한국음향학회지 제22권 제4호 pp. 308~314 (2003)

광대역 AMR 음성 압축기를 위한 효율적인 코드북 검색 방법

Efficient Codebook Search Method for AMR Wideband Speech Codec

김 윤 희*, 박 호 종*
(Younhee Kim*, Hochong Park*)

*광운대학교 전자공학과

(접수일자: 2003년 1월 17일; 수정일자: 2003년 3월 17일; 채택일자: 2003년 4월 16일)

3.4 kHz 대역폭을 가지는 협대역 음성 통신의 품질 한계를 극복하여 새로운 고품질 음성 서비스를 제공하기 위하여 7 kHz 대역폭을 가지는 광대역 음성 통신 시스템이 개발중이며, 광대역 음성 신호의 효율적인 압축을 위하여 광대역 AMR 음성 압축기가 개발되었다. 광대역 AMR 음성 압축기는 ACELP 구조를 가지며 뛰어난 음질을 제공하지만 최적의 코드벡터를 찾기 위하여 매우 많은 계산량이 필요한 단점을 가진다. 이와 같은 문제점을 해결하기 위하여 본 논문에서는 광대역 AMR 음성 압축기의 코드북을 효율적으로 검색하는 새로운 방법을 제안한다. 제안하는 방법은 대략적으로 초기 코드벡터를 구하고, 코드벡터의 각 펄스의 중요도를 계산하여 중요도가 낮은 펄스를 제거하고 새로운 펄스를 찾는 과정을 통하여 코드벡터의 성능을 점차적으로 향상시키는 기법을 사용한다. 제안한 방법을 광대역 AMR 음성 압축기에 적용하여 성능을 측정하였으며, 매우 적은 계산량으로 기존의 방법보다 약간 우수한 성능을 제공하는 것을 확인하였다.

핵심용어: 광대역 AMR 음성 압축기, ACELP, 코드북 검색, 계산량
투고분야: 음성처리 분야 (2.2)

Wideband speech communications with 7 kHz bandwidth can provide high-quality speech services that are almost impossible with current narrow-band speech communications with 3.4 kHz bandwidth, and AMR wideband codec was recently developed for these services. The performance of AMR wideband codec is excellent due to its wideband information and partially to ACELP structure, but it requires high computational complexity especially in codebook search. In this paper, to solve this problem, an efficient codebook search method for AMR wideband codec is proposed. The proposed method first determines the coarse initial codevector, then improves the performance of codevector by replacing a poor pulse in codevector with better one iteratively. Simulations show that AMR wideband codec with proposed codebook search method has higher performance with much less computational cost than conventional AMR wideband codec.

Keywords: *AMR wideband codec, ACELP, Codebook search, Computational complexity*
ASK subject classification: *Speech signal processing (2.2)*

I. 서 론

현재 대부분의 디지털 음성 통신은 입력 음성 신호를 8 kHz로 샘플링하고 300~3400 Hz 주파수 대역만을 전송하는 협대역 (narrow-band) 통신을 한다. 또한 디지털 통신에서 전송 데이터의 양을 줄여 채널을 효율적으로 사용하기 위하여 음성 압축기가 사용되며, 협대역의 음성 신호를 처리하는 협대역 음성 압축기로 EVRC, QCELP, G.729, AMR 등이 널리 사용되고 있다[1]. 최근 들어 네트워크가 완전 (All) IP망으로 진화하고 통신 기술이 발전함에 따라 현재보다 많은 대역폭을 사용할 수 있을 것으로 기대되고, 동시에 고품질 음성 서비스에 대한 소비자들의 요구가 대두됨에 따라 음성 품질을 높이기 위한 기술적 노력이 진행되고 있다. 그 중에 하나가 음성신호를 16 kHz로 샘플링하고 7 kHz까지의 주파수 성분을 전달하는

책임저자: 박호종 (hcpark@mail.kw.ac.kr)
139-050 서울 노원구 월계동 447-1
광운대학교 전자공학과
(전화: 02-940-5104; 팩스: 02-2291-1793)

광대역 (wideband) 통신이며, 광대역 통신은 매우 높은 주파수 대역의 정보를 전달함으로써 기존의 협대역 통신에 비하여 음성의 자연성이나 음질이 매우 향상되며, 또한 음성뿐만 아니라 음악 신호도 큰 왜곡없이 효과적으로 전송할 수 있어 멀티미디어 통신을 가능하게 한다.

이와 같은 광대역 음성 통신을 위하여 광대역 음성 신호를 효과적으로 압축하는 광대역 음성 압축기가 필요하고, 이에 따라 3GPP의 광대역 AMR 음성 압축기 (adaptive multi-rate wideband codec: AMR-WB)가 표준화되었고 3GPP2에서도 광대역 음성 압축기의 표준화를 추진 중에 있다[2]. AMR-WB를 이용한 통신은 7kHz의 대역폭을 가지고 있어 고품질의 음성 서비스를 제공하지만 대역폭이 증가하고 더 많은 정보의 처리가 필요함에 따라 협대역 AMR 음성 압축기 (AMR-NB)에 비해 대략 두 배의 계산량이 필요하다[3]. 따라서 기존의 협대역 음성 압축기를 위한 시스템이 제공하는 계산량으로는 AMR-WB를 동작시키는 것이 불가능하고, 결국 AMR-WB를 위하여 새로운 음성처리 모듈의 개발이 필요하게 된다. 또한 계산량의 증가는 하드웨어의 가격을 상승시키고 전력의 소모를 증가시키는 문제점을 수반한다. 만일 성능의 저하없이 AMR-WB의 계산량이 AMR-NB와 비슷하게 되면, 협대역 통신에서 광대역 통신으로의 변환을 위하여 음성 압축기 모듈의 DSP 칩은 그대로 사용하고 단지 압축기의 소프트웨어만 교체하고 샘플링 주파수의 변경에 따른 부품만 교체하면 될 것이다.

따라서 본 논문에서는 광대역 음성 압축기의 계산량 증가에 의한 문제점을 해결하기 위하여 AMR-WB에서 많은 계산량을 차지하는 ACELP 코드북 검색을 위한 새로운 효율적 방법을 제안하고, 이를 AMR-WB에 적용하여 계산량과 성능을 분석한다. 2장에서는 AMR-WB의 전체 동작과 코드북 검색을 위한 기존의 표준 방법을 알아보며, 3장에서는 제안하는 코드북 검색 방법을 소개한다. 4장에서는 제안한 코드북 검색 방법을 AMR-WB에 적용하여 기존의 방법과 비교하여 성능과 계산량을 분석하고, 5장에서 결론을 맺는다.

II. 광대역 AMR 음성 압축기

2.1. 구조 및 동작

AMR-WB는 3GPP에서 IMT-2000의 표준으로 채택한 표준 광대역 음성 압축기이다[2]. AMR-WB는 16 kHz로 샘플링된 신호를 입력으로 사용하며, AMR-NB와 동일하게 주어진 채널 환경에 따라 최적의 음질을 유지하기 위하여 통신 채널의 상태에 따라 소스 코딩과 채널 코딩의 비트 할당을 달리하는 기법을 활용한다[2,3]. 채널당 할당된 비트수가 항상 고정되어 있기 때문에 채널에 오류가 많은 경우 채널 코딩에 많은 비트를 할당하고 소스 코딩에는 상대적으로 적은 비트를 할당하여 채널 오류에 의한 정보 손실을 최소로 하고, 이와 반대로 채널에 오류가 적은 경우에는 소스 코딩에 많은 비트를 할당하여 음질의 향상에 주력한다. 한 프레임은 20 ms이고, 4개의 부프레임 (subframe)을 가지며, 6.60 kbit/s에서 23.85 kbit /s까지 9개의 모드 (mode)를 가지며, 각 모드별 전송률은 표 1과 같다.

차단 주파수가 50 Hz인 고역 통과 필터를 사용하여 입력 신호의 저주파 성분을 제거하고 전처리 과정을 통과한 후 LPC, LTP, 코드북 파라미터 추출을 위해 12.8 kHz로 다운 샘플링한다. LPC 분석은 한 프레임에 한 번 수행하며 비대칭 윈도우를 사용한다. 5ms look-ahead를 사용하고 Levinson-Durbin 알고리즘을 이용해 LPC 계수를 구한다. 계산된 LPC 계수는 코사인 영역에서 채비세프 다항식을 이용하여 ISP (immittance spectral pair)로 변환한 후 양자화한다.

적응 코드북 검색은 AbS (Analysis-by-synthesis)방식에 의해 수행되며 먼저 검색을 간단하게 하기 위하여 개루프 검색을 통해 정수 지연 값을 찾고 그 값 주변을 세밀하게 폐루프 검색한다. 개루프 검색은 6.60 kbit/s 모드의 경우 매 프레임마다 한 번, 다른 모드들은 매 프레임에 두 번씩 실시한다. 폐루프 검색은 부프레임 단위로 수행하며 6.60 kbit/s과 8.85 kbit/s 모드의 경우 1/2, 나머지 모드는 1/4의 분해능을 사용한다.

음성 스펙트럼의 포먼트 성분과 주기적인 특성이 추출되고 남는 잔여 신호를 ACELP 코드북을 이용하여 양자화

표 1. AMR-WB의 각 모드별 전송률
Table 1. Bit rates of AMR-WB.

Mode	1	2	3	4	5	6	7	8	9
Bit Rate (kbit/s)	6.60	8.85	12.65	14.25	15.85	18.25	19.85	23.05	23.85

한다. 트랙별로 제한된 위치에 펄스가 존재하고 각 펄스의 크기가 +1과 −1로 이루어지며, 펄스의 수가 증가할수록 다양한 형태의 신호를 표현할 수 있어 음질이 향상되는 장점이 있으나 계산량이 늘어나는 단점을 가진다. 코드북 검색은 다음의 과정으로 진행된다. 먼저, 입력 신호에서 LPC 필터와 적응 코드북으로 표현되는 성분을 제거하여 코드북 검색의 목표 신호 $x_2(n)$을 구한다. 다음 코드북의 각 코드벡터 c_k와 가중치가 적용된 LPC 필터의 충격응답 $h(n)$의 컨벌루션으로 $z(n)$을 구하여 $x_2(n)$과 $z(n)$의 차이를 최소로 하는 c_k를 최적의 코드벡터로 정한다. 이러한 과정을 하나의 식으로 간단히 정의하면 식 (1)로 정의된 Q_k 값을 최대로 하는 c_k가 최적의 코드벡터이다.

$$Q_k = \frac{\left(\sum_{n=0}^{N-1} x_2(n)z(n)\right)^2}{\sum_{n=0}^{N-1} z(n)z(n)} = \frac{(d^t c_k)^2}{c_k^t \Phi c_k} \qquad (1)$$

여기서 c_k는 인덱스 k일 때 코드벡터, d는 목표 신호와 충격 응답의 상관관계, Φ는 충격 응답의 상관관계 행렬이다.

2.2. 표준 코드북 검색 방법

AMR-WB의 고정 코드북의 트랙 구조와 펄스 수는 표 2와 같다. 6.60 kbit/s 모드를 제외한 나머지 8개의 모드들은 4개의 트랙 T0, T1, T2, T3를 가지며, 모드에 따라 한 트랙당 1개에서 6개씩의 펄스를 가지게 되며, 특히 14.25 kbit/s와 19.85 kbit/s 모드에서는 첫 번째 트랙 T0와 두 번째 트랙 T1에서 가지는 펄스의 개수가 세 번째 트랙 T2와 네 번째 트랙 T3에서 가지는 펄스의 개수보다 한 개씩 더 많다.

최적의 코드벡터 검색은 5 msec (64 샘플)의 부프레임 단위로 수행한다. 이와 같이 광대역 신호의 처리를 위하여 부프레임의 길이가 길어지고, 각 부프레임에 포함되는 펄스의 수가 많아짐에 따라 최적의 코드벡터를 검색하는데 매우 많은 계산량이 요구된다. 계산량을 줄이기 위하여 표준 검색 방법에서는 각 펄스의 부호를 별도로 구하지 않고 미리 구하여진 특정 신호 $b(n)$ 값에 따라 부호를 결정하며, 각 펄스에 해당하는 위치에서 $b(n)\geq 0$이면 양의 부호를 가지고 $b(n)<0$이면 음의 부호를 가지도록 고정시킨다.

트랙의 수가 4개인 모드 중에서 가장 간단한 8.85

표 2. AMR-WB의 ACELP 코드북 구조
Table 2. Algebraic codebook structure of AMR-WB.

(a) 트랙 구조
(a) Track structure

Track	Pulse Position															
T0	0	4	8	12	16	20	24	28	32	36	40	44	48	52	56	60
T1	1	5	9	13	17	21	25	29	33	37	41	45	49	53	57	61
T2	2	6	10	14	18	22	26	30	34	38	42	46	50	54	58	62
T3	3	7	11	15	19	23	27	31	35	39	43	47	51	55	59	63

(b) 모드별 트랙 개수와 한 부프레임에서의 펄스 개수
(b) The number of tracks and subframe pulses for each mode

Mode (kbit/s)	# of Tracks	# of Pulses	# of Pulses for each track
23.85	4	24	6
23.05	4	24	6
19.85	4	18	T0,T1 : 5
			T2,T3 : 4
18.25	4	16	4
15.85	4	12	3
14.25	4	10	T0,T1 : 3
			T2,T3 : 2
12.65	4	8	2
8.85	4	4	1
6.60	2	2	1

kbit/s 모드를 예로 하여 표준 검색 방법을 간단히 설명한다. 먼저 T0에서 $b(n)$ 값이 큰 펄스 위치 4개와 T1의 16개의 모든 펄스 위치 조합에 대해 식 (1)의 Q_k 값을 구하여 가장 큰 Q_k 값을 가질 때의 펄스 위치를 선택한다. 즉, T0와 T1에서 각각 한 개씩의 펄스를 가지는 임시 코드벡터에 대하여 Q_k 값을 구하고 최대값을 가지는 경우를 구한다. 이 검색에서는 T0에서 4가지, T1에서 16가지의 펄스 위치가 가능하므로, 총 4×16 번의 Q_k 값 계산이 필요하다. 다음, 이와 같이 구하여진 T0와 T1에서의 두 개의 펄스를 고정시키고, T2에서 $b(n)$값이 큰 펄스 위치 8개와 T3의 16개의 모든 펄스 위치 조합에 대해 Q_k 값을 구하여 가장 큰 Q_k 값을 가질 때의 펄스 위치를 2개 추가로 선택하며, 이 검색을 위하여 총 8×16 번의 Q_k 값 계산이 필요하다. 이와 같이 4개의 펄스로 구성된 코드벡터를 구할 때, 계산량을 줄이기 위하여 4개 펄스의 모든 조합에 대하여 검색하지 않고 2개씩 묶어 검색하는 방법을 사용하여 Q_k 값을 (4×16 + 8×16)번만 계산한다. 만일 모든 조합에 대하여 검색을 하면 16×16×16×16 번의 Q_k 계산이 필요하다.

이와 같이 하나의 코드벡터 후보를 찾게 되며, 펄스를 찾는 트랙의 순서를 바꾸어 동일한 방법으로 코드벡터 후보를 3개 더 찾는다. 즉, T1-T2에서 2개의 펄스를 찾고 다시 T3-T0에서 2개의 펄스를 찾아 또 다른 코드벡터 후보를 구하며, 검색하는 트랙의 순서가 다르므로 검색 결과도 다르게 된다. 이와 같이 하여 총 4개의 코드벡터 후보를 구하고, 이 중에서 가장 큰 Q_k를 가지는 코드벡터를 최종 코드벡터로 결정하게 된다.

최적의 코드 벡터를 찾기 위하여 서로 다른 펄스 위치 조합을 가지는 매우 많은 수의 코드벡터에 대하여 Q_k 값을 모두 계산하고 이 중에서 최대값을 가지는 경우를 최적으로 정하게 되므로, 본 논문에서 코드북 검색의 계산량을 최적의 코드 벡터를 찾기 위하여 계산되는 Q_k 값의 회수로 정의한다. 이는 결국 검색 과정에 포함되는 코드벡터의 수와 같으며, 가능한 코드벡터를 모두 검사하지 않고 가능성이 높은 일부분의 코드벡터만 검사하여 최적의 코드벡터를 찾는 것이 효율적인 코드북 검색 방법의 목표이다. 이와 같은 계산량 정의에 의하면 8.85kbit/s 모드의 코드북 검색 계산량은 (4×16 + 8×16) × 4 = 768이 된다.

나머지 모드의 코드북 검색 방법은 펄스 수에 따라 약간씩 다르며, 근본적으로 앞에서와 같이 두 트랙에서 두 개의 펄스를 찾는 과정을 순차적으로 반복하여 원하는 수의 펄스를 구하는 방법을 사용한다. 자세한 검색 방법은 [2]을 참조한다.

III. 제안하는 코드북 검색 방법

기존의 ACELP 코드북 검색 방법은 주어진 여러 개의 코드벡터에 대하여 Q_k 값을 모두 계산하고 최대의 Q_k 값을 가지는 하나의 코드벡터를 찾는 방식이다. 그러나, 이 방법이 최적에 근접한 우수한 성능의 코드벡터를 찾기 위하여서는 매우 많은 코드벡터 후보에 대한 Q_k 값을 모두 계산하여 하고, 이로 인하여 많은 계산량을 가지게 된다. 이에 비하여 제안하는 코드북 검색 방법은 많은 코드벡터 후보에 대하여 Q_k 값을 계산하여 성능을 비교하는 것이 아니라, 임의의 코드벡터를 정하고 이 코드벡터의 성능을 점차적으로 향상시켜 최적에 근접한 코드벡터를 구하는 방식이다. 여기서 코드벡터의 성능은 Q_k 값으로 정의되므로 주어진 코드벡터의 펄스 위치를 적절히 변경하여 Q_k 값이 계속적으로 증가하도록 하는 방식이다. 즉, 기존의 "비교 검색" 개념이 아니라 "성능 개선" 개념으로 최적의 코드벡터를 찾는 것이며, [4]에서 최초로 제안한 "펄스 교환 방법"에 기초한다

펄스 교환 방식은 하나의 초기 코드벡터를 대략적으로 찾고 그 코드벡터를 이루는 각각의 펄스를 더 좋은 위치의 다른 펄스로 바꾸어 코드벡터의 성능을 향상시키는 과정으로서 다음의 두 단계로 구분한다. 첫 번째 단계에서는 매우 간단한 방법 (예로, 완전 순차적 검색 방법)으로 초기 코드벡터를 대략적으로 찾는다. 다음, 일반적으로 초기 코드벡터의 성능은 매우 떨어지므로 초기 코드벡터의 성능을 향상시키기 위하여 코드벡터를 이루는 각 펄스의 중요도를 계산하여 중요도가 낮은 펄스를 제거하고 새로운 펄스로 교환한다. 펄스의 중요도를 적절하게 정의하고 Q_k 값이 증가하도록 새로운 펄스를 선택하면, 펄스 교환에 의한 새로운 코드벡터의 Q_k 값은 원 코드벡터의 Q_k 값보다 커지게 되어 성능이 향상된다. 이와 같은 펄스 교환을 수 차례 반복하게 되면 코드벡터의 성능이 계속 향상되고 점차적으로 최적의 코드 벡터에 근접하게 된다. 중요도가 낮은 펄스를 제거하고 동일 트랙에서 새로운 펄스를 선택하면 절대 Q_k 값이 감소하지는 않으며, 만일 제거되었던 펄스가 다시 선택되면 Q_k 값은 변하지 않게 되고 성능 향상이 멈추게 되어 검색을 마치게 된다. 이상의 코드북 검색 방법의 흐름도가 그림 1에 주어

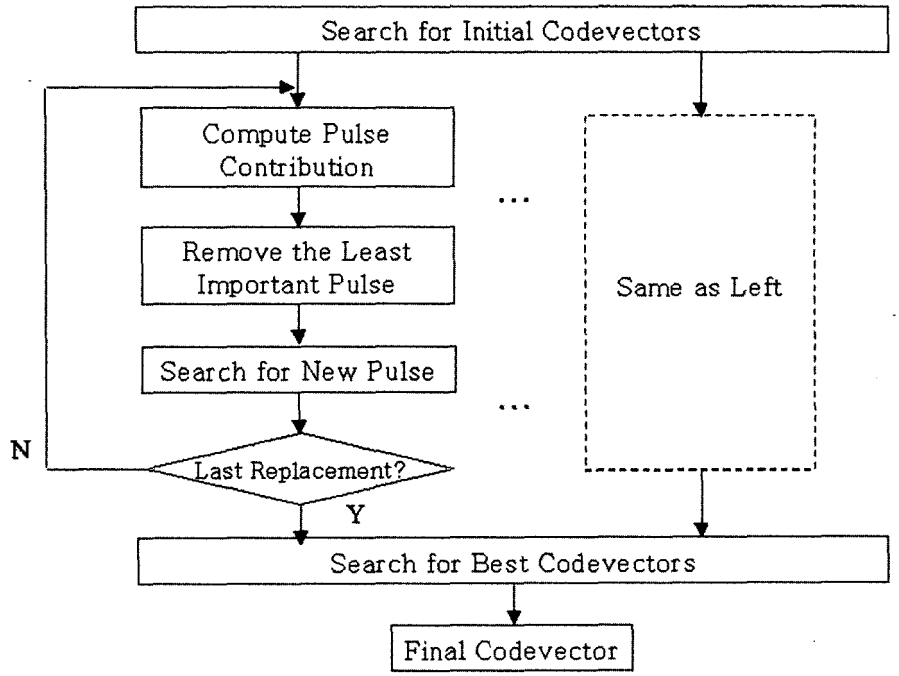

그림 1. 제안하는 코드북 검색 방법의 흐름도
Fig. 1. Flowchart of proposed codebook search method.

졌다.

제안하는 코드북 검색 방법에서 가장 중요한 것은 각 펄스의 중요도를 계산하는 방법이며, 본 논문에서는 다음과 같이 펄스 중요도를 정의한다. 예로 4개의 펄스로 구성된 코드벡터에서 각 펄스를 p_0, p_1, p_2, p_3라 하자. 이 때, p_0를 제외하고 나머지 3개의 펄스 p_1, p_2, p_3로 구성된 코드벡터의 Q_k 값을 구하고, 만일 p_0 제거 후에 Q_k 값이 크다면 이는 p_0 없이도 계속 우수한 성능의 코드벡터를 유지하는 것을 의미하므로 그 만큼 p_0의 역할은 작게 된다. 반면에 p_0 제거 후에 Q_k 값이 매우 작아지면 성능 저하의 원인이 p_0의 제거이므로 p_0의 역할은 매우 크게 된다. 따라서, 각 펄스에 대하여 펄스를 제거한 후 코드벡터의 Q_k 값을 계산하여 이 값의 역수를 해당 펄스의 중요도로 정의한다. 더욱이 이와 같이 하면 Q_k 값을 통하여 코드벡터의 성능뿐만 아니라 각 펄스의 중요도도 계산하게 되어 동일한 계산 모듈을 공동으로 사용할 수 있게 된다.

제안하는 코드북 검색 방법은 초기 코드벡터로부터 시작하여 계속적으로 향상된 코드벡터를 구하므로 최종 코드벡터는 초기 코드벡터에 의존한다. 따라서 보다 우수한 성능의 코드벡터를 구하기 위하여 그림 1과 같이 다수의 초기 코드벡터를 선택하고, 각각에 대하여 펄스 교환을 적용하여 모든 결과 중에서 최종적으로 하나의 코드벡터를 선택하도록 한다.

제안한 코드북 검색 방법의 계산량은 {[초기 코드벡터 계산량 + (펄스의 개수 + 펄스 검색 위치) × 교환 회수] × 초기 코드벡터의 개수}로 정의된다. 여기서 "펄스의 개수"는 각 펄스의 중요도를 계산하는 부분이며, 각 펄스의 중요도 계산을 위하여 각 펄스 제거 후 Q_k를 계산하므로 이와 같이 정해진다. "펄스 검색 위치"는 새로운 펄스를 찾는 계산량에 해당하며, 펄스가 제거된 트랙에서 다시 한 개의 펄스를 찾기 위하여 해당 트랙의 모든 위치에 대하여 Q_k 값을 계산하므로 이 값을 가진다.

8.85 kbit/s 모드에 대하여 제안한 검색 방법을 보다 자세히 설명한다. 첫 번째 단계에서는 간단하게 4개의 펄스로 구성된 초기 코드벡터를 찾는다. 이 과정은 완전 순차적으로 각 트랙에서 하나의 펄스씩을 검색하며, 각 트랙에 16개의 펄스 위치가 있으므로 4개의 펄스로 구성된 초기 코드벡터를 찾기 위하여 (16+16+16+16)번의 계산량이 필요하다. 또한, 검색 트랙의 순서에 따라 다른 결과가 나오므로 트랙 순서를 T0-T1-T2-T3, T1-T2-T3-T0, T2-T3-T0-T1, T3-T0-T1-T2 등의 4 가지 방법으로 하여 총 4개의 초기 코드벡터를 구한다.

두 번째 단계에서는 각 초기 벡터에 대하여 코드벡터를 이루는 각 펄스의 중요도를 계산한다. 가장 중요도가 낮은 펄스를 제거하고, 제거된 펄스가 있던 트랙의 16개의 위치 중에서 새로운 펄스를 검색하여 다시 4개의 펄스로 구성된 코드벡터를 구한다. 이 과정을 수 차례 반복하여

성능이 향상된 최종 코드벡터를 구한다. 만일 펄스 교환을 2번 실시하면 8.85 kbit/s 모드에서 제안하는 코드북 검색 방법의 계산량은 {(16 × 4) + (4 + 16) × 2} × 4 = 416이고, 이는 표준 방법 계산량인 728에 비하여 약 57%에 불과하다.

다른 모드에서도 동일한 개념으로 초기 코드벡터를 구하고 각 초기 코드벡터에 대한 펄스 교환 방법을 적용하여 적은 계산량으로 우수한 성능의 코드벡터를 구할 수 있다.

IV. 성능 평가

제안한 방법을 6.60 kbit/s 모드를 제외한 AMR-WB에 적용하여 코드북 검색을 위한 계산량과 주관적/객관적 성능을 측정하여 기존의 방법과 비교 분석하였다. 6.60 kbit/s 모드는 표준 검색 방법에서 가능한 모든 코드벡터에 대하여 완전 검색을 하므로 새로운 검색 방법을 적용하는 것이 의미를 가지지 못한다. 표 3은 계산량을 비교한 것으로서 제안하는 코드북 검색 방법은 표준 코드북 검색 방법에 비해 50-60%의 계산량을 가진다.

검색 방법의 성능은 객관적인 방법과 주관적인 방법으로 평가하며, 먼저 객관적인 성능 평가로서 각 방법에 의하여 구하여진 최종 코드벡터의 Q_k 값을 비교하여 분석하였다. 검색의 목표가 Q_k 값이 최대인 코드벡터를 구하는 것이므로 Q_k 값이 검색 방법의 성능을 측정하는 객관적인 기준이 된다. 동일한 환경에서 각 검색 부프레임에 대하여 표준 코드북 검색 방법을 이용하여 구한 코드벡터의 Q_k 값과 제안하는 코드북 검색 방법을 이용하여 구한 코드벡터의 Q_k 값을 구하여 부프레임 단위로 성능비

$$R_n = \frac{\text{제안한 방법에 의한 } Q_k}{\text{표준 방법에 의한 } Q_k},\quad n = \text{부프레임 인덱스}$$

를 구하였다. $R_n \geq 1.0$이면 제안한 방법이 더 우수한 성능의 코드벡터를 찾은 것을 말하며, 전체 부프레임에 대하여 통계적으로 성능을 검증하였다. 예로 R_n의 전체 평균이 1.0이면 통계적으로 동일한 성능의 코드벡터를 찾은 것이다. 또한, 평균뿐만 아니라 전체 부프레임에 대하여 $R_n \geq 1.0$인 부프레임의 빈도(%)를 구하였으며, $R_n \geq 1.0$인 부프레임이 전체 부프레임의 50%이면 동일한 성능의 코드벡터를 찾은 것으로 볼 수 있다. 표 4에 정리되어 있듯이 대부분의 모드와 입력 신호에서 제안한 검색 방법이 표준 방법에 비하여 약간 우수한 성능을 가지는 코드벡터를 찾는 것을 알 수 있다.

주관적인 성능 평가는 영어 여성과 남성, 한국어 여성과 남성 각각 2문장을 표준 코드북 검색 방법과 제안하는 코드북 검색 방법을 사용하는 AMR-WB를 통하여 압축/복원하여 임의의 순서로 들려주고 더 좋은 음질을 가지는 것을 선택하게 하는 A-B 선호도 테스트를 통하여 측정하였다. 표 5에 정리되어 있듯이 동일한 음질이라는 의견이 대부분을 차지하고 있으며 제안하는 코드북 검색 방법의 음질이 표준 코드북 검색 방법을 이용한 것보다 우수하다는 의견이 높았다.

이상의 계산량 비교와 객관적 및 주관적으로 측정한 성능을 분석한 결과, 제안하는 코드북 검색 방법은 매우 적은 계산량으로 표준 검색에 비하여 약간 우수한 성능을 가지는 코드벡터를 찾는 것을 확인할 수 있다.

표 3. 제안한 방법과 표준 방법의 계산량 비교
Table 3. Computational complexity of proposed method and standard method.

23.85	848	2	1	464	51.72
23.05	1696	2	2	928	54.72
19.85	1680	2	3	1068	63.57
18.25	1728	2	3	960	55.56
15.85	1664	2	3	992	59.62
14.25	1664	2	4	848	50.96
12.65	1280	2	4	704	55.00
8.85	728	2	4	416	54.17

표 4. 코드북 검색 방법의 객관적인 성능 비교
Table 4. Objective performance of codebook search methods.

< Average of R_n >

Mode	English Female	English Male	Korean Female	Korean Male	Average
23.85	1.03	1.04	1.06	1.05	1.05
23.05	1.04	1.04	1.07	1.06	1.05
19.85	1.03	1.09	1.09	1.10	1.08
18.25	1.08	1.09	1.10	1.08	1.09
15.85	1.05	1.04	1.05	1.03	1.04
14.25	1.02	1.02	1.02	1.01	1.02
12.65	1.06	1.07	1.05	1.05	1.06
8.85	0.98	0.98	0.97	0.98	0.98

< Percentage of subframes with $R_n \geq 1.0$ >

Mode	English Female	English Male	Korean Female	Korean Male	Average (%)
23.85	58.0	53.1	66.0	61.0	59.5
23.05	54.3	56.4	67.0	67.0	61.2
19.85	75.4	77.6	79.0	80.0	78.0
18.25	75.4	75.0	78.0	80.0	77.1
15.85	62.5	57.9	60.5	56.0	59.2
14.25	52.0	49.0	49.0	44.0	48.5
12.65	74.1	67.0	67.1	67.3	68.9
8.85	44.1	44.0	41.0	40.0	42.3

V. 결 론

본 논문에서는 AMR-WB의 코드북 검색을 위한 새로운 방법을 제안하였다. 기존의 코드북 검색 방법이 다수의 코드벡터 후보를 찾아 그 가운데 가장 우수한 성능을 가지는 하나의 코드벡터를 최종 코드벡터로 선택하는데 반해, 제안하는 코드북 검색 방법은 대략적으로 초기 코드벡터를 찾고 이 코드벡터를 이루는 펄스의 중요도를 따져 중요도가 낮은 펄스를 제거하고 새로운 펄스로 교환하여 점차적으로 코드벡터의 성능을 향상시키는 방법을 사용한다. 제안한 검색 방법을 AMR-WB에 적용하여 성능을 측정하였으며, 제안한 방법이 표준 코드북 검색 방법의 50%~60%의 계산량으로 객관적인 성능 비교인 Q_k 값 비교와 주관적인 성능 비교인 A-B 선호도 테스트에서 보다 우수한 성능의 코드벡터를 검색하는 것을 확인하였다.

표 5. 코드북 검색 방법의 주관적인 성능 비교
Table 5. Subjective performance of codebook search methods.

Mode	Proposed method preferred	No preference	Standard method preferred
23.85	10.71%	71.43%	17.86%
23.05	28.57%	64.29%	7.14%
19.85	35.71%	57.14%	7.14%
18.25	17.86%	75.00%	7.14%
15.85	25.00%	67.86%	7.14%
14.25	25.00%	67.86%	7.14%
12.65	28.57%	67.86%	3.57%
8.85	32.14%	53.57%	14.29%

감사의 글

본 연구는 광운대학교 2002학년도 교내 학술 연구비의 지원으로 이루어졌습니다.

참 고 문 헌

1. L. Hanzo, F. C. A. Somerville, and J. P. Woodard, *Voice Compression and Communications*, IEEE Press, 2001.
2. 3GPP TS 26.190 "AMR wideband speech codec; Transcoding functions," 3GPP Technical Specification, 2001.
3. ETSI EN 301 704, "Adaptive multi-rate (AMR) speech transcoding," 2000.
4. H.-C. Park, "Efficient codebook search method for EVRC speech codec," *IEEE Signal Processing Letters*, 1 (1), 2~3, January, 2000.

저자 약력

● **김 윤 희 (Younhee Kim)**
2001년 2월: 광운대학교 전자공학과 (공학사)
2003년 2월: 광운대학교 전자공학과 (공학석사)
※ 주관심분야: 음성/오디오 신호처리

● **박 호 종 (Hochong Park)**
1986년 2월: 서울대학교 전자공학과 (공학사)
1987년 12월: Univ. of Wisconsin-Madison, 전자공학과 (M.S.)
1993년 5월: Univ. of Wisconsin-Madison, 전자공학과 (Ph.D.)
1993년 9월~1997년 8월: 삼성전자 선임연구원
1997년 9월~현재: 광운대학교 전자공학과 부교수
※ 주관심분야: 음성/오디오 신호처리, 통신 신호처리, 영상 신호처리