone tarvitsee hieman avustusta mallin pinnan purkamisessa ja etenkin hahmomallit ovat miltei aina tarpeeksi monimutkaisia vaatiakseen manuaalista työtä. Manuaaliseen unwrappaukseen kuuluu saumojen merkitseminen malliin. Virtuaaliset saumat toimivat juuri kuten oikean elämän vaatekappaleiden saumat, eli ne ohjaavat tietokonetta "leikkaamaan" sauman eri osat irti toisistaan UV-kartalle. (Blender Reference Manual, 2018; Sosa, J. 2014; Wikipedia, 2018.)

UV-sauma luo ylittämistään vertekseistä duplikaatit, jotta se voi tallentaa molempien saumasta syntyvien UV-saarten koordinaatit. Pelinsisäinen verteksimäärä vaikuttaa pelin suorituskykyyn. Tämän vuoksi saumoja ei kannata vedellä yltäkylläisesti, ja mikäli UV-sauma sopii samaan kohtaan kuin malliin merkitty terävä reuna, ne kannattaa sijoittaa samoille vertekseille, sillä myös terävät reunat lisäävät tuplaverteksin ja verteksien määrä ei voi kasaantua siitä ylöspäin. (Nimimerkki EarthQuake, 2012.)

Kun UV-saumat on merkitty, mallin voi unwrapata. Unwrapatut UV-saaret, jotka edustavat mallin 3D-pintaa UV-tilassa, sijoitellaan manuaalisesti tai pakkausohjelmalla mahdollisimman optimaalisesti UV-tilaan. Monessa hahmomallissa on osia, jotka ovat symmetrisiä. Tekstuurikartan tilan käytön voi maksimoida, kun symmetristen osien UV-saaret

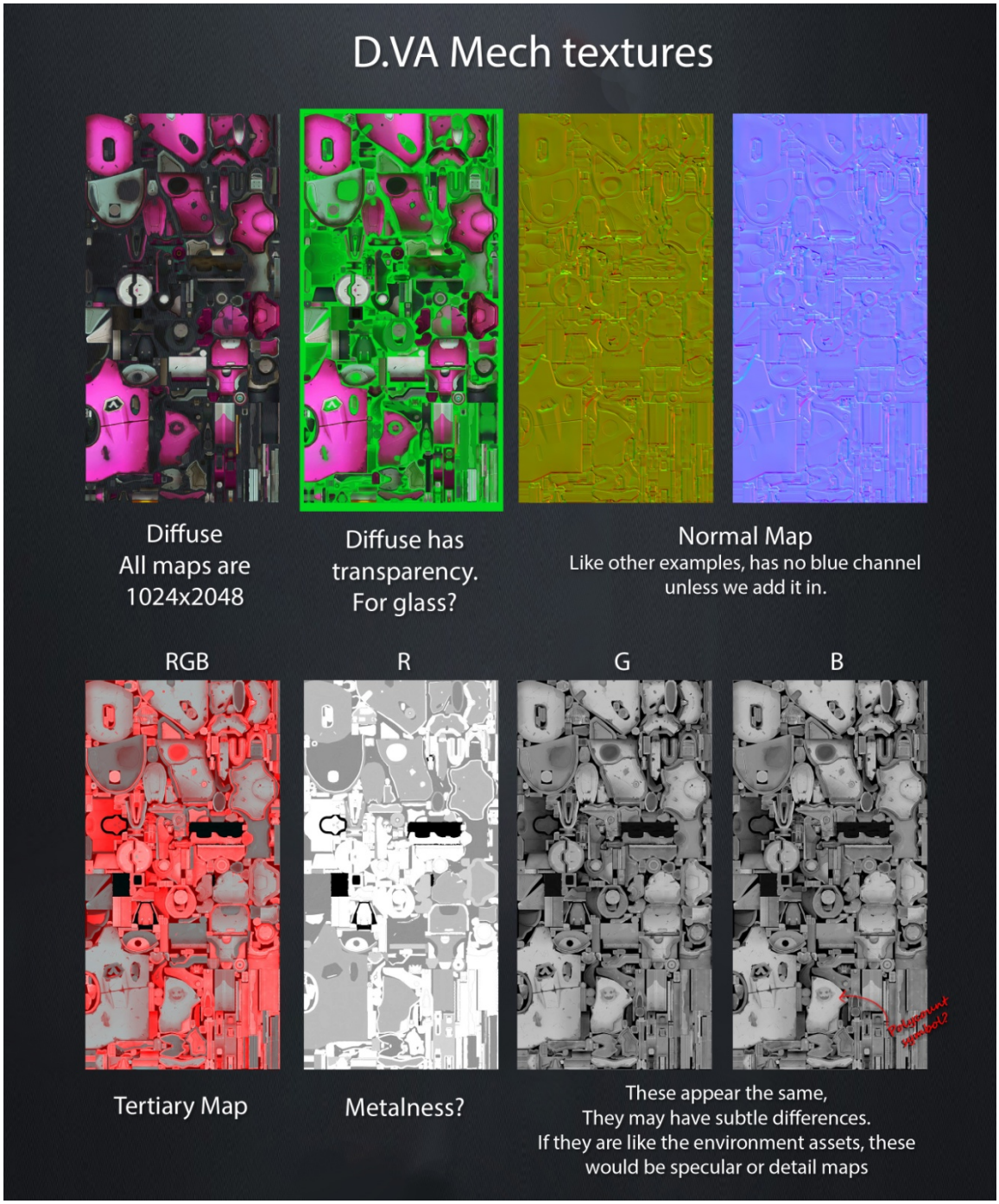
peilataan niin, että ne vievät UV-kartalla vain puolet siitä tilasta, jonka ne peilaamatta veisivät. (Blender Reference Manual, 2018; Sosa, J. 2014; Wikipedia, 2018.)

3.3.2 Tekstuurikartoista yleisesti

Tekstuurikartat ovat kuvia, jotka vaikuttavat mallin pinnan ominaisuuksiin. Ne voidaan jakaa kategorioihin kuten värikartat tai kohokartat. Yleisimmin videopeleissä nykyään käytetään diffuusi-, emissio-, normaali-, opasiteetti-, spekulaari- ja kiiltokarttoja. Erilaiset karttatyypit yhdessä ohjaavat tietokonetta renderöimään lopullisen kuvan (kuva 15). Kartat maskaavat materiaalin ominaisuuksia pikselikohtaisesti, jolloin materiaalin ominaisuuksia voidaan kontrolloida hyvin tarkasti. (Polycount Wiki, 2015.)

Kuva 15. Eri karttatyyppien lisääminen objektiin vaikuttaa sen ulkonäköön. Kuvassa vasemmalta oikealle näkyy tynnyrin topologia eli sen rautalankamalli, materiaaliton tynnyri valaistuna, tynnyrin diffuusikartta, joka syöttää tiedon pinnan väristä, tynnyrin spekulaarikartta, joka syöttää tiedon pinnan spekulaarisuudesta ja tynnyrin normaalikartta, joka kertoo, missä kulmassa valo osuu tynnyriin. (Pettit, N. 2014.)

Mobiilipelinkehittäjän tulisi olla erityisen varovainen joitakin karttoja, kuten normaalikarttoja, käyttäessään. Tämän hetken mobiilipelien grafiikan teknistä kärkeä edustava Shadowgun käyttää normaalikarttoja esimerkiksi vain hahmomalleissa. Hyvä nyrkkisääntö on, että kaikki kartat, jotka vaativat reaaliaikaista simulointia (normaalikartta simuloi reaaliajassa valon mallin pintaan osumista), on kevyempi beikata valmiiksi informaatioksi kuvatiedostoon mobiililaitteen prosessointikyvyn maksimoimiseksi. Mobiilipeleissä voidaan harvoin hyödyntää yhtä monta eri karttaa kuin PC- tai konsolipeleissä. Vertailun vuoksi kuvassa 16 näkyvät kaikki tekstuurikartat, joita PC-pelin Overwatch hahmo D.Va hyödyntää. (Unity, 2018.)

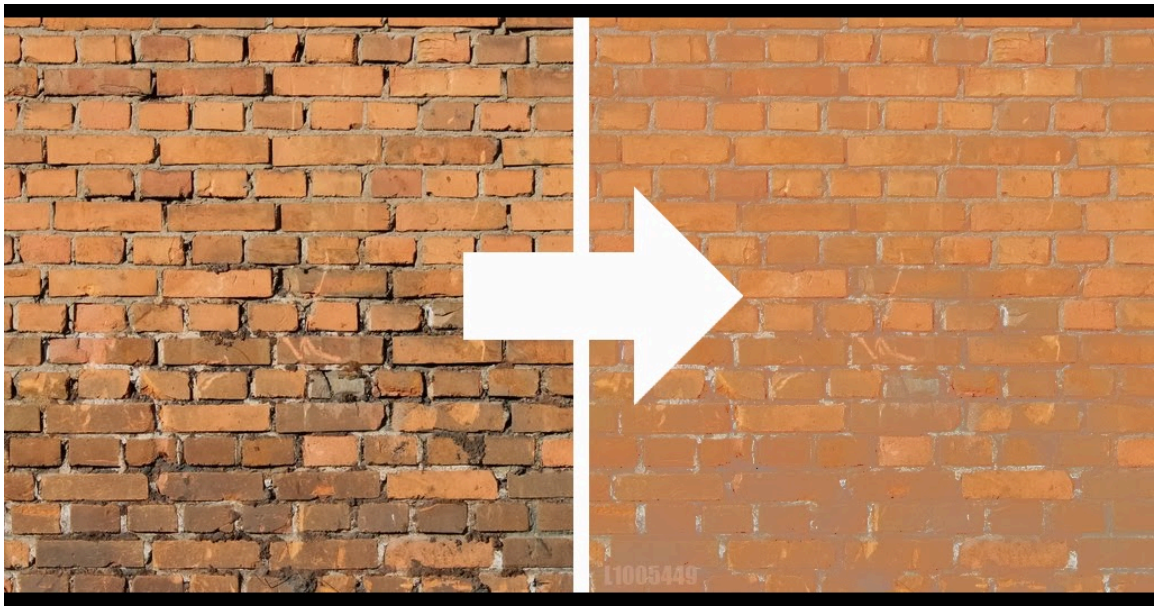


Kuva 16. Kaikki tekstuurikartat, joita PC-pelin Overwatch hahmo D.Va hyödyntää: diffuusi-, läpinäkyvyys-, normaali-, tertiääri-, metallisuus- ja spekulaari- tai detaljikartta. (Nimimerkki Agilethief, 2016.)

3.3.3 Diffuusikartta

Värikartoista kaikista yleisin on diffuusi- eli albedokartta. Diffuusikartta syöttää mallille tiedon pinnan väristä eli se määrittää hajaantuvan valon värin. Mikäli diffuusikartta on ainoa käytössä oleva kartta, kuten mobiilipeleissä usein on, tieto valaistuksesta ja ambientista okkluusiosta voidaan maalata karttaan ja näin simuloida monimutkaisempien shaderien vaikutusta (kuva 17). Nykyaikaisia shadereita käytettäessä diffuusikarttaan voisi taltioida myös vain tiedon pinnan väridatasta (kuva 18), sillä valot ja varjot generoidaan muista karttatyypeistä kuten spekulaarikartasta ja normaalikartasta. (Polycount Wiki, 2012 ja 2015; Wilson, J. 2015.)

Kuva 17. 3D-hahmoista oikeanpuoleinen käyttää vain diffuusikarttaa, mutta hahmo näyttää valaistulta, sillä sen pintaan on "maalattu" tieto valosta ja varjosta. Vasemmanpuoleinen hahmo on oikeasti valaistu ja renderöintiohjelma saa tiedon valon osumisesta mallin pintaan normaalikartasta. Kuvan oikeassa reunassa on hahmon diffuusikartta 2D-muodossa.

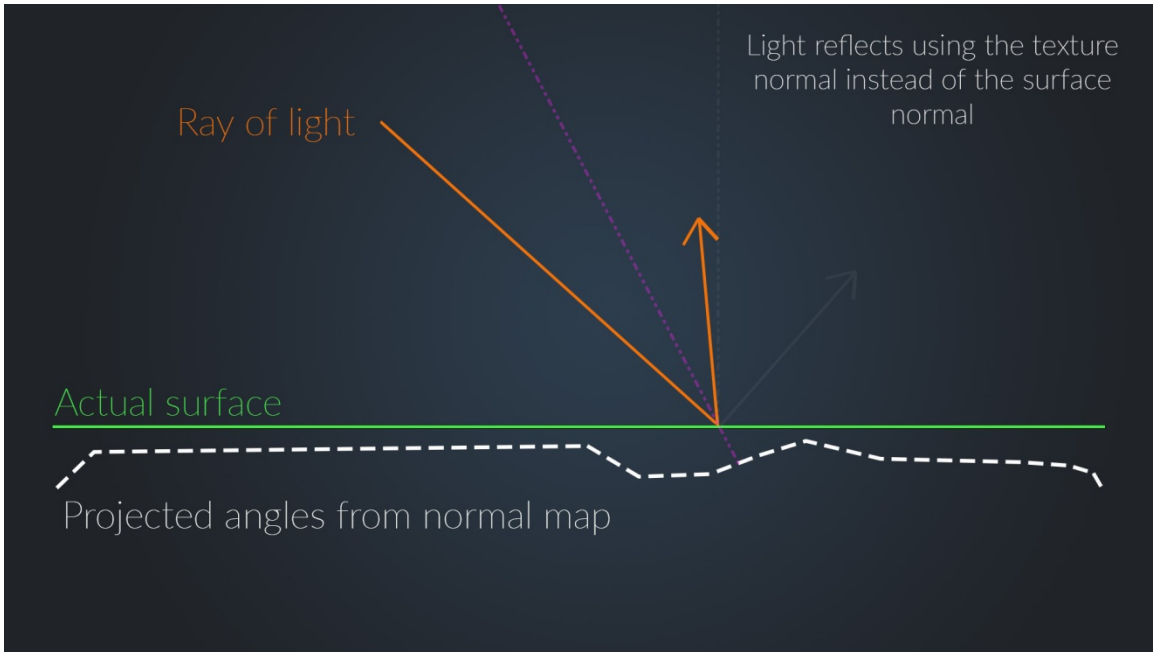


Kuva 18. Tiiliseinä, jossa näkyvät valot ja varjot (vas.) ja tiiliseinä, johon on taltioitu tieto vain ja ainoastaan pinnan väristä. (Du, K. C. C. 2013.)

3.3.4 Normaalikartta

Videopeleissä kuvataajuus (FPS) tarkoittaa näytölle sekunnissa piirrettyjen kuvien määrää. Nykypäivän videopelien kuvataajuuden odotetaan ylittävän reaaliajassa renderöitynä vähintään 60 kuvaa sekunnissa. Tämän vuoksi esimerkiksi 3D-veistettyjä, hyvin raskaita malleja ei voida käyttää peleissä, sillä ne kuluttavat liian suuren osan laitteiston resursseista. Grafiikan suorituskyvyn ei kuitenkaan tarvitse vaarantaa visuaalien laatua, sillä kevytmalleihin voidaan siirtää dataa raskasmallin pinnasta tekstuurikartoilla. (Lampel, J. 2017; OpenGL Tutorials, 2017; Wikipedia, 2018.)

Tekstuurikartat, kuten normaali- ja korkeuskartta, ovat erityislaatuisia kuvatekstuureja, jotka vaikuttavat siihen, miten valon osuu mallin pintaan. Ne luovat illuusion syvyydestä saaden valon kimpoamaan pinnan piirteistä, joita ei todellisuudessa ole (kuvat 19 ja 20). Normaali- ja/tai korkeuskarttoja käytetään kevytmalleissa silloin, kun malliin halutaan lisätä yksityiskohtia, joiden toteuttaminen 3D-geometrialla olisi liian raskasta laitteistolle. Niitä voidaan käyttää myös yksinkertaisen objektin pinnan pyöristämiseksi niin, että valo näyttää osuvan pintaan pehmeämmin ja objekti näyttää sileämmältä. (Lampel, J. 2017; OpenGL Tutorials, 2017; Oshchepkov, A. 2015; Polycount Wiki, 2016.)

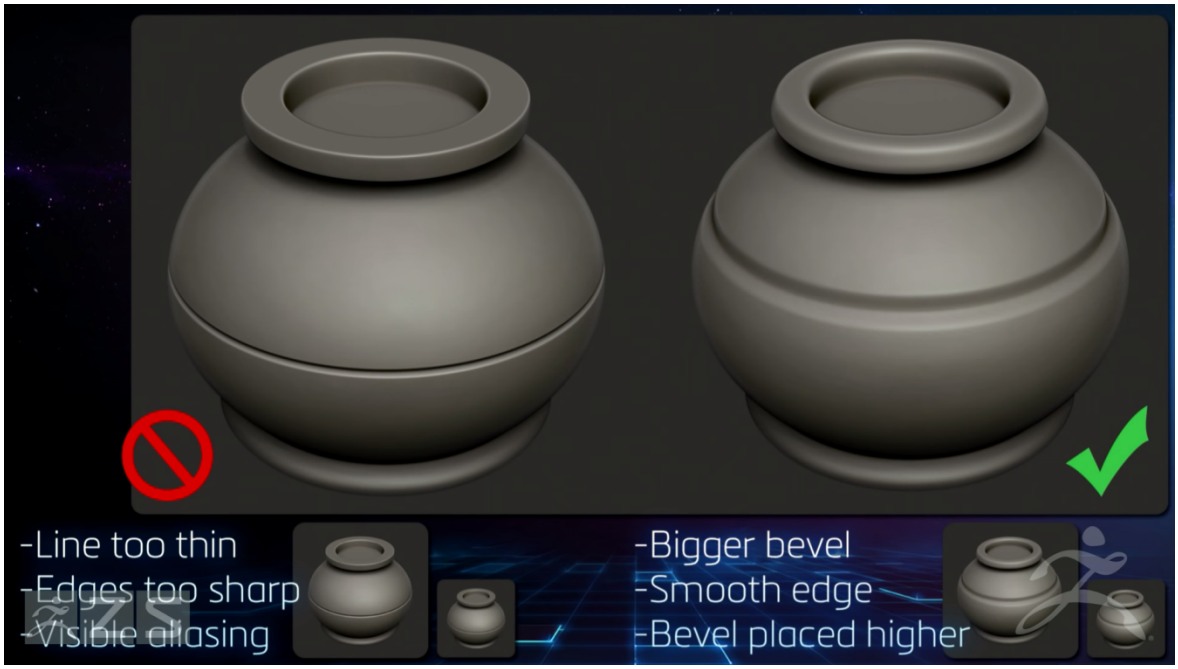


Kuva 19. Normaalikartan vaikutus mallin pintaan. Vihreä viiva edustaa mallin todellista pintaa, valkoinen katkoviiva esittää normaalikarttaan tallennettua tietoa pinnan halutusta kaltevuudesta ja oranssi viiva havainnollistaa, miten valo reagoi mallin pintaan osuessaan. Normaaliarvot määrittävät joka pikselin kirkkauden riippuen mallin todellisen pinnan kulmasta, normaalin kulmasta normaalikartalla ja valonlähteiden sijainnista ja suunnasta. (Lampel, J. 2017; OpenGL Tutorials, 2017.)

Kuva 20. 3D-malli, joka käyttää normaalikarttaa (vas.), sama malli ilman normaalikarttaa (kesk.) ja malli niin, että normaalikarttaa käytetään diffuusikarttana. (Chadwick, E. 2015.)

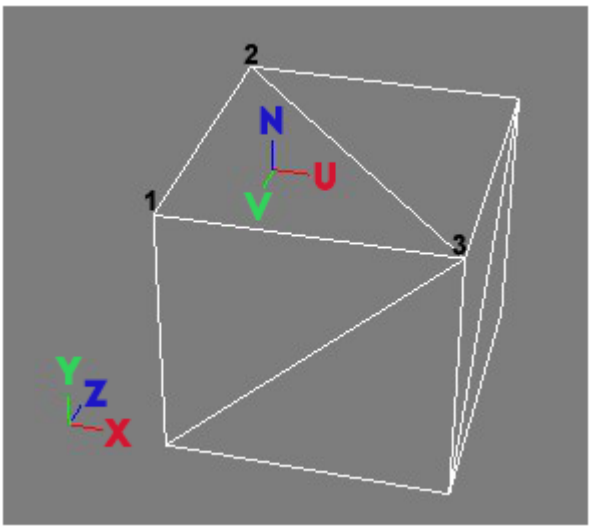
Normaali- ja korkeuskarttojen ilmeisin ero näkyy, kun niitä tarkastelee 2D-tilassa - normaalikartat ovat värikkäitä ja korkeuskartat mustavalkoisia. Korkeuskartat pystyvät kuvastamaan ainoastaan korkeuseroja pinnassa: musta merkitsee syvennystä, valkoinen kohoumaa ja neutraali harmaa muuttumatonta korkeutta. Korkeuskarttoihin nähden normaalikarttojen suurin etu on se, että niillä voidaan määritellä pinnan suunta eli pinnat voivat olla vinoja. Sitä ei voida tehdä pelkästään tiedolla pinnan korkeuseroista, sillä renderöintiohjelma ei osaa tulkita, *mihin suuntaan* pintaa pitäisi taivuttaa. Terävistä reu-

noista voidaan tehdä normaalikartoilla pehmeämmän näköisiä, millä on yllättävän iso vaikutus, sillä oikeassa elämässäkään ei ole olemassa täysin teräviä reunoja. Varsinkin pienellä ruudulla vinoja reunoja kannattaa liioitella, sillä pelaajan on helpompi tunnistaa objektit, kun niiden siluetti on suurpiirteisempi ja reunoista kimpoaa enemmän valoa (kuva 21). Normaalikarttojen jäljittelemät yksityiskohdat ovat feikkejä, sillä pinnan geometriaan ei todellisuudessa lisätä resoluutiota eli normaalikartta ei voi vaikuttaa mallin siluettiin. (Lampel, J. 2017; Oshchepkov, A. 2015; Pluralsight, 2014; Polycount Wiki, 2016.)

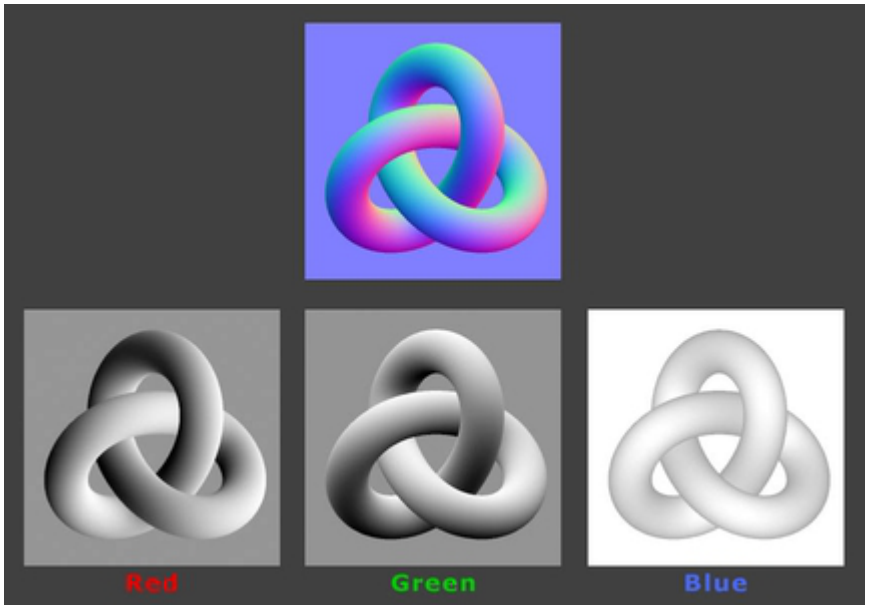


Kuva 21. "Heroes of the Storm"-pelin assetissa on liioiteltu vinoa reunaa niin, että pelaajan on helpompi tunnistaa se ruudulla, vaikka assetti on pelissä pieni ja näkyy kaukaa. (Vicente, M. 2016.)

Moni tuntee normaalikartat omituisista sinipunaisista väreistään, joilla on kuitenkin tarkka tarkoitus. Jokaiseen tekseliin normaalikartassa on tallennettu tieto mallin pinnan halutusta suunnasta: 3D-tilan XYZ-akselit kuvastavat, mihin suuntaan XYZ-arvot kasvavat maailman tilassa ja niitä vastaavat tangenttiavaruudessa UVN-akselit (kuva 22). Normaalikartta käyttää kuvatiedoston RGB-kanavia vastaavien tangenttiavaruuskoordinaattien tallentamiseen eli pinnan normaalin suunnan määrittämiseen (kuva 23). (OpenGL Tutorials, 2017; Oshchepkov, A. 2015; Pluralsight, 2014; Polycount Wiki, 2016.)



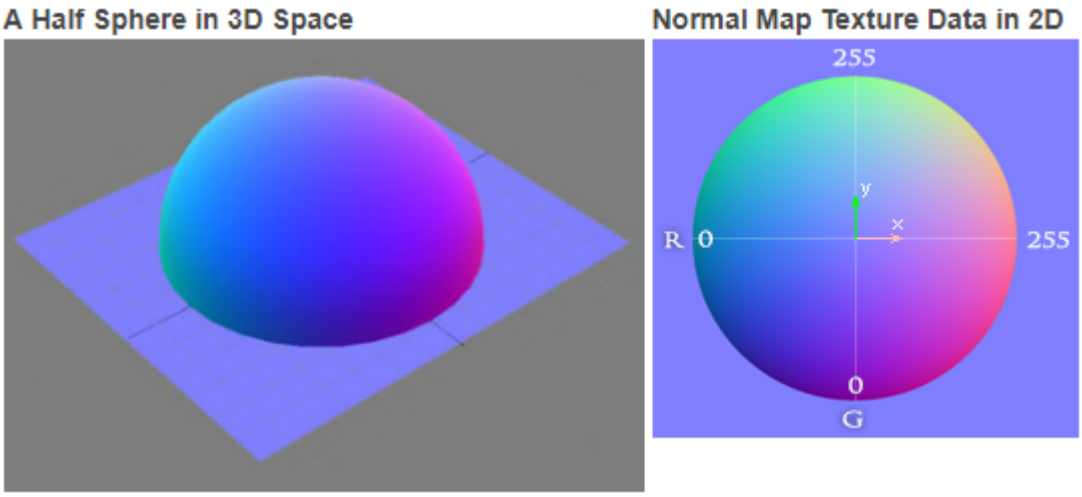
Kuva 22. Akselien visualisointi: 3D-tilan X- ja Y-akselit vastaavat tangenttiavaruuden U- ja V-akseleita ja Z-akseli vastaa N-akselia eli polygonin normaalia. (Oshchepkov, A. 2015.)



Kuva 23. Normaalikartta ja sen punainen, vihreä ja sininen kanava toisistaan erotettuna. (Oshchepkov, A. 2015.)

Kuvassa 24 vasemmanpuoleinen puolipallo on korkearesoluutioinen 3D-objekti, jonka pinta on täysin sileä. Se beikkaamalla saadaan kartoitettua kuvatiedostoon ääritapaukset kaikista mahdollisista R-, G- ja B-arvoista, joita normaalikartassa voidaan käyttää. RGB-kanavista punainen kanava vastaa X-/U-akselia, vihreä kanava vastaa Y-/V-akselia ja sininen kanava vastaa Z-/N-akselia. Kun kuvaan halutaan tallentaa tieto suorasta Z-/N-akselin -mukaisuudesta, käytetään neutraalia vaaleanviolettia väriä. Yksinker-

taistettuna punainen kanava on oikea/vasen, vihreä kanava on ylös/alas ja sininen kanava on poispäin pinnasta eli katsojaan päin (kuva 23). (OpenGL Tutorials, 2017; Oshchepkov, A. 2015.)



Kuva 24. Täydellisen pyöreä pallo ja siitä beikattu normaalikartta, jossa näkyvät kaikki mahdolliset normaalikartan RGB-arvot. (OpenGL Tutorials, 2017.)

Normaaleita, jotka osoittaisivat poispäin katsojasta, ei oteta normaalikartoissa huomioon, sillä ne on tehty karttaa ylhäältäpäin katsovan kameran näkökulmasta. Kaikki normaalikarttoihin tallennettu tieto noudattaa tätä kameran näkökulman periaatetta. Kameralla tässä yhteydessä ei tarkoiteta pelin kameraa, vaan kuvitteellista kameraa, jota käytetään normaalikartan taltioimiseen: 3D-tilassa "oikean" pelikameran kulma voi olla umpimähkäinen, sillä siinä vaiheessa normaalikarttojen koordinaatit on jo laskettu. (OpenGL Tutorials, 2017.)

Normaalikartta luodaan yleensä beikkaamalla se korkearesoluutioisesta meshistä, mistä voi lukea lisää luvussa "3.4.1 Beikkaus". Normaalikartan voi myös konvertoida valokuvasta tai maalata digitaalisesti 2D-sovelluksessa kuten Photoshop. Beikatut mallit toimivat yleensä paremmin kuin 2D-generoidut normaalikartat, jotka soveltuvat lähinnä 3D-malleihin, joiden pinta on yhteneväisen suuntainen (esimerkiksi seinät tai maasto). Normaalikartan voi tehdä myös hybriditekniikalla beikkaamalla isot ja keskikokoiset yksityiskohdat 3D-mallista ja yhdistämällä ne 2D-kuvasta generoituihin pieniin yksityiskohtiin kuten ihohuokosiin tai muuhun pinnan tekstuuriin. (Pluralsight, 2014; Polycount Wiki, 2016.)

Normaalikarttoja on olemassa monenlaisia: tangentti-, objekti- ja maailmaavaruusnormaalikarttoja. Yleisin normaalikarttatyyppi, jonka tunnistaa sinipunaisesta

ulkonäöstään, on tangenttiavaruusnormaalikartta. Karttojen nimet tulevat siitä, että 3D-maailmassa käytetään monia erilaisia koordinaattisysteemejä, kuten maailma-avaruutta, objektiavaruutta, kamera-avaruutta ja niin edelleen. Tangenttiavaruuskoordinaattisysteemin tarkoitus on määritellä tekstuurikoordinaattien sijainti mallin pinnalla. UV-koordinaatiston U- ja V-akselit vastaavat 3D-tilan X- ja Y-akseleita ja kolmas, Z-akselia vastaava akseli on polygonin pinnan normaali eli pinnan suunta N. (Lampel, J. 2017; Oshchepkov, A. 2015; Polycount Wiki, 2016.)

Tangenttiavaruusnormaalikartat käyttävät erityistä verteksidatalaskutoimitusta, jota kutsutaan tangenttiperustaksi. Valonsäteet ovat maailma-avaruudessa, mutta normaalikarttaan tallennetut normaalit ovat tangenttiavaruudessa. Kun normaalikartoitettu malli renderöidään, valonsäteet pitää muuntaa maailma-avaruudesta tangenttiavaruuteen tangenttiperustalla. Silloin kaikkia valoja verrataan normaalien suuntiin normaalikartassa ja määritetään, miten kukin pikseli valaistaan. Vaihtoehtoisesti valonsäteiden muuntamisen sijaan jotkin shaderit muuntavat normaalikartan normaalit maailma-avaruuteen, jolloin maailma-avaruusnormaaleja verrataan valonsäteisiin ja malli valaistaan vastaavasti. Lopputulos on sama, mutta kahden metodin eroavaisuus voi aiheuttaa ongelmia eri beikkaus- ja renderöintisovellusten välillä varsinkin UV-saumojen kohdalla. (Oshchepkov, A. 2015.)

Tangenttiavaruusnormaalikartta laskee normaalin suunnan joka sivun tangentin perusteella. Sen vuoksi se toimii normaalikartoista parhaiten mesheissä, joiden on deformoiduttava animaatioissa, kuten pelihahmoissa. Vaikka tangenttiavaruusnormaalikartta on kaikista yleisin normaalikarttatyyppi, muiden normaalikarttojen toimintaperiaatteiden ymmärtäminen on myös hyödyllistä. (Lampel, J. 2017; Pluralsight, 2014.)

Objektiavaruusnormaalikartta laskee normaalin suunnan koko objektin sijainnin perusteella eli sen ei tarvitse laskea sitä jokaiselle polygonisivulle erikseen ja sen vuoksi se on hieman kevyempi renderöidä. Samasta syystä se ei ole yhtä joustava kuin tangenttiavaruusnormaalikartta, sillä objektiavaruusnormaalikartassa oikea puoli on erivärinen kuin vasen puoli (2D-tilassa). Eriväristen puolien vuoksi UV-saarissa ei voi käyttää peilinnystä, joten UV-tilaa menee enemmän hukkaan. Toinen haittapuoli on se, että jos objektia käännetään, varjostus kääntyy sen mukana. (Lampel, J. 2017.)

Maailma-avaruusnormaalikartat ovat kaikista vähiten joustavia, sillä ne käyttävät globaaleja koordinaatteja. Objektit eivät saa kääntyä lainkaan, mikäli varjostusta ei haluta sotkea. Maailmatilanormaalikarttoja voi käyttää vain isoissa, staattisissa ja epäsymmetrisissä objekteissa kuten peliympäristöissä. (Lampel, J. 2017.)

3.3.5 Normaalikartat mobiilipeleissä

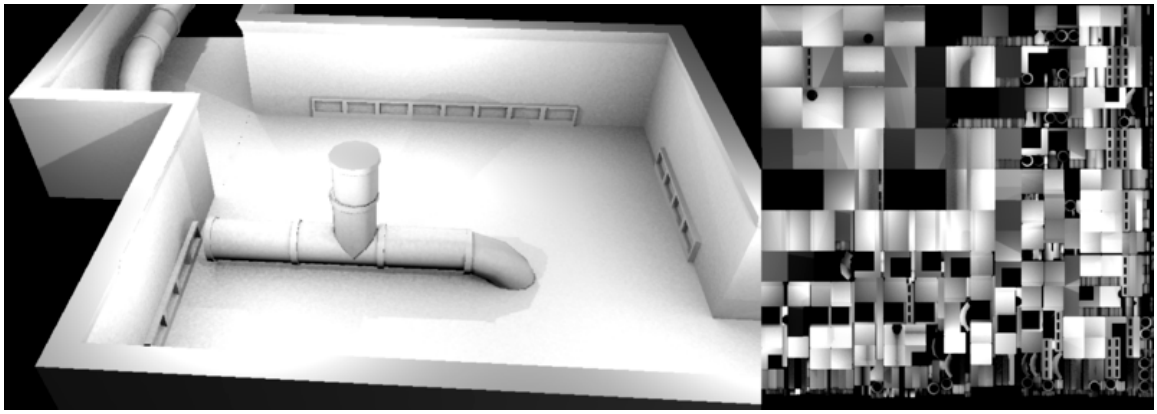
Kun tietokone- ja konsolialustojen shaderteknologia kehittyi noin loppuvuodesta 2003, normaalikarttojen käyttö peleissä yleistyi. Ensimmäinen konsoli jolla oli normaalikartoitukseen erikoistunut laitteisto oli Sega Dreamcast, 1998, mutta ensimmäinen pelikonsoli, joka laajalti hyödynsi normaalikarttoja vähittäismyyntipeleissä oli Xbox, 2001. Mobiililaitteisto on kehittynyt vasta hiljattain sille tasolle, että normaalikarttoja voidaan käyttää mobiilipeleissä. (Pluralsight, 2014; Wikipedia, 2018.)

Jokainen uusi tekstuurikartta tekee 3D-assetista hieman raskaamman suorituskyvylle. Esimerkiksi diffuusikartta on halpa renderöidä, mutta normaalikartta on kalliimpi. Malli, joka käyttää diffuusi-, normaali- ja spekulaarikarttaa on vähän yli kaksi kertaa kalliimpi renderöidä kuin malli, joka käyttää vain diffuusikarttaa. (Nimimerkki aggsol, 2015; Nimimerkki klownzie, 2015; Unity, 2018)

Suurimmalla osalla mobiililaitteista on PC- tai konsolilaitteeseen verrattuna hyvin heikko grafiikan prosessointiteho. Jotkin pelit käyttävät jo normaalikarttoja; esimerkiksi ShadowGun Legends -mobiilipelin hahmoissa on normaalikartat, vaikkakaan sen ympäristögrafiikassa ei ole (kuva 25). Suurin rajoite normaalikarttojen käyttämisessä mobiililla on valaistus, sillä dynaaminen pikselikohtainen valaistus on mobiililaitteille kohtuuttoman raskas. Dynaamisessa valaistuksessa jokainen valon koskema objekti joudutaan piirtämään ylimääräisen kerran per jokainen dynaaminen valo, mikä hidastaa peliä nopeasti. Dynaamisen valaistuksen sijaan voidaan käyttää valokarttoja ja valoluotaimia tai valaistusta per verteksi. (Nimimerkki aliyeredon2, 2016; Nimimerkit bluescrn, Rajmahal, 2014; Cupisz, R. 2011; Goldstone, W. 2012; Nimimerkki jbooth, 2016; Unity, 2018.)

Kuva 25. Kuvakaappaus Shadowgun Legends -mobiilipelistä, jossa hahmojen shadereissa käytetään normaalikarttoja, mutta joka silti saavuttaa kuvataajuuden 60 fps iPad 2 -alustalla ja 30 fps iPad, iPhone 4 & 3GS -laitteilla. (Cupisz, R. 2011; MADFINGER Games, 2018.)

Valokartta on tekstuuri, johon on beikattu ennaltarenderöity tieto valaistuksesta (kuva 26). Valokarttoja käytetään yleensä staattisissa objekteissa kuten ympäristögrafiikkassa. Valoluotaimet ovat toiminnaltaan samankaltaisia kuin valokartat eli ne varastoivat beikattua tietoa 3D-tilan valaistuksesta. Niiden ero on siinä, että kun valokartat varastoivat tietoa siitä, miten valo osuu tilan *pintoihin*, valoluotaimet varastoivat tiedon valon matkustamisesta tyhjän tilan poikki. Valoluotainten tärkein käyttötarkoitus on tarjota korkealaatuista valaistusta tilan poikki liikkuville objekteille, kuten hahmoille, jotka käyttävät normaalikarttaa. (Polycount Wiki, 2015; Unity, 2018; Wikipedia, 2018)



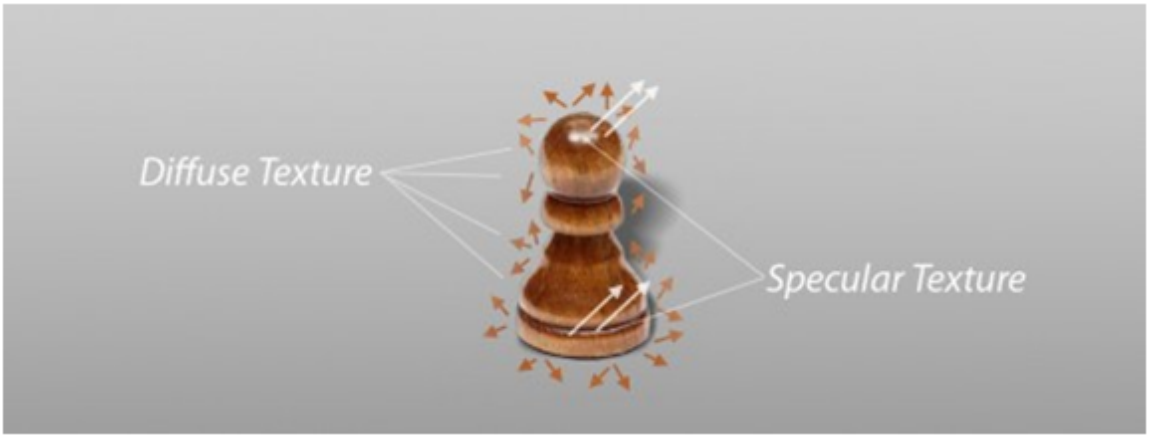
Kuva 26. Kuvassa vasemmalla on 3D-skene, johon on beikattu valaistus ja oikealla valokartta 2D-muodossa. (Nimimerkki Narpas, 2016.)

Vaikka normaalikarttoja ei vielä käytetä yleisesti mobiilipeleissä, niitä voidaan hyödyntää teksturointiprosessissa. Normaalikarttojen vihreään kanavaan (RGB-kanavista) on tallennettu tieto vertikaalisesti suuntautuneista normaaleista. Vihreää kanavaa voidaan käyttää esimerkiksi kertovassa sekoitustilassa diffuusikartassa, mikä luo illuusion malliin ylhäältä alaspäin osuvasta valosta. Lopputulos ei ole ihan samannäköinen kuin normaalikartoitettu malli (kuva 27), mutta tekniikka on silti kätevä. (Nimimerkit Fingus, Joopson 2013; Oberson, P. 2005.)

Kuva 27. Jäljitelmä normaalikartasta: tieto malliin osuvasta valosta sisällytettynä diffuusikartan tietoon (vas.) sekä malli, joka käyttää normaalikarttaa. (Oberson, P. 2005.)

3.3.6 Spekulaarikartta

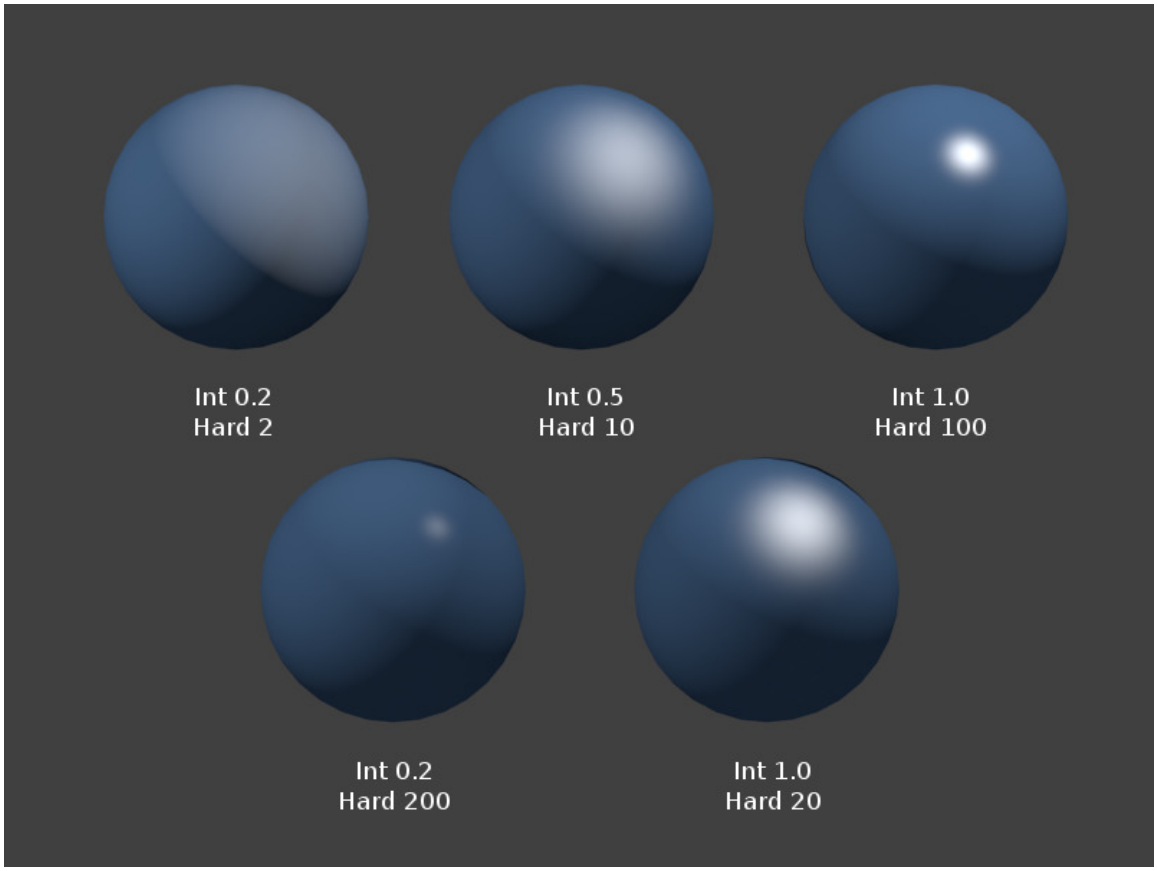
Kaikki asiat, jotka näemme oikeassa maailmassa ja jotka eivät hohda valoa, ovat näkyviä, sillä ne heijastavat valoa. Kaikki pinnat heijastavat valoa kahdella eri tavalla - diffuusisesti eli hajottavasti ja spekulaarisesti eli heijastavasti (kuva 28). Diffuusisessa heijastuksessa valonsäteet hajaantuvat eri suuntiin, mutta spekulaarisessa heijastuksessa valonsäteet heijastuvat pinnasta samassa kulmassa kuin ne tulevat. (Toledo, P. 2010.)



Kuva 28. Pinnan diffuusi- ja spekulaariheijastus visualisoituna. (Toledo, P. 2010.)

Spekulaarisuudella tarkoitetaan pinnan peilimäisiä heijastuksia. Peilimäisyys on siinä mielessä kehno suomennos spekulaarisuudelle, että tietokonegrafiikassa ympäristön muut pinnat eivät yleensä peilaannu spekulaarisesta heijastuksesta. Ympäristöstä heijastuvien yksityiskohtien laskelmoimiminen olisi hyvin raskasta laitteistolle, joten spekulaarinen heijastus heijastaa usein vain valonlähteitä. (Wikipedia, 2018.)

Tietokonegrafiikassa materiaalin spekulaarisuus määrittää, kuinka paljon pinta heijastaa valoa. Spekulaarisuuden tärkein tehtävä on määrittää valon kohokohtien kirkkaus, heijastusten väri ja pinnan hohtavuus, eli se, heijastaako pinta valoa hajaantuneesti vai keskittyneessä kohokohdassa (kuva 29). (Nimimerkki spek, 2008; Wikipedia, 2018.)



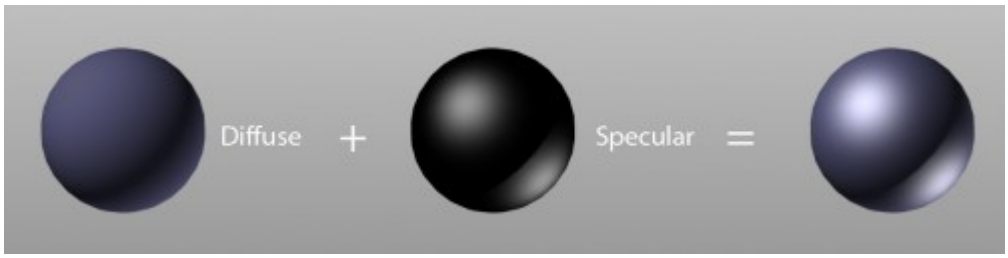
Kuva 29. Palloja, joiden spekulaarisuuden voimakkuus vaihtelee. (Blender, 2018.)

Sopiva pinnan spekulaarisuus on tärkeää, kun halutaan määrittää, mitä materiaalia pinta on. Esimerkiksi kiiltävä metallinen materiaali on hyvinkin heijastava materiaali, toisin kuin mattapintainen sementti. 3D-objektin spekulaarisuutta voidaan hallita spekulaarikartalla. (Nimimerkki spek, 2008; Pluralsight, 2014; Polycount Wiki, 2014; Wikipedia, 2018.)

Spekulaarikartat toimivat niin, että kuvakartassa tummat tai mustat alueet määrittävät, että pinta heijastaa vähän tai ei lainkaan valoa ja vaaleat tai valkoiset alueet heijastavat paljon valoa. Spekulaarikartat ovat hyödyllisiä, kun objektin eri osien halutaan heijastavan valoa eri voimakkuuksilla. Esimerkiksi hahmon iho heijastaa valoa vähemmän kuin huulikiilto ja puuvillavaatteet heijastavat valoa vähemmän kuin iho (kuva 30). Hyvin tehty spekulaarikartta voi saada aikaan huikean eron grafiikan laadussa. (Nimimerkki spek, 2008; Splash Damage Wiki, 2017; Unity, 2018.)

Kuva 30. Esimerkki hahmon kasvojen spekulaarisuudesta. Huulten, poskipäiden ja nenänpään alueet ovat vaaleammat eli kiiltävämmät kuin muu iho. (Oberson, P. 2005.)

Renderöinnin aikana pinnan diffuusisuus ja spekulaarisuus lasketaan eri hetkinä (kuva 31). Ensin lasketaan pinnan diffuusisuus ja sitten spekulaarinen heijastus. Lopuksi spekulaarisuus lisätään diffuusiin yksinkertaisella matemaattisella yhtälöllä ja lopullinen tulos renderöidään ruudulle. (Toledo, P. 2010.)



Kuva 31. Spekulaarisuus lisätään pinnan diffuusin päälle. (Toledo, P. 2010.)

Matemaattisesti spekulaarikartta lisää siis vain yhden ylimääräisen kertoimen lisää valaistusyhtälöön, joten sen koetaan olevan halpa. Kartat kuten normaalikartta, jotka vaikuttavan pinnan topografiaan, ovat kalliimpia. Spekulaarishaderi on toki kalliimpi kuin pelkkä diffuusishaderi, sillä spekulaarikartasta koituvat yleiskustannukset kuten valon suunta tuottavat laitteistolle lisää työtä. (Nimimerkit Everspace, poistettu nimimerkki, 2015.)

Spekulaarikarttaa voidaan käyttää mobiilipelissä normaalikartan jäljittelemiseen sisällyttämällä normaalikartan valaistusdata diffuusitekstuuriin (kts. luku "4.4.6 Teksturointi") ja

käyttämällä sen sijaan spekulaarikarttaa (kuva 32). (Nimimerkki aliyeredon2, 2016; Nimimerkki LightingBox2, 2016.)

Kuva 32. Kuvan ympäristögrafiikassa käytetään spekulaarikarttaa normaalikartan sijasta samankaltaisen efektin jäljittelemiseksi halvemmin. (Nimimerkki aliyeredon2, 2016.)

3.4 Optimointi

Optimoinnilla tietokonegrafiikassa tarkoitetaan prosessia, jolla grafiikka-assetti saadaan toimimaan mahdollisimman tehokkaasti. Optimointia tarvitaan, silllä ilman sitä laitteisto ei jaksa renderöidä peliassetteja. Videopelimallit tulee optimoida esimerkiksi topologialtaan, verteksimäärältään ja materiaaleiltaan. (Merriam-Webster, 2018; Yot, R. 2012.)

3.4.1 Beikkaus

Beikkaus on termi, jota käytetään usein pelinkehityksessä, kun puhutaan tietokonegrafiikasta. Kaiken 3D-skenen informaation renderöiminen reaaliajassa olisi tietokoneelle hyvin raskasta, joten osa tiedosta voidaan tallentaa etukäteen beikkaamalla. Beikkaus tarkoittaa datan tiivistämistä yksinkertaisempaan ja pysyvämpään muotoon. Beikkaamalla voidaan tallentaa tietoa esimerkiksi tekstuureista, animaatioista, simulaatioista ja valaistuksesta. (Pluralsight, 2014; Polycount Wiki, 2018; Trammell, K. 2016.)

Tekstuurien eli kuvakarttojen beikkaus on kaikista yleisin beikkauksen käyttötarkoitus. Menetelmän voima piilee sen kyvyssä muuntaa geometrialtaan raskaan 3D-pinnan yksityiskohdat 2D-kuvatiedostoksi, eli yhdestä 3D-mallista "siirretään" yksityiskohtia toiseen (kuva 33). Beikkausprosessissa on mahdollista generoida monta erilaista karttatyyppiä, jotka taltioivat pinnan yksittiäisiä piirteitä, kuten ambientin okkluusion tai normaalien suunnat. (Polycount Wiki, 2018; Trammell, K. 2016.)

Kuva 33. Raskasmalli (vas.) on liian raskas pelikäyttöön, sillä se koostuu noin viidestä miljoonasta polygonikolmiosta. Kevytmalli (kesk.) on sopiva pelikäyttöön, sillä siinä on vain 6 000 kolmiota. Raskasmallin pinnan data saadaan siirrettyä kevytmalliin beikkaamalla siitä normaalikartta (oik.).

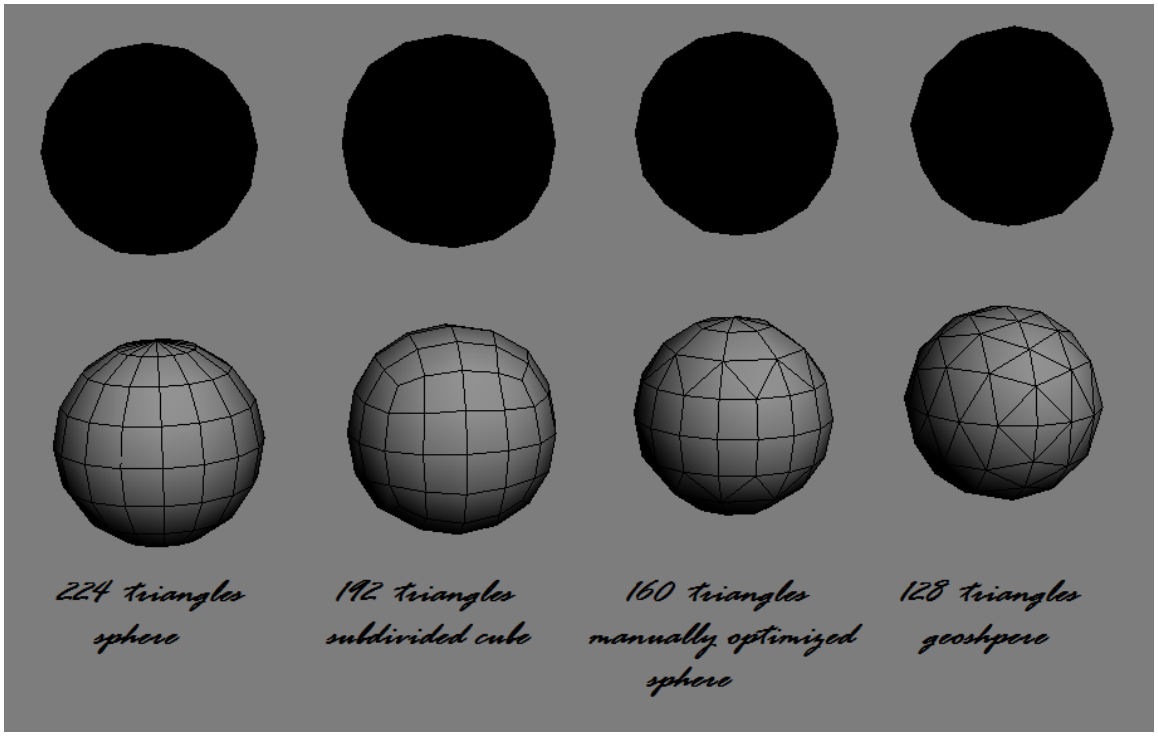
Beikkaus toimii teknisesti niin, että beikkausohjelman sisällä 3D-tilassa sijaitsee beikkaustyökalu tietyn etäisyyden päässä 3D-mallista, jonka UV-karttaan beikkaus kohdistetaan (esimerkiksi pelikäyttöön tarkoitettu kevytmalli). Beikkaustyökalu langettaa säteen sisäänpäin toista, beikattavaa mallia (esimerkiksi raskasta 3D-veistosta) kohti. Kun säde leikkaa sisemmän mallin, se taltioi pinnan yksityiskohdat ja tallentaa ne tekstuurikarttaan, joka vastaa ensimmäisen mallin tekstuurikoordinaatteja (kuva 34). (Polycount Wiki, 2018.)

Kuva 34. Kevytmalli, sen UV-kartta ja raskasmallista beikattu normaalikartta.

Myös 3D-skenen valaistus voidaan beikata ja se tehdään yleensä pelimoottorissa. Peleissä voidaan käyttää staattista tai dynaamista valaistusta. Dynaamiset valot reagoivat interaktiivisesti muutoksiin 3D-skenessä, kun taas staattiset valot ja varjot ovat vastaavasti muuttumattomat. Staattisten valojen käyttäminen säästää renderöintiresursseja, mikä on etenkin mobiililla tärkeää. Valaistuksen beikkaaminen liittyy kuitenkin enemmän ympäristögrafiikkaan, sillä hahmot ovat miltei aina dynaamisia, joten niihin ei voi tai kannata beikata staattista valodataa. (Trammell, K. 2016.)

3.4.2 Retopologia

Topologia tarkoittaa mallin pohjapiirustusta eli sitä, miten verteksit ja reunat on sijoiteltu mallin pinnalle. Hyvä topologia on vaihteleva käsite, sillä "parhaat" topologiakäytänteet riippuvat mallin tulevasta käyttötarkoituksesta (kuva 35). Mikäli tarkoitus on esimerkiksi mallintaa pohjamalli digitaaliseen veistämiseen, topologiassa on tärkeää käyttää mahdollisimman paljon nelikulmaisia polygoneja. Vastaavasti mobiilipelikevytmalli voi sisältää enemmän kolmioita, sillä mobiilipelissä kevyt verteksimäärä on pinnan jaettavuutta tärkeämpi. Videopelien polygonimäärä on rajattu ja se hyödynnetään parhaiten silloin, kun polygoneja käytetään vain siellä, missä niitä tarvitaan. (Polycount Wiki, 2017; Unity, 2018.)



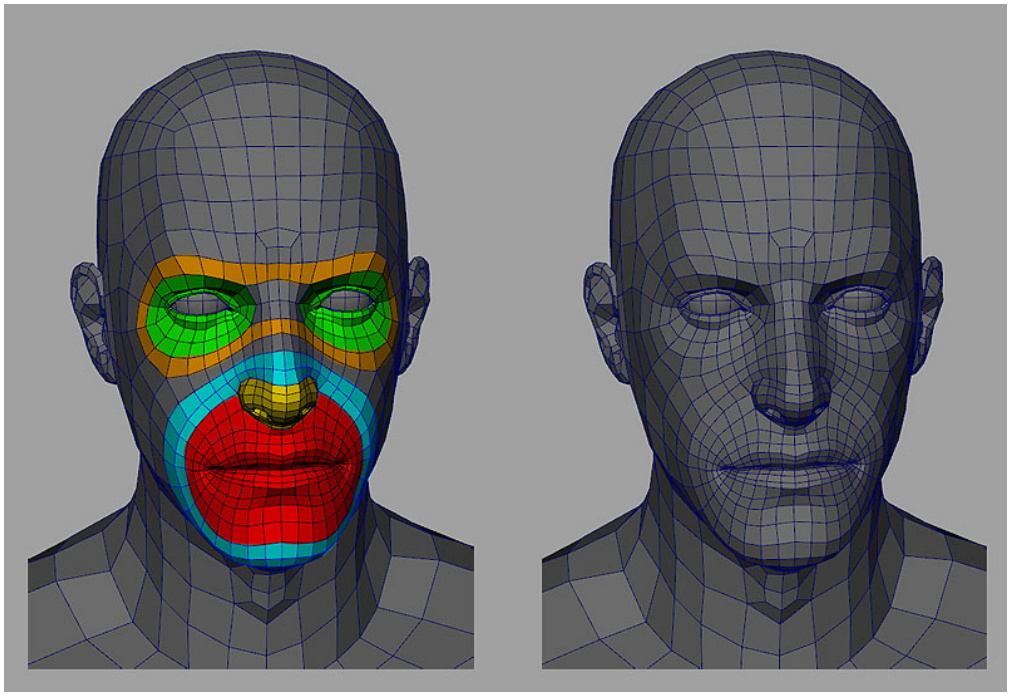
Kuva 35. Esimerkkejä erilaisista tavoista rakentaa 3D-pallon topologia. (Hennepe, S. 2010.)

Topologian sottaisuudella tarkoitetaan kohtuuttoman korkeaa polygonimäärää tai topologiaa, joka aiheuttaa ongelmia UV-saumoja merkatessa, valon mallin pintaan osuessa tai mallin deformoituessa. Topologialtaan sottasen mallin, kuten 3D-veistoksen, pinta joudutaan usein täysin uudelleenrakentamaan eli retopologisoimaan, sillä videopelimallin topologian tulee olla suorituskyvyltään tehokas. (Polycount Wiki, 2017; Unity, 2018.)

Pinnan polygonit määrittävät mallin muodon eli siluetin (kuva 35). Optimoidusta polygonibudjetista ei kannata tuhlata verteksejä sellaisiin kohtiin, jotka eivät vaikuta mallin siluettiin. Liika kitsastelukin voi kostautua, sillä liian harvasti tai epäoptimaalisesti sijoiteltu topologia saattaa aiheuttaa terävän kulman mallin siluetissa. Liian harva topologia voi johtaa myös verteksien normaalien välisiin isoihin muutoksiin, jotka voivat aiheuttaa varjostusongelmia. Pyöreät tai kaarevat pinnat vaativat enemmän polygoneja kuin suorat pinnat. (Polycount Wiki, 2017.)

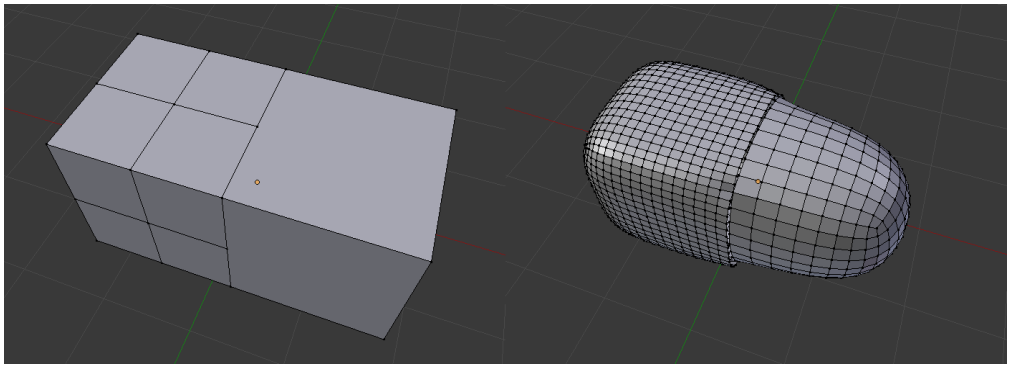
Hyvässä topologiassa reunaluupit on sijoiteltu harkiten niin, että malli voi deformoitua eli vääristyä vapaasti. Verteksien ja reunojen sijaintien on oltava tietynlaiset, jotta malli voi joustaa, taipua ja venyä animaatioissa (kuva 36). Oikeaoppinen topologia on kaikista tärkeintä siellä, mistä mallin on venyttävä paljon, kuten haarovälin ja takapuolen alueella, kainaloissa ja olkapäissä, suupielissä ja poskissa, polvissa, kyynärpäissä, käsissä ja

sormissa ja niin edelleen. Toisin sanoen hahmomallin hyvä topologia on tärkeämpi kuin staattisen objektin kuten kiven. (Polycount Wiki, 2017.)



Kuva 36. Hyvä kasvojen topologia, jossa reunaluuppien virtaus seuraa deformoinnin vaatimuksia. (Parker, T. 2011.)

Hyvään topologiaan eivät kuulu T-verteksit (kuva 37), päällekkäiset pinnat, raot, väärin päin kääntyneet pinnat, mallin sisäiset, näkymättömät pinnat, leijailevat verteksit tai muut kammotukset. (Polycount Wiki, 2017.)



Kuva 37. Esimerkki siitä, mihin huono topologia (T-verteksi) voi johtaa. (Wikipedia 2018.)

3.4.3 Retopologiaohjelmistot

Erilaisia retopologiaohjelmia ovat esimerkiksi 3D-Coat, TopoGun ja Wrapit. Muitakin vaihtoehtoja löytyy - muun muassa mallinnusohjelmia Blenderiä, Mayaa tai Modoa voidaan käyttää retopologisointiin. (Polycount Wiki, 2017; nimimerkit SwdPwnzDggr, WarrenM, 2015; Williamson, J. 2009.)

On olemassa myös autoretopologiatyökaluja, kuten ZBrushin ZRemesher. Se on toimiva ratkaisu joillekin malleille, mutta esimerkiksi "hard surface"-pinnat tuottavat sille vaikeuksia. Myöskään assettien, joiden on liikuttava ja taivuttava animaatioissa paljon, retopologiaa ei voi täysin automatisoida, sillä tietokone ei osaa sijoittaa optimaalisia deformaatioreunaluuppeja. Autoretopologia voi olla toimiva ratkaisu esimerkiksi kiville tai muille liikkumattomille, orgaanisille muodoille. (Nimimerkit DerRazputin, FildoSaggins, OccultMonk, 2017.)

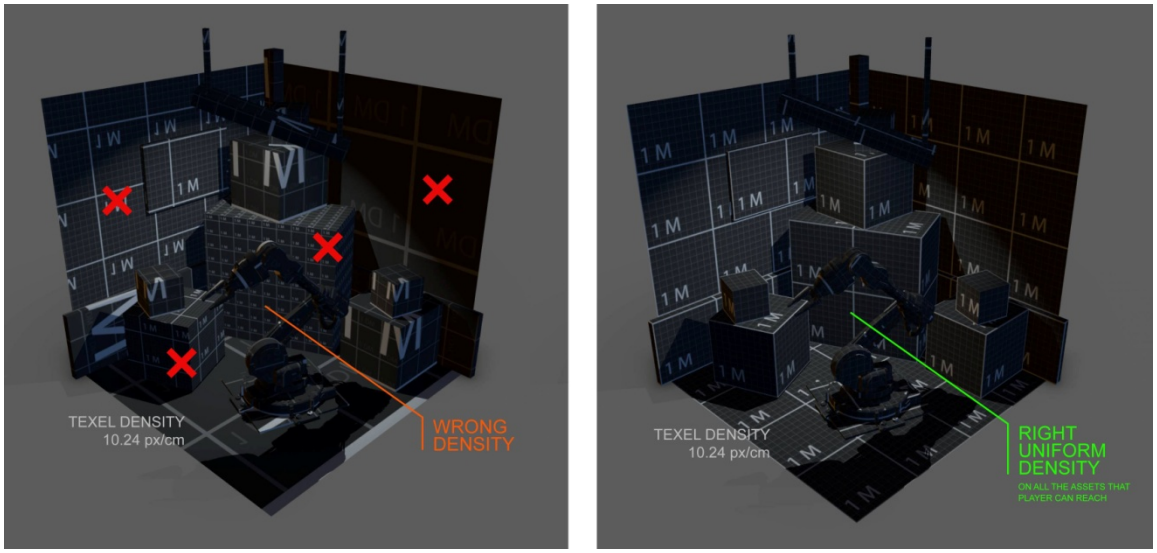
ZBrushin automaattiset retopologiatyökalut kuitenkin kehittyvät koko ajan. ZBrushin kehittäjä Pixologic tiedostaa, että automaattinen työnkulku ei ole vielä täydellinen, mutta käyttäjä voi auttaa ZRemesheriä esimerkiksi piirtämällä kontrollikurveja topologiavirtauksen opastamiseksi. ZRemesherin toivotaan poistavan esteet taiteellisen vapauden tieltä automatisoimalla tekniset työvaiheet ja ehkäpä tulevaisuudessa retopologian hoitavat tietokoneet. (Pixologic, 2018.)

3.4.4 Tekstuurien optimointi

Tekstuurikartat toimivat kaikista tehokkaimmin, mikäli niiden kuvakoko on kahden potenssissa, kuten 512x512, 1024x1024 tai 256x1024 pikseliä. Pelinkehityksessä tekstuurikartat pyritään pitämään mahdollisimman tiiviissä koossa, vaikka mikään ei tietenkään estä kuvatiedostojen työstämistä isommassa koossa ennen niiden peliin lataamista. Tekstuurikartan minimi- tai maksimikoon voi selvittää parhaiten testaamalla teksturoitua objektia pelissä ja pohtimalla, kuinka pienessä koossa tekstuurikartta vielä toimii sekä performanssin että estetiikan kannalta. (Nimimerkki Gintas_, 2013; Unity, 2018.)

Tekstuurin ulkomuotoon pelissä vaikuttavat tekstuurikartan kuvakoko, se miten 3D-malli on unwrapattu, 3D-mallin koko ja 3D-skenen koko. Tekstuurikartan kokoa määrittäessä tulee pitää mielessä se, että kaikilla objekteilla peliskenessä tulisi olla yhtenäinen tekstuurien skaala (tekselitiheys), sillä suureksi venytetty matalaresoluutioinen tekstuuri

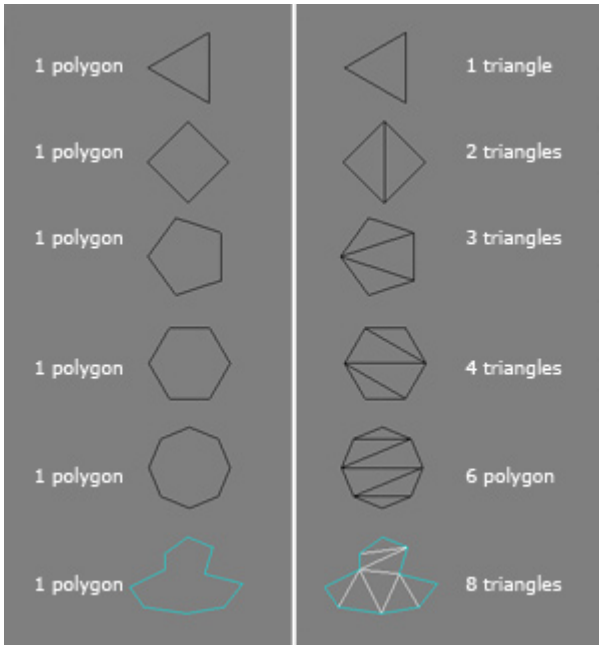
näyttää huonolta korkearesoluutioisen tekstuurin vieressä (kuva 38). Tekstuurikoon valintaan vaikuttaa myös se, kuinka lähelle assettia pelaaja voi mennä ja se, kuinka tärkeä assetti on. Pelihahmot ovat usein läheltä tarkasteltavia ja tärkeitä pelielementtejä, joten niillä on monesti suhteessa ympäristöassetteja kookkaammat tekstuurikartat. Mobiilipelihahmolle hyvä tekstuurikartan koko saattaisi olla esimerkiksi 1024x1024 pikseliä, mutta täydellisiä sääntöjä ei ole olemassa ja vastaus tulee selvittää projektikohtaisesti. (Nimimerkit CheeseOnToast, Quasar, 2011; Iezzi, L. 2016; Polycount Wiki, 2016.)



Kuva 38. Mallikuva epäyhtenäisestä (vas.) ja yhtenäisestä (oik.) tekselitiheydestä. (Iezzi, L. 2016.)

3.4.5 Polygoni- ja verteksimäärä

Kaksi yleistä mittayksikköä videopelimallin "hinnalle" ovat polygonimäärä ja verteksimäärä. Polygonimäärää mitattaessa puhutaan todellisuudessa kolmiomäärästä, sillä grafiikkaprosessorit mittaavat mallit vertekseissä ja kolmioissa - ne jakavat esimerkiksi nelikulmaisen polygonin kahdeksi kolmioksi (kuva 39). 3D-mallinnusohjelman ilmoittama polygonimäärä saattaa olla harhaanjohtava, sillä pelimoottoriin siirrettäessä malli konvertoidaan aina kolmioiksi. (Polycount Wiki, 2017.)



Kuva 39. Havainnollistava kuva siitä, miten tietokone jakaa monikulmaiset polygonit kolmioiksi. (Chadwick, E. 2015.)

Verteksimäärä on tärkeämpi kuin kolmiomäärä, vaikka artistit mittaavatkin mallit tavallisesti kolmioissa historiallisista syistä. Ruohonjuuritasolla kolmiomäärä ja verteksimäärä voivat olla yhtäläiset, mikäli kaikki kolmiot yhdistyvät toisiinsa: yksi kolmio käyttää kolme verteksiä, kaksi kolmiota käyttää neljä verteksiä, kolme kolmiota käyttää viisi verteksiä ja niin edelleen. Asia ei kuitenkaan ole ihan niin yksinkertainen, sillä esimerkiksi UV-saumat, tasoittavat ryhmät sekä materiaalin vaihtuminen kolmioiden välillä aiheuttavat fyysisen "katkoksen" mallin pinnalla, kun malli renderöidään pelissä. Verteksit joudutaan duplikoimaan katkoskohdissa, jotta malli voidaan lähettää renderöitävissä palasissa näytönohjaimelle. Aiheeton tuhlaus ylimääräisiin UV-saumoihin, tasoittaviin ryhmiin ja useisiin materiaaleihin aiheuttaa siis isomman verteksimäärän ja on raskaampaa laitteistolle. Se myös lisää mallin muistin käyttöä, sillä verteksejä on enemmän lähetettävänä ja varastoitavana. (Polycount Wiki, 2017.)

Videopeliassetin maksimikolmiomäärä on yksilöllinen ja riippuu monesta tekijästä. Lopputulokseen päästään testaamalla assetteja ja tekemällä yhteistyötä esimerkiksi projektin johtavan ohjelmoijan kanssa. (Polycount Wiki, 2017.)

3.4.6 Piirtopyynnöt ja päällepiirtäminen

Vaikka niistä puhutaan paljon, polygonimäärän tai tekstuurikarttojen koon ei koeta olevan suurin pullonkaula nykysukupolven pelejä optimoidessa, vaan ongelmaksi koituvat usein piirtopyynnöt ja päällepiirtäminen. Ne ovat teknisiä yksityiskohtia mobiilipelin suorituskyvyn kannalta ja ne on mainittu siksi, että ymmärretään, että mobiilipeliassetin optimointiin kuuluu monia muitakin seikkoja kuin maksimipolygonimäärä tai sopivin tekstuurin koko. (Pinto, H. 2010; Soenke, C. 2010.)

Yksi piirtopyyntö edustaa yhtä kutsua materiaalilta näytölle Piirtopyyntö sisältää kaiken tiedon puhelimen grafiikkaprosessorille tekstuureista, tiloista, shadereista, renderöitävistä objekteista, buffereista ja muista sellaisista kapseloituneena suorittimen työhön, kun se valmistelee piirtämisresursseja grafiikkaprosessorille. Kaiken tämän tiedon muuntaminen laitteistokomennoiksi grafiikkaprosessorille on hyvin kallista suorittimelle. Koska jokainen skenessä sijaitseva materiaali tuottaa piirtopyynnön, useampi uniikki objekti ja materiaali 3D-skenessä nostaa piirtopyyntöjen määrää. Suoritin ei aina pysty valmistelemaan kaikkea tietoa grafiikkaprosessorille ajoissa, mikä voi johtaa kuvataajuuden putoamiseen ja nykimiseen. (Nimimerkit bumble864, Tseng, razormax 2012; Freeman, J. 2017; Jukić, T. 2015.)

Päällepiirtäminen on arvo, joka kertoo, kuinka monta kertaa joka pikseli joudutaan päällekirjoittamaan renderöintiprosessissa yhden kuvan tai kuvasarjan aikana. Grafiikkaprosessoreilla on tietty toteutusaste eli määrä pikseleitä, jonka näytönohjain voi renderöidä tietyssä ajassa. (Sanakirja.org, 2018.)

Assetit voi optimoida piirtämiskutsuja ja päällepiirtämistä vastaan monella tavalla. Piirtämiskutsujen määrää vähentääkseen tulee käyttää mahdollisimman vähän eri materiaaleja, jonka voi tehdä niputtamalla pienet objektit yhteen sekä yhdistämällä niiden tekstuurit tekstuuriatlakseen. Läpinäkyvät materiaalit ovat raskaita päällepiirtämisen kannalta ja kannattaa miettiä, voisiko saman asian ajaa käyttämällä 3D-geometriaa. Graafisesti tasoittavia ryhmiä ja UV-saumoja tulee käyttää mahdollisimman vähän. (Soenke, C. 2010.)

Tiivistettynä optimointiprosessissa ei kannata repiä hiuksia niin omasta kuin 3D-hahmonkaan päästä kolmiomäärän vuoksi. Optimisaatio on ehdottomasti olennaista, mutta on mietittävä, milloin työn määrä ja tulosten korrelaatio suorituskyvyn kannalta ei ole enää vaivan arvoista. Teknologia kehittyy nopeasti eteenpäin ja nykypäivänä piirtä-

miskutsut ja päällepiirtäminen ovat usein isompia ongelmia kuin verteksimäärä tai kuva-koko. (Nimimerkki CrazyButcher, 2007.)

4 PROJEKTITYÖ

Projektityöosuudessa kehitettiin mobiilipelihahmo digitaalisesti veistämällä ZBrushissa. Retopologia- ja teksturointiprosessi suoritettiin Blenderissä ja Photoshopissa ja lopputulos eli kevyt hahmomalli renderöitiin Marmoset Toolbag 3:ssa.

4.1 Suunnittelu

Projektityö aloitettiin kartoittamalla, mitä sillä halutaan saavuttaa. Tavoitteet konkretisoituivat aiheanalyysia tehdessä: haluttiin edistää omaa ammattiosaamista sekä kirjoittaa informatiivinen dokumentaatio 3D-veistämisestä. Työn prioriteetti oli oppiminen eikä nopeus tai tehokkuus.

Veistäessä tai taidetta yleensä luodessa on tärkeää käyttää referenssikuvia. Ennen veistämistä kerättiin verkosta hahmon toteutusta varten kuvia esimerkiksi peleistä ja elokuvista, joiden tyylittely oli halutunlainen, sekä muutamia kuvia vaatetuksesta ja hiustyyleistä. Kuvat järjesteltiin omiin kuvatiedostoihinsa, joita käytettiin inspiraation lähteenä ja referenssinä.

Viimeistelty konseptikuva jäi puuttumaan. Mallinnusvaiheessa huomattiin, että se olisi ollut hyvin tärkeää olla olemassa, sillä se nopeuttaa mallinnusta ja pitää suunnan yhtenäisenä läpi pitkän prosessin.

4.2 ZBrushin käyttöliittymä

ZBrushin käyttöliittymää alettiin opiskelemaan videosarjan "An Introduction to ZBrush with Kurt Papstein" mukana. Noin kymmenen minuutin pituisia jaksoja oli kymmenen kappaletta ja niiden mukana oli helppo seurata. Parhaiten asiat jäivät mieleen, kun videoiden mukana teki harjoituksia. (Papstein, K. 2015.)

ZBrush nojaa jonkin verran pikanäppäinten käyttämiseen ja sen uniikki käyttöliittymä pelotti aluksi. Oppimisprosessissa auttoi tärkeimpien pikanäppäinkomentojen kirjoittaminen näkyviin muistilapulle, kunnes ne tulivat selkäytimestä. Ylös kirjoitettiin esimerkiksi kameran kontrollit, transformointityökalut, sekä tärkeimpien siveltimien pikanäppäinpolut.

Aloittelijan ei kannata alkaa kikkailemaan eri sivellinten kanssa, vaan on tärkeintä oppia käyttämään perussiveltimiä monipuolisesti. Esimerkiksi Blizzard Animation seniorihahmomallintaja Shannon Thomas käyttää reilusti suurimman osan ajasta noin kuutta perussivellintä (kuva 40).

Kuva 40. Shannon Thomas suosittelee aloittelijoillekin näitä kuutta sivellintä: Standard Brush, Damien Standard Brush, Smooth, Pinch, Move, Polish, Clay/Clay BuildUp ja Trim Curve. (Thomas, S. 2016.)

4.3 3D-veistäminen

4.3.1 Veistämispohjan luominen

3D-veistämisen voi aloittaa muutamalla eri metodilla: jotkut veistäjät aloittavat muovaamalla mallin yhdestä pallosta tai kokoamalla sen primitiivimesheistä. Opinnäytetyössä kokeiltiin ZSphere-metodia, jossa käytetään ZBrushin uniikkia ZSphere-työkalua. Mallin muovaaminen aloitettiin valitsemalla työkalupaletista ZSphere-työkalu ja piirtämällä se kankaalle. Näin syntyi juuri-ZSphere ja kun se oli piirretty, voitiin siirtyä muokkaustilaan. Mikäli muokkaustilaan ei mennä, ZBrush jatkaa ZSpherejen piirtämistä 2.5D-kankaalle. 2.5D-tilan kankaan voi tyhjentää Ctrl+N -pikanäppäinkomennolla. Mikäli malli katoaa näkyvistä, takaisin perusnäkymään voi kohdentaa F-pikanäppäimellä. Hahmo tuli olemaan

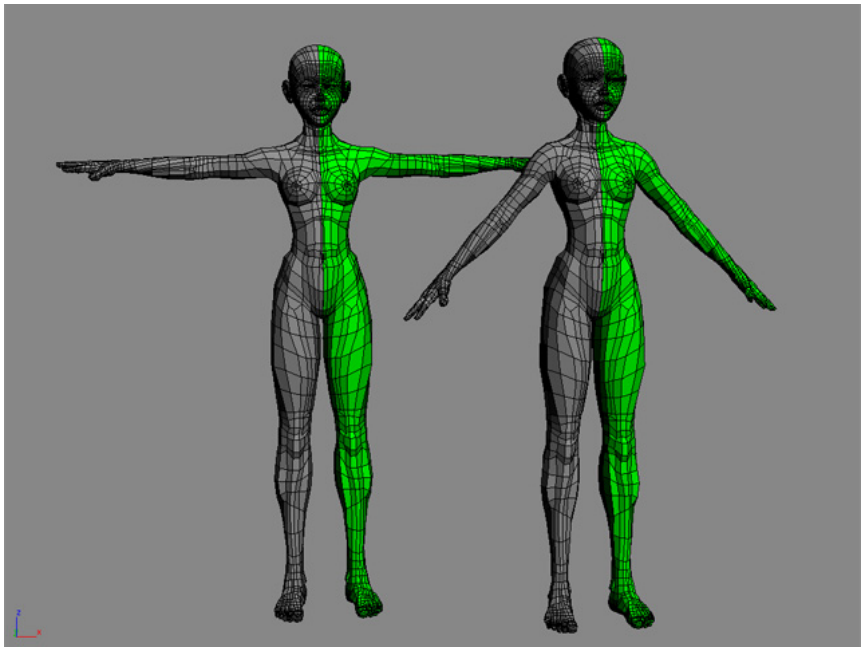
symmetrinen, joten jo ZSphere-tilassa aktivoitiin symmetria. ZBrush käyttää oletusakselina etuperspektiivistä kuvatulle, pystysuuntaiselle symmetrialle X-akselia.

Juuri-ZSphereen voi lisätä ZSpherejä piirtämistilassa. Uusien ZSpherejen lisääminen toimii klikkaamalla ja raahaamalla juuri-ZSpherestä. Uusia ZSpherejä voi liikuttaa tai skaalata. Niillä luonnostelemalla hahmolle luotiin rintakehä, käsivarret, pää ja jalat. Kahden ZSpheren välistä voi klikata luodakseen siihen uuden ZSpheren, eli kun hahmon vyötäröstä haluttiinkin kapeampi, klikattiin vain lantion ja rintakehän välistä tilaa ja luotiin siihen pienempi ZSphere.

ZSphere-työkalu on tehokas orgaanisten mallien luonnosteluun ja sen käyttö oli melko intuitiivista - jokaisen artistin kannattaa kokeilla sitä löytäkseen itsellensä sopivimman metodin. Sen käyttäminen oli jouhevaa ja nopeaa kokemattomuudesta huolimatta. Kuva 41 esittää ZSphere-luonnostelun lopputuloksen.

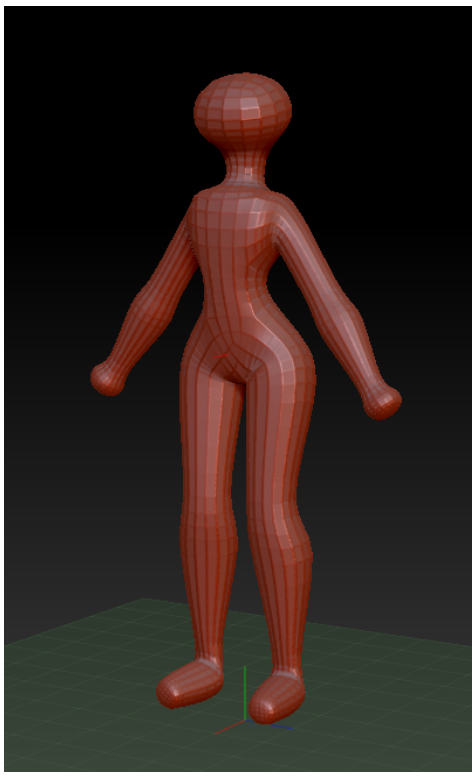
Kuva 41. ZSphere-työkalulla piirretty luonnos hahmon mittasuhteista.

Hahmoa luonnostellessa mietittiin, kannattaisiko se poseerata niin kutsuttuun T-asentoon vai A-asentoon eli tulisiko käsivarsien olla 90 vai 45 asteen kulmassa (kuva 42). Päätettiin käyttää A-asentoa, sillä ihmisen olkapäät ovat suurimman osan ajasta lähempänä A-asentoa.



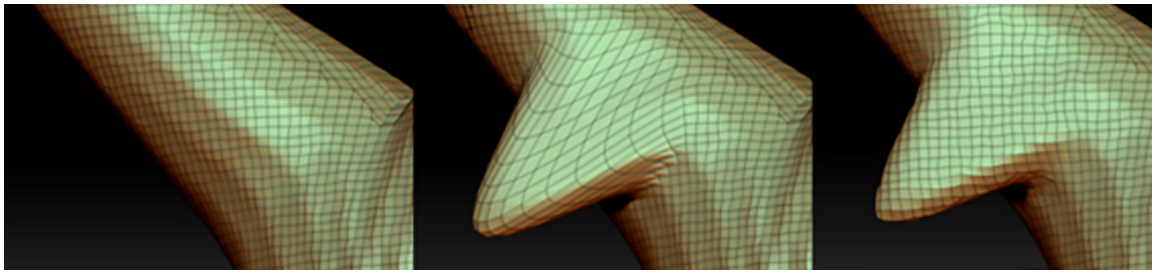
Kuva 42. T- ja A-asennot. (Wiro, A. 2005.)

Kun ZSphere-luonnokseen oltiin tyytyväisiä, siitä generoitiin polygonimesh Adaptive Skin -työkalulla matalalla resoluutiolla (kuva 43). 3D-veistämisessä on olennaista pitää pinnan geometria kevyenä, sillä kevytmalliin on helpompi tehdä suuria muutoksia mallin muodon muuttumatta muhkuraiseksi.



Kuva 43. ZSphere-luonnoksesta Adaptive Skin -työkalulla generoitu mesh.

Seuraavaksi käytettiin ZBrushin DynaMesh-työkalua. DynaMesh-tila toimii niin, että se jakaa mallin pinnan tasaisen tiheästi nelikulmaisiin polygoneihin ja sen voi päivittää helposti vastaamaan mallin uutta muotoa (kuva 44). DynaMesh on vahvimmillaan silloin, kun hahmon muotoa vielä haetaan ja geometriaa on päivitettävä usein. DynaMesh-malliin voi myös sulauttaa uusia meshejä, sillä se osaa hitsata törmäävän topologian. DynaMesh-mallin voi aina uudelleenrakentaa ZRemesher-työkalulla alajaetuksi malliksi ja toisinpäin.



Kuva 44. DynaMesh-työkalu jakaa mallin pinnan tasaisen tiheästi ja sen voi päivittää helposti vastaamaan uutta muotoa.

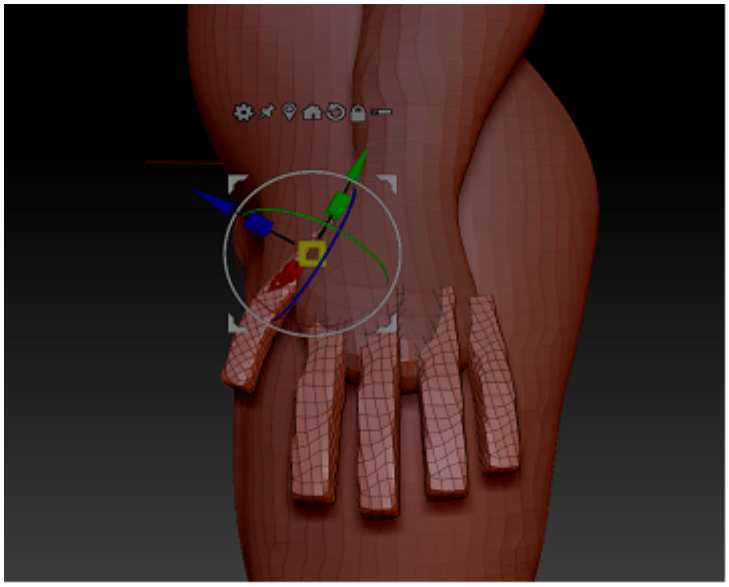
Hahmon muokkaus aloitettiin isokokoisella Move-siveltimellä, jolla saatettiin siirrellä ja muovata isoja osia mallin pinnasta kerralla. Myös Smooth- sivellin oli hyödyllinen tässä vaiheessa. Malli pidettiin edelleen niin kevyenä kuin mahdollista. Tämän vaiheen lopputulos esitetään kuvassa 45.

Kuva 45. Hahmo esitettynä ennen ja jälkeen DynaMesh-tilassa muovaamisen.

4.3.2 Käsi

Käden voi veistää monella tapaa. Opinnäytetyöprosessissa käsi haluttiin veistää alusta asti itse oppimiskokemuksen vuoksi, mutta aikaa olisi voitu säästää käyttämällä valmista käsisivellintä tai lataamalla verkosta veistämispohja.

Sormien lisääminen käteen aloitettiin liittämällä Cube 3D -kuutiomesh, josta muovailtiin sormen muotoinen ja kokoinen transformointityökaluilla. Sormi siirrettiin DynaMesh-tilaan, jotta sen geometriaa voitiin muokata helposti. Sivellinvalikosta valittiin MaskLasso-sivellin ja sormi maskattiin joka nivelen kohdalta, jotta sitä voitiin taivuttaa luonnollisempaan asentoon. Transformointityökaluilla voitiin kopioida helposti neljä sormea lisää asetellen ne suurin piirtein oikeille paikoilleen (kuva 46). Sormien väliin jätettiin mahdollisimman paljon tilaa, ettei koko hahmon DynaMesh-tilan päivittäminen sulauttaisi sormia yhteen.

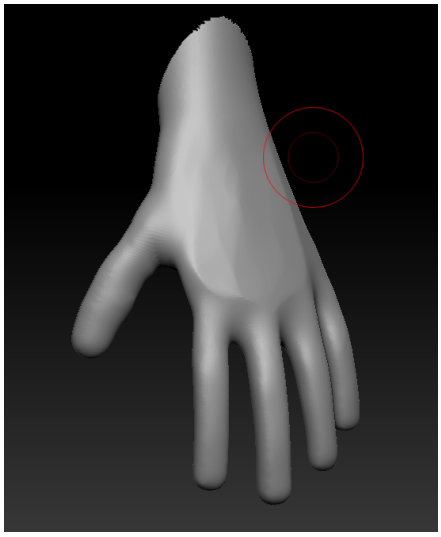


Kuva 46. Hahmon sormet.

Sormet kopioitiin hahmon toiseen käteen Mirror and Weld -komennolla. Kun peilaaminen ei jonkin vuoksi toiminut, ongelma ratkaistiin käyttämällä ensin pelkkää peilauskomentoa. Sormet olivat oma alatyökalunsa, joka yhdistettiin vartaloalatyökaluun valitsemalla Merge Down -toiminto alatyökaluvalikosta. DynaMesh-tila vaati hieman isomman resoluution, jotta sormet eivät sulautuneet toisiinsa kiinni.

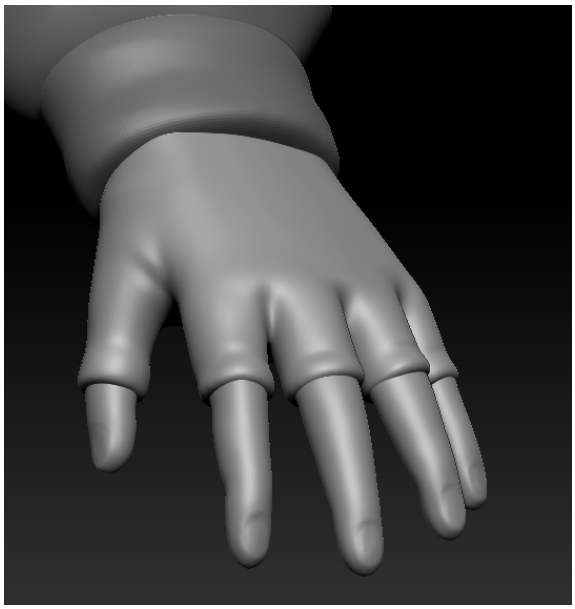
Käsi veistettiin loppuun vasta muun vartalon jälkeen, mutta selkeyden vuoksi sen työvaiheet käydään loppuun asti tämän väliotsikon alla. Muu vartalo maskattiin ja piilotettiin,

jotta voitiin keskittyä paremmin käsien veistämiseen. Kättä hiottiin enemmän kädenmuotoiseksi Smooth-, HPolish- ja Clay BuildUp -siveltimillä (kuva 47).



Kuva 47. Käden ja sormien muovaaminen oikean näköisiksi.

Hahmolle haluttiin tehdä kynsikäshanskat, joten käsi maskattiin ranteesta sormien puoliväliin asti. Kynsikäs luotiin ekstraktointitoiminnolla ja siitä veistettiin enemmän hanskannäköinen pehmentämällä sitä reunoista ja piirtämällä sormien väliin kankaan rypyt referenssikuvien mukaan. Sormiin tehtiin vielä kynnet Trim Dynamic- ja HPolish-siveltimillä litistämällä sormenpäätä ja piirtämällä kynsinauha Orb Cracks- ja Pinch -siveltimillä (kuva 48). Mitä pienemmäksi yksityiskohtien taso meni, sitä enemmän tarvittiin pinnan geometriaa.

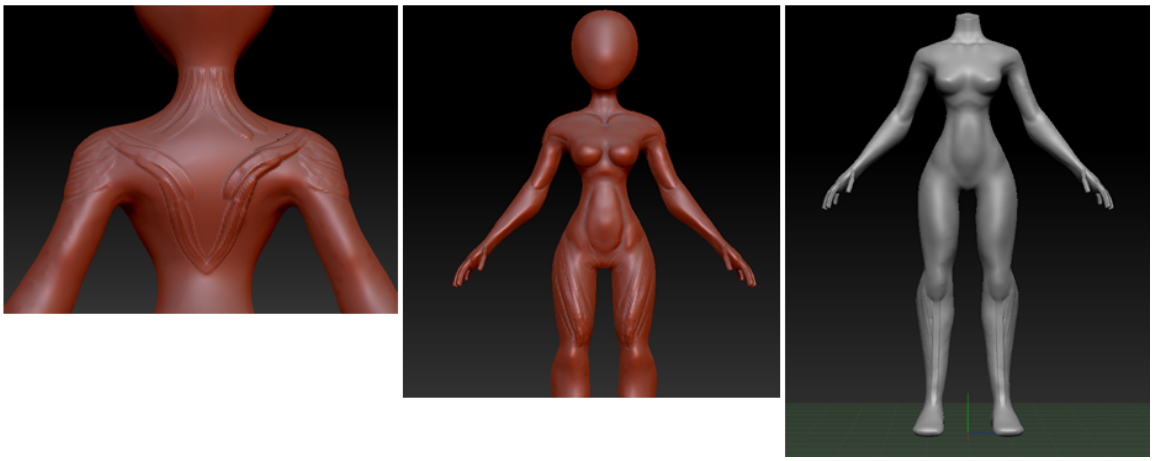


Kuva 48. Lopullinen hanska ja käsi.

4.3.3 Lihasanatomian veistäminen

Työn tavoitteena oli kasvattaa tietotaitoa veistämisestä, minkä vuoksi hahmolle koitettiin veistää lihaspohja. Prosessista oli ehkä enemmän haittaa kuin hyötyä, sillä lopullinen hahmo ei tullut olemaan lihaksikas ja liian erottuvat lihakset saivat sen näyttämään omituiselta. Oppimisprosessina lihasten veistäminen oli kuitenkin hyödyllinen askel.

Anatomiareferenssiä seuraten Clay Build Up -siveltimellä ikään kuin piirrettiin lihakset paikalleen. Syntynyt pinta tasoitettiin silottavalla siveltimellä, jotta se näyttäisi enemmän iholta. Tässä vaiheessa vaihdettiin myös punainen materiaali, joka oli jäännös ZSphere-työkalusta, perinteiseksi harmaaksi MatCap Gray -materiaaliksi. Kuvassa 49 esitetään prosessi askeleittain.



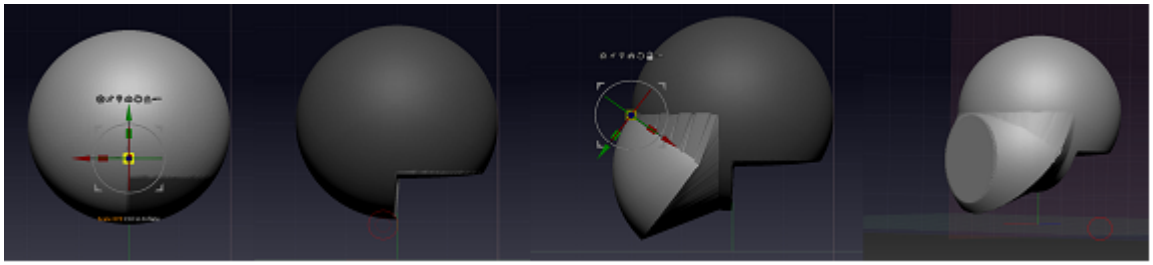
Kuva 49. Hahmon lihasanatomian veistäminen.

4.3.4 Pään veistäminen

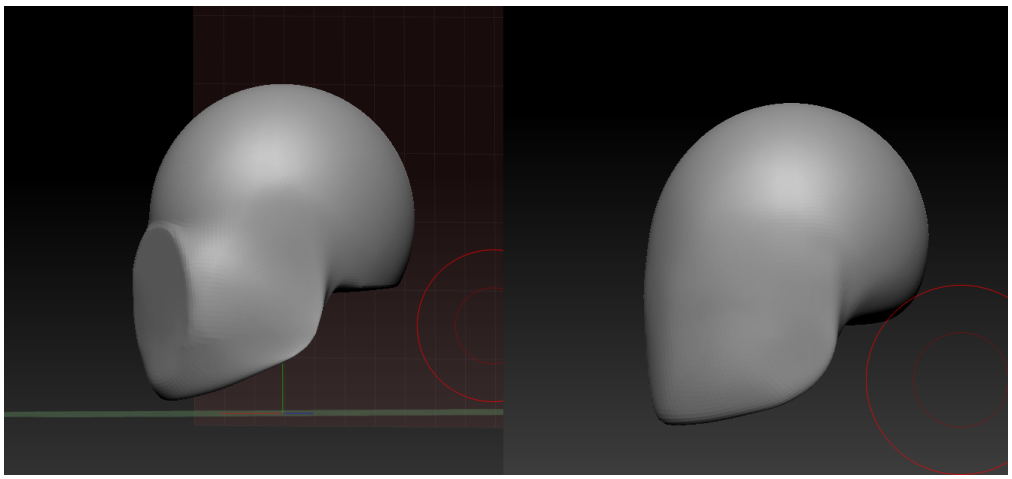
Oppimiskokemuksen vuoksi pää haluttiin veistää itse alusta asti. Apuna käytettiin Danny Macin videosarjaa "How to sculpt a stylized head in Zbrush", jossa selitettiin pään veistämisen askelet. Tutoriaalivideon mukana oli helppo seurata, vaikka aiempi kokemus kasvojen veistämisestä oli karkeasti nolla. (Mac, D. 2017.)

Työskentely aloitettiin lisäämällä uusi Sphere 3D -pallomesh, kytkemällä symmetria päälle ja piilottamalla vartaloalatyökalu. Vastaluodussa pallomeshissä oli epätoivottu topologia, joten sen pinta uudelleenrakennettiin DynaMesh-työkalulla. Päätä muovattiin maskaus-, klippaus- ja transformointityökaluilla tutoriaalivideon mukana (kuva 50). Kun

päähän saatiin lisättyä leuan ja takaraivon muoto, sitä pehmennettiin Move- ja Smooth-siveltimillä (kuva 51).

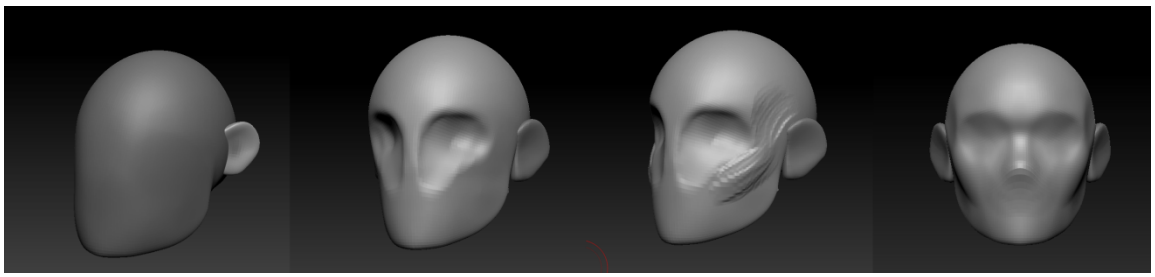


Kuva 50. Pään muodon veistäminen pallosta.



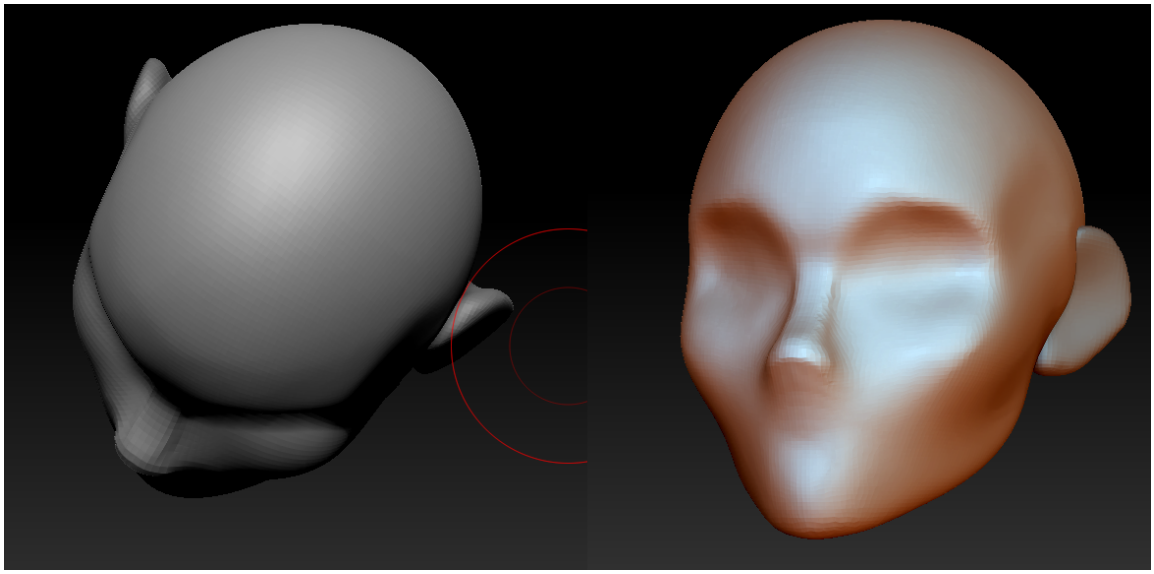
Kuva 51. Pään muodon pehmentäminen.

Kun pään muoto oli saatu hahmoteltua, alettiin kaivertamaan silmäkuoppia Standard-siveltimellä pitäen pohjassa Alt-pikanäppäintä, jolloin 3D-massan lisäämisen sijaan sivellin vähensi sitä. Poskipäiden ja ohimon muoto luonnosteltiin Clay BuildUp-siveltimellä ja nenälle luotiin pohja Standard-siveltimellä. Korvat tehtiin lisäämällä sylinterialatyökalu ja siirtämällä se transformointityökaluilla suurin piirtein kohdilleen. Sylinteri muovailtiin korvan muotoiseksi DynaMesh-tilassa. Kasvojen veistämisen työvaiheet ja korvan lisääminen esitetään kuvassa 52.



Kuva 52. Hahmon korvien lisääminen ja kasvojen hahmottelu

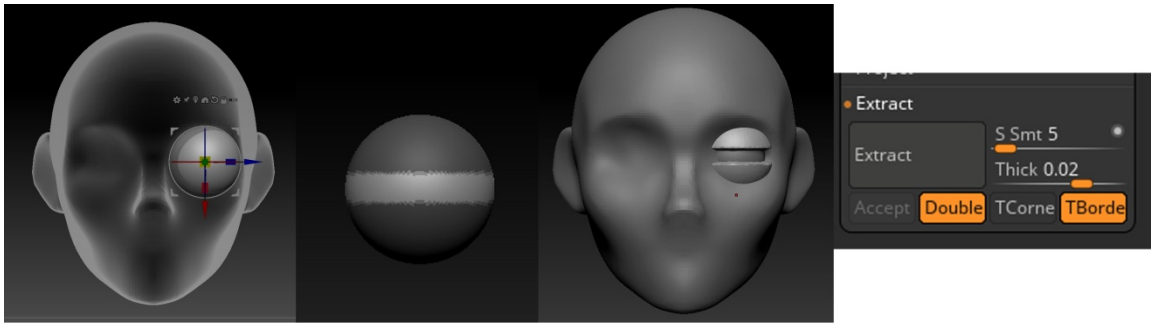
Meshin materiaalin vaihtaminen virkisti silmiä ja auttoi huomaamaan virheet mallissa paremmin. 3D-mallia kannatti myös tarkastella eri kuvakulmista ja varmistaa, että se näytti oikealta joka suunnasta. Erityisesti äärimmäiset kuvakulmat, kuten jyrkästi ylhäältä tai alhaalta, auttoivat pitämään mallin enemmän oikean muotoisena. (kuva 53)



Kuva 53. Veistoksen kriittistä tarkastelua.

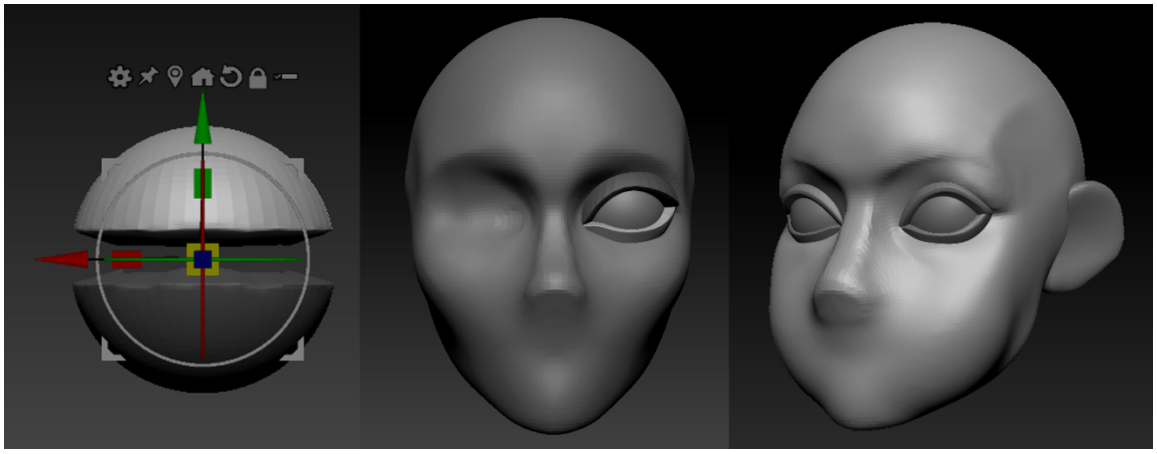
4.3.5 Silmät

Silmämuna luotiin lisäämällä Sphere 3D -pallomesh ja sijoittelemalla se silmäkuoppaan. Silmämuna maskattiin ylä- ja alareunasta niin, että keskelle jäi ohut maskaamaton suikale. Maskatusta osasta luotiin silmäluomet ekstraktointikomennolla (kuva 54).



Kuva 54. Silmäluomien ekstraktoiminen silmäkuoppaan asetellusta silmämeshistä sekä käytetyt ekstraktointiasetukset.

Seuraavaksi transformointityökalu keskitettiin uuden silmäluomimeshin keskelle. Yläsilmäluomi maskattiin ja alasilmäluomi aseteltiin silmäluomelle luonnollisempaan asentoon rotatointityökalulla ja Move-siveltimellä. Vastaava työvaihe toistettiin seuraavaksi yläsilmäluomelle. Lopuksi silmäluomet peilattiin toiselle puolelle (kuva 55).

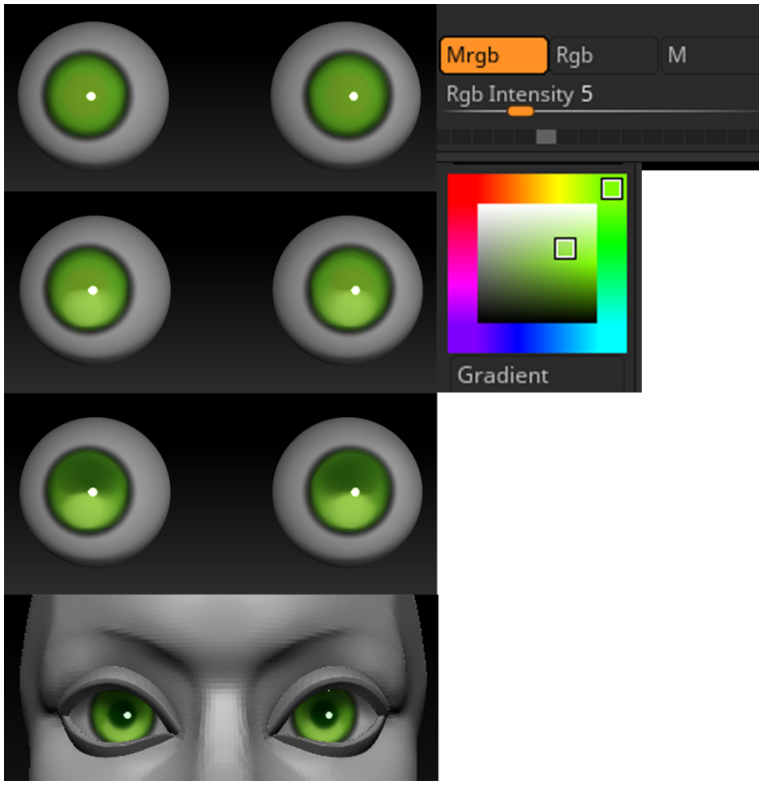


Kuva 55. Silmäluomien maskaus, muotoilu ja peilaus.

Silmille haluttiin antaa niille ominaisempi tekstuuri, sillä ne oikeannäköiset silmät opastaisivat muiden kasvonpiirteiden veistämistä paremmin. Valittiin silmämuna-alatyökalu ja täytettiin se ToyPlastic-materiaalilla niin, että käyttöliittymän Mrgb-painike oli aktivoituna. Näin silmät säilyttäisivät materiaalinsa, vaikka aktiivinen materiaali vaihtuisi myöhemmin. Seuraavaksi valittiin Paint-sivellin ja musta väri. Silmät olivat säilyttäneet alkuperäisen topologiansa siitä, kun ne oli lisätty Sphere 3D -alatyökaluna eli kun hahmon rautalankamalli asetettiin näkyväksi, topologia kulminoitui pallon huipulla. Silmämunaa käännettiin 90 astetta eteenpäin, jotta topologian osoittama keskikohta sijaitsi samassa paikassa kuin pupilli. Silmämunan pinta jaettiin polygonimaalausta varten niin, että sen polygonimäärä oli todella korkea. Sivellin sijoiteltiin suurin piirtein keskelle silmää ja siihen painettiin tumma ympyrä (kuva 56).

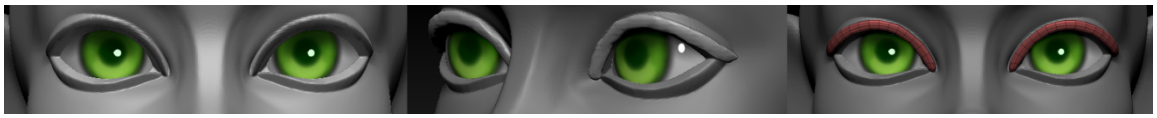
Kuva 56. Silmät täytettiin kiiltävällä valkoisella materiaalilla ja iiris maalattiin.

Silmästä tehtiin realistisempi maalamalla vihreällä, hieman pienemmällä siveltimellä toinen ympyrä edellisen keskelle. Oikean silmän kolmiulotteista muotoa jäljiteltiin värittämällä vihreän iiriksen yläpuoli tummemmaksi ja alapuoli vaaleammaksi, ikään kuin valo osuisi siihen. Lopuksi pienellä siveltimellä piirrettiin pupilli (kuva 57).



Kuva 57. Silmien maalaamisen työvaiheet ja lopulliset silmämunat.

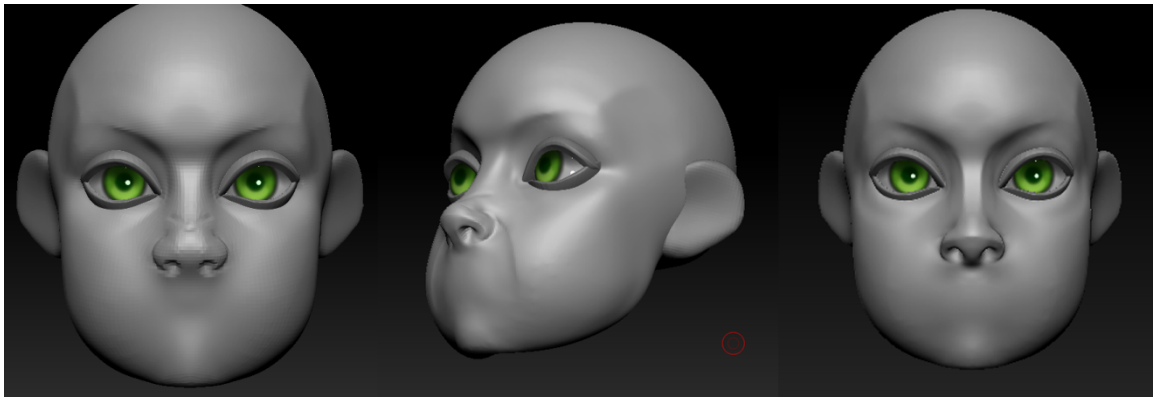
Hahmolle tehtiin yksinkertaiset silmäripset maskaamalla silmäluomen reuna ja ekstraktoimalla siitä uusi alatyökalu. Ripsiä muovattiin DynaMesh-tilassa ennen niiden retopologisoimista ZRemesher-toiminnolla. Ne täytettiin lopuksi tummanruskealla värillä (kuva 58).



Kuva 58. Ripset ekstraktoitiin silmäluomista.

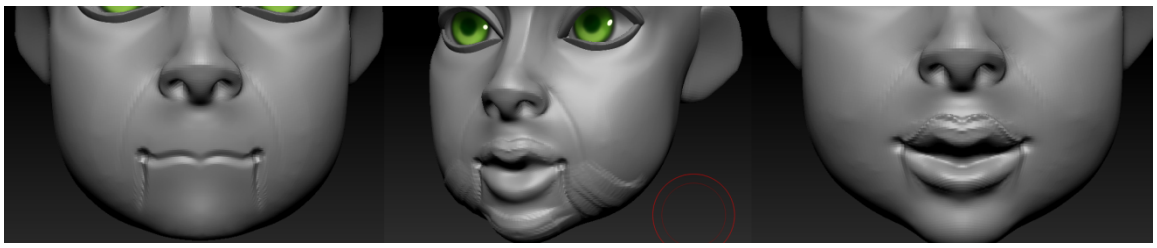
4.3.6 Nenä ja suu

Nenän veistäminen onnistui helposti tutoriaalivideon avulla, sillä nenän anatomia on melko yksinkertainen. Muoto rakennettiin Clay BuildUp- ja Smooth-siveltimillä ja sieraimet kaiverrettiin Standard-siveltimellä. Hahmosta haluttiin hieman tyylitelty, joten nenän sillan ja kärjen tasoittaminen HPolish-siveltimellä auttoi saavuttamaan toivotun ulkoasun (kuva 59).



Kuva 59. Nenän veistäminen.

Suun veistäminen aloitettiin piirtämällä niin kutsutut marionettiviivat sekä huulten kaari ja rakentamalla niiden ympärille Clay BuildUp -siveltimellä lihasanatomiaa (kuva 60). Suu päätettiin veistää suljetuksi, sillä hahmolle ei tulisi kasvoanimaatioita.

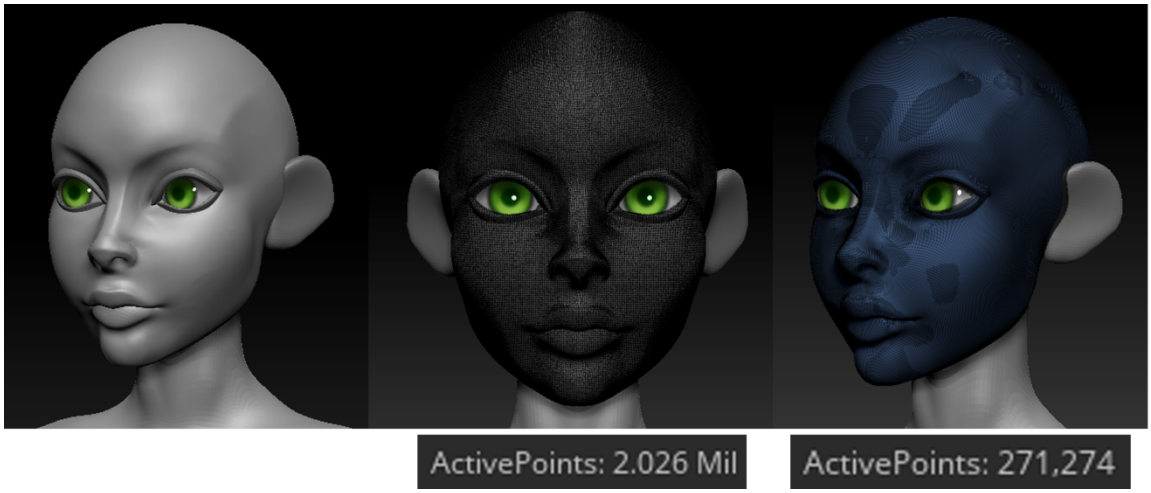


Kuva 60. Suun anatomian veistäminen.

4.3.7 Autoretopologia ja palautteen kysyminen

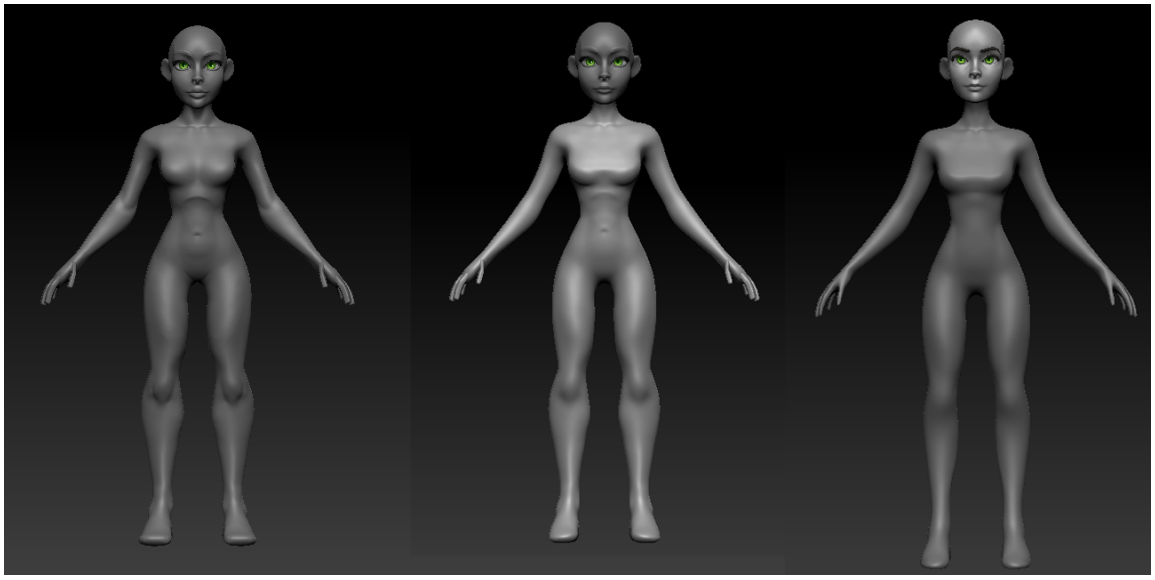
Kasvojen anatomia alkoi olla hyvässä kunnossa. DynaMesh-malli oli melko raskas, mutta se voitiin nyt retopologisoida ZRemesher-toiminnolla, jolla verteksimäärä saatiin pudotettua kahdesta miljoonasta 270 000:n verteksiin. Ennen retopologisointia DynaMesh-päästä tehtiin duplikaatti, jotta uuteen päähän voitaisiin projisoida retopologiaprosessis-

sa katoavat yksityiskohdat. Uuden pään pinta jaettiin niin, että siinä oli saman verran tai enemmän polygoneja kuin vanhassa päässä, kaikki muut alatyökalut paitsi päät piilotettiin ja yksityiskohdat projisointiin Project All -toiminnolla. Projisoinnin jälkeen voitiin siirtyä takaisin matalampaan resoluutioon ja yksityiskohtien taso säilyi silti samana (kuva 61).

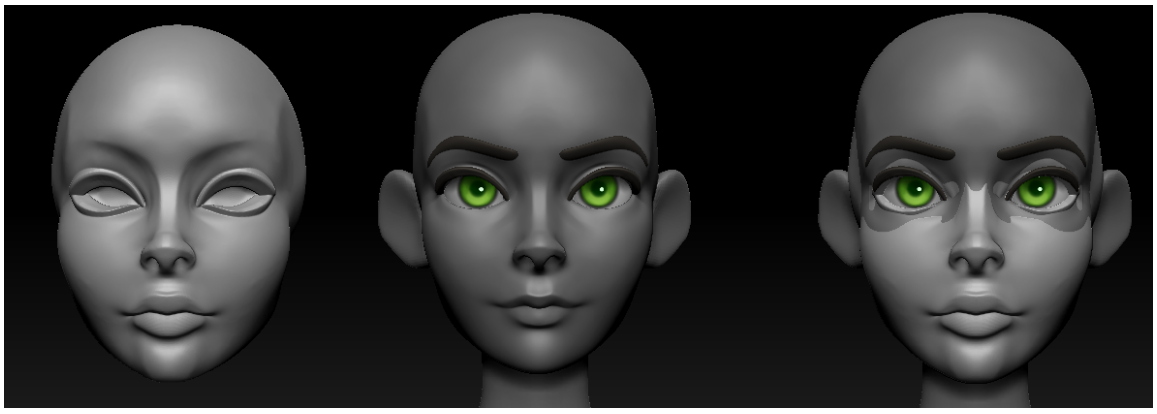


Kuva 61. Veistetty pää ja sen pinnan topologia ennen ja jälkeen autoretopologisoinnin.

Palautteen kysyminen koetaan usein pelottavaksi, mutta se on yksi tehokkaimmista tavoista saada näkyvyyttä ja kasvaa artistina. Hahmo näytti oudolta, mutta omat silmät olivat prosessin aikana sokaistuneet sille, joten työstä pyydettiin palautetta ystäviltä ja kollegoilta. Heidän mukaansa hahmon vartalo ei ollut tarpeeksi tyylitelty ja se sai ison pään näyttämään hassulta. Jalkaterät olivat liian pienet ja säärissä oli epäluonnollinen kurvi, joka ei ollut anatomisesti korrekti. Hahmon kasvot näyttivät etenkin nenän ja suun alueelta oudoilta ja leuan ja takaraivon pitkä välietäisyys sai hahmon kaulan näyttämään lyhyeltä. Työhön tehtiin palautteen mukaiset korjausmuutokset, joiden jälkeen hahmo näytti huomattavasti miellyttävämmältä (kuvat 62 ja 63).



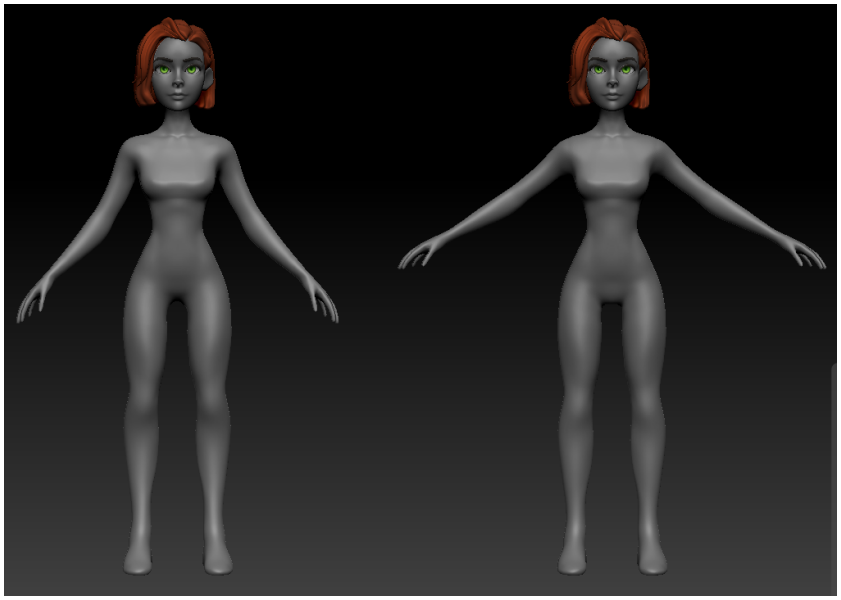
Kuva 62. Hahmon vartaloa korjattiin palautteen mukaisesti.



Kuva 63. Hahmon kasvoja korjattiin ja niistä koitettiin poistaa epämiellyttävä outolaakso-vaikutelma. Järjestyksessä vasemmalta oikealle vanhat kasvot, uudet kasvot ja vanhat ja uudet kasvot päällekkäin.

4.3.8 Vaatteet

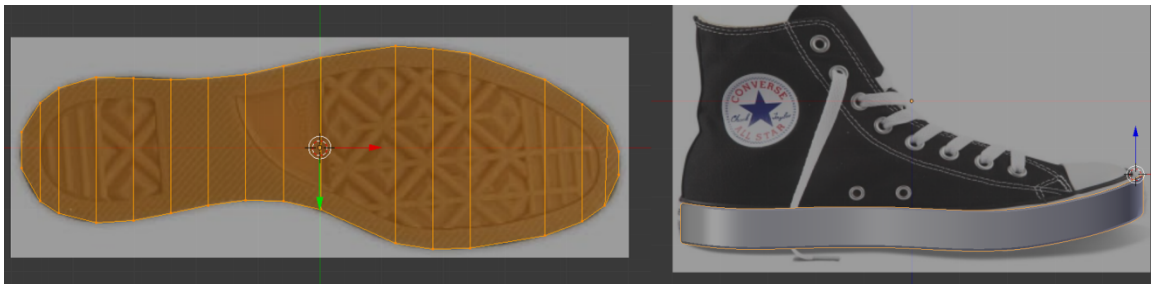
Ennen vaatteiden veistämistä haluttiin korjata käsien asentoa niin, että kainaloihin jäisi paremmin tilaa veistää huppari. Hahmo maskattiin olkapäitä myöten ja valittiin transformointityökalu. Työkalu siirrettiin lukitsemattomassa tilassa hahmon olkapäänivelen kohdalle, jossa se lukittiin ja käsivartta käännettiin, kunnes asento näytti tilavammalta (kuva 64). Syntyneet vääristymät olkapäihin korjattiin ja käsivartta suoristettiin.



Kuva 64. Hahmon käsien asento ennen ja jälkeen muutosten.

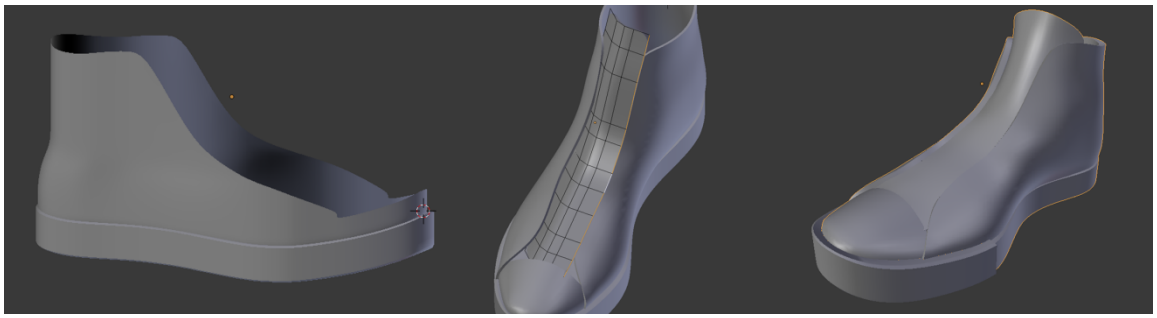
4.3.9 Kengän mallintaminen ja veistäminen

Hahmon jalkaterä oli melkoinen möhkäle eikä se tuntunut hyvältä pohjalta aloittaa kengän veistämistä. Hahmolle haluttiin tehdä tennarit, joiden muoto on melko tasainen ja tarkka. Koettiin, että kengälle olisi helpompi tehdä pohja mallinnusohjelmassa perinteisesti mallintamalla. Verkosta etsittiin referenssikuvat tennarista ja ne tuotiin Blender-mallinnusohjelmaan. Kengän pohjaa alettiin mallintaa ekstruusiomallintamalla, joka oli nopeaa ja helppoa. Kun kengänpohjan siluetti oli valmis, sille annettiin vertikaalista paksuutta ekstruusoimalla sitä ylöspäin ja sen pinta pyöristettiin pinnanjakamismodifikaattorilla. Kengät reunat pitivät muotonsa modifikaattorista huolimatta, sillä ne merkittiin Blenderin edge crease -komennolla (kuva 65).



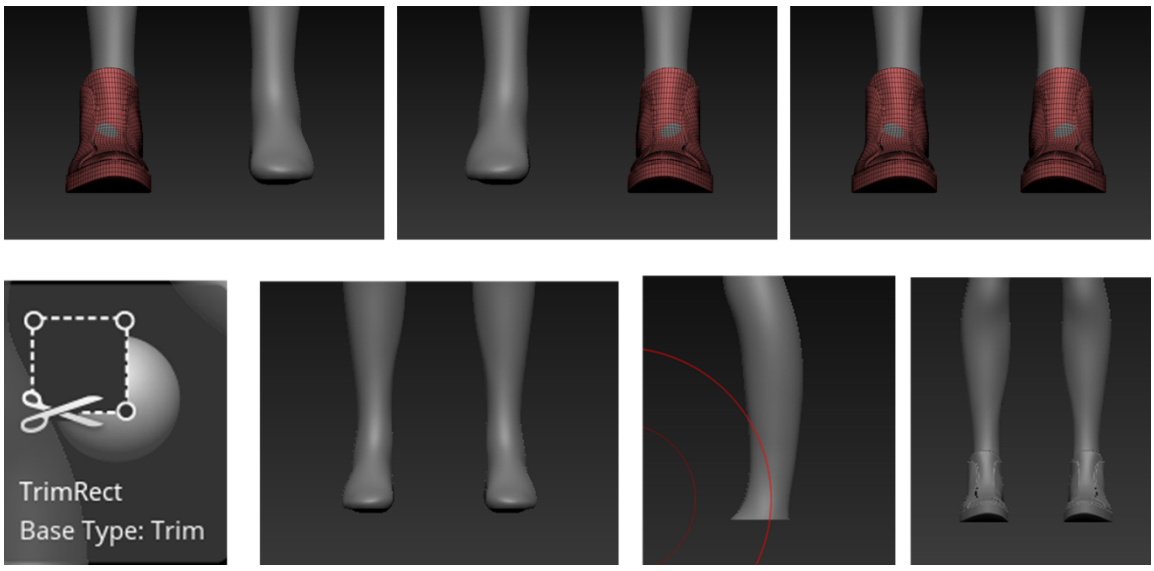
Kuva 65. Kengänpohjan mallinnus aloitettiin ekstruusiomallintamalla mallikuvan mukaan.

Loput kengästä mallinnettiin duplikoimalla kengänpohja ja ekstruusiomallintamalla sitä edelleen ylöspäin. Oikeanlaisesta topologiasta ei oltu kovin huolissaan, sillä se voitaisiin rakentaa ZBrushissa uudelleen. Kengän kangasosille annettiin paksuutta solidify-modifikaattorilla. Kenkää ei koettu tarpeelliseksi viedä kuvassa 66 esitettyä tasoa pidemmälle, sillä prosessia olisi helpompi jatkaa ZBrushissa. Kenkä tuotiin ulos Blenderistä OBJ -tiedostona.



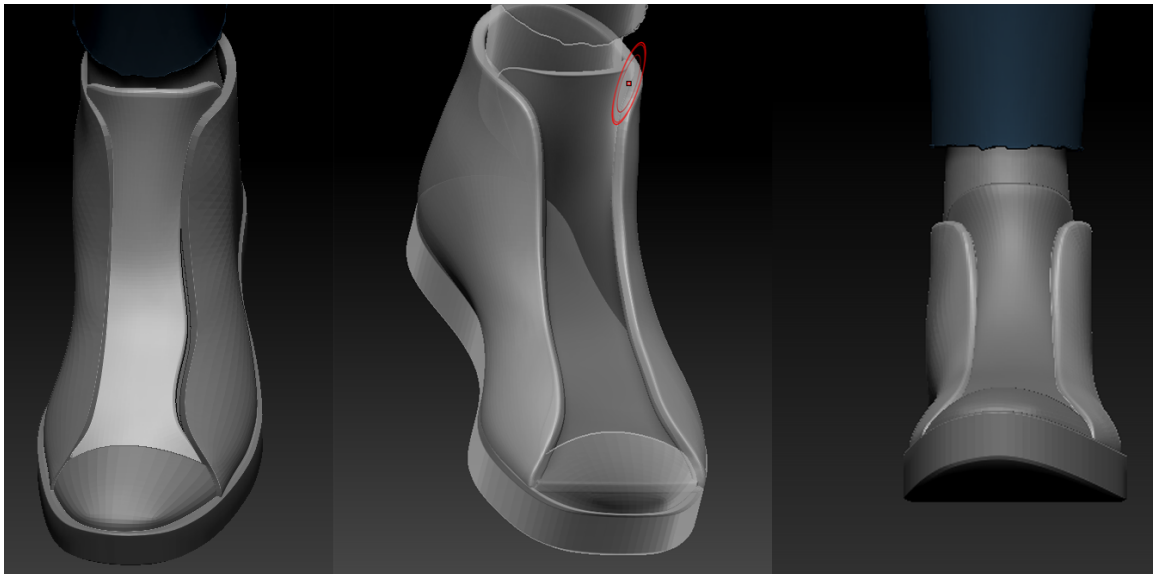
Kuva 66. Kengän mallinnusprosessia ei viety tämän pidemmälle Blenderissä.

Seuraavaksi kenkä vietiin ZBrushiin. Se oli hyvin iso suhteessa hahmoon, joten sitä skaalattiin ja se siirreltiin oikealle paikalleen jalkaterän päälle. Kammottava topologia korjattiin ZRemesher-työkalulla. Molempiin jalkoihin lisättiin kengät peilaamalla Mirror and Weld -toiminnolla. Hahmon jalkaterä pisti läpi kengästä eikä sitä enää tarvittu, joten jalkaterät katkaistiin Trim-siveltimellä (kuva 67).



Kuva 67. Ylärivillä on havainnollistettuna kengän peilaaminen. Alarivillä näkyy jalkaterien pätkäiseminen Trim-siveltimellä ja kenkien ulkoasu sen jälkeen.

Seuraavaksi kenkien muotoa muokattiin ZBrushissa. Kengän kankaan ja läpän väliin jäi kiero reikä, joka suoristettiin ZBrushin Move-siveltimellä ja jota siloiteltiin HPolish-siveltimellä (kuva 68). HPolish-sivellin on loistava väline, kun halutaan tehdä tyyliteltyä jälkeä, sillä sillä on helppo veistää tasaisia pintoja ja teräviä reunoja.



Kuva 68. Kengän läpän korjaaminen ZBrushissa.

Seuraavaksi kengännauhoille lisättiin kulkutiet IMM-SpaceShip-siveltimen alla löytyvällä PM3D_Ring3D_1-työkalulla. Tosielämän pitkävartisessa Converse-tennarissa on seitsemän reikää nauhoille, mutta yksityiskohdista haluttiin tehdä hieman isommat, jotta ne näkyisivät pienellä mobiililaitteen ruudulla paremmin eli reikiä lisättiin vain viisi. Kun torukset oli piirretty paikalleen, ne irrotettiin kengän alatyökalusta. Ne aseteltiin paremmin paikoilleen Move Topological -siveltimellä ja niihin lisättiin geometriaa jakamalla pinta. Kenkien kankaaseen kaiverrettiin vastaavat reiät Standard-siveltimellä. Lopuksi torukset peilattiin toiseenkin kenkään (kuva 69).

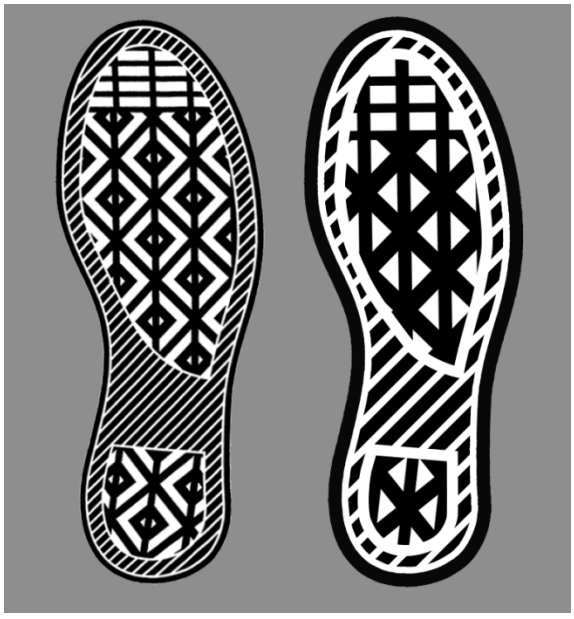
Kuva 69. Kengännauhojen reiät paikoillaan.

Kengännauhat tehtiin ZBrushissa CurveSnapStrap-siveltimellä (kuva 70). Nauhat vedettiin yksitellen ja niitä siirreltiin paremmin paikoilleen Move-siveltimellä. Kun kaikki nauhat oli piirretty, ne erotettiin omaksi alatyökalukseen ja niiden pinnan geometria jaettiin. Kengän läppään veistettiin hienovaraiset rypytykset kuvaamaan kankaan kanssakäymistä kengännauhojen kanssa. Lopuksi aukinaisten kengännauhojen päät piirrettiin IMM Curve -siveltimen alta löytyvällä Shoe Lace -siveltimellä. Pyöreiden nauhojen muotoa litistettiin Trim Dynamic- ja Polish-siveltimillä. Lopulliset kengät näkyvät kuvassa 71.

Kuva 70. Kengännauhat luotiin "CurveSnapStrap"-siveltimellä.

Kuva 71. Lopulliset kengännauhat ja kengän kangas.

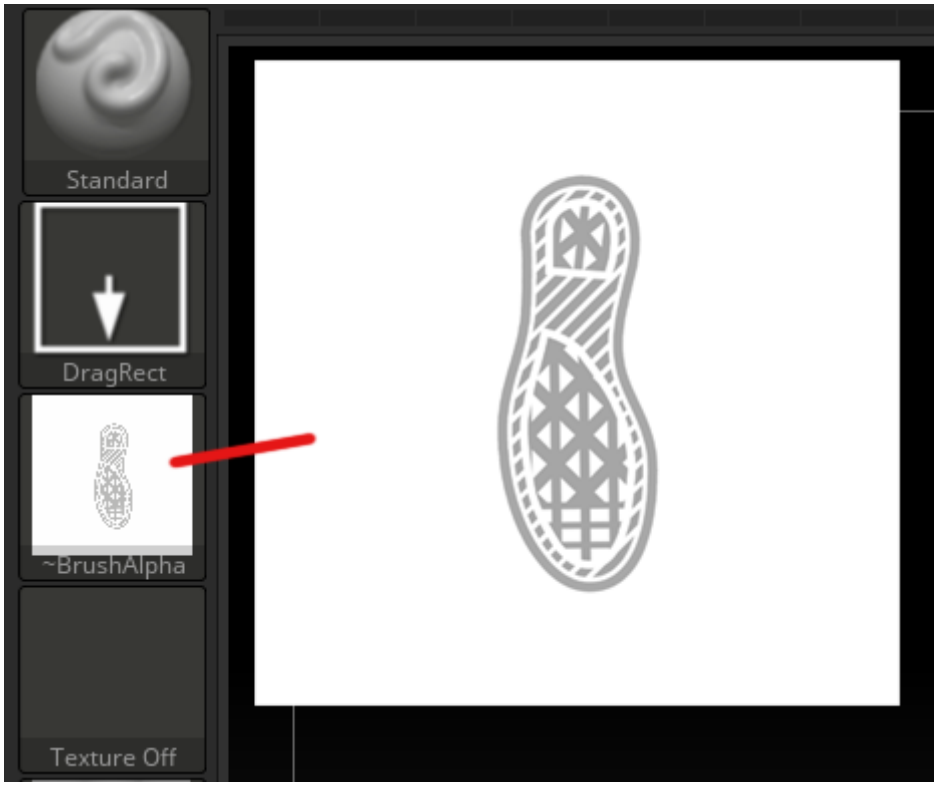
Kengänpohjissa haluttiin kokeilla ZBrushin siveltimen alfaominaisuutta. Ensin tehtiin kustomoitu mustavalkoinen alfatekstuuri kengänpohjasta, jotta se voitaisiin painaa kengänpohjaan kuin leimasimesta. Pohjan kuviosta haluttiin tehdä samanlainen kuin tosielämän Converse-tennareissa, mutta koska hahmo tulisi mobiilipelikäyttöön, kengänpohjan yksityiskohtia päätettiin yksinkertaistaa. Kaikkia pohjan yksittäisiä designelementtejä suurennettiin Photoshopissa (kuva 72).



Kuva 72. Oikean Converse-tennarin pohjasta muokattiin pelkistetympi versio.

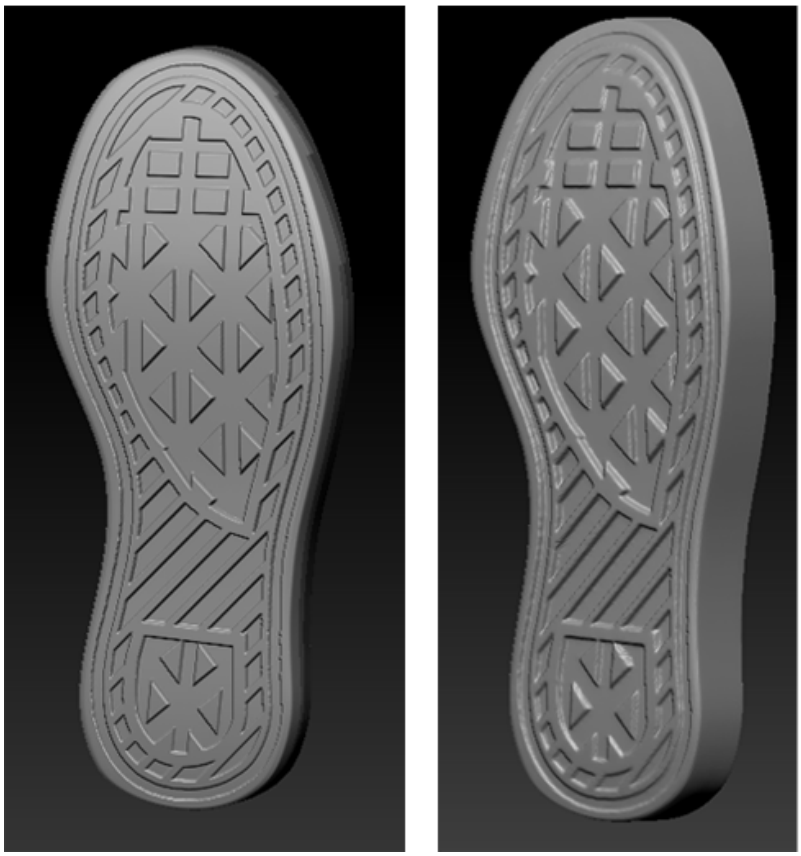
Kuvatiedosto kengänpohjasta tallennettiin neliönmuotoisena. Standard-siveltimen asetuksia muutettiin niin, että siveltimen alfaksi valittiin kuvatiedosto kengänpohjasta ja si-

veltimen vedoksi muutettiin DragRect-valinta (kuva 73). Lopuksi kuvio painettiin kengänpohjaan. Prosessi vaati paljon geometriaa, sillä meshin pinnalla oli oltava tarpeeksi polygoniresursseja, jotta monimutkainen kuvio voitiin toistaa.

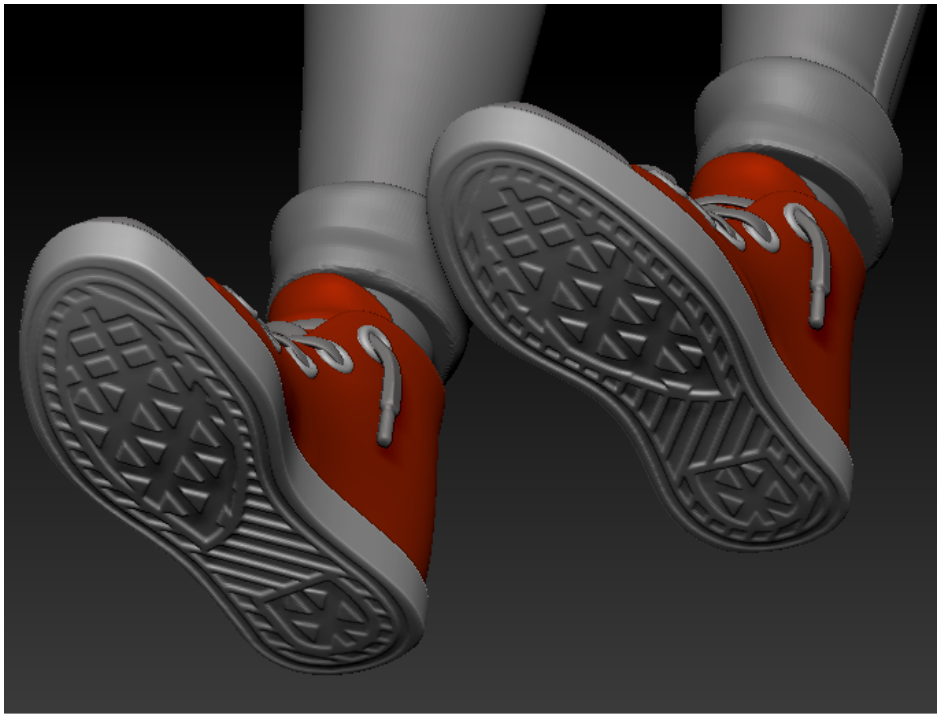


Kuva 73. Siveltimen asetukset, kun pohjan kuvio "painettiin" kengänpohjaan.

Lopuksi kengänpohjat retopologisoitiin ZRemesher-työkalulla, sillä niiden verteksimäärä oli todella korkea. ZBrushin autoretopologia ei osannut toistaa muotoja tarpeeksi tarkasti (kuva 74), joten ennen retopologisointia kengänpohjasta duplikoitiin tarkka DynaMesh-versio, josta yksityiskohdat projisoitiin retopologisoituun meshiin. Yksityiskohdat suttuuntuivat silti hieman, mutta niistä saatiin miellyttävän näköiset Polish By Features -toiminnolla (kuva 75). Lopullisten kengänpohjien polygonimäärä saatiin näin matalammaksi.



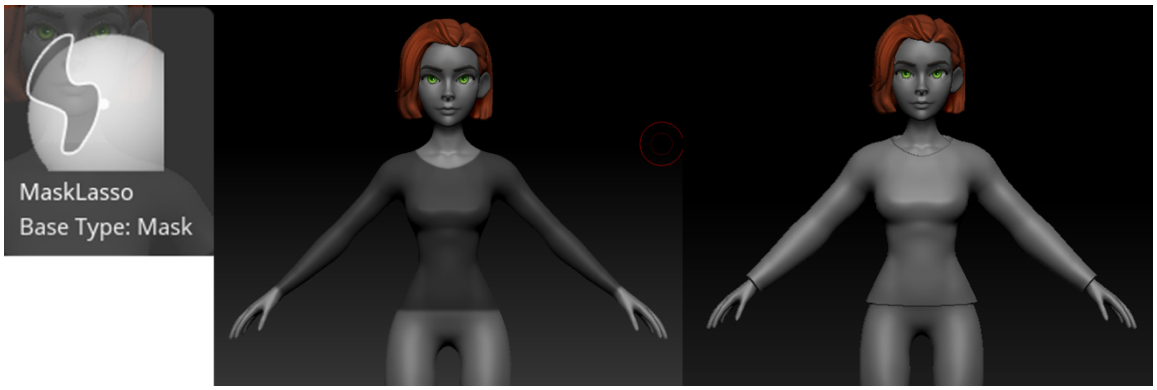
Kuva 74. Kengänpohjien yksityiskohtien terävyys ennen autoretopologisaatiota (vas.) ja heti autoretopologisoinnin jälkeen.



Kuva 75. Lopulliset kengänpohjat, joita siistittiin Polish by Features -toiminnolla.

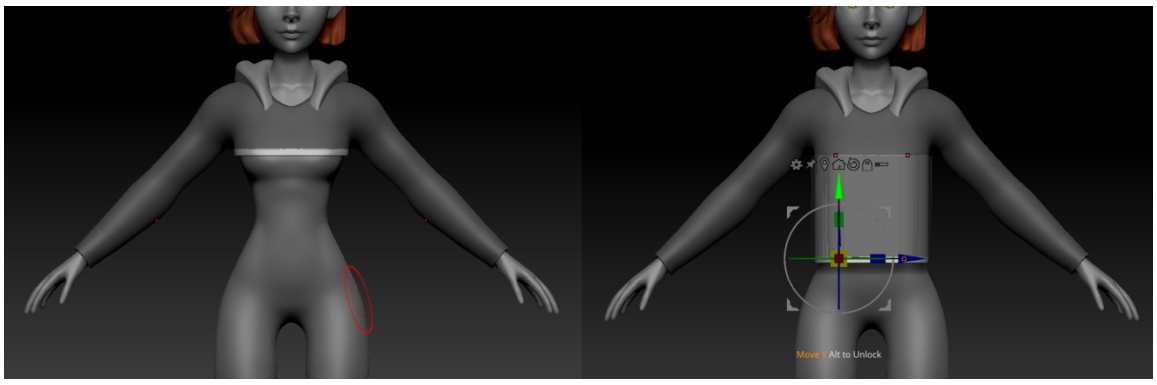
4.3.10 Vaatteet

Kaikki hahmon vaatteet luotiin ensin maskaamalla hahmon vartalosta se osa, jonka vaate peittäisi. Esimerkiksi hupparin veistäminen aloitettiin maskaamalla hahmo ranteista lantioon ja kaula-aukkoon asti. Sopiva ekstraktointipaksuus paksuus etsittiin ZBrushin Extract-komennon esikatselutilassa ja kun se näytti hyvältä, uusi mesh hyväksyttiin ja ZBrush loi uuden alatyökalun (kuva 76). Hahmon farkut luotiin samalla metodilla maskaamalla vartalo vyötäröstä nilkkoihin.



Kuva 76. Hupparin alueen maskaaminen ja ekstraktointi.

Hupparista päätettiin poistaa Trim-siveltimellä vatsan ja selän peittävä osuus, sillä paidasta haluttiinkin löysä eikä ihonmyötäinen. Hupparista maskattiin kaikki muu paitsi sen alareuna rintakehän huipulla, josta löysä kangas laskeutuisi (kuva 77). Alareunaa vedettiin alaspäin transformointityökalulla. DynaMesh-tila päivitettiin, jolloin uusi mesh oli topologialtaan veistettävissä. Lopputulos näytti enemmän vaatekappaleelta ja vähemmän paksulta iholta.

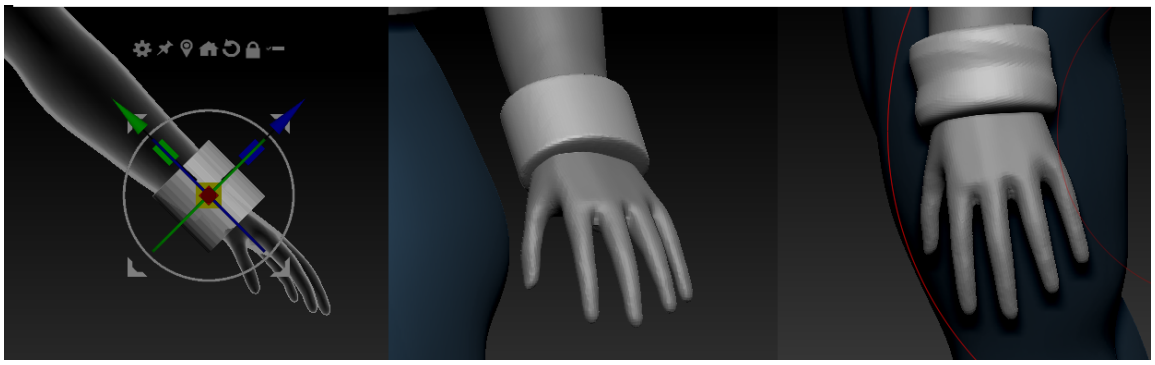


Kuva 77. Hupparista maskattiin kaikki muu kuin torson alareuna (vas.) ja sitä vedettiin alaspäin niin, että se laskeutui enemmän kuin oikea kangas.

Sitten hupparia alettiin muovaamaan enemmän hupparin näköiseksi Move-, Inflate-, Smooth- ja Clay BuildUp -siveltimillä (kuva 78). Hihojen päitä taputeltiin varovasti Inflate-siveltimellä, jolloin ne saivat paksuutta ja kankaan paino laskostui kohti ranteita. Se näytti luonnollisemmalta, kun hihoihin lisättiin resorit, jotka pysäyttivät kankaan etenemisen.

Kuva 78. Hupparin muodon hakeminen.

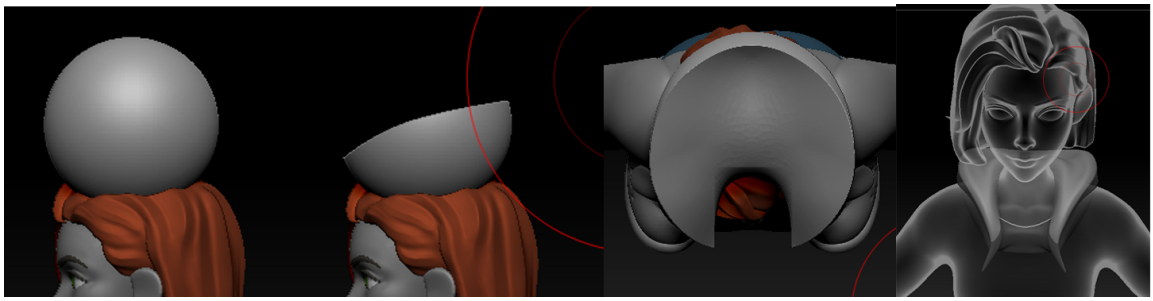
Hupparin resori luotiin Cylinder3D-sylinterimeshistä, joka skaalattiin oikean kokoiseksi ja siirrettiin transformointityökaluilla ranteen kohdalle. Sylinteriä muovailtiin enemmän kangasmaiseksi Smooth- ja Move-siveltimillä DynaMesh-tilassa. Resoluutiota nostettiin ja kankaaseen piirrettiin tyylitellyt laskokset isokokoisella Orb_Cracks-siveltimellä. Farkun lahkeet tehtiin myöhemmin samalla metodilla. Prosessi esitetään kuvassa 79.



Kuva 79. Hupparin resorin veistäminen vaiheittain.

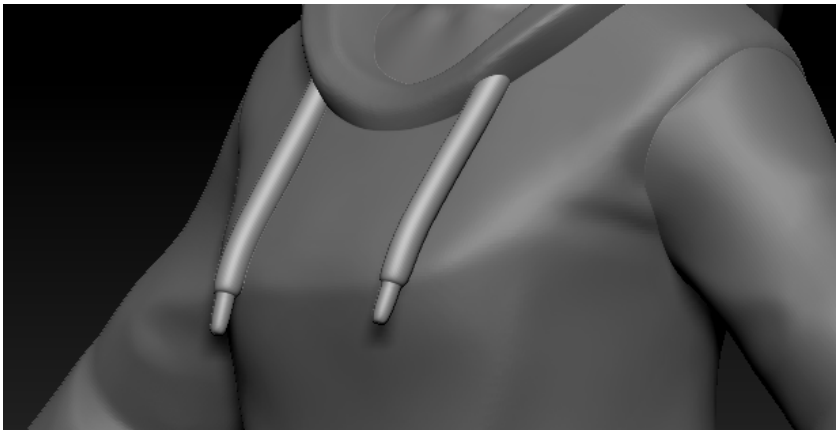
Hupun veistäminen aloitettiin lisäämällä uusi Sphere 3D -pallomesh, joka leikattiin puoliksi TrimCurve-siveltimellä. Huppuun työnnettiin kaula-aukko ja sitä muovailtiin Move-

siveltimellä, kunnes se muistutti enemmän huppua (kuva 80). Kankaan laskokset piirrettiin Orb_Cracks-siveltimellä.



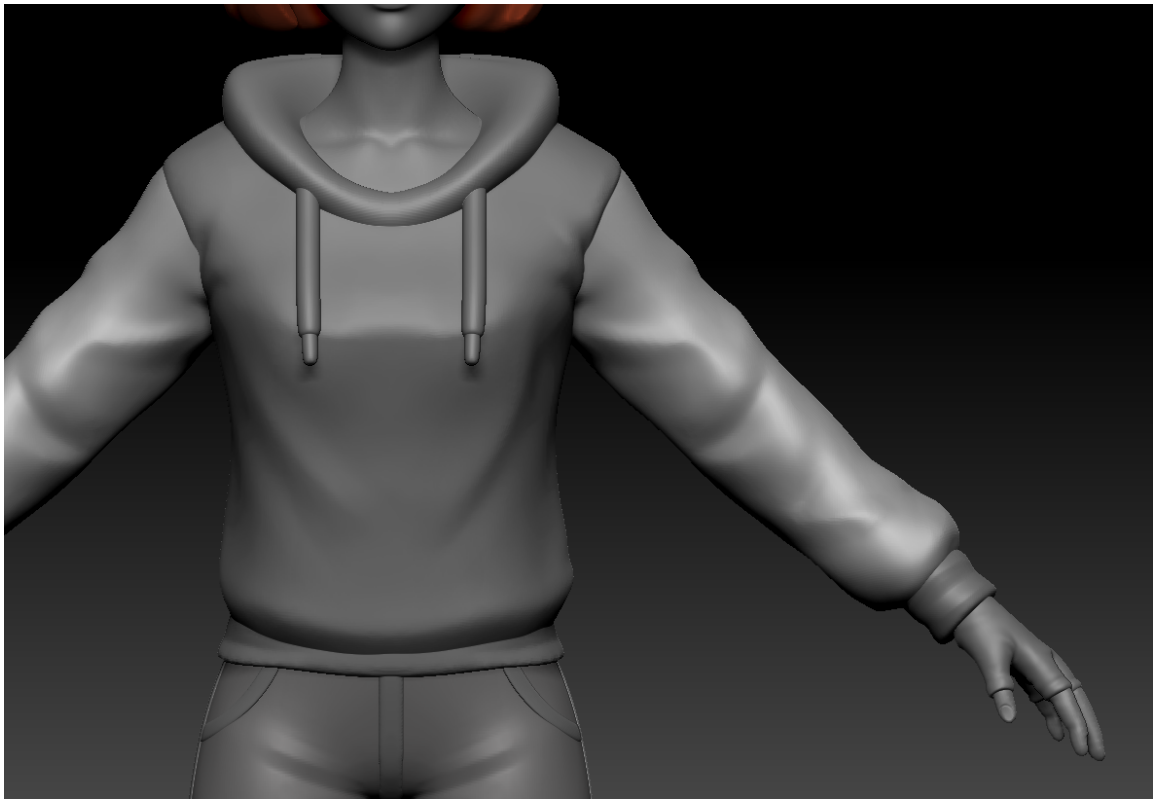
Kuva 80. Hupun veistäminen Sphere 3D-pallomeshistä.

Hupparin narut veistettiin samalla IMM Curve Shoe Lace -siveltimellä, jolla tehtiin myös tennarin narujen päät. Niistä haluttiin tehdä liioitellun suuret, jotta ne näkyisivät paremmin pieneltä näytöltä. Naruista tehtiin myös keskenään eripituiset todentuntuisemman ja tyylitellymmän lopputuloksen saavuttamiseksi (kuva 81).



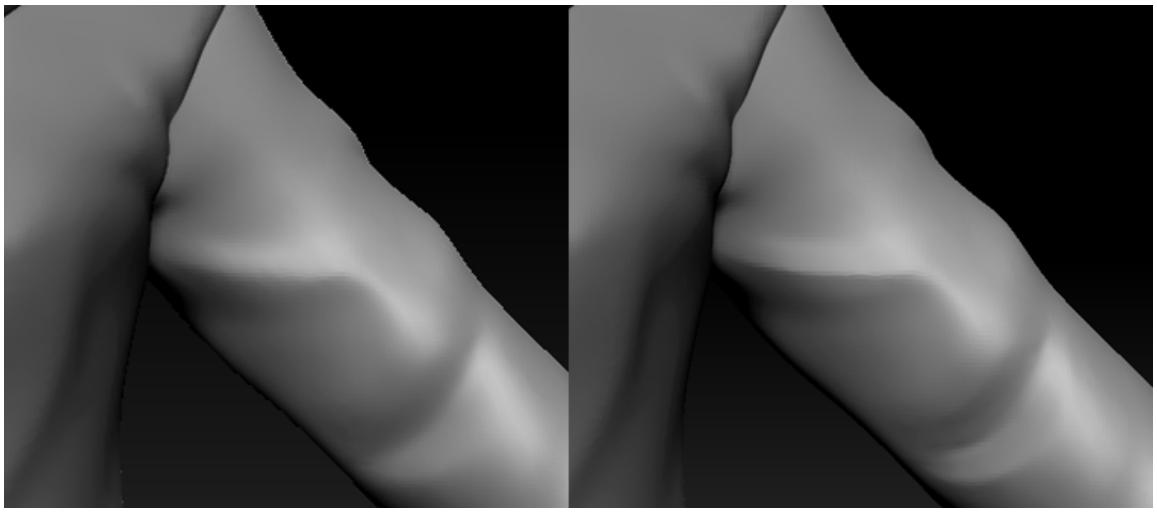
Kuva 81. Hupparin narut.

Seuraavaksi hupparista maskattiin hihat ja ne erotettiin omaksi alatyökalukseen Split Masked -komennolla, sillä hupparin torsosta aiottiin tehdä epäsymmetrinen ja hihoista symmetriset. Torson alatyökalun symmetria käännettiin pois päältä ja ryppyjä alettiin veistämään referenssikuvien mukaan. Prosessin olisi voinut tehdä tehokkaammin, jos olisi ollut tarkka mielikuva siitä, millaista kangasta hupparin haluttiin olevan - kun käytettiin muutamaa eri kuvaa referenssinä, oli jonkin verran ongelmia pitää kankaan paksuus ja jäykkyys yhtenäisenä. Suurimman menestyksen ryppyjä veistäessä tuotti Standard- ja Clay BuildUp -siveltimien rohkea käyttö ja yksityiskohtien silottaminen Smooth-siveltimellä. Rypyt nipistettiin enemmän oikean kankaan muotoon Pinch-siveltimellä ja kankaan isot pinnat tasattiin HPolish-siveltimellä. Uusi versio on esitetty kuvassa 82.



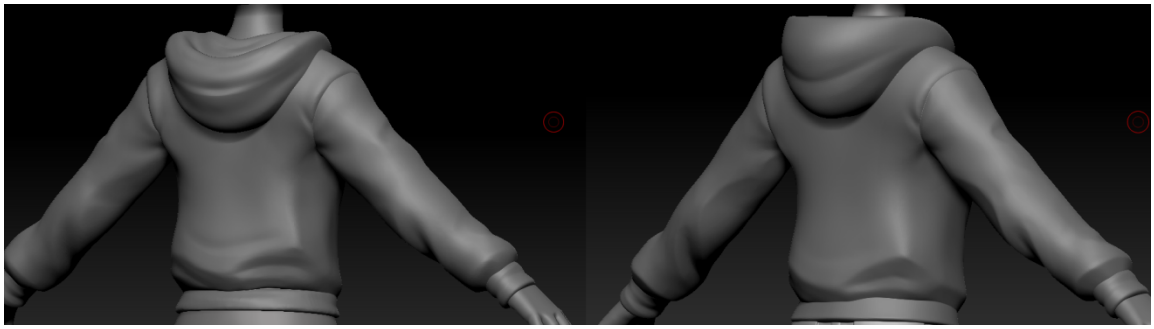
Kuva 82. Yksityiskohtaisempi versio hupparista.

Kun kangas alkoi näyttää oikealta, topologia uudelleenrakennettiin ZRemesher-toiminnolla. Ryppyjä hiottiin tarkemmin pienellä HPolish-siveltimellä tyylitellyn ulkoasun saavuttamiseksi (kuva 83). Joitain pieniä ryppyjä veistettiin Orb_Cracks-siveltimellä sekä lisäys- että vähentämistilassa.



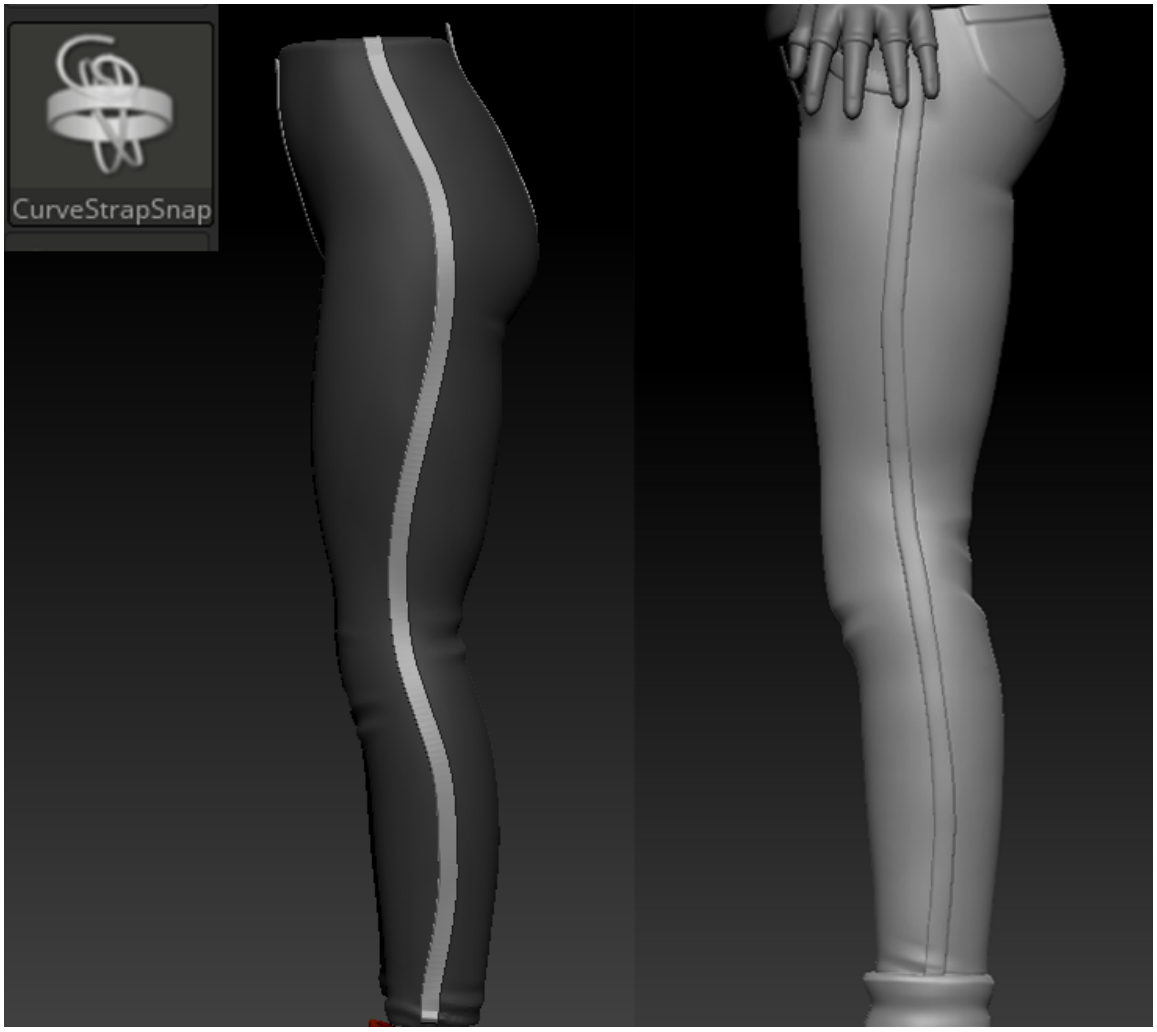
Kuva 83. Rypyt tyyliteltiin HPolish-siveltimellä.

Vaatetukseen ei haluttu liikaa pieniä yksityiskohtia. Hupparin takaosa, joka oli veistetty melko tarkasti valokuvan mukaan, oli näyttävä mutta tuntui liian "meluisalta" yksityiskohtineen, joten tehtiin päätös tasoittaa yksityiskohtia pois Smooth- ja HPolish-siveltimillä (kuva 84). Myös huppu koki saman kohtelun.



Kuva 84. Hupparin takaosa ennen (vas.) ja jälkeen yksityiskohtien tason karsimisen.

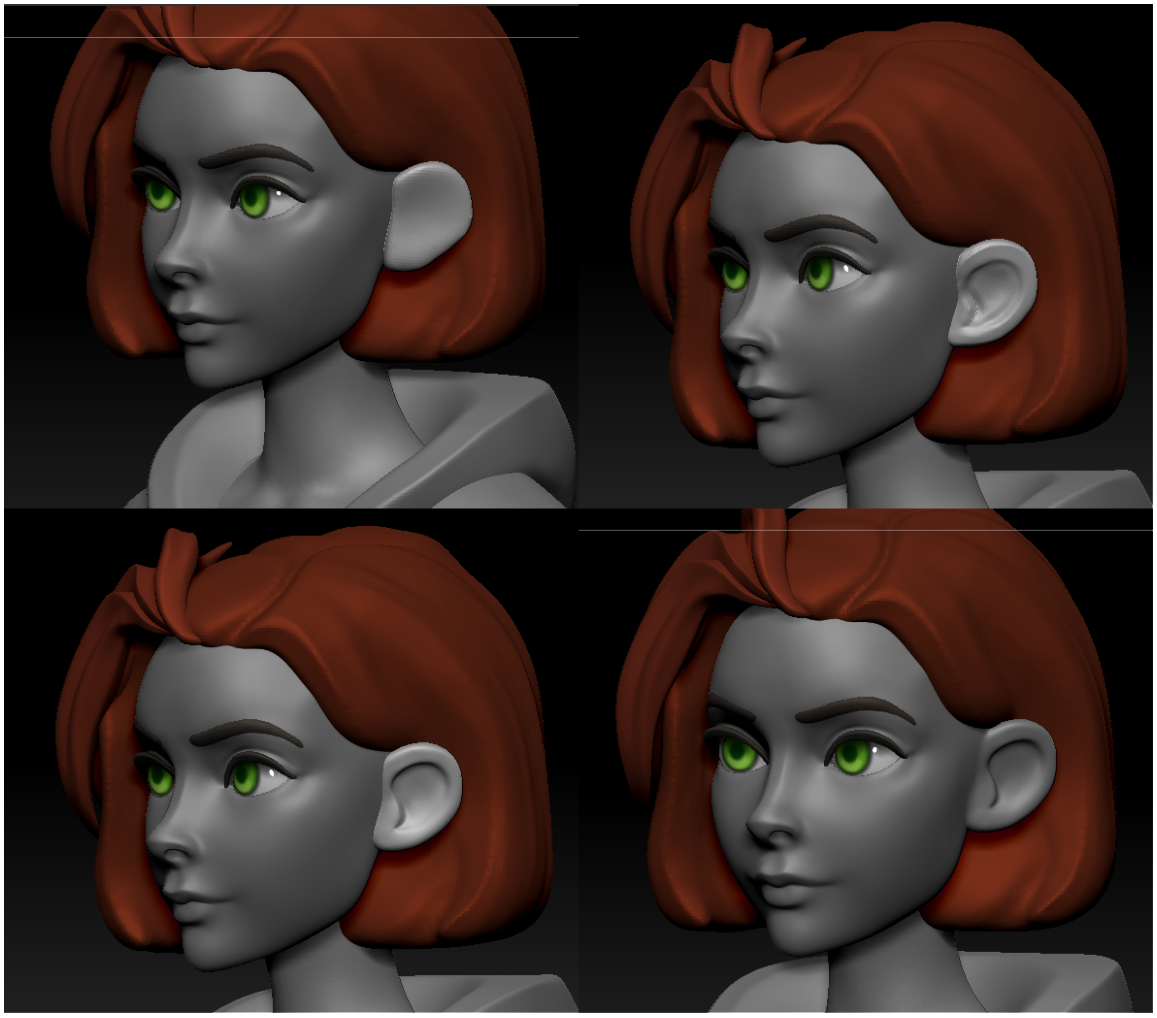
Hahmon farkkujen veistäminen oli samankaltainen prosessi, joskin vielä helpompi, sillä niiden muoto oli ihonmyötäinen ja ne veistettiin täysin symmetrisesti. Farkuille uniikki piirre olivat niiden saumat. Ne olisi voitu veistää vetämällä Standard-siveltimellä ura ja nipistämällä se yhteen Pinch-siveltimellä, mutta saumoista haluttiin tehdä näkyvämmät. Valittiin CurveStrapSnap-sivellin, jolla piirrettiin isot saumat. Ne irrotettiin farkkujen meshistä ja niistä tehtiin pehmeämmän näköiset jakamalla pinnan geometria (kuva 85). Lopuksi saumojen asentoa korjailtiin Move-siveltimellä.



Kuva 85. Farkkujen saumat piirrettin CurveStrapSnap-siveltimellä.

4.3.11 Korva

Hahmon korva oli nopea veistää, sillä tyyliin sopii yksinkertaisuus. Korvan sisuksen suurpiirteinen muoto kaiverrettiin Standard-siveltimellä ja kun siihen oltiin tyytyväisiä, resoluutiota nostettiin ja yksityiskohtia hiottiin. Sitten korva yhdistettiin pään alatyökaluun ja työkalu päivitettiin DynaMesh-tilassa niin, että meshit hitsaantuivat topologialtaan yhteen. Kun korva oli osa pään meshiä, poskipään ja korvan väli siloiteltiin Clay BuildUp- ja Smooth-siveltimillä. Lopuksi kasvojen topologia uudelleenrakennettiin ZRemesher-työkalulla ja yksityiskohdat projisoitiin pään varakopiosta niin kuin aiemmissa vaiheissa. Työvaiheet esitetään kuvassa 86.



Kuva 86. Korvan veistäminen ja sen yhdistäminen pään meshiin

4.3.12 Hiukset

Hahmon hiusten veistäminen oli haastavaa. Niistä tehtiin useampi iteraatio ja lopputulokseen ei silti oltu täysin tyytyväisiä. Veistäminen aloitettiin lisäämällä uusi Sphere 3D -alatyökalu, jota muovattiin enemmän hiusten muotoiseksi DynaMesh-tilassa. Sitten hiusmöhkälettä muovattiin muun muassa Clay BuildUp-, Smooth-, Move- ja Trim Dynamic -siveltimillä matalassa resoluutiossa. Kun hiusten muoto oli tyydyttävä, siirryttiin korkeampaan resoluutioon, jossa hiusten "virtaus" veistettiin tai ennemminkin piirrettin Orb_Cracks-siveltimellä. Lopuksi hiussuortuviin piirrettiin hienovaraisempia yksityiskohtia pienemmällä siveltimellä. Pienen siveltimen kokoa vaihdeltiin, jottei lopputulos olisi homogeeninen.

Otsahiukset luotiin lisäämällä Sphere 3D -alatyökalu, jota venytettiin transformointityökalulla pituussuunnassa. Meshiä nipistettiin molemmista päistä kapeammaksi transformointivalikon taper-työkalulla. Pötkylän läpileikkauksesta hiottiin DynaMesh-tilassa kulmikkaampi ja mesh taivutettiin S-muotoon otsalle transformointivalikon kurvityökalulla. Otsahiuksen alatyökalu duplikoitiin kahdesti ja sitä muovattiin hieman niin, etteivät kaikki kolme otsahiussuortuvaa olleet identtisiä. Lopuksi otsahiuksiin piirrettiin Orb_Cracks-siveltimellä pienempiä yksityiskohtia. Kuvassa 87 esitetään hahmon lopulliset hiukset.

Kuva 87. Hahmon lopulliset hiukset.

4.4 Mallin optimointi mobiilille

Valmiin veistetyn mallin verteksimäärä oli miltei kymmenen miljoonaa, joka on aivan liian korkea nykyhetken videopelikäyttöön. Hahmon tavoiteltava kolmiomäärä mobiilipelissä saattaisi olla esimerkiksi 5 000 - 10 000 kolmiota. Malli ei siis ollut vielä valmis, vaan se tuli optimoida.

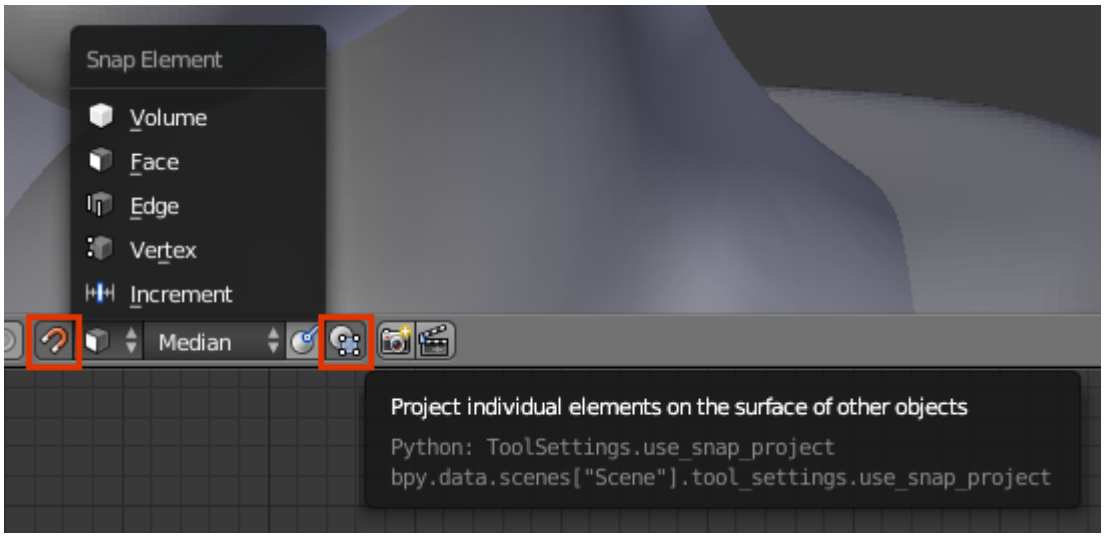
4.4.1 Retopologia

Mallin topologia uudelleenrakennettiin Blenderissä. Se oli melko jouheva prosessi ja antoi hyvän vertailupohjan tulevaisuuden projekteille, joissa saatetaan tutkia muita retopologiaohjelmia kuten 3D-Coat, Topogun tai ZBrush.

4.4.2 Kasvojen retopologisointi

Retopologiaprosessi aloitettiin hahmon kasvoista, sillä siihen löytyi hyvä tutoriaalivideosarja "How to Retopologize a Head like a Boss". Videosarjassa Danny Mac uudelleenrakensi topologian 3D-Coatissa, mutta ohjeita pystyi soveltamaan Blenderiin vaivatta. (Mac, D. 2017.)

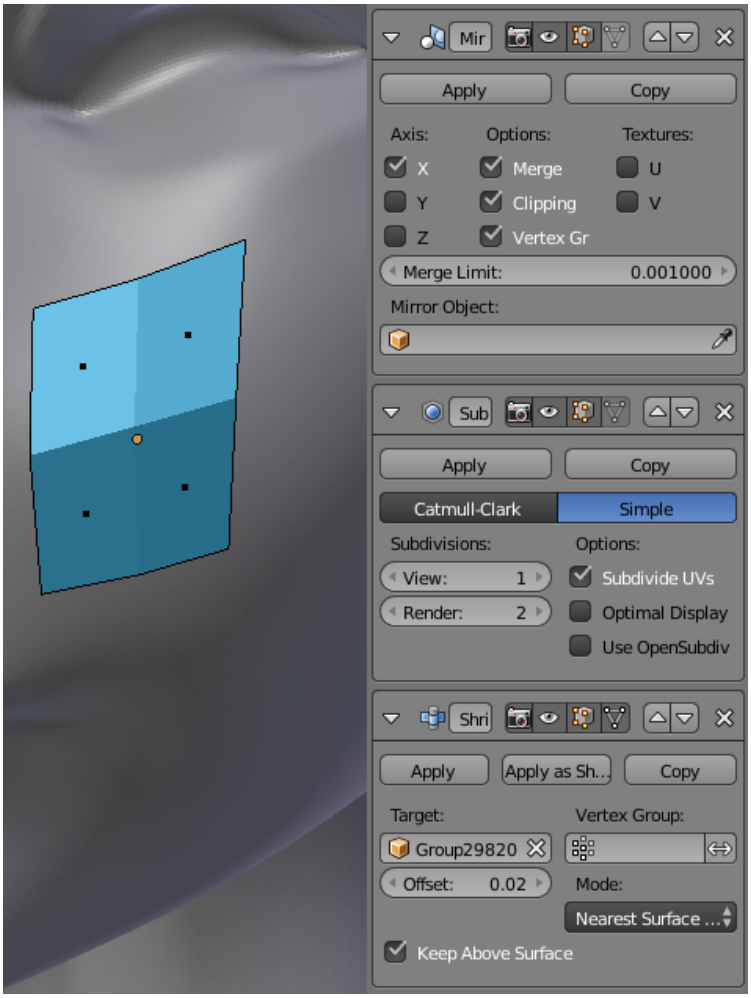
OBJ-tiedosto hahmon päästä tuotiin Blenderiin, jossa käyttöliittymä valmisteltiin retopologiaprosessiin Zacharias Reinhardtin tutoriaalivideon "How to set up Retopology in Blender (Tutorial EN)" mukaan (kuva 88). (Reinhardt, Z. 2017.)



Kuva 88. Napsahtamisen ja itsenäisten elementtien muiden objektien pinnalle projisoiminen salliminen Blenderin käyttöliittymässä.

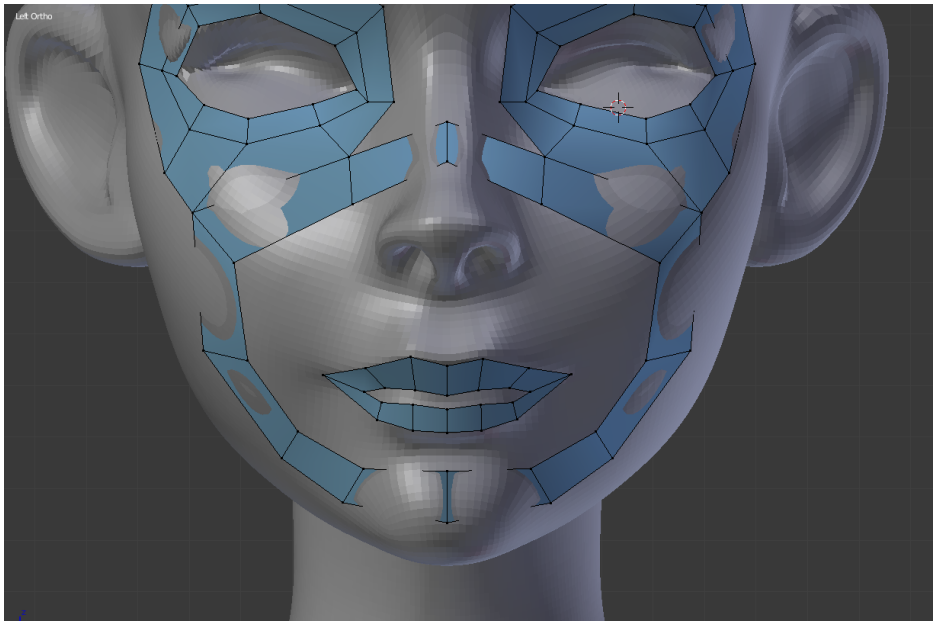
Varsinainen topologian uudelleenrakentaminen aloitettiin lisäämällä Blenderissä Planetasomesh, johon lisättiin muutama modifikaattori. Ensin lisättiin klippaava peilaus sekä pinnan jakaminen simppelillä algoritmilla. Tärkein retopologiamodifikaattori oli shrinkwrap- kiristekalvo, joka "kääri" uuden meshin pinnan raskasmallin ympärille

Kiristekalvomodifikaattorissa määritettiin, että uusi malli leijailee hieman vanhan mallin pinnan yläpuolella offsetpainoarvolla 0,02. Lopuksi kaikki modifikaattorit säädettiin näkyviksi muokkaustilassa ja uudelle mallille annettiin erivärinen materiaali, jotta se olisi helpompi erottaa alla olevasta raskasmallista. Modifikaattorien vaikutus litteän plane-tasomeshin pintaan esitetään kuvassa 89.



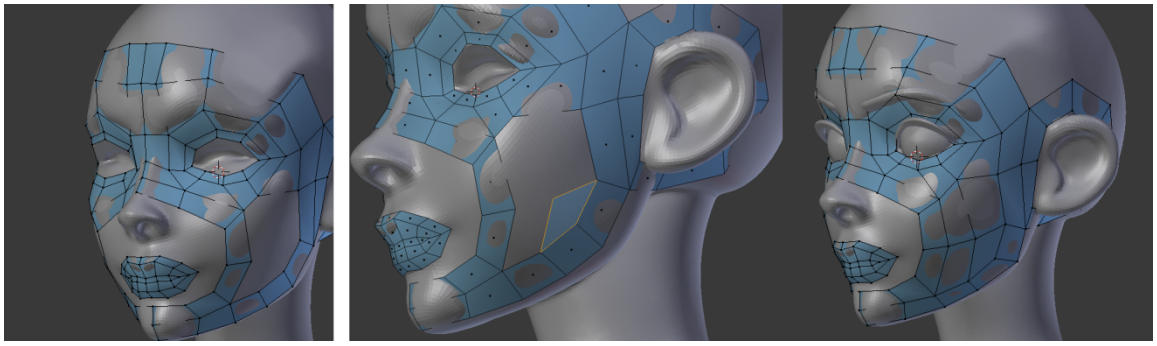
Kuva 89. Retopologiamodifikaattorit sekä raskasmallin pintaan takertuva plane-mesh.

Seuraavaksi mallia alettiin retopologisoimaan tutoriaalivideon mukaan. Ensin kasvoihin rakennettiin topologian tärkeimmät reunaluupit muun muassa silmien ja suun ympärille vetämällä uusia sivuja alkuperäisestä plane-meshistä ekstruusiomallintamalla (kuva 90).

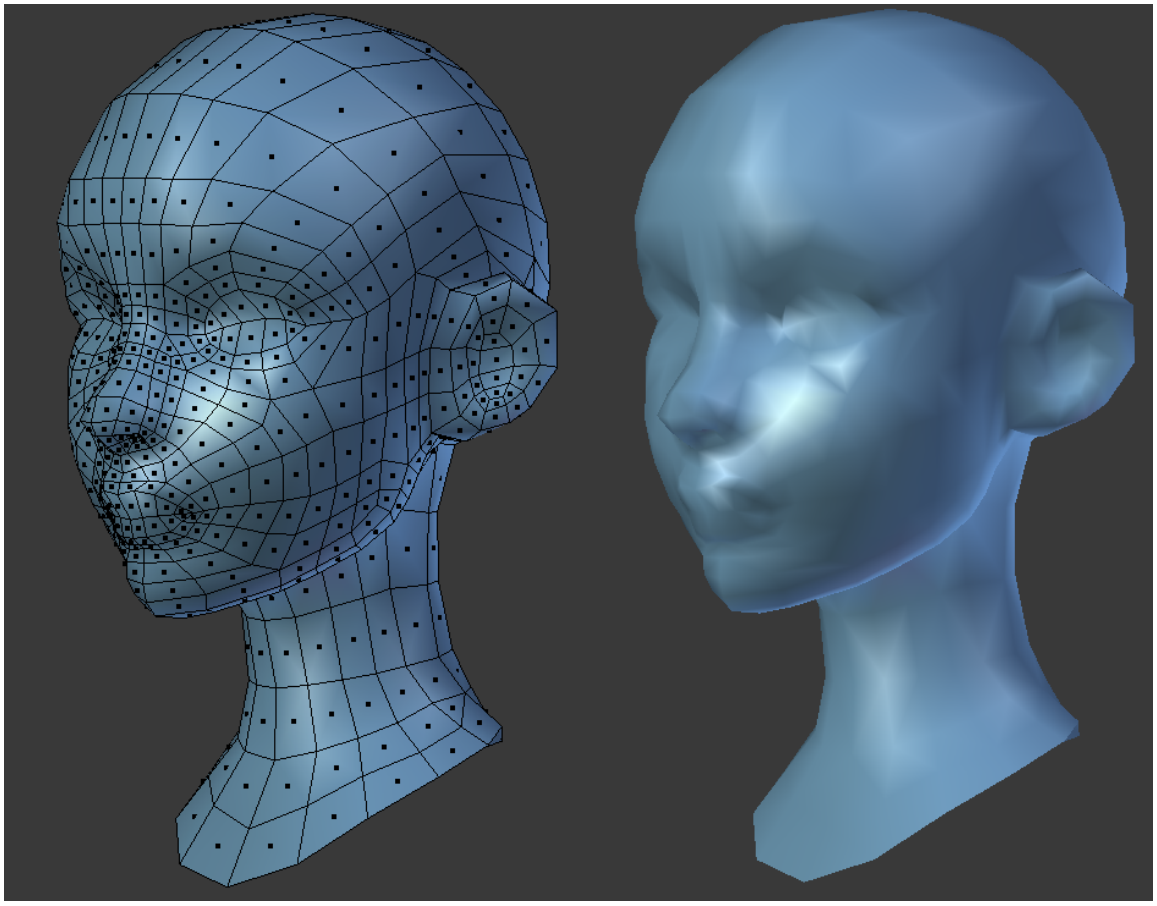


Kuva 90. Kasvojen tärkeimmät reunaluupit.

Kun kasvojen topologian tukiluupit oli luotu, niiden väliset raot täytettiin (kuva 91). Prosessi oli helppo ja suoraviivainen tutoriaalivideota seuraamalla. Koko pään topologia uudelleenrakennettiin vastaavasti ja pään lopullista topologiaa voi tarkastella kuvasta 92.



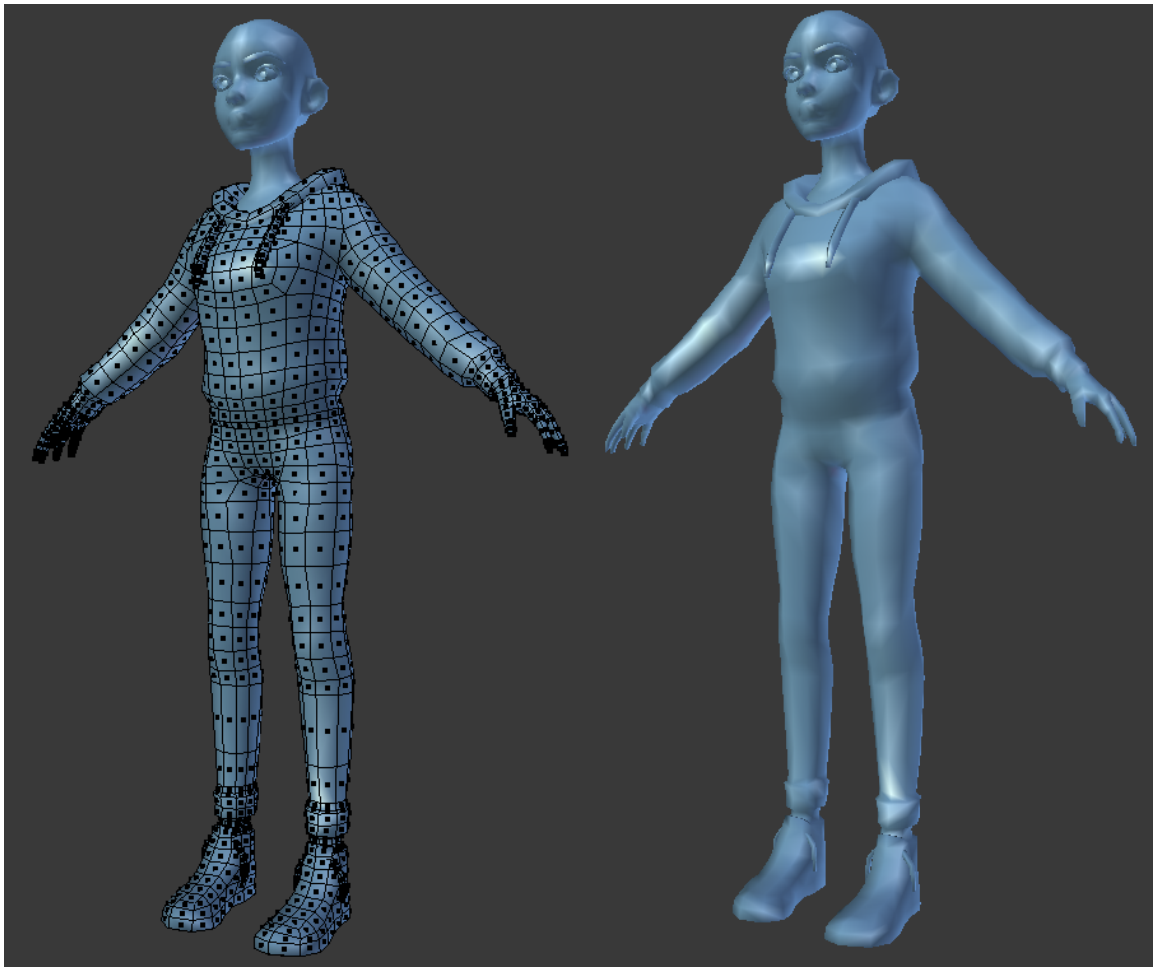
Kuva 91. Luuppien välinen tyhjä tila täytettiin polygoneilla.



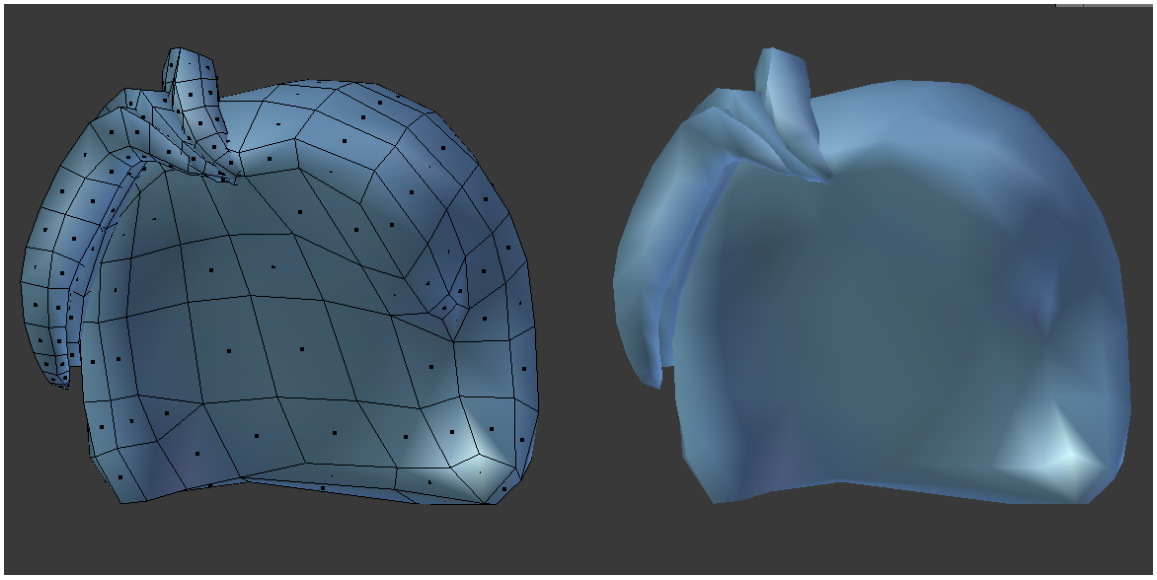
Kuva 92. Pään lopullinen topologia, 1 328 kolmiota.

4.4.3 Kehon ja hiusten retopologisointi

Kehon ja hiusten topologia uudelleenrakennettiin pään retopologiaprosessissa opituilla tekniikoilla. Suurin haaste oli epäsymmetrinen torso, joka haluttiin yhdistää symmetriseen alaresoriin ja hihoihin. Ongelma ratkaistiin iteroimalla topologiaa, kunnes yhdistymiskohdat toimivat aiheuttamatta suuria venymiä UV-saarissa. Hyviin topologiakäytänteisiin katsottiin esimerkkiä Polycountin BodyTopology-kirjastosta, vaikka esimerkkikuvien soveltaminen vaatetettuun hahmoon ei onnistunutkaan ihan täydellisesti. Kuvissa 93 ja 94 esitetään kehon ja hiusten lopullinen topologia: koko hahmon lopulliseksi kolmiomääräksi tuli 6 190 kolmiota. (Polycount Wiki, 2015.)



Kuva 93. Kehon lopullinen topologia, 3 446 kolmiota.

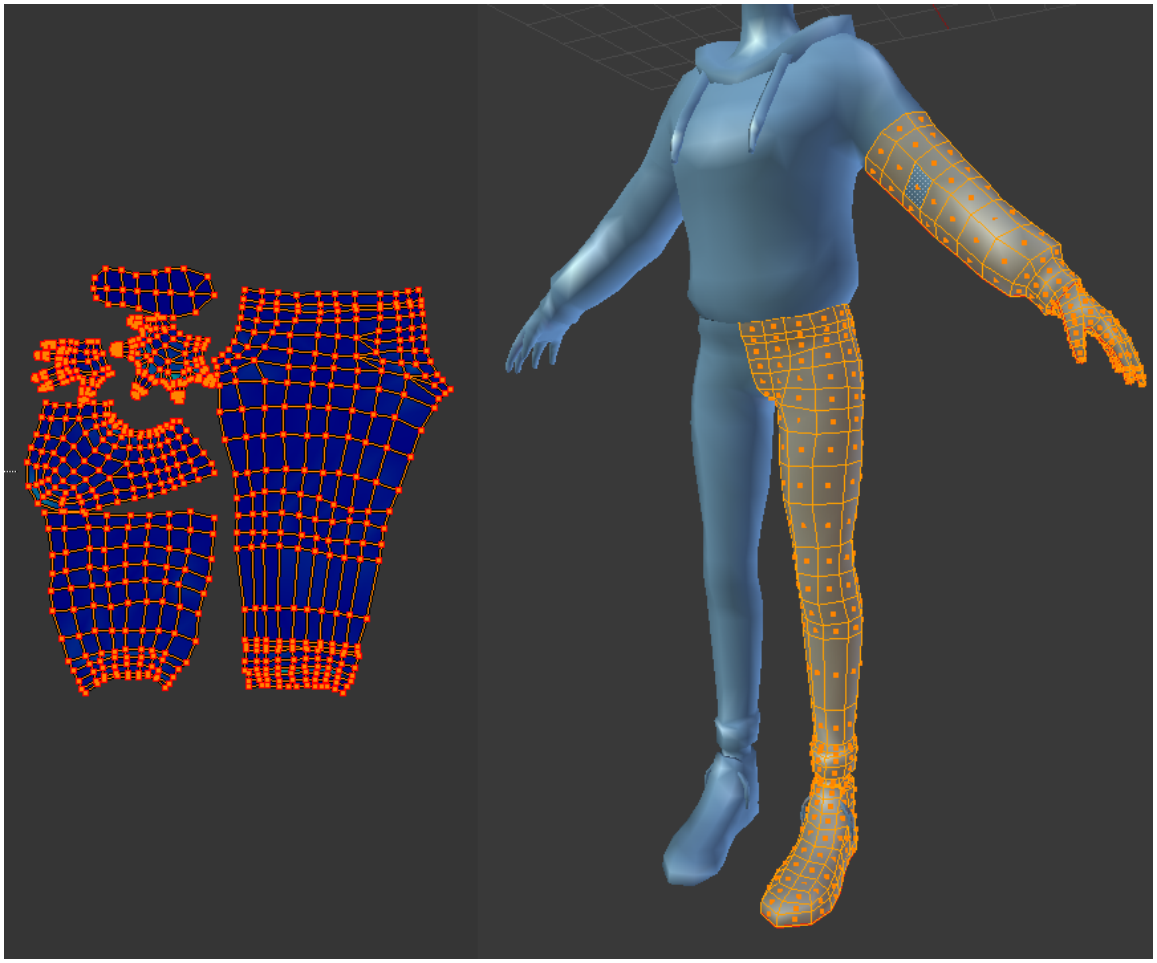


Kuva 94. Hiusten lopullinen topologia, 708 kolmiota.

4.4.4 Unwrappaus

Unwrappausprosessin alussa koko hahmon tekstuurikartan kooksi päätettiin 1024x1024 pikseliä ja hiusten tekstuurikartan kooksi 512x512 pikseliä. Hahmon hiukset säilytettiin erillisenä objektina, sillä ne haluttiin pitää modulaarisena.

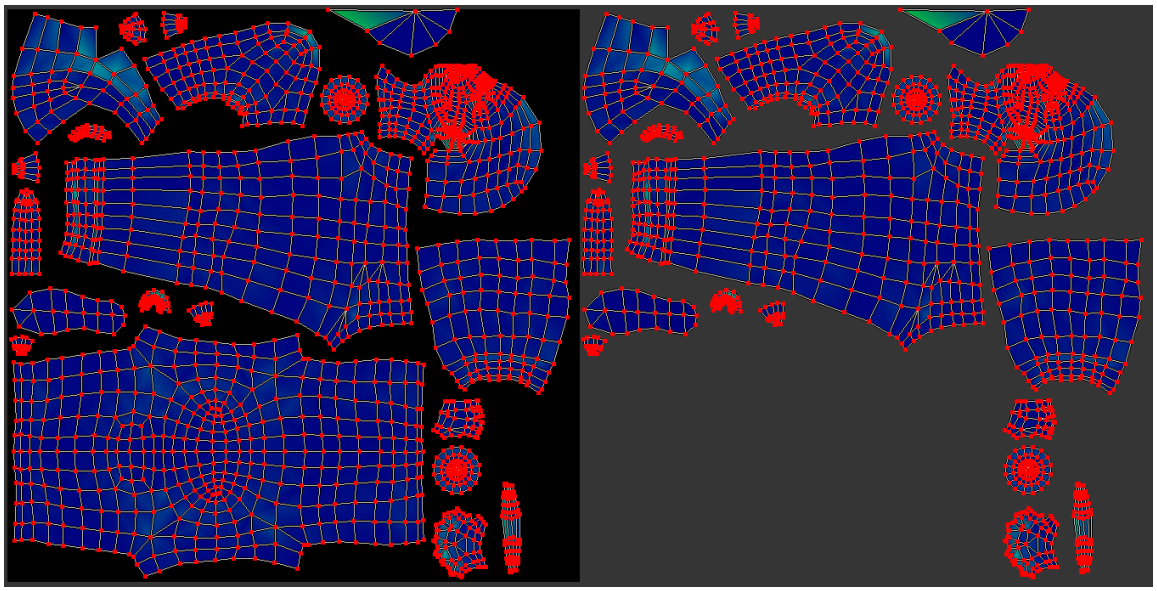
Seuraavaksi määritettiin, mitkä osat hahmosta voisivat hyödyntää symmetrisiä tekstuureita eli kaikki muut paitsi hupparin torso, sillä se oli veistetty ZBrushissa epäsymmetriseksi. Selkeyden vuoksi symmetriset ja epäsymmetriset osat eroteltiin omiksi alaobjekteikseen. UV-saumat merkittiin mahdollisimman piilotetusti, kuten vaatekappaleiden leikkauskohtiin tai jalan sisäpuolelle. Kun saumat oli merkitty, symmetriset osat unwrapattiin (kuva 95) ja tarkastettiin, ettei suurempia venymiä esiintynyt. Lopuksi epäsymmetrinen torso unwrapattiin ja kaikki mallin osat yhdistettiin.



Kuva 95. Hahmon symmetrisistä osista unwrapattiin vain puolet, sillä toinen puoli voitiin peilata myöhemmin.

UV-saarten pakkaukseen käytettiin Shotpacker-lisäosaa Blenderissä. Shotpacker ei tue kirjoittamishetkellä päällekkäisiä UV-saaria, joten hahmon peilimodifikaattoria ei applikoitu ennen UV-pakkausta. Lisäosan asetuksista sallittiin saarien rotatointi ja kääntäminen. Sopivaksi marginaaliarvoksi eli saarten väliseksi minimireunapehmikkeeksi asetettiin 0,01, joka vastaa tekstuurikartassa yhtä prosenttia (eli esimerkiksi kymmentä pikseliä tekstuurissa, joka on kooltaan 1024x1024 pikseliä). Koska saarten pakkauksessa ei ollut kiire, valittiin iso arvo (200) pakkauksen iteroimiseen ja annettiin ohjelman tehdä työnsä. Kun se oli valmis, peilimodifikaattori applikoitiin ja peilauksesta syntyneet tuplaverteksit poistettiin. (Nimimerkki ambi, 2017.)

Koska hahmolle haluttiin tulevaisuudessa beikata normaalikartta teksturointiprosessia varten, peilattujen osien UV-saaria siirrettiin tekstuurin koon verran sivuun, jotta beikkausohjelma tietäisi tarkalleen mihin osaan geometriaa beikkaus kohdistetaan. Valittiin kaikki peilatut osat ja niiden UV-saaria siirrettiin tekstuurin koon verran eli 1024 pikseliä oikealle UV-tilassa (kuva 96). (Polycount Wiki, 2018.)

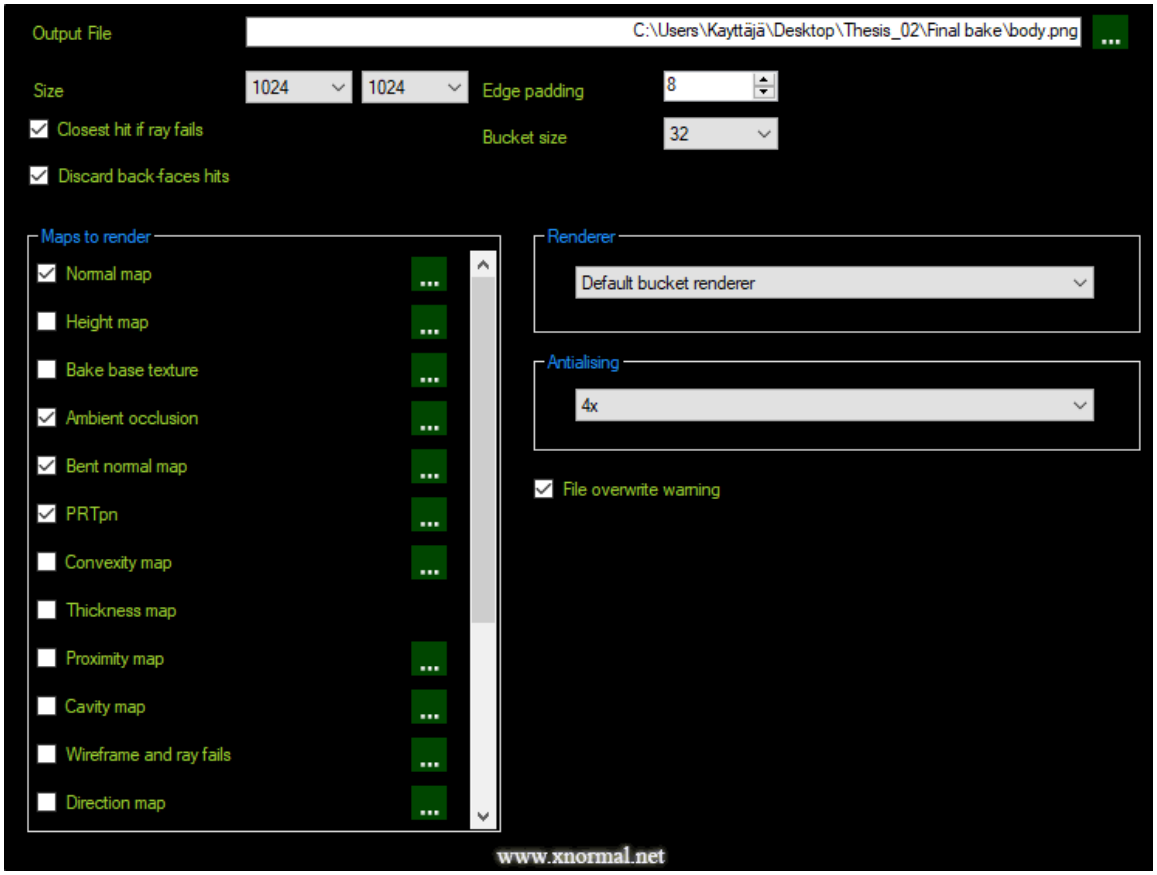


Kuva 96. Hahmon UV-saaret. Peilattuja saaria on siirretty 1024 pikseliä oikealle normaalikarttojen beikkauksen vuoksi.

4.4.5 Beikkaus

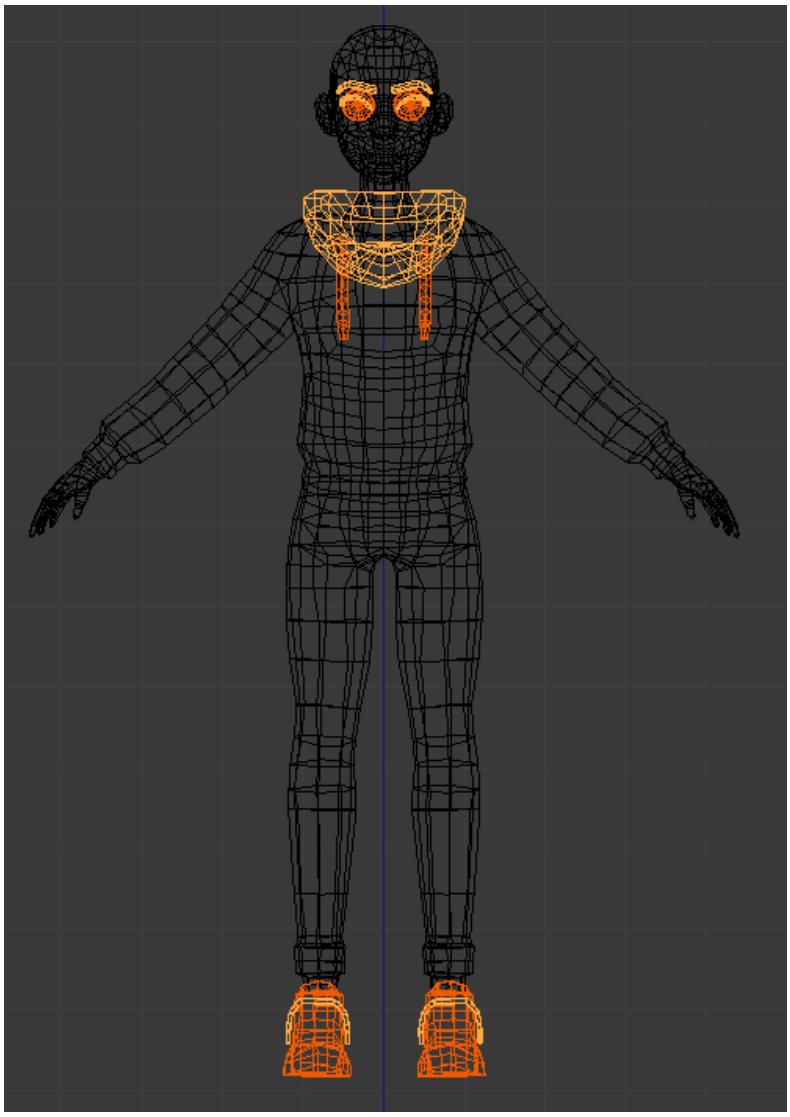
Raskasmallia hyödynnettiin teksturointiprosessissa beikkaamalla siitä erilaisia tekstuurikarttoja xNormal-beikkaustyökalulla. Beikkauksessa voi käyttää myös esimerkiksi modernimpia Marmoset Toolbag 3- tai Substance Painter -ohjelmistoja.

xNormalissa High definition meshes -välilehteen valittiin ZBrushista tuotu raskasmalli ja Low definition meshes -välilehteen Blenderissä retopologisoitu kevytmalli. Teksturointiprosessiin otettiin mallia Polycount-foorumiketjusta "Is it possible to bake normal maps into diffuse maps?", jossa nimimerkki Fingus käytti hahmonsa teksturoinnissa normaalikartasta johdettua kaviteettikarttaa, taipunutta normaalikarttaa sekä PRTpn-normaalikarttaa. Valittiin renderöitäväksi kaikki nämä kartat sekä niiden lisäksi ambientti okkluusio. Osa kartoista renderöitiin ensin matalaresoluutioisena ja niitä testattiin Marmoset Toolbag 3 -renderöintiohjelmassa, jotta korkearesoluutioisten karttojen beikkaamiseen ei käytettäisi paljon aikaa vain sen huomaamiseksi, että mallissa tai sen unwrappauksessa on jokin virhe. Testibeikkauksen jälkeen osan renderöitävistä kartoista asetuksissa muutettiin säteiden määrää korkeammaksi ja sallittiin jitter-tärinä, joka tuo varjoihin kohinaa, jolloin ne näyttävät pehmeämmiltä. Antialiasing-arvoksi asetettiin 4x ja tekstuurikooksi 1024x1024 pikseliä, jolloin sopivaksi reunapehmikkeen kooksi valittiin kahdeksan pikseliä. Hiusmallin tekstuurikartta oli puolet pienempi, joten sille myös riitti puolet pienempi reunapehmike. Kuvassa 97 näkyvät kaikki käytetyt beikkausasetukset. (Nimimerkit Fingus, Joopson, 2013; Polycount Wiki, 2017.)



Kuva 97. xNormalissa käytetyt beikkausasetukset.

Testibeikkauksessa tuli vastaan ongelma, kun meshin päällekkäiset osat leikkasivat toisiinsa kartoissa ja aiheuttivat häiriöitä; esimerkiksi hahmon hupparin narut heijastuivat hupparin etuosaan 2D-kuvana. Tämän vuoksi beikattavat mallit piti niin kutsutusti "räjäyttää" (explode-työvaihe) eli jakaa elementteihin, jotka eivät ole liian lähellä toisiaan ja aiheuta beikkaushäiriöitä. Malli jaettiin kolmeen osaan (kuva 98), joka oli tarpeeksi ongelman välttämiseksi. (Polycount WIki, 2018.)



Kuva 98. Osat, joihin malli "räjäytettiin" beikkauksen ajaksi - ensimmäiseen osaan (kuvassa musta rautalankamalli) kuului suurin osa hahmosta, toiseen osaan (kuvassa keltainen rautalankamalli) kuuluivat hahmon kulmakarvat, ripset, huppu sekä kengännauhat ja kolmanteen osaan (kuvassa oranssi rautalankamalli) kuuluivat hahmon silmät, hupparin narut ja kengät.

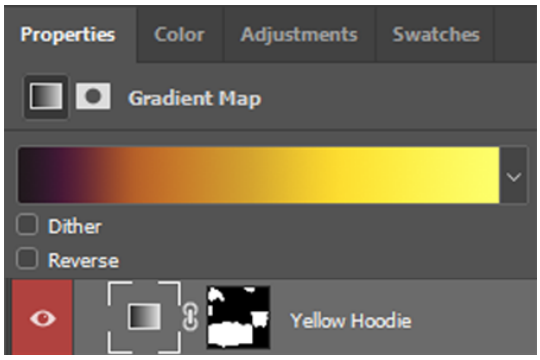
4.4.6 Teksturointi

Hahmo teksturoitiin Adobe Photoshop CC 2018 -ohjelmassa. Ensin kuvatiedoston taustaväriksi valittiin neutraali harmaa (RGB-arvo 128/128/128), jonka päälle kartat ladottiin. Alimmaiseksi tasoksi kopioitiin ja liitettiin PRT-p-normaalikartan vihreä kanava. Sen sekoitustilaksi valittiin pehmeä valo ja läpinäkyvyysasteeksi 100 %. Koettiin, että efekti oli heikko, joten taso monistettiin ja uuden kopion läpinäkyvyysasteeksi valittiin testaamalla 50 %. Kuvassa 99 esitetään tekstuuri tämän vaiheen jälkeen.

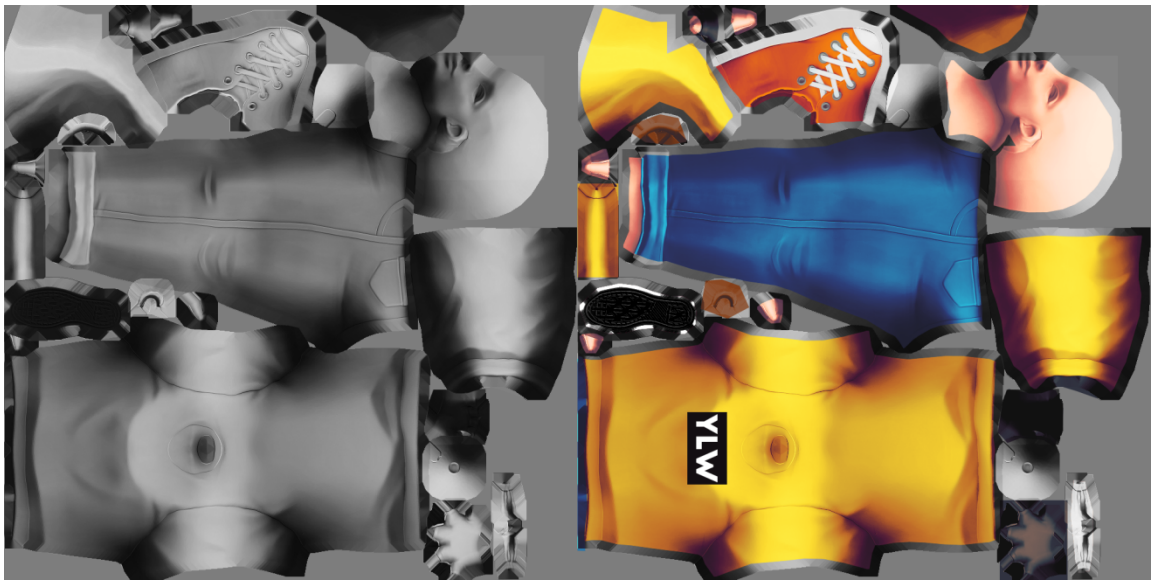
Kuva 99. Hahmomalli Blenderissä valaisemattomassa materiaalinäkymässä, kun diffuusitekstuuri koostuu neutraalista harmaasta ja PRT-p-normaalikartan vihreästä kanavasta.

Seuraavalle tasolle liitettiin ambientti okkluusio -kartta kertovassa sekoitustilassa läpinäkyvyysasteella 50 %. Kolmannelle tasolle liitettiin taipuneen normaalikartan vihreä kanava pehmeä valo -sekoitustilassa läpinäkyvyysasteella 50 %. Kaviteettikartta kartta generoitiin xNormalissa tangenttiavaruusnormaalikartasta Tools-välilehdestä löytyvällä Tanget-space normal map to cavity map -toiminnolla asetuksilla Bright 0,18, Contrast 0,8 ja Method EMB. Kaviteettikartta lisättiin Photoshopiin päällimmäiseksi tasoksi sulauttavassa sekoitustilassa läpinäkyvyysasteella 100 %. Lopputulos oli mustavalkoinen tekstuurikartta, joka käytti erilaisia normaalikarttoja ja ambienttia okkluusiota luodakseen vakuuttavan pohjan diffuusikartalle (kuva 101).

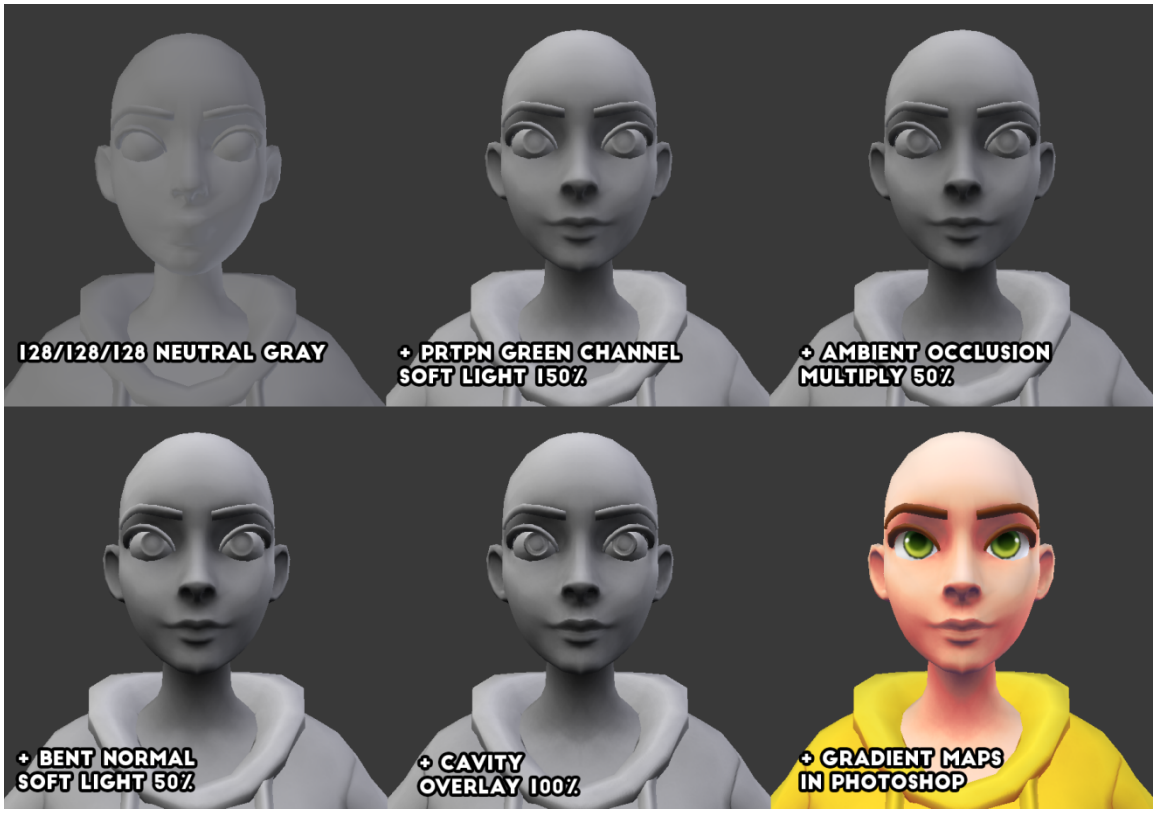
Seuraavaksi Photoshopissa maskattiin UV-kartasta tekstuurin eriväriset osat ja hahmo väritettiin gradienttikartoilla (kuva 100) samaan tapaan kuin Arthur Gimaldinovin videossa "Gradient map". Hahmossa haluttiin käyttää kirkkaita värejä, jotka sopisivat mobiilipeliin - esimerkiksi hupparista tehtiin keltainen, sillä se on huomiotaherättävä väri ja saattaisi saada mobiilipelaajan kiinnostumaan pelistä. Hiusten teksturointi suoritettiin samalla tavalla, joskin se oli vielä helpompaa, sillä tekstuurikartan peitti vain yksi maskaamaton gradienttikartta. Teksturointiprosessista otettiin kuva jokaisen kartan ja värien lisäämisen jälkeen: vaiheita voi tarkastella kuvasta 102. (Gimaldinov, A. 2016.)



Kuva 100. Huppari väritettiin keltaisella gradienttikartalla.

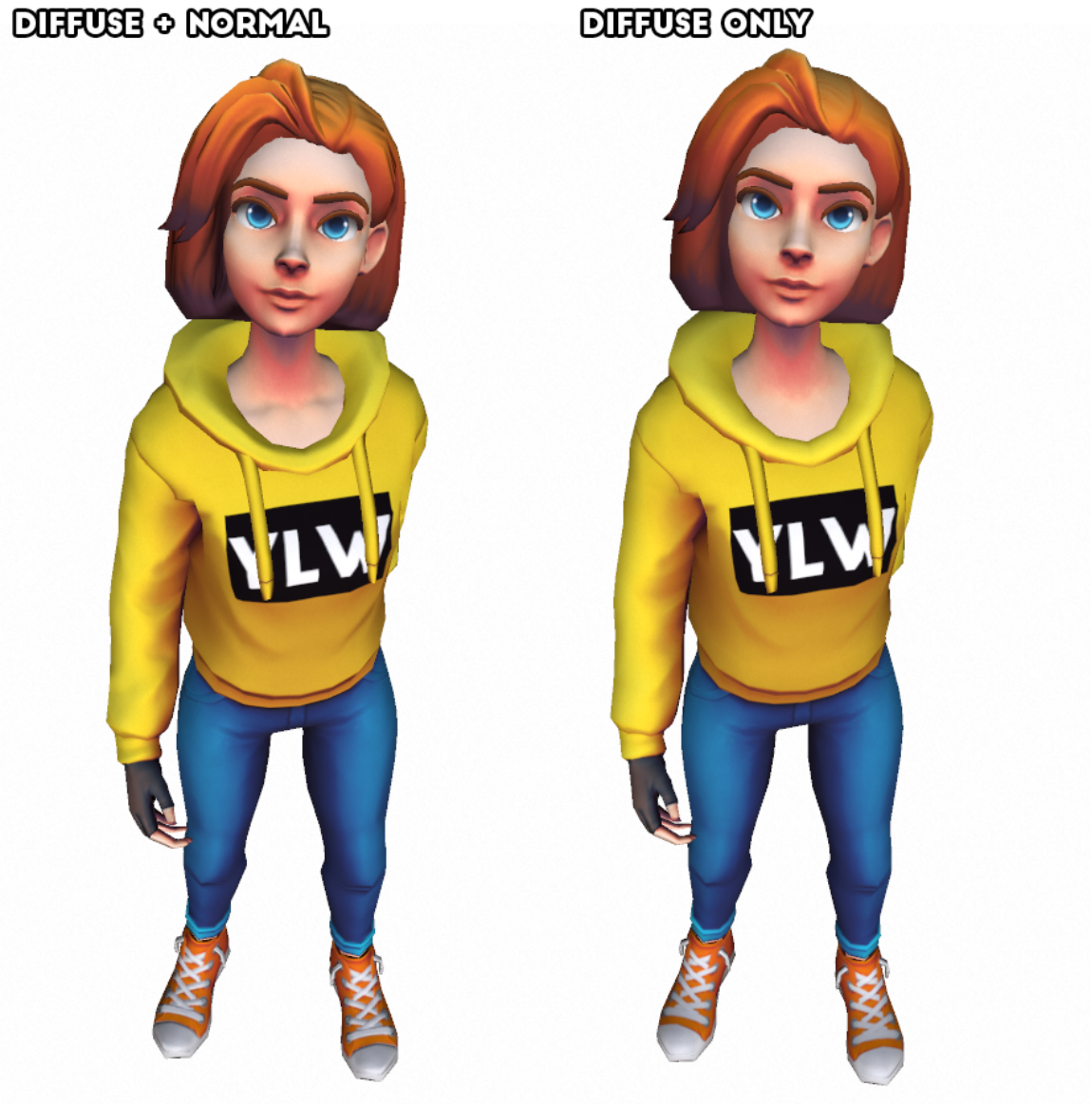


Kuva 101. Beikatuista kartoista johdettu mustavalkoinen tekstuuri sekä gradienttikartoilla väritetty tekstuuri.

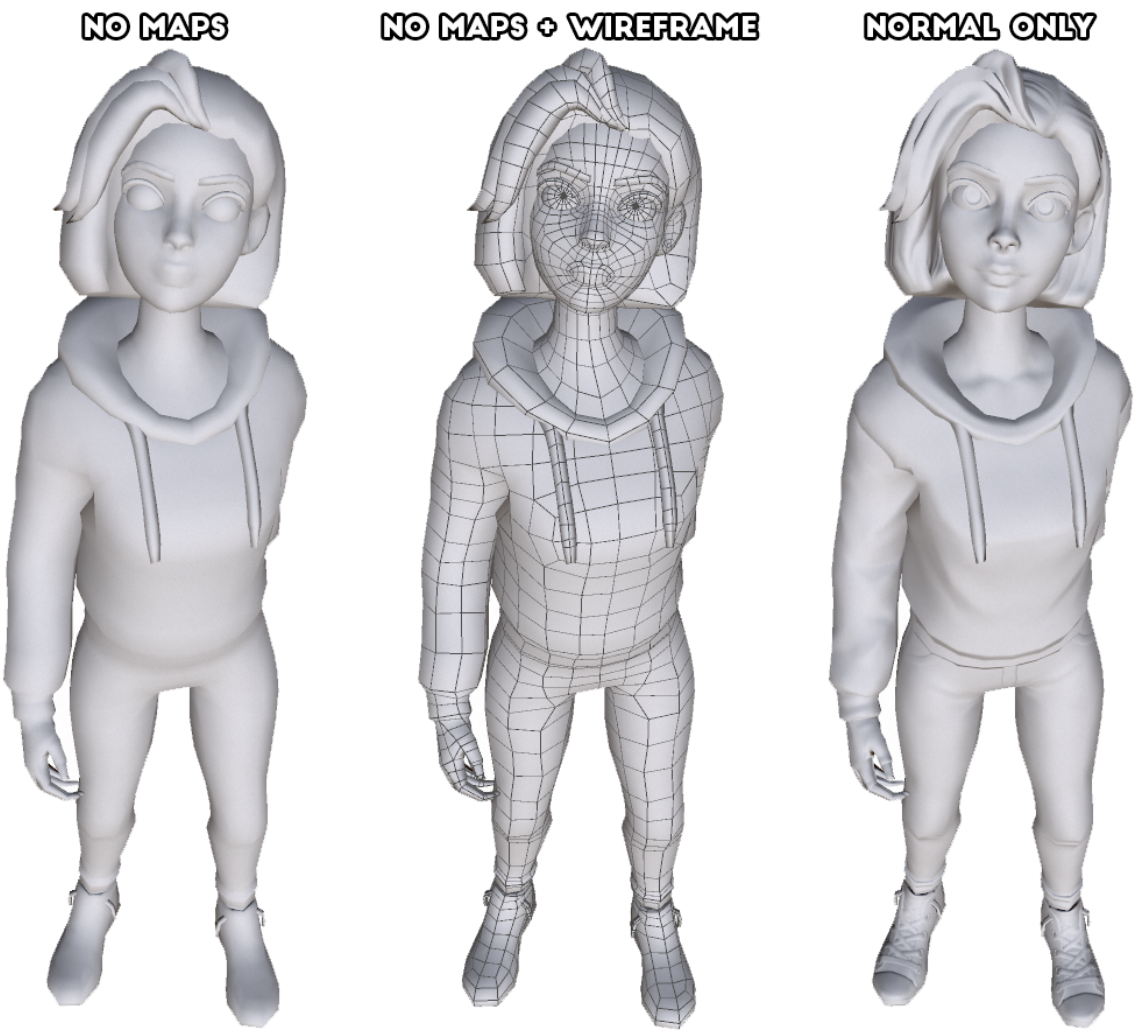


Kuva 102. Teksturointiprosessi askelittain Blenderin valaisemattomassa materiaalinäkymässä.

Normaalikarttoja hyödyntämällä voi tallentaa paljon pinnan yksityiskohtia hahmon diffuusikarttaan. Kuvassa 103 voi verrata toisiinsa hahmoa, joka käyttää sekä normaalikarttaa että diffuusikarttaa ja hahmoa, joka käyttää vain diffuusikarttaa - ero ei ole valtava, vaikka diffuusikarttaa ei ole edes hiottu huippuunsa. Kuvassa 104 voi tarkastella lopullista mallia myös ilman mitään tekstuurikarttoja, mallia niin, että rautalankaverkko on näkyvillä sekä mallia niin, että se käyttää normaalikarttaa muttei diffuusikarttaa.



Kuva 103. Hahmo sekä normaali- että diffuusikartalla (vas.) ja pelkällä diffuusikartalla.



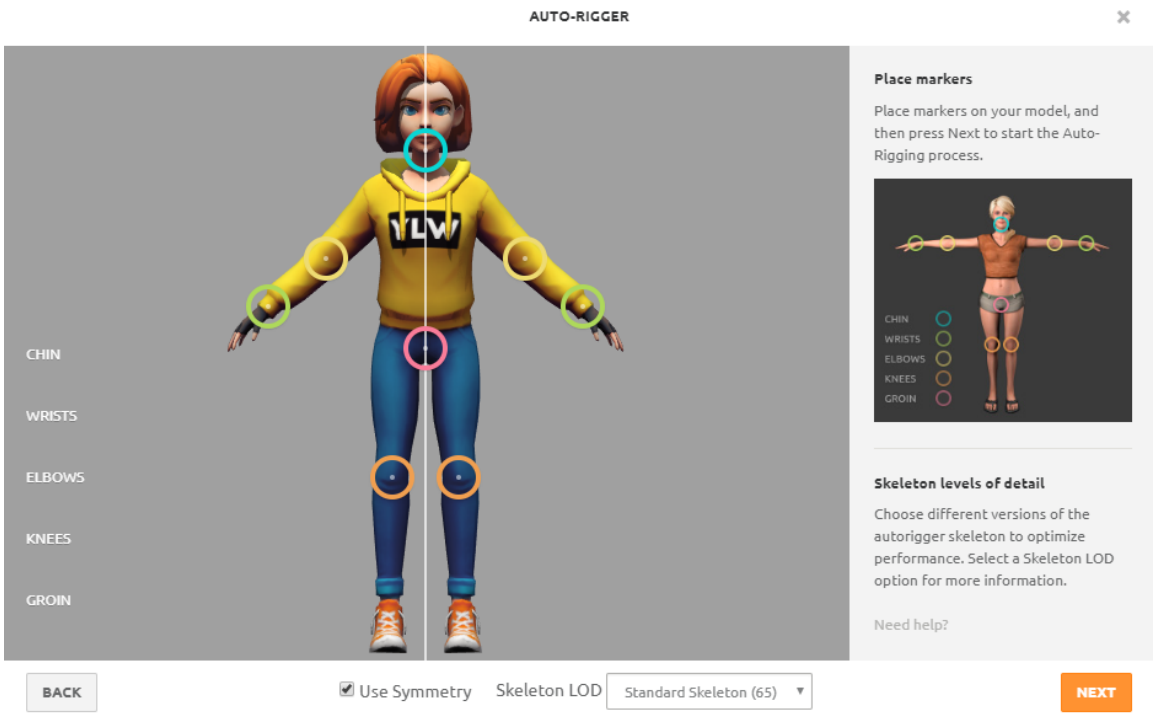
Kuva 104. Lopullinen hahmomalli renderöitynä ilman mitään tekstuurikarttoja (vas.), ilman tekstuurikarttoja rautalankamalli näkyvillä (kesk.) ja pelkän normaalikartan kanssa.

Veistetty malli oli teksturointiprosessissa hyödyllinen ja lopputuloksesta tuli parempi kuin mitä olisi osattu tehdä käsin teksturoiden. Sen lisäksi raskasmallista saatiin juonnettua myös normaalikartta. Veistämisprosessi oli hitaampi kuin hahmon perinteinen polygonimallintaminen, mutta se johtui osittain siitä, että ei ollut aikaisempaa kokemusta veistämisestä. Kuvassa 105 näkyy runsas vuosi sitten laatikkomallintamani hahmo sekä opinnäytetyöhön nyt veistetty hahmo. Hahmoja on hieman ontuvaa verrata, sillä kehitystä artistina on tapahtunut muutenkin, mutta koen, että veistäessä oli helpompi kanavoida luovuutta ja toteuttaa päänsisäinen visio.

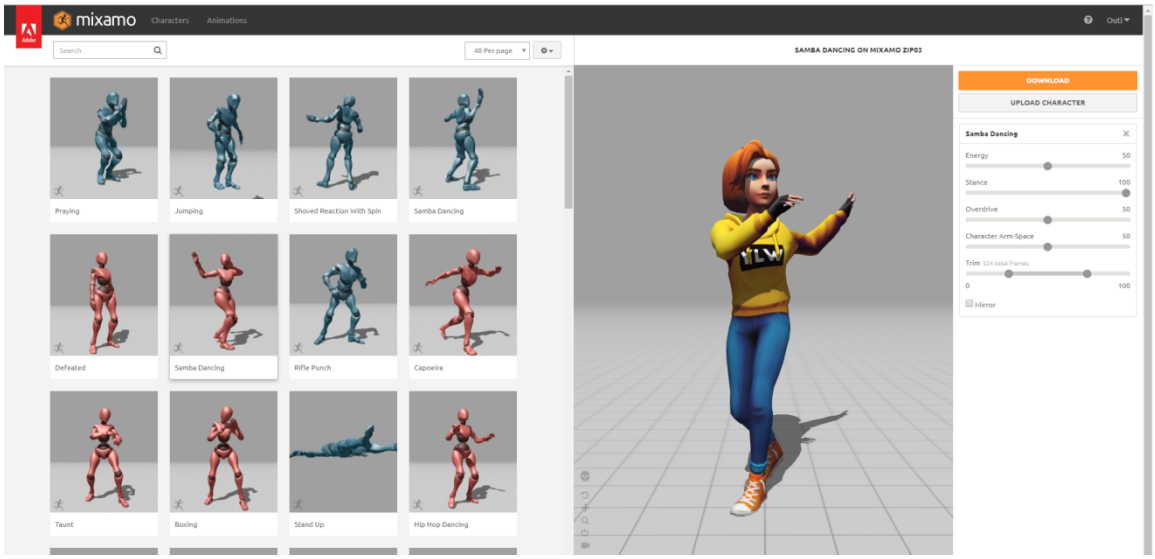
Kuva 105. Noin vuosi sitten laatikkomallinnettu ja käsinmaalattu hahmo (vas.) ja nyt digitaalisesti veistetty ja optimoitu hahmo.

4.4.7 Animointi ja renderöinti

Hahmoa ei varsinaisesti animoitu itse, vaan se tuotiin FBX-tiedostona Adoben Mixamo-palveluun, jossa 3D-hahmon voi nopeasti rigata ja animoida. Käyttäjän tarvitsee vain kertoa ohjelmalle, missä hahmon leuka, ranteet, kyynärpäät, polvet ja nivuset sijaitsevat, minkä perusteella ohjelmisto osaa rigata hahmon (kuva 106). Mixamossa on suuri valmisanimaatiokirjasto (kuva 107), josta animaatiot voi tallentaa FBX-tiedostoina, jotka voidaan ladata esimerkiksi renderöintiohjelmaan tai jopa pelimoottoriin.



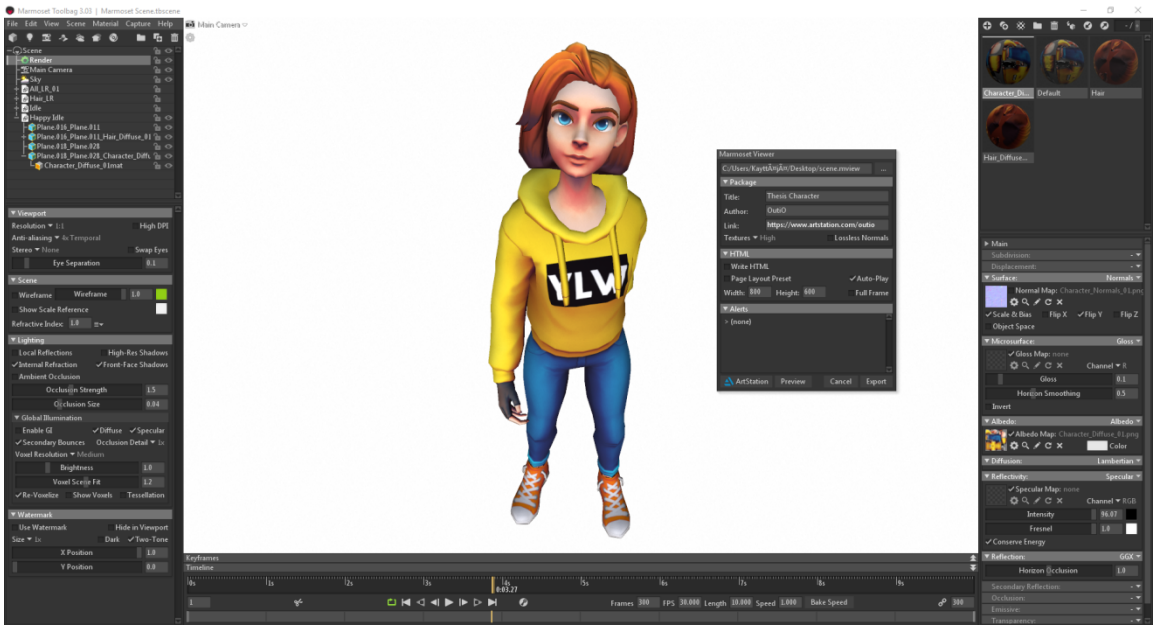
Kuva 106. Hahmon autoriggaus Mixamossa.



Kuva 107. Mixamon valmisanimaatiokirjasto, jota autorigattu hahmo voi hyödyntää.

Hahmo renderöitiin Marmoset Toolbag 3:ssa. Prosessi oli yksinkertainen: Mixamosta ladattu FBX-tiedosto raahattiin Marmoset Toolbag 3:n käyttöliittymään ja animaatio kelattiin haluttuun kohtaan. Hahmon materiaaliin raahattiin hahmon tekstuurikartat ja sitä säädettiin hieman vähemmän kiiltäväksi. 3D-tilan valaistus muutettiin neutraalimmaksi ja taustaväriksi valittiin valkoinen. Kamera siirrettiin kohdalleen. Lopuksi tiedostovalikosta vietiin ulos Marmoset Viewer -tiedosto, jolla voi ladata 3D-mallista 3D-upotteen verkko-

portfolioon niin, että katsoja voi tarkastella mallia joka kuvakulmasta. Kuvassa 108 näkyy hahmomalli Marmoset Toolbag 3 -ohjelmiston sisällä.



Kuva 108. Hahmo Marmoset Toolbag 3:ssa.

4.4.8 Testaus

Ajanpuutteen vuoksi hahmoa ei valitettavasti voitu testata käytännössä mobiilipelissä, mutta mallin verteksimäärän ja materiaalien puolesta sen pitäisi toimia hyvin. Hahmon ulkoasun viehättävyyttä mobiilipelaajan näkökulmasta ei voida käytettävillä resursseilla konkreettisesti arvioida, mutta siinä pyrittiin parhaaseen mahdolliseen lopputulokseen käyttämällä kirkkaita värejä ja helposti lähestyttävää taidetyylisuuntaa.

5 YHTEENVETO JA POHDINTA

Opinnäytetyön päätarkoitus oli testata digitaalisen veistämisen ottamista osaksi 3D-mobiilipelihahmonkehityksen työnkulkua ja kartuttaa tietotaitoa digitaalisesta veistämisestä. Teoriaosuudessa tarkasteltiin mobiilipelien menneisyyttyä ja tulevaisuutta sekä erilaisia metodeja kehittää videopelihahmo. Käytännön työn päätulokset olivat, että veistäminen on luovempaa ja intuitiivisempaa kuin perinteinen polygonimallintaminen ja lopputuloksesta tuli näyttävä, mutta prosessi oli hieman hitaampi. Oppimistavoitteissa onnistuttiin osittain: työssä opittiin esimerkiksi, millaisia ongelmia eri työvaiheissa voi tulla vastaan ja miten niitä voi välttää, mutta kaikkiin osa-alueisiin ei ehditty syventyä niin paljon kuin olisi haluttu.

Työn tulosten merkitys on subjektiivinen, sillä kaikki artistit eivät työskentele samalla tavalla – jollekulle toiselle perinteinen mallintaminen voi tuntua luontevammalta kuin veistäminen. Työn tuloksia voidaan hyödyntää ottamalla digitaalinen veistäminen osaksi videopeligraafikkojen opintopolkua, sillä se voi olla intuitiivisempi tapa oppia mallintamaan esimerkiksi opiskelijalle, jolla on jo taustaa perinteisistä taiteista. Opinnäytetyön aihetta olisi voitu rajata enemmän, mikä olisi mahdollistanut syventymisen johonkin tiettyyn aihealueeseen, kuten autoretopologiavaihtoehtojen tutkimiseen, modernien mallinnus- ja teksturointiohjelmistojen vertailuun, viehättävän mobiilipelihahmon suunnitteluun tai PBR-varjostuksen mobiililla tutkimiseen.

Suurin osa opinnäytetyön projektityössä ilmenneistä ongelmista onnistuttiin ratkaisemaan pyytämällä apua tai palautetta lähipiirin artisteilta. Työssä olisi kannattanut hyödyntää myös verkkosivuja kuten Polycount-foorumi, jossa maailmanluokan artistit ovat valmiina auttamaan.

Joitakin opinnäytetyön askelia oltaisiin voinut nopeuttaa tai ulkoistaa. Esimerkiksi 2D-konseptikuva olisi ollut hyvä olla olemassa, joihinkin työvaiheisiin (kuten kengänpohjien veistämiseen) käytettiin suhteettomasti aikaa ja hahmo olisi voitu veistää nopeammin käyttämällä apuna jonkinlaista perusmeshiä. Toisaalta hahmon perinpohjainen veistäminen alusta asti itse oli tietoinen valinta, joka auttoi ymmärtämään itse ohjelmistoa ja veistämistä paremmin.

Opinnäytetyö lisäsi tietoa aihepiiristä. Suomenkielistä dokumentaatiota digitaalisesta veistämisestä on hankala löytää, joten ehkä kirjoitus raivaa tietä tuleville töille. Toivotta-

vasti työ inspiroi opiskelevia artisteja koettamaan taitojaan digitaalisessa veistämisessä ja saa lukijan ymmärtämään mobiilin potentiaalin tulevaisuuden videopelialustana.

Lähteet

Nimimerkki Agilethief. (2016, toukokuu). Technical Study: Overwatch [Image heavy]. Haettu 3.2.2018, sivustolta Polycount Forum internetosoite: http://polycount.com/discussion/170394/technical-study-overwatch-image-heavy

Nimimerkki aggsol. (2014, 15. joulukuuta). Normal mapping for Android and iOS. Haettu 27.1.2018, sivustolta StackExchange internetosoite: https://gamedev.stackexchange.com/questions/59630/normal-mapping-for-android-and-ios

Nimimerkit aliyeredon2, LightingBox2. (2016, joulukuu). Advanced image effect for mobile. Haettu 27.1.2018, sivustolta Unity Forums internetosoite: https://forum.unity.com/threads/advanced-image-effect-for-mobile.445790/

Nimimerkki ambi. (2017, huhtikuu). [Addon] UV packing. Haettu 18.4.2018, sivustolta Blender Artists Community internetosoite: https://blenderartists.org/t/addon-uv-packing/687964

Amos, E. (2014, 30. kesäkuuta). Nokia-NGage-LL.jpg. Haettu 13.5.2018, Wikipedia - Vapaa tietosanakirja internetosoite: https://en.wikipedia.org/wiki/N-Gage_(device)#/media/File:Nokia-NGage-LL.jpg

Andaur, C. (2004, 1. syyskuuta). Unwrapping Suzanne. Haettu 30.1.2018, sivustolta Blender Documentation Volume I - User Guide internetosoite: https://docs.blender.org/api/htmlI/x5336.html

Autodesk. (2018). Buy online. Haettu 21.4.2018, sivustolta Autodesk internetosoite: https://www.autodesk.eu/buy-online

Autodesk. (2016, 11. toukokuuta). Polygonal Modeling. Haettu 9.4.2018, sivustolta Autodesk internetosoite: https://knowledge.autodesk.com/support/maya/learn-explore/caas/CloudHelp/cloudhelp/2016/ENU/Maya/files/GUID-7941F97A-36E8-47FE-95D1-71412A3B3017-htm.html

Bautista, B. C. (2014, 4. marraskuuta). 66 percent of players ditch free Android and iOS games within a day, study says. Haettu 27.3.2018, sivustolta Digital Trends internetosoite: https://www.digitaltrends.com/mobile/users-ditch-freemium-games/#!OJ5ao

Blender Reference Manual. (2018). Seams. Haettu 30.1.2018, sivustolta Blender Reference Manual internetosoite: https://docs.blender.org/manual/en/dev/editors/uv_image/uv/editing/unwrapping/seams.html

Blender Reference Manual. (2018). Specular Shaders. Haettu 16.5.2018, sivustolta Blender Reference Manual internetosoite: https://docs.blender.org/manual/en/dev/render/blender_render/materials/properties/specular_shaders.html

Nimimerkit bluescrn, Rajmahal. (2014, kesäkuu). Are normal maps typically used on mobile games? Haettu 27.1.2018, sivustolta Unity Forums internetosoite: https://forum.unity.com/threads/are-normal-maps-typically-used-on-mobile-games.253574/

Nimimerkit Brian Ortiz, Katana314, munificent, M2tM, yusef ghatavi. (2010-2016). What to consider when deciding on 2D vs 3D for a game? Haettu 27.1.2018, sivustolta StackExchange internetosoite: https://gamedev.stackexchange.com/questions/631/what-to-consider-when-deciding-on-2d-vs-3d-for-a-game

Nimimerkit bumble864, Tseng, razormax. (2012, maaliskuu). How many polygons for a mobile scene? Haettu 3.2.2018, sivustolta Unity Forums internetosoite: https://forum.unity.com/threads/how-many-polygons-for-a-mobile-scene.129292/

Chadwick, E. (2015, 12. toukokuuta). Normalmap stairs. Haettu 3.2.2018, sivustolta Polycount Wiki internetosoite: http://wiki.polycount.com/wiki/File:Normalmap_stairs.jpg

Chadwick, E. (2015, 21. toukokuuta). Triangles. Haettu 19.4.2018, sivustolta Polycount Wiki internetosoite: http://wiki.polycount.com/wiki/File:Triangles.jpg

Nimimerkit Nimimerkit CheeseOnToast, Quasar. (2011, tammikuu). How to determine what size texture map to use. Haettu 2.6.2018, sivustolta Polycount Forum internetosoite: http://polycount.com/discussion/80329/how-to-determine-what-size-texture-map-to-use

Nimimerkki CrazyButcher. (2007, marraskuu). [Technical Talk] - FAQ: Game art optimisation (do polygon counts really matter?). Haettu 19.4.2018, sivustolta Polycount Forums internetosoite: http://polycount.com/discussion/50588/technical-talk-faq-game-art-optimisation-do-polygon-counts-really-matter

Creative Bloq Staff. (2014, 21. elokuuta). The top 7 sculpting apps for 3D artists. Haettu 3.4.2018, sivustolta Creative Bloq internetosoite: https://www.creativebloq.com/audiovisual/sculpting-apps-81412712

Cowley, R. (2017, 14. heinäkuuta). Mobile games revenues surpassed PC and console revenues in 2016 for the first time. Haettu 28.3.2018, sivustolta Pocket Gamer internetosoite: http://www.pocketgamer.biz/news/66201/mobile-games-revenues-surpassed-pc-and-console/

Cross, K. (2017, 19. joulukuuta). How the legal battle around loot boxes will change video games forever. Haettu 28.3.2018, sivustolta The Verge internetosoite: http://www.pocketgamer.biz/news/66201/mobile-games-revenues-surpassed-pc-and-console/

Cupisz, R. (2011, 8. kesäkuuta). Advanced Shading and Lighting for Mobile. Haettu 24.4.2018, sivustolta Unity Blog internetosoite: https://blogs.unity3d.com/2011/06/08/advanced-shading-and-lighting-for-mobile/

Nimimerkit DerRazputin, FildoSaggins, OccultMonk. (2017, huhtikuu). Best auto-retopology tool? Haettu 20.4.2018, sivustolta Polycount Forum internetosoite: http://polycount.com/discussion/186194/best-auto-retopology-tool

Difference Between. (2017). Difference between 2D and 3D. Haettu 3.2.2018, sivustolta Difference Between internetosoite: http://www.differencebetween.info/difference-between-2d-and-3d

Diver, M. (2015, 27. toukokuuta). If Mobile Is the Future of Gaming, Then the Future of Gaming Is Really Sad. Haettu 28.3.2018, sivustolta Vice internetosoite: https://www.vice.com/sv/article/exqng4/if-mobile-is-future-of-video-gaming-then-the-future-of-video-gaming-is-really-sad-522

Diver, M. (2015, 18. toukokuuta). Konami, Like Atari, Is Risking its Legacy in Striving for Longevity. Haettu 28.3.2018, sivustolta Vice internetosoite: https://www.vice.com/en_uk/article/dp58xj/konami-like-atari-is-risking-its-legacy-to-strive-for-longevity-639

Nimimerkki DMGregory. (2017, 2. helmikuuta). UV World mapping in shader with Unity. Haettu 30.1.2018, sivustolta StackExchange internetosoite: https://gamedev.stackexchange.com/questions/136652/uv-world-mapping-in-shader-with-unity

Don College Design Wiki (2016, 10, elokuuta). Box Modelling. Haettu 21.4.2018, sivustolta DC Design Don College Design Wiki internetosoite: http://dcdesign.wikidot.com/wiki:box-modelling

Doose, S. (2016, 28. marraskuuta). Illidan Stormrage - Commission work. Haettu 3.4.2018, sivustolta ZBrushCentral internetosoite: http://www.zbrushcentral.com/showthread.php?199638-Sheridan-Doose-Art-Dump&p=1194660&viewfull=1#post1194660

DOS Games Archive. (2018). Duke Nukem 3D. Haettu 10.4.2018, sivustolta DOS Games Archive internetosoite: https://www.dosgamesarchive.com/download/duke-nukem-3d/

Du, K. C. C. (2013, 18. lokakuuta). Photo texture to albedo map test. Haettu 3.2.2018, sivustolta Art Of Kai - The Last Of Us Environment Final year project internetosoite: http://artofkaidufyp.blogspot.fi/2013/10/photo-texture-to-albedo-map-test.html

Nimimerkki EarthQuake. (2012, lokakuu). You're making me hard. Making sense of hard edges, uvs, normal maps and vertex counts. Haettu 30.1.2018, sivustolta Polycount Forum internetosoite: http://polycount.com/discussion/107196/youre-making-me-hard-making-sense-of-hard-edges-uvs-normal-maps-and-vertex-counts

Nimimerkit Everspace, poistettu nimimerkki. (2015). Is there a performance impact for using different texture maps such as bump maps, specular and such as opposed to just the plain texture? Haettu 19.5.2018, sivustolta Reddit - Unity 3D internetosoite: https://www.reddit.com/r/Unity3D/comments/29qtbt/is_there_a_performance_impact_for_using_different/

Nimimerkit Fingus, Joopson. (2013, marraskuu). Is it possible to bake normal maps into diffuse maps? Haettu 17.4.2018, sivustolta Polycount Forum internetosoite: http://polycount.com/discussion/128428/is-it-possible-to-bake-normal-maps-into-diffuse-maps

Freeman, J. (2017, 17. toukokuuta). What are draw calls? Haettu 3.2.2018, sivustolta Lynda.com internetosoite: https://www.lynda.com/Unity-tutorials/What-draw-calls/599611/616812-4.html

Nimimerkit Galamoth, IcyEyes, Jagged85, Vigorousjammer. (2017, 23. tammikuuta). 2.3D. Haettu 10.4.2018, sivustolta Giant Bomb internetosoite: https://www.giantbomb.com/25d/3015-660/

Giant Bomb. (2018). Top-Down Perspective Haettu 2.6.2018, sivustolta Giant Bomb internetosoite: https://www.giantbomb.com/top-down-perspective/3015-788/

Nimimerkki GiantCowGilms. (2015, 20. kesäkuuta). What techniques are used to begin creating new models? Haettu 21.4.2018, sivustolta Blender Stack Exchange internetosoite: https://blender.stackexchange.com/questions/26869/what-techniques-are-used-to-begin-creating-new-models

Gimaldinov, A. (2016, 18. kesäkuuta). Gradient map. Haettu osoitteesta https://www.youtube.com/watch?v=A3WB63SKZ90

Nimimerkki Gintas_. (2013, 4. heinäkuuta). What textures size do you recommend for Android? Haettu 3.2.2018, sivustolta Unity Forums internetosoite: https://forum.unity.com/threads/what-textures-size-do-you-recommend-for-android.189088/

Goldstone, W. (2012, 23. maaliskuuta). Shadowgun: Optimizing for Mobile Sample Level. Haettu 3.2.2018, sivustolta Unity Blog internetosoite: https://blogs.unity3d.com/2012/03/23/shadowgun-optimizing-for-mobile-sample-level/

Google Play. (2018). Tuottavimmat: Pelit. Haettu 27.1.2018, sivustolta Google Play internetosoite: https://play.google.com/store/apps/category/GAME/collection/topgrossing

Hearthstone Wiki. (2015-2017). Design and Development of Hearthstone. Haettu 28.1.2018, sivustolta Hearthstone Wiki internetosoite: https://hearthstone.gamepedia.com/Design_and_development_of_Hearthstone#Initial_development

Heginbotham, C. (2018, maaliskuu). What is 3D Digital Sculpting? Haettu 3.4.2018, sivustolta Concept Art Empire internetosoite: https://conceptartempire.com/what-is-3d-sculpting/

Hekkala, E. (2017). Videopelihahmon mallintaminen ja teksturointi nykymenetelmin. Haettu 3.6.2018, sivustolta Theseus internetosoite: http://www.theseus.fi/handle/10024/134720

Helder, P. (2010, tammikuu). # of sides on a barrel. Haettu 19.4.2018, sivustolta Polycount Forums internetosoite: http://polycount.com/discussion/69183/of-sides-on-a-barrel

Hennepe, S. (2010, toukokuu). Reinventing the wheel. Haettu 3.2.2018, sivustolta Polycount Forum internetosoite: http://polycount.com/discussion/72514/reinventing-the-wheel

Iezzi, L. (2016, 17. marraskuuta). Figuring Out Textel Density. Haettu 2.6.2018, sivustolta 80 Level internetosoite: https://80.lv/articles/textel-density-tutorial/

IMGA. (2017, 25. syyskuuta). The 5 Trends Shaping the Future of Mobile Gaming. Haettu 28.3.2018, sivustolta International Mobile Gaming Awards Global internetosoite: http://www.imgawards.com/news/industry/five-trends-shaping-future-mobile-gaming/

IMGA. (2017, 7. elokuuta). Why Mobile Gaming is the Future of Esports. Haettu 28.3.2018, sivustolta International Mobile Gaming Awards Global internetosoite: https://www.imgawards.com/news/industry/mobile-gaming-future-esports/

Nimimerkki jbooth. (2016, 11. helmikuuta). Object Space Normal Maps on Mobile. Haettu 11.4.2018, sivustolta Unity Forums internetosoite: https://forum.unity.com/threads/object-space-normal-maps-on-mobile.385212/

Jukić, T. (2014, 24. maaliskuuta). Draw calls in a nutshell. Haettu 3.2.2018, sivustolta Medium internetosoite: https://medium.com/@toncijukic/draw-calls-in-a-nutshell-597330a85381

Just Total Tech. (2014, 24. maaliskuuta). 2D Versus 3D Gaming. Haettu 28.1.2018, sivustolta Just Total Tech internetosoite: http://www.justtotaltech.co.uk/blog/2d-3d-gaming

Kemppainen, M. (2012). 3D-hahmojen toteutus mobiilipeliin. Haettu 3.6.2018, sivustolta Theseus internetosoite: http://www.theseus.fi/handle/10024/53744

King. (2018). Candy Crush Saga. Haettu 28.1.2018, sivustolta Google Play internetosoite: https://play.google.com/store/apps/details?id=com.king.candycrushsaga&hl=fi

Nimimerkki klownzie. (2015). Is there a performance impact for using different texture maps such as bump maps, specular and such as opposed to just the plain texture? Haettu 11.4.2018, sivustolta Reddit, Unity3D internetosoite: https://www.reddit.com/r/Unity3D/comments/29qtbt/is_there_a_performance_impact_for_using_different/

Lampel, J. (2017, 15. kesäkuuta). Normal vs. Displacement Mapping & Why Games Use Normals. Haettu 3.2.2018, sivustolta CG Cookie internetosoite: https://cgcookie.com/articles/normal-vs-displacement-mapping-why-games-use-normals

Lehtiniitty, M. (2017, 18. joulukuuta). Razer Phone nyt myynnissä Suomeen - 750 euron huippupuhelimessa on pari erikoisuutta. Haettu 27.3.2018, sivustolta mobiili.fi internetosoite: https://mobiili.fi/2017/12/18/razer-phone-nyt-myynnissa-suomeen-750-euron-huippupuhelimessa-on-pari-erikoisuutta/

Mac, D. (2017, 12. huhtikuuta). How to Retopologize a Head like a Boss. Haettu osoitteesta https://www.youtube.com/watch?v=iWEDWFMjA9E

Mac, D. (2017, 5. tammikuuta). How to sculpt a stylized head in ZBrush - Tutorial Part 1. Haettu osoitteesta https://www.youtube.com/watch?v=cjZt0ICfutQ

Martin, M. (2016, 5. maaliskuuta). 8 of the worst consoles ever made, from the Atari Jaguar to the Barcode Battler and Philips CD-i. Haettu 2.6.2018, sivustolta Digital Spy internetosoite: http://www.digitalspy.com/gaming/feature/a785474/8-of-the-worst-consoles-ever-made-from-the-atari-jaguar-to-the-barcode-battler-and-philips-cd-i/

Merriam-Webster. (2018, 10. huhtikuuta). Optimization. Haettu 20.4.2018, sivustolta Merriam-Webster internetosoite: https://www.merriam-webster.com/dictionary/optimization

Nimimerkki MarkG (2009, 17. toukokuuta). What is the purpose of Retopology, the BIG why? Haettu 3.4.2018, sivustolta 3D Coat Forums internetosoite: http://3dcoat.com/forum/index.php?/topic/2620-what-is-the-purpose-of-retopology-the-big-why/

Masters, M. (2015, 28. tammikuuta). 3ds Max, Maya LT or Blender - Which 3D Software Should I Choose for Asset Creation? Haettu 21.4.2018, sivustolta Pluralsight internetosoite: http://blog.digitaltutors.com/3ds-max-maya-lt-blender-3d-software-choose-asset-creation/

Masters, M. (2015, 29. tammikuuta). 3ds Max vs. Maya: Is One Better than the Other? Haettu 21.4.2018, sivustolta Pluralsight internetosoite: http://blog.digitaltutors.com/3ds-max-vs-maya-is-one-better-than-the-other/

Nimimerkki Narpas. (2016, 21. elokuuta) Lightmapped Scene with Lightmap. Haettu 24.4.2018, sivustolta Wikipedia - Vapaa tietosanakirja internetosoite: https://en.wikipedia.org/wiki/File:Lightmapped_Scene_with_Lightmap.png

Naser, A. (2015, 16. huhtikuuta). Why to Develop 2D Games for Mobile Instead of 3D Games? Haettu 27.1.2018, sivustolta LinkedIn internetosoite: https://www.linkedin.com/pulse/why-develop-2d-games-mobile-instead-3d-ahmad-hammad/

Niantic. (2018). Pokémon Go. Haettu 28.1.2018, sivustolta Google Play internetosoite: https://play.google.com/store/apps/details?id=com.nianticlabs.pokemongo&hl=fi

Oberson, P. (2005, 7. heinäkuuta). Texturing tricks tutorial. Haettu 3.2.2018, sivustolta Pier Oberson internetosoite: http://pioroberson.com/tuts/tut_texturing_tricks.htm

OpenGL Tutorials (2017). OpenGL Normal Mapping (DOT3 Bump Mapping). Haettu 3.2.2018, sivustolta OpenGL Game Development Tutorials internetosoite: http://wiki.polycount.com/wiki/FaceTopology

Oshchepkov, A. (2015, tammikuu). A Practical Guide On Normal Mapping for Games. Haettu 15.5.2018, sivustoilta Polycount Forum internetosoite: http://polycount.com/discussion/146667/a-practical-guide-on-normal-mapping-for-games, https://drive.google.com/file/d/0B02lElvs8BcvYllmQWpXUGxod3M/view

Papstein, K. (2015). An Introduction to ZBrush with Kurt Papstein Haettu osoitteesta http://pixologic.com/zclassroom/lesson/an-introduction-to-zbrush-with-kurt-papstein

Parker, T. (2011). Face Topology. Haettu 3.2.2018, sivustolta Polycount Wiki interneto-soite: http://wiki.polycount.com/wiki/FaceTopology

Pettit, N. (2015, syyskuu). Asset Workflow for Game Art: 3D Modeling. Haettu 9.4.2018, sivustolta Treehouse internetosoite: http://blog.teamtreehouse.com/asset-workflow-game-art-3d-modeling

Pettit, N. (2014, lokakuu). Real-Time 3D on the Web. Haettu 1.2.2018, sivustolta Tree-house internetosoite: http://blog.teamtreehouse.com/real-time-3d-web

Pixologic. (2018). Sculpting. Haettu 3.4.2018, sivustolta Pixologic Documentation inter-netosoite: http://docs.pixologic.com/user-guide/3d-modeling/sculpting/

Pixologic. (2018). The Pixol. Haettu 3.4.2018, sivustolta Pixologic Documentation internetosoite: http://docs.pixologic.com/getting-started/basic-concepts/the-pixol/

Pixologic. (2018). Your First Creature Creation. Haettu 3.4.2018, sivustolta Pixologic Documentation internetosoite: http://docs.pixologic.com/getting-started/your-first-creature/

Pixologic. (2018). ZRemesher. Haettu 20.4.2018, sivustolta Pixologic Documentation internetosoite: http://docs.pixologic.com/user-guide/3d-modeling/topology/zremesher/

Pluralsight. (2014, 14. elokuuta). Eliminate Texture Confusion: Bump, Normal and Displacement Maps. Haettu 12.5.2018, sivustolta Pluralsight internetosoite: https://www.pluralsight.com/blog/film-games/bump-normal-and-displacement-maps

Pluralsight. (2014, 16. tammikuuta). Start Mastering Important 3D Texturing Terminology. Haettu 4.4.2018, sivustolta Pluralsight internetosoite: https://www.pluralsight.com/blog/film-games/cover-bases-common-3d-texturing-terminology

Polycount Wiki. (2015). BodyTopology. Haettu 18.4.2018, sivustolta Polycount Wiki internetosoite: http://wiki.polycount.com/wiki/BodyTopology

Polycount Wiki. (2012). Diffuse map. Haettu 1.2.2018, sivustolta Polycount Wiki internetosoite: http://wiki.polycount.com/wiki/Diffuse_map

Polycount Wiki. (2017). Edge padding. Haettu 17.4.2018, sivustolta Polycount Wiki internetosoite: http://wiki.polycount.com/wiki/Edge_padding

Polycount Wiki. (2015). Light map. Haettu 24.4.2018, sivustolta Polycount Wiki internetosoite: http://wiki.polycount.com/wiki/Light_map

Polycount Wiki. (2016). Normal map. Haettu 1.2.2018, sivustolta Polycount Wiki internetosoite: http://wiki.polycount.com/wiki/Normal_map

Polycount Wiki. (2017). Polygon Count. Haettu 19.4.2018, sivustolta Polycount Wiki internetosoite: http://wiki.polycount.com/wiki/Polygon_Count

Polycount Wiki. (2017). ReTopologyModeling. Haettu 3.2.2018, sivustolta Polycount Wiki internetosoite: http://wiki.polycount.com/wiki/ReTopologyModeling

Polycount Wiki. (2014). Specular color map. Haettu 16.5.2018, sivustolta Polycount Wiki internetosoite: http://wiki.polycount.com/wiki/Specular_color_map

Polycount Wiki. (2018). Texture Baking. Haettu 4.4.2018, sivustolta Polycount Wiki internetosoite: http://wiki.polycount.com/wiki/Texture_Baking

Polycount Wiki. (2018). Texture Coordinates. Haettu 30.1.2018, sivustolta Polycount Wiki internetosoite: http://wiki.polycount.com/wiki/Texture_Coordinates

Polycount Wiki. (2015). Texture types. Haettu 30.1.2018, sivustolta Polycount Wiki internetosoite: http://wiki.polycount.com/wiki/Texture_types

Polycount Wiki. (2016). Texturing. Haettu 19.4.2018, sivustolta Polycount Wiki internetosoite: http://wiki.polycount.com/wiki/Texturing

Polycount Wiki. (2017). Tools. Haettu 21.4.2018, sivustolta Polycount Wiki internetosoite: http://wiki.polycount.com/wiki/Tools

Polycount Wiki. (2017). Topology. Haettu 21.4.2018, sivustolta Polycount Wiki internetosoite: http://wiki.polycount.com/wiki/Topology

Razer. (2018). Razer Phone. Haettu 27.3.2018, sivustolta Razer internetosoite: https://www.razer.com/mobile/razer-phone

Red Apple Tech. (2017, 3. helmikuuta). 2D Or 3D Art, Which Is the Better Choice for a Game Development Company. Haettu 28.1.2018, sivustolta Red Apple Tech internetosoite: http://www.redappletech.com/2d-3d-game-development-company/

Reinhardt, Z. (2017, 19. tammikuuta). How to set up Retopolgoy in Blender (Tutorial EN). Haettu osoitteesta https://www.youtube.com/watch?v=2hEHtKH55Us

Sakamoto, D. (2015). Hearthstone: How to create an Immersive User Interface. Haettu 28.1.2018, osoitteesta https://www.gdcvault.com/play/1022036/Hearthstone-How-to-Create-an

Sanakirja.org. (2018). Overdraw. Haettu 19.4.2018, sivustolta Sanakirja.org internetosoite: http://www.sanakirja.org/search.php?q=overdraw&l=3&l2=17

Sanakirja.org. (2018). Sprite. Haettu 3.2.2018, sivustolta Sanakirja.org internetosoite: http://www.sanakirja.org/search.php?id=174024&l2=3

Slick, J. (2017, 17. lokakuuta). 7 Common Modeling Techniques for Film and Games. Haettu 21.4.2018, sivustolta Lifewire internetosoite: https://www.lifewire.com/common-modeling-techniques-for-film-1953

Soenke, C. (2010, joulukuu). Current gen standards. Haettu 19.4.2018, sivustolta Polycount Forums internetosoite: http://polycount.com/discussion/comment/1249172#Comment_1249172

Sosa, J. (2014, 23. marraskuuta). Texture Coordinates - UV Tutorials & Threads. Haettu 30.1.2018, sivustolta Polycount Wiki internetosoite: http://wiki.polycount.com/wiki/Texture_Coordinates#UV_Tutorials_.26_Threads

Splash Damage Wiki. (2017). Specular Maps. Haettu 16.5.2018, sivustolta Splash Damage Wiki internetosoite: http://wiki.splashdamage.com/index.php/Specular_Maps

Super Evil Megacorp. (2018). Vainglory 5v5. Haettu 27.3.2018, sivustolta Google Play internetosoite: https://play.google.com/store/apps/details?id=com.superevilmegacorp.game&hl=fi

Supercell. (2018). Clash of Clans. Haettu 28.1.2018, sivustolta Google Play internetosoite: https://play.google.com/store/apps/details?id=com.supercell.clashofclans&hl=fi

Nimimerkki spek. (2008, 7. helmikuuta). What are specular maps? Haettu 19.5.2018, sivustolta GameDev.net internetosoite: https://www.gamedev.net/forums/topic/482021-what-are-specular-maps/

Nimimerkit SwdPwnzDggr, WarrenM. (2015, huhtikuu). What is the best retopo tool out there? Haettu 20.4.2018, sivustolta Polycount Forum internetosoite: http://polycount.com/discussion/151637/what-is-the-best-retopo-tool-out-there

Takahashi, D. (2017, 26. huhtikuuta). Newzoo: Mobile game revenue will frow 66% from $38 billion in 2016 to $65 billion in 2020. Haettu 27.3.2018, sivustolta VentureBeat internetosoite: https://venturebeat.com/2017/04/26/newzoo-mobile-game-revenue-will-grow-66-from-38-billion-in-2016-to-65-billion-in-2020/

Technopedia. (2018). Mobile Games. Haettu 27.3.2018, sivustolta Technopedia inernetosoite: https://www.techopedia.com/definition/24261/mobile-games

Terävä, T. (2017). Workflows for Creating 3D Game Characters. Haettu 3.6.2018, sivustolta Theseus internetosoite: http://www.theseus.fi/handle/10024/131241

Thomas, S. (2.6.2016). Official ZBrush Summit 2016 Presentation - Blizzard Entertainment Haettu osoitteesta https://youtu.be/eGHU8DI6fo4?t=31m3s

Trammell, K. (2016, 3. toukokuuta). Big Idea: "Baking". Haettu 4.4.2018, sivustolta CG Cookie inernetosoite: https://cgcookie.com/articles/big-idea-baking

Toledo, P. (2010, 21. lokakuuta). Brief Considerations About Materials. Haettu 19.5.2018, sivustolta Manufato inernetosoite: http://www.manufato.com/?p=902

Unity. (2018). Art Asset best practice guide. Haettu 30.1.2018, sivustolta Unity Documentation inernetosoite: https://docs.unity3d.com/Manual/HOWTO-ArtAssetBestPracticeGuide.html

Unity. (2018). Company Facts. Haettu 3.2.2018, sivustolta Unity inernetosoite: https://unity3d.com/public-relations

Unity. (2018). Graphics Methods. Haettu 3.2.2018, sivustolta Unity Documentation inernetosoite: https://docs.unity3d.com/Manual/MobileOptimizationGraphicsMethods.html

Unity. (2018). Light Probes. Haettu 27.1.2018, sivustolta Unity Documentation inernetosoite: https://docs.unity3d.com/Manual/LightProbes.html

Unity. (2018). Specular. Haettu 16.5.2018, sivustolta Unity Documentation inernetosoite: https://docs.unity3d.com/Manual/shader-NormalSpecular.html

Unity. (2018). Usage and Performance of Built-in Shaders. Haettu 3.2.2018, sivustolta Unity Documentation inernetosoite: https://docs.unity3d.com/Manual/shader-Performance.html

Van Der Bryl, L. (2002). Photorealistic Texturing for Dummies. Haettu 30.1.2018, sivustolta Unity inernetosoite: http://www.3dlinks.com/downloads/texturing.pdf

Vicente, M. (26.10.2016). Official ZBrush Summit 2016 Presentation - Blizzard Entertainment Haettu osoitteesta https://youtu.be/eGHU8DI6fo4?t=1h7m27s

Walters, A. (2014, huhtikuu). How 3D Modeling Changed Online Gaming. Haettu 10.4.2018, sivustolta Citizen Tekk internetosoite: http://www.citizentekk.com/3d-modeling-changed-online-gaming/

Wang, Y. (2018). The History of Mobile Phone Games. Haettu 27.3.2018, sivustolta Sutori internetosoite: https://www.sutori.com/story/the-history-of-mobile-phone-games

Webopedia. (2018). Side-scroller. Haettu 2.6.2018, sivustolta Webopedia inernetosoite: https://www.webopedia.com/TERM/S/side_scroller.html

Wheeler, R. (Nimimerkki Zephyris) (2008, 18. maaliskuuta). UV mapping. Haettu 30.1.2018, sivustolta Wikipedia - Vapaa tietosanakirja internetosoite: https://en.wikipedia.org/wiki/UV_mapping

Wikipedia. (2018). 2.5D. Haettu 28.1.2018, sivustolta Wikipedia - Vapaa tietosanakirja internetosoite: https://en.wikipedia.org/wiki/2.5D

Wikipedia. (2018). 2D computer graphics. Haettu 3.2.2018, sivustolta Wikipedia - Vapaa tietosanakirja internetosoite: https://en.wikipedia.org/wiki/2D_computer_graphics

Wikipedia. (2018). 2D-grafiikka. Haettu 3.2.2018, sivustolta Wikipedia - Vapaa tietosanakirja internetosoite: https://fi.wikipedia.org/wiki/2D-grafiikka

Wikipedia. (2018). 3D computer graphics. Haettu 3.2.2018, sivustolta Wikipedia - Vapaa tietosanakirja internetosoite: https://en.wikipedia.org/wiki/3D_computer_graphics

Wikipedia. (2018). 3D modeling. Haettu 3.2.2018, sivustolta Wikipedia - Vapaa tietosanakirja internetosoite: https://en.wikipedia.org/wiki/3D_modeling

Wikipedia. (2018). Digital Sculpting. Haettu 3.4.2018, sivustolta Wikipedia - Vapaa tietosanakirja internetosoite: https://en.wikipedia.org/wiki/Digital_sculpting

Wikipedia. (2018). Kuvataajuus. Haettu 3.2.2018, sivustolta Wikipedia - Vapaa tietosanakirja internetosoite: https://fi.wikipedia.org/wiki/Kuvataajuus

Wikipedia. (2018). Lightmap. Haettu 24.4.2018, sivustolta Wikipedia - Vapaa tietosanakirja internetosoite: https://en.wikipedia.org/wiki/Lightmap

Wikipedia. (2018). Nokia N-Gage. Haettu 27.3.2018, sivustolta Wikipedia - Vapaa tietosanakirja internetosoite: https://fi.wikipedia.org/wiki/Nokia_N-Gage

Wikipedia. (2018). Normal Mapping. Haettu 3.2.2018, sivustolta Wikipedia - Vapaa tietosanakirja internetosoite: https://en.wikipedia.org/wiki/Normal_mapping

Wikipedia. (2018). Polygonal Modeling. Haettu 9.4.2018, sivustolta Wikipedia - Vapaa tietosanakirja internetosoite: https://en.wikipedia.org/wiki/Polygonal_modeling

Wikipedia. (2018). Renderöinti. Haettu 3.2.2018, sivustolta Wikipedia - Vapaa tietosanakirja internetosoite: https://fi.wikipedia.org/wiki/Renderointi

Wikipedia. (2018). Specularity. Haettu 16.5.2018, sivustolta Wikipedia - Vapaa tietosanakirja internetosoite: https://en.wikipedia.org/wiki/Specularity

Wikipedia. (2018). T-vertices. Haettu 3.2.2018, sivustolta Wikipedia - Vapaa tietosanakirja internetosoite: https://en.wikipedia.org/wiki/T-vertices

Wikipedia. (2018). UV mapping. Haettu 30.1.2018, sivustolta Wikipedia - Vapaa tietosanakirja internetosoite: https://en.wikipedia.org/wiki/UV_mapping

Wikipedia. (2018). Video Game Graphics. Haettu 3.2.2018, sivustolta Wikipedia - Vapaa tietosanakirja internetosoite: https://en.wikipedia.org/wiki/Video_game_graphics

Wikipedia. (2018). ZBrush. Haettu 3.4.2018, sivustolta Wikipedia - Vapaa tietosanakirja internetosoite: https://en.wikipedia.org/wiki/ZBrush

Williamson, J. (2009, 17. elokuuta). Re-Topologize a Game-Ready Alien Head in Blender. Haettu 20.4.2018, sivustolta Envato Tuts+ internetosoite: https://cgi.tutsplus.com/tutorials/re-topologize-a-game-ready-alien-head-in-blender--cg-817

Wilson, J. (2015, 1. lokakuuta). Physically-Based Rendering, And You Can Too! Haettu 1.2.2018, sivustolta Marmoset LLC internetosoite: https://www.marmoset.co/posts/physically-based-rendering-and-you-can-too/

Wiro, A. (2005). Subdivision Body Model by Wiro. Haettu 3.2.2018, sivustolta Moridin internetosoite: http://www.moridin.com/tutorials/wiro/female_body/index8.php

Yot, R. (2012, 19. maaliskuuta). Retopology... [does character need to be retopologized before animating?]. Haettu 20.4.2018, sivustolta Foundry internetosoite: http://community.foundry.com/discuss/topic/64323