Ville Raitio

Liike- ja eleohjaus mobiilipelissä

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Mediatekniikan koulutusohjelma

Insinöörityö

27.4.2016

Metropolia

Tekijä Otsikko Sivumäärä Aika	Ville Raitio Liike- ja eleohjaus mobiilipelissä 32 sivua 27.4.2016
Tutkinto	Insinööri (AMK)
Koulutusohjelma	Mediatekniikka
Suuntautumisvaihtoehto	Digitaalinen media
Ohjaajat	Lehtori Antti Laiho Yliopettaja Harri Airaksinen

Insinöörityön tarkoituksena oli luoda kiihtyvyysanturia ja eleohjausta hyödyntävä mobiilipeliprototyyppi. Työssä selvitettiin, miten hyvin liike- ja eleohjaus toimivat yhteistyössä mobiilipeleissä. Työssä tutustuttiin liike- ja eleohjauksen historiaan ja tekniikkaan sekä verrattiin sitä vaihtoehtoisiin ohjausmenetelmiin. Taustojen selvittämisellä pyrittiin löytämään mobiilipelien eri ohjaustapojen hyötyjä ja heikkouksia ja selvittämään pystyykö liike- ja eleohjaus parempiin tuloksiin. Tärkeinä lähteinä tutkimuksessa ja vaikuttajana prototyypin luomisessa olivat muut pelit.

Peli luotiin Android-puhelimilla toimivaksi Unityn pelimoottorilla. Tutkimuksessa tutustuttiin myös perinteisempään Android-ohjelmointiin liitännän luomisen muodossa, mutta liitännät jäivät prototyypin lopputuloksesta pois.

Peli hyödyntää kiihtyvyysanturia hahmon kääntämiseen kallistelemalla puhelinta oikealle tai vasemmalle. Pyyhkäisyeleillä hahmo voi syöksyä oikealle tai vasemmalle pyyhkäisysuunnan perusteella.

Valmis prototyyppi hyödynsi yksinkertaista eleohjausta kiihtyvyysanturilla toimivan liikeohjauksen lisäksi. Prototyypin testauksessa kokeiltiin myös vaihtoehtoja kosketuseleille ohjauksen yksinkertaistamiseksi. Vaihtoehtoinen ohjausmenetelmä hyödynsi nopeampia ja yksinkertaisempia eleitä ohjaamisen tasapainottamiseksi, mutta rajoitti jatkokehityksen suunnittelua. Prototyypin kehitys jatkuu vielä, ja tasapainottamisen ja toiminnallisuuden lisäksi rautalankamalli korvataan tulevaisuudessa graafisella ulkoasulla.

Avainsanat	kiihtyvyysanturi, liike, kosketuseleet, peli, mobiili, Unity

Author	Ville Raitio
Title	Motion and gesture control in a mobile game
Number of Pages	32 pages
Date	27 April 2016
Degree	Bachelor of Engineering
Degree Programme	Media Technology
Specialisation option	Digital Media
Instructors	Antti Laiho, Senior Lecturer Harri Airaksinen, Principal Lecturer

The goal of this study was to create a mobile game utilizing accelerometers and gesture control and see how well motion and gesture control work combined. The study looks into the history and technology of motion and gesture control and their alternatives. Research of the backgrounds on different control methods on the mobile platform brought answers to what their pros and cons were, and whether motion and gesture control could offer better solutions. Other games worked as important source materials and references for the game.

The game was built for Android phones using 3ds Max 2016 for 3D modelling and Unity 3D for game development. The use of Android plugins for Unity 3D was also considered, but left out of the final prototype.

The accelerometer is used to rotate the player by tilting the phone to the left or right. The player may also dash to the left or right by swiping to the preferred direction on the screen.

The final prototype utilized simple touch gestures combined with motion control via the accelerometer of smartphones. The study took note on alternative ways to use finger gestures to simplify the controls. The alternative ways simplified the controls at the cost of control design for future development. The prototype is still under development. In addition to the refining of controls and functionality, the wireframe look of the prototype will be replaced by a new graphical design.

Keywords	accelerometer, motion, gesture, mobile, game, Unity

Sisällys

1 Johdanto

Insinöörityössä tutustutaan liike- ja eleohjaamiseen mobiilipeleissä ja verrataan niiden vahvuuksia ja heikkouksia muihin ohjausmenetelmiin. Työn tavoitteena on luoda toimiva prototyyppi kiihtyvyysanturia ja kosketuseleitä hyödyntävälle pelille.

Epätavalliset ohjausmenetelmät ovat aina kiehtoneet peleissä, mutta ne ovat samalla olleet monien pettymysten ja turhautumisten aiheita. Nintendo Wii ja Xbox Kinect ovat saaneet paljon suosiota etenkin satunnaisten pelaajien keskuudessa, kun taas perinteisten pelien harrastajille liikeohjauksen epätarkkuus voi vaikuttaa ratkaisevasti pelikokemukseen. Kiihtyvyysanturilla toimivan ohjaimen heiluttaminen voi olla hauskaa, mutta perinteisten konsoli- ja tietokonepelien ohjauksessa käytetään mieluummin tavallisia peliohjaimia. Wii ja Kinect ovatkin saaneet juhlimispelien maineen, niin hyvässä kuin pahassa.

Konsoleiden liikeohjauksen suosion laantumisesta huolimatta liikeohjaukselle on paikkansa, etenkin mobiilimarkkinoilla. Älypuhelimilla pelatessa monissa peleissä häiritsevät joko näytölle simuloidut peliohjaimet tai tarve pitää sormia näytön tiellä. Monet pelit ovat ratkaisseet tämän ongelman siten, että pelin esineet ja olennot ovat itsessään paineltavia. Esimerkiksi Rovion Angry Birdsin käyttöliittymä ei vaadi juuri mitään ylimääräistä, sillä lingon venyttäminen toimii yhden sormen raahauseleellä. Vaikka kyseisten pelien suunnittelu onkin käytännöllistä ja visuaalisesti miellyttävää, kiihtyvyysanturit tuovat ongelmaan osittain paremman ratkaisun. Kiihtyvyysanturia hyödyntämällä sormia ei tarvitse pitää näytön edessä, eivätkä käyttöliittymän osat rajoita pelin näkökenttää. Todennäköisesti suurin syy kiihtyvyysantureiden käyttöön on nimenomaan näytön vapauttaminen tarpeettomista napeista. Vaikka kiihtyvyysanturit eivät kuitenkaan ole täydellinen ratkaisu epätarkkuutensa takia, jokaisen mobiilipelisuunnittelijan kannattaa tiedostaa niiden vahvuudet ja heikkoudet.

Kiihtyvyysantureiden ohella insinöörityössä tutkimuksen pääpaino on puhelimien kosketuseleissä ja siinä, miten niitä voi hyödyntää peleissä. Lähes jokainen mobiilisovellus hyödyntää kosketuseleitä jossakin muodossa, sillä jo tavallinen näpäytyskin on oma eleensä. Yleisellä tasolla eleistä puhuttaessa tarkoitetaankin usein monimutkaisempia ja epätavallisempia eleitä, kuten pyyhkäisyä tai pitkää painallusta. Kiihtyvyysanturi ja eleet ovat molemmat ohjausmenetelmiä, jotka eivät vaadi käyttöliittymältä nappeja toi-

miakseen. Tutkimus selvittää myös, sopivatko nämä ohjausmenetelmät yhteen vai hankaloittavatko ne ohjausta liikaa.

2 Liike-ohjauksen kehittyminen ja teknologiat

2.1 Liike- ja eleohjauksen historia

Liikehallinnan ja muiden todellisuutta jäljittelevien ohjausmenetelmien historian peleissä voidaan katsoa alkaneen jo 1970-luvulla Shooting Galleryllä Odyssey -konsolille. Pelin ideana oli lelukiväärillä ampua TV-ruudussa liikkuvaa valopistettä. Alkeellinen mutta kokeileva peli ei kuitenkaan saanut suurta suosiota ja jäi unohduksiin. (1.)

Todenmukaisen ohjauksen kehityksen keskipiste siirtyi kotikonsoleilta pelihalleihin, joissa sellaiset pelilaitteet kuin Segan vuonna 1976 julkaisema nyrkkeilyhanskoilla ohjattava Heavyweight Champ ja myöhemmin julkaistu KO Punch -nyrkkeilysäkki (kuva 1) saivat jalansijaa. Sega kehitti nyrkkeilypelien lisäksi myös muita kehon liikettä vaativia pelejä, kuten moottoripyöräpeli Hang-On vuodelta 1985. (2.)

Kuva 1. Segan KO Punch -nyrkkeilypelissä peliä ohjattiin fyysistä nyrkkeilysäkkiä lyömällä (3).

Segan vallatessa pelihalleja mullistavalla ohjauksella alkoi Nintendo kehittää liikeohjauslaitteita kotikonsoleille. Nintendo onnistui siinä, missä muut epäonnistuivat, ja toi

kotitalouksiin toimivia liikehallintapelejä, kuten Shooting Galleryn tapaisen Duck Huntin vuodelta 1986 ja 1986 Power Pad -liikuntamaton. (2.)

Liikehallinta kotikonsoleille otti kuitenkin 1980-luvun kehityksen jälkeen suuria taka-askeleita muutaman suuren epäonnistumisen jälkeen. Vuonna 1989 julkaistu Nintendo Power Glove suuresta markkinoinnista huolimatta epäonnistui epätarkan ohjattavuuden ja kömpelöiden television päälle asennettavien sensoreiden takia. Sega vastaavasti yritti julkaista kotitalouksiin sopivaa kehon liikkeitä tunnistavaa laitetta jonka ohjaus toimisi puhtaasti kehon liikkeillä. Vaikka laitteen voisi nähdä merkittävänä paaluna liikeohjauksessa, se oli kuitenkin Power Gloven tavoin epätarkka ja kömpelö. 1990-luvun aikana liikeohjausta ei juurikaan kehitetty, mahdollisesti 1980-luvun epäonnistuneiden laitteiden vuoksi. (2.)

Konsoleilla liikehallinnan nykytilan voi katsoa alkaneen vuonna 2006, kun Nintendo julkaisi liikeohjaukseen perustuvan Wii-kotikonsolin, joka toi vihdoin toimivan vaihtoehdon perinteiselle ohjaukselle kotikäyttöön. Ennen Wiitä pelaaminen koettiin epäsosiaalisena ja vaikeasti lähestyttävänä aktiviteettina, mutta Nintendo onnistui markkinoimaan konsolinsa perheille ja satunnaisemmille pelaajille. Wii tarjosi helposti ymmärrettäviä seurapelejä, joiden puute oli ollut suuri aukko videopelien kulttuurissa 1990- ja 2000-luvuilla. (4.)

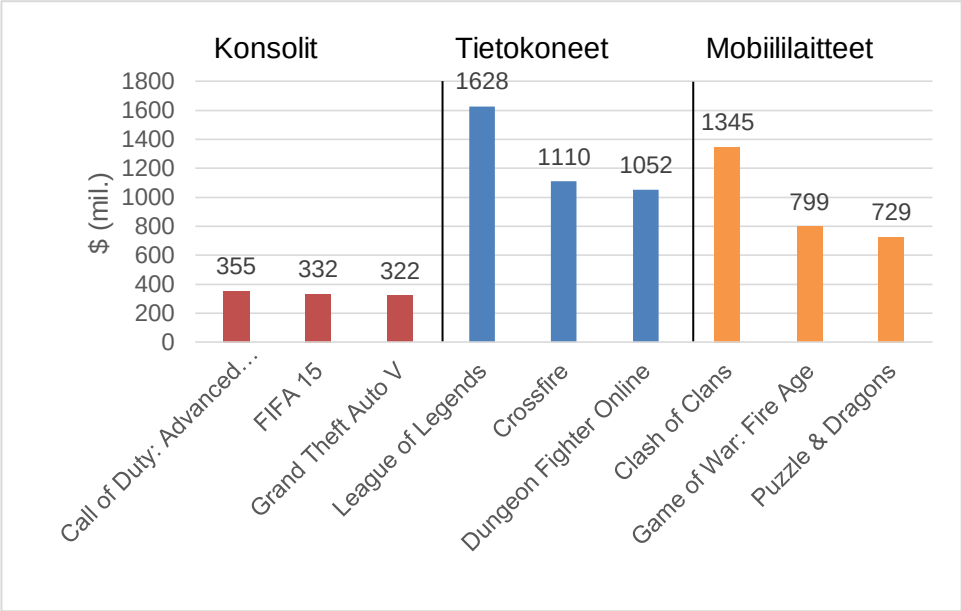
Wii ja sen seuraaja Wii U (2012) hyödyntävät ohjaimissa olevia kiihtyvyysantureita (2). Nintendo julkaisi Wiille myös erillisiä ohjaimia ja ohjaimien lisäosia, kuten useita muovisia laajennuksia, Wii MotionPlussan ja kuvan 2 Wii Fit -tasapainoalustan. Osa laajennuksista oli vain muovikappaleita, joiden tarkoitus on lisätä todenmukaisuutta ja mahdollisesti kiinnipitämisen asentoa. Esimerkiksi Wiin ohjauspyörä ei muuta ohjausta mitenkään, vaan se on lähinnä esteettinen kappale muovia. Wii MotionPlus paransi tavallisen Wiimoten tarkkuutta gyroskoopilla. Alkuperäinen Wii-ohjain käytti vain kiihtyvyysanturia, mutta gyroskooppilaajennus mahdollisti ohjaimen kääntelyn tulkitsemisen, siinä missä ainoastaan kiihtyvyysanturilla saatiin selville ohjaimen kiihtyvyys ja sijainti xyz-akseleilla. Laajennuksen suurin ongelma oli, että kaikki pelit eivät tukeneet sitä (5).

Kuva 2. Wii Fit tasapainoalusta (6).

Wii Fit -tasapainoalusta oli erillinen ohjain, jonka ohjaus mukaili esimerkiksi laskettelulaudan tasapainottelua. Wii Fit -ohjain toimi myös vaakana: se laski käyttäjän painoindeksin tämän syötettyä pituustietonsa laitteelle. (6.)

Toisin kuin Wii, Microsoftin Kinect (2010) pyrki seuraamaan pelaajan liikkeitä kameran avulla (7). Seurapelien lisäksi Kinect osoittautui toimivaksi ja edulliseksi liikkeenkaappaajaksi, jota 3D-animoijat voivat kotioloissa hyödyntää todenmukaisia liikkeitä tavoitellessaan (8). Wiin tavoin kiihtyvyysantureita on myös nykypäivän älypuhelimissa. Älypuhelimien yleistyminen on avannut Wiin ja Kinectin tavoin uusia portteja pelaamiselle, ja kiihtyvyysanturit ovat mahdollistaneet pelien ohjaamisen laitetta kääntelemällä (9).

Älypuhelinten sovellusten ja pelien myynti on kasvussa, ja niiden pelimarkkinat ovat jo edellä kotikonsoleita. Tietokoneidenkin tuottavimmat pelit ovat ilmaisia pelejä, jotka saavat suurimman osan tuotoistaan pelien sisäisistä ostoksista. Samaa markkinamallia hyödyntävät myös useimmat mobiilipelit, kun taas konsolilaitteiden käyttäjät ehkä vastahakoisemmin ostavat pelien sisäisiä tuotteita. Kuvassa 3 verrataan konsoleiden, tietokoneiden ja mobiililaitteiden kolmea myydyintä tuotetta. Kuvasta näkee, kuinka tietokoneiden ja mobiililaitteiden ilmaiset pelit johtavat markkinoita konsoleihin verrattuna.



Kuva 3. Vuoden 2015 kolme tuottavinta peliä konsoleilta, tietokoneilta ja mobiililaitteilta (10).

On kuitenkin otettava huomioon, että mikään myyvimmistä mobiilituotteista ei käytä liike- tai eleohjausta perinteisten ohjausmetodien sijaan. Vaikka liikeohjaus onkin kiehtova idea, ei sillä itsessään saavuta samanlaista myyntiä kuin esimerkiksi Clash of Clansin kaltaisilla peleillä.

Tampereen yliopiston tekemässä tutkimuksessa (11) jaettiin vuoden 2012 kesäkuussa eniten ladatut pelit ohjausmenetelmien mukaan (taulukko 1). Ohjausmenetelmät oli jaettu seitsemään eri tyyppiin:

- napautus
- peliohjaimen emulointi
- pyyhkäisy
- valikko
- kosketus
- kirjoittaminen
- liikesensori.

Napautuksella viitataan lyhyisiin kosketuksiin, kun taas kosketuksella tarkoitetaan pidempiä painalluksia (raahaamista). Pyyhkäisyllä tutkimuksessa viitattiin eleisiin. Tutki-

musjoukossa on verrattu 62 peliä, joista kutakin oli ladattu yli 10 miljoonaa kertaa Google Playsta. Taulukon vertailujoukossa puolestaan tutkittiin 20:tä suosituinta peliä Suomen latauslistoilta.

Taulukko 1. Suosituimpien mobiilipelien ohjausmenetelmät. (11)

	Tutkimusjoukko	**Vertailujoukko**
Kosketus	36 (59%)	8 (40%)
Liikesensori	11 (18%)	2 (10%)
Napautus	8 (13%)	8 (40%)
Pyyhkäisy	3 (5%)	8 (40%)
Peliohjaimen emulointi	3 (5%)	2 (10%)
Valikko	1 (2%)	1 (5%)
Kirjoittaminen	0 (0%)	1 (5%)

Tutkimus osoittaa, että liikesensoreilla toimivia pelejä on huomattava määrä Androidin suosituimpien pelien joukossa ja niille on kysyntää, joskin vertailujoukossa niitä oli vain kaksi. Huomattavaa on, että peliohjaimen emulointia hyödyntäviä pelejä oli vain kolme (2 vertailujoukosta), mistä voi tulkita kuluttajien haluavan mobiilipeleiltä puhelimelle luonnollisesti soveltuvia ohjausmenetelmiä perinteisten ohjaimien emuloinnin sijaan. (11.)

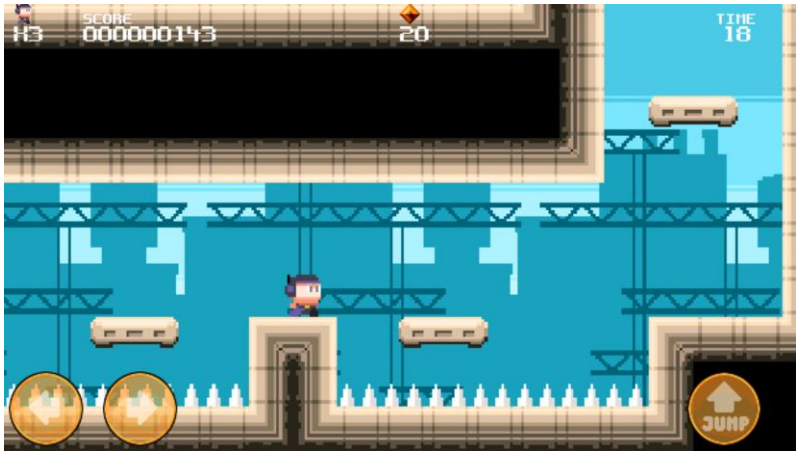
2.2 Perinteinen peliohjaus

Perinteisesti pelejä on ohjattu peliohjaimilla, joissa tyypillisesti on yksi tai useampia ohjaustikkuja ja/tai nuolinäppäimistö sekä muihin toimintoihin tarvittavia nappeja. Kuvan 4 Atarin peliohjain on yksi varhaisimpia modernien peliohjaimien malleja.

Kuva 4. Atari -peliohjain on yksinkertaisuudessaan hyvin käytännöllinen.

Atarin ohjain toimii mainiona esimerkkinä peliohjaimesta, sillä sen yksinkertaisuutta on hyvä rinnastaa mobiilipelimarkkinoihin. Yksi menestyneen mobiilipelin tärkeimpiä ominaisuuksia on helposti hallittavuus, joten ohjainta simuloidessa on hyvä miettiä, kuinka pelin saa pidettyä yksinkertaisena kahdelle peukalolle.

Vaikka puhelimille soveltuvia ohjaimia nykyään valmistetaankin, antavat monet pelit vaihtoehdon pelata näytössä näkyvillä ohjaimilla. Kuvassa 5 on ruudunkaappaus Meganoid-pelistä, jossa pelin päälle on piirretty napit, joita painamalla hahmoa ohjataan (12).



Kuva 5. Meganoidissa peliä voi ohjata näytölle piirretyillä napeilla (12).

Nappien etuna on tarkkuus, mutta ne voivat herkästi tehdä pelin ulkoasusta epämiellyttävän. Käyttöliittymä ei saisi täyttyä napeista, mutta samalla nappien on oltava käytännöllisen kokoisia, sillä pienet napit voivat hankaloittaa ohjausta. Suurista napeista voi joskus olla hyötyä, kuten kuvan 6 Street Fighter IV -älypuhelinversiossa. Pelinkehittäjän on tärkeää ymmärtää, onko tarkoitus panostaa visuaaliseen ilmeeseen vai pelattavuuteen.

Kuva 6. Street Fighter IV iPhonelle (13).

Vaikka suurien nappien käyttöliittymä ei näytä houkuttavalta, tarkkuutta vaativissa peleissä on usein järkevämpää käyttää ohjaimia simuloivia nappeja intuitiivisten kosketustoimintojen sijasta. Napit voivat kuitenkin peittää olennaisia osia ruuduista ja aiheuttaa vaikeustason nostoa tahattomasti, samalla kun pelin visuaalinen ilme kärsii. (14, s. 278.)

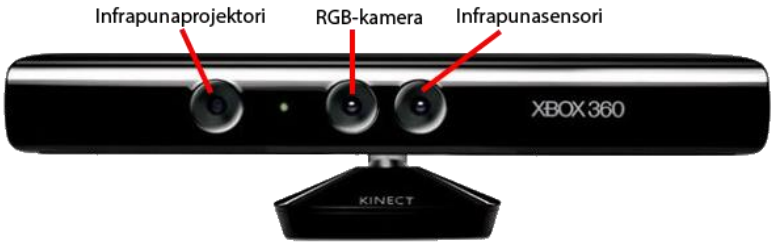
2.3 Liikkeenkaappaus

Vaikka liikkeenkaappausta ei voi pitää varteenotettavana tapana ohjata sovelluksia ja pelejä mobiililaitteilla, sen merkitys yleisesti peliteknologiassa on hyvin merkittävä.

Liikkeenkaappaus on pitkään ollut peliteknologiassa animointia helpottava tekniikka, jolla 3D-hahmoille saa todenmukaiset liikkeet seuraamalla oikean näyttelijän liikkeitä. Liikkeenkaappaus on kehittynyt paljon viime vuosien aikana, ja siinä käytetään monia eri tekniikoita. Perinteinen metodi liikkeenkaappaukselle on seurata näyttelijän vartaloon kiinnitettyjä merkkipisteitä (kuva 7). (15.)

Kuva 7. Näyttelijän liike kaapataan seuraamalla puvun valkoisia merkkejä (15).

Nykyteknologia mahdollistaa myös liikkeenkaappauksen ilman pukuja tai muita tarvikkeita. Esimerkiksi Microsoftin Kinect kykenee tunnistamaan ihmisten liikkeitä huoneesta ilman erillisiä pukuja tai tarvikkeita. Tavallisen RGB-kameran lisäksi Kinectissä on infrapunaprojektori ja -sensori (kuva 8). Infrapunaa Kinect käyttää saadakseen syvyyskuvaa, joka antaa tarkemman liiketunnistuksen kuin tavallinen RGB-kamera antaisi. Kinect projisoi infrapunan huoneeseen, josta se heijastuu infrapunasensoriin (16).



Kuva 8. Xbox 360 Kinect toimii infrapunateknologialla (17).

Älypuhelimille liikkeenkaappaus ei ole kovin intuitiivinen tapa ohjaukseen, sillä käyttäjän kädet ovat tavallisesti varattuina laitteen pitelemiseen. Liikkeenkaappaus voisi kuitenkin toimia kasvojen liikkeiden ja eleiden tunnistamisessa. Esimerkiksi silmien liikettä seuraava tekniikka voisi tuoda mahdollisuuksia pelialalle, mutta tekniikka on toistaiseksi liian kallista kuluttajamarkkinoille. Tekniikka on kuitenkin kehittymässä ja sen kustannukset laskussa (14, s. 145). Jää nähtäväksi voiko silmäelehallinta kehittyä niin tarkaksi, että se voisi kilpailla muiden ohjausmenetelmien kanssa.

2.4 Eleet

Mobiililaitteiden sovellusten kehityksessä eleillä viitataan sormen liikkeillä suoritettaviin komentoihin. Eleitä voi erottaa toisistaan

- painallukseen käytetyllä ajalla
- sormien määrällä
- sormien liikkeen suunnalla
- liikkeiden nopeudella
- liikkeen paineella.

Eleitä käytetään sovelluksissa hyvin yksinkertaisiin ja arkipäiväisiin toimintoihin, kuten kohdentamiseen ja pyörittämiseen. Useissa sovelluksissa eleillä voi myös käynnistää sovelluksen valikon tai siirtyä näkymästä toiseen pyyhkäisemällä sivulle. Monikoskettuksen avulla sovellusten käyttöliittymät ovat monipuolistuneet, ja käyttöliittymien suunnittelussa nappien merkitys on vähentynyt (18).

Painalluseleiden lisäksi uusimmat laitteet voivat myös rekisteröidä ilmaeleitä. Ilmaeleet ovat toistaiseksi kuitenkin niin alkeellisessa vaiheessa, että niitä ei käytetä sovelluskehityksessä.

Mobiilipeleissä eleitä on käytetty jo useissa peleissä, esimerkkinä Jujubee S.A:n Spellcrafter, jossa pelaaja pystyy käyttämään erilaisia loitsuja piirtämällä niiden riimut puhelimen näytölle (kuva 9).

Kuva 9. Spellcrafter -pelissä käyttäjä voi loihtia piirtämällä tiettyjä kuvioita puhelimen näytölle (19).

Yksi menestyneimpiä eleohjauksen pelejä on Chair Entertaiment Groupin vuodesta 2010 eteenpäin julkaisema Infinity Blade -pelisarja, jossa pelaaja pyyhkäisee näytöllä pyydetyillä tavoilla lyödäkseen tai torjuakseen vihollisia tai ylittääkseen esteitä (20). Tällaista ohjausmenetelmää kutsutaan ”lyhyen ajan tapahtumaksi” (engl. quick time event), jossa pelaajan pitää suorittaa pyydetty toiminto lyhyellä varoitusajalla.

2.5 GPS ja kiihtyvyysanturit

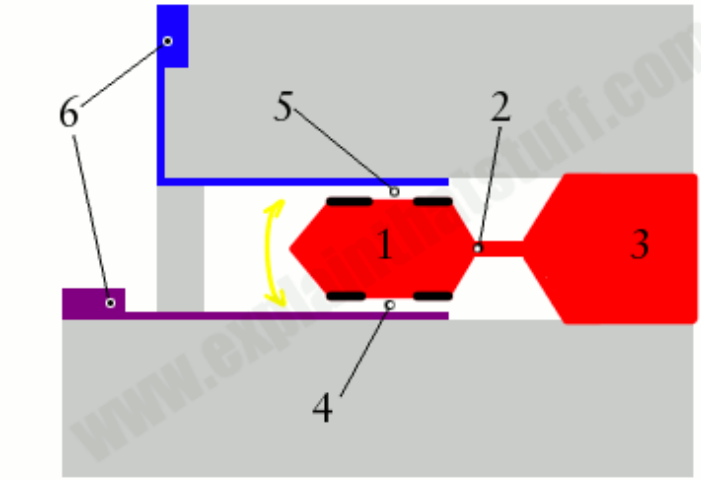
GPS-paikannuksella toimivia pelejä on ollut mobiililaitteiden markkinoilla jo jonkin aikaa. Sijaintipohjaiset pelit poikkeavat luonteeltaan merkittävästi muista peleistä siinä, että GPS-ohjauksella mitattavan liikkeen suuruus on aivan eri mittakaavassa kuin muilla menetelmillä eikä pientarkkuudella ole niin suurta merkitystä. (14, s. 135–136)

Kiihtyvyysantureilla mitataan nimensä mukaisesti laitteen kiihtyvyyttä, mutta samalla niitä voi käyttää esimerkiksi laitteen asennon muutoksen mittaamiseen. Kiihtyvyysantureita käytetään nykypäivänä paljon tutkimuksessa, mutta myös arkipäivän laitteistoissa. Autoissa kiihtyvyysanturit voivat ennakoida törmäyksiä paremmin, kun taas kannettavissa tietokoneissa anturit pyrkivät estämään kiintolevyjen vahingoittumista tietokoneen pudotessa (21).

Peliteollisuudessa kiihtyvyysanturit yleistyivät Nintendo Wiin julkaisun myötä, ja nykyään niitä hyödynnetään paljon mobiilipelien kehityksessä. Lähes jokaisessa nykyään julkaistavassa älypuhelimessa on kiihtyvyysanturi. Kiihtyvyysanturin avulla voidaan esimerkiksi katkaista saapuva puhelu yksinkertaisesti kääntämällä puhelin nurinpäin.

Yksinkertaisimmatkin sovellukset hyödyntävät kiihtyvyysanturia vaihtamalla näkymää pystyasennosta vaaka-asentoon käyttäjän kääntäessä puhelinta. (21.)

Tavallisesti kiihtyvyysanturi vaatii toimiakseen jonkin liikkuvan osan, jossa on riittävästi massaa. Luonnollisesti nykypäivän kapeisiin älypuhelimiin ei ole mahdollista sovittaa painavia värähteleviä komponentteja, joten ne käyttävät niin sanottuja puolijohdekiihtyvyysantureita. Puolijohdekiihtyvyysanturit koostuvat pienistä komponenteista, jotka on kiinnitetty silikoninpalaseen. Kuvassa 10 ja sen jälkeisessä tekstissä on selitetty puolijohdekiihtyvyysanturin osat ja niiden toiminnat.



Kuva 10. Puolijohdekiihtyvyysanturin osat ja toiminta (22).

Kuvassa 10 näkyvän anturin keskellä oleva punainen elektrodi (1) tekee pientä liikettä kiihtyvyysanturia käänneltäessä. Tätä elektrodia tukee kannatuspalkki (2) joka on riittävän joustava liikkeen mahdollistamiseen. Kuvan kolmas osa on sähköjohdin, jolla edellä mainitut elektrodi ja kannatuspalkki saadaan viritettyä mikrosirun ulkopuolelle. Elektrodin ylä- ja alapuolella olevat violetti ja sininen elektrodi (4 ja 5) mittaavat niiden ja punaisen elektrodin välisen etäisyyden muutosta. Punaisen elektrodin mustat osat estävät törmäämisen muihin elektrodeihin nopeissa liikkeissä. Kuvan kuudennessa osassa ovat punaisen ja violetin elektrodin liitännät sirun ulkopuolelle, josta ne saadaan johdettua eteenpäin laitteeseen. (22.)

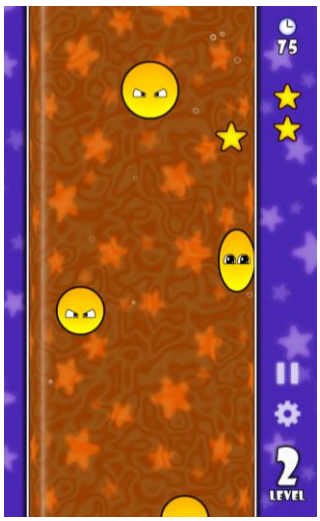
Myös uusimmat puhelimien lisäosina markkinoidut älykellot, kuten Apple Watch, sisältävät gyroskooppeja ja kiihtyvyysantureita (23). Hyvin toteutettuna rannekellon käyttö peliohjaimena voisi avata uusia mahdollisuuksia mobiilipelimarkkinoille, mutta mikäli

toteutus on yleisesti huonoa, mobiilipelaajat luultavasti pitäytyvät puhelimien omissa ohjausmenetelmissä.

3 Prototyypin suunnittelu

3.1 Markkinoilla olevat tuotteet

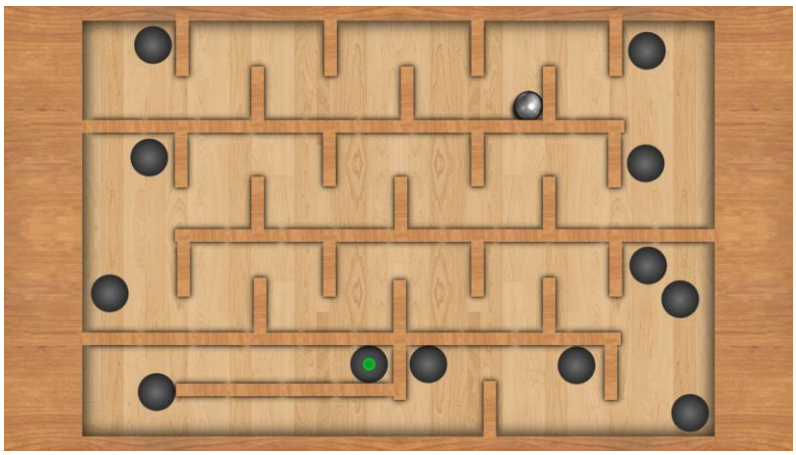
Pelin prototyypin suunnittelun ensimmäinen vaihe on hakea vaikutteita muista peleistä ja tutkia, mitkä asiat toimivat ja mitkä ovat kehityksessä haastavampia. Yksi arkityyppinen esimerkki kiihtyvyysanturia hyödyntävästä pelistä on cmyksoftin Lava Bubble Adventure (kuva 11), jossa pyöreällä hahmolla kerätään tähtiä ja yritetään päästä maaliin väistelemällä samalla vihollisia ja esteitä. Pelin ohjaus toimii yksinomaan kiihtyvyysanturia hyödyntäen. (24.)



Kuva 11. Lava Bubble Adventure –peli (24).

Ohjaaminen pelissä on tarkkaa, ja asetuksista on mahdollista säätää ohjauksen herkkyyttä. Herkkyyden säätömahdollisuus on olennainen, koska eri laitteiden kiihtyvyysanturien tarkkuus voi vaihdella. Pelin suurin kompastuskivi on kuitenkin laitteen asennon määrittämisessä, sillä pelin lähtöasetelma vaatii laitteelta noin 30 asteen kulman maan tasoon nähden.

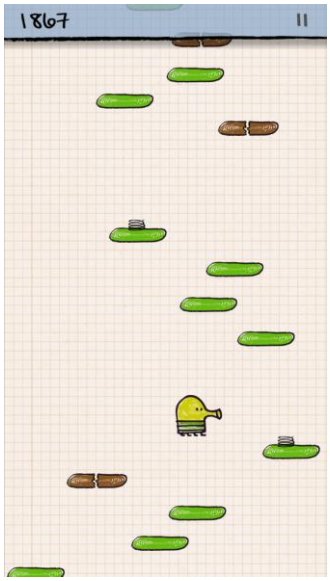
Samankaltaisista ongelmista kärsivät monet muut pelit, kuten Teeter Pro (kuva 12). Teeterissä ohjataan kuulaa perinteisen puulabyrintin tyyppisten sokkeloiden läpi vihreällä pisteellä merkittyyn maaliin. (25)



Kuva 12. Teeter Pro –peli (25).

Toisin kuin Lava Bubble Adventuressa, Teeterissä puhelinta pidetään vaaka-asennossa pystyasennon sijaan. Ohjaamisen tarkkuus on Bubble Adventuren tavoin miellyttävää, mutta kärsii samoin pelaajan laitteen asennon rajoittamisesta.

Kuvan 13 Doodle Jump Lima Sky LLC:ltä on hyvä esimerkki luovasta kiihtyvyysanturin käytöstä pelin ohjauksessa. Hahmo hyppii tasaisesti itsekseen, mutta pelaaja voi muuttaa hypyn suuntaa kääntämällä puhelinta oikealle tai vasemmalle. Lisäksi pelaaja voi ampua napauttamalla näyttöä. Ammuksen suunta määräytyy napautuksen kohdalla x-akselilla. (26.)



Kuva 13. Doodle Jump –peli (26).

Mobiilipelimarkkinat ovat täynnä autosimulaattoripelejä, joissa ohjauspyörää mukaillaan kiihtyvyysanturilla. Kuvassa 14 on ruudunkaappaus Incredible Race Car Simulatorista (GT Race Games). Pelissä ei ole päämäärää, joten peli on nimensä mukaisesti simulaatiopeli tai niin sanottu hiekkalaatikkopeli, jossa pelaajalla on vapaat kädet tehdä mitä tahansa pelin fysiikan ja ohjauksen rajoissa. (27.)

Kuva 14. Incredible Race Care Simulator –peli (27).

Toisin kuin aiemmissa esimerkeissä, IRCS käyttää kiihtyvyysanturin lisäksi ohjaukseen nappeja (kuten kuvassa näkyvät kaasu, pakki ja turbo). Vaikka napit ovat tarpeellisia, niiden painaminen ohjauksen aikana voi häiritä käännösten tarkkuutta. Nappien lisäksi ohjaus tuntui hieman liukuvalta paikoitellen, sillä kiihtyvyysanturin nollapistettä on välillä hankala löytää.

Nykyään yhä useammat pelit antavat käyttäjälle vaihtoehdon käyttää kiihtyvyysanturia perinteisen peliohjainsimuloinnin sijasta, kuten aikaisemmin esimerkkinä ollut Meganoid. On kuitenkin huomioitavaa, että tarkkuutta vaativissa sivultapäin kuvatuissa tasohyppelyissä ongelmia tuottaa etenkin pysähtyminen. Pysäyttääkseen hahmon pelaajan tulee kääntää puhelinta niin että kiihtyvyysanturi tulkitsisi asennon nollatilanteeksi. Tämä tarkoittaa, että pelaajan pyrkiessä pysähtymään liian pienet laitteen käännökset pitävät hahmon liikkeessä, kun taas liian suuret käännökset voivat laittaa hahmon juoksemaan vastakkaiseen suuntaan. Tämä on erityisen epämukavaa, jos peli on täynnä kuolettavia pudotuksia tai muita väisteltäviä esteitä.

3.2 Laitteiston haasteet pelikehityksessä

Peliä älypuhelimelle suunnitellessa ja kehittäessä on otettava huomioon monta asiaa käyttäjän kannalta, kuten

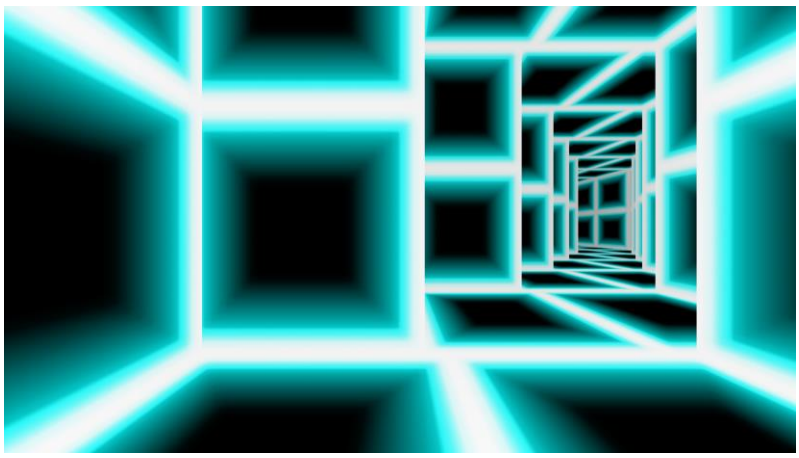
- kohdeyleisö ja -laitteet
- sovelluksen koko ja suorituskyky
- näytön koko
- ohjattavuus
- peliaika
- yksinkertaisuus ja lähestyttävyys
- muuttuva ympäristö.

Kohdelaitteisto on olennainen kysymys, sillä puhelimien suorituskyvyt poikkeavat toisistaan. Jotta peli toimisi vanhemmilla ja halvemmilla laitteilla, pitäisi sen olla riittävän kevyt ja koodikirjastoltaan yhteensopiva vanhojen käyttöliittymien kanssa. Pelin suorituskykyyn vaikuttavat monet yksityiskohdat, kuten graafinen ulkoasu tai verkkoyhteyden käyttö. Sovelluksille ja peleille suositellaan pientä kokoa, jotta käyttäjien olisi järkevää pitää niitä puhelimilla. Kuitenkin muutamat raskaat pelit ovat hyvin suosittuja pelimarkkinoilla, kuten Blizzardin Hearthstone, joka suuresta koostaan huolimatta on saanut yli 10 miljoonaa latausta Android -markkinoilla (28). Käyttöliittymän suunnittelussa näytön koko on merkittävä tekijä. Elleivät kohdelaitteena ole ainoastaan tabletit, käyttöliittymän nappien ja muiden tärkeiden visuaalisten osien on oltava riittävän suuria, jotta käyttäjä pystyy huomioimaan tai käyttämään niitä luotettavasti. Tämä on yksi syy, miksi mallin-

nettuja ohjaimia on hyvä välttää, mikäli mahdollista. Eleohjauksella ja kiihtyvyysanturilla toimivat pelit voivat kärsiä tarkkuudesta mutta voivat vapauttaa pelin näytöstä todella paljon tilaa ja avaavat uusia mahdollisuuksia visuaaliselle suunnittelulle. (14, s. 275–279.)

Peliajalla tarkoitetaan pelaajan yksittäisiin peli-istuntoihin käytettyä aikaa. Älypuhelinpelaajat pelaavat usein lyhyitä hetkiä, esimerkiksi odotellessaan bussia, joten pelin luomisessa tulee suunnitella lyhyitä tasoja tai tallennuspisteitä lyhyin aikavälein, jotta pelissä olisi mahdollista edetä käyttämättä pelaamiseen muutamaa minuuttia enempää. Peliajan lyhyyden vuoksi pelin rungon on oltava yksinkertainen ja helposti käsitettävissä. Suuret ja monimutkaiset pelit, jotka ovat suosittuja konsoleilla ja tietokoneilla, ovat suurimmalle osalle puhelinkäyttäjistä liian massiivisia satunnaiseen pelailuun, eikä niihin ehdi uppoutua lyhyillä istunnoilla. Viimeisenä luetelluista huomioista kehityksessä on hyvä huomioida mobiili peliympäristö. Mobiililla peliympäristöllä tarkoitetaan pelin ulkopuolisia tekijöitä, kuten sää, liikenne ja valo. Vaikka näitä muuttujia on vaikeaa ottaa huomioon pelin kehityksessä, ne on silti hyvä tiedostaa. Esimerkiksi jos pelissä on tärkeää kuunnella ääniä, liikenteen ja luonnon äänet voivat häiritä pelikokemusta ja rajoittaa pelattavuutta vain hiljaisiin olosuhteisiin. Mobiilipeli on siis jo lähtöpisteestä hyvä suunnitella sellaiseksi, että se soveltuu eri tilanteisiin. (14, s. 279–280.)

Insinöörityön peliprototyyppi on tyylilajiltaan ”loputon juoksija”, jossa pelaajan ohjaama hahmo liikkuu jatkuvasti eteenpäin ja pysähtyy vain epäonnistumisiin ja maaleihin. Android-markkinoilla on jo useita loputtomia juoksijoita, joista kuitenkin iso osa on kaksiulotteisia. Vertailun vuoksi kokeilin Kapo Gamesin Death Run 3D:tä (kuva 15), jossa pelaaja ohjaa itseään ahtaassa tilassa eteenpäin, väistellen eteen tulevia esteitä (29).



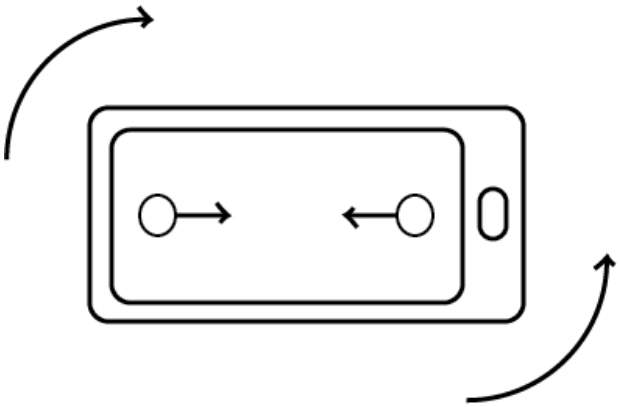
Kuva 15. Death Run 3D –peli (29).

Prototyypin ohjauksen pääpainona hyödynnetään puhelimen kiihtyvyysanturia, mikä mahdollistaa ohjaustyylin jossa puhelinta kääntämällä auton ohjaustangon tavoin hahmo kääntyy samalla, kun se jatkaa matkaa eteenpäin.

Kiihtyvyysanturin lisäksi pelissä voi yksinkertaisilla elekomennoilla vaikuttaa hahmon liikkumiseen. Eleohjausta suunnitellessa on useita asioita, jotka on hyvä ottaa huomioon. Vaikka eleistä voi olla hyötyä pelin ohjaamisessa, niihin ei kannata kiintyä liikaa. Eleiden käytön helpottamiseksi on hyvä huomioida ainakin seuraavat asiat:

- laitteen asento
- eleiden pituus
- ymmärrettävyys ja intuitiivisuus
- yksinkertaisuus.

Laitteen asennon huomioiminen pelin suunnittelussa on erittäin merkittävää, sillä se vaikuttaa hyvin paljon eri eleiden käytettävyyteen ja mukavuuteen. Mikäli laitetta on tarkoitus pitää kahdella kädellä, kannattaa eleet suunnitella niin, että ne voi toteuttaa pelkästään peukaloilla. Kuva 16 havainnollistaa pelissä käytettyjä ohjausmenetelmiä.



Kuva 16. Pelin ohjaukseen tarvittavat liikkeet ja eleet.

Eleiden pituudessa otetaan huomioon, kuinka pitkiä painalluksia pelissä on järkevää käyttää. Pidemmät ja liikettä vaativat kosketukset voivat vaikuttaa asennon mukavuuteen ja kuvan tukkimiseen kädellä.

Eleiden on oltava ymmärrettäviä ja intuitiivisia. Tämä tarkoittaa sitä, että käyttäjän on joko itse tajuttava käytettävissä olevat eleet tai ne tulee selittää jollain yksinkertaisella

tavalla. Mikäli ele ei tunnu ohjauksessa luonnolliselta, on parempi miettiä muuta ratkaisua.

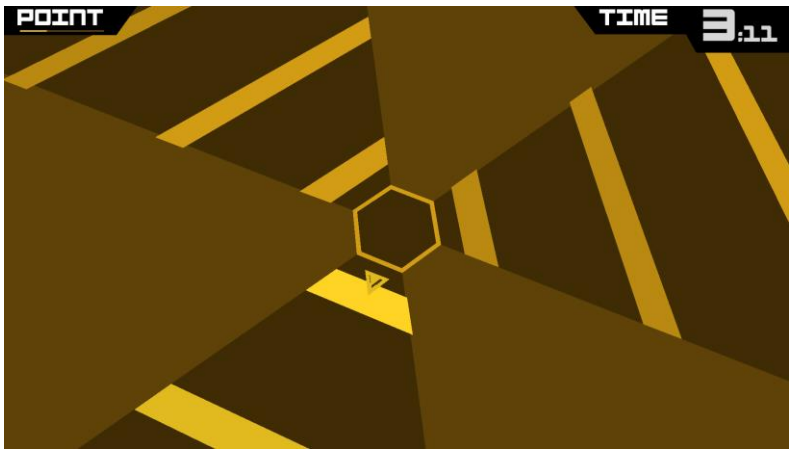
Älypuhelinten sovellusmarkkinoilla ohjauksen yksinkertaisuus on monille tärkeimpiä tekijöitä. Vaikeat ja monimutkaiset pelit menestyvät paremmin tietokoneilla ja konsoleilla, joiden ohjaimet on suunniteltu suorittamaan monimutkaisia toimintoja tarkasti ja vähällä vaivalla. (30.)

3.3 Pelitasojen kehitys

Peliin on hyvä suunnitella erilaisia esteitä ja haasteita. Vaikutteita on hyvä hakea muista peleistä, joten tutkiessani muita pelejä kiinnitin erityistä huomiota tasojen suunnitteluun.

Nintendon rallipeli Mario Kart on hyvä pelitasojen kehityksen vaikuttajana. Mario Kartin Rainbow Road -taso on tunnettu vaikeustasostaan ja seinien puutteesta. Rainbow Road on hyvä esimerkki tasosuunnittelusta, jossa vaikeustason pääpaino on seinien puute. Putoamisen mahdollisuuden lisäksi Mario Kartissa pelaajia vaikeuttavat viholliset, äkkinäiset seinät ja muut esteet (31). Pelaajia myös helpottavat maastossa olevat hyppyrit ja nopeuttajat. Prototyypissä nopeuttajat vaikeuttaisivat huomattavasti hahmon ohjattavuutta, mutta ne mahdollistaisivat nopeamman suorituksen jolla pelaaja saisi paremmat pisteet radasta.

Aiemmin mainittu Death Run 3D sisälsi vain seiniä, joihin törmäämällä pelin häviää. Peli muistuttaa visuaaliselta ilmeeltään kuvan 17 Super Hexagonia, jossa pienellä nuolella pyritään väistämään seiniä ja liikkumaan nopeasti muuttuvassa pyörivässä ympäristössä. (29; 32.)



Kuva 17. Super Hexagon –peli (32).

Super Hexagon ja Death Run 3D ovat molemmat esimerkkejä peleistä, joissa minimalistisella tasosuunnittelulla voi luoda todella haastavia kenttiä. Super Hexagon hyödyntää myös toimivasti vaikeustason nostamista kierrättämällä edellisiä tasoja suuremmalla nopeudella. (29; 32.)

4 Abyss Runnerin toteutus

Pelin nimi Abyss Runner viittaa pelin radan ympäröivään tyhjyyteen toisin sanoen pohjattomaan kuiluun. Runner-loppuosa viittaa myös hyvin vahvasti pelilajiin eli loputtomaan juoksijaan (engl. endless runner). Pelin tavoite on päästä maaliin törmäämättä esteisiin tai putoamatta tyhjyyteen.

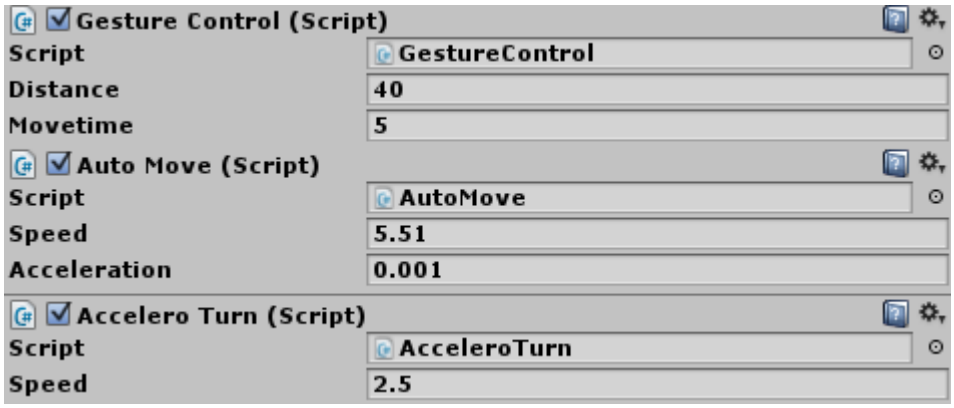
4.1 Hahmon ohjaaminen

Abyss Runnerissa hahmo liikkuu rataa pitkin, kunnes putoaa tai törmää esteeseen. Hahmoa käännetään kääntelemällä puhelinta kuvan osoittamalla tavalla. Kääntyminen on toteutettu Unityssä acceleration-muuttujaa hyödyntäen. Muuttujaa kutsumalla voi mitata puhelimen kääntymistä kolmiulotteisessa ympäristössä, mutta koska pelissä käännytään ainoastaan sivuille, tarvitaan muuttujasta ainoastaan arvo X-akselin muutokselle. (33.)

Hahmo voi myös suorittaa syöksyliikkeen oikealle tai vasemmalle. Liike tapahtuu pyyhkäisemällä ruudulla vasemmalta oikealle tai päinvastoin. Liikeen tunnistus tapahtuu Unityssä yksinkertaisesti kutsumalla GetTouch-metodia, joka palauttaa muun muassa

sovellukselle tarpeelliset sormen koordinaatit sekä eleen vaiheen. Tärkeimmät ominaisuudet pelin kannalta ovat painalluksen alun ja lopun X-koordinaatit. Yksinkertaisen pyyhkäisyeleen saa vertaamalla näitä kahta pistettä keskenään. Hienovaraisempiin eleisiin on olennaista seurata etenkin liikkeiden nopeutta. (34)

Unityssä pelin kehittämistä ja hienosäätöä helpottaa muuttujien muokattavuus työtilassa, sillä esimerkiksi hahmon alkunopeuden ja kiihtyvyyden arvoa ei tarvitse määrittää koodissa, mikäli ne on asetettu julkisiksi muuttujiksi (kuva 18).



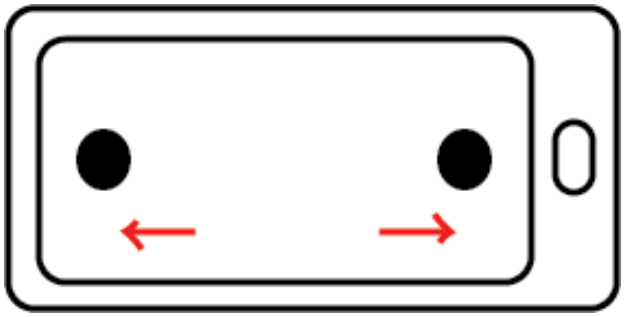
Kuva 18. Hahmon liikkumiseen käytettyjä muuttujia voi hienosäätää koskematta koodiin.

”Gesture Control” -koodin ”Distance” määrittää, kuinka paljon hahmo liikkuu jokaista kuvaa kohden (n. 60 kuvaa sekunnissa) ja ”Movetime” määrittää, kuinka monen kuvan aikana liike tapahtuu. Kuvan esimerkissä yksi liike kestää siis noin 1/12 sekuntia, jonka aikana hahmo liikkuu 200 Unityn yksikköä haluttuun suuntaan. ”Auto Move” liikuttaa hahmoa eteenpäin. Kuvan esimerkissä eteenpäin liikkumisen nopeus (”Speed”) kasvaa joka kuvalla yhdellä yksikön tuhannesosalla; esim. sekunnissa nopeus on kasvanut kuudella sadasosalla 5,57:een. ”Accelero Turn” käsittelee kiihtyvyysanturin liikkeet. Kiihtyvyysanturin herkkyyttä säädetään ”Speed”-muuttujalla, jota nostamalla kiihtyvyysanturin hallinta muuttuu herkemmäksi.

Abyss Runner vaikeutuu tasaisesti jo pelkällä juoksunopeuden kasvattamisella. Tasainen luonnollinen vaikeustason nostaminen helpottaa pelin vaikeustason suunnittelua, sillä pelkästään pelaamalla saman ruudun läpi eri nopeuksilla pelaajan reaktiokyky ja ohjaustarkkuus joutuvat koetuksille suurissa nopeuksissa. Myös kääntyvyyden herkkyyttä kiihdyttämällä peliä voisi vaikeuttaa, mutta liian herkkä ohjaus voi tehdä pelin vaikeudesta liian turhauttavan käyttäjälle. Tasaisen kiihtymisen sijaan hahmon nopeutta voisi nostaa porrastetusti. Porrastettu kiihtyminen mahdollistaisi vaikeustason nos-

tamisen esimerkiksi estesarjan jälkeen, mikä antaa pelaajalle käsityksen siitä, että vaikeustaso oikeasti vaihtuu.

Super Hexagonin ohjausmenetelmää mukaillen eleohjauksen voisi yksinkertaistaa käyttämällä pyyhkäisyjen sijasta vain painalluksia. Tässä vaihtoehtoisessa ohjaustavassa peli vertaisi painalluksen sijaintia laitteen näytön keskipisteeseen, jonka perusteella syöksyliikkeen suunta olisi siihen suuntaan, millä puolella sormen painallus on. Vaihtoehtoinen ohjaustapa on havainnollistettu kuvassa 19. Vaihtoehtoinen painallusohjaus yksinkertaistaa painalluksia ja helpottaa ohjausta käyttäjän näkökulmasta.



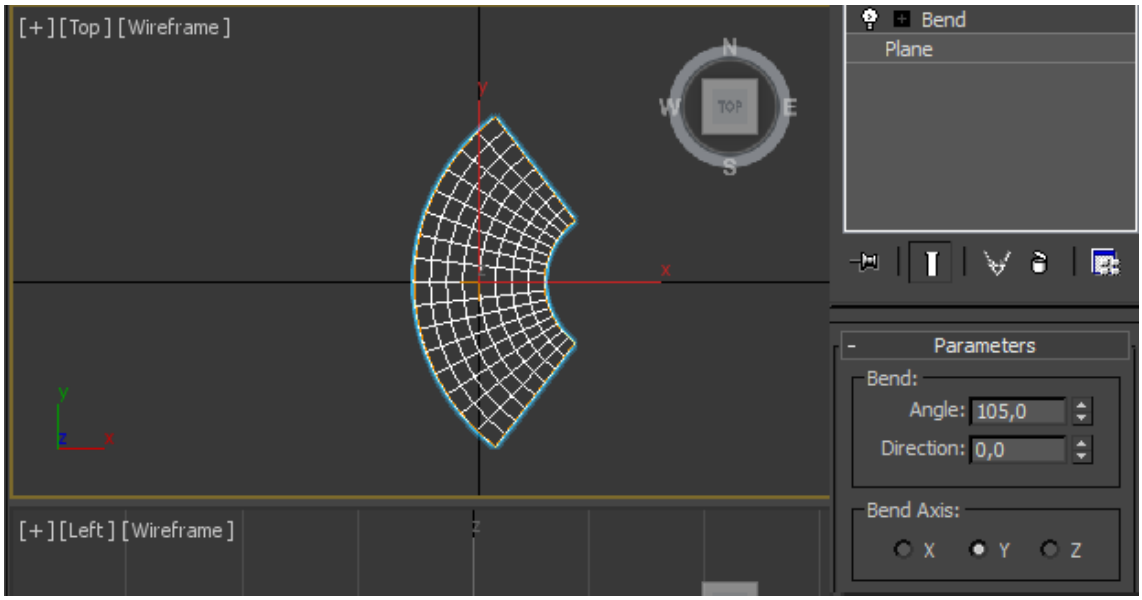
Kuva 19. Yksinkertaistetussa eleohjauksessa yksittäiset painallukset riittävät syöksyliikkeisiin.

Alkuperäisessä ohjauksessa on kuitenkin omat puolensa. Siinä missä painallustekniikka on pieniliikkeisempää kuin pyyhkäisytekniikka, se sitoo pelaajan käyttämään kummankin käden sormia, kun pyyhkäisytekniikka taas antaa pelaajalle mahdollisuuden käyttää yhtä peukaloa ohjaamiseen. Pyyhkäisytekniikkaa on myös mahdollista laajentaa tarkkailemalla pyyhkäisyjen nopeutta ja pituutta, mikä mahdollistaa eripituisia syöksyliikkeitä. Napautuksen voi myös pyyhkäisytekniikassa säilyttää tulevaisuuden kehityksessä muita komentoja varten, kuten esimerkiksi hyppyyn.

4.2 Tasosuunnittelu

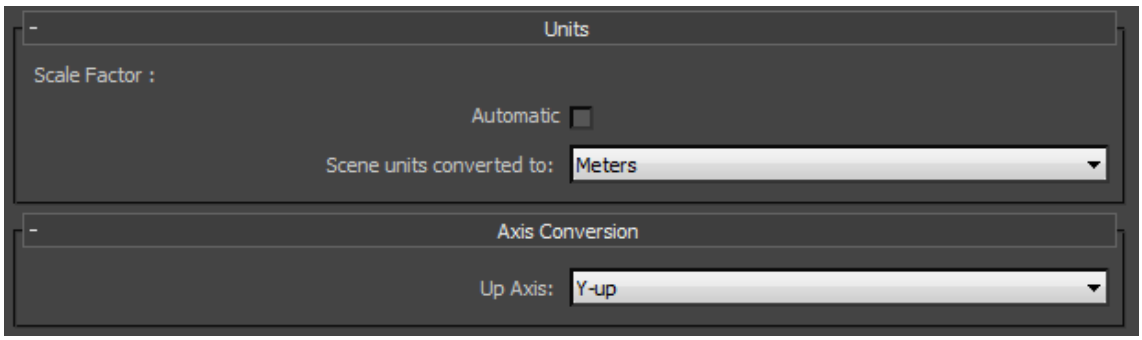
Pelin tasojen kehityksessä käytettiin Autodeskin 3ds Max 2016 -ohjelmaa, jolla muotoilin suoria ja käyriä ratapalasia. Tasoja rakentaessa on käytännöllistä käyttää monistettavia palasia kokonaisten kenttien muotoilun sijasta. Osista koottua tasoa on myös helpompi muokata testauksen jälkeen, esimerkiksi jos vaikeustaso ei miellytä.

Koska tarkoitus oli tehdä pelille runko, jossa ei ole tarkempaa 3D-mallinnusta, maasto muodostettiin vain tasaisista pinnoista. Mutkat luotiin käyttämällä Maxin ”Bend”-ominaisuutta (kuva 20). Jotta taivuttelu toimisi, on syytä tarkistaa, että mallissa on tarpeeksi segmenttejä. 4 metrin pituisessa ja 2 metrin levyisessä tasossa käytettiin 16 x 8 segmenttiä.



Kuva 20. Tasoja taivutetaan 3ds Maxin ”bend”-toiminnolla.

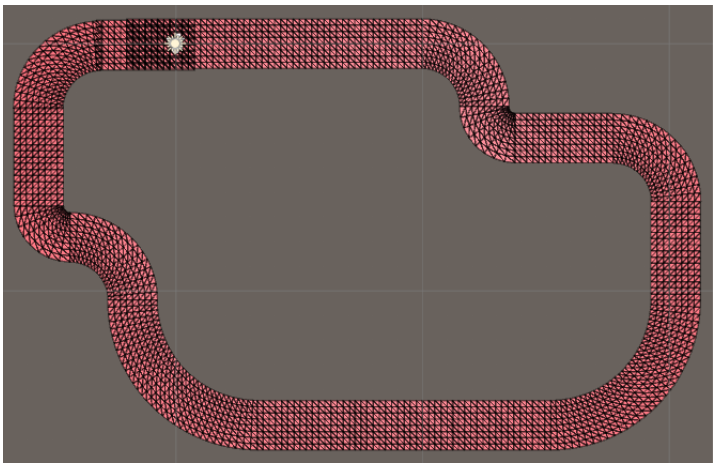
Vietäessä 3D-malleja Unityyn 3ds Maxista ne on syytä muuttaa FBX-muotoon. Max -tiedosto on mahdollista viedä sellaisenaan, mutta se on tiedostokooltaan huomattavasti suurempi, ja sen koordinaatisto on väärinpäin Unityyn nähden. Unityn koordinaatisto poikkeaa Maxista ja monista muista 3D-ohjelmista siinä, että sen Y-akseli osoittaa ylös eikä eteenpäin. Kuva 21 näyttää tärkeimmät säädöt, jotka on syytä tarkistaa malleja viedessä.



Kuva 21. Tarvittavat säädöt vietäessä malleja 3ds Maxista Unityyn.

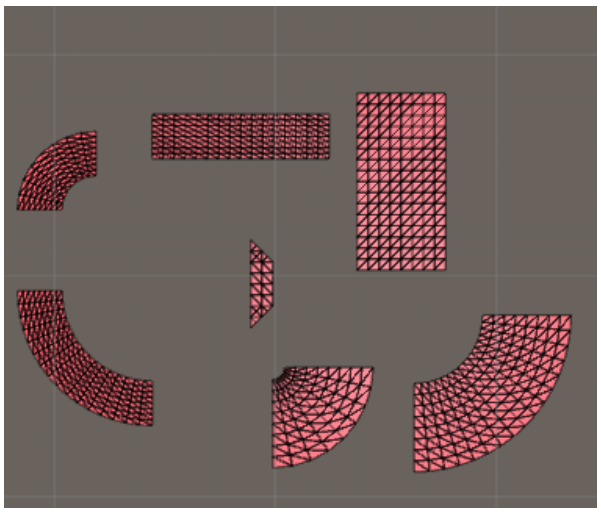
Peliin ei ole ensisijaisesti tarkoitus tehdä vihollisia, joten ainoat tarvittavat 3D-mallit ovat ratapalasia (suoria ja mutkia) ja esteitä, joihin pelaaja voi törmätä.

Kuvan 22 ensimmäinen taso toimi prototyyppikehityksessä testialustana, jonka tarkoitus oli saada hyvä kuva siitä, miten pelin ohjaus toimii suhteutettuna peliympäristöön. Testaamalla ensimmäistä yksinkertaista ruutua sai kartoitettua kiihtyvyysanturin herkkyyttä, sopivaa juoksuvauhtia suhteessa kääntymisvauhtiin sekä esteiden vaikeustasoa.



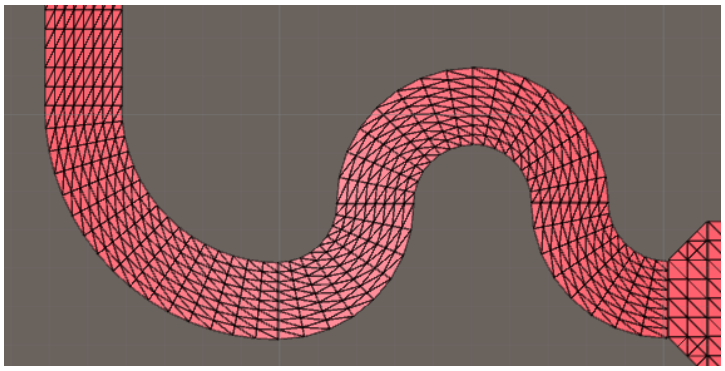
Kuva 22. Ensimmäinen prototyyppitaso Unityssä.

Ensimmäinen testitaso on tarkoituksellisesti tehty silmukaksi, mikä mahdollistaa rajattoman kiihtyvyyden testauksen yhdellä pelikerralla. Korkeita nopeuksia voi toki testata suoraan asettamalla aloitusnopeuden arvon suuremmaksi, mutta vaikeustason kasvamisen kannalta kiihtyvyys on tärkeää testata. Tärkeimmät testausta vaativat elementit olivat kuitenkin sopivan rataleveyden suunnittelu, sekä syöksyliikkeen nopeus ja sen kulkema matka. Esimerkiksi liian kapeissa kohdissa syöksyliike saattaa herkästi viedä pelaajan radan ulkopuolelle tämän tahtomatta. Ensimmäisestä prototyyppitasosta saatu tieto helpottaa muiden ruutujen suunnittelua. 2 metrin levyiset radat vaikuttivat olevan vaikeustasoltaan liian hankalaa kuljettavaa etenkin alkutason maastoksi. Kapeat rataosat sopivat kuitenkin hyvin esteiksi. Kuvassa 23 on esiteltynä vanhojen ratapalojen lisäksi leveämmät kappaleet.



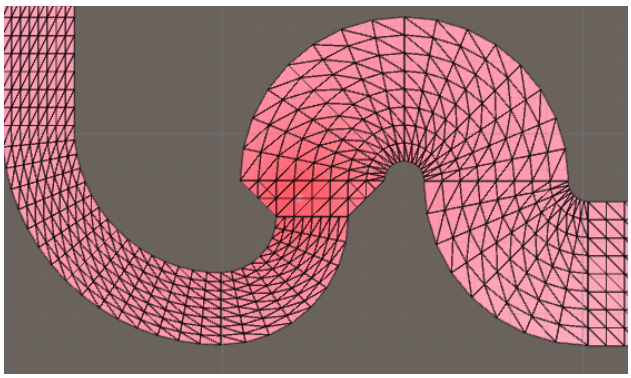
Kuva 23. Toisessa testitasossa käytetyt ratakappaleet.

Toisen radan suunnitteluvaiheessa vastaan tuli nopeasti pelaajalle liian suuria haasteita, etenkin alkupuolen tasoksi. Kuvassa 24 on esimerkki liian haastavasta ratasuunnittelusta.



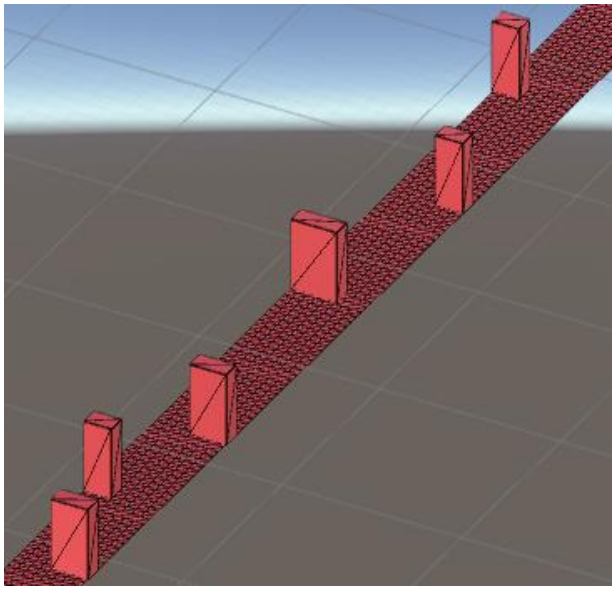
Kuva 24. Liian tiheään asetetut mutkat voivat olla turhauttavan vaikeita.

Mutkaongelma korjattiin leventämällä yhtä mutkaosaa, mikä antaa pelaajalle enemmän tilaa suorittaa vaikea kääntyminen. Mutkakohdan ratkaisu on esitetty kuvassa 25.



Kuva 25. Hankala mutka helpottuu leventämällä mutkan palasia.

Toisessa testiruudussa on erikokoisten ratapalojen lisäksi seinäesteitä (kuva 26). Seinäesteiden tarkoitus pelissä on antaa pelaajalle syytä käyttää syöksyliikettä.

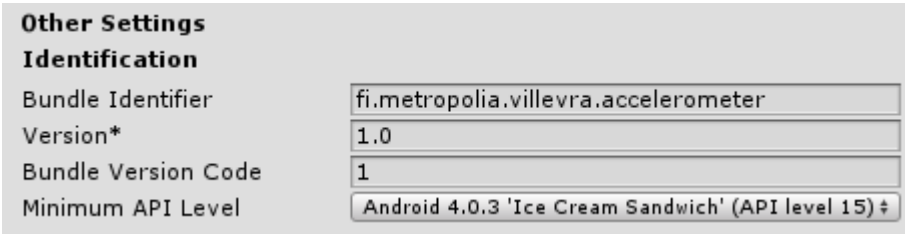


Kuva 26. Testiruudun esteitä.

Vaikka pelaaja voi halutessaan yrittää väistellä seiniä kääntelemällä hahmoa, tiheästi asetettuja seiniä on todella haastavaa ellei jopa mahdotonta väistellä pelkällä kääntelemisellä.

4.3 Projektin rakentaminen ja Android-liitännät Unityssä

Android-projektin rakentaminen on tehty yksinkertaiseksi Unityssä. Edellytyksenä Android-sovelluksen rakentamiselle asennettuna täytyy olla tarvittavat osat Android-sovelluskehityspakkauksesta, esimerkiksi pelin tukemien Android versioiden ohjelmointirajapinnat. Sovellusta rakentaessa on Unityn ”Player Settings” -kohdasta tarkistettava ainutlaatuinen sovellustunniste, joka on muotoa fi.yrityksen.sovellus (esim. fi.metropolia.accelerometer) sekä pelin vaatima minimi Android-ohjelmointirajapinta (kuva 27).



Kuva 27. Pelin sovellustunnisteen ja minimi Android-version säätimet Unityssä.

Unityllä tehtyjä pelejä ja sovelluksia on mahdollista laajentaa Android-liitännällä, joka mahdollistaa sovellukselle Unityn kirjastojen ulkopuolisen ohjelmoinnin. Yksi liitännän tarkoitus on päästä käsiksi Androidin ohjelmointirajapintoihin, joihin Unity ei itsessään kykene. Liitäntää voi myös hyödyntää muihin kolmannen osapuolen Android-kirjastoihin, mikä mahdollistaa valtavan määrän Unitylta tavallisesti poissuljettuja ominaisuuksia. (35.)

Liitännän luominen onnistuu Android Studiolla, mutta se onnistuu muillakin ohjelmilla (kuten Eclipse). Jotta liitäntä voisi toimia, se tarvitsee Unityn asennushakemistosta (”Unity\Editor\Data\PlaybackEngines\androidplayer\release\bin”) Classes.jar -tiedoston, joka liitetään kirjastoksi Android-projektiin. Tämän jälkeen liitäntää rakennetaan Android-ohjelmoinnilla Javalla. Haluttu koodi kootaan lopulta uudeksi JAR-tiedostoksi (Java Archive), joka viedään Unity-projektin ”Plugins/Android” -kansion alle. Liitännän sovellustunnisteen ja ohjelmointirajapinnan vähimmäisvaatimuksen täytyy olla samat kuin Unityn projektissa. Unityssä kirjaston metodeja voidaan kutsua peliobjektiin (esim. kameraan) liitetyllä koodilla ”call”-metodilla. Huomioitavaa on, että on yksinkertaisempaa lähettää kutsuja Unitystä liitäntään kuin päinvastoin. (35.)

Liitäntä olisi esimerkiksi voinut mahdollistaa ilmaeleet pelin ohjauksessa. Idea jäi kuitenkin toteuttamatta aikarajoitteiden ja teknisten haasteiden takia. Kehitystä rajoitti esimerkiksi se, että liitännän sisältävän projektin rakentaminen vaatii maksullisen lisenssin Unitysta, joten kehittäminen kotioloissa ei onnistunut.

4.4 Abyss Runnerin nykytila

Abyss Runner -pelissä on tällä hetkellä vain muutama testitaso, joiden tarkoitus on kartoittaa pelin rajoituksia ja ongelmia. Pelin ohjauksen runko on valmis, mutta etenkin nopeuden ja kiihtyvyyden arvot voivat vaatia hiomista tasosuunnittelun edetessä. Myös syöksyliikkeen suhteuttaminen peliympäristöön ja hahmon nopeuteen vaatii jatkokehittelyä.

Pelin on todettu toimivan hyvin Samsung Galaxy S4 -puhelimella, mutta ottaen huomioon rajoittuneen grafiikan sen pitäisi toimia myös vanhemmilla puhelimilla. Pelin laitekuormitus voi kuitenkin nousta pelikehityksen edetessä, joten jatkotestauksessa on tärkeää testata peliä vanhemmillakin laitteilla.

4.5 Abyss Runnerin tulevaisuus

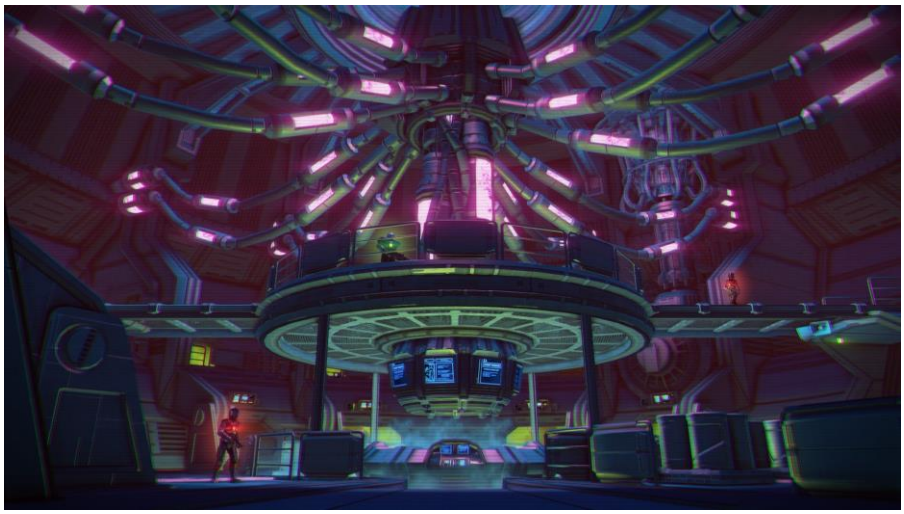
Abyss Runner on vielä prototyyppivaiheessa, ja sen kehittäminen jatkuu. Tulevaisuuden kehityksessä olennaista on luoda pelille rautalankamallin sijaan graafinen ilme ja lisää tasoja. Tämänhetkisen prototyypin tasot ovat lähinnä toimineet pelin toimivuuden kokeilemisessa. Liike- ja eleohjaus pidetään projektin tulevaisuudessakin yksinkertaisena, joten toimintojen lisääminen siihen on toissijaista.

Yksi luonnollinen kehityssuunta pelille olisi satunnaistetut tasot. Satunnaistetuissa tasoissa taso on joko latauduttuaan arponut radan tai lataa lisää ratapalasia pelaajan edetessä. Satunnaistetut tasot ovat suosittuja etenkin roguemaisissa peleissä (engl. ”Roguelike” nimi viittaa peliin ”Rogue”), joissa pelaaja joutuu aloittamaan alusta kuoltuaan. Pelityyliltään Abyss Runneria lähempänä oleva Super Hexagon käyttää myös satunnaistettua tasorakennusta. Vaikka Super Hexagonin tasorakennus onkin satunnaista, sen tasohaasteet ovat toistuvia. Tämä tarkoittaa, että pelaajan saapuessa tietyn esteen eteen hänellä on mahdollisuus reagoida haasteeseen aiempien kokemusten perusteella. Satunnaistamisella ei siis ole tarkoitus pyrkiä täysin satunnaisiin ruutuihin

vaan käyttää ennalta rakennettuja ratakokonaisuuksia. Satunnaistamalla säästää aikaa tasokehityksessä, mutta samalla tasojen vaikeusastetta on vaikeampi tasapainottaa. (32.)

Graafinen ulkoasu tulisi todennäköisesti muistuttamaan aiemmin mainittuja 3D Death Runia ja Super Hexagonia. Luultavimmin pelin taustana olisi vain pimeää, mahdollisesti tähtiä sisältävää taustaa, joka loisi pelaajalle tunnelman tyhjyydestä. Ratapalaset ja esteet olisivat todennäköisesti yksinkertaisia muodoltaan, mutta niiden väritykseen ja valaistukseen tulisi kiinnittää erityistä huomiota. (29; 32.)

Yksi idea graafiseen ulkoasuun oli 80-luvun väri- ja äänimaisemat. Useat viimeaikaiset pelit ovat mukailleet aiempia vuosikymmeniä pelisuunnittelussa ja jotkut niistä ovat heijastaneet silloisten pelien lisäksi yleistä populäärikulttuuria, kuten elokuvia ja musiikkia. Kuvan 28 ”Far Cry 3 - Blood Dragon” sijoittuu 80-lukuhenkiseen kybermaailmaan, jossa on hyvin vahvoja neonvärejä. (36.)



Kuva 28. Far Cry 3 - Blood Dragon -peli (36).

Farcryn tyylisissä peleissä korostuvat hohtavat neonvärit ja syntetisaattoripainotteinen musiikki. Hohtavat neonvalot ja syntetisaattorimusiikki antaisivat pelille vanhanaikaisen futuristisen luonteen. Äänimaiseman suunnittelu on kuitenkin toissijaista ottaen huomioon, että iso osa mobiilipelaajista pelaa äänettömänä. (36)

Tasosuunnittelussa tärkeätä olisi luoda haastavuudeltaan eritasoisia ruutuja. Pelin ensimmäiset tasot olisivat helppoja, ja ne totuttaisivat käyttäjän pelin ohjaukseen ja esteisiin. Tavoitteena vaikeusasteessa on nostaa sitä tasaisesti joka tason välissä. Loppu-

puolen tasot voivat jo olla haastavuudeltaan lähes täydellistä suoritusta vaativia, mutta tärkeää on antaa myös heikommille pelaajalle mahdollisuus suorittaa hyvä osuus pelistä.

5 Yhteenveto

Opinnäytetyössä luotiin yksinkertainen liikeohjausta hyödyntävä Android-peliprototyyppi taustatutkimusten pohjalta. Peliä kehitettiin käyttäen 3ds Max 2016 –ohjelmaa sekä Unity 3D:tä. Lopullinen pelin prototyyppi hyödynsi puhelimen kiihtyvyysanturia sekä pyyhkäisyelettä. Työn edetessä tutustuttiin Android-liitännäisiin Unityssä, mutta ne jäivät lopullisesta prototyypistä pois niiden tarpeettomuuden ja projektin aikarajoitteiden vuoksi.

Tavallisesta poikkeavat ohjausmenetelmät ovat jo pitkään kiehtoneet niin pelaajia kuin peleistä vähemmän kiinnostuneita. 1980-luvun loppupuolen teknisistä syrjähypyistä huolimatta liikeohjaus nousi menestykseen 2000-luvulla Wiin ja Kinectin myötä. Liikeohjattujen konsoleiden suosio on esimerkki siitä, kuinka kysyntää liikeohjaukselle oli aina ollut, mutta vanha teknologia ei ollut vielä tarpeeksi käytännöllisellä tai toimivalla tasolla. Osittain liikeohjauksen heikkoon menestykseen 1980- ja 1990-luvuilla saattoi vaikuttaa taloudellinen tilanne. 2000-luvun teknologia oli sekä toimivaa että edullista. Nintendon Wiitä seurannut Wii U ei kuitenkaan menestynyt yhtä hyvin, mikä puolestaan saattaa viestiä siitä, että Wii ja Kinect olivat jo täyttäneet kuluttajien tarpeet liikeohjatusta kotikonsolista, eikä niin lyhyellä aikavälillä ollut kannattavaa julkaista hyvin samankaltaista laitetta.

Älypuhelimien käytön yleistyminen on vaikuttanut siihen, miten pelaaminen mielletään. Aiemmin pelaaminen saattoi antaa kuvan eristäytyneestä toiminnasta pimeässä huoneessa, mutta mobiililaitteiden suosion myötä yhä useammat ihmiset pelaavat odottaessaan esimerkiksi julkisia kulkuneuvoja tai lääkärin vastaanottoa. Älypuhelimet ovat tuoneet pelaamisen helpommin lähestyttäväksi ihmisille, jotka eivät omista kotikonsolia tai pelikelpoista tietokonetta. Samalla kun puhelimet ovat yleistäneet mobiilipelaamista, ne ovat avanneet markkinoita liike- ja eleohjaukselle. Ehkä yksi tärkeimmistä mobiilipelisuunnittelun alkeista on ymmärtää nykyisten puhelimien käyttöliittymä ja kehittää peli sille sopivaksi. Yleisen älypuhelinmallin rakenteen takia perinteiset peliohjaimet eivät ole käytännöllisin ratkaisu pelisuunnittelussa, joten suurin osa peleistä hyödyntää joko eleohjausta (napautukset, raahaamiset, pyyhkäisyt), liikeohjausta (kiihtyvyysanturit) tai sosiaalista mediaa (esimerkiksi Foursquare/Swarm).

Monissa nykypäivän mobiilipeleissä on kiihtyvyysantureita hyödyntävää liikeohjausta, mutta hitaammissa vuoropohjaisissa strategia- ja mietintäpeleissä liikeohjaus ei ole vielä saanut jalansijaa. Kiihtyvyysanturit mahdollistavatkin monipuolista ohjausta

enemmän nopeatempoisemmissa pelilajeissa, joissa reaktionopeudella ja tarkkuudella on enemmän merkitystä kuin matemaattisella ajattelulla. Abyss Runnerin tarkoitus on olla yksinkertainen ele- ja liikeohjausta yhdistävä mobiilipeli. Etenkin tämänkaltaisessa fuusio-ohjaamisessa on tärkeää pitää ohjaus yksinkertaistettuna, joten käytettävien eleiden määrä pysyi pelin kehityksen aikana minimissä.

Liike- ja eleohjaus ovat tärkeitä resursseja mobiilipelikehityksessä, eikä niiden merkitystä ja käytännöllisyyttä ole syytä vähätellä. Siinä missä peliharrastajat ovat voineet kokea liikeohjauksen rajoittavana tekijänä tietokoneilla ja konsoleilla, varsinkin kosketusnäytöllisillä mobiililaitteilla se on pikemminkin laajentanut pelien kirjoa antamalla vauhdikkaammille peleille ainutlaatuisemman ohjausmenetelmän.

Lähteet

1 Plunkett, Luke. 2011. The World's First True Shooting Game. Verkkodokumentti. Kotaku. <http://kotaku.com/5815694/the-worlds-first-true-shooting-game>. Päivitetty 27.6.2011. Luettu 3.2.2016.

2 Motion Control. 2015. Verkkodokumentti. Giant Bomb. <http://www.giantbomb.com/motion-control/3015-474>. Päivitetty 28.1.2015. Luettu 2.2.2016.

3 KO Punch. 2015. Verkkodokumentti. Segaretro. <http://segaretro.org/KO_Punch>. Päivitetty 16.10.2015. Luettu 22.2.2016.

4 Houghton, David. 2007. How the Wii has changed gaming, one year later. Verkkodokumentti. Gamesradar. <http://www.gamesradar.com/how-the-wii-has-changed-gaming-one-year-later/>. Päivitetty 19.11.2007. Luettu 3.2.2016.

5 Caron, Frank. 2008. Of Gyroscopes and gaming: the tech behind the Wii MotionPlus. Verkkodokumentti. Ars Technica. <http://arstechnica.com/gaming/2008/08/wii-motion-sensor/>. Päivitetty 26.8.2008. Luettu 29.3.2016.

6 Ramsay, Randolph. 2008. Wii Fit Review. Verkkodokumentti. Gamespot. <http://www.gamespot.com/reviews/wii-fit-review/1900-6190546/>. Päivitetty 8.5.2008. Luettu 29.3.2016.

7 Greenwald, Will. 2010. Kinectrospective: A Brief History of Controller-Free Gaming. Verkkodokumentti. PCMag. <http://www.pcmag.com/article2/0,2817,2372058,00.asp>. Päivitetty 4.11.2010. Luettu 3.2.2016.

8 Dent, Steve. 2015. How I turned my Xbox's Kinect into a wondrous motion-capture device. Verkkodokumentti. Endgadget. <http://www.engadget.com/2015/03/08/using-the-kinect-for-motion-capture/>. Päivitetty 3.8.2015. Luettu 3.2.2016.

9 Gujarati, Paresh. 2015. What is Accelerometer and how does it work on smartphones. Verkkodokumentti. Techulator. <http://www.techulator.com/resources/8930-How-does-smart-phone-accelerometer-work.aspx>. Päivitetty 3.10.2015. Luettu 3.2.2016.

10 Worldwide digital games market: 2015 total. 2016. Verkkodokumentti. Superdata. <https://www.superdataresearch.com/blog/us-digital-games-market/>. Päivitetty 26.1.2016. Luettu 22.2.2016.

11 Rantanen, Iiro; Liukkonen, Tapani; Hyrynsalmi, Sami; Smed, Jouni. 2015. Mobiilipelien menestystekijät. Pelitutkimuksen vuosikirja, 2015, s. 133-144. Tampereen yliopisto.

12 Meganoid. 2013. Verkkodokumentti. Google Play. <https://play.google.com/store/apps/details?id=com.meganoid.engine>. Päivitetty 1.7.2013. Luettu 10.3.2016.

13 Street Fighter IV. 2014. Verkkodokumentti. iTunes. <https://itunes.apple.com/us/app/street-fighter-iv/id354655665?mt=8>. Päivitetty 6.1.2014. Luettu 9.3.2016.

14 Scolastici, Claudio & Nolte, David. 2013. Mobile Game Design Essentials. Birmingham: Packt Publishing.

15 Motion Capture. Verkkodokumentti. Organic Motion. <http://www.organicmotion.com/motion-capture/>. Luettu 9.2.2016.

16 Tanz, Jason. 2011. Kinect Hackers Are Changing the Future of Robotics. Verkkodokumentti. Wired. <http://www.wired.com/2011/06/mf_kinect/all/1>. Päivitetty 28.6.2011. Luettu 4.3.2016.

17 Kinect Sensor. Microsoft Developer Network. Verkkodokumentti. <https://msdn.microsoft.com/en-us/library/hh438998.aspx>. Luettu 4.3.2016.

18 Torbochkin, Becky. 2013. Using Gestures in Mobile Game Design. Verkkodokumentti. Gamasutra. <http://www.gamasutra.com/blogs/BeckyTorbochkin/20131118/205079/Using_Gestures_in_Mobile_Game_Design.php>. Päivitetty 18.11.2013. Luettu 11.2.2016.

19 Spellcrafter. 2015. Verkkodokumentti. Google Play. <https://play.google.com/store/apps/details?id=pl.jujubee.sorcerer>. Päivitetty 30.5.2015. Luettu 11.2.2016.

20 Infinity Blade. Verkkodokumentti. Chair Entertainment Group. <http://infinityblade.com/the-game/>. Luettu 26.2.2016.

21 Goodrich, Ryan. 2013. Accelerometers: What They Are & How They Work. Verkkodokumentti. Livescience. <http://www.livescience.com/40102-accelerometers.html>. Päivitetty 1.10.2013. Luettu 4.2.2016.

22 Woodford, Chris. 2015. Accelerometers. Verkkodokumentti. Explain That Stuff. <http://www.explainthatstuff.com/accelerometers.html>. Päivitetty 30.6.2015. Luettu 4.3.2016.

23 Morrison, Jim & Yang, Daniel. Inside the Apple Watch: Technical Teardown. 2015. Verkkodokumentti. Chipworks. <https://www.chipworks.com/about-

chipworks/overview/blog/inside-the-apple-watch-technical-teardown-blog>. Päivitetty 24.4.2015. Luettu 29.3.2016.

24 Lava Bubble Adventure. 2014. Verkkodokumentti. Google Play. <https://play.google.com/store/apps/details?id=com.cmyksoft.lavabubble>. Päivitetty 22.12.2014. Luettu 29.2.2016.

25 Teeter Pro. 2016. Verkkodokumentti. Google Play. <https://play.google.com/store/apps/details?id=pl.surix.teeterpro>. Päivitetty 24.1.2016. Luettu 29.2.2016.

26 Doodle Jump. 2016. Verkkodokumentti. Google Play. <https://play.google.com/store/apps/details?id=com.lima.doodlejump&hl=fi>. Päivitetty 10.3.2016. Luettu 12.3.2016.

27 Incredible Race Car Simulator. 2015. Verkkodokumentti. Google Play. <https://play.google.com/store/apps/details?id=com.car.incrediblecar.racecar&hl=fi>. Päivitetty 26.6.2015. Luettu 13.3.2016.

28 Hearthstone. 2016. Verkkodokumentti. Google Play. <https://play.google.com/store/apps/details?id=com.blizzard.wtcg.hearthstone>. Päivitetty 15.3.2016. Luettu 21.3.2016.

29 Death Run 3D. 2014. Verkkodokumentti. Google Play. <https://play.google.com/store/apps/details?id=com.kapogames.deathrun3d&hl=fi>. Päivitetty 16.12.2014. Luettu 13.3.2016.

30 Accelerometer input. 2015. Verkkodokumentti. Unity. <https://unity3d.com/learn/tutorials/modules/beginner/platform-specific/accelerometer-input>. Päivitetty 24.7.2015. Luettu 9.2.2016.

31 Mario Kart. 2008. Verkkodokumentti. Nintendo. <http://www.mariokart.com/wii/launch/>. Luettu 28.2.2016.

32 Super Hexagon. 2015. Verkkodokumentti. Google Play. <https://play.google.com/store/apps/details?id=com.distractionware.superhexagon&hl=fi>. Päivitetty 27.11.2015. Luettu 9.2.2016.

33 Acceleration. Verkkodokumentti. Unity. <http://docs.unity3d.com/ScriptReference/Input-acceleration.html>. Luettu 9.2.2016.

34 GetTouch. Verkkodokumentti. Unity. <http://docs.unity3d.com/ScriptReference/Input.GetTouch.html>. Luettu 9.2.2016.

35 Horakeri, Sujit. 2015. Create An Android Plugin For Unity Using Android Studio. Verkkodokumentti. The Game Contriver.

<http://www.thegamecontriver.com/2015/04/android-plugin-unity-android-studio.html>. Päivitetty 4.4.2015. Luettu 15.3.2016.

36 Far Cry 3 - Blood Dragon. 2013. Verkkodokumentti. Steam. <http://store.steampowered.com/app/233270/>. Luettu 1.4.2016.