REAL-TIME IMPLEMENTATION OF A FREQUENCY-DOMAIN BEAMFORMER ON THE TI C62X EVM

MARK E. ELLEDGE[1], MICHAEL E. LOCKWOOD[2], ROBERT C. BILGER[2], ALBERT S. FENG[2], DOUGLAS L. JONES[2], CHARISSA R. LANSING[2], WILLIAM D. O'BRIEN[2], AND BRUCE WHEELER[2]

[1]*Motorola, Austin, TX, USA*
elledge11@yahoo.com

[2]*University of Illinois, Urbana, IL, USA*

A minimum-variance, frequency-domain beamformer has been developed to provide speech separation for hearing-aid applications. The beamformer has the ability to cancel interfering sound sources from different spatial directions at different frequencies while preserving the desired source. A real-time system has been implemented on the Texas Instruments C62x evaluation module. Hardware and fixed-point issues (including quantization and scaling) will be discussed. A simple one-talker experiment is shown to demonstrate the effectiveness of the algorithm at attenuating unwanted sound sources. This research was funded by the Beckman Institute at the University of Illinois at Urbana-Champaign.

INTRODUCTION

A real-time implementation of a frequency-selective array processing system has been developed to provide direction-based noise cancellation. Many people with hearing loss have difficulties separating desired speech (for example, the person talking from across the table in a noisy cafeteria) from interfering speech. Array processing algorithms that remove noise from all but a single, desired direction would give back to the user the ability to listen selectively.

Once an algorithm has demonstrated the ability to effectively attenuate simulated interference sources, a real-time implementation is crucial for evaluation. Not only is the processing time for simulated data greatly reduced, but interactive performance evaluations can be made using data from real environments. Qualitative measures such as speed of adaptation and fidelity of output can be investigated. With a real-time implementation, the algorithm designer can assume the role of the end user.

Implementing an algorithm on a fixed-point DSP board requires judiciously scaling the algorithm so that the precision of the processing is not compromised. Understanding both the algorithm and the capabilities of the DSP processor are essential for retaining the most precision possible while developing an efficient implementation. System implementation is discussed from signal processing and hardware requirement standpoints. Quantization and scaling issues are also examined. The performance of the system is illustrated with tests utilizing real data.

1. SYSTEM OVERVIEW

Our beamforming system is designed to allow a user to listen to sounds from a single direction [1]-[2]. The desired direction in the current implementation is *on-axis*, or equidistant from both elements in the array [3]-[4]. Given that desired direction has a unity response, the array seeks to minimize the total output power. The user only needs to physically steer the two-element array in the direction of the desired source; the array then presents the desired signal cleanly while rejecting sounds from all other directions.

1.1 Algorithm

Our system is a two-element array processor which generates its spatial filter taps based on a minimum variance criterion. Linearly constrained minimum variance (LCMV) methods solve for the set of filter taps that best reduces output power given a set of linear constraints. The spatial filter parameters are directly computed using the statistics of the inputs. The statistical quantities, autocorrelations and cross-

correlations, of the two input signals constitute the correlation matrix **R**.

To preserve on-axis signals, we use a point constraint to force the on-axis gain to unity. Our system is now left with one degree of freedom for optimization. It can be shown that the optimal two-element LCMV filter weights (given our constraints) are

$$w_1 = \frac{\mathbf{R}_{22} - \mathbf{R}_{12}}{\mathbf{R}_{11} + \mathbf{R}_{22} - \mathbf{R}_{12} - \mathbf{R}_{21}} \quad (1)$$

and

$$w_2 = 1 - w_1 \quad (2)$$

where $\mathbf{R}_{ij}$ is the $(i,j)^{th}$ entry in **R** [4].

To allow different spatial filters at different frequencies, the minimum variance algorithm operates in the frequency domain. That is, an *N*-point FFT breaks down each input signal into *N* input channels. A minimum variance beamformer then operates independently in each channel.

The block diagram of the system is shown in Figure 1. Successive FFTs are performed on the digitized input signals to produce a frequency buffer which contains the most recent FFT results. Typical simulations compute FFTs every 16 samples.

For each bin of the FFTs, a LCMV calculation is made to calculate the best set of filter taps given the on-axis constraint. The weights are then applied to the data in the frequency buffer to perform the filtering. To save computation and prevent processing artifacts, the filter weights are only updated with every eighth FFT calculation. As with the frequency buffer creation, IFFTs are computed every 16 samples in order to generate 16 output time samples.

Note that while the input and output samples are purely real, the rest of the algorithm is comprised of complex variables and arithmetic. To implement on a DSP processor, the algorithm has to be broken into real and imaginary components. Memory space has to be allocated for the real and imaginary components of the frequency buffers, the cross-correlations, the LCMV weights, and the weighted frequency data.

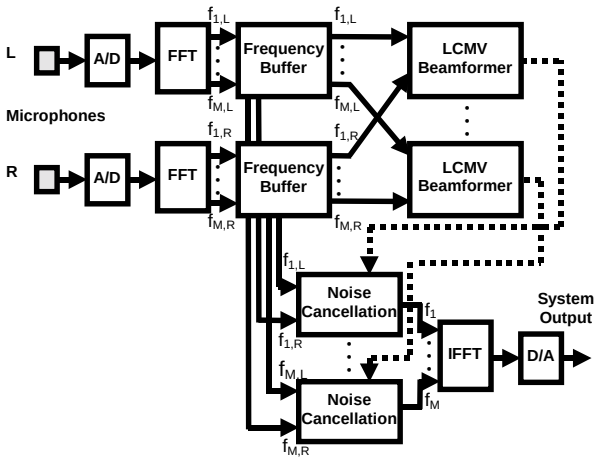

Figure 1- Array processing system

1.2 Hardware

The hardware required for the real-time implementation is shown in Figure 2. The required components include two microphones, two microphone preamps, a DSP evaluation board, a headphone amp, and headphones.

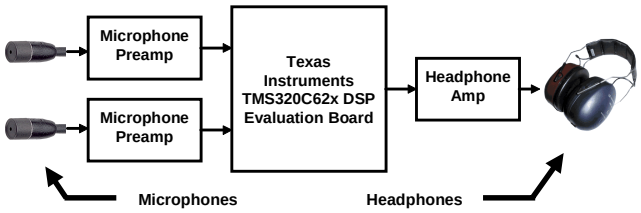

Figure 2 - Hardware for real-time system

The microphones can be mounted either in the free field on a stand or on the head of KEMAR, an acoustic mannequin. Two types of miniature microphones, omnidirectional and cardioid, have been used with the real-time system. The omnidirectional microphones in use are the Sennheiser MKE-2 Gold microphones. The Sennheiser MKE 104 microphones have a cardioid response.

The output level of microphones is much lower than nominal line-level audio signals. Typical preamps providing up to 60 dB gain are used to amplify microphone signals to line-level. The preamp used for the real-time system is the Millennia Media HV-3B with optional high-resolution gain switches. This two-channel microphone preamp has user-selectable gain from 8.5 to 60.5 dB and the two channels of the preamp are matched to 0.04 dB.

The line-level signals from the microphone preamp are

sent to the Texas Instruments (TI) TMS320C62x evaluation board for processing. The TI C62x is a high-performance fixed-point DSP processor. The processor's very long instruction word core allows up to eight parallel instructions per execution cycle; running at 133 MHz, the DSP is capable of up to 1064 millions of instructions per second. Audio input and output is provided via the evaluation board's 16-bit A/Ds and D/As.

The TI C62x evaluation board is provided with an efficient C compiler. The C compiler greatly speeds up code development, since the algorithm can be written in C instead of the DSP's assembly language. Particularly slow sections of the C implementation can then be coded in assembly for computational efficiency.

The processed output of the DSP evaluation board is then fed to a Headroom Home headphone amp, which can drive two pairs of Sennheiser HDA 200 audiometric headphones. The Sennheiser headphones are large over-the-ear headphones which help seal out outside sounds. Reducing the intensity of outside sounds allows the user to attend only to the processed output and not the original sounds in the room.

2. FIXED-POINT AND QUANTIZATION

Fixed-point DSP processors are faster and cheaper than their floating-point counterparts. While simulations are predominantly run in floating point, real-time implementations often require the extra speed that fixed-point processors can provide. In many applications, the transition from floating point to fixed point is not a trivial one.

The fixed-point issues addressed here are divided into two sections. First, the input/output and weight application elements are examined. The reasoning behind the scaling of the weights and the weight calculation itself is discussed second.

2.1 I/O and weight application

The A/D and D/A converters on the evaluation board have 16-bit resolution. To maximize precision, the input data should be assumed to be close to full scale. The processing performed in the DSP should retain as much resolution as possible and then return the appropriate 16 bits to the D/A. All instances of right-shifting should be postponed as late as possible in the processing chain.

Since a 256-point FFT will have a maximum possible gain of 256, the input data would need to be right-shifted eight bits before a 16-bit, 256-point FFT to prevent overflow within the FFT. To avoid throwing away half of our precision, a 32-bit FFT was used. The data from the A/D is now left-shifted by eight bits prior to the FFT, preserving all of the initial resolution of the input signal.

The LCMV weights are 32-bit quantities and have been left-shifted by 23 bits; the subsequent section explains why this particular scaling was chosen. Multiplying the 32-bit FFT data with the 32-bit LCMV weights results in a 64-bit result. The execution of this complex multiplication is performed with full 64-bit precision.

The 64-bit weighted frequency data must now be converted to a time signal. After right-shifting by eight bits (to prescale for the IFFT), the upper 32 bits of the frequency data are stored. The weighted frequency data is then converted to a 32-bit time sequence. The upper 16 bits are passed along to the D/A for output.

2.2 Weight calculation

In order to calculate the filter taps, the elements of **R** must be estimated. Calculating each element of **R** entails multiplying 32-bit frequency results together. The product is calculated with full 64-bit precision and then the top 40 bits are taken.

Equation (1) illustrates that the calculation of the weights requires the division of two quantities; these quantities are a simple function of the elements of **R**. The numerator and denominator are calculated as 40-bit quantities. Now the numerator and denominator must be scaled such that the calculated weights have the highest possible precision.

The real and imaginary parts of the weights can take on values above and below unity. In fixed-point implementations, accurately representing the weights requires scaling. It can be shown in a one-talker simulation that the imaginary component of the weights can rise as high as 250 with a sensor spacing of 14.4 cm; a conservative approach would allow the weights eight bits of headroom above the binary point. Thus if the weights are 32-bit signed elements, the binary point can be placed 23 bits up.

Even though the scaling of the weights has been determined, the problem of implementing the scaling still persists. One approach would be to left-shift the numerator by 23 bits prior to the division. However, if the numerator is larger than 16 bits, the numerator would overflow. Conversely, the denominator could be

right-shifted by 23 bits prior to the division but that obviously compromises the precision of the calculation and introduces potential divide-by-zero problems.

A good solution to the problem is to combine the two approaches into a variable scaling approach. First check how many bits of headroom x are left in the numerator. If x is equal to or greater than 23, left-shift by 23 bits and continue on to the division. If x is less than 23, left-shift the numerator by x and right-shift the denominator by (23 - x). Finally, the value of the shifted denominator is checked; if the denominator is zero, both weights are set to a nominal value of 0.5.

3. RESULTS

A single-talker test will be discussed to illustrate the performance of the real-time system. Speech materials were recorded every 5° around the microphone array. The materials consisted of 14 s of male speech from a loudspeaker. Data were taken with both the omnidirectional and cardioid microphones; 72 recordings were made with each microphone set. We will only look at the omnidirectional data here, but the directional data has been analyzed [4].

The microphone spacing was 15.2 cm and the loudspeaker was located 91 cm from the midpoint of the linear array. The microphones are mounted in the free field on a stand. The sign convention for identifying azimuth is established as negative for the region left of the array and positive for the region right of the array.

A single loudspeaker rotates around the microphone array. The energy of the processed output is a measure of the array's performance, since the talker is considered noise for all but $\theta = 0°$. The energy of the signals at the array will serve as the reference for calculating the amount of spatial filtering.

The average input energy and the energy of the processed signal are shown in Figure 3. Notice that the energy has been reduced for all angles except 0° and 180°. Clearly the algorithm was able to cancel sources off-axis. Note that sources from 180° were not attenuated since they are indistinguishable from those that originate from 0°.

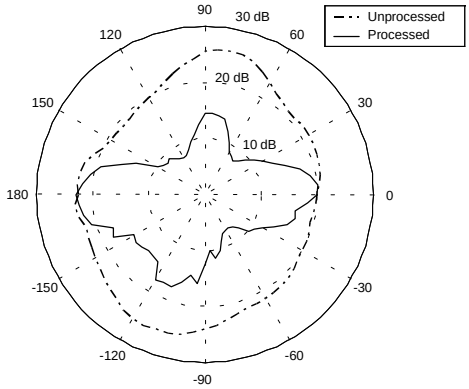

Figure 3 - Average input and processed signal energies

Taking the difference of the two curves in Figure 3, the total energy reduction can be calculated as shown in Figure 4. The plot simply shows that the array is capable of cancelling out off-axis sources from all but the front and back 20° windows. Note that the strongest level of attenuation is 14.3 dB at 60°.

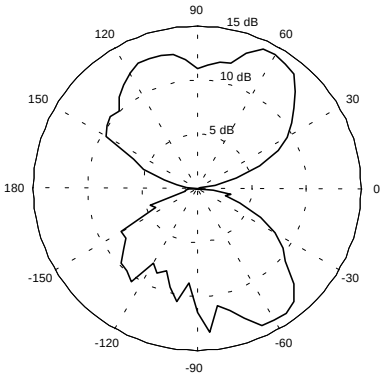

Figure 4 - Energy reduction due to array processing

Ideally, the processing algorithm can attenuate a single off-axis source from the left or right equally well. However, Figure 4 shows an uneven energy reduction from -90° to -150° that is not present on the other side of the array (i.e., from 90° to 150°). Additionally, the response from 180° to 90° should be identical to that from 90° to 0°. While that behavior is basically obeyed on the right side of the array, the left side of the array violates it.

Symmetry violations are most likely due to multipath variations. Moving the loudspeaker throughout the room creates a room response that varies with angle. Note that this test shows that the array processor is able to attenuate interfering sources with data from real microphones. However, the previous data were collected in only one room, while using only one sample of each microphone set. Clearly, a much more rigorous study of the algorithm's performance in real environments would be needed to make solid conclusions.

4. CONCLUSIONS

The real-time system developed can significantly reduce off-axis noise while preserving the on-axis source. A striking feature of this algorithm is that on-axis speech sounds very natural. Adaptation to the signal environment is exceptionally quick, since the algorithm computes new optimal weights every eight processing blocks.

The scaling of the algorithm is designed to maximize the resolution of the calculations within a fixed-point context. Lower resolution implementations compromise the algorithm's ability to attenuate interfering sources as well as the underlying noise floor. Unfortunately, carrying a high level of precision throughout the processing raises the computation complexity. The 64-bit complex multiplications, which enable the weighting procedure to incur no additional quantization, are implemented in pure assembly in order to reduce the considerable computational load.

The moving-loudspeaker test verifies the system's ability to remove off-axis sources. Extending this test to include multiple rooms, sound sources, and microphone batches would yield a richer dataset for analysis. However, it is important to not let engineering tests completely overshadow human subject tests, since helping people distinguish speech in noisy environments is the very objective of this research.

REFERENCES

[1] M.E. Lockwood, D.L. Jones, R.C. Bilger, M.E. Elledge, A.S. Feng, M. Goueygou, C.R. Lansing, C. Liu, W.D. O'Brien Jr., and B.C. Wheeler, "A minimum-variance frequency domain algorithm for binaural hearing aid processing", presented at 138th Meeting of the Acoustical Society of America, Columbus, OH, 1999.

[2] M.E. Lockwood, "Development and testing of a frequency domain minimum-variance algorithm for use in a binaural hearing aid," M.S. thesis, University of Illinois, Urbana, IL, 1999.

[3] M.E. Elledge, M.E. Lockwood, R.C. Bilger, M. Goueygou, D.L. Jones, C.R. Lansing, W.D. O'Brien Jr., B.C. Wheeler, "A real-time dual microphone signal processing system for hearing aids," presented at 138th Meeting of the Acoustical Society of America, Columbus, OH, 1999.

[4] M.E. Elledge, "Real-time implementation of a frequency-domain array processor," M.S. thesis, University of Illinois, Urbana, IL, 2000.