Efficient frequency domain filtered-x realization of phantom sources

Piet C.W. Sommen, Ronald M. Aarts, Alexander W.M. Mathijssen, John Garas, Haiyan He

Abstract— **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. Generation of such a virtual sound image using adaptive filters that employ the filtered-x NLMS algorithm is discussed. Such filters are long and complex for the virtual source to cover the whole audio frequency range. 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 derived and the different implementations are compared.**

Keywords— **Active Noise Cancellation, Filtered-x, Frequency Domain Adaptive Filters, Loudspeakers, Phantom Sound Source, Sound Reproduction**

I. Introduction

When audio signals are played by a normal stereo setup, the sound received by a listener contains both directional and distance information, from which the listener can locate the positions of the sound sources. Experience shows that when a stereo signal is reproduced by two separate loudspeakers, the perceived sound image would be between the two loudspeakers if the listener is in the center of the listening area. However, in some applications like TV and monitors, the distance between the loudspeakers has to be limited in order to obtain a compact arrangement. To give the loudspeaker base a virtual broadening, we made use of the phantom sound source concept. A phantom sound source is a virtual source which is perceived by the listener as a source at a different position than that of the real sources. The basic idea behind phantom source generation is to alter the loudspeakers' input signals in such a way that the Sound Pressure Level (SPL) at the listener's eardrums generated by the real loudspeakers is the same as that would be generated by the phantom source [4].

P. Sommen, A. Mathijssen, J. Garas and H. He are with the Eindhoven University of Technology, Building Eh 6.33, P.O.Box 513, 5600 MB Eindhoven, The Netherlands, E-mail: p.c.w.sommen@ele.tue.nl

R. Aarts is with the Philips Research Laboratories WY81, 5656 AA Eindhoven, The Netherlands, E-mail: rmaarts@natlab.research.philips.com

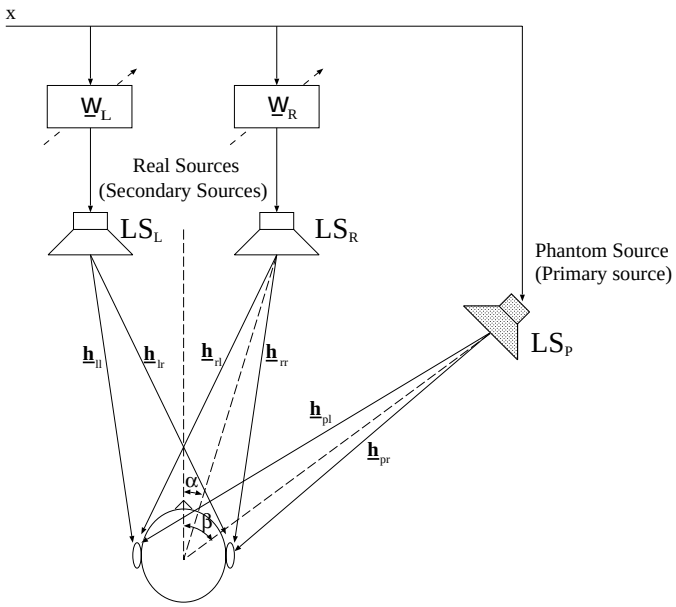

Fig. 1. Set-up for generating a phantom source.

Fig. 1 shows a situation where a phantom sound source LS_p at angle β is generated by two real loudspeakers at angle α. $\underline{\mathbf{w}}_L$ and $\underline{\mathbf{w}}_R$ are filters which are used to alter the input signals to LS_L and LS_R. The design of the filters $\underline{\mathbf{w}}_L$ and $\underline{\mathbf{w}}_R$ is the main task in generating a phantom source. These filters represent, in general, long and complex impulse responses which are also position dependent. Therefore it is difficult to obtain an analytical solution for these filters. For this reason, long adaptive filters are used which are implemented very efficiently in the frequency domain. These filters are applied as follows

(see Fig. 1): Initially three loudspeakers are used, one un-filtered at the position of the phantom source, the other two filtered by adaptive filters. White noise is played through loudspeaker LS_p while $\underline{\mathbf{w}}_L$ and $\underline{\mathbf{w}}_R$ are adapted to minimize the SPL measured at the listener's eardrums. The method described above can hence be regarded as an active noise cancellation solution. The coefficients of $\underline{\mathbf{w}}_L$ and $\underline{\mathbf{w}}_R$ are then frozen and the third loudspeaker is removed. When the two loudspeakers are then driven through the $-\underline{\mathbf{w}}_L$ and $-\underline{\mathbf{w}}_R$, only the illusion of a third source results.

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. 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. This algorithm updates the adaptive weight vector $\underline{\mathbf{w}}$ in the negative direction of the gradient vector $\underline{\nabla}$. It is shown in [1] that an estimate of this gradient vector is given by $\underline{\mathbf{x}} \cdot r$, in which $\underline{\mathbf{x}}$ is the input signal vector and r the residual signal and the update equation becomes:

$$\underline{\mathbf{w}}_{\text{new}} = \underline{\mathbf{w}}_{\text{old}} - \frac{2\alpha}{\sigma_x^2}\,\underline{\mathbf{x}} \cdot r \qquad (1)$$

where α is the adaptation coefficient which is normalized by the variance of the input signal $\sigma_x^2 = E\{x^2[k]\}$.

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 than that given by (1); 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 $\underline{\hat{\mathbf{h}}}_s$ as shown in Fig. 2.

A. Block Frequency Domain Adaptive Filtering

Because of the system complexity, real-time realization of a phantom source 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

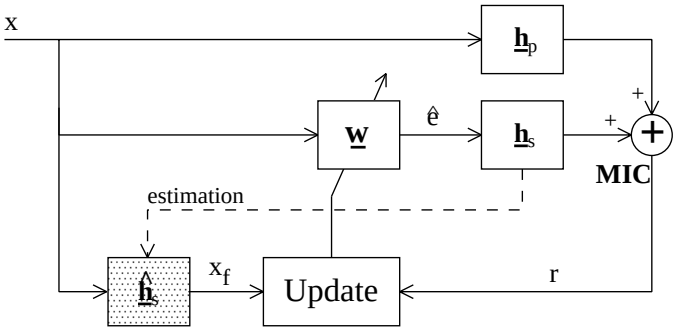

Fig. 2. Single point system model.

be used. In this work, the Block Frequency Domain Adaptive Filter (BFDAF) [3] has been utilized. In this subsection, the BFDAF will be described to introduce the notation and in subsequent sections, the filtered-x part will be discussed in more detail.

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.

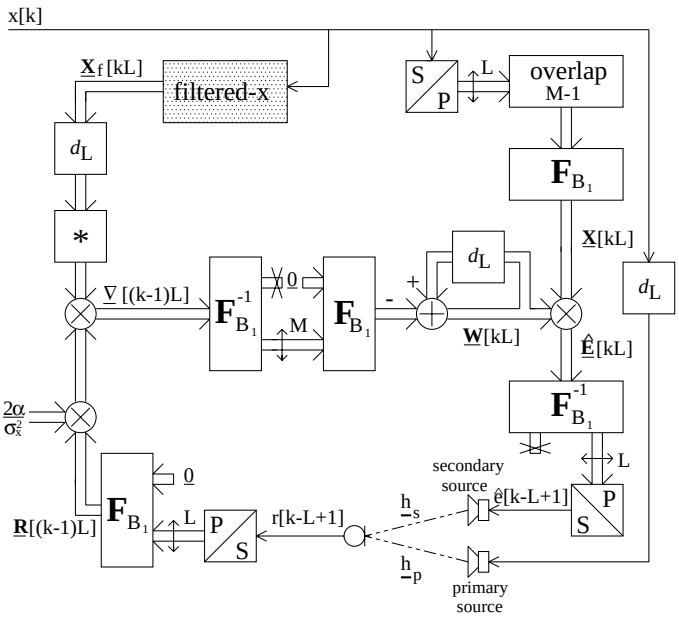

Fig. 3. Block Frequency Domain Adaptive Filter.

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 paral-

lel 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').

For a gradient update algorithm such as the NLMS considered here, two main operations have to be computed [1]:

- a (linear) *convolution* to perform the filtering of the input signal x with the M adaptive weights
- a (linear) *correlation*, between the input signal x and the residual signal r.

The first calculates the filter output, and the second calculates an estimate of the gradient $\underline{\bigtriangledown}$ that is needed for the update of the adaptive weights.

For large filter lengths M these operations can be carried out very efficiently by using a block processing approach, such as the overlap-save method [2], implemented in the frequency domain. In this case, Fast Fourier Transforms (FFTs) are used to perform the signal transformations back and forth between time and frequency. In the following the above mentioned convolution and correlation will be described when the overlap-save method is used (see Fig. 3).

The *convolution* (performed by the right-hand part of Fig. 3) of the 'infinite' input sequence (the input signal samples $x[k]$) with the 'finite' length sequence (the M adaptive weights) 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 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 $\underline{\mathbf{W}}[kL]$

$$\underline{\hat{\mathbf{E}}}[kL] = \underline{\mathbf{X}}[kL] \otimes \underline{\mathbf{W}}[kL]. \qquad (2)$$

where the symbol $\otimes$ is used to denote element-wise multiplication. 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 *correlation* operation between the 'infinite' sequence of input signal samples x and the 'finite' sequence of residual signal samples r is performed by the left-hand part of Fig. 3. 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 resulting in the vector $\underline{\mathbf{X}}_f^*[(k-1)L]$[1]. At the same time, L consecutive samples of the residual signal are collected, padded with zeros and transformed to the frequency domain resulting in the vector $\underline{\mathbf{R}}[(k-1)L]$. The, scaled, gradient is now calculated by

$$\underline{\bigtriangledown}[(k-1)L] = \frac{2\alpha}{\sigma_x^2} \cdot \left(\underline{\mathbf{X}}_f^*[(k-1)L] \otimes \underline{\mathbf{R}}[(k-1)L]\right)$$

in which the scaling is done by $\frac{2\alpha}{\sigma_x^2}$. Since the correlation is performed in the Fourier domain, the result is also cyclic and the gradient vector $\underline{\bigtriangledown}[(k-1)L]$ 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.

[1]Note that the input signal to the correlation operation, here the filtered-x signal $\underline{\mathbf{X}}_f[kL]$, needs to be delayed by d_L for synchronization reasons and the residual signal $\underline{\mathbf{R}}[(k-1)L]$ contains the same delay d_L.

B. Filtered-x in frequency domain

The 'filtered-x' operation, that is needed for the update algorithm in Fig. 3, is depicted in more detail in Fig. 4. The function of this 'filtered-x' block is to fil-

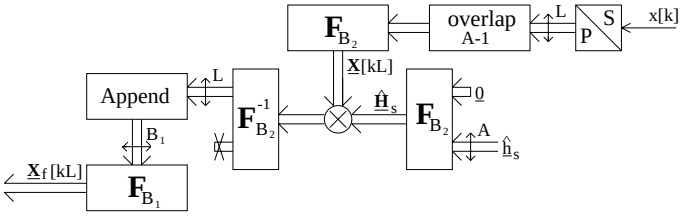

Fig. 4. Filtered-x implementation type 1.

ter (convolve) the input signal $x[k]$ with, an estimate of, the A weights of the secondary path $\underline{\hat{\mathbf{h}}}_s$. This convolution is performed in frequency domain using the overlap-save method as explained before.

Each block contains L new input signal samples while the segments have an overlap of $A-1$ samples. The resulting segments contain $B_2 = A + L - 1$ input signal samples. These are transformed to the frequency domain ($\underline{\mathbf{X}}[kL]$) and the convolution is performed by the element-wise vector multiplication with $\underline{\hat{\mathbf{H}}}_s$ which is obtained by padding the impulse response $\underline{\hat{\mathbf{h}}}_s$ with $B_2 - A + 1$ zeros. The correct L samples of the cyclic convolution have to be appended in time domain to A-1 previously calculated samples to form segments of length B_1. These are transformed to frequency domain and result in $\underline{\mathbf{X}}_f[kL]$ that used in the correlation operation described in the previous subsection.

From Fig. 4 it follows that the 'filtered-x' operation in itself costs four FFT's. In the following section we will show that these four FFT's can be reduced step by step to only one.

III. Efficient filtered-x

The reduction from four FFT's to one is done in three steps [5]. The first is reduction to three FFT's; and obtained by combining more carefully the FFT's in Fig. 3 and Fig. 4. 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 again FFT operations.

Step 1: Combining two FFT's

The first optimization step is to combine the two separate Fourier transforms that are used to calculate the transformed input signal vector $\underline{\mathbf{X}}[kL]$ of length B_1 in Fig. 3 and length B_2 in Fig. 4. The FFT $\mathbf{F}_{B_2}$ in Fig. 4 can then be left out. The result is depicted in Fig. 5.

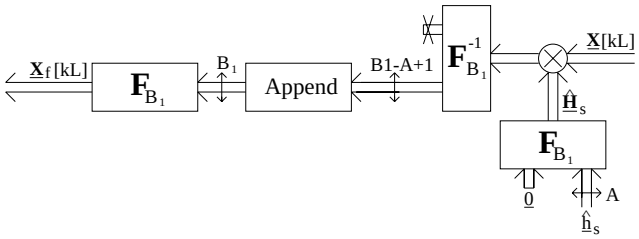

Fig. 5. Filtered-x implementation type 2.

Here we have assumed that the length A of the secondary path is smaller than the length M of the adaptive weight vector $\underline{\mathbf{W}}$ ($A < M$). By augmenting the impulse response vector $\underline{\hat{\mathbf{h}}}_s$ of the secondary path with more zeros than before a length B_1 vector is obtained that is transformed to the frequency domain resulting in $\underline{\hat{\mathbf{H}}}_s$. This vector is multiplied element-wise by the transformed input signal vector $\underline{\mathbf{X}}[kL]$ that was available in Fig. 3. The rest of the operations in Fig. 5 are exactly the same as in Fig. 4. The result is one FFT less in comparison with type 1 implementation shown in Fig. 4.

Step 2: Interchanging convolution/correlation

The second optimization step is to interchange the order of the convolution operation of the 'filtered-x' and the correlation operations. In Fig. 3, the overlap-save method has to be used in the filtered-x block since the input signal is infinitely long. Because of the cyclic nature of this convolution, the correct samples have to be selected in time domain costing one inverse FFT and one forward FFT as shown in Fig. 4 and Fig. 5.

Noting that both the impulse response (A) and the residual signal block (L) are of 'finite' length, it is more efficient to first calculate the convolution of these two 'finite' sequences. This will result in a 'finite' sequence of length $A + L - 1$. Implementing this operation in frequency domain, with Fourier transforms

of appropriate length, results in a frequency domain vector that contains no cyclic (incorrect) results. This frequency domain vector can now be used straight on in the correlation with the 'infinite' input signal as before. The result of this approach is depicted in Fig. 6. This realization can also be referred to as 'filtered-r'

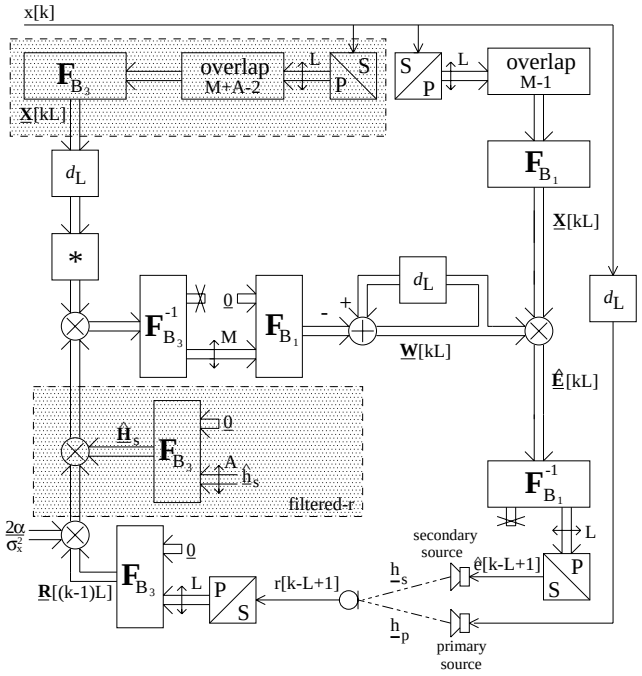

Fig. 6. Filtered-x implementation type 3.

because the 'filtered-x' operation is in fact calculated by filtering first the residual signal ('filtered-r'). No extra inverse Fourier transform is needed, as in Fig. 5. We only have to take care of a correct Fourier transform length $B_3 = M+L+A-2$. Finally we note that only two FFT's are needed in comparison with Fig. 4 and Fig. 5.

Step3: Combining two FFT's

Finally an equivalent step as described in step 2 can be used here. The Fourier transform $\mathbf{F}_{B_3}$ that is needed in Fig. 6 to calculate the transformed input signal vector $\underline{\mathbf{X}}[kL]$ can be combined with the Fourier transform $\mathbf{F}_{B_1}$ that is needed at the right hand side of the same figure for the convolution with the adaptive weight vector. The result is depicted in Fig. 7. From this figure it follows that the output of the adaptive filter produces each block $A+L-1$ correct results from which only L samples are used and $A-1$ are not used. However the total result is that we have introduced an

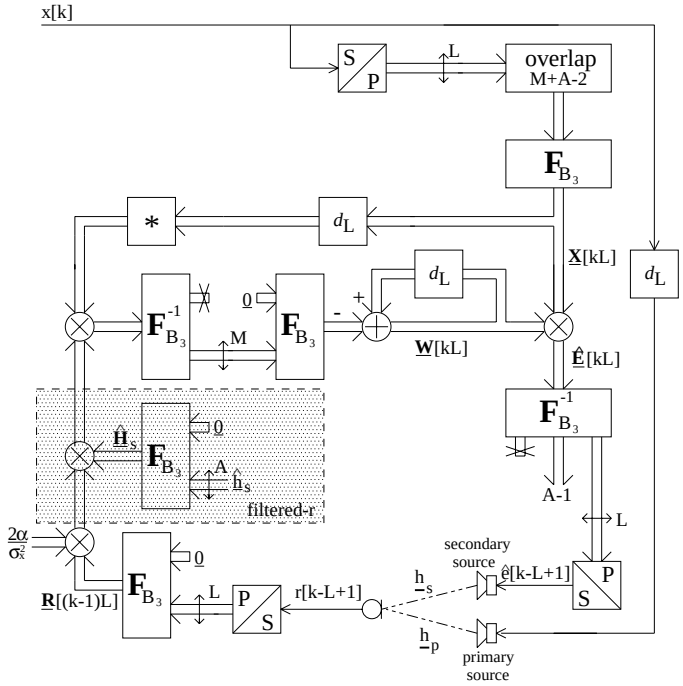

Fig. 7. Filtered-x implementation type 4.

efficient implementation of the 'filtered-x' (or 'filtered-r') algorithm in frequency domain that costs one more FFT compared to the original BFDAF.

IV. Complexity comparison

In the previous section, the number of FFT's in the filtered-x BFDAF implementations is reduced from 9 to 6. 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. 8 shows the complexity of the four implementations described above. In this figure we have used a fixed length for the secondary impulse response ($\hat{\underline{\mathbf{h}}}_s$); $A = 128$. 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 [5]. 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

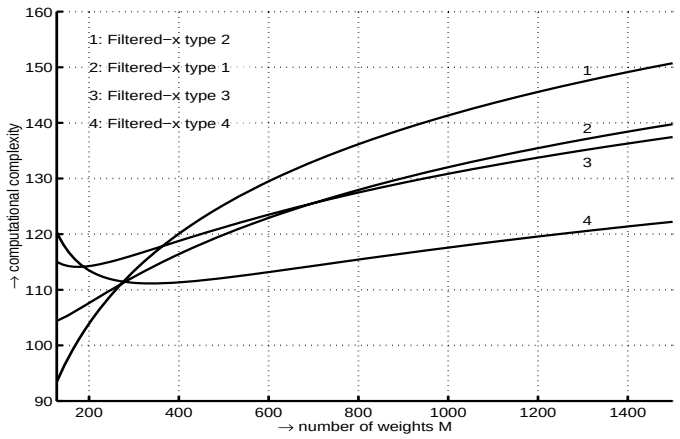

Fig. 8. Computational complexity of the implementations.

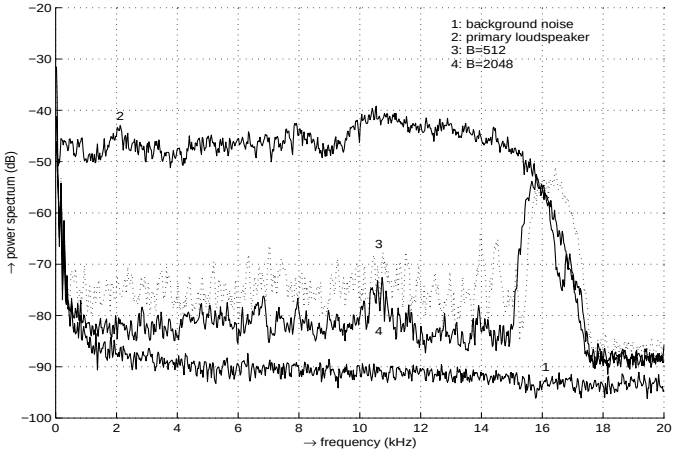

Fig. 9. Power spectrum of the background noise (curve 1), the residual signal before cancellation (curve 2) and after cancellation (curve 3, B=512 and curve 4, B=2048).

set, type 2 has the lowest complexity when $M < 276$ (curve 1) and type 4 when $M > 276$ (curve 4). Thus for applications where A is relatively small compared to the adaptive filter length M (e.g. for noise cancellation applications with headphones), type 4 is preferred. However when A is larger than say half the filter length, type 2 is preferred.

V. Simulation and experimental results

Experiments have been performed [5] on a one point noise cancellation system. The set-up for this system consists of a primary loudspeaker, a secondary loudspeaker and a microphone, all placed in an anechoic room, where the primary loudspeaker is placed further away from the microphone than the secondary loudspeaker. The estimated impulse response $\underline{\hat{\mathbf{h}}}_s$ is fixed during the noise cancellation.

Fig. 9 shows the measured power spectra of the residual signal. Curve 1 is the spectrum of the background noise. Curve 2 is the spectrum of the primary loudspeaker. Curve 3 and 4 are the spectra of the residual signal after cancellation with respectively $M - 1 = L = 256$ and 1024. The average attenuation over a frequency range from 1 to 15 kHz is 29 dB for $M = 257$ and 36 dB for $M = 1025$.

VI. Conclusions

Generation of a virtual sound image using adaptive filters that employ the filtered-x LMS algorithm has been discussed. Such complex filters are best implemented in real-time using block processing techniques in a unitary transformation domain, such as Fourier domain. Several implementations of the filtered-x block frequency domain adaptive filter are discussed and their complexities are compared. Simulations and experiments have shown that type 2 implementation 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, type 4 has shown to be the most efficient.

References

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

[2] A.V. Oppenheim and R.W. Schafer, *Digital Signal Processing*, Prentice-Hall, Englewood Cliffs, N.J., 1975.

[3] P.C.W. Sommen, *Adaptive Filtering Methods*, Ph.D. thesis Eindhoven University of Technology The Netherlands, June 1992, ISBN 90-9005143-0.

[4] R. Aarts, H. He, P. Sommen, *Phantom sources applied to stereo-base widening using long frequency domain adaptive filters* Proceedings of the ProRISC/IEEE workshop 1996, pp 49-54

[5] A.W.M. Mathijssen, *Generation of phantom sound sources with Block Frequency Domain Adaptive Filtering*, Master thesis of Eindhoven University of Technology, September 1997.