Toteutusdokumentti

Jäljittäjä - A* - algoritmiin perustuva polunetsintä liikkuvaan maaliin

Harjoitustyö: Tietorakenteet ja Algoritmit

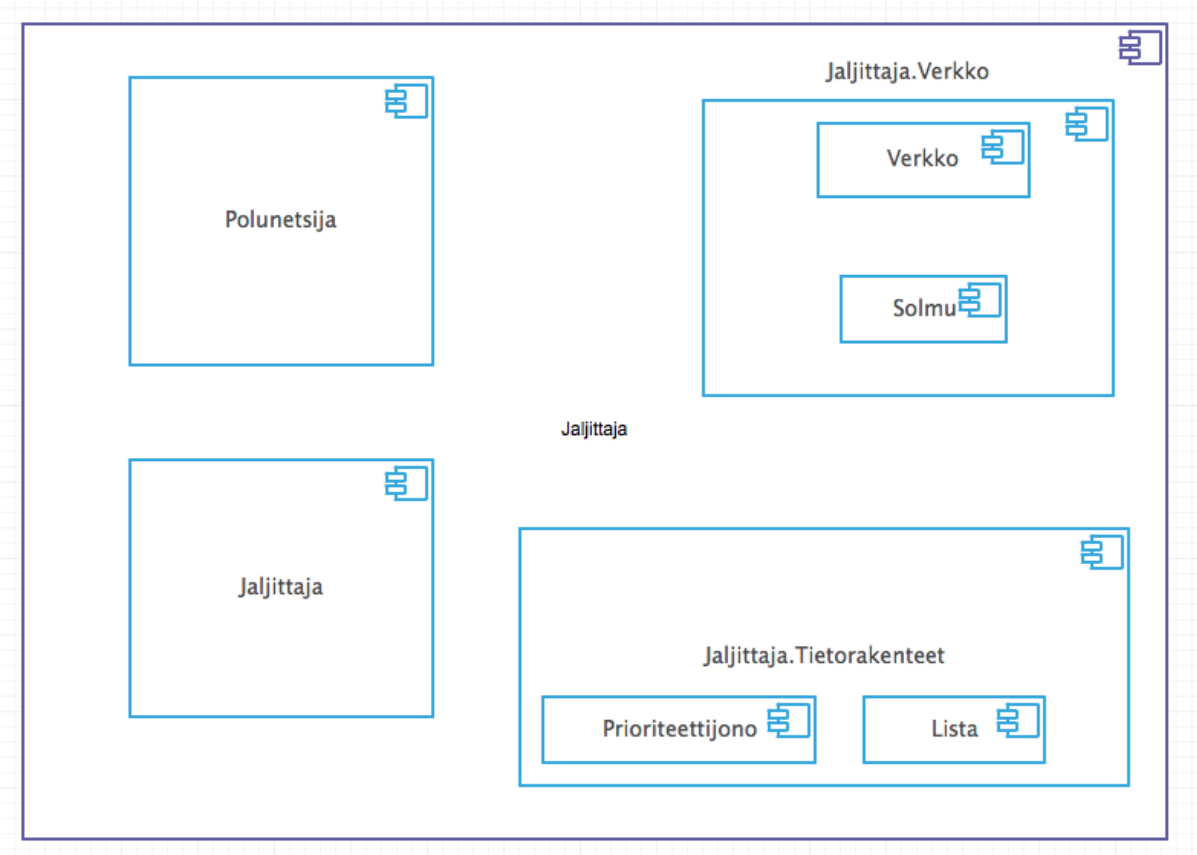
Jouni Männistö
Heinolankatu 6 A 17
00520 Helsinki
jouni.mannisto@cs.helsinki.fi

Ohjelman yleisrakenne

Jäljittäjä-ohjelman ytimen muodostavat samannimisessä paketissa olevat Jaljittaja- ja Polunetsija-luokat.
Jaljittaja-luokassa on ohjelman käynnistävä päämetodi. Polunetsija-luokkaan on toteutettu nimensä mukaisesti varsinainen polunetsintäalgoritmi.
Jaljittaja-paketin sisällä ovat myös ohjelmassa käytetyt Lista- ja Prioriteettijono -tietorakenteet (Jaljittaja.Tietorakenteet) sekä polunetsintäalgoritmin syötteenä saama Verkko (Jaljittaja.Verkko).

Lisäksi ohjelmassa on käyttöliittymä, jonka avulla polunetsintäalgoritmin käyttäytymistä voi havainnoida empiirisesti mm. eri kokoisissa verkoissa ja eri nopeuksilla. Käyttöliittymältä on myös mahdollista ajaa algoritmia ilman liikkuvaa maalia.

Ohjelmassa on myös ns. Massasuorittaja-luokka, jonka avulla polunetsintä voidaan suorittaa useita kertoja (koodiin kovakoodatuilla parametreilla) ja kirjoittaa suoritusten tulos csv-tiedostoon myöhempää vertailua varten.
Edellä mainittu toiminnallisuus käyttää CSVKirjoittaja-luokkaa tiedostoon kirjoittamiseen sekä SuorituksenInfo-luokkaa tiedonkeruuseen.



Kuva 1. Komponettikaavio Jäljittäjän ytimestä

Tulokset

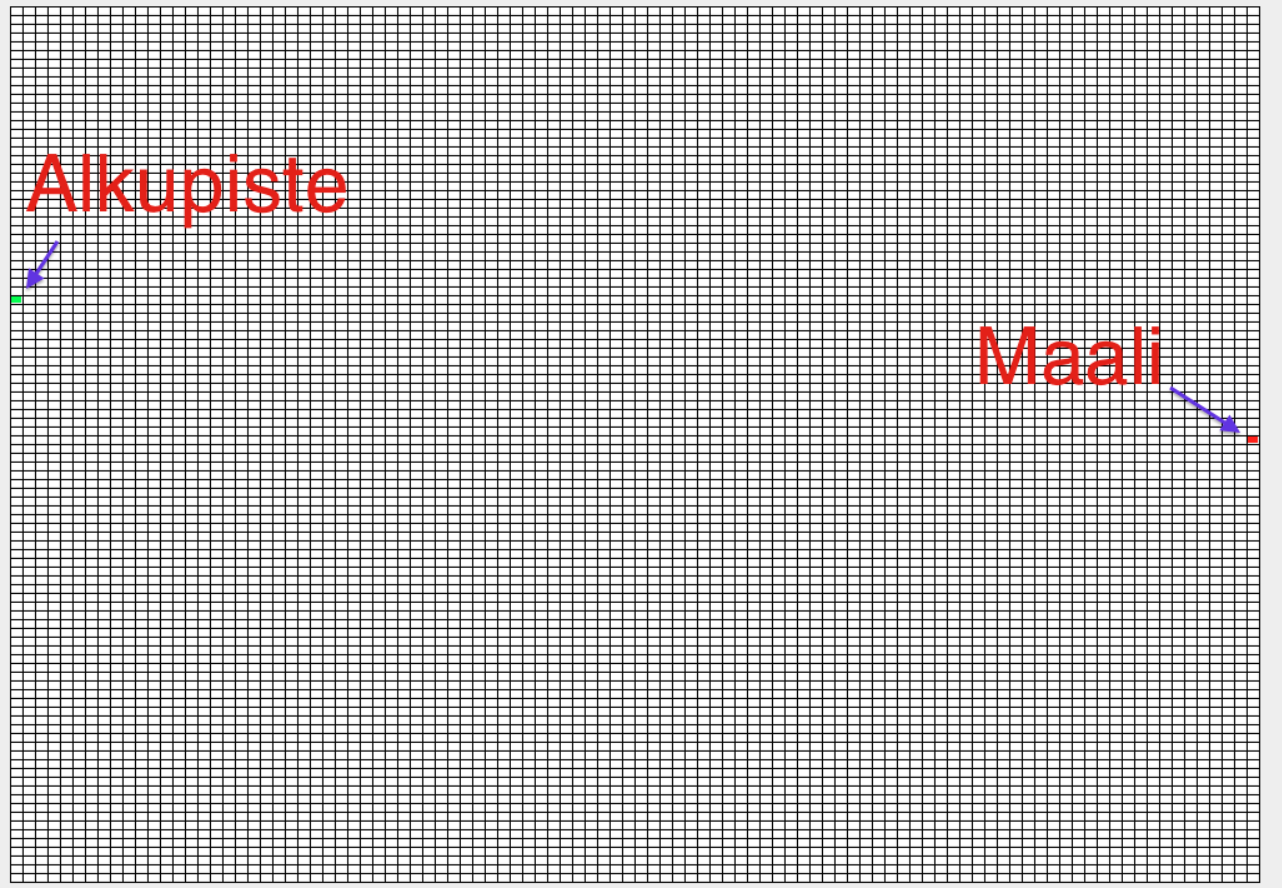
Taustaa

Tavalliseen polunetsintään verrattuna Jäljittäjä-ohjelmalla on suuri haaste: etsiä liikkuva maali. Algoritmi arvioi etäisyyden ja kustannuksen (g(x) ja h(x)) vain käsittelemättömien solmujen kohdalla.
Tällä voi olla merkittävä hyöty nopeudessa, koska maalin liikkuminen karsii prioriteettijonoon päätyneiden “naapurisolmujen” määrää (osan prioriteetti putoaa usien “naapurien” tullessa jonoon) ja alkuvaiheessa algoritmi pääsee etenemään kohti maalia nopeasti.
Vastaavasti riski epäonnistumisesta (= maalia ei saada kiinni) kasvaa, koska maalin liikkuessa jo käsiteltyjen solmujen “alueelle”, se saattaa päästä Jäljittäjän ulottumattomiin.

Tässä luvussa testiolosuhteet ovat seuraavat:

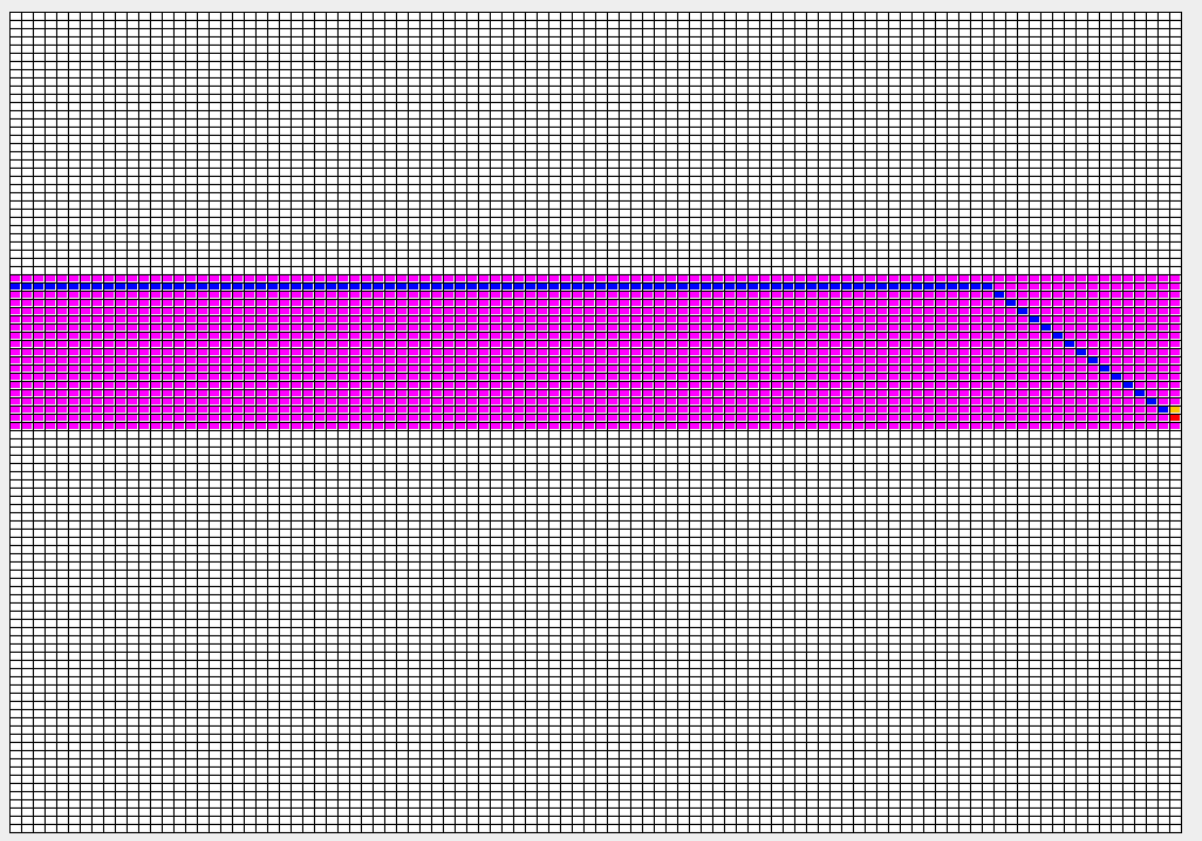
- verkon koko 10000 solmua (100 x 100 matriisi)
- alkupisteen koordinaatit [0, 33], maalin [99,49]
- molemmat pätevät sekä käyttöliittymältä että massa-ajona suoritettuihin ajoihin
- massa-ajo tekee 100 suoritusta
- olosuhteet ovat toistettavissa (muuttamalla käyttöliittymällä verkon sivuiksi 100)

Lähtötilanne:



Kuva 1.

Suoritus staattiseen maaliin:



Kuva 2. Läpikäydyt solmut purppuralla, löytynyt polku sinisellä.

Suoritus massa-ajona

1. Ajo.Liikkuva maali: **false**, alkupiste: [0, 33], maali [99,49]
Solmuja: 10000, kierroksia 100, polku löytyi 100% suorituksista, aikaa/suoritus n 41.72 ms

2. Ajo. Liikkuva maali: **true**, alkupiste: [0, 33], maali [99,49]
Solmuja: 10000, kierroksia 100, polku löytyi 96% suorituksista, aikaa/suoritus n 127.45 ms

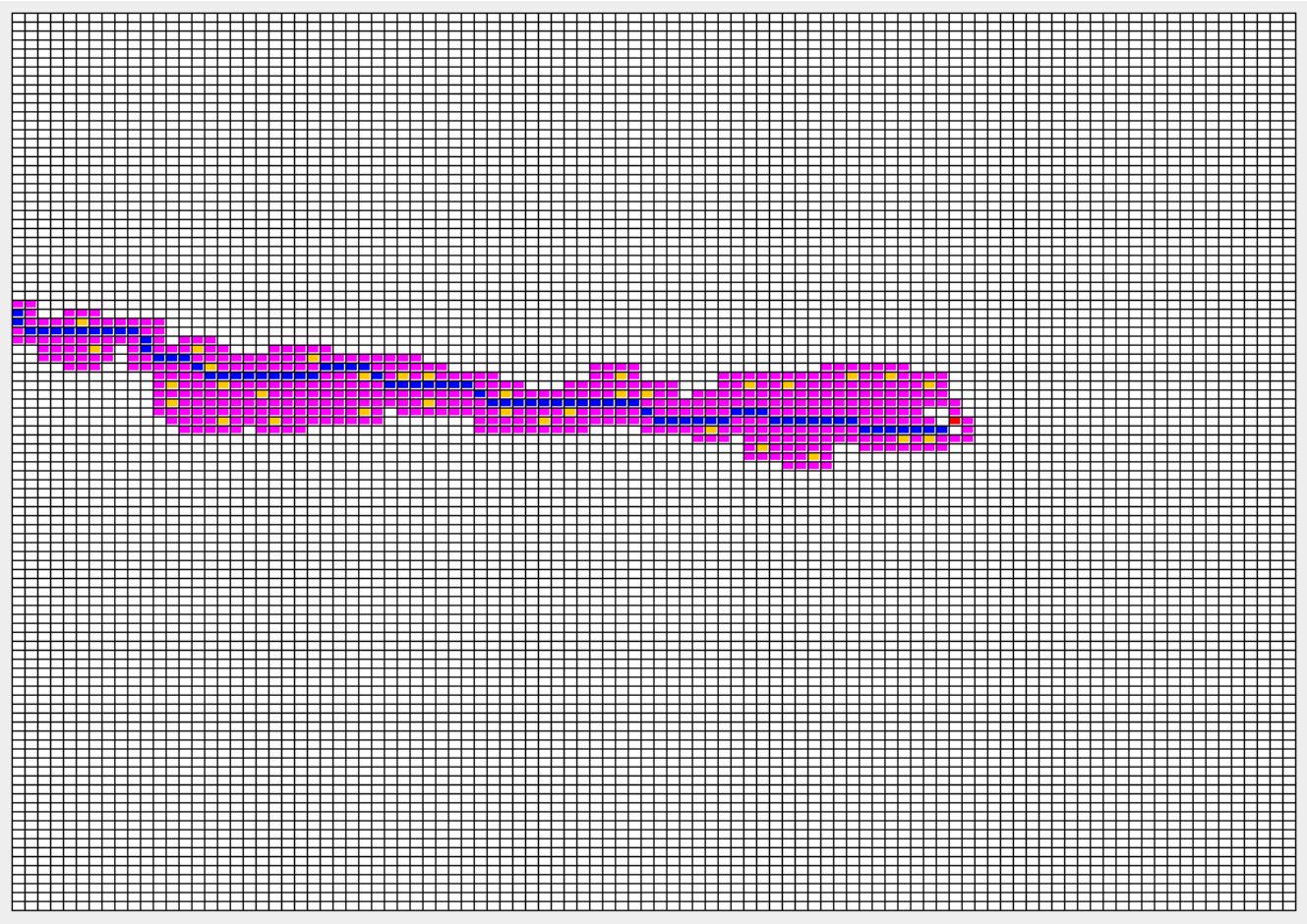
3. Ajo. Liikkuva maali: **true**, alkupiste: [0, 33], maali [99,49]
Solmuja: 10000, kierroksia 100, polku löytyi 95% suorituksista, aikaa/suoritus n 129.17 ms

Ensimmäinen ajo paikallaan pysyvään maaliin antaa vertailuajan 41.72ms. Keskimääräistä suoritusaikaa liikkuvan maalin tapauksessa heikentää juuri se, että jos maali pääsee pakoon, käsittelee algoritmi lähes kaikki solmut.

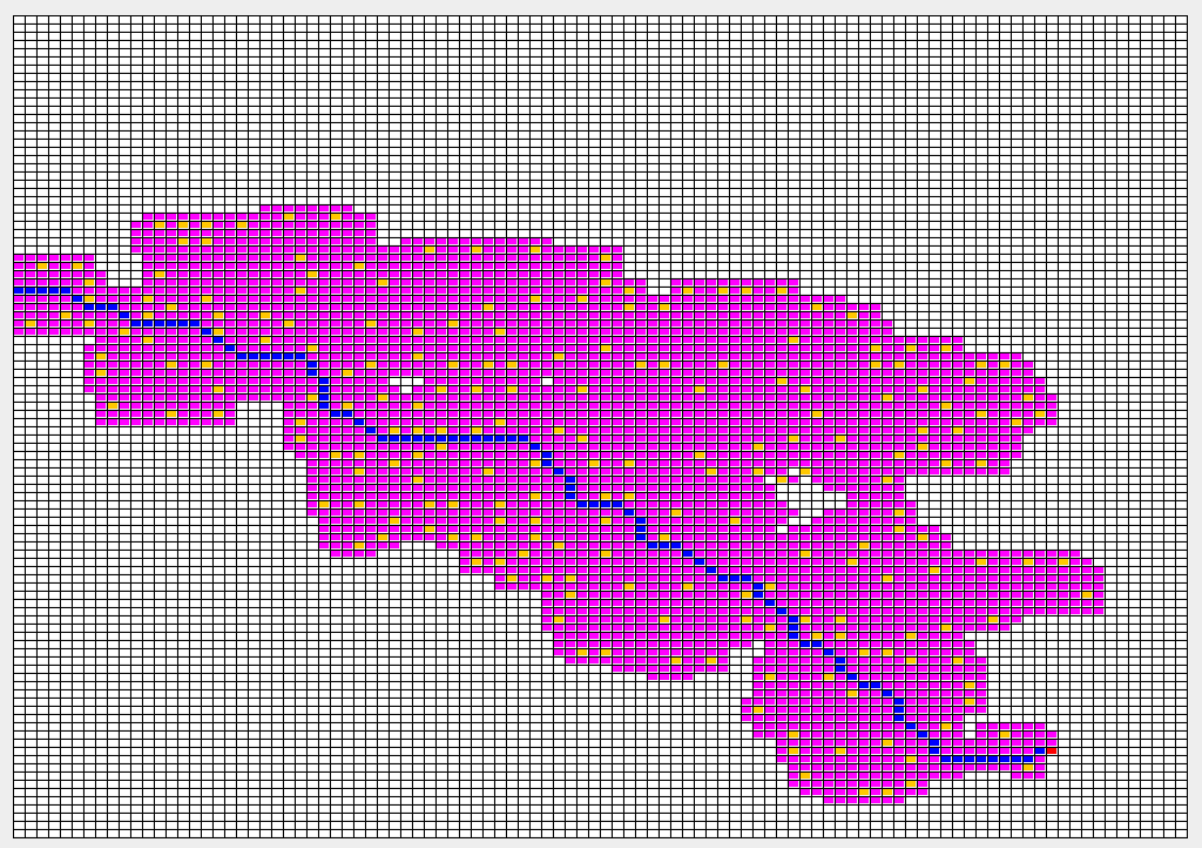
Muutama esimerkki kolmannen ajon suorituksista:

Kesto (ms)	Maalin askeleet
3	341*
4	500*
7	557*
271	4349
295	4647
1373	10000 (ei löytynyt)**

*) Mikäli maali liikkuu suotuisasti algoritmin kannalta, voi algoritmi suoriutua nopeammin kuin jos maali olisi paikoillaan. Kts. kuva 3.

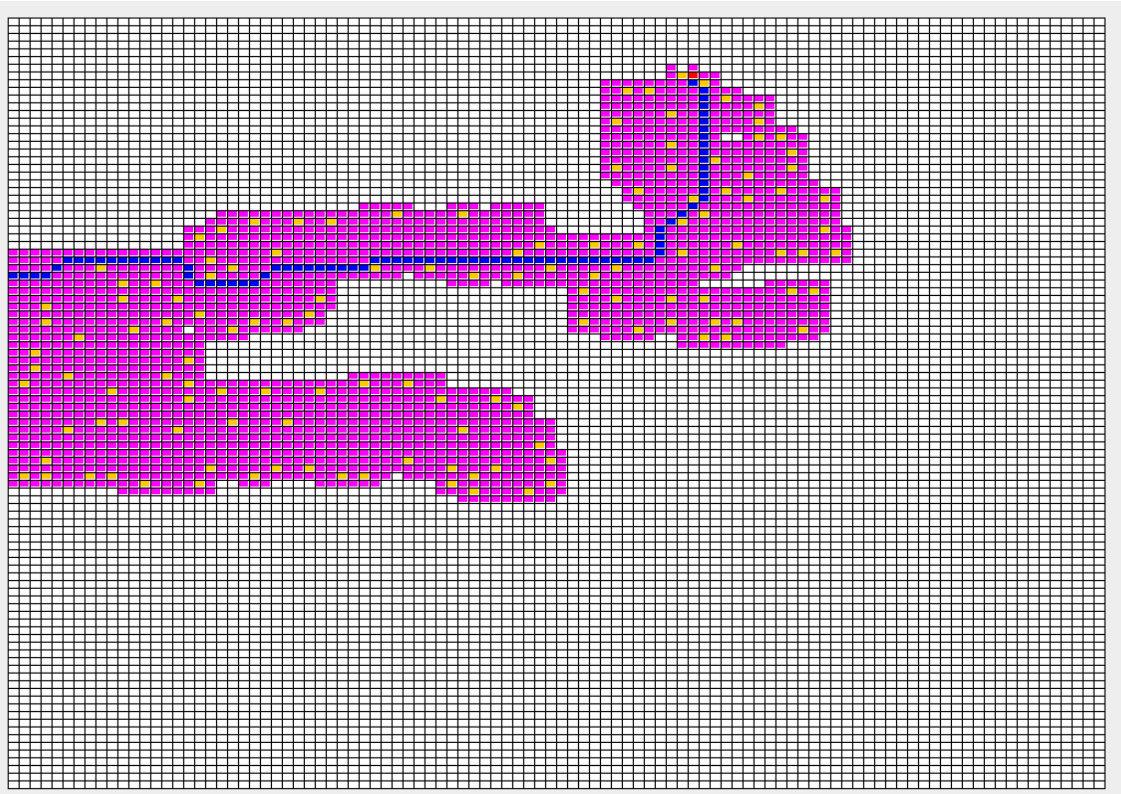


Kuva 3. Jos maali liikkuu pienellä alueella ja/tai kohti Jäljittäjää, niin suoritus voi olla nopea. Ns. paras tapaus.

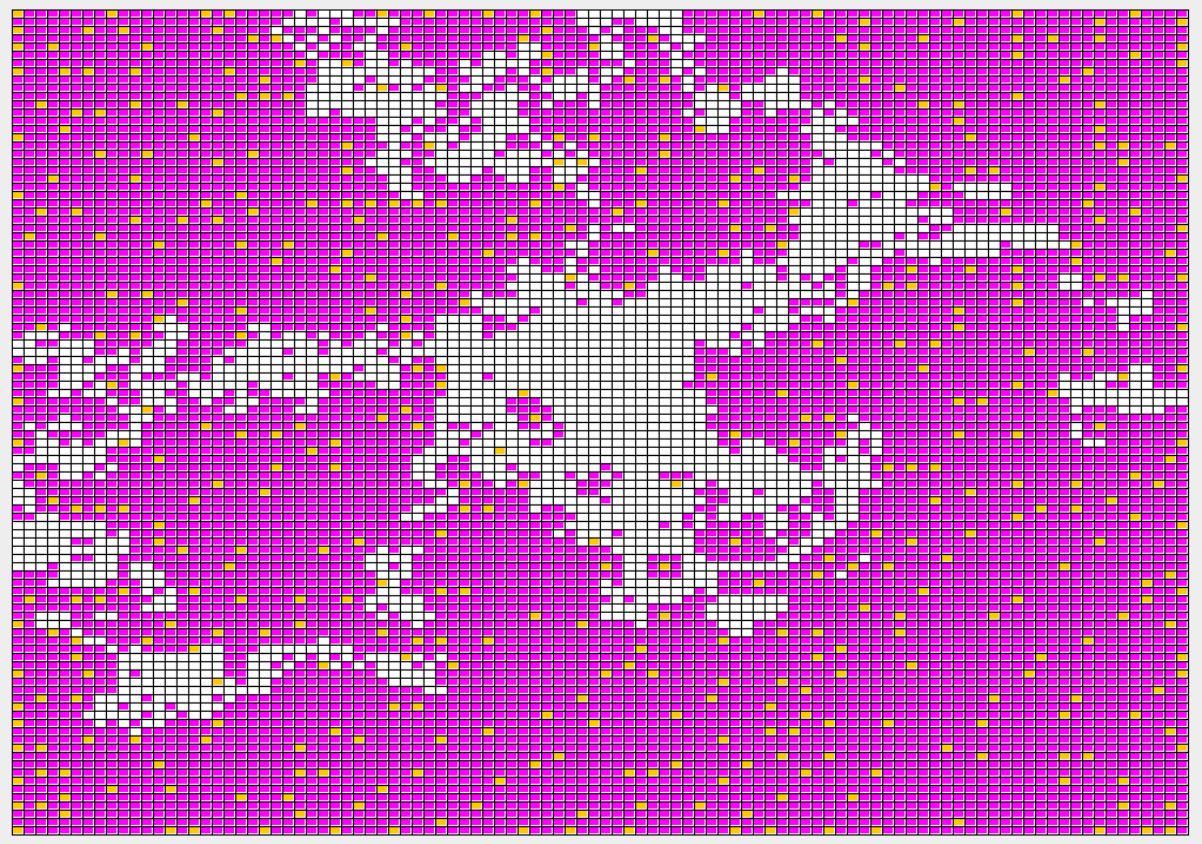


Kuva 4. Kuvasta huomaa, että polunetsijä on joutunut laajentamaan aluetta suorituksen loppuvaiheessa.

Kuva 5. Maali on suunnannut aikaisessa vaiheessa “pohjoiseen” jolloin prioriteettijonosta on löytynyt alkupään solmuja käsiteltäväksi. Polku suuntaakin suoraan ylöspäin alkupisteestä.



Kuva 6. Kuvasta voi nähdä, kuinka maalin liikkuessa ylöspäin polku kääntyy 90 astetta pohjoiseen.



Kuva 7. Huonoin skenaario: maali pääsee livahtamaan jo käsiteltyjen solmujen alueelle, jolloin Jäljittäjän mahdollisuudet “korjata” reittiä pienenevät. Valkoiset solmut ovat maalin “jälkiä”.

Parannuksia, ideoita

- Polunetsintää voisi parantaa esim. siten, että mikäli havaitaan maalin liikkuvan tietyt raja-arvot ylittäen, voisi jo käsiteltyjä solmuja ottaa uudelleen käsiteltäväksi. Tällä olisi tietysti vaikutusta suorituksen kestoon, mutta sillä voisi päästä lyhyempään polkuun. Nyt reitti päivittyy vain käsittelemättömien solmujen kohdalla, joten polku voi tehdä suuria mutkia.
- Liikkuvaan maaliin suuntaava polunetsintä ruokkii mielikuvitusta erilaisten pelien alustana; “saalis vs. saalistaja”, “projektipäällikkö vs. asiantuntija avokonttorissa”, ohjuksen väistö...

Lähteet

http://theory.stanford.edu/~amitp/GameProgramming/AStarComparison.html
http://en.wikipedia.org/wiki/A*_search_algorithm

Versiohistoria

12.5.2015	Jouni	Ensimmäinen versio
13.5.2015	Jouni	Ideat, parannukset, kielioppivirheitä...