ABSTRACT
This paper considers the implementation of an Active Noise Control (ANC) system over a network of distributed acoustic nodes. Single-channel nodes composed of one microphone, one loudspeaker, and a processor with communication capabilities have been considered. An equivalent solution to the Multiple Error Filtered-x Least Mean Square algorithm (Me-FxLMS) has been chosen because it is a widely used algorithm in ANC systems with centralized processing. The proposed algorithm has been implemented with block-data processing as commonly happens in practical systems. Furthermore, the algorithm works in the frequency domain and with partitioning of the filters for improving its efficiency. Therefore, we present a new formulation to introduce a distributed algorithm based on the Me-FxLMS together with an incremental collaborative strategy in the network. Results demonstrate that the scalable and versatile distributed algorithm exhibits the same performance than the centralized version. Moreover, the computational complexity and some implementation aspects have been analyzed.

Index Terms—Distributed Networks, Active Noise Control, Filtered-x Least Mean Square,
Fig. 1. Schemes of (a) a centralized ANC system, (b) a distributed ANC system with single-channel nodes.

| \(I\) | Number of reference signals |
| \(K\) | Number of error signals (monitoring sensors) |
| \(L\) | Length of the adaptive filters |
| \(M\) | Length of the FIR filters that model the acoustic paths |
| \(j\) | Number of secondary sources (actuators) |
| \(s\) | M-length estimation of the acoustic path that links the \(j\)th secondary source with the \(k\)th monitoring sensor |
| \(w\) | Coefficients of the adaptive filter of length \(L\) during the \(n\)th block iteration |
| \(W\) | FFT of size \(2B\) of the \(f\)th partition of the coefficients of the adaptive filter \(w\) during the \(n\)th block iteration |

Table 1. Notation of the description of the algorithms

The filter coefficients are updated in the frequency domain by calculating the correlations between the reference signal \(x[n]\) that is filtered through the estimated secondary path \(S^p\), \(V[n]\), and the error signal, \(e_B[n]\). To this end, the following operations are performed

\[
V[n] = \sum_{p=1}^{P} S^p \circ X[n - p + 1],
\]

\[
\hat{\mu}^f[n] = E[n] \circ V[n - f + 1]^{*},
\]

where

\[
E[n] = FFT[0_B \ e_B[n]].
\]

The update of the coefficients of each partition of the adaptive filter at the \(n\)th block iteration is calculated as follows

\[
W^f[n + 1] = W^f[n] - \mu FFT(\phi^f[n] 0_B),
\]

where \(\mu\) is the step-size parameter, and the vector \(\phi^f[n]\) corresponds to the first B samples of the 2B-IFFT of the partition \(\hat{\mu}^f[n]\)

\[
\text{IFFT}(\hat{\mu}^f[n]) = [\phi^f[n] \ \overline{\phi}^f[n]].
\]

Equations (3)-(6) are performed for each partition \((f=1,\ldots,F)\).

2.2. The FPBFxLMS for a distributed ANC system

The proposed distributed ANC system is composed of \(N\) single-channel nodes, and therefore, \(N\) error sensors and \(N\) secondary sources. Now, there exists a global state network, which is defined by \(N\) adaptive filters, one of each node. The global network adaptive filter, \(W[n]\), can be defined as

\[
W[n] = [W_1[n], W_2[n], \ldots, W_N[n]].
\]
\[ W_k[n] = [W^1_k[n], W^2_k[n], \ldots, W^F_k[n]], \quad (8) \]

where \( W[n] \) is a \([2B \times FN]\) matrix composed of the concatenation of the adaptive filter of each node. Matrix \( W^j_k[n] \) of size \([2B \times F]\) is the adaptive filter of the \( k \)th node, and vector \( W^j_k[n] \) of size \( 2B \), is the \( j \)th partition of the adaptive filter in frequency domain of the \( k \)th node. For calculating the output signal, each node takes its filters from the global filters when they are completely adapted. Hence, the \( k \)th node uses \( W^F_N[n] \) to calculate its output signal like in (1). Moreover, we define

\[ V_k[n] = [V_1[k][n], V_2[k][n], \ldots, V_N[k][n]], \quad (9) \]

where \( V_k[n] \) is a matrix of size \([2B \times FN]\) of the \( k \)th node. It is composed of the concatenation of the reference signal filtered through all the secondary paths that links the \( j \)th loudspeaker with the \( k \)th microphone for \( j = 1, \ldots, N \). Matrix \( V_j[k][n] \) of size \([2B \times F]\) is calculated as stated in (2) for each secondary path and each partition

\[ V_j[k][n] = [V^1_j[k][n], V^2_j[k][n], \ldots, V^F_j[k][n]]. \quad (10) \]

Furthermore, each node takes its error signal from its sensor to form its error vector as in (4). Then, each node replicates its error vector \( FN \) times forming the matrix \( E_k[n] \) of size \([2B \times FN]\). Equation (3) is redefined for the \( k \)th node as

\[ \hat{\mu}_k[n] = E_k[n] \odot V_k[n]^* \quad (11) \]

Matrix \( \hat{\mu}_k[n] \) of size \([2B \times FN]\) is used at each node to calculate the adaptation matrix of the node, \( \Psi_k[n] \), as

\[ \text{IFFT}(\hat{\mu}_k[n]) = [\hat{\phi}_k[n], \hat{\phi}_k[n]], \quad (12) \]

\[ \Psi_k[n] = \text{FFT}([\hat{\phi}_k[n], 0_{B \times FN}]), \quad (13) \]

where \( \hat{\phi}_k[n] \) and \( 0_{B \times FN} \) are matrices of size \([B \times FN]\). Moreover, the operators FFT and IFFT perform direct and inverse fast fourier transforms of size \( 2B \) of each column of the matrices involved. Finally, each node calculates its own estimate of the global adaptive filters using the global estimate of the previous node, and its own adaptation matrix \( \Psi_k[n] \). The global adaptive filters are adapted at the \( k \)th node as

\[ W_k[n + 1] = W_{k-1}[n] - \mu \Psi_k[n], \quad (14) \]

Once all the nodes have finished the actualization of the filters, the global updated vector \( W^F_N[n] \) is disseminated to the rest of the nodes for the next iteration. Moreover, note that \( W_0[n] = W_0[n - 1] \).

### 3. IMPLEMENTATION ASPECTS

The sampling rate \( (f_s) \) and the block size \( (B) \) are two important parameters when thinking about a real-time implementation of a distributed ANC system. \( B \) describes the number of transferred discrete-time samples per iteration and thereby determines the latency of the algorithm. The latency is the time spent from when the input-data buffer is filled up until this data buffer is processed and sent back to the output-data buffer. We refer to the time spent to fill up the input-data buffers as \( t_{buff} \), and is defined as \( B/f_s \). The choice of these parameters is crucial for the performance of the system because there are two conditions that must be satisfied:

- **The real-time condition.** The algorithm works in real time if \( t_{proc} < t_{buff} \), where \( t_{proc} \) is the execution delay. In a centralized ANC system \( t_{proc} \) is the processing delay of the algorithm. In a distributed ANC system, it also includes the delays of transmitting the global network state between the nodes. However, each node process the algorithm simultaneously except the addition of the global network state of the previous node. Moreover, as we consider single-channel nodes, each node has to process less operations than a multichannel centralized system.

- **The causality condition.** The algorithm has to satisfy the condition \( t_{buff} + \tau_s < \tau_n \) [12], where \( \tau_s \) is the maximum delay of the secondary paths that join the actuators with the error sensors, and \( \tau_n \) is the minimum delay of the paths that join the noise source with the error sensors. This condition guarantees the causality of the system.

It is obvious that the time \( t_{proc} \) increases with the number of nodes. If the time \( t_{proc} \) increases, the time \( t_{buff} \) must increase in order to satisfy the real-time condition. It means that, with a fixed \( f_s \), the block size must be increased. Section 4.1 will show that when \( B \) increases, the convergence performance of the algorithm gets worse, until the causality condition is no longer satisfied. Therefore, it seems that some kind of trade off between these parameters must be considered in order to satisfy both conditions.

### 4. RESULTS

Some experiments were performed to validate the distributed ANC system. In a first stage, both the noise reduction and the convergence performance of the distributed ANC system are evaluated and compared with the centralized ANC system. In a second stage, we evaluate and compare the computational complexity of both ANC systems.

#### 4.1. Simulation Results

In this section, some simulation results are presented to validate the performance of the FFBfLMS algorithm in a distributed network with a ring topology and an incremental approach. The simulations have been carried out using real acoustic channels between microphones and loudspeakers sampled at 2kHz. These channels have been measured in a listening room. Some examples of the impulse responses of this listening room are available [13]. We have considered a zero-mean Gaussian random noise with unit variance as the
disturbance noise. Furthermore, a block-size of $B = 512$ and a filter length of $L = 1024$ have been considered. This means that two partitions are carried out. In order to evaluate the performance of the algorithm, we define the instantaneous Noise Reduction ratio at the $k$th node as

$$NR_k[n] = 10 \log_{10} \left( \frac{s_k^2[n]}{d_k^2[n]} \right),$$

(15)

where $e_k[n]$ and $d_k[n]$ are the signals measured at the $k$th microphone with and without the ANC operation, respectively. Moreover, the power of these signals have been estimated using an exponential windowing.

First, we compare the noise reduction of a square centralized ANC system with a 1:4:4 configuration and a distributed ANC system with 4 single channel nodes. Fig.2 shows the noise reduction of both the centralized and the distributed implementations of the FPBFxLMS algorithm. Fig.2 illustrates the results for the microphone with best and worst performance in the centralized implementation, and the node with the best and worst performance in the distributed implementation. As expected, the distributed implementation has exactly the same results than the centralized implementation in terms of convergence speed and final residual noise.

Another important property related with the causality condition is the stability limit. In the literature, some contributions have studied the convergence behavior of the block filtered-x LMS algorithm (BFxLMS). In [14], the maximum $\mu$ parameter that leads to the fastest convergence rate was derived as

$$0 \leq \mu < \frac{1}{B\lambda_{\text{max}}}$$

(16)

where $\lambda_{\text{max}}$ is the maximum eigenvalue of the filtered reference signal autocorrelation matrix $R_{vv}$, defined as $R_{vv} = E[VV^T]$. Therefore, the convergence performance of the algorithm depends on the statistics of the reference signal, the acoustic paths, and the block length $B$. For the same reference signal, the step-size parameter $\mu$ depends on $B$, so the maximum $\mu$ value increases by reducing the size of $B$, and, consequently, the convergence speed is improved by reducing $B$. However, as commented in section 3, the size of $B$ is also limited by the real-time condition. Therefore, there is a minimum value of $B$ for a given configuration that assures the real-time condition and maximum convergence speed.

Fig. 3 illustrates the convergence behavior of the worst node in a distributed network of four nodes when the size of $B$ changes between 256 and 2048. As expected, it shows that the algorithms converge faster with a smaller block size, $B$. As these results show, the maximum $\mu$ is more or less doubled when $B$ is halved. This fact can be explained from (16), where, for the same reference signal, the maximum $\mu$ is doubled by reducing the size of $B$ by half.

### 4.2. Computational complexity

Table 2 compares the computational complexity in terms of multiplications, additions, and FFTs per iteration of the FPBFxLMS algorithm implemented for a centralized and a distributed ANC system. For the centralized implementation, we consider a multichannel ANC system with one disturbance noise and the same number ($N$) of microphones and loudspeakers (1:$N$:$N$ configuration). For the distributed implementation, we consider a network of $N$ single-channel nodes. However, we only compute the operations of one single-channel node because each node could perform all the operations independently, except the last addition of the global adaptive filters calculated by the previous node. Since

<table>
<thead>
<tr>
<th></th>
<th>N=1</th>
<th>N=4</th>
<th>N=8</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1)</td>
<td>MUX</td>
<td>$4LN + 4LN^2$</td>
<td>$8L$</td>
</tr>
<tr>
<td>ADD</td>
<td>$LN + 3LN^2$</td>
<td>$4L$</td>
<td>$52L$</td>
</tr>
<tr>
<td>MUX</td>
<td>$2 + 6N$</td>
<td>8</td>
<td>26</td>
</tr>
<tr>
<td>ADD</td>
<td>$L + 3LN$</td>
<td>4L</td>
<td>13L</td>
</tr>
<tr>
<td>MUX</td>
<td>$3 + 5N$</td>
<td>8</td>
<td>23</td>
</tr>
</tbody>
</table>
we use a value of $M = L$, and $B = L/2$ (two partitions) the computational complexity only depends on $L$ and $N$.

First, the third column of table 2 shows the computational complexity of both algorithms related to $L$ and $N$. Then, the computational complexity is particularized for $N = 1$, $N = 4$ and $N = 8$. As expected, when $N = 1$, both implementations make the same operations. This is because both the distributed and the centralized ANC system become a single-channel system. When $N = 4$, we compare the operations of the centralized ANC system with a 1:4:4 configuration (16 channels) with the operations of a single-channel node of a network of 4 nodes. The same is done for $N = 8$. Results show that in a centralized ANC system, the computational complexity increases significantly with the number of channels. This fact constitutes a bottleneck in massive multichannel ANC systems. Otherwise, the increase of computational complexity at each node in a distributed ANC system is not so significant. However, in a distributed ANC system we also have to consider the delay in transmitting the global network filters.

In a real implementation of the centralized ANC system with a 16-channel configuration, the computational complexity only depends on $L$. In a real implementation, the value of $B$ has to be chosen to satisfy both the real-time and the causality conditions. On the one hand, if $B$ increases, the system has more time for processing, allowing a better exchange of information between nodes or the possibility to add more nodes to extend the quite zone. On the other hand, if $B$ decreases, it has been proven that the algorithm converges faster.

Moreover, the computational complexity of the distributed algorithm has been studied and compared with the centralized version. Since in the distributed algorithm, each node can perform almost all the operations independently, the computational complexity is significantly reduced at each node. However, in a real implementation, the time used to transfer the network information between nodes would have to be considered. Therefore, in practical implementations, a trade-off between some aspects of the implementation like the size of $B$, the number of nodes ($N$), and the network data transfer rate have to be considered.

5. CONCLUSIONS

An scalable and versatile distributed implementation of the FPBFxLMS algorithm for an ANC system using an incremental strategy in the network has been presented. It has been demonstrated that the proposed algorithm has the same performance than the centralized version when there are no communication constraints in the network. Moreover, some implementation aspects have been studied regarding the block size of the algorithm. In a real implementation, the value of $B$ has to be chosen to satisfy both the real-time and the causality conditions. On the one hand, if $B$ increases, the system has more time for processing, allowing a better exchange of information between nodes or the possibility to add more nodes to extend the quite zone. On the other hand, if $B$ decreases, it has been proven that the algorithm converges faster.

Moreover, the computational complexity of the distributed algorithm has been studied and compared with the centralized version. Since in the distributed algorithm, each node can perform almost all the operations independently, the computational complexity is significantly reduced at each node. However, in a real implementation, the time used to transfer the network information between nodes would have to be considered. Therefore, in practical implementations, a trade-off between some aspects of the implementation like the size of $B$, the number of nodes ($N$), and the network data transfer rate have to be considered.

REFERENCES


