PAN ARCHIVES OF ACOUSTICS
Vol. **39**, No. 4, pp. 591–597 (2014)

DOI: 10.2478/aoa-2014-0063

Driver Filter Design for Software-Implemented Loudspeaker Crossovers

Shu-Nung YAO

School of Electronic, Electrical and Computer Engineering, University of Birminghamg
Edgbaston, Birmingham, B15 2TT, UK; e-mail: SXY043@bham.ac.uk

(*received January 12, 2014; accepted October 23, 2014*)

A hybrid method is presented for the integration of low-, mid-, and high-frequency driver filters in loudspeaker crossovers. The Pascal matrix is exploited to calculate denominators; the locations of minimum values in frequency magnitude responses are associated with the forms of numerators; the maximum values are used to compute gain factors. The forms of the resulting filters are based on the physical meanings of low-pass, band-pass, and high-pass filters, an intuitive idea which is easy to be understood. Moreover, each coefficient is believed to be simply calculated, an advantage which keeps the software-implemented crossover running smoothly even if crossover frequencies are being changed in real time. This characteristic allows users to efficiently adjust the bandwidths of the driver filters by subjective listening tests if objective measurements of loudspeaker parameters are unavailable. Instead of designing separate structures for a low-, mid-, and high-frequency driver filter, by using the proposed techniques we can implement one structure which merges three types of digital filters. Not only does the integration architecture operate with low computational cost, but its size is also compact. Design examples are included to illustrate the effectiveness of the presented methodology.

Keywords: crossover, loudspeaker, pole-zero placement, Pascal matrix, driver filter.

1. Introduction

Most high-end audio systems contain several loudspeakers of various sizes. Each loudspeaker caters to a frequency band within its range. To split frequency components between different loudspeakers, an audio crossover is needed. It divides an audio stream into several frequency bands and then sends the split signals to corresponding loudspeakers. For example, a three-way crossover splits an audio feed into high-, mid-, and low-frequency components separately routed to tweeters, mid-ranges, and woofers. Audio crossovers are typically made of analog networks. Because of the advances in audio digitization (Kulka, 2011) digital signal processors (DSPs) and very-large-scale integration (VLSI) are increasingly popular. With the advent of computer technology, computers are now able to process both audio and video signals in real time. Therefore, software crossovers are implemented in this paper. The driver filter design procedure is also proposed for three-way crossovers.

There have been various ways to design digital filters. One of the most efficient methods is the bilinear transformation, a design procedure which has been simplified by utilizing the Pascal matrix (Psenicka *et al.*, 2002; Konopacki, 2005). As an alternative to the modification of traditional design procedures, the pole-zero placement techniques (Pradabpet *et al.*, 2003; Waterschoot, Moonen, 2007) enable people to design biquad filters in the digital domain. Here, we extract the features of the Pascal matrix (Psenicka *et al.*, 2002) and the pole-zero placement techniques (Pradabpet *et al.*, 2003), adding a magnitude scaling method, to develop a new design procedure. In real time, the resulting filter structure operates more efficiently than that based on Butterworth prototype filters (Yao *et al.*, 2012). As long as the cut-off frequencies of the desired high-pass and low-pass driver filter and the center frequencies of the desired band-pass driver filter are all at the same place, the resulting transfer functions can share the same denominator, therefore can be implemented with more compact size and lower computation cost. While the full pole-zero placement techniques (Pradabpet *et al.*, 2003; Waterschoot, Moonen, 2007) only cope with second-order systems, the hybrid design procedure is compatible with higher-order digital filters. Numerical examples of fourth-order infinite impulse response (IIR) filters are demonstrated in this paper. Also, it presents fewer matrix operations and a more intuitive approach than the full Pascal matrix approaches (Psenicka *et al.*, 2002; Konopacki, 2005).

2. Driver filter design procedure

The general form of an Nth-order analog low-pass filter can be expressed as

$$H(s)=\frac{A_0+A_1s+...+A_Ns^N}{B_0+B_1s+...+B_Ns^N}=\frac{\sum\limits_{n=0}^{N}A_ns^n}{\sum\limits_{r=0}^{N}B_rs^r}, \quad (1)$$

while its corresponding digital low-pass filter is in the following form

$$H(z)=\frac{a_0+a_1z^{-1}+...+a_Nz^{-N}}{b_0+b_1z^{-1}+...+b_Nz^{-N}}=\frac{\sum\limits_{n=0}^{N}a_nz^{-n}}{\sum\limits_{r=0}^{N}b_rz^{-rn}}. \quad (2)$$

In order to use the concept of the bilinear transformation, we have to let

$$s=c\frac{1-z^{-1}}{1+z^{-1}}, \quad (3)$$

where

$$c=\cot\left(\frac{\pi f_c}{f_s}\right) \quad (4)$$

and f_c is the cut-off frequency of the desired digital low-pass filter; f_s, the sampling frequency. As a result, (2) can be rewritten as (5) by combining (1) and (3), and the fact that $H(z)$ is an Nth-order filter, so $n+r$ always equals N.

$$H(z)=\frac{\sum\limits_{n=0}^{N}A_nc^n(1-z^{-1})^n(1+z^{-1})^{N-n}}{\sum\limits_{r=0}^{N}B_rc^r(1-z^{-1})^r(1+z^{-1})^{N-r}}$$

$$=\frac{\sum\limits_{n=0}^{N}A_nc^n\left[\sum\limits_{k=0}^{n}C_k^n(-1)^kz^{-k}\sum\limits_{i=0}^{N-n}C_i^{N-n}z^{-k}\right]}{\sum\limits_{r=0}^{N}B_rc^r\left[\sum\limits_{q=0}^{r}C_q^r(-1)^qz^{-q}\sum\limits_{g=0}^{N-r}C_g^{N-r}z^{-q}\right]}, \quad (5)$$

in which the numbers C_k^n are known as the binomial coefficients denoted as

$$C_k^n=\frac{n!}{(n-k)!k!}. \quad (6)$$

2.1. Determination of the denominator

The computational cost of (5) is high, and therefore, the Pascal matrix (Psenicka *et al.*, 2002) was proposed to compute the coefficients. In order to reduce the computational load further, we only consider the associated part of obtaining the denominator. Through using the Pascal matrix, the denominator can be represented as

$$\mathbf{b}=\mathbf{P}\times\mathbf{B}', \quad (7)$$

where

$$\mathbf{b}=[b_0\ \ b_1\ \ b_2\ \ \ldots\ \ b_N]^{\mathrm{T}}, \quad (8)$$

$$\mathbf{B}'=[B_0\ \ B_1c\ \ B_2c^2\ \ \ldots\ \ B_Nc^N]^{\mathrm{T}}, \quad (9)$$

and the $(N+1)\times(N+1)$ Pascal matrix $\mathbf{P}$ is formed by the following steps:

1. Each element in the first row is equal to 1.
2. The elements $P_{j,N+1}$ in the last column are equal to $(-1)^{j-1}\frac{N!}{(N-j+1)!(j-1)!}$, in which $j=1,2,\ldots,N{+}1$.
3. Finally, we calculate the remaining elements by

$$P_{j,m}=P_{j-1,m}+P_{j-1,m+1}+P_{j,m+1}, \quad (10)$$

where

$$j=2,\ 3,\ 4,\ \ldots,\ N,\ N{+}1$$

and

$$m=N,\ N{-}1,\ N{-}2,\ \ldots,\ 2,\ 1.$$

When it comes to designing digital high-pass filters or band-pass filters with the same cut-off or center frequency, Psenicka *et al.* (2002) and Konopacki (2005), the elements inside the Pascal matrix have to be changed. According to the filter design by Pradabpet *et al.* (2003), it is found that the poles of the band-pass filter can be placed at the same position as those of the low-pass filter if the center frequency of a band-pass filter equals the cut-off frequency of a low-pass filter. The idea is applied to driver filter design in this paper, thereby reducing real-time computational complexity. That is, the Pascal matrix can remain the same and it is unnecessary to do any other matrix operations.

The stability of filters depends on the positions of poles relating to the roots of the denominator. The bilinear transformation only transfers the roots of analog prototype filters on the left-half s-plane to the z-plane, so that the resulting digital filters will be exactly stable and causal (Proakis, Manolakis, 1996).

2.2. Determination of the numerator

The zeros on the z-plane can make the magnitude of $H(z)$ be minimal since they are the roots of numerator. Therefore, their positions provide us the information to decide the numerators of the filters.

For digital low-pass filters, the minimum values of magnitude frequency responses happen at the highest frequency in the Nyquist interval, so we locate zeros at π, i.e., $z=e^{j\omega}|_{\omega=\pi}$, where ω is the frequency in radians and j is defined as $\sqrt{-1}$ in the rest of this paper. Thus, $N_{LP}(z)$, the numerator of the resulting digital low-pass filters, can be expressed as $(1+z^{-1})^N$, which means N zeros should be put at π on the z-plane when designing

Nth-order low-pass filters. On the other hand, in digital high-pass filters, zeros have to be placed at the lowest frequency in the Nyquist interval, i.e., $z = e^{j\omega}|_{\omega=0}$. The zero position makes $N_{HP}(z)$, the numerator of the digital high-pass filters, in the form $(1 - z^{-1})^N$.

As to designing digital band-pass filters, the minimum magnitude simultaneously happens at the highest and lowest frequency, π and 0, so the numerator $N_{BP}(z)$ is written in form as $[(1 + z^{-1})(1 - z^{-1})]^{N/2}$.

2.3. Determination of the gain

The transfer function of digital filters now is divided into two parts as shown in the following formula

$$H(z) = \frac{\sum\limits_{n=0}^{N} a_n z^{-n}}{\sum\limits_{r=0}^{N} b_r z^{-r}} = GH'(z), \tag{11}$$

where

$$H'(z) = \frac{N(z)}{D(z)}. \tag{12}$$

$H'(z)$ has already been obtained, and thereby in this subsection we apply a magnitude scaling method (Oppenheim, Schafer, 1999; Proakis, Manolakis, 1996) to get the gain factor, G.

In (11), $H'(z)$ is the transfer function of the desired digital filter without normalization, so we merely have to calculate the maximum value of $|H'(z)|$, and then G is equal to its inverse which can normalize the maximum value of the magnitude frequency response to be 0 dB.

In the cases of digital low-, high-, and band-pass filters, we expect the maximum values happening at 0, π, and ω_c respectively, so

$$G_{LP} = \frac{1}{|H'_{LP}(e^{j\omega}|_{\omega=0})|}, \tag{13}$$

$$G_{HP} = \frac{1}{|H'_{HP}(e^{j\omega}|_{\omega=\pi})|}, \tag{14}$$

and

$$G_{BP} = \frac{1}{|H'_{BP}(e^{j\omega}|_{\omega=\omega_c})|}. \tag{15}$$

From the above two subsections, the locations of minimum values in frequency magnitude responses are used to decide numerators and the maximum magnitude values are used to compute gain factors. It is clear that Chebyshev or Elliptic filters having equiripple passband or stopband will make the locations of minimum and maximum magnitude ambiguous. Therefore, provided that the magnitude frequency response of an IIR filter is monotonic in the passband and the stopband, like that of a Butterworth filter, the maximum and minimum points are always at the special frequencies such as 0, π, or ω_c.

3. Examples

The following examples will show how to design low-, high-, and mid-frequency driver filters in a three-way audio crossover based on the proposed algorithm.

In the first example, we design a fourth-order low-frequency driver filter with 3-dB pass-band corner frequency 1 kHz and sampling frequency 48 kHz. To begin with, the Butterworth prototype filter is

$$H(s) = \frac{1}{s^4 + 2.613s^3 + 3.414s^2 + 2.613s + 1} \tag{16}$$

and the parameter c can be computed by (4), i.e.,

$$c = \cot\left(\frac{1000\pi}{48000}\right) = 15.2571. \tag{17}$$

Together with (7), the coefficients can be obtained by the matrix operation

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 4 & 2 & 0 & -2 & -4 \\ 6 & 0 & -2 & 0 & 6 \\ 4 & -2 & 0 & 2 & -4 \\ 1 & -1 & 1 & -1 & 1 \end{bmatrix} \begin{bmatrix} 1.0000 \\ 39.866 \\ 794.71 \\ 9280.2 \\ 54186. \end{bmatrix}. \tag{18}$$

Thus, the denominator is represented as

$$\begin{aligned} D(z) = {} & 64302 - 235221z^{-1} + 323533z^{-2} \\ & - 198260z^{-3} + 45662z^{-4}. \end{aligned} \tag{19}$$

Then, by using the zero placement, the numerator $N_{LP}(z)$ is $(1 + z^{-1})^4$. Finally, the gain factor G_{LP} is computed by the magnitude scaling method, thereby equaling 1. The responses can be found in Fig. 1.

The second example illustrates the procedure of designing the corresponding high-frequency driver filter with the same cut-off and sampling frequency. First, the denominator remains the same as shown in (19). Second, the zeros are located at 0, which makes the numerator $N_{HP}(z)$ be $(1 - z^{-1})^4$. Finally, $\left|D(e^{j\omega}|_{\omega=\pi})\right|$ divided by $\left|N_{HP}(e^{j\omega}|_{\omega=\pi})\right|$ equals G_{HP}, i.e., 54185. The responses are shown in Fig. 2.

In the final example, a mid-frequency driver filter is designed, operating at a rate of 48 kHz and having 1 kHz center frequency. In the first place, the denominator is the same as (19). Then, the center frequency in radian units can be presented as

$$\omega_c = \frac{2\pi \times 1000}{48000} = 0.1309. \tag{20}$$

Through the concept of the zero placement, the numerator of mid-frequency driver filters is $[(1 + z^{-1})(1 - z^{-1})]^2$. To determine the gain factors, G_{BP}, which equals 329.1476, can be calculated by (15). The responses are presented in Fig. 3.

Group delay will absolutely be introduced into the audio signal when using IIR filters. By looking into

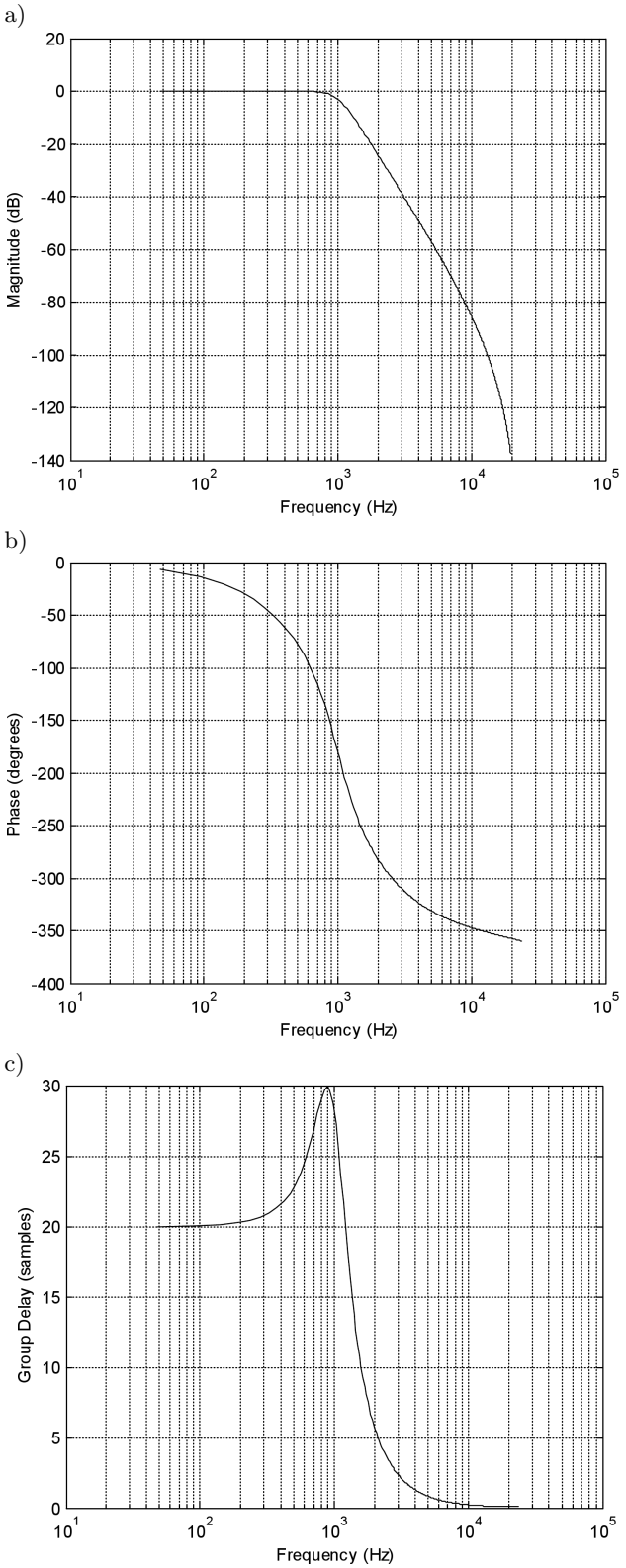

Fig. 1. The a) magnitude, b) phase frequency response, and c) group delay of the low-frequency driver filter.

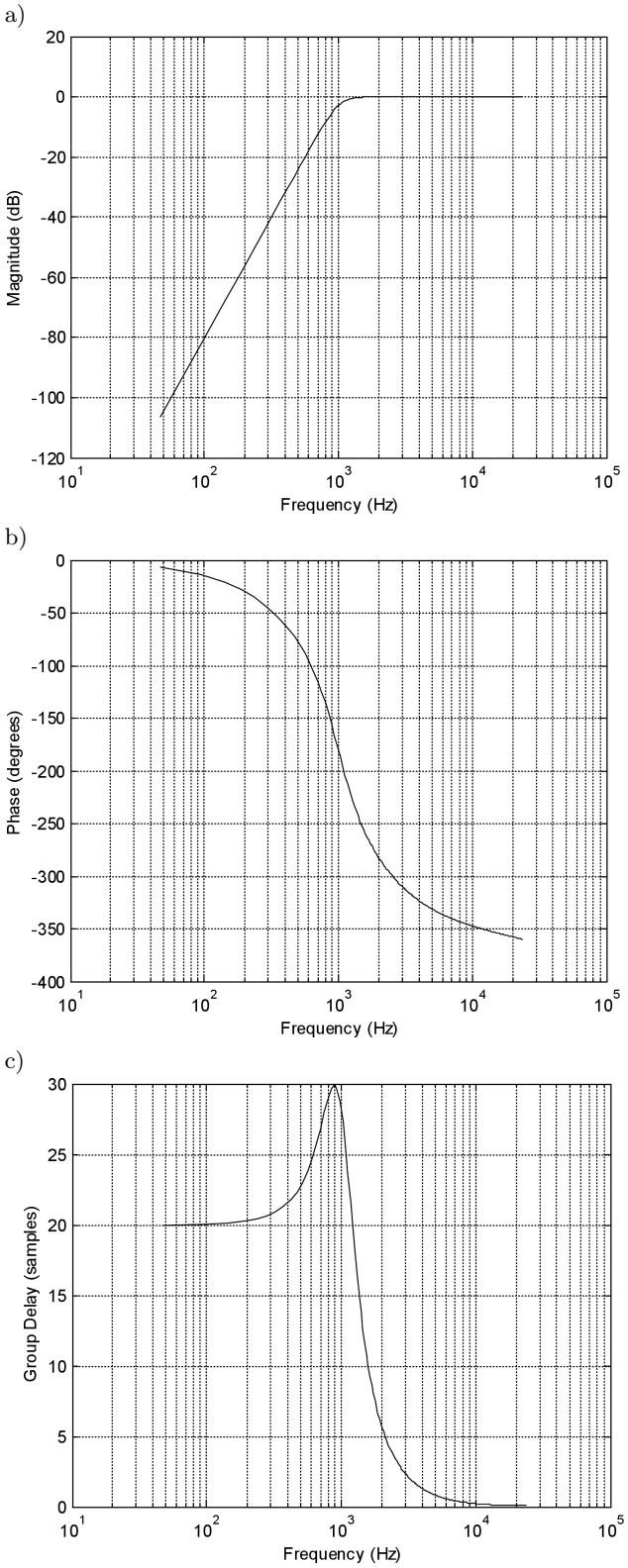

Fig. 2. The a) magnitude, b) phase frequency response, and c) group delay of the high-frequency driver filter.

the frequency responses in Fig. 1, Fig. 2, and Fig. 3, interestingly, we notice the similar group delay curves in Fig. 1c, Fig. 2c, and Fig. 3c.

The resulting fourth-order digital filters with the same denominator can be compactly implemented together. By considering the mid-frequency driver fil-

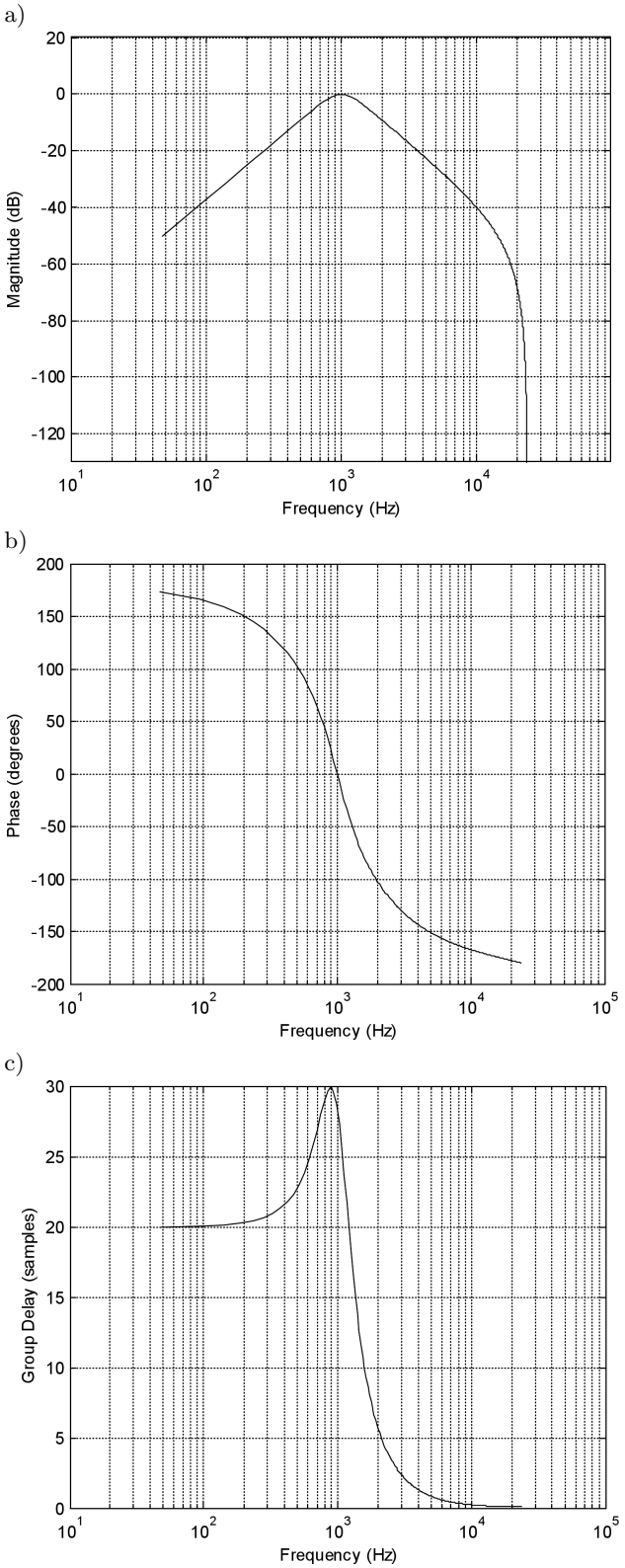

Fig. 3. The a) magnitude, b) phase frequency response, and c) group delay of the mid-frequency driver filter.

ter as the basic building block, the low- and high-frequency driver filter can be rewritten as following transfer functions:

$$H_{LP}(z) = G_{BP}\frac{N_{BP}(z)}{D(z)}\frac{G_{LP}}{G_{BP}}\frac{(1+z^{-1})^2}{(1-z^{-1})^2}, \tag{21}$$

$$H_{HP}(z) = G_{BP}\frac{N_{BP}(z)}{D(z)}\frac{G_{HP}}{G_{BP}}\frac{(1-z^{-1})^2}{(1+z^{-1})^2}. \tag{22}$$

Figure 4 shows the example of transposition structure (Oppenheim, Schafer, 1999) as well as the cascade and parallel implementation of (21) and (22). The integration architecture simultaneously reproduces three kinds of frequency responses with lower computational complexity. For hardware implementations, given low-pass, high-pass, and band-pass driver filters with the same cut-off or center frequencies can be merged into one realization structure. However, according to the filter coefficients as shown in Fig. 4, the high dynamic range of filter coefficients may make the hardware architecture sensitive to electronic noise. In order to keep from the sensitivity, the proposed system is implemented in software.

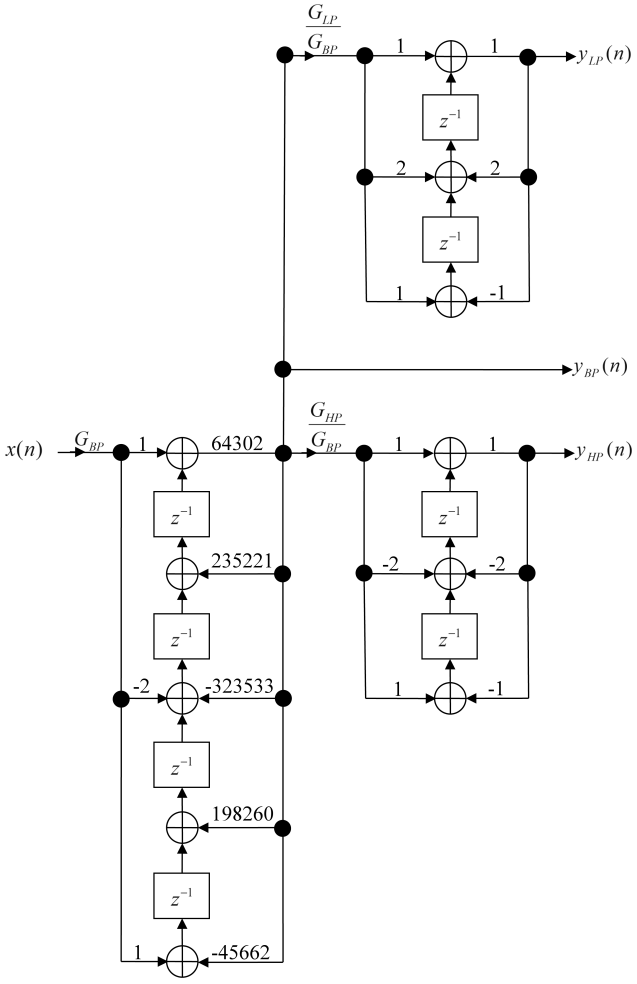

Fig. 4. The block diagram of the resulting filters.

4. Software realization of loudspeaker crossovers

The software crossovers are designed in VST specification. A stereophonic sound file needs two VST

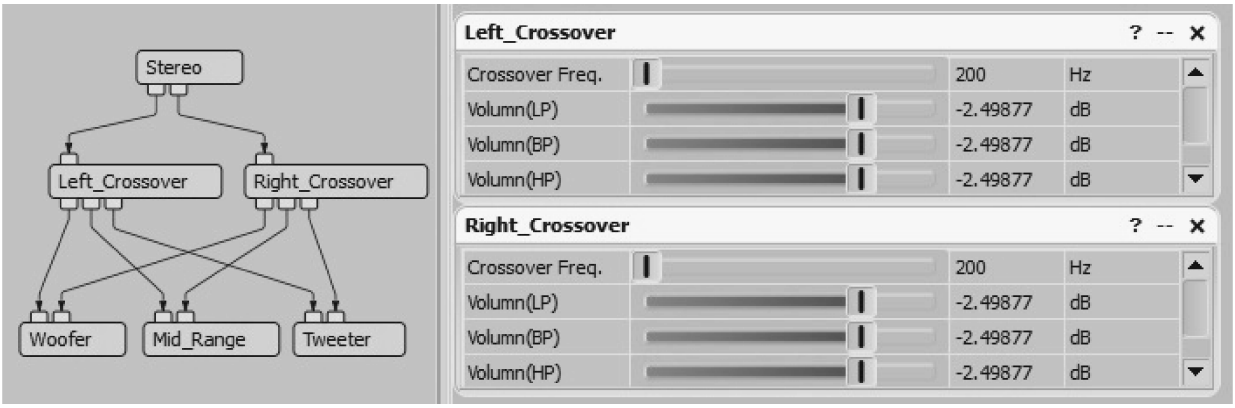

Fig. 5. The screen shot of the proposed VST plug-in setting for a stereophonic sound source.

plug-ins. The one is for a left channel; the other for a right channel. Figure 5 presents the screen grab of the stereo system and Fig. 6 shows the corresponding diagram of a crossover plug-in. Users can make adjustments to the crossover frequency as well as volume controls by moving sliders. The volume can be adjusted effectively and immediately. The crossover frequency adjustment, however, will take effect from next audio frame in order to keep listeners from hearing audible clicks in real time. The plug-ins are compatible with two-way loudspeakers, when users choose to float the middle outputs. The proposed software crossover is available by downloading from our website (http://postgrad.eee.bham.ac.uk/yaos/).

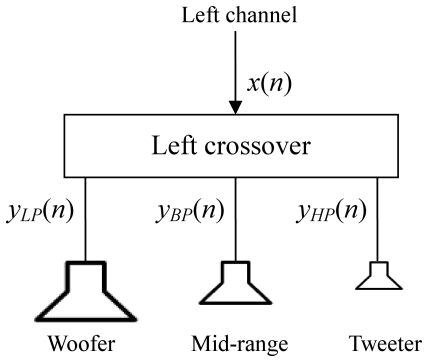

Fig. 6. A single loudspeaker crossover for the left audio channel.

5. Comparisons

Generally speaking, each sample of the signal needs processing $2(N+1)$ multiplications in an Nth-order IIR filter, so it needs $3 \times 2(N+1)$ multiplications for passing by an Nth-order high-pass, low-pass, and band-pass IIR filter. By using the proposed integration technique, the additional transfer functions, like the last terms of (21) and (22), designed for high-pass and low-pass driver filters are $\frac{N}{2}$th-order, and therefore the total number of multiplications should be about

$$2(N+1) + 2 \times 2\left(\frac{N}{2} + 1\right).$$

To be more precise, the odd-order coefficients of the numerator in the proposed band-pass driver filter are always 0. Plus, the proposed architecture needs additional three gains used for magnitude normalization. As a result, the total number of multiplications actually equals

$$2(N+1) + 2 \times 2\left(\frac{N}{2} + 1\right) - \frac{N}{2} + 4.$$

Table 1 specifically shows the computational costs from fourth-order to tenth-order systems. Through looking into the percentages of reduced computational complexity in Table 1, we notice that the higher the order is, the more efficient the algorithm is.

Table 1. Computational complexities for the proposed system and a general system.

System order	Number of multiplications of a general system	Number of multiplications of the proposed system	Computational burden reduced
4th-order	30	24	20%
6th-order	42	31	26.19%
8th-order	54	38	29.63%
10th-order	66	45	31.82%

6. Conclusion

In this paper, we combine the characteristics of the pole-zero placement and the Pascal matrix with the magnitude scaling method, to produce a simple procedure for designing and integrating driver filters for three- or two- way loudspeaker crossovers. On the one hand, full pole-zero placement techniques are intuitive, but for high-order digital filters, the procedure would be complicated; on the other hand, the bilinear transform by the Pascal matrix allows the design for any filter order. However, when considering different kinds of filters, the matrix needs modification, a change which increases the number of matrix operations. Therefore, the proposed procedure not only includes the advantages of these methods but also overcomes the problems they have raised. The software implementation of audio crossovers with an adjustable crossover frequency in real time is presented.

Further objective and subjective measurements are planned for future work. The further experimental results of using real loudspeakers may reveal additional engineering applications of the proposed algorithm. The potential for the compensation of nonlinear or time-variant distortions in loudspeaker will be of great interest.

References

1. Konopacki J. (2005), *The frequency transformation by matrix operation and its application in IIR filters design*, IEEE Signal Processing Letters, **12**, 1, 5–8.
2. Kulka Z. (2011), *Advances in digitization of microphones and loudspeakers*, Archives of Acoustics, **36**, 2, 419–436.
3. Oppenheim A.V., Schafer R.W. (1999), *Discrete-time signal processing*, Prentice Hall, New Jersey.
4. Pradabpet C., Yimman S., Hinjit W., Chivapreecha S., Dejhan K. (2003), *Design and implementation of biquad digital filter*, Proceedings of the IEEE Asia-Pacific Conference on Communications, 1138–1142.
5. Proakis J.G., Manolakis D.G. (1996), *Digital Signal Processing*, Prentice Hall, New Jersey.
6. Psenicka B., Garcia-Ugalde F., Herrera-Camacho A. (2002), *The bilinear Z transform by Pascal matrix and its application in the design of digital filters*, IEEE Signal Processing Letters, **9**, 11, 368–370.
7. Waterschoot V.T., Moonen M. (2007), *Pole-zero placement technique for designing second-order IIR parametric equalizer filters*, IEEE Trans. Audio Speech Lang. Process., **15**, 8, 2561–2565.
8. Yao S.N., Collins T., Jančovič P. (2012), *Hybrid method for designing digital Butterworth filters*, Computers & Electrical Engineering, **38**, 4, 811–818.