

Mars-luotainten reitinhakualgoritmit

Jerry Mesimäki

Kandidaatintutkielma
HELSINGIN YLIOPISTO
Tietojenkäsittelytieteen laitos

Helsinki, 18. lokakuuta 2014

Tiedekunta — Fakultet — Faculty		Laitos — Institution — Department	
Matemaattis-luonnontieteellinen		Tietojenkäsittelytieteen laitos	
Tekijä — Författare — Author			
Jerry Mesimäki			
Työn nimi — Arbetets titel — Title			
Mars-luotainten reitinhakualgoritmit			
Oppiaine — Läroämne — Subject			
Tietojenkäsittelytiede			
Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages	
Kandidaatintutkielma	18. lokakuuta 2014	2	
Tiivistelmä — Referat — Abstract			
Tiivistelmä.			
Avainsanat — Nyckelord — Keywords			
avainsana 1, avainsana 2, avainsana 3			
Säilytyspaikka — Förvaringsställe — Where deposited			
Muita tietoja — Övriga uppgifter — Additional information			

Sisältö

1	Ensimmäinen luku	1
2	Globaalin reitinsuunnittelun toteutus Field D*-algoritmillä	1
2.1	Kustannusarvion parantaminen interpoloinnin avulla	1
	Lähteet	2

1 Ensimmäinen luku

Tiedettä tekstimuodossa ja muutama kuva.

Esimerkkilause ja lähdeviite [1].

2 Globaalin reitinsuunnittelun toteutus Field D*-algoritmillä

2.1 Kustannusarvion parantaminen interpoloinnin avulla

Algoritmin perustana toimii metodi, jossa jokaisesta ruudukossa sijaitsevasta solmusta lasketaan halvin mahdollinen kustannusarvio haluttuun kohdepisteeseen. Perinteisesti ruudukkopohjaisessa reitinsuunnittelussa on käytetty seuraavaa kaavaa:

$$g(s) = \min_{s' \in nbrs(s)} (c(s, s') + g(s')),$$

missä $nbrs(s)$ on joukko kaikista solmun s naapureista, $c(s, s')$ on kustannus kulkemiseen kaaren s ja s' välillä, sekä $g(s')$ on kustannusarvio solmulle s' .

Kaavassa oletetaan, että solmusta s voidaan siirtyä tämän naapureihin ainoastaan suoraa linjaa pitkin, joka taas johtaa aikaisemmin mainittuun ongelmaan muodostaa parhaita mahdollisia reittejä robotin rajoittuneen suuntaamisen takia. Tämä voitaisiin korjata sijoittamalla s' :n tilalle s_b , jossa s_b on mikä tahansa piste solmuun liittyvän ruudun reunalla. Näitä pisteitä on kuitenkin ääretön määrä, joka tekee jokaisen pisteen laskennasta mahdotonta.

Muokkaamalla verkkoa voimme silti muodostaa approksimaation jokaiselle pisteelle s_b käyttäen lineaarista interpolointia. Sen sijaan, että solmu sijoitettaisiin ruudun keskelle, asetetaan solmu jokaisen ruudun kulmaan ja täten kaaret kulkevat ruudukon reunoja pitkin. Nyt yhden kaaren kustannus voidaan valita siten, että se on pienempi niistä kahdesta ruudusta joiden välillä se kulkee.

Tämä muokkaus johtaa siihen, että paras mahdollinen reitti kulkee jotkin kaksi naapurisolmua $\overrightarrow{s_1 s_2}$ yhdistävän kaaren läpi. Solmulle s voidaan nyt laskea kustannusarvio, kun reitti kulkee sen ruudun läpi em. kaarelle. Laske- mista varten tarvitaan arviot solmujen s_1 ja s_2 sekä keskimmäisen ruudun c ja alemman ruudun b kustannuksista.

Kustannusarvion tuottamiseen käytetään vielä oletusta, että kustannusarvio mille tahansa pisteelle s_y , joka sijaitsee kaarella $\overrightarrow{s_1 s_2}$, on funktioiden $g(s_1)$ ja $g(s_2)$ lineaarikombinaatio:

TAAS YKS KAAVA JA SIT SELOSTUKSET =))))

Tulee kuitenkin huomata, että s_y *eiväلتtämättä ole em. funktioiden lineaarikombinaatio, muttatäm...*

Approksimaation perusteella solmun s kustannus kun tiedetään s_1 , s_2 , ruutukustannukset c ja b voidaan laskea seuraavasti:

SIT TAAS YKS NIIN JEBIN KAAVA TÄHÄN BUDDY ET EI MIDIST
RAJOI :D:D:D HUOM. MUISTA SELITYKSET JÄBÄ :D:D::D

Tehdään vielä oletus, että (x^*, y^*) ovat $x:n$ ja $y:n$ arvot, joilla ylläoleva kaava saadaan ratkaistua. Lineaarisen interpoloinnin johdosta toinen arvoista on joko yksi tai nolla. Mikäli kustannus liikkua ruudun c yli on pienempi kuin ruudun reunoja pitkin kulkeminen niin halvin reitti halkaisee ruudun c ja täten joko $x^* = 0$ tai $y^* = 1$. Jos taas polku ei halkaise ruutua c niin $y^* = 0$. Täten polku on kulkee solmusta s suoraan alareunaa pitkin kohtia solmua s_1 , siirtyy jonkin matkan x alareunalla ja leikkaa tämän jälkeen ruudun halki suoraan solmuun s_2 , tai halkaisee ruudun c kulkemalla suoraan solmusta s johonkin oikean reunan pisteeseen s_y . *Halvin polku riippuu $c : njab : nkoosta, sekä $s_1 : njas_2 : nkustannuserosta f = g(s_1) - g(s_2)$. Mikäli $f < 0$ niin paras mahdollinen polku on 1. tapaus, josta $asf = b$ niin polun kustannus kulkien jonkin matkan alareunaa pitkin on f .$*

KAAVAA TAAS TÄNNE NÄ-IN :)))

Lopputulos on sama huolimatta siitä kuljetaanko alareunaa pitkin, joten merkitseväksi tekijäksi jää se kumpaa reunaa kulkeminen tulee halvemmaksi. Mikäli $f < b$ niin käytetään oikeaa reunaa ja lasketaan polun kustannus arvolla $k = f$. Jos taas $b < f$ niin käytetään alareunaa jolloin $k = b$ ja $y^* = 1 - x^*$. Täten algoritmi halvimman polun laskemiseen solmusta s mihin tahansa pisteeseen kaarelle, joka sijaitsee vierekkäisten naapurien s_a ja s_b välissä laskemiseen on seuraava :

ALGO :))

Lähteet

- [1] Rand, Ayn: *The Fountainhead*.