Fractionally-addressed delay lines

Davide Rocchesso

Abstract— **While traditional implementations of variable-length digital delay lines are based on a circular buffer accessed by two pointers, we propose an implementation where a single fractional pointer is used both for read and write operations. On modern general-purpose architectures, the proposed method is nearly as efficient as the popular interpolated circular buffer, and it behaves well for delay-length modulations commonly found in digital audio effects. The physical interpretation of the new implementation shows that it is suitable for simulating tension or density modulations in wave-propagating media.**

I. Introduction

The digital delay line is a fundamental component of many signal processing architectures in several application fields. In audio signal processing, delay lines are used to implement audio effects, such as reverberation or pitch shifting, or to model wave propagation in physical models of musical instruments.

Since the early days of computer music, the delay line has been proposed as a building block in software synthesis languages such as Music V [1]. In those implementations, a fixed delay of an integer number D of samples was implemented by means of a circular queue having length D. At each time sample, a read and then a write operations were performed at the location pointed by the single circulating pointer[1]. In order to have fractional lengths, linear interpolation between adjacent memory locations right behind the pointer was introduced.

Since dynamic variations of the delay length are required by many important applications, such as pitch shifting, variable delay lines were introduced in several signal processing environments and languages. The classic implementation of the variable-length digital delay line uses a circular buffer, which is accessed by a writing pointer followed by a reading pointer[2] [2]. When the delay length has to be made variable, the relative distance between the reading pointer and the writing pointer is varied sample by sample. In order to allow for fractional lengths and click-free length modulation, some form of interpolation has to be applied at the reading point [3], [4], [5]. The following properties should be ensured by the interpolation device:

1. flat magnitude frequency response

2. linear phase response

3. transient-free response to variations of the delay.

FIR filters, usually in the form of Lagrange interpolators [4], [3], are widely used. Even though they can not satisfy property 1, they are certainly compliant to property 3 and can also satisfy property 2 to a great extent on a wide frequency range [4]. On the other hand, allpass filters, often designed to have a maximally-flat delay response at low frequencies, satisfy property 1 exactly but are quite nonlinear in their phase response [4]. Moreover, a rather complicated structure has to be devised in order to attain property 3 by means of allpass filters [6], [7].

All of the prior realizations, as far as a fixed delay length is considered, are linear and time-invariant systems, thus being completely described by their frequency response. Vice versa, we are proposing a realization which is time-varying even in the case of constant delay. This realization was first proposed and implemented by the author as part of a thesis work [8]. Afterwards, improved implementations and input-output analyses were sparsely presented at some conferences [9], [10]. This paper systematizes the main ideas and results, and gives some hints to use the technique in musical applications.

In section II we recall the classic FIR realization of the delay line and show the magnitude and phase responses in the case of Lagrange quadratic interpolation. In particular, we show how length modulation affects the spectrum of an incoming signal. In section III we describe the novel Fractionally-Addressed Delay (FAD) line, and in section IV we analyze its input-output behavior in terms of signal-to-error ratio and transient response to delay variations. In section V we interpret the FIR and FAD lines as physical wave-propagating media, and we illustrate the application of the FAD line in physical modeling of resonators. In section VI the FIR and FAD lines are compared in terms of performance in general-purpose superscalar architectures, and the convenience of using FAD lines in digital audio effects is briefly discussed.

II. Length-Modulated Delay Lines

Nowadays, a plethora of techniques is available to design FIR filters to be used as interpolators in fractional delay lines [4]. Such interpolators can be designed in order to be optimal in one of several senses. However, most of the optimal FIR interpolators are based on sets of precomputed coefficients or use windowing to compute the coefficients online. On the other hand, the coefficients of Lagrange interpolators are easy to compute and the resulting filter has the desirable property of having maximally-flat magnitude in low frequency. Moreover, the magnitude response is guaranteed to be less than unity at any frequency, and this is a fundamental property if the fractional delays have to

D. Rocchesso is with the Università di Verona - Dipartimento Scientifico e Tecnologico, Strada Le Grazie, 15 - 37134 Verona, Italy, E-mail: rocchesso@sci.univr.it. Phone: ++39.045.8027979, FAX: ++39.045.8027982.

[1]This is the way the `delay` unit generator is still implemented in the popular sound processing language Csound [1].

[2]This is the way the `vdelay` unit generator is implemented in Csound.

be used within feedback structures, such as physical models. For these reasons, Lagrange-interpolated delay lines are very popular in sound synthesis, digital audio effects, and wherever the delay length has to be modulated at runtime. In this section, we consider the Lagrange interpolator as a reference case, focusing on the second order FIR filter used e.g., in [11]. We introduce an approximate analysis of the length-modulated delay that will be useful to understand the dynamic behavior of the new FAD line.

A. Lagrange-interpolated delay lines

A Lagrange interpolator can be characterized at any fractional delay by its magnitude and phase delay responses [4]. At any frequency f and for a given interval of fractional delays, the interpolator has a magnitude response ranging from $A_{\min}$ to $A_{\max}$ and a phase delay ranging from $\tau_{\min}$ to $\tau_{\max}$. For example, a second-order (quadratic) interpolator can be constructed by using the coefficients [6]

$$\begin{aligned} h_0 &= d(1+d)/2 \\ h_1 &= (1+d)(1-d) \\ h_2 &= -d(1-d)/2 \end{aligned} \tag{1}$$

in the FIR transfer function $H(z) = h_0 + h_1 z^{-1} + h_2 z^{-2}$. If we enforce $d \in [-1.0 \ldots 1.0]$, the magnitude is constrained to be less than one, and the phase delay deviates roughly sinusoidally around a straight line going from 0 to 2 samples. Figures 1.a and 1b show the magnitude and excess phase-delay responses at $f = F_s/4$ as functions of $D \stackrel{\triangle}{=} 1 - d$. Ideally, one would like the excess phase delay to be zero, in such a way that the ideal phase delay is equal to D.

B. Approximate analysis of length-modulated delay lines

If the parameter d is continuously varied around 0, it is clear that frequency-dependent amplitude and phase modulations are both applied to the signal. In order to understand how these modulations affect the spectrum, we assume that the magnitude and phase delay vary sinusoidally[3], and that the frequency of the magnitude sine is twice the frequency ω_M of the phase-delay sine. This latter assumption is justified by the aspect of figure 1, where there are two minima in figure 1.a and one minimum in figure 1.b. If the input is a cosine wave at frequency ω_0, the output signal takes form

$$v_d = A_m \left(1 + m \cos 2\omega_M t\right) \cos\left(\omega_0 t + \tau_{\max}\omega_0 \sin \omega_M t\right), \tag{2}$$

where

$$m = \frac{1 - A_{\min}}{1 + A_{\min}}, \tag{3}$$

$$A_m = \frac{1 + A_{\min}}{2}, \tag{4}$$

and $A_{\min}$ and $\tau_{\max}$ are the minimum magnitude and maximum delay for the given interval of variability of d and

[3]This is quite a strong assumption, but the goal of this analysis is just to outline a qualitative behavior of the spectrum. The actual behavior will be slightly different, especially because of deviations from zero-mean and sinusoidal variation of d.

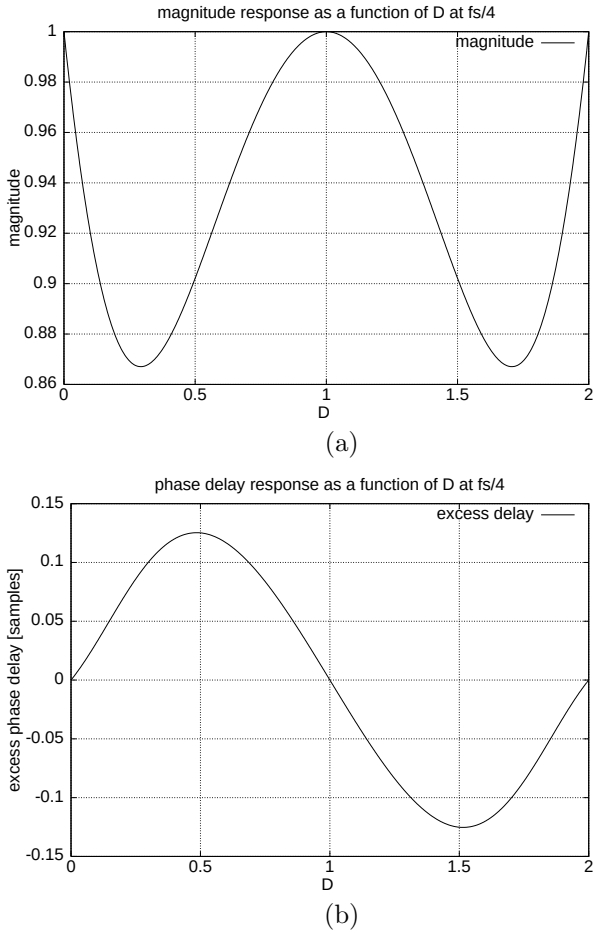


Fig. 1
Magnitude (a) and excess phase-delay (b) responses for a quadratic Lagrange interpolator as functions of the parameter $D = 1 - d$ at frequency $f = F_s/4$

frequency ω_0. If we enforce $d \in [-0.5 \ldots 0.5]$, the extremal magnitude and phase delay responses as functions of frequency are plotted in figures 2.a and 2.b, respectively.

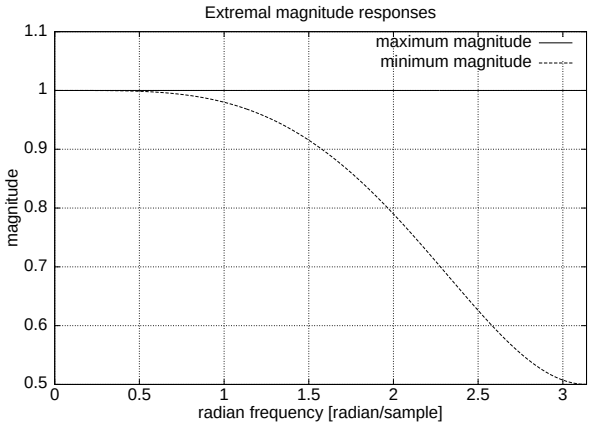
The amplitude modulation gives rise to a carrier and two side bands:

$$\begin{aligned} v_d &= A_m \cos\left(\omega_0 t + \tau_{\max}\omega_0 \sin \omega_M t\right) \\ &\quad + \frac{m A_m}{2} \cos\left((\omega_0 - 2\omega_M)t + \tau_{\max}\omega_0 \sin \omega_M t\right) \\ &\quad + \frac{m A_m}{2} \cos\left((\omega_0 + 2\omega_M)t + \tau_{\max}\omega_0 \sin \omega_M t\right). \end{aligned} \tag{5}$$

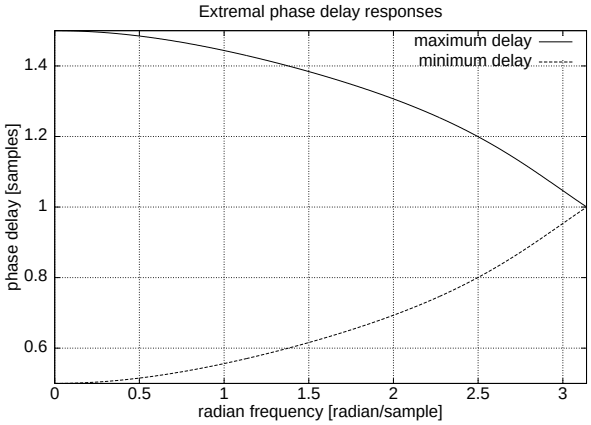
The phase modulation generates infinitely many sidebands of the three components of amplitude modulation. However, if we assume that the sidebands of order higher than two are negligible [4], the resulting signal can be expressed as

$$\begin{aligned} v_d &= A_0 \cos \omega_0 t \\ &\quad - A_1 \cos\left(\omega_0 - \omega_M\right)t + A_1 \cos\left(\omega_0 + \omega_M\right)t \end{aligned} \tag{6}$$

[4]This assumption is justified by experimental observations.



(a)



(b)

Fig. 2
Extremal magnitude (a) and phase delay (b) responses for a quadratic Lagrange interpolator

$$+A_2 \cos(\omega_0 - 2\omega_M)t + A_2 \cos(\omega_0 + 2\omega_M)t + \dots ,$$

where the coefficients A_i are given by table I and J_i is the i-th order Bessel function of the first kind evaluated in $\tau_{\max}\omega_0$.

A_0	A_1	A_2
$A_m J_0 + m A_m J_2$	$A_m J_1 - \frac{m A_m}{2} J_1$	$A_m J_2 + \frac{m A_m}{2} J_0$

TABLE I
Amplitude of the sidebands of an amplitude- and phase-modulated signal

Figure 3 shows the magnitude of the carrier and two side modulation products as a function of frequency, where $D = 1$ ($d = 0$) and the sinusoidal length modulator has amplitude 0.5 (i.e. it modulates the central part of figure 1).

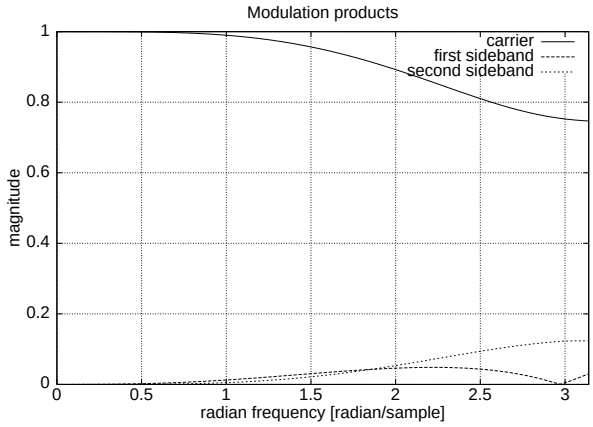


Fig. 3
Carrier and side modulation products as a function of carrier frequency

C. Delay-length dithering

The general trend expressed by figure 3 is that modulation acts as a lowpass filter on the carrier and as a kind of highpass filter for the side components. An interesting point is made by moving the center of modulation slightly away from 0 and observing that we get output signal components that are very similar to those of figure 3. On the other hand, a fixed-length interpolated delay would exhibit different magnitude responses at different values of d, in a range bounded by the curves of figure 2.a. As a consequence, we can use delay-length modulation in order to have a more uniform frequency response for different values of d, as it is often required in applications such as physical modeling (see section V). In other words, small-range modulation can be used as a sort of dithering in the delay length, and possibly put on top of long-range modulations. As an example, figure 4 shows the sonogram of the response of a non-modulated and modulated FIR line to a pulse train. In both cases, the delay length is slowly linearly increased in order to explore different fractional values of d. In figure 4.a the valleys corresponding to the minimum-magnitude curve of figure 2.a are visible as darker areas. In figure 2.b the darker areas are more uniformly spread along the horizontal axis, thus allowing magnitude compensation by fixed high-pass filtering.

A different issue is the audibility of the side components introduced by modulation. This depends on the strength of the side components and on their position relative to the carrier signal. If these components lay below the threshold of masking for the carrier signal [12] they are not audible. If the modulation frequency is around $60 - 80Hz$ the side components turn out to be below that threshold for partials laying in the first $2 - 3$kHz.

III. A Fractionally-Addressed Delay Line

An alternative realization of the delay line can be developed by observing that a single pointer is sufficient for both the read and write accesses. If the delay line has fixed in-

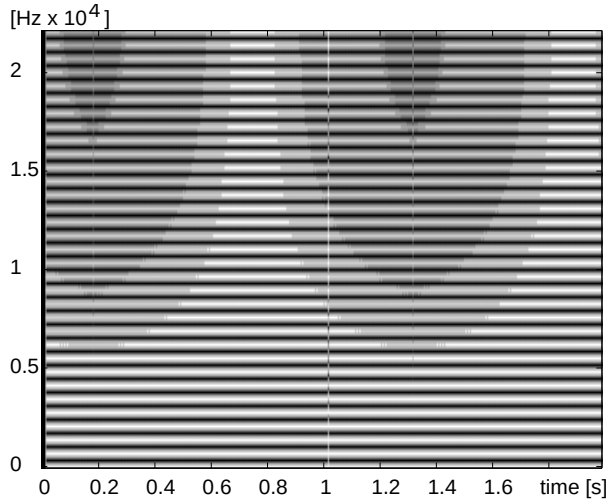


Fig. 4
SONOGRAM OF THE RESPONSE OF A NON-MODULATED (A) AND MODULATED (B) FIR DELAY LINE WITH LINEARLY-INCREASING LENGTH TO A PULSE TRAIN. HANNING WINDOWS OF 256 SAMPLES ARE USED IN ANALYSIS. MAGNITUDE (IN DB) IS SMALLER WHERE THE POINTS ARE DARKER.

teger length B, it is possible to use a buffer exactly B-cells long and a single pointer whose entry is first read and then written. At every sample the phase pointer is incremented to point to the following cell. In the same buffer we can also implement any delay which is an integer fraction B/I just by incrementing the pointer at steps of I samples. We are going to show how this scheme can be generalized to non-integer fractions of the total buffer length. The resulting technique can be seen as an extension of the table-lookup oscillator [13], [14], with the fundamental difference that every read is followed by one or more writes, in such a way that the waveform is continuously re-stored while being read.

Given a buffer size of B samples, and a sample rate F_s, a (fractional) increment of I samples gives a delay in seconds

```
delay = Q/P * lenbuf; %delay in samples
framelen = floor(delay);
for n=1:nframes
  bufout = resample(buffer, P, Q)'; %read
  output = [[output, bufout]];
  fwrite(fid_out, bufout, 'int16');
  bufin = fread(fid_in, framelen, 'int16');
  buffer = resample(bufin, Q, P); %write
end
```

Fig. 5
MATLAB CODE FOR A FRAME-BASED REALIZATION OF THE FAD LINE

equal to

$$T = \frac{B}{I \cdot F_s}. \tag{7}$$

Since this realization is related to waveform generation by fractional addressing [14], we call it the Fractionally-Addressed Delay (FAD) line.

A. Frame-Based Realization

The most immediate way to code the FAD line is by means of frame-based processing, e.g., using the `resample` function of the MATLAB Signal Processing Toolbox[5]. The code chunk of figure 5 is intended to be inserted into the framework of a frame-based digital audio effect, as prescribed by the COST-G6 action on Digital Audio Effects (DAFX)[6] [15]. It uses a `buffer` of length `lenbuf` to implement a delay having fractional length `Q/P * lenbuf` (i.e. $I =$ `lenbuf/delay` $=$ `P/Q`).

At every frame, the `buffer` is first **read** with a decimating factor equal to `P/Q` and written to the output file `fid_out`. Then `framelen` values of the input stream, coming from file `fid_in`, are interpolated with factor `P/Q` and **written** into the `buffer`. The `resample` operation, which uses a polyphase implementation of the interpolation filters [16, pages 677–679], takes care of changing the effective length of the buffer to a fractional number, in such a way that the output stream is the `delay`-ed version of the input stream. According to equation (7), using an increment ranging from 2 to 1 we can implement any fractional delay ranging from $B/2$ to B samples, where B is the buffer length. For the purpose of this paper we limit the variability of the delay length to half its nominal size. Nothing prevents to reduce the length even further, but this is not computationally convenient in the implementation presented in section III-B.

The implementation of figure 5 shows the close relationship between sample-rate conversion [16], [2], [3] and delay interpolation. This connection can be useful to understand the fractional delay thoroughly. For instance, we usually

[5]MATLAB is a registered trademark of The MathWorks, Inc., http://www.mathworks.com

[6]See the COST-G6 DAFX home page http://echo.gaps.ssr.upm.es/COSTG6/

adopt $I > 1$ because otherwise we would write a down-sampled (and therefore bandlimited) version of the signal into the buffer, thus making it impossible to recover the high-frequency components in the following read.

In the implementation of figure 5, continuous variations of the effective delay length can not be imposed within a single frame, since the interpolation factor is varied frame by frame. Continuous delay modulations are essential for constructing digital audio effects such as choruses and flangers [17], so that a frame-by-frame MATLAB implementation of these effects should use alternative interpolating functions where the interpolation factor can be prescribed in a local per-sample basis (see e.g. the MATLAB function `interp1`). However, there are many applications of fractional delay lines within signal processing flowgraphs exhibiting feedback-connected modules. In general, these modules can not be computed on a frame-by-frame basis due to circular dependencies spanning time delays different from the frame length. Therefore, we are mainly interested in devising a sample-by-sample implementation of the FAD line and to study its behavior in terms of signal accuracy and computational complexity.

B. Sample-by-Sample Realization

As we have mentioned, the FAD line can be interpreted as an extension of the table-lookup oscillator, where the waveform is written while it is being read. The read operation can be treated in exactly the same way as in the table-lookup oscillator, being possible to apply truncation, polynomial interpolation, or multirate interpolation techniques [2], [3]. More complicated is the injection of a new value, to be done right after the read, in such a way that no "holes" are left in the current pass through the buffer. A fractional increment would correspond to a variable number of writes at each step. For instance, for $I = 1.5$, three writes have to be performed for every couple of reads. Interpolation in write has been proposed by Välimäki et al. in the context of digital waveguide modeling [18], [6], and there called deinterpolation. A deinterpolator can be obtained from a FIR interpolator by transposing its structure and inverting the sequence of coefficients [6, pages 128–134]. These operations lead to a structure which is a sequence of cumulative additions into a delay line. Equivalently, interpolation can be performed on the input stream by using some extra unit delays, thus saving read and write operations in the buffer. This latter implementation is depicted in figure 6 for the case of second-order read and write interpolators, with an increment $I = 1.25$. The dash-dotted lines in the figure should indicate how resampling is performed on the fly on both the read and the write side of the access to the buffer. If further writes have to be performed after the first one in order to fill the blanks, the quantity `d` of figure 6 has to be updated to `d + 1` and the corresponding coefficients of the writing interpolator have to be recomputed.

The FAD line can be expressed by the pseudo-code of figure 7. Actual MATLAB and C functions implementing the FAD line can be found in the software repository of the COST-G6 action on Digital Audio Effects[7].

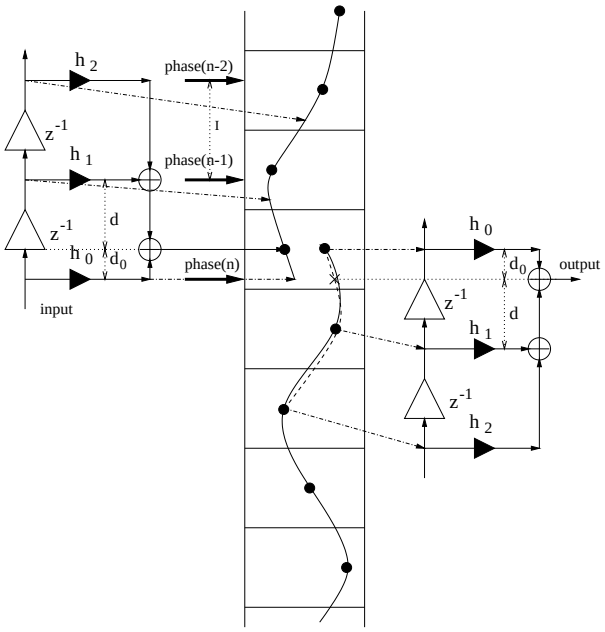


Fig. 6
Interpolated read and write access to a circular buffer

```
loop
     fph = floor(phase);
     output = interpolated_read(table[fph],
              table[fph+1], ...);
     ph = (phase_old + 1) MOD length_table;
     while (ph <= fph) {
       table[ph] = interpolated_write(
         ..., table[phase_old], input);
       ph = (ph + 1) MOD length_table;
     }
     phase_old = fph;
     phase = (phase + Increment);
     if (phase > length_table)
          phase = phase - length_table;
endloop
```

Fig. 7
Pseudo-code for a sample-by-sample realization of the FAD line

Notice that the `interpolated_read` uses samples following the `phase` pointer, while the `interpolated_write` uses samples preceding the pointer.

In the following sections we are going to analyze the performance of a FAD line using quadratic interpolation in both read and write operations.

IV. Input-Output Analysis

[7]The software repository can be reached from the home page of the COST-G6 DAFX action:
http://echo.gaps.ssr.upm.es/COSTG6/

A. Experimental results

The FAD line is a time-varying system, and therefore it is difficult to characterize in terms of frequency response. Figure 8 shows the magnitude spectrum of a delayed $5 - kHz$ sine wave when quadratic interpolation is used in reading and writing. When a sinusoidal input feeds the FAD line, spurious components are added to the main spectral line. The magnitude of these components might be dependent on the frequency of the input sine wave and the initial (fractional) phase of the FAD-line pointer [8]. The signal-to-error noise ratio (SNR) as a function of these two parameters shows a very mild dependence on initial phase. Therefore, it makes sense to plot the average SNR as a function of the input frequency only (figure 9). We can see that low frequencies are affected by high SNR, thus indicating that the FAD line has an acceptable behavior for practical sounds. The noise error has been computed as the sum of the squared differences between the input and output waveform samples[9] [13]. In figure 9 we have considered input sine waves having periods that perfectly divide the delay length, and a time delay that is two thirds of the delay length (increment is 1.5). By applying such SNR analysis to a low-order FIR filter we would obtain a curve lower than that of figure 9 [10]. We do not report that curve here because it can be misleading. In fact, while the error of the FAD line can be interpreted as noise, the error for a FIR line is only given by phase displacement and magnitude attenuation, and no spurious signal components are introduced.

Especially for applications such as waveguide modeling of musical instruments [19], it is important to consider the attenuation that different frequencies are subject to when fed into the delay line. For instance, the decay time of a frequency partial in a waveguide string depends on the attenuation of the interpolated delay line at that frequency. The attenuation of the main peak of the output spectrum turns out to be dependent on the initial phase. In order to have a rough idea of the attenuation property of the quadratically-interpolated FAD line, we plot in figure 10 the minimum, maximum, and mean attenuation as a function of frequency of the input sine wave, for the same set of frequencies as in figure 9, and for a delay such that $I = 1.5$. Again, direct comparison with the FIR line is problematic because the results would be highly dependent on the chosen delay length.

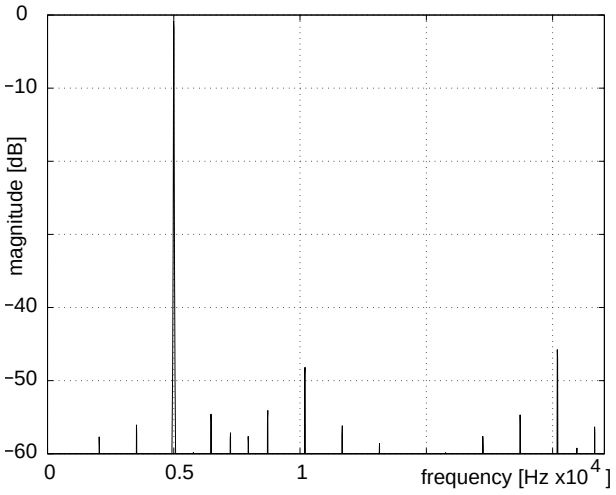


Fig. 8

Magnitude spectrum of the output signal of a FAD line, where the input signal is a sine wave at 5000Hz, the delay is 0.74378s, the sampling rate is 44.1kHz and the buffer is $44100 -$ samples long.

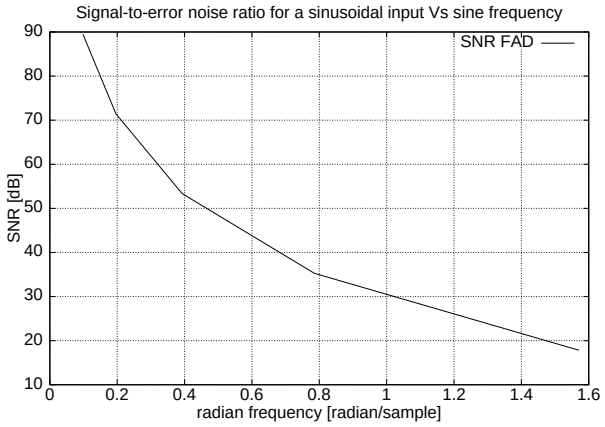


Fig. 9

Experimental signal-to-error noise ratio Vs. sine frequency for the FAD line with quadratic interpolation. The input frequencies are such that the interpolation phase is a multiple of 2π.

B. Signal-to-error noise analysis

In section II we have seen how delay-length modulations add side components to the peaks of a frequency spectrum, as a result of modulations in magnitude and phase responses of the Lagrange interpolator. In the implementation of the FAD line, the magnitude and phase delay of the interpolators are varied sample by sample, as the value of d (see figure 6) is changed at every sample. If we exclude the degenerate cases (such as that obtained with increment $I = 1$) we can assume that, in consecutive accesses to the buffer, the magnitude and the phase delay vary sinusoidally around their mean. Therefore, we can use the approximate analysis of section II, where an input sinewave at frequency ω_0 is subject to phase modulation with a certain modulating frequency ω_M and to amplitude modulation with frequency $2\omega_M$.

The modulating frequency ω_M is related to the fractional

[8] As an example of dependence on initial phase, consider the increment $I = 1$. If the initial phase is 0 the pointer always falls on samples. If the initial phase is 0.5 the pointer always falls between samples. In the two cases, the actual shape of the delayed output is different.

[9] A normalizing factor $\sqrt{2/N}$ has been applied, being N the number of samples per period.

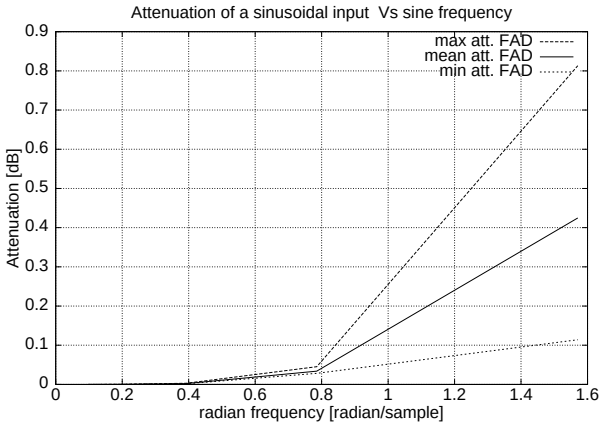


Fig. 10

Attenuation of a sinusoidal input Vs. sine frequency for the quadratically-interpolated FAD line. Sampling rate is set equal to the buffer length and delay is set to $D = 2/3$s

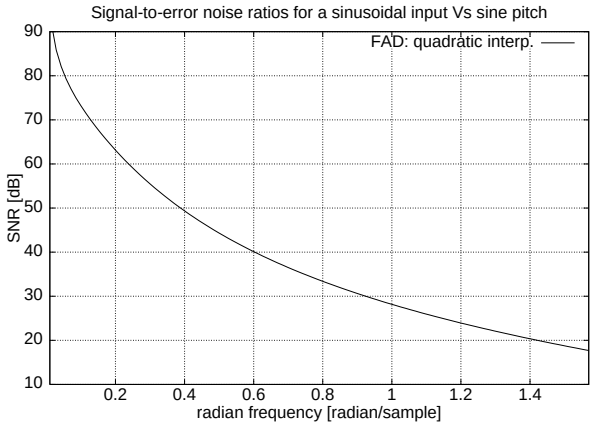


Fig. 11

Signal-to-error noise ratio for the quadratic FAD line

part I_f of the increment I by

$$\omega_M = 2\pi I_f F_s \,. \tag{8}$$

If $I_f \gtrsim 0.3$, the modulation products are folded into the high-frequency region for practical sounds, i.e. for sounds whose energy is mainly concentrated in a bandwidth smaller than $F_s/8$. This means that these modulation products are less audible (especially when the sample rate is larger than $44kHz$), and it is easier to eliminate them by lowpass filtering.

As opposed to the case analyzed in section II, in the FAD line there are two modulations, one acting on input and the other acting on output. However, we can still use the values of table I to compute an estimate of the SNR of the FAD line, in the pessimistic case when the two modulations produced by the read and write interpolators operate constructively. An approximate value of this SNR is

$$SNR = \frac{{A_0}^2}{2\sqrt{2}\sqrt{{A_1}^2 + {A_2}^2 + \ldots}} \,, \tag{9}$$

where the carrier appears at the numerator and the sidebands sum up at the denominator. Figure 11 depicts this signal-to-error noise ratio as a function of the input frequency. The curve gives a tight lower bound to the values of SNR measured in simulations and reported in figure 9, thus justifying the assumptions taken in order to simplify the analysis.

C. Behavior for time-varying delay

The FAD line shows an unconventional behavior when the delay length is dynamically varied. This is illustrated in the following analysis through comparison with the FIR implementation.

Starting from the steady state of a delay line fed with a stationary signal, suppose to vary the delay length as a linear function of time t. Namely, we start at time 0 with the nominal delay τ_0 and decrease it at the rate of k seconds per second:

$$T(t) = \tau_0 - kt \,. \tag{10}$$

The FIR implementation responds with an instantaneous pitch shift in the output signal. In other words, we get a Doppler effect and the pitch shift is

$$\Delta f = 1 + k \,. \tag{11}$$

On the other hand, the FAD line provides a steady pitch shift

$$\Delta f = e^k \tag{12}$$

after a transient time

$$\tau_i = \frac{\tau_0}{k}(1 - e^{-k}) \,. \tag{13}$$

A similar transient is observed when the delay ramp is stopped.

The transient time (13) can be calculated by feeding the delay line with an impulse at time 0. It will come out of the line at time instant τ_i such that

$$\int_0^{\tau_i} I(t)dt = \frac{B}{F_s} \,, \tag{14}$$

where $I(t)$ is the time-dependent increment which produces the desired ramp in delay length. Equation (14) can be rewritten, using (7) and (10), as

$$\int_0^{\tau_i} \frac{1}{\tau_0 - kt} dt = 1 \,, \tag{15}$$

which is solved by (13).

The steady-state transposition (12) can be calculated by observing that a second impulse entering the line at time T_i "sees" an instantaneous delay of $\tau_0 - kT_i$ seconds. It gets out of the line at time $\frac{\tau_0 - kT_i}{k}(1 - e^{-k}) + T_i$, exactly $T_i e^{-k}$ seconds after the impulse which entered at time 0.

It is interesting to notice that the dynamic behavior of the FAD line is similar to the behavior of the analog

CCD delay line [20], which was built in MOS technology as a sampling system where packets of charge were shifted through the channel of a multigate MOSFET transistor[10]. In that case, if B was the number of gates and F_s was the control frequency of a multiphase clock, the delay was still given by (7), provided that I is set to one. In order to vary the delay of the CCD line, F_s had to be made time varying, and it is easy to see that (12) and (13) held for that line as well.

If the delay ramp is applied for 1.11 seconds starting at time 0 with an empty line, the response of the FAD line to a steady sinusoidal input is displayed as a sonogram in figure 12. The transient is clearly visible in the output when the ramp is stopped. This sonogram should be compared to the one obtained with the FIR line (figure 13).

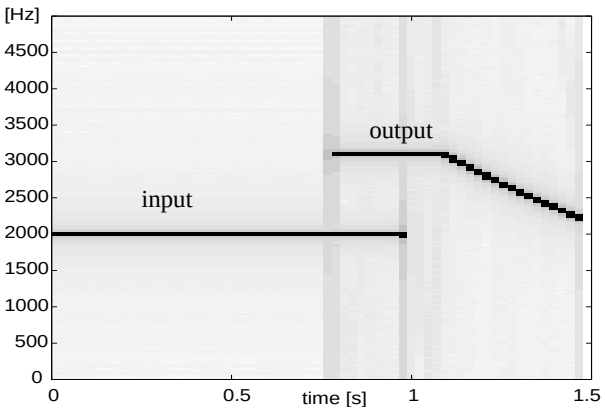


Fig. 12
FAD line: delay ramp from 0.99 s to 0.5 s in 1.11 s; 1 s of sinusoidal input

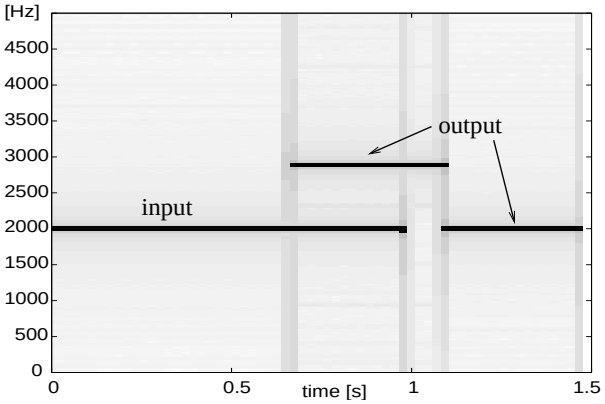


Fig. 13
FIR line: delay ramp from 0.99 s to 0.5 s in 1.11 s; 1 s of sinusoidal input

A different behavior is also reported in response to sinusoidal modulations of the delay length. These modulations are essential for effects such as flanging or phasing. Modulations of the quadratic FAD and FIR lines are reported in figure 14 and 15, respectively. The figures show that the FAD line is less sensitive to artifacts clearly visible (and audible) as faint waves figure 15, which are essentially due to modulations induced by the nonideal response of the interpolator.

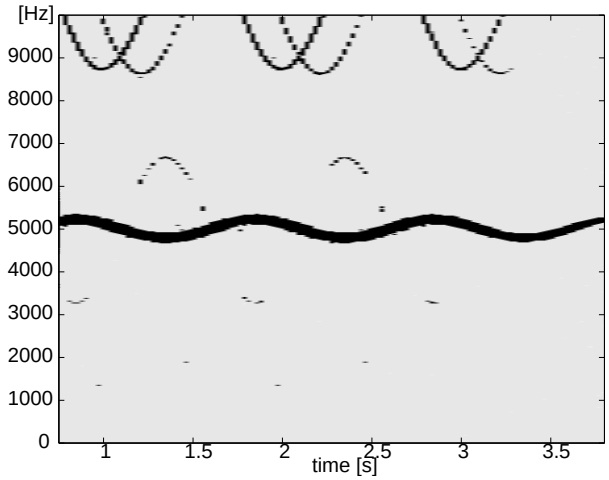


Fig. 14
FAD line with quadratic interpolation: delay-length vibrato

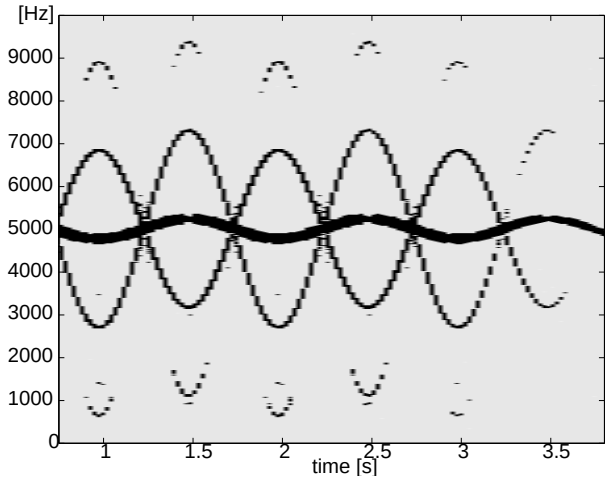


Fig. 15
FIR line with quadratic interpolation: delay-length vibrato

V. Physical Interpretation

If the dynamic behavior of the delay lines is closely analyzed, we see that the FAD and the FIR realizations actually simulate two different physical phenomena. In both cases, the lines can be thought of as a one-dimensional medium where waves propagate. However, when the delay length is dynamically reduced we have two different physical analogies in the two cases. The shortening of the FIR line corresponds to the receiver getting closer to the

[10]Thanks to Giovanni De Poli for pointing out this similarity.

transmitter, and therefore we have a tight simulation of the Doppler effect. On the other hand, the shortening of the FAD line corresponds to increasing the speed of propagation in the medium while maintaining the same physical distance between the two ends.

Figures 16–18 illustrate what happens when using different implementations of the delay lines in the simulation of a one-dimensional waveguide resonator, such as a string. In this application, sketched in fig. 16, there is a couple of delay lines in a feedback connection, each representing propagation of waves in one direction. For the sake of simplicity, the terminations are supposed to be perfectly reflecting. If one of the delays is fed by three periods of a fast sine wave, this packet propagates, gets reflected, feeds the other line, and comes back to the excitation point for another reflection. Under ideal conditions, the packet keeps going back and forth without attenuation or losses. Suppose that, right after a reflection at the left end, suddenly the string gets lengthened by some amount, as we would do for lowering the pitch of the string. This can be simulated, in the classic FIR line implementation, by moving the reading pointer backwards. However, this operation exposes again the wave packet which has just been reflected, thus modifying the "duty cycle" of the waveform, as reported on figure 17. A correct waveform can be obtained by adding a write operation right after the read to the FIR line implementation. This write takes care of erasing the waveform as it passes the reading pointer. Another way of lowering the pitch is that of changing the string tension instantaneously. In a waveguide simulation this corresponds to changing the spatial sampling via a change in the waveguide speed of propagation [19]. This can be achieved by the FAD line implementation just by changing the phase increment, and the result is illustrated in figure 18. A mixture of tension and length increase might be obtained by using the FAD line with a variable buffer size.

A different approach to dynamic tension variations in string models has been recently proposed by Välimäki et al. [11], [21]. They recast the resampling process induced by tension modulation into delay length modulation of a FIR line controlled by the buffer content. This reformulation works under the assumption of a single observation point. The control circuitry is composed of a power estimator (which needs to sum the squares of the content of several delay cells) and a numerical integrator.

Fig. 16

Waveguide model of an ideal string. A sudden pitch lowering is depicted for the FIR implementation of delay lines (dashed line) and for the FAD implementation of delay lines (dash-dotted line). The string before the pitch transition is represented by the dotted line.

The two ways of producing a pitch shift are not equivalent as far as timbre is concerned, as it can be deduced from figures 17 and 18. In fact, the FAD-line pitch change produces a contraction of the whole spectrum, thus modifying the position of formants, while the FIR-line pitch change comes without moving the formants. Moreover, when no waveform erasing is applied out of the reading pointer in the FIR line, the width of the main formant gets narrowed. On the other hand erasing after read preserves both the formant widths and positions.

Summarizing, different implementations of the delay line do have practical consequences on the timbre produced by dynamically-varying waveguide models.

VI. Computational Performance and Application to Digital Audio Effects

The FAD line has only one pointer for accessing data in the buffer. It exhibits spatial locality because any short sequence of accesses spans over a small neighborhood of the pointed buffer cell. On the other hand, a FIR line has two pointers, thus exhibiting two distinct spatial localities. As a consequence, we expect that the FAD line makes better use of the cache in general purpose computer architectures. However, the FAD performs more writes than reads. In order to attain a 50% of delay variability, we have to accept up to two writes for each read. This overhead is partially compensated by the highest efficiency of write operations in modern architectures [22].

These two observations justify the fact that the FAD line, despite of its higher complexity, does not run much slower than the FIR line on a general purpose computer. A benchmark for quadratically interpolated FIR and FAD lines has been performed on an $AMD-K6$ architecture by repeatedly delaying a soundfile stored in an array. The experiment was done using a Linux operating system with the machine in stand-alone single-user configuration. To avoid the effects of context switching due to the underlying operating system, we took the fastest of 14 repetitions, thus obtaining the results summarized in figure 19, where running times are reported for varying buffer size.

A first comment is about the difference in performance between the two algorithms. This is not as big as one might

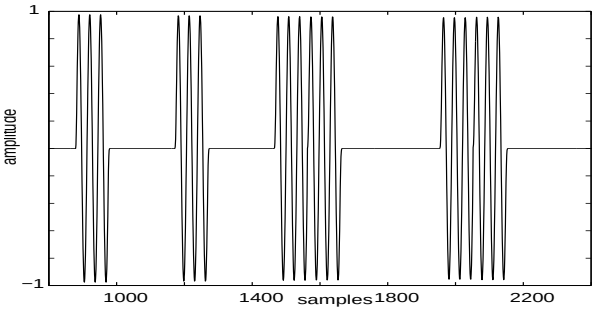


Fig. 17

Note transition waveform: FIR line without erase after read

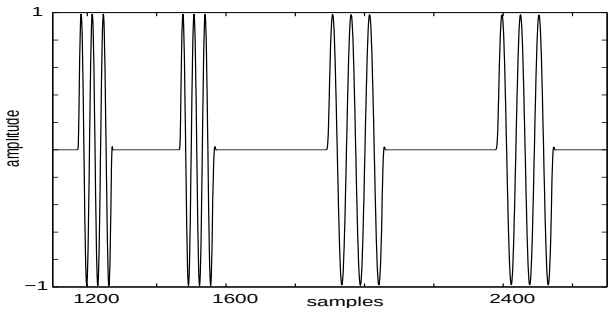


Fig. 18
NOTE TRANSITION WAVEFORM: FAD LINE

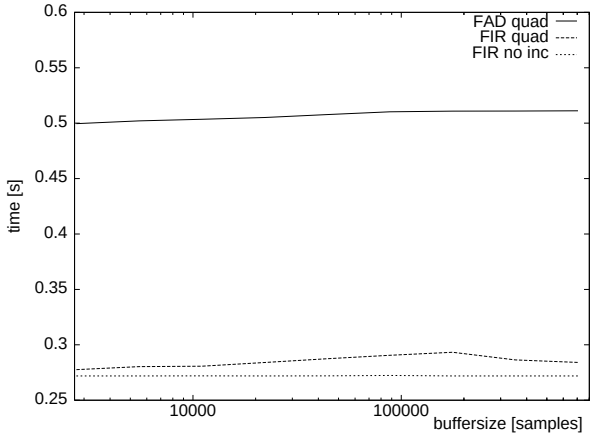


Fig. 19
PERFORMANCE ON AN AMD-K6 OF THE QUADRATICALLY-INTERPOLATED FAD LINE, FIR LINE, AND FIR LINE WITH NO PHASE INCREMENT, AS A FUNCTION OF BUFFER SIZE.

expect, especially if we consider that the quadratically-interpolated FAD line has about three times as many multiplies, twice as many adds, three extra divides, and three times as many tests as the FIR line. A second comment is about the fact that the curves tend to be monotonically increasing. This indicates that more and more cache misses are encountered when using larger buffers. However, the FIR line shows an increase in capacity and conflict misses right after the size of 65536 samples which, when translated in bytes (we use 8-byte double floating point numbers), gives the size of the level-2 cache[11].

Using the FIR line and eliminating the phase increment it is possible to measure the overall cost of caching, wich is visible from figure 19 as the difference between the two lower curves. Since it turns out to be less than 6% even for very large buffers, we can argue that delay lines are not affected much by the memory hierarchy. Similar values of the percentage cost of caching have been measured in other architectures (e.g., the Intel Pentium II), even though the actual shape of the curves is slightly different.

In our implementations, we have not done aggressive code optimization, so that the relative performance of the three realizations might vary in practice from what we have shown. In particular, float-to-integer conversions are expensive, and it would be wise to perform them by direct bit manipulation, as suggested in [23]. Moreover, the memory access patterns show that hardware or software prefetching techniques might be used effectively, especially in the FAD line that has only one locality.

High-quality digital audio effects, such as choruses or flangers, can be easily constructed around delay-modulated FAD lines. The audio quality is good due to good noise rejection for time-varying delay, as shown in sec. IV-C. The fact that the FAD line does not respond instantaneously to variations of the delay length turns out to be useful for achieving natural sounding effects by means of random variations of the delay length. For instance, "random walk" pitch modulations can easily be achieved by low-rate random variations of the delay length. Namely, it is sufficient to call a random function, say, every 100 samples, and to perform piecewise linear interpolation between its values. The resulting output spectrum exhibits partials which are floating around without sudden changes. In the FIR implementation, a similar effect would require either a complicated modulating function to be read at audio rate or high-order interpolation performed on the undersampled control signal [24].

[11] And also of the memory covered by the Translation Lookaside Buffer, which is responsible for fast translation of virtual addresses to physical addresses.

VII. Conclusion

We have proposed a realization of the digital delay line which is based on an extension of the table-lookup oscillator. The proposed realization exploits the features of modern computer architectures and shows good performance in terms of signal-to-error noise ratio, frequency-dependent attenuation and dynamic behavior. We expect this delay line will be considered as a building block for physically-based sound synthesis and for sound effects such as flangers and choruses.

References

[1] Curtis Roads, *The Computer Music Tutorial*, MIT Press, Cambridge, Mass., 1996.

[2] S. J. Orfanidis, *Introduction to Signal Processing*, Prentice Hall, Englewood Cliffs, N.J., 1996.

[3] Udo Zölzer, *Digital Audio Signal Processing*, John Wiley and Sons, Inc., Chichester, England, 1997.

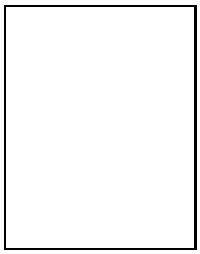
[4] Timo I. Laakso, Vesa Välimäki, Matti Karjalainen, and Unto K. Laine, "Splitting the Unit Delay—Tools for Fractional Delay Filter Design," *IEEE Signal Processing Magazine*, vol. 13, no. 1, pp. 30–60, Jan 1996.

[5] S. Tassart and P. Depalle, "Analytical approximations of fractional delays: Lagrange interpolators and allpass filters," in *Proc. Int. Conf. Acoustics, Speech, and Signal Processing, Munich*, Apr. 1997, pp. 455–458.

[6] Vesa Välimäki, *Discrete-Time Modeling of Acoustic Tubes Using Fractional Delay Filters*, Ph.D. thesis, Elec. Eng. Dept., Acoustics Lab., Helsinki University of Technology, Dec. 1995, Otaniemi 1995 / Report 37.

[7] Vesa Välimäki and Timo I. Laakso, "Suppression of transients in variable recursive digital filters with a novel and efficient cancellation method," *IEEE Trans. Signal Processing*, vol. 46, no. 12, pp. 3408–3414, Dec 1998.

[8] Davide Rocchesso, "Realizzazione di Risuonatori Dispersivi in Tempo Reale," Tesi di laurea, Università di Padova, Dipartimento di Elettronica e Informatica, Feb. 1992.
[9] Davide Rocchesso, "A digital delay line based on fractional addressing," in *Proc. XII Colloquium Mus. Inform.*, Gorizia, Italy, Sept. 1998, AIMI.
[10] Davide Rocchesso, "Fractionally-addressed delay lines," in *Proc. Workshop On Digital Audio Effects (DAFX-98),* Barcelona, Spain, Nov. 1998, pp. 40–43.
[11] Vesa Välimäki, Tero Tolonen, and Matti Karjalainen, "Signal-dependent nonlinearities for physical models using time-varying fractional delay filters," in *Proc. International Computer Music Conference*, Ann Arbor, Michigan, 1998, ICMA, pp. 264–267.
[12] E. Zwicker and H. Fastl, *Psychoacoustics: Facts and Models*, Springer Verlag, Berlin, Germany, 1990.
[13] F. Richard Moore, "Table lookup noise for sinusoidal digital oscillators," *Computer Music J.*, vol. 1, no. 1, pp. 26–29, 1977.
[14] William M. Hartmann, "Digital waveform generation by fractional addressing," *J. Acoustical Soc. of America*, vol. 82, no. 6, pp. 1883–1891, 1987.
[15] Daniel Arfib, "Different ways to write digital audio effects programs," in *Proc. Workshop On Digital Audio Effects (DAFX-98),* Barcelona, Spain, Nov. 1998, pp. 188–191.
[16] Sanjit K. Mitra, *Digital Signal Processing: A computer-Based Approach*, McGraw-Hill, New York, 1998.
[17] P. J. Bloom, "High-Quality Digital Audio in the Entertainment Industry: an Overview of Achievements and Challenges," *IEEE Acoustics, Speech and Signal Processing Magazine*, vol. 2, pp. 2–25, Oct. 1985.
[18] Vesa Välimäki, Matti Karjalainen, and Timo I. Laakso, "Fractional Delay Digital Filters," in *Proc. Int. Symp. Circuits and Systems*, Chicago, Illinois, May 1993, IEEE, pp. 355–358.
[19] Julius O. Smith, *Principles of Digital Waveguide Models of Musical Instruments*, vol. Applications of Digital Signal Processing to Audio and Acoustics, pp. 417–466, Kluwer Academic Publishers, 1998, M. Kahrs and K. Brandenburg, eds.
[20] Johan A. Catrysse, Gilbert Declerck, and Christine C. De Meyer, "CCD delay lines in audio," *J. Audio Eng. Soc.*, vol. 28, no. 11, pp. 800–808, Nov. 1980.
[21] Vesa Välimäki, Tero Tolonen, and Matti Karjalainen, "Plucked-string synthesis algorithms with tension modulation nonlinearity," in *Proc. Int. Conf. Acoustics, Speech, and Signal Processing*, Phoenix, Arizona, March 1999, IEEE, pp. 977–980.
[22] Victor J. Duvanenko, "Two writes make a read," *IEEE Computer*, vol. 31, no. 9, pp. 8, Sept. 1998.
[23] Roger B. Dannenberg and Nick Thompson, "Real-time software synthesis on superscalar architectures," *Computer Music J.*, vol. 21, no. 3, pp. 83–94, 1997.
[24] P. Fernández-Cid and F. J. Casajús-Quiros, "Enhanced quality and variety for chorus/flange units," in *Proc. Workshop On Digital Audio Effects (DAFX-98),* Barcelona, Spain, Nov. 1998, pp. 35–39.



Davide Rocchesso received the *Laurea in Ingegneria Elettronica* degree from the University of Padova in 1992, and the Ph.D. degree from the same university in 1996. His Ph.D. research involved the design of structures and algorithms based on feedback delay networks for sound processing applications. In 1994 and 1995 he was a visiting scholar at the Center for Computer Research in Music and Acoustics (CCRMA) at Stanford University. Since 1991 he has been collaborating with the *Centro di Sonologia Computazionale* (CSC) at the University of Padova as a researcher and a live-electronic designer. Since march 1998 he has been with the *Dipartimento Scientifico e Tecnologico* at the University of Verona, as an Assistant Professor. His main interests are in sound processing, physical modeling, sound reverberation and spatialization, multimedia systems. His home page on the web is http://www.sci.univr.it/~rocchess.