Proseduraalisen sisällönluonnin menetelmät roguelike-peleissä

Turun yliopisto
Tietotekniikan laitos
TkK-tutkielma
Tietotekniikka
Toukokuu 2025
Kalle Sova

TURUN YLIOPISTO
Tietotekniikan laitos

Kalle Sova: Proseduraalisen sisällönluonnin menetelmät roguelike-peleissä

TkK-tutkielma, 21 s.
Tietotekniikka
Toukokuu 2025

Tämä tutkielma käsittelee proseduraalisen sisällönluonnin (engl. Procedural content generation, PCG) menetelmiä roguelike-genren peleissä, keskittyen sen vaikutukseen pelikokemukseen ja pelitasapainoon. Tutkimuksen tavoitteena on selvittää, miten PCG:ia hyödynnetään roguelike-peleissä, miten sillä luodaan tasapainoinen ja kiinnostava pelikokemus, sekä miten pelimekaniikat kuten permadeath vaikuttavat PCG-suunnitteluun. Tutkimus perustuu kirjallisuuskatsaukseen sekä tunnettujen roguelike-pelien, The Binding of Isaac ja Spelunky -perustuviin tapaustutkimuksiin.

PCG:n avulla luodaan dynaamisia ja ainutlaatuisia pelikokemuksia, kuten satunnaisesti generoituja kenttiä, esineitä ja vihollisia. Tutkimuksessa havaittiin, että vaikka The Binding of Isaac ja Spelunky käyttävät erilaisia menetelmiä, molemmat pitävät satunnaisuuden hallittuna ennalta määriteltyjen sääntöjen avulla. Esimerkiksi The Binding of Isaac hyödyntää BFS-hakuun perustuvaa algoritmia huoneiden luomiseeen, kun taas Spelunky yhdistää valmiita osioita ja satunnaisia elementtejä. Näillä menetelmillä varmistetaan, että pelikokemus on sekä haasteellinen että johdonmukainen.

Tutkimus osoitti, että PCG on keskeinen osa roguelike-genren pelejä, ja sen onnistuminen riippuu huolellisesta suunnittelusta ja testauksesta. PCG:in avulla voidaan luoda pelikokemuksia, jotka säilyttävät pelaajan kiinnostuksen ja tarjoavat uudelleenpelattavuutta. Jatkotutkimuksissa voitaisiin tarkastella generatiivisen tekoälyn hyödyntämistä PCG:ssä sekä sen mahdollisuuksia luoda dynaamisia tarinoita pelaajan valintojen perusteella.

Asiasanat: proseduraalinen sisällönluonti, roguelike, videopelit, algoritmi

Sisällys

Kuvat

Termistö

NPC Non-playable character

PCG Procedural content generation

1 Johdanto

Tämä työ käsittelee roguelike-videopeleissä hyödynnettäviä proseduraalisen sisällönluonnin (engl. *procedural content generation*, PCG) algoritmien toimintaa ja vaikutusta pelikokemukseen. Valitsin työn aiheen sen kiinnostavuuden takia. PCG-algoritmien vaikutus roguelike genren peleihin on kiinnostava, sillä pelejä pelatessa ei tule välttämättä ajatelleeksi niiden vaikutusta pelikokemukseen vaikkakin koko pelikokemus pitkälti perustuu juuri algoritmien toimivuuteen ja saumattomuuteen. Tämän lisäksi koen, että aiheesta ei ole tarpeeksi tarkempaa tutkimusta.

Työssä vastataan seuraaviin tutkimuskysymyksiin:

TK1 Miten PCG:ia hyödynnetään roguelike-peleissä?

TK2 Miten PCG:in avulla luodaan tasapainoinen pelikokemus, joka säilyttää pelaajan kiinnostuksen, mutta on silti haasteellinen?

TK3 Miten pelimekaniikat, kuten permadeath ja resurssienhallinta, vaikuttavat PCG suunnitteluun?

Nämä valikoituivat tutkimuskysymyksiksi, sillä koen, että näihin kysymyksiin ei löydy vastausta kandidaatin tutkielman tasoisesta tutkimuksesta.

Hakuprosessi aloitettiin etsimällä yleisesti Turun yliopiston kirjaston Volter-palvelusta sekä Google Scholarista "procedural content generation" sekä myös "roguelike games". Tällä hakulauseella löytyi konferenssijulkaisu *A Hybrid Approach to Procedural Generation of Roguelike Video Game Levels*. Tämä valikoitui yhdeksi

lähteistä, sillä se sisälsi tutkielmalle oleellista tietoa. Tarkensin hakulauseketta muotoon "procedural content generation" AND "algorithm". Etsintää suoritettiin myös hakusanoilla "PCG" AND "games", "roguelike algorithm" sekä yleisesti "procedural content generation". Volterista löytyi hakulauseella “procedural content generation” kirja *Procedural Content Generation in Games*, joka valikoitui toiseksi lähteeksi työhön, sillä se sisälsi paljon yleisesti hyödynnettävää tietoa proseduraalisesta sisällönluonnista videopeleissä. Termillä “roguelike” löysin artikkelin *Roguelike Games: The way we play*, joka valikoitui kolmanneksi artikkeliksi. Neljänneksi artikkeliksi valikoitu *Game Dynamics: Best Practices in Procedural and Dynamic Game Content Generation*, joka löytyi Volterista hakulauseella "Procedural Content Generation". Viidenneksi lähteeksi valikoitui *The story of Rogue* -verkkoartikkeli, sillä se sisälsi paljon hyvää tietoa Rogue-pelistä pelinkehittäjän itsensä kertomana. Lähde löytyi Googlesta hakulauseella "rogue history". Kuudes ja seitsemäs lähde löytyivät verkkopelialusta Steamista etsimällä pelien nimet "The Binding of Isaac" ja "Spelunky". Valikoin ne slähteiksi, sillä ne sisälsivät yleistä ja luotettavaa tietoa peleistä suoraan niiden kehittäjiltä. Kahdeksas lähde *Dungeon Generation in Binding of Isaac* -verkkoartikkeli, sisältää paljon pelin kehitykseen liittyvää tietoa, joten se valikoitui lähteeksi, jota hyödynnettiin ensimmäisessä tapaustutkimuksessa. Lähde löytyi Googlesta hakulauseella "binding of isaac dungeon generation". Yhdeksäs lähde on tieteellinen artikkeli *A Procedural Method for Automatic Generation of Spelunky Levels*. Artikkeli sisälsi paljon tarkkaa tietoa Spelunky pelistä, jonka vuoksi se valikoitui yhdeksi lähteistä. Artikkeli löytyi Volterista hakusanalla "Procedural Generation Spelunky". Kymmenenneksi lähteeksi valitsin teoksen *With Fate Guiding My Every Move: The Challenge of Spelunky*. Teos toi lisää tietoa tason luontiin toisessa tapaustutkimuksessa. Teos löytyi hakulauseella "challenge spelunky" Google Scholarista. Yhdenneksitoista lähteeksi valikoitui artikkeli *Player-adaptive Spelunky*

level generation, jota hyödynnettiin toisessa tapaustutkimuksessa. Artikkeli löytyi hakulauseellla "spelunky level generation" Google Scholarista.

Toisessa luvussa käsitellään proseduraalisen sisällönluonnin taustaa ja roguelike-pelien historiaa lyhyesti. Kolmannessa luvussa käsitellään proseduraalisen sisällönluonnin suunnittelua ja toimintaa. Neljänessä luvussa käsitellään tapaustutkimuksia, joissa hyödynnetään proseduraalista sisällönluontia eri tavoilla. Viidennessä luvussa muodostetaan johtopäätös aikasemmissa luvuissa vastattuihin tutkimuskysymyksiin liittyen.

2 Proseduraalisen sisällönluonnin tausta ja roguelike-pelien historia

Tämä luku esittää proseduraalisen sisällönluonnin taustan ja lyhyesti historian, jotta tutkielman seuraavat luvut ovat ymmärrettävissä. Luvun alussa esitellään proseduraalisen sisällönluonnin perusteet ja sen merkitys pelinsuunnittelussa. Sen jälkeen keskitytään roguelike-genren historiaan ja sen keskeisiin piirteisiin, kuten permadeath-mekaniikkaan ja satunnaisesti generoituihin pelimaailmoihin.

2.1 Proseduraalisen sisällönluonnin perusteet

Työssä tullaan käsittelemään termiä procedural content generation useasti, joten siitä käytetään jatkossa lyhennettä PCG työn luettavuuden helpottamiseksi. PCG tarkoittaa jatkuvaa sisällönluontia erilaisten algoritmien avulla. PCG:ia hyödynnetään videopeleissä esimerkiksi karttojen, hahmojen sekä esineiden luontiin ilman, että niitä joudutaan luomaan manuaalisesti. Manuaalista sisällönluontia saatetaan vältellä esimerkiksi pelinkehittäjien vähäisten resurssien vuoksi, jolloin PCG hyödyntäminen näyttäytyy parempana vaihtoehtona. PCG:iksi ei lasketa pelimoottorin luontia tai ei-pelattavien hahmojen (engl. *non-playable character*, NPC) käyttäytymisalgoritmeja. PCG:iksi lasketaan kuitenkin jokin metodi, jonka pelaaja voi itse manuaalisesti käynnistää, ja jonka jälkeen tietokone suorittaa automaattisesti loput. [1]

2.2 Roguelike-genren pelien perusteet sekä historia

Työssä puhutaan peleistä, jolla tässä tapauksessa tarkoitetaan videopelejä riippumatta pelialustasta. Tämän lisäksi työssä käsitellään tarkemmin videopelejä, jotka kuuluvat roguelike- tai roguelite-genreen. Roguelike-genren nimitys tulee vuonna 1980 julkaistusta pelistä *Rogue*. Roguen tekijät Glenn Wichmann ja Michael Toy loivat Roguen alunperin *Dungeons & Dragons* -roolipelin inspiroimana. Wichmannin sanojen mukaisesti *"Rogue was also a very well balanced game. It was notoriously hard to beat, but you did not have to beat it to enjoy it. It was easy to learn and understand"* [2]. Roguen keskiössä ovat siis yksinkertaiset, mutta satunnaisesti luodut kartat sekä pelimekaniikan ydin, joka keskittyy pelaajan epäonnistuessa alusta aloittamiseen [3]. Tätä uudelleen aloittamisen pelimekaanikkaa kutsutaan myös *permadeath*- mekaniikaksi, ja sitä tullaan jatkossa kutsumaan työssä tällä nimellä. Roguelike siis nimensä mukaisesti viittaa Rogue:n kaltaisiin peleihin. Ei ole kuitenkaan virallista konsesusta siitä, millaiseksi roguelike-genre määriteltäisiin, sillä pelit voivat erota paljon toisistaan. Tämän lisäksi on myös roguelite-pelejä, jotka ovat toiminnallisuudeltaan pitkälti samankaltaisia kuin roguelike-pelit, mutta sallivat osittaisen hahmon kehittämisen pelikertojen välillä täten osittain kumoten roguelike-peleille ominaisen permadeath mekaniikan. Roguelike-pelien menestys perustuu uudelleenpelaamiseen ja voiton hankaluuteen sekä siihen, että lähes jokainen pelikerta on erilainen. [2] [3] [4]

2.3 Proseduraalisuuden toiminnallisuus roguelike-genren peleissä

Roguelike-genre on pääsääntöinen osa-alue PCG:in hyödyntämisessä peleissä, sillä roguelike-pelien pohjana on uudelleenpelattavuus. PCG:in hyödyntäminen peleissä tuo kaksi keskeistä hyötyä roguelike-pelin suunnitteluun. Ensimmäiseksi aiemmin

mainitun uudelleenpelattavuuden, ja toisena PCG lisää täysin ainutlaatuisia pelikokemuksia ilman manuaalisesta syötettä pelinkehittäjiltä. Tämä luo peliin syvyyttä, josta pelaaja hyötyy. Tämän lisäksi PCG:in hyödyntäminen pystyy vähentämään pelin tekijöiden taakkaa minimoimalla sisällön "kovakoodausta" (engl. *hard coding*), jossa vastuu arvojen tai datan syöttämisestä jää pelinkehittäjälle. [3]

2.4 Tapaustutkimuksissa tarkasteltavat pelit

Työn neljännessä luvussa käsitellään tapaustutkimuksia siitä, miten PCG:ia on hyödynnetty eri tavoilla peleissä. Seuraavissa luvuissa käsitellään näiden pelien perustietoja sekä taustaa, jotta tapaustutkimuksien lukeminen olisi mahdollisimman sujuvaa.

Ensimmäinen tapaustutkimus käsittelee *The Binding of Isaac* -peliä. *The Binding of Isaac* on 28.9.2011 julkaistu Edmund McMillenin ja Florian Himsl:in kehittämä roguelike-peli, joka tunnetaan erityisesti huoneiden luonnin (engl. dungeon generation) satunnaisuudestaan. Tämä takaa, että jokainen pelikerta on täysin erilainen, mikä vaikuttanut *The Binding of Isaacin* suureen suosioon. [5]

Toisessa tapaustutkimuksessa käsitellään Mossmouthin kehittämään ja 8.8.2013 julkaisemaa *Spelunky HD* -peliä. Spelunky HD on kaksiuloitteinen roguelike-tasohyppelypeli, jossa tasot ja niiden sisällöt generoituvat satunnaisesti. [6] *Spelunky HD:ssa* pelaaja ohjaa *Spelunker* -hahmoa, tehtävänään päästä huoneesta seuraavaan. [7] Työssä tullaan käyttämään jatkossa *Spelunky* nimeä kun puhutaan *Spelunky HD:sta* työn luettavuuden helpottamiseksi. Toisessa tapaustutkimuksessa tarkastellaan Spelunkyn tasojen ja esineiden vaikutusta pelikokemukseen.

3 Proseduraalisen sisällönluonnin suunnittelu ja toiminta

Proseduraalisen sisällönluonti on yleistynyt etenkin indie-peleissä 2000-luvulla. Pääasiallisena syynä tähän on pitkälti se, että pienillä pelinkehitysyhtiöillä ei ole resursseja tai osaamista luoda suuria kokonaisuuksia peleihin samalla tavalla kuin suuremilla yrityksillä on. Jos pienet indie-peliyritykset eivät turvaudu PCG:hen joutuvat he luultavasti luomaan paljon pienemmän pelin, kuin mitä markkinoiden suuremmat yhtiöt kehittävät. Turvautuessaan PCG:n luomaan sisältöön voivat pienemmätkin peliyhtiöt kehittää pelejä, jotka luovat mukaansatempaavan pelikokemuksen useampaan otteeseen. [8]

Yksi yleinen tapa hyödyntää PCG:ia pelinkehityksessä on luoda ympäristöjä tai maastoa erilaisilla algoritmeilla. Ympäristön tai maaston luonti ei kuitenkaan ole ainoa tapa hyödyntää PCG:ia pelien kehityksessä, sillä lähes kaiken pelin sisällön voi luoda PCG:illa. *Game Dynamics: Best Practices in Procedural and Dynamic Game Content Generation* -kirjassa määritellään, että PCG voidaan *lajitella neljään* eri kategoriaan.

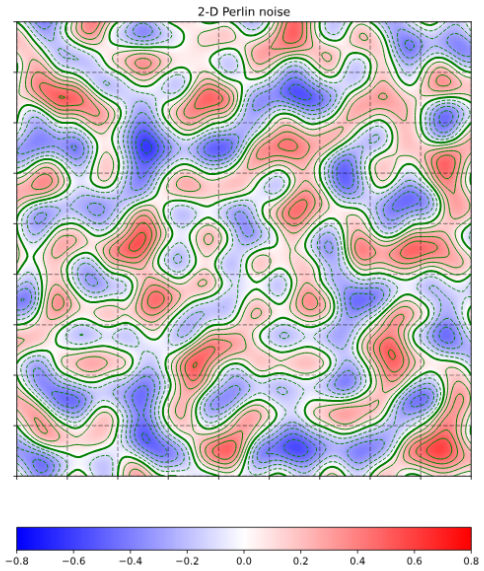
1. Pelin osiin, esim. ääni, tekstuurit
2. Pelin maailmaan, esim. ympäristö, maasto
3. Pelin järjestelmä, miten pelin sisäiset osat käyttäytyvät toistensa kanssa

4. Pelin skenaariot, tarina

Kirjassa esitetyssä taulukossa tuodaan ilmi, että esimerkiksi *Rogue* (1980) sisällytetään pelin maailman sekä pelin skenaario kategorioita hyödyntäväksi. Tässä tapauksessa pelin maailma tarkoittaa ympäristön luontia, josta *Rogue* tuli tunnetuksi, ja pelin skenaario tarinaa, joka muovautuu jokaisen pelikerran mukana. [8]

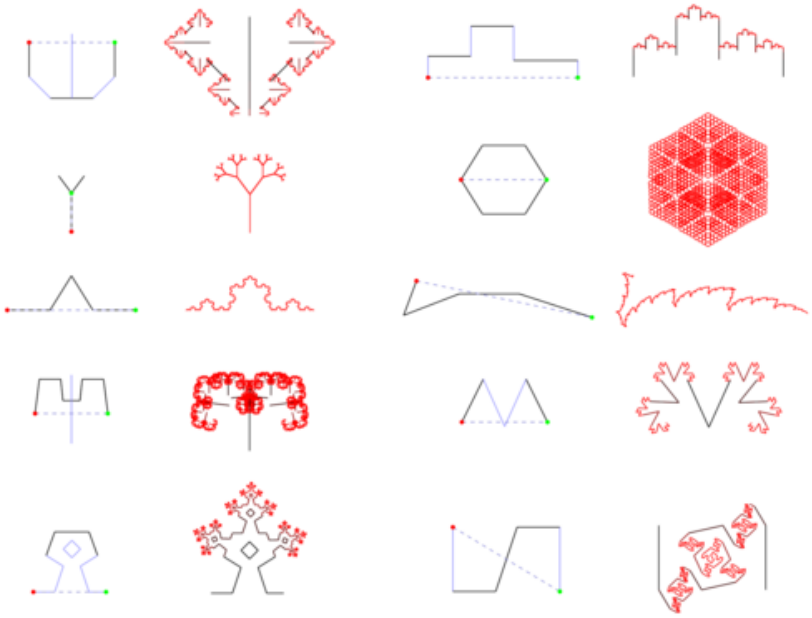
3.1 Algoritmien rooli proseduraalisessa sisällönluonnissa

PCG:n toiminnallisuus perustuu erilaisiin algoritmeihin, jotka mahdollistavat pelimaailmojen sekä muiden aikaisemmin mainittujen elementtien dynaamisen luomisen. Yleisiä pelin maailman luomiseen liittyviä menetelmiä ovat Perlin-noise (kuva 3.1) ja Simplex-noise algoritmit, jotka mahdollistavat satunnaisen, mutta hallitun käsittelyn maaston muokkaamiseen realistisesti.



Kuva 3.1: Perlin-noise menetelmä visualisoitu kaksiuloitteisesti. Kuva lisensoitu CC0 1.0 Universal -lisenssillä (`https://creativecommons.org/publicdomain/zero/1.0/deed.en`).

Toinen menetelmä, jota hyödynnetään pelin maailman satunnaisuudessa ovat fraktaalit ja L-systeemit (kuva 3.2). Kuvassa 3.2 on hyödynnetty L-systeemiä erilaisten kuvioiden luonnissa. L-systeemeitä voi hyödyntää esimerkiksi puiden ja muun kasvillisuuden luomisessa, sillä ne luovat toistuvia rakenteita, joilla on mahdollista luoda maailmasta luonnollisen näköinen. [8]



Kuva 3.2: L-systeemi kuvioiden luonnissa. Kuva lisensoitu CC0 1.0 Universal -lisenssillä (`https://creativecommons.org/publicdomain/zero/1.0/deed.en`).

Pelin skeenarioita voidaan muovata Markovin prosesseilla ja generatiivisella kieliopilla. Nämä menetelmät muovaavat tarinoiden, tehtävien ja NPC:eiden käyttäytymisen satunnaisuutta pelissä, mikä tuottaa vaihtelua ja uusia tarinallisia kokemuksia pelaajalle. Yleisimpiä pelin järjestelmään vaikuttavia meneltelmiä ovat evoluutiomenetelmät ja neuroverkot, jotka mahdollistavat pelin sisällön mukauttamisen pelaajan toiminnan perusteella. Niillä voidaan luoda dynaamisempia ja pelaajalähtöisiä pelikokemuksia. [8]

3.2 Proseduraalisen sisällönluonnin kontrollointi ja säädettävyys

Proseduraalinen sisällönluonti tarjoaa lähes rajattoman määrän vaihtelua pelikokemukseen, mutta pelinkehittäjien ja suunnitelijoiden on kuitenkin tärkeää huolehtia siitä, että lopputulos on pelattavuuden ja teemojen kannalta mielekästä. Tämän vuoksi pelinkehittäjien ja suunnitelijoiden on ajateltava PCG:in vaikutusta pelikokemukseen erilaisten menetelmien kautta. [8]

Pelinkehittäjät ja suunnittelijat voivat hallinoida algoritmeille syötettäviä parametrejä, esimerkiksi säätämällä minimi- tai maksimiarvoja, joita algoritmi käsittelee. Täten voidaan esimerkiksi vaikuttaa maaston korkeuseroihin tai vihollisten vahvuuteen. Pelinkehittäjien täytyy myös validoida sisältöä testaamalla eri parametrejä, jotta generoitu sisältö on pelattavaa ja johdonmukaista pelin tarinan ja maailman kanssa. Testausta voi suorittaa niin manuaalisesti, kuin myös luomalla testausalgoritmejä, joilla voidaan karsia pois mahdottomia esteitä tai pelin kulkua haittaavia rakenteita. Hienosäätötestaus kannattaa kuitenkin suorittaa manuaalisesti, jotta pelikokemuksesta syntyy mahdollisimman hiottu. Pelinkehittäjät voivat vaikuttaa PCG:in toimintaan luomalla algoritmeja, jotka voivat muokata generoitua sisältöä dynaamisesti pelaajan edetessä. Täten voidaan vaikuttaa reaaliaikaisesti pelaajan etenemisen ja pelityylin perusteella, mikä luo pelaajalle immersiivisen pelikokemuksen ja auttaa pitämään pelin haasteet tasapainoisena. [8]

3.3 Proseduraalisen sisällön vaikutus pelikokemukseen

PCG lisää merkittävästi pelin uudelleenpelattavuutta tarjoamalla ainutlaatuisia pelikokemuksia jokaiseilla eri pelikerralla. Satunnaisuuden ja algoritmien luomien sään-

töjen yhdistelmä luo dynaamisen ja monimuotoisen pelikokemuksen, jossa pelaaja joutuu mukautumaan jatkuvasti muuttuviin olosuhteisiin. PCG voi vaikuttaa merkittävästi pelin vaikeustason hallintaan. Dynaamisesti generoidut elementit voivat parhaimmassa tapauksessa tarjota pelaajalle sopivia, mukaansatempaavia haasteita tai johtaa epätasapainoiseen kokemukseen, jos sisältöä ei ole optimoitu huolellisesti. Pelinkehittäjien on tärkeä huomioida, että peli mukautuu pelaajan taitotason ja pelikulun mukaisesti. Vaikka proseduraalisuus voi luoda monimuotoisuutta ja jatkuvasti muuttuvan pelimaailman, se voi myös estää vaikuttavan tarinnallisen kerronnan luomista pelaajalle. Satunnaisesti tuotettu sisältö voi tuntua irralliselta tai jäädä vaille selkeää narratiivista rakennetta. Tämän vuoksi usein yhdistetään manuaalisesti suunniteltuja elementtejä ja proseduraalisia menetelmiä, jotta saavutetaan tasapaino immersiivisen ja muuttuvan pelikokemuksen välillä. [8]

4 Tapaustutkimukset

Tässä luvussa analysoidaan PCG:n käyttöä kahdessa tunnetussa roguelike-pelissä: *The Binding of Isaac* ja *Spelunky*. Luku alkaa lyhyellä esittelyllä tapaustutimuksien tavoitteista ja menetelmistä, jonka jälkeen syvennytään tarkemmin kummankin pelin PCG-ratkaisuihin. Tutkimukset tarkastelevat erityisesti satunnaisen generoinnin algoritmeja, niiden vaikutusta pelikokemukseen sekä suunnittelullisia valintoja, joilla satunnaisuutta hallitaan.

4.1 The Binding of Isaac

Tämä tapaustutkimus tarkastelee The Binding of Isaac pelin kartanluontia PCG:n avulla. Tarkastelun kohteena on huoneiden generointiin käytetty algoritmi, satunnaisuuden vaikutus pelikokemukseen ja vaikeustasoon sekä suunnitteluratkaisut, joilla satunnaisuutta hallitaan.

4.1.1 Satunnainen huoneiden luonnin algoritmi

The Binding of Isaac hyödyntää satunnaista huoneiden generointijärjestelmää, jossa algoritmi luo huoneen aina edeltävän huoneen perusteella. The Binding of Isaac taso rakentuu 9x8 ruudukon päälle (kuva 4.1, kohta a). Algoritmi sijoittaa aloitushuoneen aina soluun 35, josta se lähtee sijoittamaan neljään eri suuntaan huoneita algoritmin määrittelemästi niin, että huoneet eivät kuitenkaan muodosta silmukoita (kuva 4.1, kohta b).

(a) Aloitushuone sijoitetaan ruutuun 35. (b) Mahdollinen ensimmäisen tason kenttä

Kuva 4.1: Huoneiden muodostus kuvitettuna.

Yllä olevassa kuvassa b) kauppahuone (engl. *shop room*) on merkattu vihreänä, ja pomohuone (engl. *boss room*) on merkattu oranssilla. Kuvasta hahmottuu aiemmin mainittu algoritmin tapa sijoittaa tasolle ominaiset huoneet niin, että pomohuoneet ja kauppahuoneet sijoitetaan aina kartan reunoille.

Pelin huoneiden luonnin algoritmi käyttää leveyssuuntaista hakua (engl. *breadth first search*, BFS). BFS-hakuun pohjautuvan algoritmin ansiosta huoneiden välillä ei synny silmukoita. Pomohuone sijoittuu algoritmin ansiosta aina kauimpaan huoneeseen aloitushuoneesta. Salahuoneet (engl. *secret rooms*) määritellään lisättäväksi huoneiden risteyskohtiin, jotta mahdollisuus löytää huone olisi mahdollisimman suuri. Pelin kehittäjän mukaan luotujen huoneiden määrä määritellään yksinkertaisella funktiolla [9]:

$$\textbf{random(2)} + \textbf{5} + \textbf{level} * \textbf{2.6}$$

- random(2) määrittelee satunnaisesti luvun 0 tai 1.
- +5 määrittelee vähimmäismäärän huoneita eli siis 5 huonetta.

- level * 2.6 vaikuttaa niin, että tasoja edetessä huoneiden määrä kasvaa tason luku kerrottuna 2.6:lla.

Seuraava ohjelmalistaus 1 kuvaa kyseistä funktiota kirjoitettuna Python-ohjelmointikielellä.

Ohjelmalistaus 1 Funktio Pythonilla.

```
import random as rd
n = 1
result = rd.randint(0, 1) + 5 + (n * 2.6)
print(result)
```

4.1.2 Satunnaisuuuden vaikutus pelikokemukseen

Satunnaisesti luotu sisältö varmistaa, että pelaajan kokemus on joka pelikerralla erilainen. Uudet haasteet ja satunnaiset huoneet tekevät pelikerroista ainutlaatuisia vaikka peli onkin mekaniikaltaan yksinkertainen ja toistuva. Algoritmien luoma satunnaisuus ei kuitenkaan luo täysin kaootista pelikokemusta, vaan pelinkehittäjät ovat onnistuneet rajaamaan satunnaisuuden, niin että pelikokemus pysyy vakaana.

Jotta huoneet eivät olisi sekavia, pelinkehittäjät ovat tehneet etukäteen luotuja pohjia, joihin huoneen sisältö pohjautuu. Pelissä on esimerkiksi 174 mahdollista pohjaa normaaleille huoneille, jotka esiintyvät pelin ensimmäisessä luvussa. Huoneen valmis pohja sisältää itse huoneessa esiintyvät esineet ja viholliset, mutta valmiiseen pohjaan on myös mahdollista lisätä esimerkiksi minipomovihollisia (engl. *mini boss*) tai värjättyjä kiviä (engl. *tinted rocks*), jotka antavat pelaajalle tuhoutuessaan palkinnon. [9] Vaikka The Binding of Isaac pohjautuu rogueliken permadeath mekaniikan ympärille, palkitaan pelaajaa kuitenkin erilaisilla hahmoilla ja esineillä, jotka aukeavat useiden pelikertojen välillä. Tämä luo pelaajalle ainutlaatuisen pelikokemuksen, joka on vaikuttanut erityisesti The Binding of Isaacin suosioon.

4.2 Spelunky

Tämä tapaustutkimus käsittelee Spelunky pelin kenttien ja tavaroiden (engl. item) luontia PCG:in avulla, ja sitä miten tämä vaikuttaa pelikokemukseen. Tapaustutkimus tarkastelee menetelmiä, joilla kenttä ja sen esineet generoidaan, satunnaisuuden vaikutusta pelikokemukseen ja vaikeustasoon sekä suunnitteluratkaisuja, jolla satunnaisuutta hallitaan.

4.2.1 Satunnainen kentän luonti

Ennen kuin voimme tarkastella tavaroiden ja vihollisten luontia, täytyy meidän ensiksi selvittää miten ja millaisia tasoja Spelunkyssa käytetty algoritmi tuottaa, sillä tavarat ja viholliset määräytyvät tason perusteella. [10] Tapaustutkimus keskittyy kuitenkin tasossa olevien tavaroiden luontiin, ei niinkään itse tason luontiin.

Spelunky käyttää kaksiosaista järjestelmää tasojen luontiin. Taso luodaan "hybrid" -lähestymistavalla, joka koostuu seuraavista menetelmistä: [10]

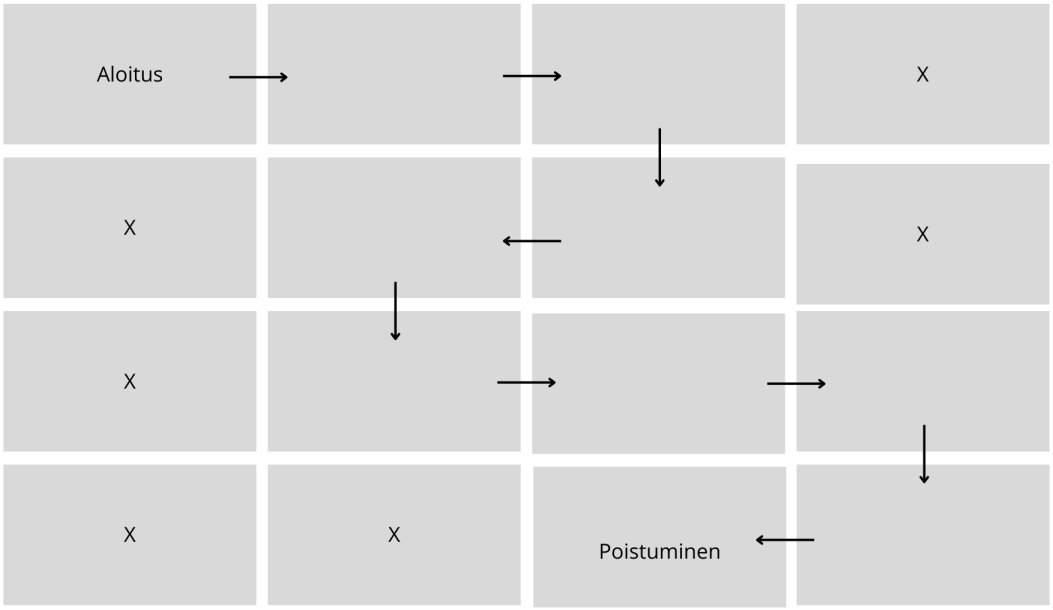
1. Malliin pohjautuva luonti (engl. *Template based generation)*
2. Satunnainen siemen (engl. *Random seeding*)

Malliin pohjautuva luonti yhdistää valmiiksi generoituja osioita (engl. chunks) tasosta, luoden täten uuden ainutlaatuisen tason pelaajalle. Valmiit osiot määräävät esimerkiksi sen, että viholliset ja esineet sijoitetaan loogisesti oikeisiin paikkoihin, jotta pelikokemus pysyisi haastavana ja luonnollisena, eikä olisi täysin satunnaisuuden varassa. Satunnainen siemen luo pelin kentän sen sisällön antaman datan perusteella, joka määrittelee kentän sisällön.

Seuraavat kuvat hahmottavat tason luontia Spelunkyssa. Taso aloitetaan 4 x 4 ruudukosta (kuva 4.2). [7] Tämän jälkeen määräytyy reitti aloitusovelta poistumisovelle, mikä määrittelee mitkä huoneet ovat välttämättömiä poistumisovelle pääsyä

varten. Aloitusovi sijoitetaan aina yhdelle ylimmän rivin ruuduista, ja poistumisovi sijoitetaan aina yhdelle alimman rivin ruuduista. X:llä merkatut huoneet voivat sisältää vihollisia tai esineitä, mutta ne eivät ole välttämättömiä tason läpäisyä varten (kuva 4.3). [7]

Kuva 4.2: 4 x 4 ruudukko, jonka pohjalle taso rakentuu



Kuva 4.3: Generoitu reitti tasolle

Kun tason reitti on määritelty, valitaan jokaiselle huoneelle pohja, joka voidaan yhdistää viereisiin huoneisiin. Viimeisenä tason huoneisiin lisätään esineet ja vastustajat valmiiksi määrättyjen todennäköisyyksien perusteella ja ympäristö huomioiden. [7]

4.2.2 Spelunkyn esineet

Spelunkyssa on 51 erilaista esinettä. [11] Nämä esineet voidaan jakaa kolmeen eri kategoriaan:

1. Kertakäyttöiset esineet (engl. *Consumable items*)
2. Varusteet (engl. *Accessories*)
3. Aseet (engl. *Weapons*)

Kertakäyttöisiä esineitä voi nimensä mukaan käyttää vain kerran. Tällaisia esineitä ovat esimerkiksi köydet sekä pommit. Kertakäyttöisiä esineitä voi löytää kentästä tai ostaa kaupasta. Varusteet ovat esineitä, jotka pysyvät pelaajalla koko pelin läpi, paitsi jos ne korvataan jollain toisella varusteella. Varusteet antavat etuja (engl. *buffs*) pelaajalle. Esimerkkejä varusteista ovat viitta ja silmälasit. Varusteet, jotka parantavat saman osa-alueen mekaniikkaa korvaavat aina toisensa. Esimerkiksi lentämisen mahdollistava rakettireppu korvaa viitan, joka hidastaa putoamista. Näiden lisäksi Spelunkyssa on myös aseita. Aseet jaetaan lähiaseisiin ja etäaseisiin. Lähiaseet vaativat lähikontaktia vastustajan kanssa kun taas etä-aseita voidaan käyttää pidemmältä kantamalta. [11]

4.2.3 Satunnaisuuden vaikutus pelikokemukseen

Aiemmin mainittujen PCG:in suunnitelmallisuuden ja hallinnan avulla Spelunkyn pelikokemus pysyy aina ainutlaatuisena, mutta kuitenkin vakaana. Tasojen satunnaisuus takaa sen, että pelaaja ei koe jokaista pelikertaa samana, eikä voi esimerkiksi opetella tasojen rakennetta ulkoa. Tällaisella satunnaisuudella peli pysyy jokaisella kerralla tuoreena ja pitää pelaajan mielenkiinnon pelissä pidempään.

Vaikka tasojen rakenne onkin satunnainen, malliin pohjautuva luonti varmistaa, että satunnaisuus on hallittua, ja esineet sekä tavarat sijoittuvat loogisesti, jolloin pelaajan immersio ei rikkoudu. Tällainen satunnaisuuden kontrollointi varmistaa myös sen, että tasot eivät ole liian helppoja taikka liian hankalia. Joissakin peleissä satunnaisuus saattaa tehdä pelistä helpomman tai vaikeamman, mutta Spelunkyn tasojen reitinluonti varmistaa, että tasojen vaikeusaste pysyy suhteellisen vakiona, sillä pelaaja voi olettaa, että reitti aloitusovesta lopetusovelle on aina lähes yhtä pitkä. Koska esineiden ja vihollisten sijoittelu ei ole ennalta-arvattavissa, joutuu pelaaja tekemään nopeita päätöksiä tason läpäisyä varten. Tämä luo pelaajalle val-

lantunten, joka lisää immersiota pelaajalla, täten luoden posiitivisen vaikutuksen pelaajaan.

5 Yhteenveto

Tämä tutkielma käsitteli PCG:in vaikutusta ja roolia roguelike-genren peleissä, keskittyen erityisesti sen vaikutukseen pelikokemukseen ja pelin tasapainoon. Tutkielma vastasi seuraaviin tutkimuskysymyksiin.

(Tk 1.) PCG:illa luodaan ainutlaatuisia dynaamisia pelikokemuksia pelaajalle. Näitä ovat esimerkiksi satunnaisesti luodut kentät, esineet sekä viholliset. Tapaustutkimuksista kävi ilmi, että vaikka Spelunky ja The Binding of Isaac ovat samantyylisiä pelejä, ne käyttävät täysin erilaisia mekaniikkoja ja menetelmiä tasojen, esineiden ja vihollisten luontiin. Niiden hyödyntämät pelimekaaniikat varmistavat, että jokainen pelikerta on ainutlaatuinen, mutta kuitenkin hallittu. (Tk 2.) PCG:in avulla voidaan luoda erittäin tasapainoisia ja toimivia pelikokemuksia, jotka säilyttävät pelaajan mielenkiinnon. Tapaustutkimuksen esimerkkinä oleva Spelunky osoitti, että sijoittamalla esineet ja viholliset loogisesti, voidaan luoda toimiva sekä tasapainoinen pelikokemus. The Binding of Isaac osoitti, että rajoittamalla satunnaisuus ennalta määriteltäviin pohjiin, voidaan luoda tasapainonen pelikokemus. (Tk 3.) Permadeathin tyyliset pelimekaniikat vaativat sen, että PCG:llä luotu sisältö on tasapainoista ja reilua. PCG-algoritmien täytyy ottaa huomioon pelaajan taitotaso sekä edistyminen.

Tutkielmaa luodessani opin, että PCG:ia on mahdollista hyödyntää monilla eri tavoilla riippuen pelistä. PCG:in sulavuuden ja onnistumisen määrittää pitkälti sen suunnitelmallisuus, testaus ja hallinta. Yllätyin siitä kuinka paljon PCG ohjaa ja

vaikuttaa pelikokemukseen, sekä siitä kuinka se vaikuttaa pelin immersioon ja uudeelleenpelattavuuteen. Aiemmin ajattelin, että PCG:in satunnaisuus toimii ainoastaan pelin teknisenä ratkaisuna, mutta nyt ymmmärrän sen merkityksen myös pelin suunnitelun kannalta. Jälkikäteen olisin halunnut tutkia enemmän pelaajalähtöistä PCG:ia kuten vaikkapa pelikokemukseen mukautuvat neuroverkot, joka olisi tuonut lisänäkökulman tutkimukseen.

Tutkimuksessa olisi voinut hyödyntää laajemmin vertaisarvioituja artikkeleja varsinkin tekoälyyn pohjautuvista PCG-järjestelmistä. Tutkimusprosessi herätti myös uusia kysymyksiä, kuten miten PCG tulee kehittymään tulevaisuudessa tekoälyn kehityksen myötä sekä voiko PCG yksinään luoda täysin ainutlaatuisen ja dynaamisen tarinan, joka kehittyisi pelaajan valintojen myötä?

Yhteenvetona voidaan todeta, että PCG on äärimmäisen keskeinen osa roguelikegenren videopelejä, ja se vaikutaa hyvin toimiessaan pelin suosioon. Tutkimus toi esiin PCG:in monimuotoisuutta ja sen merkitystä pelin suunnittelun kannalta. Jatkossa aihetta voisi laajentaa käsittelemään useampia peligenrejä.

Lähdeluettelo

[1] N. Shaker, J. Togelius ja M. J. Nelson, *Procedural Content Generation in Games*. Springer International Publishing, 2016, ISBN: 9783319427164. DOI: 10.1007/978-3-319-42716-4.

[2] J. Froholt. "The story of Rogue". (2025), url: https://spillhistorie.no/the-story-of-rogue/ (viitattu 07. 02. 2025).

[3] A. Gellel ja P. Sweetser, "A Hybrid Approach to Procedural Generation of Roguelike Video Game Levels", teoksessa *International Conference on the Foundations of Digital Games*, sarja FDG '20, ACM, syyskuu 2020, s. 1–10. DOI: 10.1145/3402942.3402945.

[4] S. G. Orbán, G. N. Szabados, É. Bácsné Bába, V. Fenyves, Z. Bács, A. Molnár, G. Ráthonyi ja H. Rizwan, "Roguelike games: The way we play", *International Journal of Engineering and Management Sciences*, vol. 7, nro 4, s. 80–92, huhtikuu 2023, ISSN: 2498-700X. DOI: 10.21791/ijems.2022.4.7..

[5] E. McMillen. "The Binding of Isaac". (2025), url: https://store.steampowered.com/app/113200/The_Binding_of_Isaac/ (viitattu 12. 02. 2025).

[6] Mossmouth. "Spelunky". (2025), url: https://store.steampowered.com/app/239350/Spelunky/ (viitattu 12. 03. 2025).

[7] D. Stammer, T. Günther ja M. Preuss, ”Player-adaptive Spelunky level generation”, teoksessa *2015 IEEE Conference on Computational Intelligence and Games (CIG)*, 2015, s. 130–137. DOI: 10.1109/CIG.2015.7317948.

[8] O. Korn, *Game Dynamics*. Springer International Publishing, 2017, ISBN: 9783319530888. DOI: 10.1007/978-3-319-53088-8.

[9] BorisTheBrave. ”Dungeon Generation in Binding of Isaac”. (2020), url: https://www.boristhebrave.com/2020/09/12/dungeon-generation-in-binding-of-isaac/ (viitattu 12. 02. 2025).

[10] W. Baghdadi, F. S. Eddin, R. Al-Omari, Z. Alhalawani, M. Shaker ja N. Shaker, ”A Procedural Method for Automatic Generation of Spelunky Levels”, teoksessa *Applications of Evolutionary Computation*. Springer International Publishing, 2015, s. 305–317, ISBN: 9783319165493. DOI: 10.1007/978-3-319-16549-3_25.

[11] T. Thompson, ”"With Fate Guiding My Every Move": The Challenge of Spelunky.”, teoksessa *FDG*, 2015. url: http://www.fdg2015.org/papers/fdg2015_paper_18.pdf (viitattu 19. 03. 2025).