Jose Uusitalo

Virtuaalitodellisuussovellus rakennusalan työturvallisuuskoulutukseen

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Tieto- ja viestintätekniikka

Insinöörityö

29.5.2017

Metropolia

Tekijä Otsikko Sivumäärä Aika	Jose Uusitalo Virtuaalitodellisuussovellus rakennusalan työturvallisuuskoulutukseen 72 sivua + 10 liitettä 29.5.2018
Tutkinto	insinööri (AMK)
Tutkinto-ohjelma	Tieto- ja viestintätekniikka
Ammatillinen pääaine	Ohjelmistotuotanto
Ohjaajat	lehtori Simo Silander tutkimusinsinööri Kristian Lukander

Insinöörityön aiheena oli virtuaalitodellisuussovellus, jolla suoritetaan rakennusalan työturvallisuuskoulutuksia. Sovellus tehtiin ketteränä kehitystyönä Työterveyslaitoksen tuotekehitysprojektiin, jonka tavoitteena oli selvittää, kuinka hyvin virtuaalitodellisuus soveltuu työturvallisuuskoulutuksen tarpeisiin sekä millaisia mahdollisuuksia ja rajoituksia se aiheuttaa koulutusten sisältöön. Päätavoitteena oli perehtyä rakennusalalla käytettyyn turvapuistokoulutukseen ja valita sen oppimista tukevat ominaisuudet osaksi sovellusta sekä korjata siinä havaittuja ongelmia. Työn aikana ei ollut tarkoitus luoda lopullisia turvallisuuskoulutuksia, vaan rakentaa sovellus, jolle koulutuksia voidaan myöhemmin toteuttaa.

Sovellus toimii Windows-käyttöjärjestelmillä, se kehitettiin käyttämällä Unity-pelimoottoria ja HTC Vive -virtuaalitodellisuusjärjestelmää Työterveyslaitoksen tiloissa, virtuaalitodellisuuskehitykseen omistetussa laboratoriossa. Koulutussovellukseen toteutettiin kaksi esimerkkikoulutusta havainnollistamaan ohjelmiston eri ominaisuuksia, joita esiteltiin myös työturvallisuuden seminaareissa. Seminaareista ja sovelluksesta tehdystä arviointitutkimuksesta hankittu käyttäjäpalaute oli olennainen osa käyttöliittymän suunnittelussa ja käytettävyyden parantamisessa kehityksen aikana.

Työn tuloksena todetaan, että virtuaalitodellisuutta voidaan hyödyntää työturvallisuuskoulutukseen, ja se mahdollistaa erityisesti vaarallisten tilanteiden havainnollistamisen aiheuttamatta todellista vaaraa käyttäjälle. Teknologioiden nykyinen taso aiheuttaa vielä rajoituksia erityisesti hienomotoriikkaa vaativiin tehtäviin. Käyttäjäpalautteiden perusteella havaittiin, että sovelluksen helppokäyttöisyys vaati lisää työtä. Toteutettu sovellus tukee erityisesti behavioristista oppimiskäsitystä soveltavia koulutuksia, mutta selvittämättä jäi vielä tapa, jolla parantaa koulutuksia soveltamalla laajemmin muitakin oppimiskäsityksiä.

Valmistuneelle sovellukselle kehitetään jatkossa lisää koulutussisältöjä ja kehitystä jatketaan tulevaisuudessa. Virtuaalikoulutuksissa nähtiin sekä laitoksen että mahdollisten asiakkaiden puolelta huomattavaa potentiaalia. Työterveyslaitos tulee kaupallistamaan tuotteen myöhemmässä vaiheessa, se mahdollisesti viedään myös kansainvälisille markkinoille.

Avainsanat	kehitys, käytettävyys, koulutus, rakennusala, työturvallisuus, virtuaalitodellisuus (VR)

Author	Jose Uusitalo
Title	Virtual Reality Application for Construction Industry Work Safety Education
Number of Pages	72 pages + 10 appendices
Date	29 May 2018
Degree	Bachelor of Engineering
Degree Programme	Information and Communications Technology
Professional Major	Software Engineering
Instructors	Simo Silander, Senior Lecturer Kristian Lukander, Research Engineer

The subject of the bachelor's thesis was the development of a virtual reality application for use in work safety education in the construction industry. Work was done utilizing agile development methods for a product development project at the Finnish Institute of Occupational Health. The goal of the project was to determine how well suited virtual reality is for work safety education as well as the opportunities and limitations it imposes on the contents of the safety lessons. The primary objective was to get acquainted with the education provided by the Finnish safety parks that are currently used by the construction industry, use features from it in the application that have been noted as having a positive effect on learning, and fix discovered issues with it in the application. The purpose of the work was not to develop final safety lessons but to build an application where those lessons can later be implemented.

The application runs on Windows operating systems and was developed using the Unity game engine and HTC Vive virtual reality system in the company's offices in a dedicated virtual reality laboratory. Two example lessons were created to demonstrate features of the application which were also exhibited at work safety seminars. User feedback from the seminars and a study done on the application were integral during development in designing the user interface and improving usability.

As a result of the work it is determined that virtual reality can be utilized in work safety education and it specifically allows for the demonstration of dangerous situations without placing the user in actual danger. The current level of the associated technologies imposes limitations especially on tasks requiring fine motor movements and based on user feedback it was noticed that the usability of the application requires additional work. The developed software supports the behaviourist learning theory in particular, however, methods for greater adaptation of other learning theories to improve the lessons were not examined.

Development of the application will continue in the future and additional lessons will be created. Great potential was seen in virtual safety lessons by the Finnish Institute of Occupational Health and possible clients. The company will commercialize the product at a later date and may also bring it to international markets.

Keywords	development, construction industry, education, usability, virtual reality (VR), work safety

Sisällys

Lyhenteet ja käsitteet

3D	3-dimensional (engl.), kolmiulotteinen.
AR	Augmented reality (engl.), lisätty todellisuus. Katso luku 3.5.
AV	Augmented virtuality (engl.), lisätty virtuaalisuus. Katso luku 3.3.
BR	Blended reality (engl.), sekoitettu todellisuus. Katso luku 3.4.
determinismi	Tietotekniikassa deterministiset järjestelmät tuottavat aina saman tuloksen samoille syötteille.
diegesis	Videopeleissä termiä käytetään usein adjektiivina ”diegeettinen” käyttöliittymien yhteydessä. Sillä viitataan sellaisiin käyttöliittymän osiin, jotka ovat osa pelimaailman ympäristöä ja sisältöä.
DRY	Don’t repeat yourself (engl.), älä toista itseäsi. Suunnittelumalli, jossa korostetaan koodin uudelleenkäyttöä ja pyritään välttämään samankaltaisen koodin kirjoitusta moneen kertaan.
DVE	Desktop virtual environment (engl.), työpöytäkoneella oleva virtuaalinen ympäristö.
DynaMITe	Dynaaminen, multimodaalinen interaktioteknologia. Työterveyslaitoksen Topeliuksenkadun toimipisteen erään monitoimi- ja laboratoriotilan nimi, jossa työn virtuaalitodellisuuskehitys tehtiin.
entiteetti	Osa entiteetti–komponentti-järjestelmää (luku 8.1.1). Unity-pelimoottorissa luokan ilmentymä (olio), joka koostuu komponenteista, eikä sisällä merkittävää toiminnallisuutta.
ER	Extended reality (engl.), laajennettu todellisuus. Katso luku 3.6.
HMD	Head mounted display (engl.), päähän kiinnitetty näyttö(laite). Katso visiiri.

IDP	Interpupillary distance (engl.), pupillien välinen etäisyys. Katso liite 3.
immersio	Tunne, jossa käyttäjä kokee olevansa osa ympäristöä tai uppoutuvansa siihen, koska aistien havaitsema virtuaalinen tai todellinen ympäristö on yhdenmukainen ja uskottava.
IVE	Immersive virtual environment (engl.), immersiivinen virtuaalinen ympäristö. Usein virtuaalitodellisuudessa oleva kolmiulotteinen ympäristö.
komponentti	Osa entiteetti–komponentti-järjestelmää (luku 8.1.1). Unity-pelimoottorissa luokan ilmentymä (olio), joka on osa entiteettiä ja sisältää entiteetin tarvitsemaa toiminnallisuutta.
koulutusrasti	Immersiivinen virtuaalinen ympäristö, jonka sisältönä on jokin koulutus.
liikeohjain	HTC Vive -laitteiston mukana tuleva syöttölaite, ellei työn tekstissä erikseen toisin mainita. Laitteisto seuraa ohjaimen liikettä ja siirtää sen virtuaalitodellisuuteen suurella tarkkuudella.
MoSaC	Modern safety learning for construction (engl.), modernia turvallisuusoppimista rakennusalalle. Työterveyslaitoksen tutkimushanke liittyen virtuaalitodellisuudessa oppimiseen.
MR	Mixed reality (engl.), sekoitettu todellisuus tai merged reality (engl.) sulautettu todellisuus. Katso luku 3.4.
ohjain	Katso liikeohjain.
OpenVR	Open virtual reality (engl.), avoin virtuaalitodellisuus. Unity-pelimoottorin sisään rakennettu rajapinta, jonka avulla käytetään muun muassa HTC Vive -laitteistoa.

POD	Plain old data (engl.), pelkkä tai selvä data. Lähdekoodin luokka tai muu rakenne, joka ei sisällä lainkaan, tai ainoastaan hyvin vähän, logiikkaa.
rasti	Katso koulutusrasti.
SRP	Single responsibility principle (engl.), yhden vastuun periaate. Suunnittelumalli, jossa jokaisella luokalla tulisi olla vain yksi vastuu, joka on kokonaan kapseloitu sen sisään.
SteamVR	Steam virtual reality (engl.), Steam-virtuaalitodellisuus. Valve Corporationin kehittämä työpöytäsovellus, jota tarvitaan HTC Vive -laitteiston käyttöön.
VE	Virtual environment (engl.), virtuaalinen ympäristö. Katso virtuaaliympäristö.
virtuaalialue	Katso virtuaalitodellisuusalue.
virtuaalitodellisuusalue	HTC Vive -virtuaalitodellisuusjärjestelmän käyttäjälle rajattu alue oikeassa todellisuudessa, jossa käyttäjän on turvallista liikkua virtuaalitodellisuudessa.
virtuaaliympäristö	Digitaalinen tietokoneella kehitetty, usein kolmiulotteinen ympäristö. Tässä työssä termiä käytetään viittaamaan virtuaalitodellisuudessa oleviin kolmiulotteisiin immersiivisiin ympäristöihin.
visiiri	Myös virtuaalivisiiri, -lasit tai -kypärä sekä VR-lasit tai -kypärä tai HMD. Laite, jonka kautta virtuaalitodellisuutta kokeva käyttäjä näkee virtuaalisen ympäristön. Visiirissä voi olla sisäänrakennettu näyttö tai siinä voi olla pelkät linssit sen sisään laitettavaa ulkoista näyttölaitetta varten.
VR	Virtual reality (engl.), virtuaalitodellisuus. Katso luku 3.1.

VRTK	Virtual reality toolkit (engl.), virtuaalitodellisuuskirjasto. Avoimen lähdekoodin yhteisökehitetty C#-kielinen kirjasto Unity-pelimoottorille, joka rakentuu SteamVR-kirjaston päälle ja tarjoaa lukuisia Unity-komponentteja ja luokkia virtuaalitodellisuussovelluksen kehitykseen.
VR-turvapuisto	Virtuaalitodellisuusturvapuisto. Insinöörityössä esitellyn projektin ja sovelluksen kehitysnimi.
XR	X reality (engl.), määrittelemätön todellisuus tai cross reality (engl.), risteytetty todellisuus. Katso luku 3.6.

1 Johdanto

Insinöörityön aiheena on virtuaalitodellisuuskoulutussovelluksen kehitys Työterveyslaitoksen tuotekehitysprojektiin. Työterveyslaitos on työhyvinvoinnin asiantuntija, joka tutkii, palvelee ja kouluttaa. Se kehittää asiakkaiden kanssa hyviä työyhteisöjä ja turvallisia työympäristöjä sekä tukee työntekijöiden työkykyä. Sen asiakkaita ovat työpaikat, päättäjät, kansalaiset, työterveysyksiköt sekä muut työhyvinvointia kehittävät organisaatiot. Työterveyslaitoksen visio on ”Hyvinvointia työstä”, sillä terveellinen, turvallinen ja mielekäs työ luo hyvinvointia. Toimipisteitä on Helsingissä, Kuopiossa, Oulussa, Tampereella ja Turussa sekä työntekijöitä on yhteensä noin 550.

Koulutussovellukseen tuotetaan aluksi sisältönä rakennusalan työturvallisuuskoulutuksia, mutta itse sovellus toimii alustana, jolle on myöhemmin mahdollista rakentaa myös muiden alojen koulutuksia. Tavoitteena on tarjota nykyistä rakennusalan työturvallisuuskoulutusta vastaavaa tai parempaa koulutusta ja tämän kautta laskea alan työtapaturmataajuutta Suomessa ja myöhemmin myös kansainvälisillä markkinoilla. Laitoksella on useita turvallisuusalan tutkijoita, joiden sisältöosaamista hyödynnetään tulevaisuudessa koulutusrastien kehittämisessä. Rakennusala valittiin sovelluskohteeksi, koska rakennustyömailla sattuu tapaturmia tilastollisesti muita toimialoja enemmän.

Tuotekehitysprojekti on hyvin linjassa Työterveyslaitoksen arvojen kanssa. Sen tavoitteena on luoda koulutuskokonaisuus, joka voitaisiin ottaa osaksi nykyisiä työturvallisuuskoulutuksia niiden rinnalle, mutta alkuvaiheessa ei vielä pyritä korvaamaan olemassa olevaa koulutusmateriaalia. Ohjelmiston nähdään tuovan arvoa asiakasyrityksille sen digitaalisen luonteen vuoksi: sen sijaan, että työväki kuljetetaan erityiselle koulutusalueelle, koulutusalue tuodaan yrityksen tiloihin virtuaalisena. Tämän tuotekehitysprojektin lisäksi Työterveyslaitos tutkii MoSaC-hankkeessa, onko virtuaalitodellisuuden avulla mahdollista oppia turvallisuusasioita paremmin kuin tavanomaisella työturvallisuuskoulutuksella [1]. Kirjoittaja on osa hankkeen tutkimusryhmää.

Projektia ideoitiin Työterveyslaitoksella jo ennen kirjoittajan palkkausta kesätyöntekijäksi kesäkuussa 2017, mutta ohjelmistokehitys alkoi vasta sen jälkeen. Kirjoittajan lisäksi projektissa on mukana kaksi muuta henkilöä ohjaamassa sen tavoitteita, pohtimassa tulevia koulutussisältöjä ja niiden tuomia mahdollisuuksia. Projektin aikana pyritään selvittämään,

- kuinka hyvin virtuaalitodellisuus soveltuu työturvallisuuskoulutuksen tarpeisiin
- millaisia koulutuksia voidaan toteuttaa virtuaalitodellisuudessa
- mitä mahdollisuuksia virtuaalitodellisuus tuo koulutussisältöihin
- millaisia rajoituksia virtuaalitodellisuusteknologian nykyinen taso aiheuttaa koulutuksiin
- voidaanko työntekijän turvallisuuskäsityksiä muuttaa sovelluksen avulla.

Tämä insinöörityö kertoo projektiin tehdystä työstä aikavälillä 6.2017–5.2018. Projektissa ei vielä keskitytä koulutussisältöjen tuottamiseen, vaan tavoitteena on koulutuksissa tarvittavien komponenttien, työkalujen ja teknisten ratkaisujen kehitys sille tasolle, että niitä voitaisiin käyttää ilman merkittävää jatkokehitystä kaikissa tulevissa koulutuksissa. Tarkoituksena oli luoda kesään 2018 mennessä pienin toimiva tuote.

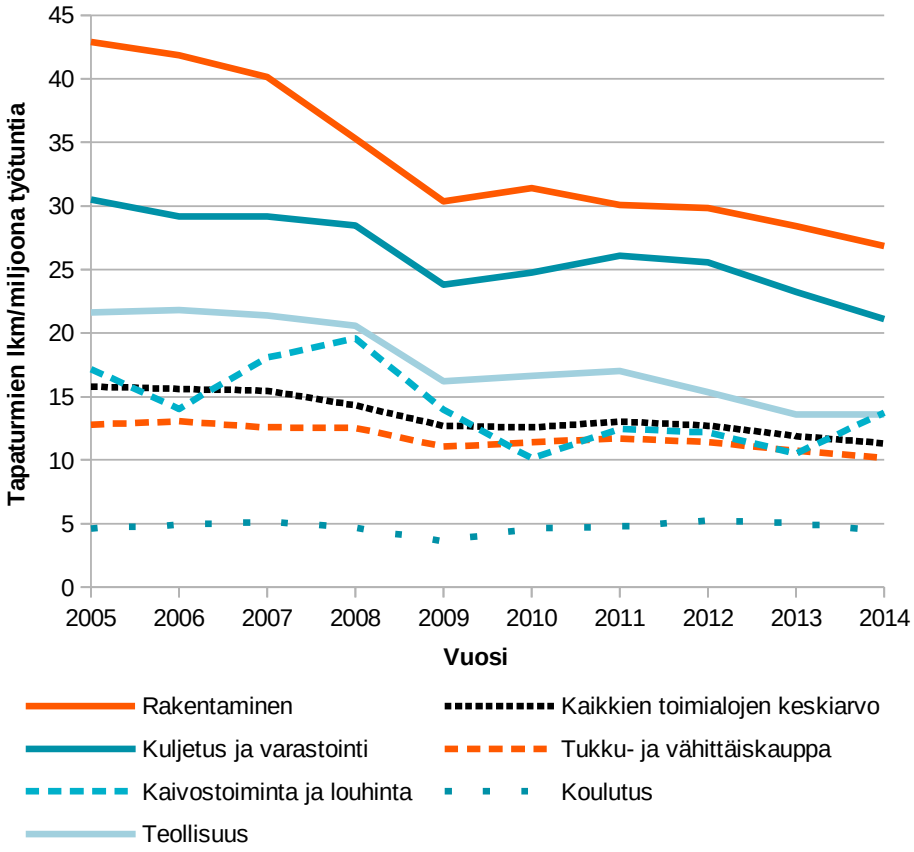
Tämä raportti rakentuu pääpiirteittäin kuudesta osasta.

1. Ensimmäinen osa (luvut 2 ja 3) antaa lukijalle tarvittavan teoriapohjan muun raportin ymmärtämiseen. Siinä kerrotaan turvapuistokoulutuksesta, jota projektissa pyritään virtualisoimaan, eri todellisuuksista ja niiden ominaisuuksista.
2. Toisessa osassa (luku 4) avataan lyhyesti tuotekehitysprojektin taustaa, projektin suunnittelua ja hallintaa.
3. Kolmannessa osassa (luku 5) lukijalle esitellään virtuaalitodellisuuslaitteistoja ja fyysistä tilaa, jossa virtuaalitodellisuuskehitys tapahtuu.
4. Neljäs osa (luku 6) vertailee virtuaalitodellisuuskehitystä tavanomaiseen ohjelmistokehitykseen ja kertoo lyhyesti kehitysympäristöstä ja käytetyistä ohjelmistokirjastoista.
5. Viidennessä osassa (luku 7) esitellään kehitetty koulutussovellus kertomalla sen tarkoituksesta, käyttäjistä, kehitetyistä koulutuksista ja käyttöliittymästä. Osan lopussa on myös lyhyesti kerrottu aikaisemmin toteutetusta käyttäjätestistä ja sen tuloksista.
6. Viimeinen osa (luku 8) on muuta raporttia teknisempi ja avaa sovelluksen ohjelmistoarkkitehtuuria korkealla tasolla, nostaa esiin kaksi sovelluksen kehityksessä ilmennyttä ongelmaa sekä esittää niiden ratkaisut.

2 Turvapuistokoulutus

Suomessa turvapuistoja on insinöörityön kirjoitushetkellä kolme: Espoossa Rudus Oy:n vuonna 2009 perustama Rudus-turvapuisto, Oulussa Pohjois-Suomen Turvapuisto Ry:n vuonna 2014 perustama Turvapuisto Pohjois-Suomi sekä Kuopiossa Pelastusopiston

vuonna 2017 perustama työturvallisuuden harjoitusalue. Turvapuistot ovat laajoja ulkoilma-alueita, joilla tehostetaan työturvallisuusosaamista erilaisten koulutusrastien avulla. Puistojen koulutuksissa keskitytään erityisesti rakennusalalle ominaisiin työtehtäviin, sillä toimialan tapaturmataajuus on suurin kaikista Tapaturmavakuutuskeskuksen seuraamista toimialoista [2, taulukko 1.3]. Kuvassa 1 nähdään tämän lisäksi myös se, että vaikka tapaturmataajuus on ollut laskussa rakennusalalla, se on silti ollut koko ajan yli kaksi kertaa keskitasoa suurempi.



Kuva 1. Palkansaajien vähintään neljän päivän työkyvyttömyyteen johtaneiden työpaikkatapaturmien taajuus valituilla päätoimialoilla vuosina 2005–2014. [2, taulukko 1.3; 3.]

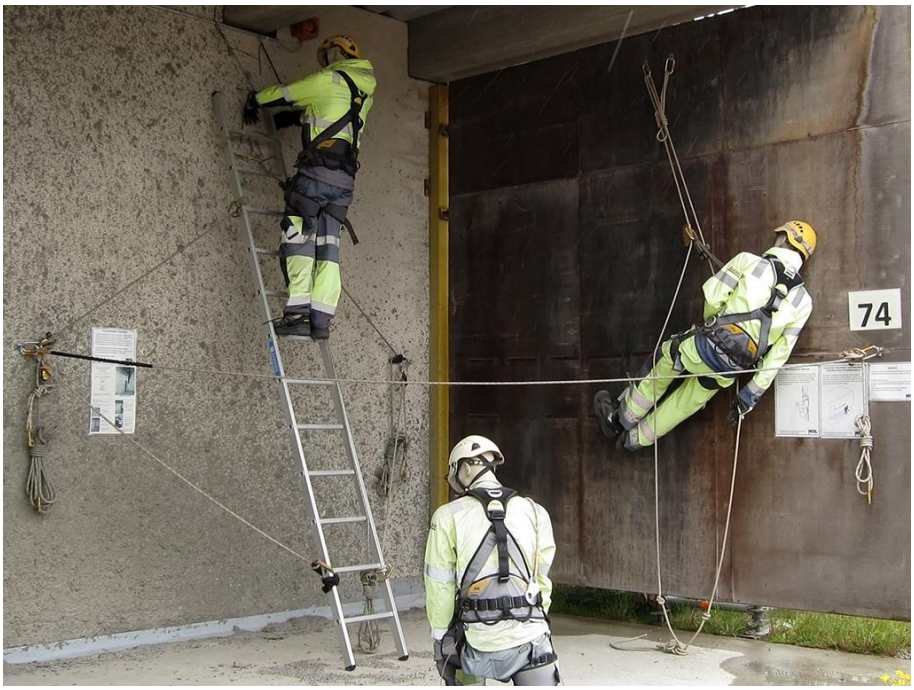
Koulutusrastit ovat usein yksittäisiä rakennelmia tai alueita, jotka on rakennettu ja lavastettu vastaamaan tavanomaisia työtehtäviä ja tilanteita rakennustyömailla. Monilla rasteilla käsitellään työtapaturmia, joissa tapaturmatilanne on havainnollistettu mallinukeilla ja tilanteeseen liittyvillä turvavälineillä, työvälineillä ja työkoneilla. Teollisuuteen liittyvien koulutusten lisäksi mukana on myös esimerkiksi yleiseen työhygieniaan, ympäristön suojeluun [4] ja johtamiseen [5] liittyviä koulutusrasteja. Kuvassa 2 näkyy useita koulutusrasteja Espoossa sijaitsevasta Rudus-turvapuistosta.

Kuva 2. Valokuva Espoon Rudus-turvapuistosta.

Turvapuistokoulutus tapahtuu ryhmissä, joissa on yleensä korkeintaan 15 henkilöä [6, s. 52], joita kouluttajat kierrättävät opiskelijoita ympäri puistoa käyden läpi useita eri rasteja yhdellä kierroksella. Useimmat koulutusrastit perustuvat teoriakoulutukseen, jossa kouluttaja selittää,

- mitä rastilla on tapahtunut
- miksi tapaturma on sattunut
- millaiset väärät työtavat ovat johtaneet tapaturmaan
- millaisia oikeita työtapoja olisi tullut käyttää tapaturman välttämiseksi.

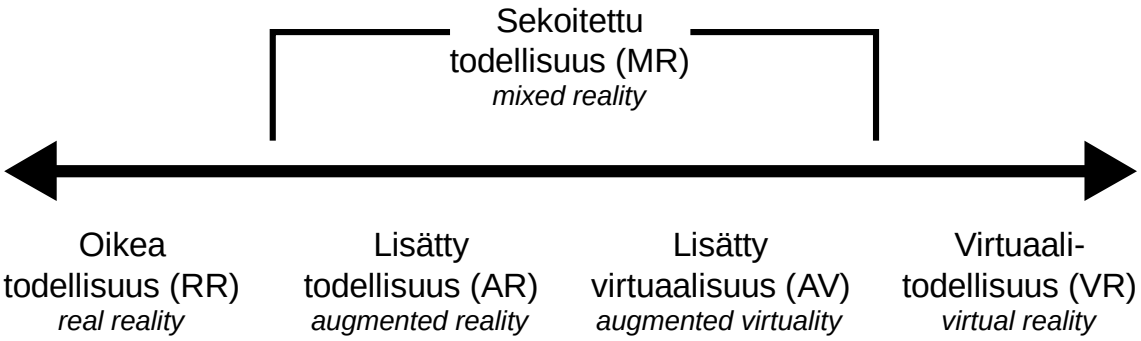
Kuvassa 3 nähdään esimerkki kahdesta tyypillisestä koulutusrastista: työtilanteet on lavastettu mallinukeilla käyttäen oikeita työkaluja, varusteita ja suojaimia. Rastilla opiskelija pääsee vapaasti tutustumaan lavastettuun sisältöön läheltä, esimerkiksi kuvassa esitetyissä rasteissa tutkimaan köysissä käytettyjä solmuja.



Kuva 3. Valokuva tikasavusteisen työskentelyn (vasen) ja roikkuvan köysityöskentelyn (oikea) koulutusrastista Espoon Rudus-turvapuistosta.

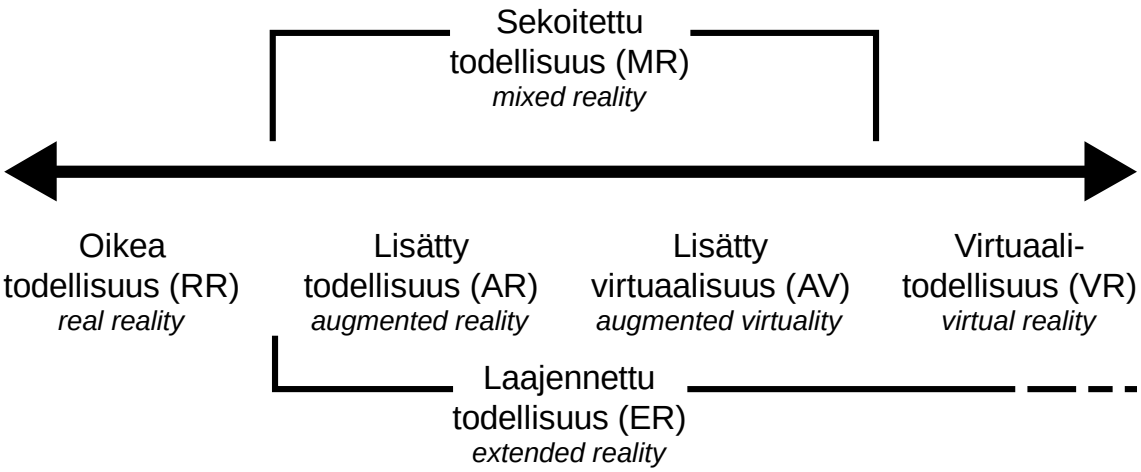
3 Todellisuudet Milgramin virtuaalisuusjatkumolla

Paul Milgramin ja Fumio Kishinon vuonna 1994 kehittämää luokittelujärjestelmää eri sekoitetun todellisuuden toteutuksille kutsutaan usein lyhyesti Milgramin virtuaalisuusjatkumoksi, joka on esitetty kuvassa 4. Kuvan järjestysasteikosta on hyvä huomata, ettei termien välillä ole tarkkaan määriteltyjä rajoja, vaan käsitteet sekoittuvat toisiinsa rajoilla. Kaavion jatkumo on myös ääretön, eikä se määrittele etäisyyksiä termien välille.



Kuva 4. Milgramin ja Kishinon virtuaalisuusjatkumo eri todellisuuksista. [Vrt. 7, kuva 1.]

Teknologia on kuitenkin kehittynyt viimeisten vuosikymmenien aikana ja mukaan on tullut laajennettu todellisuus -termi (luku 3.6). Kuva 5 kuvaa paremmin tämän hetken tilannetta todellisuuksien terminologiasta, jota noudatetaan myös tässä työssä. Laajennettu todellisuus -termin ja sen muiden muotojen määritelmä on vielä laaja, sitä käytetäänkin usein viitattaessa useaan eri luokkaan Milgramin jatkumolla.



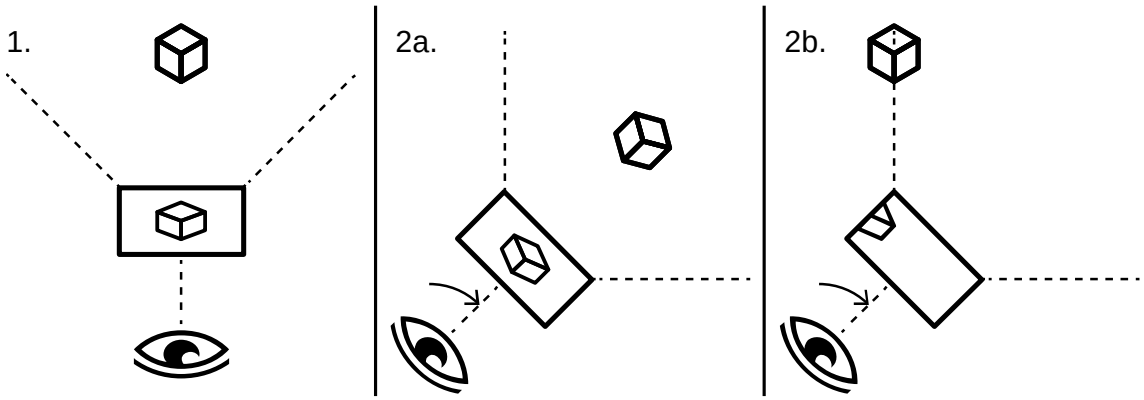
Kuva 5. Laajennettu Milgramin virtuaalisuusjatkumo eri todellisuuksista. [Vrt. 7, kuva 1.]

Englanninkielinen VR-lyhenne sekoitetaan toisinaan samannäköisiin sisarlyhenteisiin AR, MR, XR ja ER. Lyhenteiden tarkoittamat käsitteet ovat samankaltaisia, mutta niissä on pieniä eroja. Seuraavissa luvuissa määritellään laajennetulla Milgramin virtuaalisuusjatkumolla (kuva 5) esitetyt eri todellisuuksien termit.

3.1 Virtuaalitodellisuus (VR)

Virtuaalitodellisuus termille on vaikeaa kirjoittaa kaikkiin tarkoituksiin sopivaa, yksiselitteistä määritelmää. Tässä työssä kuitenkin virtuaali-, teko- tai keinotodellisuudella (engl. virtual reality, kirjainlyhenne VR) tarkoitetaan tietokoneen avulla tuotettua, kokonaan keinotekoista digitaalista ympäristöä, jossa käyttäjä voi erinäisten oheislaitteiden avulla aistia ympäristöä ja olla vuorovaikutuksessa sen kanssa. Virtuaalinen ympäristö (alan englanninkielisessä kirjallisuudessa käytetään usein lyhennettä VE, eli englanniksi virtual environment tai IVE, immersive virtual environment eli immersiivinen virtuaalinen ympäristö) on täysin riippumaton tilasta, jossa virtuaalitodellisuutta koetaan, ja sen päätavoite on luoda käyttäjälle vahva immersion ja/tai läsnäolon tunne (luku 3.1.1).

Virtuaalitodellisuuden keskeinen ominaisuus on virtuaaliympäristön näkymästä riippumaton ankkurointi. Tavanomaiset kaksiulotteiset kuvat ja videot on ankkuroitu niitä näyttävään kuvapintaan tai muuhun laitteeseen: kuvapintaa liikuteltaessa kuva näyttää liikkuvan pinnan mukana jokaiselle havaitsijalle samalla tavalla. Virtuaalisen ympäristön ankkurointi on toteutettu niin, että sitä näyttävä näyttölaite ei vaikuta ympäristön asentoon tai sijaintiin, vaan on kokonaan erillään siitä: havaitsija voi vapaasti käännellä ja liikuttaa näyttöä tilassa ilman, että ympäristö liikkuu näytön mukana. Näyttölaite luo ikään kuin ikkunan, jonka läpi voi katsoa virtuaaliympäristöön. Lisäksi jokaisella havaitsijalla on oma, muista havaitsijoista riippumaton näkymä virtuaaliseen ympäristöön. Kuvassa 6 esitetään näkymästä riippuvan (kuvasarja 1 ja 2a) ja riippumattoman (kuvasarja 1 ja 2b) ankkuroinnin ero. Näkymästä riippuvassa ankkuroinnissa, kun näyttölaitetta käännetään oikealle, ruudun keskelle piirtynyt virtuaalinen esine siirtyy näyttölaitteen mukana ja pysyy koko ajan näkymän keskellä (kuvassa 6, kuva 2a). Näkymästä riippumattomassa ankkuroinnissa, kun näyttölaitetta käännetään oikealle, ruudulle piirtynyt virtuaalinen esine ei pysy näkymän keskellä, vaan se siirtyy näkymän vasempaan laitaan (kuvassa 6, kuva 2b).



Kuva 6. Piirros näkymästä riippuvasta ankkuroinnista (kuvat 1 ja 2a) ja näkymästä riippumattomasta ankkuroinnista (kuvat 1 ja 2b).

Virtuaalitodellisuuslaitteistoissa on huomattavia eroja, ja ne pystyvät usein tuottamaan ominaisuuksia myös muista luokitelluista todellisuuksista. Jotta täytetään tässä työssä esitetty määritelmä ja todellisuuden keskeinen ominaisuus, virtuaalitodellisuusjärjestelmän vähimmäisvaatimuksiksi määritellään seuraavat kaksi ominaisuutta:

- Katselulaitteen tulee peittää käyttäjän näkökenttä niin, että käyttäjä näkee ainoastaan näkymän virtuaalitodellisuuteen, eikä ollenkaan laitteen ulkopuolista ympäristöä.

- Järjestelmän tulee seurata katselulaitteen kiertoa kolmiulotteisessa avaruudessa niin, että virtuaalitodellisuusnäkymän asento on synkronoitu fyysisen katselulaitteen vaaka-, pysty- ja sivusuuntaiseen kiertoon riittävällä tarkkuudella.

3.1.1 Immersio ja läsnäolo

Virtuaalitodellisuuskokemusten tulisi pyrkiä tuottamaan käyttäjälle mahdollisimman vahva immersion tai läsnäolon tunne, jotta ne koettaisiin miellyttävinä. Immersiolla tarkoitetaan käyttäjän aistien havaitseman virtuaalisen ympäristön yhdenmukaisuutta ja uskottavuutta. Katselija voi uppoutua kuvakaappauksessa (kuva 7) esitettyyn immersiiviseen virtuaaliseen ympäristöön toimistotuolistaan ja tutkia animaatiohahmon olohuonetta myös niiltä osin, joita ei kuvassa näy. Immersion tunne ei riipu ympäristön visuaalisesta tai auditiivisesta tarkkuudesta, eikä sen tarvitse olla oikean todellisuuden mukainen (kuvan 7 ympäristö on selvästi kuvitteellinen), mutta korkealaatuinen audiovisuaalinen kokemus parantaa immersion tunnetta.

Kuva 7. Kuvakaappaus immersiivisestä virtuaalisesta ympäristöstä virtuaalitodellisuudessa katsottavasta elokuvasta Henry (2015). [8.]

Läsnäolon tunne viittaa laajasti käyttäjän yleiseen kokemukseen virtuaalisessa ympäristössä. Se on jaettavissa eri ulottuvuuksiin käyttötarkoitusten mukaan, mutta tässä työssä huomioidaan vain kognitiivinen ja sosiaalinen ulottuvuus. Kognitiivisen läsnäolon tunteen vahvistuessa ajatukset keskittyvät enemmän virtuaaliseen kuin todelliseen ympäristöön:

kokemuksen aikaiset muistot liittyvät virtuaalitodellisuuteen oikean todellisuuden sijaan ja virtuaalisessa ympäristössä käytetään oikean todellisuuden ongelmanratkaisutapoja. Heikosti kognitiivisesti läsnä oleva käyttäjä voisi pyrkiä väistämään nopeasti lähestyvää virtuaalista autoa siirtämällä virtuaalista kehoaan ohjaimen avulla, kun taas vahvasti läsnä oleva henkilö pyrkisi hyppäämään pois tieltä oikeassa todellisuudessa. [9.] Kuvassa 8 esitetään virtuaalitodellisuudessa vahvasti kognitiivisesti läsnä oleva käyttäjä. Hän on putoamassa korkealta torninosturin puomin päältä ja valmistautumassa virtuaaliseen maahan osumiseen ottamalla leveän haara-asennon ja joustamalla polviaan oikeassa todellisuudessa, vaikka tämä ei vaikuta virtuaalitodellisuuden tapahtumiin millään tavalla. Virtuaalisen ympäristön audiovisuaalinen todentuntuisuus on tuottanut käyttäjälle vahvan immersion tunteen, ja hetkellinen kävely nosturin puomin päällä ennen putoamista on tehostanut kognitiivista läsnäoloa.

Kuva 8. Valokuva virtuaalitodellisuudessa olevasta käyttäjästä leveässä haara-asennossa.

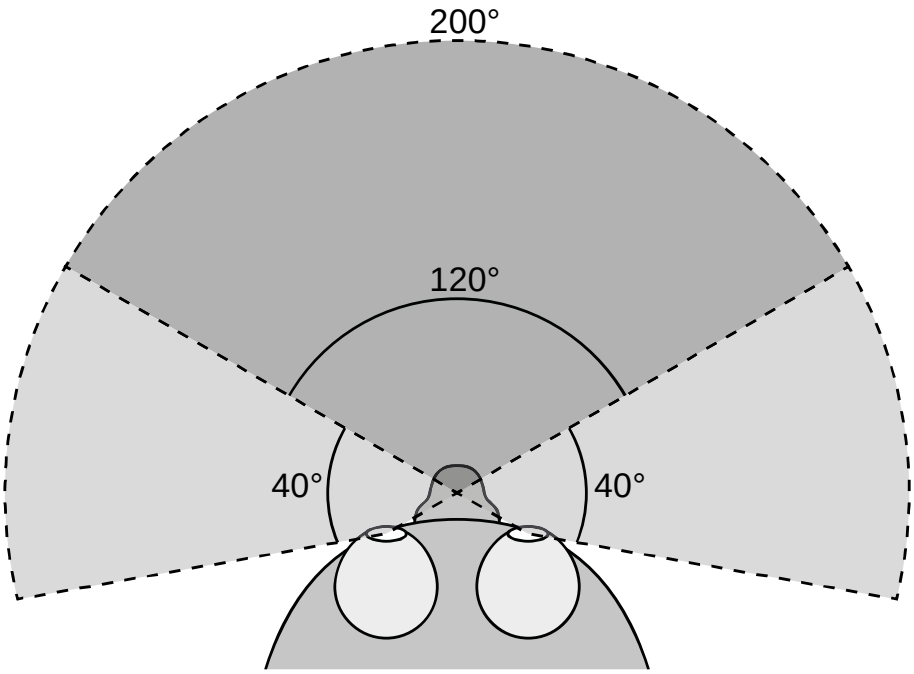
Frank Biocca määrittelee sosiaalisen läsnäolon seuraavasti:

> Sosiaalisen läsnäolon vähimmäistaso saavutetaan, kun käyttäjä tuntee, että muoto, käytös tai aistituntemus osoittaa toisen älyllisen olennon läsnäolon. Sosiaalisen läsnäolon tunne kasvaa, kun käyttäjä tuntee pääsevänsä käsiksi toisen olennon älyyn, aikomuksiin ja aistihavaintoihin. (Lainaus käännetty englannista) [10.]

Sosiaalisen läsnäolon tunteen parantaminen on haastavaa, sillä uskottavien digitaalisten ihmishahmojen kehittäminen on vaikeaa. Virtuaalitodellisuus tarjoaa tähän uusia työkaluja verrattuna työpöytäkoneilla suoritettaviin virtuaalisiin ympäristöihin (engl. desktop virtual environment, kirjainlyhenne DVE). Työpöytäkoneiden virtuaaliset ympäristöt (kuten esimerkiksi pelimaailmat) eivät ole yhtä todentuntuisia kuin virtuaalitodellisuuden tarjoamat ympäristöt, joten on mahdollista, että käyttäjät ovat näissä ympäristöissä anteeksiantavampia sosiaalisen läsnäolon suhteen. Virtuaaliympäristöjen todentuntuisuuden lisääminen tehostaa kokemusta: käyttäjät olettavat virtuaalisten hahmojen olevan todellisempia, joten hahmojen tekniset epäkohdat kuten luonnottomat liikkeet, epätarkat pintakuvat tai matalatarkkuuksiset 3D-mallit voivat heikentää sosiaalisen läsnäolon tunnetta merkittävästi.

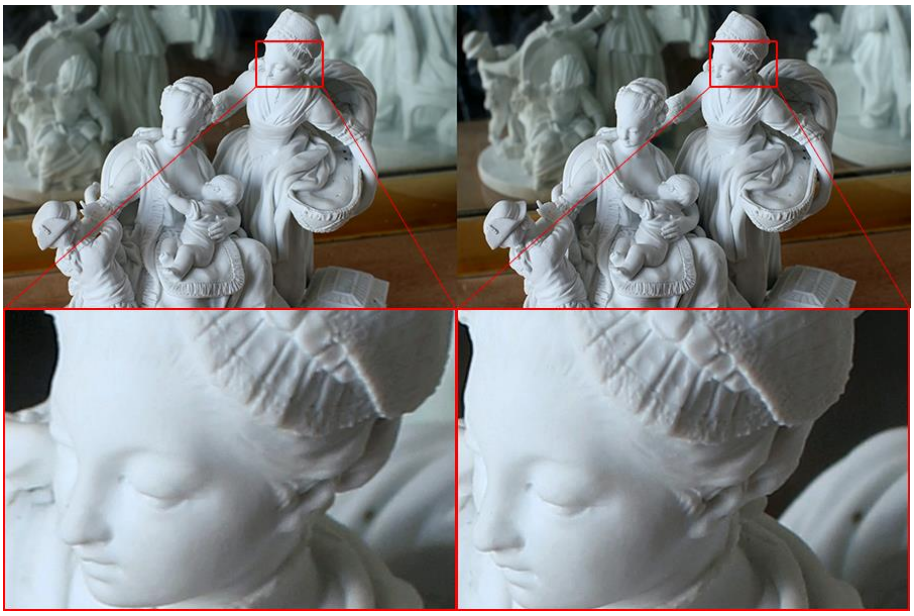
3.1.2 Stereoskooppinen kuva ja stereonäkö

Ihmisellä on kahden silmän ansiosta binokulaarinen näkö, joka mahdollistaa stereonäön. Stereonäön avulla aivojen on mahdollista rakentaa kolmiulotteisia rakenteita molempien silmien havaintojen perusteella. Näin ihmiset ja monet muut eläimet pystyvät muun muassa arvioimaan näkökentän kohteiden etäisyyttä itsestään. Stereonäkö on näkökentän keskellä, noin 120 asteen alueella, jossa molempien silmien havainnot ovat päällekkäin. Tämän alueen ulkopuolelle on noin 40 asteen monokulaarinen näköalue, jossa vain toinen silmä havaitsee ympäristöä, eikä stereonäköä ole. [11, s. 398.] Kuvassa 9 esitetään ihmisen näkökenttä, jossa eriteltynä binokulaarisen (tumman harmaa alue keskellä) ja monokulaarisen näön (vaalean harmaat alueet sivuilla) alueet. Koko näkökentän leveys on noin 200 astetta, mutta vaihtelee yksilöittäin. Huomataan, että stereonäön alue alkaa vasta nenän edestä.



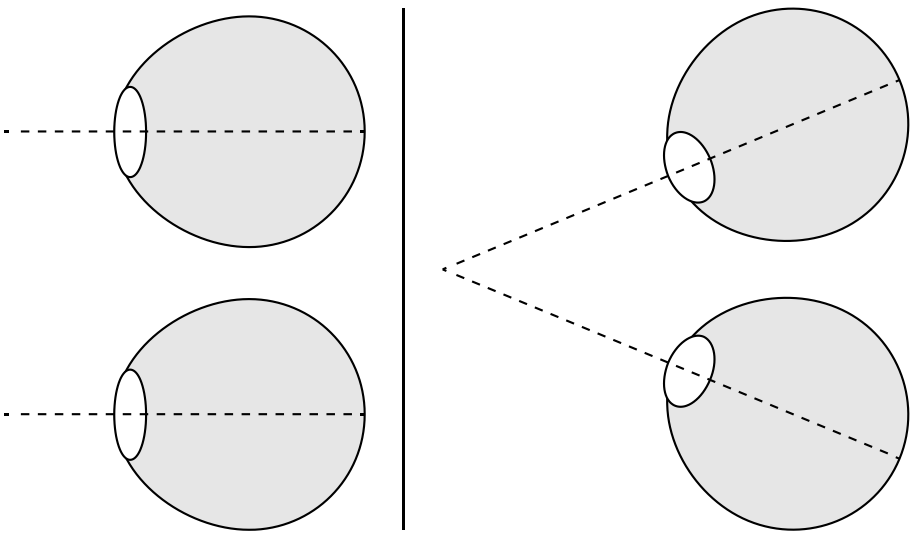
Kuva 9. Piirroksessa kuvattu ihmisen pää ylhäältä päin, silmämunat, binokulaarisen näön osuus (tumman harmaa alue keskellä) koko näkökentästä sekä monokulaariset näköalueet (vaalean harmaat alueet sivuilla).

Silmien tuottama stereokuva tarjoaa silmien kuvaeron perusteella syvyysvihjeitä, joita aivot käyttävät syvyyshavainnon luomiseen. Se muodostuu, kun aivot tulkitsevat kahta kuvaa samasta näkymästä, joiden näkökulma eroaa hieman toisistaan. Ihmisillä silmät ovat vaakatasossa, joten näkökulma muuttuu vaakatasossa, kuten kuvan 10 esittämässä stereokuvaparissa. Kuva sisältää kaksi valokuvaa samasta kohteesta, joista toinen on otettu sen jälkeen, kun kameraa on siirretty muutamia senttejä sivusuunnassa. Näin simuloidaan ihmisen silmien erotusta vaakatasossa. Kuvan alapuoliskon suurennokset on otettu samasta kohdasta molemmista valokuvista ja siitä näkee selvästi, kuinka näkymä kuvien välillä on muuttunut.



Kuva 10. Stereoskooppinen valokuva veistoksesta, johon lisätty suurennos yksityiskohdasta. [Lähdettä 12 mukaillen.]

Toinen binokulaarinen syvyysvihje on silmien konvergenssi, jolla tarkoitetaan silmien kiertymistä sisäänpäin, kohti nenää. Kun katseen kohdistaa kaukana olevaan kohteeseen, silmälihakset kääntävät silmät suoraan eteenpäin (kuvan 11 vasen puoli). Kun katseen kohdistaa lähellä olevaan kohteeseen, silmät kääntyvät sisäänpäin kuvan 11 oikean puolen mukaisesti. Aivot saavat silmälihasten ärsykkeistä lisää tietoa sen kohteen etäisyydestä, johon katse on kohdistettu.



Kuva 11. Piirros silmien konvergenssista ja akkommodaatiosta.

Binokulaaristen syvyysvihjeiden lisäksi on monokulaarisia syvyysvihjeitä, jotka eivät vaadi stereonäköä ja antavat myös tietoa havaitsijalle kohteiden etäisyydestä. Näitä ovat muun muassa:

- Akkommodaatio, joka antaa tietoa silmän linssin lihasten supistumisen avulla. Tämä näkyy myös kuvassa 11, jossa linssin muoto on pyöreämpi, kun katse on kohdistettu lähelle.
- Ilmaperspektiivi, joka antaa tietoa kohteiden kontrastin ja värien haalistumisen perusteella, johtuen auringonvalon sironnasta ilmakehässä erityisesti päiväsaikaan.
- Parallaksi, joka antaa tietoa liikkuvalle havaitsijalle kohteiden suhteellisen näennäisen liikkeen perusteella.
- Peittyvyys ja puolipeittyvyys: lähempänä olevat esineet peittävät kauempana olevat.
- Perspektiivi, joka antaa tietoa yhdensuuntaisten linjojen näennäisen kaarevuuden pohjalta.
- Pintojen rakenteen yksityiskohtaisuuden muutos: kauempana olevat pinnat näyttävät sileämmiltä ja yksinkertaisemmilta.

Nykyään kaikkien markkinoilla olevien virtuaalitodellisuusjärjestelmien teknologian olennaisena osana on stereoskooppinen kuva, eli virtuaalivisiiri tuottaa kummallekin silmälle oman kuvan, jotta stereonäön käyttö on mahdollista. Käyttäjä pystyy myös hyödyntämään monia monokulaarisia syvyysvihjeitä (kuten parallaksia tai pintojen rakenteen muutosta) virtuaalitodellisuudessa ilman, että kolmiulotteista ympäristöä näyttävään sovellukseen tarvitsee toteuttaa lisäominaisuuksia. Nämä ovat perusominaisuuksia monissa kolmiulotteisen tietokonegrafiikan piirtomoottoreissa.

3.2 Oikea tai todellinen todellisuus (RR)

Tässä insinöörityössä oikea todellisuus (engl. true reality tai real reality, kirjainlyhenne RR [13]) määritellään ihmisen aivojen tulkitsemien digitaalisesti ehostamattomien aistihavaintojen avulla koottua näkymää. Oikean todellisuuden todellisen fysikaalisen luonteen, aistihavaintojen epätarkkuuden ja aivojen tulkinnan erehtyväisyyden pohdinta on tämän työn laajuuden ulkopuolella.

3.3 Lisätty virtuaalisuus (AV)

Lisätty virtuaalisuus (engl. augmented virtuality, kirjainlyhenne AV) on Milgramin virtuaalisuusjatkumolla (luku 3) lähempänä virtuaalitodellisuutta kuin oikeaa todellisuutta. Tällä tarkoitetaan enimmäkseen virtuaalista näkymää, johon on lisätty todellisia elementtejä. [7.] Esimerkiksi virtuaaliseen kolmiulotteiseen ympäristöön voidaan lisätä reaaliaikaista videokuvaa oikeassa todellisuudessa liikkuvasta henkilöstä tai oikean todellisuuden esineitä voidaan tuoda virtuaaliympäristöön, kuten kuvassa 12 on esitetty. Kuvan vasemmalla puolella nähdään virtuaalinen pöytä, jonka päälle on laskettu virtuaalinen liikeohjain ja oikealla puolella on valokuva oikeasta todellisuudesta samasta tilanteesta, josta nähdään todellisen pöydän olevan samassa kohdassa niin virtuaalisessa kuin todellisessakin ympäristössä.

Kuva 12. Kuva virtuaalisessa ympäristössä olevasta pöydästä ja liikeohjaimesta (vasen puoli) ja valokuva oikeasta todellisuudesta samasta tilanteesta (oikea puoli).

3.4 Sekoitettu todellisuus (MR)

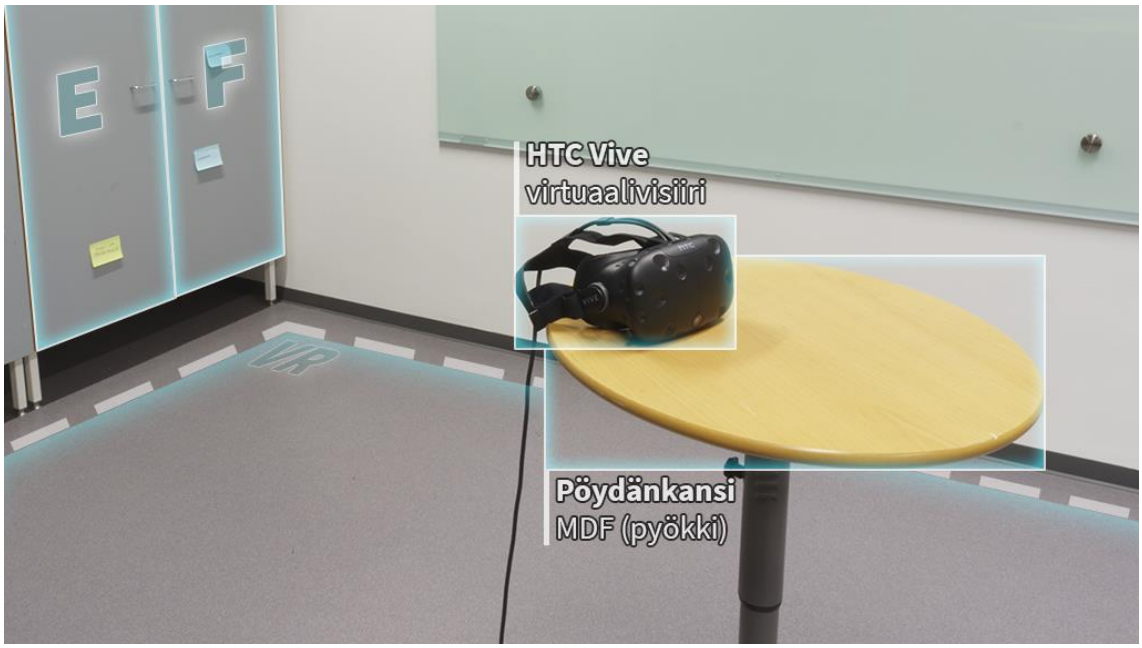
Milgramin alkuperäisellä virtuaalisuusjatkumolla sekoitetulla todellisuudella tai hybriditodellisuudella (engl. mixed reality, kirjainlyhenne MR) katetaan koko jatkumo oikeasta todellisuudesta (RR) virtuaalitodellisuuteen (VR). Tässä työssä termi kuitenkin kattaa vain lisätyn todellisuuden (AR) ja lisätyn virtuaalisuuden (AV). Toisin sanoen kaikki oikean

todellisuuden ja virtuaalitodellisuuden sekoitukset. Termiä käytetään viitattaessa järjestelmään, joka sekoittaa virtuaalitodellisuutta oikeaan todellisuuteen enemmän kuin lisätty todellisuus. Tällainen sekoitetun todellisuuden sovellus voi esimerkiksi sallia käyttäjän vuorovaikutuksen virtuaalisiin elementteihin oikeassa todellisuudessa tai virtuaalisten elementtien havaita esineitä tai olla vuorovaikutuksessa niiden kanssa oikeassa todellisuudessa. Virtuaalinen ympäristö voi myös mukautua todellisen ympäristön virtuaalialueeseen tai reagoida todellisuudessa tapahtuviin muutoksiin alueella.

Jotkin valmistajat käyttävät todellisuudesta eri termejä, koska sekoitettu todellisuus on laaja käsite. Intel Corporation käyttää termiä sulautettu todellisuus (engl. merged reality, kirjainlyhenne MR) Project Alloy -virtuaalivisiirin markkinoinnissa [14]. Sprout-tietokoneen kehittäjä HP Inc. kuvailee laitetta sekoitetun todellisuuden (engl. blended reality, kirjainlyhenne BR) tietokoneeksi [15].

3.5 Lisätty todellisuus (AR)

Lisätyllä todellisuudella (engl. augmented reality, kirjainlyhenne AR) tarkoitetaan enimmäkseen todellista näkymää, johon on tietokoneavusteisesti lisätty virtuaalisia elementtejä. Tällaiset näkymät on usein toteutettu heijastamalla tietokoneen tuottama kuva läpinäkyvälle näytölle, jonka läpi käyttäjä näkee todellisen maailman, sekä sen päällä olevat virtuaaliset elementit. Lisätyssä todellisuudessa virtuaaliset elementit reagoivat näkymän muuttumiseen, kuten esimerkiksi käyttäjän pään kääntämiseen tai ohi kulkevaan henkilöön. Järjestelmästä riippuen käyttäjä voi vuorovaikuttaa virtuaalisten elementtien kanssa, tai näkymä voi olla kokonaan passiivinen. Kuvassa 13 on simuloitu lisätyn todellisuuden näkymää, johon on lisätty virtuaalisia elementtejä kertomaan lisätietoja näkymän esineistä.



Kuva 13. Jälkikäsittelyllä simuloitu lisätyn todellisuuden näkymä, jossa oikean todellisuuden näkymään on lisätty virtuaalisia elementtejä.

3.6 Risteytetty ja laajennettu todellisuus (ER/XR)

Edellä esitettyjen termien ylijoukkona voidaan käyttää termejä laajennettu todellisuus (engl. extended reality, kirjainlyhenne ER tai XR) tai risteytetty todellisuus. (Englanniksi x reality, variable reality tai cross reality, kirjainlyhenne XR. Termissä “x reality” x-kirjainta käytetään muuttujana viittamaan tuntemattomaan tai määrittelemättömään. [16.]) Termien käyttötapa ei ole vielä vakiintunut teknologian nopean kehityksen takia, mutta tämä insinöörityö erittelee käsitteet seuraavasti.

- Laajennettu todellisuus on liukuva käsite, joka kattaa Milgramin virtuaalisuusjatkumon oikean todellisuuden oikean puolen (kuvan 5 mukaisesti) ja tarkoittaa usean eri käsitteen tai todellisuuden sekoitusta.
- Risteytetty todellisuus on diskreetti käsite, joka viittaa yhteen tai useampaan käsitteeseen Milgramin virtuaalisuusjatkumolla, mutta ei niiden sekoitukseen.

4 VR-turvapuisto-projekti

4.1 Projektin aikataulu

Sovelluksen kehitys alkoi vuoden 2017 kesäkuussa ja tarkoituksena oli kolmen kuukauden aikana selvittää, onko projektissa mahdollisuuksia jatkoa varten. Käytännössä selvitys tapahtui tekemällä kaksi koulutusrastia niin valmiiksi, että loppukäyttäjän olisi niitä mahdollista käyttää, lukemalla virtuaalitodellisuuskoulutukseen liittyviä tutkimuksia sekä suunnittelemalla ohjelmiston tulevia ominaisuuksia. Jo noin kahden kuukauden kehityksen jälkeen oli selvää, että koulutus virtuaalitodellisuudessa on mahdollista ja virtuaalitodellisuuskoulutuksessa tunnistettiin muutamia etuja verrattuna tavanomaiseen rakennusalan työturvallisuuskoulutukseen. Projektia päätettiin jatkaa alkuperäisen kolmen kuukauden lisäksi, mutta edelleen sisäisenä tuotekehitystyönä rajallisella budjetilla. Insinöörityön kirjoitushetkellä virallisen tuotekehitysprojektin on tarkoitus alkaa 2018 kesän loppupuolella. Alustavan aikataulun mukaan ensimmäinen myytävä versio ohjelmistosta valmistuu vuoden 2019 alussa.

4.2 Ohjelmiston suunnitteluprosessi

Ohjelmistokehityksen alussa ei ollut selkeää kuvaa siitä, mitä lähdetään toteuttamaan, tai siitä, millainen lopullinen tuote on. Työterveyslaitoksella ei ole aiemmin tehty vastaavaa työtä, eikä mukaan tuotu osaamista yhteistyöyritysten kautta, vaan sovellusta lähdettiin toteuttamaan sisäisenä tuotekehitysprojektina. Tarkoituksena oli päästä mahdollisimman pitkälle käyttäen ilmaisia ja omia resursseja. Lisäksi projektissa painotettiin sen kokeilevaa luonnetta: Työterveyslaitos ei ollut valmis panostamaan näin uudenlaiseen projektiin merkittäviä määriä rahaa, ennen kuin jatkosta varmistuttaisiin ja asiakaskunta löydettäisiin. Näistä syistä ohjelmistolle ei ole aluksi, tai missään myöhemmässäkään vaiheessa, tehty laajaa vaatimusmäärittelyä tai minkäänlaista teknistä suunnitelmaa. Alkuperäinen idea oli kopioida Espoon Rudus-turvapuisto virtuaalitodellisuuteen, josta projektin nimikin on peräisin: VR-turvapuisto (virtuaalitodellisuusturvapuisto).

Ohjelmiston sisällön suunnittelun malliksi otettiin turvapuistokoulutus ja Työterveyslaitoksen Turvapuistot työturvallisuuskoulutuksen oppimisympäristöinä -hankkeen loppuraportti. [6.] Ensimmäinen projektin tavoite oli kopioida ainakin yksi koulutusrasti Espoon Rudus-turvapuistosta virtuaalitodellisuuteen, joten muutama päivä projektin aloituksen

jälkeen siellä käytiin vierailulla. Turvapuistovierailun tavoitteena oli tutustuttaa projektiryhmä turvapuistokoulutukseen ja puiston alueeseen sekä valokuvata materiaalia virtuaaliympäristöjen suunnittelua ja 3D-mallien pintakuvia varten.

Vierailun lisäksi perehdyttiin Turvapuistot työturvallisuuskoulutuksen oppimisympäristöinä -hankkeen loppuraporttiin. Kyseisen tutkimushankkeen tavoitteena oli selvittää turvapuistokoulutuksen vaikuttavuus työturvallisuuden kehittymiseen tutkittavissa yrityksissä. [6.] Tutkimukseen osallistui kuusi eri yritystä talonrakennus-, korjausrakennus- ja palvelualoilta. Turvapuistokoulutuksiin osallistuneita työntekijöitä ja esimiehiä haastateltiin koulutusten jälkeen, näiden haastattelujen tulokset olivat erittäin tärkeitä ohjelmiston suunnittelussa. Tulosten pohjalta tehtiin kaksi taulukkoa (liitteet 1 ja 2) ominaisuuksista, jotka haluttiin toteuttaa sovelluksen ensimmäiseen myytävään versioon. Raportin taulukoiden 2 ja 3 [6, s. 22, 27] rivit jaettiin kolmeen luokkaan (maininneita yrityksiä -sarake) sen perusteella, kuinka moni kuudesta yrityksestä oli maininnut asian haastatteluissa: rivit laitettiin tärkeysjärjestykseen luokan mukaan (enemmän mainintoja on tärkeämpi) sekä asiaan kuulumattomat rivit poistettiin. Lopuksi toteutustapa-sarakkeeseen kirjoitettiin lyhyt suunnitelma siitä, miten jokainen ominaisuus toteutetaan.

Liitteessä 1 (Turvapuistokoulutuksen oppimista tukevien havaintojen ominaisuustaulukko) on ensimmäinen edellä mainituista taulukoista, jossa on tutkimuksen haastatteluissa esiintyneet havainnot sellaisista turvapuistokoulutusten piirteistä, joiden koettiin tukevan oppimista. Nämä ominaisuudet haluttiin toteuttaa sovelluksessa, jotta virtuaalitodellisuuskoulutus olisi yhtä tasokas kuin todellinen turvapuistokoulutus. Liitteessä 2 (Turvapuistokoulutuksen kehitysehdotusten ominaisuustaulukko) on toinen edellä mainituista taulukoista, jossa on tutkimuksen haastatteluissa esiintyneiden kehitysehdotusten lisäksi havainnot sellaisista turvapuistokoulutusten piirteistä, joiden koettiin haittaavan oppimista. Näiden ominaisuuksien toteutus tekisi virtuaalitodellisuuskoulutuksesta laadukkaampaa kuin todellisesta koulutuksesta. Taulukoita on myöhemmin päivitetty insinöörityön kirjoituksen yhteydessä, josta kerrotaan lisää luvussa 9.

4.3 Projektinhallinta

Projektin alussa harkittiin Scrum-viitekehyksen käyttöä, mutta lopulta päädyttiin kevyempään Kanban-viitekehykseen, koska projektissa on ainoastaan yksi ohjelmistokehittäjä.

Projektinhallintatyökaluina Työterveyslaitoksella käytetään usein Trello-palvelua Kanban-taulujen luomiseen, sekä Flowdock-pikaviestinpalvelua sähköpostin lisäksi viestimiseen. Tehtävät on jaettu Trellossa 15 tauluun ja kolmeen kategoriaan taulukon 1 mukaisesti. Kehityksen aikana tauluihin on kertynyt satoja tehtäviä.

Taulukko 1. Projektin Kanban-taulujen jakautuminen eri kategorioihin.

Kategoria	**Taulu**
tekemättömät tehtävät	hankinnat, eli tulevaisuudessa ostettavat ohjelmistot yms.
	ideat tuleville koulutuksille (kaksi taulua)
	koulutus- ja elämysrastien tekniset kehitystehtävät (seitsemän taulua)
	yleiset sovelluksen ytimen tekniset kehitystehtävät ja muu tutkimustyö
	käyttöliittymän tekniset kehitystehtävät
työn alla olevat tehtävät	työn alla olevat tehtävät
tehdyt tehtävät	kirjoittajan mielestä valmiit tehtävät
	kollegan valmiiksi varmistamat tehtävät, eli tuotannossa olevat ominaisuudet

Trello-projektin taulujen jako on muotoutunut usean kuukauden kehityksen aikana. Alkujaan tauluja oli vain kolme: tekemättömät tehtävät, työn alla olevat tehtävät ja valmiit tehtävät. Pian kehityksen alettua kollega kuitenkin halusi myös itse varmistaa, että kaikki tehdyiksi väitetyt tehtävät olivat oikeasti valmiita, joten varmistetuille tehtäville lisättiin oma taulu. Kehityksen edetessä uusien koulutusten ja koulutusideoiden määrän kasvaessa yksittäinen tekemättömät-taulun tehtävät eroteltiin rastien mukaan omiin tauluihin. Tässä järjestelmässä on myös se etu, että taulusta näkee nopeasti, kuinka monimutkainen tai kesken rastin kehitys on. Myöhemmässä projektin vaiheessa oli tarpeen ostaa ohjelmistoja ja laitteistoa projektin käyttöön, joten hankinnoille tehtiin oma taulu.

5 Virtuaalitodellisuuslaitteistot ja projektin tilaresurssit

5.1 Modernit virtuaalitodellisuusjärjestelmät

Kaikki vuoden 2013 jälkeen julkaistut virtuaalitodellisuusjärjestelmät voidaan jakaa kahteen eri kategoriaan ja neljään eri luokkaan taulukon 2 mukaan. Luokittelu perustuu siihen, missä laitteessa virtuaalista ympäristöä suoritetaan, sekä laitteen suoritustehoon, joka laskee luokkanumeron kasvaessa.

Taulukko 2. Virtuaalitodellisuusjärjestelmien nelitasoinen luokittelu.

	Kategoria		**Luokka**	
Suoritusteho ⇧	A	Tietokonelaitteet	1	Tietokonepohjaiset laitteet
			2	Pelikonsolipohjaiset laitteet
	B	Mobiililaitteet	3	Itsenäiset laitteet
			4	Älypuhelinpohjaiset laitteet

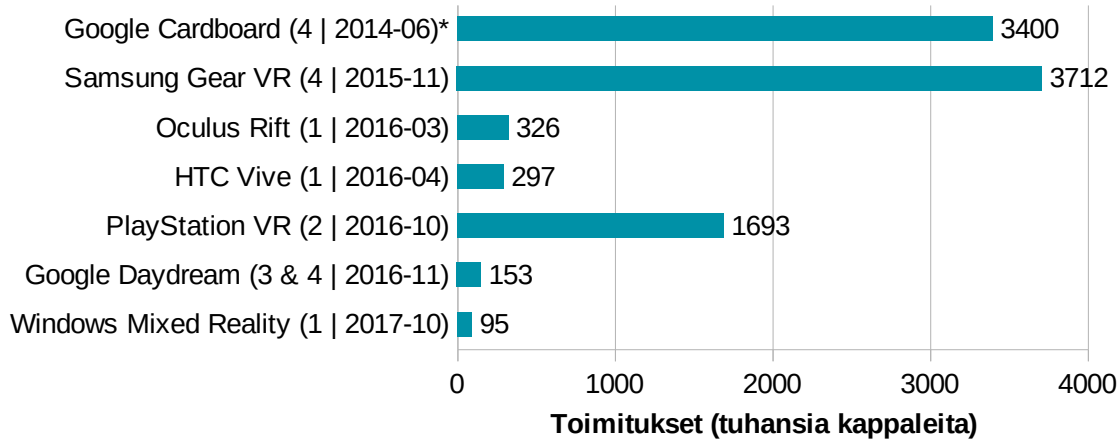
Eri kategorioiden laitteissa on merkittäviä eroja erityisesti suoritustehon, mutta myös niiden tarjoaman liikkeenseurannan teknologisen toteutuksen ja tarkkuuden suhteen. Kategorioiden sisällä laitteiden keskinäiset erot pienenevät. Joskin itsenäiset virtuaalitodellisuusjärjestelmät ovat vielä uusia markkinoilla, joten niiden suoritustehon merkittävä kasvu on mahdollista. Teknologian kehittyessä itsenäisten laitteiden suoritusteho kasvanee ja alkaa vastaamaan pelikonsolien tasoa. Kaikissa laitteissa luokasta riippumatta on kuitenkin yksi ominaisuus, jota tässä insinöörityössä pidetään virtuaalitodellisuuslaitteiston määrittelevänä ominaisuutena: laitteen visiirin kierron seuranta, jonka avulla käyttäjä pystyy vapaasti katselemaan ympärilleen virtuaalitodellisuudessa.

Tietokonetta (esimerkiksi HTC Vive) ja pelikonsolia (esimerkiksi Sonyn PlayStation VR) käyttävissä järjestelmissä kaikki laskenta tehdään ulkoisessa laitteessa ja itse virtuaalivisiiri toimii pääasiassa pelkkänä näyttönä, joka näyttää toisen laitteen rakentaman kuvan. Kuva voidaan siirtää virtuaalivisiiriin langattomasti (esimerkiksi TPCastin langattomalla adapterilla), mutta usein visiiri on kiinni emolaitteessa kaapeleilla.

Itsenäisissä laitteissa virtuaalitodellisuuden kaikki laskenta tehdään virtuaalivisiirin sisäänrakennetulla, vielä älypuhelimen tasoisella tietokoneella (esimerkiksi Qualcommin

Snapdragon VR820). Etuna tietokonepohjaisiin laitteisiin on kaapelien puuttuminen, mutta haittapuolena taas on pienempi laskentateho ja laitteen tukeutuminen ladattavaan akkuun verkkovirran sijaan.

Älypuhelinpohjaiset järjestelmät (esimerkiksi Samsungin Gear VR) ovat selvästi muita laitteita halvempia, sillä niissä ei ole lainkaan sisäänrakennettua näyttöä. Virtuaalivisiirin sisään asetettava älypuhelin toimii sekä virtuaalitodellisuutta pyörittävänä suorittimena että visiirin näyttönä. Tällaiset järjestelmät ovat pikemminkin lisävaruste älypuhelimelle, mikä voi selittää niiden kuvasta 14 nähtävän suosion. Laitteiston julkaisukuukausi ja taulukon 2 mukainen luokkanumero on suluissa laitteen nimen jälkeen. Laitteet on järjestetty julkaisukuukauden mukaan ylhäältä alas.



Kuva 14. Valittujen virtuaalitodellisuusjärjestelmien toimitukset vuonna 2017. [17, 20.]

5.2 HTC Vive -virtuaalitodellisuusjärjestelmä

Projektin kehityksessä käytetään HTC Corporationin ja Valve Corporationin kehittämää HTC Vive -virtuaalitodellisuusjärjestelmää. Laitteisto oli hankittu muutamia kuukausia ennen projektin alkua vuoden 2017 alussa, se oli hankinnan aikana markkinoiden parhaimpia virtuaalitodellisuusjärjestelmiä. Projektissa halutaan käyttää mahdollisimman tehokasta virtuaalitodellisuusjärjestelmää ja tietokonetta, jotta on mahdollisuus kehittää visuaalisesti ja teknisesti näyttävä tuote. Visuaalista näyttävyyttä voidaan myöhemmin tarpeen vaatiessa heikentää esimerkiksi siirryttäessä älypuhelinpohjaisille alustoille, joilla suorituskyky on heikompi. Projektin pääpaino on kuitenkin kehittää tuote yksinomaan tietokonepohjaiselle virtuaalitodellisuusjärjestelmälle. Koska tuotetta kehitetään ensin

yritystenvälisille markkinoille ja ensisijaisesti kuluttajille myytävälle virtuaalitodellisuuslaitteistolle, on Työterveyslaitos kehittäjän roolissa uniikissa asemassa, jossa asiakasyritysten voidaan vaatia käyttävän juuri HTC Vive -laitetta. Tämä vähentää kehityskustannuksia, sillä ei ole tarpeen kehittää sovellusta usealle eri laitteistolla, eikä kuluttajille suunnatun Viven hankinta ole suuri kulu yritykselle.

HTC Vive -virtuaalitodellisuusjärjestelmä sisältää kolme olennaista osaa: kaksi tukiasemaa, kaksi langatonta liikeohjainta ja langallisen virtuaalivisiirin. Järjestelmän laitteiden liikkeenseuranta on toteutettu sisältä–ulos-tekniikalla: laitteen sijainti lasketaan laitteessa olevan, ulkoisista majakoista ärsykkeitä saavan anturin datan perusteella. Ulkoa–sisään-tekniikka (jota käytetään esimerkiksi Oculus Rift -laitteessa) toimii taas päinvastoin: laitteen sijainti lasketaan laitteen ulkopuolisen, laitteesta ärsykkeitä saavan anturin datan perusteella. Molempien tekniikoiden toteutukseen on erilaisia ratkaisuja, kuten infrapunavalo tai kuva-analyysi, mutta ulkoa–sisään-tekniikka vaatii aina erillisen ulkoisen anturin itse virtuaalitodellisuuslaitteen lisäksi.

Vive toteuttaa liikkeenseurannan infrapunavalon, gyroskoopin ja kiihtyvyysanturin avulla. Tukiasemat sisältävät kaksi nopeasti pyörivää infrapunalaseria, jotka ovat 90 asteen kulmassa toisiinsa nähden ja pyyhkäisevät vuorotellen huoneen infrapunavalolla. Laserien lisäksi tukiasema lähettää myös koko huoneen täyttävän infrapunavalopulssin tietyin väliajoin. Liikeohjaimissa ja virtuaalivisiirissä on molemmissa lukuisia pieniä infrapuna-antureita (kuvan 15 ovaalit syvennykset virtuaalivisiirissä ja liikeohjaimissa), jotka virittyvät tukiaseman infrapunalaserin pyyhkäisystä ja pulssista. Molemmissa liikkeenseurantatekniikoissa ongelmana on laitteen näkölinjan katkeaminen tukiasemaan. Viven laitteet ratkaisevat ongelman seuraamalla omaa sijaintiaan gyroskoopilla ja kiihtyvyysanturilla, kun infrapunavalo on estetty [19].

Kuva 15. Kuva koko HTC Vive -virtuaalitodellisuusjärjestelmästä. [20.]

Langattomat liikeohjaimet vaativat virtuaalivisiiriä toimiakseen, sillä ne lähettävät anturidatan Valven omalla protokollalla virtuaalivisiirille [21]. Visiiri edelleen lähettää ohjainten ja oman anturidatan tietokoneelle, jossa laitteiden sijainti avaruudessa lasketaan alle millimetrin tarkkuudella [22].

5.3 Virtuaalitodellisuuslaboratorio

Projektissa päätettiin jo alusta alkaen lähteä kehittämään koko huoneen kokoisia (engl. room-scale) virtuaalitodellisuusympäristöjä, joissa käyttäjä pääsee vapaasti liikkumaan. Toinen mahdollisuus oli luoda koulutuksia, joissa käyttäjä istuu koko ajan tuolilla tai seisoo paikallaan eikä liikkuminen ole mahdollista. Paikallaan pysyvien virtuaalitodellisuuskoulutusten kehitys olisi helpompaa, halvempaa ja nopeampaa, sekä ne olisivat huomattavasti helpommin siirrettävissä muille alustoille (kuten älypuhelinpohjaisiin virtuaalitodellisuusjärjestelmiin), mutta yksi projektin tavoitteista on kehittää immersiivisiä koulutuksia. Vapaa liikkuminen suurella alueella vahvistaa kognitiivisen läsnäolon tunnetta, joten oli selvää, että tarvitsemme virtuaalitodellisuuskehitystä varten oman kokonaisen huoneen, jonne rajataan esteistä tyhjä virtuaalitodellisuusalue.

Virallisesti HTC Vive tukee virtuaalitodellisuusalueita 1,5 m × 2,0 m alueesta noin 3,5 m × 3,5 m kokoiseen alueeseen [23, s. 64], eikä valmistaja takaa liikkeenseurannan tarkkuutta suuremmilla alueilla. Alueen lattian tulee olla tasainen ja täysin tyhjä kaikista esteistä, jotta vältetään käyttäjän kompastuminen, sillä virtuaalitodellisuudessa liikuttaessa käyttäjä ei näe lainkaan todellista tilaa, missä hän liikkuu. Työterveyslaitoksen päätoimi-

pisteessä Helsingin Topeliuksenkadulla on tarkoitukseen sopiva monitoimi- ja laboratoriotila (Kutsutaan DynaMITe-laboratorioksi, esitetty kuvassa 16. Nimi on akronyymi sanoista dynaaminen, multimodaalinen interaktioteknologia.), jonne virtuaalitodellisuusjärjestelmä oli pysyvästi asennettu jo ennen projektin alkua. Kuvassa näkyy virtuaalivisiirin (keskellä) lisäksi myös ylhäällä keskellä Viven musta neliönmuotoinen tukiasema. Vaikka alueen koko on suurempi kuin laitevalmistajan suositus (noin 4,8 m × 3,8 m), liikkeenseurannassa ei ole kehityksen aikana ilmennyt ongelmia.

Kuva 16. Valokuva Työterveyslaitoksen Topeliuksenkadun DynaMITe-laboratoriosta, jonne HTC Vive -laite on asennettu.

Alueen rajaus tai koko ei kuitenkaan ole fyysinen este virtuaalitodellisuudessa liikkumiselle. Virtuaalitodellisuudessa on mahdollista vapaasti liikkua myös rajatun alueen ulkopuolella, ja ohjelmasta riippuen käyttäjä voi saada jonkinlaisen palautteen virtuaalitodellisuusalueelta poistumisesta.

- Käyttöliittymään voi ilmestyä varoitusviesti.
- Havahduttavan tehoste, kuten ruudun himmeneminen tai kokonaan sammuminen, kiinnittää käyttäjän huomion.
- Vivellä myös visiirin edessä olevan kameran automaattinen käynnistyminen on mahdollista, jolloin käyttäjä näkee kameran kuvaa virtuaalitodellisuudessa.

Virtuaalitodellisuusalueen tarkoituksena on vain antaa käyttäjälle tietoa ja palautetta siitä alueesta, millä on turvallista liikkua. Virtuaalinen ympäristö voi muuttua virtuaalitodellisuusalueen koon ja muodon perusteella. Tässä projektissa tällaista ominaisuutta ei koettu tärkeäksi vielä tässä vaiheessa, sillä Työterveyslaitoksen on mahdollista määritellä tarkkaan laitteisto ja alueen koko, millä ohjelmistoa tulee käyttää. Myöhemmässä vaiheessa, kun ohjelmisto tulee laajempaan jakeluun, sovelluksen tulee huomioida käyttäjän virtuaalitodellisuusalueen koko ja varmistaa, että kaikki tarvittavat interaktiiviset kohteet virtuaalitodellisuudessa ovat käyttäjän saavutettavissa alueen sisällä.

6 Virtuaalitodellisuusohjelmistokehitys

Virtuaalitodellisuuskehitys eroaa hieman perinteisestä ohjelmistokehityksestä. Perinteisessä ohjelmistokehityksessä käyttöjärjestelmän valinta on yksi tärkeimmistä valinnoista niin kehityksen kuin markkinoinninkin kannalta. Ohjelmistojen siirtäminen toiselle käyttöjärjestelmälle on mahdollista, mutta se vaatii aikaa ja kehitystyötä. Kun tehdään maksullisia ohjelmistoja, on tärkeää tehdä arvio siitä, kuinka paljon lisätuottoa ohjelmiston vieminen uudelle markkinalle (käyttöjärjestelmälle) tuo ja kattaako se edes siirtämiseen käytettyjä kehityskuluja. Vaikka itse tietokoneiden suoritusteho vaihtelee merkittävästi, eri tietokoneet ovat ulkoisilta toiminnoiltaan ohjelmistojen kannalta erittäin homogeenisiä: on turvallista olettaa, että jokaisella käyttäjällä on samat syöttölaitteet, eli näppäimistö ja hiiri, joissa molemmissa on tietyt vähimmäisominaisuudet. Harvoin kehittäjät myös pohtivat, onko käyttäjällä äänentoistolaitetta.

Pelikonsolit ovat vielä enemmän homogeenisiä kuin tietokoneet, sillä yhdelle konsolille tehty ohjelmisto pyörii usein vain juuri sillä laitteella. Nykyisin myös uudemmat pelikonsolit voivat suorittaa pelejä, jotka on tehty laitteen vanhemmille versioille, mutta yhteensopivuus ei toimi toiseen suuntaan. Tietyn konsolin peliohjaimissa esiintyy harvoin suuria eroja käytettävien nappien suhteen ja eri konsolienkin ohjainten väliset erot ovat pieniä. Koska konsoleja käytetään yleensä televisioiden kanssa, äänentoistolaitteen olemassaolo on käytännössä varmaa.

Virtuaalitodellisuuskehittäjän rooli on pelikonsoliohjelmistokehittäjän ja tietokoneohjelmistokehittäjän välillä, eikä se sovi hyvin kummankaan nimikkeen alle. Virtuaalitodellisuuskehittäjät ovat itsenäinen joukkonsa, vaikka heidän ohjelmistot pyörivätkin tietoko-

neilla ja älypuhelimilla: rooli erottautuu muista syöttölaitteen perusteella. Heidän on tehtävä vaikeampia valintoja ohjelmistonsa ominaisuuksien suhteen kuin edellä mainitut kehittäjät ja heidän on hankalampi tehdä oletuksia käyttäjien laitteistosta. On toki mahdollista luoda yksinkertainen sovellus, jossa ainut virtuaalitodellisuutta hyödyntävä ominaisuus on visiirin kierron seuranta: tällainen sovellus toimii (jakelualustaa tai pelimoottoria huomioon ottamatta) jokaisella virtuaalitodellisuusjärjestelmällä, mutta samalla rajoitetaan sovelluksen monimutkaisuutta ja houkuttelevuutta merkittävästi.

Markkinoilla on nykyään lukuisia erilaisia virtuaalitodellisuusjärjestelmiä, joissa on monenlaisia ominaisuuksia. Lähdettäessä kehittämään virtuaalitodellisuutta hyödyntävää ohjelmistoa, on tärkeää pohtia jo hyvissä ajoin, mitä kehitettävä ohjelmisto vaatii itse virtuaalitodellisuuslaitteistolta. Liitteessä 3 (Virtuaalitodellisuuskehityksessä huomioitavia asioita) on listattu kysymyksiä, joita virtuaalitodellisuuskehittäjän tulisi pohtia ennen ohjelmistokehityksen aloitusta.

6.1 Unity-pelimoottori

Projektin alkuvaiheessa oli tärkeää selvittää mahdollisimman nopeasti, kuinka hyvin virtuaalitodellisuus palvelee projektin tarpeita sekä oliko projekti edes laajemmin toteutettavissa. Tästä syystä ilmaisen, että yleiskäyttöisen pelimoottorin käyttö oli ainoa järkevä vaihtoehto. Vuoden 2017 kesällä markkinoilla oli neljä pelimoottoria, jotka tukivat Viveä OpenVR-rajapinnan kautta:

- Unity
- Unreal Engine 4
- CRYENGINE
- Torque 3D.

Unity Technologiesin kehittämä Unity oli helppo valinta, koska kirjoittaja oli projektin ainoa ohjelmistokehittäjä ja hänellä oli jo ennestään kokemusta pelimoottorin käytöstä. Unityn Asset Store tarjoaa myös laajan valikoiman ilmaisia lisäosia, joiden odotettiin nopeuttavan kehitystä erityisesti projektin alkuvaiheessa.

Unity on pelimoottori, jonka avulla on mahdollista kehittää kaksi- ja kolmiulotteisia videopelejä ja muita sovelluksia tietokoneille, pelikonsoleille ja mobiililaitteille. Pelimoottori tar-

joaa laajan valikoiman työkaluja pelinkehitykseen sekä valmiita toteutuksia usein peleissä tarvittuihin ominaisuuksiin esimerkiksi fysiikkamoottorin, käyttöliittymäjärjestelmän, tuen eri syöttölaitteille ja kameranhallintalogiikan. Ohjelman tarkoituksena on helpottaa ja nopeuttaa pelinkehitystä merkittävästi ja tarjota kehittäjälle lähtökohdan tai pohjan, jonka päälle rakentaa oma sovellus.

6.2 Kehitysympäristö ja ohjelmistokirjastot

Työterveyslaitoksen kehityskoneilla on Microsoft Corporationin kehittämä Windows 10 -käyttöjärjestelmä ja Unity-pelimoottorin mukana toimitetaan Microsoft Visual Studio -ohjelmointiympäristö. Visual Studion mukaan on paketoitu Visual Studio Tools for Unity -lisäosa, joka tarjoaa tuen lukea Unityn dokumentaatiota suoraan ohjelmointiympäristöstä, laajennetun tuen Unityn C#-viestijärjestelmän syntaksikorostukselle ja automaattitäydennykselle, debugaustyökalut peleille yms. Ohjelmointikieleksi valikoituu C#, sillä tarvittuja kirjastoja ei ole toteutettu JavaScript-kielellä.

Valve Corporation on yksi Vive-virtuaalitodellisuusjärjestelmän kehittäjistä, ja he ovat tehneet Unity-pelimoottoriin C#-kielisen SteamVR-ohjelmistokirjaston, jonka avulla Unity-pelimoottori saa tuen HTC Vivelle Valven OpenVR-rajapinnan kautta. Viveä voi käyttää pelkän SteamVR-kirjaston kanssa, mutta kehityksen nopeuttamiseksi käyttöön otettiin myös VRTK-C#-kirjasto, jossa on useita valmiita komponentteja virtuaalitodellisuudessa tarvittaviin toimintoihin. VRTK-kirjasto käyttää SteamVR-kirjaston tarjoamaa rajapintaa, jonka kautta se viestii Vive-laitteiston kanssa.

6.3 Yksikkötestaus ja versionhallinta

Projektin alkuvaiheessa keskityttiin tuottamaan mahdollisimman paljon sisältöä ja ominaisuuksia lyhyessä ajassa, joten yksikkötestaus jäi kokonaan pois suunnitelmista. Myös projektin kokeilevan luonteen ja alun epävarman tulevaisuuden takia yksikkötestaamiseen ei ole käytetty aikaa alkukuukausienkaan jälkeen, vaan on keskitytty uusien toiminnallisuuksien kehittämiseen ja kokeilemaan erilaisia teknisiä ratkaisuja. Versionhallintaan käytetään Unityn käyttöliittymään sisäänrakennettua Collaborate-palvelua sen helppokäyttöisyyden takia.

7 VR-turvapuisto-sovellus

7.1 Koulutussovellus

Päällisin puolin sovelluksen kehitys muistuttaa suuresti tavanomaista pelinkehitystä: ohjelmiston ytimenä on valmis pelimoottori, päivittäisessä kehityksessä käytetään laaja-alaisesti muitakin taitoja pelkän ohjelmoinnin lisäksi, kuten 3D-mallinnusta, valokuvien käsittelyä tekstuureiksi, äänitehosteiden muokkausta ja tekemistä, sekä alustavien turvallisuuskoulutusten suunnittelua, joka peliteollisuudessa vastaa hyvin tehtävien ja seikkailujen suunnittelua. Projektin koulutusympäristöt voivat keskivertokäyttäjän näkökulmasta muistuttaa videopeliä – ohjelmisto ei kuitenkaan ole videopeli, vaikka sen kehitykseen peliteollisuuden taitoja ja työkaluja hyödynnetäänkin.

Sovelluksen rooli on hyvin erityinen. Se on digitaalinen työkalu, jonka avulla käyttäjälle opetetaan haluttuja asioita käyttäen ennalta toimiviksi todettuja mekanismeja. Työkaluna ohjelmisto on kokonaan omanlainen opetusvälineensä, jonka rinnalla ovat esimerkiksi kirjat, digitaaliset diaesitykset ja sosiaalinen kanssakäyminen. Sovelluksen käyttämisen ei välttämättä tarvitse olla hauskaa, ympäristöjen ei tarvitse olla visuaalisesti näyttäviä, eikä käyttäjälle ole välttämätöntä tarjota mukaansatempaavaa juonta tai hahmoja. Tällainen työkalu kyllä täyttäisi behavioristisen oppimiskäsityksen tiedonsiirron kriteerin [24], mutta on tarkoituksenmukaista luoda käyttäjälle miellyttävä oppimisympäristö, jossa oppia uutta. Yksi projektin tavoitteista on luoda opetustyökalu, jota on vaivatonta ja miellyttävää käyttää riippumatta käyttäjän teknisistä taidoista. Käyttäjää pyritään motivoimaan positiivisella palautteella ja tulevien koulutusten käsikirjoitukset ohjaavat käyttäjää omaksumaan hyväksi todettuja turvallisuuskäytäntöjä. On tärkeää välttää käyttäjän kognitiivisten resurssien liiallista kuormittamista teknisesti haasteellisella sisällöllä, jotta hän voi keskittyä koulutuksen asiasisältöön.

Koulutukseen halutaan tuoda mukaan myös kokemuksellinen oppimisen piirteitä käytännön harjoitusten ja havainnollistavien animaatioiden muodossa. [25.] Koulutusten ei tule olla virtuaalisia diaesityksiä, vaan on tärkeää hyödyntää kaikkia virtuaalitodellisuuden tuomia etuja. Esimerkiksi animoitujen ihmishahmojen avulla voidaan havainnollistaa työtehtävän suoritusta, jossa käyttäjällä on mahdollisuus liikkua hahmon vierellä ja keskittyä häntä kiinnostavaan työvaiheeseen. Lopputulos on parempi kuin valmiiksi leikatun kaksiulotteisen videon katsominen ja vastaa hyvin työssäoppimista, jossa ohjaava työntekijä

esittää työtehtävän suorituksen opiskelevalle työntekijälle. Erityisen mielenkiintoisia ovat käytännön tehtävät, joissa käyttäjät pääsevät suorittamaan edellisissä koulutuksissa opittuja asioita käytännössä. Virtuaalinen ympäristö tarjoaa käyttäjälle turvallisen tavan harjoitella myös mahdollisesti vaarallisten työkalujen, kuten esimerkiksi kääntöpöytäsahan tai sorvin käyttöä.

7.2 Asiakkaat ja käyttäjien tuomat haasteet

Koulutussovelluksen käyttäjäkohderyhmä on laaja, käytännössä kaikki työikäiset henkilöt, vaikka nykyinen toteutus on rajattu lähinnä rakennusalan ammattiryhmiin. Työterveyslaitos pyrkii tuomaan tuotteen koko Suomen markkinoille ja tarjoamaan sitä kaikille Suomessa toimiville rakennusalan yrityksille. Tiivis yhteistyö asiakasyritysten kanssa mahdollistaa laadukkaiden mukautettujen virtuaalikoulutussisältöjen luomisen asiakkaiden omiin tarpeisiin. Projektin edessä keskitytään kuitenkin ensin luomaan valmiita koulutuspaketteja niiden koulutusmateriaalien pohjalta, mitä Työterveyslaitoksella jo on. Näiden ensimmäisten koulutusten kohderyhmänä ovat työntekijät, jotka liikkuvat ja tekevät fyysistä työtä rakennustyömailla.

Laaja kohderyhmä itsessään aiheuttaa jo merkittävän haasteen käyttöliittymän kehitykselle, mutta erityisen tärkeää on pyrkiä tuottamaan hyvä käyttäjäkokemus. Käyttäjä ei tule käyttämään sovellusta, koska se on mielenkiintoinen, visuaalisesti näyttävä tai teknisestä vaikuttava, vaan koska hän haluaa oppia uutta. Sovelluksen tulisi toimia näkymättömänä ja intuitiivisena opetusvälineenä, jolloin käyttäjä keskittyy itse oppimiseen ja opetusmateriaaliin sovelluksen sijaan. Virtuaalitodellisuuden lisääminen tähän yhtälöön moninkertaistaa haasteen, sillä käyttäjän tulee pukea päälleen visiiri, joka peittää hänen näkökentän; puristaa hieman päätä ja lisää painoa kasvoille. Teknologian fyysinen käyttökokemus on paljon enemmän kietoutunut teknologiaan kuin tietokoneen edessä tuolilla istuminen, mikä tekee teknologisen välikäden unohtamisesta koulutuskokemuksen aikana haastavampaa.

7.3 Kehitetyt virtuaaliympäristöt ja koulutukset

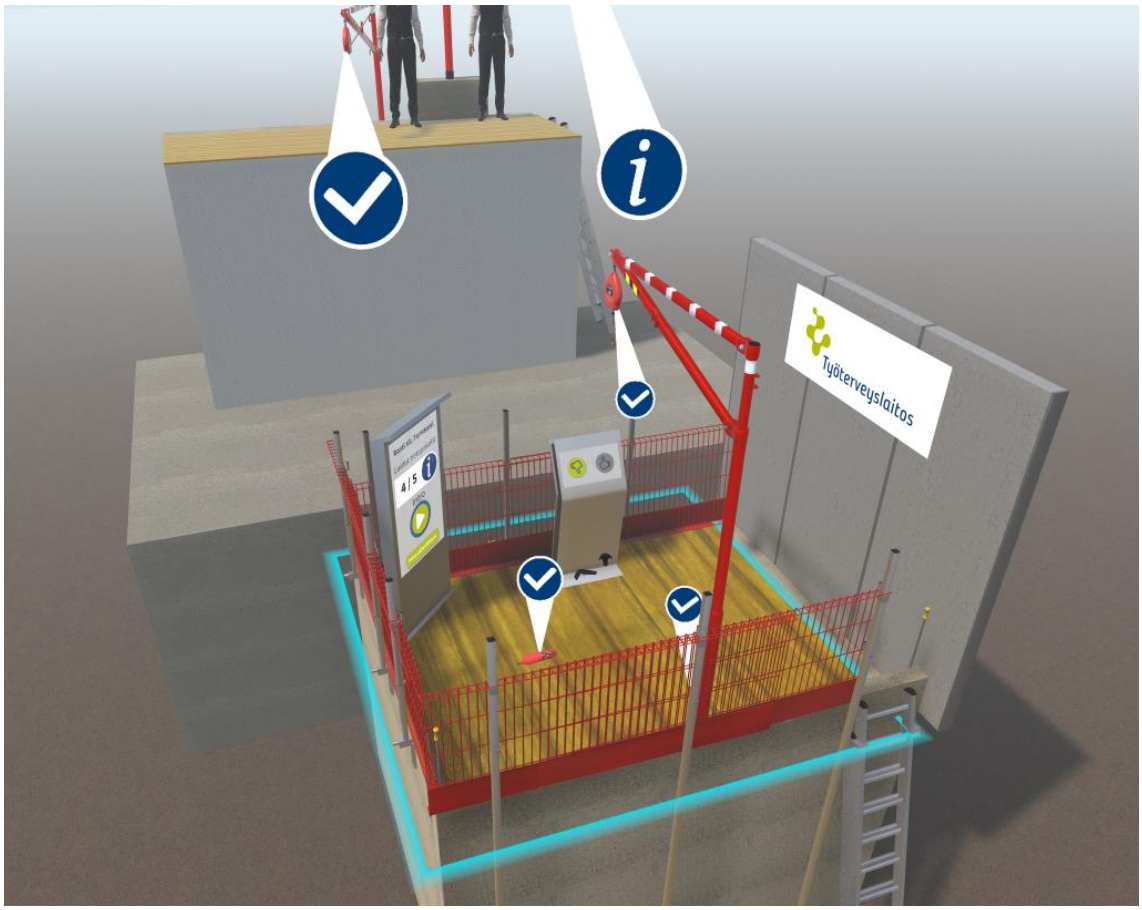
Vaikka projektissa ei vielä luoda lopullisia koulutussisältöjä, teknisten ratkaisujen kokeilemista ja käyttöliittymäkomponenttien työstämistä varten oli tarpeen toteuttaa muutama

koulutusrasti oikeankaltaisella sisällöllä. Turvapuistokoulutusten pohjalta kehitettiin kaksi koulutusrastia. Näiden koulutusrastien sisältö ja ympäristön visuaalinen ilme eivät vastaa lopullisen tuotteen laatua niiden nykyisessä tilassa, sillä rastien on tarkoitus toimia vain havainnollistavina esimerkkeinä tulevia koulutuksia varten. Myös kuvissa esiintyvät ihmishahmot vaihdetaan lopullisessa versiossa vastaamaan paremmin kohdeammattiryhmän henkilöitä. Lisäksi päätettiin tutkia lyhyesti lisätyn virtuaalisuuden (luku 3.3) vaikutusta käyttäjän immersioon luomalla yksi todentuntuinen ympäristö, jolla ei ole koulutusarvoa. Luvun kuvat esittävät koulutusrastien tilaa toukokuussa 2018.

7.3.1 Turvaorsi-koulutusrasti

Vuoden 2017 kesällä kehitettiin ensimmäinen koulutusrasti, jossa opetetaan putoamissuojauksen perusteita. Koulutuskohteina on turvaorren (kuvan 17 etualan punainen tolppa) oikea asennuskorkeus sekä vaijeritarraimen toimintaperiaate putoamistilanteessa. Koulutus valittiin sen takia, että se oli helpoiten kopioitava koulutusrasti Espoon Rudus-turvapuistosta [26]: koulutus oli lyhyt, sisälsi pelkkää teoriaa ja itse koulutusalue oli hyvin lähellä laboratorion virtuaalitodellisuusalueen kokoa. Tarkoituksena oli myöhemmin vertailla virtuaalista ja todellista koulutusrastia käyttäjätesteillä, jotka toteutettiin syksyllä 2017. Käyttäjätesteistä kerrotaan lisää luvussa 7.7.

Virtuaalisen koulutusrastin ympäristö mukailee Rudus-turvapuiston saman koulutusrastin aluetta, mutta turvapuistokoulutuksen opetustapaa, jossa kouluttaja selostaa sisällön käyttäen alueen varusteita esimerkkeinä, ei kopioitu. Teoriasisältö paloiteltiin viiteen lyhyempään tietopakettiin ja ne jaettiin virtuaaliympäristössä eri kohteisiin sisällön perusteella: vaijeritarraimesta kertova teoria avataan käyttämällä vaijeritarrainta, väärin asennetusta turvaorresta kerrotaan käyttämällä liian matalle asennettua turvaortta jne. Kuvassa 17 näkyy neljä luettua (✓-symboli) tietopakettia ja yksi lukematon (i-symboli) tietopaketti. Tietopaketit ovat koulutuksen aluksi piilossa, ja ne tulevat näkyviin käyttäjälle yksitellen sen jälkeen, kun edellinen tietopaketti on luettu. Uusi tietopaketti toistaa ilmestyessään tilaäänimerkin, joka kiinnittää käyttäjän huomion. Tällä ohjataan käyttäjää käymään koulutuksen sisältö läpi loogisessa järjestyksessä ja vältetään hämmennystä ja kognitiivista kuormitusta, joka voisi aiheutua äkillisesti siirtymisestä yksinkertaisesta valikkoympäristöstä monimutkaiseen teoriakoulutusympäristöön.



Kuva 17. Kuvakaappaus putoamissuojauskoulutusrastista. Käyttäjälle saavutettava alue näkyy turkoosina suorakulmiona.

Koulutuksen lopuksi, kun kaikki teoriasisältö on käyty läpi, käyttäjä aloittaa putoamisdemonstraation. Demonstraatio esittää animoiduilla ja fysiikkamallinnetuilla ihmishahmoilla, miten putoamistapaturma eroaa, jos turvaorsi on asennettu väärin (kuvan 18 vasen hahmo) tai oikein (kuvan 18 oikea hahmo). Väärin asennetulla turvaorrella henkilö putoaa liian pitkälle eikä pääse itse takaisin ylös.

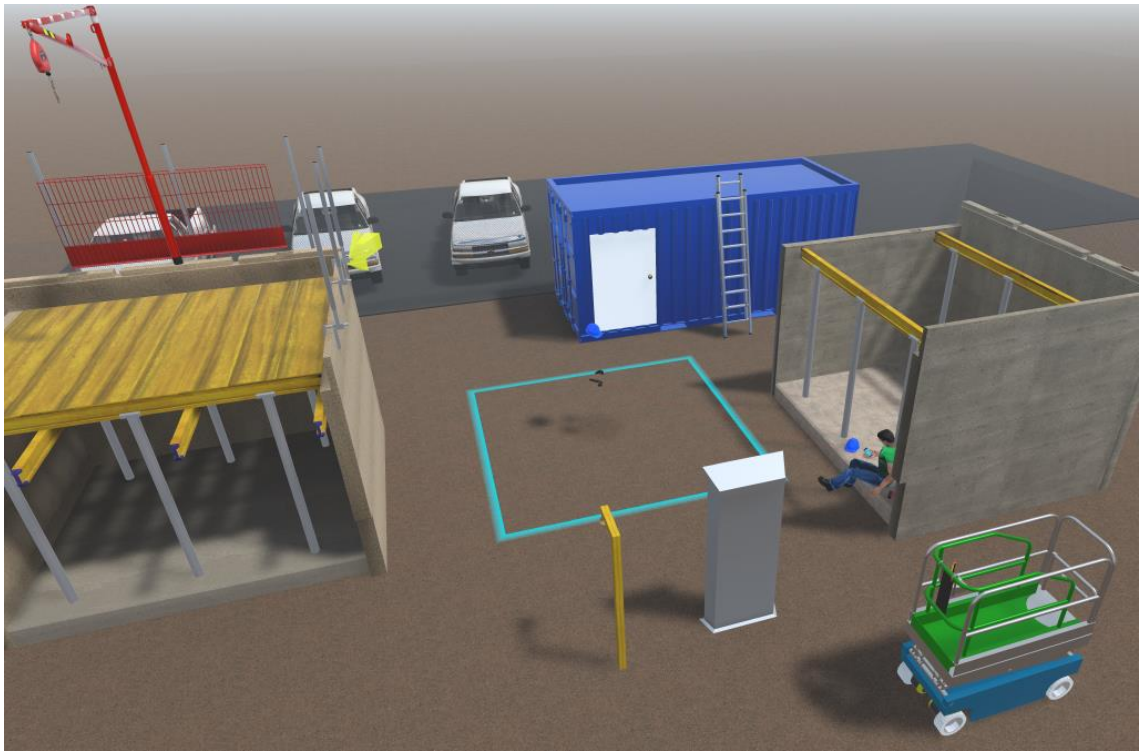
Kuva 18. Kuvakaappaus putoamissuojauskoulutusrastin putoamisdemonstraatiosta.

7.3.2 Korotettu työskentely -koulutusrasti

Toinen koulutusrasteista on interaktiivinen, testin kaltainen koulutus, joka olisi tarkoitus suorittaa koulutusmoduulin (joita ei ole vielä kehitetty) lopuksi. Koulutuksen sisältönä on turvallinen korotetulla alustalla työskentely sekä yleinen rakennustyömaaturvallisuus. Rastin tehtävänä on auttaa työtoveria kiinnittämään korkealla olevan putoamissuojausaidan kiinnike järjestämällä työalue sellaiseen tilaan, että työtehtävän voi suorittaa turvallisesti alhaalta käsin.

Koulutus ei sisällä teoriaopintoja vaan pelkästään tehtävänannon, jonka jälkeen käyttäjän tulee itse pohtia, mitä hänen tulee tehdä, missä järjestyksessä ja millä tavalla tehtävä tulee suorittaa ympäristössä olevien esineiden avulla käyttäen apunaan moduulin aikaisempia teoriaopintoja. Kuvan 19 keskellä on näkymätön käyttäjä virtuaalinen kypärä päässä ja liikeohjaimet kädessä. Käyttäjän ympärillä oleva turkoosi suorakulmio vastaa todellisen virtuaalitodellisuusalueen rajoja, joiden yli käyttäjän ei tule liikkua. Kuvasta kui-

tenkin näkyy, että koulutusrastin alue, jossa käyttäjän tulee liikkua, on merkittävästi todellista virtuaalitodellisuusaluetta suurempi. Ratkaisu tähän ongelmaan on esitetty luvussa 7.5.3.



Kuva 19. Kuvakaappaus korotetun työskentelyn koulutusrastin alueesta. Käyttäjälle saavutettava alue näkyy turkoosina suorakulmiona.

Ympäristössä on muutama kohde, joita käyttäjä voi käyttää edistyäkseen koulutuksessa. Kohteita ei ole erikseen merkitty, vaan käyttäjän tulee itse huomata rastin turvallisuuspoikkeamat (esimerkiksi työtoverilla ei ole kypärää päässään) ja muut tarvittavat toiminnot kuten henkilönostimen siirtäminen. Käytettävien kohteiden käytöstä kerrotaan lisää luvussa 7.5.4.

Koulutuksessa on kolme vaihtoehtoista lopetustilaa: työtehtävän turvallinen tai epäturvallinen suoritus henkilönostimen avulla ja epäturvallinen, tapaturmaan johtava suoritus jatkotikkaiden avulla (kuvassa 20). Koulutuksen päätteeksi, mikäli rastin suoritus epäonnistui, käyttäjälle kerrotaan, mitä asioita hänen olisi tullut huomioida koulutuksen aikana.

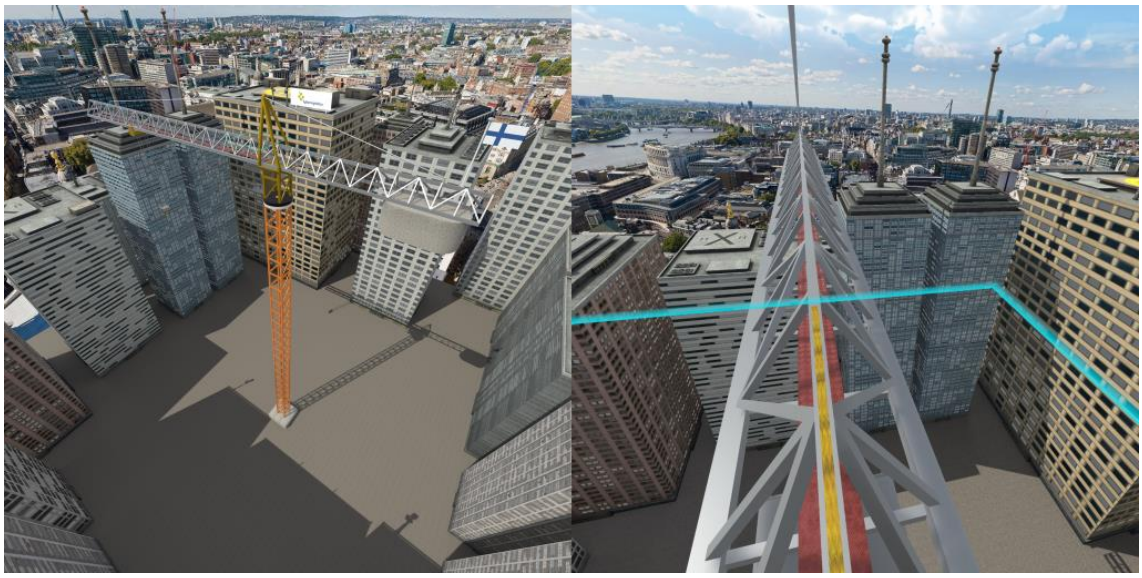
Kuva 20. Kuvakaappaus korotetun työskentelyn koulutusrastin epäonnistuneesta suorituksesta, jossa työtoverille sattuu työtapaturma.

7.3.3 Torninosturi-elämysrasti

Vuoden 2017 syksyllä pohdittiin, millä tavalla lisättyä virtuaalisuutta voitaisiin ottaa mukaan koulutuksiin, millä tavalla se vaikuttaa käyttäjien läsnäolon tunteeseen sekä olisiko siitä mitään hyötyä työturvallisuuskoulutuksissa. Immersion parantamiseksi ympäristössä päätettiin panostaa visuaaliseen laatuun enemmän kuin aikaisemmissa ympäristöissä, joten päädyttiin luomaan elämysrasti, jolla ei ole koulutusarvoa. Kehitetyssä kuvan 21 kaupunkiympäristössä rakennustyömaata ympäröi korkeat kolmiulotteiset rakennukset, joiden keskellä olevan torninosturin puomin päällä käyttäjä kävelee. Kauempana oleva kaupunkinäkymä on kaksiulotteinen panoraamavalokuva. Näkymä käyttäjän näkökulmasta puomin päältä on kuvan 21 oikealla puolella. Auditiivista immersiota kehitettiin luomalla useasta eri äänileikkeestä koostuva, käyttäjää ympäröivä kolmiulotteinen

äänimaisema, joka kuulostaa vilkkaalta kaupungilta, jota peittää voimakas tuulen kohina. Ympäristöä uusiokäytetään tulevaisuudessa kehitettävässä, torninosturin ohjausta kouluttavassa rastissa.

Lisätyn virtuaalisuuden elementtejä tuotiin virtuaaliseen ympäristöön lisäämällä todellinen ohut puulankku virtuaalitodellisuusalueelle, alentamalla huoneen lämpötilaa ilmastointilaitteilla sekä puhaltamalla ilmaa käyttäjän suuntaan tehokkaalla tuulettimella. Käyttäjää opastetaan ympäristössä kävelemään virtuaalisen lankun päällä (näkyy kuvan 21 alalaidassa), jolla on samat mitat kuin todellisella lankulla. Näiden elementtien lisäämisellä elämysrasti huijaa käyttäjän näkö-, kuulo-, lämpötila- ja tasapainoaistia luoden uskottavan vaikutelman siitä, että henkilö todella on nosturin puomin päällä. Käyttäjätestien palautteiden (katso luku 7.7) perusteella todellisen lankun tunteminen jalkojen alla on lisännyt immersiota merkittävästi, ja äänimaailma on ollut vaikuttava, mutta käyttäjät eivät ole antaneet lainkaan palautetta tuulettimesta tai lämpötilasta.



Kuva 21. Kuvakaappaus torninosturielämysrastista virtuaalitodellisuusalueen ulkopuolelta (vasen puoli) sekä käyttäjän näkökulmasta (oikea puoli).

7.4 Liikeohjain syöttölaitteena

Erityisesti tietokonemaailmassa hiiren käyttö osoittimena on vakiintunut ja ohjelmistokehittäjän on helppo olettaa käyttäjän osaavan käyttää osoitinta erinomaisesti. Projektissa käytetään kuitenkin HTC Vive -laitteistoa, jonka mukana tulee kaksi liikeohjainta (ku-

vassa 22), joissa on merkittävästi enemmän mahdollisuuksia käyttöliittymän kanssa vuorovaikutukseen verrattuna tavanomaiseen kaksinappiseen rullalla varustettuun hiireen. Käyttäjäkokemusten perusteella ohjaimen käyttö on ollut haastavaa monille käyttäjille myös sen uutuusarvon takia, sillä monet eivät ole nähneet kyseistä ohjainta ennen. Ohjaimen käyttöä helpottaa kuitenkin erittäin tarkka 3D-malli liikeohjaimesta virtuaalitodellisuudessa, jonka sijainti vastaa tarkasti todellisen liikeohjaimen sijaintia. Vaikka käyttäjä ei näe omaa kättään virtuaalimaailmassa, hän pystyy silti asentoaistin avulla siirtämään sormensa oikean napin kohdalle katsomalla liikeohjaimen 3D-mallia virtuaalitodellisuudessa.

Kuva 22. HTC Viven liikeohjain. [20.]

Liitteen 5 (Tietokonehiiren, sekä Viven ja Gear VR:n liikeohjainten toimintojen vertailu) taulukossa vertaillaan tavanomaisen tietokonehiiren ja HTC Viven liikeohjaimen toimintoja. Vertailun vuoksi mukana on myös älypuhelinpohjaisen Gear VR -laitteen liikeohjain. Liikeohjain on selvästikin hiirtä kehittyneempi osoitintyökalu, joka antaa kehittäjälle enemmän vapauksia luoda juuri haluamansa kaltaisia vuorovaikutuksia, mutta on myös tämän vuoksi hankalampi uusille käyttäjille. Ohjain onkin kehitetty videopelit mielessä pitäen, ja toimintojen määrä on hyvin verrattavissa pelikonsolien mukana tuleviin ohjaimiin. Vaikka liikeohjain tarjoaa suuren määrän toimintoja, on tarkoituksenmukaista käyttää mahdollisimman pientä osaa niistä ja luoda yksinkertainen, intuitiivinen syöttöjärjestelmä liikeohjaimelle, jotta vähennetään käyttäjän kognitiivista kuormaa.

7.5 Liikeohjaimen toiminnallisuudet sovelluksessa

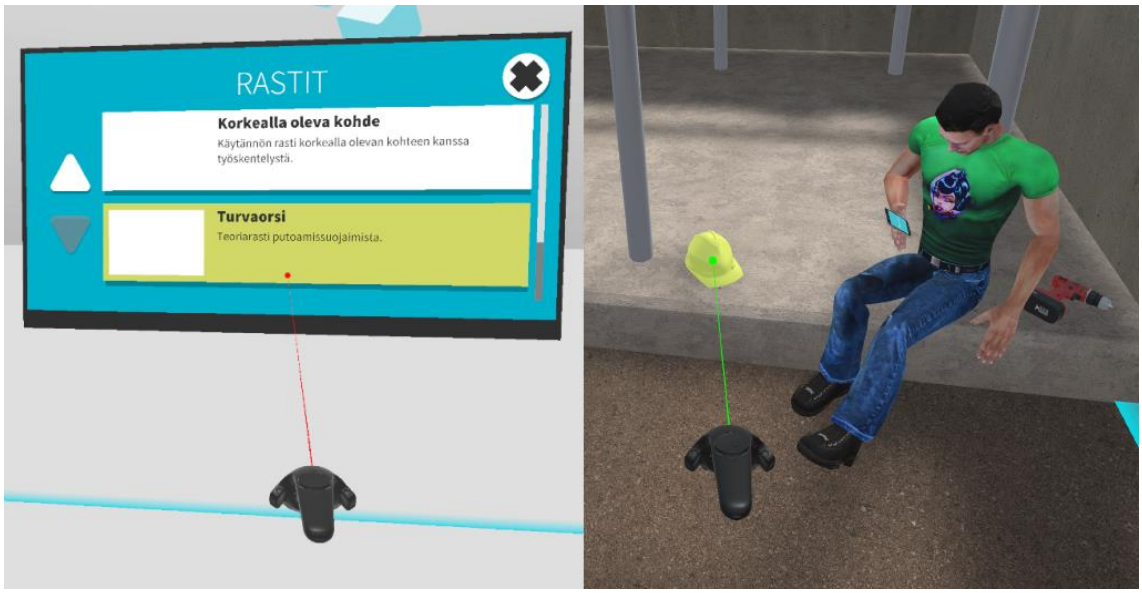
Kehitetyissä koulutusrasteissa on neljä erilaista toiminnallisuutta liikeohjaimelle:

- laserosoitin
- esineisiin tarttuminen
- nopeutettu liikkuminen
- diegeettisten käyttöliittymäelementtien käyttö.

7.5.1 Laserosoitin

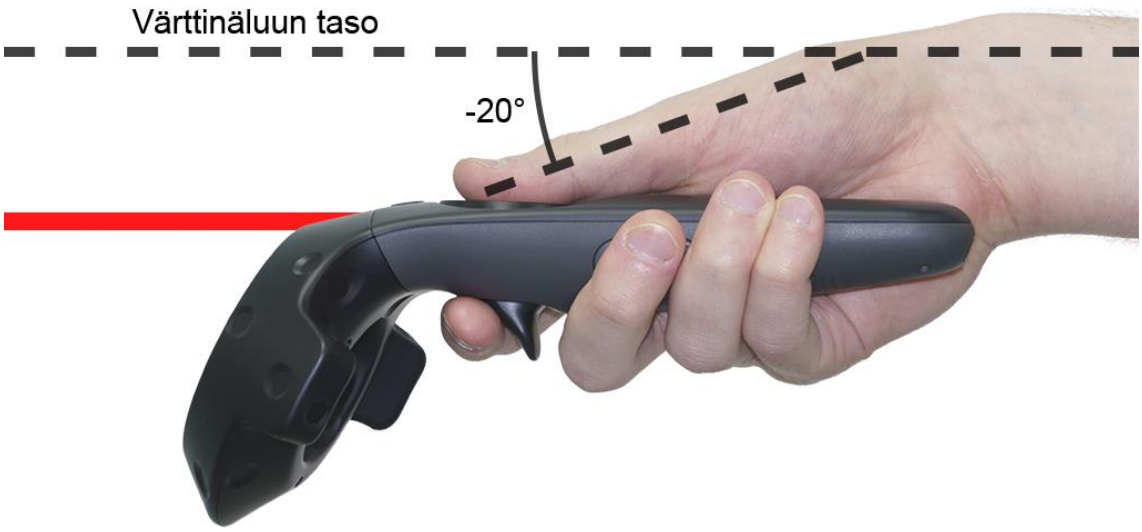
Laserosoitin on tärkein liikeohjaimen toiminnallisuus, sillä se on virtuaaliympäristöissä ensisijainen tapa, jolla käyttäjä vuorovaikuttaa kohteisiin. Se mahdollistaa kaksiulotteisten käyttöliittymäelementtien käytön kolmiulotteisessa maailmassa ilman, että käyttäjän tulee koskettaa elementtejä ohjaimella. Laserosoitin vastaa tietokonehiiren osoitinta tavanomaisessa graafisessa käyttöliittymässä.

Mikäli koulutuksessa tarvitaan laserosoitinta, on se aina päällä toisessa liikeohjaimessa. Visuaalisesti tämä näyttää liikeohjaimesta lähtevältä ohuelta punaiselta putkelta. Kun laserilla tähtää kohdetta, jota voi käyttää laserosoittimella, kohde reagoi vaihtamalla väriä vihreäksi. Kuvassa 23 on esimerkki tästä, jossa osoitetaan kaksiulotteista käyttöliittymäelementtiä kuvan vasemmalla puolella ja kolmiulotteista kohdetta oikealla puolella. Kolmiulotteisten käyttöliittymäelementtien kanssa myös laserin väri muuttuu vihreäksi. Molemmat muutokset ovat tarpeellisia, sillä jos vain laserin väri vaihtuu, käyttäjä ei välttämättä heti huomaa, mitä hän osoittaa. Jos pelkästään osoitettava kohde muuttuu, käyttäjä ei välttämättä huomaa, että hän edes osoittaa käytettävää kohdetta. Vastaava ominaisuus puuttuu vielä kaksiulotteisten käyttöliittymäkomponenttien käytöstä. Laserosoitinta jatkokehitetään myöhemmin lisäämällä visuaalisten ärsykkeiden lisäksi myös hiljainen ääniefekti ja haptinen palaute, mikä auttaa sovelluksen käyttöä henkilöillä, joilla on poikkeuksellinen värinäkö. Laserosoittimella kohteiden käyttö tehdään aina painamalla etusormen alla oleva liipaisin pohjaan. Toiminto valittiin, koska tarkoituksena on simuloida tietokonehiiren vasemman napin painallusta.



Kuva 23. Laserosoittimen käyttö kaksiulotteisen käyttöliittymäelementin (vasen) ja kolmiulotteisen käyttöliittymäelementin (oikea) kanssa.

Käytetty laserosoitinkomponentti on muokattu VRTK-kirjaston mukana tulevasta komponentista. Tämä komponentti luo laservalon oletuksena liikeohjaimesta suoraan eteenpäin (tai ylöspäin riippuen siitä, millaiseksi ohjaimen oletusasennon mieltää), jolloin laser on tasossa liikeohjaimen varren kanssa (kuva 24) ja sen käyttö vastaa melko hyvin kynänmallisen laserosoittimen käyttöä. Sillä tulee tähdätä kohteita kuin niitä osoittaisi sauvalla tai television kaukosäätimellä. Tätä laserosoitinta kutsutaan myöhemmin sauvalaseriksi.



Kuva 24. Liikeohjaimen ote, kun käytetään sauvalaseria. Laservalo lisätty kuvaan jälkikäsittelynä havainnollistamaan käyttäjän virtuaalitodellisuudessa näkemää laseria.

Joissain HTC Viveä tukevissa virtuaalitodellisuussovelluksissa laserosoittimen valo on kuitenkin tasossa liikeohjaimen pään kanssa (kuva 25), jolloin sen käyttö tuntuu enemmänkin pistoolilta, erityisesti jos liipaisin on käyttönappi. Tämän laserosoittimen toteutustavan on tarkoitus simuloida sormella osoittamista, sillä laservalo osoittaa samaan suuntaan kuin käyttäjän etusormi ojennettuna kuvan 25 mukaisessa liikeohjaimen otteessa. Tätä laserosoitinta kutsutaan myöhemmin sormilaseriksi.

Kuva 25. Liikeohjaimen ote, kun käytetään sormilaseria. Laservalo lisätty kuvaan jälkikäsittelynä havainnollistamaan käyttäjän virtuaalitodellisuudessa näkemää laseria.

Ergonomian kannalta sormilaser on parempi, sillä tavallisessa käytössä sitä ei pysty käyttämään epäergonomisesti. Käyttäjän ranne pysyy melko suorassa ja peukalo osoittaa ylöspäin (kuten kuvassa 25) riippumatta siitä, pitääkö käyttäjä liikeohjainta lähellä kehoaan koukistaen kyynärniveltä (kuva 26) tai käsi suorana edessään ampuma-aseen tavoin. Sormilaserin käyttö molemmilla tavoilla on mieluisaa.

Kuva 26. Sormilaserin käyttö lähellä kehoa. Miellyttävä, mutta epätarkka tapa käyttää laserosoitinta. Laservalo lisätty kuvaan jälkikäsittelynä havainnollistamaan käyttäjän virtuaalitodellisuudessa näkemää laseria.

Sauvalaseria on miellyttävämpi pitää lähellä kehoa kuvan 26 mukaisesti. Jos sitä pidetään kehon edessä käsi suorana kuvassa 27 esitetyllä tavalla, käyttäjän ranne toisinaan taipuu epämiellyttävästi kyynärluun suuntaan, jopa värttinäluun tason alapuolelle kuvan 24 mukaisesti. Pidemmässä käytössä myös peukalon jänteen venyminen voi rasittaa käyttäjän kättä ja haitata käyttökokemusta.

Kuva 27. Sauvalaserin käyttö käsi suorana. Epämiellyttävä, mutta tarkka tapa käyttää laserosoitinta. Laservalo lisätty kuvaan jälkikäsittelynä havainnollistamaan käyttäjän virtuaalitodellisuudessa näkemää laseria.

Kokeilujen perusteella sormilaserin käyttö ei tunnu luonnolliselta, eikä tähtääminen ole yhtä tarkkaa kuin sauvalaserin kanssa. Oletuksena sovelluksessa käytetään kuitenkin sormilaseria, sillä se on käyttäjäystävällisempi ergonomian kannalta. Käyttäjällä on vapaus muuttaa laserosoittimen tyyppiä sovelluksen asetuksista.

7.5.2 Esineisiin tarttuminen

Koulutusrasteissa voi myös olla fysiikkamallinnettuja esineitä, joiden kanssa käyttäjä voi olla vuorovaikutuksessa, esimerkiksi siirtämällä niitä paikasta toiseen. Jos toimintoa käytetään koulutuksessa, jossa on käytössä myös laserosoitin, fysiikkamallinnetut esineet saavat vihreän ääriviivan (kuvan 28 vasen puoli), kun niihin tähdätään laserilla. Ääriviiva ilmestyy myös esineeseen, kun käyttäjä vie liikeohjaimen tarpeeksi lähelle sitä. Samalla liikeohjaimen kolmiulotteinen malli muuttuu avoimeksi kädeksi (kuvan 28 oikea puoli) ja laserosoitin katoaa, mikäli se oli käytössä. Tarkoituksena on viestiä käyttäjälle, ettei kyseistä esinettä voi käyttää laserosoittimella, vaan sille pitää tehdä jotain, mitä voi tehdä

todellisen käden kanssa. Kun käyttäjä puristaa ohjainta (kuin tarttuakseen virtuaaliseen esineeseen), liikeohjaimen sivuilla olevat puristusnapit painuvat pohjaan. Tämän jälkeen ohjaimen malli muuttuu suljetuksi kädeksi (simuloiden käyttäjän omaa kättä, kuvan 28 alaosa) ja esine alkaa seuraamaan ohjaimen sijaintia, ja sen siirtely on mahdollista.

Kuva 28. Kuvakaappaussarja virtuaalitodellisuudesta korotetun työskentelyn rastista, jossa tähdätään jatkotikkaita laserosoittimella (vasen puoli), viedään liikeohjain tikkaiden lähelle (oikea puoli) ja siirretään tikkaita (alaosa).

Tikkaiden käsittely on kuitenkin haastavaa virtuaalitodellisuudessa, sillä ne ovat niin suuret, että niitä on vaikeaa nähdä kokonaan. Myös fyysisen palautteen puute (esineen paino) tekee kappaleen käsittelystä luonnotonta, joten on vaikeaa tietää tarkalleen, mistä kohtaa tikkaita on tartuttu kiinni ja kuinka paljon niitä on nostettu maasta ylös. Monet käyttäjät ovat rastia suorittaessaan yrittäneet asetella tikkaita seinää vasten samalla työntäen ne maan sisään. Unityn fysiikkamoottori korjaa tilanteen työntämällä tikkaita maan sisältä takaisin sen pinnalle suurella voimalla, joka usein johtaa tikkaiden lentämiseen korkealle taivaalle. Tätä ongelmaa ei ole vielä ratkaistu.

7.5.3 Nopeutettu liikkuminen

Korotetun työskentelyn koulutusrastin alue on huomattavasti suurempi kuin laboratorion virtuaalitodellisuusalue, joten tuli kehittää ratkaisu ongelmaan, miten liikkua virtuaaliympäristössä suuremmalla alueella kuin todellinen virtuaalitodellisuusalue sallii. Virtuaalitodellisuudessa liikkumiseen on kehitetty useita eri laitteisto- ja ohjelmistopohjaisia ratkaisuja ja ongelmaa tutkivat niin pelinkehittäjät kuin akateemiset tutkijatkin. Työn kirjoitukseen mennessä esiin ei ole noussut yhtään ratkaisua, joka miellyttäisi kaikkia tai toimisi joka sovelluksessa, mutta teleportaatio ja sen eri muodot ovat erittäin yleisiä virtuaalitodellisuussovelluksissa. Virtuaalitodellisuudessa liikkuminen on erittäin laaja aihe, josta saisi kokonaan oman insinöörityön, joten tässä luvussa keskitytään ainoastaan erittelemään valittua ja muita tutkittuja ratkaisuja.

Virtuaalitodellisuudessa liikkumisen suurin ongelma on virtuaalitodellisuuspahoinvointi. Virtuaalitodellisuuspahoinvoinnilla (vanhemmassa englanninkielisessä kirjallisuudessa käytetään toisinaan cybersickness-termiä, eli kyberpahoinvointi) tarkoitetaan käyttäjän kokemaa epämiellyttävää olotilaa virtuaalitodellisuuden käytön aikana ja sen jälkeen. Olotila on verrattavissa matkapahoinvointiin, ja se on hyvin samankaltainen simulaattoripahoinvoinnin kanssa, joskin oireissa on havaittu eroja virtuaalitodellisuuspahoinvoinnin aiheuttamaan olotilaan. Se luokitellaankin usein erilleen [27]. Oireet ovat yksilöllisiä ja vaihtelevat laajasti hikoilusta ja sekavasta olotilasta huimaukseen ja oksentamiseen. Vaivaa on tutkittu jo yli 20 vuotta [28], mutta ongelmaan ei ole vielä löydetty ratkaisua.

Yksi suosituin teoria virtuaalitodellisuuspahoinvoinnin lähteestä on niin kutsuttu aistillinen ristiriita -teoria (engl. sensory conflict theory). Teoria selittää pahoinvoinnin silmien ja korvan tasapainoelimen aistihavaintojen yhteensopimattomuudella. Ristiriita esiintyy, kun käyttäjän näkymä liikkuu virtuaalisessa ympäristössä, mutta käyttäjä pysyy paikallaan oikeassa todellisuudessa. Koska virtuaalivisiiri peittää käyttäjän koko näkökentän, aivot voivat tulkita näkymän liikkeen todellisena liikkeenä visuaalisten ärsykkeiden pohjalta. Todellisessa liikkeessä tasapainoelimen ärsykkeet normaalisti viestisivät aivoille liikkeen lisäksi myös kehon kierrosta. Virtuaalitodellisuuden näkymän liikkeen aikana nämä ärsykkeet puuttuvat, ja näin aivot saavat ristiriitaisia ärsykkeitä havaintoelimistä, josta pahoinvointi aiheutuu. [29.] Näkymän ja käyttäjän liikkeen ristiriita toimii myös toisin päin. Pahoinvointia ilmenee myös tilanteessa, jossa käyttäjä todellisen liikkeen seuran-

nassa on häiriöitä ja virtuaalitodellisuuden näkymä ei enää seuraakaan käyttäjän todellista liikettä tai siinä on merkittävää viivettä. Toisin sanoen käyttäjä liikkuu ja saa tasapainoelimistä ärsykkeitä liikkeestä, mutta silmien havaitsema näkymä pysyy paikallaan.

Liikkumisen ongelmaan etsittiin ratkaisua ohjelmistotasolla: useita eri liikkumistoteutuksia tutkittiin kokeilemalla niitä eri virtuaalitodellisuuspeleissä, toteuttamalla ne VRTK-kirjaston tarjoamien valmiiden komponenttien avulla ja katsomalla videoita niiden käytöstä. Tavoitteena oli löytää toteutus, joka olisi mahdollisimman helppokäyttöinen, miellyttävä käyttää (toisin sanoen aiheuttaa mahdollisimman vähän virtuaalitodellisuuspahoinvointia), mutta myös helppo toteuttaa. Taulukossa 3 listataan kaikki tutkitut liikkumistavat sekä niistä usein käytetyt englanninkieliset termit. Eri toteutustavat on eritelty kolmeen kategoriaan: sormia ja käsiä käyttävät, kävelyä hyödyntävät, sekä teleportaatiota käyttävät liikkumistavat.

Taulukko 3. Taulukko tutkituista virtuaalitodellisuuden liikkumistavoista.

Kategoria	**Liikkumistapa**	**Englanninkieliset termit**
Teleportaatio	teleportaatio, samanaikainen	teleportation
	teleportaatio, mustan ruudun kautta	blink teleportation
	teleportaatio, liike kohteeseen	dash teleport
	välimatkateleportaatio	dash step, blink step
Kävely	maailman pysäytys reunoilla	WalkAbout system
	muuttuva maailma	dynamic world, environmental redirection
	nopeutettu tai vahvistettu liike	gained walking, walking gain
	paikallaan kävely	walking in place
	uudelleenohjattu kävely	redirected walking, reoriented walking
Kädet / sormet	liike kosketuslevyn tai ohjaintatin avulla	direct movement, smooth locomotion
	ohjainten heiluttelu edestakaisin/ylös-alas	paddling, swinging

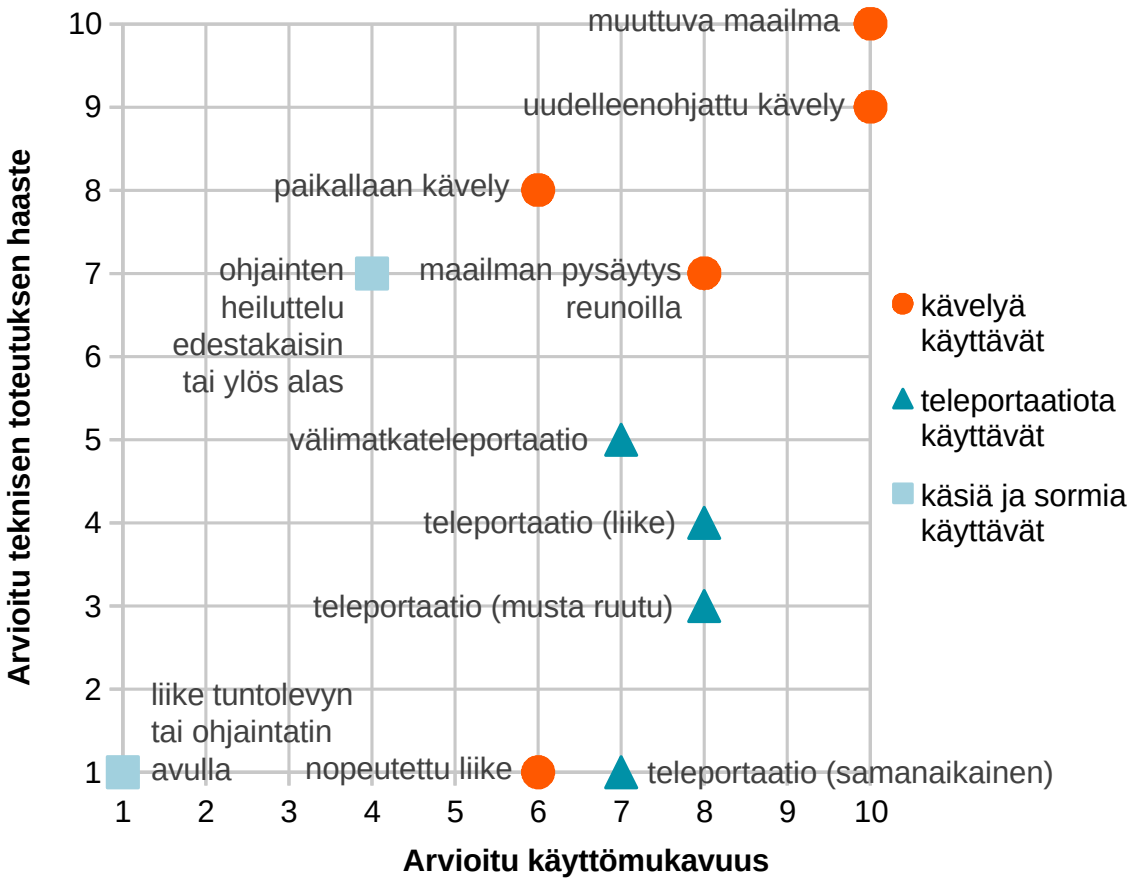
Ohjaintatin (tai HTC Viven tapauksessa kosketuslevyn) avulla liikkuminen on alkeellisin liikkumistapa, joka on hieman toteutuksesta riippuen lähes identtinen monissa videope-

leissä toteutetun hahmojen liikuttelun kanssa. Tämä liikkumistapa yksinkertaisesti liikuttaa käyttäjän kameraa (usein) vakionopeudella siihen suuntaan, mihin ohjaintatti osoittaa. Kamera siis liukuu sulavasti ympäristön läpi. Internetin keskustelupalstoilla tätä liikkumistapaa pidetään huonona, sillä se aiheuttaa eniten virtuaalitodellisuuspahoinvointia. Virallista tutkimusta liikkumistapojen vertailusta ei ole kuitenkaan tehty, mikä vahvistaisi yleisen mielipiteen.

Teleportaatioliikkumiseen on useita hieman erilaisia toteutuksia myös taulukon 3 lisäksi, mutta perusidea on sama: käyttäjä osoittaa liikeohjaimen osoittimella, joka on usein laserosoitin, maahan kohtaan johon hän haluaa siirtyä ja painaa nappia siirtyäkseen siihen. Taulukon 3 teleportaatiotapojen siirtymisen toteutukset ovat seuraavat.

- Samanaikainen, jolloin käyttäjä ilmestyy heti osoitettuun kohtaan. Tarkoituksena on poistaa kaikki mahdollinen näennäinen liike, millä pyritään poistamaan virtuaalitodellisuuspahoinvointi kokonaan.
- Muutamia satoja millisekunteja kestävä siirtyminen, jonka aikana käyttäjälle näytetään musta ruutu, joka simuloi silmien räpäytystä. Yleinen mielipide on, että tämä vähentää pahoinvointia.
- Käyttäjä liukuu kohteeseen joko nopeasti alle sekunnissa, tai siirtyminen voi kestää jopa useita sekunteja. Liukumisen toteutus pyrkii parantamaan ohjaintatin avulla liikkumista tekemällä liikkeen niin nopeasti, ettei käyttäjän aistit ehdi kunnolla reagoimaan siihen. Hitaassa liukumisessa tarkoitus on hidastaa liikettä niin paljon, että se olisi mieluisaa käyttäjälle ja tuntuisi melkein siltä, että käyttäjä liikkuu tasaisella vauhdilla raiteilla.
- Välimatkateleportaatio eroaa muista toteutuksista rajoittamalla matkaa, jonka käyttäjä siirtyy. Matka voi olla vakio (yleensä lyhyt, alle metrin mittainen) tai vaihteleva riippuen esimerkiksi käyttäjän sormen sijainnista kosketuslevyllä. Tämä toteutus lähenee ohjaintatin avulla liikkumista, sillä käyttäjä ei valitse tarkkaa kohtaa mihin hän siirtyy kuten muissa teleportaatiototeutuksissa, vaan pikemminkin suunnan, johon hän haluaa kulkea.

Eri liikkumistapojen tutkimisen jälkeen ne pisteytettiin kahdessa kategoriassa: teknisen toteutuksen haaste (asteikolla 1–10, epämukavasta mukavaan) ja käyttömukavuus (asteikolla 1–10, helposta haastavaan). Kuva 29 esittää edellä mainittujen liikkumistapojen pisteytyksen kahdessa ulottuvuudessa, jossa x-akselilla on käyttömukavuus ja y-akselilla teknisen toteutuksen haaste. Käytetty VRTK-kirjasto tarjoaa valmiin toteutuksen kaikkiin 1-asteelle arvioituihin liikkumistapoihin, joka on syynä niiden alhaiseen pisteytykseen.



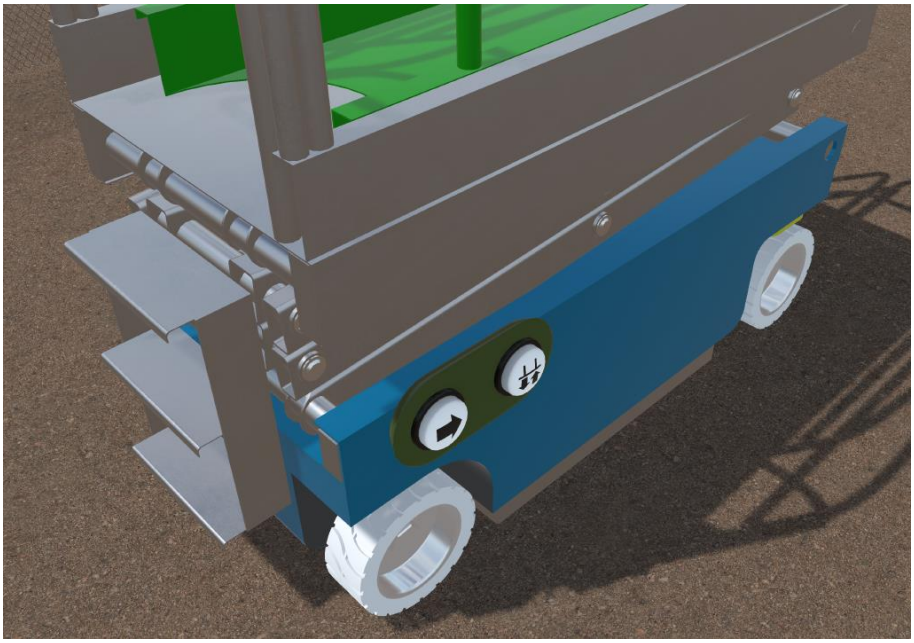
Kuva 29. Eri virtuaalitodellisuuden liikkumistapojen pisteytys asteikolla 1–10 käyttömukavuus-tekninen haaste -koordinaatistossa.

Liikkumisen haluttiin olevan mahdollisimman luonnollista ja todenmukaista. Niiden helpon toteutuksen takia vain teleportaatiota ja nopeutettua liikettä harkittiin sovelluksen ensisijaiseksi liikkumistavaksi. Sovelluksessa päädyttiin valitsemaan nopeutettu liike, vaikka teleportaatioliikkuminen todettiin käyttäjälle mieluisammaksi, koska teleportaation yksinkertaisessa toteutuksessa käyttäjän visuaalinen jatkuvuus rikkoutuu teleportaation aikana, eikä se ole mahdollista oikeassa todellisuudessa. Käyttäjäpalautteena on kuitenkin tullut kritiikkiä liikkumisen hankaluudesta korotetun työskentelyn koulutusrastissa (luku 7.3.2), joten tulevaisuudessa eri liikkumistapoja tutkitaan lisää ja nopeutettu liike korvataan toisella tavalla.

7.5.4 Diegeettisten käyttöliittymäelementtien käyttö

Molemmissa koulutusrasteissa on erilaisia kaksi- ja kolmiulotteisia kohteita, joita käyttämällä ympäristössä tapahtuu jotain, mitä käyttäjä haluaa tehdä. Nämä diegeettiset käyt-

töliittymäelementit on muotoilultaan suunniteltu osaksi ympäristöä, ja ne sijaitsevat ympäristön kolmiulotteisessa avaruudessa eivätkä ole kiinni esimerkiksi kameran tasossa kuten perinteiset käyttöliittymät. Tällaiset käyttöliittymäelementit sallivat käyttäjän tehdä sellaisen toiminnon, joka ei välttämättä olisi mahdollista, mikäli kyseinen ympäristö olisi todellinen. Kuvassa 30 on esimerkki saksinostimen kyljessä olevista napeista, joita painamalla käyttäjä voi suorittaa saksinostimen kaksi mahdollista toiminnallisuutta. Nappeja tulee painaa työntämällä niitä sisään liikeohjaimella, jolloin toiminto tapahtuu. Vasemmanpuoleinen nappi laukaisee nostimen liikkumistoiminnon, jonka jälkeen nostin liikkuu automaattisesti oikeaan kohtaan koulutusrastin ympäristössä, jotta koulutusta voi jatkaa. Tällainen monimutkaisen toiminnon yksinkertaistaminen yhdeksi napiksi oli tarpeen, sillä saksinostimen ajaminen ei kuulu rastissa koulutettavaan turvallisuussisältöön.



Kuva 30. Kuvakaappaus virtuaalisesta saksinostimesta, jonka sivuun on mallinnettu kaksi suurta nappia.

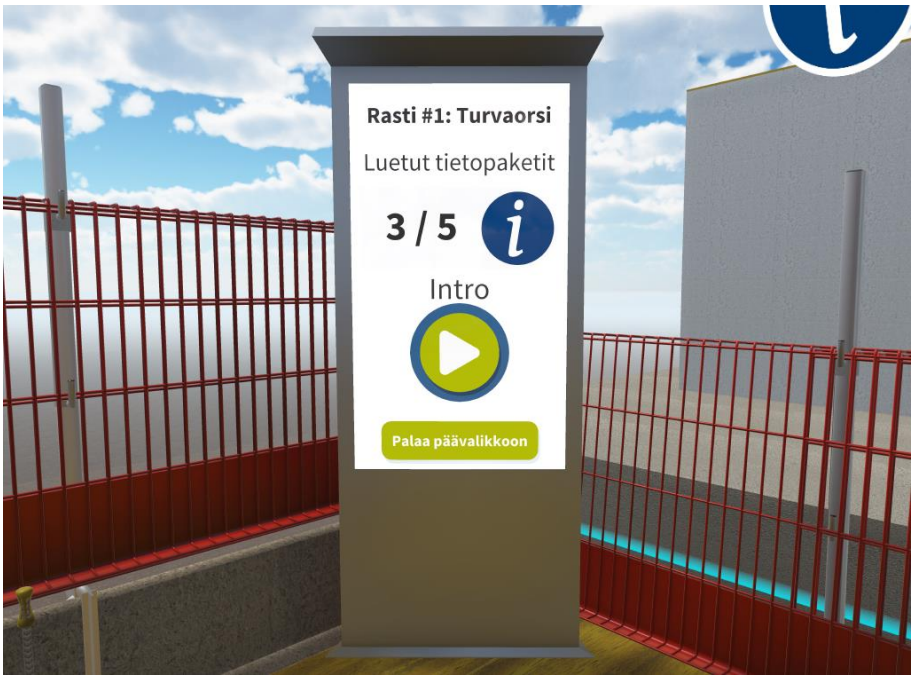
Käyttäjäpalautteesta on kuitenkin ilmennyt, että edellä mainitun esimerkin nappien toteutus ei ole luonteva. Kaikki muut käytettävät elementit sovelluksessa toimivat laserosoittimen avulla, joten käyttäjän kannalta selkeämpää olisi rajoittaa erilaisia kohteiden käyttötapoja, ellei koulutussisältö vaadi monimutkaisempaa toimintoa. Tämä asia on todettu myös Virtuaaliturvapuiston arviointitutkimuksessa [30, s. 17].

7.6 Käyttöliittymä

Sovelluksessa pyritään käyttämään mahdollisimman vähän tavanomaisia kameraan kiinnitettyjä kaksiulotteisia käyttöliittymäelementtejä. Tarkoituksena on parantaa kognitiivisen läsnäolon tunnetta siirtämällä käyttöliittymä pois kameran kaksiulotteisesta tasosta osaksi kolmiulotteista ympäristöä. Kuvassa 31 näkyvä koulutusrastin päävalikko muistuttaa tavanomaista kaksiulotteista valikkoa, mutta se on itse asiassa yli kahdeksan metrin päässä käyttäjästä simuloiden huikeaa 350 tuuman televisiota. Television valikkoa käytetään laserosoittimen avulla.

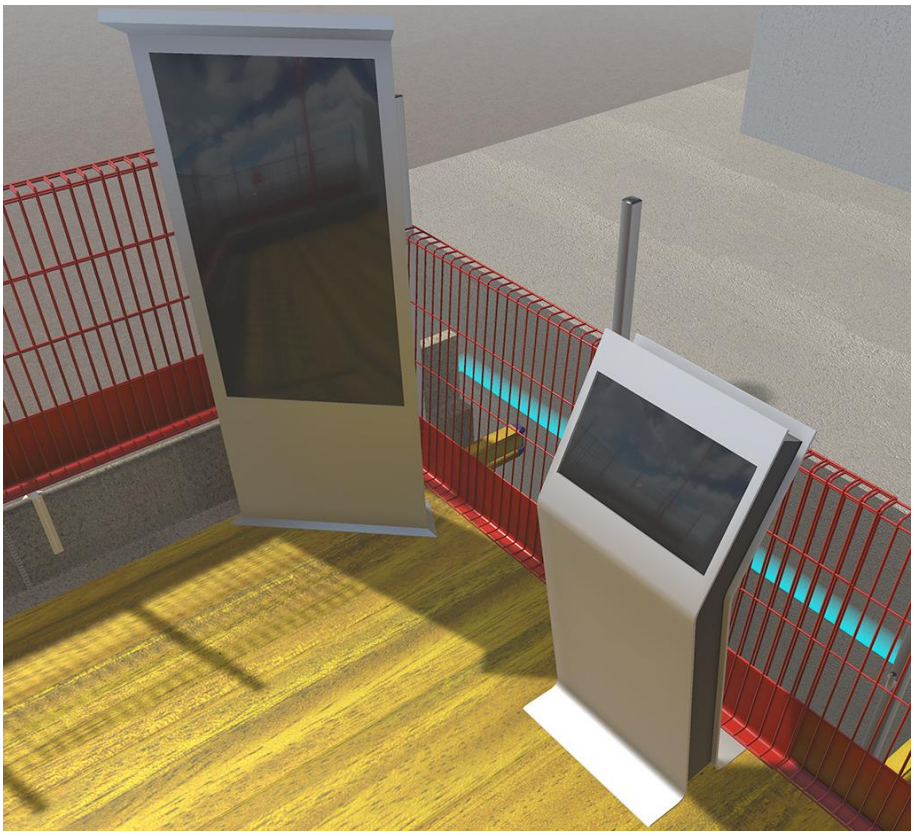
Kuva 31. Sovelluksen päävalikko.

Diegeettisen käyttöliittymän periaatetta pyritään toteuttamaan myös kaikkialla muualla sovelluksessa. Koulutusrastien valikot on toteutettu rastien ympäristöissä olevien infonäyttöjen avulla. Kuvassa 32 on turvaorsirastin (luku 7.3.1) infonäyttö, joka kertoo käyttäjälle, kuinka monta tietopakettia hän on lukenut, kuinka monta niitä rastissa on sekä sisältää myös lyhyen äänitteen, joka kertoo käyttäjälle, mitä rastissa tehdään.



Kuva 32. Turvaorsirastin infonäyttö koulutuksen puolivälissä.

Infonäyttöjen visuaalinen ilme on suunniteltu luomaan suuri kontrasti koulutusrastien ympäristöjen kanssa. Kuten kuvista 32 ja 33 ilmenee, ne ovat väritykseltään vaaleita, niissä on suuria heijastavia metallipintoja, niiden pinnat ovat puhtaita ja ne ovat muotoilultaan moderneja, pelkistettyjä ja suurikokoisia. Tämä kuitenkin rikkoo sovelluksen diegeettisen käyttöliittymän periaatetta hieman, sillä kohderyhmän käyttäjät ymmärtävät heti, että tällaisia laitteita ei ole todellisen maailman työmailla. Periaatetta on kuitenkin rikottu tarkoituksella, jotta parannetaan sovelluksen käytettävyyttä. Koska laitteet luovat kontrastia muuhun ympäristöön, käyttäjät ymmärtävät (ehkä alitajuisesti), että laitteet eivät kuulu itse koulutussisältöön, vaan ovat osa virtuaalitodellisuussovellusta. Sovelluksessa on vain kaksi eri infonäyttömallia (esitetty kuvassa 33), joita käytetään koulutusrasteissa, joten käyttäjä oppii nopeasti laitteiden tarkoituksen.



Kuva 33. Sovelluksen kaksi eri infonäyttöä: vasemmalla monoliitti ja oikealla pöytämalli.

7.7 Virtuaaliturvapuiston arviointitutkimus

Josefin Reuterin Työterveyslaitokselle elokuussa 2017 tekemässä Virtuaaliturvapuiston arviointitutkimuksessa vertailtiin Rudus-turvapuistokoulutuksen ja tässä työssä esitellyn sovelluksen koulutusten käyttäjäkokemuksia. [30.] Tutkimuksessa oli mukana seitsemän koehenkilöä, jotka olivat kaikki Työterveyslaitoksen työntekijöitä, eikä kenelläkään ollut rakennusalan työkokemusta. Koehenkilöt jaettiin kahteen ryhmään (joiden koot olivat kolme ja neljä henkilöä), jotka suorittivat koepäivän aamupäivällä kaksituntisen koulutuksen Espoon Rudus-turvapuistossa. Iltapäivällä jokainen koehenkilö suoritti itsenäisesti kaikki kolme tässä työssä mainittua rastia (mukaan lukien elämysrastin) puolen tunnin aikana, jonka päätteeksi koehenkilö täytti kyselylomakkeen liittyen virtuaaliseen koulutukseen ja simulaattoripahoinvointikyselyn [Vrt. 31]. Lopuksi molempia ryhmiä haastateltiin erikseen noin tunnin mittaisessa keskustelutuokiossa. Työn kirjoittaja osallistui tutkimuksen suunnitteluun.

Kyselylomake sisälsi 60 väittämää, jotka oli jaettu kahdeksaan eri dimensioon (esitetty taulukossa 4) ja koehenkilöt vastasivat niihin asteikolla 1–5: täysin eri mieltä–täysin samaa mieltä. Taulukossa 4 on esitetty kaikkien dimensioiden väittämien keskiarvot. Siitä nähdään, että koehenkilöt pitivät erityisesti virtuaalisen koulutuksen interaktiivisuudesta ja kokivat tämän koulutuksen olevan myös paremmin omaksuttavissa, kuin todellisen turvapuistokoulutuksen. Sovelluksen helppokäyttöisyyttä kritisoitiin eniten.

Taulukko 4. Virtuaaliturvapuiston arviointitutkimuksen kyselylomakkeen eri dimensioiden tulosten keskiarvot. [30, taulukko 3.]

Dimensio	**Keskiarvo**
Interaktiivisuus	4,3
Omaksuttavuus ja muistettavuus	4,0
Hyödynnettävyys	3,9
Tunteellisuus	3,8
Realistisuus	3,7
Miellyttävyys	3,6
Havainnollistavuus	3,4
Helppokäyttöisyys	3,1

Ryhmähaastatteluissa tuli esille, kuten myös Räsänen ym. (2017) tutkimuksessa [6], että turvapuistokoulutus on hyvin passiivinen kokemus ja koulutuksen asiasisältö on liian suuri muistettavaksi yhdellä kertaa. Virtuaalikoulutuksissa asia on korjattu pilkkomalla koulutussisältö pienempiin osakokonaisuuksiin ja panostamalla interaktiivisuuteen mahdollisuuksien mukaan. Koehenkilöitä havainnointiin virtuaalikoulutusten suorituksen aikana ja kaikissa kehitetyissä rasteissa havaittiin useita ongelmia ohjelmiston käytettävyydessä. Mielenkiintoinen havainto on se, että aikaisempi kokemus virtuaalitodellisuudesta (koehenkilöistä 71 % oli kokeillut aikaisemmin) ei vaikuttanut merkittävästi koulutussuorituksen sujuvuuteen.

8 Tekniset ratkaisut

Tässä luvussa on esitelty sovelluksen ohjelmistoarkkitehtuurin lisäksi kaksi sovelluksessa esiintynyttä ohjelmistoteknistä ongelmaa ja niiden ratkaisut.

8.1 Ohjelmistoarkkitehtuuri

8.1.1 Entiteetti–komponentti-järjestelmä

Unity-pelimoottori käyttää entiteetti–komponentti-järjestelmää (engl. entity–component system, kirjainlyhenne ECS). Ympäristö (Unityssä scene-tiedosto, suomeksi skene tai ympäristö) koostuu entiteeteistä, jotka koostuvat yhdestä tai useammasta komponentista. Tällaisessa järjestelmässä entiteetti ei ole muuta kuin pelkkä tunniste, joka sisältää komponentteja, joskin Unityn entiteetissä (GameObject-oliossa) on mukana myös muita tunnistetietoja ja apumetodeja. Tämä arkkitehtuurimalli korostaa erityisesti koodin uudelleenkäytettävyyttä (DRY-suunnittelumalli [32], englanniksi don't repeat yourself eli älä toista itseäsi) ja pienten luokkien kirjoitusta, joilla on vain yksi toiminto tai vastuualue (SRP-suunnittelumalli [33], englanniksi single responsibility principle eli yhden vastuun periaate). Tämän takia mallia käytettäessä luokkia tulee kirjoitettua enemmän kuin esimerkiksi MVC-mallia (engl. model-view-controller, suomeksi malli-näkymä-ohjain) soveltavassa ohjelmistossa: periytymistä (engl. inheritance) ei tule käytettyä paljon, koska entiteetit koostuvat (engl. composition, suomeksi myös kompositio) pienistä osista.

8.1.2 Ohjelmistoarkkitehtuurikaavion lukeminen

Liitteen 4 (VR-turvapuisto-sovelluksen ohjelmistoarkkitehtuurikaavio) kaaviossa on esitetty projektin ohjelmiston arkkitehtuuri korkealla tasolla menemättä yksityiskohtiin. Entiteetti–komponentti-järjestelmän takia ohjelmistossa ei ole selkeitä tasoja, komponenttien välinen kommunikaatio on tiivistä ja luokkien määrä on suuri. Kaaviossa on listattu vain luokkien ja rajapintojen nimet jaoteltuina eri laajuisiin kokonaisuuksiin. Kokonaisuudet rakentuvat sisäkkäisistä laatikoista ja listoista.

- Suurimmat kokonaisuudet, kuten koko ohjelmisto tai Unity-pelimoottori, on laatikoitu ja nimetty käyttäen lihavoitua versaalitekstiä.
- Toisen tason selkeästi toisistaan erilliset kokonaisuudet on laatikoitu ja nimetty versaalitekstillä.
- Pienten kokonaisuuksien nimi on alleviivattu ja sen alle on listattu kyseisen kokonaisuuden rajapintojen (lihavoitu), abstraktien luokkien (kursivoitu), sekä konkreettisten ja staattisten luokkien (ei eritelty) nimet.
- VRTK-laajennukset ja liikeohjaimet -kokonaisuuksissa käytetty katkoviivalaatikko on lisätty selkeyttämään näiden pienten kokonaisuuksien kommunikaation kuvausta, eikä vaikuta niiden rooliin toisen tason kokonaisuuden sisällä.

Huomataan, että joidenkin luokkien nimet on sommittelusyistä jouduttu kirjoittamaan kapealla tekstillä tai jakamaan kahdelle riville. Kaaviossa esitetyt laatikot ja listat eivät ole ohjelmistoteknisiä rakenteita eivätkä ne vaikuta sovelluksen toimintaan, vaan ne on luotu kaaviota varten auttamaan sovelluksen arkkitehtuurin ymmärtämisessä. Joidenkin pienten kokonaisuuksien sisällä luokat ovat tiukasti kytkettyjä niin sisällöllisesti kuin ohjauksellisestikin [34], eli oliot hakevat toistensa tietoja ja kutsuvat toistensa metodeja.

Tyyppien jaottelun lisäksi kaavioon on kuvattu yleisimmät kokonaisuuksien välisen kommunikaatiot (metodi- ja funktiokutsut) nuolien avulla. Kaikkia luokkien välistä kommunikaatiota ei ole kuvattu, vaan nuoli lisätty kaavioon vain, jos yli puolet kokonaisuuden luokista kutsuu joidenkin toisen kokonaisuuden luokkien metodeja.

- Musta katkoviivanuoli osoittaa luokkien periytymistä. Yli puolet pisteen alla olevan kokonaisuuden luokista perii joitain nuolipäädyn luokkia.
- Punainen yksisuuntainen nuoli osoittaa, että yli puolet pisteen alla olevan kokonaisuuden luokista käyttää joitain nuolipäädyn luokkia ja olioita.
- Pinkki yksisuuntainen nuoli osoittaa, että yli puolet neliön alla olevan kokonaisuuden sekä sen sisältämien muiden kokonaisuuksien luokista käyttää joitain nuolipäädyn luokkia ja olioita.
- Sininen kaksisuuntainen nuoli tarkoittaa samaa kuin punainen nuoli, mutta kommunikaatiota tapahtuu molempiin suuntiin.

Sovelluksen rakenteiden lisäksi kaavioon on otettu mukaan kaksi yleisintä sovelluksessa käytettyä kirjastoa sekä itse pelimoottori.

8.1.3 Ohjelmistoarkkitehtuurikaavion tulkinta

Liitteen 4 kaaviosta huomataan, kuinka tiukasti sovelluksen koodi on kytketty Unity-pelimoottoriin. Merkittävä osa luokista perii pelimoottorin omia luokkia, mutta tämä on tarpeen, sillä luokan ilmentymää ei voi liittää entiteettiin (jolloin sitä kutsutaan komponentiksi) perimättä Unityn rajapinnan MonoBehaviour-luokkaa. Kytkökset kolmannen osapuolen kirjastoihin ovat taas rajoittuneet pelkästään VRTK-kirjastoon.

HTC Vive -laitteiston liikeohjainten ja visiirin tietoihin pääsee käsiksi vain Unityn mukana tulevan OpenVR-rajapinnan kautta, joten tämän sovelluksen kaikki virtuaalitodellisuuslaitteistoa käyttävät funktiokutsut kulkevat VRTK-laajennukset ja Liikeohjaimet-kokonaisuuksien kautta. Tämä selkeyttää ohjelmiston rakennetta. VRTK-laajennukset-kokonai-

suuden luokat nimensä mukaiset perivät VRTK-kirjaston luokkia ja lisäävät niihin toiminnallisuutta korvaten perityn luokan käytön tässä sovelluksessa. Korvaavien luokkien nimeäminen on tehty loogisesti: TPVR_InteractableObject korvaa VRTK_InteractableObject-luokan. Unity-komponentit-kokonaisuuden Muut komponentit -listassa on enimmäkseen sekalaisia apukomponentteja, jotka helpottavat kehitystä.

Käyttöliittymäkomponentit-kokonaisuus jakautuu kolmeen osaan.

- Unity-editori-luokat laajentavat Unityn editorin käyttöliittymää ja tuovat siihen lisätoiminnallisuutta kehityksen nopeuttamiseksi ja helpottamiseksi. Komponentin oletuseditorin korvaaminen omalla mahdollistaa esimerkiksi komponentin tiettyjen tietojen päivityksen sovelluksen suorituksen aikana.
- Esineen toiminnallisuus -luokat tuovat kolmiulotteisiin käyttöliittymäelementteihin (esimerkiksi porakoneeseen) tarvittavan logiikan. Näiden luokkien logiikka on yleistetty, jotta niitä voidaan käyttää kaikissa koulutuksissa. Elementtien rastikohtainen logiikka on kyseisen koulutusrastin luokassa.
- Käyttöliittymäkomponentit-listan luokat ovat perinteisiä kaksiulotteisia käyttöliittymäelementtejä. Esimerkiksi ComplexButton-luokka tarjoaa kaikki samat ominaisuudet kuin Unityn oma Button-luokka, mutta se mahdollistaa useasta eri kuvakomponentista muodostuvan napin kuvien tilan päivityksen samanaikaisesti.

Tapahtumat-kokonaisuus sisältää enimmäkseen Unityn geneerisen abstraktin UnityEvent<T>-luokan konkreettisia toteutuksia eri tyypeille. Konkreettinen toteutus tarvitaan, jotta tapahtuma saadaan näkymään Unityn editorissa. Listassa on myös muita, useita tapahtumia ja tapahtumiin liittyviä toiminnallisuuksia tarjoavia komponentteja. Esimerkiksi ColliderEvents-komponentti mahdollistaa samassa entiteetissä olevien Collider-komponenttien (suomeksi törmääjä) törmäystapahtumien erottelun tavanomaisille fysiikkamallinnetuille törmäyskomponenteille sekä trigger-tyyppisille (suomeksi laukaisija) törmäyskomponenteille, joilla ei ole fysiikkamallinnusta.

Data-kokonaisuudessa on niin kutsuttuja POD-rakenteita (engl. plain old data, suomeksi selvää tai pelkkää dataa), joissa ei ole lainkaan tai vain hyvin vähän logiikkaa. (Olennaista näissä on, että ei kutsuta funktioita oman luokan ulkopuolelta.) Niissä käytetään usein struct-tietorakennetta (suomeksi tietue), tavallisia luokkia tai Unityn ScriptableObject-luokkaa, jolloin ilmentymän tietoja voi helposti muokata Unityn editorin kautta. Sovelluksessa näitä olioita käytetään, kun viedään dataa eri luokkien, mutta erityisesti eri kokonaisuuksien välillä ja tilanteissa, joissa on tarpeen tallentaa monta muuttujaa assosiaatiotauluun (engl. associative array, C#-kielessä dictionary). Esimerkiksi

InfoMedia-luokka tallentaa tekstin ja äänileikkeen, jonka tarkoitus on sisältää olioon tallennettu teksti ääneen puhuttuna. Kun tämä olio annetaan käyttöliittymäelementille, se voi näyttää tekstin lisäksi painikkeen, jonka avulla toistetaan äänileike. Näin toisiinsa tiukasti kytketty data pysyy yhdessä.

Hallinta- ja logiikkaluokat ovat sovelluksen ytimessä. Ne ovat ainokaisia (engl. singleton), eli niistä on vain yksi ilmentymä sovelluksen suorituksen aikana, tai staattisia luokkia, jos niiden ei tarvitse olla komponenttina entiteetissä. Ne eivät sovella välittäjä-suunnittelumallia (engl. mediator), vaan niiden on tarkoitus tarjota toiminnallisuutta usealla samoja asioita käyttäville komponenteille sekä toimia globaalina viestiseinänä kaikille komponenteille. Esimerkiksi staattinen AudioManager-luokka pitää muistissa puhetta sisältävää äänileikettä toistavan äänilähdekomponentin viitteen. Nämä luokat sitovat sovelluksen yhtenäiseksi kokonaisuudeksi: ne kertovat muille komponenteille, mitä tulee tehdä, luovat ja tuhoavat entiteettejä käyttäen muun muassa Tehtaat-kokonaisuuden tehdasluokkia. Tämä huomataan liitteen 4 kaaviosta siitä, että kokonaisuudesta lähtee nuoli melkein jokaiseen muuhun sovelluksen kokonaisuuteen.

Hallintaluokat jakautuvat ympäristökohtaisiin ja globaaleihin luokkiin. Globaalit hallintakomponentit ovat kaikissa koulutusympäristöissä, koska esimerkiksi ympäristön vaihtoa tarvitaan kaikissa koulutuksissa. Infokylttihallintakomponenttia taas tarvitaan vain sellaisessa ympäristössä, jossa on infokylttejä. Hallintaluokat luovat pieniä järjestelmiä sovelluksen sisään, joilla hallitaan tietyn osa-alueen komponentteja ja ovat luonteeltaan muita luokkia laajempia ja teknisempiä.

Logiikkaluokat sisältävät koulutuksen suorittamiseen tarvittavan logiikan sekä päävalikon eri valikkojen toiminnot. Ne alustavat koulutusten ympäristöt aloitustilaan, seuraavat koulutusten etenemistä ja huolehtivat niiden lopettamisesta. Suuri osa sovelluksen sisältöön liittyvästä koodista on tässä listassa. Luokkien koodi ei ole erityisen monimutkaista, sillä niiden tarkoitus on vain pitää muistissa lukuisia viitteitä eri komponentteihin ja päivittää niiden tiloja.

Järjestelmä-kokonaisuus sisältää erinäisiä sovelluksen toiminnan kannalta kriittisiä luokkia, jotka eivät sovi muihin kokonaisuuksiin. Huomataan kuitenkin, että tämän kokonaisuuden luokkia ja rajapintoja käytetään kaikkialla sovelluksessa (kaaviossa violetti nuoli).

C#-kielessä on laajennusominaisuus, jonka avulla voi kirjoittaa omia metodeja luokkiin, joihin ei tavallisesti voisi kirjoittaa uusia metodeja (esimerkiksi enum-tyyppi) tai joiden lähdekoodia ei muuten voi muuttaa (esimerkiksi Unityn luokat). Laajennukset-kokonaisuuteen on koottu lukuisia laajennusmetodeja, jotka helpottavat kehitystä. Esimerkiksi Unityn entiteettiluokkaan (GameObject) on lisätty metodi (esitetty koodissa Esimerkkikoodi 1), joka tarkistaa, onko entiteetissä halutun tyyppistä komponenttia ja palauttaa sen tai luo uuden komponentin entiteettiin. Tämä metodi nopeuttaa koodin kirjoitusta ja antaa kehittäjän työkalun jolla varmistetaan, että komponentti, jota halutaan käyttää, on olemassa entiteetissä.

```
public static T GetOrAddComponent<T>(this GameObject go) where T : Component {
      T comp = go.GetComponent<T>();
      return comp == null ? go.AddComponent<T>() : comp;
}
```

Esimerkkikoodi 1. Koodiesimerkki laajennusmetodista.

8.2 Komponenttien alustusjärjestysongelma

8.2.1 MonoBehaviour-luokka ja skriptien suoritusjärjestyksen ongelma

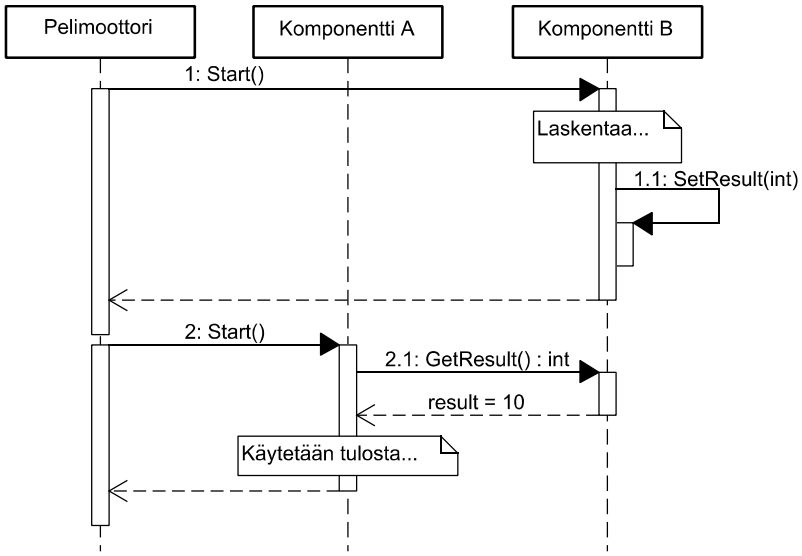
Unityn rajapinnassa MonoBehaviour-luokka on kaikkien komponenttien (entiteetteihin lisättävien olioiden) yläluokka. Luokassa on useita ylikirjoitettavia metodeja, joita pelimoottori kutsuu tietyin väliajoin ja tietyissä kohdissa komponentin elinkaarta. Taulukko 5 havainnollistaa MonoBehaviour-komponentin elinkaaren aikana suoritettavien metodien kategoriat ja nimeää muutamat olennaiset metodit. Taulukosta on jätetty pois suuri osa metodeista selkeyden vuoksi. Kehittäjän itsensä määrittelemät metodit eivät kuulu komponentin elinkaareen, sillä moottori ei kutsu kehittäjän luomia metodeja automaattisesti. Aika kuluu taulukossa ylhäältä alas.

Taulukko 5. Unity-pelimoottorin sisäisten tapahtumafunktioiden kutsujärjestys (lyhennetty). [Vrt. 35.]

	Kategoria	**Metodi**
Aika ↓	Alustus	Awake
		OnEnable
		Start
	Editori	
	Fysiikat	FixedUpdate
	Syöttötapahtumat	
	Logiikka	Update
		(kehittäjän rutiinit)
	Skenen piirto	
	Käyttöliittymän piirto	
	Simulaation pysäytys	
	Purku	OnDisable
		OnDestroy

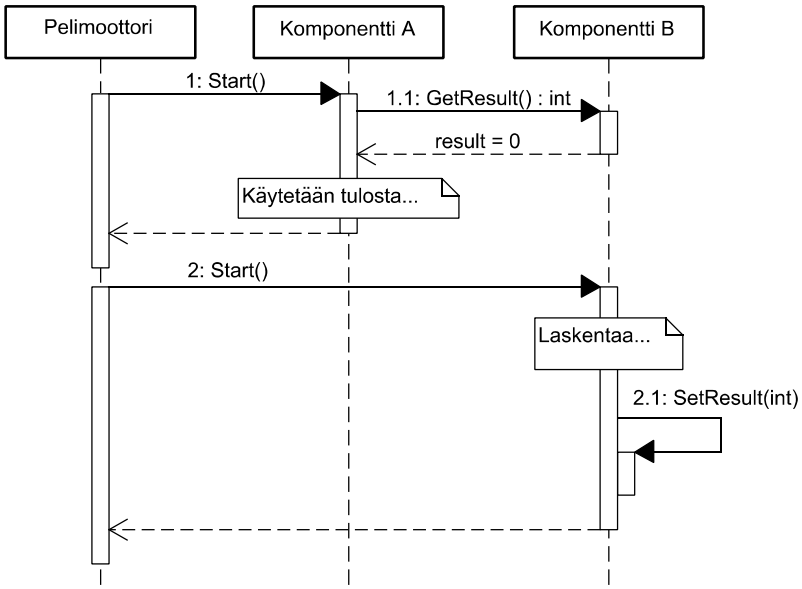
Metodien suoritusjärjestys on järkevä: esimerkiksi Start()-metodia kutsutaan juuri ennen kuin ensimmäinen ruutu piirretään, mikäli komponentti on aktiivinen. Tämän jälkeen esimerkiksi FixedUpdate()-metodia kutsutaan tasaisin ennalta määritetyin väliajoin. FixedUpdate()-metodissa ja kaikissa tämän metodin kutsumissa muissa metodeissa on turvallista käyttää Start()-metodissa asetettuja arvoja, koska voidaan luottaa, että moottori on suorittanut kyseisen metodin ennen FixedUpdate()-metodin kutsua.

Ongelma esiintyy, kun yhden komponentin tulee käyttää toisessa komponentissa olevia julkisia metodeja, eritoten komponentin alustusmetodeissa (Start(), yms.): toista komponenttia, jonka metodeja käytetään, ei välttämättä ole vielä alustettu, jolloin metodikutsu voi aiheuttaa virheen tai väärän tuloksen. Näin tapahtuu, kun kyseinen julkinen metodi käyttää jotain Start()-metodissa laskettua tai asetettua arvoa. Kuvassa 34 on esitetty kahden komponentin alustus sekvenssikaaviona. Tässä kaaviossa ei esiinny ongelmaa komponenttien alustusjärjestyksessä, sillä komponentin A tarvitseman komponentin B alustus tehtiin ensin. Komponentti A saa komponentilta B oikean tuloksen metodikutsun 2.1 paluuarvona.



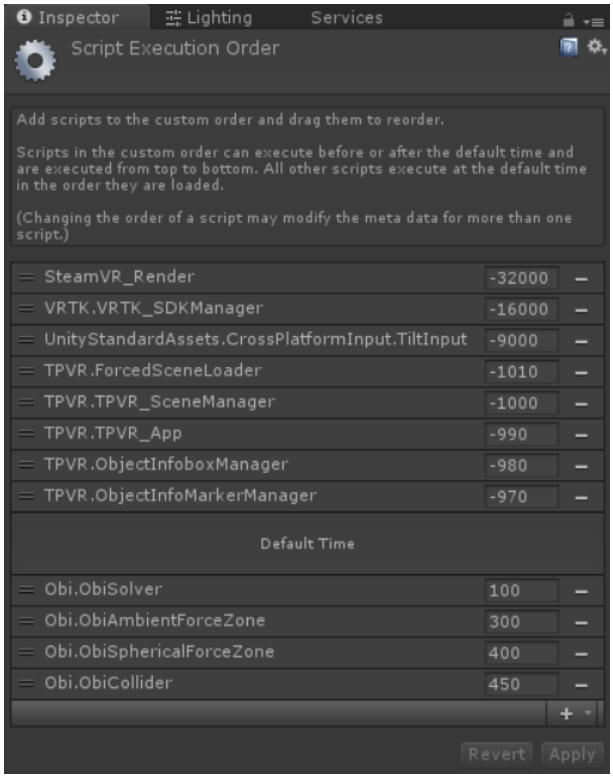
Kuva 34. Sekvenssikaavio kahden Unity-komponentin alustuksesta, jossa alustusjärjestyksessä ei ole ongelmaa.

Vaikka Unity takaa komponentin sisällä taulukon 5 mukaisen metodien suoritusjärjestyksen, eri komponenttien alustusjärjestys on satunnainen. Toisella suorituskerralla kuvan 34 tilanne voi näyttää kuvassa 35 esitetyltä tilanteelta, vaikka lähdekoodia tai komponenttien arvoja ei ole muutettu. Komponentti A alustui ensin ja sai komponentilta B väärän arvon, koska kyseistä komponenttia ei ollut vielä alustettu.



Kuva 35. Sekvenssikaavio kahden Unity-komponentin alustuksesta, jossa alustusjärjestyksessä on ongelma.

Moottorissa on ominaisuus (Script Execution Order -valikko, kuva 36), jonka avulla kehittäjä voi määritellä eri skriptien suoritusjärjestyksen (vaikuttaa alustusjärjestykseen), mutta tämän listan päivittäminen käsin vie merkittävästi aikaa. Se ei myöskään ratkaise ongelmaa, jossa kaksi saman luokan ilmentymää tarvitsevat toistensa tietoja, sillä saman luokan komponenttien alustusjärjestys on myös satunnainen. Kuvan 36 TPVR-nimiavaruudessa (TPVR.-alkuiset nimet) skriptit ovat tähän ohjelmistoon tehtyjä luokkia, joten tätäkin ominaisuutta on hyödynnetty sen puutteista huolimatta.



Kuva 36. Kuvakaappaus sovelluksen Script Execution Order -valikosta.

8.2.2 Järjestelmän käyttötarve

Sovelluksessa on jokaiselle koulutusrastille oma luokka, joka sisältää kyseisen rastin koulutuksen suorittamiseen tarvittavan logiikan, mukaan lukien eri koulutustehtävien tilan seurannan. Korotetun työskentelyn koulutusrastissa (luku 7.3.2) käyttäjän on mahdollista liikutella jatkotikkaita. Tämän rastin virtuaaliympäristö sisältää eri alueita, jotka seuraavat, mitkä entiteetit ovat alueiden sisällä. Koulutusrastia ladatessa, ennen kuin käyttäjä pääsee ympäristöön, on tarpeen tietää, ovatko tikkaat tiettyjen alueiden sisällä, jotta koulutus voidaan päivittää oikeaan tilaan ennen kuin se alkaa.

Tikkaiden sijainti ympäristössä on kuitenkin aina sama koulutusrastin alussa, joten olisi mahdollista kovakoodata tikkaiden sijainti koulutusrastin koodiin. Tämä voisi kuitenkin aiheuttaa bugeja tulevaisuudessa, mikäli tikkaiden sijaintia muutetaan, sillä olisi tarpeen muistaa päivittää myös koodissa olevaa sijaintia. Koska tarvittava koodi tikkaiden sijainnin seuraamiseen on jo olemassa, on helppoa yksinkertaisesti kysyä alueiden entiteettejä seuraavilta komponenteilta, ovatko tikkaat alueiden sisällä ja näin asettaa tikkaiden sijainnin arvo dynaamisesti koulutuksen alussa.

On kuitenkin mahdollista, että koulutusrastin logiikkakomponentti alustetaan ennen kuin entiteettien seurantakomponentti on alustettu. Tässä tilanteessa, kun logiikkakomponentti kysyy seurantakomponentilta, ovatko tikkaat alueen sisällä, vastaus on aina negatiivinen, koska seurantakomponentin lista alueen sisällä olevista entiteeteistä on tyhjä. Ongelmaan on kaksi ratkaisua.

1. Pakotetaan kaikki seurantakomponentit alustumaan ennen logiikkakomponentteja, jolloin niitä on turvallista käyttää logiikkakomponenteista.
2. Pakotetaan logiikkakomponentti odottamaan käytettyjen seurantakomponenttien alustusta, ennen kuin tarvittava tieto kysytään.

Ensin mainittu ratkaisu voidaan toteuttaa käyttäen Unityn omaa Script Execution Order -toimintoa. Edellä mainittujen puutteiden vuoksi ongelma päätettiin kuitenkin ratkaista jälkimmäisellä tavalla kehittämällä oma komponentin alustusjärjestelmä.

8.2.3 Alustusjärjestelmä

Alustusjärjestelmä on laajalti käytössä muualla ohjelmistossa, myös luvussa 8.2.2 esitetyn esimerkin lisäksi. Järjestelmä koostuu kolmesta osasta:

- IInitializable-rajapinta
- abstrakti InitializableBehaviour-luokka
- InitializationNotifier-luokka.

Ohjelmistoarkkitehtuurikaaviossa (liite 4) järjestelmä on Järjestelmä ja Unity-komponentit-laatikoissa. Koko järjestelmän koodi on liitteessä 7 (VR-turvapuisto-sovelluksen alustusjärjestelmän koodi) ilman dokumentaatiota, kommentteja, debugaustulostuksia ja epäolennaisia muodostimia.

IInitializable-rajapinta (liitteessä 7) on hyvin yksinkertainen ja määrittelee alustusjärjestelmän kaksi olennaisinta osaa: IsInitialized-ominaisuuden, jolla tarkistetaan, onko rajapinnan toteuttava komponentti alustettu ja julkisen Initialize(bool)-metodin, jolla komponentti alustetaan. On myös huomattava, että IsInititialized-ominaisuudella ei ole julkista asetustoimintoa, joten ainoastaan komponentti itse voi muuttaa sen arvoa. Rajapinnan OnInitialized(IInitializable)-tapahtuma mahdollistaa liitteen 7 InitializationNotifier-luokan toiminnan.

Abstrakti InitializableBehaviour-luokka (liitteessä 7) toteuttaa IInitializable-rajapinnan ja perii Unityn MonoBehaviour-luokan. Luokkaa on tarkoitus käyttää MonoBehaviour-luokan korvaajana, mikäli mahdollista, sillä se poistaa toisteista koodia ja tarjoaa kehitystä helpottavaa lisätoiminnallisuutta alustettaviin komponentteihin. Start()-metodiin on lisätty kutsu perivän luokan Initialize(bool)-metodiin toisteisen koodin vähentämiseksi ja kannustamaan tämän alustusjärjestelmän käyttöä Unityn omien alustusmetodien sijaan. SetInitialized()-metodi helpottaa kehitystä tarjoamalla yksinkertaisen tavan päättää komponentin alustus. On kuitenkin huomattava, että oletuksena järjestelmä ei salli komponentin alustustilan vaihtamista takaisin alustamattomaksi. Initialize(bool)-metodi on asetettu abstraktiksi, jotta perivien luokkien on pakko kirjoittaa sille oma toteutus. Metodi määrittelee yhden totuusarvoparametrin, jonka oikea käyttö ratkaisee ongelman luokkien periytymishierarkiassa (esitelty luvussa 8.2.4), jossa InitializableBehaviour on luokan isovanhempi vanhemman sijaan.

InitializationNotifier-luokan (liitteessä 7) ilmentymät ovat kertakäyttöisiä olioita, joiden on tarkoitus olla elossa vain niin kauan, kunnes kaikki olion muodostimelle annetut IInitializable-rajapinnan toteuttavat komponentit on alustettu. Alustusten jälkeen olio kutsuu sille annettua takaisinkutsuoliota ja siirtyy roskienkeruuseen seuraavan ruudun piirron alussa, koska seurattavien komponenttien tapahtumarekisteröinti on poistettu.

InitializationNotifier-olion käyttö on demonstroitu esimerkkikoodissa 2, jossa ControllerTriggerToucher-komponentin alustuksessa tulee odottaa, että samassa entiteetissä oleva InteractTouchControllerReplacer-komponentti on alustettu ensin. InteractTouchControllerReplacer-komponentti luo entiteettiin uusia lapsientiteettejä dynaamisesti, joista yhdessä on tämän komponentin tarvitsema TriggerColliderEvents-komponentti. Huomataan, että alustusmetodissa luotua ilmoitusolion viitettä ei tallenneta mihinkään, vaan olio siirtyy roskienkeruuseen itsestään. Esimerkkikoodissa 2 on ilmoitusolion muodostimelle annettu metodiviitteen sijaan anonyymi lambda-metodi, koska

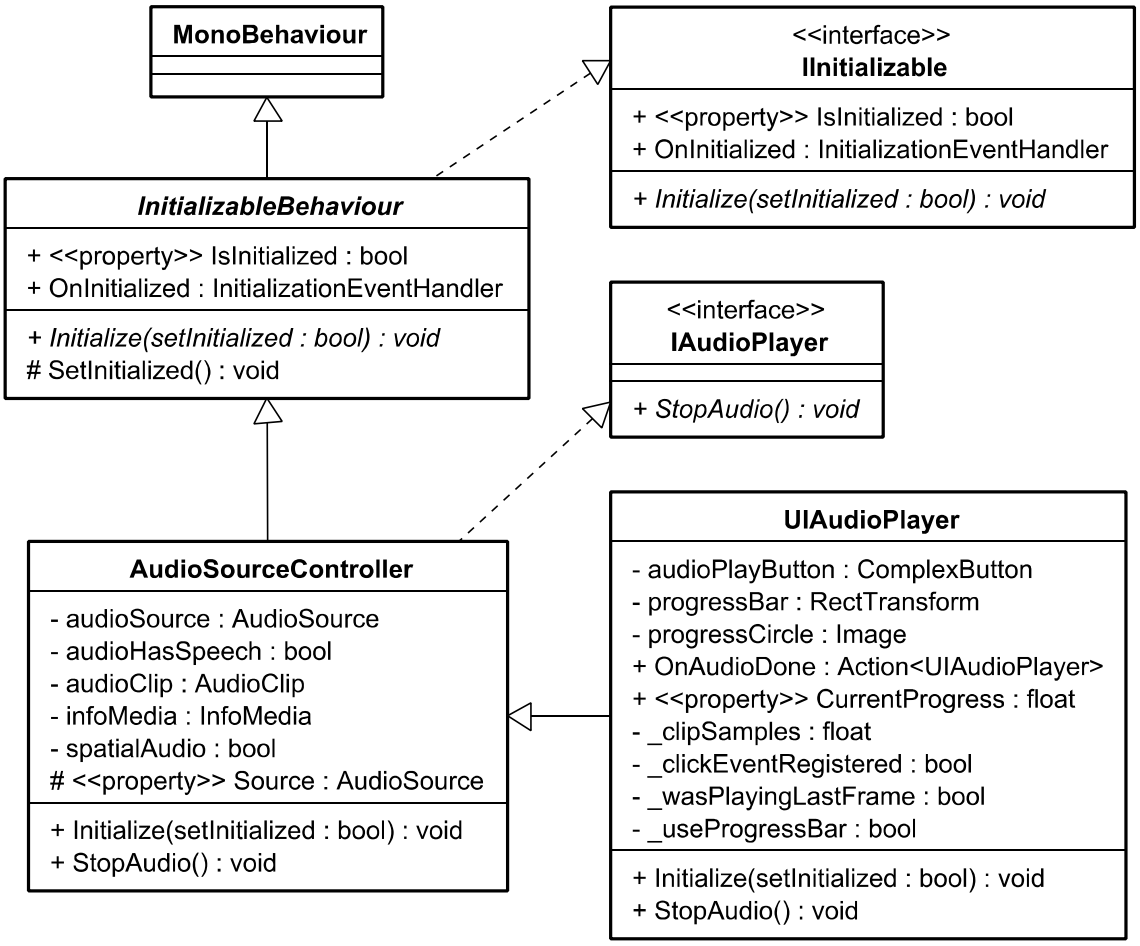
metodia kutsutaan vain kerran ja ettei setInitialized-parametria tarvitsisi tallentaa väliaikaisesti muistiin.

```
public override void Initialize(bool setInitialized = true) {
     _controllerReplacer = GetComponent<InteractTouchControllerReplacer>();

     new InitializationNotifier(this, () => {
          _triggerEvents = GetComponentInChildren<TriggerColliderEvents>();

          if (_triggerEvents != null) {
               _triggerEvents.OnTriggerEntered += ControllerTriggerEnter;
               _triggerEvents.OnTriggerExited += ControllerTriggerExit;
          }

          if (setInitialized) {
               SetInitialized();
          }
     }, _controllerReplacer);
}
```

Esimerkkikoodi 2. ControllerTriggerToucher-luokan Initialize(bool)-metodi ilman dokumentaatiota, kommentteja, debugaustulostuksia ja -muuttujia.

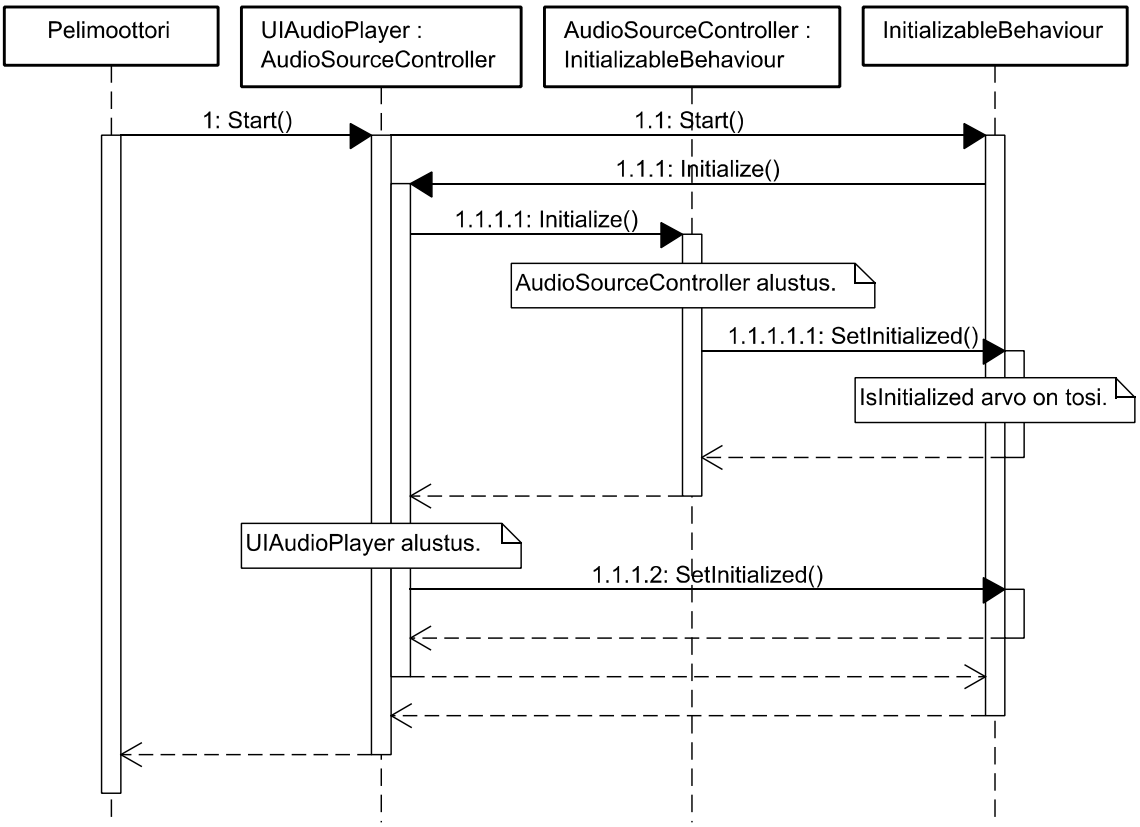
8.2.4 Järjestelmän periytymisongelman ratkaisu

Kehitetyssä komponentin alustusjärjestelmässä oli ongelma periytymisen kanssa. Esimerkiksi ohjelmiston UIAudioPlayer-luokka (luokkakaavio kuvassa 37) hallitsee äänien toistoa käyttöliittymän napin kautta. Tämä luokka perii AudioSourceController-luokan, joka hoitaa äänien toiston yleisellä tasolla. Se pitää huolen esimerkiksi siitä, että kahta äänileikettä, joissa on puhetta, ei toisteta samaan aikaan. Tämä luokka perii InitializableBehaviour-luokan, jolloin InitializableBehaviour-luokka on UIAudioPlayer-luokan isovanhempi. UIAudioPlayer-luokan periytymishierarkia on esitetty kuvassa 37 yksinkertaistettuna: mukana on vain jäsenmuuttujat ja ominaisuudet, eikä Unityn omaa MonoBehaviour-luokkaa ole avattu nimeä enempää.



Kuva 37. Luokkakaavio UIAudioPlayer-luokan periytymishierarkiasta, jossa esitettynä sen olennaiset osat lyhennettynä.

Jotta UIAudioPlayer-komponenttia voidaan käyttää, tulee ensin suorittaa sen perimän AudioSourceController-luokan alustusmetodi, sillä komponentti tarvitsee sen vanhemman alustamia arvoja. Ilman Initialize(bool)-metodin totuusarvoparametrin tarkistusta komponentin IsInitialized-ominaisuuden arvo asetettaisiin todeksi liian aikaisin. Tämä ongelma on esitetty kuvassa 38, jossa Initialize()-metodilla ei ole totuusarvoparametria. Huomataan, että IsInitialized-ominaisuuden arvo asetetaan todeksi, kun vanhemman alustus on valmis ja ennen kuin UIAudioPlayer-komponentin alustus edes alkaa.



Kuva 38. UIAudioPlayer-komponentin alustuksen metodikutsut sekvenssikaaviona. Tässä UIAudioPlayer-komponentin alustustilan arvo muuttuu liian aikaisin.

Ongelmaan on yksinkertainen ratkaisu: kun suoritetaan komponentin alustusmetodi, annetaan argumenttina totuusarvo (ei kuvattu sekvenssikaaviossa), joka kertoo metodille, saako sen suorituksen lopuksi asettaa komponentin IsInitialized-ominaisuuden todeksi ja laukaista alustustapahtuman. Mikäli sekvenssikaavion Initialize()-metodikutsussa 1.1.1.1 olisi totuusarvo ja se olisi epätosi, jäisi metodikutsu 1.1.1.1.1 pois ja SetInitialized()-metodia kutsuttaisiin ensimmäisen kerran vasta metodikutsun 1.1.1 jälkeen, eli UIAudioPlayer-komponentin alustuksen lopuksi.

Totuusarvoparametri ratkaisee ongelman tehokkaasti ja yksinkertaisesti, mutta ongelmana on, että kehittäjän tulee muistaa toteuttaa arvon tarkistus jokaisen komponentin Initialize(bool)-metodissa. Mikäli periytymishierarkiassa on yksikin komponentti, joka ei ole toteuttanut tätä tarkistusta, alustusjärjestelmä toimii yhä, mutta komponentin käytössä voi esiintyä odottamattomia ongelmia alustamattomien muuttujien tai väärien tulosten kanssa. Muut komponentit luulevat komponentin olevan alustettu kokonaan, vaikka alustus on vielä kesken tai sitä ei ole tehty ollenkaan.

8.2.5 Järjestelmän jatkokehitys

Tällä hetkellä yksinkertainen alustusjärjestelmä palvelee sovelluksen tarpeita, mutta sitä olisi mahdollista kehittää pidemmälle. Yksi ongelma on epäselvyys siitä, voiko komponentin alustaa useamman kuin yhden kerran. Tällä hetkellä monet järjestelmää käyttävät komponentit ovat käärineet koko Initialize(bool)-metodin sisällön ehtolauseen sisään, joka tarkistaa, onko komponentti jo alustettu. Muussa tapauksessa metodi ei tee mitään. Jokaisen alustusmetodin dokumentaatiossa on kuitenkin mainittu, voiko metodia kutsua useamman kuin yhden kerran. Esimerkki tällaisesta metodista on esimerkkikoodissa 3. Kehityksen kannalta olisi mielekästä luoda esimerkiksi Reinitialize(bool)-metodi, jotta uudelleenalustuksen mahdollisuus olisi selkeämpi.

```
/**
 * <summary>
 * Can only be initialized once.
 * </summary>
 * <param name="setInitialized">set component state to initialized (default:
true)</param>
 */
public override void Initialize(bool setInitialized = true) {
      if (!IsInitialized) {
            _manager = TPVR_App.RequireManager<AnimatorStateManager>(this);
            _animator = GetComponent<Animator>();
            _manager.OnAnimationStop += OnAnimationStop;
            _animator.enabled = true;

            if (setInitialized) {
                  SetInitialized();
            }
      }
}
```

Esimerkkikoodi 3. InitializableBehaviour-luokan perivän TPVR_AnimationPlayer-luokan Initialize(bool)-metodi kokonaisuudessaan.

Järjestelmä ei myöskään tue alustuksen perumista (IsInitialized-ominaisuuden arvon muuttamista takaisin epätodeksi), joka voisi olla hyödyllinen ominaisuus erityisesti monta kertaa alustettavien komponenttien kanssa. Näiden komponenttien kanssa voisi alustuksen tilan vaihtaa takaisin alustamattomaksi, kun komponentin tilaa (olion jäsenmuuttujia) muutetaan tavalla, joka vaatii komponentin alustuksen, ennen kuin uudet arvot tulevat käyttöön.

8.3 VRTK-kirjaston liikeohjaimen alustusjärjestysongelma

VRTK-kirjastossa on VRTK_TrackedController-luokka, joka abstrahoi eri virtuaalitodellisuusjärjestelmien liikeohjaimet yhdeksi komponentiksi. Tämän komponentin kautta on mahdollista käyttää esimerkiksi HTC Viven tai Oculus Riftin liikeohjaimen toimintoja. Luokassa on ControllerModelAvailable-tapahtuma, joka lähettää tiedon tapahtuman seuraajille siitä, että tämä liikeohjainentiteetti on valmis käytettäväksi. Tapahtuma laukaistaan aikaisintaan Unity-komponentin alustuksen OnEnable-metodissa (metodin paikka komponentin elinkaaressa on esitetty taulukossa 5), mutta laukaisu voi myös tapahtua myöhemmin. Olennaiset osat luokan koodista ovat esimerkkikoodissa 4, jossa on mukana myöhemmin lisättyjä selventäviä kommentteja.

```
public class VRTK_TrackedController : MonoBehaviour {
     public event VRTKTrackedControllerEventHandler ControllerModelAvailable;

     protected virtual void OnEnable() {
          // OnControllerModelAvailable-metodia kutsutaan pitkän
metodikutsuketjun päätteeksi, joka alkaa tästä.
     }

     public virtual void
OnControllerModelAvailable(VRTKTrackedControllerEventArgs e) {
          // Jos tapahtumassa on tilaajia.
          if (ControllerModelAvailable != null) {
               // Laukaise tapahtuma.
               ControllerModelAvailable(this, e);
          }
     }
}
```

Esimerkkikoodi 4. VRTK_TrackedController-luokan olennaiset osat.

Ongelma esiintyy, kun tapahtumaan rekisteröidään tilaaja sen laukeamisen jälkeen. Näin voi käydä, kun tilaaja rekisteröidään esimerkiksi Start-metodissa, joka suoritetaan OnEnable-metodin jälkeen tai kun liikeohjain on alustunut odottamattoman aikaisin. Tällöin myöhemmin rekisteröity tilaaja ei saa ikinä tietoa siitä, että ohjain on valmis käytettäväksi. Mikäli ohjainta tarvitseva komponentti on rekisteröinyt tilaajan myöhässä ja se on ohjelmoitu odottamaan ohjaimen alustusta, komponentti ei välttämättä toimi lainkaan. Tämä johtaa ei-deterministisiin bugeihin, koska Unity-pelimoottori ei suorita eri komponenttien OnEnable-metodeja aina samassa järjestyksessä. Edellä mainittu komponentti voi yllättäen lakata toimimasta jollain suorituskerralla, vaikka koodia tai komponentteja ei olisi muutettu suoritusten välissä. Tässä sovelluksessa tämä ongelma esiintyi ympäristön vaihdon aikana, kun ohjaimet vaihdetaan ympäristön koulutukseen sopiviksi.

Ongelman korjaaminen vaati VRTK_TrackedController-luokan muokkaamista. Luokan muokattu versio on liitteessä 8 (Muokattu VRTK_TrackedController-luokka), jossa mukana on myöhemmin lisättyjä selventäviä kommentteja. OnControllerModelAvailable-metodi tallentaa nyt ohjaimen alustustiedot muistiin yksityiseen jäsenmuuttujaan. ControllerModelAvailable-tapahtuma muutettiin ominaisuudeksi, jotta ei rikota jo olemassa olevaa koodia ja itse tapahtumalle luotiin uusi yksityinen jäsenmuuttuja. Nyt tilaajan rekisteröinnin jälkeen tarkistetaan, onko ohjain alustunut, ja tapahtuma laukaistaan uudelleen heti rekisteröinnin jälkeen, mikäli ohjaimen alustustiedot ovat muistissa.

Luokan muuttaminen ratkaisi esitetyn ongelman tehokkaasti yksinkertaisella tavalla, mutta toi lisävaatimuksia tapahtuman käyttäjille. Alkuperäinen järjestelmä laukaisi tapahtuman vain yhden kerran, kun liikeohjain alustettiin, koska ohjain voi alustua vain kerran. Nyt tapahtuma voi laueta useamman kerran: aina, kun tapahtumaan rekisteröidään uusi tilaaja liikeohjaimen alustuksen jälkeen. Jos tapahtuman käyttäjät haluavat tiedon ohjaimen alustuksesta vain yhden kerran, tulee käyttäjien poistaa tilaajan rekisteröinti sen jälkeen, kun tapahtuma on lauennut. Tämä ei kuitenkaan ole aiheuttanut ongelmia tapahtumaa käyttävien VRTK-kirjaston komponenttien kanssa, eikä muiden luokkien koodia ole ollut tarpeen muuttaa.

9 Yhteenveto

Insinöörityön tavoitteena oli kehittää virtuaalitodellisuuskoulutussovellus ja siihen muutama esimerkkikoulutus, josta voidaan myöhemmässä vaiheessa jatkokehittää kaupallinen ohjelmistotuote, joka tukee eri oppimistapoja ja on helppokäyttöinen riippumatta käyttäjän tietoteknisistä taidoista. Kaupalliseen sovellukseen on tarkoituksena kehittää sisältönä ensisijaisesti rakennusalan työturvallisuuskoulutuksia. Projektin ohjelmistokehitys toteutettiin ketterästi ja kokeilevasti ilman tarkkaa suunnitelmaa. Pääpainona oli kokeilla erilaisia käytettävyysratkaisuja. Insinöörityön aikana keväällä 2018 keskityttiin erityisesti:

- sovelluksen lähdekoodin refaktorointiin
- aikaisemmin havaittujen ongelmien korjaamiseen
- käyttöliittymän ja käytettävyyden ehostamiseen
- vanhojen ja uusien käyttäjäkokemusten arviointiin
- projektin suunnitelman laajentamiseen

- uusien ominaisuuksien ideoimiseen.

Projektin aikana todettiin, että virtuaalitodellisuutta voidaan hyödyntää työturvallisuuskoulutukseen, joskin projektin käytössä olleen teknologian nykyinen taso aiheuttaa joitain rajoituksia koulutusten sisällöille. Laaja käyttäjäkohderyhmä aiheutti haasteita sovelluksen käyttökokemuksen kehityksessä, mutta erityisen haasteellisiksi aiheiksi havaittiin kaikki hienomotoriikkaa vaativat toimenpiteet, kuten sormien käyttö tai pienten esineiden käsittely. Virtuaalitodellisuuden tuomista mahdollisuuksista havaittiin, että tavallisesti vaarallisten tilanteiden havainnollistaminen on mahdollista toteuttaa turvallisesti virtuaaliympäristössä aiheuttamatta käyttäjälle todellista vaaraa. Insinöörityön kirjoituksen päätteeksi liitteiden 1 ja 2 taulukoiden toteutettu-sarakkeen arvot päivitettiin vastaamaan sovelluksen tilaa toukokuussa 2018: ✓-merkintä rivillä tarkoittaa toteutettua ominaisuutta, ~-merkintä kesken olevaa ominaisuutta ja X-merkintä toistaiseksi toteuttamatta olevaa ominaisuutta. Päivityksen jälkeinen tilanne sovelluksen ominaisuuksista on tilastoitu taulukkoon 6, josta nähdään, että suurin osa syksyllä 2017 suunnitelluista ominaisuuksista saatiin toteutettua.

Taulukko 6. Sovellusten ominaisuuksien toteutuksen tilanne toukokuussa 2018.

Ominaisuus	**Alkuperäinen tavoite**	**Toteutettu**	**Osittain toteutettu**
Turvapuistokoulutuksessa oleva ominaisuus	9	6	1
Turvapuistokoulutuksen kehitysehdotukset	6	5	0
Yhteensä	**15**	**11**	**1**

Reuter (2017) tutkimuksen [30] aikana havaitut ongelmat ohjelmistossa on koottu liitteen 9 (Virtuaaliturvapuiston arviointitutkimuksen tulosten ominaisuustaulukko) taulukkoon. Jokaisen ongelman kohdalle on lisätty sen tila sovelluksessa toukokuussa 2018: korjattu-sarakkeessa ✓-merkintä rivillä tarkoittaa korjattua, ~-merkintä osittain korjattua ja X-merkintä ei korjattua ongelmaa. Tästä nähdään, että noin 48 % syksyllä havaituista ongelmista on korjattu, ainakin osittain, sovelluksessa insinöörityön loppuun mennessä. Raportissa tuotiin esille ongelma liittyen korotetun työskentelyn koulutusrastin haastavuuteen, mutta tätä ei huomioida ongelmana, sillä lopputestinä kyseisen koulutusrastin on tarkoitus olla muita haastavampi.

Sovellusta on esitelty monille mahdollisille yhteistyökumppaneille, Työterveyslaitoksen vieraille sekä yhteensä sadoille ihmisille vuoden 2017 Työterveyspäivillä ja vuoden 2018 Kiss my Safety -seminaarissa. Esittelyistä saatu positiivinen palaute ja kritiikki on ollut tärkeää sovelluksen ominaisuuksien suunnittelussa. Insinöörityön kirjoituksen aikana ei kuitenkaan toteutettu mitattuja käyttäjätestejä, eikä koulutussisältöjä kehitetty tarpeeksi, jotta olisi voitu selvittää sovelluksen vaikutus työntekijän turvallisuuskäsitykseen. Käyttäjiltä on tullut eniten kritiikkiä käyttöliittymästä ja helppokäyttöisyydestä, joten on todettava, että tähän tavoitteeseen ei päästy, ja se vaatii vielä jatkokehitystä. Kehitettyjen ympäristöjen immersiivisyyttä on kuitenkin kehuttu, vaikka niiden visuaaliseen ilmeeseen ei panostettu. Sovelluksen haluttiin tukevan eri opetustapoja ja vaikka koulutuksen sisältö ei ole riippuvainen sovelluksesta, itse opetustyyli on rajoitettu. Tällä hetkellä ohjelmistolla on mahdollista rakentaa koulutuksia, joissa.

- tavoite on tarkoin määritelty jokaiselle koulutusrastille
- yhden aihealueen sisältö jaettu pieniin kokonaisuuksin (koulutusrasteihin)
- koulutus etenee vaiheittain ja suoraviivaisesti rastin aikana
- sovellus ohjaa käyttäjää koulutusten läpi
- koulutustehtävien yhteydessä käyttäjä saa palautetta toiminnastaan.

Toisin sanoen ensisijaisesti tuetaan vain behavioristisen opetuksen periaatteita [24].

Edellä mainitun ominaisuuksien toteutustilan päivityksen lisäksi Räsänen ym. (2017) tutkimuksen [6] loppuraportin haastattelujen tulokset käytiin uudelleen läpi ja liitteiden 1, 2 ja 9 taulukoiden sisältö koottiin liitteen 10 (Sovelluksen lopullinen ominaisuustaulukko) taulukkoon uusien ominaisuuksien kanssa. Se on sovelluksen jatkokehitystä ohjaava ominaisuuslista. Tämän päivityksen yhteydessä ominaisuuslistaan lisättiin 16 uutta ominaisuutta, eli alkuperäisten ominaisuuksien määrä yli kaksinkertaistettiin. Sovellusta ei saatu vielä valmiiksi, vaan ohjelmistokehitys jatkuu aktiivisesti myös tulevaisuudessa. Työterveyslaitos on kuitenkin tyytyväinen kehitettyyn sovellukseen, joten siitä lähdetään kehittämään myytävää ohjelmistotuotetta. Sovellukseen kehitettäviä työturvallisuuskoulutuksia käytetään monissa yrityksissä parantaen näiden henkilöstön turvallisuusosaamista ja toiveena on myös edistää turvallisuuskoulutuksen yleistä tasoa. Lisäksi on mahdollista, että tuote viedään tulevaisuudessa kansainvälisille markkinoille.

Lähteet

1 Modernia turvallisuusoppimista rakennusalalle – MoSaC (2018-2020). 2018. Verkkoaineisto. Työterveyslaitos. <https://www.ttl.fi/tutkimushanke/modernia-turvallisuusoppimista-rakennusalalle-mosac/>. Luettu 9.5.2018.

2 Tyotapaturmat - Tilastokirjan taulukot 2015. 2015. Verkkoaineisto. Tapaturmavakuutuskeskus. <http://www.tvk.fi/templates/vinha/services/download.aspx?fid=333495&hash=ffb7e1642be961b1fca2cfff0b3d6245c75244af0a700ab7ad52787000e88b7f>. Luettu 6.2.2018.

3 Luokitukset. 2010. Verkkoaineisto. Tapaturmavakuutuskeskus. <http://www.tvk.fi/tietopalvelu-ja-julkaisut/tilastot/luokitukset/>. Luettu 6.2.2018.

4 Rastit. Verkkoaineisto. Rudus Oy. <http://www.turvapuisto.fi/rastit>. Luettu 6.2.2018.

5 Toiminta. Verkkoaineisto. Pelastusopisto. <http://www.ttha.fi/toiminta/>. Luettu 6.2.2018.

6 Räsänen ym. 2017. Turvapuistot työturvallisuuskoulutuksen oppimisympäristöinä. Helsinki: Työterveyslaitos.

7 Milgram & Kishino. 1994. A Taxonomy of Mixed Reality Visual Displays. IEICE Transactions on Information and Systems 1.12.1994, s. 1321–1329.

8 Henry2.zip. 2017. Verkkoaineisto. Oculus VR. <https://en.oculusbrand.com/assets/experiences>. Luettu 23.4.2018.

9 Núñez & Blake. 2001. Cognitive Presence as a Unified Concept of Virtual Reality Effectiveness. Kapkaupunki: University of Cape Town.

10 Biocca, Frank. 1997. The Cyborg's Dilemma: Progressive Embodiment in Virtual Environments. Journal of Computer-Mediated Communication 1.9.1997.

11 Schneck ym. 2011. Prosthetic Vision Assessment. Visual prosthetics: Physiology, bioengineering, rehabilitation 1.1.2001, s. 385–412.

12 Coyau / Wikimedia Commons / CC BY-SA 3.0. 2011. Verkkoaineisto. <https://commons.wikimedia.org/wiki/File:Ch%C3%A2teau_de_Versailles,_petit_appartement_de_la_reine_(2e_%C3%A9tage),_salle_%C3%A0_manger,_La_Nourrice,_Josse-Fran%C3%A7ois-Joseph_Leriche_d'ap._Louis_Boizot_04.jpg>. Luettu 13.4.2018.

13 Real Reality (RR). Verkkoaineisto. Techopedia Inc. <https://www.techopedia.com/definition/12261/real-reality-rr>. Luettu 19.1.2018.

14 Lang, Ben. Hands-on: Intel's Project Alloy 'Merged Reality' Roomscale Multiplayer Demo. 2017. Verkkoaineisto. <https://www.roadtovr.com/intel-project-alloy-merged-reality-multiplayer-hands-on-ces-2017/>. 17.1.2017. Luettu 19.1.2018.

15 Molitch-Hou, Michael. HP Announces Sprout Pro for Blended Reality and 3D Printing. 2016. Verkkoaineisto. <https://3dprintingindustry.com/news/hp-announces-sprout-pro-mixed-reality-computer-65105/>. 19.1.2016. Luettu 19.1.2018.

16 What is Extended Reality (XR)? All you need to know. 2017. Verkkoaineisto. Reality Dome. <http://www.realitydome.com/what-is-extended-reality-xr/>. 4.10.2017. Luettu 19.1.2018.

17 2017 Year in Review. 2018. Verkkoaineisto. SuperData Research Holdings, Inc. <https://www.superdataresearch.com/market-data/market-brief-year-in-review/>. Luettu 22.2.2018.

18 Sales of consumer VR headsets worldwide from 2016 to 2017 (in million units), by platform. 2018. Verkkoaineisto. Statista. <https://www.statista.com/statistics/755777/consumer-vr-headset-unit-sales-by-platform-worldwide/>. Luettu 22.2.2018.

19 Malventano, Allyn. SteamVR HTC Vive In-depth - Lighthouse Tracking System Dissected and Explored. 2016. Verkkoaineisto. <https://www.pcper.com/reviews/General-Tech/SteamVR-HTC-Vive-depth-Lighthouse-Tracking-System-Dissected-and-Explored>. 5.4.2016. Luettu 5.4.2018.

20 Vive Photos. 2016. Verkkoaineisto. HTC Corporation. <http://dl3.htc.com/us/press-kit/htc-vive/htc-vive-images-20160414.zip>. 14.4.2016. Luettu 3.4.2018.

21 Reid, Stephen "Rockjaw". Re: Controllers on Bluetooth or WiFi. 2016. Verkkoaineisto. <https://community.viveport.com/t5/General-Vive-Discussion/Controllers-on-Bluetooth-or-WiFi/m-p/4940/highlight/true#M1262>. 15.12.2016. Luettu 19.2.2018.

22 Welcome to Steamworks. Verkkoaineisto. Valve Corporation. <https://partner.steamgames.com/vrlicensing>. Luettu 19.2.2018.

23 User guide. 2018. Verkkoaineisto. HTC Corporation & Valve Corporation. <http://dl4.htc.com/web_materials/Manual/Vive/Vive_User_Guide.pdf>. Luettu 21.5.2018.

24 Pylkkä, Outi. Behavioristinen oppimiskäsitys ja oppimisen ohjaaminen. Verkkoaineisto. <http://oppimateriaalit.jamk.fi/oppimiskasitykset/oppimiskasityksista-oppimisen-ohjaamiseen/behavioristinen-oppimiskasitys-ja-oppimisen-ohjaaminen/>. Luettu 23.4.2018.

25 Pylkkä, Outi. Kokemuksellinen oppiminen ja oppimisen ohjaaminen. Verkkoaineisto. <http://oppimateriaalit.jamk.fi/oppimiskasitykset/oppimiskasityksista-oppimisen-ohjaamiseen/kokemuksellinen-oppiminen-ja-oppimisen-ohjaaminen/>. Luettu 23.4.2018.

26 Ratkaisuja putoamissuojainten kiinnittämiseksi – Taivaskoukku. Verkkoaineisto. Rudus Oy. <http://www.turvapuisto.fi/rastit/20286/ratkaisuja-putoamissuojainten-kiinnittamiseksi-taivaskoukku?groupid=1590>. Luettu 21.5.2018.

27 Stanney ym. 1997. Cybersickness is Not Simulator Sickness. Proceedings of the Human Factors and Ergonomics Society Annual Meeting 1.10.1997, s. 1138–1142.

28 Kolasinski, Eugenia. 1995. Simulator Sickness in Virtual Environments. Yhdysvallat: United States Army Research Institute for the Behavioral and Social Sciences.

29 Davis ym. 2015. Comparing the onset of cybersickness using the Oculus Rift and two virtual roller coasters. Proceedings of the 11th Australasian Conference on Interactive Entertainment 30.1.2015, s. 30.

30 Reuter, Josefin. 2017. Virtuaaliturvapuiston arviointitutkimus. Helsinki: Työterveyslaitos.

31 Kennedy ym. 1993. Simulator Sickness Questionnaire: An Enhanced Method for Quantifying Simulator Sickness. The International Journal of Aviation Psychology 1.7.1993, s. 203–220.

32 Smith, Steve. Don't Repeat Yourself. 2009. Verkkoaineisto. <http://programmer.97things.oreilly.com/wiki/index.php/Don%27t_Repeat_Yourself>. 24.11.2009. Luettu 14.5.2018.

33 Single Responsibility Principle. Verkkoaineisto. OODesign.com. <https://www.oodesign.com/single-responsibility-principle.html>. Luettu 14.5.2018.

34 Lappalainen & Tervonen. Ohjelmistotekniikka: Luento 5. 2015. Verkkoaineisto. <https://noppa.oulu.fi/noppa/kurssi/811335a/luennot/811335A_luento_5.pdf>. 21.1.2015. Luettu 14.5.2018.

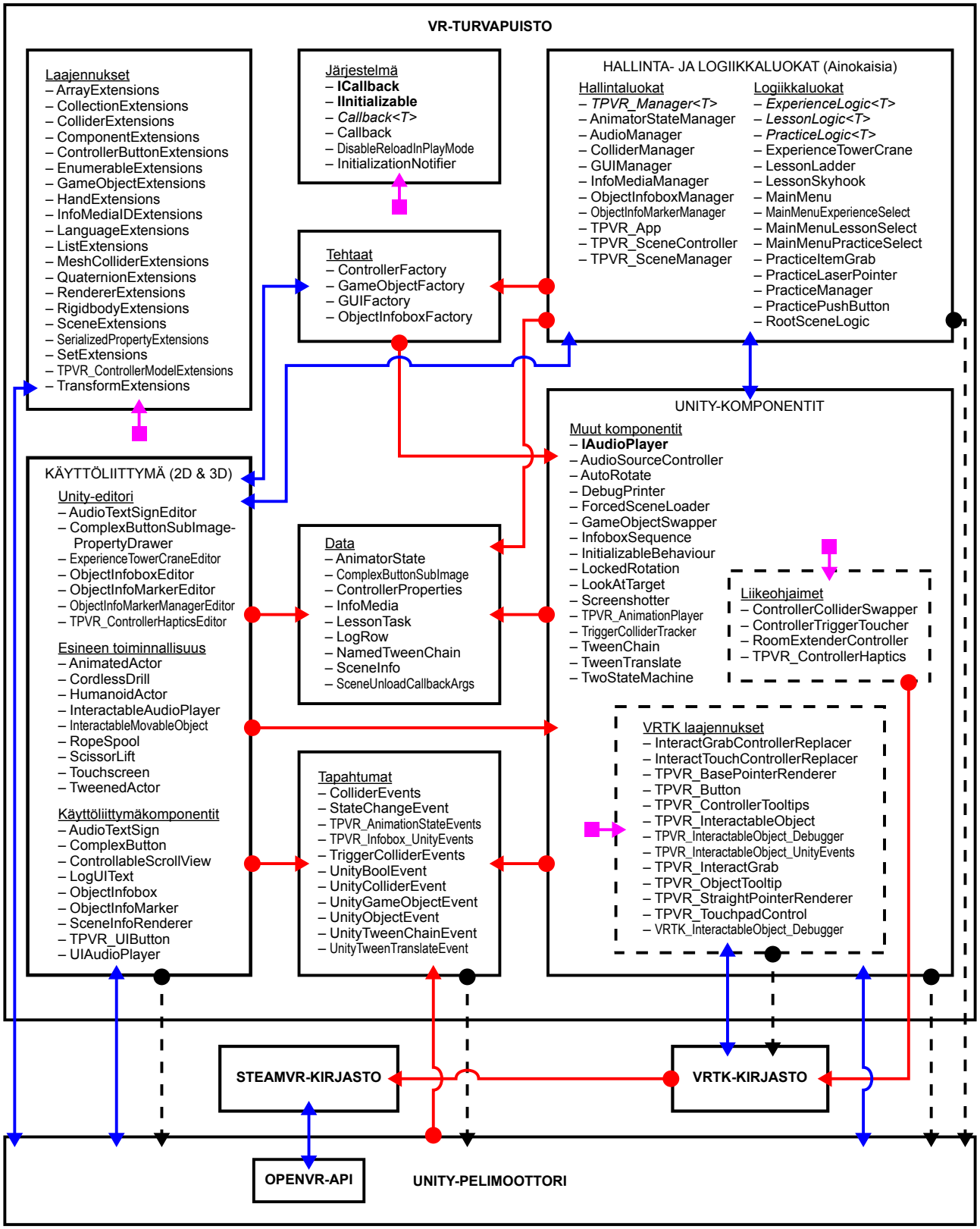
35 Execution Order of Event Functions. 2018. Verkkoaineisto. Unity Technologies. <https://docs.unity3d.com/2017.4/Documentation/Manual/ExecutionOrder.html>. 30.4.2018. Luettu 21.5.2018.

Oppimista tukeva havainto	**Maininneita yrityksiä**	**Toteutettu**	**Toteutustapa**
Mallinuket havainnollistavia	3 tai enemmän	✓	Animoidut ihmishahmot
Onnettomuuksien demonstrointi on hyvä		✓	Tapaturma-animaatiot ja -simulaatiot
Suojauksien ja suojainten käytön esittely on hyvä		✓	Koulutusrasti putoamissuojaimista
Hyvä perehdytyspaikka kaikille		~	Sovelluksen tulee olla helppokäyttöinen
Havainnolliset työnkuvaukset auttavat		X	Animoitu työtehtävän suoritusesimerkki
Kokeilumahdollisuus töiden turvallisesta suorittamisesta		X	Käyttäjä suorittaa työtehtävän liikeohjaimilla
Katolla työskentely, erilaiset kiinnitysmekanismit, turvavaljaiden käyttö, teline ja tikastyöskentely ym. koulutukset	2	✓	Koulutusrasti korkealla työskentelystä
Puiston sijainnilla ei väliä	1	✓	Sovelluksen laitteisto tuodaan yrityksen tiloihin
Virheiden etsiminen opettaa		✓	Koulutusrasteissa kaikki ei ole oikein, poikkeamia tulee huomioida
Puiston sijainnilla ei väliä	1	✓	Sovelluksen laitteisto tuodaan yrityksen tiloihin
Virheiden etsiminen opettaa		✓	Koulutusrasteissa kaikki ei ole oikein, poikkeamia tulee huomioida
Merkinnät: ✓ = kyllä, ~ = osittain, X = ei			

Oppimista hankaloittava havainto tai kehitysehdotus	**Maininneita yrityksiä**	**Toteutettu**	**Toteutustapa**
Aktivoi koulutettavia enemmän koulutuksen aikana	3 tai enemmän	✓	Interaktiiviset koulutukset
Sääolosuhteet haittaavat koulutusta		X	Erilaiset simuloidut säätilat
Kaikkia työntekijöitä ei voida kouluttaa, liian iso ryhmä	2	✓	Koulutukset ovat henkilökohtaisia, suoritetaan omaan tahtiin
Kierrokseen käytetty liian vähän aikaa, läpijuoksua		✓	Koulutukset etenevät käyttäjän määräämällä tahdilla vaiheittain
Puistossa voisi olla myös jotain elämyksellisempää	1	✓	Kehitetään elämysrasteja, joilla ei ole koulutusarvoa
Seinillä olevat tekstit eivät toimi hyvin		✓	Teoriasisältö tarjotaan äänen luettuna ja tekstinä
Merkinnät: ✓ = kyllä, X = ei			

- Virtuaalitodellisuusjärjestelmän ominaisuudet
    - Mitä ohjelmistorajapintaa järjestelmä käyttää?
    - Millä alustoilla järjestelmää voidaan käyttää?
    - Onko järjestelmä itsenäinen vai vaatiko se emolaitteen?
    - Mikä on laitteiston suositeltu suoritustehovaatimus emolaitteelle
- Virtuaalivisiirin ominaisuudet
    - Onko visiirissä kamera? Kuinka monta?
    - Mikä on näytön resoluutio?
    - Kuinka laaja näkökenttä on?
    - Onko visiirissä linssien vaakatason etäisyydensäätö?
    - Onko visiirissä nappeja? Kuinka monta?
    - Onko visiiri langaton? Kuinka kauan akku kestää yhdellä latauksella?
    - Onko visiirissä USB-portteja oheislaitteille?
- Ohjaimet
    - Tuleeko laitteiston mukana liikeohjain? Kuinka monta?
    - Onko ohjaimessa kosketuslevy tai ohjaintatti?
    - Onko ohjaimessa nappeja? Kuinka monta?
    - Millainen liikkeenseuranta ohjaimessa on?
- Ääni
    - Onko virtuaalivisiirissä sisäänrakennetut kuulokkeet tai kaiuttimet?
    - Onko visiirissä kuulokeliitäntää?
- Liikkeenseuranta
    - Pystyykö virtuaalitodellisuudessa liikkumaan?
    - Tarvitaanko liikkeenseurantaan ulkoisia laitteita?
    - Mikä on suurin etäisyys, jolla laitteisto pystyy seuraamaan visiiriä?

Mitä enemmän sovelluskehittäjät ottavat näitä ominaisuuksia huomioon sovelluksen kehityksessä, sen laajemmille markkinoille tuotetta voidaan viedä. Linssien vaakatason etäisyyden säätö on tärkeää, koska käyttäjän tulee katsoa virtuaalivisiirin näyttöä linssin läpi mahdollisimman läheltä sen keskikohtaa. Kuva näyttää sumealta, jos käyttäjän pupillien välinen etäisyys (engl. interpupillary distance, kirjainlyhenne IDP) ei vastaa linssin keskipisteiden välistä etäisyyttä tarvittavan tarkasti.



Selitykset

- Vasen pääty perii oikean päädyn luokkia
- Vasen pääty käyttää oikean päädyn luokkia/olioita
- Neliön alla oleva ja sen sisältämät laatikot käyttävät oikean päädyn luokkia/olioita
- Molemmat päädyt käyttävät toistensa luokkia/olioita

LIHAVOITU VERSAALITEKSTI : Suuren kokonaisuuden nimi
VERSAALITEKSTI : Kokonaisuuden nimi
Alleviivattu teksti : Pienen kokonaisuuden nimi
Lihavoitu teksti : Rajapinnan nimi
Kursivoitu teksti : Abstraktin luokan nimi
Normaali tai kapea teksti : Luokan nimi

Kategoria	Tavanomainen hiiri	Toimintoja	HTC Viven liikeohjain	Toimintoja	Samsung Gear VR:n liikeohjain	Toimintoja
Liike& kierto	liike x-akselilla	2	kierto x-akselilla	2	kierto x-akselilla	2
	liike y-akselilla	2	kierto y-akselilla	2	kierto y-akselilla	2
			kierto z-akselilla	2	kierto z-akselilla	2
			liike x-akselilla	2		
			liike y-akselilla	2		
			liike z-akselilla	2		
Napit	vasen nappi, pohjaan	1	liipaisin, pohjaan	1	liipaisin, pohjaan	1
	oikea nappi, pohjaan	1	liipaisin, liukuva	1	tuntolevy, pohjaan	1
	rulla, pohjaan	1	tuntolevy, pohjaan	1	tuntolevy, kosketus	1
			tuntolevy, kosketus	1	tuntolevy, pohjaan ylhäällä	1
			tuntolevy, pohjaan ylhäällä	1	tuntolevy, pohjaan alhaalla	1
			tuntolevy, pohjaan alhaalla	1	tuntolevy, pohjaan vasemmalla	1
			tuntolevy, pohjaan vasemmalla	1	tuntolevy, pohjaan oikealla	1
			tuntolevy, pohjaan oikealla	1	edellinen-nappi, pohjaan	1
			valikkonappi, pohjaan	1	kotinappi, pohjaan	1
			järjestelmänappi, pohjaan	1	äänenvoimakuus ylös, pohjaan	1
			vasen tai oikea puristusnappi, pohjaan	1	äänenvoimakuus alas, pohjaan	1

Vieritys	rullaus eteen & taakse	2	tuntolevyn x-akseli	2	tuntolevyn x-akseli	2
			tuntolevyn y-akseli	2	tuntolevyn y-akseli	2
Toimintojen lukumäärä		**9**		**27**		**21**

Liite poistettu julkisesta versiosta salaisena.

```
public delegate void InitializationEventHandler(IInitializable source);

public interface IInitializable {
     bool IsInitialized { get; }
     event InitializationEventHandler OnInitialized;
     void Initialize(bool setInitialized = true);
}

public abstract class InitializableBehaviour : MonoBehaviour, IInitializable {
     public bool IsInitialized { get; private set; }
     public event InitializationEventHandler OnInitialized;

     protected virtual void Start() {
          Initialize();
     }

     public abstract void Initialize(bool setInitialized = true);

     protected void SetInitialized() {
          if (!IsInitialized) {
               IsInitialized = true;
               OnInitialized?.Invoke(this);
          }
     }
}

internal class InitializationNotifier {
     public List<IInitializable> UninitializedBehaviours { get; private set; }
= new List<IInitializable>();
     private ICallback _callback;

     public InitializationNotifier(ICallback callback, List<IInitializable>
behaviours) {
          _callback = callback;

          behaviours.ForEach(behaviour => {
               behaviour.OnInitialized += BehaviourInitialized;

               if (!behaviour.IsInitialized) {
                    UninitializedBehaviours.Add(behaviour);
               }
          });

          AttemptCallback();
     }

     public InitializationNotifier(ICallback callback, params IInitializable[]
behaviours) : this(callback, new List<IInitializable>(behaviours)) { }

     private void BehaviourInitialized(IInitializable behaviour) {
          UninitializedBehaviours.Remove(behaviour);
          behaviour.OnInitialized -= BehaviourInitialized;

          AttemptCallback();
     }

     private void AttemptCallback() {
          if (UninitializedBehaviours.Count() < 1) {
               _callback.Invoke();
          }
     }
}
```

```
public class VRTK_TrackedController : MonoBehaviour {
     // VRTKTrackedControllerEventArgs on struct-tyyppinen, eikä normaalisti
salli null-arvoa.
     // Tehdään siitä nullable-tyyppinen ?-lyhenteellä (kääntyy tyypiksi
Nullable<VRTKTrackedControllerEventArgs>) ja asetetaan oletukseksi null-arvo.
     private VRTKTrackedControllerEventArgs? modelAvailableArgs = null;

     // Oikea tapahtuma on yksityinen, jotta tilaajarekisteröinnit menee
julkisen ominaisuuden kautta.
     private event VRTKTrackedControllerEventHandler
RealControllerModelAvailable;

     // Julkinen ominaisuus, joka rekisteröi tilaajan tapahtumaan.
     public event VRTKTrackedControllerEventHandler ControllerModelAvailable {
          add {
               // Lisätään tilaaja.
               RealControllerModelAvailable += value;

               // Jos ohjain on alustunut.
               if (modelAvailableArgs != null) {
                    // Laukaistaan tapahtuma heti.
                    RealControllerModelAvailable(this,
(VRTKTrackedControllerEventArgs)modelAvailableArgs);
               }
          }

          remove {
               // Tilaajan poisto.
               lock (RealControllerModelAvailable) {
                    RealControllerModelAvailable -= value;
               }
          }
     }

     protected virtual void OnEnable() {
          // OnControllerModelAvailable-metodia kutsutaan pitkän
metodikutsuketjun päätteksi, joka alkaa tästä.
     }

     public virtual void
OnControllerModelAvailable(VRTKTrackedControllerEventArgs e) {
          // Tallenna alustumistiedot muistiin tulevaa käyttöä varten.
          modelAvailableArgs = e;

          // Jos tapahtumassa on tilaajia.
          if (RealControllerModelAvailable != null) {
               // Laukaise tapahtuma.
               RealControllerModelAvailable(this, e);
          }
     }
}
```

Aihealue	Havaittu ongelma	Sivunumero	Korjattu	Korjaustapa
Turvaorsi-koulutusrasti	Infomerkkikylttiä ei voi käyttää	15	✓	Infomerkkikyltti ja sen osoittama kohde molemmat avaavat teoriasisällön.
	A- ja B-nappeja ei voi käyttää laserilla	15	✓	Napit korvattu infonäytöllä, joka toimii laserin avulla.
	Putoamisdemonstraatio alkoi käyttäjän huomaamatta	15	~	Putoamisdemonstraation viivettä lisätty. Audiovisuaalinen tehoste demonstraation alkamisesta puuttuu yhä.
	Putoamisdemonstraation oikeellisuutta ei pääse itse tarkastelemaan	19	X	Käyttäjän tulee päästä viereiseen rakennelmaan tutkimaan turvaorsia itse.
Korotettu työskentely -koulutusrasti	Liikeohjaimen toiminnot ovat epäselvät	17	✓	Kehitetty erilliset harjoitusrastit, joissa kokeillaan liikeohjaimen toimintoja yksi kerrallaan.
	Nopeutettu liikkuminen on haastavaa	17	X	Liikkumistapa tulee muuttaa tulevaisuudessa.
	Esineisiin tarttuminen on haastavaa	17	X	Ongelmaa vaatii lisätutkimusta.
	Kellon soittaminen ei onnistu	17	✓	Kelloa voi käyttää laserilla kosketuksen lisäksi.
	Työtoveri ehdottaa epäsuorasta tikkaiden käyttöä (joita ei saa käyttää)	17	✓	Viittaus tikkaisiin poistettu dialogista.
	Tikkaiden asettelu seinää vasten on haastavaa	17	X	Tikkaiden käsittelyn fysiikat vaativat jatkokehitystä.
	Saksinostimen toiminnot ovat epäselvät	17	X	A- ja B-toimintonappien grafiikka korvattu toimintoa vastaavalla kuvakkeella.
	Saksinostin liikkuu itsestään	18	X	Saksinostimen toiminnot vaativat lisää suunnittelua.
	Saksinostimen käyttö ilman tikkaiden huomiointia johtaa rastin epäonnistumiseen	18	✓	Tikkaiden aloituspaikka siirretty pois saksinostimen reitiltä.

	Saksinostimen tukia ei huomattu laskea	18	X	Saksinostimen toiminnot vaativat lisää suunnittelua.
	Työtoverin puutteellista varustusta ei huomioitu	18	X	Koulutusrastin tehtävänantoa tulee parantaa.
	Rasti on liian haastava, toiminnallisuudet ovat piilossa	19	e.h.	Koulutusrasti on lopputesti, sen on tarkoitus olla haastava. Rastia edeltävät koulutukset puuttuvat yhä.
	Rastin alun ohjetekstit ovat liian pitkät	20	~	Tekstit korvataan interaktiivisilla harjoitusrasteilla myöhemmin.
Torninosturi-elämysrasti	Voimakas taustaääni koulutusrastissa haittaa yhteydenpitoa kouluttajien kanssa	18	✓	Rastiin toteutettu liikeohjaimen toiminto, joka vaimentaa taustaäänet.
	Korkealla oleminen tuntui epämukavalta	20	X	Ennen rastin lataamista käyttäjältä varmistetaan rastin lataus ja varoitetaan ympäristön todentuntuisuudesta ja korkealla olemisesta.
VR-koulutus	Virtuaalinen ympäristö tuntuu liian vieraalta, haittaa opiskelua	19	~	Kehitetty erilliset harjoitusrastit. Käyttäjän tulisi suorittaa harjoitusrastit ennen koulutuksia, kehotus puuttuu käyttöliittymästä.
	Turhautuminen jos ei tiedä mitä tulee tehdä seuraavaksi	19	X	Kehitetään järjestelmä, joka antaa vinkkejä koulutuksen tilan perusteella.
	Käyttäjä ei uskalla tai ymmärrä tutkia toimintoja, joista ei ole erikseen kerrottu	20	X	Käyttäjän koulutuksen tulee alkaa yksinkertaisista rasteista, jotta hän pystyy tutustumaan eri koulutusten toimintoihin ennen haastavampia koulutuksia.
Merkinnät: ✓ = kyllä, ~ = osittain, X = ei, e. h. = ei huomioida				

Aihealue	**Sovelluksen ominaisuus / oppimista tukeva tai hankaloittava havainto / ongelma sovelluksessa**	**Toteu-tettu**	**Ominaisuuden toteutustapa / ongelman korjaustapa**
Korotettu työskentely -koulutusrasti	Kellon soittaminen ei onnistu	✓	Kelloa voi käyttää laserilla kosketuksen lisäksi.
	Liikeohjaimen toiminnot ovat epäselvät	✓	Kehitetty erilliset harjoitusrastit, joissa kokeillaan liikeohjaimen toimintoja yksi kerrallaan.
	Saksinostimen käyttö ilman tikkaiden huomiointia johtaa rastin epäonnistumiseen	✓	Tikkaiden aloituspaikka siirretty pois saksinostimen reitiltä.
	Työtoveri ehdottaa epäsuorasta tikkaiden käyttöä (joita ei saa käyttää)	✓	Viittaus tikkaisiin poistettu dialogista.
	Rastin alun ohjetekstit ovat liian pitkät	~	Tekstit korvataan interaktiivisilla harjoitusrasteilla myöhemmin.
	Esineisiin tarttuminen on haastavaa	X	Ongelmaa vaatii lisätutkimusta.
	Nopeutettu liikkuminen on haastavaa	X	Liikkumistapa tulee muuttaa tulevaisuudessa.
	Saksinostimen toiminnot ovat epäselvät	X	A- ja B-toimintonappien grafiikka korvattu toimintoa vastaavalla kuvakkeella.
	Saksinostimen tukia ei huomattu laskea	X	Saksinostimen toiminnot vaativat lisää suunnittelua.
	Saksinostin liikkuu itsestään	X	Saksinostimen toiminnot vaativat lisää suunnittelua.
	Tikkaiden asettelu seinää vasten on haastavaa	X	Tikkaiden käsittelyn fysiikat vaativat jatkokehitystä.
	Työtoverin puutteellista varustusta ei huomioitu	X	Koulutusrastin tehtävänantoa tulee parantaa.
	Rasti on liian haastava, toiminnallisuudet ovat piilossa	e.h.	Koulutusrasti on lopputesti, sen on tarkoitus olla haastava. Rastia edeltävät koulutukset puuttuvat yhä.

Koulutusrastit	Katolla työskentely, erilaiset kiinnitysmekanismit, turvavaljaiden käyttö, teline ja tikastyöskentely ym. koulutukset	✓	Koulutusrasti korkealla työskentelystä
	Puistossa voisi olla myös jotain elämyksellisempää	✓	Kehitetään elämysrasteja, joilla ei ole koulutusarvoa
	Suojauksien ja suojainten käytön esittely on hyvä	✓	Koulutusrasti putoamissuojaimista
	Alkusammutusrastit puuttuivat	X	Koulutusrasti alkusammutuksesta
	Betonipumppuletkun räjähdystä kuvaava rasti	X	Koulutusrasti betoniauton kanssa työskentelystä
	Elementtirakentamisen, taakan kiinnittämisen ja noston koulutus puuttuu	X	Koulutusrasti elementin nostosta
	Kaivantojen sortumisriskin ja suojaamisen koulutus	X	Koulutusrasti kaivantojen suojaamisesta
	Kiire tulee ottaa huomioon työmaalla	X	Aikarajoitetut koulutusrastit
	Koulutus järjestyksestä ja siisteydestä	X	Koulutusrasti työmaan siisteydestä
	Kuopat ja erilaiset esteet työmaalla	X	Koulutusrasti kuopista ja esteistä työmaalla
	Monikielisen kommunikaation huomiointi työmaalla	X	Koulutusrasti monikielisestä kommunikaatiosta
	Nosturityön demonstraatio ja käsimerkkien opettelun koulutus	X	Käytännön rasti torninosturin ohjauksesta
	Pölyltä suojautumisen koulutusrasti	X	Koulutusrasti pölynpoistosta
	Silmien suojauksen koulutusrasti	X	Koulutusrasti henkilösuojaimista
	Työkoneisiin liittyvien vaarojen (raskas liikenne) koulutus	X	Koulutusrasti työmaalla liikkumisesta

	Vaikuttaa koti- ja vapaa-ajan tapaturmiin	X	Koti- ja vapaa-ajan koulutussisältö
Torninosturi -elämysrasti	Voimakas taustaääni koulutusrastissa haittaa yhteydenpitoa kouluttajien kanssa	✓	Rastiin toteutettu liikeohjaimen toiminto, joka vaimentaa taustaäänet.
	Korkealla oleminen tuntui epämukavalta	X	Ennen rastin lataamista käyttäjältä varmistetaan rastin lataus ja varoitetaan ympäristön todentuntuisuudesta ja korkealla olemisesta.
Turvaorsi -koulutusrasti	A- ja B-nappeja ei voi käyttää laserilla	✓	Napit korvattu infonäytöllä, joka toimii laserin avulla.
	Infomerkkikylttiä ei voi käyttää	✓	Infomerkkikyltti ja sen osoittama kohde molemmat avaavat teoriasisällön.
	Putoamisdemonstraatio alkoi käyttäjän huomaamatta	~	Putoamisdemonstraation viivettä lisätty. Audiovisuaalinen tehoste demonstraation alkamisesta puuttuu yhä.
	Putoamisdemonstraation oikeellisuutta ei pääse itse tarkastelemaan	X	Käyttäjän tulee päästä viereiseen rakennelmaan tutkimaan turvaorsia itse.
VR-koulutus	Aktivoi koulutettavia enemmän koulutuksen aikana	✓	Interaktiiviset koulutukset
	Kaikkia työntekijöitä ei voida kouluttaa, liian iso ryhmä	✓	Koulutukset ovat henkilökohtaisia, suoritetaan omaan tahtiin
	Kierrokseen käytetty liian vähän aikaa, läpijuoksua	✓	Koulutukset etenevät käyttäjän määräämällä tahdilla vaiheittain
	Mallinuket havainnollistavia	✓	Animoidut ihmishahmot
	Onnettomuuksien demonstrointi on hyvä	✓	Tapaturma-animaatiot ja -simulaatiot
	Puiston sijainnilla ei väliä	✓	Sovelluksen laitteisto tuodaan yrityksen tiloihin
	Seinillä olevat tekstit eivät toimi hyvin	✓	Teoriasisältö tarjotaan äänen luettuna ja tekstinä

	Virheiden etsiminen opettaa	✓	Koulutusrasteissa kaikki ei ole oikein, poikkeamia tulee huomioida
	Hyvä perehdytyspaikka kaikille	~	Sovelluksen tulee olla helppokäyttöinen
	Virtuaalinen ympäristö tuntuu liian vieraalta, haittaa opiskelua	~	Kehitetty erilliset harjoitusrastit. Käyttäjän tulisi suorittaa harjoitusrastit ennen koulutuksia, kehotus puuttuu käyttöliittymästä.
	Havainnolliset työnkuvaukset auttavat	X	Animoitu työtehtävän suoritusesimerkki
	Kokeilumahdollisuus töiden turvallisesta suorittamisesta	X	Käyttäjä suorittaa työtehtävän liikeohjaimilla
	Käyttäjä ei uskalla tai ymmärrä tutkia toimintoja, joista ei ole erikseen kerrottu	X	Käyttäjän koulutuksen tulee alkaa yksinkertaisista rasteista, jotta hän pystyy tutustumaan eri koulutusten toimintoihin ennen haastavampia koulutuksia.
	Sääolosuhteet haittaavat koulutusta	X	Erilaiset simuloidut säätilat
	Tarinat mukavia	X	Koulutusmoduulien sisällöissä hyödynnetään tarinamaista jatkumoa
	Turhautuminen jos ei tiedä mitä tulee tehdä seuraavaksi	X	Kehitetään järjestelmä, joka antaa vinkkejä koulutuksen tilan perusteella.
	Uusien asioiden esittely hyödyllistä	X	Koulutussisältöä päivitetään tulevaisuudessa
	Uusintakierros vain, jos jotain muuttuu puistossa	X	Koulutussisältöjen päivityksestä ilmoitetaan asiakkaille
Turvapuisto-koulutus	Asiasisältö on suuri	✓	Koulutukset pilkottu pienempiin osiin
	Koulutukset ovat pitkiä	✓	Koulutukset ovat lyhyempiä kuin turvapuistossa
Merkinnät: ✓ = kyllä, ~ = osittain, X = ei, e. h. = ei huomioida			