Viikkoraportti 3

Marc Alingué

September 2024

Viikolla 3 oppimaani

Viikon kolme ydin oli lähteä viemään ohjelmaani valmiimpaan suuntaan ja hyödyntää bowyer-watson -algoritmini tuottamaa verkostoa. Looginen etenemissuunta itselleni oli vain jatkaa suoraan eteenpäin ja päädyin hyödyntämään projektissani kruskal algoritmia ja A* -algoritmia.
Kruskalin algoritmia käytin kehittämään bowyer-watson -algoritmin luomasta verkosta minimum spanning treen, joka toimii kartan lopullisena pohjana. Kruskalin algoritmin sijaan olisin voinut käyttää muita minimum spanning tree -algoritmeja, mutta päätin luottaa tuttuun tira kurssilla opittuun algoritmiin.

Kruskalin algoritmin jälkeen meillä on siis tiedossamme haluamamme huoneet, jotka eivät ole päällekkäisiä, meillä on myös polkuina toimivat reunat, eli kaksi pistettä joiden välille haluamme luoda käytävän pelissä. Tämä ongelma on ratkaistavissa mielestäni helpoiten reitinhakualgoritmilla, ja tahdoin toteuttaa sen nimenomaan A* -algoritmilla. A* siis ajetaan jokaiseen huoneiden keskipisteiden völiseen reunaan jotka mst -rakenteeseen on valittu, ja etsii lyhimmän reitin pisteiden välille.

A* on suoraan yhteydessä tapaan jolla on mahdollista piirtää grafiikkaa näytölle. Itselleni selkeä valinta on muuttaa kartta ja siinä esiintyvät pisteet, huoneet ja käytävät matriisiksi. A* ottaakin siksi parametrikseen matriisin, joka on muodostettu kartasta, jossa on huoneiden, ja niiden keskipisteiden sijainnit. Käytän A* -algoritmia sitten ottamaan ylös jokaisen reunan matriisin "ruutujen" välillä, ja sijoitan matriisiin kyseisten ruutujen kohdalle tunnisteen, jolla tiedän mihin kohtaan piirtää käytävä.

Nyt minulla on siis rakennettuna matriisi, jota voin iteroida ja syöttää jokaiseen eri merkin omaavaan kohtaan haluamanlaistani grafiikkaa. Tällä tavalla luon haluamani kokoisista ruuduista koostuvan matriisin ja lopulta joka kerta erilaisen luolaston.

Projekti vaatii kuitenkin vielä hiomista ja sulavoittamista, ja testauksen kehittäminen on vielä hiukan uupuva. Aikavaatimukset pitäisi vielä tarkistaa, mutta tähän mennessä minulla on useita $O(n^2)$ aikaan suoriutuvia algoritmeja.