고속 알고리즘을 이용한 음장 효과 구현

손성용(ICU), 서정일(ETRI), 한민수(ICU)

<차 례>

<Abstract>

Sound Field Effect Implementation Using Fast Algorithm

Sung Young Son, Joung Il Seo, Minsoo Hahn

It is difficult to implement sound field effect on real time using linear convolution in time domain because linear convolution needs much multiply operations. In this paper three ways is introduced to reduce multiplication operations. Firstly, linear convolution in time domain is replaced with circular convolution in frequency domain. It means that it operates multiplication in place of convolution. Secondly, one frame will be divided into several frames. It will reduce the multiplication operation in processing that transforms time domain into frequency domain. Finally, QFT will be used in place of FFT. Three ways result much reduction in multiplication operations. The reduction of the multiplication operation makes the real time implementation possible.

* Keywords: linear convolution, circular convolution, QFT (Quick Fourier Transform)

1.서 론

1.1. 음장 효과

음장 효과는 "무향실에서 녹음된 음으로부터 콘서트홀이나 동굴에서 발생하는 울림을 음에 부여하는 것"으로 정의한다. 음장 효과에 부여할 울림음은 울림이 존재하는 공간에서 임의의 임펄스성 신호를 발생시켜 얻을 수 있다. 이러한 울림음을 룸임펄스라고 한다[1].

룸임펄스는 크게 두 부분으로 모델링이 가능하다. 룸임펄스의 시작 부분인 초기 반사(early reflection), 크기가 상당히 떨어지는 부분인 추후 반사(reverberation)로 모델링 된다. 초기 반사는 음원에서 청취자 간에 존재하는 반사체에 의한 직접 반사를 나타내고 있으며, 추후 반사는 그러한 반사파가 다시 반사의 반사를 거듭한 끝에 난반사의 형태를 띄어 청취자에게 도달하는 것을 의미한다.

1.2. 구현 목표

음장 효과의 구현은 룸임펄스와 원신호를 시간 영역에서 선형 복적분을 수행하여 구현할 수 있다. 그러나 시간 영역에서 선형 복적분을 이용하여 음장 효과를 구현하기 위해서는 엄청난 양의 곱 연산을 수행하여야 한다. 이러한 점은 음장 효과의 실시간 구현을 어렵게 한다. 위의 단점은 제한된 룸임펄스를 이용함으로써 곱의 연산량을 줄여 실행 시간을 줄일 수 는 있지만, 이에 따라 룸임펄스의 특성을 그대로 음장 효과에 전달하지 못하는 단점이 또한 발생한다. 본 논문에서는 시간 영역에서의 선형 복적분을 주파수 영역에서의 오버랩세이브 방법으로 대치, 수행하고, 또한 원신호와 룸임펄스 신호를 여러 개의 서브 프레임으로 나눔으로써 연산량 감소의 효과를 가져 올 것이다. 또한 주파수 영역으로의 전환을 위해 FFT (Fast Fourier Transform) 대신 QFT (Quick Fourier Transform)를 이용함으로써 연산량 감소를 유도할 것이다. 세 가지 방법으로 곱의 연산량을 줄임으로써 음장 효과의 실시간 구현 가능성을 보여주며 응답 시간이 긴 룸임펄스를 이용하여 음장 효과를 구현하더라도 정보 손실 없이 룸임펄스의 특성을 반영한 양질의 음장 효과의 결과를 가져올 것이다.

본 논문의 구성은 2장에서 시간 영역에서의 선형 복적분과 주파수 영역에서의 오버랩세이브 방법에 대하여 설명하고, 두 가지 방법에 대한 곱의 연산량을 계산하여 비교한다. 3장에서는 신호를 일정한 크기의 서브프레임으로 나누는 과정과 그에 따른 곱의 연산량을 계산하고 2장의 결과와 비교한다. 4장에서는 기존의 FFT 알고리즘 보다 수행 속도가 향상된 QFT에 대하여 간단히 알아 볼 것이며, 5장에서는 실제 데이터를 이용, 음장 효과를 구현한 결과를 비교 분석하며, 6장에서 결

론을 맺을 것이다.

2. 주파수 영역에서의 복적분

2.1. 시간 영역에서의 복적분

시간 영역에서의 선형 복적분은 다음과 같이 정의할 수 있다.[2]

$$y(n)=\sum_{m}h(n-m)x(m) \tag{1}$$

식 (1)에서 $h(n)$은 룸임펄스를 $x(m)$은 원신호를 나타낸다. 선형 복적분의 연산은 $h(-n)$의 신호를 순차적으로 이동하여 각각의 데이터의 곱과 그 곱의 합의 연산으로써 이루어진다. 따라서 선형 복적분의 곱의 연산량은 각각의 $x(m)$ 신호마다 모든 $h(n)$의 데이터가 곱해지기 때문에 $N_1\times N_2$ 같이 정의할 수 있다. 여기서 N_1, N_2은 각각 x(n)과 $h(n)$의 길이를 의미한다. 선형 복적분의 곱의 연산량은 만약 N_1의 길이가 1이 증가한다면 곱의 연산량은 N_2만큼 증가하게 된다. 이러한 특징으로 인해 N_1, N_2의 길이가 길어질수록 곱의 연산량은 급속도로 증가하게 된다. 이러한 특성으로 인해 실시간 구현이 어려워진다.

2.2. 주파수 영역에서의 복적분

선형 복적분은 주파수 영역에서 오버랩세이브 방법으로 구현될 수 있다[2]. 오버랩세이브를 수행하기 위해서 각각의 신호 x(n)과 h(n)은 FFT 변환 크기로 분할 또는 확장되어야 한다. 먼저, 원신호 x(n)을 벡터열 식 (2)와 룸임펄스 h(n)를 식 (3)과 같이 나타 낼 수 있다.

$$X=[x(0),x(1),\ldots,x(N_1-1)]^T \tag{2}$$

$$H=[h(0),h(1),\ldots,h(M-1)]^T \tag{3}$$

여기서 T는 전치 행열 변환 연산자이다. 이때 $N'\geq M$과 $N'\geq 2^r(r=1,2,\ldots)$을 만족하는 임의의 N'에 대해서 X는 식 (4)와 같이 서브 프래임 벡터열로 표현될 수 있으며, X_k는 식 (5)와 식 (6)같이 나타낸다.

$$X=[X_0, X_1, \ldots, X_l] \tag{4}$$

$$X_k=[0, \ldots, 0, x(N'-M+1), .., x(N'-1)]^T k=0 \tag{5}$$

$$X_k=[x(N'-M+1), .., x(N'+k(N'-M+1))]^T k\neq 0 \tag{6}$$

단, $k=N_1/M+2$이다. $N' \geq M$일 경우 벡터열 H는 다음과 같이 N'의 크기의 백터열 H'로 확장되어야 한다. 또한 룸임펄스는 식 (7)과 같이 나타낼 수 있다.

$$H \cong H'=[\ h(0), h(1), \ldots, h(M-1), 0, \ldots, 0]^T \tag{7}$$

오버랩세이브 방법은 식 (8)와 같이 H'과 X_k의 주파수 영역에서의 곱의 연산으로부터 출력 신호열 Y_k를 구한다.

$$Y_k=IFFT[FFT(x_k) \circ FFT(H')] \tag{8}$$

여기서 IFFT는 고속푸리에 역변환을 나타내며, 연산자 $\circ$는 각각의 벡터열의 곱을 나타낸다. 이때 FFT 수행 후의 각각의 벡터열의 곱은 시간 영역에서 두 벡터열의 순환 복적분의 연산과 같다[2]. H'과 X_k의 시간 영역에서 순환 복적분을 고려하여 Y_k를 다음과 같이 시간 영역에서 처리함으로써 최종 결과를 얻을 수 있다.

$$Y_k=[0, \ldots, 0, y(N'-M+1), .., y(N'-1)]^T \tag{9}$$

이로써 순환 복적분은 선형 복적분 연산으로 대치될 수 있다. 이때 출력 신호열 $y(n)$는 식 (10)으로 얻어진다.

$$Y=[Y_1, Y_2, \ldots Y_L]^T=[y(0), y(1), \ldots, y(N_1-1)]^T \tag{10}$$

2.3. 곱의 연산량

주파수 영역에서의 복적분 방법인 오버랩세이브의 곱의 연산량을 계산해 보면 다음과 같다. 룸임펄스 $h(n)$의 길이를 N_2, $x(n)$의 길이를 N_1이라 하고 오버

랩 되는 길이 M-1을 N_2의 길이와 동일하게 정의한다. 또한 $N_1 \geq N_2$이라고 가정하며 실제 FFT의 크기는 $2N_2$로 계산하게 된다. FFT 크기가 $2N_2$이 되는 이유는 실험 결과에 의해 얻어진 값이며 뒷부분에서 다시 설명할 것이다. 우선 N'-Point FFT를 수행하기 위하여 필요한 곱의 연산량은 $(N'/2)\times\log_2(N')$으로 정의할 수 있다. 여기서 N'는 $2N_2$로 정의한다.

오버랩세이브를 수행하기 위해서는 총 3번의 N'-Point FFT를 수행하게 된다. 즉, 원신호와 룸임펄스 신호의 FFT와 Y_k를 얻기 위한 IFFT를 수행하게 된다. 그러므로 FFT를 위한 곱의 연산량은 식 (11)과 같아진다.

$$V'' = (N_2 \times \log_2(2N_2)) \times (2N_1/N_2 + 1) \tag{11}$$

다음으로 주파수 영역에서의 곱의 연산량은 $2N_2$의 길이를 갖는 두개의 벡터열의 곱이므로 $2N_2$이 된다. 그러나 이는 복소수의 곱이기 때문에 실수의 곱으로 바꾸어 계산해주어야 한다.

$$(a+bj)(c+dj) = ac - bd + ad + bc \tag{12}$$

식 (12)에서 보는 것과 같이 한번의 복소수의 곱은 ac, ad, bc와 bd 와 같이 4번의 실수 곱이 수행되어진다. 결과적으로 주파수 영역에서의 총 곱의 연산량 V'는 $2N_2$에 4를 곱해준 식 (13)로 정의할 수 있다.

$$V' = (N_2 \times \log_2(2N_2) + 4 \times 2N_2) \times (2N_1/N_2 + 1) \tag{13}$$

<표 1> 곱의 연산량 비교

N_2 $N_1 = 2N_2$	시간 영역에서의 수행	주파수 영역에서의 수행	감소율
8 points	64 번	384 번	0.16
128 points	16,384 번	8,192 번	2
512 points	262,144 번	36,864 번	7.11
2,048 points	4,194,304 번	163,840 번	25.64
4,096 points	16,777,216 번	344,064 번	48.78

<표 1>에서 시간 영역에서 선형 복적분의 곱의 연산량과 주파수 영역에서의 오버랩세이브의 곱의 연산량을 비교하였다. <표 1>에서 볼 수 있듯이 N_2를 8 points로 설정했을 때 곱셈의 감소율은 오히려 주파수 영역에서의 곱의 연산이 많은 것으로 나타나 있다. 이는 FFT 연산량 때문이다. 즉, FFT를 위한 곱의 연산량이 주파수영역에서 수행함으로써 얻는 곱의 감소율 보다 크기 때문이다. 그러나 N_2의 point 수가 증가할수록 시간 영역에서의 곱의 연산량은 급속도로 증가하는 반면 주파수 영역에서의 곱의 연산량은 상대적으로 작은 비율로 증가하게 된다. 때문에 N_2의 point가 크면 클수록 곱셈에서의 감소율은 많아지는 결과를 가져온다.

3. 신호의 서브 프레임화

신호를 여러 개의 서브 프레임으로 나누어주는 이유는 FFT를 수행할 때 발생하는 곱의 연산량을 줄여주기 위함이다. 신호의 서브 프레임화는 룸임펄스 h(n)와 원신호 x(n)을 일정한 크기 L로 나누어준다. L로 나누어준 각각의 서브 프레임 $h_k(n)$에 대하여 $x_k(n)$과의 오버랩세이브를 수행하게 된다. N-Point FFT에서 N-point를 L 크기를 갖는 여러 개의 서브프래임으로 나누어준 후의 FFT를 위한 총 곱의 연산량은 식 (14)이 된다.

$$N/(2L)\times\log_2(N/L)\times L = N/2\times\log_2(N/L) \tag{14}$$

그러므로 여러 개의 서브프레임으로 나누어 오버랩세이브를 수행하였을 때 총 곱의 연산량은 식 (13)과 식 (14)을 이용하여 구하면 식 (15)과 같이 정의될 수 있다.

$$\begin{aligned} V = {} & 2N_2\times\log_2(2N_2/L)+(N_1+N_2)/L+N_2/2\times\log_2(N_2/L) \\ & +4\times(N_1N_2/L^2)-1 \end{aligned} \tag{15}$$

<표 2> 곱의 연산량 비교

N_2 $N_1=2N_2$	L	주파수 영역에서의 수행	주파수영역 + 프레임화	감소율
8 points	4 points	384 번	91 번	4.34
128 points	64 points	8,192 번	991 번	8.27
512 points	256 points	36,864 번	3,871 번	9.52
2,048 points	1,024 points	163,840 번	15,391 번	10.64
4,096 points	2,048 points	344,064 번	30,751 번	11.19

<표 2>는 주파수 영역에서 음장 효과를 수행하였을 때의 곱의 연산량과 신호를 여러 개의 서브프레임으로 나누어 주파수 영역에서 수행한 곱의 연산량을 비교하여 나타내었다. <표 2>에서 볼 수 있듯이 신호를 여러 개의 서브 프레임으로 나누어 주파수 영역에서 수행함으로써 곱의 연산량을 상당 부분 줄일 수 있다. 또한 <표 1>의 경우와 마찬가지로 FFT point가 증가할수록 곱의 연산량에 대한 감소율이 커지게 된다. <표 2>에서 L를 FFT point의 절반의 값으로 정한 이유는 <표 3>에 있다. <표 3>에서 보는 것과 같이 동일한 FFT 크기에 대하여 각각 다른 크기의 L을 적용하였을 때 L이 값이 적을수록 곱의 연산량이 증가하게 된다. 이는 L이 값이 적어질수록 식 (15)에서 $(N_1+N_2)/L$와 $4\times(N_1N_2/L^2)$의 부분에서 증가하는 곱의 연산량보다 $2N_2\times\log_2(2N_2/L)$와 $N_2/2\times\log_2(N_2/L)$에서 감소시키는 곱의 연산량이 상대적으로 적기 때문에 L 값이 작을수록 연산량이 증가하기 때문이다.

<표 3> 곱의 연산량 비교

N_2 $N_1=2N_2$	L	주파수영역 + 프레임화
4,096 points	4 points	8,511,478 번
4,096 points	64 points	114,687 번
4,096 points	256 points	63,487 번
4,096 points	1,024 points	41,087 번
4,096 points	2,048 points	30,751 번

4. QFT (Quick Fourier Transform)

주파수 영역으로의 전환을 위한 방법으로 FFT 대신 QFT를 이용함으로써 곱의 연산량을 줄일 수 있다. QFT는 FFT와는 달리 데이터의 길이와 상관없이 빠른 DFT (Discrete Fourier Transform)를 수행할 수 있다. 임의의 길이를 가진 데이터의 QFT는 입력을 실수와 허수 부분으로 나누고 또 다시 각 부분을 우함수와 기함수 부분으로 나누어서 DFT 공식에서 식 (16)에 곱하여 N만큼 더했을 때 기함수 부분은 0이 되고 우함수 부분은 대칭성을 사용하여 계산할 수 있으므로 계산량을 줄일 수 있다.

$$e^{-j2\pi nk/N} = \cos(2\pi nk/N) - j\sin(2\pi nk/N) \tag{16}$$

만약 데이터의 길이가 2^M이라면 QFT는 궤환 공식을 이용해서 Discrete Cosine Transform과 Discrete Sine Transform의 사용으로 더 빠른 계산 효과를 가져올 수가 있다.

$$DFT(k,N,x) = \sum_{n=0}^{N-1} x(n) e^{-j2\pi nk/N} \qquad k=0,\cdots,N-1 \tag{17}$$

$$DCT(k,N+1,x) = \sum_{n=0}^{N} x(n)\cos(\pi nk/N) \qquad k=0,\cdots,N \tag{18}$$

$$DST(k,N-1,x) = \sum_{n=1}^{N-1} x(n)\sin(\pi nk/N) \qquad k=1,\cdots,N-1 \tag{19}$$

여기서 식 (17)는 Discrete Fourier Transform(길이 N)를 나타내며 식 (17)은 다시 식 (18)은 Discrete Cosine Transform(길이 N+1)와 식(19) Discrete Sine Transform(길이 N-1)로 나타낼 수 있다. sine과 cosine의 대칭성 관계를 알아보면 식 (20), 식 (21)과 같고 우함수와 기함수는 식 (22), 식 (23)으로 나타낼 수 있는 것을 알 수 있다.

$$\cos\left(\frac{2\pi(N-n)k}{N}\right) = \cos\left(\frac{2\pi nk}{N}\right) \tag{20}$$

$$\sin\left(\frac{2\pi(N-n)k}{N}\right) = -\sin\left(\frac{2\pi nk}{N}\right) \tag{21}$$

$$x_e(n) = x(n) + x(N-n) \qquad n=1,\cdots,N/2-1 \tag{22}$$

$$x_o(n) = x(n) - x(N-n) \qquad n=1,\cdots,N/2-1 \tag{23}$$

위의 식을 길이 (N/2+1)의 DCT와 길이 (N/2-1)의 DST로 만들면 식 (24)와 식 (25)와 같다.

$$DFT(k,N,x) = DCT(k,N/2+1,x_e) - jDST(k,N/2-1,x_o)$$
$$k = 1,\cdots,N/2-1 \quad (24)$$

여기서 $DFT(0,N,x) = DCT(0,N/2+1,x_e)$이고,

$$DFT(N-k,N,x) = DCT(k,N/2+1,x_e) + jDST(k,N/2-1,x_o)$$
$$k = 1,\cdots,N/2-1 \quad (25)$$

여기서 $DFT(N/2,N,x) = DCT(N/2,N/2+1,x_e)$이다.

그리고 우리는 Fast DCT, DST 알고리즘을 사용하며, 또한 궤환 분해(recursive decomposition)를 DCT, DST에 각각 적용함으로써 곱의 연산량을 줄일 수 있다[3].

5. 실험 및 결과

5.1. 실험 환경

실험 환경은 <표 4>과 같다. 원신호는 무향실에서 트럼펫 연주를 녹음하여 사용하였다. 룸임펄스는 장충체육관에서 임펄스성 신호를 발생하여 녹음하였다. 룸임펄스는 녹음 위치에 따라 초기 반사가 도달하는 시간, 추후 반사의 크기 등이 달라지기 때문에 각기 다른 형태를 가진다. 본 실험에서는 같은 높이의 전방 10m에서 녹음 된 룸임펄스를 사용하였다.

5.2. 실험 결과

<표 4> 실험 환경

	원신호	룸임펄스
녹음 장소	무향실	장충 체육관
샘플링 주파수	44.1 kHz	44.1 kHz
모 드	모노	모노
실행 시간	18.5 초	2.9 초
데이터 량	819,200 개	129,687 개
C P U	500 MHz	

<표 5> 실험 결과

		QFT 제외한 수행시간	QFT 수행 시간	총 수행 시간
선형복적분	.	4802 초	.	4802 초
주파수 영역 & 프레임화 (QFT 크기)	512	121 초	0.81 초	121 초
	2,048	29 초	6 초	35 초
	8,192	10 초	7 초	17 초
	32,768	4 초	10 초	14 초
	65,536	2 초	16 초	18 초

실험은 총 3회에 걸쳐 수행 한 실행 시간의 평균값을 이용하여 비교하였다. 실험 결과 시간 영역에서 선형 복적분을 이용하여 음장 효과를 실행하였을 때 걸린 시간은 약 80분인 반면에 주파수 영역에서의 오버랩세이브와 신호의 서브 프레임화와 QFT를 이용하여 음장 효과를 실행하였을 때 걸린 시간은 17초이다. 이는 곱의 연산량의 감소로 인한 것으로 79분 43초, 약 282배의 시간을 단축할 수 있었다. <표 5>는 QFT 수행 시간과 QFT 수행 시간을 제외한 데이터들의 곱과 합의 수행 시간을 QFT 크기에 따라 나타내었다. 여기서 총 수행 시간을 결정하는데 비중을 차지하는 요소를 분석해 보면, QFT 크기가 작을수록 QFT를 수행하기 위한 실행 시간의 비중은 거의 없는 반면 순수 곱의 연산 시간의 비중이 커지게 된다. 이유는 QFT 크기가 작을수록 IQFT 이후의 더하기 연산의 급증으로 수행 시간이 길어져 순수 연산의 수행 시간이 커지기 때문이다. 또한 QFT 크기가 커질수록 순수 연산 시간은 줄어 총 수행 시간을 결정하는데 비중이 작아지는 반면 QFT를 수행하기 위한 시간이 증가하여 총 수행 시간을 결정하데 비중이 커지게 된다. 즉 QFT 크기가 어느 이상 커지게 되면 총 수행 시간이 다시 증가하게 된다. 그러므로 적절한 QFT 크기의 선택이 요구된다. 실험 결과 QFT의 크기가 32,768일 때 총 수행 시간이 가장 짧게 나타났으나 하드웨어 구현 시에는 32,768은 문제가 발생할 수 있으므로 8,192를 선택하는 것이 바람직 할 것이다.

6. 결 론

본 논문은 시간 영역에서 선형 복적분을 수행함으로써 발생하는 곱의 연산량 문제를 세 가지 방법을 이용하여 상당 부분 해결하였다. 첫 번째 방법은 시간 영역에서의 선형 복적분을 주파수 영역의 오버랩세이브 방법을 이용함으로써 연산량 감소를 유도하였다. 여기서 두 데이터의 길이가 커질수록 연산량의 감소율이

커진다는 것을 알 수 있었다. 두 번째 방법인 신호의 서브 프레임화를 통하여 FFT 내부의 연산량을 줄임으로써 연산량 감소를 유도하였다. 마지막으로 FFT 대신 QFT를 수행함으로써 곱의 연산량을 줄일 수 있었다. 최종 실험 결과 QFT 크기가 작을수록 QFT 제외한 수행 시간이 증가하여 총 수행 시간이 증가하는 결과를 가져 왔으며, QFT 크기가 커질수록 QFT 제외한 수행 시간은 줄어드는 반면 QFT 수행 시간이 증가하여 총 수행 시간이 증가한 sruf과를 가져 왔다. 때문에 위의 사항을 고려한 적절한 QFT 크기의 선택이 요구된다. 실험 결과 곱의 연산량 감소를 유도하여 실시간 구현의 가능성을 제시하였으며 다양한 종류의 룸임펄스를 제한없이 사용하여 양질의 음장 효과를 구현할 수 있었다.

참 고 문 헌

[1] J. Atagi, "Effect of modulated delay time of reflection on autocorrelation function and perception of echo", *Journal of sound and vibration,* Vol. 258, pp.443-450, 2002

[2] A. V. Oppenheim, R. W. Schafer, "Discrete-time signal processing", New Jersey: Prentice Hall, 1999.

[3] H. Guo, S. Member, "The Quick Fourier Transform: An FFT Based on Symmetries", *IEEE Transaction on signal processing,* Vol. 46, No. 2, 1998.

접수일자: 2003년 9월 1일
게재일자: 2003년 9월 17일

▶ 손성용(Sung Young Son)
주소: 305-732 대전광역시 유성구 화암동 58-4번지 한국정보통신대학원대학교
소속: 한국정보통신대학원대학교(ICU) 음성/음향 정보 연구실
전화: 042) 866-6196
E-mail: thill@icu.ac.kr

▶ 서정일(Joung Il Seo)
주소: 305-732 대전광역시 유성구 가정동 161번지 한국전자통신연구원
소속: 한국전자통신연구원 전파방송연구소 방송 미디어 연구부
전화: 042) 866-6206
E-mail: seoji@etri.re.kr

▶ 한민수(Minsoo Hahn)
주소: 305-732 대전광역시 유성구 화암동 58-4번지 한국정보통신대학원대학교
소속: 한국정보통신대학원대학교(ICU) 음성/음향 정보 연구실
전화: 042) 866-6123
E-mail: mshahn@icu.ac.kr