

Associação da técnica ENNS ao algoritmo LBG

Edson Mata da Silva Filho

Centro de Ciências e Tecnologia

Universidade Católica de Pernambuco

Recife, PE, Brasil

Email: edsonmata@hotmail.com

Waslon Terllizzie Araújo Lopes

Departamento de Engenharia Elétrica

Universidade Federal de Campina Grande

Campina Grande, PB, Brasil

Email: waslon@ieee.org

Francisco Madeiro

Centro de Ciências e Tecnologia

Universidade Católica de Pernambuco

Recife, PE, Brasil

Email: madeiro@dei.unicap.br

Resumo—Uma técnica popular de projeto de quantizadores vetoriais é o algoritmo LBG (*Linde-Buzo-Gray*). Este trabalho apresenta uma alternativa para acelerar o algoritmo LBG. A aceleração, obtida por meio de redução de complexidade aritmética, é alcançada introduzindo o algoritmo ENNS (*Equal-average Nearest Neighbor Search*) na etapa de particionamento do LBG. Resultados de simulação mostram que a alternativa leva a economias de até 94,97% em termos de tempo de projeto de quantizadores vetoriais de imagens.

I. INTRODUÇÃO

A compressão de sinais digitais, como, por exemplo, voz, imagem, áudio e vídeo, visa diminuir a quantidade de bits que representam os sinais, tornando mais eficiente a sua transmissão ou o seu armazenamento. Existem duas categorias técnicas de compressão: técnicas de compressão com perdas, em que o sinal reconstruído não é idêntico ao sinal original, ou seja, sofre uma degradação, e técnicas de compressão sem perdas, em que o sinal reconstruído é idêntico ao original.

A quantização vetorial [1, 2, 20, 21, 22] é uma técnica de compressão com perdas, que consiste no mapeamento de um vetor x , de um espaço euclidiano K -dimensional, em um vetor pertencente a um subconjunto finito W de \mathbb{R}^K :

$$Q : \mathbb{R}^K \rightarrow W.$$

Denomina-se dicionário este subconjunto finito, onde cada elemento é denominado vetor-código, cujo número de componentes se denominada dimensão (K). O tamanho do dicionário é a quantidade de vetores-código, denotada por N . Em diversos sistemas de codificação de voz [3-5] e imagem [6-8], a QV tem sido utilizada com sucesso, permitindo a obtenção de elevadas taxas de compressão. A QV também tem sido utilizada em outras aplicações, tais como, identificação vocal [9, 10], esteganografia, marca d'água digital [11], detecção de bordas, casamento de padrões [12, 14], reconhecimento de palavras manuscritas [15].

Pode-se afirmar que a QV é extensão de quantização escalar em um espaço multidimensional, sendo seu desempenho diretamente ligado aos dicionários projetados. A elaboração desses dicionários se chama projeto de dicionários. O algoritmo mais conhecido para o projeto de dicionários é o *Linde-Buzo-Gray* (LBG) [16]. Em se tratando de projeto de quantizadores vetoriais, o algoritmo LBG pode ser entendido como uma alternativa para se “aprender” a obter um conjunto de padrões representativos de um conjunto de treino. Um

dicionário de boa qualidade tem N vetores-código que devem ser o mais representativo possíveis de um conjunto de treino de M vetores, com $M \gg N$.

Neste trabalho, é apresentada uma alternativa de aceleração do algoritmo LBG. A aceleração é obtida por meio da redução da complexidade aritmética associada à etapa de particionamento do algoritmo LBG. A redução é alcançada por meio da acomodação do algoritmo ENNS (*Equal-average Nearest Neighbor Search*) [18] na etapa de particionamento. Cumpre salientar que a aplicação de origem do algoritmo ENNS é a etapa de codificação por distância mínima, da quantização vetorial. No presente trabalho, o ENNS é usado em outro contexto: no projeto de dicionário por meio do algoritmo LBG. Resultados de simulação apontam que a acomodação do ENNS no algoritmo LBG contribui para reduzir em até 94,97% o tempo de processamento gasto para projetar dicionários voltados para quantização vetorial de imagens.

O trabalho encontra-se organizado de acordo com as seções a seguir. A Seção II aborda o algoritmo LBG. A Seção III aborda as técnicas PDS e ENNS para acelerar o processo de projeto de dicionário. Os resultados e os comentários finais são apresentados na Seção IV e Seção V, respectivamente.

II. ALGORITMO LBG

O algoritmo *Linde Buzo Gray* (LBG) [16] é também conhecido como *Generalized Lloyd Algorithm* (GLA) ou *K-means*. Seja a iteração do algoritmo LBG denotada por n . Dados K , N e um limiar de distorção $\epsilon \geq 0$, o algoritmo LBG consiste da seguinte sequência de passos:

- Passo 1) Inicialização: dado um dicionário inicial W_0 e um conjunto de treino $X = \{x_m; m = 1, 2, \dots, M\}$, faça $n = 0$ e $D_1 = \infty$;
- Passo 2) Particionamento: dado W_n (dicionário na n -ésima iteração), aloque cada vetor de treino (vetor de entrada) na respectiva classe segundo o critério do vetor-código mais próximo; calcule a distorção

$$D_n = \sum_{i=1}^N \sum_{m=1}^M d(x_m, w_i);$$

- Passo 3) Teste de convergência (critério de parada): se $(D_{n-1} - D_n)/D_n \leq \epsilon$ pare, com W_n representando o dicionário final (dicionário projetado); caso contrário, continue;

- Passo 4) Atualização do dicionário: compute os novos vetores-código como os centróides das classes de vetores; faça $W_{n+1} \leftarrow W_n$; faça $n \leftarrow n + 1$ e retorne ao Passo 2.

III. TÉCNICAS PDS E ENNS PARA ACELERAR A ETAPA DE PROJETO DE DICIONÁRIO

O algoritmo LBG utiliza o cálculo da distância euclidiana para determinar todos os vetores do dicionário. Convencionalmente, o LBG, na determinação do vizinho mais próximo (VMP), realiza uma busca total (*Full Search* - FS), que consome um tempo de execução computacional que poderia ser reduzido, pois ocorrerão casos na busca total em que cálculos são desnecessários. Podemos minimizar estes cálculos através de técnicas eficientes de busca do VMP, como os algoritmos PDS (*partial distance search*) e ENNS (*Equal-average Nearest Neighbor Search*). O algoritmo de busca por distância parcial, denotado como PDS, proposto em [17] determina se, para algum $i < K$, a distância acumulada para as primeiras i amostras do vetor de entrada é maior que a menor distância até então encontrada no processo de busca; quando isto ocorre, o vetor-código em questão não representa o vizinho mais próximo. Assim não é necessário realizar o cálculo de distância euclidiana em sua totalidade. Inicia-se este cálculo para o próximo vetor-código do dicionário. Com o PDS, o tempo de busca do VMP é reduzido, portanto o tempo de execução e a complexidade aritmética para projetar o dicionário, com o uso do PDS, diminuem.

O ENNS [18] usa o valor médio das componentes para rejeitar vetores-códigos distantes, ou seja, como a média do VMP é geralmente próxima da média do vetor de entrada, então caso as médias dos vetores-código sejam distantes, eles são descartados do processo de busca. Esse algoritmo reduz bastante o tempo computacional em relação à busca total [19].

O Algoritmo ENNS, descrito em [18], é originalmente usado na fase de codificação por distância mínima, da quantização vetorial, realizando apenas uma ordenação das médias, pois os vetores de dicionário já se encontram projetados. No entanto, na fase de projeto de dicionário, o algoritmo LBG, a cada iteração, atualiza seus vetores-código, tornando, assim, necessário realizar a ordenação dos vetores-código em cada iteração. Portanto o funcionamento do LBG aplicado com o ENNS será:

- Passo 1) inicialização: dado um dicionário inicial W_0 e um conjunto de treino $X = \{x_m; m = 1, 2, \dots, M\}$, faça $n = 0$ e $D_1 = \infty$;
- Passo 2) Calcula a Média dos vetores de entrada (m_x);
- Passo 3) Calcula a Média dos vetores-código (m_i);
- Passo 4) Ordena de forma crescente a média dos vetores-código;
- Passo 5) Particionamento, para o vetor-código que obtiver a menor diferença absoluta na média em relação ao vetor de entrada, é realizado o cálculo da distância euclidiana, que é denotado por d_{min} . Em seguida, os vetores-código para os quais $m_i \geq m_x + \sqrt{d_{min}/K}$ ou $m_i \leq m_x - \sqrt{d_{min}/K}$ são eliminados. Caso não sejam eliminados,

o PDS é aplicado para calcular a distância e atualizar d_{min} .

- Passo 6) Calcula a distância:

$$D_n = \sum_{i=1}^N \sum_{m=1}^M d(x_m, w_i);$$

- Passo 7) Teste de convergência (critério de parada): se $(D_{n-1} - D_n)/D_n \leq \epsilon$ pare, com W_n representando o dicionário final (dicionário projetado); caso contrário, continue;
- Passo 8) Atualização do dicionário: compute os novos vetores-código como os centróides das classes de vetores; faça $W_{n+1} \leftarrow W_n$; faça $n \leftarrow n + 1$ e retorne ao Passo 3.

IV. RESULTADOS

As simulações foram realizadas em um *Intel core I5-2450m* (2.50GHz) usando três imagens de 512 x 512 pixels, “Aerial”, “Baboon” e “Lena”, com 256 escalas de cinza, ilustradas na Figura 1, 2 e 3. Foram projetados dicionários com N assumindo 32, 64, 128 e 256 e $K = 16$ ou $K = 64$. As técnicas PDS e ENNS acomodados ao algoritmo LBG são comparadas em termo de tempo de CPU e complexidade aritmética (número médio de cálculos de distância por vetor de entrada) para diferentes tamanhos do dicionário e dimensão do vetor, como mostrado nas Tabelas I, II, III, IV, V e VI.



Figura 1. Imagem “Aerial” com resolução 512x512, 8bpp [13].

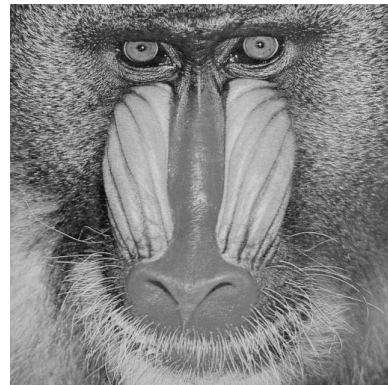


Figura 2. Imagem “Baboon” com resolução 512x512, 8bpp [13].



Figura 3. Imagem “Lena” com resolução 512x512, 8bpp [13].

Todos os valores gerados nas tabelas são médias calculadas a partir de 30 execuções. Além disso, foram usados os mesmos dicionários iniciais para cada valor do tamanho do dicionário em cada algoritmo. Nas Tabelas I, II, III, IV, V e VI são mostrados o tempo de execução e a complexidade aritmética de cada algoritmo considerado.

Tabela I

TEMPO DE EXECUÇÃO E COMPLEXIDADE ARITMÉTICA REALIZADA PELOS ALGORITMOS FS, PDS E ENNS PARA IMAGEM AERIAL COM $K = 16$.

N	Tempo CPU (s)			Complexidade		
	FS	PDS	ENNS	FS	PDS	ENNS
32	32,10	11,90	3,85	32,00	9,21	3,35
64	43,34	15,69	6,94	64,00	18,09	5,59
128	132,18	37,57	11,18	128,00	26,94	10,03
256	256,46	58,28	16,59	256,00	46,67	18,10

Um aspecto a destacar, na Tabela I, é o fato de que os algoritmos PDS e ENNS levam a tempos de execução muito menores que os obtidos com uso do procedimento convencional FS. Para o tamanho de dicionário 256, por exemplo, o FS requereu 256,46 s, ao passo que o ENNS requereu 16,59 s na etapa de projeto de dicionário, ou seja, o ENNS consumiu apenas 6,46% do tempo do FS.

Tabela II

TEMPO DE EXECUÇÃO E COMPLEXIDADE ARITMÉTICA REALIZADA PELOS ALGORITMOS FS, PDS E ENNS PARA IMAGEM AERIAL COM $K = 64$.

N	Tempo CPU (s)			Complexidade		
	FS	PDS	ENNS	FS	PDS	ENNS
32	29,39	11,66	3,51	32,00	13,97	6,19
64	33,32	11,84	6,44	64,00	24,53	11,27
128	59,34	16,74	9,01	128,00	39,28	18,32
256	104,80	25,13	10,17	256,00	71,66	33,04

Na Tabela II o algoritmo em destaque foi o ENNS por apresentar os melhores resultados tanto em tempo de execução como complexidade aritmética. Este algoritmo, com $N = 256$, apresenta uma melhora de 10,30 vezes, em seu tempo para projetar o dicionário e 7,74 vezes em seu número médio de operações, quando comparado ao algoritmo convencional de busca exaustiva (FS), além de uma melhora de 2,47 e 2,16

vezes para o tempo e complexidade respectivamente, quando comparado com o PDS.

Tabela III

TEMPO DE EXECUÇÃO E COMPLEXIDADE ARITMÉTICA REALIZADA PELOS ALGORITMOS FS, PDS E ENNS PARA IMAGEM BABOON COM $K = 16$.

N	Tempo CPU (s)			Complexidade		
	FS	PDS	ENNS	FS	PDS	ENNS
32	21,14	8,94	3,32	32,00	12,53	4,61
64	52,75	20,65	7,23	64,00	22,45	8,11
128	104,57	35,07	12,86	128,00	35,94	13,69
256	156,10	46,27	20,76	256,00	60,21	23,69

Na Tabela III, também é observada a superioridade do ENNS, pois, por exemplo, $N = 32$ seu tempo de execução é apenas 15,70% do algoritmo FS para obter os mesmos resultados. O PDS, por sua vez, precisou de 42,28% do tempo do FS. Em termos de complexidade, para a mesma configuração, o ENNS e o PDS obtiveram 14,40% e 39,15% de redução em relação ao algoritmo FS.

Tabela IV

TEMPO DE EXECUÇÃO E COMPLEXIDADE ARITMÉTICA REALIZADA PELOS ALGORITMOS FS, PDS E ENNS PARA IMAGEM BABOON COM $K = 64$.

N	Tempo CPU (s)			Complexidade		
	FS	PDS	ENNS	FS	PDS	ENNS
32	13,86	7,06	4,15	32,00	15,35	6,91
64	24,40	11,01	5,65	64,00	27,54	12,56
128	41,59	15,76	10,94	128,00	45,59	20,86
256	78,94	28,02	13,77	256,00	85,62	37,66

Na Tabela IV, observa-se que o algoritmo ENNS obteve uma redução no número de operações em cerca de 85,28%, com $N = 256$, em relação ao FS. Já o PDS reduziu 66,55%.

Tabela V

TEMPO DE EXECUÇÃO E COMPLEXIDADE ARITMÉTICA REALIZADA PELOS ALGORITMOS FS, PDS E ENNS PARA IMAGEM LENA COM $K = 16$.

N	Tempo CPU (s)			Complexidade		
	FS	PDS	ENNS	FS	PDS	ENNS
32	21,88	5,68	2,31	32,00	6,02	1,22
64	37,44	8,88	3,45	64,00	10,15	2,24
128	86,36	17,14	5,33	128,00	15,29	3,83
256	209,74	38,00	7,26	256,00	26,46	7,17

A complexidade aritmética, ou seja, número médio de cálculos de distância por vetor de entrada, em destaque na Tabela V foi do algoritmo ENNS, pois, por exemplo, com $N = 256$ é de apenas 2,86% do algoritmo FS, enquanto o PDS necessitou de 10,33%. Em se tratando de tempo de execução, para a configuração, o ENNS e o PDS obtiveram uma melhora de 28,88 e 5,51 vezes respectivamente, em relação ao algoritmo FS.

Na Tabela VI podemos observar que para $N = 256$ o algoritmo FS utilizou 104,38 s. Por sua vez, o ENNS precisou de apenas 5,25 s para projetar o dicionário, ou seja, o ENNS consumiu cerca de 5,03% do FS. Ao analisar o PDS, com mesma configuração, observou-se que necessitou de 14,74% do FS.

Tabela VI

TEMPO DE EXECUÇÃO E COMPLEXIDADE ARITMÉTICA REALIZADA PELOS ALGORITMOS FS, PDS E ENNS PARA IMAGEM LENA COM $K = 64$.

N	Tempo CPU (s)			Complexidade		
	FS	PDS	ENNS	FS	PDS	ENNS
32	23,79	6,30	2,18	32,00	9,38	1,85
64	32,49	6,88	2,28	64,00	11,50	3,39
128	54,80	8,70	3,23	128,00	19,08	5,93
256	104,38	15,39	5,25	256,00	35,12	10,50

V. COMENTÁRIOS FINAIS

Um problema relevante em quantização vetorial (QV) é o projeto de dicionário. O desempenho de sistemas de processamento digital de sinais baseados em QV depende dos dicionários projetados. Uma técnica popular de projeto de dicionário é o algoritmo LBG (*Linde-Buzo-Gray*). Este trabalho apresentou uma alternativa de aceleração do algoritmo LBG. Mais precisamente, a alternativa consistiu em acomodar o algoritmo ENNS (*Equal-average Nearest Neighbor Search*) na etapa de particionamento do algoritmo LBG. Com a “acomodação” do ENNS no algoritmo LBG, obteve-se uma redução do tempo de projeto de dicionário quando comparada à versão original do algoritmo LBG. Em simulações envolvendo projeto de dicionário para QV de imagens, foram obtidas economias de até cerca de 94,97% em termos de tempo de execução.

AGRADECIMENTOS

Este trabalho contou com apoio financeiro do CNPq, do programa PIBITI da Universidade Católica de Pernambuco.

REFERÊNCIAS

- [1] A. Gercho and R. M. Gray, “Vector Quantization and Signal Compression”. Kluwer Academic Publishers, Boston, MA, 1992.
- [2] R. M. Gray, “Vector Quantization”. IEEE ASSP Magazine, pp. 4-29, April 1984.
- [3] K. K. Paliwal and B. S. Atal. “Efficient Vector Quantization of LPC Parameters at 24 Bits/Frame”. IEEE Transactions on Speech and Audio Processing, vol. 1, no. 1, pp. 3–14, January 1993.
- [4] Y. Agiomyrgiannakis and Y. Stylianou. “Conditional Vector Quantization for Speech Coding”. IEEE Transactions on Audio, Speech, and Language Processing, vol. 15, no. 2, pp. 377–386, February 2007.
- [5] S. Chatterjee and T. Sreenivas. “Conditional PDF-Based Split Vector Quantization of Wideband LSF Parameters”. IEEE Signal Processing Letters, vol. 14, no. 9, pp. 641–644, September 2007.
- [6] K. Sasazaki, S. Saga, J. Maeda and Y. Suzuki. “Vector Quantization of Images with Variable Block Size”. Applied Soft Computing, no. 8, pp. 634–645, 2008.
- [7] A. Averbuch, D. Lazar and M. Israeli. “Image Compression Using Wavelet Transform and Multiresolution Decomposition”. IEEE Transactions on Image Processing, vol. 5, no. 1, pp. 4–15, January 1996.
- [8] W. T. A. Lopes, F. Madeiro, B. G. Aguiar Neto and M. S. Alencar. “Combining Modulation Diversity and Index Assignment to Improve Image VQ for a Rayleigh Fading Channel”. Learning and Nonlinear Models, vol. 2, no. 1, pp. 22-33, 2004.
- [9] J. He, L. Liu and G. Palm. “A Discriminative Training Algorithm for VQ-based Speaker Identification”. IEEE Transactions on Speech and Audio Processing, vol. 7, no. 3, pp. 353–356, May 1999.
- [10] F. Madeiro, J. M. Fachine, W. T. A. Lopes, B. G. Aguiar Neto and M. S. Alencar. “Identificação Vocal por Frequência Fundamental, QV e HMMS”. In Em-TOM-Ação: A Prosódia em Perspectiva, edited by M. A. M. Aguiar and F. Madeiro, chapter 4, pp. 91–120. Editora Universitária da UFPE, Recife, PE, first edition, 2007.
- [11] W.-C. Chen and M.-S. Wang. “A Fuzzy c-means Clustering-based Fragile Watermarking Scheme for Image Authentication”. Expert Systems with Applications, vol. 36, no. 2, pp. 1300–1307, March 2009.
- [12] K. L. Oehler and R. M. Gray, “Combining image classification and image compression using vector quantization”. Proceedings Data Compression Conference, pp. 2-11, Snowbird, UT, March 1993.
- [13] USC-SIPI Image Database - University of Southern California. Endereço web: <http://sipi.usc.edu/database/>. Acesso em Junho/2012
- [14] M. Atallah, Y. Genin, W. Szpankowski, “Pattern Matching Image Compression: Algorithmic and Empirical Results”. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 21, pp. 618-627, 1999.
- [15] L. R. Veloso. “Sistema de Reconhecimento de Palavras Manuscritas Dependente do Usuário”. Tese (Doutorado em Engenharia Elétrica), Universidade Federal de Campina Grande, 2009.
- [16] Y. Linde, A. Buzo and R. M. Gray. “An Algorithm for Vector Quantizer Design”. IEEE Transactions on Communications, vol. 28, no. 1, pp. 84–95, January 1980.
- [17] C. D. Bei and R. M. Gray, “An Improvement of the Minimum Distortion Encoding Algorithm for Vector Quantization”. IEEE Transactions on Communications, vol. COM-33, no. 10, pp. 1132-1133, October 1985.
- [18] S.W. Ra, J.K. Kim, “Fast mean-distance-ordered partial codebook search algorithm for image vector quantization”. IEEE Transactions on Circuits Systems II 40 (9) pp. 576–579, 1993.
- [19] S. C. Chu, Z. M. Lu and J. S. Pan, “Hadamard Transform Based Fast Codeword Search Algorithm for High-Dimensional VQ Encoding”. Elsevier Science Inc., vol. 177, pp. 734-746, February 2007.
- [20] J S Pan, Z M Lu, and S H Sun, “An Efficient Encoding Algorithm for Vector Quantization Based on Subvector Technique”. IEEE Transactions on image processing, vol. 12, no. 3 pp. 265-70, March 2003.
- [21] Y C Liaw, J Z C Lai, and W Lo, “Image Restoration of Compressed Image Using Classified Vector Quantization”. Pattern Recognition, vol. 35, no.2, pp. 181-92, 2002.
- [22] T Kaukoranta, P Fronti, and O Nevalainen, “A fast exact GLA based code vector activity detection”. IEEE Transactions on Image Processing, vol. 9, no.8, pp. 1337-42, 2000.