MENETELMIÄ HENKILÖN TUNNISTAMISEEN KÄVELYASKELEEN TUOTTAMASTA PAINESIGNAALISTA

JAAKKO SUUTALA

Suutala Jaakko (2004) Menetelmiä henkilön tunnistamiseen kävelyaskeleen tuottamasta painesignaalista. Oulun yliopisto, Sähkö- ja tietotekniikan osasto. Diplomityö, 90 s.

TIIVISTELMÄ

Työssä tutkittiin ja kehitettiin menetelmiä kävelijän henkilöllisyyden tunnistamiseen paineherkän lattia-anturin signaalista. Anturimateriaalina käytettiin elektromekaanista kalvoa (EMFi), joka on asennettu osaksi tutkimuslaboratorion älykästä huonetta kattaen 100 neliömetrin alan. Yksittäiset kävelyaskeleet segmentoitiin raakasignaalista ja niistä irroitettiin kuvaavia piirteitä, joiden avulla muodostettiin luokittelumallit. Luokitteluun käytettiin kahta eri menetelmää: oppivaa vektorikvantisointia (LVQ) ja diskreettejä kätkettyjä Markov-malleja (HMM). Valitut piirteet perustuivat pääasiassa askelsignaalin ääriarvopisteisiin, kuten kantapään iskun ja varpaiden nousun kohtiin, sekä niiden suhteisiin. Lisäksi askeleen taajuustasosta valittiin amplitudispektrin arvoja piirteiksi.

LVQ-luokittelu perustuu koodikirjaan, jossa jokainen luokka alustetaan äärellisellä määrällä koodikirjavektoreita. Opetuksen aikana koodikirjavektoreita siirrellään piirreavaruudessa kuvaamaan luokkarajat opetusnäytteiden avulla. Tuntemattoman näytteen luokittelu tapahtuu lähimmän koodikirjavektorin mukaan. HMM-luokittelussa yksittäinen askel esitetään ajansuhteen muuttuvien piirrevektorien havaintosekvenssinä. Opetetun LVQ-koodikirjan avulla nämä sekvenssit kuvataan diskreeteiksi symboleiksi, joiden avulla opetetaan kullekin luokalle HMM-malli. Lopuksi tuntematon havaintosenvenssi luokitellaan suurimman todennäköisyyden antaman mallin mukaan.

Standardi-LVQ-algoritmien lisäksi työssä käytettiin niiden laajennuksia lisäämään tunnitusjärjestelmien luotettavuutta ja adaptiivisuutta. Erotusherkkää LVQ:a (DSLVQ) käyttämällä voitiin havaita informatiivisimmat yksittäiset piirteet automaattisesti tunnistuksen parantamiseksi. Työn aikana kehitettiin myös 2-tasoinen tunnistusmenetelmä, joka käyttää kolmea peräkkäistä askelta ja hylkäysmahdollisuutta lopullisen luokittelupäätöksen tekemiseen.

Standardi-LVQ:lla saavutettiin 67% kokonaistunnistus 11 henkilön yksittäisten askelten osalta, kun HMM-malleilla päästiin 52% tulokseen. Kun henkilöiden määrää vähennettiin viiteen, saavutettiin molemmissa menetelmissä paljon luotettavammat tulokset niiden ollessa 91% ja 84%. DSLVQ:a käytettäessä 11 henkilön kokonaistunnistus nousi 70%, kuvaavien piirteiden valinnan johdosta. Hylkäävällä 2-tasoisella luokittelumenetelmällä saavutettiin luotettavia tuloksia myös 11 henkilölle. Oikein luokiteltujen askelien kokonaisosuus oli tässä tapauksessa 90%, kun 20% näytteistä hylättiin. Tulokset osoittavat, että kävelyaskeleen tuottamaa painesignaalia voidaan käyttää pienen ihmisjoukon tunnistamiseen osana älykästä ympäristöä.

Avainsanat: älykäs ympäristö, astujan tunnistus, paineherkkä lattia, LVQ, HMM, hahmoluokittelu

Suutala Jaakko (2004) Methods for Person Identification from Pressure Signal of Walking Steps. University of Oulu, Department of Electrical and Information Engineering. Diploma Thesis, 90 p.

ABSTRACT

In this work, methods for the identification of walkers based on measurements by a pressure-sensitive floor were studied and developed. A 100 square meter pressure-sensitive floor was covered with ElectroMechanical (EMFi) material and used as part of a smart room in a research laboratory. Single footsteps were segmented from the raw data and featurized to train the classification models. Two different classification methods were studied: Learning Vector Quantization (LVQ) and discrete Hidden Markov Models (HMM). The features used in identification were mainly based on the extreme points of the signal, such as heel strike and toe-off peak, and their interrelations. The amplitude spectrum values from the frequency level presentation were also used as features.

In LVQ, the classification is based on a codebook consisting of a finite set of codebook vectors assigned to every class. The initialized codebook is trained with featurized training footsteps to describe the class boundaries in the feature space. Finally, an unknown sample is classified to the closest codebook vector. In HMM classification, a single footstep is presented as a sequence of temporally variable feature vectors, called observations. These sequences are encoded as discrete symbols using a trained LVQ codebook. HMMs are initialized for every class and trained with sequences of observed symbols. An unknown observation sequence is classified to the most probabilistic model.

In addition, extended versions of LVQ learning algorithms were studied to increase the reliability and adaptiveness of the identification systems. A distinction-sensitive LVQ (DSLVQ), which is able to automatically detect the most informative features during the training of the codebook, was applied to footstep identification. Furthermore, a two-level identification method developed in this project was proposed. It uses three consecutive footsteps and a reject option to make a final decision on classification.

LVQ showed a 67% overall success rate in classifying single footsteps of 11 different walkers, while HMM was able to identify 52% correctly. When the number of occupants was decreased to five, the total recognition rates were much more reliable, 91% and 84% for LVQ and HMM, respectively. DSLVQ was able to select the most relevant features, which increased the recognition rate up to 70% in identification tests of 11 persons. The reject-optional two-level classifier gave very promising results with a recognition rate of 90% and a rejection rate of 20% in identifying eleven walkers. The results show that it is possible to recognize a small number of occupants from the pressure signals of walking steps as part of an intelligent environment.

Keywords: intelligent environment, gait-based identification, pressure-sensitive floor, LVQ, HMM, pattern classification

SISÄLLYSLUETTELO

ALKULAUSE

Tämä diplomityö on tehty Oulun yliopiston sähkö- ja tietotekniikan osaston tietokonetekniikan laboratorion älykkäiden järjestelmien tutkimusryhmässä. Se on toteutettu osana Beacon (Behaviour Modelling in Context Aware Systems) - projektia, joka kuluu Suomen Akatemian proaktiivisen tietotekniikan tutkimusohjelmaan.

Haluan erityisesti kiittää ohjauksesta ja arvokkaista neuvoista työn valvojaa professori Juha Röningiä sekä toista tarkastajaa professori Tapio Seppästä. Lisäksi kiitokset menevät tutkija Susanna Pirttikankaalle perehdyttämisestä aiheeseen ja ohjauksesta työn edetessä, Kalle Koholle yhteistyöstä signaalin esikäsittelyn parissa sekä myös muille tutkimusryhmän henkilöille mukavasta ja inspiroivasta työympäristöstä.

Oulussa 23.05.2004

Jaakko Suutala

LYHENTEIDEN JA MERKKIEN SELITYKSET

$D_c(.)$	oikein luokiteltujen näytteiden esiintymistiheysfunktio
$D_e(.)$	väärin luokiteltujen näytteiden esiintymistiheysfunktio
$norm(.)$	DSLVQ-algoritmin normalisointifunktio
$P(.)$	todennäköisyys
$P(.\|.)$	ehdollinen todennäköisyys
P	luokittelijan tehokkuusfunktio
P_{id}	luokittelijan ideaalinen tehokkuusfunktio
$sgn(.)$	sigmoid-funktio
$sgn'(.)$	sigmoid-funktion derivaatta
$\mathbf{A}$, a_{ij}	HMM-mallin tilansiirtotodennäköisyysmatriisi sekä yksittäinen todennäköisyys siirryttäessä tilasta i tilaan j
$\mathbf{B}$, $b_j(k)$	HMM-mallin havaintojen todennäköisyysjakauma sekä yksittäinen todennäköisyys tilassa j
c	LVQ:n lähimmän koodikirjavektorin indeksi
C_c	oikein luokiteltavien näytteiden kustannusarvo
C_e	väärin luokiteltavien näytteiden kustannusarvo
C_N	normalisoitu kustannusarvo
C_r	hylättävien näytteiden kustannusarvo
d_e	euklidinen etäisyys
d_w	painotettu euklidinen etäisyys
$h[n]$	konvoluutiomaski
$\mathbf{m}$	LVQ:n koodikirjavektori
M	HMM-mallin symbolien lukumäärä
N	HMM-mallin tilojen lukumäärä
$\mathbf{O}$	HMM-mallin havaintosekvenssi
$\underline{\mathbf{O}}$	LVQ-luokittelijan ulostulovektori
O_{2WIN}	LVQ:n toiseksi lähimmän koodikirjavektorin etäisyys
O_{max}	LVQ:n opetusaineiston suurin O_{WIN} etäisyys
O_{WIN}	LVQ:n lähimmän koodikirjavektorin etäisyys
q_t	HMM-mallin tila ajanhetkellä t
$\mathbf{Q}$	HMM-mallin paras yksittäinen tilasekvenssi
R_c	oikein luokiteltujen näytteiden suhteellinen osuus
R_c^0	oikein luokiteltujen näytteiden suhteellinen osuus ilman hylkäystä
R_e	väärin luokiteltujen näytteiden suhteellinen osuus
R_e^0	väärin luokiteltujen näytteiden suhteellinen osuus ilman hylkäystä
R_r	hylättyjen näytteiden suhteellinen osuus
$R_r^{(c)}$	hylättyjen oikein luokiteltujen näytteiden suhteellinen osuus
$R_r^{(e)}$	hylättyjen väärin luokiteltujen näytteiden suhteellinen osuus
$s(t)$	OLVQ1:n 2-arvoinen luokittelukerroin
S_i	HMM-mallin tila i
v_k	HMM-mallin k:s tilahavainto
w	LVQ:n ikkunan leveys
$\mathbf{w}$	piirteiden painokerroinvektori
$\mathbf{x}$	näytevektori/piirrevektori
$\overline{x}$	piirteen näytekeskiarvo
$\hat{x}$	normalisoitu piirre

$x[t]$	sisääntuleva signaali
$X[t]$	signaalin Fourier-muunnos
$y[t]$	konvoluutiovaste
$\alpha(t)$	LVQ:n opetuskerroin
$\alpha(t)_c$	OLVQ1:n opetuskerroin yksittäiselle koodikirjavektorille
$\alpha_t(i)$	eteenpäin-muuttuja
$\beta_t(i)$	taaksepäin-muuttuja
$\gamma_t(i)$	todennäköisyys sille, että ollaan tilassa S_i ajanhetkellä t, annettujen havaintojen ja mallin mukaan
$\delta_t(i)$	yksittäisen reitin suurin todennäköisyys ajanhetkellä t
$\epsilon(t)$	LVQ3:n korjauskerroin
λ	HMM-malli
$\mu(x)$	GLVQ:n korjauskerroin
σ	hylkäyskynnys
$\underline{\sigma}$	optimaalinen hylkäyskynnys
σ_a	hylkäyskynnys päätösalueista kaukana oleville näytteille
σ_b	hylkäyskynnys luokkien päällekkäisillä päätösalueilla oleville näytteille
σ_{id}	ideaalinen hylkäyskynnys
σ_k	keskihajonta piirteelle k
σ_k^2	varianssi piirteelle k
π, π_i	HMM-mallin alkutilan todennäköisyysmatriisi ja tilan i todennäköisyys
$\xi(i,j)$	todennäköisyys sille, että ollaan tilassa S_i ajanhetkellä t ja tilassa S_j ajanhetkellä $t+1$
$\psi_t(j)$	HMM-mallin parhaan tilansiirtymän tallentamiseen käytettävän taulukon alkio ajanhetkellä t
Ψ	luotettavuusevaluaattori
Ψ_a	luotettavuusevaluaattori näytteiden hylkäämiseen liian kaukana päätösalueista
Ψ_b	luotettavuusevaluaattori näytteiden hylkäämiseen luokkien päällekkäisiltä päätösalueilta
2D	2-Dimensional, 2-ulotteinen
3D	3-Dimensional, 3-ulotteinen
CV	Cross-Validation, ristiin validointi
DSLVQ	Distinction-Sensitive Learning Vector Quantization, erotusherkkä oppiva vektorikvantisointi
EM	Expectation Maximation, estimointimenetelmä
EMFi	ElectroMechanical Film, elektromekaaninen kalvo
FFT	Fast Fourier Transformation, nopea Fourier-muunnos
GLVQ	Generalized Learning Vector Quantization, yleistetty oppiva vektorikvantisointi
GRF	Ground Reaction Force, askeleen sensoriin kohdistama voima
GRLVQ	Generalized Relevance Learning Vector Quantization, yleistetty relevanssi oppiva vektorikvantisointi
HMM	Hidden Markov Models, kätketyt Markov-mallit
KNN	K-Nearest Neighbours, k-lähimmän naapurin luokittelumenetelmä

L-R	Left-to-Right, vasemmalta-oikealle-tyyppinen HMM-malli
LVQ	Learning Vector Quantization, oppiva vektorikvantisointi
ML	Maximum Likelihood, suurin uskottavuus
MLP	Multi-Layer Perceptron, monikerrosneuroverkko
NN	Nearest Neighbours, lähimmän naapurin luokittelumenetelmä
RFID	Radio Frequency Identification, radiotaajuuksilla toimiva tunnistin
RLVQ	Relevance Vector Quantization, relevanssi oppiva vektorikvantisointi
SOM	Self-Organizing Maps, itseorganisoituvat kartat
SSMM	Segmental Semi-Markov Models, segmentiaaliset semi-Markovin mallit
VQ	Vector Quantization, vektorikvantisointi

1. JOHDANTO

Tietokoneiden laskentatehon kasvun, laitekoon pienentymisen ja sensorien kehityksen myötä uusien tekniikoiden käyttöön on avautunut entistä enemmän mahdollisuuksia viime vuosina. Prosessorit ja sensorit ovat tulossa lähemmäksi arkielämäämme helpottamaan päivittäisiä rutiineitamme erilaisten älykkäiden laitteiden ja ympäristöjen muodossa. Tästä johtuen kiinnostus erilaisten älykäiden ympäristöjen tutkimukseen ja suunnitteluun on vilkastunut. Tutkimuksessa on meneillään useita suuria projekteja, joissa uusimpia tekniikoita kehitetään älykotiympäristöissä [1, 2, 3] sulauttamalla käytettävä tekniikka kaikkialla läsnäolevaksi (ubiquitous) ja läpitunkevaksi (pervasive) [4]. Tällöin tekniikka voidaan tuoda lähemmäksi ihmistä mahdollistaen paremman vuorovaikutuksen ihmisen ja koneen välillä kuin perinteisemmässä laitekeskeisessä tietotekniikassa on aikaisemmin ollut mahdollista. Jo vuonna 1991 Mark Weiser [5] visioi kaikkialla läsnäolevasta laskennasta, jolla ihmisen arkirutiineita voitaisiin helpottaa erilaisten ympäristöön sulautettavien laitteiden ja sensorien avulla häiritsemättä kuitenkaan ihmisen omaa toimintaa.

Tärkeä osa älykästä ympäristöä on sensoritietoon perustuva ihmisen toiminnan mallintaminen, jotta läpinäkyvä ja sulautettu tekniikka voisi automaattisesti hyödyttää sen käyttäjää [6, 7]. Ihmisen mallintaminen on perustunut pääasisassa kameroiden, mikrofonien sekä erilaisten kannettavien laitteiden ja tunnistismien tuottaman sensoritiedon käyttöön. Esimerkiksi konenäkömenetelmiin perustuvilla kasvontunnistusmenetelmillä saadaan luotettavasti määritettyä käyttäjän henkilöllisyys ja ilmeiden perusteella jopa hänen mielialansa. Liikkeenmuutoksia seuraava kamera taas pystyy paikantamaan ihmisen älykkäässä huoneessa, jossa käyttäjä voi antaa seiniin kiinnitettyjen mikrofoninen avulla komentoja ympäristölle [7].

Vaikka edellä mainitut menetelmät voivat tarjota luotettavaa tietoa ihmisen toiminnasta, sisältyy niihin myös ongelmia. Kameroihin perustuva ihmisen mallintaminen vaatii oikean valaistuksen, mikrofoneihin pohjautuva puheentunnistus ei voi tapahtua hiljaisesti, kärsii usein taustamelusta ja useasta lähteestä saapuvien ääninäytteiden erottaminen voi olla vaikeaa. Kannettavat sensorit sitovat käyttäjän aina tähän laitteeseen, jolloin menetetään tiettyä vapautta.

Yksi ratkaisu läpinäkyvään käyttäjän paikantamiseen ja tunnistamiseen tarjoavat ihmisen kävelyaskeleen signaalia mittaavat lattia-anturit. Tällaiset sensorit eivät tarvitse valaistusta, eivät ole herkkiä taustamelulle, eivätkä vaadi käyttäjältä mukana kannettavia laitteita, sillä itse sensori on sulautettu näkymättömästi ympäristöön ja tarjoaa luonnollisella tavalla tapahtuvan käyttäjän paikantamisen ja tunnistamisen. Lattia-antureiden käyttöä on tutkittu muutamissa älykkäisiin ympäristöihin liittyvissä projekteissa, joissa anturit ovat olleet niihin kohdituvaa voimaa mittaavia varaussoluja [8], [9] sekä yksinkertaisia ON/OFF-tyyppisiä liuska-antureita, jotka ilmoittavat niihin kohdistuvan osuman [10]. Näissä järjestelmissä henkilön tunnistamismenetelminä on käytetty kätkettyjä Markovin malleja (HMM), lähimmän naapurin luokitinta (NN) sekä monikerrosneuroverkkoja (MLP).

Tässä työssä henkilön tunnistamiseen käytettiin paineenmuutosta ilmaisevaa EMFi-sensoria, joka kattaa tutkimuslaboratoriomme koko 100 neliömetrin lattia-alan. Tarkoituksena oli tutkia ja kehittää menetelmiä henkilön luotettavaan tunnistamiseen uutta sensorimateriaalia käyttäen. Tunnistusmenetelminä käytettiin oppivaa vektorikvantisointia (LVQ) [11, 12, 13]. Lisäksi työssä kokeiltiin

myös HMM-malleja [14]. EMFi-signaalin esikäsittelymenetelmiä on myös kehitetty soveltamalla askelsignaaliin mallipohjaista ja tilastollista segmentointimenetelmää. Sen avulla raakasignaalista voidaan segmentoida yksittäisiä askeleita paikantamiseen ja tunnistamiseen [15]. Esikäsittelyä ja paikannusta tutkitaan kuitenkin toisessa työssä ja sen käsittely on tämän työn osalta vähäistä.

Kehitettyjen menetelmien pohjalta on tarkoitus rakentaa reaaliaikainen järjestelmä tutkimuslaboratorioon, jossa ihmisen tunnistamista voidaan käyttää osana älykästä ympäristöä. Tarkoituksena on ihmisen mallintamisen lisäksi käyttää anturilattiaa mobiilin robotin liikkeiden havaitsemiseen ja yhteistyöhön ihmisen kanssa. Työssä kehitetyt tunnistusmenetelmät ovat offline-tyyppisiä, joissa tunnistusmalli ei muutu sen opetuksen jälkeen. Tulevaisuudessa menetelmiä voidaan kehittää vielä pitemmälle, jolloin ne pystyisivät toimimaan adaptiivisesti ja muuttumaan tilanteen mukaisesti, esimerkiksi havaitsemaan systeemille entuudestaan tuntemattomat henkilöt ja muokkautumaan jo tunnistettujen henkilöiden käyttäytymisen muutoksiin.

2. ÄLYKÄS YMPÄRISTÖ JA HENKILÖN TUNNISTAMINEN

Tässä luvussa käsitellään älykkään ympäristön perusominaisuuksia ja vaatimuksia, ja tarkastellaan millaisia sensorijärjestelmiä ja menetelmiä tällaisissa ympäristöissä käytetään ihmisen ja ympäristön tilan mallintamiseen. Lisäksi käsitellään tarkemmin henkilön tunnistamismenetelmiä ja niiden soveltuvuutta älykkäiden ympäristöjen vaatimuksiin. Tässä työssä toteutetut tunnistusmenetelmät liittyvät biometrisiin tunnistusmenetelmiin, ja tarkemmin ihmisen kävelytyylin mallintamiseen, joten luku sisältää myös katsauksen biometrisiin tunnistusmenetelmiin ja tämän työn kanssa läheisesti samankaltaiseen muualla tehtyyn tutkimukseen. Luvun loppupuolella esitetään myös askeltunnistuksen vaiheet, joihin seuraavien lukujen käsittely pääasiassa perustuu.

2.1. Älykäs ympäristö

Älykkään ympäristön yleispiirteenä voidaan pitää sen kykyä tiedostaa ja vuorovaikuttaa. Älykäs ympäristö reagoi ihmisen käskyihin, tiedottaa ihmisen läsnäolon, ymmärtää ihmisen suhteen ympäristöön ja mukautuu vastaamaan ihmisen tarpeita, tottumuksia ja tuntemuksia. Älykkään ympäristön voidaan ajatella koostuvan neljästä osasta: läsnäolosta, tietoisuudesta, älykkyydestä ja vuorovaikutuksesta. Läsnäololla tarkoitetaan ympäristöä, jossa ihminen on ympäröity monilla erilaisilla kommunikoivilla laitteilla, jotka ovat usein myös piilotettuja käyttäjältä. Tietoisuus merkitsee järjestelmän itsensä kykyä hankkia tietoa automaattisesti. Älykkyys on ympäristön kyky analysoida ja hyödyntää hankkimaansa tietoa mukautumalla ihmisen tarpeisiin sekä oppia erilaisia toimintoja. Vuorovaikutuksella tarkoitetaan ihmisen ja esimerkiksi tietokoneen välistä vuorovaikutusta erilaisten käyttöliittymien avulla. [16]

Älykkään ympäristön tulee siis olla sulautettu ja monimuotoinen, jossa ihmisellä on mahdollisuus vuorovaikuttaa ympäristön kanssa luonnollisilla tavoilla. Tämä vaatii älykkäältä ympäristöltä monenlaisia sensoreita. Ympäristö käyttää kameroita silminään, mikrofoneja korvinaan ja suuren määrään erilaisia kehittyneitä anturijärjestelmiä, joilla reaalimaailman havainnot saadaan yhdistettyä käytettyyn teknologiaan. Esimerkiksi konenäköön ja puheentunnistukseen perustuvat ihmisen mallinnusmenetelmät auttavat saavuttamaan tämän luonnollisella tavalla tapahtuvan kommunikaation ihmisen ja ympäristön välillä. [17]

Luonnollisella tavalla tapahtuvaa kommunikaatiota voidaan kuvata läsnäolevaksi aistinnaksi (ubiquitous sensing), jossa erilaisten sensorien tietoa voidaan käyttää ihmisen paikantamiseen, henkilön tunnistamiseen, aktiviteetin tunnistamiseen, laitteiden ohjaamiseen ja erilaisten rutiinien oppimiseen. Tällaisten sensorijärjestelmien ja menetelmien kehityksessä on otettava huomioon monia tärkeitä vaatimuksia, jotta älykäs ympäristö vastaisi sille asetettuja vaatimuksia. Sensoreilla pitää olla kyky kalibroitua itsestään adaptiivisen ympäristön tarpeiden mukaan, ja niiden pitää pystyä kommunikoimaan toisten sensorien kanssa, jotta ympäristön tilamalli saadaan rakennettua. [6]

Lisäksi sensoritiedon sovellusmenetelmien pitää olla riittävän joustavia ja adaptiivisia sekä laskennallisesti toteutettavissa reaaliajassa, jotta niitä olisi järkevä soveltaa tositilanteisiin.

2.2. Sensorit ja ihmisen mallintaminen

Erilaisten sensorien käyttöä voidaan pitää älykkään ympäristön perustana. Niiden avulla voidaan aistia ja mallintaa ihmisen toimintaa, ympäristön tila ja toteuttaa käyttäjän vaatimat komennot [18]. Käytettävät sensorit voidaan jakaa kahteen ryhmään. Ne ovat joko staattisia, jolloin ne kiinnitetään ympäristöön, eikä käyttäjän tarvitse välttämättä olla edes tietoinen niiden läsnäolosta. Toisena vaihtoehtona ovat sensorijärjestelmät, joissa käyttäjät kantavat mukanaan jotain kannettavaa tai puettavaa sensoria. Tällöin kannettava laite voi tuottaa informaatiota suoraan sen käyttäjälle tai kommunikoida ympäristön kanssa, esimerkiksi langattomien tiedonsiirtotekniikoiden avulla.

Älykkäässä ympäristössä sensorien tulisi olla riittävän joustavia, ja niiden käyttö pitäisi olla luonnollista. Tämä tarkoittaa, että ne on sulautettu läpinäkyvästi ympäristöön eivätkä häiritse käyttäjää. Tällöin ensimmäisellä sensorivaihtoehdolla saavutetaan parhaiten älykkään ympäristön vaatimukset, koska ne eivät vaadi käyttäjältä mitään mukana olevaa laitteistoa. Kuitenkin toisen ryhmän sensorit voivat antaa tarvittavaa lisätietoa, pääasiassa käyttäjän näkökulmasta.

Sensoritietoon perustuvassa ihmisen mallintamisessa puhutaan usein aistittavasta älykkyydestä (perceptual intelligence). Ihmisen mallintamisesta voidaan esittää viisi kysymystä: kuka, mitä, missä, milloin ja miksi? Jos kone kykenee aistimaan nämä seikat, se tulee tietoiseksi ympäristöstään [19], ja luonnollinen yhteistyö koneen, ympäristön ja ihmisen välillä on mahdollista. Seuraavassa on esitelty sensoritekniikoita, joita on käytetty älykkäissä ympäristöissä sekä käsitellään niihin liittyviä menetelmiä ihmisen mallintamiseksi ja aistinnallisen älykkyyden saavuttamiseksi.

2.2.1. Visuaaliset sensorit

Näköaisti ohjaa suurta osaa ihmisen toiminnasta. Niinpä on luonnollista, että älykäs ympäristö voi tehdä havaintoja erilaisten kamerapohjaisten järjestelmien avulla. Yleensä kamerat ovat kiinnitettyjä ympäristöön, jolloin ne tarjoavat tietoa ympäristön perspektiivistä, eli seuraavat käyttäjän toimintoja. Tämän lisäksi kamerat voivat olla kiinnitettyjä käyttäjän vaatteisiin, jolloin ne tarjoavat läheisemmän keinon yksittäisen henkilön tarkkailemiseen. Kamerapohjaiset järjestelmät tarjoavat monipuolista tietoa ja ovat sen vuoksi yksi eniten käytetyimmistä sensorityypeistä älykkäissä ympäristöissä [20]. Vaikka visuaaliset sensorit tarjoavat joustavan ja monimuotoisen tavan toteuttaa ympäristössä tapahtuvaa havainnointia, sisältyy niihen myös aina ongelmia, sillä ne tarvitsevat oikean valaistuksen ja stabiilin taustan kohdetta tunnistettaessa.

Ihmistä aistivien kameroiden lisäksi älykkäässä ympäristössä voidaan tarjota myös toisen suuntaista visuaalisesti havaittavaa informaatioita, jolloin visuaaliset järjestelmät toimivat aktuaattoreina. Esimerkiksi suurten, seiniin kiinnitettyjen, näyttöjen avulla voidaan tarjota tietoa käyttäjän suuntaan. Yhdistettynä ne muihin sensorijärjestelmiin voidaan saavuttaa joustava käyttöliittymä koneen ja käyttäjän välille. [17]

Ihmisen mallintaminen perustuu kamerapohjaisia järjestelmiä käytettäessä konenäkömenetelmiin mahdollistaen hyvin monenlaisen tiedon käytön. Ihmisen paikantaminen ja liikkeen analyysi voi tapahtua 2-ulotteisessa (2D) tasossa, jol-

loin kuvista voidaan erottaa ihmisen vartalon eri osia ja yhdistellä osista etukäteistietoa käyttävä mallipohjainen rakenne. Se voi perustua myös suoraan peräkkäisistä kuvista irroitettujen piirteiden muutosten kuvaamiseen, jolloin ihmisen vartalo voidaan mallintaa joko 2D-kuvasta tai luoda usean kameran avulla 3-ulotteinen (3D) malli. [21]

Henkilöllisyyden ja aktiviteetin tunnistus voidaan jakaa kahteen osa-alueeseen, jolloin mallinnetaan joko henkilön kasvoja tai vartaloa. Kasvojen tunnistuksessa pyritään segmentoimaan kasvot kuvasta ja löytämään henkilöä kuvaavat piirteet tunnistukseen. Tälläiset henkilöllisyyden tunnistusmenetelmät kärsivät usein muuttuvista kuvakulmista eli menetelmät vaativat, että tunnistettava henkilö saadaan kuvattua suoraan edestä käsin. Kasvojen tunnistuksen lisäksi ihmisen erilaisia ilmeitä on pyritty tunnistamaan, sillä ne voivat antaa lisätietoa ympäristölle, esimerkiksi käyttäjän mielialasta [22]. Vartaloon perustuvaan tunnistukseen on käytetty pääasiassa kahta tekniikkaa: mallikuvion sovitusta (template matching) sekä tila-avaruusmallia (state-space model). Mallikuvion sovituksessa pyritään peräkkäisistä kuvista irrottamaan kuvaavia piirteitä, johon ennalta määrättyä mallia sovitetaan. Tila-avaruusmallia käytettäessä luodaan kuvasekvenssistä (aikasarjasta) tilastollinen malli, kuten HMM-malli [21]. Vartaloa mallinnettaessa henkilöllisyyden tunnistus voi perustua kävelytyylin tunnistamiseen, joka on esitellyistä konenäkömenetelmistä lähimpänä tämän työn lattia-anturiin perustuvaa tunnistusta. Kävelyylin tunnistaminen on esitelty tarkemmin kohdassa 2.3.2.

2.2.2. Audiosensorit

Näköaistin lisäksi kuuloaisti on tärkeä ihmisen mallintamisessa. Mikrofoneihin ja kaiuttimiin perustuvat audiojärjestelmät tarjoavat luonnollisen tavan kommunikaatioon ihmisen ja ympäristön välillä. Vaikka puheen- ja äänentunnistusta on tutkittu paljon, se sisältää paljon ongelmia. Esimerkiksi signaali-kohinasuhde kasvaa nopeasti etäisyyden kasvaessa mikrofonin ja kohteen välillä. Tämä aiheuttaa rajoitteita käyttäjän liikkeisiin ympäristössä, jos halutaan esimerkiksi puheentunnistukseen perustuva häiriötön henkilön tunnistus [20]. Rajoitusten vähentämiseksi ympäristöön voidaan asentaa laaja-alaisia staattisia vaiheistettuja mikrofoniryhmiä [23], jotka mahdollistavat joustavan käyttäjän paikantamisen ja tarjoavat äänisignaalia puheentunnistukseen [6].

Mikrofonien avulla tapahtuvan aistinnan lisäksi älykkäässä ympäristössä voidaan toteuttaa kaiuttimien avulla toisen suuntaista kommunikaatiota. Tällöin ympäristö voi välittää tietoa ja ohjeita käyttäjän suuntaan. Yhdessä suurten seinänäyttöjen kanssa kaiutinjärjestelmä voi tarjota luonnollisen käyttöliittymän. [17]

2.2.3. Puettavat ja liikuteltavat sensorit

Staattisten kameroiden ja mikrofonien lisäksi älykkäässä ympäristössä voidaan käyttää erilaisia mukana kannettavia ja liikuteltavia sensoreita. Kamerat ja mikrofonit voivat esimerkiksi olla puettavia, jolloin ne kulkevat käyttäjän mukana antaen näin tietoa käyttäjän näkökulmasta [24], ja vastaavasti tarjoten käyttäjälle

tilannekohtaista tietoa ympäristöstä. Liikuteltavat sensorit voivat olla myös pienpäätelaitteita kuten kämmenmikroja, jotka kommunikoivat langattoman lähiverkon kautta ympäristön ja käyttäjän välillä. Langattoman verkon avulla mukana kannettava laite voidaan paikantaa [25], ja tämän myötä myös sen käyttäjä. Tällaiset laitteet voivat sisältää myös omia antureita, kuten kiihtyvyysantureita, joiden avulla voidaan tunnistaa esimerkiksi käyttäjän aktiviteetti tietyssä tilanteessa [26].

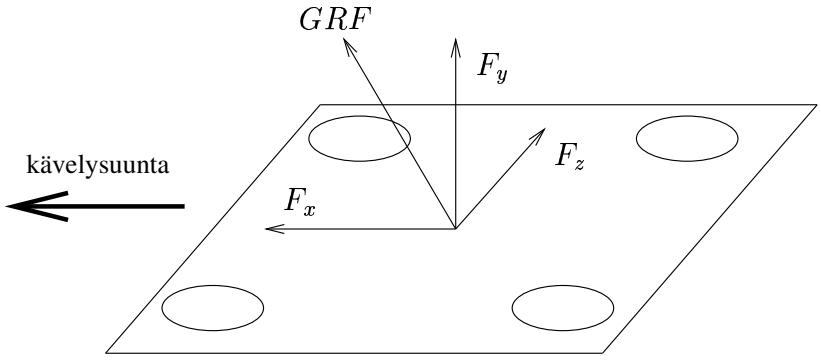
Muita kannettavia tekniikoita edustavat pienet infrapunavalolla toimivat Active Badge -tunnistimet [27] sekä radiotaajuuksilla toimivat RFID-tagit [28]. Yhteistä tällaisille järjestelmille on, että mukana kannettavan pienen tunnistinosan lisäksi ympäristöön sijoitetaan näiden tunnistimien kanssa kommunikoivia vastaanottimia. Tällaisilla järjestelmillä voidaan paikantaa käyttäjä ja sitoa henkilön tunnistus tähän pieneen laitteeseen. Puettavat sensorit voivat olla myös biosensoreita, jotka välittävät pelkän tunnistuksen ja paikkatiedon lisäksi tietoa käyttäjän fysiologisista ominaisuuksista, kuten sykkeestä ja verenpaineesta. Näitä tietoja voidaan käyttää henkilön tarkkailuun ja aktiviteetin tunnistamiseen. Edellä esitetyissä järjestelmissä huonona puolena on juuri ihmisen mallinnuksen sitoutuminen itse laitteeseen eikä ihmiseen. Tämä heikentää aistivan ympäristön näkymättömyyttä ja joustavuutta.

2.2.4. Lattiasensorit

Tässä työssä henkilön tunnistamiseen käytettiin lattiaan asennettua paineen tunnistavaa sensorijärjestelmää. Lattiaan asennettuja sensoreita voidaan käyttää henkilön tunnistamisen lisäksi käyttäjän havaitsemiseen, paikantamiseen ja jopa yksinkertaisten aktiviteettien tunnistamiseen. Ne tarjoavat menetelmän, joka on läpinänäkyvä käyttäjälle ja tarjoaa näin luonnollisen ja joustavan tavan käyttäjän mallintamiseen ilman puettavia sensoreita. Seuraavassa on esitelty samankaltaisia sensorijärjestelmiä kuin tässä työssä käytetty järjestelmä on. Tässä työssä käytetyn järjestelmän tarkempi kuvaus on esitetty luvussa 3.

Lattiasensorit ovat perustuneet pääasiassa voima-anturien käyttöön. Tällaisesta lattian ja askeleen välisestä voimasta käytetään nimitystä *Ground Reaction Force* (GRF), jolla voidaan mitata kolmeen eri suuntaan vaikuttavia voimia (F_x, F_y, F_z). Ne kuvaavat askeleen vaikutusta lattiaan vertikaalisessa, kävelyn etenemis- ja sivuttaissuunnassa jalan suhteen. Kuvassa 1 on esitetty GRF-voimasignaaliin vaikuttavat komponentit. Kuva mukailee lähteessä [29] esitettyä. Henkilöllisyyden tunnistuksen ja paikantamisen lisäksi GRF-signaalia on käytetty lääketieteessä kävelytyyliin vaikuttavien sairauksien tunnistamiseen [30].

SmartFloor lattia-anturijärjestelmä on kehitetty ihmisen tunnistamiseen ja paikantamiseen. Se perustuu voima-anturiin, joka mittaa siihen kohdistuvaa painoa ja inertia voimia GRF-profiilin määrittämiseksi. Systeemissä käytetään GRF-voiman pystysuuntaista komponenttia, jolloin askelsignaalin amplitudista erottuvat ajansuhteen päkijän ja varpaiden isku, sekä niiden välinen siirtymä. Laitteistolla voidaan mitata myös kiihtyvyyttä ja paikkaa. Systeemi koostuu kolmesta komponentista: varaussolusta, teräslevystä ja datankeruulaitteistosta. Mittausyksikkö on 50x50 cm kokoinen teräslevy, jonka jokaisessa neljässä nurkassa on varaussolu voiman mittaamista varten. Varaussolut ovat sylinterin muotoisia, joiden halkaisija on 25 mm ja paksuus 3 mm. Mittausyksikkö on sijoitettu lat-



Kuva 1. Voimasensori mittaa laattaan kohdistuvaa kolmesta suuntakomponentista koostuvaa GRF-voimaa. Askelanalyysissä on käytetty pääasiassa kohtisuoraa komponenttia F_y, jota esimerkiksi laatan joka kulmiin kiinnitetyt varaussolut tai pietsoelektrodiset anturit mittaavat.

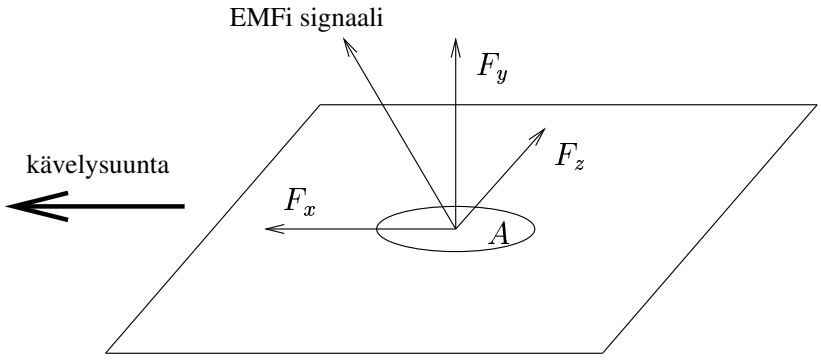
tiamaton alle, jolloin se on huomaamaton käyttäjälle eikä aiheuta liukastumisia. Analoginen signaali digitalisoidaan 150 Hz näytteistystaajuudella ja tallennetaan PC-koneelle mallintamista varten. [8]

ActiveFloor käyttää samaa mekanismia kuin SmartFloor, jossa varaussolut on asetettu teräslevyjen nurkkiin. Mittausyksiköitä on käytössä 16, jotka muodostavat 4x4 matriisin. Mittausjärjestelmän koostuessa useista yksiköistä usealle levylle jakaantuneet askeleet aiheuttavat haasteita signaalinkäsittelyyn. Kaikkien 16 levyn tarjoama signaali digitalisoidaan 500 Hz näytteistyksellä ja tallennetaan PC-koneelle. Järjestelmällä havaitaan lattiaan kohdistuvat 50 g suuruiset painonmuutokset. Koska varausolut mittaavat staattisia voimia, voidaan niitä käyttää henkilön paikantamisen lisäksi myös esineiden paikantamiseen. Lattiaa voidaan käyttää myös mittaamaan siihen kohdistuvaa kokonaismassaa. Tällöin voidaan esimerkiksi tarkkailla, onko henkilö vienyt jotain mukanaan tilasta poistuessaan. [9]

Edellä esitettyjen varaussolujen lisäksi voima-antureina voidaan käyttää pietsoelektronisia antureita, jotka toimivat kapasitiivisesti eli mittaavat siihen kohdistuvaa voimaa dynaamisesti. Tällöin GRF-signaalin F_y komponentti havaitaan sen derivaattana, mutta voidaan esimerkiksi numeerisesti integroida esittämään F_y signaalia. Tällaisia antureita on myös kaupallisesti myynnissä [31]. Työssä [29] käytettiin kolmea pietsoelektroniseen anturitekniikkaan perustuvaa sensorilaattaa, jotka olivat 60x60 cm suuruisia. Järjestelmää sovellettiin henkilön tunnistamiseen yhdessä kamerapohjaisen järjestelmän kanssa.

Tässä työssä on käytetty lattia-anturina EMFi-kalvoa, joka mittaa voiman sijasta siihen kohdistuvaa paineen muutosta. Tällaisella anturilla voidaan mitata lattiaan kohdistuvia dynaamisia vaikutuksia, kuten pietsoelektronisia voima-antureita käytettäessä. Staattiset voimat, kuten älykkään ympäristön kalusteet, eivät näy pysyvänä vaikutuksena anturin tuottamassa signaalissa. Anturin ja lattian ominaisuudet kuvataan tarkemmin luvussa 3. Kuvassa 2 on esitetty EMFi-sensorin toiminta. Se mittaa paineenmuutosta eli tietylle pinta-alalle kohdistuvaa voimaa. EMFi-signaalissa voidaan ajatella näkyvän kaikki GRF-komponentit summattuna.

UbiFloor puolestaan käyttää 144 ON/OFF-tyyppistä 14x2,5 cm kokoista anturiliuskaa. Liuskat on asetettu limittäin 12,5 cm etäisyydelle toisistaan. Yksi anturisolu on pinta-alaltaan 900 cm^2 sisältäen 4 liuskaa. Koko anturilattian ala



Kuva 2. EMFi-sensori mittaa siihen kohdistuvaa paineen muutosta eli voiman aiheuttamaa vaikutusta tietylle pinta-alalle A. EMFi-signaali voidaan ajatella koostuvan suuntakomponenttien summana, näitä ei ole kuitenkaan mahdollista mallintaa erikseen kuten GRF-sensorin kohdalla.

on 1x3 m, ja sitä on käytetty henkilön tunnistamiseen. [10]

Lattiasensorit tarjoavat käyttäjän kannalta näkymättömän tavan aistia ympäristöä eikä ihmisen tarvitse käyttää mitään puettavaa tai liikuteltavaa laitteistoa mukanaan. Kamerapohjaisiin järjestelmiin verrattuna lattiasensoreita voidaan pitää juostavampina, koska ne eivät tarvitse toimiakseen minkäänlaista valaistusta. Ihmisen tunnistamisen ja paikantamisen lisäksi lattia-antureita voidaan käyttää myös esimerkiksi robotin havaitsemiseen ja yhteistyöhön ihmisen kanssa. Lattiasensoreilla voidaan lisäksi tunnistaa yksinkertaisia käyttäjän aktiviteetteja, kuten kyyristyminen, hyppy ja alastulo sekä istuminen ja nousu ylös [32].

2.2.5. Muut sensorit

Lattiasensoreiden lisäksi painesignaaliin perustuvia tunnistusmenetelmiä voidaan soveltaa myös huonekaluihin, kuten sänkyihin ja tuoleihin, jotka ovat sulautettu ympäristöön. Esimerkiksi tuoliin kiinnitetyllä painesensoreilla voidaan havaita käyttäjän staattisia istuma-asentoja ja käyttää tätä tietoa älykkään ympäristön ja monimuotoisten käyttöliittymien osana [33].

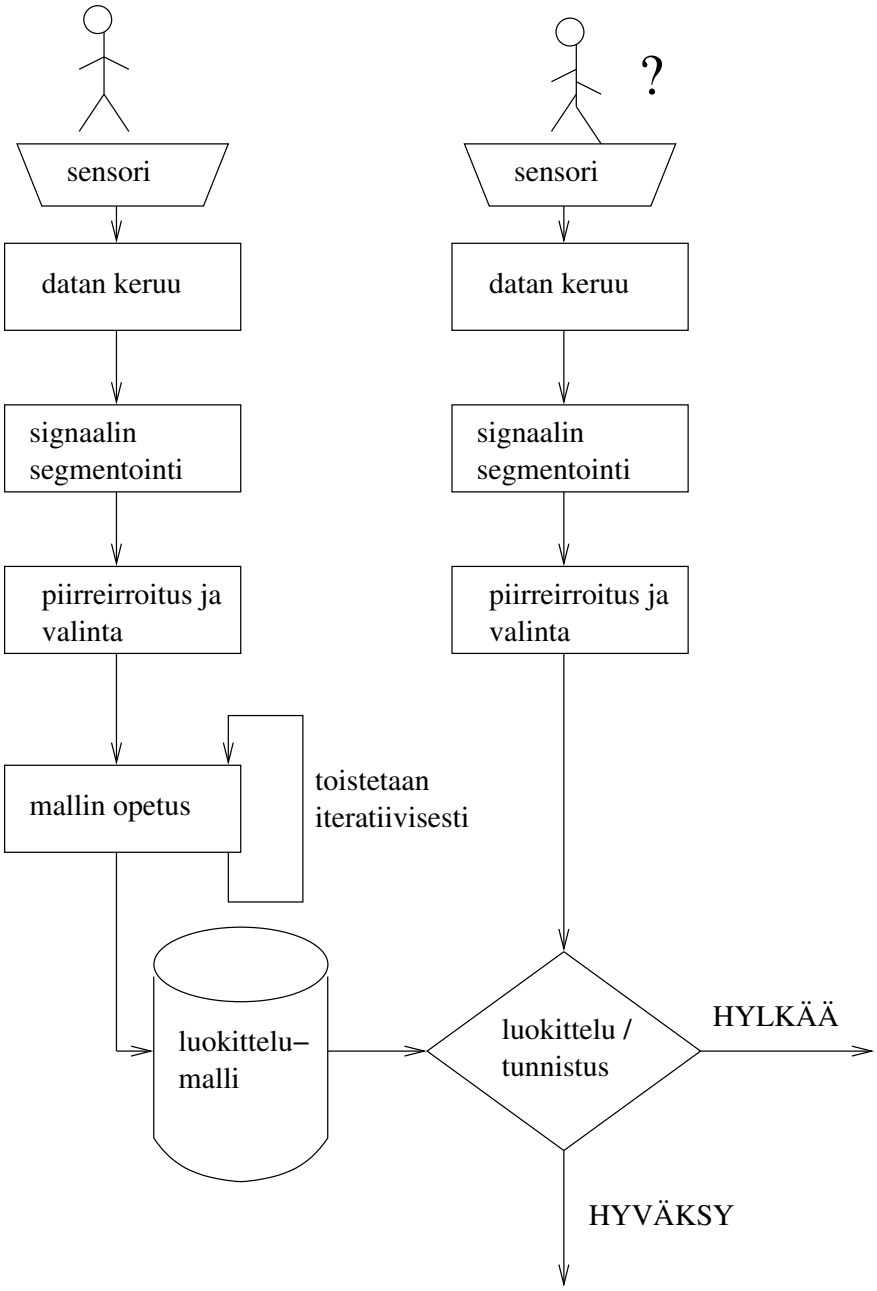
Pelkän käyttäjän liikkeiden tunnistamiseen tai esimerkiksi murtautujan havaitsemiseen voidaan käyttää monia yksinkertaisia sensoreita. Ne voivat olla esimerkiksi seiniin kiinnitettäviä liikkeentunnistimia, jotka toimivat passiivisten infrapunasensorien, mikroaaltojen tai ultraäänen avulla. Tällaiset sensorit voivat antaa hyvin rajoittunutta, mutta myös hyödyllistä tietoa käyttäjästä. [34]

2.3. Henkilön tunnistaminen

Henkilön tunnistaminen on prosessi, jossa tuntemattomalle henkilölle liitetään jotain menetelmää käyttäen henkilöllisyys. Henkilöllisyyden tunnistaminen voi olla varmentamista (autentikointi), jolloin ihminen ilmoittaa systeemille mahdollisen henkilöllisyytensä, eli “olenko se, joka väitän olevan?”. Tällöin henkilöllisyys varmennetaan vertaamalla tuntematonta mallia ilmoitetun henkilön opetettuihin malleihin hyväksymällä tai hylkäämällä annettu väite. Henkilöllisyyden tunnistus

voi olla myös tunnistusta (identifiointi), jolloin tuntemattoman henkilön mallia verrataan kaikkien henkilöiden opetettuihin malleihin, eli "kuka olen?". Tällöin valitaan jollakin kriteerillä tuntemattoman henkilön ominaisuuksia eniten vastaava malli opetetuista malleista. [35]

Kuvassa 3 on esitetty kaavio tavanomaisesta henkilön tunnistusjärjestelmästä. Ensimmäisessä osassa opetetaan tunnettujen henkilöiden pohjalta luokittelumalli ja toisessa osassa mallia käytetään tuntemattoman henkilön luokitteluun. Henkilö voidaan myös hylätä, jos häntä ei voida tunnistaa luotettavasti yhdeksikään ennalta tunnistetuista henkilöistä. Jos järjestelmää vastaavasti käytetään autentikointiin, tunnistettava henkilö ilmoittaa henkilöllisyytensä jonkin toisen menetelmän avulla. Tämän jälkeen annettua henkilöllisyyttä verrataan kyseessä olevan järjestelmän tulokseen, ja varmennus voidaan hyväksya tai hylätä.



Kuva 3. Kaaviokuva henkilön tunnistusjärjestelmästä.

Henkilöllisyyden varmentamista käytetään yleensä turvallisuusjärjestelmissä, joissa halutaan äärimmäisen luotettava varmennus henkilöllisyydestä. Luokittelun kannalta sitä voidaan pitää binääriluokitteluna, kun taas identifiointi on monen luokan ongelma, ja usein myös 2-luokkaista tapausta kompleksisempi. Älykkäissä ympäristöissä tunnistamiseen käytetään kuitenkin useimmin identifiointia, kuten tässä työssä. Vaikka identifointimenetelmien luotettavuus on varmennusta alhaisempaa, ne tarjoavat usein riittävän joustavia, ympäristöön sulautettuja menetelmiä, jotka ovat älykkäiden ympäristöjen vaatimusten mukaisia.

Henkilön identifiointi voi perustua joko mukana kannettavaan (esim. ID-tag, avain) tai biometriseen tunnisteeseen (kasvot, silmän iiris, käden geometria, ääni, kävelytyyli). Biometriset tunnistusmenetelmät sitovat tunnistuksen aina kyseessä olevaan henkilöön, kun taas esimerkiksi älykorttia tai tunnuslukua käytettäessä ei voida olla varmoja, että käyttäjä on juuri se henkilö, jolle kortti tai tunnusluku oikeasti kuluu.

2.3.1. Biometrinen henkilön tunnistaminen

Biometrisellä henkilön tunnistamisella tarkoitetaan järjestelmää, jossa pyritään ihmisen fysiologisten tai käyttäytymiseen perustuvien ominaisuuksien avulla määrittämään henkilöllisyys. Fysiologisilla ominaisuuksilla tarkoitetaan sellaisia ominaisuuksia, jotka ovat hyvin stabiileja, eivätkä muutu ajan myötä. Näitä ovat esimerkiksi silmän iiris, sormenjälki, ja käden muoto. Henkilön käyttäytymiseen liittyvät ominaisuudet ovat sitä vastoin muuttuvia, eli ne voivat vaihdella iän, vammojen tai jopa mielialan mukaan. Tällaisia ominaisuuksia ovat esimerkiksi ääni ja puhetyyli, käsiala sekä kävelytyyli. [29]

Ihmisen fysiologisiin ominaisuuksiin perustuvilla biometrisillä henkilön tunnistusmenetelmillä päästään yleensä erittäin luotettaviin tuloksiin [35]. Yleensä ne kuitenkin tarvitsevat jonkin erityisen laitteen käyttöä, jolloin älykkään ympäristön vaatimusten mukaan menetetään tiettyä vapautta. Ihmisen käyttäytymiseen perustuvat menetelmät sitä vastoin tarjoavat luonnollisella tavalla tapahtuvan tunnistuksen.

2.3.2. Kävelytyyliin perustuva henkilön tunnistaminen

Henkilön kävelytyyliin perustuvaa tunnistamista voidaan joustavuutensa ja luonnollisuutensa ansiosta pitää yhtenä eniten älykkään ympäristön vaatimuksiin soveltuvana menetelmänä. Tässä työssä toteutetut menetelmät liittyvät juuri henkilön kävelytyylin mallintamiseen, ja tarkemmin yksitäisten askelien tuottaman paineanturisignaalin analyysiin. Kävelytyylin mallinnusta on lääketieteen alueella tehty jo kauan. Vuonna 1964 Murray [36] tutki ihmisen kävelytyyliä ja tulokset osoittivat, että ihmisen kävelytyyli on ainutlaatuinen, kun kaikki kävelyliikkeeseen vaikuttavat ominaisuudet otetaan huomioon. Ihmisen on todettu myös helposti tunnistavan tuttujen henkilöllisyys pelkän kävelytyylin perusteella [37]. Vaikka kävelytyyliä voidaan pitää hyvänä henkilöllisyyden tunnistimena, on tietokonepohjaisissa järjestelmissä usein mahdotonta kehittää sensorijärjestelmää, jolla nämä kaikki kävelyliikettä kuvaavat dynaamiset piirteet saataisiin irroitettua.

Kävelytyylin tunnistamiseen ja mallintamiseen on käytetty kahta eri päätekniikkaa. Yleisin tekniikka on perustunut kamerapohjaisiin järjestelmiin konenäkömenetelmiä käyttäen, jolloin tunnistus perustuu sekvenssiin peräkkäisiä kuvia videokameralta. 3D-mallia käytettäessä kameroita voi olla useita. Kuvasekvensseistä lasketaan tiettyjä kuvaavia vartalon ja raajojen asentoihin sekä kävelyn taajuus- ja vaiheominaisuuksiin liittyviä piirteitä [38]. Liikkuvan kohteen tehokkaaseen mallintamiseen voidaan käyttää myös parametristä ominaisavaruusesitystä [39]. Tällöin kävelysekvenssi voidaan kuvata dynaamisilla mallikuviopiirteillä (template features), jonka jälkeen nämä piirteet muunnetaan alkuperäistä mallia pienempidimensioiseen ominaisavaruuteen ortogonaalisella muunnoksella ja suoritetaan luokittelu lähimmän opetussekvenssin mukaan uudessa piirreavaruudessa [40].

Tavallisten mallikuvioiden sovitteiden lisäksi enemmän tilastollisia menetelmiä on sovellettu esimerkiksi jatkuvien HMM-mallien muodossa käyttämällä malliin kävelijän ääriviivakuvaa (silhuetti) jokaisesta kävelysekvenssin kuvasta. Tällaisella menettelyllä päästiin 5 henkilön testiaineistoa käyttäen noin 90% kokonaistunnistukseen. [41]

Yksinkertaista menetelmää, jossa piirteinä käytettiin henkilön pituutta sekä astuntaan liittyviä piirteitä, kuten askeleen pituus sekä askelten määrä ajan suhteen, on myös sovellettu. Luokittelu suoritettiin lähimmän naapurin menetelmällä. Piirteet eivät yksistään antaneet tarpeeksi luotettavaa tunnistusta, mutta tarjoavat kuitenkin kävelijästä ominaista ja erottavaa tietoa, jota voidaan käyttää hyväksi yhdessä dynaamisten piirteiden kanssa [42]. Parhaiden tunnistustulosten saavuttamiseksi eri tekniikoita on myös yhdistelty. Esimerkiksi kävelytyylin ja kasvojen tunnistuksen yhdistämisellä on saavutettu todella luotettavia tuloksia henkilön tunnistuksessa [43].

Toinen tekniikka on lattiaan asennettuihin antureihin perustuvat järjestelmät. Järjestelmästä saatava informaatio on nyt rajoittuneempaa kuin videojärjestelmissä, sillä tietoa saadaan vain diskreeteistä yksittäisten askelten ominaisuuksista ja mahdollisesti niiden välisistä etäisyyksistä ja ajasta. ActiveFLoor [9] ja SmartFloor [8] käyttävät tunnistamiseen GRF-profiilin vertikaalista voimaa(F_y), kuten aikaisemmin jo mainittiin. ActiveFloor-projektissa tunnistusmenetelmänä käytettiin HMM-malleja. Piirresekvenssi laskettiin segmentoidun askeleen mukaan käyttäen liukuvaa ikkunaa. Paras kokonaistunnistus 15 henkilön askeldatalle oli 91%. SmartFloor projektissa käytettiin yksinkertaisempaa mallia. Piirteitä valittiin askeleen amplitudin ja ajan muutoskohdista ja luokittelu suoritettiin käyttämällä NN-menetelmää. Tällä menettelyllä saavutettiin 93% kokonaistunnistus 15 henkilön osalta. Lisäksi tutkimuksessa selvisi, ettei kengän tyypillä ollut suurta vaikutusta henkilöllisyyden tunnistuksessa GRF-profiilia mittaavaa anturia käytettäessä.

UbiFloorin [10] yksinkertaiset anturit eivät vastaavasti anna yhtäpaljon informaatiota kuin edelliset, koska anturit ovat pelkästään ON/OFF-tyyppisiä. Ne tarjoavat rajoitettua paikkatietoa, miten jalka on osunut sensoreihin sekä vertikaalisessa että horisontiaalisessa suunnassa. Lisäksi tarkkailtiin yksittäisten askelten välisiä ominaisuuksia käyttäen tunnistukseen viittä peräkkäistä askelta. 10 hengen testiryhmällä saavutettiin noin 90% kokonaistunnistus käyttäen tunnistukseen MLP-menetelmää.

Työssä [29] toteutettiin autentikointijärjestelmä, joka yhdistää voimaa mittaavan lattia-anturin sekä kävelyä sivusuunnasta tallentavan kameran. Molemmista

sensoreista irroitettiin dynaamisia kävelyyn liittyviä piirteitä. Lattia-anturin piirteinä käytettiin GRF-profiilin derivaattaa, josta laskettiin ikkunoitu tehontiheysspektri. Kameran tuottamista kuvista etsittiin yksi kävelysekvenssi, josta valittiin histogrammi- ja mallikuviopiirteitä. Tämän jälkeen piirrevektorien dimensioita vähennettiin pääkomponenttimuunnoksella. Kahden sensorin data fuusioitiin ja henkilöllisyys varmennettiin käyttäen Bayesin päätöskriteeriä. Testiryhmä koostui 16:sta henkilöstä, ja jokaiselta kerättiin 10 kävelysekvenssiä, joista 6 käytettiin mallin opettamiseen ja loput 4 sen testaamiseen. Kokonaisvirhe henkilöllisyyden varmennuksen osalta oli 1.6%, kun lattia-anturilla päästiin yksistään 5.3% virheeseen.

2.3.3. Askeltunnistuksen vaiheet

Tässä työssä toteutettu kävelijän henkilöllisyyden tunnistus perustuu kuvassa 3 esitettyyn rakenteeseen. Ensimmäiseksi tarvitaan sensori, jonka avulla tunnistettava henkilö saadaan havaittua. Tämän jälkeen sensorilta saatava signaali kerätään tiedonkeruulaitteistoa käyttäen ja tallennetaan jatkoanalyysiä varten. EMFi-sensori ja tiedonkeruulaitteisto esitetään tarkemmin luvussa 3.

Koska sensori ja laitteisto valittiin jo ennen tämän työn aloittamista, työssä keskitytään signaalin jatkokäsittelymenetelmien kehittämiseen. Toisin sanoen, kävelijän tunnistusmenetelmät pyritään kehittämään olemassa olevaan sensoritietoon perustuen. Näin ollen seuraavat luvut keskittyvät pääasiassa tunnistusjärjestelmän luokittelu- ja tunnistusvaiheen kuvaamiseen kehitettyjen menetelmien avulla.

Olennainen osa henkilön tunnistusjärjestelmää on signaalin esikäsittely. Tyypillisesti tiedonkeruulaitteistolta saatavasta signaalista pyritään irroittamaan jotain olennaista tietoa. Tässä työssä tunnistus perustuu siis henkilön kävelytyyliin ja tarkemmin paineanturilta havaittujen yksittäisten askelten profiilin mallintamiseen. Signaalinkäsittelyn ensimmäisessä vaiheessa, kuten kuvassa 3 on esitetty, yksittäiset askeleet pyritään segmentoimaan raakasignaalista. Signaalin esikäsittelymenetelmiä kehitetään toisessa projektissa, ja ne on esitetty lyhyesti tämän työn toteutusosassa (ks. kohta 7.2.1).

Segmentoinnin jälkeen käytössä on yksittäisiä askelprofiileita, joiden perusteella henkilön identifiointi toteutetaan. Tunnistuksen toteuttamiseen käytetään tilastollisia hahmontunnistus- ja luokittelumenetelmiä, jotka opetetaan eri henkilöiden segmentoiduilla askelprofiileilla iteratiivisesti. Askeltunnistusmenetelmien teoria esitellään luvussa 4. Jotta tunnistusmenetelmät toimisivat hyvin ja tunnistettavan henkilön ominaisuudet saataisiin kuvattua mahdollisimman hyvin, askelprofiilista pitää irroittaa kuvaavia piirteitä. Niiden tulisi kuvata saman henkilön näytteet mahdollisimman yhtenäisesti, ja toisaalta erottaa eri henkilöiden askeleet.

Piirteiden määrä tulisi valita sopivan kokoiseksi. Tähän voidaan käyttää esimerkiksi osapiirrejoukon testausta, jolloin suuresta joukosta valitaan jotain kriteeriä käyttäen tilannetta parhaiten kuvaava osajoukko. Piirteenvalinta voidaan liittää myös osaksi mallin opetusta, jolloin tunnistusjärjestelmän adaptiivisuus paranee menetelmän pystyessä automaattiseen piirteen skaalaukseen ja valintaan. Piirreirroitus- ja piirrevalintamenetelmät esitellään tarkemmin luvussa 5 ja valitut piirteet askeldataan liittyen toteutusosassa (ks. kohta 7.3).

Luokittelumenetelmiin liittyy aina epävarmuutta. Henkilöä tunnistettaessa menetelmiin voidaan liittää hylkäyskriteeri, jolloin epävarmat näytteet voidaan hylätä kuulumattomina mihinkään ennalta opetettuun luokkaan, jolloin järjestelmän luotettavuus kasvaa. Epävarmuus voi liittyä esimerkiksi sensorilta saataviin kohinaisiin näytteisiin ja mittausvirheisiin, tai askeltunnistuksessa henkilö ei ole kävellyt luonnollisella tavallaan ja askelprofiili on täten huono. Hylätyn askeleen jälkeen voidaan esimerkiksi kerätä lisää näytteitä, jos halutaan varmistua henkilöllisyydestä tai hylätä tunnistus kokonaan tämän käyttäjän osalta. Tässä työssä kehitettiin 2-tasoinen luokittelumenetelmä, joka käyttää hylkäyskriteeriä ja kolmea peräkkäistä askelta luotettavan tunnistuksen aikaan saamiseksi. Lisäksi tämä 2-tasoinen luokittelumenetelmä parantaa järjestelmän adaptiivisuutta, sillä se tarjoaa yhden keinon tuntemattomien henkilöiden automaattiseen havaitsemiseen. Nämä menetelmät esitetään luvussa 6.

Edellä mainittujen vaiheiden jälkeen henkilön tunnistusjärjestelmä on valmis ja sitä voidaan käyttää tuntemattoman ihmisen identifiointiin, kuten kuvan 3 oikeapuoli osoittaa. Tällöin tuntemattoman kävelijän signaali havaitaan sensorilla ja kerätään talteen. Seuraavassa vaiheessa raakasignaalista segmentoidaan yksittäiset askeleet ja niistä irroitetaan vastaavasti valitut piirteet. Lopuksi tuntematonta piirrevektoria verrataan opetettuun luokittelumalliin, ja valitaan käytetyn kriteerin mukaan identieetti. Jos tätä ei kyetä tekemään luotettavasti, henkilö voidaan myös hylatä ja suorittaa jotain muita toimenpiteitä. Tunnistusmenetelmien käytännön toteutus ja testaus esitetään luvussa 7.

2.4. Älykkään ympäristön sovelluskohteet

Älykkäässä ympäristössä tapahtuva ihmisen mallintaminen ja sen soveltaminen voidaan jakaa kolmeen osa-alueeseen. Ensimmäinen osa on turvallisuus, eli pyritään tunnistamaan vaaralliset tilanteet ja estäämään tuntemattomien henkilöiden pääsy heille sallimattomiin paikkoihin. Toisena osana ovat taloudelliset ja ympäristöön liittyvät seikat, joilla tarkoitetaan esimerkiksi erilaisten toimilaitteiden säätämistä käyttäjän tarpeiden maksimoimiseksi ja energian kulutuksen minimoimiseksi. Kolmantena osana ovat erilaiset mukavuus asiat, kuten tehokkuuden parantaminen kotiaskareissa ja kehittyneet ympäristöön sulautetut käyttöliittymät ihmisen ja koneen välillä. [34]

Älykkäiden ympäristöjen sovelluskohteita on monenlaisia. Yleisimpänä ovat älykodit. Näissä voidaan soveltaa kaikkia kolmea edellä mainittua osa-aluetta. Tuntemattomien henkilöiden pääsy voidaan estää älykkäillä henkilöllisyyden tunnistusmenetelmillä. Lapsia, vanhuksia sekä sairaita voidaan valvoa [6] ja tarjota heille palveluja. Talossa käytettäviä toimilaitteita kuten lämmistystä ja valaistusta voidaan säätää käyttäjien tarpeiden mukaan, jolloin voidaan esimerkiksi säästää energiaa oppimalla ihmisten käyttäytymisrutiinit [44]. Älykotien lisäksi tekniikoita voidaan soveltaa julkisiin ympäristöihin kuten työpaikkojen toimistoihin [45], ja kokoushuoneisiin mahdollistaen esimerkiksi aika- ja paikkariippumattomat videopuhelut. Kouluissa voidaan parantaa vuorovaikutusta ja tehostaa etäopetusta [46]. Viihteellisemmästä käytöstä voidaan mainita erilaiset virtuaaliympäristöt sekä lasten interaktiiviset pelit [47].

3. EMFI-SENSORI OSANA ÄLYKÄSTÄ YMPÄRISTÖÄ

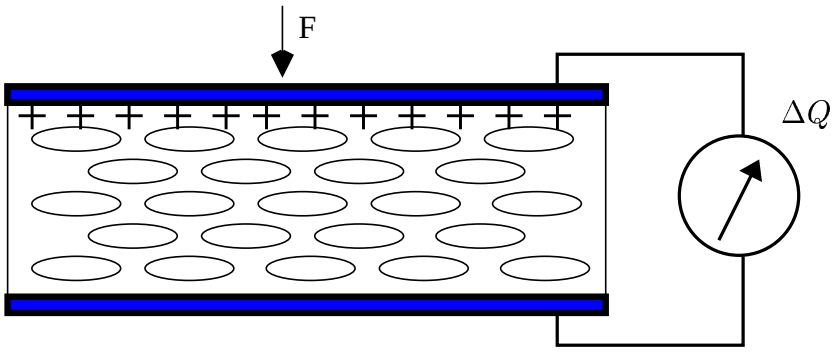
Tässä luvussa käsitellään EMFi-anturimateriaalin ominaisuuksia ja sen mahdollisia sovelluskohteita. Lisäksi esitellään tähän työhön liittyvä EMFi-lattia ja mittauslaitteisto sekä järjestelmän mahdollisia sovelluskohteita osana älykästä ympäristöä. EMFi-sensori mahdollistaa kävelijän havaitsemisen ja mittauslaitteistolla tunnistettavien henkilöiden askelprofiilit saadaan tallennettua jatkokäsittelyä varten henkilön tunnistusjärjestelmän ensimmäisen vaiheen mukaisesti.

3.1. EMFi-anturi

EMFi-kalvo (Electromechanical film) on hyvin ohut noin 0,01 mm paksuinen ja joustava kaksisuuntaisesti orientoitunut metallipäällysteinen polypropyleenikalvo. Sisäisen rakenteen ansiosta muovikerrokseen saadaan aikaan staattinen sähkövaraus, joka tapahtuu käyttämällä joko korona- tai elektronisuihkumenetelmää. EMFi-kalvon valmistusprosessissa saadaan kalvossa vaikuttavat sähköstaattiset ja sähkömagneettiset voimat mahdollisimman suuriksi ohuilla kalvokerroksilla. Valmistusprosessi tapahtuu jatkuvana puhallus- tai tasokalvoprosessina tavallisimmista muovimateriaaleista. [48]

Akustisen tai mekaanisen voiman kohdistuessa EMFi-kalvoon sen paksuudessa tapahtuu muutos. Tämä muutos synnyttää suuren varauksen johtavien metallikerrosten välillä, ja tämä varaus voidaan havaita jännitteenä. Tällöin EMFi-kalvo toimii anturina. Kalvoa voidaan käyttää myös päinvastoin, jolloin se muuntaa sähköisen energian värähtelyksi ja ääneksi. Tällöin kalvo toimii aktuaattorina. Kalvon pysyvä polariteetti mahdollistaa sovellukset ilman tehonkulutusta ja kustannuksia lisäävää ulkoista jännitelähdettä. [49]

EMFi-kalvo on mukautuva erilaisiin paikkoihin, sillä kalvosta voidaan leikata haluttuja muotoja ja siihen voidaan tehdä reikiä ilman, että sen mittausominaisuudet muuttuvat [48]. Lisäksi se on myös erittäin taipuisa materiaali. Kuvassa 4 on esitetty poikkileikkaus EMFi-kalvon sisäisestä rakenteesta.



Kuva 4. EMFi-kalvon poikkileikkauskuva, jossa metallikerrosten välissä olevat kaasukuplat muodostavat pysyvän varauksen polypropyleeniin. Pintaan kohdistuva voima F aiheuttaa paineenmuutoksen, jonka tuottama varausmuutos ΔQ voidaan havaita jännitteenä.

3.2. EMFi-sovellukset

EMFi-kalvoon perustuvilla antureilla voidaan mitata dynaamista voimaa tai painetta sekä värähtelyjä. Antureita on käytetty erilaisiin lääketieteellisiin mittauksiin, esimerkkinä sykkeen mittaaminen ranteesta [50]. Halvan hintansa myötä EMFi-antureita voidaan käyttää myös laaja-alaisemmissa sovelluksissa. Materiaali voidaan sijoittaa lattiapinnoitteen alle, jolloin se mahdollistaa lattiaan aiheutuvien painemuutosten hyödyntämiseen osana älykästä ympäristöä. Lattia-anturia on sovellettu esimerkiksi virtuaalisessa peliympäristössä tietokonepelin ohjaamiseen [51]. Muita sovelluksia ovat esimerkiksi koe-eläinten häkkien pohjaan sijoitettavat anturiliuskat, motoristen aktiviteettien mittaamiseen ja liikkeen rekisteröintiin sekä urheilun puolella sovelluksia on kehitetty mäkihypyn alastulokohdan paikantamiseen, jolloin hypyn pituus voidaan mitata automaattisesti ilman mittamiehiä [52]. Myöskin mäkihypyn ponnistuskohdan tarkkailemiseen on kehitetty järjestelmä, jossa hyppyrin nokalle asennettujen EMFi-liuskojen avulla voidaan tarkkailla hyppääjän mäkeen aiheuttamia voimia [53].

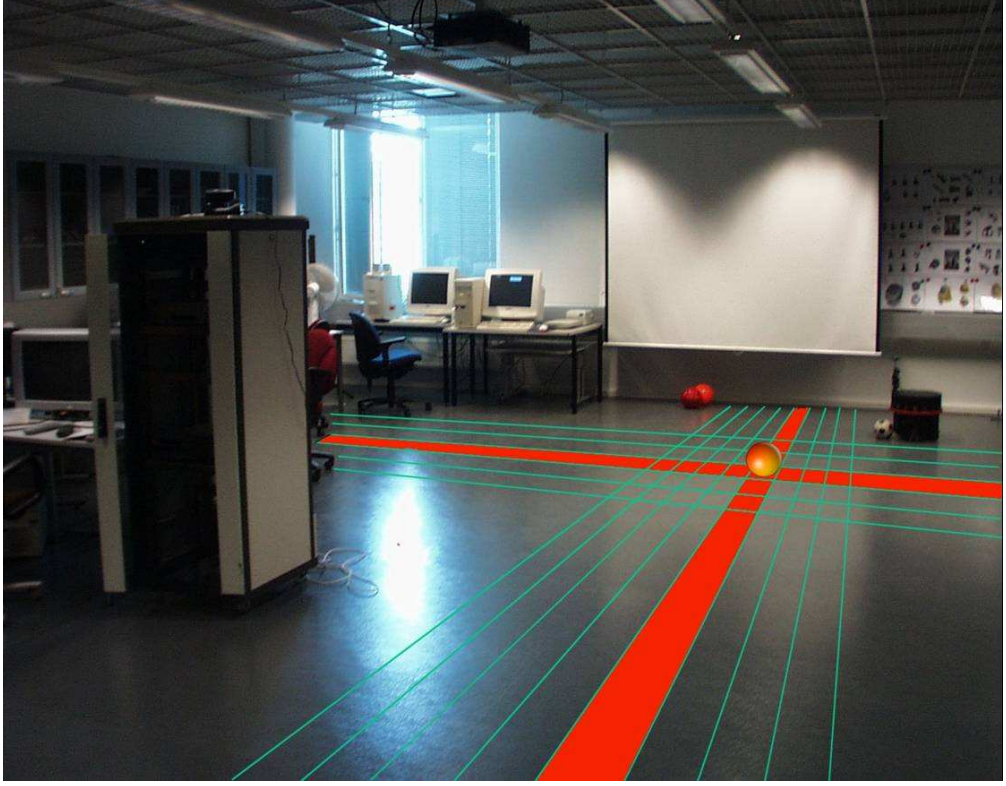
EMFi-anturi toimii laajalla taajuusalueella, joten se mahdollistaa niiden käytön erikokoisina akustisina antureina. Anturipinnoilla voidaan peittää kokonaisia katto-, lattia- ja seinäpintoja akustisten voimien tarkkailemiseen. EMFi-kalvo voidaan tarvittaessa myös vahvistaa niin herkäksi, että se rekisteröi jopa lattialla seisovan ihmisen sydämen lyönnit [54]. Tällöin kuitenkin voimakkaampien signaalien, kuten kävelyaskelten, tarkkailu kärsii. EMFi-materiaali on suosittu myös pienialaisissa sovelluksissa, kuten erilaisten kielisoitinten mikrofoneissa sekä tasomaisissa kaiuttimissa [48].

Integroimalla tasomainen mikrofoni, kaiutin ja tarvittava ohjauselektroniikka, on mahdollista rakentaa järjestelmä, joka kuuntelee ympäristöään ja vaimentaa vastaäänellä melua tai vahvistaa toivottuja ääniä. Sovelluskohteita ovat esimerkiksi rakennusten, autojen ja lentokoneiden matkustamojen äänieritys. Konserttisalien akustiikan muokkaaminen on myös mahdollista tällaisen järjestelmän avulla. [54]

3.3. EMFi-lattia

Älykkäiden järjestelmien tutkimusryhmässä EMFi-materiaalia on käytetty laajan lattia-anturin toteuttamiseen, jota tässä työssä käytettiin. Lattian peittoala on noin 100 neliömetriä kattaen yhden kokonaisen huoneen. EMFi-lattia koostuu 64:stä pitkästä anturiliuskasta, jotka on sijoitettu huoneen normaalin lattiapinnoitteen alle. Pitkät liuskat muodostavat 30x34 ristikkorakenteen, jossa jokainen liuska ylettyy vastakkaisesta seinästä toiseen yksittäisen liuskan ollessa 30 cm leveä. Kuvassa 5 on esitetty tutkimuslaboratorio, johon EMFi-lattia on asennettu.

Pitkien liuskojen käyttäminen on valittu siksi, että prosessoitavien kanavien ja johdotusten määrä saadaan mahdollisimman vähäiseksi säilyttäen kuitenkin painemuutosten riittävän tarkka paikantaminen. Jos lattia olisi toteutettu pienillä neliön muotoisilla anturipaloilla, olisi käsiteltävien kanavien ja johdotusten määrä helposti noussut tuhansiin kappaleisiin, kun taas toteutetulla järjestelyllä niitä tarvittiin vain 64 kappaletta. Pienien palojen avulla henkilön paikantaminen saataisiin kuitenkin suoraan palojen sijainnista, kun taas ristikkorakennetta käyt-



Kuva 5. EMFi-lattia koostuu pitkittäisistä lattiapinnoitteen alle sijoitetuista liuskoita. Muutaman liuskan sijoittuminen voidaan nähdä kuvassa.

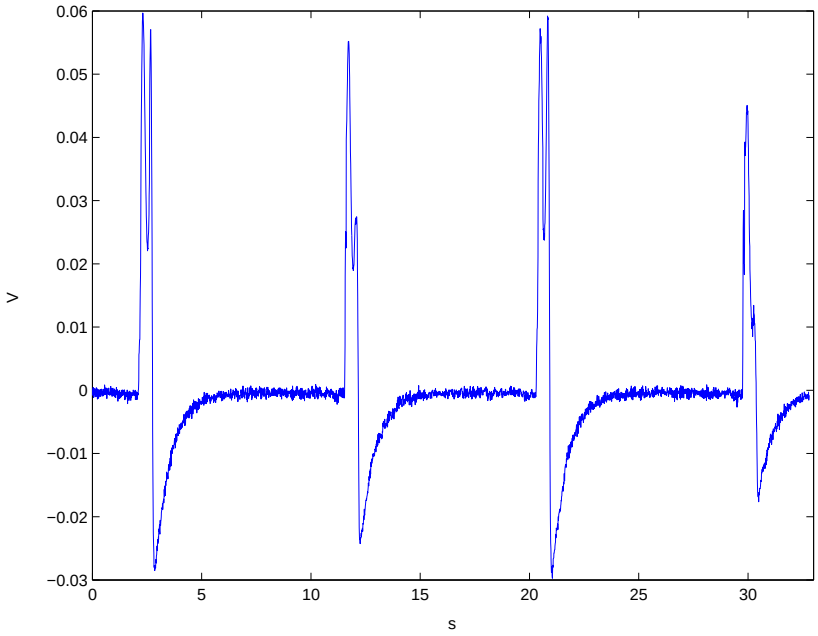
täen on mahdollista, että yhdellä kanavalla näkyy usean eri henkilön tuottamaa signaalia samalla ajanhetkellä. Tämä aiheuttaa haasteita signaalin käsittelyyn usean yhtäaikaisen henkilön paikannuksessa.

3.4. EMFi-laitteisto ja -signaali

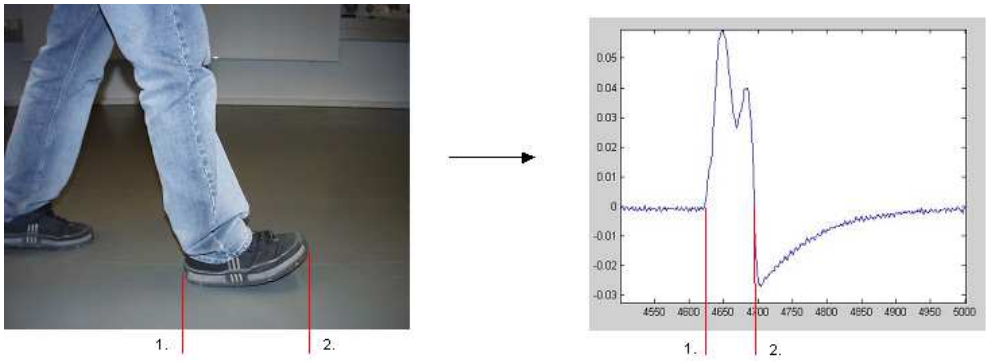
EMFi-lattian kaikki 64 kanavaa tuottavat jatkuva-aikaista signaalia, jotka ohjataan PC-tietokoneelle (Pentium 1700 MHz, 256 MB muistia). Analoginen signaali muutetaan digitaaliseen muotoon ja vahvistetaan PCI-väylään kytketyllä AD-muuntimen omaavalla National Instrumentin tiedonkeruukortilla (malli PCI-6033E) käyttäen 100 Hz näytteenottotaajuutta. Kortissa on 64 analogista tuloa, 16 bitin resoluutiolla ja näytteistystaajuus on mahdollista valita 0,1 - 1,56 kHz väliltä. 100 Hz taajuus valittiin sopivaksi askeldatan mallintamiseksi, sillä tyypillisen kävelyaskeleen pituus on noin 60-80 ms, eivätkä muutokset ole kovin nopeita, joten hyödyllinen informaatio on matalilla taajuuksilla. Tässä työssä tiedonkeruukortilta saatava digitaalinen signaali tallennettiin tiedostoon jatkokäsittelyä varten.

Kuvassa 6 voidaan nähdä tyypillistä kävelyaskelsignaalia yhdeltä EMFi-liuskalta nauhoitettuna. EMFi-lattian kapeiden liuskojen takia askeleet eivät osu aina kokonaan yhdelle liuskalle. Esimerkiksi kuvassa 6 ensimmäinen ja kolmas askel ovat osuneet tälle liuskalle, kun taas neljännen askeleen alkupuolisko eli kantapään vaikutus näkyy tällä kanavalla, kun taas päkijän ja varpaiden vaikutus on kohdistunut vierekkäiselle kanavalle. Kuvassa 7 on esitetty hyvän kävelyaskeleen muodostuminen kantapään iskusta (1.) varpaiden nousuun (2.). Segmentoiduissa

askelissa voidaan nähdä askelsignaalin lopussa negatiivinen jännite (ks. kuva 7), joka voidaan käsittää EMFi-kalvon palautumisajaksi painemuutoksen päätyttyä. Osassa tunnistuskokeista segmentoituun askeleeseen otettiin mukaan tämä kalvon palautuminen, kun taas osassa yksittäisen askeleen loppupiste valittiin negatiivisimman jännitekohdan mukaan eikä itse palautumista huomioitu.



Kuva 6. EMFi-signaalia yhdeltä kanavalta nauhoitettuna. X-akseli kuvaa aikaa (sekunteja) ja y-akseli painesignaalin vaikutusta jännitteenä (voltteja).



Kuva 7. Yhdelle liuskalle osuva hyvä askel. Kuvaan on merkitty askelsignaalin alku (1.), kun kantapää osuu liuskaan sekä signaalin loppu (2.) päkijän ja varpaiden vaikutuksen jälkeen.

3.5. EMFi-lattiaan perustuvat sovellukset

Älykkäiden järjestelmien tutkimusryhmässä EMFi-lattiaa tullaan käyttämään osana älykkään ympäristön tutkimusta. Tavoitteena on rakentaa ympäristö, joka

on proaktiivinen eli kykenee adaptoitumaan muutoksiin, oppimaan rutiineja, vuorovaikuttamaan sekä ennustamaan ihmisen liikkeitä ja käyttäytymistä. Näiden asioiden totettamiseksi ensimmäisiä tutkimuskohteita ovat signaalin esikäsittely ja segmentointi [15], joiden avulla toteutetaan ensimmäiset sovellukset liittyen henkilön paikantamiseen sekä tässä työssä toteutettuun kävelijän tunnistamiseen [14, 11, 12, 13] EMFi-lattian avulla. Tarkoituksena on painesignaalin avulla saada reaaliaikaista tietoa henkilöiden fyysisestä sijainnista ja henkilöllisyydestä. Henkilön tunnistaminen ja paikantaminen mahdollistavat jatkossa korkeamman tason henkilökohtaisten toimintojen toteutuksen.

Henkilön tunnistamisessa ja paikantamisessa pyritään mahdollistamaan reaaliaikaisuus, jolloin tunnitusmenetelmiltä vaaditaan adaptiivisuutta. Uusien ihmisten tunnistaminen ja mukautuminen aikaisemmin tunnistettujen henkilöiden muutoksiin tarjoavat haasteista askeltietoon perustuvaan luokitteluun ja tunnistamiseen. Esimerkkisovellus voisi olla käyttäjätunnuksen verifiointi älykkäässä huoneessa olevalle tietokoneelle. Tällöin käyttäjän henkilöllisyys tunnistettaisiin luonnollisesti tämän kävellessä huoneeseen. Myöskin muiden erilaisten kotielektroniikkalaitteiden, kuten audiolaitteiden ja digitaalisen television ohjaaminen voisi osittain perustua henkilökohtaiseen tunnistamiseen ja paikantamiseen tarjoten näin kullekin henkilölle itsenäisiä palveluja.

Lattiaa voidaan soveltaa liikkeentunnistuksen avulla tapahtuvaan monitorointiin älykotiympäristössä, jossa vanhusten tai lasten liikkeitä tunnistettaisiin ja mahdollisista epäkohdista hälytettäisiin. Liiketunnistuksen lisäksi esimerkiksi sairaaloissa tai vanhainkodeissa paineherkkäanturi voitaisiin asentaa tarkkailtavan henkilön vuoteeseen, jolloin liiketiedon lisäksi voitaisiin tarkailla myös henkilön biosignaaleja, kuten sykettä.

Ihmisen lisäksi EMFi-lattian avulla on tarkoitus myös mallintaa oppivan robotin käyttäytymistä. Sen liikkeitä pyritään seuraamaan lattian sekä langattoman verkon paikannuksen ja kattoon kiinnitettyjen kameroiden avulla. Usean robotin ja ihmisen yhteistyötä pyritään myös soveltamaan. Tällöin sovelluskohteita ovat esimerkiksi lasten leikeissä, kotiaskareissa ja vanhusten auttamisessa autonomisen robotin avulla. Kuvassa 8 on hahmoteltu tilanne tutkimuslaboratoriosta, jossa älyhuone on jaettu useammiksi pienemmiksi huoneiksi. Tällaisella järjestelyllä on mahdollista mallintaa älykodin toimintoja tunnistuksesta ja paikannuksesta aina laitteiden ohjaamiseen ja päivittäisten rutiinien oppimiseen.

Kuva 8. Laboratorioon hahmoteltu älykotiympäristö, jossa jokainen väliseinällä erotettu huone sisältää paineenmuutoksia mittaavan lattian.

4. ASKELTUNNISTUKSEN MENETELMÄT

Luvussa esitetään askeltunnistukseen sovellettujen luokittelumenetelmien ja niihin läheisesti liittyvien tekniikoiden teoriaa. Kuten aikaisemmissa luvuissa on esitetty, askeltunnistus perustuu yksittäisten EMFi-signaalista segmentoitujen askelten mallintamiseen. Askelprofiili esittää lattiaan kohdistuvan paineenmuutoksen jännitteenä, jolloin mittaukset ja niistä johdetut piirteet saadaan kvantitatiivisinä muuttujina. Tilastolliseen luokitteluun ja neuroverkkohin perustuvat menetelmät toimivat yleensä hyvin tämänkaltaisten ilmiöiden kuvaamiseen. Koska tunnistettavien henkilöiden oikea henkilöllisyys tiedetään, voidaan ongelmaan soveltaa ohjatun oppmisen menetelmiä. Tässä työssä tunnistusmenetelmiksi valittiin LVQ sekä LVQ:n ja HMM-mallien yhdistelmä.

LVQ tarjoaa yksinkertaisen prototyyppivektoreihin perustuvan tilastollisen menetelmän, jota on käytetty menestyksekkäästi monissa luokitteluongelmissa. Yksittäiset askelprofiilit voidaan esittää piirrevektoreina, joita LVQ käyttää suoraan koodikirjan opettamiseen ja luokitteluun. Lisäksi LVQ on malliriippumaton eli tunnistettavien luokkien jakaumista ei tarvita etukäteistietoa, vaan ne approksimoidaan suoraan opetusdatasta. Toiseksi menetelmäksi valittiin hieman monimutkaisempi HMM-luokittelu, joka on myös LVQ:n tavoin malliriippumaton. Nyt tunnistettava kohde esitetään kuitenkin piirresekvenssinä ja luokittelu perustuu suurimman uskottavuuden eli todennäköisimmän mallin mukaan. HMM-malleilla on kuvattu menestyksekkäästi aikariippuvia signaaleja, jollainen askelsignaalikin on. Tässä työssä HMM-mallit ja LVQ-koodikirja yhdistettiin, jolloin HMM-malleille tulevat piirrevektorisekvenssit saatiin kvantisoitua ja ne voitiin mallintaa äärellisellä määrällä kokonaislukusymboleita.

Menetelmien valintaan vaikuttivat lisäksi lähteissä [9] ja [8] esitetyt menetelmät. Ensimmäisessä GRF-profiilin tunnistamiseen käytettiin menestyksekkäästi HMM-malleja. Piirrevektorisekvenssien kvantisointiin käytettiin kuitenkin VQ:a, joka on ohjaamattoman opetuksen menetelmä. Koska tunnistettavat luokat tunnetaan entuudestaan, tässä työssä VQ-menetelmä vaihdettiin LVQ-menetelmäksi, jolloin ohjatun opetuksen hyötyä voitiin käyttää hyväksi piirresekvenssien kvantisoinnissa. Toisessä lähteessä vastaavasti GRF-profiilin mallintamiseen käytettiin yksinkertaista NN-menetelmää, jolloin yksittäinen askel esitetään yhtenä piirrevektorina ja tunnistukseen käytetään suoraan opetusnäytteiden etäisyyksiä piirreavaruudessa. Koska myös tämä menetelmä antoi luotettavia tunnistustuloksia, tässä työssä päätettiin käyttää LVQ-luokitinta myös itsenäisesti, sillä sen toiminta on todella lähellä lähimmän naapurin menetelmää. Lisäksi LVQ tarjoaa mahdollisuuden erilaisiin laajennuksiin, kuten automaattiseen piirrevalintaan sekä hylkäyskriteerin käyttöön. Nämä laajennukset esitetään luvuissa 5 ja 6.

Valittujen luokittelumenetelmien lisäksi tässä luvussa esitetään muutamia niihin läheisesti liittyviä menetelmiä. Ohjaamattoman opetuksen menetelmät vektorikvantisointi ja itseorganisoituvat kartat ovat olleet perustana LVQ kehitykseen. Myöskin k-lähimmän naapurin menetelmä liittyy läheisesti LVQ:in ja sitä voidaan käyttää myös opetetun LVQ-koodikirjan avulla tapahtuvaan luokitteluun. Lisäksi KNN-menetelmää käytetään tässä työssä osapiirrejoukkojen testaamiseen (ks. kohta 7.3.2).

4.1. Vektorikvantisointi

Vektorikvantisoijaa voidaan pitää systeeminä, joka kuvaa joukon jatkuvia tai diskreettejä tulovektoreita tallennusta ja tiedonsiirtoa hyödyttävään muotoon. Systeemin tarkoituksena on vähentää tarvittavan tiedon määrää. Vektorikvantisoija sisältää enkooderin, joka suorittaa sisääntulevien vektoreiden kvantisoinnin sekä dekooderin, joka kuvaa kvantisoidut arvot takaisin lähtövektoreiksi aiheuttaen aina signaaliin häiriötä. [55]

Luokittelun kannalta vektorikvantisointia voidaan pitää ohjaamattoman opetuksen menetelmänä, jonka tarkoituksena on pyrkiä approksimoimaan tulovektoreiden todennäköisyystiheysfunktioita äärellisellä määrällä koodikirjavektoreita. Kun koodikirja on valittu, tulovektori kvantisoidaan etsimällä koodikirjavektori $\mathbf{m_c}$, joka on lähin tulovektoriin $\mathbf{x}$ nähden [56, s. 59–64]. Tämä voidaan kuvata yhtälöllä

$$||\mathbf{x} - \mathbf{m}_c|| = min_i\{||\mathbf{x} - \mathbf{m}_i||\}, \tag{1}$$

jossa minimietäisyys määritetään käyttäen jotain etäisyysmittaa. Yleisin käytössä oleva etäisyysmitta on niin sanottu euklidinen etäisyys, joka voidaan esittää muodossa

$$d_e(\mathbf{x}, \mathbf{y}) = ||\mathbf{x} - \mathbf{y}|| = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}, \tag{2}$$

jossa d_e kuvaa kahden vektorin $\mathbf{x}$ ja $\mathbf{y}$ euklidista etäisyyttä. Vektorin pituus on n.

4.2. Itseorganisoituvat kartat

Itseorganisoituvat kartat (SOM) on tunnettu ohjaamattomaan opetukseen perustuva ryhmittely ja visualisointi menetelmä. Sen tarkoituksena on kuvata sisääntulevat moniulotteiset piirrevektorit 2-ulotteiseksi malliksi, joka mallintaa opetusdataa tietämättään haluttuja vasteita. Menetelmän toiminta perustuu topologisesti järjestettyyn neuroverkkomalliin, jossa sisääntulevan piirrevektorin avulla päivitetään lähimmän neuronin ja sen naapuruston painokertoimia opetuskertoimien avulla. Opetuksen seurauksena kartta oppii reagoimaan samalla tavalla samankaltaisille syötteille. [56]

4.3. K-lähimmän naapurin menetelmä

K-lähimmän naapurin (KNN) menetelmä [57, s. 174–188] on yksinkertainen ja tehokas luokittelumenetelmä, jonka avulla voidaan määrittää malliriippumattomia kompleksisia luokkarajoja. Menetelmä perustuu kaikkien opetusnäytteiden tallentamiseen, ja tästä johtuen sitä nimitetään muistiin sekä esimerkkeihin perustuvaksi oppimiseksi [58, luku 8.]. Tuntematon näyte luokitellaan etsimällä

euklidista etäisyyttä käyttäen (ks. yhtälö 2) k lähintä naapuria opetusdatasta, ja valitsemalla luokka enemmistön mukaan.

4.4. Oppiva vektorikvantisointi

Oppiva vektorikvantisointi (LVQ) on vektorikvantisointiin ja itseorganisoituviin karttoihin perustuva tilastollinen luokittelu- ja tunnistusmenetelmä. Lisäksi LVQ on toiminnaltaan ja luokitteluominaisuuksiltaan lähellä KNN-luokitinta. Oppivan vektorikvantisoinnin käsittely tässä kappaleessa perustuu pääosin lähteeseen [56], ellei toisin mainita.

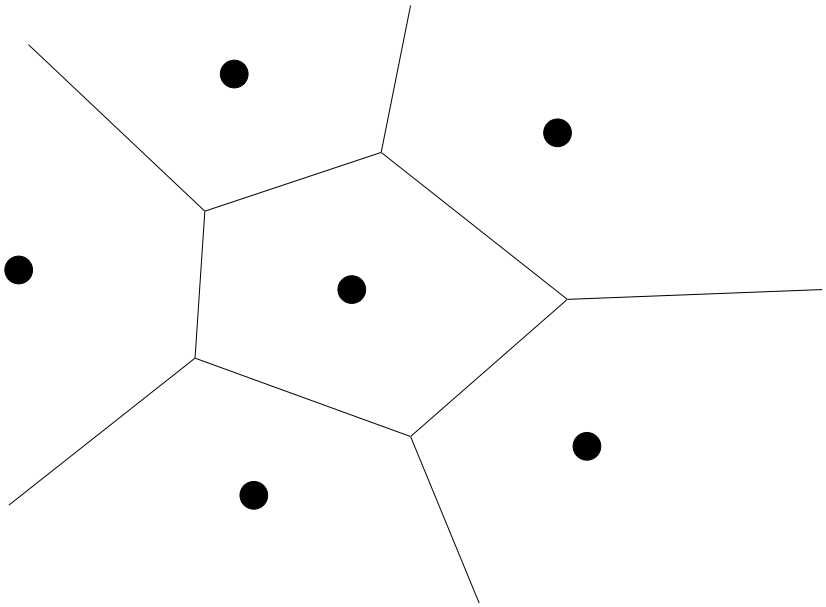
LVQ:ta on käytetty esimerkiksi tekstuurien analyysissä [59], puheentunnistuksessa [60] ja kuvankäsittelyssä [61]. Toisin kuin itseorganisoituvat kartat ja vektorikvantisointi, se perustuu ohjattuun oppimiseen, jossa opetetusnäyteiden luokat tunnetaan entuudestaan. Oppivan vektorikvantisoinnin tarkoituksena on määrittää tunnistettavien luokkien rajat piirreavaruudessa ja luokitella tuntemattomat näytevektorit lähimmän koodikirjavektorin mukaan. Tämän luokittelumenetelmän perusperiaate on, että piirreavaruutta kuvataan tietyllä määrällä kuhunkin luokkaan kuuluvia koodikirjavektoreita, joiden paikkoja siirretään opetusaineiston mukaan. Opetuksen aikana piirreavaruuteen muodostuu paloittain jatkuvat lineaariset luokkarajat, joita voidaan moniulotteisessa tapauksissa nimittää hypertasoiksi. Kuvassa 9 on esitetty 2-ulotteisessa tapauksessa koodikirja vektorien suhteen syntyvät päätöspinnat, kun luokitteluun käytetään lähimmän naapurin menetelmää. Luokkien päätösalueita kutsutaan myös Voronoi-soluiksi.

Oppivaa vektorikvantisointia voidaan pitää KNN-luokitinta kompressoivana menetelmänä, sillä LVQ:ssa luokat esitetään yleensä opetusnäytteitä pienemmällä määrällä prototyyppivektoreita. Menetelmät antavat yleensä hyvin samankaltaisia luokittelutuloksia [62], kuitenkin LVQ voi tietyissä tapauksissa tarjota vähemmän kompleksiset luokkarajat, jolloin luokkajakaumat tasoittuvat ja luokittimen yleistyskyky paranee. Tiedon pakkauksen johdosta LVQ vähentää myös muistin tarvetta ja laskenta-aikaa itse luokittelussa verrattuna KNN-menetelmään [62, s. 216]. Toisaalta KNN-menetelmä ei vaadi ollenkaan opetusaikaa, joten luokitin on heti valmis käytettäväksi.

4.4.1. Koodikirjan alustaminen

Oppivassa vektorikvantisoinnissa koodikirja täytyy alustaa ennen opetusta. Vaikka luokkien jakaumat voivat olla hyvin erilaisia, on käytännön sovelluksissa hyvä aloittaa samalla määrällä koodikirjavektoreita luokkaa kohden. Koska luokat on kuvattu äärellisellä määrällä prototyyppivektoreita, alustuksessa on mahdollista pyrkiä tasoittamaan luokkien välisiä eroja. Tämä tapahtuu laskemalla kunkin luokan alustettujen koodikirjavektoreiden lyhimmän etäisyyden mediaani. Jos jonkin luokan etäisyys eroaa muista, voidaan luokkien välisiä eroja tasoittaa lisäämällä tai poistamalla kyseessä olevan luokan koodikirjavektoreita.

Koodikirjavektorit voidaan alustaa käyttäen opetusdatasta oikein luokiteltuja näytteitä. Esimerkiksi käyttäen KNN-menetelmää voidaan määrittää yksittäiselle näytteelle luokka muiden näytteiden suhteen. Jos luokittelu on oikein, hyväksytään näyte prototyyppivektorin alkuarvoksi. Itse LVQ-koodikirjan opetuksessa



Kuva 9. LVQ:n koodikirjavektorien väliset päätösrajat 2-ulotteisessa piirreavaruudessa, kun käytetään lähimmän naapurin luokittelumenetelmää.

täytyy kuitenkin käyttää kaikkia opetusnäytteitä itsenäisesti, huolimatta siitä, ovatko ne oikein tai väärin luokiteltuja. Koodikirjan alustamiseen voidaan käyttää myös ohjaamattoman opetuksen menetelmiä: k-means ja SOM. Esimerkikisi SOM voi tarjota paremman alustuksen, jos luokkien jakaumat ovat monihuippuisia.

4.4.2. Koodikirjan opettaminen

Koodikirjan alustuksen jälkeen sitä pyritään opettamaan ennalta määrätyillä opetusvektoreilla. Jokaiselle tunnistettavalle luokalle on alustuksessa määrätty äärellinen määrä koodikirjavektoreita, joita siirretään opetusaineiston mukaan pyrkien minimoimaan luokitteluvirhe eli approksimoimaan optimaaliset Bayesin luokkarajat. Opettamisen jälkeen tuntematon piirrevektori $\mathbf{x}$ luokitellaan kuuluvaksi samaan luokkaan, johon lähin koodikirjavektori $\mathbf{m}_i$ kuuluu. Voidaan määrittää

$$c = \underset{i}{\operatorname{argmin}}\{||\mathbf{x} - \mathbf{m}_i||\}, \tag{3}$$

jossa c on lähimmän koodikirjavektorin $\mathbf{m}_i$ indeksi näytteen $\mathbf{x}$ suhteen.

Itse luokitteluun voidaan käyttää myös KNN-menetelmää, jolloin luokiteltavalle näytteelle haetaan k lähintä koodikirjavektoria opetetusta koodikirjasta ja luokittelu suoritetaan näiden prototyyppien luokkien enemmistön mukaan. Tämä voi tulla kysymykseen, jos luokittelussa käytetään suurta määrää prototyyppivektoreita luokkaa kohden. Seuraavaksi esitellään erilaiset LVQ-koodikirjan opettamiseen käytettävät opetusalgoritmit.

4.4.3. LVQ1

LVQ1:ssä $\mathbf{x}(t)$ kuvaa sisääntulevaa piirrevektoria ja $\mathbf{m}_c(t)$ sitä vastaavaa lähintä koodikirjavektoria ajan hetkellä t. Opetuksessa lähimmän koodikirjavektorin paikkaa korjataan piirreavaruudessa sisääntulevan näytteen mukaisesti seuraavasti:

$$\mathbf{m}_c(t+1) = \mathbf{m}_c(t) + \alpha(t)[\mathbf{x}(t) - \mathbf{m}_c(t)] \quad (4)$$
$$\mathbf{m}_c(t+1) = \mathbf{m}_c(t) - \alpha(t)[\mathbf{x}(t) - \mathbf{m}_c(t)] \quad (5)$$
$$\mathbf{m}_i(t+1) = \mathbf{m}_i(t), \text{ jos } i \neq c, \quad (6)$$

jossa $\alpha(t)$ on opetuskerroin ($0 < \alpha(t) < 1$), ja sitä pienennetään yleensä lineaarisesti opetuskierrosten kasvaessa Opetuskerroin alustetaan melko pieneksi ($\alpha < 0,1$). Yhtälöä (4) käytetään, kun $\mathbf{x}$ ja $\mathbf{m}_c$ kuuluvat samaan luokkaan ja yhtälöä (5), kun $\mathbf{x}$ ja $\mathbf{m}_c$ kuuluvat eri luokkiin. Muita koodikirjavektoreita ei päivitetä kyseisellä opetuskierroksella (yht. (6)). LVQ1-algoritmin tarkoituksena on siis minimoida luokitteluvirhe iteratiivisesti päivittämällä koodikirjavektoreita diskreeteillä ajanhetkillä $t = 0, 1, 2, \ldots$.

4.4.4. OLVQ1

Optimoidun opetuskertoimen LVQ1-algoritmi on laajennettu versio LVQ1:stä. Tarkoituksena on saavuttaa nopeampi konvergoituminen. Tämä tapahtuu siten, että jokaiselle koodikirjavektorille $\mathbf{m}_i$ määritetään oma opetuskerroin $\alpha_i(t)$. Tällöin iteratiivinen opetus tapahtuu yhtälöiden (7), (8), (9) avulla. Lähimmän koodikirjavektorin indeksi c on määritetty edelleen yhtälön (3) mukaan.

$$\mathbf{m}_c(t+1) = \mathbf{m}_c(t) + \alpha_c(t)[\mathbf{x}(t) - \mathbf{m}_c(t)] \quad (7)$$
$$\mathbf{m}_c(t+1) = \mathbf{m}_c(t) - \alpha_c(t)[\mathbf{x}(t) - \mathbf{m}_c(t)] \quad (8)$$
$$\mathbf{m}_i(t+1) = \mathbf{m}_i(t), \text{ jos } i \neq c. \quad (9)$$

Yhtälöä (7) käytetään, jos $\mathbf{x}$ on luokiteltu oikein ja yhtälöä (8), jos $\mathbf{x}$ on luokiteltu väärin. Optimaalisen nopeaan konvergoitumiseen päästään esittämällä yhtälöt (7), (8), (9) muodossa

$$\mathbf{m}_c(t+1) = [1 - s(t)\alpha_c(t)]\mathbf{m}_c(t) + s(t)\alpha_c(t)\mathbf{x}(t), \quad (10)$$

jossa $s(t) = +1$, jos luokittelu on oikein ja $s(t) = -1$, jos luokittelu on väärin. Opetetun koodikirjavektorin tilastollinen tarkkuus on myös optimaalinen, jos eri ajan hetkillä tehtyjen korjausten vaikutukset ovat opetusjakson lopussa yhtenäisesti painotettuja. Koodikirjavektori $\mathbf{m_c}(t+1)$ sisältää tietoa $\mathbf{x}(t)$:stä yhtälön (10) viimeisen termin mukaan ja tietoa aikaisemmista $\mathbf{x}(t')$:stä ($t' = 1, 2, \ldots, t-1$), $\mathbf{m}_c(t)$:n mukaan. Viimeisin tieto $\mathbf{x}(t)$:stä on skaalattu kertoimella $\alpha_c(t)$, ja sitä edellinen $\mathbf{x}(t-1)$ on skaalattu kertoimella $[1 - s(t)\alpha_c(t)]\alpha_c(t-1)$. Kun nämä kaksi skaalauskerrointa määritetään yhtäsuuriksi, niin saadaan

$$\alpha_c(t) = [1 - s(t)\alpha_c(t)]\alpha_c(t-1). \tag{11}$$

Kun tämä määritys tehdään kaikilla ajanhetkillä t, induktiolla voidaan osoittaa, että kaikki ajanhetkeen t kerääntynyt tieto näytteistä $\mathbf{x}(t')$ voidaan skaalata lopussa yhtenevästi, ja opetuskertoimien $\alpha_i(t)$ optimaaliset arvot voidaan määrittää rekursiolla

$$\alpha_c(t) \;=\; \frac{\alpha_c(t-1)}{1 + s(t)\alpha_c(t-1)}. \tag{12}$$

4.4.5. LVQ2 (LVQ 2.1)

LVQ2-algoritmissa luokittelupäätös on vastaava kuin LVQ1:ssä. Opetuksessa otetaan kuitenkin huomioon kaksi koodikirjavektoria, joita päivitetään yhtäaikaa. Koodikirjavektorit valitaan siten, että ne ovat lähimpiä sisääntulevan piirrevektorin $\mathbf{x}$ suhteen, ja toinen niistä kuuluu näytteen kanssa samaan ja toinen eri luokkaan. Lisäksi $\mathbf{x}$:n pitää osua "ikkunaan", joka on näiden koodikirjavektoreiden $\mathbf{m}_i$ ja $\mathbf{m}_j$ välissä. Piirrevektorin $\mathbf{x}$ määritetään osuneeksi "ikkunaan", jos

$$min\Big(\frac{d_i}{d_j}, \frac{d_j}{d_i}\Big) > s, \quad jossa \quad s = \frac{1-w}{1+w}. \tag{13}$$

Etäisyydet d_i ja d_j ovat $\mathbf{m}_i$:n ja $\mathbf{m}_j$:n euklidisia etäisyyksiä näytevektorin $\mathbf{x}$ suhteen ja w on "ikkunan" leveys, jonka suuruudeksi suositellaan arvoa väliltä 0,2 - 0,3.

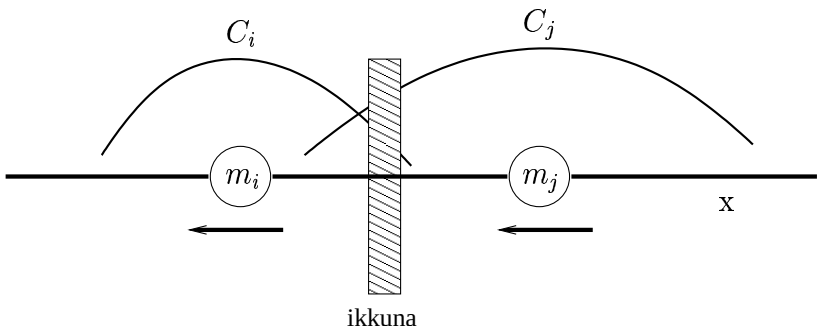
LVQ2.1 on parannettu versio LVQ2-algoritmista. Alkuperäisessä LVQ2:ssa koodikirjavektorin $\mathbf{m}_i$ täytyi olla lähin vektori sisääntulevaan näytteeseen. Nyt lähin koodikirjavektori voi olla joko $\mathbf{m}_i$ tai $\mathbf{m}_j$. LVQ2.1:n opetusalgoritmi on seuraava:

$$\mathbf{m}_i(t+1) \;=\; \mathbf{m}_i(t) - \alpha(t)[\mathbf{x}(t) - \mathbf{m}_i(t)], \tag{14}$$
$$\mathbf{m}_j(t+1) \;=\; \mathbf{m}_j(t) + \alpha(t)[\mathbf{x}(t) - \mathbf{m}_j(t)], \tag{15}$$

jossa $\mathbf{m}_i$ ja $\mathbf{m}_j$ ovat kaksi lähintä koodikirjavektoria näytteen $\mathbf{x}$ suhteen $\mathbf{x}$:n ja $\mathbf{m}_j$:n kuuluessa samaan luokkaan ja $\mathbf{x}$:n ja $\mathbf{m}_i$:n eri luokkaan. $\mathbf{x}$:n pitää lisäksi osua yhtälössä (13) esitettyyn "ikkunaan". Koodikirjavektorien päivitys on esitetty kuvassa 10, ja mukailee lähteessä [63] esitettyä.

4.4.6. LVQ3

LVQ2:n toiminta perustuu differentiaaliseen koodikirjavektorien siirtelyyn päätöspintojen rajoilla. Se ei kuitenkaan ota huomioon, mitä tapahtuu yksittäisen koodikirjavektorin $\mathbf{m}_i$ paikalle pitkällä aikavälillä opetuksen jatkuessa. LVQ3:ssa



Kuva 10. LVQ2:ssa kahta lähintä koodikirjavektoria m_j ja m_i, jotka kuuluvat samaan (C_j) ja eri luokkaan (C_i), päivitetään samanaikaisesti. Opetusnäytteen tulee osua koodikirjavektorien välissä olevaan ikkunaan.

opetukseen on lisätty korjaus, joka varmistaa, että $\mathbf{m}_i$ jatkaa luokkajakauman määrittämistä myös pitkällä aikavälillä. LVQ3:n opetusalgoritmit ovat seuraavat:

$$\mathbf{m}_i(t+1) = \mathbf{m}_i(t) - \alpha(t)[\mathbf{x}(t) - \mathbf{m}_i(t)], \quad (16)$$
$$\mathbf{m}_j(t+1) = \mathbf{m}_j(t) + \alpha(t)[\mathbf{x}(t) - \mathbf{m}_j(t)], \quad (17)$$

jossa $\mathbf{m}_j$ ja $\mathbf{m}_i$ ovat kaksi samaan ja eri luokkaan kuuluvaa lähintä koodikirjavektoria sisääntulevaan piirrevektorin mukaan. Näytevektorin $\mathbf{x}$ pitää osua yhtälössä (13) esitettyyn "ikkunaan". Tällöin koodikirjavektorien päivittämiseen käytetään yhtälöita 16 ja 17. Sitävastoin jos $\mathbf{x}$, $\mathbf{m}_i$ ja $\mathbf{m}_j$ kuuluvat samaan luokkaan, päivitetään koodikirjavektoreita seuraavasti:

$$\mathbf{m}_k(t+1) = \mathbf{m}_k(t) - \epsilon\alpha(t)[\mathbf{x}(t) - \mathbf{m}_k(t)], \quad (18)$$

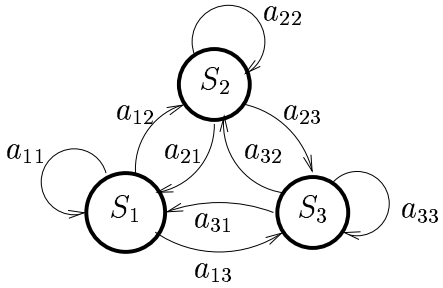
jossa $k \in \{i, j\}$ ja ϵ on korjauskerroin, ja sen arvo valitaan väliltä $0,1 - 0,5$.

4.5. Kätketyt Markov-mallit

Seuraavassa käsitellään diskreettien kätkettyjen Markov-mallien teoriaa. Kappaleen esitys perustuu pääosin lähteeseen [64], ellei toisin mainita.

4.5.1. Diskreetti Markov-prosessi

Diskreetti Markov-prosessi on N-tilainen tilakone, jossa jokaisella ajanhetkellä havaitaan yksi N:stä tilasta $S_1, S_2, \ldots, S_N$. Jokaiseen tilaan liittyy tietty tilansiirtotodennäköisyys a_{ij}, jolla havaittua tilaa vaihdetaan tai pysytään samassa seuraavana diskreettinä ajanhetkenä t. Kuvassa 11 on esitetty kolmitilainen Markov-prosessi, jossa diskreetillä ajanhetkellä vaihdetaan tilaa S tilansiirtotodennäköisyyksien a_{ij} mukaan.



Kuva 11. Kolmitilainen Markov-prosessi.

4.5.2. Kätketyt Markov-mallit ja niiden osat

Kätketyt Markov-mallit (HMM) voidaan ajatella Markov-prosessiksi, jossa itse prosessi on piilossa havainnoitsijalta. Tiedossa on vain joukko havaintoja, joilla on tietty todennäköisyysjakauma. Näin ne tarjoavat tilastollisen luokittelumenetelmän, jolla voidaan mallintaa luonnostaa aikariippuvia signaaleja. HMM-malleja on käytetty onnistuneesti puheentunnistuksessa [65]. Muita sovelluskohteita ovat olleet esimerkiksi käyttäjätilanteiden tunnistamisessa [66] sekä robotiikassa [67]. HMM-mallin rakentamiseen tarvitaan seuraavat 5 osaa:

1. N, mallissa olevien tilojen määrä
   Vaikka tilat ovat kätkettyjä, käytännön sovelluksissa usein tilojen lukumäärällä on jokin fysikaalinen merkitys mallin kannalta.

2. M, tilakohtainen havaintojen lukumäärä
   Havaitut symbolit **O** kuvaavat mallinnettavan systeemin ulostuloa.

3. Tilansiirtotodennäköisyydet **A**

$$\mathbf{A} = \{a_{ij}\}, \tag{19}$$

   jossa $a_{ij} = P[q_{t+1} = S_j | q_t = S_i], \qquad 1 \leq i, \;\; j \leq N$

4. Havaittavien symbolien todennäköisyydet **B** tilassa j, $\mathbf{B} = \{b_j(k)\}$, jossa

$$b_j(k) = P[v_k, t | q_t = S_j], \qquad \begin{array}{l} 1 \leq j \leq N, \\ 1 \leq k \leq M. \end{array} \tag{20}$$

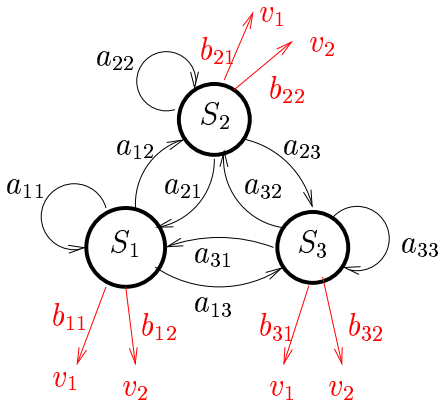
5. Alkutilan todennäköisyys $\pi = \{\pi_i\}$, jossa

$$\pi_i = P[q_1 = S_i], \qquad 1 \leq i \leq N. \tag{21}$$

Yllä olevan mukaan HMM-mallin rakentamiseen tarvitaan siis kaksi malliparametria (N ja M), havaintosymbolit **O**, sekä kolme todennäköisyysmittaa **A**, **B**, π, jolloin se voidaan esittää muodossa

$$\lambda = (\mathbf{A}, \mathbf{B}, \pi). \tag{22}$$

Kuvassa 12 on esitetty kolmitilainen HMM-malli, jossa jokainen tila generoi diskreetit tilahavainnot v_1 ja v_2, tilansiirto- ja tilahavaintotodennäköisyyksien (a_{ij} ja b_{ik}) avulla.



Kuva 12. Kolmitilainen kätketty Markov-malli.

4.5.3. HMM-mallin ongelmat

HMM-malleihin liittyy kolme perusongelmaa, jotka pitää ratkaista ennen kuin niitä voidaan soveltaa. Nämä ongelmat ovat seuraavat:

- Ongelma 1: On annettu havaintosekvenssi $\mathbf{O} = O_1 O_2 \ldots O_T$, ja malli $\lambda = (\mathbf{A}, \mathbf{B}, \pi)$, miten voidaan tehokkaasti laskea ehdollinen todennäköisyys $P(\mathbf{O}|\lambda)$?

  Ongelma 1 voidaan ajatella tapaukseksi, jossa täytyy tietää miten hyvin annettu malli vastaa havaintoja. Esimerkiksi luokittelussa tuntematonta havaintosekvenssiä verrataan ennalta opetettuihin malleihin ja luokitellaan näytesekvenssi sitä parhaiten kuvaavan mallin mukaan laskemalla ehdolliset todennäköisyydet tuntemattoman havainnon ja mallin mukaan.

- Ongelma 2: On annettu havainto sekvenssi $\mathbf{O} = O_1 O_2 ... O_T$, ja malli $\lambda = (\mathbf{A}, \mathbf{B}, \pi)$, miten voidaan määrittää tilasekvenssi, joka kuvaa parhaiten havainnot?

  Ongelman 2 tapauksessa halutaan tietää, mikä on havaintoja vastaan paras reitti HMM-mallin sisällä. Koska HMM-mallin tilat ovat piilossa, ei tarkkaa reittiä voida määrittää, mutta sen sijaan etsitään todennäköisin tilasekvenssi. Tätä voidaan käyttää esimerkiksi jatkuvassa puheentunnistuksessa tai jos halutaan tietää yksittäisten tilojen keskimääräisiä tilastotietoja.

- Ongelma 3: Kuinka mallin $\lambda = (\mathbf{A}, \mathbf{B}, \pi)$ parametrejä tulisi muuttaa $P(\mathbf{O}|\lambda)$:n maksimoimiseksi?

  Ongelma 3 liittyy HMM-mallien opetusvaiheeseen. Tarkoituksena on pyrkiä muuttamaan ennalta määrättyjen havaintosekvessien avulla mallin parametrejä siten, että ne pystyisivät kuvaamaan nämä sekvenssit parhaalla mahdollisella tarkkuudella. Luokittelussa jokaiselle luokalle on määrätty malli, jonka parametrejä opetetaan ja parannetaan tähän luokkaan kuuluvilla opetussekvensseillä.

4.5.4. Havaintosekvenssin todennäköisyys

Ongelman 1 ratkaisu voidaan toteuttaa eteenpäin-taaksepäin-algoritmilla (forward-backward procedure).

Eteenpäin-muuttuja $\alpha_t(i)$ on määritelty seuraavasti:

$$\alpha_t(i) = P(O_1 O_2 \ldots O_t, q_t = S_i|\lambda), \tag{23}$$

jossa lasketaan yhteistodennäköisyys havaintosekvenssin osalle $O_1 O_2 \ldots O_t$ (ajanhetkeen t saakka) ja tilalle S_i ajanhetkellä t, kun on annettu malli λ. Eteenpäin-muuttuja $\alpha_t(i)$ voidaan ratkaista induktiivisesti:

1. Alustus:

$$\alpha_1(i) = \pi_i b_i(O_1), \qquad 1 \leq i \leq N. \tag{24}$$

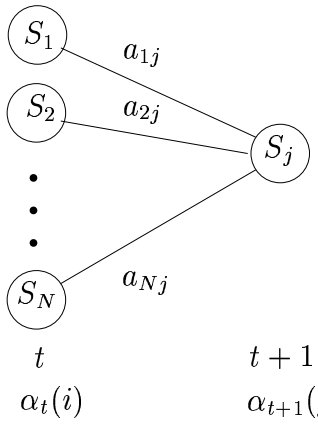
2. Induktio:

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(O_{t+1}), \qquad 1 \leq t \leq T-1, \quad 1 \leq j \leq N. \tag{25}$$

3. Lopetus:

$$P(\mathbf{O}|\lambda) = \sum_{i=1}^{N} \alpha_T(i). \tag{26}$$

Kohdassa 1. eteenpäin-muuttuja $\alpha_t(i)$ alustetaan tilan S_i ja alkuhavainnon O_1 yhteistodennäköisyyksien mukaan. Kohdassa 2. lasketaan tilatodennäköisyydet edellisen ajanhetken tilahavaintotodennäköisyyksien $\mathbf{B}$, tilasiirtymätodennäköisyyksien $\mathbf{A}$ ja havainnon O mukaan. Kohdassa 3. lasketaan lopuksi kokonaistodennäköisyys kaikkien tilojen todennäköisyyksien summana. Kuvassa 13 on esitetty eteenpäin-algoritmin induktiovaiheen laskenta muuttujan $\alpha_{t+1}(j)$ suhteen.



Kuva 13. Eteenpäin-algoritmin laskenta muuttujalle $\alpha_{t+1}(j)$.

Taaksepäin-algoritmissa muuttuja $\beta_t(i)$ määritetään vastaavasti kuin eteenpäin-muuttuja $\alpha_t(i)$, mutta nyt havaintojoukossa liikutaan lopusta alkuun. Taaksepäin-algoritmia ei tarvita ongelman 1 ratkaisuun, mutta sitä käytetään ongelman 3 ratkaisussa myöhemmin.

Taaksepäin-muuttuja on määritelty seuraavasti:

$$\beta_t(i) = P(O_{t+1}O_{t+2}\ldots O_T|q_t = S_i, \lambda), \tag{27}$$

jossa lasketaan yhteistodennäköisyys havaintosekvenssin osalle $t+1$:stä sekvenssin loppuun, ja tilalle S_i ajanhetkellä t, kun on annettu malli λ. Taaksepäin-muuttuja $\beta_t(i)$ voidaan myös ratkaista induktiivisesti:

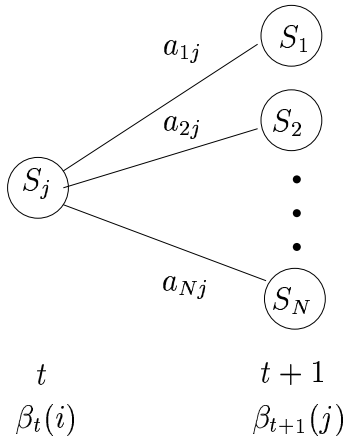
1. Alustus:

$$\beta_T(i) = 1, \qquad 1 \leq i \leq N. \tag{28}$$

2. Induktio:

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j), \qquad t = T-1, T-2, \ldots, 1,$$
$$1 \leq i \leq N. \tag{29}$$

Kuvassa 14 on esitetty taaksepäin-algoritmin induktiovaiheen laskenta muuttujan β_{t+1} suhteen.



Kuva 14. Taaksepäin-algoritmin laskenta muuttujalle $\beta_{t+1}(j)$.

4.5.5. Todennäköisimmät tilat

Ongelman 2 ratkaisuun voidaan käyttää Viterbi-algoritmia [68], jonka avulla etsitään paras yksittäinen havaintoja $\mathbf{O} = \{O_1O_2\ldots O_T\}$ kuvaava tilasekvenssi $\mathbf{Q} = \{q_1q_2\ldots q_T\}$. Määrätään suure

$$\delta_t(i) = \max_{q_1, q_2, \cdots, q_{t-1}} P[q_1q_2\ldots q_t = i, O_1O_2\ldots O_t|\lambda], \tag{30}$$

jossa $\delta_t(i)$ on yksittäisen reitin suurin todennäköisyys ajanhetkellä t ottaen huomioon havainnot $1, 2, \ldots, t$ ja päättyen tilaan S_i. Induktiolla saadaan

$$\delta_{t+1}(j) = [\max_i \delta_t(i)a_{ij}]b_j(O_{t+1}). \tag{31}$$

Tilasekvenssin laskemiseksi tallennetaan $\delta_{t+1}(j)$:n maksimoiva tilasiirtymä. Tähän käytetään taulukkoa $\psi_t(j)$. Parhaan tilasekvenssin laskeminen voidaan toteuttaa seuraavasti:

1. Alustus:

$$\begin{aligned} \delta_1(i) &= \pi_i b_i(O_1), \qquad 1 \leq i \leq N & (32)\\ \psi_1(i) &= 0. & (33) \end{aligned}$$

2. Rekursio:

$$\begin{aligned} \delta_t(j) &= \max_{1\leq i\leq N}[\delta_{t-1}(i)a_{ij}]b_j(O_t), \qquad 2 \leq t \leq T \\ &\qquad\qquad 1 \leq j \leq N. & (34)\\ \psi_t(j) &= \operatorname*{argmax}_{1\leq i\leq N}[\delta_{t-1}(i)a_{ij}], \qquad 2 \leq t \leq T \\ &\qquad\qquad 1 \leq j \leq N. & (35) \end{aligned}$$

3. Lopetus:

$$\begin{aligned} P^* &= \max_{1\leq i\leq N}[\delta_T(i)] & (36)\\ q_T^* &= \operatorname*{argmax}_{1\leq i\leq N}[\delta_T(i)]. & (37) \end{aligned}$$

4. Polun etsiminen taaksepäin:

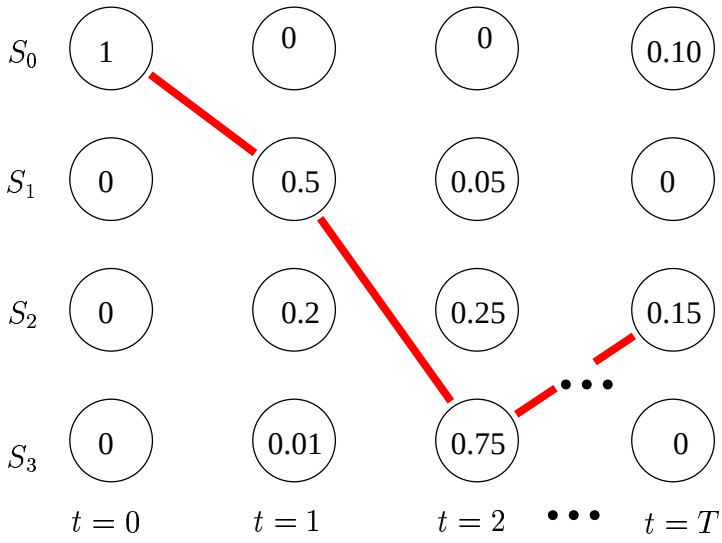
$$q_t^* = \psi_{t+1}(q_{t+1}^*), \qquad t = T-1, T-2, \ldots, 1. \tag{38}$$

Viterbi-algoritmin toteutus on kohtaa 4. lukuunottamatta vastaava kuin eteenpäin-algoritmissa. Nyt kuitenkin summan sijaan lasketaan edellisten tilojen todennäköisyyksien ja tilasiirtymätodennäköisyyksien tulojen maksimi sekä tallennetaan tilasiirtymä optimaalisen reitin etsimistä varten. Kuvassa 15 on esitetty optimaalisen reitin määritys hilarakenteen avulla.

4.5.6. Mallin parantaminen opettamalla

Kolmannen ongelman ratkaisuun voidaan käyttää Baum-Welch-algoritmia [69] (tai vastaavasti EM-algoritmia [70]). Se on iteratiivinen menetelmä, jonka avulla voidaan parantaa mallia, jotta se kuvaisi annettuja havaintoja parhaalla mahdollisella tavalla. Ensiksi määritetään $\xi_t(i,j)$, joka on todennäköisyys sille, että ollaan tilassa S_i ajanhetkellä t, ja tilassa S_j ajanhetkellä $t+1$. Tämä todennäköisyys määritetään annetun mallin ja havaintosekvenssin avulla seuraavasti:

$$\xi_t(i,j) = P(q_t = S_i, q_{t+1} = S_j | O, \lambda). \tag{39}$$

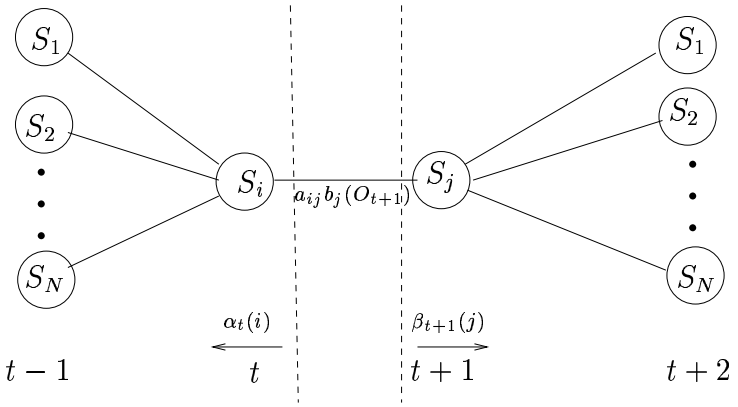


Kuva 15. Viterbi-laskenta voidaan toteuttaa hilarakenteella. Optimaalinen reitti löydetään valitsemalla maksimitodennäköisyys joka ajanhetkellä t.

Kuvassa 16 on esitetty $\xi_t(i,j)$ laskenta. Eteenpäin- ja taaksepäin-muuttujan määritelmien mukaan voidaan $\xi_t(i,j)$ kirjoittaa muotoon

$$\xi_t(i,j) = \frac{\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)}{P(\mathbf{O}|\lambda)} = \frac{\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)}, \tag{40}$$

josta haluttu todennäköisyysmitta saadaan.



Kuva 16. Yhteistapahtuman $\xi_t(i,j)$ laskenta tilassa S_i ajanhetkellä t ja tilassa S_j ajanhetkellä $t+1$.

Todennäköisyyksien laskemiseksi määritetään vielä muuttuja $\gamma_t(i)$, jonka avulla saadaan todennäköisyys sille, että ollaan tilassa S_i ajanhetkellä t, annettujen havaintojen ja mallin mukaan

$$\gamma_t(i) = \sum_{j=1}^{N}\xi_t(i,j). \tag{41}$$

Kun $\gamma_t(i)$ summataan koko aikavälin t yli, saadaan odotusarvo tilasta S_i tehtyjen siirtojen lukumäärälle. Vastaavasti, kun summataan $\xi_t(i,j)$ koko aikavälin yli, saadaan odotusarvo tehtyjen siirtojen lukumäärälle tilasta S_i tilaan S_j.

$$\sum_{t=1}^{T-1} \gamma_t(i) = \text{Odotusarvo tilasta } S_i \text{ tehdyille tilansiirtojen lukumäärälle.} \quad (42)$$

$$\sum_{t=1}^{T-1} \xi_t(i,j) = \quad \text{Odotusarvo tilasta } S_i \text{ tilaan } S_j \text{ tehdyille tilansiirtojen lukumäärälle.} \quad (43)$$

Yhtälöiden (42) ja (43) avulla voidaan määrätä HMM-mallin $\lambda = (A, B, \pi)$ parametrien päivitysyhtälöt, jotka on esitetty seuraavassa:

$$\bar{\pi}_i = \text{Odotusarvo tilan } S_i \text{ esiintymistaajuudelle ajanhetkellä (t=1)} = \gamma_1(i). \quad (44)$$

$$\begin{aligned} \bar{a}_{ij} &= \frac{\text{Odotusarvo tilasta } S_i \text{ tilaan } S_j \text{ tehdyille tilansiirtojen lukumäärälle}}{\text{Odotusarvo tilasta } S_i \text{ tehdyille tilansiirtojen lukumäärälle}} \\ &= \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}. \end{aligned} \quad (45)$$

$$\begin{aligned} \bar{b}_j(k) &= \frac{\text{Odotusarvo tilassa } S_i \text{ oltujen kertojen lukumäärälle havaitessa } v_k}{\text{Odotusarvo tilassa } S_j \text{ oltujen kertojen lukumäärälle}} \\ &= \frac{\sum_{\substack{t=1 \\ O_t = v_k}}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}. \end{aligned} \quad (46)$$

Yhtälöiden (44), (45) ja (46) avulla voidaan laskea uusi malli $\bar{\lambda} = (\bar{\mathbf{A}}, \bar{\mathbf{B}}, \bar{\pi})$, jonka Baum on todistanut olevan joko (1) $\bar{\lambda} = \lambda$, jolloin alkuperäinen malli λ kuvaa Maximum Likelihood -funktion (ML) maksimia tai (2) on löydetty uusi malli $\bar{\lambda}$, joka kuvaa paremmin annetun havaintosekvenssin.

Yllä määrätyn proseduurin avulla voidaan alustuksen jälkeen määrätä ML-funktion maksimi korvaamalla aina vanha malli λ uudella mallilla $\bar{\lambda}$. Näin iteratiivisesti jatketaan opetusta, kunnes se ei enää paranna mallin todennäköisyyttä. Vaikka proseduurin käyttämä eteenpäin-taaksepäin-algoritmi on monotonisesti kasvava, se voi johtaa vain paikalliseen maksimiin. Näin ollen mallin alustus on tärkeä osa opetusta.

Parametrien alustus voidaan toteuttaa esimerkiksi siten, että ennalta määrättyjen opetussekvenssien avulla estimoidaan tilajakaumat ja tilansiirtotodennäköisyydet. Tämä tapahtuu siten, että opetusdata segmentoidaan yhdenmukaisesti ja lasketaan jokaiselle segmentille yksittäisten havaintosymbolien lukumäärä. Sen jälkeen nämä lukumäärät normalisoidaan ja niiden avulla määritetään mallin tilajakaumat jokaiselle tilalle. Sitten käytetään edellä esitettyä Viterbi-algoritmia uudelleen segmentointiin ja parametrien uudelleen laskemiseen etsimällä mallista opetushavainto parhaiten vastaava tilasekvenssi. Viterbi-laskennan avulla saadaan myös mallin hyvyyttä kuvaaava log-likelihood-todennäköisyys. Tätä jatketaan iteratiivisesti, kunnes tietty määrä iteraatioita on tehty tai mallin log-likelihood-todennäköisyys konvergoituu. Tilansiirtotodennäköisyydet saadaan jokaisella iteraatiokierroksella laskemalla lukumäärät tapahtuneille siirroille Viterbi-reitissä ja normalisoinnissa. [71]

4.5.7. HMM-mallien tyypit

HMM-mallit voivat olla tilansiirtomatriisin suhteen erilaisia. Kun HMM-mallissa on mahdollisuus tehdä tilasiirtymiä kaikkien mahdollisten tilojen välillä, puhutaan ergodisesta tai täysin kytketystä mallista, ja vielä tarkemmin jokainen tila voidaan saavuttaa toisista tiloista äärellisellä määrällä askelia. Ergodisella 4-tilaisella HMM-mallilla tilansiirtomatriisi $\mathbf{A}$ on muotoa

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix}, \tag{47}$$

jossa alaindekseillä on esitetty siirtymän tilat. Esimerkiksi a_{12} on todennäköisyys sille, että siirrytään tilasta 1 tilaan 2.

Ergodisesta mallista päästään muun tyyppisiin malleihin asettamalla tilansiirtomatriisin tietyt siirtymät nolliksi. Usein käytetty malli on niin sanottu vasemmalta-oikealle-malli (L-R), jonka ominaisuutena tilaindeksi kasvaa tai pysyy samassa ajan kasvaessa. Tällainen malli soveltuu hyvin kuvaamaan ilmiöitä, joissa ominaisuudet muuttuvat ajan suhteen. Tässä työssä käytetään L-R-tyyppistä HMM-mallia, sillä askeleen ominaisuuksien voidaan ajatella muuttuvan juuri ajan suhteen. L-R-mallilla on ominaisuus

$$a_{ij} = 0, \qquad j < i \tag{48}$$

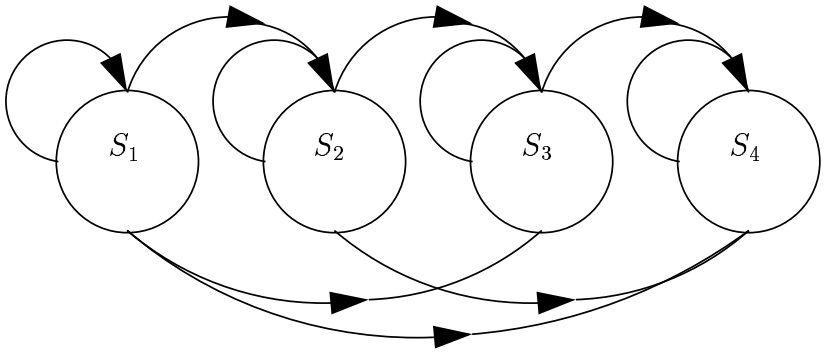
eli tilasiirtymiä ei sallita niihin tiloihin, joiden indeksit ovat pienempiä kuin nykyisen tilan indeksi. Lisäksi alkutilatodennäköisyys esitetään muodossa

$$\pi_i = \begin{cases} 0, & i \neq 1 \\ 1, & i = 1, \end{cases} \tag{49}$$

jossa tilasekvenssin pitää alkaa tilasta 1 (ja päättyä tilaan N). Kun edellä esitetyt ominaisuudet otetaan käyttöön saadaan 4-tilaisen L-R-mallin tilansiirtomatriisi $\mathbf{A}_{L-R}$ muotoon

$$\mathbf{A}_{L-R} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & 0 \\ 0 & a_{22} & a_{23} & a_{24} \\ 0 & 0 & a_{33} & a_{34} \\ 0 & 0 & 0 & a_{44} \end{bmatrix}. \tag{50}$$

Vasemmalta-oikealle-mallin viimeisen tilan todennäköisyys sille, että pysytään viimeisessä tilassa on aina yksi, ja vastaavasti 0 muille tilansiirtymille. Kuvassa 17 on esitetty 4-tilainen L-R-malli. Ergodisen ja L-R-mallin lisäksi niiden väliltä on olemassa erityyppisiä muunnelmia ja yhdistelmiä [64].



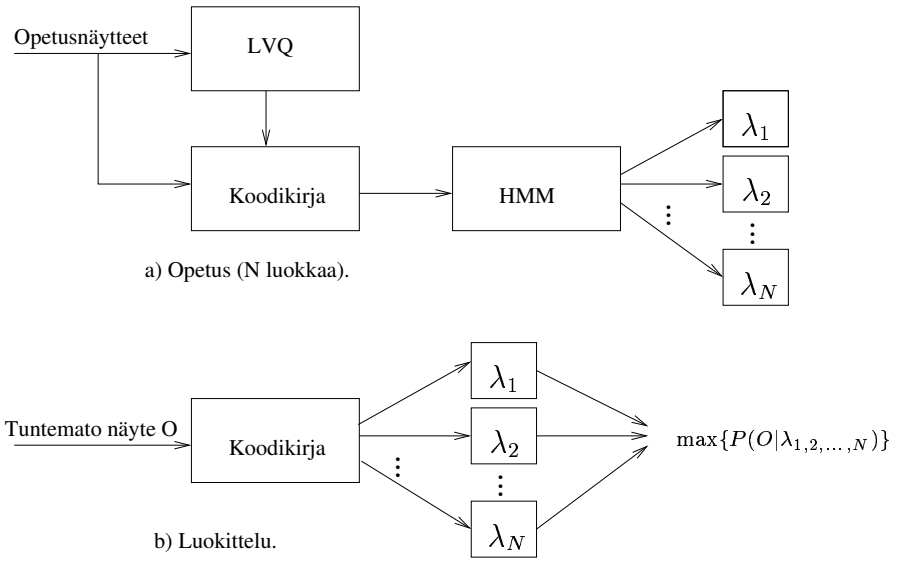
Kuva 17. Nelitilainen vasemmalta-oikealle HMM-malli.

4.6. LVQ-HMM-yhdistelmä

Tässä työssä käytetään menetelmää, joka yhdistää oppivan vektorikvantisoinnin ja diskreetit kätketyt Markov-mallit. Menetelmää on sovellettu menestyksekkäästi puheentunnistukseen [72]. Menetelmä perustuu siihen, että opetusdatan sisältämien piirrevektoreiden avulla opetetaan LVQ-koodikirja. Nyt koodikirjaa ei käytetä suoraan luokitteluun, kuten kohdassa 4.4 esitettiin, vaan jokaiselle opetetulle prototyyppivektorille annetaan sen perusteella indeksi.

Kun indeksit on määrätty koodikirjasta haetaan piirrevektoria vastaava lähin koodikirjavektori, ja annetaan piirrevektorille kyseisen koodikirjavektorin indeksi. Kun tämä tehdään kaikille sekvenssin piirrevektoreille, saadaan ne esitettyä diskreettinä sekvessinä koodikirjavektoreiden indeksejä. LVQ:n voidaan tässä tapauksessa ajatella olevan enkooderi, joka kuvaa diskreetit piirrevektorit diskreeteiksi symboleiksi.

Tämän jälkeen HMM-mallit opetetaan normaalisti käyttäen näitä koodikirjan avulla muutettuja koodisekvenssejä. Tuntemattomalle piirrevektorisekvenssille haetaan myös koodikirjasta lähimpien prototyyppivektorien indeksit ja luokitellaan suurimman uskottavuuden antaman HMM-mallin mukaan. Kuvassa 18 on esitetty rakennekuva LVQ-HMM-yhdistelmän opettamisesta ja luokittelusta.



Kuva 18. Rakennekuva LVQ-HMM-yhdistelmän opettamisesta ja tuntemattoman näytesekvenssiin luokittelusta.

5. PIIRTEENVALINTA

Tässä luvussa käsitellään kuvaavien piirteiden ominaisuuksia ja oikean abstraktiotason eli piirrevektorin dimensionaalisuuden valintaa. Kuten kohdassa 2.3.3 todettiin, piirrrevalintaa voidaan pitää tunnistusjärjestelmän olennaisena osana, jotta käytetyt tunnistusmenetelmät saataisiin toimimaan parhaalla mahdollisella tavalla. Askeltunnistuksen piirteiden irroituksessa ja valinnassa on tarkoitus löytää yksittäisten henkilöiden yhtenäiset sekä muista erottuvat ominaisuudet ja vastaavasti tarjota tunnistusmenetelmille sopiva määrä piirteitä yleistyskyvyn säilyttämiseksi.

Luvussa esitetään kaksi erilaista, tässä työssä käytettyä, menetelmää piirteenvalintaan sekä niihin läheisesti liittyviä tekniikoita. Ensimmäistä menetelmää käytetään siten, että irroitetuista piirteista valitaan optimaalinen osajoukko ennen luokittimen opetusta. Toisessa menettelyssä lisätään järjestelmän adaptiivisuutta lisäämällä piirrevalinta käytetyn luokittimen opetusvaiheeseen. Tässä työssä se liitetään LVQ-luokittimeen, jolloin piirteiden valinta ja skaalaus tapahtuu automaattisesti yksittäisten piirteiden hyvyyden mukaan [13].

5.1. Kuvaavat piirteet ja abstraktiotaso

Luokittelutehtävissä kuvaavien piirteiden määrittäminen on luokittimen onnistumisen kannalta välttämätöntä. Luokittelumenetelmästä riippumatta niiden tulisi kuvata tunnistettavien luokkien käyttäytyminen mahdollisimman oikealla tavalla. Suurin osa luokittelun onnistumisesta riippuu onnistuneista piirrevalinnoista ja kohteen oikeanlaisen käyttäytymisen kuvaamisesta. [73]

Luokittelussa kuvaavat piirteet tulisi valita siten, että ne toisaalta pystyvät kuvaamaan samaan luokkaan kuuluvien näytteiden yhtenäiset ominaisuudet että erottamaan eri luokkiin kuuluvat näytteet toisistaan mahdollisimman hyvin. Ideana on siis etsiä erottavat piirteet, jotka ovat muuttumattomia sisääntulon merkityksettömille muunnoksille. [57, s. 11–12]

Piirteiden suuri määrä on yleisin virhe piirrevalinnassa. Tällöin liian yksityiskohtaisten piirteiden valinta aiheuttaa väärän abstraktiotason [73], jolloin malli sisältää liikaa vapausasteita. Kun opetusnäytteiden määrä pysyy samana piirremäärän kasvaessa, luokittelumallin tuntemattomien parametrien määrä kasvaa [74], jolloin opetuksesta riippumattomien näytteiden luokittelu vastaavasti huononee. Myös piirteiden liian vähäinen määrä voi johtaa huonoon lopputulokseen sillä, tällöin piirrejoukosta voi puuttua oleellisia ilmiötä kuvaavia piirteitä, ja ollaan taas väärässä abstraktiotasossa [73]. Piirteenvalinnassa on siis tärkeää valita ilmiötä kuvaavat piirteet ja toisaalta pitää tuntemattomien parametrien määrä tarpeeksi pienenä.

5.2. Piirteenirroitus

Ilmiötä kuvaavien piirteiden löytämiseen on olemassa monenlaisia menetelmiä. Piirteiden valintaa voi tapahtua jo datan esikäsittelyvaiheessa, jolloin moniulotteinen ilmiö kuvataan matalampaan abstraktiotasoon. Tällöin puhutaan piirteenirroituksesta tai projisoinnista, jossa sisääntuleva data kuvataan jonkin line-

aarisen tai epälineaarisen muunnoksen tai piirteiden yhdistelmien avulla uuteen matalampaan abstraktitasoon. Yleisesti käytettyjä menetelmiä on esitetty lähteessä [74].

5.3. Piirteenvalinta

Piirteiden valintaa voi tapahtua myös niiden irroituksen jälkeen. Tähän voidaan käyttää osapiirrejoukon testaamista, jossa suuresta joukosta valmiiksi laskettuja piirteitä valitaan optimaalinen, ilmiötä parhaiten kuvaava, osajoukko maksimoimalla käytetty hyvyysmitta riippumattomalla testiaineistolla. Kun piirteiden lukumäärä pidetään tarpeeksi pienenä, säästetään luokittelukustannuksissa ja saavutetaan tarkempia luokittelutuloksia. Osapiirrejoukkojen valintamenetelmät tapahtuvat yleensä ennen luokittimen todellista opetusta, ja perustuvat yleensä saman tai samantapaisen luokittelumenetelmän käyttämiseen kuin itse luokittelussa. Yleisesti käytetty menetelmä on KNN, jolle valitaan aina piirreosajoukko ja testataan sen toimivuus riippumattomalla testiaineistolla. Luokitteluvirhe voi kuitenkin muuttua, jos itse luokittelussa käytetään eri menetelmää kuin piirteiden valinnassa.

Osapiirrejoukkojen valinta perustuu hakumenetelmiin, jossa testataan eri piirrejoukkojen kombinaatioita. Kaikista suorin ja yksinkertaisin menetelmä on käyttää kaikkien eri piirrekombinaatioiden kokeilemista ja sen jälkeen valita parhaimman luokittelutuloksen antava yhdistelmä. Tässä menetelmässä eri osajoukkojen määrä voi kuitenkin kasvaa erittäin suureksi, kun valintaa tehdään suuresta määrästä piirteitä, ja se on monesti laskennallisesti mahdoton toteuttaa. Tällöin on syytä käyttää kehittyneempiä menetelmiä. Hakuavaruutta voidaan pienentää käyttämällä niin sanottua haaraudu ja rajoitu -menetelmää (branch-and-bound). Tarkoituksena on suorittaa haku puumaisesti lisäämällä piirteitä osajoukkoon, kunnes luokittelutulos ei enää parane. Tätä suurempia osajoukkoja ei tarvitse enää testata vaan voidaan jatkaa puun toisesta haarasta. [74]

Kaksi edellä mainittua hakumenetelmää löytävät aina optimaalisen osajoukon, kuitenkin branch-and-bound-menetelmänkin kompleksisuus voi olla eksponentiaalista kokoluokkaa, ja on syytä käyttää laskennallisesti tehokkaampia menetelmiä. Eteenpäinhaku (forward fearch) on menetelmä, jossa aluksi kaikkia piirteitä testataan yksitellen ja valitaan parhaimman luokittelutarkkuuden antava piirre osajoukkoon ensimmäiseksi. Jäljelle jääneille piirteille toistetaan aina sama operaatio, kunnes kaikki piirteet on käyty läpi. Eteenpäinhaun ongelmana on se, että se löytää yksittäiset hyvät piirteet, mutta ei pysty havaitsemaan hyviä monen piirteen yhdistelmiä. Tämä voidaan välttää ottamalla käyttöön taaksepäinhaku (backward search). Taaksepäinhaussa aloitetaan kaikilla piirteillä ja otetaan yksi piirre kerrallaan pois joukosta. Se piirre, jota ilman luokitin antaa huonoimman tuloksen, asetetaan viimeiseksi, ja toistetaan operaatiota pienemmille osajoukoille, kunnes piirteitä ei ole enää jäljellä. Eteenpäintaaksepäin hakua (forward-backward search) voidaan käyttää, jos halutaan löytää vielä enemmän piirteiden välisiä ominaisuuksia. Tässä menetelmässä aloitetaan lisäämällä piirteitä eteenpäinhaun tavoin. Kun 2 piirrettä on valittu käytetään taaksepäinhakua valitsemaan toinen pois. Tämän jälkeen jatketaan lisäämällä aina kaksi piirrettä ja poistamalla yksi, kunnes kaikki piirteet ovat mukana. [75, luku 10.]

5.4. Piirteenvalinta opetuksen yhteydessä

Edellisessä kohdassa esitetty osapiirrejoukon valinta on usein laskennallisesti raskas, koska siinä täytyy kokeilla eri piirreyhdistelmien kombinaatioita. Lisäksi se toteutetaan ennen luokittimen opetusta, jolloin se pidentää esikäsittelyvaihetta luokitinta rakennettaessa.

Seuraavassa esitellään oppivaan vektorikvantisointiin perustuvia piirteen valinta ja skaalausmenetelmiä, jotka laajentavat luvussa 4 esitettyjä LVQ-opetusalgoritmeja. Näille menetelmille yhteistä on se, että ne kaikki perustuvat painotettuun euklidiseen etäisyysmittaan, joka pyrkii skaalaamaan sisääntulevan piirrevektorin painokerroinvektorilla. Piirteiden skaalaus ja valinta tapahtuu näissä menetelmissä automaattisesti, kun se yhdistetään koodikirjavektoreiden opetukseen. Näissä menetelmissä piirteenvalinta voidaan siis suorittaa automaattisesti luokittimen opetuksen aikana. Ne piirteet, jotka tarjoavat vähän informaatiota, saavat opetuksen aikana pienet painokertoimet, kun taas hyvät piirteet saavat suuremmat kertoimet. Kun tuntematon näytevektori tämän jälkeen luokitellaan käyttäen painotettua etäisyysmittaa, ei-informatiivisten piirteiden vaikutus häviää pienten painokerrointen myötä.

5.4.1. RLVQ

Relevanssi LVQ (RLVQ) [76] perustuu LVQ1-opetusalgoritmiin (ks. kohta 4.4.3). Etäisyysmittana käytetään painotettua euklidista eäisyyttä $|\mathbf{x}-\mathbf{m}|_w$, joka on määritetty näytteen $\mathbf{x}$ ja koodikirjavektorin $\mathbf{m}$ välille yhtälöllä

$$d_w = |\mathbf{x}-\mathbf{m}|_w = \sqrt{\sum_{i=1}^{N} w_i (x_i - m_i)^2}, \tag{51}$$

jossa $\mathbf{w} = (w_1, w_2, \ldots, w_N) \geq 0$ ($\sum_{i=1}^{N} w_i = 1$) on painokerroinvektori, $\mathbf{x}$ on näytevektori ja $\mathbf{m}$ koodikirjavektori. Painokerroinvektorin alkiot alustetaan yhtäsuuriksi $w_i = 1/N$ N:n ollessa piirteiden lukumäärä. Painokertoimia päivitetään LVQ1:n opetuksen aikana tai sen jälkeen seuraavasti:

1. Määritetään opetusnäytettä $\mathbf{x}$ vastaava lähin koodikirjavektori $\mathbf{m}$ laskemalla etäisyys kaikkiin koodikirjavektoreihin

2. Lasketaan kaikille painokertoimille w_i uusi arvo:

$$w_i = \begin{cases} \max\{w_i - \alpha|x_i - w_i|, 0\}, & \text{jos } \mathbf{x} \text{ ja } \mathbf{m} \text{ ovat samasta luokasta} \\ w_i + \alpha|x_i - w_i|, & \text{muulloin.} \end{cases} \tag{52}$$

3. Normalisoidaan painokertoimet $w_i = w_i/|w|$.

Yhtälössä (52) parametri α on opetuskerroin painokerrointen päivittämiseen. Painokerrointen päivitys yhdessä normalisoinnin kanssa antaa oikeaan luokitteluun johtaville piirteille suuren kertoimen ja toisaalta vähentää ei-relevanttien, väärään luokittelu tulokseen johtavien piirteiden painokertoimia. Algoritmista on kehitetty myös parannettuja versioita. Batch-RLVQ-menetelmä välttää iteraation joutumista paikallisiin minimikohtiin [77].

5.4.2. GRLVQ

Yleistetty relevanssi LVQ (GRLVQ) [78] perustuu yleistettyyn LVQ:in (GLVQ) [79], joka on vastaavasti parannettu version LVQ2.1 opetusalgoritmista (ks. kohta 4.4.5) ja pyrkii parantamaan opetuksen epästabilisuutta pitkällä aikavälillä. Koodikirjan opetus tapahtuu yhtälöillä:

$$\Delta \mathbf{m}_j = \alpha \frac{sgd'(\mu(\mathbf{x}))d_k}{(d_j + d_k)^2}(\mathbf{x} - \mathbf{m}_j) \tag{53}$$

$$\Delta \mathbf{m}_k = \alpha \frac{sgd'(\mu(\mathbf{x}))d_j}{(d_j + d_k)^2}(\mathbf{x} - \mathbf{m}_k), \tag{54}$$

joissa $\mathbf{x}$ on näytevektori, $\mathbf{m}_j$ ja $\mathbf{m}_k$ ovat lähimmät samaan ja eri luokkaan luokkaan kuuluvat koodikirjavektorit. d_j ja d_k ovat neliölliset etäisyydet näytteen ja koodikirjavektorien välillä ja $\mu(\mathbf{x}) = (d_j - d_k)/(d_j + d_k)$. Monotonisesti kasvava funktio $sgd(x) = (1 + \exp(-x))^{-1}$ tarjoaa korjauskertoimen koodikirjavektorien päivitykseen. Yhtälöä (53) käytetään päivittämään lähintä opetusnäytteen kanssa samaan luokkaan kuuluvaa koodikirjavektoria, kun taas yhtälöä (54) käytetään lähimmän eri luokkaan kuuluvan näytteen päivittämiseen α:n ollessa opetuskerroin.

GRLVQ-menetelmä saadaan lisäämällä edelliseen neliöity painotettu euklidinen etäisyysmitta d_w^2 (ks. kohta 5.4.1) sekä painokerrointen päivitys seuraavasti:

$$w_i = w_i - \alpha sgd'(\mu(\mathbf{x})) \cdot \Big(\frac{d_k}{(d_j + d_k)^2}(x_i - m_i^j)^2 - \frac{d_j}{(d_j + d_k)^2}(x_i - m_i^k)^2 \Big), \tag{55}$$

jossa yksittäisen piirteen i $(i = 1, 2 \ldots N)$ painokerrointa w_i päivitetään m_i^j:n ja m_i^k:n ollessa piirrettä vastaava arvo lähimmissä koodikirjavektoreissa. Parametri α on opetuskerroin, ja se valitaan standardi-algoritmien tapaan (ks. kohta 4.4.3). Päivityksen jälkeen painokertoimet normalisoidaan kuten RLVQ:ssa (ks. kohta 5.4.1).

5.4.3. DSLVQ

Erotusherkkä LVQ (DSLVQ) [80] on LVQ3-algoritmin (ks. kohta 4.4.6) laajennus ja on samankaltainen GRLVQ-menetelmän kanssa. Koodikirjavektoreiden opettaminen toteutetaan vastaavasti kuin LVQ3:ssa, mutta painotettua euklidista etäisyysmittaa käytetään mittamaan koodikirjavektorin $\mathbf{m}$ ja sisääntulevan näytteen $\mathbf{x}$ etäisyyttä. Painotettu etäisyysmitta d_w esitetään nyt muodossa

$$d_w(\mathbf{w}, \mathbf{x}, \mathbf{m}) = \sqrt{\sum_{n=1}^{N} [\max(0, w_n)(x_n - m_n)]^2}, \tag{56}$$

jossa $\mathbf{w} = (w_1, w_2, \ldots, w_N) \geq 0$ on painokerroinvektori, $\mathbf{x}$ on näytevektori ja $\mathbf{m}$ koodikirjavektori.

LVQ3 opetusalgoritmin aikana koodikirjavektorien päivitysten lisäksi päivitetään painokerroinvektoria $\mathbf{w}$, jossa yksittäisten piirteiden painokertoimet alustetaan ykkösiksi. Tällöin opetusalgoritmin ensimmäinen kierros vastaa LVQ3-algoritmin opetusta, koska kaikilla piirteillä on sama painokerroin etäisyysmittaa laskettaessa. Kun ajanhetkellä t näytettä $\mathbf{x}$ vastaavat lähimmät samaan ja eri luokkaan kuuluvat koodikirjavektorit ovat $\mathbf{m_j}$ ja $\mathbf{m_i}$ ja $\mathbf{x}$ osuu niiden välissä olevaan "ikkunaan", painokerroinvektoria päivitetään seuraavasti:

$$\mathbf{w}(t+1) = norm(\mathbf{w}(t) + \alpha(t)[\mathbf{nw}(t) - \mathbf{w}(t)]), \tag{57}$$

jossa

$$nw_n(t) = norm\Big(\frac{d_{i_n}(t) - d_{j_n}(t)}{\max(d_{i_n}(t), d_{j_n}(t))}\Big) \tag{58}$$

$$norm(\mathbf{y}) = \frac{\mathbf{y}}{\sum_{n=1}^{N} |y_n|} \tag{59}$$

$$d_{k_n}(t) = |x_n(t) - m_{k_n}(t)|, \qquad k \in \{i, j\}. \tag{60}$$

Yhtälössä (60) lasketaan yksittäisen piirteen etäisyys $d_{k_n}(t)$ näytevektorin $\mathbf{x}(t)$ ja kahden lähimmän koodikirjavektoreiden $\mathbf{m}_i(t)$ ja $\mathbf{m}_j(t)$ suhteen. Normalisointifunktio $norm(\mathbf{y})$ (yht. (59)) on skaalaava muunnos vektorille $\mathbf{y} = (y_1, y_2, \ldots y_n)$, jolloin skaalaustekijän tuloksena normalisoidun vektorin elementtien summa on yksi. Kun sekä $\mathbf{w}$ että $\mathbf{nw}$ (yht. (57)) normalisoidaan näin, saadaan piirteille tietty määrä luottamusta. Tämä luottamus jaetaan painokerrointen muodossa kuvaavien piirteiden välillä. Uusi painokerroin vektori $\mathbf{nw}$ lasketaan joka opetuskierroksella t yksittäisten piirteiden hyvyyden arvioinnin mukaan. Normalisoinnin myötä skaalaustekijän ollessa suuri, hyvien piirteiden lukumäärä on pieni. Tällöin painokerroinvektoria $\mathbf{w}$ siirretään kohti uutta painokerroinvektoria $\mathbf{nw}$ opetuskertoimen α mukaan (yht. (57)). Arviointifunktio vertailee yksittäisten piirteiden etäisyyttä opetusnäytteen $\mathbf{x}(t)$ ja kahden samaan ja eri luokkaan kuuluvien koodikirjavektorien $\mathbf{m}_i(t)$ ja $\mathbf{m}_j(t)$ välillä. Uusi painokerroin on suurempi, jos kyseinen piirrearvo n on lähempänä samaan luokkaan kuuluvan koodikirjavektorin vastaavaa arvoa (m_{j_n}), ja vastaavasti kauempana lähimmästä eri luokkaan kuuluvasta koodikirjavektorin arvosta (m_{i_n}).

Opetuksen aikana ei-informatiivisten piirteiden painokertoimet saavat pienet arvot, ja niiden vaikutus pienenee käytettäessä painotettua etäisyysmittaa tuntemattoman näytteen luokittelemiseen. Opetuskerroin α voidaan asettaa yhtäsuureksi kuin LVQ3:ssa, mutta jos käytetään suurta määrää koodikirjavektoreita, opetuskerrointa on hyvä pienentää, sillä painokertoimia päivitetään tällöin huomattavasti useammin kuin yksittäistä koodikirjavektoria keskimäärin.

6. LUOTETTAVUUDEN PARANTAMINEN

Tässä kappaleessa käsitellään luokittelun luotettavuuden parantamista esittelemällä hylkäyskriteeri, jolla epäluotettavat näytteet voidaan hylätä. Lisäksi käsitellään useampien luokittimien yhdistämistä parantamaan luokittelutuloksia ja esitetään työn aikana kehitetty 2-tasoinen askeltunnistin, jossa käytetään useaa sisääntulevaa samaan luokkaan kuuluvaa näytettä luokitteluvirheen pienentämiseksi [12]. Näiden menetelmien avulla lopullinen luokittelupäätös voidaan hyväksyä tai hylätä tunnistettavan henkilön osalta.

6.1. Epäluotettavien näytteiden hylkääminen

Vaikeissa luokitteluongelmissa epäluotettavuutta ja tunnistustuloksia voidaan parantaa hylkäyskriteerin avulla. Tämä tarkoittaa, että epäluotettavat näytteet voidaan hylätä kuulumattomina mihinkään ennalta määritettyyn luokkaan. Seuraavassa esitelty hylkäysoptio perustuu lähteissä [81, 82] kehitettyyn menetelmään. Kohdassa 6.1.1 esitetyt kuvat mukailevat myös lähdettä [81].

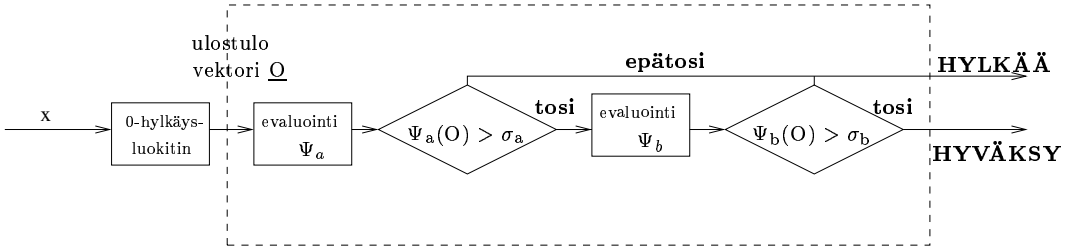
Menetelmän tarkoituksena on käyttää kahta erilaista hylkäyskriteeriä epäluotettavien näytteiden hylkäämiseen. Niistä ensimmäinen hylkää näytteet, jotka ovat kaukana kaikista opetetuista luokkarajoista. Toinen kriteeri hylkää näytteet, jotka asettuvat kahden tai useamman luokkarajan päällekkäiselle alueelle, eikä niitä näin ollen voida luokitella luotettavasti mihinkään tunnettuun luokkaan.

Hylkäämisen tarkoituksena on siis pyrkiä eliminoimaan mahdollisimman suuri osuus niistä näytteistä, jotka luokiteltaisiin väärin ilman hylkäyskriteeriä. Toisaalta huonona puolena on se, että osa oikein luokitelluista näytteistä hylätään. Tämän vuoksi menetelmässä on tarkoituksena määrittää optimaaliset kynnysarvot hylkäämiselle, sovelluskohtaisten kustannusarvojen mukaan. Hylkäysmenetelmä on riippumaton käytetyn luokittimen arkkitehtuurista, ja voidaan liittää suoraan luokittimen jatkoksi. Kuitenkin hylkäyskriteerien laskentaan käytettävät evaluointifunktiot täytyy määrätä käytetyn luokittelijan ulostulon mukaan. Jatkossa käytetään termiä 0-hylkäysluokitin, kun puhutaan luokittimesta ilman hylkäys mahdollisuutta.

6.1.1. Hylkäyskriteeri ja luotettavuuden evaluointi

Hylkäyskriteeri perustuu luotettavuuden evaluointiin, johon käytetään estimaattia Ψ. Evaluaattorin arvo määräytyy luokittelijan ulostulovektorin $\underline{O}$ mukaan ja se voi saada arvoja väliltä $[0, 1]$. Kun sisääntuleva näyte on suurempi kuin kiinnitetty kynnysarvo σ, luokittelu hyväksytään. Jos evaluaattorin arvo sitävastoin alittaa asetetun kynnysarvon, näytevektori hylätään. Kuvassa 19 on esitetty arkkitehtuuri, jossa 0-hylkäysluokittimen jatkoksi on liitetty hylkäysevaluaattorit.

Hylkäyskriteerin avulla väärin luokitellut näytteet, joiden Ψ arvo alittaa σ arvon hylätään. Tämä on haluttu tilanne, ja saadaan aikaan asettamalla väärin luokitelluille näytteille hylkäystä suurempi kustannusarvo. Toisaalta vastakkaisena vaikutuksena oikein luokitellut näytteet, joiden Ψ arvo on σ arvon alapuolella myös hylätään. Hylkäyskriteerin optimoimiseksi määritetään tehokkuusfunktio



Kuva 19. Hylkäävä luokitin.

P, jonka avulla saadaan suhteelliset painotukset näille kahdelle tilanteelle käytettävän luokittelijan mukaan.

Hylkäyskriteerin tehokkuutta voidaan arvioida asettamalla kustannusarvot väärin luokitellulle (C_e), hylätylle (C_r) ja oikein luokitellulle näytteelle (C_c). Ne ovat yleisiä funktioita luokittelu-, virhe- ja hylkäyssuhteille. Kun otetaan vielä käyttöön R_c, R_e ja R_r kuvaamaan näytejoukon prosentuaalisia osuuksia oikein ja väärin luokitelluille sekä hylätyille näytteille, voidaan tehokkuusfunktio P määrittää seuraavasti:

$$P = P(C_c R_c, C_e R_e, C_r R_r). \tag{61}$$

Funktio P mittaa hylkäyskriteerin tehokkuutta, ja sen täytyy olla kasvava R_c:n suhteen sekä vähenevä R_r:n ja R_e;n suhteen, jotka voidaan määrittää niiden osittaisderivaattojen avulla:

$$\frac{\partial P}{\partial R_c} > 0 \text{ ja } \frac{\partial P}{\partial R_r} < 0, \frac{\partial P}{\partial R_e} < 0. \tag{62}$$

Kun väärin luokittelulle annetaan hylkäystä suurempi negatiivinen vaikutus P funktioon, saadaan tehokkuusfunktiolle vielä ehto

$$\left| \frac{\partial P}{\partial R_r} \right| < \left| \frac{\partial P}{\partial R_e} \right|. \tag{63}$$

Kun tehokkuusfunktio P pidetään itsenäisenä ja erossa varsinaisesta 0-hylkäysluokittimesta, voidaan 0-hylkäysluokittimelle määrittää luokittelu- ja virhesuhteet R_c^0 ja R_e^0, jolloin yhtälö (61) saadaan muotoon

$$P = P(C_c(R_c - R_c^0), C_e(R_e - R_e^0) - C_r R_r). \tag{64}$$

Monissa käytännön sovelluksissa edellä esitetyt kustannukset (C_c, C_e ja C_r) voidaan pitää vakioina sekä funktio P lineaarisena, jolloin P on verrannollinen oikein luokiteltujen näytteiden kokonaisvaikutukseen sekä hylättyjen ja väärin luokiteltujen näytteiden kustannuksiin. Näiden oletuksien ja yhtälön (64) mukaan P voidaan kirjoittaa muotoon

$$P = C_c(R_c - R_c^0) - C_e(R_e - R_e^0) - C_r R_r, \tag{65}$$

joka on yhtäpitävä yhtälön (64) kanssa ehdolla $C_e > C_r$.

Jatkossa P funktion oletetaan olevan yhtälön (65) muotoa. Koska oikein ja väärin luokiteltujen sekä hylättyjen näytteiden suhteelliset osuudet riippuvat hylkäyskynnyksen σ arvosta, yhtälö (65) on myös σ:n funktion. Tehokkuusfunktion P riippuvuus muuttujaan σ voidaan esittää oikein ja väärin luokiteltujen näytteiden esiintymistiheyksinä, jotka ovat luotettavuusevaluaattorin Ψ arvon funktioita. Ne merkitään muodossa $D_c(\Psi)$ ja $D_e(\Psi)$.

Tiheysfunktioiden $D_c(\Psi)$ ja $D_e(\Psi)$ integraalit välillä $[\Psi_1, \Psi_2]$ kuvaavat prosentuaaliset osuudet oikein ja väärin luokitteluille näytteille, kun Ψ arvo muuttuu $\Psi_1 - \Psi_2$ verran. Näiden määritysten avulla R_c^0 ja R_e^0 voidaan laskea seuraavasti:

$$R_c^0 = \int_0^1 D_c(\Psi)\,\mathrm{d}\Psi \text{ ja } R_e^0 = \int_0^1 D_e(\Psi)\,\mathrm{d}\Psi. \tag{66}$$

Näistä jakaumista voidaan hylättyjen näytteiden suhteellinen osuus laskea, kun 0-hylkäysluokittimella saadaan suhteelliset osuudet oikein ja väärin luokitelluille näytteille. Ψ:n arvot, jotka alittavat kynnyksen σ vastaavat suoraan $R_r^{(c)}$:n ja $R_r^{(e)}$:n suhteellisia osuuksia, joten $R_r(\sigma)$ voidaan laskea tiheysfunktioiden $D_c(\Psi)$ ja $D_e(\Psi)$ avulla:

$$\begin{aligned} R_r(\sigma) &= R_r^{(c)}(\sigma) + R_r^{(e)}(\sigma) \\ &= \int_0^\sigma D_c(\Psi)\,\mathrm{d}\Psi + \int_0^\sigma D_e(\Psi)\,\mathrm{d}\Psi. \end{aligned} \tag{67}$$

Yhtälön (67) tilanne on esitetty kuvassa 20. Samalla tavalla voidaan määrittää suhteelliset osuudet oikein ja väärin luokitelluille näytteille, jotka on esitetään kynnysarvon σ avulla. Tämä tapahtuu seuraavasti:

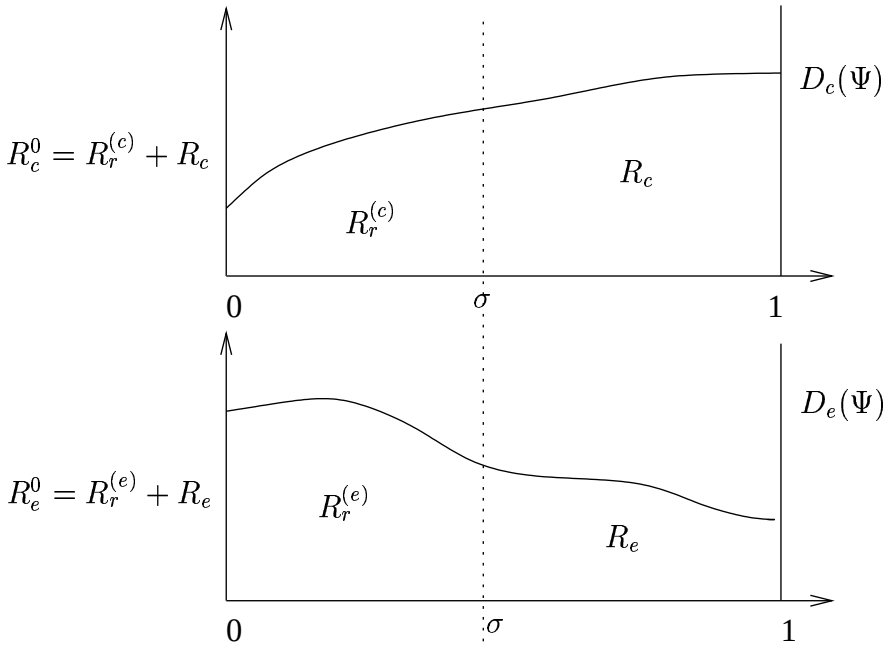
$$R_c(\sigma) = \int_\sigma^1 D_c(\Psi)\,\mathrm{d}\Psi \text{ ja } R_e(\sigma) = \int_\sigma^1 D_e(\Psi)\,\mathrm{d}\Psi. \tag{68}$$

Kun sijoitetaan yhtälöt (67) ja (68) yhtälöön (65) saadaan

$$\begin{aligned} P(\sigma) &= C_c\Big(\int_\sigma^1 D_c(\Psi)\,\mathrm{d}\Psi - \int_0^1 D_c(\Psi)\,\mathrm{d}\Psi\Big) \\ &- C_e\Big(\int_\sigma^1 D_e(\Psi)\,\mathrm{d}\Psi - \int_0^1 D_e(\Psi)\,\mathrm{d}\Psi\Big) \\ &- C_r\Big(\int_0^\sigma D_c(\Psi)\,\mathrm{d}\Psi + \int_0^\sigma D_e(\Psi)\,\mathrm{d}\Psi\Big), \end{aligned} \tag{69}$$

ja sievennetyssä muodossa

$$\begin{aligned} P(\sigma) &= (C_e - C_r)\int_0^\sigma D_e(\Psi)\,\mathrm{d}\Psi \\ &- (C_r + C_c)\int_0^\sigma D_c(\Psi)\,\mathrm{d}\Psi. \end{aligned} \tag{70}$$



Kuva 20. Esiintymistiheysjakaumat $D_c(\Psi)$ ja $D_e(\Psi)$. $R_r^{(c)}$ ja $R_r^{(e)}$ ilmoittavat hylättyjen näytteiden prosentuaaliset osuudet, kun kynnysarvo σ on määritetty. Vastaavasti R_c ja R_e ovat hyväksyttyjen oikein ja väärin luokiteltujen näytteiden prosentuaaliset osuudet hylkäyskriteerin jälkeen.

Koska yhtälön (70) molemmat integraalifunktiot ovat monotonisesti kasvavia ja $C_e > C_r$, σ:n kasvattaminen kasvattaa P:n arvoa ensimmäisen termin osalta ja pienentää sitä toisen termin osalta.

Tehokkuusfunktion P avulla voidaan määrittää optimaalinen kynnysarvo $\underline{\sigma}$ hylkäykselle, joka on

$$\underline{\sigma} : P(\underline{\sigma}) \geq P(\sigma), \qquad \forall \sigma \in [0,1] \tag{71}$$

kustannusarvojen ollessa kiinnitettyinä. Näin ollen P:n maksimiarvo riippuu jakaumista D_c ja D_e, joiden muoto riippuu käytetystä luotettavuusevaluaattorista. Tämän takia luotettavuusevaluaattori on sitä parempi, mitä lähempänä nollaa D_e:n arvot ovat ja mitä lähempänä ykköstä D_c:n arvot ovat. Ideaalisessa tilanteessa 0-hylkäysluokittimelta tulevat väärin luokitellut arvot voidaan hylätä ja vastaavasti oikein luokitellut näytteet hyväksyä. Näin tapahtuu, jos luotettavuusevaluaattorin antamat jakaumat ovat seuraavanlaiset:

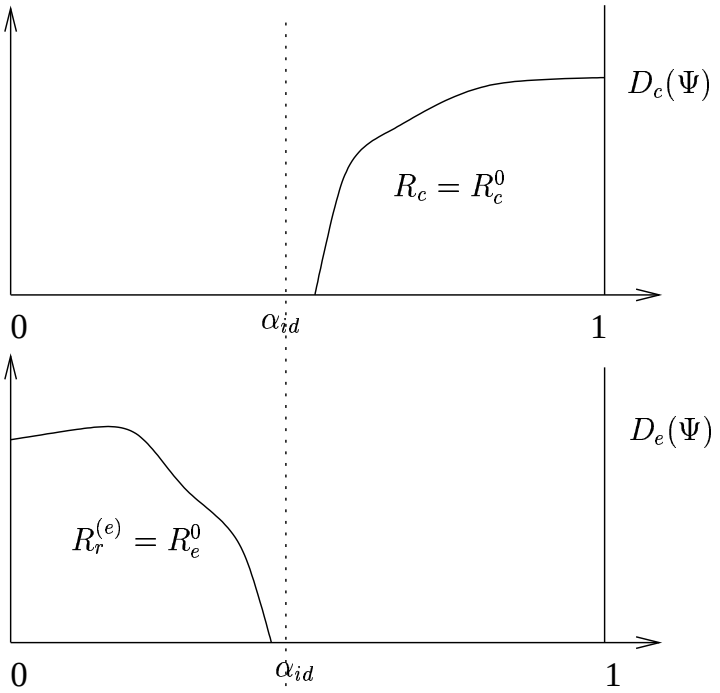
$$\begin{aligned} D_c(\Psi) &= 0, \qquad \forall \Psi \leq \Psi_1 \quad \text{ja} \\ D_e(\Psi) &= 0, \qquad \forall \Psi \geq \Psi_2, \quad \text{kun} \Psi_1 \leq \Psi_2. \end{aligned} \tag{72}$$

Tehokkuusfunktion P ideaalinen arvo P_{id} voidaan määrittää valitsemalla kynnysarvo σ_{id} väliltä $[\Psi_1, \Psi_2]$, jolloin yhtälöstä (70) saadaan

$$P_{id} = (C_e - C_r) \int_0^{\sigma} D_e(\Psi)\, \mathrm{d}\Psi = (C_e - C_r) R_e^0. \tag{73}$$

Ideaalinen kynnysarvon α_{id} valintatilanne on esitetty kuvassa 21. Siinä kaikki väärin luokitellut näytteet voidaan hylätä, ja oikein luokitellut näytteet voidaan

hyväksyä. Koska edellä mainittua tilannetta on vaikea saavuttaa todellisissa toteutuksissa, luotettavuusevaluaattoreille pitää pyrkiä löytämään mahdollisimman lähellä ideaalista tilannetta oleva ratkaisu.



Kuva 21. Kuvassa on esitetty esiintymistiheysjakaumat tilanteesta, jossa voidaan määrittää optimaalinen P_{id}:n arvo. Kaikki väärin luokitellut näytteet, jotka ovat kynnysarvon α_{id} alapuolella, voidaan hylätä, ja vastaavasti kaikki oikein luokitellut näytteet voidaan hyväksyä kynnysarvon yläpuolella.

6.1.2. Hylkäykriteerien määrittäminen opetusdatasta

Hylkäyskriteerit voidaan määrittää annetulle 0-hylkäysluokittimelle suoraan opetusdatasta. Kun luotettavuusevaluaattorit Ψ_a ja Ψ_b on määritetty ja jakaumat D_c ja D_e laskettu, voidaan hylkäyskriteerin optimaalinen kynnysarvo $\underline{\sigma}$ valita. Tähän käytetään 0-hylkäysluokitinta, jonka avulla luokitellaan data, jota on myös käytetty luokittimen opettamiseen. Olettamalla että kyseinen datajoukko pystyy kuvaamaan kohdeympäristön kattavasti, voidaan optimaalinen hylkäyskynnys määrittää etsimällä σ yhtälön (71) mukaan. Tämän laskemiseksi määritetään yhtälön (70) derivaatan ääriarvokohdat muuttujan σ suhteen seuraavasti:

$$C_N D_e(\sigma) - D_c(\sigma) = 0, \tag{74}$$

jossa $C_N = (C_e - C_r)/(C_r + C_c)$, normalisoitu kustannusarvo.

Yhtälön (74) ratkaiseminen tapahtuu numeerisella algoritmilla, sillä funktioita $D_c(\Psi)$ ja $D_e(\Psi)$ ei voida esittää analyyttisessä muodossa. Kun kustannuskertoimet ovat kinnitettyjä, voidaan $\underline{\sigma}$ määrittää seuraavalla opetusalgoritmilla.

1. Opetusdata luokitellaan 0-hylkäys luokittimella ja jaetaan kahteen näytejoukkoon: S_c oikein luokitelluille näytteille ja S_e väärin luokitelluille näytteille.

2. Luotettavuus evuluaattoreiden arvot Ψ_a and Ψ_b määritetään joka näytteelle, näytejoukoissa S_c and S_e. Tämän jälkeen lasketaan esiintymistiheysfunktiot $D_c(\Psi_a)$, $D_c(\Psi_b)$, $D_e(\Psi_a)$ ja $D_e(\Psi_b)$.

3. Lasketaan σ_a ja σ_b arvot, jotka toteuttavat yhtälön (74).

4. Lopuksi valitaan kynnysarvoiksi σ_a ja σ_b ne arvot, jotka kohdassa 3 valituista arvoista maksimoivat yhtälön (65).

Yhtälöstä (74) nähdään, että optimaalinen kynnysarvo $\underline{\sigma}$ riippuu normalisoidusta kustannuksesta C_N. Kaikki kolme kustannusarvoa $(C_c - k, C_r + k, C_e + k)$ antavat saman arvon normalisoidulle kustannukselle k:n vaihdellessa ja saman ratkaisun yhtälölle (74). Tyyppillisesti C_c on huomattavasti muita kustannuskertoimia pienempi, jolloin normalisoitu kustannus $C_N \approx (C_e/C_r) - 1)$ ja optimaalinen hylkäyskynnys riippuu vain suhteesta C_e/C_r.

6.1.3. Luotettavuusevaluointi oppivalle vektorikvantisoinnille

Seuraavassa esitetään luotettavuusevaluaattori oppivalle vektorikvantisoinnille. Kuten luvussa 4 esitettiin, LVQ-luokittelun ulostulovektori määritetään sisääntulevan näytteen etäisyytenä jokaisesta opetetusta koodikirjavektorista. Näyte $\mathbf{x}$ luokitellaan siis kuuluvaksi luokkaan k seuraavasti:

$$class(\mathbf{x}) = k : |O_k = O_{WIN} = min_i\{d(\mathbf{m}_i, \mathbf{x})\}|, \tag{75}$$

jossa O_{WIN} on minimietäisyys koodikirjavektorin $\mathbf{m}_i$ ja näytteen $\mathbf{x}$ välillä.

Ensimmäinen luotettavuusevaluaattori Ψ_a voidaan määrittää muodossa

$$\Psi_a = \begin{cases} 1 - \frac{O_{WIN}}{O_{max}}, & \text{jos } O_{WIN} \leq O_{max} \\ 0, & \text{muuten,} \end{cases} \tag{76}$$

jossa O_{max} on suurin O_{WIN}:n arvo opetusaineistossa. Yhtälön (76) toinen ehto estää Ψ_a:n negatiiviset arvot, jos $O_{WIN} > O_{max}$. Luotettavuusevaluaattoria Ψ_a voidaan käyttää hylkäämään sellaiset näytevektorit, jotka liian kaukana kaikista opetetuista prototyyppivektoreista.

Toinen luotettavuusevaluattori Ψ_b pyrkii eliminoimaan näytteet, jotka ovat kahden tai useamman luokan päällekkäisellä alueella, ja voidaan laskea yhtälöstä

$$\Psi_b = 1 - \frac{O_{WIN}}{O_{2WIN}}, \tag{77}$$

jossa O_{WIN} on niin ikään lähimmän koodikirjavektorin etäisyys luokiteltavasta näytevektorista ja O_{2WIN} on etäisyys toiseksi lähimmästä koodikirjavektorista. Mitä suuremman arvon Ψ_b saa sitä kauempana toiseksi lähin koodikirja on näytteestä suhteessa lähimpään prototyyppivektoriin.

6.2. 2-tasoinen luokittelumenetelmä

6.2.1. Luokittimien yhdistäminen

Luokittimien yhditämisellä voidaan lisätä luotettavuutta sekä parantaa usein luokittelutuloksia. Luokittimien yhdistäminen voidaan jakaa karkeasti kahteen tasoon. Piirretasolla tapahtuva yhdistäminen pyrkii piirteenirroitusvaiheessa löytämään eri menetelmien avulla ilmiötä kuvaavia piirteitä, ja yhdistämään ne paremman luokittelutuloksen saavuttamiseksi. Toinen yleisimmin käytetty menetelmä tapahtuu päätöstasolla. Tämä tarkoittaa, että yksittäisten luokittimien päätöstä käytetään yhteispäätökseen. [83]

Tyypillisesti päätöstason luokittimien yhdistämisessä käytetään usean erilaisen luokittelijan yhteispäätöstä luokittelutuloksen parantamiseksi. Tällöin erilaiset itsenäiset luokittimet voivat tarjota kattavamman kuvan tunnistettavasta kohteesta käyttäen esimerkiksi erilaisia piirrejoukkoa sisääntulevan näytteen suhteen tai kokonaan eri opetusaineistoja. Päätöstason luokittimet voivat olla myös samanlaisia, jolloin ne voidaan opettaa käyttämällä eri alustusarvoja [83]. Usean itsenäisen LVQ-luokittimen yhdistämistä on käytetty esimerkiksi käsinkirjoitettujen merkkien tunnistamisessa [84].

6.2.2. Hylkäävä 2-tasoinen askeltunnistin

Tässä työssä on kehitetty 2-tasoinen askelluokitin [12], jossa tunnistamiseen käytetään kolmea peräkkäistä askelta. Menetelmä poikkeaa tavanomaisista luokittimien yhdistämisestä, sillä tavallisesti yhdistäminen kohdistuu yksittäisen näytteen tunnistamiseen. Sovelluskohde tarjoaa kuitenkin mahdollisuuden käyttää usean näytteen yhteispäätöstä. Kun tunnistettava henkilö kävelee älykkääseen huoneeseen, kolme askelta saadaan tallennettua noin kolmen sekuntin aikana yhteispäätöstä varten. Jos taas tilanne on se, että henkilö on jo huoneessa ja liikkuu toiseen paikkaan voidaan henkilön usean askeleen tunnistus sitoa henkilön paikantamiseen etsimällä paikannusta vastaavat aikasarjassa lähellä olevat tapahtumat.

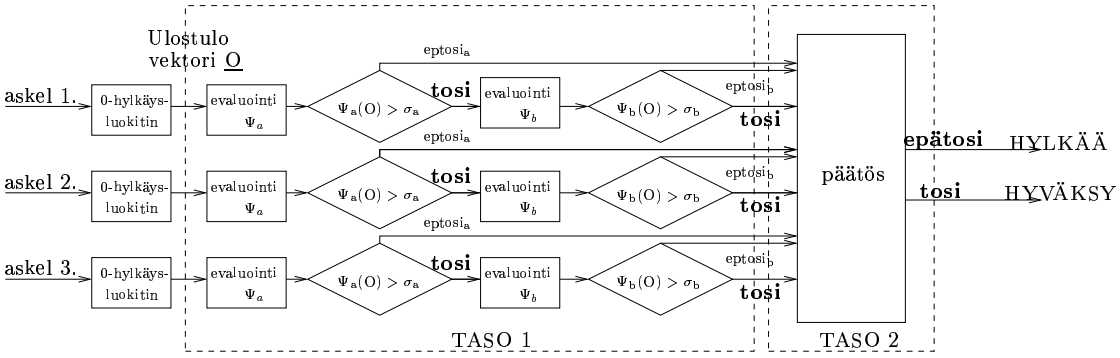
2-tasoinen tunnistin on esitelty kuvassa 22. Ensimmäinen taso luokittelee yksittäiset askeleet käyttäen yhtä ennalta opetettua LVQ-koodikirjaa. Ensimmäisellä tasolla yksittäinen askel luokitellaan tai hylätään. Toinen taso käyttää näiden kolmen yksittäisen askeleen luokittelutietoa lopullisen päätöksen tekemiseen. Lopullinen päätös on määrätty seuraavasti:

- HYLKÄÄ
    1. Jos kolmen askeleen enemmistö on hylätty Ψ_a:n mukaan.
    2. Jos yksi kolmesta askeleesta on hylätty Ψ_a:n mukaan ja yksi on hylätty Ψ_b:n mukaan.
    3. Jos kaikki askeleet on luokiteltu eri luokkiin kuuluviksi.
- HYVÄKSY
    1. Jos enemmistö on luokiteltu samaan luokkaan.

2. Jos kaksi askelta on hylätty Ψ_b:n mukaan ja yksi on luokiteltu johonkin luokkaan kuuluvaksi.

Jotta saataisiin luotettava kolmen askeleen luokittelutulos, eri askeleet on luokiteltu ensimmäisellä tasolla toisista riippumatta. Tämä tarjoaa mahdollisuuden sille, että yksittäiset huonot askeleet eivät vaikuta lopputulokseen, mutta toisaalta taas epäluotettavat kolmen askeleen ryhmät voidaan hylätä. Edellä esitetyn algoritmin ensimmäinen hylkäys perustuu siihen, että enemmistö askelista on liian kaukana kaikista protoyyppivektoreista opetetussa koodikirjassa, ja tällaisessa tilanteessa voidaan olettaa, ettei henkilö kuulu opetettujen henkilöiden joukkoon. Seuraavassa kohdassa yksi askel on kaukana kaikista opetetuista prototyyppivektoreista, ja toinen askel on liian lähellä usean luokan päätösrajaa. Tällöinkään ei voida henkilöä luokitella kovin luotettavasti, sillä yksittäiset askeleet poikkeavat niin paljon, että henkilö ei joko kuulu opetettuun joukkoon tai yksittäiset askeleet ovat häiriöisiä. Myöskään viimeisessä säännössä, jossa kaikki askeleet on luokiteltu eri luokkiin, ei voida luotettavasti valita oikeaa luokkaa.

Ensimmäinen hyväksymissääntö luokittelee tuntemattoman sekvenssin enemmistön mukaan, tällöin mukana voi olla esimerkiksi yksi häiriöinen askel, mutta sekvenssiä ei sen takia hylätä. Toinen sääntö tarjoaa liikkumavaraa luokkien rajoilla. Kun kaksi askelista on hylätty luokan rajalta, voidaan olettaa, ettei yksittäiset askeleet ole kovin häiriöisiä, vaan ne kuuluvat johonkin luokkarajaa lähellä olevaan luokkaan. Tällöin päätös tehdään kolmannen askeleen perusteella, jos sitä ei ole hylätty ensimmäisellä tasolla.



Kuva 22. 2-tasoinen askeltunnistin.

7. ASKELTUNNISTIMIEN TOTEUTUS JA TUNNISTUSTULOKSET

Tässä luvussa esitetään askeltunnistusmenetelmien käytännöntoteutus ja tunnistustulokset. Luvun esitys etenee kronologisessa järjestyksessä, kuten kohdassa 2.3.3 esitettiin. Koska tällaisella sensorilla ei ole aikaisemmin tehty vastaavanlaista tutkimusta, toteutuksen eteneminen ei ollut kuitenkaan käytännössä niin suoraviivaista. Luvussa käsitellään pääasiassa lopullinen toteutus ja parhaat tunnistustulokset, joihin päästiin monien erilaisten kokeilujen ja erehdysten kautta, joita tässä luvun alussa hieman valotetaan.

Alkuvaiheessa kävelyaineistoa kerättiin siten, että koehenkilö käveli vapaasti ympäri huonetta. Kuitenkin pian huomattiin, että tällaisella järjestelyllä lattialta saatava aineisto ei olisi kovin hyvää tunnnistuksen kannalta. Ensinnäkin suurin osa yksittäisistä askelista oli jakautunut usealle anturille, joten kokonaista askelprofiilia olisi vaikea määrittää. Toisaalta yksittäisten askelten segmentoiminen raakasignaalista olisi haastaavaa, sillä esikäsittely- ja segmentointimenetelmiäkin vasta kehitettiin yhtäaikaa tunnistusmenetelmien kanssa. Tämän vuoksi tunnistusongelmaa varten kerättiin uusi aineisto. Yksityiskohdat aineiston keräämisestä esitetään kohdassa 7.1 ja esikäsittelyyn liittyviä menetelmiä ja ongelmia kuvataan kohdassa 7.2.

Myöskään piirteenirroitus ja -valinta eivät olleet alussa selviä, ja ne kulkivat käsikädessä tunnistusmenetelmien kehityksen kanssa, eli tiettyjä piirreyhdistelmiä testattiin erilaisilla luokittimilla ja tarvittaessa valittiin taas uusia kuvaavampia piirteitä, jos sillä hetkellä käytetyt eivät antaneet hyviä tuloksia. LVQ-luokittelussa piirteenvalinnan pohjana käytettiin lähteessä [8] esitettyjä askelprofiilin geometrisiä ominaisuuksia. Lisäksi signaalista irroitettiin myös paljon muita mahdollisia piirteitä liittyen sekä geometrisiin piirteisiin että yksittäisen askeleen taajuusominaisuuksiin. HMM-luokittelun alkuvaiheessa kokeiltiin lähteessä [9] esitettyä menettelyä, jossa piirresekvenssi muodostettiin aikasarjaa ikkunoimalla. Kolmen henkilön tunnistustulokset on esitelty lähteessä [14]. Tämä menettely ei toiminut odotetulla tavalla suuremmalle joukolle tunnistettavia henkilöitä, joten HMM-luokittelussa siirryttiin myös käyttämään geometrisiin ominaisuuksiin liittyviä piirteitä, jotka laskettiin kolmeosaisena sekvenssinä yksittäisen askeleen ajansuhteen tapahtuvien muutosten mukaan. Piirteenlaskentaa molempien tunnistusmenetelmien osalta on kuvattu kohdassa 7.3.

Piirteenlaskennan lisäksi menetelmien kehityksessä oli oleellista myös itse luokittimien erilaisten rakenteiden kokeileminen. Kuten kohdassa 7.4.1 esitetään LVQ:n osalta testattiin erilaisia koodikirjojen kokoja, opetusalgoritmeja ja opetusiteraatioiden määrää. Vastaavasti HMM-luokittelussa säädettiin LVQ-koodikirjaa sekä HMM-mallin ominaisuuksia, kuten tilojen määrää sekä tilansiirtoja tilatodennäköisyyksien alustusta. Valitut ominaisuudet on kerrottu kohdassa 7.4.2.

Valittujen luokittelumenetelmien tunnistulokset on esitetty kohdassa 7.5. Aluksi on esitetty parhaimmat tulokset yksittäisten askelten tunnistuksessa 11 henkilön osalta sekä mallinnettu luokiteltavien henkilöiden määrän vaikutusta luokitteluvirheeseen. Toisessa osuudessa käsitellään LVQ-laajennuksien toimivuutta ja osoitetaan niiden vielä parantavan edellä esiteyillä standardi-LVQ-algoritmeilla saatuja tuloksia.

7.1. Aineiston kerääminen

Aineiston keräämiseen käytettiin luvussa 3 esiteltyä järjestelmää, jossa luettiin tiedonkeruukortilta tuleva signaali tiedostoihin jatkokäsittelyä varten. Testiaineisto kerättiin helmikuussa 2003 sisältäen 11 eri henkilön kävelysignaalia EMFi-lattialla. Testiryhmään kuului 2 naista ja 9 miestä. Askelia tallennettiin jokaiselta 64:ltä kanavalta. Yksi EMFi-liuska oli merkitty teipillä, ja testihenkilöt pyrkivät astumaan "hyvälaatuisen" kokonaisen askeleen tälle liuskalle kävellessään, jotta saataisiin luokittelua varten kerättyä mahdollisimman hyviä yksittäisiä askelia ja niiden paikantaminen ja segmentointi olisi helppoa tässä tutkimuksen ensimmäisessä vaiheessa.

Jokainen henkilö kulki suoraviivaisesti edestakaisin huoneen päätyseinältä ovelle liuskojen suuntaisesti (ks. kuva 5, luku 3). Merkityltä kohdalta kerättiin noin 20 askelta molemmilta jaloilta. Henkilöt käyttivät omia jalkineitaan ja kävelivät tasaista, mahdollisimman luonnollista tahtia. Lisäksi jokaiselta henkilöltä kerättiin 10 askelta molemmilta jaloilta ilman kenkiä, jotta jalkineiden vaikutusta voitiin vertailla ilman kenkiä tuotettuihin askeliin. Myöskin kahdelle liuskalle osuvia askelia kerättiin jokaiselta testihenkilöltä, jotta niiden käyttäytymistä voitaisiin mallintaa.

7.2. EMFi-signaalin esikäsittely

7.2.1. Askelsignaalin segmentointi

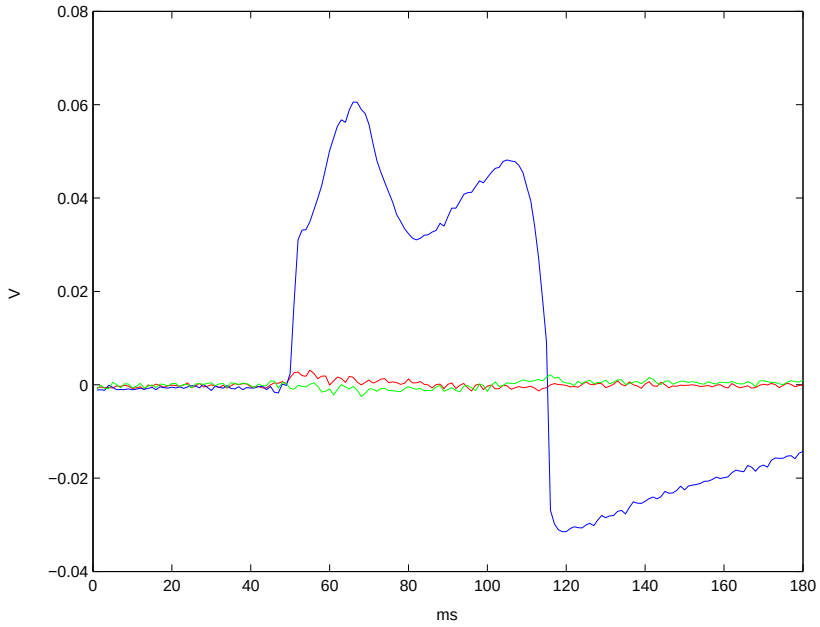
Ennen kuin yksittäisiä askelia voidaan tunnistaa, ne pitää segmentoida raakasignaalista. Tämä on haasteellinen tehtävä, sillä kanavan kohina sekä ei halutut signaalimuodot vaikeuttavat hyvien askelten erottamista. Tämän työn testiaineiston askelten segmentoitiin käytettiin reunanilmaisumenetelmiä, joissa haettiin FIR-mediaani-hybridi suodatuksen [85], konvoluution ja kynnystyksen avulla halutut signaalimuodot kanavasta. Menetelmä toimii hyvin, jos kanavassa on hyviä ja kokonaisia askelia sopivin välein esiintyen, kuten kerätty aineisto merkityn liuskan kohdalta näissä testeissä oli. Kynnystykseen perustuva segmentointi ei ole kuitenkaan riittävä reaalitoteutuksen kannalta. Segmentointia tutkitaan tarkemmin toisessa projektissa, joten sen käsittely on tässä työssä vähäistä.

Jotta haluttuja askelia voitaisiin löytää ja segmentoida reaaliaikaisesti vaihtelevan laatuisista kanavista, on kehitetty ja sovellettu tilastollista mallikuvion sovitusta. Menetelmä on nimeltään segmentiaaliset semi-Markovin mallit (SSMM), ja sitä voidaan pitää kätkettyjen Markovin mallien laajennuksena sisältäen tilahavaintojakaumien ja tilasiirtymätodennäköisyyksien lisäksi tilakestojakaumat. SSMM-menetelmä ei tarvitse erikseen opetusaineistoa, vaan siinä käytetään lineaarisesti paloittain jatkuvaa esimerkkimallia, jota sovitetaan tutkittavaan signaaliin. Sovitettava malli voidaan määrittää suoraan alkuperäisestä aaltomuodosta. Tarkemmat kuvaukset ovat alkuperäisen menetelmän osalta lähteessä [86], ja askelsignaalin segmentointiin sovellettu menetelmä on esitetty lähteessä [15].

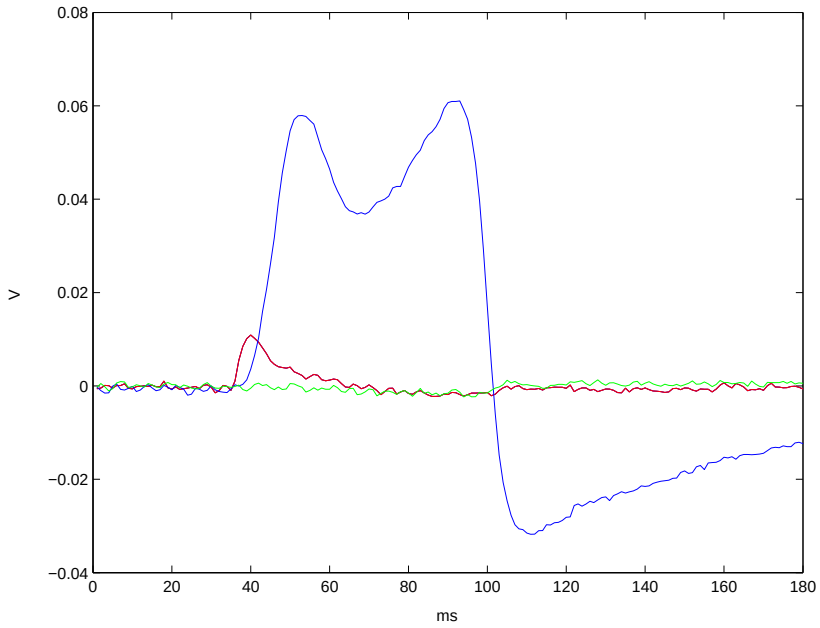
7.2.2. Signaalin jakautuminen usealle anturille

Seuraavassa on esitetty tilanteita, jotka vaikeuttavat hyvien kokonaisten askelten saamista tunnistusprosessiin. Koska mittausalusta koostuu kapeista liuskoista, yksittäiset askeleet jakaantuvat helposti kahdelle tai useammalle kanavalle. Kuvassa 23 on yksi kokonainen askel, kun signaali on otettu kolmelta peräkkäiseltä poikittaisliuskalta. Keskimmäisellä kanavalla näkyy kokonainen askel viereisten kanavien amplitudien pysytellessä lähellä nollaa. Kuvassa 24 askel on osunut pääasiassa keskimmäiselle liuskalle, mutta pieni osa askeleen alusta on osunut ensimmäiselle liuskalle. Tälläinen tilanne vaimentaa keskimmäiseen kanavaan osuvan pääaskeleen alkuosan muotoa, jolloin yksittäisen askeleen tunnistustarkkuus heikkenee. Koska askelosat käyttäytyvät epälineaarisesti, ei näiden kahden kanavan suora summaaminen anna vasteena vastaavaa kokonaista askelta kuten kuvassa 23. Kuvassa 25 askel on osunut puoliksi kahteen kanavaan, jolloin suora summaaminen toimii kohtuullisella tarkkuudella, ja tällaisia askelia voidaan käyttää tunnistuksessa.

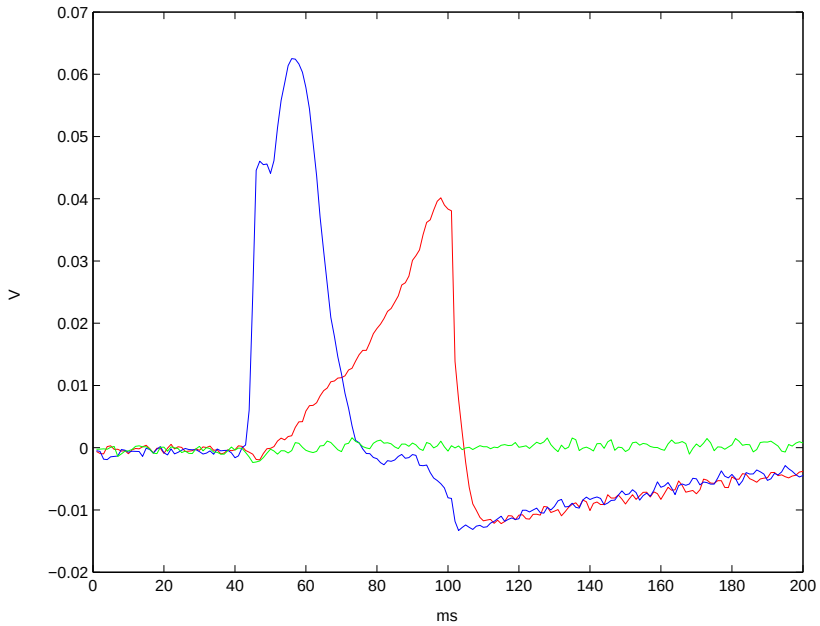
Kaikki edelliset esimerkkisignaalit nauhoitettiin kävelysuunnan suhteen poikittaisilta liuskoilta. Kuvassa 26 on esitetty tilanne, jossa testihenkilö on kävellyt suoraan pitkin yhtä pitkittäistä liuskaa. Tällaisissa tilanteissa kalvon palautuminen näkyy askelsignaalin amplituditason muutoksena, sillä kalvo ei ehdi palautua nollatasoon ennen uuden askeleen alkua. Tämä aiheuttaa myös vaikeuksia askeltunnistukseen, eikä niitä käytetty tämän työn testiaineistoissa. Jotta tällaisia askelia voitaisiin käyttää tunnistuksessa, täytyy niiden amplituditaso ensin normalisoida, varsinkin jos luokittelussa käytetyt piirteet liittyvät amplitudiarvoihin.



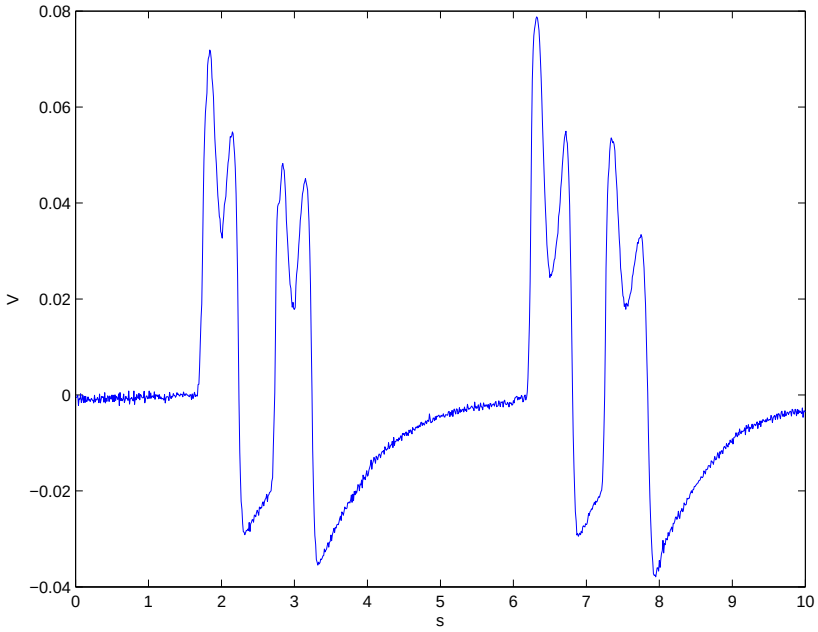
Kuva 23. Yhdelle EMFi-kanavalle osunut kokonainen askel.



Kuva 24. Pääasiassa yhdelle kanavalle osunut askel. Kuitenkin askeleen alku on osunut hieman viereiselle kanavalle, ja tämä vääristää pääkanavan askeleen muotoa.



Kuva 25. puoliksi kahdelle kanavalle osunut askel.



Kuva 26. Peräkkäisiä askelia yhdeltä pitkittäiseltä EMFi-liuskalta.

7.3. Piirteenlaskenta

7.3.1. Yksittäisen askeleen ominaisuudet

Kuvassa 27 on esitetty yksi kokonainen askel, johon on merkitty koordinaattipisteet sen tärkeistä vaiheista signaalin amplitudin ja ajan suhteen. Askeleen alku on merkitty koordinaateilla $[x_{start}, 0]$ signaalin amplitudin lähtiessä nousemaan. $[x_{max1}, y_{max1}]$ on maksimikohta, jolloin kantapää on osunut lattiaan, ja tämä nähdään ensimmäisenä amplitudin paikallisena maksimikohtana signaalissa. Tämän jälkeen kantapää irtoaa lattiasta ja amplitudi laskee. Koordinaattipiste $[x_{min}, y_{min}]$ ilmoittaa minimikohdan kantapään ja päkijän iskun välillä. Toinen huippukohta $[x_{max2}, y_{max2}]$ aiheutuu juuri päkijän iskusta lattiaan. Tämän jälkeen askel alkaa olla lopuillaan ja varpaat nousevat lattiasta. Kohtaa $[x_{mid}, 0]$ voidaan pitää varsinaisen askeleen loppupisteenä, kuitenkin piirteen laskennassa mukaan otettiin lisäksi kalvon ylipäästöluonteesta johtuva negatiivinen loppuosa, jonka jälkeen kalvon palautuminen alkaa. Seuraavassa esitellään piirteiden valintaa, joka pohjautuu pääasiassa edellä esitettyjen askeleen pääkohtien ja niiden suhteiden tarkastelemiseen. Piirreirroitus toteutettiin Matlab-ympäristössä [87]. Ensimmäiseksi segmentoidusta askeleesta määritettiin ääriarvokohdat (x_{max1} ja x_{max1}). Niiden määrittämiseen käytettiin reunanilmaisua ja konvoluutiota seuraavasti:

$$y[t] = \sum_{n=0}^{7} x[t-n]h[n], \tag{78}$$

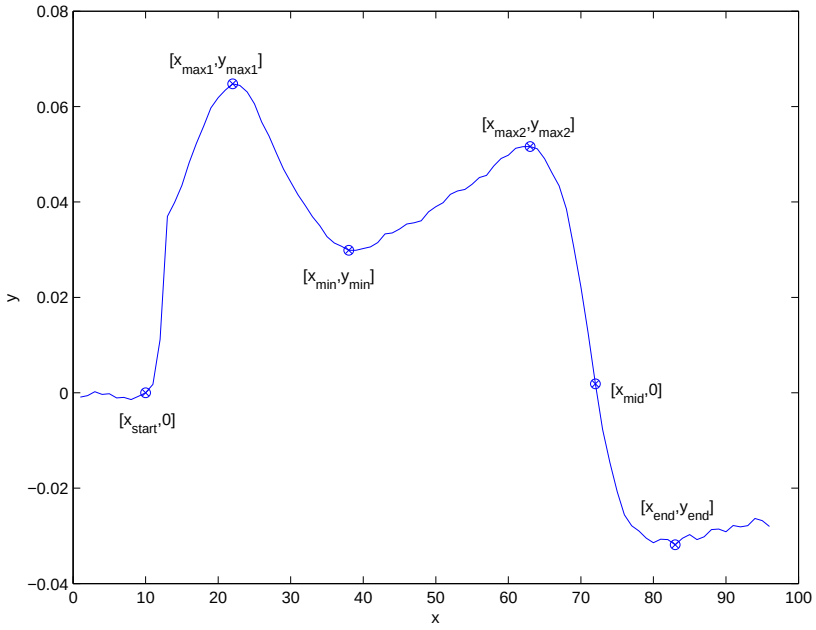
jossa

$$h[n] = \begin{cases} -1, \text{ kun} & n = 0 \\ 0, \text{ kun} & n = 1, 2 \ldots, 6 \\ 1, \text{ kun} & n = 7 \end{cases} \tag{79}$$

Yhtälössä (78) $y[t]$ kuvaa ajanhetkellä t konvoluution ulostuloa, kun $x[t]$ on sisääntuleva näyte. h on 8 näytteen mittainen konvoluutiomaski esitettynä yhtälössä (79). Ääriarvot löytyvät kynnystämällä alkuperäisestä signaalista laskettu konvoluutiovaste. Paikallisten maksimikohtien avulla saadaan määritettyä loput aika- ja amplituditason piirteet.

7.3.2. LVQ:n piirteenlaskenta

LVQ-luokittelun piirteenirroitusta varten segmentoitu askelsignaali jaettiin kahteen osaan, kannan ja päkijän aiheuttamien maksimien välisen minimin mukaan. Kuvassa 27 on esitetty koordinaattipisteet, joiden perusteella luokittelussa käytetty aika- ja amplitudipiirteet määritettiin.



Kuva 27. Yksittäisen askeleen koordinaattipisteet, joita käytettiin perustana LVQ-luokittelun piirrevalinnassa.

Yksittäisestä askeleesta irroitettiin yhteensä 31 piirrettä sisältäen pääasiassa ääriarvokohtien koordinaattipisteiden arvoja (ks. kuva 27), niiden välisiä suhteita sekä tilastollisia suureita, kuten keskiarvo ja keskihajonta tietyllä välillä. Taajuustasoesityksen laskemiseksi jokainen askel muutettiin 64 näytteen mittaiseksi. Tähän käytettiin interpolointia ja desimointia riippuen siitä, oliko alkuperäisen askeleen pituus suurempi tai pienempi kuin 64 aikayksikköä. Taajuusesitys laskettiin käyttäen nopeaa Fourier-muunnosta (FFT) [88] seuraavasti:

$$X[k] = \sum_{j=0}^{N} x[j]\omega_N^{(j-1)(k-1)}, \tag{80}$$

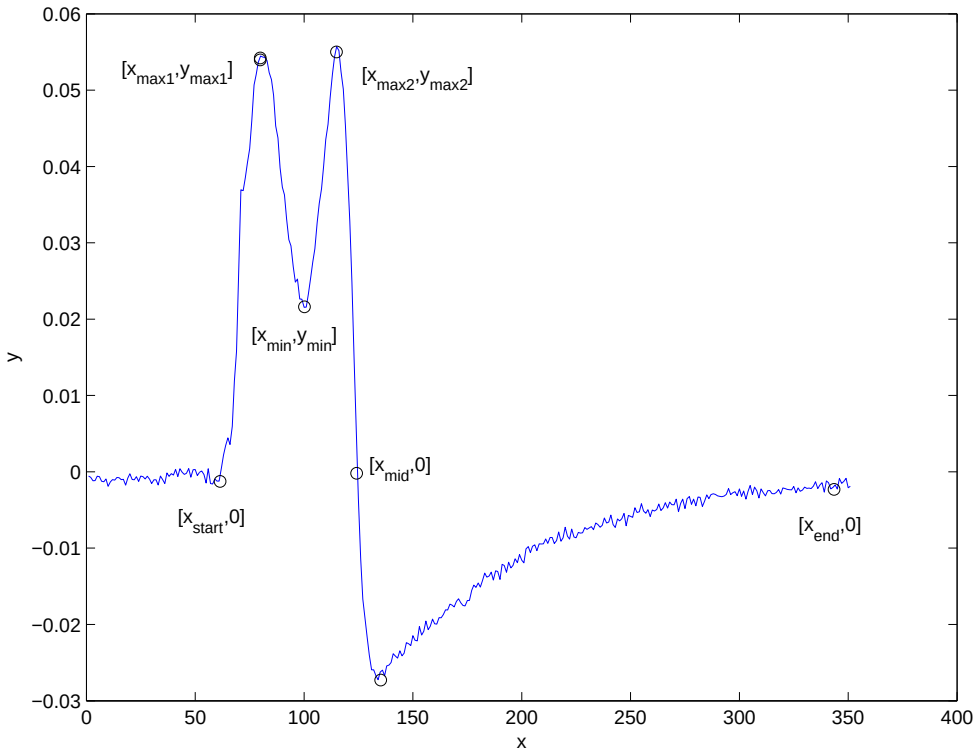
jossa $\omega_N = e^{(-2\pi)/N}$. N on tässä tapauksessa 63. Diskreetin Fourier-muunnoksen amplitudispektrin $|X[k]|$ $(k = 0, 1, \ldots, 63)$ 4 ensimmäistä kerrointa valittiin mukaan piirrejoukkoon, sillä suurin osa kävelyaskeleen taajuusinformaatiosta sisältyy pienille taajuuksille.

Ensimmäisessä vaiheessa kuvaavia piirteitä valittiin KNN-menetelmällä käyttäen LNKnet-ohjelmistoa [89]. Ohjelmistolla testattiin suuresta piirrejoukosta osapiirrejoukkoja käyttämällä kohdassa 5.3 esitettyä eteenpäin-taaksepäin-hakua valitsemaan paras osajoukko. Valitut piirteet olivat askeleen pituus, ensimmäisen osan (kanta) keskiarvo ja hajonta, toisen osan (päkijä) keskiarvo ja hajonta. Lisäksi valittiin kannan ja päkijän maksimiamplitudit ja niiden kohdat ajansuhteen askeleen alusta, niiden välisen minimikohdan amplitudin arvo ja aika sekä maksimiamplitudien suhde minimiamplitudiin. Myöskin askeleen päättymisen (negatiivinen jännite) kohdan amplitudi, kuten lähteessä [11] on esitetty. Valittuja piirteitä oli siis yhteensä 13. Toisessa vaiheessa luokitteluun käytettiin kaikki askeleesta irroitetut 31 piirrettä. Nyt piirteitä skaalattiin ja valittiin automaattisesti opetuksen yhteydessä käyttäen erotusherkkää LVQ:a (DSLVQ, kohta 5.4.3). Kaikki 31 LVQ-luokittelussa käytettyä piirrettä on esitelty tarkemmin liitteessä 1.

7.3.3. HMM-mallien piirteenlaskenta

Ensimmäisessä vaiheessa kätketyille Markovin malleille laskettiin piirresekvessit ikkunoimalla segmentoitu askel. Ikkunat olivat hieman päällekkäisiä ja jokaisessa aikaikkunassa signaalista laskettiin neljä piirrettä, jotka olivat signaalin maksimiarvo, minimiarvo, keskiarvo sekä keskihajonta. Erilaisia aikaikkunan kokoja ja ikkunan päällekkäisyyksiä testattiin. Parhaat tulokset saatiin 15 aikayksikön mittaisilla ikkunoilla, päällekkäisyyden ollessa 5 näytettä. Nämä piirresekvenssit mallinnettiin 6-tilaisilla diskreeteillä HMM-malleilla. VQ:n avulla muodostettiin koodikirja näille neljä piirrettä sisältäville piirrevektorien sekvensseille, ja määrättiin opetetusta koodikirjasta niille kokonaislukuarvo lähimmän koodikirjavektorin mukaan. Piirrevalinnan kuvaus ja tulokset kolmen henkilön tunnistuksesta on esitetty tarkemmin lähteessä [14].

Toisessa vaiheessa piirresekvenssi valittiin käyttäen osittain samoja piirteitä kuin LVQ-luokittelun yhteydessä. Piirrevalintaan liittyvät askeleen pääkohdat on esitetty kuvassa 28. Askel jaettiin kolmeen osaan: Askeleen alusta kantapään ja päkijän iskun väliseen minimikohtaan $(x_{start} - x_{min})$, minimikohdasta askeleen välipisteeseen $(x_{min} - x_{mid})$ sekä välipisteestä signaalin palautumiseen $(x_{mid} - x_{end})$. Jokaiseen osaan valittiin 8 piirrettä ja koodikirja tehtiin LVQ:n avulla. Näitä kolmen piirrevektorin sekvenssejä mallinnettiin kolmitilaisilla HMM-malleilla, jotka osoittautuivat paremmiksi askelsignaalin kuvaamiseen kuin ensimmäisessä vaiheessa [14] käytetyt 6-tilaiset HMM-mallit. Valitut piirteet on esitelty tarkemmin liitteessä 2.



Kuva 28. Yksittäisen askeleen koordinaattipisteet, joita käytettiin perustana HMM-tunnistuksen piirrevalinnassa.

7.3.4. Piirteiden normalisointi

Luokittelutehtävissä eri piirteiden arvojen kokoluokka voi vaihdella. Tällöin suuren kokoluokan piirteiden vaikutus luokitteluun on suurempi kuin pienemmän kokoluokan piirteillä. Ratkaisuna tähän on piirteiden normalisointi, jolloin ne saadaan yhteismitallisiksi. Yleisin tapa piirteiden normalisointiin on odotusarvon ja varianssin estimaattien laskenta. Kun käytössä on N kappaletta piirrevektoreita, saadaan k:s piirre yhteismitalliseksi seuraavasti:

$$\overline{x}_k = \frac{1}{N}\sum_{i=1}^{N} x_{ik},\ k = 1, 2, \ldots, l. \tag{81}$$

$$\sigma_k^2 = \frac{1}{N-1}\sum_{i=1}^{N}(x_{ik} - \overline{x}_k)^2. \tag{82}$$

$$\hat{x}_{ik} = \frac{x_{ik} - \overline{x}_k}{\sigma_k}. \tag{83}$$

Yhtälöllä (81) lasketaan näytekeskiarvo $\overline{x}_k$ opetusaineistossa ja yhtälöllä (82) varianssi σ_k^2. Lopuksi yhteismitallisuus saadaan käyttäen yhtälöä (83), jossa jokaisesta piirrearvosta vähennetään näytekeskiarvo $\overline{x}_k$ ja jaetaan se keskihajonnalla σ_k. Tuloksena piirteet normalisoituvat nollakeskiarvoisiksi yhteisellä

keskihajonnalla. Muita lineaarisia normalisointimenetelmiä ovat piirrearvojen skaalaaminen välille $[0, 1]$ tai $[-1, 1]$. [90, s. 141]

Tässä työssä yhteismitallisuuden saavuttamiseksi piirrevektorit skaalattiin välille [0,1] käyttäen yhtälöä

$$\hat{x}_{ik} = \frac{x_{ik} - \min(x_{1k}, x_{2k} \ldots x_{Nk})}{\max(x_{1k}, x_{2k} \ldots x_{Nk}) - \min(x_{1k}, x_{2k} \ldots x_{Nk})}, \tag{84}$$

jossa $\hat{x}_{ik}$ on k:s normalisoitu piirre x_{ik}:n ollessa piirteen alkuperäinen arvo. max(.) laskee koko opetusaineiston maksimin ja min(.) mimimin kyseessä olevalle piirteelle.

7.4. Luokittimien toteutus

7.4.1. LVQ-luokitin

LVQ-luokittimet toteutettiin pääasiassa käyttämällä LVQ-PAK-ohjelmistoa [91]. Ohjelma sisältää standardi-LVQ:n opetusalgoritmit: LVQ1, OLVQ1, LVQ2.1 ja LVQ3. Koodikirjan alustukseen ohjelma tarjoaa kaksi vaihtoehtoa. Ensimmäisessä vaihtoehdossa valitaan yhtä monta koodikirjavektoria luokkaa kohden, kun taas toisessa vaihtoehdossa käytetään luokkien prioritodennäköisyyksiä määrittämään luokkakohtaiset prototyyppivektorien määrät. Koodikirja alustustuksessa opetusnäytteitä verrataan koodikirjaan yksi kerrallaan valitsemalla opetusnäytteiksi ne piirrevektorit, jotka ovat KNN-menetelmän mukaan kyseisen luokan luokkarajojen sisällä.

Alustuksen jälkeen ohjelmassa on vielä mahdollisuus hienosäätää prototyyppivektorien määrää siten, että yksittäisiä koodikirjavektoreita lisätään tai poistetaan tarpeen mukaan. Tähän käytetään kaikkia opetusnäytteitä yhtenäistämällä kaikkien luokkien lyhimmät mediaanit luokan koodikirjavektorien suhteen. Tämän jälkeen koodikirjaa opetetaan edellä esitetyillä opetusalgoritmeilla. Tässä työssä koodikirjat alustettiin samalla määrällä koodikirjavektoreita luokkaa kohden, jonka jälkeen luokkien lyhimmät mediaanit yhtenäistettiin lisäämällä ja poistamalla yksittäisiä koodikirjavektoreita.

Piirrevalintaan ja kolmen askeleen tunnistukseen käytettävät LVQ-laajennukset toteutettiin Matlab-ympäristössä [87]. Automaattiseen piirteenvalintaan käytettävien LVQ-algoritmien toteutuksessa käytettiin lisäksi apuna Matlabin SOM Toolbox -sovellusta [92], jonka yhteyteen ne liitettiin.

7.4.2. HMM-luokitin

Ensimmäisessä vaiheessa HMM-luokittimen toteutukseen käytettiin HTK-toolkit -ohjelmistoa [71], joka sisältää vektorikvantisointiin perustuvan koodikirjan laskennan. Toisessa vaiheessa käytettiin Matlab-ympäristön Hidden Markov Model Toolbox -sovellusta [93]. LVQ-koodikirja opetettiin LVQ-PAK-ohjelmistolla [91], ja tämän jälkeen HMM-mallien laskenta toteutettiin matlab-ympäristössä.

HMM-malleja estimoitaessa käytössä on äärellinen määrä opetusdataa. Tämän vuoksi on mahdotonta, että kaikki mahdolliset havaintosekvenssit pystytään ku-

vaamaan, ja tällöin on mahdollista, että tietyn havainnon todennäköisyys $b_j(k)$ tilassa i menee nollaksi. Parannuksena voidaan tilahavaintotodennäköisyyksille asettaa rajoite, jolloin niitä ei päästetä tietyn kynnysarvon alapuolelle ja vältetään nolla todennäköisyydet mallissa. Tämä tapahtuu seuraavasti:

$$b_j(k) = \delta, \text{ jos } b_j(k) < \delta, \tag{85}$$

jossa δ on kynnysarvo ja se asetetaan $b_j(k)$:n uudeksi arvoksi, jos iteraatiossa laskettu $b_j(k)$:n arvo alittaa kynnysarvon. [64]

7.5. Tunnistustulokset

Seuraavassa esitetään askeltunnistuksen tuloksia. Aineistoa oli kerätty siis yhteensä yhdeltätoista eri henkilöltä. Tämän aineiston avulla oli tarkoitus testata valittujen menetelmien toimivuus. Ensimmäisessä vaiheessa 11 henkilön tunnistus yksittäisten askelien osalta toteutettiin käyttäen sekä oppivaa vektorikvantisointia että kätkettyjä Markov-malleja, jossa käytettiin LVQ-kvantisointia. Menetelmien toimivuuden lisäksi tarkoituksena oli yleensäkin selvittää, miten hyvin painelattialta saatavat askelprofiilit sopisivat henkilöiden tunnistukseen, eli olisivatko ne tarpeeksi erottuvia eri henkilöiden välillä ja toisaalta yhteneviä saman henkilön osalta sekä mahdollisesti löytää kuvaavat piirteet.

Vaikka ihmisen kävelytyylissä onkin paljon yksilöllisiä ominaisuuksia, voidaan eri henkilöiden askelprofiileita pitää samankaltaisina. Tämän vuoksi seuraavassa vaiheessa kokeiltiin, miten paljon testihenkilöiden määrä vaikuttaa tunnistustuloksiin, sillä on hyvin todennäköistä, että henkilöiden määrän kasvun myötä enemmän samankaltaisia askelprofiileita ilmenee. Tätä varten tehtiin älykotiskenaario, jossa tunnistettavien henkilöiden määrää pudotettiin viiteen ja tälle joukolle suoritettiin luokittelu. Testit suoritettiin myös LVQ:ta ja HMM-malleja käyttäen. Lisäksi LVQ:n avulla tarkkailtiin luokitteluvirheen kehitystä lähtien kahden henkilön tunnistuksesta ja edeten aina yhdentoista henkilön mallintamiseen.

Koska LVQ:lla saatiin lupaavia tuloksia yksittäisten askelien tunnistuksessa, menetelmää kehitettiin eteenpäin, jotta esimerkiksi tällainen 11 henkilön ryhmä voitaisiin tunnistaa luotettavammin. Lisäksi taustalla oli ajatus tunnistusjärjestelmän adaptiivisuudesta, jolloin voitaisiin tehdä esimerkiksi automaattista piirteiden painotusta kuvaavimpien piirteiden havaitsemiseksi, ja tunnistaa systeemille entuudesta tuntemattomat henkilöt. Näiden toteuttamiseksi 11 henkilön luokitteluun sovellettiin piirteiden automaattista valintamenetelmää DSLVQ sekä myös hylkäävää LVQ-luokitinta, joka koostui yhdestä sekä kolmesta peräkkäisestä askeleesta lopullisen tunnistamisen tekemiseksi. Menetelmien toimivuutta verrattiin standardi-LVQ-algoritmien tunnistustuloksiin.

11 henkilön tunnistukseen käytettiin noin 40 askelta jokaiselta henkilöltä, jotka oli pyritty nauhoittamaan tietyltä EMFi-liuskalta. Kuitenkin liuskan kapeus aiheutti sen, että joidenkin henkilöiden osalta askel jakaantui kahteen kanavaan. Tällaisissa tapauksissa askelsignaalit osat summattiin kyseiseltä aikaväliltä, jotta kokonainen askel saatiin tunnistukseen. Tämä aiheutti joissain tapauksissa muutosta signaalin muotoon varsinkin askeleen jakautuessa epätasaisesti kahdelle liuskalle.

Mallin valintaan, tarkkuuden arvioitiin ja algoritmien vertailemiseen käytettiin kahta eri menetelmää. Yksittäisten askelien tunnistuksessa aineisto jaettiin opetus- ja testiaineistoon, jolloin puhutaan holdout-menetelmästä (hold-out method) [94]. Tätä toistettiin 10 kertaa valitsemalla näytteet satunnaisesti opetus- ja testiaineistoon. Lisäksi LVQ-algoritmien vertailussa käytettiin k-osaista ristiin validointia (CV) [94], jossa aineisto jaettiin k:hon osaan ja opetettiin malli aina k-1 kappaleella aineiston osia ja testattiin jäljelle jäävällä osalla. Tätä toistettiin niin kauan, että kaikki mahdolliset osa-aineistojen kombinaatiot oli suoritettu. Lopulliset tunnistustulokset ilmoitetaan näiden useiden testien keskiarvona ja keskihajontana.

7.5.1. 11 henkilön yksittäiset askeleet

Ensimmäisessä vaiheessa LVQ-luokitinta kokeiltiin 11 henkilön tunnistukseen. Luokittelumallien rakentamiseen ja testaamiseen käytettiin yhteensä 427 askelta, joille laskettiin kohdassa 7.3.2 sekä liitteessä 1 esitetyt 13 piirrettä, ja ne normalisoitiin opetusaineiston avulla välille $[0, 1]$. Aineisto jaettiin satunnaisesti opetus- ja testiaineistoihin. LVQ-luokittimen opettamiseen käytettiin 2/3 opetusnäytteistä (n. 25 askelta / henkilö), ja loput käytettiin mallin testaamiseen. Erilaisia opetusalgoritmeja sekä opetusiteraatioiden määriä kokeiltiin.

LVQ-luokittelussa testattiin myös erikokoisia koodikirjoja. Niiden koot vaihtelivat 55:stä koodikirjavektorista (n. 5 vektoria / luokka) 200:aan (n. 18 vektoria / luokka) koodikirjavektoriin. Vaikka koodikirjavektorien määrä vaihtelikin paljon, tulokset olivat hyvin lähellä toisiaan. 18 koodikirjavektorin käyttö antoi keskimäärin parhaimman tuloksen. Kun koodikirjavektorien määrää kasvatettiin kohti opetusnäytteiden määrää, eivät tunnistustulokset enää parantuneet. Jokaisessa testissä luokat alustettiin samalla määrällä koodikirjavektoreita. Tämän jälkeen niiden määriä tasoitettiin, jotta jokaisella luokalla olisi yhtenäinen vaikutus koodikirjaan. Opetus aloitettiin OLVQ1-opetusalgoritmilla, jonka avulla saatiin alustetut luokkarajat. Tämän jälkeen opetusta jatkettiin sekä LVQ1- että LVQ3-algoritmeilla luokkarajojen hienosäätämiseksi.

Koodikirja, jossa oli 18 prototyyppivektoria luokkaan kohden antoi kymmenen satunnaisesti valitun riippumattoman testiaineiston mukaan kokonaistunnistuksen keskiarvoksi 67,37% keskihajonnan ollessa 5,53. 5 prototyyppivektoria käytettäessä tulokset olivat vastaavasti 66,60% (4,50). Muun kokoisilla koodikirjoilla saatiin vastaavasti tuloksia edellisten väliltä. Taulukossa 1 on esitetty konfuusiomatriisi tyypillisestä 11 henkilön LVQ-tunnistuksesta. LVQ-koodikirjaan valittiin 18 koodikirjavektoria luokkaa kohden ja opetettiin OLVQ1-, LVQ1- ja LVQ3-algoritmeilla. Tulokset on esitetty oikein luokiteltujen askelten prosenttiosuuksina.

Luokkakohtaiset tunnistusprosentit vaihtelevat paljon. Useiden henkilöiden kohdalla päästiin noin 90% tunnistukseen, kun taas joidenkin henkilöiden osalta ne jäivät alle 50%. Taulukossa 1 esimerkiksi henkilöiden 7, 10 ja 11 tunnistukset jäävät melko mataliksi. Tämä johtuu pääasiassa näytteiden huonosta laadusta, sillä näiden henkilöiden aineistossa oli paljon askelia, jotka olivat osuneet kuvan 24 tavoin kahdelle liuskalle, ja niiden muoto kärsi summausprosessissa.

HMM-mallien opettamisessa ja testaamisessa käytettiin myös 427 askelta. Niiden opetusta ja testaamista varten aineisto jaettiin kahteen osaan (n. 25 askelta

Taulukko 1. 11 henkilön LVQ-tunnistuksen konfuusiomatriisi. Ensimmäisessä sarakkeessa on tunnetut henkilöt. Riveillä on esitetty prosenttiosuudet yksittäisten askelten tunnistuksesta

	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10	H11
H1	92,31	7,69	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
H2	8,33	91,67	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
H3	0,00	0,00	66,67	0,00	8,33	0,00	0,00	25,00	0,00	0,00	0,00
H4	0,00	0,00	0,00	91,67	0,00	8,33	0,00	0,00	0,00	0,00	0,00
H5	0,00	0,00	0,00	7,69	69,23	7,69	15,38	0,00	0,00	0,00	0,00
H6	0,00	0,00	0,00	0,00	0,00	90,91	9,09	0,00	0,00	0,00	0,00
H7	0,00	0,00	0,00	0,00	16,67	0,00	41,97	16,67	8,33	8,33	8,33
H8	0,00	8,33	8,33	0,00	8,33	0,00	0,00	75,00	0,00	0,00	0,00
H9	0,00	0,00	0,00	0,00	0,00	0,00	10,00	0,00	90,00	0,00	0,00
H10	0,00	0,00	0,00	0,00	0,00	0,00	16,67	8,33	0,00	58,33	16,67
H11	0,00	0,00	0,00	8,33	16,67	16,67	8,33	0,00	0,00	0,00	50,00

opettamiseen ja n. 15 testaamiseen kaikissa luokissa) ja testit suoritettiin kymmenen satunnaisesti valittujen aineistojen mukaan. Jokaiselle segmentoidulle askeleelle laskettiin kohdassa 7.3.3 ja liitteessä 2 esitetyt piirteet. Tällöin yksittäiselle askeleelle saatiin havainnot, jotka koostuivat kolmen piirrevektorin sekvensseistä, jossa jokainen piirrevektori sisälsi 8 piirrettä ajansuhteen muuttuvasta askelsignaalista. Kaikki piirrevektorit normalisointiin opetusaineiston mukaan välille $[0, 1]$ ennen luokittelumallin opetusta. Normalisoidut piirrevektorisekvessit koodattiin LVQ-koodikirjan avulla kokonaislukusymboleiksi. Eri kokoisia koodikirjoja kokeiltiin, ja ne vaihtelivat 32:sta 256:een koodikirjavektoriin, Parhaat tunnistustulokset saavutettiin 128 vektorin kokoisella koodikirjalla, joka opetettiin käyttäen OLVQ1- ja LVQ1-algoritmeja.

Jokaiselle tunnistettavalle luokalle muodostettiin oma HMM-malli. Mallit olivat kolmitilaisia oikealta-vasemmalle-malleja, joissa tilansiirtotodennäköisyydet alustettiin satunnaisesti ja tilahavaintojakaumat alustettiin tasaisesti. Koska opetusaineistoa oli suhteellisen vähän, tilahavaintotodennäköisyydet rajoitettiin kynnysarvolla 0,001, sillä kaikkia HMM-mallin reittejä on mahdoton oppia. 11 henkilön osalta kokonaistunnistus oli 52,09% (3,67). Kun käytettiin suurempitilaisia malleja ja aikaikkunaan perustuvaa piirteenlaskentaa [14], 11 henkilön tunnistus jäi alle 40%. Taulukossa 2 on esitetty tyypillinen konfuusiomatriisi 11 henkilön HMM-tunnistuksesta. LVQ-koodikirjaan valittiin 128 koodikirjavektoria luokkaa kohden, ja opetettiin OLVQ1- ja LVQ1- algoritmeilla. Itse HMM-mallit olivat kolmitilaisia oikealta-vasemmalle-malleja.

7.5.2. 5 henkilön yksittäiset askeleet

11 henkilön tunnistuksen lisäksi tunnistettavien henkilöiden määrää pudotettiin viiteen. Tarkoituksena oli vertailla henkilöiden määrän vaikutusta tunnistustuloksiin ja mallintaa myös mahdollista älykotiskenaariota. Tällaisessa skenaariossa ei välttämättä tunnistettavien henkilöiden määrä tarvitse olla kovin suuri. Esimerkiksi viiden henkilön mallintaminen kuvaa hyvin keskikoisen perheen henkilöiden määrää. Aineistona käytettiin samaa kuin 11 henkilön tunnistuksessa ja

Taulukko 2. 11 henkilön HMM-tunnistuksen konfuusiomatriisi. Ensimmäisessä sarakkeessa on tunnetut henkilöt. Riveillä on esitetty prosenttiosuudet yksittäisten askelten tunnistuksesta

	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10	H11
H1	62,50	37,50	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
H2	0,00	100,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
H3	0,00	50,00	50,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
H4	0,00	0,00	0,00	87,50	0,00	0,00	0,00	0,00	0,00	12,50	0,00
H5	0,00	0,00	0,00	12,50	25,00	12,50	0,00	0,00	0,00	0,00	50,0
H6	0,00	0,00	0,00	0,00	0,00	88,9	0,00	0,00	0,00	0,00	11,11
H7	0,00	0,00	0,00	0,00	0,00	14,28	28,60	42,86	0,00	0,00	14,28
H8	0,00	0,00	0,00	12,50	0,00	0,00	25,00	50,00	12,50	0,00	0,00
H9	0,00	0,00	14,28	0,00	14,28	0,00	0,00	14,28	57,10	0,00	0,00
H10	0,00	0,00	0,00	0,00	0,00	14,28	28,57	0,00	14,28	42,90	0,00
H11	0,00	0,00	0,00	0,00	0,00	25,00	12,50	0,00	0,00	0,00	62,50

joukkoon valittin sekä miehiä että naisia. Aineistossa oli eri painoisia henkilöitä, joilla oli myös erilaisia jalkineita.

Luokittelumallit olivat samanlaisia kuin 11 henkilön kohdalla. LVQ-luokittimessa käytettiin 18 prototyyppivektoria luokkaa kohden ja opetettin OLVQ1-, LVQ1- ja LVQ3-algoritmeilla. Keskiarvoinen tunnistusprosentti oli 90,86% (4,45). Vastaavasti HMM-mallit olivat 3-tilaisia ja koodikirjan koko oli 128 prototyyppivektoria. Kokonaistunnistusprosentti HMM-luokittelussa 5 henkilön osalta oli 84,33% (4,66). Taulukossa 3 on esitetty tyypillinen 5 henkilön tunnistus LVQ-luokittimella toteutettuna ja vastaavasti taulukossa 4 on HMM-tunnistuksen konfuusiomatriisi viiden henkilön osalta.

Taulukko 3. 5 henkilön LVQ-tunnistuksen konfuusiomatriisi. Koodikirjan koko oli 18 koodikirjavektoria henkilöä kohden, ja sitä opetettiin OLVQ1, LVQ1 sekä LVQ3 algoritmeilla. Ensimmäisessä sarakkeessa on tunnetut henkilöt. Riveillä on esitetty prosenttiosuudet yksittäisten askelten tunnistuksesta

	Henkilö1	Henkilö2	Henkilö3	Henkilö4	Henkilö5
Henkilö1	92,31	0,00	7,69	0,00	0,00
Henkilö2	8,33	91,67	0,00	0,00	0,00
Henkilö3	0,00	0,00	91,67	8,33	0,00
Henkilö4	0,00	0,00	0,00	100,00	0,00
Henkilö5	10,00	0,00	0,00	0,00	90,00

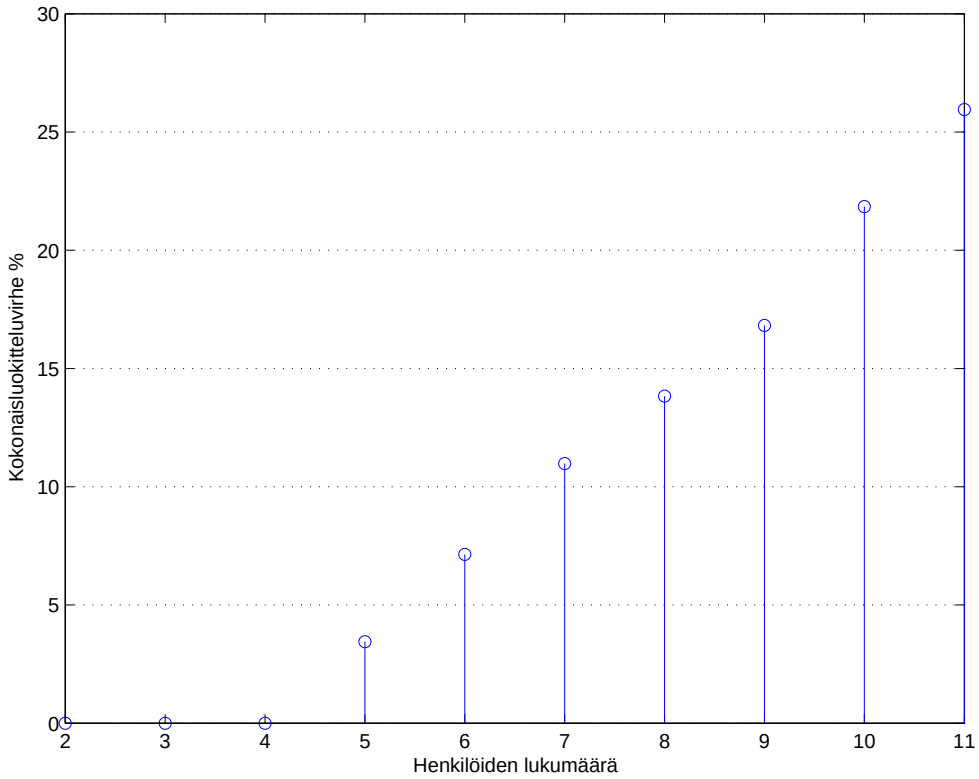
Molemmissa tapauksissa sekä luokkakohtaiset tunnistusprosentit että kokonaistunnistus paranivat. LVQ-luokittelussa saavutettiin jo melko luotettava tunnistus, sillä useimpien henkiöiden tunnitus on 100% lähellä. Luokkakohtaiset arvot eivät poikkea kovin paljoa toisistaan, kuten 11 henkilön tapauksessa. HMM-luokittelussa sitä vastoin joidenkin luokkien tunnistusprosentit jäävät alle 70%, eikä tunnitus ole näin vieläkään näiden henkilöiden osalta kovin luotettava.

Kuvassa 29 on esitetty LVQ-luokittimella henkilöiden määrän vaikutusta luokitteluvirheeseen. Kuvassa vaaka-akselilla on esitetty luokittelussa olleiden henkilöiden lukumäärä ja pystyakselilla niitä vastaavat kokonaistunnistusvirheet pro-

Taulukko 4. 5 henkilön HMM-tunnistuksen konfuusiomatriisi. LVQ-koodikirjan koko oli 128 koodikirjavektoria, jota opetettiin OLVQ1 sekä LVQ1 algoritmeilla. HMM-mallit olivat kolmitilaisia vasemmalta-oikealle-malleja. Ensimmäisessä sarakkeessa on tunnetut henkilöt. Riveillä on esitetty prosenttiosuudet yksittäisten askelten tunnistuksesta

	Henkilö1	Henkilö2	Henkilö3	Henkilö4	Henkilö5
Henkilö1	66,67	0,00	16,67	8,33	8,33
Henkilö2	7,69	92,31	0,00	0,00	0,00
Henkilö3	16,67	0,00	66,67	8,33	8,33
Henkilö4	0,00	0,00	0,00	100,00	0,00
Henkilö5	0,00	0,00	0,00	0,0	100,00

sentteina. Virheprosentit on saatu jakamalla aineisto satunnaisesti kahteen osaan. Aineistosta 2/3 käytettiin mallin opettamiseen ja 1/3 sen testaamiseen. Koodikirjan kokoa kasvatettiin aina, kun henkilöitä lisättiin siten, että 18 koodikirjavektoria henkilöä kohti pysyi vakiona. Koodikirjaa opetettiin OLVQ1-, LVQ1- ja LVQ3-algoritmeillä kuten aikaisemmissa testeissä. Kuvasta 29 voidaan nähdä, että henkilöiden lukumäärän kasvatus kasvattaa melko lineaarisesti luokittelun kokonaisvirhettä.



Kuva 29. Luokiteltujen henkilöiden lukumäärän vaikutus kokonaisluokitteluvirheeseen. Luokittelussa käytettiin oppivaa vektorikvantisointia.

Taulukko 5. 11 henkilön kokonaistunnistus käyttäen eri LVQ- algoritmeja. Tunnistustulos on keskiarvo 5-kertaisesta ristiin validoinnista. Keskihajonta on esitetty suluissa. N on aineistossa käytettyjen piirteiden lukumäärä

N	LVQ1	LVQ3	DSLVQ
13	66,8% (±5,4)	67,4% (±5,4)	70,2% (±5,7)
31	65,8% (±5,0)	66,5% (±4,7)	69,4% (±6,4)

7.5.3. LVQ-laajennukset

Seuraavassa on esitelty LVQ-luokittelutuloksia, jotka saatiin laajentamalla yksittäisten askelten tunnistusta. Automaattista piirteenvalinta menetelmää DSLVQ käytettiin 11 henkilön luokittelussa [13]. Lisäksi LVQ-luokittimelle opetettiin koodikirjan lisäksi hylkäyskriteerit, joilla epäluotettavat näytteet voidaan hylätä kuulumattomana mihinkään luokkaan. Hylkäysoptiota laajennettiin vielä tapaukseen, jossa luokitteluun käytetään saman henkilön kolmen askeleen yhteistunnistusta [12]. Näiden menetelmien avulla luokittelua saadaan luotettavammaksi, koska huonolaatuiset näytteet voidaan hylätä.

7.5.4. Automaattinen piirteenvalinta

Automaattisella piirteenvalinnalla saadaan lisättyä askeltunnistuksen adaptiivisuutta. Tämä tapahtuu käyttäemällä DSLVQ-menetelmää, joka laajentaa standardi-LVQ-opetusalgoritmeja. LVQ-opetuksen aikana menetelmä skaalaa piirteitä siten, että informatiiviset piirteet saavat suurempia skaalauskertoimia kuin ei-informatiiviset. Näin voidaan automaattisesti valita luokittelutilanteseen kuvaavimmat piirteet painotettua etäisyysmittaa laskettaessa.

Menetelmää verrattiin muihin LVQ-algoritmeihin 11 henkilön aineistolla. Testeissä käytettiin sekä 13 että 31 piirrettä. Koodikirjoissa käytettiin 18 prototyyppivektoria luokkaa kohden, ja ne kaikki alustettiin OLVQ1-opetuksella. Tämän jälkeen opetusta jatkettiin LVQ1-, LVQ3- ja DSLVQ-algoritmeilla. Testeissä käytettiin samoja aineistoja kuin edellä. Testit suoritettiin käyttäen 5-osaista ristiin validointia. Taulukossa 5 on esitetty kokonaistunnitustulokset, jotka ovat ristiinvalidoinnin keskiarvoja. Suluissa on esitetty CV:n keskihajonnat. Testit suoritettiin siis kaikille eri opetusalgoritmeille kahdella eri piirrejoukolla, jotka normalisoitiin välille $[0, 1]$.

Taulukosta 5 nähdään, että DSLVQ-algoritmi parantaa kokonaistunnistusta molemmilla piirrejoukolla. Se pystyy skaalaamaan hyvin valitun piirrejoukon ($N = 13$) piirteitä, jotka oli valittu käyttäen KNN-luokitinta ja eteenpäin-taaksepäin-hakua. Kun käytetään alkuperäistä piirrejoukkoa ($N = 31$), jossa on paljon korreloivia piirteitä, DSLVQ pystyy valitsemaan kaikista merkittävämmät piirteet ja antaa parhaimman tunnistustuloksen muihin LVQ-algoritmeihin verrattuna.

7.5.5. Hylkäysoptio ja 3 askeleen yhteistunnistus

Hylkäysoption avulla saadaan parannettua tunnistustuloksia ja lisättyä luotettavuutta hylkäämällä epäluotettavia näytteitä. Hylkäysoptio käyttää kahta eri kynnysarvoa hylkäämiseen. Se hylkää näytteet, jotka ovat kaukana kaikista opetetuista prototyyppivektoreista sekä näytteet, jotka ovat kahden tai useamman luokan päätösrajalla, eikä niitä voida luotettavasti luokitella mihinkään luokkaan. Aluksi hylkäysoptiota sovellettiin yksittäisten askelten luokitteluun. Lisäksi käytettiin menetelmää, jossa ensimmäisellä tasolla toteutettiin yksittäisten askelien luokittelu ja toisella tasolla tehtiin lopullinen päätös käyttäen henkilön kolmea peräkkäistä askelta.

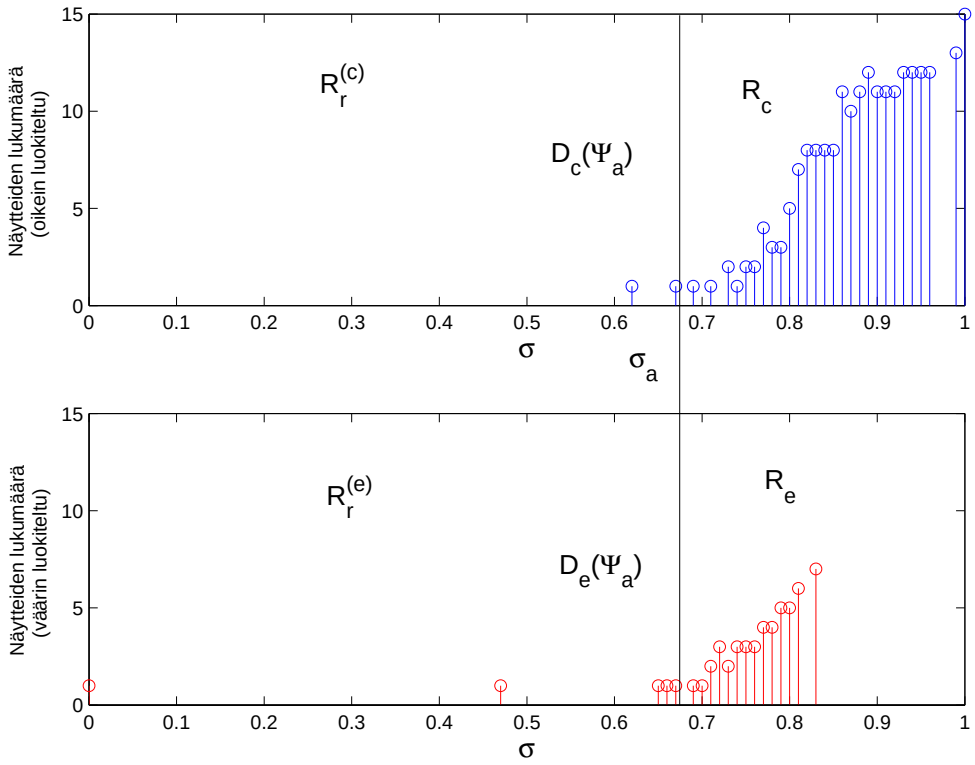
Hylkäysoption testaamiseen käytettiin samaa aineistoa kuin aikaisemminkin 11 henkilön osalta. Se jaettiin satunnaisesti opetus- ja testiaineistoihin, joissa 2/3 käytettiin mallien opettamiseen ja 1/3 niiden testaamiseen. Piirrevektorit normalisoitiin välille $[0, 1]$. LVQ-koodikirjoissa käytettiin 7 prototyyppivektoria luokkaa kohden ja ne opetettiin OLVQ1- ja LVQ1-algoritmeilla.

Hylkäyskriteerit määrättiin opetusaineston avulla. Kustannusarvo oikein luokitelluille näytteille kiinnitettiin ykköseksi ja väärinluokiteltujen ja hylättävien näytteiden kustannusarvoja muutettiin ja etsittin sopivat testaamalla erilaisia vaihtoehtoja. Kuvassa 30 on esitetty tyypillinen kävelyaskelaineiston esiintymistiheysjakauma luotettavuusevaluaattorin Ψ_a mukaan. Tässä kuvassa kynnysarvon σ_a alittavat näytteet hylätään, koska ne ovat liian kaukana luokkien prototyyppivektoreista. Vastaavasti kuvassa 31 on tilanne, jossa kynnysarvon σ_b alittavat näytteet hylätään luokkien päällekkäisiltä alueilta.

Taulukossa 6 on esitetty 10 satunnaisesti valitun aineiston parhaat kustannusarvot (C_e, C_r), luokittelu- ja hylkäystulokset sekä keskiarvo kussakin tilanteessa. Taulukossa on tulokset kolmen peräkkäisen askeleen tunnistuksesta sekä yksittäisten askelten tunnistus toteutettuna hylkäävällä luokittimella ja ilman sitä. Hylkäyskriteerien avulla tunnistuksen luotettavuutta saatiin parannettua ja luokitteluvirhe pieneni. Yksittäisten askelien tunnistuksessa saatiin kokonaistunnistusta parannettua noin 5 prosenttiyksikön verran 0-hylkäysluokittimeen verrattuna, kun näytteistä hylättiin noin 10%. Kolmen askeleen tunnistuksessa päästään jo luotettavaan tunnistukseen sen ollessa noin 90%, kun 20% kolmen askeleen sekvensseistä hylättiin. Väärinluokiteltavien ja hylättävien näytteiden kustannusarvojen suhde pidettiin pienenä, ettei näytteitä hylättäisi liikaa. Askelluokittelussa luokkien päällekkäisyys on voimakasta, sillä käyttämällä esimerkiksi kustannusarvoja (C_e, C_r)=(3,2), noin 65% testinäytteistä hylättiin. Keskimäärin parhaimmat tulokset saavutettiin käyttäen kustannusarvoja (C_e, C_r)=(15,14).

7.6. Tulosten arviointi

LVQ-luokitin osoittautui HMM-luokitinta paremmaksi sekä 11 että 5 henkilön tunnistuksessa. 11 henkilön tunnistuksessa LVQ antoi noin 70% kokonaistunnistuksen, kun taas HMM-malliella jäätiin noin 50% tunnistuksiin. Myös yksittäisten henkilöiden tunnistusosuudet olivat parempia LVQ:lla, sillä HMM tunnistuksessa joidenkin henkilöiden luokittelut jäivät alle 30%. Myös LVQ:ssa osa luokista jäi alle 50%, eikä tunnitusta voida niiden osalta pitää kovin luotettavana. Luokkakohtaiset erot johtuvat pääasiassa askelten jakaantumisesta kahdelle liuskalle,



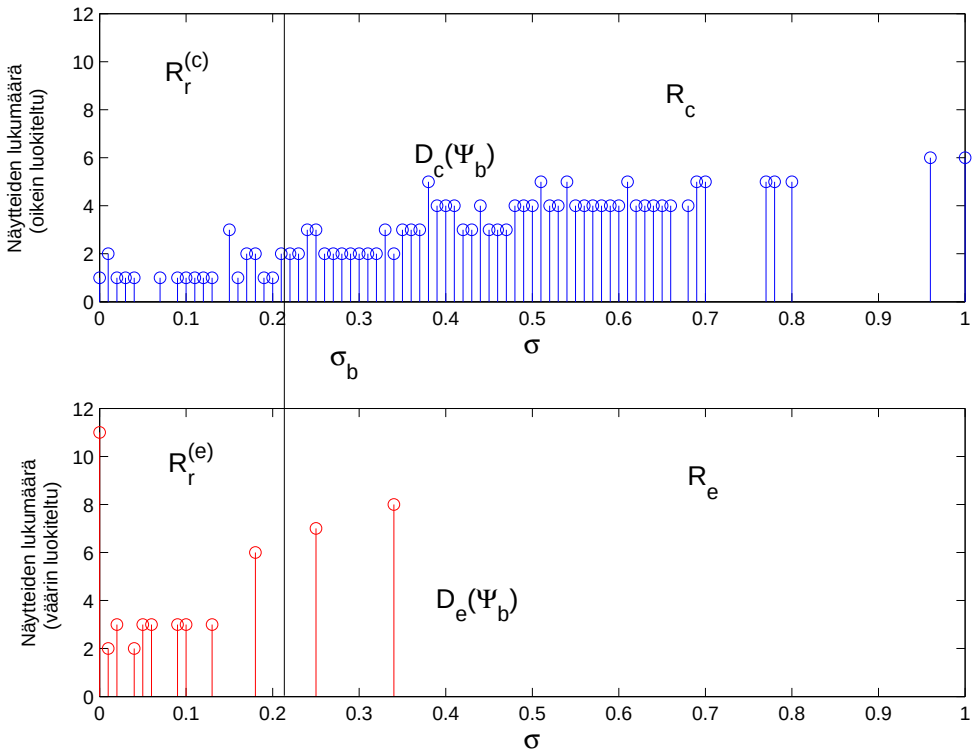
Kuva 30. Esimerkki esiintymistiheysfunktioista $D_c(\Psi_a)$ and $D_e(\Psi_a)$ oikein ja väärin luokitellulle askeldatalle käyttäen luotettavuusevaluaattoria Ψ_a. $R_r^{(c)}$ ja $R_r^{(e)}$ ovat hylkäysprosentit oikein ja väärin luokitelluille näytteille annetun kynnysarvon σ_a mukaan, kun taas R_c ja R_e ovat hyväksyttyjen prosenttiosuudet hylkäysluokittimella.

sillä niiden muoto kärsi summausprosessissa.

Näissä testeissä aineiston määrä luokkaa kohden oli melko rajallinen. Suuremmalla määrällä opetusnäytteitä mallit voitaisiin saada stabiilimmiksi. HMM-mallit näyttivät kärsivän enemmän datan vähyydestä, sillä ne ovat monimutkaisempia tilastollisia malleja, joissa on yleensä paljon vapausasteita, ja saattaisivat toimia paremmin suuremmalla määrällä opetusdataa. LVQ on sitävastoin yksinkertainen menetelmä, jossa päätöspinnat muodostetaan suoraan piirreavaruuteen. Reaaliaikaista toteutusta ajatellen opetusnäytteitä on kuitenkin yleensä hyvin rajattu määrä, jolloin yksinkertaisen nopeasti oppivan mallin, kuten LVQ, käyttöä voidaan pitää järkevänä.

Henkilöiden määrällä on suuri vaikutus luokittelun onnistumiseen. 5 henkilön LVQ-luokittelussa saadaan jo todella luotettavia tuloksia kokonaistunnistuksen ollessa noin 90%. Myöskin HMM-luokittelulla saavutettiin parempia tuloksia kokonaistunnistuksen ollessa noin 80%. Älykotisovelluksessa tälläisen pienemmän henkilömäärän mallintaminen voisi olla jo riittävä.

Kun vertaillaan erilaisia LVQ-luokittimia, koodikirjan koolla ei havaittu suuria eroja. Tällöin pieniä koodikirjoja on syytä käyttää, jos laskenta-aika itse tunnistuksessa halutaan saada mahdollisimaan lyhyeksi. LVQ-laajennukset myöskin paransivat luokittelutuloksia. DSLVQ nosti hieman kokonaistunnistusta 11 henkilön osalta ja tarjosi mahdollisuuden automaattiseen piirteiden valitsemiseen. Lisäksi kolmen askeleen hylkäävä LVQ-luokitin tarjosi luotettavan noin 90% tunnistuksen myös 11 henkilön mallintamisessa.



Kuva 31. Esimerkki esiintymistiheysfunktioista $D_c(\Psi_b)$ and $D_e(\Psi_b)$ oikein ja väärin luokitellulle askeldatalle käyttäen luotettavuusevaluaattoria Ψ_b. $R_r^{(c)}$ ja ovat $R_r^{(e)}$ ovat hylkäysprosentit oikein ja väärin luokitelluille näytteille annetun kynnysarvon σ_a mukaan, kun taas R_c ja R_e ovat hyväksyttyjen prosenttiosuudet hylkäysluokittimella.

Reaaliaikasovellusta ajatellen automaattinen piirteenvalinta sekä hylkäävä luokitin voivat yhdessä tarjota adaptiivisuutta käytettäviin menetelmiin. Hylkäävän luokittimen avulla voidaan mahdollisesti automaattisesti havaita uudet systeemille tuntemattomat henkilöt ja opettaa tämän jälkeen automaattisesti uusi luokitin. Myös kuvaavat piirteet saattavat muuttua uusien henkilöiden myötä. Tällöin DSLVQ voi tarjota automaattisen skaalausmenetelmän uutta luokitinta opetettaessa.

7.7. Avoimia kysymyksiä ja jatkokehitys

Kolmen askeleen yhteistunnistuksessa päästiin melko luotettaviin kokonaistunnistustuloksiin, joita on mahdollista käyttää hyväksi rakennettaessa itsestään tietoista ja vuorovaikutteista ympäristöä. Kuitenkin tuloksiin sisältyy aina epävarmuutta ja käytettäviin menetelmiin ja järjestelmiin ongelmia. Eräs suuri ongelma on toteutuksen yhteydessä, kohdassa 7.2.2 esitetty askelsignaalin jakautuminen usealle EMFi-liuskalle. Tässä tutkimuksessa pyrittiin keräämään yksittäiselle liuskalle osuneita askelia. Kuitenkin liuskan kapeudesta johtuen signaali jakautui useissa tapauksissa, ja on väistämätöntä, että tällaisia tilanteita tulee myös reaaliaikatoteutuksessa ihmisten liikkuessa vapaasti lattialla. Tässä tutkimuksessa jakaantunut signaali summattiin, mutta sen todettiin heikentävän kokonaisen askeleen muotoa, sillä signaalin jakautuminen on todella epälineaarista.

Taulukko 6. 10 satunnaisesti valitun testiaineiston tunnistustulokset yhdelletoista kävelijälle. Väärin luokiteltujen C_e ja hylättyjen C_r näytteiden kustannukset on valittu seuraavasti: (2,1),(4,3),(6,5),(10,9),(15,14) and (-) (ilman hylkäystä ensimmäisellä tasolla). Kustannusarvo C_c on kiinnitetty ykköseksi. Ensimmäisessä sarakkeessa on kunkin testikierroksen parhaimman tunnistustuloksen antavat kustannukset. 2. ja 3. sarake ilmoittavat parhaimmat tunnistustulokset ja niitä vastaavan hylkäysprosentin, kun tunnistuksessa käytetään kolmea peräkkäistä askelta. 4. ja 5. sarake esittävät vastaavat arvo, kun luokittelussa käytettiin yksittäistä askelta. Viimeinen sarake ilmoittaa kokonaistunnistusprosentin 0-hylkäys LVQ-luokittimille

Aineisto	(C_e, C_r)	tunnistus 3 askel	hylkäys 3 askel	tunnistus 1 askel	hylkäys 1 askel	0-hylkäys luokittelija 1 askel
1.	(-)	93.2	14.3	66.4	0.0	66.4
2.	(6,5)	84.8	21.4	67.8	10.7	67.2
3.	(15,14)	87.9	14.3	74.2	22.1	66.4
4.	(15,14)	88.6	9.5	75.7	21.4	70.2
5.	(-)	87.9	16.7	62.6	0.0	62.6
6.	(15,14)	100	21.4	75.4	14.5	70.2
7.	(-)	86.4	11.2	67.2	0.0	67.2
8.	(15,14	90.2	11.9	70.9	1.5	69.5
9.	(6,5)	95.5	30.9	69.2	10.7	68.0
10.	(15,14)	75.0	28.6	59.8	11.5	58.0
Keskiarvo		89.0%	18.0%	68.9%	9.2%	66.6%

Tämä on suuri ongelma juuri tunnistamisen kannalta, koska mentelmät vaativat hyvän askelprofiilin. Sitä vastoin henkilön paikantaminen lattialta voidaan toteuttaa hyvin myös jakaantuneista askelista.

Signaaliosien yhdistämiseen on vaikea rakentaa yhtenäistä menetelmää, sillä kun lattialla liikutaan vapaasti erilaisten osumien määrä kasvaa äärettömän suureksi. Toisaalta sidottaessa tunnistus tilanteeseen, jossa henkilö astuu huoneeseen, on kävely tällöin lattiasensorien suunnassa suoraviivaista, ja osien yhdistämiseen on mahdollista kehittää menetelmiä. Toinen vaihtoehto on tyytyä valitsemaan luokitteluun vain ne askeleet, jotka ovat osuneet tarpeeksi hyvin yhdelle liuskalle. Tämä onnistuu kohdassa 7.2.1 esitetyllä SSMM-menetelmällä, tällöin kuitenkin käyttäjän tunnistaminen viivästyy, sillä hyvien askelten muodostumista täytyy odottaa kauemmin.

Järjestelmän adaptiivisuus on eräs seikka, joka tulee kysymykseen jatkokehityksen kannalta, kun todellisia sovelluksia kehitetään. Järjestelmällä tulisi olla kyky pieniin muutoksiin, sillä ihmisen kävelytyyli tai paino voi muuttua ajan kuluessa. Lisäksi erilaisten jalkineiden käyttö voi muuttaa askelprofiilia. Kun testihenkilön luokittelumalli opetetettiin tiettyä jalkinetta käyttäen, tämän henkilön tunnistaminen ilman jalkinetta oli heikkoa. Lisäksi kävelyaskeleen mallissa käytetyt piirteet ovat riippuvaisia kävelynopeudesta. Tämän vuoksi tunnistus voi hyvinkin heiketä, jos käyttäjä kävelee eri nopeudella kuin luokittelumalliin käytetyissä opetusnäytteissä.

Tässä työssä esitetyt LVQ-laajennukset ovat yksi ratkaisu kohti adaptiivisuutta. DSLVQ:n avulla systeemi voi löytää kuvaavimmat piirteet ja painottaa niitä

enemmän. Tällöin esimerkiksi, jos henkilön askelprofiili muuttuu hieman, voi siitä löytyä uusia kuvaavampia piirteitä, jotka voidaan havaita uutta luokittelijaa opetettaessa. Toisaalta hylkäävä luokitin edesauttaa huonojen askelprofiilien havaitsemista yhdessä SSMM-menetelmän kanssa, jolloin tunnistukseen voidaan kerätä lisää parempia askelia tai todeta henkilö tuntemattomaksi järjestelmälle, ja aloittaa opetusaineiston kerääminen uuden askelmallin luomiseksi. Myös luonnollisen kävelytyylin aikaansaaminen opetustilanteessa voi olla vaikeaa, eikä aineisto vastaa aina koehenkilön oikeita ominaisuuksia. Tästä johtuen opetusaineiston kerääminen pitäisi tapahtua koehenkilön tietämättä, mikä on kuitenkin käytännössä vaikea järjestää.

Koska biometrisillä, ja erityisesti ihmisen kävelytyyliin ja askelmuotoon perustuvilla tunnistusmenetelmillä on mahdoton päästä täydelliseen 100% tunnistukseen, erilaisten sensorien yhdistelemistä voitaisiin käyttää tunnistuksen parantamiseen. Esimerkiksi fuusioimalla henkilön kävelytyyliä tai kasvonpiirteitä mallintava kamerajärjestelmä yhdessä lattia-anturin kanssa voitaisiin saavuttaa paljon luotettavampi tunnistus suuremmalle määrälle käyttäjiä, kuin mitä tässä työssä käytettiin.

8. YHTEENVETO

Paineenmuutoksia aistivalla EMFi-lattia-anturilla voidaan toteuttaa henkilön askelmuotoon perustuva henkilöllisyyden tunnistus pienelle ihmisjoukolle. Tällainen sensorijärjestelmä on käyttäjälle läpinäkyvä ja tarjoaa älykkään ympäristön vaatimusten mukaan luonnollisen menetelmän tunnistamiseen.

Työssä tutkittiin ja kehitettiin menetelmiä kävelyaskeleen tuottaman painesignaalin luokitteluun kävelijän henkilöllisyyden tunnistamiseksi, sillä EMFi-anturia ei ole aikaisemmin sovellettu askelprofiilin identifiointiin. Menetelmät perustuivat askelsignaalista laskettaviin piirteisiin, joiden on tarkoitus kuvata henkilön askelprofiilin ominaisuudet. Itse luokittelumenetelminä käytettiin opetettujen prototyyppivektorien ja tuntemattomien näytteiden etäisyyteen perustuvaa LVQ:a, jolla tunnistettiin yksittäisiä askelia sekä saman henkilön kolmen askeleen ryhmiä. Lisäksi yksittäisten askelten tunnistamiseen kokeiltiin todennäköisyyslaskentaan ja tilamalleihin pohjautuvia HMM-malleja.

LVQ-luokittelussa yksittäiselle askeleelle laskettiin paikkatietoon sekä taajuusesitykseen perustuvia piirteitä. Opetusaineiston avulla opetettiin koodikirja luokittelua varten. Lisäksi LVQ-algoritmeja laajennettiin siten, että koodikirjan opetukseen lisättiin automaattinen piirteenskaalausmenetelmä DSLVQ, joka perustuu hyvien yksittäisten piirteiden havaitsemiseen ja niiden painottamiseen luokittelun etäisyysmittaa laskettaessa. Koska askelsignaalin muoto kärsii sen jakaantuessa useammalle kuin yhdelle sensoriliuskalle, LVQ-tunnistukselle määritettiin myös hylkäysoptio, jonka avulla epäluotettavat näytteet voidaan hylätä. Tätä ominaisuutta sovellettiin 2-tasoiseen askeltunnistimeen, jossa tunnistukseen käytetään henkilön kolmea peräkkäistä askelta. Menetelmän avulla saatiin parannettua askeltunnistimen luotettavuutta ja pienenettyä luokitteluvirhettä. Menetelmällä voidaan mahdollisesti havaita järjestelmälle tuntemattomat käyttäjät automaattisesti, mikä lisää järjestelmän adaptiivisuutta.

11 henkilön yksittäisten askelien tunnistuksessa saavutettiin 67% kokonaistunnistus, kun koodikirjan opetus suoritettiin DSLVQ:n avulla päästiin noin 70% tunnistukseen. Lisäksi menetelmä kykeni valitsemaan suuresta 31 piirteen joukosta informatiivisimmat piirteet. 2-tasoisella luokittimella, jossa käytettiin kolmea peräkkäistä askelta sekä hylkäys mahdollisuutta, saavutettiin jo paljon luotettavampia tuloksia kokonaistunnistuksen ollessa 11 henkilön osalta noin 90%, kun 20% kolmen askeleen ryhmistä hylättiin.

HMM-luokittelussa käytettiin diskreettejä oikealta-vasemmalle-malleja, jotka olivat 3 tilaisia. Käytetyt piirteet olivat vastaavia paikkatietoon perustuvia, kuten LVQ:ssa, mutta malli rakennettiin kolmen piirrevektorin sekvenssinä. Nämä piirrevektorit valittiin askeleen ajansuhteen muuttuvina ominaisuuksina. Esimerkiksi ensimmäinen piirrevektori sisälsi piirteitä askeleen alusta eli kantapään vaikutuksesta lattiaan. Piirrevektorisekvenssit kvantisointiin opetettua LVQ-koodikirjaa käyttäen kokonaislukusymboleiksi, joiden pohjalta HMM-mallit rakennettiin. HMM-luokittelu ei toiminut yhtä hyvin kuin LVQ 11 henkilön kokonaistunnistuksen ollessa 52%.

Koska ihmisen askelprofiilit ovat samankaltaisia, on mahdollista että tunnistettavien henkilöiden määrän kasvaessa myös samankaltaisia askelprofiileja ilmenee. Tämän vuoksi tunnistusta testattiin myös pienemmällä henkilö määrällä. Esimerkiksi 5 henkilöä voisi olla älykotiskenaariota ajatellen hyvinkin sopiva määrä. Tällöin tunnistustulokset olivat yksittäisille askelille jo huomattavasti luotettavampia niiden ollessa LVQ-tunnistuksessa 91% ja HMM-malleja käytettäessä

84%. Monen kävelyaskeleen yhdistävässä LVQ-tunnistuksessa saatiin luotettavia ja lupaavia tuloksia, ja sitä voidaan hyvinkin soveltaa osana älykästä ympäristöä joustavan ja luonnollisen henkilöprofiilin määrittämiseen käyttäen ympäristöön sulautettua ja piilotettua EMFi-lattiaa. Systeemi voi jatkossa toimia korkeamman tason sovellusten, kuten henkilön aktiviteetin tunnistuksen ja rutiinien oppimisen perustana.

9. LÄHTEET

[1] Aware home (23.05.2004). URL: http://www.cc.gatech.edu/fce/ahri/.

[2] Ambient intelligence (23.05.2004). URL: http://www.research.philips.com.

[3] Brummit B., Meyers B., Krumm J., Kern A. & Shafer S. (2000) Easyliving: Technologies for intelligent environments. In: *2nd International Symposium of Handheld and Ubiquitous Computing (HUC 2000)*, Springer-Verlag, New York, USA, s. 12–29.

[4] MIT's oxygen project (23.05.2004). URL: http://oxygen.lcs.mit.edu/.

[5] Weiser M. (1991) The computer for the 21st century. Scientific American 265, September 1991, s. 66–75.

[6] Essa I.A. (2000) Ubiquitous sensing for smart and aware environments: Technologies towards the building of an aware home. IEEE Personal Communications, October 2000, Special issue on networking the physical world, s. 47–49.

[7] Pentland A. (1996) Smart rooms. Scientific American 274, s. 68–76.

[8] Orr R. & Abowd G. (2000) The smart floor: A mechanism for natural user identification and tracking. In: *Proceedings of 2000 Conf. Human Factors in Computing Systems (CHI 2000)*, ACM Press, The Hague, Netherlands.

[9] Addlesee M., Jones A., Livesey F. & Samaria F. (1997) ORL active floor. IEEE Personal Communications 4, s. 35–41.

[10] Yun J.S., Lee S.H., Woo W.T. & Ryu J.H. (2003) The user identification system using walking pattern over the ubifloor. In: *Proceedings of International Conference on Control, Automation, and Systems (ICCAS 2003)*, Gyeongju, Korea.

[11] Pirttikangas S., Suutala J., Riekki J. & Röning J. (2003) Learning vector quantization in footstep identification. In: Hamza M., editor, *Proceedings of 3rd IASTED International Conference on Artificial Intelligence and Applications (AIA 2003)*, IASTED, ACTA Press, Benalmadena, Spain, s. 413–417.

[12] Suutala J., Pirttikangas S., Riekki J. & Röning J. (2004) Reject-optional LVQ-based two-level classifier to improve reliability in footstep identification. In: Ferscha A. & Mattern F., editors, *Proceedings of 2nd International Conference on Pervasive Computing (PERVASIVE 2004)*, Springer-Verlag, Linz/Vienna, Austria, Vol. 3001 of *Lecture Notes in Computer Science*, s. 182–187.

[13] Suutala J. & Röning J. (2004) Towards the adaptive identification of walkers: Automated feature selection of footsteps using distinction-sensitive LVQ. In: *Proceedings of International Workshop on Processing Sensory Information for Proactive Systems (PSIPS 2004)*, Oulu, Finland, in press.

[14] Pirttikangas S., Suutala J., Riekki J. & Röning J. (2003) Footstep identification from pressure signals using Hidden Markov Models. In: Huttunen H., Gotchev A. & Vasilache A., editors, *Proceedings of Finnish Signal Processing Symposium (FINSIG 2003)*, Tampere, Finland, s. 124–128.

[15] Koho K., Suutala J., Seppänen T. & Röning J. (2004) Footstep pattern matching from the pressure signals. In: *Proceedings of 12th European Signal Processing Conference (EUSIPCO 2004)*, Vienna, Austria, in press.

[16] Tuulari E. (2003) Älykäs ympäristö jokaisen ulottuville. VTT Elektroniikka - Asiakaslehti 1/2003 URL: http://www.vtt.fi/ele/tuloksia/aslehti1_2003/ (saatavilla 23.05.2004).

[17] Coen M.H. (1998) Design principles for intelligent environment. In: *Proceedings of the 15th National Conference on Artificial Intelligent (AAAI98)*, Madison, Wisconsin, USA, s. 547–554.

[18] Krumm J., Shafer S. & Wilson A. (2001) How a smart environment can use perception. In: *UBICOMP 2001 Workshop on Perception for Ubiquitous Computing*, Atlanta, Georgia, USA.

[19] Pentland A. (2000) Perceptual intelligence. Communications of the ACM 43, s. 35–44.

[20] Pentland A. (2000) Looking at people: Sensing for ubiquitous and wearable computing. IEEE Transactions on Pattern Analysis and Machine Intelligence 22, s. 107–119.

[21] Aggarwal J.K. & Cai Q. (1999) Human motion analysis: A review. Computer Vision and Image Understanding 73, s. 428–440.

[22] Essa I.A. (1999) Computers seeing people. AI Magazine 20, s. 69–82.

[23] Silverman H., Patterson W. & Flanagan J. (1998) The huge microphone array. IEEE concurrency 6, s. 36–46.

[24] Clarkson B., Mase K. & Pentland A. (2000) Recognizing user's context from wearable sensors: Baseline system. Technical report, Vismod Tecnical Report 519, Summer 2000, 5 s.

[25] Ekahau (2002) Ekahau positioning engine 2.0. Technical report, Ekahau, Inc., Technology White Paper, 19 s.

[26] Mäntyjärvi J., Himberg J. & Seppänen T. (2001) Recognizing human motion with multiple acceleration sensors. In: *Proceedings of the International IEEE Conference on Systems, Man, and Cybernetics (SMC 2001)*, Tuscon, USA, s. 747–752.

[27] Want R. & Hopper A. (1992) Active badges and personal interactive computing objects. IEEE Transactions on Consumer Electronics 38, s. 10–20.

[28] Werb J. & Lanzl C. (1998) Designing a positioning system for finding things and people indoors. IEEE Spectrum 9, s. 71–78.

[29] Cattin P. (2002) Biometric authentication system using human gait. Ph.D. thesis, ETH-Zürich, Institute of Robotics, Switzerland, 124 s.

[30] Köhle M. & Merk D. (1997) Clinical gait analysis by neural networks: Issues and experiences. In: *Proceedings of the IEEE Symposium on Computer-Based Medical Systems*, Maribor, Slovenia, s. 138–143.

[31] Kistler force plate (23.05.2004). URL: http://www.kistler.com.

[32] Headon R. & Curwen R. (2001) Recognizing movements from the ground reaction force. In: *Proceedings of the 2001 Workshop on Perceptive User Interfaces*, Orlando, Florida, USA.

[33] Honz Z., Slivovsky L. & Pentland A. (2001) A sensing chair using pressure distribution sensors. IEEE Transactions on Mechatronics 6, s. 261–268.

[34] Ivanov B., Ruser H. & Kellner M. (2002) Presence detection and person identification in smart homes. In: *International Conference Sensors and Systems*, State Technical University Saint-Petersburg, Russia, s. 80–85.

[35] Jain A., Hong L. & Pankanti S. (2000) Biometric identification. Communications of the ACM 43, s. 91–98.

[36] Murray M., Dought A. & Kory R. (1964) Walking patterns of normal men. Journal of Bone and Joint Surgery 46, s. 335–360.

[37] Cutting J. & Kozlowski L. (1977) Recognizing friends by their walk: Gait perception without familiarity cues. Bulletin of the Psychonomic Society 9, s. 353–356.

[38] Little J. & Boyd J. (1998) Recognizing people by their gait: the shape of motion. MIT Press Journal Videre 1, s. 1–32.

[39] Murase H. & Sakai R. (1996) Moving object recognition in eigenspace presentation: Gait analysis and lip reading. Pattern Recognition Letters 17, s. 155–162.

[40] Huang P., Harris C. & Nixon M. (1998) Comparing different template features for recognizing people by their gait. In: Carter J. & Nixon M., editors, *Proceedings of the British Machine Vision Conference 1998 (BMVC 1998)*, British Machine Vision Association, Southamton, UK.

[41] Kale A., Rajagopalan A., Cuntoor N. & Krüger V. (2002) Gait based recognition of humans using continuous HMMs. In: *Proceedings of the International Conference on Face and Gesture Recognition*, Washington DC, USA, s. 321–326.

[42] BenAbdelkader C., Cutler R. & Davi L. (2002) Person identification using automatic height and stride estimation. In: *Proceedings of International Conference on Pattern Recognition (ICPR 2002)*, Quebec, Canada.

[43] Kale A., RoyChowdhury A. & Chellappa R. (2004) Fusion of gait and face for human identification. In: *International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2004)*, Montreal, Canada.

[44] Mozer M. (1998) The neural network house: An environment that adapts to its inhabitants. In: Coen M., editor, *Proceedings of the American Association for Artificial Intelligence Spring Symposium on Intelligent Environments*, Menlo Park, California, USA, s. 110–114.

[45] Johanson B., Fox A. & Winograd T. (2002) The interactive workspaces project: Experiments with ubiquitous computing rooms. IEEE Pervasive Computing Magazine 1, s. 71–78.

[46] Abowd G. (1999) Classroom 2000: An experiment with the instrumentation of a living educational environment. IBM Systems Journal, Special issue on Pervasive Computing 38, s. 508–530.

[47] Bobick A., Intille S. & Davis J. (1999) The kidsroom: A perceptually-based interactive and immersive story environment. Teleoperators and Virtual Environments 8, s. 367–391.

[48] Paajanen M., Lekkala J. & Kirjavainen K. (2000) Electromechanical film (emfi) - a new multipurpose electret material. Sensors and actuators A 84, s. 95–102.

[49] Kokko V-M. (1999) EMFI-anturimateriaalin ominaisuudet. Diplomityö. Oulun yliopisto, Sähkö- ja tietotekniikan osasto, Oulu, 67 s.

[50] Sorvoja H. (1999) Ranteesta tapahtuvaan sykkeen ja verenpaineen mittaukseen soveltuvan anturin kehitystyö. Lisensiaatintyö. Oulun yliopisto, Sähkö- ja tietotekniikan osasto, Oulu, 78 s.

[51] Väätänen A., Strömberg H. & Räty V.P. (2001) Nautilus: A game played in interactive virtual space. In: *Graphics Interface 2001*, Ottawa, Ontario, Canada.

[52] Räisänen L. (1992) Elektromekaanisen kalvon anturisovellukset. Lisensiaattitutkimus. Kuopion yliopisto, Matematiikan, fysiikan ja kemian osasto, Sovelletun fysiikanosasto, Kuopio, 58 s.

[53] Emfit take-off force measuring system in ski jumping (23.05.2004). URL: http://www.emfitech.com/downloads.

[54] Tammisto M. (1999), Emfi-kalvosta moneen käyttöön: Aktiivinen äänen hallinta, potilasta tarkkaileva lattia, älykäs golfmaila? Tampereen kauppakamarilehti 6/99, URL: http://www.tammistoknuutila.fi/kauppakamarilehti (saatavilla 23.05.2004).

[55] Gray R.M. (1984) Vector quantization. IEEE Acoustics, Speech, And Signal Processing Magazine 1, s. 4–29.

[56] Kohonen T. (2001) Self-organizing Maps (3. edition). Springer-Verlag, Berlin, Heidelberg, New York, 501 s.

[57] Duda R., Hart P. & Stork D. (2001) Pattern Classification (2. edition). Wiley-Interscience, New York, 654 s.

[58] Mitchell T.M. (1997) Machine Learning. Computer Science Series, McGraw-Hill International Editions, 414 s.

[59] Livens S., Scheunders P., Van de Wouwer G., Van Dyck D., Smets H., Winkelmans J. & Bogaerts W. (1995) LVQ classification of corrosion images from wavelet features. In: *Proceedings of Trinocular Joint Meeting on Electron Microscopies*, Lausanne, Switzerland.

[60] Duchon A. & Katagiri S. (1993) A minimum-distortion segmentation/LVQ hybrid algorithm for speech segmentation. Journal of the Acoustical Society of Japan (Eng.) 14, s. 37–42.

[61] Cheng K.S., Sun R. & Chow N.H. (2002) Cancerous liver tissue differentiation using LVQ. In: *Proceedings of the 1st International Conference on Artificial Neural Networks in Medicine and Biology (ANNIMAB-1)*, Gothenborg, Sweden.

[62] Mitchie D., Spiegelhalter D.J. & Taylor C.C. (1994) Machine Learning, Neural and Statistical Classifiers (STATLOG project). Ellis Horwood, 290 s.

[63] Kohonen T. (1990) Improved versions of learning vector quantization. In: *Proceedings of the International Joint Conference on Neural Networks*, San Diego, United States, s. 545–550.

[64] Rabiner L.R. (1989) A tutorial on hidden markov models and selected applications in speech recognition. Proceedings of the IEEE 77, s. 257–286.

[65] Levinson S., Rabiner L. & Sondhi M. (1983) An introduction to the application of the theory of probabilistic functions of a Markov process to automatic speech recognition. Bell Syst. Tech. J. 62, s. 1035–1074.

[66] Mäntylä V-M., Mäntyjärvi J., Seppänen T. & Tuulari E. (2000) Hand gesture recognition of a mobile device user. In: *IEEE International Conference on Multimedia and Expo (ICME 2000)*, Vol. 1, s. 281–284.

[67] Shatkay H. & Kaelbling L. (2002) Learning geometrically-constrained hidden markov models for robot navigation: Bridging the topological-geometrical gap. J. Artificial Intelligence Research 16, s. 167–207.

[68] Forney G.D. (1973) The viterbi algorithm. Proceedings of the IEEE 61, s. 268–277.

[69] Baum L.E. (1966) Statistical inference for probabilistic functions of finite state markov chains. Ann. Math. Stat. 37, s. 1554–1563.

[70] Dempster A.P., Laird N.M. & Rubin D.B. (1977) Maximum likelihood from incomplete data via the EM algorithm. J. Royal Statistical Society 39, s. 1–22.

[71] Young S.J., Evermann G., Kershaw D., Moore G., Odell J., Ollason D., Valtchev V. & Woodland P. (2002) The Hidden Markov Model Toolkit book. Cambridge University Engineering Dept., United Kingdom, 276 s.

[72] Iwamida H., Katagiri S., McDermott E. & Tohkura Y. (1990) A hybrid speech recognition system using HMMs with an LVQ-trained codebook. In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 1990)*, Albuquerque, New Mexico, USA, s. 489–492.

[73] Taipale O. & Jurva E. (1999) Muuttujien valinta ja mittausaineiston muodostaminen. Technical report, Tekes, Helsinki, Finland, 123 s.

[74] Jain A.K., Duin R.P.W. & Mao J. (January 2000) Statistical pattern recognition: A review. IEEE Transactions on Pattern Analysis and Machine Intelligence 22, s. 4–31.

[75] Fukunaga K. (1990) Introduction to Statistical Pattern Recognition (2. edition). Academic Press, New York, 592 s.

[76] Bojer T., Hammer B., Schunk D. & von Toschanowitz K.T. (2001) Relevance determination in learning vector quantization. In: Verleysen M., editor, *European Symposium on Artificial Neural Networks (ESANN 2001)*, D-facto publications, Bruges, Belgium, s. 271–276.

[77] Hammer B. & Villman T. (2002) Batch-RLVQ. In: Verleysen M., editor, *European Symposium on Artificial Neural Networks (ESANN 2002)*, D-facto publications, Bruges, Belgium, s. 295–300.

[78] Hammer B. & Villmann T. (2002) Generalized relevance learning vector quantization. Neural Networks 15, s. 1059–1068.

[79] Sato A. & Yamada K. (1995) Generalized learning vector quantization. In: Tesauro G., Touretzky D. & Lee T., editors, *Advances in Neural Information Processing systems (NIPS)*, MIT Press, Vol. 7, s. 423–429.

[80] Pregenzer M., Pfurtscheller G. & Flotzinger D. (1996) Automated feature selection with a distinction sensitive learning vector quantizer. Neurocomputing 11, s. 19–29.

[81] De Stefano C., Sansone C. & Vento M. (2000) To reject or not to reject: That is the question - an answer in case of neural classifiers. IEEE Transactions on Systems, Man, and Cybernetics-part c: Applications and Reviews 30, s. 84–94.

[82] Cordella P., De Stefano C., Tortorella F. & Vento M. (1995) A method for improving classification reliability of multilayer perceptrons. IEEE Transaction on Neural Networks 6, s. 1140–1147.

[83] Kittler J., Hatef M., Duin R.P.W. & Matas J. (1998) On combining classifiers. IEEE Transactions on Pattern Analysis and Machine Intelligence 20, s. 226–239.

[84] Ho T.K. (1993) Recognition of handwritten digits by combining independent learning vector quantization. In: *Proceedings of the Second International Conference on Document Analysis and Recognition*, Tsukuba Science City, Japan, s. 818–821.

[85] Heinonen P. & Neuvo Y. (1987) FIR-median hybrid filters. IEEE Trans. Acoust., Speech, Signal Processing ASSAP-35, s. 832–838.

[86] Ge X. & Smyth P. (2000) Deformable markov model templates for time-series pattern matching. In: *Proceedings of the sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Boston, MA, USA, s. 81–90.

[87] The Mathworks: MATLAB for technical computing (23.05.2004). URL: http://www.mathworks.com.

[88] Brigham E.O. (1974) The Fast Fourier Transform. Prentice-Hall, 252 s.

[89] Kukolich L. & Lippmann R. (1999) LNKnet User's Guide. Massachusetts Institute of Technology, Lincoln Laboratory, USA, 198 s.

[90] Theodoridis S. & Koutroumbas K. (1998) Pattern Recognition. Academic Press, New York, 625 s.

[91] Kohonen T., Hynninen J., Kangas J., Laaksonen J. & Torkkola K. (1996) LVQ_PAK: The learning vector quantization program package. Technical report, Helsinki University of Technology, Finland, Report A30, 30 s.

[92] Vesanto J., Himberg J., Alhoniemi E. & Parhankangas J. (1999) Self-organizing map in matlab: the SOM toolbox. In: *Proceedings of the Matlab DSP Conference 1999*, Espoo, Finland, s. 35–40.

[93] Hidden markov model (HMM) toolbox for matlab (23.05.2004). URL: http://www.ai.mit.edu/~murphyk/Software/HMM/hmm.html.

[94] Kohavi R. (1995) A study of cross-validation and bootsrap for accuracy estimation and model selection. In: *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI 1995)*, Montreal, Quebec, Canada, s. 1137–1143.

10. LIITTEET

Liite 1 Askelpiirteet LVQ-tunnistuksessa

Liite 2 Askelpiirteet HMM-tunnistuksessa

Taulukko 7. Käytetyt askelpiirteet LVQ-tunnistuksessa. Piirteet eivät ole paremmuusjärjestyksessä. Kuitenkin piirteitä 1-13 käytettiin ensimmäisessä vaiheessa tunnistukseen ja kaikkia 31 piirrettä DSLVQ-testien yhteydessä. Nimi on piirrettä kuvaava nimi, jota on käytetty toteutusosan tekstissä, ja joissakin kohdissa toisten piirteiden kuvauksen määrittämiseen. Kuvauksella kerrotaan tekstimuodossa, mihin kohtaan askelta piirre liittyy tai miten se lasketaan

piirteen numero	nimi	kuvaus
1.	x_{max1}	Kantapään iskun maksimikohta askeleen alusta
2.	y_{max1}	Kantapään iskun maksimiamplitudi
3.	x_{min}	Kantapään ja päkijän iskujen välinen minimikohta
4.	y_{min}	Kantapään ja päkijän iskujen välinen minimiamplitudi
5.	x_{max2}	Päkijän iskun maksimikohta askeleen alusta
6.	y_{max2}	Päkijän iskun maksimiamplitudi
7.	x_{end}	Askeleen loppukohta ennen kalvon palautumista
8.	y_{end}	Askeleen loppukohdan amplitudi
9.	$mean_1$	Keskiarvo askeleen alusta miminikohtaan (x_{min})
10.	std_1	Keskihajonta askeleen alusta miminikohtaan (x_{min})
11.	$mean_2$	Keskiarvo miminikohdasta (x_{min}) välipisteeseen (x_{mid})
12.	std_2	Keskihajonta miminikohdasta (x_{min}) välipisteeseen (x_{mid})
13.	$mean_{max}$	y_{max1}, y_{max2} ja y_min erotusten keskiarvo
14.	$area_1$	Pinta-ala askeleen alusta minimikohtaan (x_{min})
15.	$area_2$	Pinta-ala minimikohdasta (x_{min}) välipisteeseen (x_{mid})
16.	x_{heel}	Kantapään alkukohta (kun amplitudi ylittää x_{min})
17.	x_{ball}	Päkijän loppukohta (kun amplitudi alittaa x_{min})
18.	$length_{heel}$	$(x_{heel} - x_{min})$
19.	$length_{ball}$	$(x_{min} - x_{ball})$
20.	$mean_3$	Keskiarvo askelee loppuosalle (x_{mid}, x_{end})
21.	std_3	Keskihajonta askelee loppuosalle (x_{mid}, x_{end})
22.	$shape_{heel}$	$((y_{max1} - y_{min})/(x_{min} - x_{heel}))$
23.	$shape_{ball}$	$((y_{max2} - y_{min})/(x_{ball} - x_{mid}))$
24.	x_{rel1}	(x_{max1}/x_{end})
25.	x_{rel2}	(x_{max2}/x_{end})
26.	y_{rel1}	(y_{max1}/x_{end})
27.	y_{rel2}	(y_{max2}/x_{end})
28.	fft_1	Amplitudispektrin 1. kerroin
29.	fft_2	Amplitudispektrin 2. kerroin
30.	fft_3	Amplitudispektrin 3. kerroin
31.	fft_4	Amplitudispektrin 4. kerroin

Taulukko 8. Käytetyt askelpiirteet HMM-tunnistuksessa. Askelosa kertoo, mistä kohden askelta piirteet on ajallisesti valittu. Piirrevektorisekvenssin ensimmäiseen vektoriin valittiin piirteet, joissa lukee "alku" (piirteet 1-8), toiseen "keski" (piirteet 9-16) sekä viimeiseen "loppu" (piirteet 17-24), näin sekvenssi koostuu kolmesta osasta. Nimi kuvaa kyseistä piirrettä ja kuvaus ilmoittaa tekstimuodossa kunkin piirteen kohdan tai laskennan askelsignaalista

askelosa	numero	nimi	kuvaus
alku	1.	$mean_1$	Keskiarvo askeleen alusta miminikohtaan (x_{min})
	2.	std_1	Keskihajonta askeleen alusta miminikohtaan (x_{min})
	3.	$area_1$	Pinta-ala askeleen alusta minimikohtaan (x_{min})
	4.	$length_1$	Alkuosan pituus
	5.	x_{max1}	Kantapään iskun maksimikohta askeleen alusta
	6.	y_{max1}	Kantapään iskun maksimiamplitudi
	7.	x_{heel}	Kantapään alkukohta (kun amplitudi ylittää x_{min})
	8.	$shape_{heel}$	$((y_{max1} - y_{min})/(x_{min} - x_{heel}))$
keski	9.	$mean_2$	Keskiarvo miminikohdasta (x_{min}) välipisteeseen (x_{mid})
	10.	std_2	Keskihajonta miminikohdasta (x_{min}) välipisteeseen (x_{mid})
	11.	$area_2$	Pinta-ala minimikohdasta (x_{min}) välipisteeseen (x_{mid})
	12.	$length_2$	Keskiosan pituus
	13.	x_{max2}	Päkijän iskun maksimikohta askeleen alusta
	14.	x_{ball}	Päkijän loppukohta ((kun amplitudi alittaa x_{min})
	15.	$shape_{ball}$	$((y_{max2} - y_{min})/(x_{ball} - x_{mid}))$
	16.	x_{mid}	Välipiste
loppu	17.	$mean_3$	Keskiarvo välipisteestä (x_{mid}) loppuun (x_{end})
	18.	std_3	Keskihajonta välipisteestä (x_{min}) loppuun (x_{end})
	19.	$area_3$	Pinta-ala välipisteestä (x_{mid}) loppuun (x_{end})
	20.	$lenth_3$	Loppuosan pituus
	21.	x_{neg}	Negatiivisen amplitudin kohta
	22.	y_{neg}	Negatiivisin amplitudin arvo
	23.	$shape_{end}$	$(length_3/y_{neg})$
	24.	rel_{end}	(y_{max2}/x_{end})