Teknillinen korkeakoulu
Akustiikka ja äänenkäsittelytekniikka
S-89.122 Signaaliprosessorit ja äänenkäsittely, 1996

Mr Dave Direction, Hr Wilken Direksson, Hra Keino Suuntonen

Topi Kaaresoja, TiN
36281M
Panu Maijala, SN
37847W

10.5.1996

Sisällysluettelo

Herra Keino Suuntonen

Topi Kaaresoja
tka@cc.hut.fi
Panu Maijala
pantse@cc.hut.fi
TKK

Tiivistelmä

Tässä työssä toteutimme suuntakuuntelijan signaaliprosessoria ohjelmoiden. Ohjelma laskee ristikorrelaation ja tämän perusteella ITD:n (Interaural Time Delay) kahden (kummankin korvan) signaalin välille. Lopuksi käännetään aikaeroa vastaavan poikkeaman verran askelmoottorilla suuntakuuntelijaa.

1. Johdanto

Tiedekeskus Heurekaan oli tulossa ”Illuusiot”- aivojen toimintaa kuvaava näyttely. Professori Matti Karjalaisen idea oli saada näyttelyyn laite, joka esittelee ihmisen suuntakuuloa. Laite olisi pää, joka kääntyisi havaitsemansa äänen suuntaan. Työn tarkoitus oli tehdä assembler-ohjelma signaalinkäsittelyprosessorille, joka ohjaa askelmoottorin kääntämänä päätä.

Toteutuksessa päädyttiin Motorolan DSP56002EVM-korttiin, joka on valmis, toimiva kokeilukortti. Algoritmin valinta oli suoraviivainen: *ristikorrelaation* laskeminen korviin tulevien signaalien välille ja lopputuloksen analysoinnin perusteella askelmoottorin kääntäminen.

Työ osoittautui kuitenkin varsin laajaksi. Karkeasti tehdyn työn voisi jakaa seuraavasti:

- ratkaisumenetelmän valinta ja sen simulointi Matlab-ympäristössä (30h)
- Motorolan ympäristöön tutustuminen, algoritmin valinta ja ohjelmointi (700h)
- mekaanisen rakenteen valinta ja toteutus (30h)
- akustiset mittaukset (HRTF) ja tulosten käsittely Matlab-ympäristössä (15h)

2. Käsitteitä

2.1 Interaural Time Delay

Ihmisen suuntakuulo perustuu pääasiassa binauraaliseen kuulemiseen. Tärkein tekijä suunnan havaitsemiseksi on äänilähteestä tulevan signaalin aikaero, Interaural Time Delay *(ITD),* korvien välillä. Signaalinkäsittelyn kannalta selkein keino ilmaista viive kahden signaalin välillä on *ristikorrelaation* laskeminen niiden kesken.

2.2 Ristikorrelaatio

Ristikorrelaatio on matemaattinen funktio, jota voidaan käyttää mm. kahden samanmuotoisen signaalin aikaeron (viiveen) esiintuomiseen. Ristikorrelaatio voidaan ajatella siten, että kaksi signaalia liukuu toistensa ohi samaan tapaan kuin konvoluutiossa. Kun samanmuotoiset signaalit ovat kohdakkain, on ristikorrelaation arvo suurimmillaan. Tämän maksimin sijainti nollaan nähden on täsmälleen signaalien välinen viive.

Ristikorrelaatio voidaan määritellä seuraavasti:

Olkoon x1(n) ja x2(n) samanmuotoiset diskreetit signaalit, jolloin niiden välinen ristikorrelaatio r12(n) on

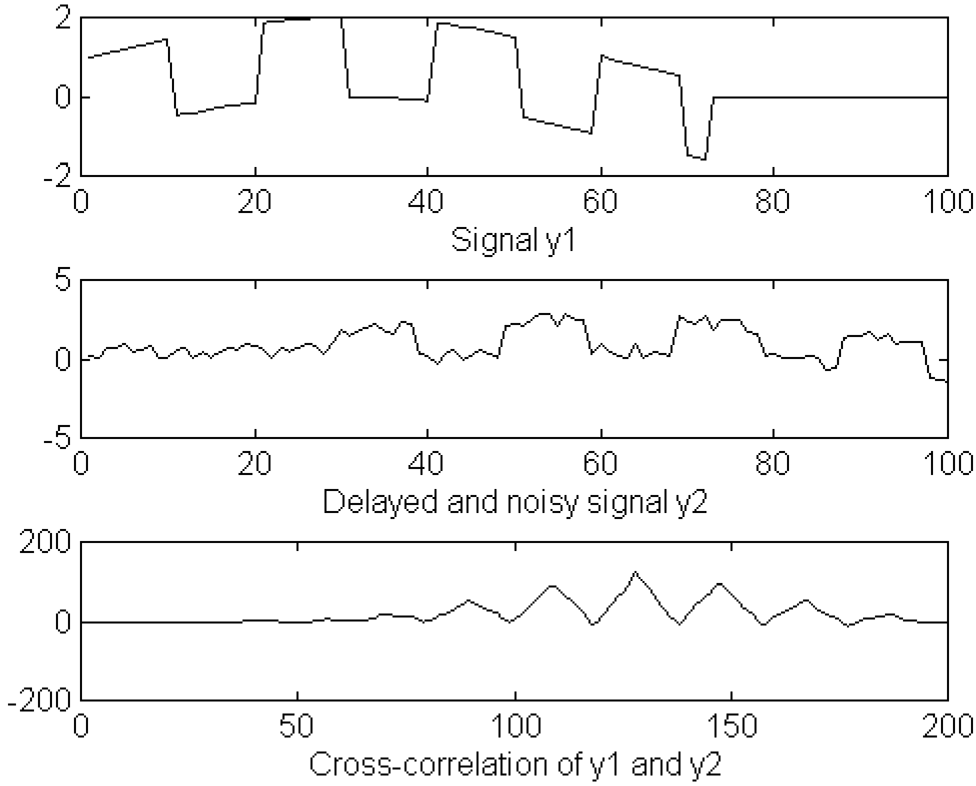
$$r_{12}(j) = \frac{1}{N}\sum_{n=0}^{N-1} x_1(n)x_2(n+j) \tag{1}$$

jossa j= -(N-1), -(N-2), ..., N-2, N-1

Jos tätä vertaa tätä konvoluution kaavaan

$$y(n) = \sum_{m=0}^{N-1} x(m)h(n-m) \tag{2}$$

jossa x(n) on signaali ja h(n) suotimen impulssivaste, todetaan, että kaavat ovat melko samanmuotoiset, jolloin ohjelmointi signaaliprosessorille sujuu samaan tapaan ristikorrelaatiolle ja konvoluutiolle. Kuvassa 1 on esimerkki aikaerosta ja ristikorrelaatiosta. Signaali y2 on hieman viivästetty signaali y1 moduloituna vaihtelevalla häiriölähteellä. Häiriöstä huolimatta ristikorrelaation maksimi on helposti löydettävissä.



Kuva 1. Esimerkki aikaerosta ja ristikorrelaatiosta signaalien y1 ja y2 välillä.

Tässä työssä ristikorrelaatiota käytettiin ITD:n laskemiseen. Jos pää oletetaan palloksi, jonka halkaisija on D, ITD on verrannollinen äänen tulokulmaan approksimaatiolla (Karjalainen, s.23)

$$\tau = \frac{D}{2c}\left[\varphi + \sin(\varphi)\right] \tag{3}$$

missä: τ = ITD
D on pään halkaisija
c on äänen nopeus

Maksimi ITD saadaan, kun ääni tulee sivulta, jolloin $\tau \approx 700\mu s$.

3. Ohjelman rakenne ja versiot

3.1 Ristikorrelaation laskeminen

3.1.1 Ensimmäinen versio

Ensimmäisessä versiossa ristikorrelaatiota laskettiin sirkulaaristen puskurien avulla siten, että käytettiin vain yhtä puskuriparia, molemmille kanaville siis omansa. Puskureihin piti lisätä N-1 (N = näytejonon pituus) nollaa, koska ristikorrelaatiota laskettiin sirkulaarisesti, aivan kuten sirkulaarista konvoluutiotakin laskettiin. Kun puskurit oli luettu täyteen, keskeytettiin lukeminen ja alettiin laskea ristikorrelaatiota hyvin suoraviivaisesti. Etuna oli osoittimien suoraviivainen käyttö ja helposti ymmärrettävä toiminta. Ratkaisu oli kuitenkin erittäin hidas ja tehoton.

3.1.2 Lopullinen versio

Lopulliseksi ratkaisuksi ristikorrelaation laskemiselle muodostui sellainen, että käytetään kahta puskuriparia, joista toiseen luetaan näytteitä samalla kun toisesta lasketaan ristikorrelaatiota. Ristikorrelaation lasku on hajautettu näytteidenluvun väliin siten, että jokaisen näytteenluvun välissä lasketaan kaksi kappaletta ristikorrelaation arvoja paitsi viimeisellä kerralla, jossa ei ole laskettavana kuin yksi, koska ristikorrelaatiovektorin pituus on 2*N-1.

Lopullisessa versiossa luovuttiin myös sirkulaarisesta laskennasta osoittimien käytön helppouden kustannuksella. Osoittimia joudutaan siirtelemään melkoisesti aina yhden ristikorrelaation laskun välissä. Ratkaisu on kuitenkin huomattavasti tehokkaampi kuin edellinen, jossa kolme neljäsosaa tehdyistä kertolaskuista oli nollalla kertomisia.

Laskeminen tapahtuu siten, että alussa toinen laskemiseen käytetyistä osoittimista osoittaa X-muistissa sijaitsevan puskurin (xseq) alkuun ja toinen Y-muistissa sijaitsevan puskurin (yseq) loppuun. Ensimmäinen ristikorrelaation arvo on yksinkertaisesti osoittimien osoittaminen lukujen tulo. Tämän jälkeen siirretään yseq:n osoitinta yhdellä taaksepäin ja lasketaan kahden kertolaskun summa:

xseq(1)*yseq(N-1)+xseq(2)*yseq(N)

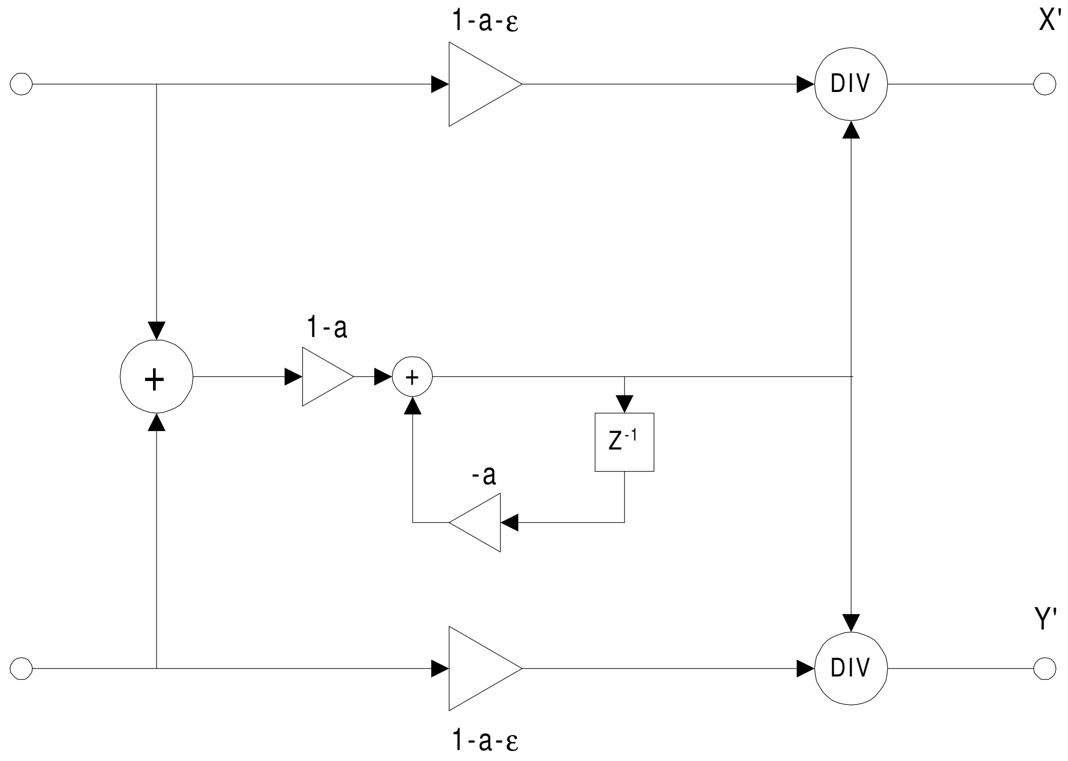
Tässä indeksit tarkoittavat puskureiden alkioita l. näytteitä.

Seuraavaksi siirretään taas yseq:n osoitinta yhdellä taaksepäin ja lasketaan kolmen kertolaskun summa jne.

Edellä kuvattu algoritmi on helppo ymmärtää, jos ajattelee puskureiden ikäänkuin liukuvan toistensa ohi ja aina kohdakkain olevat alkiot kerrotaan keskenään ja kertolaskut lasketaan yhteen.

3.2 Tasonsäätö ja alukkeenkorostusrutiini

Ohjelmaan tehtiin myös tasonsäätö ja alukkeenkorostusrutiini, mutta lopullisessa versiossa sitä ei käytetty, koska sitä ei saatu toimimaan. Sen periaate olisi ollut se, että ensimmäiset näytteet menevät läpi sellaisenaan, mutta sen jälkeen takaisinkytketyn suodattimen läpi tasoa aletaan laskea sen mukaan, mitä sisään tulee. Ohessa rutiinin kaaviokuva.



Kuva 2. Kaavio GainCtrl -algoritmista

3.3 Askelmoottorin ohjaus

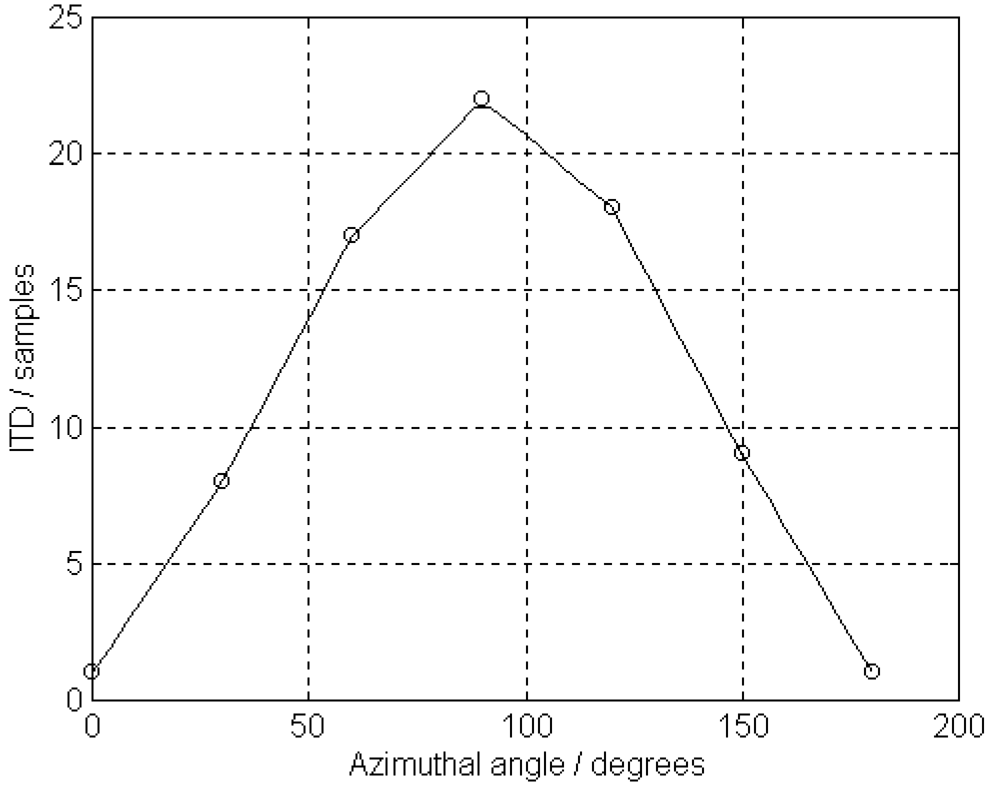
3.3.1 Ohjaimelle menevät signaalit

Askelmoottorin ohjain vaati sisään suunnan sekä moottoria käännettävien pulssien määrän TTL-tasoisina pulsseina. Suunta oli joko ‘1’ tai ‘0’ ja kääntymisen suuruus kutakin pulssia kohden määräytyi ohjaimesta ennalta asetetun kääntökytkimen mukaan.

3.3.2 Askelten lukumäärä

Kutakin kulmapoikkeamaa vastaavat askeleet määritettiin aluksi karkeasti mittaamalla 30 asteen välein ITD laitteistoon käytetystä muovipäästä ja tämän jälkeen lineaarisella interpoloinnilla laskettiin tarvittavat kulma-askel vastaavuudet. Kuvassa 3 on laskettuna ITD Keinolle. Aika ero on näytteiden lukumääränä ja käytetty näytetaajuus on 48 kHz. Tämän mukaan Keinon korvien etäisyys on

$$r = 340\frac{m}{s} * \left(\frac{48000\frac{1}{s}}{22} \right) \approx 15.6cm \qquad (4)$$

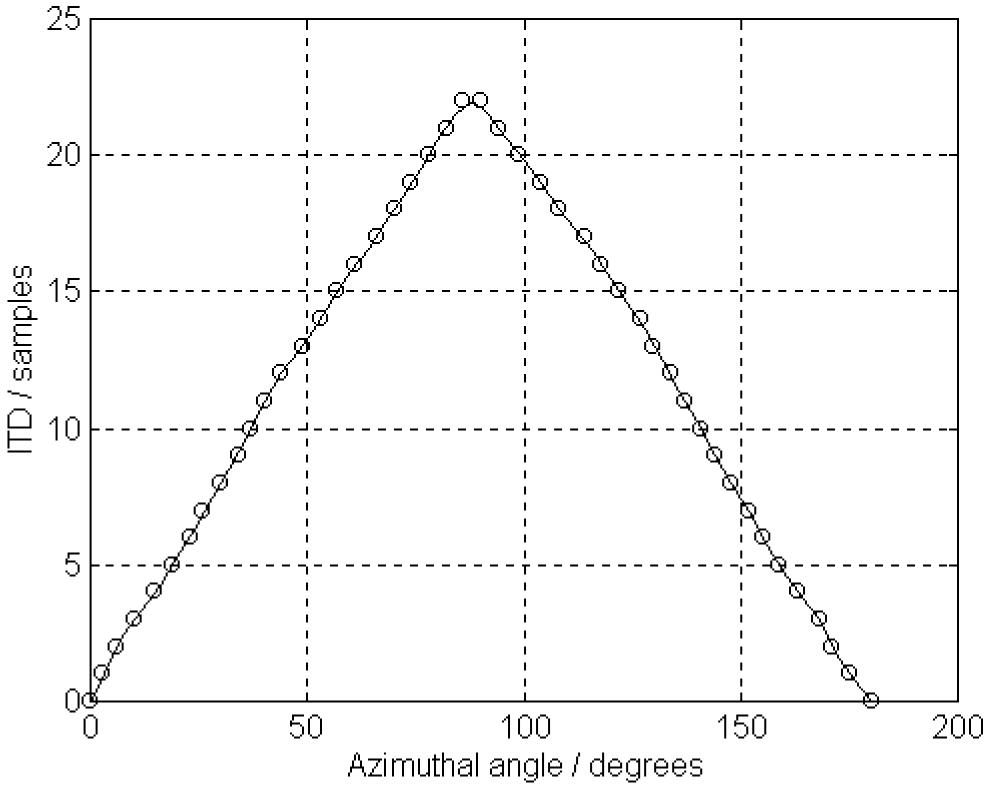


Kuva 3. Keinon HRTF interpoloiden karkeat mittaustulokset (katso teksti).

Tämä ei kuitenkaan antanut tyydyttävää tarkkuutta, joten mittasimme maksimi-kulmaa vastaavan askelien (22) määrää vastaavien kulmien välein (4.1 astetta) uuden taulukon. Käytimme pistemäistä kohinaherätettä (pink noise) kahden metrin etäisyydeltä ja etsimme päätä kääntämällä maksimeja ristikorrelaatio-taulukosta samplen välein. Kun maksimi oli löytynyt, määritimme käännetyn kulman suuruuden ja laskimme poikkeamaa (nollasta) vastaavien askelien määrän. Seuraavassa taulukossa mittaustulokset:

Kulma (astetta)	Ristikorrelaation maksimin paikka
90	22
86	21
81	20
76	19
72	18
66	17
62	16
58	15
53	14
50	13
46	12
43	11
39	10
36	9
32	8
Kulma	Maksimin paikka
28	7
25	6
21	5
17	4
12	3
9	2
5	1
0	0
3	-1
6	-2
10	-3
15	-4
19	-5
23	-6
26	-7
30	-8
Kulma	Maksimin paikka
34	-9
37	-10
40	-11
44	-12
49	-13
53	-14
57	-15
61	-16
66	-17
70	-18
74	-19
78	-20
82	-21
86	-22
90	-23

Nämä 'kääntäen' määritetyt ITD:t osoittautuivat erittäin tarkoiksi. Mittauksen ansiosta laitteiston tarkkuuteen vaikuttavista virhelähteistä poistui HRTF:n epätarkkuus verrattuna liian pienen näytetaajuuden aiheuttamaan virheeseen. Lopputulos näyttää graafisesti seuraavalta:



Kuva 4. Keinon HRTF mitattuna kääntäen (katso teksti).

3.4 Muuta

3.4.1 PLL:n kellotaajuuden nostaminen

PLL:n (Phase Locked Loop) kellotaajuutta pystyttiin helposti kasvattamaan oletusarvoisesta 40 MHz:ista 64 Mhz:iin. Sopiva käsky löytyi CODEC.ASM-tiedostosta:

```
movep #$261009,x:PLL
```

Tämä käsky asettaa taajuudeksi 10*4 MHz, eli 40 MHz. Vaihtamalla viimeisen $09 $0F:ksi saadaan kellotaajuudeksi 16*4 MHz. Kerroin on siis tuo heksaluku+1.

DSP56002 toimii 4.00 Mhz:n kiteellä. DSP56002:n PLL:ää käytetään kertomaan tämä taajuus prosessorin toimintataajuudeksi. Esim. 40MHz DSP56002:lla maksimi suositeltu kerroin on 10. Käytimme 66 Mhz:n prosessoria, joten PLL:n kertoimen nostaminen ei tuottanut minkäänlaisia ongelmia.

3.5 Ohjelmaversiot

3.5.1 Keino 1.0

Ensimmäinen versio Keino Suuntosen ohjausohjelmasta oli yksinkertaisesti sellainen, että luettiin vasemmasta ja oikeasta kanavasta X- ja Y-muisteihin 2 ms ajalta näytteitä. Tämän jälkeen laskettiin näiden kahden näytejonon välinen ristikorrelaatio. Ristikorrelaatio laskettiin vielä sirkulaarisena (vrt. sirkulaarinen konvoluutio), eli näytejonoihin piti lisätä nollia (zero pad) N-1 kpl kumpaankin. Ristikorrelaation laskemiseen meni paljon aikaa, joten sisääntuleva signaali katkeili aina 2 ms välein. Lopuksi etsittiin ristikorrelaation maksimi hyvin suoraviivaisella algoritmilla, joka sekin lisäsi sisääntulevan näytejono katkeaman pituutta.

3.5.2 Keino 2.0, 2.1 ja 2.2

Seuraavassa versiossa parannuksia tehtiin kovasti: Alettiin käyttää kahta puskuriparia, joista toiseen luetaan näytteitä samalla kun toisesta lasketaan ristikorrelaatiota. Ristikorrelaation laskeminen täytyi näin hajauttaa näytteiden luvun väliin. Koska ristikorrelaation arvoja on lopulta 2*N-1 kpl, jokaisen näytteen välissä lasketaan kaksi ristikorrelaation arvoa.

Huima algoritmin tehokkuusparannus oli myös olla käyttämättä ristikorrelaation sirkulaarista versiota ja nollia signaalien täytteenä, koska tällöin kolme neljäsosaa kertolaskuista oli nollalla kertomisia. Pointtereita sopivasti siirtelemällä ristikorrelaation lasku saatiin toteutettua ilman sirkulaarisuutta.

Maksimin etsiminen integroitiin myös ristikorrelaation yksittäisen pistetulon laskemisen yhteyteen.

Tähän versioon lisättiin myös askelmoottorin ohjauspulssien tuottamisalgoritmi. Askelmoottorin askeleet oli tallennettu taulukkoon, joka oli saman mittainen kuin ristikorrelaatiotaulukko. Maksimin kohdalta otettiin oikea askeleiden lukumäärä ja käännettiin askelmoottoria tämän luvun verran. Tasonsäätö- ja alukkeenkorostusalgoritmi implementoitiin myös, mutta sitä ei ehditty saada toimimaan edes lopulliseen versioon.

PLL:n kellotaajuus nostettiin 40 MHz:ista 64 MHz:iin.

3.5.3 Keino 2.3 ja 2.4

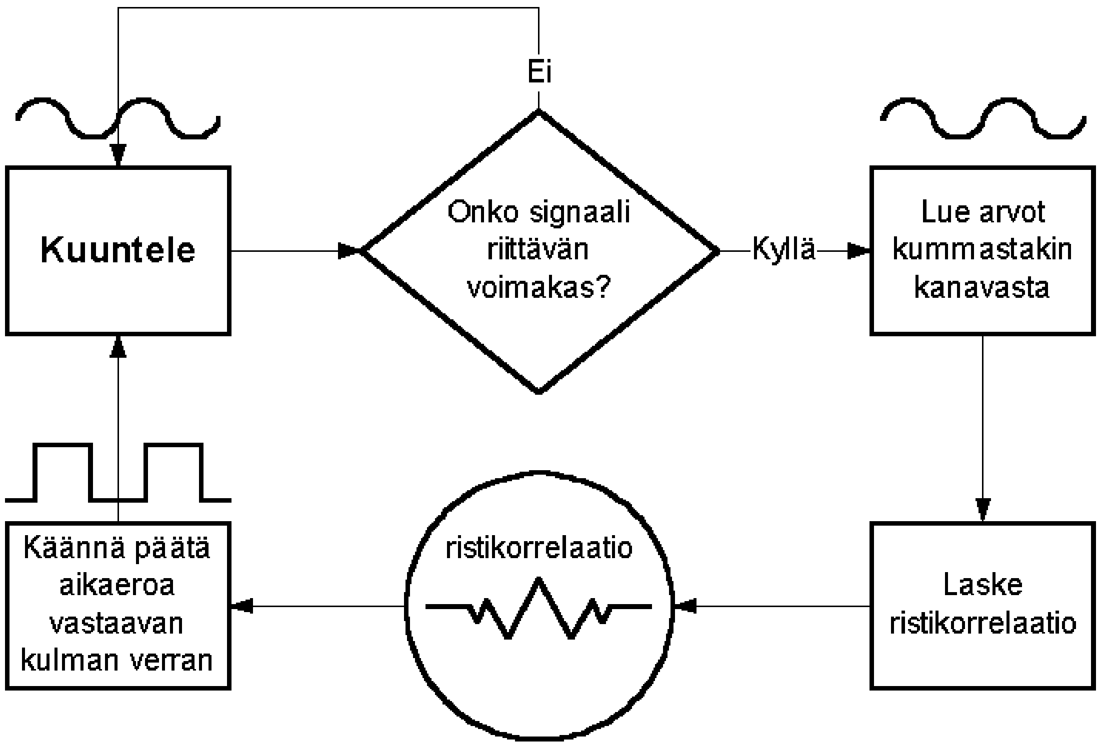
Edellä mainituista parannuksista huolimatta täytyi vielä korjailla monia pieniä bugeja ennenkuin ohjelma varsinaisesti toimi versiossa 2.4.

3.5.4 Keino 3.0

Ongelmaksi edellisissä versioissa, joissa jo päätä oikeasti käänneltiin, muodostui askelmoottorin ääni, joka vaikutti sisääntuleviin signaaleihin radikaalisti. Samoin ongelmia tuotti ristikorrelaation laskenta pään käännön aikana, mikä aiheutti maksimin siirtymisen väärään paikkaan. Näiden ongelmien poistamiseksi tyydyttiin ratkaisuun, jossa ääntä 'kuunnellaan' useamman ristikorrelaation verran, lasketaan näiden painotettu keskiarvo ja tehdään päätös äänen tulosuunnasta. Tämän jälkeen 'suljetaan korvat' ja käännetään päätä haluttuun suuntaan, jossa taas 'kuunnellaan'. Tässä versiossa askeleiden lukumäärätaulukko tehtiin käytetyn pään HRTF-mittausten perusteella.

3.5.5 Keino 3.1

Koodia siistittiin ja turhia hyppyjä ja looppeja poisteltiin. Versio oli ensimmäinen kaikenkaikkiaan käyttökelpoinen. Kuunteluvaiheeseen lisättiin pieni gate, joka ei päästä ohjelmaa eteenpäin mikäli tuleva signaali on liian heikko. Tämä rauhoitti pään liikkeitä. Seuraavassa kuvassa tämän virallisella versionumerolla 1.0. toimitetun ohjelman algoritmin vuokaavio.



Kuva 5. Keino 1.0. Yksinkertainen kaavio algoritmista

4. Keinon hardware

4.1 DSP EVM56 EVM

Motorolan on tehnyt edullisen kokeiluympäristön DSP56002EVM evaluaatio-modulin. Piirilevyllä on 24-bitin tarkkuudella laskeva prosessori DSP56002, 32 kilosanaa ulkoista SRAM ja stereo audiokoodekki (Crystal Semiconductor's CS4215).

Käytimme kortin kuulokeliitäntää ulostulona, sillä tarvitsimme tasajännitetasoja ohjaamaan askelmoottoriohjainta ja vain kuulokeliitäntä oli DC-kytketty.

4.2 Flash EEPROM

Evaluaatio- kortille sai optiona flash EEPROM muistipiirin. Käytimme valmistajan suosittelemaa piiriä malliltaan Atmel AT29C256-25PC (250ns). Se on 256 kilobitin 5 Voltin read and write PEROM flash.

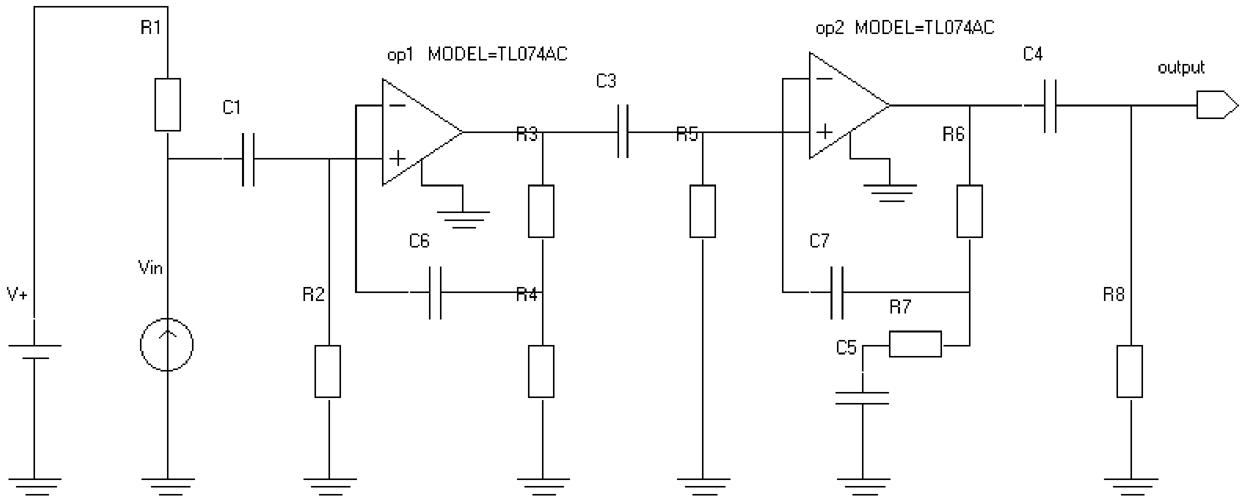
4.3 Mikrofonit

4.3.1 Mikrofonikapselit

Käytimme Sennheiserin valmistamia pieniä (5 mm halkaisija) kondensaattorimikrofoni-kapseleita. Ne sopivat erinomaisesti käyttötarkoitukseen. Tuimme ne E-A-R -merkkisten korvatulppien avulla muovipään 'korviin'. Korvatulpat eristivät ne näin suoralta mekaaniselta kosketukselta kovaan muovipäähän, josta olisi saattanut tulla haitallisia, häiritseviä värähtelyjä.

4.3.2 Mikrofoniesivahvistin

Mikrofonit vaativat polarisointijännitteen toimiakseen ja tason sovituksen EVM-kortin sisääntuloon. Seuraava kytkentäkaavio toteutti tämän tehtävän.



Kuva 6. Mikrofoniesivahvistimen yhden kanavan kytkentäkaavio

4.4 Mekaniikka

4.4.1 Muovipää

Tiedekeskus Heureka toimitti laboratorioon kovan, mutta kevyen mallinuken pään. Pään korvien etäisyys oli 14.0 cm. Pää istutettiin pleksi-istukkaan, joka laakeroitiin tukevasti alumiinista sorvattuun laakeripesään.

4.4.2 Pyöritysmekanismi

4.4.2.1 Moottori

Moottorin kytkentä on seuraava:

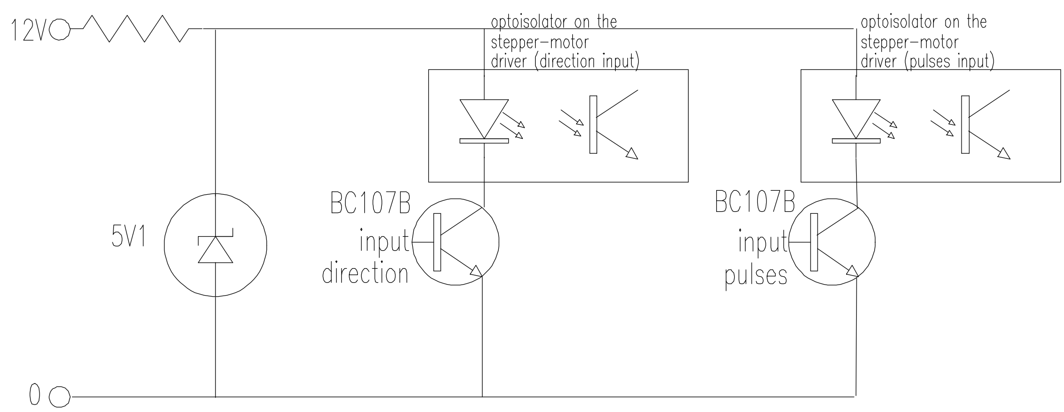
green	18a
black/white	18c
orange	22a
green/white	22c
red	24a
white	24c
black	28a
red/white	28c

Moottori on vahva 25 W nelikääminen askelmoottori. Sen käämit kytkettiin pareittain rinnan.

4.4.2.2 Ohjain

Valmis moduli. Pystyy ohjaamaan kaksi-käämisiä askelmoottoreita maksimivirralla 2A. Sovelluksessamme käytetty kytkentä (katso 4.4.2.1 moottorin kytkentä):

10a ja 14a on kytketty 5V1:n jännitteeseen, joka saadaan ohjaimen 12V:n käyttöjännitteestä etuvastuksen ja zenerdiodin avulla.
Koska koodekki kykenee antamaan maksimissaan $2V_{pp}$ jännitteen, jouduimme käyttämään puskuritransistoreja vahvistamaan ulostulosignaalia, jotta se riittäisi moottoriohjaimen sisääntulon optoeristimille.

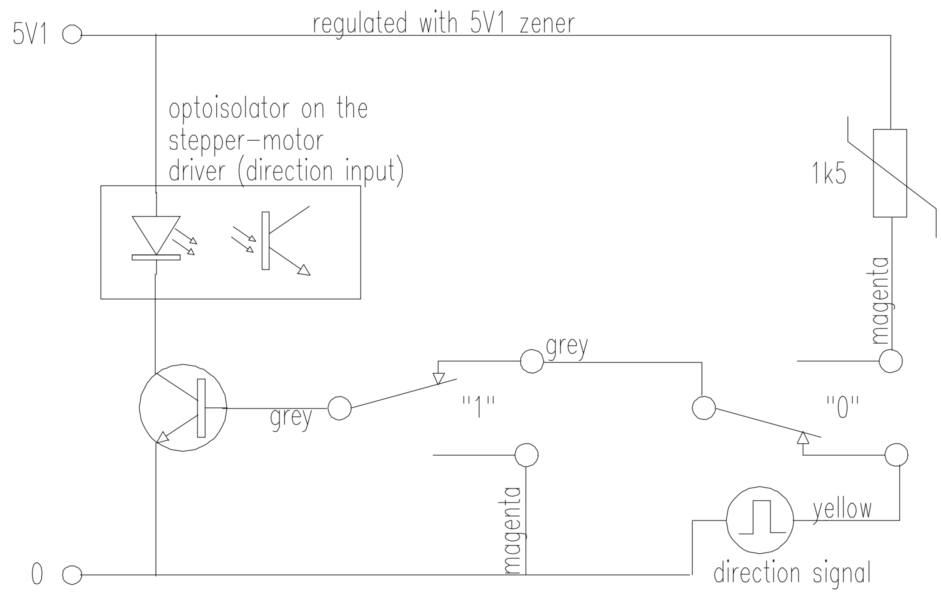


Kuva 7. Moottoriohjaimen puskuripiiri

4.4.2.3 Liikkeenrajoittimet sekä mikrokytkimet

Koska prosessorille ei tule tietoa absoluuttisesta asennosta, täytyi pään liikettä rajoittaa fyysisesti jotta mikrofoneilta tulevat johdot eivät katkeaisi pään kääntyessä akselinsa ympäri. Tämä onnistui akselin läpi laitetulla sauvalla ja kahdella rajoitintapilla.

Käytetty askelmoottori oli niin jäykkä (vahva), ettei se antanut lainkaan periksi vaan olisi vääntänyt tapit kumoon. Tästä syystä jouduimme laittamaan rajoitintappeihin kiinni mikrokytkimet (Cherry Elect. Prod.Corp. E22 3amp 125-250 VAC), jotka johdotettiin kuvan 5 mukaisesti. Kun akselin mukaan liikkuva sauva tulee rajoitintappiin, vaihtaa kytkin asentoaan ja kytkennän mukaisesti kääntää ohjaimelle suunnan ilmoittavan loogisen ykkösen nollaksi ja päinvastoin. Mikäli ääni tulee jatkuvasti pään rajoitetun liikealueen ulkopuolelta (laitteen takaa), heiluu pää kytkimen asennon vaihdosta aiheutuvaa liikerataa - siis lyhyttä liikettä edestakaisin rajoitintapin tuntumassa.



Kuva 7. Liikkeenrajoittimen kytkentäkaavio

4.4.3 Kotelointi

Vanha mittalaite kotelo kalustettiin Keino Suuntosen yläkehoksi. Sinne mahtuivat moottori ohjaimineen, DSP56002EVM-kortti sekä mikrofoniesivahvistin.

Virtalähteet ovat kotelon ulkopuolella. Niitä tarvittiin 12 V 2 A, erityisesti Keinon mikrofoniesivahvistinta varten suunniteltu 2x9V 50mA lähde sekä 9V 1A virtalähde.

5. Ongelmia

5.1 Muistialueiden käyttö

Ongelmia aiheutti alussa muistialueiden käyttö, joka ei toiminut niinkuin olisi olettanut. Ohjelma sotkeentui, jos ohjelma alkoi osoitteesta P:$200. Tämä korjattiin aloittamalla ohjelma osoitteesta P:$40.

Vaikeuksia oli myös ymmärtää EVM-kortin mukana tullut CODEC.ASM -tiedoston toiminta, vaikka harjoitustyö oli ajateltu tehtävän niin, ettei sitä tarvitsisi ymmärtää. CODEC.ASM on EVM-kortin codecin alustustiedosto. Keino Suuntosen ohjausohjelman laajuudesta ja luonteesta johtuen jouduimme tutustumaan ko. tiedostoon perinpohjin. Lähes kaikista Motorolan prosessoriin ja EVM-korttiin liittyvistä epäselvistä asioista jouduimme itse ottamaan selvää. EVM-kortti oli myös huonosti dokumentoitu toimintansa kannalta.

5.2 Pulssien muodostaminen

Koska moottoriohjain otti sisään TTL-tasoisia signaaleja ja kortin digitaalilähtöihin ei ollut dokumentointia saatavilla, jouduimme ohjaamaan koodekilla ohjainta.

5.3 Stand-alone toteutus

Keino Suuntosen oli määrä päätyä tiedekeskukseen ja näin ollen systeemi täytyi saada mahdollisimman ‘helppohoitoiseksi’. Prosessorikortti vaati PC:n rinnalleen, jotta ohjelma olisi saatu ladattua sen muistiin. Tämä ei luonnollisesti tullut näyttelykäytössä kyseenkään ja kortti täytyi saada itse lataavaksi.

Päädyimme FLASH-muistiin helpon päivitettävyyden vuoksi, mutta siihen ei taas löytynyt dokumentteja kovinkaan helposti. Lopulta saimme mutkien takaa apua Yhdysvalloissa Oregonin yliopistossa opiskelevalta *Johen Forrerilta*. Hänen ystävänsä oli tehnyt sopivan ohjelman tähän tarkoitukseen. Tosin paketin mukana tullut ‘readme’ -tiedosto ei aivan heti avannut koodin alustusten tarkoituksia. Onneksi siihen saatiin taas apua naapurilaboratorion *Jarkko Vuorelta*.

Kortin muistialueet kuvautuivat seuraavan taulukon mukaan flash-muistiin:

Muistin alkuosoite (hex)	Kuvautuva alue
E000	EXT P
D200	EXT Y
CF00	INT Y
CC00	INT X
C600	INT P
C000	Internal use
8000	EXT X

Taulukko 2. Flash EEPROM muistialueiden sisältö

5.4 DSP-kortin huono suojaus

Huomasimme pian, että DSP56002EVM on todella herkkä sähkönpurkauksille. Kortilla todennäköisesti tulee pariteettivirheitä, joista se ei toivu ja menee jumiin, mikäli lähistöllä on staattisia purkauksia ilmaan aiheuttavia sähkölaitteita. Esimerkiksi loisteputken välkyntä saattaa kaataa kortilla pyörivän ohjelman.

6. Parannusehdotuksia

Keino Suuntosen suuntakuulo on tällä hetkellä varsin yksinkertainen: Kun sisääntuleva signaali on kyllin voimakas, luetaan näytteitä sisään, lasketaan 32 ristikorrelaation painotettu keskiarvo ja käännetään päätä maksimin suuntaan. Yksinkertaisuuden vuoksi pään käännön ajaksi suljetaan vielä sisääntulot.

Parannuksina ohjelmaan voisi tehdä seuraavaa:
Sisääntulevan signaalin tason vertailun voisi tehdä adaptiiviseksi, jolloin keskimääräisessä metelissä Keino vain pyörittelisi päätä tai tekisi jotain muuta sijaistoimintoa ennenkuin tulee jokin metelistä selvästi erottuva äänes.

Alukkeen tunnistus voisi parantaa Keinon tarkkuutta ja nopeutta äänen tulosuunnan määrityksessä.

Koska yli 1.5 kHz:in signaalit alkavat heiketä tulosuunnan vastakkaisessa korvassa ja ihmisaivotkin tekevät korkeille taajuuksille lähinnä ILD-analyysiä (Interaural Level Difference), voisi Keinon seuraavaan versioon lisätä tämän ominaisuuden ITD-analyysin lisäksi.

Näytetaajuuden kasvattaminen toisi lisää tarkkuutta kulman tunnistamiseen.

DSP56002EVM- kortti pitäisi suojata ilman kautta eteneviltä staattisilta sähkönpurkauksilta toimintahäiriöiden välttämiseksi (vrt. edellä).

7. Sovelluksia ja pohdintaa

Keino Suuntosessa käytetylle tekniikalle on helppo keksiä monia eri käyttötarkoituksia. Yksi voisi olla videokameran ohjaus. Kamera voisi olla valvontatarkoituksessa tai kuvaamassa tärkeissä kokouksissa kulloistakin puhujaa.

Lisäämällä kolmannen mikrofonin saisimme helposti etu- takaerottelun aikaiseksi ja näin täydet 360 astetta käyttöön. Lisäksi lisäämällä toisen mikrofoniparin olisi myös elevaatioerottelu mahdollista.

Näytteenottotaajuutta nostamalla saisimme erottelytarkkuudenkin erittäin tarkaksi. Tällä hetkellä Keino Suuntonen ottaa 32000 näytettä sekunnissa ja tarkkuus on teoriassa ja hyvissä olosuhteissa käytännössä 4 astetta.

8. Lähteet

Backman, J., Huopaniemi, J. ja Rahkila, M., Akustiikan seminaari 1995. Tilakuuleminen ja auralisaatio. Raportti 36, Teknillinen Korkeakoulu, Akustiikan ja äänenkäsittelyn laboratorio, s. 7-175.

Backman, J. ja Karjalainen, M., Modelling of Human Directional and Spatial Hearing Using Neural Networks, ICASSP-93, s. I-125 - I-127

Karjalainen, M. 1996. Kommunikaatioakustiikka / opetusmoniste K-96 erä 3 Teknillinen korkeakoulu, Akustiikan ja äänenkäsittelyn laboratorio.

Matlab. 1993. Signal Processing Toolbox User's Guide. Mathworks, Inc, s. 2-174

Yost, W.A., Perceptual Models for Auditory Localization, AES 12th International Conference, s.155-168

9. Liitteet

LIITE 1. Ohjelma-algoritmi

LIITE 2. Ohjelman lähdekoodi kommentoituna

LIITE 3. FLASH EEPROM- ohjelmoinnissa käytetty koodi