Esko Hokkanen

PELIMAAILMAN GENEROINTI PAIKKA-TIEDON AVULLA

Opinnäytetyö

Tekniikan ammattikorkeakoulututkinto

Peliohjelmoinnin koulutus

2024

Kaakkois-Suomen ammattikorkeakoulu

Tutkintonimike	Insinööri (AMK)
Tekijä	Esko Hokkanen
Työn nimi	Pelimaailman generointi paikkatiedon avulla
Toimeksiantaja	Virtuaalinen sairaalaympäristö 1.0 -hanke
Vuosi	2024
Sivut	36 sivua
Työn ohjaaja	Teemu Saarelainen

TIIVISTELMÄ

Opinnäytetyössä kehitetään ohjelmisto, jolla voidaan luoda 3D-visualisointi maailmanlaajuisen paikkatiedon pohjalta Unity-pelimoottorissa. Työ tehdään huomioiden ensisijaisesti pelinkehityksen tarpeet 3D-visualisoinnin generoimiseen. Tavoitteena on hyödyntää 3D-visualisointia käyttökelpoisen pelimaailman luomiseen. Tämän vuoksi visualisoinnin luomisessa priorisoidaan sen käytettävyyttä datan realistisesti kuvaamisen yli.

Opinnäytetyön tutkimusongelmana on 3D-visualisaation luominen alueesta oikean maailman sijainnin pohjalta Unity-pelimoottorissa. Tutkimusongelmaan vastataan osana ohjelmistonkehitysprosessia. Työn teoriaosiossa tarkastellaan Unity-pelimoottorille kehitettyjä ohjelmistoja paikkatiedon 3D-visualisointiin ja niiden käyttämiä tekniikoita. Sen lisäksi käydään läpi erinäisiä käyttökohteita paikkatiedon hyödyntämiselle peleissä. Tekniseen toteutukseen valituista datalähteistä ja ohjelmointirajapinnoista kerrotaan pääpiirteittäin, minkä lisäksi tarkastellaan syvemmin niiden käyttämiä tietomalleja. Datasta tutkitaan 3D-visualisaatiolle kriittisen datan määrää ja laatua sekä mahdollisia ratkaisuja sen epäkohtien ja puutteiden käsittelemiseen. Viimeiseksi datasta luodaan 3D-visualisaatio ohjelmallisesti Unity-pelimoottorissa.

Teknistä toteutusta varten valitaan kaksi avointa datalähdettä: OpenStreetMap ja OpenTopography, joiden ohjelmointirajapintojen kautta päästään käsiksi maailmanlaajuiseen kartta- ja korkeusdataan Unity-pelimoottorissa. Työssä ratkaistaan 3D-visuaalisoinnille kriittisen datan puutteellisuus täydentämällä se olemassa olevan datan perusteella. Visualisoinnin laadun ylläpitämiseksi hylätään data, jonka muotoilussa ilmenee epäkohtia. Pelimaailman ohjelmallisesti luomiseen käytetään useita eri tekniikoita. Maaston luomiseen hyödynnetään Unity-pelimoottorin Terrain-järjestelmää. Rakennuksien polygonimallit luodaan ekstruusiona niiden kaksiulotteisesta muodosta, kun taas teiden polygonimallit generoidaan spline mesh -työkalulla. Kasvusto puolestaan luodaan instanssoituina prefab-objekteina.

Asiasanat: geoinformatiikka, paikkatietojärjestelmä, proseduraalinen mallinnus

South-Eastern Finland University of Applied Sciences

Degree title	Bachelor of Engineering
Author	Esko Hokkanen
Thesis title	Generating a game world based on geographic information
Commissioned by	Project Virtual Hospital Environment 1.0
Time	2024
Pages	36 pages
Supervisor	Teemu Saarelainen

ABSTRACT

The software developed in this thesis enables the creation of a 3D visualization based on global geographic information in Unity. Development of the software primarily considers the needs of game development for generating 3D visualizations. The objective is to make use of 3D visualization for creating a functional game world. Therefore, the emphasis is on usability rather than representing the data realistically.

The research problem of the thesis is the creation of a 3D visualization based on a real-world location in Unity. The research problem is solved as a part of the software development process. The theoretical part of the thesis examines software developed for Unity for the purpose of visualizing geographic information in 3D and the techniques utilized by the software. Additionally, various use cases are covered for utilizing geographic information in games. The technical implementation starts by outlining the selected data sources and application programming interfaces, followed by a more comprehensive examination of the data models they employ. The quantity and quality of data critical for 3D visualization is assessed, together with possible solutions to addressing defective data. Finally, the data is utilized programmatically to create a 3D visualization in Unity.

Two open data sources are chosen for the technical implementation: OpenStreetMap and OpenTopography. The map and elevation data they provide can be accessed through programming interfaces in Unity. Any missing data that is critical for 3D visualization is supplemented by existing data. To maintain the quality of visualization, defective data is discarded. Various methods are utilized for programmatically creating the game world. The Unity Terrain system is used for terrain creation. Building meshes are created by extruding the 2D shape of the building, while road meshes are generated using a spline mesh tool. Conversely, vegetation is instantiated from prefabs.

Keywords: geoinformatics, geographic information system, procedural modeling

SISÄLLYS

1 JOHDANTO

Opinnäytetyö käsittelee pelimaailman luomista ohjelmallisesti hyödyntäen kartta- ja korkeusdataa. Työssä selvitetään, kuinka maailmanlaajuista paikkatietoa voidaan hyödyntää alueen kolmiulotteiseen visualisointiin pelituotantoa varten. Työssä kehitetään ohjelmisto, jolla voidaan luoda pelimaailma hyödyntäen kahta avointa datalähdettä.

Työssä tarkastellaan vastaavia Unity-pelimoottorille kehitettyjä ohjelmistoja ja niiden käyttämiä tekniikoita sekä erinäisiä käyttökohteita paikkatiedon hyödyntämiselle peleissä. Teknisestä toteutuksesta kerrotaan ensin valituista datalähteistä ja niiden valintaperusteista. Lähteiden datasta käsitellään syvemmin 3D-visualisointiin käytettävän datan määrää ja laatua sekä sen tuomia komplikaatioita. 3D-visualisaatiolle kriittisen datan epäkohtiin ja vajeisiin etsitään sopivat ratkaisut. Lopulta käydään läpi, kuinka ympäristö saadaan luotua ohjelmallisesti Unity-pelimoottorissa.

Karttadatan kattaman tiedon laajuuden vuoksi opinnäytetyö on rajattu käsittelemään neljän pääkokonaisuuden generoimista: maasto, rakennukset, tiet ja kasvusto. Jokainen pääkokonaisuus hyödyntää erilaisia metodeja paikkatiedon 3D-visualisointiin. Kyseiset neljä pääkokonaisuutta integroidaan yhtenäiseksi pelimaailmaksi.

Opinnäytetyön aihe nousi esiin halusta toteuttaa pelimaailman generointi ohjelmallisesti. Kyseinen aihe on kuitenkin laajalti käsitelty vastaavissa teoksissa. Tämän vuoksi opinnäytetyön aihe tarkennettiin vähemmän käsiteltyyn aihealueeseen, joka hyötyy ohjelmallisesta pelimaailman generoinnista. Pelimaailman generointi paikkatiedon avulla valittiin aiheeksi, sillä siitä ei löytynyt vastaavia alalla toteutettuja opinnäytetöitä.

2 TUTKIMUSASETELMA

Tässä luvussa käsitellään tarkemmin opinnäytetyön tavoitteita sekä niiden saavuttamiseen liittyviä haasteita. Lisäksi käydään läpi opinnäytetyön tutkimusongelma ja siitä johdetut tutkimuskysymykset. Lopuksi kerrotaan työn toimeksiantajasta.

2.1 Opinnäytetyön tavoitteet

Opinnäytetyössä on kyseessä toimintatutkimus. Tavoitteena on suunnitella ja valmistaa ohjelmisto, jolla voidaan generoida pelimaailma Unity-pelimoottorissa kartta- ja korkeusdatan perusteella. Työ tehdään täyttämään toimeksiantajan tarvetta tuoda Kotka pelimaailmaan. Toteutuksen perustana hyödynnetään aikaisempaa teoriaa aiheesta. Pelimaailman generoimisen täytyy olla mahdollista hyödyntäen maailmanlaajuista dataa sekä pelin suoritusaikana että Unity-editorissa. Ohjelmisto luodaan huomioiden ensisijaisesti pelinkehityksen tarpeet. Käytännössä se tarkoittaa, että tuotoksen käytettävyyttä pelinkehityksessä priorisoidaan datan realistisesti kuvaamisen yli. Työn onnistumista arvioitaessa on olennaista huomioida tuotoksen hyöty hankkeelle sekä laajempi hyöty pelinkehityksen tarpeisiin.

Opinnäytetyön tekninen toteutus pohjataan kokonaisuudessaan valitun datalähteen dataan. Sen täydentämiseksi ei tehdä ollenkaan tiedonkeruuta. Työn kannalta on siis olennaista valita sopiva datalähde, joka soveltuu opinnäytetyön tavoitteeseen. Datan laajuus aiheuttaa tietynlaisia komplikaatioita, jotka on käsiteltävä ennen pelimaailman luomista. 3D-visualisaation luomiseksi täytyy ratkaista, kuinka karttaelementtien geometria määritetään ohjelmallisesti. Sen lisäksi on ratkaistava, kuinka korkeus- ja karttadatan pohjalta luodut elementit saadaan sovitettua yhteen luontevasti. Pelimaailman luomisprosessi pyritään ohjelmoimaan riittävän tehokkaaksi kokonaisuudessaan, jotta se ei vie tarpeettoman paljon aikaa suhteutettuna luotavan alueen kokoon.

2.2 Tutkimuskysymykset

Opinnäytetyön tutkimusongelmana on 3D-visualisaation luominen alueesta oikean maailman sijainnin pohjalta Unity-pelimoottorissa. Tutkimusongelmasta on johdettu seuraavat kysymykset:

- Miten saadaan tarpeeksi kattavaa dataa?
- Miten datasta saadaan käyttökelpoinen pelimaailma?
- Miten pelimaailma luodaan ohjelmallisesti?

Näihin kolmeen kysymykseen vastaaminen on oleellista työn onnistumiselle. Niihin pyritään vastaamaan opinnäytetyössä osana ohjelmistonkehitysprosessia.

2.3 Toimeksiantaja

Opinnäytetyön toimeksiantajana toimii digitaalisen talouden TKI (tutkimus-, kehittämis- ja innovaatiotoiminta) Virtuaalinen sairaalaympäristö -hanke, joka on rahoitettu REACT-EU-kehitysrahastosta osana Euroopan unionin covid-19-pandemian vuoksi toteutettuja toimia. Hankkeen tavoite on kehittää Kymenlaakson virtuaalisairaalan ensimmäinen versio, Kotkasaaren ympäristön virtuaalimalli, sekä opetusta edistäviä skenaarioita. (Hannula 2022.) Työ tehdään huomioiden Virtuaalinen sairaalaympäristö -hankkeen tarpeita Kotkan seudun 3D-mallin täydentämisestä.

3 PAIKKATIEDON 3D-VISUALISOINTI

3D-visualisointi avaa monia uusia mahdollisuuksia paikkatiedon soveltamiselle. Sen avulla voidaan luoda oikeaa maailmaa jäljitteleviä ympäristöjä, jotka ovat heti tunnistettavissa. Nykyiset pelimoottorit mahdollistavat paikkatiedon kokoamisen interaktiiviseksi, realistisen näköiseksi ympäristöksi. Ympäristön tarkkailun lisäksi kolmiulotteinen virtuaalinen maailma mahdollistaa siinä liikkumisen ja sen simuloimisen. Nykypäivänä on saatavilla laaja valikoima erilaisia työkaluja, jotka mahdollistavat paikkatietojärjestelmien hyödyntämisen peleissä. (Heidelberg 2021.)

3.1 Paikkatiedon 3D-visualisointi Unity-pelimoottorissa

Unity-pelimoottorilla on toteutettu useita eri implementaatioita paikkatiedon 3D-visualisoinnista erinäisiin tarkoituksiin. Tässä luvussa käsitellään kahta im-

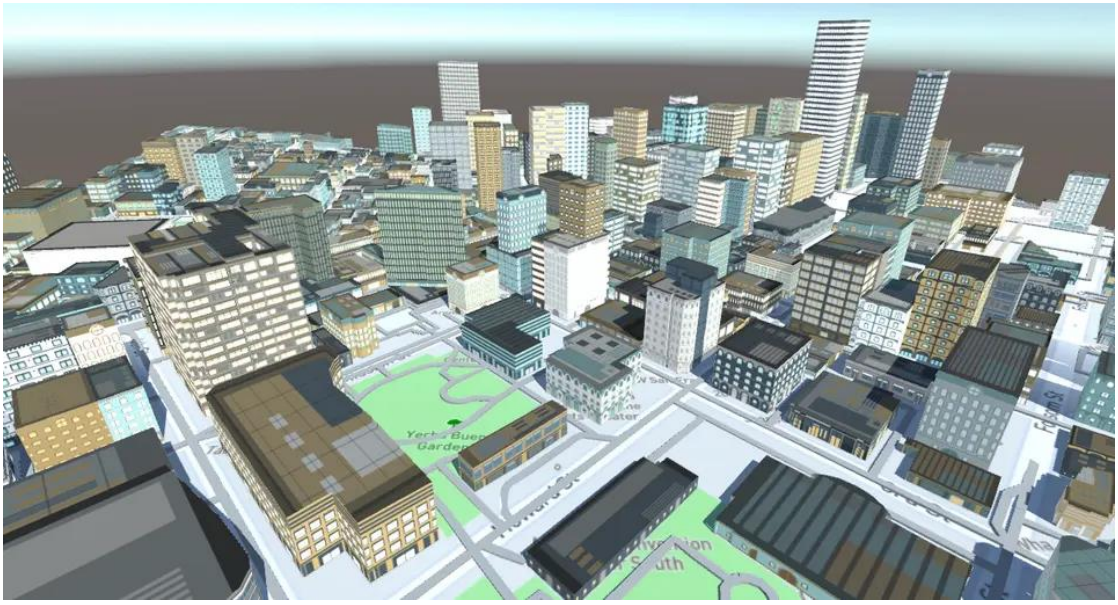
plementaatiota: Mapbox Unity SDK ja CityGen3D, jotka ovat opinnäytetyön tavoitteita läheisimmät. Kyseisistä ohjelmistoista selvitetään, miten ne ratkaisevat opinnäytetyölle relevantteja 3D-visualisaation haasteita.

3.1.1 Mapbox Unity SDK

Mapbox Unity SDK (Software Development Kit, ohjelmistokehityspaketti) on vain pieni osa Mapbox-alustan tarjoamista palveluista. Sen kautta pääsee käsiksi laajaan valikoimaan ohjelmointirajapintoja, kuten kartta-, sijainti- ja navigointipalveluihin. Opinnäytetyölle relevanttia on ensisijaisesti, kuinka ohjelmistokehityspaketin tarjoama valmis ratkaisu luo kolmiulotteisen visualisoinnin paikkatiedon pohjalta. Kyseinen ratkaisu on kohdennettu pääasiassa karttapalveluihin. Täten se sopii parhaiten peleihin, jotka hyötyvät karttapalveluista, kuten sijaintipohjaiset pelit. Mapbox Unity SDK mahdollistaa kaksi- ja kolmiulotteisten karttojen generoimisen pelin suoritusaikana. Kartan kolmiulotteiset elementit sisältävät maaston, rakennukset ja tiet. (Maps SDK for Unity s.a.) Visualisalisointiin käytetty karttadata perustuu pääosin OpenStreetMap-palvelun kautta saatuun dataan (Mapbox Streets v7 s.a.).

Mapbox Unity SDK käyttää maaston luomiseen Mapbox Terrain -dataa, joka koostuu useista eri valtioiden tarjoamista dataseteistä sekä kolmansien osapuolten kaupallisesta datasta (Mapbox Terrain v2 s.a.). Maasto voidaan luoda joko tasaiseksi tai korkeuskartan pohjalta. Maaston teksturointiin pystyy käyttämään vektorikarttaa, satelliittikuvaa tai Mapbox Studion avulla luotua tyyliä. Maxbox Studio mahdollistaa monipuolisten karttatyylien määrittämisen selaimessa ja sovelluksessa. Luotu tyyli on heti valmis käyttöön ohjelmistokehityspaketin kautta Unity-pelimoottorissa. (Maps SDK for Unity s.a.)

Mapbox Unity SDK luo rakennukset ekstruusiona niiden kaksiulotteisesta muodosta kartassa. Käytännössä siis kaksiulotteista muotoa venytetään pystysuunnassa 3D-objektin luomiseksi. Rakennusten korkeus voidaan määritellä karttadatalla, minimi- ja maksimi korkeudella tai yksittäisarvolla. Rakennuksien teksturointi toimii osittain satunnaisesti, minkä vuoksi se on erilainen riippuen luomiskerrasta. Mapbox tarjoaa useita erilaisia valmiita tyylejä rakennuksien teksturointiin. Sen lisäksi käyttäjä itse voi määritellä rakennusten teksturointiin käytettävän tyylin. (Maps SDK for Unity s.a.)



Kuva 1. Kaupunki luotuna Mapbox Unity -ohjelmistokehityspaketilla (Maps SDK for Unity s.a.)

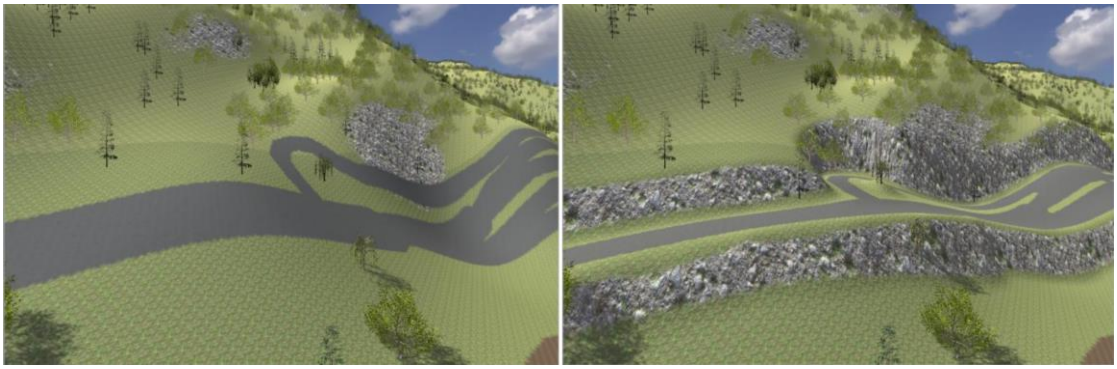
Teiden geometrian laskeminen toimii vastaavalla tavalla kuin rakennuksissa. Reitin perusteella tielle määritellään kaksiulotteinen muoto, josta muodostetaan 3D-objekti ekstruusiona. Tien korkeudelle on vastaavat vaihtoehdot kuin rakennuksissa. Kuvasta 1 voidaan havaita, kuinka kyseinen metodi luo hyvin yksinkertaisen, kulmikkaan tien. Sen lisäksi tietä ei soviteta luodun maaston muotoihin. Täten tiet eivät sovellu käytettäväksi osana pelimaailmaa, jossa kuuluisi pystyä liikkumaan pelihahmolla, mutta tekee tehtävänsä kolmiulotteisessa kartassa.

3.1.2 CityGen3D

CityGen3D on Unity-laajennus, jolla pystyy lataamaan ja prosessoimaan karttadataa sekä luomaan 3D-maailman kyseisen datan pohjalta. Virtuaalimaailma pohjautuu OpenStreetMap-dataan, joka mahdollistaa pelimaailman luomisen minkä vain oikean maailman sijainnin pohjalta. CityGen3D on tehty erityisesti Unity-editor ympäristössä käytettäväksi ja ei täten tue alueen luomista pelin suoritusaikana. Laajennus ei myöskään vaadi käyttäjältä ohjelmointiosaamista sen käyttöliittymäkeskeisen käyttökokemuksen vuoksi. (CityGen Technologies 2023.)

CityGen3D sallii NASADEM-datan lataamisen, ulkopuolisen korkeuskartan tuonnin ja Perlin-kohinan hyödyntämisen maaston luomista varten. CityGen3D

tekee korkeuskarttaan muokkauksia karttadatan perusteella ennen maaston luomista. Tämän tarkoituksena on parantaa kartta- ja maastodatan yhteensovittamista. Esimerkiksi vesistöt luodaan tekemällä syvennys maastoon, jotta vedenpinta on maaston yläpuolella. Maasto voidaan myös halutessa teksturoida automaattisesti omien sääntöjen perusteella tai käyttämällä Mapbox ohjelmointirajapinnan (API, application programming interface) kautta ladattua maastokuvaa. (CityGen Technologies 2023.)



Kuva 2. Tiet ennen maaston tasoitusta ja sen jälkeen CityGen3D-laajennuksessa (CityGen Technologies 2023)

Teiden luomisessa CityGen3D hyödyntää proseduraalista geometriaa, maaston muokkausta ja dekaaleja. Tie luodaan teksturoimalla se suoraan maastoon ja tasoittamalla maastoa tarvittaessa, kuten kuvan 2 tapauksessa maaston ollessa todella jyrkkä tien kohdalla. Tien varrelle lisätään proseduraalisesti tienvarsielementtejä kuten suojakaiteita ja lyhtypylväitä. Tien luomismetodin vuoksi se ei tue siltoja. (CityGen Technologies 2023.)

Rakennukset luodaan ekstruusiona niiden kaksiulotteisesta muodosta kartassa. Rakennuksille ei luoda sisätiloja vaan ne on tarkoitettu enimmäkseen taustaelementeiksi peliin. Rakennus teksturoidaan sen tyypin perusteella. Käyttäjä voi muokata rakennustyyppien teksturointiin käytettävää tyyliä ja vaikuttaa sitä kautta luotavan pelimaailman visuaaliseen ilmeeseen. (CityGen Technologies 2023.)

3.2 Paikkatiedon käyttö peleissä

Peleissä tavoite on luoda virtuaalimaailma interaktiivisena osana peliä eikä vain tarkkailtavaksi. Siispä useat nykyaikaiset paikkatietoa hyödyntävät pelit

vaativat vähintäänkin yhtä tarkkoja ja yksityiskohtaisia datasettejä kuin ammattimaiset paikkatietojärjestelmät. On olemassa monia eri tapoja integroida paikkatieto peliin. Pelin pohjana voidaan hyödyntää olemassa olevaa palvelua. Esimerkkejä kyseisestä lähestymistavasta ovat MapsTD, joka hyödyntää Google Maps -palvelun karttaa pelin maailmana sekä erinäiset visailupelit, kuten Geoguesrr, joka hyödyntää Google Maps -palvelun Street View -näkymää. Jotkin pelinkehittäjät puolestaan luovat oman sovelluksen, jolla luoda ja päivittää paikkatietomallejaan. Tästä yksi esimerkki on Microsoft Flight Simulator, joka hyödyntää maailmanlaajuista paikkatietoa koko maapallon 3D-visualisointiin. Sen lisäksi paikkatietoa voidaan hyödyntää pohjana paikallisen sijainnin tarkkaan 3D-visualisointiin, kuten SSX-laskettelupelissä, jossa on jäljennetty oikeita vuoristoja NASA:n korkeusmallien pohjalta. Paikkatietoa käyttävät myös hyötypelit, kuten koulutukseen sovelletut pelit. Oikean sijainnin pohjalta luodussa virtuaalimaailmassa voidaan simuloida paljon harjoitustilanteita, joita oikeassa maailmassa ei voida replikoida. (Chądzyńska & Gotlib 2016.) Tässä luvussa käsitellään syvemmin kahta esimerkkiä peleistä, jotka hyödyntävät paikkatietoa keskeisenä osana peliä.

3.2.1 Pokémon Go

Pokémon Go:n julkaisu vuonna 2016 demonstroi maantieteellisen tietojenkäsittelyn potentiaalin luoda massiivisen suosittu peli. Peli on osoittanut vaikuttavaa pysyvyyttä ja inspiroinut seuraavan sukupolven AR-pohjaisia pelejä. Pelin ainutlaatuisuus ja vetovoima maailmanlaajuisesti perustuivat mahdollisuuteen löytää olentoja todellisista paikoista paikkatietoteknologian avulla. (USC 2021.)

Sovelluksessa pelaajat näkevät avatarinsa liikkuvan kartalla, joka sovitetaan heidän omaan maantieteelliseen sijaintiinsa. Oma ympäristö vaikuttaa siihen minkälaisia Pokémoneja löytää todennäköisimmin. Esimerkiksi vesityypin Pokémonit löytyvät todennäköisimmin järvien ja jokien lähistöltä. Joitakin Pokémoneja puolestaan voi löytää vain tietyistä osista maailmaa. Sen lisäksi ympäristöstä erottuvat paikat voidaan nimittää poké-pysäkeiksi ja poké-saleiksi, jonka kautta pelaaja saa hyödyllisiä resursseja. (USC 2021.)

Alun perin Pokémon Go käytti Google Maps -palvelun paikkatietoja. Kuitenkin vuoden 2017 lopulla tehdyn merkittävän päivityksen myötä peli siirtyi käyttämään OpenStreetMap-palvelua paikkatiedon lähteenä. Muutos mahdollisti käyttäjien tekemät muokkaukset ja nopeamman sopeutumisen muutoksiin oikeassa maailmassa. (USC 2021) Monet pelaajat ovat tehneet OpenStreetMap karttaan hyväntahtoisia muokkauksia, jotka heijastavat todellisuutta. Osa pelaajista on kuitenkin lisännyt virheellistä tietoa tai poistanut karttaelementtejä yrittäessään saada aikaan muutoksia pelin sisältöön. (Pokémon Go 2023.)

3.2.2 Microsoft Flight Simulator (2020)

Microsoft Flight Simulator on lentosimulaatiopeli, jota pelataan maapallon digitaalisessa kaksosessa, jonka luomiseksi Microsoft partneroitui Blackshark.ai startupin kanssa. Blackshark rekonstruoi 1.5 miljardia rakennusta Flight Simulator -peliä varten hyödyntäen Bing Maps -palvelun satelliittikuvia. Rekonstruktio toteutettiin tietokonenäkömenetelmin, joissa hyödynnettiin koneoppimistekniikoita. Rakennusten rekonstruktio vaati koneoppimisen käyttöä erityisesti tilanteissa, joissa osa rakennuksesta jää peittoon, mikä hankaloittaa niiden tunnistamista. Rakennuksien korkeuksien määrittelyssä käytettiin saatavilla olevia paikkatietoja, joiden puutteiden täydentämiseen tekoäly voi käyttää esimerkiksi rakennusten varjoja ja katon ulkoisia piirteitä. Todellisuudessa AI-järjestelmä on kuitenkin musta laatikko, jonka toiminnallisuudesta ei voida olla täysin varma. Kyseinen menetelmä tuottaa virheitä, ja ne ovat nähtävissä Flight Simulator -pelissä, mutta ottaen huomioon rakennusten valtavan määrän perinteinen laadunvarmistus ei ole mahdollista. (Lardinois 2020.) Lentosimulaation luonteen vuoksi ympäristöä tarkkaillaan suurimman osan ajasta kaukaa ja interaktio luodun maailman kanssa on minimaalista, joten ympäristön pienet virheet eivät korostu tavallisessa pelisessiossa.

Pilviteknologia mahdollistaa Microsoft Flight Simulator -pelin vaatimien valtavien datamäärien prosessoinnin. Pilvipalvelujen pohjana peli hyödyntää Azure-arkkitehtuuria. Peli vähentää riippuvuutta paikallisen tietokoneen suorituskyvystä siirtämällä suuren osan prosessoinnista pilveen. Täten tarjottu kokemus ylittää käyttäjän tietokoneen rajoitukset, jonka seurauksena saadaan luotua korkealaatuinen 3D-maailma, jossa on reaaliaikaiset olosuhteet. (Microsoft 2021.)

4 KARTTA- JA KORKEUSDATA

Kartta- ja korkeusdata toimii perustana, jonka pohjalta koko pelimaailma luodaan. Korkeusdata muodostaa perustan maaston kolmiulotteiseen esitykseen. Karttadataa puolestaan hyödynnetään rakennusten, teiden ja kasvuston visualisoinnissa. Yhdistämällä kartta- ja korkeusdata 3D-visualisoinnin luomiseksi saavutetaan huomattavasti havainnollisempi kuvaus ympäristöstä kuin karttatai korkeusdatalla itsenäisesti toteutettuna.

Tässä luvussa tarkastellaan valittuja datalähteitä: OpenStreetMap ja OpenTopography. Lisäksi analysoidaan valittujen datalähteiden ominaisuuksia ja niiden soveltuvuutta 3D-visualisoinnin luomiseen ja tutkitaan prosessia, jolla lähteiden dataa täydennetään ennen pelimaailman luomista.

4.1 Datalähteet

Kriittinen osa teknistä toteutusta on käytettävä datalähde. Sen valinnalla on valtava merkitys luodun 3D-mallin laatuun. Lähteen datan kuuluu kattaa suuri osa maapallosta ja samalla olla riittävän laadukasta ja kattavaa paikallisesti. Näin pystytään luomaan kolmiulotteinen visualisointi lähes minkä tahansa oikean maailman sijainnin pohjalta. Datalähteen valinnassa kuuluu myös huomioida saatavilla olevat ohjelmointirajapinnat sekä niiden käyttörajoitteet ja käyttöön liittyvät kustannukset. Datasettien laajuuden vuoksi data tulee sisältämään puutteita lähteestä riippumatta. Ne on huomioitava pelimaailman generoinnissa, jotta lopputulos olisi mahdollisimman hyödyllinen pelinkehityksen tarpeisiin.

4.1.1 OpenStreetMap

OpenStreetMap (OSM) Foundation on kansainvälinen, voittoa tavoittelematon, demokraattinen organisaatio, jonka tehtävänä on tukea OSM-projektia, ylläpitää ja suojella OSM-tietokantaa sekä tuoda se kaikkien saataville (OpenStreetMap Foundation 2023). OSM-projekti on kasvanut huomattavasti yli alkuperäisestä tavoitteestaan kartoittaa teitä. Sen lopullisena tavoitteena on saada kirja kaikista planeetan maantieteellisistä piirteistä. Fyysisen maantieteen li-

säksi projekti kartoittaa abstraktimpia ideoita kuten hallinnollisia rajoja, maankäytön yksityiskohtia ja bussireittejä. Projekti hyödyntää joukkoistamista karttadatan keruussa, eli datankeruuprosessi ulkoistetaan OSM-yhteisölle. (Bennett 2010, 27.) Yli 10 miljoonaa ihmistä on rekisteröitynyt OSM-palveluun 2023 vuoteen mennessä, mikä tekee siitä yhden menestyneimmistä avoimen lähdekoodin, joukkoistetuista geokartoista (HeiGIT 2023).

OSM-data on saatavilla ilmaisella ja avoimella lisenssillä (OpenStreetMap Foundation 2023). Teknisessä toteutuksessa OSM-data integroidaan Unity-pelimoottoriin Overpass-ohjelmointirajapinnan kautta. Kyseistä ohjelmointirajapintaa käytetään pääasiassa, koska se sallii suurten datamäärien lataamisen päivittäin. Käyttörajoitteena Overpass odottaa käyttäjien lähettävän enintään noin kymmenentuhatta kutsua päivässä ja pitävän latausmääränsä alle gigatavun päivässä. Suuremman kysynnän käsittelemiseksi suositellaan oman instanssin luomista, mikä on pyritty tekemään mahdollisimman helpoksi. (Commons s.a.) Se ei ole kuitenkaan tarpeen opinnäytetyön tavoitteen kannalta.

4.1.2 OpenTopography

OpenTopography on Yhdysvaltain kansallisen tiedesäätiön (National Science Foundation, NSF) rahoittama tietopalvelu, joka pyrkii tasa-arvoistamaan korkearesoluutioisen maastotiedon saannin. OpenTopography toimii ensisijaisesti ilmaisena ja julkisena topografisena tietoarkistona, jonka tavoite kokonaisvaltaisesta tietoarkistosta on saavutettu vaihtelevalla menestyksellä riippuen sijainnista. Kehittyneimmissä maissa, kuten Yhdysvalloissa ja Euroopan unionin jäsenvaltioissa, on topografisten tietojen keruutoimia, jotka on suunnattu säännölliseen korkearesoluutioisen datan keruuseen. Muilla alueilla ei välttämättä kuitenkaan ole resursseja, joita osoittaa korkearesoluutioisen paikkatiedon keruuseen. Tämän vuoksi kyseiset alueet ovat heikommin edustettuina. (O'Donohue 2021.)

Pelkkä raakadata on harvoin käyttäjän tavoite. Siksi OpenTopography tarjoaa reaaliaikaisia analyysityökaluja käyttäjien tarpeisiin. Nämä työkalut on optimoitu selainkäyttöä varten, mikä mahdollistaa San Diego Supercomputing Centerin omistamien resurssien ja prosessointitehon hyödyntämisen (O'Donohue 2021). Lisäksi OpenTopography on kehittänyt ohjelmistoja, jotka

mahdollistavat muun muassa digitaalisen maastomallin laskennan lidar-pistepilvitiedon perusteella sekä suhteellisen korkeusmallin luomisen normalisoimalla sen joen korkeuteen (OpenTopography Open… s.a.). Kyseiset ohjelmistot julkaistaan pääsääntöisesti avoimen lähdekoodin lisenssillä (Terms of Use 2022).

OpenTopographyn tarjoamat tiedot ovat vapaita kaikista tekijänoikeusrajoituksista ja täysin vapaasti saatavilla sekä ei-kaupalliseen että kaupalliseen käyttöön. Tietyillä tietokokonaisuuksilla voi kuitenkin olla erityisehtoja kuten CC BY 4.0 -lisenssin mukaisia vaatimuksia. (Terms of Use 2022). OpenTopography API puolestaan edellyttää API-avainta rajapintakutsujen toteuttamiseksi. Ilmaisella API-avaimella käyttäjällä on mahdollisuus tehdä enintään 500 kutsua 24 tunnin aikana. Laajempaa tai kaupallista käyttöä varten tarvitaan Enterprise API-avain, jonka hinnoittelutiedot eivät ole julkisesti saatavilla (OpenTopography for Developers s.a.) Ilmainen API-avain soveltuu kuitenkin ohjelmiston kehitykseen.

4.2 Tietomalli

Tietomalli on abstrakti esitys tietoelementeistä ja niiden välisistä suhteista. Tietomallit voidaan jakaa käsitteelliseen-, loogiseen- ja fyysiseen tietomalliin. Käsitteellinen tietomalli (Conceptual Data Model, CDM) luo kattavan ja abstraktin käsityksen datasta ja havainnollistaa, miten eri ominaisuudet ovat keskenään yhteydessä. Sen ymmärtäminen ei edellytä syvällistä teknistä osaamista. Looginen tietomalli (Logical Data Model, LDM) tarkentaa käsitteellisen tietomallin konseptia. Se määrittelee suhteet täydellisesti ja syventää keskeisten elementtien rakennetta ja yksityiskohtia. Looginen tietomalli sisältää kunkin entiteetin attribuutit, entiteettien väliset suhteet ja näiden suhteiden liitynnät. Fyysinen tietomalli (Physical Data Model, PDM) kuvaa, kuinka malli rakennetaan tietokantaan. Se kertoo tietomallin teknologia- ja tietokantakohtaisesta toteutuksesta ja on viimeinen askel loogisen tietomallin muuntamisessa toimivaksi tietokannaksi. Fyysinen tietomalli sisältää kaikki tarvittavat fyysiset tiedot tietokannan rakentamiseen. (Rivera 2023.) Tässä luvussa tarkastellaan käytettävän OpenStreetMap- ja OpenTopography -datan loogisia tietomalleja. Sen kautta saadaan luotua riittävä ymmärrys datan rakenteesta, jotta voidaan integroida se omaan järjestelmään.

OpenStreetMap käyttää omaa tietomalliaan, joka poikkeaa muista paikkatietojärjestelmistä. Tietomalli käyttää vain kolmea primitiivistä tyyppiä: node, way ja relation, joista voi nähdä esimerkit XML-muodossa kuvassa 3. Node kuvailee yksittäistä sijaintia. Se on ainoa primitiivi, joka sisältää sijaintitietoa. Muiden primitiivien sijaintieto pohjautuu täysin node-primitiiveihin. Way on järjestelty lista node-primitiivejä. Se kuvailee lineaarisia ominaisuuksia, kuten teitä ja jokia, sekä suljettuja muotoja, kuten rakennuksia ja järviä. Suljetun way-primitiivin täytyy muodostaa yksinkertainen monikulmio. Jos vaaditaan monimutkaisempaa muotoa, se kuuluu tehdä relation-primitiivillä. Relation on lista primitiivejä mukaan lukien muita relation-primitiivejä. Sen tarkoitus on mahdollistaa monimutkaisempien muotojen kuvailemisen linkittämällä useampi primitiivi toisiinsa. Kaikki primitiivit ovat osittain itsenäisiä. Tämä johtuu siitä, että primitiivit eivät sisällä toisia primitiivejä vaan pelkästään toisen primitiivin ID:n. Täten way-primitiivi ei muutu siihen kuuluvien node-primitiivin muuttuessa. (Bennett 2010, 110, 112–113, 120–121.)

```
<relation id="1315128" version="3" timestamp="2021-07-05T11:34:27Z"
          changeset="107437036" uid="75947" user="latsku">
  <member type="way" ref="88929426" role="outer"/>
  <member type="way" ref="88929469" role="inner"/>
  <member type="way" ref="88929433" role="inner"/>
  <tag k="building" v="apartments"/>
  <tag k="type" v="multipolygon"/>
</relation>

<way id="88929469" version="1" timestamp="2010-12-10T22:03:05Z"
     changeset="6610653" uid="7458" user="Geosapiens">
  <nd ref="1032151977"/>
  <nd ref="1032152426"/>
  <nd ref="1032151550"/>
  <nd ref="1032151696"/>
  <nd ref="1032151977"/>
</way>

<node id="1032151977" lat="60.4646360" lon="26.9399021" version="1"
      timestamp="2010-12-10T22:02:47Z" changeset="6610653" uid="7458" user="Geosapiens"/>
```

Kuva 3. Esimerkki relation-, way- ja node-primitiiveistä XML-muodossa

Primitiivien tukena OpenStreetMap käyttää vapaamuotoista tägi-järjestelmää. Primitiiveillä voidaan kuvata kaikkia kartan fyysisiä ominaisuuksia. Tägit kertoo puolestaan avain-arvo-parilla, mitä kyseiset ominaisuudet kartassa kuvaavat. Esimerkiksi kuvan 3 relation-primitiivin tägin avaimen ”building” arvo ”apartments” kertoo, että relation kuvaa huoneistorakennusta. Primitiivit voivat sisältää mielivaltaisen määrän tägejä. Tägien sisältöä ei säännellä vaan niiden sisältö on täysin OpenStreetMap-yhteisön harkinnan varassa. On kuitenkin

muodostunut yleisiä tägejä, joita odotetaan käytettävän sen sijaan, että luodaan oma tägi olemassa olevalle ominaisuudelle. Tägejä ei myöskään kuuluisi luoda renderöintiä varten. Tämä tarkoittaa, että tägin luomisen tulisi perustua varsinaisiin tietoihin ja niiden asianmukaiseen kuvailuun, eikä pelkästään ulkoisen visuaalisen esityksen optimointiin renderöintiprosessissa. Esimerkiksi aluetta ei pitäisi merkata metsäksi vain sen takia, että se näkyisi vihreänä kartassa. Nämä ovat kuitenkin vain yhteisön luomia käytänteitä, ja mikään ei estä yksittäistä käyttäjää toimimaan niitä vastoin. Tägit ja niiden vapaus on kuitenkin tietomallin vahvuus, sillä se mahdollistaa monimuotoisen tiedon ilmaisemisen kartassa. Kuvaillun tietomallin avulla voidaan määritellä tarkasti käytännössä kaikki maantieteelliset ominaisuudet. (Bennett 2010, 110, 124, 127–128.)

OpenTopography mahdollistaa korkeusdatan lataamisen OpenTopography Global DEM API:n kautta kolmena formaattina: GeoTiff, ASCII-ruudukko ja ERDAS Imagine. Teknisessä toteutuksessa käytetään ASCII-ruudukko-formaattia ensisijaisesti sen selkeän ja helposti luettavan rakenteen vuoksi, mikä mahdollistaa manuaalisen tarkastelun ja analysoinnin vaivattomasti. ASCII-ruudukko koostuu otsikkotiedoista ja ruudukosta arvoja. Otsikkotiedot ovat avain-arvo-pareja. Ne sisältävät ruudukon leveyden ja korkeuden ruuduissa, alavasemman ruudun keskipisteen- tai alavasemman kulman sijainnin, ruudun koon ja puuttuvan arvon indikaattorin. Formaatin itse datakomponentti on ruudukko korkeusarvoja, jotka ovat joko kokonais- tai desimaalilukuja. Ruudukossa arvot erotetaan toisistaan erotin merkillä. Ruudukossa ei välttämättä tarvita erillistä rivinvaihtoa, sillä otsikkotiedot määrittävät rivin pituuden erikseen. (Esri ASCII raster format 2021.)

4.3 Datan laatu ja määrä

Suuri osa 3D-visualisointiin tarvittavasta datasta ei näy millään tavalla perinteisessä 2D-kartassa. Täten kyseinen tieto on vähäistä dataseteissä, joita käytetään ensisijaisesti 2D-karttojen visualisointiin. Rakennuksen visualisointiin kaksiulotteisessa kartassa vaaditaan vain sen ulkoseinien rajaama muoto. Kolmiulotteinen visualisointi puolestaan vaatii muodon lisäksi rakennuksen korkeuden. Vain 2,81 prosenttia OSM-karttadatan rakennuksista sisältää rakennuksen korkeuden. Korkeusdataa tukeva kerrosten määrä on määritelty

4,94 prosentissa rakennuksista, mikä on yhä suhteellisen harvassa rakennuksessa huomioiden tavoitteen luoda jokainen karttadatan rakennus pelimaailmassa. (Building s.a.) Teiden kolmiulotteisessa visualisoinnissa puolestaan tarvitaan tien reitin lisäksi sen leveys. Tien tarkka leveys on saatavilla vain 1,06 prosentissa teistä. Sen sijaan kaistojen lukumäärä on määritelty jopa 6,33 prosentissa teistä. (Highway s.a.) Se toimii täten hyvänä tukena tien leveyden määrittämisessä. Siitä huolimatta suurin osa teiden leveyksistä on määriteltävä jollakin toisella metodilla.

OpenStreetMap-palvelin tekee hyvin vähän tietojen tarkistusta. Niin kauan kuin toimitetut tiedot ovat johdonmukaisia, niitä ei hylätä. (Bennett 2010, 111.) Datan tarkistaminen on siis tehtävä itse toteutuksessa 3D-visualisoinnin laadun takaamiseksi. Poikkeuksia on kuitenkin lukemattomia erilaisia, minkä vuoksi niiden kaikkien huomioiminen ja korjaaminen on hyvin vaikeaa. Sen sijaan luodaan vain ne osat maailmasta, jotka noudattavat OpenStreetMap-datan hyviä käytänteitä. Näin saadaan ylläpidettyä lopullisen mallin laatu datan poikkeuksista huolimatta. Datan sisältöön ja sen todenmukaisuuteen ei puolestaan oteta mitään kantaa.

Tavoite tarkkuus luodulle maastolle on 1 m resoluutio, mikä tarkoittaa, että korkeuskartta sisältää korkeusarvoja metrin välein. OpenTopography Global datasets -ohjelmointirajapinnan kautta saatu satelliittidata on kuitenkin maailmanlaajuisesti vain 30 m resoluutioista (OpenTopography for Developers s.a.). OpenTopography palvelun kautta on saatavilla tarkempaa paikallista dataa, mutta käytetyn ohjelmointirajapinnan kautta pääsee käsiksi vain laaja-alaisiin satelliittidatasetteihin. Valittua lähestymistapaa hyödynnetään datan tarkkuudesta huolimatta, sillä se täyttää tavoitteen maailmanlaajuisesta datasta.

4.4 Datan täydentäminen

Ennen pelimaailman generointia on käsiteltävä kartta- ja korkeusdatan puutteellisuus. Ongelmaan on useita eri lähestymistapoja. Yksi vaihtoehto on jättää karttaelementit, joissa on puutteellista dataa kokonaan luomatta, kuten tehdään OpenStreetMap-datan hyviä käytänteitä rikkovien tapausten kanssa.

3D-visualisointiin tarvittavaa puuttuvaa dataa on kuitenkin liikaa tähän ratkaisuun. Sen sijaan voidaan täydentää puuttuvaa dataa tiedossa olevan datan perusteella. Tämä tehdään käyttämällä dataa muista komponenteista, jotka muistuttavat eniten komponenttia, johon dataa pyritään täydentämään. Tämän lähestymistavan laatu on sitä heikompi mitä vähemmän tunnettua dataa on olemassa. Datan täydentämiseen voidaan hyödyntää myös muuta kyseisen komponentin dataa, josta voidaan päätellä haluttu data, kuten rakennuksen korkeuden määrittely kerrosten määrän perusteella. Lopuksi voidaan käyttää myös täysin keksittyä dataa, kuten satunnaisia arvoja tai itse määriteltyjä vakioarvoja.

4.4.1 Rakennuksen korkeus

Rakennuksen kolmiulotteisen visualisoinnin edellyttämän korkeusdatan vajavaisuuden vuoksi on välttämätöntä täydentää puuttuvia tietoja. Ensisijaisesti puuttuva rakennuksen korkeus lasketaan kerrosmäärästä käyttäen oletusarvoa kerroksen korkeudesta. Siinä tapauksessa, että rakennuksen kerrosten lukumäärä ei ole tiedossa, rakennuksen korkeus määritellään muiden rakennuksien perusteella. Kohderakennuksen ympäriltä etsitään ne rakennukset, joilla on olemassa korkeusarvo ja lasketaan kyseisille rakennuksille painoarvot. Painoarvojen perusteella valitaan rakennus, jonka korkeusarvoa käytetään kohderakennuksen korkeusarvona.

Käytettävää korkeutta valittaessa painotetaan rakennuksia neljällä perusteella:

- etäisyys kohderakennusta
- pinta-ala
- rakennustyyppi
- korkeusdatan alkuperäisyys

Näiden neljän painoarvon valinnalla on tavoitteena saada aikaan alueita, joissa samanlaiset rakennukset ovat saman korkuisia. Siten saadaan luotua uskottavan näköisiä naapurustoja. Samalla pyritään myös mahdollisimman kattavaan korkeusdatan täydennykseen.

Pinta-alaa ja rakennustyyppiä käytetään rakennuksien yhteneväisyyden arviointiin. Mitä paremmin nämä arvot vastaavat tarkkailtavan rakennuksen arvoja, sitä todennäköisemmin korkeusdata määritellään kyseisen rakennuksen

perusteella. Pinta-alan laskemisen vaatimus on sama kuin rakennuksen kaksiulotteisen visualisoinnin minimivaatimus, sillä pinta-ala voidaan laskea suoraan ulkoseinien rajaamasta muodosta. Pinta-alan saa siis selville kaikista rakennuksista, jotka luodaan. Pinta-alan lisäksi rakennustyyppiä käytetään rakennusten yhteneväisyyden arvioinnissa. Rakennustyyppi kuvaa rakennuksen käyttötarkoitusta ja se saadaan suoraan karttadatasta. Rakennuksista 19,85 prosentissa on määritelty rakennuksen tyyppi (Building s.a.). Rakennustyyppi ei ole siis saatavilla läheskään yhtä useasta rakennuksesta kuin pinta-ala. Se ilmenee kuitenkin merkittävästi useammassa rakennuksessa kuin täydennettävä korkeusdata. Rakennustyyppi toimii täten toissijaisena tietona yhteneväisyyden arvioinnissa, missä pinta-ala toimii pääarviointiperusteena.

Etäisyys kohderakennuksesta ja korkeusdatan alkuperäisyys hallitsee korkeusdatan leviämistä. Kohderakennusta lähempiä rakennuksia painotetaan yli kauempien. Rakennusten etäisyydellä on myös asetettu maksimiraja, jota kauempana olevia rakennuksia ei oteta huomioon korkeusdataa täydentäessä. Tämä on tehty ensisijaisesti korkeusdatan täydentämisprosessin nopeuttamiseksi, sillä se vähentää vertailtavien kohteiden määrää huomattavasti. Maksimirajan vaikutus lopputulokseen on minimaalinen, sillä kaukana olevat rakennukset saavat joka tapauksessa heikon painotuksen etäisyytensä vuoksi.

Korkeusdatan kattavuuden parantamiseksi sallitaan sen leviäminen useamman viittauksen kautta rakennuksesta toiseen. Tätä prosessia hallitaan määrittelemällä rakennuksen korkeusdatalle sen alkuperäisyys. Korkeusdatan alkuperäisyyttä kuvataan sillä, kuinka monen viittauksen kautta se on haettu. Rakennuksia, joilla on alkuperäisempi korkeusdata, painotetaan yli muiden. Sen lisäksi viittausten määrälle on asetettu maksimiraja, jota useamman viittauksen kautta dataa ei voi saada.

Lähellä olevan rakennuksen mainitut neljä painoarvoa pitää saada yhdessä riittävän hyvän pisteytyksen, jotta rakennuksen korkeutta käytetään. Jos korkeimman painoarvon saanut rakennus ei ylitä vähimmäispainoarvoa, sitä ei oteta huomioon korkeustiedon täydentämisessä. Sen sijaan rakennukselle käytetään oletuskorkeutta.

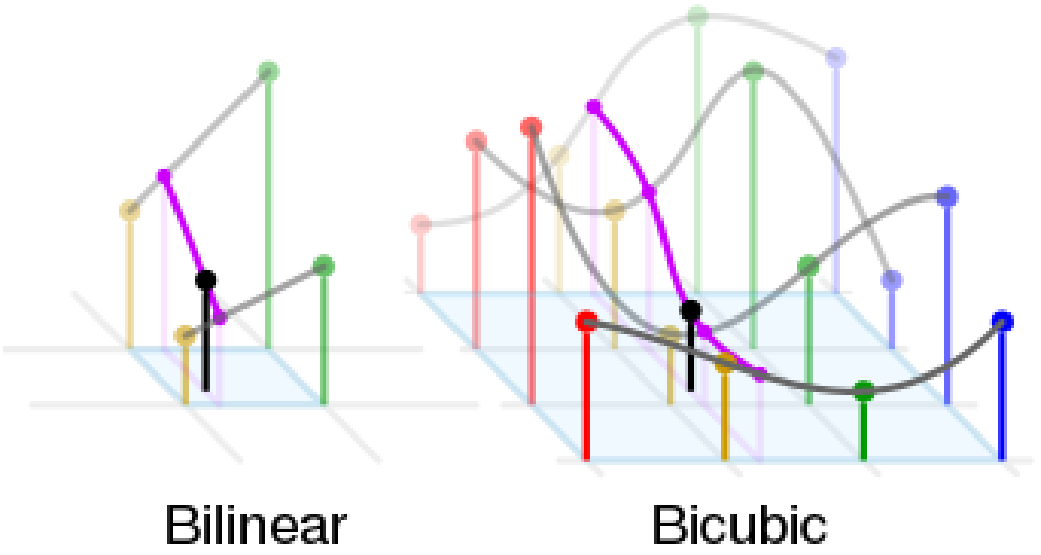
4.4.2 Tien leveys

Tien luomista varten on täydennettävä teiden puutteelliset leveystiedot, kuten tehdään rakennuksien korkeuksien suhteen. Rakennuksissa käytetty datan täydennysmetodi ei kuitenkaan sovellu teiden datan täydentämiseen. Leveistä teistä risteävät kapeammat tiet ja teiden leveyksien muutokset ovat hyvin tavanomaisia. Sen lisäksi liian leveä tie voi aiheuttaa ei-toivottua päällekkäisyyttä muiden karttaelementtien kanssa. Tämän vuoksi tien leveyttä arvioidessa suositaan kapeampaa vaihtoehtoa yleisemmän sijaan.

Ensisijaisesti puuttuva tien leveys lasketaan kaistojen määrästä hyödyntäen oletusarvoa kaistan leveydestä. Mikäli kaistojen lukumäärä ei ole tiedossa, määritellään tien leveys tietyypin perusteella. Teissä tyyppitieto on määritelty huomattavasti useammin kuin rakennuksissa, mikä on odotettavissa huomioiden tietyypin merkityksen perinteisissä navigointisovelluksissa. Vain 0,02 prosenttia teiden tyypeistä on luokiteltu tuntemattomaksi (Highway s.a.). Olemassa olevat tietyypit ovat yleistasoisia, mutta auttavat silti tien leveyden määrittämisessä. Yleisimmille tietyypeille annetaan niiden määritelmästä johdettu leveys, jota käytetään tien leveyden määrittämiseen. Jos leveyttä ei saada täydennettyä tietyypin perusteella, käytetään tielle oletusleveyttä.

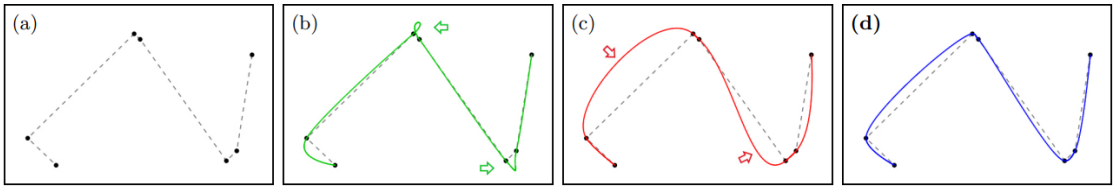
4.4.3 Korkeuskartan interpolointi

Korkeusdatan relatiivisen alhaisen resoluution vuoksi sen dataa täydennetään ennen maaston generoimista. Tämä tehdään laskemalla uusia arvoja olemassa olevien arvojen välille. Kyseistä prosessia kutsutaan interpoloinniksi. Yksi suoraviivaisimmista menetelmistä toteuttaa maaston interpolointi on bilineaarinen interpolointi. Siinä puuttuvat arvot lasketaan lineaarisena janana olemassa olevien arvojen välille kuvassa 4 näkyvällä tavalla. Kyseinen interpolointimetodi edustaa datan kuvaamaa maastoa todenmukaisesti, mutta tuottaa epäluonnollisen kulmikkaan maaston.



Kuva 4. Kahden kaksiulotteisen interpolointi metodin vertailu (Bicubic interpolation s.a.)

Bicubic-interpolointi saa aikaan bilineaarista interpolointia pyöristetymmän maaston hyödyntämällä spline-interpolointia, mitä havainnollistaa kuva 4. Menetelmässä keskimmäisen välin pisteet koostavat polynomin, jonka määrittelee neljä peräkkäistä datapistettä. Yhdessä välit muodostavat paloittain määritellyn polynomin. Tällaista jatkuvaa ja derivoituvaa paloittain määriteltyä polynomia kutsutaan nimellä spline. Spline-käyriä on monia erilaisia eri tarkoituksiin. Interpoloinnin toteutukseen on valittu Catmull–Rom-spline. Se täyttää ohjelmiston tarpeet maaston interpoloinnin suhteen. Catmull–Rom-spline ei vaadi ylimääräisiä ohjauspisteitä datapisteiden lisäksi, se kulkee tiukasti datapisteiden kautta ja yksittäinen datapiste vaikuttaa spline-käyrän muotoon vain paikallisesti. (Yuksel ym. 2009.)



Kuva 5. Kolme eri parametrointia Catmull-Rom splinelle samalla ohjausmonikulmiolla (a), uniform (b), chordal (c) ja centripetal (d) (Yuksel ym. 2009)

Catmull–Rom-spline-käyrän muotoon vaikuttaa merkittävästi sen parametrisointi. Yksi suosituimmista Catmull–Rom-spline-käyrän parametrisoinneista on uniform-parametrisointi sen heikkouksista huolimatta. Kyseinen parametrisointi voi johtaa ei-toivottuihin artefakteihin, kuten kuvan 5 b-kohdassa näky-

viin kohoumiin ja itseleikkauksiin. Lisäksi käyrän etäisyys ohjausmonikulmiosta on rajoittamaton, mikä tekee käyrästä vaikeasti hallittavan. Chordal- ja centripetal-parametrisointi puolestaan tuottaa visuaalisesti miellyttävämpiä tuloksia ilman edellä mainittuja artefakteja, kuten kuvista 5c ja 5d voidaan havaita. Sen lisäksi centripetal-parametrisointi johtaa spline-käyrään, joka on lähempänä ohjausmonikulmiota kuin uniform- ja chordal-parametrisointi, kun otetaan huomioon koko käyrä. (Yuksel ym. 2009.) Täten centripetal-parametrisointi sopii parhaiten maaston interpolointiin.

```
int segmentCount = (int)Mathf.Ceil((float)scaledHeight / rowsPerSegment);

Parallel.For(0, segmentCount, (currentSegment) =>
{
    int start = currentSegment * rowsPerSegment;
    int end = Mathf.Min(start + rowsPerSegment, scaledHeight);

    for (int x = start; x < end; x++)
    {
        ...

        for (int z = 0; z < scaledWidth; z++)...
    }
});
```

Kuva 6. Rinnakkaislaskennan hyödyntäminen maaston interpoloinnissa

Maaston interpolointi vaatii massiivisen korkeusarvomäärän laskemisen. Jo pelkkä yhden neliökilometrin korkeuskartta koostuu miljoonasta korkeusarvosta. Hyödyntämällä rinnakkaislaskentaa saadaan nopeutettua näiden suurien iteraatiomäärien prosessointia. Perinteisesti maasto interpoloidaan ensin x-akselin suuntaisesti ja sen jälkeen z-akselin suuntaisesti. Z-akselin suuntaisesti lasketut arvot ovat riippuvaisia vain kyseisessä z-koordinaatissa olevista korkeusarvoista. Maasto voidaan siis jakaa z-akselin suuntaisiin riveihin, jotka lasketaan toisistaan riippumattomina. Täten maaston interpolointiin voidaan hyödyntää rinnakkaislaskentaa. Kuvassa 6 näkyy, kuinka rinnakkaislaskenta otetaan käyttöön koodissa. Ensin lasketaan, kuinka moneen osioon korkeuskartan rivit jaetaan. Osioiden läpi iteroidaan Parallel For -silmukassa, joka poikkeaa tavallisesta for-silmukasta niin, että se käsittelee useampaa iteraatiota samanaikaisesti. Jokaiselle osiolle määritellään väli rivejä laskemalla välin alku ja loppu. Rivit käydään läpi ulkoisessa for-silmukassa ja sisäisessä silmukassa lasketaan rivin jokaiselle koordinaatille korkeusarvo.

5 PELIMAAILMAN GENEROINTI

Generoitava pelimaailma koostuu neljästä pääosasta: maasto, rakennukset, tiet ja kasvusto. Pelimaailman osat on tehty toisistaan riippumattomiksi, jotta ne voidaan luoda erillään toisistaan. Vain samanaikaisesti luotuna ne sovitetaan yhteen hyödyntämällä toistensa dataa. Pelimaailma luodaan ohjelmallisesti kartta- ja korkeusdatan pohjalta. Sen luomiseen käytetään Unity-pelimoottorin Terrain-järjestelmää, ohjemallisesti luotuja polygonimalleja ja instanssoituja prefab-objekteja.

5.1 Maasto

Maasto luodaan ensimmäisenä koko pelimaailmasta. Sen luomiseen hyödynnetään Unity-pelimoottorin Terrain-järjestelmää. Maastoalue jaetaan ruuduksi, joka keskitetään pelimaailman origoon. Maastoaluetta keskitettäessä on huomioitava, että pyydetty ja ladattu korkeuskartta eivät täsmää täysin. Keskittäminen tulee täten suorittaa ladatun datan perusteella. Maaston ruudun koon mukaan voidaan määrittää jokaisen ruudun rajat. Yksittäinen maaston ruutu koostuu yhdestä Terrain-komponentista, jonka koko, sijainti ja resoluutio määritellään sen rajojen ja koko maastoalueen maksimikorkeuden perusteella. Ruudun rajojen avulla Terrain-komponentille määritellään kyseistä aluetta vastaava korkeuskartta koko maastoalueen korkeuskartan pohjalta. Lopuksi Terrain-komponentille määritellään naapuroivat Terrain-komponentit, mikä takaa maaston saumattomuuden.

Karttadatan pohjalta luodut pelimaailman komponentit käyttävät maastoa korkeuden saamiseksi. Tämä on toteutettu laskemalla maaston korkeus pelimaailman sijainnin pohjalta. Kyseisellä metodilla määritellään jokaiselle karttadatan node-komponentille korkeus. Tämä tarkoittaa myös, että karttaelementit eivät saa korkeustietoa, jos maastoa ei luoda samassa yhteydessä. Siinä tapauksessa pelimaailma luodaan tasaiselle tasolle.

5.2 Rakennukset

Rakennukset koostuvat yhdestä tai useammasta osasta, jotka luodaan lähes itsenäisesti toisistaan. Kukin osa rakennuksesta perustuu pohjan muotoon ja

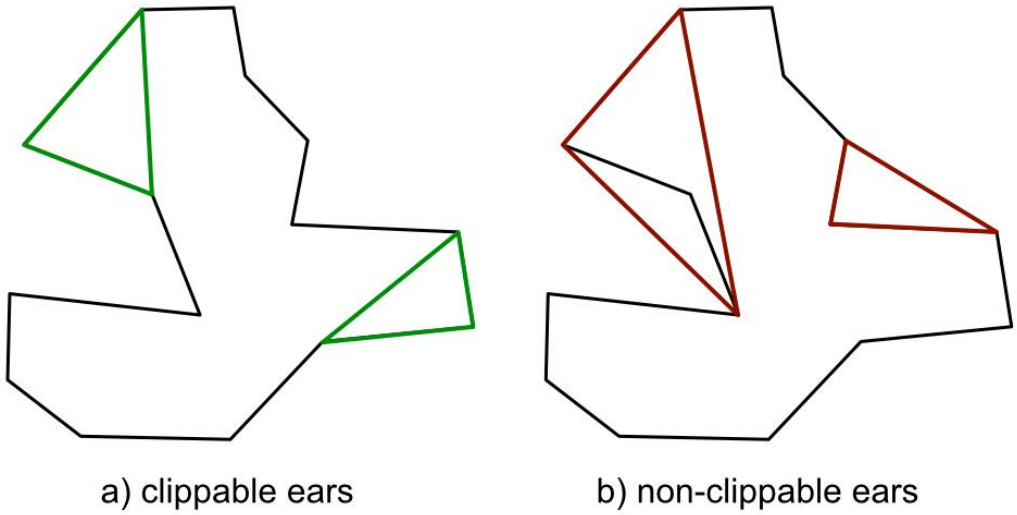
seinien korkeuteen. Nämä osat muodostavat yhdessä yksinkertaisen 3D-visualisaation rakennuksesta, jossa on pystysuorat seinät ja tasainen katto.

Rakennuksen nollakorkeus määritellään aikaisemmin luodun maaston perusteella. Rakennuksen alueella on vaihteleva maastonkorkeus, joten rakennukselle määritellään yhteinen nollakorkeus, jonka perustella korkeudet lasketaan. Maaston korkein kohta rakennuksen node-komponenttien sijainneista toimii rakennuksen yhteisenä nollakorkeutena. Kuitenkin maan tasolla olevat rakennuksen osat ulottuvat alimpaan korkeuteen asti, mikä varmistaa niiden yhteyden maastoon joka puolella rakennusta. Tietyille rakennuksen osille on määritelty minimikorkeus karttadatassa, mikä määrittää kyseisen osan pohjan korkeuden suhteessa maanpintaan. Korkeuden laskeminen aloitetaan rakennuksen nollakorkeudesta. Maanpinnan yläpuolelta alkava rakennus vaatii katon lisäksi myös pohjan, sillä sen pohja voi olla osittain ilman tukea. Muutoin se voidaan luoda samalla tavalla kuin muut rakennuksen osat.

Rakennuksen osan seinä muodostetaan suorakulmaisista elementeistä rakennuksen osan korkuisena ja sen muodon määrittävien node-komponenttien välille. Riippumatta node-komponenttien välisestä etäisyydestä, suorakulmio jaetaan kahtia, muodostaen kaksi kolmiota. Lasketuista kolmioista koostuu seinän polygonimalli (mesh), joka määrittelee seinän kolmiulotteisen muodon. Seinien päälle luodaan tasakatto. Tasakaton luomiseksi on määriteltävä kolmiot, joista rakennuksen osan muoto koostuu. Samalla tiedolla voidaan luoda myös pohja rakennuksen osille, jotka alkavat maanpinnan yläpuolelta. Kaikki rakennuksien ulkoseinien rajaamat muodot ovat yksinkertaisia monikulmioita, eli seinien kesken ei ole yhtään leikkauspisteitä. Täten rakennuksen muodon kolmioihin jakaminen voidaan tehdä ear clipping -algoritmilla.

Ear clipping -algoritmissa monikulmiosta leikataan niin sanottuja ”korvia”. Korvat ovat kolmen peräkkäisen monikulmion kärjen muodostamia kolmioita. Korvassa keskimmäisen kärki on kupera, jolloin kyseisessä kärjessä sisäkulma on alle 180 astetta. Sen lisäksi ensimmäisen ja viimeisen kärjen välisen janan muodostama monikulmion lävistäjä on kokonaan monikulmion sisäpuolella. Vähintään neljäkärkisessä yksinkertaisessa monikulmiossa ilmenee aina vähintään 2 ei-päällekkäistä korvaa. (Eberly 2002.) Kuvan 7 monikulmio a sisäl-

tää vihreällä kaksi esimerkkiä kolmioista, jotka täyttävät kaikki leikattavan korvan ehdot. Monikulmiossa b on punaisella kaksi korvaksi kelpaamatonta kolmiota. Vasemmanpuoleinen punainen kolmio ei koostu kolmesta peräkkäisestä kärjestä, minkä vuoksi kahden ensimmäisen kärjen välinen jana ei myöskään ole monikulmion sisäpuolella. Oikeanpuoleisen punaisen kolmion keskimmäinen kärki ei ole kupera, mikä aiheuttaa myös sen, että ensimmäisen ja viimeisen kärjen välinen jana ei ole monikulmion sisäpuolella.



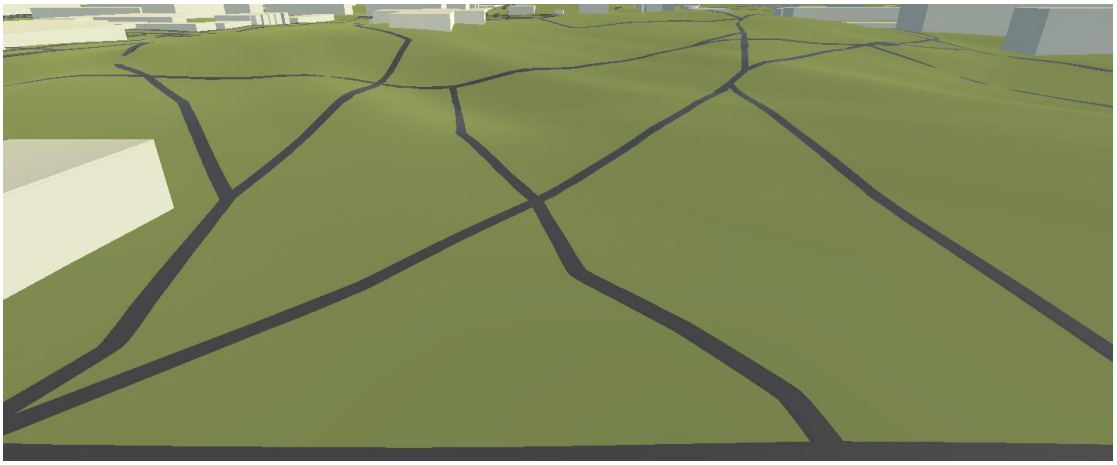
Kuva 7. Monikulmion leikattavat korvat (Neumüller 2021)

Monikulmion kärjistä tehdään myötäpäivään lista, jota käydään läpi tarkkaillen kolmen peräkkäisen kärjen muodostamia kolmioita. Jos kolmio täyttää kaikki korvan ehdot, se poistetaan monikulmiosta ja lisätään kolmioihin, joista polygonimalli koostetaan. Jäljelle jäävälle yksinkertaiselle monikulmiolle toistetaan sama toimenpide, kunnes koko monikulmio on jaettu kolmioiksi. (Eberly 2002.)

5.3 Tiet

Luotavan alueen karttadatassa tie koostuu aina koko way-komponentista riippumatta siitä, onko tie kokonaisuudessaan luotavalla alueella. Sen lisäksi tien node-komponenttien välimatkat vaihtelevat huomattavasti. Tämä ei ole käytännöllistä tien maastoon sovittamisen kannalta. Siksi teitä luodessa ne jaetaan lyhyisiin, tasapituisiin tieosuuksiin, jotka saadaan sovitettua maastoon tarkemmin. Sen avulla saadaan myös rajattua tien luominen tarkemmin alueen sisäpuolelle jättämällä alueen ulkopuolella olevat tieosuudet luomatta.

Tieosuuksien päädyille määritellään samassa yhteydessä korkeus ja rotaatio maaston perusteella. Tämä voi johtaa siihen, että tien laidat jäävät maan alle tapauksissa, joissa maaston korkeus muuttuu paljon lyhyellä matkalla. Kyseisessä tapauksessa tietä nostetaan ylöspäin uponneisuuden verran.



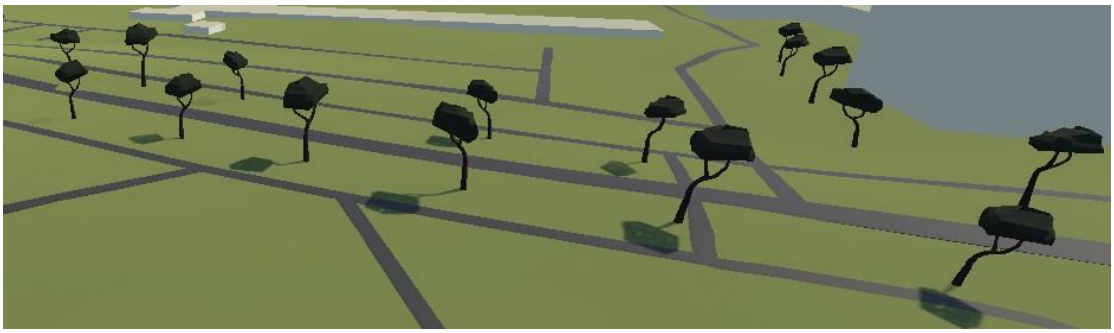
Kuva 8. Teiden mukautuminen maastoon

Tien itse polygonimalli luodaan spline mesh -työkalulla. Työkalu hyödyntää polygonimallin luomiseen Bezier-spline-käyrää. Sen luomista varten määritellään spline-käyrän ohjauspisteet tien reittiä kuvaavien sijaintien perusteella. Ohjauspisteet on määritelty siten, että tie seuraa karttadatassa määriteltyä reittiä hyvin läheisesti, jotta vältetään tarkoituksetonta päällekkäisyyttä muiden elementtien kanssa. Tämä johtaa kuvassa 8 näkyvään kulmikkaan oloiseen tiehen. Lopullinen tien kolmiulotteinen muoto määritellään kaksiulotteisella muodolla, josta polygonimalli luodaan ekstruusiona spline-käyrää pitkin. Kaksiulotteiseksi muodoksi on valittu tasakylkinen puolisuunnikas, jotta siirtymä maastosta tiehen on sulava.

5.4 Kasvusto

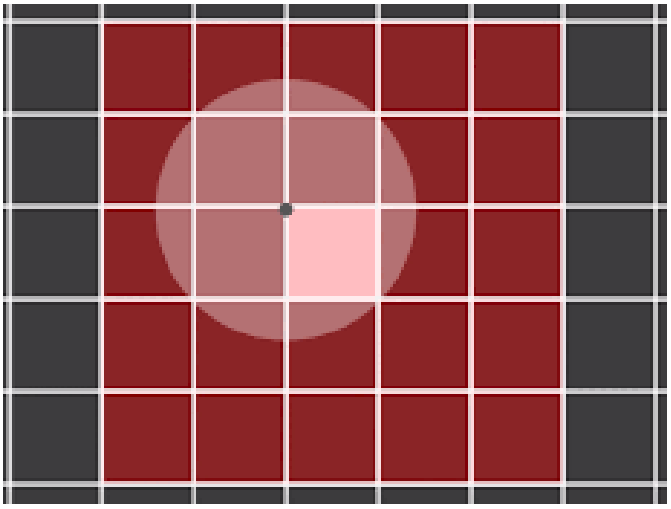
Ihmisperäisten rakennusten lisäksi karttadata sisältää kasvustoa. Kasvuston osalta teknisessä toteutuksessa keskitytään puustoon. Pelimaailmaan lisätään niin yksittäisiä puita kuin metsäalueita. Toisin kuin rakennuksissa ja teissä, puusto luodaan valmiista 3D-malleista sen sijaan, että luotaisiin puun polygonimalli ohjelmallisesti.

Yksittäisten puiden luomisprosessi on relatiivisen yksinkertainen. Valmis puuprefab lisätään datassa määriteltyyn sijaintiin maaston määrittämälle korkeudelle. Pelkät puiden sijainnit määrittelemällä puustot näyttävät kuitenkin epäluontevilta puiden ollessa kaikki samoin päin. Siksi puiden kokoa, korkeutta ja rotaatiota satunnaistetaan. Puulle määritellään sen pituus- ja leveyspiirin perusteella siemenluku, jota hyödynnetään satunnaisten elementtien alustamiseen. Siten yksittäiseen puuhun tehdyt muutokset pysyvät samoina luontikerrasta ja luodusta alueesta riippumatta. Kuvasta 9 voidaan havaita, kuinka puukohtaiset muutokset auttavat useamman puun kokonaisuuksia näyttämään paljon luontevammille, vaikka muutokset itsessään ovat suhteellisen pieniä.



Kuva 9. Ryhmä yksittäin lisättyjä puita satunnaistetuilla ominaisuuksilla

Metsien luomisessa puiden lisääminen tapahtuu samalla tavalla kuin yksittäisten puiden kanssa. Metsissä ei kuitenkaan ole puille määriteltyjä sijainteja, joita voisi hyödyntää puiden lisäämisessä. Sen sijaan metsälle on määritelty alue, jolla puita ilmenee. Itse puiden sijaintien määrittelyyn alueella käytetään Poisson disk sampling -algoritmia.



Kuva 10. Yksittäinen Poisson Disk -näyte (Hemalatha 2019)

Poisson Disk Sampling -algoritmissa aloitusnäytteen sijainti valitaan satunnaisesti koko näytealueelta painottamatta mitään sijaintia. Kuvassa 10 minimietäisyyttä näytteiden välillä havainnollistetaan ympyrällä, jonka säde määritellään yhtä suureksi kuin neliön muotoisen solun lävistäjä. Tämä takaa, että yksi solu sisältää enintään yhden näytteen. Näytteen ympäriltä valitaan ennalta määritelty määrä sijainteja satunnaisiin suuntiin ympyrän säteen r ja 2r väliseltä etäisyydeltä. Näyte lisätään valittuun sijaintiin, jos se ei ole liian lähellä toista näytettä tai näytealueen ulkopuolella. Tämä prosessi toistetaan jokaiselle uudelle luodulle näytteelle, kunnes uusia näytteitä ei saada enää luotua. (Hemalatha 2019.)

Kuva 11. Metsän puiden jakauma laskettuna Poisson Disk Sampling -algoritmilla

Pääsyyt Poisson Disk Sampling -algoritmin valinnalle ovat luodun näytejakauman tasaisuus ja tiheyden tarkka säätö. Kuvasta 11 voidaan havaita, kuinka Poisson Disk Sampling -algoritmi saa aikaan tasaisen, luonnollisen näköisen näytejakauman näytteenottoalueella. Jakauman tiheys on riippuvainen näytteiden välisestä minimietäisyydestä, mikä tekee jakauman tiheyden hallinnasta helppoa. (Hemalatha 2019.)

Koska metsäalueella voi ilmetä rakennuksia ja teitä, on puita lisättäessä tarkistettava, että puun viemä tila on tyhjä ennen kuin se lisätään sille määriteltyyn sijaintiin. Koska kasvusto luodaan viimeisenä, tarkistus voidaan tehdä

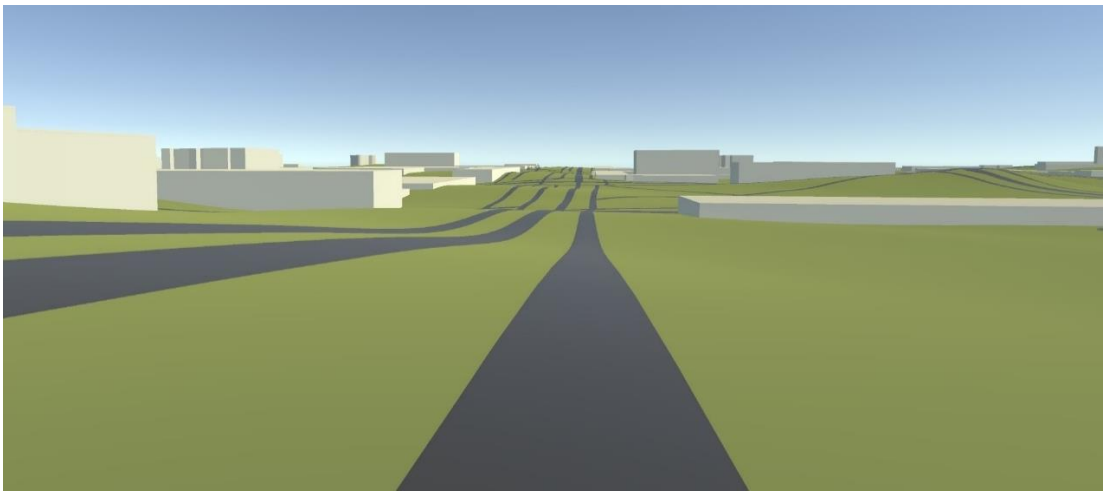
luotujen objektien perusteella eikä pelkän datan pohjalta, mikä nopeuttaa toimenpidettä huomattavasti. Puun luomisen yhteydessä tarkistetaan, onko se päällekkäin rakennusten tai teiden kanssa. Mikäli havaitaan päällekkäisyyttä, puu jätetään luomatta.

6 POHDINTA

Opinnäytetyössä onnistuttiin tavoitteessa luoda 3D-visualisaatio oikean maailman sijainnin pohjalta Unity-pelimoottorissa. 3D-visualisaation luominen mahdollistettiin sekä pelin suoritusaikana että Unity-editorissa. Karttadatan laajuuden vuoksi luotavat elementit täytyi rajoittaa maastoon, rakennuksiin, teihin ja kasvustoon. Näin luodut elementit saatiin luotua hyväksyttävällä tarkkuudella huomioiden pelinkehityksen tarpeet. Ohjelmisto toimii vahvana pohjana mahdolliselle jatkokehitykselle.

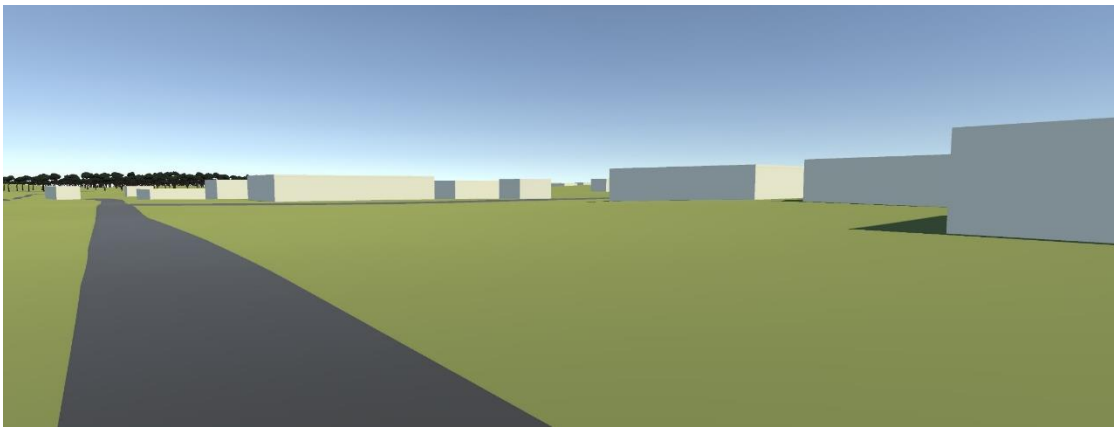
6.1 Käytettävyyden arviointi

Opinnäytetyössä saatiin aikaan hyvä tekninen pohja pelimaailman generointiin. Pelimaailman käytettävyyttä heikentää kuitenkin sen yksinkertainen visuaalinen ilme. Pelimaailman yksinkertaiseen visuaaliseen ilmeeseen vaikuttaa eniten sen yksivärinen teksturointi, mikä saa eri alueet näyttämään melko samanlaisilta. Visuaalista toistoa lisää se, että puissa käytetään vain yhtä prefab-objektia. Pelimaailmaan jää myös paljon tyhjiä alueita, sillä luotuja karttaelementtejä on vielä suhteellisen rajallinen määrä. Yksinkertaisesta visuaalisesta ilmeestä huolimatta generoitu alue toimii hyvänä pohjana pelimaailman kehittämiselle.



Kuva 12. Aaltoilevan maaston vaikutus ensimmäisestä persoonasta

Vähäresoluutioisen korkeuskartan interpolointi saa aikaan kuvassa 12 näkyvää aaltoilevaa maastoa, joka ei näytä pelimaailmassa erityisen luontevalta. Se korostuu etenkin, kun luotuja karttaelementtejä on vielä suhteellisen vähän. Valittu interpolointimetodi kuitenkin minimoi kyseisen efektin. Maaston aaltoilevuudesta huolimatta se toimii pelimaailman laatua nostattavana osana. Maaston korkeuden vaihtelusta eniten hyötyvät alueet, joista ei ole rakennuksien korkeusdataa, kuten kuvan 13 alue. Se, mikä olisi muuten täysin tasainen kattosiluetti, saa maastosta vaihtelua korkeuteen. Tämä tekee rakennuksien toistuvasta korkeudesta vähemmän silmiinpistävän peliympäristössä.



Kuva 13. Ensimmäisen persoonan näkymä alueesta, josta ei ole olemassa rakennuksien korkeusdataa

Tiet ovat pelimaailman ainoa osa, jonka luominen ei saavuttanut tavoiteltua tasoa prosessin nopeuden suhteen. Siihen vaikuttaa merkittävästi se, että tien geometria on tarpeettoman monimutkainen siihen nähden, miten yksinkertaisia teiden muodoista lopulta tuli. Sen lisäksi tiet eivät aina yhdisty toisiinsa tarkoituksenmukaisesti risteyksissä ja muissa teiden liittymäkohdissa.

Maaston tarkkuus ei täyttänyt hankkeen tarpeita, sillä hankkeessa käytetty maasto oli luotu huomattavasti tarkemmasta paikallisesta korkeusdatasta. Muita pelimaailman elementtejä, kuten teitä, pystyttiin kuitenkin hyödyntämään erillään maastosta pohjana alueen käsin luomiseen.

6.2 Jatkokehitys

Karttadata sisältää paljon elementtejä, joiden luomista opinnäytetyö ei käsitellyt. Niistä suurimpina puutteina korostuivat sillat, vesistöt ja pellot. Siltojen puute aiheuttaa paljon epätoivottuja risteymiä pelimaailmassa. Koska tiet luotiin polygonimallina, sillat voidaan luoda tien variaationa säätämällä tien korkeutta ja muotoa. Vesistöjen ja peltojen puute puolestaan jätti paljon tyhjiä alueita luotuun pelimaailmaan. Niiden implementointi nostattaisi tuotoksen näyttävyyttä huomattavasti. Vesistöjen sovittaminen maastoon olisi mahdollista tehdä syventämällä maastoa vesistöjen alueilta ja luomalla vedenpinnalle polygonimalli. Pellot puolestaan pystyisi luomaan samankaltaisesti kuin metsät täydentämällä alue halutulla kasvilla.

Pelimaailman yksinkertaisen visuaalisen ilmeen parantamiseksi pitäisi ottaa paljon vapauksia ympäristön esittämisen suhteen. Karttaelementtien visuaalisista piirteistä, kuten väristä ja materiaalista, on niin vähän dataa, että työssä käytettyjä täydennysmetodeja ei voida hyödyntää tehokkaasti. Visuaaliset piirteet täytyisi määritellä osittain satunnaisesti. Maasto voitaisiin puolestaan teksturoida sen piirteiden, kuten korkeuden ja jyrkkyyden, mukaan. Sen tukena hyödynnettäisiin kartassa ilmeneviä elementtejä, kuten vesistöjä, metsiä ja peltoja.

Osaan luoduista elementeistä jäi selkeitä puutteita ja optimoinnin aiheita. Esimerkiksi rakennuksille, joilla on sisäpiha, jäi toteuttamatta reiällinen katto. Yksi tapa toteuttaa se ilman muutoksia kolmiointialgoritmiin on yhdistää sisäseinät ulkoseiniin niin, että ne muodostavat yhdessä yksinkertaisen monikulmion. Polygonin kolmiointiin olisi myös olemassa nopeampia algoritmeja kuin käytetty ear clipping -algoritmi. Siitä huolimatta suurin optimoinnin tarve jäi tien polygonimallin luomiseen. Tien polygonimallin geometriaa ja sitä käsittelevää algoritmia olisi syytä yksinkertaistaa huomattavasti.

LÄHTEET

Bennett, J. 2010. OpenStreetMap: Be Your Own Cartographer. Birmingham: Packt Publishing. E-kirja. Saatavissa: https://www.packtpub.com/product/openstreetmap/9781847197504 [viitattu 15.10.2023].

Bicubic interpolation s.a. Wikipedia. WWW-dokumentti. Päivitetty 3.12.2023. Saatavissa: https://en.wikipedia.org/wiki/Bicubic_interpolation [viitattu 22.1.2024].

Building s.a. OpenStreetMap Taginfo. WWW-dokumentti. Päivitetty 13.9.2023. Saatavissa: https://taginfo.openstreetmap.org/keys/building [viitattu 14.9.2023].

Chądzyńska, D. & Gotlib, D. 2016. Spatial data processing for the purpose of video games*. *Polish Cartographical Review* 1, 41–50. Verkkolehti. Saatavissa: https://doi.org/10.1515/pcr-2016-0001 [viitattu 26.2.2024].

CityGen Technologies. 2023. CITYGEN3D V1.11 Procedural Scene Generation for Unity. PDF-dokumentti. Saatavissa: https://www.citygen3d.com/_files/ugd/c070e1_04530ef7d1ad4734a73f65cdd5aa54d7.pdf [viitattu 1.2.2024].

Commons s.a. Overpass API User's Manual. WWW-dokumentti. Saatavissa: https://dev.overpass-api.de/overpass-doc/en/preface/commons.html [viitattu 20.12.2023].

Eberly, D. 2002. Triangulation by Ear Clipping. PDF-dokumentti. Päivitetty 5.7.2022. Saatavissa: https://www.geometrictools.com/Documentation/TriangulationByEarClipping.pdf [viitattu 10.10.2023].

Esri ASCII raster format. 2021. ArcMap. WWW-dokumentti. Saatavissa: https://desktop.arcgis.com/en/arcmap/latest/manage-data/raster-and-images/esri-ascii-raster-format.htm [viitattu 17.10.2023].

Hannula, T. 2022. Virtuaalinen sairaalaympäristö 1.0. Kaakkois-Suomen ammattikorkeakoulu. WWW-dokumentti. Saatavissa: https://www.xamk.fi/tutkimus-ja-kehitys/virtuaalinen-sairaalaymparisto/ [viitattu 31.7.2023].

Heidelberg, S. 2021. GIS Visualization and Storytelling In 3D. Blogi. Saatavissa: https://www.esri.com/arcgis-blog/products/arcgis/3d-gis/gis-visualization-and-storytelling-in-3d/ [viitattu 17.3.2024]

HeiGIT. 2023. Exploring OSM editor statistics by combining data from OSHDB and changeset DB. Blogi. Saatavissa: https://heigit.org/exploring-osm-editor-statistics-by-combining-data-from-oshdb-and-changeset-db/ [viitattu 15.10.2023].

Hemalatha, M. 2019. Poisson Disc Sampling. Blogi. Päivitetty 9.4.2019. Saatavissa: https://medium.com/@hemalatha.psna/implementation-of-poisson-disc-sampling-in-javascript-17665e406ce1 [viitattu 27.9.2023].

Highway s.a. OpenStreetMap Taginfo. WWW-dokumentti. Päivitetty 13.2.2024. Saatavissa: https://taginfo.openstreetmap.org/keys/highway [viitattu 13.2.2024].

Lardinois, F. 2020. Meet the startup that helped Microsoft build the world of Flight Simulator. *TechCrunch* 17.8.2020. Verkkolehti. Saatavissa: https://techcrunch.com/2020/08/17/meet-the-startup-that-helped-microsoft-build-the-world-of-flight-simulator/ [viitattu: 9.1.2024].

Mapbox Streets v7 s.a. Mapbox Docs. WWW-dokumentti. Saatavissa: https://docs.mapbox.com/data/tilesets/reference/legacy/mapbox-streets-v7/ [viitattu 29.2.2024].

Mapbox Terrain v2 s.a. Mapbox Docs. WWW-dokumentti. Saatavissa: https://docs.mapbox.com/data/tilesets/reference/mapbox-terrain-v2/ [viitattu 29.2.2024].

Maps SDK for Unity s.a. Mapbox Docs. WWW-dokumentti. Saatavissa: https://docs.mapbox.com/unity/maps/guides/ [viitattu 29.2.2024].

Microsoft. 2021. Microsoft Flight Simulator: The Future of Game Development. Blogi. Saatavissa: https://developer.microsoft.com/en-us/games/articles/2021/07/microsoft-flight-simulator-the-future-of-game-development/ [viitattu 11.1.2024].

Neumüller, T. 2021. Generating terraced terrain from perlin noise: Part 2 – Triangulation. Blogi. Saatavissa: https://neumueller.dev/2021/11/18/terraced-terrain-triangulation/ [viitattu 10.10.2023].

O'Donohue, D. 2021. Open Topography – Infrastructure for Topographic Data. Mapscaping. Podcast. Saatavissa: https://mapscaping.com/podcast/open-topography-infrastructure-for-topographic-data/ [viitattu 27.7.2023].

OpenTopography for Developers s.a. OpenTopography. WWW-dokumentti. Saatavissa: https://opentopography.org/developers [viitattu 16.6.2023].

OpenStreetMap Foundation. 2023. Mission Statement. WWW-dokumentti. Saatavissa: https://wiki.osmfoundation.org/wiki/Mission_Statement [viitattu 27.7.2023].

OpenTopography Open Source Software s.a. OpenTopography. WWW-dokumentti. Saatavissa: https://opentopography.org/otsoftware [viitattu 9.1.2024].

Pokémon Go. 2023. OpenStreetMap Wiki. WWW-dokumentti. Päivitetty 5.1.2023. Saatavissa: https://wiki.openstreetmap.org/wiki/Pokémon_Go [viitattu: 21.11.2023].

Rivera, S. 2023. Conceptual vs logical vs physical data models. WWW-dokumentti. Saatavissa: https://www.thoughtspot.com/data-trends/data-modeling/conceptual-vs-logical-vs-physical-data-models [viitattu: 20.10.2023].

Terms of Use. 2022. OpenTopography. WWW-dokumentti. Saatavissa: https://opentopography.org/usageterms [viitattu 2.1.2024].

USC. 2021. Geospatial Data Is Super Effective! The Use of GIS and Cartography in Pokémon Go. Blogi. Saatavissa: https://gis.usc.edu/blog/use-of-gis-and-cartography-in-pokemon-go/ [viitattu: 20.11.2023].

Yuksel, C., Schaefer, S. & Keyser, J. 2009. On the Parameterization of Catmull-Rom Curves. *SPM '09: 2009 SIAM/ACM Joint Conference on Geometric and Physical Modeling* 5.10.2009, 47–53. Verkkolehti. Saatavissa: https://doi.org/10.1145/1629255.1629262 [viitattu 22.1.2024].