제15회 음성통신 및 신호처리 워크샵(KSCSP 98 15권1호)

벡터 차의 절대값 합을 이용한 고속 벡터 부호화 알고리즘

A Fast VQ Encoding Algorithm Using Sum of Absolute Difference of Vectors

백성준, 강상기[0], 성굉모
SeongJoon Baek, SangKi Kang, and Koeng-Mo Sung

서울대학교 공과대학 전기공학부

School of Electrical Engineering, Seoul National University, Seoul 151-742, KOREA
TEL. (02) 880-7263, FAX. (02) 882-4657, E-mail: bsj@acoustics.snu.ac.kr

본 연구는 현대전자의 지원으로 이루워졌습니다.

요약

본 논문에서는 벡터양자화기의 부호화 단계에서 계산량을 줄이는 새로운 알고리즘을 제안한다. 벡터양자화기의 부호화는 주어진 입력벡터에 가장 가까운 코드워드를 찾는 것인데 모든 코드워드와 거리계산을 필요로 하기 때문에 많은 계산량이 소요되므로 효율적인 알고리즘이 필요하다. 본 논문에서는 입력벡터와 코드워드와의 유클리디안 거리계산 대신에 벡터 차의 절대값 합을 이용하여 주어진 입력벡터에 최단거리의 코드워드가 될 수 없는 코드워드를 제외함으로써 유클리디안 거리계산을 최소화하여 계산량을 줄이는 알고리즘을 제안한다. 특히 제안된 방법을 고정 소수점 연산을 이용한 DSP 칩에 효과적이며 이는 실험 결과를 통하여 확증할 수 있다.

I. 서 론

벡터양자화(이하 VQ)는 데이터 압축에 매우 효율적인 방법이다[1][2]. VQ는 벡터의 차수가 증가함에 따라 rate-distortion 한계에 근사하는 코딩 성능을 지닌다. 하지만 벡터의 차수가 증가함에 따라 부호화의 복잡성도 지수함수적으로 증가하게 되는데 이것이 VQ의 사용을 크게 저해한다. VQ 부호화는 k차원 유클리드 공간 R^k에서 해당 차원의 유한 부분집합 Y로의 사상으로 볼 수 있다. 다시 말해서 $Q:R^k \rightarrow Y$인데 여기서 $Y=\{y_i | i=1,2,\ldots,N\}$는 코드북이며, N은 그 코드북의 크기이다. 입력벡터와 출력벡터 사이의 유사도를 측정하는 방법으로 $d^2(x,y_i)=\sum_{n=1}^{k}(x_n-y_{in})^2$로 정의되는 유클리드 거리가 가장 널리 쓰이고 있다.

VQ 부호화 단계에서는 코드북 Y중에서 입력 x에 가장 가까운 코드워드를 찾아야 하는데, 그것은 전체 탐색을 이용할 경우 N번의 거리계산 $d(x,y_i)$을 필요로 한다. 따라서 코드북의 크기가 커질수록 VQ 부호화 단계에서 계산량을 줄이는 문제는 많은 분야에서 VQ가 응용되기 위해서 매우 중요한 과제가 된다. 반면 복호화 단계에서는 전송된 코드북의 인덱스를 이용해서 코드북 테이블의 벡터값으로 대치시키기만 하면 되므로 계산량이 아무런 문제가 되지 않는다.

그런 이유로 VQ 부호화 단계에서 계산량을 줄이기 위해 많은 연구들이 행해졌는데 이러한 연구는 크게 두 개의 그룹으로 나눌 수 있다. 첫 번째 그룹은 정확한 해를 구하는 대신에 근사해를 구하는는 것인데 대신 부호화에 용이한 구조를 채택한다. 일반적으로 이러한 방법에서는 트리구조를 이용해서 코드북을 구성하는데 TSVQ, K-d tree 등의 방법이 그 예이다[3][4]. 두 번째 그룹은 정확한 해를 구하면서 계산량을 줄이는 방법으로 Bei and Gray의 부분 거리 탐색(partial distance search, PDS) 알고리즘, Orchard의 fast nearest neighbor search(이하 FNNS) 알고리즘, 벡터의 projection을 이용하는 알고리즘이 있다[5][6][7]. 벡터의

projection을 이용하는 알고리즘으로는 벡터의 평균을 이용하는 equal-average nearest neighbor search (ENNS), ENNS를 확장하여 평균이외에 분산에 대한 거리 관계식을 이용하여 계산량을 줄이는 방법 등이 이후에 제시되었다[8][9][10].

본 논문에서 제안하는 방법은 후자의 그룹에 속하는 방법으로서 입력벡터와 코드워드 차이 절대값의 합을 이용해 필요한 유클리드 거리(이하 거리로 약칭) 계산을 줄이는 방법이다. 이를 위해 본 논문에서는 II 장에서 거리와 절대값의 합 사이의 관계식을 유도하며 새로운 알고리즘을 제안한다. III장에서는 제안된 알고리즘의 성능을 비교하기 위한 실험 결과와 이에 대한 분석 결과를 보여 주었으며, IV장에서는 결론 및 향후 연구과제에 대해서 언급하였다.

II. 제안된 알고리즘

이 장에서는 새로운 알고리즘을 제시하기 전에 먼저 알고리즘에 사용될 두 벡터사이의 거리와 그 차의 절대값의 합 사이에 성립하는 부등식을 유도하고, 이후에 본 알고리즘을 선명하기로 한다.

정리 : $x=(x_1, x_2, \ldots, x_k)$는 주어진 입력 벡터이고 거리계산을 하고자 하는 코드워드를 $y=(y_1, y_2, \ldots, y_k)$라고 하며, 거리는 유클리드 거리로 정의되었다고 하면 다음 식(1)이 성립한다.

$$d^2(x, y) \geq \left(\sum_{i=1}^{k}|x_i - y_i|\right)^2/k \qquad (1)$$

(증명) 위의 식에서 $\sum_{i=1}^{k}|x_i - y_i| = \sum_{i=1}^{k} p_i(x_i - y_i)$ 와 같이 표현할 수 있으며, 이때 p_i는 $x_i \geq y_i$일 때는 값이 1이고, 그렇지 않을 때는 값이 -1로 정의되는 벡터이다. 증명에 앞서 다음과 같은 새로운 변수를 도입하기로 한다.

$$\alpha_i = p_i x_i - p_i y_i, \quad \beta = \sum_{i=1}^{k}(p_i x_i - p_i y_i)/k.$$

이때 다음 부등식은 자명하다.

$$\sum_{i=1}^{k}(\alpha_i - \beta)^2 = \sum_{i=1}^{k}\alpha_i^2 - 2\beta\sum_{i=1}^{k}\alpha_i + \beta^2 \geq 0.$$

이 식에 $\sum_{i=1}^{k}\alpha_i = \sum_{i=1}^{k}(p_i x_i - p_i y_i) = k\beta$ 을 사용하면 $\sum_{i=1}^{k}\alpha_i^2 - k\beta^2 \geq 0$ 이 된다. 그 중 첫 번째 항이 다음 식

$$\sum_{i=1}^{k}\alpha_i^2 = \sum_{i=1}^{k}p_i^2(x_i - y_i)^2 = \sum_{i=1}^{k}(x_i - y_i)^2 = d^2(x, y)$$

을 만족하므로 $d^2(x, y) \geq k\beta^2$ 가 성립한다. 따라서 식(1)이 증명됨을 알 수 있다. ■

위의 관계식을 벡터부호화기에 사용할 때에는 주어진 입력벡터 x에 대해 임의의 코드워드 y가 최소거리에 있는 코드워드인지 판단하기 전에 식(1)이 함축하는 바, 즉 $\left(\sum_{i=1}^{k}|x_i - y_i|\right)^2/k \geq d_{\min}^2$ 와 같은 관계식이 성립한다면 y는 x에 가장 가까운 코드워드일 수 없다는 사실을 이용하여 불필요한 거리계산을 줄일 수 있다.

하지만 위의 관계식을 이용한 알고리즘은 비교판단을 위한 계산량이 오버헤드로 작용하기 때문에 그 크기가 얼마만큼인지 살펴보아야 한다. 한 코드워드에 대해 식(1)을 이용한 판단에 필요한 연산수를 생각해보면 덧셈연산(뺄셈을 포함)이 2k-1회와 곱셈연산(나눗셈 포함) 2회, 그리고 비교연산(절대값을 연산을 위한) k회이다. 이에 반해 거리계산에는 덧셈연산 2k-1회와 곱셈연산 k회가 필요하다

코드북의 크기가 N이라고 하고 거리계산 횟수를 R이라고 할 때 제안된 알고리즘의 연산횟수를 자세히 조사해 보면, 거리계산을 하는 경우에 곱셈 R(k+2)회, 덧셈 R(3k-2)회 그리고 비교 Rk회가 필요하다. 비교판단 후 거리계산이 필요없는 경우에는 곱셈 2(N-R)회, 덧셈 (N-R)(2k-1) 그리고 비교 (N-R)k가 필요하다. 따라서 전체적으로는 곱셈 2N+Rk, 덧셈 N(2k-1)+R(k-1), 그리고 비교 Nk이 필요하게 되어, 전체 탐색법의 경우인 곱셈 Nk, 덧셈 N(2k-1)에 비교하면 곱셈에서는 N(k-2)-Rk만큼 이득을 그리고 덧셈과 나눗셈에서는 각각 R(k-1)회 그리고 비교 Nk를 손해보게 된다..

따라서 제안된 알고리즘은 비교연산과 덧셈연산을 늘리는 대신 곱셈연산을 줄이는 효과를 가져온다. 일반적으로 곱셈연산에 비해 덧셈연산과 비교연산은 훨씬 작은 실행시간을 가지므로 전체 실행시간에서는 큰 이득을 보게 된다. 이는 특히 고정소수점을 이용한 계산을 하는 DSP 칩의 경우에 그러하다. 따라서 고정소수점을 이용한 벡터부호화기의 구현에 있어서 제안된 알고리즘은 좋은 대안이 될 것으로 생각된다.

III. 실험 및 토론

제안된 알고리즘의 성능을 평가하기 위해서 네 개

의 화상을 이용한 실험이 행해졌다. 각 화상은 512x512 흑백 정지 화상이며 각 화소는 256 개의 흑백 단계로 이루어져 있다. 벡터의 차원은 4x4=16을 이용했으므로 한 화상의 훈련 데이타는 16 차원 벡터 16384개로 주어진다. 제안된 알고리즘은 거리계산 횟수 측면에서 전체 탐색법과 비교되었으며, 참조를 위해 고속알고리즘으로 알려진 ENNS와도 비교되었다. 실험에 사용된 코드북은 Lena 화상으로부터 LBG 알고리즘을 이용하여 얻었으며, 이 코드북을 사용하여 4개의 화상 (lena, peppers, jet, and baboon)을 각각 부호화했다.

표. 평균거리계산 횟수

크기	방법	화 상			
		Lena	Boat	Man	Baboon
128	Full Search	128	128	128	128
	ENNS	10.20	14.50	13.72	28.74
	Our Method	4.70	6.06	5.67	10.83
256	Full Search	256	256	256	256
	ENNS	17.64	26.43	25.10	55.16
	Our Method	5.76	7.92	7.77	16.51
512	Full Search	512	512	512	512
	ENNS	32.81	49.50	46.55	105.51
	Our Method	7.18	10.63	10.41	24.38
1024	Full Search	1024	1024	1024	1024
	ENNS	57.01	93.12	87.23	202.35
	Our Method	9.01	14.60	14.33	36.05

표는 각 화상을 부호화하는데 필요한 평균거리계산 횟수 측면에서 코드북의 크기를 128부터 1024까지 변화시키면서 구한 결과이다. 표를 보면 제안된 알고리즘은 거리계산에 필요한 코드워드의 개수가 앞서 얘기했듯이 매우 적어짐을 알 수 있다. 이것은 전체 탐색법과 비교했을 때에도 그러하지만 고속알고리즘으로 알려진 ENNS와 비교해서도 그러하다.

표에서 계산량 감소폭은 각 화상에 따라 조금씩 다른데 이것은 화상의 데이타가 코드워드를 중심으로 어떻게 분포되어 있는가에 따른 것이다. 일반적으로 얘기하면 부호화된 화상의 MSE가 높을수록 계산량의 감소폭이 떨어진다는 걸 확인할 수 있다.

보다 구체적으로 곱셈연산에서의 이득이 얼마만큼인지를 표에서 256개의 코드워드로 이루어진 코드북을 이용한 lena 화상의 예를 들어 살펴보기로 한다. 표에 따르면 이때 평균거리계산 횟수가 5.76으로 주어지기 때문에 곱셈에서 얻는 득은 앞서 보여주었던 식처럼 256(16-2) - 5.76x16 = 3491.84 이며, 반면 덧셈은 5.76x(16-1)=86.4 그리고 비교연산에서 256x16=4096 만큼의 손해를 보게 된다. 위에서와 같은 계산식을 사용하면 부호화에 사용된 DSP 칩 등의 연산처리속도 명세서를 이용하여 전체적으로 얻는 이득이 얼마인지 분명하게 계산될 수 있다.

IV. 결 론

본 논문에서는 입력벡터와 코드워드와의 유클리디안 거리계산 대신에 절대값의 합을 이용하여 주어진 입력벡터에 최단거리의 코드워드가 될 수 없는 코드워드를 제외함으로써 유클리디안 거리계산을 최소화하여 계산량을 줄이는 알고리즘을 제안하였다. 제안된 방법은 부호화에 필요한 곱셈연산을 효과적으로 줄여주는데 이것을 실험 결과를 통하여 확증하였다.

제안된 알고리즘에 사용된 판단 식은 기존의 고속 부호화 알고리즘에 결합되기 용이하므로 이에 대한 연구가 더 이루어져야 할 것이며, 비교 판단에 필요한 연산 오버헤드를 보다 줄일 수 있는 방법은 없는지에 대해서도 연구가 더 필요할 것이다.

참 고 문 헌

[1] R. M. Gray, "Vector Quntization," *IEEE ASSP Magazine*, vol. 1, pp. 4-9, Apr. 1984.

[2] J. Makhoul, S. Roucos, and H. Gish, "Vector Quantization," *Proc. IEEE*, vol. 71, pp. 1551-1588, Nov. 1985.

[3] N. Moayeri, D. L. Neuhoff, and W. E. Stark, "Fine-coarse vector quantization," *IEEE Trans. Commun.*, vol. COM-33, no. 10, pp. 1132-1133, Oct. 1985.

[4] V. Ramasubramanian and K. K. Paliwal, "Fast k-dimensional tree algorithms for nearest neighbor search with application to vector quantization encoding," *IEEE Trans. Signal Processing*, vol. 40, no. 3, pp. 518-531, Mar. 1992.

[5] C. D. Bei and R. M. Gray, "An improvement of the minimum distortion encoding algorithms for vector quantization and pattern matching," *IEEE Trans. Commun.*, vol. COM-33, no. 10, pp. 1132-1133, Oct. 1985.

[6] M. Orchard, "A fast nearest neighbor search algorithm," in *Proc. IEEE ICASSP*, Toronto, Canada, pp. 2297-2300, May 1992.

[7] C. M. Huang, Q. Bi, G. S. Stiles, and R. W. Harris, "Fast full search equivalent encoding algorithms for image compression using vector quantization," *IEEE Trans. Image Processing*, vol. 1, no. 3, July 1992.

[8] L. Guan and M. Kamel, "Equal-average hyperplane partitioning method for vector quantisation of image data," *Pattern Recognition Lett.*, pp. 693-699, 1992.

[9] S. W. Ra and J. K. Kim, "A fast mean-distance ordered partial codebook search algorithm for image vector quantization," *IEEE Trans. Circuits Syst. II*, vol 40, no. 9, pp. 576-579, Sept. 1993.

[10] S.J. Back, B.K. Jeon, and K.-M. Sung, "A Fast Encoding Algorithm for Vector Quantization," *IEEE Signal Processing Letters*, vol. 4, No. 12, pp. 325-327, Dec., 1997