TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

VILLE PARKKINEN

ITSEORGANISOITUVA KARTTA

Kandidaatintyö

Tarkastaja: Simo Ali-Löytty
3.5.2018

TIIVISTELMÄ

VILLE PARKKINEN: Itseorganisoituva kartta
Tampereen teknillinen yliopisto
Kandidaatintyö, 17 sivua, 1 liitesivua
Toukokuu 2018
Teknis-luonnontieteellinen koulutusohjelma
Pääaine: Matematiikka
Tarkastajat: Tohtori Simo Ali-Löytty
Avainsanat: Itseorganisoituva kartta, koneoppiminen, klusterointi, data-analyysi

Itseorganisoituva kartta on korkeaulotteisen datan visualisointiin käytettävä neuroverkkomalli. Itseorganisoituva kartta perustuu ohjaamattomaan oppimiseen: algoritmi etsii datasta samankaltaisista alkioista koostuvia ryppäitä eli klustereita, jotka se kuvaa kaksiulotteiselle kartalle säilyttäen datassa esiintyvät klustereiden väliset topologiset suhteet. Itseorganisoituva kartta on laajasti käytössä eri tieteen- ja teollisuuden aloilla.

Kartan alkutilan parametrit sekä opetusvaiheen valinnat vaikuttavat merkittävästi kartan lopputilaan. Kartan muoto ja koko, sekä mallivektorien alkuarvot ovat merkittävimpiä alkutilan parametreja. Opetusvaiheessa myös naapuruusfunktion valinta vaikuttaa erityisesti klustereiden välisten suhteiden näkymiseen lopullisella kartalla.

Tässä työssä käsitellään itseorganisoituvan kartan teoreettista taustaa sekä oleellisimpia käytännön kysymyksiä liittyen alkutilan ja opetusvaiheen version valintaan. Opetusvaiheen versioista tarkastellaan kahta yleisintä, askelittaista ja sarjalaskentaista versiota. Lopuksi rakennetaan esimerkkinä itseorganisoituva kartta WLAN-signaalien voimakkuusmittauksista.

SISÄLLYS

LYHENTEET JA MERKINNÄT

M	kartan solmujen lukumäärä
s_i	kartan solmu
n	syötedatan vektorien ulottuvuus
k	syötedatan vektorien lukumäärä
X	syötedatamatriisi, koko $k \times n$
t	opetuskierroksen numero
T	opetuskierrosten kokonaismäärä
$x(t)$	opetuskierroksella t syötettävä syötedatavektori
m_i	solmuun s_i liittyvä mallivektori
$m_i(t)$	mallivektorin m_i arvo opetuskierroksella t
m_i^*	mallivektorin m_i lopullinen arvo
h_{ji}	kartan solmujen s_j ja s_i välinen naapuruusfunktio
E	odotusarvo
c	voittajasolmun indeksi
$\alpha(t)$	oppimisnopeus
N_i	solmun s_i naapuruston solmujen indeksien joukko
n_i	sarjalaskentaisessa algoritmissa solmun s_i listan syötevektorien lukumäärä

1. JOHDANTO

Tietokoneiden laskentatehon ja muistin määrän kasvaessa myös saatavilla olevan datan määrä on kasvanut viime vuosikymmeninä räjähdysmäisesti. Tätä voidaan käyttää hyväksi eri tieteenaloilla sekä teollisuudessa, mutta suuriin määriin liittyy kuitenkin myös ongelma: ihmisaivot eivät pysty sellaisenaan käsittelemään suuria datamääriä, joten datan joukosta on poimittava oleellinen tieto, ja se on esitettävä yksinkertaisemmassa, ihmisen ymmärtämässä muodossa.

Yllä mainitun ongelman ratkaisemiseksi voidaan käyttää erilaisia koneoppimisen menetelmiä, kuten klusterointia ja ulottuvuuksien vähentämistä. Klusteroinnissa tietoalkiot jaotellaan ennalta määräämättömiin ryhmiin siten, että samankaltaiset alkiot päätyvät keskenään samaan ryhmään. Klusterointiongelmaa ratkovat algoritmit siis etsivät datasta sisäisiä rakenteita ja luovat ryhmät niiden perusteella. Ulottuvuuksien vähentämisellä puolestaan tarkoitetaan korkeauloitteisen datan kuvaamista yksinkertaisempaan, matalaulotteiseen avaruuteen.

Eräs työkalu edellä mainittujen menetelmien toteuttamiseen on Teuvo Kohosen 1980-luvulla julkaisema itseorganisoituva kartta [8]. Itseorganisoituva kartta on neuroverkkomalli, joka perustuu ohjaamattomaan oppimiseen. Se kuvaa korkeaulotteisen datan matalaulotteiselle, usein kaksiulotteiselle kartalle siten, että tärkeimmät alkioiden väliset topologiset ja metriset suhteet säilyvät. Itseorganisoituva kartta on yksi siteeratuimmista suomalaisista tieteellisisistä tuloksista, ja se on laajasti käytössä luonnontieteiden ja lingvistiikan tutkimuksessa sekä teollisuudessa ja rahoitusalalla [9].

Tässä työssä tarkastellaan itseorganisoituvan kartan alustusta, opettamista sekä valmiin kartan käyttämistä klusterointiin ja ulottuvuuksien vähentämiseen. Lisäksi karttaa sovelletaan WLAN-verkkojen signaalien voimakkuusmittauksissa kerättyyn dataan.

2. YLEISKATSAUS ITSEORGANISOITUVAAN KARTTAAN JA SEN KÄYTTÖTARKOITUKSIIN

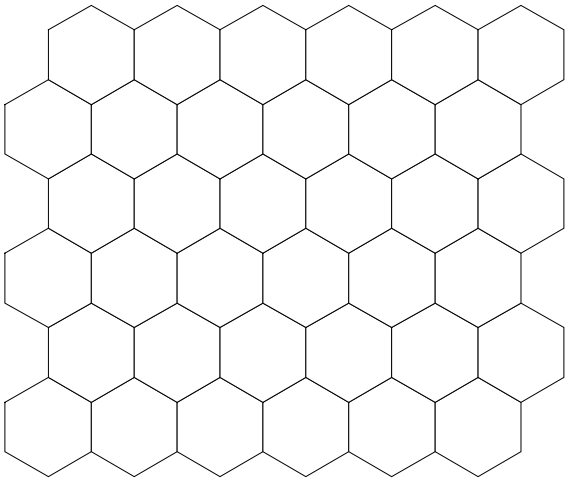
Itseorganisoituva kartta (Self-Organizing Map, SOM) on algoritmi, jonka avulla voidaan visualisoida korkeaulotteista dataa [8]. Algoritmi koostuu kolmesta vaiheesta: alustus, opetus ja luokittelu. Alustusvaiheessa luodaan kartta, joka koostuu solmuista (Node). Solmut on järjestetty kartaksi, joka on yleensä kaksiulotteinen ruudukko. Tyypillinen ruudukko on esitetty kuvassa 2.1. Kuhunkin solmuun liittyy mallivektori (Model). Algoritmin opetusvaiheessa tavoitteena on mallivektorien arvoja muokkaamalla saada aikaan kartta, joka kuvaa mahdollisimman tarkasti syötedatan alkioiden välisiä topologisia suhteita. Mallivektorien dimensio on sama kuin syötevektoreilla, ja niiden jakauma antaa approksimaation syötedatan jakaumasta.

Valmiissa kartassa lähekkäin oleviin solmuihin kuuluvat mallivektorien keskinäinen etäisyys syötedatan avaruudessa on pieni, ja kaukana toisistaan olevien solmujen mallivektorit ovat kaukana toisistaan myös syötedatan avaruudessa. Kartta pyrkii löytämään syötedatan sisäisestä jakaumasta ryppäitä eli klustereita. Luokitteluvaiheessa jokaiselle luokiteltavan datan alkiolle etsitään kartan mallivektoreista se, joka on lähimpänä kyseistä alkiota. Alkio katsotaan kuuluvaksi tätä mallivektoria vastaavan solmun osoittamaan luokkaan. Näin voidaan luokitella suuriakin datamääriä samankaltaisten alkioiden ryhmiin ja saada kokonaiskuva datasta ryhmien välisiä suhteita tarkastelemalla.

Vastaavanlaiseen luokitteluun voidaan käyttää muitakin menetelmiä, kuten perinteistä vektorikvantisointia (vector quantization) [4]. Itseorganisoituvan kartan erottaa muista vastaavista menetelmistä sen kyky järjestää mallivektorit siten, että samankaltaiset mallivektorit kuuluvat kartalla lähekkäin sijaitseviin solmuihin, ja vastaavasti toisistaan poikkeavat mallivektorit kuuluvat kaukana toisistaan oleville solmuille [8].

Yleisessä tapauksessa syöte- ja mallivektorien ei välttämättä tarvitse olla vektoreita, kunhan syöte- ja mallivektorien välille voidaan määrittää jokin etäisyysfunktio. Itseorganisoituvaa karttaa voidaan käyttää myös esimerkiksi merkkijonojen analysointiin [8], mutta jatkossa tässä työssä keskitytään vektorimuotoiseen dataan ja

etäisyyden mittana käytetään euklidista etäisyyttä.



Kuva 2.1 *Tyypillinen kuusikulmainen itseorganisoituva kartta. Kuvassa esitetty* 6×6 *-kokoinen kartta, jossa kukin kuusikulmio vastaa yhtä kartan solmua.*

3. ITSEORGANISOITUVAN KARTAN ALUSTUS

Lopputila, jota kohti itseorganisoituva kartta suppenee, riippuu useista eri alkutilan parametreista, kuten kartan solmujen lukumäärästä, keskinäinestä järjestyksestä, mallivektorien alkuarvoista sekä naapuruusfunktion valinnasta [7]. Hyvään lopputulokseen pääseminen edellyttää yleensä useita yrityksiä: eri alkutiloja kokeilemalla pyritään saavuttamaan käyttötarkoitukseen parhaiten sopiva lopputulos. Tässä luvussa käsitellään sekä kartan muotoa ja solmujen lukumäärää että solmuihin liittyvien mallivektorien alkuarvoja. Naapuruusfunktiota puolestaan käsitellään luvussa 4.1.

3.1 Mallivektorien alustus

Itseorganisoituva kartta suppenee, vaikka mallivektorien komponenttien arvot $m_i(0)$ alustettaisiin sattumanvaraisesti. Hyvän alkutilan valinnalla voidaan kuitenkin saada algoritmin suoritusaika jopa kertaluokkaa nopeammaksi [8]. Satunnaista alustusta käytetään lähinnä osoittamaan, että algoritmi suppenee myös mielivaltaisesta alkutilanteesta, ja että käytännön sovelluksissa alkutila kannattaakin valita sellaiseksi, että mallivektorit ovat jo valmiiksi likimäärin järjestyksessä. Hyvän alkutilan valintaan on kehitetty useita eri menetelmiä, joista yleisimpiä ovat lineaariseen projektioon sekä k-means-algoritmiin perustuvat menetelmät [1].

Itseorganisoituvan kartan mallivektorien alustuksessa käytetään tyypillisesti hyväksi pääkomponenttianalyysiä: mallivektorit alustetaan tasaiseen ruudukkoon k kappaleesta syötedatavektoreita koostuvan matriisin $X = [x(0), \ldots, x(k-1)]$ kahden suurimman ominaisarvon omaavan komponentin virittämälle tasolle. Yllä mainittua menetelmää käyttäen mallivektorien komponentit antavat jo alkutilanteessa hyvän approksimaation kartan lopputilasta, sillä kyseinen projektio säilyttää syötedatan merkittävimmät keskinäiset suhteet [2].

Alustamalla mallivektorit säännölliseen suorakulmaiseen ruudukkoon hukataan kuitenkin osa lineaarisen projektion tuottamista tuloksista. Onkin esitetty, että ottamalla huomioon syötevektorien jakauma edellisessä kappaleessa mainitulla tasolla

ja alustamalla mallivektorit epäsäännölliseen ruudukkoon jakauman mukaisesti voidaan päästä parempiin lopputuloksiin [1].

3.2 Kartan muoto ja koko

Yleisimmin itseorganisoituvan kartan solmut järjestetään kaksiulotteiseksi, kuusikulmaiseksi ruudukoksi, joka on esitetty kuvassa 2.1. Syynä kuusikulmaisuuteen on se, ettei kuusikulmainen ruudukko suosi kartan vaaka- ja pystysuuntia vastaavia komponentteja yhtä paljon kuin ilmeisin valinta, nelikulmainen ruudukko [8]. Ruudukon vaaka- pystysuuntaisten pituuksien suhteen tulee olla likimäärin sama kuin syötedatamatriisin kahden suurimman pääkomponentin ominaisarvojen suhteen.

Yllä kuvatun perusmuodon lisäksi on myös olemassa erityistilanteisiin sopivia, poikkeavia kartan muotoja. Esimerkiksi toroidin tai pallon muotoista karttaa voidaan käyttää luonnostaan syklisen datan esittämiseen tai häivyttämään kartan reunoilla esiintyviä vääristymiä ja epäjatkuvuuskohtia [10]. Kolmiulotteinen kartta puolestaan on todettu käyttökelpoiseksi esimerkiksi georeferoidun, eli maantieteellisen sijaintiin sidotun datan visualisointiin [3] sekä liikkuvan kuvan analysointiin [11].

Solmujen lukumäärä riippuu syötedatan laadusta ja kartan käyttötarkoituksesta. Suurta syötedatamäärää analysoitaessa pienikin määrä solmuja voi riittää, mikäli datassa esiintyy vain muutamia suuria klustereita. Suuriin klustereihin voi kuitenkin jäädä piiloon datan pienempiä sisäisiä rakenteita, joten ensimmäinen hyvältä vaikuttava lopputulos ei välttämättä ole paras mahdollinen. Tyypillisen itseorganisoituvan kartan koko on muutamasta kymmenestä muutamaan sataan solmuun. Erikoistapauksissa koko voi kuitenkin nousta tuhansista jopa miljoonaan solmuun [10].

4. ITSEORGANISOITUVAN KARTAN VERSIOT

Tässä kappaleessa tarkastelemme kahta tärkeintä itseorganisoituvan kartan versiota. Merkitään tarkasteluissa syötedataa jonona n-ulotteisia vektoreita $x(t)$, jossa t on opetuskierroksen numero. Mallivektoreille käytetään merkintää $m_i \in \mathbb{R}^n$, jossa i on itseorganisoituvan kartan kyseistä mallivektoria vastaavan solmun indeksi. Jokaiseen mallivektoriin liittyy jono $m_i(t)$. Mallivektorin m_i alkuarvo on $m_i(0)$ ja lopullinen arvo $m_i(T)$, kun T on opetuskierrosten kokonaismäärä.

4.1 Askelittainen algoritmi

Alkuperäisessä, askelittaisessa itseorganisoituvan kartan versiossa syötedatan vektorit $x(t)$ käsitellään yksitellen, ja karttaa päivitetään jokaisen syötevektorin yhteydessä. Se on yksinkertainen ymmärtää ja toteuttaa, mutta hitaampi kuin myöhemmin kappaleessa 4.2 esitettävä sarjalaskentainen versio.

Askelittainen algoritmissa jokaisella opetuskierroksella mallivektorin m_i arvo päivitetään kaavalla

$$m_i(t+1) = m_i(t) + h_{ci}(t)[x(t) - m_i(t)], \tag{4.1}$$

jossa h_{ci} on naapuruusfunktio, jolle pätee $h_{ci}(t) \to 0$, kun $t \to \infty$ ja solmujen välinen etäisyys $\|s_c - s_i\| \to 0$. Naapuruusfunktion alaindeksi c puolestaan määritellään kaavalla

$$c = \arg\min_i \{\|x(t) - m_i(t)\|\}, \tag{4.2}$$

eli c on sen mallivektorin indeksi, jonka euklidinen etäisyys syötevektoriin $x(t)$ on lyhin. Kyseistä mallivektoria kutsutaan voittajavektoriksi (best-matching model, winner). Mallivektoreita siis siirretään kohti syötettyä vektoria, ja naapuruusfunktio sekä mallivektorin etäisyys syötevektoriin määräävät kuinka paljon kutakin mallivektoria siirretään.

Naapuruusfunktion valintaan on useita eri vaihtoehtoja. Naapuruusfunktio määritellään aina kullekin opetuskierrokselle ja kahden solmun välille. Oleellisinta on, että sen arvo pienenee solmujen välisen etäisyyden kasvaessa. Itseorganisoituvan kartan kyky säilyttää syötedatan topologiset suhteet on seurausta juurikin tästä: kartalla vierekkäisiin solmuihin liittyvät mallivektorit alkavat opetuksen myötä muistuttaa enemmän toisiaan kuin kaukana toisistaan oleviin solmuihin liittyvät. Näin myös valmiilla kartalla vierekkäisiin solmuihin luokiteltavat datavektorit ovat samankaltaisia. Naapuruusfunktion arvon tulee myös pienentyä opetuskierroksen indeksin t kasvaessa, eli opetuksen alkuvaiheessa mallivektorien siirtymät ovat suurempia kuin loppuvaiheessa.

Yleinen valinta naapuruusfunktioksi on Gaussin naapuruusfunktio [10]

$$h_{ji}(t) = \alpha(t) e^{-\frac{\left\| s_j - s_i \right\|^2}{2\sigma^2(t)}}. \tag{4.3}$$

Kaavassa (4.3) s_j ja s_i ovat mallivektoreihin m_i ja m_j liittyvät kartan solmut. Funktio $\alpha(t)$ on kartan oppimisnopeus (Learning rate). Oppimisnopeus on monotonisesti laskeva skalaarifunktio. Myös oppimisnopeusfunktion valintaan on useita vaihtoehtoja, funktio voi olla esimerkiksi lineaarinen, kuten $\alpha(t) = \frac{1}{t}$ tai eksponentiaalinen, $a(t) = \frac{b^t}{T}$, missä b on vakio ja $0 < b < 1$. Vakio T on kartan opetuskierrosten kokonaislukumäärä. Varianssi $\sigma^2(t)$ on toinen laskeva funktio, joka määrää Gaussin funktion kuvaajan leveyden. Itseorganisoituvan kartan tapauksessa leveys tarkoittaa sitä, kuinka suurella säteellä naapuruusfunktio vaikuttaa. Toinen yleinen, Gaussin naapuruusfunktiota yksinkertaisempi vaihtoehto on niin kutsuttu kuplafunktio [12]. Kuplafunktio määritellään kaavalla

$$h_{ji} = \begin{cases} \alpha(t) & j \in N_i \\ 0 & j \notin N_i \end{cases} \tag{4.4}$$

missä N_i on niiden solmujen indeksien joukko, jotka ovat kartalla tietyn säteen sisällä mallivektoriin m_i liittyvästä solmusta. Kuplafunktiota käytettäessä ei päästä niin hyviin lopputuloksiin kuin Gaussin naapuruusfunktiolla [12].

4.2 Sarja-algoritmi

Käytännön sovelluksissa on suositeltavaa käyttää sarjalaskentaista itseorganisoituvan kartan versiota (Batch map), sillä se on nopeampi ja suppenee varmemmin kuin

edellä mainittu askelittainen versio [9]. Sarjalaskentaisessa versiossa kaikki syötevektorit syötetään kartalle samanaikaisesti, mikä nopeuttaa algoritmin suoritusta verrattuna askelittaiseen versioon, jossa karttaa päivitetään jokaisen yksittäisen syötevektorin syöttämisen jälkeen. Tutkitaan seuraavaksi askelittaisen algoritmin stabiilia lopputilaa, josta johdetaan sarjalaskentaisessa algoritmissa mallivektoreiden päivittämiseen käytettävä kaava. Stabiilissa tilassa kunkin mallivektorin perättäisten tilojen odotusarvojen täytyy olla yhtä suuria. Itseisarvon lineaarisuuden nojalla saadaan yhtälö

$$\begin{aligned} &\mathrm{E}_t\{m(t+1)\} = \mathrm{E}_t\{m(t)\} \\ &\Rightarrow \mathrm{E}_t\{m(t+1) - m(t)\} = 0, \end{aligned}$$

kun $t \to \infty$. Sijoittamalla vektorin $m_i(t+1)$ paikalle arvo kaavasta (4.1) ja kirjoittamalla odotusarvo auki saadaan yhtälö

$$\begin{aligned} &\mathrm{E}_t\{m_i(t) + h_{ci}(t)[x(t) - m_i(t)] - m(t)]\} = 0 \\ &\Rightarrow \mathrm{E}_t\{h_{ci}(t)[x(t) - m_i(t)]\} = 0 \\ &\Rightarrow \sum_{t=1}^{T} \frac{1}{t}\{h_{ci}(t)[x(t) - m_i(t)]\} = 0, \end{aligned} \tag{4.5}$$

jossa c on voittajasolmun indeksi syötevektorille $x(t)$, eli sen solmun indeksi, jonka mallivektori on lähimpänä vektoria $x(t)$. Yllä kuvatussa stabiilissa lopputilassa mallivektorin $m_i(t)$ arvo ei enää riipu parametrista t, joten merkitään niin sanottua lopullista mallivektoria m_i^*. Näin mallivektorin lopulliselle arvolle saadaan kaava

$$
\begin{aligned}
&\sum_{t=1}^{T} \frac{1}{t}\{h_{ci}(t)[x(t) - m_i^*]\} = 0 \\
&\Rightarrow \sum_{t=1}^{T}\{h_{ci}(t)[x(t) - m_i^*]\} = 0 \\
&\Rightarrow \sum_{t=1}^{T} h_{ci}(t)m_i^* = \sum_{t=1}^{T} h_{ci}(t)x(t) \\
&\Rightarrow m_i^* \sum_{t=1}^{T} h_{ci}(t) = \sum_{t=1}^{T} h_{ci}(t)x(t) \\
&\Rightarrow m_i^* = \frac{\sum_{t=1}^{T} h_{ci}(t)x(t)}{\sum_{t=1}^{T} h_{ci}(t)}.
\end{aligned}
\tag{4.6}
$$

Mallivektorien päivittäminen sarjalaskentaisessa algoritmissa perustuu yllä johdettuun lopullisen mallivektorin kaavaan. Tarkastellaan seuraavaksi algoritmin käytännön toteutusta, jonka jälkeen sarjalaskentaisessa versiossa mallivektorien päivittämiseen käytetty kaava muokataan lopulliseen muotoonsa.

Sarjalaskentaisessa algoritmissa kuhunkin kartan solmuun liittyy lista vektoreita, joka tyhjennetään jokaisen algoritmin kierroksen alussa. Kierroksen aikana jokaiselle syötevektorille $x(t)$ etsitään se mallivektori m_i, joka on lähimpänä kyseistä syötevektoria. Vektori $x(t)$ lisätään mallivektoria m_i vastaavan solmun listaan. Kun kaikki syötevektorit on käyty läpi ja jaoteltu solmujen listoihin, lasketaan jokaisen listan sisällä kyseisen listan vektoreiden keskiarvo $\bar{x}$. Merkitään solmun s_j listan keskiarvoa vektorilla $\bar{x}_j$. Muuttamalla kaavaa (4.6) siten, että oikeanpuolen summissa käydään läpi solmuja vastaavat listat yksittäisten syötevektoreiden sijaan, saadaan kaava muotoon

$$
m_i^* = \frac{\sum_{j=1}^{M} h_{ji} n_j \bar{x}_j}{\sum_{j=1}^{M} h_{ji} n_j}, \tag{4.7}
$$

jossa n_j on solmun s_j listassa olevien syötevektoreiden lukumäärä. Kierroksen lopuksi siis jokaisen mallivektorin arvo päivitetään kaavan (4.7) mukaisesti. Yleensä halutun lopputuloksen saavuttamiseksi riittää muutamasta muutamaan kymmeneen kierrosta [8, 9]. Käyttämällä naapuruusfunktiona yllä kuvattua kuplafunktiota, ja asettamalla algoritmin oppimisnopeudeksi vakiofunktio $\alpha(t) = 1$, saadaan mallivektorien päivitys yksinkertaistettua muotoon

$$m_i^* = \frac{\sum_{j \in N_i} n_j \bar{x}_j}{\sum_{j \in N_j} n_j}, \tag{4.8}$$

jossa N_i niiden solmujen indeksien joukko, jotka ovat kuplafunktiossa määritetyn säteen sisällä solmusta s_i. Tällöin siis mallivektorin m_i päivitetyksi arvoksi asetetaan kaikkien solmun s_i naapuruston solmujen listoissa olevien syötevektoreiden keskiarvo.

5. WLAN-SIGNAALIEN VOIMAKKUUKSISTA MUODOSTETTU ITSEORGANISOITUVA KARTTA

Seuraavaksi tarkastellaan itseorganisoituvan kartan ominaisuuksia esimerkin avulla. Syötedatana käytetään tuloksia langattomien verkkojen signaalien voimakkuuksien mittauksista [5, 6]. Syötevektorit ovat mittauspisteitä, joiden alkiot ovat kyseisessä pisteessä mitattuja eri tukiasemien signaalin voimakkuuksia MAC-osoitteiden mukaan lajiteltuina. Jokainen syötedatamatriisin rivi vastaa yhtä mittaustulosta, ja sarake yhtä MAC-osoitetta. MAC-osoitteita voi olla useampi kuin yksi yhtä tukiasemaa kohden. Mittauspisteiden eli syötedatan vektoreiden lukumäärä $k = 160$. Eri MAC-osoitteita on 171 kappaletta, eli syötedatamatriisin $X = [x(0), \ldots, x(159)]$ vektorit ovat pystyvektoreita, joiden ulottuvuus $n = 171$. Datan luonteesta johtuen suurin osa, noin 80%, matriisin arvoista on tyhjiä, eli kyseiseltä mittauspisteeltä ei ole havaittu signaalia tietystä MAC-osoitteesta. Signaalin voimakkuudet on ilmaistu desibelimilliwatteina, joka on logaritminen asteikko. Pienin epätyhjä matriisissa oleva arvo on -95, ja tyhjät arvot on korvattu arvolla -96. Mitta-asteikon logaritmisuudesta johtuen tämä vastaa kymmenen kertaa pienintä mitattua arvoa pienempää signaalin voimakkuutta. Jokaisesta mittauspisteestä on tallennettu pisteen maantieteelliset leveys- ja pituusasteet, joita käytetään lopuksi itseorganisoituvan kartan analysoinnissa. Tavoitteena on muodostaa itseorganisoituva kartta syötedatasta, ja tutkia muodostuvia klustereita suhteessa mittauspisteiden fyysiseen sijaintiin.

Syötedata on kerätty Tampereen Teknillisen Yliopiston kampukselta, Sähkötalorakennuksesta. Mittaustuloksia on kerätty kolmesta eri kerroksesta. Selkeyden vuoksi itseorganisoituva kartta rakennetaan vain kolmannen kerroksen mittaustuloksista.

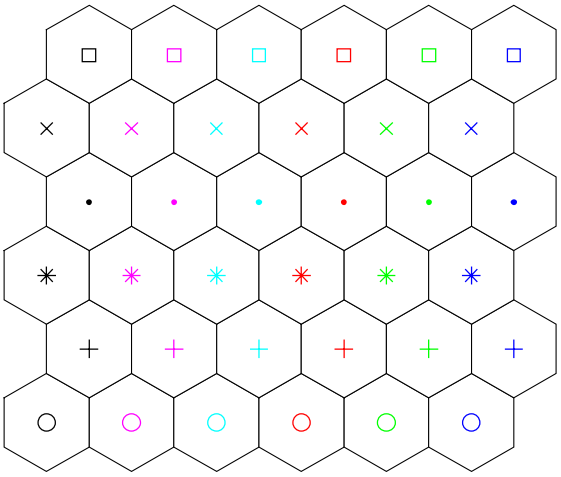
Muodostetaan aluksi kartta, jonka koko on 36 solmua. Solmut on järjestetty kuusikulmaiseen ruudukkoon kuvan 5.1 osoittamalla tavalla. Naapuruusfunktiona käytetään euklidista naapuruusfunktiota ja opetukseen luvussa 4.2 esitettyä sarja-laskentaista algoritmia.

Valmis kartta on esitetty kuvassa 5.2. Kuten esimerkiksi kuvan ylälaidasta huomataan, itseorganisoituvan kartan signaalin voimakkuuksien perusteella luomat klusterit eivät täysin muodostu sijainnin perusteella, vaan eri klustereihin kuuluvat al-

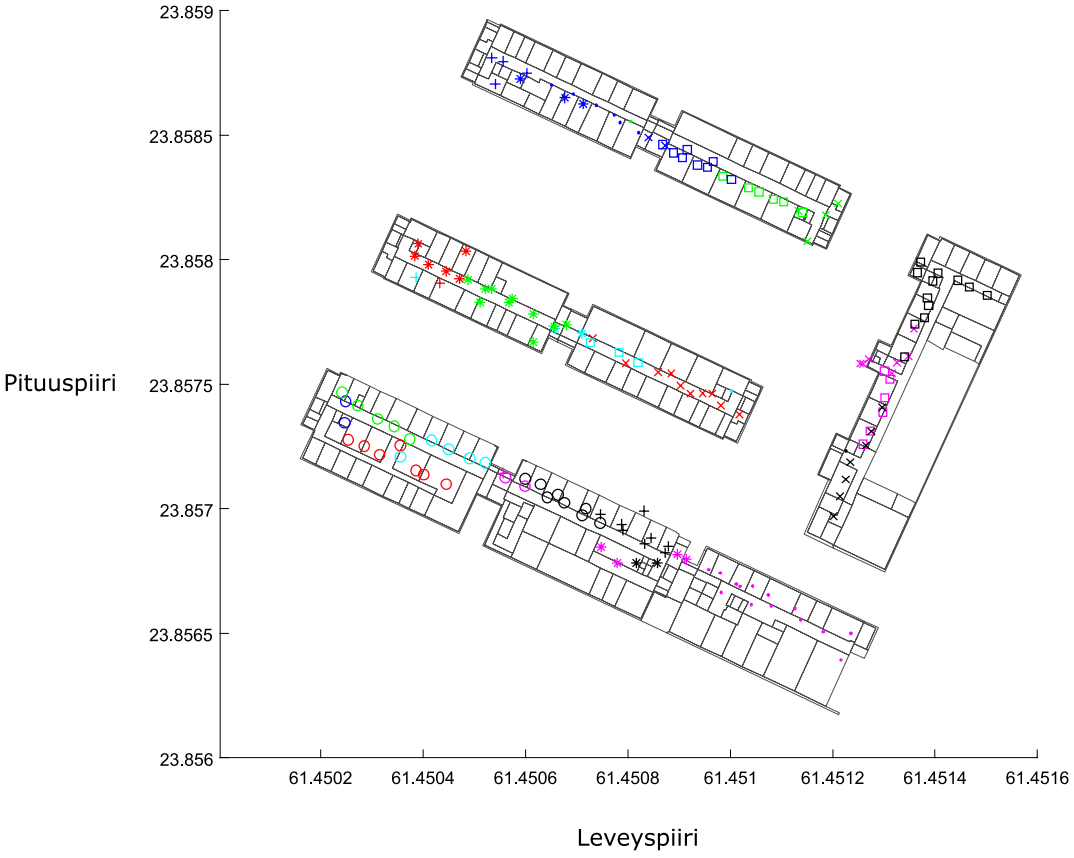
kiot sekoittuvat keskenään etenkin klustereiden laidoilla. Tämä johtuu siitä, että signaalien voimakkuuksiin mittauspisteessä vaikuttaa tukiaseman etäisyyden lisäksi etenkin seinät ja muut rakenteet, jotka vaimentavat signaaleja.

Itseorganisoituvan kartan luomaa klustereiden välistä järjestystä voidaan tarkastella vertailemalla kuvassa 5.1 esitettyä kartan rakennetta ja kuvan 5.2 valmista karttaa. Erityisesti ulkoseinien rajoittamien erillisten siipien sisällä vierekkäisten solmujen klusterit sijaitsevat myös fyysisesti lähekkäin. Paksut ulkoseinät rajoittavat signaaleja merkittävästä, ja esimerkiksi kuvan 5.2 oikeassa yläkulmassa lähekkäin, mutta eri siivissä sijaitsevat vihreällä rastilla ja mustalla neliöllä merkityt klusterit ovat itseorganisoituvan kartan rakenteessa eri laidoilla. Ulkoseinien vaikutus näkyy myös siinä, ettei yksikään klusteri sisällä mittauspisteitä useammasta eri siivestä.

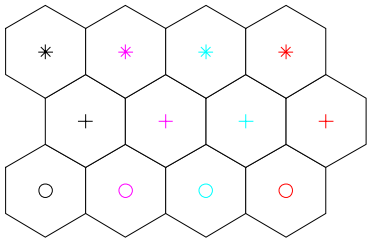
Seuraavaksi muodostetaan kartta samasta syötedatasta, mutta pienennetään kartan kokoa kolmasosaan, 12 solmuun. Kartan rakenne on esitetty kuvassa 5.3 ja valmis kartta on kuvassa 5.4. Pienentämällä solmujen lukumäärää pienenee myös kartan resoluutio, eli osa pienemmistä klustereista sulautuu yhteen muodostaen suuria klustereita. Vertailemalla kuvia 5.2 ja 5.4 huomataan kuitenkin, että klustereiden rajat ovat selkeämmät 12 solmun kuin 36 solmun kartassa. Erityisesti rakennusten eri osat erottuvat 12 solmun kartassa selkeämmin toisistaan, mikä voi olla haluttu lopputulos esimerkiksi tietyissä sisätilapaikannuksen sovelluksissa. Luotua karttaa voitaisiin käyttää likimääräisen sijainnin määrittämiseen. Optimaalinen itseorganisoituvan kartan koko riippuu siis tässäkin tapauksessa kartan käyttötarkoituksesta.



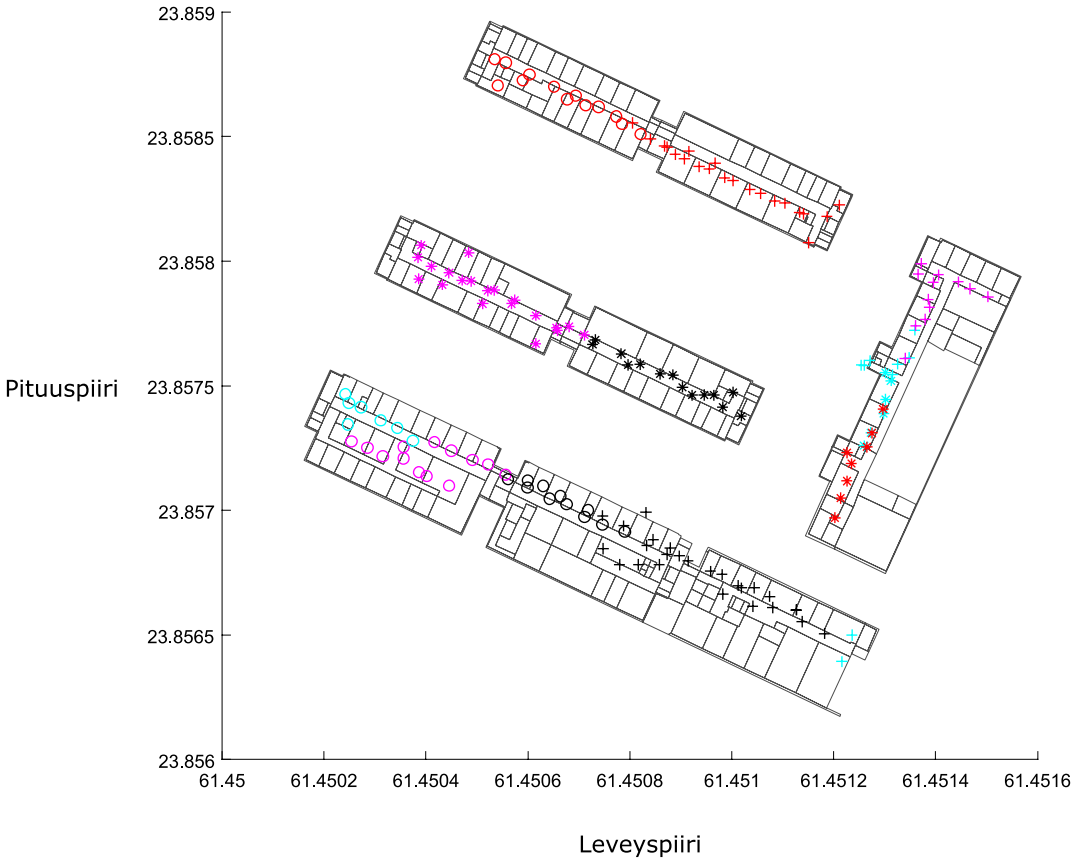
Kuva 5.1 *36 solmun itseorganisoituvan kartan rakenne. Kuusikulmion sisässä oleva kuvio on vastaavan solmun klusteriin kuuluvien mittauspisteiden merkintä, jota käytetään kuvassa 5.2.*



Kuva 5.2 *Itseorganisoituvan kartan avulla klusteroidut mittauspisteet, kartan koko 36 solmua. Mittauspisteet on piirretty koordinaattiensa mukaan rakennuksen pohjapiirustuksen päälle. Samalla värillä ja merkillä piirretyt pisteet kuuluvat samaan klusteriin. Klustereiden ulkoasu määräytyy itseorganisoituvan kartan tekemän luokittelun perusteella kuvan 5.1 mukaisesti.*



Kuva 5.3 *12 solmun itseorganisoituvan kartan rakenne. Kuusikulmion sisässä oleva kuvio on vastaavan solmun klusteriin kuuluvien mittauspisteiden merkintä, jota käytetään kuvassa 5.4.*



Kuva 5.4 *Itseorganisoituvan kartan avulla klusteroidut mittauspisteet, kartan koko 12 solmua. Mittauspisteet on piirretty koordinaattiensa mukaan rakennuksen pohjapiirustuksen päälle. Samalla värillä ja merkillä piirretyt pisteet kuuluvat samaan klusteriin. Klustereiden ulkoasu määräytyy kuvan 5.3 mukaisesti.*

6. YHTEENVETO

Itseorganisoituva kartta on yleisesti useilla eri tieteen- ja teollisuudenaloilla käytössä oleva algoritmi suurten tietomäärien analysointiin. Sen avulla voidaan visualisoida datan sisäisiä rakenteita, sekä luokitella data-alkioita samankaltaisista alkioista muodostuviin klustereihin. Itseorganisoituva kartta järjestää lisäksi myös klusterit keskenään siten, että samankaltaiset klusterit sijaitsevat kartalla lähekkäin, mikä tekee siitä erityisen käyttökelpoisen klusterointialgoritmin.

Itseorganisoituvan kartan soveltamisessa oleellisinta on löytää käyttötarkoitukseen sopivat alkutilan parametrit. Erityisesti sopivan koon, eli kartan solmujen lukumäärän, löytäminen voi vaatia useita yrityksiä. Myös kartan muodolla ja naapuruusfunktion valinnalla voi olla merkitystä lopputuloksen kannalta. Mallivektorien alustuksella voidaan lopullisen kartan toimivuuden lisäksi vaikuttaa merkittävästi myös algoritmin suoritusnopeuteen.

Itseorganisoituvaa karttaa voidaan käyttää WLAN-signaalien voimakkuusmittausten tulosten analysoinnissa. Mittauspisteet voidaan luokitella pelkkien signaalin voimakkuuksien perusteella siten, että fyysisesti lähekkäin sijaitsevat pisteet kuuluvat samoihin klustereihin. Paksut rakenteet vähentävät klustereiden välisten suhteiden merkittävyyttä, mutta itseorganisoituvalla kartalla on potentiaalisia käyttökohteita sisätilapaikannuksen alalla.

LÄHTEET

[1] M. Attik, L. Bougrain, and F. Alexandre, Self-organizing Map Initialization, in Artificial Neural Networks: Biological Inspirations – ICANN 2005, W. Duch, J. Kacprzyk, E. Oja, and S. Zadrożny, eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 357–362.

[2] A. Ben-Hur and I. Guyon, Detecting Stable Clusters Using Principal Component Analysis. Totowa, NJ: Humana Press, 2003, pp. 159–182. Saatavissa: https://doi.org/10.1385/1-59259-364-X:159

[3] J. M. L. Gorricha and V. J. A. S. Lobo, On the Use of Three-Dimensional Self-Organizing Maps for Visualizing Clusters in Georeferenced Data. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 61–75. Saatavissa: https://doi.org/10.1007/978-3-642-19766-6_6

[4] R. Gray, Vector quantization, IEEE ASSP Magazine, pp. 4–29, 1984.

[5] V. Honkavirta, T. Perälä, S. Ali-Löytty, and R. Piché, A comparative survey of WLAN location fingerprinting methods, in In Proceedings of the 6th Workshop on Positioning, Navigation and Communication 2009 (WPNC'09), Mar 2009, pp. 243–251.

[6] V. Honkavirta, A comparative survey of WLAN location fingerprinting methods, Master's thesis, Tampere University of Technology, Nov 20098.

[7] T. Kohonen, The self-organizing map, Proceedings of the IEEE, pp. 1464–1480, 1990.

[8] T. Kohonen, Self-organizing maps. Springer, 2001.

[9] T. Kohonen, Essentials of the self-organizing map, Neural networks : the official journal of the International Neural Network Society, pp. 52–65, 2013.

[10] T. Kohonen, MATLAB Implementations and Applications of the Self-Organizing Map. Unigrafia Oy, 2014.

[11] U. Seiffert and B. Michaelis, Three-dimensional self-organizing maps for classification of image properties, in Proceedings 1995 Second New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems, Nov 1995, pp. 310–313.

[12] P. Stefanovic and O. Kurasova, Investigation on Learning Parameters of Self-Organizing Maps, Baltic Journal of Modern Computing, p. 45, 2014.

LIITE A: ITSEORGANISOITUVAN KARTAN LUOMISEEN KÄYTETTY MATLAB-KOODI

```
% Syötedatamatriisi, koko 171 x 160
% (160 kappaletta 171-ulotteisia vektoreita)
x = inputData;

% Kartan leveys ja korkeus (36 solmun kartalle):
dimension1 = 6;
dimension2 = 6;

% 12 solmun kartalle:
% dimension1 = 4;
% dimension1 = 3;

% Mallivektorien alustuskierrosten lukumäärä
coverSteps = 100;

% Naapuruusfunktion leveys alkutilassa (solmuina)
initNeighbor = 4;

% Kartan muoto, 'hextop'= kuuskikulmainen ruudukko
topology = 'hextop';

% Etäisyysfunktio, 'dist' = euklidinen etäisyys
distanceFunc = 'dist';

% Luodaan kartta
net = selforgmap([dimension1 dimension2], coverSteps, ...
    initNeighbor, topology, distanceFunc);

% Opetetaan kartta
[net,tr] = train(net,x);

% Tallennetaan syötevektorien luokat vektoriin 'classes',
% koko 160 x 1
classes = vec2ind(net(x))';
```