OULUN SEUDUN
AMMATTIKORKEAKOULU

Arto Rajaniemi

MAKSIMIVOIMAMITTARIN ALGORITMIKEHITYS

MAKSIMIVOIMAMITTARIN ALGORITMIKEHITYS

Arto Rajaniemi
Opinnäytetyö
Kevät 2011
Tietotekniikan koulutusohjelma
Oulun seudun ammattikorkeakoulu

OULUN SEUDUN AMMATTIKORKEAKOULU TIIVISTELMÄ

Koulutusohjelma	Opinnäytetyö	Sivuja	+	Liitteitä
Tietotekniikka	Insinöörityö	40	+	14
Suuntautumisvaihtoehto	Aika			
Langaton tietoliikenne	2011			
Työn tilaaja	Työn tekijä			
Manne Hannula	Arto Rajaniemi			
Työn nimi				
Maksimivoimamittarin algoritmikehitys				
Avainsanat				
Chronos, algoritmi, ristikorrelaatio				

Tässä insinöörityössä oli tavoitteena toteuttaa Matlab-ohjelmistolla suunniteltu algoritmi C-ohjelmointikielelle. Työ tehtiin Oulun seudun ammattikorkeakoulun tekniikan yksikön MUSTI-projektiin, jossa on vuodesta 2004 alkaen kehitetty maksimivoimateknologiaan perustuvaa mittalaitetta kuntosaliharrastajien käyttöön. Algoritmi tuli saada toimimaan reaaliaikaisesti Texas Instrumentsin ez430-Chronos-kehitystyökalun ja PC:n välillä.

Algoritmin tehtävänä on havaita käyttäjän tekemät virheelliset liikkeet. Algoritmi vertaa käyttäjän tekemiä liikkeitä referenssisignaaliin, jonka käyttäjä tekee ennen varsinaisia liikkeitä. Vertailua tehdään mm. paikallisten maksimien ja minimien sekä ristikorrelaation avulla.

Työn lopputuloksena syntyi C-ohjelmointikielellä toteutettu algoritmi, joka ilmoittaa käyttäjälle äänimerkillä onnistuneista ja virheellisistä liikkeistä. Ohjelma laskee myös tehtyjen toistojen määrän.

OULU UNIVERSITY OF APPLIED SCIENCES ABSTRACT

Degree programme	Thesis	Number of pages	+	Appendices
Information Technology and Telecommunications	B.Sc.	40	+	14

Line	Date
Wireless Devices	2011

Commissioned by	Author
Manne Hannula	Arto Rajaniemi

Thesis title

Algorithm Development of Maximum Strength Meter

Keywords

Chronos, algorithm, cross correlation

The purpose of this thesis was to implement a Matlab designed algorithm for the C programming language. This work was done in MUSTI project at the School of Engineering of Oulu University of Applied Sciences, which has been developing a maximum strength technology-based measuring device for gym enthusiasts since 2004. The algorithm was made to operate in real time via Texas Instrument's ez430-Chronos development tool and PC.

The purpose of the algorithm is to detect the user's incorrect movements. The algorithm compares the user's movements to a reference signal, which is made by the user before actual movements. Comparisons are made with the help of local maximas and minimas as well as with the cross correlation.

The result of this project is an algorithm, which is implemented by the C programming language. The algorithm indicates successful and incorrect movements by a sound mark. The program also counts the number of repetitions.

ALKUSANAT

Suuret kiitokset Oulun seudun ammattikorkeakoulun tekniikan yksikölle opinnäytetyön tarjoamisesta. Haluan kiittää yliopettaja Ensio Sieppiä työn ohjauksesta sekä tutkijayliopettaja Manne Hannulaa työn tilauksesta. Kiitokset myös lehtori Tuula Hopeavuorelle opinnäytetyön tekstinohjauksesta. Haluan myös kiittää koko tietotekniikan osastoa asiantuntevasta opetuksesta opiskelujen varrella. Erityiskiitos kaikille projektissa työskenteleville osapuolille yhteistyöstä sekä hyvän ilmapiirin luomisesta.

Oulussa 26.4.2011

Arto Rajaniemi

SISÄLTÖ

ALKUSANAT....5
SISÄLTÖ....6
1 JOHDANTO....7
2 MUSTI-PROJEKTI....8
3 KEHITYSALUSTA....9
4 KIIHTYVYYSANTURI....10
5 ALGORITMIN TOIMINTA....12
5.1 Referenssisignaali....14
5.2 Näytesignaali....15
5.3 Signaalin suodatus....15
5.4 Aloitus- ja lopetuskohdan määrittäminen signaalista....18
5.5 Paikallisten maksimien ja minimien määrittäminen....23
5.5.1 Nollanylityskohtien määrittäminen....24
5.5.2 Maksimien ja minimien määrittäminen derivoidusta signaalista....25
5.5.3 Maksimien ja minimien määrittäminen alkuperäisestä signaalista....26
5.5.4 Maksimien ja minimien suodatus....28
5.5.5 Vertailu maksimien ja minimien avulla....30
5.6 Ristikorrelaatio....32
5.6.1 Ristikorrelaatio akseleittain....32
5.6.2 Vertailu ristikorrelaation avulla....33
5.7 Testaus....34
6 YHTEENVETO JA JATKOKEHITYS....36
7 POHDINTA....38
LÄHTEET....39
LIITTEET....40

1 JOHDANTO

Insinöörityön aiheena oli maksimivoimamittarin algoritmikehitys. Työ tehtiin projektiluontoisesti hyvinvointiteknologian MUSTI-projektiin, jossa on vuodesta 2004 alkaen kehitetty maksimivoimateknologiaan perustuvaa mittalaitetta kuntosaliharrastajien käyttöön. Maksimivoimamittari pyritään saamaan jo lähitulevaisuudessa kaupalliseksi ja kilpailukykyiseksi tuotteeksi markkinoille. Laite on tarkoitettu sekä kuntosaliharrastelijoille että ammattilaisille voimaharjoittelun mittaamiseen ja analysointiin.

Eräs aikaisempi insinöörityö samaan projektiin saatiin valmiiksi juuri ennen kuin tätä insinöörityötä alettiin tehdä. Työn tekijänä toimi tuolloin Veli-Pekka Välimaa. Hän kehitti maksimivoimamittariin uuden algoritmin, jonka avulla toistomäärälaskurilla päästään parempiin tuloksiin kuin aikaisemmin. Algoritmin kehitys toteutettiin tuolloin Math Worksin Matlab-ohjelmistoa käyttäen. Tässä insinöörityössä Matlab-ohjelma toteutettiin C-ohjelmointikielellä reaaliaikaiseen laitteeseen. Erityisesti haasteita toivat reaaliaikaisuus sekä Matlab-ohjelmiston omat kirjastot, joita algoritmissa käytettiin paljon. Tavoitteena oli tehdä toimiva C-kielinen ohjelma, joka perustuu uuteen algoritmiin.

2 MUSTI-PROJEKTI

Vuonna 2004 käynnistyneessä projektissa kehitetään maksimivoimateknologiaan perustuvaa mittalaitetta kuntosaliharrastajien käyttöön. Tuotteesta pyritään lähitulevaisuudessa lanseeraamaan markkinoille kilpailukykyinen laite. Tuotteen lanseeraus on pitkäjänteistä toimintaa, joka ei tapahdu nopeasti. Tämän vuoksi MUSTI-projektissa on otettu käyttöön eri yksiköiden välinen toiminta, minkä avulla hyödynnetään eri osa-alueiden asiantuntijoita. Projektin toiminnassa on mukana tällä hetkellä yli 30 henkilöä, jotka muodostuvat eri yksiköiden opiskelijoista sekä opettajista.

Maksimivoimamittarin ensimmäinen prototyyppi oli langallinen laite. Prototyypin avulla laitetta on kehitetty aina vain parempaan suuntaan. Laite kehitettiin langattomaksi vuonna 2010, mutta se on edelleen riippuvainen PC-koneesta. Yksi seuraavista kehitysaskeleista on se, että laite saadaan toimimaan täysin itsenäisesti ilman PC-konetta. Laitteeseen kehitettiin uusi algoritmi vuonna 2010, jossa toistomäärälaskurilla päästään parempiin tuloksiin. Algoritmi on suunniteltu Matlab-ohjelmalla, joka soveltuu hyvin erityisesti signaalinkäsittelyyn ja erilaisten algoritmien kehitykseen. Tässä insinöörityössä sama algoritmi toteutetaan reaaliaikaisena toimivaksi C-ohjelmointikielellä. (1.)

Koska laitteesta on tarkoituksena lanseerata markkinoille kilpailukykyinen tuote, laitteen muotoilu on myös hyvin tärkeässä roolissa. MUSTI-projektissa on yhteistoiminnassa mukana Lapin yliopiston teollisen muotoilun koulutusohjelmasta neljä henkilöä, jotka suunnittelevat laitteen ulkomuotoa. Liiketalouden yksiköstä projektissa on mukana muutama henkilö. He vastaavat mm. tuotteen markkinoinnista ja liiketoiminnan kehittämisestä. Projektissa on myös opiskelijoita sosiaali- ja terveysalan yksiköstä. Heiltä löytyy asiantuntemusta mm. fysioterapian osa-alueelta. Tekniikan yksiköstä projektissa on mukana sekä tietotekniikan että hyvinvointiteknologian osastoilta useampi taho. Heidän osa-alueenaan on mm. tutkia ja kehittää eri teknologioita, joita MUSTI-projektissa voidaan hyödyntää.

3 KEHITYSALUSTA

Kehitysalustana käytettiin Texas Instrumentsin valmistamaa eZ430-Chronos-kehitystyökalua (kuva 1), johon on valmiiksi integroituna paljon erilaisia toimintoja. Laite sisältää mm. 96-segmenttisen LCD-näytön (Liquid Crystal Display), painesensorin ja kolmiakselisen kiihtyvyysanturin. Kelloon on integroitu langaton sovitin, jonka avulla laitteeseen voidaan yhdistää lisälaitteita, kuten askelmittarin tai sykepannan. e430-Chronos tarjoaa myös lämpötilan mittauksen ja pariston jännitemittauksen sekä USB-pohjaisen (Universal Serial Bus) langattoman yhteyden kellon ja PC:n välille. (2, s. 8.)

KUVA 1. eZ430-Chronos kehitystyökalu (2, s. 8.)

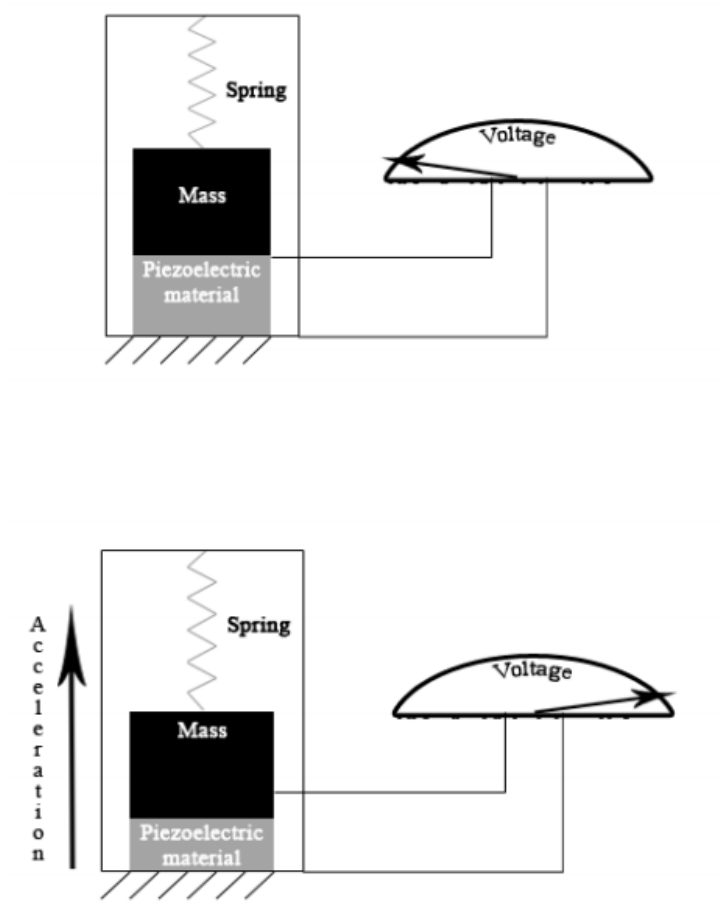
Kellon voi myös purkaa uudelleenohjelmoitavaksi omia sovelluksia varten. Ohjelmointi tapahtuu mukana tulevan USB-ohjelmointisovittimen avulla. Tässä insinöörityössä ei kuitenkaan ohjelmoitu itse kelloa, koska uusi algoritmi liian laaja kellon sisäisien muistien kannalta. (2, s. 8.)

4 KIIHTYVYYSANTURI

Kiihtyvyysanturia käytetään nykyään monissa paikoissa. Sen käyttö on yleistynyt viime vuosina huomattavasti, ja koko ajan keksitään uusia käyttötarkoituksia. Kiihtyvyysanturin toimintaperiaate on hyvin yksinkertainen. Anturissa on massa, johon liikkeestä kohdistuva voima mitataan. Sen toiminta perustuu Newtonin toiseen lakiin. (3.)

Anturi täytyy kalibroida maan vetovoiman suhteen, koska kiihtyvyys näyttää 9,81 m/s^2 kiihtyvyyttä ylöspäin. Anturi mittaa voimien vaikutusta kolmelta eri akselilta, jotka ovat X, Y ja Z. Antureita voidaan valmistaa käyttötarkoituksen mukaan mittaamaan joko yhden, kahden tai kolmen akselin suuntaan. (3.)

Kiihtyvyysantureita voidaan valmistaa toimimaan usealla eri tekniikalla. Yksi yleisimmin käytetty tekniikka perustuu pietsosähköisiin aineisiin, jossa mittaus perustuu aineen ominaisuuteen luoda jännite päidensä välille puristuessaan kokoon. Jännite on suoraan verrannollinen aineeseen vaikuttavaan voimaan. Pietsosähköiset aineet ovat sähköä johtamattomia kiteitä, keraameja tai polymeereja. Kuvassa 2 on esitetty pietsosähköisen kiihtyvyysanturin toimintaperiaate. (3.)

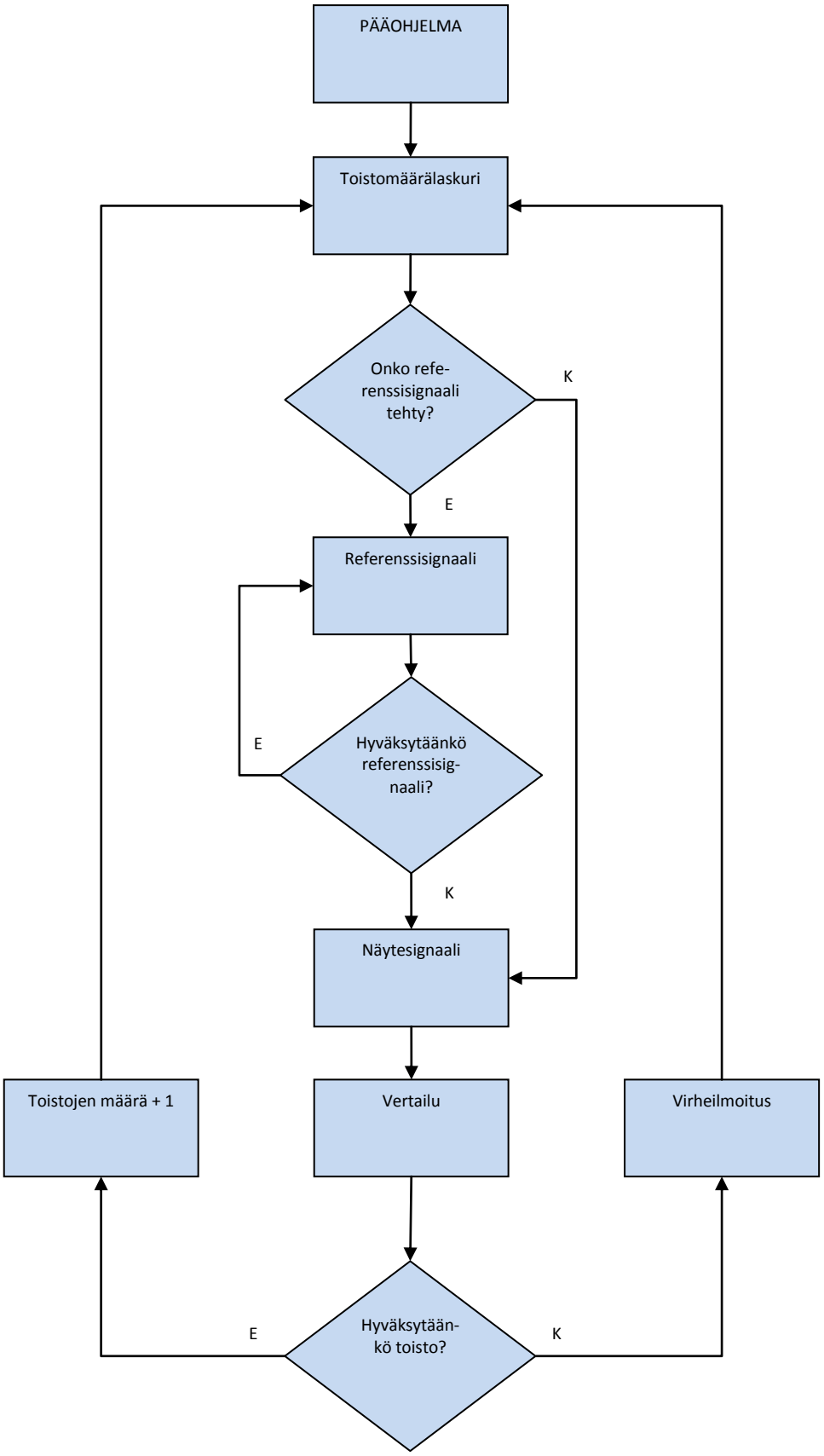


KUVA 2. Pietsosähköisen kiihtyvyysanturin toimintaperiaate (3.)

Kiihtyvyyden tapahtuessa ylöspäin massa painaa pietsosähköisen materiaalin kasaan, jolloin materiaalin päihin syntyy jännite. Tämä jännite osoittaa voiman suuruuden. On olemassa myös monia muita tapoja kuin pietsosähköisiin aineisiin perustuva tekniikka, kuten esimerkiksi kapasitiivisuuteen tai magneettikenttään perustuvia. (3.)

5 ALGORITMIN TOIMINTA

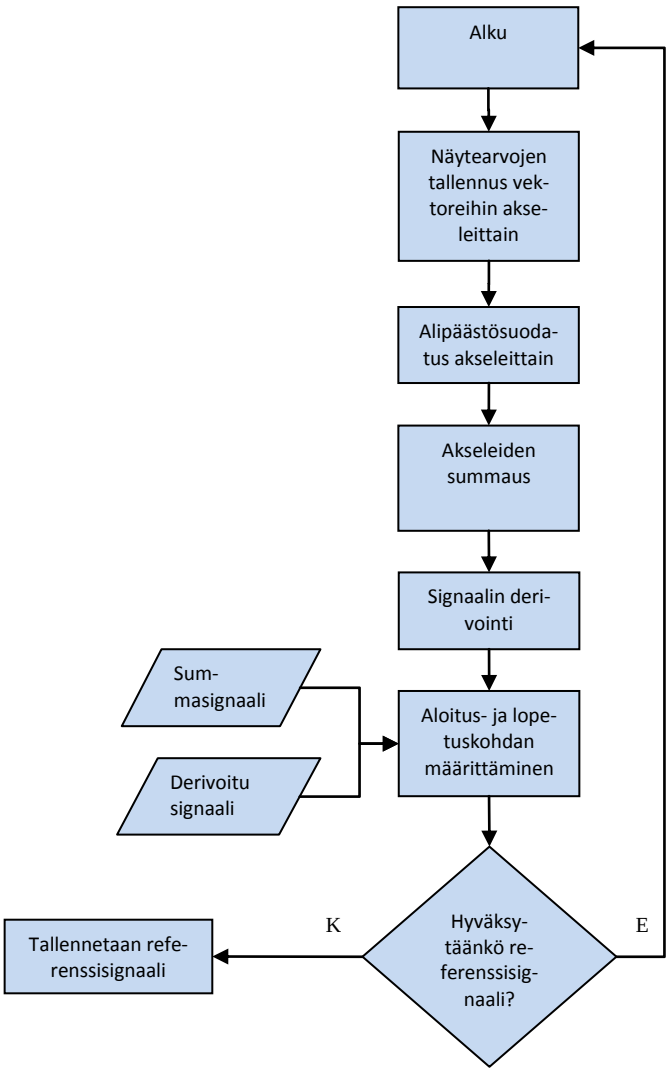
Insinööriopiskelija Veli-Pekka Välimaan kehittämän algoritmin toiminta perustuu referenssisignaaliin, johon käyttäjän tekemiä suoritteita verrataan. Käyttäjän täytyy tehdä referenssiliike jokaiselle eri liikkeelle. Referenssiliikkeen onnistuminen on hyvin tärkeässä roolissa, jotta liikesarjassa tehdyt liikkeet tunnistetaan oikein. Liikesarjassa tehtyä liikettä vertaillaan referenssiliikkeeseen paikallisien maksimien ja minimien sekä ristikorrelaation avulla. Kuvassa 3 oleva vuokaavio esittää ohjelman kokonaisuutta. Liitteessä 1 on pääohjelman ohjelmalistaus.



KUVA 3. Toistomäärälaskurin kokonaisuus

5.1 Referenssisignaali

Referenssisignaalin tallennus tehdään ohjelmassa aina ennen kuin varsinaista liikesarjaa aletaan tehdä. Referenssiliikettä tehtäessä tallennus aloitetaan vasta, kun kello on ollut hetken paikoillaan. Käyttäjä voi siis laittaa toistomäärälaskurin käyntiin, minkä jälkeen hän voi rauhassa ottaa aloitusasennon. Kun kello on ollut tietyn ajan paikoillaan, referenssiliikkeen voi aloittaa äänimerkin jälkeen. Referenssiliikkeen tekemiseen on aikaa noin kolme sekuntia. Kuvassa 4 oleva vuokaavio havainnollistaa referenssiliikkeen tallennustilannetta. Liitteessä 2 on ohjelmalistaus referenssisignaalin tallennuksesta.



KUVA 4. Referenssisignaalin tallennus

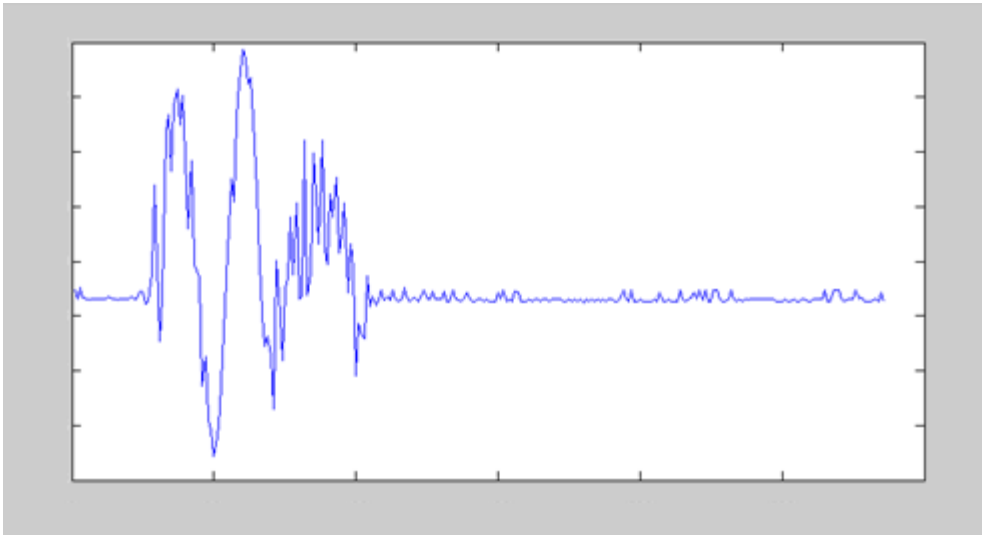
5.2 Näytesignaali

Näytesignaalilla tarkoitetaan tässä yhteydessä yksittäistä liikettä, jota verrataan referenssisignaaliin. Näytesignaalin tallennus poikkeaa referenssisignaalin tallennuksesta siten, että erillistä aikaa ei ole määritelty. Aikaa ei voida määritellä, koska ohjelman täytyy toimia reaaliaikaisesti. Sen sijaan ohjelmassa tutkitaan koko ajan tiettyjä raja-arvoja.

Aluksi ohjelmassa siirrytään silmukkaan, jossa tutkitaan kiihtyvyysantureilta saatua dataa. Jos kiihtyvyysarvot eivät pysy raja-arvojen sisällä tietyn näytemäärän verran, siirrytään seuraavaan silmukkaan, jossa tutkitaan, milloin kiihtyvyysarvot ovat pysyneet raja-arvojen sisällä tietyn näytemäärän verran. Toisin sanoen aluksi tutkitaan, milloin liike aloitetaan, ja sen jälkeen tutkitaan, milloin liike lopetetaan. Tämän jälkeen edetään lähes kuten referenssisignaalin kohdalla. Liitteessä 3 on ohjelmalistaus näytesignaalin käsittelystä.

5.3 Signaalin suodatus

Kiihtyvyysantureilta tuleva data saadaan luettua ohjelmassa jokaiselta eri akselilta erikseen. Akseleita nimitetään X-, Y- ja Z-akseliksi. Kiihtyvyysantureilta saatu data on aluksi hyvin häiriöistä (kuva 5), minkä vuoksi ohjelmassa on digitaalinen suodatin.



KUVA 5. Suodattamaton signaali. X-akseli kuvastaa aikaa ja Y-akseli jännitearvoja.

Mittausten absoluuttiset arvot on skaalattu suhteellisiin arvoihin. Suodattimella signaalille tehdään alipäästösuodatus, eli matalat taajuudet päästetään läpi ja korkeat taajuudet suodatetaan. Suodattimen kertoimia voidaan arvioida helposti Matlabin komennolla

```
[N, Wn] = buttord(Wp, Ws, Rp, Rs),
```

missä `N` on ulostulona saatava aste ja `Wn` on rajataajuus. Tarvittavat argumentit ovat seuraavat:

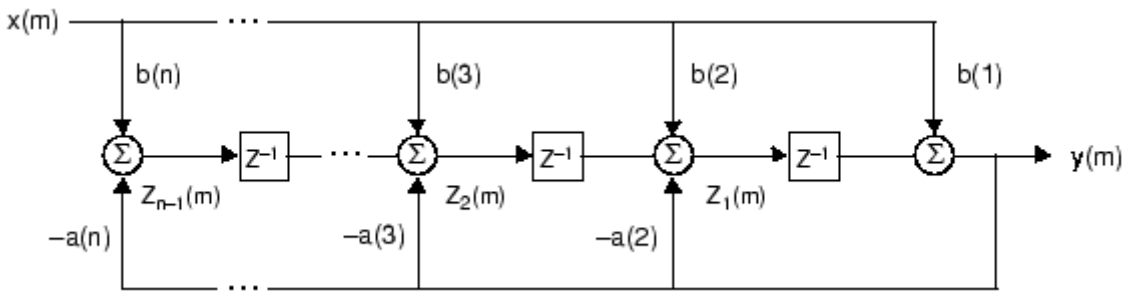
- `Wp` on päästökaistan rajataajuus
- `Ws` on estokaistan rajataajuus
- `Rp` on suurin sallittu värähtely päästökaistalla desibeleinä
- `Rs` on estokaistan minimivaimennus.

Tämän jälkeen kertoimien lukuarvot saadaan komennolla

```
[b, a] = butter(N, Wn),
```

missä tuloksena saadaan kaksi vektoria, jotka sisältävät tarvittavat kertoimet.

Digitaalinen suodatus on toteutettu Direct-Form-II-Transposed-struktuurin mukaisesti (kuva 6). Struktuurin toiminnan voi myös esittää kaavan 1 mukaisesti. Liitteessä 4 on suodattimen ohjelmallinen toteutus.

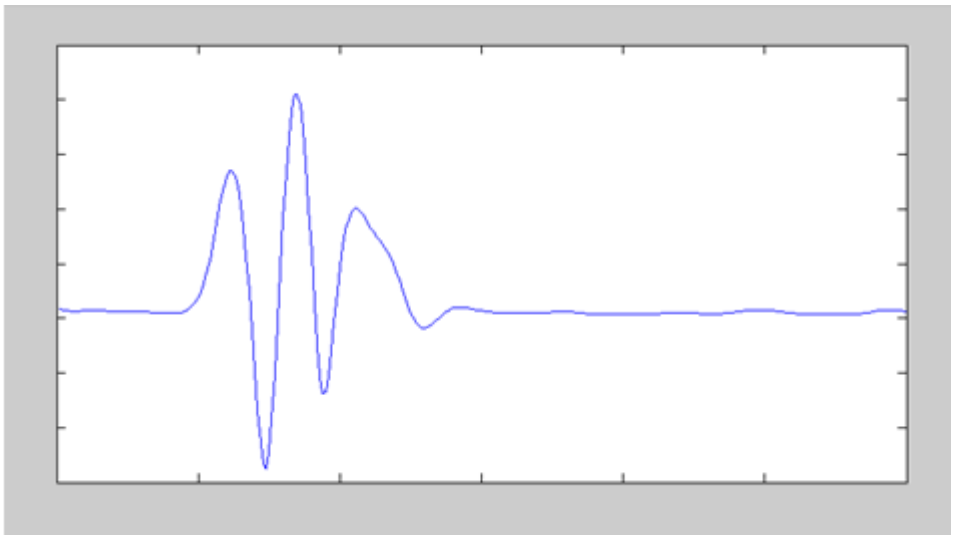


KUVA 6. Direct-Form-II-Transposed-struktuuri (4)

$$y(m) = b(1) \cdot x(m) + b(2) \cdot x(m-1) + \ \ldots + b(nb+1) \cdot x(m-nb) \\ - a(2) \cdot y(m-1) - \ \ldots - a(na+1) \cdot y(m-na)$$ *KAAVA 1.*

Suodattimella suodatetaan erikseen X-, Y- ja Z-akselin signaali, minkä jälkeen kaikkien akseleiden signaalit summataan yhteen kaavan 2 mukaisesti. Kuvasta 7 voi tarkastella suodatuksen lopputulosta, jossa ylimääräinen kohina on poistunut summasignaalista.

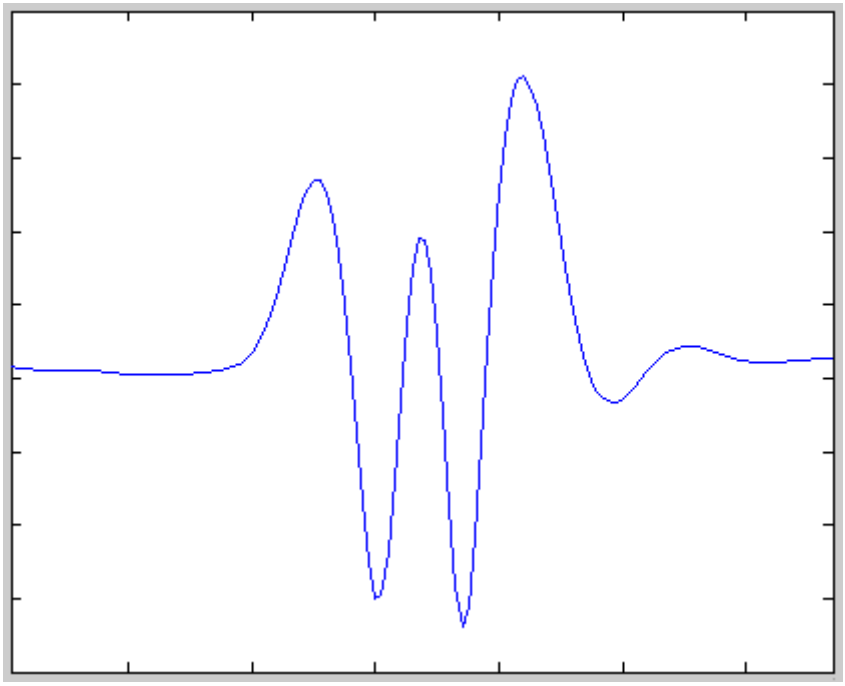
$$a = \sqrt{x^2 + y^2 + z^2}$$ *KAAVA 2.*



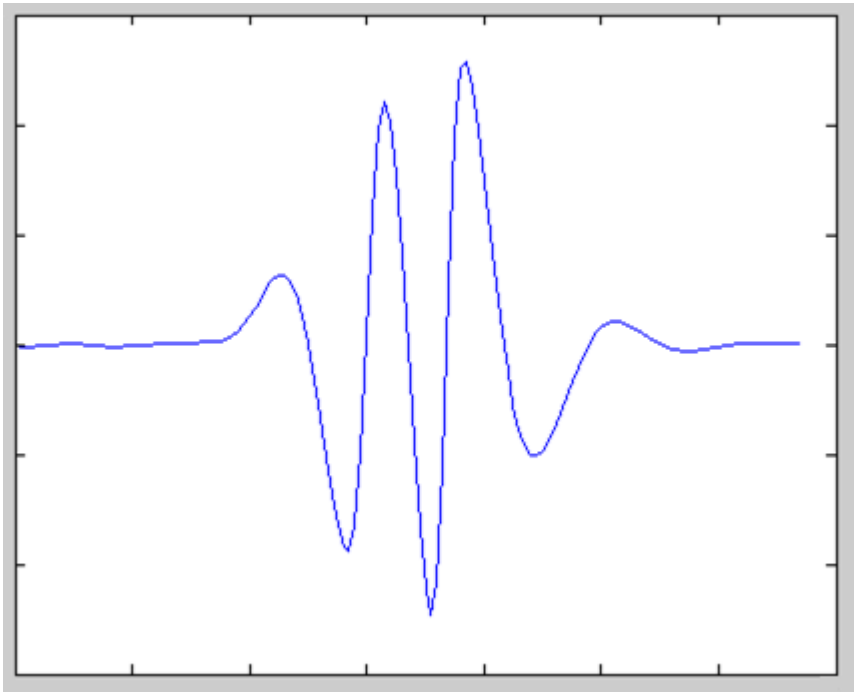
KUVA 7. Suodatettu signaali

5.4 Aloitus- ja lopetuskohdan määrittäminen signaalista

Seuraavassa vaiheessa signaalista täytyy etsiä todellinen aloitus- ja lopetuskohta. Aloituskohdalla tarkoitetaan signaalissa sitä kohtaa, kun näytearvoissa alkaa tapahtua merkittävämpiä muutoksia verrattuna lepotilanteeseen. Lopetuskohdalla tarkoitetaan signaalissa sitä kohtaa, kun näytearvoissa ei enää havaita merkittäviä muutoksia. Tämän avulla signaalista saadaan olennaisin tieto talteen. Aloitus- ja lopetuskohdan määrittämisessä tarvitaan alkuperäistä signaalia (kuva 8) ja sen derivoitua signaalia. (Kuva 9.)

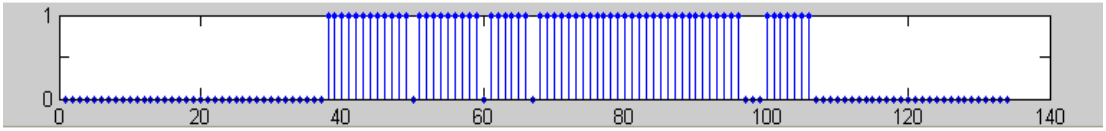


KUVA 8. Alkuperäinen signaali



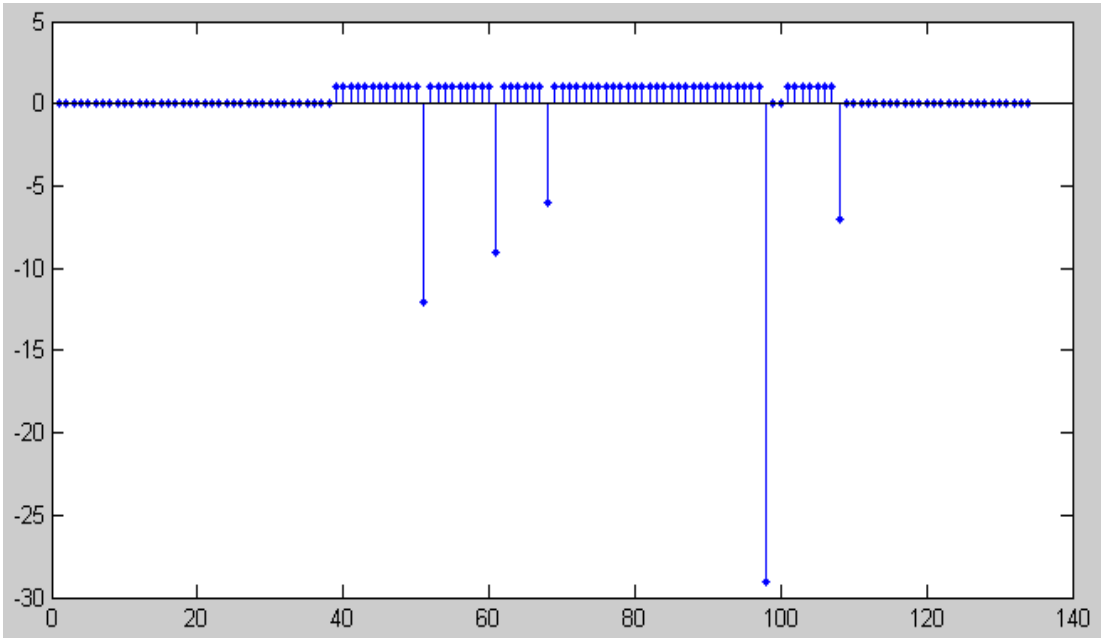
KUVA 9. Derivoitu signaali

Aluksi tutkitaan derivoitua signaalia tiettyjen raja-arvojen avulla. Jos näytteen lukuarvo ylittää raja-arvot, tallennetaan vektoriin lukuarvo 1. Samalla tavalla käydään läpi koko signaali, minkä jälkeen vektorin jokaisessa alkioissa on lukuarvo 1 tai 0. (Kuva 10.)



KUVA 10. Raja-arvojen avulla muodostunut vektori. Alkion arvo on joko 0 tai 1.

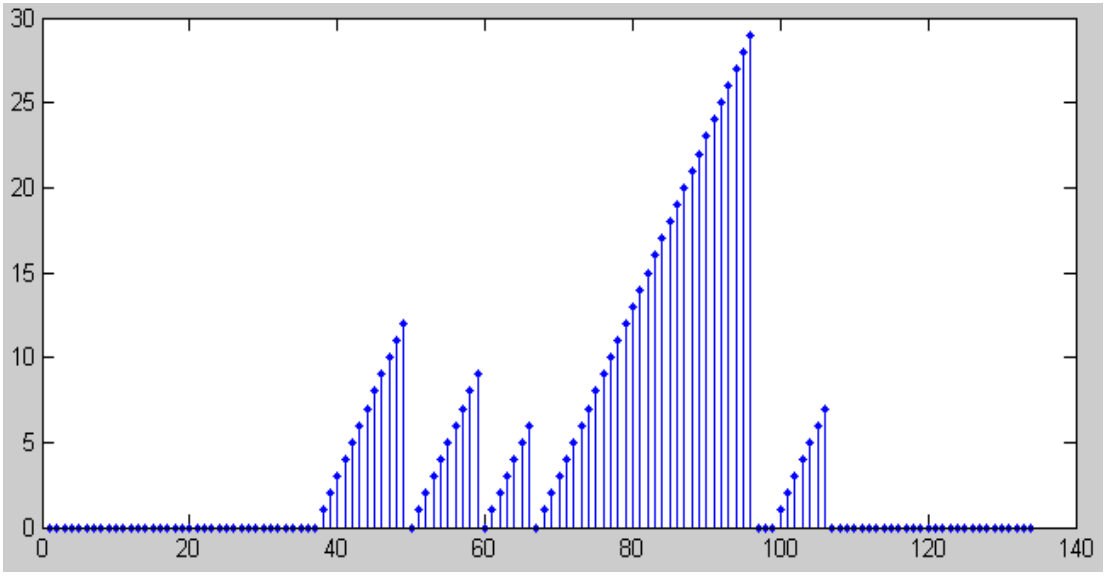
Seuraavaksi edellisestä vektorista etsitään paikkatiedot, joissa alkion arvo muuttuu 0:sta 1:een tai 1:stä 0:aan. Kyseiset paikkatiedot vähennetään toisistaan ja ne asetetaan vektoriin negatiivisena lukuarvona. Lukuarvot sijoitetaan vektorin alkioon, joissa muutos on tapahtunut 1:stä 0:aan. Alkioon saadaan näin peräkkäisten ykkösien lukumäärä negatiivisena lukuna. (Kuva 11.)



KUVA 11. Vektori, johon on lisätty peräkkäisien ykkösien lukumäärä negatiivisena lukuna

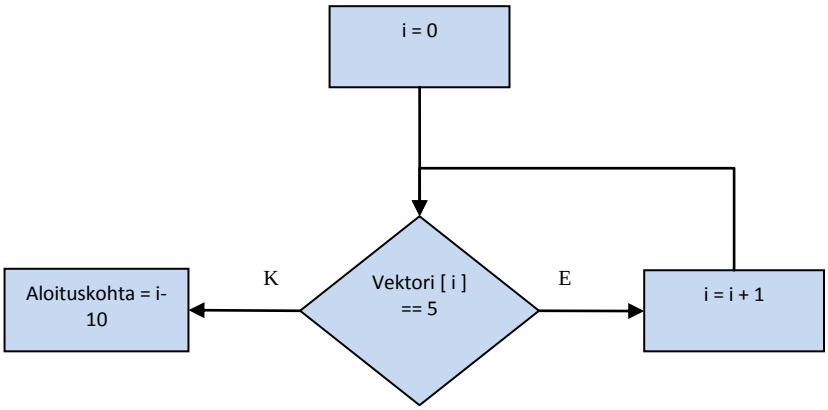
Seuraavaksi vektorin lukuarvoista lasketaan kumulatiivinen summa kaavan 3 mukaisesti. Liitteestä 5 voi tarkastella kumulatiivisen summan toteutusta ohjelmallisesti. Tuloksena saadaan kuvan 12 mukainen vektori.

$C_i = \sum_{j=1}^{i} V_j$ *KAAVA 3.*



KUVA 12. Kumulatiivinen summa

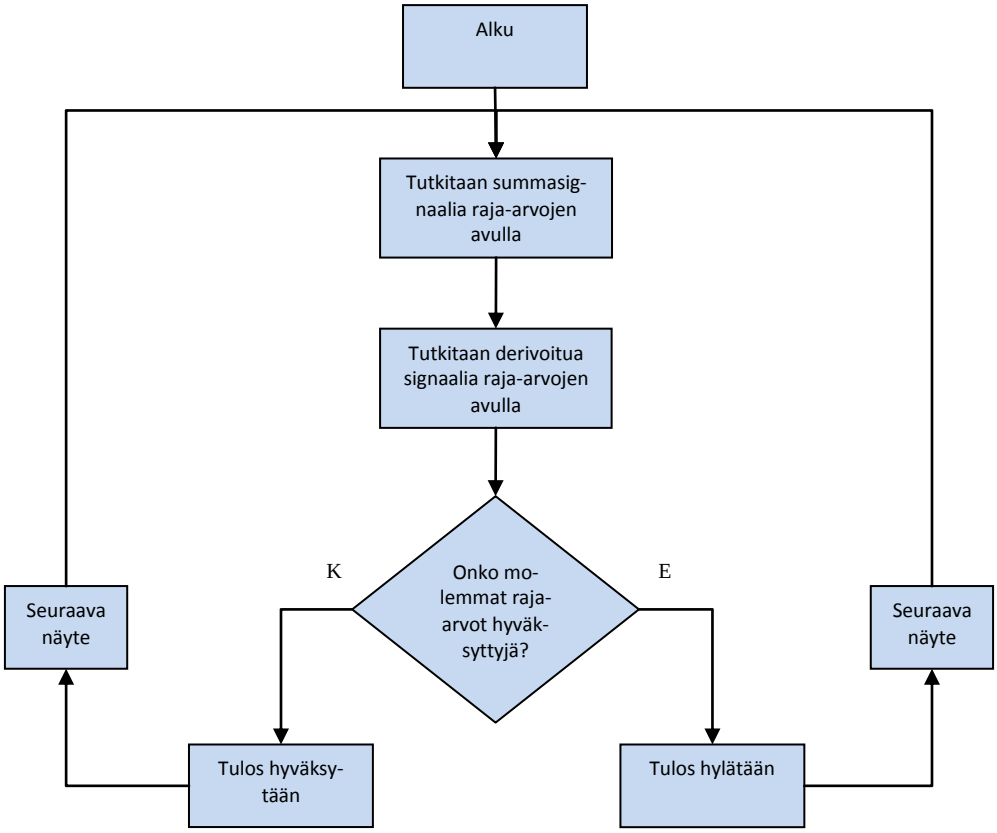
Tässä vaiheessa voidaan helposti päätellä, milloin liike alkaa ja milloin liike loppuu. Aloituskohdan määrittäminen voidaan toteuttaa esimerkiksi kuvan 13 mukaisesti. Liitteessä 6 on kyseisen kohdan ohjelmalistaus.



KUVA 13. Aloituskohdan määrittäminen

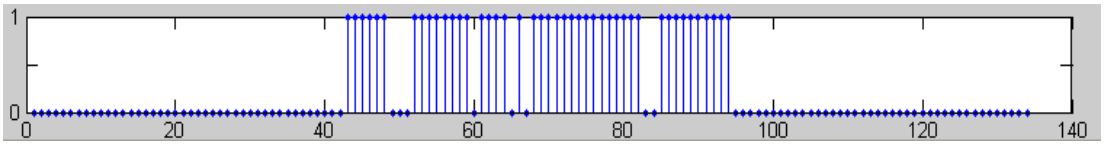
Kun aloituskohta on tiedossa, voidaan sen vektorin alkiosta eteenpäin etsiä lopetuskohtaa. Lopetuskohdan etsiminen tapahtuu samalla periaatteella kuin aloituskohdankin etsiminen, mutta loogiset arvot täytyy kääntää päinvastaisiksi raja-arvotarkastelukohdassa. Kun näytteen näytearvo pysyy raja-arvojen sisällä, tallennetaan vektoriin lukuarvo 1.

Seuraavaksi signaalille täytyy tehdä keskiarvoistaminen, Tällä keinolla voidaan varmistaa se, onko tehdyn liikkeen liikerata liian lyhyt tai liian pitkä. Keskiarvo lasketaan alkuperäisestä signaalista, aloitus- ja lopetuskohdan välisistä näytearvoista. Keskiarvoistamisen jälkeen voidaan alkaa määrittää todellista aloitus- ja lopetuskohtaa raja-arvojen avulla, missä käytetään hyväksi myös keskiarvoa. Raja-arvotarkastelussa käydään lävitse sekä alkuperäisen signaalin että derivoidun signaaliin jokainen näyte yksi kerrallaan. Raja-arvotarkastelua havainnollistaa kuvan 14 mukainen vuokaavio. Liitteessä 7 on ohjelmallisesti toteutettu raja-arvotarkastelu.



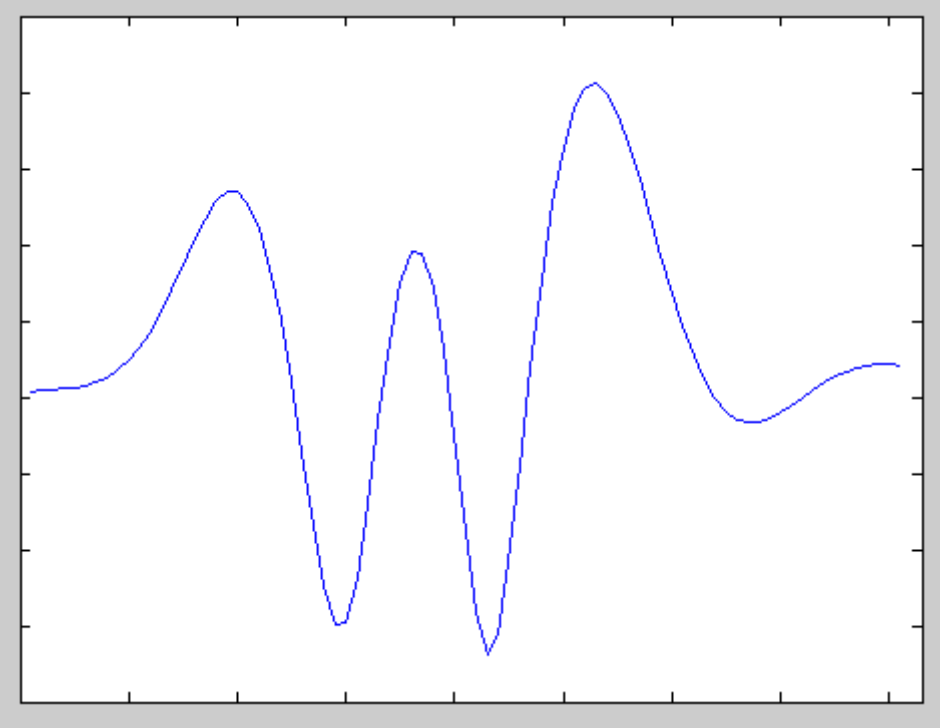
KUVA 14. Raja-arvotarkastelu summasignaalin sekä derivoidun signaalin avulla

Molempien ehtojen pitää täyttyä näytearvon kohdalla, jotta se hyväksytään lopulliseen tulokseen. Tässä vaiheessa vektorissa on jälleen lukuarvo 1 tai 0 kuvan 15 mukaisesti.



KUVA 15. Raja-arvojen avulla muodostunut vektori. Alkion arvo on joko 0 tai 1.

Tässä vaiheessa ollaan periaatteessa taas alkutilanteessa, eli todellinen aloitus- ja lopetuskohta etsitään samalla tavalla kuin jo aikaisemmin on esitetty. Lisätään siis peräkkäisten ykkösien lukumäärä negatiivisena lukuna vektorin kohtiin, joissa lukuarvo muuttuu 1:stä 0:aan, minkä jälkeen lasketaan vektorista kumulatiivinen summa. Sen jälkeen voidaan kumulatiivisen summan perusteella helposti määritellä todellinen aloitus- ja lopetuskohta. Kun kohdat sijoitetaan alkuperäiseen signaaliin, saadaan lopputuloksena kuvan 16 mukainen signaali.



KUVA 16. Signaali, josta on etsitty aloitus- ja lopetuskohta

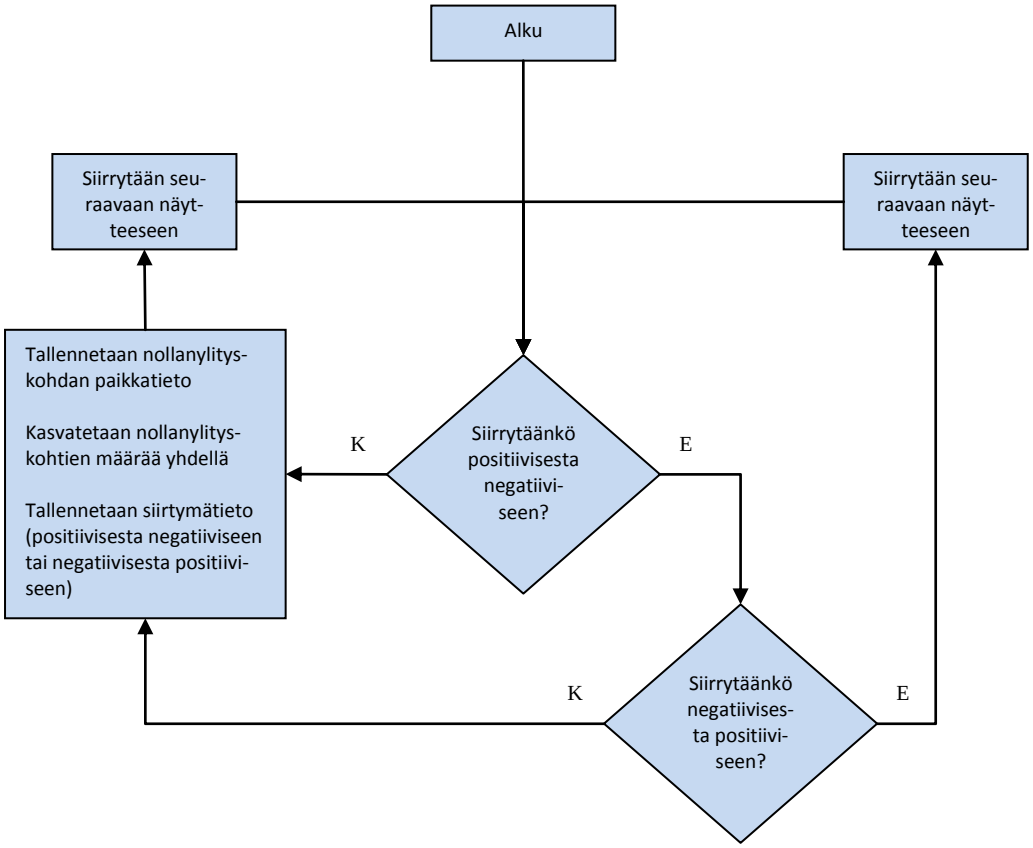
5.5 Paikallisten maksimien ja minimien määrittäminen

Kun aloitus- ja lopetuskohta on määritetty, voidaan alkaa määrittää signaalista paikallisia maksimeja ja minimejä, minkä avulla voidaan tehdä ensimmäinen vertailu referenssi- ja näytesignaalin kesken.

Paikallisten maksimien ja minimien määrittämisessä käytetään hyväksi ensimmäistä derivaattaa. Signaali derivoidaan, jolloin nollanylityskohdat ovat funktion mahdollisia ääriarvokohtia. Seuraavaksi luokitellaan ääriarvokohdat minimikohtiin ja maksimikohtiin. Luokittelussa minimikohtaa ohittaessaan derivaatan arvo muuttuu negatiivisesta tangentin kulmakertoimesta positiiviseksi, ja maksimikohtaa ohittaessaan derivaatan arvo muuttuu positiivisesta tangentin kulmakertoimesta negatiiviseksi. Signaalin nollanylityskohtien pisteiden välistä arvoista lasketaan maksimi- ja minimikohdat. (5.)

5.5.1 Nollanylityskohtien määrittäminen

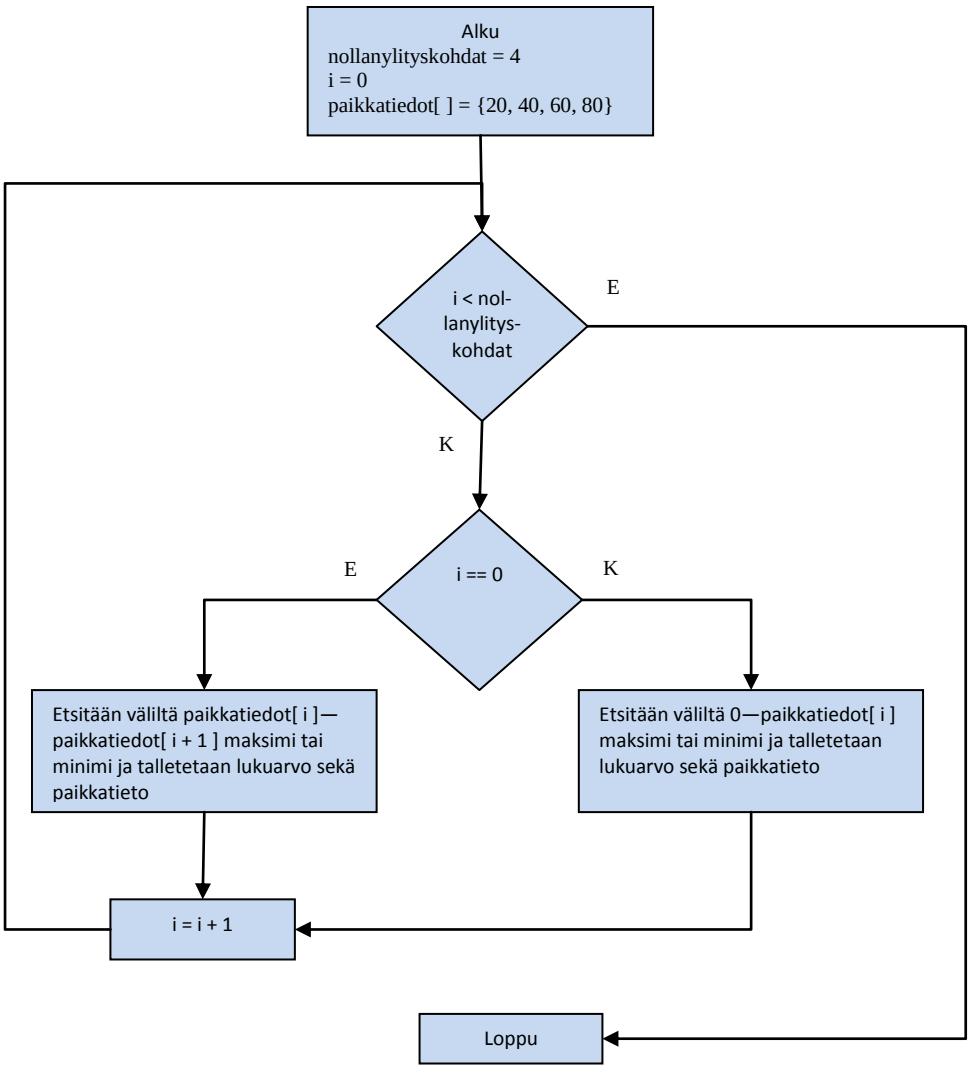
Ohjelmallisessa toteutuksessa tutkitaan aluksi derivoitua signaalia. Tutkimalla derivoidun signaalin tietyn alkion näytearvoa sekä siitä seuraavaa näytearvoa saadaan tietoon nollanylityskohta. Tiettyä muuttujaa kasvatetaan aina, kun nollakohta ylitetään, jolloin saadaan tieto siitä, kuinka monta nollakohtaa signaalissa on. Tallennetaan myös nollanylityskohtien indeksit, jolloin saadaan tieto siitä, missä kohtaa vektoria nollanylitys tapahtuu. Tutkitaan myös samalla, muuttuuko näytteen lukuarvo positiivisesta negatiiviseen vai negatiivisesta positiiviseen. Saatua tietoa tallennettaessa esimerkiksi lukuarvo 0 ilmaisee, että siirryttiin negatiivisesta positiiviseen, ja lukuarvo 1 ilmaisee, että siirryttiin positiivisesta negatiiviseen. Kuva 17 havainnollistaa ohjelman toimintaa. Liitteestä 8 voi tutkia tämän kohdan ohjelmalistausta.



KUVA 17. Nollanylityskohtien tutkiminen derivoidun signaalin avulla

5.5.2 Maksimien ja minimien määrittäminen derivoidusta signaalista

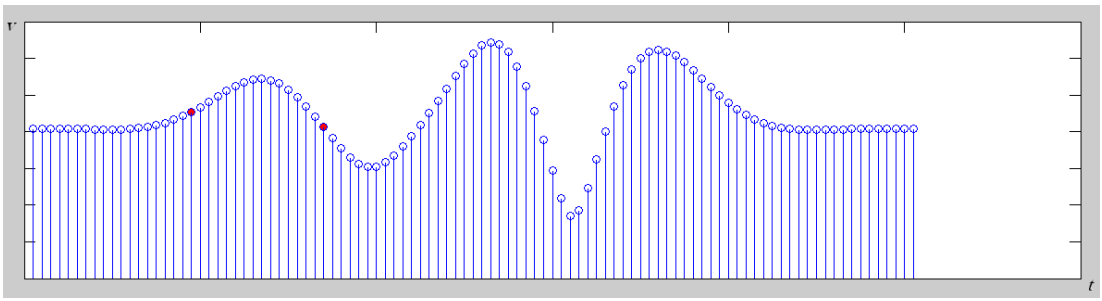
Seuraavaksi määritetään derivoidusta signaalista maksimit ja minimit. Ohjelmaan on tehty silmukka, joka pyörähtää ympäri niin monta kertaa, kuin nollanylityskohtia on. Silmukkaan on tehty sisäinen silmukka, jossa käytetään hyväksi nollanylityskohtien paikkatietoja. Paikkatietojen avulla voidaan silmukkaan määrittää signaalista tutkittava alue. Siirtymätiedon (siirtymä positiivisesta negatiiviseen tai negatiivisesta positiiviseen) avulla voidaan silmukkaan määrittää ehtolause, etsitäänkö paikkatietojen määrittämästä alueesta maksimi vai minimi. Kun maksimi tai minimi on etsitty, tallennetaan sen lukuarvo sekä paikkatieto ja siirrytään tutkimaan seuraavaa aluetta. Kuvassa 18 on esimerkkitapaus ohjelman toiminnasta. Liitteessä 9 on tämän kohdan ohjelmalistaus.



KUVA 18. Maksimien ja minimien etsiminen derivoidusta signaalista.

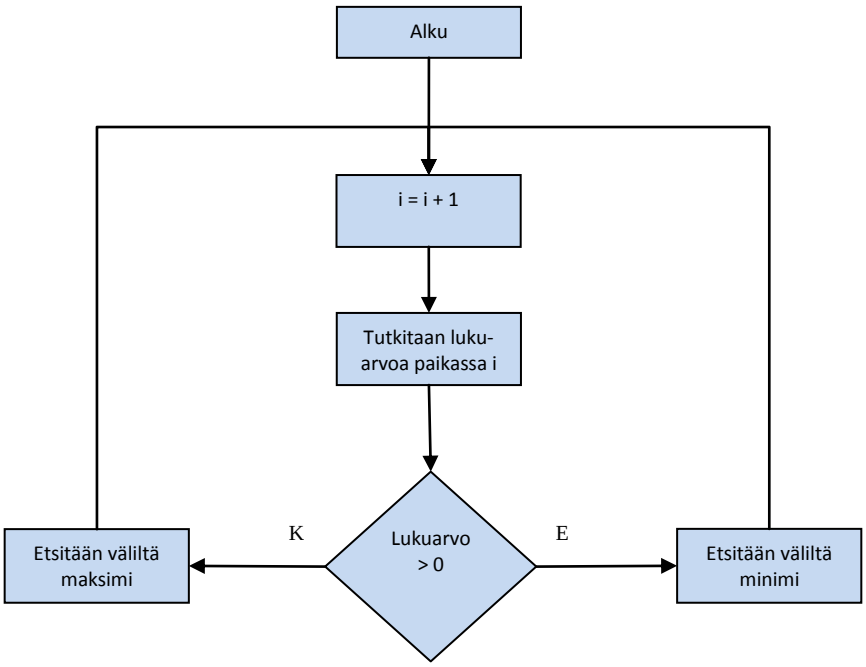
5.5.3 Maksimien ja minimien määrittäminen alkuperäisestä signaalista

Kun derivoidusta signaalista on saatu etsittyä maksimi- ja minimikohdat, voidaan niiden avulla määrittää maksimi- ja minimiarvot alkuperäisestä signaalista. Kun derivoidun signaalin ensimmäinen maksimi- ja minimikohta sijoitetaan alkuperäiseen signaaliin, kohdat sijoittuvat siihen kuvan 19 mukaisesti.



KUVA 19. Derivoidusta signaalista saadut paikkatiedot on sijoitettu alkuperäiseen signaaliin

Kun pisteiden väliltä etsitään maksimi, saadaan alkuperäisestä signaalista oikea maksimiarvo. Samalla tekniikalla käydään lävitse koko signaali, jolloin saadaan määritettyä kaikki maksimi- ja minimiarvot. Kuva 20 havainnollistaa yksinkertaisesti, miten ohjelma on toteutettu. Liitteessä 10 on ohjelmalistaus maksimien ja minimien määrittämisestä alkuperäiseen signaaliin.

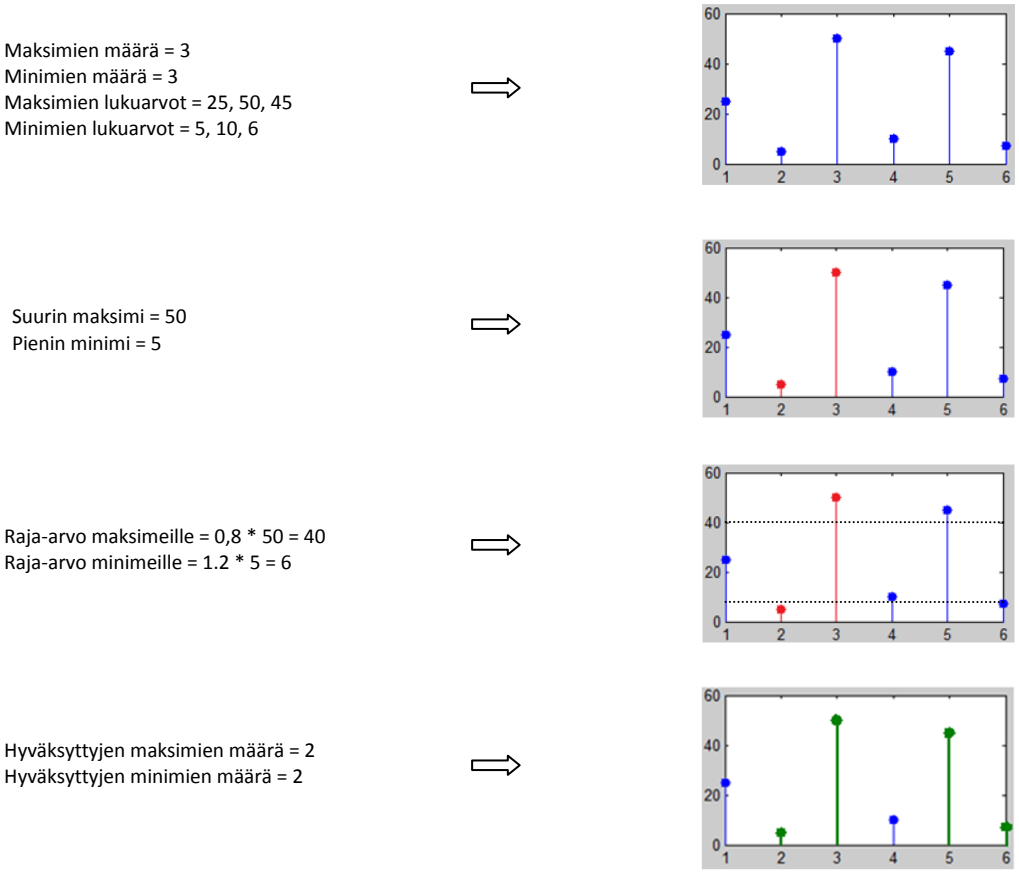


KUVA 20. Maksimien ja minimien etsiminen alkuperäisestä signaalista

5.5.4 Maksimien ja minimien suodatus

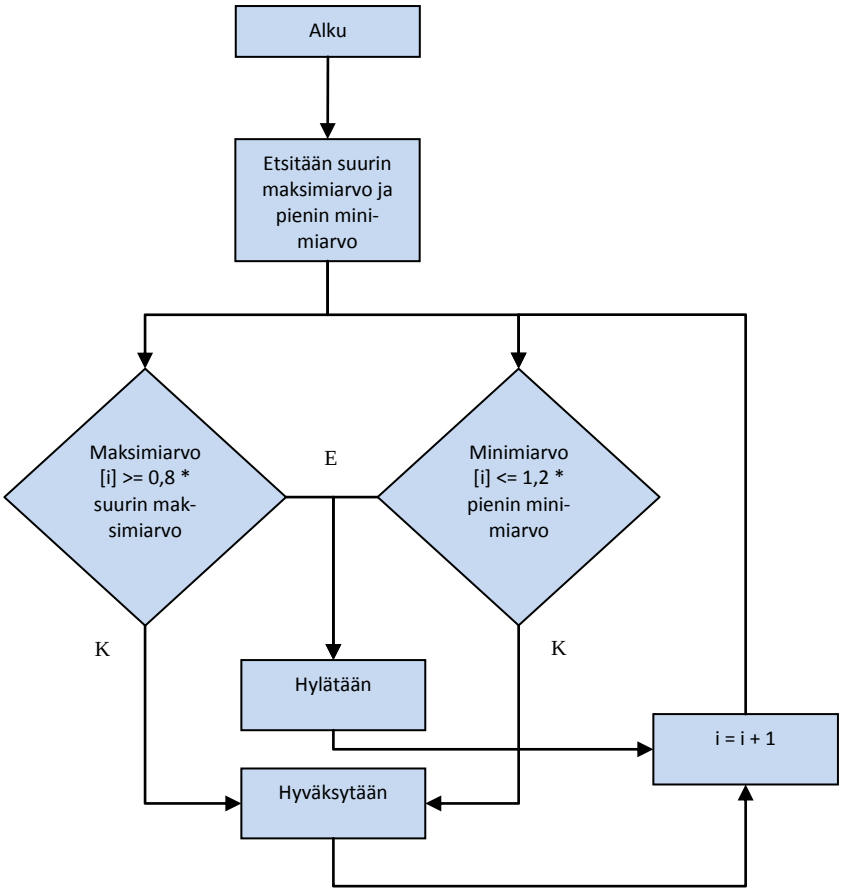
Kun maksimi- ja minimiarvot on saatu etsittyä sekä referenssisignaalista että näytesignaalista, niille tehdään pienimuotoinen suodatus. Tarkoituksena on suodattaa pois kaikista vähiten merkitsevät maksimi- ja minimiarvot. Tämä ei välttämättä ole kaikista paras mahdollinen menetelmä, mutta sillä saadaan kuitenkin huomioitua isoimmat huiput tai laaksot, joilla on yleensä eniten merkitystä muotojen tunnistamisen kannalta.

Suodatus tehdään suurimman maksimiarvon ja pienimmän minimiarvon avulla. Lukuarvoja kerrotaan tietyllä kertoimella ja saatuja tuloksia käytetään raja-arvoina. Esimerkiksi jos halutaan, että kaikkien maksimiarvojen täytyy olla vähintään 80 % suurimmasta maksimiarvosta, suurinta maksimiarvoa kerrotaan 0,80:llä. Jokainen maksimiarvo, joka ylittää raja-arvon, hyväksytään maksimiarvoksi. Kuvassa 21 on havainnollistava esimerkki minimien ja maksimien suodatuksesta



KUVA 21. Maksimien ja minimien suodatus

Kuvassa 22 oleva vuokaavio demonstroi ohjelmallista toteutusta, joka löytyy liitteestä 11.

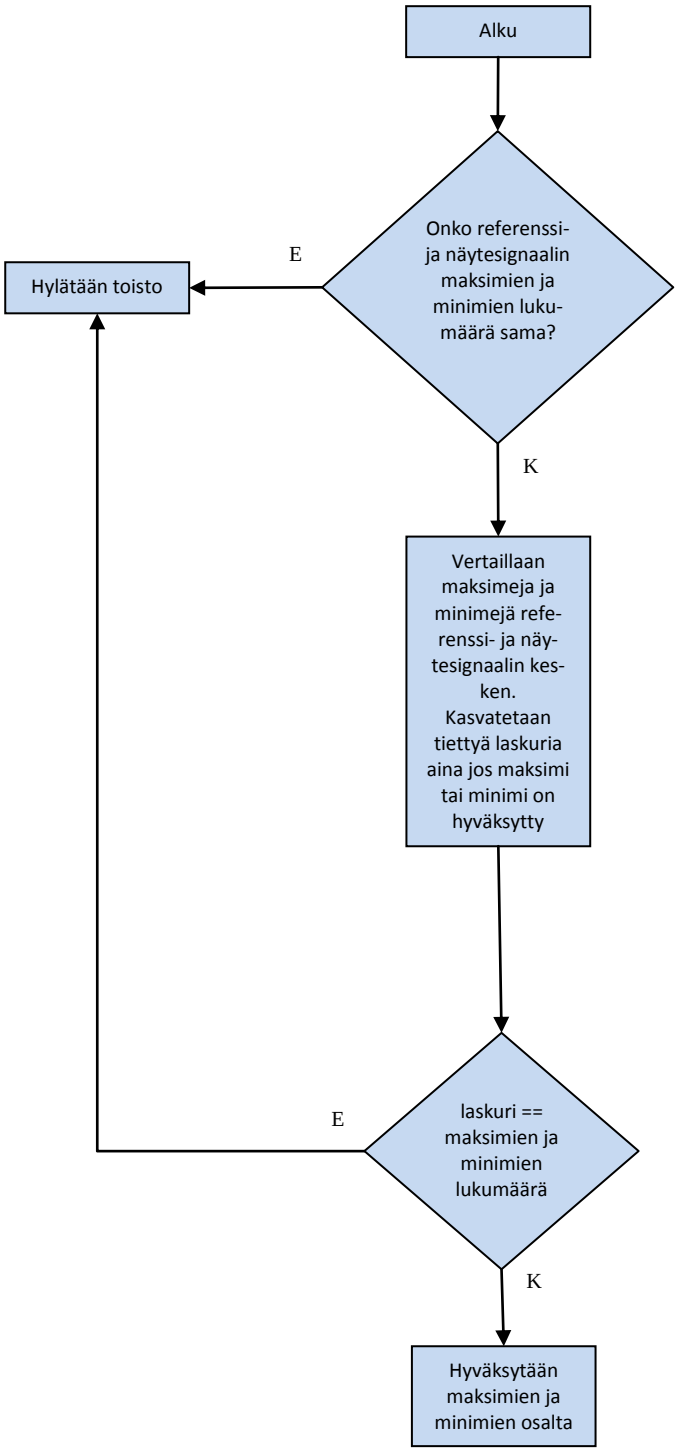


KUVA 22. Vuokaavio maksimien ja minimien suodatuksesta

5.5.5 Vertailu maksimien ja minimien avulla

Suodatettujen maksimien ja minimien avulla voidaan tehdä vertailu referenssisignaalin ja näytesignaalin kesken. Ensimmäinen vertailu voidaan tehdä jo maksimien ja minimien lukumäärään perustuen, koska niitä tulisi olla aina sama määrä. Mikäli määrä on eri, toisto määritetään virheelliseksi. Jos määrä on kuitenkin sama molemmissa, vertailussa siirrytään eteenpäin. Vertailussa tutkitaan keskenään referenssi- ja näytesignaalin maksimien ja minimien lukuarvoja. Raja-arvot muodostuvat kertomalla referenssisignaalin maksimi- ja minimiarvoja tietyllä kertoimella. Jos esimerkiksi halutaan, että näytesignaalin maksimiarvojen täytyy olla vähintään 90 % referenssisignaalin maksimiarvoista, kerrotaan referenssisignaalin maksimiarvoja 0,90:llä, jolloin saadaan kaikille maksimeille raja-arvot. Vastaavasti minimiarvoja kerrotaan 1,1:llä, jolloin saadaan kaikille minimeille raja-arvot.

Ohjelmassa kasvatetaan aina tiettyä laskuria, kun raja-arvo ylittyy. Kun kaikki maksimi- ja minimiarvot on tutkittu, täytyy laskurin arvon olla sama kuin aikaisemmin maksimien ja minimien lukumäärä. Sen avulla voidaan tehdä helposti viimeinen päätös siitä, hylätäänkö tulos vai jatketaanko eteenpäin. Kuvassa 23 on esitetty yksinkertaistettu vuokaavio vertailun toiminnasta. Liitteessä 12 on ohjelmalistaus maksimien ja minimien vertailusta.



KUVA 23. Maksimien ja minimien vertailu referenssi- ja näytesignaalin kes-ken

5.6 Ristikorrelaatio

Kun maksimien ja minimien vertailu on saatu käytyä hyväksyttävästi lävitse, seuraavaksi vertailua tehdään ristikorrelaation avulla. Ristikorrelaation avulla voidaan tutkia kahden signaalin samankaltaisuutta. Mitä paremmin signaalit muistuttavat toisiaan, sitä suurempi ristikorrelaation tulos on. Tämän vaiheen perusteella tehdään viimeinen päätös siitä, hyväksytäänkö näytesignaali hyväksytyksi vai virheelliseksi toistoksi.

5.6.1 Ristikorrelaatio akseleittain

Ristikorrelaatio tehdään referenssi- ja näytesignaalin jokaiselle akselille erikseen, jotta siitä saadaan paras mahdollinen hyöty. Ristikorrelaatio täytyy tehdä yhteensä yhdeksän kertaa taulukon 1 mukaisesti.

TAULUKKO 1. Ristikorrelaatiot akseleittain. R tarkoittaa referenssisignaalia ja N näytesignaalia.

	X-akseli	**Y-akseli**	**Z-akseli**
Ristikorrelaatio X1	R & N		
Ristikorrelaatio X2	R & R		
Ristikorrelaatio X3	N &N		
Ristikorrelaatio Y1		R & N	
Ristikorrelaatio Y2		R & R	
Ristikorrelaatio Y3		N &N	
Ristikorrelaatio Z1			R & N
Ristikorrelaatio Z2			R & R
Ristikorrelaatio Z3			N &N

Kaikista ristikorrelaatioista etsitään maksimiarvo, minkä jälkeen niille tehdään kaavojen 4, 5 ja 6 mukaiset laskuoperaatiot.

$$X = \frac{Ristikorrelaatio\ X1}{\sqrt{Ristikorrelaatio\ X2 \cdot Ristikorrelaatio\ X3}}$$ *KAAVA 4.*

$$Y = \frac{Ristikorrelaatio\ Y1}{\sqrt{Ristikorrelaatio\ Y2 \cdot Ristikorrelaatio\ Y3}}$$ *KAAVA 5.*

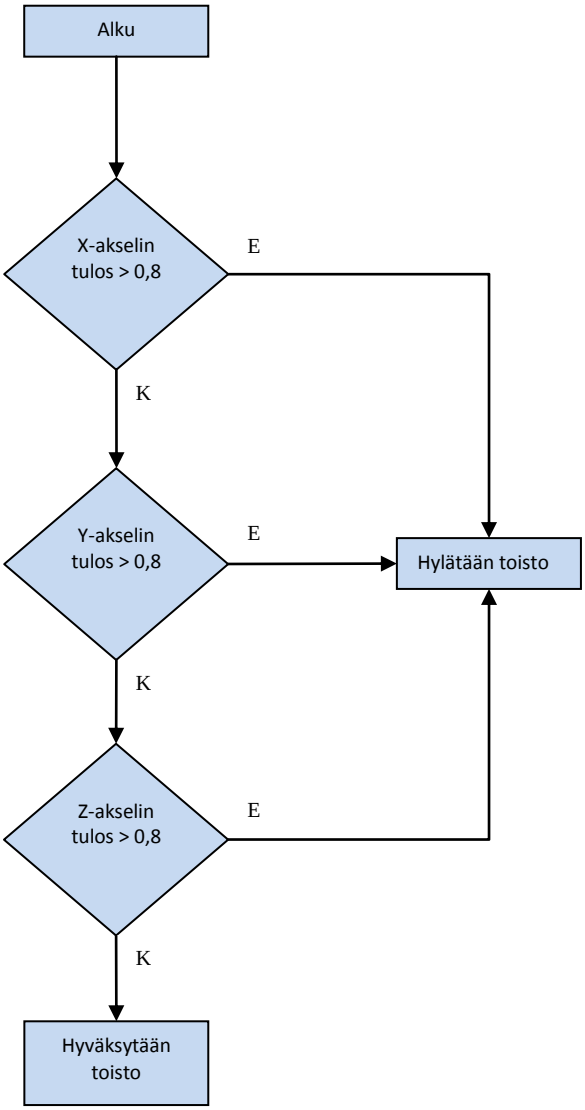
$$Z = \frac{Ristikorrelaatio\ Z1}{\sqrt{Ristikorrelaatio\ Z2 \cdot Ristikorrelaatio\ Z3}}$$ *KAAVA 6.*

Operaatioiden avulla jokaisen akselin tulokset saadaan normalisoitua välille 0—1, minkä jälkeen on helppoa tehdä vertailu.

Liitteessä 13 on ohjelmalistaus ohjelmasta, joka laskee ristikorrelaation kahden vektorin kesken ja palauttaa maksimiarvon. Liitteessä 14 on ohjelmalistaus laskuoperaatioista.

5.6.2 Vertailu ristikorrelaation avulla

Kun ristikorrelaatiot on laskettu, voidaan niiden perusteella tehdä helposti päätelmiä, onko toisto hyväksytty vai hylätty. Vertailuarvot ovat nyt normalisoituneet välille 0—1. Mitä suurempi luku on, sitä enemmän signaalit muistuttavat toisiaan. Näin ollen vertailuun voidaan yksinkertaisesti asettaa jokin tietty raja-arvo, joka täytyy ylittyä jokaisen akselin kohdalla. Kuva 24 demonstroi vertailua. Ohjelmallisesti vertailu voidaan toteuttaa yksinkertaisimmillaan yhdellä ehtolauseella.



KUVA 24. Vertailu ristikorrelaation avulla

5.7 Testaus

Algoritmin toimintaa testattiin penkkipunnerrusliikkeellä. Taulukossa 2 on tulokset testistä, jossa on tehty 10 toiston sarjoja mahdollisimman hyvin. Sarjojen alussa referenssiliike on tehty aina erikseen. Tuloksista voi huomata, että ohjelma on hyväksynyt toistoja suhteellisen hyvin.

TAULUKKO 2. Penkkipunnerrusliikkeellä tehdyt testitulokset. Liikkeet on pyritty tekemään mahdollisimman hyvin.

	Toistot	Onnistuneet	Virheelliset
Sarja 1	10	10	0
Sarja 2	10	10	0
Sarja 3	10	9	1
Sarja 4	10	8	2
Sarja 5	10	10	0
Sarja 6	10	10	0
Sarja 7	10	9	1
Sarja 8	10	10	0
Sarja 9	10	10	0
Sarja 10	10	10	0

Taulukossa 3 on tulokset testistä, jossa on tehty 10 toiston sarjoja tietoisesti virheellisesti. Sarjojen alussa referenssiliike on tehty aina erikseen. Tuloksista voi havaita, että ohjelma huomaa virheelliset liikkeet suhteellisen hyvin. Toistoja tehdessä virheliikkeen on oltava melko suuri. Testejä tulisi tehdä esimerkiksi fysioterapeuttien kanssa, jotta virheellisen liikkeen määrääviä raja-arvoja voitaisiin säätää tarkemmin.

TAULUKKO 3. Penkkipunnerrusliikkeellä tehdyt testitulokset. Liikkeet on tehty tietoisesti väärin.

	Toistot	Onnistuneet	Virheelliset
Sarja 1	10	0	10
Sarja 2	10	2	8
Sarja 3	10	1	9
Sarja 4	10	1	9
Sarja 5	10	0	10
Sarja 6	10	1	9
Sarja 7	10	1	9
Sarja 8	10	0	10
Sarja 9	10	0	10
Sarja 10	10	1	9

6 YHTEENVETO JA JATKOKEHITYS

Tässä insinöörityössä toteutettiin maksimivoimamittariin toistomäärälaskurin Matlab-ohjelmistolla toteutettu algoritmi C-ohjelmointikielellä. Algoritmin on kehittänyt Veli-Pekka Välimaa. C-kielinen versio toteutettiin reaaliaikaiseksi, jotta sitä voidaan hyödyntää, kun algoritmia aletaan toteuttaa sulautettuun laitteeseen. Algoritmin toimintaperiaate perustuu käyttäjän tekemään referenssiliikkeeseen, johon käyttäjän muita suoritteita verrataan. Suurimmat vertailupäätökset tehdään paikallisten maksimien ja minimien sekä ristikorrelaation avulla.

Ennen kuin referenssisignaalia aletaan käsitellä, se täytyy alipäästösuodattaa, koska kiihtyvyysanturilta saatu data on häiriöistä. Alipäästösuodatus täytyy tehdä myös jokaiselle näytesignaalille. Alipäästösuodatus tehdään jokaiselle akselille erikseen, minkä jälkeen niistä tehdään summasignaali. Summasignaalista karsitaan pois ylimääräinen data eli etsitään signaalista aloitus- ja lopetuskohta. Tämän jälkeen määritetään paikalliset maksimit ja minimit, joiden avulla tehdään ensimmäinen vertailu referenssi- ja näytesignaalin kesken. Jos tulokset hyväksytään, siirrytään vertailemaan ristikorrelaatioita referenssi- ja näytesignaalin kesken. Ristikorrelaatiota vertaillaan jokaisen akselin kohdalla erikseen. Jos ristikorrelaation tulokset hyväksytään, todetaan näytesignaali hyväksytyksi eli onnistuneeksi toistoksi.

Pienissä sulautetuissa laitteissa on yleensä hyvin rajallinen määrä muistia. Matlab-ohjelmaa ei alettu kääntää suoraan Matlabin C-kääntäjällä, koska kääntäminen ei ole esimerkiksi muistien kannalta kovin optimaalinen. Matlabin C-kääntäjä tekee suhteellisen suuren erillisen kirjastotiedoston, jota täytyisi käyttää. Jos vertaillaan Matlabin tekemää C-ohjelmakoodia sekä tässä työssä toteutettua C-ohjelmakoodia, päästään tässä työssä tehdyllä koodilla noin 80 % pienempään tiedostokokoon.

Algoritmia kehitetään jatkuvasti ja kehitystyötä tehtiin myös samaan aikaan, kun tätä insinöörityötä tehtiin. Myös laitteen muista ominaisuuksista on alettu

tehdä tutkimustöitä ja opinnäytetöitä. Samaan aikaan kun tätä insinöörityötä tehtiin, insinööriopiskelija Arto Tiitto alkoi jatkokehittää Veli-Pekka Välimaan algoritmia Matlab-ohjelmistolla. Kehitystyössä tutkittiin jatkuvaa Wavelet-muunnosta (Continuous Wavelet Transform). Tästä saadaan apua esimerkiksi lyhyiden ja nopeiden toistojen väliseen tarkkuuteen. Jos toistoja tehdään esimerkiksi 30, viimeisimmillä toistoilla liikkeen aika saattaa pidentyä, kun taas sarjan alussa liikkeet ovat todennäköisesti lyhempiä. Nykyinen algoritmi tulkitsee tällaiset toistot yleensä virheellisiksi ristikorrelaation takia.

Vielä on mahdotonta sanoa, onko Wavelet-muunnokseen perustuva algoritmi viimeisin versio algoritmista. Haasteelliseksi voi myös muodostua Wavelet-muunnoksen kääntäminen C-ohjelmointikielelle. Wavelet-muunnokseen perustuvan algoritmin valmistuttua tullaan tekemään päätöksiä, jatketaanko toistomäärälaskurin toteutusta Veli-Pekka Välimaan algoritmilla vai vaihdetaanko algoritmin toteutus. Mikäli algoritmia päätetään vaihtaa, Välimaan kehittämä algoritmi tulee muuttumaan noin 25–33 %.

7 POHDINTA

Työn alkaessa epäröin hieman omia ohjelmointitaitojani, koska langattomaan tietoliikenteeseen suuntautuneilla opiskelijoilla ohjelmointiin liittyviä opintojaksoja on vähemmän kuin ohjelmistokehitykseen suuntautuneilla opiskelijoilla. Aihe vaikutti kuitenkin erittäin mielenkiintoiselta, joten päätin tarttua haasteeseen. Signaalinkäsittelyn hallitsen mielestäni suhteellisen hyvin, joten siltä osa-alueelta ei koitunut haasteita niin paljoa.

Alun perin oli tarkoitus toteuttaa algoritmi C-ohjelmointikielelle ja toteuttaa se sulautettuun laitteeseen. Työn edetessä ja asiaan perehtyessä kävi kuitenkin ilmi, että työmäärä olisi liian iso ja työstä tulisi liian laaja, jos toteutukseen asti mentäisiin. Työmäärää aiheessa on esimerkiksi toiseen opinnäytetyöhön.

Yksi haaste oli päästä alkuun ja ymmärtää Matlab-ohjelmalla toteutetun koodin toiminta. Ohjelmassa on käytetty paljon Matlabin omia funktioita, joiden toiminta täytyi selvittää tarkkaan ennen C-kielistä toteutusta. Lisää haastetta toi myös reaaliaikaisuus. Raja-arvot täytyi asettaa tarkasti, jotta liikkeiden välissä osataan tallentaa signaali oikeasta kohti. Raja-arvot tulisi toteuttaa jollain tavalla dynaamisiksi liikesarjojen nopeuden mukaan. Näin saataisiin parempia tuloksia käyttäjän kannalta.

Näin jälkikäteen ajatellen monia asioita olisi voinut toteuttaa paljon eri tavoilla, mutta yksi opinnäytetyön tavoitehan on kehittää opiskelijan tietoja ja taitoja. Tietyissä kohdissa järkevin tapa olisi ollut suunnitella aluksi paperille jonkinlainen vuokaavioesitys, jonka pohjalta koodi olisi ollut helpompaa kirjoittaa. Sen sijaan tein asiat lähes aina päinvastaisessa järjestyksessä. Kuljin myös aika pitkälti käsi kädessä Matlab-koodin mukana rivi riviltä. Jälkeenpäin ajateltuna jotkin kohdat olisi voinut toteuttaa varmasti rivimäärältään hieman lyhyemmilläkin ohjelmakoodeilla, erityisesti työn alussa tehdyt käännökset. Loppuvaiheessa käännöstä aloin myös miettiä optimaalisempia toteutustapoja.

LÄHTEET

1. Hannula, Manne 2011. Tutkijayliopettaja, Oulun seudun ammattikorkeakoulu, tekniikan yksikkö, MUSTI-projekti.

2. eZ430-Chronos™ Development Tool, User's Guide. 2010. Saatavissa: http://focus.ti.com/lit/ug/slau292c/slau292c.pdf. Hakupäivä 1.2.2011.

3. Kannela, Sami 2009. Hissin ajokäyrän raja-arvojen määrittäminen. Saatavissa: https://publications.theseus.fi/bitstream/handle/10024/4336/Hissin%20ajokayran%20raja-arvojen%20maarittaminen.pdf?sequence=1. Hakupäivä 17.3.2011.

4. Filter. Saatavissa: http://www.mathworks.com/help/techdoc/ref/filter.html. Hakupäivä 17.1.2011.

5. Pitkänen, Jaakko 1997. Muutosten matematiikka. Saatavissa: http://nww.evtek.fi/n/jaakkop/TM03S/MUUMAT11.doc. Hakupäivä: 10.1.2011.

LIITTEET

LIITE 1 (OAMK:n tekniikan yksikön käytössä)

LIITE 2 (OAMK:n tekniikan yksikön käytössä)

LIITE 3 (OAMK:n tekniikan yksikön käytössä)

LIITE 4 (OAMK:n tekniikan yksikön käytössä)

LIITE 5 (OAMK:n tekniikan yksikön käytössä)

LIITE 6 (OAMK:n tekniikan yksikön käytössä)

LIITE 7 (OAMK:n tekniikan yksikön käytössä)

LIITE 8 (OAMK:n tekniikan yksikön käytössä)

LIITE 9 (OAMK:n tekniikan yksikön käytössä)

LIITE 10 (OAMK:n tekniikan yksikön käytössä)

LIITE 11 (OAMK:n tekniikan yksikön käytössä)

LIITE 12 (OAMK:n tekniikan yksikön käytössä)

LIITE 13 (OAMK:n tekniikan yksikön käytössä)

LIITE 14 (OAMK:n tekniikan yksikön käytössä)