A New Segment Quantization Using Lempel-Ziv Algorithm and Its Application to Quantization of Line Spectral Frequencies

doi: 10.1109/TCOMM.2007.894090

A New Segment Quantization Using Lempel–Ziv Algorithm and Its Application to Quantization of Line Spectral Frequencies

Minoru Kohata, *Member, IEEE*, Motoyuki Suzuki, Akinori Ito, *Member, IEEE*, and Shozo Makino, *Member, IEEE*

***Abstract*—A new segment quantization method using the Lempel–Ziv algorithm is proposed, and it is applied to quantize line spectral frequency parameters in speech codec. The proposed segment quantizer can save four bits per frame, compared with the ITU-T G.729 speech codec (18 bits/frame), without degradation of subjective or objective speech quality.**

***Index Terms*—Lempel–Ziv (LZ) coding, segment quantization (SQ), speech coding, vector quantization (VQ).**

I. Introduction

A LOW-BIT-RATE speech coder can use only a small number of bits to represent speech, so it is quite difficult to preserve quality. In a low-bit-rate speech coder, line spectral frequencies (LSF) are often used to represent the spectral information which is extracted from every speech frame in 10–30 ms periods.

We can observe temporal correlation between the spectral parameters of the successive frames, since the short-time speech spectrum evolves slowly with time. Therefore, some quantizers named memory vector quantization (VQ), which use the temporal redundancy to reduce bitrate, were proposed [1]–[3]. As prevailing examples of such quantizers, there are finite-state VQ (FSVQ) [4], and predictive VQ (PVQ), which includes the autoregressive type (AR-PVQ) and moving-average type (MA-PVQ) [3].

Another simple but convenient spectrum quantization, which belongs to memory VQ, is the segment quantization (SQ) [5]–[7]. The SQ is a simple extension of VQ to the temporal domain, and the same codebook training algorithm as that for the VQ can be used, such as the generalized Lloyd (GL) algorithm [8]. But the rate-distortion characteristics of the SQ are impractical, because the quantization distortion cannot be decreased without an optimal segmentation algorithm. The difference between these two frameworks of the memory VQ, that is, between the PVQ and the SQ, is whether it exploits any parametric model or not. The advantage of exploiting the parametric model is that we can optimize the model by adjusting its parameters using statistical methods. However, the performance of the PVQ depends on which parametric model is adopted, and we cannot obtain good performance if the adopted model is not adequate for the source. The latter framework, the SQ, is inherently a nonparametric method, so it does not depend on any stochastic model. Then, we may obtain a better segment quantizer than PVQ, if we design the SQ codebook with a more sophisticated segmentation method and codebook training algorithm.

The method proposed in this letter reduces the interframe redundant information that exists in LSF vectors by using a new segmentation scheme [9], which is similar to the incremental parsing of the LZ78 [10] algorithm. By using this method, we can design an excellent segment quantizer for LSF, which can achieve lower quantization distortion at a lower bitrate than PVQ.

In the following, the proposed method is briefly explained, and then it is applied to a well-known speech codec, ITU-T Rec. G.729 CS-ACELP (Conjugate Structure Algebraic Code Excited Linear Prediction). Then, the speech quality synthesized by the G.729 codec's original quantizer and by the proposed quantizer is objectively and subjectively compared.

II. Lempel–Ziv Segment Quantiztion

The Lempel–Ziv (LZ) coding method [10], [11] is one of the universal coding algorithms that use a dictionary obtained by decomposing an input sequence. The original LZ coding methods can be applied to only a discrete information source; therefore, they cannot be directly applied to speech coding parameters such as LSF vectors.

First, x_n^m is defined as the subsequence from continuous information $(x_n^m \in R^{m-n+1})$, where n and m represent the start and end address of the subsequence, respectively, and this subsequence is termed a "segment." Then the jth subsequence is denoted as $x_{n(j)}^{n(j+1)-1}$, which comes from the jth decomposition.

Next, the condition of parsing a segment is defined as (1). Here, a threshold value TH, and distance measure, $\mathrm{dist}(\cdot)$ are introduced

$$\begin{aligned} &\text{if} \quad \mathrm{dist}\left(x_{n(j)}^{n(j+1)-2}, x_{n(r)}^{n(r+1)-1}\right) < \mathrm{TH} \\ &\text{then} \quad x_{n(j)}^{n(j+1)-1} \cong x_{n(r)}^{n(r+1)-1} x_{n(j+1)-1}^{n(j+1)-1} \\ &\text{else} \quad x_{n(j)}^{n(j+1)-1} = x_{n(j)}^{n(j+\mathrm{MINSEGLEN})}. \end{aligned} \tag{1}$$

As the distance measure, $\mathrm{dist}(\cdot)$, a weighted Euclidean distance is used in this letter. The same weighting coefficients as those used in ITU-T G.729 are used.

Equation (1) means that the segment $x_{n(j)}^{n(j+1)-1}$ is decomposed to $x_{n(r)}^{n(r+1)-1}$ combined with $x_{n(j+1)-1}^{n(j+1)-1}$, if the distance between $x_{n(j)}^{n(j+1)-2}$ and $x_{n(r)}^{n(r+1)-1}$ is lower than TH, where

Paper approved by C. S. Ravishankar, the Editor for Speech Processing of the IEEE Communications Society. Manuscript received March 12, 2004; revised December 27, 2005 and June 13, 2006. This paper was presented in part at the IEEE International Conference on Acoustics, Speech, and Signal Processing, Philadelphia, PA, 2005.

M. Kohata is with the Department of Information and Network Science, Chiba Institute of Technology, Narashino, Chiba 275-0016, Japan (e-mail: kohata@net.it-chiba.ac.jp).

M. Suzuki, A. Ito, and S. Makino are with the Graduate School of Engineering, Tohuko University, Sendai, Miyagi 980-8579, Japan.

Digital Object Identifier 10.1109/TCOMM.2007.894090

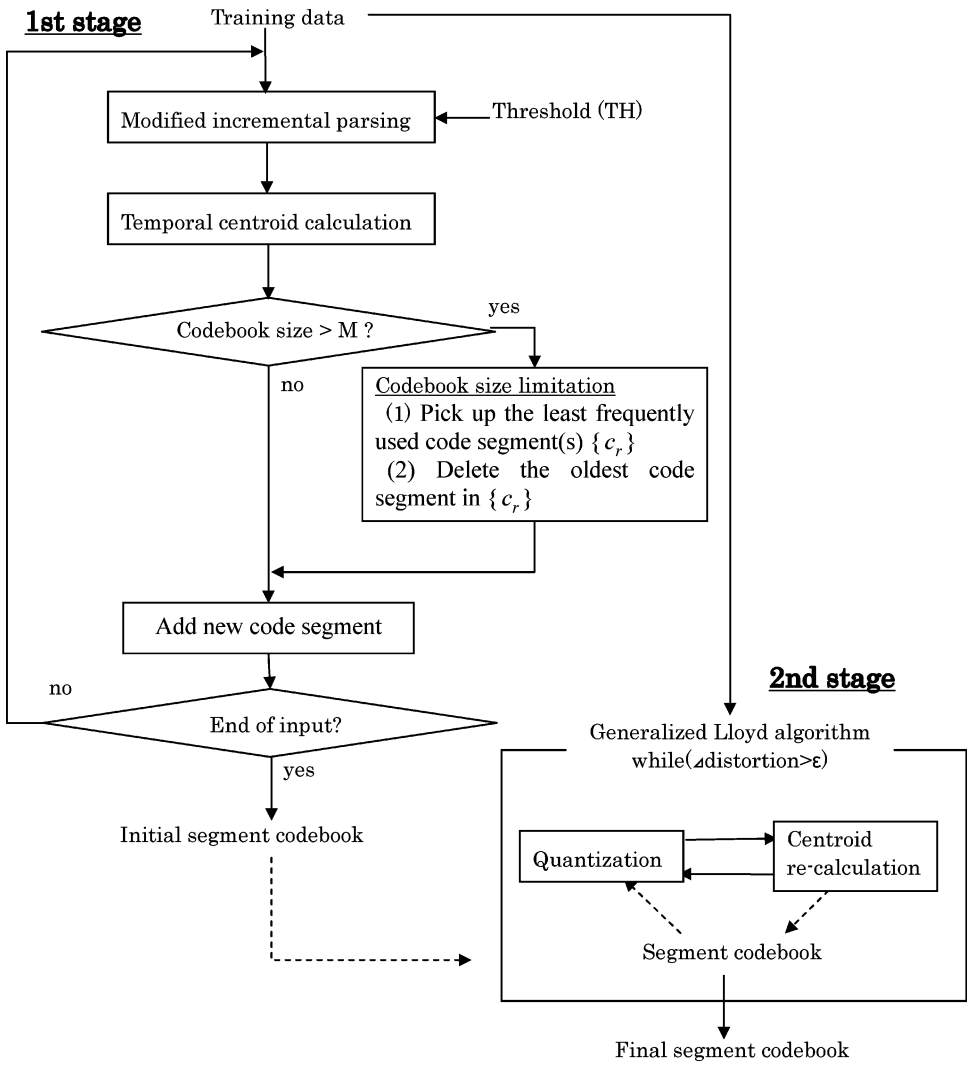

Fig. 1. Segmentation and training process.

$x_{n(r)}^{n(r+1)-1}$ is one of the segments in the current dictionary (codebook) nearest to the $x_{n(j)}^{n(j+1)-2}$. Otherwise, the segment $x_{n(j)}^{n(j+1)-1}$ is decomposed as a fixed length segment whose length is MINSEGLEN. Normally, MINSEGLEN is set to a small integer (≥ 1).

This lossy LZ segment quantizer using a threshold value has already been proposed and applied to applications such as image coding [13], [14]. But this simple extension of the LZ algorithm for a continuous information source has the following two major problems:

1) the additional information for the extra vector a_j attached to the index impedes the reduction of the total bit rate;
2) the distortion does not decrease meaningfully, because the obtained code segment in the dictionary is not close enough to the statistically optimal centroid.

These problems are likely due to using incremental parsing as in the original LZ algorithm. The second problem suggests that we need some kind of codebook training to represent the statistically optimal centroids.

Then, the original LZ algorithm is modified so that it can quantize a continuous information source with low bit and low quantization distortion. The proposed method uses incremental parsing only for segmentation, and then codebook training is carried out to build a fixed size codebook. That is, the incremental parsing is carried out only for building an initial codebook. This method is termed LZSQ. The segmentation and training process is shown in Fig. 1 and explained below.

1) In the original incremental parsing, the segments stored in the codebook are fixed and never changed. But for a continuous input source, it is preferable to modify the code segments somehow, because the code segment registered earlier than others dominates the following decomposition by (1). Then, the code segment is modified by calculating the temporal centroid of the segments. The temporal centroid s_r is calculated by (2)

$$s_r = \frac{c_r \cdot s_r + x_{n(j)}^{n(j+1)-2}}{c_r + 1} \quad (1 \leq r \leq M) \tag{2}$$

where M is the codebook size, and c_r is the number of segments that have satisfied (1) for s_r till the time $n(j)-1$.
2) The original incremental parsing does not limit the codebook size, but it must be limited in the new method, which

will carry out SQ in a canonical way with a fixed codebook obtained after the segmentation and training. The codebook size limitation procedure is as follows.

a) When the number of codes exceeds the limit, pick up the code segment(s) corresponding to the minimum count of c_r.
b) Among the code segment(s) picked up in a), delete the oldest code segment.
c) Add a new code segment to the codebook, in place of the deleted one.

This procedure ensures that the codebook size does not exceed the limit and that the more frequently used code segments survive.

3) The decomposed segments are used to train a codebook to make the distortion lower, stochastically. For this purpose, generally, quantization and centroid recalculation are carried out alternatively, as in the GL algorithm [8]. As shown in Fig. 1, at first, the initialization of the codebook is carried out by the segmentation method introduced in 1) and 2) (first stage), then quantization and centroid calculation by the GL algorithm are iterated till the distortion converges (second stage). In the centroid recalculation, weighted square error is used as a distance measure. For the SQ in the GL algorithm, a quantization criterion must be prepared, because the length of the code segment is variable. Then, the input vectors are quantized to the code segment in the "current" codebook, whose distortion per vector (frame) is minimum. As a result, the input vectors are segmented by the length of this code segment.

To decrease the quantization distortion would require an increase in codebook size, which means an exponential increase in the required training data. This problem is more imminent in SQ than in VQ, because a segment can be regarded as a high-dimensional vector. In order to solve this problem, we adopt Split-SQ. The Split-SQ coupled with the LZSQ (Split-LZSQ) can simply reduce the dimension of segment per one quantizer, thus reducing the total quantization distortion efficiently. In order to ensure the LSF order after the quantization, we applied the reordering procedure, which is similar to that used in the G.729 codec.

In the LZSQ, the longer code segments are used relatively infrequently. Then the Huffman code is applied to allocate binary code to the code segments to reduce the bit rate.

III. Performance Evaluation

Under the conditions shown in Table I, the performance of the Split-LZSQ for LSF quantization was evaluated, varying the threshold TH_k, and codebook size M_k, where k denotes the split segment number ($k = 1, 2, 3$). It would be best to fix the parameters, TH_k and M_k, theoretically, but we have no valid theoretical analysis of the Split-LZSQ for the LSF quantization, so these two parameters were sampled and combined, and the codebook training was carried out for each of them. If there are C_k combinations of (TH_k, M_k), which are applied to the Split-LZSQ quantizer for the kth split part, then the number of the total combinations becomes $C_1 \cdot C_2 \cdot C_3$.

The Split-LZSQ codebook is designed for G.729 speech codec, that is, the codebook training is carried out using the LSF data from G.729 coder. Fig. 2 shows the rate-distortion characteristics of the Split-LZSQ. In Fig. 2, only 300 points are plotted, which correspond to the conditions, where the product of distortion and rate are smaller than the other combinations. The rate and the distortion for the MA-PVQ, which is the LSF quantizer of G.729 codec, are also shown in this figure. This result shows that the Split-LZSQ can outperform the MA-PVQ of G.729 by about 4 b/frame, at a spectral distortion of 1.3 dB. It is reported that the theoretical lower boundary of the rate of a first-order AR-PVQ is around 16 b/frame if the spectral distortion is kept within 1.0 dB [12]. Thus, the result for the Split-LZSQ of 14 b/frame is close to the theoretical limit of a first-order AR-PVQ, though the result for the Split-LZSQ is an experimental result. In Fig. 2, the rate and the distortion of the multistage VQ (a memoryless VQ) of the mixed excitation linear prediction (MELP) codec [15] are also shown for reference. The bitrate for the Split-LZSQ is an average value, because it varies with quantized segment length. Then, the maximum and the minimum bitrate were measured. As a result, the maximum bitrate was 23.5 b/frame, and the minimum bitrate was 6.2 b/frame. This result shows the maximum bitrate is not so high, and the percentage of the frames whose rate is more than 20 b/frame is less than 2%. Next, the outlier characteristic of the Split-LZSQ is measured. Fig. 3 shows the outlier, the percentage of the quantized LSF frames whose

TABLE I
Experimental Condition for Split-LZSQ

Sampling frequency		8 kHz
LSF order		10 (3:3:4)
Frame period		10 ms
LPC window size		30 ms
Speech data	training	2502 s
(ASJ speech database)	test	985 s
Codebook size M		2^{13}, 2^{14}, 2^{15}, 2^{16}
TH		0.005 - 0.035
MINSEGLEN		2
Input buffer size		10 frames

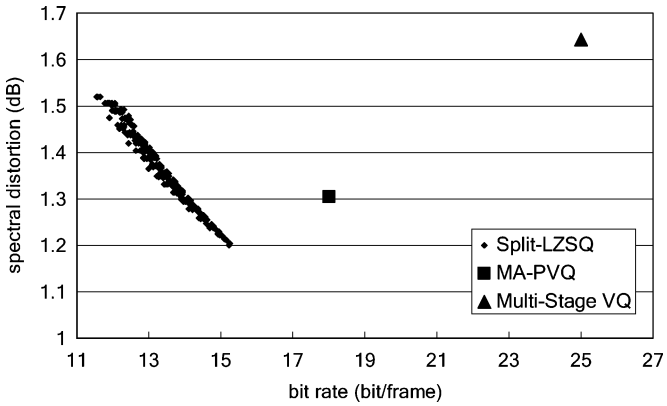

Fig. 2. Rate-distortion characteristics of Split-LZSQ.

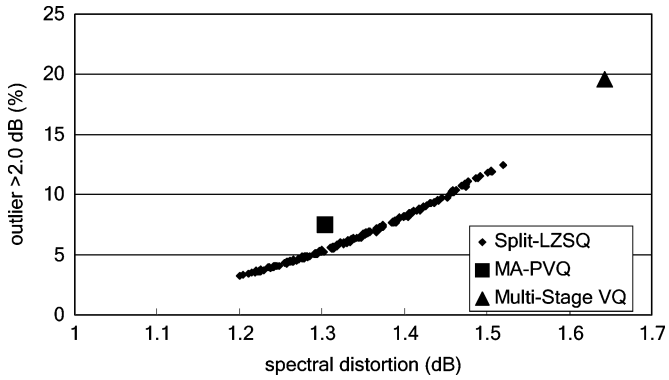

Fig. 3. Outlier characteristic of split-LZSQ

spectral distortions are higher than 2.0 dB. In Fig. 3, the outlier of the MA-PVQ is also shown. We can confirm that the outlier of the Split-LZSQ is lower than that of the MA-PVQ, at the same spectral distortion.

The quantization in the Split-LZSQ does not depend on the past quantization result, so error propagation does not occur between quantized segments; the effect of the error is confined within a segment. As for the algorithmic delay, it depends on the length of the input buffer. In order to observe temporal redundancy, a long input buffer is preferable, but it will cause a long algorithmic delay. In this experimental condition, the mean algorithmic delay was 9.7 frames.

As the final evaluation of the proposed method, we carried out subjective listening tests for the synthesized speech by the Split-LZSQ. The speech was synthesized by substituting the original LSF quantizer in G.729 codec with the Split-LZSQ, and then it was compared with synthesized speech by G.729 codec. Regarding the type of subjective test, the $A-B$ comparison test was adopted, where the Split-LZSQ was set to make the same spectral distortion as that of G.729 ($\cong$1.3 dB). The subjective tests were carried out using ten listeners with a data set which included eight Japanese sentences from the ASJ database. This listening test was not "conversational," so the effect of the algorithmic delay of the Split-LZSQ on the subjective quality was not measured. The performance of Split-LZSQ was evaluated under an error-free channel, therefore the robustness against channel errors was not evaluated in this letter.

As a result, the subjective scores were 47.5% for G.729, and 52.5% for the Split-LZSQ. As for the hypothesis testing result, the null hypothesis, "H_0: the speech qualities by the Split-LZSQ (14 b/frame) are equivalent to those by G.729 (18 b/frame)," cannot be rejected at the 95% confidence level. We were afraid of the possibility of subjective quality degradation caused by the segmentwise quantization in the Split-LZSQ, which might not be observed through objective measures such as spectral distortion. But this result shows that there is no degradation in the Split-LZSQ, compared with MA-PVQ operating at the same spectral distortion.

IV. Conclusion

We have proposed a new segment quantizer for LSF quantization based upon the incremental parsing of the LZ algorithm followed by a codebook training algorithm. Finally, a quantizer named Split-LZSQ can quantize LSF with the same spectral distortion, and the same subjective quality as G.729 codec, saving 4 b/frame.

References

[1] P. C. Chang and R. M. Gray, ""Gradient algorithms for designing predictive vector quantizers," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-34, no. 4, pp. 679–690, Jul. 1986.

[2] T. Eriksson, J. Linden, and J. Skoglund, "Interframe LSF quantization for noisy channels," *IEEE Trans. Speech Audio Process.*, vol. 7, no. 5, pp. 495–509, Sep. 1999.

[3] H. Ohmuro, T. Moriya, K. Mano, and S. Miki, "Coding of LSP parameters using interframe moving average prediction and multi-stage vector quantization," in *Proc. IEEE Workshop Speech Coding Telecommun.*, 1993, vol. 1, pp. 63–64.

[4] Y. Hussain and N. Farvardin, "Finite-state vector quantization over noisy channels and its application to LSP parameters," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 1992, pp. 133–136.

[5] D. Y. Wong, B. H. Yuang, and D. Y. Cheng, "Very low data rate speech compression with LPC vocoder and matrix quantization," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Apr. 1983, pp. 65–68.

[6] C. Tsao and R. M. Gray, "Matrix quantizer design for LPC speech using the generalized Lloyd algorithm," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-33, no. 3, pp. 537–545, Jun. 1985.

[7] S. Roucos, R. Schwartz, and J. Makhoul, "Segment quantization for very-low-rate speech coding," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, May 1982, pp. 1565–1570.

[8] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. Commun.*, vol. COM-28, no. 1, pp. 84–95, Jan. 1980.

[9] M. Kohata, I. Mitsuya, M. Suzuki, and S. Makino, "Efficient segment quantization of LSP coefficients for very low bit speech coding," in *Proc. Int. Conf. Spoken Lang. Process.*, 2000, vol. 3, pp. 826–829.

[10] J. Ziv and A. Lempel, "Compression of individual sequences via variable-rate coding," *IEEE Trans. Inf. Theory*, vol. IT-24, no. 5, pp. 530–536, Sep. 1978.

[11] J. Ziv and A. Lempel, "Coding theorems for individual sequences," *IEEE Trans. Inf. Theory*, vol. IT-24, no. 4, pp. 405–412, Jul. 1978.

[12] J. Samuelsson and P. Hedelin, "Recursive coding of spectrum parameters," *IEEE Trans. Speech Audio Process.*, vol. 9, no. 5, pp. 492–503, Oct. 2001.

[13] W. A. Finamore and M. A. Lesiter, "Lossy Lempel-Ziv algorithm for large alphabet sources and applications to image compression," in *Proc. Int. Conf. Image Process.*, 1996, vol. 1, pp. 225–228.

[14] C. Constantinescu and J. A. Storer, "On-line adaptive vector quantization with variable size codebook entries," in *Proc. Data Compress. Conf.*, 1993, vol. 1, pp. 32–41.

[15] A. V. McCree and J. C. DeMartin, "A 1.7 kb/s MELP coder with improved analysis and quantization," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 1998, pp. 593–596.