Implementation of a broad band multipoint acoustic noise canceller

A.C.P. van Meer, J. Garas, P.C.W. Sommen
Eindhoven University of Technology
P.O.Box 513; 5600 MB Eindhoven
The Netherlands
Tel: +31 40 2473312; fax: +31 40 2448375
email: a.c.p.v.meer@ele.tue.nl

J.L. van de Laar
Prodrive B.V.
P.O. Box 28030; Eindhoven
The Netherlands
Tel: +31 40 2676212; fax: +31 40 2676201
email: ljl@prodrive.nl

Abstract— *This paper describes a real-time realization of a broad band Multi Point Acoustic Noise Canceller (MPANC) demonstration setup using adaptive filters that employ the filtered-x NLMS algorithm. In order to cover the whole frequency range the used filters are long and complex. Due to this complexity, real-time implementation on a sample-by-sample basis is not possible and block processing techniques, such as block frequency domain adaptive filters, have to be used. Several optimized versions of the filtered-x algorithm in frequency domain are given and the different implementations are compared.*

I. THE MPANC PROJECT

In the MPANC project, subsidized by the Dutch Ministry of Economic Affairs, the Signal Processing group of Eindhoven University of Technology and Prodrive B.V. co-operate in order to develop a broad band Multi Point Acoustic Noise Cancellation demonstration system. The project aims at developing a real-time demonstration system that cancels noise signals at a listeners' ears to enhance music quality transmitted from another sound source (see Fig. 1). In the present paper we discuss the MPANC project using adaptive filters that employ the so called filtered-x algorithm. For this setup two microphones (M_L and M_R) are placed near the listeners' ears who wants to listen to a music signal s that is disturbed by a broadband noise source x, played through the primary loudspeaker L_P. Via two unknown acoustic paths $\underline{\mathbf{h}}_{PL}$ and $\underline{\mathbf{h}}_{PR}$ this noise x enters the microphones M_L and M_R respectively. The same noise source x is fed to two secondary speakers L_L and L_R through two adaptive filters $\underline{\mathbf{W}}_L$ and $\underline{\mathbf{W}}_R$. These adaptive filters are updated to minimize the residual signals (r_L and r_R) at the microphone positions, leaving only the (desired) music signal s present. More precisely, in this system the filtered-x algorithm controls two adaptive filters $\underline{\mathbf{W}}_L$ and $\underline{\mathbf{W}}_R$. The impulse response $\underline{\mathbf{W}}_L$ is updated in real-time by the algorithm in such a way that the acoustic paths $\underline{\mathbf{h}}_{LL}$ (between the secondary loudspeaker L_L and microphone M_L) and $\underline{\mathbf{h}}_{LR}$ (between the secondary loudspeaker L_L and microphone M_R) together with $\underline{\mathbf{W}}_L$ cancel out the acoustic path $\underline{\mathbf{h}}_{PL}$. The equivalent holds for $\underline{\mathbf{W}}_R$. All together the (broadband) noise x is cancelled (in real-time) and the music signal s is undistorted at the positions of the microphones M_L and M_R. The broadband Multiple Point Acoustic Noise Canceler developed in this project forms the basis for different other research projects carried out in the Signal Processing group such as the phantom sound source project [3]. Within sound reproduction a phantom sound source is a virtual sound image which can be utilized in many applications like stereo base widening, multimedia, and virtual reality engines. Because of the acoustic environment, the used filters are long and complex for the noise canceler to cover the audio frequency range. Therefore very efficient real-time algorithms are needed. To give some idea about the processing complexity we note that MPANC demonstration system consists of 2-input 2-output filtered-x algorithms that have to be implemented. Each of these two filtered-x algorithms implement three convolutions and one update. Two of these convolutions (secondary paths) are typically of order 250-500 coefficients. The third one is an adaptive filter which typically consists of 500-1000 coefficients and all of these coefficients have to be updated every update period. The Signal Processing group has developed an efficient block frequency domain algorithm of the filtered-x algorithm that meets these requirements [2]. The MPANC project is implemented on a TIMEX system of Prodrive B.V. consisting of a multiprocessor environment with four TMS320C40 DSP's working on a 44,1 kHz sampling rate.

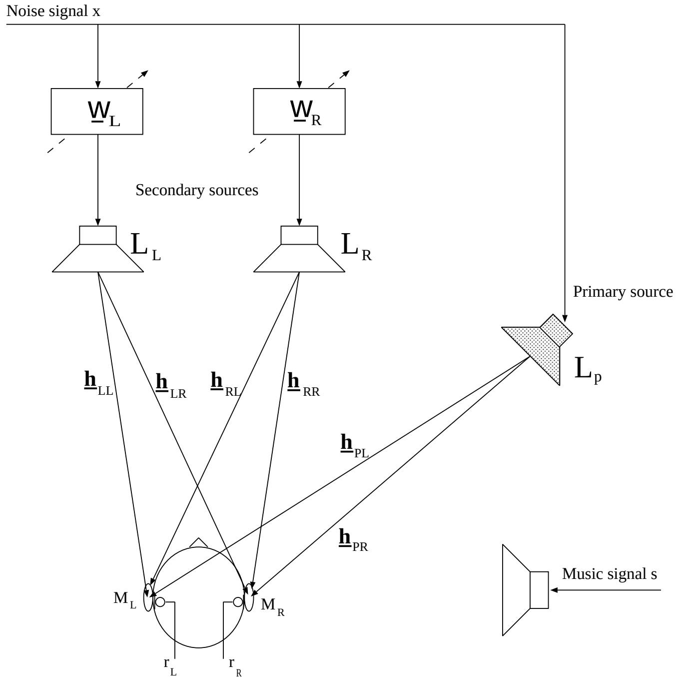

Fig. 1. MPANC demonstration setup

II. Filtered-x algorithm

Fig. 2 models a simplified (single point) version of the system shown in Fig. 1, with one primary ($\underline{\mathbf{h}}_p$) and one secondary ($\underline{\mathbf{h}}_s$) acoustic paths. In the sequel the music signal s, that in our demonstration setup will be added to the microphone ('**MIC**'), is left out for simplicity. The coefficients of the adaptive filter $\underline{\mathbf{w}}$ are updated by an algorithm that is based on the Normalized Least Mean Square (NLMS) algorithm [1]. This algorithm updates the adaptive weight vector $\underline{\mathbf{w}}$ in the negative direction of the gradient vector $\underline{\nabla}$. As can be seen in Fig. 2, the output signal of the adaptive filter $\hat{e}$ is first filtered by the secondary acoustic path $\underline{\mathbf{h}}_s$ before it is added in the microphone. For this reason, we need a slightly different algorithm: the so-called filtered-x algorithm [1]. The update part of this algorithm uses a filtered version x_f instead of the input signal x itself. This is achieved by filtering x by an estimate of the secondary acoustic path $\hat{\underline{\mathbf{h}}}_s$ as shown in Fig. 2.

III. Block Frequency Domain Adaptive Filter (BFDAF)

Because of the system complexity of the MPANC setup, real-time realization that covers the whole audio frequency range is almost impossible, with the current technology, if carried out on a sample by sample basis. Therefore, block processing techniques have to be used. Fig. 3 shows the BFDAF of the single point model shown in Fig. 2. This figure is almost equivalent to the original BFDAF implementation. The only difference is that the input signal is filtered, in the box 'filtered-x', before it enters the update part of the algorithm.

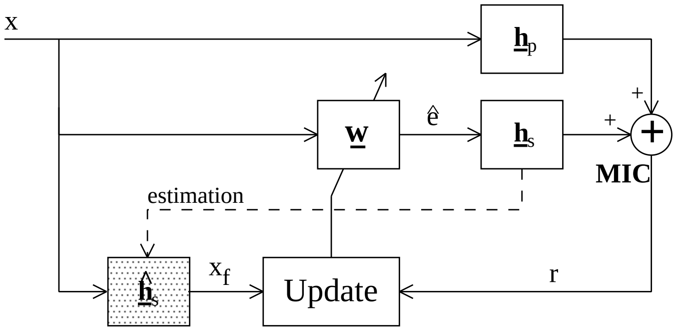

Fig. 2. Single point system model.

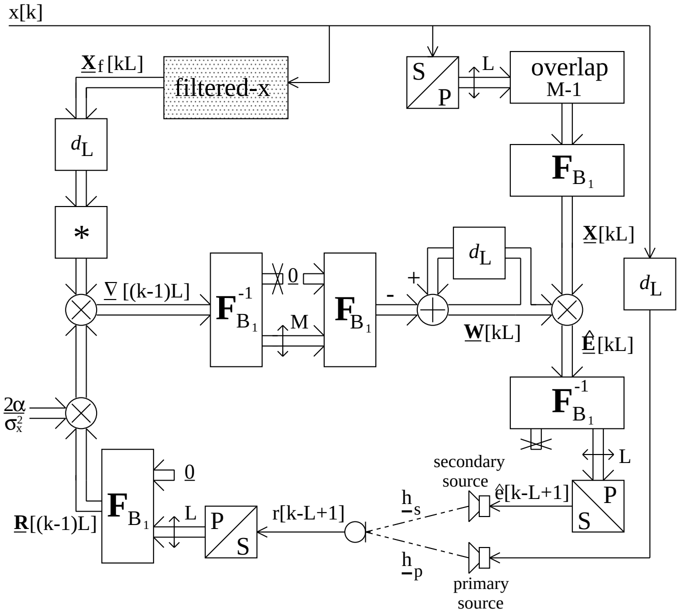

Fig. 3. BFDAF algorithm.

Working on a block basis is reflected in the fact that the whole implementation is performed on blocks of data rather than on samples. Each block contains L new samples, with $L \geq 1$. The input signal samples are converted from serial to parallel (box 'S/P') while the output samples are converted again from parallel to serial (box 'P/S'). A direct consequence of this block processing approach is a processing delay of L samples; it takes L samples before the first block of $\hat{e}$ can be produced. On the other hand, this processing delay has to be compensated, for synchronization reasons, in the primary source path (box 'd_L'). The right-hand part of Fig. 3 performs the (linear) *convolution.* This operation is broken down into a sequence of convolutions between two ‘finite’ sequences of length B_1. The ‘infinite’ input sequence is first divided into overlapping blocks. The overlap of $M-1$ samples is needed because the M adaptive weights have first to be shifted in the sequence of B_1 samples. On the other hand, performing the convolution in frequency domain produces a cyclic convolution result. Therefore the overlap is used to obtain a linear convolution result from this cyclic one. Together with the fact that every block contains L new samples, the chosen block length equals $B_1 = M + L - 1$. As depicted in Fig. 3, the blocks with B_1 input signal samples are transformed to the frequency domain with an FFT of length B_1 (‘$\mathbf{F}_{B_1}$’) resulting in a block that, for notational reasons, is represented by the transformed input signal vector $\underline{\mathbf{X}}[kL]$. The update of the M adaptive weights are first augmented by $B_1 - M$ zeroes to obtain blocks of length B_1, then transformed to frequency domain resulting in $\underline{\mathbf{W}}[kL]$. The cyclic convolution is now performed by element-wise multiplying the two vectors $\underline{\mathbf{X}}[kL]$ and $\underline{\mathbf{W}}[kL]$. Only the last L samples (in time) from this cyclic convolution represent the desired linear convolution result. Therefore, the result $\underline{\hat{\mathbf{E}}}[kL]$ is transformed back to time domain by an inverse FFT ($\mathbf{F}_{B_1}^{-1}$) and the correct L samples are sent to the secondary loudspeaker after going through the parallel to serial operation. The left-hand part of Fig. 3 performs a *correlation* operation. The only difference between the correlation and convolution operations is mirroring in time domain. This mirror operation can be performed in the Fourier domain by simply applying a complex conjugate (‘*’) operator. At the same time, L consecutive samples of the residual signal are collected, padded with zeros and transformed to the frequency domain. Since the correlation is performed in the Fourier domain, the result is also cyclic and the gradient vector $\underline{\triangledown}[(k-1)L]$ has to be transformed back to time domain to select the M correct gradient coefficients, as done in the convolution operation. These gradient coefficients are then augmented by zeros and the resulting vector of length B_1 is transformed to frequency domain and used to update the adaptive weights.

IV. Filtered-x and filtered-r BFDAF

The function of the ‘filtered-x’ block in Fig. 3 is to filter (convolve) the input signal $x[k]$ with, an estimate of, the A weights of the secondary path $\underline{\hat{\mathbf{h}}}_s$. When performing this convolution in the Fourier domain, using the overlap-save method, this will cost 4 extra FFT’s for the ‘filtered-x’ operation. In [2] we have shown that these four FFT’s can be reduced step by step to only one FFT. The first reduction to 3 FFT’s is obtained by using the transformed vector $\underline{\mathbf{X}}[kL]$, that is available at the right-hand side of Fig. 3, in the filtered-x operation. This resulting ‘filtered-x’ box is depicted in Fig. 4. With this the complete ‘filtered-x’ BFDAF implementation costs 3+5=8 FFT’s. The next reduction to two FFT’s stems from the fact that the convolution operation, needed for calculating the filtered-x input signal, and the correlation operation, needed for the gradient estimate, can be interchanged. The last reduction to one FFT is obtained by combining more carfully the different lengths of the FFT operations. With $B_3 = M + L + A - 2$ this ‘filtered-r’ BFDAF is given in Fig. 5. The number of FFT’s in the ‘filtered-x’ BFDAF equals 8 while only 6 FFT’s

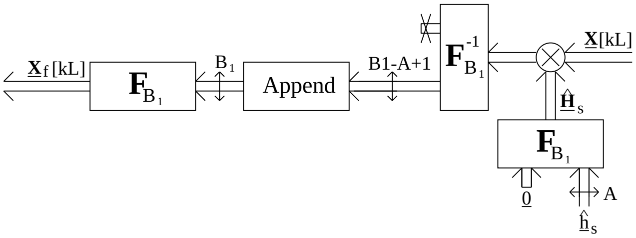

Fig. 4. Filtered-x box (3 FFT’s).

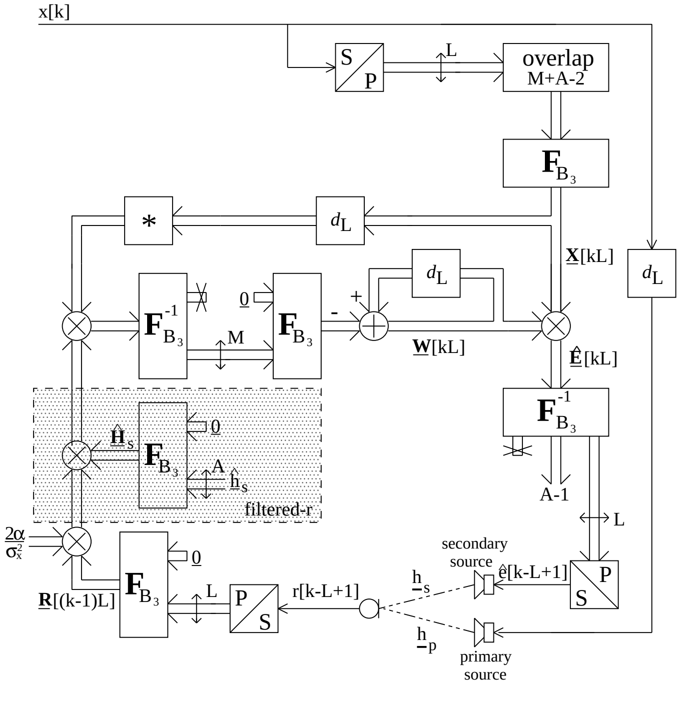

Fig. 5. Filtered-r BFDAF.

are needed in the 'filtered-r' BFDAF. However, the FFT lengths are not equal and depend on the parameters A, L and M. Therefore it is difficult to compare the complexity of the different types just by looking at the number of FFT's. Fig. 6 gives a comparison of the complexity of the two implementations described above. In this figure we have used a fixed length for the secondary impulse response $(\underline{\hat{\mathbf{h}}}_s)$; $A = 128$. Furthermore we have chosen the block length L equal to the adaptive filter length (minus one), thus $L = M - 1$. We have approximated the complexity by the total number of multiplications needed for the computation of one output sample. The total number of multiplications stems from the used FFT's of length B, each having a complexity of $(B \cdot \log_2(\frac{B}{8}) + 4)$ (for real input (FFT) and real output (IFFT)), the element-wise multiplications $(2B - 2)$ and multiplications by a real scalar (B). From this comparison it follows that, for the chosen parameter set, the 'filtered-x' BFDAF has the lowest complexity when $M < 276$ (curve 1) and 'filtered-r' BFDAF when $M > 276$ (curve 2). Thus for applications where A is relatively small compared to the adaptive filter length M (e.g. for noise cancellation applications with headphones), 'filtered-r' is preferred. However when A is larger than say half the filter length, 'filtered-x' is preferred.

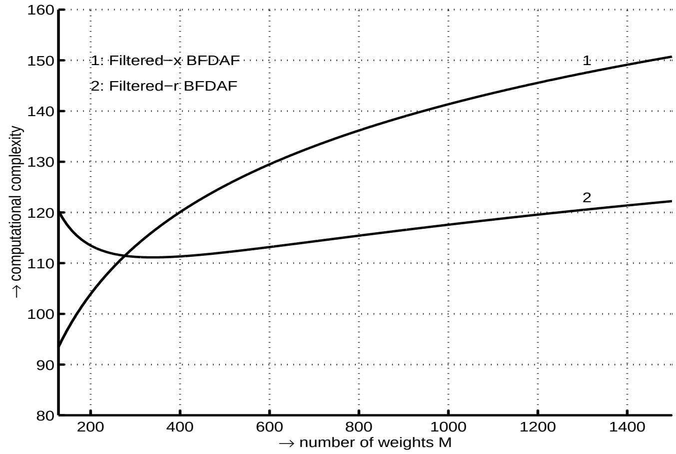

Fig. 6. Complexity comparison.

V. CONCLUSIONS

The Multi Point Acoustic Noise Canceller (MPANC) project has been described. It is shown that this system makes use of the filtered-x algorithm. The used filters are very long because of the acoustic environment. Such complex filters are best implemented in real-time using block processing techniques in a unitary transformation domain, such as Fourier domain. Two different BFDAF implementations are discussed and their complexities are compared. It is shown that the 'filtered-x' BFDAF is the most efficient when the length of the secondary source path estimate is longer than half the length of the adaptive filter. When the adaptive filter length is much longer than the secondary path estimate, 'filtered-r' BFDAF has shown to be the most efficient. The MPANC system consists of 2-input 2-output filtered-x algorithms that have to be implemented. Each of these stereo filtered-x algorithm implements three convolutions and one update. This setup is implemented on a TIMEX system of Prodrive B.V. consisting of a multiprocessor environment with four TMS320C40 DSP's working on a 44.1 kHz sampling rate.

REFERENCES

[1] B. Widrow and S.D. Stearns, *Adaptive Signal Processing*, Prentice-Hall, Inc., Englewood Cliffs, N.J., 1985.

[2] Piet C.W. Sommen et. al. *Efficient frequency domain filtered-x realization of phantom sources* Proceedings of the ProRISC/IEEE workshop 1997, pp 589-595

[3] John Garas, *Improving virtual sound source robustness using multiresolution spectral analysis and synthesis*, 105th AES convention, San Francisco, sept. 98 (http://www.ses.ele.tue.nl/signalen/persons/garas)