Antti Norja

Talon 3D-mallintaminen ja WebGL-esittelysovelluksen toteutus mobiiliin

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Mediatekniikan koulutusohjelma

Insinöörityö

28.11.2016

Metropolia

Tekijä Otsikko Sivumäärä Aika	Antti Norja Talon 3D-mallintaminen ja WebGL-esittelysovelluksen toteutus mobiiliin 40 sivua 28.11.2016
Tutkinto	Insinööri (AMK)
Koulutusohjelma	Mediatekniikka
Suuntautumisvaihtoehto	Digitaalinen media
Ohjaaja	Yliopettaja Harri Airaksinen

Insinöörityön tarkoituksena oli toteuttaa talon 3D-mallinnus ja sen esittelysovellus hyödyntämällä WebGL-ohjelmointirajapintaa. Mallin pohjana käytettiin yksityishenkilölle suunniteltua talomallia, ja tavoitteena oli optimoida sovellus toimimaan mobiililaitteilla.

Työssä perehdyttiin WebGL:n teknisiin toiminallisuuksiin ja kehitystyöhön. WebGL on Khronos Groupin kehittämä ohjelmointirajapinta, jonka avulla voidaan esittää 3D-grafiikkaa verkkoselaimessa. Lisäksi työssä tutkittiin pääpiirteittäin virtuaalitodellisuuden kehitystä ja tutustuttiin käytännön esimerkkeihin WebGL:ää hyödyntävistä VR-sovelluksista. Tuloksena selvisi, että VR-sovelluksia voidaan kehittää WebGL-apukirjastojen avulla, jotka sisältävät VR-kehitykselle vaadittavat kamera- ja liikkeenohjauskomponentit.

WebGL-sovellusta varten tehtiin suunnitelma, jossa määriteltiin työn tavoitteet, ominaisuudet ja vaadittavat työkalut. Sovelluksen kehitys aloitettiin talon 3D-mallintamisesta annettujen lähdemateriaalien mukaisesti, huomioiden mobiilijärjestelmien suorituskykyrajoitteet 3D-grafiikan tuottamisessa. Malli tuotiin PlayCanvas-pelimoottoriin, jossa lisättiin kaikki toiminnallisuudet. Lopputuloksena oli mobiiliympäristöön suunniteltu WebGL-sovellus, joka sisälsi rakennuksen virtuaaliesittelyyn vaadittavia toimintoja ja virtuaalitodellisuuslaseille suunnitellun näkymän.

Sovelluksen suorituskykyä mitattiin kahdella eri käyttökokeella, ja lopputuloksena todennettiin sovelluksen mobiilioptimoinnin toteutuneen osittain. Verkkosivustojen latausaikojen vertailussa sovellus pärjäsi hyvin suhteessa muihin mobiilisivuihin, ja se sai Googlen mobiilisivustojen soveltuvuustestistä hyvät arvosanat. Sovellus toimi 3D-grafiikan tuottamisessa vaihtelevasti eri laitteiden välillä suorituskykyongelmien vuoksi. Jatkokehitysehdotuksena sovellusta tulisi testata laajemmalla laitevalikoimalla, jotta saataisiin kattavampi määrä vertailudataa. Lisäksi sovelluksen käyttöliittymää voisi kehittää käyttäjäystävällisemmäksi.

Avainsanat	WebGL, 3D-mallinnus, virtuaalitodellisuus, PlayCanvas

Author Title Number of Pages Date	Antti Norja Modeling a 3D house and developing a mobile virtual tour application with WebGL 40 pages 28 November 2016
Degree	Bachelor of Engineering
Degree Programme	Media Technology
Specialisation option	Digital Media
Instructor	Harri Airaksinen, Principal Lecturer

The purpose of this final year project was to develop a mobile optimized virtual tour application for a 3D house model using WebGL. The house design plan was acquired from a private person.

The study began with a look at technical functionalities and the development of WebGL. WebGL is an open source programming interface, which allows displaying 3D graphics in a web browser without using any plugins. The report also reviewed the development of virtual reality with examples of VR applications utilizing WebGL. The study revealed that VR applications can be developed with APIs including a VR camera and motion components.

A plan was made for the application, which defined goals, features and required toolkits to carry out the task. The development of the application started with the modelling of the 3D house with the acquired house plan. Hardware restrictions had to be considered, while designing 3D graphics for the mobile platform, since there is a lot of variation between system performances. The finalized model was imported into the PlayCanvas game engine and the application was finished in the platform. The final WebGL application was completed with a working design for the mobile environment, featuring a virtual house tour and a VR model view.

Application performance was tested with two different methods and the successful implementation of mobile optimization was verified partially. The application got good scores in the mobile network and suitability tests. However, it did not succeed perfectly in performance tests, since some of the test devices had problems with the quality and rendering of 3D graphics. As a proposal for future development, there should be a new performance test with a higher volume of devices for more comprehensive comparison data. Additionally, the user interface could be improved to be more user-friendly.

Keywords	WebGL, 3D modeling, virtual reality, PlayCanvas

Sisällys

Lyhenteet

WebGL	Web Graphics Library on 3D-grafiikan esittämiseen verkkoselaimessa kehitetty ilman liitännäisiä toimiva ohjelmointirajapinta.
GLSL	OpenGL Shader Language on varjostinkieli, jota kutsutaan, kun grafiikkaa halutaan renderöidä näytölle OpenGL-standardissa.
VR	Virtual Reality on yleisesti käytetty lyhenne virtuaalitodellisuudesta.
CAD	Computer Aided Design on käsite ja lyhenne tietokoneavusteisesta suunnittelusta. Liitetään usein 2D- tai 3D-grafiikan piirtoon, suunnitteluun tai luonnosteluun tarkoitettuihin ohjelmistoihin.
FBX	Filmbox on tiedostoformaatti, joka on kehitetty yhteensopivaksi tiedostomuodoksi mallinnusohjelmien välille.
JSON	JavaScript Object Notation on tiedonvälitykseen kehitetty tiedostomuoto.
FPS	Frames Per Second on kuvataajuuden lyhenne, jota käytetään ilmoittamaan sekunnin aikana näytölle piirrettyjen kuvien määrän.

1 Johdanto

Insinöörityön tarkoituksena on toteuttaa talon 3D-mallinnus ja esittelysovellus hyödyntämällä WebGL-ohjelmointirajapintaa ja virtuaalitodellisuuden tekniikoita. 3D-mallin pohjana on arkkitehdin yksityishenkilölle suunnittelema talomalli.

Tavoitteena on suunnitella WebGL-sovellus mobiiliympäristö edellä siten, että kaikki sen toiminnot ja käyttöliittymä on optimoitu mobiilipäätelaitteilla toimiviksi. Sovelluksessa tulee olemaan ominaisuutena käyttää virtuaalitodellisuusnäkymää hahmotettaessa 3D-grafiikkaa. Näkymä suunnitellaan yhteensopivaksi käyttää virtuaalitodellisuuslasien kanssa hyödyntäen mobiililaitteen gyroskooppia ja kiihtyvyysanturia ympäristössä navigoimiseen.

Keskeisenä tutkimusongelmana insinöörityössä on, kuinka talon 3D-malli ja WebGL-sovellus tulisi toteuttaa, jotta ne toimisivat järjestelmäriippumattomasti mahdollisimman monessa päätelaitetyypissä. Sovelluksen tulisi olla tiedostokooltaan kevyt, jotta sen käyttö olisi sujuvaa myös mobiiliverkossa.

Insinöörityön raportti aloitetaan esittelemällä käytetyt teknologiat ja niiden tekniset ratkaisut. Sovelluksen kehityksen kulku alkaa suunnittelusta ja tämän jälkeen jatketaan itse toteutukseen. Kaikki insinöörityössä käytettävät työkalut esitellään työn suunnitteluvaiheessa, ja niiden käyttö perustellaan. Lisäksi työssä esitellään esimerkkejä muista samankaltaisista toteutuksista ja pohjustetaan omia ratkaisuja ja menetelmiä niiden pohjalta. Raportin loppuosassa esitellään saatuja tuloksia ja pohditaan sovelluksen jatkokehitystä.

2 WebGL-ohjelmointirajapinta

2.1 WebGL:n toiminnallisuus ja tekninen toteutus

Vuonna 2011 julkaistiin OpenGL ES 2.0:aan perustuva avoimen lähdekoodin ohjelmointirajapinta WebGL, jonka avulla voidaan esittää 3D-grafiikkaa verkkoselaimessa. WebGL, eli Web Graphics Library, mahdollistaa interaktiivisten kolmiulotteisten virtuaaliympäristöjen piirtämiseen HTML5-sivulle hyödyntäen laitteistokiihdytystä ilman selain-

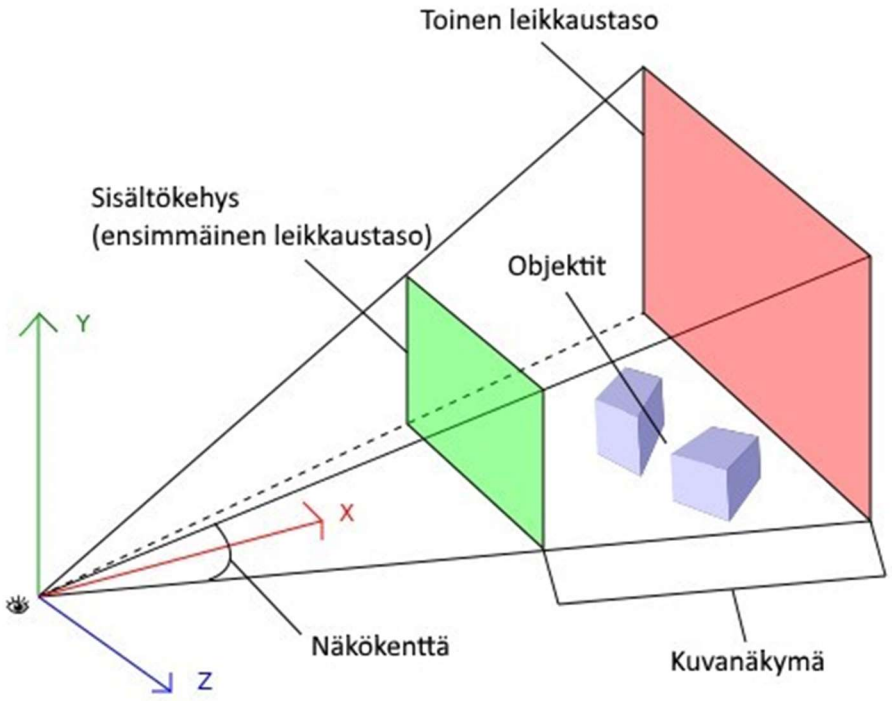
laajennusohjelmia. [1.] Tämän vuoksi suurin osa verkkoselaimista tukee WebGL:ää, eivätkä erilliset ohjelmistot ja niiden yhteensopivuudet verkkoselaimien kanssa rajoita ohjelmointirajapinnan hyötykäyttöä. WebGL ei ole laitesidonnainen, joten sen hyödyntäminen on mahdollista kaikissa yleisimmissä päätelaitteissa ja käyttöjärjestelmissä. [2.]

Jotta HTML5-sivustolla voitaisiin esittää 3D-grafiikka, tulee sivuston koodissa olla Canvas-elementti, johon kuva tuotetaan. Elementti toimii sisältökehyksenä 3D-grafiikalle. Sen ulkoasua ja mittasuhteita voidaan muokata vapaasti sivustolle istuviksi samalla tavalla kuin muitakin HTML5-elementtejä. Kaikki 3D-grafiikan visualisointi WebGL:n avulla tapahtuu Canvas-elementissä. Kun lähdekoodi sisältää elementin, voidaan WebGL:ään päästä käsiksi JavaScript-ohjelmointikielen avulla. [3.] Javascriptin avulla voidaan luoda komponentteja, kuten objekteja, valonlähteitä ja kameroita, kolmiulotteiseen avaruuteen ja sijoittaa niitä koordinaatistossa X-, Y- ja Z-akseleilla. Kamera kuvaa kolmiulotteisen tilan kaksiulotteisena näkymänä ja renderöi kuvakentän sisältökehykseen. Jos objekteja ei ole sijoitettu siten, että ne osuvat kameran kuvakenttään, ne eivät näy HTML5-sivustolla sisältökehykseen piirrettynä. [2.]

Kuten todellisessa elämässä, myös interaktiivisessa 3D-grafiikassa tulee tilaan olla sijoitettuna valonlähde, jotta objektit näkyvät kameraan renderöidyssä kuvassa. Valokomponentin ei tarvitse olla sijoitettuna koordinaatistossa kuvakentän hahmottamalle alueelle, jotta valo vaikuttaisi näkymän objekteihin. [2.] OpenGL-ohjelmoinnissa on käytössä suunta-, spotti- ja pistevaloja, jotka tuottavat erilaisia valaistustehosteita näkymässä. Jotta valot vaikuttaisivat objekteihin, tulee niihin olla määritelty pintavarjostimet, jotka reagoivat niihin kohdistuvaan valoon. Pintavarjostimet voivat muun muassa heijastaa tai absorboida valoa [4.]

WebGL:n avulla luodaan kolmiulotteisia malleja 3D-mallinnusohjelmien ja videopelimoottorien tyylillä koordinaatistoon sijoitetuilla vertekseillä, jotka muodostavat yhdistettyinä polygonimuotoja. Nämä muodot yhdessä tai yksin muodostavat kolmiulotteisen mallin tila-avaruudessa. Kolmiulotteista illuusiota sisältökehyksessä voidaan simuloida kamerakulman tai objektien sijoittelun avulla siten, että objektit eivät ole täysin kohtisuorassa kulmassa kameran kuvakenttään nähden. WebGL mahdollistaa myös komponenttien liikkeen ja interaktiivisuuden JavaScript-ohjelmoinnilla. [2.]

Kuvassa 1 on havainnollistettu kameran toimintaa kolmiulotteisessa tila-avaruudessa. Kuvassa oleva silmä esittää kameraa, ja sen ensimmäinen leikkaustaso on sama kuin HTML5-verkkosivulla upotetun Canvas-elementin sisältökehys. Näkökenttä asettaa kuvakulman koon, joka voidaan määritellä kameralle ohjelmoimalla. Ensimmäinen ja toinen leikkaustaso määrittelevät yhdessä, kuinka suuri kuvanäkymä on. Näkymään asetetut objektit renderöidään ja näytetään kehyksessä ja selaimessa. [2.]

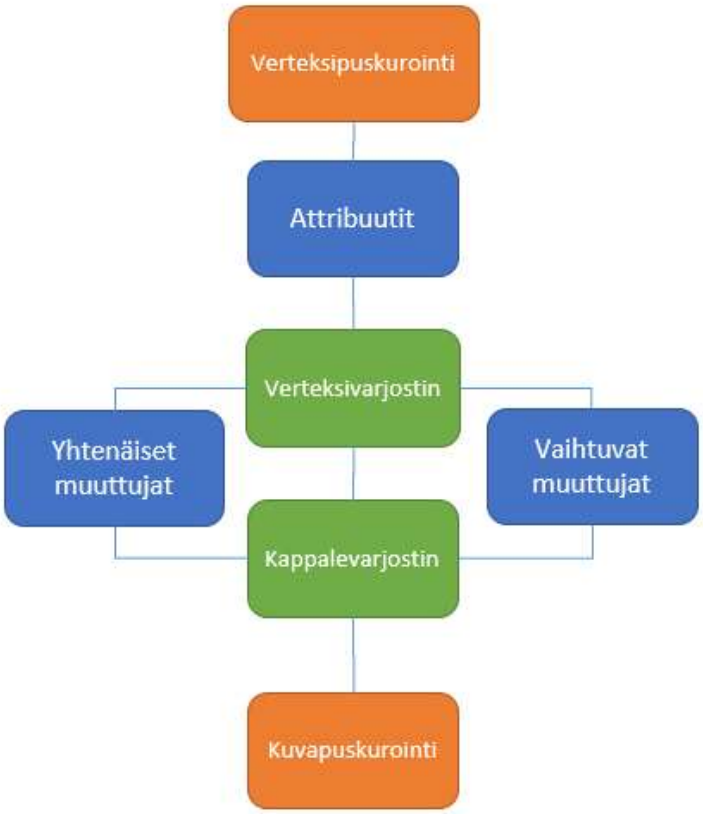


Kuva 1. Kameran toiminnan havainnollistaminen kolmiulotteisessa tila-avaruudessa [2].

Näytöllä esitetty grafiikka tuotetaan laitteen grafiikkasuorittimen avulla, minkä takia ohjelmoinnin tulee olla sellaisessa muodossa, jonka prosessori voi tulkita nopeasti. Vaikka WebGL on JavaScript-ohjelmistorajapinta, käytetään JavaScriptiä pelkästään grafiikan tietojen syöttämiseen ja niiden ulostuloon. Grafiikkasuorittimet eivät prosessoi JavaScriptiä, vaan niille toimitetaan WebGL:ssä verteksipuskuroinnin kautta verteksivarjostimelle ja kappalevarjostimelle funktiot GLSL-varjostinkielellä luettavaksi. [5.] GLSL, eli OpenGL Shader Language, on kulmakivi koko OpenGL-standardille, jota kutsutaan, kun grafiikkaa halutaan renderöidä näytölle. [6.] Verteksivarjostin antaa prosessorille

tietoa verteksien sijainnista ja mahdollistaa niiden rasteroinnin pisteiksi, viivoiksi tai kolmioiksi. Tämän jälkeen työ siirtyy kappalevarjostimelle, joka tuottaa värin ja pinnan jokaiselle pisteelle, joka piirretään puskuroituun kuvaan. [5.]

Kuvassa 2 on esitetty grafiikan renderöinnin työprosessi vaiheittain järjestelmässä. Verteksipuskurointi sisältää vaadittavat attribuuttitiedot objektien piirtämisestä, kuten verteksipisteiden koordinaatit ja värit. Tiedot lähetetään varjostimille, jotka tuottavat kaksiulotteisen kuvan, joka puskuroidaan ulos päätelaitteen näytölle. [2.]



Kuva 2. Kuvan tuottaminen grafiikkaprosessorissa [2].

2.2 WebGL:n kehitystyö

WebGL:n kehitystyöstä vastaa Khronos Group, joka perustettiin vuonna 2000. Se on monen suuren teknologiayhtiön voittoa tavoittelematon yhteenliittymä, jonka perustaja-jäseniin kuuluvat muun muassa ATI, Intel, NVIDIA ja Sun Microsystems. Kaikki nämä yritykset ovat suoritin- ja näytönohjainvalmistajia tai omistavat niitä valmistavan yksi-

kön. Yhteenliittymän tarkoituksena on kehittää avoimen lähdekoodin ohjelmointirajapintoja, jotta alalle saadaan päätelaiteriippumattomia standardeja. [7.] Erityisesti WebGL:n kehitystyön resursoinnista vastaavat Apple Inc., Google Inc., Mozilla Foundation ja Opera Software, joilla kaikilla on markkinoilla omat verkkoselainohjelmansa [1]. Muita Khronos Groupin kehittämiä ohjelmointirajapintoja on muun muassa OpenGL, OpenGL ES ja COLLADA [7].

Vuonna 2006 Mozillalla työskentelevän Vladimir Vukićevićin kehitti prototyyppejä Canvas 3D:stä, mikä tulkitaan WebGL:n ensiaskeliksi. Mozilla ja Opera olivat molemmat kehittämässä omia toteutuksiaan verkkoselaimessa suoritettavasta interaktiivisesta 3D-grafiikasta. Vuonna 2009 Khronos Group perusti yritysten kehitystyöhön pohjautuvan työryhmän. WebGL:n ensimmäinen versio julkaistiin kaksi vuotta työryhmän perustamisen jälkeen, ja se on ollut siitä lähtien jatkuvan kehityksen alla. [8.] Rajapinnan tueksi on kehitetty apukirjastoja ja ohjelmointikehyksiä, jotka helpottavat WebGL:n käyttöä [2]. Siitä on myös ollut vuodesta 2013 asti kehityksessä WebGL 2 -julkaisuversio, joka pohjautuu OpenGL ES 3.0:aan [8].

2.3 Apukirjastot tukena WebGL-ohjelmoinnille

WebGL-sovellusten ohjelmointi vaatii paljon työtä GL-kielten karkean olomuodon takia. Siihen on kuitenkin ajan mittaan kehitetty eri apukirjastoja työprosessin nopeuttamiseksi ja helpottamiseksi. [2.] JavaScript-apukirjastot ovat yleensä ulkoisia tiedostoja, jotka sisältävät ohjelmointia helpottavia valmiiksi tehtyjä komponentteja. WebGL:n tapauksessa kirjastoissa on esimerkiksi valmiiksi koodattuja malleja, kameroita ja valokomponentteja. Näiden lisäksi apukirjastoissa on työkaluja sisältökehyksen renderöintiin ja ulkoisten 3D-mallien lataamisen ja liittämiseen työhön. Apukirjastoja sisältäviä funktioita ja muuttujia voidaan hyödyntää kehitettävässä työssä, kun koodit on yhdistetty tai linkitetty toisiinsa erillisinä tiedostoina. [2; 9.]

Taulukossa 1 on listattuna WebGL-apukirjastoja ja WebGL:ää tukevia ohjelmistoja. Taulukossa on esitetty ominaisuuksia ja ulkoisten mallien formaattituet. Kehyksissä, jotka eivät tue mallintamista tai animointia, tulee käyttää apuna erillistä 3D-mallinnusohjelmistoa. [10.]

Taulukko 1. WebGL-apukirjastoja ja sitä tukevia ohjelmistoja [10].

Nimi	3D-mallinnus	Animointi	Mallien tuetut formaatit	Ominaisuudet
BabylonJS	EI	KYLLÄ	OBJ, FBX, STL, Babylon	Ohjelmointikehys pelien suunnitteluun
Blend4Web	KYLLÄ	KYLLÄ	3DS, DAE, FBX, DXF, OBJ, LWO, BVH, SVG, PLY, STL, VRML, VRML97, X3D	Pelisuunnitteluun kehitetty, täysin yhteensopiva Blender 3D -ohjelmiston ja sen ominaisuuksien kanssa
Clara.io	KYLLÄ	KYLLÄ	OBJ, FBX, Blend, STL, STP	Selainpohjainen 3D-grafiikkatyökalu
Goo Create	EI	KYLLÄ	FBX, OBJ	Pelimoottoori, jossa selainkäyttöliittymä
PlayCanvas	EI	KYLLÄ	FBX, OBJ	Pelimoottoori, jossa selainkäyttöliittymä
SceneJS	EI	KYLLÄ	OBJ	3D-visualisointiin kehitetty moottori
Three.js	EI	EI	FBX, OBJ, STL	Monikäyttöön tarkoitettu laajasti selaimiin yhteensopiva apukirjasto
Unity 5	KYLLÄ	KYLLÄ	FBX, OBJ	Pelimoottori, jossa WebGL-tuki

WebGL-apukirjastoja on kehitetty moniin eri 3D-grafiikkaa hyödyntäviin tarkoituksiin. Osalle niistä on tehty alustoja, kuten pelimoottoreita ja editoriohjelmistoja, joiden avulla niiden hyödyntäminen on entistä käytännöllisempää. Editoriohjelmistot sisältävät usein valmiita komponentteja muokattavaksi ja mahdollistavat sovellusten tuottamisen testinäkymässä ilman koodausta. [10.]

3 WebGL ja virtuaalitodellisuus

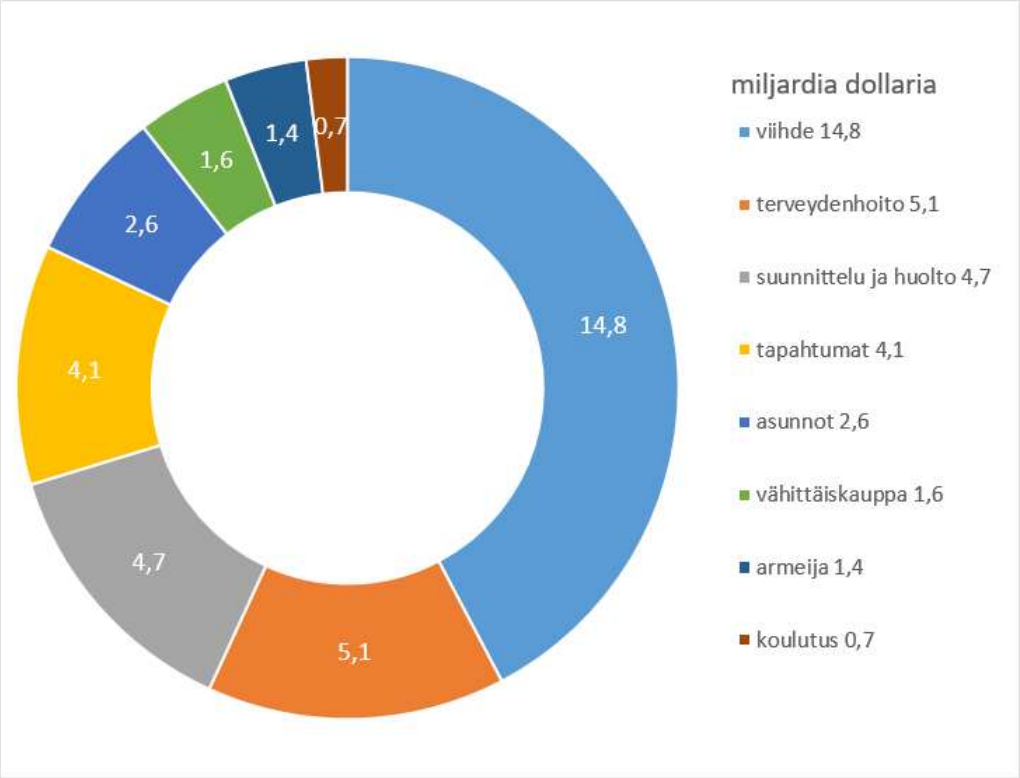
3.1 Virtuaalitodellisuus tekniikkana

Virtuaalitodellisuudella tarkoitetaan sitä, että käyttäjän aistit suljetaan hahmottamaan keinotekoisesti luotua maailmaa pois todellisesta ympäristöstä. Yleensä käsitteellä tarkoitetaan tietokoneelle luotua 3D-grafiikkaa, joka esitetään virtuaalitodellisuuslaseja hyödyntäen. Lasit sulkevat näköaistit riippuvaisiksi virtuaalisesta näkökentästä, joka hahmotetaan laseihin integroidulla tai yhteensopivalla näyttölaitteella. Lasien tueksi voidaan käyttää esimerkiksi surround-äänentoistoa tukevia kuulokkeita tai liiketunnis-

tusohjaimia, jotka tehostavat todellisuudentuntua altistamalla enemmän aisteja maailmalle. Teknologian kehittäjien määrä alalla on virtuaalitodellisuuden nopean kasvun ansiosta korkea. Merkittävimpinä laitevalmistajina voidaan pitää sellaisia yrityksiä kuin Oculus VR, HTC ja Sony. Yritysten tuotteet ovat Oculus Rift, HTC Vive ja Playstation VR. Teknologiasta käytetään yleisesti myös lyhennettä VR, joka tarkoittaa englanniksi Virtual Reality. [11.]

Älypuhelimia tukevien laitteiden ja oheissovelluksien merkittäviä kehittäjiä ovat Samsung ja Google. Lisäksi markkinoilla on myös monia pienempiä laitevalmistajia. Puhelimen kanssa käytettävät laitteet eivät vaadi näyttö- tai komponenttiteknologiaa, mikä laskee yritysten kynnystä tuottaa virtuaalilaseja kustannustehokkaasti. Kehitys keskittyy ensisijaisesti peli- ja viihdeteollisuuteen, mutta sovelluksia toteutetaan aktiivisesti myös muille aloille, kuten terveydenhoitoon ja koulutukseen. [11.]

Kuvassa 3 on esitetty Goldman Sachs -pankin arvio virtuaalitodellisuus- ja lisätty todellisuus -ohjelmistojen liikevaihdosta vuoteen 2025 mennessä. Luvut on esitetty miljardeina Yhdysvaltain dollareina. Liikevaihdossa ei ole otettu huomioon laitemyyntiä. Yhteensä pankki arvioi markkinoiden nousevan 80 miljardiin dollariin vuonna 2025. Vuonna 2016 alan kokonaisliikevaihto on noin viisi miljardia dollaria. Vertailuna esimerkiksi konsoli-, PC- ja mobiilipelien liikevaihto on 100 miljardia dollaria. [12.]



Kuva 3. Virtuaalitodellisuuden ja lisätyn todellisuuden arvioitu liikevaihto vuonna 2025 [12].

Virtuaalitodellisuuden lisäksi lisätty todellisuus on yksi kokemuksellisten teknologioiden nousevista aloista. Lisätyssä todellisuudessa keinotekoinen maailma tuodaan oikeaan ympäristöön koettavaksi, kun taas virtuaalitodellisuudessa pyritään viemään käyttäjä kokonaan keinotekoiseen maailmaan. Tietokonegrafiikka liitetään joko kameratekniikkaa käyttäen tai lisätylle todellisuudelle kehitetyillä laseilla esitettynä ympäristöön nähtäväksi. Laitteen sensorit hahmottavat kuvatun kohteen ja laskevat algoritmien avulla grafiikalle paikan ja rotaation siten, että se sopii ympäristöön luontevasti. Tekniikkaa hyödynnetään monissa älypuhelinsovelluksissa ja laitteissa, kuten Microsoftin kehittämissä HoloLens-virtuaalilaseissa. [11.]

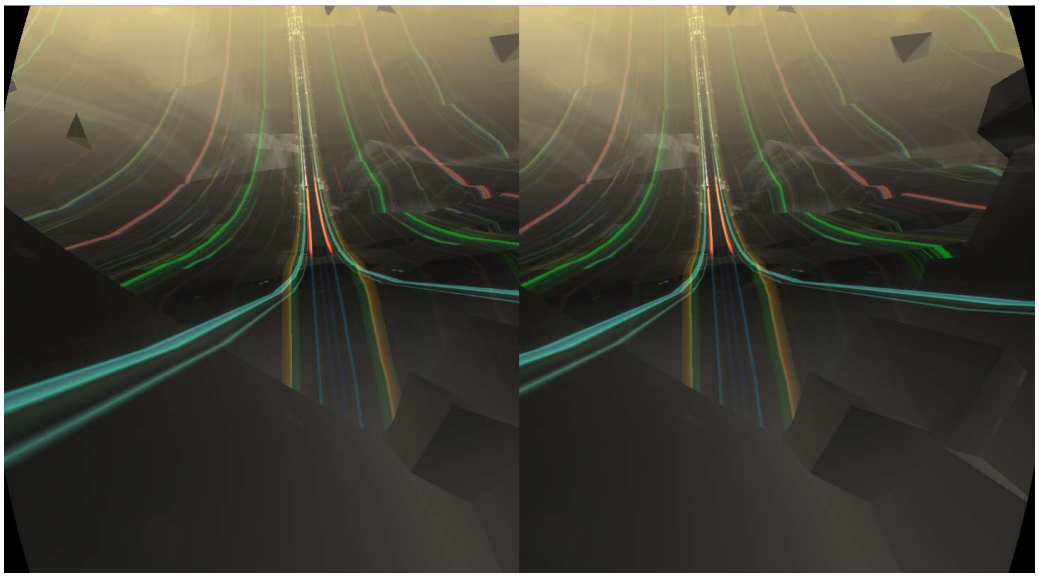
3.2 WebGL-rajapintaa hyödyntävät VR-sovellukset

WebGL:n tarjoamat mahdollisuudet avaavat mobiilisovelluskehityksen ovet JavaScript-ohjelmoijille, eikä mobiiliohjelmoinnin tuntemus ole enää välttämätöntä kehittäjille. Koska WebGL on grafiikan renderöintiin tarkoitettu rajapinta, sitä voidaan käyttää myös virtuaalitodellisuuden hahmottamiseen. Suurin osa selainpohjaisista virtuaalitodellisuussovelluksista pohjautuu laitteen orientaatiota ja liikettä seuraavien komponenttien

hyötykäyttöön, joita verkkoselain voi kuunnella. WebGL-kamerakomponentti voidaan määritellä seuraamaan laitteen liikkeitä ja imitoimaan vastaavat liikkeet kameran näkymässä. Kun mobiililaite on kytkettynä VR-laseihin, kamera imitoi käyttäjän pään liikkeitä laitteen liikkuessa. [13.]

Stereoskooppista kuvantamista voidaan käyttää VR-laseille suunnitelluissa sovelluksissa. Tekniikan avulla luodaan kuvaan kolmiulotteinen efekti kahdentamalla se molemmille silmille, mikä lisää syvyysvaikutelmaa. Kuva jaetaan keskeltä kahdeksi omaksi näkymäksi, jotka simuloivat vasemman ja oikean silmän näkökenttiä. Luonnollisesti silmien näkökentät eivät kohdistu täysin samaan pisteeseen, vaan kohdistuspisteet menevät hieman ristiin toisistaan. Tämä voidaan havaita esimerkiksi siten, että paperilla olevaa pistettä katsotaan vuorotellen kummallakin silmällä liikuttamatta päätä. Piste siirtyy näkökentässä riippuen siitä, millä silmällä sitä katsoo. Stereoskooppisessa kuvantamisessa simuloidaan näkökenttiä siten, että molemmat kuvat näkymässä havainnollistavat tilaa hieman eri kulmista. Tästä syystä kuvat poikkeavat toisistaan, kun niitä katsotaan erikseen. VR-lasien kautta katsottuna kuvat sulautuvat toisiinsa luoden syvyysvaikutelman ja 3D-efektin näkökenttään. [14.]

Kuvassa 4 on kuvakaappaus Goo Create -kehyksellä tehdystä WebGL-virtuaalitodellisuussovelluksesta. Stereoskooppisen kameratekniikan avulla laitteen näytölle on renderöity kahden kameran kuva simuloimaan ihmisen silmän näkökenttää. Kuvassa esitetty sovellus on interaktiivinen musiikkivideo, jossa 3D-maailmaa voidaan tarkastella VR-laseilla liikkeentunnistusliitännäisen avulla, joka käyttää laitteen gyroskooppia. [15.] Goo Create on PlayCanvaksen tapaan avoimen lähdekoodin pelimoottori, joka tarjoaa selainpohjaista editoria pelien kehitykseen. Erona PlayCanvakseen Goo Create ei vaadi käyttäjältä välttämättä ollenkaan koodausta, vaan tapahtumien ja ohjauksen ohjelmoinnin voi tehdä visuaalista rakennekaaviota käyttämällä. [16.] Sovellus on julkaistu Googlen ylläpitämän Chrome Experiments -yhteisön esittelysivustolla [15].



Kuva 4. WebGL-sovelluksen stereoskooppinen kameranäkymä [15].

Mobiililaitteen liikeohjaimiin päästään käsiksi apukirjaston liitännäisellä. Liitännäisellä tarkoitetaan tässä asiayhteydessä valmiiksi tehtyä kehittäjäyhteisön työkalua apukirjastojen käytön tueksi. Se on yleensä ulkoinen JavaScript-tiedosto, joka sisältää valmiin koodin tietylle komponentille. [13.] Apukirjastoja, joille on kehitetty tai jotka hyödyntävät virtuaalitodellisuutta tukevia liitännäisiä, ovat muun muassa A-Frame, Goo Create, Three.js ja PlayCanvas. A-Frame on Mozillan ylläpitämän virtuaalitodellisuustyöryhmän kehittämä alusta, jonka tarkoitus on tehdä 3D-grafiikan ja VR:n esittämisestä nopeampaa ja helpompaa. Alusta perustuu Three.js:n päälle rakennettuun kehykseen, ja sillä voidaan luoda sovelluksia mobiililaitteille, Oculus Riftille ja HTC Vivelle. Three.js on yksi edistyksellisimmistä WebGL-rajapinnan apukirjastoista, ja sen ympärille on luotu useita erilaisia liitännäisiä. [17.] PlayCanvas on selainpohjainen ohjelmointikehys ja apukirjasto, joka tarjoaa myös omia liitännäisiä valmiina käytettäväksi alustallaan.

3.3 Unity 5 ja pelimoottorien WebGL-tuki

Pelimoottoreilla, joita ei ole suunniteltu ensisijaisesti WebGL- tai HTML5-peleille, virtuaalitodellisuussovellusten julkaisut ovat yleensä itsenäisiä ladattavia tuotteita. Syy siihen on verkkoselainten suorituskyvyssä ja siinä, että kun peliä ei suoriteta käyttäjän toimesta lokaalisti, muisti ja tallennustila ovat täysin riippuvaiset palveluntarjoajasta. Selainten heikompi suorituskyky ja epävakaus lisäävät komplikaatioiden todennäköisyyttä etenkin raskaissa toteutuksissa. Lisäksi verkkoselainten versioiden ja tuottajien

erot hankaloittavat vakaiden tuotteiden kehittämistä. Pelimoottorien omat järjestelmät, jotka on rakennettu vain pelejä varten, ovat tässä tapauksessa toivotumpia, kun kyseessä on virtuaalitodellisuuspeli. Kuitenkin suositut pelimoottorit, kuten Unity 5 ja Unreal 4 Engine, tukevat WebGL-julkaisumahdollisuutta.

Unity 5 on ilmaisella tai maksullisella sopimuksella lunastettava pelimoottori, jolla voidaan luoda 2D- ja 3D-pelejä pelieditorilla. Unity 5 on uusin versio Unity-ohjelmistoperheessä, ja se tukee laajasti eri julkaisualustoja ja käyttöjärjestelmiä. Se on yksi suosituimmista pelimoottoreista, ja sillä on suuri ja aktiivinen kehittäjäyhteisö. Unityn valttikortteja on sen helppokäyttöisyys ja laaja valikoima eri toiminallisuuksia ja komponentteja pelejä varten. [18.]

Unity 5 sisältää kattavan dokumentoinnin virtuaalitodellisuuspelien suunnittelusta ensisijaisesti Oculus Rift- ja Samsung Gear -VR-laseille rakennettaville peleille. Virtuaalitodellisuutta tukevien pelien rakentaminen on tehty todella helpoksi, sillä VR-kameran renderöinti voidaan aktivoida pelkällä valikkovalinnalla ilman ohjelmointia. VR-näkymä on sisäänrakennettuna Unityn pelimoottorissa eikä vaadi mitään liitännäisiä toimiakseen. [19.] WebGL:n ja VR-kameran käyttö samanaikaisesti ei kuitenkaan ole mahdollista Unity-alustalla itsenäisesti tällä hetkellä. Nämä kaksi toimintoa pelimoottorissa sulkevat toisensa ulkopuolelle kehitysalustassa. Virtuaalitodellisuuden hyötykäyttö WebGL:n kanssa on kuitenkin mahdollista ulkoisella WebVR-rajapinnalla. WebVR mahdollistaa itsenäisten virtuaalilasien, kuten Oculus Rift ja HTC Vive, käytön WebGL-toteutuksissa. WebVR-rajapintaa hyödyntäen Unity 5:lle on kehitetty julkaisualusta ja kamerakomponentteja, joiden avulla pelit toimivat lasien kanssa. [20.]

Kuvassa 5 on esitetty kuvakaappaus SECVRITY-VR-pelistä, joka on Unity 5:llä kehitetty WebGL-virtuaalitodellisuuspeli. Peli käyttää WebVR-rajapintaa liitännäisenä Unity 5:ssä, ja toimiakseen se vaatii sitä tukevan selaimen ja Oculus Rift -VR-lasit. Pelissä voidaan tarkastella tilaa päätä kääntelemällä, kun käytössä on Oculus Rift -lasit, ja suorittaa tehtäviä kohdistamalla kuvan keskiössä oleva piste pelin tietokonenäyttöjen päälle. Kuvassa 5 näytön reunat hohtavat, kun piste on kohdistettu siihen. Pelin on kehittänyt Unity-pelimoottorille Jump, joka on verkkopelien kehittäjäyhteisö päämääränä kehittää järjestelmäriippumattomia selainpohjaisia pelejä Jump-palveluunsa. Palvelu julkaistaan vuoden 2016 aikana. [20.]

Kuva 5. Kuvakaappaus SECVRITY-VR-pelistä [20].

Unity ei täysin tue WebGL-sovellusten suorittamista mobiililaitteilla. Syy on se, että se ei käytä suoraan alkuperäistä WebGL:ää sovellusten julkaisussa, vaan erillistä vientityökalua. Työkalu kääntää ohjelmoinnin C#-kielestä C++-kieleen, minkä jälkeen se optimoidaan JavaScriptille. Prosessi tuottaa suuren määrän koodia ja on raskas verkkoselaimen muistille puskuroida läpi. Tämän vuoksi mobiiliselaimilla selainmuisti loppuu usein kesken ja sovellus kaatuu. [21.]

4 Rakennusten 3D-mallintaminen ja virtuaalinäytöt

4.1 Visuaalinen kehitys asuntoesittelyissä

Vuonna 1961 Ivan Sutherland suunnitteli ensimmäisen CAD-ohjelmiston, jota pidetään tietokonegrafiikan piirron ensiaskelina. CAD-, eli Computer Aided Design, ohjelmistolla tarkoitetaan 2D- tai 3D-grafiikan piirtoon, suunnitteluun ja luonnosteluun tarkoitettua käyttöliittymää sisältävää ohjelmistoa. Asuntojen mallintamista voidaan tehdä sille suunnitelluilla CAD-ohjelmistoilla tai yleisillä 3D-mallinnusohjelmistoilla. 3D-grafiikan piirtotyökaluilla on ollut suuri vaikutus arkkitehtisuunnitteluun, sillä se on mullistanut tavan, jolla kaavoja voidaan visualisoida yleisölle. Grafiikalla mahdollistetaan rakennusten realistisempi kuvantaminen, animointi ja tarkastelu kolmiulotteisessa ympäristössä. [22.]

Virtuaalitodellisuus tuo sisällön visualisoinnille uuden ulottuvuuden ja vaikuttaa näin myös luonnollisesti asuntojen esittelyyn. Virtuaalitodellisuutta hyödynnetään nykyisin asuntokaupassa kuvaamalla kohteita 360-asteisilla kameroilla tai yhdistämällä kuvat jälkeenpäin 3D-kuvakartoiksi. 3D-kuvakartalla tarkoitetaan kuvaryhmää, jossa kuvat muodostavat yhdessä 360-asteisen kuvan ympäristöstä ja ne voidaan esittää 360-asteisena panoraamakuvana verkkoselaimessa. Valokuvat mahdollistavat huoneiston tarkastelun virtuaalilaseilla, mutta eivät kuitenkaan vastaa täydellistä VR-sisältöä, jossa käyttäjä pystyy liikkumaan ja maailma on 3D-mallinnettu. Kuvien käyttö virtuaalilasien kanssa esittelyissä on tällä hetkellä suositumpaa, koska ne ovat yksinkertaisempia ja toimivat sulavammin mobiililaitteilla. [23.] Lisäksi asuntojen 3D-mallintaminen asuntonäyttöjä varten voi nykyisillä menetelmillä olla aikaa vievää tai liian kallista suhteutettuna sen lisäarvoon markkinoinnissa. Uudiskohteiden ennakkomarkkinoinnissa voi kuitenkin olla käytännöllistä käyttää virtuaalitodellisuutta talon esittelyssä. [24.] Rakenteilla olevista kohteista tehdään nykyisin usein 3D-malli talon visualisoinniksi, eikä sen liittäminen pelimoottoriin ole vaikeampaa kuin minkään muunkaan mallin.

Virtuaalitodellisuuden ja laitteiden yleistyessä asuntonäytöistä tulee entistä helpompia ja vaivattomampia. Ostajan ei tarvitse saapua kohteeseen tutustuakseen siihen kokonaisvaltaisesti, ja saman istunnon aikana voi tutustua moneen asuntoon samanaikaisesti. Ajallinen hyöty näkyy selvästi tilanteissa, joissa kohde sijaitsee maantieteellisesti kaukana ostajasta. Virtuaalitodellisuus ei kuitenkaan sivuuta perinteistä asunnon ostamista Suomessa vielä hetkeen. Vaikka kokonaiskuvan asunnosta voi saada virtuaalilaseilla, niillä on vaikea havaita virheitä. Suomen lain mukaan ostajalla on velvollisuus tarkastaa ennakkoon ostettava kohde ennen kaupantekoa. Mahdollisiin virheisiin, jotka tarkastuksen aikana olisi tullut voida havaita, ei voi vedota jälkeenpäin kaupan purkamiseksi. Näin ollen virtuaalinäytöt ja muutkin markkinointimateriaalit eivät toimi yksittäisinä perusteina asunnon ostoon. [24.]

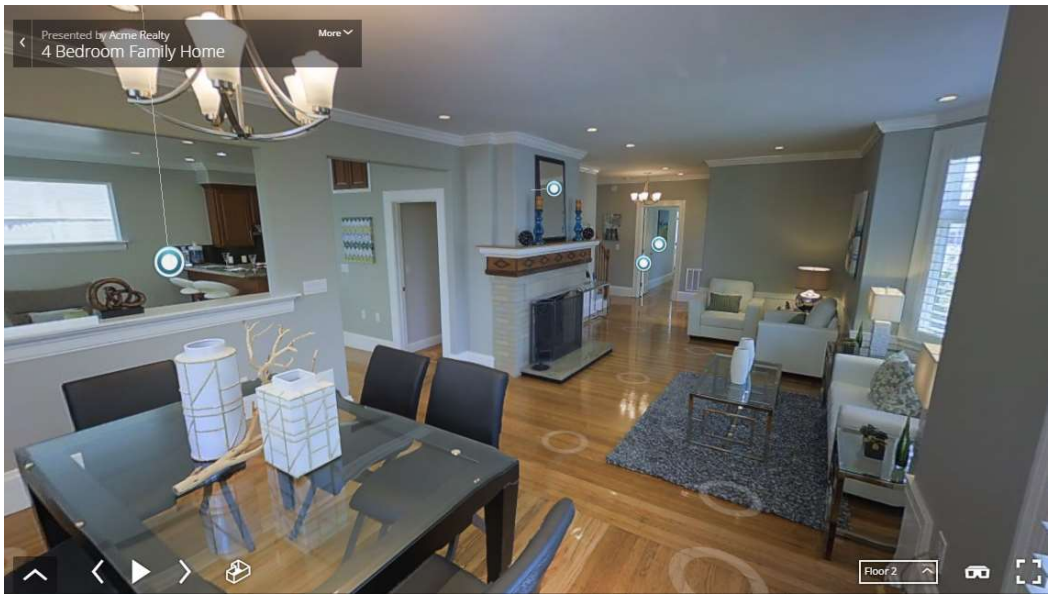
4.2 Asuntojen virtuaaliesittelyratkaisut Suomessa

Suomessa virtuaalitodellisuutta hyödyntäviä kiinteistönvälitysyrityksiä on vielä toistaiseksi melko vähän. Yrityksiä, jotka tuottavat virtuaaliesittelysisältöä kohteistaan, on muun muassa Aktia Kiinteistönvälitys, Aurea LKV, Sievitalo ja SRV. Aktia Kiinteistönvälitys kuvaa asuntoja 360-asteisilla kameroilla, ja esittelyjä voidaan katsoa asunnonmyyntiin perustuvien medioiden kautta tai yrityksen omilta sivuilta. Palvelu on suunni-

teltu yhdessä Realest3D:n kanssa, joka on asuntomarkkinoinnin visuaalisten toteutuksien tuottajaksi erikoistunut yritys. [25.]

Realest3D tarjoaa palveluina asuntojen kuvaamista, virtuaaliesittelyjä olemassa olevista kohteista, verkkosivuja ja 3D-malleja rakentamattomista kohteista. 3D-malleja voidaan tarkastella ilman liitännäisiä verkkoselaimessa sovelluksen kautta, joka sisältää tuen VR-lasien käyttöön mobiililaitteilla. Sovelluksen pohjana on Archilogic-ohjelmisto, joka on suunniteltu talomallien muuttamiseen 3D-virtuaaliesittelyihin soveltuviksi. [26.]

Kuvassa 6 on esitetty kuvakaappaus Realest3D:n käyttämästä Matterport-esittelyalustasta. Tila on rakennettu ottamalla kohteesta 360-asteisia panoraamakuvia. Lattialla olevia kehiä klikatessa kamera siirtyy niiden luo. Sovellus lataa tällöin uuden kuvan, joka on otettu kyseisestä kuvakulmasta. Näkymää voidaan haluttaessa tarkastella myös älypuhelimille tarkoitetuilla virtuaalitodellisuuslaseilla. [27.]



Kuva 6. Matterportin käyttöliittymä [27].

Matterport on erikoistunut palveluna tilojen 3D-kuvaamiseen. Se tarjoaa pilvipalvelua toteutusten esittämiseen ja omaa 3D-kuvaamiseen kehitettyä kameraansa. Kameraa käytetään mobiilisovelluksen kautta, eikä se välttämättä vaadi vankkaa kokemusta kuvaamisesta tai ohjelmistojen käytöstä. Valmiit kuvat voidaan siirtää sellaisenaan pilvipalveluun, joka skannaa ne ja järjestää ne hahmottamaan kohteen. Kohdetta voi tämän

jälkeen tarkastella palvelun kautta esittelynäkymässä. Aktia Kiinteistönvälityksen lisäksi Aurea LKV käyttää Matterport-palvelua kohteiden kuvaamiseen. [27.]

5 WebGL-sovelluksen ja 3D-mallintamisen suunnittelu

5.1 Sovelluksen ominaisuuksien ja tavoitteiden määrittely

Insinöörityönä tehdään mobiilioptimoitu talon esittelysovellus, johon sisältyy eri asetuksia ja ominaisuuksia. Mobiilioptimointi toteutetaan seuraamalla sen perusperiaatteita ja valitun apukirjaston kehittäjien antamia suosituksia mobiilisovelluksille. Tavoitteen saavuttamiseksi sovellusta testataan kehitystyön aikana jatkuvasti ja suorituskykyä vertaillaan eri päätelaitteiden välillä. Testauksessa käytetään mobiililaitteiden lisäksi pöytätietokonetta, joka toimii testeissä objektiivisena vertailupisteenä. Pöytätietokoneen tulokset eivät ole suoraan vertailukelpoisia mobiililaitteiden tulosten kanssa, sillä tietokoneen suorituskyky prosessoida 3D-grafiikkaa on huomattavasti tehokkaampi mobiililaitteisiin verrattuna tehokkaampien komponenttien takia. Sen tulokset antavat kuitenkin huomionarvoista tietoa WebGL-sovelluksen suuntaa antavasta maksimaalisesta suorituskyvystä.

Sovelluksen sisältöä varten tehdään 3D-malli siihen soveltuvalla 3D-grafiikan mallinnusohjelmalla. Mallina toimii arkkitehdin yksityishenkilölle suunnittelema talomalli. Mallintamisessa on käytössä lähdemateriaalina arkkitehdin teettämät talon pohjapiirustukset, kaavoitus- ja erilaisia havainnekuvia sisustuksesta. Mallinnuksessa otetaan huomioon pohjapiirustuksen mittasuhteet siten, että malli noudattaisi mahdollisimman tarkasti talosuunnitelmaa. Tavoitteena ei kuitenkaan ole tehdä täysin realistista mallia suunnitellusta talosta, vaan työ noudattaa ensisijaisesti mobiilioptimoinnin perusteita. Tästä syystä ennalta annettujen lähdemateriaalien ohjeita voidaan muokata työhön soveltuvaksi. Niistä voidaan myös poiketa, jos ne eivät ole teknisesti mahdollisia toteuttaa tai ovat liian haastavia.

Talomallista on tehty ennestään 3D-malli ja Three.js-apukirjaston avulla rakennettu WebGL-toteutus, jotka toimivat lisämateriaalina sovelluksen suunnittelussa ja kehityksessä. Aiempi 3D-malli ja WebGL-toteutus eivät ole optimoituja suoraan mobiilipäätelaitteille, mikä tulee ottaa huomioon, kun toteutuksia vertaillaan toisiinsa.

3D-mallin vaihdon ja muokkaamisen WebGL-kehyksessä tulisi olla helppoa siten, että sovellusta voitaisiin käyttää runkona muihin virtuaaliesittely- tai virtuaalitodellisuusprojekteihin. Kehitettävät ominaisuudet palvelevat ensikädessä esittelysovelluksen tarpeita, ja ne pyritään pitämään mahdollisimman yksinkertaisina ja selkeinä. Koko projekti rakennetaan avoimen lähdekoodin ohjelmistoilla, kehyksillä, apukirjastoilla ja rojaltivapailla tekstuurimateriaaleilla. Materiaalien kehittäjät tulee mainita työssä merkityin säädöksin, paitsi jos lisenssit eivät sitä vaadi. Noudattaen edellä mainittuja sääntöjä työn hyödyntäminen tai soveltaminen kaupalliseen käyttöön on haluttaessa mahdollista.

Sovellus sisältää päänäkymän ja virtuaalitodellisuuslaseille käytettäväksi suunnitellun näkymän. Päänäkymässä 3D-mallia voidaan tarkastella käyttäen kosketusnäytöllisille mobiililaitteille ominaisia kosketusliikkeitä. Valikon ominaisuuksien avulla voidaan navigoida näkymien välillä, tarkastella talon huoneita tarkemmin ja muokata 3D-piirron asetuksia. Asetuksilla vaikutetaan sovelluksen suorituskykyyn eri vaativuusasteilla. Käyttäjällä on näin mahdollisuus asettaa piirron nopeuteen vaikuttavat tekijät omalle laitteelle sopiviksi ja parantaa käyttökokemusta. 3D-piirron asetukset mahdollistavat sovelluksen jouhevan toiminnan mahdollisimman laajalla laitevalikoimalla.

Virtuaalitodellisuusnäkymässä päästään observoimaan rakennuksen sisätiloja ensimmäisen persoonan kuvakulmasta ja hyödyntäen mobiililaitteen gyroskooppia ohjauksessa. Tila esitetään stereoskooppisen kameratekniikan avulla, mikä lisää 3D-vaikutelmaa sovellukseen.

5.2 PlayCanvas-grafiikkamoottori

Insinöörityössä käytetään sovelluksen pohjana PlayCanvas-kehystä. PlayCanvas on ilmainen avoimen lähdekoodin WebGL-grafiikkamoottori, jota voidaan käyttää erillisenä apukirjastona sovelluksissa tai vaihtoehtoisesti sovelluksen voi rakentaa kokonaisuudessaan PlayCanvaksen kehyksen päälle. Apukirjaston tueksi on kehitetty selaimessa toimiva editori, jolla voidaan kehittää pelejä, 3D-sovelluksia tai visuaalisia tuotteita. Editori muistuttaa käyttöliittymältään 3D-pelimoottorialustoja, kuten Unity 5:tä, mutta sen toiminnallisuudet rajoittuvat WebGL-sovelluskehittämiseen. Ominaisuuksiltaan editorissa voidaan luoda muun muassa WebGL-komponentteja, tuoda ulkoisia 3D-malleja käytettäväksi sovellukseen ja käyttää PlayCanvaksen pilvipalvelua palvelimena. [28.] Editori vaatii käyttäjän luomista järjestelmään, ja vaihtoehtoina on valita ilmainen tai kuu-

kausimaksullinen sopimus. Ilmaisen ja maksullisen palvelun suurimmat erot ovat työn tallennuksessa, tallennustilassa ja sovelluksen viemisessä ulos järjestelmästä. Ilmaisessa suunnitelmassa käyttäjä ei voi tallentaa yksityisiä projekteja ja muut käyttäjät voivat vapaasti käyttää ja tarkastella työtä. Maksullinen suunnitelma mahdollistaa yksityiset projektit, laajemmat tallennustilat ja projektin viemisen ulos järjestelmästä. [29.]

Valitsin PlayCanvaksen sovelluksen pohjaksi sen helposti omaksuttavan käyttöliittymän ja tarjottavien ominaisuuksien takia. PlayCanvaksen editorissa on paljon optimointiin perustuvia asetuksia, joiden koodaaminen itse olisi haastavaa ja aikaa vievää ilman kokemusta ja ymmärrystä kaikista eri WebGL:n optimointimahdollisuuksista. Vertailtaessa PlayCanvasta Three.js:ään, jota pidin toisena vaihtoehtona sovelluksen kehykseksi, optimointi on tehty helpommalla kaavalla. Three.js ei tarjoa yhtä kattavaa dokumentointia optimoinnista kuin PlayCanvas, ja niiden selvittely vaatisi laajempaa tutkimista. Sen editori on myös paljon pelkistetympi, mutta käytännössä sillä pitäisi pystyä saamaan aikaiseksi samalainen lopputulos kuin PlayCanvaksella.

Kolmantena vaihtoehtona oli Unity 5 -pelimoottori, jota voidaan pitää näistä kaikkein kehittyneimpänä järjestelmänä ominaisuuksiltaan. Se ei kuitenkaan tue VR:ää suoraan WebGL:ssä ja on todistetusti paljon raskaampi mobiililaitteille kuin PlayCanvas. Sen moottori ei kompressoi yhtä tehokkaasti sovelluksia WebGL:ssä, ja ne voivat olla jopa 21 kertaa raskaampia kuin PlayCanvaksen. Latausnopeus eri laitteilla on noin 43 kertaa nopeampi ja kuvataajuus neljä kertaa korkeampi kuin Unity 5:llä. [21.]

5.3 Mobiilikehittämisen haasteet

Mobiililaitteiden komponentit asettavat kehittäjille haasteita ja mahdollisuuksia. 3D-grafiikan tuottamisessa järjestelmäriippumattomasti tulee ottaa huomioon laitteiden eriävät suoritintehot, näyttötekniikka, tiedostoformaatit ja verkkoyhteydet. [30.] Mobiililaitteiden laitteistotehot ovat huomattavasti rajoittuneempia pöytätietokoneisiin verrattuna. Niiden kehitys on suosion kasvaessa kuitenkin suhteessa nopeampaa tällä hetkellä kuin pöytätietokoneiden. Suosion takia markkinoilla on valtava kirjo erinäköisiä, -tehoisia ja erilaista tekniikkaa sisältäviä laitteita, joille yhdenmukainen sovelluskehitys haastaa luovia suunnittelijoita ja ohjelmistokehittäjiä luomaan sovelluksia, jotka toimisivat mahdollisimman joustavasti laitteissa. [31.]

Keskus- ja grafiikkasuorittimilla on usein rajattu laskentateho ja muisti, mistä syystä monimutkaisten tai määrällisesti monen 3D-mallin piirtäminen näyttölaitteelle samanaikaisesti on kuormittavaa. Valojen ja varjostimien käyttö on myös yksi suurimmista syistä laskentatehon laskuun. Suorittimen kuormitus ilmenee usein sovelluksen kaatumisena, tekstuurien ja objektien hitaana latausnopeutena tai matalana kuvataajuutena. Rasituksen keventäminen mobiililaitteilla on kriittistä, jotta käyttäjäystävällisyys toteutuisi. [32.]

Insinöörityössä käsitellään suorituskykyongelmia erilaisilla optimointityökaluilla ja rasituksenhallinnalla. Mallinnettaessa tulee käyttää mahdollisimman vähän verteksejä ja polygoneja. Jokainen yksityiskohta mallissa vie laskenta-aikaa suorittimelta, joten turhia pisteitä ja tasoja tulisi välttää. Lisäksi samanaikaisten verteksien määrän näkymässä tulisi olla matala. Tarkkaa määrää ei ole, mutta 3D-mallit kuvanäkymässä eivät saisi huomattavasti laskea kuvataajuutta ja aiheuttaa viivettä. PlayCanvaksessa jokainen mallin instanssi, määritelty osa, jolle voidaan antaa esimerkiksi oma tekstuuri, tekee piirtokutsun suorittimelle. Piirtokutsujen määrä mobiililaitteissa saa olla suositusten mukaan enimmillään 100–200 välillä. [32; 33.]

Koska työ tehdään selainpohjaisesti ja optimoidaan mobiililaitteille toimivaksi, verkkoyhteydellä on olennaisesti vaikutusta sisällön lataamisen nopeuteen päätelaitteelle. Latausaikojen vertailu laitteiden välillä on haastavaa, koska niiden kaikkien tulisi olla kytkettynä samaan verkkoon ja datasiirtonopeuden ja verkkoselaimen tulisi olla samat. Sovelluksen valmistuttua siirretyn datan määrää vertaillaan ja pyritään optimoimaan mobiiliverkkosivuille sopiviin rajoihin.

5.4 Blender-mallinnusohjelma

Talojen ja rakennusten 3D-mallintamiseen on tarjolla monia käyttötarkoitukseen suunniteltuja ohjelmistoja. Koska insinöörityössä käytetään avoimen lähdekoodin työkaluja, valitsin mallintamista varten Blender-ohjelmiston. Blender on ilmainen 3D-mallinnusohjelma ja sisältää paljon erilaisia ominaisuuksia, kuten animoinnin, 3D-veistämisen ja fotorealistisen renderöinnin. Sitä voidaan soveltaa myös videoeditointiin ja pelikehitykseen. Koska Blender on kokonaisvaltainen 3D-ohjelmisto, sitä ei ole suunniteltu suoraan rakennusten mallintamiseen. 3D-mallintamista varten ohjelmisto sisältää kuitenkin kattavan valikoiman työkaluja, minkä takia rakennusten luomisen ei

tulisi tuottaa ongelmia. Blender tukee myös mallien ulos vientiä FBX- ja OBJ-formaatissa, jota vaaditaan mallien tuontiin PlayCanvaksessa. Näin ei tarvita kolmatta osapuolta formaattien kääntämiseen ohjelmien välillä. [34.]

Syinä Blenderin valintaan mallintamisessa oli myös mahdollisuus verteksien ja polygonien hallintaan helposti. Koska sovellus optimoidaan mobiililaitteille, tulee mallien sisältää mahdollisimman vähän verteksejä ja polygoneja. Blenderissä kappaleiden määrien hallinta ja liittäminen yhteen suuremmiksi instansseiksi on mahdollista. Lisäksi polygoneja voidaan yhdistää suuremmiksi kokonaisuuksiksi säilyttämällä kuitenkin sama ulkomuoto objektilla.

6 Sovelluksen rakentaminen

6.1 Talon 3D-mallintaminen

Talon mallintaminen aloitettiin Blenderissä lataamalla asunnon pohjapiirustukset järjestelmään display-valikon kautta skenessä. Blenderissä voidaan käyttää mallintaessa tukena taustakuvia, jotka esittävät mallia eri kuvakulmista. Tämä toiminto helpottaa mallin visualisoimasta ja mittasuhteiden noudattamista. Lisäksi pohjapiirustuksia ei tarvitse verrata malliin ulkoisen ohjelman kautta tai paperilta, mikä helpottaa työnkulkua huomattavasti. Talon pohjapiirustukset ladataan Blenderiin JPG- tai PNG-kuvaformaatissa, ja tämän jälkeen valitaan, mitä kuvakulmaa taustakuva vastaa. Pohjapiirustukset asetetaan näkymään vain, kun tila-avaruutta tarkastellaan pystysuorassa kulmassa mallin yläpuolelta. Näin taustalla oleva kuva ei häiritse työntekoa, kun mallia katsotaan eri kulmista.

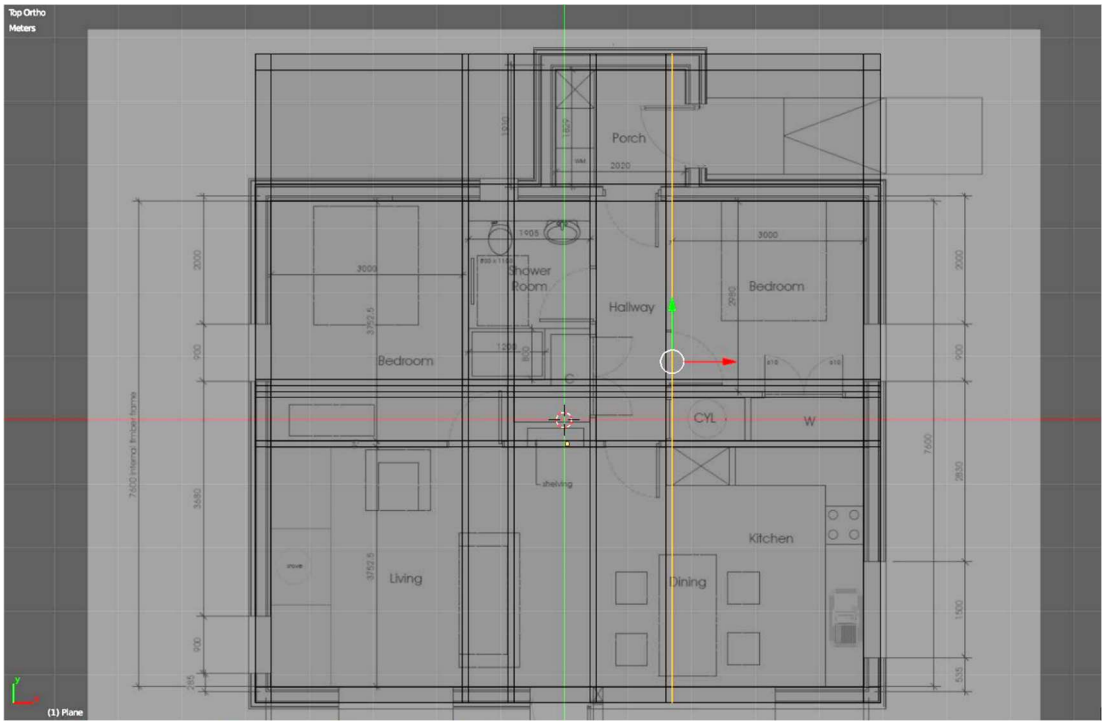
Talon mallintamiseen Blenderissä on monia keinoja, ja on mallintajasta itsestään kiinni, mitä työmenetelmiä haluaa käyttää ja nähdään toimivaksi ratkaisuksi. Työ aloitetaan neliönmallisesta tasosta, jonka sivut siirretään siten, että kulmat vastaavat talon kulmia. Koska talo ei ole täysin neliön muotoinen, yksi talon sivuista noudattaa laajennuksen uloimman sivun kaavaa. Taso vastaa talon seiniä ja sijoitetaan X-, Y-, Z-koordinaatistossa nollakohtaan. Seinätaso on kiinteä objekti, jossa sen verteksit ja sivut muodostavat kiinteän polygonitason mallille. Pohjapiirustus peittyy näin tason taakse, koska taustakuvataso on hierarkkisesti taaempana objekteista. Jotta mallinnettaessa pohjapiirustus voidaan nähdä seinätason takaa, käytetään mallintaessa näkymää, jos-

sa objektien kiinteiden tasojen pintoja ei piirretä. Näin saadaan läpinäkyvä malli ja taustakuva näkyväksi sen takaa.

Blender-ohjelmassa on kaksi eri muokkaustilaa. Object Mode -tilassa objekteja voidaan mitoittaa, kääntää ja skaalata yksin tai erikseen. Edit Mode -tilassa voidaan muokata objektikohtaisia verteksejä ja pintoja. Tasoon tehdään Edge Loop -työkalun avulla lisää sivuja, verteksejä ja tasoja objektiin. Jotta työkalua voidaan käyttää, tulee Blender-ohjelman olla Edit Mode -tilassa.

Edge Loop -työkalun avulla objektiin lisätään silmukoita, jotka kiertävät objektin ja muodostavat verteksikulmia aina sivun kohdatessaan. Koska polygonit muodostuvat vertekseistä, malliin syntyy aina uusia pintoja, kun silmukoita lisätään objektiin. Liittäminen tehdään Ctrl- ja R-näppäinyhdistelmällä, ja niiden sijainti voidaan määritellä hiirenkohdistimella vetämällä tai koordinaattiohjaimilla.

Silmukoilla määritellään tasoon pohjapiirustuksessa kuvattujen seinien rajat kuvan 7 mukaisesti. Jokaisen seinän kohdalle pohjapiirustuksessa muodostuu sitä vastaavat pinnat tasoon, ja niitä voidaan hyödyntää mallintamisen seuraavissa vaiheissa. Kuvassa 7 on esitetty Blender-ohjelma ja skene, johon malleja luodaan. Skenessä ovat esillä talomallin pohjapiirustukset taustakuvana. Sen päällä näkyy seinätaso läpinäkyvänä objektina hieman tummennetulla pinnalla. Mustat linjat kuvastavat tason sivuja ja silmukoita. Yksi silmukka on kuvassa valittuna muotoilua varten. Se näkyy keltaisena viivana, ja sen keskellä vihreä ja punainen nuoli esittävät X- ja Y-akseleita, joilla silmukkaa voidaan liikuttaa.



Kuva 7. Silmukoiden lisääminen objektiin Blender-mallinnusohjelmassa.

Silmukkalinjat muodostavat uusia pintamuotoja objektiin, ja niistä osa tulee poistaa. Tasoja halutaan sijaitsevan vain seinille tarkoitetuilla alueilla, jotta niistä voidaan jatkossa muodostaa talomallin seinät. Ylimääräisten tasojen poisto on Blender-ohjelmassa helppoa Delete-näppäintä painamalla ja valitsemalla haluttu poistomenetelmä. Kun ylimääräiset pinnat on poistettu, jäljelle jäävät vain seinien pintoja esittävät tasot objektissa.

Ylimääräiset pinnat tulee poistaa mallista mobiilioptimoinnin periaatteita noudattaen. Mallien tulisi sisältää mahdollisimman vähän tasoja ja verteksejä. Sen sijaan, että polygonit poistettaisiin kokonaan, ne yhdistetään suuremmiksi kokonaisuuksiksi mallissa. Tämä tehdään samalla tavalla kuin pintojen poistaminen, mutta poistonvalintanäkymässä valitaan sivujen hajottamistoiminto. Toiminto poistaa ylimääräisen sivun ja samalla siihen kytketyt verteksit, jotka eivät tuota kulmia muiden sivujen kanssa.

Kuvassa 8 näkyy ylimääräisten polygonien poistamisen vaikutus malliin. Kuvassa on liitetty skenestä kaksi kuvakaappausta, joista vasemmanpuoleinen on otettu ennen poistoa ja oikeanpuoleinen hajottamistoiminnon jälkeen. Hajottamistoiminnon jälki näkyy talon seiniä esittävien vaaleanharmaiden osien alueella, kun kuvia vertaa toisiinsa. Oikealla puolella on mustia linjoja, polygonien sivut, joita on karsittu huomattavasti ver-

rattuna alkuperäiseen. Lopputuloksena on yksinkertaisempi malli ja pienempi määrä polygoneja.

Kuva 8. Polygonien poiston vaikutuksen vertailu Blender-ohjelmassa.

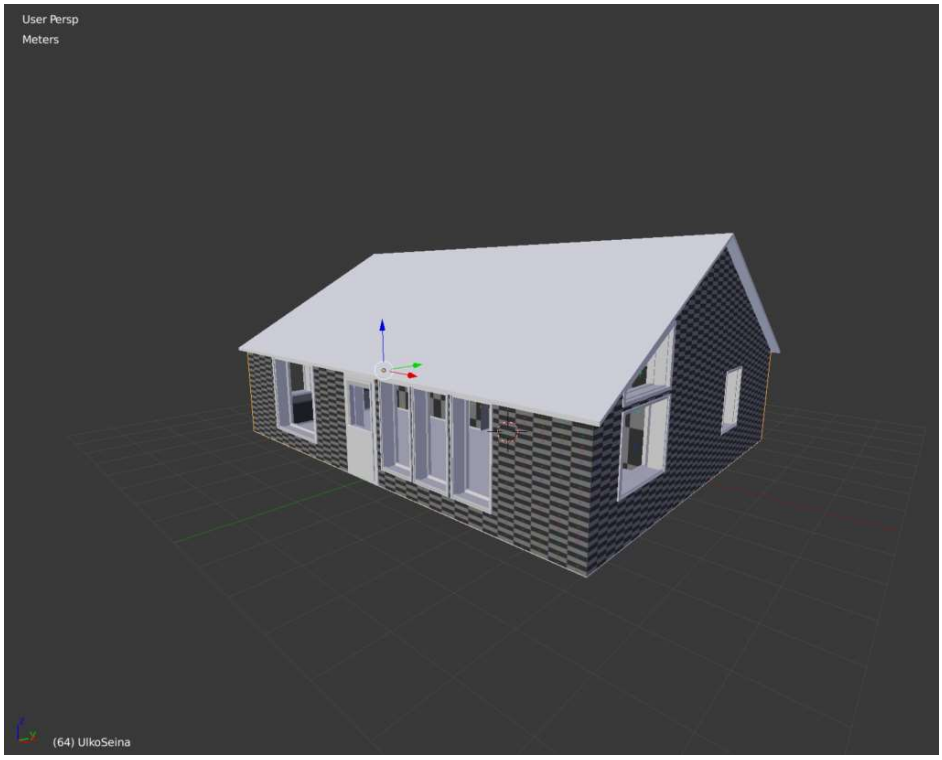
Kun talon seinien rajat on määritelty, voidaan mallille luoda seuraavaksi kiinteät seinät. Seinien mallintaminen tehdään Extrude-työkalulla, jolla niin sanotusti pursotetaan polygoneista niiden muotoja vastaavia ulokkeita. Pursotus on hyvin yleinen työkalu mallintamisessa, koska sillä voidaan laajentaa mallia pintojen avulla helposti. Esimerkiksi jos yksittäinen sivutaso pursotetaan, siitä muodostuu kuutionmallinen objekti. Seinien tapauksessa kaikki polygonit aktivoidaan ja pursotetaan Z-akselilla ylöspäin tila-avaruudessa muodostaen seinät. Korkeus määräytyy piirustuksien mukaisen korkeimman pisteen mukaan. Koska talolla on sahalaitainen katto, osa seinistä on liian korkeita. Nämä seinät voidaan leikata myöhemmässä vaiheessa, kun malliin lisätään katto.

Seiniin tulee lisätä aukot ikkunoille ja oville, mikä tehdään samalla kaavalla kuin polygonien poistaminen. Seiniin on tätä vaihetta varten jätetty linjat vastaamaan kohtia, joihin ikkunat ja ovet tulevat, mikä helpottaa työn vaihetta huomattavasti. Aukkojen lisäämisen jälkeen seinät ovat valmiit ja samalla koko mallin runko, johon suhteutettuna muut objektit rakennetaan.

Talo koostuu useasta kappaleesta, jotka ovat lattia, seinät, katto, listat, ovet ja ikkunat. Näistä seinät ovat useamman instanssin muodostama kokonaisuus. Koska seinille halutaan antaa eri tekstuurit, tulee niiden olla tässä tapauksessa myös eri instansseina.

Seinät on jaettu ulkoseinäksi, sisäseinäksi ja katoksi. Kattoja mallissa on kaksi. Toinen esittää ulkokattoa ja toinen katon sisäpintaa.

Kuvassa 9 näkyy talomalli, kun siihen on lisätty seinien lisäksi puuttuvat kappaleet. Seinillä on ruutumallinen tekstuuri, jonka avulla voidaan havainnollistaa tekstuurin laskeutumista pinnoille ja havaita väärystymiä. Mallista puuttuvat vielä huonekalut, jotka lisätään myöhemmin.



Kuva 9. Valmis talomalli Blender-ohjelmassa.

Talomallin katto muodostuu pääkatosta, eteisen ulokkeen lisäkatosta ja talon ylemmän tason seinistä. Seinillä oleva mallitekstuuri asetetaan käyttämällä UV Unwrap -työkalua, jolla objektien pinnat avataan 2D-tasolle, jossa tekstuuri asetetaan mallille. UV-kirjainyhdistelmä tulee 2D-tason koordinaattiakseleista, jotka on nimetty kirjaimien mukaan. Jos tekstuuri ei asetu tasaisesti, ruutumallin mukaisesti, siinä voi ilmetä vääristymiä renderöitäessä. [35.] Työssä tekstuurit lisätään vasta PlayCanvaksen editorissa, jossa ne yhdistetään materiaaleihin.

Talon sisustukseen otettiin mallia pohjapiirustuksen suunnitelmasta ja aiemmin tehdystä 3D-mallista. Apuna mallintamiseen käytettiin TurboSquid-palvelun tarjoamia rojalti-

vapaita 3D-malleja. Malleja on muokattu sopiviksi työhön tai käytetty esikuvina osaan huonekaluista. Suoraan käytettyjä ja muokattuja huonekaluja työssä ovat sohva, wc-istuin ja pöytä. Objektien polygonien määrä on pidetty pienenä, jotta polygonien vaikutus kuvataajuuteen ja tiedoston kokoon olisi mahdollisimman alhainen.

Lopullisessa talon 3D-mallissa on yhteensä 23 objektia, 9 118 sivua ja 19 838 kolmionmallista polygonia. 3D-malli sisältää talon, sisustuksen ja lähiympäristön. Ympäristö on yksinkertainen mallinnus talon tonttialueesta ja koostuu maastosta, pihasta ja tiestä. Tiedoston koko Blenderin blend-tiedostoformaatissa on yhteensä 2,21 Mt.

6.2 Mallin tuonti pelimoottoriin

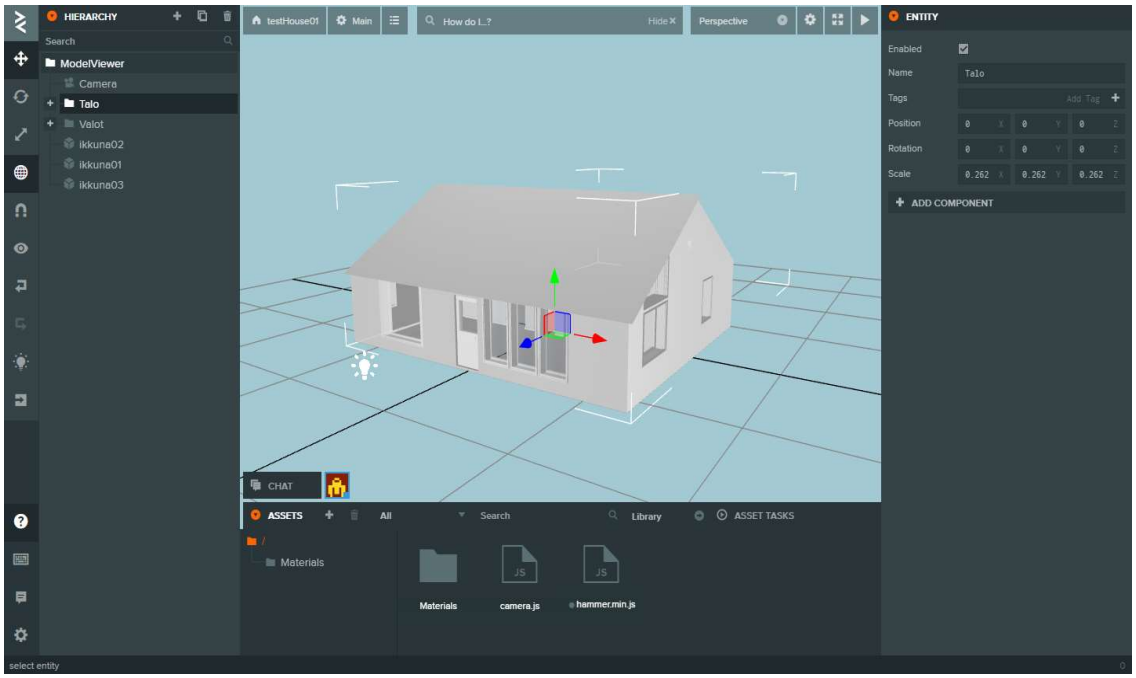
Talomalli, pois lukien katto, tuotiin insinöörityössä PlayCanvas-kehykseen yhtenä kokonaisuutena. PlayCanvas osaa lukea kappaleet itsenäisinä malleina, vaikka ne tuotaisiin yhtenä tiedostona pelimoottoriin. Tämä nopeuttaa prosessia siten, ettei kappaleita tarvitse tallentaa yksitellen. Tiedoston kokoon objektien pakkaaminen ei vaikuta huomattavalla tavalla, koska kokonaisuudessaan se sisältää saman datan kuin objektien tallentaminen eri tiedostoihin. Kaikki halutut objektit valitaan Blender-ohjelmassa, ja tallennetaan FBX (Filmbox) -formaattiin, joka on kehitetty yhteensopivaksi formaatiksi mallinnusohjelmien väliseen tiedostojen jakoon.

Ennen kuin malli voidaan tuoda PlayCanvakseen, siinä tulee luoda uusi projekti. Projektia muokataan verkkoselaimessa käynnistettävän editorin avulla. Editori muodostuu lähdemateriaalien kansionäkymästä, 3D-mallien skenestä, työkaluvalikosta ja materiaalien tarkasteluvalikosta. Editori voi sisältää monta eri skeneä, ja ne voidaan julkaista sovelluksina. Sovellus tallennetaan automaattisesti pilvipalvelimelle ja siihen päästään käsiksi generoidun linkin kautta. Ennen julkaisua sovellusta voidaan tarkastella testausnäkymässä.

FBX-tiedosto 3D-malleista voidaan avata PlayCanvas-editorissa raahaamalla se editorin lähdemateriaalien kansioon. PlayCanvas muuntaa FBX-tiedoston automaattisesti JSON-formaattiin, jota voidaan käyttää skenessä. JSON-malli taas voidaan raahata skeneen, jossa se tuottaa siitä automaattisesti entiteetin. JSON-lyhenne tulee sanoista JavaScript Object Notation, ja se on tiedonvälitykseen kehitetty tiedostomuoto. Entiteetteihin voidaan kytkeä erinäisiä komponentteja, ja se toimii niiden yksilöivänä kehyk-

senä. Komponentit voivat olla valmiita PlayCanvas-kehykseen kehitettyjä ominaisuuksia tai ohjelmakoodeja.

Kuvassa 10 on talomallin ensimmäinen versio tuotuna editoriin. Mallin objekteille ei ole vielä määrätty tekstuureita, joten oletusvärinä on valkoinen. Taustalle on määrätty sininen väri ja skeneen lisätty kaksi valokomponenttia, jotta objektin pinnat ja muodot erottuisivat. Kuvassa objektit on tuotu erillisinä tiedostoina, mikä sittemmin on korvattu talon tuomisella järjestelmään yhtenä kokonaisuutena. Vasemmalla näkyy entiteettien hierarkia ja kansiorakenne. Skenen alla on lähdemateriaalien kansio ja oikealla materiaalien tarkasteluvalikko.



Kuva 10. Talomalli tuotuna PlayCanvas-editoriin.

Talomallin onnistuneen latauksen jälkeen malli tuotiin järjestelmään kokonaisuudessaan sisustuksen ja ympäristön kanssa. Lopullisten FBX-tiedostojen koko insinöörityössä oli 362 kilotavua. PlayCanvaksen muuntaessa ne JSON-formaattiin tiedostojen koko kasvoi 1,64 megatavuun. Mallin lisäksi editoriin ladattiin kolme rojaltivapaata tekstuuria, joiden tiedostokoko oli yhteensä 687 kilotavua.

6.3 Interaktiivisuus ja valikko

Sovellukseen suunnitellut toiminnot toteutettiin PlayCanvaksen tarjoamien kehittäjätyökalujen, apukirjastojen, koulutusmateriaalien ja keskustelupalstalta löytyneiden ratkaisujen avulla. Sovellukseen suunnitellun päänäkymän rakentaminen oli vaivatonta, sillä PlayCanvas tarjoaa valmiita alustoja, joista aloittaa projektin teko. Käytetty alusta sisältää kamerakomponentin ja ohjainkoodin, joita käytettiin päänäkymässä. Kameraa voidaan liikuttaa kosketusnäytöllä kiertoradalla, jonka keskiössä 3D-malli on, ja tarkentaa siirtyen kameralla lähemmäs tai kauemmas mallista. Ohjauskoodi täytyy toimiakseen olla liitettynä samaan entiteettiin kuin kamerakomponentti.

VR-näkymää varten hyödynnettiin dokumentoitua PlayCanvaksen demoprojektia, joka sisältää VR-kameran komponenttikoodit ja virtuaalitodellisuus sovelluksille suunnitellut ohjainkoodit. VR-kameralle on kehitetty JavaScript-apukirjasto, joka jakaa tietoa laitteen rotaatiosta ja mahdollistaa tuen gyroskoopin käytölle. Apukirjastoa ei tarvitse liittää erilliseen komponenttiin. Koodin tukena toimii stereoskooppista kameraa varten kehitetty koodi, joka liitettynä kameraan luo kahtia jaetun kuvan sisältönäkymään. [36.]

PlayCanvaksen demoprojektissa ohjainkoodien toiminta on toteutettu säteenjäljityksellä. Säteenjäljityksessä ammutaan näkymätön säde halutusta pisteestä määriteltyyn suuntaan. Säde havaitsee kohteita ympäristössä, kun ne osuvat sen piirtämään linjaan. Tekniikkaa käytetään yleisesti peleissä esimerkiksi luotien liikeradan laskemiseen ja niiden osumien havaitsemiseen. [37.] Demossa voidaan kohdentaa säde objekteihin, jotka aktivoituvat ja esittävät erillisen grafiikan kohdistuksen aikana. Toiminto on samankaltainen kuin raportissa edellä mainitussa SECVRITY-VR-pelissä. Demon ohjainkoodien avulla ei kuitenkaan pystytä alkuperäisessä muodossa liikkumaan 3D-ympäristössä, joten niitä tulee muokata projektiin soveltuviksi.

Kameralle ja objekteille on suunniteltu omat koodit säteenjäljitystä varten. Kameran koodi liitetään kamera-entiteettiin, jolloin se toimii säteen lähettäjänä. Säteen on määritelty lähtevän eteenpäin kohtisuorassa suunnassa entiteetin olinpaikasta. Kun kamera on liitettynä entiteettiin, lähetettävä säde kohdistuu kuvanäkymän keskipisteeseen. Kun koodi on liitettynä entiteettiin, kamera asetetaan skenessä talon sisälle simuloimaan henkilön näkökenttää.

Säteen vastaanottava koodi tulee liittää siihen entiteettiin, jonka halutaan olla vuorovaikutuksessa säteen kanssa. Sovelluksessa koodi liitettiin talon sisällä olevaan liikkumispistettä indikoivaan objektiin. Kameran lähettämä säde voi siten tunnistaa objektin ja suorittaa halutun funktion.

Lähdekoodia muokattiin soveltumaan työhön siten, että se tunnistaa kaikki liikkumispisteet määriteltyjen entiteettinimien perusteella. Nimeämisen avulla voidaan eritellä kaikki objektit ja antaa niille omat funktionsa. Vuorovaikutuksen syntyessä, kun lähetetty säde kohdistuu objekteihin, kamera liikkuu määritetylle pisteelle X-, Y- ja Z-akseleilla. Kameran liike toteutetaan Lerp-toiminnolla, joka esittää siirtymän liikkeenä pisteiden välillä. [38.] Taloon asetettiin liikkumispisteet neljään eri huoneeseen, joihin käyttäjä voi liikkua. Liikkumispisteitä symboloivat pyöreät mallit, joihin näytön keskiöön asetettu tähtäin voidaan kohdistaa. Objekteihin liitettiin lisäksi yksinkertainen koodi, joka animoi mallit pyörimään akseliensa ympäri.

Esimerkkikoodissa 1 on esitetty lyhyt ote liikkumispisteisiin liitetystä koodista, johon on eritelty yhden liikkumispisteen toteutus. Riviltä 4 alkaen etsitään sovelluksesta entiteetti nimeltä W01 ja luodaan sille rajattu kehä, jonka nimi lähetetään eteenpäin säteen tuottavalle koodille, jotta se tunnistaisi sen. Kun säde kohdistuu liikkumispisteen kehään, laukaisee koodi onHover01-funktion, joka toteuttaa liikkeen kehän luo. Liike toteutetaan funktiossa kutsumalla rivin 11 kameraentiteettiä ja kuljettamalla se rivillä 12 määriteltyyn pisteeseen lerp.js-koodin avulla. Funktion sisällä hallinnoidaan myös muita funktioita asettamalla niitä päälle tai toimintakyvyttömiksi. Näin voidaan estää kameran liikkuminen talon seinien läpi ei-toivottuihin paikkoihin.

```
***selectable.js
var Selectable = pc.createScript('selectable');
Selectable.prototype.initialize = function() {
   this.w01 = this.app.root.findByName ('W01');
   this._sphere01=
   new pc.BoundingSphere(this.w01.getPosition().clone(), 0.10);
   this.app.fire("selectorcamera:add",this.w01, this._sphere01);
   this.w01.on("selectorcamera:hover", this.onHover01, this);
   this.entity.on("selectorcamera:selectionprogress",this
.onSelectionProgress, this);
   this.cameraEntity = this.app.root.findByName ('CameraVR');
```

```
   this.position01 = new pc.Vec3 (0.02, 1.4, 3.3);
};
Selectable.prototype.onHover01 = function (dt) {
    this.w02.on("selectorcamera:hover", this.onHover02, this);
    this.w03.on("selectorcamera:hover", this.onHover03, this);
    this.w04.off("selectorcamera:hover", this.onHover04, this);

this.cameraEntity.script.lerp.targetPosition=this.position01;};
```

Esimerkkikoodi 1. Ote liikkumispisteiden ohjelmoinnista.

Näkymien väliseen navigoimiseen ja lisäominaisuuksien käyttöä varten kehitettiin päänäkymään graafinen valikko. Valikko sisältää painikkeita, jotka on tehty sprite.js-apukirjaston avulla. Entiteettiin, joka sisältää sprite.js-koodin, voidaan lisätä 2D-grafiikka PlayCanvaksen editorin kautta. Painikkeiden grafiikka on Googlen avoimesta materiaalikirjastosta ladattuja kuvia, jotka muokattiin Paint.NET-kuvankäsittelyohjelmalla. PlayCanvaksen dokumentointi tarjoaa painikkeiden käytöstä myös käytännön esimerkin, jonka pohjalle valikon toiminta perustuu. [39.]

Valikon koodi liitettiin entiteettien juurikansioon, jotta se vaikuttaa kaikkiin painikkeisiin samanaikaisesti. Koodissa haetaan esimerkkikoodi 1:n tavoin kaikki painikkeet entiteettinimien mukaisesti ja määritellään klikin käsittelyfunktiot. Funktioiden toiminnallisuudet tehtiin todella yksinkertaiseen muotoon. Käytännössä aina, kun painikkeita painaa, se joko aktivoi entiteettejä tai tekee niistä toimintakyvyttömiä. Valojen ja kattotason asetuspainikkeita klikattaessa aktivoidaan aina uusi painike ja kuva.

6.4 Sovelluksen suorituskyvyn optimointi

Sovelluksen suorituskyvyn optimointi mobiililaitteille perustuu graafisen sisällön ja tiedostokokojen hallintaan. 3D-mallintamisessa otettiin huomioon mallin tiedostokoko käyttämällä mahdollisimman vähän polygoneja ja verteksejä. Lisäksi objektien määrä skenessä pidettiin mahdollisimman pienenä. PlayCanvaksen puolella suorituskykyyn vaikutettiin tekstuurien ja valokomponenttien avulla.

Tekstuureja työssä haluttiin käyttää mahdollisimman vähän, koska niiden lataaminen ja renderöinti vie laskenta-aikaa laitteelta. Tätä varten Blender-mallinnusohjelmassa liitet-

tiin objekteja toisiinsa muodostamaan suurempia objektikokonaisuuksia. Näin voidaan ladata kahden objektin ja tekstuurin sijaan yksi objektiryhmä, jossa kaikilla on sama tekstuuri. Sovelluksessa on yhteensä 25 objektia, joilla on omat materiaalit, mutta vain kolmella niistä on tekstuurit. Loput objekteista käyttävät materiaaleja, joihin on asetettu värejä ja pintaefektejä. Tekstuureja käytetään lattian lisäksi sohvasta ja tuoleista koostuvassa tekstiilipintojen objektiryhmässä. Kokonaisuudessaan ulkoisten tekstuurien tiedostokoko on 687 kilotavua.

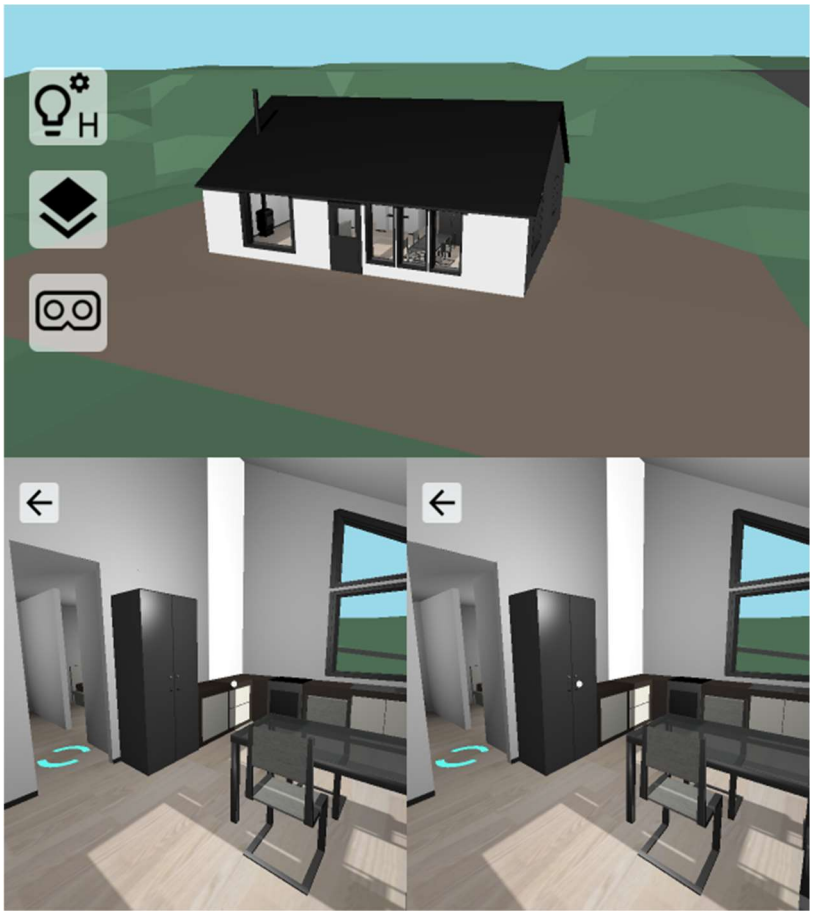
Teksturoinnin lisäksi työssä käytettiin valon ja varjojen kartoitustyökalua, jonka avulla sovelluksen renderöimiseen kuluu entistä vähemmän aikaa. Kartoitustyökalulla valosta ja varjoista tehdään tekstuurikartta, joka asetetaan pintamateriaalien päälle. Tämän toiminnon avulla sovelluksen ei tarvitse laskea valon vaikutusta materiaaleihin reaaliaikaisesti. Valon tekstuurikarttaa on hyvä käyttää silloin, kun skenessä on liikkumattomia objekteja, koska niiden luomat varjot ja vaikuttavat valot ovat tällöin staattisia. Valon laskeutumista materiaaleille on turhaa antaa sovelluksen laskea reaaliaikaisesti, jos skenessä ei ole dynaamisia objekteja. [40.] Valon kartoitustyökalua käytetään siten, että objektien asetuksissa aktivoidaan valokartoitusominaisuus. Tämän jälkeen valokomponentin asetuksissa asetetaan valo vaikuttamaan vain niihin materiaaleihin, jotka on valokartoitettu. Tällöin valon laskeutumisen teksturointiin vaadittava laskenta tehdään vain yhden kerran sovelluksen käynnistyessä päivittämättä sitä kertaakaan.

Mobiililaitteen suorituskyvyn rajat ja sovelluksen vaadittavat laskentatehot vaikuttavat grafiikan kuvataajuuteen. Kuvataajuudella tarkoitetaan sitä, kuinka monta kertaa sekunnissa näytön esittämä kuva päivittyy. Mitä enemmän sisältönäkymässä esiintyy objekteja, valovarjostimia ja muita ominaisuuksia, sitä raskaampi se on renderöidä laitteille. Jos kuvataajuus laskee alle 30 kuvaan sekunnissa, voi kuvatiheys alkaa vaikuttaa tökkivältä. PlayCanvaksessa optimi kuvataajuus on asetettu 60 kuvaan sekunnissa, jolloin kuvanpäivitys on jouhevaa. [33.]

Valokomponentit ovat yksi suurimmista kuvataajuuden tekijöistä 3D-mallien koon ja määrän lisäksi. Sovelluksessa käytetään kahdeksaa eri valokomponenttia, joita hyödynnetään valikon valoasetuksissa. Asetuksiin asetetaan kolme eri valaistustasoa, mikä säätelee sitä, kuinka monta valokomponenttia skenessä on päällä. Ensimmäinen on vaadittavalta laskentateholta korkein taso, jossa käytetään yhtä kohtisuoraa valoa ja viittä pistevaloa, jotka esittävät lamppujen tuottamaa valoa 3D-mallin sisällä. Toinen on keskitaso, jolloin on yksi kohtisuora valo aktiivisena. Kolmannessa tasossa skenessä

on yksi pistevalo päällä. Pistevalot ovat suotuisampia käyttää kuin kohtisuorat valot, koska niiden tuottamat varjot ovat kevyempiä tuottaa. Valaistustasojen avulla käyttäjä voi tarvittaessa laskea sovelluksen vaatimia tehoja ja korottaa kuvataajuutta.

Kuvassa 11 on esitetty kaksi kuvakaappausta valmiin sovelluksen molemmista näkymistä mobiililaitteella. Pöytätietokoneella käytettäessä VR-näkymä ei renderöi stereoskooppista kameraa, vaan ensimmäisen persoonan kameran samoilla toiminnoilla. Yläpuolella olevassa päänäkymässä on esillä 3D-mallinnus ja valikon painikkeet vasemmalla sivulla. Painikkeet esittävät valoasetuksia, kattotason pois sulkemista ja VR-näkymään siirtymistä. VR-näkymässä on esitettynä stereoskooppinen kamera toiminnassa 3D-mallinnuksen keittiön sisustus. Vasemmalla on näkyvillä sininen liikkumispiste, jonka luo voidaan liikkua kohdistamalla kuvassa esiintyvän tähtäin sen päälle. Stereoskooppisuuden takia tähtäimet eivät osoita samaan pisteeseen, mutta käytettäessä VR-laseja silmät tuottavat kuvista yhden kokonaisuuden, jossa piste sijaitsee keskellä kuvaa. Kokonaisuudessaan suunnitteluun ja toteutukseen kului noin 130 tuntia.



Kuva 11. Sovelluksen molemmat näkymät.

6.5 Optimoinnin testaus ja lopputulosten analyysi

Valmiin sovelluksen optimointia testattiin kahdella eri menetelmällä, jotka ovat laitevertailu ja verkkosivuston latausaika. Testauksella pyrittiin selvittämään, onko WebGL-sovellus rakennettu soveltuvaksi käyttää mobiililaitteilla, kuten oli suunniteltu. Johtopäätökset tehtiin analysoimalla saatuja lopputuloksia ja arvioimalla yleistä toimivuutta käytännön kokeiluilla.

Laitevertailuun otettiin mukaan neljä mobiililaitetta, tabletti ja pöytätietokone. Taulukossa 2 on esitetty kaikkien laitteiden tulokset eri kategorioissa. Laitteiden suorituskykyä vertaillaan keskiarvokuvataajuustuloksilla eri valoasetuksilla pää- ja VR-näkymässä. Kuvataajuudet on esitetty FPS-, Frames Per Second, yksikkönä, joka ilmaisee kuvien piirtomäärää sekunnissa. Piirtojen määrää sekunnissa voidaan käyttää esitettäessä laitteiston laskentatehon nopeutta ja sitä, kuinka nopeasti se pystyy renderöimään kuvia. Keskiarvo-FPS laskettiin sitä indikoivalla JavaScript-apukirjastolla, joka laskee sisältönäkymän kuvataajuuden ja ilmaisee sen grafiikkana näkymässä.

Taulukko 2. Laitevertailun tulokset.

LAITEMALLI	PÄÄNÄKYMÄ KORKEA	PÄÄNÄKYMÄ KESKITASO	PÄÄNÄKYMÄ MATALA	VR-NÄKYMÄ KORKEA	VR-NÄKYMÄ KESKITASO	VR-NÄKYMÄ MATALA
iPhone 6S Apple	60 FPS	60 FPS	60 FPS	60 FPS	60 FPS	60 FPS
Xperia M4 Aqua Sony	32 FPS	38 FPS	55 FPS	10 FPS	16 FPS	33 FPS
Galaxy S5 Neo Samsung	19 FPS	22 FPS	32 FPS	8 FPS	16 FPS	29 FPS
Lumia 640 Microsoft	-	-	-	-	-	-
Tabletti (Transformer Pad K010)	60 FPS	60 FPS	60 FPS	20 FPS	30 FPS	45 FPS
Pöytätietokone (AMD Radeon HD 7700, 8GB, Quad-Core Processor 3.60 GHz)	60 FPS	60 FPS	60 FPS	60 FPS	60 FPS	60 FPS

Tulosten perusteella sovelluksen toimivuus on selvästi laitteistoriippuvainen. IPhone 6S sai kaikista mobiililaitteista korkeimmat tulokset ja ylsi pöytätietokoneen suorituskyvyn

tasolle. IPhone 6S on myös kaikista testatuista mobiililaitteista tehokkain laitteistoltaan, mistä syystä korkein tulos testissä ei ole yllättävä. [41.] Microsoftin Lumia 640 -laitteella WebGL-sovellus ei toiminut halutulla tavalla ja tuloksia ei voitu julkaista. Laite ei kyennyt esittämään 3D-malleja sisältönäkymässä. Puhelimista vain iPhone 6S ja Galaxy S5 Neo sisälsivät gyroskoopin, minkä takia VR-näkymän käyttömukavuus muilla mobiililaitteilla ei ole paras mahdollinen. Pöytätietokoneella, VR-näkymää käytettäessä, sovellus ei jaa sisältönäkymää stereoskooppiseksi.

Laitevertailun tuloksista käy ilmi, että VR-näkymä on selvästi raskaampi prosessoida kuin päänäkymä. Syynä on kahden kameran tuottama kuva sisältönäkymään, mikä kuormittaa enemmän kuin yhden kameran generointi. Talon 3D-mallin sisällä on enemmän objekteja ja yksityiskohtia piirtää näytölle kuin ulkopuolella, mikä vaikuttaa laskentaan. [32.] Päänäkymässä on esillä vähemmän objekteja, koska ne ovat piilossa talon sisällä. Testauksessa selvisi myös, että mitä lähemmäs malleja kameran kohdensi päänäkymässä, sitä enemmän kuvataajuus mobiililaitteilla laski. Taajuus taas kasvoi, kun malleja tarkasteltiin kauempaa.

Valoasetusten vaikutuksen suhde kuvataajuuteen ja suorituskykyyn varmistettiin testin tuloksilla. Matalimmalla tasolla, jolloin skenessä on vain yksi pistevalokomponentti, kuvan päivitys oli kaikista nopeinta. Päänäkymässä FPS kasvoi matalalla valoasetuksella noin 70 % suhteessa korkean tason kuvataajuuteen. VR-näkymässä sama lukema oli noin 245 %. Keskitason valoasetukset olivat tuloksissaan hieman lähempänä korkean tason kuvataajuutta kuin matalan. Keskitason kuvataajuus ei näin vastaa todellista keskiarvoa valoasetusten välillä.

Mobiilisivustojen käyttäjäystävällisyyden yhtenä kriteerinä on verkkosivuston latausnopeus. Nopeuteen vaikuttaa sivuston koko, ladattavan aineiston määrä ja verkon tiedonsiirtonopeus. Mobiiliverkkojen nopeudet eivät ole aina stabiileja, mistä syystä mobiilisivustojen tulisi olla melko kevyitä ladata. Taulukossa 3 WebGL-sovelluksen latausnopeutta vertaillaan kahdeksan muun mobiilisivuston kanssa. Sovelluksen tulokset on esitetty nimetyllä sivulla ja vihreällä pohjalla. Muut sivustot kuuluvat tavoittavuuden mukaan suosituimpiin suomalaisiin verkkosivuihin. [42.] Taulukossa 3 on esitettynä verkkosivustojen latausaika sekunteina ja tarvittava tiedonsiirron määrä. Testissä käytettiin 3G-mobiiliverkkoa, jonka latausnopeus on 750 kt/s, siirtonopeus 250 kt/s ja 100 millisekunnin latenssi, ja Google Chrome -selainta. Testissä ei käytetty välimuistia hyödyksi sivuston latauksessa.

Taulukko 3. Sivustojen latausajan vertailu.

SIVUSTO	LATAUSAIKA	TIEDONSIIRTO
sivusto 1	9,89 s	0,9 Mt
sivusto 2	10,74 s	0,9 Mt
sivusto 3	13,53 s	1,1 Mt
sivusto 4	14,57 s	1,2 Mt
sivusto 5	13,85 s	1,1 Mt
playcanv.as/p/HjT3DkVw	15,17 s	1,2 Mt
sivusto 6	16,17 s	1,5 Mt
sivusto 7	19,12 s	1,7 Mt
sivusto 8	24,67 s	2,1 Mt

WebGL-sovellus asettuu vertailussa keskitasolle muihin verkkosivuihin verrattuna. Erona muihin sivustoihin on, että sovellus käyttää esilatausmenetelmää. Sovellus lataa kaiken sisällön kerrallaan, eikä näytä sitä ennen kuin kaikki elementit näkymässä on ladattu. Muissa kohteissa sisältöön voi päästä käsiksi jo ennen varsinaista sivuston latauksen päättymistä. Voidaan kuitenkin todeta, että verrattaessa todellista latausaikaa osaan Suomen suosituimmista mobiilisivustoista sovellus pärjäsi tuloksissa hyvin.

Vertailutulosten lisäksi sovellus ladattiin Googlen tarjoaman verkkosivujen mobiilioptimointipalvelun läpi. Palvelu rakentaa sivustosta raportin, jossa kerrotaan sen soveltuvuus mobiiliympäristöön. WebGL-sovellus sai raportin tuloksissa mobiiliystävällisyydestä parhaat mahdolliset 100 pistettä ja sivuston nopeudesta 92 pistettä. Sivuston parannusehdotuksina raportti ehdotti JavaScript-tiedostojen supistamista.

6.6 Sovelluksen jatkokehitys

Sovelluksen mobiilioptimoinnin tulokset rajoittuvat neljän älypuhelinlaitteen vertailuun. Jotta sen toimivuudesta saataisiin kattavammat tulokset, sitä tulisi testata suuremmalla

määrällä laitteita. Tällöin saataisiin kattavampi määrä vertailudataa, jota hyödyntämällä sovellusta voitaisiin kehittää laajemmalle kirjolle laitteita. WebGL:n käyttöä ja 3D-grafiikan puskurointia rajoittavat laitteistojen komponentit ja alustat. Jatkotesteissä voitaisiin vertailla puhelinmallien lisäksi selainten ja käyttöjärjestelmien yhteensopivuutta sovelluksen kanssa.

Virtuaalisen esittelysovelluksen ja 3D-mallin puskurointi toimivat moitteettomasti ylemmän laitteistotason puhelinmallilla käytettäessä. 3D-mallintaessa otettiin huomioon käyttää mahdollisimman vähän verteksejä, polygoneja ja objekteja. Insinöörityön tuloksissa ei kuitenkaan kyetty todentamaan, mikä olisi ollut todellinen optimitaso yksityiskohtien määrässä mobiililaitteille. Kipukynnyksiä voitaisiin testata piirtämällä aluksi vain muutamia polygoneja sisältönäkymään ja kasvattamalla määrää siihen pisteeseen asti, kunnes huomattaisiin selkeitä laskuja kuvataajuudessa. Vertailu tehtäisiin eri laitteistotasojen välillä, ja niistä voitaisiin tehdä yhteenveto siitä, mitkä ovat ylärajat ja alarajat polygonien puskuroinnissa.

WebGL-sovelluksen toiminnallisuuksia ja ominaisuuksia on mahdollista kehittää entistä käyttäjäystävällisemmiksi. Osassa toimintoja on havaittavissa selkeitä puutteita, joita kehittämällä yleiskokemus ja käyttömukavuus kasvaisivat huomattavasti. Sovelluksen jo käynnistyessä voisi näkymässä esiintyä ilmoitus, jossa kehotetaan käyttäjää kääntämään mobiililaite vaakatasoon käyttökokemuksen parantamiseksi. Sisältönäkymä ja sovelluksen valikko asettuvat parhaiten mobiililaitteen näytölle, kun sitä käytetään vaakatasossa. Kehotusviestillä varmistettaisiin, että käyttäjä ymmärtää, ettei sovelluksen toimintoja ole optimoitu käytettäväksi mobiililaitteen ollessa pystyasennossa.

Päänäkymän ja VR-näkymän välistä siirtymätilaa voitaisiin kehittää siten, ettei VR-moodiin voitaisi päästä ilman mobiililaitteen gyroskooppia. Stereoskooppisen kameran käyttäminen ei tuo lisäarvoa, jos mobiililaitetta ei voida käyttää VR-lasien kanssa. VR-näkymäpainiketta klikattaessa sovellus voisi tarkistaa sisältääkö laite gyroskoopin. Jos kysely palautuu negatiivisena tuloksena, käyttäjää huomautettaisiin gyroskoopin puutteesta näytölle ilmestyvällä viestillä. Käyttäjälle voisi antaa tämän jälkeen mahdollisuuden vielä siirtyä VR-näkymään tai palata takaisin päänäkymään. Aktivoituvalla ilmoituksella varmistettaisiin, että käyttäjä ymmärtää, ettei mobiililaite tue kyseistä toiminnallisuutta.

Kahden näkymän lisäksi sovellukseen voisi kehittää kolmannen näkymän, jossa sisätiloihin voisi tutustua ilman, että laitteen tulisi olla VR-moodissa. PlayCanvaksen dokumentointi sisältää valmiit ensimmäisen persoonan kameraohjainkoodit, joita voitaisiin soveltaa työssä käytettyyn alustaan. Liikkuvuus on dokumentoiduissa ohjainkoodeissa kehitetty toimimaan käyttämällä tietokoneen näppäimistöä, mutta niitä voitaisiin soveltaa VR-näkymässä käytettyihin liikkumistoimintoihin. Näppäinohjaus korvattaisiin kameraan liitettävällä säteen lähettämiskoodilla, jota hyödynnettäisiin liikkumispisteiden kanssa tilassa navigoimiseen.

7 Yhteenveto

Insinöörityön toteutus onnistui määriteltyjen suunnitelmien mukaisesti. Tuloksena syntyi mobiilioptimoitu 3D-mallin WebGL-esittelysovellus, joka sisältää kaksi eri kameranäkymää, valikon ja kolme eri toiminnallisuutta. Suunnitteluun ja toteutukseen kului kokonaisuudessaan noin 130 tuntia. VR-näkymä toteutettiin käyttämällä stereoskooppista kameratekniikkaa, jonka liittäminen JavaScript-apukirjaston avulla oli vaivatonta. Sovelluksen käyttö rajoittuu kuitenkin suorituskyvyltään keskiluokan ja sitä tehokkaammille laitteille, jotka sisältävät 3D-grafiikan piirtämiseen vaadittavan suoritintehon. Tiedostokoot kyettiin optimoimaan mobiiliverkon käyttöön soveltuviksi, ja latausaika onnistuttiin saamaan standardien mukaiselle tasolle.

3D-mallin yksityiskohtien vaikutuksesta ei saatu tarkkoja julkaistavia tuloksia insinöörityöraporttiin. Epäselväksi jäi täsmällinen polygonien määrä, joka olisi mahdollista tuottaa korkean suorituskyvyn omaaville mobiililaitteille ennen havaittavissa olevaa kuvataajuuden laskua. Tämä olisi ollut kiinnostava tieto saada julki mahdollista jatkokehitystä varten, koska se antaisi tietyn kipurajan 3D-malleille. Jatkokehitystä ajatellen sovellusta tulisi testata laajemmalla laitevalikoimalla, jotta voitaisiin määritellä sen toimivuus selkeämmin.

Projekti sisälsi paljon eri osa-alueita ja työkaluja hallittavaksi, kuten mallintaminen, Blender-mallinnusohjelman käyttö, PlayCanvaksen editori ja JavaScript-syntaksi. Aikaisempi kokemus 3D-mallintamisesta ja WebGL-toteutuksista helpotti työn aloittamista ja suunnittelua. Kuitenkin Blender-ohjelman ja PlayCanvaksen käyttö tuli opetella alusta asti insinöörityötä varten, sillä työ haluttiin tehdä kokonaan avoimen lähdekoodin ohjelmistoilla.

Mielestäni WebGL-rajapinnalla on valoisa tulevaisuus kokemuksellisten sovellusten kehitysalustana etenkin virtuaalitodellisuustoteutuksille. Virtuaalitodellisuus on vasta tulossa kuluttajapuolelle käytettäväksi, ja sitä halutaan testata ennen VR-laitteiden ostopäätöksien tekemistä. Selainpohjaisuuden etuna on, ettei käyttäjän tarvitse ladata fyysistä sovellusta päätelaitteeseensa, mikä on usein esteenä, jos toteutuksia halutaan esitellä nopeasti ja sulavasti. Insinöörityössä tehdyn kaltaiset toteutukset soveltuisivat hyvin esimerkiksi messuille markkinointityökaluksi, jolloin käyttäjä voi kokeilla tuotetta omalla päätelaitteellaan ja halutessaan palata siihen myöhemmin.

Lähteet

1 Khronos Releases Final WebGL 1.0 Specification to Bring Accelerated 3D Graphics to the web without Plug-ins. 2011. Verkkodokumentti. Khronos Group. <https://www.khronos.org/news/press/khronos-releases-final-webgl-1.0-specification>. Luettu 15.10.2016.

2 Parisi, Tony. 2012. WebGL: Up and Running. Verkkodokumentti. O'Reilly Media, Inc. <http://proquestcombo.safaribooksonline.com/book/animation-and-3d/9781449326487/firstchapter>. Luettu 15.10.2016.

3 Cantor, Diego & Jones, Brandon. 2012. WebGL Beginner's Guide (1). Verkkodokumentti. Packt Publishing. <http://site.ebrary.com.ezproxy.metropolia.fi/lib/metropolia/reader.action?docID=10571194>. Luettu 15.10.2016.

4 GLSL 1.2 Tutorial. Verkkodokumentti. Lighthouse3D. <http://www.lighthouse3d.com/tutorials/glsl-12-tutorial/>. Luettu 22.10.2016.

5 WebGL Fundamentals. Verkkodokumentti. WebGLFundamentals. <http://webglfundamentals.org/webgl/lessons/webgl-fundamentals.html>. Luettu 22.10.2016.

6 OpenGL Shading Language. 2015. Verkkodokumentti. OpenGL. <https://www.opengl.org/wiki/OpenGL_Shading_Language>. Luettu 15.10.2016.

7 About The Khronos Group. Verkkodokumentti. Khronos Group. <https://www.khronos.org/about>. Luettu 15.10.2016.

8 Tolkach, Yuliya. 2016. WebGL: what is it and what is it used for? Verkkodokumentti. Altabel Group. <https://altabel.wordpress.com/2016/01/04/webgl-what-is-it-and-what-is-it-used-for/>. Luettu 15.10.2016.

9 Chapman, Steven. 2016. What Exactly Is JavaScript? Verkkodokumentti. About-Tech. <http://javascript.about.com/od/reference/p/javascript.htm>. Luettu 22.10.2016.

10 List of WebGL frameworks. Verkkodokumentti. Revolvy. <https://www.revolvy.com/main/index.php?s=List%20of%20WebGL%20frameworks>. Luettu 22.10.2016.

11 Jackson, Brian. 2015. What is Virtual Reality? Verkkodokumentti. Marxentlabs. <http://www.marxentlabs.com/what-is-virtual-reality-definition-and-examples/>. Luettu 23.10.2016.

12 Liimatainen, Karoliina. 2016. Lisätty todellisuus läikkyy jo työpaikoille. Helsingin Sanomat 25.9.2016, s. 6–7.

13 Catanzariti, Patrik. 2015. How to Build VR on the Web Today. Verkkodokumentti. Planet Nodejs. <http://www.planetnodejs.com/article/56004f2db01cdd0e004d68a7/how-to-build-vr-on-the-web-today>. Luettu 22.10.2016.

14 What is stereoscopic 3D? Verkkodokumentti. The Foundry. <https://www.thefoundry.co.uk/products/ocula/about-stereoscopic-3d/>. Luettu 23.10.2016.

15 Östergren, Magnus. 2016. SAVIO : NIGREDO. Verkkodokumentti. Chrome Experiments. <https://www.chromeexperiments.com/experiment/savio-nigredo>. Luettu 20.9.2016.

16 Make Games And VR For The Browser. Verkkodokumentti. Goo Create. <https://goocreate.com/product/>. Luettu 20.9.2016.

17 What is A-Frame?. Verkkodokumentti. A-Frame. <https://aframe.io/docs/0.3.0/introduction/>. Luettu 20.9.2016.

18 Create games, connect with your audience, and achieve success. Verkkodokumentti. Unity. <https://unity3d.com/unity>. Luettu 23.10.2016.

19 VR Overview. Verkkodokumentti. Unity. <https://unity3d.com/learn/tutorials/topics/virtual-reality/vr-overview?playlist=22946>. Luettu 29.9.2016.

20 Palma, Anthony. 2016. Exporting An Indie Unity Game to WebVR. Verkkodokumentti. Mozilla. <https://hacks.mozilla.org/2016/05/exporting-an-indie-unity-game-to-webvr/>. Luettu 20.9.2016.

21 PlayCanvas versus Unity WebGL. 2016. Verkkodokumentti. PlayCanvas. <https://blog.playcanvas.com/playcanvas-versus-unity-webgl/>. Luettu 29.9.2016.

22 Rasmussen, Alexander. 2013. How 3D Modeling Changed Architectural Presentations. Verkkodokumentti. 3D Printing. <http://3dprinting.com/products/ architecture/how-3d-modeling-changed-architectural-presentations/>. Luettu 29.10.2016.

23 Niipola, Jani. 2016. Virtuaalitodellisuus muuttaa kaiken. Verkkodokumentti. Kauppalehti. <http://www.kauppalehti.fi/uutiset/virtuaalitodellisuus-muuttaa-kaiken/9iz7TYp9>. Luettu 29.10.2016.

24 Taipale, Taru. 2016. Virtuaalinäyttöihin luottaminen voi käydä kalliiksi: asunnonostajalla on velvollisuus tarkastaa asunto ennalta. Verkkodokumentti. Helsingin Sanomat. <http://www.hs.fi/koti/a1476334967017>. Luettu 29.10.2016.

25 Lehtiniitty, Markus. 2016. Virtuaalitodellisuus tulee myös asuntonäyttöihin – Aktia alkaa lainata VR-laseja. Verkkodokumentti. MTV. <http://www.mtv.fi/lifestyle/digi/artikkeli/virtuaalitodellisuus-tulee-myos-asuntonayttoihin-aktia-alkaa-myos-lainata-vr-laseja/6114048>. Luettu 29.10.2016.

26 Frequently asked questions. Verkkodokumentti. Archilogic. <https://about.archilogic.com/faq/>. Luettu 2.11.2016.

27 How it Works. Verkkodokumentti. Matterport. <https://matterport.com/how-it-works/>. Luettu 2.11.2016.

28 PlayCanvas WebGL Game Engine. Verkkodokumentti. GitHub. <https://github.com/playcanvas/engine>. Luettu 29.9.2016.

29 Pricing. Verkkodokumentti. PlayCanvas. <https://playcanvas.com/plans>. Luettu 29.9.2016.

30 Sellers Calvin. 2013. 6 Tips on Designing Graphics for Mobile Devices. Verkkodokumentti. <http://uxmovement.com/mobile/6-tips-on-designing-graphics-for-mobile-devices/>. Luettu 20.9.2016.

31 Practical Guide to Optimization for Mobiles. Verkkodokumentti. Unity. <https://docs.unity3d.com/Manual/MobileOptimizationPracticalGuide.html>. Luettu 20.9.2016.

32 Optimizing graphics performance. Verkkodokumentti. Unity. <https://docs.unity3d.com/Manual/OptimizingGraphicsPerformance.html>. Luettu 20.9.2016.

33 Optimization Guidelines. Verkkodokumentti. PlayCanvas. <http://developer.playcanvas.com/en/user-manual/optimization/guidelines/>. Luettu 29.9.2016.

34 Features. Verkkodokumentti. Blender. <https://www.blender.org/features/>. Luettu 20.9.2016.

35 Textures/Mapping/UV/Unwrapping. Verkkodokumentti. Blender. <https://wiki.blender.org/index.php/Doc:2.4/Manual/Textures/Mapping/UV/Unwrapping>. Luettu 6.10.2016.

36 Cardboard VR. Verkkodokumentti. PlayCanvas. <http://developer.playcanvas.com/en/tutorials/beginner/cardboard-vr/>. Luettu 29.9.2016.

37 Raycasting. Verkkodokumentti. Unity. <https://unity3d.com/learn/tutorials/topics/physics/raycasting>. Luettu 20.9.2016.

38 Verkkokeskustelu. 2012. PlayCanvas. <http://forum.playcanvas.com/t/lerping-camera-motion-example-please/2143>. Luettu 10.10.2016.

39 Sprites. Verkkodokumentti. GitHub. <https://github.com/playcanvas/sprites>. Luettu 10.10.2016.

40 Runtime Lightmap Generation for WebGL. 2016. Verkkodokumentti. PlayCanvas. <https://blog.playcanvas.com/runtime-lightmap-generation-for-webgl/>. Luettu 29.9.2016.

41 Tilastotiedot. Verkkodokumentti. Phonearena. <http://www.phonearena.com/phones/compare/Samsung-Galaxy-S5-Neo,Apple-iPhone-6s,Sony-Xperia-M4-Aqua/phones/9526,9501,9265>. Luettu 2.11.2016.

42 Tilastotiedot. 2016. Verkkodokumentti. TNS Metrix 42/2016. <http://tnsmetrix.tns-gallup.fi/public/>. Luettu 2.11.2016.