ASYMPTOTIC CLOSED-LOOP DESIGN OF PREDICTIVE MULTI-STAGE VECTOR QUANTIZERS

Hosam Khalil and Kenneth Rose

Signal Compression Laboratory
Department of Electrical and Computer Engineering
University of California, Santa Barbara, CA 93106, USA
hkhalil@scl.ece.ucsb.edu rose@ece.ucsb.edu

ABSTRACT

This work considers the design of a multi-stage vector quantizer (MSVQ) for application to the motion compensated prediction error of video signals. It is well known that the design of predictive vector quantizers suffers from fundamental difficulties due to the prediction loop, which have an impact on the convergence and the stability of the design procedure. In this paper we propose an approach to predictive MSVQ design that enjoys the stability of open-loop design while ensuring ultimate optimization of the closed-loop system. The proposed design method is tested on video compression at low bit rates, where it significantly outperforms widely-used closed-loop design techniques, and achieves improvement over the H.263 standard.

1. INTRODUCTION

Digital video data transmission over communication channels with limited bandwidth requires the use of lossy compression techniques. Low bit rates, particularly in the range of 16-32 kbits/s, are gaining in importance due to the Internet and wireless mobile communications. Compression to achieve acceptable quality at very low bit rates is a considerably difficult task, and it is not known which fundamental compression technique is best suited for it.

The most widely used approach to efficient video compression is based on motion compensated prediction and quantization. A typical video coder consists of a frame prediction module and a prediction error (residual) compression module. The first module exploits the temporal redundancy implicit in the similarity between consecutive frames and typically involves block-based motion compensation. The second module is the lossy part of the codec where the prediction error, or prediction residual, is compressed to the appropriate bit rate. The predominant residual compression approach involves application of the Discrete Cosine Transform (DCT), and this is the method of choice in major standards such as H.263 and MPEG.

An important justification for the use of DCT in still image compression hinges on the assumption that the signal can be well modeled as a Gauss-Markov process with a high autocorrelation coefficient. It has been shown that the performance of the optimal (Karhunen-Loeve) transform on such a signal is closely approximated by that of DCT [1]. However, this argument does not hold for the prediction residual signal whose statistics are considerably different from those of a still image. It is, therefore, plausible that some other approach to residual compression, which takes into account the actual signal statistics, would provide substantial gains. An interesting example of a recent successful method, which departs from main-stream DCT techniques, is based on matching pursuits [2] where the residual is approximated using entries from a library of predefined two-dimensional functions (an over-complete dictionary of separable Gabor functions).

In this paper, we pursue a known alternative approach which is based on vector quantization (VQ). Generally, there are several arguments in support of VQ for video compression. Shannon's theory implies that vector quantizers are asymptotically optimal, where asymptotic here is in terms of vector length. In our case, typical blocks in video coding correspond to long vectors. Another important argument is that VQ is a very general framework and subsumes, for example, DCT compression as a special constrained case [3]. Thus, it may be argued that DCT can not outperform the best VQ. On the other hand, there exist two main objections to the use of VQ in video coding. The first objection is that of complexity: The VQ complexity grows exponentially with the product of vector dimension and rate. This objection naturally leads to the use of multi-stage VQ, which has been found to be an efficient technique, with acceptable search and memory complexity. The second objection is concerned with the interplay of prediction and quantization. As will be explained in detail, traditional design of predictive VQ (PVQ) is problematic, and often fails to produce an optimal, or even a good, quantizer. We previously proposed a solution to this problem in [4] for the simpler case of single-stage vector quantizers. In this paper, we extend the new predictive quantizer design to the more-complex, but more useful, case of entropy-constrained predictive multi-stage vector quantizers (EC-PMSVQ).

The paper is organized as follows: In Section 2, we summarize the traditional design algorithms of EC-PMSVQ and introduce a design technique based on asymptotic closed loop optimization. In Section 3, we present simulation re-

This work is supported in part by the NSF under grant MIP-9707764, the University of California MICRO Program, Conexant Systems, Inc., Fujitsu Laboratories of America, Inc., Lernout & Hauspie Speech Products, Lucent Technologies, Inc., and Qualcomm, Inc.

sults demonstrating the performance of EC-PMSVQ design in the context of video coding.

2. EC-PMSVQ DESIGN TECHNIQUES

2.1. Traditional Design

A major issue in predictive quantizer design involves the need to obtain a stable training set that accurately represents the true signal statistics. In [5], two techniques were introduced, and have been widely used since, for both predictive VQ and predictive MSVQ design.

2.1.1. Open-loop Design

In the open-loop (OL) approach, a training set of prediction error vectors is extracted directly from the original, *unquantized* source vectors. The approach is called "open-loop" because the reconstructed vectors are not fed back through the predictor. Specifically, given a set of source vectors, $X : \{x_0, x_1, x_2, \cdots, x_N\}$, and prediction operator P, we generate the required training set via

$$t_n = x_n - P[x_{n-1}], \qquad n = 1, 2, \ldots, N. \tag{1}$$

The main advantage of the OL approach is that the training set, $T : \{t_1, t_2, \cdots, t_N\}$ is fixed. Therefore, we can design the multi-stage quantizer by applying a standard optimization technique such as in [6] or [7]. Since the training set remains unchanged, the design algorithm is ensured to converge to a locally optimal solution. However, the OL approach suffers from a serious shortcoming. The decoder does not have access to the original source vector for prediction. Therefore, during the actual operation of the compression system, prediction must be performed using reconstructed source vectors. Thus, the training set of prediction errors is statistically different from the prediction errors to be quantized in practice. The statistical mismatch, which is exacerbated by error build-up via feedback through the prediction loop, results in poor performance.

2.1.2. Closed-loop Design

In the closed-loop (CL) approach, a real system is used to generate the prediction errors in an iterative fashion to avoid the statistical mismatch problem of the OL method. Given an MSVQ at iteration $i-1$, which we denote by $\boldsymbol{Q}^{(i-1)}$, a training set of prediction errors $T^{(i)} : \{t_1^{(i)}, t_2^{(i)}, \cdots, t_N^{(i)}\}$, is generated for iteration i:

$$t_n^{(i)} = x_n - P[\hat{x}_{n-1}^{(i)}], \tag{2}$$

where

$$\hat{x}_n^{(i)} = P[\hat{x}_{n-1}^{(i)}] + \boldsymbol{Q}^{(i-1)}(t_n^{(i)}). \tag{3}$$

Note that we must alternate between (2) and (3) in order to generate the prediction errors for $n = 1, 2, \ldots, N$. Given the resulting set of prediction errors, a new quantizer, $\boldsymbol{Q}^{(i)}$, is designed. The design of $\boldsymbol{Q}^{(i)}$ involves the optimization of the stage codebooks $Q_p^{(i)}$, $p = 1, 2, \ldots, P$. Next, a new sequence of prediction errors is generated for iteration $i+1$, and so on. The OL method is used to obtain $\boldsymbol{Q}^{(0)}$.

Since the training residuals were generated by the actual closed-loop coder, they are expected to match input residual error statistics to be encountered during operation. However, convergence of the algorithm is not guaranteed, as the training set changes every iteration in an unpredictable fashion due to the feedback loop.

The instability problems are more pronounced in the case of multi-stage quantizers. Training vectors that have accumulated considerable error (as a result of the closed loop) will enforce undesirable modification of the codebooks in an effort to accommodate outliers. More specifically, the objective of the training algorithm is to adjust the parameters of the quantizer so as to minimize the training distortion. When vectors with excessive error accumulation are included with the training set, the design algorithm will attempt to minimize coding distortion by allocating precious resources to unrepresentative vectors. The instability of the design is clearly apparent as the degraded codebooks will produce greater errors in the next iteration which, in turn, will further corrupt the codebooks. The approach presented next is largely motivated by the realization that a stable and effective training procedure for PMSVQ, or EC-PMSVQ, is imperative for an efficient and competitive coder.

2.2. The Asymptotic Closed-Loop Design

Motivated by the shortcomings of the existing methods, we propose the Asymptotic Closed-Loop (ACL) approach, which offers improved design stability. The ACL design enjoys the best of both worlds, namely, it inherits the design stability of open-loop techniques while ultimately optimizing the system for closed loop operation. The ACL procedure for EC-PMSVQ design is illustrated in Fig. 1, and is explained below.

The main objective is to avoid accumulation of errors due to mismatched quantization through the prediction loop. We therefore base our prediction on the reconstructed vectors of the *previous iteration*. The training set is, in effect, generated by

$$t_n^{(i)} = x_n - P[\hat{x}_{n-1}^{(i-1)}], \qquad n = 1, 2, \cdots, N. \tag{4}$$

Having collected the set of training vectors we optimize a new MSVQ, $\boldsymbol{Q}^{(i)}$, by individually optimizing the stage codebooks $Q_p^{(i)}$, $p = 1, 2, \ldots, P$. The resulting MSVQ is then used to generate the new set of reconstruction vectors:

$$\hat{x}_n^{(i)} = P[\hat{x}_{n-1}^{(i-1)}] + \boldsymbol{Q}^{(i)}(t_n^{(i)}), \qquad n = 1, 2, \cdots, N. \tag{5}$$

Compare (4) with (1) for open-loop, and with (2) for the closed-loop design. The CL design alternates between (2) and (3) while generating each training vector. In our approach, execution of (4) alone is used for the generation of the *entire* training sequence, without the effect of quantization error accumulation. Once the training set is generated, we design a new quantizer and then calculate (5) for *all* n, before proceeding with the next iteration.

Note that the quantizer $\boldsymbol{Q}^{(i)}$ is used to encode *exactly* the same prediction error vectors used for its design. Neglecting the possible problems of local optima, this is the best quantizer for these vectors. We are thus assured that

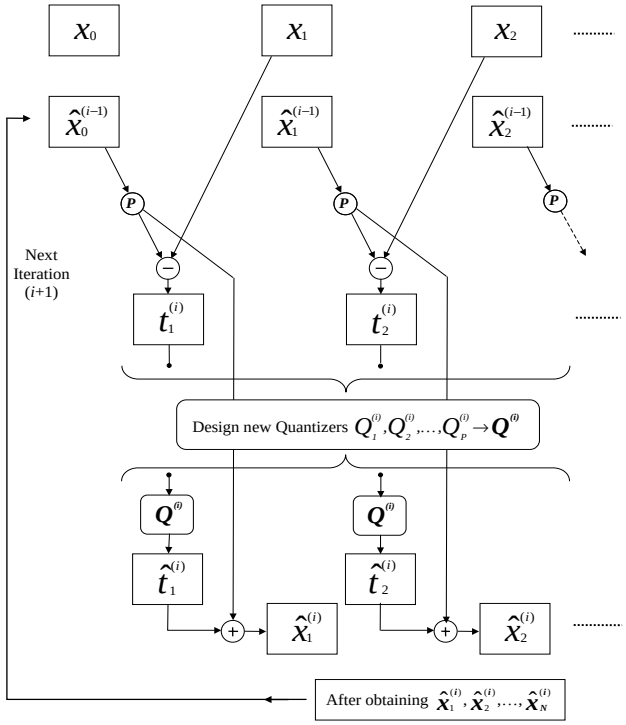

Figure 1: Proposed ACL procedure: $\boldsymbol{Q}^{(i)}$ is the EC-PMSVQ trained on the prediction error sequence from iteration i. Note that the newly designed $\boldsymbol{Q}^{(i)}$ is used in the same iteration i to generate new reconstructed vectors in preparation for the next iteration $i+1$. The main difference between this design and the CL design is that there is NO FEEDBACK; quantized prediction error is not fed back into the closed-loop system.

the resulting reconstruction is improved, and this results in better prediction. Under the reasonable and common assumption that smaller prediction errors lead to smaller quantization errors (and vice versa), and the assumption that given a training set the MSVQ design is optimal, we obtain monotonic improvement throughout the process.

Note that the entire design is in open-loop mode since we compute prediction errors for the entire sequence before quantization. As the distortion is generally decreasing, we expect the process to converge. At convergence, further iterations do not modify the quantizer

$$\boldsymbol{Q}^{(i+1)} = \boldsymbol{Q}^{(i)}, \tag{6}$$

which immediately ensures that the reconstruction sequence is fixed,

$$\hat{x}_n^{(i+1)} = \hat{x}_n^{(i)}, \tag{7}$$

and that the next-frame prediction sequence is fixed:

$$P[\hat{x}_{n-1}^{(i)}] = P[\hat{x}_{n-1}^{(i-1)}]. \tag{8}$$

This implies that the prediction would be unchanged if it were based on the reconstruction of the current iteration,

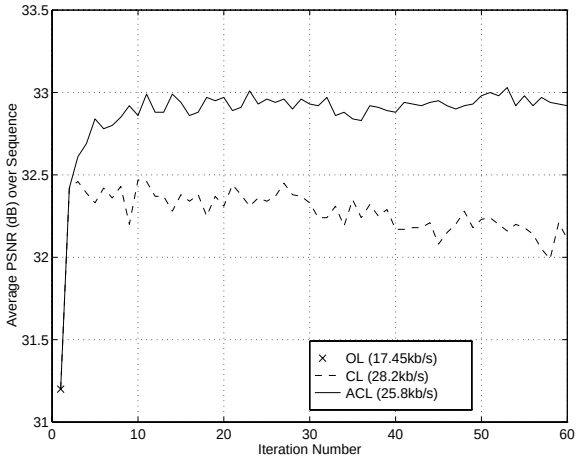

Figure 2: Performance comparison of standard CL design and the proposed ACL approach to EC-PMSVQ design. Average PSNR (in dB) over the training sequence "Carphone" is shown for the EC-PMSVQ available at the end of each iteration. Both techniques are initialized with the outcome of OL design.

instead of on the reconstruction from the previous iteration. In other words, the procedure is *asymptotically equivalent* to closed-loop design, but the algorithm is running at all times in open loop. The procedure is thus "open-loop" in nature, yet it converges to optimization of the closed-loop performance.

3. SIMULATION RESULTS

The proposed EC-PMSVQ design was tested in the context of low bit rate video coding. We implemented a video codec where 8×8 blocks of residuals are used as vectors. The video sequences are in QCIF format with a frame rate of 10 frames/sec. The system uses half-pixel motion compensation, and is basically a "bare-bones" H.263 scheme where the DCT/quantization module was replaced with the EC-PMSVQ, and each 8×8 block is considered as a separate macroblock. After the first frame, all frames are compressed in interframe mode.

Two main experiments have been performed. In the first experiment, a total of 20 frames (luminance component) of the sequence "Carphone" were used as both the training and testing sequence. We design an EC-PMSVQ using each of the OL, CL, and ACL techniques described in section 2. Fig. 2 compares the performance of EC-PMSVQ designed by the proposed ACL design method with that of the standard OL and CL designs. The PSNR shown is that of the actual closed-loop performance of the coder using an EC-PMSVQ obtained at each iteration (after convergence of each iteration) and is equal to the average PSNR over the training video sequence. Note that both systems start their iterations by using an OL-designed codebook, and thus have the same performance at the first iteration. Both systems improve performance in the first few iterations. However, the CL design algorithm leads to gradual accumulation of

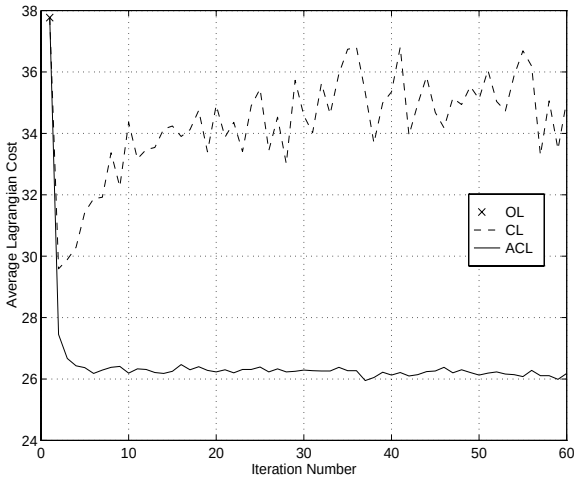

Figure 3: Performance comparison in terms of average rate-distortion Lagrangian cost on the *three independent test sequences*. Results are for the EC-PMSVQ available at the end of each iteration. Both techniques are initialized with the outcome of OL design. Note the stability and superiority of the ACL approach.

error in the system and causes a subsequent drop in overall PSNR. It is thus important to note the instability of the CL algorithm even within the training set. On the other hand, the proposed ACL approach shows persistent improvements, and eventually provides performance that is superior by about 0.8 dB. Moreover, the average bit rate of the closed-loop design is 28.2 kbits/s versus 25.8 kbits/s for ACL. Both coders used identical bit rates for all other side information (including motion vectors). The quantizer parameters for this test were as follows: $P = 3$ stages, $N = 32$ vectors per stage, and $M = 3$ candidates for the M-search.

We next present results demonstrating performance outside the training set. In this second experiment, in order for the EC-PMSVQ to be more statistically representative, we used a total of 13 video sequences in the training phase. Each video sequence is of length 20 frames. The test set is composed of the three independent (i.e., unused for training) video sequences, namely, "Salesman", "Claire", and "Akiyo", each also of 20 frames. In Fig. 3, we show the average rate-distortion Lagrangian cost incurred over the *test* sequences in normal closed-loop operating mode, at the end of each iteration of the design procedure. Note that the ACL algorithm improves with iterations, while the CL algorithm is very unstable. Notice in particular how the CL algorithm becomes unstable right after its first few iterations. These results are in terms of rate-distortion Lagrangian (rather than PSNR) as it allows meaningful averaging over several different input sequences with differing rate requirements. Additional results are shown in Table 1. For the comparison, we use the best-performing CL quantizer and show its performance relative to H.263 and the proposed ACL method. ACL provides gains over H.263 of up to 0.4 dB. The parameters used for the quantizer are as follows: $P = 3$ stages, $N = 64$ vectors per stage, and $M = 3$ survivors for the M-search.

Table 1: Performance comparison of H.263 with CL and ACL designs of EC-PMSVQ. The comparison is in terms of PSNR in dB, rate in kbits/s, and the rate-distortion Lagrangian cost per pixel.

Sequence	Coder	PSNR	Rate	$D + \lambda R$
Salesman	H.263	31.82	19.75	45.07
	CL (best)	31.61	20.08	47.23
	ACL	**32.22**	**19.66**	**41.33**
Claire	H.263	36.73	12.96	15.35
	CL (best)	36.21	16.78	17.54
	ACL	**36.80**	**12.92**	**15.11**
Akiyo	H.263	34.96	13.80	22.39
	CL (best)	34.68	15.77	23.99
	ACL	**35.13**	**13.82**	**21.59**

4. CONCLUSION

This paper describes a new approach to training predictive multi-stage vector quantizers, which does not suffer from the statistical mismatch typical of OL training algorithms, nor from the instability experienced in CL approaches. The proposed iterative algorithm is open-loop in nature but asymptotically optimizes the closed-loop system. Simulation results were presented for a simple EC-PMSVQ system for video coding, and showed the superiority of the proposed design algorithm over conventional approaches.

5. REFERENCES

[1] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *IEEE Trans. Computers*, vol. C-23, pp. 90-93, Jan. 1974.

[2] R. Neff and A. Zakhor, "Very low bit rate video coding based on matching pursuits," *IEEE Trans. Circuits and Sys. for Video Tech.*, vol. 7, pp. 158-171, Feb. 1997.

[3] A. Gersho and R, M. Gray, *Vector Quantization and Signal Compression*. Boston, MA: Kluwer, 1992.

[4] K. Rose, H. Khalil, and S. L. Regunathan, "Open-loop design of predictive vector quantizers for video coding," *Proc. IEEE Intl. Conf. on Image Processing*, ICIP'98, Chicago, IL, vol. 3, pp. 953-957, Oct. 1998.

[5] V. Cuperman and A. Gersho, "Vector predictive coding of speech at 16 kbits/s," *IEEE Trans. Communications*, vol. COM-33, no. 7, pp. 685-696, July 1985.

[6] C. F. Barnes and R. L. Frost, "Vector quantizers with direct sum codebooks," *IEEE Trans. Information Theory*, vol. 39, no. 2, pp. 565-580, Mar. 1993.

[7] H. Khalil and K. Rose, "A selective splitting approach to entropy-constrained single/multi-stage vector quantization design," *Image and Video Communications and Processing 2000,* San Jose, CA, Jan. 2000.