OULUN YLIOPISTO
UNIVERSITY of OULU

KANDIDAATINTYÖ

Oskari Kampman
Virpi Karhula
Markus Sonntag

SÄHKÖ- JA TIETOTEKNIIKAN OSASTO

2009

OULUN YLIOPISTO
UNIVERSITY of OULU

KANDIDAATINTYÖ

Puhekomentojärjestelmä

Oskari Kampman
Virpi Karhula
Markus Sonntag

Ohjaajat: Juha Röning, Teemu Tokola

SÄHKÖ- JA TIETOTEKNIIKAN OSASTO

2009

Kampman O., Karhula V., Sonntag M. (2009) Puhekomentojärjestelmä. Oulun yliopisto, sähkö- ja tietotekniikan osasto. Kandidaatintyö, 78 s.

TIIVISTELMÄ

Kandidaatintyössä toteutettiin sulautettujen ohjelmistojen järjestelmä, jonka toimintaa ohjattiin puheentunnistuksen avulla.

Työn suunnittelun ja puheentunnistusratkaisun pohjana käytettiin kirjallisuuskatsausta ja esiselvitystä puheentunnistuksen alalta, jonka avulla selvitettiin alan tämän hetkinen tilanne ja perehdyttiin puheentunnistuksessa käytettyihin tekniikoihin ja metodeihin.

Sulautetun järjestelmän toimintaympäristöksi toteutettiin laivanupotuspeli, jota pelattiin verkon yli muiden vastaavien järjestelmien kanssa. Verkkotoiminnallisuus toteutettiin käyttäen SNMP-protokollaa.

Puheentunnistusjärjestelmä opetettiin tunnistamaan yksittäisiä sanoja. Puheentunnistus toteutettiin käyttäen Mel Frequency Cepstral Coefficients–metodia (MFCC) ja Dynamic Time Warping –hahmontunnistusmenetelmää (DTW).

Sovellusalustana oli Elektor Internet Radio 1.0 (EIR), johon kuuluu Atmel AT91SAM7SE512 RISC mikrokontrolleri, VLSI:n VS1053 audiodekooderi ja Davicom DM9000E ethernetkontrolleri. Mikrokontrollerilla on 512 kilotavua nopeaa Flash-muistia. Ohjelmakoodin ja datan käytössä on 64 MT RAM muistia.

Puheentunnistustoteutuksen onnistumista arvioitiin soveltamalla sitä järjestelmän päävalikon valintojen tekemiseen. Järjestelmä pystyi tunnistamaan yhden käyttäjän lausumana sanat ”yksinpeli”, ”verkkopeli”, ”asetukset” ja ”tulostus”. Suoritettujen testien perusteella saavutettiin 88 %:n tunnistustarkkuus.

Avainsanat: puheentunnistus, sulautettu järjestelmä, MFCC, DTW.

Kampman O, Karhula V A, Sonntag M M (2009) Isolated word speech recognizer. University of Oulu, Department of Electrical and Information Engineering. Candidate's Thesis, 78 p.

ABSTRACT

Subject of this Candidate's thesis was to implement software that could be used via speech detection for an embedded system.

Basis of this work was a extensive state of the art review on the field of speech recognition. During early days of this project we got a wide knowledge about today's speech recognition techniques and methods .

Operational environment for the speech detection software was a battleship-game that could be played over a network with other similar systems. The network was implemented using SNMP-protocol.

Speech recognition system was trained to recognize isolated words. Speech recognition was implemented using Mel Frequency Cepstral Coefficients method (MFCC) and Dynamic Time Warping (DTW) pattern recognition technique.

The project was implemented on Elektor Internet Radio 1.0 (EIR) embedded system, which included Atmel AT91SAM7SE512 RISC microcontroller, VS1053 audio decoder from VLSI and Davicom DM9000E ethernet controller. The microcontroller has 512 kilobytes of fast, internal flash memory. There is 64 megabytes of RAM available for software and data.

The speech recognition system was evaluated by using it to control the main menu of the program. The system was able to recognize the words "yksinpeli", "verkkopeli", "asetukset" and "tulostus" spoken by a single speaker. Based on the tests, the system was able to recognize words with accuracy of 88 percents.

Key words: speech recognition, embedded systems, MFCC, DTW.

SISÄLLYSLUETTELO

ALKULAUSE

Kandidaatintyön tehtävään kuului laivanupotuspelin, verkkotoiminnallisuuden ja puheentunnistuksen toteuttaminen sekä integrointi toimivaksi kokonaisuudeksi sulautetulle alustalle. Tehtävänanto oli laaja ja haasteellinen. Puheentunnistus oli uusi aihealue, joka vaati perehtymisen alan kirjallisuuteen sopivan puheentunnistusmenetelmän valitsemiseksi.

Lopulta laivanupotuspeli, verkkotoiminnallisuus ja yksittäisten sanojen tunnistukseen perustuva puheentunnistusjärjestelmä saatiin toimimaan sulautetussa järjestelmässä. Tämän laajan ja opettavaisen tehtävän onnistuneesta loppuunsaattamisesta kiitos kuuluu meille kaikille.

Oulussa 29.5.2009

Oskari Kampman, Virpi Karhula ja Markus Sonntag

LYHENTEIDEN JA MERKKIEN SELITYKSET

API	Application Programming Interface, ohjelmointirajapinta
ASR	Automatic Speech Recognition, automaattinen puheentunnistus
CPU	Central Processing Unit, keskusyksikkö
DCT	Discrete Cosine Transform, diskreetti kosinimuunnos
DFT	Discrete Fourier Transform, diskreetti fourier-muunnos
DTW	Dynamic Time Warping, dynaaminen ajansiirto
EIR	Elektor Internet Radio
FFT	Fast Fourier Transform, nopea fourier-muunnos
LPC	Linear Predictive Coding, lineaarinen ennustava koodaus
MFCC	Mel Frequency Cepstral Coefficients, Mel-taajuuskepstrikertoimet
RAM	Random Access Memory, hajasaantimuisti
SNMP	Simple Network Management Protocol
UART	Universal Asynchronous Receiver/Transmitter, universaali asynkroninen vastaanotin/lähetin

$s(n)$	Näytteistetty puhesignaali
$\widetilde{s}(n)$	Esivahvistettu puhesignaali
$x_l(n)$	Kehyksiin jaettu puhesignaali
$w(n)$	Hamming-ikkuna
$\widetilde{x}_l(n)$	Hamming-suodatettu puhesignaali
H(z)	Esisuodatuksen siirtofunktio

Hz	Hertsi
Mel	Melody
MT	Megatavu

1. JOHDANTO

Puheentunnistusta on tutkittu jo viime vuosituhannen alkupuolelta lähtien ja ensimmäiset toimivat toteutukset ovat yli 50 vuoden takaa.

Nykyään automaattisen puheentunnistuksen kehittyminen on tehnyt mahdolliseksi sen, että puheentunnistusta voidaan hyödyntää mm. käyttöliittymissä, sanelutilanteissa ja puhelinvaihteissa.

Puheentunnistusjärjestelmien kehittyminen on tuonut uuden ulottuvuuden laitteistojen käyttöliittymiin. Puheentunnistusta voidaan käyttää apuna tilanteissa, joissa kädet ovat varatut muihin tehtäviin ja puheella voidaan ohjata lisälaitteiden toimintaa esimerkiksi auton ohjaamossa.

Parhaat puheentunnistuksen tulokset saavutetaan, kun järjestelmää on opetettu tunnistamaan tietyn puhujan puhetta ja sanaston laajuus on rajoitettu tiettyä erikoisalaa koskevaksi. Automaattista puheentunnistusta käyttäen on mahdollista täydentää esim. sairaskertomusta suoraan lääkärin sanelusta.

Puheentunnistusjärjestelmän toiminta jakaantuu kahteen vaiheeseen: järjestelmän opettamiseen ja sanojen tunnistamiseen. Opetusvaiheessa esimerkkipuhetta analysoidaaan ja sanoista tunnistetaan piirteitä, joihin tunnistusvaiheen puhesyötettä verrataan. [1]

Tämän työn tavoitteena oli toteuttaa sulautetulle järjestelmälle puheentunnistusta hyödyntävä ratkaisu. Käytännön sovelluskohteeksi tehtiin laivanupotuspeli, jolla pystyi pelaamaan verkon yli muiden vastaavien toteutusten kanssa. Projektin haastavin osuus oli puheentunnistuksen toteuttaminen ja puheen käyttäminen vaihtoehtoisena syötteenä järjestelmän käyttöliittymässä. Tämän toteuttamiseksi kartoitettiin puheentunnistuksessa käytettävät menetelmät ja valittiin projektin tarpeisiin niistä sopivimmat.

2. TAUSTA

2.1. Puheentunnistuksen historiaa

Bellin laboratoriossa Fletcher kollegoineen perehtyi ihmisen tapaan tunnistaa puhetta vuosien 1918 ja 1950 välillä. Tutkimukset tähtäsivät puhelimessa puhutun puheen teknisen laadun parantamiseen [2].

Aikaisimmat yritykset koneella tehtyyn puheentunnistukseen tapahtuivat 1950-luvulla, kun useat tutkijat yrittivät hyödyntää akustisen fonetiikan periaatteita [3]. Vuonna 1952 Bell-laboratoriossa rakennettiin järjestelmä yhden puhujan yksittäisten sanojen tunnistamiseen. Järjestelmä perustui spektrin taajuuden mittaamiseen vokaalien ääntämisen aikana [3, 4].

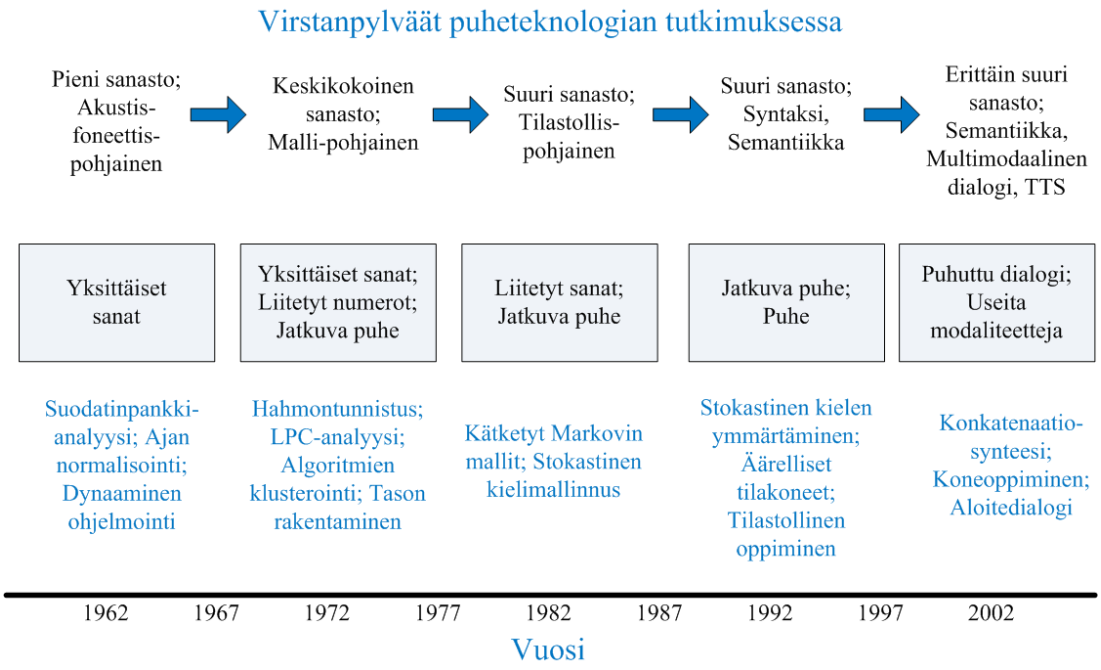
1960-luvulla monia puheentunnistuksen periaatteita keksittiin ja julkaistiin. Useat japanilaiset laboratoriot rakensivat erikoislaitteistoja vokaalien ja foneemien tunnistamiseksi [3, 4]. Tutkimuksessa keskityttiin ajallisesti epäyhtenäisen puheen aiheuttamien ongelmien ratkaisuihin ja jatkuvan puheen tunnistukseen käyttäen foneemien dynaamista tunnistamista [3].

1970-luvulla yksittäisten sanojen tunnistus kehittyi. Venäläisten tutkijoiden tutkimus auttoi hahmontunnistuksen käyttöönottoa puheentunnistuksessa. Japanilaiset tutkivat dynaamista ohjelmointimenetelmien hyödyntämistä [3, 4]. Itakuran tutkimus selvitti, kuinka lineaarisen ennustavan koodauksen (Linear Predictive Coding, LPC) taajuusparametrejä voitiin käyttää [3]. IBM aloitti 1970-luvulla laajan sanaston puheentunnistusjärjestelmän kehittämisen [3, 4]. AT&T Bell-laboratorio aloitti puhujasta riippumattoman puheentunnistusjärjestelmän kehittämisen. 1970-luvulla tutkimus keskittyi yksittäisten sanojen tunnistamiseen, kun taas 1980-luvulla painopiste siirtyi yhdistettyjen sanojen tutkimiseen. Tavoitteena oli rakentaa järjestelmä, joka kykenisi luotettavasti tunnistamaan puhevirrasta sanoja [3].

Puheentunnistus siirtyi 1980-luvulla malleihin perustuvista tekniikoista tilastollisiin mallintamismenetelmiin, kuten kätkettyihin Markovin malleihin. Kätkettyjä Markovin malleja sovellettiin 1980-luvun puolivälissä laajalti eri tutkimuslaboratorioissa ympäri maailman. Neuroverkot tulivat 1980-luvun loppupuolella puheentunnistuksen ongelmanratkaisun käyttöön. [3]

Tällä hetkellä suurin osa käytössä olevista kehitysympäristöistä perustuu 1980-luvulla kehitetyille tekniikoille, joita paranneltiin 1990-luvulla [4].

Kuvasta 1 [5] käy ilmi, kuinka puheentunnistuksen tutkimus on kehittynyt neljän vuosikymmenen aikana. Tutkimuksen painopiste on siirtynyt yksittäisten sanojen tunnistuksesta jatkuvan puheen tulkintaan. Samalla sanaston laajuus on kasvanut voimakkaasti.



Kuva 1. Puheentunnistuksen ja -ymmärryksen merkittävät edistysaskeleet viimeisen 40 vuoden ajalta.

2.2. Käytössä olevat puheentunnistusjärjestelmät

Puheentunnistuksen sovelluskohteet ovat muuttuneet yksittäisiä sanoja tunnistavista järjestelmistä jatkuvaa puhetta tunnistamaan kykeneviksi laajojen sanavarastojen järjestelmiksi.

Englantia äidinkielenään puhuvien henkilöiden puheen tulkinnassa päästään tarkkaan puheentunnistukseen. Jos kieli ei ole englanti ja järjestelmä on vähemmän opetettu, voidaan saavuttaa 80 % tarkkuus. [6]

Puhelinnumeroiden [7] tai osoitteiden palvelupyynnöt suurista tietokannoista voidaan hoitaa automaattisen assistentin avulla edullisemmin kuin henkilökohtaisesti palvellen. Automaattiseen puheentunnistukseen perustuvia tietokantahakuja on käytössä myös musiikin tai videoiden hallinnassa, liiketoiminta ja tuotetietojen yhteenvedoissa, hintatiedusteluissa ja konferenssien tiedotusjärjestemissä. [8]

Eräs ryhmämme jäsen muisti työtehtävää hoitaessaan soittaneensa muun muassa tulostimia valmistavan Xeroxin asiakastukeen ja huomanneensa, että asiakastuki käyttää puhesynteesiä ja puheentunnistusta puhelujen ohjaamisessa. Soittajaa pyydetään kertomaan muun muassa laitteen kirjaimia ja numeroita sisältävä tuotekoodi ennen puhelun yhdistämistä.

Kaupallisesti on saatavilla useita puheentunnistusjärjestelmiä.

IBM tarjoaa asiakkaille WebSphere Voice Server -palvelua. Palvelu sisältää puheentunnistusjärjestelmän, joka pystyy tunnistamaan puhelimessa puhuttuja sanoja ja muuntamaan ne kirjoitetuksi tekstiksi, sekä myös tekstistä puhetta syntetisoivan sovelluksen puhelinlinjan yli. Palveluun kuuluu AIX pohjainen IBM WebSphere Voice Server v4.2 sekä mahdollisuus kehitystyökalujen käyttöön [9].

Nuance tarjoaa Vocon sulautettua puheentunnistusjärjestelmää käytettäväksi mm. autoissa ja peleissä. Tuote soveltuu puhelun käynnistämiseen, navigointipäätepisteen valintaan ja komentojen antamiseen [10].

Microsoftin visiona on soveltaa puheentunnistusta erilaisille laitealustoille mukaan lukien tietokone, TabletPC, matkapuhelin, PDA tai tavallinen lankapuhelin. Microsoft Speech Server tarjoaa mahdollisuuden korvata osan hiirellä tai näppäimistöllä annetuista komennoista puheen avulla Windows Vista–käyttöjärjestelmässä [11].

2.3. Yleistä puheentunnistuksesta

Automaattisen puheentunnistuksen (Automatic Speech Recognition, ASR) pyrkimyksenä on tarkasti ja tehokkaasti muuntaa puhesignaali tekstiksi puhujasta tai ympäristöstä riippumatta [3]. Rajallisten resurssien vuoksi tätä tavoitetta ei ole kuitenkaan vielä tähän päivään mennessä saavutettu, vaikka puheentunnistusta on tutkittu 1940-luvun lopulta alkaen ja tietokoneetkin ovat tänä aikana kehittyneet suuresti. Parhaimmat puhekomentojärjestelmät pääsevät nykypäivänä kuitenkin jo lähelle sadan prosenttiyksikön tarkkuutta ideaalisissa olosuhteissa.

Puhetunnistusjärjestelmien suorituskykyä mitataan yleisimmin sanavirhetodennäköisyydellä [12]. Se perustuu ASR-järjestelmän palautteeseen suhteessa puhuttuihin sanoihin ja saadaan kaavalla (1).

$$E=100*(S+I+D)/N \quad (1)$$

missä S on järjestelmän korvaamien sanojen lukumäärä, I on lisättyjen sanojen lukumäärä, D on poistettujen sanojen lukumäärä ja N on testijoukon sanojen kokonaismäärä.

Puheentunnistusjärjestelmän rakenne monimutkaistuu, jos järjestelmän tulee tunnistaa useiden puhujien puhetta useista eri maanosista tai murrealueilta. Samoin jatkuvan puheen tunnistaminen on suurempi haaste kuin yksittäisten sanojen tunnistaminen [13].

Puheentunnistusta on mahdollista käyttää henkilön tunnistamiseen [14] tai hänen puhumansa kielen tunnistamiseen [15].

Jokaisella ihmisellä on oma persoonallinen puhetapansa. Puheen rytmitys ja painotus lauseessa ja sanan sisällä vaihtelevat puhujan mukaan. Puhujan sanasto, murre ja koulutus vaikuttavat tapaan esittää asiansa [4]. Sukupuoli vaikuttaa äänen keskikorkeuteen, mikä havaitaan taajuusalueella.

Mikrofonilaitteiston ominaispiirteet ja ympäristön meluisuus vaikuttavat selkeästi puheentunnistusjärjestelmän suorituskykyyn. Yleisesti mikrofonin kautta tallennettua puhetta on helpompi analysoida kuin puhelimen kuulokejärjestelmän (engl. headset) kautta talletettua puhesignaalia, sillä mikrofonin puhesignaalin laajempi kaistanleveys tarjoaa monipuolisemman akustisten piirteiden valikoiman puheentunnistusjärjestelmän käyttöön [13].

Seuraavassa on lueteltu puheentunnistusjärjestelmiä erottavia ominaisuuksia [16, 17]:

- Sanaston koko
- Puheen syöttötapa

- Käyttäjäriippuvuus tai –riippumattomuus
- Kieliriippuvuus tai monikielisyys
- Käyttöympäristö.

Puheentunnistusjärjestelmissä sanaston koko vaihtelee käyttötarkoituksen mukaan. Pieni sanasto tarkoittaa muutamia kymmeniä ja keskisuuri sanasto muutamia satoja tai tuhansia tunnistettavia sanoja. Suuri sanasto puolestaan voi kattaa jopa satatuhatta tunnistettavaa sanaa. Pienissä sanastoissa käytetään tyypillisesti kokonaisia sanoja tunnistusyksikkönä. Suuremmissa sanastoissa on käytettävä pienempiä tunnistusyksikköjä, kuten tavuja tai foneemeja.

Järjestelmän käyttötarkoitus vaikuttaa myös käytettävään puheen syöttötapaan. Komentojärjestelmässä käyttäjä sanoo yhden sanan kerrallaan. Sanelujärjestelmässä käyttäjä voi sanoa useita sanoja kerrallaan pitämällä lyhyen tauon kunkin sanan jälkeen. Jatkuvassa tunnistuksessa käyttäjä voi puhua normaalisti. Sanojen rajat saattavat kuitenkin olla vaikeasti tunnistettavissa, koska jotkin sanat artikuloidaan yhteen. Toisin kuin muissa menetelmissä, jatkuvassa tunnistuksessa tarvitaan myös kielimallia, joka sisältää kielioppiin ja merkityssisältöön liittyviä rajoitteita. Taulukossa 1 on vertailtu puhekomento-, sanelu- sekä jatkuvan puheen järjestelmien etuja ja haittoja.

Taulukko 1. Eri tyyppisten ASR-järjestelmien vertailu

	Puhekomento-järjestelmät	**Sanelujärjestelmät**	
		Sana kerrallaan	**Jatkuva puhe**
Puheen syöttö	Käyttäjät lausuvat yksittäisen sanan tai sanaryhmän komennon antaessaan.	Käyttäjien tulee lausua sanat hitaasti ja huolellisesti.	Käyttäjät lausuvat sanat normaalin puheen tavoin.
Edut	Erittäin tarkka	Soveltuu sanelunomaiseen käyttöön. Suuri sanavarasto Kohtuulliset prosessointivaatimukset	Soveltuu sanelumaiseen käyttöön. Suuri sanavarasto Helppo ja luonnollinen käyttää
Haitat	Rajalliset soveltamiskohteet Ei sovi sanelunomaiseen käyttöön Pieni sanavarasto	Rajallinen tarkkuus Hidas Hankalakäyttöinen Luonnoton puhetapa	Rajallinen tarkkuus Erittäin suuret prosessointi-vaatimukset

Käyttäjäriippumaton järjestelmä on luonnollisesti monikäyttöisempi, mutta myös vaikeampi toteuttaa. Käyttäjäriippuvaisella järjestelmällä saavutetaan parempi suorituskyky, koska yksittäisen käyttäjän puhe on akustiselta vaihtelultaan suppeampaa kuin usean eri käyttäjän puhe.

Puheentunnistusjärjestelmät ovat joko kieliriippuvaisia tai -riippumattomia. Tyypillisesti järjestelmät suunnitellaan kuitenkin vain yhdelle kielelle kielien erilaisuuksista (ääntäminen, sanapaino, lauserakenne) johtuen.

Puheentunnistusjärjestelmiä käytetään hyvin monenlaisissa ympäristöissä ja tämä vaikuttaa järjestelmän suunnitteluun ja toteutukseen. Monet ASR-järjestelmät toimivat hyvin hiljaisissa ja vakaissa laboratorioissa, mutta suorituskyky heikkenee vaikeammissa ympäristöissä. Tämä johtuu puhesignaalin korruptoitumisesta taustahälyn tai mikrofonin aiheuttaman särön seurauksena. Yleisesti ottaen ASR:t toimivat hyvin samoissa ympäristöissä, joihin ne on opetettu.

2.4. Puheentunnistusmenetelmiä

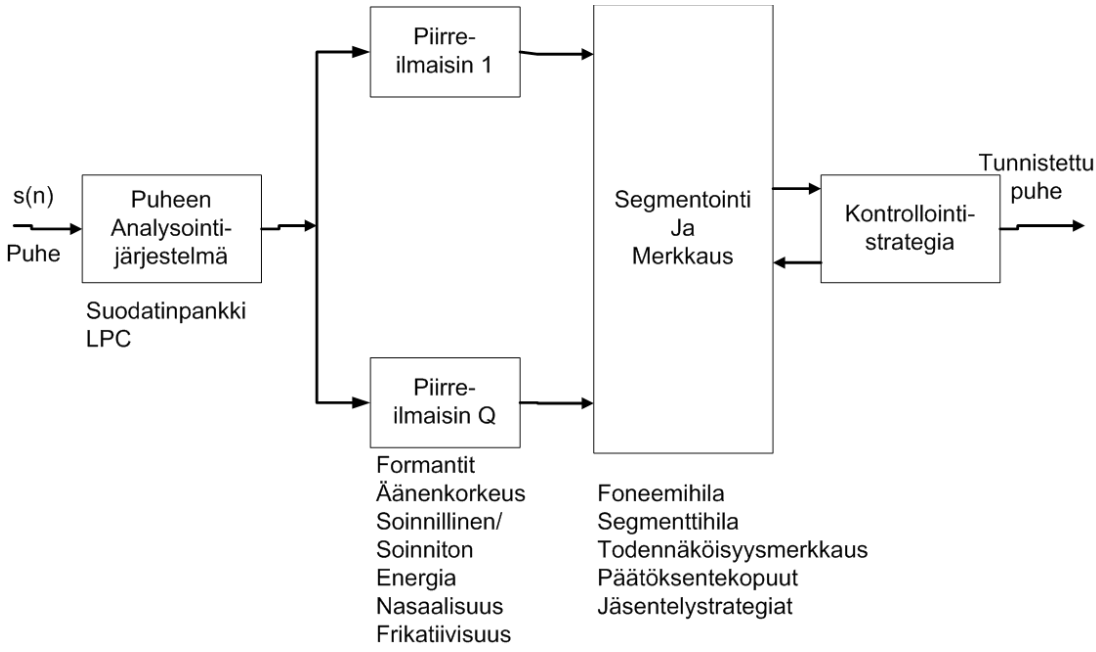
Puheentunnistusmenetelmät voidaan jakaa karkeasti kolmeen eri lähestymistapaan [3]:

1) akustis-foneettinen
2) tekoäly
3) hahmontunnistus

2.4.1. Akustis-foneettinen lähestysmistapa

Akustis-foneettinen lähestymistapa perustuu akustis-foneettiseen teoriaan, joka olettaa puhutussa kielessä olevan äärellisen määrän erotettavissa olevia foneemeja, jotka ovat nähtävissä puhesignaalissa tai sen spektrissä [3]. Tämä lähestymistapa on kaksivaiheinen. Ensiksi puhesignaali segmentoidaan diskreettiaikaiseksi, ja kullekin segmentille asetetaan yksi tai useampi foneettinen yksikkö vastaamaan puhutun kielen yksikköä. Varsinainen puheentunnistus suoritetaan toisessa vaiheessa, jossa foneemeista muodostetaan sana tai sanoja [3].

Kuvassa 2 [3] on tyypillisen akustis-foneettisen puheentunnistusjärjestelmän lohkokaavio. Analysointilohkossa puhesignaalia s(n) käsitellään siten, että foneemit ovat siitä helpommin havaittavissa [12]. Analysointi tapahtuu yleensä suodatinpankeilla tai LPC:llä ja sen tarkoituksena on datan minimointi ja erottelukyvyn maksimointi. Seuraavaksi käsitelty puhe ajetaan rinnakkaisesti ilmaisimien läpi, jotka karakterisoivat puheen ominaisuuksia halutuiksi yksiköiksi. Segmentoinnissa yksittäisistä yksiköistä muodostetaan varsinainen tunnistettu sana käyttäen esimerkiksi kätkettyjä Markovin malleja tai neuroverkkoja tai näiden hybridiä.



Kuva 2. Akustis-foneettisen puheentunnistusjärjestelmän lohkokaavio.

2.4.2. Tekoäly

Tekoälyyn perustuvassa lähestymistavassa yhdistellään ominaisuuksia sekä akustis-foneettisesta että hahmontunnistusta hyödyntävästä lähestymistavasta. Tavoitteena on jäljitellä ihmisen tapaa käsitellä puhetta visualisoimalla, analysoimalla ja tekemällä päätös sanasta tai sanoista mitattujen akustisten ominaisuuksien perusteella [3].

Kielimalliin on ohjelmoitu kielioppisääntöjä ja se antaa todennäköisyydet sanoille ja sanayhdistelmille perustuen yleensä suureen kieliaineistoon, jossa tehtävään liittyvä sanasto ja sanontatavat esiintyvät oikeissa tilastollisissa suhteissa. Esimerkiksi suomen kielessä vokaalin 'A' jälkeen ei voi suoraan esiintyä vokaalia 'Ä' ja sanaryhmä ”hyvää päivää” on paljon todennäköisempi kuin ”päivää hyvää”. Tämän kaltaisilla säännöillä puheentunnistuksesta saadaan robustimpaa ja sanavirhetodennäköisyys pienenee. [18, 19]

2.4.3. Hahmontunnistus

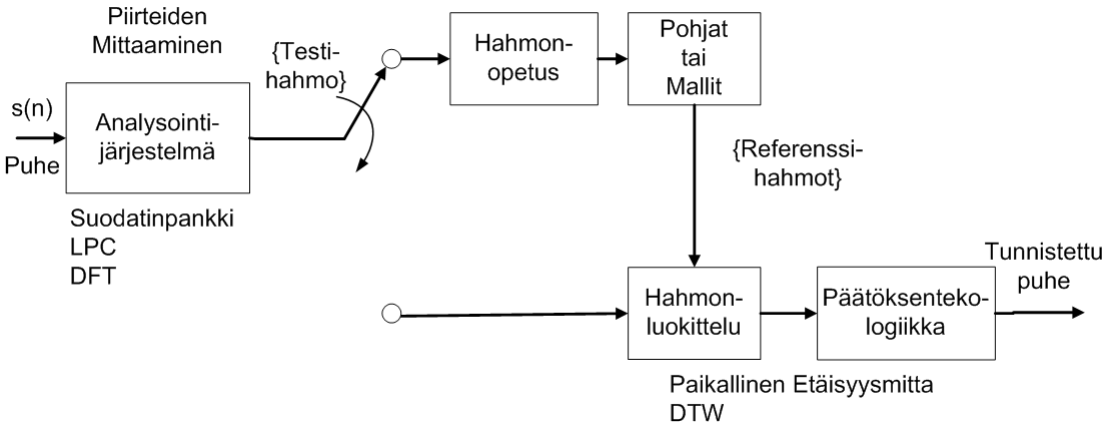
Hahmontunnistuslähestymistapa on yksinkertaisempi kuin akustis-foneettinen lähestymistapa, koska puhesignaalia ei tarvitse segmentoida. Se on myös kaksivaiheinen. Ensimmäinen vaihe on puhealueiden, kuten foneemien, kirjainten tai sanojen, opettaminen järjestelmälle. Varsinaisessa puheentunnistusvaiheessa pyritään tunnistamaan käyttäjän puhetta vertaamalla sitä opetettuihin puhealueisiin.

Kuvassa 3 on esitetty hahmontunnistusta käyttävän puheentunnistimen lohkokaavio. Puhesignaalin s(n) käsittelyyn voidaan käyttää esimerkiksi suodatinpankkeja tai LPC:tä kuten akustis-foneettisessa lähestymistavassa, mutta

myös diskreettiä Fourier-muunnosta (Discrete Fourier Transform, DFT). Ensimmäisessä, niin kutsutussa opetusvaiheessa puheentunnistusjärjestelmälle kerrotaan puheentunnistuksessa käytettävät yksiköt. Yleisesti ottaen, mitä enemmän järjestelmälle annetaan näytteitä, sitä virhesietoisemmaksi se tulee. Kutakin puheentunnistuksessa käytettävää yksikköä kohden luodaan yksi malli, joka voidaan kuvata analysoinnista keskiarvoistamalla tai karakterisoivalla tilastollisella referenssimallilla. Hahmonluokittelu-lohkossa verrataan sisään tulevan puhenäytteen samankaltaisuutta kuhunkin referenssimalliin, jotka pisteytetään perustuen seuraavaan kahteen mittaan:

- Paikallinen etäisyysmitta: Kahden hyvin määritellyn (referenssimalli ja sisään otettu puhe) piirrevektorin euklidinen etäisyys eli samankaltaisuus.
- Ajan sovitus: Kompensoidaan puhujan mahdollisia puhenopeuseroja dynaamisella ajansovitus-algoritmilla (Dynamic Time Warping, DTW).

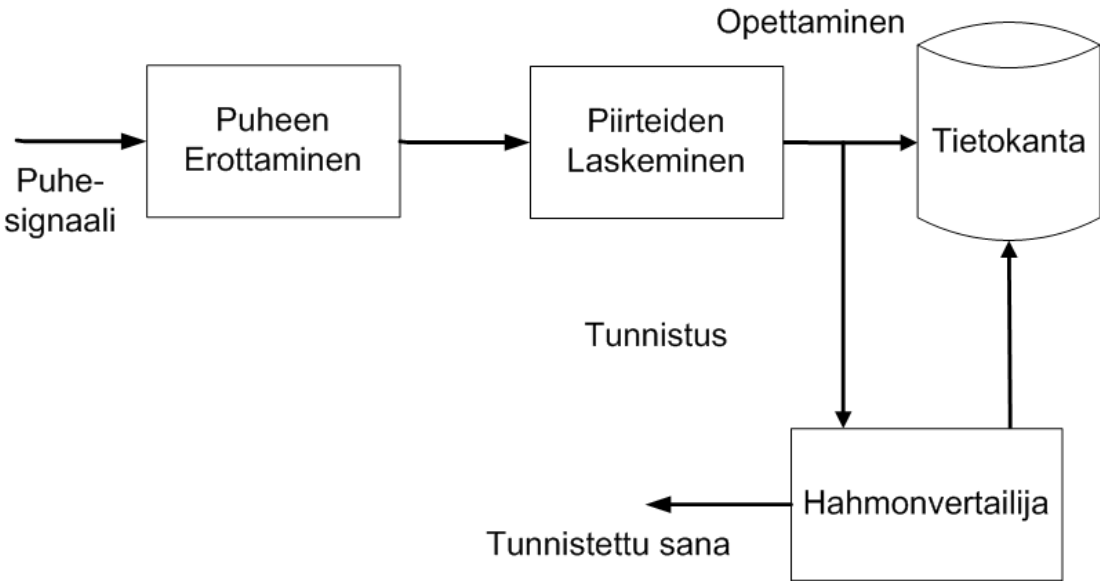
Erilaisia hahmontunnistuslähestymistapoja erottavia tekijöitä ovat puhesignaalin analysointimenetelmät, referenssimallien valinta sekä testimallien luokittelumetodit. Päätöksentekologiikka-lohkossa valitaan pisteytykseen perustuen se referenssimalli, joka vastaa parhaiten tuntematonta testimallia. [3]



Kuva 3. Hahmontunnistusta käyttävän puheentunnistimen lohkokaavio

2.5. Puheentunnistusjärjestelmän yksityiskohdat

Puheentunnistimen yleinen rakenne on esitetty kuvassa 4 [1]. Puheen erottaminen-lohkon tarkoituksena on rajata käyttäjän puhumat sanat taustakohinasta. Sanan alku- ja loppukohdan tarkka erottaminen sekä parantaa puheentunnistuksen laatua, että vähentää tarvittavaa laskentaa. Piirteiden laskeminen-lohkossa puhesignaali S(n):lle lasketaan piirrevektoreita, joita käytetään sanojen opetusvaiheessa ja varsinaisessa puheentunnistusvaiheessa. Hahmonvertailija-lohkossa verrataan opetettuja referenssisanoja puheentunnistusvaiheessa saataviin sanoihin.

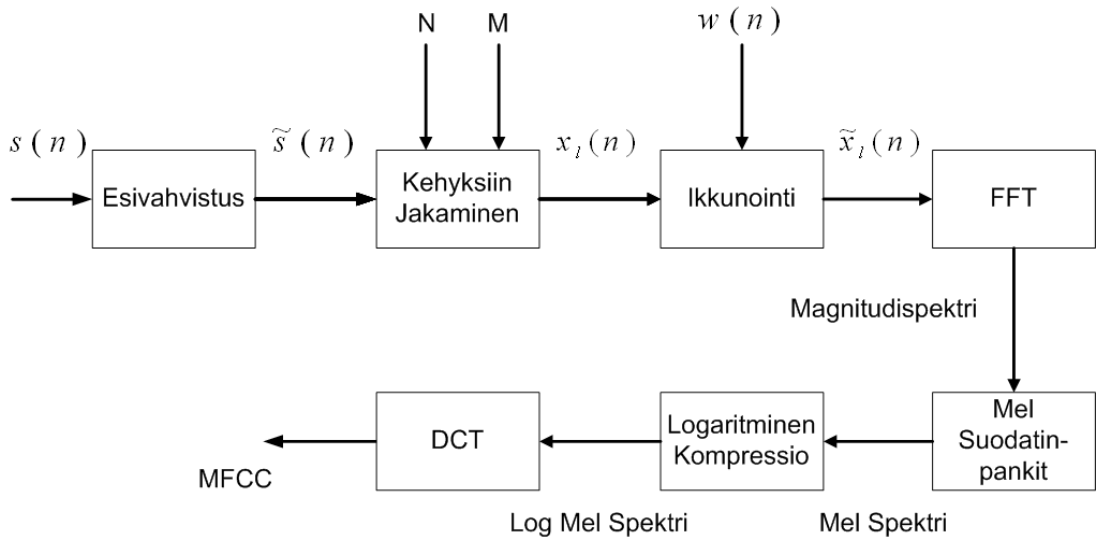


Kuva 4. ASR:n yleinen rakenne.

Tässä projektissa käytetään ASR:ssä luvussa 2.4.3. kuvattua hahmontunnistusmenetelmää, koska sen suoraviivainen toteutus sopii hyvin sulautetuille järjestelmille [20] ja projektin tiukalle aikataululle. Puhesignaalin analysointi ja sanojen piirrevektoreiden laskeminen on toteutettu MFCC:llä (Mel-Frequency Cepstral Coefficients) ja referenssisanojen vertailu tunnistettaviin sanoihin DTW:llä.

2.5.1. Mel-Frequency Cepstral Coefficients

MFCC:n toimintaperiaatteena on mallintaa ihmisen korvan kuulemaa taajuusvastetta. Sen vuoksi Mel-suodatinpankki on lineaarinen 1000 Hz asti ja logaritminen korkeammilla taajuuksilla. Kuvassa 5 [1] on MFCC:n lohkokaavio. Seuraavassa on lueteltu eri lohkojen tarkoitus ja toiminta: [1, 21]

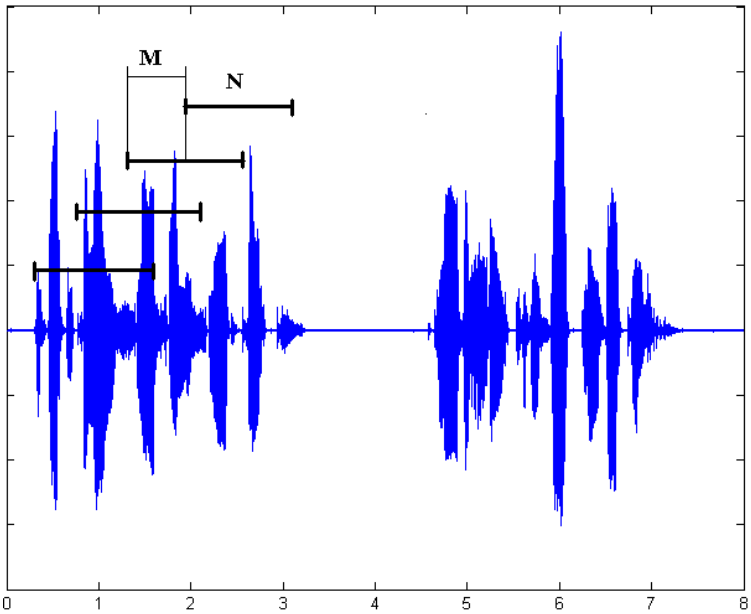


Kuva 5. MFCC:n lohkokaavio.

- Esivahvistus: Matalat taajuudet sisältävät enemmän mielenkiintoista informaatiota, joten matalia taajuuksia vahvistetaan yksitappisella FIR-suodattimella kaavan (2) mukaisella siirtofunktiolla. Lisäksi esisuodatuksella tasoitetaan signaalin spektriä. Kun käytetään kiinteän pisteen laskentaa, painokerroin a asetetaan usein arvoon 0,9375 [1].

$$H(z) = 1 - a*z^{-1}, \quad 0{,}9 < a < 1{,}0 \qquad (2)$$

- Kehyksiin jakaminen: Esisuodatettu puhesignaali s´(n) lohkotaan N:n näytteen kehyksiin, jotka erotetaan toisistaan M:llä näytteellä kuvan 6 mukaisesti. Käytettäessä 8 kHz:n näytteistystaajuutta ja 256-näytteen kehyksiä yhden kehyksen ajalliseksi kestoksi tulee siis 32 ms.



Kuva 6. Signaalin ositus.

- Ikkunointi: Ikkunoinnilla minimoidaan signaalin epäjatkuvuus kehyksen alussa ja lopussa. Samalla saadaan vähennettyä häiriötä signaalin spektrissä. Ikkunointiin käytetään Hamming-ikkunaa (3).

$$w(n) = 0{,}54 - 0{,}46*\cos(2*PI*n/(N-1)) \quad (3)$$

- Nopea Fourier-muunnos (FFT): Koska kehyksen pituudeksi valittiin 256 näytettä, voidaan spektrin laskennassa käyttää nopeaa Fourier-muunnosta. FFT:n toteutuksessa käytettiin Tom Robertsin kehittämää kiinteän pisteen laskentaa käyttävää avoimen lähdekoodin FFT-algoritmia.
- Mel-suodatinpankit: Approksimoidaan ihmisen korvan taajuusvastetta käyttämällä niin sanottua Mel-suodatinpankkia (Melody), joka on lineaarinen 1000 Hz asti ja logaritminen sen jälkeen. Mel-suodattimien kertoimet laskettiin etukäteen referenssitoteutuksella ja siirrettiin EIRin muistiin laskennan säästämiseksi (4).

$$\text{Mel-taajuus} = 2595 * \log(1 + f/700) \quad (4)$$

- Logaritminen kompressio: Mel-suodattimista saatavista arvoista otetaan luonnollinen logaritmi. Näin saadaan signaalin spektri suunnilleen Gauss-jakautuneeksi.
- Diskreetti kosinimuunnos (DCT): Siirrytään takaisin aikatasoon. Samalla saadaan typistettyä kehys p:n mittaiseen piirrevektoriin, jota käytetään seuraavassa vaiheessa referenssimallien tallentamiseen ja sanojen tunnistamiseen (5).

$$\mathrm{MFCC}_k = \sqrt{\frac{2}{M}} \sum_{m=1}^{M} X_m \cos\left(\frac{\pi k(m - 0{,}5)}{M}\right), \quad 1 \le k \le p \quad (5)$$

missä MFCC_k on Mel-taajuuskepstrikertoimet, M on Mel-suodattimien lukumäärä ja X_m on käsitelty puhesignaali.

Liitteessä 2 on kuvattu puhekomentojärjestelmän tuottamat näytteistetyt signaalit vaihe vaiheelta.

2.5.2. Dynamic Time Warping

Euklidista etäisyyttä käytetään laajalti staattisten aikajaksojen evaluointiin [SMunderDTWD08, 22], mutta koska puhuttujen sanojen kestot vaihtelevat,

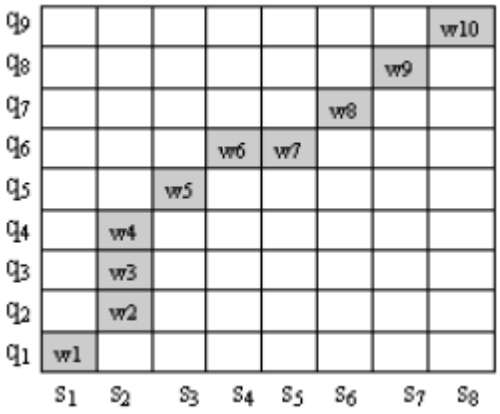
tarvitaan puheentunnistuksessa parempi ratkaisu. Dynamic Time Warping on usein käytetty ratkaisu signaalinkäsittelyssä ja puheentunnistuksessa.

Kuvassa 7 [23] on esitetty graafisesti DTW:n toimintaa. Yksittäiset ruudut kuvaavat MFCC-vektoreiden alkioita, rivit referenssivektoreita ja sarakkeet testivektoreita. Euklidinen etäisyys (6) lasketaan kumulatiivisesti etenemällä aina joko ylös, oikealle tai oikealle yläviistoon pienimmän euklidisen etäisyyden ohjaamana. Jos D(i,j) on globaali etäisyys pisteeseen (i,j) ja paikallinen etäisyys pisteessä (i,j) on d(i,j), kun

$$d(x,y) = \sum \sqrt{(x_i - y_j)^2} \qquad 1 \le i \le n,\ 1 \le j \le n \tag{6}$$

$$D(i,j) = \min[D(i-1,j-1), D(i-1,j), D(i,j-1)] + d(i,j).. \tag{7}$$

Näin polku etenee vasemmasta alanurkasta oikeaan ylänurkkaan. Polun laskenta suoritetaan jokaiselle ASR:n tukemalle sanalle erikseen ja valitaan tunnistetuksi sanaksi se, jota vastaan DTW antaa pienimmän kokonaispistemäärän kaavan (7) mukaisesti [23, 24].



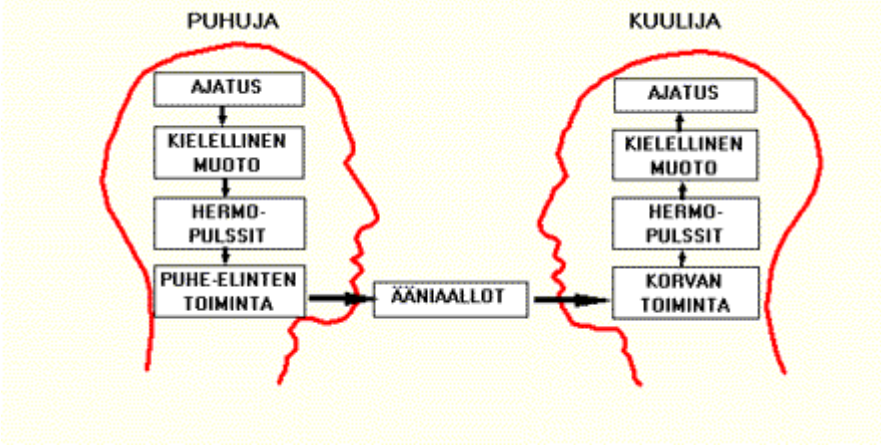
Kuva 7. DTW-algoritmi.

Pitkäkestoisen signaalin tutkiminen DTW:llä vaatii laitteistolta paljon muistia [GOp2006, 25], mutta käyttämällä yksittäisiin sanoihin perustuvaa puheentunnistusta, noin kymmenen sanan tunnistettavuutta ja 8 kHz näytteistystaajuutta muistivaatimukset eivät kasvaneet liian suuriksi.

2.6. Fonetiikka

Fonetiikka on kielitieteen erikoisalue, joka tutkii puhetta sen kaikissa muodoissa. Tieteenalan perinteinen jaottelu erottelee **puheen tuottamisen** eli miten puhuja muuttaa aikomansa kielellisen sanoman ääntöelimistön toiminnaksi, **puheen ääniopin** eli mitkä ovat ääntöelimistön toiminnan akustiset seuraukset ja **puheen havaitsemisen** eli miten kuulija muuttaa ääniaallot kielelliseksi sanomaksi, jossain määrin itsenäisiksi, mutta perusteiltaan toisistaan riippuviksi tutkimusalueiksi. Fonetiikassa tutkimuskohdetta lähestytään kolmesta näkökulmasta, sillä puhe on

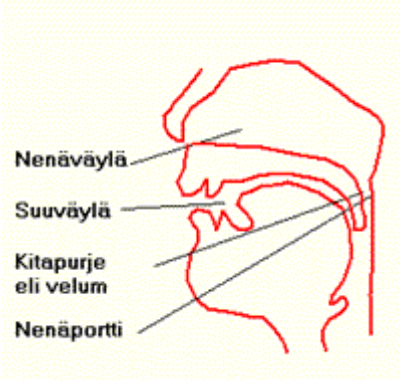
yksilöllinen, kielikohtainen ja universaali ilmiö. Toisin sanoen puheessa on puhujalle ominaisia, yksilöllisiä ominaisuuksia ja kielikohtaisia piirteitä sekä kaikille maailman kielille yhteisiä ominaisuuksia. Kuva 8 [26] kuvaa yksinkertaista kommunikointitilannetta, jossa puhuja sanoo kuulijalle jotakin. [26, 27, 28]



Kuva 8. Puhetapahtuman vaiheet.

Artikulatorinen fonetiikka tutkii muun muassa puheentuottamismekanismia (kuva 9 [29]) ja sen toimintaa. Mekanismin muodostavat keuhkot, henkitorvi, kurkunpää ja ääntöväylät (kuva 10 [29]) eli suu ja nenä. Puhe syntyy uloshengityksen aikana ja siihen tarvitaan keuhkoja. Ilma virtaa henkitorvea pitkin kohti kurkunpäätä, joka on esitetty kuvassa 11 [29]. Kun kurkunpäässä sijaisevat äänihuulet (kuva 11) värähtelevät, syntyy puheen sointi ja puheääni. Kurkunpäästä ilma nousee ylöspäin ja kulkee ulos ääntöväylien kautta.

Kuva 9. Puheentuottamismekanismi.



Kuva 10. Ääntöväylät, suu ja nenä.

Kuva 11. Kurkunpää, äänirako ja äänihuulet.

Kuva 12. Artikulaatiovyöhykkeet.

Artikulaatioelimistössä tuotetaan erilaiset äänteet. Kuvassa 12 [29] on kuvattu eri artikulaatiovyöhykkeet, joiden mukaan äänteitä nimitetään. Äänivyöhykkeet ovat:

1) huulet (labia),
2) hampaat (dentes),
3) hammasvalli (alveolum),
4) kitalaki eli suulaki (palatum),
   4a) kova kitalaki (palatum durum),
   4b) kitapurje (velum),
5) kitakieleke (uvula),
6) nielu (farynks),
7) äänirako (glottis),
8) kielen kärki (apeks),
9) kielen lapa (korona),
10) kielen laiteet (latera),
11) kielen selkä (dorsum),
12) kielen tyvi (radiks).

Äänteiden luokittelu sen perusteella, miten ja missä kohdassa artikulaatioelimistöä ne syntyvät, on myös osa artikulatorista fonetiikkaa.

Akustinen fonetiikka tutkii puhetta, kun se matkaa ääniaaltoina puhujalta kuulijalle. Se käsittelee puheen akustista rakennetta ja fysikaalisia ominaisuuksia sekä äänteiden luokittelua akustisin perustein. **Auditiivinen fonetiikka** tutkii puhetta kuulijan kannalta eli puheen havaitsemista. Korvan rakenne ja ominaisuudet ovat myös osa auditiivista fonetiikkaa. Muita fonetiikan tutkimusalueita ovat muun muassa teoreettinen fonetiikka ja soveltava fonetiikka. [26]

2.7. Laivanupotus

Laivanupotuspelissä kaksi amiraalia yrittää tuhota toistensa laivaston. Amiraalit ampuvat vuorotellen tykistötulta vihollisen alueelle yrittäen upottaa viholliset laivat. Laivat on sijoitettu 10 x 10 pelialueelle ilman, että vastapuoli näkee laivojen sijaintia. Jokaista laukausta kohden pelaaja saa ilmoituksen menikö ammus ohi tai osuiko se laivaan. Jos ammus upotti laivan, ilmoitetaan laivan uppoamisesta. [30]

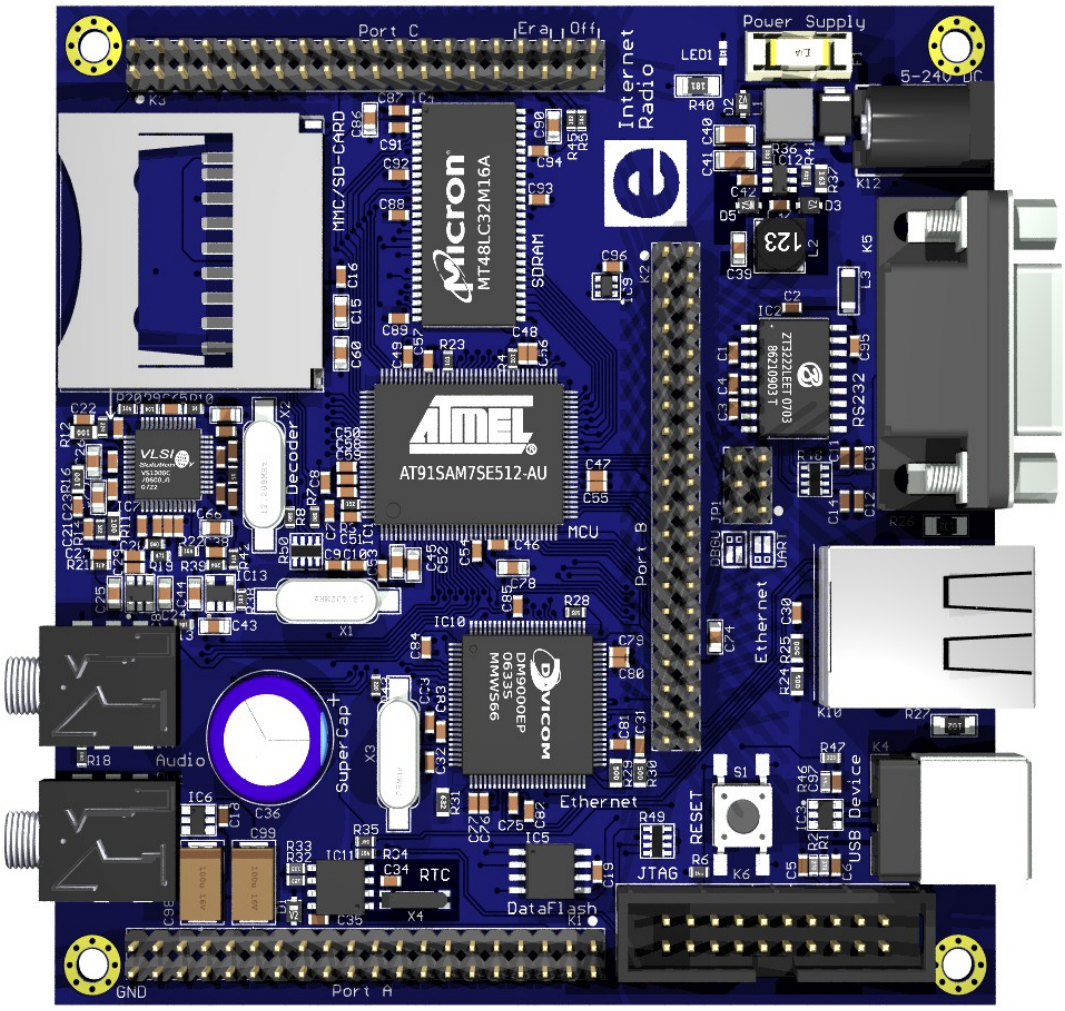
2.8. Käytettävä laitteisto

Tässä projektissa käytettävä laitteisto on Elektor Internet Radio versio 1.0 eli EIR-sulautettu järjestelmä. Kuva 13 [31] esittää EIR-lautaa ylhäältäpäin katsottuna. Sen tärkeimmät ominaisuudet [31] ovat:

- AT91SAM7SE512 mikrokontrolleri 512 kT:n nopealla Flash muistilla.
- VS1053 audiodekooderi, joka tukee MP3, AAC+, WMA ja Ogg Vorbis -formaatteja
- Audion sisääntulo and ulostulo 1/8 tuuman stereo-liittimillä.

- Kaksisuuntainen dupleksi, IEEE 802.3 yhteensopiva 10/100 Mbps Ethernet kontrolleri RJ-45 liittimellä.
- 2 sarjaporttia, yksi RS232 DB-9 liitännässä.
- 64 MT RAM:ia ohjelmakoodille ja datalle.
- 4 MT sarjamuotoista Flash muistia.
- MultiMedia/SD-korttipaikka.
- 16 digitaalista I/O linjaa erityisfunktioilla.
- 4 analogista sisääntuloa
- Reaaliaikainen kello EDLC varmistuksella.
- LED ilmaisimet käyttöjännitteelle ja Ethernet toiminnalle.
- Joustava käyttöjännite 5-24V DC.
- Laudan koko: 100 x 100 mm (3.9 x 3.9 tuumaa).

EIR:oon on liitetty myös I/O-lauta, joka sisältää muun muassa näppäimistön, joystickin, liittimen mikrofonille ja kiihtyvyysanturin. EIR:n ohjelmointiin käytetään avoimeen lähdekoodiin perustuvaa Nut/OS reaaliaikaista käyttöjärjestelmää (RTOS) [32].



Kuva 13. EIR-lauta ylhäältäpäin.

3. RATKAISUN KUVAUS

3.1. Johdanto

Projektityön tavoitteena oli toteuttaa sulautetulle järjestelmälle automaattista puheentunnistusta hyödyntävä ratkaisu. Käytännön sovelluskohteena oli laivanupotuspelin toteuttaminen. Projektin haastavin osuus oli puheentunnistuksen hyödyntäminen siten, että vaihtoehtoisena syötteenä pelille voitiin käyttää ihmispuhetta. Puheentunnistuksen menetelmät kartoitettiin ja niistä soveltuvat menetelmät valittiin projektin tarpeisiin.

Puheentunnistusjärjestelmäksi valittiin yksittäisten sanojen tunnistamiseen perustuva järjestelmä, jolla oli mahdollista saavuttaa hyvä tarkkuus puheentunnistuksessa. Tunnistettavien sanojen määrä rajoittui 4 sanaan, joten yksittäisten sanojen tunnistusjärjestelmä soveltui hyvin käyttötarkoitukseen.

Puhekomentojärjestelmä toteutettiin hahmontunnistukseen perustuvalla lähestymistavalla. Piirreanalyysi päädyttiin tekemään Mel Frequency Cepstral Coefficients –menetelmällä (MFCC) ja vertailussa valittiin käytettäväksi Dynamic Time Warping –menetelmää (DTW). Näiden menetelmien tarkemmat kuvaukset on esitetty kappaleessa 2.5. Puheentunnistusjärjestelmän yksityiskohdat.

Nykyisin sekä FFT- että LPC-pohjaiset tekniikat ovat laajalti käytössä hahmontunnistuksessa. FFT on häiriösietoinen meluisassa ympäristössä ja on suosittu, koska FFT-menetelmän vaiheet ovat samankaltaiset ihmisen kuulojärjestelmän toiminnan kanssa [1, 33]. MFCC käyttää FFT:tä osana toteutusta. LPC-menetelmän havaittu laskennallinen raskaus puolsi myös MFCC-menetelmän valintaa.

Ratkaisu perustuu yhden puhujan puheen tunnistamiseen. Laitealustana toimi Elektor Internet Radio eli EIR sulautettu järjestelmä, johon kuului erillinen I/O-lauta. Järjestelmän täytyi pystyä kommunikoimaan lähiverkon yli muiden samanlaisten laitteiden kanssa mahdollistaen kahden pelaajan välisen laivanupotuspelin pelaamisen. Tämä toiminto toteutettiin käyttäen SNMP-protokollaa (Simple Network Management Protocol).

3.2. Rajoitukset

EIR-laudan prosessori- ja muistikapasiteetti asettivat rajoituksensa valittavalle ratkaisulle. Rajallisen prosessoritehon vuoksi yritimme välttää rekursion ja muiden laskennallisesti raskaiden metodien käyttöä. Koska EIR ei sisällä liukulukuyksikköä, skaalasimme liukuluvut sopiviksi kokonaisluvuiksi laskennan nopeuttamiseksi. Suhteellisen pienen muistikapasiteetin vuoksi käytimme jonkin verran dynaamista muistin varausta ja osoittimia funktiokutsuissa.

Projektin tiukasta aikataulusta johtuen toteutusta yksinkertaistettiin jonkin verran ja muun muassa joystickin toiminta jätettiin toteuttamatta. Alkuperäisenä ajatuksena oli käyttää sitä apuna ammuskoordinaattien valitsemisessa merta kuvaavalta ruudukolta.

3.3. Tietorakenteet

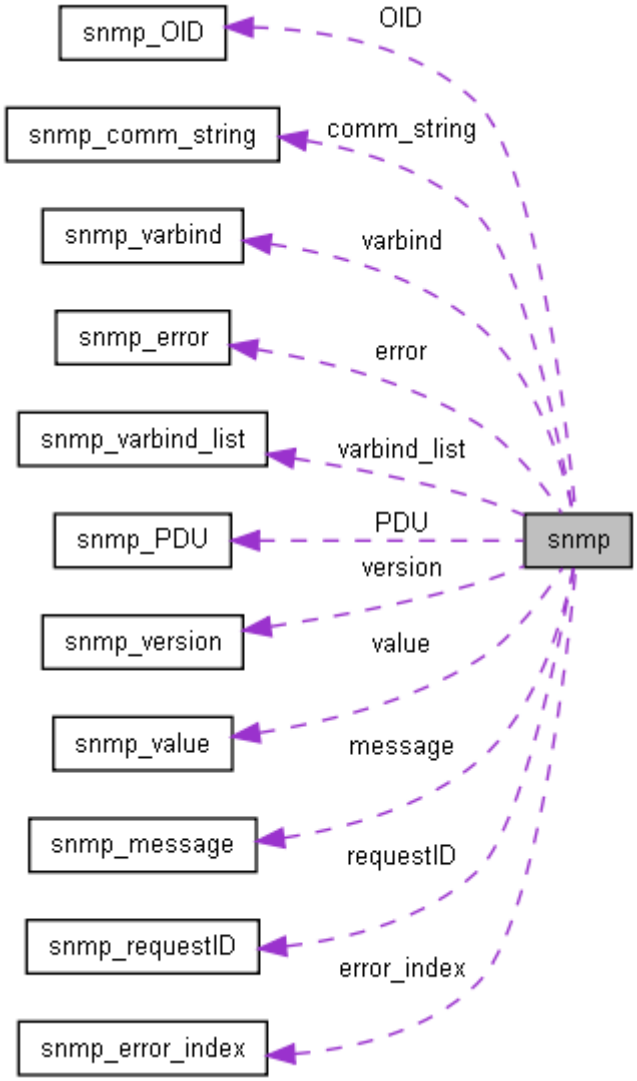
Tietorakennekuvaukset on luotu käyttäen Doxygen-dokumentointiohjelmaa [34]. Lähdekoodi on dokumentoitu Doxygenin vaatimusten mukaisesti ja tietorakennekuvaukset on generoitu automaattisesti ja muokattu tämän dokumentin tyyliin sopiviksi. Tähän dokumenttiin on käytännön syistä otettu vain muutama tietorakennekuvaus ja projektin täydellinen API-dokumentaatio (Application Programming Interface) löytyy internetistä osoitteesta http://www.ee.oulu.fi/~codec. Käyttäjätunnus on: sop ja salasana S0pp1 (toinen merkki on nolla ja viimeinen numero yksi).

Taulukossa 2 on kuvattu esimerkinomaisesti verkkotoimintoihin ja tarkemmin sanottuna snmp-tietueeseen liittyvät tietueet ja datakentät. Kuvassa 14 on esitetty snmp-tietueen yhteistyökaavio.

Taulukko 2. Snmp-tietueen yhteistyökaavio.

Tietue	***Tietueen datakentät***
snmp	struct **snmp_message message**
	struct **snmp_version version**
	struct **snmp_comm_string comm_string**
	struct **snmp_PDU PDU**
	struct **snmp_requestID requestID**
	struct **snmp_error error**
	struct **snmp_error_index error_index**
	struct **snmp_varbind_list varbind_list**
	struct **snmp_varbind varbind**
	struct **snmp_OID OID**
	struct **snmp_value value**
snmp_message	u_char **message_type**
	u_char **message_length**
snmp_version	u_char **version_type**
	u_char **version_length**
	u_char **version_value**
snmp_comm_string	u_char **comm_string_type**
	u_char **comm_string_length**
	u_char **comm_string_value** [10]
snmp_PDU	u_char **PDU_type**
	u_char **PDU_length**

snmp_requestID	u_char **requestID_type**
	u_char **requestID_length**
	u_char **requestID_value** [5]
snmp_error	u_char **error_type**
	u_char **error_length**
	u_char **error_value**
snmp_error_index	u_char **error_index_type**
	u_char **error_index_length**
	u_char **error_index_value**
snmp_varbind_list	u_char **varbind_list_type**
	u_char **varbind_list_length**
snmp_varbind	u_char **varbind_type**
	u_char **varbind_length**
snmp_OID	u_char **OID_type**
	u_char **OID_length**
	u_char **OID_value** [7]
snmp_value	u_char **value_type**
	u_char **value_length**
	int **value_value** [50]

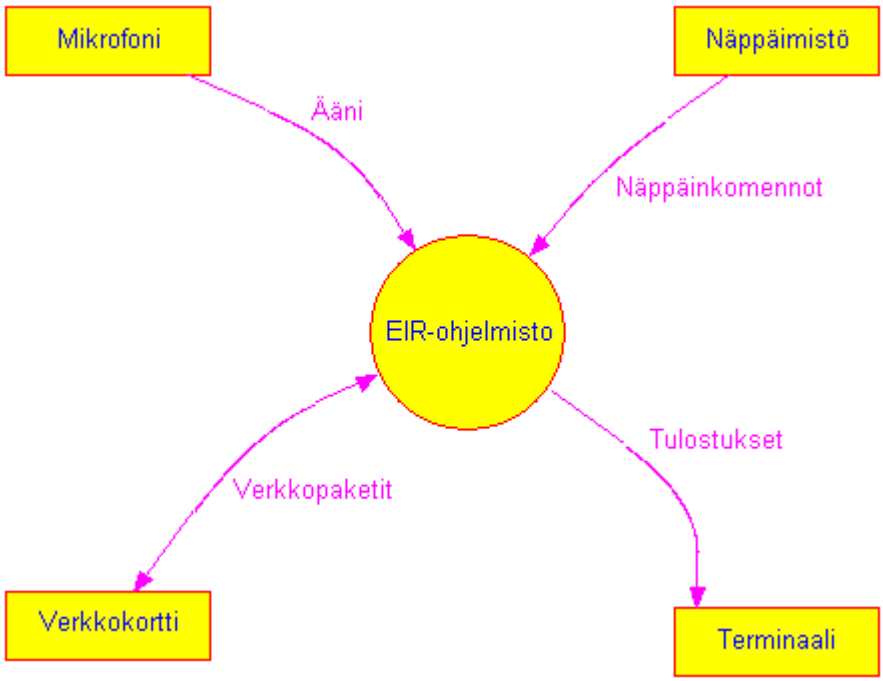


Kuva 14. Snmp-tietueen yhteistyökaavio.

3.4. Arkkitehtuurikuvaus

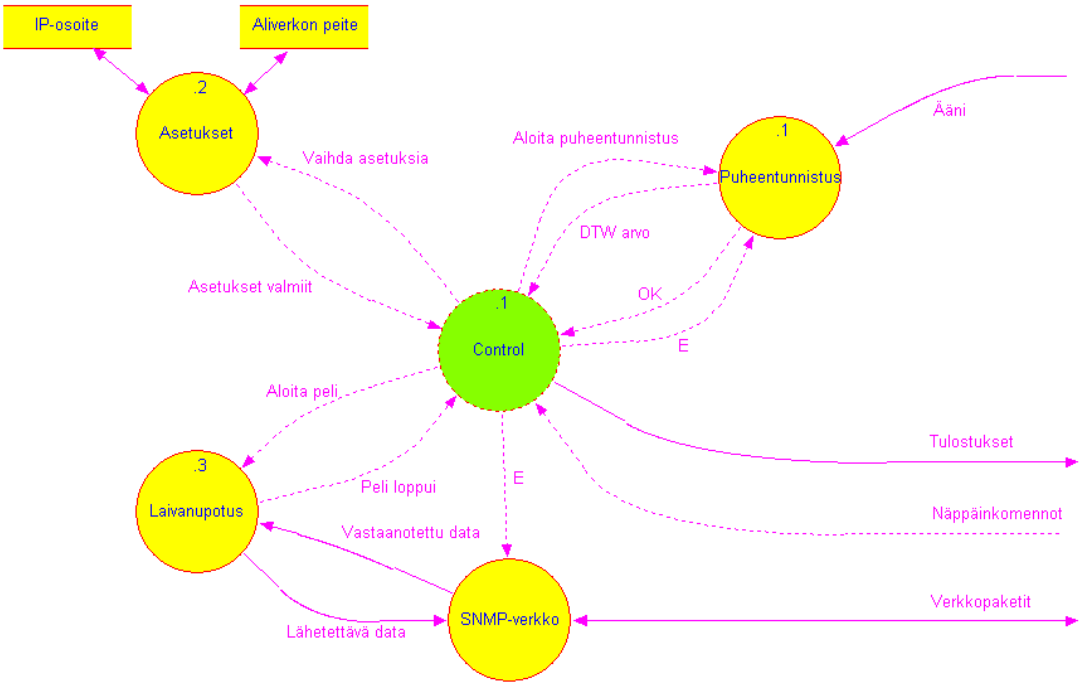
3.4.1. SA/SD

Kuvassa 15 on esitetty ohjelmiston yhteys käytettävään laitteistoon.



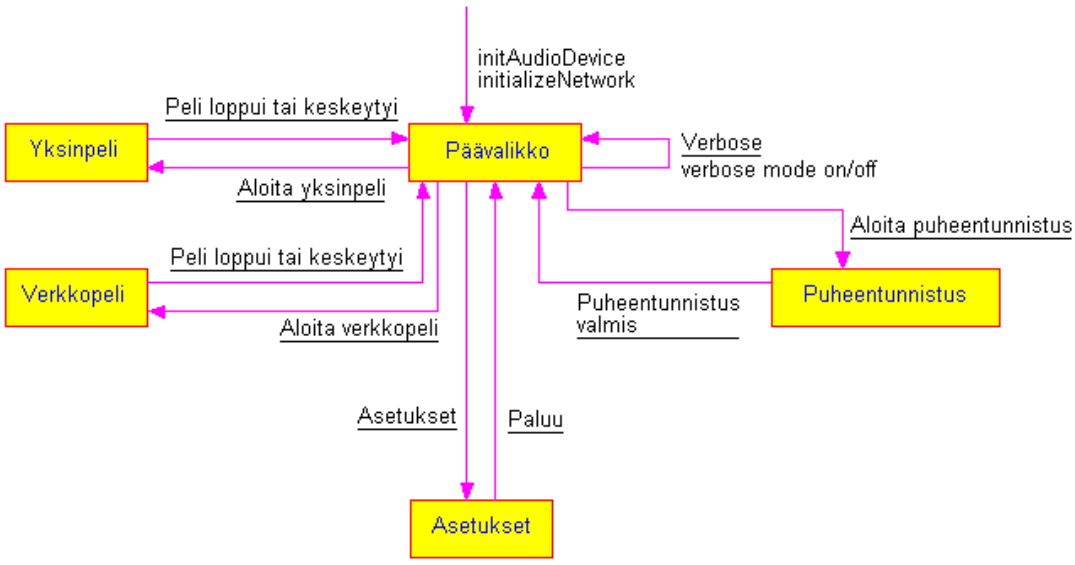
Kuva 15. Järjestelmän kontekstikaavio.

Kuvassa 16 on esitetty järjestelmän nollataso. Control-ohjausmuunnoksessa sijaitseva main-funktio ohjaa koko järjestelmän toimintaa. Ohjelman käynnistyessä suoritetaan alustukset äänipiirille, verkkokortille ja UART:lle. Äänipiirin alustuksessa EIR:n VS1053 audiodekooderi asetetaan tilaan, jossa se on valmis äänen tallentamista varten. Verkkokortin alustuksessa asetetaan oletus IP-osoite 10.10.4.1 ja julkinen aliverkon peite 255.255.0.0. Näitä asetuksia voi muuttaa ohjelman ajon aikana changeIP- ja changeSNM-funktioilla, jotka ovat Asetukset-valikossa. Pelin päävalikosta valitaan haluttu toiminto, kuten yksin- tai kaksinpeli. UART:n kautta suoritetaan tulostukset PC:n ja EIR:n välillä sarjakaapelia käyttäen.



Kuva 16. Järjestelmän nollataso.

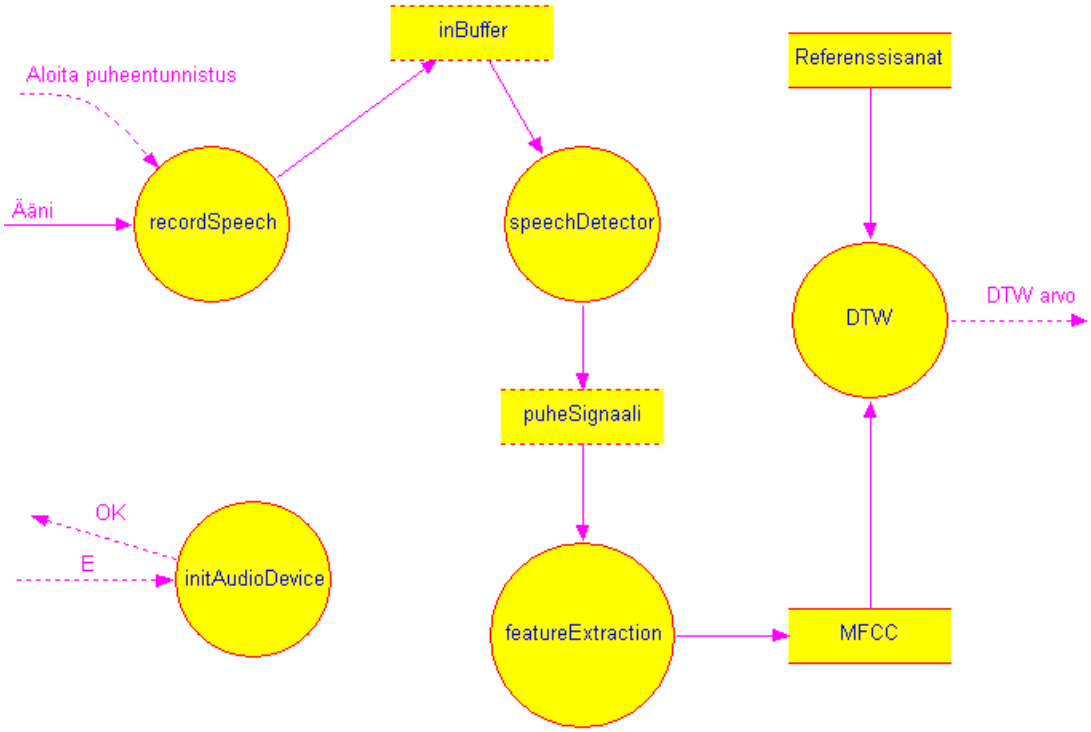
Main-funktiosta (kuva 17) ohjataan kaikkia järjestelmän toimintoja. Switch-case-rakenteella valitaan haluttu toiminto kappaleen 3.6 käyttöliittymäkuvauksen mukaisesti. Käyttöliittymää voidaan ohjata joko näppäimistöllä tai puheentunnistuksella.



Kuva 17. Main-funktion tilakaavio.

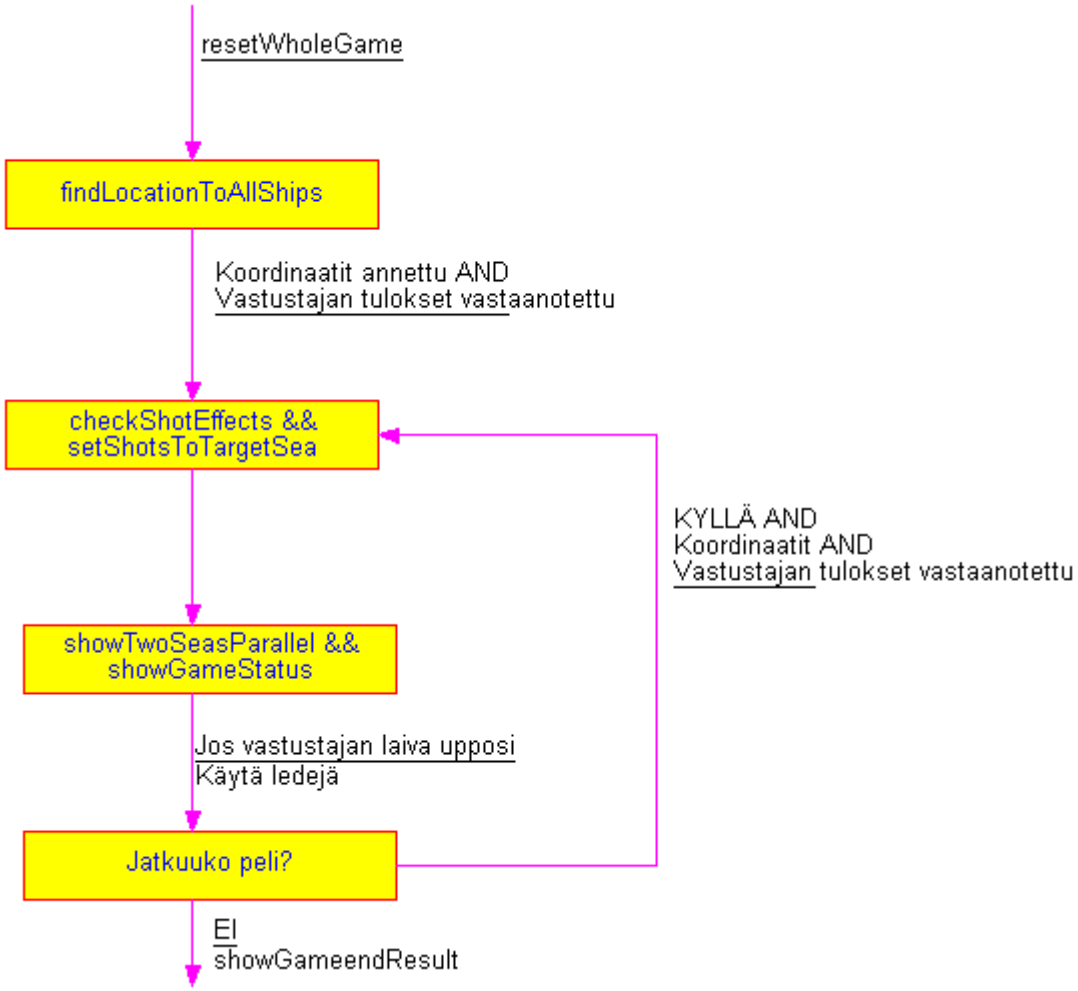
Kuvassa 18 esitetty puheentunnistusmoduuli suorittaa varsinaisen puheentunnistuksen. Puhe nauhoitetaan puskuriin käyttäen pohjana esimerkkikoodina annettua vsutils-ohjelmaa. Sen jälkeen signaalista erotetaan varsinainen puhesignaaliosa kohinasta. Puhesignaali annetaan syötteenä MFCC-kertoimet laskevalle moduulille, jonka tarkempi kuvaus on kappaleessa 2.5.1. Kun kertoimet on laskettu, verrataan käyttäjän puhumaa sanaa kaikkiin järjestelmän

muistiin tallennetuista sanoista, ja valitaan tunnistetuksi sanaksi kappaleessa 2.5.2 kuvattua DTW-algoritmia käyttäen se, joka on lähinnä käyttäjän sanomaa sanaa.



Kuva 18. Puheentunnistuksen tietovuokaavio.

Kuvassa 19 on esitetty laivanupotuspelin tilakaavio ja taulukossa 3 moduulien minispesifikaatiot.



Kuva 19. Laivanupotuspelin tilakaavio.

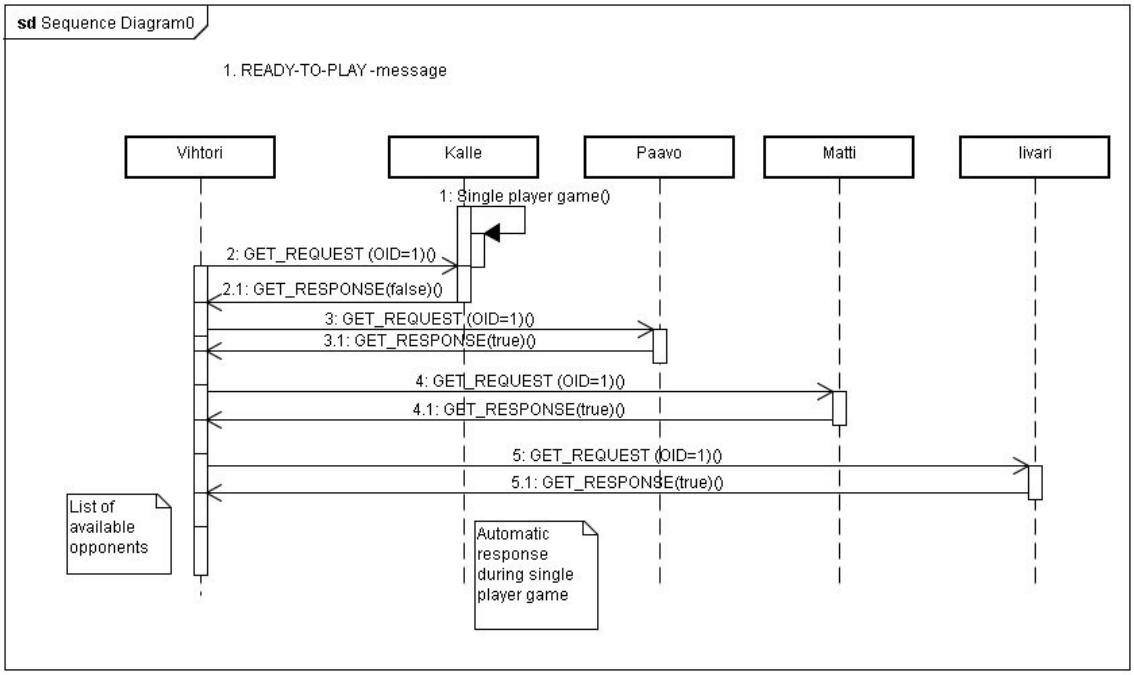
Taulukko 3. Moduulien minispesifikaatiot.

Moduuli	*Minispesifikaatio*
Asetukset	settings-funktiolla voidaan vaihtaa järjestelmän IP-osoitetta ja aliverkon peitettä changeIP- ja changeSNM-funktioiden avulla.
Verkko	Verkkomoduuliin toteuttiin säie, joka kuuntelee jatkuvasti kanavan viestejä ja vastaa niihin tarvittaessa spesifikaation mukaisesti. Kanavasta tulevat SNMP-viestit parsittiin snmp-tietueeseen, jolloin paketeissa kulkeva data saatiin erotettua muusta viestistä. Viestien lähetys toteutettiin myös tietueiden kautta, jolloin lähetettävät viestit saatiin helposti parsittua ja lähetettyä kanavaan. Verkko- ja laivanupotusmoduulin välisessä viestinnässä käytettiin lippujärjestelmää ilmaisemaan vastaanotetun datan valmiutta ja lähetettävien viestien oikea aikaista lähettämistä.
recordSpeech	Näyte talletetaan muistiin käyttäen 16020 näytteen puskuria. Sanalle on näin ollen varattu noin kaksi sekuntia tilaa näytteistystaajuuden ollessa 8000 Hz.

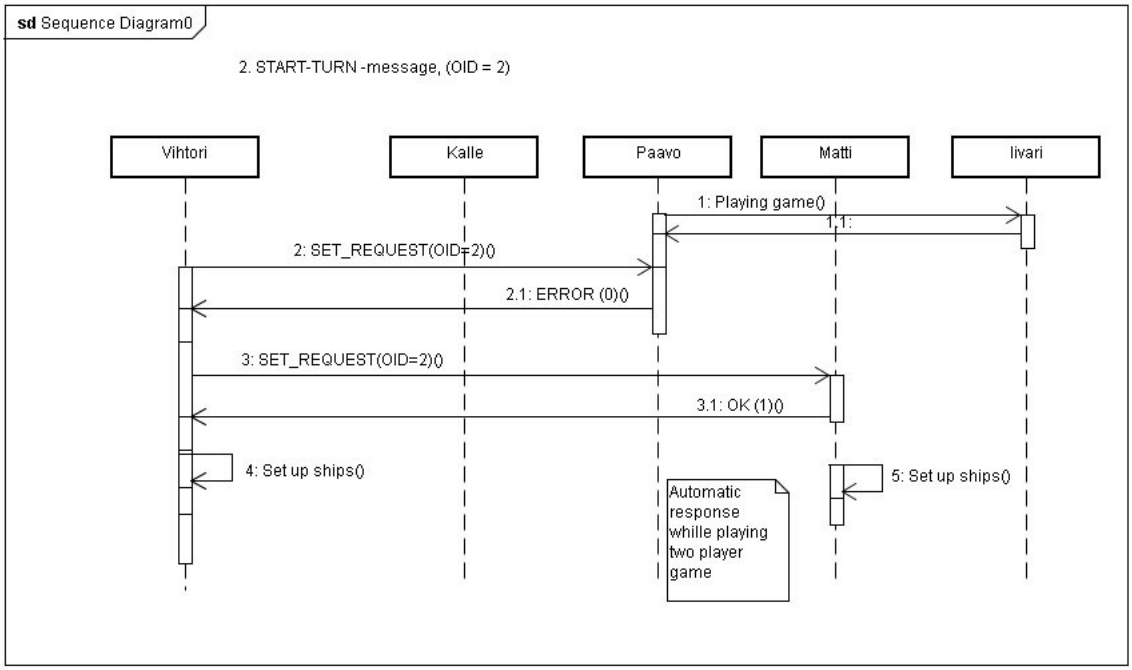
speechDetector	16020 näytteen puskurista leikataan pois pelkkää kohinaa sisältävät osat, jolloin puheentunnistuksesta tulee robustimpaa. Leikkaus on toteutettu asettamalla kohinalle raja-arvo. Kun raja-arvo ylitetään, on puskurissa puhetta. Sen jälkeen näytteitä tarkastellaan ennakoiden 1500 seuraavaa näytettä. Kun puhuttu sana loppuu, leikataan puskurissa oleva kohina pois, eikä se näin häiritse puheentunnistuksen piirrevektoreiden laskentaa.
FeatureExtraction	Puheelle lasketaan mel-taajuuskepstrikertoimet kappaleen 2.5.1 mukaisesti.
DTW	Dynaaminen ajan sovitus vertaa käyttäjän puhumaa sanaa järjestelmän muistiin talletettuihin referenssisanoihin ja valitsee niistä parhaan kappaleessa 2.5.2 kuvatun algoritmin mukaisesti.

3.4.2. Kahden pelaajan pelitapahtumat

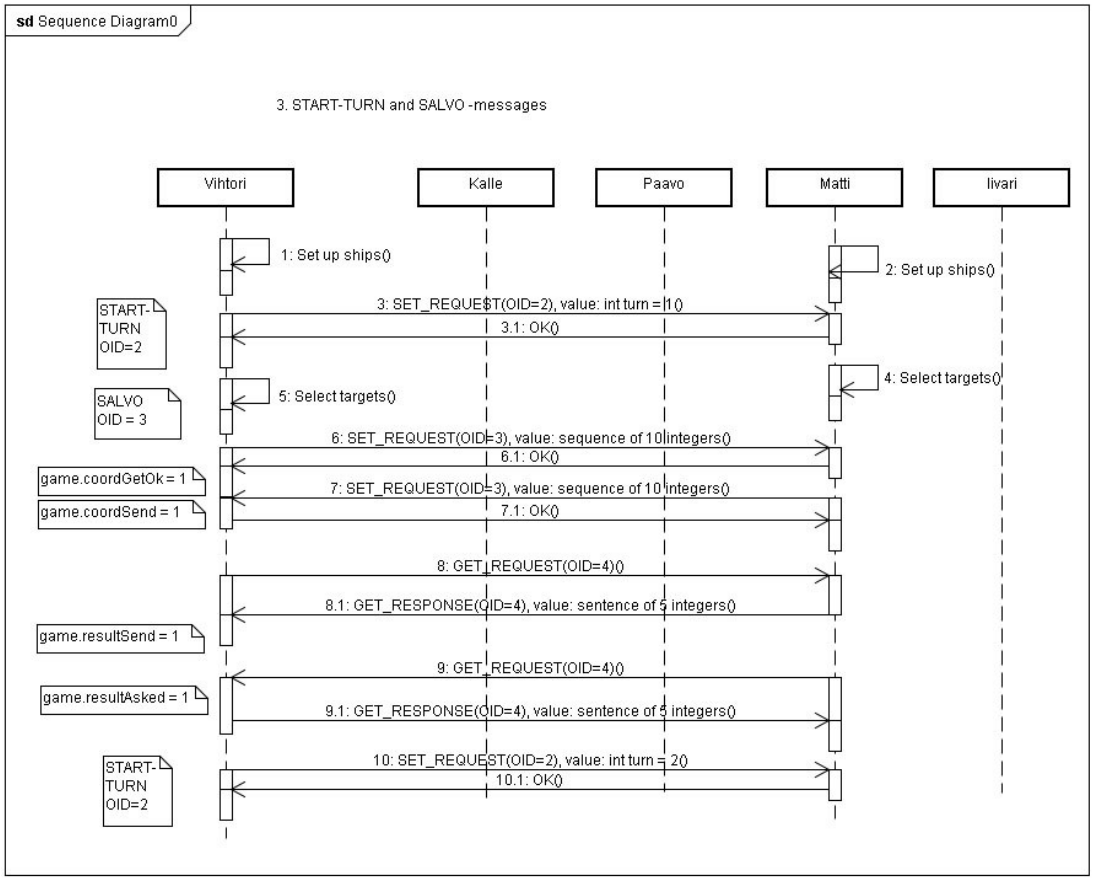
Pelitapahtumien eteneminen kahden pelaajan pelin eri pelitilanteissa on kuvattu sekvenssikaavioilla kuvissa 18-22. Viestinvälitys tapahtui käyttäen SNMP -protokollaa.



Kuva 18. Kahden pelaajan pelin aloitus. Pelaaja Vihtori etsii pelikaveria lähettämällä GET_REQUEST-viestin muille pelaajille. Kalle pelaa yhden pelaajan peliä ja ilmoittaa, ettei ole käytettävissä. Muut pelaajat ilmoittavat suostumuksesta. Vihtori saa listan vapaista pelikavereista.



Kuva 19. Vihtori päättää valita pelikaveriksi Paavon, ja lähettää vain Paavolle pelipyynnön. Vihtorin miettiessä pelaajat Paavo ja Iivari ehtivät aloittaa oman pelin. Paavo vastaa virheviestillä. Seuraavaksi Vihtori lähettää pelipyynnön Matille, joka lähettää myönteisen vastauksen. Molemmat pelaajat asettavat alukset omille pelialueilleen.



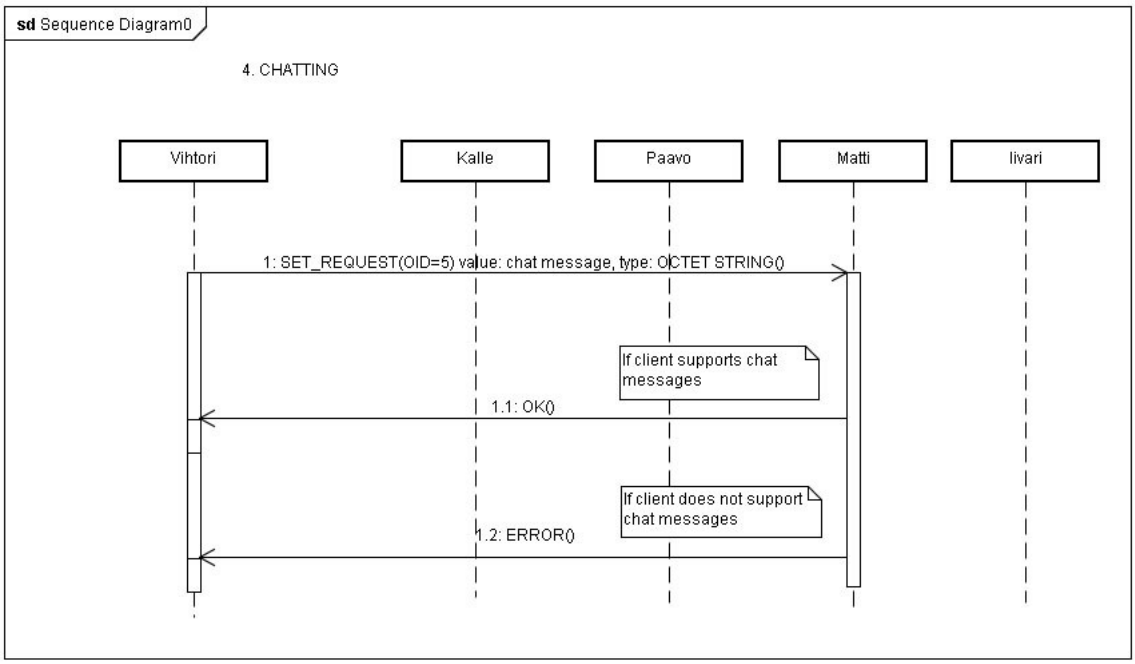
Kuva 20. Kun Vihtori on asettanut laivansa pelialueelle, Vihtori lähettää SET_REQUEST-viestin Matille (kierros, turn=1), joka vastaa siihen kun on valmis. Molemmat pelaajat valitsevat kohdekoordinaattinsa ja lähettävät ne SET_REQUEST(OID=3) viestissä kymmenen kokonaisluvun sarjana. Toinen pelaaja vastaa OK. Molemmat pelaajat kysyvät tulitaistelun tuloksia ja saavat niihin vastaukset viitenä peräkkäisenä kokonaislukuna. Sen jälkeen alkaa uusi tulitaistelu.

Tulitaistelun tulokset ilmoitetaan numeerisesti seuraavasti: 0 = ohi, 1 = osuma, 2 = hävittäjä upposi, 3 = risteilijä tai sukellusvene upposi, 4 = taistelulaiva upposi ja 5 = lentotukialus upposi.

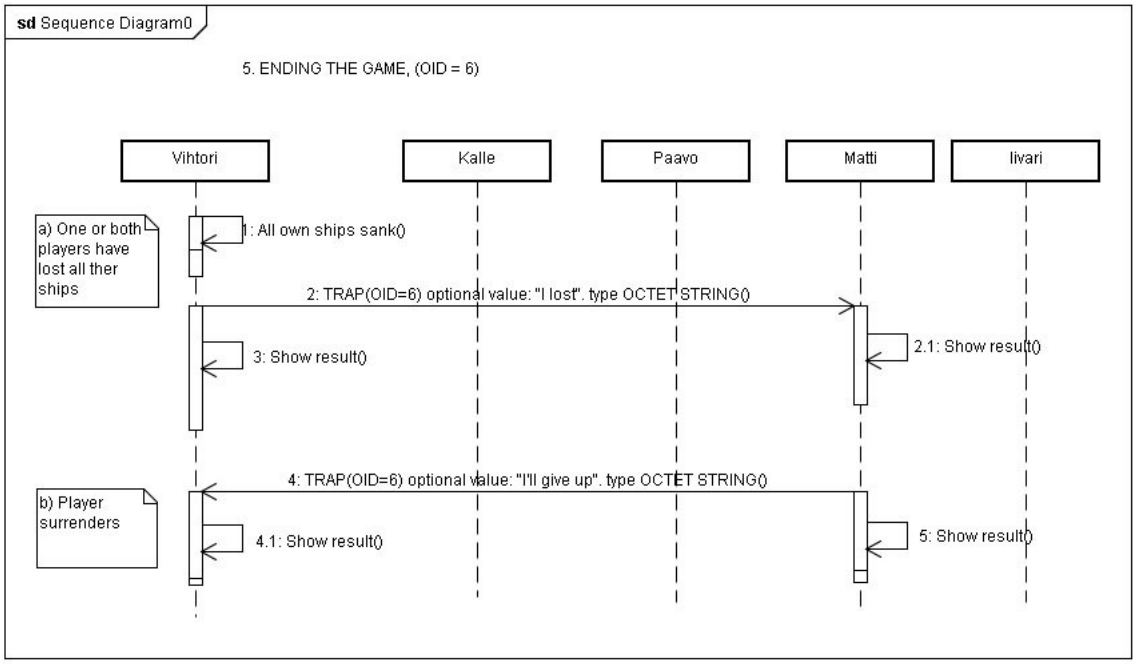
Jos koordinaatit osuvat pelialueen ulkopuolelle, ilmoitetaan ohi menneestä ammuksesta.

Jos vastapuoli osuu useasti samaan koordinaattiin, pitää aina ilmoittaa sama tulos kuin ensimmäisellä kerralla. Uponneiden laivojen lukumäärä ei saa tällaisessa tilanteessa kasvaa.

Kun aluksia uppoaa, laivaston omistajan tulee huolehtia, että vähentää ammuttavien ammusten määrää saamastaan viestistä, jossa lähetetään aina viisi koordinaattia.



Kuva 21. Pelaajat voivat lähettää viestejä toisilleen. Vastapuoli voi vastata joko OK, tai virheviestillä, jos ominaisuutta ei ole toteutettu.



Kuva 22. Pelin loppuminen.

Peli loppuu TRAP-viestillä (OID=6), jos toiselta tai molemmilta pelaajilta uppoavat laivat. Pelin voi myös lopettaa kesken kaiken keskeyttämällä. Tällöin pelin tilamuuttuja game.gameStatus = CANCELLED. Pelin loputtua tai keskeydyttyä pelaajat ilmoittavat pelituloksen omilla näytöillään.

3.5. Rajapinnat

Rajapintakuvaukset on luotu käyttäen Doxygen-dokumentointiohjelmaa [34]. Lähdekoodi on dokumentoitu Doxygenin vaatimusten mukaisesti ja rajapintakuvaukset on generoitu automaattisesti ja muokattu tämän dokumentin tyyliin sopiviksi. Tähän dokumenttiin on käytännön syistä otettu vain muutama rajapintakuvaus ja projektin täydellinen API-dokumentaatio löytyy internetistä osoitteesta http://www.ee.oulu.fi/~codec. Käyttäjätunnus on: sop ja salasana S0pp1 (toinen merkki on nolla ja viimeinen numero yksi).

3.5.1. Funktiot

Alla on esitetty esimerkinomaisesti puheentunnistukseen liittyviä funktiokuvauksia.

- void **initAudioDevice** (void)
  Funktio, joka suorittaa äänipiirin alustuksen.
- int **speechRecognizer** (int mfcc[][NMBROFFILTERS])
  Funktio, joka suorittaa puheentunnistuksen.
- void **recordSpeech** (short *inbuffer)
  Funktio, joka nauhoittaa signaalin mikrofonilta.
- void **speechDetector** (short *signaali, short *puheSignaali, int *puheenPituus)
  Funktio, joka erottaa puhesignaalin taustakohinasta.
- void **preEmphasis** (int *puheSignaali, int *suodPuheSignaali, u_short lkm)
  Funktio, joka suorittaa esisuodatuksen.
- void **frameBlocking** (int *filtered, int frames[][NAYTTEET], u_short nmbrOfFrames)
  Funktio, joka jakaa puhesignaalin kehyksiin.
- void **hammingWindowing** (int frameIn[][NAYTTEET], int frameOut[] [NAYTTEET], u_short nmbrOfFrames)
  Funktio, joka suorittaa Hamming-ikkunoinnin.
- void **melFiltering** (int signal[], int result[])
  Funktio, joka suorittaa 256-pisteen FFT:n tuloksen Mel-suodatin pankilla. Symmetrian vuoksi riittää ensimmäisen 128 FFT:n tuloksen käsittely.
- int **naturalLogarithm** (int input)
  Funktio, joka laskee luonnollisen logaritmin käyttäen hakutaulukkoa.
- void **discreteCosineTransform** (int input[], int output[])
  Funktio, joka suorittaa diskreetin kosini muunnoksen.
- int **dynamicTimeWarping** (int mfcc[][NMBROFFILTERS], int mfcc2[] [NMBROFFILTERS], int nmbrOfFrames1, int nmbrOfFrames2)

Funktio, joka suorittaa dynaamisen ajansovituksen.

void initAudioDevice (void)

Funktio, joka suorittaa äänipiirin alustuksen.

Palauttaa:

void.

int speechRecognizer (int *mfcc*[][NMBROFFILTERS])

Funktio, joka suorittaa puheentunnistuksen.

Parametrit:

***mfcc* Kaksiuloitteinen taulukko, johon talletetaan puhesignaalin MFCC-kertoimet.

Palauttaa:

nmbrOfFrames Palauttaa kehyksien lukumäärän puhesignaalissa.

void recordSpeech (short * *inbuffer*)

Funktio, joka nauhoittaa signaalin mikrofonilta.

Parametrit:

**inbuffer* Taulukko, johon näytteistetty signaali talletetaan.

Palauttaa:

void.

void speechDetector (short * *signaali*, short * *puheSignaali*, int * *puheenPituus*)

Funktio, joka erottaa puhesignaalin taustakohinasta.

Parametrit:

**signaali* Näytteistetty signaali.
**puheSignaali* Näytteistetystä signaalista erotettu puhesignaali.
**puheenPituus* Puhesignaalin näytteiden lukumäärä.

Palauttaa:

void.

void preEmphasis (int * *puheSignaali*, int * *suodPuheSignaali*, u_short *lkm*)

Funktio, joka suorittaa esisuodatuksen.

Parametrit:

**puheSignaali* Näytteistetystä signaalista erotettu puhesignaali.
**suodPuheSignaali* Suodatettu puheSignaali.

lkm Puhesignaalin näytteiden lukumäärä.

Palauttaa:

void.

void frameBlocking (int * *filtered*, int *frames*[][NAYTTEET], u_short *nmbrOfFrames*)

Funktio, joka jakaa puhesignaalin kehyksiin.

Parametrit:

**filtered* Suodatettu puhesignaali.
***frames* Kehykset.
nmbrOfFrames Kehyksien lukumäärä.

Palauttaa:

void.

void hammingWindowing (int *frameIn*[][NAYTTEET], int *frameOut*[][NAYTTEET], u_short *nmbrOfFrames*)

Funktio, joka suorittaa Hamming-ikkunoinnin.

Parametrit:

***frameIn* Ikkunoitava kehys.
***frameOut* Ikkunoitu kehys.
nmbrOfFrames Kehyksien lukumäärä.

Palauttaa:

void.

void melFiltering (int *signal*[], int *result*[])

Funktio, joka suorittaa 256-pisteen FFT:n tuloksen Mel-suodatin pankilla. Symmetrian vuoksi riittää ensimmäisen 128 FFT:n tuloksen käsittely.

Parametrit:

**signal* Suodatettava signaali.
**result* Suodatettu signaali.

Palauttaa:

void.

int naturalLogarithm (int *input*)

Funktio, joka laskee luonnollisen logaritmin käyttäen hakutaulukkoa.

Parametrit:

input Luku, josta otetaan logaritmi.

Palauttaa:

result Luonnollinen logaritmi.

void discreteCosineTransform (int *input*[], int *output*[])

Funktio, joka suorittaa diskreetin kosini muunnoksen.

Parametrit:

**input* input taulukko.
**output* output taulukko.

Palauttaa:

void.

int dynamicTimeWarping (int *mfcc*[][NMBROFFILTERS], int *mfcc2*[][NMBROFFILTERS], int *nmbrOfFrames1*, int *nmbrOfFrames2*)

Funktio, joka suorittaa dynaamisen ajansovituksen.

Parametrit:

***mfcc* Ensimmäisen sanan MFCC-kertoimet.
***mfcc2* Toisen sanan MFCC-kertoimet.
nmbrOfFrames1 Ensimmäisen sanan kehyksien lukumäärä.
nmbrOfFrames2 Toisen sanan kehyksien lukumäärä.

Palauttaa:

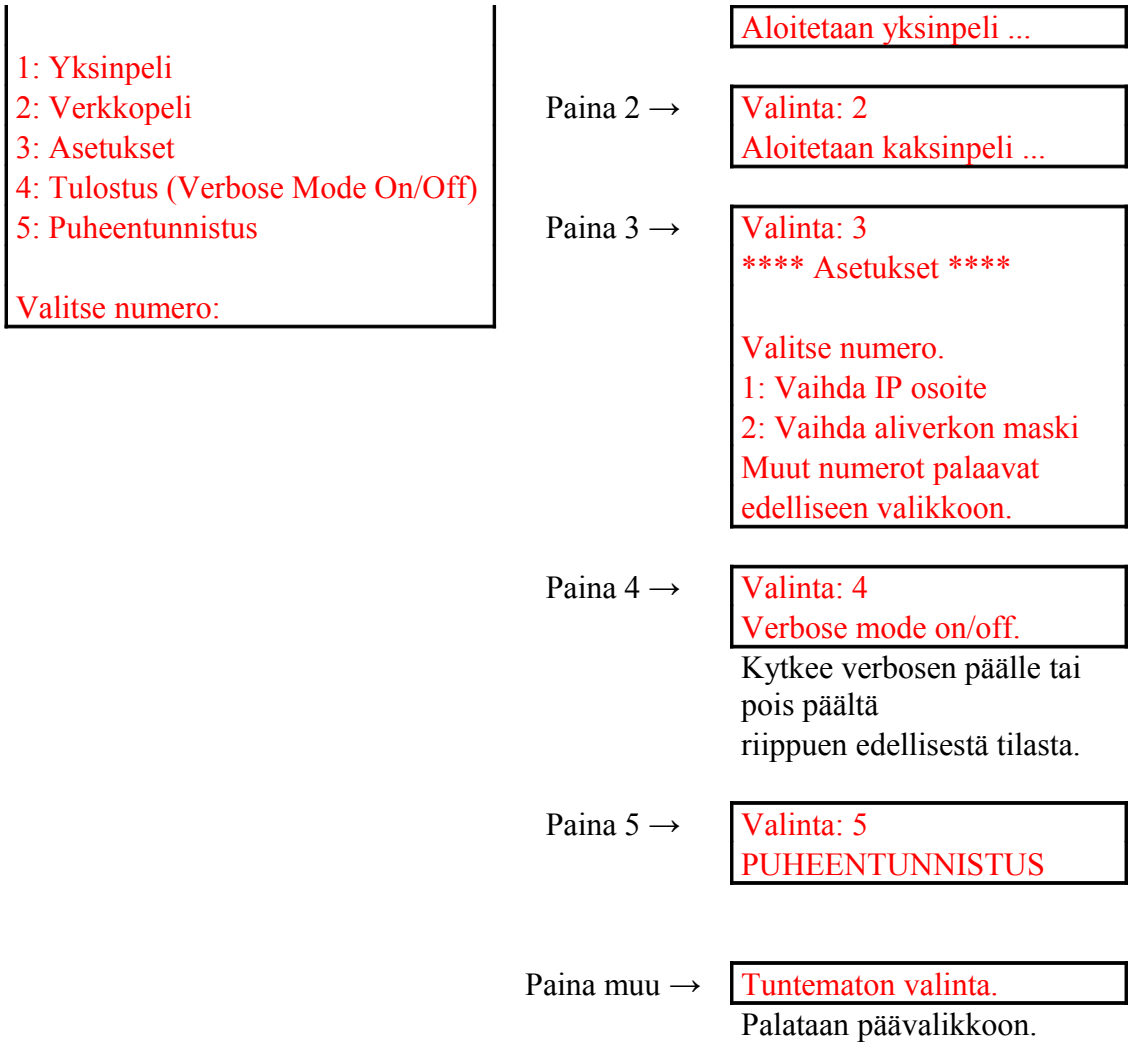
dtw[][] Yksi DTW-taulukon alkio.

3.6. Käyttöliittymäkuvaus

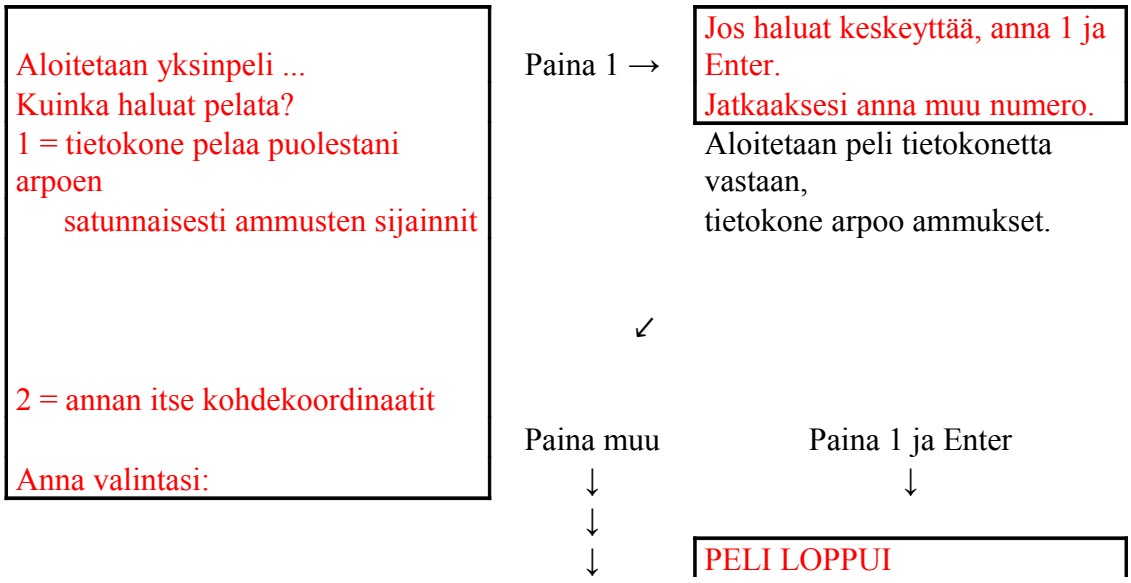
Järjestelmän käyttöliittymä on suunniteltu mahdollisimman selkeäksi ja yksiselitteiseksi, jolloin käyttäjä ei tarvitse erillistä ohjekirjaa. Järjestelmä tulostaa graafisen käyttöliittymän tietokoneen näytölle Tera Term-terminaaliohjelmaan ja sille voi sarjaportin kautta antaa tietokoneen näppäimistöllä syötteenä numeroita (0-9) ja kirjaimia (a-z). Lisäksi järjestelmä sisältää mikrofonin, jonka kautta voi antaa äänikomentoja. Järjestelmä tunnistaa komennot käyttäen puheentunnistusta ja muuttaa ne vastaaviksi käskyiksi. Vasteena järjestelmä voi tuottaa tekstiä ja erilaisia merkkejä tietokoneen näytölle terminaaliohjelman avulla.

Ohjelman käynnistyessä avautuu ensimmäisenä päävalikko eli ”Laivanupotuspeli”, jonka toiminta on kuvattu seuraavassa:

*** LAIVANUPOTUSPELI ***	Paina 1 →	Valinta: 1



Valitsemalla päävalikosta vaihtoehdon numero 1, käynnistetään yksinpeli, jonka toiminta on kuvattu seuraavassa:



↓ Tulokset: Peli keskeytettiin.
↓ Peli loppuu.
↓
→ PELI LOPPUI
Tulokset: ME VOITIMME!
Peli jatkuu kunnes se on pelattu loppuun.
Pelin voittaja ilmoitetaan.

Paina 2 → Anna yksi ammuskoordinaatti, arvot välillä 0-9: x y
Peli kysyy sinulta aina 5 ammus-
koordinaattia ja ampuu niihin.

Syötä 5 ammuskoordinaattia
↓

Jos haluat keskeyttää, anna 1 ja Enter.
Jatkaaksesi anna muu numero.

↙

Paina muu
↓
Peli kysyy joka kierroksella uudet ammus-koordinaatit
↓
→

Paina 1 ja Enter
↓
PELI LOPPUI
Tulokset: Peli keskeytettiin.
Peli loppuu.

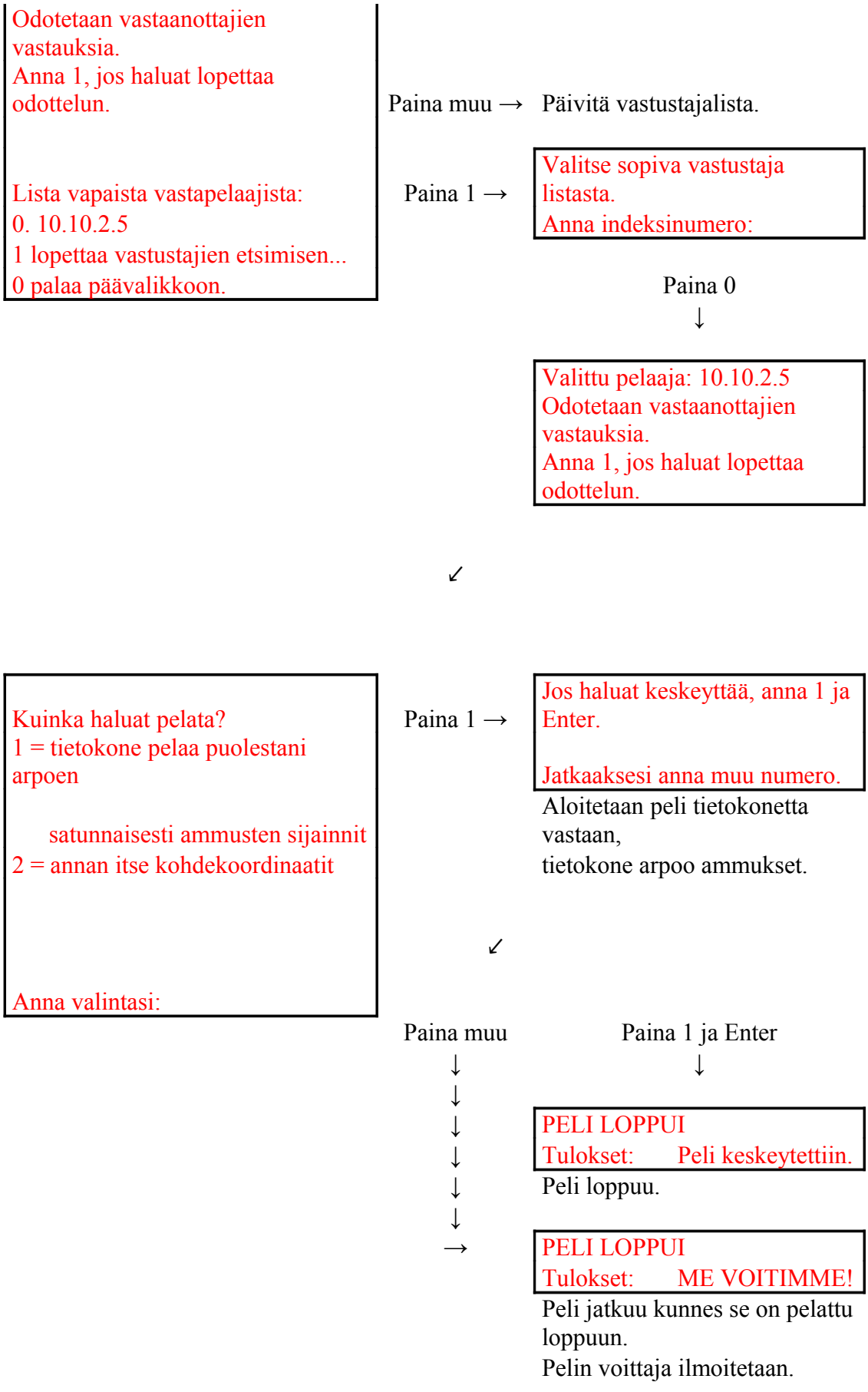
PELI LOPPUI
Tulokset: ME VOITIMME!
Peli jatkuu kunnes se on pelattu loppuun.
Pelin voittaja ilmoitetaan.

Paina muu → Sama komento kuin "Paina 2"

Valitsemalla päävalikosta vaihtoehdon numero 2, käynnistetään kaksinpeli, jonka toiminta on kuvattu seuraavassa:

Aloitetaan kaksinpeli ...

Paina 0 → Palataan päävalikkoon.


Odotetaan vastaanottajien vastauksia.
Anna 1, jos haluat lopettaa odottelun.
Lista vapaista vastapelaajista:
0. 10.10.2.5
1 lopettaa vastustajien etsimisen...
0 palaa päävalikkoon.
Paina muu →
Päivitä vastustajalista.
Paina 1 →
Valitse sopiva vastustaja listasta.
Anna indeksinumero:
Paina 0
Valittu pelaaja: 10.10.2.5
Odotetaan vastaanottajien vastauksia.
Anna 1, jos haluat lopettaa odottelun.
Kuinka haluat pelata?
1 = tietokone pelaa puolestani arpoen
satunnaisesti ammusten sijainnit
2 = annan itse kohdekoordinaatit
Anna valintasi:
Paina 1 →
Jos haluat keskeyttää, anna 1 ja Enter.
Jatkaaksesi anna muu numero.
Aloitetaan peli tietokonetta vastaan,
tietokone arpoo ammukset.
Paina muu
Paina 1 ja Enter
PELI LOPPUI
Tulokset: Peli keskeytettiin.
Peli loppuu.
PELI LOPPUI
Tulokset: ME VOITIMME!
Peli jatkuu kunnes se on pelattu loppuun.
Pelin voittaja ilmoitetaan.

Paina 2 → Anna yksi ammuskoordinaatti, arvot välillä 0-9: x y

Peli kysyy sinulta aina 5 ammus-
koordinaattia ja ampuu niihin.

Syötä 5 ammuskoordinaattia
↓

Jos haluat keskeyttää, anna 1 ja Enter.
Jatkaaksesi anna muu numero.

↙

Paina muu
↓
Peli kysyy joka kierroksella uudet ammus-koordinaatit
↓
→

Paina 1 ja Enter
↓

PELI LOPPUI
Tulokset: Peli keskeytettiin.
Peli loppuu.

PELI LOPPUI
Tulokset: ME VOITIMME!
Peli jatkuu kunnes se on pelattu loppuun.
Pelin voittaja ilmoitetaan.

Paina muu → Sama komento kuin "Paina 2"

Valitsemalla päävalikosta vaihtoehdon numero 3, käynnistetään Asetukset-valikko, jonka toiminta on kuvattu seuraavassa:

**** Asetukset ****

Valitse numero.
1: Vaihda IP osoite
2: Vaihda aliverkon maski
Muut numerot palaavat edelliseen valikkoon.

Paina 1 → ****IP valikko****
Anna uusi IP osoite

IP-valikko avautuu ja ohjelma pyytää
käyttäjää syöttämään uuden IP:n.

Syötä uusi IP

↓

Uusi IP on: "IP-osoite"
Ohjelma tulostaa uuden IP:n.

Paina 2 → **** Aliverkon peite ****
Valitse numero.
1:Aliverkon peite: 255.255.255.0
2:Aliverkon peite: 255.255.0.0
Muut numerot palaavat edelliseen valikkoon.

Paina muu → Tuntematon valinta.
Palataan päävalikkoon.

Asetukset-valikosta valitsemalla vaihtoehdon numero 2 pääsee Aliverkon peite-valikkoon, jonka toiminta on kuvattu seuraavassa:

**** Aliverkon peite ****
Valitse numero.
1:Aliverkon peite: 255.255.255.0
2:Aliverkon peite: 255.255.0.0
Muut numerot palaavat edelliseen valikkoon.

Paina 1 → Aliverkon peite: 255.255.255.0
Asetetaan aliverkon peite ja
Palataan päävalikkoon.

Paina 2 → Aliverkon peite: 255.255.0.0
Asetetaan aliverkon peite ja
Palataan päävalikkoon.

Paina muu → Tuntematon valinta.
Palataan päävalikkoon.

Valitsemalla päävalikosta vaihtoehdon numero 5, käynnistetään puheentunnistus, jonka avulla käyttäjä voi antaa järjestelmän päävalikossa olevat komennot puhumalla. Puheentunnistuksen toiminta on kuvattu seuraavassa:

PUHEENTUNNISTUS

Sanasto: Yksinpeli, Verkkopeli,
Asetukset, Tulostus
Paina jotain painiketta alottaaksesi.

Paina jokin → Recording sample 1
Mikäli järjestelmä tunnistaa jonkin
sanaston sanoista, se lähtee
suorittamaan kyseistä toimintoa.

Puheentunnistus epäonnistui.
Mikäli järjestelmä ei tunnista

komentoa,
se palaa päävalikkoon.

4. TESTIEN TULOKSET

4.1. Testaussuunnitelma

Projektin aikana testasimme seuraavia toimintoja:

1. Käyttöliittymä:
    - Käyttöliittymän toiminta.

2. Verkko-toiminnot:
    - Viestien vastaanottaminen kanavasta.
    - Viestien lähetys kanavaan.
    - Viesteihin vastaaminen.

3. Puheentunnistus:
    - Äänen tallentaminen laitteen muistiin.
    - Speech detection – lohkon testaaminen, puhesignaalin eristäminen.
    - Feature extraction – lohkon testaaminen, piirrevektorien laskeminen.
    - Pattern classifier – lohkon testaaminen, hahmontunnistus.

4. Laivanupotus:
    - Yksinpeli
        i. Laivojen satunnainen sijoittaminen merelle
        ii. Ammuskoordinaattien satunnainen arvonta
        iii. Ammuskoordinaattien syöttäminen
    - Kaksinpeli
        i. Toisen pelaajan kutsuminen peliin
        ii. Osallistuminen toisen pelaajan aloittamaan peliin
        iii. Kieltäytyminen toisen pelaajan aloittamasta pelistä
        iv. Laivojen satunnainen sijoittaminen merelle
        v. Ammuskoordinaattien satunnainen arvonta
        vi. Ammuskoordinaattien syöttäminen
        vii. Ammuskoordinaattien lähettäminen

Testaussuunnitelma on jaettu jokaisen testattavan komponentin osalta eri vaiheisiin. Kaikkia toimintoja ei testattu jokaisessa vaiheessa vaan testejä suoritettiin, kun komponentin valmistus alkoi tai siihen tehtiin muutoksia.

4.2. Testit

4.2.1. Käyttöliittymän testaus

Aluksi käyttöliittymän toimintaa testattiin lisäämällä testitulostuksia koodiin. Näin tiedettiin, että ohjelma lähtee suorittamaan toimintoa, jonka käyttäjä on valinnut käyttöliittymästä. Myöhemmässä vaiheessa nämä tulostukset korvattiin kyseisen toiminnon suorittavalla koodilla. Käyttöliittymän testaus on rajattu kattamaan käyttöliittymän päävalikko. Käyttöliittymän testaus vaiheessa 1 on esitetty taulukossa 1 ja niin edelleen.

Taulukko 1. Käyttöliittymän testaus vaiheessa 1

#	Testattavat komponentit	Aloitus-tila	Toimen-piteet	Testin kuvaus / hyväksymisehdot	Tulos	Kommentit
1	Käyttöliittymä	Päävalikko	Paina ’1’	Käynnistä yksinpeli	OK	
2	Käyttöliittymä	Päävalikko	Paina ’2’	Käynnistä kaksinpeli	OK	
3	Käyttöliittymä	Päävalikko	Paina ’3’	Avaa Asetukset	OK	
4	Käyttöliittymä	Päävalikko	Paina ’4’	Aseta Verbose päälle/pois päältä	OK	
5	Käyttöliittymä	Päävalikko	Paina ’5’	Käynnistä puheentunnistus	OK	
6	Käyttöliittymä	Päävalikko	Paina muu näppäin	Palaa päävalikkoon	OK	Funktio-näppäimet käynnistävät pelin

Taulukko 2. Käyttöliittymän testaus vaiheessa 2

#	Testattavat komponentit	Aloitus-tila	Toimen-piteet	Testin kuvaus / hyväksymisehdot	Tulos	Kommentit
1	Käyttöliittymä	Asetukset	Paina ’1’	Avaa IP-valikko	OK	
2	Käyttöliittymä	Asetukset	Paina ’2’	Avaa Subnet Mask-valikko	OK	
3	Käyttöliittymä	Asetukset	Paina muu näppäin	Palaa päävalikkoon	OK	Funktio-näppäimet käynnistävät pelin

Taulukko 3. Käyttöliittymän testaus vaiheessa 3

#	Testattavat komponentit	Aloitus-tila	Toimen-piteet	Testin kuvaus / hyväksymisehdot	Tulos	Kommentit
1	Käyttöliittymä	IP-valikko	Syötä uusi IP-osoite	Tulosta uusi IP-osoite	OK	
2	Käyttöliittymä	Subnet Mask-valikko	Paina ’1’	Aseta Subnet maskiksi 255.255.255.0	OK	
3	Käyttöliittymä	Subnet Mask-valikko	Paina ’2’	Aseta Subnet maskiksi 255.255.0.0	OK	

4	Käyttöliittymä	Subnet Mask-valikko	Paina muu näppäin	Palaa päävalikkoon	OK	Funktio-näppäimet käynnistävät pelin

4.2.2. Verkkotoimintojen testaus

Verkkotoimintojen testaus vaiheessa x on esitetty taulukossa x ja niin edelleen.

Taulukko 5. Verkkotoimintojen testaus vaiheessa 1.

#	Testattavat komponentit	Testausvaihe	Toimen-piteet	Testin kuvaus / hyväksymisehdot	Tulos	Kommentit
1	Verkko	1	Viestin lähettäminen broadcast-osoitteeseen	Yleismuotoisen viestin lähettäminen verkon yli / Yleismuotoinen viesti näkyy Wiresharkissa	OK	
2	Verkko	1	Viestin lähettäminen haluttuun IP-osoitteeseen	Yleismuotoisen viestin lähettäminen verkon yli / Yleismuotoinen viesti näkyy Wiresharkissa	OK	
3	Verkko	1	SNMP-viestin vastaanottaminen	SNMP-viestin vastaanottaminen verkosta / Viesti vastaanotettu	FAIL	Vastaanotettu viesti parsittiin virheellisesti
4	Verkko	1	SNMP-viestin vastaanottaminen	SNMP-viestin vastaanottaminen verkosta / Viesti vastaanotettu	OK	

Taulukko 6. Verkkotoimintojen testaus vaiheessa 2.

#	Testattavat komponentit	Testausvaihe	Toimen-piteet	Testin kuvaus / hyväksymisehdot	Tulos	Kommentit
1	Verkko	2	SNMP-viesteihin vastaaminen	SNMP-viesteihin vastaaminen lähettäjälle / Lähettäjä vastaanottaa vastaamamme viestin	FAIL	Lähetetyn SNMP-viestin muoto virheellinen
2	Verkko	2	SNMP-viesteihin vastaaminen	SNMP-viesteihin vastaaminen lähettäjälle / Lähettäjä vastaanottaa vastaamamme viestin	FAIL	Lähetetyn SNMP-viestin kenttien pituudet virheellisiä
3	Verkko	2	SNMP-viesteihin vastaaminen	SNMP-viesteihin vastaaminen lähettäjälle / Lähettäjä vastaanottaa vastaamamme viestin	OK	

Taulukko 7. Verkkotoimintojen testaus vaiheessa 3.

#	Testattavat komponentit	Testausvaihe	Toimen-piteet	Testin kuvaus / hyväksymisehdot	Tulos	Kommentit
1	Verkko	3	SNMP-viestin lähettäminen	SNMP-viestin lähettäminen verkon yli / Lähettämämme viesti näkyy oikeanlaisena Wiresharkissa	FAIL	Lähetetyn SNMP-viestin muoto virheellinen
2	Verkko	3	SNMP-viestin lähettäminen	SNMP-viestin lähettäminen verkon yli / Lähettämämme viesti näkyy oikeanlaisena Wiresharkissa	FAIL	Lähetetyn SNMP-viestin kenttien pituudet virheellisiä
3	Verkko	3	SNMP-viestin lähettäminen	SNMP-viestin lähettäminen verkon yli / Lähettämämme viesti näkyy oikeanlaisena Wiresharkissa	OK	

4.2.3. Laivanupotuspelin testaus

Laivanupotuspelin testaus vaiheessa x on esitetty taulukossa x ja niin edelleen.

Taulukko 9. Laivanupotuspelin toimintojen testaus vaiheessa 1

#	Testattavat komponentit	Testausvaihe	Toimen-piteet	Testin kuvaus / hyväksymisehdot	Tulos	Kommentit
1	Laivanupotus	1	Laivojen satunnainen sijoittaminen merelle	Yksinpeli, 5 laivaa sijoitetaan merelle, niin etteivät ne ole vierekkäin	FAIL	Vain 4 laivaa oli sijoitettu merelle
2	Laivanupotus	1	Laivojen satunnainen sijoittaminen merelle	Yksinpeli, 5 laivaa sijoitetaan merelle, niin etteivät ne ole vierekkäin	OK	Tarkistusmuutt ujalla katsotaan, että merellä on tarpeeksi suuri tila laivalle
3	Laivanupotus	1	Ammuskoordina attien satunnainen arvonta	Yksinpeli, Satunnaisesti arvotut ammuskoordinaatit ammutaan vastustajan merelle	OK	
4	Laivanupotus	1	Ammuskoordina attien syöttäminen	Yksinpeli, Manuaalisesti syötetyt ammuskoordinaatit ammutaan vastustajan merelle	OK	

Taulukko 10. Laivanupotuspelin toimintojen testaus vaiheessa 2

#	Testattavat komponentit	Testausvaihe	Toimen-piteet	Testin kuvaus / hyväksymisehdot	Tulos	Kommentit
1	Laivanupotus	2	Laivojen satunnainen sijoittaminen merelle	Kaksinpeli, 5 laivaa sijoitetaan merelle, niin etteivät ne ole vierekkäin	OK	
2	Laivanupotus	2	Ammuskoordinaattien satunnainen arvonta	Kaksinpeli, Ammuskoordinaattien satunnainen arvonta	OK	
3	Laivanupotus	2	Ammuskoordinaattien syöttäminen	Kaksinpeli, Ammuskoordinaattien manuaalinen syöttäminen	OK	
4	Laivanupotus	2	Ammuskoordinaattien lähettäminen	Kaksinpeli, Ammuskoordinaatit ammutaan vastustajan merelle onnistuneesti	OK	

Taulukko 11. Laivanupotuspelin toimintojen testaus vaiheessa 3

#	Testattavat komponentit	Testausvaihe	Toimen-piteet	Testin kuvaus / hyväksymisehdot	Tulos	Kommentit
1	Laivanupotus	3	Toisen pelaajan kutsuminen peliin	Kaksinpeli, Toinen pelaaja vastaanottaa kutsun osallistua peliin	FAIL	Viestit kirjoittautuvat toistensa päälle
2	Laivanupotus	3	Toisen pelaajan kutsuminen peliin	Kaksinpeli, Toinen pelaaja vastaanottaa kutsun osallistua peliin	OK	
3	Laivanupotus	3	Osallistuminen toisen pelaajan aloittamaan peliin	Kaksinpeli, Toisen pelaajan aloittamaan peliin liittyminen onnistuneesti	OK	
4	Laivanupotus	3	Kieltäytyminen toisen pelaajan aloittamasta pelistä	Kaksinpeli, Kieltäytyminen toisen pelaajan aloittamasta pelistä onnistuneesti	OK	

4.2.4. Puheentunnistuksen testaus

Puheentunnistuksessa käytettyjä lohkoja on testattu koko projektin ajan vertaamalla toteutuksemme antamia arvoja ja signaalimuotoja Matlab-ohjelmistolla saatuihin vertailuarvoihin ja –signaalimuotoihin. Esimerkkitapaukset signaaleista on esitetty liitteessä 2.

Puheen tunnistamisessa keskityttiin neljän sanan tunnistamiseen ja sanojen tunnistamisessa käytettävien referenssimallien valintaan. Liitteessä 1 esitetään kahden testitapauksen tulokset. Testisanan ja jokaisen referenssimallin välinen DTW-algoritmin antama pienimmän kustannusfunktion arvo on tulostettu testitapauksiin. Raportista käy ilmi tunnistettava sana ja sen tunnistanut referenssimalli.

5. SAAVUTUKSET JA POHDINTA

Puhesignaalin tunnistamista testattiin kahdella testitapauksella, joiden tulokset on esitetty liitteessä 1. Puhesignaalin testaukset kohdistuivat neljän tunnistettavan sanan valintaan ja niiden tunnistamisessa käytettävien referenssimallien valintaan. Testisanan ja jokaisen referenssimallin välinen DTW-algoritmin antama pienimmän kustannusfunktion arvo tulostettiin testiraporttiin (liite 1). Testiajon 2 tulokset on koottu yhteenvetona taulukkoon //xxx.

Taulukko xxx. Testisanojen tunnistus eri referenssimalleilla (testitapaus 2). Lihavoidut numerot kuvaavat oikein tunnistettuja sanoja kyseisellä referenssimallilla. Alleviivatut numerot kuvaavat virheellisiä tunnistuksia. Referenssimallit y1 ja y2 ovat sanan ”yksinpeli” referenssimalleja. Vastaavasti v1, v2 ja v3 tunnistavat sanaa ”verkkopeli”, a1-a5 sanaa ”asetukset”, m1 ja m2 sanaa ”moninpeli” ja t1 ja t2 sanaa ”tulostus”.

Testisanat		Näytteet													
	Lkm	y1	y2	v1	v2	v3	a1	a2	a3	a4	a5	m1	m2	t1	t2
“Yksinpeli”	10	**9**	**1**												
“Verkkopeli”	10			**9**	**0**	**0**					1				
“Asetukset”	10						**1**	**0**	**0**	**0**	**9**				
”Moninpeli”	10				2							**2**	**6**		
”Tulostus”	12				1								2	**2**	**7**
Poistetut sanat												-	-		
Valitut referenssi-mallit		**x**	**x**	**x**	**-**	**-**	**x**	**-**	**-**	**-**	**x**	**-**	**-**	**-**	**x**

Sana ”yksinpeli” on tunnistettu oikein jokaisella testikerralla. Referenssimalli y1 on tunnistanut ”yksinpeli”-sanan yhdeksän kertaa ja referenssimalli y2 kerran.

Rererenssimalleilla v2 ja v3 ei ole tunnistettu yhtään sanaa oikein, referenssimalli v1 voidaan valita ainoaksi referenssimalliksi testatuista malleista. Referenssimalli v2 antoi kolme virheellistä tunnistusta, yhden sanalle ”tulostus” ja kaksi sanalle ”moninpeli”.

Sanan ”asetukset” tunnistamisessa oli edellisillä testikierroksilla (testitapaus 1 ja aiemmat testit) ongelmia, mikä selittää runsasta referenssimallien määrää. Ennen testitapausta 2 lisätty referenssimalli a5 tunnistaakin ”asetukset”-sanaa kiitettävästi (9/10 sanasta). Rererenssimalli a1 tunnistaa sanan kerran.

Testisana ”moninpeli” oli vaihtoehtoinen sanalle ”verkkopeli”. Sanan ”verkkopeli” tunnistus onnistui paremmin kuin sanan ”moninpeli”. Sanan ”verkkopeli” tunnistuksessa tapahtui yksi virheellinen tunnistus. Sanan ”moninpeli” tunnistuksessa virheitä tuli kaksi. ”Moninpeli”-sanan referenssimalli m2 tunnisti lisäksi virheellisesti kaksi ”tulostus”-sanaa.

Sanan ”verkkopeli” tunnistukseen käytetty referenssimalli v2 ei tunnistanut yhtään ”verkkopeli”-sanaa, mutta aiheutti kolme virheellistä tunnistusta: kaksi ”moninpeli”-sanaa ja yhden ”tulostus”-sanan.

”Tulostus”-sanan tunnistamisen tarkkuus olisi parantunut yhden tunnistuksen osalta poistamalla ”moninpeli”-sanan referenssimalli m2. Tässä tapauksessa sanan oma t2 referenssimalli olisi antanut toiseksi pienimmän kustannusfunktion arvon.

”Tulostus”-sanan oikein tunnistetuissa tapauksissa referenssimalli t2 olisi antanut t1 jälkeen seuraavaksi pienimmän tuloksen. Täten referenssimalli t2 voidaan valita ainoaksi malliksi tunnistamaan sanaa ”tulostus”.

Testitapauksessa 2 referenssimalleja testattiin 52 puhenäytteellä. Referenssimallit olivat yhden puhujan puheeseen perustuvia, ja niitä testattiin saman puhujan puhenäytteillä. Puheentunnistusjärjestelmä tunnisti oikein 88% sanoista (46/52). Kuusi näytettä sai pienemmän DTW-kustannusfunktion arvon toisen sanan referenssimallilla (12 %). Sana ”verkkopeli” tunnistettiin paremmin kuin ”moninpeli”, ja se valittiin tunnistettavien sanojen joukkoon.

Tuloksia analysoitaessa voidaan todeta, että referenssimallien lisääminen ei välttämättä johda parempaan lopputulokseen. Runsas referenssimallien määrä voi johtaa myös tunnistettavien sanojen harhautumiseen muiden sanojen referenssimalleihin. Näin voi käydä, jos esim. sanan alku- tai loppukohdan erottaminen ei onnistu nauhoitetusta äänisignaalista. Mahdollisimman osuvien referenssimallien valinnalla voidaan nopeuttaa laskentaa.

Testitapauksen 2 tulokset olisivat olleet paremmat, jos virheellisiä tuloksia aiheuttaneet referenssimallit olisi poistettu vaihtoehtojen joukosta. Taulukon //xxx alimmalle riville on merkitty x-merkilllä käyttöön valitut referenssimallit, joita käyttäen testitapaus 2 olisi tuottanut vain yhden väärän tunnistuksen (”verkkopeli”-sanan tunnistaminen a5-referenssimallilla). Tässä tapauksessa oikein tunnistettujen sanojen osuus olisi noussut 98 %. Lisäksi sanan tunnistusaika olisi nopeutunut alle puoleen (6/14) nyt kuluneesta ajasta, kun läpilaskettavien referenssimallien lukumäärä olisi vähentynyt.

Taulukossa //xxx valituksi merkityt referenssimallit otettiin käyttöön, ja tämän jälkeen testattiin puheentunnistusta usean puhujan sanoilla. Tulokset tästä on esitetty taulukossa //x+1. Virpin sanat tunnistettiin 100 % tarkkuudella (4/4). Oskarilla oikein tunnistettujen sanojen osuus oli 80 % (4/5). Vain yksi Oskarin sana tulkittiin väärin (20 %). Kahden puhujan yhteenlaskettujen oikein tunnistettujen sanojen osuus oli 89 % kaikista sanoista (8/9). Tätä voidaan pitää hyvänä tuloksena. Järjestelmä oli opetettu vain toisen puhujan (Virpi) puheella.

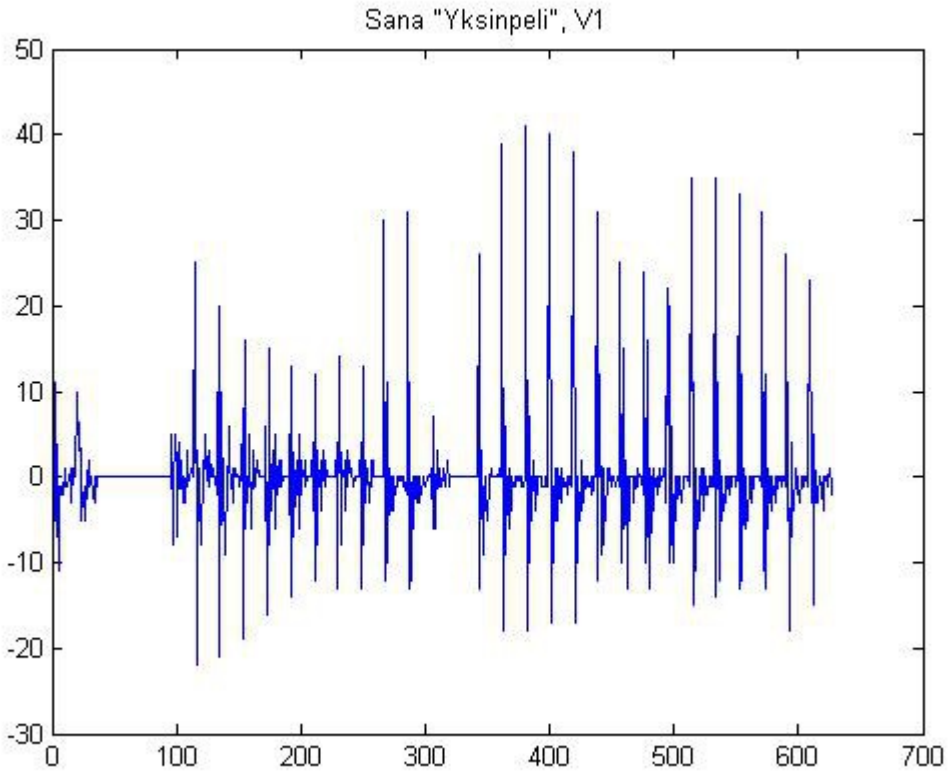
Taulukko //x+1. Testitapaus 3. Oikein tunnistetut sanat on merkitty lihavoinnilla. Joukossa oli ainoastaan yksi virheellinen tulos (alleviivattu).

Testisanat		Näytteet			
	Lkm	“Yksinpeli”	“Verkkopeli”	“Asetukset”	“Tulostus”
Virpi:					
“Yksinpeli”	1	**1**			
“Verkkopeli”	1		**1**		
“Asetukset”	1			**1**	
”Tulostus”	1				**1**
Oskari:					
“Yksinpeli”	1	**1**			
“Verkkopeli”	2	1	**1**		
“Asetukset”	1			**1**	
”Tulostus”	1				**1**

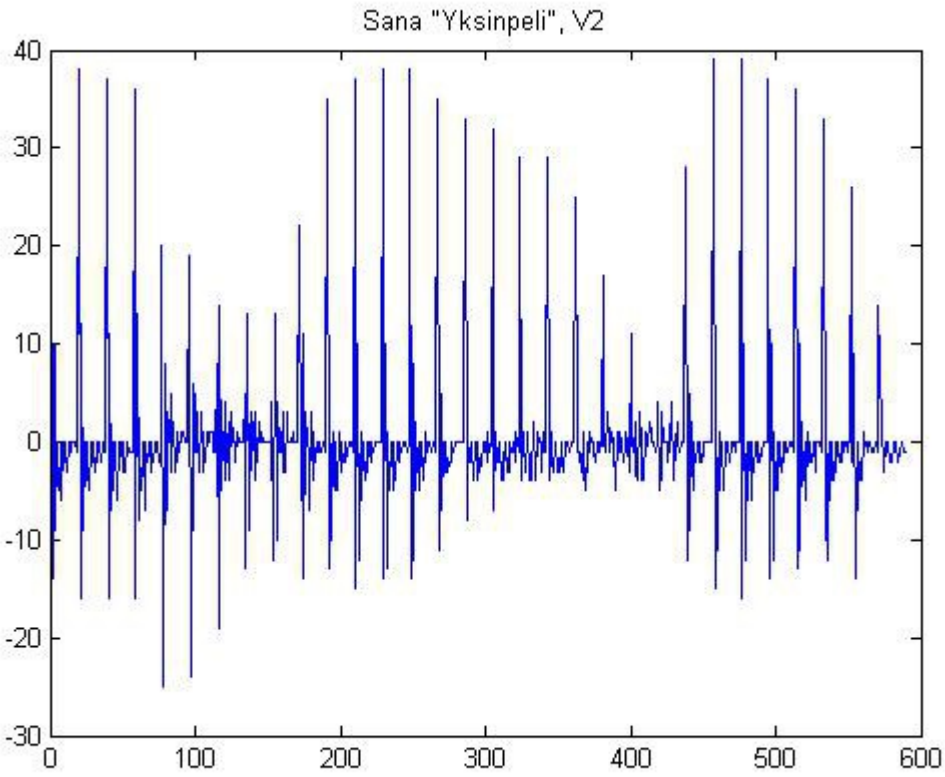
5.1. Referenssimallit

Kuvissa x1-x7 esitetään esimerkkejä referenssimalleista. Kuvissa x1 ja x2 esitetään sanan ”yksinpeli” referenssimallit y1 ja y2. Kuvissa x3 ja x4 on sanan ”verkkopeli” kaksi referenssimallia v1 ja v2. v2 ei tunnistanut yhtään ”verkkopeli”-sanaa. Sen sijaan referenssimalli v2 muistuttaa ”moninpeli”-sanan referenssimallia m2. Yhdenkaltaisuus selittäisi kaksi virheellistä ”moninpeli”-sanan tunnistusta. Kuvassa x5 on ”asetukset”-sanan referenssimalli, joka poikkeaa muista malleista. Kuvissa x6 ja x7 on sanojen ”moninpeli” ja ”tulostus” referenssimallit.

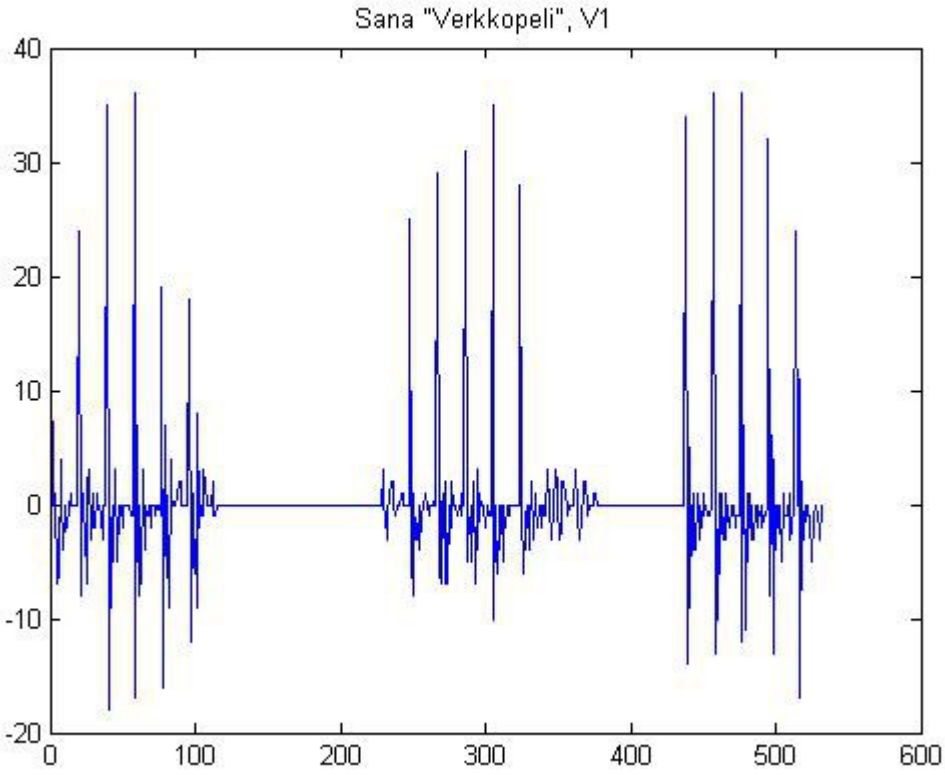
Referenssimalleista on nähtävissä, että soinnittomat äänteet rytmittävät voimakkaasti mallin muotoa.



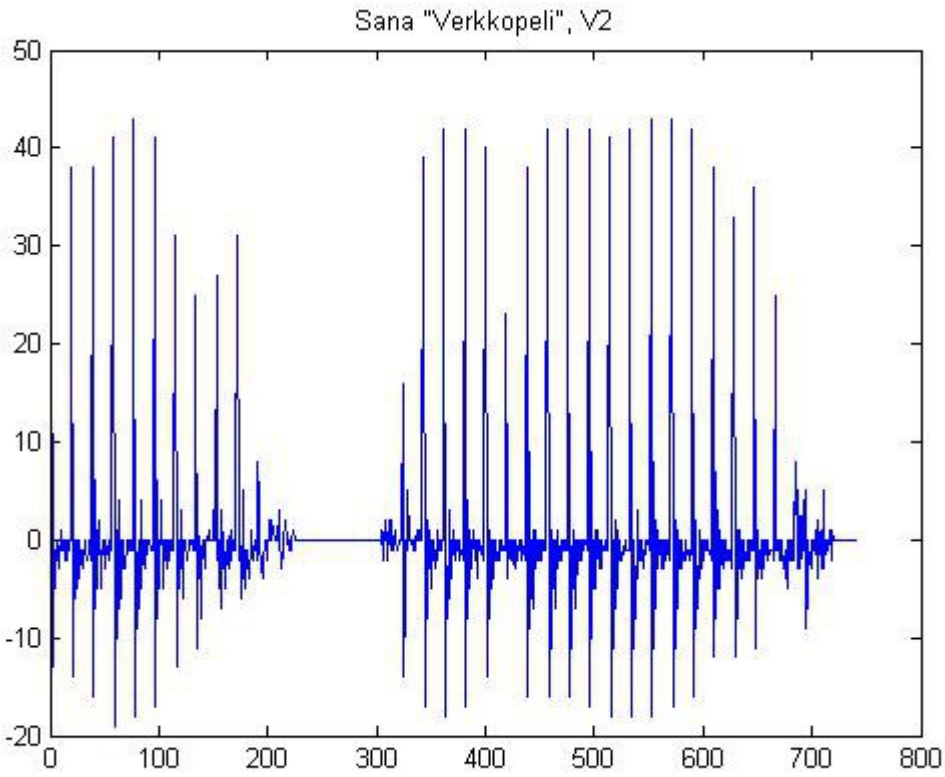
Kuva xxx. Sanan ”yksinpeli” referenssimalli y1.



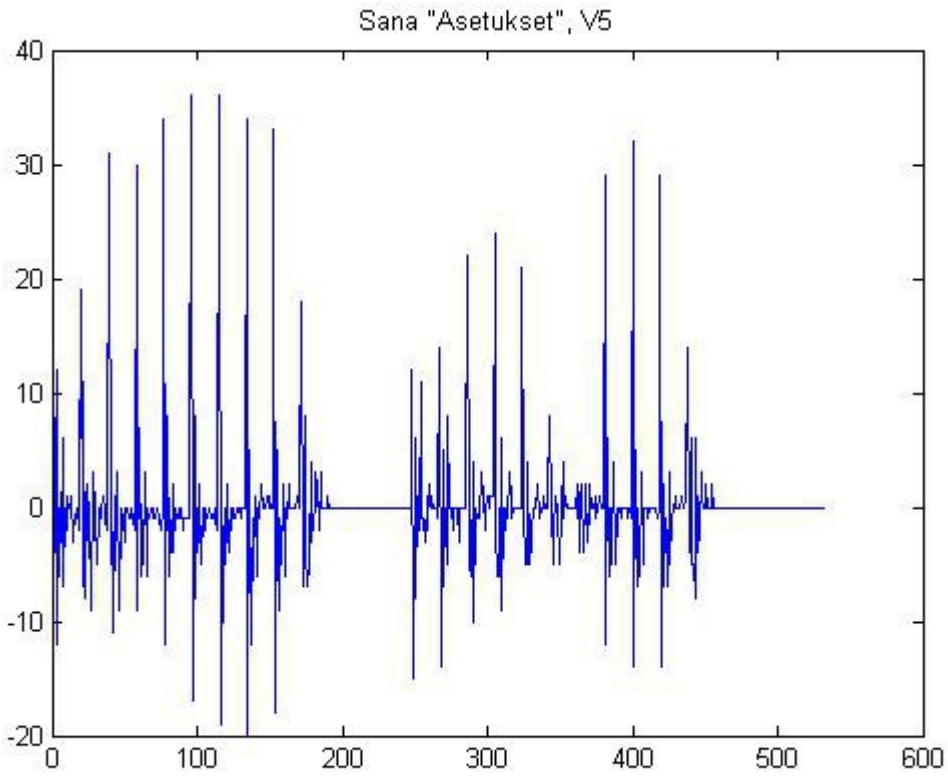
Kuva xxx. Sanan ”yksinpeli” referenssimalli y2.



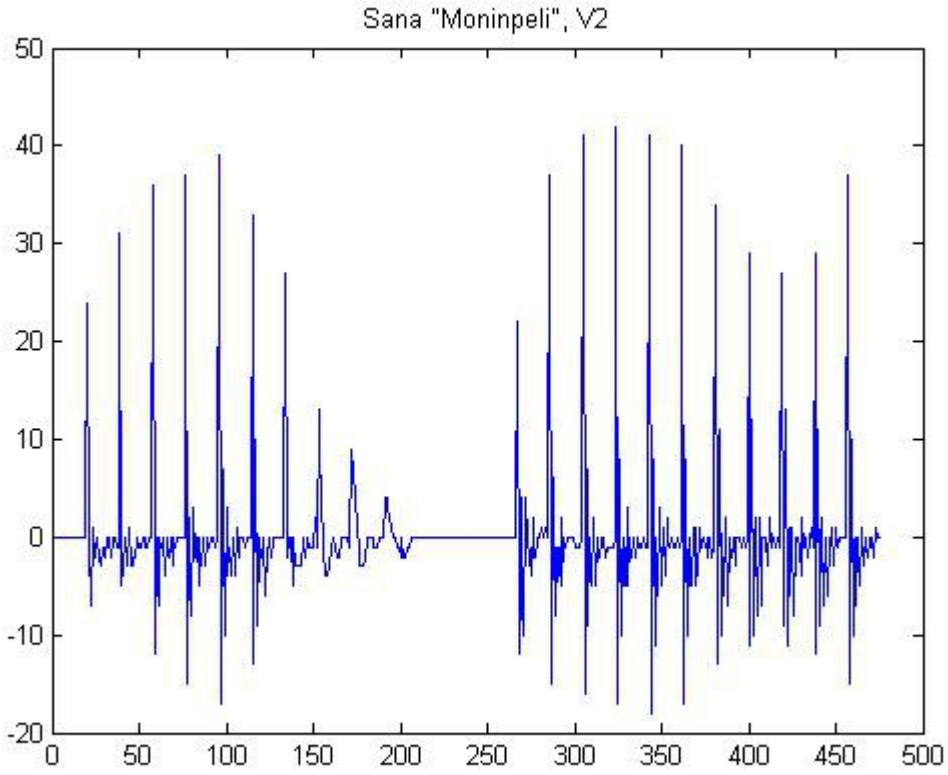
Kuva xxx. Sanan ”verkkopeli” referenssimalli v1.



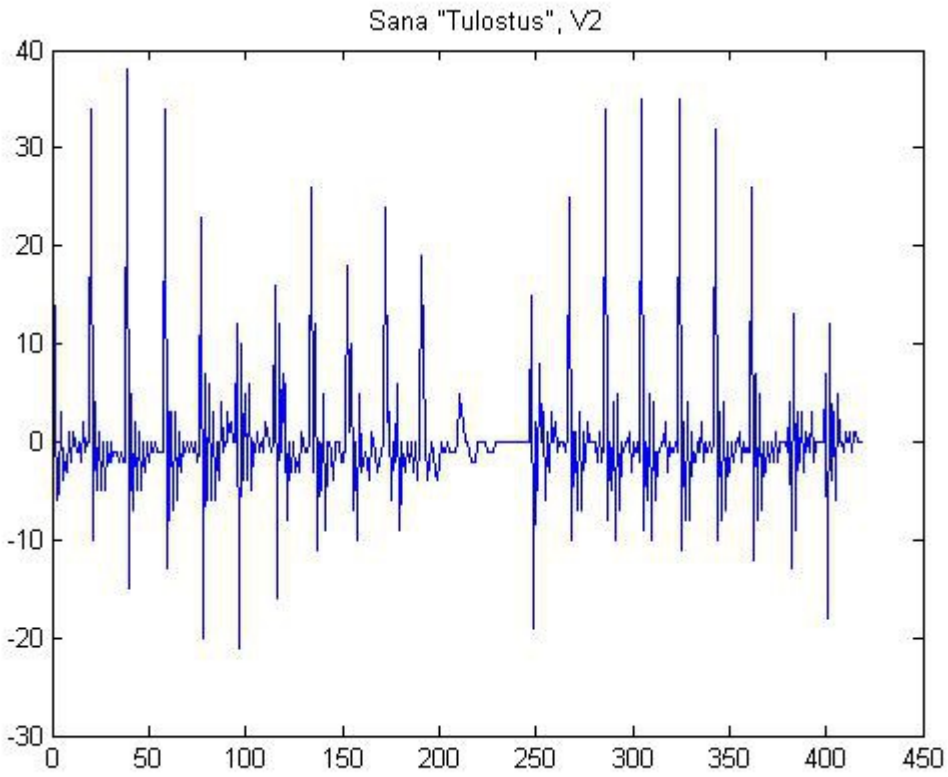
Kuva xxx. Sanan ”verkkopeli” referenssimalli v2. Referenssimalli v2 ei tunnistanut ”verkkopeliä”, sen sijaan virheellisesti sanat ”moninpeli” kaksi kertaa ja ”tulostus” kerran.



Kuva xxx. Sanan ”asetukset” referenssimalli a5.



Kuva xxx. Sanan ”moninpeli” referenssimalli m2.



Kuva xxx. Sanan ”tulostus” referenssimalli t2.

6. PROJEKTIN KUVAUS

6.1. Työnjako

Projekti aloitettiin tutustumalla annettuun tehtävään ja suorittamalla tiedonhakua puheentunnistukseen liittyen. Kaikki tekivät kirjallisuuskatsausta aiheesta. Markus kirjoitti yleistä puheentunnistuksesta, Virpi puheentunnistuksen historiaa ja Oskari valmisteli dokumenttipohjaa ja pienempiä kappaleita.

Milestone 1 jälkeen Virpi otti hoitaakseen dokumentin vaatimat muutokset. Oskari ja Markus aloittivat verkkotoiminnan koodauksen. Markus ja Virpi käyttivät aikaa tiedonhankintaan. Markus perehtyi tarkemmin puheentunnistusjärjestelmän rakenteeseen. LPC-koodauksesta luovuttiin tiedonhaun tulosten pohjalta, ja päädyttiin tekemään puheentunnistus MFCC-menetelmällä. Virpi muokkasi Excel-taulukoita siten, että työvaiheittainen ajankäyttö voitiin tulostaa graafisesti. Markus piirsi SA/SD-kaaviot ja Virpi sekvenssidiagrammit. Oskari suunnitteli käyttöliittymäkuvauksen ja testitapaukset.

Milestone 2 jälkeen Virpi koodasi laivanupotuspelin. Markus ja Oskari toteuttivat SNMP-protokollan mukaisten viestien lähetyksen ja vastaanoton. Virpi suunnitteli kaksinpelin verkkotoimintalogiikan. Virpi ja Markus integroivat kaksinpelin ja verkkopuolen toteutuksen yhteen ja samalla debuggasivat ja testasivat kaksinpelin toimintaa. Markus teki funktiot ledien käyttöä varten ja tutustui näppäimistön toimintaan. Virpi otti ledit käyttöön pelissä. Virpi piirsi pääohjelman tilakaavion ja päivitti ajankäytön kuvaajat. Oskari perehtyi äänen tallentamiseen käyttäen ensin apuna Matlab- ja Simulink-ohjelmistoja. Oskari teki Milestone 2 saadun palautteen mukaiset muutokset dokumenttiin ja työsti testaus-kappaletta.

Millestone 3 jälkeen aloitettiin puheentunnistusjärjestelmän koodaaminen. Markus koodasi FFT, logaritmin ja DCT-muunnoksen. Virpi koodasi DTW-algoritmin ja Oskari teki Matlab-koodin, joka muodosti Mel-suodattimien kertoimet. Debuggaukseen osallistuivat kaikki. Oman koodin toimintaa arvioitiin vertaamalla saatuja tuloksia Matlab-ohjelmiston antamiin tuloksiin. Järjestelmä saatettiin toimimaan Virpin puhetta tunnistaen, Virpi valitsi ja testasi järjestelmän opetusnäytteet sekä kirjoitti niiden pohjalta dokumentin tulokset. Markus piirsi SA/SD-kaaviot. Koodin ja dokumentin viimeistelyyn osallistui koko ryhmä.

6.2. Projektin erityispiirteet

Koska kaikki projektiin osallistuneet asuivat eri paikkakunnilla, asetti se oman haasteensa työskentelytapoihin. Projektipalaverit pidettiin Skype-puhelinneuvotteluina, ja tietoa vaihdettiin sähköpostitse.

Tiedonhaku oli ongelmallista ennen yksityiskohtaisia neuvoja Nellin kautta onnistuvasta IEEE Xplore -tietokannan ja sen Advanced -haun käytöstä.

Virheiden välttämiseksi viittausten käsittelyyn sovellettiin omaa menetelmää, jossa tekijän nimestä, julkaisuvuodesta ja otsikon alusta muodostettiin lyhenne [Ju00SpLa], joka korvattiin vasta lopullisessa versiossa numeerisella viitteellä.

Koodikatselmoinnit havaittiin parhaiksi tavoiksi virheiden paikallistamisessa.

6.3. Ajankäyttö

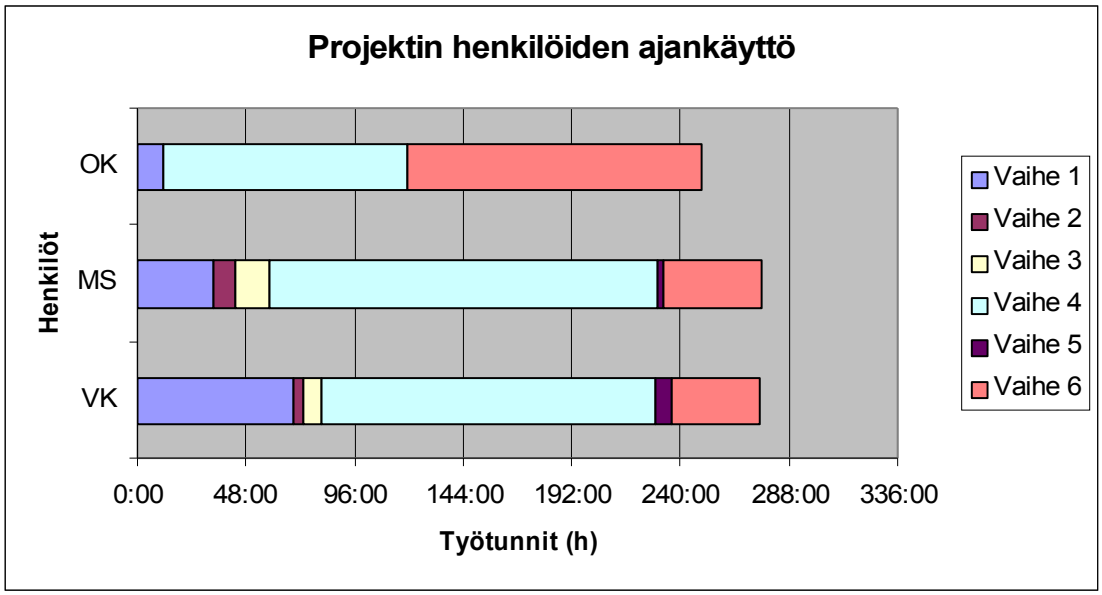
Ryhmän jäsenet pitivät kirjaa projektiin käyttämästään työskentelyajasta (taulukko 1x). Oskari käytti projektiin aikaa 249 tuntia, Virpi 275 tuntia ja Markus 276 tuntia. Kaikki käyttivät aikaa suunnitteluun, toteutukseen ja dokumentointiin. Työtunnit on jaoteltu ajallisesti suhteessa välietappeihin ja myös työvaiheisiin suhteutettuna.

Työvaiheet olivat (1) yleinen suunnittelu, (2) korkean tason suunnittelu, (3) yksityiskohtainen suunnittelu, (4) koodaus, debuggaus ja koodikatselmukset, (5) testaus ja (6) dokumentointi.

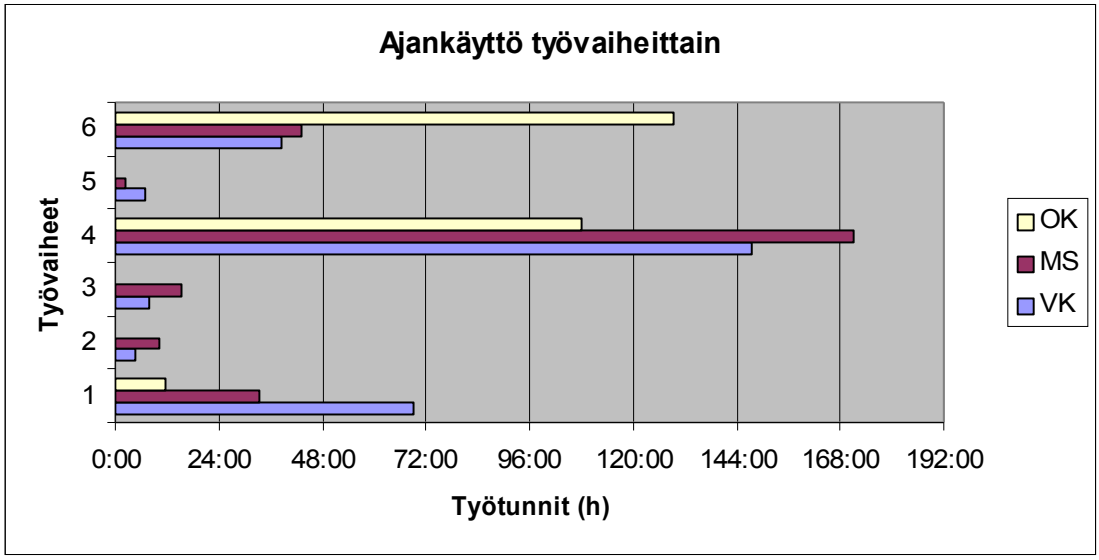
Taulukko xx. Projektin jäsenten ajankäyttö projektin etenemisen mukaisesti työvaiheisiin jaettuna (Lyhenteet: OK = Oskari, MS = Markus ja VK = Virpi)

Ajanjakso	Vaihe 1	Vaihe 2	Vaihe 3	Vaihe 4	Vaihe 5	Vaihe 6	Väli-summa	Yhteensä
Milestone 1 (VK)	9:45	0:00	0:00	0:00	0:00	6:00	15:45	
Milestone 1-2 (VK)	36:45	4:30	4:00	0:00	0:00	14:20	59:35	
Milestone 2-3 (VK)	12:45	0:00	4:00	109:45	0:30	3:30	130:30	
Milestone 3-End (VK)	10:00	0:00	0:00	37:45	6:30	14:45	69:00	274:50
Milestone 1 (MS)	4:00	0:00	0:00	0:00	0:00	10:45	14:45	
Milestone 1-2 (MS)	28:15	8:00	0:00	11:45	0:00	13:30	61:30	
Milestone 2-3 (MS)	0:00	2:00	5:30	105:15	2:30	0:00	115:15	
Milestone 3-End (MS)	1:00	0:00	10:00	54:00	0:00	19:00	84:00	275:30
Milestone 1 (OK)	5:30	0:00	0:00	0:00	0:00	16:15	21:45	
Milestone 1-2 (OK)	1:30	0:00	0:00	11:30	0:00	13:45	26:45	
Milestone 2-3 (OK)	2:30	0:00	0:00	41:00	0:00	69:00	112:30	
Milestone 3-End (OK)	2:00	0:00	0:00	55:30	0:00	30:30	88:00	249:00

Kuvissa x2 ja x3 kuvataan koko projektin ajankäyttöä. Kuvasta x2 käy ilmi ryhmän jäsenten työajan jakautuminen eri työvaiheiden kesken. Kuvassa x3 on esitetty projektin ajankäyttö työvaiheittain.

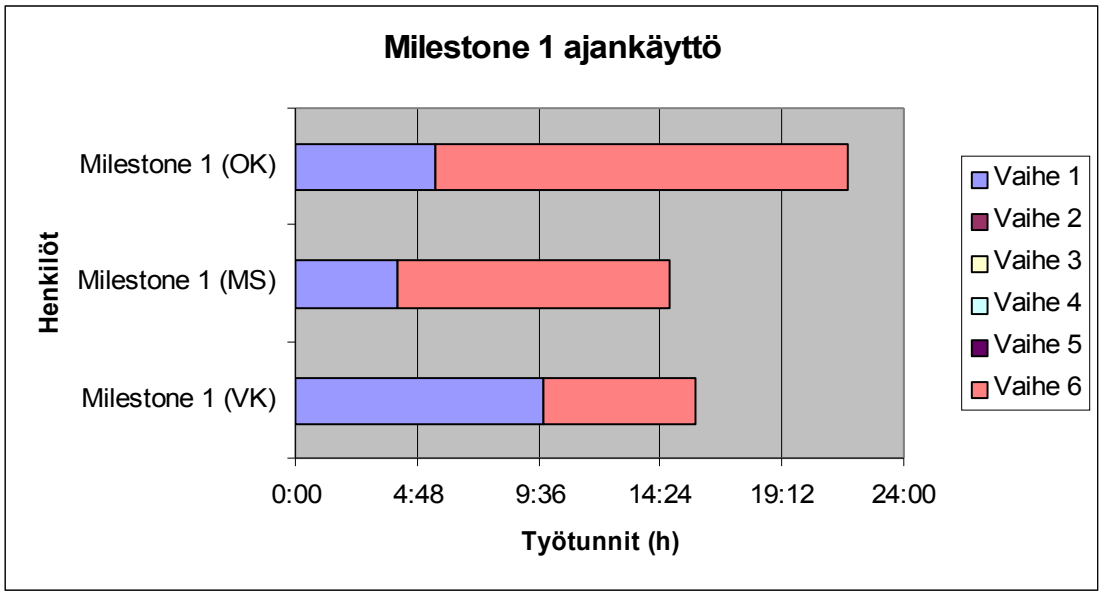


Kuva x2. Ryhmän jäsenten ajankäytön jakautuminen eri työvaiheisiin. Työvaiheet ovat (1) yleinen suunnittelu, (2) korkean tason suunnittelu, (3) yksityiskohtainen suunnittelu, (4) koodaus, debuggaus ja koodikatselmukset, (5) testaus ja (6) dokumentointi. Lyhenteet: OK = Oskari, MS = Markus ja VK = Virpi.

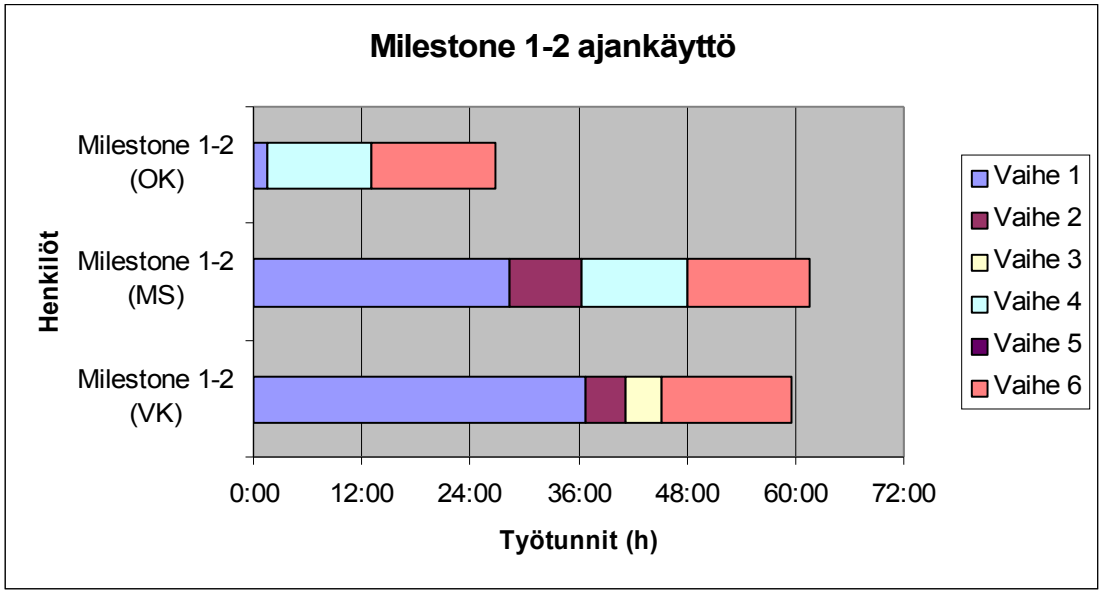


Kuva x3. Projektin ajankäyttö työvaiheittain. Lyhenteet samat kuin kuvassa //x2.

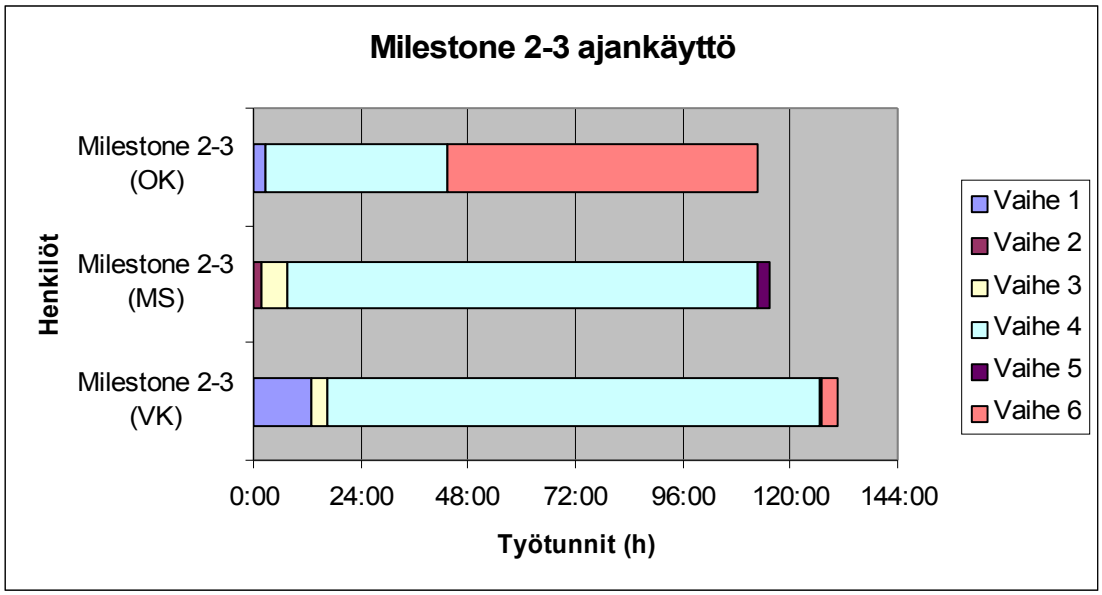
Kuvista x4 - x5 käy ilmi jokaisen ryhmän jäsenen työn määrä välietappien mukaisesti jaoteltuna.



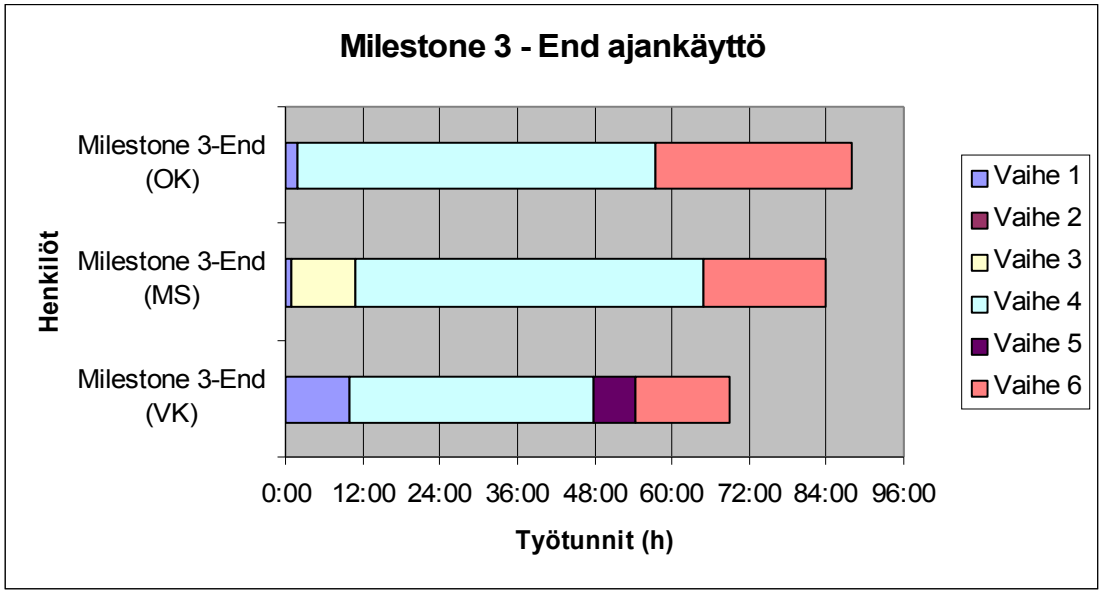
Kuva x4. Ensimmäiseen dokumenttipalautukseen (milestone 1) mennessä käytetty aika ja sen jakautuminen eri työvaiheisiin. Lyhenteet samat kuin kuvassa //x2.



Kuva x5. Ensimmäisen ja toisen dokumenttipalautuksen (milestone 2) välissä käytetty aika ja sen jakautuminen eri työvaiheisiin. Lyhenteet samat kuin kuvassa //x2.



Kuva x6. Toisen ja kolmannen dokumenttipalautuksen (milestone 3) välissä käytetty aika ja sen jakautuminen eri työvaiheisiin. Lyhenteet samat kuin kuvassa //x2.



Kuva x6. Kolmannen dokumenttipalautuksen (milestone 3) jälkeen käytetty aika ja sen jakautuminen eri työvaiheisiin. Lyhenteet samat kuin kuvassa //x2.

7. TULEVA KEHITYS

Nykyisellään puheentunnistusta on hyödynnetty pelin päävalikossa. Tunnistettavien sanojen määrää voisi kasvattaa kattamaan numerot (0-9) sekä kirjaimet (a-z) ja puheentunnistuksen lisätä vaihtoehtoiseksi syötteenantotavaksi itse laivanupotuspeliin. Sanamäärän kasvattaminen vaatisi puheentunnistusmekanismin tarkentamista. Puheen havaitsemisfunktion parametreja voisi säätää, jolloin puhesignaali saataisiin tarkemmin eroteltua kohinasta niin hiljaisessa kuin meluisassakin ympäristössä. Vielä pidemmälle vietynä toteutus voisi tutkia ympäristön melutasoa ja säätää havaitsemisfunktion parametreja automaattisesti sen mukaan. Myös Mel-taajuuskepstrikertoimien lukumäärää voisi lisätä nykyisestä 19:stä. Tällä hetkellä toteutus käyttää 12:ta diskreetin kosinimuunnoksen tuottaman piirrevektorin alkiota. Tämä ei välttämättä ole optimaalisin määrä ja muilla alkioiden lukumäärillä voitaisiin saavuttaa jopa tarkempia tunnistamistuloksia. Tunnistamisprosenttia voitaisiin kasvattaa myös tallentamalla ja testaamalla eri lukumääriä tunnistettavien sanojen mallisanoista. Tämänhetkinen toteutus käyttää 1-2 mallisanaa jokaisesta tunnistettavasta sanasta. Myös usean käyttäjän puheen tunnistaminen olisi mahdollista toteuttaa lisäämällä kaikkien ryhmän jäsenten tallentamat mallisanat toteutukseen.

Puheen tunnistaminen on toteutettu siten, että käyttäjä painaa ennalta määriteltyä näppäintä ja lausuu haluamansa komennon. Järjestelmä tallentaa ja käsittelee signaalin ja suorittaa syötettyä komentoa vastaavan toiminnon. Puheen tallentamisen voisi toteuttaa myös jatkuva-aikaisena toimintona, jolloin järjestelmä tallentaisi ääntä jatkuvasti ja havaittuaan puhesignaalin kohinan seasta suorittaisi signaalin käsittelyn ja syötettyä komentoa vastaavan toiminnon.

DTW-funktion palauttamalle kustannuspolun arvolle voitaisiin asettaa raja-arvo, jota suuremmat arvot hylättäisiin tunnistamattomina sanoina.

Saman sanan tunnistamiseen käytettyjen referenssimallien polun arvoja voitaisiin myös käyttää keskiarvoistamalla sanan tunnistukseen.

8. YHTEENVETO

Kandidaatintyön tehtävänanto oli laaja ja haasteellinen. Sulautettuun järjestelmään rakennettiin toimintaympäristöksi laivanupotuspeli. Peliin yhdistettiin SMNP-verkkotoiminnallisuus ja automaattinen puheentunnistus.

Verkkotoiminnallisuuden ja pelin yhdistäminen oli yllättävän työläs työvaihe. Määritellyt kättelyt eivät olleet riittäviä pitämään peliä samassa vaiheessa kahden pelaajan pelitilanteessa. Peliin piti lisätä useita tilamuuttujia, jotta kummankaan pelaajan peli ei etenisi ohi toisen pelitilanteesta. Emme pystyneet pelaamaan kaksinpeliä yhdenkään ulkopuolisen toteutuksen kanssa. Aina syynä oli pelitilanteen erivaiheisuus. Oman sovelluksemme kanssa kaksinpeli sujui loppuun saakka.

Tilamuuttujia ylläpidettiin sekä pelin että verkon puolella, kun seurattiin vastaanotettujen ja lähetettyjen viestien liikennettä. Oman lisähaasteensa toi SNMP-protokollan viestien taipumus kadota verkkoliikenteen noustessa riittävän suureksi.

Puheentunnistus tuntui ensin mahdottomalta tehtävältä. Sopivan algoritmin löydyttyä ja toteutuksen edetessä innostus oli tarttuvaa, ja ensimmäisen tunnistetun sanan jälkeen hymyä oli vaikea piilottaa. Signaalinkäsittelyn avulla puhesignaalia käsiteltiin niin, että tietosisältö säilyi, mutta varastointiin vaadittava tila pieneni.

Puhesignaalin jaettiin lyhyisiin 256 näytettä sisältäviin aikaikkunoihin ja käsiteltiin ylipäästösuodattimella, joka tasoitti signaalin korkeita piikkejä säilyttäen silti signaalin muodon. Hamming-ikkunointi pehmensi alku- ja loppunousuja, ja FFT-muunnoksella signaalin käsittelyssä siirryttiin taajuustasoon. Mel-taajuuskepstrikertoimilla kyettiin jäljittelemään ihmisen logaritmista tapaa aistia ääntä. Logaritmisen kompression jälkeen palattiin takaisin aikatasoon diskreetillä kosinimuunnoksella. Jokaista 256 näytteen aikaikkunaa vastasi tämän jälkeen sarja, noin 12 – 20 kpl MFCC-kertoimia, jotka kuvasivat äänisignaalin ominaisuuksia aikaikkunan alueella.

Puheentunnistus perustui aiemmin talletettujen näytteiden peräkkäisten piirrevektoreiden vertailuun puhutun signaalin vastaavien kanssa. Puhesignaalin ajallista eriaikaisuutta hyvin sietävää DTW-algoritmia käytettiin sanojen tunnistukseen.

Puheentunnistuksessa saavutettiin hyvä tarkkuus yhden puhujan yksittäisten sanojen tunnistuksessa. Kun käytettiin viiden sanan tunnistuksessa 14 eri referenssikehystä, saavutettiin 88 % tarkkuus sanojen tunnistuksessa. Vain viisi näytettä 52 näytteen joukosta tulkittiin väärin. Yhtä sanaa tunnistamassa oli 2-5 referenssikehystä. Testitapausten läpikäynnin jälkeen referenssikehysten määrää vähennettiin ja vain parhaiten tunnistavat jätettiin jäljelle. Kuudella referenssikehyksellä tunnistettiin neljää eri sanaa. Opetussanojen lausuja saavutti 100 % tarkkuuden neljän sanan aineistolla. Toinen testaaja pääsi 80 % tarkkuuteen, ja vain yksi viidestä näytteestä tunnistettiin väärin.

Kandidaatintyö saatiin suoritettua hyvin tuloksin loppuun saakka, ja työn tekeminen antoi konkreettisen tuntuman tekniikan suomiin mahdollisuuksiin signaalinkäsittelyn alalla sulautetussa järjestelmässä, mistä liukulukuprosessori puuttui.

Samalla rajoitukset tulivat konkreettisesti esille. Puheen tunnistaminen ei vielä tarkoita puheen ymmärtämistä.

9. LÄHTEET

[1]
Amin T. B. & Mahmood I. (2008) Speech Recognition using Dynamic Time Warping. Advances in Space Technologies. ICAST 2008. 2nd International Conference on 29-30 Nov. 2008, s. 74 – 79.

[2]
Allen, J.B. (1994) How Do Humans Process and Recognize Speech? Speech and Audio Processing, IEEE Transactions on Volume 2, Issue 4, Oct. 1994, s. 567 – 577.

[3]
Rabiner L. & Juang B.-H. (1993) Fundamentals of Speech Recognition, Prentice Hall, 507 s.

[4]
Juang B.-H. & Furui S. (2000) Automatic Recognition and Understanding of Spoken Language-A First Step Toward Natural Human-Machine Communication. Proceedings of the IEEE, Vol. 88, No. 8, August 2000.

[5]
Juang B. H. & Rabiner L. R. (2005) Automatic Speech Recognition—A Brief History of the Technology.

[6]
Peltola J. (2009) Introduction to the speech recognition. Opetusluento 16.1.2009.

[7]
Lennig, M. & Bielby, G. & Massicotte, J. (1994) Directory Assistance Automation in Bell Canada: Trial Results. Interactive Voice Technology for Telecommunications Applications, 1994, Second IEEE Workshop on 26-27 Sept. 1994, s. 9 – 13.

[8]
Wang Y.-Y. & Yu D.& Ju Y.-C. & Acero, A. (2008) An Introduction to Voice Search. Signal Processing Magazine, IEEE Volume 25, Issue 3, May 2008, s. 28 – 38.

[9]
IBM, Webspere Voice Server.(luettu 24.2.2009), URL: http://www-01.ibm.com/software/pervasive/voice_server/

[10]
Nuance, Vocon, (luettu 24.2.2009), URL: http://www.nuance.com/vocon/

[11]
Microsoft Speech Server, (luettu 24.2.2009), URL: http://www.microsoft.com/speech/speech2007/default.mspx

[12]
Jurafsky D. & Martin J.H. (2000) Speech and Language Processing; An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, Prentice Hall.

[13]
Spanias, A.S. & Wu, F.H. (1991) Speech Coding and Speech Recognition Technologies: A Review. Circuits and Systems, IEEE International Symposium on 11-14 June 1991, s. 572 – 577, vol.1.

[14]
Campbell, J.P., Jr. (1997) Speaker Recognition: A Tutorial. Proceedings of the IEEE Volume 85, Issue 9, Sept. 1997, s.1437 – 1462.

[15]
Muthusamy, Y.K. & Barnard, E. & Cole, R.A. (1994) Reviewing automatic language identification. Signal Processing Magazine, IEEE Volume 11, Issue 4, Oct. 1994, s. 33 – 41.

[16]
Vihola, M. (2002) Dissimilarity Measures for Hidden Markov Models and Their Application in Multilingual Speech Recognition. Diplomityö. Tampereen teknillinen korkeakoulu, tietotekniikan osasto, Tampere.

[17]
Möttönen V. & Pakanen J. & Peltola J. & Salmela M. & Seppänen T. (1998) Puheteknologian hyödyntäminen rakennusten teknisissä järjestelmissä, Valtion teknillisen tutkimuskeskuksen tiedote, Espoo, Finland.

[18]
Kotilainen S. (2008) Puheentunnistus yleistyy vihdoinkin. Tietokone-lehti 1/2008, Sanoma Magazines Finland. (Luettu 23.5.2009) URL:
http://www.tietokone.fi/lukusali/artikkelit/2008tk01/puheentunnistus.htm

[19]
Kurimo M. (2009) Puheentunnistus. Harjoitustyö, Digitaalinen signaalinkäsittely ja suodatus. Teknillinen Korkeakoulu, Informaatiotekniikan laboratorio, Helsinki. (Luettu 23.5.2009) URL:
http://www.cis.hut.fi/Opinnot/T-61.246/puheentunnistus_kurimo.pdf

[20]
Lévy C., Linarés G., Nocera P., Bonastre J.F. Reducing computational and memory cost for cellular phone embedded speech recognition system, Acoustics, Speech, and Signal Processing, 2004. Proceedings. (ICASSP '04). IEEE International Conference, s. 309-312.

[21]

Phadke S, Limaye R, Verma S, Subramanian K On Design and Implementation of an Embedded Automatic Speech Recognition System, Proceedings of the 17th International Conference on VLSI Design, 2004 IEEE, s. 127-132.

[23]
Li G, Wang Y, Li M, Wu Z Similarity Match in Time Series Streams under Dynamic Time Warping Distance, International Conference on Computer Science and Software Engineering, 2008, s.399-402.

[24]
Moshab B Speech Recognition for Disabilities People, IEEE Information and Communication Technologies, 2006, s.864-869.

[26]
http://www.finnlectura.fi/verkkosuomi/Fonologia/sivu111.htm - 16.3.2009

[27]
http://www.oulu.fi/hutk/fonetiikka/esittely.html - 16.3.2009

[28]
http://www.phon.utu.fi/ - 16.3.2009

[29]
http://www.finnlectura.fi/verkkosuomi/Fonologia/sivu12.htm - 16.3.2009

[30]
http://www.laivanupotus.net/saannot.php

[31]
Ethernut hardware manual (luettu 29.1.2009) URL:
http://www.ethernut.de/pdf/eir_um_1_0_6.pdf

[32]
Ethernut software manual, NutApi (luettu 29.1.2009) URL:
http://www.ethernut.de/api/index.html.

[33]
Picone, J.W. (1993) Signal modeling techniques in speech recognition. Proceedings of the IEEE Volume 81, Issue 9, Sept. 1993, s.1215 – 1247.

[34]
Doxygen - http://www.stack.nl/~dimitri/doxygen/

10. LIITTEET

Liite 1 Puheentunnistuksen testaaminen

Liite 2 Piirrevektoreiden laskenta vaihe vaiheelta

Liite 1 Puheentunnistuksen testaaminen

Liitteessä esitetään kaksi puheentunnistuksen testitapausta. Käytetyt referenssimallit on lueteltu testitapauksen yhteydessä. Virheelliset tunnistustapaukset on merkitty lihavoinnilla, pienin kustannusfunktio on merkitty alleviivaten. Lihavoidulla ja kursiivilla merkityt ovat kyseisen sanan referenssimallit. Rivin muut mahdolliset lihavoidut arvot ovat pienempiä kuin sanan referenssifunktiolla.

TESTITAPAUS 1
==========
Testissä käytetyt referenssimallit (13 kpl):
y1, y2, v1, v2, v3, a1, a2, a3, a4, m1, m2, t1, t2

YKSINPELI
Tunnistettu luku: **4, NäyteNro: 2**, arvot:
1208, 1152, **943**, 1383, 1475, 1439, 2174, 2489, 2581, 1521, **913**, 1272, **1017**

VERKKOPELI
Tunnistettu luku: 2, NäyteNro: 1, arvot:
1213, 1138, 613, 1213, 1083, 1293, 2000, 2195, 2276, 1378, 803, 1077, 1021

ASETUKSET
Tunnistettu luku: **5, NäyteNro: 2**, arvot:
1779, 1484, **960**, 1806, 1764, *1186, 2562, 2338, 2064*, **1440, 1349, 1002**, **953**
Tunnistettu luku: **5, NäyteNro: 2**, arvot:
1417, 1336, 1348, 1761, 1570, *1286, 2242, 2436, 2921*, **1217, 1274, 1111**, **1080**
Tunnistettu luku: **5, NäyteNro: 2**, arvot:
1842, 1550, **1213**, 1553, 1750, *1338, 2441, 2307, 1803*, 1624, 1475, **1233**, **1186**

MONINPELI

TULOSTUS
Tunnistettu luku: 5, NäyteNro: 1, arvot:
1890, 1918, 1046, 2237, 2558, 1693, 2979, 2795, 2488, 1819, 1397, 966, 1116
Tunnistettu luku: 5, NäyteNro: 1, arvot:
1686, 1697, 1563, 1998, 1975, 1353, 2338, 2205, 3255, 1374, 1342, 1063, 1092

TESTITAPAUS 2
==========
Lisättiin yksi uusi referenssimalli a5.
Testissä käytetyt referenssimallit (14 kpl):
y1, y2, v1, v2, v3, a1, a2, a3, a4, a5, m1, m2, t1, t2

YKSINPELI
y1, y2, v1, v2, v3, a1, a2, a3, a4, a5, m1, m2, t1, t2
Tunnistettu luku: 1, NäyteNro: 1, arvot:

1076, 1429, 1650, 2044, 1967, 2000, 2507, 2573, 2962, 1949, 1878, 1573, 1903, 1782
Tunnistettu luku: 1, NäyteNro: 1, arvot:
1047, 1525, 1713, 2144, 1979, 1698, 2509, 2484, 3125, 2008, 2114, 1475, 1829, 1762
Tunnistettu luku: 1, NäyteNro: 2, arvot:
1771, 1409, 1709, 1504, 1748, 2261, 2345, 2476, 2912, 1897, 1716, 1721, 2253, 1908
Tunnistettu luku: 1, NäyteNro: 1, arvot:
1420, 1493, 1848, 1815, 2233, 2222, 2453, 2387, 2602, 1582, 2106, 1953, 2028, 1942
Tunnistettu luku: 1, NäyteNro: 1, arvot:
1013, 1263, 1309, 1800, 1748, 1769, 2602, 2629, 2962, 1706, 1903, 1371, 1644, 1598
Tunnistettu luku: 1, NäyteNro: 1, arvot:
2911, 3231, 3694, 3297, 3984, 3593, 3753, 3300, 4106, 3216, 3909, 3504, 3802, 3814
Tunnistettu luku: 1, NäyteNro: 1, arvot:
1034, 1406, 1585, 2022, 1820, 1759, 2473, 2546, 2988, 1803, 1923, 1394, 1795, 1738
Tunnistettu luku: 1, NäyteNro: 1, arvot:
2108, 2381, 2852, 2745, 3339, 3320, 3371, 3133, 3467, 2769, 3122, 2791, 3345, 3044
Tunnistettu luku: 1, NäyteNro: 1, arvot:
994, 1438, 1616, 2019, 1864, 1847, 2509, 2606, 3046, 1879, 1993, 1328, 1867, 1802
Tunnistettu luku: 1, NäyteNro: 1, arvot:
2154, 2376, 2800, 2714, 2990, 3051, 2910, 2816, 3642, 2721, 3102, 2617, 2957, 3130
-> luku 1, näyte 1 (9 kpl) ja näyte 2 (1 kpl, 5 muuta näytettä ennen luku1, näytettä 1)

VERKKOPELI
y1, y2, ***v1, v2, v3***, a1, a2, a3, a4, a5, m1, m2, t1, t2
Tunnistettu luku: 2, NäyteNro: 1, arvot:
1743, 1809, 1002, 1441, 1484, 1662, 2454, 2638, 2427, 1059, 2125, 1256, 1375, 1164
Tunnistettu luku: 2, NäyteNro: 1, arvot:
1564, 1570, 883, 1407, 1165, 1607, 2399, 2625, 2713, 1210, 1686, 1116, 1444, 1308
Tunnistettu luku: **3, NäyteNro: 5**, arvot:
2063, 1772, ***1453, 1309***, 1995, 2273, 2570, 2421, 2131, **1121**, 2222, 1902, 1957, 1791
Tunnistettu luku: 2, NäyteNro: 1, arvot:
1675, 1725, 965, 1506, 1301, 1651, 2704, 2742, 2657, 1372, 1773, 1243, 1458, 1402
Tunnistettu luku: 2, NäyteNro: 1, arvot:
1579, 1528, 1068, 1526, 1430, 1543, 2628, 2773, 2950, 1444, 1639, 1519, 1578, 1503
Tunnistettu luku: 2, NäyteNro: 1, arvot:
1547, 1526, 873, 1451, 1204, 1463, 2419, 2835, 2572, 1149, 1839, 1068, 1175, 1154
Tunnistettu luku: 2, NäyteNro: 1, arvot:

1954, 1702, 1118, 1450, 1329, 1723, 2647, 2894, 3277, 1464, 1935, 1493, 1778, 1619
Tunnistettu luku: 2, NäyteNro: 1, arvot:
2024, 2119, 1100, 2404, 2981, 1874, 3113, 2958, 2698, 1331, 2357, 1490, 1267, 1373
Tunnistettu luku: 2, NäyteNro: 1, arvot:
1634, 1703, 958, 1503, 1333, 1675, 2663, 2843, 2721, 1261, 1921, 1230, 1426, 1359
Tunnistettu luku: 2, NäyteNro: 1, arvot:
1546, 1592, 823, 1371, 1216, 1515, 2525, 2733, 2634, 1242, 1770, 1145, 1452, 1364
-> luku 2: näyte 1 (9 kpl), näyte 2 (0 kpl), näyte 3 (0 kpl), virheelliset (1kpl: luku3,näyte 5)

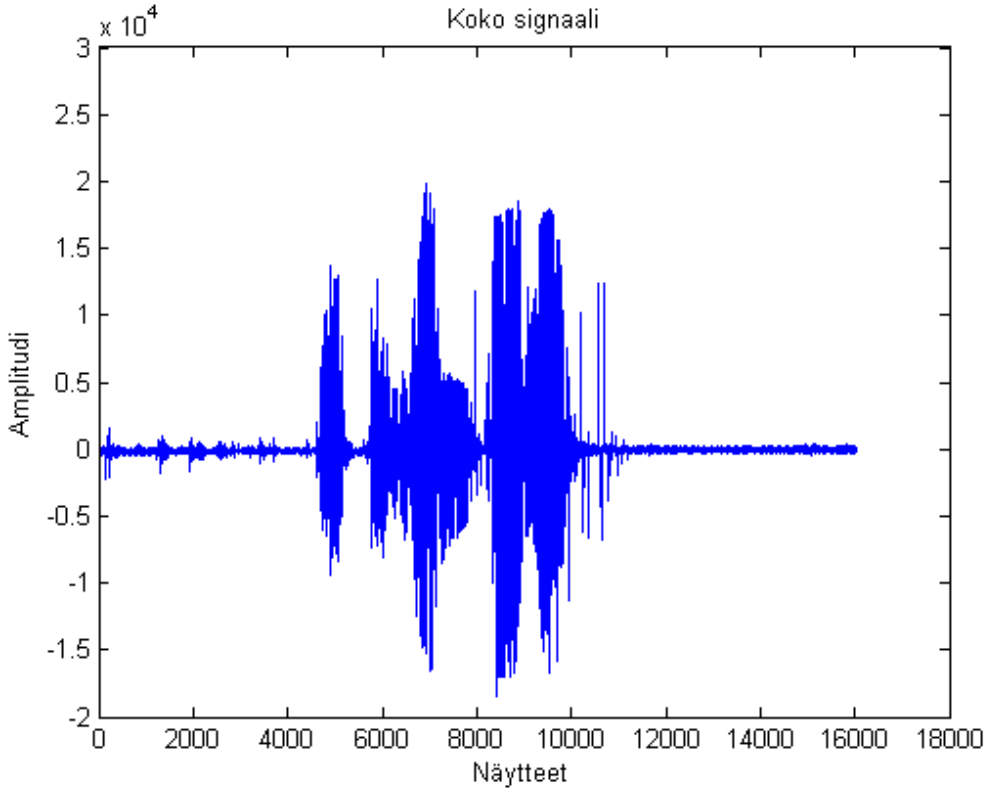
ASETUKSET
y1, y2, v1, v2, v3, ***a1, a2, a3, a4, a5***, m1, m2, t1, t2
Tunnistettu luku: 3, NäyteNro: 5, arvot:
2283, 1932, 1952, 1868, 2341, 2114, 2602, 2034, 1737, 1472, 2149, 2340, 1882, 1815
Tunnistettu luku: 3, NäyteNro: 5, arvot:
3055, 2829, 2890, 2757, 3087, 2775, 2995, 2590, 2474, 2325, 2917, 3113, 2716, 2642
Tunnistettu luku: 3, NäyteNro: 5, arvot:
2605, 2330, 2250, 2304, 2772, 2407, 3137, 2633, 1985, 1664, 3004, 2576, 2283, 2153
Tunnistettu luku: 3, NäyteNro: 5, arvot:
2317, 1996, 1903, 2120, 2449, 1990, 2545, 2224, 1850, 1403, 2447, 2328, 2009, 1884
Tunnistettu luku: 3, NäyteNro: 1, arvot:
1819, 1503, 1292, 1841, 1511, 1194, 2471, 2561, 2492, 1393, 1687, 1554, 1450, 1479
Tunnistettu luku: 3, NäyteNro: 5, arvot:
2086, 1843, 1775, 1981, 2282, 1805, 2602, 2165, 1762, 1322, 2168, 2005, 1801, 1729
Tunnistettu luku: 3, NäyteNro: 5, arvot:
2085, 1730, 1630, 1799, 1990, 1825, 2570, 2367, 1791, 1150, 2084, 1869, 1615, 1557
Tunnistettu luku: 3, NäyteNro: 5, arvot:
2194, 1927, 1724, 2012, 2337, 2037, 2679, 2393, 1922, 1351, 2335, 2133, 1761, 1742
Tunnistettu luku: 3, NäyteNro: 5, arvot:
2054, 1879, 1786, 1882, 2258, 2010, 2646, 2295, 2022, 1253, 2456, 1998, 1688, 1618
Tunnistettu luku: 3, NäyteNro: 5, arvot:
2611, 2626, 2289, 2527, 3087, 2541, 3200, 2621, 2099, 1782, 3196, 2733, 2288, 2401
-> luku 3: näyte 1 (1 kpl: verkko1 ensin, sitten luku3,näyte 2),näyte 2 (0), näyte 3 (0), näyte 4 (0), näyte 5 (9 kpl),

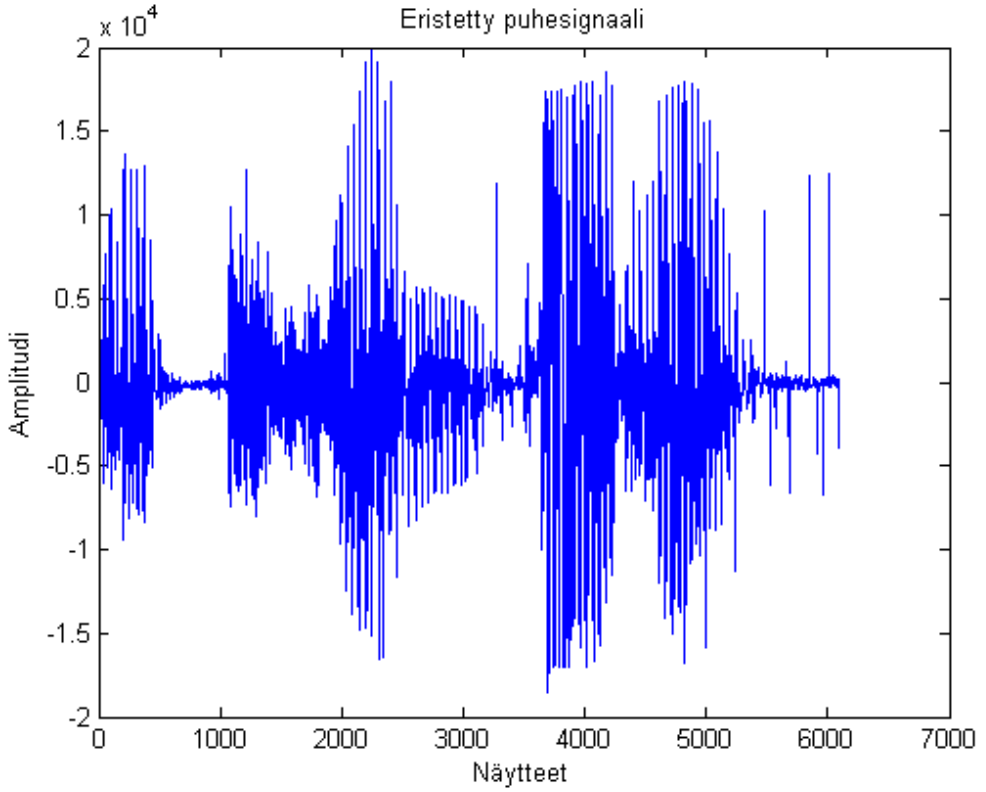
MONINPELI
y1, y2, v1, v2, v3, a1, a2, a3, a4, a5, ***m1, m2***, t1, t2
Tunnistettu luku: 4, NäyteNro: 2, arvot:
1335, 1334, 1401, 1510, 1411, 1518, 2666, 2519, 2586, 1438, 1372, 997, 1177, 1114
Tunnistettu luku: 4, NäyteNro: 2, arvot:
1215, 1126, 1164, 1318, 1091, 1457, 2596, 2586, 2751, 1379, 1040, 789, 1169, 1023
Tunnistettu luku: 4, NäyteNro: 2, arvot:
1647, 1571, 1309, 1616, 1644, 1654, 2397, 2381, 2623, 1681, 1444, 1184, 1627, 1479
Tunnistettu luku: 4, NäyteNro: 1, arvot:
2031, 1718, 1611, 1734, 1843, 2017, 2814, 2550, 2434, 1741, 1606, 1632, 1906, 1737
Tunnistettu luku: 2, NäyteNro: 2, arvot:
2116, **1336**, 1786, **1097**, **1640**, 2216, 2846, 2559, 2405, **1432**, ***1682, 1661***, 2047, 1740
Tunnistettu luku: 2, NäyteNro: 2, arvot:
2543, **1859, 1954**, **1448**, **2125**, 2499, 3253, 2820, 2327, **1746**, ***2188, 2151***, 2488, 2193
Tunnistettu luku: 4, NäyteNro: 1, arvot:
1955, 1714, 1769, 1724, 1769, 1782, 2374, 2192, 2613, 1735, 1485, 1623, 1831, 1593
Tunnistettu luku: 4, NäyteNro: 2, arvot:
1410, 1140, 1421, 1359, 1225, 1652, 2892, 2841, 2961, 1593, 1217, 972, 1403, 1226
Tunnistettu luku: 4, NäyteNro: 2, arvot:
1277, 1140, 1128, 1210, 959, 1474, 2656, 2746, 3036, 1479, 1199, 746, 1257, 1084
Tunnistettu luku: 4, NäyteNro: 2, arvot:
1382, 1975, 1450, 2093, 1899, 1516, 2042, 2471, 3457, 2320, 2194, 922, 2173, 1987
-> luku 4: näyte 1(2 kpl, toisessa näyte 2 seuraava, toisessa verkko1 pääsee väliin, sitten näyte 2), näyte 2 (6 kpl), virheelliset (2 kpl)

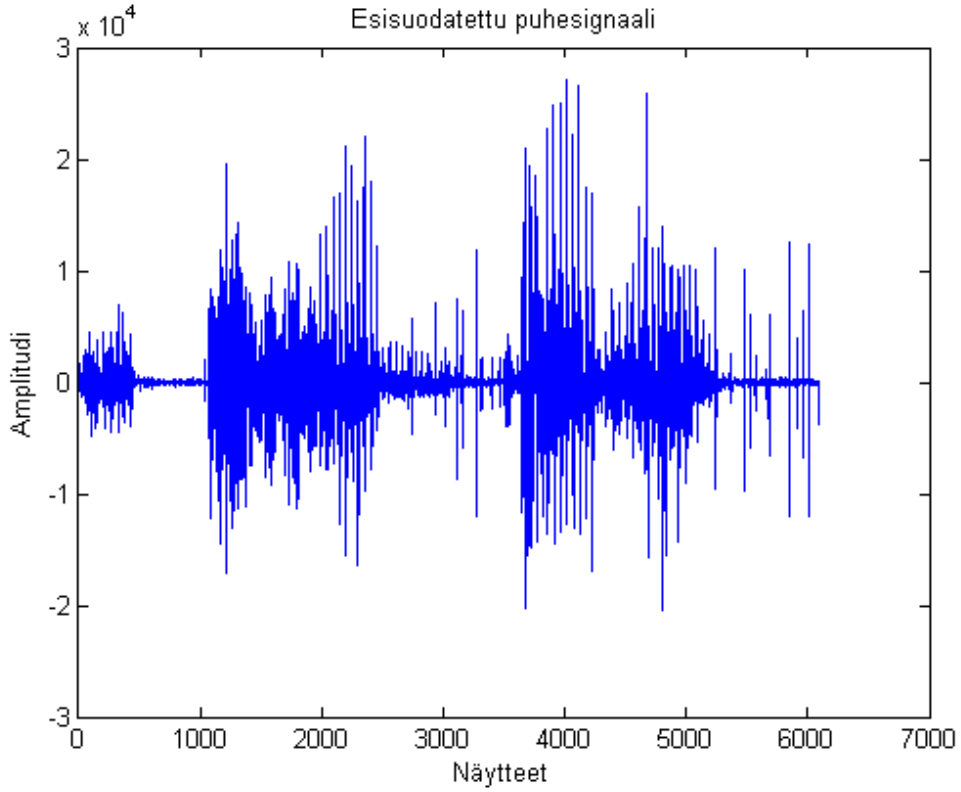
TULOSTUS
y1, y2, v1, v2, v3, a1, a2, a3, a4, a5, m1, m2, ***t1, t2***
Tunnistettu luku: 5, NäyteNro: 1, arvot:
1851, 1851, 1575, 2022, 1950, 1466, 2671, 2432, 2709, 1448, 1763, 1443, 1080, 1095
Tunnistettu luku: 5, NäyteNro: 2, arvot:
2908, 2774, 2725, 2701, 3010, 2226, 2875, 2574, 2855, 2334, 2898, 2822, 2187, 2085
Tunnistettu luku: 5, NäyteNro: 1, arvot:
2107, 2113, 1695, 2326, 2258, 1649, 2926, 2428, 2666, 1586, 2095, 1824, 1289, 1463
Tunnistettu luku: **4, NäyteNro: 2**, arvot:
1896, 2892, **1761**, 2248, 3016, **1724**, 2141, 2301, 3161, 2011, 3125, **1626**, ***2306, 1976***
Tunnistettu luku: 5, NäyteNro: 2, arvot:
1810, 1683, 1553, 1599, 1478, 1452, 2528, 2481, 2783, 1446, 1460, 1344, 1241, 938
Tunnistettu luku: 5, NäyteNro: 2, arvot:
1831, 1602, 1537, 1622, 1603, 1556, 2693, 2469, 2564, 1266, 1483, 1271, 1068, 999,

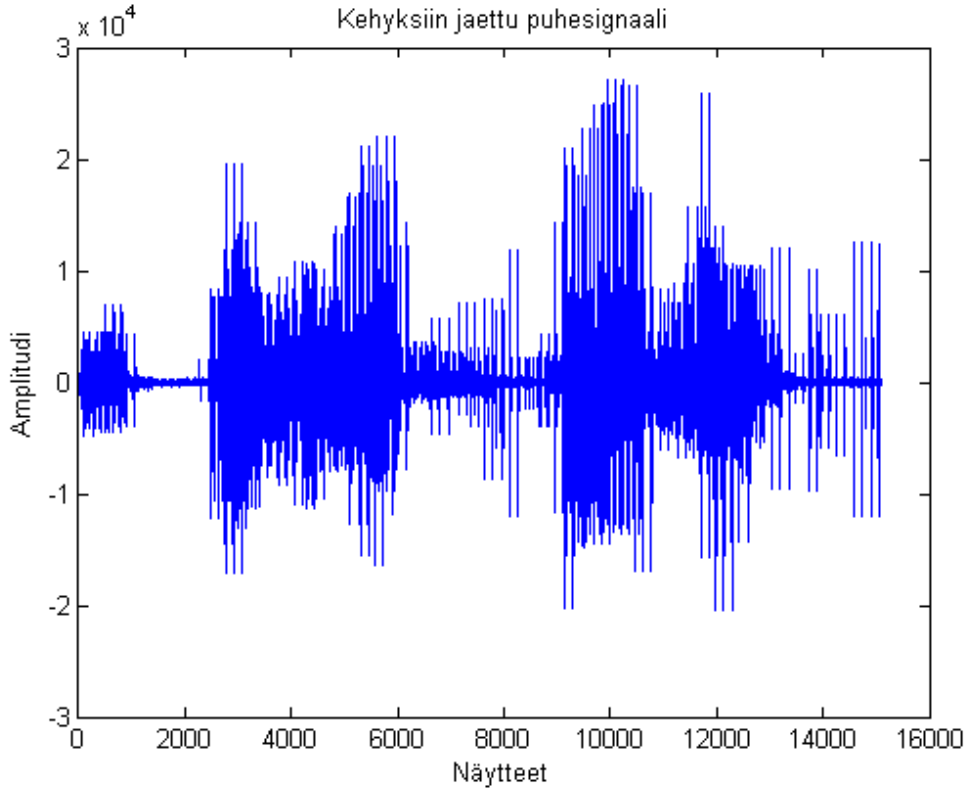
Tunnistettu luku: **4, NäyteNro: 2**, arvot:
1892, 2321, 1745, 2381, 2156, 1709, 2020, 2299, 3113, 1957, 2204, **1527**, ***1722, 1668***,
Tunnistettu luku: **3, NäyteNro: 1**, arvot:
3044, 3604, 3065, 3319, 3263, **2171**, 2974, 2875, 4028, 3085, 3457, 2843, ***2835, 2513***,
Tunnistettu luku: 5, NäyteNro: 2, arvot:
1868, 1821, 1672, 1704, 1611, 1493, 2647, 2547, 2820, 1560, 1682, 1353, 1240, 1095
Tunnistettu luku: 5, NäyteNro: 2, arvot:
1802, 1780, 1555, 1881, 1767, 1504, 2797, 2485, 2630, 1386, 1636, 1363, 1021, 987
Tunnistettu luku: 5, NäyteNro: 2, arvot:
2000, 1938, 1619, 2043, 1950, 1593, 2706, 2479, 2743, 1498, 1841, 1610, 1235, 1203
Tunnistettu luku: 5, NäyteNro: 2, arvot:
1835, 1708, 1448, 1858, 1733, 1443, 2674, 2541, 2588, 1273, 1606, 1297, 1033, 876
-> luku 5: näyte 1 (2: näyte 2 oli seuraava), näyte 2 (7), virheelliset (3: 2 kpl luku 4, näyte 2, 1 kpl luku 3, näyte 1)

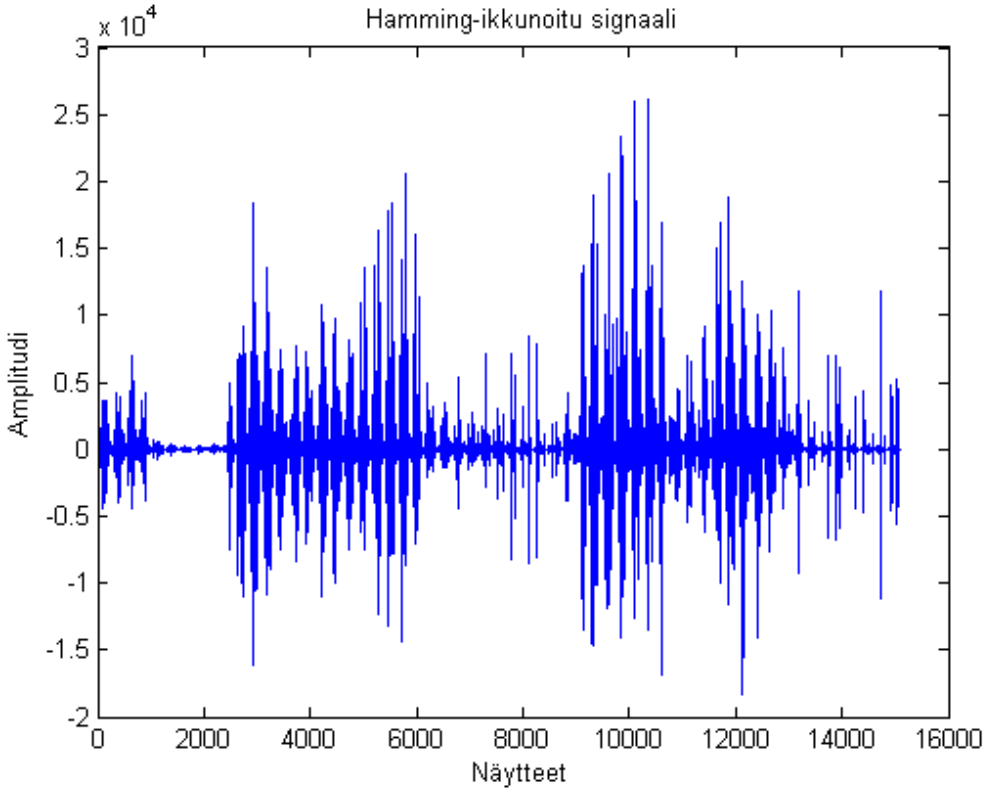
Liite 2 Signaali piirrevektoreiden laskennan eri vaiheissa.

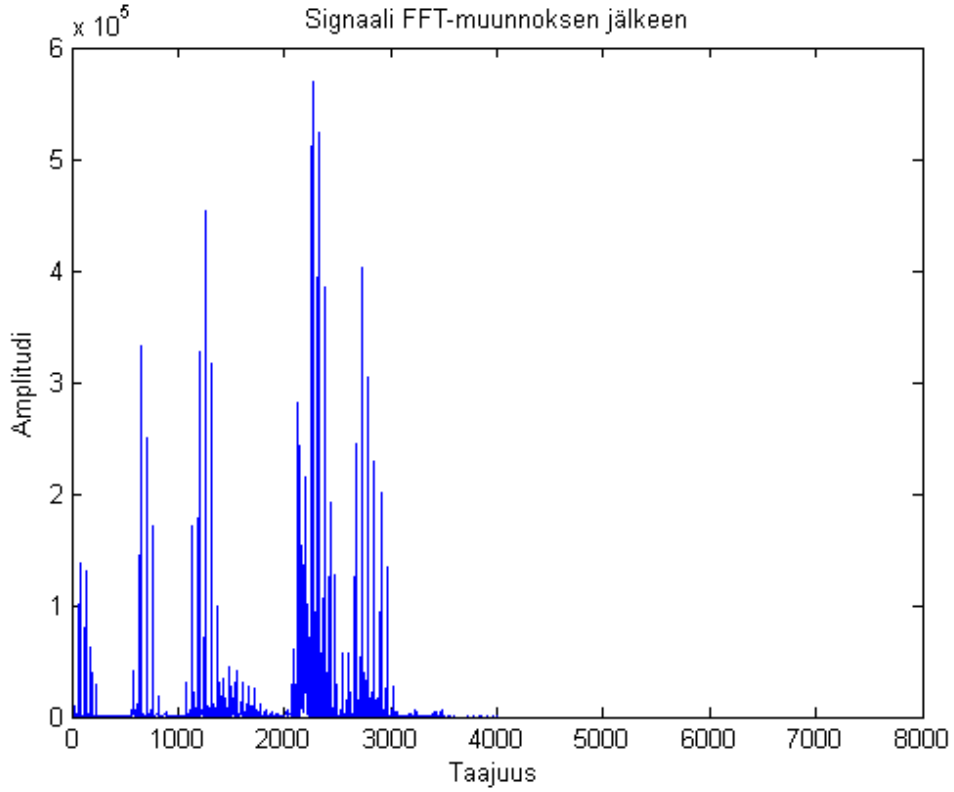


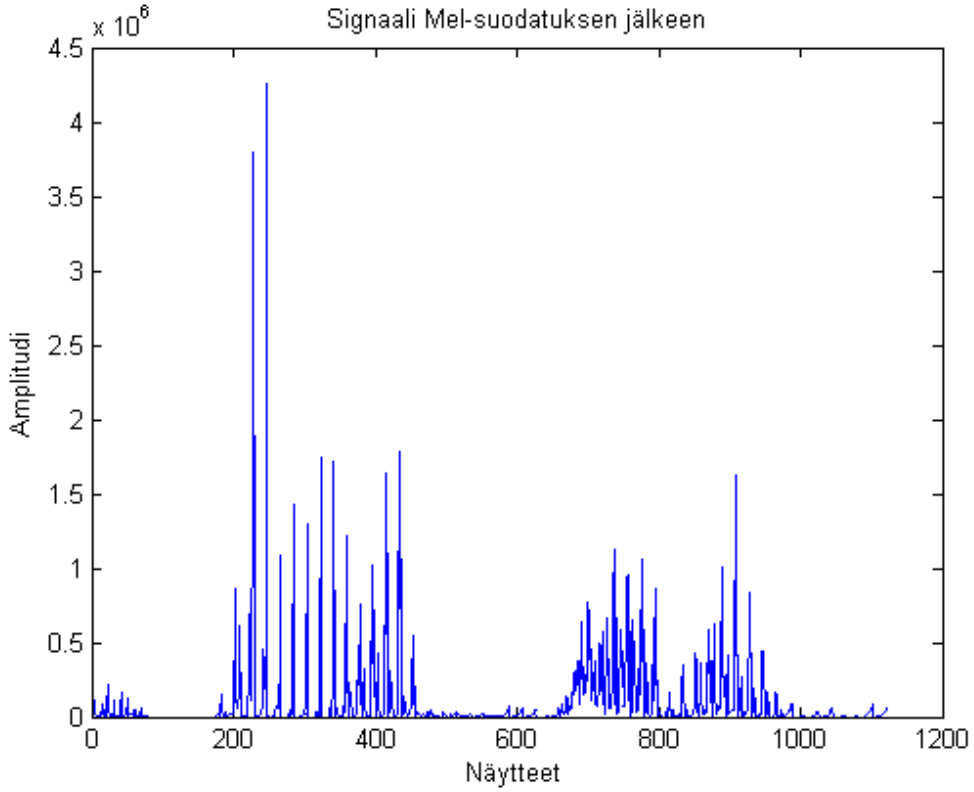


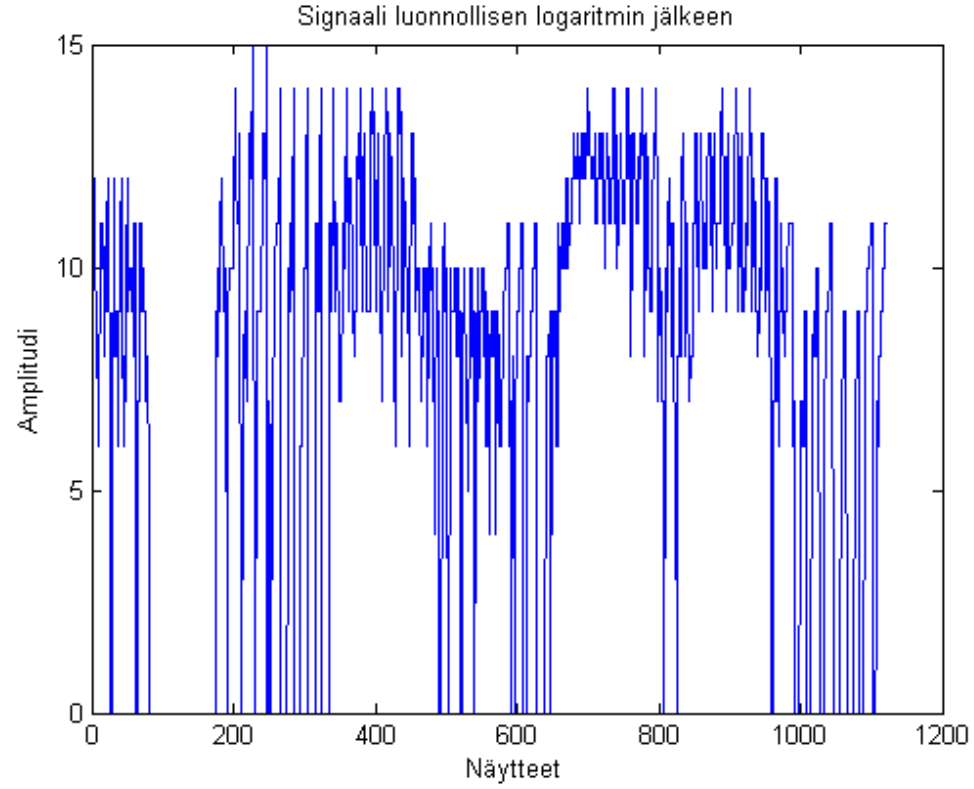

Esisuodatettu puhesignaali
x 10^4
3
2
1
0
-1
-2
-3
Amplitudi
0
1000
2000
3000
4000
5000
6000
7000
Näytteet

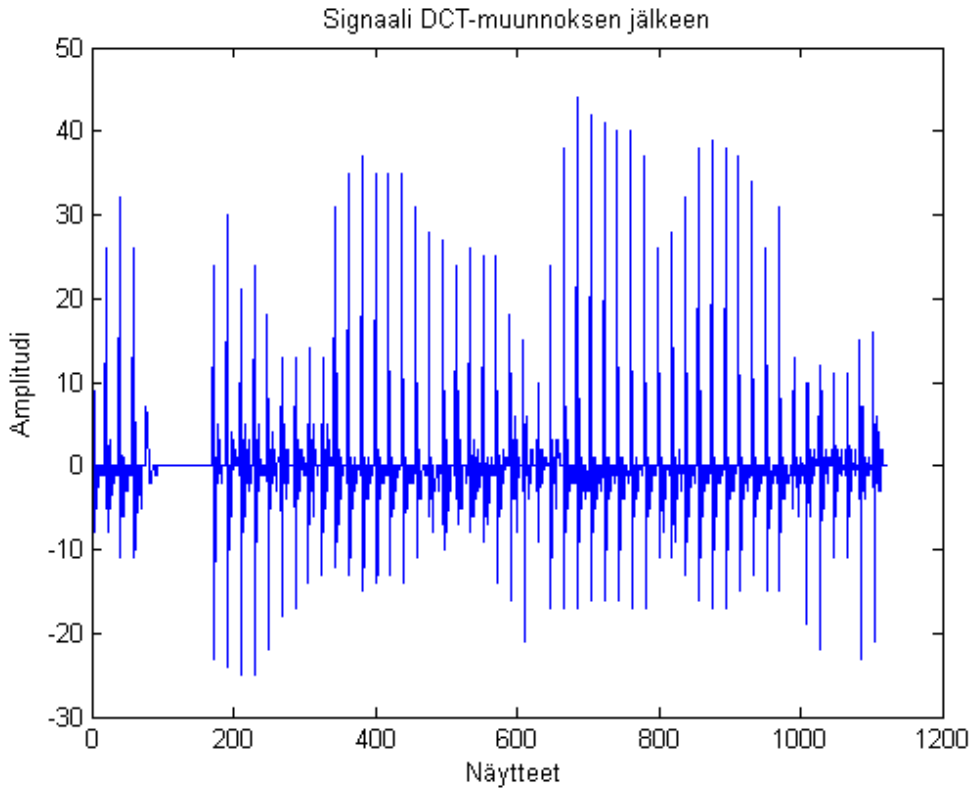

Kehyksiin jaettu puhesignaali
x 10^4
3
2
1
0
-1
-2
-3
Amplitudi
0
2000
4000
6000
8000
10000
12000
14000
16000
Näytteet


Hamming-ikkunoitu signaali
x 10^4
Amplitudi
Näytteet
3
2.5
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
0
2000
4000
6000
8000
10000
12000
14000
16000


Signaali FFT-muunnoksen jälkeen
x 10^5
Amplitudi
Taajuus
6
5
4
3
2
1
0
0
1000
2000
3000
4000
5000
6000
7000
8000


Signaali Mel-suodatuksen jälkeen
x 10^6
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
Amplitudi
0
200
400
600
800
1000
1200
Näytteet


Signaali luonnollisen logaritmin jälkeen
15
10
5
0
Amplitudi
0
200
400
600
800
1000
1200
Näytteet


Signaali DCT-muunnoksen jälkeen
Amplitudi
Näytteet
50
40
30
20
10
0
-10
-20
-30
0
200
400
600
800
1000
1200