

SMC-menetelmät

Laskennallinen tilastotiede - Harjoitustyö

Lasse Rintakumpu

04 May, 2021

Sisällys

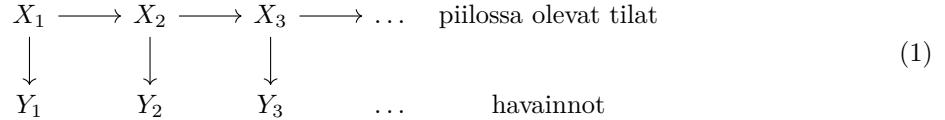
1 Johdanto	1
1.1 Suodinongelma	2
1.2 Historiaa	2
2 Bayesilainen suodin	3
3 Hiukassuodin-algoritmi	4
3.1 Parametrien valinta	5
3.1.1 Otoskoon N valinta	5
3.1.2 Uudelleenottamisen menetelmän valinta	6
3.1.3 Ehdotusjakauman valinta	6
3.2 Konvergenssituloksia	7
3.3 Marginaalijakauma	8
3.4 Aikakompleksisuus	8
4 Paikannusesimerkki	8
4.1 Koeasetelma	8
4.2 Datan kuvaus	11
4.3 Ongelman kuvaus	12
4.4 Algoritmi	12
4.5 Tulokset	12
5 Lopuksi	13
6 Lähteet	13
7 Liite 1: SMC-paikannusalgoritmin R-koodi	13

1 Johdanto

SMC-menetelmät (Sequential Monte Carlo -menetelmät, tunnetaan tilastotieteessä myös nimellä hiukassuotimet) ovat joukko 90-luvulta eteenpäin kehitettyjä Monte Carlo -algoritmeja, joiden avulla voidaan ratkaista ns. suodinongelma, kun ongelma on epälineaarinen ja/tai ongelmaan liittyvä kohina ei noudata normaalijakaumaa. Tämän tutkielman tavoitteena on esittää pääpiirteittäin SMC-menetelmien teoria sekä joitakin menetelmäperheeseen kuuluvia algoritmeja. Tutkielman esitykset seuraavat erityisesti Simo Särkkän kirjaa *Bayesian Filtering and Smoothing* (2013) sekä Fredrik Gustafssonin artikkelia “Particle Filter Theory and Practice with Positioning Applications” (2010). SMC-menetelmille on lukuisia sovellutuksia esimerkiksi Bayesilaisessa tilastotieteessä, fysiikassa ja robotiikassa. Tämän tutkielman lopussa tarkastellaan hiukassuotimen käyttöä paikannusovelluksessa.

1.1 Suodinongelma

Stokastisten prosessien teoriassa suodinongelmaksi kutsutaan tilannetta, jossa halutaan muodostaa paras mahdollinen estimaatti jonkin järjestelmän tilan arvoille, kun ainoastaan osa tiloista voidaan havaita ja/tai havaintoihin liittyy kohinaa. Tavoitteena on siis laskea jonkin prosessin posteriorijakauma kyseisten havaintojen perusteella. Tässä tutkielmanmassa keskitytään erityisesti epälineaarisen ns. Markovin piilomallin posteriorijakauman Bayesilaiseen ratkaisuun. Ongelmaa havainnollistaa kaavio (1).



Ongelmassa tiedämme, miten havaitut muuttujat kytkeytyvät "piilossa oleviin" muuttuijiin sekä osaamme sanoa jotain tilamuuttujien todennäköisyysistä. Oletamme lisäksi, että piilossa olevat tilat muodostavat Markovin ketjun. Kun aika-avaruus on diskreetti ja merkitsemme piilossa olevan prosessin tilaa ajanhetkellä k x_k ja havaittua prosessia y_k , meillä on siis olemassa mallit

$$x_{k+1} = f(x_k, \nu_k) \tag{2}$$

$$y_k = h(x_k) + e_k. \tag{3}$$

Lisäksi tiedetään prosessin alkuperäisen jakauman $x_1 \sim p_{x_1}$, tähän liittyvä kohinprosessin jakauma $\nu_k \sim p_{\nu_k}$ sekä malliin y_k liittyvä kohina $e_k \sim p_{e_k}$. Koska SMC-algoritmit pyrkivät ratkaisemaan juurikin epälineaarisen, ei-Gaussisen suodinongelman, voivat funktiot $f(\cdot)$ ja $h(\cdot)$ olla epälineaarisia eikä kohinan tarvitse olla normaalijakautunutta.

Mallit voidaan yleisemmällä tasolla esittää myös jakaumamuodossa

$$x_{k+1} \sim p(x_{k+1}|x_k) \tag{4}$$

$$y_k \sim p(y_k|x_k) \tag{5}$$

MISSÄ Tutkielman teoriaosassa käytetään ensisijaisesti yhtälöiden (4) ja (5) muotoilua. Empiirisessä osassa palataan yhtälöiden (2) ja (3) muotoiluun.

Näin määritetyjen mallien avulla SMC-menetelmät estimoivat sekventiaalisesti tilojen X_k arvot minä hyvänsä ajan hetkellä k , kun ainoastaan prosessi Y_1, \dots, Y_k tunnetaan. Estimaatit saadaan posteriorijakaumasta $p(x_k|y_1, y_2, \dots, y_k)$, jonka approksimaatio muodostetaan Bayesilaisittain havaintojen pohjalta. Kuten mainittua, ei SMC-perheen algoritmeja käytettäessä mallin $y_k = h(x_k) + e_k$ tarvitse olla lineaarinen eikä kohinaprosessien noudataa normaalijakaumaa. SMC-menetelmässä stokastisen prosessin posteriorijakauman esittämiseen käytettyjä otoksia kutsutaan myös partikkeleiksi. Suodinongelmaa lähellä on myös ns. tasoitusongelma (smoothing problem), jossa ollaan kiinnostuneita prosessin x_k posteriorijakaumasta $p(x_k|y_k)$ jokaisena ajanhetkenä $1, \dots, k$ ei ainoastaan haluttuna ajanhetkenä k . Tämä tutkielma keskittyy yksin suodinongelman ratkaisemiseen, mutta SMC-algoritmin marginalisointia koskevassa luvussa viitataan myös tasoitusongelman ratkaisuun.

1.2 Historiaa

Lineaarisen stokastisen suodinongelman ratkaisu on tunnettu ... Alaluku noudattaa suodatinongelman historian kuvaussa Crisanin artikkelia "The stochastic filtering problem: a brief historical account" (2014) ja SMC-menetelmien kuvaussa Cappé & al artikkelia "An overview of existing methods and recent advances in sequential Monte Carlo" (2007). SMC-menetelmien osalta sivutetaan siis viimeisen kymmenen vuoden kehitys. Tämä on linjassa tutkielman varsinaisen teoriaosuuden kanssa.

Epälineaariselle ei-Gaussilaiselle mallille ... Muita vaihtoehtoja ovat EKF, UKF, QKF. It should be stressed that both EKF and UKF approximate the model and propagate Gaussian distributions representitive

of the post erior while the PMF uses the original model and approximates the posterior over a grid. The particle filter (PF) also provides a numerical approximation to the nonlinear filtering problem similar to the PMF but uses an adaptive stochastic grid that automatically selects relevant grid points in the state space, and in contrast to the PMF, the standard PF has linear complexity in the number of grid points. The first traces of the PF date back to the 1950s [11, 12], and the control community made some attempts in the 1970s [13, 14]. However, the PF era started with the seminal paper [15], and the independent developments in [16, 17]. Here, an important simplifying step was introduced. The timing for proposing a general solution to the nonlinear filtering problem was perfect in that the computer development enabled the use of computationally complex algorithms in quite realistic problems. Since the paper [15] the research has steadily intensified; see the article collection [18], the surveys [19-22], and the monograph [23]. Fig. I illustrates how the number of papers increases exponentially each year, and the same appears to be true for applied papers. The PFs may be a serious alternative for real-time applications classically approached by the (E)KF. The more nonlinear model, or the more non-Gaussian noise, the more potential PFs have, especially in applications where computational power is rather cheap, and the sampling rate is moderate.

Monte Carlo -ratkaisuja (esim. Princeton)

Ensimmäisen epälineaarisen suodinongelman Bayesilaisen, Monte Carlo -estimointiin perustuvan ratkaisun esittivät Gordon, Salmond ja Smith artikkelissaan “Novel approach to nonlinear/non-Gaussian Bayesian state estimation” (1993). Gordonin, Salmondin ja Smithin ratkaisu eroaa notaatioltaan hieman tässä tutkielman esitetystä, mutta on ollenkesti sama. Suurin ero tämän tutkielman SMC-algoritmin sekä alkuperäisen SMC-algoritmin välillä on XXX MITEN XXX. Artikkelissa ratkaisu kului nimellä “bootstrap filter”, saapaneen remmisiuoden. MIKSI Termiä hiukassuoden käytti ensimmäisen kerran Del Moral artikkelissa “Nonlinear Filtering: Interacting Particle Resolution” (1996), SMC-menetelmät termiä Liu ja Chen artikkelissa “Sequential Monte Carlo Methods for Dynamic Systems” (1998). Tässä tutkielman pyritään korostamaan suotimien yhteyttä Monte Carlo -algoritmeihin ja käytetään siksi termiä SMC-menetelmät. Poikkeuksen tähän tekee varsinainen esitetty algoritmi, jota kutsutaan tutkielman hiukassuoden-algoritmiksi.

2 Bayesilainen suodin

Suodinongelmassa ollaan kiinnostuttu tilavektorin posteriorijakauman $p(x_k|y_{1:k})$ estimoinnista. Koska epälineaarisessa tapauksessa ei jakaumaa voida laskea analytisesti, pitää estimoinnissa käyttää numeerisia menetelmiä. Tässä luvussa käydään läpi epälineaarisen hiukassuotimen yleinen rekursiivinen, Bayesilainen posteriorijakauman laskenta, jonka numeerinen toteutus esitetään luvun 3 hiukassuodinalgoritmissa. Tällaista suodinongelman ratkaisua kutsutaan myös Bayesilaiseksi suotimeksi. Esitys noudattaa Fredrik Gustafssonin artikkelia “Particle Filter Theory and Practice with Positioning Applications” (2010).

Bayesilainen ratkaisu tilavektorin posteriorijakauman estimaatille $\hat{p}(x_k|y_{1:k})$ saadaan seuraavalla rekursiolla (käydään läpi jokaiselle ajanhetkelle $k = 1, \dots, t$). Lasketaan ensin

$$p(x_k|y_{1:k}) = \frac{p(y_k|x_k)p(x_k|y_{1:k-1})}{p(y_k|y_{1:k-1})}, \quad (6)$$

joka saadaan suoraan Bayesin kaavasta $P(A|B) = P(B|A)P(A)/P(B)$. Normalisointivakio lasketaan integraalina

$$p(y_k|y_{1:k-1}) = \int_{\mathbb{R}^{n_x}} p(y_k|x_k)p(x_k|y_{1:k-1}) dx_k, \quad (7)$$

joka saadaan kokonaistodennäköisyyskaavasta $P(A) = \mathbb{E}[P(A|X)] = \int_{-\infty}^{\infty} P(A|X=x)f_X(x) dx$. Merkintä R^{n_x} vastaa tässä pilossa olevien muuttujien dimensioita n .

Lopuksi lasketaan päivitysaskel ajalle, joka saadaan edelleen kokonaistodennäköisyydellä

$$p(x_{k+1}|y_{1:k}) = \int_{\mathbb{R}^{n_x}} p(x_{k+1}|x_k)p(x_k|y_{1:k}) dx_k. \quad (8)$$

Rekursion avulla voidaan laskea $p(x_k|y_{1:k})$ käymällä rekursio läpi k kertaa.

3 Hiukassuodin-algoritmi

Tässä luvussa esitetään hiukassuodin-SMC-algoritmi Bayesilaisen, epälineaarisen suodinongelman ratkaisemiseksi. Algoritmi on numeerinen toteutus luvussa 2. kuvatusta Bayesilaisesta suotimesta. Esitetty algoritmi perustuu Fredrik Gustafssonin artikkeeliin “Particle Filter Theory and Practice with Positioning Applications” (2010).

Algoritmi alustetaan jakaumasta $x_1^i \sim p_{x_0}$ generoiduilla N -kappaleella partikkeleita. Jokaiselle partikkeille annetaan alustuksessa sama paino $w_{1|0}^i = 1/N$. Algoritmi suoritetaan jokaiselle partikkeliille $i = 1, 2, \dots, N$ jokaisella ajanjäädikellä $k = 1, 2, \dots, t$. Algoritmin kuvauksessa käytetään notaatiota x_k^i , joka tarkoittaa, että tila x_k käy ajanjäädikellä k gridin pisteessä x^i . Notaatiota tarvitaan, koska SMC-algoritmin läpikäymä gridi muuttuu ajan funktiona. SYÖTE

Algoritmin ensimmäisessä vaiheessa päivitetään painot yhtälön 9 mukaan.

$$\hat{p}(x_{1:k}|y_{1:k}) = \sum_{i=1}^N w_{k|k}^i \delta(x_{1:k} - x_{1:k}^i) \quad (9)$$

Normalisointipaino c_k lasketaan puolestaan yhtälöstä (10). VASTAAVUUS BAYES-suotimeen.

$$c_k = \sum_{i=1}^N w_{k|k-1}^i p(y_k|x_k^i). \quad (10)$$

Seuraavassa vaiheessa estimoidaan p laskemalla tiheyden $\hat{p}(x_{1:k}|y_{1:k})$ MC-estimaatti yhtälön (11) perusteella

$$\hat{p}(x_{1:k}|y_{1:k}) = \sum_{i=1}^N w_{k|k}^i \delta(x_{1:k} - x_{1:k}^i) \quad (11)$$

missä $\delta(x)$ on Diracinkin deltafunktio.

Seuraavassa vaiheessa suoritetaan valinnainen uudelleenotanta. Kun uudelleenotanta tehdään jokaisella algoritmin iteraatiolla, on kyseessä SIS-algoritmi. Kun uudelleenotanta tehdään esimerkiksi efektiivisen otoskoon perusteella alla kuvatun kynnysarvoehdon $\hat{N}_{eff} < N_{th}$ täytessä, on kyseessä SIR-algoritmi. Tämä algoritmi on esitetty algoritmissa (1). Uudelleenotantaa tarkastellaan lähemmin alaluvussa 3.1.2. Lopuksi päivitetään aika (jos $k < t$) ja luodaan uudet ennusteet partikkeleille ehdotusjakaumasta (12)

$$x_{k+1}^i \sim q(x_{k+1}|x_k^i, y_{k+1}) \quad (12)$$

ja päivitetään partikkelen painot tärkeytysotannalla (13), sen mukaan kuinka todennäköisiä partikkelen ennusteet ovat

$$w_{k+1|k}^i = w_{k|k}^i \frac{p(x_{k+1}^i|x_k^i)}{q(x_{k+1}^i|x_k^i, y_{k+1})}. \quad (13)$$

Alla käsitellään algoritmiin liittyvän uudelleenotantamenetelmän, partikkelen määrän / otoskoon ja ehdotusjakauman valinta. Lopuksi esitetään algoritmin konvergenssia, marginaalijakaumaa sekä aikakompleksisuutta koskevia tuloksia.

Algorithm 1: Hiukassuodin

Result: Posteriorijakauman $p(x_{1:k}|y_{1:k})$ estimaatti.
Data: Havainnot y_k . Generoitu $x_1^i \sim p_{x_0}$ missä $i = 1, \dots, N$ ja jokainen partikkeli saa saman painon $w_{1|0}^i = 1/N$.

begin

```
for k = 1, 2, ..., t do
    for i = 1, 2, ..., N do
        begin
            | Päivitetään painot  $w_{k|k}$ .
        begin
            | Estimoidaan  $p$  laskemalla tiheydelle approksimaatio.
        begin
            | Lasketaan efektiivinen otoskoko  $\hat{N}_{eff}$ .
        if  $\hat{N}_{eff} < N_{th}$  then
            begin
                | Otetaan uudet  $N$  otosta palauttaen joukosta  $\{x_{1:k}^i\}_{i=1}^N$ , missä otoksen  $i$  todennäköisyys
                | on  $w_{k|k}^i$ .
        if  $k < t$  then
            begin
                | Aikapäivitys.
                | Luodaan ennusteet partikkeille ehdotusjakaumasta  $x_{k+1}^i \sim q(x_{k+1}|x_k^i, y_{k+1})$ ,
                | päivitetään partikkelienv painot tärkeytysotannalla.
```

3.1 Parametrien valinta

Ennen algoritmin suorittamista valitaan ehdotusjakauma $q(x_{k+1}|x_{1:k}, y_{k+1})$, uudelleenotantamenetelmä sekä partikkelienv määärä N . Ehdotusjakauman ja uudelleenotantamenetelmän valinnassa tärkeimpänä päämäääränä on välttää otosten ehtymistä, kun taas partikkelienv määärä säätelee kompromissia algoritmin suorituskyvyn ja tarkkuuden välillä.

3.1.1 Otoskoon N valinta

Yleispätevää sääntöä otoskoon/partikkelienv lukumääärän N valinnalle on vaikeaa antaa, sillä vaadittava estimointitarkkuus riippuu usein kässillä olevasta ongelmasta. Gordon & al. (1993) esittävät kuitenkin kolme tekijää, jotka vaikuttavat partikkelienv lukumääärän valintaan

- tila-avaruuden ulottuvuuksien lukumääärä n_x ,
- tyypillinen päälekäisyys priorin ja uskottavuuden välillä
- sekä tarvittava aika-askeleiden lukumääärä.

Ensimmäisen tekijän vaikutus on selvä. Mitä useammassa ulottuvuudessa otantaa tarvitsee tehdä, sen korkeammaksi on N asetettava, jotta jokainen ulottuvuuus pystytään kattamaan. Tekijät (b) ja (c) puolestaan seuraavat uudelleenotannasta. Jos se osa tila-avaruutta, jossa uskottavuus $p(y_k|x_k)$ saa merkittäviä arvoja on pieni verrattuna siihen osaan, jossa priorijakauma $p(x_k|y_{1:k-1})$ saa merkittäviä arvoja, suuri osa partikkeleista saa pieniä painoja eivätkä näin valikoidu uudelleenotantaan.

Yleisesti ottaen N kannattaa asettaa sellaiseksi, että se paitsi tuottaa riittävän tarkan estimaatin, on se käytettäväissä olevan laskentatehon sekä vaadittavan laskentanopeuden kannalta järkevä. Tähän palataan tutkielman lopuksi empiirisessä paikannusesimerkissä.

3.1.2 Uudelleenotantamenetelmän valinta

Ilman uudelleenotantaa on todennäköistä, että algoritmi alkaa kärsiä otosten ehtymisestä. Toisin sanoen kaikki painot alkavat keskityä vain muutamalle partikkeliille. Uudelleenotanta tarjoaa osittaisen ratkaisun tähän ongelmaan, mutta hävittää samalla informaatiota ja siten lisää satunnaisotantaan liittyvää epävarmuutta. Yleisesti ottaen kannattaa uudelleenotanta aloittaa vasta siinä vaiheessa algoritmin suorittamista, kun siitä on otosten ehtymisen kannalta hyötyä.

Jos alla esitetyssä algoritmissa uudelleenotanta suoritetaan jokaisella algoritmin läpikäynnillä on kyseessä ns. SIR-algoritmi (sequential importance resampling). Vaihtoehtona on hyödyntää tärkeytysotantaa ja suorittaa uudelleenotanta ainoastaan, kun otoskoon ehtymisen mittarina käytettävä efektiivinen otoskoko painuu jonkin kynnysarvon alapuolelle. Tätä kutsutaan SIS-algoritmiksi (sequential importance sampling).

Efektiivinen otoskoko saadaan laskettua variaatiokertoimesta c_ν kaavalla

$$N_{eff} = \frac{N}{1 + c_\nu^2(w_{k|k}^i)} = \frac{N}{1 + \frac{\text{Var}(w_{k|k}^i)}{(\mathbb{E}[w_{k|k}^i])^2}} = \frac{N}{1 + N^2 \text{Var}(w_{k|k}^i)}. \quad (14)$$

Nämä laskettu efektiivinen otoskoko maksimoituu ($N_{eff} = N$), kun kaikille painoille pätee $w_{k|k}^i = 1/N$ ja minimoituu ($N_{eff} = 1$), kun $w_{k|k}^i = 1$ todennäköisyydellä $1/N$ ja $w_{k|k}^i = 0$ todennäköisyydellä $(N-1)/N$. Tästä saadaan effektiiviselle otoskolle laskennallinen approksimaatio

$$\hat{N}_{eff} = \frac{1}{\sum_i (w_{k|k}^i)^2}. \quad (15)$$

Sekä määritelmälle (14) että (15) pätee $1 \leq \hat{N}_{eff} \leq N$. Yläraja saavutetaan, kun jokaisen partikkelin paino on sama. Alarajalle puolestaan päädytään, kun kaikki paino päätyy yksittäiselle partikkeliille. Tästä saadaan määritettyä algoritmile SIS-uudelleenotantaehto $\hat{N}_{eff} < N_{th}$. Gustafsson (2010) esittää uudelleenotannan kynnysarvoksi esimerkiksi $\hat{N}_{th} = 2N/3$.

Uudelleenotanta ei muuta approksimoitavan jakauma p odotusarvoa, mutta se lisää jakauman Monte Carlo -varianssia. On kuitenkin olemassa uudelleenotantamenetelmiä, jotka pyrkivät minimoimaan tämän varianssin lisäksi. Varianssitarkastelu jätetään tämän tutkielman ulkopuolelle.

3.1.3 Ehdotusjakauman valinta

Yksinkertaisin valinta ehdotusjakaumalle on $q(x_{1:k}|y_{1:k})$ eli toisin sanoen jokaisella algoritmin suorituskerralla käydään läpi koko polku $1 : k$. Tämä ei kuitenkaan ole tarkoituksemukaista, erityisesti jos kyseessä on reaalialkainen sovellus. Kirjoitetaan tämä ehdotusjakauma nyt muodossa

$$q(x_{1:k}|y_{1:k}) = q(x_k|x_{1:k-1}, y_{1:k})q(x_{1:k-1}|y_{1:k}). \quad (16)$$

Jos yhtälöstä (16) poimitaan ehdotusjakaumaksi ainoastaan termi $q(x_k|x_{1:k-1}, y_{1:k})$ saadaan tämän avulla muodostettua hyvä approksimaatio arvoille x_k . Tämä on suodinongelman kannalta riittävä, koska olemme kiinnostuneita ainoastaan posteriorijakaumasta ajanhetkellä k (tasoitusongelmassa tarvitsisimme koko polun $x_{1:k}$). Alla tarkastellaan edelleen Gustafsson (2010) seuraten kahta ehdotusjakauman valintatapaa, prioritantaa (prior sampling) sekä uskottavuusotantaa (likelihood sampling).

Ennen ehdotusjakauman tarkastelua määritellään mallille signaali-kohinasuhde uskottavuuden maksimin ja priorin maksimin välisenä suhteena

$$\text{SNR} \propto \frac{\max_{x_k} p(y_k|x_k)}{\max_{x_k} p(x_k|x_{k-1})}. \quad (17)$$

Yhdistetään lisäksi ehdotusjakaumia varten yhtälöt (9) ja (10), jolloin saadaan painojen päivitys muotoon (18).

$$w_{k|k}^i \propto w_{k-1|k-1}^i \frac{p(y_k|x_k^i)p(x_k|x^{k-1})}{q(x_k|x_{k-1}^i, y_k)} \quad (18)$$

Kun suhde (17) on matala, on prioritanta luonnollinen valinta. Tässä käytetään ehdotusjakauman tilavektorin ehdollista prioria eli

$$q(x_k|x_{1:k-1}, y_k) = p(x_k|x_{k-1}^i). \quad (19)$$

Yhtälön (19) perusteella saadaan edelleen prioritannan painoiksi (20)

$$w_{k|k}^i = w_{k-1|k-1}^i p(y_k|x_k^i) = w_{k-1|k-1}^i p(y_k|x_k^i). \quad (20)$$

Kun taas signaali-kohinasuhde on kohtalainen tai korkea, on parempi käyttää ehdotusjakaumana skaalattua uskottavuusfunktiota (22). Tarkastellaan ensin tekijöihin jakoa (ref{uskottavuusotanta-factorization}).

$$p(x_k|x_{k-1}^i, y_k) = p(y_k|x_k) \frac{p(x_k|x_{k-1}^i)}{p(y_k|x_{k-1}^i)} \quad (21)$$

Kun SNR on korkea ja uskottavuusfunktio on integroituva pätee $p(x_k|x_{1:k-1}, y_k) \propto p(y_k|x_k)$, jolloin voidaan asettaa (22)

$$q(x_k|x_{1:k-1}, y_k) \propto p(y_k|x_k). \quad (22)$$

Yhtälön (22) perusteella saadaan edelleen uskottavuusotannan painoiksi (23).

$$w_{k|k}^i = w_{k-1|k-1}^i p(x_k^i|x_{k-1}^i). \quad (23)$$

3.2 Konvergenssituloksia

Alla esitetään kaksi algoritmiin liittyvää konvergenssitulosta, se kuinka hyvin esitetyllä SMC-algoritmillä arvioitu posterioriteheys $\hat{p}(x_{1:k}|y_{1:k})$ approksimoi todellista tiheysfunktiota $p(x_{1:k}|y_{1:k})$ sekä mikä on approksimaation keskineliövirhe. Tulokset noudattavat Crisanin ja Doucet'n artikkeleita "Convergence of Sequential Monte Carlo Methods" (2000) ja "A Survey of Convergence Results on Particle Filtering Methods for Practitioners" (2002).

Konvergenssitulos 1: Kun $N \rightarrow \infty$ algoritmilelle pätee $\forall k$ tulos (24).

$$\hat{p}(x_{1:k}|y_{1:k}) \xrightarrow{a.s.} p(x_{1:k}|y_{1:k}). \quad (24)$$

Konvergenssitulos 2: Keskineliövirheelle pätee asymptoottinen konvergenssi (25).

$$\mathbb{E}(\hat{g}(x_k) - \mathbb{E}(g(x_k)))^2 \leq \frac{p_k \|g(x_k)\|}{N}. \quad (25)$$

Missä g on mikä hyvänsä piilossa olevan tila-avaruuden rajoitettu Borel-mitallinen funktio ($g \in \mathcal{B}(\mathbb{R}^{n_x})$), $\|g(\cdot)\|$ kyseisen funktion supremum-normi ja p_k jokin äärellinen vakio, jolle pätee ajanhetkestä k riippumatta $p_k = p < \infty$. Konvergenssituloksia ei tämän tutkielman puitteissa todisteta.

3.3 Marginaalijakauma

Edellä kuvattu algoritmi 1 tuottaa approksimaation koko prosessin posteriorijakaumalle $p(x_{1:k}|y_{1:k})$. Jos halutaan tietää ainoastaan posteriorijakauman $p(x_k|y_{1:k})$ estimaatti, voidaan käyttää ainoastaan viimeisestä tilasta x_k^i laskettua estimaattia

$$\hat{p}(x_k|y_{1:k}) = \sum_{i=1}^N w_{k|k}^i \delta(x_k - x_k^i). \quad (26)$$

Toinen vaihtoehto on käyttää laskennassa tärkeytyspainoa

$$w_{k+1|k}^i = \frac{\sum_{j=1}^N w_{k|k}^j p(x_{k+1}^i|x_k^j)}{q(x_{k+1}^i|x_k^i, y_{k+1})} \quad (27)$$

yllä esitetyn sijaan. Tällöin jokaisessa aikapäivitysaskeleessa lasketaan painot kaikkien mahdollisten tila-aika-avaruuspolkujen yli. Samoin kuin uudelleenotanta tämä pienentää painojen varianssia.

3.4 Aikakompleksisuus

Algoritmin perusmuodon aikakompleksisuus on $\mathcal{O}(N)$. Uudelleenotantamenetelmän tai ehdotusjakauman valinta ei suoraan vaikuta aikakompleksisuuteen. Sen sijaan marginalisointi edellä esitettyllä tärkeytyspainolla lisää algoritmin aikakompleksisuutta $\mathcal{O}(N) \rightarrow \mathcal{O}(N^2)$, koska jokaisen partikkelin kohdalla painot lasketaan jokaisen tila-aika-avaruuspolun yli. On selvää, että erityisesti isoilla otoskoon N arvoilla ei yllä esitetty marginalisointi enää ole mielekästä.

Tällaisia tilanteita varten SMC-algoritmista on olemassa $\mathcal{O}(N \log(N))$ -versioita, jotka perustuvat N:n kappaleen oppimiseen (N-body learning). Näiden algoritmien käsitteily jää tämän tutkielman ulkopuolelle, mutta katsauksen algoritmeista ovat esittäneet esimerkiksi Klaas & al. artikkelissa “Toward Practical N^2 Monte Carlo: the Marginal Particle Filter” (2012). Tutkielman kahdessa ulottuuviuudessa tapahtuvan paikannusesimerkin tapauksessa $\mathcal{O}(N^2)$ -aikakompleksisuus ei ole ongelma.

Lisäksi algoritmin aikakompleksisuuteen vaikuttavat seuraavat sekat. T. SUORITUSKYKY. BATCH VS ONLINE. Kun halutaan on-line-estimointia, siinä missä off-line-estimointiin.

4 Paikannusesimerkki

Sisätilapaikannus *** on BLAHBLAHBLAH. Koska GPS ei toimi sisätiloissa, tarvitaan muita paikannusratkaisuja. Radiosignaaleihin perustuvat paikannusratkaisut ovat BLAHBLAH. Eräs mahdollinen XXX, jossa käytetään tulokulmaa paikantamaan. Walkbase käyttää sisätilapaikannusta erityisesti ruokakaupoissa sekä tavarataloissa asiakkaiden käyttäytymistä koskevan keräämiseen. Tyypillinen .. XXX ostoskärryihin ja -koreihin XXX keräämään tärkeää paikannusdataa asiakkaiden XXX. Markkinoilla on XXX on Walkbase kehittänyt oman laitteistoratkaisunsa. SKENAARIOISSA TARVITAAN Tavoitteena on alle metrin paikannusvirhe.

Esimerkissä käytetään SMC-algoritmia Bluetooth-paikannussovelluksessa paikannustarkkuuden parantamiseen. Paikannukseen käytettävä data kerätään toimistoympäristössä liikkuvien BLE-lähettimien sekä kattoon sijoitettujen vastaanottimien avulla. Havainnot koostuvat vastaanottimien lähettimien signaalien perusteella laskemista, BLE5.1-standardin mukaisista signaalien tulokulmista eli AoA-havainnoista (angle of arrival). Paikannukseen käytetään triangulaatio-algoritmia. Lopuksi esimerkissä analysoidaan ja vertaillaan algoritmin eri versioiden suorituskykyä sekä suorituskyvyn että paikannustarkkuuden näkökulmasta.

4.1 Koeasetelma

Paikannusesimerkissä lähettimenä toimi 25 Bluetooth-paikannustagista koostuva Walkbase Foculator -testilaite (kuva 1), vastaanottimena toimistoympäristöön asennettut seitsemän Walkbase XR-2 -vastaanotinta

(kuva 2). Jokainen vastaanotin sisältää kahdeksan antennia, joiden vastaanottamien lähetinsignaalien perusteella vastaanottimet laskevat signaalin tulokulman suhteessa vastaanottimeen. Tarkka tulokulmien laskemiseen käytetty algoritmi on paikantimen antennit toimittaneen Silicon Laboratories, Inc. -yrityksen liikesalaisuus, mutta ***perusperiaate on.

AoA relies on a single-antenna transmit beacon with continuous tone extension appended to a Bluetooth packet transmission and a locator receiver device to measure the arrival angle of the signal using an array of antennas. Each antenna in the array sees phase differences due to different line of sight distances to the beacon device. The antennas in the array are switched during continuous tone extension, resulting in IQ samples with phase information for each antenna. This IQ-data is fed to an angle estimator algorithm. In this AoA use case, the receiving device tracks arrival angles for individual transmit beacon objects.

Esteettömässä ympäristössä XXX kulman virhe on hyvin pieni ja paikannusongelma voidaan ratkaista suoraan jollakin triangulaatio-algoritilla. Eräänä XXX ToTal-algoritmia. Tässäkin tilanteessa voi paikannusta parantaa suodattimen käytöllä, mutta jo triangulaatioä-algoritmin perusversio tuottaa halutun tarkkuuden. Toimistoympäristö on kuitenkin haastava, sillä erityisesti näyttöruumut sekä heijastavat että estävät radiosignaleja. Silicon Labs lupaa omalle AoA-järjestelmälle vastaavassa toimisympäristössä (seitsemällä paikantimella) kulmavirheen välillä $\pm 3.7 - 5.7$. Tämä ei kuitenkaan riitä johdonmukaisesti haluttuun alle metrin paikannustarkkuuteen, joten AoA-paikannus toimistoympäristössä tarjoaa hyvän motivaation SMC-menetelmien käyttämiseen.

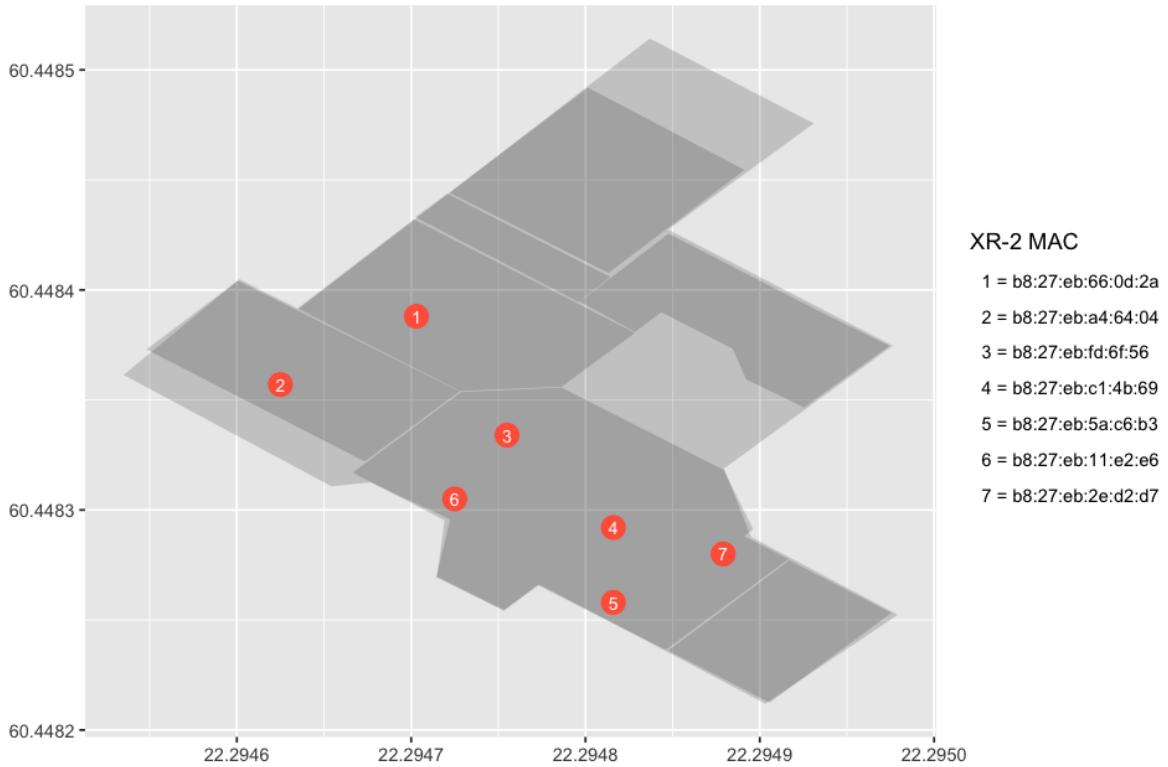


Kuva 1: Walkbase Foculator



Kuva 2: Walkbase XR-2

Data on kuvattu tarkemmin alaluvussa 4.2. Koeympäristön pohjapiirustus on esitetty kuvassa (3). Piirustuksessa XR-2-paikantimet on kuvattu punaisilla ympyröillä. Foculator-testilaitteen liikerata on kuvattu sinisellä viivalla. Toimistoympäristön pohjapiirustus on kuvattu harmaalla niin, että piirustuksesta erottuvat toimiston väliseinät.



Kuva 3: Koeasetelman pohjapiirustus

Ennen datan keräämistä testilaite asennettiin kiinni robottiin (kuva 4) 1.5 metrin korkeudelle. Robotti oli ohjelmoitu ennalta seuraamaan haluttua liikerataa vastaavia lattiamerkintöjä ennalta määritetyllä nopeudella Xm/s . Kyseinen liikerata on merkitty koeympäristön pohjapiirustukseen sinisellä katkoviivalla. Data kerättiin sinä aikana, kun robotti liikkui liikeradan lähtöpisteestä (piste A) liikeradan maalipisteeseen (piste B). Robotilta kului matkaan Y sekuntia. Sekä korkeus että nopeus valittiin niin, että ne ovat XXX TÄ-
GILLÄ MERKITYN ostoskärryjen liikkumista kauppaympäristössä. HUOMIO: Testaaminen vaikeammassa ympäristössä.



Kuva 4: Testirobotti

Data kerättiin yöaikana, jolloin toimiston käyttöaste oli minimissa. Tällä minimoitiin radiosignaalien tielle osuvien ihmisten vaikutus signaaleihin.

4.2 Datan kuvaus

Riippuen päällä olevien paikannustagien lukumäärästä, tuottaa n. havainto sekunnissa. Havaintoja on $N_{obs} = X$ kappaletta, joka kattaa Y minuuttia. Havaintojen aikaleimat on tallennettu sekunnin tuhannesosan tarkkuudella. Jokainen havainto koostuu taulukossa 1 havaintuista muuttujista.

Muuttuja	Kuvaus	Esimerkkiarvo
ts	havainnon aikaleima	21:38:20.998+00
asset_tag_mac	lähettimen MAC-osoite	5c:02:72:67:f7:4c
locator_mac	vastaanottimen MAC-osoite	b8:27:eb:66:0d:2a
locator_lat	vastaanottimen sijainti (leveyspiiri)	60.448265
locator_lon	vastaanottimen sijainti (pituuspiirit)	22.294823
bearing	suuntimakulma η (astetta)	34
height	vastaanottimen korkeus (m)	2.22
rssi	signaalin vahvuus (dBm)	-81
distance	arvioitu etäisyys lähettimeen (m)	13.5
azimuth_angle	atsimiuuttikulma ϕ (astetta)	102.7
converted_angle	napapohjoisesta laskettu kulma Φ (astetta)	34
opposite_angle	vastakkainen kulma (lähetin-vastaanotin) Φ' (radiaania)	3.735
elevation_angle	korkeuskulma (astetta)	11.9
quality_ant_phase_jitter	antennikohtainen jitter	(0.15,0.15,0.15,0.15, 0.14,0.16,0.29,0.34)
ss_jitter	antennikohtaisten jitter-arvojen neliösumma	0.3349
quality_ant_snr	antennikohtainen signaali-kohinasuhde	(24.76,37.48,40.75,36.37 28.5,28.2,16.15,11.27)
snr_jitter	antennikohtaisten signaali-kohinasuhteiden neliösumma	6996.473

Taulukko 1: Havaintomuuttujat

Etäisyys on estimoitu signaalin vahvuudesta käyttäen valmistajan omaa, tuntematonta propagaatiomallia. Etäisyyttä tai signaalin vahvuutta ei käytetä paikantamiseen, joten tämän mallin käsitteily jäätetään tutkielman ulkopuolelle. Munoz (2009) luku 2 sisältää yleiskatsauksen myös propagaatiomalleihin.

Atsimiuuttikulma ϕ lasketaan aina vastaanottimen tietyltä sivulta, joten se vastaa napapohjoista ainoastaan siinä tapauksessa, että vastaanottimen kyseinen sivu on asetettu kohtisuoraan napapohjoiseen nähdyn. Käytännön syistä tämä ei ole aina mahdollista eikä edes haluttavaa. Sen vuoksi jokaiselle vastaanottimelle on tietokantaan tallennettu oma suuntimakulma η . Triangulaatiossa alla käytetään napapohjoisesta laskettuja kulmia Φ , jotka lasketaan jokaiselle havainnolle havainnon vastaanottimen suuntimakulman avulla

$$\Phi = (\phi + \eta + 360^\circ) \mod 360^\circ. \quad (28)$$

Suuntimakulma Φ kertoo kuitenkin vastaanottimen ja lähetimen välisen kulman. Koske olemme kiinnostuneet paikantamaan lähetintä, lasketaan yhtälöllä (29) suuntimakulman perusteella edelleen lähetimen ja vastaanottimen välinen kulma Φ' . Samalla asteissa oleva kulma muunnetaan radiaaneiksi paikannusta varten.

$$\Phi' = \frac{\pi}{180^\circ}(\Phi + 180^\circ \mod 360^\circ). \quad (29)$$

Lisäksi havainnot sisältävät antennikohtaiset jitter- ja SNR-arvot sekä näistä lasketut neliösummat. Havaintomuuttujien ohella koitelanteessa tallennettiin Foculator-testilaitteen todellinen sijainti sekunnin tarkkuudella. Tallennettu sijainti perustui koeympäristön lattiaan pohjapiirrustusten ja laser-mittausten avulla tehtyihin merkintöihin. Näin saadut testimuuttujat on kuvattu taulukossa (2).

Muuttuja	Kuvaus	Esimerkkiarvo
foculator_ts	aikaleima	21:38:20+00
foculator_lat	testilaitteen sijainti (leveyspiiri)	60.44819
foculator_lon	testilaitteen sijainti (pituuspiiri)	22.29493

Taulukko 2: Testimuuttujat

Testimuuttuja käytetään alla SMC-paikannusalgoritmin paikannusvirheen laskemisessa.

4.3 Ongelman kuvaus

Tarkoituksesta on estimoida testilaitteen/robotin sijainti sekunnin tarkkuudella. Merkitään estimointavaa tilasarjaa $x_{1:k} = \{x_1, \dots, x_k\}$. Lisäksi merkitään x_o testilaitteen lähtösijaintia. Jokainen tilasarjan havainto koostuu sekä pituus- että leveyskoordinaateista. Määritellään tilalle robotin kulkua kuvaava Markov-malli (30):

$$x_{k+1} = f(x_k, \nu_k) = x_k + D_k \begin{bmatrix} \cos \psi_k \\ \sin \psi_k \end{bmatrix} + \nu_k \quad (30)$$

missä D_k on testilaitteen/robotin ajanhetkenä k kuljema matka ja ψ_k robotin atsimuutti/suuntimakulma kyseisenä ajanhetkenä. ν_k on kohinaa, NORMAALIOLETUS. Vastaavasti $y_{1:k} = \{y_1, \dots, y_k\}$ kuvailee paikantimiensä ja lähettimien välillä laskettuja MUUTTUJAT.

$$y_k = h(x_k) + e_k = \text{MULTIANGULAATIONMALLI} \quad (31)$$

MALLIN VALINTA.

4.4 Algoritmi

Koeasetelmassa käytetty SMC-algoritmi on toteuttu koetta varten alusta asti R-kielellä. Daten käsitteelyyn koodi käyttää R:n tidyverse-kirjastoa. MUUT KÄYTETYT KIRJASTOT. Algoritmin koodi löytyy tutkielman liitteestä 1.

Algoritmin toteutuksessa on otettu huomioon seuraavat suorituskykyyn liittyvät pullonkaulat.

4.5 Tulokset

Koe toistettiin K kertaa. Ensimmäisessä vaiheessa $N = 1000$ JATKA. Vertailuarvoina käytettiin puhataaseen triangulaatioon (TOTAL) perustuvaa XXX. Paikannusten keskivirheet on esitetty taulukossa (3). Parhaimman suorituskyvyn PAIKANNUSMIELESSÄ algoritmi valittiin lisätarkasteluun SMC-algoritmien parametrinvalintaan KOSKIEN.

Koeasetelma	Algoritmi	Keskimääriinen paikannusvirhe ϵ	Sijaintien lukumäärä k
Testilaite paikoillaan	ToTaL		
Testilaite paikoillaan	SIS		
Testilaite paikoillaan	SIR		
Liikkova testilaite	ToTaL		
Liikkova testilaite	SIS		
Liikkova testilaite	SIR		

Taulukko 3: Paikannusvirheet

5 Lopuksi

Tässä tutkielmanmassa on esitetty pääpiirteittäin SMC-menetelmien teoria Bayesilaisessa/tilastotieteellisessä viitekehysessä. Lisäksi tutkielmanmassa on käyty läpi kaksi versiota hiukassuodinalgoritmista, jatkuvaan uudelleenotantaa hyödyntävä SIS-algoritmi sekä uudelleenotantaa efektiivisen otoskoon perusteella hyödyntävä SIR-algoritmi. Tutkielmanmassa on myös käyty lävitse joitain algoritmien parametrien valintaan, suorituskykyyn sekä konvergenssiin liittyviä tuloksia.

Tutkielmanmassa on lisäksi tarkasteltu miten eri valinnat vaikuttavat algoritmin suorituskykyyn yksinkertaisen mutta kattavan ja todelliseen ongelmaan sekä dataan perustuvan paikannusesimerkin avulla. Jatkossa tutkielmanmassa olisi mahdollista laajentaa esimerkiksi JATKOKYSYMYKSIÄ.

6 Lähteet

- Chen & Liu (1998): **Sequential Monte Carlo Methods for Dynamic Systems.** http://stat.rutgers.edu/home/rongchen/publications/98JASA_SMC.pdf
- Crisan & Doucet (2000): **Convergence of Sequential Monte Carlo Methods.** http://www.stats.ox.ac.uk/~doucet/crisain_doucet_convergenceofSMC2000.pdf
- Crisan & Doucet (2002): **A Survey of Convergence Results on Particle Filtering Methods for Practitioners.** https://www.researchgate.net/publication/3318241_A_Survey_of_Convergence_Results_on_Particle_Filtering_Methods_for_Practitioners.
- Dahlin & Schön (2019): **Getting Started with Particle Metropolis-Hastings for Inference in Nonlinear Dynamical Models.** <https://arxiv.org/pdf/1511.01707.pdf>.
- Davidson, Collin & Takala (2010): **Application of Particle Filters for Indoor Positioning Using Floor Plans.** http://www.tkt.cs.tut.fi/research/nappo_files/Davidson_UPINLBS2010.pdf.
- Munoz, Bouchereau Lara, Vargas & Enriquez-Calderá (2009): **Position Location Techniques and Applications.** Elsevier.
- Del Moral (1996): **Nonlinear Filtering: Interacting Particle Resolution.** <https://people.bordeaux.inria.fr/pierre.delmoral/delmoral96nonlinear.pdf>.
- Gordon, Salmond & Smith (1993): **Novel approach to nonlinear/non-Gaussian Bayesian state estimation.** <http://www.irisa.fr/aspi/legland/ref/gordon93a.pdf>.
- Gustafsson (2010): **Particle Filter Theory and Practice with Positioning Applications.** <https://ieeexplore.ieee.org/document/5546308>.
- Klaas, De Freitas, Doucet (2012): **Toward Practical N2 Monte Carlo: the Marginal Particle Filter.** <https://arxiv.org/abs/1207.1396>
- Petris, Petrone & Campagnoli (2009): **Dynamic Linear Models with R.** Springer.
- Särkkä (2013): **Bayesian Filtering and Smoothing.** Cambridge University Press. https://users.aalto.fi/~ssarkka/pub/cup_book_online_20131111.pdf.

7 Liite 1: SMC-paikannusalgoritmin R-koodi