Richard Domander

Agenttien uskottavuus ja tekotyperyys digitaalisissa peleissä

Tietotekniikan
kandidaatintutkielma
26. kesäkuuta 2012

Jyväskylän yliopisto

Tietotekniikan laitos

Jyväskylä

Tekijä: Richard Domander
Yhteystiedot: richard.domander@jyu.fi
Työn nimi: Agenttien uskottavuus ja tekotyperyys digitaalisissa peleissä
Title in English: The believability of agents, and artificial stupidity in digital games
Työ: Tietotekniikan kandidaatintutkielma
Sivumäärä: 37
Tiivistelmä: Tutkielmassa tarkastellaan pelin hahmojen eli agenttien uskottavuutta. Uskottava agentti käyttäytyy pelaajan odotusten mukaisesti. Tekstissä pohditaan, mitkä seikat vaikuttavat agentin uskottavuuteen, ja miten uskottavuus vaikuttaa pelin viihdyttävyyteen. Työssä pyritään tuomaan esille, kuinka agentin etevyys pelissä, tai sen toteutuksen virheettömyys eivät yksin riitä tekemään siitä uskottavaa. Tutkielmassa esitellään myös tekotyperyyden käsite. Tekotyperä agentti on tarkoituksellisesti ohjelmoitu tekemään virheitä. Tekstissä havainnollistetaan, kuinka tekotyperyys voi parantaa agentin uskottavuutta.
English abstract: This thesis considers the believability of the characters of a digital game, i.e. agents. A believable agent meets the expectations of game players. The thesis studies which factors affect believability, and how believability relates to the enjoyability of a game. The text attempts to demonstrate, how even if an agent is error-free, or a masterful player, it may still lack believability. The concept of artificial stupidity is also presented in this treatise. Artificial stupidity denotes the intentional programming of mistakes into the behaviour of an agent. The thesis examines how artificial stupidity can improve the believability of an agent.
Avainsanat: pelit, agentit, tekoäly, tekotyperyys, uskottavuus, viihdyttävyys, haaste, immersio, Turingin testi
Keywords: games, agents, artificial intelligence, artificial stupidity, believability, entertainment, challenge, immersion, The Turing test

Sisältö

1 Johdanto

Kaikille digitaalisia pelejä pelaaville lienevät tuttuja tilanteet, joissa tietokoneen ohjaamat hahmot käyttäytyvät epätoivottavalla tavalla. Esimerkiksi toimintapelissä saatettava VIP-henkilö voi juosta suoraan tulilinjalla, tai ajopelissä vastustaja kiilata pelaajan auton kylkeen, koska noudattaa sokeasti parasta ajolinjaa. *Civilization*-sarjan peleissä taas tietokoneen ohjaamat pelaajat suorastaan huijaavat pysyäkseen kansakuntien kilpajuoksussa mukana. Kun pelien hahmot käyttäytyvät näin, niiden ohjelmallinen luonne paljastuu. Pelaajan on tällöin vaikea uskoa kamppailevansa voitosta tasaväkisten vastustajien kanssa, ja pelin viihdyttävyys kärsii.

Toisin kuin akateemisten tai teollisten sovellusten, pelien ensisijaisena tavoitteena on olla mahdollisimman viihdyttäviä [17]. Peleissä viihdyttävyys on tärkeämpää kuin etevä tekninen toteutus. Tutkielmassa tarkastellaan, kuinka pelien hahmojen *uskottavuudesta* huolehtiminen parantaa niiden viihdyttävyyttä. Lyhyesti uskottavuudella tarkoitetaan sitä, että hahmojen käyttäytyminen on sekä tarkoituksenmukaista että vastaa pelaajan odotuksia.

Pelien viihdyttävyys koostuu pitkälti sopivasta haasteesta [31]. Uskottava pelin hahmo tarjoaa pelaajalle sopivan haasteen. Pelaajan tulisi kokea, että pelin lopputulos — voitto tai häviö — on hänen omaa ansiotaan. Hän ei saisi voittaa, koska pelin vastustajat esimerkiksi juuttuvat maaston esteisiin, tai hävitä, koska ne tähtäävät erehtymättä. Tutkielmassa esitellään kuinka pelin hahmoihin tarkoituksellisesti ohjelmoidut virheet, eli *tekotyperyys* voi parantaa niiden uskottavuutta ja pitää haasteen sopivana.

Luvussa 2 selitetään *tekoälyn* käsite. Siinä tuodaan esille, kuinka tekoälyllä varustetun ohjelman toiminta on vain näennäisen älykästä. Luvussa esitellään Turingin testi keinona kiertää tietokoneohjelman älykkyyden mittaamisen ja määrittelemisen hankaluudet. Luvussa käsitellään myös pelien historiaa tekoälykehityksessä, sekä nykypelien tarjoamat mahdollisuudet tutkimukselle.

Luvussa 3 tuodaan esille, kuinka tekoäly peleissä jakaantuu useaksi itsenäiseksi osaohjelmaksi eli *agentiksi*. Siinä pyritään esittämään riittävä kuvaus agentin käsitteelle. Luvussa esitellään myös lyhyesti erilaisia agentin toteutustyyppejä, sekä miten peligenre vaikuttaa agenttien toteutukseen.

Luku 4 alkaa pelien agenttien eri tehtävien tarkastelulla. Tehtävä vaikuttaa siihen, mistä sen uskottavuus tarkemmin koostuu. Luvussa 4.2 käsitellään uskottavuutta tarkemmin: sen subjektiivisuutta sekä seikkoja, jotka parantavat tai heikentävät agenttien uskottavuutta peleissä. Luvussa 4.3 avataan tekotyperyyden käsitettä esimerkkien avulla, ja tutkitaan sen vaikutusta uskottavuuteen. Luku 4.4 esittelee

tapoja mitata uskottavuutta. Luvussa 4 esitellään myös, kuinka agentin uskottavuus kytkeytyy pelin viihdyttävyyteen.

Luvussa 5 pohditaan pelien tekoälytekniikoita yleisesti, sekä tuodaan esille uusia tapoja soveltaa tekoälyä peleissä. Luku 5.1 esittelee muutaman agenttien toteutuksessa käytetyn menetelmän. Luku 5.2 tarkastelee teknisiä keinoja parantaa agentin uskottavuutta ja pelin viihdyttävyyttä.

2 Tekoäly

Ihmisiä on jo pitkään kiehtonut kysymys, onko mahdollista rakentaa älykäs kone. Tekoälyllä tarkoitetaan tietokoneohjelmaa, joka toimii ilmeisen älykkäällä tavalla. Michael Negnevitskyn [24] mukaan tekoäly on kuitenkin haastava käsite, sillä edes *älykkyyden* määritelmästä ei olla yksimielisiä. Esimerkiksi filosofit kiistelevät siitä, onko älykkyys itsenäinen tekijä, joka voidaan erottaa muista inhimillisistä piirteistä, kuten luovuudesta, tunteista tai moraalista. Tällaisen määritelmän mukaan *älykkyys* on hyvin inhimillinen ominaisuus. Toiset filosofit uskovat, että ainakin periaatteellisella tasolla koneet pystyvät kaikkeen mihin ihminenkin. Negnevitsky määrittelee älykkyyden kyvyksi oppia, ymmärtää, ratkaista ongelmia ja tehdä päätöksiä. Matemaatikko Alan Turing [32] kiersi määritelmien hankaluudet kokonaan muotoilemalla tekoälytutkimukselle hyvin konkreettisen tavoitteen (ks. luku 2.1).

Filosofi John Searle [27] pohti tekoälyn mahdollisuuksia ja määritelmää Kiinalaisena huoneena tunnetun ajatuskokeen avulla. Kokeessa kiinaa osaamaton mies on kiinankielistä tekstiä täynnä olevassa huoneessa. Teksti jakautuu kahteen joukkoon: syötteisiin ja tulosteisiin. Miehellä on myös täydelliset säännöt syötteiden ja tulosteiden vastaavuuksista. Nämä säännöt on kirjoitettu miehen ymmärtämällä kielellä. Kun mieheltä kysyy asioita kiinankielisillä syötteillä, saa kysyjä oikeat kiinankieliset vastaukset. Ulkopuolinen saa vaikutelman, että huoneessa on älykäs toimija, vaikka kirjastonhoitaja vain sokeasti noudattaa ohjeita ymmärtämättä syötteitä tai tulosteita. Searlen luokittelussa heikko tekoäly on kuin ajatuskokeen mies kirjastossa: se toimii näennäisen älykkäästi. Toisin sanoen heikko tekoäly ylläpitää älykkyyden illuusiota suorittamalla toimintoja, joihin ulkopuolinen tarkkailija voi luulla vaadittavan älyä. Vahvalla tekoälyllä on toiminnan lisäksi ymmärrys tekemästään. Esimerkissä vahva tekoäly olisi mies, joka ymmärtää kiinan kieltä. Vahva tekoäly ei ole vain simulaatio mielestä — se on mieli. Vahva tekoäly on nykypäivänä kuitenkin vain tieteiskirjallisuutta, ja niinpä kun tässä tutkielmassa jatkossa puhutaan tekoälystä, tarkoitetaan nimenomaan heikkoa tekoälyä.

2.1 Turingin testi

Vuonna 1950 Alan Turing julkaisi artikkelin [32], jossa hän esitteli tavan testata tekoälyjä. Turing koki ongelmalliseksi koko kysymyksen koneiden mahdollisuudesta ajatella. Hänen mielestään kysymys oli huonosti aseteltu, sillä ensin pitäisi määritellä, mitä käsitteet *kone* ja *ajatella* tarkoittavat. Negnevitsky [24] huomauttaa, että määritelmästä riippuen käsite *ajatella* voi tarkoittaa jotakin, johon lähtökohtaisesti kykenee vain ihminen. Toisaalta ihmistä voidaan tarkastella biologisena koneena. Negnevitskyn tulkinnan mukaan Turing esitti artikkelillaan, että parempi kysymys tekoälytutkimuksen pohdittavaksi olisi, voivatko koneet läpäistä älykästä käyttäytymistä mittaavan testin.

Turingin testin alkuperäisessä muotoilussa kuulustelijaksi valittu henkilö keskustelee esimerkiksi tietokoneen välityksellä kahden eri huoneissa olevan kumppanin kanssa. Toinen kumppaneista on mies ja toinen nainen. Testin ensimmäisessä vaiheessa kuulustelijan on pääteltävä kumppaneiden sukupuoli oikein. Sääntöjen mukaan miehen on yritettävä hämätä ja naisen vakuutettava olevansa nainen. Toisessa vaiheessa miehen korvaa tekoäly, joka samalla tavalla yrittää petkuttaa olevansa nainen. Jos kone onnistuu huiputtamaan kuulustelijaa yhtä hyvin kuin ensimmäisen kierroksen mies, katsotaan sen läpäisseen testin. Turing kutsui testiään myös imitaatiopeliksi. Turing ennusti, että vuoteen 2000 mennessä olisi kehitetty niin hyviä tekoälyjä, että testissä kuulustelijalla olisi 30 prosentin mahdollisuus tulla koneen huijaamaksi. [32] Vuodesta 1991 asti on vuosittain järjestetty Loebnerin kilpailu, josta Turingin testin läpäisevän tekoälyn kehittäjä voittaa sata tuhatta Yhdysvaltain dollaria. Yksikään kone ei ole vielä onnistunut päihittämään testiä. [28] Livingstone [19] kuitenkin huomauttaa että, vaikka tekoäly läpäisisi Turingin testin, se ei tarkoita, että ohjelma todella ajattelisi. Nykyisin testistä käytetään muotoa, jossa ei välitetä keskustelijoiden sukupuolesta, vaan ainoastaan siitä, erehtyykö kuulustelija luulemaan tekoälyä ihmiseksi.

Muun muassa Shieber [28] on kuitenkin kritisoinut Loebnerin palkintoa. Hän esittää, että kilpailu kyllä lisää tekoälytutkimuksen julkisuutta, mutta ei paranna suuren yleisön ymmärtämystä aiheesta. Shieberin mielestä kilpailu ei myöskään tarjoa selkeää tieteellistä tavoitetta tekoälytutkijoille, koska siinä on käytössä Turingin testin rajoitettu versio, eikä rajoitus ole kovin onnistunut. Erona Loebnerin kilpailuun Turingin testin alkuperäisessä muotoilussa ei ole lainkaan selvää, että kuulustelija tietää etukäteen, että yksi keskustelijoista on tekoäly [19]. Toisaalta kilpailussa mikään ei estä jotakuta ihmiskeskustelijoista esittämästä tarkoituksellisesti kömpelöä tekoälykeskustelijaa. Keskustelijoiden pisteytyksestä on havaittu, että kilpailun

tuomareiden kykyyn erottaa tekoäly ihmisestä vaikuttaa voimakkaasti heidän ikänsä, koulutustaustansa ja etenkin heidän tietotekniset taitonsa [14]. Lisäksi Turingin testi saattaa luonnostaan rajoittaa kykyämme tunnistaa älykkyyttä, sillä esimerkiksi vuoden 2003 Loebnerin kilpailussa pitivät tuomarit joitakin testin ihmiskeskustelijoita vain todennäköisesti ihmisenä [19].

Turing oivalsi, että läpäistäkseen testin on kuulusteltavan tekoälyn paikoin erehdyttävä [32]. Se on tarkoituksellisesti ohjelmoitava tekemään joskus inhimillisiä virheitä. Jos tekoäly esimerkiksi vastaisi aina nopeasti ja virheettömästi kuulustelijan matemaattisiin kysymyksiin, sen todellinen luonne keksittäisiin välittömästi. Siis esimerkiksi laskuvirheiden lisääminen testin keskustelevan tekoälyyn parantaa sen mahdollisuuksia huiputtaa kuulustelijaa. Kun tekoäly tekee testissä virheitä, se yllläpitää paremmin illuusiota inhimillisestä keskustelijasta.

2.2 Pelien historia tekoälytutkimuksessa

Idea peliä pelaavasta koneesta on peräisin jo ajalta ennen digitaalisia tietokoneita ja Charles Babbagen mekaanista differenssikonetta. Yhtenä ensimmäisistä idean ilmentymistä voidaan pitää Turkkilaista (saks. *Schachtürke*), joka oli shakkia pelaava mekaaninen kone. Turkkilaiseen kuului suuri puulaatikko, jonka päällä oli shakkilauta ja turkkilaista miestä esittävä nukke. Koneen rakensi itävaltalainen keksijä Wolfgang von Kempelen vuonna 1769 tehdäkseen vaikutuksen keisarinna Maria Teresiaan. Von Kempelen ja muut kiersivät kaiken kaikkiaan 85 vuotta esittelemässä konetta ympäri maailmaa, kunnes vuonna 1854 se tuhoutui tulipalossa. Kiertueensa aikana Turkkilainen ehti voittaa shakissa muun muassa Napoleonin vuonna 1809. Turkkilainen kuitenkin paljastui huijaukseksi. Tosiasiassa nuken käsiä oli ohjannut mies laatikon sisältä. Huijauksen paljastumiseen meni kauan, koska von Kempelen oli rakentanut laatikon nerokkaalla tavalla. Turkkilaisen omistaja saattoi esitellä laatikon sisältöä eri puolilta paljastamatta liikkuvien väliseinien takana piileskelevää koneenkäyttäjää. [6]

Lautapelit ovat olleet tärkeässä asemassa tekoälyn kehityksessä alusta alkaen. Shannon kirjoitti jo vuonna 1950 artikkelin mahdollisuuksista ohjelmoida tekoäly shakkipeliin. Ensimmäinen tekoäly peliin toteutettiin vuonna 1952. Se pelasi tammea. Pian tammitekoälyn jälkeen julkaistiin ensimmäiset shakkia pelaavat tekoälyt. Osaavan shakkitekoälyn kehittämisestä tulikin suosittu tutkimuskohde. Tutkijat eivät tosin aluksi hahmottaneet ongelman laajuutta ja ihmispelaajan tason saavuttamisen haastavuutta. [26]

1970- ja 1980-luvuilla tietokonepelien tutkimus keskittyi kehittämään tekoälyä

shakkiin. Suosittuja tuohon aikaan olivat niin sanotut raa'an voiman (engl. *brute force*) menetelmät. Northwestern Universityn shakkiohjelmien sarja CHESS osoitti, että menetelmän hakunopeus ja koneen etevyys pelissä korreloivat vahvasti. Tekoälytutkimus keskittyikin pitkään optimoimaan algoritmien hakunopeutta, ja mielenkiinto peliagentteja kohtaan hiipui. 1980-luvun loppupuolella heräsi uudelleen tekoälytutkimuksen alkuaikojen vertainen kiinnostus pelien tekoälyn kehittämiseen. Mielenkiinto kasvoi, kun lyhyen ajan sisään julkaistiin monta uutta menetelmää. 1990-luvulla tekoälyt voittivat ensimmäistä kertaa parhaat ihmispelaajat. Vuonna 1994 Chinook-niminen ohjelma voitti mestaruuden tammessa, ja vuonna 1997 Deep Blue päihitti shakkimestari Garri Kasparovin. [26]

Schaeffer et al. [25] ratkaisivat tammen täydellisesti vuonna 2007. He kehittivät peliin tekoälyn, joka ei häviä koskaan. Täydellistä shakkitekoälyä ei vielä ole kehitetty pelin valtavan ratkaisuavaruuden vuoksi. Pelissä on yksinkertaisesti liian monta mahdollista tilannetta, että parasta pelistrategiaa voitaisiin suoraviivaisesti laskea [14]. Tammessakin on jo $5 \cdot 10^{20}$ eri pelitilannetta [25]. Shakin ratkaiseminen ei kuitenkaan ole NP-täydellinen ongelma [14]. Schaefferin ja van den Herikin [26] mukaan shakki ja tammi ovatkin molemmat tekoälykehityksen kannalta suhteellisen helppoja pelejä. Ne ovat sekä deterministisiä että täyden informaation pelejä. Tämä tarkoittaa, että koko pelitilanne on kaikkien pelaajien nähtävissä ja pelin tulevat tilanteet ovat täysin lueteltavissa nykyisen tilanteen perusteella. Schaeffer ja van den Herik antavat vastaesimerkiksi pokerin: se on sekä epätäydellisen informaation peli, sillä kaikki kortit eivät ole nähtävissä, että epädeterministinen pakan satunnaisen järjestyksen vuoksi. Epädeterministisiä peleja kutsutaan myös stokastisiksi peleiksi. Moniin perinteisiin lautapeleihin on suhteellisen yksinkertaista toteuttaa tekoäly myös siksi, että koneen täytyy pystyä vastaamaan pelaajan haasteeseen vain muutamalla inhimillisen älyn osa-alueella, kuten ratkaisun etsimisessä ja päätöksenteossa [16]. Näissä peleissä agentin ei esimerkiksi tarvitse kyetä päättelemään vastustajan pelityyliä tai ennustamaan pelaajan tulevaa käytöstä vain muutaman havainnon pohjalta, koska kaikki informaatio on aina saatavilla. Lautapelien säännöt voidaan usein myös verrattain helposti esittää ohjelmoinnin vaatimassa formaalissa muodossa.

2.3 Nykypelien rooli tekoälyn kehityksessä

Tekoälyjä kehitettiin aluksi tosimaailman pelien digitaalisiin vastineisiin (ks. luku 2.2), mutta nykyajan pelit tarjoavat tekoälylle mitä moninaisimpia ympäristöjä. Pelin ympäristö voi olla simulaatio tosimaailmasta, tai se voi olla oma synteettinen

maailmansa [16]. Jälkimmäisessä tapauksessa on helpompaa toteuttaa pelin tekoäly, sillä sen tarvitsee noudattaa vain pelin keinotekoisen ympäristön lainalaisuuksia. Jos pelin ympäristö on simulaatio tosimaailmasta, voi pelaajalla olla siitä enemmän tietoa ja oletuksia kuin tekoälyllä. Synteettisestä ympäristöstä voi ihmispelaaja tietää korkeintaan yhtä paljon kuin pelin paras tekoäly. Samakin ympäristö voi tarjota tekoälylle monia eri tehtäviä ja rooleja. Pelien ympäristöjen alati kasvava realismi tekee niistä myös varteenotettavan alustan tekoälytutkimukselle robotiikan tai kalliiden simulaattorien sijaan [16].

Lairdin ja van Lentin [16] mukaan pelejä ja kaupallisten pelien tekoälyjä on akateemisesti tutkittu vielä verrattain vähän. Toisaalta peliteollisuudessa työskentelee paljon korkeakoulutettua työvoimaa. Laird ja van Lent esittävät myös, että yhteistyö peliteollisuuden ja tekoälytutkijoiden välillä voisi olla hedelmällistä, sillä:

1. hyvä tekoäly on pelille kilpailuvaltti
2. paremmat tekoälyt voivat mahdollistaa aivan uudentyyppisiä pelejä
3. pelialalla on tarjota rahoitusta tutkimukseen
4. pelikonsolit ja tietokoneiden näytönohjaimet tarjoavat valtavasti edullista laskentatehoa
5. tekoäly ei peleissä ole kehittynyt samaa tahtia grafiikan ja fysiikkamallinnuksen kanssa

Lisäksi pelit tuovat tutkimukselle ja kehitykselle suuren yleisön mielenkiinnon. Yleisö voi loppukäyttäjän roolin lisäksi toimia myös tekoälyn testaajana, jopa vapaaehtoisesti.

Koska pelikehityksessä huomio on ollut kiinnittynyt parempaan grafiikkaan, on tekoäly peleissä vielä verrattain kehittymätöntä. Esimerkiksi peliala otti yleisesti käyttöön akateemisessa maailmassa laajalti tunnetun A*-algoritmin (lausutaan: ”A tähti”) vasta vuonna 1996. Pelit ovat ihanteellinen sovellusalue ihmistä imitoivan tekoälyn toteuttamiseen, koska kehittyneemmille tekoälyille on kysyntää. Esimerkiksi moninpelien suosio johtuu osin realistisempien vastustajien kaipuusta. Vielä vuonna 2002 tekoälyn ohjaamat hahmot käyttäytyivät melko alkeellisesti, koska niiden toteutus käytti yksinkertaisia sääntöpohjaisia järjestelmiä ja tilakoneita. [26]

3 Agentit

Tekoälyn ohjaamia tietokoneohjelman osia kutsutaan agenteiksi [22]. Agentti on itsenäisesti ja dynaamisesti toimiva ohjelma tai osaohjelma. Agentti ei ohjelmana ainakaan lähdekooditasolla eroa välttämättä lainkaan muunlaisista ohjelmista — siksi se tunnistetaan ja määritellään usein toiminnallisten erityispiirteidensä avulla. Agentilla on sensoreita, joilla se havainnoi toimintaympäristöään, ja aktuaattoreita eli toimilaitteita, joilla se vaikuttaa ympäristöönsä. [8] Agentin ympäristö voi olla täysin synteettinen, esimerkiksi digitaalisen pelin maailma. Toisaalta agentti voi toimia tosimaailmassa, esimerkiksi ohjatessaan robottia. Kuitenkaan pelkkä ympäristön havainnoiminen ja siihen vaikuttaminen ei vielä tee ohjelmasta agenttia [22]. Näiden lisäksi agenttia kuvaavat Woolridgen ja Jenningsin mukaan [22] muun muassa seuraavat erityispiirteet:

- Autonomisuus: agentit toimivat ilman muiden väliintuloa, ja ne jollakin tapaa hallitsevat toimintaansa ja sisäistä tilaansa.
- Sosiaaliset taidot: agentit kommunikoivat toisten agenttien ja mahdollisesti myös ihmisten kanssa.
- Reaktiokyky: agentit reagoivat ympäristönsä tapahtumiin enemmän tai vähemmän reaaliaikaisesti.
- Ennakoivuus: agentit eivät pelkästään reagoi ympäristönsä tapahtumiin, vaan toimivat aloitteellisesti saavuttaakseen päämääriänsä.

Erään hyvän määritelmän agentille antavat Franklin ja Grasser [8]: *"Autonominen agentti on järjestelmä, joka sijaitsee tietyssä ympäristössä ja on osa sitä. Se havainnoi ympäristöään ja toimii siinä jatkuvasti tavoitellen päämääriään. Agentti pyrkii vaikuttamaan siihen, mitä se havaitsee tulevaisuudesta."*[1] Nämä kaksi luonnehdintaa antavat yhdessä toisaalta riittävän tarkan ja toisaalta tarpeeksi laajan rajauksen sille, mitä agentuuri (engl. *agency*) sisältää [22].

Kun tutkielmassa jatkossa puhutaan pelien tekoälystä, on hyvä pitää mielessä, että se jakautuu useammaksi itsenäiseksi palaseksi. Vaikka agentit toimisivat ohjelmassa samalla tavalla, ajatellaan niillä kaikilla olevan oma tekoälynsä, ei yksi yhteinen logiikka. Tietynlainen yhteinen logiikka voi kuitenkin muodostua moniagenttijärjestelmässä, jossa useampi agentti toimii samassa ympäristössä keskenään vuoro-

[1] *"An autonomous agent is a system situated within and a part of an environment that senses that environment and acts on it, over time, in pursuit of its own agenda and so as to effect what it senses in the future."*

vaikutuksessa. Agentit voivat muun muassa jakaa havaintojaan, suunnitella yhteistyötä ja reagoida toisiinsa [2]. Toisaalta yksi agentti voi koostua useasta aliagentista ja muodostaa itsessään moniagenttijärjestelmän [8].

3.1 Reaktiiviset, harkitsevat ja hybridiagentit

Tämä luku perustuu Mac Nameen väitöskirjaan [22]. Agentteja voi niiden piirteiden lisäksi luokitella niiden toteutustavan mukaan. Tekoälytutkimus on tähän mennessä tunnistanut toteutusarkkitehtuuriltaan kolmentyyppisiä agentteja: reaktiivisia, harkitsevia (engl. *deliberative*) sekä näiden piirteitä yhdisteleviä hybridiagentteja.

Reaktiiviset agentit ovat toteutustavaltaan agenteista yksinkertaisimpia. Niiden toiminta koostuu esimääritellyistä heräte-vaste pareista. Tietty syöte agentin sensoreihin aiheuttaa aina samat aktuaattoreiden toiminnat. Reaktiivisen agentin voi toteuttaa sääntöpohjaisena järjestelmänä (engl. *rule-based system*) tai äärellistä tilakonetta apuna käyttäen. Seuraava kaava kiteyttää reaktiivisten agenttien logiikan:

$$\textit{ympäristön nykyinen tila} \rightarrow \textit{toiminta}$$

Reaktiivisten agenttien etuna on niiden yksinkertaisuudesta johtuen helppo toteutettavuus sekä laskennallisten resurssien vähäinen kulutus. Niiden haittapuolena on käyttäytymisen kankeus sekä kyvyttömyys oppia tai laatia pitkän aikavälin suunnitelmia.

Harkitsevat agentit muodostavat ympäristöstään sisäisiä malleja, joiden avulla ne luovat suunnitelmia saavuttaakseen tiettyjä tavoitetiloja (engl. *goal state*). Seuraava kaava tiivistää agentin toiminnanohjauksen:

$$\textit{ympäristön nykyinen tila} + \textit{tavoitetila} \rightarrow \textit{suunnitelma}$$

Harkitsevien agenttien suurin etu on niiden kyky muodostaa pitkän aikavälin suunnitelmia. Ne ovat kuitenkin melko monimutkaisia eivätkä pysty takaamaan reaaliaikaisuutta. Harkitsevan agentin voi myös olla vaikea ylläpitää johdonmukaista mallia ympäristöstään, jos sen maailma on kovin nopeatempoinen ja dynaaminen.

Hybridiagentit yhdistelevät reaktiivisten ja harkitsevien agenttien parhaita puolia. Hybridiagentti voi esimerkiksi käyttäytyä reaktiivisen agentin tavoin aikakriittisissä tehtävissä, kuten törmäystarkistuksessa (engl. *collision detection*), ja harkitsevan agentin tavoin laatia pitkän aikavälin suunnitelmia. Hybridiagentti koostuu ikään kuin kahdesta aliagentista tai -järjestelmästä, ja niiden sujuva yhteistoiminta voi olla haastavaa toteuttaa.

3.2 Agentit erityyppisissä peleissä

Schaeffer ja van den Herik [26] jakavat pelit karkealla tasolla täydellisen ja epätäydellisen informaation peleihin sekä deterministisiin ja epädeterministiin eli stokastisiin peleihin. Heidän mukaansa pelityyppien asettamat haasteet agenttien ohjelmoinnille poikkeavat merkittävästi. Täydellisen informaation peleihin soveltuvat tekniikat eivät toimi stokastisissa tai epätäydellisen informaation peleissä. Nykyään 10 miljardin dollarin kasvava peliteollisuus julkaisee markkinoille mitä erilaisimpia pelejä, ja ne voidaan hienojakoisemmin luokitella eri pelityyppeihin eli genreihin. Tunnettuja peligenrejä esimerkkeineen ovat muun muassa:

- ajopelit: *Pole Position*, *FlatOut*, *Forza Motorsport 4*
- jumalpelit: *Populous*, *Black & White*, *Sim City IV*
- kamppailupelit: *Street Fighter II*, *Mortal Kombat*, *Soul Calibur V*
- roolipelit: *Nethack*, *Fallout*, *The Elder Scrolls V: Skyrim*
- seikkailupelit: *Zork*, *Myst*, *Tales of Monkey Island*
- strategiapelit: *Steel Panthers*, *Starcraft*, *Civilization V*
- toimintapelit: *Doom*, *GTA IV*, *Halo*
- urheilupelit: *Track & Field II*, *Tony Hawk's Pro Skater*, *NHL 12*

On kuitenkin huomattava, etteivät genrerajat ole kovin selkeitä, ja että pelit usein sekoittavat useampaa genreä [16].

Pelin genre vaikuttaa tarvittaviin tekniikoihin agentin toteutuksessa. Muun muassa toiminta-, urheilu- ja ajopelit vaativat reaaliaikaista reagointia, havainnointia ja päätöksentekoa. Reaaliaikainen strategiapeli (engl. *Real Time Strategy, RTS*) vaatii tekoälyä mukautumaan ihmispelaajan toimintaan. Strategiapelit ovat lisäksi useimmiten epätäydellisen informaation pelejä, mikä tekee tekoälyn seuraavien siirtojen suunnittelusta haastavaa. Jos pelissä on realistinen ympäristö, se tekee tekoälyn navigoinnin ja havainnoinnin toteutuksen mutkikkaaksi. Joskus on myös tarpeen mallintaa ihmisen kognitioita ja havainnointia. Esimerkiksi ei ole suotavaa, että tekoäly näkee pimeässä ympäristössä eri tavalla kuin ihmispelaaja. [16]

Tekoäly toimii peleissä erilaisissa rooleissa. Kaikkein perinteisin on pelaajan vastustajan rooli [29], mutta agentit toimivat usein myös kumppaneina, sivullisina, yksikköinä, selostajina tai pelin ohjaajina. Vastustajat jakaantuvat edelleen strategisiin

ja taktisiin [16]. Ohjaajalla tarkoitetaan esimerkiksi *Left 4 Dead*-toimintapelin [1] tekoälyä, jonka tehtävänä on rytmittää jännitystä lisäämällä pelimaailmaan vastustajia ja haasteita tarpeen mukaan. Agentit voivat yksittäisessä pelissäkin toimia useassa eri roolissa. Esimerkiksi *NHL 12*-jääkiekkopelissä [2] agentit toimivat kenttäpelaajina, pelaajan joukkuetovereina, maalivahteina, tuomareina, selostajina, yleisönä ja valmentajina, jotka päättävät muodostelmista ja taktiikoista.

Myös agentin pelaama rooli vaikuttaa sen toteutukseen. Esimerkiksi taktisten vastustajien ja pelaajien kumppaneiden vaatimukset poikkeavat merkittävästi toisistaan. Taktiset vastustajat pyrkivät tarjoamaan pelaajalle haastavia mittelöitä, kun taas kumppanit pyrkivät mukautumaan pelaajan tavoitteisiin [22]. Varhaisemmissa toimintapeleissä vastustajat pelkästään juoksivat pelaajaa kohti. Niistä tehtiin haastavampia yksinkertaisesti ohjelmoimalla ne tulivoimaisemmiksi tai kestävämmiksi. Usein ne myös huijasivat näkemällä taaksensa tai seinien läpi. [16] Seuraava askel toteutuksissa oli skriptata agenttien käyttäytyminen. Tällöin pelaaja saattoi helposti opetella agenttien toimintatavat ulkoa. Uudemmissa peleissä viholliset noudattavat jo yksinkertaisia taktiikoita. Ne järjestävät väijytyksiä, hakeutuvat suojaan ja kutsuvat apujoukkoja. [22] Agenttien liikkuminenkaan ei ole enää ennalta säädettyä, vaan esimerkiksi palkitussa *Half-Life*-pelissä agentit käyttävät polunsuunnittelualgoritmia (engl. *path planning*) [16].

Laird ja van Lent [16] sekä Mac Namee [22] ovat analysoineet agentteja kumppanin ja taktisen vastustajan roolissa. Kumppaniagentit ja taktiset vastustajat tarvitsevat osin samoja taitoja, koska myös kumppanit pyrkivät päihittämään vastapuolen hahmoja yhdessä pelaajan kanssa. Kuitenkin siinä missä vastustajien tulee olla autonomisia, tulee kumppaneiden toteutuksen korostaa vaivatonta yhteistyötä pelaajan kanssa. Ainakaan ne eivät saisi hankaloittaa pelin etenemistä. Kumppanien tulee sovittaa toimintaansa, ymmärtää tiimityöskentelyä, mallintaa ihmisen tavoitteita ja mukautua tämän pelityyliin. Nykypeleissä yhteistoiminta muodostuu kuitenkin lähinnä pelaajan antamista, yksinkertaisista komennoista, kuten ”hakeudu suojaan”, ”seuraa” ja ”hyökkää”.

4 Tekotyperyys ja agenttien uskottavuus

Luku alkaa pelien agenttien eri tehtävien tarkastelulla, koska niiden eri painotukset vaikuttavat siihen, mistä agentin uskottavuus muodostuu. Luvussa 4.2 tarkastellaan subjektiivista uskottavuuden käsitettä sekä pyritään havainnollistamaan, miten ul-

[2] `http://www.ea.com/fi/nhl-12`

koinen uskottavuus ei vastaa agentin sisäistä toteutusta; samalla tavalla kuin Searlen ajatuskokeen (ks. luku 2) heikon tekoälyn ulkoinen käytös on näennäisen älykästä, vaikka se ei ymmärräkään mitä tekee. Agenttien uskottavuus peleissä koostuu sekä riittävästä että sopivasta haasteesta ja viihdyttävyydestä [17]. Luvussa 4.2 keskitytään haasteen riittävyyteen agentin eri tehtävissä ja tarkastellaan minkälaiset seikat ja puutteet särkevät illuusion uskottavasta vastustajasta. Luvussa 4.3 käsitellään agentin toimintaan tarkoituksellisesti ohjelmoituja virheitä eli tekotyperyyttä. Lopuksi luvussa 4.4 esitellään lyhyesti uskottavuuden mittaamista peleissä.

4.1 Agenttien tehtävät peleissä

Livingstone [19] toteaa, että agentin uskottavuuden sisältö vaihtelee sen tehtävän mukaan. Tehtävät eivät ole toisensa poissulkevia. Usein esimerkiksi roolia esittävän agentin päämääränä on samalla tarjota viihdettä. Livingstone jakaa agentit ihmistä imitoiviin eli pelaaja-agentteihin ja roolia suorittaviin eli ei-pelaaja-agentteihin. Hienojakoisemmin agentin tarkoituksena voi eri peleissä olla:

1. ihmisvastustajan simuloiminen [19]
2. jonkin pelimaailman roolin suorittaminen [19]
3. viihteen tarjoaminen [29]
4. pelaajan taitojen harjoittaminen [29]

Yhteistä näille tehtäville on, että niissä kaikissa agentti yrittää tietyllä tapaa läpäistä pelin kontekstiin sovitettua, rajoitettua versiota Turingin testistä. Kuitenkin pelitilanne eroaa Turingin testistä siinä, että pelaaja usein tietää *a priori* olevansa tekemisissä tekoälyn kanssa. Testi on perinteisessä muodossaan varsin tyly ja erotteleva: tekoäly joko läpäisee sen tai sitten ei. Asteittaisempi testaus mahdollistaisi uskottavuuden eri puolien itsenäisen kehittämisen ja huomion kiinnittämisen tekoälyn puutteisiin [19]. Harnad [10] on kehittänyt Turingin testistä yleisemmän, viisiportaisen hierarkkisen version. Hierarkia tarjoaa joukon osatavoitteita kohti täysiveristä, ihmistä vastaavaa tekoälyä, joka läpäisisi korkeimman T5-tason testin (T niin kuin Turing). Alkuperäinen Turingin testi sijoittuu hierarkiassa tasolle T2. Matalimman t1-tason testi ei ole vielä Turingin testi, ja siksi Harnad nimittää sitä lelutestiksi (engl. *toy*). T1-tason testissä tekoäly imitoi joitakin pieniä, osittaisia inhimillisen toiminnallisuuden palasia tai yksinkertaisia mutta eheitä kognition osaprosesseita. T1-tason testissä mikä tahansa älykkyyden aspekti kelpaa imitoitavaksi — ei pel-

kästään keskustelu [19]. Tämä testi soveltuu peleille, koska niiden tarjoama haaste rajoittuu usein muutamalle inhimillisen älyn osa-alueelle [16].

Kun agentti simuloi ihmisvastustajaa, se pelaa samaa peliä kuin ihminen: samoilla säännöillä ja samoilla tavoitteilla. Agentti on tällöin samassa roolissa kuin ihmispelaaja. Se ei esitä ei-pelaaja-hahmoja, vaan simuloi tai imitoi ihmispelaajia [19]. Perinteisesti ongelmana on ollut ohjelmoida peliin tekoäly, joka pystyy voittamaan ihmisen [26]. Nykyään esimerkiksi tammi ja ristinolla on ratkaistu täydellisesti, ja monessa muussakin pelissä vain suurmestarit pelaavat tekoälyjen tasolla. Kuitenkin vaikka tekoäly voittaisi ihmisen pelissä, se ei vielä merkitse, että se onnistuneesti simuloisi ihmisvastustajaa [14]. Ihmisvastustajan simulointiin liittyy muutakin kuin mahdollisimman taitava pelaaminen, esimerkiksi useiden satojen millisekuntien reaktioajat nopeutta vaativissa peleissä [15]. Myös moninpelien ohjelmarobottien eli niin kutsuttujen bottien voidaan katsoa imitoivan ihmispelaajia. Botit voivat verkkopeleissä täydentää puuttuvia osallistujia. Ihannetapauksessa pelaaja ei pelissä erota, milloin hän kamppailee botin ja milloin toisen ihmisen kanssa [19].

Livingstone [19] tarkoittaa pelimaailman roolilla lähinnä jotakin agentille asetettua tehtävää, toimea, ammattia tai muuta päämäärää. Käsitteellä ei tässä viitata henkilörooliin tai näytelmälliseen roolisuoritukseen. Esimerkiksi toimintapeleissä agentit esittävät sotilaan roolia. Sotilasagentti toimii tavalla, joka pyrkii imitoimaan tosimaailman taistelijoiden taktiikoita, koulutusta ja toimintatapaa. Ihmisvastustajan simuloinnin ja tietyn roolin esittämisen välillä on hienosyinen mutta selvä ero. Esimerkiksi reaaliaikaisessa strategiapelissä armeijan komentajaa voi ohjata sekä tekoäly että ihminen. Komentaja vastaa strategisen tason päätöksistä, kuten tuotannosta, resurssien keräämisestä ja hyökkäyssuunnitelmista. Ihmisen taas ei voi kuvitella ohjaavan jokaista strategiapelin yksikköä, vaan ne ovat aina tekoälyn vastuulla. Kun tekoäly toimii komentajana, se simuloi ihmisvastustajaa. Kun tekoäly taas ohjaa pelin puoliautonomisia yksiköitä, esittää se yksikönmukaista roolia. Agentti suorittaa roolia silloin, kun se ei ole korvattavissa ihmispelaajalla. Tekoäly voi ohjata peleissä myös muita kuin inhimillisiä älyjä, kuten pelimaailman eläimiä, avaruusolentoja, robotteja, lohikäärmeitä ja muita fantastisia hahmoja. Myös näissä tapauksissa tekoäly esittää roolia, eikä simuloi ihmisvastustajaa. Roolin suorittaminen pelissä rinnastuu Turingin testiin, jossa tekoäly on keskustelukumppanin roolissa. Kun agentti on pelissä jossakin tietyssä roolissa, tarkoittaa uskottavuus sitä, miten hyvin roolisuoritus onnistuu. Livingstone esittää, että Harnadin t1-tason testiä voidaan edelleen soveltaa muille kuin inhimillisille älyille. Esimerkiksi eläimen tapauksessa agentti imitoi jotakin eläimellisen älyn osatekijää. Fantastisen hahmon kohdalla

agentin käyttäytymistä verrataan pelaajien odotuksiin sen älyllisestä kapasiteetista. Lisäksi agenteilla on peleissä muitakin rooleja kuin toimia pelaajan vastustajana. Ne esittävät muun muassa tämän kumppaneita sekä pelimaailmaa kansoittavia lukuisia sivullisia hahmoja ja olentoja. Eräs agentin suunnittelua hankaloittava kysymys on, pitäisikö sen roolin vastata jotakin tosimaailman roolia vai seurata pelimaailman lainalaisuuksia. Joskus pelin säännöt tai tekniset rajoitukset luovat tilanteen, jossa mielekkäin toimintatapa on mahdoton tosimaailmassa.

Kun tekoäly tarjoaa viihdettä, pelaa se periaatteessa eri säännöillä kuin pelaaja. Viihdyttävä tekoäly tarjoaa haastetta, mutta pelaa loppujen lopuksi hävitäkseen [17] — siis periaatteessa eri peliä kuin ihminen. Westin [33] mukaan sopiva haaste ja agentin realistisuusvaatimukset riippuvat paljon peligenrestä. Kun pelaaja on kahdenkeskeisessä kilpailussa agentin kanssa, esimerkiksi shakissa, pokerissa tai biljardissa, odottaa hän agentilta varsin inhimillistä käytöstä. Yksi vastaan monta tyyppisissä peleissä, kuten ensimmäisen persoonan ammuntapeleissä (engl. *first-person shooter*), oletetaan agenttien olevan heikompia kuin pelaaja. Näissä peleissä haasteen muodostaakin useimmiten vihollisten määrä. Näiden pelityylien väliin asettuvat pelit, joissa pelaaja kilpailee useamman tasavertaisen vastustajan kanssa. Tällaisten agenttien ei odoteta olevan yhtä realistisia kuin shakkitekoälyn, mutta toisaalta niiden oletetaan tarjoavan enemmän haastetta kuin ammuskelupelin vastustajien. Esimerkki kohtalaisen realistisista agenteista ovat ajopelien kanssakilpailijat.

Harjoittavan agentin ensisijaisena tehtävänä on auttaa pelaajaa kehittymään tiedoissa ja taidoissa. Pelien perehdytysjaksoissa (engl. *tutorial*) agentit voivat toimia rajoitetummin kuin varsinaisessa pelissä ja opettaa pelaajalle perustoimintoja. Joissakin peleissä on erillisiä harjoitustiloja, joissa pelaaja voi säätää agenttien tarjoamaa vastusta omien kehityskohteidensa mukaan. Esimerkiksi kamppailupelissä *Super Street Fighter IV* [3] pelaaja voi opetella torjumaan tietyntyyppisiä iskuja säätämällä agentin hyökkäämään toistuvasti. Harjoittavia agentteja esiintyy myös hyötypeleissä [29]. Hyötypelejä ovat muun muassa opetuspelit, simulaatiot ja liikuntapelit, kuten *Wii Fit*. Ammattimaisissa simulaattoreissa harjoittavat agentit ovat tärkeässä roolissa, sillä ei ole mahdollista täyttää simulaatiota pelkästään ihmispelaajilla, jo pelkästään korkeiden käyttökustannusten takia [13].

4.2 Agenttien uskottavuus

Peleissä on tärkeintä, että tekoäly keinolla millä hyvänsä läpäisee niihin sovelletun Turingin testin, eli antaa pelaajalle — joka on testin kuulustelija — kokemuk-

[3] `http://www.streetfighter.com/us/ssfiv`

sen älykkäästä toimijasta. Agentin uskottavuus, sen ylläpitämä älykkyyden illuusio on tärkeämpää kuin sen sisäisen toteutuksen etevyys. Uskottavuudelle oleellisinta on ihmisen kykyjen imitoiminen, ei tämän päihittäminen pelissä. Vaikka tekoäly voittaisi shakkimestarin, se ei vielä ole läpäissyt testiä, jos vastustaja pitää sitä koneena. [19] Krolin [14] mukaan *Deep Blue* läpäisi Turingin testin sovelletun version vuonna 1997, mutta ei siksi, koska se voitti Kasparovin, vaan koska Kasparov epäili, että hän ei pelannut pelkästään konetta vastaan. Kasparov näki *Deep Bluen* pelissä inhimillisiä piirteitä, ja siksi sen voitto oli saavutus uskottavuudelle, ei algoritmisuunnittelulle tai supertietokoneen laskentateholle. Krol kertoo, että jo vuonna 1991 Kasparov pystyi erottamaan *Deep Bluen* edeltäjän *Deep Thoughtin* pelit ihmispelaajien peleistä vain 50 prosentin tarkkuudella. Koska asiantuntijatkaan eivät enää pysty erottamaan konemaista pelaamista inhimillisestä, shakkitekoälyt imitoivat onnistuneesti ihmispelaajia. Turing pitikin shakkia tekoälylle otollisimpana osa-alueena imitoida ihmistä. Vanhemmissa toteutuksissa tekoälyt oli vielä mahdollista tunnistaa sen perusteella, että ne pitivät materiaaliylivoimaa tärkeämpänä kuin edullista pelitilannetta ja siten pelasivat passiivisemmin kuin ihmiset.

Uskottavuudella ei välttämättä ole mitään tekemistä agentin teknisen edistyksellisyyden tai kompleksisuuden kanssa. Uskottavalle agentille ei ole tärkeintä valita aina tehokkainta toimintatapaa, vaan valita järkeenkäyvästi sekä reagoida valintojensa seurauksiin realistisesti siten, että pelaaja voi uskoa sillä olevan tavoitteita ja uskomuksia [22]. Teknisen suorituskyvyn korostamisesta voidaan puhua jopa kompleksisuuden harhakäsityksestä (engl. *complexity fallacy*): ohjelmoijat helposti lisäävät tekoälyn koodin monimutkaisuutta kuvitellen, että se automaattisesti parantaa agentin näkyvän käyttäytymisen tarkoituksenmukaisuutta [17]. Schaefferin et al. [25] täydellinen tammitekoäly on hyvä esimerkki teknisesti edistyksellisestä, mutta epäuskottavasta tekoälystä. Digitaalisessa tammessa tekoälyn tehtävä on simuloida ihmisvastustajaa [29], ja ihmisvastustajat eivät ole voittamattomia. Toisaalta uskottavuutta voidaan joskus parantaa hyvin yksinkertaisilla teknisillä keinoilla. Esimerkiksi Lidén [17] kertoo, että *Half-Life*-ammuntapelin testeissä havaittiin ongelma agenttien polunetsintäalgoritmissa: ne eivät aina pystyneet pakenemaan kranaatin räjähdystä. Pelin kehittäjät eivät pyrkineet ratkaisemaan vikaa, vaan he animoivat agentit niin, että ne reagoivat tapahtumaan yrittämällä suojautua.

Livingstonen [19] mukaan Turingin testin kriteerit soveltuvat hyvin agentin uskottavuuden arviointiin, kun se simuloi ihmisvastustajaa. Ne eivät kuitenkaan hänen mielestään välttämättä auta tekoälysuunnittelijoita parantamaan agenttinsa uskottavuutta. Testin kriteerit ovat käyttökelpoisia lähinnä sellaisessa tilanteessa, jossa pelaaja ei tiedä, onko vastustaja ihminen vai tekoäly — esimerkiksi moninpelien

bottien tapauksessa. Testi ei myöskään sellaisenaan sovellu ei-pelaaja-hahmoille. Niille parempi testi on sellainen, joka ei tutki, onko agentti kone vai ihminen, vaan arvioi agentin käyttäytymistä. Livingstone esittää kolme yleistä kriteeriä uskottavuuden arviointiin:

- Suunnittelu: agentin tulisi osoittaa jonkinasteista taktista tai strategista suunnittelua, sovittaa toimintaansa toisiin pelaajiin ja agentteihin ja vaihtaa suunnitelmaa, jos nykyinen epäonnistuu toistuvasti.
- Toiminta: agentilla tulisi olla ihmisenkaltaiset kyvyt ja reaktioajat.
- Reagointi: agentin tulisi reagoida pelaajien sekä sen liittolaista ja vihollisten läsnäoloon sekä toimintaan ja huomata muutokset ympäristössään.

Näistä kriteereistä muodostuva testi sijoittuu Harnadin hierarkiassa (ks. luku 4.1) tasolle t1. Kriteerit eivät kuitenkaan sovellu kaikille agenteille. Esimerkiksi Livingstone huomauttaa, että suunnittelu ei kriteerinä välttämättä tule kysymykseen, jos agentti imitoi eläintä. Reagointi ei sovellu, jos agentin roolina on vähemmän valveutunut hahmo.

Ihmisvastustajan simuloinnin uskottavuus on sekin suhteellista. Esimerkiksi pelaajan odotuksiin vaikuttaa paljon se, tietääkö hän pelaavansa simuloivaa tekoälyä vastaan vai ei [19]. On oleellista huolehtia, ettei agentti saavuta yli-inhimillisten kykyjensä, kuten valtavan laskentatehon, ansiosta pelaajalle ilmeistä etulyöntiasemaa. Ihmispelaajan taitojen ja rajoitteiden sopivalla simuloinnilla voidaan varmistaa, että agentti tarjoaa samanveroisen haasteen. Esimerkiksi nopeutta vaativissa peleissä, kuten ammuntapeleissä, agentin reaktioaikoja tulee hidastaa, jotta ne vastaavat ihmisen toimintanopeutta [15]. Shakkisimulaatio ei kuitenkaan välttämättä parane siitä, jos agentti käyttää realistisesti kymmenen minuuttia seuraavan siirron harkitsemiseen — ellei todella yritetä saada pelaajaa uskomaan, että hän pelaa toista ihmistä vastaan.

Agentti ei ole uskottava, jos se epäonnistuu ihmispelaajalle triviaaleissa tehtävissä [17]. Batesin [22] mielestä uskottavuudeksi riittää pelkästään se, että agentti ei ole ilmeisen tyhmä tai epärealistinen. Agentin epäonnistumiset johtuvat usein *bugeista* eli virheistä (engl. *bug*) ohjelmakoodissa. Lewis et al. [21] jakavat virheet tarkastelussaan kolmeen osaan: itse virheeseen (engl. *error*), ohjelmistovikaan (engl. *fault*) ja toimintahäiriöön (engl. *failure*). Virhe on ilmiö, joka johtaa ohjelmistovikaan. Virhe voi syntyä laitteistoviasta, tai se voi olla suunnittelu- tai toteutusvirhe ohjelmistossa. Virheet ilmenevät ohjelman suoritusvaiheessa vikoina. Toimintahäiriöt ovat poikkeuksia ohjelman käyttäjälle näkyvässä toiminnassa. Virheet ovat ongelma uskotta-

vuudelle silloin, kun ne aiheuttavat agentissa toimintahäiriöitä. Tavallisia epäonnistumisia tekoälylle ovat seiniin törmäily tai muihin esteisiin juuttuminen, kulkuväylien tukkiminen ja reagoimattomuus tulitukseen. Koska toimintahäiriöt ovat poikkeuksia odotetussa käyttäytymisessä, ne ovat suhteellisia. Lewis et al. mukaan odotukset vaihtelevat sekä peligenren että agentin tehtävän ja roolin mukaan. Ammuntapeleissä vastustajan oletetaan toimivan jossakin määrin varomattomasti, mutta ei kuitenkaan tyhmänrohkeasti — ellei agentti sitten esitä esimerkiksi kamikazelentäjää. Pelaajat taas odottavat kumppaniagenteiltaan sopivaa malttia, jotta ne eivät menehtyisi heti alkuunsa tai veisi bonuksia pelaajan nenän edestä. Kuitenkaan liittolaiset eivät saa olla niin varovaisia, ettei niistä ole pelaajalle mitään apua. Tasohyppelypelien (engl. *platformer*) agenteilta ei odoteta monimutkaista käytöstä. Ne voivat pelkästään kulkea lyhyttä reittiä edestakaisin, eikä sitä pidetä toimintahäiriönä. Pelin kehittäjät eivät enää voi lohduttautua sillä, että havaittu virhe on harvinainen. Nykyään jos pelaaja keksii pelistä virheen, hän helposti jakaa löydöksensä verkossa esimerkiksi videopalvelu *YouTubessa*.

Vaikka agentti toimisi häiriöttä, se ei vielä välttämättä ole uskottava. Kuten Livingstonen listasta (ks. edellä) edellä käy ilmi, käyttäytymisen staattisuus, ennustettavuus ja reagoimattomuus riittävät ennen pitkää rikkomaan älykkään toimijan illuusion. Esimerkiksi strategiapeleissä tekoäly alkaa usein toistaa samankaltaista hyökkäystä, vaikka pelaaja torjuu sen aina. Tämä tekee tekoälyn toiminnasta ennalta arvattavaa, ja pelaaja voi vaivatta puolustaa asemiaan varustaen samalla omaa armeijaansa. Jos pelaaja löytää toistuvuutta agentin toiminnassa, hän alkaa helposti hyväksikäyttää tätä agentin heikkoutta. [19] Agenttien toiminta on usein arvattavissa, koska niiden toteutuksessa paljon käytetty äärellinen tilakone (ks. luku 5.1) on luonteeltaan deterministinen. Ei ole mitenkään epätavallista, että pelaaja oppii agentin tilasiirtymiä, eli pystyy sen tämänhetkisen tilan perusteella ennustamaan, miten se alkaa seuraavaksi käyttäytyä. Etenkin moni toimintapeleistä on toteutettu tapahtumapohjaisesti, ja agenttien toiminta on skriptattua eli ennalta säädettyä. [7] Tällaiset agentit ovat varsin staattisia, ja niiden toiminta ennustettavissa ainakin useammalla pelikerralla.

Tyypillinen esimerkki reagoimattomuudesta on agentin muuttumaton käytös, vaikka jotakin sen ympäristössä on pielessä. Esimerkiksi *Fallout Tactics*-pelissä vartija-agentit jatkavat rutiinipartiotaan, vaikka kävelisivät surmatun toverinsa yli. [19] Mac Namee [22] esittää, että agenttien käytöksen pitäisi peleissä muuttua myös pysyvämmin — etenkin jos siinä seikkaillaan laajassa, avoimessa virtuaalimaailmassa. Usein agentit kuitenkin unohtavat kohtaamisten välillä, mitä pelaaja on edellisellä kerralla tehnyt. Esimerkiksi roolipelissä pelaaja voi yhtenä päivänä heittää baari-

mestaria kolpakolla, ja seuraavana päivänä se tervehtii kuin mitään ei olisi tapahtunut. Harvassa pelissä agentti säilyttää minkäänlaista pysyvää tilaa ja monessa ne lisätään peliin vasta kun pelaaja saapuu tietylle alueelle.

Tekoälyn huijaaminen ei sinänsä ole ongelma uskottavuudelle — se haittaa sitä vain, jos se ilmenee pelaajalle [21]. Pelaajat helposti uskovat huijaavan vastustajansa yksinkertaisesti pelaavan paremmin. Esimerkiksi pelaajat voivat pitää agenttia taitavana strategina, vaikka se vain luo tyhjästä yksikköjä sinne, minne niitä tarvitaan. Kuitenkin huijauksen paljastuminen on uskottavuudelle useimmiten erittäin vahingollista. [22] Tosin on huomattava, että uskottavuus kärsii vain, jos pelaaja todella odottaa saavansa tasaväkisen vastuksen. Esimerkiksi on yleisesti tunnettua, että *Civilization IV*-strategiapelissä agentit saavat vaikeammilla tasoilla etuja [29], joita pelaaja ei. Säätäessään vaikeustasoa haastavammaksi pelaaja tietää, että agentit tulevat huijaamaan — ne pelaavat siis täysin odotustenmukaisesti.

4.3 Tekotyperyys

Westin [33] mukaan monissa pelissä, kuten shakissa tai biljardissa, tekoälyllä on valtava etulyöntiasema ihmiseen nähden. Shakkitekoäly voi suorittaa miljardeja laskutoimituksia sekunnissa suunnitellessaan seuraavaa siirtoaan. Biljardiagentti puolestaan voidaan ohjelmoida sellaiseksi, ettei se koskaan tee virheellistä lyöntiä. Ihmiset kuitenkin haluavat agenteilta vastuksen, joka kohtaa heidän oman taitotasonsa. Lidénin [17] mukaan pelaajan päihittävän tekoälyn ohjelmoiminen on suhteellisen helppoa. Esimerkiksi ohjelmoijan on helppo luoda kaikkitietävä tekoäly, jolla on pääsy kaikkiin pelisovelluksen tietorakenteisiin. Tällainen agentti saattaa esimerkiksi tietää, missä sen viholliset ovat, vaikkei sen hahmo eli avatar (sanskr. *avatāra*) pystyisi niitä pelimaailmassa havaitsemaan. Lidén [17] kertoo, että pelaajat ennen pitkää kuitenkin huomaavat tällaiset agentille suodut helpotukset. Vaikka he eivät pystyisi täsmällisesti kertomaan, millä tavalla agentti huijaa, pelaajat kokevat, että tekoäly toimii teennäisellä tavalla. Vaikeampaa on suunnitella agentti, joka häviää pelaajalle tarjoten samalla uskottavan ja viihdyttävän vastuksen. Haasteena on vakuuttaa pelaaja siitä, että agentti on pelissä taitava samalla, kun se antaa pelaajan voittaa.

Yksinkertaisin tapa tehdä agentista erehtyväinen on rajoittaa sen algoritmien suorituskykyä. Useimmiten mitä enemmän agentille annetaan aikaa laskea seuraavaa siirtoa, sitä paremmin se pelaa. Joten jos laskenta-aikaa rajoitetaan, pelaa agentti huonommin. Tämän lähestymistapa tekee kuitenkin helposti tekoälyn käyttäytymisestä epärealistista. Kun laskenta-aikaa rajoitetaan, alkaa agentti tehdä virheitä, joita

ihmispelaaja ei koskaan tekisi. Tällöin sen toiminnan keinotekoinen luonne paljastuu, ja illuusio uskottavasta vastustajasta särkyy. Pelaajan tulisi kokea voittaneensa agentin omasta ansiostaan, ja että vastustaja todella myös kamppaili voitosta. [33]

Parempi tapa rajoittaa tekoälyn kykyjä on ohjelmoida se tekemään tarkoituksellisesti virheitä [17]. Tällaisia agentteja kutsutaan *tekotyperiksi* (engl. *artificial stupidity*). Tekotyperyys terminä ei ole kuitenkaan vakiintunut, vaan sen käytössä esiinty hajaannusta. Esimerkiksi Lewis et al. [21] kutsuvat tekotyperäksi toimintahäiriöistä kärsivää tekoälyä. Tekotyperien agenttien virheistä käytetään englanniksi termejä *intentional* [17] ja *intelligent mistake* [33]. Älykkäät virheet saattavat vaikuttaa tekoälyn haksahduksilta, mutta tarkemmin kyse on huolellisesti suunnitelluista parannuksista sen viihdyttävyyteen [33]. Virheitä tekevän agentin tulee kuitenkin pelata niin, että sen voittaminen tuntuu pelaajasta hyvältä [33].

Eräs tapa toteuttaa tekotyperyyttä on lisätä agentille tasoituksia (engl. *handicap*) pelaajan hyväksi. Esimerkiksi shakissa peliä voi tasoittaa niin, että kokeneempi pelaaja poistaa aluksi muutaman nappuloistaan. Tällä tavalla molemmat pelaavat kykyjensä mukaan, ja peli on silti tasaväkinen. Paremman pelaajan ei tarvitse esittää tyhmää antaakseen heikommalle mahdollisuuden voittaa. Tietokonetta vastaan pelaavat ihmiset eivät kuitenkaan pidä tällaisista ilmeisistä tasoituksista, vaan pelaavat mieluummin täyttä peliä omaa taitotasoaan vastaavan tekoälyn kanssa. [33] Shakissa kukaan ei voita, jos vastustaja ei tee virheitä, ja siksi shakkiagentin tekotyperyys on erityisen tärkeää. Parhaat shakinpelaajat tekevät siirtoja, jotka sekä minimoivat heidän omia että lisäävät vastustajan tekemiä virheitä. Kokeneet pelaajat osaavat hyödyntää vastustajan tekemät virheet ja parhaat saavat vastustajan tekemään virheitä. [20] Shakkitekoäly *Fritz* suunniteltiin sellaiseksi, että se luo pelissä tilanteita, joita kekseliäs pelaaja voi hyväksikäyttää [33]. Jos pelaaja keksii hyödyntää tällaisen mahdollisuuden, alkaa *Fritz* jälleen yrittää voittaa pelin. *Fritz*-tekoälyä ei missään kohdassa ole yksinkertaistettu — päinvastoin: sen toteutus on hienosyisempi, jotta se vaikuttaisi vähemmän taitavalta.

Tarkkuutta vaativissa peleissä, kuten biljardissa tai ammuskelupeleissä, voi tekoäly olla erehtymätön. Jotta agentit tämäntyyppisissä peleissä tarjoaisivat viihdyttävän vastuksen, on niiden tarkkuutta laskettava. Esimerkiksi biljardissa tämä ei välttämättä yksin riitä: Westin [33] toteuttama agentti laski vain lyöntiratoja, eikä tarkastanut mihin valkoinen pallo lyönnin jälkeen jäi. Sattumalta pallo jäi usein agentin kannalta edulliseen asemaan, ja pelaajat tulkitsivat nämä jätöt tarkoituksellisiksi. West ehdottaakin, että samalla tavalla kuin shakissa tekotyperä agentti aktiivisesti tuottaa pelaajalle mahdollisuuksia voittoon, voisi tekoäly biljardipelissä tarkoituksellisesti laskea lyöntiratoja, jotka eivät pussita sen palloja ja jättävät valkoi-

sen pallon sen kannalta epäedulliseen asemaan. Ammuntapeleissä agentit voisivat joskus aktiivisesti hakeutua tulilinjalle tai poistua suojasta lähellä pelaajaa.

Laird ja Duchi [15] muuntelivat botteja *Quake II*-ammuntapelissä neljän ominaisuuden suhteen ja tutkivat, mitkä muutokset vaikuttivat eniten agenttien uskottavuuteen ja havaittuun taitotasoon. Valitut ominaisuudet olivat agentin aggressiivisuus, päätösajat, taktiikkojen monimutkaisuus sekä tähtäyksen tarkkuus. He myös huolehtivat, että agenteilla on samat havainto- ja liikkumismahdollisuudet pelin ympäristössä kuin ihmispelaajilla. Botit esimerkiksi havaitsivat sensoreillaan vain esteettömät kohteet niiden näkökeilassa ja äänet niiden lähettyvillä. Laird ja Duchi mittasivat kyselytutkimuksella ominaisuuksien muuntelun vaikutuksia pelaajien kokemuksiin sekä bottien taitavuudesta että ihmisenkaltaisuudesta. Yllättäen monimutkaisempien taktiikoiden käyttäminen ei kohentanut pelaajien näkemystä agenttien taidoista, mutta vaikutti posiitiivisesti niiden käyttäytymisen inhimillisyyteen. Tekotyperyyden kannalta huomattavaa on myös, että mitä paremmin agentti tähtäsi, sitä taitavampana pelaajat sitä pitivät, mutta botin uskottavuus kärsi. Laird ja Duchi käyttivät testeissä kolmea eri tähtäystarkkuutta: ”heikko”, ”hyvä” ja ”paras”. He olettivat, että heikoiten tähtäävä agentti pelaa huonommin kuin ihminen. Heidän mukaansa keskinkertainen agentti pelaa samalla tavalla kuin ihminen: se ampuu aina kohti vihollista ilman ennakkoa. Parhaiten tähtäävä bottiversio laski ensin, missä kohde on, kun ammus saavuttaa sen nykyisen sijainnin, ja sitten tähtäsi ennakoiden oikeaan kohtaan. Koehenkilöt pitivät heikkoa ja hyvää tähtäystä yhtä uskottavina, ja molempia uskottavampana kuin parasta tähtäystä.

Choi et al. [5] toteuttivat *Quake 3 Arena*-pelistä muokattuun *Urban Combat*-ammuntapeliin ihmisen rajallisia kognitioita simuloivan agentin. Esimerkiksi ryhmä ohjelmoi agentillensa rajalliset sensorit samalla tavalla kuin Laird ja Duchi [15]. Choi et al. kuitenkin esittävät, että rajoitukset eivät yksin riitä tekemään agentista uskottavasti ihmisenkaltaista. Heidän ICARUS-agentillaan on rajoitusten lisäksi hierarkkinen tietämyskanta. Se oppii kannan avulla inkrementaalisesti ja sen käyttäytyminen muuttuu pelin aikana. ICARUS on tavoitehakuinen agentti, ja se seuraa osatavoitteiden saavuttamista. ICARUS-agentin toiminta koostuu sykleistä. Ensin se havainnoi ympäristöään ja tekee päätelmiä tilanteestaan. Sitten se suorittaa toiminnon, joka sekä vastaa tämänhetkisiä uskomuksia että edistää nykyisen tavoitteen saavuttamista. Agentin valitsemat toimet edelleen muuttavat ympäristöä, ja sykli alkaa alusta. Tämän prosessin jatkuessa agentti oppii uusia taitoja, jotka auttavat sitä selviämään tulevista haasteista. Choi et al. esittävät, että ICARUS-agentin tiedot ja taidot vastaavat pelin alussa aloittelevaa ihmispelaajaa. Samoin kuin ihminen, ICARUS myös oppii kokemuksistaan ja pystyy tutkimaan uusia pelin ympäristöjä

systemaattisesti sen sijaan, että tuntisi ne etukäteen. Eräs arkkitehtuurin ongelma on se, että toisin kuin ihminen, kun ICARUS on kerran tutustunut ympäristöön, se muistaa sen jatkossa täydellisesti.

Pokeriagenttien tulee selvitä vaillinaisesta informaatiosta, useasta vastustajasta, riskinhallinnasta, vastustajan mallintamisesta ja vilpistä. Korkeatasoisia pokeritekoälyjä on jo toteutettu, mutta ihmisentasoiseen agenttiin on edelleen matkaa. [26] Vaikka pokeritekoälyissä on vielä kehitettävää, voidaan niiden uskottavuutta silti parantaa ohjelmoimalla ne tekemään sopivia virheitä. Westin [33] toteuttama pokeriagentti määrittää ensin parhaan seuraavan siirron laskemalla voittomahdollisuudet tunnettujen korttien perusteella ja arvioimalla vastustajan käden arvon edellisten panosten mukaan. Sitten se jättää tietyllä todennäköisyydellä tekemättä lasketun siirron — riippuen pelin vaikeustasosta. Koska tekoäly edelleen suorittaa kaikki tarvitut laskut, se ei tee triviaaleja virheitä, joita ihmispelaaja ei koskaan tekisi. Esimerkiksi jos ihmispelaaja korottaa panosta tuntuvasti, ja koneella on kortit, jotka voittavat 75 prosentin todennäköisyydellä, niin Westin tekotyperä agentti voi *foldata* eli luovuttaa (engl. *fold*). Heikot ihmispelaajat usein jatkavat peliä, vaikka heillä on huonot kortit. Samalla tavalla tekotyperä agentti voi jatkaa peliä, vaikka se olisi laskenut voittomahdollisuutensa matalaksi. Westin mukaan tämäntyyppinen tekotyperyys toimii pokerissa hyvin, koska pelin satunnaisen luonteen vuoksi pelaaja ei koskaan voi olla varma siitä, tekikö agentti virheen vai ei.

4.4 Uskottavuuden mittaaminen

Agentin uskottavuus on luonteeltaan hyvin subjektiivinen tekijä: agentti on uskottava, jos pelaaja kokee sen uskottavaksi [19]. Subjektiivisuudesta johtuen uskottavuuden mittaaminen on haastavaa. Uskottavuuskokemuksen syntyminen riippuu pelaajan kulttuuritaustasta sekä pelin että peligenren tuttuudesta [22]. Vielä ei ole kuitenkaan selvää, miten paljon kulttuurierot vaikuttavat uskottavuuden havaitsemiseen, sillä uskottavuustesteissä on tähän mennessä käytetty vain pieniä ryhmiä, joiden jäsenet ovat samalta maantieteelliseltä alueelta. Aloittelevalle pelaajalle voi pelkkä pelaaminen olla niin uusi kokemus, ettei hän erota eritasoisia tekoälyjä toisistaan [19]. Kokenut pelaaja taas voi tuntea tekoälyn niin hyvin, että osaa hyödyntää heikkouksia sen toteutuksessa ja tietää milloin agentti huijaa. Lisäksi lukuisat hienovaraiset agentin piirteet vaikuttavat sen uskottavuuteen. Esimerkiksi Mac Nameen [22] mukaan agentin avatar vaikuttaa siihen, kuinka älykästä käyttäytymistä odotamme siltä. Pelaajat ovat paljon anteeksiantavampia tyylitellyille kuin realistisille hahmoille.

Tärkein menetelmä onnistuneen tekoälyn mittaamisessa on pelitestaus. Pelitestaus on ainoa varma tapa määrittää, milloin tarvitaan hienostuneita tekoälymenetelmiä, ja milloin yksinkertaiset ohjelmointikonstit riittävät. On erittäin tärkeää testata uuden yleisön suhtautumista tekoälyn käyttäytymiseen, sillä pelaajien reaktiot ovat usein yllättäviä. Pelitestaajien valitseminen on ratkaisevan tärkeää. Testaajien tietämys tekoälymenetelmistä vaikuttaa heidän tulkintoihinsa agentin tekemisistä, ja siksi ei esimerkiksi ole hyvä valita testaajia peliteollisuudesta. Kokeneet pelaajat mieltävät agenttien käyttäytymisen ja ratkaisevat pelin haasteet eri tavalla kuin uudet. Tästä syystä jokaiseen testiin pitäisi saada uusi yleisö, vaikka tekoälytoteutusta olisi testikertojen välillä muunneltu. Luonnollisesti testiyleisön pitää olla riittävän suuri, jotta otos olisi edustava. [17]

Pelin pelaaja ei aina välttämättä ole paras tekoälyn uskottavuuden testaaja, sillä hän ei ehdi kovin valppaasti tarkkailla agenttien käyttäytymistä suorittaessaan pelin tehtäviä [15]. Pelaajat eivät aina huomaa agenttien monimutkaisuutta ja toisaalta joskus olettavat väärin agentin olevan toteutukseltaan erityisen hienosyinen [17]. Pelaaja havaitsee agentin toiminnasta vain pienen osan. Pelaaja ei tiedä, mitä agentti tekee, kun se ei ole näköpiirissä, tai mitä informaatiota sen saatavilla on. [15] Toisaalta monissa peleissä agentit eivät tee mitään, jos ne eivät ole pelaajan lähettyvillä, ja aloittavat toimintansa tyhjästä, kun pelaaja saapuu niiden alueelle [22]. Laird ja Duchi [15] käyttivätkin testeissään erillistä tarkkailijaa, jonka tehtävänä oli pelkästään seurata agenttien käyttäytymistä. Tarkkailija voi auttaa testaamisessa, mutta on huomioitava, että tärkeintä on agentin pelaajalle ilmenevä uskottavuus. Uskottavuuden subjektiivisuudesta johtuen sitä voidaan mitata kvantitatiivisesti lähinnä agentteja vertailevilla kyselytutkimuksilla [19]. Mitään absoluuttista mittaria uskottavuuden arvottamiselle ei ole.

Pelin tärkein ominaisuus on sen viihdyttävyys [17]. Luvuissa 4.1 ja 4.2 tarkasteltiin, kuinka sopivan haasteen tarjoaminen on yksi agentin tehtävistä peleissä, ja sen uskottavuuden mittari. Haaste on hyvän pelisuunnittelun tärkein puoli ja merkittävässä osassa viihdyttävän pelikokemuksen syntymisessä [31]. Tästä syystä agenttien uskottavuutta voidaan tietyssä määrin mitata myös tutkimalla pelin viihdearvoa. Tekoälyn uskottavuus vaikuttaa huomattavasti myös pelin immersiivisyyteen [19].

Viihdearvon tutkimukseen sopii Sweetserin ja Wyethin [31] kehittämä *GameFlow*-malli. Mallilla voidaan sekä suunnitella, arvioida että ymmärtää viihdyttävyyttä. Se yhdistelee useita olemassa olevia pelisuunniitteluheuristiikkoja tarkastellakseen viihtymistä kokonaisvaltaisesti. Kun viihde ja haaste yhdistyvät parhaalla mahdollisella tavalla, pelaaja voi läpikäydä niin kutsutun virtauskokemuksen (engl. *flow*).

Virtauskokemuksen synnyttävä toiminta on itsessään palkitsevaa — ihminen ei saa siitä mitään ulkoista hyötyä. Tällainen toiminta on tavoitehakuista sekä sääntöjen rajoittamaa ja vaatii henkistä ponnistusta sekä sopivia taitoja. Tyypillisiä virtauskokemuksen tunnusmerkkejä ovat täydellinen uppoutuminen ja keskittyminen sekä ajan tajun katoaminen. *GameFlow*-mallin arviointikriteerit muodostuvat virtauskokemuksen tekijöistä. Ne ovat:

1. Keskittyminen: pelin tulisi vaatia keskittymistä, ja pelaajan kyetä keskittymään siihen.
2. Haaste: pelin tulisi olla sopivan haastava ja kohdata pelaajan taitotaso.
3. Pelaajan taidot: pelin tulisi tukea pelaajan taitojen kehittymistä.
4. Hallinta: pelaajan tulisi kokea hallitsevansa toimintaansa pelissä.
5. Selkeät tavoitteet: pelin tulisi tarjota selkeitä tavoitteita.
6. Palaute: pelin tulisi tarjota sopivaa palautetta.
7. Immersio: pelaajan tulisi kyetä vaivatta uppoutumaan syvälle peliin.
8. Sosiaalinen kanssakäyminen: pelin tulisi tukea ja luoda mahdollisuuksia sosiaaliseen kanssakäymiseen.

Menetelmällä arvioidaan pelejä pisteyttämällä jokainen kategoria nollasta viiteen. Kun sillä vertailtiin pelejä *Warcraft 3* ja *Lords of EverQuest*, voitti ensimmäinen samoin kuin pelikriitikoiden arvosteluissa. Menetelmän kriteerien painoarvo vaihtelee peligenreittäin. Esimerkiksi immersio on tärkeämpi viihtymisen kriteeri ensimmäisen personaan ammuntapeleissä kuin strategiapeleissä. Osa kriteereistä on vaikeasti asiantuntijan arvioitavissa ja vaatii pelin testauttamista pelaajilla. Esimerkiksi immersion arviointiin ei riitä asiantuntijan selonteko omista kokemuksistaan, vaan vaatii testiyleisön uppoutumisen tarkkailua, kun he pelaavat arvioitavaa peliä.

5 Teknisiä menetelmiä uskottavan agentin toteuttamiseksi

Tässä luvussa keskitytään tarkastelemaan tekoälyä moderneissa, kaupallisissa peleissä — ei lautapelien digitaalisissa vastineissa. Yannakakiksen [34] mukaan perinteisesti tekoälymenetelmiä on peleissä käytetty ei-pelaaja-hahmojen toteuttamiseen. Pelinkehittäjät ovat itsenäisesti löytäneet joukon toimivia menetelmiä tavallisimpiin uskottavan agentin ohjelmoinnin ongelmiin (ks. luku 5.1). Vaikka suuri osa peleissä

käytetyistä menetelmistä on vielä melko yksinkertaisia, on osassa julkaisuja jo hienostuneempia tekniikoita. Kuuluisan pelisuunnittelijan Peter Molyneux'n *Black & White*-pelissä agentit käyttivät useita koneoppimisen menetelmiä, kuten perseptroniverkkoa ja päätöspuita. Hiiviskelyyn keskittyvä *Thief*-peli oli uraauurtava agenttien sensoreiden toteutuksessa. Osa kehittäjistä pitääkin pelien agenttien toteuttamisen ongelmia jo käytännössä ratkaistuina. He myös katsovat, että parannukset olemassa oleviin toteutuksiin vaatisivat valtavasti resursseja.

Uskottavan agentin toteuttamisen ongelma jakautuu useaan osaongelmaan agentin roolin ja pelin genren mukaan (ks. luku 3.2). Esimerkiksi agentin, joka toimii ammuntapelissä taktisen vastustajan roolissa, tulee vuorovaikuttaa ympäristönsä, toisten agenttien ja ihmispelaajien kanssa sekä sopeutua näihin. Sen käyttäytymisen tulee esimerkiksi reaktioajoiltaan vastata ihmispelaajan käyttäytymistä, ja sen tulee pystyä sekä navigoimaan että järjestämään toimintaansa. Lisäksi agentin toteutus ei saisi olla laskennallisesti raskas tai kovin monimutkainen ohjelmoida. Koska agentilla on laajalti vaatimuksia, sen toteuttaminen koskettaa useita eri tekoälytutkimuksen osa-alueita, kuten kognitioiden mallintaminen, vastustajan mallintaminen sekä eleiden tunnistaminen. Jotta agentti pystyisi esimerkiksi navigoimaan ammuntapelin monimutkaisessa ympäristössä, sen tulee kyetä spatiaaliseen ja temporaaliseen päättelyyn sekä polunhakuun. Sen tulee reagoida ympäristönsä muutoksiin, ja sen sensorien tulee imitoida ihmispelaajan havainnointia. [16]

Yannakakiksen [34] mukaan kehittäjät ovat alkaneet nykyään käsitellä pelien tekoälyä kokonaisvaltaisemmin: hyvä tekoäly ei aina välttämättä tarkoita uskottavampia agentteja. Toisin sanoen, tekoälyä on alettu tarkastella muissakin tehtävissä kuin agentin ohjauksessa. Yannakakis pitää oleellisimpina uusina pelin tekoälyn tutkimusalueina proseduraalista sisällöntuottamista, uusia agentin toteutustapoja, pelikokemuksen mallintamista sekä mallintamiseen vahvasti liittyvää tiedonlouhintaa.

Yksi uusi pelien tekoälyn tutkimusalue on pelikokemuksen mallintaminen (engl. *player experience modeling*). Pelikokemusta mallintava tekoäly pyrkii tunnistamaan pelaajan tyypin ja päättelemään seikkoja esimerkiksi tämän mielentilasta. Mallintavan tekoälyn avulla voisi esimerkiksi mukauttaa peliä sopivammaksi erilaisille pelaajille. Mallia varten kerätään aluksi tietoja eri tavoilla: pelaajakyselyillä, tarkkailemalla pelaajia pelin aikana tai seuraamalla pelinsisäisiä muuttujia. Seuraavaksi mallintava tekoäly opetetaan tunnistamaan ja arviomaan pelaajia kerätyn tiedon avulla. Sen toteutuksessa voidaan käyttää eri tekoälytekniikoita, kuten neuro- tai Bayesverkkoa. [34] Esimerkiksi Spronck ja den Teuling [29] tutkivat pelaajatyypin mallintamista *Civilization IV*-strategiapelissä. Aluksi he keräsivät tietoja mallia varten

pelauttamalla agentteja toistuvasti toisiaan vastaan. Tarkkailtavana oli kuusi muuttujaa: aggressio, eli taipumus sodan julistamiseen ja laajentumiseen, sekä resurssien jakautuminen kulttuuriin, talouteen, väestönkasvuun, armeijaan, uskontoon ja tieteeseen. Sitten he opettivat neuroverkkonsa kerätyn tiedon avulla. Spronck ja den Teuling havaitsivat, että malli pystyi tunnistamaan agenttien pelityylejä, mutta se ei kyennyt kovin hyvin erottamaan ihmispelaajia toisistaan. He esittivät, että jotkin valituista muuttujista eivät olleet kovin valideja. Esimerkiksi väestönkasvu on tärkeää kaikille pelaajille, eikä siten erottele pelityylejä. Malli kuitenkin erotti paremmin toisistaan kokeneet kuin aloittelevat pelaajat.

Yannakakis [34] tarkoittaa proseduraalisella sisällöntuotannolla algoritmeja, jotka tuottavat automaattisesti sisältöä peliin. *Sisältö* tarkoittaa pelissä mitä tahansa muunneltavaa elementtiä, joka voi vaikuttaa pelikokemukseen. Tällaisia ovat esimerkiksi maasto, musiikki, tehtävät (engl. *quest*) ja säännöt. Nareyekin [23] mukaan pelaajien kasvavat realismiodotukset tekevät pelimaailmojen luomisesta jatkuvasti haastavamman ja kalliimman tehtävän. Proseduraalisen sisällöntuotannon avulla voidaan helposti tuottaa suuria pelimaailmoja. Esimerkiksi *Elder Scrolls IV: Oblivion*-pelissä metsiköt generoidaan automaattisesti. Nareyek kuitenkin varoittaa, että automaattisten menetelmien käytössä tulee olla huolellinen, sillä algoritmit voivat helposti tuottaa epäjohdonmukaista sisältöä — esimerkiksi talon jokeen. Proseduraalisen sisällöntuotannon avulla voidaan myös räätälöidä peliä pelaajalle sopivammaksi. Esimerkiksi Shaker et al. [34] yhdistivät sisällöntuotannon ja pelikokemuksen mallintamisen menetelmiä toteuttaessaan järjestelmän, joka tuotti uusia tasoja *Super Mario*-peliin. Järjestelmä pyrki mallin avulla tuottamaan esimerkiksi mahdollisimman ”hauskan” tason. Tiedot järjestelmän malliin oli kerätty pyytämällä pelaajia arvostelemaan pelin tasoista muokattuja esimerkkejä.

5.1 Yleisiä pelien agenttitekniikoita

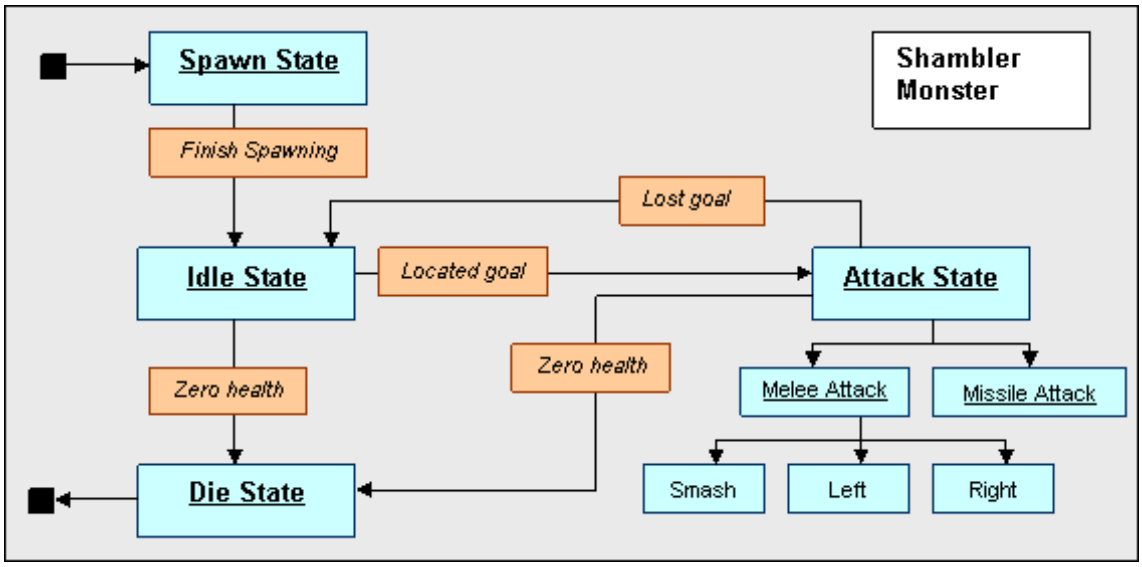
Kaupalliset pelit käyttävät usein melko yksinkertaisia agenttitekniikoita akateemisiin ja teollisiin sovelluksiin verrattuna [7]. Pääosin tämä johtuu siitä, että pelikehityksen mielenkiinto on keskittynyt grafiikan parantelemiseen, ohjelmoijat eivät luota epädeterministisiin tekniikoihin, kuten koneoppimiseen, ja tekoälyn kehittämiselle ei varata riittävästi aikaa [22]. Toisaalta akateemisella tutkimuksella ei ole ollut tarjota merkittävästi parempia, peleihin skaalatuvia menetelmiä [34]. Peleissä toteutuksia hallitseekin muutama yksinkertainen, deterministinen ja laskennallisesti tehokas tekniikka, kuten A*-algoritmi, äärellinen ja sumea tilakone sekä joukko keinoelämän menetelmiä, kuten parveilualgoritmit [7]. Seuraavaksi luvussa tarkas-

tellaan A*-algoritmia ja äärellistä tilakonetta esimerkkeinä yleisimmistä agenttitekniikoista.

Mac Nameen [22] mukaan toimiva polunhaku on tärkeää lähes jokaisessa peligenressä. Siksi pelinkehittäjät ovat kiinnittäneet siihen enemmän huomiota kuin mihinkään muuhun tekoälyn osa-alueeseen. Ensimmäinen askel polunetsinnässä on täyttää pelin maailma diskreetillä navigointidatalla. Navigointidata koostuu usein pelimaailmaan sijoitetuista näkymättömistä solmuista (engl. *node*), joihin sisältyy tieto siitä, mihin miltäkin alueelta pääsee. Agentit käyttävät pelissä navigointidataa polunhakuun, ja suosituin menetelmä tähän on A*-algoritmi. Tämä heuristinen haku laskee kustannusarvion kullekin solmulle seuraavan kaavan mukaan:

$$f(n) = g(n) + h(n),$$

missä $f(n)$ on solmun n kokonaiskustannus, $g(n)$ kustannus polun alusta solmuun n ja $h(n)$ arvio jäljellä olevasta kustannuksesta solmusta n tavoitesolmuun [30]. A*-algoritmi löytää lyhimmän reitin niin kauan kuin kustannusarvio $h(n)$ on korkeintaan yhtä suuri kuin todellinen kustannus tavoitesolmuun. A* ei kuitenkaan välttämättä aina toimi riittävän nopeasti, etenkään jos navigoitava kartta on suuri tai funktio $h(n)$ on huonosti valittu. Polunhaku on harkitsevan agentin (ks. luku 3.1) piirre, koska siinä agentti etsii reitin määränpäähänsä eli muodostaa suunnitelman tavoitteensa saavuttamiseksi.



Kuva 1: *Shambler*-agentin tilakone pelistä *Quake* [3].

Äärellinen tilakone (ks. kuva 1) on varsin yleisesti käytetty tekoälymenetelmä. Tilakone on yksinkertainen järjestelmä, joka koostuu äärellisestä joukosta tiloja se-

kä näiden välisistä suunnatuista siirtymistä. Kun tilakone ohjaa tekoälyn käyttäytymistä, tilat ilmenevät pelimaailmassa erilaisina agentin toimintoina. Tilasiirtymät kuvaavat kuinka muutokset pelimaailmassa tai avatarin ominaisuuksissa vaihtavat agentin tilasta toiseen. Tilakoneiden etuina ovat helppotajuisuus sekä vähäinen laskenta-ajan ja muistin kulutus. [22] Tilakoneesta on kuitenkin haastavaa tehdä täysin kattava, sillä on vaikeaa ennakoida kaikkia mahdollisia pelitilanteita [7]. Lisäksi mitä enemmän tiloja koneeseen ohjelmoidaan, sitä vaikeammaksi muuttuu tilasiirtymien suunnittelu [22]. Tilakone myös helposti tekee agentin käyttäytymisestä jäykkää ja yllätyksetöntä [7]. Tilakone on reagoivan agentin piirre, koska siinä tilasiirtymän ehdot kuvaavat herätteet, ja muutokset agentin tilassa vasteet.

Huijaaminen on tekniikka, jonka avulla on helppo kasvattaa agentin tarjoamaa haastetta tai kiertää teknisiä ongelmia. Useimmiten huijausta käytetään toiminta- ja strategiapelissä. Esimerkiksi agentilla voi toimintapelissä olla rajattomasti ammuksia ja strategiapelissä loputtomasti resursseja armeijansa varustamiseen. [22] Varsin yleinen huijaustapa ajopeleissä, kuten *Burnout 3: Takedown*[4], on ollut niin kutsuttu kuminauhaefekti. Siinä agentti kuroo hetkessä kiinni pelaajan saavuttaman etumatkan, jos tämä karkaa liian kauas.

5.2 Tekniikoita uskottavuuden ja pelikokemuksen parantamiseksi

Tässä luvussa esitellään esimerkinomaisesti teknisiä menetelmiä, jotka voivat parantaa agentin uskottavuutta. Tekstissä tuodaan lyhyesti esille, kuinka pelien kasvava realismi muuttaa tekoälyn toteuttamisen vaatimuksia. Luvussa esitellään lisäksi erilaisia keinoja parantaa pelikokemusta, muun muassa pelin haastetta ja sisältöä mukauttamalla.

Agentin avatar, liikkuminen ja elehtiminen vaikuttavat sen uskottavuuteen [22]. Koska navigointipisteet ovat kohtalaisen harvassa, voi aiemmin esiteltyä A*-algoritmia (ks. luku 5.1) käyttävä agentti liikkua melko kulmikkaasti ja epärealistisesti [9]. Lisäksi nykyaikaiset fysiikkamoottorit mahdollistavat varsin dynaamisia muutoksia pelin ympäristöön, mutta perinteistä A*-algoritmia käyttävät agentit eivät näihin pysty reagoimaan. A*- ja muiden polunhakualgoritmien tarvitsemaa navigointidataa ei voida tuottaa reaaliaikaisesti tehtävän laskennallisen vaativuuden vuoksi. Koska navigointidata on laskettu etukäteen peliä ladattaessa, on agentin kuva maailmasta staattinen.

Graham et al. [9] mukaan tavanomainen A*-algoritmi ei myöskään ota huomioon minkäänlaisia taktisia seikkoja laskiessaan lyhintä polkua. Se ei esimerkiksi

[4] `http://www.giantbomb.com/rubber-band-ai/92-35/games/`

laske reittiä, joka tarjoaa eniten mahdollisuuksia vihollisen tulelta suojautumiseen. Ongelma voidaan ratkaista muuttamalla sitä, minkälaisia tietoja solmusta algoritmin kustannusheuristiikkaa käyttää. Lidén [18] onkin esittänyt menetelmän, jonka avulla navigointidatan esilaskennassa voidaan solmuihin lisätä tietoja kohdan vaarallisuudesta. Siinä vaarallisuus muodostuu alueen näkyvyydestä eri suuntiin. Lidénin menetelmässä tieto on kuitenkin edelleen staattista, eikä agentti sen avulla pysty varautumaan vastustajan alati muuttuvaan uhkaan.

Agentin dynaamisempi ja todenmukaisempi liikkuminen voitaisiin toteuttaa käyttämällä reaaliaikaisia polunhakualgoritmeja. Reaaliaikainen A*-algoritmi on eräs soveltuva menetelmä. Se poikkeaa tavallisesta A*-algoritmista ainoastaan siinä, että lyhimmän polun löytymiselle on asetettu aikaraja. Jos polkua ei löydy aikarajan umpeutumiseen mennessä, käytetään tämänhetkisten tietojen mukaan parasta polkua. Toisen mahdollisuuden tarjoavat ohjausalgorimit (engl. *steering algorithm*). Tällaisia ovat esimerkiksi säteenheittoon (engl. *ray casting*) tai voimiin perustuva ohjaus. Edellisessä agentti etsii sensoreillaan ympäristöstä mahdollisimman esteettömiä linjoja, ja jälkimmäisessä esteet ikään kuin hylkivät agenttia sitä enemmän, mitä lähempänä se on. Graham et al. tutkivat myös koneoppimisen mahdollisuuksia agentin navigoinnissa. Tutkimuksen agentit käyttivät toteutuksessaan neuroverkkoa. Tulosten perusteella neuroverkko pystyi oppimaan reaaliaikaisen polunhaun perustaidot, mutta se osoittautui erittäin haastavaksi opettaa.

Pelit ovat pitkäveteisiä, jos ne ovat liian helppoja ja turhauttavia, jos ne ovat liian haastavia. Mahdollisuudet muuttaa vaikeustasoa ovat kuitenkin useimmissa peleissä vielä melko joustamattomia. Tämä voi johtaa siihen, että pelaajan taidot ja pelin tarjoama haaste eivät kohtaa. [12] Pelien mukautuvuutta voidaan parantaa ohjelmoimalla ne muuttamaan vaikeustasoaan tai sisältöään dynaamisesti mallintamalla pelaajan taitoja ja mieltymyksiä [4]. Mukautuminen tulee kuitenkin suorittaa hienovaraisesti, sillä pelaajat tuntevat helposti tulleensa huijatuksi, jos he huomaavat pelin muuttuvan heitä varten [12].

Charles et al. [4] esittävät, että peli on sitä immersiivisempi (ks. luku 4.4), mitä paremmin se vastaa pelaajan yksilöllisiin tarpeisiin. Nykyistä mukautuvammat pelit myös tavoittaisivat suuremman ja monipuolisemman yleisön. Heidän mukaansa pelit voivat mukautua muuttamalla pelaajan hahmoa, agenttien hahmoja tai pelin tilaa tai ympäristöä. Pelaaja tuntee hahmonsa paremmin kuvastavan häntä, jos se muuttuu pelin tapahtumien ja pelaajan pelityylin mukaan. Peli voi muuttua myös tekemällä vastustaja-agenteista haastavampia tai muuttamalla kumppaniagenttien käyttäytymistä. Ei-pelaaja-hahmot voivat antaa vihjeitä tai avustaa pelaajaa tämän tarpeiden ja mieltymysten mukaan. Näitä hahmoja voidaan käyttää mitä mielikuvi-

tuksellisimmilla tavoilla vaikuttamaan pelin tarinaan ja pelaajalle mahdollisiin valintoihin. Myös pelin ympäristö voi mukautua. Esimerkiksi toimintapelissä ympäristöön voi syntyä luodinreikiä, tai esineet rikkoutua tulituksessa. *Fable*-roolipelissä taas maisemat muuttuvat pelaajan tekemien eettisten valintojen mukaan.

Tavanomaisin tapa mallintaa pelaajaa on ollut vastustajan mallintaminen (engl. *opponent modeling*) [29]. Vastustajamalli on abstrakti kuvaus pelaajasta tai tämän käyttäytymisestä pelissä [11]. Malli voi sisältää tietoja esimerkiksi kohteen mieltymyksistä, strategiasta, taidosta, voimavaroista, heikkouksista ja tietämyksestä. Mallia käytetään sekä vastustajan päihittämiseen että tämän avustamiseen — esimerkiksi pelaajaa harjoittaessa. Van den Herikin et al. [11] mukaan mallintamista on tutkittu akateemisesti esimerkiksi pokerissa ja vangin dilemman ratkaisemisessa, mutta kaupallisissa peleissä se on vielä melko vähän käytetty tekniikka. Iteroidussa eli toistetussa vangin dilemmassa agentin tulee mallintaa vastustajansa halukkuus yhteistyöhön valitakseen parhaan strategian. Pystyäkseen pelaamaan pokeria riittävän hyvin, agentin tulee mallintaa vastustajansa heikkoudet. Pelissä agentti käyttää mallia vastustajan seuraavan teon ennustamiseen sekä tämän voittomahdollisuuksien arvioimiseen. Van den Herik et al. esittävät, että vastustajan mallintaminen olisi parempi menetelmä agenttien toiminnan ohjaukseen kaupallisissa peleissä kuin nykyisin tehtävään käytetty skriptaus.

Hunicke [12] kehitti tutkimuksessaan *Half-Life*-peliä varten Hamlet-järjestelmän, jonka tehtävänä oli dynaamisesti muunnella pelin vaikeustasoa. Hamlet arvioi jatkuvasti pelaajan menestymistä eri suureiden, kuten menehtymistodennäköisyyden avulla ja yrittää muuttaa peliä noudattaen mukautustavoitettaan. Yksi sen vaihtoehtoisista mukautustavoitteista on *mukavuus*, jolloin se huolehtii, että pelaaja kokee pelin haastavaksi, ja että tämän terveydentila pysyy tietyissä rajoissa. Hamlet voi muuttaa peliä esimerkiksi lisäämällä pelaajaa auttavien varusteiden määrää pelimaailmassa tai tekemällä pelaajan hahmosta tulivoimaisemman. Hunicke testautti Hamlet-järjestelmää tietotekniikan opiskelijoilla. Hamletin tekemät muutokset eivät vaikuttaneet aloittelevien pelaajien kokemuksiin, mutta pelaajat, joille ammuntapelit ovat tuttuja, viihtyivät hieman paremmin sen ohjaaman *Half-Lifen* kuin pelin tavanomaisen version parissa. Hunicke havaitsi myös, että suurimmalla osalla koehenkilöistä oli ideoita siitä, millä tavalla peli voisi auttaa heitä. Testaajat toivoivat esimerkiksi parempia valaistus- ja äänivihjeitä, apua tähtäyksessä ja vihjeitä varusteiden sijainnista.

Esimerkki kaupallisesta, dynaamisesti mukautuvasta pelistä on suomalaisen Remedy Entertainmentin kehittämä *Max Payne*. Peli seuraa pelaajan selviytymistä tilastoimalla esimerkiksi tämän osumatarkkuutta sekä terveydentilaa, ja säätää sitten

näiden muuttujien perusteella vihollisten määrää ja haastavuutta. [4]

Yksi Hunicken [12] kehittämän Hamlet-järjestelmän tavoista mukauttaa *Half-Life*-peliä oli säädellä sen agenttien osumatarkkuutta. Hunicke ei kuitenkaan artikkelissaan erittele eri muutosten vaikutuksia pelikokemukseen, joten ei ole selvää, millä tavalla esimerkiksi madallettu tarkkuus vaikutti pelin viihdyttävyyten. Suurin osa järjestelmän tekemistä muutoksista kosketti joko pelaajan avatarin kykyjä tai pelimaailmaa, ei agenttien käyttäytymistä. *Max Payne*-peli sääteli vaikeustasoaan muuttamalla vihollisagenttien määrää tai kestävyyttä, ei niiden käyttäytymistä [4]. West [33] toteutti agentteja peliin *World Series of Poker*. Ne olivat tarkoituksellisesti ohjelmoitu tekemään virheitä tietyllä todennäköisyydellä. Todennäköisyys riippui pelaajan valitsemasta vaikeustasosta. Pelin vaikeustaso oli staattinen, mutta voisi olla mahdollista toteuttaa järjestelmä, joka mukauttaa haastetta dynaamisesti säätämällä agenttien virhetodennäköisyyksiä.

Lidén [17] antaa esimerkin yksinkertaisesta tavasta säilyttää pelin haaste sopivana tekotyperyyden avulla: *Half-Lifen* kehittäjät ohjelmoivat pelin agentit taistelemaan ”kungfu”-tyylillä, eli hyökkäämään korkeintaan kaksi kerrallaan niiden lukumäärästä riippumatta. Agentit eivät koordinoineet yhteistoimintaansa, vaan pelissä oli erillinen järjestelmä, joka jakoi niille hyökkäysvuoroja. Kun agentit odottivat vuoron vapautumista, ne esimerkiksi hakeutuivat suojaan tai latasivat aseensa. Lidénin mukaan agentit säilyttivät uskottavuutensa, vaikka ne jättivät hyödyntämättä ylivoimansa. Lidén luettelee myös muita mahdollisia tapoja säädellä haastetta: agentti voidaan esimerkiksi ohjelmoida päästämään ääni, kun pelaaja lähestyy, jotta se ei pääsisi yllättämään tätä. Se voidaan säätää vetäytymään, jos pelaaja joutuu pahasti alakynteen. Peliä helpottaa myös, jos sen agentit ilmaisevat aikeensa pelaajalle, esimerkiksi antamalla näennäisesti käskyjä toisilleen.

6 Yhteenveto

Tutkielmassa tarkasteltiin aluksi tekoälyä, ja sen ilmenemistä peleissä. Tekoälyllä tarkoitetaan tietokoneohjelmaa, joka toimii näennäisen älykkäällä tavalla. Se ei kuitenkaan ymmärrä tekemisiään [27]. Tekoälyn tärkein tehtävä peleissä on ohjata erilaisia hahmoja [22]. Pelien tekoälyn ensisijaisena tehtävänä ei ole esitellä mahdollisimman edistyneitä toteutustekniikoita, vaan tarjota viihdettä [17]. Teknisesti edistyneempi tekoäly ei välttämättä tuota uskottavampia pelihahmoja. Tämä unohtuu helposti jopa pelien ohjelmoijilta.

Voidaan ajatella, että pelin jokaista hahmoa ohjaa oma tekoälynsä. Tekoälyä tarkastellaankin peleissä usein itsenäisinä osaohjelmina eli agentteina [22]. Tutkiel-

massa esiteltiin lyhyesti agenttien toteutustapoja, sekä erityyppisten pelien ja roolien niille asettamia vaatimuksia. Tutkielmassa keskityttiin erityisesti agenttien uskottavuuden käsittelemiseen. Uskottavuus on suhteellinen käsite, joka riippuu sekä agentin tehtävästä pelissä, pelin genrestä että pelaajan odotuksista [19]. Tärkeä agentin uskottavuuden tekijä on sen pelaajalle tarjoama haaste. Haasteen tulee olla sekä riittävä että pelaajan taidoille sopiva. Uskottavan agentin on tärkeämpää tarjota tasaväkinen vastus kuin päihittää pelaaja. Peli on sitä viihdyttävämpi, mitä sopivamman haasteen se tarjoaa. Uskottavuutensa säilyttävä agentti myös parantaa pelin immersiivisyyttä. Immersiivisyys on tärkeä pelin viihdyttävyyden osatekijä.

Tekotyperyys eli virheiden tarkoituksellinen ohjelmoiminen tekoälyyn liittyy vahvasti agenttien uskottavuuteen. Käsitteellä ei tutkielmassa tarkoiteta toimintahäiriöistä kärsivää tekoälyä — vaikka tekoälyn vikoja voidaankin naamioida tarkoituksellisiksi virheiksi [17]. Tekotyperyys on pitkään ollut ilmiö ilman käsitettä: jo vuonna 1950 Turing [32] huomio suunniteltujen virheiden tarpeellisuuden ehdottamassaan agentissa. Ohjelmoimalla pelin agentti rajalliseksi ja erehtyväiseksi, sen tarjoama haaste voidaan paremmin säilyttää sopivana [33]. Uskottava, tekotyperä agentti tarjoaa tasaväkisemmän vastustajan, sillä se ainakin näennäisesti noudattaa samoja sääntöjä ja rajoituksia kuin ihmispelaaja. Agentin virheiden suunnittelu myös säilyttää sen käyttäytymisen uskottavampana kuin sen algoritmien suorituskyvyn rajoittaminen, esimerkiksi niiden käyttämän laskenta-ajan lyhentäminen.

Vaikka joukko pelinkehittäjiä pitää agentin toteuttamisen ongelmia jo käytännöllisesti katsoen ratkaistuina, riittää agentin uskottavuutta parantavissa tekniikoissa vielä kehitettävää. Esimerkiksi agentit kärsivät edelleen usein toimintahäiriöistä [21]. Toisaalta pelien alati kasvava realismi asettaa uusia haasteita myös perinteisille toteutustekniikoille, kuten A*-algoritmille (ks. luku 5.2). Pelit ja pelien tekoälyt tarjoavat nykyään myös kohteen akateemiselle tutkimukselle [16]. Näytönohjaimet antavat tilaisuuden käyttää peleissä vaativampia tekoälymenetelmiä, koska niiden ansiosta keskusprosessorille jää aikaa suorittaa muitakin tehtäviä kuin hahmontaa (engl. *render*) grafiikkaa. Tulevaisuudessa Yannakakiksen esittelemät tekoälytekniikat (ks. luku 5) yleistynevät edelleen. Esimerkiksi proseduraalista sisällöntuotantoa tarvitaan yhä useammin, koska pelaajat odottavat yhä laajempia ja todenmukaisempia pelimaailmoja [23]. Yannakakis [34] myös olettaa, että tutkimuksen painopiste siirtyy jatkossa vaihtoehtoisiin tapoihin toteuttaa pelin agentti. Yksi uusista tavoista on muunnella pelimaailma ja -mekaniikkaa agentin uskottavuuden parantamiseksi sen sijaan, että puututtaisiin itse agentin toteutukseen.

Lähteet

[1] Michael Booth, *The AI Systems of Left 4 Dead*, saatavilla WWW-muodossa <URL: http://www.valvesoftware.com/publications/2009/ai_systems_of_l4d_mike_booth.pdf>, viitattu 20.6.2012.

[2] Bobby D. Bryant, *Evolving Visibly Intelligent Behavior for Embedded Game Agents*, väitöskirja, The University of Texas, Austin, Teksas, Yhdysvallat, 2006.

[3] Jason Brownlee, *A Practical Analysis of FSM within the domain of first-person shooter (FPS) computer game*, saatavilla WWW-muodossa <URL: http://ai-depot.com/FiniteStateMachines/FSM.html>, viitattu 21.6.2012.

[4] Darryl Charles, Michael McNeill, Moira McAlister, Michaela Black, Adrian Moore, Karl Stringer, Julian Kücklich ja Aphra Kerr, *Player-Centred Game Design: Player Modelling and Adaptive Digital Games*, CHI '06 extended abstracts on Human factors in computing systems, ACM, New York, New York, Yhdysvallat, 2006, s. 1731–1734.

[5] Dongkyu Choi, Tolga Konik, Negin Nejati, Chunki Park ja Pat Langley, *A Believable Agent for First-Person Shooter Games*, Artificial Intelligence and Interactive Digital Entertainment, The AAAI Press, 2007, s. 71–73.

[6] Henry Ridgely Evans, *The Romance of Automata*, The Open Court, 1905 (2010), s. 131–140.

[7] Chris Fairclough, Michael Fagan, Brian MacNamee ja Pádraig Cunningham, *Research Directions for AI in Computer Games*, Proceedings of the Twelfth Irish Conference on Artificial Intelligence and Cognitive Science, National University of Ireland, Maynooth, Faculty of Philosophy, Maynooth, Irlanti, 2001, s. 333–344.

[8] Stan Franklin ja Art Graesser, *Is It an agent, or just a program?: A taxonomy for autonomous agents*, Intelligent Agents III Agent Theories, Architectures, and Languages, (Möller, Jörg, toim.), kirjasarjasta Lecture Notes in Computer Science, Springer, Berlin / Heidelberg, Saksa, 1997.

[9] Ross Graham, Hugh McCabe ja Stephen Sheridan, *Realistic Agent Movement in Dynamic Game Environments*, Proceedings of DiGRA 2005 Conference: Changing Views Worlds in Play, Authors & Digital Games Research Association DiGRA, 2005.

[10] Stevan Harnad, *Minds, Machines and Turing*, Journal of Logic, Language and Information, 9 (2000), s. 425–445.

[11] H.J. van den Herik, H.H.L.M. Donkers, P.H.M. Spronck, *Opponent Modelling and Commercial Games*, IEEE 2005 Symposium on Computational Intelligence and Games, CIG'05, 2005, s. 15–25.

[12] Robin Hunicke, *The Case for Dynamic Difficulty Adjustment in Games*, Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology, ACM, New York, New York, Yhdysvallat, 2005, s. 429–433.

[13] Randolph M. Jones, John E. Laird, Paul E. Nielsen, Karen J. Coulter, Patrick Kenny ja Frank V. Koss, *Automated Intelligent Pilots for Combat Flight Simulation*, AI Magazine, 20 (1999), s. 27–41.

[14] Marina Krol, *Have We Witnessed a Real-Life Turing Test?*, Computer, 32 (1999), s. 27–30.

[15] John E. Laird, John C. Duchi, *Creating Human-like Synthetic Characters with Multiple Skill Levels: A Case Study using the Soar Quakebot*, 2001 AAAI Spring Symposium on Artificial Intelligence and Computer Games, konferenssijulkaisu, s. 54–58.

[16] John E. Laird, Michael van Lent, *Human-Level AI's Killer Application: Interactive Computer Games*, AI magazine, 22 (2001), s. 15–25.

[17] Lars Lidén, *Artificial Stupidity: The Art of Intentional Makes*, kirjassa AI Game Programming Wisdom 2, Charles River Media, Lontoo, Yhdistynyt kuningaskunta, 2003, s. 41–48.

[18] Lars Lidén, *Using Nodes to Develop Strategies For Combat with Multiple Enemies*, Artificial Intelligence and Interactive Entertainment I, AAAI Press, Menlo Park, Kalifornia, Yhdysvallat, 2001, s. 59–63.

[19] Daniel Livingstone, *Turing's Test and Believable AI in Games*, Computers in Entertainment — Theoretical and Practical Computer Applications in Entertainment, 4 (2006)

[20] Steven A. Lopez, *Intelligent mistakes*, saatavilla WWW-muodossa <URL: `http://www.chessbase.com/newsdetail.asp?newsid=2579`>, viitattu 1.6.2012.

[21] Chris Lewis, Jim Whitehead, Noah Wardrip-Fruin, *What Went Wrong: A Taxonomy of Video Game Bugs*, Proceedings of the Fifth International Conference on the Foundations of Digital Games, FDG '10 konferenssijulkaisu, 2010, s. 108–115.

[22] Brian Mac Namee, *Proactive Persistent Agents — Using Situational Intelligence to Create Support Characters in Character-Centric Computer Games*, väitöskirja, University of Dublin, Trinity College, Dublin, Irlanti, 2004.

[23] Alexander Nareyek, *Game AI is Dead. Long Live Game AI!*, IEEE Intelligent Systems, 22 (2007), s. 9–11.

[24] Michael Negnevitsky, *Artificial Intelligence: a guide to intelligent systems*, Pearson Education, Upper Saddle River, New Jersey, Yhdysvallat, 2005.

[25] Jonathan Schaeffer, Neil Burch, Yngvi Björnsson, Akihiro Kishimoto, Martin Müller, Robert Lake, Paul Lu ja Steve Sutphen, *Checkers Is Solved*, Science, 317 (2007), s. 1518-1522.

[26] Jonathan Schaeffer ja H. Jaap van den Herik, *Games, computers and artificial intelligence*, Artificial Intelligence, 134 (2002), s. 1–8.

[27] John R. Searle, *Minds, Brains, and Programs*, Behavioral and Brain Sciences, 3 (1980), s. 417–457.

[28] Stuart M. Shieber, *Lessons from a Restricted Turing Test*, Communications of the ACM, 37 (1994), s. 70–78.

[29] Pieter Spronck ja Freek den Teuling, *Player Modeling in Civilisation IV*, kirjassa Proceedings of the Sixth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (toim. G. Michael Youngblood ja Vadim Bulitko), The AAAI Press, Stanford, Kalifornia, Yhdysvallat, 2010.

[30] Bryan Stout, *Smart Moves: Intelligent Pathfinding*, Game Developer, 10 (1996).

[31] Penelope Sweetser ja Peta Wyeth, *GameFlow: A Model for Evaluating Player Enjoyment in Games*, Computers in Entertainment — Theoretical and Practical Computer Applications in Entertainment, 3 (2005).

[32] Alan Turing, *Computing machinery and Intelligence*, Mind, LIX (1950), s. 433–460.

[33] Mick West, *Intelligent Mistakes: How to Incorporate Stupidity Into Your AI Code*, saatavilla WWW-muodossa <URL: `http://www.gamasutra.com/view/`

feature/3947/intelligent_mistakes_how_to_.php?print=1>, viitattu 1.6.2012.

[34] Georgios N. Yannakakis, *Game AI revisited*, Proceedings of the 9th conference on Computing Frontiers, ACM, New York, New York, Yhdysvallat, 2012, s. 285–292.