GPU-based Dynamic Wave Field Synthesis using Fractional Delay Filters and Room Compensation

Jose A. Belloch*, *Member, IEEE,* Alberto Gonzalez, *Senior Member, IEEE,* Enrique S. Quintana-Ortí, Miguel Ferrer, *Member, IEEE,* and Vesa Välimäki, *Fellow, IEEE*

Abstract

Wave Field Synthesis (WFS) is a multichannel audio reproduction method, of a considerable computational cost that renders an accurate spatial sound field using a large number of loudspeakers to emulate virtual sound sources. The moving of sound source locations can be improved by using fractional delay filters, and room reflections can be compensated by using an inverse filter bank that corrects the room effects at selected points within the listening area. However, both the fractional delay filters and the room compensation filters further increase the computational requirements of the WFS system. This paper analyzes the performance of a WFS system composed of 96 loudspeakers which integrates both strategies. In order to deal with the large computational complexity, we explore the use of a graphics processing unit (GPU) as a massive signal co-processor to increase the capabilities of the WFS system. The performance of the method as well as the benefits of the GPU acceleration are demonstrated by considering different sizes of room compensation filters and fractional delay filters of order 9. The results show that a 96-speaker WFS system that is efficiently implemented on a state-of-art GPU can

Jose A. Belloch and Enrique S. Quintana-Ortí are with the Depto. de Ingeniería y Ciencia de Computadores, Universitat Jaume I, 12071, Castellón, Spain.

* Corresponding author. E-mail: jbelloch@uji.es.

A. Gonzalez and M. Ferrer are with the Institute of Telecomunications and Multimedia Applications, Universitat Politècnica de València, 46022, Valencia, Spain.

V. Välimäki is with the Dept. of Signal Processing and Acoustics, Aalto University, Espoo, Finland.

Dr. Jose A. Belloch is supported by the postdoctoral fellowship from Generalitat Valenciana APOSTD/2016/069. This work has been partially supported by EU together with Spanish Government through TEC2015-67387-C4-1-R and TIN2014-53495-R (MINECO/FEDER), and Generalitat Valenciana through PROMETEOII/2014/003.

synthesize the movements of 94 sound sources in real time and, at the same time, can manage 9,216 room compensation filters having more than 4,000 coefficients each.

Index Terms

Audio systems, interpolation, parallel architectures, parallel processing, signal synthesis

I. Introduction

In the last few decades, there has been increasing interest in enhancing and improving listening experience, especially spatial audio rendering [1]. One of the spatial audio technologies available today is Wave Field Synthesis (WFS) in which a sound field is synthesized in a wide area by means of loudspeaker arrays, which are referred to as secondary sources [1]. WFS is usually tackled via digital signal processing techniques to reproduce complex auditory scenes consisting of multiple acoustic objects, which are generally denoted as primary or virtual sources. The WFS concept was introduced at the Delft University of Technology in the 1980s. Berkhout developed the first investigations in this field [2], [3], which were followed by several dissertation works [4], [5], [6], [7], [8].

One of the practical problems of implementing WFS is the interaction of the speaker array with the listening room. The room introduces echoes that are not part of the signal to be reproduced, thus altering the synthesized sound field and disturbing the spatial effect. One solution that can be added to the WFS system to minimize the undesirable interaction of the array with the listening room is a Room Compensation (RC) block. A common RC block is based on a multichannel inverse filter bank and corrects the room effects at selected points within the listening area [9], [10]. This formula was validated in [11], where significant improvements were presented in the acoustic field when an RC block is applied to a WFS system. In practice, the application of this spatial audio system (WFS + RC) in real environments (theaters, cinemas, etc.) requires a real-time solution with high computational requirements.

Another special situation in WFS occurs when it is necessary to render a moving sound source through a specific trajectory. In a WFS system that is implemented using discrete-time processing, accurate modeling of propagation times requires a signal to be delayed by a number of samples that is not an integer multiple of the sampling intervals. To this end, we propose the use of fractional delay filters [12], [13] to render the audio signals in this scenario. These filters are used to interpolate a signal value between the sampling points. Other applications of fractional delay filtering in audio signal processing include digital audio effects [14], [15] and physical sound synthesis [16], [17]. Franck et al. have studied techniques to interpolate time delays in WFS and have analyzed artifacts that arise when fractional delays

are not considered [18], [19], [20]. The use of fractional delays in a WFS system has also been analyzed in [21], [22], [23].

A large-scale WFS system with massive additional filtering requires costly computational operations in real time. One solution to this problem is to perform all audio processing tasks in a Graphics Processing Unit (GPU). Accelerators of this type have already been applied to different problems in acoustics and audio processing. Some applications include room acoustics modeling [24], [25], [26], [27], additive synthesis [28], [29], full 3-D model of drums in a large virtual room [30], sliding phase vocoder [31], beamforming [32], audio rendering [33], [34], [35], multichannel IIR filtering of audio signals [36], dynamic range reduction using multiple allpass filters [37], and adaptive filtering [38], [39], [40].

In recent years, there have been several studies aimed at implementing a WFS system. An approach that benefited from time-invariant preprocessing in order to reduce the computational load is presented in [41]. In [42], the authors propose a minimal processor architecture that is adapted to WFS-based audio applications. They estimated that their system could render (in real time) up to 32 acoustic sources when driving 64 loudspeakers. In [43], the same researchers presented a WFS implementation on different multi-core platforms, including a GPU-based implementation that controlled more than 64 sources when driving 96 loudspeakers. They concluded that GPUs are suitable for building immersive-audio, real-time systems. In [44], the authors also introduced a GPU-based hybrid time-frequency implementation of WFS. Real-time issues in WFS were tackled in [45] using the NU-Tech framework [46].

In comparison with previous efforts, the WFS implementation proposed in this work improves the quality of the virtual sound by adding room compensation and time-varying fractional delays filters. Moreover, we design our application to achieve high performance on GPUs by exploiting a Kepler GK110 [47] co-processor. This architecture can be found on the Tegra K1 (TK1) systems-on-chip (SoC), which is embedded in the Jetson development kit (DevKit) [48]. It is also integrated into current mobile devices such as the Google Nexus 9 tablet [49]. Therefore, this implementation can be ported to perform efficiently on GPUs which are currently embedded in mobile devices.

This paper extends our previous work [50] in various ways. First, we use fractional delay filters in order to reproduce a sound source position with high accuracy and to be able to move a sound source smoothly using time-delay steps of less than one sample along the way. Second, we now implement the WFS system with different sample buffer sizes, which establish the minimum and the maximum time that a sound source can stay in one position and thus the different possibilities regarding the speed of the sound source. Finally, for shorter buffer sizes, we now implement a uniformly-partitioned, FFT-based convolution, which can efficiently convolve FIR filters that have a large number of coefficients with audio

buffers with smaller size. The algorithm that we implement was introduced in [51] and uniformly splits the room compensation filters into blocks of the same size as the audio buffer. This improvement is crucial since it reduces the latency of the system independently of the size of the room compensation filters.

This paper presents the performance of a GPU-based dynamic WFS implementation that (i) allows accurate synthesis of virtual sound source positions and thus accurate movement trajectories using Fractional Delay Filters; (ii) leverages an inverse filter bank to improve the spatial effect of the WFS; and (iii) is capable of synthesizing a large number of virtual sound sources in real time. We also analyze the potential of using time-varying fractional delay filters in WFS and the influence on the computational time for different sizes of audio buffers and room compensation filters.

This paper is structured as follows. Section II briefly describes the key architecture aspects of the target GPUs from NVIDIA and highlights the features to be accounted for when implementing a real-time WFS on this type of accelerator. Section III offers a brief overview of the WFS theory. Section IV enumerates different kinds of fractional delay filters, assesses which fractional delay filter is the most appropriate for a WFS system, and tests subjectively the different techniques for carrying out sound source movements inside this system. Section V presents a detailed description of a GPU-based implementation of the WFS system. In Section VI, we evaluate the computational performance of the WFS system, and we summarize our results in Section VII.

II. Use of a GPU in a Real-Time WFS System

Dealing with real-time audio applications on GPUs requires a basic understanding of how architectures of this type are programmed. This section provides a brief description of the GPU data flow and outlines some relevant issues to take into account when developing a real-time WFS application on a graphics accelerator from NVIDIA.

A. GPU and CUDA

Following Flynn's taxonomy [52], from a conceptual point of view, a GPU can be viewed as a Single Instruction Multiple Data machine (SIMD), i.e., a computer in which a single flow of instructions is executed on different data sets. Implementations of this model usually work synchronously with a common clock signal. An instruction unit sends the same instruction to the processing elements, which execute this instruction on their own data simultaneously. A GPU is composed by multiple Stream Multiprocessors (SM) with 192 pipelined cores per SM, for NVIDIA's 3.5 capability (Kepler architecture [47]).

A GPU device has a large amount of off-chip device memory (*global-memory*) and a fast, but smaller, on-chip memory (*shared-memory*, *registers*). The shared-memory is normally used by threads that must share data. There are also read-only cached memories, which are called *constant-memory* and *texture-memory*. *Constant-memory* is optimized for broadcast (e.g., all threads have to read the same memory location), while the *texture-memory* is oriented to graphics operations. Fig. 1 shows the organization of a GPU. Advanced GPU devices (beyond 2.x capability) come with an L1/L2 cache hierarchy that is used to cache global-memory. The L1 cache uses the same on-chip memory as shared-memory.

An important aspect when reading from or writing to global-memory is to perform these accesses in a *coalesced* manner, as this can significantly reduce the memory-access time. Coalescing means that the threads have to read from/write to a small range of memory addresses that match a certain pattern. Let `idx` be the identification of a thread and `array` a pointer to *global-memory*. We attain perfect coalescence when the `idx` thread accesses the `array[idx]` and `idx+1` accesses the `array[idx+1]`.

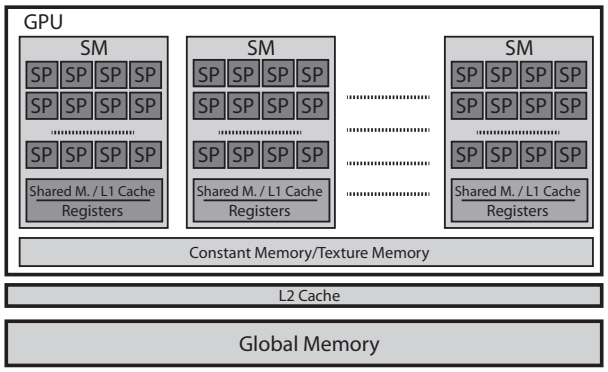

Fig. 1. The GPU is configured by 16 Stream Multiprocessors (SMs), each of which has 192 pipelined cores (SP).

CUDA is an extension of the C language to ease the development of GPU-oriented efficient solvers for complex problems with high computational cost [53]. This interface can be used to leverage the vast number of execution threads that are available in a state-of-the-art GPU. In CUDA, the programmer defines the kernel function that contains code (operations) to be executed on the GPU. This kernel routine is invoked from the main program, which also has to define a grid configuration stating the number of execution threads and how they are distributed and grouped.

B. Real-Time Processing of a WFS System on a GPU

The target WFS system is located at the *Universitat Politècnica de València* (UPV) and is operated by the Audio and Communications Signal Processing Group (GTAC) [54]. This system is composed of

N=96 loudspeakers that are positioned in an octogonal geometry, with a separation of 18 cm between neighboring loudspeakers (see Fig. 2 for a graphical representation of this configuration).

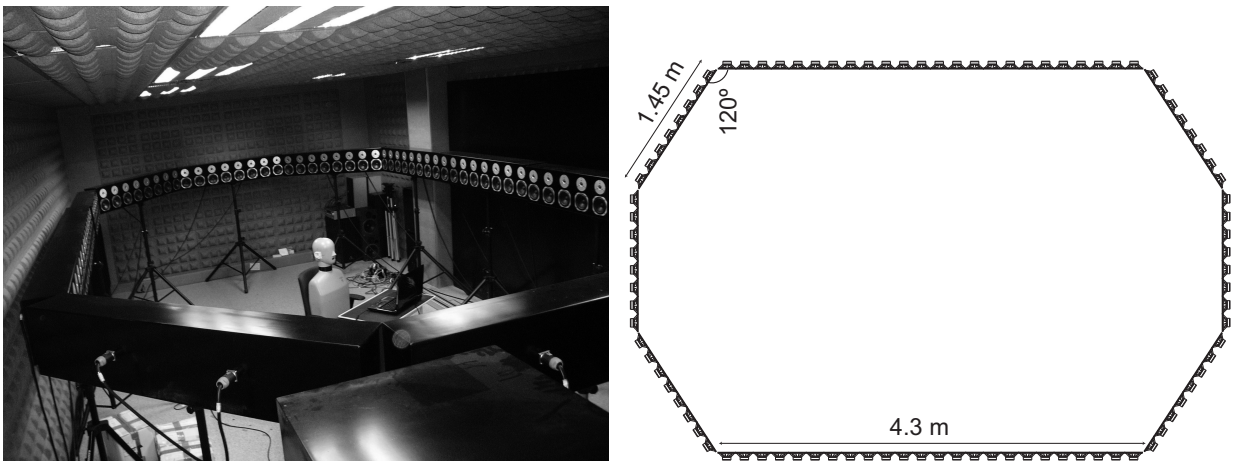

Fig. 2. Configuration of the WFS loudspeaker array available in the laboratory of the GTAC at UPV.

The loudspeakers are connected to four MOTU 24I/O audio cards that use the ASIO (Audio Stream Input/Output) driver to communicate with the CPU. This driver works with blocks of L samples obtained with a sampling rate f_s. Thus, every Lf_s seconds, the audio card requires the loudspeakers to reproduce N output-data buffers of size L. This time is denoted as t_{buff} [35] and is independent of the number of loudspeakers and the number of virtual sound sources (M) in the WFS system. In contrast, the processing time t_{proc} depends both on M and N. Here, t_{proc} includes the time spent on data transfers between the GPU and the CPU and the time used for the computation on the GPU (CUDA kernels). These data transfers are carried out via the PCI-e X 16 bus, which has an approximate bandwidth rate of 8 GB/s. Therefore, the WFS system works in real time provided $t_{\mathrm{proc}} < t_{\mathrm{buff}}$. When this condition no longer holds, the application can still work off-line (i.e., processing the audio samples in order to reproduce them later).

III. Fundamentals of WFS

Wave Field Synthesis is a sound rendering method that is based on fundamental acoustic principles [2]. It enables the generation of sound fields with natural temporal and spatial properties within a volume or area bounded by secondary sources (arrays of loudspeakers, see Fig. 2). This method offers a large listening area with uniform and high reproduction quality.

The theoretical basis for WFS is given by Huygens' principle [3]. According to this principle, the propagation of a wave front can be described by adding the contribution of a number of secondary-point sources distributed along the wave front, where a synthesis operator is derived for each secondary source.

This principle can be used to synthesize acoustic wave fronts of any arbitrary shape. For simplicity, the general 3-D solution can be transformed into a 2-D solution, which is sufficient to be able to reconstruct the original sound field in the listening plane [4], [5], [55]. For that purpose, a linear array of loudspeakers is used to generate the sound field of virtual sources.

Following a model-based rendering in which point sources and plane waves are used [56], the field rendered by a sound source m at a point R, within the area surrounded by N loudspeakers, can be expressed as

$$P(\mathbf{x}_R, \omega) = \sum_{n=0}^{N-1} Q_n(\mathbf{x}_m, \omega) \frac{e^{\frac{-j\omega r_{nR}}{c}}}{r_{nR}}, \tag{1}$$

where c is the speed of the sound, $\mathbf{x}_m$ is the position of the virtual sound source m, $\mathbf{x}_R$ is the position of the point R, and $r_{nR} = \mid \mathbf{x}_n - \mathbf{x}_R \mid$ is the distance between the n-th loudspeaker and the point R.

The driving signal of the n-th loudspeaker is represented by $Q_n(\mathbf{x}_m, \omega)$, which is given by

$$Q_n(\mathbf{x}_m, \omega) = S(\omega) \sqrt{\frac{j\omega}{2\pi c}} K \frac{1}{\sqrt{r_{mn}}} \cos(\theta_{mn}) e^{\frac{-j\omega r_{mn}}{c}}, \tag{2}$$

where K is a geometry-dependent constant, $r_{mn} = \mid \mathbf{x}_m - \mathbf{x}_n \mid$ and $\mathbf{x}_n$ is the position of the loudspeaker n. Fig. 3 shows the geometry of the system, where θ_{mn} is the angle between the line that connects $\mathbf{x}_m$ and $\mathbf{x}_n$ and the normal vector $\mathbf{n}$ of the loudspeaker n. The piano represents the sound source m. The driving signal (2) consists of several elements that have different functionalities. The term

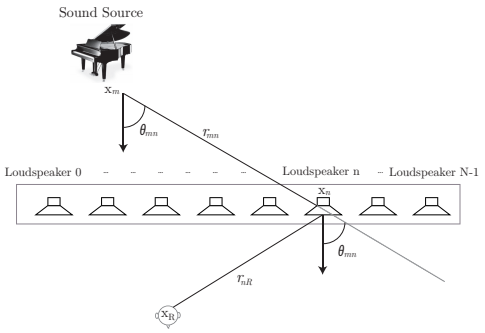

Fig. 3. Geometry of a WFS system with the sound source m (piano), N loudspeakers, and the distances among the sound source, the loudspeakers, and the listener (R).

$S(\omega)$ is the frequency-domain characteristic of the source signal, while the term

$$H(\omega) = \sqrt{\frac{j\omega}{2\pi c}} \tag{3}$$

represents a filtering operation that is independent of the position of the virtual source. In [57], (3) is referred to as a WFS pre-equalization filter that represents a lowpass filter with a constant slope of 3 dB/octave when the loudspeaker is considered a monopole secondary source. If the loudspeaker is considered a dipole secondary source, the WFS pre-equalization filter corresponds to a highpass filter with a magnitude increase of 3 dB/octave. An important contribution in (2) is

$$a_{mn} = \frac{K}{\sqrt{r_{mn}}} \cos(\theta_{mn}), \tag{4}$$

which denotes an amplitude factor that depends on the positions of the sound source m and the loudspeaker n. Finally, the term $e^{\frac{-j\omega r_{mn}}{c}}$ represents the phase shift corresponding to a time delay that depends on the distance between the virtual sound source m and the loudspeaker n.

The driving signal shown in (2) can also be expressed in the time domain as

$$q_n^m(t) = a_{mn} s_m(t) * h(t) * \delta(t - \frac{\mid \mathbf{x}_m - \mathbf{x}_n \mid}{c}), \tag{5}$$

where $*$ denotes the convolution operator, $s_m(t)$ is the signal of sound source m, and $h(t)$ is the inverse Fourier transform of $H(\omega)$ in (3).

In a multi-source system composed of M virtual sound sources, the loudspeaker driving signal of the nth loudspeaker is

$$q_n(t) = \sum_{m=0}^{M-1} q_n^m(t). \tag{6}$$

In a discrete-time signal processing system with sampling frequency f_s, (5) and (6) boil down to

$$q_n[k] = \sum_{m=0}^{M-1} a_{mn} s_m[k] * h[k] * \delta[k - \tau_{mn}], \tag{7}$$

where k is the sample index, and

$$\tau_{mn} = \frac{\mid \mathbf{x}_m - \mathbf{x}_n \mid}{c} f_s \tag{8}$$

is the delay in number of samples.

A. Room Compensation in a WFS System

The interaction of the driving signals with the listening room can deteriorate the rendering properties of the WFS system. The synthesized sound field can be altered by new echoes that are introduced by the listening room, reducing the spatial effect. In [11], [58], the authors designed and validated a multichannel

inverse filter bank that corrects these room effects at selected points within the listening area. However, in a WFS system composed of N loudspeakers, this implies inserting N^2 FIR filters to the system, considerably increasing its computational demands. The operations that are carried out in a multichannel inverse filter bank with every driving signal are given by

$$y_n(t) = \sum_{j=0}^{N-1} q_j(t) * f_{jn}(t). \tag{9}$$

Thus, the final signal to be reproduced by the n-th loudspeaker $y_n(t)$ is a combination of all of the filtered signals (as illustrated in Fig. 4), where the filter $f_{0n}(t)$ transmits the driving signal $q_0(t)$ to the loudspeaker n.

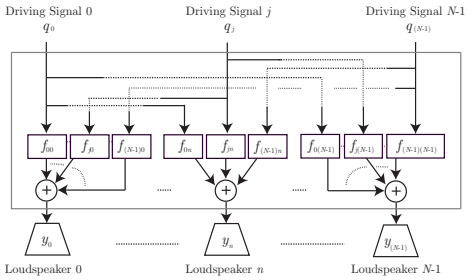

Fig. 4. Multichannel inverse filter bank, where each driving signal is convolved by N filters. The signal that is reproduced by a loudspeaker is a combination of all of the filtered signals.

The calculation of the inverse filters can be carried out in a setup stage since the main room reflections at low frequencies, can be considered to be invariant for this specific room. Different methods have been proposed to obtain the bank of correction filters. There are methods that compute an approximate solution in the frequency domain using the FFT [59]. In this work, we instead leverage a method to compute the correction filters f_{jn}, which guarantee a minimal square error solution in the time domain [60]. A detailed description of the operations that were carried out for the computation of the filters is reported in [11].

B. Accurate Location in Dynamic Sound Sources

The virtualization of the sound source movements is carried out by smoothly varying the virtual positions of sound sources over time. In practice, this implies switching the synthesis of the sound source from the actual position to a new position. Considering a linear loudspeaker array composed of 24 loudspeakers, such as that in Fig. 5, we can define a uniform grid of points where a virtual sound source and a possible trajectory of the sound source (dark thin arrow from left to right) can be set.

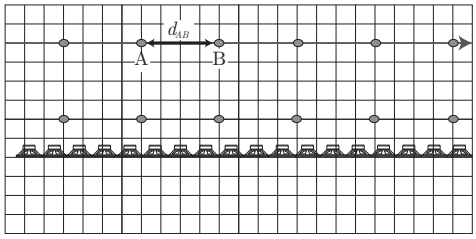

Fig. 5. Trajectory of the sound source (dark thin arrow from left to right), where d_{AB} is the trajectory resolution of the WFS system (dark double arrow between point A and point B).

TABLE I

SPEED OF A VIRTUAL SOUND SOURCE IN A WFS SYSTEM FOR DIFFERENT COMBINATIONS OF BUFFER SIZE L AND RESOLUTION d_{AB} AT $f_s = 44.1$ KHZ.

d_{AB} (in m)	Sizes of buffers L (samples)		
	64	256	1024
0.0001	0.068 (m/s)	0.017 (m/s)	0.004 (m/s)
0.0010	0.689 (m/s)	0.172 (m/s)	0.043 (m/s)
0.0025	1.722 (m/s)	0.430 (m/s)	0.107 (m/s)
0.0050	3.445 (m/s)	0.861 (m/s)	0.215 (m/s)
0.0100	6.890 (m/s)	1.722 (m/s)	0.430 (m/s)

Depending on the displacement between the grid points of the sound source, we can also define d_{AB} as the minimum distance between two contiguous points (dark double arrow between point A and point B in Fig. 5). If the sound source movement is limited to a fixed displacement, d_{AB} will denote the trajectory resolution of the WFS system. The trajectory resolution will also constrain the speed that a virtual sound source can achieve, since this is given by $d_{AB}f_s/L$, where f_s is the sampling frequency. Several speed examples obtained from the combination of different sizes of input-data buffers and trajectory resolutions d_{AB} are shown in Table I with the sampling frequency of 44.1 kHz.

Table II shows the maximum variation that is observed in the amplitudes a_{mn} and the delays τ_{mn} of the driving signals for different trajectory resolutions d_{AB} when a virtual sound source m is shifted following the trajectory marked in Fig. 5. This means that, if this virtual sound source is initially synthesized at point A, and later at point B, its parameters a_{mn}^A and τ_{mn}^A radically change to a_{mn}^B and τ_{mn}^B. The second and third columns of Table II illustrate the differences $\max \mid a_{mn}^A - a_{mn}^B \mid$ and $\max \mid \tau_{mn}^A - \tau_{mn}^B \mid$, taking

TABLE II

MAXIMUM CHANGES IN THE AMPLITUDES AND DELAYS IN THE DRIVING SIGNALS WHEN A VIRTUAL SOUND SOURCE IS SHIFTED IN STEPS OF d_{AB} IN METERS.

d_{AB} (in m)	$\max \mid a_{mn}^{A} - a_{mn}^{B} \mid$	$\max \mid \tau_{mn}^{A} - \tau_{mn}^{B} \mid$
0.0001	0.0001	0.0126
0.0010	0.0014	0.1260
0.0025	0.0035	0.3149
0.0035	0.0048	0.4408
0.0050	0.0070	0.6298
0.0070	0.0090	0.8815
0.0075	0.0105	0.9447
0.0080	0.0112	1.0077
0.0090	0.0124	1.1334
0.0100	0.0140	1.2596
0.0500	0.0721	6.2959
0.1000	0.1387	12.5864
0.3000	0.3969	37.6863

into account that all 24 loudspeakers are reproducing this particular sound source displacement.

Table II indicates that the growth of d_{AB} yields larger differences in amplitudes and delays during the displacement. This can lead to discontinuities in the synthesized signals (i.e., to the appearance of non-linear artifacts). At this point, we face two possible scenarios. First, we can use a crossfade technique to reduce artifacts as proposed in [61]. This technique synthesizes a sound source in both positions (point A and point B) and then combines them by means of a gradual increase in the sound rendered by the new position (fade-in) while the sound rendered by the old position decreases (fade-out) in the same proportion. Thus, it doubles the number of operations to compute. The second option consists in using small d_{AB} values, since the variations in amplitudes are insignificant as the virtual sound source shifts. However, the delays vary substantially in comparison with the amplitudes. In fact, the use of a trajectory resolution of less than 8 mm requires the introduction of an interpolation technique that allows delay values that are smaller than one sample interval to be produced. Hence, in order to achieve suitable trajectory resolutions, and thus to delay a signal by a number of samples that is not an integer value, we propose an alternative approach based on the use of fractional delay filters.

IV. TIME-VARYING FRACTIONAL DELAY FILTERS

Computing the delays τ_{mn} implies delaying a signal by a number of samples that is not always an integer value. A common solution to this problem consists in rounding τ_{mn} to the nearest integer. However, this can lead to acoustic artifacts [20]. Fractional delay filters allow a digital signal to be delayed by a fractional number of samples [12]. Different fractional delay techniques, such as linear interpolation, cubic interpolation, and Lagrange interpolation have been presented in [16]. Linear interpolation is achieved by filtering the signal through a first-order FIR filter

$$y[k-\alpha] = (1-\alpha)y[k] + \alpha y[k-1], \tag{10}$$

where α is a decimal number so that $0 \leq \alpha <1$.

Cubic interpolation is achieved by filtering the signal through a third-order FIR filter

$$y[k-\alpha] = \sum_{j=0}^{3} h_{fd}[j]y[k-j], \tag{11}$$

where

$$\begin{aligned} h_{fd}[0] &= -(1/6)(D(\alpha)-1)(D(\alpha)-2)(D(\alpha)-3), \\ h_{fd}[1] &= (1/2)D(\alpha)(D(\alpha)-2)(D(\alpha)-3), \\ h_{fd}[2] &= -(1/2)D(\alpha)(D(\alpha)-1)(D(\alpha)-3), \\ h_{fd}[3] &= (1/6)D(\alpha)(D(\alpha)-1)(D(\alpha)-2), \end{aligned} \tag{12}$$

and $1 < D(\alpha) < 2$.

A more accurate technique for fractional delay FIR filter design was introduced in [62]. It is based on truncating a Lagrange fractional delay filter. This approach deletes a number of coefficients at the beginning and at the end of the coefficient vector of a prototype Lagrange fractional delay filter. This technique can be interpreted as a hybrid method that combines properties of the Lagrange and the truncated sinc fractional delay filters [16]. The design of the coefficients is computationally efficient and is based on a polynomial formula. In practice, the P-th order truncated Lagrange fractional delay filter $h_{fd}[k]$ is obtained by discarding the K_1 coefficients from each end of the T-th order prototype Lagrange Fractional Delay filter $h_L[k]$ as

$$h_{fd}[k] = \begin{cases} 0 & 0 \leq k \leq K_1 - 1 \\ h_L[k] & K_1 \leq k \leq P + K_1 \\ 0 & P + K_1 + 1 \leq k \leq T \end{cases}$$

$$h_L[k] = \prod_{p=0, p\neq k}^{T} \frac{D-p}{k-p}, \tag{13}$$

where $T > P$, K_1 is a positive integer ($K_1 > T/2$) and D is a real number that depends on the fractional delay [12], [63].

In order to assess the effect of the fractional delays in a WFS system, we provide an objective comparison among three different WFS driving signals. In all cases, our test sounds are composed of one specific tone f and have a duration of 3 seconds (signals composed of $3f_s$ samples). The worst case scenario corresponds to a frequency $f = 15$ kHz at $f_s = 44.1$ kHz since this frequency stays within the standard audio bandwidth of 20 kHz and causes large variations in the sound wave. Although a WFS system presents an aliasing frequency that reduces the spatial effect, the human auditory system is not very sensitive to these aliasing artifacts [56] and all the signal components are rendered without limitation. Thus, it is reasonable to evaluate the behavior of the WFS system at high frequencies in order to obtain an error bound. There are three different WFS driving signals generated from the virtual sound source:

$Q_{I,n}$ is the reference ideal signal. From a_{mn} and τ_{mn} (m is equal to 1), it is computed as

$$Q_{I,n} = a_{mn} \sin(2\pi \frac{f}{f_s}(k - \tau_{mn})); \tag{14}$$

where k is the sample index $k \in \{0, 1, 2, \ldots, 3f_s - 1\}$, $n \in \{0, 1, 2, \ldots, 23\}$. The number of loudspeakers that are involved in the WFS system is 24, following the setup shown in Fig. 5.

$Q_{R,n}$ is obtained by delaying a computed WFS signal by $\widehat{\tau}_{mn}$ samples, where $\widehat{\tau}_{mn}$ is obtained by rounding τ_{mn} to the nearest integer,

$$\begin{aligned} q_n &= a_{mn} \sin(2\pi \frac{f}{f_s} k), \text{and} \\ Q_{R,n} &= q_n * \delta[k - \widehat{\tau}_{mn}]. \end{aligned} \tag{15}$$

$Q_{fd,n}$ is obtained from the fractional delay filter. To this end, a WFS signal must first be obtained by delaying a computed WFS signal by $\widetilde{\tau}_{mn} = \lfloor \tau_{mn} \rfloor$. Then, the obtained signal is filtered through the fractional delay filter h_{fd}, whose coefficients depend on the difference between $\widetilde{\tau}_{mn}$ and τ_{mn}.

$$\begin{aligned} q_n &= a_{mn}\sin(2\pi\frac{f}{f_s}k), \\ Q_{R,n} &= q_n * \delta[k-\widetilde{\tau}_{mn}], \text{and} \\ Q_{fd,n} &= Q_{R,n} * h_{fd}. \end{aligned} \tag{16}$$

The signal $Q_{fd,n}$ takes different forms depending on how the coefficients of the fractional delay filter were obtained: by means of linear interpolation, $Q_{fd_L,n}$, cubic interpolation, $Q_{fd_C,n}$, or truncated Lagrange interpolation, $Q_{fd_T,n}$. In the last case, the fractional delay filters are configured with a 9th-order truncated filter from the 29th-order prototype since this presents a wider range of fairly flat frequency response than the standard Lagrange Interpolation filter (see [62]). Note that filter h from (3) is not taken into account in the previous equations since it has the same influence on all signals.

A. Evaluation of Fractional Delay Filters

We perform a sound source movement following the trajectory marked in Fig. 5 (grey thin arrow from left to right) so that all 24 loudspeakers are active (reproducing) during the entire movement. This array is implemented with the 24 loudspeakers that are located in one of the horizontal sides of Fig. 2. The sound source movement is generated for five different trajectory resolutions $d_{AB} \in \{0.0001\,\text{m}, 0.0010\,\text{m}, 0.0025\,\text{m}, 0.0050\,\text{m}, 0.0100\,\text{m}\}$ and the four proposed synthesized signals: $Q_{R,n}$, $Q_{fd_L,n}$, $Q_{fd_C,n}$, and $Q_{fd_T,n}$.

We compare the four synthesized signals with the reference $Q_{I,n}$ to measure which group of the synthesized signals best matches the theoretical WFS signals. Therefore, we compute the Mean Relative Error (MRE):

$$\text{MRE}_j = \frac{\sum_{n=1}^{24}\sum_{k=1}^{3f_s}(Q_{I,n}[k]-Q_{j,n}[k])^2}{\sum_{n=1}^{24}\sum_{k=1}^{3f_s}(Q_{I,n}[k])^2}, \tag{17}$$

where the subscript $j \in \{R,\ fd_L,\ fd_C,\ fd_T\}$ represents each one of the four proposed synthesized signals: $Q_{R,n}$, $Q_{fd_L,n}$, $Q_{fd_C,n}$, and $Q_{fd_T,n}$. Table III shows that, as the trajectory resolution increases, the MRE grows because it is more difficult to match the ideal signal. The MRE is reduced when the synthesis is carried out using fractional delays instead of rounding the delays to the nearest integer. In fact, the truncated Lagrange interpolation delivers the closest approximation to the ideal case.

B. Subjective Evaluation

We have carried out an informal listening test where we have compared three different techniques for carrying out a sound source movement inside a Wave Field Synthesis system. All of the sound source

TABLE III

MRE FOR DIFFERENT TRAJECTORY RESOLUTIONS d_{AB} (IN METERS) FOR THE PROPOSED SYNTHESIZED SIGNALS IN DECIBELS AT $f_s = 44.1$ KHZ.

d_{AB} (in m)	Mean Relative Error (in dB)			
	$Q_{R,n}$	$Q_{fd_L,n}$	$Q_{fd_C,n}$	$Q_{fd_T,n}$
0.0001	-8.382	-16.496	-24.369	-74.629
0.0010	-8.379	-16.488	-24.290	-54.602
0.0025	-8.365	-16.476	-23.561	-46.769
0.0050	-8.307	-16.457	-23.184	-41.303
0.0100	-8.245	-16.243	-23.991	-37.522

movements had the same duration and consisted in a piano sound that followed the trajectory illustrated in Fig. 6.

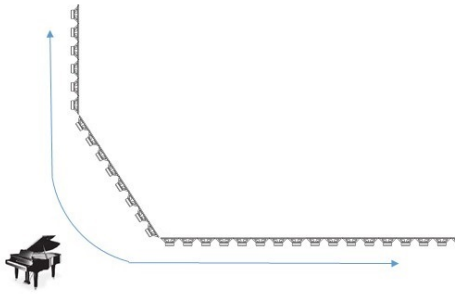

Fig. 6. Trajectory that followed the sound source for carrying out the subjective tests. When the piano arrives to one of the ends, it turns back and continues successively till the end of the sound.

The three techniques were generated and labeled as:

NO No interpolation. The sound source is moving by drastically changing spatial positions by small jumps of d_{AB} = 0.1 m (the virtual sound source is initially synthesized at point A, and later at point B. Its parameters a_{mn}^A and τ_{mn}^A drastically change to a_{mn}^B and τ_{mn}^B).

CR Crossfading. This technique synthesizes a sound source in both positions (point A and point B), and then combines them by means of a gradual gain increase in the sound rendered by the new position (fade-in) while the sound rendered by the old position decreases (fade-out) in the same proportion. This technique requires high computational resources, as it doubles the number of all operations.

FD Fractional delay filtering (proposed method). This option, which is computationally simpler than CR above, consists of using a small trajectory resolution (d_{AB} = 0.001 m was used) and changes the spatial position in smaller steps and more quickly to obtain a duration equal to the previous signals. A trajectory resolution of less than 8 mm requires an interpolation technique producing fractional delay values. To achieve a suitable trajectory resolution, we use of the truncated Lagrange interpolation, which delivers a close approximation to the ideal case (see Sec. V.A).

A subjective test was carried out in order to reveal which technique produces the most realistic movement, taking into account the human perception. We carried out a test in which the three techniques were compared using a hidden reference paradigm [64].

The three techniques were compared by pairs in a test of six questions. A total of 21 people participated in the listening experiment; their ages were between 23 and 35. The hearing of all test subjects was tested using standard audiometry. None of them had a reportable hearing loss, which could affect the results. Fig. 7 shows the preference of the subjects when the three techniques were pair-compared.

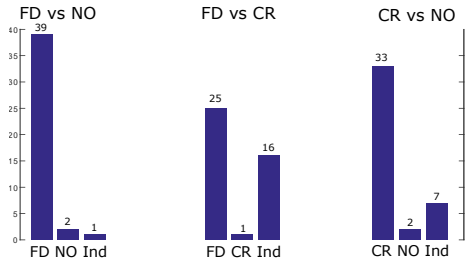

Fig. 7. Preference of the test subjects when the three techniques were pair compared. **Ind** refers to indifference (i.e., no preference).

The results in Fig. 7 show that the techniques CR and FD are preferred to the NO technique. This symbolizes that additional processing must be carried out in order to synthesize a realistic movement. Between CR and FD, the test subjects preferred FD. This implies that they identify the use of small trajectory resolution and a fractional delay interpolation as a more realistic movement than the one which is carried out by using a crossfading technique. This is a highly useful result, since the implementation based on fractional delay filters, which sounds better, requires also significantly less computing than the crossfading technique.

V. Implementation of the WFS Processing on the GPU

As introduced in Section II-B, the WFS system consists of several multichannel audio cards that provide audio buffers every Lf_s seconds, where L is the buffer size in samples. We denote the input-data buffer of L samples of the sound source m by $\mathbf{s}_{\mathrm{buff}_m}$, and the output-data buffer of the L samples that feeds the loudspeaker n by $\mathbf{y}_{\mathrm{buff}_n}$. We use the GPU to accelerate all of the processing tasks of a WFS system that integrates fractional delay filters and room compensation filters. For this purpose, the operations are applied simultaneously on all of the buffers and on each sample.

In previous work [50], we implemented a WFS system based on an overlap-save technique in the frequency domain. However, in this work we reduce the number of real-time filtering computations by convolving filter h (which is independent of the position of the virtual sound source) with all the filters that compose the multichannel inverse filter bank following the equation

$$\widetilde{f}_{jn} = f_{jn} * h, \tag{18}$$

where $n, j \in [0, N-1]$. Thus, our WFS implementation only requires delaying and weighting the source signal:

$$q_n[k] = \sum_{m=0}^{M-1} a_{mn} s_m[k] * \delta[k - \tau_{mn}]. \tag{19}$$

As equation (19) is rather simple, we perform the delay τ_{mn} and the weight a_{mn} of the sound signal in the time domain for this WFS implementation.

The WFS system starts from a virtual sound source, which is defined by its position $\mathbf{x}_m$, and the audio samples (which are given by audio buffers $\mathbf{s}_{\mathrm{buff}_m}$). The distance r_{mn} and the angle θ_{mn} are computed from this position $\mathbf{x}_m$ and the location of the loudspeakers $\mathbf{x}_n$ (see Fig. 3). Algorithm 1 describes all the operations that are necessary to execute the WFS system. The input variables for this algorithm are the number of sound sources M, the parameters r_{mn} and θ_{mn}, the audio buffers $\mathbf{s}_{\mathrm{buff}_m}$, and the filters $\widetilde{f}_{jn}$.

The following subsections present a detailed description of the GPU implementation of the two key processing blocks: Driving Signals of WFS and Room Compensation filtering.

A. Driving Signals of WFS

We use the following CUDA kernels to compute steps 2 to 14 in Algorithm 1.

Kernel 1 launches NM threads and computes steps 5, 6, and 7. The kernel inputs are the coordinates of the virtual sound sources and the positions of the loudspeakers. Each thread computes a simple factor τ_{mn}, $\widetilde{\tau}_{mn}$, and a_{mn}.

Kernel 2 computes the coefficients of the fractional delay filters (step 8). The highest accuracy is obtained with a 9th-order filter based on truncated Lagrange. In this case, each filter is composed of ten coefficients, and, since there is a filter per sound source and loudspeaker, $10NM$ threads are spawned. Each thread computes a coefficient of one of the filters.

Kernel 3 computes steps 9 and 10. A tridimensional matrix composed of the audio buffers $\mathbf{q}^m_{\mathrm{buff}_n}$ is configured: the number of rows in this matrix matches the number of sources M; the number of columns is $2L$ (number of samples per buffer, see section II-B); and the third dimension corresponds to the number of loudspeakers $N = 96$ in this WFS system (see Fig 8). Therefore, $2LNM$ threads are spawned. The

Algorithm 1 WFS system with Room Compensation and Fractional Delay Filters.

Input: M, θ_{mn}, r_{mn}, $\mathbf{s}_{\mathrm{buff}_m}$, $\widetilde{f}_{jn}$

Output: $\mathbf{y}_{\mathrm{buff}_n}$

```
/*----- Driving Signals of WFS ----*/
for n = 0,...,N-1 do                                   ▷Computation of amplitudes and delays
    q_buff_n = 0;                                      ▷Initialization of buffers with zeros
    for m = 0,...,M-1 do                               ▷Combination Loudspeaker-Sound Source
        a_mn = K/sqrt(r_mn) cos(θ_mn).                 ▷r_mn =| x_m − x_n |;
        τ_mn = (r_mn/c) f_s.                           ▷see Fig (3) for θ_mn.
        τ̃_mn = ⌊τ_mn⌋.
        h_fd = Compute_Fractional_Delay_Filters(τ_mn − τ̃_mn).
        q^m_buff_n = a_mn · (s_buff_m * δ[k − τ̃_mn]).
        q_buff_n = q_buff_n + (q^m_buff_n * h_fd).
    end for
end for
/*----- Room Compensation filtering ----*/
for n = 0,...,N-1 do
    y_buff_n = 0;                                      ▷Initialization of output buffers with zeros
    for j = 0,...,M-1 do
        y_buff_n = y_buff_n + (q_buff_j * f̃_jn).       ▷Filter h is included. See Equation (18)
    end for
end for
```

task of the threads is to compute and to group all output audio samples by considering $\widetilde{\tau}_{mn}$ and a_{mn} (combining each sound source with each loudspeaker). The configuration of this matrix is crucial to be able to efficiently perform the next steps of the implementation. The threads access global-memory in a coalesced manner starting from the integer part of τ_{mn}, which is used internally as a pointer to inform the GPU threads which audio samples must be used for the processing. The CUDA device has a compute capability 3.5, and the compiler is forced to use the read-only memory path in order to load audio samples from the global-memory since this memory has a high bandwidth. The last feature to be configured in the CUDA programming is the L1 cache, which is set to 48 kB in order to cache memory loads from global-memory. These three combined actions reduce the possibility of memory conflicts when multiple threads access the global-memory concurrently.

Kernel 4 is devoted to accumulating all of the samples of each loudspeaker (step 10). For this purpose, $2LN$ threads are spawned and each thread performs M additions.

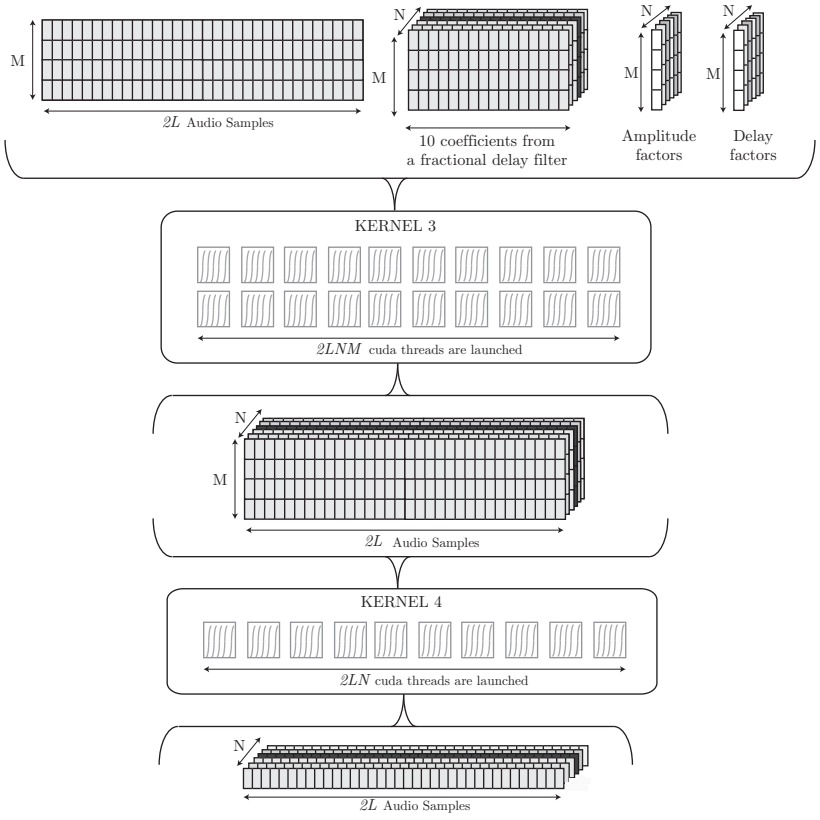

Fig. 8. Kernel 3 and Kernel 4 perform the computation of the driving signals of a WFS system.

If a crossfade technique is used, two tridimensional matrices composed of audio samples are configured since there are two amplitudes a^A_{mn} and a^B_{mn}, and two delays τ^A_{mn} and τ^B_{mn} (A and B represent the two points between which the movement in the sound source is rendered). Therefore, twice the number of threads are spawned by **Kernels 1**, **2**, and **3** in this case (see [50] for implementation details of the crossfade technique).

B. Room Compensation

In previous work [50], we used filters with the same size as the sample buffers. Specifically, the size that was previously considered was $L = 512$. However, the filters that carry out the Room Compensation usually have a large number of coefficients [65]. Therefore, if audio buffers of the same size are used, the latency of the system will substantially increase. This also implies that the movements of the virtual sound sources will become slow since the buffer size L is very large.

TABLE IV

AVERAGE PERCENTAGE OF THE TOTAL PROCESSING TIME THAT EACH KERNEL REQUIRES FOR $M = 300$ AND $B = 6$.

Processing Blocks	Percentage
Kernel 1	3%
Kernel 2	1.5%
Kernel 3	2%
Kernel 4	1.5%
Room Compensation Block	92%

For this proposal, we consider the implementation that we presented in [35]. In that work, we presented a GPU-based implementation that efficiently filters audio buffers of a size that is much smaller than that of the filters. To this end, the approach leverages the algorithm presented in [51], [66], based on the uniformly-partitioned fast convolution algorithm using the overlap-save technique. This means that the room compensation filters are uniformly split into blocks of the same size as that of the input-data buffer. Thus, denoting the size of the filters $\widetilde{f}_{jn}$ by l_f, we can define $B = \frac{l_f}{L}$ as the number of partitions of the room compensation filter. The GPU-based implementation of this room compensation stage is described in [35]. Moreover, CUDA-like pseudo-codes of this stage are detailed in the dissertation [67]. Table IV shows the average percentage of the total processing time that each kernel requires. To carry out this measurement, we set the number of sound sources to $M = 300$ and the number of partitions to $B = 6$. As can be observed, most of the time is consumed by the 9216 filtering operations. If the room compensation block is not considered, the highest computational demands occur at Kernel 1 since the computation of τ_{mn} depends on a cosine operation, which is computationally expensive.

VI. COMPUTATIONAL PERFORMANCE

We have evaluated our WFS system on an NVIDIA device K20Xm board that belongs to the Kepler-based family of GPUs [53], [68], owns a compute capability of 3.5, and is composed of a read-only cache memory and 2688 cores. We set the audio card to provide blocks of $L \in \{64, 256, 1024\}$ samples with a sample frequency $f_s = 44.1$ kHz. This means that t_{buff} takes the values 1.45 ms, 5.80 ms, and 23.22 ms for the different buffer sizes. We assess our WFS system by gradually increasing the number of sources M while measuring t_{proc} for the target environments. Keep in mind that our WFS system operates under real-time conditions as long as $t_{\text{proc}} < t_{\text{buff}}$.

The performance of the WFS system has been analyzed for different variables: size of audio buffers L, length of room compensation filters l_f, number of partitions B that can be executed at the room

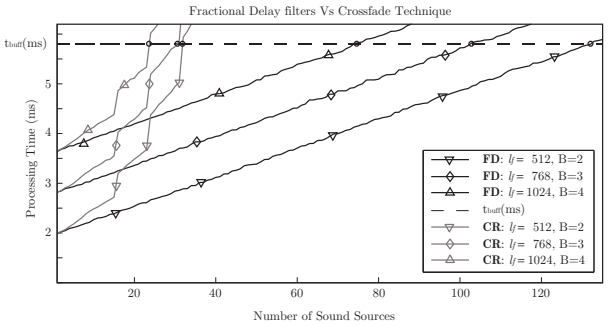

Fig. 9. Performance of the WFS system using a buffer size of $L = 256$ samples for three room compensation filter lengths: 512, 768, and 1024. The black curves synthesizes the virtual sound sources by using NM 9th-order fractional delay filters and the grey curves synthesizes the virtual sound sources by using the crossfade technique.

compensation filters to reduce latency, possible accuracy requirements in the sound localization (use of 9th-order fractional delay filters), and use of any trajectory resolution for sound source movements (use of the crossfade technique).

Fig. 9 shows the variation of the processing time t_{proc} when the WFS system uses an input-data buffer of $L = 256$ samples. The computational performance has been measured when the filters $\widetilde{f}_{jn}$ consist of 512, 768, and 1024 coefficients, which imply a partition of the filter into $B = 2$, 3, and 4 partitions, respectively. Fig. 9 represents the time t_{proc} for the case when WFS uses MN 9th-order fractional filters to render the virtual sound source (low trajectory resolution and high accuracy), see the black curves. As the number of partitions increases, the number of sound sources that can be rendered in real time decreases in approximately the same proportion.

The use of fractional delays provides sound source synthesis with better accuracy, but it also requires that sound source displacements be carried out between closer points in order to avoid high variations of amplitudes and delays (see Table II). The consequence of limiting the sound source movement to a short displacement also constrains the speed of the sound source since this factor has a direct relation to the trajectory resolution.

In order to synthesize any sound source speed (i.e. to shift between any two positions in the WFS system), we need to carry out the crossfade technique. This way audible artifacts are avoided. Fig. 9 shows also the time t_{proc} for this scenario, see the grey curves. The crossfade technique involves more than twice the number of operations compared with the fractional delay filters. As a result, it reduces the maximum number of sound sources that can be reproduced in real time by a maximum factor of six. Note that slopes can be observed in grey curves when the sound sources in the WFS system are 15,

23, and 31. Up to these values, all parallel resources are being used since the curve is quite flat. From there on, the slope becomes steeper because there is data that cannot be processed in parallel and must wait for other data to be computed. In Section II-A, the GPU architecture was shown to be composed of multiple SMs. Before the computation begins, the data is distributed among the SMs. Specifically, when there are 16 sound sources, there are few SMs that have more data to process than others. As the number of sound sources increases, the parallel resources are efficiently used. However, when the number of sound sources reaches 24, the volume of data does not match the parallel resources in the SMs. The same occurs with 32 sound sources.

Tables V and VI repeat the above experimentation with $L = 64$ and $L = 1024$ and room compensation filters of different sizes. Column 3 shows the number of partitions B of size $2L$ that is produced in each one of the 96 $\times$ 96 filters $\widetilde{f}_{jn}$ that make up our inverse filter bank (Fig. 4). Column 5 indicates the maximum number of sound sources that can be rendered by the system in real time. The time t_{proc} used by the GPU to compute the target number of sound sources is shown in column 6. In these tables, the times t_{proc} for $L = 256$ are extracted from Fig. 9. Fig. 10 illustrates the ratio that relates the maximum number of sound sources that can be rendered in real time using fractional delay filters and the crossfade technique for different numbers of partitions B and sizes of buffer L. In both cases, as L increases, the time t_{buff} increases and this allows a larger number of sound sources to be achieved in real time. However, as B increases, the number of operations increases, which implies a larger t_{proc}, and thus a smaller number of sound sources in real time.

In summary, we want to highlight that the decision to use the crossfade technique must be thoroughly assessed since it requires a great amount of computational resources and significantly penalizes the performance of a WFS system. Therefore, we recommend the use of WFS systems with a minimum trajectory resolution and Fractional Delay Filters.

VII. Conclusion

The use of GPUs in large-scale audio systems is gaining momentum. One of the audio systems that requires a great amount of processing power is Wave Field Synthesis with room compensation and moving virtual sound sources. In this paper, we have analyzed how a state-of-the-art GPU can be used to develop a high-performance solution for this problem.

In terms of accuracy of the sound source localization, we have studied the impact of synthesizing a moving sound source via time-varying fractional delay filters. Our results show that filters of this kind offer the best approach for the theoretical WFS signal. Specifically, the best results were obtained with

TABLE V

MAXIMUM PERFORMANCE THAT THE WFS SYSTEM ACHIEVES WHEN THE SOUND SOURCES ARE SYNTHESIZED USING 9-TH ORDER FRACTIONAL DELAY FILTERS. BUFFER SIZES OF $L = 64$ AND $L = 1024$ AND DIFFERENT SIZE OF ROOM COMPENSATION FILTERS l_f ARE EVALUATED.

L	t_{buff} (ms)	B	Size of $\widetilde{f}_{jn}$	Max. Sources Real Time	t_{proc} (ms) Real Time
64	1.451	1	64	92	1.433
		2	128	68	1.449
		3	192	51	1.449
		4	256	36	1.442
		5	320	20	1.436
		6	384	5	1.446
256	5.805	1	256	198	5.752
		2	512	132	5.793
		3	768	102	5.770
		4	1024	73	5.744
		5	1280	45	5.750
		6	1536	18	5.791
1024	23.22	1	1024	262	23.175
		2	2048	162	23.062
		3	3072	130	23.198
		4	4096	94	23.092
		5	5120	60	23.199
		6	6144	25	23.121

the truncated Lagrange interpolation technique.

When shifting a virtual sound source between two points, audible non-linear artifacts can appear due to the large changes in amplitudes and delays that must be rendered by the WFS system. To avoid this, we have also evaluated the WFS system using the crossfade technique, which more than doubles the number of operations in comparison with the use of fractional delay filters.

In addition, we have improved our previous GPU-based implementation by filtering audio buffers that are smaller than the size of room compensation filters. This allows the design of room compensation filters with a large number of coefficients and thus higher spatial sound quality. In order to implement the system efficiently, we have convolved the WFS pre-equalization filter h with the room compensation filters before starting the real-time processing.

Finally, our implementation exploits the resources of GPUs with Kepler architecture, which can currently be found in new-generation mobile devices such as modern tablets.

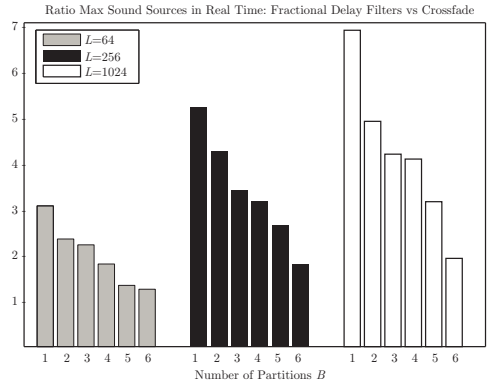

Fig. 10. Ratio that relates the maximum number of sound sources that can be rendered in real time using fractional delay filters and crossfade for different number of blocks B and sizes of buffer L.

TABLE VI

MAXIMUM PERFORMANCE THAT THE WFS SYSTEM ACHIEVES WHEN THE SOUND SOURCES ARE SYNTHESIZED USING THE CROSSFADE TECHNIQUE. BUFFER SIZES OF $L = 64$ AND $L = 1024$ AND DIFFERENT SIZE OF ROOM COMPENSATION FILTERS l_f ARE EVALUATED.

L	t_{buff} (ms)	B	Size of $\widetilde{f}_{jn}$	Max. Sources Real Time	t_{proc} (ms) Real Time
64	1.451	1	64	30	1.247
		2	128	29	1.442
		3	192	23	1.326
		4	256	20	1.447
		5	320	15	1.436
		6	384	4	1.446
256	5.805	1	256	38	5.642
		2	512	31	5.027
		3	768	30	5.747
		4	1024	23	5.336
		5	1280	17	5.776
		6	1536	10	5.786
1024	23.22	1	1024	38	21.361
		2	2048	33	23.149
		3	3072	31	22.468
		4	4096	23	20.500
		5	5120	19	22.917
		6	6144	13	23.132

GPU code of this work and instructions for compiling it together with multimedia materials are available at http://www.gtac.upv.es/enlaces.asp.

ACKNOWLEDGMENT

This work was initiated when Dr. Jose A. Belloch was a visiting postdoctoral researcher at Aalto University in 2015 and was finished during his stay at Universidad Complutense de Madrid in 2016.

REFERENCES

[1] S. Spors, H. Wierstorf, A. Raake, F. Melchior, M. Frank, and F. Zotter, "Spatial sound with loudspeakers and its perception: A review of the current state," *Proc. IEEE*, vol. 101, no. 9, pp. 1920–1938, Sept. 2013.

[2] A. Berkhout, "A holographic approach to acoustic control," *J. Audio Eng. Soc.*, vol. 36, no. 12, pp. 977–995, Dec. 1988.

[3] A. Berkhout, D. de Vries, and P. Vogel, "Acoustic control by wave field synthesis," *J. Acoust. Soc. Am.*, vol. 93, no. 5, pp. 2764–2778, May 1993.

[4] P. Vogel, "Application of Wave Field Synthesis in Room Acoustics," Ph.D. dissertation, Delft University of Technology, Delft, Netherlands, 1993.

[5] E. Start, "Direct sound enhancement by Wave Field Synthesis," Ph.D. dissertation, Delft University of Technology, Delft, Netherlands, 1997.

[6] E. Verheijen, "Sound reproduction by Wave Field Synthesis," Ph.D. dissertation, Delft University of Technology, Delft, Netherlands, 1997.

[7] J.-J. Sonke, "Variable acoustics by Wave Field Synthesis," Ph.D. dissertation, Delft University of Technology, Delft, Netherlands, 2000.

[8] E. Hulsebos, "Auralization using Wave Field Synthesis," Ph.D. dissertation, Delft University of Technology, Delft, Netherlands, 2004.

[9] E. Hulsebos, D. de Vries, and E. Bourdillat, "Improved microphone array configurations for auralization of sound fields by Wave-Field Synthesis," *J. Audio Eng. Soc.*, vol. 50, no. 10, pp. 779–790, Oct. 2002.

[10] S. Spors, H. Buchner, and R. Rabenstein, "Efficient active listening room compensation for Wave Field Synthesis," in *Proc. 116th AES Conv.*, Berlin, Germany, May 2004.

[11] J. J. Lopez, A. Gonzalez, and L. Fuster, "Room compensation in wave field synthesis by means of multichannel inversion," in *Proc. IEEE Workshop Appl. Signal Process. Audio Acoust. (WASPAA)*, New Paltz, NY, USA, Oct. 2005, pp. 146–149.

[12] T. I. Laakso, V. Välimäki, M. Karjalainen, and U. K. Laine, "Splitting the unit delay: Tools for fractional delay filter design," *IEEE Signal Process. Mag.*, vol. 13, no. 1, pp. 30–60, Jan. 1996.

[13] M. Karjalainen, T. Paatero, J. Pakarinen, and V. Välimäki, "Special digital filters for audio reproduction," in *Proc. 32nd AES Conf.*, Hillerød, Denmark, Sept. 2007.

[14] J. Dattorro, "Effect design—Part 2: Delay-line modulation and chorus," *J. Audio Eng. Soc.*, vol. 45, no. 10, pp. 764–788, Oct. 1997.

[15] U. Zölzer, *DAFX: Digital Audio Effects*, 2nd ed. Chichester, UK: Wiley, 2011.

[16] V. Välimäki, "Discrete-time modeling of acoustic tubes using fractional delay filters," Ph.D. dissertation, Helsinki Univ. Tech., Espoo, Finland, Dec. 1995.

[17] V. Välimäki, J. Pakarinen, C. Erkut, and M. Karjalainen, "Discrete-time modelling of musical instruments," *Reports on Progress in Physics*, vol. 69, no. 1, pp. 1–78, Jan. 2006.

[18] A. Franck, A. Gräfe, T. Korn, and M. Strauss, "Reproduction of moving sound sources by wave field synthesis: an analysis of artifacts." in *Proc. 32nd AES Conf.*, Hillerød, Denmark, Sept. 2007.

[19] A. Franck, K. Brandenburg, and U. Richter, "Efficient delay interpolation for Wave Field Synthesis," in *Proc. 125th AES Conv.*, San Francisco, CA, USA, Oct. 2008.

[20] A. Franck, "Efficient algorithms for arbitrary sample rate conversion with application to wave field synthesis," Ph.D. dissertation, Technical University of Ilmenau, Ilmenau, Germany, 2012.

[21] J. Ahrens, M. Gier, and S. Spors, "Perceptual assessment of delay accuracy and loudspeaker misplacement in wave field synthesis," in *Proc. 128th AES Conv.*, London, UK, May 2010.

[22] C. Salvador, "Wave field synthesis using fractional order systems and fractional delays," in *Proc. 128th AES Conv.*, London, UK, May 2010.

[23] F. Winter and S. Spors, "On fractional delay interpolation for local wave field synthesis," in *Proc. European Signal Process. Conf. (EUSIPCO)*, Budapest, Hungary, Sept. 2016, pp. 2415–2419.

[24] L. Savioja, "Real-time 3D finite-difference time-domain simulation of low- and mid-frequency room acoustics," in *Proc. Int. Conf. Digital Audio Effects (DAFX)*, Graz, Austria, Sept. 2010, pp. 1–8.

[25] A. Southern, D. Murphy, G. Campos, and P. Dias, "Finite difference room acoustic modelling on a general purpose graphics processing unit," in *Proc. 128th AES Conv.*, London, UK, May 2010.

[26] C. J. Webb and S. Bilbao, "Computing room acoustics with CUDA—3D FDTD schemes with boundary losses and viscosity," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Prague, Czech Republic, May 2011, pp. 317–320.

[27] B. Hamilton and C. J. Webb, "Room acoustics modelling using GPU-accelerated finite difference and finite volume methods on a face-centered cubic grid," in *Proc. Int. Conf. Digital Audio Effects (DAFX)*, Maynooth, Ireland, Sept. 2013, pp. 1–8.

[28] L. Savioja, V. Välimäki, and J. O. Smith, "Real-time additive synthesis with one million sinusoids using a GPU," in *Proc. 128th AES Conv.*, London, UK, May 2010.

[29] ——, "Audio signal processing using graphics processing units," *J. Audio Eng. Soc.*, vol. 59, no. 1–2, pp. 3–19, Jan.-Feb. 2011.

[30] S. Bilbao and C. J. Webb, "Physical modeling of timpani drums in 3D on GPGPUs," *J. Audio Eng. Soc.*, vol. 61, no. 10, pp. 737–748, Oct. 2013.

[31] R. Bradford, J. Ffitch, and R. Dobson, "Real-time sliding phase vocoder using a commodity GPU," in *Proc. Int. Computer Music Conf. (ICMC)*, Huddersfield, UK, Aug. 2011, pp. 587–590.

[32] J. Lorente, G. Piñero, A. Vidal, J. Belloch, and A. Gonzalez, "Parallel implementations of beamforming design and filtering for microphone array applications," in *Proc. European Signal Process. Conf. (EUSIPCO)*, Barcelona, Spain, Aug. 2011, pp. 501–505.

[33] N. Tsingos, W. Jiang, and I. Williams, "Using programmable graphics hardware for acoustics and audio rendering," *J. Audio Eng. Soc.*, vol. 59, no. 9, pp. 628–646, Sept. 2011.

[34] J. A. Belloch, M. Ferrer, A. Gonzalez, F. Martinez-Zaldivar, and A. M. Vidal, "Headphone-based virtual spatialization of sound with a GPU accelerator," *J. Audio Eng. Soc.*, vol. 61, no. 7/8, pp. 546–561, Jul.-Aug. 2013.

[35] J. A. Belloch, A. Gonzalez, F. Martinez-Zaldivar, and A. M. Vidal, "Multichannel massive audio processing for a generalized

crosstalk cancellation and equalization application using GPUs," *Integrated Computer-Aided Engineering*, vol. 20, no. 2, pp. 169–182, Apr. 2013.

[36] J. A. Belloch, B. Bank, L. Savioja, A. Gonzalez, and V. Välimäki, "Multi-channel IIR filtering of audio signals using a GPU," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Florence, Italy, May 2014, pp. 6692–6696.

[37] J. A. Belloch, J. Parker, L. Savioja, A. Gonzalez, and V. Välimäki, "Dynamic range reduction of audio signals using multiple allpass filters on a GPU accelerator," in *Proc. European Signal Process. Conf. (EUSIPCO)*, Lisbon, Portugal, Sept. 2014, pp. 890–894.

[38] M. Schneider, F. Schuh, and W. Kellermann, "The generalized frequency-domain adaptive filtering algorithm implemented on a GPU for large-scale multichannel acoustic echo cancellation," in *Proc. Speech Communication; 10. ITG Symposium.*, Braunschweig, Germany, Sept. 2012, pp. 1–4.

[39] J. Lorente, A. Gonzalez, M. Ferrer, J. A. Belloch, M. De Diego, G. Piñero, and A. M. Vidal, "Active noise control using Graphics Processing Units," in *Proc. Int. Congr. Sound Vibr.*, Vilnius, Lithuania, July 2012, pp. 1 – 8.

[40] J. Lorente, M. Ferrer, M. De Diego, and A. Gonzalez, "GPU implementation of multichannel adaptive algorithms for local active noise control," *IEEE Trans. Audio Speech Lang. Process.*, vol. 22, no. 11, pp. 1624–1635, Nov. 2014.

[41] L. Romoli, P. Peretti, S. Cecchi, L. Palestini, and F. Piazza, "Real-time implementation of wave field synthesis for sound reproduction systems," in *Proc. IEEE Asia Pacific Conf. Circ. Syst. (APCCAS)*, Kuala Lumpur, Malaysia, Nov. 2008, pp. 430–433.

[42] D. Theodoropoulos, G. Kuzmanov, and G. Gaydadjiev, "A minimalistic architecture for reconfigurable WFS-based immersive-audio," in *Proc. 2010 Int. Conf. Reconfigurable Computing and FPGAs (ReConFig)*, Cancun, Mexico, Dec. 2010.

[43] ——, "Multi-core platforms for beamforming and Wave Field Synthesis," *IEEE Trans. Multimedia*, vol. 3, no. 2, pp. 235–245, Apr. 2011.

[44] R. Ranjan and W. S. Gan, "Fast and efficient real-time GPU based implementation of wave field synthesis," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Florence, Italy, May 2014, pp. 7550–7554.

[45] A. Lattanzi, E. Ciavattini, S. Cecchi, L. Romoli, and F. Ferrandi, "Real-time implementation of Wave Field Synthesis on NU-Tech framework using CUDA technology," in *Proc. 128th AES Conv.*, London, UK, May 2010.

[46] A. Lattanzi, F. Bettarelli, and S. Cecchi, "NU-Tech: The entry tool of the hArtes toolchain for algorithms design," in *Proc. 124th AES Conv.*, Amsterdam, The Netherlands, May 2008.

[47] K20, "NVIDIA Kepler Architecture," http://www.nvidia.com/content/PDF/kepler/NVIDIA-Kepler-GK110-Architecture-Whitepaper.pdf, 2014, (accessed 2016 Oct. 11).

[48] Jetson, "Mobile GPU: Jetson," https://developer.nvidia.com/jetson-tk1, 2015, (accessed 2016 Oct. 11).

[49] Nexus, "Google's nexus 9," http://blogs.nvidia.com/blog/2014/10/17/nvidia-tegra-k1-google-nexus-9/, 2015, (accessed 2016 Oct. 11).

[50] J. A. Belloch, M. Ferrer, A. Gonzalez, J. Lorente, and A. M. Vidal, "GPU-based WFS systems with mobile virtual sound sources and room compensation," in *Proc. 52nd AES Conf.*, Guildford, UK, Sept. 2013.

[51] B. D. Kulp, "Digital equalization using Fourier transform techniques," in *Proc. 85th AES Conv.*, Los Angeles, CA, USA, Nov. 1988.

[52] M. J. Flynn, "Some computer organizations and their effectiveness," *IEEE Trans. Comput.*, vol. 21, no. 9, pp. 948–960, Sep. 1972.

[53] "Nvidia CUDA Developer Zone," https://developer.nvidia.com/cuda-zone, (accessed 2016 Oct. 11).

[54] "Audio and Communications Signal Processing Group at Universitat Politecnica de Valencia," *http://www.gtac.upv.es*.

[55] M. M. Boone, E. N. G. Verheijen, and P. F. Van Tol, "Spatial sound-field reproduction by wave-field synthesis," *J. Audio Eng. Soc.*, vol. 43, no. 12, pp. 1003–1012, Dec. 1995.

[56] S. Spors, A. Kuntz, and R. Rabenstein, "An approach to listening room compensation with wave field synthesis," in *Proc. 24th AES Conf.*, Banff, Canada, May 2003.

[57] S. Spors and J. Ahrens, "Analysis and improvement of pre-equalization in 2.5-dimensional wave field synthesis," in *Proc. 128th AES Conv.*, London, UK, May 2010.

[58] L. Fuster, J. J. Lopez, A. Gonzalez, and P. Faus, "Time and frequency domain room compensation applied to wave field synthesis," in *Proc. Int. Conf. Digital Audio Effects (DAFX)*, Madrid, Spain, Sept. 2005, pp. 1–6.

[59] O. Kirkeby, P. A. Nelson, H. Hamada, and F. Orduna-Bustamante, "Fast deconvolution of multichannel systems using regularization," *IEEE Trans. Speech Audio Process.*, vol. 6, no. 2, pp. 189–194, Mar. 1998.

[60] M. Miyoshi and Y. Kaneda, "Inverse filtering of room acoustics," *IEEE Trans. Acoust, Speech Signal Process.*, vol. 36, no. 2, pp. 145–152, Mar. 1988.

[61] M. Gasparini, P. Peretti, S. Cecchi, L. Romoli, and F. Piazza, "Real-time reproduction of moving sound sources by wave field synthesis: Objective and subjective quality evaluation," in *Proc. 130th AES Conv.*, London, UK, May 2011.

[62] V. Välimäki and A. Haghparast, "Fractional delay filter design based on truncated Lagrange interpolation," *IEEE Signal Process. Lett.*, vol. 14, no. 11, pp. 816–819, Nov. 2007.

[63] E. Hermanowicz, "Explicit formulas for weighting coefficients of maximally flat tunable FIR delayers," *Electron. Lett.*, vol. 28, no. 2, pp. 1936–1937, Sept. 1992.

[64] H. David, *The Method of Paired Comparisons*. London, UK: Griffin, 1988.

[65] H. Kuttruff, *Room Acoustics*, 5th ed. Abingdon, UK: Taylor & Francis, Oct. 2000.

[66] F. Wefers and M. Vorländer, "Optimal filter partitions for real-time FIR filtering using uniformly-partitioned FFT-based convolution in the frequency-domain," in *Proc. Int. Conf. Digital Audio Effects (DAFX)*, Paris, France, Sept. 2011, pp. 155–161.

[67] J. A. Belloch, "Performance Improvement of Multichannel Audio by Graphics Processing Units," Ph.D. dissertation, Universitat Politècnica de València, Valencia, Spain, 2014.

[68] *CUDA Programming: A Developer's Guide to Parallel Computing with GPUs*, 1st ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2013.

Jose A. Belloch received the degree in Telecommunications Engineering in 2007, a Master's degree in Parallel and Distributed Computing in 2010 and a Ph.D. degree in Computer Science in 2014. All of them at the Universitat Politècnica de València (Valencia, Spain). His PhD thesis was recognized with the Extraordinary PhD Thesis Award. Between 2010 and 2014 he was a PhD grant holder of the Spanish Ministry of Science and Innovation. Between 2014 and 2016 he held a postdoctoral position at the HPC&A group of the Universitat Jaume I de Castellón (Castellón de la Plana, Spain), where he was involved in two EU projects (EXA2GREEN and INTERTWinE). In July 2016 he was awarded with a Post-Doctoral Fellowship of the Regional Government of Valencia in order to carry out research at Universitat Jaume I de Castellón in collaboration with the Universidad Complutense de Madrid (Madrid, Spain). He was a visiting researcher at Department of Signal Processing and Acoustics, Aalto University School of Electrical Engineering (Espoo, Finland) as a pre-doc researcher in 2013 and as a post-doc researcher in 2015. He carried out an internship at Department of Measurement and Information Systems, Budapest University of Technology and Economics (Budapest, Hungary). His research interests are centered in applying the new parallel architectures into signal processing algorithms. He has developed several real-time audio applications related with multichannel massive filtering, binaural sound, wave field synthesis systems, and sound source localization using General Purpose Graphic Processing Units and ARM architectures.

Alberto Gonzalez works as Professor at the Universitat Politècnica de València (UPV), Spain. He attended the Universitat Politècnica Catalunya, Spain, where he graduated in Telecommunications Engineering with highest honours. In 1997 he was awarded a Doctorate (PhD), magna cum laude, from the UPV. During 1995 he worked as a visiting researcher in the Institute of Sound and Vibration Research (ISVR) at the University of Southampton, UK. Currently, he is the head of the research group in Audio and Communications Digital Signal Processing.

Alberto Gonzalez has published over 100 papers in international technical journals and renowned conferences in the fields of signal processing and applied acoustics. He is a member of the senate of the UPV and serves as Dean of the Telecommunications Engineering School since June 2012. His current research interests include optimization of computation methods for detection and decoding in digital communications and distributed sound signal processing.

Enrique S. Quintana-Ortí received his bachelor and Ph.D. degrees in Computer Sciences from the Universitat Politècnica de València (Spain) in 1992 and 1996. Currently he is professor in Computer Architecture in the Universitat Jaume I de Castellón (Spain). He has published more than 100 papers in international conferences and journals, and has contributed to software libraries like SLICOT and libflame. His research interests include parallel programming, linear algebra, power consumption, as well as advanced architectures and hardware accelerators.

Miguel Ferrer graduated in Telecommunications Engineering in the year 2000 at the Universitat Politècnica de València (UPV), Spain. He was collaborating with the Audio and Communications Signal Processing Group (GTAC) since a year before, performing a six months research stay in the Instituto de Investigación Aplicada al Automóvil, Tarragona, Spain (Automobile Applied Research Institute). Subsequently, he was awarded several grants offered both by the Communications Department of the UPV and by the Research, Development and Innovation Vice-Chancellery of the same university, enabling him to start his Doctorate studies and to collaborate in different research projects within the GTAC. During this period he has authored or co-authored over sixty papers related with signal processing in renowned journals and conferences. Since 2005 he works as an assistant lecturer in the Communications Department of the UPV. His research activity is focused on the study of adaptive algorithms and its application to audio digital processing and noise active control, a subject about which he developed his doctoral thesis.

Vesa Välimäki (S'90–M'92–SM'99–F'15) received the M.Sc. in Technology and the Doctor of Science in Technology degrees in electrical engineering from the Helsinki University of Technology (TKK), Espoo, Finland, in 1992 and 1995, respectively.

He was a Postdoctoral Research Fellow at the University of Westminster, London, UK, in 1996. In 1997–2001, he was a Senior Assistant at TKK. In 2001–2002, he was a Professor of signal processing at the Pori unit of the Tampere University of Technology, Pori, Finland. In 2006–2007, he was the Head of the TKK Laboratory of Acoustics and Audio Signal Processing. In 2008–2009, he was a Visiting Scholar at Stanford University, CA, USA. He is currently a Professor of audio signal processing at Aalto University, Espoo, Finland. His research interests are related to signal processing techniques in audio and music technology.

Prof. Välimäki is a Fellow of the Audio Engineering Society and a Life Member of the Acoustical Society of Finland. In 2007–2013 he was a Member of the Audio and Acoustic Signal Processing Technical Committee of the IEEE Signal Processing Society and is currently an Associate Member. He is a Founding Member of the EURASIP Special Area Team in acoustic, sound and music signal processing (2015–). He served as an Associate Editor of the IEEE SIGNAL PROCESSING LETTERS in 2005–2009 and of the IEEE TRANSACTIONS ON AUDIO, SPEECH AND LANGUAGE PROCESSING in 2007–2011. He was in the Editorial Board of the *Research Letters in Signal Processing*, the *Journal of Electrical and Computer Engineering*, and *The Scientific World Journal*. He was the Lead Guest Editor of a special issue of the IEEE SIGNAL PROCESSING MAGAZINE in 2007 and of a special issue of the IEEE TRANSACTIONS ON AUDIO, SPEECH AND LANGUAGE PROCESSING in 2010. In 2015, he was a Guest Editor of the special issue of the IEEE SIGNAL PROCESSING MAGAZINE on signal processing techniques for assisted listening. In 2015–2016, he was the Lead Guest Editor of the special issue of *Applied Sciences* on audio signal processing. In 2008, he was the General Chair of DAFX-08, the 11th International Conference on Digital Audio Effects. He has been a Senior Area Editor of the IEEE/ACM TRANSACTIONS ON AUDIO, SPEECH AND LANGUAGE PROCESSING since 2015.