

# Tekoälystä tietokonepeleissä

TURUN YLIOPISTO  
Informaatioteknologian laitos  
Tietojenkäsittelytiede  
LuK-tutkielma  
13.10.2008  
Vesa Nieminen

TURUN YLIOPISTO  
Informaatioteknologian laitos

NIEMINEN, VESA: Tekoälystä tietokonepeleissä

LuK-tutkielma, 34 s.  
Tietojenkäsittelytiede  
lokakuu 2008

---

Tekoäly on tieteenala ja insinööritaito (engl. engineering), jossa suunnitellaan ja rakennetaan älykkäitä koneita ja erityisesti älykkäitä tietokoneohjelmia. Tekoälytekniikoita hyödyntäviä tunnettuja sovelluksia ovat esimerkiksi puheen, kasvojen ja käsialan tunnistaminen, ihmiskielien ymmärtäminen, älykkäät roskapostisuodattimet, asiantuntijajärjestelmät ja pelien pelaaminen.

Tietokonepelit on interaktiivisen viihteen muoto, jossa käyttäjä eli pelaaja voi vaikuttaa tietokoneella simuloituun peliin syötelaitteilla, joita ovat yleisimmin näppäimistö, hiiri tai erityisesti pelaamiseen suunniteltu peliohjain. Tulostelaitteina käytetään tavallisesti jonkinlaista näyttölaitetta, kuten tietokonemonitoria, televisiota tai projektoria, kaiuttimia pelin ääniä varten ja mahdollisesti peliohjainta, joka antaa palautetta pelistä fyysisen voiman muodossa.

Videopelit on käsite, johon kuuluvat kaikki elektronisille järjestelmille eli alustoille tehdyt pelit, jotka käyttävät videonäyttöä tulostelaitteena. Tietokonepelit ovat osa videopelejä.

Tässä tutkielmassa esitellään pääasiassa kahden kaupallisen tietokonepelin epätavanomaisia tekoälytoteutuksia. Tämän lisäksi esitellään lyhyesti, miten tietokonepeleissä on tavanomaisesti toteutettu tekoäly.

Avainsanat: tekoäly, tietokonepelit, videopelit, tekoäly tietokonepeleissä

# Sisällys

1 Johdanto.....	1
2 Tekoäly.....	3
2.1 Tekoälyn määritelmä.....	3
2.2 Akateeminen tekoäly.....	5
2.3 Pelien pelaaminen.....	6
2.4 Filosofisia pohdintoja tekoälyn tulevaisuudesta.....	7
3 Tietokonepelit.....	9
3.1 PC-laitteistojen kehittämisestä.....	9
3.2 Tutkielmassa käsiteltävistä peleistä lyhyesti.....	11
3.2.1 Thief: The Dark Project.....	11
3.2.2 Black and White.....	12
4 Tekoäly tietokonepeleissä.....	14
4.1 Tietokonepelitekoälyn määritelmä.....	14
4.2 Tekoälyn nykytila tietokonepeleissä.....	15
4.3 Thief: The Dark Project.....	16
4.3.1 Suunnittelu.....	16
4.3.2 Aistimusjärjestelmistä.....	17
4.3.3 Toteutus.....	19
4.3.4 Johtopäätökset.....	22
4.4 Black and White.....	23
4.4.1 Suunnittelu.....	23
4.4.2 Agenttiarkkitehtuuri.....	23
4.4.3 Mielipiteiden oppiminen.....	26
4.4.4 Empaattisuus.....	27
4.4.5 Persoonallisuusvaatimus.....	28
4.4.6 Tulevaisuuden suuntauksia.....	29
5 Johtopäätökset.....	30
LÄHTEET.....	31
Kirjallisuus.....	31

Pelit.....	33
Elokuvat.....	34

# 1 Johdanto

Ensimmäinen kaupallinen videopeli julkaistiin vuonna 1971 [Bushnell 1996] eli 37 vuotta sitten, ja nykyään videopeliteollisuus on jo jopa elokuvateollisuuteen verrattuna isoa liiketoimintaa (ks. luku 3). Kustannusnäkökulmasta ajateltuna on selvää, että videopelin tuotantoyritys (videopeliteollisuudessa käytetään yleensä termiä julkaisija, engl. publisher) ottaa melkoisen taloudellisen riskin ryhtyessään tuottamaan videopeliä, jonka kehittäminen parhaimmillaan kestää usean vuoden ajan jopa yli sadan ihmisen työskennellessä sen parissa, ja on luonnollista, että tämä taloudellinen riski halutaan minimoida (ks. luku 3). Yksi keino lieventää riskejä on, että pelistä tehdään viihdyttävä jollakin yleisellä tavalla, jolloin se mahdollisesti vetoaa kohderyhmään paremmin kuin ei-viihdyttävä peli.

Vaikka absoluuttisia mittareita videopelin viihdyttävyyden mittaamiseen ei tietenkään ole olemassa, niin joitain suuntaa antavia menetelmiä on kyllä tutkittu. Mm. Yannakakis ja Hallam [2007] esittävät, että mikäli pelaajan vastustajana olevan tekoälyn käyttäytyminen pelissä on älykästä eli jos se esimerkiksi oppii pelin aikana uutta tai on jollain tavalla arvaamaton, niin tämä vaikuttaa positiivisesti pelaajan subjektiiviseen näkemykseen pelin viihdyttävyydestä.

Tietokonepelitekoälyn kehitykseen käytettyjen resurssien eli prosessointiajan ja pelkästään tekoälyn keskittyneiden kehittäjien määrä on kasvanut pelitaloissa (pelejä tekevissä yrityksissä) vuodesta 1997 asti merkittävästi. Jo vuoden 2000 Game Developer Conferencen jälkeen oli selvää, että tekoälyn tärkeys pelin suunnittelussa ja toteutuksessa oli välittynyt niin pelin varsinaisille kehittäjille kuin myös projektipäälliköille ja tuottajille. Tekoälyn ei enää ajatella olevan pelin kannalta vähäpätöinen ominaisuus ja välttämätön riesa, joka on peliprojektin aikataulussa jätetty yleensä viimeiseksi ja jonka toteutus on annettu pelitalon osa-aikaisen harjoittelijan työtehtäväksi. Sen sijaan se mielletään yhdeksi pelin tärkeäksi ominaisuudeksi, joka vaikuttaa pelaajan pelikokemukseen samoin kuin pelin graafinen ulkoasukin. [Woodcock 2000.]

Tutkielman luvussa 2 esitellään tekoälyn peruskäsitteitä sekä -kysymyksiä ja luvussa 3 käsitellään videopelejä ja videopeliteollisuutta.

Luvussa 4 syvennyttään tutkielman varsinaiseen aiheeseen esittelemällä lyhyesti tietokonepelitekoälyn nykytilaa ja sen lisäksi tutkitaan kahden kaupallisen tietokonepelin epätavanomaisia tekoälytoteutuksia.

Luvussa 5 esitellään lyhyesti tutkielman johtopäätökset.

## 2 Tekoäly

Akateemisena tieteenalana tekoäly syntyi vuonna 1956 kesällä Dartmouth Collegessa ([www.dartmouth.edu](http://www.dartmouth.edu)) Yhdysvalloissa pääasiassa John McCarthyn halukkuudesta tutkia automaattiteoriaa, neuroverkkoja ja yleisesti älykkyyttä muiden aiheesta kiinnostuneiden tutkijoiden kanssa. Tuolloin myös päätettiin tieteenalan nimi, vaikkakin näin jälkikäteen voisi sanoa, että ”laskennallinen rationaalisuus” olisi ollut ehkä parempi valinta tekoälyn sijaan. [Russell ja Norvig 2003.]

Tekoälyssä tutkitaan ihmisälykkyyttä ja rationaalisuutta (ideaali käsite älykkyydelle). Tavoitteena on toteuttaa autonominen toimija (engl. agent), joka kykenee ratkomaan sille asetettuja ongelmia itsenäisesti. Esimerkiksi tästä sopii pölynimurirobotti, joka kykenee pitämään huoneen puhtaana imuroimalla sitä sopivin väliajoin. [Russell ja Norvig 2003.]

### 2.1 Tekoälyn määritelmä

Jotta erilaisia tekoälyn määritelmiä olisi helppo verrata toisiinsa, ne voidaan jakaa nelikenttään, jossa jaottelu tehdään ihmismäisyyden ja rationaalisuuden sekä toiminnan ja ajattelun välillä. Rationaalisuus ja ihmismäisyys erotellaan toisistaan, koska ihmiset eivät tiettävästi aina toimi järkevästi, vaikka siihen olisikin mahdollisuus. Näin saadaan neljä erilaista lähestymistapaa tekoälyn määritelmälle: ihmismäinen toiminta, ihmismäinen ajattelu, rationaalinen ajattelu ja rationaalinen toiminta. [Russell ja Norvig 2003.]

Ihmismäisen toiminnan kautta tarkasteltuna tekoälyä voidaan havainnollistaa Turingin testin [Turing 1950] avulla, missä tekoälyn älykkyyttä mitataan vertaamalla sen toimintaa eittämättä älykkään olion eli ihmisen toimintaan. Turingin testissä arvosteltavana oleva tekoäly läpäisee testin, mikäli sen toimintaa ei ole mahdollista erottaa ihmisen toiminnasta. On kuitenkin ehkä kyseenalaista väittää, että älykkään olion toiminnan monistaminen takaisi tämän olion olevan myös todellisuudessakin

älykäs. Ilmailuteollisuuden tarkoituksena ei esimerkiksi ole valmistaa lentokoneita, jotka lentävät yhtä ”pulumaisesti” kuin kyyhkys ja jotka pystyvät puijaamaan tässä asiassa itse kyyhkysiäkin. [Russell ja Norvig 2003.]

Ihmismäinen ajattelu -lähestymistapa tekoälyyn luonnollisesti vaatii ymmärrystä siitä miten ihmiset käytännössä ajattelevat eli joko pyrimme tutkimaan omaa ajatteluamme itse (kvalitatiivinen tutkimus) tai järjestämme joukon psykologisia testejä ja teemme päättelyitä niiden pohjalta (kvantitatiivinen tutkimus). Arvioitava kohde on siis älykäs, jos tämä kykenee ihmismäiseen ajatteluun. [Russell ja Norvig 2003.]

Rationaalinen ajattelu on käytännössä loogista päättelyä. Tekoälyn määritelmien suhteen sillä tarkoitetaan, että mikäli arvioitava kohde pystyy loogiseen päättelyyn, niin se on älykäs. Tämä suuntaus on hankala kahdesta pääsystä. Ensiksikin epätasällisen tiedon (esim. fyysisen todellisuuden tilan) esittäminen täsmällisellä loogisella notaatiolla ei ole helppoa, ja toiseksi ongelman periaatteellisen ja käytännöllisen ratkaisun välillä on suuri ero, koska looginen päättely voi kestää hyvinkin pitkään, jolloin optimaalinen ratkaisu ongelmaan ei ole enää ehkä mahdollinen. [Russell ja Norvig 2003.]

Viimeisenä nelikentässä on rationaalinen toiminta. Tämän mukaan arvioitava kohde on älykäs, jos se kykenee toimimaan saavuttaakseen parhaan lopputuloksen tai parhaan mahdollisen lopputuloksen tietyn ongelman suhteen. Russell ja Norvig keskittyvät kirjassaan nimenomaan rationaalisen toimijan peruseriaatteisiin (engl. principles) ja komponentteihin, joilla näitä toimijoita voidaan rakentaa. Syytä valintaansa he perustelevat sillä, että rationaalinen ajattelu (looginen päättely) on vain yksi tavoista saavuttaa rationaalisuus ja että rationaalinen toiminta soveltuu paremmin tieteelliselle kehitykselle kuin ihmisyyteen keskittyvät käyttäytymistavat ja se mahdollistaa lisäksi yleisemmän ja selvemman älykkyyden määritelmän. [Russell ja Norvig 2003.]

Ian Millingtonin mukaan tekoäly on sitä, että tietokone saadaan suorittamaan ihmisille ja eläimille ominaisia ajattelutoimintoja [Millington 2006]. Tämä määritelmä sijoittuu Russellin ja Norvigin nelikenttämallissa kohtaan ihmismäinen ajattelu, joten se ei ole kovinkaan täsmällinen, sillä ihmismäinen käyttäytyminen, josta ajattelua voidaan pitää yhtenä ilmentymänä, on sopeutunut hyvin vain yhteen ympäristöön ja on toistaiseksi

tuntemattoman evolutiivisen prosessin tulosta [Russell ja Norvig 2003]. Siitä huolimatta tätä voidaan pitää varsin pätevänä määritelmänä tietokonepelitekoälyn kannalta, koska peleissä tekoälyn tarkoituksena on yleensä olla ihmismäinen (tai eläimellinen), eikä niinkään erehtymättömän eksakti.

Toisaalta tietokonepeleissä agenteilta vaaditaan myös ihmismäistä toiminnallisuutta eikä pelkästään ajattelukykyä, jolloin Millingtonin määritelmä ei ole kuitenkaan riittävä. Tässä tutkielmassa tekoälyä tarkastellaan siitä syystä ihmismäisenä toimintana, jota pelimaailman agentit pyrkivät matkimaan.

## **2.2 Akateeminen tekoäly**

Akateemisesti tekoälyä tutkivista henkilöistä osa on kiinnostunut tekoälyn filosofisesta näkökulmasta, jossa pyritään ymmärtämään ajatuksen ja älykkyyden perimmäistä luonnetta sekä rakentamaan ohjelmia, joilla mallinnetaan, miten ajattelu voisi toimia. Osa tutkijoista on kiinnostunut psykologisesta näkökulmasta eli ihmisaivojen ja ajatteluprosessien mekaniikan ymmärtämisestä. Toiset taas ovat kiinnostuneita tekoälystä insinööritaitona, jossa rakennetaan algoritmeja, jotka kykenevät suorittamaan ihmismäisiä töitä. Tekoälyn tarkastelu insinööritaitona on tietokonepelien kannalta oleellisin, koska peleissä ollaan pääasiassa kiinnostuneita käytännöllisten algoritmien rakentamisesta. [Millington 2006.]

Ian Millington jakaa akateemisen tekoälyn karkeasti kolmeen, osittain päällekkäiseen, aikakauteen: varhaiseen, symboliseen ja luonnolliseen aikakauteen. [Millington 2006]

Varhainen aikakausi käsittää myös ajan ennen tietokoneita, jolloin esimerkiksi mielenfilosofiassa esitettiin kysymyksiä, kuten ”Mikä saa aikaan ajatuksen?” ja ”Voiko ei-elävän olion elvyttää?”. Varhaisen aikakauden loppupuolella Turing julkaisee kuuluisan filosofisen artikkelinsa [Turing 1950]. [Millington 2006.]

Symbolisen aikakauden aikana, joka ajoittuu 1950-luvun lopusta 1980-luvun alkuun, tekoälytutkimuksessa keskityttiin pääasiassa symbolisiin järjestelmiin. Symbolisessa järjestelmässä algoritmi on jaettu kahdeksi eri komponentiksi, tietämysjoukoksi, joka koostuu symboleista, kuten sanoista, numeroista ja lauseista, ja sitä manipuloivaksi

päättyä algoritmi, joka tuottaa uusia symbolien yhdistelmiä, jotka esittävät ratkaisuja ongelmiin tai mahdollisesti uutta tietoa. Asiantuntijajärjestelmät ovat yksi esimerkki symbolisesta järjestelmästä. [Millington 2006.]

Symbolisia järjestelmiä toteutettaessa on tyypillistä tehdä kompromissi tietämisyökon koon ja päättyä algoritmin suoritusajan välillä. Mitä enemmän tietoa ongelmasta on, sitä vähemmän on tarvetta hakea ratkaisua päättyä algoritmin avulla. Sama pätee myös päinvastaisena, eli mitä enemmän aikaa voidaan käyttää päättyä algoritmin suorittamiseen, niin sitä vähemmän tietoa tarvitaan ongelman ratkaisun löytämiseen. Tätä Millington kutsuu tekoälyn kultaiseksi säännöksi. [Millington 2006.]

Luonnollinen aikakausi alkaa 1980-luvun puolessa välissä, jolloin kiinnostus luonnollisen laskennan (engl. natural computing) tekniikoita kohtaan alkoi kasvaa. Näitä tekniikoita ovat mm. neuroverkot, geneettiset algoritmit ja jäädytyksen simulointi (engl. simulated annealing). Luonnollisen laskennan tekniikoita ei kuitenkaan keksitty 1980-luvulla vaan jopa ennen symbolista aikakautta. Neuroverkot esimerkiksi esitettiin ensimmäisen kerran jo vuonna 1943. [Millington 2006.]

Kiinnostus neuroverkkoja kohtaan kasvoi 1980-luvun puoli välissä, koska ainakin neljä eri tutkijaryhmää keksi uudelleen back-propagation oppimisalgoritmin, jonka Bryson ja Ho löysivät jo vuonna 1969, ja sitä hyödynnettiin tuloksekkaasti moniin oppimisongelmiin tietojenkäsittelytieteessä ja psykologiassa. Näistä älykkäistä järjestelmistä alettiin tuolloin käyttää nimitystä konnektionistimallit (engl. connectionist models). [Russell ja Norvig 2003.]

### **2.3 Pelien pelaaminen**

Nykyään on mahdollista ohjelmoida tietokone ratkaisemaan aritmeettisia, lajittelu- ja hakuongelmia yli-inhimillisellä nopeudella. Tämän lisäksi on olemassa tietokoneohjelmia, jotka esimerkiksi kykenevät pelaamaan joitakin lautapelejä paremmin kuin kukaan ihminen. [Millington 2006.]

29. huhtikuuta 2007 julistettiin tammipeli ratkaistuksi Albertan yliopiston tietojenkäsittelytieteiden laitoksella Kanadassa. Chinook-niminen tietokoneohjelma

kykenee pelaamaan siten, että alkutilanteesta lähtien mustalle pelaajalle, joka siirtää ensin, on taattu virheetön tasapeli ja valkoiselle, joka siirtää toisena, on myös taattu tasapeli riippumatta siitä, millaisen siirron musta pelaaja tekee aloituksessa. Tammipelissä on noin 500 miljardia miljardia ( $5 \times 10^{20}$ ) erilaista pelitilannetta, ja tällöin sen ratkaiseminen eli lopputilanteen päättely kummankin pelaajan osalta virheettöä on ihmisille, jotka yleisesti pystyvät pitämään mielessä enintään 5–9 asiaa kerralla [Miller 1956], hyvin vaikea ongelma. Aiemmin tekoälyteknologiaa on hyödynnetty mm. šakin pelaamisessa [Schaeffer ja Plaatt 1997], jolloin sitä käytettiin tehokkaan heuristiikan generointiin, millä laskettiin tietyssä pelitilanteessa paras mahdollinen siirto. Pelin ratkaiseminen on astetta vaativampi operaatio, koska se korvaa heuristiikan täydellisyydellä. [Schaeffer et al. 2007.]

## **2.4 Filosofisia pohdintoja tekoälyn tulevaisuudesta**

On mielenkiintoista nähdä, saadaanko joskus järkevästi päättävä tai ajatteleva yleinen ihmismäinen tekoäly aikaan. Voi tietenkin olla mahdollista, että se on loppujen lopuksi turha yritys. Ihmistietämys on nimittäin harvoin eksaktia ja usein myös epätarkkaa ja puutteellista [Knuutila 2006], mikä tekee sen mallintamisen hankalaksi jo pelkästään määritelmällisesti. Inhimillisyyden käsitteeseen kuuluu itsessään tietynlainen luonnollinen erehtyväisyys, ja sen mallintaminen vaatisi ensinnäkin meidän oman ajattelumme täydellistä ymmärtämistä, mitä meillä ei yksinkertaisesti vielä ole. Erehtymätön älykkyys olisikin luultavasti helpompi tutkimusaihe, mutta millä tavalla tällaista ajatusta tulisi sitten lähestyä? Voiko erehtyväinen ihminen todellakin luoda jotakin, mikä on kykenemätön virheeseen? Nämä kysymykset lähenevät jo filosofisia ja psykologisia pohdiskeluita.

Toinen lähestymistapa aiheeseen on teologinen näkökulma, jossa pohditaan muun muassa koko tutkimusalan moraalisuutta ja oikeellisuutta. Osittain tähän ovatkin perustuneet scifikirjallisuudessa ja elokuvissa esiintyvät tulevaisuuden uhkakuvat, joissa pitkälle kehitetty (engl. advanced) tekoäly on valjastettu hallinnoimaan ja ylläpitämään kaikenlaisia tietojärjestelmiä. Tämän jälkeen, syystä tai toisesta, tekoäly saa yleensä sille tarkoittamatonta valtaa, esimerkiksi rajoittamattomat käyttöoikeudet

sotilastietojärjestelmään, minkä jälkeen se alkaa tuhota, alistaa tai muuten vain yleisesti häiritä ihmiskuntaa.

### 3 Tietokonepelit

Aiemmin tänä vuonna (2008) videopeli nimeltä Grand Theft Auto IV [Rockstar Games 2008] rikkoi viihdeteollisuuden myyntiennätyksiä tuottamalla maailmanlaajuisesti 310 miljoonaa dollaria yhdessä vuorokaudessa ja 500 miljoonaa dollaria ensimmäisen viikon aikana [CNET Networks 2008]. Elokvateollisuudessa rikottiin myös ennätyksiä heinäkuussa, kun The Dark Knight [WB 2008] tuli elokuvateattereihin. Elokuva tuotti tekijöilleen 158 miljoonaa dollaria ensimmäisen vuorokauden aikana ja 300 miljoonaa dollaria 10 päivän kuluessa [IMDb 2008].

Videopeliteollisuus tuotti 41,9 miljardia dollaria maailmanlaajuisesti vuonna 2007, ja sen on ennakoitu kasvavan 10,3 % vuodessa, niin että se saavuttaa 68,3 miljardin dollarin vuotuisen tuoton maailmanlaajuisesti vuonna 2012 [PricewaterhouseCoopers 2008]. Elokvateollisuus tuotti 26,7 miljardia dollaria maailmanlaajuisesti lippukassatuloina vuonna 2007, mikä merkitsi 4,9 %:n kasvua vuodesta 2006 [MPAA 2008].

Videopeliteollisuus on nykyään siis merkittävää liiketoimintaa, ja mainittujen lukujen perusteella sen voi sanoa olevan suuri jopa elokuvateollisuuteen verrattuna. Videopeliteollisuuden kaupalliset huippupelit maksavat tuotantoyrityksille nykyään miljoonia dollareja [BBC News 2008], ja näiden kehittämisessä kuluu useita vuosia [Future Publishing 2007] jopa yli sadan ihmisen työskennellessä täysipäiväisesti niiden parissa [Rosmarin R. 2006].

#### **3.1 PC-laitteistojen kehittämisestä**

Vuodesta 1997 asti PC-laitteistoissa huomattavaa suorituskyvyn parannusta on tietokonepelikehittäjien näkökulmasta tapahtunut pääasiassa näytönohjainkortissa. Tämän takia viime vuosikymmenen ajan suurinta kehitystä on tapahtunut enimmäkseen tietokonepelien graafisessa eli kuvallisessa laadussa ja näyttävyydessä [Champandard 2004].

Tietokonepelien voidaan sanoa vaikuttaneen merkittävästi PC-tietokoneiden ja PC-laitteistojen myyntiin, koska uusimmat ja myydyimmät tietokonepelit vaativat koneelta yleensä suorituskyykyä, jota ainoastaan viimeisimmät laitteistovalmistajien tuotteet voivat tarjota. PC-laitteistot vanhenevat tietokonepelikäytössä nopeasti, uusien ja suorituskyykyisimpien tuotteiden tullessa markkinoille tavallisesti n. 6 – 24 kuukauden sykleissä.

Vuonna 2005 Advanced Micro Devices, Inc ja vuonna 2006 Intel Corporation julkaisivat kuluttajille suunnatut tuplaydinprosessorinsa eli teoriassa ohjelmien suoritukseen käytettävissä oleva yleinen suorituskyyky kaksinkertaistui saman tien. Asia ei kuitenkaan ole näin yksinkertainen, koska moniydinprosessorien rinnakkaista laskentaa tehokkaasti hyödyntävien sovelluksien suunnittelu ja toteuttaminen vaatii käytännössä kyseisen ohjelman pilkkomisen kahdeksi tai useammaksi rinnakkaiseksi suoritussäikeeksi, ja tällöin ohjelman kompleksisuus kasvaa muun muassa muistinhallinnan osalta.

Moniydinprosessorit mahdollistavat tietokonepeleissä esimerkiksi paremman fysiikan ja tekoälyn simuloinnin kuin aiemmin, koska näihin ominaisuuksiin käytettävissä olevan laskentatehon määrä on yksinkertaisesti huomattavasti suurempi kuin yksiydinprosessorissa. Tällöin pystytään käyttämään kompleksisempia algoritmeja hyväksi kuitenkin ottamatta laskentatehoa pois muilta pelille tärkeiltä prosesseilta.

Nykyiset grafiikkakortit, jotka ovat massiivisesti rinnakkaistettuja ja sisältävät jopa satoja suoritusyksiköitä [Amd 2008], tarjoavat myös mielenkiintoisia mahdollisuuksia. Nykyisellään grafiikkakortteja on käytetty pääasiallisesti graafisesti näyttävän videon tuottamiseen, mutta esimerkiksi tämän vuoden SIGGRAPH-konferenssissa (Special Interest Group in Graphics Conference) esiteltiin tutkimustuloksia, joissa väitetään tietyn kuluttajille suunnatun ATI-grafiikkakortin suorittavan yhtäaikaaisesti 65000 tekoälyagentin simulointia – tässä tapauksessa pääasiassa reitinhakua – reaaliajassa. Lisäksi tutkimuksessa väitetään samaisen grafiikkakortin päihittävän neljäytimellisen tavanomaisen AMD:n suorittimen laskentatehon tekoälyagenttien simuloinnin suhteen 45-kertaisesti. Jotta tämänlainen laskentateho saadaan aikaan, tulee tekoälylaskentaa luonnollisesti myös rinnakkaistaa. [Shopf et al. 2008.]

## 3.2 Tutkielmassa käsiteltävistä peleistä lyhyesti

Black and White [Lionhead Studios 2001] on PC-tietokonepeli, joka kaupallisena tuotteena on myynyt hyvin [Lionhead Studios 2008] ja joka perustuu peliteknisesti merkittävällä tavalla tekoälyn simulointiin. Olen valinnut kyseisen pelin tutkielman tarkastelun kohteeksi siitä syystä, että se on vaikuttanut sekä pelikehittäjien että pelaajien näkemykseen siitä, mitä peleissä on mahdollista toteuttaa ja mitä niiltä voidaan nykyään myös odottaa [Woodcock 2002, Woodcock 2003].

Lisäksi olen valinnut pelin nimeltä Thief: The Dark Project (Thief) [Looking Glass Studios 1998], jossa tekoälyllä on hyvin keskeinen, joskin tavanomaista erilaisempi, rooli pelissä [Leonard 1999].

### 3.2.1 Thief: The Dark Project

Thief on ensimmäisestä perspektiivistä eli pelihahmon silmistä kuvattu 3D-toimintaseikkailupeli, jossa pääpaino on hiiviskelyssä. Thief on ensimmäisiä pelejä, joissa hiiviskely on niin keskeisessä osassa, että käytännössä koko peli on rakennettu tämän ominaisuuden ympärille. Tästä syystä pelissä on kiinnitetty huomiota erityisesti aistien

Thief: The Dark Project Looking Glass Studios Kehittäjien lukumäärä: 19 Julkaisupäivä: Joulukuu 1998 Kohdealusta: Windows 95/98 Projektin budjetti: ~\$3M Projektin pituus: 2.5 vuotta
--

[Leonard 1999]

mallinnukseen, koska pelaajan on esimerkiksi tarkoitus kulkea huomaamattomasti alueilla, joissa tekoälyn ohjaamat vartijat partioivat, pysytellen varjoissa ja liikkuen äänettä [Leonard 2003]. Pelaaja ohjaa keskiaikaisessa fantasiamaailmassa varasta nimeltä Garrett, jonka tehtävänä on tunkeutua rakennuksiin ja ryöstää näistä esineitä. Pelaajalla on tehtävän suorittamiseen yleensä monta erilaista vaihtoehtoa, ja pelimaailman dynaamisuus mm. vapaasti liikuteltavat objektit, mahdollistaa vieläkin vapaamuotoisemman pelattavuuden.

Thief-pelin kehittäjätiimi halusi luoda täysin erilaisen ensimmäisen perspektiivin

pelikokemuksen, kuin aiemmin oli tehty kuitenkin niin, että peli vetoaisi myös tavanomaisempien vastaavanlaisten pelien pelaajiin. Ideana oli tehdä kevyesti käsikirjoitettu (engl. scripted) pelimaailma, joka antaisi pelaajalle mahdollisuuden monipuoliseen pelilliseen interaktioon ja improvisaatioon ylittäen näin Looking Glass Studios -tiimin aiemmat tuotokset. [Leonard 1999.]

Pelin keskeinen mekaniikka haastoi tavanomaiset markkinoilla olevat ensimmäisen perspektiivin 3D-toimintapelit siinä mielessä, että kun tavallisesti nämä pelit ovat nopeatempoisia ammuskelupelejä, joissa pelihahmolla on yli-inhimillinen nopeus ja kestävyys ja pelissä itsessään keskitytään pääasiassa konflikteihin, niin nyt Thief-pelissä pelihahmo liikkuukin hitaasti, väsyä ja on kuolevainen. Konfliktien välttäminen on täten hyödyllistä, ja pelimaailman ihmisten eliminointia tulee välttää, sillä se vähentää pisteitä itse pelissä. [Leonard 1999.]

### 3.2.2 Black and White

Black and White (B&W) on maailmasimulaatio, jossa pelaaja on jumala, jolla on oma kansansa ja lemmikkiotuksensa. Pelin ajatuksena on antaa pelaajalle valta toimia pelimaailmassa hyvin vapaasti ja siten nähdä, minkälainen jumala pelaaja olisi oikeasti. Pelin pääsuunnittelija Peter Molyneux kertoo tavoitteena olleen luoda

Black & White
Lionhead Studios
Kehittäjien lukumäärä: 25
Julkaisupäivä: Maaliskuun 30. 2001
Kohdealusta: Windows 95/98/2000/ME
Projektin budjetti: ~£4M (~\$5.7M)
Projektin pituus: 3v 1kk 10pv
Projektin koko: ~2M koodiriviä

[Molyneux 2001]

pelin, joka on avoimempi, joustavampi ja vetovoimaisempi kuin mikään aiempi peli [Molyneux 2001]. Molyneux halusi, että pelaajan olisi nimenomaan mahdollista tehdä pelimaailmassa melkein mitä tahansa ja miten tahansa.

Pelaajan toiminnot vaikuttavat pelimaailmaan sekä suorasti, että epäsuorasti. Jos pelaaja esimerkiksi haluaa nostaa maasta kivenjätkäleen ja paiskata sen oman kansansa kylässä olevaa taloa päin niin suora seuraus tästä on, että talo hajoaa, ja epäsuora, että pelaajan

kansa mahdollisesti alkaa palvoa tätä pelosta. Mikäli pelaaja haluaa pelata pahana jumalana, niin on mahdollista, että pelimaailman maasto alkaa vähitellen muuttua kuvaamaan tätä esimerkiksi kuolleen kasvillisuuden tai synkän sään muodossa.

Yhtenä merkittävimmistä tavoitteista oli lisäksi mallintaa pelaajalle lemmikkiotus, joka kykenisi pelin aikana oppimaan ja toimimaan pelaajan apurina. Tutkielmassa keskitytään tämän pelin kohdalla siihen, miten lemmikkiotuksen oppimiskyky toteutettiin (ks. luku 4.5).

## 4 Tekoäly tietokonepeleissä

Tietokonepelitekoälyksi saatetaan laskea monia erilaisia tietokonepeleissä käytettyjä tekniikoita, joiden voidaan mieltää toteuttavan pelissä jonkinasteista älykkyyttä. Joidenkin pelikehittäjien mielestä esimerkiksi agentin tekemä optimaalisen kulkureitin laskeminen peliympäristössä tai jopa kappaleiden törmäystarkastelu kuuluvat tietokonepelitekoälyyn [Woodcock 2003].

### 4.1 Tietokonepelitekoälyn määritelmä

Brian Schwab määrittelee tietokonepelitekoälyn sellaiseksi, että se saa tietokoneohjatut vastapelaajat tai kanssapelaajat näyttämään pelissä siltä, että nämä tekevät viisaita päätöksiä silloin, kun pelissä on useampi kuin yksi tapa toimia, mistä seuraa käyttäytymisiä, jotka ovat relevantteja, tehokkaita ja hyödyllisiä. Schwab tekee lisäksi selvän erottelun tietokonepelitekoälyn ja akateemisen tekoälyn välille. Schwabin mukaan akateemisessa tekoälyssä pyritään yleensä tiettyssä tilanteessa tekemään parhaita tai eniten oikeassa olevia rationaalisia päätöksiä, kun taas tietokonepelitekoälyssä keskitytään ihmismäiseen toimintaan eikä niinkään totaaliseen rationaalisuuteen. Syynä tähän on tarkoituksien ero. Pelit ovat pääasiassa viihdettä varten, kun taas akateeminen tekoäly on tutkimusta, jossa kehitetään tiedettä. [Schwab 2004.]

Bourg ja Seeman taas määrittelevät pelitekoälyn vielä laajemmin kuin Schwab. Heidän mukaansa kaikki, mikä luo riittävää älykkyyden illuusiota pelissä, tehden siitä immerstiivisen, haastavan ja erityisesti hauskan, on tietokonepelitekoälyä. [Bourg ja Seeman 2004.]

Myös tässä tutkielmassa keskitytään tietokonepelitekoälyn laajaan käsitteeseen. Kaikki yksinkertaisista reitinhakumenetelmistä monimutkaisempiin oppiviin neuroverkkotekniikoihin sisällytetään termiin tietokonepelitekoäly.

## **4.2 Tekoälyn nykytila tietokonepeleissä**

Tietokonepelitekoäly on yleisesti vielä melko varhaisessa vaiheessa. Käytetyimpinä tekniikoina peliteollisuudessa ovat pääasiassa yhä vanhat ja toimiviksi havaitut menetelmät, kuten käsikirjoitetut (engl. scripted) agenttien käyttäytymiset ja A\*-algoritilla toteutettu reitinhaku. Tätä tilannetta voitaisiin verrata pelien grafiikkapuolella siihen, että nykyiset pelit käyttäisivät edelleen 2D-grafiikkaa 3D-grafiikan sijaan. Uudempaa teknologiaa ei ole suuremmassa mitassa vielä käytössä. [Champanard 2004.]

Malliesimerkkinä nykyaikaisesta, tyypillisestä ja hyvin toteutetusta ”tekoälystä” on Call of Duty 4 (Infinity Ward 2007). Pelilogiikan matalalla tasolla kyseisessä pelissä on ainoastaan hyödynnetty varsinaisia tekoälytekniikoita agenttien liikkeiden animoinnissa ja reitinhaussa. Kaikki muu eli käytännössä koko lopullinen pelaajalle näkyvä tekoälytoteutus nojaa vahvasti käsikirjoittamiseen, jossa tekoälyagentti ohjelmoidaan hyvin spesifisesti ja aina tiettyä pientä ongelmaa varten. Pelin tuotantovaiheessa näitä pieniä ongelmia on yleensä tuhansia, jolloin on sanomattakin selvää, että Call of Duty 4:n tekoälyn toteutus vie paljon aikaa. Pelin tekoäly ei siis osaa luoda minkäänlaisia korkeamman asteen tavoitteita itse, vaan se pitää ohjelmoida täysin manuaalisesti ja tilannekohtaisesti. Lopputuloksena on 205 718 riviä käsikirjoitettua koodia, josta ei voida käyttää uudelleen juuri mitään. Jos pelin tuotantoyrityksellä ei ole tavoitteena luoda vankkaa pohjaa tekoälylle, jonka päälle rakentaa seuraavassa peliprojektissa, niin tämä on kyllä järkevä tapa toimia, koska riski epäonnistua on pieni. On myöskin selvää, että jos jokin tapa tunnetaan yleisesti toimivaksi, niin silloin ei ole välttämättä pakottavaa tarvetta siirtyä kohti jotain parempaa tapaa, kuten korkeamman tason tekoälytekniikoiden hyödyntämistä, varsinkin jos tämä parempi tapa tuo mukanaan tuntemattomia riskejä. [Champanard tammikuu 2008 ja Champanard heinäkuu 2008.]

Uudempiakin tekniikoita on kyllä käytössä muutamissa peleissä, mutta ne ovat vielä ennemminkin poikkeus kuin sääntö. Esimerkiksi Colin McRae Rally 2.0 [Codemasters 2001] hyödyntää neuroverkkoja ja oppimista, jolloin tekoälyn käyttäytymistä ei tarvitse ohjelmoida pelkästään käsin vaan voidaan antaa sopivat kehykset joissa tekoälyn

halutaan toimivan [Hannan 2001]. Kyseisen pelin kohdalla uudemmilla tekoälyteknikoilla haluttiin pääasiassa helpottaa pelin tuotantovaihetta eikä niinkään tehdä lopputuloksesta muulla tavalla erikoista. Uudempaa teknologiaa siis hyödynnettiin, jotta saatiin aikaan sama vanha toiminnallisuus eli tässä tapauksessa tekoälyrallikuskit, jotka kykenevät ajamaan kilpaa pelaajan kanssa. Tämä ei siis ole siinä mielessä pelillisesti mitenkään erikoinen saavutus. [Champanard 2004.]

Aiemmasta esimerkistä hieman kehittyneempänä versiona voidaan pitää sitä kun uutta tekoälyteknologiaa hyödynnetään niin, että tarkoituksena on saada agentit käyttäytymään tavanomaista tietokonepelitekoälyä monipuolisemmin ja kehittyneemmin. Tällöin tekoäly otetaan huomioon jo pelisuunnittelussa, ja silloin peli yleensä rakennetaan myös tekoälytoteutuksen ympärille eikä päinvastoin. Esimerkkejä tällaisesta ovat mm. Thief: The Dark Project ja Black and White. [Champanard 2004.]

### **4.3 Thief: The Dark Project**

Thiefissä pelaajahahmon hiiviskelyllä, eli varovaisella liikkumisella pelimaailmassa, on pelimekaanisesti keskeinen asema. Jotta tällainen ominaisuus voitiin toteuttaa, niin peliin tuli sisällyttää tekoälyagenttien aistien mallinnus. Kyseisen pelin kannalta näkö- ja kuuloaistien mallinnus oli oleellisinta. [Leonard 2003.]

#### **4.3.1 Suunnittelu**

Pelin suunnittelussa määriteltiin tekoälylle kolme päävaatimusta. Ensisijaisena vaatimuksena oli luoda aistimusjärjestelmä, joka olisi monipuolisesti säädettävissä siten, että pelissä ei olisi aina täysin selvää, ovatko tekoälyn ohjaamat hahmot keksineet pelaajan olemassaolon. Silloin, kun pelaaja ei oikeasti tiedä, onko hänen pelihahmonsä nähty vai ei, syntyy jännitettä, koska pelaaja ei oikeasti tiedä, mitä seuraavaksi saattaa tapahtua. Hiiviskelypeliin voidaan luoda hauskuutta, kun tätä turvallisuuden ja vaaran välistä harmaata aluetta venytetään ja muokataan sopivasti. [Leonard 2003.]

Toissijaisena vaatimuksena oli, että aistimusjärjestelmä olisi aktiivisempi ja pitäisi

kirjaa suuremmasta määrästä, olioita kuin on tavallista ensimmäisen perspektiivin toimintapeleissä. Pelaajan aiemmin tekemät toiminnot vaikuttavat pelimaailman agenttien toimintaan riippumatta siitä, onko pelaaja paikalla vai ei, kun agentti tekee havainnon. Esimerkiksi sammutettu kynttilä huoneessa A otetaan huomioon, vaikka pelaaja olisi sammuttanut kynttilän jo kauan sitten ja olisi aivan eri puolella pelimaailmaa huoneessa B silloin, kun tekoälyn ohjaama hahmo huomaa kynttilän tilaan tapahtuneen muutoksen. Yhdessä ensisijaisen vaatimuksen kanssa nämä vaatimukset aiheuttivat mielenkiintoisen suorituskykyongelman pelin kehittäjille. [Leonard 2003.]

Kolmantena vaatimuksena oli, että aistimusjärjestelmän syötteiden ja tulosteiden tulisi olla sekä pelaajille että pelin suunnittelijoille ymmärrettäviä ja koherentteja eli että pelaajan aiemmin näkemä aistisyöte – aistitulostepari pitäisi myöhemminkin paikkansa. Tämä ohjasi kohti ratkaisua, jossa syötteiden määrää, jonka pelaaja pystyisi havainnoimaan, rajoitettiin ja jossa nämä syötteet sitten johtaisivat diskreetteihin tulosteisiin. Tällöin tekoälyagentin toiminta olisi siis jossakin määrin ennakoitavissa ja täten ihmismäinen, mikä on pelin tekoälylle tietenkin suotavaa. [Leonard 2003.]

### **4.3.2 Aistimusjärjestelmistä**

Thiefin aistimusjärjestelmän toteutus perustuu samoihin peruskäsitteisiin, joita Half-Life-tietokonepelin [Valve Software 1998] tekoälyn toteutuksessa ollaan käytetty. Half-Lifen tekoälyn toteutuksessa tosin on keskitytty vahvasti modernin taktisen aseellisen taistelun simulointiin eikä niinkään hiiviskelyyn, mutta pelissä tarvitaan kuitenkin kohtalaista, joskin astetta yksinkertaisempaa, aistimusjärjestelmää kuin Thiefissä. Tästä syystä aistimusjärjestelmien perusteiden esittely tarkastelemalla Half-Lifen toteutusta on hyödyllistä, ennen kuin paneudutaan Thiefin vastaavaan ja monimutkaisempaan järjestelmään. [Leonard 2003.]

Yksinkertaisissa aistimusjärjestelmissä tekoälyn ohjaamat agentit katsovat ja kuuntelevat maailman tilaa tietyin aikaväleihin. Toisin kuin todellisessa maailmassa, jossa esimerkiksi ohiajavan ambulanssin äänen kuulemiselta ei voi välttyä, tämä on aktiivinen tapahtuma, eli jos agentti ei suorita kuuntelua tiettyinä ajanhetkenä, niin se ei kuule

mitään. Tämänlainen toteutus auttaa järjestelmän suorituskyvyn hallinnassa, jolloin voidaan keskittyä pelimekaniikan kannalta oleellisimpien aistimuksien havainnointiin kuormittamatta prosessoria kuitenkaan turhaan. Half-Lifen perusaistimuslogiikka on listauksen 1 mukainen. [Leonard 2003.]

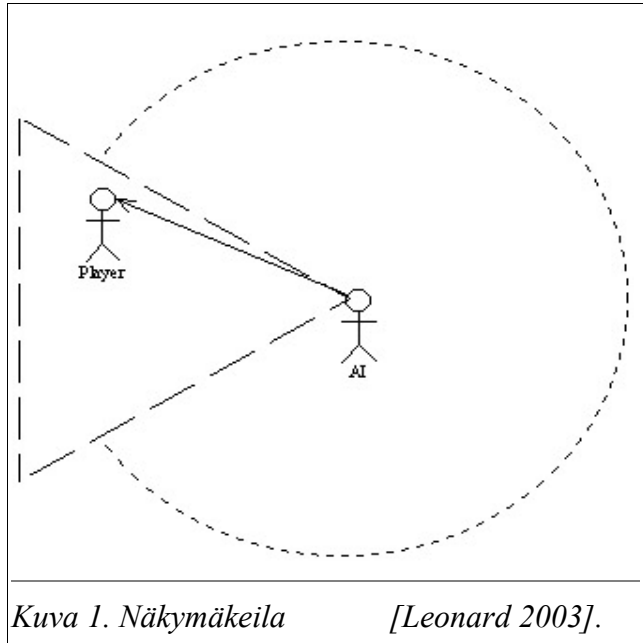
```
If I am close to player then...
  Begin look
    Gather a list of entities within a specified distance
    For each entity found...
      If I want to look for them and
      If they are in my viewcone and
      If I can raycast from my eyes to their eyes then...
        If they are the player and
        If I have been told to not see the player until they see me and
        If they do not see me
          End look
        Else
          Set various signals depending on my relationship with the seen entity
    End look
  Begin listen
    For each sound being played...
      If the sound is carrying to my ears...
        Add the sound to a list of heard sounds
        If the sound is a real sound...
          Set a signal indicating heard something
        If the sound is a "smell" pseudo-sound
          Set a signal indicating smelled something
    End listen
```

*Listaus 1. Half-Lifen perusaistimuslogiikka*

*[Leonard 2003].*

Listauksen 1 pseudokoodista on ensinnäkin nähtävissä, että mikäli tekoälyagentti ei ole pelaajaa lähellä, niin se ei ole pelin kannalta oleellinen (ensimmäinen rivi pseudokoodissa), jolloin sen ei tarvitse aistia maailman tilaa. Tämä on tehty pääasiassa pelin viihdyttävyyden vuoksi, ja se sopii hyvin esimerkiksi siitä, miten korkeamman tason pelisuunnittelun päämääriä voidaan toteuttaa yksinkertaisesti matalan tason tekniikoilla. Pelaajan kannalta ei ole nimittäin juuri oleellista se, mitä pelimaailman toisella puolella ja näkymättömissä olevat tekoälyagentit tekevät. Koodista on myös nähtävissä, että aistimukset on läheisesti kiinnitetty tekoälyn ominaisuuksiin (esim. agentin sijaintiin pelimaailmassa ja halukkuuteen etsiä pelaajaa) ja sen suhteeseen havaittavana olevaan kohteen kanssa. Näkökykyä mallinnetaan etäisyydellä kohteeseen, näkymäkeilalla (ks. kuva 1), näköyhteydellä (engl. line-of-sight) ja silmän sijainnilla. Jokaisella tekoälyagentilla on etäisyysrajoitettu kaksiulotteinen näköalue (engl. field of view), jonka sisäpuolella oleviin kiinnostaviin objekteihin agentti pyrkii suuntaamaan säteitä (engl. cast rays) silmästä silmään menetelmällä näin varmistaakseen, että näköyhteys on olemassa ja että pelaaja voi tällöin havaita agentin. [Leonard 2003.]

Huomionarvoisina asioina voidaan pitää, että aistimusoperaatiot (etäisyys, näkymäkeila jne.) on järjestetty halvimmasta kalliimpaan (etäisyys < onko pelaaja agentin näkymäkeilassa < suoran näköyhteyden tarkastelu) ja että näköyhteyden laskeminen pelaajaan tehdään silmästä silmäänmenetelmällä. Pelaajan on vaikea arvioida pelihahmonsa kehon ja raajojen ulottuvuuksia, ja tästä syystä



Kuva 1. Näkymäkeila [Leonard 2003].

tekoälyagentin näkökykyä rajoitetaan tällä tavalla. Tekoälyagentti ei siis näe pelimaailmassa pelaajaa, mikäli hänestä näkyy vain esimerkiksi jalat tai osa kättä. Tämä parantaa yleistä pelattavuutta, sillä pelaaja saattaisi nimittäin tuntea itsensä huijatuksi, jos hän ei esimerkiksi näe vartijaa, joka tekee pelissä hälytyksen. [Leonard 2003.]

Kuuloaistimuksen logiikka on huomattavasti yksinkertaisempi kuin näön. Jos pelimaailmassa esiintyvän äänen voimakkuus kerrottuna agentin ”kuuloherkkyydellä” on enintään yhtä paljon kuin agentin etäisyys äänen tuottaneeseen olioon, niin agentti havainnoi sen. Mielenkiintoisena triviaaliteettina Half-Lifessa hajuaistimus on toteutettu ääniaistimuksen avulla. Hajuaistimus on toteutettu pelimaailmassa siis samalla kaavalla kuin ääni kuuluu. [Leonard 2003.]

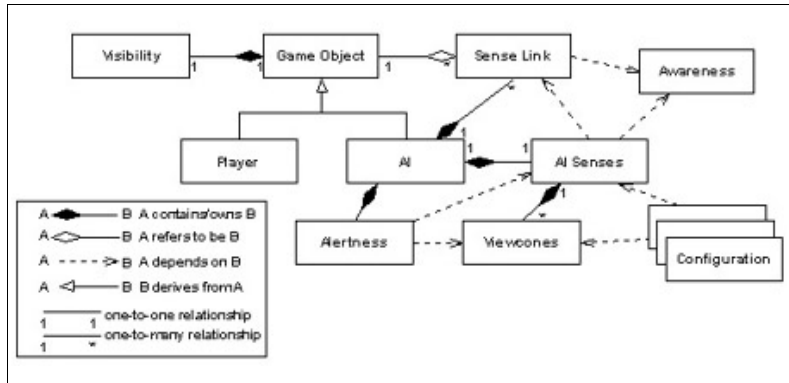
### 4.3.3 Toteutus

Kuten sanottu, Thieffissä toteutettu aistimusjärjestelmä on hyvin samankaltainen kuin Half-Lifessa. Thieffissä on näkymäkeila ja säteensuuntaukseen (engl. raycast) perustuva näköjärjestelmä (näköyhteyttä tarkastellaan silmästä silmään menetelmällä) ja yksinkertainen kuulojärjestelmä sekä joukko erilaisia ohjelmallisia takaisinkutsukoukkuja (engl. callback hook) mm. optimointeja varten. Kuten Half-Lifessa on suurin osa aistihavaintojen keräämisestä Thieffissä eriytetty

päätöksentekoprosessista. [Leonard 2003.]

Thiefin aistimusjärjestelmässä tekoälyagenttien aistihavainnot kapseloidaan termiin ”tietoisuus” (engl. awareness) (ks. kuva 2) diskreetteinä tiloina niin, että ne ovat pelin sisällön suunnittelijoille ja tuottajille ainoa näkyvä osa aistimusjärjestelmästä.

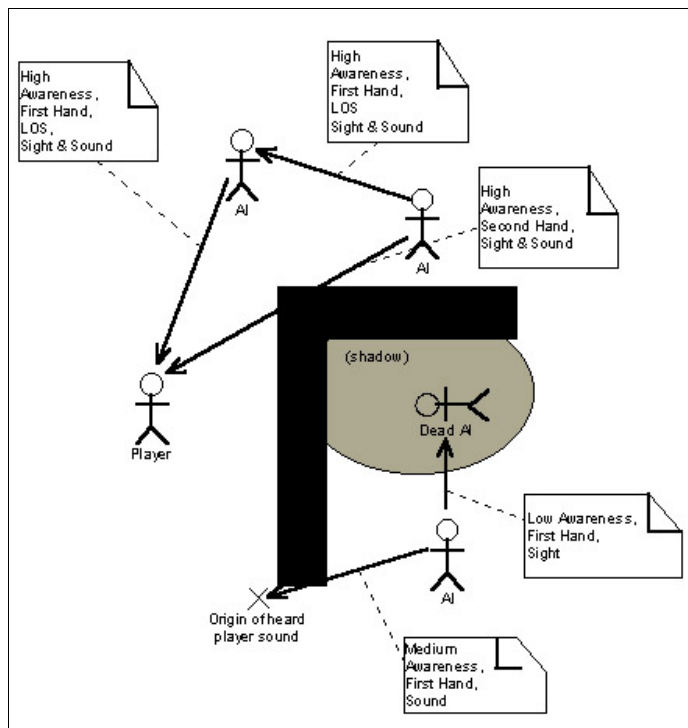
Tietoisuutta kuvataan siis diskreeteillä tiloilla, jotka esittävät sitä, miten varma tekoälyagentti on tarkasteltavana olevan kohteen läsnäolosta, sijainnista ja identiteetistä. Valppaus (engl. alertness) on



Kuva 2. Thiefin aistimusjärjestelmä [Leonard 2003].

tietoisuuden kanssa korreloiva korkeamman tason tekoälykäsite Thiefissä, ja sitä käytetään monella eri tavalla syötteenä aistimusjärjestelmään itseensä. Valppauden tarkoituksena on muuttaa järjestelmän käyttäytymistä. Esimerkiksi jos uutta aistihavaintoa ei saada tietyn ajan kuluessa niin tekoälyn valppaustasoa lasketaan. [Leonard 2003.]

Tietoisuus tallennetaan aistilinkkeihin (engl. sense links), jotka yhdistävät tietyn agentin joko toiseen peliohjektiin tai sijaintiin pelimaailman avaruudessa. Näistä saadut aistilinkkirelaatiot sisältävät asiaankuuluvia tietoja aistimuksesta, kuten esimerkiksi ajan, paikan ja näköyhteyden,

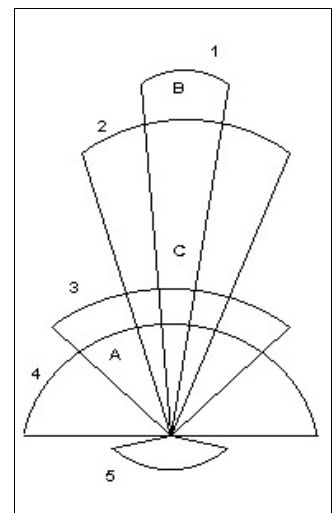


Kuva 3. Aistilinkkien siirtäminen [Leonard 2003].

kuin myös cache-arvoja, joissa säilytetään aiemmin laskettua tietoa, jota käytetään pelissä luonnollisesti tehokkuussyistä. Aistilinkit toimivat käytännössä tekoälyagenttien muistina. Puheen ja havainnoinnin avulla aistilinkkejä voidaan siirtää agentilta toiselle. Esimerkiksi kuvan 3 toiseksi ylin agentti (kuvassa AI) ei näe pelaajaa (kuvassa Player), joka on seinän takana, mutta saa sekundäärisenä tietona aistilinkin toiselta agentilta, joka on havainnut pelaajan. [Leonard 2003.]

Jokaisella tarkasteltavalla peliobjektilla on ominainen näkyvyysarvo riippumatta siitä, kuka objektin havaitsee. Pelitilanteesta ja objektin luonteesta riippuen näkyvyysarvon tarkkuutta ja päivitystiheyttä rajoitetaan, jottei operaatio kuluttaisi liikaa prosessointiaikaa. Näkyvyysarvo määrittellään objektin valaistuksen, liikkeen ja erottuvuuden avulla eli sen perusteella miten suuri kyseinen objekti on ja miten paljon se eroaa muista lähistöllä olevista objekteista. Näiden merkitys on kytköksissä pelillisiin vaatimuksiin. Pelihahmon valaistus esimerkiksi määrittyy sen alla olevan lattian valaistuksen mukaan, jolloin pelaaja kykenee arvioimaan pelimaailman eri paikkojen turvallisuutta. Mitä pimeämpi paikka, sitä turvallisempi se on pelaajan kannalta. Nämä arvot ja niiden kokonaissumma eli näkyvyysarvo tallennetaan jatkuvalla arvovälillä 0..1. [Leonard 2003.]

Thiefissä agentit havainnoivat pelimaailmassa aistimuksia kolmiulotteisilla suunnatuilla näkymäkeiloilla (ks. kuva 4) siten, että havainnointia tapahtuu ainoastaan silloin, kun havaittavissa oleva kohde on jonkin näkymäkeilan alueen sisäpuolella tai rajalla. Näkymäkeiloja on jokaisella agentilla viisi kappaletta. Näkymäkeiloille on määritetty omat herkkyysmuuttujansa, joilla voidaan määrittellä tarkemmin, miten hyvin aistimus tulee havaituksi tietyissä näkymäkeilassa. Näkymäkeilat ovat suunnattu tekoälyagentin pään asennon mukaan ja niitä on useita siitä syystä, että pelissä on haluttu mallintaa suoraa näkyvyyttä ja ääreisnäkyvyyttä kohteeseen eli sitä, onko kohde suoraan edessä vai enemmänkin näkökentän sivulla. Tämän lisäksi näkymäkeiloilla on haluttu mallintaa sitä, onko kohde edessä



Kuva 4. Näkymäkeilat  
[Leonard 2003].

ja ylhäällä vai alhaalla vai samalla tasolla. Yksi näkymäkeila on suunnattu mielenkiintoisesti agentin taakse. Tässä tarkoituksena oli saada aikaan tekoälyagenteille eräänlainen kuudes aisti, jolloin agentin takana hiippailevaa pelaajaa olisi ehkä mahdollista aistia jollain asteella ja antaa tästä palautteena pelaajalle esim. pieni jännitystä nostava kommentti kuten ”Hey is there somebody behind me?” tai ”Did I just hear something...?”. [Leonard 2003.]

Kun kohde havaitaan, otetaan huomioon ainoastaan ensimmäinen näkymäkeila, jonka alueen sisäpuolella kohde on. Esimerkiksi kuvassa 4 pisteessä A oleva kohde havaitaan vain näkymäkeilassa 3 ja pisteessä B tai C oleva kohde vain näkymäkeilassa 1. Kun havainto on tapahtunut, niin saatu ominainen näkyvyysarvo ja näkymäkeila välitetään katsomisheuristiikalle (engl. ”look” heuristic) parametrina, jolloin saadaan tulosteena diskreetti tietoisuusarvo eli tarkka arvo sille, miten tietoinen tietty agentti on jostain kohteesta. [Leonard 2003.]

#### **4.3.4 Johtopäätökset**

Koska Thiefin aistimusjärjestelmä on tehty ohjelmistopohjaiselle 3D-grafiikkamoottorille niin se edellyttää suoraa yhteyttä peliobjektien graafisiin ominaisuuksiin. Asiakas-palvelin-tyylisessä pelimoottorissa, jossa käytetään laitteistokiihdytettyä 3D-grafiikkamoottoria, kuten nykyään on hyvin tavanomaista, tämä ei välttämättä ole totta, Niinpä tämänlaisen aistijärjestelmän toteuttaminen on mahdollisesti paljon kompleksisempaa tai jopa mahdotonta. [Leonard 2003.]

Koska Thieffissä aistimusjärjestelmä on lisäksi hyvin keskeisessä osassa, niin se kuluttaa ei-triviaalin määrän pelin tekoälyä varten varatusta laskentatehosta. Tällöin tekoälyn muille osille, kuten esimerkiksi reitinhauille ja päätöksenteolle, jää normaalia vähemmän laskentatehoa. Siitä huolimatta aistien mallintaminen pelissä on Leonardin mukaan hyödyllistä, sillä se mahdollistaa monipuolisen ja tavanomaisesta poikkeavan toiminnallisuuden tekoälyagenteille, kuitenkin juurikaan monimutkaistamatta agenttien päätöksen tekoprosessia. [Leonard 2003.]

## 4.4 *Black and White*

Black and Whiteissa (B&W) pelaajan lemmikkiotus kykenee oppimaan pelin kuluessa kaikenlaisia uusia asioita, esimerkiksi faktoja, jotka liittyvät pelimaailman ympäristöön, miten tehdä tiettyjä tehtäviä, miten käyttäytyä tiettyjen objektien läheisyydessä ja mitä toimintoja tehdä tietyissä tilanteissa. Pelimaailma on lemmikkiotukselle siinä mielessä avoin, että tämä pystyy vapaasti vaikuttamaan melkeinpä kaikkiin siinä oleviin objekteihin. Otus kykenee myös tekemään omaan empiriiseen kokemukseen perustuvia päätelmiä objekteista, joihin tämä on vaikuttanut. Otukselle kehittyy myös ajan myötä oma persoonallisuus. Aiemmin vastaavanlaista kompleksisuutta ei ollut kaupallisissa tietokonepeleissä nähty, ja tästä syystä tämän uuden innovaation toteuttaminen oli pelinkehittäjillä yhtenä suurimmista haasteista.

### 4.4.1 Suunnittelu

B&W:n lemmikkiotuksen tuli toteuttaa kaksi hyvin erilaista vaatimusta. Otuksen haluttiin olevan persoonallinen eli tämän pelin tapauksessa *uskottava*, *mukautuvainen* ja *empaattinen*. Toiseksi sen haluttiin olevan pelaajalle hyödyllinen apuri, joka ei ole vain koe-eläin vaan jota voi oikeasti opettaa tekemään uusia asioita. Ensi silmäyksellä nämä kaksi vaatimusta näyttävät olevan ristiriitaisia. Persoonallisuusvaatimus antaa ymmärtää, että otus on autonominen toimija, kun taas hyödyllisyys näyttää viittaavaan siihen, että pelaaja pystyy alistamaan sen omaan tahtoonsa otuksen persoonallisuudesta riippumatta. [Evans 2002b.]

### 4.4.2 Agenttiarkkitehtuuri

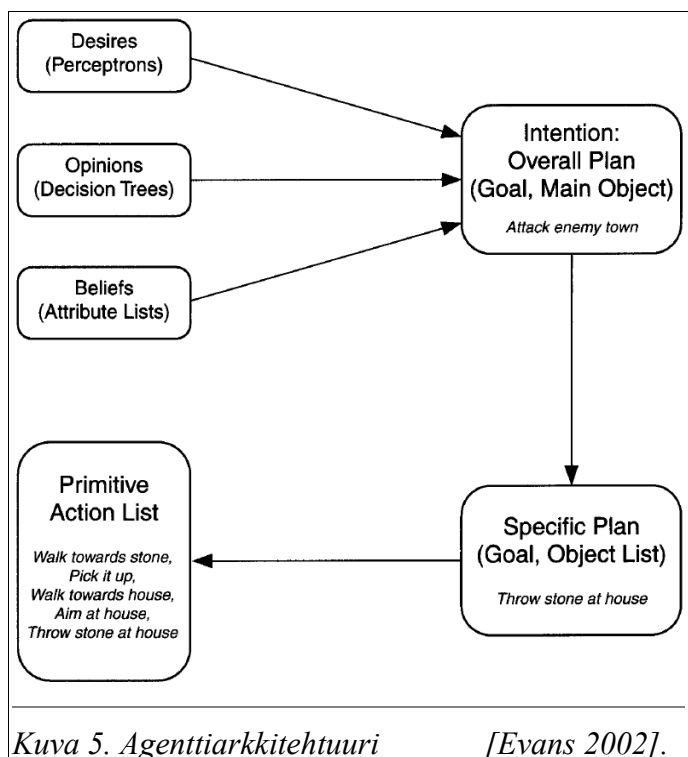
Ensinnäkin otuksen on siis oltava psykologisesti uskottava, jotta pelaaja pystyy kokemaan sen persoonalliseksi [Evans 2002b]. Tätä varten otettiin käyttöön filosofiasta ammennettu käsittekokonaisuus Uskomus-Halu-Aikomus (UHA, engl. Belief-Desire-Intention) [Bratman 1987], joka on pelitekoälyohjelmoijien keskuudessa itsenäiselle toimijalle jo melko yleinen ratkaisumalli. UHA:n ajatuksena on, että Uskomukset ja

Halut eivät ole vielä riittäviä luomaan mieltä, vaan lisäksi tarvitaan erillinen kategoria, Aikomukset, jotka esittävät Haluja, joita agentti on sitoutunut tekemään. B&W:n kohdalla Uskomukset säilövät tietoa tietyistä pelimaailman yksittäisistä objekteista ja Halut ovat tavoitteita, joita otus pyrkii pelissä täyttämään. Tämän lisäksi käsittekokonaisuutta laajennettiin vielä Mieli-pide-osalla (engl. Opinion), jonka avulla saadaan selville tietyn halun parhaiten täyttävät objektityypit. Kuva 5 esittää, miten Uskomukset, Halut ja Mieli-piteet luovat yhdessä Aikomuksen, jota sitten puolestaan käytetään tietyn suunnitelman (engl. specific plan) toteuttamiseen. Tässä suunnitelmalla tarkoitetaan toimintolistaa, jolla tietty aikomus saadaan täytettyä. [Evans 2002.]

Syynä tämänlaisen metodologian valintaan oli se, että arkkitehtuurissa ei haluttu toteuttaa yhdenmukaista rakennetta kaikille esitystavoille, esim. käyttämällä ainoastaan symbolista tai konnektionistista toteutusta, vaan sen sijaan käyttää monia erilaisia esitystapoja hyväksi tilanteen mukaan. Täten yksittäisiä objekteja koskevat uskomukset esitetään avainarvopareina; mielipiteet, joita otuksella on tietyistä objektityypeistä esitetään päätöspuina, eli tietyn tilanteen odotettavissa olevina tuloksina; halut esitetään perseptoreina, eli syötteiden ja tulosteiden verkkona, jossa syötteet ovat suoraan yhdistettyinä tulosteisiin; ja aikomukset esitetään suunnitelmina. Tämä on melko

luonnollinen tapa jaotella näitä esitystapoja, koska tässä uskomukset ja aikomukset ovat kovia symbolisia rakenteita, kun taas halut ovat hataria ja epäselviä eli tavallaan niin kuin todellisuudessakin. [Evans 2002.]

Suunnitelmaa tehtäessä agentti käy läpi jokaiseen aktiiviseen haluun soveltuvat uskomukset ja valitsee uskomuksen, josta sillä on paras mieli-pide (ks. kuva 7). Näin agentti muodostaa



Kuva 5. Agenttiarkkitehtuuri

[Evans 2002].

suunnitelman jokaiselle tavoitteelle (engl. goal) eli halulle. Tämän jälkeen agentti vertaa suunnitelmia näiden utiliteetin mukaan:

UTILITEETTI(HALU, OBJEKTI) = INTENSITEETTI(HALU) \* MIELIPIDE(HALU, OBJEKTI).

Tämän jälkeen agentti valitsee suunnitelman, jolla on suurin utiliteetti. Tämä suunnitelma on agentin aikomus. [Evans 2002.]

Jotta B&W:n otus olisi pelaajan mielestä psykologisesti *uskottava* agentti, niin silloin täytyy olla aina olemassa syy sille minkä takia otus on tiettyssä mielentilassa. Erityisesti silloin, jos agentilla on jotain objektia koskeva uskomus, tulee tämän uskomuksen pohjautua siihen, miten agentti on aiemmin havainnut kyseisen objektin. B&W:ssa otus on nimenomaan toteutettu tällä tavalla, eli se ei siis kykene saamaan tietoa muulla tavalla kuin aistiensa välityksellä. Evans käyttää tästä nimitystä tiedollinen todellisuuden tuntu (engl. epistemic verisimilitude). [Evans 2002.]

Lisäksi, jos otuksella on pelissä jokin halu, niin sille täytyy löytyä selitys. Jos otus on esimerkiksi loukkaantunut, niin se saattaa tulla vihaiseksi. Jokaisella halulla on tietty määrä erilaisia halulähteitä, jotka yhdessä määräävät sen, miten voimakas halu on kyseessä. [Evans 2002.]

Toinen vaatimus sille, että pelaaja kykenisi näkemään otuksen persoonallisena, on *mukautuvaisuus*. Lyhyesti kuvailtuna mukautuvaisuuteen kuuluu asioiden oppimiskyky monenlaisissa eri tilanteissa. Oppimista voi tapahtua UHA:n mukaisessa järjestelmässä seuraavilla kolmella eri tavalla: uskomusten oppiminen, eli otuksen kyky oppia esimerkiksi, että läheisessä kylässä on paljon ruokaa; dominoivien halujen oppiminen ja se miten herkkä otus on tiettyä halua kohtaan, eli esimerkiksi alhaisen energiatason ja nälän korrelaation oppiminen; mielipiteiden oppiminen esimerkiksi siitä, minkä tyylliset objektit parhaiten täyttävät tiettyjä haluja, eli minkälaisia objekteja kannattaa syödä ja minkälaisen kimppuun kannattaa hyökätä. [Evans 2002.]

Oppimisprosessi voi saada alkunsa seuraavilla neljällä eri tavalla: pelaajan suoran palautteen kautta eli palkitsemalla tai torumalla otusta; muita agenteja seuraamalla eli esimerkiksi pelaajan toimintoja havainnoimalla ja päättelämällä toimintojen päämääriä; käskyn kautta, eli jos otusta käsketään tekemään jotain, niin se oppii, että sellainen on tarkoituksenmukaista; lopuksi vielä oman kokemuksen kautta, eli jos otus tekee tietyn

toiminnon, koska haluaa saada jonkin halun täytetyksi, niin se voi jälkikäteen vielä analysoida lopputuloksen ja muuttaa toimintoon liittyvän mielipiteensä painoarvoa. Näin otus kykenee vastaisuudessa toimimaan mahdollisesti eri tavalla. [Evans 2002.]

What He Ate	Feedback—"How Nice It Tasted"
A big rock	-1.0
A small rock	-0.5
A small rock	-0.4
A tree	-0.2
A cow	+0.6

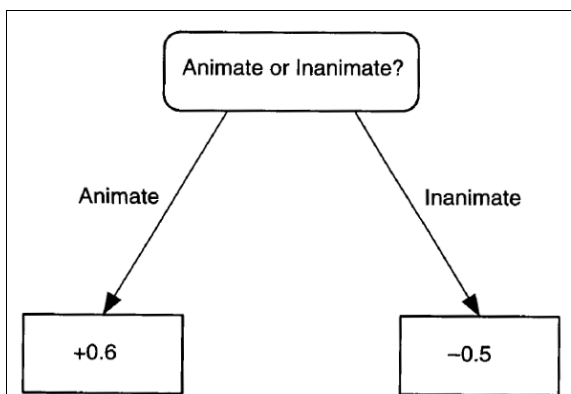
Kuva 6. Pelimaailman objekteihin liittyviä palautearvoja [Evans 2002].

Kokemuksen kautta oppiminen vastaa Russelin ja Norvigin [2003] oppivien agenttien yleistä mallia, jossa suorituskyykelementtinä on halun täytyminen, oppimiselementtinä mielipiteet (päättöpuu), ongelmageneraattorina itse pelimaailma ja kritiikkinä otuksen halut kokonaisuudessaan.

Kaikki edellä esitetyt oppimiskyvyt ja oppimisprosessin alkuunpaneuvat toiminnot on toteutettu B&W:ssa. Tässä tutkielmassa esitellään tarkemmin ainoastaan, miten mielipiteiden oppiminen tapahtuu B&W:ssa ja siihen paneudutaan seuraavaksi.

### 4.4.3 Mielipiteiden oppiminen

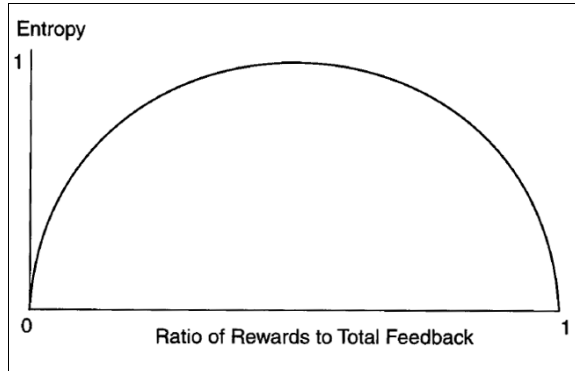
B&W:ssa otukset eli tekoälyagentit oppivat faktoja havainnoimalla tapahtumia ja tallentamalla niitä pitkäaikaiseen muistiinsa uskomuksina. Sen jälkeen, kun agentti on tehnyt jotain, tämä voi saada palautetta joko pelaajalta tai pelimaailmasta, jolloin se kykenee päättämään, miten hyvä tietty toiminto oli tehdä siinä tietyssä tilanteessa. [Evans 2002.]



Kuva 7. Mielipiteet [Evans 2002].

Miten agentti sitten oppii esimerkiksi minkälaisia objekteja on hyvä syödä? Se rakentaa päättöpuun kaikista syömistään objekteista sen mukaan, miten hyvältä ne maistuivat. Jos otus esimerkiksi on aiemmin syönyt kuvan 6 taulukon mukaisia objekteja, niin se

rakentaisi kuvan 7 mukaisen päätöspuun selittämään kyseiset syötteen. Kuvan 7 päätöspuussa objektit on jaoteltu erikseen mukaan, ovatko ne eläviä (engl. animate) vai eivät (engl. inanimate). Päästöpuu rakennetaan tarkastelemalla ominaisuuksia, jotka jakavat oppimiskokemukset parhaiten joukkoihin,



Kuva 8. Entropia [Evans 2002].

joilla on vastaavanlaiset palautearvot (engl. feedback values). Paras päätöspuu on sellainen, joka minimoi palautearvojen entropian jonkin objektin ominaisuuden mukaan (ks. kuva 8). [Evans 2002.]

Entropia kertoo, miten satunnaisia palautteet ovat. Jos palautteet ovat aina 0 tai 1, niin entropiaa ei ole, ja jos palautteet vaihtelevat välillä 0 ja 1, niin palaute on satunnaista ja entropia on tällöin suurta. Algoritmi, jota käytetään päätöspuiden dynaamiseen rakentamiseen entropian minimoimiseksi B&W:ssa, perustuu Quinlanin ID3-järjestelmään [Quinlan 1993], jossa puun rakennuksen jokaisessa vaiheessa valitaan ominaisuus, joka minimoi entropian. [Evans 2002.]

#### 4.4.4 Empaattisuus

Kolmantena vaatimuksena sille, että B&W:n otus olisi pelaajan mielestä persoonallinen, on *empaattisuus* eli tunneyhtymä pelaajan ja otuksen välillä. Empaattinen kiintymys on jo sinänsä molemminpuolista, koska se, minkä takia esimerkiksi unilelua kohtaan osoitetut tunteet ovat lapsellisia, johtuu siitä, että unilelu ei koskaan anna vastaavanlaista tunnetta takaisin. Tästä voidaan helposti päätellä, että mikäli halutaan luoda tietokonepeliin lemmikkiotus, johon pelaaja tuntee olevansa tunteellisesti sidottuna, tulee ensin varmistaa, että lemmikkiotus on myös empaattinen. Yleensä tietokonepeliagentit käyttäytyvät kuin vakavasti autistiset ihmiset, jotka kykenevät kyllä havainnoimaan maailman objekteja mutta jotka eivät kykene näkemään muita ihmisiä ihmisinä ja ennakoimaan näiden tekoja. [Evans 2002.]

B&W:ssa pelaajan lemmikkiotuksella on pelaajan mielestä yksinkertainen malli. Otus pyrkii formuloimaan tavoiteloja, jotka selittäisivät pelaajan tekojen syitä, ja tallentaa yksinkertaisen persoonallisuusmallin pelaajasta, mitä otus sitten käyttää hyväksi päätöksenteossaan. Tämän lisäksi otuksella on myös tavoitteita, jotka liittyvät suoraan pelaajaan: halu auttaa isäntäänsä, halu leikkiä isäntänsä kanssa ja halu saada huomiota. [Evans 2002.]

Jotta otus kykenisi tällaiseen muiden toimijoiden mentaaliseen mallintamiseen, niin agenttiarkkitehtuurin on tullut B&W:ssa nimenomaan perustua symbolisiin tietorakenteisiin. Jos arkkitehtuuri on nimittäin ainoastaan verkko numeerisia yhteyksiä (konnektionistinen), niin se on silloin jakamaton (holistinen) ja läpinäkymätön, jolloin erillisen mentaalisen mallin lyhyen kuvauksen, esim. pelaajan mielen mallin, irrottaminen järjestelmästä on merkittävästi haastavampaa. Tämä ei kuitenkaan tarkoita, että konnektionistisia oppimismenetelmiä tulisi silti hylätä, ja tästä syystä B&W agenttiarkkitehtuuri käyttää kynnyksifunktioita (engl. threshold functions) halujen toleranssin säätämiseen ja entropiafunktioita kohinan arvioimiseen. Nämä sumeat funktiot ovat sulautettuna B&W:ssa symboliseen arkkitehtuuriin. [Evans 2002.]

#### **4.4.5 Persoonallisuusvaatimus**

B&W:n otukset ovat siis persoonallisia, mutta B&W:ssä toinen merkittävistä pelillisistä asetetuista vaatimuksista otukselle on, että sen täytyy olla pelaajalle oikeasti myös hyödyllinen apuri. Nämä kaksi näyttävät olevan vastakkaisia vaatimuksia joten niiden ratkaiseminen on tärkeää. Ratkaisuna B&W:ssa päädyttiin siihen, että pelin alussa otus on täysin autonominen, mutta ajan mittaan ja opettamisen avulla pelaaja pystyy muokkaamaan otustaan. Tämä antaa pelaajalle tyydytyksen tunnetta, koska hän on tällöin itse opettanut otuksen toimimaan haluamallaan tavalla ja olemaan hyödyksi pelissä. Huonona puolena on, että otus kenties menettää osan vetovoimastaan, koska siitä tulee tällöin enemmän ja enemmän robottimainen. [Evans 2002b.]

#### 4.4.6 Tulevaisuuden suuntauksia

Mitä voitaisiin tehdä, jotta saadaan aikaan realistisempia agenteja tietokonepeleihin? Yhtenä vastauksena on luoda jotenkin loputon määrä tavoitteita. B&W:ssa otuksella on rajoitettu määrä tavoitteita, mutta ihmisillä sen sijaan on rajoittamaton määrä tavoitteita. Esimerkiksi: jos haluan, niin voin vaikka samantien päättää lähteä matkalle kohti Pariisia junalla, lentäen tai kävellen. B&W:ssa tavoitteet on mallinnettu haluina, joilla on sama kombinatorinen rakenne kuin ihmiskielellä, ja se täytyy tallentaa eksplisiittisesti. [Evans 2002b.]

Toinen vaihtoehto on luoda loputon määrä tavoitteiden täyttymiskeinoja eli implementoida reaaliaikainen suunnittelu peliin. B&W:ssa otus käyttää suunnittelukirjastoja päättääkseen, mitä tehdä. Syntymästään lähtien otus tietää, että on olemassa k tapaa täyttää jokin tietty tavoite eikä mitään muuta mahdollisuutta. Kun jotain tavoitetta pyritään täyttämään, niin otus käy vain läpi kaikki k tapaa ja valitsee sen, joka on tilanteeseen sopivin. Ihmisillä sen sijaan on tapana suunnitella tavoitteitaan reaaliajassa eli ns. ”hetkessä” ja tällöin on mahdollista keksiä jotain aivan uutta. Tämä on laskennallisesti hyvin vaativaa, mutta tekee agentista sopeutuvamman ja luovemman. [Evans 2002b.]

## 5 Johtopäätökset

Kaupallisissa tietokonepeleissä yleisesti hyödynnetyt tekoälytekniikat eivät ole vielä kehittyneet vastaavalla tavalla kuin esimerkiksi graafiikan piirtotekniikat, jotka mullistivat peliteollisuutta 1990-luvun loppupuolella, kun 3D-kiihdytinkortit alkoivat yleistyä markkinoilla. Nykyisin käytössä olevat menetelmät ovat yhä hyvin alkeellisia, koska ne sisältävät pääasiassa animointia ja reitinhakua, joskin ne ovat tilanteeseen sopivia. Näitä alkeellisia tekniikoita käyttämällä saadaan aikaan haluttu toiminnallisuus, ja tuotettu peli myy hyvin siitä riippumatta, koska pelaajia ei yleensä kiinnosta, miten tekoäly on toteutettu, kunhan se vain toimii riittävän hyvin. Aikaa pelin tuottamiseen kuluu enemmän, koska tekoäly täytyy tällöin ohjelmoida käytännössä ”kädestä pitäen”, jolloin riski sille, että tekoäly toimii virheellisesti, on mahdollisimman pieni. Muutamit harvat pelit näyttävät suuntaa, mihin ollaan mahdollisesti menossa, mutta ne ovat yhä harvinaislaatuisia erikoistapauksia kuten Thief: The Dark Project ja Black and White.

# LÄHTEET

## Kirjallisuus

**Amd 2008:** ATI Radeon™ HD 4800 Series - GPU Specifications,

<http://ati.amd.com/products/radeonhd4800/specs.html> , 2008, haettu 5.10.2008.

**BBC News 2005:** Cost of making games set to soar,

<http://news.bbc.co.uk/2/hi/technology/4442346.stm> , 17.11.2005, haettu 9.09.2008.

**Bourg, D. ja Seeman, G. 2004:** AI for Game Developers, O'Reilly, 2004.

**Bratman Michael 1987:** Intention, Plans and Practical Reason, Harvard University Press, 1987.

**Bushnell, N. 1996:** Relationships between fun and the computer business,

Communications of the ACM Volume 39 Issue 8 , ACM, elokuu 1996.

**Chamandard Alex J. 2004:** AI Game Development, New Riders Publishing, 2004.

**Chamandard Alex J. heinäkuu 2008:** Game AI Coming of Age,

<http://aigamedev.com/premium/presentations/game-ai-coming-of-age> , Paris Game AI Workshop, heinäkuu 2008, haettu 5.10.2008.

**Chamandard Alex J. tammikuu 2008:** Examining the AI Scripts in Call of Duty 4's

Developer Tools, <http://aigamedev.com/source/call-of-duty-4-tools> , tammikuu 2008, haettu 5.10.2008.

**CNET Networks 2008:** 'Grand Theft Auto IV' nets Guinness record,

[http://news.cnet.com/8301-13772\\_3-9942917-52.html](http://news.cnet.com/8301-13772_3-9942917-52.html), CNET Networks, Inc., 13.5.2008, haettu 9.09.2008.

**Evans Richard 2002:** Varieties of Learning, AI Game Programming Wisdom, Charles River Media 1. painos, 2002.

**Evans Richard 2002b:** AI in Games: A Personal View,

<http://www.gameai.com/blackandwhite.html> , 2002, haettu 12.10.2008.

**Future Publishing 2007:** "GTA Gets Real", PlayStation Official Magazine (UK) (6):

54–67, Future Publishing, kesäkuu 2007.

**Hannan Jeff 2001:** "Jeff Hannan Interview", <http://www.generation5.org/content/2001/>

hannan.asp, 2001, haettu 4.10.2008.

**IMDb 2008:** Dark Knight Breaks \$300 Million Record,

<http://www.imdb.com/news/ni0267871/> , Internet Movie Database, 27.7.2008, haettu 9.09.2008.

**Kuutila Timo 2006:** Johdatus tekoälyyn luentokalvot, Turun yliopiston Informaatioteknologian aineopintokurssi, luennoitu syksyllä 2006.

**Leonard Tom 1999:** Postmortem: Looking Glass's Thief: The Dark Project, Game Developer Magazine, heinäkuu 1999.

**Leonard Tom 2003:** GDC 2003: Building an AI Sensory System: Examining The Design of Thief: The Dark Project,

[http://www.gamasutra.com/gdc2003/features/20030307/leonard\\_01.htm](http://www.gamasutra.com/gdc2003/features/20030307/leonard_01.htm) , Gamasutra, maaliskuu 2003, haettu 15.9.2008.

**Lionhead Studios 2008:** The History of Lionhead Studios,

<http://www.lionhead.com/History.aspx> , 2008, haettu 15.9.2008.

**McCarthy, J. 2004:** What Is Artificial Intelligence?, [http://www-](http://www-formal.stanford.edu/jmc/whatisai/)

[formal.stanford.edu/jmc/whatisai/](http://www-formal.stanford.edu/jmc/whatisai/) Computer Science Department, Stanford University, 12.7.2008, haettu 9.09.2008.

**Miller George 1956:** "The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information", The Psychological Review, vol. 63, 81-97, 1956.

**Millington Ian 2006:** Artificial Intelligence for Games, Morgan Kaufmann, 2006.

**Molyneux Peter 2001:** Postmortem: Lionhead Studios' Black & White, Game Developer Magazine kesäkuu 2001.

**MPAA 2008:** Frequently requested statistics,

<http://www.mpa.org/researchStatistics.asp> , The Motion Picture Association of America (MPAA) , 26.8.2008, haettu 1.09.2008.

**PricewaterhouseCoopers 2008:** Global Entertainment and Media Outlook: 2008-2012, PricewaterhouseCoopers, 2008.

**Quinlan JR. 1993:** Programs for Machine Learning, Morgan Kauffman, 1993.

**Rosmarin, R. 2006:** "Take Two Takes A Hit",

<http://www.forbes.com/technology/2006/12/11/games-gta-options-tech-enter->

cx\_rr\_1211taketwo.html , Forbes, 12.11.2006, haettu 9.09.2008.

**Russell, S. ja Norvig, P. 2003:** Artificial Intelligence a Modern Approach, toinen kansainvälinen painos, Prentice Hall, 2003.

**Schaeffer Jonathan , Burch Neil , Björnsson Yngvi, Kishimoto Akihiro, Müller Martin , Lake Robert , Lu Paul ja Sutphen Steve 2007:** Checkers Is Solved, Department of Computing Science, University of Alberta, Canada, Science 14 syyskuuta 2007: Vol. 317. nro. 5844, 1518 - 1522, 2007.

**Schaeffer Jonathan ja Plaat Aske 1997:** Kasparov versus Deep Blue: The Re-match, ICCA Journal: vol. 20, nro. 2, 95-102, 1997.

**Schwab Brian 2004:** AI Game Engine Programming, Charles River Media, Inc., 2004.

**Shopf Jeremy , Barczak Joshua , Oat Christopher ja Tatarchuck Natalya 2008:** March of the Froblins: Simulation and Rendering Massive Crowds of Intelligent and Detailed Creatures on GPU, Advances in Real-Time Rendering in 3D Graphics and Games Course - SIGGRAPH 2008, 2008.

**Turing Alan 1950:** Computing machinery and intelligence, Mind, 59, 433-460, 1950.

**Woodcock Steven 2000:** Game AI: The State of the Industry, Game Developer magazine, elokuu 2000.

**Woodcock Steven 2002:** Game AI: The State of the Industry 2001–2002, Game Developer Magazine heinäkuu 2002.

**Woodcock Steven 2003:** AI Roundtable Moderator's Report, <http://www.gameai.com/cgdc03notes.html> , Game Developer's Conference, 2003 , haettu 15.9.2008, haettu 15.08.2008.

**Yannakakis Georgios N. 2005:** AI in Computer Games: Generating Interesting Interactive Opponents by the use of Evolutionary Computation, University of Edinburgh, 2005.

**Yannakakis Georgios N. Ja Hallam John 2007:** Capturing Player Enjoyment in Computer Games, University of Southern Denmark, 2007.

## **Pelit**

**Codemasters 2001:** Colin McRae Rally 2.0, 2001.

**Infinity Ward 2007:** Call of Duty 4, Activision, 2007.

**Lionhead Studios 2001:** Black and White, EA Games, 2001.

**Looking Glass Studios 1998:** Thief: The Dark Project, Eidos Interactive, 1998.

**Maxis 2000:** The Sims, Electronic Arts, 2000.

**Rockstar Games 2008:** Grand Theft Auto IV, Rockstar Games ja Capcom, 2008.

**Valve Software 1998:** Half-Life, Sierra Studios, Electronic Arts ja Valve Software, 1998.

## **Elokuvat**

**WB 2008:** The Dark Knight, Warner Bros. Pictures 2008.