한국음향학회지 제22권 제2호 pp. 129~134 (2003)

적응형 다중 비트율 음성 부호화기를 위한 효율적인 대수코드북 검색법

An Efficient Algebraic Codebook Search Method for AMR Speech Coder

변 경 진*, 정 희 범*, 한 민 수**
(Kyung-Jin Byun*, Hee-Bum Jung*, Min-Soo Hahn**)

*한국전자통신연구원, **한국정보통신대학원

(접수일자: 2002년 10월 22일; 채택일자: 2003년 1월 16일)

본 논문에서는 적응형 다중 비트율 (AMR: Adaptive Multi-Rate) 음성 부호화기의 구현 시 계산량을 가장 많이 차지하는 대수 코드북 검색과정의 계산량을 줄임으로써 효율적인 AMR 음성 부호화기를 구현하였다. 대수 코드북 검색의 계산량을 줄이기 위하여 기존의 AMR 음성 부호화기에서 사용하고 있는 깊이우선 가지 검색법 (depth first tree) 검색 방법을 개선한 고속 코드북 검색 알고리즘을 제안하였다. 제안된 방법은 검색과정에서 최적의 여기신호로 선택될 가능성이 적은 트리를 제거하여 검색의 복잡도를 줄이는 방법으로 트리 선택을 위한 추가의 계산량이 필요없으며 검색에 필요한 계산량은 기존의 깊이우선 가지 검색법에 비해 현저한 감소를 이루었으나 약간의 음질 저하가 있었다. 제안한 방법을 적용하여 AMR 음성 부호화기의 12.2 kbps 모드를 TeakLite DSP를 사용하여 구현한 결과 기존의 방법에 비해 약 40%의 계산량을 감소할 수 있었다.

핵심용어: 음성 부호화, 적응형 다중 비트율, ACELP, 대수코드북, 고속검색
투고분야: 음성처리 분야 (2.2)

In this paper, we efficiently implement the AMR speech coder by reducing the complexity of algebraic codebook search. To reduce the computational complexity of the algebraic codebook search, we propose a fast algebraic codebook search method that improves conventional depth first tree search method used in AMR speech codec algorithm. The proposed method reduces the search complexity by pruning the trees which are less possible to be selected as an optimum excitation. This method needs no additional computation for selecting the trees to be pruned and reduces the computational complexity considerably compared to the original depth first tree search method with slightly degradation of speech quality. Applying our method to the implementation of AMR speech coder with 12.2 kbps mode by using the TeakLite DSP, we reduce the search complexity about 40% compared to the conventional method.

Keywords: *Speech coding, AMR, ACELP, Algebraic codebook, Fast search*
ASK subject classification: *Speech signal processing (2.2)*

I. 서 론

디지털 이동통신 시스템에서는 전송채널의 대역폭을 효율적으로 사용하고, 무선채널 환경에서 고음질의 통화를 위하여 다양한 음성코딩 알고리즘들을 사용하고 있다. 일반적으로 CELP (Code Excited Linear Prediction) 알고리즘은 4-8 kbps의 낮은 전송율에서도 고음질을 유지하는 효과적인 코딩 방법 중의 하나이다. 하지만 기존의 CELP 코딩 방법은 코드북 검색을 위한 많은 계산량과 코드벡터를 위한 많은 메모리 사용량 때문에 구현에 어려움이 많았다. 이러한 문제점들을 극복하기 위하여 개발된 ACELP (Algebraic CELP) 알고리즘은 최근까지 G.729, EVRC (Enhanced Variable Rate Coder), 적응형 다중 비트율 (AMR: Adaptive Multi-Rate) 등의 많은 음성코딩 표준들에 채택되고 있다.

AMR 보코더는 12.2 kbps에서 4.75 kbps까지 8가지의

책임저자: 변경진 (kjbyum@etri.re.kr)
305-350 대전광역시 유성구 가정동 161
한국전자통신연구원 통신디지털 회로팀
(전화: 042-860-5831; 팩스: 042-860-6108)

다중 비트율을 가지고 있는 음성 부호화 알고리즘이다[1]. AMR 음성 부호화기에서 채택하고 있는 ACELP 알고리즘은 여기신호를 모델링하기 위한 코드북을 사용하지 않기 때문에 코드북을 위한 저장공간이 필요없고, 코드북 검색 방법도 효율적인 방법들을 사용하기 때문에 적은 계산량으로 검색을 할 수 있다. ACELP 알고리즘에서는 목표신호와 가장 오차를 적게 하는 여기신호의 펄스의 위치와 크기를 검색하여야 하는데 전체 검색방법을 사용하는 경우에는 여전히 많은 계산량이 요구된다. 계산량을 줄이기 위한 대표적인 방법으로는 집중 (focused) 검색 방법[2]과 깊이우선 가지검색법[3]이 있다. G.729 음성 부호화기에서 사용하는 집중 검색 방법은 문턱값을 사용하여 검색 범위를 제한하는 것이고, G.729A에서 사용하는 깊이우선 가지 검색법은 집중 검색 방법보다 더 효과적으로 계산량을 줄이기 위하여 자역적인 최대값을 만족하는 경로에 대해서만 검색을 수행하는 방법이다. 그 외에도 ACELP 코딩 방법에서 대수 코드북의 검색에 대한 계산량을 줄이기 위한 다양한 검색 방법 제안되어져 왔다[4-6].

본 논문에서는 AMR 음성 부호화기의 깊이우선 가지 검색법을 개선하여 보다 적은 계산량으로 대수 코드북을 검색하는 방법을 제안하였다. 이러한 방법은 검색과정 중에서 계산되어지는 정규화된 상관관계 값을 이용하여 검색 트리 중에서 최종적으로 선택될 트리를 미리 예측하여 가능성이 적은 트리는 검색과정에서 제외하는 방법으로서 검색 과정을 위한 추가의 계산이 필요없고, 선택 가능성을 예측하기 위한 비교과정만이 추가적으로 필요하다. 제 2절에서는 AMR 음성 부호화기에 대한 간략한 소개를 하고 3절에서는 대수 코드북 검색방법에 대하여 좀 더 상세한 설명을 하고, 4절에서는 본 논문에서 제안한 계산량을 줄이기 위한 고속 코드북 검색 방법에 대하여 설명한 후, 5절에서는 제안한 방법에 대한 계산량 비교, 실험 결과, 성능 시험에 대하여 기술하고 마지막으로 6절에서 결론을 맺었다.

II. 적응형 다중 비트율 음성 부호화기

AMR 음성부호화기는 8개의 비트율 (12.2, 10.2, 7.95, 7.4, 6.7, 5.9, 5.15, 4.75 kbps)로 동작할 수 있는 다중 비트율을 갖는 부호화 알고리즘으로 구성되어 있다. AMR 음성 부호화기는 8가지의 비트율로 동작되지만, 각각의 부호화 알고리즘은 그림 1과 같이 ACELP 알고리즘을 기본으로 하고 있으며, 각 변수에 대한 양자화 방법들

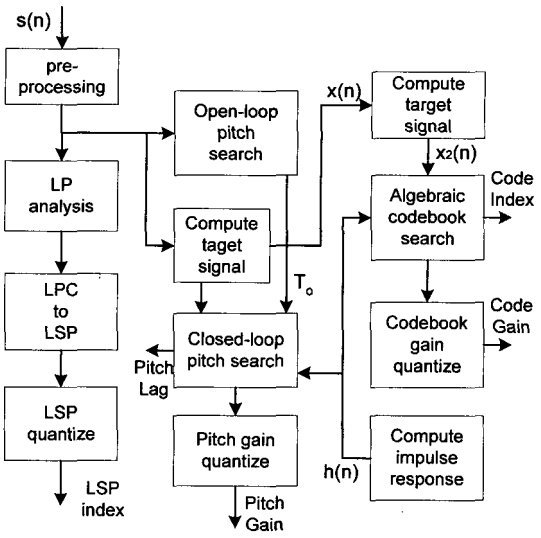

그림 1. 적응형 다중 비트율 부호화기의 블럭도
Fig. 1. AMR encoder block.

을 변화시켜서 비트율을 조정하고 있다.

AMR 음성 부호화기에서는 8 kHz로 샘플링된 음성신호 160 샘플 (20 msec)을 한 프레임으로 하여 10차의 LPC (Linear Predictive Coding) 계수를 구한다. 12.2 kbps 모드에서는 프레임 당 2번의 선형분석을 수행하고, 나머지 모드에서는 프레임 당 한번만 수행하게 된다. LPC 계수는 양자화 왜곡 및 전송오류를 줄이고, 보간 특성이 좋은 LSP (Line Spectral Pair) 계수로 변환한 후 벡터 양자화를 수행한다. 12.2 kbps 모드에서는 2번의 선형분석에서 구해진 2세트의 LSP 계수를 SMQ (Split Matrix Quantization) 방법으로 양자화하고, 나머지 모드에서는 10개의 LSP 계수를 SVQ (Split Vector Quantization) 방법으로 양자화한다.

피치 검색은 계산량을 줄이기 위하여 개루프 (open-loop) 검색 과정을 통해 우선적으로 정수 지연 값을 결정한 후, 이 값을 기준으로 주변 값들에 대해서만 폐루프 (closed-loop) 검색을 수행한다. 개루프 피치 검색은 가중화 된 음성신호상에서 검색이 이루어지며 4.75 kbps와 5.15 kbps 모드일 때만 프레임 당 한번을 수행하고, 나머지 모드의 경우는 프레임 당 두번을 수행한다. 개루프 검색이 끝나면 폐루프 검색을 위하여 임펄스응답 및 목표신호를 계산한다. 폐루프 검색에서는 앞에서 구해진 개루프 지연 값의 주변 값에 대하여 목표신호와 합성된 음성신호와의 평균 자승 오차를 최소화하는 정수값의 지연 값을 결정한다.

대수 코드북은 AMR 음성 부호화기에서 비트할당의 많은 부분을 차지하고 있다. 각 비트율에 따라 부 프레임

당 펄스의 개수를 10개 (12.2 kbps)부터 2개 (4.75 kbps)까지 사용한다. 대수 코드북에서는 부 프레임의 여기 신호를 효율적으로 모델링하기 위하여 부 프레임을 미리 정해진 트랙으로 나누고 각 트랙별로 일정한 갯수의 펄스를 할당하게 된다. 그리고 각 펄스의 크기는 계산량을 줄이기 위하여 미리 ±1로 고정하고 있다. 결과적으로 전송되는 대수 코드북의 정보는 각 트랙 내의 펄스의 위치와 부호이다. 12.2 kbps의 경우는 5개의 트랙으로 나누고 각 트랙마다 두개의 펄스를 정하여 그 위치와 부호 정보를 전송하게 된다. 10.2 kbps 모드에서는 부 프레임을 4개의 트랙으로 나누고 각 트랙마다 2개의 펄스를 사용하여 모델링 하므로 총 8개의 펄스를 사용하게 된다. 7.95 kbps와 7.4 kbps의 경우는 4개의 트랙에 대하여 각 트랙당 하나의 펄스로 모델링한다. 6.7 kbps의 경우는 총 3개의 펄스로 모델링 되며, 나머지 모드들은 2개의 펄스로 모델링 되는데 5.9 kbps의 경우는 2개의 트랙을 사용하고, 5.15 kbps와 4.75 kbps의 경우는 두개의 트랙에 대하여 각각 1개의 펄스를 사용하기는 하지만 각 부 프레임 별로 트랙을 구성하는 펄스의 위치를 변화시켜 주고 있다.

III. 대수 코드북 검색

대수 코드북의 검색은 다음의 식과 같이 입력음성과 합성음성 사이의 평균자승오차를 최소화하는 여기신호의 펄스열을 찾는 과정이다.

$$\varepsilon_k = \| \mathbf{x} - g\ \mathbf{H}\mathbf{c}_k \|^2 \tag{1}$$

여기서 $\mathbf{x}$는 적응코드북의 예측이득이 제거된 목표 신호이고, g는 코드북 이득이고, $\mathbf{H} = h^t h$이고, $\mathbf{c}_k$는 인덱스를 k로 하는 대수 코드 벡터이다. 식 (1)을 최소화하는 것은 다음의 식을 최대화하는 것과 동일하다.

$$T_k = \frac{(C_k)^2}{E_k} = \frac{(\mathbf{H}^t\ \mathbf{x}\mathbf{c}_k)^2}{\mathbf{c}_k^t\ \mathbf{H}^t\ \mathbf{H}\mathbf{c}_k} = \frac{(\mathbf{d}^t\ \mathbf{c}_k)^2}{\mathbf{c}_k^t\ \Phi\ \mathbf{c}_k} \tag{2}$$

여기서 d는 목표신호 $x(n)$과 임펄스 응답 $h(n)$의 상관관계를 나타내는 신호로써 일반적으로 역필터링 된 목표신호로 불리어진다. 그리고 $\Phi = H^t H$는 $h(n)$의 상관관계 매트릭스이다. 대수 코드벡터는 적은 수의 영이 아닌 펄스로 구성되기 때문에 식 (2)에서의 분자항은 다음의 식과 같이 표현된다.

표 1. 12.2 bps 모드의 대수 코드북
Table 1. Algebraic codebook for the 12.2 kbps.

Track	Pulses	Positions
1	i0, i5	0,5,10,15,20,25,30,35
2	i1, i6	1,6,11,16,21,26,31,36
3	i2, i7	2,7,12,17,22,27,32,37
4	i3, i8	3,8,13,18,23,28,33,38
5	i4, i9	4,9,14,19,24,29,34,39

$$C_k = \sum_{i=0}^{N_p-1} s_i d(m_i) \tag{3}$$

여기서 m_i는 i번째 펄스의 위치이고, s_i는 펄스의 부호, 그리고 N_p는 펄스의 개수를 나타낸다. 그리고 식 (2)의 분모항은 다음과 같이 표현될 수 있다.

$$E_k = \sum_{i=0}^{N_p-1} \phi(m_i, m_i) + 2 \sum_{i=0}^{N_p-1} \sum_{j=i+1}^{N_p-2} s_i s_j \phi(m_i, m_j) \tag{4}$$

앞에서 언급된 수식에서 $d(n)$신호와 상관식 $\phi(i,j)$는 검색과정에서의 계산량을 줄이기 위하여 검색 전에 미리 계산된다.

AMR 음성 부호화기의 12.2 kbps 모드의 경우 표 1과 같이 5개의 트랙에 대해 2개의 펄스들이 위치하기 때문에 하나의 부 프레임 당 10개의 펄스를 검색하여야 하는데 10개의 펄스에 대한 위치 및 크기를 결정하기 위해서는 가능한 조합의 수가 총 40C10=847*106로 많은 계산량이 요구되지만, 깊이우선 가지 검색법을 사용하면 4*(4*(8*8))=1024 번의 검색으로 펄스의 위치를 결정하게 되어 검색의 복잡도가 대폭 줄어들게 된다.

IV. 개선된 대수 코드북 검색방법

앞 절에서 설명된 바와 같이 AMR 음성 부호화기의 깊이우선 가지 검색법은 매우 효율적인 검색 방법이긴 하지만, 12.2 kbps 모드의 경우 여전히 전체 인코더의 계산량의 약 40% 정도를 차지할 정도로 많은 계산량을 차지하고 있다. 본 논문에서는 깊이우선 가지 검색법을 개선하여 검색 계산량을 더욱 단축하는 방법을 제안하였다.

제안한 고속 코드북 검색 방법은 2단계의 과정으로 이루어져 있다. 먼저 1단계에서는 최적의 펄스 위치가 존재할 트리를 예측하기 위하여 트리의 일정한 레벨까지 검색을 수행한다. 그림 2에서 볼 수 있듯이 트리의 레벨은 0 - 4까지 존재한다. 검색 결과 펄스가 존재할 확률이 높은 트리 T개를 선택하고, 나머지 트리는 검색의 대상에서

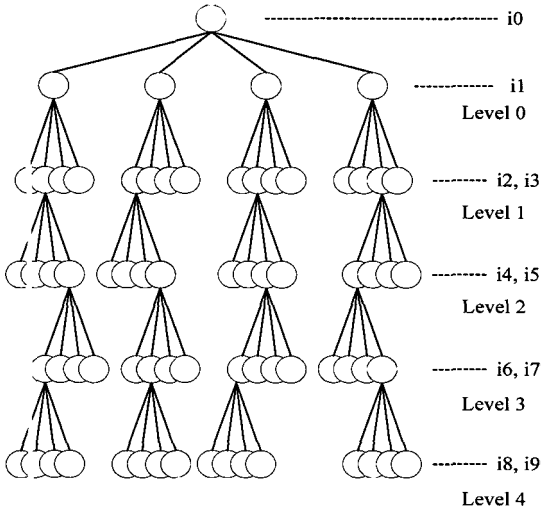

그림 2. 고속 검색 방법에서의 검색 트리와 레벨
Fig. 2. The search tree and levels in the fast search.

제거한다. 이 때 선택의 기준은 앞 절에서의 식 (2)의 계산 값인 T_k를 이용한다. 2단계에서는 선택된 트리에 대하여 깊이우선 가지 검색법과 동일하게 검색을 계속 수행하여 최종적인 펄스의 위치를 결정한다. 위의 방법은 레벨 1의 계산 값으로 트리를 선택하고, 2개의 트리에 대해서만 검색하는 것으로 가정하면 다음과 같은 과정으로 수행된다.

1) 정규화된 타겟신호와 정규화된 장구간 예측 잔여 신호의 합을 b(n)이라 하면, 이 b(n) 신호에 대하여 각 트랙에서 최대값을 하나씩 찾아서 pos_max[]에 저장한다.
2) 위의 최대값 들 중에서 가장 큰 값을 가지고 있는 트랙의 번호를 ipos[0]에 저장한다. 그러면 전체의 최대값을 갖는 위치는 pos_max[ipos[0]]에 저장되어 있게 된다.
3) 전체의 최대값을 갖는 위치는 펄스 i0로 고정하고 그 다음 트랙에서의 최대값을 갖는 위치는 펄스 i1으로 고정한다.
4) 그리고 펄스 i2, i3의 위치를 정하기 위하여 그 다음의 두개의 트랙에서 최대값을 갖는 위치를 계산하는데 8*8의 검색이 필요하게 된다.
5) 3)에서 결정한 i1 펄스의 위치를 나머지 다른 트랙에서의 최대값을 갖는 위치로 바꾸어 4)번을 다시 수행한다. 이 과정은 총 4번 반복된다. (i2, i3)까지 결정하는데 필요한 검색의 횟수는 총 4*(8*8)=256번이 된다.
6) 앞에서 i2, i3 펄스까지만 계산된 결과 값인 T_k를 비교하여 큰 값 2개만을 선택하여 계속 검색할 2개의 트리를 선택한다.
7) 앞에서 선택된 2개의 트리에 대하여 각각 (i4, i5), (i6, i7), (i8, i9)을 결정하기 위한 검색을 순차적으로 수행한다. 이 때 선택한 트리가 2개이므로 필요한 검색의 횟수는 총 2*(3*(8*8))=384번이 된다.

위의 과정에서 필요한 총 검색의 횟수는 256+384=640가 되어 원래의 깊이우선 가지 검색에서의 1024번에 비하여 62.5%의 검색으로 코드북 검색을 수행할 수 있다.

V. 검색의 복잡도 및 실험결과

제안한 고속 코드북 검색 방법에서의 검색횟수는 및 검색할 트리를 예측하기 위하여 1단계에서 검색하는 트리의 레벨과, 1단계의 결과에 의해 선택되는 트리의 갯수에 따라 달라진다. 제안한 검색방법에서의 검색의 횟수는 검색에 포함되는 트리의 갯수를 T라 하고, 제거할 트리를 결정하기 위해 계산하는 레벨의 수를 L이라 할 때, 제거할 트리를 결정하기 위한 검색의 횟수는 4*L*(8*8)이 되고, 선택된 트리에 대해서 검색하는데 필요한 검색 횟수는 T*(4-L)*(8*8)이 되어 총 검색 횟수는 {4*L*(8*8)}+ {T*(4-L)*(8*8)}이 된다. 이렇게 계산 결과를 다음의 표 2에 나타내었다. 여기서 레벨 4일 때는 원래의 방법으로써 1024번의 검색이 이루어지게 된다.

실제 검색에 필요한 계산량은 각각의 검색 단계에서 앞절의 식 (3)과 (4)를 계산해야 되므로 다수의 덧셈과 곱셈이 필요하지만 각각의 검색에 대하여 동일한 계산이 수행되기 때문에 결과적으로 실제 계산량은 검색의 복잡도와 비례한다.

표 3은 고속 코드북 검색 방법을 사용하여 검색된 결과

표 2. 검색의 복잡도
Table 2. Search complexity.

tree	level 0	level 1	level 2	level 3
1	256 (25.0%)	448 (43.8%)	640 (62.5%)	832 (81.3%)
2	512 (50.0%)	640 (62.5%)	768 (75.0%)	896 (87.5%)
3	768 (75.0%)	832 (81.3%)	896 (87.5%)	960 (93.8%)

표 3. 고속 코드북 검색방법의 hit ratio
Table 3. Hit ratio of the fast codebook search method.

tree	level 0	level 1	level 2	level 3
1	27.59%	34.18%	40.65%	48.00%
2	53.00%	59.82%	67.24%	77.71%
3	75.76%	81.71%	88.12%	91.41%

표 4. 12.2 kbps 모드의 계산량
Table 4. Complexity for the 12.2 kbps mode.

Function Block	MIPS
Pre-processing	0.39
Compute LPC	1.10
LSP quantization	4.22
Open-loop pitch search	1.79
Closed-loop pitch search	2.90
Codebook search	**8.64**
Gain quantization	0.31
Memory update, etc.	1.52
Total	**20.87**

표 5. 코드북 검색 방법의 계산량 비교
Table 5. Comparison of computational complexity.

Function Block	MIPS (original)	MIPS (fast)
Codebook block	8.64	6.62
Search only	5.81	3.57

표 6. 고속 코드북 검색방법의 성능
Table 6. Performance of the fast search method.

	tree	level 0	level 1	level 2	level 3
SNR	1	16.25	16.50	16.63	17.26
	2	16.72	17.31	17.43	18.01
	3	17.74	17.98	17.99	18.29
	4	18.46	18.46	18.46	18.46
seg SNR	1	10.64	10.88	10.97	11.39
	2	11.24	11.46	11.65	11.96
	3	11.86	11.97	12.20	12.29
	4	12.53	12.53	12.53	12.53

가 원래의 깊이우선 가지 검색법을 사용하여 검색한 결과와 어느 정도 일치하는가를 알아보기 위한 실험 결과이다. 표 2와 3에서 보면 레벨 1의 결과로 트리를 선택하고, 두개의 트리에 대해서만 검색을 하는 경우 제안한 검색방법은 검색 복잡도가 약 40% 감소하였고, 검색결과는 약 60% 정도가 원래의 방법으로 검색한 결과와 일치함을 알 수 있다.

표 4는 AMR 음성 부호화기의 12.2 kbps 모드를 실시간 구현 했을 때 인코더 블록의 계산량을 모듈 별로 나타낸 것이다. 이중에서 코드북 검색 부분은 전체 계산량의 약 40% 정도를 차지하고 있음을 알 수 있다. 그리고 제안한 방법을 16비트 고정소수점형 디지털 신호처리 프로세서(DSP: Digital Signal Processor)인 TeakLite DSP를 사용하여 구현 하였을 때 계산량 감소 효과는 다음의 표 5 를 통해 보였다. 표 5는 코드북 검색 부분에 대해서만 기존의 깊이우선 가지 검색법과 고속 검색 방법으로 구현 했을 때의 계산량을 보인 것이다. 표 5에서 코드북 블록의 계산량은 검색과정과 검색전의 상관관계의 계산 및 검색 후의 코드 생성 부분의 계산량을 전부 포함한 것이고, 검색 블록은 검색부분의 계산량 만을 나타낸 것이다. 검색 부분에서의 계산량 감소는 약 40%로 앞에서 보인 검색율의 감소량과 거의 비슷함을 알 수 있다.

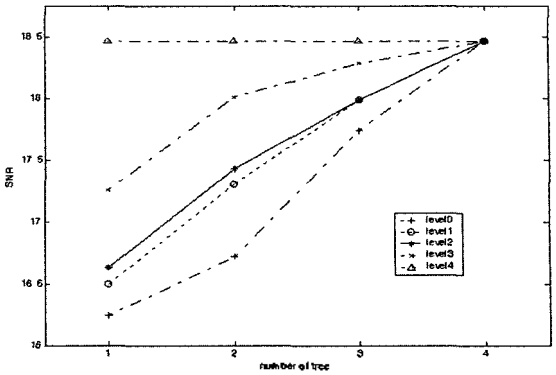

그림 3. 고속 코드북 검색방법의 성능
Fig. 3. Performance of the fast search method.

제안한 방법에 대한 성능평가는 압력음성과 인코더에서 코딩 파라미터에 의해 합성된 음성간의 신호대 잡음비와 segSNR을 측정하여 시험하였다. 표 6의 시험 결과를 보면 레벨 1과 2개의 트리에 대해 검색하였을 경우 원래의 방법 (레벨 4)에 비하여 약 1.1 dB의 음질 저하가 발생하였다. 레벨 4일 때의 신호대 잡음비는 18.46이고, segSNR는 12.53이었다.

그림 3에는 신호대 잡음비 비교의 결과를 표시하였다. 그림에서 보면 레벨 1의 결과를 사용하는 것이 계산량과 성능 측면에서 효과적임을 알 수 있다.

VI. 결 론

본 논문에서는 AMR 음성 부호화기의 대수 코드북 검색 방법인 깊이우선 가지 검색법을 개선하여 보다 적은 계산량으로 대수 코드북을 검색하는 방법을 제안하고, 제안한 방법을 적용하여 AMR 음성 부호화기를 TeakLite DSP를 이용하여 구현하고, 제안한 방법의 계산량 및 성능을 확인하였다. 제안한 고속 검색방법은 최종적으로 선택될 가능성이 높은 트리를 선택하는 단계와, 선택된 트리만을 검색하여 최적의 펄스위치를 정하는 2단계의 과정으로 이루어진다. 원래의 깊이우선 가지 검색법에 비해 검색 복잡도는 약 40% 감소하였고, 약 1.1 dB의 음질 저하가 발생하였다. 그리고 이러한 성능은 1단계의 검색

레넬과, 2단계에서 선택된 트리의 수에 따라 검색의 복잡도 변화되므로 사용자가 적절한 수준을 선택할 수 있다. 앞으로 더 적은 계산량으로 보다 나은 음질을 위하여 추가적인 연구가 필요하다고 생각된다.

참 고 문 헌

1. 3GPP (3rd Generation Partnership Project) TS 26.090, "AMR speech codec; transcoding functions," December 1999.
2. R. Salami, C. Laflamme, J.-P. Adoul, A. Kataoka, S. Hayashi, T. Moriya, C. Lamblin, D. Massaloux, S. Proust, P. Kroon, and Y. Shoham, "Design and description of CS-ACELP: A toll quality 8 kb/s speech coder," *IEEE Trans. Speech and Audio Processing*, 6, 116-130, 1998.
3. R. Salami, C. Laflamme, B. Bessette, and J.-P. Adoul, "ITU-T G.729 annex A: reduced complexity 8 kb/s CS-ACELP codec for digital simultaneous voice and data," *IEEE communications Magazine*, 56-63, Sep. 1997.
4. Nam Kyun Ha, "A fast search method of algebraic codebook by reordering search sequence," *in Proc. ICASSP*, 1, 21-24, 1999.
5. Hochong Park, "Efficient codebook search method of EVRC speech codec," *IEEE Signal Processing Letters* , 7 (1), 1-2, Jan. 2000.
6. Fu-Kun Chen, and Jar-Ferr Yang, "Maximum-take-precedence ACELP: A low complexity search method," in *Proc. ICASSP*, 2, 693-696, 2001.
7. 변경진, 최민석, 한민수, 김경수, IMT-2000 비동기식 단말기용 ASIC을 위한 적응형 다중 비트율 (AMR) 보코더의 구현, 한국음향학회지, 20 (1), 56-61, 2001.

저자 약력

● 변 경 진 (Kyung-Jin Byun)

1987년 2월: 국민대학교 전자공학과 (공학사)
2000년 2월: 한국정보통신대학원대학교 공학부 (공학석사)
2000년 3월~현재: 한국정보통신대학원대학교 공학부 박사과정 재학중
1987년 3월~현재: 한국전자통신연구원 이동통신IC 설계팀 책임연구원
※ 주관심분야: 음성 코딩 및 분석, DSP 설계 및 응용

● 정 희 범 (Hee-Bum Jung)

1981년 2월: 서강 대학교 전자공학과 (공학사)
1983년 2월: 한국과학기술원 전기및전자공학과 (공학석사)
1992년 10월: Columbia University EE (Ph.D)
1983년 3월~현재: 한국전자통신연구원 이동통신IC 설계팀 책임연구원
※ 주관심분야: 이동/무선통신용 부품기술

● 한 민 수 (Min-Soo Hahn)

1979년 2월: 서울대학교 전기공학과 (공학사)
1981년 2월: 서울대학교 대학원 (공학석사)
1989년 12월: University of Florida (공학박사)
1982년 4월~1985년 8월: 한국표준과학연구원 연구원
1990년 2월~1997년 12월: 한국전자통신연구원 책임연구원
1998년 1월~현재: 한국정보통신대학원대학교 공학부 부교수
※ 주관심분야: 디지털 신호 처리, 음성 분석, 합성, 인식 및 코딩, 적응 신호 처리, 3-D 음향 등