Jari Vitikainen

SATUNNAISUUS JA SEN SOVELTAMINEN OMASSA PELISSÄ

Case: Stratogear

Opinnäytetyö
Tietojenkäsittelyn koulutusohjelma

Joulukuu 2013

MIKKELIN AMMATTIKORKEAKOULU
Mikkeli University of Applied Sciences

KUVAILULEHTI

MIKKELIN AMMATTIKORKEAKOULU Mikkeli University of Applied Sciences	**Opinnäytetyön päivämäärä** **3.12.2013**
Tekijä(t) Jari Vitikainen	**Koulutusohjelma ja suuntautuminen** Tietojenkäsittelyn koulutusohjelma
Nimeke Satunnaisuus ja sen soveltaminen omssa pelissä	

Tiivistelmä

Jokaisella ihmisellä on jonkinlainen käsitys satunnaisuudesta. Tässä opinnäytetyössä käydään tarkemmin läpi satunnaisuuttaja tarkastellaan satunnaisuutta eri tieteiden näkökulmasta ja kuinka kyseiset tieteet käyttävä hyväkseen satunnaisuutta. Pyrin myös selventämään, kuinka satunnaisuus eroaa mielivaltaisuudesta.

Opinnäytetyössäni pyrin myös selvittämään, kuinka satunnaisuutta käytetään peleissä. Kyseisessä luvussa käyn myös läpi, kuinka peleissä käytetty satunnaisuus voidaan jakaa eri kategorioihin. Pyrin tämän lisäksi selventämään näiden kategorioiden hyvä ja huonoja puolia.

Vaikka tietokoneilla luoduissa peleissä on usein löydettävissä satunnaisuutta, ei tämä kuitenkaan ole todellista satunnaisuutta. Tästä syystä pyrin myös opinnäytetyössäni selventämään, kuinka tietokoneella luotu satunnaisuus ei ole oikeaa satunnaisuutta ja millaista tämä oikea satunnaisuus sitten on. Kyseisessä luvussa esiin nousevat pseudo-satunnaisuus ja todellinen satunnaisuus sekä kuinka näitä on käytetty hyväksi peleissä.

Käytännön toteutuksena satunnaisuudesta luotiin pelin demoversio, joka olisi tarkoituksena viedä pitemmälle valmistumisen jälkeen. Pelissä käytettävä satunnaisuus on pseudo-satunnaisuutta ja syy tähän selvenee opinnäytetyöni raportissa. Päätännässä pyrin kokoamaan satunnaisuuden helposti ymmärrettäväksi tiivistelmäksi ja selventämään, kuinka käytännön osuutena tuotettu peli tulee jatkossa muuttumaan.

Asiasanat (avainsanat)
Satunnaisuus, informaatioteoria, salausmenetelmät, tilastotiede, peliteoria, mielivaltaisuus, peli

Sivumäärä	**Kieli**	**URN**
29	Suomi	

Huomautus (huomautukset liitteistä)

Ohjaavan opettajan nimi	**Opinnäytetyön toimeksiantaja**
Jukka Selin	Mikkelin ammattikorkeakoulu

DESCRIPTION

MIKKELIN AMMATTIKORKEAKOULU Mikkeli University of Applied Sciences	**Date of the bachelor's thesis** 3rd of December 2013
Author(s) Jari Vitikainen	**Degree programme and option** Business Information Technology

Name of the bachelor's thesis

Randomness and its application in self-made game

Abstract

Every person has his/her own impression of randomness. This batchelor´s thesis introduced randomness and how it is perceived by different scientific faculties. I also intended to clarify the difference between randomness and arbitrariness and how randomness is typically used in games. In this part of the study I introduced how to divide randomness in gaming to different categories and what kind of advantages and disadvantages these categories had.

Even though games made with computers use randomness often, it is not true randomness. Therefore, I also intended to clarify why computer-made randomness was not real randomness and how randomness could be truly random. This part of the study introduced you to pseudo-randomness and true randomness as well as how they were utilized in games.

As a practical implementation of randomness, a game demo was made. It is my intention to continue developing this game even after graduation. The randomness used in this game was only pseudo-randomness and the study explained the reason for this. The study ended with an clearly understandable summary of and the plans for continuing to develop the game.

Subject headings, (keywords)

Randomness, information theory, encryption methods, statistics, game theory, arbitrariness, game

Pages	**Language**	**URN**
29	Finnish	

Remarks, notes on appendices

Tutor	**Bachelor's thesis assigned by**
Jukka Selin	Mikkeli University of Applied Sciences

SISÄLTÖ

1 JOHDANTO

Satunnaisuutta käytetään useissa erilaisissa tieteissä ja teorioissa. Mitä tämä satunnaisuus kuitenkin on ja miten se ilmenee? Opinnäytetyössäni pyrin käsittelemään satunnaisuutta yleisesti ja kuinka eri tieteenalat käyttävät satunnaisuutta. Pyrin myös selvittämään, kuinka tätä satunnaisuutta käytetään peleissä ja kuinka tätä satunnaisuutta voidaan soveltaa peleihin käytännössä.

Aluksi pyrin kertomaan, mitä satunnaisuus yleensä tarkoittaa ja kuinka ihmiset yleensä käsittävät satunnaisuuden. Tämän jälkeen pyrin kertomaan, kuinka satunnaisuutta käytetään tieteissä. Satunnaisuus on käytössä useissa eri tieteen teorioissa. Näitä teorioita ovat informaatioteoria, salausmenetelmät, tilastotiede ja peliteoria.

Peliteorian käsittelyn jälkeen on hyvä käydä hieman läpi satunnaisuuden käyttöä peleissä kautta aikain. Tarkoituksena on selvittää, kuinka pelit ovat käyttäneet hyväkseen satunnaisuutta aina ensimmäisten pelien ajoista lähtien. Lisäksi pyrin selvittämään, kuinka satunnaisuuden käyttö on muuttunut peleissä ja millaista satunnaisuutta pelit käyttävät hyväkseen. Tässä osiossa käyn myös läpi esisatunnaisuuden ja jälkisatunnaisuuden sekä kuinka nämä eroavat toisistaan.

Tavoitteenani on myös kertoa, kuinka tietokoneella luotu satunnaisuus ei yleensä ole oikeaa satunnaisuutta. Kyseisessä luvussa selitän, kuinka tietokoneella luotu satunnaisuus on yleensä pseudo-satunnaisuutta. Pseudosatunnaisuuden jälkeen kerron, millaista todellinen satunnaisuus on ja kuinka todellista satunnaisuutta voitaisiin luoda tietokoneelle.

Pelejä tehdessä on kuitenkin varottava, ettei sattumanvaraisuus muutu mielivaltaisuudeksi. Siksi on selvitettävä tarkasti, miten mielivaltaisuus eroaa satunnaisuudesta ja missä kulkee mielivaltaisuuden ja satunnaisuuden raja. Pyrin myös selventämään satunnaisuuden ja mielivaltaisuuden eroja esimerkeillä.

Kun satunnaisuudesta on saatu parempi käsitys, on hyvä lähteä kertomaan, kuinka satunnaisuutta on käytetty tekemässämme pelissä. Tarkoituksena on kertoa ensin hieman ohjelmasta, jolla peli on tehty. Tämän jälkeen käydään läpi pelin toiminnot pää-

piirteittäin ja lopuksi käyn tarkemmin läpi pelin tapahtumat, jossa satunnaisuutta on käytetty. Näitä tapahtumia läpikäydessä aion kertoa, millä tavalla satunnaisuutta on käytetty kyseisessä kohdassa ja kuinka kyseisen tapahtuman olisi voitu mahdollisesti tehdä toisin. Käyn myös läpi kohtia, joita emme pelin demoon voineet lisätä ajan puutteen vuoksi.

Lopuksi pyrin kokoamaan ajatukseni ja tulevaisuuden näkymäni käytännön osuutena toteutettuamme peliä koskien. Kyseisessä luvussa selvitän myös, mikä käytännön osuuden toteutuksessa jäi vaivaamaan minua. Pyrin myös selventämään omat ajatukseni satunnaisuudesta ja mikä on näkemykseni satunnaisuudesta käytyäni aihetta tarkemmin läpi.

2 SATUNNAISUUS JA SEN KÄYTTÖ TIETEISSÄ

Kaikilla ihmisillä on jonkinlainen käsitys siitä, mitä satunnaisuudella tarkoitetaan. Satunnaisuus yhdistetään yleisesti esimerkiksi nopan heittoon. Kun noppaa heitetään, on yhtä mahdollista saada mikä tahansa nopan luvuista. Tällöin voidaan sanoa, että nopan tulos on sattumanvarainen. Tässä vaiheessa joku voisi väittää, että tämä ei pidä paikkaansa. Heidän mukaansa noppaa heitettäessä saattaa joitain lukuja tulla useita ja joitain ei välttämättä ollenkaan. Tämä johtuu kuitenkin vain siitä, ettei noppaa ole heitetty tarpeeksi useasti. Kun noppaa heitetään paljon, pitäisi kaikkia lukuja tulla yhtä paljon.

2.1 Satunnaisuus selitettynä

Yksi parhaimmista esimerkeistä satunnaisuudesta on kolikon heitto. Kun kolikkoa heitetään, on yhtä suuri mahdollisuus saada kruuna tai klaava. Todennäköisyys saada kruuna tai klaava on tällöin 50 % tai ½. On siis täysin sattumanvaraista, kummalle puolelle kolikko putoaa heitettäessä. Paul Lutus (2010) esittelee artikkelissaan edelliseen esimerkkiin liittyen teorian uhkapelurin harhasta. Lutus (2010) kertoo, kuinka uhkapelurit uskovat siihen, että jos he ovat hävinneet uhkapelissä useita kertoja putkeen, on heidän voiton todennäköisyytensä suurempi ja he jatkavat pelaamista. Tämä uhkapelurien harha ei tietenkään pidä paikkaansa, vaan kuten Lutus (2010) toteaa artikkelissaan, on voiton todennäköisyys aina sama kolikkoa heitettäessä. Tämä tarkoit-

taa sitä, että jos ensimmäisellä heitolla on saatu klaava, on toisen heiton todennäköisyys saada klaava edelleen 50 %. Tämä pätee myös tuleviin heittoihin.

Joskus näitä satunnaisia luku on kuitenkin saatava suuri määrä. Näissä tapauksissa kukaan tuskin lähtee heittämään esimerkiksi tuhat kertaa kolikkoa. Tällaisissa tapauksissa apuna voidaan käyttää matematiikkaa. Cormer ym. (2001, 94–96) esittelevät kirjassaan tavan, jolla tällaisesta tapauksesta saadaan laskettua satunnaistapausten lukumäärä. Oletetaan siis, että joudutaan heittämään kolikkoa *n* kertaa. Kolikkoa heitettäessä on yhtä suuri todennäköisyys saada kruuna taikka klaava. Merkatkoon *Xi* sitä satunnaista muuttujaa, jolloin heitto *i* on klaava. Olkoon *X* klaavojen määrä *n* heittojen jälkeen, jolloin

$$X = \sum_{i=1}^{n} X_i \ .$$

Koska pyrimme saamaan klaavojen odotetun määrän, merkitsemme odotuksen lausekkeen molemmille puolille

$$\mathrm{E}[X] = \mathrm{E}\left[\sum_{i=1}^{n} X_i\right] \ .$$

Tämän jälkeen on helppo laskea klaavojen lukumäärä, kun kolikkoa on heitetty *n* kerta

$$\begin{aligned} \mathrm{E}[X] &= \mathrm{E}\left[\sum_{i=1}^{n} X_i\right] \\ &= \sum_{i=1}^{n} \mathrm{E}[X_i] \\ &= \sum_{i=1}^{n} 1/2 \\ &= n/2 \ . \end{aligned}$$

Tulokseksi saatiin tällöin *n/2*. Kaavan tulos näyttää nyt hyvin yksinkertaiselta, mutta tämä johtuu siitä, että käytimme esimerkissä kolikon heittoa. Kolikkoa heitettäessä on mahdollista saada vain kruuna taikka klaava yhtä suurella todennäköisyydellä, joten lopputulos on hieman yksinkertaisempi. Satunnaistapahtumien määrän vaihtuessa, muuttuu laskukaavakin sen mukaisesti.

Satunnaisuutta esiintyy monissa muissakin asioissa, kuin nopan heitossa tai kolikon heitossa. Satunnaisuutta käytetään myös monissa eri tieteissä. Satunnaisuutta käytetään informaatioteoriassa, salausmenetelmissä, tilastotieteessä ja peliteoriassa.

2.2 Informaatioteoriassa

Informaatioteoria tutkii informaation mittaamista ja siirtämistä. Yksinkertaisillaan informaatioteoria sisältää tiedon lähteen, tiedonsiirtoon vaadittavan kanavan ja tiedon vastaan ottajan. Kulkiessaan kanavan läpi tieto kuitenkin saa häiriötä ja tulee siksi häiriöineen vastaanottajalle. Informaatioteoriassa pyritään selvittämään, kuinka tieto saadaan siirrettyä vastaanottajalle mahdollisimman tiiviisti, jotta häiriö jäisi mahdollisimman vähäiseksi.

Kuinka satunnaisuus sitten liittyy informaatioteoriaan? Lasse Holmström (2012) esittää esimerkin esitelmässään, kuinka satunnaisuus liittyy informaatioteoriaan. Holmströmin (2012) esimerkissä käytetään 100 palloa, joista 10 on valkoisia ja loput mustia. Tällöin saadaan valkoisten pallojen suhteeksi 1/10 ja mustien 9/10. Näistä palloista nostetaan satunnaisesti yksi. Jos pallo on valkoinen, saadaan paljon enemmän tietoa, kuin siinä tapauksessa, että pallo on musta.

Tämä johtuu siitä, että valkoisen pallon noston jälkeen on toisen valkoisen pallon nostaminen paljon epätodennäköisempää, kuin aikaisemmin. Tämä tarkoittaa sitä, että en satunnaisesti nostettu pallo on pienemmällä todennäköisyydellä valkoinen. Väittämä pitää paikkaansa vain jos aikaisemmin nostettu pallo on poistettu noston jälkeen.

Informaatioteorian mukaan tiedon siirtymiseen liittyy aina tiedon siirron aikana tietoon on mahdollista syntyä häiriöitä. Yllä olevassa tapauksessa nämä häiriöt voisivat olla muun muassa seuraavia: pallo on unohdettu poistaa noston jälkeen, pallon väri on

merkitty väärin tietoa tallettaessa tai palloja saattaa olla enemmän kuin mitä oli tarkoitettu. Näistä esimerkeistä kaikki voivat olla mahdollisia inhimillisen virheen takia. Ihmisten siirtäessä tietoa toisilleen on hyvin todennäköistä, että siirretty tieto kärsii häiriöstä.

2.3 Salausmenetelmissä

Salausmenetelmiä on käytetty läpi ihmisten historian, ja aina kun salausmenetelmä on murrettu, on tilalle täytynyt keksiä uusi vaikeampi menetelmä. Tunnetuin salausmenetelmä on luultavasi Caesarin salaus, joka perustuu aakkosten kiertoon. Kyseisessä salausmenetelmässä jokainen viestin kirjain korvataan kirjaimella, joka saadaan kun siirrytään aakkosissa n kirjainta eteenpäin. Korhonen (2013) antaa tutkielmassaan seuraavanlaisen esimerkin Caesarin salauksesta kun $n = 3$. Alkuperäinen viesti kuuluu seuraavasti

"JULKISEN AVAIMEN SALAUS"

ja kun lisätään Caesarin salaus

"MXONLVHQDYDLPHQVDOXV".

Caesarin salauksella salattu viesti on kuitenkin helposti purettavissa nykymenetelmin, joten salausten on oltava paljon monimutkaisempia. Tässä vaiheessa mukaan tulee satunnaisluvut. Kun viestin jokainen kirjain korvataan satunnaisluvulla, on viestin selvittäminen mahdotonta ilman, että tunnetaan satunnaisluvun saamiseksi käytettävää laskukaavaa, sekä kyseisessä laskukaavassa käytettävää alkulukua.

Hyvillä salausmenetelmillä on suuri merkitys nykypäivänä. Salausmenetelmiä tarvitaan esimerkiksi salasanoissa. Monet ohjelmat ja internet sivut vaativat salasanoja, jotta sivustolla voi liikkua vapaasti. Tällöin kyseessä on yleensä kirjautuminen aiemmin luodulle tilille.

Petteri Järvinen esittelee kirjassaan (2003) kuinka voi käydä, jos salausmenetelmissä käytetty satunnaisluku on huonosti laadittu. Järvinen (2003) kertoo, kuinka kaksi

opiskelijaa oli lähtenyt tutkimaan Netscape selaimen lähdekoodia. Opiskelijat löysivät SSL-salauksessa käytettävän istuntoavaimen muodostuksen. Sen sijaan, että istuntoavain olisi muodostettu satunnaisluvulla, muodostettiin istuntoavain käyttämällä kellonaikaa, prosessin ID-numeroa ja isäntäprosessin ID-numeroa. Järvisen (2003) mukaan tämä ei ollut hyvä salausmenetelmä, koska samalle koneelle kirjautuneet pystyivät helposti laskemaan avaimen ja tämän avulla murtamaan salauksen.

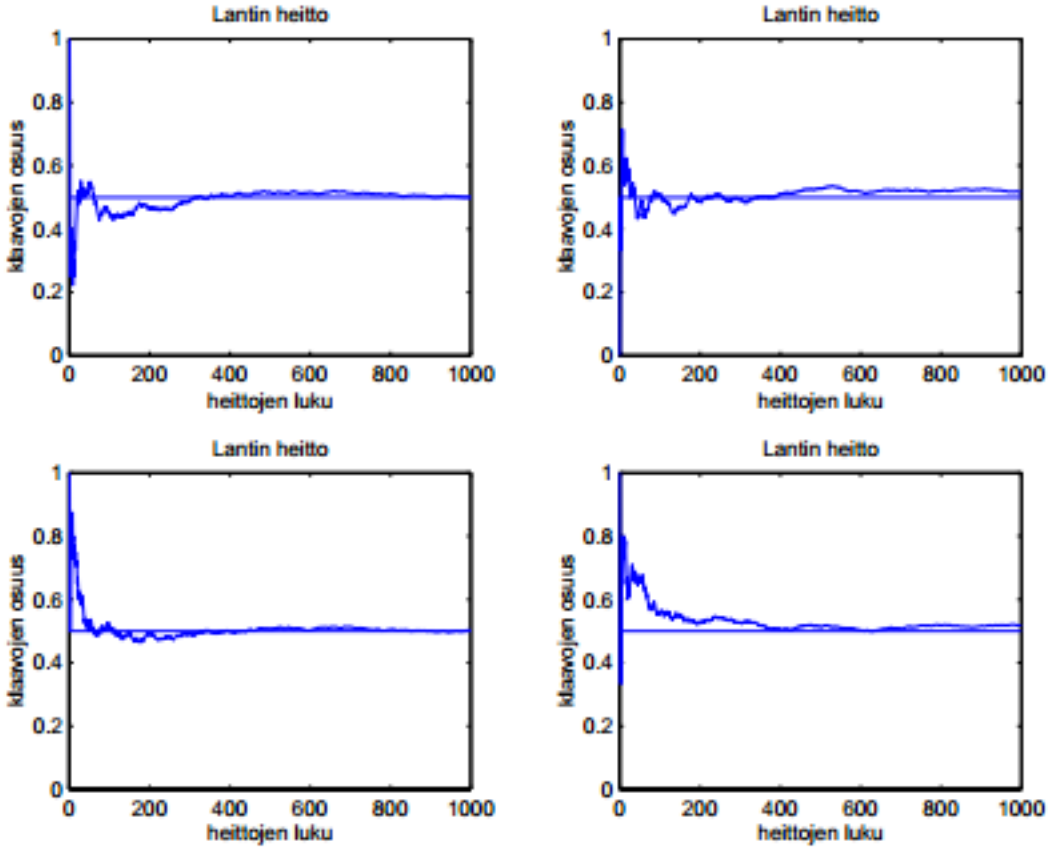
2.4 Tilastotieteessä

Vaikka tilastotieteiden nimi saattaisi antaa joillekin sellaisen käsityksen, että tilastotiede olisi oppi tilastoista ja tilastojen tuotannosta. Näin ei kuitenkaan ole, vaan tilastojen tuottaminen, analysointi ja jalostaminen ovat vain osa tilastotiedettä. Vaikka tilastotiede käyttää havainnollistamisessaan matemaattisia malleja ja menetelmiä, ei tilastotiede ole puhtaasti matematiikan osa-alue.

Kuinka satunnaisuus sitten liittyy millään tavalla tilastotieteeseen? Tilastotieteessä tutkitaan reaalimaailman ilmiöitä kuvaavia numeerisia ja kvantitatiivisia tietoja. Näihin tietoihin liittyy kuitenkin usein satunnaisuutta. Esimerkiksi tietoja kerätessä on voinut tapahtua inhimillinen virhe taikka laite, jolla tietoja kerätään, on voinut oikutella. On myös hyvin mahdollista, että tietoja kerätessä on tieto saatu uniikista tapahtumasta, joka on seuraus useasta satunnaisesta tapahtumasta. Tällaiset tapaukset muokkaavat kerättyjä tietoja ja siksi niihin liittyy aina epävarmuutta ja satunnaisuutta.

Tilastotieteellä pyritään selvittämään kerätyissä tiedoissa esiintyvien satunnaisuuksien säännönmukaisuutta. Ilkka Mellin (2010) kertoo tämän säännönmukaisuuden tarkoittavan tilastollista stabiliteettia tilastotieteessä. Ilman tätä säännönmukaisuutta on tapahtuma mielivaltaisuutta, eikä satunnaisuutta.

Selvittääksemme hieman tarkemmin satunnaisuuden osuutta tilastotieteessä, ottakaamme avuksemme jälleen vanha kunnon kolikon heitto. Heittäkäämme kolikkoa 1000 kertaa kirjaten klaavojen osuuden näistä heitoista. Kuvassa 1 näkyy Pohjavirran ja Ruhosen (2005) Mathlabilla luotu koe samanlaisesta kolikon heitosta.



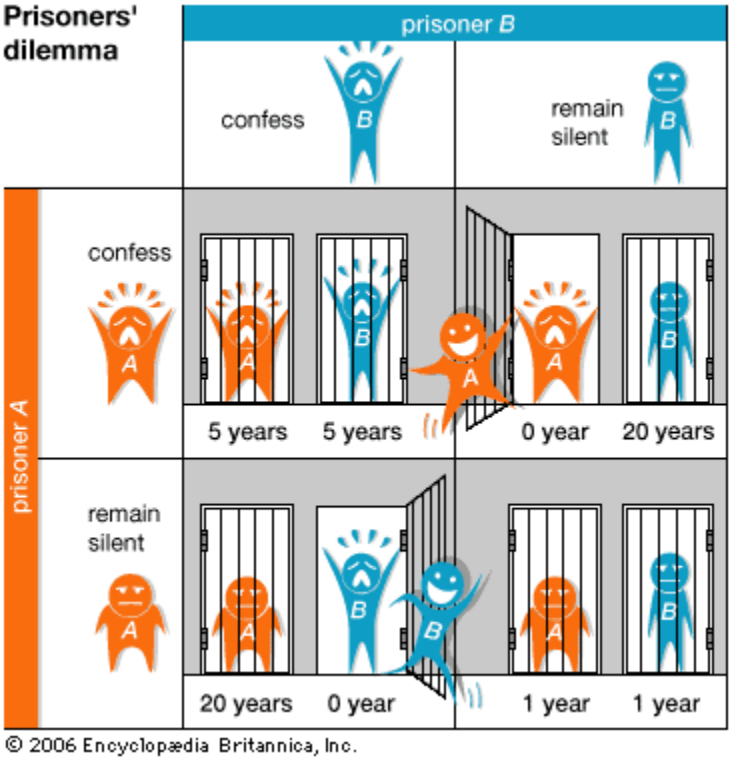
KUVA 1. Klaavojen osuus tuhannesta kolikon heitosta

Kuten kuvasta 1 on nähtävissä, alkaa klaavojen osuus stabiloitua, kun heittojen lukumäärä kasvaa. Vaikka jokainen Pohjavirran ja Ruhosen (2005) tekemistä kokeista on yksilöllinen, alkaa klaavojen osuus stabiloitumaan silmämääräisesti noin 400 heiton paikkeilla. Tilastotieteellä pyritään tutkimaan juuri tätä yllä olevassa testissä havaittua stabiloitumista satunnaisissa tapahtumissa.

2.5 Peliteoriassa

Peliteoria on tieteenala, jossa pyritään selvittämään kahden agentin tai toimijan välistä strategista kanssakäymistä ja valinnan tekemistä. Kuvitellaan esimerkiksi tilanne, jossa tapaat uuden ihmisen. Tällaisissa tilanteissa yleensä esittäydytään ja kätellään. Tässä vaiheessa joudut valinnan eteen. Esittäydytkö ja kättelet, vaiko käyttäydyt jollain aivan toisella tavalla. Peliteoria pyrkii tutkimaan juuri tätä kanssakäymistä ja sen aikana tehtäviä strategisia valintoja.

Tunnetuin peliteorian esimerkki on vangin dilemma. Mike James (2012) selventää vangin dilemman tarkoittavan tilannetta, jossa kaksi vankia on tehnyt rikoksen. Rikolliset ovat etukäteen sopineet, että kiinni jäädessään molemmat pysyvät vaiti. Rikolliset otetaan kiinni ja sijoitetaan eri selleihin niin, että he eivät pysty enää keskustelemaan keskenään. Tässä vaiheessa kummallekin annetaan vaihtoehto tunnustaa rikoksensa, sysäten kaikki syytteet toiselle vangille ja päästen itse vapaaksi kuten kuvasta 2 on nähtävissä.



KUVA 2. Vangin dilemma

Tässä tilanteessa yhteisen hyvän nimissä vankien olisi parasta pysyä hiljaa, kuten he olivat aikaisemmin sopineet, sillä silloin molempien vankeustuomio olisi vain yksi vuosi. Tilanteesta tekee monimutkaisen aikaisemmin mainittu vangitsijoiden ehdotus. Jos vain yksi vangeista tunnustaa, on hän vapaa ja hiljaa pysynyt saa 20 vuoden vankeusrangaistuksen. Toisaalta, jos molemmat vangit tunnustavat, saavat molemmat viisi vuotta vankeutta. Koska vangit eivät enää voi keskustella keskenään, ei heillä ole varmuutta siitä, onko toinen pysynyt hiljaa kuten oli sovittu.

Chris Lon artikkelin mukaan (2012) peliteoriaa olaan myös soveltamassa Los Angelesin lentokentän turvatoimissa. Lo haastatteli kyseisessä artikkelissa professori Milind

Tambea siitä, kuinka peliteoriaa voitaisiin soveltaa kyseisissä turvatoimissa. Tambea muutti lentokentän turvallisuuden peliteorian termeiksi, jolloin saatiin kaksi toimijaa. Lentokentän turvatoimet ja terroristit. Nämä kaksi toimijaa ovat jatkuvasti strategisessa kanssakäymisessä toistensa kanssa.

Lentokentällä pyritään parantamaan turvatoimia ja terroristit yrittävät kiertää kyseiset turvatoimet. Lon artikkelista (2012) saatiin selville, että lisäämällä satunnaisuus partioiden liikkumiseen ja vaihtumiseen, saadaan lentokentän turvatoimista paremmat. Tämä johtuu siitä, että terroristien on vaikeampi murtaa turvatoimia, kun strateginen tilanne vaihtuu satunnaisesti.

Satunnaisuus on aina osana ihmisten välistä strategista kanssakäymistä. Kuvitellaan esimerkiksi tilanne, jossa kaksi pelaajaa pelaa strategista tietokonepeliä. Pelatessaan pelaajat voivat yrittää päätellä toistensa valintoja ja toimia sen mukaan, mutta pelaajat eivät voi olla varmoja toisen pelaajan valinnoista, ennen kun kyseinen valinta on tullut käytäntöön. Tästä syystä toisen pelaajan valinnat voivat näyttää vastapelaajalle satunnaisilta.

3 SATUNNAISUUDEN KÄYTTÖÄ PELEISSÄ

Satunnaisuutta on käytetty peleissä todella kauan ennen tietokoneita tai konsoleja. Nopan heitto ja korttipelit ovat muiden vastaavien pelien lailla olleet ajanvietteenä jo muinaisista ajoista lähtien. Nykyään vastaavaa on nähtävissä kasinoissa ja kauppojen kolikkopeliautomaateissa. Nykyään tuskin löytyy enää sellaista henkilöä, joka ei olisi elämänsä aikana pelannut jotakin peliä, jossa satunnaisuutta ei tapahdu. Eric Lagel jakaa blogissaan (2010) nykypäivän pelien satunnaisuuden kahdeksi erilaiseksi satunnaisuudeksi.

3.1 Esisatunnaisuus

Lagelin (2010) esisatunnaisuudeksi nimeämä satunnaisuus tarkoittaa satunnaisuutta, joka sekoittaa pelielementtejä. Esisatunnaisuutta löytyy myös korttipeleistä, sillä pakan sekoittaminen pelin alussa on esisatunnaisuutta. Esisatunnaisuudella voidaan vaikuttaa pelin uudelleen pelattavuuteen ja ennalta arvaamattomuuteen. Suuren suosion

saavuttanut Minecraft käyttää myös hyväkseen tätä esisatunnaisuutta. Aina kun pelaaja luo uuden maailman, ei pelaaja voi tietää, millaiseen maailmaan hän voi vielä joutua. Koska Minecraftin maailma luodaan satunnaisuuden avulla, voi pelaaja löytää itsensä niin kuumalta aavikolta kuin kylmältä tundralta.

Yhä useammat pelit pyrkivät lisäämään satunnaisuuttaan hallitusti, sillä tämä lisää tuntuvasti pelin uudelleen pelattavuutta. Satunnaisuutta on kuitenkin pitämään kurissa ja antaa satunnaisuuden tapahtua vain hallitusti, sillä liika satunnaisuus voi tuhota muuten hyvän pelin. Tähän liittyen Sean Maher kirjoittaa blogissan, kuinka liiallinen satunnaisuus vaikutti hänen tekemäänsä peliin.

Maherin pelissä pelaajan täytyy taistella zombi laumaa vastaan, jotka käyvät pelaajan kimppuun synnyttyään pelikentän laidalle. Maherille syntyi kuitenkin ongelma, kun zombien syntymä aika, paikka ja tyyppi olivat kaikki satunnaisia. Ongelmaksi muodostui se, että välillä zombeista ei ollut mitään vastusta. Toiseksi ongelmaksi muodostui puolestaan se, että näiden rauhallisten jaksojen jälkeen saattoi syntyä voittamaton määrä zombeja.

Kuten Maherin ongelmasta voi huomata, voi liiallinen satunnaisuus tehdä pelistä pelaajan taidoista riippumattoman. Tällaisessa tapauksessa pelaaja voisi hyvin heittää pelin alussa kolikkoa, jossa klaavalla pelaaja häviää ja kruunalla voittaa pelin. Voin itsekin pelaajana sanoa, että tällainen peli ei olisi kovin mielenkiintoinen. Maher ratkaisi ongelman sillä, että uuden zombi lauman syntypaikka on satunnaisesti oikealla tai vasemmalla edelliseen zombi laumaan nähden.

Lagel huomauttaa blogissaan (2010), että esisatunnaisuuteen liittyy myös yleensä viivettä. Tämä viive mahdollistaa pelaajan reagoimisen satunnaiseen tapahtumaan. Tästä esimerkkinä Lagel (2010) käyttää Tetristä, missä pelaajalla on aikaa siirtää putoavaa palikkaa niin kauan, kunnes palikka osuu esteeseen. Samoin pelaajalla on mahdollista miettiä jo seuraavaa siirtoaan, sillä seuraava palikka näkyy jo ennalta pelikentän sivussa.

3.2 Jälkisatunnaisuus

Lagel nimeää blogissaan (2010) pelien toisen satunnaisuuden jälkisatunnaisuudeksi. Lagelin mukaan jälkisatunnaisuus on satunnaisuutta, joka tapahtuu pelaajan toimista riippuen. Esimerkkinä jälkisatunnaisuudesta Lagel käyttää roolipeleistä tuttua kriittistä osumaa ja räiskintäpeleille tuttua panosten hajontaa.

Roolipelejä pelanneille kriittinen osuma on todennäköisesti erittäin tuttu käsite. Kriittisellä osumalla tarkoitetaan osumaa, jonka pelaaja tai muu hahmo voi saada taistelun aikana. Kriittiset osumat aiheuttavat noin kaksi kertaa niin paljon vahinkoa kuin hahmon tavalliset osumat.

Räiskintäpelejä pelanneille panosten hajonta saattaa puolestaan tuoda ikäviä muistoja mieleen. Vaikka ampuessa tähtäin saattaisi olla täydellisesti kohteen kohdalla, voi pelaaja silti ampua kohteesta ohi. Tämä johtuu panosten hajoamisesta tai leviämisestä. Siinä missä oikealla kiväärillä ohi ampuminen on täysin normaalia, ottaen huomioon kaikki osumiseen vaikuttavat tekijät, on osuminen kuitenkin ampujan taidoista kiinni. Räiskintäpeleissä tämän taidon saattaa kumota satunnainen panosten hajoaminen.

Siinä missä pelaajan on mahdollista reagoida esisatunnaisuuden tapahtumiin, ei ole mahdollista jälkisatunnaisuudessa. Jälkisatunnaisuudessa pelaaja ei voi vaikuttaa satunnaisiin tapahtumiin millään tavalla. Vaikka Lagel arvosteleekin jälkisatunnaisuutta blogissaan (2010), luo se pelaajalle oman jännityksentunteensa tämän toivoessa kriittistä osumaa, jolla vihollinen saataisiin helposti kaadettua.

4 TIETOKONEELLA LUOTU SATUNNAISUUS

Satunnaisen numeron luominen tietokoneella on hankalaa. Asiaan perehtymätön voisi alkaa väittämään, että tietokoneella voi luoda satunaisia numeroita. Tämäkin väittämä on sinänsä tosi, sillä tietokoneella voidaan kirjoittaa komentoja, kuten **int rand()** -komennolla, joka luo satunnaisen numeron. Vaikka komennolla luotu satunnainen numero saattaa aluksi vaikuttaa satunnaiselta, ei tämä numero kuitenkaan ole satunnainen.

Tietokone on täysin looginen, eikä se tee mitään sattumalta. Kaikki tietokoneen satunnaisuudet syntyvät aina jonkin kaavan pohjalta. Kaavat ovat monimutkaisia laskutoimituksia, joihin tietokoneen satunaiset numerot pohjautuvat. Tästä johtuen kaavan tuntemattomalle käyttäjälle, nämä numerot vaikuttavat satunnaisilta.

Tässä vaiheessa voitaisiin kysyä, eikö tämä tietokoneella luotu satunnaisuus sitten ole tarpeeksi hyvä ja tarvitseeko sen olla todella satunnainen. Toinen hyvä kysymys voisi olla, että mihin tietokoneella luotua satunnaisuutta edes käytetään. Voitaisiin siis kysyä, että tarvitseeko tietokoneella edes luoda todellisia satunnaisia numeroita.

Tietokoneella luotuja satunaisia numeroita käytetään useissa eri asioissa. Tietokoneella luotuja satunaisia numeroita käytetään muun muassa erilaisissa salausmenetelmissä, erilaisten ilmiöiden mallintamisessa ja simuloinnissa, sekä satunnaisen näytteen valitsemisessa suurista tietomääristä. Ilman todellista satunnaisuutta olisivat nämä tapahtumat täysin pääteltävissä ja arvattavissa, eikä saataisi haluttua tulosta. Esimerkiksi, jos salausmenetelmissä käytettäisiin vain tietokoneen omaa satunnaisuutta, olisi salaus helposti purettavissa. Tästä syystä on kehitetty erilaisia keinoja, millä tietokoneella saadaan luotua monimutkaisempia satunnaisuuksia ja jopa todellista satunnaisuutta.

4.1 Pseudosatunnaisuus

Todellisen satunnaisuuden luominen ei tietokoneella ole kuitenkaan helppoa ja siksi usein turvaudutaan pelkkään pseudosatunnaisuuteen, jota tietokoneen luoma satunnaisuus tavallisesti on. Tietokoneen luoma pseudosatunnaisuus pohjautuu monimutkaiseen matemaattiseen lausekkeeseen. Howstuffworks-internet-sivuston mukaan (1998-2013) hyvän pseudosatunnaisen numeron luomiseen käytettävän matemaattisen lausekkeen tulisi olla toistumaton, hyvin numeerisesti jakautuva ja arvaamaton.

Toistumattomuudella tarkoitetaan sitä, ettei lauseke saa lähteä toistamaan itseään luodessaan satunaista numeroa. Hyvällä numeerisella jakautuvuudella tarkoitetaan sitä, että jos satunnaista numeroa haetaan väliltä 0–9, pitäisi jokaista numeroa tulla pitkällä yrittämisellä melko saman verran. Arvaamattomuudella tarkoitetaan sitä, että satunnaista numeroa ei voida päätellä ilman, että tunnetaan lauseke ja kantaluku.

Tietokone ei kuitenkaan pysty luomaan pseudosatunnaista numeroa pelkällä lausekkeella, vaan lisäksi tarvitaan kantaluku. Kantaluku on numero, johon lausekkeella vaikutetaan, jotta saadaan pseudosatunainen numero. Normaalisti tämä kantaluku on tietokoneella valmiiksi annettuna. Tietokoneen luoma pseudosatunnaisuus on huonoa siksi, että jos tunnetaan sekä kantaluku että lauseke, voidaan toisella tietokoneella luoda täsmälleen samat pseudosatunaiset luvut. Tämä tekee pseudosatunnaisuudesta huonon vaihtoehdon esimerkiksi salausmenetelmien käytössä.

Howstuffworks-internet-sivusto (1998–2013) esittelee Kerninghanin ja Ritchien (1988) kirjassaan julkaiseman yksinkertaisen lausekkeen, jolla tietokone laskee pseudosatunnaisia lukuja. Kuvasta 3 näkyy , kuinka kantaluku kerrotaan ensin 1103515245 ja lukuun lisätään 12345. Tämän jälkeen kantaluku korvataan uudella luvulla.

```
int rand() {  random_seed = random_seed * 1103515245 +12345;
return (unsigned int)(random_seed / 65536) % 32768; }
```

KUVA 3. Pseudo-satunnaisen luvun laskukaava

Kuvan 3 lauseke antaa kuitenkin tulokseksi vain pseudosatunnaisia lukuja. Howstuffworks-sivusto (1998–2013) kertoo, että useat ohjelmat käyttävät kantalukuna senhetkistä aikaa ja päiväystä muutettuna integer -muotoon saadakseen sattumanvaraisemman kantaluvun.

4.2 Todellinen satunnaisuus

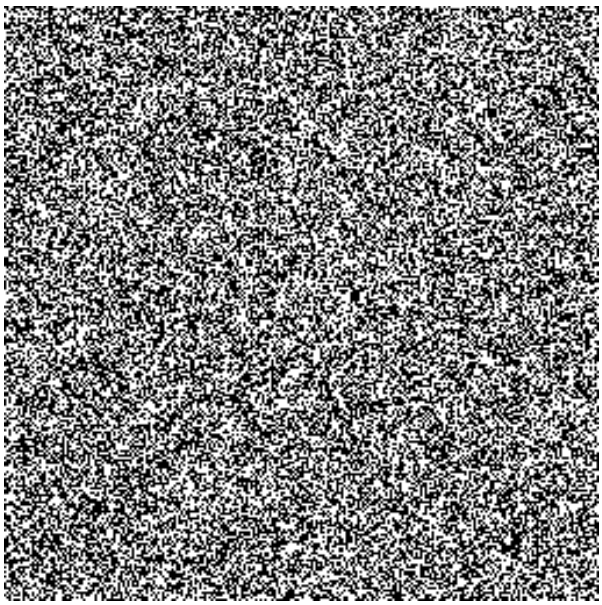
Kuinka sitten tietokoneella saadaan todellisia satunnaisia lukuja? Aikaisemmin mainitsemani nopanheitto olisi muuten hyvä vaihtoehto, mutta tietokoneiden täytyy joskus käyttää suuriakin määriä satunnaisia lukuja. Lisäksi kukapa tahtoisi pienen miehen heittelemään noppaa koneensa viereen aina kun tarvitsee käyttää satunaislukuja. Nopan heittämisessä on toinenkin iso ongelma, sillä joskus tietokoneet joutuvat hakemaan satunnaisluvun suurista joukoista. Olisihan se tietenkin näky, kun se pieni mies yrittäisi heittää esimerkiksi 10000 lukuista noppaa.

Noppa ei todellakaan ole ainoa konsti saada tietokone tuottamaan todella satunnaisia lukuja. Howstuffworks-sivustolla (1998–2013) annetaan esimerkiksi Geigermittari.

Geigermittari yhdistettäisiin tietokoneeseen ja mittarin eteen asetettaisiin radioaktiivista ainetta. Radioaktiivisen aineen hajoamisnopeus on satunnaista, joten geigermittarin antamaa lukua voitaisiin käyttää kantalukuna tietokoneen laskiessa satunnaislukua.

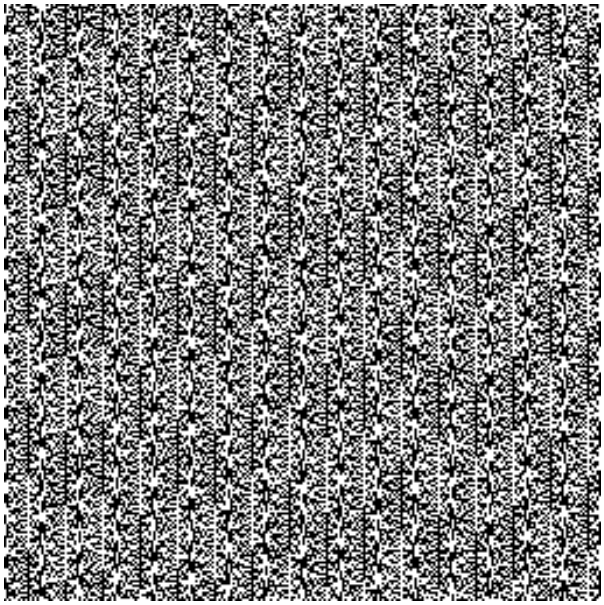
Random.org sivusto (1998–2012) käyttää puolestaan ilmakehässä tapahtuvaan kohinaa ja sen avulla tuotettuja kantalukuja satunnaisuudessaan. Tämän kohinan avulla sivusto pyrkii luomaan todellisia satunnaisia lukuja, joita voidaan käyttää esimerkiksi salausmenetelmissä. Sivustolla mainitaan myös se, että ihminen on todella hyvä hahmottamaan kuvioita. Tästä syystä ihmisen on todella helppo havaita toistuvia kuvioita, kun satunnaisuus on visualisoitu käyttämällä pisteytystä. Pisteytyksellä tarkoitan tapaa, jolla satunnaiset luvut voidaan merkitä kaksiulotteiseen taulukkoon. Tästä taulukosta ihmisen on helppo havaita, onko pisteiden numerot todella satunnaisia. Taulukossa on kuitenkin oltava tarpeeksi pisteitä, jotta voitaisiin silmämääräisesti sanoa, onko taulukon pisteet luotu todella satunnaisesti.

Bo Allen julkaisi sivustollaan (2012) suorittamansa kokeen, jossa Allen testasi random.org sivuston (1998–2012) ja Windowsin PHP:n rand()-funktion tuottamien satunnaislukujen eroja. Allen käytti apunaan bittikartta generaattoria, jotta tuloksia on helpompi verrata toisiinsa. Kuva 4 on luotu käyttäen random.org-sivuston todellisen satunnaisluvun generaattoria.



KUVA 4. Random.org sivuston luoma bittikarttakuva satunnaisesti luoduista pisteistä

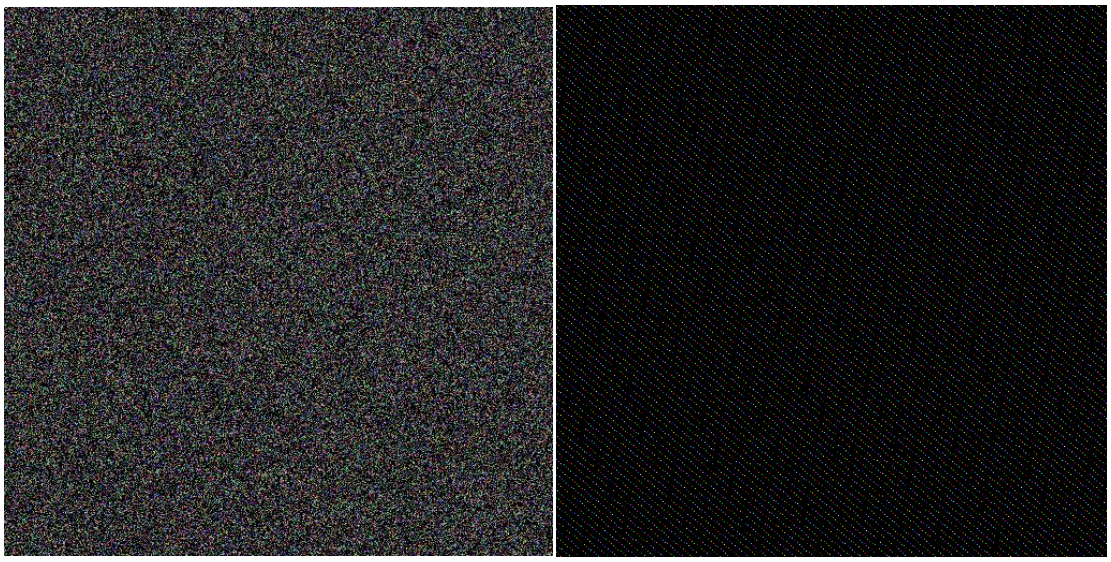
Random.org-sivusto (1998–2012) käyttää hyväkseen ilmakehässä tapahtuvaa kohinaa luodessaan satunnaislukuja. Kuten kuvasta 4 on nähtävissä, ei tällä tavalla synny selkeästi havaittavia kuvioita. Kuva 5 on puolestaan luotu käyttäen Windowsin PHP:n rand()-funktiota.



KUVA 5. Windowsin PHP:n rand()-funktion luomien satunnaislukujen bittikarttakuva

Kuten kuvasta 5 on havaittavissa, löytyy kuvasta jo selkeitä kuvioita. Kuvasta 5 on selvästi huomattavissa, kuinka pisteet ovat jakautuneet eräänlaisiksi pystypalkeiksi ja hieman heikommin havaittavissa oleviksi aalloiksi. Kuvasta 3 on myös selvästi havaittavissa melko tasaisesti esiintyviä ryhmittymiä. Allen kuitenkin huomauttaa sivustollaan (2012), että näin selvästi havaittavaa kuviointia ei synny PHP:n rand()-funktiolla, jos käytetään esimerkiksi Linuxia.

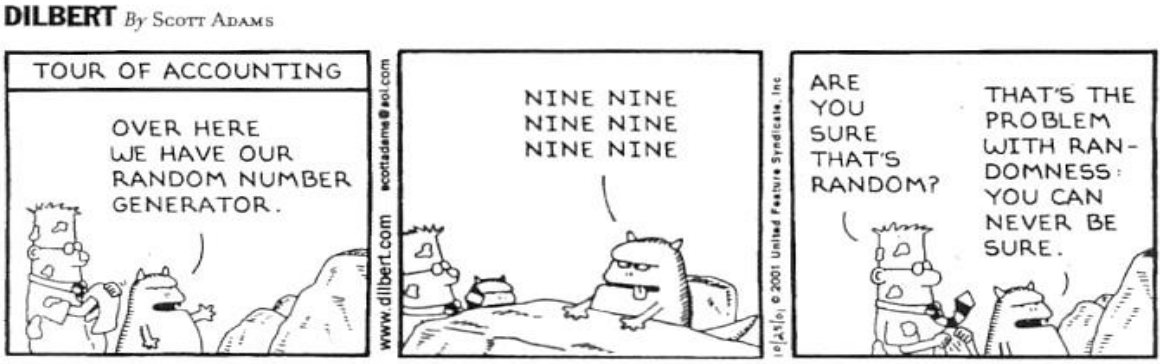
Allen testasi samaisessa kokeessa myös Windowsin PHP:n mt_rand()-funktiota. Funktio käyttää hyväkseen Mersenne Twisteriä, eikä Allenin mukaan funktiota käytettäessä synny yhtä selvästi havaittavaa kuviointia, kuin kuvassa 5. Kuvassa 6 näkyy, kuinka Mersenne Twisteriä käyttävän funktion tuottama bittikarttakuva eroaa tavallisesta rand()-funktiosta.



KUVA 6. Vasemmalla bittikarttakuva satunnaisluvuista käyttäen Mersenne Twisteriä ja oikealla bittikarttakuva käyttäen PHP:n rand()-funktiota

Kuten kuvasta 6 on huomattavissa, on rand()- ja mt_rand()-funktioiden tuottamat bittikarttakuvissa huomattava ero. Kuvan 6 oikeanpuoleisessa kuvassa on selvästi huomattavissa kuviointi ja se, että pisteet ovat jakautuneet luonnottoman tasaisesti. Vasemmanpuoleisessa kuvassa taas ei tällaista kuviointia ole syntynyt.

Mikä sitten on tämä Mersenne Twister? Mersenne Twister on Makoto Matsumoton ja Takuji Nishimuran luoma pseudosatunnaisluku generaattori, jonka he kehittivät vuosina 1996–1997. Vaikka Mersenne Twister on pseudosatunnaisluku generaattori, ei sen ja random.org sivuston (1998–2012) tuottamassa todellisessa satunnaisluku generaattorin bittikarttakuvassa ole juuri huomattavia kuvioita. Tästä voimme huomata, että vaikka satunnaisluvut olisi tuotettu pseudosatunnaisluku generaattorilla, voivat sen tuottamat luvut olla silti melko satunnaisia. Kuvassa 7 on Scott Adamsin piirtämä sarjakuva liittyen aiheeseen.



KUVA 7. Scott Adamsin sarjakuva Dilbert aiheesta satunnaisuus

Sarjakuvassa vieraillaan satunnaisluku generaattorin luona. Satunnaisluku generaattori hokee kuitenkin koko ajan numeroa yhdeksän. Kysyttäessä, onko numero varmasti satunnainen, vastataan satunnaisuuden ongelman piilevän juuri siinä. Ei siis koskaan voida todella tietää, onko satunnaisluku generaattorin tuottama luku satunnainen vai ei.

5 SATUNNAISUUS VS. MIELIVALTAISUUS

Satunnaisuuden ja mielivaltaisuuden eroista kysyttäessä ihmiset vastaavat yleensä niiden tarkoittavan samaa. Tämä ei kuitenkaan pidä paikkaansa. Vaikka satunnaisuuden ja mielivaltaisuuden voisi aluksi luulla tarkoittavan samaa, on nämä kuitenkin kaksi eri asiaa. Sanakirjasta katsottaessa mielivaltaisuutta kuvataan päähänpistoksi taikka mielijohteeksi johon ei vaikuta tarve, järki eikä laki. Satunnaisuudella taas kuvataan tapahtumaa, jolla ei ole kaavaa, tarkoitusta taikka tavoitetta.

John Debs pyrkii artikkelissaan (2008) selventämään satunnaisuuden ja mielivaltaisuuden eroja seuraavanlaisella esimerkillä. Ihmisiltä kysyttäessä numeroa yhden ja kymmenen välillä, on vastaus yleensä kolme tai seitsemän. Luku saattaa vaikuttaa sattumanvaraiselta, mutta on juuri kaikkea muuta kuin sitä. Syy, miksi juuri näitä luku valitaan paljon, on hyvin yksinkertainen. Luvut kolme ja seitsemän ovat lukuja, jotka eivät ole ääripäitä, eivätkä keskellä. Tästä johtuu, että juuri näitä lukuja saadaan eniten, kun ihmisiltä kysytään valitsemaan luku yhden ja kymmenen välillä. Jotta valittu luku olisi todella satunnainen, ei edellä mainituilla seikoilla pitäisi olla merkitystä.

Käyttäjä TF kuivaili mielivaltaisuuden ja satunnaisuuden eroja palstallaan (2010) käyttäen apunaan kolikonheittoa. Kolikkoa heitettäessä mahdollisuus saada kruuna on 50% ja klaava 50%. Kun kolikkoa heitetään, on mahdollisuus saada vaikka 100 klaavaa peräkkäin. Tapahtuman todennäköisyys on todella pieni, mutta silti mahdollinen. Voidaan siis sanoa, että saatiin 100 klaavaa peräkkäin sattumalta. Mielivaltaisesti kolikkoa heitettäessä, ei millään kolikon tuloksella ole mitään väliä, joten on turhaa seurata, onko heiton tulos kruuna vaiko klaava.

Debs tiivistää artikkelissaan (2008) mielivaltaisuuden tarkoittavan päätöksentekoprosessia, jossa päätöksen lopputuloksella ei ole merkitystä tai se on tehty päähänpistona. Satunnaisuuden Debs (2008) puolestaan tiivistää tarkoittamaan päätöksentekoprosessia, jossa jokaiselle lopputulokselle on yhtä suuri todennäköisyys toteutua.

6 CASE: STRATOGEAR

Päätin havainnollistaa selittämääni satunnaisuutta tietokonepelissä. Peli ei tosin ole ainoastaan minun tekemäni, vaan minun lisäkseni peliä oli suunnittelemassa kolme muuta opiskelijaa. Pelin käsikirjoituksesta vastasi Tero Parkkinen, mallintamisesta Heikki Mäki, graafisesta puolesta Sami Vitikainen ja koodauksesta minun lisäkseni oli vastuussa Aku Tanttu.

6.1 Ideasta eteenpäin

Pelin objektien mallintamisessa on käytetty Autodeskin 3ds Max -ohjelman opiskelija versiota. Projektia aloittaessamme mietimme pitkään, mitä ohjelmaa käyttäisimme pelin koodaamiseen. Vaihtoehtoina oli Visual Studio tai Unity 3d. Asiaa pitkään pohdittuamme ja opinnäyteohjaajaamme konsultoituamme, päädyimme Unity 3d: n ilmaisversioon.

Peliprojektin tarkoituksena on siis luoda demo pelistä, jonka olimme yhdessä suunnitelleet. Peli sijoittuu steampunk maailmaan, jossa pelaaja pelaa ilmalaivan kapteenina. Pelaaja voi lyhyen opetusosuuden jälkeen liikkua melko vapaasti pelimaailmassa. Pelaaja voi halutessaan ryhtyä kauppiaaksi, joka ostaa ja myy tuotteita kaupunkien kaupoista. Kauppojen tuotteet ovat satunnaisia ja tuotteiden hinnat riippuvat siitä, kuinka

paljon kyseistä tuotetta löytyy pelimaailmasta. Kaupungit käyttävät ja tuottavat joitain tuotteita, joten tuotteiden hinnat vaihtuvat jatkuvasti. Kauppiaana toimiessaan pelaajalla on tietenkin vaara joutua ryövärien kohteeksi.

Tahtoessaan pelaaja voi myös ryhtyä itse ryöväriksi ja ryövätä toisia laivoja saaden hyökkäystavasta riippuen ryövätyn laivan rahtia. Pelaaja voi ryövätä toisia laivoja joko tulittamalla laivan toimintakyvyttömäksi tykeillä, tai vaihtoehtoisesti yrittää miehittää aluksen saaden enemmän rahtia. Toisen laivan miehittäminen on kuitenkin vaikeampaa ja mahdollisuus menettää oman laivan miehistöä, on suuri. Ryövärinä toimiessa riesana ovat myös vartioston laivat, jotka pyrkivät puolustamaan kauppalaivoja.

Jos kumpikaan edellisistä vaihtoehdoista ei tunnu pelaajalle sopivalta uralta, voi tämä ryhtyä vartioston jäseneksi. Vartioston tehtävä on metsästää ryöväreitä ja puolustaa kauppalaivoja. Vartiostolaisena pelaajan on oltava hyvin varustautunut, sillä ryövärit ovat ahkerasti kauppalaivojen kimpussa.

Ensimmäisen ilmalaivansa lisäksi pelaaja voi ostaa kaupunkien varustamoista itselleen uusia ilmalaivoja. Ilmalaivoja on kolme erilaista: tavallinen ilmalaiva, jossa ei ole mitään erikoista, rahtilaiva, jossa on paljon ruumatilaa ja raskaampi taistelulaiva, johon mahtuu enemmän aseistusta. Jotta pelaaja voi ryövätä toisia ilmalaivoja tai puolustautua ryöväreiltä, on peliin suunniteltu useita erilaisia aseita, joilla pelaaja voi varustaa ilmalaivojaan. Tavallisten tykkien lisäksi pelaajan on mahdollista ostaa itselleen erilaisia tykkejä. Pelaaja voi pelitavastaan ja urastaan riippuen ostaa itselleen sopivia ilmalaivoja ja varustaa niitä pelitapaansa sopivilla aseilla.

Taistelun alkaessa pelaaja voi yrittää paeta taistelua lentämällä lähimpään kaupunkiin turvaan käyttäen apunaan savupommia, yhtä pelin useista aseista. Pelaaja voi taistelussa ampua vastustajan laivaa tykeillä, tai miehittää vastustajan laivan. Tykeillä ampuessaan pelaajan on otettava huomioon ilmalaivojen etäisyys toisistaan. Etäisyys vaikuttaa tykkien osumiseen ja osumisessa käytetään hyväksi satunnaisuutta. Tykkien osuessa lasketaan paljonko tykit ovat aiheuttaneet vahinkoa vastustajan ilmalaivalle. Tykkien aiheuttama vahinko ei ole vakio, vaan tykin vahinko katsotaan satunnaisuudella tykin mahdollisesta vahingosta, esimerkiksi välillä 5–10.

Miehitys tilanteessa otetaan huomioon molempien ilmalaivojen miehistön määrä. Miehitys tilanteessa lopputulokseen vaikuttaa myös, onko kummallakaan ilmalaivalla miehitystä parantavia tykkejä. Miehistön määrä ja miehityksessä käytettävät tykit lasketaan, jotta saadaan kokonaismiehitysvoima. Tätä miehitysvoimaa verrataan toisen laivan miehitysvoimaan, josta saadaan lopullinen miehityksen onnistumisen todennäköisyys.

Pelaajan ilmalaivan ottaessa vahinkoa tai aiheuttaessa vahinkoa alkaa ilmalaivan kesto laskea. Ilmalaivan kesto on aluksi vihreä, mutta keston laskiessa väri muuttuu keltaiseksi. Kun ilmalaiva on ottanut niin paljon vahinkoa, että ilmalaivan kesto on enää 20% laivan alkuperäisestä kestosta, muuttuu keston väri punaiseksi. Tässä vaiheessa pelaajan on vielä mahdollista matkustaa kaupunkiin korjaamaan laivansa.

Pelin edetessä pelaaja tulee todennäköisesti menettämään miehistöään. Koska miehistön määrä vaikuttaa suoraan miehitystilanteessa, on pelaajan pidettävä varansa, ettei joudu miehitystilanteeseen vajaalla miehistöllä, menettäen siten oman laivansa miehityksessä. Tästä syystä pelaajan kannattaa käydä aina silloin tällöin palkkaamassa lisää miehistöä kaupunkien tavernoista. Tavernassa vieraillessaan kannattaa pelaajan kysellä myös paikallisia huhuja. Huhujen avulla pelaaja saattaa saada tietoonsa mitkä tuotteet ovat alueella tarpeessa ja matkata kyseiseen kaupunkiin myymään tätä tuotetta kalliilla.

6.2 Kaupan satunnaisuus

Ensimmäinen koodaamani satunnaisuus peliin oli kauppojen tuotteiden satunnaisuus. Kauppojen tuotteiden pitää olla satunnaiset, jotta kaikissa kaupoissa ei ole samat tuotteet. Tällöin pelaajan täytyy etsiä kauppoja, joista hän voi ostaa haluamaansa tuotetta. Kauppojen tuotteet voivat olla vähissä tai tuotteita voi olla enemmän kuin kysyntää.

Pelissä pätee kysynnän ja tarjonnan laki, eli jos jotain tuotetta on kaupassa paljon, niin kyseistä tuotetta ostetaan normaalia halvemmalla. Tämä tosin tarkoittaa myös sitä, että kyseinen tuote on ostettavissa halvalla ja sen voi käydä myymässä kauppaan, jossa kyseisestä tuotteesta on pulaa. Esimerkiksi kaupassa A on paljon tuotetta X ja kaupassa B ei ole yhtään tuotetta X. Pelaaja voi tällöin käydä ostamassa halvalla tuotetta X

kaupasta A ja matkata kauppaan B. Koska kaupan B tuote X on lopussa, saa tuotteesta X suuremman hinnan, kuin normaalisti. Kuvalla 8 pyrin selventämään tätä kysynnän ja tarjonnan lakia.

KUVA 8. Stratogearin kysynnän ja tarjonnan laki

Kuvassa 8 ostin ensimmäisestä kaupungista 10 tynnyriä rommia. Tämän jälkeen siirryin toiseen kaupunkiin ja myin kaiken rommini sinne. Kuten kuvasta 8 on nähtävissä, muuttui rommin hinta kyseisessä kaupungissa pienemmäksi. Tämä johtuu siitä, että kaupungissa 2 on nyt enemmän rommia kuin siellä alunperin oli. Koska rommia on nyt kaupungissa 2 enemmän kuin sille on kysyntää, laskee rommin hinta sen mukaan.

Tuotteet kauppoihin valitaan satunnaisesti tuotelistasta, josta löytyvät kaikki pelin tuotteet. Pelin alkaessa kaupoille valikoituu satunnaisesti tuotteet siten, että kun kaupan ensimmäinen tuote on saatu, poistetaan se tilapäisesti listasta. Tämän jälkeen valikoituu kaupan toinen tuote jäljellä olevista tuotteista, jonka jälkeen tuote poistetaan taas väliaikaisesti listasta. Tämä toimenpide toistuu niin kauan, että kaupan kaikki tuotteet on valittu. Poistetut tuotteet lisätään tämän jälkeen takaisin listaan. Tuotteet poistetaan listasta, jotta kaupan tuotteisiin ei syntyisi toistoa. Esimerkiksi kaupan tuotevalikoimasta ei löydy kahdesti ruutia. Tuotteet lisätään takaisin tuotelistaan, jotta kauppojen tuotteiden vaihtuessa olisi edelliset tuotteet edelleen mahdollisia.

6.3 Tuotemäärien satunnaisuus

Kauppojen tuotemäärät ovat myös satunnaistettu. Tuotemäärien satunnaistamisella pelaajalle annetaan kuva kaupunkien jatkuvasta kaupankäynnistä ja tuotteiden määrien vaihteluista, vaikka pelin alussa ei vielä mitään olisi kaupattu. Kuvasta 9 on nähtävissä koodi, jolla edellä mainittu tuotemäärien satunnaisuus tapahtuu.

```
for(int i = 0; i < 13; i++)
{
    tuoteMaara[i] = Random.Range(40,60);
}// for
```

KUVA 9. Kaupunkien tuotemäärien satunnaisuus

Yllä olevan kuvan koodilla kukin kaupunki saa jokaista tuotetta satunnaisen määrän väliltä 40–60. Antamalla kaikille tuotteille pelin alussa satunnaisen lukumäärän on mahdollista vaihtaa kauppojen tuotteita pelin aikana. Tällöin pelin tuotteille ei tarvitse arpoa määriä pelin ollessa käynnissä.

Kaupunkien tuotteiden määrä voi muuttua vielä pelin aikana. Tämä johtuu siitä, että kaupungit myyvät, kuluttavat ja ostavat tuotteita kauppoihinsa. Jokainen kaupunki menettää yhtä tuotetta ja saa toista tuotetta satunnaisesti 10–20 sekunnin välein. Nämä poistot ja lisäykset vaikuttavat ainoastaan kaupoissa oleviin tuotteisiin. Kuvassa 10 nähdään koodi, jolla kyseinen poisto ja lisäys suoritetaan.

```
int lisaantyvaTuote = Random.Range(1,5);
int vahenevaTuote = Random.Range(1,5);

// Lisätään satunnaista tuotetta
if (lisaantyvaTuote == 1) {
    tuoteMaara[tuote1Nro] += Random.Range(10,20);
}
...
// Poistetaan satunnaista tuotetta ja varmistetaan, ettei
// tuotemäärä putoa alle nollan
if (vahenevaTuote == 1) {
    if (tuoteMaara[tuote1Nro] < 0) {
        tuoteMaara[tuote1Nro] = 0;
    }
    else {
        tuoteMaara[tuote1Nro] -= Random.Range(10,20);
    }
}
...
```

KUVA 10. Tuotemäärien lisäys ja vähennys pelin aikana

Kuvassa 10 nähtävässä koodissa valitaan ensin satunnaisesti lisääntyvä tuote ja satunnaisesti vähenevä tuote. Tämän jälkeen lisätään satunnaisesti valittua tuotetta 10–20 kappaletta. Kuvassa 10 esiintyvillä kolmella peräkkäisellä pisteellä kuvaan sitä, että edellä olevaa koodia toistetaan kasvattamalla ainoastaan **lisaantyvaTuote**:ssa ja **tuote1Nro**:ssa esiintyvää numeroa yhdellä. Tämä pätee myös tuotteen vähentymiseen vaikuttavan koodin jälkeen esiintyvään kolmeen peräkkäiseen pisteeseen. Tuotteen vähentämisessä on tosin otettava huomioon se, että tuotemäärä ei saa pudota nollan alapuolelle. Tästä syystä tarkistetaan meneekö tuotemäärä nollan alapuolelle ja jos näin tapahtuu muutetaan tuotemäärä nollaksi.

6.4 Taistelun satunnaisuus

Kauppojen ja tuotemäärien satunnaisuuksien jälkeen oli hyvin tärkeää, että pelaaja pystyy ampumaan ostamillaan tykeillä taistelutilanteessa. Taistelu on oleellinen osa peliä, joten pelaajan on varauduttava näitä tilanteita varten ostamalla tykkejä ilmalaivoihinsa. Taistelu tilanteessa pelaajalla on kolme eri vaihtoehtoa. Pelaaja voi paeta taistelusta ja lentää lähimpään kaupunkiin turvaan.

Jatkuva pakoon juokseminen ei kuitenkaan ole kannattavaa, vaan pelaajan on pyrittävä ostamaan ilmalaivaansa tarpeeksi miehistöä ja tykkejä puolustusta varten. Pelaajalla on kaksi vaihtoehtoa hyökätessään toisen ilmalaivan kimppuun. Pelaaja voi joko yrittää miehittää kohteen tai tulittaa tämän alas tykeillä. Molemmissa vaihtoehdoissa on kuitenkin omat hyvät, että huonot puolensa.

Tykkitaistelussa pelaaja käyttää ostamiaan pitkän kantaman tykkejä, joilla pyritään pudottamaan kohteen kesto nollaan. Tykkitaistelussa jokaiselle laukaukselle katsotaan, onko laukaus osunut kohteeseen. Tämä tapahtuu satunnaislukuja apuna käyttäen. Jokaiselle laukaukselle annetaan luku väliltä 1–20. Tätä lukua verrataan kohteen panssariin, vahingon sietokykyyn ja väistämiskykyyn, mutta käyttäkäämme panssaria kuvaamaan näitä ilmalaivan ominaisuuksia välttyä vahingolta. Jokaisella ilmalaivalla on oma panssarinsa riippuen ilmalaivan tyypistä.

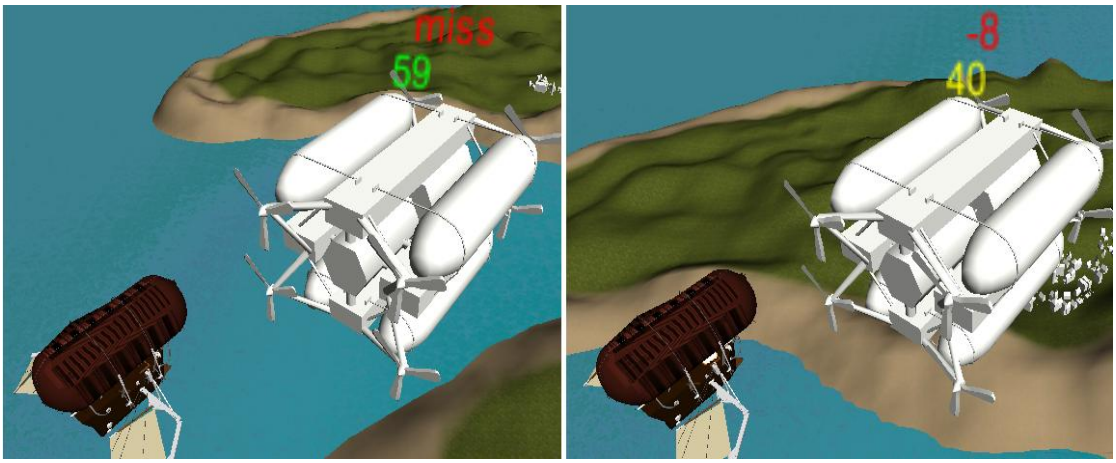
Tavallisella ilmalaivalla ja rahtilaivalla tämä panssari on viisi, mutta raskaalla ilmalaivalla panssari on 8. Kun laukauksen satunnaislukua verrataan kohteen panssariin, on

laukaus osuma, jos satunnaisluku on suurempi tai yhtä suuri kuin kohteen panssari. Osumatilanteessa kohde kärsii vahinkoa riippuen tykistä, jolla osuma on saatu. Tavallisella tykillä tämä vahinko on väliltä 5–10. Jokaisella laukauksella pelaajalle näytetään punaisella tekstillä kohteen päällä, onko laukaus mennyt ohi, tai osuessa kuinka paljon vahinkoa kyseinen laukaus aiheutti. Kohteen keston laskiessa nollaan, on kohde tuhottu. Kohteen tuhoutuessa pelaaja saa pienen osan kohteen lastista ja sen hetkisestä rahamäärästä palkkiona. Kuvasta 11 läpikäyn vielä askel kerrallaan, kuinka ampumatilanteessa määritellään osuminen ja vahingon määrä.

```
void Shooting(){
    // Osuuko laiva kohteeseen
    accuracy = Random.Range(1,20);
    // Jos laiva osuu kohteeseen, vähennetään kohteelta kestoa
    if (accuracy >= armor){
        damage = Random.Range(5,10);
        health -= damage;
```

KUVA 11. Osumisen ja vahingon määrittäminen koodilla

Kuvassa 11 tarkastellaan tapahtumaa Shooting eli ampuminen. Ampumatapahtumassa määritellään ensin ilmalaivan accuracy eli tarkkuus. Tämän jälkeen verrataan tätä tarkkuutta kohteen armoriin eli panssariin. Jos ampujan tarkkuus on suurempi kuin kohteen panssari, on laukaus osuma. Osuman määrittämisen jälkeen tarkastetaan damage eli vahinko, jonka laukaus aiheuttaa. Kun vahingon määrä on saatu selville, vähennetään se kohteen kestosta. Alla olevassa kuvassa 12 nähdään kyseinen tapahtuma pelaajan näkökulmasta.



KUVA 12. Taistelutilanne pelaajan näkökulmasta

Kohteen tuhoaminen tykeillä on kuitenkin pitkäkestoinen urakka, jossa vaarana on kohteen pakeneminen kaupunkiin. Tästä syystä pelaajan on myös mahdollista yrittää vallata kohde miehittämällä. Miehitys tilanne on ratkaistu nopeammin kuin tykkitaistelu, mutta valtaaminen on huomattavasti vaikeampaa. Kohde ilmalaivan valtaamisessa on myös se hyvä puoli, että tällöin on mahdollista saada vallattu laiva pelaajan omaan käyttöön. Tässä tilanteessa palaaja saa itselleen kohteen rahdin ja rahavarat.

Toisen ilmalaivan miehittäminen ei kuitenkaan ole helppoa, sillä se vaatii tavallisesti suuremman miehistön kuin kohteella. Miehitystilanteessa tarkistetaan molempien laivojen miehistön määrä, sekä tarkistetaan onko kummassakaan laivassa miehitykseen vaikuttavia aseita. Peliin on suunniteltu useita miehitykseen vaikuttavia aseita. Näitä aseita käyttämällä on toisten ilmalaivojen miehittäminen helpompaa. Kun molempien ilmalaivojen miehistön määrä ja ilmalaivojen aseistus on saatu selville, lasketaan näiden avulla ilmalaivoille miehityksessä käytettävä luku. Näitä lukuja verratessa saadaan miehityksen onnistumisen todennäköisyys.

Esimerkiksi pelaajalla on hallussaan tavallinen ilmalaiva, jossa on 12 miehistön jäsentä. Tämän lisäksi pelaajalla on ostettu ilmalaivaansa miehityksessä vaikuttava scatter gun, joka ampuu rautaromua kohteen miehistöä kohden. Kohteella on myös tavallinen ilmalaiva, jossa on täysi 12:sta miehen miehistö. Jokaisesta miehistön jäsenestä ilmalaivat saavat yhden pisteen miehityspisteisiinsä, mutta koska pelaajalla on ostettuna scatter gun, saa pelaaja näihin pisteisiin kaksi pistettä lisää. Pelaajalla on tällöin 14 miehityspistettä ja kohteella 12 miehityspistettä. Näitä lukuja verrataan ja pelaajan pisteistä vähennetään kohteen pisteet, jolloin pelaaja voittaa kahdella pisteellä.

Tämä ei kuitenkaan tarkoita sitä, että pelaaja olisi voittanut miehityksen, vaan jokaista jäljelle jäävää pistettä kohden pelaajan miehityksen onnistumisen todennäköisyys nousee. Jokaisesta jäljelle jäävästä pisteestä pelaaja saa 10 pistettä lisää miehityksen onnistumiseen. Tässä vaiheessa annetaan satunnaisluku väliltä 1-100. Jos luku on pienempi tai yhtä suuri kuin pelaajan miehityksen onnistumisen pisteet, on pelaaja vallannut laivan. Jokaisella miehitysyrityksellä molemmista laivoista kuitenkin kuolee satunnainen määrä miehistöä, joka voi muuttaa miehityksen todella vaikeaksi.

6.5 Huhujen satunnaisuus

Pelaajan käydessä kaupungissa on hänen mahdollista kysyä huhuja tavernassa. Huhuja kysyessä pelaajan on mahdollista saada tietoa esimerkiksi kauppareiteistä, tuotepulista tai tuotetulvista. Näitä tietoja apuna käyttäen pelaajan on mahdollista ansaita suuria summia rahaa käyttämättä aikaansa kaupunkien läpikäymiseen etsiessään hyvää myynti- tai ostopaikkaa.

Pelaajan on otettava kuitenkin huomioon se, että huhut ovat vain huhuja. Tämä tarkoittaa sitä, että pelaajan saamat huhut eivät aina pidä paikkaansa. Huhun paikkaansa pitävyys vaihtelee satunnaisesti ja vaikka huhu pitäisikin paikkaansa, voi huhu olla pienellä todennäköisyydellä täysin hyödytön pelaajalle. Kuvasta 13 nähdään, kuinka huhu valikoituu satunnaisesti.

```
// Lisätään nappi jolla kysytään huhuja
if (GUI.Button(new Rect(165,(Screen.height*7)/100,70,20),"Rumors")){
    Debug.Log("Kysellään huhuja");
    huhu = Random.Range(1,6);
}
if (huhu == 1) {
    GUI.Label (new Rect (25,(Screen.height*15)/100,50,50),
    " Porkkanamajoneesi \n on hyvaa "+huhu,Tyyli);
}
if (huhu == 2) {
    GUI.Label (new Rect (25,(Screen.height*15)/100,50,50),
    " Porkkanamajoneesi \n on mahtavaa "+huhu,Tyyli);
}
if (huhu == 3) {
    GUI.Label (new Rect (25,(Screen.height*15)/100,50,50),
    " Porkkanamajoneesi \n on loistavaa "+huhu,Tyyli);
}
if (huhu == 4) {
    GUI.Label (new Rect (25,(Screen.height*15)/100,50,50),
    " Porkkanamajoneesi \n on parasta "+huhu,Tyyli);
}
if (huhu == 5) {
    GUI.Label (new Rect (25,(Screen.height*15)/100,50,50),
    " Porkkanamajoneesi \n on jumalallista "+huhu,Tyyli);
}
```

KUVA 13. Huhujen satunnaisuus

Kuvassa 13 käsitellään koodilla huhujen satunnaisuutta. Yllä olevassa kuvassa olevat huhut eivät tule varsinaiseen demoon pelistä. Kyseiset huhut lisättiin testivaiheessa, jotta huhujen toimivuus voitiin tarkistaa. Kuvasta 13 on kuitenkin jo nähtävissä kuinka

huhujen satunnaisuus toimii. Huhua kysyessä annetaan satunnainen numero, tässä tapauksessa väliltä 1–6. Kyseinen koodi toimii kuitenkin siten, että satunnaisluku ei voi olla kuin väliltä 1–5. Tästä syystä huhuja on vain viisi. Kun satunnaisluku on saatu, näytetään kyseistä lukua vastaava huhu kuten on nähtävissä alla olevasta kuvasta 14.

KUVA 14. Huhut pelaajan näkökulmasta

Kaupunkien huhut vaihtelevat myös pelaajan pelitavasta riippuen. Jos pelaaja pelaa kauppiaana, on suurempi todennäköisyys saada tietoja kauppareiteistä ja tuotteista. Vaihtoehtoisesti pelaajan pelatessa ryövärinä, on pelaajan mahdollista saada tietoja kauppalaivojen kauppareiteistä tai vartioston partioiden sijainneista. Vartiostolaisena pelaajan on mahdollista saada tietoja ryövärien liikkeistä tai mahdollisista kauppailmalaivojen puolustuksista.

Huhuja kysyessään pelaajalle arvotaan satunnaisesti huhun paikkaansa pitävyys. Huhu voi olla tällöin totta, valhetta tai pelaajalle hyödyttömiä totuuksia. Kun huhun paikkaansa pitävyys on selvitetty, annetaan pelaajalle satunnainen numero. Tämä numero muuttuu pelaajan pelitavasta riippuen, jolloin pelaajan on mahdollista saada itselleen sopivia huhuja. Huhujen idea piilee siinä, että pelaaja ei voi tietää etukäteen, onko huhu totta vai ei.

6.6 Suunnitteilla olevat satunnaisuudet

Pelin kehittämisideoita on monia, joista suurimman osan jouduimme jättämään pois pelimme demosta. Yksi näistä ideoista oli uusi ryhmittymä peliin. Tutkimusmatkailija.

Tutkimusmatkailijana pelaajan olisi mahdollista etsiä kadonneita aarteita pelimaailman saaristoista ja matkata pelimaailman ulkopuolelle etsien uusia mantereita. Kuvassa 15 graafikkomme Sami Vitikaisen piirtämä luonnos tutkimusmatkailijasta.

KUVA 15. Luonnos tutkimusmatkailijasta

Tutkimusmatkailijana pelaajan olisi mahdollista etsiä kadonneita aarteita, joista pelaajan olisi mahdollista löytää joko tuotteita, tai mahdollisia lisäosia laivaansa. Näistä lisäosista esimerkkinä erilaiset tykit. Aarteiden etsiminen ei kuitenkaan ole täysin vaaratonta. Aarteita etsiessään pelaajan on myös mahdollista törmätä erilaisiin vahingollisiin tilanteisiin. Aarteiden etsimisen vaaroina ovat: vihamielisten alkuasukkaiden hyökkäys, miehistön sairastuminen tai vaikkapa muinaisen haudan ryöstämisestä johtuva kirous.

Aarteita etsiessään pelaajalle annetaan satunnainen luku väliltä 1–100. Aarteen löytämisen vaikeudesta riippuen katsotaan, onko pelaaja onnistunut löytämään aarteen. Muussa tapauksessa pelaaja kohtaa yllämainittuja vaaroja. Vaaratilanteen vaikeus riippuu myös aarteen löytämisen todennäköisyydestä.

Pelaajan törmätessä alkuasukkaisiin, joutuu pelaajan ilmalaivan miehistö taisteluun. Taistelumekaniikka tällaisissa tilanteissa toimii todennäköisesti samalla tavalla kuin miehitystilanteessa. Koska tutkimusmatkailijaa ei ole lisätty demoon, ei lopullisesta taistelutavasta olla vielä päätetty.

Tutkimusmatkailijan toisena vaihtoehtona on matkustaa etsimään uusia mantereita. Uusien mantereiden etsimisessä käytetään samaa periaatetta, kuin aarteiden etsimisessä. Uusien mantereiden etsiminen on kuitenkin enemmän "high risk, high reward" -tyylistä. Tämä tarkoittaa siis sitä, että pelaajan on mahdollista törmätä suurempiin vaaroihin uusia mantereita etsiessään. Onnistumisen palkkiot ovat kuitenkin huomattavasti suurempia.

Pelaajan epäonnistuessa olisi pelaajan mahdollista menettää ilmalaivansa tai ilmalaivan palaaminen aavelaivana ilman miehistöä ja pahasti vaurioituneena. Onnistuessaan pelaaja voi kuitenkin luoda uusia kauppareittejä, tuoden eksoottisempia tuotteita markkinoille. Vaihtoehtoisesti pelaaja voi monopolisoida näitä tuotteita, saaden pienen rahallisen osuuden jokaisesta kyseisestä myydystä tuotteesta.

7 PÄÄTÄNTÄ

Aloittaessani tämän opinnäytetyön ei minulla ollut kuin peruskäsitys satunnaisuudesta. Työni edetessä ja asiaa tutkiessani sain aiheesta laajemman käsityksen. Näiden tietojen pohjalta pystyin soveltamaan satunnaisuutta tekemässämme pelissä.

Satunnaisuutta peliin soveltaessani jäi minua kuitenkin harmittamaan se seikka, että pelissä käytetty satunnaisuus on pseudo-satunnaisuutta. Vaikka pelaajat eivät todennäköisesti huomaa eroa todellisen satunnaisuuden ja pseudo-satunnaisuuden välillä, olisi minusta ollut mukavaa, jos pelissä käytetty satunnaisuus ollut todellista satunnaisuutta. Todellisen satunnaisuuden luominen tietokoneella ei kuitenkaan ole niin helppoa, eikä minulla ole tietotaitoa luoda todellista satunnaisuutta peliin.

Vaikka tietokoneella luotu satunnaisuus ei aina olisikaan todellista satunnaisuutta, voidaan aina pyrkiä lähemmäs tätä satunnaisuutta. Pelien satunnaisuus on parantunut

jatkuvasti, eikä pelien satunnaisuus ole mielivaltaista. Jatkuva pelien satunnaisuuden kehittäminen johtaa siihen, että jossain vaiheessa pelien satunnaisuus voi olla todellista satunnaisuutta. Tätä ennen joudumme kuitenkin tyytymään pseudo-satunnaisuuteen. Yleisesti ottaen nykyiset pseudo-satunnaisuudet ovat kuitenkin hyvin luotuja, eivätkä pelaajat yleensä huomaa satunnaisuudessa eroja. Tämä johtuu pitkälti siitä, että pseudo-satunnaisuus on saatu näyttämään todelliselta satunnaisuudelta, vaikkei se sitä olisikaan.

Aivan kuten pelien satunnaisuus kehittyy, on tarkoituksenamme jatkaa pelimme demon kehittämistä lopulliseksi peliksi. Projektin alusta lähtien meillä on ollut monia ideoita pelin jatkamiseksi, joista osa on ollut hyviä ja osa vähemmän sitä. Jokainen meistä on miettinyt pelin jatkamista omasta näkökulmastaan ja kuinka peliä voisi yleisesti kehittää.

LÄHTEET

Bo, Allen 2013. Pseudo-random vs. true random. WWW-dokumentti. http://boallen.com/random-numbers.html. Päivitetty 26.2.2012. Luettu 1.10.2013.

Cormer, Leiserson, Riverst, Stein 2001. Introduction to algorithms second edition. Massachusetts: The MIT Press.

Debs, John 2008. Arbitrary vs. random. WWW-dokumentti. http://johnthedebs.blogspot.fi/2008/01/arbitrary-vs-random.html. Päivitetty 29.1.2008. Luettu 25.10.2013.

Howstuffworks, Inc 1998-2013. How can a totally logical machine like a computer generate a random numeber?. WWW-documentti. http://computer.howstuffworks.com/question697.htm. Päivitetty 1.10.2013. Luettu 1.10.2013.

James, Mike 2012. The revolution in evolutionary game theory - Prisoners dilemma solved?. WWW-dokumentti. http://www.i-programmer.info/news/112-theory/4670-the-revolution-in-evolutionary-game-theory-prisoners-dilemma-solved.html. Päivitetty 19.8.2012. Luettu 26.10.2013.

Järvinen, Petteri 2003. Salausmenetelmät. Porvoo: Docendo.

Korhonen, Terhi 2013. Modernin kryptografian RSA-salausmenetelmä ja sen lukuteoreettinen tausta. Helsingin yliopisto. Matematiikan ja tilastotieteen laitos. Pro gradu -tutkielma.

Lagel, Eric 2010. Randomness in games... why?. WWW-dokumentti. http://www.gamasutra.com/blogs/EricLagel/20100827/87924/Randomness_in_games_why.php. Päivitetty 27.8.2010. Luettu 28.10.2013.

Lo, Chris 2012. Game theory: introducing randomness to airport security. WWW-dokumentti. http://www.airport-technology.com/features/featuregame-theory-airport-security-teamcore-stackelberg/. Päivitetty 26.7.2012. Luettu 26.10.2013.

Lutus, Paul 2010. Randomness as meaning. WWW-dokumentti. http://www.arachnoid.com/randomness/. Päivitetty 1.10.2013. Luettu 1.10.2013.

Maher, Sean. Randomness in games: Less random is mo better. WWW-dokumentti. http://deadpanic.com/randomness_in_games. Päivitetty 28.10.2013. Luettu 28.10.2013.

Mellin, Ilkka 2010. Tilastotiede tieteenä. Power Point-esitys.

Pohjavirta, Armo & Ruohonen, Keijo 2005. Laaja tilastomatematiikka. PDF-dokumentti. http://math.tut.fi/~ruohonen/LTM.pdf. Päivitetty -.-.-. Luettu 25.10.2012.

Random.Org 1998-2012. WWW-dokumentti. http://www.random.org/. Päivitetty 1.10.2013. Luettu 1.10.2013.

Staff, Edge 2013. Games of chance: What does randomness bring to videogames?. WWW-dokumentti. http://www.edge-online.com/features/games-of-chance-what-does-randomness-bring-to-videogames/. Päivitetty 5.7.2013. Luettu 28.10.2013.

TF 2010. Arbitrary vs. random. WWW-dokumentti. http://newsgroups.derkeiler.com/Archive/Alt/alt.usage.english/2010-03/msg02898.html. Päivitetty 18.3.2010. Luettu 25.10.2013.