Puhekäyttöliittymät videopeleissä

Turun yliopisto
Tietotekniikan laitos
LuK-tutkielma
Tietojenkäsittelytiede
Joulukuu 2024
Heikki Viljanen

TURUN YLIOPISTO
Tietotekniikan laitos

Heikki Viljanen: Puhekäyttöliittymät videopeleissä

LuK-tutkielma, 35 s., 3 liites.
Tietojenkäsittelytiede
Joulukuu 2024

Puhekäyttöliittymä voi parantaa pelin saavutettavuutta ja parantaa pelin immersiota. Tässä työssä suoritetun haun perusteella löydettiin 22 vertaisarvioitua artikkelia puhekäyttöliittymistä videopelien kontekstissa. Lisäksi niitä on haun perusteella tutkittu muissa konteksteissa noin 2700 artikkelin laajuudessa. Tämän tutkielman tarkoitus on perehtyä videopeleissä esiintyviin puhekäyttöliittymiin, sekä puhekäyttöliittymien toteutustapoihin kirjallisuuskatsaukseen perustuen. Esimerkkipelejä haetaan myös vapaalla internet haulla.
Puhekäyttöliittymä on useimmiten toteutettu vain osittaisena toiminnallisuutena, jolloin sen rinnalle vaaditaan myös jokin fyysinen sisääntulolaite, kuten tietokoneen hiiri ja näppäimistö tai peliohjain. Osittainen puhekäyttöliittymä on esimerkiksi peleissä SOCOM U.S. Navy SEALs ja Tom Clancy's Rainbow Six 3. Pelkän puheen tai äänen avulla ohjattavia käyttöliittymiä löytyi kirjallisuuskatsauksessa esiintyneistä prototyyppipeleistä sekä pelistä Tom Clancy's Endwar, jossa pelaaminen on mahdollista täysin puheohjatusti, täysin ilman puheohjausta tai sekä puheohjauksella että jollakin toisella sisääntulolla.
Kaupallisesti julkaistujen pelien puhekäyttöliittymien arkkitehtuurista ei kirjallisuuskatsauksessa löytynyt tarkkaa määritelmää. Tässä tutkielmassa kuitenkin tehtiin prototyyppi-implementaatio Unityn omaan puheentunnistus-API:in pohjautuvasta määritelmästä, jota voidaan hyödyntää tulevissa peliprojekteissa. Sen sijaan kirjallisuuskatsauksen artikkelien tutkimuksissa kehitetyissä prototyyppipeleissä käytetään jotakin tekoälymallia puheentunnistukseen ohjelmointirajapintojen välityksellä, tai varta vasten kehitettyä tekoälymallia. Syväoppivia konvoluutioneuroverkkoja hyödyntävät ratkaisut kykenevät tunnistamaan puhekomentoja tehokkaasti.
Jatkotutkimussuuntana esitetään kaupallisten pelien puhekäyttöliittymien arkkitehtuurien selvittäminen esimerkiksi kehittäjien haastattelujen kautta. Toteutustapoihin liittyy varmasti yhtiösalaisuuksia, joita ei todennäköisesti ole mahdollista saada. Tässä kuitenkin olisi mahdollisuuksia tarkempia jatkotutkimuksia varten.

Asiasanat: puhekäyttöliittymä, videopeli, puheohjaus, voice user interface, speech user interface

Sisällys

1 Johdanto

Videopelit ja niiden pelaaminen ovat yleistyneet huomattavasti. Vuosien 1991 ja 2017 välisenä 25 vuoden aikana digitaalisten pelien pelaaminen on nelinkertaistunut koko väestön tasolla [1]. Maailmanlaajuisesti voidaan sanoa, että vuosittainen pelaajien määrä on kasvanut pienestä markkinaraosta yli 3 miljardiin pelaajaan [2].

Jotta minkään pelin pelaaminen olisi mahdollista, tarvitsee se jonkinlaisen käyttöliittymän. Hyvin toteutettu käyttöliittymä on erittäin tärkeä osa ihmisen ja tietokoneen välistä vuorovaikutusta, *HCI* (engl. human-computer interaction) [3]. Erilaisia käyttöliittymiä on useita. Tavanomaisia käyttöliittymiä ovat esimerkiksi komentorivikäyttöliittymät, eli *CLI*:t (engl. command-line interface) ja erityisesti erilaiset graafiset käyttöliittymät, *GUI*:t (engl. graphical user interface). Graafista käyttöliittymää käytetään osoitinlaitteiden, eli *HID*:ien (engl. human interface device), sekä erilaisten näyttölaitteiden avulla. Osoitinlaitteita ovat esimerkiksi tietokoneen hiiri, näppäimistö ja erilaiset peliohjaimet sekä kosketusnäytöt. Osoitinlaitteet ovat siis GUI:n sisääntuloja (engl. input). GUI:lla on oleellisena osana myös ulostulo (engl. output), joka on näyttölaite, kuten televisio tai tietokoneen näyttö. Viime aikoina ovat yleistyneet erilaiset luonnolliset käyttöliittymät, eli *NUI*:t (engl. natural user interface), joissa inputlaitteita käytetään esimerkiksi puheen, kosketuksen tai eleiden kautta [4].

Käyttöliittymän voi kuitenkin toteuttaa mitä erilaisimmin keinoin, esimerkiksi Nintendo Switch -konsolille on tarjolla erilaisia pahvista rakennettavia lisäosia tuo-

temerkillä *LABO*[1], joilla konsolia voi ohjata. LABO siis muuttaa esimerkiksi sitä, miten inputit syötetään laitteeseen, ja luo siten hyvin erilaisen käyttäjäkokemuksen. On kuitenkin paljon sellaisia henkilöitä, jotka eivät fyysisten rajoitteidensa takia kykene käyttämään mitään tavanomaista käyttöliittymää, tai niiden käyttömahdollisuudet ovat hyvin rajallisia. Eräs tapa tuoda pelit laajemman yleisön ulottuville voisi olla kehittää pelejä, joita voi ohjata esimerkiksi omalla äänellään. Kun peliä ohjataan äänen avulla, inputlaitteena toimii mikrofoni, joka muuttaa äänen aiheuttamat ilmanpaineen vaihtelut sähköiseksi signaaliksi. Pelkän äänen avulla ohjaaminen tuo runsaasti rajoitteita siihen, millainen videopeli on mahdollista toteuttaa. Olisi esimerkiksi hyvin vaikeaa toteuttaa jokin nopeatempoinen autopeli tai ensimmäisen persoonan ammuntapeli, eli *FPS* (engl. first person shooter) pelkän ääniohjauksen avulla. Edes osittainenkin puhekäyttöliittymä voisi kuitenkin tehdä pelikokemuksesta intiimimmän, ja tällainen lähestymistapa oli esimerkiksi vuonna 2002 Playstation 2 -konsolille julkaistussa, amerikkalaisen Zipper Interactiven kehittämässä SOCOM U.S. Navy SEALS pelissä. SOCOMissa ohjattiin 4 erikoisjoukkosotilaan ryhmää, jossa tekoälyn ohjaamia ryhmän jäseniä oli mahdollista komentaa pelin mukana tulleen kuulokemikrofonin välityksellä [5].

Yleisesti ottaen puheohjatut käyttöliittymät, *VUI*:t (engl. voice user interface), ovat yleistyneet viimeisen kymmenen vuoden aikana järjestelmissä, kuten älykaiuttimet (enlg. smart speakers) ja *IoT*-laitteiden (engl. internet of things) toteutuksissa. Toisin kuten esimerkiksi GUI:den tapauksessa, VUI:den kehitykseen ja suunnitteluun ei kuitenkaan vielä ole vakinaistunutta ohjenuoraa, joka taas voi johtaa käytettävyysongelmiiin, koska ne kehitetään usein erillisinä toteutuksina ilman yleistä johdonmukaisuutta. [6]

Videopelit ovat mielenkiintoinen tutkimuskohde niiden viihdearvon ja runsaan vuorovaikutuksen johdosta. Tässä tutkielmassa tutkitaan minkälaisia puheohjattu-

[1] https://www.nintendo.com/au/games/nintendo-switch/nintendo-labo-variety-kit/

jen käyttöliittymien toteutuksia ja toteutustapoja on videopeleissä. Jonkinlainen puheohjaus löytyy monestakin nykyaikaisesta pelistä, mutta HCI-alan tutkimusta pelien puhekäyttöliittymien käytettävyydestä ja niiden suunnittelusta ja rakenteesta on vain vähän [7]. Tähän haetaan täydennystä kahden tutkimuskysymyksen kautta:

1. Tutkimuskysymys 1 (TK1): Minkälaisia puhekäyttöliittymiä videopeleissä on esiintynyt tai tutkittu?

2. Tutkimuskysymys 2 (TK2): Miten puhekäyttöliittymät on toteutettu?

Tutkielma suoritettiin kirjallisuuskatsauksena. Tiedonlähteitä haettiin tietokannoista IEEE Xplore, Web of Science, ACM Digital library, Google Scholar ja arXiv. IEEE Xplore ja ACM Digital library valikoituivat mukaan, sillä ne liittyvät keskeisesti tietojenkäsittelytieteen alaan. Web of science valittiin, sillä se sisältää tieteellisiä julkaisuja monialaisesti. Materiaalia ei ole paljon, mutta ala on kehittynyt huimasti ja siksi on mahdollista, että löytyy vertaisarvioimattomia tuoreita julkaisuja esimerkiksi *NLP*:n alalta (engl. natural language processing), jotka löytyvät vain ArXivista. Google Scholar valittiin hakutietokannaksi, sillä se tuottaa tyypillisesti paljon hakutuloksia, joskin näiden indeksointi saattaa olla kyseenalaista. Hakusanoina käytetään *("voice user interface"OR "speech-based*"OR "voice ui"OR "speech user interface"OR "speech ui") AND ("video game*"OR gaming*)*. Aluksi aineiston karsintaan käytetään seuraavia kriteerejä:

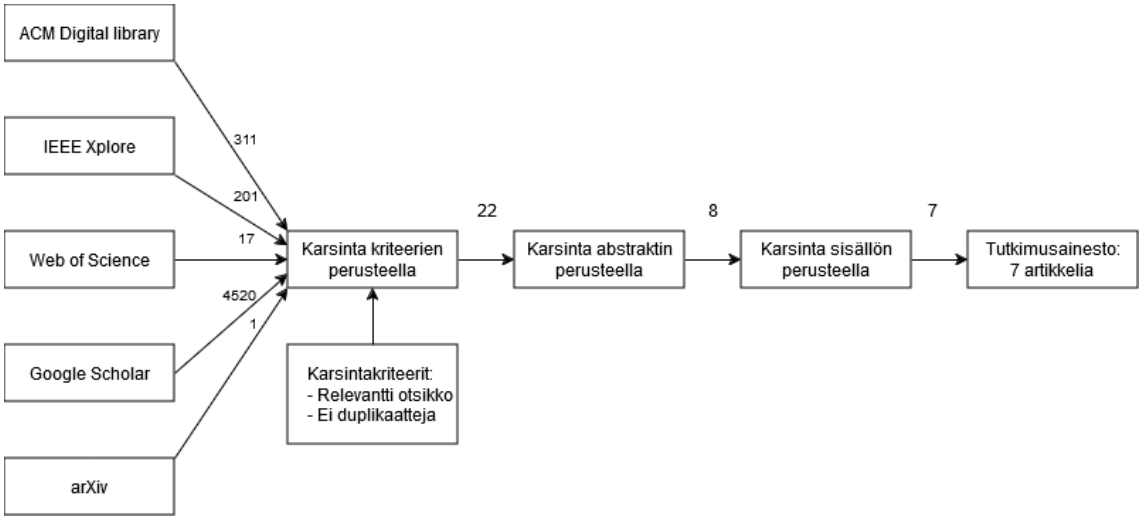
1. otsikon on oltava relevantti

2. aineistossa ei saa olla duplikaatteja

Seuraavaksi aineistosta karsittiin sopivimmat abstraktiin perustuen. Lopullinen karsinta tehtiin sisällön perusteella. Lisäksi käytettiin alan kirjallisuutta ja suoritettiin verkkohakua aiheeseen liittyvistä tieteellisistä julkaisuista. Verkkohakuna etsittiin myös esimerkkipelejä, joissa on käytetty jonkinlaista puhekäyttöliittymää. Lisätie-

toa yksittäisten pelien julkaisutiedoista haettiin verkkohaulla, jossa hakusanana oli pelin nimi.

Tiedonhaku IEEE Xplore tietokannassa tuotti 201 hakutulosta. ACM Digital library tuotti 311 hakutulosta. Web of Science tuotti 17 hakutulosta. ArXiv tuotti vain yhden hakutuloksen, kun taas Google Scholar tuotti 4520 hakutulosta. Kaikki hakutulokset käytiin aluksi läpi otsikkotasolla. Otsikkotason tarkastelulla tässä tarkoitetaan sitä, että otsikon on oltava tutkielman aiheen kannalta relevantti, eikä aineistossa saa samaan artikkelia moneen kertaan. Google Scholarin osalta tämä tarkoitti hieman karkeampaa, merkkijonohakuun perustuvaa tarkastelua hakutulosten valtavan määrän takia. Otsikkotason tarkastelun jälkeen aineistossa oli 22 vertaisarvioitua artikkelia. Tämän jälkeen kaikista jäljellä olevista artikkeleista luettiin abstrakti, jonka perusteella aineistoon jäi 8 artikkelia. Näiden sisältö tarkasteltiin pintapuolisesti ja lopulta aineistoon jäi 7 artikkelia. Tiedonhakuprosessia on mallinnettu kuvassa 1.1.



Kuva 1.1: Kaavio tiedonhakuprosessista.

Seuraavaksi kerrotaan tutkielman rakenteesta. Luvussa 2 kerrotaan tutkielman taustasta. Luvun 2 ensimmäisessä aliluvussa tarkastellaan ensin pelien toteutettuja puhekäyttöliittymiä ja millaisia puhekäyttöliittymiä on tutkittu pelien näkökulmas-

ta. Luvun 2 toisessa aliluvussa tarkastellaan, miten pelien toteutuneet puhekäyttöliittymät on toteutettu ja esimerkiksi millaisia arkkitehtuureja niissä on käytetty. Luvussa 5 koostetaan tutkielman yhteenveto ja esitetään tutkielman tuloksiin liittyvää pohdintaa ja mahdollisia jatkotutkimuskysymyksiä.

2 Puhekäyttöliittymien konsepteja (TK1)

Tässä luvussa kerrotaan tutkielman tuloksista ensimmäisen tutkimuskysymyksen pohjalta. Aliluvussa 2.1 kerrotaan VUI-pelien toteutuksista kirjallisuuskatsaukseen perustuen ja aliluvussa 2.2 vapaaseen internethakuun perustuen. Kirjallisuuskatsauksessa käytetyt artikkelit ovat taulukossa 2.1.

Taulukko 2.1: Kirjallisuuskatsauksessa käytetyt artikkelit.

Artikkeli	Vuosi
Voice Stimulated Inclusive Multiplayer Game Development with Speaker Recognition	2024
Investigating and Comparing the Perceptions of Voice Interaction in Digital Games: Opportunities for Health and Wellness Applications	2023
Player Identity Dissonance and Voice Interaction in Games	2015
"I Know What You Mean": Context-Aware Recognition to Enhance Speech-Based Games	2024
Towards Usability Heuristics for Games Utilizing Speech Recognition	2012
Playing Maze using Voice Commands	2023
"I Didn't Catch That, But I'll Try My Best : Anticipatory Error Handling in a Voice Controlled Game	2022

2.1 VUI-pelit kirjallisuuskatsauksessa

Pelin ohjaaminen äänen avulla voidaan jakaa karkeasti kahteen osa-alueeseen: verbaalisiin ja ei-verbaalisiin VUI:hin. Verbaalinen VUI pohjautuu puhuttuihin sanoihin, kun taas ei-verbaalinen VUI pohjautuu äänen voimakkuuteen ja sävelkorkeuteen. Ei-verbaaliset VUI:t suoriutuvat tunnistamisesta verbaalisia paremmin, mutta niiden toiminta on kuitenkin rajoitetumpaa [8]. Kirjallisuuskatsauksessa löydettiin useita esimerkkipelejä, joissa käytetään VUI:ta joko pelkästään, tai monimodaalisuuden kautta (engl. multimodal), jossa käytetään VUI:ta ja jotakin toista UI:ta tai inputia yhdessä.

Halosen ym. tutkimuksessa [9] analysoitiin erityisesti Ubisoftin vuonna 2008 julkaisemaa *RTS*, eli reaaliaikastrategiapeliä (engl. real-time strategy) Tom Clancy's Endwar, jossa taistelukentällä olevia yksiköitä voi ohjata ennalta määrätyillä, valikoiduilla äänikomennoilla, jotka tarvittaessa on mahdollisuus saada näkyviin GUI:lle komentovalikon avulla. Endwar antaa palautetta sekä visuaalisesti GUI:lla että äänipalautetta annetuista äänikomennoista. Pelissä käytetään ääniaktivointipainiketta (engl. push-to-talk -button) äänikomentojen antamiseen.

Buchta ym. kehittivät tutkimuksessaan Unreal Enginessä tutkimustyökalun [10], jonka avulla voi verrata GUI:n ja VUI:n eroja neljässä erilaisessa skenaariossa interaktiivisessa virtuaalisessa ympäristössä, *IVE*:ssä (engl. interactive virtual environment). Pelin skenaarioissa suoritetaan tehtäviä virtuaalisessa toimistossa ja keskustellaan Bobbyn, virtuaalisen avustajan kanssa, joka antaa tehtäviin tarvittaessa vinkkejä. Peliä ei ohjata pelkän äänen avulla vaan monimodaalisesti mikrofonin, HTC Vive virtuaalilasien, eli VR- lasien (engl. virtual reality headset) sekä Vive 2.0 -ohjainten avulla.

Hariprasad ym. [2] tutkimuksessa pyrittiin kehittämään inklusiivinen moninpeli pelaajantunnistuksella. Tutkimuksen alussa todetaan, että lähes kolmanneksella pelaajista on jokin krooninen vamma tai jotain toiminnallisia rajoitteita. Näistä

pelaajista 66 % ilmoittaa kokevansa vaikeuksia pelaamisessa, 91 % ilmoittaa rajoitteidensa vaikuttavan pelituloksiin. Lisäksi 71 % ihmisistä, joilla on vammoja tai rajoitteita ilmoittavat, että pelaisivat pelejä mieluusti - jos ne vaan olisivat saavutettavia. Peli on 2-ulotteinen Unity[1]-pelimoottorilla kehitetty tasohyppelypeli (engl. platformer), jossa on kaksi pelaajahahmoa ja käärme, joka on tekoälyn ohjaama vihollinen. Pelaajilla on käytössään yksinkertaisia komentoja, kuten ”Move left”, ”Move Right”, ”Jump” ja ”Shoot”. Peli tallentaa ensimmäisellä käynnistyskerralla ääninäytteitä, joiden avulla pelaaja myöhemmin tunnistetaan. Jos pelaajia on kaksi, kummallekin pelaajalle tallennetaan omat ääninäytteet. Pelissä kumpikin pelaaja voi antaa äänikomentoja samaan aikaan. Tässä pelissä käyttöliittymä on puhdas VUI eikä se vaadi monimodaalisuutta.

Sainadh ym. esittävät tutkimuksessaan [11] puheohjauksen toteuttamista sokkelopeliin (enlg. maze game) *CNN*:n, eli konvoluutioneuroverkon (engl. convolutional neural netrwork), avulla koulutettua teköälymallia äänikomentojen tunnistamiseen. GUI ohjaa pelaajaa aluksi antamaan sokkelon generointiin liittyvien parametrien löytämiseksi, jonka jälkeen peli generoi satunnaisen sokkelon annettujen parametrien perusteella. Pelin aloittamiseen tarvitaan kuitenkin minkä tahansa näppäimen painallus. Äänikomennotkin vaativat näppäinpainalluksen, jonka jälkeen sekunnin mittainen ääninäyte tallennetaan ja prosessoidaan komennoksi. Pelin edetessä pelaajan kulkema polku merkitään sinisillä timanttisymboleilla ja pelaajan sijaintia merkitään mustalla timanttisymbolilla. Peli loppuu pelaajan saavuttaessa maalipisteen. Peliä ohjataan äänikomennoilla ”Left”, ”Right, ”Up”, ”Down”, ”Go” ja ”Stop”. tutkimuksen tuloksissa kerrotaan, että peli tunnistaa annetut äänikomennot 94,57 % tarkkuudella. Tämä peli on käyttöliittymältään kuitenkin jossain määrin monimodaalinen, sillä myös näppäinpainalluksia vaaditaan. Tutkimuksen yhteenvedossa mainitaankin pelin rajoitteena se, että jatkuvan äänisyötteen käyttäminen ohjaami-

[1] https://unity.com/

seen ei heidän toteutuksessaan ole mahdollista.

Carvalho ym. [4] tutkivat VUI:den mahdollisuuksia erityisesti terveyteen liittyvissä peleissä ja keskittyivät tutkimuksessaan kolmeen skenaarioon:

- peli vuorovaikutuksessa pelaajan kanssa puheen kautta
- pelaaja on vuorovaikutuksessa toisen pelaajan kanssa puheen kautta
- pelaaja on vuorovaikutuksessa pelin kanssa puheen kautta

Tässä tutkielmassa ollaan luonnollisesti kiinnostuneita viimeisestä kohdasta - pelaaja on vuorovaikutuksessa pelin kanssa puheen kautta. Tutkimuksessa suoritettiin kysely, johon vastasivat sekä ihmiset että generatiivisen tekoälyn sovellukset, kuten ChatGPT. Tekoälysovellukset kuvailivat VUI:ta ilmaisuvoimaisempana sekä immersiota parantavana, kun taas ihmisvastaajat suosivat enemmän interaktiivisia valikoita. Tutkimuksessa todetaan, että VUI:t voivat mahdollistaa yksinkertaisen, kontekstitietoisen (engl. context-aware) tiedonvaihdon. VUI:t voivat siten erottua edukseen muista modaalisuuksista ja tuoda uusia pelikokemuksia erityisesti ihmisille, joilla on esimerkiksi näkörajoitteita. Tutkimuksessa mainittiin pelit Hey, you Picachu! ja Star Trek: Bridge crew, joissa on VUI. Näistä peleistä mainittiin vain, ettei pelin Hey, you Picachu! VUI:n toteutus ollut erityisen onnistunut. Hey you, Picachu! on vuonna 1998 julkaistu Nintendo-64-peli, jossa Picachun[2] kanssa voi olla vuorovaikutuksessa yksinkertaisten äänikomentojen avulla [7]. Vaikka kyseinen tutkimus ei lopulta ollutkaan aivan niin relevantti tämän tutkielman kannalta, kuin ennalta oltiin kuviteltu, löydettiin sen avulla *CHI Play '15* -konferenssin artikkeli [7] jossa esimerkkipelejä on paljon.

CHI Play 2015-konferenssissa julkaistussa artikkelissa [7] luotiin katsaus pelien puhekäyttöliittymiin. Julkaisu on jo verrattain vanha, joten viime vuosien nopea

[2]https://www.pokemon.com/fi/pokedex/pikachu

kehitys tekoälysektorilla ei vielä näy tutkimuksen tuloksissa. Tutkimuksessa mainitaan kuitenkin paljon hyviä esimerkkejä ja sen vuoksi se valikoitui lähdeartikkeliksi. Tutkimuksen mukaan eräs aikaisimmista VUI-peleistä oli vuonna 1995 pc-alustalle julkaistu sukellusvenesimulaattori Command: Aces of the deep, jossa VUI oli osa monimodaalista käyttöliittymää. Vuonna 1999 Sega Dreamcast -konsolille julkaistu Seaman oli lemmikkisimulaattori, jossa lemmikin kanssa keskusteltiin Dreamcastin mikrofonin kautta. Vuonna 2003 julkaistu Tom Clancy's Rainbow six 3 oli jo aiemmin mainitun SOCOMin kaltainen sotapeli, jossa joukkuetovereita on mahdollista käskyttää puhekomennoin. Vuonna 2003 julkaistu Lifeline on roolipeli, jossa ohjaaminen taas tapahtuu lähestulkoon pelkästään VUI:n kautta. Hyviä esimerkkejä VUI:sta, joskin monimodaalisesta, ovat vuonna 2004 julkaistu Singstar ja vuonna 2007 julkaistu Rock Band. Sekä Singstar että Rock Band ovat karaokepelejä, jotka myytiin yhdessä soveltuvan mikrofonin kanssa, ja siten VUI on näissä peleissä hyvin keskeisessä roolissa. SingStar-pelisarjaan on myöhemmin julkaistu lukuisia pelejä. Tutkimus keskittyi kuitenkin erityisesti neljään peliin: Tomb Raider: Definitive edition, Splinter Cell: Blacklist, FIFA 2014 ja Ryse: Son of Rome.

Pelissä Tomb Raider: Definitive edition käytettiin puhekäyttöliittymää monimodaalisesti esimerkiksi pelin kartan näyttämiseen tai varusteiden vaihtamiseen esimerkiksi komennoilla "show map" tai "pistol". Pelin tauottaminen onnistuu niin ikään puhekomennolla. Tutkimuksen mukaan pelaajat eivät kuitenkaan olleet tyytyväisiä Tomb Raiderin puhekomentoihin, sillä ne eivät olleet erityisen luotettavia ja napin painaminen koettiin nopeampana. Lisäksi äänikomentojen antaminen koettiin epämiellyttävänä ja jopa nolona. [7]

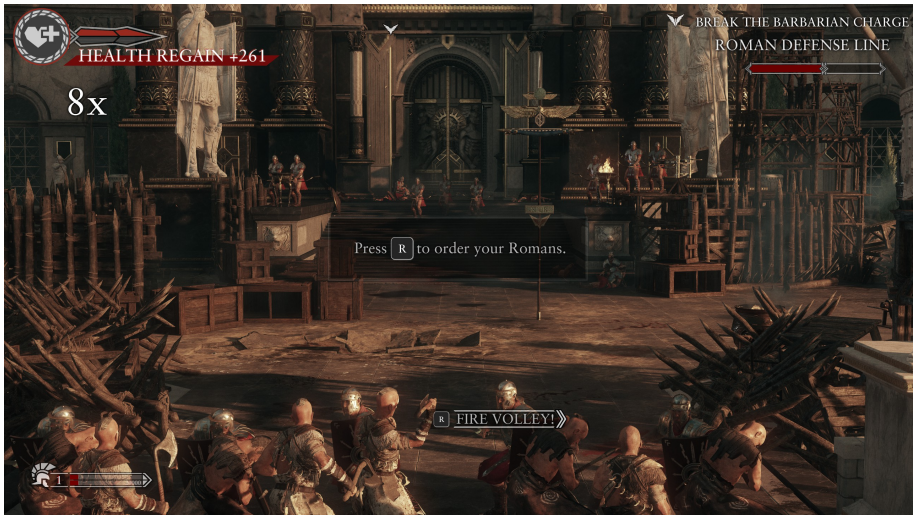
Vuonna 2013 julkaistu Splinter Cell: Blacklist käytti VUI:ta monimodaalisesti hieman samaan tapaan kuin jo aiemmin mainitt SOCOM: U.S. Navy SEALs ja Tom Clancy's Rainbow six 3. Splinter Cell-pelin idea on ohjata pelin päähenkilöä, Sam Fisheriä, läpi vihollisten miehittämien alueiden mahdollisimman huomaamatta.

Splinter Cell -pelissä puhekomentojen avulla voidaan esimerkiksi yllättää vihollisia ennalta määrätyillä puhekomennoilla, kuten sanomalla ”Hey, you”, jolloin ohjattava pelihahmo puhuu saman lauseen pelimaailmassa. Splinter Cellin lähestymistapa koettiin paljon miellyttävämpänä, kuin esimerkiksi Tomb Raiderissa. Puhekomentojen ansiosta päähenkilöön on helppo samaistua ja pelaajat olisivat toivoneet peliin enemmänkin puhekomentoja. [7]

FIFA 2014 jalkapallosimulaatiopelissä äänikomentoja pystyy käyttämään esimerkiksi vaihtopelaajan vaihtamiseen ja pelikuvioiden muuttamiseen ja taktikointiin. Esimerkiksi pelaajavaihto FIFA:ssa käynnistettäisiin yhdistämällä komento ”substitute” halutun pelaajan nimeen. Myös FIFA-pelissä VUI on monimodaalinen ja samat komennot voidaan antaa myös perinteiseen tapaan peliohjaimen kanssa. Tutkimuksen mukaan FIFA:n toteutus VUI:sta oli erityisen onnistunut ja äänikomennot koettiin toimivina sekä luontevina. [7]

Vuonna 2013 Xbox One -konsolille julkaistussa kolmannen persoonan toimintapelissä Ryse: Son of Rome, on tiettyjen pelitapahtumien yhteydessä mahdollista antaa puhekomentoja, kuten ”fire volley” tai ”soldiers move out!”. Ryse:n toteutuksessa pelihahmo ei kuitenkaan toista äänikomentoa aina samoilla sanoilla pelimaailmassa, jonka koettiin jossain tapauksissa laskevan immersiota. Rysen UI on monimodaalinen. Pelaajat olivat erityisen tyytyväisiä Ryse:n VUI:n toteutukseen ja negatiiviset kommentit liittyivät lähinnä siihen, että puheohjausta olisi voinut olla enemmänkin. [7] Kuva 2.1 esittää, miten pelin PC-versiossa puhekomennot on jopa hieman harmillisesti korvattu pelkillä painikkeilla.

Zarghamin ym. [12] tutkimuksessa mainitaan, että jo vuonna 1970 julkaistussa VoiceChess pelissä oli mahdollista pelata shakkia antamalla puhekomentoja. Esimerkkipeleinä mainitaan myös vuonna 2012 ilmestynyt Mass Effect 3 ja vuonna 2013 ilmestynyt Forza Motorsport 5, joissa oli osittainen puhekäyttöliittymä. Tässäkin tutkimuksessa kerrotaan, että pelaajat kokevat puhekäyttöliittymät usein epä-



Kuva 2.1: Kuvankaappaus pelin Ryse: Son of Rome PC-versiosta, jossa puhekomennot on korvattu painikkeilla. Kuvankaappauksen on ottanut tämän kanditutkielman kirjoittaja.

miellyttävinä ja hankalina, ja joiden käyttäminen on noloa. Tutkimuksen fokuksessa oli VUI:den virhekäyttäytyminen, joka usein on syynä pelaajien turhautumiseen. Lisäksi virheellinen puhekomennon tunnistaminen lisää virheellisen tunnistamisen riskiä seuraavissa puhekomennoissa, joka taas johtuu juuri pelaajan turhautumisesta ja ärsyyntymisestä. Tutkimuksessa kehitettiin peli ”Listen Sparky!”, joka käyttää puhekomentojen tunnistamisessa ennakoivaa virheenkäsittelyä (engl. anticipatory error handling). Ennakoivan virheenkäsittelyn idea on, että virhetilannetta ei ilmoiteta pelaajalle, vaan suoritetaan jokin lokaalisti optimoitu toiminto pelin tavoitteiden edistämiseksi. Sparky on paimenkoira, jonka tehtävänä on paimentaa lammaslaumaa pelimaailmassa vaarallisia esteitä vältellen. Puhekomennoillaan pelaaja antaa karanneen lammaslauman roolista ohjeita Sparkylle, joka puolestaan paimentaa lampaat turvallisesti takaisin maaliin. Prototyyppipelin kahdeksasta tasosta neljä ensimmäistä opettavat pelaajaa pelaamaan peliä uusien puhekomentojen avulla. Peli vaikeutuu edetessään, sillä tasosta 2 alkaen mukana susi, jota ei saa päästää lähelle lampaita. Pelin puhekomennot vaativat aktivointipainikkeen painamista. Aktivointipainikkeen ollessa pohjassa peli tallentaa ääninäytteen ja yrittää prosessoida sen

komennoksi. Pelin puhekomennot ovat ”Walk towards”, ”Flank Left”, ”Flank Right”, ”Back” ja ”Bark at wolf”. ”Walk towards” -komennolla koira kävelee kohti lampaita ja ajaa niitä omaan kulkusuuntaansa. ”Flank Left” ja ”Flank Right” -komennoilla koira kiertää lampaiden vasemmalle tai oikealle puolelle ohjaten niitä kääntymään vastakkaiseen suuntaan. ”Back” -komennolla koira palaa takaisin aloituspaikkaansa. ”Bark at wolf” -komennolla koira liikkuu sutta kohti ja haukkuu, jolloin susi jähmettyy aloilleen joksikin aikaa. Puhekomentoja ei ole pakko antaa kirjaimellisesti, vaan peli saattaa tunnistaa esimerkiksi komennon ”Right side!” komennoksi ”Flank Right”. Ennakoiva virheenkäsittely yrittää virhetunnistuksen sattuessa pelaajan tiedostamatta arvata tilanteeseen sopivan komennon, kuitenkin priorisoiden komentoja siten, ettei susi pääse syömään lampaita.

Zarghamin ym. tutkimuksessa [8] tutkittiin uutta, kontekstitietoista puheentunnistusta. Kontekstitietoinen puheentunnistus hyödyntää pelimaailman ympäristöä ja mahdollisia toimintoja lisäinformaationa, jotta puheentunnistus pystyisi tarkemmin päättelemään pelaajan antaman puhekomennon. Tutkimuksessa kehitettiin VUI-peli Escape the echo. Escape the echo on pakohuonepeli, jossa suljetussa huoneessa olevaa Sophie-nimistä hahmoa ohjataan ratkaisemaan pulmia puhekomentojen avulla. Pelissä Sophielle annetaan komentoja Sophielle annetun kommunikointivälineen välityksellä ja pelaaja näkee pelimaailman Sophien videokameran kautta. Pelin UI on monimodaalinen, sillä se vaatii hiiren käyttöä. Hiiren avulla pelaaja voi katsella ympärilleen, mutta Sophien liikuttaminen tai pelimaailman esineiden tutkiminen vaativat puhekomentojen käyttöä. Pelin puhekomennot ovat ennalta määriteltyjä ja siten niitä on rajallinen määrä. Tutkimuksessa kontekstitietoista puheentunnistusta verrattiin tavanomaiseen puheentunnistukseen. Tutkimuksen mukaan Kontekstitietoinen puheentunnistus voisi parantaa puheentunnistuksen tarkkuutta.

Kirjallisuuskatsauksessa löydettiin useita esimerkkipelejä, joissa on jonkinlainen VUI. Osa peleistä tutkimustarkoituksessa kehitettyjä prototyyppipelejä ja osa kau-

pallisesti julkaistuja pelejä. Tutkielmassa esitetyt kaupallisesti julkaistut pelit vanhimmasta uusimpaan ovat taulukossa 2.2.

Taulukko 2.2: Kaupallisia pelejä, joissa on VUI-ominaisuus.

Pelin nimi	Vuosi	Julkaisija	Alusta	VUI
Command: Aces of the deep	1995	Sierra On-Line	PC	verbaalinen
Hey you, Picachu!	1998	Nintendo	Nintendo 64	verbaalinen
Seaman	1999	Vivarium	Dreamcast	verbaalinen
SOCOM: U.S. Navy SEALs	2002	Sony Computer Entertainment	PS2	verbaalinen
Tom Clancy's Rainbow six 3	2003	Ubisoft	PC	verbaalinen
Lifeline	2003	Sony Computer Entertainment	PS2	verbaalinen
Singstar	2004	Sony Computer Entertainment	PS2	ei tiedossa
Rock Band	2007	MTV Games	useita	verbaalinen
Tom Clancy's Endwar	2008	Ubisoft	useita	verbaalinen
FIFA 2014	2013	EA Sports	Xbox One	verbaalinen
Ryse: Son of Rome	2013	Crytec	Xbox One	verbaalinen
Splinter Cell: Blacklist	2013	Ubisoft	Xbox One	verbaalinen
Tomb Raider: Definitive edition	2014	Square Enix Europe	useita	verbaalinen
Star Trek Bridge Crew	2017	Ubisoft	useita	verbaalinen

2.2 Vapaa internethaku

Esimerkkipelejä etsittiin myös vapaana verkkohakuna Google Search -hakukoneella useilla eri hakusanoilla. Esimerkiksi hakusanoilla "games with voice user interface" löytyi eräs kiinnostava 19 pelin listaus [13] peleistä, jotka käyttävät puheentunnistusta. Näistä Tom Clancy's Endwar, Hey you, Picachu!, Seaman ja Lifeline mai-

nittiin jo aiemmin tässä luvussa. Osasta listan peleistä löytyy vain vähän tietoa, eikä niiden VUI-ominaisuuksista saatu juurikaan käsitystä. Tässä tutkielmassa esitellään lyhyesti listan peleistä sellaiset, joiden lopullinen versio on julkaistu Steam-kauppapaikalla[3], ja joiden omilla verkkosivuilla kerrotaan niiden VUI-ominaisuuksista. Esitellyt pelit on taulukoitu taulukkoon 2.3.

One Hand Clapping on vuonna 2021 julkaistu kaksiulotteinen tasohyppelypeli, jossa ratkaistaan pulmatehtäviä hyräilemällä tai laulamalla mikrofoniin [14]. Vuonna 2015 julkaistu In Verbis Virtus on toiminta-pulmapeli, jossa ohjataan velhoa ja käytetään taikoja puheentunnistuksen avulla [15]. Vuonna 2015 julkaistu tieteis-reaaliaikastrategiapeli There Came an Echo mahdollistaa pelaajan yksiköiden ohjaamisen puheentunnistuksen avulla [16]. Vuonna 2020 julkaistu Radio General on toiseen maailmaansotaan sijoittuva reaaliaikastrategiapeli, jossa pelaajan tehtävänä on toimia kenraalina, joka antaa joukoilleen komentoja pelin puhelimen välityksellä [17]. Escape the Ayuwoki on vuonna 2019 julkaistu kauhuseikkailupeli, jossa pelaajan tehtävänä on paeta riivatusta kartanosta. Pelaajaa vaanii Ayuwoki-niminen olento, joka kuulee pelaajan mikrofoniin päästämät äänet, ja kykenee äänten perusteella jäljittämään pelaajaa. [18]

Taulukko 2.3: Vapaalla internet-haulla löydettyjä VUI-pelejä.

Pelin nimi	Vuosi	Julkaisija	Alusta	VUI
In Verbis Virtus	2015	Indomitus Games	PC	verbaalinen [15][19]
There Came an Echo	2015	Iridium Studios	PC, PS4	verbaalinen [20][16]
Escape the Ayuwoki	2019	Deadly Crow Games	PC	ei-verbaalinen [18][21]
Radio General	2020	Foolish Mortals Games	PC	verbaalinen [17][22]
One Hand Clapping	2021	HandyGames	useita	verbaalinen/ei-verbaalinen [14]

[3]https://store.steampowered.com/

3 Puhekäyttöliittymien arkkitehtuureja (TK2)

Tässä luvussa kerrotaan tutkielman tuloksista toisen tutkimuskysymyksen pohjalta. Aliluvussa 3.1 kerrotaan VUI-pelien toteutustavoista kirjallisuuskatsaukseen perustuen. Kirjallisuuskatsauksessa käytetyt artikkelit ovat samat kuin luvun 2 taulukossa 2.1.

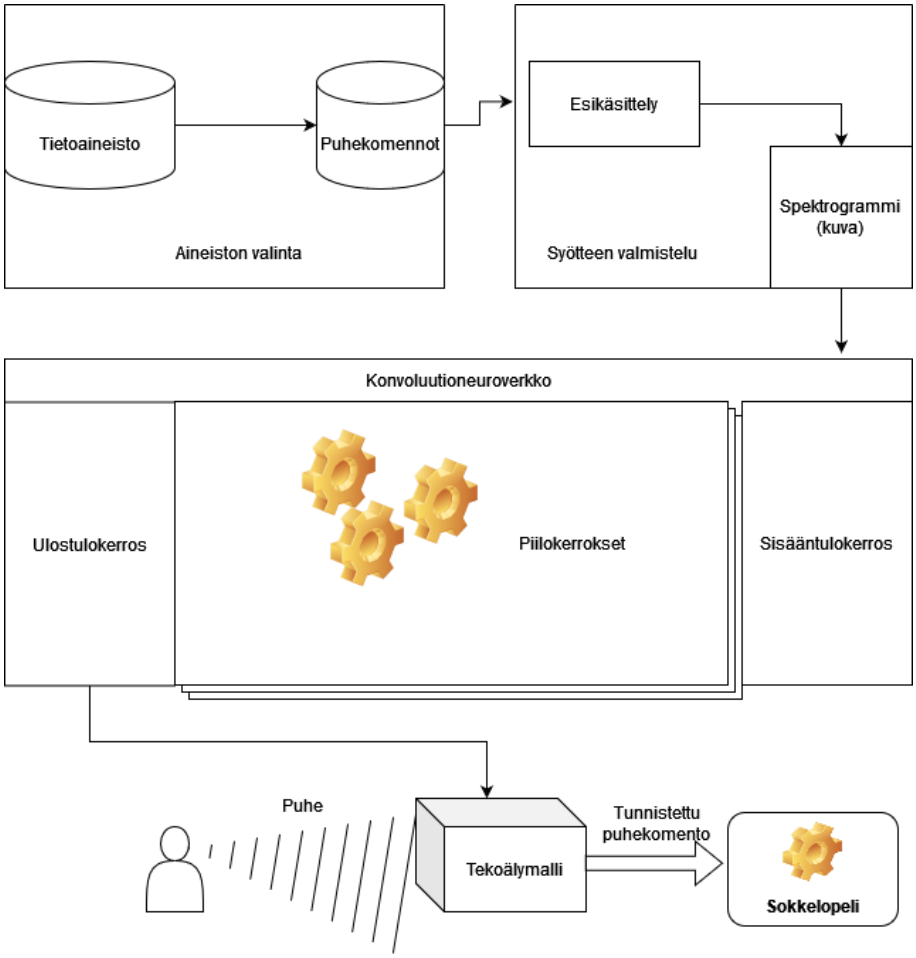
3.1 Puhekäyttöliittymien toteutustapoja

Tässä tutkielmassa käsitellyistä arkkitehtuureista moni käyttää puheentunnistukseen jotakin ohjelmointirajapintaa, eli *API*:a (engl. application programming interface). Esiteltyjen kaupallisesti julkaistujen pelien puheentunnistus-teknologioista ei kirjallisuuskatsauksessa löytynyt tarkkaa tietoa. Toisaalta tutkimuksissa esiintyneille prototyyppipeleille, löydettiin melko tarkat määrittelyt. Tällaisille prototyyppipelien esimerkkiarkkitehtuureille yhteistä on jonkinlaisen tekoälymallin hyödyntäminen.

Sainadhin ym. tutkimuksessa [11] pyrittiin laajentamaan puheentunnistuksen sovellettavuutta peleissä. Tutkimuksessa kehitettiin sokkelopeli, jossa navigoidaan puhekomentojen avulla. Peli puheentunnistus perustuu konvoluutioneuroverkon avulla koulutettuun tekoälymalliin, jonka kouluttamiseen käytettiin ”Google Speech Commands” -tietoaineistoa (engl. dataset). Tietoaineisto koostuu sekunninmittaisista ääninäytteistä, joissa on joko puhuttu englanninkielinen sana tai taustamelua. Tietoai-

neistossa eri sanoja on 30, joista jokainen sisältää ääninäytteitä eri äänen puhumina. Tekoälymalli pyrittiin kouluttamaan siten, että se pystyy tunnistamaan kahdeksan eri komentoa taustamelun peittämästä puheesta. Pelin käyttämät komennot ovat ”Left”, ”Right, ”Up”, ”Down”, ”Go” ja ”Stop”. Neuroverkon kouluttamisessa tietoaineiston muut sanat merkittiin tuntemattomiksi. Aluksi tietoaineisto esikäsiteltiin siten, että jokainen ääninäyte korjattiin sekunninmittaiseksi lisäämällä dataan tarvittaessa nollia. Myös taustakohinasta tehtiin sekunninmittaisia intervalleja. Ääninäytteistä generoidut spektrogrammit ovat neuroverkon syöte. Konvoluutioneuroverkko sisältää sisääntulokerroksen ja ulostulokerroksen lisäksi useita piilokerroksia ja miljoonia parametreja, joiden ansiosta se pystyy oppimaan toistuvia kuvioita kuvista ja siten se oppii luokittelemaan niitä. Tutkimuksessa testattiin neuroverkkoa erilaisilla piilokerroksilla ja lopulta neuroverkossa oli 22 piilokerrosta, jossa lähes samanlainen 4 kerroksen rakenne toistui viisi kertaa. Tietoaineiston 105000 ääninäytteestä 80 % käytettiin neuroverkon kouluttamiseen ja 20 % neuroverkon testaamiseen. Ehdotettu tekoälymalli tunnistaa pelin komennot 94,57 % tarkkuudella. Tutkimuksen metodeja on havainnollistettu kuvassa 3.1.

Halosen ym. tutkimuksessa [9] tutkittiin pelin Tom Clancy’s Endwar puhekäyttöliittymää käytettävyyden näkökulmasta heuristiikkojen avulla. Tutkimuksen tarkoitus oli kehittää heuristiikkoja pelien VUI:den toteutuksen ja suunnittelun avuksi. Tutkimuksessa ehdotetut heuristiikat ovat taulukossa 3.1. Tutkittu peli oli pelin Windows-versio, ja oli mainosten mukaan täysin pelattavissa pelkkien puhekomentojen avulla. Peli antaa palautetta multimodaalisti sekä äänen että GUI:n kautta. Peli ei tosin aina hyväksy puhekomentoja, vaikka GUI kertoisikin onnistuneesta komennosta. Jotkin komennot aiheuttivat poikkeuksellisen paljon virhetunnistuksia, jolloin pelaajan täytyy etsiä oikea komento GUI:n komentovalikon kautta. Osa puhekomennoista oli näppäinpainallukseen verrattuna hitaita ja aiheuttivat turhautumista. Puhekomennot olivat luonteva tapa ohjata joukkoja ja pelaajalla oli mahdollisuus har-



Kuva 3.1: Sokkelopelin puheentunnistuksen ja neuroverkon metodeja. Kuva on piirretty tutkimuksen [11] kuvitukseen perustuen.

joitella niitä. Peli ei tuntunut oppivan pelaajan puhetapaa tai ääntä vaan pelaaja joutui mukauttamaan omaa puhumistaan pelin puheentunnistukseen sopivaksi, eikä puhekomentoja pystynyt muokkaamaan. Tarkempaa erittelyä pelin arkkitehtuurista ja käytetyistä puheentunnistuksen menetelmistä ei kirjallisuuskatsauksessa löytynyt. Tutkimuksessa esitettiin 8 heuristiikkaa. Tässä tutkielmassa esiintyneiden prototyyppipelien pohjalta tutkimuksen [9] heuristiikkataulua voisi täydentää vielä uusilla heuristiikoilla ennakoivalle virheentunnistukselle (S09) sekä kontekstitietoiselle puhekomentojen tunnistamiselle (S10). Ehdotetut lisäheuristiikat on lisätty taulukkoon 3.1, jossa niiden numerot ovat selkeyden vuoksi laitettu sulkeiden sisään.

Hariprasadin ym. tutkimuksessa [2] kehitettiin kaksiulotteinen, puheohjattu mo-

Taulukko 3.1: VUI:den kehittämiseen ehdotettuja heuristiikkoja [9].

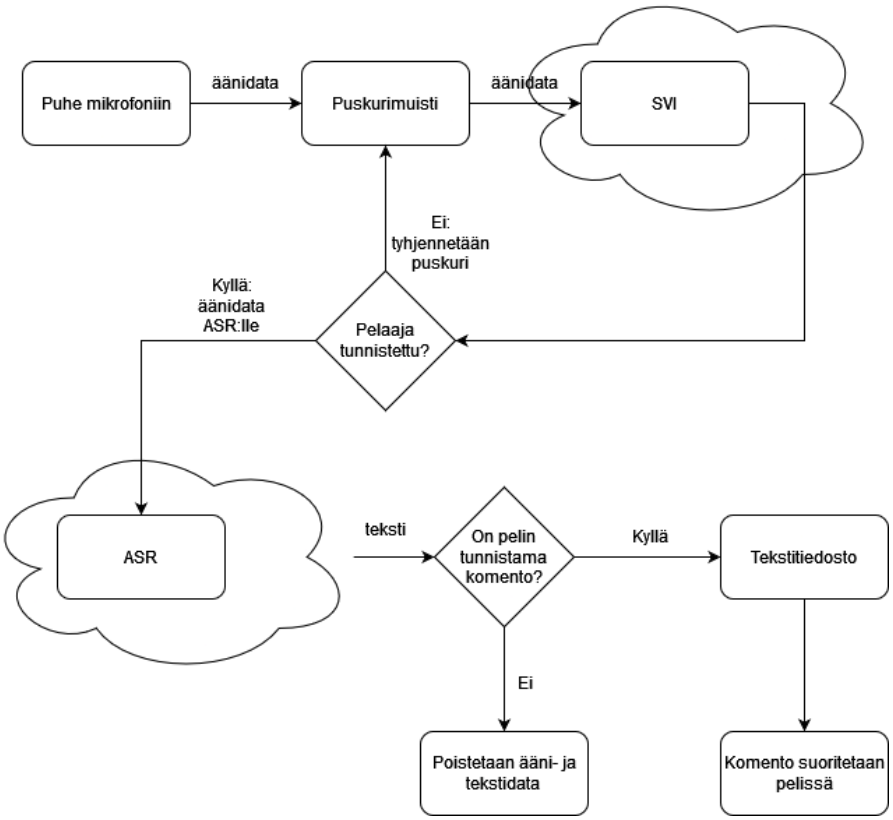
Numero	Heuristiikka	Selitys	Tom Clancy's Endwar
S01	peli antaa monimodaalista palautetta	peli vastaa puhekomentoon äänen, GUI:n, peliohjaimen värinän yms. kautta	ääni ja GUI [9]
S02	puhekomennot ovat täsmällisiä	puhekomennot eroavat toisistaan selkeästi	jotkin komennot aiheuttavat paljon virhetunnistuksia [9]
S03	puheentunnistuksen rajoitteet ovat pelaajan tiedossa	pelaaja tietää, mitä puhekomentoja on milloinkin käytettävissä	kyllä [9]
S04	puheentunnistus on käytännöllinen tapa ohjata peliä	puhekomennot tuntuvat järkeviltä ja käytännöllisiltä käyttää	jotkin komennot ovat liian hitaita [9]
S05	puheentunnistus on koherentti osa pelikokemusta	puhekomennot kohdistuvat toimintoihin, joissa ne ovat haluttu tapa ohjata peliä	kyllä [9]
S06	pelaaja voi harjoitella puhekomentoja	pelissä ominaisuus puhekomentojen harjoittelemiseen	kyllä [9]
S07	peli oppii tunnistamaan pelaajan äänen	peli pystyy tunnistamaan pelaajan äänen perusteella ja erottamaan eri pelaajat toisistaan	ei [9]
S08	puhekomennot ovat pelaajan määriteltävissä	pelaaja voi määritellä puhekomentoja itse	ei [9]
(S09)	peli kykenee tunnistamaan puhekomentoja tarkemmin pelitilanteen kontekstin perusteella	peli rajaa kontekstiin liittyvät puhekomennot pelitilanteen perusteella, ja käyttää suodattimia puhekomennon kontekstitietoiseen tunnistamiseen	ei tiedossa
(S10)	peli käyttää puhekomentojen tunnistamiseen ennakoivaa virheenkäsittelyä	virhetunnistuksen sattuessa peli valitsee pelaajan kannalta suotuisimman komennon	ei tiedossa

ninpeli. Peli on kehitetty Unity-pelimoottorilla. Peli kykenee ottamaan vastaan puhekomentoja kahdelta pelaajalta ja tunnistamaan pelaajan äänen avulla, kuten jo luvussa 2 mainittiin. Pelaajan ja puheen tunnistamiseen pelissä käytetään kahta Microsoft Azure[1]:n tekoäly-API:a. Pelaajan tunnistukseen, *SVI*:hin (engl. speaker verification and identification) käytetään *Speaker Recognition* -API:a. Automaattiseen puheentunnistukseen, *ASR*:n (engl. automatic speech recognition) *Speech to text* -API:a. Kommunikointi Azuren pilvipalveluihin tapahtuu Python-algoritmilla, joka välittää ääninäytteet prosessoitaviksi .wav-tiedostoina ja vastaanottaa prosessoinnin tulokset JSON-muotoisena datana. Pelin alussa pelaajien tunnistamiseen käytetyt ääninäytteet tallennetaan tietokoneen pysyvään muistiin. Ääninäytteet prosessoidaan SVI:n avulla, joka prosessoi niistä pelaajien äänitunnisteet. Palvelimella tapahtuva prosessointi perustuu syväoppivaan neuroverkkoon, *DNN* (engl. deep neural netrwork) sekä CNN:n. Pelaajien pelin aikana antamat puhekomennot tallentuvat puskurimuistiin, ja lähetetään ensin SVI:hin pelaajan tunnistusta varten. Jos pelaajan tunnistus onnistuu, lähetään ääninäyte ASR:lle, jolloin puheesta saadaan tekstimuotoinen. Jos edelleen ASR:n tuottama teksti vastaa jotakin pelin hyväksymää komentoa, se tallennetaan pelaajakohtaiseen tekstitiedostoon, josta Unity lopulta saa syötteen pelihahmon liikuttamista varten. Tutkimuksessa ehdotetut metodit ovat havainnollistettuna kuvassa 3.2.

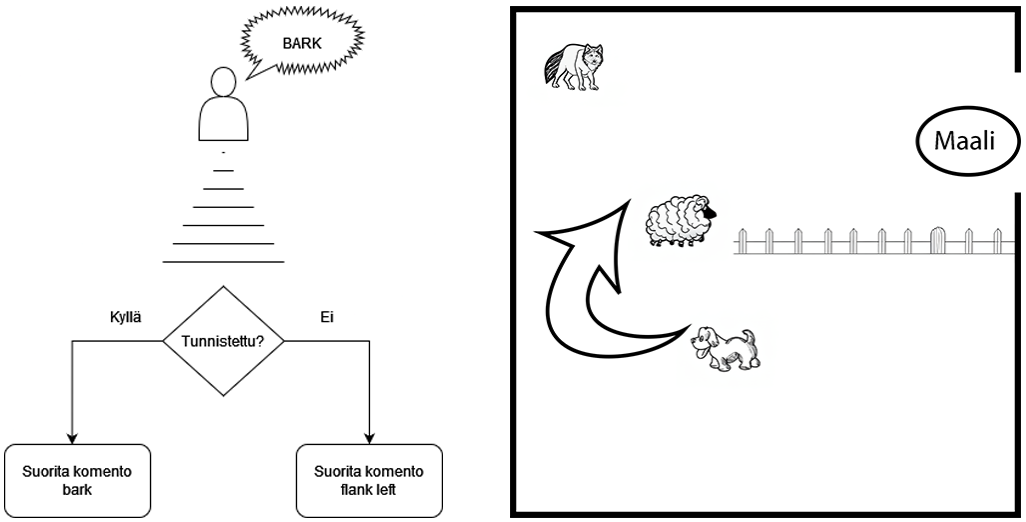
Luvussa 3.1 mainittu paimenkoirapeli on kehitetty Unity-pelimoottorilla. Pelin ASR:nä käytettiin Googlen pilvipalveluiden Speech to text[2] -API:a. Pyynnöt lähetetään suoraan Googlen API:in, jolloin prototyyppipeli on yhteensopiva eri alustojen kanssa. Prototyyppipelistä tehtiin versiot Linux, Windows sekä Mac OS -käyttöjärjestelmille. ASR:n palauttama tekstimuotoinen komento prosessoidaan pelin ennakoivan virheenkäsittelylogiikan avulla pelikomennoksi kuvaparissa 3.3 hahmoteltuun tapaan. [12]

[1] https://azure.microsoft.com/en-us/products/ai-services/ai-speech/

[2] https://cloud.google.com/speech-to-text



Kuva 3.2: Inklusiivisen moninpelin puheentunnistuksen metodeja. Kuva on piirretty tutkimuksen [2] kaavion pohjalta.



(a) Puheentunnistuslogiikka.

(b) Pelitilanne, jossa pelaaja yrittää antaa komennon "bark".

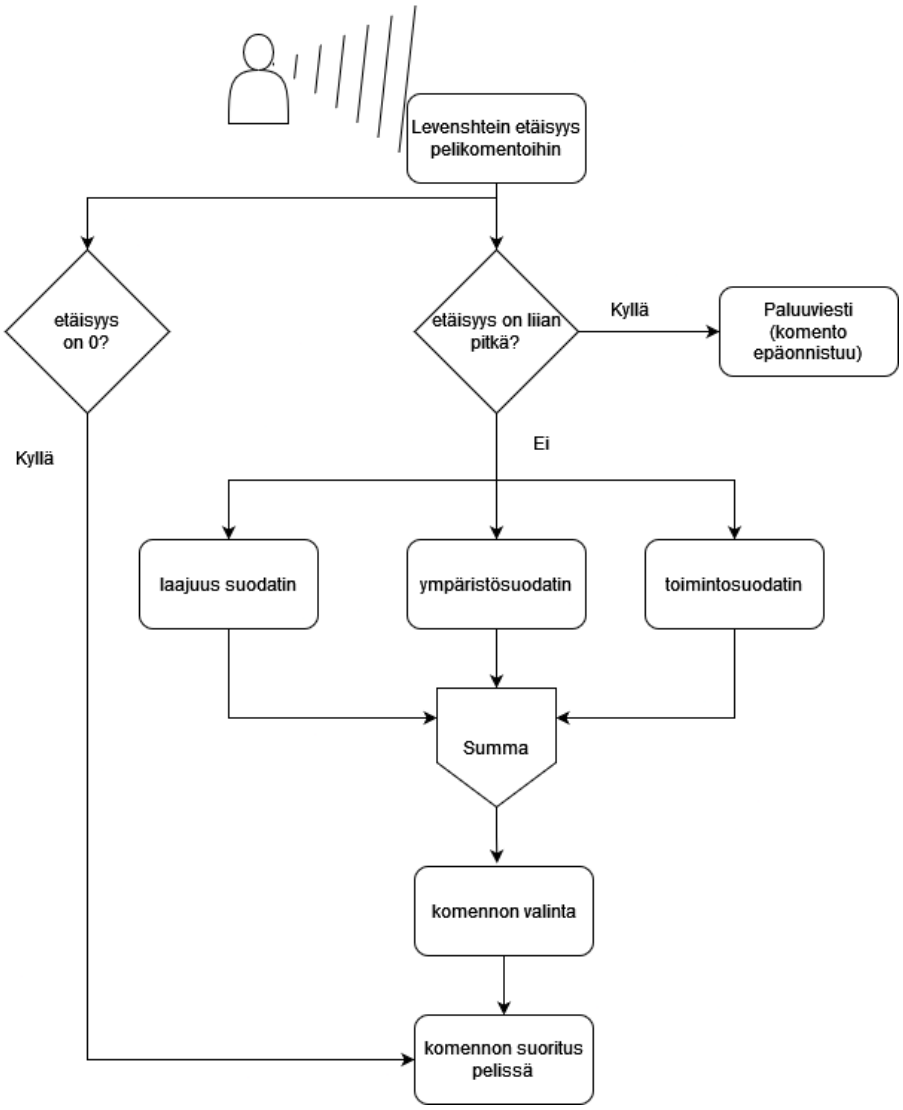
Kuva 3.3: Hahmotelma ennakoivasta virheenkäsittelystä pelitilanteessa. Kuva on piirretty tutkimuksen [12] kuviin perustuen.

Zarghamin ym. tutkimuksessa [8] kehitetty prototyyppi peli on kehitetty Unity-pelimoottorilla. Pelin ASR:nä käytettiin aluksi Unityn omaa Windows käyttöjärjestelmälle tarkoitettua ASR:ää, joka taas käyttää Windowsin sisäistä Windows Speech Recognition[3] -API:a. Prototyyppipeli siirtyi kuitenkin käyttämään VOSK Speech Recognition -API:a, sillä se ei vaadi verkkoyhteyttä. Peli tunnistaa komentoja joko kirjaimellisesti, kun ASR:n tunnistama fraasi vastaa kirjaimellisesti jotakin komentoa, tai arvioimalla sitä erilaisten suodattimien avulla. Laajuussuodatin (engl. scope filter) vertaa annettua komentoa pelin mahdollisiin komentoihin Levenshtein-etäisyyden perusteella. Ympäristösuodatin (engl. environment filter) laskee puhekomennon avulla pisteytyksen pelin sen hetkisessä kontekstissa saatavilla oleviin komentoihin. Toimintosuodatin (engl. actions filter) laskee puhekomennon pisteytyksen valittuna olevan peliobjektin mahdollisille toiminnoille. Lopulta suodattimien pisteytykset lasketaan yhteen. Summat vaihtelevat 4 ja 140 välillä, jossa suurempi luku tarkoittaa suurempaa luottamusta kyseiseen komentoon. Tutkimuksessa ehdotettu kontekstitietoinen lähestymistapa voi parantaa pelattavuutta ja pelikokemusta merkittävästi. Pelin käyttämä kontekstitietoinen metodi on hahmoteltu kuvaan 3.4.

Microsoftin julkaisemille Xbox 360 ja Xbox One -pelikonsoleille valmistetussa Kinect-liikeohjaimessa oli käytössä Windowsin puheentunnistuksen API[9]. Esimerkiksi pelien Splinter Cell: Blacklist ja Ryse: Son of Rome Xbox One -versioissa puhekomennot annettiin Kinect-sensorille [7]. Kinect on poistunut tuotannosta, mutta uusissa Xbox Series X ja Xbox Series S -konsoleissa on mahdollisuus puheohjaukseen Amazon Alexa tai Google Assistant äänitekoälysovellusten avulla [23].

Jo aiemmin mainittu, NUX projektissa kehitetty IVE on tehty Unreal Engine -pelimoottorille. Tutkimustyökaluksi kehitetty peli käyttää HTC Vive VR-lasien sisäistä mikrofonia äänilähteenään. Peliin on toteutettu lokitustoiminto, jonka avulla näppäinpainallukset, silmien liikekuviot sekä ääninäytteet aikaleimoineen voidaan

[3]https://learn.microsoft.com/en-us/windows/apps/design/input/speech-recognition



Kuva 3.4: Kontekstitietoinen puhekomennon tunnistus Escape the echo -pelissä. Kuva on piirretty tutkimuksen [8] kaavioon pohjautuen.

tallentaa .csv-tiedostoon myöhempää tarkastelua varten. IVE:n kehitys on ollut iteratiivinen prosessi ja projektin alussa dialogien kehityksen apuna käytössä oli Wizard of Oz -menetelmä. [10]

4 Prototyyppipelin implementaatio

Tässä luvussa esitetään prototyyppi VUI-pelin toteutukselle. Prototyypin kehitti tämän tutkielman kirjoittaja. Se tehtiin, koska kirjallisuuskatsauksessa kaupallisten pelien VUI:lle ei löytynyt tarkkaa määritelmää, ja tätä puutetta haluttiin korjata kirjoittajan omalla, Unity-pelimoottorin puheentunnistukseen pohjautuvalla määritelmällä. Lisäksi sen avulla haluttiin konkretisoida tämän tutkielman löydöksiä. Esitetty prototyyppi on vain yksi vaihtoehto ja sen soveltuvuus kaupallisiin peleihin on tapauskohtaista.

4.1 Prototyypin kuvaus

Prototyyppi tehtiin Unity 6 pelimoottorilla. Se on hyvin yksinkertainen prototyyppi, jossa liikutellaan palloa eteen, taakse, vasemmalle, oikealle, ylös ja alas kolmiulotteisessa pelimaailmassa komennoilla ”forward”, ”back”, ”left”, ”right”, ”up” sekä ”down”. Nämä komennot valittiin, sillä ne ovat lyhyitä ja yksinkertaisia. Tässä haluttiin vain implementoida valikoima komentoja, joilla peliä voi ohjata pelkästään äänen avulla.

4.2 Prototyypin toteutus

Prototyyppi hyödyntää Unityn omaa puheentunnistus API:a, joka puolestaan käyttää Windows-käyttöjärjestelmän sisäistä puheentunnistusta. Windowsin puheentunnistus on tuettu vain englannin, ranskan, saksan, japanin, mandariinikiinan ja es-

panjan kielillä [24]. Windowsin puheentunnistus edellyttää, että käyttöjärjestelmän kieliasetus vastaa puheentunnistukselle asetettua kielivalintaa, joka ilmeni asetusten asettamisen yhteydessä. Käytännön syistä kielivaihtoehdoista prototyyppiprojektin kieleksi valikoitui englanti. Tässä toteutuksessa päätettiin käyttää jatkuvaa puheentunnistusta, joka tunnistaa pelaajan puheesta ennalta määriteltyjä avainsanoja.

Liitteen A koodilistauksessa 2 on esitettynä pallon liikuttelua varten kehitetty Movement-luokka. Tässä otetaan käyttöön Unityn nimiavaruus ja luokka periytyy Unityn omasta MonoBehaviour-luokasta. Metodi Input saa parametrinaan puheentunnistuksen tuottaman merkkijonon, jonka perusteella if-lausekkeet valitsevat oikean toiminnon. Jokainen toiminto tulostaa tietoa komennosta Unityn konsoliin. Kukin komento liikuttaa palloa lisäämällä sen paikkavektoriin halutun liikkumissuunnanmukaisen yksikkövektorin. Tämä liikuttaa palloa pelimaailmassa.

Liitteen A koodilistauksessa 1 on esitettynä puheentunnistuksen sisältävä ASR-luokka. ASR ottamaan lisäksi käyttöön nimiavaruuden, josta löytyvät Unityn puheentunnistukseen liittyvät luokat. Tässä luodaan KeywordRecoqnizer-tyyppinen attribuutti sekä alustetaan taulukko halutuista komennoista merkkijonoina. [25] Tässä otetaan myös viittaus Movement-luokkaan ja viittaus tehdään suoraan Unityssa vetämällä ja pudottamalla Movement-skriptin sisältävä pallo oikeaan kenttään.

Start-metodi on Unityn suoritusympäristön kutsuma metodi ja siinä luodaan uusi KeywordRecognizer-olio, joka saa parametreinaan taulkon puhekomennoista ja tunnistuksen tarkkuuden rajan määrittävän ConfidenceLevel.low, joka on enum-tyyppinen. Tällä haluttiin madaltaa komentojen tunnistamisen kynnystä. Sitten KeywordRecognizer-oliolle rekisteröidään takaisinkutsumetodi, jonka jälkeen puheentunnistus käynnistetään kutsumalla KeywordRecognizer-olion Start-metodia. Takaisinkutsumetodi saa parametrinaan tapahtumankäsittelijän struct-tyypisen viittauksen, josta puheentunnistuksen tunnistama merkkijono löytyy ominaisuudesta args.text. Takaisinkutsumetodi kutsuu edelleen Movement-olion Input-metodia ja

välittää merkkijonon sille. [25]

4.3 Havaintoja ja vertaus heuristiikoihin

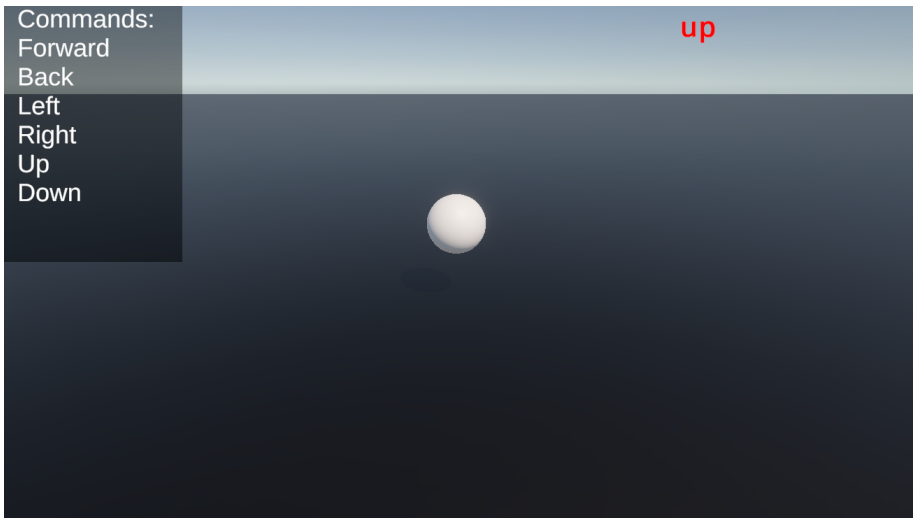
Prototyypin toteutuksessa havaittiin, että komennon tunnistaminen aiheuttaa noin sekunnin mittaisen viiveen. Peräkkäisten komentojen väliin vaaditaan myös noin puolen sekunnin mittainen hiljaisuus, jotta tunnistaminen on tarkempaa. Aluksi prototyyppi kehitettiin ilman heuristiikkoja. Prototyyppiä kuitenkin kehitettiin edelleen heuristiikkoihin perustuen vaikka kaikkia ei tässä kontekstissa ollut ajallisesti mahdollista toteuttaa. Lopputulos oli käyttäjäystävällisempi, kun heuristiikat otettiin harkintaan. Taulukossa 4.1 verrataan prototyyppipeliä aiemmin esitettyihin heuristiikkoihin. Heuristiikkaa numero 7 ei pysty toteuttamaan prototyypin käyttämällä API:lla. Pelaajan tunnistamiseen tarvitaan siis jokin toinen API. Heuristiikka numero 8 ei toteudu prototyypissä, mutta sen vaatiman toiminnallisuuden lisääminen olisi suoraviivaista. Heuristiikkojen numero 9 ja 10 toteuttaminen ei onnistu KeywordRecognizerin avulla, sillä avainsanat joko tunnistetaan tai ei. Jos siis halutaan toteuttaa kaikki heuristiikat, pitää valita jokin toinen lähestymistapa.

Taulukko 4.1: Prototyyppipelin vertaus heuristiikoihin [9].

Numero	Heuristiikka	Prototyyppi
S01	peli antaa monimodaalista palautetta	peli tulostaa komennon tekstin ruudulle ja toistaa lyhyen äänen
S02	puhekomennot ovat täsmällisiä	kyllä
S03	puheentunnistuksen rajoitteet ovat pelaajan tiedossa	kyllä, komennot ovat koko ajan näytöllä
S04	puheentunnistus on käytännöllinen tapa ohjata peliä	kyllä, joskin ne ovat hieman hitaita
S05	puheentunnistus on koherentti osa pelikokemusta	kyllä
S06	pelaaja voi harjoitella puhekomentoja	kyllä
S07	peli oppii tunnistamaan pelaajan äänen	ei
S08	puhekomennot ovat pelaajan määriteltävissä	ei
(S09)	peli kykenee tunnistamaan puhekomentoja tarkemmin pelitilanteen kontekstin perusteella	ei
(S10)	peli käyttää puhekomentojen tunnistamiseen ennakoivaa virheenkäsittelyä	ei

Prototyyppiin lisättiin graafinen elementti, jossa kaikki käytössä olevat puhekomennot ovat näkyvillä. Onnistunut tunnistus tulostaa komennon ruudulle ja toistaa lyhyen äänimerkin. Kuvassa 4.1 on kuvankaappaus prototyyppipelistä tilanteessa,

jossa on komento ”up” on tunnistettu ja pallo on liikkunut ylös. Taulukossa 4.2 on prototyyppipelin puhekomentojen tunnistamisen tarkkuuksia, kun kukin komento annettiin 100 kertaa. Komennon ”up” tunnistus oli tässä testissä selkeästi huonoin. Testin aikana ympäristö oli melko meluisa, joka luultavasti vaikutti lopputulokseen.



Kuva 4.1: Kuvankaappaus prototyyppipelistä.

Taulukko 4.2: Prototyyppipelin komentojen tunnistustarkkuus 100 komennolla.

Komento	Tunnistustarkkuus 100 komennolla
forward	80 %
back	91 %
left	91 %
right	92 %
up	67 %
down	91 %

5 Pohdinta

Tässä luvussa kerrataan tutkielman löydökset ja tulokset esitettyjen tutkimuskysymysten TK1 ja TK2 pohjalta. Lopuksi esitetään tutkielman rajoitteet, yhteenveto sekä tulevaisuuden tutkimussuuntia.

5.1 Käytännön implikaatiot

Tutkielman löydöksiä voidaan käyttää tulevien VUI-pelien kehityksen tukena. Analyysissa ilmenneitä VUI:den ongelmakohtia sekä niiden ehdotettuja ratkaisuja voidaan hyödyntää tulevien pelien VUI:den käyttäjäkokemuksen sekä saavutettavuuden ja immersion parantamiseen. Tulevat tutkimukset voivat hyötyä tässä tutkielmassa tehdystä analyysista sekä sisällöllisesti että lähdeluettelon perusteella. Tutkielman löydöksiä voidaan hyödyntää myös muiden, kuin videopelien VUI:den tutkimuksessa.

Tulevissa kehitysprojekteissa ja tutkimuksissa voidaan hyödyntää tutkielman tuloksiin pohjautuvaa, kuvan 5.1 karkeata päätöksenteon rakennetta, jonka tarkoitus on auttaa tasapainoilussa VUI:n saavutettavuuden ja immersion välillä. Rakenne on tarkoitus käydä läpi ylhäältä alas, jossa ensimmäinen päätös on modaalisuus, eli onko VUI:n lähtökohta saavutettavuuden vai immersion parantaminen. Jos halutaan mahdollisimman saavutettava peli, jota voi ohjata pelkän puheen avulla, on valinta unimodaalinen. Multimodaalinen taas soveltuu paremmin immersion parantamiseen, jossa puheohjaus on vapaavalintaista tai vain osittaista. Ei-verbaalinen

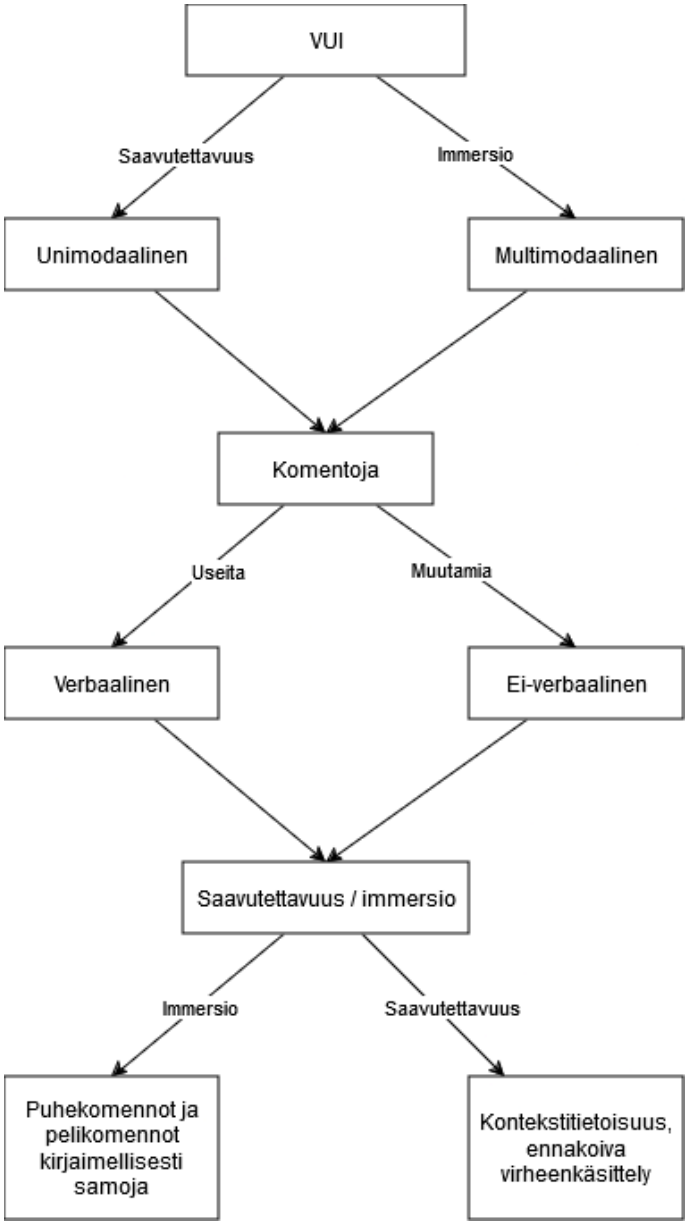
VUI taas voi tulla kyseeseen, jos komentoja on vain muutamia. Koska kaaviota voi käyttää sekä saavutettavan että immersiivisen VUI:n määrittelyyn, on viimeinen valintalaatikko näiden kahden välillä. Valinnat eivät kuitenkaan ole toisensa täysin poissulkevia, vaan niiden tarkoitus on vain ilmaista, mikä ratkaisu soveltuisi paremmin kuhunkin valintaan. Jos haluttaisiin esimerkiksi toteuttaa immersiivinen VUI, jossa pelaaja voi kutsua pelihahmonsa koiraa viheltämällä, valinnat olisivat multimodaalinen, ei-verbaalinen sekä komentojen kirjaimellisuus. Edellisessä esimerkissä kirjaimellisuudella tarkoitetaan sitä, että pelaajan pelihahmo toistaa vihellyksen pelimaailmassa.

Lisäksi olisi hyvä pohtia, onko tuleva peli sellainen, jossa tulisi olla VUI, ja erityisesti miten pelin VUI-ratkaisu pyrkii vastaamaan luvussa 3.1 esitettyihin heuristiikkoihin. Luvussa 4 kehitetty prototyyppipeli voi toimia alustavana lähtökohtana Unity pelimoottorilla kehitettäviin VUI-peliprojekteihin. Prototyyppipeli on yksinkertainen, mutta siinäkin hyödyttiin heuristiikkojen käytöstä. Sitä voi edelleen jatkokehittää vastaamaan myös niihin heuristiikkoihin, joihin prototyyppi ei pystynyt vastamaan.

Yleisesti voidaan todeta, että videopelien VUI:t tulisi toteuttaa niin, että se parantaa pelin immersiota ja myös saavutettavuutta. Immersio voi rikkoontua huonosti toteutetuista tai huonosti soveltuvista VUI:sta. Puhekomentojen tulisi olla järkevä tapa kontrolloida peliä ja niiden käytön on oltava luontevaa. Pelaajan kannalta voisi lisäksi olla parempi, ettei puhekomennon virhetunnistus näy ulospäin, tai että VUI hyödyntää esimerkiksi kontekstitietoista virheentunnistusta tai kontekstitietoista suodatusta komennon päättelyyn.

5.2 Implikaatiot ja kontribuutiot teoriaan

Tätä tutkielmaa voidaan pitää suppeana koosteena videopelien VUI-tutkimuksista viimeisen viiden vuoden ajalta. Analyysissa käsiteltiin myös vanhempia artikkeleita,



Kuva 5.1: Tässä tutkielmassa käsittelylukujen ja prototyyppitoteutuksen pohjalta johdettu VUI:n saavutettavuuden ja immersion valintakaavio

sillä niiden koettiin olevan relevantteja tutkielman kannalta. Analyysin perusteella voidaan todeta, että syväoppivat konvoluutioneuroverkot ovat tärkeä menetelmä ja keskeinen osa näissä järjestelmissä. Tutkielmassa koostetut ohjelmointirajapinnat voivat hyödyttää tulevia tutkimuksia ja pelinkehitysprojekteja arkkitehtuuriratkaisuissa.

Tutkielmassa esiintyneiden prototyyppipelien pohjalta esitettiin kaksi uutta heu-

ristiikkaa VUI:den arviointiin. Ensimmäisessä ehdotetaan kontekstitietoiseen puheentunnistukseen liittyvää heuristiikkaa. Toisessa ehdotetaan ennakoivaan virheentunnistukseen liittyvää heuristiikkaa. Kumpikin ehdotetuista heuristiikoista pyrkii piilottamaan VUI:n virhetunnistuksia, jotka usein aiheuttavat turhautumista pelaajien keskuudessa. Tällaiset metodit eivät siis välttämättä ole hyviä ratkaisuja muissa sovelluksissa, joissa virhetilanne olisi esitettävä käyttäjälle selkeästi.

Tutkielman laajuudessa käsitellyt VUI-pelit edustavat vain murto-osaa VUI-peleistä. Erityisesti kaupallisesti julkaistujen pelien VUI-ratkaisuista ei saatu hyvää käsitystä ja tätä olisi syytä tutkia enemmän tarkemmissa tulevaisuuden tutkimuksissa. Hyvin toteutettu VUI voi parantaa pelin saavutettavuutta ja immersiota. Toisaalta huonosti toteutettu tai huonosti peliin soveltuva VUI aiheuttaa usein turhautumista tai voi tuntua epämiellyttävältä tai nololta käyttää. Tulevien VUI-pelien kannalta voisi olla suotuisaa tutkia tarkemmin erityisesti sekä erittäin huonoja että erittäin hyviä VUI-toteutuksia ja vertailla, minkälaisia ominaisuuksia ja eroavaisuuksia niissä on.

5.3 Yhteenveto

Aineistossa esiintyneiden videopelien VUI:t on useimmiten toteutettu lisätoiminnallisuutena tavanomaisten sisääntulojen rinnalla. Vain muutama esimerkkipeli oli täysin puheohjattu, ja esimerkiksi pelissä Tom Clancy's Endwar täysi puheohjaus on kuitenkin vapaavalintainen. VUI-komennot löytyvät monesta pelistä myös valikoiden kautta. Pelissä Tomb Raider: Definitive edition puheohjausta käytettiin esimerkiksi pelihahmon varusteiden vaihtamiseen ja pelin tauottamiseen. Taktisissa ammuntapeleissä, kuten SOCOM: U.S. Navy SEALs ja Rainbow six 3 puheohjausta voidaan käyttää joukkuetovereiden komentamiseen. Puhekomentojen aktivointi vaatii usein jonkin push-to-talk-painikkeen käyttöä, sillä useimmissa VUI-peleissä puheentunnistus ei ole jatkuvaa.

VUI:t voidaan jakaa karkeasti verbaalisiin ja ei-verbaalisiin VUI:hin. Verbaalisissa VUI:ssa pelikomennoiksi tunnistetaan ASR:n avulla sanoja ja lauseita. Puhekomennot ovat ennalta määriteltyjä ja niitä on rajallinen määrä. Ei-verbaaliset VUI:t perustuvat äänen voimakkuuden ja sävelkorkeuden tunnistamiseen. Esimerkiksi pelissä One Hand Clapping pelin pulmia ratkaistaan hyräilemällä tai laulamalla. Ei-verbaalisten komentojen tunnistaminen on tarkempaa kuin puhuttujen sanojen tunnistaminen, mutta myös toiminnallisesti rajoitetumpaa.

Pelien VUI-toteutusten koettiin joissakin peleissä parantavan immersiota ja sen koettiin parantavan samaistumisen tunnetta ohjattavaan pelihahmoon. Esimerkiksi pelin Splinter cell: Blacklist monimodaalinen VUI oli erityisen onnistunut; pelihahmo toistaa pelaajan antamia komentoja samoilla sanoilla, joka tekee pelikokemuksesta intiimimmän. Tomb Raider: Definitive Editionissa VUI taas koettiin hankalana ja jopa nolona käyttää. Pelien VUI:den todettiin mahdollisesti parantavan pelin saavutettavuutta. Karaokepeleissä, kuten SingStar ja Rockband, VUI on hyvin keskeisessä roolissa.

Eräs VUI:den ongelma on komentojen virheellinen tunnistaminen. Puhekomennon virhetunnistus pelitilanteessa lisää pelaajan turhautumista ja turhautuminen taas vaikuttaa pelaajan antamiin puhekomentoihin tavalla, joka lisää virhetunnistuksen riskiä seuraavissa puhekomennoissa. Tätä pyrittiin tutkimuksissa korjaamaan ennakoivalla virheenkäsittelyllä ja kontekstitietoisella puheentunnistuksella. Eräässä tutkimuksessa käytettiin erilaisia suodattimia puhekomennon tunnistamiseen pelin hetkellisessä kontekstissa saatavilla olevien komentojen valintaan. Suodattimien käyttö perustuu pisteytykseen, joiden perusteella puhekomento voidaan tulkita pelikomennoksi.

Tutkielmassa esiintyneiden kaupallisten pelien osalta VUI:den teknisen toteutuksen yksityiskohdat jäivät epäselviksi. Osa kaupallisista peleistä tosin toteutti VUI:n vain pelin XBox One versiossa Kinect-liikeohjaimen kautta. Aineiston tutki-

musten prototyyppipelien VUI:sta sen sijaan saatiin hyvä käsitys. VUI:t tarvitsevat ASR:n, joka hyödyntää puheentunnistukseen koulutettua tekoälymallia puhekomentojen tunnistamiseen. Moni tutkimus hyödynsi valmiiksi saatavilla olevia puheentunnistuksen API:a. Eräässä tutkimuksessa puheentunnistusta varten kehitettiin oma syväoppiva konvoluutioneuroverkko. Syväoppivat konvoluutioneuroverkot soveltuvat hyvin puheentunnistuksen tehtävissä käytettävien tekoälymallien kouluttamiseen.

Puheentunnistuksen API:ja on saatavilla pilvipalveluina, paikallisesti ja esimerkiksi Windows-käyttöjärjestelmään sisäänrakennettuna. Unity pelimoottori käyttää puheentunnistukseen Microsoftin sisäistä puheentunnistus API:a. Pilvipalvelut ovat yleensä maksullisia. Paikallisesti, ilman verkkoyhteyttä toimiva API on esimerkiksi VOSK Speech recoqnition.

Rajoitukset

Tutkielmassa keskityttiin yksinomaan videopelien VUI:hin, joten muiden kuin pelien järjestelmien osalta tämä tutkielma ei välttämättä ole relevantti. Tutkielmassa ei esitelty kaikkia mahdollisia VUI:n sisältäviä pelejä, joten tutkielman katsaus peleihin ei ole täydellinen.

Analyysissä ei saatu yksityiskohtaista tietoa kaupallisten pelien arkkitehtuurista, sillä tätä tietoa ei lähdemateriaalista löytynyt ja puuttuvan tiedon hankkiminen olisi vaatinut muita tutkimusmenetelmiä kirjallisuuskatsauksen lisäksi tai huomattavasti laajempaa kirjallisuusaineistoa.

Tutkielmassa esiteltiin vain aineistossa esiintyneet prototyyppipelit ja aineistossa esiintyneet kaupallisesti julkaistut pelit. Vapaalla verkkohaulla löydetyistä peleistä valikoitiin mukaan sellaiset pelit, joilla on omat verkkosivut sekä myyntisivu Steam-kauppapaikalla. Aineistossa ei siis ole lainkaan julkaisuun tulossa olevia pelejä eikä hyvin pieniä yksityisten kehittäjien pelejä.

Tulevaisuuden tutkimussuunnat

Tulevissa tutkimuksissa voisi yhtenä tutkintalinjana pitää kehittäjien haastatteluihin pohjautuvaa tiedonhankintaa kaupallisten pelien VUI:den yksityiskohtaisista teknisistä toteutuksista. Tutkielmassa esitetty listaus VUI-peleistä ei ole täydellinen, joten laajempi katsaus VUI-pelien kokonaistilanteesta vaatii laajempaa tutkimusta. Analyysi julkaisuun tulossa olevista VUI-peleistä voisi antaa paremman kuvan tulevista kehityssuunnista ja uusista innovaatioista.

Lähdeluettelo

[1] J. Haaramo, *Tilastokeskus - Vapaa-ajan osallistuminen 2017*, fi, Publisher: Tilastokeskus. url: https://stat.fi/til/vpa/2017/02/vpa_2017_02_2019-01-31_tie_001_fi.html (viitattu 04. 10. 2024).

[2] R. Hariprasad, N. Dhariwal ja P. Swarnalata, "Voice Stimulated Inclusive Multiplayer Game Development with Speaker Recognition", teoksessa *2023 Third International Conference on Smart Technologies, Communication and Robotics (STCR)*, vol. 1, joulukuu 2023, s. 1–6. DOI: 10.1109/STCR59085.2023.10396908.

[3] S. F. Ismail, R. Hashim ja S. Z. Z. Abidin, "Collaborative bridge game: A comparative study on User Interface design", teoksessa *2010 International Conference on Information Retrieval & Knowledge Management (CAMP)*, maaliskuu 2010, s. 34–39. DOI: 10.1109/INFRKM.2010.5466950.

[4] V. M. Carvalho ja M. A. Formico Rodrigues, "Investigating and Comparing the Perceptions of Voice Interaction in Digital Games: Opportunities for Health and Wellness Applications", teoksessa *2023 IEEE 11th International Conference on Serious Games and Applications for Health (SeGAH)*, ISSN: 2573-3060, elokuu 2023, s. 1–8. DOI: 10.1109/SeGAH57547.2023.10253798.

[5] *SOCOM U.S. Navy SEALs*, [DVD-ROM], Austria: Sony Computer Enternainment America Inc., 2002.

[6] C. Murad, H. Candello ja C. Munteanu, "What's The Talk on VUI Guidelines? A Meta-Analysis of Guidelines for Voice User Interface Design", en, teoksessa *Proceedings of the 5th International Conference on Conversational User Interfaces*, Eindhoven Netherlands: ACM, heinäkuu 2023, s. 1–16, ISBN: 9798400700149. DOI: 10.1145/3571884.3597129.

[7] M. Carter, F. Allison, J. Downs ja M. Gibbs, "Player Identity Dissonance and Voice Interaction in Games", en, teoksessa *Proceedings of the 2015 Annual Symposium on Computer-Human Interaction in Play*, London United Kingdom: ACM, lokakuu 2015, s. 265–269, ISBN: 978-1-4503-3466-2. DOI: 10.1145/2793107.2793144.

[8] N. Zargham, M. L. Fetni, L. Spillner, T. Muender ja R. Malaka, ""I Know What You Mean": Context-Aware Recognition to Enhance Speech-Based Games", teoksessa *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*, sarja CHI '24, New York, NY, USA: Association for Computing Machinery, toukokuu 2024, s. 1–18, ISBN: 9798400703300. DOI: 10.1145/3613904.3642426.

[9] A. Halonen, S. Hyrynsalmi, K. Kimppa, T. Knuutila, J. Smed ja H. Hakonen, *Towards Usability Heuristics for Games Utilizing Speech Recognition*. helmikuu 2012, Pages: 69. DOI: 10.13140/RG.2.1.1530.8569.

[10] K. Buchta, P. Wójcik, M. Pelc et al., "NUX IVE - a research tool for comparing voice user interface and graphical user interface in VR", teoksessa *2022 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, maaliskuu 2022, s. 982–983. DOI: 10.1109/VRW55335.2022.00342.

[11] K. V. Sainadh, K. Satwik, V. Ashrith ja S. Vekkot, "Playing Maze using Voice Commands", teoksessa *2023 14th International Conference on Com-*

puting Communication and Networking Technologies (ICCCNT), ISSN: 2473-7674, heinäkuu 2023, s. 1–7. DOI: 10.1109/ICCCNT56998.2023.10307127.

[12] N. Zargham, J. Pfau, T. Schnackenberg ja R. Malaka, "“I Didn't Catch That, But I'll Try My Best”: Anticipatory Error Handling in a Voice Controlled Game", en, teoksessa *CHI Conference on Human Factors in Computing Systems*, New Orleans LA USA: ACM, huhtikuu 2022, s. 1–13, ISBN: 978-1-4503-9157-3. DOI: 10.1145/3491102.3502115.

[13] K. Howarth, C. M. Teressa ja S. Go, *Games That Feature Speech Recognition*, en, Section: Lists, lokakuu 2021. url: https://www.thegamer.com/games-with-voice-commands-speech-recognition/ (viitattu 23. 10. 2024).

[14] *One Hand Clapping | Music Puzzle Platform Adventure*, en-US. url: https://handy-games.com/en/games/one-hand-clapping/ (viitattu 23. 10. 2024).

[15] *Indomitus Games – Development and consulting for games and VR*, en-US. url: https://www.indomitusgames.com/b2b/ (viitattu 23. 10. 2024).

[16] *Home*, en-US. url: https://playiridium.com/ (viitattu 23. 10. 2024).

[17] *Radio General*. url: http://www.foolish-mortals.net/radiogeneral (viitattu 23. 10. 2024).

[18] *Escape the Ayuwoki | DCG*. url: https://playescapetheayuwoki.com/ (viitattu 23. 10. 2024).

[19] *In Verbis Virtus on Steam*, en. url: https://store.steampowered.com/app/242840/In_Verbis_Virtus/ (viitattu 19. 11. 2024).

[20] *There Came an Echo on Steam*, en. url: https://store.steampowered.com/app/319740/There_Came_an_Echo/ (viitattu 19. 11. 2024).

[21] *Escape the Ayuwoki on Steam*, en. url: https://store.steampowered.com/app/1177660/Escape_the_Ayuwoki/ (viitattu 19. 11. 2024).

[22] *Radio General on Steam*, en. url: https://store.steampowered.com/app/1011610/Radio_General/ (viitattu 19. 11. 2024).

[23] *Using voice commands to turn your Xbox on and off | Xbox Support*. url: https://support.xbox.com/en-US/help/hardware-network/digital-assistant-voice-commands/turn-on-turn-off-with-voice-commands (viitattu 10. 10. 2024).

[24] Microsoft, *Windows Speech Recognition commands - Microsoft Support*. url: https://support.microsoft.com/en-us/windows/windows-speech-recognition-commands-9d25ef36-994d-f367-a81a-a326160128c7 (viitattu 14. 12. 2024).

[25] Unity Technologies, *Unity - Scripting API: KeywordRecognizer*, en. url: https://docs.unity3d.com/6000.0/Documentation/ScriptReference/Windows.Speech.KeywordRecognizer.html (viitattu 14. 12. 2024).

Liite A Prototyyppipelin koodi

Tässä liitteessä esitetään prototyyppipelin ohjelmakoodi Unity 6 projektissa. Ensin esitetään luokka ASR ja seuraavana luokka Movement.

Ohjelmalistaus 1 ASR-luokka.

```
{csharp}
    using TMPro;
    using UnityEngine;
    using UnityEngine.Windows.Speech;
    public class ASR : MonoBehaviour
    {
        private KeywordRecognizer recognizer;
        private string[] keywords = {"forward",
        "left", "right", "back", "up", "down"};
        AudioSource audioSource;
        [SerializeField]
        private Movement m;

        [SerializeField]
        TextMeshProUGUI commandText;

        private void Start()
        {
            audioSource = GetComponent<AudioSource>();
            recognizer = new KeywordRecognizer(keywords,
            ConfidenceLevel.Low);
            recognizer.OnPhraseRecognized += OnPhraseRecognized;
            recognizer.Start();
        }

        private void OnPhraseRecognized(PhraseRecognizedEventArgs args)
        {
            m.Input(args.text);
            commandText.text = args.text;
            audioSource.PlayOneShot(audioSource.clip);
        }

    }
```

Ohjelmalistaus 2 Movement-luokka.

```
{csharp}
      using UnityEngine;

      public class Movement : MonoBehaviour
      {
          public void Input(string axis)
          {
              if (axis == "left")
              {
                  Debug.Log("Moving left");
                  transform.position += Vector3.left;
              }
              if (axis == "forward")
              {
                  Debug.Log("Moving forward");
                  transform.position += Vector3.forward;
              }
              if (axis == "right")
              {
                  Debug.Log("Moving right");
                  transform.position += Vector3.right;
              }
              if (axis == "back")
              {
                  Debug.Log("Moving back");
                  transform.position += Vector3.back;
              }
              if (axis == "up")
              {
                  Debug.Log("Moving up");
                  transform.position += Vector3.up;
              }
              if (axis == "down")
              {
                  Debug.Log("Moving down");
                  transform.position += Vector3.down;
              }
          }
      }
```