2000년도 한국음향학회 학술발표대회 논문집 제19권 제1(s)호

G.723.1 음성 부호화기의 LSF 계수 양자화를 위한 고속화 알고리즘 연구

손창용*, 성호상**, 강상원*, 성유나***
*한양대학교 전자컴퓨터공학부, **한국전자통신연구원, ***C&S Technology

A study on a fast algorithm for the LSP coefficient quantization of G.723.1 speech codec

Chang-yong Son*, Ho-sang Sung**, Sang-won Kang*, Yu-na Sung***
*School of Electrical Eng. and Computer Science, Hanyang University, **ETRI, ***C&S Technology
E-mail) swkang@selab.hanyang.ac.kr

요 약

본 논문에서는 멀티미디어 서비스들 중에서 음성 또는 오디오 신호를 저속으로 압축할 때 사용되는 G.723.1 부호화기의 line spectral frequency(LSF) 계수 양자화 방식을 고속으로 처리하는 알고리즘을 제안하였다. 제안된 고속탐색 방법은 LSF 계수의 순서성질을 이용하여 코드북의 탐색 범위를 줄임으로써 계산량을 크게 감소시킨다.

제안된 고속탐색 방법을 predictive split VQ(PSVQ) 구조를 갖는 G.723.1 에 적용한 결과 spectral distortion(SD) 성능 감쇄 및 추가적인 메모리 증가없이 최적 코드벡터를 찾기위한 코드북 탐색 과정에서 코드북의 평균 탐색 범위가 20.1% 감소했으며, 이는 additions, subtractions, multiplies 및 comparisons 수가 각각 19.1%, 20.1%, 19.4% 및 12.2% 감소하는 결과를 얻었다.

1. 서 론

저속 음성 부호화기에서 고 음질의 음성 부호화를 위해서는 음성신호의 단구간 상관도를 나타내는 linear predictive coding(LPC) 계수를 효율적으로 양자화하는 것이 매우 중요하다. LPC 필터의 최적 선형 예측 계수 값은 입력 음성 신호를 프레임 단위로 나누어 각 프레임 별로 예측 오차의 에너지를 최소화 시키는 개념으로 구해진다. LPC 필터는 일반적으로 10 차 all-pole 필터이며, 이때 사용되는 10 개의 선형 예측 계수들의 양자화를 위하여 많은 비트가 할당된다. LPC 필터의 계수를 직접 양자화 할 경우, 필터의 특성이 계수의 양자화 오차에 매우 민감하고 계수 양자화 후의 LPC 필터의 안정성이 보장되지 않는 문제점이 있다. 따라서 LPC 계수는 양자화 특성이 좋고 합성필터의 안정도를 검사하기 쉬운 LSF 계수로 변환된 후 양자화 된다. LSF 계수는 식(1)과 같은 순서성질을 만족해야 합성필터의 안정성이 보장된다.

$$0 < \omega_1 < \omega_2 < \cdots < \omega_p < \pi \qquad (1)$$

LSF 계수의 효율적인 양자화를 위해 최근에 개발된 표준 음성 압축기들은 벡터 양자화 방법을 많이 사용한다. 벡터 양자화에서 전체 벡터를 한꺼번에 양자화 하는 것은 벡터 테이블의 크기가 너무 커지고 검색 시간이 많이 소요되므로 사용 불가능하다. 이를 해결하기 위하여 전체 벡터를 여러 개의 부벡터로 나누어 각각을 독립적으로 벡터 양자화 하는 방법이 개발되었는 데, 이를 split vector quantization (SVQ)라 한다. SVQ 에 예측기를 추가한 방식인 predictive split VQ(PSVQ)는 효율적인 양자화를 위하여 LSF 계수의 프레임간 상관관계를 이용한다. 즉, 현재 프레임의 LSF 를 직접 양자화하지 않고 과거 프레임의 LSF 값 정보로부터 현재 프레임의 LSF 를 예측하고 예측 오차를 양자화 하는 것이다. LSF 값은 음성 신호의 주파수 특성과 밀접한 관계가 있으며, 따라서 시간적으로 예측이 가능하고 상당히 큰 예측 이득을 얻을 수 있다.

본 논문에서는 PSVQ 구조를 갖는 G.723.1 음성 코덱의 LSF 계수 양자화를 위한 고속 탐색 방법을 제안한다. 제안된 방식은 성능 감쇄없이 코드북 탐색 과정의 복잡도를 효과적으로 감소시킨다.

2. G.723.1 음성 부호화기에 사용된 PSVQ

SVQ 에서 LSF 벡터는 전체 벡터를 여러 개의 부벡터로 나누어 각각을 독립적으로 벡터 양자화 하므로 벡터 테이블의 크기와 검색 시간을 줄인다. 보다 많은 부벡터로 나누면 벡터 테이블의 크기가 줄어들어 메모리을 절약 할수 있고 검색 시간을 줄일수 있는 장점이 있으나, 각 부벡터를 독립적으로 양자화 하므로 부벡터 사이의 상관 관계를 충분히 이용하지 못하고 전체 벡터에 대한 최적화를 하지 못하는 단점이 있다. 따라서 이러한 trade-off 를 고려하여 부벡터의 수를 결정해야 한다.

PSVQ 는 효율적인 양자화를 위해 SVQ 에 predictive vector quantization(PVQ) 기술을 추가한 방식이다. PVQ 는 LSF 계수의 프레임간 상관 관계를 이용하여 현재 프레임의 LSF 를 직접 양자화하지 않고 과거 프레임의 LSF 값 정보로부터 현재 프레임의 LSF 를 예측하고 예측 오차를 양자화하는 것이다. 프레임간의 상관 관계가 큰 음성 신호에서 PVQ 를 사용하여 양자화 할 경우 상당히 큰 예측 이득을 얻을 수 있다. 현재 주로 사용되는 선형 예측 방법은 auto regressive(AR) 필터와 moving average(MA) 필터를 사용하는 두가지 방법이 있다. AR 필터는 예측 성능이 우수한 반면 계수 전달 오류의 영향이 수신측에서 프레임의 진행에 따라 계속 전파되는 단점이 있다. MA 필터는 AR 필터에 비하여 예측 성능은 떨어지지만 전달 오류의 영향이 시간적으로 제한되는 장점이 있다.

G.723.1 에서는 그림 1 과 같이 1 차 AR 예측기를 사용하는 PVQ 와 3 개의 부벡터로 나누어 각각 독립적으로 양자화 하는 SVQ 를 사용하는 PSVQ 방식을 사용한다.

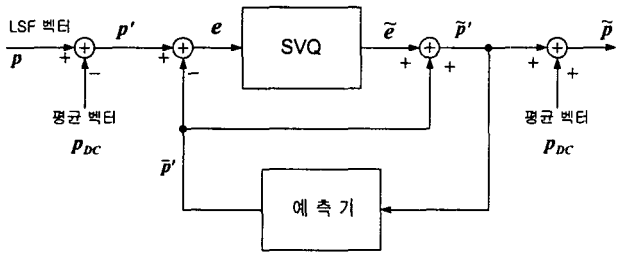

그림 1. Predictive Split VQ 구조

그림 1 에서 AR 예측기의 차수는 1 차 이며, $\boldsymbol{p}$ 는 LSF 계수벡터이다. $\boldsymbol{p}$ 에서 LSF 계수값의 DC 성분인 평균벡터 $\boldsymbol{p}_{DC}$ 를 빼면 $\boldsymbol{p}'$ 이 되며, 이전 프레임의 복호화된 LSF 벡터 $\tilde{\boldsymbol{p}}'$ 과 1 차 예측기 계수 b(=12/32)를 이용하여, 예측 벡터 $\bar{\boldsymbol{p}}'$ 과 LSF 에러벡터인 $\boldsymbol{e}$ 를 다음과 같이 구한다.

$$\bar{p}'_{(n)} = b\tilde{p}'_{(n-1)} \tag{2}$$

$$e = p' - \bar{p}' \tag{3}$$

여기서 n 은 현재 프레임을 의미한다. LSF 에러 벡터인 $\boldsymbol{e}$ 는 각각 3, 3, 4 차원을 가지는 3 개의 부벡터들로 나누어지며, 각 부벡터는 8 비트로 벡터 양자화 된다.

이때 각 부벡터들에 대한 최적 코드벡터 값은 다음 에러 criterion $E_{l,m}$ 을 최소화하는 것을 선택한다.

$$E_{l,m} = (\boldsymbol{p}_m - \tilde{\boldsymbol{p}}_{l,m})^T \boldsymbol{W}_m (\boldsymbol{p}_m - \tilde{\boldsymbol{p}}_{l,m}), \quad \begin{matrix} 0 \le m \le 2 \\ 1 \le l \le 256 \end{matrix} \tag{4}$$

여기서 $\boldsymbol{W}_m$ 은 m 번째 부벡터를 위한 weighting matrix 이며, 양자화되지 않은 LSF 벡터인 $\boldsymbol{p}$ 로 부터 구해진다. 그리고 양자화된 LSF 벡터 $\tilde{\boldsymbol{p}}_{l,m}$ 은 다음 식으로 나타낼 수 있다.

$$\tilde{\boldsymbol{p}}_{l,m} = \bar{\boldsymbol{p}}'_m + \boldsymbol{p}_{DC} + \tilde{\boldsymbol{e}}_{l,m}, \quad \begin{matrix} 0 \le m \le 2 \\ 1 \le l \le 256 \end{matrix} \tag{5}$$

(5)식을 (4)식에 대입하면 다음과 같이 나타낼 수 있다.

$$E_{l,m} = (\mathbf{e}_m - \tilde{\mathbf{e}}_{l,m})^T \mathbf{W}_m (\mathbf{e}_m - \tilde{\mathbf{e}}_{l,m}), \quad \begin{matrix} 0 \le m \le 2 \\ 1 \le l \le 256 \end{matrix} \tag{6}$$

식 (6)에 의하면 $E_{l,m}$ 은 $\boldsymbol{e}_m$ 과 $\tilde{\boldsymbol{e}}_{l,m}$ 에 대한 식으로 표현되므로 실제 256 개의 크기를 갖는 코드북에 대해 $E_{l,m}$ 을 계산할 때 계산량을 많이 줄일 수 있다. 여기서 $\boldsymbol{e}_m$ 은 코드북 탐색을 위한 target 벡터이며 $\tilde{\boldsymbol{e}}_{l,m}$ 은 m 번째 부 벡터의 l 번째 에러 코드벡터에 해당한다. 그러므로 이 값들을 이용해서 $E_{l,m}$ 을 최소화하는 코드북 인덱스 l 을 채널을 통해 전송한다.

3. PSVQ 을 위한 고속탐색 방법

본 논문에서는 PSVQ 의 효율적인 양자화를 위한 방법으로 최적 코드벡터를 찾기위한 코드북 탐색 과정을 수행 하기전에 LSF 계수의 순서성질을 이용하여 탐색하는 코드벡터를 제한하는 고속 탐색 방법을 제안한다. 그러나 그림 1 과 같은 PSVQ 구조에서는 LSF 계수를 직접 양자화 하지않고 평균 값을 제거한 LSF 계수에 예측 값을 뺀 값을 양자화하기 때문에 양자화하는 target 벡터는 순서성질을 갖지 않는다. 따라서 제안된 고속 탐색 방법을 그림 1 과 같은 PSVQ 를 이용한 LSF 양자화기에 적용하기 위해서는 양자화 대상인 target 벡터와 해당 코드벡터를 순서성질을 갖도록 변환해주는 과정과 탐색하는 코드벡터의 범위 결정을 손쉽게 하기위한 코드벡터들의 재정렬 과정이 필요하다. 그래서 본 장에서는 순서 성질 복원 및 코드북 재정렬 과정을 묘사하고, 재정렬된 코드북을 이용해서 양자화를 수행하는 과정을 언급한다.

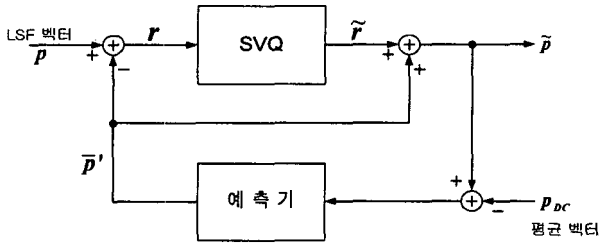

그림 2. 수정된 PSVQ 구조

A. 순서성질 복원 및 코드북 재정렬 과정

그림 1 과 식(6)에서, target 벡터 $\boldsymbol{e}_m$ 과 해당 코드북은 에러 벡터에 해당하므로 순서성질을 갖지 않는다. 그래서 그림 1 과 식(6)을 그림 2 와 식(7)과 같이 바꾸어서 순서성질을 갖는 target 벡터 형태로 구성한다.

$$E_{l,m}=(r_m-c_{l,m})^T W_m(r_m-c_{l,m}) \quad , \quad m=1,2,3 \ , 1\le l\le L_m \qquad (7)$$

그림 2 와 식(7)에서 r_m 은 p_{DC} 값을 제거하지 않은 $(p_m-\overline{p}'_m)$값이고 $\overline{p}'_m$ 은 평균 값을 제거한 LSF 계수로 부터 1 차 AR 예측기에 의해 예측된 값으로 LSF 벡터 p_m 의 평균 변위에 비해 매우 작은 값이다. 따라서 r_m 값은 그림 3 과 같이 순서성질을 유지하게 된다. 실제로 20,000 프레임(20ms/frame)의 음성 샘플을 실험한 결과 벡터 r_m 의 10 개 요소값은 순서 성질을 유지하였다.

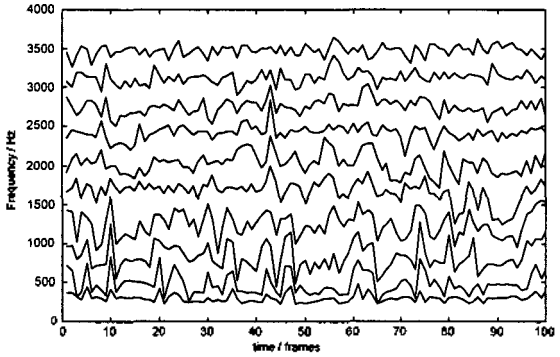

그림 3. LSF 벡터 p_m 과 그것의 예측 벡터인 $\overline{p}'_m$ 간의 에러 벡터인 r_m 의 trajectories

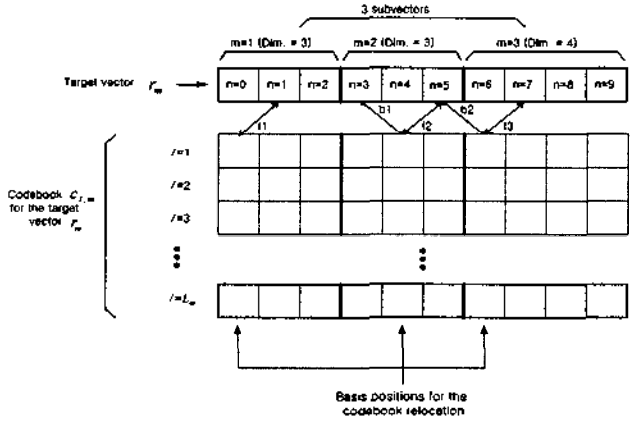

그림 4. 코드북 탐색 범위 결정을 위한 순방향 및 역방향 비교

또한 에러 코드벡터 $\tilde{e}_{l,m}$에 해당하는 기존 코드북에 순서 성질을 갖는 p_{DC} 값을 더한 $c_{l,m}$ 은 10 개의 p_{DC} 값들의 변위가 $\tilde{e}_{l,m}$에 비해 매우 크므로 순서 성질을 갖게된다. 최적 코드벡터의 고속 탐색을 위해 에러 코드벡터로 이루워진 기존 코드북에 p_{DC} 값을 더한 새로운 코드북은 다음과 같이 코드벡터의 재정렬 과정을 수행한다.

Step 1) 많은 training input data 를 사용하여 3 개 코드북에 대하여 실험적으로 최적인 정렬 위치(N_m)를 결정한다.

본 논문에서는, 그림 4 와 같이 첫번째, 세번째 코드북은 첫번째 열, 두번째 코드북은 두번째 열을 정렬을 위한 기준열로 정하였다. 기준열은 실험적으로 결정되었다.

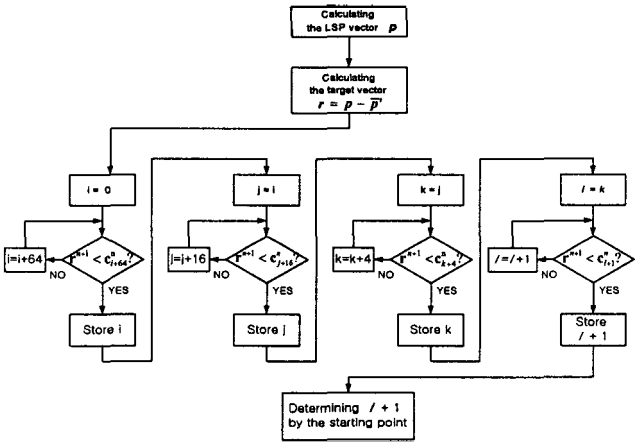

그림 5(a). 순방향 비교로 코드북 탐색의 시작점을 구하는 방법

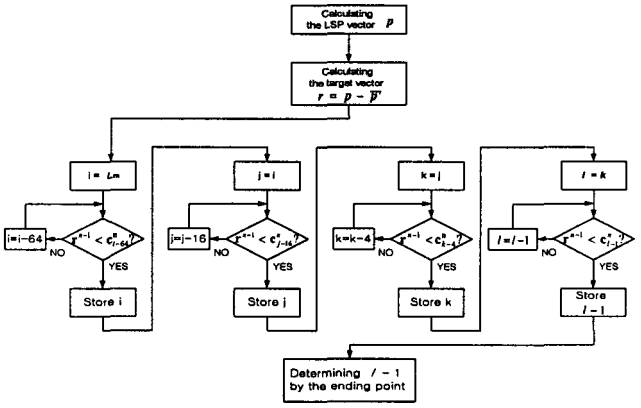

그림 5(b). 역방향 비교로 코드북 탐색의 끝점을 구하는 방법

Step 2) 각 코드북의 L_m 개 코드벡터의 정렬 순서를 정해진 기준열(N_m)의 요소값에 따라 내림차순으로 정렬시킨다.

Step 3) 기존 코드북을 내림 차순으로 정렬된 새로운 코드북으로 대체한다.

B. 고속 탐색 방법을 이용한 양자화 수행 과정

순서 성질을 갖는 target 벡터 r_m 과 최적 코드벡터 $c_{l,m}$ 은 각 요소들 간에 다음과 같은 순방향 비교식(8)과 역방향 비교식(9)을 만족한다.

$$r_m^{n+1} > c_{l,m}^n \ , \quad 1\le l\le L_m, m=1,2,3, 0\le n\le 8 \qquad (8)$$

$$r_m^{n-1} < c_{l,m}^n \ , \quad 1\le l\le L_m, m=1,2,3, 1\le n\le 9 \qquad (9)$$

본 알고리즘에서는 계산량이 많은 error criterion $E_{l,m}$ 의 계산을 수행하여 최적 코드벡터를 찾는 코드북 탐색 과정 전에 L_m 개

대상 코드벡터 중에서 식(8), (9)를 위배하는 코드벡터를 제외하므로써 $E_{l,m}$ 의 계산 횟수를 크게 줄인다. 그림 4 에서 f1, f2, f3 및 b1, b2 는 결정된 기준열 $N_m(N_1=0, N_2=4, N_3=6)$ 에 따라 각각 순방향 및 역방향 비교에 사용되는 코드벡터 요소 값과 target 벡터값을 나타낸 것이다.

정렬된 코드북 탐색과정은 다음과 같이 두단계로 이루워진다.

Step 1) 그림 5 의 순방향과 역방향 비교 알고리즘에 의해, 코드북의 기준열 N_m 의 요소값과 target vector 의 기준열 전후 요소값들을 비교하여 순서 성질을 만족하는 코드벡터의 시작점과 끝점을 구한다.

Step 2) 구해진 시작점과 끝점 사이의 코드북에 대해서만 $E_{l,m}$ 값을 구하며, $E_{l,m}$ 을 최소화시키는 코드북 index l 을 채널을 통해서 전송한다.

4. 실험 결과

개발된 고속탐색 방법의 성능 실험은 ITU-T 의 표준 음성 부호화기인 G.723.1 을 통해 수행되었다. Training 음성 데이타는 23,500 프레임(20ms/frame)을 사용했으며, 이는 잡음과 비잡음에서의 음성을 포함 한다. 성능 평가를 위해 평균 탐색 범위와 사용된 operation 수 측면에서 기존 방식과의 비교 실험과 SD 성능 평가를 수행하였다.

표 1 는 고속 탐색 방법을 사용할 때 코드북의 탐색 코드벡터의 감소율을 보여준다.

본 알고리즘에서 각 코드북의 시작점과 끝점을 구하기 위한 계산량은 부벡터당 단지 평균 20 개의 비교문이 추가적으로 사용된다. 그리고 탐색하는 코드북의 코드벡터 수가 N 일 때 첫 번째와 두 번째 코드북에서 최적 코드벡터를 찾는데 요구되는 operation 수는 addition 6N, multiply 8N, subtraction N, comparator N 이 필요하고 세 번째 코드북에서는 addition 8N, multiply 10N, subtraction N, comparator N 이 필요하다. 표 2 는 고속 탐색 방법을 사용할 때 operation 수의 감소율을 보여준다.

기존 방식과 제안된 방식에서 구해진 코드북 인덱스를 비교한 결과 23,500 프레임 중 0.4%인 94 개 프레임의 인덱스가 다르게 발생했다. 이러한 94 개의 인덱스의 코드벡터는 원래 인덱스의 코드벡터와 상당히 비슷한 값이므로 SD 값의 차이가 거의 없다. 표 3 는 기존방식과 제안된 방식에서 구한 평균 SD 값이 차이가 없음을 보여준다.

표 1. 코드북의 평균 탐색 코드벡터 수 비교

	코드북 크기	코드북의 평균 탐색 범위 (=끝점 평균 - 시작점 평균)	감소율 (%)
1st 부벡터	256	177 (= 177 – 0)	30.9
2nd 부벡터	256	209 (= 224 – 15)	18.4
3rd 부벡터	256	228 (= 242 – 14)	10.9
Per frame	768	614	20.1

표 2. 프레임당 평균 operations 수 비교

	Additions	Subtractions	Multiplies	Comparisons
기존 방식	5120	768	6656	768
제안된 방식	4140	614	5368	674
평균 감소율 (%)	19.1%	20.1%	19.4%	12.2%

표 3. 기존 방식과 제안된 방식에서 결정된 코드북의 인덱스 비교

	Number of frames(percent)	Average SD difference(dB) (proposed – conventional)
Frame with diff. index	94 (0.4%)	0.12
Frame with same index	23,406 (99.6%)	0
Total	23,500 (100%)	0.00048

5. 결론

본 논문에서는 고속 탐색 방법을 사용하여 LSF 계수의 효율적인 양자화 방법을 연구하였다. 제안된 고속 탐색 방법은 LSF 벡터의 순서 성질을 이용하여 탐색하는 대상 코드벡터의 범위를 제한하는 방법이며, PSVQ 구조를 갖는 G.723.1 음성 코덱에 적용하였다. 성능 평가는 탐색하는 대상 코드북의 크기와 요구되는 operation 수 관점에서 복잡도를 비교 분석 했으며, SD 성능 감쇄 여부를 평가하였다. 실험 결과, 추가적인 메모리 증가 및 SD 성능 감쇄가 전혀 없이 프레임당 탐색하는 대상 코드북의 크기가 평균 20.1% 감소하였으며, 이는 additions, subtractions, multiplies 및 comparisons 수가 각각 19.1%, 20.1%, 19.4% 및 12.2% 감소하는 결과를 얻었다.

[참 고 문 헌]

[1] ITU-T Recommendation G. 723. 1, "Dual Rate Speech Coder for Multimedia Communications Transmitting at 5.3 and 6.3 kbit/s," March 1996.

[2] F. Itakura, "Line Spectrum Representation of Linear Predictive Coefficients, J.Acoust.Soc.Amer.," vol.57, pp.s35(A), 1975.

[3] Kuldip K.Paliwal and Bishnu S.Atal,"Efficient Vector Quantization of LPC Parameter at 24bits/frame,"IEEE Trans. Speech,audio processing, vol.1, no.1, pp.3-14, Jan.1993.

[4] F.K.Soong and B.H.Juang, "Line Spectrum Pair(LSP) and Speech Data Compression," proc. ICASSP, pp.1.10.1-10.4, 1984.

[5] N.Sugamura and N.Farvarin, "Quantizer design in LSP Speech analysis-synthesis," IEEE J.Seleet.Areas in Commun.,vol.6, pp.432-440, Feb.1988.