Disclaimer: This document was part of the DSP Solution Challenge 1995 European Team Papers. It may have been written by someone whose native language is not English. TI assumes no liability for the quality of writing and/or the accuracy of the information contained herein.

Adaptive Speech Beamforming Using the TMS320C40 Multi-DSP

Authors: P. Wauters, K. Eneman, K. Delaet, R. Lauwereins

EFRIE, France

December 1995
SPRA305

TEXAS INSTRUMENTS

IMPORTANT NOTICE

Texas Instruments (TI) reserves the right to make changes to its products or to discontinue any semiconductor product or service without notice, and advises its customers to obtain the latest version of relevant information to verify, before placing orders, that the information being relied on is current.

TI warrants performance of its semiconductor products and related software to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Certain application using semiconductor products may involve potential risks of death, personal injury, or severe property or environmental damage ("Critical Applications").

TI SEMICONDUCTOR PRODUCTS ARE NOT DESIGNED, INTENDED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT APPLICATIONS, DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS.

Inclusion of TI products in such applications is understood to be fully at the risk of the customer. Use of TI products in such applications requires the written approval of an appropriate TI officer. Questions concerning potential risk applications should be directed to TI through a local SC sales office.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards should be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein. Nor does TI warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used.

Copyright © 1997, Texas Instruments Incorporated

TRADEMARKS

TI is a trademark of Texas Instruments Incorporated.

Other brands and names are the property of their respective owners.

CONTACT INFORMATION

US TMS320 HOTLINE	(281) 274-2320
US TMS320 FAX	(281) 274-2324
US TMS320 BBS	(281) 274-2323
US TMS320 email	dsph@ti.com

Contents

Figures

Tables

Adaptive Speech Beamforming Using the TMS320C40 Multi-DSP

Abstract

This application report describes the process of designing an adaptive speech beamformer using the Graphical Rapid Prototyping Environment (GRAPE-II). The GRAPE-II environment simplifies programming, compiling, simulation, debugging, and testing of real-time digital signal processor (DSP) algorithms. This project demonstrates the feasibility of this approach by actually designing a functioning beamformer.

A multiprocessor consisting of four Texas Instruments (TI™) TMS320C40 DSP processors, mounted on PC plug-in cards, is used for signal processing. The output of the beamformer is sent to a dual digital-to-analog (D/A) converter. The result is a real-time prototype working at a sampling rate of 8 kHz.

This report includes beamforming theory, broadband filter-and-sum beamforming, implementing the spatial selector algorithm, beamformer design, and a description of the hardware used in the design.

This document was an entry in the 1995 DSP Solutions Challenge, an annual contest organized by TI to encourage students from around the world to find innovative ways to use DSPs. For more information on the TI DSP Solutions Challenge, see TI's World Wide Web site at www.ti.com.

Product Support on the World Wide Web

Our World Wide Web site at www.ti.com contains the most up to date product information, revisions, and additions. Users registering with TI&ME can build custom information pages and receive new product updates automatically via email.

Introduction

The goal of this project was to develop an adaptive speech beamformer with the Graphical Rapid Prototyping Environment GRAPE-ll. The beamformer is designed to pick out a speaker in a noisy environment. The beamformer should improve the intelligibility of the speaker and improve the signal to noise ratio. A typical application for beamforming is teleconferencing with large audiences. The design consists of two parts:

- A set of fixed direction beamformers.
- An adaptive beam selection unit

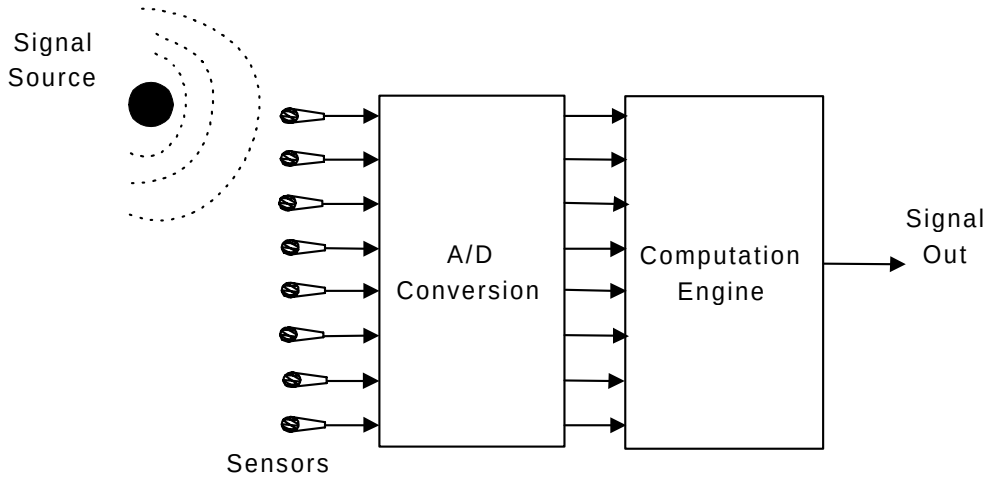
Each beamformer is the sum of eight microphone signals; each filtered with a FIR-filter. The filter-and-sum approach was used to obtain good broadband performance with a limited number of microphones. The selection unit continuously monitors the speech activity in each beam. The beam, in which significant speech activity is present, is activated. This speech detection is based on a complex heuristic algorithm.

The whole system consists of an array of eight microphones sampled by an eight-channel A/D converter. A single DSP converts the output format and performs an initial bandpass filter. A multiprocessor consisting of four TMS320C40 DSP processors is used for signal processing. The output of the beamformer is sent to a dual D/A converter. The results are a real-time prototype working at a sampling rate of 8 kHz.

Beamforming Theory

Beamforming is a technique that has been used for several decades in telecommunications radar and antenna array applications. Recently beamforming has been adopted for broadband applications such as speech enhancement, as shown in Figure 1).

Figure 1. Simplified Beamfinder



A beamformer is a set of sensors with outputs that can be steered electronically. In this case, the sensors are linear microphones. The sensors are positioned in an array. The array can be linear, planar, or 3-dimensional. Electronic steering employs signal processing techniques that exploit the spatial information in the sensor signals. This technique requires spatio-temporal filtering operations that may or may not be adaptive.

Signal sources emit spherical waves that eventually reach the sensor array. The sensor array comprises linear microphones that extract spatial information from the spherical waves. The spatial information obtained from the spherical waves is used by the beamfinder for spatial selection and spatial separation.

Beamforming uses the phase of the spherical wave signals received by the sensor array. The signals received at the sensor array are routed to a computation engine. The computation engine then digitizes the signals and sends them to a set of digital filters. Because of this computation logic, the beamformer calculates angular selectivity. The computation engines performance can be visually illustrated by means of an angular response diagram. Beamforming is the most common technique for spatial selection. It is well suited for disturbance suppression and de-reverberation.

This project focuses on broadband beamforming for acoustic applications. Acoustic beamformers should deal with the broadband character of speech and audio, that is, the non-stationary and dynamic range. The frequency band of interest lies between 300 and 3500 Hz, because this band is the most important to speech intelligibility.

In telecommunications classical beamforming deals with narrow band signals and relies upon a delay-and-sum principle. The various antenna signals are delayed and then summed together. Varying the delays allows the designer to steer the beamformer. A delay-and-sum beamformer has a specific angular selectivity pattern and restricted sidelobe suppression. An extension to this is the weighted-sum beamformer. The sensor signals are delayed, amplified, and summed. This allows the designer to shape the angular selectivity to the required pattern.

The classical beamforming techniques mentioned above were developed for narrow band telecommunication applications. The characteristics of the delay-and-sum and weighted-sum beamformer are extremely frequency dependent. To overcome this dependency, extensions for broadband acoustic beamforming are required.

Adaptive Beamforming

Adaptive beamforming integrates adaptive signal processing into the classical beamforming framework. Adaptive signal processing uses signal-enhancing techniques such as noise suppression, echo cancellation, and equalization.

The beamformer can be steered toward the selected source by specifying a steering direction and proper adjustment of the classical beamforming part. The adaptive filters attempt to filter out jammer signals. Jammer signals are received by the sensors from directions other than the selected signal source. To avoid cancellation of the selected source signal due to signal leakage, a blocking matrix is used to isolate the jammers. Because adaptive filters steadily converge into an acoustic transfer path, the filters should be reasonably long, especially in a strong reverberating environment**.** Reverberation due to signal leakage and increased computational loads degrades the performance of adaptive beamformers.

Sub-Band Beamforming

A second approach attempts to use subarrays to extrapolate the classical narrow-band beamforming techniques to broadband. For example, octave band splitting converts the broadband design into narrow band equivalents. Sub-band beamforming is easy to implement, but its performance is degraded for frequencies near the octave band edges.

Narrow-Band Weighted-Sum Beamforming

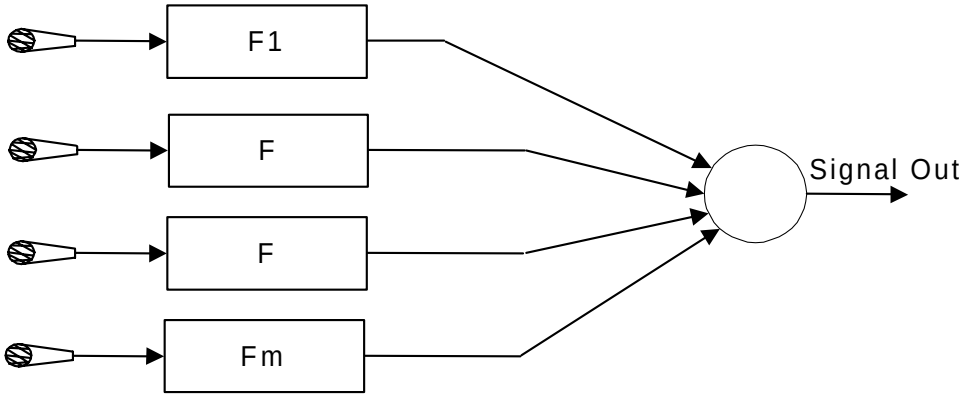
The third approach extended the narrow-band weighted-sum beamformer to broadband using a filter-and-sum structure. The various sensor signals are then routed to digital filters. The digital filters outputs are then combined by simple addition. A filter-and-sum beamformer is a logical extension for broadband applications since it can be thought of as a beamformer with frequency dependent weights. Optimal weights can be found for each frequency in the band of interest. These weights are finally combined into microphone dependent filters to obtain a set of fixed beams.

Broadband Filter-and-Sum Beamforming

The broadband beamformers design must be capable of suppressing *directive* jammers and acoustic *echoes*. This design can be achieved if the beamformers angular characteristics are selective. That is, the design must be directive for all frequencies of interest.

A directive-frequency, independent angular selectivity could be a reasonable objective. In this way, the optimization problem is formulated well and the design objectives can be clearly met. The filter-and-sum approach leads to spatio-temporal filters that offer a framework for achieving these objectives by reducing the broadband design to a set of narrow band designs (see Figure 2).

Figure 2. Filter and Sum Beamformer



This design philosophy requires a predefined angular selectivity pattern. The steering direction, sidelobe level, and shape can be chosen more or less freely. The pattern should however be directive.

NOTE:
The actual design was limited by the number of microphones and the computing power available.

A useful angular selectivity pattern can be obtained with the famous Dolph-Chebyshev algorithm. A directive narrow-band pattern with uniform sidelobe level is designed based on a user defined steering direction, sidelobe suppression and array size. The Dolph-Chebyshev algorithm was used in this project to obtain predefined angular patterns.

Subsequently, the array topology must be defined. In narrow-band applications the arrays are normally uniform. An array suitable for broadband applications is logarithmically separated to optimally replace the ideal narrow-band array for each frequency of interest.

Optimal complex microphone weights are required so that the predefined narrow-band pattern can be approximated sufficiently on the broadband array for a set of design frequencies covering the envisaged frequency band.

The optimization problem is formulated as a set of linear equations, that can be solved in least squares sense. Iterative algorithms, for example, based on steepest descent are used to smooth the patterns. The microphone weights are plotted against frequency. A filter for each microphone can be designed because its amplitude and phase response are observed. The optimal complex microphone weight optimization problem must be solved by avoiding filters that are too long.

Spatial Selector

Along with beamforming, a selection algorithm must be implemented to steer the beamformer automatically and track moving sources. Determining broadband source localization and estimating the angle of incidence for non-stationary speech-like signals is difficult. The selection algorithm has to be robust and operate with a limited computational complexity.

The selection algorithm is energy based. Energy is used as an indicator for speech activity, but not as a direct decision criterion. The fixed beams used in this project operate in parallel. Based on the energy evolution in each of the beams, one or more can be active, or audible. A beam is selected only if the average beam energy is sufficiently high.

The selection algorithm is implemented as follows:

1) The beam energies are estimated at regularly spaced points in time (time intervals of *frame_length*).

2) If the energy exceeds the predefined threshold, a variable called *activity* is increased. If the energy is less than the predefined threshold the *activity* variable is decreased.

3) *Activity* is restricted to the interval [0, *max_threshold*].

4) If the beam activity variable exceeds the threshold, *trans_0H*, the corresponding beam enters the active region and is considered activated.

5) For flexibility, an intermediate zone was added between the active and non-active region with reduced signal volume.

6) Further improvement was achieved by defining the linear-fading velocities *activity_rise* and *activity_fall.* These two velocities influence the rate by which *activity* is increased and decreased.

The selection algorithm will allow only the active speaker to be audible. For sufficiently large signal-to-noise ratios directive background noise will not exceed the activity threshold and thus will be suppressed. An appropriate frame length choice such as 300 ms enables the system to suppress sporadic noises such as coughs and taps.

An active source remains selected even when it falls below the activity threshold. This allows for gaps between words in normal speech. The active speaker should not be de-activated because there are gaps between words. However, frame length should not be too long. If frame length is too long overall flexibility of the algorithm is reduced. A new speaker is activated only after frame length time is complete.

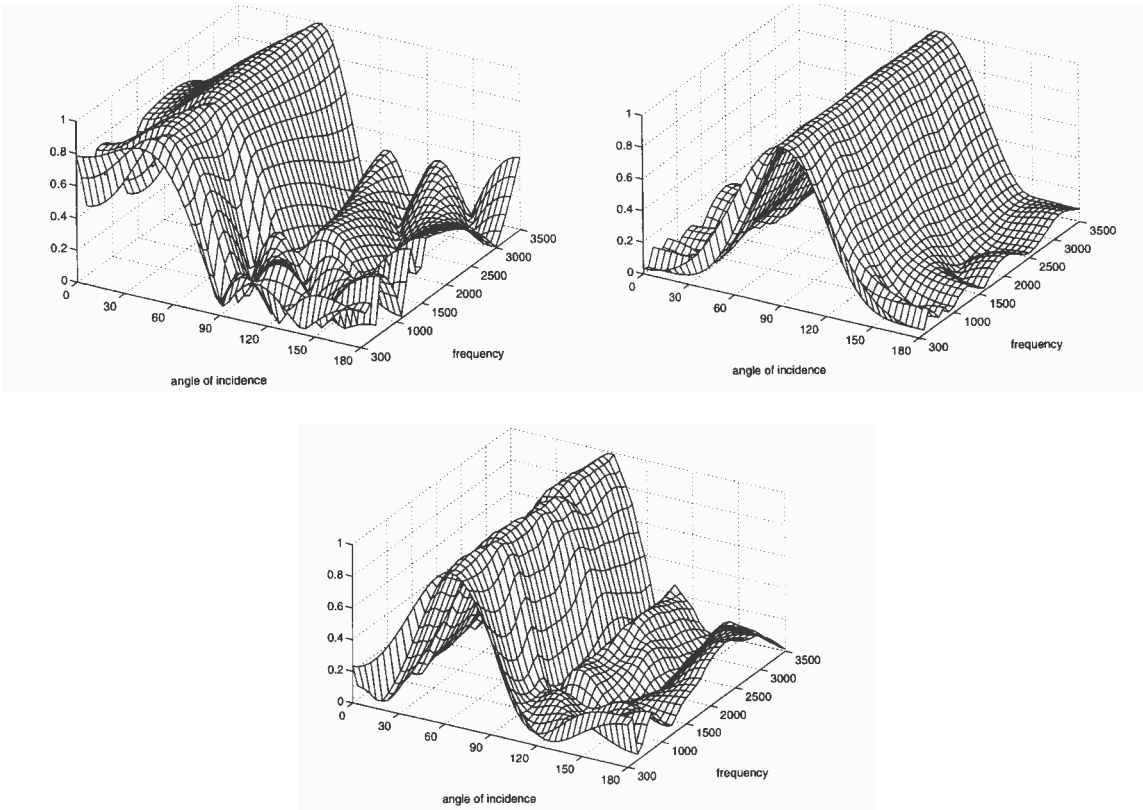
Designing the Beamformer

The design was limited by the processing power of the DSP and the number of microphones used. Taking into account the computation power of the DSP and the overhead required for communication between the parallel processors it was deemed feasible to implement a fixed beam design with a 8 kHz sampling rate. The selected steering angles were 0° (broadside direction), 65°, 115°, 45°, and 135°. The microphones were positioned at ±2.5 cm, ±7.5 cm, ±20 cm, and ±50 cm.

Sets of eight FIR filters were designed for all steering directions to obtain a frequency independent angular pattern for frequencies between 300 and 3500 Hz. There were 50 taps required for broadside, 81 taps for 65° and 115°, and 100 taps for 45° and 135°.

The plots in Figure 3 show the broadband angular selectivity pattern for 45°, broadside, and 65° as a function of frequency; 45° is shown in the upper left plot, broadside is shown in the upper right plot, and 65° is shown in the lower plot.

Figure 3. Angular Selectivity



The performance of the design was limited by the number of microphones. The filter lengths did not appear to be crucial design parameters. The total array size was one meter, which is short for 300 Hz. A limited directivity, especially for low frequencies is anticipated. The expected performance is summarized in Table 1.

Table 1. Summary of Beam Properties

	90°	65°	45°
3 dB bandwidth	48.3°	33.1°	50.3°
Maximum sidelobe level	-15.9 dB	-9.5 dB	-7.0 dB
Mean sidelobe level	-22.3 dB	-17.4 dB	-15.1 dB

Performance

A fixed filter-and-sum beamformer design must be robust with respect to system parameter uncertainties or deviations. The positioning of the microphones and differences in microphone characteristics, adjustment, and amplification are crucial design considerations. A well-designed beamformer is robust and the performance should not be degraded significantly from expected parameter variations.

The actual performance of the beamformer can deviate from the design requirements or from the anticipated performance if the beamformer is not robust enough or the parameters deviate more than anticipated.

The beamformer was designed with far-field assumptions that the acoustic signals will reach the microphone array as plane waves. These far-field assumptions no longer apply in many acoustic beamforming applications. As a consequence, the performance degradation is limited.

The performance of the system also depends on the reflectivity of the recording environment. In an anechoic room the performance was well within acceptable levels. Within a strongly reverberating environment performance quickly degrades, but the somewhat disappointing increase in signal-to-noise-ratio may still be considerable for intelligibility improvement.

Hardware

The design used a linear array of eight onni directional microphones (Sennheiser MKE 102).

The M16 multi-channel, 16 bit, A/D converter from Applied Speech Technologies, Inc. digitized the microphone signals. The output format of the M16 is a bit-serial digital signal in which the microphone samples were multiplexed in time (Motorola Serial Synchronous Interface protocol).

A Motorola 56000 DSP processor receives the signal from the M16 on its SSI serial link. The 56000 was used for two purposes:

- Provide protocol translation by converting the bit serial output protocol of the M16 into a bit-parallel protocol DSPLINK. The parallel protocol DSPLINK is a proprietary communication protocol of LSI (Loughborough Sound Images LTD). This proved to be the easiest way to transport the output of the M16 to the C40 multi-processor.
- The 56000 was also used as a data conditioner to filter out the frequencies below 300 Hz and above 3500 Hz.

The TMS320C40 DSP multiprocessors receive samples over the DSPLINK in a time-multiplexed technique. The output of the beamformer is sent to a D/A converter, a LSI Crystal Analog daughter module, the AM/D16DS.

In this project only commercially available LSI (multi) processor boards were used. A board with a single Motorola 56000 DSP and two boards (LSI DPCC40) each with two Texas Instruments TMS320C40 DSPs running at 40 MHz were used. All DSP boards are PC plug-in cards. The four Texas Instruments TMS320C40 DSPs were connected via their communication ports.

Rapid Prototyping and Software Description

The ever-increasing complexity and data rates of DSP applications demand application-specific ICs (ASICs) and hardware. Development costs for such hardware and ASICs are high. To reduce costs algorithms should be thoroughly tested and optimized before implementation at all design stages. Currently, most tests and optimizations are performed by analysis and simulation tools on workstations or super-minicomputers. Application prototypes are built only during the final design stage. However, prototyping in the earlier stages of the design can:

- Test parameters quicker
- Optimize parameters under real-time conditions
- Evaluate an algorithm's subjective qualities, for example, by listening to the actual results of the algorithm. This puts real-time constraints on the prototype.
- Prove the feasibility of implementing a design

Despite these benefits, prototyping is usually not done in the early design stages because designing dedicated prototyping hardware is time-consuming and expensive.

A rapid-prototyping alternative to designing dedicated prototyping hardware uses general-purpose hardware to minimize development costs and advanced programming tools to reduce programming time. The general-purpose hardware consists of commercial DSP processors linked together to form a powerful, heterogeneous multiprocessor.

The Prototyping Environment GRAPE-II, used in this project, simplifies programming, compiling, simulation, debugging, and testing of real-time DSP algorithms. This project demonstrated the feasibility of this approach by actually designing a functioning example. The support of GRAPEII for the run-time modification of algorithmic parameters, for example *frame_length* and *threshold,* was very useful.

It would have been time-consuming or even impossible to determine valid values for these parameters by using off-line simulation and real-time playback. For example, during real-time experiments, words starting with an "s" did not trigger the selection algorithm if the energy threshold was too high. This was not true for other words.

A top-level diagram of the beamformer application is shown in Figure 4. The input of the application is an array of eight microphone samples. The Array Transmitter sends a copy of the array samples to a set of fixed beamformers: Beam 45, Beam 90, and Beam 25. The Adaptive Beam Selection block receives the outputs from the fixed beamformers and selects the beam or beams with significant speech activity. It then sends the significant speech activity to the output of the application.

A slider bar is used to modify the parameters at run-time. There were 15 adjustable parameters. Each of the 15 parameters has a slider bar. For clarity only one of the 15 parameter-slider bars is shown.

Figure 4. Top Level of the Hierarchical Description of the Beamformer

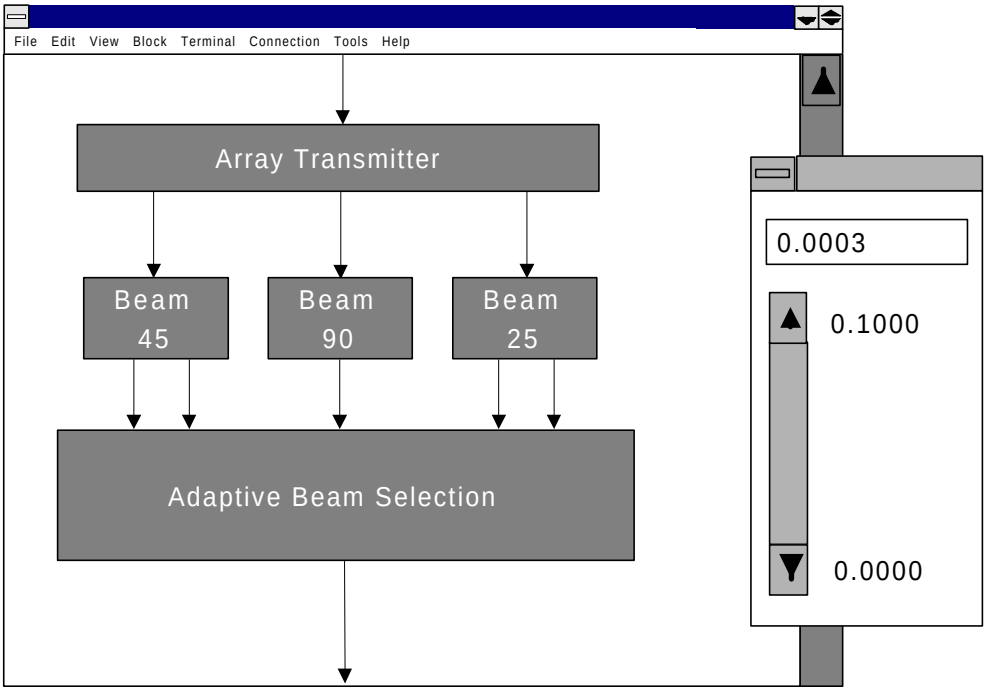


Figure 5 shows the hierarchical refinement of a single filter-and-sum beamformer directed towards 90°.

The Distribute block receives an input of samples from an array of eight microphones. It then distributes each individual sample to its corresponding filter ("Fir"). The SUM block then sums the outputs of the filters.

To map the algorithm efficiently onto the target hardware GRAPE-II requires an accurate estimate of the timing of the individual blocks. Therefore each block is executed on the target hardware, and the computation time is measured. Table 2 gives an overview of the results.

Figure 5. Hierarchical Refinement of the Beamformer Directed Toward 90°

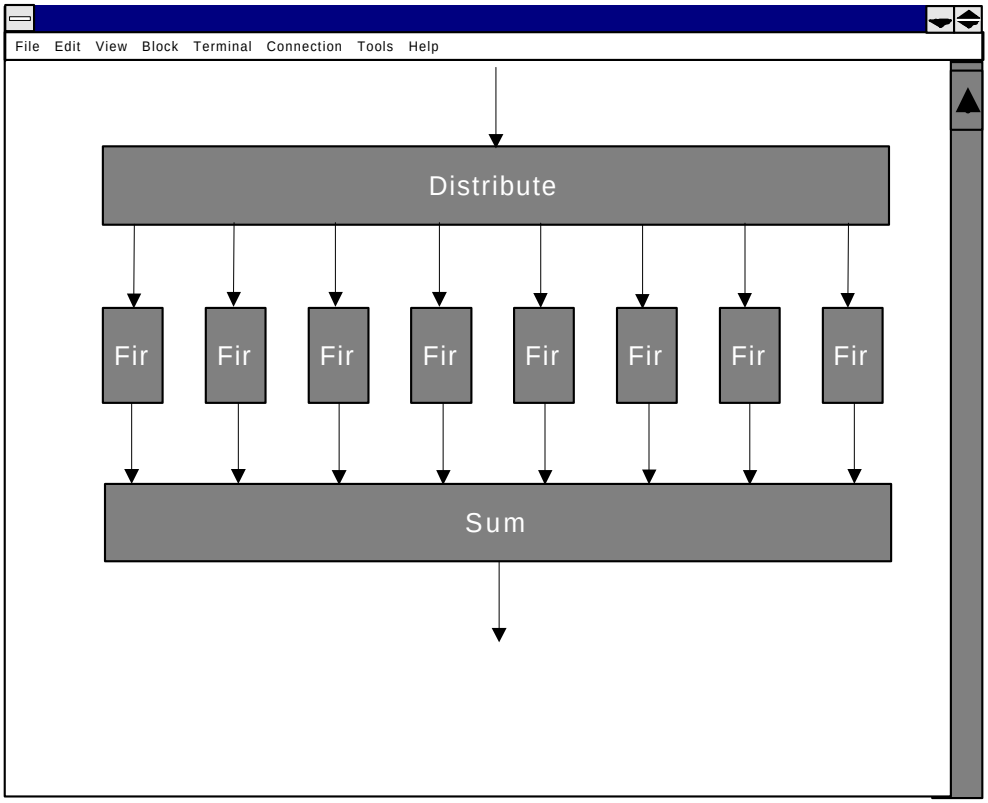


Table 2. Block Execution Time

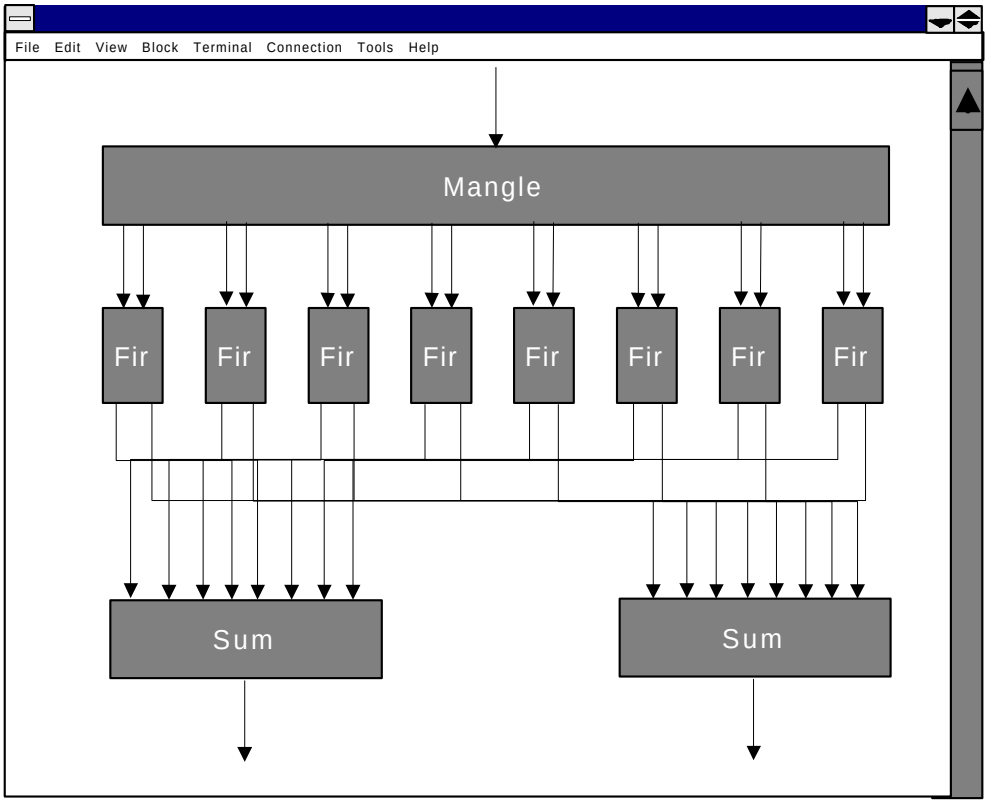
Block	#cycles	Block	#cycles
Distribute	88		
Fir90x	122	Beam90	1146
Fir25x	152	Beam25	1386
Fir45x	172	Beam45	1546
Sum	82		
Select	1206		
Triplicate	214		
Fir25x2	252	Beam25	2322
Fir45x2	292	Beam90	2642
Mangle	142		

Notes: 1) One cycle requires 50 ns.

The timing estimate for the "Select" block was inaccurate because its execution time depends on both the values of the algorithmic parameters and the input data. Other experiments showed that the real execution time was ±1825 cycles.

A quick calculation shows that the execution of the beams requires at least 7224 cycles. For a sample frequency of 8 kHz there are 2500 cycles available per sample per processor. If one processor is reserved for the "Select" block, three processors can be used for the beamforming. Since in these calculations I/O and communication overhead is not included, it's obvious that this approach will not lead to a feasible solution. Therefore, the following approach was taken: given a beam looking at a certain direction, of 45°, a beam looking at 45° is obtained with the same filters, but by reversing the microphone ordering. In this way the two beamformers can be combined for both ±45° and ±25° into two single blocks.

Figure 6. Hierarchical Refinement of the Beamformer Looking at ±45°



The result is shown in Figure 6. In this way the function-calling and status-handling overhead can be reduced. Moreover, only one set of filter coefficients for two beams had to be stored in memory. This technique allowed all the filter coefficients to be stored in internal chip memory. This was technique was necessary in order to execute a multiply||add in one instruction cycle. To further reduce the loop overhead two filter are executed in the same loop:

```
for (i=C;i<n;i++)
{
yl+= C * x1++;
y2+= C++ * x2++;
}
```

However, the C compiler generated code for the inner loop that executed in 3 instead of 2 cycles. Therefore, the assembly code had to hand optimize. These improvements resulted in faster code, as shown in Table 2.

Summary

The first experiments demonstrated the anticipated behavior of the real-time implementation of the algorithms. However, since the measurements were made in a very hostile environment (strong reverberation, non-uniform reflective surfaces, multiple noise sources, computer ventilators, and air conditioning) a slight deviation of the theoretical prediction was measured. Nevertheless, it is anticipated that in real operating environments, such as in an auditorium, system performance would improve. More experiments are necessary to fine-tune the algorithm's parameters.

This project showed the feasibility of the rapid prototyping approach, especially in determining good values for parameters, which determine subjective properties of the algorithm.

References

Lauwereins, Engels, Ade, "GRAPE-II: A System-Level Prototyping Environment for DSP Applications", IEEE Computer, February, 1995, PP 35-43.

Flanagan J.L. et.al., "Autodirective Microphone Systems", ACUSTICA, 73(1):58-91, 1991

Ward D.B., Kenedy R.A. and Williamson R.C., "Theory and Design of Broadband Sensor Arrays with frequency Invariant Far-Field Beam Patterns", 3. Acoust. Soc. Am., 97(2): 1024-1034, February 1995.

Koen Eneman, Kristof Delaet, "Bundelvorming met behulp van microfoonroosters in reele tija,', master thesis, May 1995. (in Dutch)

Piet Wauters, Stefaan Van Gerven, Marc Engels, Rudy Lauwereins, J.A. Peperstraete, "Rapid Prototyping of an Adaptive Speech Beamformer using GRAPE-II" proceedings of the International Conference on Signal Processing Applications & Technology (ICSPAT'95), October 1995.

Stefaan Van Gerven, Dirk Van Compernolle, Piet Wauters, Wim Verstraeten, Koen Eneman, Kristof Delaet, "Multiple Beam Broadband Beamforming: Filter Design and Real-Time Implementation", proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, October 1995.