Osaamista
ja oivallusta
tulevaisuuden
tekemiseen

Kasperi Koskelo

3D-grafiikan käyttö web-ympäristössä

Metropolia Ammattikorkeakoulu

Medianomi

Viestinnän koulutusohjelma

Opinnäytetyö

13.11.2018

Tekijä(t) Otsikko Sivumäärä Aika	Kasperi Koskelo 3D-grafiikan käyttö web-ympäristössä 23 sivua 13.11.2018
Tutkinto	Medianomi AMK
Koulutusohjelma	Viestinnän koulutusohjelma
Suuntautumisvaihtoehto	Digitaalinen viestintä
Ohjaaja(t)	lehtori Mari Silver

Tämän työn tarkoitus on selventää web-ympäristöön suunnatun 3D-grafiikan työstämiseen vaadittuja resursseja ja työkaluja. Keskeisimpänä kohderyhmänä löydetylle tiedolle ovat web-kehityksen parissa työskentelevät ohjelmistokehittäjät sekä suunnittelijat. Työssä tutustutaan 3D-grafiikkaa hyödyntäviin tärkeimpiin sovelluskohteisiin, joihin lukeutuvat esimerkiksi lisätty todellisuus ja virtuaalitodellisuus. Lisäksi työssä tarkastellaan näiden soveltuvuutta web-ympäristöön erilaisin lisätyökaluin. 3D-grafiikan tuottamiseen tärkeimpänä tarkastelun kohteena on WebGL-ohjelmointirajapinta ja sen päälle rakennetut liitännäiset.

Tieto tähän työhön on kerätty lähtökohtaisesti internetistä löytyvistä ohjelmointirajapintojen dokumentaatioista ja muista varteenotettavista web-kehitystä koskevista nettilähteistä. Syynä tiedonhaun rajaantumiselle internettiin on dokumentaatioista löytyvän tiedon varma ajantasaisuus.

Yhtenä case-esimerkkinä tässä työssä esitellään Crasman Oy:n alaisuudessa tehty työelämän proof-of-concept sovellus, joka hyödyntää 3D-grafiikkaa web-ympäristössä.

Lopputulemana tälle opinnäytetyölle on rajattu selvennys siitä mikä on nykyisellään web-ympäristössä 3D-grafiikan kannalta mahdollista, mikä saattaa olla lähitulevaisuudessa mahdollista ja mihin web-teknologia ei tällä hetkellä taivu.

Avainsanat	3D-grafiikka, web, WebGL

Author(s) Title Number of Pages Date	Kasperi Koskelo Usage of 3D graphics in web environment 23 pages 13.11.2018
Degree	Medianomi AMK
Degree Programme	Bachelor of Culture and Arts
Specialisation option	Digital Communications
Instructor(s)	senior lecturer Mari Silver

In this thesis, the main focus is kept on how to produce 3D graphics in the web environment. Since the introduction of WebGL, it has been possible to produce rich 3D graphics with the web browsers' own features without the need of additional plugins.

The aim of the thesis is to give a better overview about the usage of 3D graphics in the web for both designers and code oriented people. The approach to produce 3D graphics is, however, from the developers side but all information is presented in such a way that anyone who has worked with the computers can understand aspects presented.

The data for this thesis are collected from several different resources but the focus was kept on sources found on the Internet. The reason for this is simply that the current up-to-date information is found quickly from the documentation of programming API's examined by this thesis.

In this thesis, several different technologies and 3D production tools are examined including augmented reality, virtual reality and the basic tools used for the web suitable 3D graphics. Also, this thesis looks into one real life case executed in the work environment carried out under the employment of Crasman Oy.

The overall result of the thesis is the conclusion about what kind of 3D graphic solutions are simply possible to produce at the moment for the web environment with the tools and technologies examined earlier.

Avainsanat	3D grapics, web, WebGL

Sisällys

1 Johdanto

2010-luvun alussa selainkehittäjät toivat selaimiinsa mahdollisuuden 3D-grafiikan tuottamiselle natiiviominaisuudeksi, osaksi selaimia ilman erillisiä kolmannen osapuolen lisäosia. Ennen HTML5-standardin mukaisten natiiviominaisuuksien julkaisua web-kehittäjät hyödynsivät kolmannen osapuolen kehittämiä selainliitännäisiä erilaisten graafisten ratkaisujen tuottamiseen. Näihin selainliitännäisiin lukeutuvat esimerkiksi jo yli 20 vuotta vanhat Java ja Adoben kehittämä Flash. Nämä tarjosivatkin yli kymmenen vuoden ajan web-alan standardien mukaisen alustan selainkäyttäjän interaktion taltioiviin toteutuksiin.

Tavoitteeni tämän tutkielman aiheena toimivan 3D-grafiikan tutkimiseen kumpuaa puhtaasti henkilökohtaisesta mielenkiinnosta tietokonetta kohtaan visuaalisen informaation ilmentämisen mahdollistajana ja harrastuneisuudesta tietokonegrafiikan tuottamisen parissa. Näiden tekijöiden pohjalta pyrin tutkielmani avulla saavuttamaan ammatillista tietoa ja taitoa valmentaakseni itse itseäni mahdollisia työelämässä vastaan tulevia projekteja varten. Täten pyrkimykseni onkin ottaa selkeä edistysaskel kohti ammattimaisempaa lähestymistapaa 3D-grafiikkaa sisältävien web-sovelluksien kehittämistä varten.

Virtuaalitodellisuuden ja lisätyn todellisuuden toteutukset ovat riippuvaisia tietokoneiden kyvystä ilmentää grafiikka kolmiulotteisessa tilassa, joka visualisoidaan käyttäjälle kaksiulotteisen pinnan kautta. Tällainen kaksiulotteinen pinta on useimmissa tapauksissa tietokoneen tai mobiililaitteen näyttö. Vaikka virtuaalitodellisuuteen ja lisättyyn todellisuuteen liittyvät sovellukset voidaan katsoa jonkinasteisiksi trendi-ilmiöiksi, ovat ne tulleet jäädäkseen yhdeksi sovelluskehityksen potentiaalisimmista kehitysalueista. Niiden todellinen potentiaali suuremmassa mittakaavassa on kuitenkin tähän mennessä jäänyt vielä käyttämättä.

Näiden tekijöiden pohjalta voidaankin tutkielmani päällimmäiseksi tutkimustehtäväksi ja tavoitteeksi katsoa web-pohjaisten 3D-grafiikkaa sisältävien sovelluksien kehittämisessä ilmenevien haasteiden tunnistaminen vuonna 2018. Aihetta lähestytään puhtaasti teknisen toteutuksen näkökulmasta. Teknisestä näkökulmasta lähestyttäessä tarkoitus ei ole tarkastella 3D-grafiikan tuottamiseen vaadittavan ohjelmoinnin sisältöä, vaan tutkia yleisellä tasolla kolmiulotteisen tietokonegrafiikan vaatimuksia esimerkiksi selaimelta ja selainta ajavalta laitteelta. Tutkimus kohdistuu yleisesti tunnistettavissa oleviin kehitystyössä ilmeneviin teknisiin menetelmiin ja metodeihin.

Työkaluina tutkimuskysymyksen asettamien haasteiden tunnistamiseen käytän omakohtaista kokemustani kokeilemalla erilaisia 3D-grafiikan työstämiseen vaadittavia menetelmiä web-ympäristössä, tarkastelemalla aihetta koskevaa kirjallisuutta ja ennen kaikkea tutustumalla internetistä löytyvään ohjelmointirajapintoja käsittelevään dokumentaatioon. Yksi tärkeimmistä kohteista on tutustua web-ympäristöön suunnattuihin ohjelmointirajapintoihin, jotka tarjoavat selainten natiiviominaisuuksien päälle rakennettujen projektien parissa työskentelyä helpottavia ja nopeuttavia työkaluja sovelluskehityksessä. Haasteena ohjelmointirajapintojen kanssa on nopealla tahdilla vaihtuva alan trendien ja ilmiöiden seuraaminen, joiden kulkua ohjelmointirajapintojen suosio mukailee.

Omakohtainen kokemukseni 3D-grafiikan parissa kumpuaa nuorena alkaneesta kiinnostuksesta demoskeneä kohtaan. Demoskenessä keskitytään kilpailemaan tietokonegrafiikkaan pohjautuvilla tuotoksilla, joissa pääpaino on näyttävien visuaalisten ja graafisten efektien aikaansaaminen. Kilpailukategorioilla rajoitetaan käytettävissä olevien resurssien määrää esimerkiksi asettamalla rajoituksia tuotoksen lopulliseen tietokoneen kovalevyllä viemään kokoon. Demojen eli tuotoston tekijät joutuvat toimimaan asetettujen rajoitteiden puitteissa tehdessään kilpailutyötään vaatien tekijöiltään aina edistyksellisempiä tapoja toteuttaa halutut efektit.

Materiaalina tutkimukselle toimivat lähtökohtaisesti netistä löytyvät artikkelit sekä rajapintadokumentaatiot niiden täsmällisyyden ja ajankohtaisuuden takia. Aihetta koskevasta kirjallisuudesta saatu tieto on suhteellisen vanhentunutta teknologian nopean kehityksen takia verrattuna jatkuvasti päivitettäviin dokumentaatioihin, ja täten jätänkin sen siitä syystä tiedonhankinnan ulkopuolelle.

Tutkimustyön ohessa työstän Crasman Oy:ssä työsuhteeni puitteissa asiakasprojektia, jonka tarkoituksena on toimia asiakkaan tuotteiden esittelyn tukena. Tuotoksen on tarkoitus antaa sekä loppukäyttäjälle että Crasman Oy:ltä ohjelmiston toteutuksen tilanneelle asiakkaalle uuden näkökulman tuotteen myyntiprosessiin. Toteutettavan tuotoksen ei ole tarkoitus toimia myyntiprosessin ensimmäisenä vaiheena, vaan ennemminkin auttaa harkitun päätöksen tekemisessä visualisoimalla tuotteen sen oikeassa ympäristössä käyttäen kaksiulotteiselle pinnalle piirrettyä 3D-mallia.

Tulokulmana tässä opinnäytetyössä tarkasteltaviin asioihin toimii käyttäjälähtöisyys. Käytännössä asioita tarkastellaan käyttäjän kannalta optimaalisimmasta näkökulmasta,

käytettävyyden helppoutena ja vaivattomuutena. Tämän oletetaan olevan lukijalle entuudestaan tuttua, eikä siksi tutkimuksessa syvennytä käytettävyyden teoriaan.

Opinnäytetyön tarkoitus on toimia 3D-grafiikkaa web-ympäristöön työstävien kehittäjien ja suunnittelijoiden ymmärrystä parantavana pintaraapaisuna siitä mikä on suinkin resursseista riippuen mahdollista toteuttaa web-selaimella katseltavaan sivustoon. Tutkimuksen näkökulmana haasteita lähestytään tekniseltä suunnalta, mutta saatu informaatio on niin suunnittelijoiden kuin teknisempien kehittäjienkin jäsenneltävissä.

2 Sanastoa

Tutkielmassa käytettävä sanasto on painottunut alan tekniseen termistöön, jota avataan tässä luvussa.

Lisätty todellisuus (eng. augmented reality), eli lyhennettynä englanninkielisen käännöksen mukaisesti AR, on tapa näyttää tietokoneella tuotettua grafiikkaa läpikatseltavan ruudun kautta oikean ympäristön päällä. Esimerkiksi 3D-objektin lisääminen reaaliaikaisesti mobiililaitteen kameranäkymään, jossa 3D-objekti näkyy reaaliaikaisesti kameranäkymän esikatselukuvan päällä, hyödyntää lisättyä todellisuutta.

Virtuaalitodellisuus (virtual reality) on tapa näyttää käyttäjälle sisältöä kaksiulotteisen pinnan, eli tässä tapauksessa tietokoneen tai mobiililaitteen ruudun, kautta tietokonesimulaatiota hyödyntäen. Virtuaalitodellisuus hyödyntää teknologian suomia mahdollisuuksia mukailla oikeaa ympäristöä virtuaalisesti luomalla täysin tietokoneella tuotetun ympäristön, jossa käyttäjä sovelluksesta riippuen voi toimia eri tavoin. Verrattuna lisättyyn todellisuuteen, jossa nimensä mukaisesti lisätään todelliseen ympäristöön tietokoneavusteisesti elementtejä, virtuaalitodellisuuteen perustuvissa sovelluksissa käyttäjän katselema ympäristö on lähtökohtaisesti täysin tietokoneella tuotettuun grafiikkaan perustuva.

Internetselain tarkoittaa World Wide Webistä löytyvien, lyhennettynä WWW tai web, internetsivujen katseluun käytettävää tietokoneohjelmaa. Selain hakee verkosta käyttäjän haluaman sisällön ja tulkkaa sen käyttäjän nähtäväksi visuaalisesti katseltavaan muotoon.

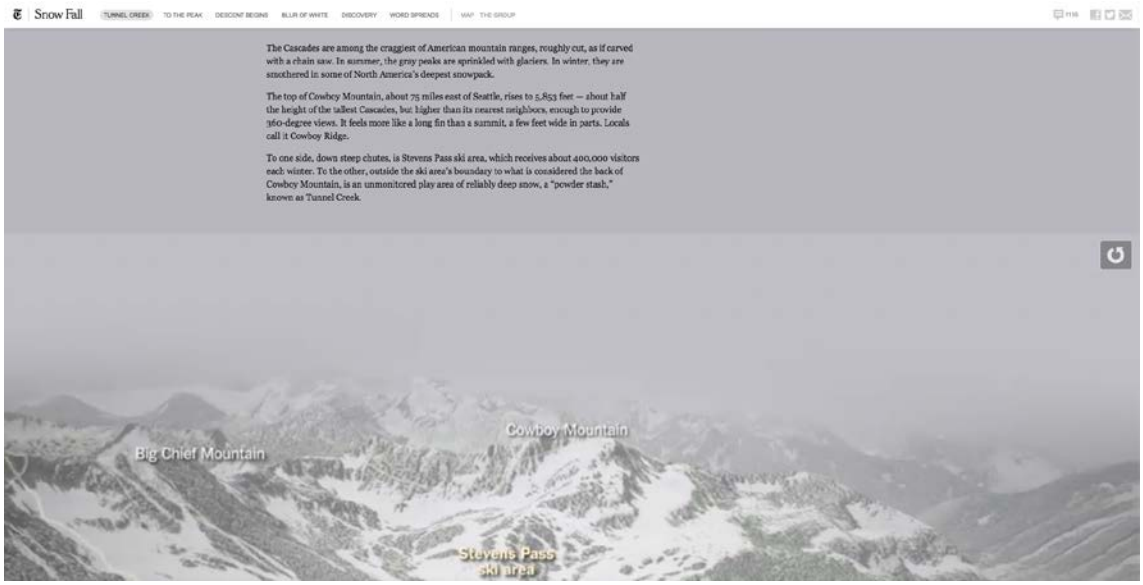
Web-ympäristöllä viitataan internetselaimen käytettävissä oleviin resursseihin ja mahdollistamaan visuaaliseen näkymään. Tämä onkin tutkielman kannalta oleellinen rajaus sulkemaan pois kaikki web-selaimen ulkopuoliset toteutukset. Tähän kuitenkin lukeutuu useita eri laitealustoja mukaan lukien mobiililaitteet, joiksi lasketaan perinteiset älypuhelimet ja tabletit, sekä tietokoneiden työpöytäversiot selaimista.

Ohjelmointirajapinta on ohjelmoinnissa eli tietokoneelle tuotettavien sovelluksien kehityksen parissa käytettävä tapa mahdollistaa tiedon jakaminen erilaisten työkalujen välillä. Tässä tekstissä termillä tarkoitetaan web-ympäristöön suunnattujen ohjelmointikirjastojen käyttöä, joilla erilaisia mutkikkaita toimintoja pystytään suorittamaan huomattavasti kevyemmällä toteutuksella.

3 Tietokonegrafiikan käyttötarkoitukset web-ympäristöt

Tietokonepohjaisen grafiikan käyttötarkoitukset ovat tulleet vuosien saatossa yhä lähemmäksi perinteistä kuluttajaa laitteiden laskentatehon huiman kasvun myötä. Nykyiset mobiililaitteet kykenevät huomattavasti nopeampaan prosessointiin kuin ensimmäisiksi kotimikroiksi mielletyt laitteet. Selkeiten asia näkyy kuluttajille suunnatussa markkinoinnissa huomion keskipisteen siirtymisellä laitteiden perinteisistä ominaisuuksista laitteiden välisen komponenttitason vertailuun. Kasvaneen prosessointitehon myötä ovat myös web-sisällön kulutuksen tavat muuttuneet yhä visuaalisemmiksi.

Tarkasteltaessa netin suurimpia uutisjulkaisuja on lähtökohtaisesti aina uutisen tai jutun aiheen halutun viestin tukena jokin visuaalinen kuva, joka tuo näin lisää ulottuvuutta informaation välittymiseen. Uudeksi ilmiöksi 2010-luvulla on noussut pidempien tekstien tukeminen reaaliaikaisesti tuotettujen tai valmiiksi työstettyjen videoiden avulla. Näitä yleisesti perinteisestä tyylistä poikkeavia pitempiä tuotoksia on esiintynyt esimerkiksi Suomessa Ylen julkaisemana. Informaation virtaa näissä dokumenteissa ohjataan useampaan lohkoon tai osuuteen jaetuilla katkelmilla tekstistä ja multimedia ratkaisuksi muodostuneilla kuvituksilla, joissa esiintyy joko täysin tuotettua sisältökuvitusta tai aiheeseen liittyviä videotallenteita. Lisäksi näissä mediakokonaisuuksissa saatetaan käyttäjää ohjata tekemään erilaisia valintoja näin pelillistäen lukukokemuksen ja ohjaten lukijaa valintojen perusteella kohti juttukokonaisuuden eri lopputuloksia.



Kuvio 1. Ruutukaappaus The New York Timesin julkaisemasta The Snow Fall artikkelikokonaisuudesta (Branch 2012).

Suurin 3D-grafiikan hyödyntäjä on valtavaksi bisnekseksi muodostunut peliteollisuus. Juuri peliteollisuus on yksinään suurimpia tekijöitä graafisten elementtien prosessointiin vaaditun laskentatehon kasvamiseen, mutta ala on vaikuttanut myönteisesti myös tehokkaampien prosessointialgoritmien kehitykseen. Peliteollisuuden ohella myös elokuva-alalle on syntynyt tilanne, jossa kokopitkiä elokuvia toteutetaan täysin tietokoneella tuotettua grafiikkaa hyödyntäen. Peliteollisuuden ja elokuva-alan ero kuitenkin näkyy peliteollisuuden nojautuessa tietokoneiden ja pelikonsolien kykyyn tuottaa 3D-grafiikkaa reaaliaikaisesti. Elokuva-alan tuotannot vastaavasti nojaavat valmiiksi pureskeltuun videomateriaaliin, joka ei aseta samanlaisia vaatimuksia elokuvan katseluun käytetyn laitteen prosessointiteholle ja antaa näin vapaammat kädet tekijöille yksityiskohtaisempiin toteutuksiin.

Kuvio 2. Ruutukaappaus selaimessa pelattavasta massiivimoninpeli RuneScapesta (Jagex 2018).

Web on ympäristönä toiminut jo 2000-luvun alkupuolelta alustana kevyemmin toteutetuille peleille. Internet täyttyi 2000-luvun puolivälin jälkeen satoja tai jopa tuhansia pieniä Flash- tai Java-pohjaisia pelejä sisältävistä sivustoista, joilla käyttäjät pystyivät pelaamaan asennettuaan selaimeensa pelien vaatiman lisäosan. Webistä löytyy kuitenkin

myös peliteollisuuden isompiin julkaisuihin rinnastettavissa olevia pelikokonaisuuksia, joihin on tuotu mukaan massiiviset moninpelio minaisuudet.

Informaation yksiselitteisessä välittymisessä suurien kaavamaisuutta noudattavien datamassojen jäsentely graafiseen muotoon on lähes aina tiedon vastaanottajan kannalta helpottava tekijä. Web onkin ympäristönä muovautunut tiedon graafisessa välittämisessä helpoksi alustaksi visualisoida informaatiota eri muodoissa. Yksittäin tätä tehtävää varten on web-ympäristöön rakennettu lukuisia eri käyttötarkoituksia palvelevia ohjelmointikirjastoja, kuten D3.js ja Chart.js, jotka tarjoavat valmiit työkalut suurienkin informaatiomassojen visualisoimiseen.

4 Selaimet ja web-ympäristön standardit

Internetin selaamiseen tarvittavia verkkoselaimia, jotka tulkkaavat verkkosivujen kuvauskielen visuaaliseen muotoon, on vuoteen 2018 mennessä kehitetty useampia erilaisia eri käyttötarkoituksia varten. 2000-luvun vaihteen jälkeen suosituimpia selaimia olivat Netscape Navigator ja Microsoft Internet Explorer (W3C 2000). Nykyisin suurin käyttäjäkunta on Googlen kehittämällä Chrome-selaimella, jota käyttää yli 60 prosenttia netin käyttäjistä. Kilpailevia selaimia ovat esimerkiksi Applen kehittämä Safari, Mozillan kehittämä Firefox ja norjalaisen Opera Softwaren kehittämä Opera. (StatCounter 2018.)

Taulukko 1. Käytettävien laitteiden jakauma syyskuussa 2018 (Statcounter 2018)

Mobiili	Työpöytä	Tabletti
51,7%	*44,1%*	*4,2%*

Mobiililaitteiden suosion ja kehittymisen myötä selainkehittäjät ovat joutuneet kääntämään selaimensa myös suurimmille mobiilialustoille. Nykyisin verkkosuunnittelussa noudatetaan mobile first -lähtökohtaa, jossa verkkosisältö suunnitellaan mobiilialustoista aloittaen ylöspäin ruutukokojen mukaisesti. Näin taataan sisällön yhdenmukaisuus riippumatta käytettävän laitteen näyttökoosta. Tilastojen mukaan verkkosivuja käytetään nykyään enemmän mobiililaitteilla kuin perinteisillä pöytätietokoneilla. (Awio Web Services LLC 2018.)

Taulukko 2. Käytettävien selaimien jakauma syyskuussa 2018. (Awio Web Services LLC 2018)

Chrome	Safari	Firefox	Internet Explorer	Opera
62.2%	*13.4%*	*7.1%*	*6.3%*	*3.0%*

Mobiililaitteille tyypillistä on niiden huomattavasti pienempi näytön tarkkuus ja koko verrattuna suurempiin perinteisten tietokoneiden näyttöihin. Myös käyttäjän interaktion tallentaminen kosketuksen kautta asettaa mobiililaitteille huomattavasti enemmän rajoitteita niiden käytettävyydessä. Tällaisen asetelman takia mobiilille suunnitellut selaimet eivät vastaa ominaisuuksiltaan lähellekään työpöytäversioita selaimista. Esimerkiksi vaikka Android-käyttöjärjestelmälle sovellusten kehittäminen tapahtuu käyttäen Java-ohjelmointikieltä, eivät Androidille suunnatut selaimet tue samalla tapaa Java-pohjaisia

web-toteutuksia kuin työpöytäversiot selaimista. Sama tilanne pätee myös Adoben kehittämään Flash-lisäsosaan, jolle ei löydy virallista tukea isoimpien mobiililaitevalmistajien toimesta. Tämä ohjasikin älypuhelimien vallatessa markkinoita sovelluskehittäjiä kehittämään työpöytätietokoneilla toimivista perinteisistä verkkosovelluksistaan mobiililaitteille omia natiiviapplikaatioitaan. Koska mobiililaitteille ei löydy yhtä yhdenmukaista käyttöjärjestelmää, on natiiviapplikaatioista kehitettävä yleisesti tiedostetun käytännön mukaisesti Androidille ja Apple iOS:lle omat versionsa. Käytettävyyden kannalta näissä saattaa olla laitteiden käyttöominaisuuksista johtuen erilaisia käyttöliittymäratkaisuja, jotka entisestään lisäävät eroavaisuuksia yhdenmukaisuudessa näiden kahden käytetyimmän mobiililaitekäyttöjärjestelmän välillä.

HTML5-standardi julkaistiin virallisesti vuonna 2014, mutta jo ennen sitä oli se käytettävissä vuodesta 2008 eteenpäin. Kuitenkin vuonna 2014 W3C otti sen käyttöön suositukseksi web-kehityksessä, jotta kehitetyt sivut vastaisivat nykyaikaisia web-standardeja. HTML5-standardointi toi mukanaan useita uusia elementtejä HTML-kuvauskieleen, joihin lukeutuvat esimerkiksi <*canvas*>-, <*video*>- ja <*audio*>-elementit sekä lisäksi jaottelua näkyvien ja näkymättömien HTML-elementtien välillä parannettiin tuomalla mukaan nykyaikaista web-suunnittelua vastaavat elementit <*header*>-, <*nav*>- ja <*footer*>-elementit. (Bright 2018; The Mozilla Foundation 2018.)

Nykyisin web-ympäristössä näytettävä 3D-grafiikka tukeutuu pitkälti HTML5-standardin mukana lanseerattuun <*canvas*>-elementin käyttöön WebGL-rajapinnan avulla. WebGL-rajapinta on JavaScript-ohjelmointikielelle rakennettu OpenGL (*Open Graphics Library*) -rajapintaominaisuus, jonka avulla selain pystyy käyttämään OpenGL:stä tuttuja grafiikan tuottamismenetelmiä natiivisti selaimessa. OpenGL on yleisimpiin käyttöjärjestelmäriippumattomiin tietokonegrafiikan tuottamiseen tarkoitettuihin ohjelmointirajapintoihin kuuluva kirjasto (OpenGL.org, 2018). Se on laajalti käytössä 3D-grafiikkaa vaativissa tuotannoissa sekä peliteollisuudessa. OpenGL:n etuna sen kilpailijoihin on sen käyttöjärjestelmäriippumattomuus verrattuna esimerkiksi Microsoftin kehittämään DirectX-rajapintaan, joka vaatii toimiakseen Microsoft Windows -käyttöjärjestelmän tai Microsoft Xbox -pelikonsolin tai vaihtoehtoisesti Applen kehittämään Metal API:in.

HTML5-standardin myötä oletuksena voidaan pitää, että kaikki viimeisen parin vuoden sisällä valmistetut laitteet tukevat WebGL-rajapintaa niin mobiililaitteilla kuin myös pöy-

tätietokoneilla. Riippumatta käytettävästä laitteesta pitäisi katseltavan sivun toimia ongelmitta. Haastavaksi tekijäksi tässä tietenkin muodostuu suunnittelun lähtökohdat toteutetun sivun mobiiliystävällisyyteen. (Caniuse 2018; WebGLStats 2018.)

3D-grafiikkaa sisältävien toteutuksien rakentaminen mobiiliystävällisiksi muuttuu ongelmalliseksi näyttökoon kutistuessa hyvin kapeaksi. Pienikokoisilla näytöillä suurta tilaa vaativien graafisten toteutuksien näyttäminen kadottaa helposti informaation välittämisen soljuvuuden grafiikan “mössöytyessä” tulkintakelvottomaksi tai grafiikan syödessä käytettävästä pinnasta liian suuren osan. Myös käytettävän selaimen ja käyttöjärjestelmän luomat käyttöliittymäratkaisut saattavat hankaloittaa huomattavasti käytettävissä olevan tilan käyttöä.

Vaikka älypuhelimet ja pöytätietokoneet pystyvät toistamaan web-ympäristöön tuotettuja 3D-sovelluksia usein sulavasti, ei oletuksena voida kuitenkaan pitää jouhevaa toistoa mutkikkaiden tuotosten äärellä. Koska tuotokset rakentuvat selaimen omien ominaisuuksien päälle, on niiden vaatima prosessointiteho usein huomattava, jos graafisen elementin ala kattaa suuren osuuden näytöstä. Tämä ilmenee usein toiston hitautena ja töksähtelynä.

5 WebGL-grafiikan tekninen toteutus web-ympäristöön

Tässä luvussa keskitytään WebGL-grafiikan tekniseen toteutukseen ja vaatimuksiin selaimelta. Vertailukohtana grafiikan tuottamiseen kykenevänä pintana käytetään JavaScriptin mahdollistamaa canvas-elementin 2D-kontekstia, vaikka se ei läheskään vastaa WebGL:n monimutkaisuutta. Luvun aihepiiristä on rajattu pois muut 3D-grafiikan ja tietokonegrafiikan visualisoimisen mahdollistavat selainliitännäiset, koska nämä vaativat erillisen asennuksen selaimeen. WebGL-tuen voidaan olettaa löytyvän kaikista tämän päivän moderneista ja yleisesti käytetyistä selaimista, mukaan lukien mobiiliselaimet.

Grafiikkaa sisältäviä web-toteutuksia tehtäessä on syytä miettiä toteutuksen kannalta optimaalisin ja kevyin tapa halutun näkymän aikaansaamiseksi. Jos suunniteltu näkymä ei esimerkiksi sisällä 3D-grafiikkaa, on sellaista tuottavien ohjelmointikirjastojen käyttö usein työlästä ja hyödytöntä kaksiulotteisen grafiikan työstämiseen.

Koska WebGL on tietokoneen grafiikkakorttia ohjaava rajapinta, on sen käyttö suoraan grafiikan tuottamiseen hyvin työlästä ja teknisesti vaativaa. Siksi asiaa helpottamaan onkin tehty lukuisia JavaScript-pohjaisia kirjastoja, joiden avulla sovelluskehitystä pystytään nopeuttamaan. Ohjelmointikirjastot tuovat kehittäjälle WebGL:n päälle rakennettuja työkaluja ja komentoja edesauttamaan helppokäyttöisyyttä. Siinä missä WebGL:llä yksinkertaisen grafiikan tuottaminen vaatii kymmeniä rivejä koodia, pystyy tällaisella kirjastolla sen toteuttamaan huomattavasti nopeammin pienemmällä määrällä koodirivejä. Esimerkkejä tällaisista kirjastoista ovat esimerkiksi Three.js, Pixi.js ja D3.js.

Työtä nopeuttavien kirjastojen käyttö ei kuitenkaan aina ole toteutuksen vaatimuksista riippuen järkevää. Käytettävä kirjasto asettaa tietyt rajat sen mahdollistamiin toteutusteknisiin ratkaisuihin, ja näiden rajojen ulkopuolella toimiminen luo riskin sovelluksen epävakaudesta. Myös huonosti optimoidut kirjastot saattavat lisätä selaimen prosessointikuormaa tarpeettomilla laskentaprosesseilla. Sen takia suoraan WebGL-syntaksin kirjoittaminen yksinkertaista grafiikkaa sisältäviin toteutuksiin saattaakin olla välttämätöntä.

Tietokoneista löytyvä grafiikkakortti eli lyhyemmin GPU (*Graphics Processing Unit*) sisältää tuhansia tietokoneella tuotetun grafiikan laskemiseen käytettäviä prosessoriytimiä. Verrattuna perinteiseen keskussuorittimeen eli CPU:hun (*Central Processing Unit*), jolla suoritetaan tietokoneen sovelluksiin liittyvät laskennalliset prosessit, on ytimien määrä huomattavasti suurempi. Grafiikkakortti vastaa tietokoneen näytölle piirrettävän

ulosannin prosessoinnista. Grafiikkakorteista löytyvien suoritinytimien määrä on eduksi tietokonegrafiikan piirrossa vaadittuun reaaliaikaiseen prosessointiin, jossa näytettävä näkymä piirretään useita kertoja uudestaan sekunnin aikana saumattoman liikkeen saavuttamiseksi. Tämä toteutetaan grafiikkakortin suorittimien mahdollistaman rinnakkaisprosessoinnin avulla. (PC Mag 2018.)

WebGL toimii selainympäristössä rajapintana selainta ajavan tietokoneen tai mobiililaitteen grafiikkakorttia ohjaaville ajureille. Käytännössä selaimen ajama WebGL:ää sisältävä koodi välittää tiedon halutuista laskutoimituksista grafiikkakortin ajurille, joka välittää pureskellun tiedon grafiikkakortille. Grafiikkakortilta palautuu käyttäjän ruudulle selainnäkymässä näkyvä kuva. Suoraan WebGL:llä kirjoitettu koodi on rivimäärältään suurempaa ja luettavuudeltaan hankalampaa. Esimerkiksi JavaScriptin 2D-ominaisuuksilla HTML:n canvas-elementille piirretty värillinen kolmio vaatii alle alle kymmenen riviä koodia, kun vastaavan toteuttaminen WebGL:ää käyttäen vaatisi useita kymmeniä. Esimerkissä tulee kuitenkin ottaa huomioon käytettyjen työkalujen käyttötarkoitus, ja näin ollen eivät ne ole täysin vertailukelpoisia muutoin kuin syntyneen tuloksen puolesta. WebGL:llä grafiikkaa piirrettäessä tulee grafiikkakortille välittää huomattavan paljon piirtoon käytettäviä asetuksia. Sen lisäksi piirtäminen WebGL:ää käyttäen vaatii hyvän matemaattisen hahmottamiskyvyn WebGL:n ottaessa sille annetun datan sisään käyttäen vektoreita ja matriisitaulukoita.

```
<!DOCTYPE html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>JS Triangle</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
  </head>
  <body>
      <canvas id="canvas"></canvas>
      <script>
        function draw() {
          var canvas = document.getElementById('canvas');
          if (canvas.getContext) {
            var ctx = canvas.getContext('2d');

            ctx.beginPath();
            ctx.moveTo(75, 50);
            ctx.lineTo(100, 75);
            ctx.lineTo(100, 25);
            ctx.fill();
          }
        }
        draw();
      </script>
  </body>
</html>
```

Kuvio 3. JavaScriptin 2D-ominaisuuksilla kolmion piirtoon vaadittu koodi.

Koska WebGL on OpenGL:n päälle rakennettu rajapinta, on se syntaksiltaan hyvin lähellä OpenGL:n muiden ohjelmointikielien rajapintavastineita, vaikka itse ohjelmointi tapahtuukin käyttäen JavaScriptiä. WebGL:lle kirjoitetut shader-ohjelmat kuitenkin mukailevat GLSL:n (OpenGL Shading Language) mukaista syntaksia, joka muistuttaa läheisesti C- ja C++ -ohjelmointikieliä. Shaderit ovat pieniä grafiikan työstämisessä tarvittuja ohjelmia, jotka kirjoitetaan erillään itse grafiikan tuottavasta sovelluksesta. Päällimmäinen käyttötarkoitus shaderien käytöllä on erilaiset grafiikan väritykseen liittyvät tehtävät, kuten valoisuuksien ja heijastuksien laskeminen. (Vivo 2018).

Koska WebGL on syntaksiltaan hyvin vaikeaselkoista, on sen päälle rakennettu web-ympäristön 3D-ominaisuuksien valjastamisen helppoutta ja jouhevuutta tukemaan ohjelmointikirjastoja, jotka tarjoavat valmiiksi lukuisia työkaluja 3D-grafiikan käsittelyyn. Huomattavin näistä on Three.js-kirjasto, joka pitää sisällään lukuisan määrän valmiita kutsuja 3D-grafiikan nopeaan tuottamiseen. Käyttämällä Three.js-kirjastoa ei kehittäjän esimerkiksi tarvitse huolehtia itse WebGL:n vaatimista monimutkaisista datan välittämiseen tarvittavista asetuksista.

6 Lisätty todellisuus, virtuaalitodelisuus ja 360-kuvantaminen

Vaikka virtuaalitodellisuus ja lisätty todellisuus ovat termeinä vetäneet huomion puoleensa IT-alalla ensimmäisen kerran jo useampia vuosia sitten, on näiden teknologioiden valjastaminen niiden todelliseen potentiaaliin jäänyt hyvin vähäiseksi.

Vuonna 2015 Google otti YouTube-videopalvelussaan käyttöön tuen 360-videoille (Bonnington 2018). Tuki astui voimaan työpöytäselaimilla ja Android-pohjaisilla mobiililaitteilla. Facebook otti käyttöön vastaavan tuen 360-videoille ja kuville sosiaalisen median palveluunsa vuonna 2017 (O'Kane 2017). Nämä toteutukset sallivat videon tai kuvan katsomisen käyttäjän haluamaan suuntaan tarjoamalla sisällön tuottaneelle taholle mahdollisuuden käyttäjän ohjaamiseen kiinnostavaa sisältöä kohden. Näin molempiin palveluihin saatiin lisää käyttäjäinteraktiota tuottavaa sisältöä riippumatta käytettävästä päätelaitteesta. Eniten kyseisistä ominaisuuksista saa irti vasta hankittuaan VR-laseiksi kutsutun lisäosan, joka reagoi käyttäjän pään liikkeisiin ja liikuttaa kuvakulmaa käyttäjän valitsemaan suuntaan. Sisältöä pystyy tarkastelemaan myös perinteisillä päätelaitteilla käyttäen joko mobiililaitteiden kosketusnäyttöjä, mobiililaitteiden gyroskoopin toimintaan perustuvaa laitteen kääntelyä tai työpöytäversiolla hiirellä kuvakulman raahamista.

Uusimpana lisäyksenä sosiaalisen median palveluunsa Facebook lisäsi vuoden 2018 puolivälissä mahdollisuuden 3D-kuvien lisäämiselle kaksikameraisilla älypuhelimilla. Kuvat ovat käytännössä kuitenkin vain älypuhelimen kahdelle vierekkäisellä kameralla tehtyjä stereoskooppisia kuvia. Älypuhelimen kamerat ottavat kaksi hiukan toisistaan perspektiivissä eroavaa kuvaa, ja näiden perusteella luodaan syvyyskartta, jonka avulla mallinnetaan kuvaan stereoskooppista efektiä luova näkymä. Kuvaa voi tarkastella hivenen eri kuvakulmista katseluun käytettävää laitetta kääntelemällä tai pöytätietokoneella hiirellä kuvakulmaa liikuttelemalla. (Robertson, The Verge, 2018.)

Myös peliteollisuus on herännyt lisätyn todellisuuden ja virtuaalitodellisuuden mahdollistamiin toteutustapoihin. Viime vuosien tunnetuimmaksi ja suosituimpien AR-pohjaisten pelien joukkoon on noussut yhdysvaltalaisen Niantic Labsin kehittämä lisättyyn todellisuuteen tukeutuva Pokemon GO, jossa pelaajat seikkailevat ympäri kaupunkeja pelin hyödyntäessä kaupunkien karttoja pelimaailmanaan. Peli sijoittaa karttojen perusteella pelille ominaiset elementit oikeisiin kohteisiin ja saa pelaajat liikkumaan ympäri kaupunkia. Pokemon GO on saavuttanut suuren suosion ja kerää pelaajia eri ikäryhmistä ympäri maailmaa pelin äärelle. (Perez 2018).

AR.js-niminen rajapintakirjasto tarjoaa JavaScriptin päälle mahdollisuuden simuloida lisättyä todellisuutta fyysisen paperiarkin avulla. Paperiarkilla tulee olla ohjelmiston tunnistama mustavalkoinen QR-koodia muistuttava geometrinen muoto, jonka päälle kehittäjä pystyy sijoittamaan haluamansa tyylisiä 3D-elementtejä tai muuta vastaavaa sisältöä. Nykyisen selaintuen puitteissa ainoa kunnollinen tapa toteuttaa AR-ominaisuuksia sisältäviä sovelluksia on tukeutua fyysiselle paperiarkille tulostettuun kuvioon. (Etienne 2018).

Toiminta geometriseen muotoon perustuvassa AR.js -kirjastossa pohjautuu mobiililaitteen tai tietokoneen web-kameran avulla geometrisen muodon tunnistamiseen sen symmetrisyyden takia ympäröivästä ympäristöstä. Tämä vaatii aina kuitenkin fyysiseen kontaktipisteeseen nojautumisen, jonka virkaa hoitaa joko piirretty tai tulostettu kuvio.

AR-toteutuksissa vaaditaan itse käyttöympäristöltä hyvä valaistus, jotta käytettävän laitteen kamera kykenee toimimaan liikkeen tunnistamiseksi ympäröivästä ympäristöstä. Sen lisäksi ympäristön tulee pitää sisällään selkeitä kiintopisteitä, joiden avulla laite kykenee laskelmoimaan katseltavaan kuvaan halutut elementit. Hyvänä esimerkkinä selkeästä kiintopisteestä toimii ihmiskasvot, joiden päälle erilaisissa sosiaalisen median palveluissa pystytään liittämään tietokoneella tuotettuja elementtejä.

7 Case: AR-projekti asiakastyönä

Toimiessani Crasman Oy:ssä web-kehittäjänä vedettiin minut mukaan suunnittelutiimistä lähtökohtaisesti kummunneeseen projektiin 3D-grafiikan parissa aikaisemman harrastuneisuuteni takia. Visiona oli toteuttaa asiakkaalle proof-of-concept-tyyppinen web-toteutus, jossa hyödynnettäisiin AR-teknologian mahdollisuuksia ilmentää 3D-malleja mobiililaitteen avulla. Tavoitteena oli saada asiakkaalle myytävä esimerkkituotos, joka vastasi toiminnallisuuksiltaan karkeasti lopullista sovellusta kuitenkin jättäen pois käytettävyyden kannalta oleellisia osia ja viimeisen käyttöliittymän silottelun. Proof-of-concept-tuotoksen päällimmäinen tarkoitus olikin vain kartoittaa toteutettavissa olevan web-sovelluksen vaatimat tekniset resurssit ja yleisesti sen onnistumisen varmistaminen. Kohde asiakkaan toteutukselle oli hirsitaloja valmistava yritys. Projektin lopputuloksena oli siis tarkoitus saada nimenomaisesti web-selaimessa toimiva käyttöjärjestelmäriippumaton 3D-grafiikkaan tukeutuva tuotteita esittelevä sovellus. Tuotteen kohderymänä ovat asiakkaan omat paikallismyyjät, jotka pystyisivät sovelluksen avulla havainnollistamaan asiakkaalle 3D-grafiikkaa hyödyntäen hirsirakennuksen asettumisen asiakkaan tontille.

7.1 Tekninen toteutus

Proof-of-concept-tuotoksen valmistelu alkoi Crasman Oy:n suunnittelutiimin kanssa yhteisellä kokouksella, kartoittamalla ja tarkastelemalla suunnittelijoiden visioita lopullisesta tuotoksesta. Ollessani vastuussa teknisestä toteutuksesta, oli minun saneltava kriteerit ja rajanveto kokemukseni mukaisen toteutuksen mahdollistamiseen.

Kokoustamisesta seuraava vaihe oli tutkia nykyisen selaintarjonnan mahdollistamat toteutustavat. Tutustuin erilaisiin selaimille tarkoitettuihin AR-toteutuksia mahdollistaviin rajapintoihin ja työkaluihin. Näihin lukeutuivat esimerkiks Three.js:n päälle Googlen rakentama ilmainen three-js-ar -rajapintalaajennus, AR.js -kirjasto, ARToolKit.js ja muut AR-työkalut. Hyvin nopeasti ilmeni, että aikaisemmin kaavailtu aito web-selaimessa toimiva AR-toteutus ei tulisi olemaan mahdollinen nykyisillä selaimilla. Googlen kehittämä three-js-ar -rajapintalaajennus Three.js:n päälle on lähin asian mahdollistava työkalu lähitulevaisuudessa AR-teknologiaa sisältäviin web-toteutuksiin. Tarkemmalla tutkimisella selvisi, että vaikka käytännössä työkalu on täysin toimintakykyinen nykyisellään, puuttuu mobiiliselainten natiiviominaisuuksista kyky käyttää mobiililaitteiden kameraa AR-tarkoituksiin.

Samalla tutustuin myös jo olemassa oleviin vastaaviin AR-toteutuksiin, joista lähimmäksi suunnittelemaamme konseptia osui IKEA:n kehittämä IKEA Place -sovellus, jolla käyttäjä pystyy tarkastelemaan mobiililaitteensa ruudun avulla IKEA:n huonekaluja niille suunnittelluilla paikoilla omassa huoneistossaan reaaliaikaisesti. Sovelluksessa pystyy siis kokeilemaan, miltä IKEA:n huonekalu-katalogin tuotteet näyttäisivät paikoillaan ennen ostopäätöksen tekemistä ja näin edesauttaa asiakkaan ostokokemusta jo ennen IKEA:n liikkeeseen astumista. IKEA Place on saatavilla ilmaiseksi Android- ja Apple iOS -käyttöjärjestelmille valmistajien sovelluskaupoista. Rajoittavaksi tekijäksi muodostuu kuitenkin vaatimus AR-teknologiaa tukevan mobiililaitteen omistamisesta. Tuettuihin mobiililaitteisiin eivät vielä lukeudu kaikki modernit muutaman viime vuoden aikana julkaistut älypuhelimet.

Lisätyn todellisuuden toteutukset vaativat mobiililaitteilla sovelluksen natiiviapplikaatioksi paketoimisen sekä lisäksi Androidille ARCore-lisäosan ja Applen tuotteille ARKit-lisäosan asentamisen.

Todettuamme aidon AR-toteutuksen olevan mahdoton, muotoiltiin projektin tavotteita uudelleen. Käytännössä tämä tarkoitti sitä, että toteutettavasta web-sovelluksesta jätettiin kokonaan oikea AR-ulottuvuus pois, sen teknisten resurssien vaatiman määrän takia ja tilalle kehitettiin AR-toteutuksia mukaileva tapa näyttää 3D-malli käytettävän laitteen kameralla otetun valokuvan päällä. Vaikka varsinaisen AR-toteutuksen tekeminen ei olisi lainkaan ollut teknisesti mahdotonta, päätettiin pitäytyä suunnitellussa web-ympäristöön toteutettavassa toteutuksessa, kun kyse oli vasta proof-of-concept-tason tuotoksesta. Natiiviapplikaation toteutukseen vaadittavat resurssit ja välineet eroavat kuitenkin huomattavasti web-toteutukseen vaadittavista resursseista.

Vaihtoehtoiseksi toteutustavaksi pohdittiin myös AR.js -kirjaston mahdollistamaa tapaa, jossa sovelluksen apuna olisi toiminut fyysinen paperiarkki, jonka pinnalle tulostettu mustavalkoinen kuvio olisi toiminut sovelluksen kiintopisteenä 3D-mallin sijainnille. Luovuimme ideasta hyvin nopeasti todettuamme, että 3D-mallina käytettävät talomallit ovat fyysisessä muodossaan niin kookkaita, että todentuntuisen näkymän saavuttaminen muodostuisi hyvin ongelmalliseksi. Lisäksi haasteeksi muodostuisi rakennuksien suuren koon aiheuttama ongelma, jossa käyttäjä ei pääsisi tarpeeksi kauas paperiarkin päälle visualisoidusta mallista ennen kuin sovellus ei enää kykenisi paperille tulostettua merkkiä tunnistamaan ympäristöstään.

Käytännössä toteutetussa proof-of-concept web-sovelluksessa sivun auetessa eteen aukeaa näkymä, jossa on käyttäjälle mahdollisuus lisätä itse otettu kuva sovellukseen. Kuvan olisi tarkoitus olla tyhjästä tontista, johon käyttäjä voisi sijoittaa haluamansa rakennuskatalogin mukaisen 3D-mallin nähdäkseen vastaisiko se läheskään suunniteltua näkemystä rakennuksesta sen oikealla paikalla. Vaihtoehtoisesti työpöytäselaimella valokuvan voi valita jo ennestään tietokoneelta löytyvistä valokuvista toteutuksen ollessa alustariippumaton. Teknisesti kuvan lisääminen sovellukseen tapahtuu HTML:n *input*-kenttää hyödyntäen, joka avaa mobiililaitteilla automaattisesti puhelimen käyttöjärjestelmän natiivivalikon ja kysyy haluttua kuvan toimittamistapaa. Toimitustapoja on joko kännykän sisäisen kameran käyttäminen kuvan ottamiseen sovelluksen käyttöhetkellä tai jo ennestään löytyvän kuvan valitseminen laitteen muistista.

Oikeiden AR-ominaisuuksien jäädessä pois keskityttiin toteutuksessa muodostamaan selkeä ja helppokäyttöinen web-toteutus, joka toimisi mahdollisimman monella laitteella eli jopa hiukan heikommat tehot omaavilla mobiililaitteilla.

7.2 Haasteet

Ensimmäisenä projektin toteutuksesssa vastaan tulleena haasteena voidaan katsoa suunnittelutiimin haaveileman lopputuloksen mahdottomuus toteutukseen suoduilla resursseilla. Vaikka lähtökohdat projektiin olivat toteutuksen kannalta hyvin kokeilumieliset jo sen aloituksesta alkaen, ei suunniteltuun lopputulokseen päästy aivan siinä mielessä kuin olisi toivottu aidon AR-toteutuksen implementoinnin sovellukseen jäädessä pois. AR-ominaisuuksien pois jättäminen laajensi mahdollista käyttäjäkuntaa laajemman laitekannan kuuluessa web-toteutuksen tukemaan piirin.

Toiseksi suurimmaksi haasteeksi voidaan katsoa käytettävyyden kannalta 3D-mallin paikalleen sijoittamiseen liittyvä käyttöliittymän suunnittelu ja toteutus. Kuvan ladattuaan käyttäjällä on mahdollisuus kääntää ja sijoittaa taloa mukaileva 3D-malli mahdollisimman luonnolliseen asentoon, jotta näkymästä saataisiin mahdollisimman autenttinen ja todenmukainen vaikutelma. Proof-of-concept toteutukseen käyttöliittymän toteutus tapahtui kolmannen osapuolen JavaScript-kirjastoa käyttäen, jolla sivuun luotiin hiirellä tai mobiililaitteen tapauksessa sormella liikuteltavat palkit talomallin kääntelyä varten. Tämä osoittautui kuitenkin mobiilissa hankalaksi. Vaihtoehtoiseksi ratkaisuksi pohdittiin erillisten

nappuloiden asettamista ruutuun, vaiheistettua asettelua jolloin jokainen akseli asetellaan kohdilleen yksittäisessä vaiheessa nappuloita käyttäen tai *pinch-to-zoom* -menetelmää, jossa hyödynnetään mobiililaitteiden usean interaktion yhtäaikaista tallentamista

Vaikka saimmekin kaikista talomalleista 3D-mallit valmiina, ei niitä pystynyt käyttämään projektin toteutuksessa suoraan. Tämä johtui niiden sisältämistä ylimääräisistä taustaelementeistä, jotka jouduin ohjelmoinnin lomassa siivoamaan itse käyttäen ilmaiseksi netistä saatavaa Blender-mallinnusohjelmaa. Myös mallien keskipiste tuli kohdistaa Blenderiä käyttäen uudestaan keskelle 3D-malleja.

Ohjelmointi projektin toteutuksen kannalta oli hyvin suoraviivaista ja haasteetonta. Sovellus toteutettiin Crasmanin omaan CMS-sisällönhallintajärjestelmään (*Content Management System*), joka mahdollisti yhtäaikaisen testauksen työpöytätietokoneella sekä mobiililaitteilla. Näin varmistuttiin pääasiallisen toimintaympäristön, eli mobiililaitteiden yhteensopivuudesta sovelluksen kanssa.

7.3 Projektin eteneminen jatkossa

Projektin lopputulemana oli proof-of-concept -tyyppinen sovellus, jota näyttämällä lopullinen tuotos voidaan todentaa varmuudella toimivaksi. Näin saatiin lopullista ja valmista versiota varten Crasman Oy:lle resursointiin selkeämpi kuva vaaditusta työmäärästä ja mahdollisuuksista vastaaviin toteutuksiin. Tuotos toimikin pilottiprojektina yrityksen sisäisesti.

8 Pohdintaa

Esiteltyjen teknologioiden pohjalta voidaan todeta web:in tarjoavan ympäristönä mielenkiintoa ja mahdollisuuksia herättävän alustan tuottaa uudenlaista sisältöä. Tapamme käyttää internettiä on muuttunut tiedonhaun ja sähköpostin pääasiallisesta käytöstä yhä hektisemmäksi ja kiinteämmäksi osaksi päivittäistä elämäämme. Sisällön kulutukseen liittyvien tapojen muuttuessa jatkuvasti, on web-sivujen visuaalisella ulosannilla yhä suurempi merkitys käyttäjän huomion säilyttämisessä.

Lisätyn todellisuuden ja virtuaalitodellisuuden myötä pyritään digitaalisesti tuotettua sisältöä tuomaan ulos kaksiulotteisena pintana toimivan ruudun ulkopuolelle. Utopistisena ajatuksena voidaan miettiä tietokonesovellusten alustariippuvaisten rajojen särkyvän hiljalleen ja käytön tulevan yhä lähemmäksi käyttäjää erilaisin lisätyökaluin. Näihin työkaluihin voisi lukeutua esimerkiksi älylasit, jonka hyödyntämä lisätty todellisuus hämärtäisi oikean maailman ja tietokoneella luodun maailman välistä eroa tuomalla keinotekoisesti tuotetun grafiikan osaksi näkemäämme ympäristöä.

Keräämäni tiedon pohjalta voin todeta selainympäristöön tarkoitetun 3D-grafiikan työstämiseen vaadittujen työkalujen olevan hyvin ajan tasalla ja tarjoavan hyvän pohjan kirjavien web-toteutuksien toteuttamiseen. On kuitenkin otettava huomioon tarjolla olevien teknologioiden jatkuva kehityskaari. Vaikka tällä hetkellä esimerkiksi lisätty todellisuus ja virtuaalitodellisuus ovat mahdottomissa toteuttaa juuri web-ympäristöön on oletettavissa niiden toteutuskelpoisuus ja teknologian esiinmarssi lähitulevaisuudessa. Uskoakseni emme tule näkemään äkillistä virtuaalitodellisuuden ja lisätyn todellisuuden siirtymistä web-ympäristöön näiden vaatiman prosessointitehon kasvaessa sitä myöten mitä monimutkaisemmaksi toteutus etenee. Sen sijaan uskon vakaasti näkevämme kevyempiä web-ympäristöön toteutettuja näitä teknologioita hyödyntäviä sovelluksia, esimerkiksi pelillistämisen ja mainonnan välistä eroa hämärtävässä muodossa.

Opinnäytetyötäni varten syventyessäni erilaisiin WebGL:ää sisältäviin toteutuksiin tulin todenneeksi 3D-grafiikan hyödyntämisen tarpeellisuuden olevan hyvin häilyvissä rajoissa. Törmäsin useisiin toteutuksiin joissa 3D-grafiikka tarjosi lähinnä esteettisen ilmeen tueksi silmäkarkkia ilman suurempaa informaatiosisältöä. Onkin todettava ettei WebGL:n mahdollistama 3D-grafiikan hyödyntäminen tulisi olla itseisarvo, vaan tulisi sen palvella kokonaisuutta paremman informaation välittymisen kannalta. Erilaisia toteutuksia tehdessä tulisi siis asioita lähestyä ennemmin tavalla jossa mietitään informaation

esittämiselle optimaalisin toteutustapa. Tällä tavalla tulisi automaattisesti otetuksi huomioon suunnittelussa käyttäjälähtöisyys, joka ei läheskään aina toteudu runsaasti prosessointitehoja syövää 3D-grafiikka sisältävissä toteutuksissa.

Opinnäytetyötä työstäessäni ymmärrykseni 3D-grafiikan toteuttamisesta otti selkeän harppauksen eteenpäin yhtäaikaisesti graafiikkaa sisältävien tuotosten suunnittelua varten vaaditun tiedon ja esteettisen näkemyksen kanssa. Selkein huomio kokonaisuuden suhteen oli internetistä löytyvän 3D-grafiikan huono kytkeytyminen sitä kannattelevaan muuhun sivuun. Tästä voidaankin vetää johtopäätös siitä ettei kolmiulotteisia graafisia elementtejä hyödynnetä nykyisellään täysin niiden suoman potentiaalin mukaisesti, joka vastaakin täysin ennakkokäsitystäni asiasta. Koenkin web:in olevan alustana jatkuvassa murrostilassa, jossa alan standardit ja trendit vaihtuvat päivittäin. Kolmiulotteisen grafiikan hyödyntäminen kuitenkin ei kokemukseni mukaan ole noussut missään vaiheessa trendimäiseksi ilmiöksi sen potentiaalista huolimatta. Nähtäväksi jääkin ottaako WebGL vakiintuneesti asemansa web-kehittäjien työkaluna informaation suorasukaisempaan välittämiseen.

Lähteet

Bonnington Christina 13.3.2015. You can now watch and upload 360-degree videos on YouTube. Wired. https://www.wired.com/2015/03/youtube-360-degree-video/. (luettu 23.10.2018)

Branch John, The Snow Fall. The New York Times 2012. http://www.nytimes.com/projects/2012/snow-fall/index.html#/?part=tunnel-creek. (luettu 23.10.2018)

Bright Peter 29.10.2014. HTML5 specification finalized, squabbling over specs continues, Ars Technica. https://arstechnica.com/information-technology/2014/10/html5-specification-finalized-squabbling-over-who-writes-the-specs-continues/. (luettu 15.10.2018)

Caniuse.com WebGL 2018. https://caniuse.com/ (luettu 20.10.2018)

Etienne Jerome, AR.js 2018. https://github.com/jeromeetienne/AR.js/blob/master/README.md. (luettu 12.11.2018)

Gonzalez Vivo Patrizio 2018. The Book of Shaders. https://thebookofshaders.com/. (luettu 12.11.2018)

Mozilla Foundation 2018. https://developer.mozilla.org/fi/docs/Web/API/WebGL_API (luettu 12.10.2018)

Mozilla Foundation 2018. https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/HTML5. (luettu 15.10.2018)

O'Kane Sean 23.8.2017. Facebook now lets you shoot 360-degree photos inside its app. The Verge. https://www.theverge.com/2017/8/23/16190584/facebook-360-degree-photos-app-camera. (luettu 1.10.2018)

PC Mag Encyclopedia 2018. https://www.pcmag.com/encyclopedia/term/43886/gpu. (luettu 12.11.2018)

Perez Sarah 2018. ARKit-only apps top 13 million installs, nearly half from games. Tech Crunch. https://techcrunch.com/2018/03/28/arkit-only-apps-top-13-million-installs-nearly-half-are-games/. (luettu 12.11.2018)

Robertson Adi 11.10.2018. Facebook will let you post '3D photos' in your News Feed, The Verge. https://www.theverge.com/2018/10/11/17964686/facebook-3d-photos-dual-camera-depth-pictures-rollout. (luettu 12.11.2018)

RuneScape Wiki, Jagex 2018. https://runescape.wiki. (luettu 26.10.2018)

StatCounter.com 2018. *http://gs.statcounter.com/platform-market-share/desktop-mobile-tablet*. (luettu 1.10.2018)

The Khronos Group 2018. https://www.khronos.org/webgl/. (luettu 12.10.2018)

The Khronos Group 2018. https://www.opengl.org/about/ (luettu 12.11.2018)

Three.js 2018. https://threejs.org/docs/. (luettu 1.10.2018)

Three-AR.js 2018 https://github.com/google-ar/three.ar.js. (luettu 28.9.2018)

WebGLStats.com 2018. (luettu 20.10.2018)

W3Counter, Awio Web Services LLC 2018. *https://www.w3counter.com/global-stats.php*. (luettu 25.9.2018)

W3Counter, Awio Web Services LLC 2018. https://www.w3counter.com/global-stats.php?year=2018&month=08. (luettu 25.9.2018)

W3C 2000. https://www.w3schools.com/browsers/. (luettu 13.11.2018)