LAUREA
AMMATTIKORKEAKOULU
Yhdessä enemmän

Eleohjaustoiminnallisuuksien kehittäminen tv-grafiikkajärjestelmään

Piiroinen, Timo

2017 Laurea

Laurea-ammattikorkeakoulu

Eleohjaustoiminnallisuuksien kehittäminen tv-grafiikkajärjestelmään

Piiroinen Timo
Tietojenkäsittelyn koulutusohjelma
Opinnäytetyö
Huhtikuu, 2017

Laurea-ammattikorkeakoulu **Tiivistelmä**
Tietojenkäsittelyn koulutusohjelma
Tradenomi

Timo Piiroinen

Eleohjaustoiminnallisuuksien kehittäminen tv-grafiikkajärjestelmään

Vuosi 2017 Sivumäärä 35

Tämän opinnäytetyön aiheena oli tutkia eleohjaimen käyttöä tv-grafiikkasovelluksen ohjaamisessa. Kehittämistyö toteutettiin Maisti Oy:n tarpeesta kehittää uudenlaisia interaktiivisia grafiikanohjausmenetelmiä. Tutkimuksessa oli tavoitteena kehittää eleohjaustoiminnallisuudet CasparCG-nimiselle tv-grafiikkasovellukselle ja tuottaa tietoa eleohjauksen kehittämisestä Maisti Oy:n tuotekehitysprosessin tehostamiseksi. Tutkimuksessa selvitettiin myös Leap Motion -ohjaimen toimintaa tv-studiossa.

Opinnäytetyön teoriaosa käsittelee tv-grafiikkaa, eleohjauksen teoriaa ja Leap Motion -eleohjaimen sovelluskehitystä. Työn empiirinen osuus käsittelee eleohjauksen tutkimusta ja kehittämistä CasparCG-ohjelmistolle.

Tutkimus toteutettiin laadullisen tutkimusmenetelmän keinoin. Tutkimuksessa käytettiin tiedonkeruumenetelminä havainnointia ja avointa haastattelua. Havainnointi suoritettiin osallistuvana havainnointina kehittämisen aikana ja tarkkailevana havainnointina testauksen aikana. Avoin haastattelu suoritettiin testaustilanteessa havainnoinnin jälkeen. Haastattelulla pyrittiin todentamaan havainnot oikeiksi ja keräämään tietoa eleohjaimen käytöstä.

Eleohjaustoimintojen kehittäminen aloitettiin käyttötapausten määrittelyllä, joiden perusteella tehtiin vaatimusmäärittely. Vaatimusmäärittely sisälsi sellaiset eleohjaustoiminnot, jotka kehitetyllä järjestelmällä tuli voida suorittaa. Kehittäminen toteutettiin vaiheittain, missä jokainen vaihe sisälsi kehittämisen ja testauksen.

Eleohjaustoiminnallisuudet CapsarCG-ohjelmistolle kehitettiin yhdistämällä teoria kehittämisen ja testien aikaisiin havaintoihin ja haastatteluissa ilmaantuneeseen tietoon. Näin saatiin myös tietoa, jota voidaan hyödyntää tulevaisuudessa toimeksiantajan tuotekehityksessä.

Asiasanat: eleohjaus, luonnollinen käyttöliittymä, tv-grafiikka

Laurea University of Applied Sciences **Abstract**
Degree Programme in Business Information Technology
Bachelor's Thesis

Timo Piiroinen

Gesture control for video and graphics playout system

Year 2017 Pages 35

This Bachelor's thesis deals with gesture control for video and graphics playout system called CasparCG. The goal of the study was to develop gesture control functions for CasparCG and gather information about gesture control for the commissioner's future product development. The commissioner of this thesis was a startup company called Maisti Oy.

The theoretical section discusses broadcast graphics, gesture control and developing Leap Motion application. The empirical part deals with researching and developing gesture control for CasparCG. The thesis was based on qualitative methods. The data was collected with interviews and observation during the development and during the testing.

The developing was started by creating use cases and defining the gesture control requirements for the system. The development was made in stages, where each stage consisted of developing and testing phases.

Gesture control functions for CasparCG were created by combining theory and data based on observation and interviews. This thesis also provides information for designing and developing gesture control systems which can be used in future development projects in Maisti Oy.

Keywords: gesture control, natural user interface, broadcast graphics

Sisällys

1 Johdanto

Eleohjaus tekee ihmisen ja tietokoneen välisestä kommunikoinnista ihmiselle luonnollisempaa. Tarkkailemalla ihmisen toimintaa erilaisten sensorien avulla voidaan ohjaus toteuttaa luonnollisen käyttöliittymän avulla, jossa sovellusta ohjataan ihmisen fyysisillä ominaisuuksilla osoitinlaitteiden sijaan. Eleohjauksen käyttäminen järjestelmien ohjauksessa yleistyi 2000-luvun puolivälin jälkeen pelikonsolien ja älypuhelimien kehityksen myötä.

Kehittämistutkimus toteutettiin toimeksiantajayrityksen tavoitteista kehittää eleohjaustoiminnallisudet tv-grafiikan ja videonäyttöjen ohjaamiseen. Kehittämistutkimuksen tavoitteena oli kehittää eleohjausjärjestelmä ja tuottaa tietoa yrityksen tuotekehitykseen. Kehitettävälle järjestelmälle asetettiin vaatimuksena sen toteuttaminen yrityksellä olemassa olevien teknologioiden avulla.

Kehittämistutkimus toteutettiin laadullisen tutkimusmenetelmän keinoin. Käytetyt tiedonhankintamenetelmät olivat havainnointi ja avoin haastattelu. Menetelmien avulla pyrittiin selvittämään tekniikan soveltuvuus uuteen käyttöympäristöön sekä kehittää järjestelmää havaintoihin perustuen. Projektin aikana kehittäminen ja tutkimus toteutettiin vaiheittain. Jokaisessa vaiheessa kehitettiin toiminnallisuus, joka testattiin ennen järjestelmän kehittämisen jatkamista.

Opinnäytetyön teoriaosuus käsittelee tv-grafiikkaa ja eleohjauksen suunnittelua. Esitettyä teoriaa käytettiin apuna eleohjausjärjestelmän suunnittelussa. Opinnäytetyössä kuvataan tutkimusprosessi sekä kehitetyn järjestelmän eletoiminnallisuuksien kehittäminen.

2 Kehittämistyön lähtökohdat

Tämän kehittämistyön toimeksianto tuli työpaikaltani, kun vuoden 2016 syksyllä yrityksessä ryhdyttiin tutkimaan edullisia ja uudenlaisia tapoja tuottaa grafiikkaa tv-lähetyksiin. Tarkoitus oli kehittää erilaisia teknisiä ratkaisuja, joiden avulla yhteistyökumppanit erottuvat kilpailijoistaan. Kehittämisen lähtökohtana oli ratkaista tilanne, jossa ollaan tekemisissä niin isojen videonäyttöjen kanssa, että kosketusnäyttö ratkaisut eivät ole sopivia vaihtoehtoja. Päädyttiin testaamaan eleohjaimen käyttöä grafiikan ohjauksessa. Syntyi tarve tutkia eleohjausta ja kehittää uusi tuote yrityksen käyttöön. Kehitykseen osallistui myös ohjelmoijana toiminut yrityksen toinen työntekijä.

Tutkimuksessa kehittämiseen tarvittavat tekniikat, jotka tässä tapauksessa olivat eleohjain ja tv-grafiikkalaite, olivat jo yrityksellä käytössä. Laitteita oli käytetty aikaisemmin muihin

tarkoituksiin ja nyt tuli ensimmäinen kerta, kun ne yhdistettäisiin ja eleohjain vietäisiin uuteen käyttöympäristöön.

Kehittämistyön toimeksiantaja Maisti Oy on vuonna 2016 perustettu yritys, jonka palveluita ovat reaaliaikaisen grafiikan ja olemassa olevan tiedon visuaalinen esittäminen suorissa televisio lähetyksissä ja tapahtumissa. Lisäksi yritys tarjoaa järjestelmiä ja räätälöityjä käyttöliittymiä edellä mainittuihin tarkoituksiin. Yrityksen tavoitteena on tuottaa uudenlaisia ratkaisuja tiedon esittämiseen, joiden avulla luodaan erottuvia käyttökokemuksia, toiminnan pysyessä helppona, nopeana ja luotettavana.

2.1 Kehittämistyön tavoitteet ja rajaukset

Kehittämistyön ensisijaisena tavoitteena oli tuottaa tutkimuksen toimeksiantajalle uusi tuotekonsepti ja kehittää toiminnallisuudet tv-grafiikan ohjaamiseen eleohjaimen avulla. Lisäksi tutkittiin, miten tutkimuksessa käytettävä eleohjain soveltuu kyseiseen käyttötarkoitukseen. Laitteisto ja toiminnallisuudet testattiin niiden oikeassa toimintaympäristössä. Testaamalla eleohjain tv-studiossa saatiin mahdollisimman oikea käsitys sen toiminnasta, sekä muiden laitteiden, kuten valojen ja kameroiden mahdollisesti aiheuttamista ongelmista.

Toisena tavoitteena oli tuottaa toimeksiantajan tuotekehitykseen uutta tietoa eleohjauksesta ja sen suunnittelusta. Opinnäytetyön teoreettinen viitekehys käsittelee luonnollisen käyttöliittymän teoriaa ja eleohjauksen suunnittelua olemassa olevaan tietoon ja aikaisempiin tutkimuksiin perustuen. Näin pyritään saamaan mahdollisimman hyvä tietoperusta tutkimukselle ja kehitettävälle eleohjausjärjestelmälle. Yhdistämällä olemassa oleva tieto tutkimuksessa esille tulevaan tietoon, pyritään saamaan mahdollisimman kattava tietopaketti yrityksen käyttöön tuleviin tuotekehitysprojekteihin.

Kehittämistyö rajattiin tutkimaan eleohjauksen toimintaa tv-grafiikan ohjauksessa tutkimuksessa käytettävän eleohjaimen ja tv-grafiikkalaitteen avulla. Kehitetyt toiminnallisuudet kuvataan, jotta voidaan selittää erilaiset ratkaisut teoriaan ja havainnointiin perustuen. Tutkimuksessa ei vertailla erilaisia eleohjaimia ja tv-grafiikkasovelluksia.

2.2 Keskeiset käsitteet

HTML on lyhenne sanoista Hyper Text Markup Language. Se on verkkosivujen sisällön rakenteen kuvaamiseen käytettävä merkintäkieli, jonka avulla selain luo näkymän sivusta. (w3schools 2017a.)

Cascading Style Sheets (CSS) on HTML-elementtien tyylien määrittämiseen käytettävä kieli. Sen avulla määritetään miltä HTML-tiedosto näyttää eri medioissa. (w3schools 2017b.)

JavaScript on yleinen ohjelmointikieli, jonka avulla voidaan luoda vuorovaikutteista toiminnallisuutta verkkosivuille (Mozilla 2017).

JQuery on JavaScript-kirjasto, joka helpottaa ja nopeuttaa JavaScript-ohjelmointia verkkosivuilla. JQueryn sisäisten funktioiden avulla usean rivin JavaScript-toiminnot voidaan suorittaa yhden rivin mittaisella koodilla. (w3schools 2017c.)

GreenSock Animation Platform (GSAP) on JavaScript-animaatiokirjasto. Sen avulla voidaan luoda animaatioita, jossa HTML-elementit animoituvat aikajanalla järjestyksessä tai ne voidaan asettaa alkamaan samanaikaisesti tai viiveellä. (GreenSock 2017.)

JSON lyhenne tulee sanoista JavaScript Object Notation. Se on tiedon tallennukseen ja vaihtamiseen käytettävä tiedostomuoto. JSON on teksti muotoista tietoa ja sitä voidaan lukea ja käyttää missä tahansa ohjelmointikielessä. (w3schools 2017d.)

WebSocket on protokolla, jonka avulla muodostetaan jatkuva yhteys asiakas- ja palvelinohjelmiston välille. WebSocket välittää tiedon viestimuotoisena ja sen avulla molemmat ohjelmistot voivat lähettää tietoa milloin tahansa. (West 2013.)

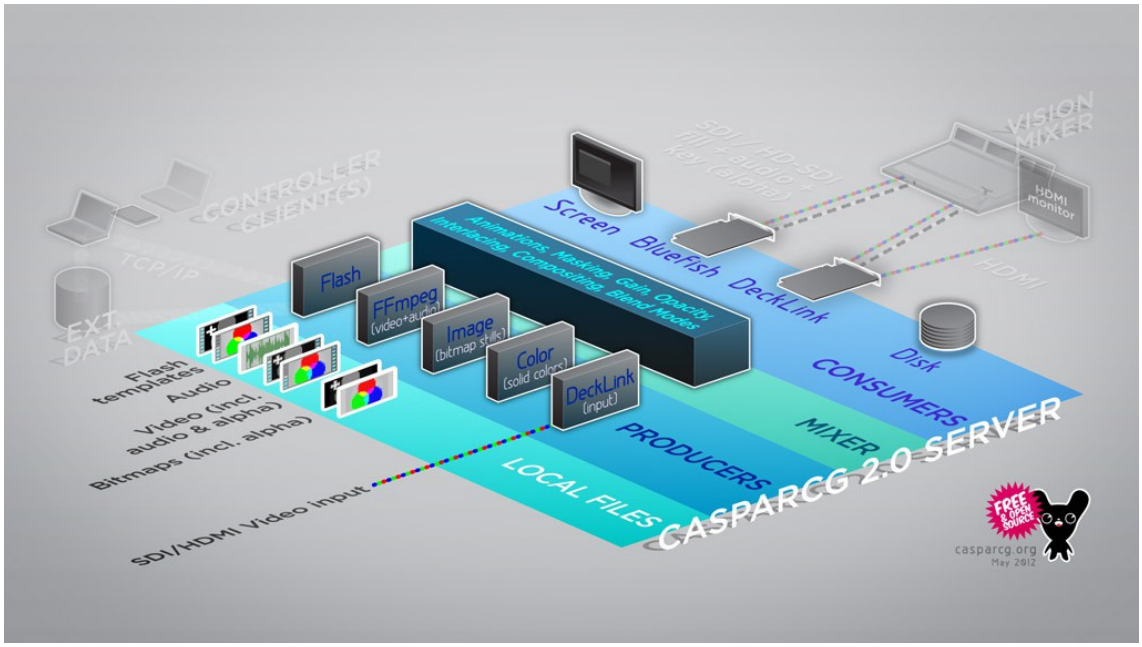
Avoin lähdekoodi tarkoittaa tapaa kehittää ja jakaa tietokoneohjelmistoja. Avoimen lähdekoodin ohjelmistot ovat toimittajariippumattomia ja niiden toteutukset ovat kaikkien käytettävissä. Avoimen lähdekoodin ohjelmille ei ole standardoitua määritelmää, mutta niille on asetettu erilaisia vaatimuksia jotka tulisi täyttää. (Coss 2017.)

Ohjelmointirajapinta määrittelee miten muut sovellukset ja tietojärjestelmät saavat ohjelman tiedot ja palvelut käyttöönsä. Rajapinta voi olla datarajapinta, josta luetaan tietoa toiseen järjestelmään tai se voi olla toiminnallinen rajapinta, jonka kautta saadaan käyttöön laskenta-algoritmeja ja voidaan muuttaa järjestelmän tietoja. (Avoinrajapinta 2014.)

3 TV-grafiikka

TV-grafiikalla tarkoitetaan televisiolähetyksissä videokuvan päällä näytettäviä erilaisia graafisia elementtejä, jotka ovat esimerkiksi urheiluohjelmissa näkyvät pelikellot ja tulokset sekä uutisissa käytettävät henkilöiden nimet, aiheotsikot ja säätiedotukset. Jokaisella grafiikalla on ohjelmassa oma tehtävänsä. Grafiikan tarkoitus on vahvistaa ohjelman välittämää viestiä katsojalle, luoda ohjelmalle tyyli ja tunnelma, joka ohjelmalla halutaan välittää sekä helpottaa katsojan ymmärrystä ohjelman kulusta ja sisällöstä. Ohjelman alkutunnuksella pyritään kiinnittämään katsojan huomio alkavaan ohjelmaan ja vastaavasti lopputunnus ilmoittaa katsojalle ohjelman päättymisen. Suunniteltaessa grafiikkaa tv-lähetykseen tulee miettiä vastaukset seuraaviin kysymyksiin: Miten kyseinen grafiikka helpottaa katsojan ymmärrystä aiheesta? Mikä on grafiikan tarkoitus? Toimiiko käytettävät elementit siihen tarkoitukseen, johon niitä aiotaan käyttää? (Millerson & Owens 2008, 275, 276.)

TV-tuotannossa grafiikka ja videokuva yhdistetään yhdeksi signaaliksi videomikserin avulla. Kuviossa 1 esitetään grafiikkalaitteen signaalien kytkentä videokortilta videomikseriin. Videomikseri asettaa grafiikan videokuvan päälle täyden ruudun kokoisena. Tämän vuoksi grafiikka täytyy avaintaa (eng. Keying). Avaintamisella tehdään kuvan ne osat läpinäkyviksi, joiden ei ole tarkoitus näkyä. Avaintamisessa kuvasta tehdään mustavalkoinen versio, jossa valkoiset kohdat ovat täysin näkyviä, mustat ovat täysin näkymättömiä ja harmaan eri sävyt ovat osittain läpinäkyviä. (Weston 2009, 2.)



Kuvio 1: Grafiikkalaitteen signaalien kytkentä videomikseriin (CasparCG 2013).

Avaintaminen voidaan tehdä kolmella eri tavalla, joita ovat Chroma Key, Luma Key ja Linear Key. Chroma Key on menetelmä, jossa kuvan läpinäkyvyys luodaan jonkun värin perusteella. Yleisimmin käytettävät värit ovat sininen ja vihreä. Väri on tärkeää huomioida grafiikan suunnittelussa. Avaintamiseen käytettävää väriä ei voi käyttää grafiikka elementeissä, sillä myös niistä tulee läpinäkyviä. Luma Key tekee läpinäkyvyyden kuvan kirkkauden perusteella, joten kuvan värien tulee olla taustavideota kirkkampia. Tämän vuoksi grafiikoissa olevat suuret värien kirkkaus erot voivat aiheuttaa ongelmia, jolloin grafiikka ei näytä sellaiselta kuin sen on tarkoitus näyttää. Linear Key on menetelmä, jossa videomikserille lähetetään grafiikkakoneelta kaksi signaalia. Toinen on grafiikan täyttökuva ja toinen on mustavalkoinen versio samasta kuvasta. Videomikseri yhdistää kuvat ja tekee läpinäkyvyyden mustavalkokuvan perusteella. Tätä menetelmää käytettäessä voidaan käyttää kaikkia värejä huoletta, koska läpinäkyvyys ei perustu täyttösignaaliin, eli kuvaan joka näkyy videon päällä. (Weston 2009, 2-4.)

Ohjelman aikana videokuvan päällä näytettävät grafiikat tehdään grafiikkapohjiksi, joissa toiminnallisuus pysyy vakiona, mutta sisällön voi vaihtaa helposti. Tällä pyritään reaaliaikaiseen työskentelyyn, joka on suorissa tv-lähetyksissä ehdotonta. Sisältö voi olla esimerkiksi aiheeseen liittyviä kuvia ja tekstiä, jotka vaihtuvat grafiikassa ohjelman ajankohdasta riippuen. Grafiikkapohjista luodaan kaikki kyseisessä tv-ohjelmassa tarvittavat versiot grafiikkakoneen (eng. Character Generator) ohjaussovelluksen ajolistalle, josta ne voidaan näyttää ohjelmassa haluttuna aikana. Character Generator on yleinen määritelmä laitteistolle, jolla voidaan näyttää grafiikkaa televisiolähetyksissä (Millerson ym. 2008, 278).

4 CasparCG

Tässä opinnäytetyössä käytettiin CasparCG-nimistä grafiikkaohjelmistoa. CasparCG on Ruotsin yleisradioyhtiön SVT:n kehittämä avoimen lähdekoodin videon- ja grafiikantoistojärjestelmä tv-tuotanto käyttöön (Nagy 2014). CasparCG sisältää HTML-moduulin, joka mahdollistaa grafiikkapohjien luomisen HTML-kuvauskielen avulla. Lisäksi se mahdollistaa grafiikoiden animaatioiden tekemisen JavaScript-ohjelmointikielen ja CSS:n avulla. (Karlsson 2014.)

CasparCG toimii asiakas-palvelin-mallin mukaisesti. Asiakas-palvelin-mallissa järjestelmä koostuu asiakas- ja palvelinohjelmistoista ja malli toimii niin, että asiakasohjelmisto käyttää palvelimen tarjoamia palveluita (Vihavainen 2014). CasparCG sovelluksen verkkosivuilla on ladattavissa palvelinohjelmisto ja SVT:n kehittämä asiakasohjelmisto, jolla voidaan ohjata palvelinta. CasparCG käyttää asiakas- ja palvelinohjelmistojen väliseen kommunikointiin kahta eri protokollaa, jotka ovat Advanced Media Control Protocol (AMCP) ja Open Sound Control (OSC) (CasparCG 2016).

AMCP-protokolla on SVT:n kehittämä kommunikointi protokolla CasparCG-palvelimen ohjaukseen, jossa viestit ovat UTF-8 koodatussa tekstimuodossa (Nagy 2014). OSC-protokollan välityksellä palvelin lähettää tilatietoja toiminnastaan asiakasohjelmistolle (CasparCG 2016). OSC-protokolla on viestintäprotokolla tietokoneille, äänisyntetisaattoreille sekä erilaisille multimedialaitteille, jotka ovat varustettu nykyaikaisella verkkotekniikalla (Open Sound Control 2011).

5 Eleohjaus

Eleohjaus on teknologian kehityksen myötä syntyneiden uusien sensoreiden ja laitteiden mahdollistamaa ihmisen ja teknologian välistä moniaistista vuorovaikutusta. Sen ansiosta ihmisen ja tietokoneen välinen kommunikointi muuttuu lähemmäs ihmiselle luonnollisempaa käyttäytymistä, jossa tietokonetta voidaan ohjata puheella, kosketuksilla, eleillä, ilmeillä, katseella tai jopa tunteilla. (Lappalainen, Rakkolainen, Korkalainen, Okkonen, Mäkilä, Pääkylä, Lehtonen & Lakkala 2015, 102.)

Eleohjaus yleistyi kuluttajien keskuudessa 2000-luvun puolivälin jälkeen, kun Nintendo julkaisi vuonna 2006 Wii-pelikonsolin, jota voidaan ohjata langattoman ohjaimen avulla. Vuotta myöhemmin Apple julkaisi kaksi eleohjattavaa tuotetta, kosketusnäytöillä ohjattavat iPod Touch sekä iPhone. Näiden tuotteiden myötä eleohjauksesta tuli kuluttajille arkipäiväinen tapa kommunikoida, mutta eleohjauksen historia on kuitenkin paljon pidemmällä kuin 2000-luvulla. Ensimmäinen kosketusnäytöllä ohjattava Simon-niminen matkapuhelin julkaistiin vuonna 1994 ja kosketusnäytöllä ohjattavia laitteita tehtiin jo 1970-luvun alussa. Vuonna 1975 Myron Krueger julkaisi järjestelmän, jota kutsuttiin nimellä Videoplace. Sitä pidetään ensimmäisenä ihmisen vartalon liikkeillä ohjattavana teknologian järjestelmänä. Siinä ihmiset pystyivät kommunikoimaan toistensa kanssa virtuaalisesti eri huoneissa. (Saffer 2009, 8-11.) Videoplace-järjestelmän ensimmäinen versio sisälsi videoprojektorin ja videokamerat, joiden avulla kuvat otettiin ja yhdistettiin. Myöhemmin järjestelmä kehittyi niin, että siinä voitiin yhdistää reaaliaikaisen videon muodossa ihminen ja ihmisen liikkeisiin reagoiva tietokone grafiikka. (Media Art Net 2005.)

Eleohjaukseen käytettävät eleet voivat olla mitä tahansa ihmisen tekemiä fyysisiä liikkeitä, jotka on mahdollista tunnistaa ja johon tietokone voi reagoida ilman perinteistä osoitinlaitetta (Saffer 2009, 2). Perinteisillä osoitinlaitteilla tarkoitetaan tässä tapauksessa näppäimistöä ja hiirtä. Eleohjauksessa nämä ovat korvattu ihmisen vartalon liikkeet tunnistavilla sensoreilla ja graafinen käyttöliittymä on vaihtunut luonnolliseen käyttöliittymään.

5.1 Luonnollinen käyttöliittymä

Eleohjausta käyttävissä järjestelmissä ohjaus tapahtuu luonnollisen käyttöliittymän (eng. Natural User Interface, NUI) kautta. Luonnollinen käyttöliittymä mahdollistaa uudenlaisen vuorovaikutuksen digitaalisten objektien kanssa. Sen avulla päästään lähemmäs tilannetta, jossa niitä voidaan käsitellä samalla tavalla kuin fyysisiäkin objekteja. (Kaushik & Raim 2014, 1.)

Graafisen käyttöliittymän ja luonnollisen käyttöliittymän yksi suuri ero on sovelluksen elementtien ohjaaminen. Perinteiset graafisen käyttöliittymän sovellukset ovat WIMP (window, icon, menu, pointing device) ja työpöytä sovelluksia, joissa elementtejä ohjataan hiiren ja näppäimistön avulla käyttöliittymän kautta. Luonnollisessa käyttöliittymässä ohjaus kohdistuu suoraan sovelluksen elementteihin. (Polaine 2009.) Esimerkiksi Windows tietokoneella, ohjauslaitteiden ollessa näppäimistö ja hiiri voidaan valokuvaa suurentaa käyttöliittymän liukusäätimen avulla, mutta kosketusnäytöllä varustetuissa laitteissa voidaan kuvaa suurentaa laittamalla etusormi ja peukalo kuvan päälle ja viemällä sormet kauemmaksi toisistaan.

Koska luonnollinen käyttöliittymä ei ole riippuvainen syötelaitteista, sen määritelmän tekeminen ei ole niin yksinkertaista kuin komentorivikäyttöliittymän tai graafisen käyttöliittymän määritelmä. Luonnollisessa käyttöliittymässä toiminnan tarkastelu kohdistuu ihmiseen ja hänen tekemiseen. Siinä pyritään saamaan ihmisen toiminta niin luonnolliseksi, että käyttöliittymä häviää. Tällä tarkoitetaan sitä, että ihminen ei sovellusta käyttäessään kiinnitä mitään huomiota sen käyttöliittymään. (Shamonsky 2014.) Luonnollisuudella tarkoitetaan tapaa, jolla henkilö käyttää sovellusta sekä mitä hän ajattelee ja tuntee käyttäessään sitä. Sen voidaan olettaa olevan sovelluksen erillinen ominaisuus. (Wigdor & Wixon 2011, 24.)

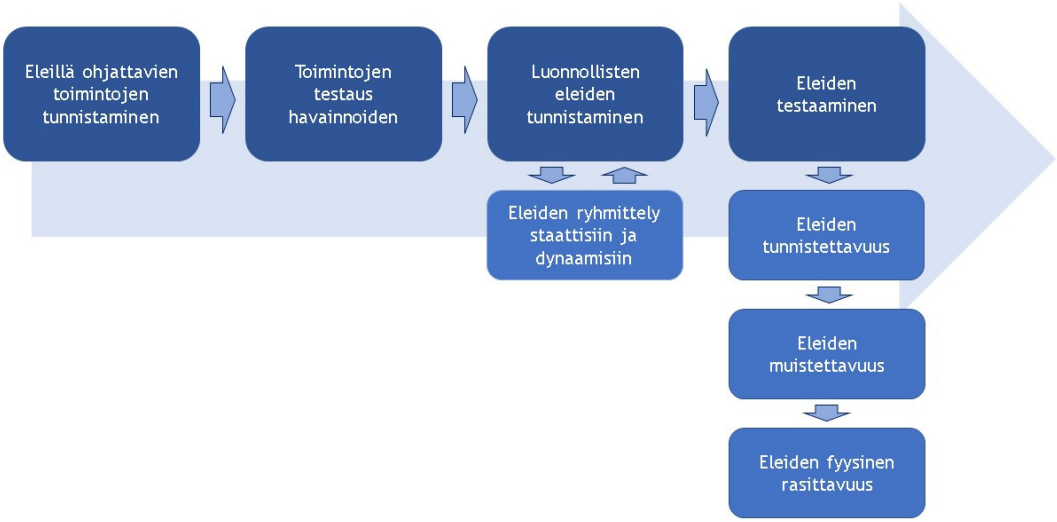
5.2 Eleohjauksen suunnittelu

Luonnollista käyttöliittymää ei voida käyttää missä tahansa ja mihin tahansa. Esimerkiksi kirjoittaminen on helpompaa ja nopeampaa näppäimistön avulla graafisen käyttöliittymän kautta. Luonnollinen käyttöliittymä tulee nähdä vaihtoehtona muiden käyttöliittymien ohella ja suunnittelun lähtökohtana tulee olla helpommin ja tehokkaammin käytettävä sovellus. Ei vain halu tehdä luonnollinen käyttöliittymä. Käytettävän käyttöliittymän valinnan tulee perustua sovelluksen käytön tarpeeseen. (Nielsen, Störring, Moeslund & Granum 2003, 1.)

Luonnollisen käyttöliittymän suunnittelussa pyritään tilanteeseen, jossa sovellusta käyttävä henkilö oppii toiminnallisuudet mahdollisimman nopeasti ja mieltää käytettävät eleet luonnollisiksi niille tarkoitettuihin toimintoihin. Yksi luonnollisen käyttöliittymän tärkeimmistä ominaisuuksista on sujuva ja saumaton toiminta, joka on myös tavoitteena suunnittelussa. Sovelluksen käytön sujuvuus vaikuttavaa tunteeseen, joka ihmiselle tulee hänen käyttäessään sovellusta sekä siihen kokeeko hän sovelluksen käytön luonnolliseksi. Nopealla oppimisella saadaan ihmiset itsevarmemmiksi ja saumattomuudella poistetaan mahdollisia epäilyjä käyttöliittymää kohtaan, jolloin sovelluksen käyttö on mielekkäämpää. (Wigdor ym. 2011, 42, 58, 61.)

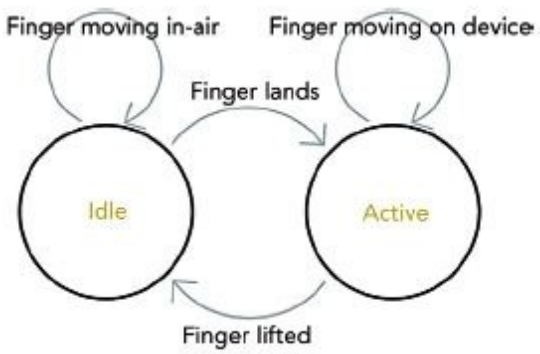
Eleohjauksessa käytettävien eleiden valintaa voidaan lähestyä kahdella eri tavalla. Nämä ovat konekeskeinen lähestymistapa ja ihmiskeskeinen lähestymistapa. Konekeskeinen lähestymistapa keskittyy eleisiin, jotka ovat teknisesti helposti tunnistettavissa. Ihmiskeskeinen lähestymistapa keskittyy ihmisten toimintaan pyrkimällä hyvään käytettävyyteen. Tähän tavoitteeseen pyritään noudattamalla käytettävyyden periaatteita, jotka ovat opittavuus, tehokkuus, muistettavuus, vähäinen virheprosentti ja kattavuus. (Nielsen ym. 2003, 2-3.)

Sopivien eleiden suunnittelussa ja valinnassa voidaan käyttää apuna neljästä vaiheesta koostuvaa menetelmää (Kuvio 2). Ensimmäisessä vaiheessa tunnistetaan ne sovelluksen toiminnot, joita eleillä halutaan ohjata. Toisessa vaiheessa suoritetaan toimintojen testaus. Testissä havainnoidaan ihmisiä kameran välityksellä heidän kuvaillessaan testin ohjaajalle, miten he suorittaisivat ensimmäisessä vaiheessa esille tulleet toiminnot. Kolmannessa vaiheessa etsitään ne eleet, joita testeissä henkilöt käyttivät luonnollisesti. Lisäksi määritetään, olivatko heidän käyttämänsä eleet staattisia vai dynaamisia. Staattisissa eleissä käsi ja sormet eivät ole liikkeessä. Dynaamisissa eleissä on kyse käden liikkeestä ja liikkeen ominaisuuksista, kuten nopeudesta tai liikeradasta. Neljännessä vaiheessa eleiden suunnittelussa eleet testataan kolmiosaisella testillä. Ensin testataan niiden tunnistettavuus yhdistämällä oikeat eleet ja toiminnallisuudet. Toiseksi testataan eleiden muistettavuus ja kolmanneksi niiden fyysinen rasittavuus. (Nielsen ym. 2003, 2, 6-7.)



Kuvio 2: Eleiden valintamenetelmäprosessin vaiheet.

Eleohjauksen kehittämisen helpottamiseksi voidaan käyttää erilaisia ratkaisuja järjestelmän suunnittelussa. Eleohjain huomioi kaikki tunnistamansa liikkeet. Myös ne, joihin sen ei ole tarkoitus reagoida. Tämä voidaan ratkaista rajoittamalla alue, jonka sisällä ohjain tunnistaa eleet. (Wigdor ym. 2011, 112, 115.) Lisäksi sovelluksen käytön aikana kaikkien eleiden ei tarvitse olla käytettävissä koko ajan. Rajaamalla eleet ohjelman tilojen mukaan helpotetaan sovelluksen käyttöä ja kehittämistä (Nielsen ym. 2003, 2). Luonnollisen käyttöliittymän suunnittelua voidaan helpottaa myös syötelaitteen tilaa kuvaavalla tilakaaviolla (Kuvio 3). Tilakaavioon merkitään laitteen jokainen tila sekä kaikki mahdolliset käyttäjän tekemät ohjausliikkeet nuolina tilojen välillä. Tilakaavioiden avulla saadaan kokonaisvaltainen kuva käyttöliittymän toiminnoista ja nähdään saako käyttäjä sovellukselta palautteen toiminnastaan. (Wigdor ym. 2011, 80, 83, 86.)



Kuvio 3: Esimerkki syötelaitteen tilakaaviosta (Wigdor ym. 2011, 83).

Pelialalle kehitetyn MDA-mallin käyttäminen luonnollisen käyttöliittymän suunnittelussa toimii kehittäjän ajattelumallina suunnitteluprosessin aikana. Malli ei anna yksityiskohtaisia suuntaviivoja suunnitteluun, mutta sen avulla voidaan havaita miten sovelluksen käyttäjät ja kehittäjät kokevat tuotteen eri tavalla. Siitä syystä MDA-malli auttaa ymmärtämään käyttäjän näkökulman sovellukseen paremmin. (Wigdor ym. 2011, 128.)

Mechanics, Dynamics and Aesthetics (MDA) -malli yhtenäistää kehittäjien, suunnittelijoiden, kriitikoiden ja tutkijoiden ymmärryksen peleistä. Mekaniikka tarkoittaa pelin sääntöjä ja rakennetta. Kaikkea sitä mitä pelaaja voi tehdä, niitä asioita joita hän voi käyttää sekä niitä jotka vaikuttavat hänen tekemiseen ja etenemiseen pelissä. Dynamiikka tarkoittaa mekaniikan käyttäytymistä pelin aikana. Miten eri mekaniikat käyttäytyvät suhteessa toisiinsa ja minkälainen vuorovaikutus niillä on pelaajan kanssa. Estetiikka tarkoittaa sitä, miten käyttäjä tuntee vuorovaikutuksen dynamiikan kanssa. MDA-mallissa kehittäjän näkökulmasta mekaniikka ohjaa dynamiikkaa, jonka myötä pelaaja saa tietynlaisen kokemuksen pelistä. Pelaajan näkökulmasta estetiikka määrittää tunteen, joka tulee havaitusta dynamiikasta ja käytettävissä olevasta mekaniikasta. (Hunicke, LeBlanc & Zubek 2004, 1-4.) Voidaankin todeta, että kehittäjän ja pelaajan näkemykset tuotteesta ovat täysin päinvastaiset.

5.3 Eleohjain

Eleohjauksella toimivien sovellusten toiminnallisuudet ja käyttötarpeet ovat yksilöllisiä ja siksi niiden käytössä tarvitaan erilaisia ohjaimia. Ohjaimet mittaavat erilaisia asioita ja määreitä, jonka takia ne eroavat toisistaan teknisten ominaisuuksien perusteella. Erilaisuudestaan huolimatta kaikki eleohjausjärjestelmät sisältävät sensorin, vertailijan ja toimilaitteen. Nämä kolme osaa voivat olla kaikki yhdessä fyysisessä laitteessa tai ne voivat olla jokainen omia laitteitaan. (Saffer 2009, 12.)

Sensori tunnistaa ympäristön muutokset, jotka voivat liittyä esimerkiksi ääneen, valoon, liikkeeseen tai matkaan. Käytettävä sensori määrittää minkälaisia muutoksia voidaan havaita. Sensori lähettää tietonsa vertailijalle, joka vertaa saatua ohjaimen tilatietoa edelliseen tilaan ja päättää mitä sovelluksessa tehdään. Vertailijan tekemä päätös välittyy toimilaitteelle, joka suorittaa komennon saadun päätöksen perusteella. (Saffer 2009, 13, 16.) Tässä opinnäytetyössä tutkittiin eleohjauksen toimintaa tv-grafiikan ohjauksessa. Edellä mainitut eleohjausjärjestelmän osat sijoittuvat seuraavasti: Sensorina toimii eleohjain, vertailijana ohjelmoitu grafiikkapohja ja toimilaitteena tv-grafiikkasovellus yhdessä grafiikkapohjan kanssa.

Yksi sensori voi tunnistaa vain yhdenlaisia muutoksia, mutta yksi eleohjain pystyy tunnistamaan useita erilaisia ympäristön muutoksia, koska ne voivat sisältää useamman kuin yhden

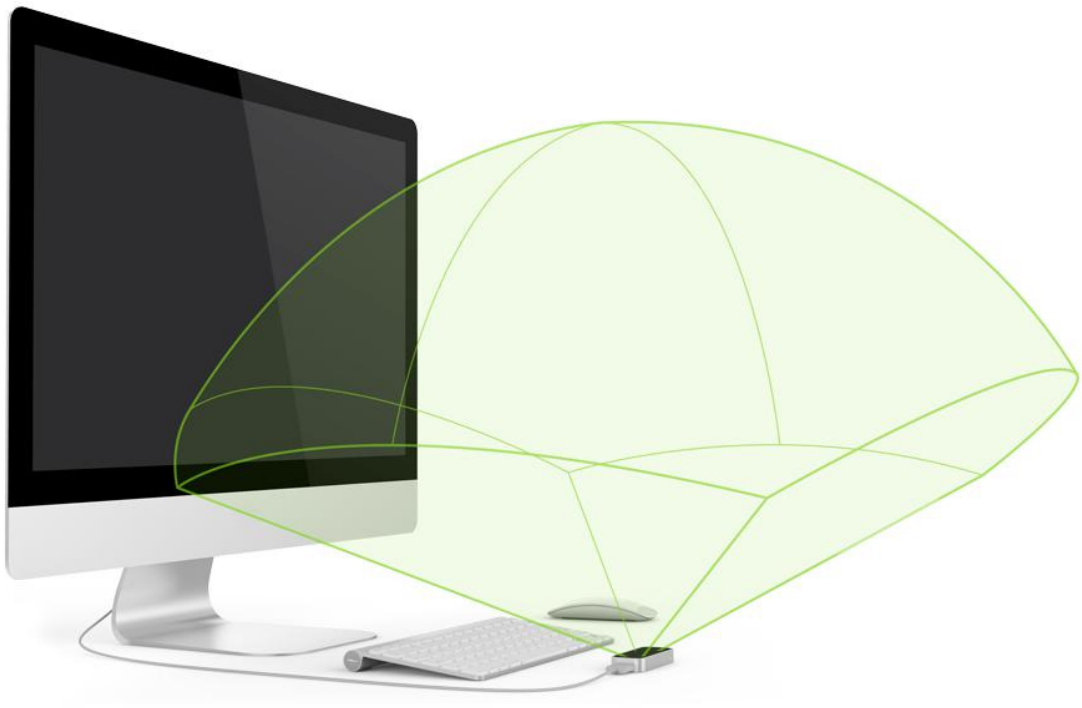
sensorin. Useammalla sensorilla laitteesta saadaan monipuolisempi ja tarkempi. Esimerkiksi kolmiulotteisten eleiden tunnistaminen on mahdotonta vain yhdellä sensorilla. (Saffer 2009, 14.)

5.4 Leap Motion

Leap Motion –ohjain on samannimisen yrityksen kädet ja sormet sekä niiden liikkeet tunnistava eleohjain. Ohjaimen avulla voidaan ohjata tietokonesovelluksia sekä virtuaalilasien avulla käytettävää virtuaalisovellusta. Tässä opinnäytetyössä tutkitaan laitteen toimintaa tietokoneeseen yhdistettynä. Siinä ei oteta kantaa sen käyttöön virtuaalilasien kanssa.

Leap Motion -ohjain kytketään tietokoneeseen laitteessa olevan USB-väylän kautta, jota pitkin välittyy laitteen keräämä tieto sekä sen tarvitsema virta. Sen lisäksi, että erillistä virtalaitetta ei tarvita, käyttöä helpottaa myös laitteen pieni koko. Laite on 8 cm leveä, 3 cm syvä ja 1,2 cm korkea. (Stringer 2013.)

Ohjaimen toiminta perustuu sen sisällä oleviin komponentteihin ja tietokoneelle asennettavaan ohjelmistoon. Sisällä on kaksi kameraa ja kolme infrapuna-LED-valoa, jotka luovat laitteen päälle näkymättömän kolmiulotteisen alueen, jonka sisällä laite tunnistaa liikkeen (Kuvio 4). Alueen korkeus on 80 cm ja sen alareunat lähtevät laitteen reunoilta 150 asteen kulmassa kaikkiin suuntiin muodostaen alueesta ylösalaisin olevan kartion. Ohjaimen luettua liikkeet se tallentaa tiedon sen omaan sisäiseen muistiin ja suorittaa ensimmäisen vaiheen tiedon prosessoinnista. Tämän jälkeen tieto siirtyy tietokoneella olevalle Leap Motion Control Panel -ohjelmalle, jossa matemaattisten algoritmien avulla saadaan eroteltua käsien ja sormien liikkeet laitteen näkemässä tilassa. Leap Motion Control Panel -ohjelma välittää tiedon eteenpäin käytettävälle sovellukselle TCP- tai WebSocket-protokollan välityksellä. (Colgan 2014.)



Kuvio 4: Leap Motion-ohjaimen toiminta-alue (Leap Motion 2017b).

Leap Motion -ohjaimella käytettäviä sovelluksia voi ladata yrityksen sovelluskaupasta, tai niitä voi kehittää itse valmiiden ohjelmointirajapintojen avulla. Valmiita rajapintoja on usealle eri ohjelmointikielelle, joita ovat C++, C#, Objective-C, Java, Python ja JavaScript sekä lisäksi liitännäiset Unity- ja Unreal Engine sovelluksille. (Leap Motion 2017a.)

5.5 Leap Motion JavaScript-rajapinta

Tässä opinnäytetyössä kaikki Leap Motion-ohjaimeen liittyvä ohjelmointi on toteutettu JavaScript-ohjelmointikielen avulla. Kaikki ohjaimen toiminnallisuudet on sisällytetty eleillä ohjattaviin grafiikkapohjiin. Tässä luvussa kuvataan Leap Motion sovelluskehitystä sen JavaScript-rajapinnan avulla.

Kehitettäessä sovelluksia Leap Motion -ohjaimelle tarvitaan sovelluskehitystyökalut, jotka voidaan ladata yrityksen verkkosivuilta kehittäjäportaalista. Ladattava sovelluskehitystyökalupaketti ei kuitenkaan sisällä JavaScript-ohjelmointiin tarvittavaa kirjastoa. Avoimen lähdekoodin JavaScript-kirjasto jaetaan erikseen GitHub arkistossa, johon löytyy linkki yrityksen verkkosivuilla olevasta ohjelmointi dokumentaatiosta (Leap Motion 2015c).

Leap Motion -ohjaimen JavaScript tuki koostuu kahdesta pääkomponentista, jotka ovat WebSocket-palvelin ja JavaScript-kirjasto. WebSocket-palvelin mahdollistaa sovellusten pää-

syn Leap Motion -ohjaimen seurantatietoon JSON-muotoisten viestien välityksellä. Yhteys sovellusten ja WebSocket-palvelimen välille muodostetaan JavaScript-kirjaston avulla. JavaScript-kirjaston välityksellä voidaan lähettää ja vastaanottaa viestejä JSON-muodossa. (Leap Motion 2015c.)

Ohjain tarkkailee liikettä sen sisällä olevien sensorien näkökentässä ottamalla kuvia maksimissaan 200 kuvan sekuntinopeudella. JavaScript-rajapinta tarjoaa näiden kuvien perusteella ohjelmoijan käyttöön Frame-objektin, joka on rajapinnan pääobjekti. Frame-objekti sisältää luettelon havaituista käsien, sormien ja sormen kaltaisten esineiden sijainnista, liikkeistä ja suunnasta. Lisäksi se sisältää tiedon rajapintaan valmiiksi ohjelmoiduista eleistä, joita ovat pyöräytys, pyyhkäisy, sormen painallus alaspäin sekä sormen painallus eteenpäin. (Leap Motion 2015a.) Valmiiksi ohjelmoidut eleet eivät sisälly automaattisesti Frame-objektiin. Jos eleet halutaan käyttöön, tulee sovelluksesta lähettää eleet käyttöönottava viesti WebSocket-palvelimelle. (Leap Motion 2015d.)

Leap Motion JavaScript-rajapinnan nimiavaruus on nimeltään Leap. Viittaamalla Leap-nimiavaruuteen ohjelmoitavan sovelluksen koodissa saadaan yhteys ohjaimen rajapintaan. Ohjain saadaan käyttöön luomalla uusi Controller-objekti sovelluksen koodissa tai käyttämällä rajapinnan sisältävää loop-funktiota. Käytettäessä Controller-objektia saadaan yhteys ohjaimeen lähettämällä kysely WebSocket-palvelimen välityksellä. Tällä tavoin voidaan hallita, milloin sovelluksen ajonaikana ohjain on käytettävissä. Loop-funktiota käytettäessä ei tarvitse luoda omaa Controller-objektia, sillä funktio luo yhteyden ohjaimeen ja WebSocket-palvelimeen. Loop-funktion avulla ohjain on koko ajan käytettävissä, sillä funktio suoritetaan säännöllisin päivitysvälein. Funktion suorittamisen päivitysväli voidaan asettaa kahdella tavalla, jotka ovat animationFrame ja deviceFrame. AnimationFrame on käytössä oletuksena ja se on riippuvainen käytettävästä selaimesta. Pääsääntöisesti nopeus on 60 kuvaa sekunnissa. DeviceFrame on Leap Motion -ohjaimen kehysnopeus, joka on riippuvainen käytettävän tietokoneen suorituskyvystä ja käyttäjän asettamista asetuksista. Käyttämällä deviceFrame vaihtoehtoa voidaan päästä jopa 200 kuvan kehysnopeuteen. (Leap Motion 2015b.)

6 Tutkimusmenetelmät

Tämän kehittämistutkimuksen tavoitteena oli tarkastella tekniikan soveltuvuutta sille asetetussa toimintaympäristössä ja tuottaa uutta tietoa sen toiminnasta tutkimuksen toimeksiantajan käyttöön tulevia kehitysprojekteja varten. Tutkimus toteutettiin laadullisen tutkimusmenetelmän keinoin. Laadullinen tutkimusmenetelmä pyrkii kuvaamaan luonnollisia tilanteita todellisesta elämästä. Laadulliselle tutkimusmenetelmälle on ominaista tutkittavan kohteen kokonaisvaltainen tarkastelu, jonka avulla pyritään löytämään tosiasioita ennemmin kuin todentaa olemassa olevia väittämiä. (Hirsjärvi, Remes & Sajavaara 2014, 161.)

Kehittämistutkimuksen ei katsota olevan oma tutkimusmenetelmä vaan se on monimenetelmäinen tutkimusstrategia, jossa yhdistyvät laadullinen ja määrällinen tutkimusmenetelmä. Kehittämistutkimuksessa kehittäminen perustuu tutkimuksen taustalla olevaan teoriaan ja uuden tai luotettavan tiedon tuottaminen edellyttää tiedon dokumentointia sekä tieteellisten menetelmien käyttöä. (Kananen 2012, 19, 21.)

Kehittämistutkimus pyrkii jonkun ongelman poistamiseen tai kehittämään jotain asiaa paremmaksi. Toisin kuin laadullisessa tutkimuksessa kehittämistutkimuksessa ei riitä ilmiön kuvailu tai ymmärtäminen, vaan siinä on tavoitteena löytää asiantiloille parempia vaihtoehtoja. (Kananen 2012, 21.)

Laadulliselle tutkimukselle on tyypillistä aineiston kerääminen luonnollisissa tilanteissa sekä ihmisten käyttäminen tiedon hankinnan välineenä. Tutkimuksen aineisto voidaan kerätä kyselyin, haastattelemalla, havainnoimalla ja valmiita dokumentteja käyttämällä. (Hirsjärvi ym. 2014, 164, 192.) Tämän kehittämistutkimuksen aineiston hankinta toteutettiin havainnoinnin ja avoimen haastattelun avulla.

Havainnointi toteutetaan tutkimuksen aikana kahdella tavalla, tarkkailevana havainnointina sekä osallistuvana havainnointina. Tarkkaileva havainnointi soveltuu hyvin ennakoimattomiin tilanteisiin (Vilkka 2014, 38). Tarkkailevalla havainnoinnilla tarkastellaan millä tavalla ihmiset ja tekniikka toimivat testitilanteessa. Pyrkimyksenä on selvittää, onko henkilöiden toiminta sellaista kuin he sanovat sen olevan ja miten he kokevat sovelluksen käyttämisen. Osallistuvaa havainnointia käytetään tutkimuksen kehittämisprosessin aikana kirjaamalla ylös kaikki ilmiötä koskevat havainnot. Kananen (2012, 95) kirjoittaa osallistuvan havainnoinnin etuna olevan, että se auttaa tutkijaa ymmärtämään ilmiön syvän olemuksen.

Havainnoinnin lisäksi aineistoa pyritään tarkentamaan avoimen haastattelun avulla. Sillä pyritään selvittämään henkilöiden ajatuksia, mielipiteitä ja tunteita. Avoimessa haastattelussa ei ole kiinteää runkoa, joten asiat pyritään selvittämään sitä mukaa kun ne nousevat keskustelussa esille (Hirsjärvi ym. 2014, 209). Haastattelut suoritetaan osana testaustilannetta, heti havainnoinnin jälkeen.

Kvalitatiivinen tutkimus on joustavaa ja siinä ongelma saattaa muuttua tutkimuksen edetessä (Hirsjärvi ym. 2014, 126, 164). Alasuutari kirjoittaa (1999, 84) aineistolle olevan ominaista sen moniulotteisuus ja rikkaus. Tämä tarkoittaa, että tutkimuksessa havainnointi ja haastattelut dokumentoidaan mahdollisimman yksityiskohtaisesti, jotta niistä saatava aineisto voidaan tutkia erilaisista näkökulmista katsoen. Tällöin aineisto on mahdollisimman kokonaisvaltainen ja sitä voidaan analysoida eri tavoin.

6.1 Validiteetti ja reliabiliteetti

Validiteetti ja reliabiliteetti ovat luotettavuuskäsitteitä, joiden avulla pyritään arvioimaan tutkimuksen luotettavuutta. Validiteetti kuvaa tutkimuksen pätevyyttä. Sillä tarkoitetaan tutkimuksessa asetettujen mittarien sekä tutkimusmenetelmän kykyä mitata tukittavaa asiaa. (Hirsjärvi ym. 2014, 231, 16.) Laadullisessa tutkimuksessa, jonka tarkoituksena ei ole suoraviivaisesti kuvata totuutta, voidaan validius ymmärtää ennemminkin uskottavuudeksi ja vakuuttavuudeksi (Saaranen-Kauppinen & Puusniekka 2006).

Reliabiliteetilla tarkoitetaan tutkimuksen tulosten toistettavuutta, eli kykyä tuottaa ei-sattumanvaraisia tuloksia. Tutkimuksen tulokset voidaan todeta reliaabeleiksi esimerkiksi, kun eri tutkijat päätyvät tutkimuksessa samanlaiseen tulokseen tai kun saman tutkittavan henkilön kanssa päädytään samaan tulokseen eri tutkimuskerroilla. (Hirsjärvi ym. 2014, 231; Holopainen & Pulkkinen 2008, 17.)

Tämän kehittämistutkimuksen validiteetin arvioinnissa huomioidaan tutkimuksen aikana kehitettyjen eleohjaustoiminnallisuuksien kehittämisen tapahtuneen opinnäytetyön teoriaosuudessa tutkittuun tietoon sekä vaiheittain toteutettuun kehittämiseen ja havainnoinnin analysointiin perustuen. Valittujen menetelmien avulla saatiin tutkimuksessa asetetut tavoitteet saavutettua. Näistä syistä tutkimuksen validiuden voidaan arvioida toteutuneen.

Tutkimuksen reliaabiliuden arviointi on erittäin vaikeaa, koska tutkimuksen aikana kehitetyt toiminnallisuudet on mahdollista tehdä toisella tavalla ja eri ohjelmointikieltä käyttämällä. Nämä valinnat ovat tutkimukseen osallistuvien henkilöiden päätöksiä jotka voivat vaikuttaa tutkimuksen lopputulokseen.

6.2 Lähdekritiikki

Tutkimusta tehdessä on pyrittävä kriittisyyteen lähteiden valinnassa ja tulkitsemisessa (Hirsjärvi ym. 2014, 113). Alasuutari (1999, 95) kirjoittaa lähdekritiikin olevan olennainen osa tutkimuksen analyysiä.

Lähdekritiikki voidaan jakaa ulkoiseen ja sisäiseen lähdekritiikkiin. Ulkoinen lähdekritiikki tarkoittaa lähteen aitoutta ja sitä liittyykö se tutkittavaan ilmiöön. Sisäinen lähdekritiikki tarkoittaa arviointia onko lähde aito, täsmällinen ja relevantti sekä sitä mitä siitä saadaan irti. (Koskennurmi-Sivonen.)

Kriittisessä lähteiden valinnassa tulisi ennen lukemista ja valitsemista kiinnittää huomiota seuraaviin seikkoihin: Onko kirjoittaja tunnettu tai arvostettu alallaan? Onko lähteestä saatava tieto riittävän tuoretta ja mikä on sen alkuperä? Onko julkaisija uskottava? Onko lähteestä saatava tieto objektiivista, eikä millään tavalla puolueellista tai väärää? (Hirsjärvi ym. 2014, 113-114.)

Tämän opinnäytetyön lähteiden valinnassa käytettävät kriteerit olivat lähteiden kirjoittajan arvostettavuus sekä lähteen uskottavuus. Uskottavuuden takaamiseksi pyrittiin valitsemaan lähteiksi aikaisempia tutkimuksia ja mahdollisimman tuoreita julkaisuja. Kirjoittajan arvostettavuus pyrittiin todentamaan valitsemalla alan asiantuntijoiden sekä käytettyjen teknologioiden kehittäjien julkaisuja.

7 Eleohjaustoiminnallisuuksien kehittämisprojekti

Projektin tavoitteena oli kehittää eleohjaustoiminnallisuudet CasparCG-nimiselle grafiikkaohjelmistolle. Projektissa käytettiin Leap Motion -eleohjainta, jonka avulla voidaan ohjata tietokonesovelluksia. Kehittämistutkimuksen tavoitteena oli luoda eleohjaustoiminnallisuudet sekä kerätä mahdollisimman paljon tietoa eleohjauksen suunnittelusta ja toteutuksesta, jota voidaan hyödyntää Maisti Oy:n tuleviin tuotekehitysprojekteihin.

Projektin aikana kehitetyt toiminnallisuudet toteutettiin vaiheittain. Jokainen vaihe koostui toiminnallisuuden kehittämisestä ja testauksesta. Projektin kehittämisvaiheessa oli mukana myös yrityksen tekniikasta ja ohjelmoinnista vastaava työntekijä.

7.1 Suunnittelu ja vaatimusmäärittely

Tutkimuksessa oli kyse tuotekehitysprojektista, joten tuotteen kehittäminen aloitettiin suunnittelemalla ensimmäisenä sen käyttötapaukset. Käyttötapauksien suunnittelu perustui kehittämiseen osallistuneiden aiempaan usean vuoden työkokemukseen tv-alalla ja erityisesti kosketusnäyttösovellusten kehitysprojekteihin tv-tuotantokäytössä. Eleohjainjärjestelmälle löydettiin kaksi erillistä käyttötapausta, jotka olivat isojen näyttöpintojen sisällönohjaaminen ja tv-grafiikan ohjaaminen televisiolähetyksissä.

Löydettyjen käyttötapauksien pohjalta määriteltiin toiminnallisuudet, jotka eleohjausjärjestelmällä tulisi voida suorittaa. Määritellyt toiminnallisuudet olivat sisällön vaihtaminen näy-

töllä, elementtien valitseminen sekä sisällön tuominen näkyviin ja pois näkyvistä. Näiden lisäksi, elementtien liikuttaminen näytöllä käden liikkeen mukana määriteltiin yhdeksi toiminnallisuudeksi, joka haluttiin myös testata tutkimuksessa.

Televisio-ohjelmassa eleohjauksen käyttäminen on aina esiintyvien henkilöiden tehtävä. Järjestelmä tehtiin lähtökohtaisesti tv-ohjelman juontajan käytettäväksi, mutta mahdollisuuksien mukaan myös muut ohjelmassa esiintyvät voivat sitä käyttää. Muut esiintyjät voivat olla ohjelman vieraita tai jonkun alan asiantuntijoita. Eleohjausjärjestelmä ei voi merkittävästi vaikeuttaa esiintyjien työtä, joten sen tulisi olla helppokäyttöinen ja toimintavarma. Lisäksi jos järjestelmän käyttäjä ei ole ohjelman vakituinen esiintyjä, niin sen tuli olla helposti opittava.

Käytettävien eleiden suunnittelu ja valinta toteutettiin tässä opinnäytetyössä kappaleessa 5.2 kuvatun ihmiskeskeisen lähestymistavan mukaan. Tutkimuksessa pyrittiin löytämään sellaiset eleet, jotka ovat helposti opittavissa ja muistettavissa. Lisäksi kiinnitettiin huomiota eleiden virheherkkyyteen.

CasparCG:n HTML-moduulin mahdollistamien HTML-grafiikkapohjien ja Leap Motion -ohjaimen tarjoaman JavaScript-rajapinnan myötä eleohjaustoiminnallisuudet päädyttiin kehittämään JavaScript-ohjelmointikielen avulla. Lisäksi se arvioitiin nopeimmaksi tavaksi toteuttaa toiminnallisuudet, eikä sen käyttö vaatisi uuden asiakasohjelmiston kehittämistä CasparCG-palvelimen ohjaamiseksi.

7.2 Kehittäminen ja testaus

Kehittämisen ensimmäisessä vaiheessa luotiin grafiikkapohja, jossa voidaan käden liikkeellä vaihtaa tekstiotsikoita näytöllä (Kuvio 5). Toiminnallisuus luotiin Leap Motion -ohjaimen rajapintaan valmiiksi kehitetyn pyyhkäisyeleen avulla. Pyyhkäisemällä grafiikkaelementit liikkuivat käden suuntaan oikealle tai vasemmalle, toinen näytöltä pois ja toinen näytölle näkyviin. Toiminnallisuuden ollessa valmis suoritettiin testi, jonka avulla haluttiin selvittää eleen rasittavuus sekä toimintavarmuus. Testi suoritettiin kehitysprojektiin osallistuneiden ja yhden yrityksen työntekijän toimesta. Tässä vaiheessa tutkimusta ei ollut tarvetta järjestää testiä suuremmalle testiryhmälle.

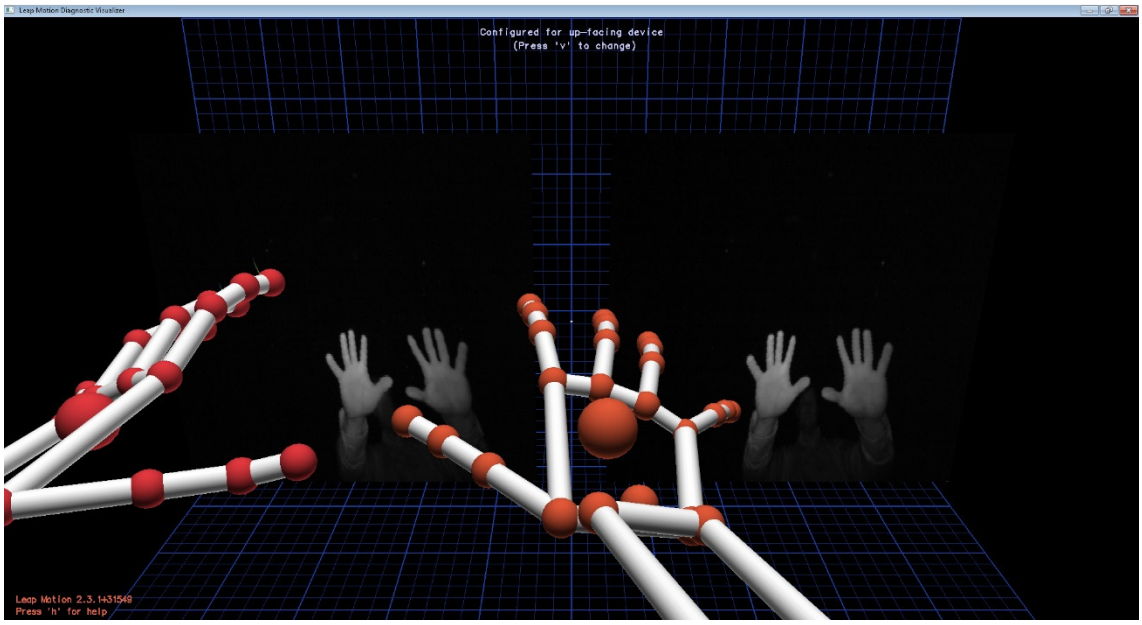
Kuvio 5: Käden liikkeellä ohjattava otsikkografiikka.

Testin aikana tehdyt havainnot osoittivat eleen rasittavuuden vähäiseksi, mutta toimintavarmuuden huonoksi. Kun eleet otettiin käyttöön Leap Motion -ohjaimen rajapinnasta, tuli pyyhkäisyn lisäksi kolme muutakin elettä käytettäväksi. Havainnot osoittivat, että käden liikerata ei ollut täysin vaakasuora, vaan käsi liikkui myös pystysuunnassa liikkeen aikana. Tämä aiheutti sen, että eleohjain tunnisti liikkeen pyyhkäisyn sijasta pyöräytykseksi. Useampi kuin yksi neljästä pyyhkäisystä tulkittiin ohjaimen toimesta virheellisesti, jonka vuoksi ohjaimen sisäänrakennettujen eleiden käytöstä luovuttiin.

Projektia jatkettiin pyyhkäisyeleen korvaavan toiminnallisuuden kehittämisellä, jossa käden liike oli samankaltainen kuin pyyhkäisyeleessä. Suurin ero oli käden asento, joka muuttui sellaiseksi, että sormet osoittavat eteenpäin ja kämmen alaspäin. Toiminnallisuus perustui Leap Motion -ohjaimen käden tunnistukseen, jossa se tunnistaa kämmenen keskikohdan. Toiminallisuus osoittautui jo kehittämisen aikana toimintavarmaksi, jonka myötä voitiin suorittaa testaus grafiikan ohjaukselle sekä tekniikan toimivuudelle sen oikeassa käyttöympäristössä.

Testi suoritettiin tv-studiossa, joka laitettiin vastaamaan täysin tv-lähetyksen olosuhteita. Studio sisälsi kaksi juontopaikkaa, joissa molemmissa suoritettiin eleohjaimen testaaminen. Testin avulla pyrittiin saamaan mahdollisimman paljon tietoa olosuhteiden vaikutuksesta laitteen toimintaan. Testiin osallistui neljä henkilöä, joiden toimintaa havainnoitiin ja tallennettiin videolle. Lisäksi heiltä pyrittiin saamaan lisää tietoa eleohjauksen käytöstä avoimen haastattelun avulla.

Testin aikana tekniikan toiminnasta tehdyt havainnot osoittivat, että kirkkaat valo-olosuhteet, jossa valo ei tule suoraan laitteen päältä, eivät vaikuta laitteen toimintaan. Heti kun studio oli täydessä valaistuksessa, tietokoneessa jossa Leap Motion -ohjain oli kiinni, tuli ilmoitus Leap Motion Control Panel -ohjelmalta kirkkaasta valaistuksesta. Ilmoitus ei kuitenkaan vaikuttanut laitteen toimintaan millään tavalla. Laitteen kameroiden näkymän voi nähdä Leap Motion Diagnostic Visualizer -ohjelmalla, jonka avulla voitiin seurata valojen ja laitteen käyttäytymistä. Heilutettaessa ohjainta sen kuvasta pystyi näkemään kaikki laitteen kameroiden kuvaamat valot, mutta kun ohjaimen antoi olla paikoillaan, se mukautui olosuhteisiin niin, että valot eivät olleet nähtävissä (Kuvio 6).



Kuvio 6: Leap Motion -ohjaimen näkymä pöydältä tv-studion täydessä valaistuksessa.

Testin aikana suoritetut havainnot osoittivat eleohjaimen toimintavarmuuden käytettävän ohjauksen kohdalla. Grafiikanohjauksessa käytettävä pyyhkäisyele tehtiin hieman yli 100 kertaa peräkkäin, josta saatiin onnistumisprosentiksi 100. Pyyhkäisyeleet tehtiin molempiin suuntiin, joista kaikki onnistuivat. Havainnot osoittivat myös CasparCG:n aiheuttavan viivettä grafiikan liikkeisiin. Suoritetun eleen ja grafiikan liikkeen välillä havaittiin hieman alle sekunnin viive. Tällaista viivettä ei havaittu testattaessa ohjausta Chrome-selaimen avulla.

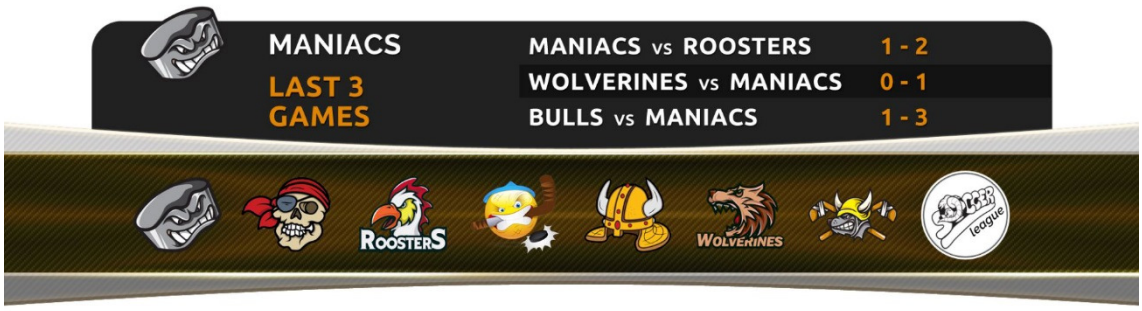
Toimintavarmuus testattiin, kun testiin osallistuneet henkilöt olivat jo kokeilleet ja käyttäneet laitetta jonkin aikaa. Laitteen käyttökokemus vaikutti testitulokseen positiivisesti. Havainnot osoittivat, että laitetta ensimmäistä kertaa käyttäessään testihenkilöiden eleet olivat todella nopeat, jolloin eleohjain ei tunnistanut eleitä. Henkilöitä opastettiin käyttämään rauhallisempia liikkeitä, jonka myötä ohjaus muuttui sujuvammaksi. Havainnot osoittivat lisäksi osan henkilöistä tekevän eleet niin, että heidän sormet menevät laitteen päältä. Tämä aiheutti ongelmia toimintavarmuudessa, koska eleohjain tarkkailee kämmenen sijaintia ja se ei tunnistanut heidän kämmentä. Opastamalla testihenkilöitä tekemään liikkeen lähempänä eleohjainta, järjestelmän toimintavarmuus parani.

Testin aikana suoritetuissa haastatteluissa saatiin selville, että Leap Motion -eleohjaimen käyttäminen testissä käytettyyn käyttötarkoitukseen koettiin toimivaksi ja nopeasti opittavaksi. Eleen nopea oppiminen ja sen soveltuvuus sille valittuun käyttötarkoitukseen on yksi luonnollisen käyttöliittymän tavoitteista. Haastattelussa nousi esille myös, että kun keskittyy esiintymään kameralle, niin keskittyminen eleohjaukseen heikkenee. Kun näin tapahtuu, niin se vaikuttaa liikkeiden nopeuteen ja käden sijaintiin ohjaimeen nähden.

Muita Leap Motion -eleohjaimen käyttöön tai tekniseen toimivuuteen liittyviä tutkimukseen merkittäviä asioita ei noussut esille haastatteluissa. Sen sijaan käyttökohteisiin liittyviä kommentteja nousi esille ja yhdeksi mahdolliseksi mainittiin lisätty todellisuus, jossa virtuaalisia elementtejä yhdistetään reaalimaailmaan. Tämä tarkoittaisi graafisten elementtien sijoittelun kamerakuvaan siten, että ne näyttävät olevan osa lavastusta.

Kehittämisprojektin seuraava vaihe oli todentaa erilaisten eleiden toimivuus sekä selvittää minkälainen prosessi on grafiikkapohjan eleohjaustoiminnallisuuksien siirtäminen toiseen grafiikkapohjaan. Jälkimmäinen koettiin yhdeksi projektin tärkeimmistä ominaisuuksista, sillä se vaikuttaa tuotekehitysprosessin läpivientiin merkittävästi. Eleohjausjärjestelmän kehittäminen jatkui projektin alussa määriteltyjen toiminnallisuuksien kehittämisellä, joita ei vielä ollut testattu. CasparCG:n aiheuttaman sekunnin viiveen vuoksi elementtien seuraamisesta käden liikkeen mukana luovuttiin sellaisenaan, mutta osittain sitä vastaava toiminnallisuus toteutettiin. Lisäksi elementtien valitseminen sekä sisällön tuominen näkyviin ja pois näkyvistä haluttiin vielä testata.

Uudet toiminnallisuudet kehitettiin grafiikkapohjaan, jossa on kuvitteellisten joukkueiden logoja (Kuvio 7). Logoja painamalla yhdellä sormella osoittaen voidaan näyttää jokaiselle joukkueelle oma grafiikka. Joukkueen valinta tehdään käden liikkeellä vaakatasossa ja grafiikan rivitystä voidaan vaihtaa pystysuuntaisella liikkeellä.



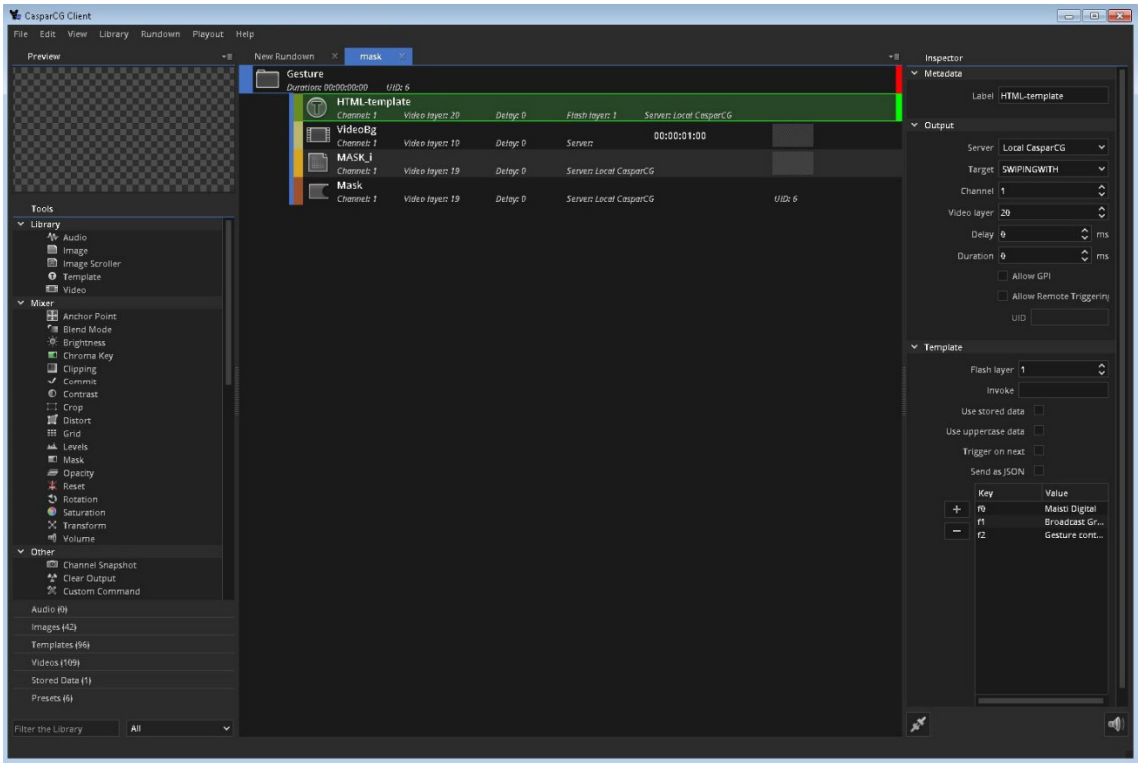
Kuvio 7: Grafiikkapohja, jossa on avattuna yhden joukkueen grafiikka.

Uusien toiminnallisuuksien kehittämisen myötä havaittiin, että eleohjaustoiminnallisuudet voidaan siirtää helposti eri grafiikkapohjien välillä. Eleiden toiminnallisuudet siirtyvät kopioimalla tarvittavat muuttujat sekä loop-funktio, jonka avulla saadaan yhteys Leap Motion -ohjaimen JavaScript-rajapintaan. Lisäksi tv-studiossa tehty tutkimus osoitti, että jos eleet ovat yksinkertaiset, niiden oppiminen on helppoa. Tästä syystä uudet kehitetyt toiminnallisuudet testattiin sisäisesti kehittämistyön toimeksiantajayrityksessä, eikä suuremmalle testiryhmälle koettu tarvetta.

Havainnot osoittivat Leap Motion -ohjaimen liikkeen tunnistuksen niin tarkaksi, että pienikin käden liike aiheutti väärän grafiikan avautumisen. Lisäksi toiminnallisuus, jolla vaihdettiin grafiikassa näkyvää riviä, osoittautui vaikeaksi käyttää. Sen toiminta perustui näkymättömään alueeseen, joka käden täytyi halkaista grafiikan animaation käynnistämiseksi. Ongelma johtui siitä, että alueen sijainnista ei saanut mitään visuaalista vihjettä. Ongelmallisiksi havaitut toiminnallisuudet muokattiin ohjelmakoodiin muuttamalla käden sijaintitiedon lukutapaa sekä lisäämällä ehtoja grafiikkaelementtien liikuttamiseen. Näin eleet voitiin pitää samoina ja niistä saatiin toimintavarmempia.

7.3 Eleohjaustoiminnallisuuksien toteutus

Eleohjausjärjestelmä koostui eleohjaimen lisäksi CasparCG-palvelimesta ja sen asiakasohjelmistosta sekä HTML-grafiikkapohjista. Palvelimen tehtävänä oli avaintaa grafiikka ja näyttää se videokuvan päällä. Asiakasohjelmiston avulla ohjattiin palvelinta sekä rakennettiin kokonaisuus, jossa yhdistetään video ja grafiikkapohja yhdeksi näytettäväksi grafiikaksi. Kuviossa 8 kuvataan asiakasohjelmiston ajolista, jossa on tekstielementit sisältävä HTML-grafiikkapohja sekä niiden taustavideo ja maskit asetettuna ryhmän sisään. Ryhmälle annettavalla play-komennolla voidaan esittää kaikki sen sisältämät elementit yhtenä grafiikkana. Asiakasohjelmiston avulla voidaan lisätä, poistaa tai muokata grafiikan sisältämiä tekstejä. Tekstit ovat erillisiä DIV-elementtejä HTML-grafiikkapohjassa, jotka ovat yksilöity id:n avulla. Asiakasohjelmistossa teksti voidaan määrittää halutulle DIV-elementille id:n perusteella.



Kuvio 8: CasparCG asiakasohjelmiston käyttöliittymä.

Grafiikkapohjien sisältö on samankaltainen kuin HTML-merkintäkielen avulla tehdyt internet-sivut. Suurin eroavaisuus on grafiikkapohjiin tehtävät paikalliset JavaScript-ohjelmoinnit ja CSS-tyylit. Internet-sivujen rakenteessa nämä ovat yleensä erillisissä tiedostoissa.

Grafiikkapohjien kehittämisessä käytettiin erilaisia JavaScript-kirjastoja, jotka ovat jQuery ja GreenSock Animation Platform (GSAP) sekä rajapinnat CasparCG:lle ja Leap Motion -ohjaimelle. Kehittämisprojektin ensimmäinen testi osoitti, että Leap Motion -ohjaimen rajapinnan tarjoamat valmiiksi ohjelmoidut eleet eivät ole käyttökelpoisia. Lisäksi havaittiin CasparCG:n aiheuttavan sekunnin viiveen elementtien ohjaukseen. Näistä syistä ohjaimen toiminta luotiin sellaiseksi, että sen havaitessa ohjaukseen määritelty ele, suoritetaan jokin animaatio. Nämä animaatiot tehtiin GreenSock Animation Platform-animaatiokirjastoa sekä sen liitännäisiä hyödyntäen.

Järjestelmän eleohjaustoiminnot sisällytettiin HTML-grafiikkapohjiin. Yhteys eleohjaimen rajapintaan luotiin loop-funktion avulla, jolloin ohjain on koko ajan käytettävissä. Loop-funktion määrittelyssä asetettiin WebSocket-palvelimen IP-osoite ja määritettiin grafiikka-pohja vastaanottamaan Frame-objekteja myös silloin kun sovellus on käynnissä taustalla. WebSocket-palvelin sijaitsee tietokoneessa, jossa ohjain on käytössä. Kuviossa 9 on kuvattuna loop-funktion rakenne sekä ohjaimelle asetettavat asetukset.

```
var controllerOptions = {
    host: leapHost,
    background: true
};

var controller = Leap.loop(controllerOptions, function(frame) {

var hand = frame.hands.length;

if(hand){
frame.hands.forEach(function(hand, index) {
    var position = hand.palmPosition;
    var posX = position[0];
    var posY = position[1];
    var posZ = position[2];

    if(posX>-50 && posX<50 && posZ<60 && posZ>-60 && posY<2000){

        if(newsReaderMode){
            handleXcoord(-posX);
        }
        else{
            handleXcoord(posX);
        }
    }
    });
} else {
animationsDone = true;
}

});
```

Kuvio 9: Leap Motion -ohjaimen yhdistävä loop-funktio.

Grafiikoiden ohjaus kehitettiin käden liikkeiden seurannalla. Käden sijainti saadaan ohjaimen rajapinnasta Hand-objektin palmPosition-attribuutin avulla. Ohjain lukee käden sijaintia koko ajan käden ollessa ohjaimen näkökentässä, käyttäen hyväksi forEach-funktiota. PalmPosition-attribuutti tarjoaa tiedon JavaScript-taulukkona, joka sisältää kolme arvoa. Arvot ovat järjestyksessään käden sijainti x-akselilla, y-akselilla sekä z-akselilla. Käden liikkeen seuranta toteutettiin tallentamalla käden sijaintitieto kahteen muuttujaan, joita vertailemalla saatiin selville liikkeen suunta.

TV-studiossa testattuun grafiikkapohjaan, jossa vaihdetaan tekstejä käden liikkeellä, rajattiin käden sijainnin perusteella 10 cm leveä, 12 cm syvä ja 20 cm korkea alue ohjaimen yläpuolelta, joka käden tuli halkaista animaatioiden käynnistämiseksi. Alueen rajaamisella pyrittiin vähentämään ohjaimen virheelliset eleiden tunnistamiset. Käden liikkeen lisäksi tallennettiin liikkeen aloitus- ja lopetusaika, joiden perusteella laskettiin liikkeen nopeus.

Nopeutta käytettiin tekstin vaihtumisanimaation nopeutena. Lisäksi nopeudelle asetettiin maksimi- ja minimiajat, jotta animaatio ei olisi liian hidas tai nopea.

Projektin aikana kehitettiin grafiikka, jossa logoja painamalla voidaan näyttää kyseiseen joukkueeseen liittyvä grafiikka. Grafiikkapohja sisälsi kaksi riviä logoja, joista vain toinen rivi on näkyvissä. Käden liikkeellä ylös tai alaspäin saatiin vaihdettua näkyvää riviä. Toiminnallisuus toteutettiin Controller-objektin frame-funktion avulla. Leap Motion -ohjaimen rajapinnasta saadaan tieto laitteen tallentamista kuvakehyksistä komennolla Control.frame(20), jossa numero 20 osoittaa kuinka monta kuvaa nykyhetkestä taaksepäin tieto esitetään. Ohjain kuvaa oletuksena 60 kuvaa sekunnissa, joten 20 on yksi kolmasosasekunti. Toiminnallisuus toteutettiin tallentamalla kuvat 40 ja 20 kuvakehystä taaksepäin. Vertaamalla käden sijaintia näissä kohdissa saatiin selville mihin suuntaan käsi on liikkunut. Lisäksi liikkeelle asetettiin ehdoksi, että muutoksen on oltava yli 5 cm, jotta animaatio suoritetaan.

Grafiikassa käden liikettä seuraava osoitin elementti, toteutettiin seuraamalla käden sijaintia x-akselilla palmPosition-attribuutin avulla. Osoittimen liikuttamiseksi asetettiin ehdoksi, että laite tunnistaa kädestä enemmän kuin kaksi sormea. Tällä ehdolla vähennettiin virheellisten grafiikoiden aukeamista. Käden liikkeen mukaan siirrettiin osoittimen sijaintia näytöllä jQueryä hyödyntäen. Osoittimen sijainti oli mahdollista lukita osoittamalla yhdellä sormella eteenpäin laitteen näkökentässä. Laitteen tunnistamat sormet saadaan rajapinnasta taulukkona Hand-objektin Fingers-attribuutin avulla. Taulukon elementtien lukumäärä on sama kuin tunnistettujen sormien lukumäärä.

Grafiikkaelementtien näyttäminen ja piilottaminen tehtiin sormien lukumäärää tarkkailevaa toimintoa sekä käden sijainnin seurantaa hyödyntäen. Osoittamalla yhdellä sormella sekä painamalla eteenpäin laitteen z-akselin suuntaisesti yli 5 cm verran, avautuu valitun joukkueen grafiikka. Avatun grafiikan ollessa näkyvillä, grafiikkapohja ei ota muita käskyjä kuin painamisen eteenpäin. Tällä suojataan mahdolliset virhetilanteet, joissa grafiikka poistuu näkyvistä tahattomasti. Tekemällä painamiseleen uudelleen voidaan grafiikka piilottaa.

8 Yhteenveto ja johtopäätökset

Kehittämistutkimuksen tavoitteiksi asetettiin kehittää uusi tuotekokonaisuus toimeksiantajan käyttöön sekä tuottaa tietoa tutkimuksessa käytettävän eleohjaimen toiminnasta, sille tarkoitetussa toimintaympäristössä. Lisäksi tavoitteena oli saada mahdollisimman paljon tietoa eleohjauksesta ja sen kehittämisestä, jota voidaan hyödyntää yrityksen eleohjausjärjestelmien tuotekehitysprosessissa.

Tutkimuksessa käytettiin laadullisen tutkimusmenetelmän tiedonkeruutapoja, jotka olivat havainnointi ja avoin haastattelu. Havainnointia suoritettiin testaustilanteissa sekä koko kehitysprojektin ajan. Havainnoinnin avulla saatiin tietoa eleohjausjärjestelmän teknisestä toimivuudesta sekä sen käyttämisestä. Avoimen haastattelun avulla voitiin varmentaa havaintojen paikkansa pitävyys ja saatiin tietoa järjestelmän käytön haasteista.

Opinnäytetyön teoriaosuudessa käsiteltiin tv-grafiikkaa sekä eleohjausta ja sen suunnittelua. Luonnollisen käyttöliittymän suunnittelussa ei lähtökohtana pidä olla vain halu tehdä sellainen, vaan kehityksen täytyy perustua johonkin tarpeeseen. Tämän kehittämisprojektin tarpeet olivat ohjaus isoille näyttöpinnoille, joita ei voida ohjata muulla tavoin niiden koon takia sekä yritykselle esitetty mielenkiinto eleohjauksella ohjattavia grafiikoita kohtaan.

Projektin aikana kehitettiin eleohjausjärjestelmä yhdistelemällä teoriaosuudesta saatua tietoa ja havainnoinnin avulla kerättyä tietoa. Projektin alussa suunniteltiin eleohjauksen toteuttamista kehittämällä uusi asiakasohjelmisto CasparCG-palvelimen ohjaamiseksi C#-ohjelmointikielen avulla. Tästä luovuttiin kuitenkin kehittämisen alkaessa. Projektin aikana päätös osoittautui hyväksi, sillä eleohjauksen kehittäminen JavaScript-ohjelmointikielen avulla todettiin nopeaksi ja helposti muutettavaksi. Tämän ansiosta voidaan luoda uusia toimintoja ja lisätä käytettäviä eleitä uusiin tai olemassa oleviin grafiikoihin helposti. Jos olisi päädytty kehittämään uusi asiakasohjelmisto, niin siitä täytyisi luoda aina uusi versio, kun lisättäisiin uusi toiminnallisuus.

Tutkimus osoitti Leap Motion -ohjaimen tarkkuuden aiheuttavan ajoittain ongelmia. Niitä ilmeni erityisesti sormien lukumäärän ilmoittamisessa, jos käden asento oli sellainen, että sormet olivat kiinni toisissaan. Lisäksi tarkkuutta vaativat liikkeet, kuten joukkueiden valinta tutkimuksen aikana kehitetyssä grafiikassa, jossa oli logoja, tuntui välillä haastavalta ohjaimen tarkkuuden vuoksi. Näiden havaintojen perusteella nousi epäilyjä laitteen soveltuvuudesta kehitettyyn käyttötarkoitukseen. Itse arvioin Leap Motion -ohjaimen toimivan hyvin sellaisissa tapauksissa, jossa ohjaukseen käytettävät eleet ovat laajoja ja yksinkertaisia, kuten vaakasuuntainen pyyhkäisyele otsikoiden vaihtamiseen tarkoitetussa grafiikassa. Tutkimus osoitti myös ohjauksen olevan haasteellista silloin, kun täytyy keskittyä esiintymään kameralle ja ohjaamaan grafiikkaa samanaikaisesti. Käden sijoittaminen laitteen näkökenttään oikeaan kohtaan ei aina onnistunut ilman katseen siirtymistä pöytään, joka vähentää eleohjaimen käytön luonnollisuutta. Tästä syystä Leap Motion -eleohjain ei mielestäni ole paras vaihtoehto grafiikoiden ohjaukseen. Tällaiseen käyttöön paremmin sopiva eleohjain olisi mielestäni sellainen, jonka toiminta ei perustu tietyn alueen sisällä tapahtuvaan liikkeiden tunnistukseen. Opinnäytetyön toimeksiantajalla on käytössään myös Myo-eleohjaimia, jotka laitetaan käden ympärille ja joiden liikkeentunnistus perustuu lihasten liikkeiden tunnistamiseen

sekä kiihtyvyysanturin avulla tehtyyn mittaukseen, jolloin eleiden suorittaminen on käden sijainnin perusteella vapaampaa. Tällöin ohjaus on luonnollisempaa, koska ei tarvitse keskittyä käden sijaintiin ohjaimeen nähden ja siksi Myo-ohjain olisi mahdollisesti parempi vaihtoehto eleohjaimeksi tällaiseen järjestelmään. Tämän tutkimuksen tuottamaa tietoa voidaan hyödyntää myös Myo- tai muiden eleohjainten käyttöön perustuvassa järjestelmässä, sillä sen avulla saatiin kerättyä yleisesti eleohjauksen suunnitteluun perustuvaa tietoa.

Projektin aikana saavutin tavoitteet, jotka olin asettanut itselleni. Vaikka eleohjausjärjestelmään ei toteutettu mitään C#-ohjelmointikielen avulla, niin ehdin tutustua ja opetella tekemään sen avulla asiakasohjelmiston jolla voidaan ohjata CasparCG-palvelinta. Lisäksi opin luonnollisen käyttöliittymän ja eleiden suunnittelun menetelmiä, joita on mahdollista hyödyntää tulevaisuudessa.

9 Jatkotutkimuskohteet

Opinnäytetyön jatkotutkimuskohteiksi voidaan esittää viimeisimpien kehitettyjen eleohjaustoiminnallisuuksien testaamisen tv-studiossa, tutkimuksen ulkopuolisten testaajien toimesta. Nämä eleohjaustoiminnallisuudet ovat grafiikan ohjaus käden pystysuuntaisella liikkeellä ylöspäin ja alaspäin, sekä sormella eteenpäin painaminen. Testaamalla ne ulkopuolisilla testikäyttäjillä, saataisiin tietoa eleiden käytettävyydestä niiden käyttötarkoitukseen sekä niiden muistettavuudesta. Lisäksi voitaisiin todentaa eleiden visuaalinen näyttävyys kameroiden välityksellä ja tarvittaessa poistaa sellaiset eleet, joiden suorittaminen ei näytä luonnolliselta ja hyvältä.

Lähteet

Painetut

Alasuutari, P. 1999. Laadullinen tutkimus. 3., uudistettu painos. Tampere: Vastapaino.

Hirsjärvi, S., Remes, P. & Sajavaara, P. 2014. Tutki ja kirjoita. 19. painos. Helsinki: Tammi.

Holopainen, M. & Pulkkinen, P. 2008. Tilastolliset menetelmät. 5., uudistettu painos. Porvoo: WSOY Oppimateriaalit.

Kananen, J. 2012. Kehittämistutkimus opinnäytetyönä: Kehittämistutkimuksen kirjoittamisen käytännön opas. Jyväskylä: Jyväskylän ammattikorkeakoulu.

Wigdor, D. & Wixon, D. 2011. Brave NUI World, Designing Natural User Interfaces for Touch and Gesture. Burlington: Elsevier Inc.

Sähköiset

Avoinrajapinta. 2014. Avoimen rajapinnan määritelmä. Viitattu 25.3.2017. http://avoinrajapinta.fi/

CasparCG. 2013. Main Page. Viitattu 14.4.2017. http://casparcg.com/wiki/Main_Page

CasparCG. 2016. Protocols. Viitattu 12.3.2017. http://casparcg.com/wiki/CasparCG_Server#Protocols

Colgan, A. 2014. How Does the Leap Motion Controller Work? Viitattu 17.2.2017. http://blog.leapmotion.com/hardware-to-software-how-does-the-leap-motion-controller-work/

Coss. 2017. Avoin lähdekoodi. Viitattu 25.3.2017. https://coss.fi/avoimuus/avoin-lahdekoodi/

GreenSock. 2017. Viitattu 25.3.2017. https://greensock.com/

Hunicke, R., LeBlanc, M. & Zubek, R. 2004. MDA: A Formal Approach to Game Design and Game Research. Viitattu 12.2.2017. http://www.cs.northwestern.edu/~hunicke/MDA.pdf

Karlsson, P. 2014. CasparCG Server 2.0.7 released. Viitattu 12.3.2017. http://www.casparcg.com/forum/viewtopic.php?f=5&t=2857

Kaushik, M. & Jain, R. 2014, Natural User Interfaces: Trend in Virtual Interaction. Viitattu 7.2.2017. http://arxiv.org/ftp/arxiv/papers/1405/1405.0101.pdf

Koskennurmi-Sivonen, R. Historiantutkimus. Viitattu 1.4.2016. http://www.helsinki.fi/~rkosken/hist.html

Lappalainen, Y., Rakkolainen, I., Korkalainen, T., Okkonen, J., Mäkilä, T., Pääkylä, J., Lehtonen, T. & Lakkala, M. 2015. Tulevaisuudennäkymiä. Teoksessa Lappalainen, y., Poikolainen, M. & Trapp, H. (toim.) Tila haltuun! Suositukset virtuaalisen suomen opiskelun toteuttamiseen. Viitattu 7.2.2017. https://www.utu.fi/fi/yksikot/braheakeskus/esittely/Documents/Tila%20haltuun!%20Suositukset%20virtuaalisen%20suomen%20opiskelun%20toteuttamiseen.pdf

Leap Motion. 2015a. API Overview. Viitattu 17.2.2017. https://developer.leapmotion.com/documentation/v2/javascript/devguide/Leap_Overview.html

Leap Motion. 2015b. API Reference. Viitattu 11.3.2017. https://developer.leapmotion.com/documentation/v2/javascript/api/Leap_Classes.html#leap-namespace

Leap Motion. 2015c. SDK Libraries. Viitattu 11.3.2017. https://developer.leapmotion.com/documentation/v2/javascript/devguide/Leap_SDK_Overview.html?proglang=javascript

Leap Motion. 2015d. WebSocket Communication. Viitattu 11.3.2017. https://developer.leapmotion.com/documentation/v2/javascript/supplements/Leap_JSON.html?proglang=javascript

Leap Motion. 2017a. Leap Motion SDK and Plugin Documentation. Viitattu 17.2.2017. https://developer.leapmotion.com/documentation/index.html?proglang=current

Leap Motion. 2017b. Think Big. Viitattu 17.2.2017. https://www.leapmotion.com/product/desktop#107

Media Art Net. 2005. Myron Krueger, Videoplace. Viitattu 10.2.2017. http://www.medienkunstnetz.de/works/videoplace/

Millerson, G. & Owens, J. 2008. Video Production Handbook. Viitattu 25.2.2017. http://home.fa.utl.pt/~cfig/Anima%E7%E3o%20e%20Cinema/Realiza%E7%E3o%20Cinematogr%E1fica/Video%20Production%20Handbook,%20Fourth%20Edition.pdf

Mozilla. 2017. JavaScript-asetukset vuorovaikutteisten verkkosivujen avaamiseen. Viitattu 25.3.2017. https://support.mozilla.org/t5/Manage-preferences-and-add-ons/JavaScript-asetukset-vuorovaikutteisten-verkkosivujen-avaamiseen/ta-p/21406

Nagy, R. 2014. Workshop-CasparCG. Viitattu 12.3.2017. https://tech.ebu.ch/docs/events/opensource12/presentations/Workshop-CasparCG-Robert-Nagy.pdf

Nielsen, M., Störring, M., Moeslund, T., B. & Granum, E. 2003. A procedure for developing intuitive and ergonomic gesture interfaces for man-machine interaction. Viitattu 11.2.2017. https://www.researchgate.net/publication/2556937_A_Procedure_For_Developing_Intuitive_And_Ergonomic_Gesture_Interfaces_For_Man-Machine_Interaction

Open Sound Control. 2011. OSC. Viitattu 22.3.2017. http://opensoundcontrol.org/osc

Polaine, A. 2009. Talk to the Hand: Dan Saffer and gestural interfaces. Viitattu 6.2.2017. http://www.core77.com/posts/12522/talk-to-the-hand-dan-saffer-and-gestural-interfaces-by-andy-polaine-12522

Saaranen-Kauppinen, A. & Puusniekka, A. 2006. 3.3.1 Validiteetti. Viitattu 26.2.2017. http://www.fsd.uta.fi/menetelmaopetus/kvali/L3_3_1.html

Saffer, D. 2009. Designing Gestural Interfaces. Viitattu 6.2.2017. https://static.aminer.org/pdf/PDF/000/334/249/movement_and_music_designing_gestural_interfaces_for_computer_based_musical.pdf

Shamonsky, D. 2014. The Idea of a Natural User Interface is Not Naturally Easy to Grasp. Viitattu 6.2.2017. http://www.ics.com/blog/idea-natural-user-interface-not-naturally-easy-grasp

Stringer, L. 2013. What is Leap Motion?. Viitattu 17.2.2017. http://www.geeksquad.co.uk/articles/what-is-leap-motion

Vihavainen, A. 2014. Web-palvelinohjelmointi. Viitattu 12.3.2017. http://wepa.mooc.fi/index.html#chapter2-2

Vilkka, H. 2014. Tutki ja havainnoi. Viitattu 26.2.2017. http://hanna.vilkka.fi/wp-content/uploads/2014/02/Tutki-ja-havainnoi.pdf

w3schools. 2017a. HTML Introduction. Viitattu 25.3.2017. https://www.w3schools.com/html/html_intro.asp

w3schools. 2017b. CSS Introduction. Viitattu 25.3.2017. https://www.w3schools.com/css/css_intro.asp

w3schools. 2017c. jQuery Introduction. Viitattu 25.3.2017. https://www.w3schools.com/jquery/jquery_intro.asp

w3schools. 2017d. JSON - Introduction. Viitattu 12.3.2017. https://www.w3schools.com/js/js_json_intro.asp

West, M. 2013. An Introduction to WebSockets. Viitattu 25.3.2017. http://blog.teamtreehouse.com/an-introduction-to-websockets

Weston, B. 2009. The basics of video Keying. Viitattu 27.2.2017. https://www.renewedvision.com/downloads/tfwm_keying_article.pdf

Kuviot