Hybrid Filtering: Frequency Domain Approximate Filtering and Time Domain Compensation

Yoshiki Kutsuwa*, Nobutaka Ukeji*, Koichi Ichige* and Rokuya Ishii*
* Department of Electrical and Computer Engineering, Yokohama National University
Yokohama 240-8501, Japan. E-mail: koichi@ynu.ac.jp

***Abstract*—This paper presents an efficient FIR filtering method using frequency domain approximate processing and time domain compensation. Most of basic operations in signal/image processing are based on FIR filtering, i.e., discrete convolution. Fast Fourier Transform (FFT) can be used to implement discrete convolution in frequency domain in order to reduce computational complexity, for certain signal and impulse response durations. However FFT often needs zeros to be added to signals, since it is designed for the signals whose duration is power of two. Therefore the operation sometimes contains many redundant calculations. This paper presents a computationally efficient filtering method using modified frequency domain processing and compensation in time domain which can improve efficiency in convolution operation. Its effectiveness is verified through some computer simulation and DSP implementation.**

I. Introduction

Finite Impulse Response (FIR) filter is one of the very fundamental operators in digital signal processing [1]. It is well-known that the output of the filter is given by the discrete convolution of the input signal and the impulse response, and it can also be derived by cyclic convolution in frequency domain via Discrete Fourier Transform (DFT) and Inverse DFT. The cyclic convolution can be more efficiently computed if we use Fast Fourier Transform (FFT) instead of DFT, when the input signal and impulse response durations are given as 2^n where n is some natural number. Note that zeros are added if the durations are not equal to 2^n, those zeros are of course redundant but still more efficient than DFT operation.

Overlap-Save method [2] is a way to avoid such redundant zero addition. This method divides the input signal into the blocks of a finite duration, while each block is overlapped for some samples with neighboring blocks, and calculates the filter output using FFT. The cost for this method is often superior to that for cyclic convolution and direct discrete convolution, however it still contains redundant computations to be improved.

Note that the input signal is often divided into the blocks whose duration is 2^n in signal processing applications, and the filter output is often slightly longer than 2^n due to the impulse response duration. We have to add many redundant zeros for such case, almost a half at most, since FFT size should be 2^{n+1} not to make any error in frequency domain processing.

This paper presents a computationally efficient filtering method using modified frequency domain processing and compensation in time domain which can improve efficiency in convolution operation. The proposed approach is called Hybrid Filtering since it employs the operations both in frequency domain (approximate filtering with error) and time domain (compensation). We employ approximate cyclic convolution whose FFT size is 2^n, instead of exact computation with the FFT size 2^{n+1}. After that we compensate the error arising in 2^n FFT operation. The effectiveness of the proposed method is verified through some computer simulation and DSP implementation, and we specify the range of the number of inputs and the filter length where the proposed method becomes superior than the conventional methods.

II. Preliminaries

A. Convolution Operation

Consider a general discrete convolution problem in time domain. The discrete convolution of the input $f(n)$ and the impulse response of FIR filter $h(n)$ is given by

$$g(n) = f * h(n) = \sum_{\ell=-\infty}^{\infty} f(\ell)h(n-\ell) \tag{1}$$

The discrete convolution can also be computed by the multiplication in frequency domain via DFT/IDFT operation. The output g in (1) is also obtained by so-called cyclic convolution, i.e.,

$$g = \text{IDFT}[FH]. \tag{2}$$

where

$$F = \text{DFT}[f], \tag{3}$$
$$H = \text{DFT}[h]. \tag{4}$$

Note that the duration N of the output g becomes $N = K + L - 1$, where the duration of the input f and that of the impulse response h are K and L, respectively. All the DFT/IDFT operations in (1)-(3) are processed as N-point DFT/IDFT. In case $\log_2 K$ is an integer, i.e., K is represented by 2^k with some integer k, FFT/IFFT can replace DFT/IDFT operations. Generally zeros are added in order that the duration of the target sequence becomes 2^k with some integer k. This approach is summarized as the following 3 steps.

B. Overlap-Save method [2]

Overlap-Save method is briefly reviewed. This method divide input signal into the blocks of a finite duration, and calculate the filter output using FFT.

First the input $f(k)$ is divided into the blocks of which duration is $N = 2^n$, each block is overlapped for $(L-1)$ samples with neighboring blocks. Here N denotes the size of FFT, and the value of N should be about four times of L [2]. Then the filter output is calculated by the following 3 steps.

Step1

Cyclic convolution is applied to each block. Zeros are added to the impulse response h so that its duration becomes N, in advance of cyclic convolution. Let the obtained sequence from r-th block g_{rp}.

Step2

The first $(L-1)$ samples are ignored due to the edge effect of cyclic convolution. Finally the filter output $g(n)$ is obtained as follows [2].

$$g(n) = \sum_{r=0}^{\infty} g_r(n - r(K - L + 1) + L - 1) \tag{5}$$

where

$$g_r(n) = \begin{cases} g_{rp}[n], & (n = L-1, L, \ldots, K-1) \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

III. Proposed Method

The proposed hybrid filtering method is developed in this section. The proposed method first employs short-FFT/IFFT (number of FFT/IFFT samples is shorter than output) and therefore the result has some error. After this intentional operation, the error is compensated by time-domain processing.

A. Cyclic Convolution and Its Characteristics

Consider the system whose impulse response is given by h (duration L). For the input f (duration K), the output g (duration $N = K+L-1$) is given by the discrete convolution of f and h, i.e.,

$$g(m) = \frac{1}{N} \sum_{k=0}^{N-1} \sum_{\ell=0}^{N-1} f(k)h(\ell) \sum_{n=0}^{N-1} W_N^{n(k+\ell-m)} \tag{7}$$

where $W_N = e^{-j\frac{2\pi}{N}}$ is a twiddle factor. Here the sum of twiddle factors becomes

$$\sum_{n=0}^{N-1} W_N^{n(k+\ell-m)} = \begin{cases} N, & k+\ell-m = 0, N \\ 0, & k+\ell-m \neq 0, N \end{cases} \tag{8}$$

then (7) can be decomposed as

$$g(m) = g_1(m) + g_2(m), \tag{9}$$

$$g_1(m) = \sum_{k=0}^{K-1} f(k)h(m-k) \tag{10}$$

$$g_2(m) = \sum_{k=0}^{K-1} f(k)h(N+m-k) \tag{11}$$

using two terms $g_1(m)$ and $g_2(m)$. Considering the domain where $g_1(m)$ and $g_2(m)$ have non-zero values, they can be reduced to

$$g_1(m) = \begin{cases} \displaystyle\sum_{k=0}^{m} f(k)h(m-k), & (m=0,1,\ldots,L-1) \\ \displaystyle\sum_{k=m-L+1}^{m} f(k)h(m-k), & (m=L,L+1,\ldots,K-1) \\ \displaystyle\sum_{k=m-L+1}^{K-1} f(k)h(m-k), & (m=K,K+1,\ldots,K+L-1) \end{cases} \tag{12}$$

$$g_2(m) = \sum_{k=N+m-L+1}^{K-1} f(k)h(N+m-k), \quad (m = 0,1,\ldots,K+L-2-N) \tag{13}$$

Here $K+L-1-N$ becomes greater than or equal to zero when we choose the appropriate value of the FFT size N, therefore we have $g_2(m) = 0$ and thus the result of convolution depends only on $g_1(m)$. The result $g''(m)$ when using the appropriate FFT size can be written as

$$g''(m) = g_1''(m) + g_2''(m) = \begin{cases} \displaystyle\sum_{k=0}^{m} f(k)h(m-k), & (m=0,1,\ldots,L-1) \\ \displaystyle\sum_{k=m-L+1}^{m} f(k)h(m-k), & (m=L,L+1,\ldots,K-1) \\ \displaystyle\sum_{k=m-L+1}^{K-1} f(k)h(m-k), & (m=K,K+1,\ldots,K+L-2) \end{cases} \tag{14}$$

Note that $K+L-1$ is larger than N, hence $g_1(m)$ is truncated by $0 \le m < N$, it leads the fact that $g_2(m)$ becomes non-zero. Substituting $m = p - N$ to (13), we have

$$g_2(m) = \sum_{k=p-L+1}^{K-1} f(k)h(p-k), \quad (p = N, N+1, \ldots, K+L-2) \tag{15}$$

This is equivalent to (14) for $N \le m < K+L-1$. Therefore it holds that the sequence $g''(m), (N \le m < K+L-1)$ can be derived from $g(m), (0 \le m < K+L-N)$.

B. Proposed Computation Procedure

The computation procedure of the proposed method is described. Let denote the input f (duration $K = 2^n$, zero is added if shorter than 2^n), the impulse response h (duration L), and FFT size $N = K$. The result of cyclic convolution with FFT size $N = K$ is given by $g(m)$ which contains error, the result of time-domain convolution for compensation is $g'(m)$, and the final result is given by $g''(m)$ Here the time-domain convolution is given as

$$g'(m) = \sum_{k=0}^{K-1} f(k)h(m-k), \quad (m = 0, 1, ..., K+L-2) \tag{16}$$

Step1

First, the input f and the impulse response h are FFT-transformed with FFT size $N = K$.

$$F(k) = \text{FFT}[f(n)] \tag{17}$$

$$H(k) = \text{FFT}[h(n)] \tag{18}$$

Then $g(m)$ is given by a cyclic convolution:

$$g(m) = \text{IFFT}[F(k)H(k)] \tag{19}$$

Here the duration of the output g becomes K, that is shorter than the expected duration $(K + L - 1)$. That means the first $(L - 1)$ samples contain error to be compensated. Therefore the true (without error) sequences is restricted to

$$\begin{aligned} g''(m) &= g(m), \\ &(m = K + L - 1 - N, K + L - N, \ldots, N - 1) \end{aligned} \tag{20}$$

Step2

The first and last $L/2$ samples are directly computed. as follows.

$$g''(m) = \begin{cases} g'(m), & (m = 0, 1, \ldots, K + \frac{L}{2} - N - 1) \\ g'(m), & (m = K + \frac{L}{2}, K + \frac{L}{2} + 1, \ldots, K + L - 2) \end{cases} \tag{21}$$

Step3

By using the results of Step 1 and 2, the still-remaining error is compensated.

$$g''(m) = \begin{cases} g(m) - g'(m+N), & (m = \frac{L}{2}, \frac{L}{2} + 1, \ldots, L - 2) \\ g(m-N) - g'(m-N), & (m = K, K + 1, \ldots, K + \frac{L}{2}) \end{cases} \tag{22}$$

Then the whole g'' can be obtained from (16), (20) and (22). This procedure is also depicted as Fig. 1.

IV. SIMULATION

Figure 2 shows the number of additions and that of multiplications for the case of the number of inputs K=512 and various filter length L. We see from Fig. 2 that the proposed method is effective for some values of L in this case. More detail will be discussed via DSP implementation in the next section.

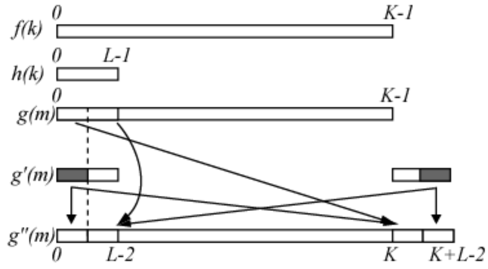
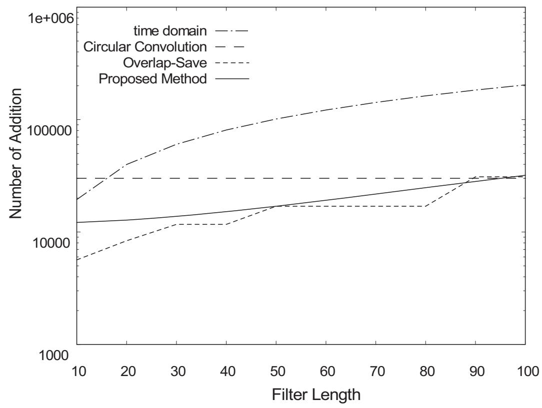
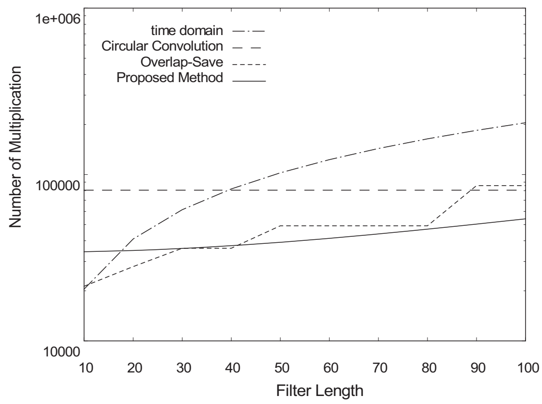


Fig. 1. Overview of the proposed method.



(a) number of additions



(b) number of multiplications

Fig. 2. Comparison of the number of additions/multiplications

V. EXPERIMENTS

The proposed algorithm is also evaluated through DSP implementation on the board TMS320C6713 by Texas Instruments. Its major specifications are; 32-bit floating-point processing DSP, clock 225MHz, and 2 Multiply and ACcumuration (MAC) operators.

The proposed method is implemented on the above DSP and its processing speed (CPU cycles) is evaluated for each process. Table I shows the processing speed of the proposed and that of the conventional cyclic convolution methods when the number of inputs K=256 and the filter length L = 33. Table I shows that the proposed method will be effective for

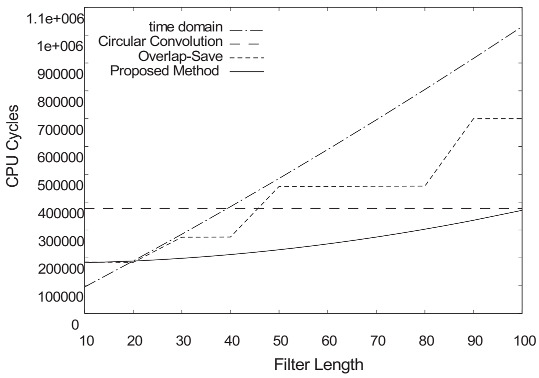
TABLE I
COMPARISON OF CPU CYCLES (NUMBER OF INPUTS $K = 256$, FILTER LENGTH $L = 33$, SIZE OF FFT $N = K$)

Cyclic convolution		Proposed method	
FFT (size $2N$)	260,020	FFT (size N)	115,734
IFFT (size $2N$)	93,692	IFFT (size N)	41,824
		short filtering	20,876
		compensation	12,036
whole processing	398,388	whole processing	202,056

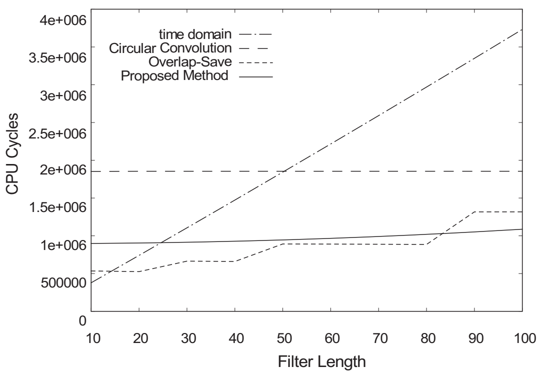
this case, i.e., L is smaller than K.

Also the DSP processing speed is evaluated for the proposed method, direct computation (general convolution in time domain), cyclic convolution and Overlap-Save method [2] when the number of inputs K is 256 and 1024. Figure 3 shows the results for the case of (a) $K = 256$ and (b) $K = 1024$. We see that the proposed method is superior for some range of K and L, and the overall behavior is almost the same with the result of computer simulation discussed in the previous section.

We tested those methods for various values of K and L, and evaluates which methods is the most effective for each case. The result is shown in Fig. 4, we found that the proposed method is not effective for any value of K and L, but there exists a certain range of (K, L) where the proposed method is superior than the others. We could specify the range as in Fig. 4.



(a) in the case K=256



(b) in the case K=1024

Fig. 3. CPU cycles for various filter length L and the number of inputs K

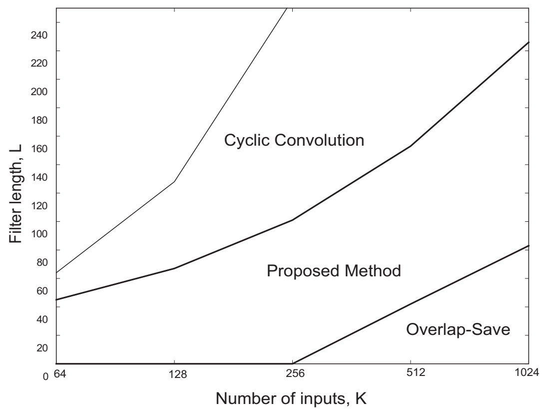


Fig. 4. Effective regions for each filtering methods

VI. CONCLUDING REMARKS

This paper presented a computationally efficient FIR filtering method using modified frequency domain processing and compensation in time domain. Performance of the proposed approach was evaluated through some computer simulation and DSP implementation, and we clarified that the proposed approach is effective to a certain range of filter length and the number of inputs. Also we specified the range where the proposed method is superior than the conventional approaches.

ACKNOWLEDGMENT

The authors would like to thank Professor Hitoshi Kiya at Tokyo Metropolitan University for his valuable comments.

REFERENCES

[1] A. J. Oppenheim, R. Schafer, Discrete-Time Signal Processing, Prentice-Hall, 1989.
[2] M. Borgerding, "Turning Overlap-Save into a Multiband Mixing, Downsampling Filter Bank", IEEE Signal Processing Magazine, vol. 23, no. 2, pp. 158-161, Mar. 2006.