

Rotated Partial Distance Search for Faster Vector Quantization Encoding

James McNames

Abstract— **Partial Distance Search (PDS)** is a method of reducing the amount of computation required for vector quantization encoding. The method is simple and general enough to be incorporated into many fast encoding algorithms. This paper describes a simple improvement to PDS, based on principal components analysis, that rotates the codebook without altering the interpoint distances. Like PDS, this new method can be used to improve many fast encoding algorithms. The algorithm decreases the decoding time of PDS by as much as 44% and decreases the decoding time of k-d trees by as much as 66% on common vector quantization benchmarks.

Keywords— **Vector Quantization Encoding, Nearest Neighbor, Principal Components Analysis, k-d Tree, Eigenvector Method**

I. INTRODUCTION

VECTOR quantization is a popular and powerful method of compression. During encoding, the nearest neighbor in a codebook is found for each signal vector and the index of the nearest neighbor is used to encode, or represent, the signal vector. The most common measure of nearness is the Euclidean distance, or equivalently, the square of the Euclidean distance, $d^2(s, x_i) = \sum_{j=1}^{n_d} (s_j - x_{i,j})^2$, where s is the signal vector, x_i is the i th vector in the codebook, and n_d is the dimension of the vectors.

Although vector quantization is theoretically a powerful method of compression, the large computational cost of finding the nearest neighbor during encoding imposes practical limits on the codebook size and the compression rate. In some applications, it is impractical to use a brute force approach of exhaustively calculating the distance to each vector in the codebook. This has motivated the development of many fast vector quantization encoding algorithms, which are also known as fast nearest neighbor algorithms in the literature.

A. Partial Distance Search

Cheng *et al.* originally proposed a method called partial distance search (PDS)¹ that has gained widespread acceptance as a more efficient method than the brute force method [4]. PDS consists of a simple modification to the way that distances are calculated. During the calculation of the distance sum, if the partial distance exceeds the distance to the nearest neighbor found so far, the calculation

is aborted. This method substantially decreases the computation despite the added cost of performing a comparison for each element of the distance sum.

B. Previous Work

Many other encoding algorithms have been proposed to overcome the large computational cost of the brute force method. Each estimates a lower bound on the distance between the signal vector and a vector, or set of vectors, in the codebook. If the lower bound is greater than the distance to the nearest neighbor found so far, the point can be eliminated without explicitly calculating the distance to that point. When the bounding criteria fail, the fastest algorithms use PDS to calculate the true distance [5].

This work describes how PDS can be improved by a principal component rotation (PCR) of the codebook. Other methods have incorporated PCR as part of their bounding criteria, but if the bounds are not satisfied, most of these methods revert to calculating the full distance in the original coordinates using either the brute force method or PDS [6–10]. PCR has also been reported to improve performance in a voronoi-based encoding algorithm, but this method did not use PDS or discuss the potential of PCR to increase performance in other encoders [11].

II. CODEBOOK ROTATION

Principal components analysis (PCA) can be used to rotate the signal vector and code vectors without changing the interpoint distances. For example, if we represent the codebook as a matrix A such that each row contains the transpose of a codebook vector, the singular value decomposition of this matrix, $A = U\Sigma V^T$, can be used to find the principal directions in the matrix V . Since V is a unitary matrix, that is $V^T V = I$, multiplying the signal vector and the codebook vectors by this matrix does not change the distance between them. Specifically,

$$\begin{aligned} d^2(Vs, Vx_i) &= (Vs - Vx_i)^T (Vs - Vx_i), \\ &= (s - x_i)^T V^T V (s - x_i), \\ &= (s - x_i)^T (s - x_i), \\ &= d^2(s, x_i). \end{aligned}$$

The matrix V can also be calculated by an eigenvector decomposition of the correlation matrix $C = A^T A$.

If the vectors in the matrix V are arranged in order of decreasing singular values, this rotation causes the distance along the directions of largest variance to be calculated first. Thus, on average, the distance summations can be aborted more quickly than in the original coordinates.

J. McNames is with the Electrical and Computer Engineering Department at Portland State University. Mail: Electrical Engineering Department, Portland State University, Post Office Box 751, Portland, Oregon 97207-0751. Phone: 503.725.5390. Fax: 503.725.3807. E-mail: mcnames@ee.pdx.edu. Web: www.ee.pdx.edu/~mcnames.

¹Others have also proposed PDS independently [1–3].

This new method is called rotated partial distance search (RPDS).

A strong advantage of RPDS is that it can be incorporated into other fast encoding algorithms just like PDS. When the bounding criteria of the other methods fail, the true distance can be calculated using RPDS instead of PDS. PCR also improves the effectiveness of the bounding criteria of some encoders which further decreases the average encoding time [11].

III. PERFORMANCE

To determine how much principal component rotation improves performance of encoders in vector quantization applications, a range of different codebooks were constructed for two speech signals and two gray-scale images that are commonly used as benchmarks for vector quantization algorithms.² The codebooks were constructed using the clustering algorithm described by Franti *et al.* [12].

For each codebook the average encoding time was measured for the brute force method (Brute), PDS, and RPDS. To illustrate that RPDS can also be used to improve the performance of established fast encoding algorithms such as k-d trees [13], the average encoding time was also measured for k-d trees with PDS (KD-PDS) and k-d trees with RPDS (KD-RPDS)³.

Table I shows the average encoding times for the speech signals calculated using 35,000 signal vectors and Table II shows the average encoding times for the benchmark images *Lena* and *Baboon*⁴. For the images, the average encoding times were calculated using 20,000 signal vectors for the 2×2 image blocks and 15,000 signal vectors for the 4×4 image blocks. In all cases, the signal vectors were taken from the original data sets used to construct the codebook.

The performance improvement of RPDS as compared to PDS ranged from slightly worse to more than a 65% decrease in encoding time. The performance was slightly worse for low dimensional vectors due to the increased overhead of RPDS. The improvement was more dramatic in higher dimensions because the distance sums were larger and there was greater potential decrease in encoding time from truncating the distance sums.

IV. DISCUSSION

In some applications, RPDS requires approximately twice as much storage as PDS because both the original codebook and the rotated codebook must be stored. Once the index of the nearest neighbor is found using the rotated codebook, the original vector is retrieved from the original codebook. This extra storage can be eliminated by storing only the rotated codebook and rotating the vector back to the original coordinates by a multiplication with V^T .

²At the time of writing, the data sets were available at <http://www.ece.pdx.edu/~mcnames/DataSets>.

³The results reported here were generated using a Pentium 200 MHz MMX processor with 512 KB Cache, 128 MB RAM, Windows NT 4.0 SP5, and Visual C++ 6.0 SP1.

⁴This image is sometimes called *Mandrill*.

TABLE I

ALGORITHMS' AVERAGE ENCODING TIMES (MILLISECONDS) TO FIND THE NEAREST NEIGHBOR FOR A CODEBOOK CONSTRUCTED FROM THE SPEECH SIGNALS *WetSucker* AND *DarkSide*. THE RESULTS ARE SHOWN FOR FOUR CODEBOOK SIZES AND TWO VECTOR SIZES.

Algorithm	8 Dimensions				16 Dimensions			
	1024	2048	4096	8192	1024	2048	4096	8192
Brute	1.959	3.779	7.782	15.577	3.684	7.278	14.544	29.672
PDS	0.613	1.127	2.286	4.443	0.924	1.691	3.121	5.976
RPDS	0.512	0.933	2.035	3.989	0.609	1.096	2.294	4.661
Decrease	16.5%	17.2%	11.0%	10.2%	34.1%	35.2%	26.5%	22.0%
KD-PDS	0.200	0.230	0.320	0.342	0.649	0.930	1.197	1.528
KD-RPDS	0.132	0.152	0.219	0.226	0.365	0.503	0.661	0.840
Decrease	33.8%	34.0%	31.7%	33.8%	43.7%	45.9%	44.8%	45.0%

(a) WetSucker

Algorithm	8 Dimensions				16 Dimensions			
	1024	2048	4096	8192	1024	2048	4096	8192
Brute	1.951	3.878	7.776	15.836	3.611	7.314	14.212	29.656
PDS	0.622	1.200	2.348	4.660	0.959	1.802	3.204	6.276
RPDS	0.530	0.995	1.831	4.149	0.599	1.165	2.128	4.539
Decrease	14.8%	17.0%	22.0%	11.0%	37.6%	35.4%	33.6%	27.7%
KD-PDS	0.179	0.250	0.377	0.496	0.604	0.869	1.231	1.556
KD-RPDS	0.107	0.147	0.225	0.373	0.219	0.312	0.418	0.613
Decrease	40.5%	41.3%	40.4%	24.8%	63.8%	64.1%	66.0%	60.6%

(b) DarkSide

TABLE II

ALGORITHMS' AVERAGE ENCODING TIMES (MILLISECONDS) TO FIND THE NEAREST NEIGHBOR FOR A CODEBOOKS CONSTRUCTED FROM THE IMAGES *Baboon* AND *Lena*. THE RESULTS ARE SHOWN FOR FOUR CODEBOOK SIZES AND TWO VECTOR SIZES.

Algorithm	2×2				4×4			
	1024	2048	4096	8192	1024	2048	4096	8192
Brute	1.085	2.134	4.253	8.618	3.704	6.965	14.570	30.024
PDS	0.529	1.045	2.032	4.067	1.075	1.846	3.530	6.378
RPDS	0.515	0.968	1.911	4.037	0.645	1.019	2.184	4.457
Decrease	2.6%	7.3%	6.0%	0.7%	40.0%	44.8%	38.1%	30.1%
KD-PDS	0.074	0.094	0.108	0.135	0.754	1.100	1.479	1.184
KD-RPDS	0.074	0.092	0.114	0.140	0.321	0.447	0.582	0.589
Decrease	0.7%	2.2%	-5.6%	-3.7%	57.4%	59.4%	60.7%	50.3%

(a) Baboon

Algorithm	2×2				4×4			
	1024	2048	4096	8192	1024	2048	4096	8192
Brute	1.100	2.103	4.289	8.565	3.414	7.013	14.447	29.855
PDS	0.504	0.963	1.961	3.919	0.581	1.155	2.284	4.593
RPDS	0.480	0.927	1.871	3.823	0.463	0.955	1.833	4.009
Decrease	4.8%	3.7%	4.6%	2.5%	20.3%	17.3%	19.8%	12.7%
KD-PDS	0.054	0.071	0.088	0.112	0.230	0.334	0.432	0.479
KD-RPDS	0.049	0.063	0.082	0.106	0.107	0.164	0.198	0.240
Decrease	9.3%	11.3%	6.8%	5.4%	53.5%	50.9%	54.3%	49.9%

(b) Lena

Since V is unitary, $V^T V x_i = x_i$. In most cases, the extra computation of this technique is negligible compared to the cost of finding the nearest neighbor. For vector quantization applications this problem is averted entirely if the rotated codebook is stored by the transmitter and the original codebook is stored in the receiver.

Encoding algorithms are also often part of the clustering algorithm used for codebook construction. Unlike most encoding applications, the codebook construction algorithms are sensitive to the amount of preprocessing required because during each iteration of the clustering algorithm the data set changes and the preprocessing of the encoding al-

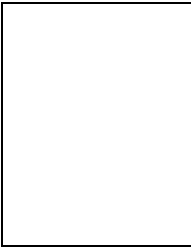
gorithm must be repeated. PDS is one of the most popular encoding algorithms for codebook construction because it is substantially faster than the brute force method and does not require any preprocessing. RPDS is also an appealing algorithm for codebook construction because it is significantly faster than PDS and the amount of preprocessing can be controlled by limiting the number of data set vectors used to estimate the principal components.

V. CONCLUSIONS

This paper described how partial distance search (PDS), a general-purpose improvement to fast encoding algorithms, could be improved by a principal component rotation (PCR) of the codebook. The new method, called RPDS, was compared to PDS on several different vector quantization benchmarks. When used alone the reduction in average encoding time was as much as 44%. When combined with k-d trees the encoding time was decreased by as much as 66%.

REFERENCES

- [1] Chang-Da Bei and Robert M. Gray, "An improvement of the minimum distortion encoding algorithm for vector quantization," *IEEE Transactions on Communications*, vol. 33, no. 10, pp. 1132–1133, Oct. 1985.
- [2] Jack Bryant, "A fast classifier for image data," *Pattern Recognition*, vol. 22, no. 1, pp. 45–48, 1989.
- [3] Patrick J. Grother, Gerald T. Candela, and James L. Blue, "Fast implementations of nearest neighbor classifiers," *Pattern Recognition*, vol. 30, no. 3, pp. 459–465, 1997.
- [4] De-Yuan Cheng, Allen Gersho, Bhaskar Ramamurthi, and Yair Shoham, "Fast search algorithms for vector quantization and pattern matching," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, Mar. 1984, vol. 1, pp. 9.11.1–9.11.4.
- [5] James McNames, *Innovations in Local Modeling for Time Series Prediction*, Ph.D. thesis, Stanford University, 1999.
- [6] Chang-Hsing Lee and Ling-Hwei Chen, "High-speed closest codeword search algorithms for vector quantization," *Signal Processing*, vol. 43, pp. 323–331, 1995.
- [7] Yih-Chuan Lin and Shen-Chuan Tai, "Dynamic windowed codebook search algorithm in vector quantization," *Optical Engineering*, vol. 35, no. 10, pp. 2921–2929, Oct. 1996.
- [8] Chin-Chen Chang, Wen-Tsai Li, and Tung-Shou Chen, "Two improved codebook search methods of vector quantization based on orthogonal checking and fixed range search," *Journal of Electronic Imaging*, vol. 7, no. 2, pp. 357–366, Apr. 1998.
- [9] S. C. Tai, C. C. Lai, and Y. C. Lin, "Two fast nearest neighbor searching algorithms for image vector quantization," *IEEE Transactions on Communications*, vol. 44, no. 12, pp. 1623–1628, Dec. 1996.
- [10] Chin-Chen Chang and Dai-Chuan Lin, "An improved VQ codebook search algorithm using principal component analysis," *Journal of Visual Communication and Image Representation*, vol. 8, no. 1, pp. 27–37, 1997.
- [11] V. Ramasubramanian and Kuldip K. Paliwal, "Fast nearest neighbor search based on voronoi projections and with its application to vector quantization encoding," *IEEE Transactions on Speech and Audio Processing*, vol. 7, no. 2, pp. 221–226, Mar. 1999.
- [12] Pasi Fränti, Timo Kaukoranta, and Olli Nevalainen, "On the splitting method for vector quantization codebook generation," *Optical Engineering*, vol. 36, no. 11, pp. 3043–3051, Nov. 1997.
- [13] Jerome H. Friedman, Jon Louis Bentley, and Raphael Ari Finkel, "An algorithm for finding best matches in logarithmic expected time," *ACM Transactions on Mathematical Software*, vol. 3, no. 3, pp. 209–226, Sept. 1977.



James McNames is currently an assistant professor at Portland State University. He received his B.S. degree in Electrical Engineering from California Polytechnic State University, San Luis Obispo, in 1992. He received his M.S. degree in Electrical Engineering from Stanford University in 1995 and his Ph.D. degree in Electrical Engineering from Stanford University in 1999. His research is primarily in the areas of biomedical signal processing, nonlinear modeling, and time series prediction.