OULUN AMMATTIKORKEAKOULU

LIVE2D -OHJELMISTON KÄYTTÖ PELINKEHITYKSESSÄ

2.5D-grafiikat peliympäristössä

Jenna Qvickman
Opinnäytetyö AMK
Syksy 2024
Viestinnän tutkinto-ohjelma
Oulun ammattikorkeakoulu

TIIVISTELMÄ

Oulun ammattikorkeakoulu
Viestinnän tutkinto-ohjelma
Visuaalisen suunnittelun suuntautumisvaihtoehto

Tekijä(t): Jenna Qvickman
Opinnäytetyön otsikko: Live2D -ohjelmiston käyttö pelinkehityksessä, 2.5D-grafiikat peliympäristössä
Työn ohjaaja(t): Tuukka Uusitalo
Työn valmistumislukukausi ja -vuosi: Syksy 2024
Sivumäärä: 79

Tutkielmassa tarkastellaan *Live2D*-animaatiota ja *2.5D*-grafiikkaa videopeleissä käyttäen Live2D Cubism-ohjelmaa, joka mahdollistaa 2D nukke-, pala- ja luurankoanimaatioiden luomisen.

Tutkielmassa perehdytään 2.5D-grafiikan määritelmiin ja sovelluksiin. Live2D Cubismin työkalut esitellään ja käydään läpi hahmon luonnin eri vaiheet. Lisäksi käsiteltiin muita pelejä, joissa on käytetty nukke-, pala- ja luurankoanimaatioita, ja esiteltiin omia peliprojekteja, kuten *Jellyfish*-peliä.

Tutkielman keskeinen tulos oli, että Live2D Cubism on tehokas työkalu hahmoanimaatioiden luomiseen pelinkehityksessä. Sen avulla voidaan saavuttaa sujuvia ja luonnollisia liikkeitä sekä rikkaampaa visuaalista ilmettä pelimaailmassa.

Live2D-animaation heikkouksia ovat tarkka suunnittelun tarve ennen animaatioiden tekoa ja suuri ajallinen investointi oppimisvaiheessa. Dokumentaatio oli toisinaan rajallista. Lisäksi Live2D-ohjelmisto on maksullinen, mikä voi olla haaste pienille kehittäjille.

Johtopäätöksenä todettiin, että Live2D-tekniikkaa kannattaa hyödyntää laajemmin pelinkehityksessä, erityisesti projekteissa, joissa yhdistetään 2D- ja 3D-grafiikkaa. Kehitysehdotuksena esitettiin, että Live2D Cubism-ohjelman käyttöä tulisi laajentaa ja parantaa tarjoamalla enemmän resursseja ja oppimateriaaleja, jotta ohjelmisto olisi helpommin lähestyttävä peligrafiikasta kiinnostuneille kehittäjille.

ABSTRACT

Oulu University of Applied Sciences
Degree Programme in Communications
Option of Visual Design

Author(s): Jenna Qvickman
Title of thesis: Title of thesis: Usage of Live2D software in game development, 2.5D-graphics in a game environment
Supervisor(s): Tuukka Uusitalo
Term and year when the thesis was submitted: Autumn 2024
Number of pages: 79

Thesis examines Live2D animation and 2.5D graphics in video games using the Live2D Cubism software, which enables the creation of 2D puppet-, cutout-, and skeletal animations.

The thesis delves into the definitions and applications of 2.5D graphics. Live2D Cubism's tools were showcased, and the stages of character creation were demonstrated. Additionally, the thesis discussed other games that employ similar animations and presented personal game projects.

The main finding of the thesis was that Live2D Cubism is an effective tool for creating character animations in game development. It allows for smooth and natural movements, enhancing the visual richness of the game world.

However, the limitations of Live2D animation include the need for precise planning before creating animations and the significant time investment required to master the techniques. Documentation was occasionally limited. Moreover, Live2D software is paid, which can be a challenge for small development teams.

In conclusion, it was suggested that Live2D techniques should be more widely utilized in game development, especially in projects combining 2D and 3D graphics. It was recommended to expand and improve the use of Live2D Cubism by providing more resources and learning materials to make the software more accessible to developers interested in game graphics.

SISÄLLYS

SANASTO

2D-grafiikka	Kaksiulotteinen grafiikka, jossa yleisimmin käytetään spritejä (Rasinkangas 2014. 6,8.).
2.5D-grafiikka	Kolmiulotteiselta vaikuttava grafiikka, joka perspektiivin lukituksen avulla esitetään kaksiulotteisesti (Ste 2021.).
3D-grafiikka	Kolmiulotteinen tietokone grafiikka, joka rakentuu polygoneista (Mansikka 2015, 19.).
Frame ja FPS	Ruutu ja ruutuja sekunnissa, animaatioiden ja pelien termi, jolla mitataan nopeutta (Brunner 2024.).
Live2D	kaksiulotteisen tasoisen kuvan leikkuu osiin, riggaus ja animointi, tekniikka tai ohjelma (Wikipedia 2024a.).
Mesh	Viereisiä polygoneja, jotka yhdistyvät vertekseillä, monikulmioverkko (Awati 2024.).
Polygoni	Monikulmio, joka koostuu kolmesta tai useammasta verteksistä kolmiulotteisessa avaruudessa (Mansikka 2015b, 22.).
Rig	Mallin osien liikuttelun määrittäminen kontrolliobjekteilla ja sen vaikuttaminen muihin osiin (Adobe 2024.).
Sprite, Spritesheet	Kaksiulotteinen kuva (sprite) ja kuva-arkki (sheet), jossa monta kuvaa hahmon liikeanimaatiosta (Spritesheet Editor s.a.).
Tekstuuriatlas	Tekstuurit ja tasot pakattuna yhdessä kuvassa tilan säästämiseksi (Lark Editorial Team 2024a.).
Verteksi	(eng. Vertex) Polygonin kärkipiste tai kulma, jonka välille polygoni muodostuu (Mansikka 2015, 22.).

1 JOHDANTO

Opinnäytetyössä tutkitaan tarkemmin Live2D-animaatiota, joka sisältää 2D-luuranko-, pala- ja nukkeanimaation tekniikoita. Ohjelmistossa on muista animaatio-ohjelmistoista tuttuja työkaluja ja tekniikoita, jopa 3D-mallinnuksenkin termistöjä ja työkaluja hyödynnetään, vaikka lopputuloksena onkin kaksiulotteista animaatiota – tai tarkemmin – 2.5D-animaatiota. Tässä tutkielmassa tutkitaan ohjelman käyttöä pelinkehityksen kannalta, miten ohjelman tuotokset toimivat pelimoottorissa.

Opinnäytetyötä edeltävässä käytännön osuudessa kehitettiin 2.5D-peli, *Jellyfish*. Peli on tuotettu Unity-moottorilla, jossa live2D:nä animoitua meduusaa ohjataan läpi kolmiulotteisesti mallinnettujen merenalaisten kenttien. Aiemmin olin käyttänyt live2D:tä myös Oulu Game Labissa julkaistussa Unityllä tuotetussa pelissä *Gnome Attack*, ja Scream Game Jameilla tehdyssä *Ghost Trap*issä, joka oli tuotettu Unreal Enginellä.

Koronapandemian aikana Live2D-ohjelmisto nousi keskeiseksi osaksi opintoja lisääntyneen internetin käytön myötä. Virtuaaliset suoratoisto vaikuttajat, VTuberit, kiinnittivät huomiota reaaliaikaisten, sulavien avatarien animaatioillaan. Tämä johti Live2D-ohjelmiston käytön tutkimiseen ja hyödyntämiseen.

Live2D-ohjelmiston avulla luotiin useita avatareja sekä itselle että ystäville, ja sitä hyödynnettiin myös erilaisissa kouluprojekteissa: E. Ojalan kirjoittaman *Nanostrofi*n yhteydessä luotiin virtuaalinen lukukokemus animaatioineen. Lisäksi Live2D:tä hyödynnettiin kurssilla lisätyn todellisuuden projektiin Arilyn-sovelluksessa. Live2D mahdollisti tarkkojen kuvitusten laadun säilyttämisen ja kevyen animoimisen, käyttäen tekniikoita, jotka olivat ennalta tuttuja 3D-mallinnuksesta.

Pelialalla oli kertynyt jonkin verran kokemusta jo ennen opintoja ja opintojen aikana osallistuttiin erilaisiin Game Jam -tapahtumiin. Opintojen jälkeen on tarkoitus palata pelinkehityksen pariin ja hyödyntää siellä tutkinnon aikana opittuja Live2D-animaatioita.

2 2.5D

Luvussa avataan enemmän tutkielmassa esiin nousevaa 2.5D-grafiikkaa ja selvennetään mitä 2.5D-peligrafiikalla tarkoitetaan. Kyseinen termi pitää sisällään laajan skaalan peligenrejä, eikä yksiselitteistä määritelmää löydä kyseiselle graafiselle tyylisuunnalle.

“D” on lyhenne englannin kielen sanasta ulottuvuus, *dimension*. 2D-pelit ovat siis kaksiulotteisessa maailmassa, jossa hallitsee leveys ja korkeus. Maailma on litteä, kuin paperinpala. 3D-pelissä mukaan tulee kaksiulotteisuuden lisäksi syvyys, eli maailmaan voi lähes upota sisälle, tuntuen aidolta tilalta. 2.5D-grafiikka putoaa näiden termien väliin; kamera ja liikerata lukitsevat pelaajan tietyille akseleille, eikä pelaaja pääse liikkumaan yhtä vapaasti kuin puhtaassa 3D-ulottuvuudessa, jossa liike ja kamera voivat vapaasti kulkea geometrisessä koordinaatistossa X-, Y- ja Z-akselissa, eli leveydessä, korkeudessa ja syvyydessä. Grafiikassa voidaan hyödyntää laidasta laitaan niin 3D-grafiikasta 2D-bittigrafiikkaan, sillä perspektiivi tai liike määrittelevät 2.5D-grafiikan. 2.5D-grafiikka syntyi kolmiulotteisen grafiikan tuotannon vaikeudesta, lähinnä kehityskustannuksista tai vaatimuksesta liian korkeasta tehosta sen ajan maailman alustoille, joten 2.5D kiertää luovilla ratkaisuilla suoraa 3D-grafiikkaa. Pääasiallinen tarkoitus on luoda illuusio kolmiulotteisesta syvyydestä (Mansikka 2015, 12-23; Pietiläinen 2020, 8-11.).

2.5D-grafiikka pitää sisällään toisenkin alagenren; tiettyjen pelien kohdalla voidaan grafiikkaa kutsua pseudo-3D:ksi (Kevuru Games 2024.). Tomohiro Nishikadon tekemä ja Taiton julkaisema *Interceptor* (1975) onkin ensimmäisiä pseudo-3D-pelejä. Interceptor on pelaajan perspektiivistä kuvattu lentotaistelu simulaattori *arcade*-pelikoneelle, josta valitettavasti ei ole säilynyt montaa toimivaa pelikonetta tähän päivään. Pelissä vihollishävittäjien spritejen kokoa skaalataan luomaan etäisyyden illuusiota; hävittäjän ollessa suuri, se on lähellä ja kaukana, kun se skaalataan pieneksi. Hävittäjää pystyy liikuttamaan X- ja Y-akselilla (Giantbomb 2019.).

Tunnetuin Pseudo-3D edustaja lienee kuitenkin id Softwaren vuoden 1993 *DOOM*, joka tuotettiin John Carmackin suunnittelemalla 3D-pelimoottorilla.

Vaikka ympäristö onkin kolmiutteista, viholliset ja objektit ovat kaksiulotteisia spritejä, jotka on kuvattu useasta eri kuvakulmista, vaihtuen riippuen mihin suuntaan vihollinen osoittaa. DOOMissa ei myöskään ole päällekkäisiä kerroksia vaan jokainen kenttä sisältää sektoreita, jonka pohjana on kaksiulotteinen kuvio sekä kaksi numeroa määrittämässä kuinka korkea kerroksen pohja ja katto on. Tämän avulla DOOMiin voi tehdä esimerkiksi rappuset tai dynaamisemmin käytettynä hissin, jossa kerros ja katto liikkuvat samanaikaisesti merkatussa sektorissa. Sektoripohjaisessa moottorissa on omat rajoitteensa: jokaisen sektorin on oltava tasainen, eivätkä seinät voi olla kuin suoria tai täydellisen vertikaaleja. Vihollisten ja räjähdysten törmäysfysiikat eli merkattu alue, jonka objekti tilasta vie, ovat loputtoman korkeita DOOMissa yksinkertaistamaan törmäysfysiikoita. Niitä ei siis voi ylittää, vaikka visuaalisesti vihollinen ei olisikaan katonrajassa. DOOMissa pelaaja ei voi katsoa ylös tai alas, koska kolumnipohjainen renderöinti vääristäisi näkymän perspektiivin (Borogk 2021.). Pseudo-3D flirttailee niin 2D- kuin 3D-grafiikoiden kanssa ja vaikeuttaa selkeätä kuvausta 2.5D:een määrittelyssä, rikkoen jotain määritelmiä, mutta toteuttaen toisia.

2.1 Markkinat, studiot ja alustat

Nykyään suuret pelistudiot, jotka tuottavat pelejä tuoreimman sukupolven alustoille, eivät juuri käytä kaksiulotteisuutta. Nykyajan pelimoottorit ja alustat pyörittävät suuria ja lähes rajattomia kolmiulotteisia valtakuntia (Asikainen & Taskinen 2023, 14.). Peliala nimittää eri kategorioihin peli tuotantojen koot, vastapäinä ollen *indie* ja *AAA*. *Triple-A* pelit ovat suurien studioiden pelejä, joissa budjetti saattaa ylittää miljoonat. Tiimit ovat suuria, sadoissa tai vaikeasti mitattavissa; esimerkiksi, jos studio ulkoistaa jotain asioita muille firmoille. Näin suurilla resursseilla, suuret pelistudiot tähtäävät usein realistiseen ja yksityiskohtaiseen grafiikkaan (RocketBrush Studio 2023.). Kaksiulotteisuus onkin tullut omaksi tyylisuunnakseen. Etenkin indie-pelikehittäjät, (eng. independent game developer) itsenäiset pelikehittäjät, suosivat 2.5D sen kustannustehokkuuden ansiosta, itsenäisellä kehittäjällä harvemmin on resursseja, rahoitusta tai työntekijöitä, joten erilaiset rajoitteet helpottavat pelin tuotantoa (Helpshift 2023.). Kevyempää suorituskykyä ja selkeämpää visuaalista viestiä tarvitaan mobiilipeleissä. Mobiilipeleissä

vähemmän on enemmän alustan pienen näyttökoon ja heikomman suorituskyvyn takia kuin esimerkiksi viimeisimmissä pelikonsolissa tai pöytätietokoneissa. Mobiilipelaaminen on yleistynyt, vieden 56 % pelialan markkinaosuudesta, ja jatkaa kasvuaan sen ollen lähes jokaisen käden ulottuvilla (Data.ai 2023.).

“Do you guys not have phones?” (Cheng 2018.).

Mobiilipelaaminen on siis otettava varteenotettavasti huomioon pelinkehityksessä alustaa ja kohdeyleisöä suunniteltaessa.

2.2 Tyyli

2.5D ei lukitse kehittäjää tiettyyn tyyliin, mutta 2.5D on useasti lähtökohtaisesti optimoidumpi tai helpommin toteutettavissa kuin täysi 3D. Täysin 2D-pelejä ei tätä nykyä enää juuri valmisteta koska 2.5D muuntaminen voi olla niin pienestä kiinni, kuin myöhemmin tutkielmassa käsiteltävän parallax scrollauksen lisääminen taustaan (Knight 2021.).

2.5D-peleissä voi olla niin 3D-assetteja, jotka ovat kolmiulotteisessa ulottuvuudessa polygonisesti mallinnettuja, spritejä ja spritesheettejä, joissa kaksiulotteinen kuva määrittää grafiikan. Spritesheetissä, yhteen kuva-arkkiin animoidaan kuva kuvalta esimerkiksi hahmon liike. Spritesheettinä grafiikka ei vie liiaksi muistia, kuin yksittäisinä spriteinä. Grafiikoissa voidaan käyttää vektoreita, joissa matemaattisilla yhtälöillä muodostetaan viivoja visuaalin luomiseksi. Käytännössä 2.5D-grafiikassa voidaan käyttää laidasta laitaan erilaisia grafiikoita (Pietiläinen 2020, 7-24; Asikainen & Taskinen 2023, 16-17.).

Resoluutio vaikuttaa merkittävästi pelien tyyliin. Varhaisissa peleissä teknologian rajoitteet pakottivat luoviin ratkaisuihin, kuten selkeisiin siluetteihin rajallisella määrällä pikseleitä. Esimerkiksi *NES*:in 8-bittinen teho mahdollisti vain 64 värin samanaikaisen esiintymisen ruudulla, mikä rajoitti visuaalista ilmaisua. Nykyään 8-bittinen grafiikka on synonyymi pikseligrafiikalle, jossa hahmot ja maailma koostuvat pikseleistä ja rajoitetusta väripaletista (Pocket Gamer 2015.).

Pikseligrafiikassa käytetään spritesheettejä, jotka voidaan piirtää pikseli kerrallaan tai rotoskooppaamalla, kuten *Prince of Persiassa* (1989). Viime vuosina tehokkaaksi tavaksi tuottaa pikseligrafiikkaa on noussut Motion Twinin *Dead Cells* (2017), jossa hyödynnetään 3D-mallinnuksia ja animaatioita esirenderöimällä ne kaksiulotteisiksi kuviksi. Kuvia *kvantisoidaan* ja värit rajoitetaan, jotta saadaan pikseligrafiikan tyyliä (Smeaf 2023.). Dead Cells käyttää myös *normal map*peja, normaali karttoja, jotka ovat 3D-grafiikasta tuttuja tekstuurikarttoja, antamaan tietoa objektin muodoista ja orientaatioista (Pluralsight 2022.).

Kun rajoitteet poistetaan ja siirrytään ohjelmistollisesti tai alustallisesti korkeampiin grafiikoihin, on tärkeää hallita perusasiat, kuten selkeä muotokieli ja värit. Ilman suunnittelua voi helposti tuottaa ristiriitaisia graafisia elementtejä. Pelien visuaalisia tyylejä ovat realistinen, tyylitelty ja abstrakti, joiden alle tulee tarkennettuja tyylejä, kuten fotorealistinen ja sarjakuvamainen (Asikainen & Taskinen 2023, 17-19.). Tyylisuuntia on vielä enemmänkin, mutta tyylit polveutuvat kolmesta päätyylistä. Tyyleissä nousee esille muotojen, värien ja liikkeen vaikutus pelaajan psyykkeeseen, esimerkiksi väreillä ollen vaikutusta pelaajan verenpaineeseen, metabolismiin tai mielentilaan (Garver, Adamo-Villani, Dib 2018, 2-3.).

Myöhemmin tutkielmassa käsitellään eri perspektiivejä ja niiden määritelmiä, mutta tässä kappaleessa on hyvä korostaa perspektiivin vaikutusta. Esimerkiksi suomalainen Colossal Order joutui kritiikin kohteeksi, kun heidän pelinsä *Cities: Skylines 2* (2023) ei julkaisussa sisältänyt kevyempiä LOD-malleja (level of detail, yksityiskohtaisuus) ihmishahmoista, mikä heikensi pelin suorituskykyä, etenkin kun pelinäkymä on useasti isometrisesti kaupunkinäkymässä, eikä maanpinnalla katutasossa kuvattuna (Yin-Poole 2023.). Tämä korostaa suunnittelun tärkeyttä visuaalisen luettavuuden ja optimoinnin kannalta.

3 ANIMAATIO

Kappaleessa käsitellään animaation teoriaa Disneyn animaattoreiden ohjekirjan kautta ja syvennytään selventämään nukke-, pala- ja luurankoanimaatioita, joiden sisälle tutkielman Live2D Cubismikin sisältyy.

Käsittelyssä on myös proseduraalinen animaatio, jossa parametreistä ja luurangoista koottuja assetteja voi myös hyödyntää. Esiin nousee myös tutkielmassa aiemmin mainittu parallaksivieritys, tai parallax scrolling, jota monissa sivulta kuvatuissa videopeleissä esiintyy.

3.1 Animaation 12 periaatetta

Pelianimaatiot noudattavat enemmän tai vähemmän Disneyn animaattoreiden ohjekirjaa *The Illusion of Life: Disney Animation*, animaation 12 periaatteesta. Periaatteet käydään pintapuoleisesti läpi. Syvennymme vielä joidenkin periaatteiden kohdalla, miten niitä hyödynnetään pelille sopivammiksi.

Litistyminen ja venyminen, Squash and Stretch

Hahmojen venyvyys noudattaa tilavuutta. Todellisuudessakin esiintyy venymistä tai litistystä, tämä luo eloa animaatioon (Thomas & Johnston 1995, 46-49.). Esimerkkinä *Ori and the Blind Forest* (2015), (Kuva 1) jossa hahmon hypätessä tai pudotessa, hahmo venyy vaikuttaakseen liikkuvalta tai nopealta, ja litistyy

osuessaan maahan, luoden mielikuvan painosta ja joustavuudesta.

KUVA 1. GDC -esityksestä editoitu kuvakaappaus pelihahmojen venymisestä ja litistymisestä (Benson 2015.).

Antisipaatio, Anticipation

Ennen varsinaista liikkettä animoidaan siihen valmistavaa liikettä (Thomas & Johnston 1995, 49-51.), esimerkiksi kuvassa Team Cherryn *Hollow Knight*issa (2017) pelaajahahmo voi ladata voimakkaampaa iskua (Kuva 2), vetäen käden taaksepäin, pelaajan saaden odotuksen tunteen ja valmistautumisen seuraavaan liikkeeseen.

KUVA 2. Hollow Knight -pelin hahmoliikkeet on selkeästi esitetty (Hoeschen 2024.).

Yleisesti “hitaat” tai etenkin sitoutumista vaativat pelaajan toiminnot, joita ei voi “keskeyttää” ovat epäsuosittuja videopeleissä. Ne tekevät hitaamman ja rankaisevamman pelituntuman. Joissain peleissä se on tarkoitusperäistä, toiminta täytyy miettiä ja ajoittaa tarkasti. Pelianimaatioissa antisipaatio monesti korvataan jälkeisanimaatiolla, *windup*illa, jotta saadaan nopeita responsiivisa toimintoja, jossa säilyy myös animaation voimakkuus (Floyd 2020a.).

Lavalle asettelu, Staging

Teatterin periaatteisiin kuuluu että, tapahtumat kuvattava selkeästi (Thomas & Johnston 1995, 51-54.), animaatiossa ja peleissä tämä on kameran ja kuvakulman keskiössä. Sommittelu ja hahmojen sijainti kuuluvat tähän periaatteeseen myös, tärkein nyrkkisääntö on ylläpitää silmäillen luettavat siluetit. Kenttäsuunnittelijan tärkein tehtävä on tuottaa luettavat ja helposti ymmärrettävät ympäristöt, joissa hyödynnetään pelin sisäisiä mekaniikkoja. Hyvät kentät eivät vaadi liikaa

immersiota häiritseviä viittauksia oikeisiin suuntiin, välillä kuitenkin leikitellen salaisilla piiloilla, jotka lisäävät pelaajan halua tutkia ympäristöjä tarkemmin (Garcia 2024a.).

Animointi suoraan tai asennosta asentoon, Straight Ahead and Pose to Pose

Suoraan animoinnissa liikettä tehdään ruutu ruudulta ensimmäisestä asennosta lähtien. Liike on spontaania sekä virtaavaa, ollen suosittu toimintakohtauksissa. Asennosta asentoon tehdään ensin *keyframe*t, eli avainruudut, liikkeen huippukohdat, joiden välille tehdään *inbetweens*it, eli väliruudut, punomaan animaatio yhtenäiseksi liikkeeksi. Lopputuloksena on selkeä ja johdonmukainen animaatio (Thomas & Johnston 1995, 54-56.).

Seuraava ja päällekkäinen liike, Follow Through and Overlapping Action

Seuraava liike tulee jäljessä, hahmon juostessa eteenpäin, hiukset voivat jatkaa liikettä heilahtaen eteenpäin, kun hahmo pysähtyy nopeasti (Thomas & Johnston 1995, 57-60.). Saman efektin voi luoda tekemällä pelihahmon hiuksiin vaikka “fysiikat”, jotka aktivoituvat tiettyjen osien liikkeistä, kuten pään heilumisesta. Päällekkäinen liike tapahtuu samanaikaisesti muun liikkeen kanssa tai yhteisesti. Liikkeet luovat elollisuutta animaatioon (Floyd 2021.).

Hidastus alussa ja lopussa, Slow In and Slow Out

Nimi selittää hyvin itsessään periaatteen: Paraskaan formula-auto ei saa täyttä kiihtyvyyttään heti kaasun tallattua pohjaan. Liikkeen alkaessa liike on hitaampi kuin huipussaan ja valmistautuessa pysähtymiseen se taas hidastuu. Avainkuvien on hyvä antaa elää pidempään lisäämällä ruutuja (eng. frame) sen ympärille (Thomas & Johnston 1995, 60.). Pelissä tekniikalla voi lisätä painavuuden tuntua suuriin hahmoihin, tai muutenkin elävöittää animaatioita (Floyd 2020b.).

Kaaret, Arcs

Liikkeet monesti kaareutuvat tai lainehtivat luonnossa. Kaarta vähentämällä voi myös tehdä illuusion nopeasta tai elottomasta liikkeestä (Thomas & Johnston 1995, 60-61.). Lähes kaikissa peleissä, joissa on hyppymekaniikka, on niissä

myös kaaret. Vaikeudeksi pelianimaatioissa kuitenkin nousee, jos kamera on vapaasti pelaajan ohjattavissa, jolloin animaattorin täytyy tehdä animaation kaaret luettavaksi kaikista kuvakulmista (Floyd 2024.).

Toissijainen liike, Secondary Action

Toissijainen liike lisää luonnetta ja tunnelmaa animaatioon, mutta sen ei saa viedä huomiota ensisijaiselta liikkeeltä. Toissijaisena liikkeenä voi olla surullinen kohtaus, jossa taustahahmo pyyhkii silmiään (Thomas & Johnston 1995, 61-62.).

Ajoitus, Timing

Perinteisen animaation termi, jossa animoidaan ykkösillä ja kakkosilla. Kakkosissa sama piirros saattoi toistua toisessa peräkkäisessä ruudussa, säästäen aikaa. Ykkösiä suositaan esimerkiksi tarkkuutta ja huolellisuutta vaativammissa, tai nopeutta ja räjähtävyyttä vaatimissa liikkeissä. Perinteisessä animaatiossa ajoitus siis viittaa siihen, kuinka monta ruutua liike kestää ja kuinka ruudut jakautuvat liikkeen aikana (Thomas & Johnston 1995, 62-63.). *Cuphead* (2017) on peli, joka tunnetaan erityisesti sen 1930-luvun piirrettyjen tyylistä ja animaatiosta. Cupheadin animaatiot suunniteltiin mahdollisimman perinteisesti, paperille piirtäen, ykkösiä ja kakkosia hyödyntäen (Clark 2017.). Cupheadista kuvakaappaus pelistä (Kuva 3), jonka voisi kuvitella olevan suoraan vanhasta animaatiosta.

KUVA 3. Pelin Steam -sivuilta mainoskuva pelistä (Studio MDHR Entertainment Inc. 2022.).

Puhuttaessa ajoituksesta peleissä, voidaan tarkoittaa kuitenkin vähän eri periaatteita, kuin suoraan perinteisten ykkösien ja kakkosien animaation tekniikoita. Sulavassa pelissä on tarkoin mitattuja ajoituksia eri toiminnoille, jotka lasketaan ruuduilla, eli yksittäisien ruutujen määrällä sekunnissa (eng. frames per second, lyhenne FPS). Liikkeen sulavuus ja luonnollisuus hoidetaan animaatioiden ajoituksella, sekä pelaajan ja pelin tapahtumien välisellä synkronoinnilla, johon liittyy pelaajan komennot ja pelin vastauksen viive (Floyd 2019.).

Liioittelu, Exaggeration

Liioittelu painottaa asioiden ytimen korostamista: raivon piti olla vihaisempaa ja onnen iloisempaa, mutta raja on, ettei ylitetä uskottavuuden rajaa haitaten katselukokemusta (Thomas & Johnston 1995, 63-64.). Tappelupeleissä liikkeissä suositellaan liikkeiden liioittelua, jotta nopeatempoiset liikkeet on helppo havaita samalla ylläpitäen liikkeen luettavuuden. Jokainen liike on erittäin selkeä ja osa animaatioiden ruuduista on rajoitettu napakan tuntuman saamiseksi. Tämän lisäksi jokaisella hahmolla on selkeästi eri liike tyyli pelkästään *idle*-animaatiossaan, eli tyhjäkäydessään, luoden voimakkaita hahmopersoonia pelkästään idlaavien animointien puolesta (Root 2019.).

Piirustustaito, Solid Drawing

Yksiselitteinen nimensä kanssa, animaattoreilla on hyvä osata piirtää malli ja anatomiaa eri kulmista, oikeilla tilavuuksilla samalla pysyen määritellyssä tyylisuunnassa (Thomas & Johnston 1995, 64-65.). Peleissäkin selkeä ja miellyttävä, hyvin ja yhteneväisesti tuotettu grafiikka nostavat pelikokemusta. Hyvä esimerkki Vanillawaren pelit, kuten vaikka *Dragon's Crown* (2013); pelin hahmot ja ympäristöt ovat yksityiskohtaisesti piirrettyjä. Studion perustajista George Kamitani nimittääkin piirtotyyliä *tebineriksi*, käsin muovailluksi. Yksityiskohtaiset kädenjäljet paljastavat piirrokset antavat peleille taiteellisen tyylin. Taide yhdistettynä Adobe Animaten (entinen Flash) kaltaisiin animaatio-ohjelmistoihin pelien animaatiotyylin puolesta luovat kolmiulotteisen vaikutelman, mutta Kamitani ei ole erikseen

maininnut ohjelmistoa (Wikipedia 2024b.). Käsittelen lisää Vanillawarea myöhemmin tutkielmassani.

Viehättävyys, Appeal

Hahmosuunnitelun osa-alue, jossa jokaisen hahmon tulee olla viehättäviä, jotta katsoja samaistuisi niihin. Vihollisistakin tehdään omalla tavalla viehättäviä herättämään katsojassa empatiaa. Hahmosuunnitteluun ei sisälly pelkkä ulkonäkö ja siluetti, vaan liike vaikuttaa myös viehättävyyteen (Thomas & Johnston 1995, 66-68.). Japanissa viehättävyys on hiottu viimeisen päälle, monien tunnistavan japanin kielen sanan “kawaii” (かわいい), eli söpö. Kontekstissa sitä käytetään viehättävyyden synonyymina, kawaii voi ohjata käyttäjän tunteita; vähentää stressiä sekä parantaa suoritusta ja henkilökohtaista sitoutumista kohdetta kohtaan. *Animal Crossing*ia (2001) Wiredin toimittaja Cecilia D'Anastasio sanookin *“loputon kawaii kapitalismin sykliksi”* (Wong 2020.).

3.2 2D-Pala-, nukke- ja luurankoanimaatio

2D-pala-animaatiossa (eng. cutout animation) hahmo pilkotaan erikseen liikuteltaviin osiin, jolloin hahmoa voi kuva kuvalta ohjata liikkumaan (Koulukino 2024.). Nukkeanimaatio (eng. puppet animation) ja pala-animaatio on *stop motion*-animaatiotekniikoiden nimityksiä, jossa usein juuri käsinkosketeltavan ja taivuteltavan nuken tai palojen eri asennoista otetaan kuva jokaisen pienen asennon muutoksen jälkeen. Termi ei täysin kuvaa 2D-pala- ja nukkeanimaatioiden tekniikoita niiden ollen digitaalista ja hyödyntäen 3D-mallinnuksen kaltaisia työkaluja. Nimitykset tarvitsisivat itsenäisemmät nimet, jotka erottavat ne paremmin stop motion-animaatiotekniikoista. Etenkin, kun pelimaailmassa on useita stop motion-tekniikoilla tehtyjä tuotoksia, kuten vuoden 1996 *The Neverhood* (Erhard 2022.).

Luurankoanimaatiossa (eng. skeletal animation) ja pala-animaatiossa hahmo pilkotaan osiin, mutta eroava tekijä on, ettei pala-animaatio aina vaadi riggausta (Brown 2024.). Palasista koottua hahmoa voi myös animoida *sprite swap*paamalla, jolloin vaikka neutraalin suun sprite tai mesh voidaan vaihtaa toiseen, vaikka dynaamisempaan irvistykseen, ilman suurempia animointeja (Varrio 2024,

29-30.). Digitaalisesti animaatioihin voidaan hyödyntää *tweening*iä, joka tulee aiemmin mainitusta sanasta inbetween, eli “välissä”, välianimaatioita. Yleensä väliruudut piirretään käsin, mutta ohjelmisto animoi automaattisesti määriteltyjen avainkuvien välille puuttuvan liikkeen interpolaatiolla.

Sprite-animaatioissa ei voi kontrolloida animaatioita muokkaamatta suoraan niiden pohjataidetta, kun taas pala- ja luurankoanimaatioissa valmis rigi on helposti muokattavissa. Nämä tekniikat ovat usein tiedostokooltaan kevyempiä, ja animaatioiden muokkaaminen, esimerkiksi nopeuden säätäminen, säilyttää animaatioiden sulavuuden (Vitaly 2023.).

“Joints can be constraints" - (Plummer 2021.)

Tyriq Plummer korostaa, ettei automaattiseen tweeningiin kannata luottaa liikaa. Ruutujen lisääminen ei välttämättä tarkoita parempaa animaatiota, ja animaation perusteet pätevät edelleen. Huonoja avainruutuja ei pelasta hyvätkään väliruudut, eivätkä hyvät avainruudut takaa, että väliruudut olisivat toimivia. Hyvin asetetussa rigissä on tärkeää hienosäätää animaatioiden ajoituksia. Nivelet voivat myös rajoittaa animaatioiden toimintoja, ja mitä monimutkaisemmaksi animaatiot menevät, sitä enemmän työtä ne vaativat (Plummer 2021.). Palaan tarkemmin tutkielmassani erilaisiin pala-, nukke- ja luurankoanimaatioita käyttäviin peleihin.

3.3 Parallaksivieritys, Parallax scrolling

Parallax scrolling, parallaksi tai päällekkäinen rullaus, on 2D-grafiikoissa hyödynnetty tekniikka. Etenkin sivuttaisessa perspektiivissä rikkaampi visuaalinen kokemus saavutetaan luomalla syvyyden illuusio eri nopeuksilla kulkevilla tasoilla. Tasot liikkuvat riippuen niiden oletetusta etäisyydestä kameraan; esimerkiksi ruohikko etualalla, jonka takana vuoristo ja taivas. Nurmikon ollen lähellä, liikkuu se nopeammin x-akselilla, vuoristo hitaammin ja taivas voi olla lähes paikoillaan (Timonen 2022, 30-31.). Peligrafiikat oppivat paljon muusta mediasta, juontaen tämäkin juurensa perinteisistä animaatioista opituista tekniikoista. Aikaisimmista kuuluisista sovelluksista lienee Disneyn animaatioklassikko, *Lumikki ja Seitsemän Kääpiötä* (1937), jossa monitasokameralla ympäristö liikkuu tapahtumien

mukana uskottavammin, kun eri tasoilla olevia kuvia liikutellaan eri tahdissa panoraamaisesti (Sasaguay 2023.).

3.4 Proseduraalinen animaatio

Proseduraalisella animaatiolla tarkoitetaan tekniikkaa, jolla reaaliajassa generoidaan ohjelmallisesti responsiivisia ja dynaamisia animaatioita matemaattisten algoritmien avulla, noudattaen määritettyjä sääntöjä. Yksinkertaisemmin ilmaistuna, ohjelmistosta ohjataan animaatioita. Pelimaailmasta voi saada täten elävämmän ja ennalta arvaamattomamman, lisäten uudelleenpeluuarvoa, koska pelaajat voivat kohdata ainutlaatuisia ja odottamattomia pelikokemuksia. Proseduraalisuutta voidaan hyödyntää myös ympäristöjen luomisessa. Proseduraalinen työtapa on toimiessaan nopeaa ja tehokasta, vähentäen pelinkehityksen kustannuksia. Animaatioiden osalta voidaan luoda uniikkeja liikkeitä, jotka mukautuvat eri tilanteisiin, mikä lisää pelin visuaalista ilmettä ja immersiota sekä, vähentää yksittäisten animaatioiden luomiseen kuluvaa työmäärää. Tämä voi myös auttaa vähentämään animaatiodatan muistinkulutusta. Proseduraalisen animaation heikkoutena on sen monimutkaisuus; se vaatii edistynyttä ohjelmointia ja fysiikkasimulaatiota, mikä tekee toteutuksesta haastavaa ja aikaa vievää. Ohjelmoijan ja animaattorin on tehtävä tiivistä yhteistyötä ja prosessi vaatii paljon testaamista sekä hienosäätöä. Vaikka animaatiodatan muistinkulutus vähenee, laskentatehon tarve kasvaa, mikä voi rasittaa laitteistoa ja johtaa suorituskykyongelmiin. Kaikkiin projekteihin sitä ei siis voi suositella, etenkään pienellä tiimillä, eli pelin laajuus on hyvä pitää mielessä (Lark Editoral Team 2024b.)

Proseduraaliseen animaatioon liittyvät myös termit *Inverse Kinematics* (IK), eli käänteinen kinematiikka ja *Forward Kinematics* (FK), eli eteenpäin kinematiikka. IK mahdollistaa hahmon nivelten asettamisen tiettyyn asemaan, esimerkiksi jalan asennon sopeutumisen ylämäkeen. IK:ssa *end effector* on kohde, johon hahmon osan tahdotaan päätyvän, ja *root joint* on alkupiste, josta muu IK-ketju alkaa (Ruuskanen 2018, 12.). FK on sitten taas toisinpäin, eli jalka esimerkkiin palaten, jalkaa ohjataan lonkan pallonivelestä ja jalan muut osat muokkautuvat tämän mukaan. FK:ssa tarkkojen pääte osien hallinta täytyy tehdä manuaalisemmin, jos polvi tai jalapohja halutaan tiettyyn asentoon (Varrio 2024, 27.).

Todellisen maailman esimerkkinä *Rain World* (2017) jossa maailma on täynnä proseduraalisuutta ja tilanteisiin mukautuvaa tekoälyä. Joar Jakobsson lähestyi peliä ekosysteemin näkökulmasta; maailma ja sen otukset elävät dynaamisesti. (Kuva 4) Kuvassa pelaajahahmo koittaa vältellä vihollislintua.

KUVA 4. Steam mainoskuva Rain Worldista (Akupara Games 2017.).

"Art is the goal and programming is the means" - (Jakobsson 2017.)

Otukset toimivat itsenäisesti ja reagoivat ympäristöönsä. Pelaajaohjattava Slugcat-hahmo on ruokapyramidin pohjalla. Muut otukset toimivat tekoälyn määrittämällä tavalla, havaitsemislogiikan ja suhteiden kehittyessä maailmassa (ThatGuyGlen 2023.).

4 LAITETAAN SE PERSPEKTIIVIIN

Videopeleissä yleisimmät kameran perspektiivit ovat *first-person*, eli ensimmäisen persoonan, *second-person* eli toisen persoonan ja *third-person* eli kolmannen persoonan. Nimikin antaa jo osviitan kuvakulmasta; ensimmäisessä "minä liikun", toisessa "sinä liikut" ja kolmannessa persoonassa "hän liikkuu".

Ensimmäisen persoonan kamerassa pelaajan kuvakulma on pelihahmon silmistä kuvattu, pelaaja siis ottaa pelattavan hahmon roolin. Monesti pelattavasta hahmosta näkee vain raajat, kuten kädet tai aseen edessä, joskus ei sitäkään. Kuvakulma on suosittu monissa ammuntapeleissä, joita kutsutaankin genressään *first-person shootereiksi.* Ensimmäisen persoonan ammuskeluista lukeutuu aiemmin käsitelty pseudo-3D *DOOM*. Kuvakulman vahvuutena on immersio, mutta kenttää suunnitellessa on otettava huomioon pelaajan rajattu näkökenttä ja etäisyys. Jossain peleissä pelaaja itse voi säätää näköetäisyyttään (eng. field-of-view, lyhenne FOV) parametreilla. Etenkin virtuaalitodellisuuden peleissä, matala näköetäisyys voi aiheuttaa pahoinvointia (Techtarget 2022.).

Toisen persoonan kamerakulma on harvinainen videopeleissä sen vaikean toteutuksen takia. Kamera toimii eräänlaisena hahmona, joka seuraa pelattavan hahmon liikkeitä. Tunnetuin esimerkki tästä on *Super Mario 64* (1996), jossa (kuva 5) Lakitu kuvaa Mariota pilven päältä. Pelissä on kaksi pääasiallista kameratyyppiä: pelihahmoa seuraava kamera, joka ottaa kentän muodot huomioon tekoälyn avulla ja vuorovaikuttava kamera, jota pelaaja voi itse ohjata (Wikipedia 2024c.). Toisen persoonan kamera voi kuitenkin toimia epätoivotulla tavalla, jääden helposti jumiin perspektiiviin, josta pelaaja ei näe mihin pitäisi liikkua (Medium 2019.).

KUVA 5. Lakitu Koopa tervehtii Mariota (Grubdog 2021.).

Kolmannen persoonan kuvakulmassa kamera leijuu yleensä pelaajan takana, vaikka olkapään yllä, *over the shoulder*, nähden koko hahmon, ympäristöä ja seuraten pelihahmon liikkumista. Kuvakulma tarjoaa kattavan näkemyksen, joka auttaa hahmottamaan ympäristöä ja hahmoa (Stegner 2020.). Peli *Chantelise - A Tale of Two Sisters* (2006) tarjoaa kolmannen persoonan kolmiulotteisen maailman 2D-sprite hahmoilla (Wikipedia 2024d.). Näennäisesti pelin animaatioita ja kameraa tarkastellessa (kuva 6), se tuntuu olevan lukittu ja hahmon spriteistä on kahdeksasta kuvakulmasta omat spritet, jotta hahmo ei kameran kääntyessä pyörisi paikoillaan.

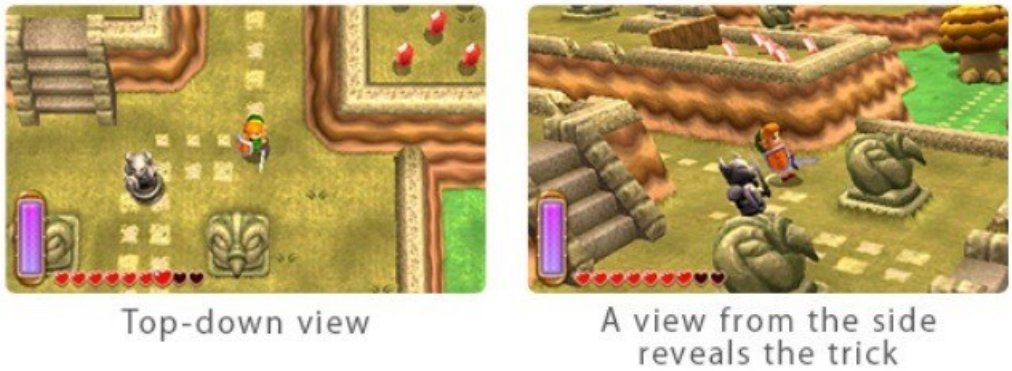
KUVA 6. Kuvassa pelissä taistelusta (Carpe Fulgur LLC 2011.).

Kolmannen persoonan perspektiivin alle laitetaan monesti myös ylhäältä alas (eng. top-down), sivusta (eng. side-on) tai isometriset (eng. Isometric) pelit, mutta koska 2.5D-grafiikoissa yleisemmin käytetään juuri näitä perspektiivejä, täytyy tutkielmassa syventyä niihin tarkemmin seuraavissa alakappaleissa.

4.1 Ylhäältä-alas, Top-down

Top-down kuvakulma, tunnettu myös nimillä lintuperspektiivi, ylimaailman-, jumal-, ylipään- tai helikopteri näkymä. Tarkoittaa peleissä kuvakulmaa, jossa pelaaja katsoo pelimaailmaa suoraan ylhäältä alaspäin. Tämä perspektiivi antaa pelaajalle hyvän yleiskuvan ympäristöstä ja pelihahmon sijainnista (Wikipedia 2024e.). Kuvakulmaan liittyy voimakkaasti myös nimitys ¾ -kuvakulmasta, jossa esitetään kolmiulotteista ympäristöä kaksiulotteisella alustalla. Kuvakulma oli suosittu etenkin 16-bittisellä aikakaudella, kaikki Y-akselilla syvyyttä sisältävät assetit on ikään kuin kuvattu perspektiivistä, vaikka kuvakulma on kaksiulotteinen (Pietiläinen 2020, 17; Unity Documentation 2024.). Malliesimerkiksi nousee *The*

Legend of Zelda: A Link Between Worlds (2013), jossa käytettiin 3D –assetteja ja top-down kuvakulmaa. Assetit on laitettu kulmaan, jotta ne näkyisivät “oikein” pelaajalle. (Kuva 7) Vasemmalla näkyy pelaajan näkymä, mutta sivulta katsottuna, assetit ovatkin kulmassa.



KUVA 7. Peliassetit ovat vinossa, jotta kamera kaappaa halutun kuvakulman (Brian 2013.).

4.2 Sivuttainen, Side-on

Sivuperspektiivissä pelaaja liikkuu x- ja y-akseleilla kaksiulotteisessa maailmassa. Yleisesti tästä kuvakulmasta tehdään pelejä, joita kutsutaan *Side-scroller*eiksi, eli sivukelauksiksi. Sivukelaukset ovat olleet suosittuja etenkin 2D- ja 2.5D-videopeleissä niiden synnystä asti; *Super Mario Brosista* nykypäiväisempään *Hollow Knight*iin. Aiemmin animaatio -kappaleen alaluvussa käsiteltiin parallaksiavieritystä, jota yleisesti hyödynnetään side-scrollereissa luomaan syvyyden ja vauhdin tunnetta. Kuten top-down kuvakulma, se tarjoaa kattavan näkemyksen kentästä ja pelaajan sijainnista sen sisällä (Wikipedia 2024f.).

4.3 Isometrinen, Isometric

Isometrisessä perspektiivissä pelialuetta kuvataan usein *aksonometrisesti* etäältä 30- tai 45-asteen kulmassa, jolloin suuri osa kentästä on näkyvissä pelaajalle. Pelimaailma voidaan projektoida *ortografisesti*, ilman z-akselin luomaa syvyyttä, tai *perspektiivisellä* projektiolla, joka vastaa todellisen maailman kolmiulotteisuutta. Isometrinen kuvakulma luo tilaa koskevan illuusion, jossa 2D

vaikuttaa kolmiulotteiselta. Tyypillisin isometrinen projektio on 30-astetta horisontaalisesti ja vertikaalisesti, niin että x-, y- ja z-akselit ovat kaikki 120-asteen kulmassa. Myös *dimetrinen* ja *trimetrinen* projektio ovat yleisiä. Dimetrisessä akselit vaihtelevat 116,57 ja 126,87 asteen välillä, luoden 2:1 tiilityksen. Trimetrisessä projektiossa kaikki akselit ovat eri asteissa (Pietiläinen 2020, 18; Pikuma 2022.).

Isometristä kameraa käyttävät pääasiassa *strategia*- ja *simulaatio*pelit, mutta myös muut genret hyödyntävät tätä perspektiiviä. Esimerkkinä voidaan mainita Supergiant Games, indie-studio, jonka perustivat Amir Rao ja Gavin Simon vuonna 2009 (Gameinformer 2021.). Supergiant Games käyttää omaa pelimoottoriaan kaikkiin peleihinsä, kuten vuonna 2020 julkaistussa Hadeksessa. *Hades* on *action roguelike*, jossa pelaaja ohjaa Zagreusta, Hadeksen poikaa, joka yrittää paeta manalasta. Pelin grafiikat ovat tyylitellyt paksuilla tummilla ääriviivoilla, ja kamera kuvaa peliä isometrisestä näkökulmasta. Jen Zeen sarjakuvamaisten hahmopotrettien pohjalta Paige Carter on 3D-mallintanut hahmot, jotka on *pre-renderöity*, eli ennalta renderöity kuviksi ja muutettu spriteiksi (Supergiant Games 2020.).

5 LIVE2D TEKNOLOGIA JA OHJELMISTO

Luvussa syvennytään itse live2D-animaation määrittämiseen; miksi tekniikkaa voi nimittää 2.5D-animoinniksi. Live2D Cubism ohjelmistoa kuvaillaan ja tiedostomuotoja avataan. Alakappaleissa esille nostetaan VTuberit, ohjelmiston suosiossa olevat suoratoisto hahmot. Lisättyä todellisuutta käsitellään nopeasti mahdollisena käyttömuotona ja lopuksi ohjelmiston työkalut käydään läpi.

Live2D Cubism on ohjelmisto, jolla voidaan luoda eläväisiä 2D-hahmojen animaatioita ilman tarvetta monimutkaiselle 3D-mallinnukselle, tai perinteiselle ruutu kerrallaan animaatiolle. Toisin kuin 3D-animaatiossa, jossa yleisesti mallinnetaan piirroksen pohjalta, Live2D käyttää suoraan alkuperäistä 2D-kuvaa, menetelmän säilyttäen piirroksen ominaisen tyylin (Kataja 2022, 7.). Tämä tekee Live2D:sta käytännöllisen ratkaisun projekteihin, joissa halutaan säilyttää käsin piirretyn kuvan ominaispiirteet.

Termillä live2D viitataan useasti Live2D osakeyhtiön ohjelmistoon Live2D Cubismiin, jonka kehitti Tetsuya Nakashiro. Alkuun Nakashiro kehitti itsenäisesti ohjelmistoa perustaen sen ympärille Cyber Noise yhtiön vuonna 2006, muuttaen suosiostaan ja kasvustaan vuonna 2014 nimensä nykyiseksi (Wikipedia 2024g.).

JSON eli *JavaScript Object Notation*, on tiedostomuoto, joka on suhteellisen kevyt ja JSON –muotoa käyttää myös Live2D Cubismin mallit. JSON–tiedosto tallentaa tasojen tiedot ja paljon muutakin dataa ja se on yhteensopiva monien ohjelmien kanssa (Hietamies 2024, 17.). Helppolukuinen tiedostomuoto on myös JSON–tiedoston vahvuuksia.

Live2D-animaatiota kutsuukin viralliset Wikipedia sivut omaksi tekniikakseen, mutta lähde tähän väitteeseen vie yksittäishenkilön väittämään (ShiraLive2D 2024.), joten nimitystä ei voi sanoa olevan ammatillisesti käytössä. Samantyyppisten ohjelmistojen nimeäminen tekniikalle on esimerkiksi reaaliaikainen 2D-nukkeanimaatio Inochi2D:ltä (Inochi2D 2024.) ja DragonBones nimittää sen 2D–luurankoanimaatioksi (DragonBones 2017.). 2D-Pala-animaatiossa hahmo leikataan osiin, mutta live2D periaatteessa sisältää luurankoakin, joten yksiselitteisesti

ei voi sanoa onko live2D enemmän 2D-pala-, nukke- vai luurankoanimaatiota. Tutkielmassa käytetään selvyyden ja lyhyemmän kirjoitusasun vuoksi tekniikasta nimitystä live2D. Tutkielmassa muutenkin lähestytään 2D luuranko-, pala- tai nukkeanimaatoita lähinnä juuri Live2D Cubism-ohjelmiston näkökulmasta.

5.1 VTubereista novelleihin ja lisättyyn todellisuuteen

Live2D on ollut laajassa käytössä *VTuber*ien ansiosta. VTuberit, eli virtuaaliset tubettajat, jotka yleisemmin käyttävät anime avatareja, reaaliaikaisella liikkeen kaappauksella ovatkin valloittaneet sosiaalisen median, ohittaen katsojakunnassa monia "oikeita ihmisiä" (Gayton 2023.). Kuvassa (kuva 8) yksi tuotettu VTuber malli.

KUVA 8. Kettutytölle tuotettiin kasa eri ilmeitä ja eleitä.

Visuaaliset novellit ovat hyödyntäneet pelikehityksessä ehkä eniten Live2D-animaatioita (Wikipedia 2024a.). Live2D-tekniikalla tuotettiin OAMK:in Indesign kurssilla E. Ojalan kirjoittama post-apokalyptinen *Nanostrofi*. Tekniikka oli silloin vielä uusi ja vieras, mutta novelli animoitui eläväksi, lukijan silmille hyppääväksi kokemukseksi. Kuvassa (kuva 9) hahmon käsi irtoaa ja lopputuloksesta tulikin odotettua brutaalimpi.

KUVA 9. Pojan käsi animoitiin irtoamaan ja verta roiskumaan.

Opintojeni aikana yksittäisellä kurssilla käytettiin Arilyn-sovellusta, jolla kehitettiin AR-projekti. *Augmented Reality* eli lisätty todellisuus, jolla älypuhelimen kameran kautta voi vuorovaikuttaa sovelluksella todellisen maailman ympäristöjen kanssa (Marr 2021.), kuuluisin esimerkki ollen *Pokémon Go* (2016). Projektissa tuotettiin Kiivityttö, joka hyppää käyttäjän puhelinruudulle kertomaan kiiveihin liittyviä tietoiskuja avainkuvan havaitessaan, sen ollen kiivi. Animaatioita ei tuotu projektiin JSON-tiedostoina, vaan ne muutettiin videoformaattiin. AR-teknologialla voisi tuottaa eläviä sovelluksia live2D:een avulla. Valitettavasti Arilyn meni konkurssiin, eikä esimerkkiä voi tarjota Arilyn-sovelluksesta. Kuvassa (kuva 10) Kiivitytön animaatioista, luonnoksista ja avainkuvasta.

KUVA 10. Luonnoksia, lopullinen kuva. Kiivityttö vilkutti käyttäjälle ja puhekuplissa vaihtui eri kiivi tietoja.

5.2 Työkalut ja ominaisuudet

Live2D Cubism tarjoaa monia työkaluja, jotka helpottavat animaation luomista ja liikkeiden määrittelyä. Alakappaleessa käsitellään ne pintapuolisesti, hahmon luomisessa niitä tarkennetaan.

Tasojen hierarkiat tulevat ohjelmistoon siinä järjestyksessä, missä tasot on piirto-ohjelmistossa määritelty. Etäisyyksiä voidaan vielä muokata *Draw Order*illa, eli lukuarvoilla. Arvot ovat alueilla 0-1000, pienimmän arvon ollen taaimmaisin.

Samaa lukuarvoa jakavat tasot näkyvät alkuperäisen piirto-ohjelmiston määritelmien mukaan. Lukuarvoa voi myös määritellä parametreille, eli taaimmaisia osia voi tuoda animaatiossa eteen ja toisinpäin (Kataja 2022, 16). Tasoja voi piilottaa ja tuoda näkyville, mutta ne voidaan myös leikkaustunnisteella *Clipping ID*:eellä sijoittaa näkyvyydeltään toisen tason “sisälle”. Jokaiselle tasolle tehdään mesh, eli verkko, joka sisältää kasan yhteen punottuja pisteitä, verteksejä. Kalaverkkoa muistuttava peitto tehdään tason päälle (Cutie Dragon 2022.). Eroten 3D:een polygoineista, puuttuu näistä verteksiverkoista, joita ohjelmistossa *ArtMesh*eiksi nimitetään, syvyys, jolloin voi keskittyä vain kaksiulotteisten muunnosten tekemiseen, eli 2.5D-animointiin (Live2D 2024a.).

Deformer

Deformerit, tai joiden epävirallinen käännös on muodonmuuttajat, määrittävät kuinka hierarkisesti ArtMeshien verteksien muodot ja liikkeet muuttuvat:

- **Warp Deformer:** Vääntymämuuntaja, luo ArtMesheille ruudukon, jossa on kiintopisteitä. Kiintopisteiden liikuttaminen vaikuttaa valitun tai valittujen ArtMeshien muotoon.
- **Rotation Deformer**: Kiertomuuntaja, ArtMesh voi kiertyä ja liikuttaa alempia Rotation Deformereita hierarkisessa järjestyksessä, vääristämättä verteksien muotoja. Aiemmin tutkielmassa käsiteltiin IK, periaate on sama. (Live2D 2023a.)
- **Deform Path**: Vääristävä polku, työkalulla voidaan määrittää liikettä seuraavat polut, jotka vaikuttavat orgaanisten osien, kuten hiusten, liikkeisiin. (Live2D 2022a.)

Parametrit

Parametrit määrittävät hahmon liikkeet ja animaatiot. Avainruutujen avulla luodaan sujuvia liike-eroja ja automatisoituja animaatioita (Live2D 2024b.). Näihin kuuluu myös mahdollisuus linkittää eri parametrejä toisiinsa ja käyttää työkaluja kuten:

- **Synthesize Corners**: Syntetisoi kulmat, luo suuremmaksi linkitettyjen parametrien kulmien liikkeet määritettyjen parametrien avainruutujen pohjalta.
- **Reflect Motion**: Peilaa parametrin liikkeet, esimerkiksi oikealta vasemmalle, mikä säästää aikaa symmetrisissä liikkeissä. (Cutie Dragon 2021.)
- **Glue**: Liimaa eri ArtMeshien verteksit yhteen, mikä estää liikkumattomien osien virheitä ja varmistaa, että liikkeet ovat sujuvia ja luonnollisia. (Live2D 2023b.)

Fysiikka-asetukset Live2D:ssä auttavat luomaan realistisia liikkeitä, kuten painovoiman tai elastisuuden vaikutuksia. Fysiikoiden asetuksilla voidaan kontrolloida, kuinka esineet tai osat liikkuvat toistensa suhteessa reaaliajassa (Live2D 2024c.). *Expressions*, eli ilmeet, jolla luodaan erillisiä ilmeitä hahmolle parametreilla, palan vaihdolla tai muuttamalla jotain liikkeitä. Nämä parametrit on hyvä tehdä nappuloina, jolloin parametrin alue on 0 ja 1 välillä. Exp-tiedostot saa Live2D viewerissä luotua, kun hahmo on valmis (Omori 2021.).

6 MALLIN LUOMISPROSESSI

Mallin luomisprosessi Live2D:ssä aloitetaan usein hahmon piirtämisellä, mutta samoja menetelmiä voidaan käyttää myös kenttä assettien luomiseen. Myöhemmin tutkielmassa otetaan esille peli *Gnome Attack*, jossa moni vuorovaikuttava kenttäassetti tehtiin myös live2D-mallina. Prosessi jakautuu neljään päävaiheeseen: piirtäminen, pilkkominen, riggaus ja animointi, joista suurin työ kuluu riggaukseen ja liikkumisparametrien määrittelyyn.

Piirtäminen ja pilkkominen

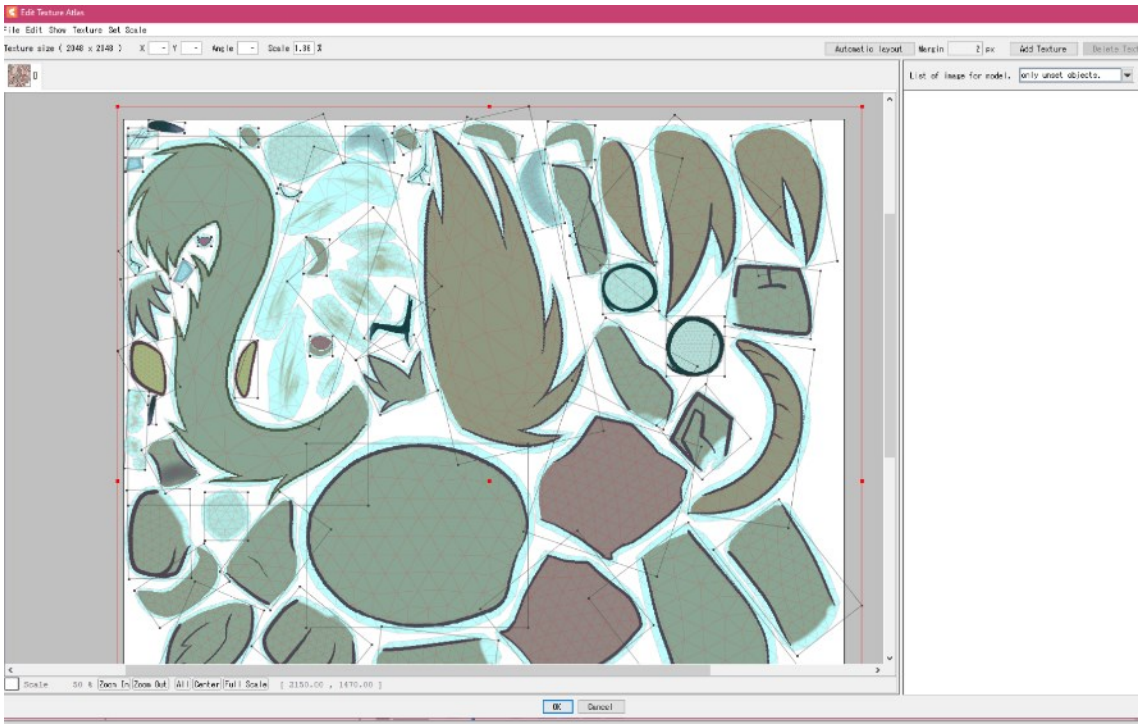
Live2D-malli ensin kuvitetaan jossakin tasoja sisältävässä ja Photoshop -tiedostoja tukevassa piirto-ohjelmassa, käytössä Clip Studio Paint (kuva 11). Tutkielmaa varten tehtiin *Cat game*-projekti, jolla havainnollistetaan Cat-hahmon luomisen vaiheet.

KUVA 11. Kuvakaappaus Clip Studio Paintista, jossa hahmon osat ovat pilkottuna eri tasoille.

Liikkuvat palaset, kuten pää, hiukset ja silmät, erotetaan erillisiksi *layer*eiksi, eli tasoiksi. Tasoja paloitellaan osiin täysinä kokonaisuuksina, vaikka pään päälle tulee hiukset, perinteisesti piirtäen päätä ei tarvitse peittyvän osan taakse piirtää. Malliin täytyy siis täyttää tyhjät tilat, jotka voivat paljastua animoitaessa. Taso paloille voi vielä tehdä *Multiply*- ja *Additive*-tasot, mikäli haluaa erillisiä varjoja tai heijastuksia. Läpinäkyvyys, eli *Opacity*-arvot säilyvät myös (Kataja 2022, 9-12), mutta niitä voi vielä säätää tai jopa animoida ohjelmistossa. Yleisesti malli on luettavissa Photoshop-tiedostona (.psd) Live2D-ohjelmistossa, jonne se tuodaan *import*illa.

Tekstuurikartta

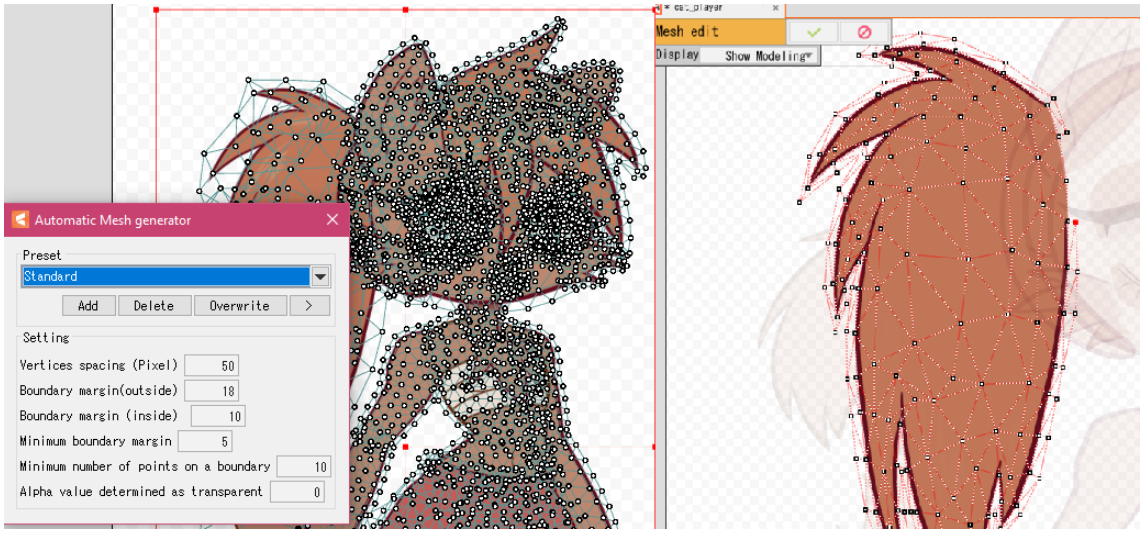
Hahmon ollessa ohjelmistossa, määritetään sen tekstuuri kartta (kuva 12). Ohjelmiston yläpalkista valitaan *Modeling* > *Texture* > *Edit Texture Atlas*, jossa tasot erotellaan, pakataan ja resoluutio määritellään. Tämä kannattaa tehdä myöhemmässä vaiheessa, etenkin jos hahmoon tulee vielä muutoksia.



KUVA 12. Tekstuuriatlasta kannattaa pakata mahdollisimman hyvin, jotta saa korkeampaa resoluutiota.

ArtMeshien luominen

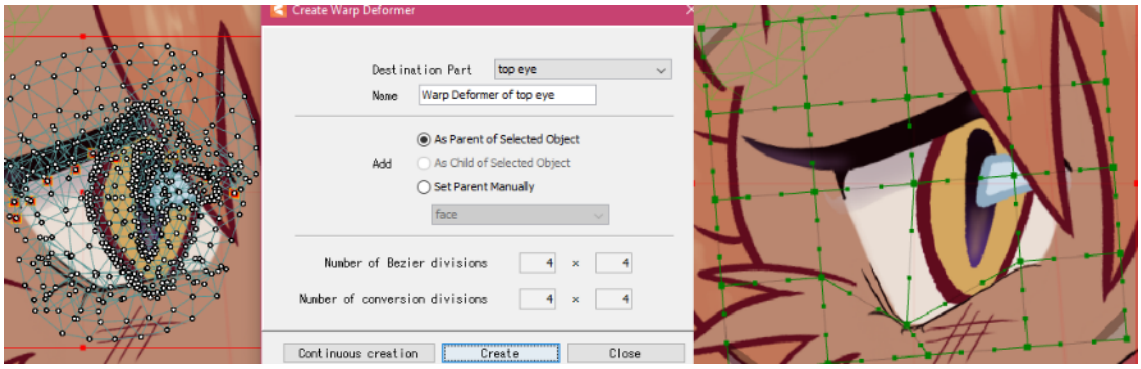
Hahmolle luodaan verkkorakenne; kullekkin tasolle tehdään 3D-mallinnuksesta tutut verteksiverkot. (kuva 13) Verkot voi editoida itse tai käyttää automaattista verkko generaattoria, A*utomatic Mesh generatoria*. Verkon ollen valmis, pystyy tason muotoa muokkaamaan vertekseistä vedellen. Kuvassa vasemmalla käytetään automaattista verkkojen luontia kaikille hahmon tasoille. Kiintopisteiden tarkkuuksia ja määriä voi virittää haluamansa mukaan. Kuvan oikealla puolella muokataan verkkoja manuaalisesti hiuksiin.



KUVA 13. Verteksien sotku vasemmalla, oikealla käsin siistittyä.

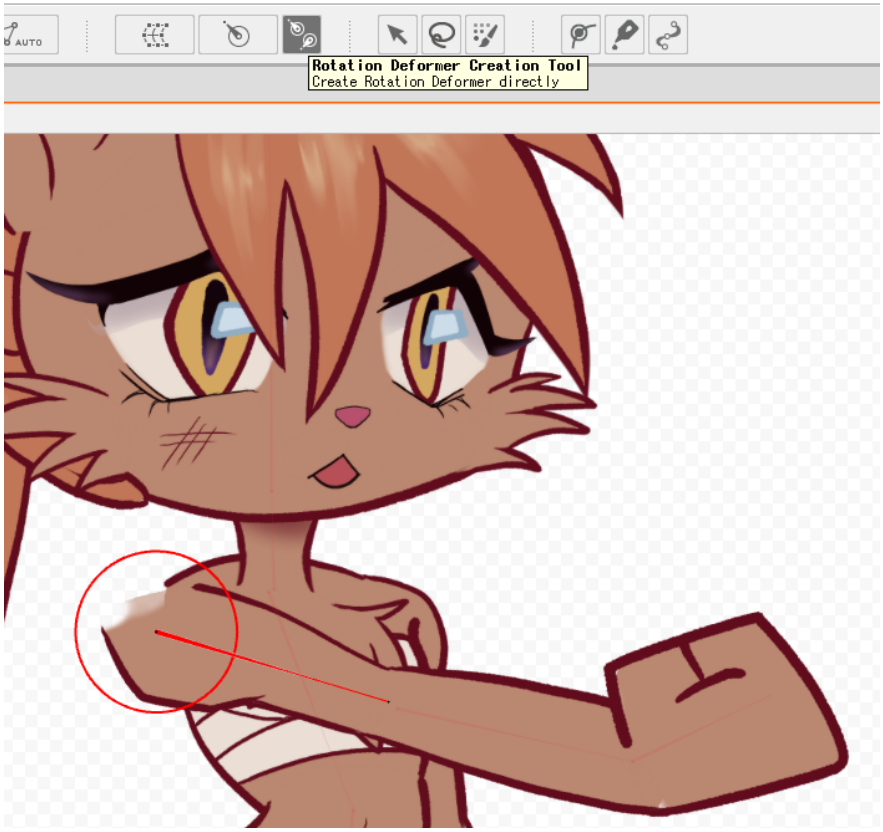
Deformerit

Tasojen liikkuvuutta voi yhdistää erilaisilla työkaluilla kuten ohjelmistossa deformereilla, muodonmuuttajilla, jolla mallin muotoja muunnellaan. Live2D Cubismissa ei tarvitse liikettä muokata suoraan vertekseistä vetämällä. Sen sijaan kannattaa hyödyntää deformereita, joihin voi liittää useita ArtMeshejä. Tämä tekee samanaikaisesta liikkeestä helpompaa ja nopeampaa toteuttaa. Warp Deformer luo ruudukon valitun tai valittujen palojen päälle (Kataja 2022, 32-33.). Kuvassa (kuva 14) silmälle luotiin Warp Deformer, jossa määritettiin 4x4 ja 4x4 määrä kiintopisteitä, joita vetelemällä läheiset verteksit valituilla ArtMesheillä liikkuvat kiintopisteen mukana.



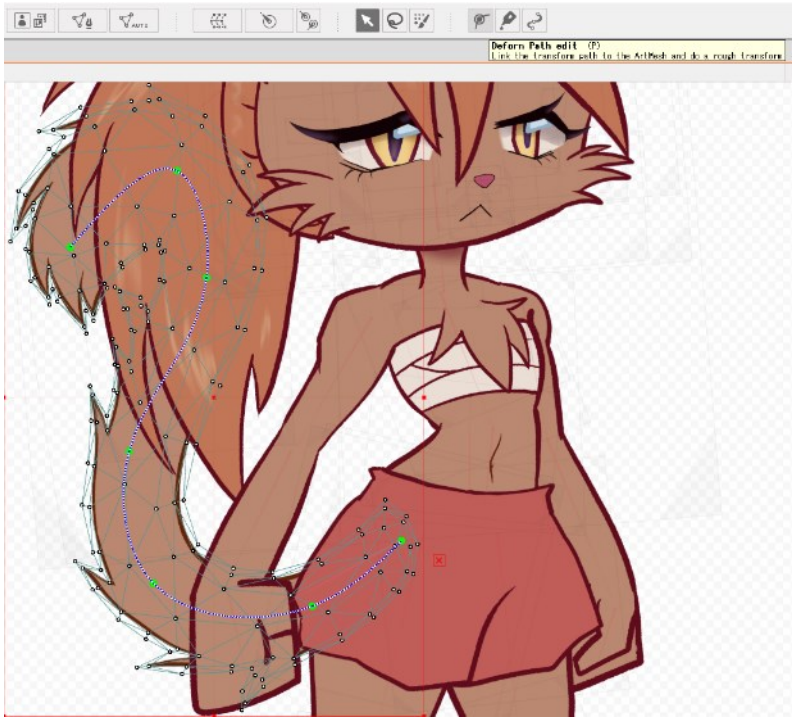
KUVA 14. Silmän päällä oleva suurehko Warp Deformer, tällä voi hallita pään kääntymistä, ja tehdä tarkemmat alenevat deformerit tarkemmille muunnoksille.

Rotation Deformer lisää valittujen ArtMeshien kiertoa määritetystä kiintopisteestä. Hierarkkisesti alenevia Rotation Deformereita voi lisätä alkuperäisen Rotation Deformerin liikkeeseen. Näin esimerkiksi käden eri osat olkapäästä sormiin liikkuvat yhtäaikaisesti, ilman että kyynärpää jäisi eri liikeradalle olkapäätä liikuttaessa. Hahmon käsi liikkuu kokonaisuudessaan olkapäästä liikuttaessa (Kuva 15) ja alenevassa hierarkiassa näkyy kyynärvarren ja kämmenen Rotation Deformerit himmeänä.



KUVA 15. Olkapää kääntyy, mutta paljastaa samalla hajoavan tekstuurin, nämä voi korjata myöhemmin samaistumaan muunnoksiin.

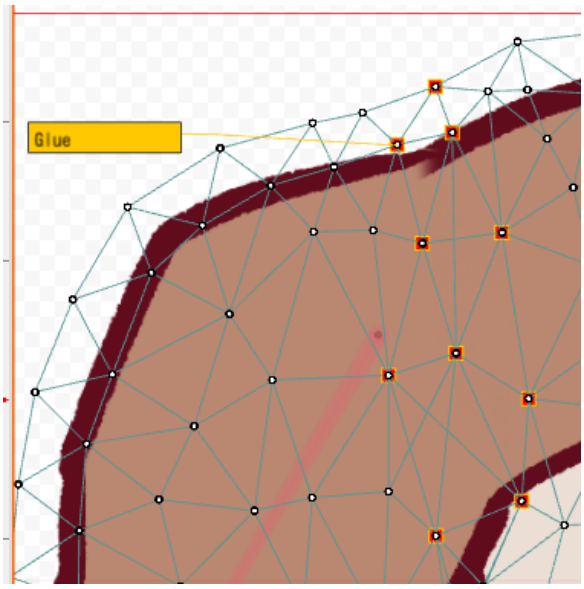
Deform Pathilla tasoon voi tehdä usean eri liikkuvan pisteen piirretyn suunnan mukaisesti. Vertekseihin vaikuttavaa muokkausvoimakkuutta ja vaikuttavaa aluetta voi säätää halutulle alueelle. Hahmon häntä (kuva 16) on orgaaninen, joten hiusten lisäksi siihenkin pystyi laittamaan Deform Pathin.



KUVA 16. Hännän tyveen voi laittaa vielä Rotation Deformerin, jotta häntää voi liikutella isolla alueella.

Liima

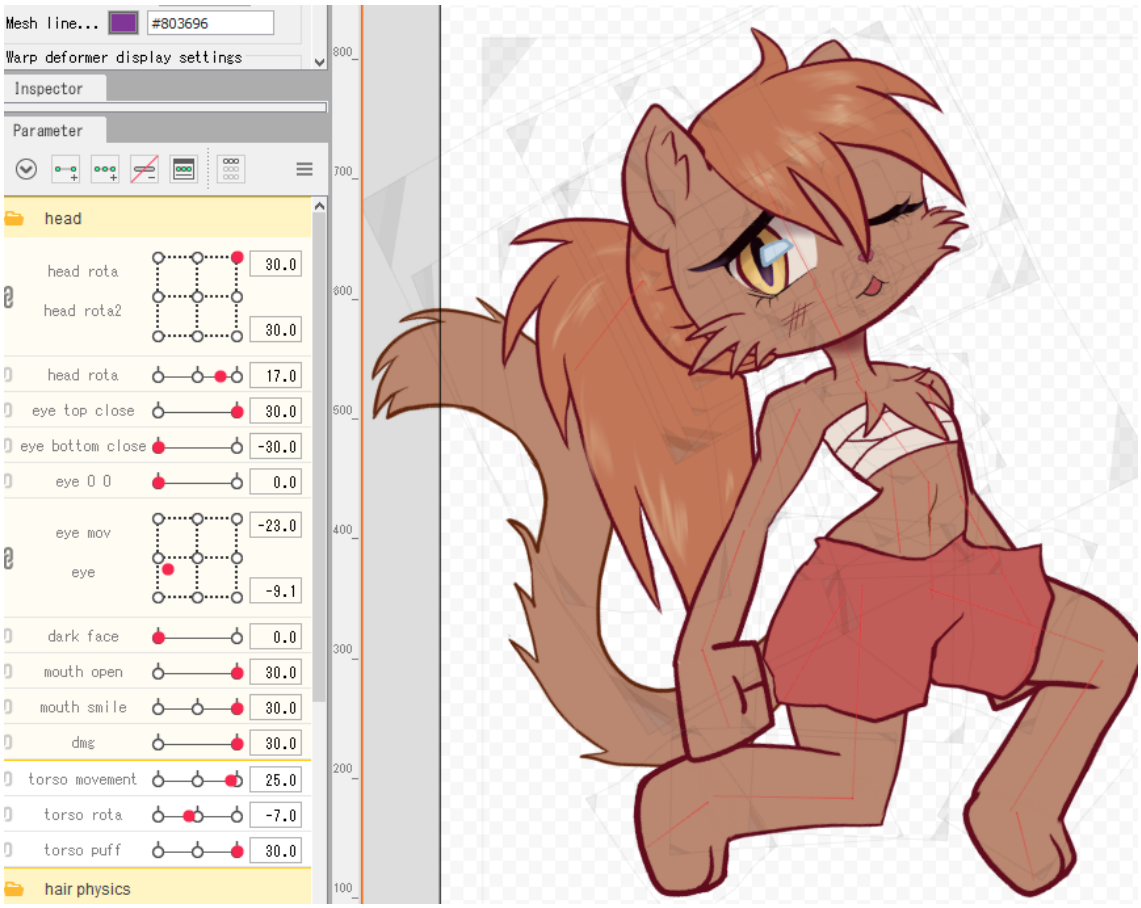
Glue, eli liima, on työkalu, jolla eri palojen verteksejä, vaikka hartia-alueen olkapään ja torson, voi liimata kiinni toisiinsa. Mahdolliset deformerit ottavat nämä liimakohdat huomioon, eikä kaikkea liikkuvuutta tarvitse käsin korjata sulavammaksi ArtMeshien välillä (Live2D 2023b.). Liimalla täytyy kuitenkin olla tarkkana, ettei tule epätoivottuja muunnoksia. Olkapää (kuva 17.) liimattiin torsoon.



KUVA 17. Verteksejä yhdistävä liima oli huonosti määritelty ja hahmo venyi ei-halutusti kättä liikkuttaessa.

Parametrit

Live2D:n tehokkuus syntyy sen kyvystä käyttää parametreja liikuttamaan osia hahmosta ja luomaan animaatioita. Nyt voi alkaa tallentamaan eri liikkeiden muutoksia muistiin parametrien avainruutuihin. Lähtökohtaisesti, yhdessä parametrissä on kolme avainruutupistettä, josta ohjelmisto interpoloi automaattisesti avainruutujen välit sulavaksi animaatioksi. Parametrejä, kuten myös avainruutuja, voi tehdä itse lisää omiin tarpeisiin. Parametriin voi tehdä nimen lisäksi oman ID:een, nimityksen, jota eri ohjelmistot, kuten myöhemmin käsiteltävät *Mocap*it, osaavat havaita, tai pelinkehityksessä ohjelmistollisesti kutsua. ID:llä voi myös merkata leikkaustunnuksen, Clipping ID:een, toiseen tasoon. Tällöin ArtMesh näkyy vain toisen ArtMeshin sisällä, esimerkiksi silmän iiriksen voi rajata näkymään vain valkuaisen sisälle. Kuvassa (kuva 18) hahmon useita parametreja on liikuteltu eri avainruutuihin. Parametreja määritellessä on hyvä tarkistaa miten hahmon eri parametrit toimivat yhdessä.

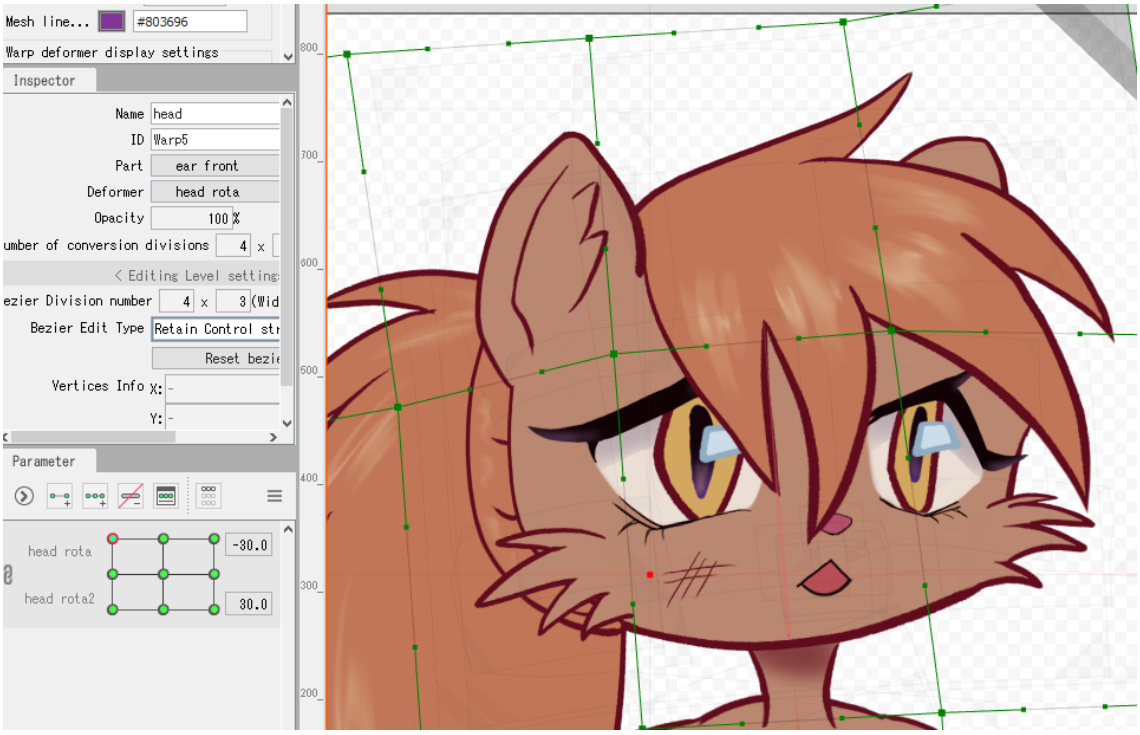


KUVA 18. Hahmo taipuu jo moneksi.

Parametrien ruutujen määrää voi itse määritellä, mutta aloitusparametrin arvoalue, “range”, ”–30” ja ”+30” riittää lähes kaikkeen, eikä se vaikuta liiaksi suorituskykyyn. Avainsääntönä, isoihin liikkeisiin ”–30” ja ”+30”, pienenpiin ”-10” ja ”+10”. Fysiikalle kohdat ”0” ja ”+1” ja vipumaisesti toimiville, päälle-pois parametreille ”0” ja ”+1” (Kataja 2022, 21). Parametrin voi merkitä toistuvaksi käyttämällä *repeat*-valintaa (Live2D 2022b.). Tämä helpottaa esimerkiksi hengitysanimaatiota, sillä keskikohdassa keuhkot olisivat täynnä ilmaa ja päädyissä tyhjillään. Parametria ei tarvitse viedä takaisin alkuun, mikä voisi näyttää kömpelöltä animaation pyöriessä takaperin.

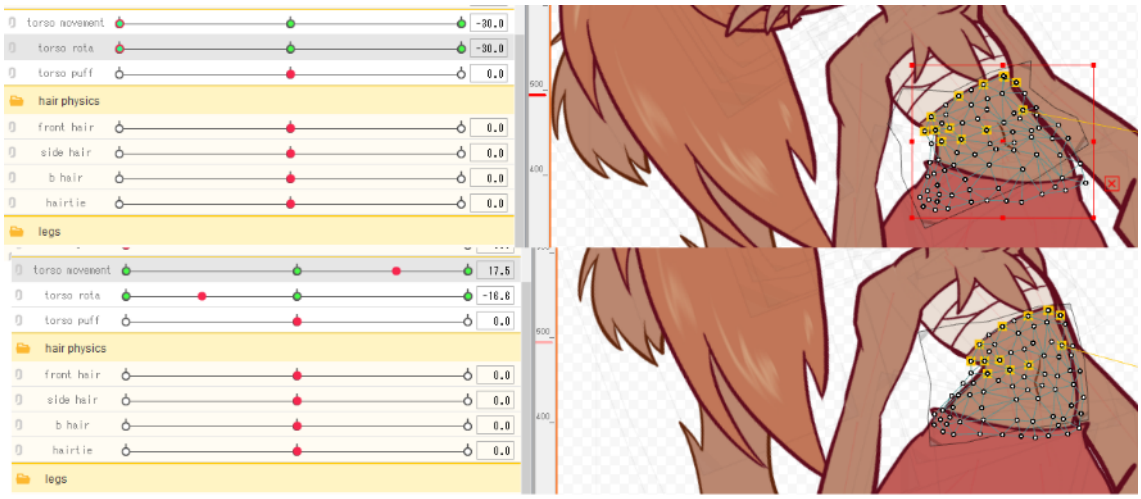
Jos kaksi hierarkkisesti päällekkäistä parametria yhdistää toisiinsa linkki -ikonilla parametrin nimen vasemmalta, saadaan kolmen pisteen sijaan yhdeksän pistettä. Yhden parametrisuoran sijaan parametria voi tällöin liikutella 3x3-alueella. Tämä helpottaa esimerkiksi pään orientaatioiden määrittelyä. Yhdeksän pisteen kanssa voi olla työlästä tehdä jokainen asento erikseen, joten kannattaa aloittaa neutraalista suoraan katsojaan suunnatusta keskiasennosta ja säätää ensin

pääasemat (ylös, alas, sivulle). Sivulle merkatusta avainruudusta voi kopioida vielä heijastamaan liikkeen muodon *Reflect motion*-työkalulla toiselle sivulle aikaa säästämään ja yhtenäistä animaatiota saavuttaakseen. Lopulta viistot voi toteuttaa syntetisoimalla kulmat *Synthesize corners* työkalulla, joka tuottaa aiemmin merkittyjen parametrien muotojen pohjalta viistojen muodot automaattisesti. Parametreihin voi vielä itse lisätä lisää avainruutupisteitä, mikäli sen kokee tarpeelliseksi. Kuvassa (Kuva 19) hahmon pään orientaatio on merkitty eri kuvakulmiin.



KUVA 19. Suunnittelin hahmon sivukelaus -peliin, hahmon ei tarvitse katsoa suoraan pelaajaan, joten hahmo on aina hieman viistossa katsoessaan sivullekkin.

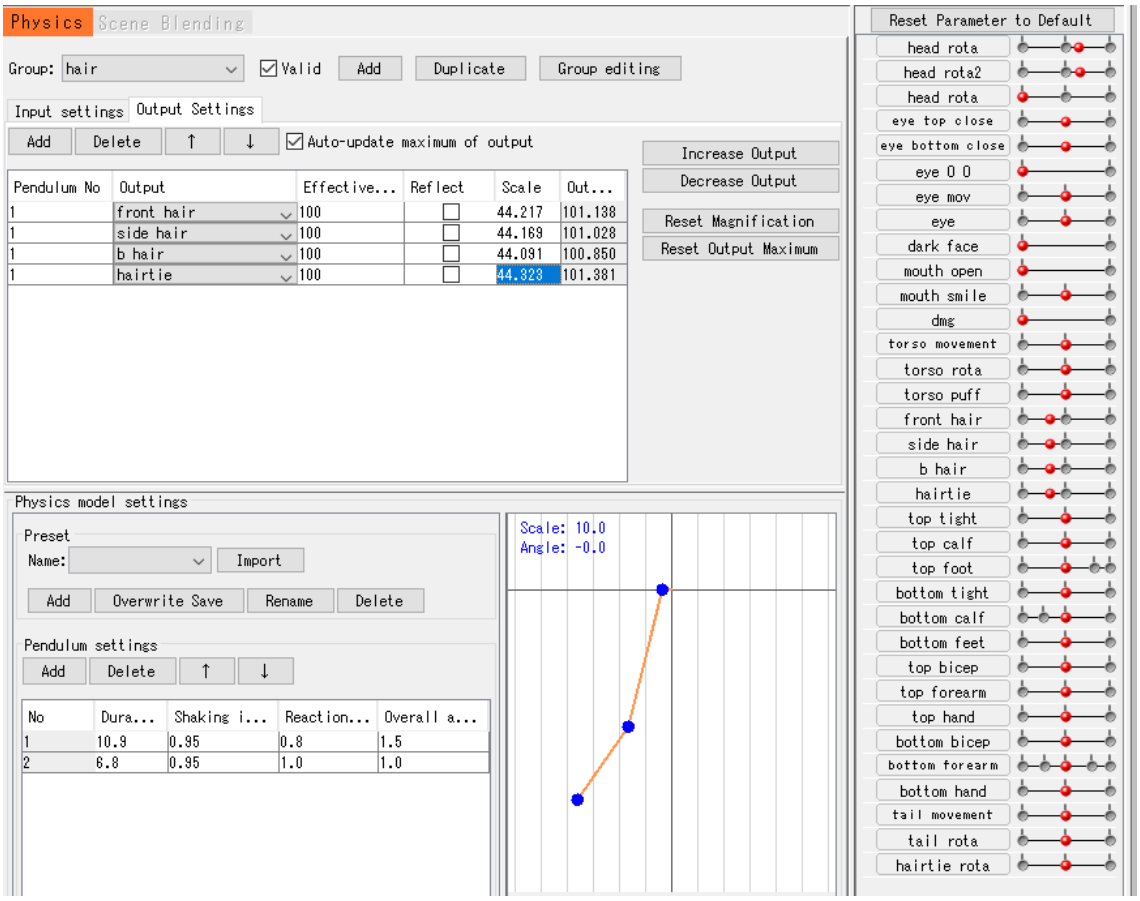
Päällekkäisten parametrien ja avainruutujen kanssa täytyy olla tarkkana. Hierarkia vaikuttaa näihin, ja joskus eri parametrien välille täytyy tehdä muokkauksia. Tämä voi luoda moniosaisia parametreja, joissa sama Deformer tai ArtMesh on käytössä eri parametreissa. Tässä on riskinä virheelliset asennot, joten kannattaa välttää liian monimutkaisia ratkaisuja. Vaihtoehtona on luoda lisää erillisiä deformereita, mutta liian monet deformerit voivat aiheuttaa epätoivottuja muunnoksia. Kuvassa (Kuva 20) hahmon vatsan verkko on suoraan merkattu kahdelle erilliselle parametrille suoraan ArtMeshistä deformerin sijaan, vihreät pallot ovat avainruudut.



KUVA 20. Käytä vasta viimeisenä oljenkortena ArtMeshin avainruutuja ja koita välttää useampia päällekkäisiä parametreja.

Fysiikat

Live2D Cubismissa on myös fysiikat, joille tehdään omat parametrit. Fysiikat voi määrittää menemällä ohjelmiston yläpalkkiin *Modeling* > *Open Physics/Scene Blending settings.* Valikossa voi valita tai lisätä uuden ryhmän, *Group*. Esimerkiksi hahmon pään parametri voidaan lisätä tuloasetuksiin, *input settings*, jolloin se vaikuttaa fysiikoihin. Lähtöasetuksiin, *output settings*, voidaan puolestaan lisätä fysiikoiksi tarkoitetut parametrit, kuten hiukset. Fysiikoissa voi määritellä kuinka moniosainen, pitkä, painava tai voimakas fysiikan vaikutus on (Kataja 2022, 22-23.). Kuvassa hiusten fysiikoita määritellään (kuva 21), tässä tapauksessa kaikki hiusten osat liikkuvat pään liikkuessa, mutta fysiikoita voi erotella useaksi eri osaksi tai jopa moniosaiseksi.



KUVA 21. Fysiikka-asetuksia on monia, mutta muutamaa valmisasetusta käyttämällä ne sujuvat helpommin. Pendulum, heiluri, simuloi fysiikan vaikutusta.

Animaatio

Kun hahmo on riggattu ja halutut deformerit määritelty, voi mallin viedä animoitavaksi. *Form animation*issa, muodosta animaatiossa, malli tuodaan *timeline*lle, aikalinjalle, josta päätetään animaation muoto, tässä tilanteessa sen ollen SDK (Unity). Animaatiota kontrolloidaan määritettyjen parametrien avulla, aikalinjan *Dopesheet*tiin, pinta-arkkiin, merkaten avainruutuja ja *Graph editor*issa, graafieditorissa, voi muokata liikkeiden sulavuutta tai suoraviivaisuutta (Kataja 2022, 24-25, 43.).

Export ja import

Lopuksi *File > Export For Runtime > Export as moc3 file*, halutessaan myös fysiikat ja samoin mahdolliset animaatiot voi viedä. Live2D Cubism assetin tuomiseksi eri pelimoottoreihin tarvitaan lisäosia, jotka saa ladattua Live2D Oy:n omilta sivuilta, Live2D Cubism SDK –osiosta. *Software Development Kit* eli SDK -lisäosan voi ladata ja asentaa halutulle pelimoottorille (Live2D 2024d.). Mahdollisesti

malleja, joita yleisemmin pelimoottoreissa kutsutaan asseteiksi, voi joutua vielä virittämään pelimoottorissa. Tekstuurikartat voivat tulla sumeina, jolloin kuvakarttojen resoluutio pitää varmistaa korkealaatuisemmaksi. Assetin piirtoetäisyys voi mennä päällekkäiseksi. Muitakin erikoisuuksia esiintyy, kuten moottorin kenttäeditoriin tuotuna, assetti ei välttämättä heti näy.

7 LIVE2D:N KÄYTTÖ PELINKEHITYKSESSÄ

Seuraavissa alakappaleissa päästäänkin suoraan itse asiaan, erilaisia käyttötapoja ja pelejä nostetaan esimerkeiksi, sekä niiden hyötyjä ja haittoja.

Ensimmäisenä alakappaleissa käsitellään animaatioiden monipuolisuutta ja mukautuvaisuutta, mukaan otetaan käytännön esimerkkejä projekteista. Syvennytään liikkeenkaappaukseen ja vertailussa Live2D Cubismia muihin live2D:een tyylisiin animaatio-ohjelmiin. Esittelyssä eri tekniikoilla tehtyjä tuotoksia ja peliteollisuuden tuotantoja. Tutkielman ollessa staattinen dokumentti, tutkielman lukemisen lisäksi suositellaan katsomaan ohjelmistolla tuotettuja animaatioita.

7.1 Monimuotoisuus animaatiossa

Nukke- ja luurankoanimaatioiden käyttäminen vähentää tiedostokokoa verrattuna perinteisiin sprite-pohjaisiin animaatioihin, joissa jokainen animaatiokehys on oma kuvansa. Staattisten spritejen ongelmana ovat paljon tilaa vievät spritesheetit jokaiseen erilliseen animaatioon. Live2D:n animaatiot luodaan liikuttelemalla ja manipuloimalla samaa grafiikkaa, joka säästää tallennustilaa (Berbece 2015.).

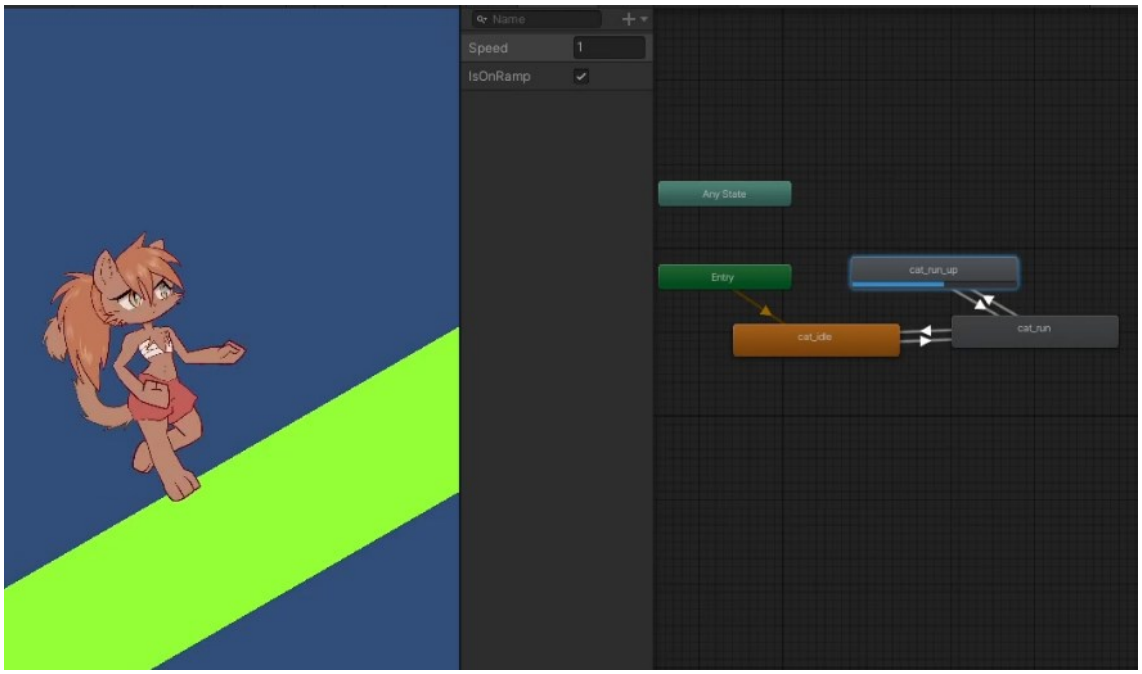
Ylämäet sivukelauksissa spriteillä voivat olla yllättävänkin ongelmallisia; juoksuanimaatiot pitää tehdä erikseen, niin ylös kuin alas, ja vaihtoehtoisesti muihin toimintoihin, kuten hyökkäysanimaatioihin. Ratkaisuna voi olla spritejen kääntäminen, mutta se ei toimi kaikissa tapauksissa (Inbound Shovel 2024.).

Parametreja voi kutsua pelimoottorissa ohjelmistollisesti, esimerkiksi Unityssä tehdyn produktion Jellyfishissä hahmon menettäessä elämäpisteitä, unohtui animoida vahinkoanimaatio. Ratkaisu oli pelihahmon läpinäkyvyyttä hallitsevan parametrin ohjelmointi *C#*-kielellä vilkuttamaan nopeaan tahtiin edestakaisin elämäpisteen vähentyessä (kuva 22), samalla tarjoten pelaajalle visuaalisen merkin milloin pelihahmolla oli iskun jälkeinen koskemattomuus.

KUVA 22. Yllä pieni osa kokonais scriptistä, alla Jellyfishin läpinäkyvyys vilkkuu elämäpisteiden vähentyessä.

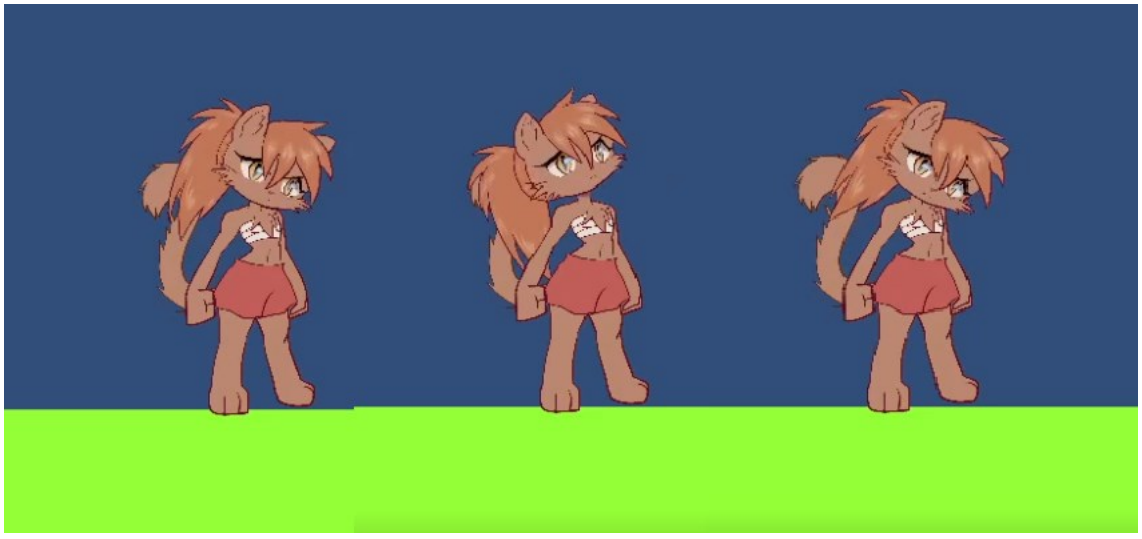
Läpinäkyvyyden voi koodata spriteille tai 3D-asseteillekkin, mutta tämä oli vain yksi mahdollinen esimerkki nimenomaan parametrin käytöstä. Parametri voi tarkoittaa kuitenkin laidasta laitaan erilaisia toimintoja. Parametreja voi luoda monimutkaisempiin tilanteisiin sopiviksi ja olla kutsuttavissa ohjelmiston avulla, eli periaatteessa on mahdollista tuottaa proseduraalisia animaatioita. Live2D-parametrien käyttö ohjelmallisesti voi tarjota paljon joustavuutta dynaamisiin animaatioihin ja pelin sisäisiin reaktioihin. Näitä parametreja voi kutsua luomaan räätälöityjä animaatioita tai reaktioita pelitapahtumiin.

Kuvassa (kuva 23) pelaajahahmo on tuotu Unity -pelimoottoriin ja animaatiot *Animator Controller*issa säädetty erottamaan ramppi, jolloin hahmon juoksuanimaatio vaihtuu ylämäkeen sopivaksi.



KUVA 23. Scripti erottaa ”rampiksi” merkatun objektin ja vaihtaa animaation oikealla olevassa animator controllerissa.

Tämä on tehty valmiina animaationa Live2D Cubismissa, mutta konsepti on mahdollinen tuottaa osaavalla ohjelmoijalla, jolloin parametreista voisi suoraan ohjata ylämäkeen liikettä sopivaksi. Osaaminen rajoittui ohjelmoinnin osalta, niin tätä ei voitu toteuttaa. Konseptin todistamiseksi kirjoitettiin pieni C# -skripti, jolla hahmon päätä liikutetaan ohjelmiston avulla, kutsuen merkittyjä parametreja edes takaisin (kuva 24).



KUVA 24. Hahmon pää liikkuu ohjelmistollisesti. Kuvassa näkyy myös tarkasti katsottuna hiusten fysiikoiden reaktio pään heilumiseen, nämä fysiikat voisivat olla merkittävästi voimakkaampia.

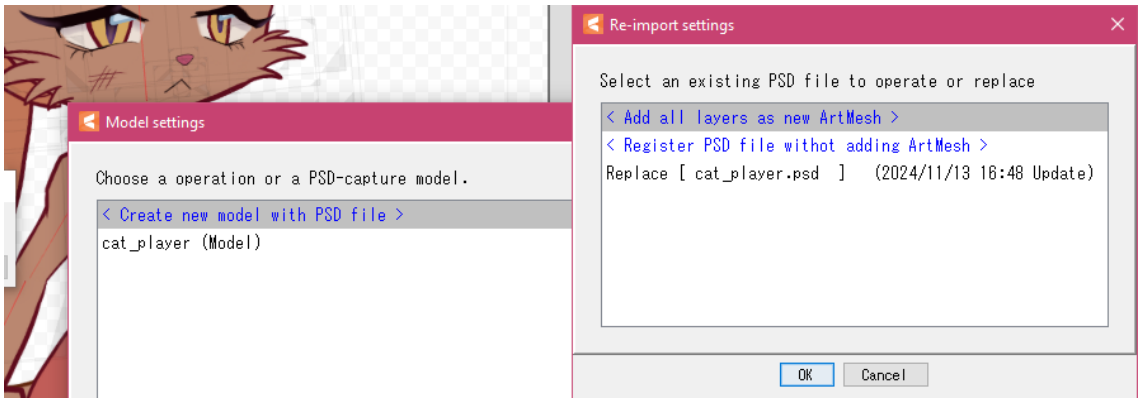
Live2D Cubism tarjoaa ilmeitä, Expressions -työkalun avulla. Ilmeet voisi tehdä suoraan kutsuttavaksi parametreista, mutta Expressions -työkalu tekee työn helpommaksi päällekkäisten animaatioiden kanssa. Tämä on hyödyllistä esimerkiksi pelissä, jossa hahmo voi näyttää uupuneelta elämäpisteiden vähentyessä.

Live2D-mallin animaatiot voivat olla erittäin modulaarisia. Kun esimerkiksi juoksuanimaatio keskeytyy hyppy- tai iskuanimaatioksi, liikkeen muutos ei ole äkillinen, kuten sprite-grafiikan vaihdoksissa. Riippuen eri parametrien määrittelyistä erillisissä animaatioissa, animaatioiden vaihtuessa liikkeet voivat mukautua interpolaation avulla, mikä tekee siirtymisestä sulavampaa.

Fysiikat, interpoloituvat animaatioiden vaihdokset, ohjelmistollisesti kutsuttavat animaatiot ja vielä Expressionit tuovat hahmon liikkeisiin dynaamisuutta, joka nostaa pelin visuaalista monimuotoisuutta merkittävästi, kun kaikki animaatiot eivät ole ennalta määriteltyjä ja staattisia.

7.2 Rigi ja sen hyödyntäminen

Yksi Live2D Cubismin vahvuuksista on rigien uudelleenkäytettävyys. Valmiista rigistä voi muokata tekstuuritasoja, lisätä tai poistaa osia ja hyödyntää aiempia parametreja ja rigauksia, mikä nopeuttaa toisen rigin luomista. Näin voi tehdä nopeita muutoksia valmiiseen malliin, kuten luoda vaihtoehtoisia pukuja samasta hahmosta. Vaihtoehtoisesti voi pohjarigin avulla luoda kokonaan uusia hahmoja. Tällöin erilaiset tasot ja tekstuurit voivat vaatia lisäkorjauksia, mutta perusrigi helpottaa prosessia merkittävästi (Live2D 2018.). Hahmolle voi valita lisääkö tai vaihtaako tasoja (Kuva 25).



KUVA 25. Valitessa vasemmalta cat_player (Model), voi jatkovalita korvaako tai lisääkö tasoja.

Oulu Game Labissa julkaistussa Gnome Attackissa katapultille ja Ring Leader tontulle tehtiin erilaisia vaihtuvia tekstuureita (kuva 26). Valitettavasti ajanpuutteen takia näitä ei lisätty itse peliin pelaajan valittavaksi. Kuvassa erilaisia asuja tai osia, joista oli vielä eriväriset versiot.

KUVA 26. Ylhäällä kuvassa tontun eri silmäkosmetiikkoja, alhaalla erilaisia katapultteja.

7.3 Motion capture

Valmista mallia voi käyttää webkameran avulla omien kasvojen liikkeen kaappaukseen, *mocap* (motion capture), joka lähettää kasvojen liikedatan

reaaliajassa live2D-mallin matkittavaksi. Tähän vaatii kuitenkin vielä jonkun erillisen ohjelmiston, joka lukee kasvon liikettä ja liikuttaa valmista mallia merkattujen ID:een mukaan parametrien sisällä, kuten VTube Studio, PrprLive tai FaceRig (Emilianavt 2024.). Viime vuosina suosioon nousseet avatareilla esiintyvät virtuaalitubettajat, VTuberit, ovatkin live2D-mallien suurinta käyttäjäkuntaa, mutta live2D:tä on käytetty myös animaatioissa ja videopeleissä.

Mocapin avulla voisi pelianimaatioitakin nopeuttaa, mikäli tarvitaan ääninäyttelijän yksi yhteen puhe ja kasvojen liike. Samalla mocapia voisi hyödyntää pelissä, jossa peliä pelataan omien liikkeiden pohjalta. Idea ei ole uusi, EyeToy teki mocap-pelaamisesta normaalia jo 2003 julkaistussa *EyeToy: Playssä* (2003) (Poskitt 2023.). Tästä huolimatta, nykypäivänä on yllättävän vähän pelejä, jotka hyödyntäisivät mocapia peleissään, jotka eivät ole VR-teknologiaan sidonnaisia. Tämä idea on vain konseptia, mutta peli, jota ohjataan pelaajan omalla liikkeellä liikuttaen peliavataria, voisi tuoda erilaisia pelattavuuksia pelimarkkinoille, joita dominoi lähinnä ohjaimet tai näppäimistö ja hiiri -kontrollerit.

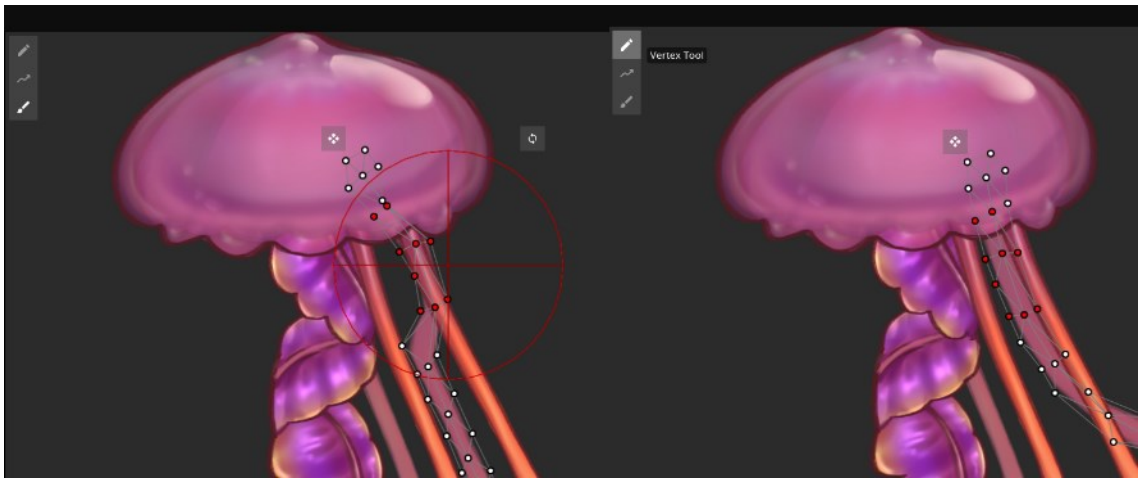
7.4 Nukke-, pala- ja luurankoanimaatio ohjelmistoja

Tässä luvussa tarkastelen lähinnä pintapuoleisesti muutamia muita animaatio-ohjelmistoja, joilla voidaan tuottaa samantyyppistä animaatiota, kuin Live2D Cubismissa.

Inochi Creator

Inochi Creator tai Inochi2D, on avoimen lähdekoodin nukkeanimointi -ohjelmisto, joka ottaa paljon mallia Live2D Cubismista. Testauksessa produktion Jellyfish -assetin käyttöä ohjelmistossa. Ohjelmisto on tyylitellympi ja miellyttävämpi silmälle, kuin Live2D Cubism, mutta ohjelmistosta puuttuu vielä joitain työkaluja, tai ne eivät ole yhtä intuitiiviset, joita Live2D Cubismissakin on tarjolla. Ohjelmistossa luodaan parametrien avainruutuihin animaatioita. Inochi2D hyödyntää "*Nodeja"*, jossa hierarkkisesti, korkeimmasta Nodesta alenevat muut Nodet. Hahmossa siis hierarkkisesti pään alle tulee hiukset, silmät ja suu, ja näilläkin Nodeilla voi olla alenevia Nodeja. Korkeimman Noden muutokset vaikuttavat aleneviin Nodeihin (Inochi2D 2022.). *Edit Meshillä* voi luoda tasoille verteksiverkot, joita voi liikutella

*Edit vertex*illä, *Path Deform Tool*illa ja *Brush Tool*illa. Ohjelmisto tarjoaa myös *Mask*auksen, jolla voi rajata tasoja kuten Live2D Cubismin Clipping ID:eillä, *Composite*llä yhdistellä, *Simple Physics* fysiikoihin ja *MeshGroup* joka muistuttaa Deform työkalua. Inochi toimii myös hyvin moitteettomasti ja nopeasti, tuntuen optimoidummalta kuin vastaavat ohjelmistot. Pelimoottoriin pitäisi pystyä tuomaan INP-tiedostoja, eli Inochi2D Puppet -tiedostoja, mutta uuden ohjelmiston käyttöönottaminen on aikaa vievää, eikä malliesimerkkejä löytynyt, muukin dokumentaatio on sekavaa tai puutteellista. Suoralla tuonnilla ei onnistuttu tuomaan INP-tiedostoa Unity projektiin, jonkin SDK tai OpenGL lisäosan puuttuessa. Lähitulevaisuudessa ohjelmiston kehittyessä, tuen ja dokumentaation laajentuessa, Inochi vaikuttaa varteenotettavalta ohjelmistolta pelinkehitykseen. Kuvassa (kuva 27) Jellyfishin lonkeroa venytellään vertekseistä vedellen Inochi2D:eessä.



KUVA 27. Jellyfishin verteksejä valitaan ja muunnellaan.

OpenToonz, Blender ja Godot

Tutkielmassa, “Avoimen lähdekoodin ohjelmien käyttö 2D-pelin grafiikoihin”, Hietamies vertailee keskenään OpenToonzia, Blenderiä ja Godotia. OpenToonz tarjosi kattavat työkalut animoimiseen, mutta tasojen erikseen tallentavat tiedostojen tuominen toisiin ohjelmistoihin osoittautui ongelmalliseksi. Tutkielman tarjoaman selvityksen pohjalta, OpenToonzilla pystyi pala-animoimaan lähes Live2D Cubismin tavalla, vaikka työkalut ja menetelmät ovat erilaisia, mutta Live2D Cubism vaikuttaa ulospäin helpommalta käyttää. Blenderissä vahvuudeksi ja heikkoudeksi nousi ohjelmiston jopa liian suuri laajuus. Blenderissä on paljon animaatioon tarjottavia työkaluja, kuten aikajana, luurankotyökalut, *greace pencil* ja

*onion skin*it. Blenderissä ei ole suoraa pala-animaatiotyökalua, eikä Hietamies onnistunut tuomaan tiedostoja ohjelmaan ilman lisäosia ja kuvat piti tuoda erillisinä *Plane*–objekteina, eli kaksiulotteisina suorakulmioina. OCA- tai COA-lisäosa auttoi tuomaan kuvan alkuperäisten tasojen sijainneilla, JSON-tiedoston avulla. Lisäosa on tätä nykyä vanhentunut ja dokumentaatio oli vähäistä. Godotkaan ei ollut ihanteellinen pala-animointiin, hahmon osia ei voinut koota alkuperäisen kuvan perusteella, vaan kuvat piti siirtää paikoilleen yksitellen. *Weight*it toimivat polygonien pisteiden kautta. Huonona puolena Hietamies vielä kommentoi dokumentaation olevan vähäistä (Hietamies 2024, 24-26).

Spine2D

Spine tai toisinaan myös Spine2D:ksi kutsuttu ohjelmisto on maksullinen 2D-luurankoanimointi-ohjelmisto, jonka tuotoksia tutkielmassa mainitaan muutamaan otteeseen. Spinessä voi tehdä luita, meshejä, jonka päälle saa vielä FFD:n, *Free-Form Deformation*in ja *Weights*ejä, suoraan muuntamaan tasoja tai auttamaan määrittelemään eri tasojen muuntautumis voimakkuuksia. Spinessä on vielä *Paths*-työkalu, jolla voi ohjata useita luita samanaikaisesti. Toistaiseksi lähes samanlaisia työkaluja, jotka ovat Live2D Cubismista tuttuja, mutta joillain eroavaisuuksilla. Lisäksi Spine tarjoaa *Bounding Box*it, jotka voi asettaa luihin kuvien tapaan, auttamaan osuma-alueiden tai fysiikoiden integraatiota. Spine painottaa animaatioissaan myös IK-teknologiaa. Spine tukee monia *toolkit*tejä ja ohjelmointikieliä sekä pääasiallinen tiedostomuoto Spinessä on JSON (Esoteric Software s.a.a.).

7.5 Peliteollisuus ja peliesimerkit

Kuten aiemmin tutkielmassani tähdennettiin, käytetään Live2D:tä synonyymina 2D-nukke- ja luurankoanimaatioiden tapaisille tekniikoille. Kappaleessa mainitsen eri pelejä, joista kaikki ei ole välttämättä tehty Live2D Cubismilla, vaan sen tyyppisillä ohjelmistoilla tai tekniikoilla.

Azur Lane ja visuaaliset novellit

Azur Lane (2017) on *sidescrolling shoot ‘em up*in ja visuaalisen novellin genrejä yhdistävä mobiilipeli, jonka on kehittänyt Manjuu ja Yongshi. Pelissä kerätään erilaisia sotalaivoja, jotka ovat anthropomorfisoitu animehahmoiksi. Azur Lane käyttää Live2D Cubismia niin sanotussa “satamassa” joka on pelin päävalikko. Päävalikossa voit valita sihteerin eri hahmoista, jotka liikkuvat ja reagoivat pelaajan toiminnoista (Azur Lane Wiki 2024.). Tutkielmassa voisi nostaa esille lisää visuaali novellien esimerkkejä, sillä etenkin Live2D:tä on hyödynnetty genren tuotoksissa suuremmin, (Wikipedia 2024h.) mutta kattavampi käyttöönotto ja “pelillisemmät” pelit antavat paremman kuvan siitä mihin kaikkeen tekniikka taipuisi.

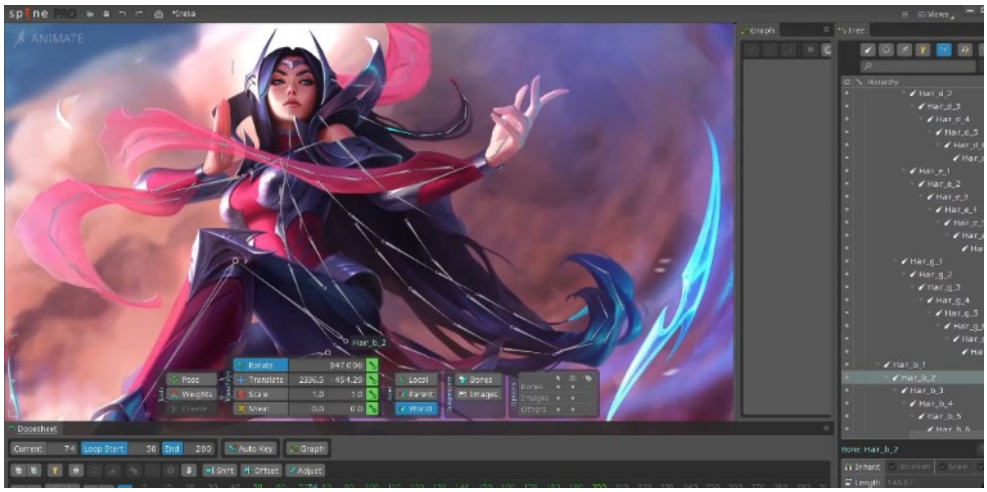
Vanillawaren taiteellinen tuotanto

Vanillawaren lähes koko tuotanto, kuten *Odin Sphere* (2007), aiemmin mainittu *Dragon’s Crown* tai *Muramasa: The Demon Blade* (2009) on tehty maalauksellisilla 2D-piirroksilla, jotka on animoitu Vanillawaren omalla studion sisäisellä ohjelmalla, missä kädet, jalat ja muut osat yhdistetään ja tekniikkaa on sanottu moniniveliseksi. Ohjelmisto muistuttaa Adobe Animatea, mutta sen suuremmin ulkopuolisia ei ole päästetty katsomaan “koneiston sisälle”. George Kamitani, Vanillawaren omistaja ja innokas 2D-grafiikan puolestapuhuja, monesti mainitsee pelien olevan tebineri-tyyliä, joka aiemminkin mainitaan tutkielmassa. Tebinerissä, miten Kamitani sen on määritellyt, tähtäimenä on tehdä kaksiulotteista maalausmaista taidetta, joka muistuttaa kolmiulotteisuutta, eli tutkielman aiheeseen palaten, 2.5D-grafiikkaa. Vanillawaren tuotantofilosofia onkin, että piirretyillä taiteilla saadaan paremmin esitettyä tunnetta ja taidetta (Sahdev 2024.). Kuvassa (kuva 28) taistelukohtauksesta otos, jossa jokainen ruutu on visuaalista herkkua silmälle.

KUVA 28. Odin Sphere Leifthrasir on remasteroitu versio alkuperäisestä, kuvassa Gwendolyn-hahmo lyömässä sieninaista keihäällä (Fernandez 2016.).

League of Legends ja Splash Art

League of Legends (2009), Riot Gamesin suosittu kilpailullinen moninpeli useilla hahmoilla, joilla on erilaisia rooleja ja kykyjä. Tavoitteena on tuhota vastustajan "Nexus", tai tukikohta. League of Legendsissä ei itse pelimoodissa ole live2D:een tyyppisiä assetteja tai animaatioita, vaan peli on kolmiulotteinen isometrisestä kuvakulmasta kuvattu 2.5D-kokemus. Hahmoilla on kuitenkin erilaisia "*Splash Artteja*", jotka lisäävät hahmojen mielenkiintoa, jossa kaksiulotteinen kuva on leikattu osiin ja animoitu mahtipontisen vaikutelman saavuttamiseksi, esimerkiksi Adobe Animatella, After Effectsillä, Spline2D:llä tai vaikka pelimoottorin 3D-työkaluilla. Virallista opetusohjelmaa ei Riot Games ole suoraan tarjonnut yleiseen jakoon, mutta moni fanianimaatio näyttää lähes identtiseltä virallisiin splash artteihin, kuten (kuva 29) Nickson Irealian animoitu kuvitus (Nickso 2021.).



KUVA 29. Irelia pelihahmon hahmoa rigataan Spinessä, kuvakaappaus videolta. (Nickso 2021.).

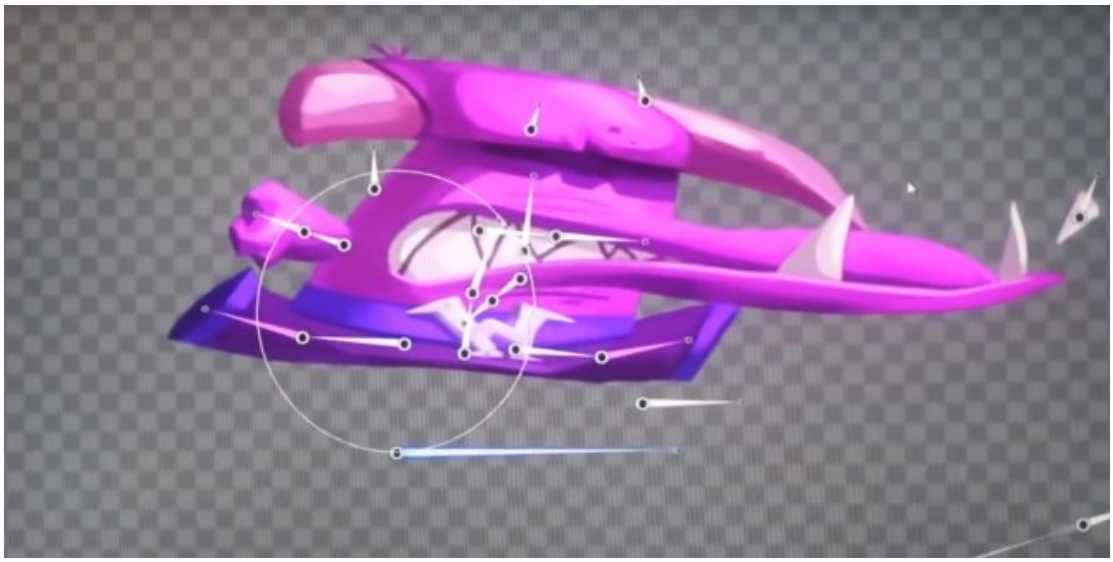
Klei Entertainmentin animaatiokirjastot

Klei Entertainmentilla on useita pelejä, joissa on hyödynnetty animaatiotekniikoita, kuten *Don't Starve* (2013), *Mark of the Ninja* (2012) tai *Shank* (2010). Klei Entertainmentin tavoite on tuottaa sulavaa 2D-animaatiota 60 ruutua sekunnissa konsoleillakin HD-laadulla. Studiolla on taustaa perinteisistä animaatioista ja hyödyntävätkin karkeita käsinpiirrettyjä animaatioita mallinaan lopullisiin Flash animaatioihin. Lopullisissa animaatioissa, piirrokset piirretään puhtaaksi *vektoreina*, ja jokainen erillinen nivel merkataan uutena *symbolina* Flashissä. Symboli Flashissä tarkoittaa viiteobjektia, joka sisältää joukon kuvia aikajanalla. Studion työtavalla lopullisia eri osien variantteja tulee useita, joista voi rakentaa moneksi taipuvia animaatioita (kuva 30). Animaatiot animoidaan 30 ruutua sekunnissa ja interpolaatio tuottaa ne korkeammiksi. Erikseen mainitaan myös, että periaate on kuin perinteisen animaation ajoituksissa, joita käsiteltiin myös animaation 12 periaatteen kappaleessa: ykkösillä animointia, josta saadaan interpolaatio kakkosilla (GDC 2025 2017.).

KUVA 30. Moneksi taipuva animaatiokirjasto Ninjalle, kuvakaappaus videolta (GDC 2025 2017.).

Raymanin UbiArt Framework

Rayman Legends (2013) on tehty UbiArt Frameworkilla, Ubisoftin omalla 2.5D-pelimoottorilla helpottamaan 2D-animaatioita (HardwareHeaven 2013.). Rayman Legendsissä on siis oma moottori, jossa nukkeanimaatioita voi tuottaa. Valitettavasti tekijöiden halusta huolimatta, sitä ei ole jaettu eteenpäin, yhtenä syynä ollen ohjelmiston opetuskäyrä (Tim 2019.). Kuvassa (kuva 31) pelihahmon luut ja venyvyydet näkyvillä.



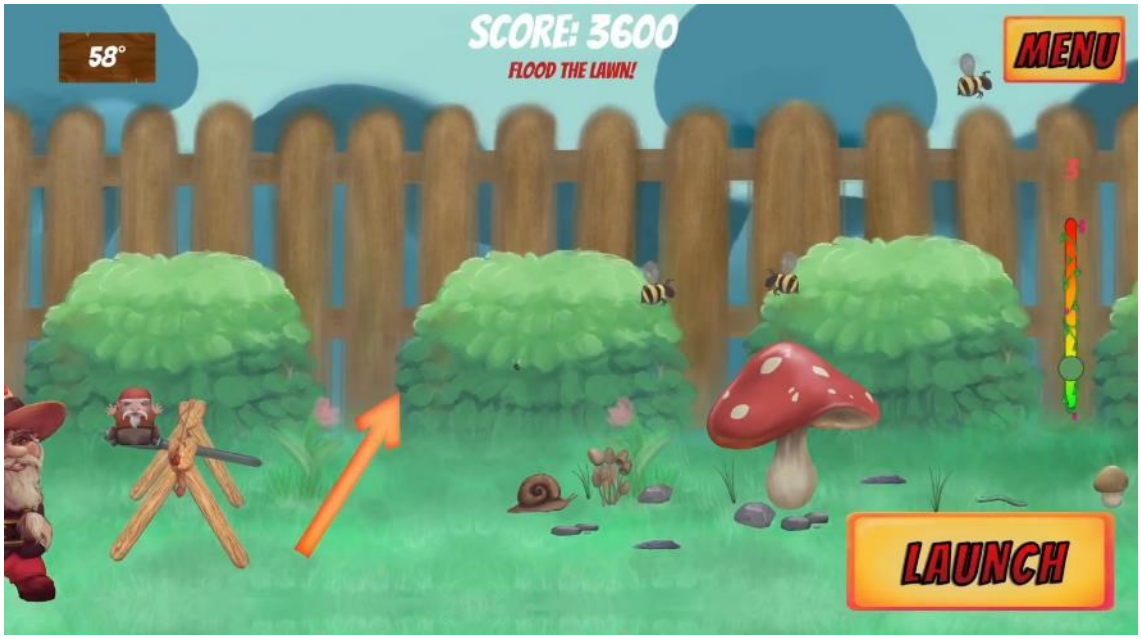
KUVA 31. UbiArt Frameworkin luutyökaluja, kuvakaappaus videolta (Hardware-Heaven 2013.).

Wulverblade ja Spine 2D

Fully Illustratedin ja Darkwind Median *Wulverblade* (2017) sijoittuu rooman aikakauden britanniaan, side-scroller *beat 'em up*issa on hyödynnetty Spine 2D:tä, mutta kehittäjät mainitsevat käyttäneensä Flashia ja Smooth Movesia ennen Spine 2D:n siirtymistä. Tätä muutosta vaikeutti, ettei hahmoja ollut suunniteltu luurankohierarkioita mielessä pitäen. Kuitenkin tekijätiimi tottui Spine 2D:een käyttöön ja he olivat vaikuttuneita lopputuloksesta (Doody 2014.).

Gnome Attack

Gnome Attack (2021), joka on tutkielmassakin muutamaan otteeseen otettu esille, oli Oulu Game Labissa tuotettu ja julkaistu 2D-sivukelaus -mobiilipeli, jossa pihatontut aiheuttavat tuhoa ympäristölleen, ampuen erilaisia tonttuja katapultilla. Alla (kuva 32) kuvakaappaus varsinaisesta pelistä, jossa pelaaja on ampumassa pihatonttua erilaisten kenttäobjektien läpi.



KUVA 32. Tonttuja ammutaan kentässä ampiaisten lävitse.

Hahmot ja vuorovaikuttavat assetit tehtiin Live2D Cubismilla. Projektin aikana opittiin käyttämään ohjelmistoa nopeasti ja erilaisiin tilanteisiin sopivaksi. Esimerkiksi rikkoutuvien esineiden hajoamisanimaatiot ovat vuorovaikutteisia.

Monimutkaisin assetti osoittautui jääkaapiksi (kuva 33), jossa jokainen osa, kuten magneetit, ovenkahvat ja jääkaapin sisältö, piti pystyä käsitellä erikseen. Pelissä tämä yksittäinen assetti pystyi muokkautumaan moniin eri tilanteisiin, muistaen oliko ovi auki, kakku hajotettu ja niin edelleen. Pelaajille saattoi tämän kentän osuuden kanssa tulla hyvin erilaisia vuorovaikutuksia (OuluGameLab 2021.).

KUVA 33. Jääkaappi assetti erilaisine parametreineen ja mahdollisina muunnoksina.

Ghost Trap

Ghost Trap (2023) on tuotettu Unreal Engine 5.1:llä, jonka teimme yhdessä Petri Launimaan kanssa *game jam*iin, lyhyeen pelinkehitystapahtumaan, “Scream Jam 2023”. Ghost Trap on 3D–peli, jossa vihollishaamut ovat Live2D Cubism malleja,

ympäristöön mallinnettiin 3D-assetteja ja käytimme lisäksi avoimen lähdekoodin 3D-kirjastoja. Peli on 3D, mutta hahmot ovat kaksiulotteisia, kääntyen aina pelaajan suuntaan parhaansa mukaan. Alkuperäinen konsepti olisi ollut, että vihollishahmojen, haamujen, raajojen syvyyksiä olisi viety syvemmiksi, luomaan vahvempi kolmiulotteisuus assetteihin, mutta lopullinen tuotos (kuva 34) omaa melko kaksiulotteisia haamuja. Tätä konseptia voisi hioa tulevaisuuteen eri projekteihin (Launimaa & Qvickman 2023.).

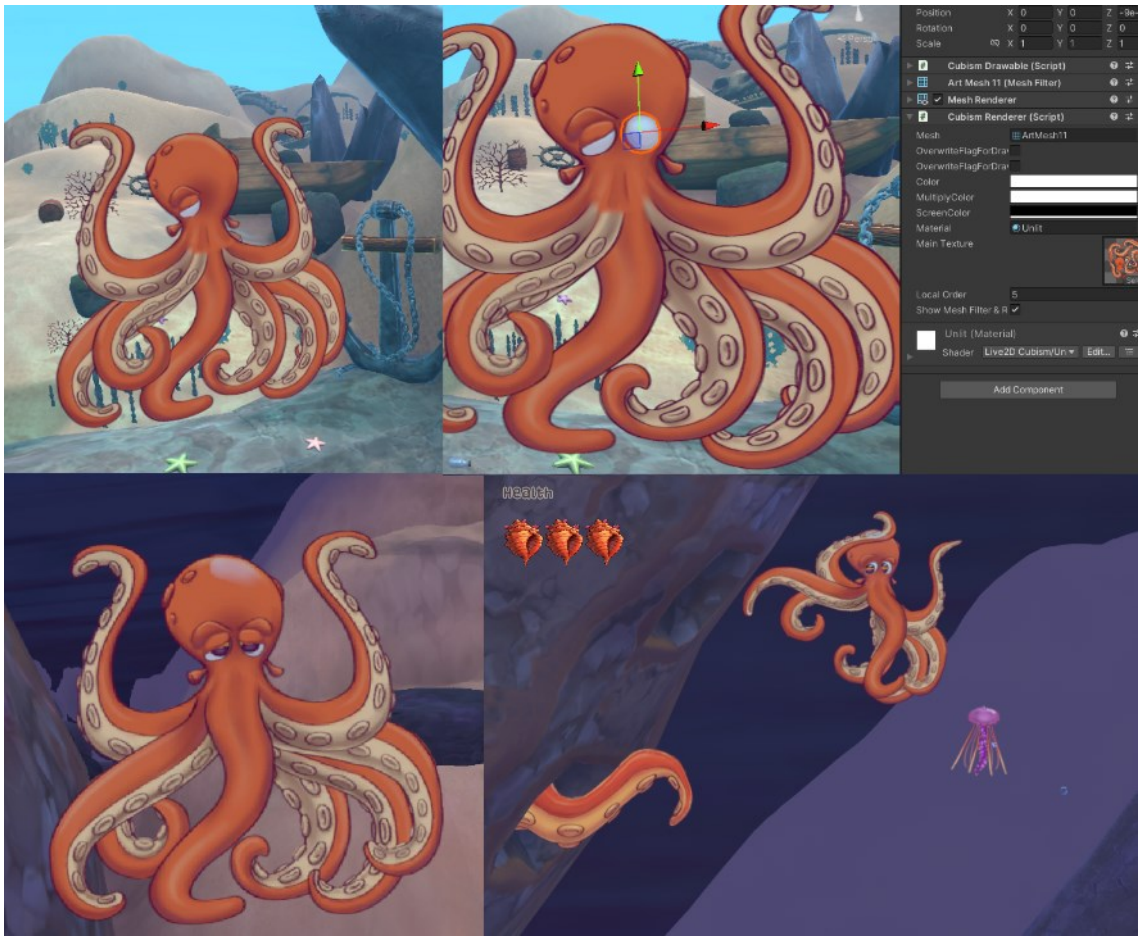
KUVA 34. vihollishaamu pelikentässä.

Jellyfish

Jellyfish on tutkinnon käytännön osuus, 2.5D-peli, jossa meduusa-pelihahmoa ohjataan läpi merenalaisten kenttien. Peli on tuotettu Unity 2022.3.20f1 versiossa. Pelin kamera säädettiin perspektiiviseen projektioon ja ohjelmistollisesti sitä tarkennettiin imitoimaan ortografista kameraa (kuva 35). Suora ortografisen kameran käyttö osoittautui liian litteäksi kolmiulotteisten ympäristöjen kannalta, mutta säädetyllä perspektiivi kameralla saatiin haluttua syvyyttä ja pientä liikkuvuutta taustaan.

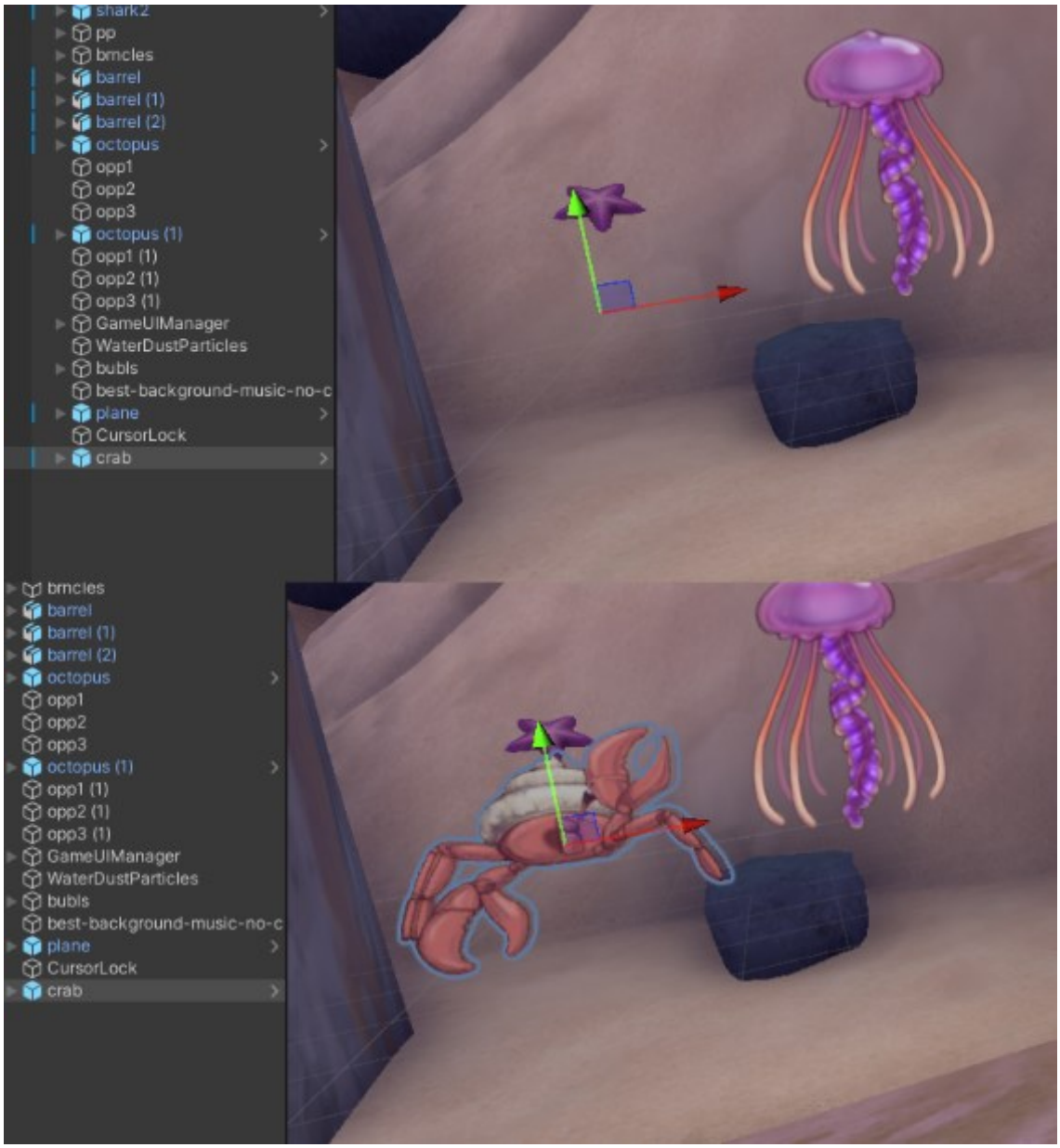
KUVA 35. Editorinäkymä yläpuolella, jossa pelikenttä kuvattu viistosti, paljastaen hahmon kaksiulotteisuuden ja alapuolella kuvakaappaus pelistä, jossa kamera on lukittu kuvakulmaan.

Assetin tuomisessa pelimoottoriin voi ilmetä odottamattomia ongelmia. Kameran vaihtaessa perspektiiviseen, assettien ArtMeshit toimivat ei-halutusti. eri tasot tulivat päällimmäisiksi niiden oikeasti sijoittuessa kauemmaksi (kuva 36). ArtMeshejä ei lähdetä vetämään *Inspector* ikkunan *Transformi*n *Position*in Z-akselilta, vaan paikallisella määrityksellä *Local Order*ista *Cubism Renderer*istä. Moniosaisten assettien kanssa tämä voi olla työlästä ja hankalaa. Valmiit muunnokset kannattaa pistää talteen tekemällä assetista *prefab*, valmiselementin, jolloin sen voi tuoda eri kenttiin joutumatta määrittelemään paikallisia määrityksiä uudelleen.



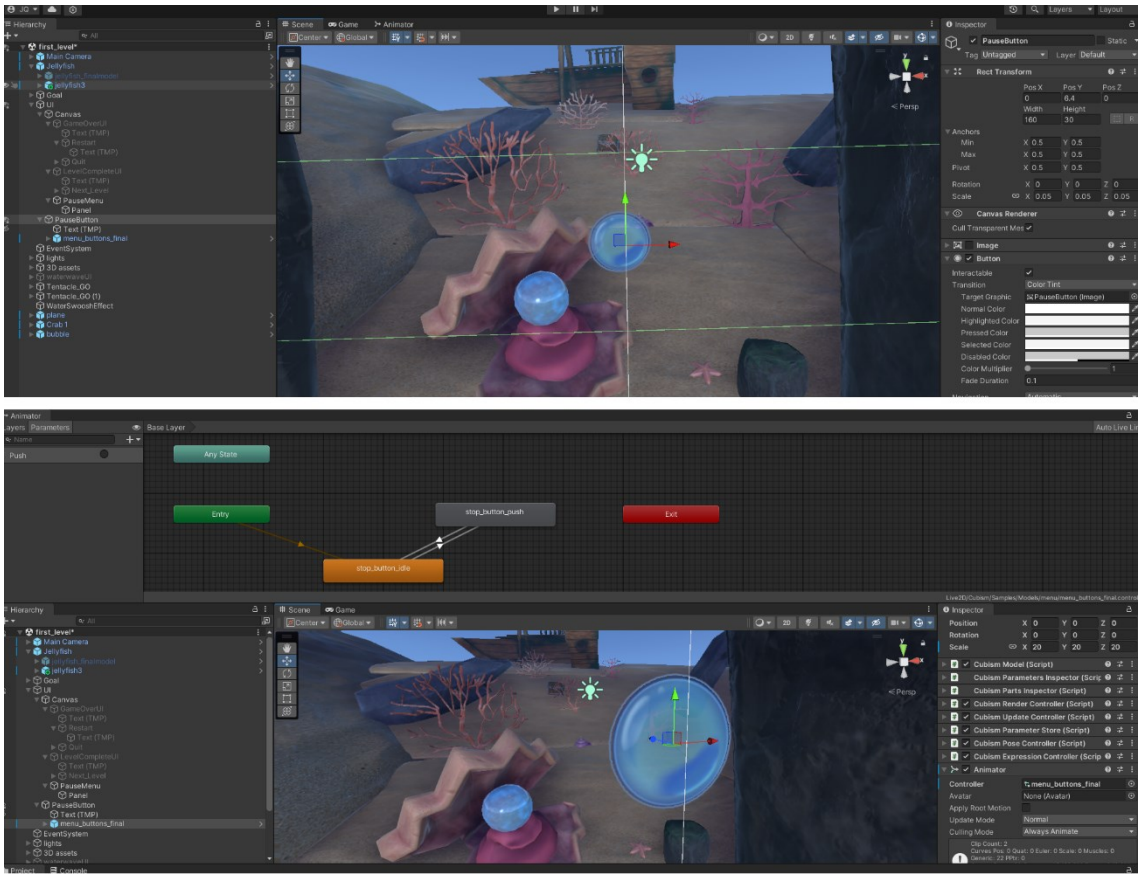
KUVA 36. Ylävasemmalla ArtMeshit päällekkäisinä. Yläoikealla Artmeshejä yksitellen määritellään Local Orderilla. Alhaalla assetti oikein asetettuna.

Toisinaan tuodessa assettia kenttäeditoriin (kuva 37), se näkyy olemassa olevana assettina editorin hierarkiassa, mutta assetti itsessään on näkymätön. Tällöin kannattaa pelitestata kenttää ja pelitestauksesta poistuttua, assetti on ilmaantunut näkyväksi editoriin. Mikäli pelitestausta estää jokin, kuten virheellinen ohjelmisto, täytyy assetti valmistella erillisessä testikentässä ja muuttaa valmiselementiksi, tai suositeltavammassa tapauksessa, virheellinen ohjelmisto korjataan.



KUVA 37. Rapu on näkymätön, mutta pelaamalla kenttää se ilmestyy.

Erityiseksi vaikeudeksi nousi Unityn UI, *User Interface*, käyttöliittymä. Se vaikutti kaikkiin muihin asetuksiin ja työkaluihin verrattuna varsin rajatulta. Projektissa olisi muutoin käytetty Live2D:lla animoituja painikkeita, mutta ne olisi pitänyt renderöidä erillisellä kameralla, joka kuvaisi vain painikkeita. Napit muutoin joko katosivat, tai olivat osa pelikenttää (kuva 38). Lopulta käyttöönotettiin sprite-pohjaiset kuvat, jotka animoitiin Unityn avulla.



KUVA 38. Painikkeita ei saatu toimimaan halutulla tavalla Unityssä.

Kaikki ympäristöassetit on 3D-mallinnettuja Blenderillä. Korkeapolygonisen assetin muodot “paistetaan” *baking*illä matalapolygoniseen assettiin jotta saadaan yksityiskohtaisia muotoja pienellä määrällä polygoneja (Timonen 2022, 21.). Paisto ja tekstuurimaalaus tuotettiin Substance Painter -ohjelmistolla. Hahmot ovat Live2D-assetteja. Tuotannossa tähdättiin tuottamaan 2.5D-peli, jossa päästäisiin 3D-mallintamaan ja hyödyntämään Live2D-assetteja. Kaikki pelin graafiset assetit on itse tuotettuja ja koodit lainattu eri lähteistä, josta mukautettu peliin sopivaksi (Qvickman 2024.).

Valitettavasti kenttien tekeminen ja koodit veivät merkittävästi enemmän aikaa rajallisella osaamisella, etenkin kun peli alun perin tuotettiin Unreal Enginessä, johon ehdittiin tekemään enemmän kenttiä ja opiskelemaan moottorin *Nanite*-teknologiaakin. Tästä aiheutuen ei ehditty syventymään live2D-teknologian käyttöön laajemmin. Unityyn siirryttiin sen helpommin ymmärrettävän käytettävyyden ja aiemman kokemuksen perusteella. Tutkielmassa otettiin muitakin projekteja ja malliesimerkkejä esille paikkaamaan näitä puutteita.

7.6 Haasteet ja rajoitteet

Live2D:n animaatio vaatii enemmän suunnittelua ja pilkkomista ennen kuin varsinaiset animaatiot voidaan tehdä. Tekniikka vaatii alkuun paljonkin ajankäyttöä, ennen kuin työmenetelmä sujuu nopeammin. Kiirehdittynä voi tuottaa heikkoja animaatioita (Plummer 2021).

Ohjelmistot eivät myöskään ole suositeltuja tai suuremmin mainostettuja peligrafiikasta kiinnostuneille. Ongelmia kohdatessa, ohjeita ei helposti löydä. Sanni Katajakin kommentoi tutkielmassaan, ”Kolmiulotteisen 2D-hahmon luonti Live2D Cubism-ohjelmalla”, miten Live2D-teknologiasta on hankala löytää japanista käännettyä ammattidokumentaatiota, joten materiaalissa joudutaan kääntymään käyttäjien tarjoamiin opastuksiin, jotka voivat olla epäluotettavia (Kataja 2022, 48.).

Monet ohjelmat ovat maksullisia, kuten Spine2D (Esoteric Software s.a.b) tai Live2D (Live2D s.a.) ja lisenssimaksut voivat olla korkeat pienille kehitystiimeille tai indie-kehittäjille. Monilla firmoilla on ollut käytössä myös omia ohjelmistoja tähän, jota ei olla julkaistu tai edes esitelty suurelle yleisölle.

Monet indie- ja 2D-pelikehittäjät valitsevat sprite-pohjaisen animaation sen selkeyden ja historian takia. Spritejä on helpompi ymmärtää ja hallita, eivätkä ne vaadi uusia työkaluja. Monissa peleissä myös siirrytään suoraan 3D-grafiikkaan ja pelimoottorit, kuten Unity ja Unreal Engine, tarjoavat tehokkaita työkaluja 3D-animaatioihin, joihin on useita opastuksia tai ilmaisia ohjelmistoja (Riendeau 2023.).

Live2D-animaatioiden käytön vähäisyys johtuu pääasiassa suurista resurssi- ja oppimiskäyrävaatimuksista.

8 YHTEENVETO

Live2D-animaatioilla voidaan saavuttaa dynaamisia lopputuloksia, sulavaa animaatiota ja joustavuutta, mutta heikkouksia on vielä monia. Vaikka spritegrafiikka ja 3D-grafiikka ovat usein käytännöllisempiä, live2D-animaatiot ovat erityisen tehokkaita projekteissa, jotka vaativat dynaamisia ja säädettäviä animaatioita pienemmillä tiedostokoilla. Teknologian laajempi hyväksyntä edellyttää työkaluja, jotka ovat helpommin saatavilla ja integroitavissa pelimoottoreihin. Avoimien lähdekoodien ohjelmistot, kuten Inochi Creator, tai luovat ratkaisut Blender-ohjelmistossa voivat alentaa tätä rimaa.

Peleissä, joihin halutaan omaleimaista tyyliä, pehmeisiin ja joustaviin liikkeisiin, voidaan hyödyntää Live2D-animaatioita. Ne sopivat ennen kaikkea visuaalisiin novelleihin ja tarinapohjaisiin peleihin, joissa niitä onkin enemmän käytetty, hahmojen ilmeiden ja eleiden ollen tärkeässä roolissa. Siitä huolimatta tutkielmassa koetin valottaa potentiaalisia käyttöjä tekniikalle myös erilaisiin peleihin ja tämän vuoksi lisättiin tutkimuskysymykseen 2.5D-grafiikan käsittely peliympäristössä, live2D:een sulautuen hyvin tyyliin ollen kaksiulotteinen tekniikka, jolla voi luoda illuusiota kolmiulotteisuudesta.

Opinnäytetyön käytännön osuudessa tuotettiin toimiva peli. Esimerkeiksi otettiin muitakin pelejä, joissa tekniikkaa oli käytetty ja muiden tuottamiin peleihinkin valittiin tutkielmaan tuotoksia, joissa animaatiotekniikkaa on käytetty laajemmin. Toivottavasti tutkielma innostaa lukijaa tutustumaan erilaisiin live2D:een tapaisiin ohjelmistoihin ja tekniikoihin. Tutkielman suurimpia haasteita oli se, että pelialalla Live2D:een kaltaista teknologiaa on hyödynnetty varsin vähän tai sen käyttö on ollut piilossa ulkopuolisilta, mikä tekee sen laajamittaisen käytön tutkimisesta haastavaa.

9 LÄHTEET

Adobe 2024. Rigging and skeletal animation: what it is and how it works. Luettavissa: https://www.adobe.com/uk/creativecloud/animation/discover/rigging.html. Luettu: 30.11.2024.

Akupara Games 2017. Rain World. Steam Store. Luettavissa: https://store.steampowered.com/app/312520/Rain_World/. Luettu: 15.11.2024.

Asikainen, Tomi & Taskinen, Ville-Veikko 2023. Grafiikkatyylin vaikutus pelin tunnelmaan. Opinnäytetyö. Jyväskylän ammattikorkeakoulu, Liiketalouden ala, Tietojenkäsittelyn tutkinto-ohjelma. Luettavissa: https://urn.fi/URN:NBN:fi:amk-202304185489. Luettu: 19.11.2024.

Awati, Rahul 2022. Techtarget. Field of view (FOV). Luettavissa: https://www.techtarget.com/whatis/definition/field-of-view-FOV. Luettu: 13.8.2024.

Awati, Rahul 2024. 3D Mesh. TechTarget. Luettavissa: https://www.techtarget.com/whatis/definition/3D-mesh. Luettu: 15.11.2024.

Azur Lane Wiki 2024. Live2D. Luettavissa: https://azurlane.koumakan.jp/wiki/Live2D. Luettu: 19.11.2024.

Benson, James 2015. The Animation Process Of Ori & The Blind Forest. GDC 20225. Video. Katsottavissa: https://www.youtube.com/watch?v=m8lOwrWN-bEY. Katsottu: 20.11.2024.

Berbece, Nicolae 2015. Game Feel: Why Your Death Animation Sucks. GDC 2025. Video. Katsottavissa: https://www.youtube.com/watch?v=pmSAG51BybY. Katsottu: 20.11.2024.

Borogk 2021. Doom engine – Limited but still 3D. Video. Katsottavissa: https://www.youtube.com/watch?v=ZYGJQqhMN1U. Katsottu: 2.8.2024.

Brian 2013. Nintendo shows its “trick” for Zelda: A Link Between Worlds’ top-down view. Nintendo Everything. Luettavissa: https://nintendoeverything.com/nintendo-shows-its-trick-for-zelda-a-link-between-worlds-top-down-view/. Luettu: 7.11.2024.

Brunner, Doug 2024. Frame Rate: A Beginner's Guide. TechSmith. Luettavissa: https://www.techsmith.com/blog/frame-rate-beginners-guide/. Luettu: 30.11.2024.

Carpe Fulgur LLC 2011. Chantelise - A Tale of Two Sisters. Steam Store. Luettavissa: https://store.steampowered.com/app/70420/Chantelise__A_Tale_of_Two_Sisters/. Luettu: 24.10.2024.

Carter, Paige 2020. Supergiant Games. Inside Hades - 3D Modeling & Rigging. Video. Katsottavissa: https://www.youtube.com/watch?v=cYJ6d1ifSqA. Katsottu: 5.8.2024.

Cheng, Wyatt 2018. Blizzcon. Pelikonferenssi 2.11.2018. Sitaatoitu 5.8.2024.

Clark, Jake 2017. Cuphead's Animation Process and Philosophy. GDC 2025. Video. Katsottavissa: https://www.youtube.com/watch?v=RmGb-jU3uVQ. Katsottu: 22.11.2024.

Cutie Dragon 2021. Live2D - Handy & Timesaving Tools. Video. Katsottavissa: https://www.youtube.com/watch?v=iyTFpGlwDY0&t=0s. Katsottu: 29.11.2024.

Cutie Dragon 2022. Live2D Cubism Tutorial - Basic Tools & Navigation. Video. Katsottavissa: https://www.youtube.com/watch?v=3GNCJb3aylk. Katsottu: 29.11.2024.

Doody, Evan 2014. Wulverblade Intro to Skeletal Animations. WulverBlade. Luettavissa: https://wulverblade.com/wulverblade-intro-to-skeletal-animations/. Luettu: 20.11.2024.

DragonBones 2017. DragonBones Animation Solution. Luettavissa: https://dragonbones.github.io/en/animation.html. Luettu: 12.11.2024.

Emilianavt 2024. Best VTuber software. Github Gist. Luettavissa: https://gist.github.com/emilianavt/cbf4d6de6f7fb01a42d4cce922795794. Luettu: 9.9.2024.

Erhard, Via 2022. 8 Brilliant Stop Motion Video Games. Gamerant. Luettavissa: https://gamerant.com/brilliant-stop-motion-video-games/. Luettu: 7.11.2024.

Esoteric Software s.a. Purchase Spine.Luettavissa: https://esotericsoftware.com/spine-purchase. Luettu: 21.11.2024.

Esoteric Software s.a. What is Spine? Luettavissa: https://esotericsoftware.com/spine-in-depth#Features. Luettu: 21.11.2024.

Fernandez, Jorge S. 2016. Odin Sphere Leifthrasir. DarkZero. Luettavissa: https://darkzero.co.uk/game-reviews/odin-sphere-leifthrasir-ps4/. Luettu: 20.11.2024.

Floyd, Dan 2019. TIMING - The 12 Principles of Animation in Games. New Frame Plus. Video. Katsottavissa: https://www.youtube.com/watch?v=rHEJZXvFc5I. Katsottu: 29.9.2024.

Floyd, Dan 2020a. ANTICIPATION - The 12 Principles of Animation in Games. New Frame Plus. Video. Katsottavissa: https://www.youtube.com/watch?v=28s1Hv3Zqlo. Katsottu: 29.9.2024.

Floyd, Dan 2020b. SLOW IN & SLOW OUT - The 12 Principles of Animation in Games. New Frame Plus. Video. Katsottavissa: https://www.youtube.com/watch?v=3jNiNctcQ4c&list=PLugegG07di3886WYN6u7v9BeBd0VFG3_J&index=4. Katsottu: 29.9.2024.

Floyd, Dan 2021. FOLLOW THROUGH & OVERLAPPING ACTION - The 12 Principles of Animation in Games. New Frame Plus. Video. Katsottavissa: https://www.youtube.com/watch?v=rYtrV1IChsA&list=PLugegG07di3886WYN6u7v9BeBd0VFG3_J&index=5. Katsottu: 29.9.2024.

Floyd, Dan 2024. ARCS - The 12 Principles of Animation in Games. New Frame Plus. Video. Katsottavissa:

https://www.youtube.com/watch?v=lOzgxMgAnxQ&list=PLu-gegG07di3886WYN6u7v9BeBd0VFG3_J&index=6. Katsottu: 29.9.2024.

Garcia, Matteo 2024. The Art of Level Design in Video Games. Gamer Tag Guru. Blogi. Luettavissa: https://gamertagguru.com/blog/level-design-in-video-games. Luettu: 29.11.2024.

Garver, S., Adamo-Villani, N. & Dib, H. 2018. The Impact of Visual Style on User Experience in Games. ResearchGate. Luettavissa: https://www.researchgate.net/publication/322323740_The_Impact_of_Visual_Style_on_User_Experience_in_Games. Luettu: 19.11.2024.

Gayton, Christina 2023. VTubers, explained, They are weird and I love them. Polygon. Luettavissa: https://www.polygon.com/videos/2023/8/10/23827568/what-are-vtubers-explained. Luettu: 1.12.2024.

GDC 2025 2017. 2D Animation at Klei Entertainment. Video. Katsottavissa: https://www.youtube.com/watch?v=8_KBjd0iaCU. Katsottu: 21.11.2024.

Grubdog 2021. Super Mario 64 – Still Wallkicking. Pietriots. Luettavissa: https://pietriots.com/2021/04/23/super-mario-64-still-wallkicking/. Luettu: 20.11.2024.

HardcoreGamer99 2019. Giantbomb. Interceptor. Luettavissa: https://www.giantbomb.com/interceptor/3030-36563/. Luettu: 31.7.2024.

HardwareHeaven 2013. How Rayman Legends Is Made!. Video. Katsottavissa: https://www.youtube.com/watch?v=y-chi097uV4&t=14s. Katsottu: 8.11.2024.

Helpshift 2023. What is an Indie Game and Why is It So Popular. Verkkosivu. Luettavissa: https://www.helpshift.com/blog/what-is-an-indie-game-and-why-is-it-so-popular/. Luettu: 5.8.2024.

Hietamies, Aureliina 2024. Avoimen lähdekoodin ohjelmien käyttö 2D-pelin grafiikoihin. Opinnäytetyö. Oulun ammattikorkeakoulu, Tietojenkäsittely. Luettavissa: https://urn.fi/URN:NBN:fi:amk-2024060119589. Luettu: 21.11.2024.

Hoeschen, Rory 2024. The Hypnotizing Simplicity of Hollow Knight. Scene and Heard. Luettavissa: https://www.sceneandheardnu.com/content/2024/5/1/the-hypnotizing-simplicity-of-hollow-knight-mark-ii. Luettu: 20.11.2024)

Inbound Shovel 2024. Why Slopes are Shockingly Difficult for Indie Game Devs. Video. Katsottavissa: https://www.youtube.com/watch?v=rHMJccb5IOM&t=635s. Katsottu: 20.11.2024.

Inochi2D 2022. Nodes. Dokumentaatio. Luettavissa: https://docs.inochi2d.com/en/latest/creator/nodes/index.html. Luettu: 21.11.2024.

Inochi2D 2024. About Inochi2D. Luettavissa: https://inochi2d.com/. Luettu: 11.10.2024.

Jakobsson, Joar 2017. The Rain World Animation Process. GDC 2025. Video. Katsottavissa: https://www.youtube.com/watch?v=sVntwsrjNe4&t=169s. Katsottu: 10.10.2024.

Kataja, Sanni 2022. Kolmiulotteisen 2D-hahmon luonti Live2D Cubism -ohjelmalla. Opinnäytetyö. Tampereen ammatikorkeakoulu, Tietojenkäsittelyn tutkinto-ohjelma. Luettavissa: https://urn.fi/URN:NBN:fi:amk-2022120225958. Luettu: 1.12.2024.

Kevuru Games 2023. 2D GAMES VS 3D GAMES: KEY FEATURES, PRICING, AND TOP GENRES [+6 TRENDS FOR 2024]. Luettavissa: https://kevurugames.com/blog/differences-between-2d-games-vs-3d-games/. Luettu: 15.7.2024.

Kiwi 2019. Medium. 3D Gaming. Luettavissa: https://medium.com/@kkwilson_11016/3d-gaming-88fd2a77f8af. Luettu: 13.8.2024.

Knight, Ste 2021. What Are 2.5D Games? How They Differ From 2D and 3D Games. MakeUseOf. Luettavissa: https://www.makeuseof.com/what-are-2-5d-games-2d-3d/. Luettu: 30.11.2024

Knight, Ste 2021. What Are 2.5D Games? How They Differ From 2D and 3D Games. MakeUseOf. Luettavissa: https://www.makeuseof.com/what-are-2-5d-games-2d-3d/. Luettu: 30.11.2024.

Koulukino 2024. Animaation eri tekniikat. Luettavissa: https://www.koulukino.fi/oppimateriaalit/boksitrollit/animaation-eri-tekniikat/. Luettu: 14.8.2024.

Kristianto, Donny 2023. 2023 Gaming Spotlight: Mobile Is Set to Surpass $108 Billion This Year — Maintaining Its 2.7x Global Revenue Lead Over PC/Mac. Blogi. Luettavissa: https://www.data.ai/en/insights/mobile-gaming/2023-gaming-spotlight-report/. Luettu: 5.8.2024.

Lark Editorial Team 2024a. Texture Atlas. LarkSuite. Luettavissa: https://www.larksuite.com/en_us/topics/gaming-glossary/texture-atlas. Luettu: 30.11.2024.

Lark Editorial Team 2024b. Procedural Animation. Larksuite. Luettavissa: https://www.larksuite.com/en_us/topics/gaming-glossary/procedural-animation. Luettu: 5.11.2024.

Launimaa, P. & Qvickman, J 2023. GhostTrap. Tposers. Luettavissa: https://tposers.com/games/view/ghosttrap. Luettu: 30.11.2024.

Live2D 2018. Re-import PSDs. Live2D Manuals & Tutorials. Dokumentaatio. Luettavissa: https://docs.live2d.com/en/cubism-editor-manual/psd-re-import/. Luettu: 9.9.2024.

Live2D 2022a. Deform Paths. Live2D Manuals & Tutorials. Dokumentaatio. Luettavissa: https://docs.live2d.com/en/cubism-editor-manual/deformpath/. Luettu: 9.9.2024.

Live2D 2022b. Repeat. Live2D Manuals & Tutorials. Dokumentaatio. Luettavissa: https://docs.live2d.com/en/cubism-editor-manual/repeat/. Luettu: 9.9.2024.

Live2D 2023a. About Deformers. Live2D Manuals & Tutorials. Dokumentaatio. Luettavissa: https://docs.live2d.com/en/cubism-editor-manual/deformer/. Luettu: 9.9.2024.

Live2D 2023b. About Parameters. Live2D Manuals & Tutorials. Dokumentaatio. Luettavissa: https://docs.live2d.com/en/cubism-editor-manual/parameter/. Luettu: 9.9.2024.

Live2D 2023b. Glue. Live2D Manuals & Tutorials. Dokumentaatio. Luettavissa: https://docs.live2d.com/en/cubism-editor-manual/glue/. Luettu: 9.9.2024.

Live2D 2024a. About ArtMeshes. Live2D Manuals & Tutorials. Dokumentaatio. Luettavissa: https://docs.live2d.com/en/cubism-editor-manual/concept-of-art-mesh/. Luettu: 9.9.2024.

Live2D 2024c. About Physics. Live2D Manuals & Tutorials. Dokumentaatio. Luettavissa: https://docs.live2d.com/en/cubism-editor-manual/physics-operation/. Luettu: 9.9.2024.

Live2D 2024d. Live2D Cubism SDK. Luettavissa: https://www.live2d.com/en/sdk/about/. Luettu: 19.11.2024.

Live2D s.a. Everything a Live2D Creator Needs. Luettavissa: https://store.live2d.com/en/. Luettu: 21.11.2024.

Mansikka, Hanna 2015. 2D- JA 3D-PELIGRAFIIKAN OMINAISUUDET. Opinnäytetyö. Lahden ammattikorkeakoulu, Tekniikan ala, Mediatekniikan koulutusohjelma. Luettavissa: https://urn.fi/URN:NBN:fi:amk-201504073979. Luettu: 28.11.2024.

Marr, Bernard 2021. What is Augmented Reality (AR) In 60 Seconds. Katsottavissa: https://www.youtube.com/watch?v=XPNUmcEOYW0. Katsottu: 1.12.2024.

Nickso 2021. Irelia Animated Illustration - League of Legends - Timelapse Fanart. Video. Katsottavissa: https://www.youtube.com/watch?v=xQggLcOLkbk. Katsottu: 19.11.2024.

Omori, Kira 2021. Live2D Cubism Expressions Tutorial - toggles, switching textures, stickers [2D VTuber Model Avatar]. Video. Katsottavissa: https://www.youtube.com/watch?v=xPJAnzFAuhU. Katsottu: 12.11.2024.

OuluGameLab 2021. Gnome Attack. Google Play. Luettavissa: https://play.google.com/store/apps/details?id=com.DefaultCompany.GnomeAttack&hl=fi&pli=1. Luettu: 30.11.2024.

Pezzi, Gustavo 2022. Pikuma. Isometric Projection in Game Development. Luettavissa: https://pikuma.com/blog/isometric-projection-in-games. Luettu: 13.8.2024.

Pietiläinen, Valtteri 2020. Kolmiulotteisuus 2D-pelissä, 2.5D- ja esirenderöity 3D-grafiikka. Opinnäytetyö. Metropolia Ammattikorkeakoulu, Medianomi, Viestintä. Luettavissa: https://urn.fi/URN:NBN:fi:amk-2020060115867. Luettu: 30.11.2024.

Plummer, Tyriq 2022. 2D Animation for Games: A Primer. GDC 2025. Video. Katsottavissa: https://www.youtube.com/watch?v=PKZJmHrG4Yw. Katsottu: 19.11.2024.

Pluralsight 2022. Differences between Displacement, Bump and Normal Maps. Blogi. Luettavissa: https://www.pluralsight.com/blog/film-games/bump-normal-and-displacement-maps. Luettu: 14.8.2024.

Pocket Gamer 2015. WHAT IS 8-BIT? | What are 8-bit graphics, anyway?. Video. Katsottavissa: https://www.youtube.com/watch?v=QaIoW1aL9GE. Katsottu: 14.8.2024.

Poskitt, Matt 2023. The Making Of EyeToy: Play, The PS2 Casual Hit That Predated Wii And Kinect. Time Extension. Luettavissa: https://www.timeextension.com/features/the-making-of-eyetoy-play-the-ps2-casual-hit-that-predated-wii-and-kinect. Luettu: 9.8.2024.

Qvickman, Jenna 2024. Jellyfish. Itch.io. Luettavissa: https://jackyq.itch.io/jellyfish. Luettu: 30.11.2024.

Rasinkangas, Teemu 2014. 2D-peligrafiikka. Opinnäytetyö. Metropolia Ammattikorkeakoulu, Mediatekniikan koulutusohjelma. Luettavissa: https://urn.fi/URN:NBN:fi:amk-201404114178. Luettu 30.11.2024.

Riendeau, Danielle 2023. The best free tools for game art. Game Developer. Luettavissa: https://www.gamedeveloper.com/art/the-best-free-tools-for-game-art. Luettu: 18.20.2024.

RocketBrush Studio. AAA, AA, Indie Games: Distinct Paths in Game Development. Blogi. Luettavissa: https://rocketbrush.com/blog/aaa-aa-indie-games-distinct-paths-in-game-development. Luettu: 14.8.2024.

Root, Dan 2019. The Animation of Indivisible. Video Game Animation Study. Video. Katsottavissa: https://www.youtube.com/watch?v=zXmArOSRPsk. Katsottu: 22.11.2024.

Ruuskanen, Antti 2018. Inverse Kinematics in Game Character Animation. Opinnäytetyö. Kajaanin ammattikorkeakoulu, Tradenomi, Tietojenkäsittely. Luettavissa: https://urn.fi/URN:NBN:fi:amk-2018120419993. Luettu: 11.11.2024.

Sahdev, Ishaan 2024. The Origins of Vanillaware - An Interview With George Kamitani. Game Design Gazette. Luettavissa: https://www.gamedesigngazette.com/2024/06/origins-vanillaware-interview.html. Luettu: 18.10.2024.

Sasaguay, Chris 2023. The Technology That Made Disney's Animated Classics More Magical. Collider. Luettavissa: https://collider.com/disney-snow-white-and-the-seven-dwarfs-multiplane-camera/. Luettu: 3.11.2024.

ShiraLive2D 2024. Exploring Live2D Animation: Definitions for Live2D Animation. Live2D Rigging Tutorials. Luettavissa: https://www.shiralive2d.com/live2d-tutorials/. Luettu: 11.10.2024.

Smeaf 2023. How Indie Games Texture EVERYTHING. Video. Katsottavissa: https://www.youtube.com/watch?v=RQkaWxp5iiM. Katsottu: 14.8.2024.

Spritesheet Editor s.a. Spritesheet. Luettavissa: https://www.spritesheeteditor.com/spritesheet.html. Luettu: 30.11.2024.

Stegner, Ben 2020. First-Person Games vs. Third-Person Games: What Are the Differences?. Luettavissa: https://www.makeuseof.com/first-person-games-vs-third-person-games-differences/. Luettu: 11.11.2024.

Studio MDHR Entertainment Inc. 2022. Cuphead - The Delicious Last Course. Steam Store. Luettavissa: https://store.steampowered.com/app/1117850/Cuphead__The_Delicious_Last_Course/. Luettu: 20.11.2024.

ThatGuyGlen 2023. How One Programmer Created Gaming's Most Complex Ecosystem. Video. Katsottavissa: https://www.youtube.com/watch?v=6Ji2q3WQE78. Katsottu: 10.10.2024.

Thomas, F. & Johnston, O. 1995. The Illusion of Life: Disney Animation. Hyperion. New York. Luettu: 12.9.2024

Tim, Geoffrey 2019. Ubisoft Boss explains why we haven't seen any UbiArt games lately. CriticalHit Gaming. Luettavissa: https://www.criticalhit.net/gaming/ubisoft-boss-explains-why-we-havent-seen-any-ubiart-games-lately/. Luettu: 8.11.2024.

Timonen, Hannu 2022. Perinteisten maalaustekniikoiden ja 3D-grafiikan yhdistäminen 2D-pelin luomisessa. Opinnäytetyö. Tampereen ammattikorkeakoulu, Tietojenkäsittelyn tutkinto-ohjelma. Luettavissa: https://urn.fi/URN:NBN:fi:amk-202203103310. Luettu: 11.11.2024.

Unity Documentation 2024. 2D game perspectives reference. Luettavissa: https://docs.unity3d.com/6000.0/Documentation/Manual/2d-game-perspective-reference.html. Luettu: 8.11.2024.

Varrio, Veera 2024. 2D-pelihahmon animaatiotekniikat. Opinnäytetyö. Metropolia Ammattikorkeakoulu, Muotoilun tutkinto-ohjelma. Luettavissa: https://urn.fi/URN:NBN:fi:amk-202404176771. Luettu: 15.11.2024.

Vitaly 2023. Quick Introduction To Skeletal Animation for Video Game Development. Night Quest Games. Luettavissa:

https://www.nightquestgames.com/quick-introduction-to-skeletal-animation-for-video-game-development/. Luettu: 20.11.2024.

Wallace, Kimberley 2021. Gameinformer. The Story Behind Supergiant Games' Bastion. Luettavissa: https://web.archive.org/web/20210416173248/https://www.gameinformer.com/2021/04/16/the-story-behind-supergiant-games-bastion. Luettu: 5.8.2024.

Wikipedia 2024a. Live2D. Luettavissa: https://en.wikipedia.org/wiki/Live2D. Luettu: 30.11.2024.

Wikipedia 2024b. Vanillaware. Luettavissa: https://en.wikipedia.org/wiki/Vanillaware. Luettu: 8.11.2024.

Wikipedia 2024c. Virtual camera system. Luettavissa: https://en.wikipedia.org/wiki/Virtual_camera_system#cite_note-11. Luettu: 12.8.2024.

Wikipedia 2024d. Chantelise – A Tale of Two Sisters . Luettavissa: https://en.wikipedia.org/wiki/Chantelise_%E2%80%93_A_Tale_of_Two_Sisters. Luettu: 11.11.2024.

Wikipedia 2024e. Top-down perspective. Video game graphics . Luettavissa: https://en.wikipedia.org/wiki/Video_game_graphics#Top-down_perspective. Luettu: 7.11.2024.

Wikipedia 2024f. Side-scrolling video game. Luettavissa: https://en.wikipedia.org/wiki/Side-scrolling_video_game. Luettu: 8.11.2024.

Wikipedia 2024g. Live2D. Luettavissa: https://en.wikipedia.org/wiki/Live2D. Luettu: 28.11.2024.

Wikipedia 2024h. Works using Live2d. Visual Novels. Live2d. Luettavissa: https://en.wikipedia.org/wiki/Live2D. Luettu: 30.11.2024.

Wong, Kristin 2020. The Power of Kawaii: How Cute, Squishy Things Influence Us. Wired. Luettavissa: https://www.wired.com/story/the-power-of-kawaii/. Luettu: 8.11.2024.

Yin-Poole, Wesley 2023. IGN. Cities Skylines 2 Dev Addresses Character Teeth Controversy. Verkkoartikkeli. Luettavissa: https://www.ign.com/articles/cities-skylines-2-dev-addresses-character-teeth-controversy. Luettu: 14.8.2024.